# 云计算 1607C 徐强

# JAVA工程师

1. 项目介绍(按照实训二做的项目写)

|  |
| --- |
| 这个项目是关于P2P个人借贷的金融项目,采用的框架是SSM框架,前后端分离,前端主要用户使用主要账户信息管理，借贷管理，资产管理等，后端由平台管理人员使用，主要有用户管理，审核管理，平台管理，安全管理等。  主要模块包括借款和投资模块，信息认证及审核模块。用户在平台中可以投资或者借款，借款有三种类型，信用贷，车贷和房贷，首先对于信用贷，是通过认证风控资料之后，后台累计一个积分数值，达到一定分值之后，并且认证了实名，基本信息以及视频认证之后，才能发起借贷。  对于信用贷来说，风控资料是最核心部分，这是证明一个人是否有偿还能力的根本。其次如果达到了借贷条件，可以申请借款，借款人发起招标，数据库生成当前申请标的信息，标的信息包括总借款金额，偿还利息，最低投标金额，标的描述信息等，发起申请之后不是立即发布到平台上，后台审核通过后，才能被所有前台用户可见可投。  对于未登陆用户，只能看到此标的的介绍信息，更详细的内容需要登录之后才能查看，已登录用户可以在标的界面查看关于招标人基本信息，认证信息，风控信息，以及当前投标人的信息等，投标人投标后账户冻结投标金额，满标后投标人和招标人也不会立即形成借贷关系，后台需要经过两次审核，如果审核不通过，投标人所投金额会解冻，招标人则可以发起下一次投标，如果两次审核都通过，则扣投标人冻结金额，增加招标人账户资金，平台从中收取手续费，并为招标人生成分期还款的信息，投标人可查看自己受益，招标人可查看自己当前账户分期偿还信息，直至还清借款后才能发起下一次借款。 |

1. 功能模块介绍(按照自己得实际项目写) 功能,技术点,流程

|  |
| --- |
| 前端:   1. 发标前审核:   后台获取未审核的标的记录，查看申请人的相关信息和资格，包括实名认证信息，基本信息，风控信息，以及此前借贷偿还信息，然后提交审核结果，数据提交后台，在标的审核记录表生成新的审核记录，因为标的存在待审状态，招标状态，满标和流标等多种状态，所以不再原记录添加审核人审核时间等参数。如果审核通过，此标进入招标状态，将被所有人在前台可见，如果审核失败，则标的进入审核失败状态，用户恢复借贷资格。  (2)满标一审及二审:  满标之后需要两次审核才能形成实际借贷关系，一审和二审业务逻辑大部分相同，都是前台传递审核结果，标的的id以及审核备注信息，后台判断此标信息是否有效，以及借款人的账户资金情况，在审核结果记录表中添加审核记录，包括标的信息，审核人信息等，在根据审核结果是否通过再执行相关业务逻辑。审核不通过，将投标用户的账户资金解冻，增加可用金额，账户流水记录添加记录，满标一审二审的逻辑都一样，不一样的的是一审通过后，标的信息变为等待二审，而二审通过后，则扣除投标人账户资金，即扣除冻结金额，然后平台从借款金额中收取手续费，为借款人生成还款对象以及  (3)借款审核:  在页面填写相关借款信息，包括借款金额（不超过信用额度），利息，还款期限，最小投标数，借款描述信息等，其中利息和最小最大、借贷金额需要在跳转页面前从后台获取平台默认的最小最大利息值，页面填写好相关信息之后，参数通过表单传到后台，首先要去判断当前用户id是否是登录用户，资金是否正常，然后借款利息，金额，期限是否都是有效的参数，没有问题的话就在数据库生成标的记录，并将用户状态改为有一个借贷流程进行的状态，等待后台审核。 |

