# B简历：

# -------------------------------------------

### 基本信息

|  |  |  |  |
| --- | --- | --- | --- |
| 基本情况 | | | |
| **姓名** | 刘学 | 性别 | 男 |
| **出生日期** | 1996-05-25 | 属相 | 鼠 |
| **现住地** | 海淀区 [永泰园(北区)](https://beijing.anjuke.com/community/view/130545" \t "https://bj.zu.anjuke.com/fangyuan/_blank) |  |  |
| 大学情况 | | | |
| **学校** | 汉口学院 | 专业 | 计算机技术与科学(软件开发) |
| **入学时间** | 2014-9 | 毕业时间 | 2018-7 |
| **专业课** | **JAVA语言、操作系统实训、Java程序设计、数据库技术、计算机网络技术，网页设计** | | |
| **学历** | **本科** | 学位 | **学士学位** |
| **学校地址** | [湖北省武汉市江夏区文化大道299号](https://api.map.baidu.com/marker?output=html&location=30.422245,114.323313&title=%E6%B1%89%E5%8F%A3%E5%AD%A6%E9%99%A2&content=%E6%B9%96%E5%8C%97%E7%9C%81%E6%AD%A6%E6%B1%89%E5%B8%82%E6%B1%9F%E5%A4%8F%E5%8C%BA%E6%96%87%E5%8C%96%E5%A4%A7%E9%81%93299%E5%8F%B7&src=vmpMiddle" \t "https://www.baidu.com/_blank) | 校长 | [罗爱平](http://www.baidu.com/link?url=QnCZh4IVZTH_mJCy_q7C9GEmtaAdgcUr57WyAvVIG-A7b0MxM5r0kiPxZU8o9PLrK7dXIkTg1U2llX2Tr0TH-qI8pvKApp_sY9aDp8EnFTu" \t "https://www.baidu.com/_blank) |
| **去学校的方式：** |  | | |
| **上小学日期：** | **2002 年9月 6岁六年** | | |
| **上初中日期：** | **2008年9月 12岁三年** | | |
| **上高中日期：** | **2011年9月 15岁三年** | | |
| **上大学日期：** | **2014年9月 18岁四年** | | |
| **毕业时间：** | **2018年 7月 22岁** | | |
| **在校学生：** | **1000左右** | | |
| **有教师：** | **不太清楚** | | |

**核心概念：**

|  |  |
| --- | --- |
| 第一家公司 ： | **河南壹测吧网络技术有限公司** |
| **地址：** | **新乡市 红旗区 电子信息科技园 4楼** |
| **入职时间：** | **2017年12月初** |
| **离职时间：** | **2019年9月** |
| **公司性质：** | **民营** |
| **职位：** | **Java工程师** |
| **项目经理名字：** | **夏炎** |
| **薪资方面：** | **刚刚毕业转正以后工资是5K左右，后来涨到7K左右** |
| **路线：** | 66路公交车牧野大道下车 |

|  |  |
| --- | --- |
| **第二家公司情况：** | **峥嵘时代科技(北京)有限公司** |
| **公司地址：** | **北京市 朝阳区 时间国际北区 506** |
| **公司规模：** | **50人左右，我们项目组7人，一个项目经理，3名开发人员，1名前端，1个测试，1个运维** |
| **公司性质：** | **民营** |
| **公司行业：** | **互联网和虚拟现实的软件产品的开发。** |
| **去公司路线：** | **地铁10号线三元桥站500m** |
| **入职时间：** | **2019年11月** |
| **离职时间：** | **2021年4月初** |
| **职位：** | **Java工程师** |
| **项目经理名字：** | **焦晨辉** |
| **税前：** | **14K** |
| **税后：** | **13K** |
| **缴税：** | **1K左右(具体原因不清楚当时公司就是这样扣的税)** |
| **期望薪资：** | **16K** |
| **薪资的构成:** | **岗位工资(5K)+绩效工资(10K)** |

|  |
| --- |
| 四年涨薪： **2017年工作的时候是5K**  **转正6K左右**  **2016上半年涨了1K左右 ，总7K左右。**  **2017年2月跳槽涨了2K左右 总9K**  **2017年12月涨1K总 10K左右**  **2018年9月工作一年涨了2K左右**  **2019年7月工作一年涨了2K左右**  **2019年11月初离职时工资为14K** |

