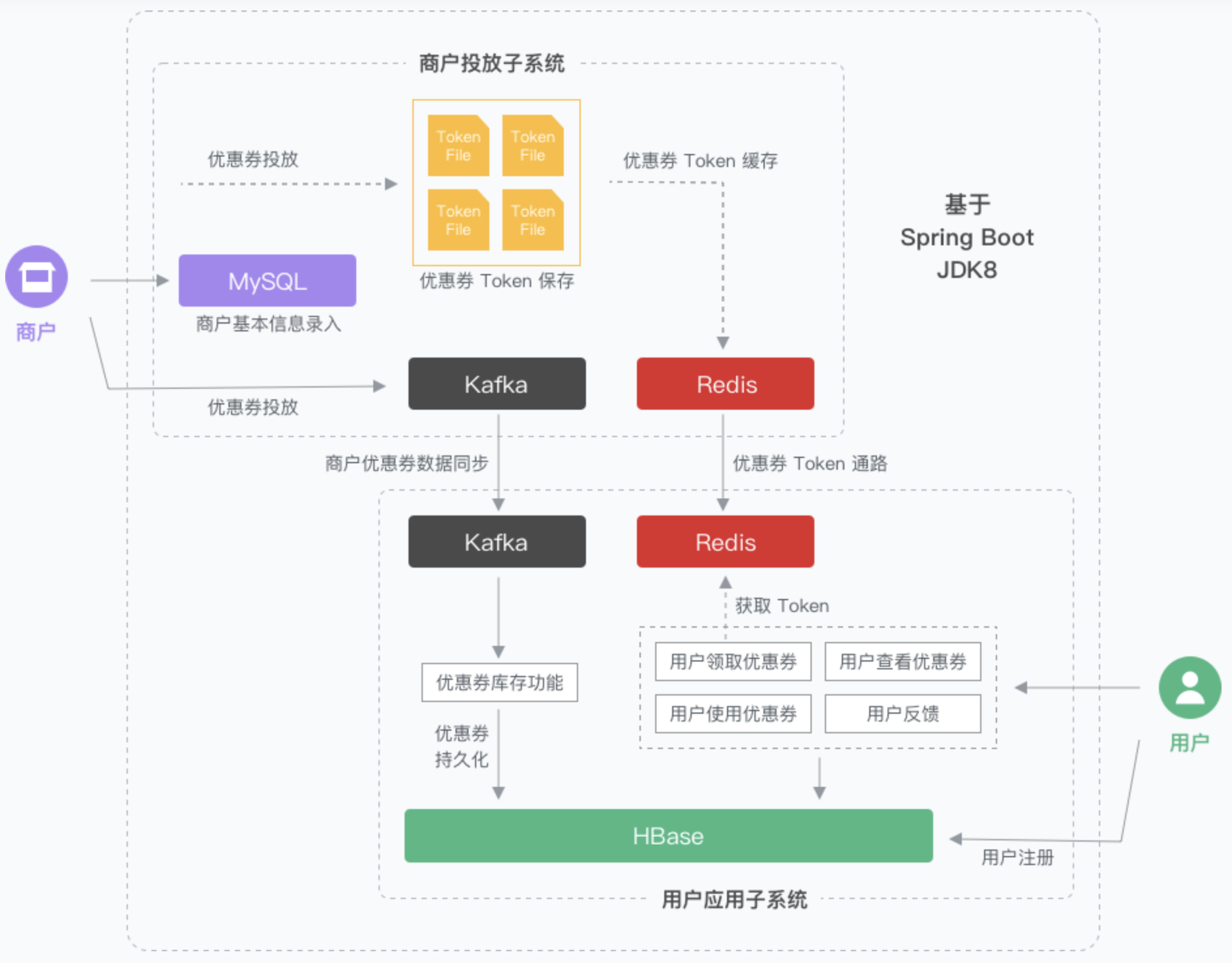
