Code Documentation

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http:/www.thymeleaf.org">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Insert title here</title>
</head>
<body>

<h2>Admin Login</h2>
<form action="#" th:action="@{/login}" th:object="${user}" method="post">
   <p>Name: <input type="text" th:field="*{name}" /></p>
   <p>Passwort: <input type="text" th:field="*{password}" /></p>
   <p><input type="submit" th:name="action" value="Submit" /></p>
</form>
<div style="color: red" th:if="${error != null}" th:text="${error}"></div>

</body>
</html>

<!DOCTYPE html>
<html lang="en" xmlns:th="http:/www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<h1>Welcome to Sporty Shoes</h1>
<h2>Successfully logged in <span th:text="${user.name}"></span></h2>
<h2>Change Password</h2>
<form action="#" th:action="@{/changePW}" th:object="${user}" method="post">
   <p>Name: <input type="text" th:field="*{name}" /></p>
   <p>Passwort: <input type="text" th:field="*{password}" /></p>
   <p><input type="submit" th:name="action" value="Submit" /></p>
</form>

<div style="color: red" th:if="${message != null}" th:text="${message}"></div>

<h2>List Users</h2>
<form action="#" th:action="@{/listUsers}" method="post">
   <p><input type="submit" th:name="action" value="List Users" /></p>
</form>

<h2>Search User</h2>
<form action="#" th:action="@{/searchUser}" th:object="${user}" method="post">
      <p>Name: <input type="text" th:field="*{searchName}" /></p>
   <p><input type="submit" th:name="action" value="Search User" /></p>
</form>

<a th:href="@{manageProducts}">Manage Products</a></br>
<a th:href="@{Purchase}">Purchases</a>

</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http:/www.thymeleaf.org">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Insert title here</title>
</head>
<body>

<h1>List Purchases</h1>

<table>
      <tr>
      <th>User</th>
      <th>Product</th>
      <th>Category</th>
      <th>Timestamp</th>
  </tr>
      <tr th:each="purchase :${purchases}">
            <td><span th:text="${purchase.user}"></span></td>
            <td><span th:text="${purchase.productName}"></span></td>
            <td><span th:text="${purchase.productCategory}"></span></td>
            <td><span th:text="${purchase.timestamp}"></span></td>
      </tr>

</table>
</body>
</html>

<!DOCTYPE html>
<html lang="en" xmlns:th="http:/www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<h1>List of Users</h1>
<table>
      <tr th:each="user :${users}">
            <td><span th:text="${user.name}"></span></td>
      </tr>

</table>


</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http:/www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>

<h2>Add Product</h2>
<form action="#" th:action="@{/addProduct}" th:object="${product}"
method="post">
   <p>Name: <input type="text" th:field="*{name}" /></p>
   <p>Category: <input type="text" th:field="*{category}" /></p>
   <p><input type="submit" th:name="action" value="Add Product" /></p>
</form>

<div style="color: red" th:if="${message != null}" th:text="${message}"></div>

<h2>Delete Product</h2>
<form action="#" th:action="@{/delProduct}" th:object="${product}"
method="post">
   <p>Name: <input type="text" th:field="*{name}" /></p>
   <p>Category: <input type="text" th:field="*{category}" /></p>
   <p><input type="submit" th:name="action" value="Delete Product" /></p>
</form>

<div style="color: red" th:if="${message2 != null}" th:text="${message2}"></div>

<a th:href="@{manageProducts}">List Products</a>
<table>
      <tr>
      <th>Product</th>
      <th>Category</th>
  </tr>
      <tr th:each="product :${products}">
            <td><span th:text="${product.name}"></span></td>
            <td><span th:text="${product.category}"></span></td>
      </tr>

</table>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http:/www.thymeleaf.org">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Insert title here</title>
</head>
<body>

<h1>Search Purchases</h1>

<form action="#" data-th-action="@{/PurchaseByCategory}" data-th-object="${categories}" method="post">
      <select id="category" name="category">
      <option value="">Select Category</option>
      <option th:each="category : ${categories}" th:value="${category}" th:text="${category}"></option>
      </select>
    <button type="submit" name="action" value="searchByCategory">Search Purchases by Category</button>
</form>

