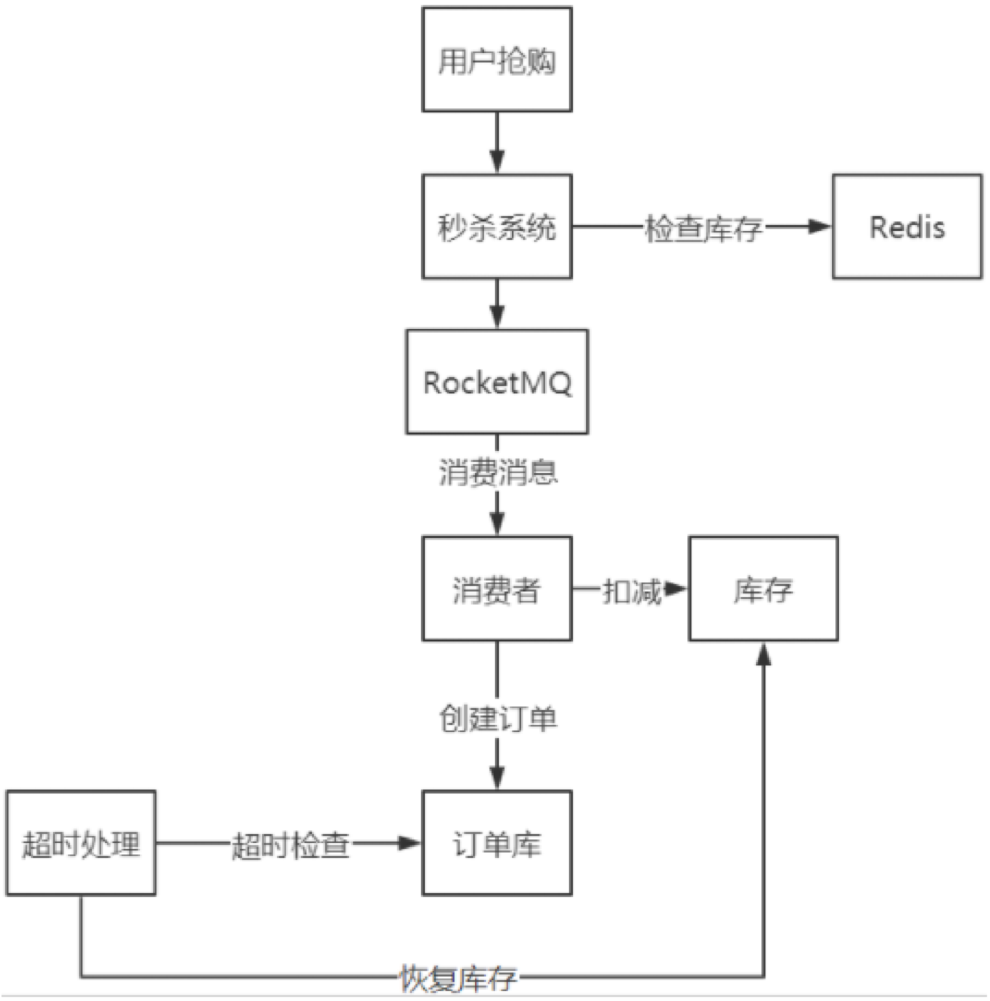
【603】基于RocketMQ设计秒杀

一、要求：

1. 秒杀商品LagouPhone，数量100个。
2. 秒杀商品不能超卖。
3. 抢购链接隐藏
4. Nginx+Redis+RocketMQ+Tomcat+MySQL

提示：



作业需提交：

1、html的截图+搭建的过程+结果截图以文档或视频演示形式提供。

2、作业实现过程的代码

注意：

1、需要把搭建过程及配置用文档写清楚。

2、运行效果：展示商品销售的过程和结果，效果最好以视频形式演示

二、思路分析

1. 可以使用redis 保存秒杀商品的库存
2. 用户点击秒杀按钮后通过redis 获取库存数量，如果还有库存，创建订单并且发送消息到秒杀队列，并且扣减库存
3. 消费端消费秒杀消息，将订单信息入库，更新订单状态为待支付，并将延迟支付消息发送到延迟队列
4. 前端用户支付，支付成功，修改订单状态
5. 前端用户固定时间不支付，（此处延迟队列级别为3,10s钟内不支付），消费端消费延迟队列消息，如果订单状态为支付，则更新订单状态为已取消，并且恢复库存

三、前置准备

3.1、安装redis集群(也可单机版)

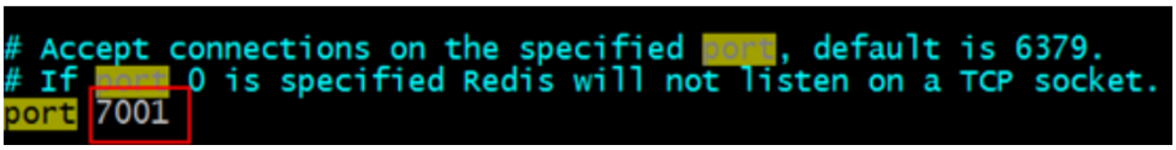
3.1.1 前置准备

```
1、redis linux 安装包，此处选的是redis-5.0.5.tar.gz (可从以下的百度云盘链接下载，也可直接
redis官网下载最新版本：https://redis.io/download)
安装包放在根目录下并解压
tar -zxvf redis-5.0.5.tar.gz

链接：https://pan.baidu.com/s/1A7VBj7IhX1dMV061JpvaLQ
提取码：9hfg
2、linux 虚拟机 1台（此次搭建集群为模拟集群，在一台虚机安装多个节点）
此处安装的ip地址：192.168.31.204   后面的集群创建命令和代码都操作以此ip的服务器
3、RedisCluster最少需要三台主服务器，三台从服务器
- 端口号分别为：7001~7006
```

3.1.2 安装步骤

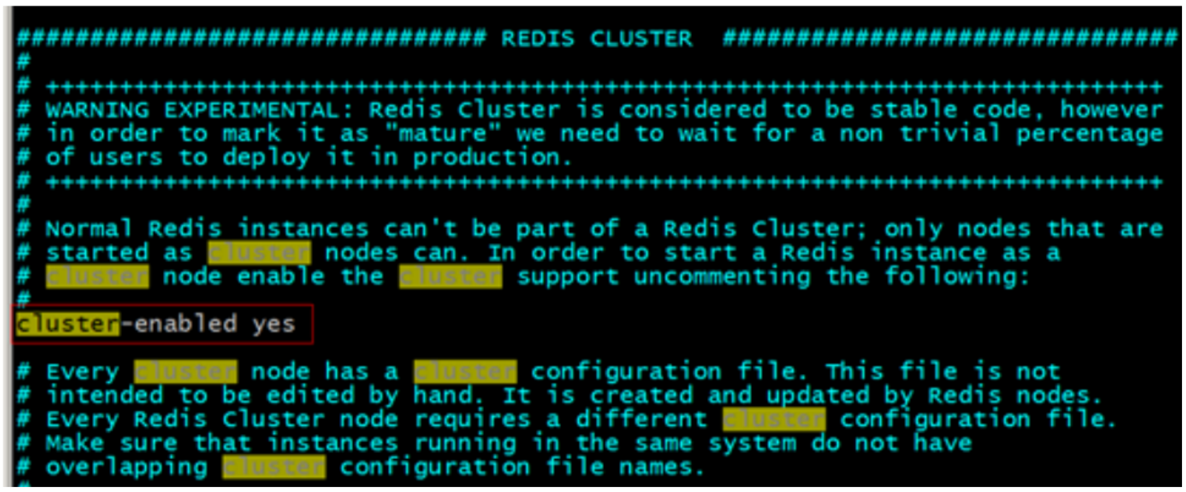
第一步：创建7001实例，并编辑redis.conf文件，修改port为7001

```
mkdir /usr/local/redis-cluster/7001
cp ~/redis-5.0.5/redis.conf /usr/local/redis-cluster/7001/bin
注意：创建实例，即拷贝单机版安装时，生成的bin目录，为7001目录
```



```
# Accept connections on the specified port, default is 6379.
# If port 0 is specified Redis will not listen on a TCP socket.
port 7001
```

第二步：修改redis.conf配置文件，打开cluster-enable yes



```
################################ REDIS CLUSTER  ###############################
#
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# WARNING EXPERIMENTAL: Redis Cluster is considered to be stable code, however
# in order to mark it as "mature" we need to wait for a non trivial percentage
# of users to deploy it in production.
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#
# Normal Redis instances can't be part of a Redis Cluster; only nodes that are
# started as cluster nodes can. In order to start a Redis instance as a
# cluster node enable the cluster support uncommenting the following:
#
cluster-enabled yes

# Every cluster node has a cluster configuration file. This file is not
# intended to be edited by hand. It is created and updated by Redis nodes.
# Every Redis Cluster node requires a different cluster configuration file.
# Make sure that instances running in the same system do not have
# overlapping cluster configuration file names.
```

第三步：复制7001，创建7002~7006实例，**注意端口修改**

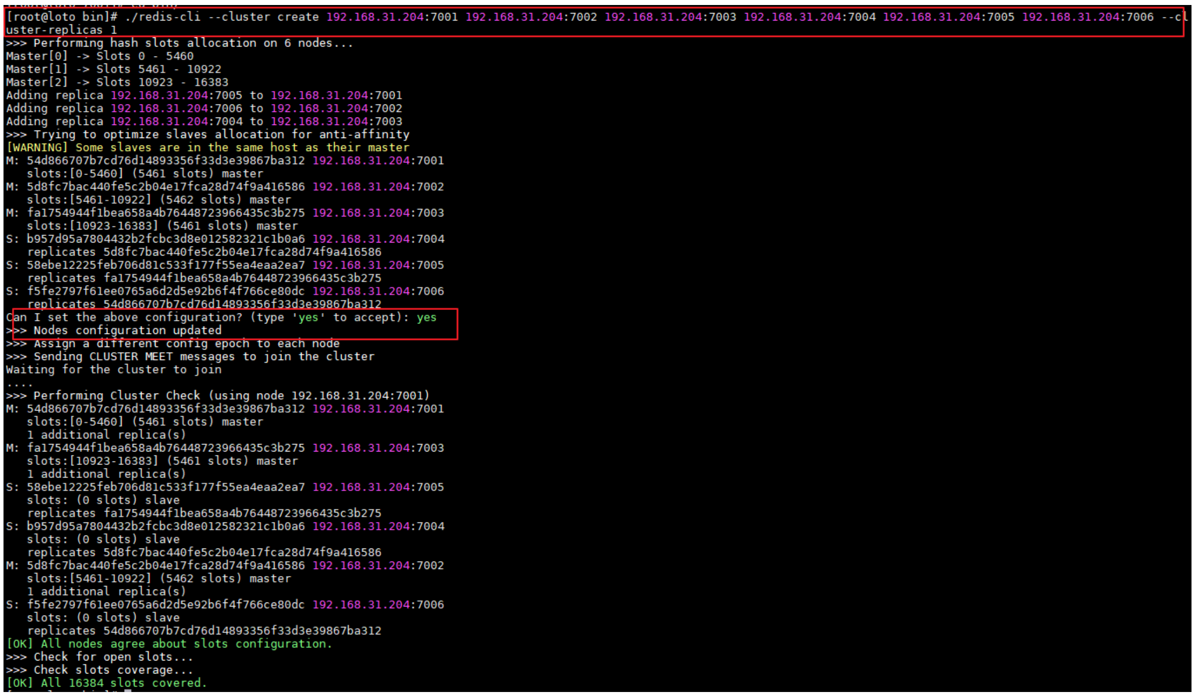
第四步：创建start.sh，启动所有的实例

```
cd 7001/bin
./redis-server redis.conf
cd ../../7002/bin
./redis-server redis.conf

cd ../../7003/bin
./redis-server redis.conf

cd ../../7004/bin
./redis-server redis.conf

cd ../../7005/bin
./redis-server redis.conf

cd ../../7006/bin
./redis-server redis.conf
```

