

Developing Spring Boot Web MVC Crud Application

USE CASE:

Product Management System

[Create New Product](#)

Id	Name	Brand	Price	Actions
1	Laptop	Dell	55000.0	Update Delete
2	Mobile	Apple	65000.0	Update Delete

Product Management System

[HOME](#)

Create New Product

Product Name

Product Brand

Product Price

[Save](#)

Product Management System

[HOME](#)

Edit Product

Product Id





























Product Name

Product Brand

Product Brand

[Update](#)

Example:

- ✖  SpringBootMvc [boot] [devtools]
 - ✖  src/main/java
 - ✖  com.seshusoft
 - >  SpringBootApplication.java
 - ✖  com.seshusoft.controller
 - >  AppController.java
 - ✖  com.seshusoft.model
 - >  Product.java
 - ✖  com.seshusoft.repo
 - >  ProductRepository.java
 - ✖  com.seshusoft.service
 - >  ProductService.java
 - ✖  src/main/resources
 -  static
 - ✖  templates
 -  edit_product.html
 -  index.html
 -  new_product.html
 -  application.properties
 - >  src/test/java
 - >  JRE System Library [JavaSE-11]
 - >  Maven Dependencies
 - >  src
 -  target
 -  HELP.md
 -  mvnw
 -  mvnw.cmd
 -  pom.xml

Step 1: Create database in mysql server.

```
use adiDB;

DROP TABLE product;

CREATE TABLE product (
  id int NOT NULL AUTO_INCREMENT,
  name varchar(20) NOT NULL,
  brand varchar(45) NOT NULL,
  price float NOT NULL,
  PRIMARY KEY (id)
);

INSERT INTO product(name,brand,price) VALUES ('Laptop','Dell',55000.00);
INSERT INTO product(name,brand,price) VALUES ('Mobile','Apple',65000.00);

SELECT *FROM product;
```

Step 2: Create Spring Starter Project with following dependencies
(Spring Web, Spring Data JPA, MySQL Driver, Thymeleaf, DevTools)

pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-
4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.5.3</version>
    <relativePath /> <!-- lookup parent from repository -->
  </parent>
  <groupId>com.example</groupId>
  <artifactId>SpringBootMvc</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>SpringBootMvc</name>
  <description>Demo project for Spring Boot</description>
  <properties>
    <java.version>11</java.version>
  </properties>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-thymeleaf</artifactId>
    </dependency>
```

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

<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <scope>runtime</scope>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-devtools</artifactId>
</dependency>

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>
</dependencies>

<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
  </plugins>
</build>

</project>
```

Step 3: Update the **application.properties** file with the following properties.

```
server.port=8181
spring.jpa.hibernate.ddl-auto=none
spring.datasource.url=jdbc:mysql://localhost:3306/adiDB
spring.datasource.username=root
spring.datasource.password=seshu
```

Step 3: Create Product model class.

```
package com.seshu.model;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class Product {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String name;
    private String brand;
    private Double price;

    public Product() {
    }

    public Product(Long id, String name, String brand, Double price) {
        super();
        this.id = id;
        this.name = name;
        this.brand = brand;
        this.price = price;
    }

    public Long getId() {
        return id;
    }

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

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getBrand() {
        return brand;
    }
}
```

```
    public void setBrand(String brand) {  
        this.brand = brand;  
    }  
  
    public Double getPrice() {  
        return price;  
    }  
  
    public void setPrice(Double price) {  
        this.price = price;  
    }  
}
```

Step 4: Create ProductRepository class.

```
package com.seshusoft.repo;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
  
import com.seshusoft.model.Product;  
  
public interface ProductRepository extends JpaRepository<Product, Long> {  
}
```

Step 5: Create ProductService class.

```
package com.seshusoft.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.seshusoft.model.Product;
import com.seshusoft.repo.ProductRepository;

@Service
@Transactional
public class ProductService {

    @Autowired
    private ProductRepository repo;

    public List<Product> listAll() {
        return repo.findAll();
    }

    public void save(Product product) {
        repo.save(product);
    }

    public Product get(long id) {
        return repo.findById(id).get();
    }

    public void delete(long id) {
        repo.deleteById(id);
    }
}
```

