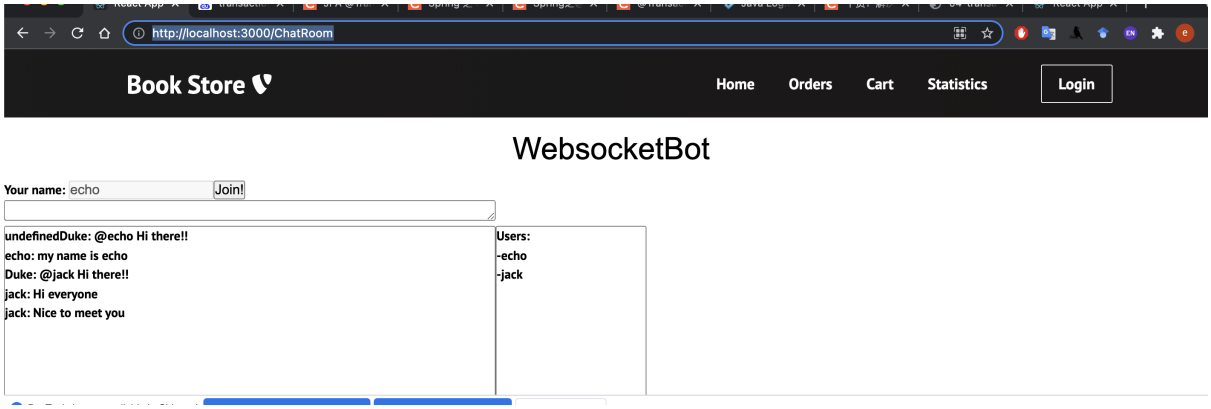
hw2: WebSocket & Transaction

康艺潇 518431910002

构造聊天室：

合并后端代码+重写前端代码，效果如下

这里在Safari浏览器打开并输入了Jack的信息



测试1

OrderService里的下单函数对应的是addFullOrder,设置为REQUIRED, 先调用OrderService的addOrderFromUser（设置为MANDATORY）来写order的数据库，之后调用GetCartService 的clearCart()（设置为REQUIRES_NEW)来清空购物车的数据库。

然后调用userService 得到对应的user,检查user是否存在，如果不存在，addFullOrder就throw error. 发现如果传入一个不存在的user, 查看数据库并没有发生变化，是因为addOrderFromUser发生了回滚。但是cart数据库已经被清空了，说明没有回滚。

MANDATORY：加入现有transaction，如果现在没有transaction就报错。

REQUIRES_NEW: 开启一个transaction，如果之前有就挂起。

这是因为addOrderFromUser加入的是addFullOrder开启的transaction，当它出现异常没有提交，那写order也回滚。然而如果是clearCart, 它自己新开了transaction，和之前的无关了，也就正常的提交了。

```java
@Override
@Transactional(propagation=Propagation.MANDATORY)
public Order addOrderFromUser(Integer user_id, Integer order_price, String date) {
    return orderDao.addOrderFromUser(user_id,order_price,date);
}


@Override
@Transactional(propagation=Propagation.REQUIRED)
public Order addFullOrder(Order order){
    Order order1=this.addOrderFromUser(order.getUserId(), order.getOrder_price(),order.getDate());
    User user = userService.getUserById(order.getUserId());


    List<Cart> cartList = getCartService.getCart();
    for(Cart c:cartList){
        System.out.println(order1.getOrderId()+" "+c.getBookId()+" "+ c.getNumber());
        OrderItem item = this.addOrderItem(order1.getOrderId(),c.getBookId(),c.getNumber());
    }
    getCartService.clearCart();
    if(!user.getUserId().equals(order.getOrderId())){
        System.out.println("the user is null");
        throw new RuntimeException("error");
    }
    System.out.println("Received order < order_id:"+order1.getOrderId() +" user_id:" +
            order1.getUserId()+" order_price:"+order1.getOrder_price() +" date:" +order1.getDate() + ">");
    Order fullOrder= orderDao.findOne(order1.getOrderId());
    return fullOrder;
}
```