- chmod u+x start.sh (赋写和执行的权限)
- ./start.sh(启动RedisCluster)

第五步：创建Redis集群（创建时Redis里不要有数据）

```
./redis-cli --cluster create 192.168.31.204:7001 192.168.31.204:7002
192.168.31.204:7003 192.168.31.204:7004 192.168.31.204:7005 192.168.31.204:7006
--cluster-replicas 1
```



```
[root@loto bin]# ./redis-cli --cluster create 192.168.31.204:7001 192.168.31.204:7002 192.168.31.204:7003 192.168.31.204:7004 192.168.31.204:7005 192.168.31.204:7006 --cl
uster-replicas 1
>>> Performing hash slots allocation on 6 nodes...
Master[0] -> Slots 0 - 5460
Master[1] -> Slots 5461 - 10922
Master[2] -> Slots 10923 - 16383
Adding replica 192.168.31.204:7005 to 192.168.31.204:7001
Adding replica 192.168.31.204:7006 to 192.168.31.204:7002
Adding replica 192.168.31.204:7004 to 192.168.31.204:7003
>>> Trying to optimize slaves allocation for anti-affinity
[WARNING] Some slaves are in the same host as their master
M: 54d866707b7cd76d14893356f33d3e39867ba312 192.168.31.204:7001
   slots:[0-5460] (5461 slots) master
M: 5d8fc7bac440fe5c2b04e17fca28d74f9a416586 192.168.31.204:7002
   slots:[5461-10922] (5462 slots) master
M: fa1754944f1bea658a4b76448723966435c3b275 192.168.31.204:7003
   slots:[10923-16383] (5461 slots) master
S: b957d95a7804432b2fcbc3d8e012582321c1b0a6 192.168.31.204:7004
   replicates 5d8fc7bac440fe5c2b04e17fca28d74f9a416586
S: 58ebe12225feb706d81c533f177f55ea4eaa2ea7 192.168.31.204:7005
   replicates fa1754944f1bea658a4b76448723966435c3b275
S: f5fe2797f61ee0765a6d2d5e92b6f4f766ce80dc 192.168.31.204:7006
   replicates 54d866707b7cd76d14893356f33d3e39867ba312
Can I set the above configuration? (type 'yes' to accept): yes
>>> Nodes configuration updated
>>> Assign a different config epoch to each node
>>> Sending CLUSTER MEET messages to join the cluster
Waiting for the cluster to join
....
>>> Performing Cluster Check (using node 192.168.31.204:7001)
M: 54d866707b7cd76d14893356f33d3e39867ba312 192.168.31.204:7001
   slots:[0-5460] (5461 slots) master
   1 additional replica(s)
M: fa1754944f1bea658a4b76448723966435c3b275 192.168.31.204:7003
   slots:[10923-16383] (5461 slots) master
   1 additional replica(s)
S: 58ebe12225feb706d81c533f177f55ea4eaa2ea7 192.168.31.204:7005
   slots: (0 slots) slave
   replicates fa1754944f1bea658a4b76448723966435c3b275
S: b957d95a7804432b2fcbc3d8e012582321c1b0a6 192.168.31.204:7004
   slots: (0 slots) slave
   replicates 5d8fc7bac440fe5c2b04e17fca28d74f9a416586
M: 5d8fc7bac440fe5c2b04e17fca28d74f9a416586 192.168.31.204:7002
   slots:[5461-10922] (5462 slots) master
   1 additional replica(s)
S: f5fe2797f61ee0765a6d2d5e92b6f4f766ce80dc 192.168.31.204:7006
   slots: (0 slots) slave
   replicates 54d866707b7cd76d14893356f33d3e39867ba312
[OK] All nodes agree about slots configuration.
>>> Check for open slots...
>>> Check slots coverage...
[OK] All 16384 slots covered.
```

3.2、搭建rocketmq (单机)

3.2.1 软件准备:

```
- RocketMQ最新版本：4.5.1

- 下载地址
  - https://www.apache.org/dyn/closer.cgi?path=rocketmq/4.5.1/rocketmq-all-4.5.1-
bin-release.zip
```

3.2.2 环境要求

```
- JDK 11.0.5

- Linux64位系统(CentOS Linux release 7.7.1908)

- 源码安装需要安装Maven 3.2.x

- 4G+ free
```

3.2.3 安装及启动

3.2.3.1 下载rocketmq

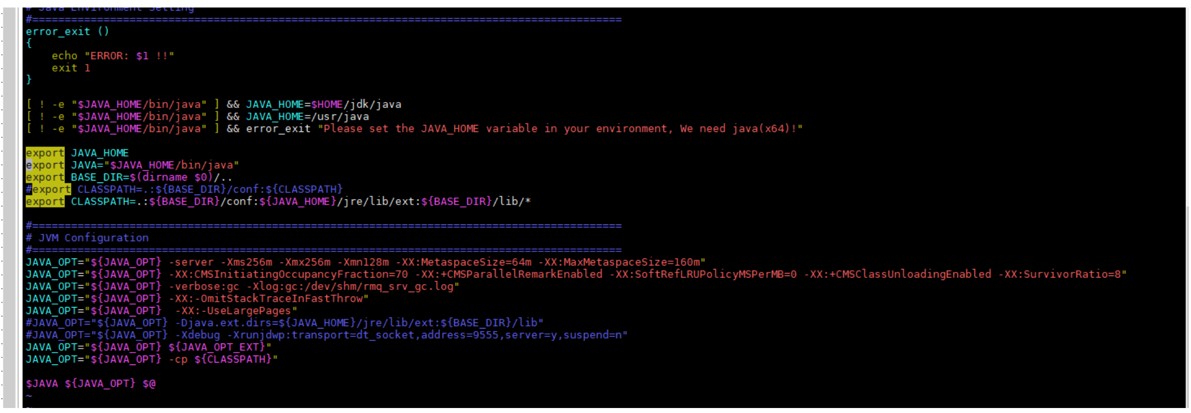
```
- 下载

- wget https://archive.apache.org/dist/rocketmq/4.5.1/rocketmq-all-

- 4.5.1-bin-release.zip
```

3.2.3.2 修改脚本

3.2.3.2.1 runserver.sh

```
bin/runserver.sh
```

修改后



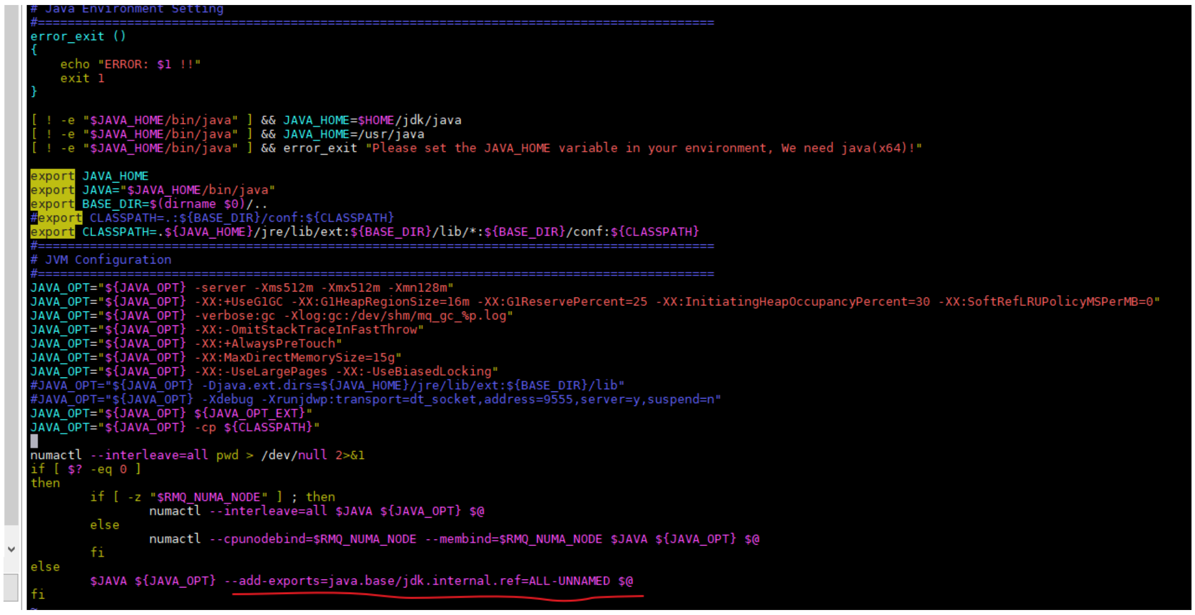
```
# Java Environment Setting
#===========================================================================================
error_exit ()
{
    echo "ERROR: $1 !!"
    exit 1
}

[ ! -e "$JAVA_HOME/bin/java" ] && JAVA_HOME=$HOME/jdk/java
[ ! -e "$JAVA_HOME/bin/java" ] && JAVA_HOME=/usr/java
[ ! -e "$JAVA_HOME/bin/java" ] && error_exit "Please set the JAVA_HOME variable in your environment, We need java(x64)!"

export JAVA_HOME
export JAVA="$JAVA_HOME/bin/java"
export BASE_DIR=$(dirname $0)/..
#export CLASSPATH=.:${BASE_DIR}/conf:${CLASSPATH}
export CLASSPATH=.:${BASE_DIR}/conf:${JAVA_HOME}/jre/lib/ext:${BASE_DIR}/lib/*

#===========================================================================================
# JVM Configuration
#===========================================================================================
JAVA_OPT="${JAVA_OPT} -server -Xms256m -Xmx256m -Xmn128m -XX:MetaspaceSize=64m -XX:MaxMetaspaceSize=160m"
JAVA_OPT="${JAVA_OPT} -XX:CMSInitiatingOccupancyFraction=70 -XX:+CMSParallelRemarkEnabled -XX:SoftRefLRUPolicyMSPerMB=0 -XX:+CMSClassUnloadingEnabled -XX:SurvivorRatio=8"
JAVA_OPT="${JAVA_OPT} -verbose:gc -Xlog:gc:/dev/shm/rmq_srv_gc.log"
JAVA_OPT="${JAVA_OPT} -XX:-OmitStackTraceInFastThrow"
JAVA_OPT="${JAVA_OPT}  -XX:-UseLargePages"
#JAVA_OPT="${JAVA_OPT} -Djava.ext.dirs=${JAVA_HOME}/jre/lib/ext:${BASE_DIR}/lib"
#JAVA_OPT="${JAVA_OPT} -Xdebug -Xrunjdwp:transport=dt_socket,address=9555,server=y,suspend=n"
JAVA_OPT="${JAVA_OPT} ${JAVA_OPT_EXT}"
JAVA_OPT="${JAVA_OPT} -cp ${CLASSPATH}"

$JAVA ${JAVA_OPT} $@
~
```

```
- vim bin/runserver.sh

- 删除

- UseCMSCompactAtFullCollection

- UseParNewGC

- UseConcMarkSweepGC

- 修改内存：

- JAVA_OPT="${JAVA_OPT} -server -Xms256m -Xmx256m -Xmn128m -

- XX:MetaspaceSize=64mm -XX:MaxMetaspaceSize=160mm"

- -Xloggc修改为-Xlog:gc
```