<form action="#" data-th-action="@{/PurchaseByDate}" data-th-object="${timestamp}" method="post">
      <select id="timestamp" name="timestamp">
      <option value="">Select Date</option>
      <option th:each="ts : ${timestamp}" th:value="${ts}" th:text="${ts}"></option>
      </select>
    <button type="submit" name="action" value="timestamp">Search Purchases by Date</button>
</form>

</body>
</html>

<!DOCTYPE html>
<html lang="en" xmlns:th="http:/www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<h1>Search for User</h1>

<div style="color: red" th:if="${userFound != null}" th:text="${userFound}"></div>

</body>
</html>
```

```java
package com.simplilearn.Phase3_Spring.model;

public class Product {

	private int id;
	private String name;
	private String category;

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getCategory() {
		return category;
	}
	public void setCategory(String category) {
		this.category = category;
	}


}
package com.simplilearn.Phase3_Spring.model;

public class User {

	public String name;
	public String password;
	public String searchName;

	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getPassword() {
		return password;
	}
	public void setPassword(String password) {
		this.password = password;
	}
	public String getSearchName() {
		return searchName;
	}
	public void setSearchName(String searchName) {
		this.searchName = searchName;
	}

}
```

```java
package com.simplilearn.Phase3_Spring.model;

import java.sql.Date;

public class Purchase {

	private String id;
	private String user;
	private String productName;
	private String productCategory;
	private Date timestamp;

	public String getId() {
		return id;
	}
	public void setId(String id) {
		this.id = id;
	}
	public String getUser() {
		return user;
	}
	public void setUser(String user) {
		this.user = user;
	}
	public String getProductName() {
		return productName;
	}
	public void setProductName(String productName) {
		this.productName = productName;
	}
	public String getProductCategory() {
		return productCategory;
	}
	public void setProductCategory(String productCategory) {
		this.productCategory = productCategory;
	}
	public Date getTimestamp() {
		return timestamp;
	}
	public void setTimestamp(Date timestamp) {
		this.timestamp = timestamp;
	}
}
```

```java
package com.simplilearn.Phase3_Spring.dao;

import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.PreparedStatementSetter;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;
import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
import org.springframework.jdbc.core.namedparam.SqlParameterSource;
import org.springframework.stereotype.Repository;

import com.simplilearn.Phase3_Spring.model.Product;
import com.simplilearn.Phase3_Spring.model.Purchase;
import com.simplilearn.Phase3_Spring.model.User;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

@Repository
public class DAO {

	@Autowired
	NamedParameterJdbcTemplate NPjdbcTemplate;

	@Autowired
	JdbcTemplate jdbcTemplate;

	public String authenticate(String user, String password) {
		SqlParameterSource namedParameters = new
MapSqlParameterSource().addValue("user", user);
		return NPjdbcTemplate.queryForObject("select password from user
where name=:user", namedParameters, String.class);

	}

	public int changePw(String user, String password) {
		int result = jdbcTemplate.update("update user set password = ? where
name = ?", password, user);
		return result;

	}

	public List<User> getUsers(){
		return jdbcTemplate.query("select * from user", new RowMapper<User>
() {

			@Override
			public User mapRow(ResultSet rs, int rowNum) throws
SQLException {
				User u = new User();
				u.setName(rs.getString(1));
				return u;
			}
		});
	}

	public String searchUser(String user) {
		SqlParameterSource namedParameters = new
MapSqlParameterSource().addValue("user", user);
```

```
            String us = null;
            try {
                  us = NPjdbcTemplate.queryForObject("select name from user
where name=:user", namedParameters, String.class);
            }catch(Exception ex){
                  System.out.println("Empty Resultset");

            }

            if(us != null && !us.isEmpty() && us.equals(user) ){
                  return us;
            }else {
                  return "not found";
            }
      }

      public List<Product> getProducts(){

            return jdbcTemplate.query("select * from product", new
RowMapper<Product> () {

                  @Override
                  public Product mapRow(ResultSet rs, int rowNum) throws
SQLException {
                        Product p = new Product();
                        p.setId(rs.getInt(1));
                        p.setName(rs.getString(2));
                        p.setCategory(rs.getString(3));
                        return p;
                  }
            });
      }

      public int setProduct(String name, String category) {

            int result = jdbcTemplate.update("insert into product (name,
category) values(?, ?)", name, category);
            return result;

