

**团队项目实训报告**

|  |  |
| --- | --- |
| **实训名称：** | **微众银行区块链项目实训** |
| **组号：** | **第三组** |
| **项目名：** | **基于BCOS的数据共享平台** |
| **组长姓名：** | **刘鼎** |
| **学生专业：** | **软件工程** |
| **开课学期：** | **2020-2021-02** |

**华南理工大学软件学院**

**2021年07月**

团队项目实训报告目录

1、小组基本情况介绍

2、数据共享平台系统项目规格说明书

3、数据共享平台系统概要说明书

4、数据共享平台系统详细说明书

5、数据共享平台系统测试报告

6、数据共享平台系统项目总结

# 小组基本情况介绍

|  |  |  |  |
| --- | --- | --- | --- |
| 组号： | **第三组** | | |
| 项目名： | **数据共享平台** | | |
| 项目地址： | **华南理工大学** | | |
| 组长  姓名： | 刘鼎 | 学号： | 201830252014 |
| 班级： | 卓越二班 | 专业： | 软件工程 |
| 主要职责（分小点写清楚项目中承担的角色和承担的主要工作任务）： | | | |
| 1、 BCOS区块链搭建  2、 智能合约编写  3、 页面设计  4、 数据大厅页面编码  5、 转移积分页面编码  6、 黑盒测试  7、 项目文档编写 | | | |
| 组员1  姓名： | 李浩然 | 学号： | 201836480251 |
| 班级： | 卓越二班 | 专业： | 软件工程 |
| 主要职责（分小点写清楚项目中承担的角色和承担的主要工作任务）： | | | |
| 1、BCOS区块链搭建  2、智能合约编写  3、页面设计与前端编码  4、黑盒测试  5、项目文档编写 | | | |
| 组员2  姓名： | 李健 | 学号： | 201830251239 |
| 班级： | 卓越二班 | 专业： | 软件工程 |
| 主要职责（分小点写清楚项目中承担的角色和承担的主要工作任务）： | | | |
| 1. BCOS区块链搭建  2. 智能合约编写  3. 后端框架搭建  4. 后端用户模块编码  5. 后端数据模块编码  6. 后端控制模块编码  7. 后端请求模块编码  8. 后端接口单元测试  9. 项目文档编写 | | | |
| 组员3  姓名： | 刘泽辉 | 学号： | 201830670368 |
| 班级： | 卓越二班 | 专业： | 软件工程 |
| 主要职责（分小点写清楚项目中承担的角色和承担的主要工作任务）： | | | |
| 1、BCOS区块链搭建  2、智能合约编写  3、后端POJO、Result的编码以及controller层的完善  4、黑盒测试  5、项目文档编写 | | | |

