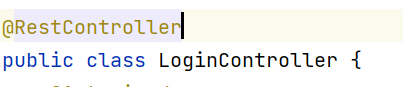
应用系统体系架构作业1

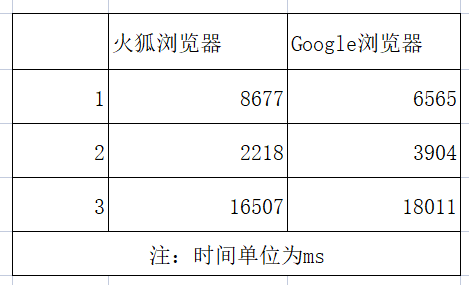
A.

代码实现：





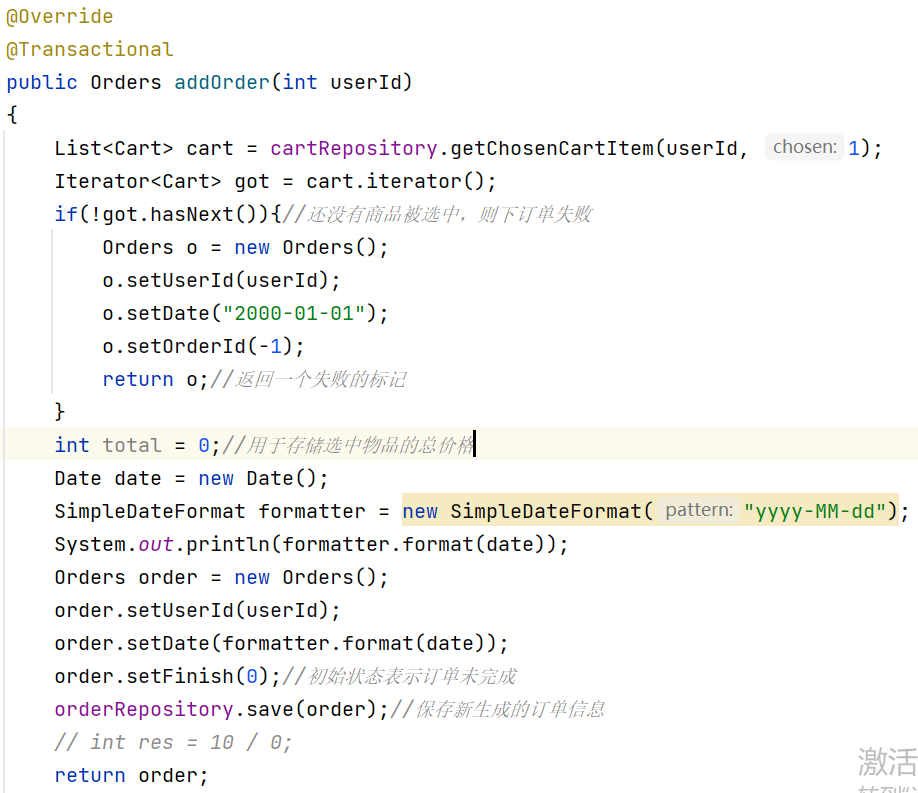
1. 我在Controller层使用的是默认的Singleton的Scope注解，因为虽然发送请求的用户可能不同，但是所有请求可以通过同一个controller分配给不同的service，所以使用了默认的singleton模式。
2. 我在Service层使用的是Session的Scope注解。因为每个用户的登录登出是一个用户专属的个人行为，并且不同用户的登录之间应该互不影响，所以不同用户处理service应该使用不同的对象，所以使用了session注释。
3. 数据表格：



B.

1. 实现的代码：







1. 表格记录：



原因解释：

1. 在所有函数均没有抛出异常的情况下，事务可以正确执行，同时itemDao与orderDao应该是共用一个事务的，因为默认为REQUIRED。
2. 当在进行orderDao中函数之后出错时，由于并没有try-catch，所以抛出的异常会导致程序异常终止，从而在还没有执行orderItemDao中函数时整个事务就直接回滚了。
3. 当在执行orderDao中函数出错时，由于没有try-catch，抛出的异常也会导致程序提前异常终止，也会使得orderItemDao中函数还没有执行就发生了事务的回滚。
4. 由于默认情况下transaction的参数为REQUIRED，故orderItemDao与orderDao中函数共用一个事务。所以当orderItemDao中函数报错时，会导致整个事务的回滚。
5. 在均正常的情况下事务可以成功执行。但是由于此时orderItemDao中函数声明为REQUIRES\_NEW，所以orderItemDao与orderDao执行的不是同一个事务，会出现外键无法关联的情况从而出错。
6. 由于此时异常发生在进入orderItemDao中的函数前，而且没有try-catch，所以抛出的异常会导致orderDao所在的事务直接终止并回滚并且不会再执行orderItemDao中函数。
7. 由于此时报错发生在orderItemDao中函数执行之后，且没有try-catch，所以会导致orderDao以及orderService所在的事务失败并回滚。但是此时orderItemDao中函数声明为REQUIRES\_NEW，所以orderItemDao执行的是与orderDao不同的事务，所以orderItemDao的事务是可以正常运行的。
8. 由于异常发生在orderDao的函数中，且没有做try-catch处理，所以虽然orderItemDao中函数会生成一个新的事务，但是orerDao会在进入orderItemDao中函数之前就崩溃，从而使得orderDao所在的事务整体回滚，且orderItemDao函数不会执行。
9. 由于异常是在orderItemDao中函数抛出的，并且声明为REQUIRES\_NEW会生成新的事务，并且orderItemDao部分做了try-catch处理，所以其报错不会影响orderDao所在的另外一个事物的正常工作。所以最终结果为orderItemDao中新生成的事务由于异常而发生回滚，但是orderDao与orderService所在的事务正常并成功运行。