Here is the implementation of the DTOs and the mapping between entities and DTOs:

BookDTO.java

package com.example.bookstoreapi.dtos;

import com.fasterxml.jackson.annotation.JsonProperty;

public class BookDTO {

private Long id;

private String title;

private String author;

private String publisher;

private int publicationYear;

public BookDTO() {}

public BookDTO(Long id, String title, String author, String publisher, int publicationYear) {

this.id = id;

this.title = title;

this.author = author;

this.publisher = publisher;

this.publicationYear = publicationYear;

}

// getters and setters

}

CustomerDTO.java

package com.example.bookstoreapi.dtos;

import com.fasterxml.jackson.annotation.JsonProperty;

public class CustomerDTO {

private Long id;

private String firstName;

private String lastName;

private String email;

private String phoneNumber;

public CustomerDTO() {}

public CustomerDTO(Long id, String firstName, String lastName, String email, String phoneNumber) {

this.id = id;

this.firstName = firstName;

this.lastName = lastName;

this.email = email;

this.phoneNumber = phoneNumber;

}

// getters and setters

}

BookMapper.java

package com.example.bookstoreapi.mappers;

import com.example.bookstoreapi.dtos.BookDTO;

import com.example.bookstoreapi.entities.Book;

import org.modelmapper.ModelMapper;

public class BookMapper {

private static ModelMapper modelMapper = new ModelMapper();

public static BookDTO bookToBookDTO(Book book) {

return modelMapper.map(book, BookDTO.class);

}

public static Book bookDTOToBook(BookDTO bookDTO) {

return modelMapper.map(bookDTO, Book.class);

}

}

CustomerMapper.java

package com.example.bookstoreapi.mappers;

import com.example.bookstoreapi.dtos.CustomerDTO;

import com.example.bookstoreapi.entities.Customer;

import org.modelmapper.ModelMapper;

public class CustomerMapper {

private static ModelMapper modelMapper = new ModelMapper();

public static CustomerDTO customerToCustomerDTO(Customer customer) {

return modelMapper.map(customer, CustomerDTO.class);

}

public static Customer customerDTOToCustomer(CustomerDTO customerDTO) {

return modelMapper.map(customerDTO, Customer.class);

}

}

Here's what's new:

DTOs: The BookDTO and CustomerDTO classes are defined to transfer data between the client and server.

Mapping Entities to DTOs: The BookMapper and CustomerMapper classes are used to map entities to DTOs and vice versa using the ModelMapper library.

Custom Serialization/Deserialization: The @JsonProperty annotation is used to customize JSON serialization and deserialization using Jackson annotations.