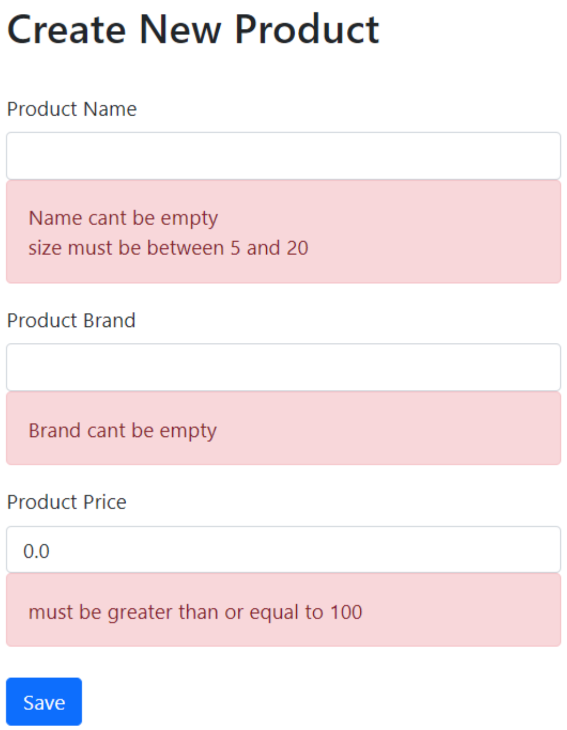
Adding Validation to Spring Boot Mvc Application



Create New Product

Product Name

Name cant be empty
size must be between 5 and 20

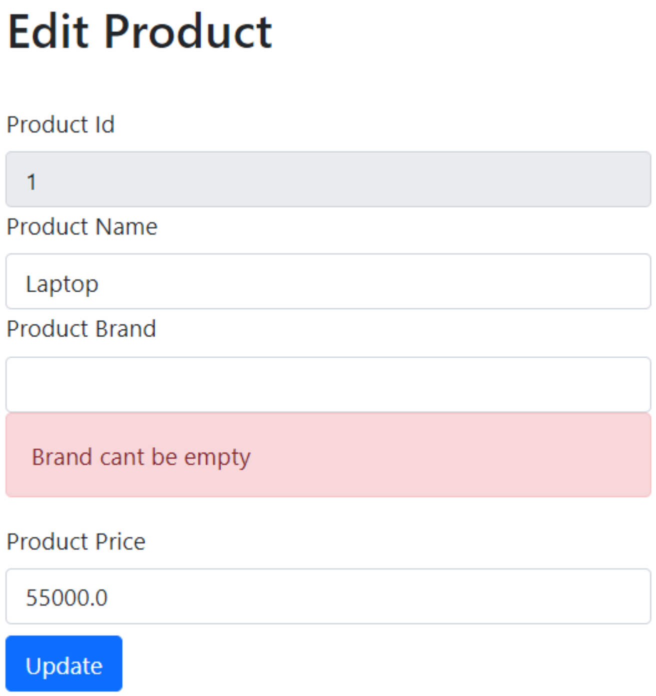
Product Brand

Brand cant be empty

Product Price

0.0

must be greater than or equal to 100



Edit Product

Product Id

1

Product Name

Laptop

Product Brand

Brand cant be empty

Product Price

55000.0

Update SpringBootMvc Project.

Steps

Step 1: Update pom.xml file with the fallowing dependencies.

```
<?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>
```

```
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-validation</artifactId>
        </dependency>
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-validator</artifactId>
            <version>6.0.10.Final</version>
        </dependency>
    </dependencies>

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

</project>
```

Step 2: Update Product.java file.

```
package com.seshusoft.model;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.validation.constraints.Min;
import javax.validation.constraints.NotEmpty;
import javax.validation.constraints.Size;

@Entity
public class Product {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotEmpty(message = "Name can't be empty")
    @Size(min = 5, max = 20)
    private String name;

    @NotEmpty(message = "Brand can't be empty")
    private String brand;

    @Min(value = 100)
    private Double price;
```

```java
    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 3: Update AppController.java

```
package com.seshusoft.controller;

import java.util.List;

import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
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(@Valid @ModelAttribute("product") Product product,
BindingResult result) {

         if (result.hasErrors()) {
              return "new_product";
       }
        service.save(product);

        return "redirect:/";
    }
```

```
	@RequestMapping(value = "/update", method = RequestMethod.POST)
	public String UpdateProduct(@Valid @ModelAttribute("product") Product product,
BindingResult result) {

		 if (result.hasErrors()) {
		 	return "edit_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 4: Update **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/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>Create New Product</h2>
                <br />
                <form action="#" th:action="@{/save}" th:object="${product}" method="post">
                        <div class="col-4">
                                <label class="form-label">Product Name</label>
                                <input type="text" class="form-control" th:field="*{name}" >
                                <div class="alert alert-danger" th:if="${#fields.hasErrors('name')}"
th:errors="*{name}"></div>
                        </div>
                        <div class="col-4">
                                <label class="form-label">Product Brand</label>
                                <input type="text" class="form-control" th:field="*{brand}" >
                                <div class="alert alert-danger" th:if="${#fields.hasErrors('brand')}"
th:errors="*{brand}"></div>
                        </div>
                        <div class="col-4">
                                <label class="form-label">Product Price</label>
                                <input type="text" class="form-control" th:field="*{price}" >
                                <div class="alert alert-danger" th:if="${#fields.hasErrors('price')}"
th:errors="*{price}"></div>
                        </div>
                        <div class="col-4">
                                <button type="submit" class="btn btn-primary mt-2">Save</button>
                        </div>
                </form>
        </div>
</body>

</html>
```

Step 5: Update **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="@{/update}" th:object="${product}" method="post">
            <div class="col-4">
                <label class="form-label">Product Id</label>
                <input type="text" class="form-control" th:field="*{id}" readonly>
            </div>
            <div class="col-4">
                <label class="form-label">Product Name</label>
                <input type="text" class="form-control" th:field="*{name}">
                <div class="alert alert-danger" th:if="${#fields.hasErrors('name')}"
th:errors="*{name}"></div>
            </div>
            <div class="col-4">
                <label class="form-label">Product Brand</label>
                <input type="text" class="form-control" th:field="*{brand}">
                <div class="alert alert-danger" th:if="${#fields.hasErrors('brand')}"
th:errors="*{brand}"></div>
            </div>
            <div class="col-4">
                <label class="form-label">Product Price</label>
                <input type="text" class="form-control" th:field="*{price}">
                <div class="alert alert-danger" th:if="${#fields.hasErrors('price')}"
th:errors="*{price}"></div>
            </div>
            <div class="col-4">
                <button type="submit" class="btn btn-primary mt-2">Update</button>
            </div>
        </form>
    </div>
</body>

</html>
```

Step 6: Run 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);
	}
}
```

Important Annotations:

@ModelAttribute:

This annotation binds a method parameter or method return value to a named model attribute and then exposes it to a web view.

@PathVariable:

This annotation is used to handle template variables in the request URI mapping