3.2.3.2.2 runbroker.sh

```
bin/runbroker.sh
```

修改后



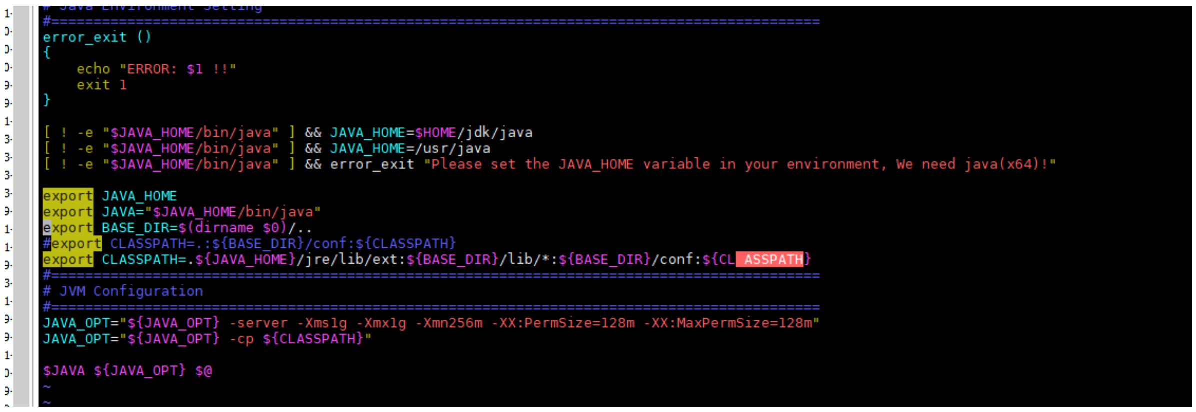
```
# Java Environment Setting
#===========================================================================================
error_exit ()
{
    echo "ERROR: $1 !!"
    exit 1
}

[ ! -e "$JAVA_HOME/bin/java" ] && JAVA_HOME=$HOME/jdk/java
[ ! -e "$JAVA_HOME/bin/java" ] && JAVA_HOME=/usr/java
[ ! -e "$JAVA_HOME/bin/java" ] && error_exit "Please set the JAVA_HOME variable in your environment, We need java(x64)!"

export JAVA_HOME
export JAVA="$JAVA_HOME/bin/java"
export BASE_DIR=$(dirname $0)/..
#export CLASSPATH=.:${BASE_DIR}/conf:${CLASSPATH}
export CLASSPATH=.${JAVA_HOME}/jre/lib/ext:${BASE_DIR}/lib/*:${BASE_DIR}/conf:${CLASSPATH}
#===========================================================================================
# JVM Configuration
#===========================================================================================
JAVA_OPT="${JAVA_OPT} -server -Xms512m -Xmx512m -Xmn128m"
JAVA_OPT="${JAVA_OPT} -XX:+UseG1GC -XX:G1HeapRegionSize=16m -XX:G1ReservePercent=25 -XX:InitiatingHeapOccupancyPercent=30 -XX:SoftRefLRUPolicyMSPerMB=0"
JAVA_OPT="${JAVA_OPT} -verbose:gc -Xlog:gc:/dev/shm/mq_gc_%p.log"
JAVA_OPT="${JAVA_OPT} -XX:-OmitStackTraceInFastThrow"
JAVA_OPT="${JAVA_OPT} -XX:+AlwaysPreTouch"
JAVA_OPT="${JAVA_OPT} -XX:MaxDirectMemorySize=15g"
JAVA_OPT="${JAVA_OPT} -XX:-UseLargePages -XX:-UseBiasedLocking"
#JAVA_OPT="${JAVA_OPT} -Djava.ext.dirs=${JAVA_HOME}/jre/lib/ext:${BASE_DIR}/lib"
#JAVA_OPT="${JAVA_OPT} -Xdebug -Xrunjdwp:transport=dt_socket,address=9555,server=y,suspend=n"
JAVA_OPT="${JAVA_OPT} ${JAVA_OPT_EXT}"
JAVA_OPT="${JAVA_OPT} -cp ${CLASSPATH}"

numactl --interleave=all pwd > /dev/null 2>&1
if [ $? -eq 0 ]
then
        if [ -z "$RMQ_NUMA_NODE" ] ; then
                numactl --interleave=all $JAVA ${JAVA_OPT} $@
        else
                numactl --cpunodebind=$RMQ_NUMA_NODE --membind=$RMQ_NUMA_NODE $JAVA ${JAVA_OPT} $@
        fi
else
        $JAVA ${JAVA_OPT} --add-exports=java.base/jdk.internal.ref=ALL-UNNAMED $@
fi
```

```
- vim bin/runbroker.sh

- 删除：

- PrintGCDateStamps

- PrintGCApplicationStoppedTime

- PrintAdaptiveSizePolicy

- UseGCLogFileRotation

- NumberOfGCLogFiles=5

- GCLogFileSize=30m
```

3.2.3.2.3 tools.sh

```
bin/tools.sh
```

修改后

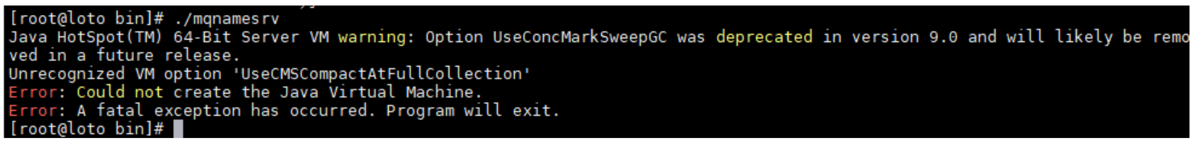


```
# Java Environment Setting
#===========================================================================================
error_exit ()
{
    echo "ERROR: $1 !!"
    exit 1
}

[ ! -e "$JAVA_HOME/bin/java" ] && JAVA_HOME=$HOME/jdk/java
[ ! -e "$JAVA_HOME/bin/java" ] && JAVA_HOME=/usr/java
[ ! -e "$JAVA_HOME/bin/java" ] && error_exit "Please set the JAVA_HOME variable in your environment, We need java(x64)!"

export JAVA_HOME
export JAVA="$JAVA_HOME/bin/java"
export BASE_DIR=$(dirname $0)/..
#export CLASSPATH=.:${BASE_DIR}/conf:${CLASSPATH}
export CLASSPATH=.${JAVA_HOME}/jre/lib/ext:${BASE_DIR}/lib/*:${BASE_DIR}/conf:${CLASSPATH}
#===========================================================================================
# JVM Configuration
#===========================================================================================
JAVA_OPT="${JAVA_OPT} -server -Xms1g -Xmx1g -Xmn256m -XX:PermSize=128m -XX:MaxPermSize=128m"
JAVA_OPT="${JAVA_OPT} -cp ${CLASSPATH}"

$JAVA ${JAVA_OPT} $@
~
~
```

```
- vim bin/tools.sh

- \# 删除 JAVA_OPT="${JAVA_OPT} -

- Djava.ext.dirs=${BASE_DIR}/lib:${JAVA_HOME}/jre/lib/ext"
```

启动NameServer

```
#启动NameServer
mqnamesrv
#查看启动日志
tail -f ~/logs/rocketmqlogs/namesrv.lo
```



```
[root@loto bin]# ./mqnamesrv
Java HotSpot(TM) 64-Bit Server VM warning: Option UseConcMarkSweepGC was deprecated in version 9.0 and will likely be remo
ved in a future release.
Unrecognized VM option 'UseCMSCompactAtFullCollection'
Error: Could not create the Java Virtual Machine.
Error: A fatal exception has occurred. Program will exit.
[root@loto bin]# 
```

启动Broker

```
# 启动Broker
mqbroker -n localhost:9876
#查看启动日志
- tail -f ~/logs/rocketmqlogs/broker.log
```

3.3、安装mysql 数据库

3.3.0 、添加 MySQL 组和 MySQL 用户

```
添加 MySQL 组和 MySQL 用户用于设置 MySQL 安装目录文件所有者和所属组groupadd mysqluseradd
-r -g mysql mysql
```

3.3.1 下载 MySQL

```
wget http://dev.MySQL.com/get/Downloads/MySQL-5.7/mysql-5.7.11-Linux-glibc2.5-
x86_64.tar.gz
```

3.3.2 解压

```
解压二进制文件tar -zxvf mysql-5.7.11-Linux-glibc2.5-x86_64.tar.gz
```

3.3.3 目录及权限设置

```
#更改 MySQL 目录名称
mv mysql-5.7.11-linux-glibc2.5-x86_64 mysql
#移动 MySQL 到/usr/local目录下
mv mysql /usr/local
#更改 MySQL 目录所属的组和用户，更改权限
cd /usr/local/mysql   chgrp -R mysql .
```

3.3.4 初始密码

```
初始化 MySQL 配置表mkdir datayum -y install libaio.so.*bin/mysqld --initialize --
user=mysql --basedir=/usr/local/mysql --datadir=/usr/local/mysql/data
运行完上面3行命令后，此处会生成 MySQL 的密码，记下来   yr!k8WQ.fs+Q
```

3.3.5 权限

```
将 MySQL 目录下除了data目录的所有文件，改回root用户所有，MySQL 用户只需作为 mysql/data/ 目
录下所有文件的所有者  chown -R root .chown -R mysql data
```

