**Exercise 7: Online Bookstore - Introduction to Data Transfer Objects (DTOs)**

Business Scenario:

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

**Create DTOs**

**1.1. Definition of DTO Classes**

**BookDTO Class**

* **Purpose**: Transfers book data between the client and server.
* **Attributes**:
  + Long id: Unique identifier for the book.
  + String title: Title of the book.
  + String author: Author of the book.
  + Double price: Price of the book.
  + String isbn: ISBN number of the book.

**CustomerDTO Class**

* **Purpose**: Transfers customer data between the client and server.
* **Attributes**:
  + Long id: Unique identifier for the customer.
  + String name: Name of the customer.
  + String email: Email address of the customer.
  + String phone: Phone number of the customer.

**2. Mapping Entities to DTOs**

**2.1. Using MapStruct for Mapping**

**MapStruct** is used to map entities to DTOs and vice versa. It simplifies the conversion process and reduces boilerplate code.

**BookMapper Interface**

* **Methods**:
  + toBookDTO(Book book): Converts a Book entity to BookDTO.
  + toBook(BookDTO bookDTO): Converts a BookDTO to Book entity.

**CustomerMapper Interface**

* **Methods**:
  + toCustomerDTO(Customer customer): Converts a Customer entity to CustomerDTO.
  + toCustomer(CustomerDTO customerDTO): Converts a CustomerDTO to Customer entity.

**3. Custom Serialization/Deserialization**

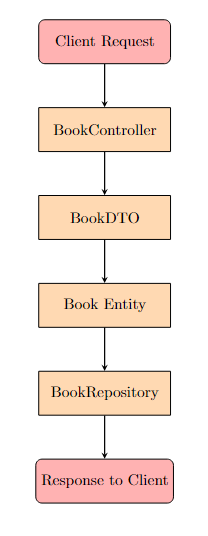
**3.1. Jackson Annotations**

**Jackson** annotations are used to customize JSON serialization and deserialization. These annotations can be applied to DTO fields or classes to control how data is represented in JSON format.

**Custom Serialization Example**:

* **Field-Level Annotation**: @JsonProperty can be used to specify the name of the property in the JSON.
* **DTOs**: Used for transferring data between client and server.
* **Mapping**: Utilizes MapStruct to convert between entities and DTOs.
* **Serialization/Deserialization**: Customizes JSON data representation using Jackson annotations

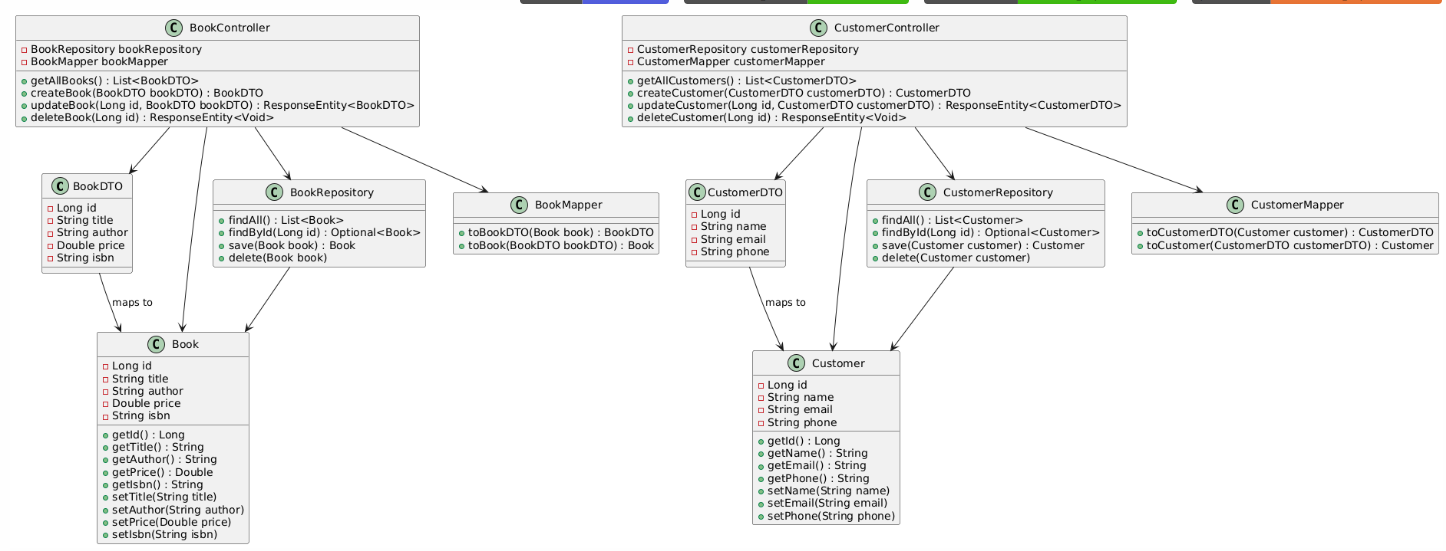
**FLOWCHART :**



**Explanation:**

1. **Client Request**: The process begins when a client sends a request to the server.
2. **BookController**: The BookController handles the request, interacting with DTOs.
3. **BookDTO**: The BookDTO is used to transfer data between the client and server.
4. **Book Entity**: The DTO is converted to a Book entity, which is a representation of the data model.
5. **BookRepository**: The BookRepository is used to perform database operations, such as saving or updating the book entity.
6. **Response to Client**: After processing, a response is sent back to the client.

**CLASS DIAGRAM :**



**Explanation:**

1. **BookDTO** and **CustomerDTO**: Data Transfer Objects used to transfer data between client and server.
2. **Book** and **Customer**: Entities that represent the database model for books and customers.
3. **BookRepository** and **CustomerRepository**: Repositories used for database operations related to books and customers.
4. **BookController** and **CustomerController**: REST controllers that handle HTTP requests and interact with DTOs, entities, and repositories.
5. **BookMapper** and **CustomerMapper**: Mappers used to convert between entities and DTOs.the arrows represent the relationships between classes, indicating how they interact with each other in the application.