**Exercise 4: Online Bookstore - Processing Request Body and Form Data**

**Business Scenario:**

Create endpoints to accept and process JSON request bodies and form data for customer registrations.

**Instructions:**

1. **Request Body:**
   * Implement a POST endpoint to create a new customer by accepting a JSON request body.
2. **Form Data:**
   * Implement an endpoint to process form data for customer registrations.

**Answers: -**

**Exercise 4: Online Bookstore - Processing Request Body and Form Data**

**1. Request Body**

**Step-by-Step Guide:**

1. **Implementing a POST Endpoint to Create a New Customer by Accepting a JSON Request Body:**
   * **Defining the Customer Entity:**
     + First, create a Customer class in the src/main/java/com/example/bookstoreapi/model package to represent the customer data.

package com.example.bookstoreapi.model;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

public class Customer {

private int id;

private String firstName;

private String lastName;

private String email;

private String password;

}

* + **Create the CustomerController Class:**
    - In the src/main/java/com/example/bookstoreapi/controller package, create a new class named CustomerController.

package com.example.bookstoreapi.controller;

import com.example.bookstoreapi.model.Customer;

import org.springframework.web.bind.annotation.\*;

import java.util.ArrayList;

import java.util.List;

@RestController

@RequestMapping("/customers")

public class CustomerController {

private List<Customer> customers = new ArrayList<>();

// POST: /customers - Create a new customer

@PostMapping

public Customer createCustomer(@RequestBody Customer customer) {

customers.add(customer);

return customer;

}

}

* + **Explanation:**
    - **@RequestBody:** This annotation is used to bind the JSON body of the HTTP request to the Customer object. The method will then process the JSON data and add the new customer to the list.
  + **Example JSON Request:**

{

"id": 1,

"firstName": "John",

"lastName": "Doe",

"email": "john.doe@example.com",

"password": "securepassword"

}

* + **Example Request:**
    - POST /customers
    - Body: (JSON data)

**2. Form Data**

1. **Implement an Endpoint to Process Form Data for Customer Registrations:**
   * **Defining the Form Data Endpoint in the CustomerController:**

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

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

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

@RestController

@RequestMapping("/customers")

public class CustomerController {

private List<Customer> customers = new ArrayList<>();

// POST: /customers - Create a new customer with JSON request body

@PostMapping

public Customer createCustomer(@RequestBody Customer customer) {

customers.add(customer);

return customer;

}

// POST: /customers/register - Register a new customer using form data

@PostMapping("/register")

public Customer registerCustomer(@RequestParam String firstName,

@RequestParam String lastName,

@RequestParam String email,

@RequestParam String password) {

Customer customer = new Customer(customers.size() + 1, firstName, lastName, email, password);

customers.add(customer);

return customer;

}

}

* + **Explanation:**
    - **@RequestParam:** This annotation is used to bind form data from the request to the method parameters. Each form field is mapped to a parameter in the method.
  + **Example Form Data Request:**
    - **Endpoint:** POST /customers/register
    - **Form Fields:**
      * firstName: John
      * lastName: Doe
      * email: john.doe@example.com
      * password: securepassword
  + **Note:**
    - The registerCustomer method processes the form data and creates a new Customer object, which is then added to the list and returned as a response.

With these endpoints, our application can now handle both JSON request bodies and form data for customer registrations, enhancing the flexibility of our RESTful service.