# 数据共享平台系统项目需求规格说明书

1. **概述**
   1. **编写目的**

本设计说明书简单阐明了数据共享平台系统的基本设计思想、基本功能、模块划分以及模块间接口，便于各模块开发人员能更好地了解该系统的基本情况及各模块详细功能。

* 1. **项目背景**

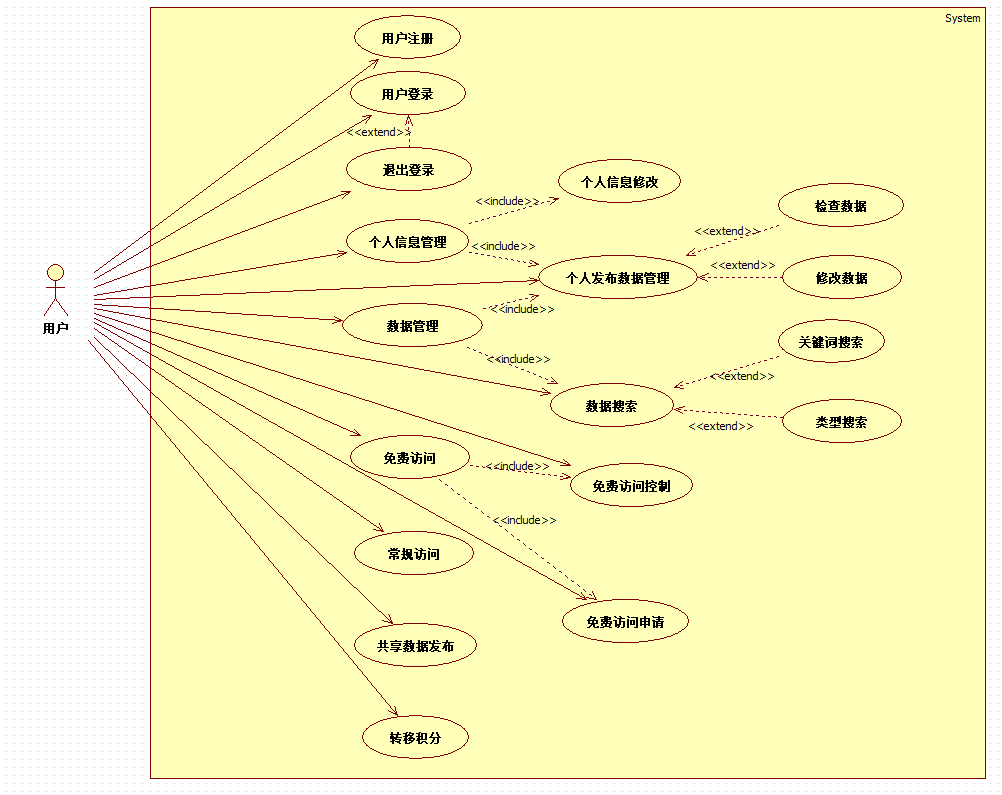
目前数据共享的应用需求不断扩大，由于主流的数据共享系统采用集中式的服务模式，在没有权威机构背书的情况下，容易导致共享数据被滥用、用户隐私遭泄露的问题，同时缺少对用户数据共享行为的有效引导。区块链技术去中心化、数据不可篡改、交易可追溯的技术特性，为实现去中心化的数据共享系统，保障共享数据安全，丰富访问控制机制带来了新的希望。

本文提出基于区块链的数据共享系统访问控制机制，实现了基于区块链的数据共享系统访问控制部分。系统基于spring boot架构，使用web vue前端技术实现表示层，通过智能合约完成对区块链上访问控制数据的操作，可向用户提供用户信息管理、共享数据管理、共享数据发布、共享数据访问控制等功能，实现去中心化的、安全高效的数据共享服务。

本文提出的访问控制机制利用区块链技术特性，让访问控制过程公开透明，保证共享数据和用户信息的安全性。同时本文提出共享积分作为访问控制机制的用户激励方式，通过共享积分引导共享数据的流通，提高用户参与数据共享的积极性。另外，本文使用智能合约执行访问控制机制，有效保障机制运行效率，避免人为因素对机制造成的干扰。

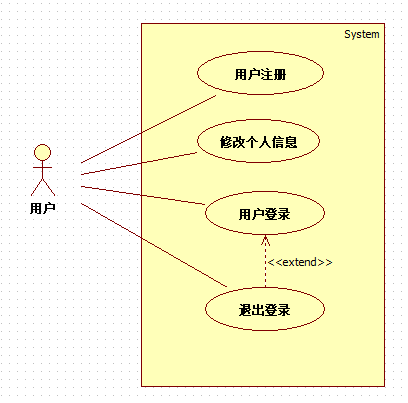
1. **系统功能需求**

**2.1. 系统功能架构**



**2.2.用户模块**

**2.2.1.用例图**



* + 1. **用例描述**

**用户登录用例**

|  |  |
| --- | --- |
| 用例简要描述 | 用户可以在登陆界面选择填写信息后登录系统，也可以注册一个新账号登陆系统。 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户打开登录界面并且尚未登录 |
| 用例后置条件 | 成功登录系统或退出系统 |
| 基本事件流 | 1. 用户打开系统  2. 系统显示欢迎界面，并要求输入账户名和密码  3. 用户输入账号和密码。  4. 用户点击 登录 按钮。  5. 用户成功登录 |
| 备选事件流1 | 1. 用户打开系统 2. 系统显示欢迎界面，并要求输入账户名和密码 3. 用户输入账户名和密码 4. 系统显示用户名错误或密码错误的提示信息 5. 用户可以重新输入登录信息，也可以选择结束该用例 |
| 备选事件流2 | 1. 基本事件流成功，进入首页面 2. 用户点击退出登录按钮，退出账户登录 3. 返回初始欢迎界面 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 用户名、密码 |
| 用例相关人（参与讨论的人员） | 刘泽辉、刘鼎、李健、李浩然 |