3.3.6 配置文件

```
# 复制配置文件
cp support-files/my-default.cnf /etc/my.cnf
#修改my.cnf关键配置
1）进入配置文件vim /etc/my.cnf
2）编写my.cnf，输入i进入编辑状态
basedir = /usr/local/mysqldatadir = /usr/local/mysql/dataport = 3306socket =
/tmp/mysql.sock
3）按下esc后，输入：wq 退出vim编辑器
```

3.3.7 tmp目录

```
#创建tmp目录，然后赋予 MySQL 权限
mkdir tmpchown -R mysql:mysql tmp
```

3.3.8 开机启动项

```
chkconfig --list mysql#将mysqld服务加入开机自启动项
cp support-files/mysql.server /etc/init.d/mysqlchmod +x /etc/init.d/mysql
# 把 MySQL 注册为开机启动的服务
chkconfig --add mysql
# 查看是否添加成功
chkconfig --list mysql
#显示结果为：
#mysql       0:off  1:off  2:on  3:on  4:on  5:on  6:off
```

3.3.9、MySQL 服务的开启和关闭

```
# MySQL 服务的开启和关闭
service mysql startservice mysql stopservice mysql restart
```

3.3.10、配置环境变量

```
（1）进入profilevim /etc/profile
（2）按i编辑放在最后一行：
PATH=$PATH:/usr/local/mysql:/usr/local/mysql/binexport PATH
（3）退出vim之后，使其修改生效source /etc/profile
（4）执行完可通过命令查看是否添加成功echo $PATH
```

3.3.11 、登录 MySQL 服务,更改密码

```
# 登录 MySQL 服务
mysql -uroot -p输入原始密码
#mysql命令登录不成功，报错：mysql: error while loading shared libraries: 
libncurses.so.5: cannot open shared object file: No such file or directory。按照百
度的方法都不成功，应该和系统版本有关，后来自己想到一个方法：yum install libncurses*，完美解
决问题。
# 登录 MySQL 服务centos8 安装mysql5.7 后更改密码：
set password for root@localhost = password('123');
use mysqlupdate user set host='%' where user='root';
# 使修改生效flush privileges;
```

3.3.12、添加一个新用户

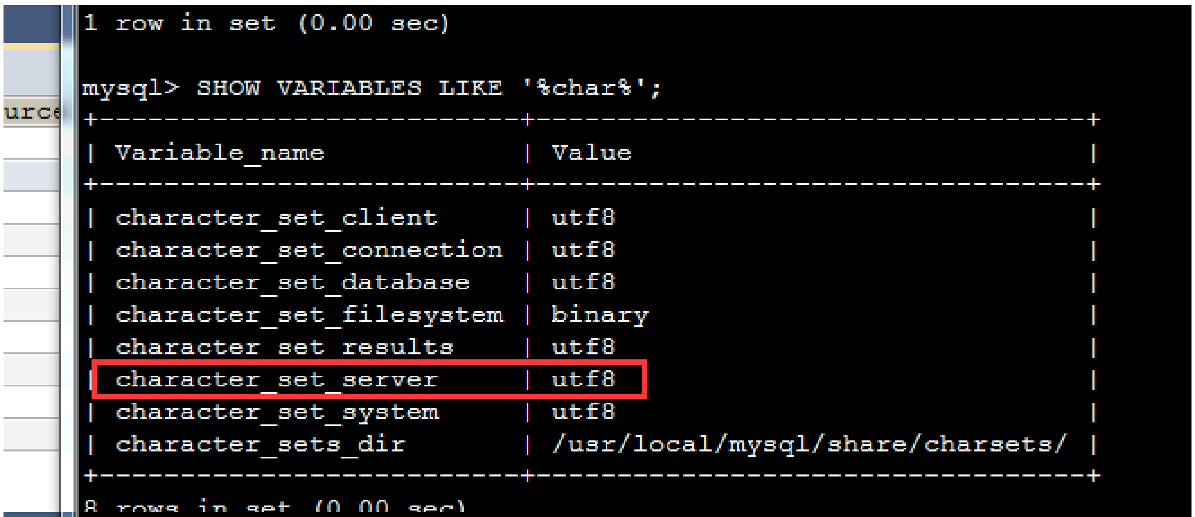
```
# 添加一个新用户
grant all privileges on *.* to 'td'@'%' identified by 'td';
use mysql;
select host,user from user;+-----------+-----------+| host     | user     |+--------
---+-----------+| %        | td       || localhost | mysql.sys || localhost | root     
|+-----------+-----------+
退出mysql：quit
```

3.3.13 登录后操作

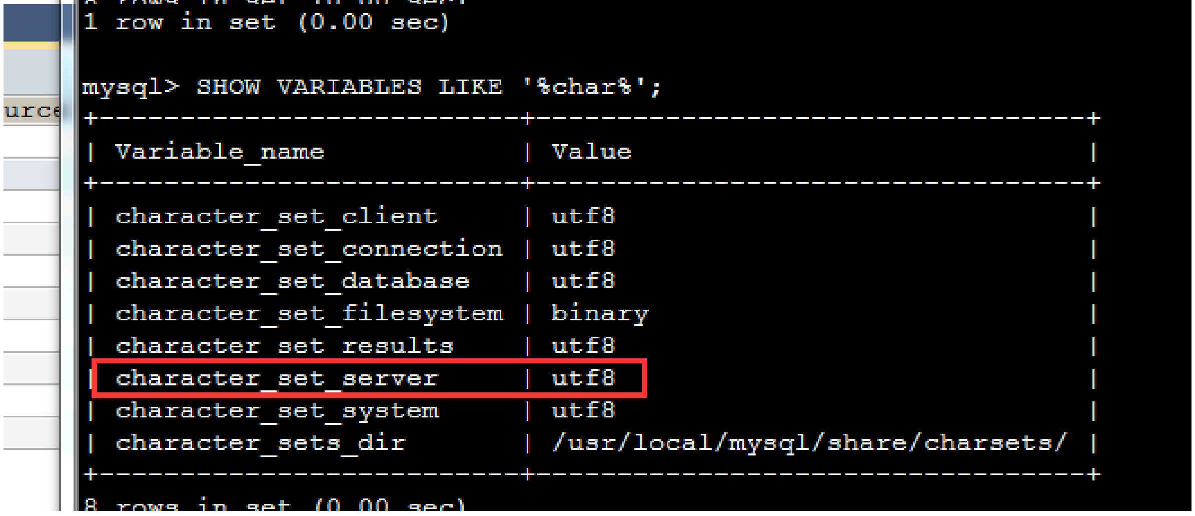
3.3.13.1 使用Navicat登录

已创建mysql用户用户名：td 密码：td

SHOW VARIABLES LIKE '%char%';

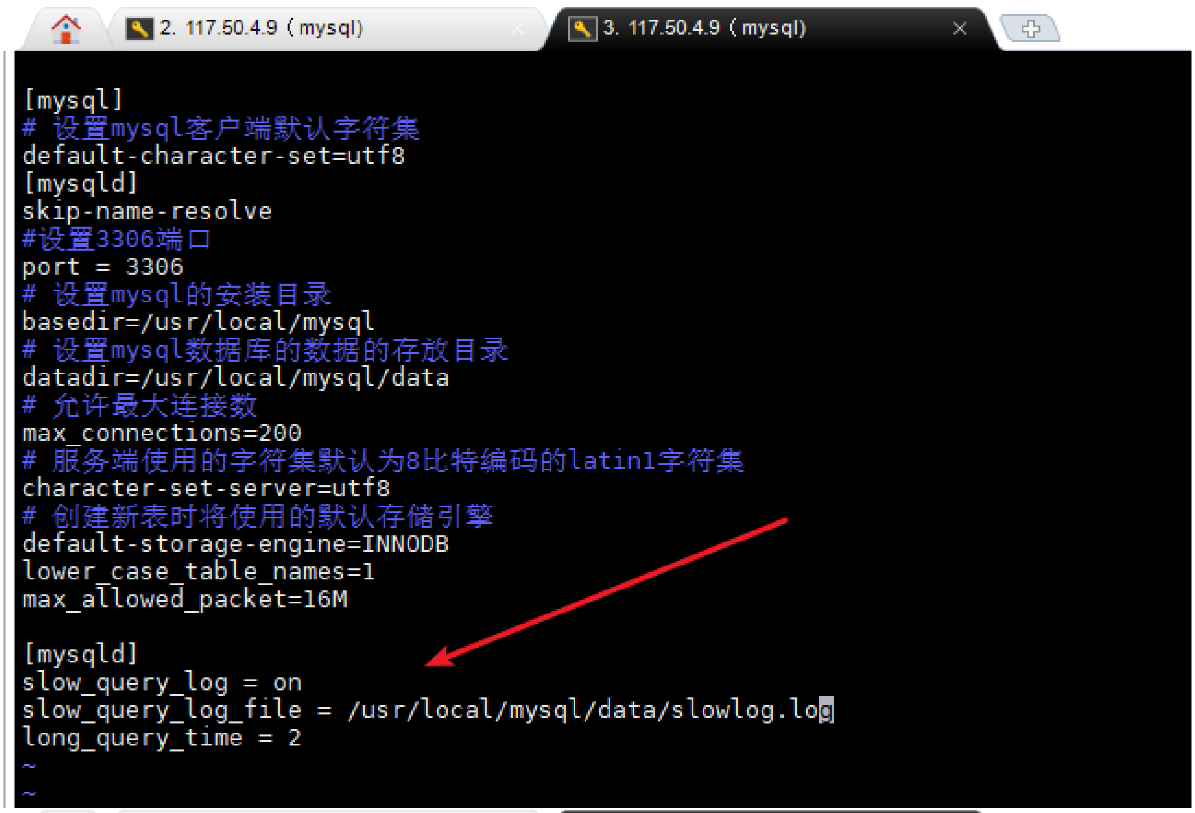


```
1 row in set (0.00 sec)

mysql> SHOW VARIABLES LIKE '%char%';
+--------------------------+-------------------------------------+
| Variable_name            | Value                               |
+--------------------------+-------------------------------------+
| character_set_client     | utf8                                |
| character_set_connection | utf8                                |
| character_set_database   | utf8                                |
| character_set_filesystem | binary                              |
| character_set_results    | utf8                                |
| character_set_server     | utf8                                |
| character_set_system     | utf8                                |
| character_sets_dir       | /usr/local/mysql/share/charsets/    |
+--------------------------+-------------------------------------+

8 rows in set (0.00 sec)
```

