**Exercise 8: Online Bookstore - Implementing CRUD Operations**

**BookController**

package com.bookstore.api.controllers;

import com.bookstore.api.dtos.BookDTO;

import com.bookstore.api.entities.Book;

import com.bookstore.api.mappers.BookMapper;

import com.bookstore.api.services.BookService;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import javax.validation.Valid;

@RestController

@RequestMapping("/books")

public class BookController {

private final BookService bookService;

private final BookMapper bookMapper;

@Autowired

public BookController(BookService bookService, BookMapper bookMapper) {

this.bookService = bookService;

this.bookMapper = bookMapper;

}

@PostMapping

public ResponseEntity<BookDTO> createBook(@RequestBody @Valid BookDTO bookDTO) {

Book book = bookMapper.bookDTOToBook(bookDTO);

book = bookService.createBook(book);

return new ResponseEntity<>(bookMapper.bookToBookDTO(book), HttpStatus.CREATED);

}

@GetMapping

public ResponseEntity<List<BookDTO>> getAllBooks() {

List<Book> books = bookService.getAllBooks();

return new ResponseEntity<>(books.stream().map(bookMapper::bookToBookDTO).toList(), HttpStatus.OK);

}

@GetMapping("/{id}")

public ResponseEntity<BookDTO> getBookById(@PathVariable Long id) {

Book book = bookService.getBookById(id);

return new ResponseEntity<>(bookMapper.bookToBookDTO(book), HttpStatus.OK);

}

@PutMapping("/{id}")

public ResponseEntity<BookDTO> updateBook(@PathVariable Long id, @RequestBody @Valid BookDTO bookDTO) {

Book book = bookMapper.bookDTOToBook(bookDTO);

book = bookService.updateBook(id, book);

return new ResponseEntity<>(bookMapper.bookToBookDTO(book), HttpStatus.OK);

}

@DeleteMapping("/{id}")

public ResponseEntity<Void> deleteBook(@PathVariable Long id) {

bookService.deleteBook(id);

return new ResponseEntity<>(HttpStatus.NO\_CONTENT);

}

}

**CustomerController**

package com.bookstore.api.controllers;

import com.bookstore.api.dtos.CustomerDTO;

import com.bookstore.api.entities.Customer;

import com.bookstore.api.mappers.CustomerMapper;

import com.bookstore.api.services.CustomerService;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import javax.validation.Valid;

@RestController

@RequestMapping("/customers")

public class CustomerController {

private final CustomerService customerService;

private final CustomerMapper customerMapper;

@Autowired

public CustomerController(CustomerService customerService, CustomerMapper customerMapper) {

this.customerService = customerService;

this.customerMapper = customerMapper;

}

@PostMapping

public ResponseEntity<CustomerDTO> createCustomer(@RequestBody @Valid CustomerDTO customerDTO) {

Customer customer = customerMapper.customerDTOToCustomer(customerDTO);

customer = customerService.createCustomer(customer);

return new ResponseEntity<>(customerMapper.customerToCustomerDTO(customer), HttpStatus.CREATED);

}

@GetMapping

public ResponseEntity<List<CustomerDTO>> getAllCustomers() {

List<Customer> customers = customerService.getAllCustomers();

return new ResponseEntity<>(customers.stream().map(customerMapper::customerToCustomerDTO).toList(), HttpStatus.OK);

}

@GetMapping("/{id}")

public ResponseEntity<CustomerDTO> getCustomerById(@PathVariable Long id) {

Customer customer = customerService.getCustomerById(id);

return new ResponseEntity<>(customerMapper.customerToCustomerDTO(customer), HttpStatus.OK);

}

@PutMapping("/{id}")

public ResponseEntity<CustomerDTO> updateCustomer(@PathVariable Long id, @RequestBody @Valid CustomerDTO customerDTO) {

Customer customer = customerMapper.customerDTOToCustomer(customerDTO);

customer = customerService.updateCustomer(id, customer);

return new ResponseEntity<>(customerMapper.customerToCustomerDTO(customer), HttpStatus.OK);

}

@DeleteMapping("/{id}")

public ResponseEntity<Void> deleteCustomer(@PathVariable Long id) {

customerService.deleteCustomer(id);

return new ResponseEntity<>(HttpStatus.NO\_CONTENT);

}

}