      }

      public int delProduct(String name, String category) {

            int result = jdbcTemplate.update("delete from product where  name =?
and category=?", name, category);
            return result;

      }


      public List<Purchase> searchAllPurchases(){
            return jdbcTemplate.query("select u.name, p.name, p.category, pu.ts
from purchase pu left join (user u, product p) on (pu.name_id = u.id and
pu.product_id = p.id)" ,
                        new RowMapper<Purchase> () {
                  @Override
                  public Purchase mapRow(ResultSet rs, int rowNum) throws
SQLException {
                        Purchase pu = new Purchase();
                        pu.setUser(rs.getString(1));
                        pu.setProductName(rs.getString(2));
                        pu.setProductCategory(rs.getString(3));
```

```
                        pu.setTimestamp(rs.getDate(4));
                        return pu;
                    }

            });

      }

      public List<Purchase> searchPurchasesByDate(String date){
            return jdbcTemplate.query("select u.name, p.name, p.category, pu.ts
from purchase pu left join (user u, product p) on (pu.name_id = u.id and
pu.product_id = p.id) where Date(pu.ts)=?" ,
                        new PreparedStatementSetter() {

                    public void setValues(PreparedStatement preparedStatement)
throws SQLException {
                        preparedStatement.setString(1, date);
                    }},
                    new RowMapper<Purchase> () {

                        @Override
                    public Purchase mapRow(ResultSet rs, int rowNum) throws
SQLException {
                        Purchase pu = new Purchase();
                        pu.setUser(rs.getString(1));
                        pu.setProductName(rs.getString(2));
                        pu.setProductCategory(rs.getString(3));
                        pu.setTimestamp(rs.getDate(4));
                        return pu;
                    }

            });

      }

      public List<Purchase> searchPurchasesByCategory(String category){
            return jdbcTemplate.query("select u.name, p.name, p.category, pu.ts
from purchase pu left join (user u, product p) on (pu.name_id = u.id and
pu.product_id = p.id) where p.category=?" ,
                        new PreparedStatementSetter() {

                    public void setValues(PreparedStatement preparedStatement)
throws SQLException {
                        preparedStatement.setString(1, category);
                    }},
                    new RowMapper<Purchase> () {

                        @Override
                    public Purchase mapRow(ResultSet rs, int rowNum) throws
SQLException {
                        Purchase pu = new Purchase();
                        pu.setUser(rs.getString(1));
                        pu.setProductName(rs.getString(2));
                        pu.setProductCategory(rs.getString(3));
                        pu.setTimestamp(rs.getDate(4));
                        return pu;
                    }

            });

      }
}
```

```java
package com.simplilearn.Phase3_Spring.controller;

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 com.simplilearn.Phase3_Spring.dao.DAO;
import com.simplilearn.Phase3_Spring.model.User;

@Controller
public class Authentication {

	@Autowired
	DAO dao;

	@GetMapping("/")
	public String welcome(Model model) {
		model.addAttribute("user", new User());
	    return "index";
	}

	@PostMapping("/login")
	public String login(@ModelAttribute("user") User user, Model model) {

		String us = user.getName();
		String pw = user.getPassword();

		String resultpw = dao.authenticate(us, pw);

		model.addAttribute("user", user);

		if(pw.equals(resultpw)) {
			return "welcome";
		}else {
			model.addAttribute("error", "Password invalid");
			return "index";
		}


	}

	@PostMapping("/changePW")
	public String changePW(@ModelAttribute("user") User user, Model model) {

		String us = user.getName();
		String pw = user.getPassword();

		int results = dao.changePw(us, pw);
		System.out.println("Results: " + results);

		if(results == 1) {
			model.addAttribute("message", "Password updated");
		}else {
			model.addAttribute("message", "Password not updated");
		}

		model.addAttribute("user", user);

		return "welcome";
```

```
	}

}


package com.simplilearn.Phase3_Spring.controller;

import java.util.List;

