**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.

**pom.xml**

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-hateoas</artifactId>

</dependency>

**Book.java**

package com.example.bookstoreapi.model;

import jakarta.persistence.\*;

import jakarta.validation.constraints.Min;

import jakarta.validation.constraints.NotNull;

import jakarta.validation.constraints.Size;

@Entity

public class Book {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@NotNull(message = "Title is mandatory")

@Size(min = 2, max = 100, message = "Title must be between 2 and 100 characters")

private String title;

@NotNull(message = "Author is mandatory")

@Size(min = 2, max = 50, message = "Author must be between 2 and 50 characters")

private String author;

@NotNull(message = "Price is mandatory")

@Min(value = 0, message = "Price must be greater than or equal to 0")

private Double price;

@Version

private int version;

// Constructors

public Book() {}

public Book(Long id, String title, String author, Double price) {

this.id = id;

this.title = title;

this.author = author;

this.price = price;

}

// Getters and Setters

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

public Double getPrice() {

return price;

}

public void setPrice(Double price) {

this.price = price;

}

public int getVersion() {

return version;

}

public void setVersion(int version) {

this.version = version;

}

}

**Customer.java**

package com.example.bookstoreapi.model;

import jakarta.persistence.\*;

import jakarta.validation.constraints.Email;

import jakarta.validation.constraints.NotNull;

import jakarta.validation.constraints.Size;

@Entity

public class Customer {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@NotNull(message = "Name is mandatory")

@Size(min = 2, max = 50, message = "Name must be between 2 and 50 characters")

private String name;

@NotNull(message = "Email is mandatory")

@Email(message = "Email should be valid")

private String email;

@Version

private int version;

// Constructors

public Customer() {}

public Customer(Long id, String name, String email) {

this.id = id;

this.name = name;

this.email = email;

}

// Getters and Setters

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 getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public int getVersion() {

return version;

}

public void setVersion(int version) {

this.version = version;

}

}

**BookResource.java**

package com.example.bookstoreapi.resource;

import com.example.bookstoreapi.model.Book;

import org.springframework.hateoas.RepresentationModel;

import org.springframework.hateoas.server.core.Relation;

@Relation(collectionRelation = "books")

public class BookResource extends RepresentationModel<BookResource> {

private Long id;

private String title;

private String author;

private Double price;

public BookResource(Book book) {

this.id = book.getId();

this.title = book.getTitle();

this.author = book.getAuthor();

this.price = book.getPrice();

}

// Getters and Setters

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

public Double getPrice() {

return price;

}

public void setPrice(Double price) {

this.price = price;

}

}

**CustomerResource.java**

package com.example.bookstoreapi.resource;

import com.example.bookstoreapi.model.Customer;

import org.springframework.hateoas.RepresentationModel;

import org.springframework.hateoas.server.core.Relation;

@Relation(collectionRelation = "customers")

public class CustomerResource extends RepresentationModel<CustomerResource> {

private Long id;

private String name;

private String email;

public CustomerResource(Customer customer) {

this.id = customer.getId();

this.name = customer.getName();

this.email = customer.getEmail();

}

// Getters and Setters

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 getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

**BookController.java**

package com.example.bookstoreapi.controller;

import com.example.bookstoreapi.model.Book;

import com.example.bookstoreapi.resource.BookResource;

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

import org.springframework.hateoas.CollectionModel;

import org.springframework.hateoas.EntityModel;

import org.springframework.hateoas.Link;

import org.springframework.hateoas.server.mvc.WebMvcLinkBuilder;

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

import java.util.List;

import java.util.stream.Collectors;

import static org.springframework.hateoas.server.mvc.WebMvcLinkBuilder.\*;

@RestController

@RequestMapping("/api/books")

public class BookController {

@Autowired

private BookService bookService;

@GetMapping("/{id}")

public EntityModel<BookResource> getBookById(@PathVariable Long id) {

Book book = bookService.getBookById(id);

BookResource resource = new BookResource(book);

Link selfLink = linkTo(methodOn(BookController.class).getBookById(id)).withSelfRel();

resource.add(selfLink);

return EntityModel.of(resource);

