JAVAEE REPORT3

SCHOOL OF SOFTWARE

任务要点	状态	解释
Restful	完成	基于作业 2
Kafka	完成	version 2.2.1

基本业务逻辑：

当有大量的用户同时提交申请参加健身课程的表时，直接进行数据库的操作会浪费很多时间，系统的消息队列也会很拥堵。所以我们将“返回页面”和“存入数据库”的两个操作进行了分离，让存储到数据库这个操作与返回界面异步提高系统的响应效率。

实现逻辑：

使用 kafka 作为消息的中间件，将提交上来的用户的信息作为消息的内容发布到 information 这个 topic 中去,返回界面，显示用户信息正在保存。

处理端监听“information”这个 topic，听到信息后，将信息解码之后存储到数据库中去。

实现代码：

（1）引入 spring-kafka 依赖（pom.xml）

```
<dependency>
    <groupId>org.apache.kafka</groupId>
    <artifactId>kafka-streams</artifactId>
</dependency>

<dependency>
    <groupId>org.springframework.kafka</groupId>
    <artifactId>spring-kafka</artifactId>
</dependency>
```

（2）配置 spring-kafka（application.properties）

```
#Kafka Topic
message.topic.name=information

spring.kafka.bootstrap-servers=localhost:9092

#Unique String which identifies which consumer group this consumer belongs to
spring.kafka.consumer.group-id=jcg-group
```

（3）配置 Producer

```
@Bean
public ProducerFactory<String, Information> producerFactory() {
    Map<String, Object> configProps = new HashMap<>();
    configProps.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG, bootstrapAddress);
    configProps.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG, StringSerializer.class);
    configProps.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG, Serialization.class);
    return new DefaultKafkaProducerFactory<>(configProps);
}
```

自定义对象序列化

（4）配置 Consumer

```
@Bean
public ConsumerFactory<String, Information> consumerFactory() {
    Map<String, Object> props = new HashMap<>();
    props.put(
            ConsumerConfig.BOOTSTRAP_SERVERS_CONFIG, bootstrapAddress);
    props.put(
            ConsumerConfig.GROUP_ID_CONFIG, groupId);
    props.put(
            ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG, StringDeserializer.class);
    props.put(
            ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG, Deserialization.class);
    return new DefaultKafkaConsumerFactory<>(props);
}

@Bean
```

自定义对象反序列化

（5）Post 申请表

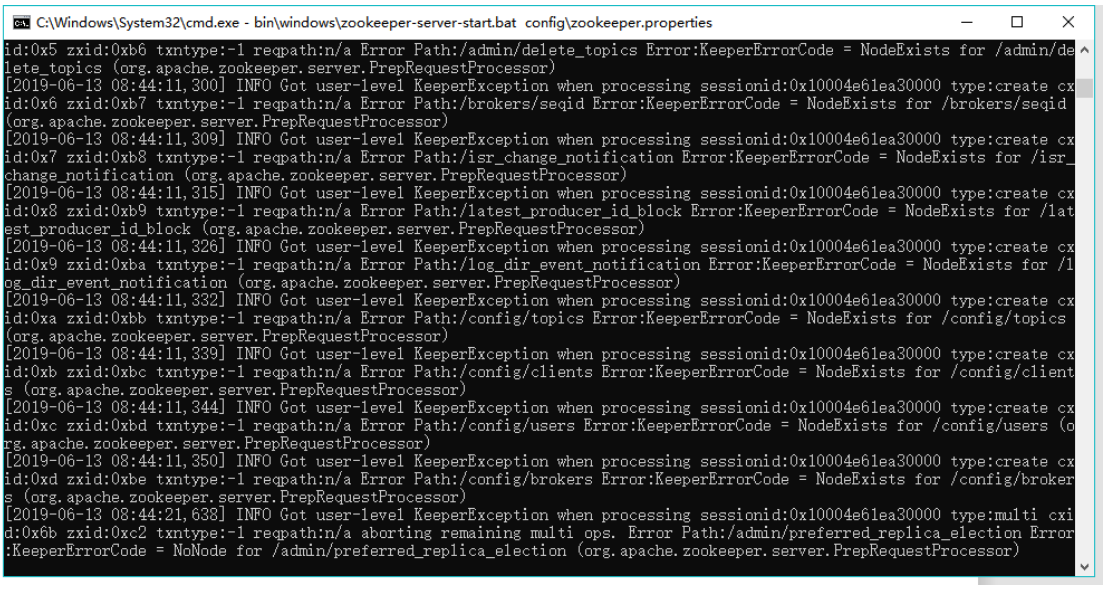
```
@RequestMapping(value = "/information", method = RequestMethod.POST)
public HttpEntity<Status> post(Information info) {
    Status status;
    try {
        kafkaTemplate.send(topicName, info);
        status = new Status( msg: "Success");
    }
    catch (Exception ex) {
        status = new Status( msg: "Failed");
    }
    return  new ResponseEntity<>(status, HttpStatus.OK);
}
```