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 com.simplilearn.Phase3_Spring.dao.DAO;
import com.simplilearn.Phase3_Spring.model.Product;
import com.simplilearn.Phase3_Spring.model.User;

@Controller
public class Products {

	@Autowired
	DAO dao;

	@GetMapping("/manageProducts")
	public String listUser(Model model) {

		List<Product> products = dao.getProducts();
		model.addAttribute("products", products);
		model.addAttribute("product", new Product());

		return "Products";
	}

	@PostMapping("/addProduct")
	public String addProduct(@ModelAttribute("product") Product product, Model
model) {

		String name = product.getName();
		String category = product.getCategory();

		int affected = dao.setProduct(name, category);

		if(affected == 1) {
			model.addAttribute("message", "Product added");
		}else {
			model.addAttribute("message", "Product not added");
		}

		return "Products";
	}

	@PostMapping("/delProduct")
	public String delProduct(@ModelAttribute("product") Product product, Model
model) {

		String name = product.getName();
		String category = product.getCategory();

		int affected = dao.delProduct(name, category);
```

```
		if(affected == 1) {
			model.addAttribute("message2", "Product deleted");
		}else {
			model.addAttribute("message2", "Product not deleted");
		}

		return "Products";
	}


}
```

```
package com.simplilearn.Phase3_Spring.controller;

import java.util.ArrayList;
import java.util.List;

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 com.simplilearn.Phase3_Spring.dao.DAO;
import com.simplilearn.Phase3_Spring.model.Purchase;
import com.simplilearn.Phase3_Spring.model.User;

@Controller
public class SearchPurchase {

	@Autowired
	DAO dao;

	@GetMapping("/Purchase")
	public String listPurchase(Model model) {

		List<Purchase> purchases = dao.searchAllPurchases();

		List<String> productCategory = new ArrayList<String>();
		List<String> timestamp = new ArrayList<String>();

		for(Purchase p : purchases){

			if(!productCategory.contains(p.getProductCategory())) {
				productCategory.add(p.getProductCategory());
			}

			String date = p.getTimestamp().toString();

			if(!timestamp.contains(date)) {

				timestamp.add(date);
			}
		}

		model.addAttribute("categories", productCategory);
		model.addAttribute("timestamp", timestamp);

		return "Purchases";
	}
```

```
	@PostMapping("/PurchaseByDate")
	public String PurchaseByDate(@ModelAttribute("timestamp") String ts, Model
model) {

		List<Purchase> purchases = dao.searchPurchasesByDate(ts);

		model.addAttribute("purchases", purchases);

		return "ListPurchases";
	}

	@PostMapping("/PurchaseByCategory")
	public String PurchaseByCategory(@ModelAttribute("category") String
category, Model model) {

		List<Purchase> purchases = dao.searchPurchasesByCategory(category);

		model.addAttribute("purchases", purchases);

		return "ListPurchases";
	}
}


package com.simplilearn.Phase3_Spring.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

import com.simplilearn.Phase3_Spring.dao.DAO;
import com.simplilearn.Phase3_Spring.model.User;


@Controller
public class SearchUsers {

	@Autowired
	DAO dao;

	@PostMapping("/listUsers")
	public String listUser(Model model) {
		List<User> users =  dao.getUsers();
		model.addAttribute("users",users);

		return "listUser";
	}


	@PostMapping("/searchUser")
	public String searchUser(@ModelAttribute("user") User user, Model model) {
		String userName = user.getSearchName();
		//System.out.println("Username: " + userName);
		String u =  dao.searchUser(userName);

		if(u.equals(userName)) {
```

```
                  model.addAttribute("userFound", "User " + user.getSearchName()
+" found");
            }else {
                  model.addAttribute("userFound", "User " + user.getSearchName()
+" not found");
            }
            //model.addAttribute("user",user);
            //System.out.println("Searchuser: " + user.getName());
            //System.out.println("userName: " + user.getSearchName());

            return "searchUser";
      }
}
```