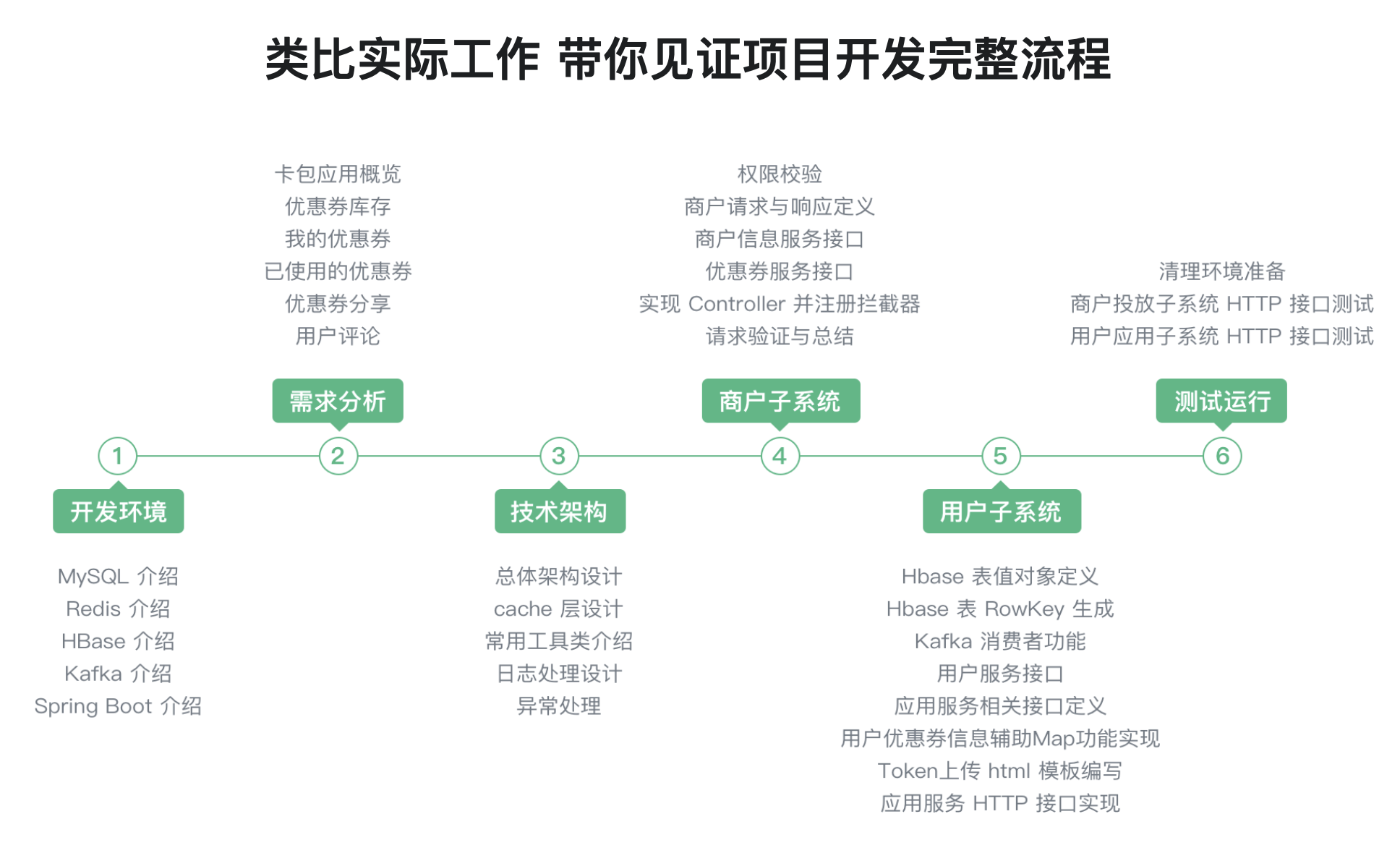
**优惠券系统笔记：**

