##### 

##### **1. Project Overview**

This document contains the code, browser website URLs, and database details for the Spring Boot application (springIOC1) developed for product catalog management. The application uses Thymeleaf for templates, H2 as an in-memory database, and Spring Data JPA for persistence. Screenshots of the browser interface and H2 database console are included.

##### **2. Code Details**

**a. application.properties**

* Location: src/main/resources/application.properties

spring.application.name=springIOC1

spring.datasource.url=jdbc:h2:mem:productdb;DB\_CLOSE\_ON\_EXIT=FALSE

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

spring.jpa.hibernate.ddl-auto=update

spring.thymeleaf.prefix=classpath:/templates/

spring.thymeleaf.suffix=.html

spring.thymeleaf.enabled=true

logging.level.org.springframework.boot.autoconfigure.jdbc=DEBUG

**b. ProductController.java**

* Location: src/main/java/com/SpringIOC/demo/controller/ProductController.java

package com.SpringIOC.demo.controller;

import com.SpringIOC.demo.entity.Product;

import com.SpringIOC.demo.service.ProductService;

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.ModelAttribute;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestParam;

@Controller

public class ProductController {

@Autowired

private ProductService productService;

@GetMapping("/")

public String showHomePage() {

return "index";

}

@GetMapping("/add-product")

public String showAddProductForm(Model model) {

model.addAttribute("product", new Product());

return "add-product";

}

@PostMapping("/add-product")

public String addProduct(@ModelAttribute Product product) {

productService.saveProduct(product);

return "redirect:/";

}

@GetMapping("/display-products")

public String displayProducts(@RequestParam(required = false) String category, Model model) {

if (category != null && !category.isEmpty()) {

model.addAttribute("products", productService.getProductsByCategory(category));

} else {

model.addAttribute("products", productService.getAllProducts());

}

return "display-products";

}

}

**c. Templates**

* Location: src/main/resources/templates/
* Files:
  + index,html:

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<title>Product Catalog</title>

<style>

body { font-family: Arial, sans-serif; text-align: center; }

.container { margin-top: 50px; }

.button {

display: inline-block;

padding: 10px 20px;

margin: 10px;

background-color: #4CAF50;

color: white;

text-decoration: none;

border-radius: 5px;

}

.button:hover { background-color: #45a049; }

</style>

</head>

<body>

<div class="container">

<h1>Product Catalog System</h1>

<a href="/add-product" class="button">Add Product</a>

<a href="/display-products" class="button">Display Products</a>

</div>

</body>

</html>

add-product.html:

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<title>Add Product</title>

<style>

body { font-family: Arial, sans-serif; }

.container { width: 50%; margin: 50px auto; }

.form-group { margin-bottom: 15px; }

label { display: block; margin-bottom: 5px; }

input[type="text"], input[type="number"] {

width: 100%;

padding: 8px;

border: 1px solid #ccc;

border-radius: 4px;

}

button {

padding: 10px 20px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

}

button:hover { background-color: #45a049; }

</style>

</head>

<body>

<div class="container">

<h1>Add New Product</h1>

<form th:action="@{/add-product}" th:object="${product}" method="post">

<div class="form-group">

<label for="name">Product Name:</label>

<input type="text" id="name" th:field="\*{name}" required>

</div>

<div class="form-group">

<label for="price">Price:</label>

<input type="number" id="price" th:field="\*{price}" step="0.01" required>

</div>

<div class="form-group">

<label for="category">Category:</label>

<input type="text" id="category" th:field="\*{category}" required>

</div>

<button type="submit">Submit</button>

</form>

<a href="/">Back to Home</a>

</div>

</body>

</html>

display-products.html:

:<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<title>Display Products</title>

<style>

body { font-family: Arial, sans-serif; }

.container { width: 80%; margin: 50px auto; }

table { width: 100%; border-collapse: collapse; margin-top: 20px; }

th, td { border: 1px solid #ddd; padding: 8px; text-align: left; }

th { background-color: #4CAF50; color: white; }

tr:nth-child(even) { background-color: #f2f2f2; }

.filter-form { margin-bottom: 20px; }

input[type="text"] { padding: 8px; }

button {

padding: 8px 15px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

}

button:hover { background-color: #45a049; }

</style>

</head>

<body>

<div class="container">

<h1>Product Catalog</h1>

<form class="filter-form" action="/display-products" method="get">

<label for="category">Filter by Category:</label>

<input type="text" id="category" name="category" placeholder="Enter category">

<button type="submit">Filter</button>

</form>

<table>

<thead>

<tr>

<th>ID</th>

<th>Name</th>

<th>Price</th>

<th>Category</th>

</tr>

</thead>

<tbody>

<tr th:each="product : ${products}">

<td th:text="${product.id}"></td>

<td th:text="${product.name}"></td>

<td th:text="${product.price}"></td>

<td th:text="${product.category}"></td>

</tr>

<tr th:if="${products.isEmpty()}">

<td colspan="4">No products found</td>

</tr>

</tbody>

</table>

<a href="/">Back to Home</a>

</div>

</body>

</html>

##### **3. Browser Website URL**

* **Home Page**: http://localhost:8080/
  + **Screenshot**: [Insert Screenshot of http://localhost:8080/ here]
    - Description: Shows the "Product Catalog System" page with "Add Product" and "Display Products" buttons.
    - (Paste the second screenshot you provided here, showing the home page with the URL http://localhost:8080/)
* **Add Product Page**: http://localhost:8080/add-product
  + **Screenshot**: [Insert Screenshot of http://localhost:8080/add-product here]
    - Description: Displays the "Add New Product" form with fields for Product Name, Price, Category, and Submit button.
    - (Paste the first screenshot you provided here, showing the URL http://localhost:8080/add-product)
* **Display Products Page**: http://localhost:8080/display-products
  + **Screenshot**: [Insert Screenshot of http://localhost:8080/display-products here]
    - Description: Shows the product catalog table with columns ID, Name, Price, and Category, including filter option.
    - (Paste the third screenshot you provided here, showing the URL http://localhost:8080/display-products)

##### 

##### **4. Database Details**

* **Database Type**: H2 In-Memory Database
* **JDBC URL**: jdbc:h2:mem:productdb
* **Username**: sa
* **Password**: (empty)
* **H2 Console Access**: http://localhost:8080/h2-console
* **Action**:
  + Start the application.
  + Open http://localhost:8080/h2-console in the browser.
  + Use the JDBC URL jdbc:h2:mem:productdb, username sa, and leave the password blank.
  + Click "Connect" and take a screenshot of the H2 console showing the database schema (e.g., the PRODUCT table with columns id, name, price, category).
  + **Screenshot**: [Insert Screenshot of H2 Console here]
    - Description: Displays the H2 database console with the PRODUCT table data.
    - (You’ll need to add this screenshot manually by accessing the H2 console and capturing it.)

##### **5. Troubleshooting Notes**

* **Whitelabel Error Resolution**: The Whitelabel Error was resolved by ensuring template files (index.html, add-product.html, display-products.html) are in src/main/resources/templates/ and match the controller return values. The template name mismatch (e.g., addproduct.html to add-product.html) was corrected.
* **H2 Shutdown Warning**: The ;DB\_CLOSE\_ON\_EXIT=FALSE in spring.datasource.url should resolve the JdbcSQLNonTransientConnectionException. If the warning persists, rebuild the project (mvn clean install) and check the shutdown logs.
* **Steps to Run**:
  1. Open the project in IntelliJ IDEA.
  2. Build the project: mvn clean install.
  3. Run the application: Right-click SpringIoc1Application.java and select "Run".
  4. Test the URLs and access the H2 console.