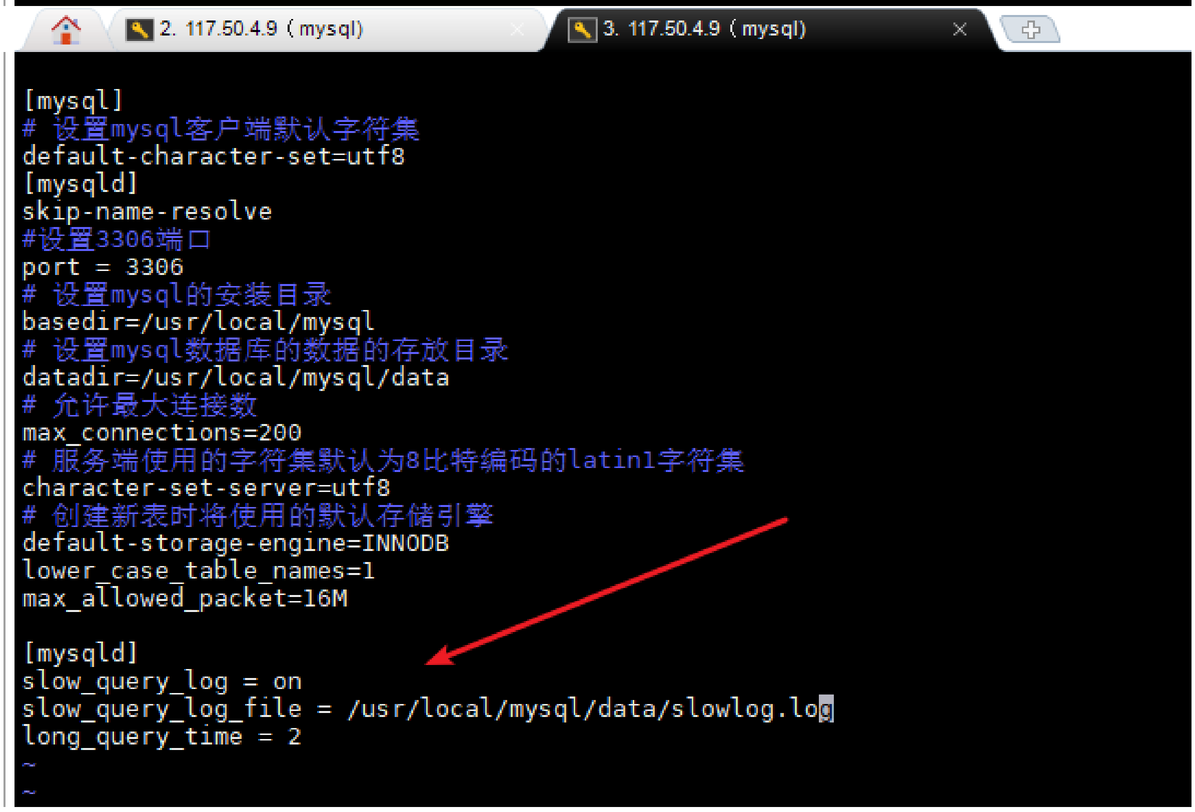


```
1 row in set (0.00 sec)

mysql> SHOW VARIABLES LIKE '%char%';
+--------------------------+-------------------------------------+
| Variable_name            | Value                               |
+--------------------------+-------------------------------------+
| character_set_client     | utf8                                |
| character_set_connection | utf8                                |
| character_set_database   | utf8                                |
| character_set_filesystem | binary                              |
| character_set_results    | utf8                                |
| character_set_server     | utf8                                |
| character_set_system     | utf8                                |
| character_sets_dir       | /usr/local/mysql/share/charsets/    |
+--------------------------+-------------------------------------+

8 rows in set (0.00 sec)
```

3.3.13.2 mysql 5.7 开启慢查询日志



```
[mysql]
# 设置mysql客户端默认字符集
default-character-set=utf8
[mysqld]
skip-name-resolve
#设置3306端口
port = 3306
# 设置mysql的安装目录
basedir=/usr/local/mysql
# 设置mysql数据库的数据的存放目录
datadir=/usr/local/mysql/data
# 允许最大连接数
max_connections=200
# 服务端使用的字符集默认为8比特编码的latin1字符集
character-set-server=utf8
# 创建新表时将使用的默认存储引擎
default-storage-engine=INNODB
lower_case_table_names=1
max_allowed_packet=16M

[mysqld]
slow_query_log = on
slow_query_log_file = /usr/local/mysql/data/slowlog.log
long_query_time = 2
~
~
```



```
[mysql]
# 设置mysql客户端默认字符集
default-character-set=utf8
[mysqld]
skip-name-resolve
#设置3306端口
port = 3306
# 设置mysql的安装目录
basedir=/usr/local/mysql
# 设置mysql数据库的数据的存放目录
datadir=/usr/local/mysql/data
# 允许最大连接数
max_connections=200
# 服务端使用的字符集默认为8比特编码的latin1字符集
character-set-server=utf8
# 创建新表时将使用的默认存储引擎
default-storage-engine=INNODB
lower_case_table_names=1
max_allowed_packet=16M

[mysqld]
slow_query_log = on
slow_query_log_file = /usr/local/mysql/data/slowlog.log
long_query_time = 2
~
~
```

[mysqld]slow_query_log = onslow_query_log_file = /usr/local/mysql/data/slowlog.loglong_query_time = 2 注：前面必须加[mysqld] 我这边配置前加了[mysql]导致读取报错。mysql: [ERROR] unknown variable 'slow_query_log=on'

四、编码实现

4.1 pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```xml
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.4.5</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.idstaa.rocket.demo</groupId>
    <artifactId>flash-sale</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>flash-sale</name>
    <description>Demo project for Spring Boot</description>
    <properties>
        <java.version>1.8</java.version>
        <rocketmq-spring-boot-starter-version>2.0.3</rocketmq-spring-boot-
starter-version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-redis</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-thymeleaf</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <scope>runtime</scope>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
        <!-- rocketmq -->
        <dependency>
            <groupId>org.apache.rocketmq</groupId>
            <artifactId>rocketmq-spring-boot-starter</artifactId>
            <version>${rocketmq-spring-boot-starter-version}</version>
        </dependency>

        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <optional>true</optional>
        </dependency>

        <dependency>
```

```
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>5.1.49</version>
            <scope>runtime</scope>
        </dependency>

        <dependency>
            <groupId>tk.mybatis</groupId>
            <artifactId>mapper-spring-boot-starter</artifactId>
            <version>2.1.5</version>
        </dependency>

        <!-- https://mvnrepository.com/artifact/com.alibaba/druid -->
        <dependency>
            <groupId>com.alibaba</groupId>
            <artifactId>druid</artifactId>
            <version>1.1.10</version>
        </dependency>

        <dependency>
            <groupId>com.github.pagehelper</groupId>
            <artifactId>pagehelper-spring-boot-starter</artifactId>
            <version>1.3.0</version>
            <exclusions>
                <exclusion>
                    <artifactId>mybatis</artifactId>
                    <groupId>org.mybatis</groupId>
                </exclusion>
                <exclusion>
                    <artifactId>mybatis-spring</artifactId>
                    <groupId>org.mybatis</groupId>
                </exclusion>
            </exclusions>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
```

redis + rocketmq + mybatis + 通用mapper + pagehelper +mysql + druid + lombok + thymeleaf

4.2 application

```
spring.application.name=springboot_rocketmq_producer

spring.redis.host=node1
spring.redis.port=6379
spring.redis.database=2
```

```
# rocketmq的nameserver的地址
rocketmq.name-server=node1:9876
rocketmq.producer.retry-next-server=true
rocketmq.producer.retry-times-when-send-async-failed=2
rocketmq.producer.group=producer_grp_02

# MySQL数据库连接配置
spring.datasource.type=com.alibaba.druid.pool.DruidDataSource
spring.datasource.url=jdbc:mysql://localhost:3306/db_flashsale?
useUnicode=true&characterEncoding=utf8&serverTimezone=UTC
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.datasource.username=root
spring.datasource.password=root

# thymeleaf 配置
spring.thymeleaf.prefix=classpath:/templates/
spring.thymeleaf.suffix=.html
spring.web.resources.static-locations=classpath:/templates/,classpath:/static/
spring.mvc.static-path-pattern=/**
spring.thymeleaf.cache=false

#json时间
spring.jackson.date-format=yyyy-MM-dd HH:mm:ss

#mybatis配置
mybatis.mapper-locations=classpath:mappers/*.xml
#开启驼峰命名匹配映射
mybatis.configuration.map-underscore-to-camel-case=true
spring.autoconfigure.exclude=org.springframework.boot.autoconfigure.jdbc.DataSou
rceAutoConfiguration
mybatis.configuration.log-impl=org.apache.ibatis.logging.stdout.StdOutImpl


# mybatis通用mapper 配置
mybatis.type-aliases-package=tk.mybatis.springboot.model
mapper.not-empty=false
mapper.identity=MYSQL

#pagehelper：四要素配置
#https://github.com/pagehelper/Mybatis-PageHelper
#指定数据库分页类型
pagehelper.helperDialect=mysql

#页码<=0 查询第一页，页码>=总页数查询最后一页
pagehelper.reasonable=true

#支持通过 Mapper 接口参数来传递分页参数：
#https://github.com/pagehelper/Mybatis-
PageHelper/blob/master/wikis/zh/HowToUse.md
pagehelper.supportMethodsArguments=true

pagehelper.params=count=countSql
```

4.3 config

```
@Configuration
public class MybatisConfig {
    @Autowired
    private DataSourceProperties dataSourceProperties;

    @Bean(name = "dataSource")
    public DataSource dataSource() {
        DruidDataSource dataSource = new DruidDataSource();
        dataSource.setUrl(dataSourceProperties.getUrl());
        System.out.println(dataSourceProperties.getUrl());

 dataSource.setDriverClassName(dataSourceProperties.getDriverClassName());
        dataSource.setUsername(dataSourceProperties.getUsername());
        dataSource.setPassword(dataSourceProperties.getPassword());
        return dataSource;
    }
}

@Configuration
public class RedisConfig {
    @Autowired
    RedisConnectionFactory factory;

    @Bean
    public RedisTemplate<String, Integer> redisTemplate(){
        RedisTemplate<String, Integer> redisTemplate=new RedisTemplate<>();

        redisTemplate.setKeySerializer(new StringRedisSerializer());
        GenericJackson2JsonRedisSerializer jackson2JsonRedisSerializer = new
GenericJackson2JsonRedisSerializer();
        redisTemplate.setValueSerializer(jackson2JsonRedisSerializer);
        redisTemplate.setHashKeySerializer(new StringRedisSerializer());
        redisTemplate.setHashValueSerializer(new StringRedisSerializer());

        redisTemplate.setConnectionFactory(factory);

        return redisTemplate;
    }
}
```

4.4 实体类

```
@Data
@Table(name="tb_order")
public class IdstaaOrder {

    @Id
    private String orderId;
    private String userId;
    private Integer orderStatus;
    private String content;
}
```

```java
public enum OrderStatus {
    NEW(0),PAYED(1),CANCEL(2);
    private int status;

    OrderStatus(int status){
        this.status = status;
    }
    public  int getStatus(){
        return this.status;
    }
}
```

4.5 控制器

OrderController

