2024-10-04

应用系统体系架构 — 作业3

# 作业要求

请你在大二开发的E-BookStore系统的基础上，完成下列任务：

## 在上次作业编写的通过Kafka消息中间件处理订单的功能的基础上，迭代更新采用WebSocket方式将订单处理的结果发送回客户端，具体要求为：

### 参考课程给出的React前端和Spring后端采用WebSocket方式通信的样例，按照WebSocketTransfer的方式：

#### 在后端的消息监听器类监听到消息处理结果Topic中的消息后，通过WebSocket发送给前端；

#### 或者，直接在Service中处理好订单后，通过WebSocket发送给前端。

### 你应该参考WebSocketTransfer样例中的方式，正确地将订单处理结果返回给下订单的客户端，不能发送给所有客户端。即，你需要对返回给前端的用户进行筛选。

### 参考课程给出的样例，在你代码中维护WebSocket的客户端Session时，选择线程安全的集合数据结构。

### 你应该将上述功能集成到你的E-Book系统中，如果你无法将上述功能集成到你的E-Book系统中，可以单独建立工程实现，但是会适当扣分。

### 请你编写一个文档，解释你的程序设计方案，包括WebSocket的消息格式设计、筛选客户端的方式设计，以及回答为什么要选择线程安全的集合类型来维护客户端Session，而你选择的类型为什么是线程安全的。将文档与代码一起上传。

## 在数据库事务管理中，持久性是指事务提交后即使数据库产生故障，事务提交的结果仍然可以在数据库中访问到，不会丢失。请编写文档回答下面的问题：

### 如果数据库系统在事务执行过程中不断地将事务操作的结果执行落盘操作，会带来什么潜在问题？可以如何处理？

### 如果数据库系统在事务执行提交后再将事务操作的结果执行落盘操作，会带来什么潜在问题？可以如何处理？

# 提交要求

## 请将你自己编写的源码、脚本和文档压缩后上传，勿压缩整个工程提交，尤其是不要压缩第三方的Jar包。

# 评分标准：

## 能够将订单处理的结果通过WebSocket方式返回给前端，并且能够在前端正确地呈现；并能够正确地对订单用户进行筛选，即对于某个订单来说，只有下该订单的用户才会收到订单处理完成的消息（2分）

## 正确回答为什么要选择线程安全的集合类型来维护客户端Session，而你选择的类型为什么是线程安全的（1分）

## 正确回答有关事务管理的两个问题，每问1分

## 如果不能将作业要求整合在你的E-BookStore中，那么请开发单独的针对作业要求完成的独立工程并提交，但是需要扣分，未整合到系统中的作业要求扣1分。