2024.3.5





分布式优惠卷系统后台 2020.07 – 2020.09  
组长 南京  
1)本项目基于Spring Boot实现优惠劵系统后台，主要分为商户投放优惠劵子系统和用户应用子系统，主要功能  
有商户投放优惠劵，用户领取优惠劵，用户消费优惠劵，用户反馈。利用到技术栈包括:MySQL，HBase，Kafka，  
Redis等。  
2)项目中主要工作包括:  
• 实现商户创建和用户基本信息录入功能并使数据持久化，MySQL存储商户基本信息，HBase存储注册用户信  
息，商户投放的优惠劵信息，用户领取的优惠劵信息以及用户反馈信息;  
• 实现商户投放优惠劵，用户领取优惠券以及消费优惠券功能，使用Kafka消息队列解决商户投放时的高并发  
请求，使用Redis存储优惠券 token，增加用户领取优惠券的响应速度，使用分布式锁和乐观锁解决优惠劵超  
卖的问题  
• 实现日志处理，方便分析系统状态。  
3)本项目成果是实现了优惠券系统后台，并利用Postman成功通过商户投放子系统HTTP接口测试和用户应用子  
系统 HTTP接口测试，并利用JMeter对系统进行了压力测试，系统表现良好

Merchants项目结构：

constant：常量定义包，所有代码的常量定义1

vo：值对象，表明了service之间的值传递，通过哪些对象传递可以实现我们的服务。

entity：merchants商户在平台上的定义

dao：数据库对象到我们java对象的OR映射

security：利用拦截器对controller的HTTP请求做校验

service：最重要的层，首先是接口定义，再之后对接口定义进行一一的实现

controller：实现HTTP请求

最后使用POSTMAN验证了这三个方法的正确性。

依赖作用：

Thymeleaf：将web相关的html语言的解析，把返回的值定位到web的页面。

Passbookd的项目结构：

constant：常量定义，可以枚举的东西，不会变的东西

controller：代表web工程，需要对外提供HTTP的接口，一个controller提供一类服务

dao：数据库操作相关 CIUD基本操作

entity：实体类，映射到mysql数据的表

mapper：hbase表记录到java对象的映射，及ORM对象关系映射，用到spring-boot-starter-hbase。之前是mysql表各个列到java对象各个属性值的映射

log：日志的封装

service：是一个服务性的东西，为了实现服务封装的类型的东西，接口是抽象定义的功能，可以编写不同的实现类来实现接口实现定制化的功能。

utils：工具类，常用时间相关的，文件处理类等等public static

vo: 值对象定义，应用在服务之间请求对象和响应对象的传递。

resource文件：表示是一个maven工程，放一些资源文件，配置。

test文件夹：编写测试类。

Bug：

fstat unimplemented unsupported or native support failed to load; see https://github.com/jruby/jruby/wiki/Native-Libraries

initialize at org/jruby/RubyIO.java:1015

open at org/jruby/RubyIO.java:1156

似乎是 macos [系统](https://so.csdn.net/so/search?q=%E7%B3%BB%E7%BB%9F&spm=1001.2101.3001.7020)与 Hbase 不兼容的问题，下载 Hbase3.0.0 版本，将原本 Hbase 路径下的[hbase-root-dir]/lib/ruby下的 jar 包用 jruby-complete-9.3.4.0.jar 换掉：

Nmdnmdnmd 按照网上的下载HBase还要配置JAVA\_HOME，还要修改什么xml文件的，

草了，我下载完不用配置就能运行，

按照它们的配置还运行不了。

Bug: kafka重启报错：每次需要删除文件/tmp/kafka-logs下面的一个文件

Md,网上一些blog说需要修改hbase的rootdir 永久存储区，说tmp.dir为临时存储区，重启数据会丢失，

我试了一下hbase完全不用修改就能运行。数据也不会丢失。

究极BUG：springboot无法连接上hbase，搞了一天

原来是少了一步去掉注释的操作,但还是没有解决啊。。。。。

Passbook和merchants两个系统怎么通信：

主要通过ConsumePassTemplate这一步实现用户消费优惠券的交互，以@Component的注解标记，作为一个bean，运行时bean会注册到容器中，同时发现bean下面有一个KafkaListener的注解，代表函数receive是一个kafka的消费者，它接收到商户投放的优惠券之后，打印日志，之后进行一条消费将passTemplate对象反序列化，然后再写入到HBase里面。通过kafka的生成者和消费者实现两个服务的互通。

平台给商户投放的优惠券设置一个了token（这个在将优惠券写入hbase时就已经实现了），平台将这些TOKEN放到redis里面的服务：upload页面，选择商户id，passTemplateId（即优惠券的rowKey存储在hbase中的key），还有对应的token文件，实现上传

优点：实现卡包投放和卡包应用的工程

缺点：

1. 很多相似代码和对象的定义，需要进一步的重构，将相似代码抽象在一个模块里，形成一个jar包，以后修改只需修改这一个模块。
2. 日志分析的进一步应用，可以写一个日志分析的服务，分析用户的喜好，为了之后更好的推荐。
3. 事务的一致性mysql，hbase和redis的一致，系统的稳定，压力测试，抛出异常能否回滚，代码的稳定性。

扩展：

1. 用户编辑的功能
2. 增加优惠券的玩法，比如对于无上限的优惠券实现空投，直接添加到各个用户的优惠券列表。领取规则的限制。
3. 卡包增加属性，可以赠送等。

下一步面试需要会的：卡包工程的分布式部署。 redis分布式锁，还有Jmeter压力测试等。