```java
@RestController
public class OrderController {
    @Autowired
    private HttpServletRequest request;

    @Autowired
    private OrderService orderService;

    @Autowired
    private RedisTemplate<String, Integer> redisTemplate;

    /**
     * 秒杀抢购
     * @param order
     * @return
     * @throws Exception
     */
    @GetMapping("/flash")
    public AppResult<IdstaaOrder> order(IdstaaOrder order) throws Exception {
        order.setOrderId(UUID.randomUUID().toString());
        order.setOrderStatus(OrderStatus.NEW.getStatus());
        String result = orderService.order(order);
        if(result.equals("fail")){
            return new ReturnWrapper<IdstaaOrder>(order).fail(result);
        }else {
            return new ReturnWrapper<IdstaaOrder>(order).success(result);
        }

    }

    /**
     * 初始化秒杀商品数量（测试用）
     * @param amount
     * @return
     * @throws Exception
     */
    @GetMapping("/initProductAmount/{amount}")
    public AppResult<String> initProductAmount(@PathVariable Integer amount)
throws Exception {
        redisTemplate.opsForValue().set("goods_count_init", amount);
        redisTemplate.opsForValue().set("goods_count", amount);
        return new ReturnWrapper<String>().success("success");
    }

    /**
     * 支付
     * @param orderId
     * @return
     * @throws Exception
     */
    @GetMapping("/pay/{orderId}")
    public AppResult<String> pay(@PathVariable String orderId) throws Exception
{
        IdstaaOrder order = new IdstaaOrder();
        order.setOrderId(orderId);
        order.setOrderStatus(OrderStatus.PAYED.getStatus());
        int i = orderService.updateByPrimaryKeySelective(order);
        if(i>0){
            return new ReturnWrapper<String>().success("支付成功");
        }else {
            return new ReturnWrapper<String>().fail("fail");
        }
    }
}
```

PageController

```
@Controller
public class PageController {
    @Autowired
    private HttpServletRequest request;

    @Autowired
    private RedisTemplate<String, Integer> redisTemplate;

    @Autowired
    private OrderService orderService;

    @GetMapping(value = {"/index","/"})

    public String orderPayView(IdstaaOrder order){
        Integer goods_count_init =
redisTemplate.opsForValue().get("goods_count_init");
        Integer goods_count = redisTemplate.opsForValue().get("goods_count");
        if (goods_count == null) {
            redisTemplate.opsForValue().set("goods_count_init", 10);
            redisTemplate.opsForValue().set("goods_count", 10);
            goods_count = 10;
        }
        int amount_percent = goods_count*100/goods_count_init;
        request.setAttribute("amount_percent", amount_percent+"%");
        request.setAttribute("current_count", goods_count);
        return "index";
    }
```

```java
    @GetMapping(value = "waitingPayView/{orderId}")
    public String waitingPayView(@PathVariable String orderId){
        request.setAttribute("orderId", orderId);
        return "waitingPayView";
    }
}
```

4.6 业务处理

```java
@Service
public class OrderServiceImpl implements OrderService {
    @Autowired
    private RedisTemplate<String, Integer> redisTemplate;

    @Autowired
    private RocketMQTemplate rocketMQTemplate;

    @Autowired
    private OrderMapper orderMapper;

    @Override
    public String order(IdstaaOrder order) {
        Integer goods_count = redisTemplate.opsForValue().get("goods_count");
        if (goods_count <= 0) return "fail";
        rocketMQTemplate.asyncSend("tp_flashSale2", order, new SendCallback() {
            @Override
            public void onSuccess(SendResult sendResult) {
                redisTemplate.opsForValue().set("goods_count", goods_count - 1);
                System.out.println("发送消息到秒杀队列"+order);
                order.setOrderStatus(OrderStatus.NEW.getStatus());
            }

            @Override
            public void onException(Throwable e) {

            }
        });
        return order.getOrderStatus() != null ? "success" : "fail";
    }

    @Override
    public int updateByPrimaryKeySelective(IdstaaOrder order) {
        IdstaaOrder queryOrder = new IdstaaOrder();
        queryOrder.setOrderId(order.getOrderId());
        IdstaaOrder select = orderMapper.selectOne(queryOrder);
        if (select != null &&
OrderStatus.PAYED.getStatus()!=select.getOrderStatus()) {
            return orderMapper.updateByPrimaryKeySelective(order);
        }else {
            return 0;
        }
    }

    @Override
    public int insertSelective(IdstaaOrder order) {
        return orderMapper.insertSelective(order);
    }

}
```

4.7 消费端rocketmq 监听器

秒杀消息监听器

```
@Slf4j
@Component
@RocketMQMessageListener(topic = "tp_flashSale2", consumerGroup =
"consumer_grp_03")
public class RocketMQOrderListen implements RocketMQListener<IdstaaOrder> {
    @Autowired
    private RocketMQTemplate rocketMQTemplate;

    @Autowired
    private OrderMapper orderMapper;
    @Override
    public void onMessage(IdstaaOrder order) {
        System.out.println("订单状态: "+ order.getOrderStatus());
        System.out.println("向数据库插入记录-service-"+order);
        orderMapper.insertSelective(order);
        // 延迟支付消息
        Message message = new Message("to_DelayPay2",
order.getOrderId().getBytes());
        org.springframework.messaging.Message springMessage =
RocketMQUtil.convertToSpringMessage(message);
        rocketMQTemplate.asyncSend(message.getTopic(), springMessage, new
SendCallback() {
            @Override
            public void onSuccess(SendResult sendResult) {
                System.out.println("发送消息到延迟支付队列");
            }

            @Override
            public void onException(Throwable e) {

            }
        },3000,3);
    }
}
```

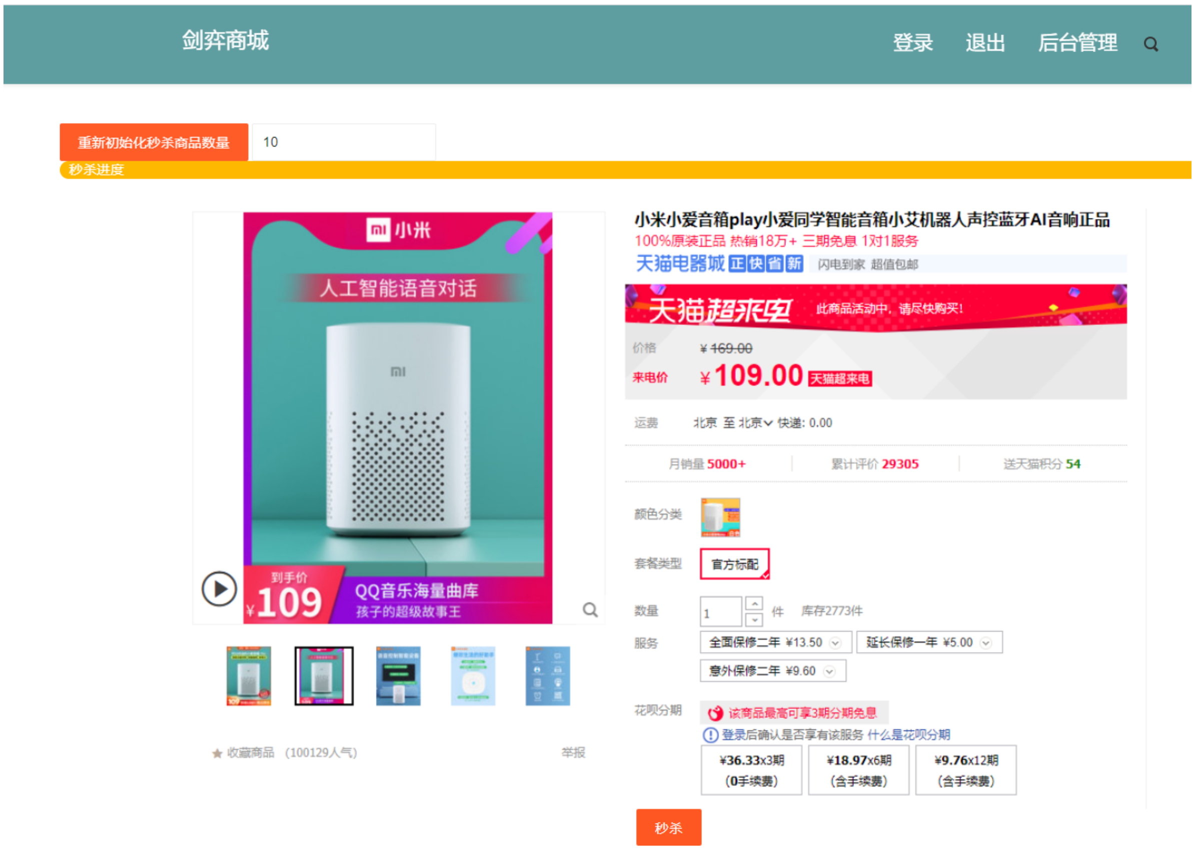
支付延迟取消消息监听器

```
@Slf4j
@Component
@RocketMQMessageListener(topic = "to_DelayPay2", consumerGroup =
"consumer_grp_02")
public class RocketMQPayListener implements RocketMQListener<String> {
    @Autowired
    private RedisTemplate<String, Integer> redisTemplate;

    @Autowired
    private OrderService orderService;

