# 个人介绍：

# 先生，你好，我叫王贺，来自黑龙江齐齐哈尔，今天来咱们公司面试java工程师这个行业，我从事这个行业快4年了，上一家公司是在北京银宝联公司，在海淀那块，期间也被外派到别的公司敲过项目，对于教育、金融、医疗都有过一些涉猎，最近敲得是一个优加健康的一个b2p平台，是关于保险的一个项目，也是属于咱们金融方面的，数据库这方面用的mysql和redis，对redis的哨兵机制，持久化，分布式锁等都有一定的了解，通过这几年敲得项目也总结了一些基本的经验，也是希望能够到贵公司能够进一步提升自己各方面的能力，希望我们接下来能够聊的愉快。

# 项目介绍：

# 1.优加健康网络平台(B2P)

优加健康网络平台是为保险公司及个人用户提供服务，致力于智能全流程管理师医疗网络服务，主要包含医保中心操作人员，定点医院操作人员，参保单位操作人员三类用户，提供了征缴业务，支付业务两大业务管理，来进行对我们整个项目总控，本项目针对于人员、单位、缴费、报销统计进行管理，以及对与门诊和住院的把控

【注册，登录模块】

再做注册的时候，我们根据公司提供的API文档，找到对应的发送短信的接口写入到我们的controller当中，在我们的service层中生成验证码，这个验证码使用random随机数生成，当时我们控制这个随机数为六位数这个范围，也就是大于等于100000小于等于999999，得到验证码后，将验证码存到我们的redis当中，并且设置这个验证码的过期时间，因为这个验证码并不会一直都有用，用户使用一次之后，就没有什么太大的用户了，处理好这些后，我们将用户的手机号和验证码放到一个map集合当中，然后将整个集合发送到rabbitMQ指定的队列上等待被消费，这里用rabbitTemplate这个模板调用converAndSend这个方法便可以直接塞入到指定的队列当中，我们在调用短信接口阿里大于的时候，因为我们这个是一个微服务的项目，所以我们专门写了一个工程，这个工程就是专门为了获取rabbitMQ队列上的数据调用阿里大于给我们客户发送短信用的，在这里我们写了一个消息监听类，引入@RabbitListener和@Component以及@RabbitHandler便可消费我们rabbitMQ上的消息，我们往rabbmtMQ传的时候用的MAP集合装的，获取的时候当然也就用Map集合来进行获取，我们在使用阿里云发送短信时，他就提供给我们一个短信工具类，这个短信工具来就可以自动的帮我们发送短信了，我们的短信模板，短信签名，手机号在阿里云官网就可以直接弄好了，他就可一直接给我们指定的手机发送短信，然后我们的短信监听类在使用阿里云提供的的短信工具类这些模板和签名的时候，直接使用@Autowired注入就好了，然后调用工具类里的发送短信方法，便可成功发送短信，我们用户在进行注册的时候，输入验证码后，去拿着我们验证码和我们redis缓存当中的验证码做对比，如果验证码为空和输入有误，则提示用户验证码有问题，如果正常的话则对我们的用户密码进行加密，这里加密的时候采用的是Bcrypt加密算法，我们当时也考虑过用MD5加密，但是后来发现这个东西能够被破解，而Bcrypt这个东西里面有时间戳，时间戳这个东西是几乎不可能被破解的，Bcrypt这个加密算法是Spring Security安全框架提供的，所以我们只需要引入Spring Security这个依赖便可以了，但是我们们引入这个安全框架之后，所有的地址就得被安全框架所控制了，而我们只是用单纯的加密算法，所用这个时候我们建立一个安全配置类，里面引入@Configuration 和@EnableWebSecurity这两个注解，便可配置所有的地址匿名访问，在启动类里面bean注入BCryptPasswordEncdor就可以加密了，我们再进行登陆效验密码的时候，BCryptPasswordEncdor给我们提供了两个方法，encode和matches方法，用encode方法进行加密，matches方法进行用户前台传入的密码和我们数据库加密后的密码进行比对，我们在登录的时候，选择了基于JWT的Token认证机制，客户端使用用户名和密码进行请求登陆，服务端收到请求，去验证用户名和密码，验证成功后，服务端会发送一个Token，再把这个Token发送给客户端，客户端收到Token后把他存储起来，比如经他放到cookie当中，客户端每次向服务器请求资源的时候，都会带着这个Token串，服务端收到请求后，会拿着前台传过来的Token串和我们服务端的Token串进行对比验证，成功就返回给客户请求的数据，

创建token方式首先导入JJWT依赖，设置签发时间，设置密钥，每次的token都是不一样的，因为载荷当中存在时间.