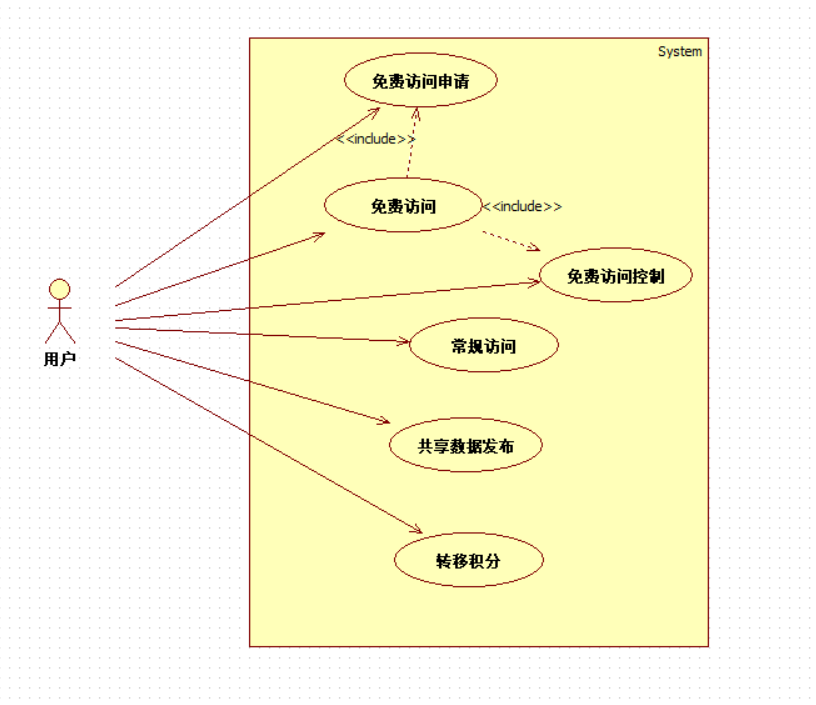
**修改个人信息用例**

|  |  |
| --- | --- |
| 用例简要描述 | 用户可以在登录成功后，进到个人中心对个人信息进行修改 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户登录成功并进入个人中心 |
| 用例后置条件 | 成功修改个人信息或进入其他页面 |
| 基本事件流 | 1. 用户登录系统，进入个人中心页面  2. 用户点击编辑信息按钮  3. 用户编辑需要修改的地方  4. 点击确认  5. 确认无误后，修改成功并刷新个人中心界面。 |
| 备选事件流1 | 1. 用户登录系统，进入个人中心页面  2. 用户点击编辑信息按钮  3. 用户编辑需要修改的地方  4. 点击确认  5. 信息违规，修改失败  6. 用户可重新键入信息修改也可退出编辑页面 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 用户名、密码 |
| 用例相关人（参与讨论的人员） | 刘泽辉、刘鼎、李健、李浩然 |

**用户注册用例**

|  |  |
| --- | --- |
| 用例简要描述 | 用户可以在登陆界面选择填注册一个新账号登陆系统。用户可以随时退出登录。 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户打开登录界面并且点击注册按钮 |
| 用例后置条件 | 成功注册账号或退出系统 |
| 基本事件流 | 1. 用户打开系统  2. 系统显示欢迎界面，并要求输入账户名和密码  3. 用户无账号，点击 注册 按钮。  4. 用户输入注册需要的信息，点击确认  5. 确认无误后，注册成功并跳转到登陆界面。 |
| 备选事件流1 | 1. 用户打开系统  2. 系统显示欢迎界面，并要求输入账户名和密码  3. 用户无账号，点击 注册 按钮。  4. 用户输入注册需要的信息，点击确认  5. 注册信息违规，注册失败  6. 用户可重新键入信息注册也可退出系统 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 用户名、密码 |
| 用例相关人（参与讨论的人员） | 刘泽辉、刘鼎、李健、李浩然 |