    @Override
    public void onMessage(String message) {
        IdstaaOrder order = new IdstaaOrder();
        order.setOrderId(message);
        // 若超时，则取消订单
        order.setOrderStatus(OrderStatus.CANCEL.getStatus());
        int i = orderService.updateByPrimaryKeySelective(order);
        if (i >= 0) {
            redisTemplate.opsForValue().increment("goods_count");
            System.out.println(message+"订单已取消。。。");
        }

        log.info("{}订单支付状态更改", order.getOrderId());
    }
}
```

五、测试及效果演示

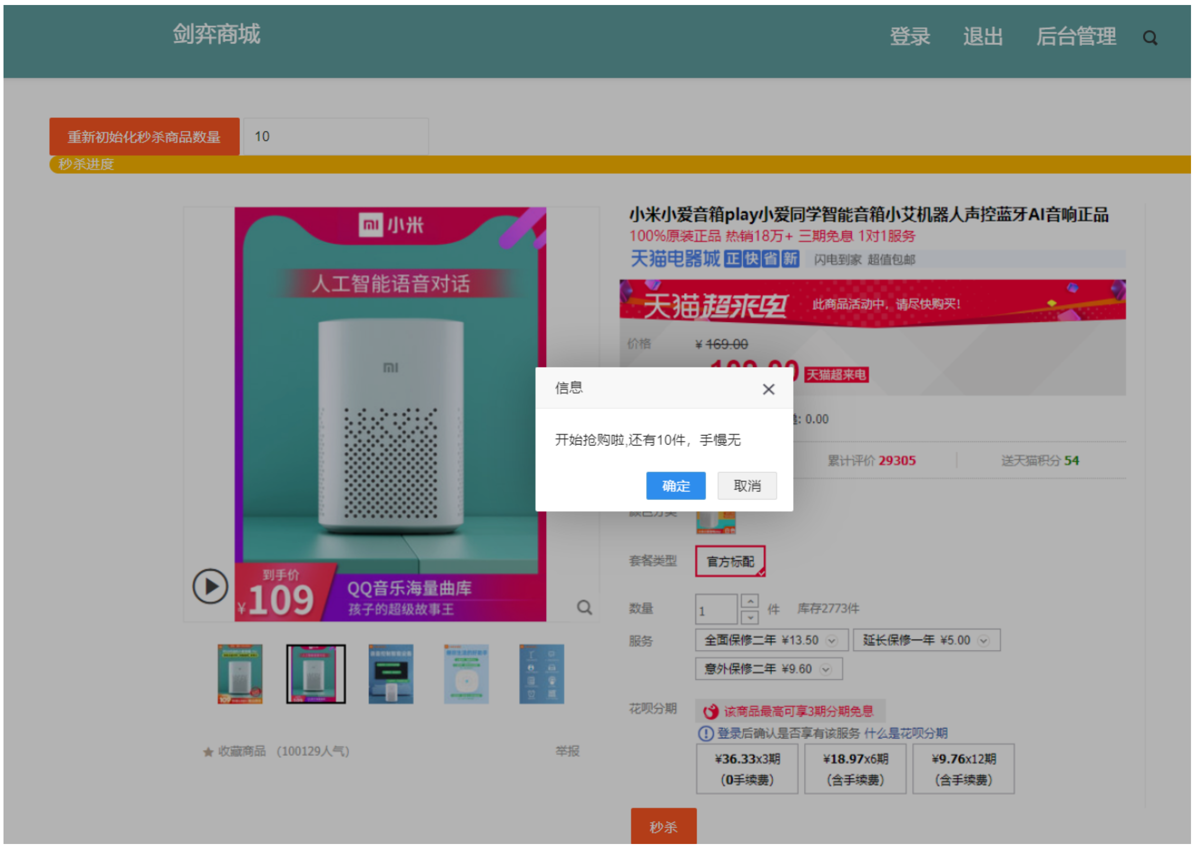
5.1 index 商详页面

http://localhost:8080/



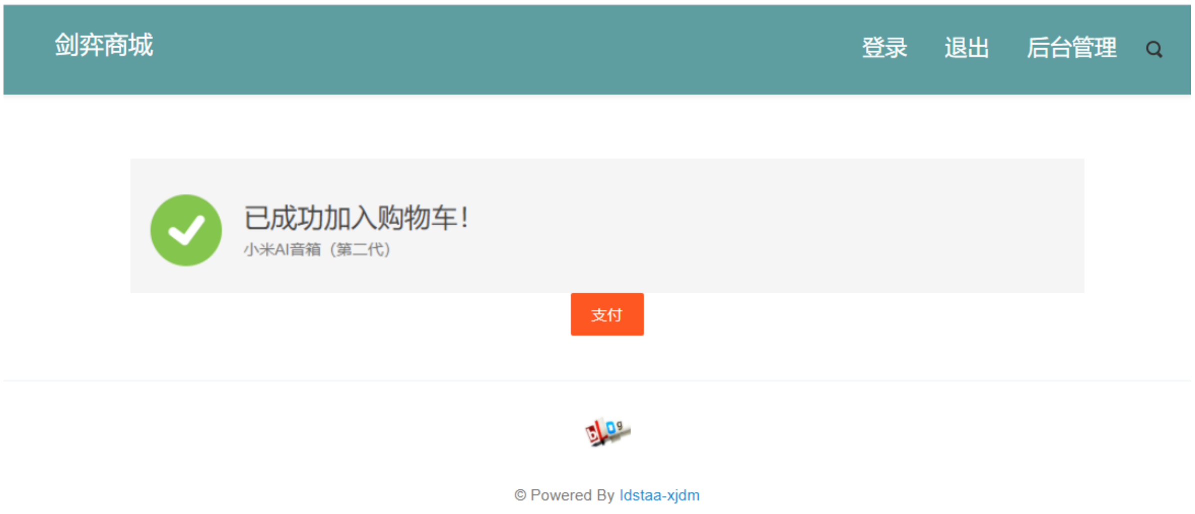
小米小爱音箱play小爱同学智能音箱小艾机器人声控蓝牙AI音响正品
100%原装正品 热销18万+ 三期免息 1对1服务
天猫电器城 正 快 省 新 闪电到家 超值包邮
天猫超来电 此商品活动中，请尽快购买！
价格 ¥ ~~169.00~~
来电价 ¥109.00 天猫超来电
运费 北京 至 北京 快递: 0.00
月销量 5000+ | 累计评价 29305 | 送天猫积分 54
颜色分类
套餐类型 官方标配
数量 1 件 库存2773件
服务 全面保修二年 ¥13.50 | 延长保修一年 ¥5.00 | 意外保修二年 ¥9.60
花呗分期 该商品最高可享3期分期免息
登录后确认是否享有该服务 什么是花呗分期
¥36.33x3期 (0手续费) | ¥18.97x6期 (含手续费) | ¥9.76x12期 (含手续费)

5.2 秒杀



开始抢购啦,还有10件，手慢无

5.3 等待支付页面



已成功加入购物车!

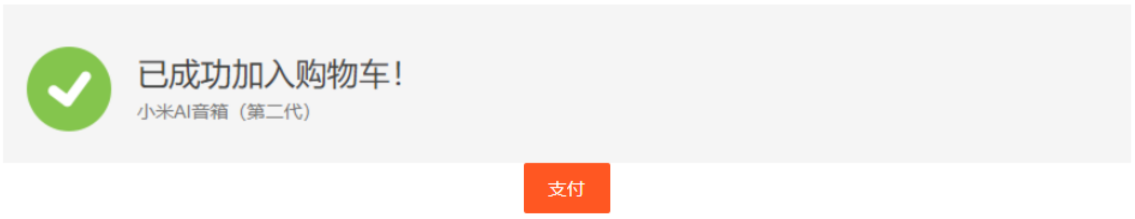
小米AI音箱（第二代）

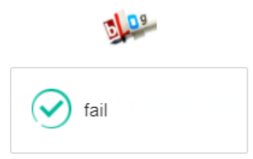
© Powered By Idstaa-xjdm

5.4 10s 内不支付

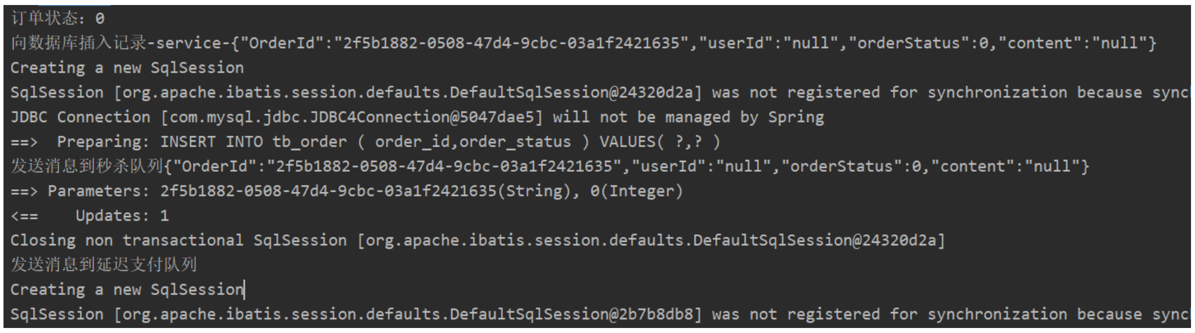
http://localhost:8080/waitingPayView/2f5b1882-0508-47d4-9cbc-03a1f2421635

订单id （2f5b1882-0508-47d4-9cbc-03a1f2421635）

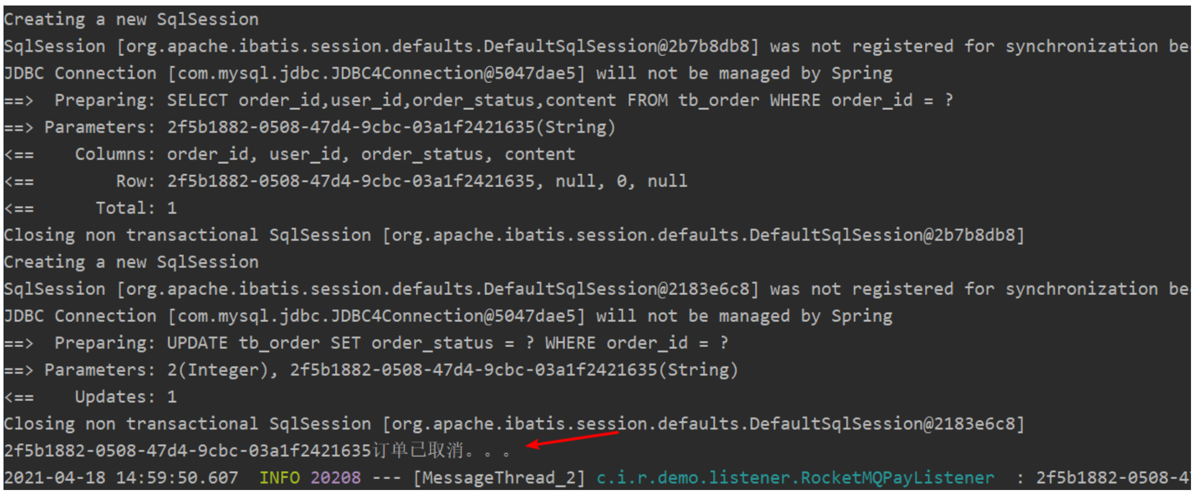




查看运行日志

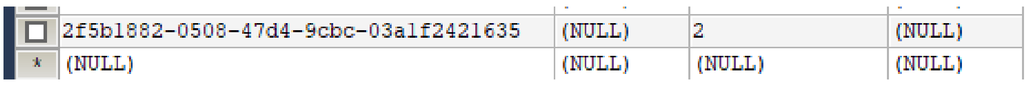


```
订单状态: 0
向数据库插入记录-service-{"OrderId":"2f5b1882-0508-47d4-9cbc-03a1f2421635","userId":"null","orderStatus":0,"content":"null"}
Creating a new SqlSession
SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@24320d2a] was not registered for synchronization because sync
JDBC Connection [com.mysql.jdbc.JDBC4Connection@5047dae5] will not be managed by Spring
==>  Preparing: INSERT INTO tb_order ( order_id,order_status ) VALUES( ?,? )
发送消息到秒杀队列{"OrderId":"2f5b1882-0508-47d4-9cbc-03a1f2421635","userId":"null","orderStatus":0,"content":"null"}
==> Parameters: 2f5b1882-0508-47d4-9cbc-03a1f2421635(String), 0(Integer)
<==    Updates: 1
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@24320d2a]
发送消息到延迟支付队列
Creating a new SqlSession
SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@2b7b8db8] was not registered for synchronization because sync
```