（6）发送给 kafka 集群

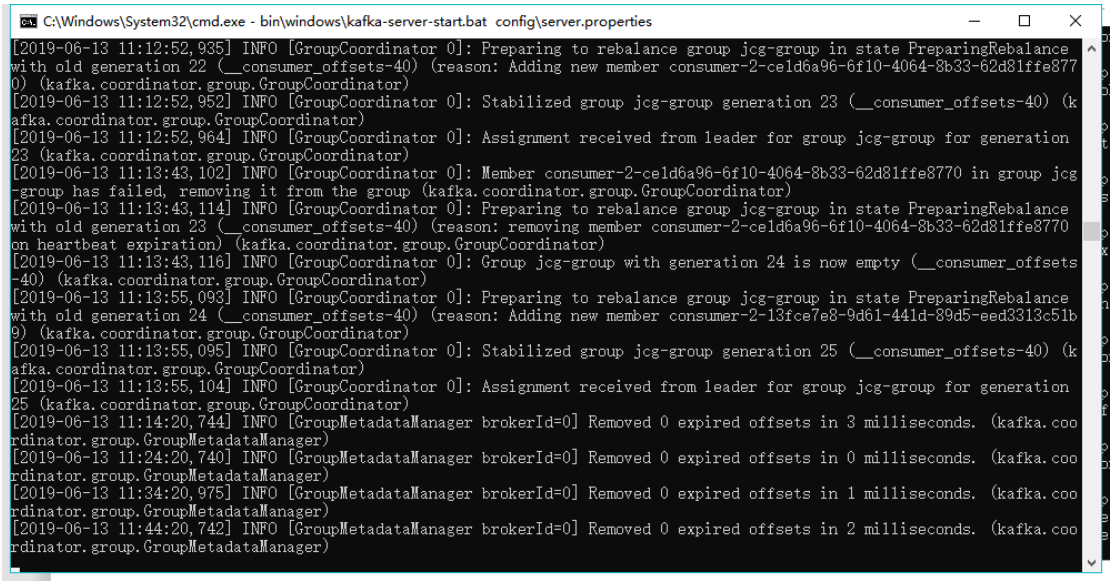
```
@KafkaListener(topics = "information", groupId = "jcg-group")
public HttpEntity<Information> listen(Information message) throws InterruptedException {
    LOG.info("Received message in JCG group: {}", message.toString());
    Thread.sleep( millis: 6000);
    service.save(message);
    return  new ResponseEntity<Information>(message, HttpStatus.OK);
}
```