**面试常见问题：**

|  |
| --- |
| **你为什么从上家公司离职？** |
| **想换个工作环境，不断的提高充实自己** |
| **你交社保了吗？为啥没交？** |
| **没交**  **在上家公司的时候，人事说咱们公司的五险一金如果要上的话都是从自己工资里面扣的，当时感觉没啥必要也就没上。** |
| |  | | --- | | **三个词形容自己**： 交流表达能力强，团队协作能力强，抗压自学能力强 (**可以综合自己自学的知识表达**) | | **你的优缺点是啥？** |   **优点:**  **抗压能力强（有一次项目就要上线了，连着加班了一周，每天都到夜里很晚，但我最后也是成功完成任务。），**  **乐于分享，**  **团队协作能力强**  **缺点: 我这个人说话比较直，你像在和团队成员探讨问题的时候，容易得罪人，**  **不过现在已经改进的差不多了** |
| **你知道五险一金都是啥吗？** |
| **养老保险、医疗保险、失业保险、工伤保险、生育保险，和住房公积金。** |
| **你能接受加班吗** |
| **能** |
| **你能接受出差吗** |
| **能** |
| 你期望薪资多少？你上家工资多少？税后拿到手的有多少？扣了多少钱的税？ |
| 期望薪资 16K 上家工资15K 税后14K左右 扣了1K左右的税 |
| 你的五年规划是啥？ |
| **前2,3年 继续加强自己的技术功底，**  **然后朝着项目经理(技术经理,产品经理)方面发展** |
| 是统招吗 是是本科吗 是能在学信网查到吗 我当时上的是民办大学，能在民教网查到！ |
| 你们公司还有什么福利？多少薪： 13薪  **车补**  220左右  **项目奖金**  8k-10k，做完项目，收完尾款的时候发。  **年终奖**  就是1个月的工资，  最后一个月和工资一块儿发/年后发  **电话补助**  260  **饭/餐补**  **15、**30/天  **电脑补**  310 |

**咱们也知道这几年这个微服务springboot和springcloud用的非常多，而我在上一个项目中使用的也是这个springboot+springcloud进行的开发，那我就先说一下这个我对这个微服务框架的理解吧：**

我觉着SpringCloud就是一个分布式微服务架构的一站式解决方案，它提供了很多组件用来解决了分布式架构所带来的一些问题。我在之前工作里用过Eureka、Ribbon、Feign、Hystrix，Zuul这么几个组件。其中Eureka在整个微服务架构中充当注册中心的角色，服务提供者将自身信息注册到Eureka Server中，然后服务消费者就可以从Eureka Server中获取注册的服务提供者的信息，然后就可以向服务提供者发起调用了。Ribbon实现了客户端的负载均衡，它提供了轮询、轮询权重、随机等一些常用的负载均衡策略。Feign我理解的就是简化服务之间的调用，让我们调用远程接口就像在调用本地方法一样。Hystrix的主要功能就是服务熔断、降级和资源隔离，用来保护我们的调用链路，避免发生服务雪崩问题。Zuul在整个微服务架构中充当服务网关的角色，提供请求转发和过滤的功能，可以在服务网关中实现统一身份验证、统一跨域请求处理等功能。以上就是我对SpringCloud的一些简单理解。

当时在做的时候用的是

Eureka建立注册中心项目，在pom文件中引入Eureka的依赖，之后在启动类中加入 @EnableEurekaServer的注解，证明该项目的作用是作为注册中心使用，并且在对应的application.properties中指明Eureka注册中心服务器的地址，方便提供者和消费者对其进行访问，为了增强注册中心的高可用性，我们做了Eureka集群，防止因为单台Eureka导致的单点故障问题。通过注册中心可以让服务端注册到注册中心上去，并且可以进行负载均衡；这样就可以达到高可用以及提高并发的目的。Eureka Client每隔30秒会向Eureka Server发送一次心跳，就如同人还活着存在的信号一样，如果Eureka Server在90秒后还未收到发来的心跳时，那么它就会认定该服务已经死亡就会注销这个服务。但是这里注销并不是立即注销，而是会在60秒以后对在这个时间段内“死亡”的服务集中注销，如果立即注销，肯定会对Eureka造成极大的负担。客户端也不需要直接调用服务端，而是通过注册中心获取地址列表，从地址列表中选一个服务端接口进行调用，后续的维护各方面也会比较方便。**Eureka自我保护机制？**

Eureka还有自我保护机制，如果在15分钟内一直都是超过85%的节点都没有正常的心跳，那么Eureka Server就认为客户端与注册中心出现了网络故障，就会进入自我保护模式。进入自我保护模式之后Eureka Server就不会再接收心跳，也不会删除任何服务。当网络故障恢复后，Eureka Server会自动退出自我保护模式。这就是Eureka的自我保护机制

**在这个微服务的搭建过程中也有可能发生微服务雪崩的问题**

，说白了就是A->B->C,如果C这个微服务出现问题，就会影响到B,同样B出现问题，就会导致A也出问题，最后导致所有的微服务都不可用。就像咱们现实生活中，一个路口堵车，就会导致整条街慢慢拥堵，再后来影响的范围就会越来越大。为了解决这个问题同时也是为了保证我们微服务的高可用性我们当时使用到了Hystrix。