```
Creating a new SqlSession
SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@2b7b8db8] was not registered for synchronization be
JDBC Connection [com.mysql.jdbc.JDBC4Connection@5047dae5] will not be managed by Spring
==>  Preparing: SELECT order_id,user_id,order_status,content FROM tb_order WHERE order_id = ?
==> Parameters: 2f5b1882-0508-47d4-9cbc-03a1f2421635(String)
<==    Columns: order_id, user_id, order_status, content
<==        Row: 2f5b1882-0508-47d4-9cbc-03a1f2421635, null, 0, null
<==      Total: 1
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@2b7b8db8]
Creating a new SqlSession
SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@2183e6c8] was not registered for synchronization be
JDBC Connection [com.mysql.jdbc.JDBC4Connection@5047dae5] will not be managed by Spring
==>  Preparing: UPDATE tb_order SET order_status = ? WHERE order_id = ?
==> Parameters: 2(Integer), 2f5b1882-0508-47d4-9cbc-03a1f2421635(String)
<==    Updates: 1
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@2183e6c8]
2f5b1882-0508-47d4-9cbc-03a1f2421635订单已取消。。。
2021-04-18 14:59:50.607  INFO 20208 --- [MessageThread_2] c.i.r.demo.listener.RocketMQPayListener   : 2f5b1882-0508-4
```

数据库数据状态查看，订单状态为2，已取消

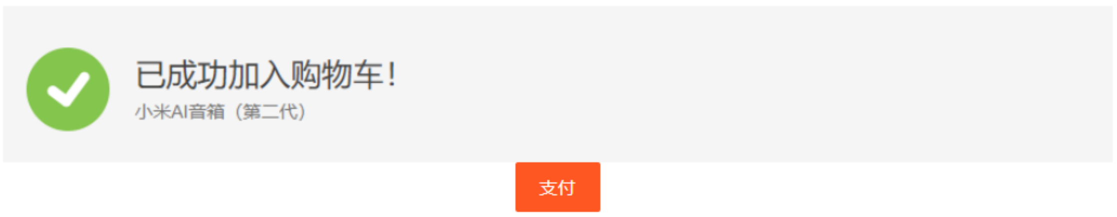


☐	2f5b1882-0508-47d4-9cbc-03a1f2421635	(NULL)	2	(NULL)
*	(NULL)	(NULL)	(NULL)	(NULL)

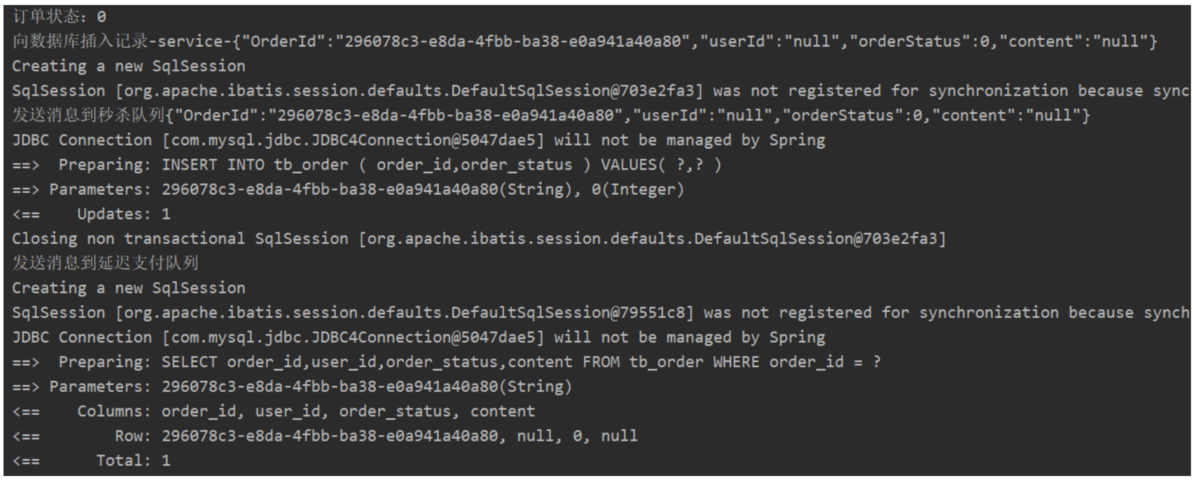
5.5 10s 内支付

http://localhost:8080/waitingPayView/296078c3-e8da-4fbb-ba38-e0a941a40a80

订单ID： 296078c3-e8da-4fbb-ba38-e0a941a40a80

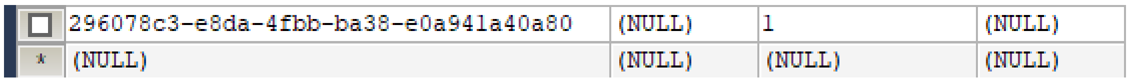


后台日志查看



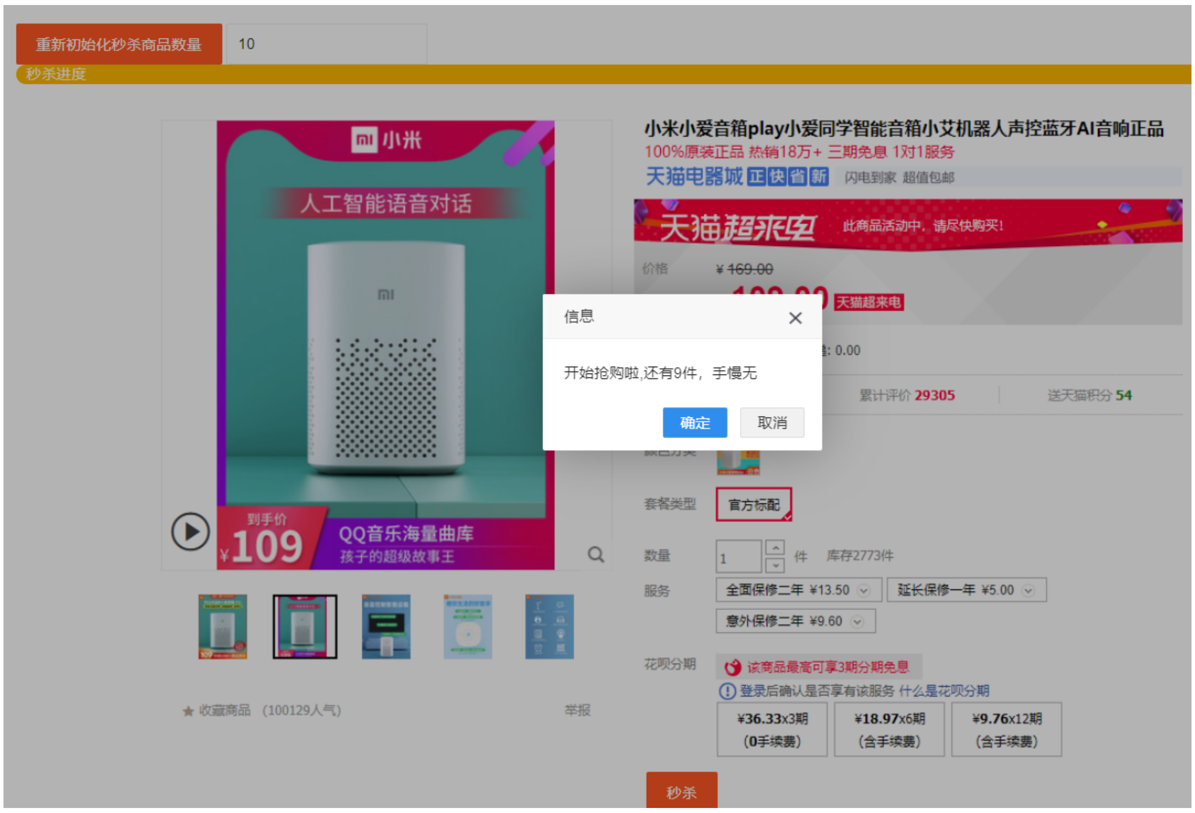
```
订单状态: 0
向数据库插入记录-service-{"OrderId":"296078c3-e8da-4fbb-ba38-e0a941a40a80","userId":"null","orderStatus":0,"content":"null"}
Creating a new SqlSession
SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@703e2fa3] was not registered for synchronization because sync
发送消息到秒杀队列{"OrderId":"296078c3-e8da-4fbb-ba38-e0a941a40a80","userId":"null","orderStatus":0,"content":"null"}
JDBC Connection [com.mysql.jdbc.JDBC4Connection@5047dae5] will not be managed by Spring
==>  Preparing: INSERT INTO tb_order ( order_id,order_status ) VALUES( ?,? )
==> Parameters: 296078c3-e8da-4fbb-ba38-e0a941a40a80(String), 0(Integer)
<==    Updates: 1
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@703e2fa3]
发送消息到延迟支付队列
Creating a new SqlSession
SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@79551c8] was not registered for synchronization because synch
JDBC Connection [com.mysql.jdbc.JDBC4Connection@5047dae5] will not be managed by Spring
==>  Preparing: SELECT order_id,user_id,order_status,content FROM tb_order WHERE order_id = ?
==> Parameters: 296078c3-e8da-4fbb-ba38-e0a941a40a80(String)
<==    Columns: order_id, user_id, order_status, content
<==        Row: 296078c3-e8da-4fbb-ba38-e0a941a40a80, null, 0, null
<==      Total: 1
```

数据库状态



☐	296078c3-e8da-4fbb-ba38-e0a941a40a80	(NULL)	1	(NULL)
*	(NULL)	(NULL)	(NULL)	(NULL)

再次查看redis 缓存


重新初始化秒杀商品数量
10
秒杀进度
小米小爱音箱play小爱同学智能音箱小艾机器人声控蓝牙AI音响正品
100%原装正品 热销18万+ 三期免息 1对1服务
天猫电器城 正快省新 闪电到家 超值包邮
天猫超来电 此商品活动中，请尽快购买!
价格 ¥469.00
天猫超来电
累计评价 29305
送天猫积分 54
套餐类型 官方标配
数量 1 件 库存2773件
服务 全面保修二年 ¥13.50 延长保修一年 ¥5.00 意外保修二年 ¥9.60
花呗分期 该商品最高可享3期分期免息
登录后确认是否享有该服务 什么是花呗分期
¥36.33x3期 (0手续费)
¥18.97x6期 (含手续费)
¥9.76x12期 (含手续费)
秒杀
收藏商品 (100129人气)
举报
小米
人工智能语音对话
到手价 ¥109
QQ音乐海量曲库
孩子的超级故事王
确定
取消

信息

开始抢购啦,还有9件，手慢无

I六、补充，github 代码地址