Wholesale Order API

Fork and clone the repository at [https://github.com/oroborous/dist-java-wholesale (Links to an external site.)](https://github.com/oroborous/dist-java-wholesale)

This repository contains two modules: client and server.

The server is a Spring Boot project that serves both a Spring MVC application and a REST API for a wholesale ordering system. The client simulates a third-party application that consumes your REST API.

Server

This module contains one Thymeleaf template: index.html. **You do not need to modify the template!**

Your goal is to create a @Controller that can place data in the Model to match this usage in the template:

<tr th:each="order : ${**orderList**}">  
 <td th:text="${**order.customer.name**}">CUSTOMER NAME</td>  
 <td th:text="${#temporals.format(**order.purchaseDate**, 'MMM dd yyyy')}">PURCHASE DATE</td>  
 <td th:text="${**order.purchaseOrderNumber**}">PURCHASE ORDER</td>  
 <td th:text="${**order.product.name**}">PRODUCT NAME</td>  
 <td th:text="${**order.terms**}">TERMS</td>  
 <td th:text="${#temporals.format(**order.shippedDate**, 'MMM dd yyyy')}">SHIPPED DATE</td>  
 <td th:text="${**order.product.cost**}">PRODUCT COST</td>  
</tr>

Create @Entity, @Service, and Repository classes as necessary.

Your completed page should be accessible from [http://localhost:8080  (Links to an external site.)](http://localhost:8080/)and will look like this:

Client

**You do not need to modify anything in this module!**

Your goal is to create a @RestController in the server module that can be  accessed by the JavaScript in script.js.

The @RestController should have one endpoint to match this AJAX call:

url: "http://localhost:8080/api/orders/"

The JSON that is generated by your API should be an array of objects with properties to match this usage:

tr.append("<td>" + **order.customerName** + "</td>");  
tr.append("<td>" + **order.date** + "</td>");  
tr.append("<td>" + **order.poNumber** + "</td>");  
tr.append("<td>" + **order.productName** + "</td>");  
tr.append("<td>" + **order.terms** + "</td>");  
tr.append("<td>" + **order.shipped** + "</td>");  
tr.append("<td>" + **order.total** + "</td>");

In addition to the @RestController, create a DTO (data transfer object) that can be serialized in your JSON response.

Your completed page should be accessible from [http://localhost:63342 (Links to an external site.)](http://localhost:63342/) (the IntelliJ preview server) and will look like this:

Database

The schema contains tables for a wholesale order system. schema.sql and data.sql have already been provided for you.

You will need to create @Entity classes for the following tables. Use @ManyToOne annotations to create the associations between Orders, Product, and Customer objects.