Step 6: Create AppController class.

```
package com.seshusoft.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.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;

import com.seshusoft.model.Product;
import com.seshusoft.service.ProductService;

@Controller
public class AppController {

    @Autowired
    private ProductService service;

    @RequestMapping("/")
    public String viewHomePage(Model model) {
        List<Product> listProducts = service.listAll();
        model.addAttribute("listProducts", listProducts);

        return "index";
    }

    @RequestMapping("/new")
    public String showNewProductPage(Model model) {
        Product product = new Product();
        model.addAttribute("product", product);

        return "new_product";
    }

    @RequestMapping(value = "/save", method = RequestMethod.POST)
    public String saveProduct(@ModelAttribute("product") Product product) {
        service.save(product);

        return "redirect:/";
    }

    @RequestMapping("/edit/{id}")
    public ModelAndView showEditProductPage(@PathVariable(name = "id") int id) {
        ModelAndView mav = new ModelAndView("edit_product");
        Product product = service.get(id);
        mav.addObject("product", product);

        return mav;
    }
}
```



```

@RequestMapping("/delete/{id}")
public String deleteProduct(@PathVariable(name = "id") int id) {
    service.delete(id);
    return "redirect:/";
}
}

```

Step 7: Create index.html

```

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:th="http://www.thymeleaf.org">

<head>
    <meta charset="utf-8" />
    <title>app</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWfSpd3yD65VohhpuuCOMLASjC "
crossorigin="anonymous">
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
</head>

<body>
    <div class="container mt-5">
        <h1 class="text-center">Product Management System</h1>
        <a href="new" class="btn btn-primary">Create New Product</a>
        <table class="table table-striped mt-3">
            <thead>
                <tr>
                    <th>Id</th>
                    <th>Name</th>
                    <th>Brand</th>
                    <th>Price</th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody>
                <tr th:each="product : ${listProducts}">
                    <td th:text="${product.id}">Id</td>
                    <td th:text="${product.name}">Name</td>
                    <td th:text="${product.brand}">Brand</td>
                    <td th:text="${product.price}">Price</td>
                    <td>
                        <a th:href="@{'/edit/' + ${product.id}}" class="btn btn-
success">Update</a>
                        <a th:href="@{'/delete/' + ${product.id}}" class="btn
btn-danger">Delete</a>
                    </td>
                </tr>
            </tbody>
        </table>
    </div>
</body>

```

</html>

Step 8: Create `new_product.html` file.

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:th="http://www.thymeleaf.org">

<head>
    <meta charset="utf-8" />
    <title>Create New Product</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWfSpd3yD65VohhpuuC0mLASjC "
crossorigin="anonymous">
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
</head>

<body>
    <div class="container mt-5">
        <h1 class="text-center">Product Management System</h1>

        <a href="/" class="btn btn-link">HOME</a>

        <h2>Create New Product</h2>
        <br />
        <form action="#" th:action="@{/save}" th:object="${product}" method="post">
            <div class="col-4">
                <label class="form-label">Product Name</label>
                <input type="text" class="form-control" th:field="*{name}">
            </div>
            <div class="col-4">
                <label class="form-label">Product Brand</label>
                <input type="text" class="form-control" th:field="*{brand}">
            </div>
            <div class="col-4">
                <label class="form-label">Product Price</label>
                <input type="text" class="form-control" th:field="*{price}">
            </div>
            <div class="col-4">
                <button type="submit" class="btn btn-primary mt-2">Save</button>
            </div>
        </form>
    </div>
</body>
</html>
```

Step 9: Create edit_product.html file

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:th="http://www.thymeleaf.org">

<head>
  <meta charset="utf-8" />
  <title>Edit Product</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWFspd3yD65VohhpuuCOMLASjC "
crossorigin="anonymous">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
</head>

<body>
  <div class="container mt-5">
    <h1 class="text-center">Product Management System</h1>

    <a href="/" class="btn btn-link">HOME</a>

    <h2>Edit Product</h2>
    <br />
    <form action="#" th:action="@{/save}" th:object="${product}" method="post">
      <div class="col-4">
        <label class="form-label">Product Id</label>
        <input type="text" class="form-control" th:field="*{id}" readonly>
      </div>
      <div class="col-4">
        <label class="form-label">Product Name</label>
        <input type="text" class="form-control" th:field="*{name}">
      </div>
      <div class="col-4">
        <label class="form-label">Product Brand</label>
        <input type="text" class="form-control" th:field="*{brand}">
      </div>
      <div class="col-4">
        <label class="form-label">Product Price</label>
        <input type="text" class="form-control" th:field="*{price}">
      </div>
      <div class="col-4">
        <button type="submit" class="btn btn-primary mt-2">Update</button>
      </div>
    </form>
  </div>
</body>
</html>
```

Step 10: Run the spring starter.

```
package com.seshusoft;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SpringBootMvcApplication {

    public static void main(String[] args) {
        SpringApplication.run(SpringBootMvcApplication.class, args);
    }

}
```