【续保模块】

用户选择要续费的保险,点击续保,会根据上期保单生成一个新的续费保单,用户开始核对保单信息,和核对完成以后,点击上传抽档信息,抽档信息就是用户近期的体检表图片,因为如果上传的是文字信息,用户可能会造假,公司搭建了一个独立的服务器来存放用户的抽当信息,同时我们遇到一个问题,用户上传抽档信息属于隐私信息,为了防止被盗用,我么采用了防盗链,防止信息被盗用,,用户上传完成后,后台由管理员审核(1到2个工作日,期间用户可以做别的事),审核完成后,调用阿里大于短信接口通知用户成功与否,成功提示用户上平台进行分配保单,失败则提示用户上平台查看失败原因,审核成功有7天有限期,用户完成剩下流程必须在7天内完成,过了有效期,保单作废.  
 保单分配:保单分为在职单和孤儿单,在职单用户输入保险经理的编号,点击分配,会自动分配到保险经理旗下的续费专员,后台有续费专员的工作量排序,默认分配到工作量最少的.孤儿单则不需要填写保险经理的信息,系统自动分配一个保险经理,且分配到其续费专员下.完成生成对应缴费单,

【保单支付模块】

常见的微信支付方式有很多种，例如付款码支付，扫码支付，App支付，小程序支付

我在这里做的是扫码支付

微信扫码支付是商户系统根据微信支付协议支付二维码，所以我们如果想要使用二维码支付方式，就必须申请公众号，这个公众号也就是我们平时所说的服务号，一般必须是企业公司，政府，媒体才能够申请，所以当时我也向公司要了营业执照等一些相关信息，接下来我们就该注册公众号了，注册这个公众号我们需要提交300块钱，一次没注册成功，下一次还要进行注册，注册的时候我们填写好资料就开始进行审核，审核的时间一般是1-5天，审核通过后，微信支付系统就会给我们一些信息，微信公众号appid，商户号，商户密钥以及sign一个数字签名，这个签名时微信官方提供给我们的一套算法生成加密信息，就是为了保证我们交易的安全性，提交给我们的信息填好后，提交，我们在线签署一个协议，签署好之后，我们就可以进行资金的交易和结算了，一般我们提前就需要准备这么多

我们在进行做这个微信支付的时候，其实微信支付开发为文档提供给我们很多个开发的API,我们也只是也只是用了两个API，这两个API都是用来对接微信平台的，也就是通过这两个API提供给我们的链接就可以把微信平台支付这个功能所需要传的参数，这两个API也就是统一下单API和是查询订单API

我在用代码操作的时候，微信平台提供给我们提供一个SDK，（可以不用）它只是帮助我们简化了代码的开发，可以更高效的使用这个微信支付功能，这个SDK现在只支持三种语言，JAVA,.NET,PHP,它能够对xml进行转换，还用sign签名算法，这个签名主要是帮助我们微信平台起到一个效验的作用，我们平台发送了一系列的参数通过这个签名进行加密后传给我们微信平台，我们微信平台对于加密后的这个进行效验，进而使我们的数据具有一定的安全性等等，所以我们调用它，可以在操纵他的时候更加简单我们具体操作的时候引入一个wxpay-sdk的一个依赖包就可以了，

**（微信支付API接口协议中包含字段nonce\_str（随机字符串），参与生成签名，随机以主要保证签名不可预测。**

**）**

因为微信平台只支持接受XML格式，所以我们这个SDK更加简化代码开发

我们主要会用到微信支付SDK的以下功能：

1. 获取随机字符串

|  |
| --- |
| WXPayUtil.*generateNonceStr*() |

1. MAP转换为XML字符串（自动添加签名）

|  |
| --- |
| WXPayUtil.*generateSignedXml*(param, partnerkey) |

1. XML字符串转换为MAP

|  |
| --- |
| WXPayUtil.*xmlToMap*(result) |

我们在具体操作java代码的时候，为了调用微信平台单独建立一个工程，我们整个微信支付流程主要是调用微信开发的一个对外的接口，我们主要也就是调用统一下单API和和查询订单API，我们在进行与微信平台进行支付的过程中，也就是通过这两个接口传给我们微信平台一些相关的参数，他们给了我们提供了对应的两个地址