运行情况

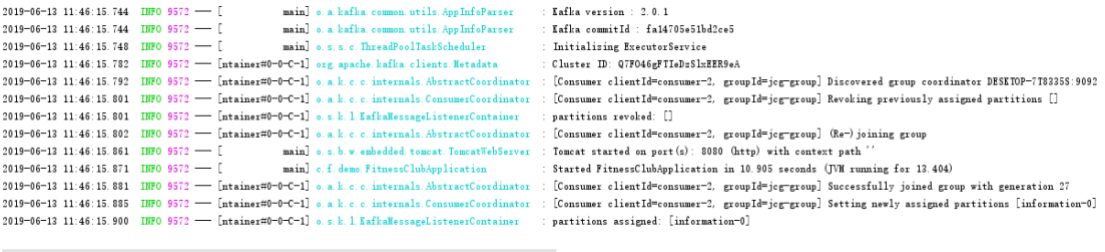
（1）运行 zookeeper



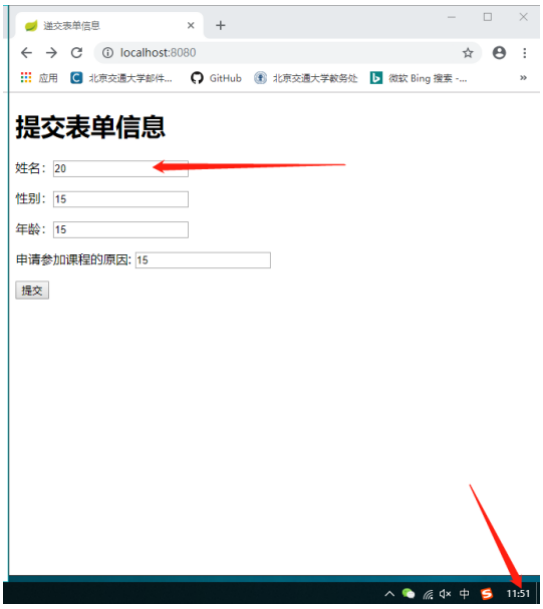
（2）运行 kafka



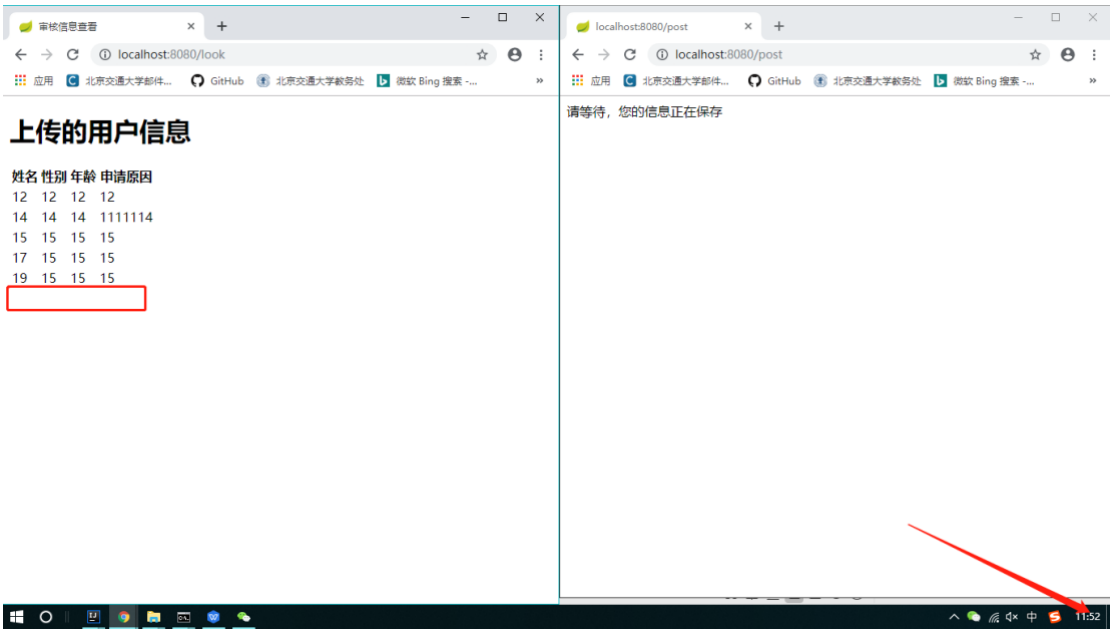
（3）运行 spring-boot 程序

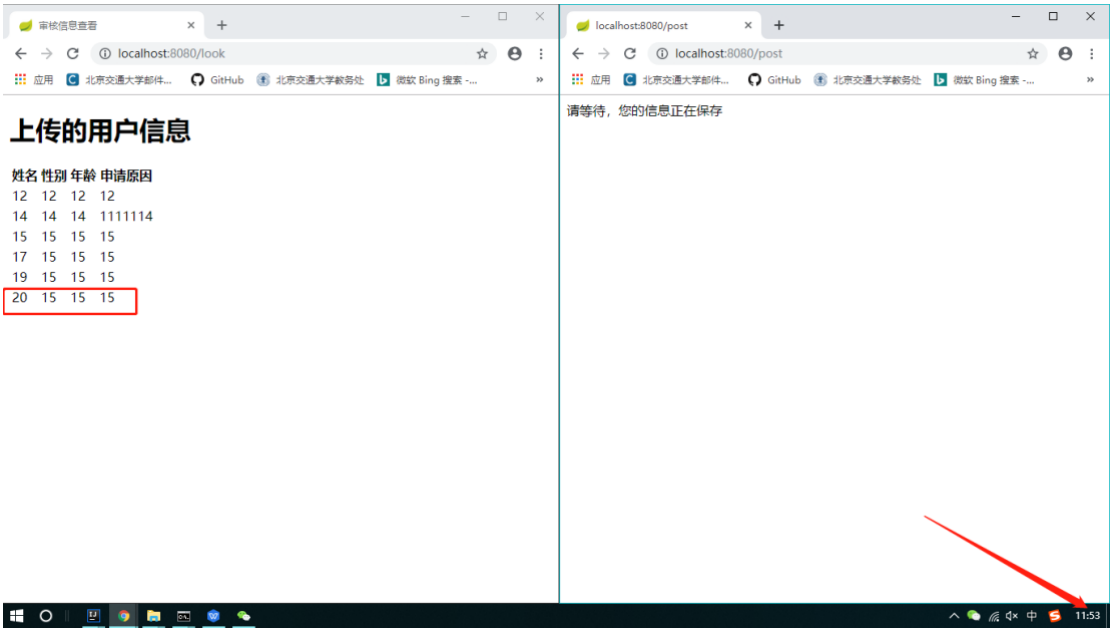


（4）提交信息


提交表单信息
localhost:8080
提交表单信息
姓名: 20
性别: 15
年龄: 15
申请参加课程的原因: 15
提交

（5）查看结果


localhost:8080/look
上传的用户信息
姓名 性别 年龄 申请原因
12 12 12 12
14 14 14 1111114
15 15 15 15
17 15 15 15
19 15 15 15
localhost:8080/post
请等待，您的信息正在保存


审核信息查看
localhost:8080/look
应用
北京交通大学邮件...
GitHub
北京交通大学教务处
微软 Bing 搜索 -...
上传的用户信息
姓名 性别 年龄 申请原因
12 12 12 12
14 14 14 1111114
15 15 15 15
17 15 15 15
19 15 15 15
20 15 15 15
localhost:8080/post
localhost:8080/post
应用
北京交通大学邮件...
GitHub
北京交通大学教务处
微软 Bing 搜索 -...
请等待，您的信息正在保存
11:53