1. 开发中遇到的技术难点以及解决办法

|  |
| --- |
| 难点1：当前项目是分布式项目，如何使用事物  TCC 方案  TCC 的全称是：Try、Confirm、Cancel。  Try 阶段：这个阶段说的是对各个服务的资源做检测以及对资源进行锁定或者预留。  Confirm 阶段：这个阶段说的是在各个服务中执行实际的操作。  Cancel 阶段：如果任何一个服务的业务方法执行出错，那么这里就需要进行补偿，就是执行已经执行成功的业务逻辑的回滚操作。（把那些执行成功的回滚）  难点2：问题：session共享问题  解决1： tomcat + redis  这个其实还挺方便的，就是使用 session 的代码，跟以前一样，还是基于 tomcat 原生的 session 支持即可，然后就是用一个叫做 Tomcat RedisSessionManager 的东西，让所有我们部署的 tomcat 都将 session 数据存储到 redis 即可。  解决2：给 sping session 配置基于 redis 来存储 session 数据，然后配置了一个 spring session 的过滤器，这样的话，session 相关操作都会交给 spring session 来管了。接着在代码中，就用原生的 session 操作，就是直接基于 spring sesion 从 redis 中获取数据了。  难点:3：跨域   1. 解决：跨域解决方案 2. 通过jsonp跨域 3. 跨域资源共享（CORS） 4. nginx代理跨域 5. nodejs中间件代理跨域 6. WebSocket协议跨域 |
|  |

1. 项目技术亮点分析 字体加粗

|  |
| --- |
| 的 亮点  1.分布式架构CAS单点登录  访问服务： SSO 客户端发送请求访问应用系统提供的服务资源。  定向认证： SSO 客户端会重定向用户请求到 SSO 服务器。  用户认证：用户身份认证。  发放票据： SSO 服务器会产生一个随机的 Service Ticket 。  验证票据： SSO 服务器验证票据 Service Ticket 的合法性，验证通过后，允许客户端访问服务。  传输用户信息： SSO 服务器验证票据通过后，传输用户认证结果信息给客户端。    2.redis  在项目中，我们使用了redis缓存热点数据，将页面展示的  借贷信息进行缓存，因为这部分数据在每个访问网站的人都可以看到，  所以缓存在redis中，采用hash的数据结构，同时，如果后台审核  通过了新的借款流程，redis也会追加最新数据，同时，我们也在  Redis缓存短信验证码，在用户改密码或者注册，短信的验证码是存  在Redis中。    3.)JWT技术:因为我们的端口是暴露在公网中，会出现一些安全问题，  所以在当时我们项目中用Jwt实现Tonken验证机制的,jwt包括头部、  载荷与签名,头部描述的是关于该JWT的最基本的信息，类似于一个  json对象,在头部指明了签名算法是HS256算法,我们进行BASE64编  码得到jwt的第一部分,载荷就是存放有效信息的地方,利用自己的一  些规则进行第二部分的加密(过期时间必须要大于签发时间),第三步  是通过前两部分进行加盐加密实现的中,将这三部分组成则就生成  一个大的字符串这就是我们的Token.在进行测试时我们利用Postman  工具进行测试,在测试的时候我们按照之前前后端的约定有一个头信  息Authorization然后在加上自己设定的简单密钥进行测试,这样  就保证我们后台的安全了。  4.dubbo六种容错策略  1．Failover Cluster 模式  失败自动切换，当出现失败，重试其它服务器。(缺省) 通常用于读操作，但重试会带来更长延迟。 可通过retries=”2”来设置重试次数(不含第一次)。  2．Failfast Cluster  快速失败，只发起一次调用，失败立即报错。通常用于非幂等性的写操作，比如新增记录。  3．Failsafe Cluster  失败安全，出现异常时，直接忽略。通常用于写入审计日志等操作。  4．Failback Cluster  失败自动恢复，后台记录失败请求，定时重发。通常用于消息通知操作。  5．Forking(fo gen) Cluster  1.并行调用多个服务器，只要一个成功即返回。通常用于实时性要求较高的读操作，但需要浪费更多服务资源。 可通过forks=”2”来设置最大并行数。  6．Broadcast(bu ruo ka si te) Cluster  广播调用所有提供者，逐个调用，任意一台报错则报错。(2.1.0开始支持)通常用于通知所有提供者更新缓存或日志等本地资源信息 |

1. 项目答辩中亮点简历审核评定标准

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **考核内容** | **简历样式是否规范** | **项目内容是否完整** | **项目中是否有亮点** | **技术难点解决办法是否高效** | **存在问题** |
| **项目经理 （评定）** |  |  |  |  |  |
| **就业处干事（评定）** |  |  |  |  |  |
| 简历是否合格 | | | | 合格 | 不合格 |

1. 项目答辩成绩评定标准

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **考核内容** | **亮点简历与项目描述是否匹配** | **项目流程是否清晰** | **学生讲解是否流畅** | **项目介绍是否到位** | **技术亮点考核是否合格** | **存在问题** |
| **项目经理 （评定）** |  |  |  |  |  |  |
| **就业处干事（评定）** |  |  |  |  |  |  |
| 项目答辩是否合格 | | | | 合格 | 不合格 |  |