咱们都知道Hystrix可以进行熔断，降级和资源隔离。熔断其实就可以理解成咱们家里面的保险丝。保险丝是为了保证在高负载情况下电路的安全，如果某个电路出现了负载过大情况，保险丝就会熔断，从而保证其他电路不受影响。Hystrix的熔断功能比较强大，它还可以进行自动检测并修复。

具体的原理是这样的，默认情况下在10s内，每当20个请求中，有50%失败时，就会触发熔断，导致hystrix从闭合状态切换到打开状态，这样就不需要每次都去调用远程微服务而是直接走调用者本地的降级方法。5秒后hystrix会进入到半打开状态，这时如果有新的请求再过来hystrix就会尝试再次进行远程微服务调用，如果调用成功就会进入到闭合状态，如果不成功就会再次进入到打开状态，继续进行熔断。这个就是hystrix的自动检测并修复的能力。

降级说白了就是进行熔断后需要执行本地的方法。也是为了保证微服务的高可用。

[说白了就像咱们看的打仗时候的急行军，为了保证按时到达作战地点，我就把身上其他的干粮，背包全部丢掉，只带上最基本作战的武器，快速前进保证最终的结果。**可以省略**]

在Hystrix中, 有两种资源隔离方式，信号量隔离和线程池隔离，我们主要用的是线程池隔离. 通常在使用的时候我们会根据调用的远程服务划分出多个线程池. 比如调用产品服务的Command放入A线程池, 调用会员服务的Command放入B线程池. 这样就可以将运行环境隔离开。就算调用服务的代码存在bug或者由于其他原因导致自己所在线程池被耗尽, 也不会对系统的其他服务造成影响。

**接下来我来说下这个Feign的使用吧**

我们当时在项目中有很多的模块需要互相调用，比如说这个购物车需要调用商品进行查询，我们当时为了解决这个问题使用的是Feign来进行微服务之间的调用

又因为Feign集成了Hystrix,所以我们当时通过在配置文件中开启hystrix,并且在具体使用Feign的接口上通过 @FeignClient 中的 Fallback指定降级时候需要执行的类即可。这样在熔断后就会自动调用Fallback类中对应的降级方法，降级方法中，可以记录下日志并且返回null。这样使用Feign的客户端在调用 远程微服务 的方法后，判断返回值是否为null,如果是null则证明走了降级方法，进行特殊处理就行。

Feign还拥有负载均衡的特性，说白了它是靠Ribbon进行的负载均衡。在 @FeignClient的注解上指明要调用的微服务的名字，这样就可以通过该服务名从注册中心Eureka中获取对应的地址列表，方便进行负载均衡的调用。

其次我们会创建对应的服务端，服务端项目中要保证 请求的方式，请求的地址以及方法名，返回值，参数都要和feign中定义的保持一致。

**Zuul**

接口安全这方面的话我们是用zuul来里面的过滤器进行处理的

因为这个外部客户端的请求都是通过Zuul网关路由到具体的微服务，所以为了保证微服务的安全，我们就在Zuul中自定义了过滤器，对所有微服务的安全进行统一的处理，还有，因为涉及到前后端分离，前端项目访问后端微服务涉及到跨域问题，所以我们在Zuul中 也自定义了关于跨域的过滤器，进行统一处理。

具体是这么做的，首先你得先定义一个继承于ZuulFilter的类，重写里面的filterType()方法filterOrder()方法，shouldFilter()方法，还有一个最核心的用来写具体业务逻辑的run()方法。其中filterType方法的作用是用来返回一个字符串，指明该过滤器的类型，经常用到的有pre类型，说白了就是在请求被发送到微服务之前调用；我们的微服务安全认证以及跨域这块都是用的pre类型的过滤器，这样对非法请求，就可以在发送到具体的微服务之前拒绝它；

还有post类型，说白了就是微服务执行完后再执行该过滤器，filterOrder()方法返回一个int类型的值，用来指明该过滤器的执行顺序，数字越小表示优先级越高，就越先执行，shouldFilter()方法返回一个boolean值，用来指明该过滤器是否执行，true表示执行,false表示不执行。

run方法中就是之前说的，用来放具体的处理逻辑。

在run方法中，首先获取一个RequestContext对象，之后就可以通过调用它的getRequest()方法来获取request对象；这样就可以获取头信息，按照之前基于token的方式进行接口的安全验证，至于这个基于token的登录说白了就是自己写的一个类似于jwt的登录，这个我在后面在做介绍。