我们在进行支付的时候因为我们采用的是一个统一下单的接口，我们也就无法根据商家下接口，这个时候我们也就使用了redis缓存所有商家的订单，我们也就可以根据商家订单查询付了多少钱，如果是单独商家进行的时候因为我们已经提前生成好了订单，所有微信支付系统也就生成了一个预支付交易传给我们的的商户后台，这个预支付界面也就是为了让我们的用户确认自己的收货地址，优惠券，联系方式什么的，

首先我们后端生成二维码，这个二维码创建过程分三步我们提供给微信平台需要的参数，我们当时把参数放到一个map集合当中，我们在进行向微信支付平台发送给这些信息的时候 采用了HttpClient这个工具类，

我们为什么不用原生的请求方式，因为原生的请求方式请求http还好，如果请求的是https会很麻烦，并且我们引入了一个HttpClient的工具类，这个工具类支持丰富的HTTP协议的客户端编程工具包，这个HttpClient就是模拟了浏览器的行为，如果我们需要向某一地址提交数据获取结果就可以直接使用HttpClient

然后根据SDK提供给我们的generateSignedXml将这些参数转换为xml格式，然后查看它是否符合HTTPS协议，设置为true,用post方式进行提交，提交之后微信会返回给我们一个链接，我们就可以根据这个链接生成二维码，并通过可以提取出我们需要的一些参数，但我们需要将他转换为map方式，因为微信平台传给我们的是xml格式，所以我们可以通过sdk提供给我们的方法转换为map，用quires前端生成二维码，我们的用户就可以进行支付了

//1.创建参数

Map<String,String> param=**new** HashMap();//创建参数

param.put("appid", appid);//公众号

param.put("mch\_id", partner);//商户号

param.put("nonce\_str", WXPayUtil.*generateNonceStr*());//随机字符串

param.put("body", "品优购");//商品描述

param.put("out\_trade\_no", out\_trade\_no);//商户订单号

param.put("total\_fee",total\_fee);//总金额（分）

param.put("spbill\_create\_ip", "127.0.0.1");//IP

param.put("notify\_url", "http://test.itcast.cn");//回调地址(随便写)

param.put("trade\_type", "NATIVE");//交易类型

**try** {

//2.生成要发送的xml

String xmlParam = WXPayUtil.*generateSignedXml*(param, partnerkey);

System.***out***.println(xmlParam);

HttpClient client=**new** HttpClient("https://api.mch.weixin.qq.com/pay/unifiedorder");

client.setHttps(**true**);

client.setXmlParam(xmlParam);

client.post();

//3.获得结果

String result = client.getContent();

System.***out***.println(result);

Map<String, String> resultMap = WXPayUtil.*xmlToMap*(result);

Map<String, String> map=**new** HashMap<>();

map.put("code\_url", resultMap.get("code\_url"));//支付地址

map.put("total\_fee", total\_fee);//总金额

map.put("out\_trade\_no",out\_trade\_no);//订单号

**return** map;

} **catch** (Exception e) {

e.printStackTrace();

**return** **new** HashMap<>();

主要进行了如下操作，

HttpClient client=**new** HttpClient(请求的url地址);

client.setHttps(**true**);//是否是https协议

client.setXmlParam(xmlParam);//发送的xml数据

client.post();//执行post请求

String result = client.getContent(); //获取结果

我们在做订单查询状态的时候也就用到了查询订单API,也是一样的方式，将我们的数据转成xml格式发送给微信平台，我们将这些数据放到整个Map集合当中，我们在做订单查询状态的时候，通过一个订单的ID进行查询，订单状态，如果查询过程中，我们始终没有获取数据，我们就一直查，这里和支付宝的支付方式不同，我们写了个死循环，直到它返还给我们状态为止我们就不再进行查询，我们既然查询到了订单状态，既然用户已经付完款，我们当然也要进行修改，这个时候我们是调用了UpdateByPrimaryKey这个方法，根据我们的订单Id进行修改状态并修改我们数据库的支付日志表

