Exercise 1: Online Bookstore - Setting Up RESTful Services

Business Scenario:

You are tasked with developing a RESTful service for an online bookstore. The service will manage books, authors, and customers.

Instructions:

1. **Setup Spring Boot Project:**
    - Initialize a new Spring Boot project named **BookstoreAPI**.
    - Add dependencies: **Spring Web, Spring Boot DevTools, Lombok**.
2. **Project Structure:**
    - Familiarize yourself with the generated project structure.
3. **What's New in Spring Boot 3:**
    - Explore and document the new features introduced in Spring Boot 3.

Exercise 2: Online Bookstore - Creating Basic REST Controllers

Business Scenario:

Implement RESTful endpoints to manage books.

Instructions:

1. **Create Book Controller:**
    - Define a **BookController** class with request mappings for /books.
2. **Handle HTTP Methods:**
    - Implement methods to handle **GET**, **POST**, **PUT**, and **DELETE** requests.
3. **Return JSON Responses:**
    - Ensure the controller returns JSON responses.
    - Define the Book entity with attributes like **id, title, author, price**, and **isbn**.

Exercise 3: Online Bookstore - Handling Path Variables and Query Parameters

Business Scenario:

Enhance the book management endpoints to handle dynamic URLs and query parameters.

Instructions:

1. **Path Variables:**
    - Implement an endpoint to fetch a book by its ID using a path variable.
2. **Query Parameters:**
    - Implement an endpoint to filter books based on query parameters like title and author.

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.

Exercise 5: Online Bookstore - Customizing Response Status and Headers

Business Scenario:

Customize the HTTP response status and headers for the book management endpoints.

Instructions:

1. **Response Status:**
    - Use **@ResponseStatus** to customize HTTP status codes for your endpoints.
2. **Custom Headers:**
    - Add custom headers to the response using **ResponseEntity**.

Exercise 6: Online Bookstore - Exception Handling in REST Controllers

Business Scenario:

Implement a global exception handling mechanism for the bookstore RESTful services.

Instructions:

1. **Global Exception Handler:**
    - Create a **GlobalExceptionHandler** class using **@ControllerAdvice**.
    - Define methods to handle various exceptions and return appropriate HTTP status codes.

Exercise: Online Bookstore - Introduction to Data Transfer Objects (DTOs)

Business Scenario:

Use DTOs to transfer data between the client and server for books and customers.

Instructions:

1. **Create DTOs:**
    - Define BookDTO and CustomerDTO classes.
2. **Mapping Entities to DTOs:**
    - Use a library like **MapStruct** or **ModelMapper** to map entities to DTOs and vice versa.
3. **Custom Serialization/Deserialization:**
    - Customize JSON serialization and deserialization using Jackson annotations.

Exercise 8: Online Bookstore - Implementing CRUD Operations

Business Scenario:

Implement Create, Read, Update, and Delete operations for the Book and Customer entities.

Instructions:

1. **CRUD Endpoints:**
    - Implement endpoints for creating, reading, updating, and deleting books and customers.
2. **Validating Input Data:**
    - Use validation annotations like **@NotNull, @Size**, and **@Min** to validate input data.
3. **Optimistic Locking:**

- Implement optimistic locking for concurrent updates using JPA versioning.

Exercise 9: Online Bookstore - Understanding HATEOAS

Business Scenario:

Enhance your REST API to follow HATEOAS principles for navigation through resources.

Instructions:

1. **Add Links to Resources:**
    - Use **Spring HATEOAS** to add links to resources in your API responses.
2. **Hypermedia-Driven APIs:**
    - Build and consume hypermedia-driven APIs.

Exercise 10: Online Bookstore - Configuring Content Negotiation

Business Scenario:

Support different media types (JSON, XML) for your bookstore's RESTful services.

Instructions:

1. **Content Negotiation:**
    - Configure Spring Boot to support content negotiation.
2. **Accept Header:**
    - Implement logic to produce and consume different media types based on the Accept header.

Exercise 11: Online Bookstore - Integrating Spring Boot Actuator

Business Scenario:

Monitor and manage your bookstore's RESTful services using Spring Boot Actuator.

Instructions:

1. **Add Actuator Dependency:**
    - Include the Spring Boot Actuator dependency in your project.
2. **Expose Actuator Endpoints:**

    - Enable and customize Actuator endpoints.
3. **Custom Metrics:**
    - Expose custom metrics for monitoring your application.

Exercise 12: Online Bookstore - Securing RESTful Endpoints with Spring Security

Business Scenario:

Secure your bookstore's RESTful endpoints using Spring Security with JWT-based authentication.

Instructions:

1. **Add Spring Security:**
    - Integrate Spring Security into your project.
2. **JWT Authentication:**
    - Implement JWT-based authentication and authorization.
3. **CORS Handling:**
    - Configure CORS to handle cross-origin requests.

Exercise 13: Online Bookstore - Unit Testing REST Controllers

Business Scenario:

Write unit tests for your bookstore's REST controllers using JUnit and Mockito.

Instructions:

1. **JUnit Setup:**
    - Set up JUnit and Mockito in your project.
2. **MockMvc:**
    - Use MockMvc to write unit tests for your REST controllers.
3. **Test Coverage:**
    - Ensure comprehensive test coverage and follow best practices for testing.

Exercise 14: Online Bookstore - Integration Testing for REST Services

Business Scenario:

Write integration tests for your bookstore's RESTful services.

Instructions:

1. **Spring Test:**
   - Set up Spring Test for integration testing.
2. **MockMvc Integration:**
   - Use MockMvc for end-to-end testing of your REST endpoints.
3. **Database Integration:**
   - Include database integration in your tests using an in-memory database like **H2**.

Scenario 15: Online Bookstore - API Documentation with Swagger

Business Scenario:

Document your bookstore's REST APIs using Swagger and Springdoc.

Instructions:

1. **Add Swagger Dependency:**
   - Include Swagger or Springdoc dependencies in your project.
2. **Document Endpoints:**
   - Annotate your REST controllers and methods to generate API documentation.
3. **API Documentation:**
   - Generate and review the API documentation using **Swagger UI** or **Springdoc UI**.