过程中特别需要注意的就是当验证不通过的时候，需要通过fastjson将要响应的数据转换为json格式的字符串，之后设置响应的内容类型为application/json并且指定utf-8的编码方式，用来处理中文乱码问题；通过setResponseBody将Json格式的字符串设置为响应的内容，最后通过setSendZuulResponse为false，禁止路由转发。如果想要将zuul过滤器中的数据传递给后端微服务中使用，则需要通过addZuulRequestHeader方法来进行，而后端微服务中就可以通过request.getHeader来获取值。这里面需要特别注意的就是，如果传递的数据中含有中文则需要通过URLEncoder进行utf-8的编码，同样在获取数据后也需要通过URLDecoder进行解码。

最后想要使自定义的过滤器生效，得进行相关的配置。我通过创建一个配置类，

并且在类上通过 @Configuration 和 方法上的 @Bean 结合起来，完成自定义filter的配置。或者直接在启动类加@Bean注解

这就是我们当时搭建这个微服务框架的一个过程

spring cloud又是基于spring boot的 我理解的SpringBoot优点呢，有三个，第一个就是使用SpringBoot可以快速的构建出来一个可以独立运行Spring项目第二个就是开箱即用 SpringBoot为我们做了大量自动配置，简化了我们的开发工作 第三个就是内置了启动器 无需依赖外部的Servlet容器就可以运行我们的项目

在 @SpringBootApplication 的内部包含了3个注解

@Configuration

@EnableAutoConfiguration

@ComponentScan

@Configuration是基于JavaConfig形式的Spring Ioc容器的配置类，可以把它看成xml配置文件中的beans标签。 @Configuration写到类上面，在类中的方法上如果写了 @Bean注解，那么它的返回值将作为一个bean注册到Spring的IoC容器，方法名默认作为bean的id。

@ComponentScan这个注解对应XML配置中的context:component-scan元素， 说白了它的作用就是自动扫描并加载符合条件的组件,比如 @Component和 @Service等或者bean定义，最终将这些bean定义加载到IoC容器中。

我们可以通过basePackages来指定 @ComponentScan 自动扫描的范围,如果不指定，则默认Spring框架实现会从声明 @ComponentScan所在类的package进行扫描。这也是SpringBoot的启动类最好是放在root package下的原因。

@EnableAutoConfiguration 这个注解是借助 @Import的帮助，将所有符合自动配置条件的bean定义加载到IoC容器中。

这个项目里我负责的是用户登录模块

用token是为了确认用户是否登录，没有登录的话就不能让他访问接口，防止有人恶意访问我们的接口。

刚开始我们的项目里用的session，结果发现session因为是作用在一个服务器的会话中，在分布式框架中不能获取到数据，然后就用了redis来解决这个问题，把数据存放到redis中，再从redis中取出数据，这样就解决了跨域问题，最后接触到了JWT技术，然后就把redis替换了。

用户在前端输入账号和密码，请求路径传到后端，会遇到一个跨域问题（解决这个跨域问题的方式有多种，可以通过jsonp,也可以通过spring给咱们提供的一个叫 @CrossOrigin的注解来实现，或者自己写一个CorsFilter来实现，但是我们当时的项目中所有的请求都会先经过zuul网关的过滤器进行安全验证，所以我就将跨域处理的相关代码写到了过滤器里面），在后端，我们配置了一个跨域拦截器，继承的ZuulFilter来解决跨域问题，然后我们是根据用户的账号查询数据库里是否有这个数据，没有的话说明他没有注册，然后重定向到注册页面让用户注册，反之，将传过来的密码进行MD5加密，用MD5加密密码是为了防止数据库密码泄露，然后判断密码是否正确，如果不正确返回一个状态码给前端，反之，将账号和密码放到map中，用JWT将map生成token令牌，再设置一个失效时间，然后将生成的token返回到前端。

前端我们用的是vue，在登录成功之后，我们把后端传过来的token存到sessionStorage中，然后我们在vue的main.js中设置了一个axios的全局设置，为了让每次的请求都添加上一个token参数，在axios的全局设置中判断本地的sessionStorage中是否有token，有的话就在请求上添加一个token参数，反之发请求的话会报没有token的错误。然后再配置一个axios全局设置的请求响应，判断返回回来的状态码，如果相同的话就返回到登录页面，让他登录。

session和token的区别就是session是存在一个服务器会话中，不能用在分布式框架中，相反token可以。

**接下来我给您介绍下我做的这个购物车模块吧**

考虑到用户操作购物车的性能问题，我当时做的是基于redis的购物车，将购物车的所有数据都存储到redis中。

当时用了redis中的hash结构来存储所有用户购物车的数据，咱们都知道在使用hash的时候，涉及到key,field,value这三个方面的参数信息，key这块我用了用户名+CartMap这个静态的常量来表示，在field这块因为每个用户都有一个购物车，为了保证唯一性，我用了商品id和商品的SKU ID表示，在value这块我是把会员对应的整个购物车的信息都转换为Json格式进行了存储。整个购物车对应的数据结构是这个样子的。