【保险吐槽、点赞、评论模块】

因为这个吐槽、点赞、评论操作，数据量较大，操作的特别频繁，价值很低，当时我们也考虑过用redis，可是我们都知道redis是把数据存储到内存当中的，其缓存的性质远远大于数据存储的性质，并且我们的这些用户的吐槽评论也许对我们来说不重要，但是对于用户来说是比较重要的， 所以当时我们选择mongodb来进行存储，原因：mongodb是一个跨平台的，面向文档的数据库，支持数据结构非常松散，是类似json的bosn格式，且支持复杂的数据类型，再操作增删改查可以添加很多条件，就像SQL数据库一样灵活

mongodb这个东西再2018-2019才开始火的 所以之前用的都是mongodb-driver的一个驱动包，之后就出现了springdatamongodb这个东西

Java链接mongodb的时候分5步

1. 首先得获得mongodb的链接 使用mongodbClient来选择端口 默认端口是27017
2. 选择对应的数据库 通过mongodbClient调用getdatabase这个方法选择自己的数据库名称
3. 根据数据库调用getcollection方法选择对应的集合
4. 如若是查询集合内的所有文档（一个文档代表一条数据）直接使用find方法查询后，for循环遍历输出便可查询全部
5. 打开mongodb这个链接后，我们也需要关闭这个链接mongodbClient调用close即可关闭

如果我们要进行多条件查询的时候，我们可以用 new BasicDBObject来构建查询条件，这里面存放的都是键值对，如果是多条件的话，就可以再new BasicDBObject中再嵌套一个实现多条件查询

添加操作直接使用insertOne即可添加，由于我们这个mongodb里面存放的都是bosn格式，所以再添加的时候，直接new hashmap，在map里通过put方法添加好数据后，用document对象可在调用insertone方法进行添加

吐槽模块采用springdatamongodb，也是SpringData之一，用来操作mongoDB的持久层框架，他的底层就帮助我们封装好了mongodb-driver，我们在具体操作的时候，只需要引入springdatamongodb这个依赖包就好了，具体的分页操作，因为是微服务项目，只要哦我们依赖common这个工程，就可以帮我们直接调用里面的分页等功能，因为我们做的是spring cloud微服务，所以要涉及到数据之间的传递，所以要实现一个serializeable这个接口，继承mongodbrepostry实现类便可进行基本操作

我们在根据上级进行吐槽查询的时候，需要根据父id查询并进行分页，这里分页用的是pageable,这个pageable是个接口，它提供了很多分页的参数，我们也只是需要通过pageRequest.of传页和条数就好了

在进行添加吐槽的时候 我们根据idworker进行自增一个主键值，然后根据这个主键值进行添加

我们在进行吐槽点赞的时候，首先我们应该判断用户是否点过赞，我们每回用户点赞过后，都会根据这名用户存储个记录，然后我们在进行判断用户是否重复点赞的时候，使用redistemplate调用get方法获取记录，然后进行判断记录是否为空，如果为空的话，说明用户并没有点过赞，我们就使用service调用点赞的方法进行根据用户id进行点赞，并且将记录通过redistemplate调用set方法存储该名用户的点赞记录放到redis当中，如果不为空则返还给我们的前台告知不能重复点赞

你的工程是如何操作MongoDB的？

在之前项目中都是引入mongodb-driver

这个驱动包的，后期用的是springdatamongodb

在项目的哪些场景下使用MongoDB ?

吐槽 、文章评论

为什么在吐槽和文章评论中使用Mongodb而不使 用mysql?

吐槽和评论都是数据量较大且价值较低的数据，为了减轻mysql的压力，我们使用 mongodb

工作时间2015年8月到2019年5月 21岁~25岁

出生日期1994年

大学：2011年9月~~2015年7月 17岁~21岁

高中：2008年9月~~2011年7月 14岁~17岁

初中：2005年9月~~2008年7月 11岁~16岁

小学：2000年-9月~~2005年-7月 6岁~11岁

毕业于北京东方学院 本科 专业：计算机科学与技术

北京银保联科技有限公司 外包公司 2016年8月到现在

地址：北京市海淀区北四环西路52号15层1501室

人员配备

3个后台，2个前端，2个测试，1个boss