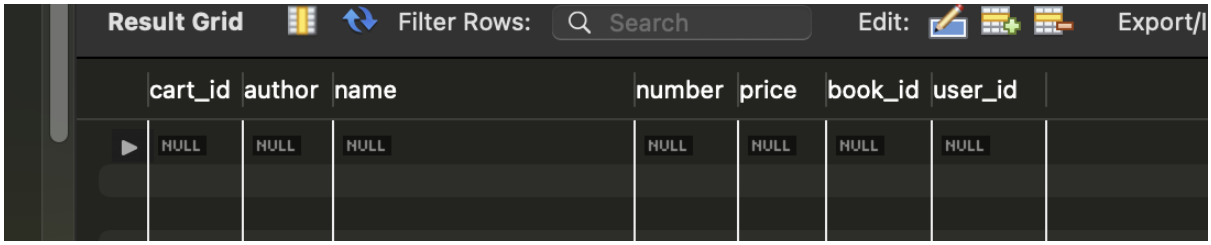
MANDATORY 回滚

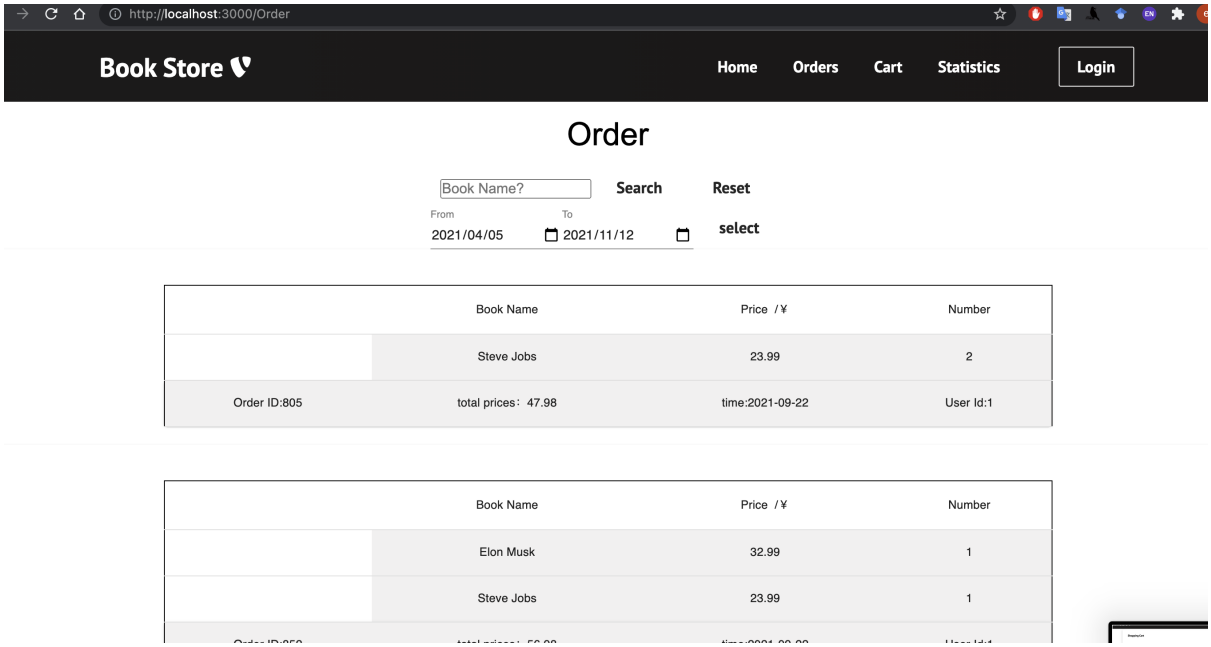
REQUIRED

REQUIRES_NEW 不回滚

```java
@Override
@Transactional(propagation= Propagation.REQUIRES_NEW)
public List<Cart> clearCart() { return cartDao.clearCart(); }
```

可以看见，cart被清空了，但是order并没有出现当天（9月29日）的orde



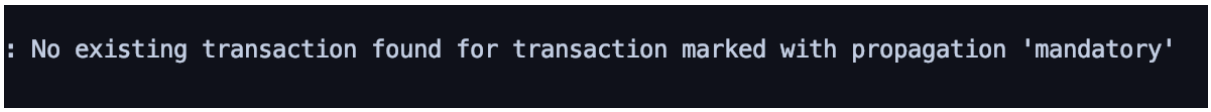


如果下单不成功其实应该清空购物车也要恢复的（逻辑上说），这里只是测试一下回滚的区别

测试2

MANDATORY

要求函数必须在一个已经开始的transaction中运行，这里刚开始我设置为Mandatory，但是此时没有transaction，console报错。



SUPPORT：如果有事务加入，如果没有就在无事务状态运行

这里如果开始是support， 不会新建，那之后调用mandatory就报错了

```java
@Override
@Transactional(propagation=Propagation.MANDATORY)
public Order addOrderFromUser(Integer user_id, Integer order_price, String date) {
    return orderDao.addOrderFromUser(user_id,order_price,date);
}


@Override
@Transactional(propagation=Propagation.SUPPORTS)
public Order addFullOrder(Order order){
    Order order1=this.addOrderFromUser(order.getUserId(), order.getOrder_price(),order.getDate());
    User user = userService.getUserById(order.getUserId());


    List<Cart> cartList = getCartService.getCart();
    for(Cart c:cartList){
        System.out.println(order1.getOrderId()+" "+c.getBookId()+" "+ c.getNumber());
        OrderItem item = this.addOrderItem(order1.getOrderId(), c.getBookId(), c.getNumber());
```

```
ption is org.springframework.transaction.IllegalTransactionStateException: No existing transaction found for transaction marked with propagation 'mandatory'
```