}

@GetMapping

public CollectionModel<BookResource> getAllBooks() {

List<BookResource> resources = bookService.getAllBooks().stream()

.map(BookResource::new)

.collect(Collectors.toList());

resources.forEach(resource -> {

Link selfLink = linkTo(methodOn(BookController.class).getBookById(resource.getId())).withSelfRel();

resource.add(selfLink);

});

Link selfLink = linkTo(methodOn(BookController.class).getAllBooks()).withSelfRel();

return CollectionModel.of(resources, selfLink);

}

@PostMapping

public EntityModel<BookResource> createBook(@RequestBody Book book) {

Book createdBook = bookService.createBook(book);

BookResource resource = new BookResource(createdBook);

Link selfLink = linkTo(methodOn(BookController.class).getBookById(createdBook.getId())).withSelfRel();

resource.add(selfLink);

return EntityModel.of(resource);

}

@PutMapping("/{id}")

public EntityModel<BookResource> updateBook(@PathVariable Long id, @RequestBody Book book) {

Book updatedBook = bookService.updateBook(id, book);

BookResource resource = new BookResource(updatedBook);

Link selfLink = linkTo(methodOn(BookController.class).getBookById(id)).withSelfRel();

resource.add(selfLink);

return EntityModel.of(resource);

}

@DeleteMapping("/{id}")

public void deleteBook(@PathVariable Long id) {

bookService.deleteBook(id);

}

}

**CustomerController.java**

package com.example.bookstoreapi.controller;

import com.example.bookstoreapi.model.Customer;

import com.example.bookstoreapi.resource.CustomerResource;

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

import org.springframework.hateoas.CollectionModel;

import org.springframework.hateoas.EntityModel;

import org.springframework.hateoas.Link;

import org.springframework.hateoas.server.mvc.WebMvcLinkBuilder;

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

import java.util.List;

import java.util.stream.Collectors;

import static org.springframework.hateoas.server.mvc.WebMvcLinkBuilder.\*;

@RestController

@RequestMapping("/api/customers")

public class CustomerController {

@Autowired

private CustomerService customerService;

@GetMapping("/{id}")

public EntityModel<CustomerResource> getCustomerById(@PathVariable Long id) {

Customer customer = customerService.getCustomerById(id);

CustomerResource resource = new CustomerResource(customer);

Link selfLink = linkTo(methodOn(CustomerController.class).getCustomerById(id)).withSelfRel();

resource.add(selfLink);

return EntityModel.of(resource);

}

@GetMapping

public CollectionModel<CustomerResource> getAllCustomers() {

List<CustomerResource> resources = customerService.getAllCustomers().stream()

.map(CustomerResource::new)

.collect(Collectors.toList());

resources.forEach(resource -> {

Link selfLink = linkTo(methodOn(CustomerController.class).getCustomerById(resource.getId())).withSelfRel();

resource.add(selfLink);

});

Link selfLink = linkTo(methodOn(CustomerController.class).getAllCustomers()).withSelfRel();

return CollectionModel.of(resources, selfLink);

}

@PostMapping

public EntityModel<CustomerResource> createCustomer(@RequestBody Customer customer) {

Customer createdCustomer = customerService.createCustomer(customer);

CustomerResource resource = new CustomerResource(createdCustomer);

Link selfLink = linkTo(methodOn(CustomerController.class).getCustomerById(createdCustomer.getId())).withSelfRel();

resource.add(selfLink);

return EntityModel.of(resource);

}

@PutMapping("/{id}")

public EntityModel<CustomerResource> updateCustomer(@PathVariable Long id, @RequestBody Customer customer) {

Customer updatedCustomer = customerService.updateCustomer(id, customer);

CustomerResource resource = new CustomerResource(updatedCustomer);

Link selfLink = linkTo(methodOn(CustomerController.class).getCustomerById(id)).withSelfRel();

resource.add(selfLink);

return EntityModel.of(resource);

}

@DeleteMapping("/{id}")

public void deleteCustomer(@PathVariable Long id) {

customerService.deleteCustomer(id);

}

}