* 1. **上传下载模块**
     1. **用例图**



* + 1. **用例描述**

**转移积分**

|  |  |
| --- | --- |
| 用例简要描述 | 用户可以将积分转移给另一名指定的用户 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户已登录华工云盘 |
| 用例后置条件 | 积分转移成功 |
| 基本事件流 | 用户打开积分转移界面  用户输入要转移的积分  用户输入转移对象的用户名  点击确定转移键  系统显示积分转移成功 |
| 备选事件流1 | 用户打开积分转移界面  用户输入要转移的积分  用户输入转移对象的用户名  点击确定转移键  系统显示“积分余额不足”或“转移对象用户名不存在”或“不能将积分转移给自己”  用户重新输入要转移的积分或 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 用户名、用户积分 |
| 用例相关人（参与讨论的人员） | 刘鼎 |

常规访问

|  |  |
| --- | --- |
| 用例简要描述 | 用户通过常规访问获得共享数据 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户点击对应数据的“申请访问” |
| 用例后置条件 | 常规访问成功 |
| 基本事件流 | 用户点击“常规访问”  成功获得共享数据 |
| 备选事件流1 | 用户点击“常规访问”  访问失败，系统显示“用户无访问资格” |
| 备选事件流2 | 用户点击“常规访问”  访问失败，系统显示“用户共享积分不足” |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 用户名、用户积分、共享数据 |
| 用例相关人（参与讨论的人员） | 刘鼎 |

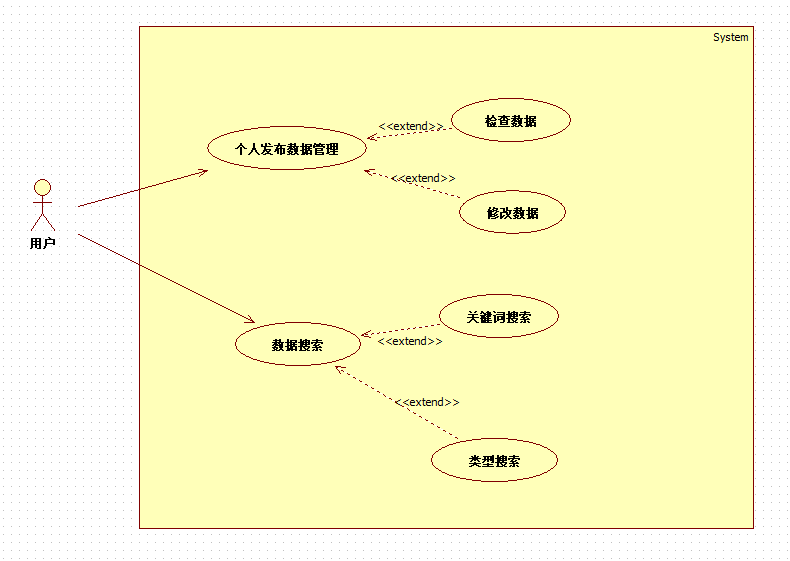
1. 免费访问申请
2. 用例描述

|  |  |
| --- | --- |
| 用例简要描述 | 用户通过免费访问获得共享数据 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户点击对应数据的“申请访问” |
| 用例后置条件 | 免费访问成功 |
| 基本事件流 | 用户点击“免费访问”  输入申请描述  （数据发布者批准）  成功获得共享数据 |
| 备选事件流1 | 用户点击“免费访问”  输入申请描述  （数据发布者拒绝）  获得共享数据失败 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 用户名、用户积分、共享数据 |
| 用例相关人（参与讨论的人员） | 刘鼎 |

1. 免费访问控制
2. 用例描述

|  |  |
| --- | --- |
| 用例简要描述 | 用户批准或者拒绝其他用户对于共享数据的免费访问 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户点开消息处理页面 |
| 用例后置条件 | 获取智能合约端处理结果 |
| 基本事件流 | 用户点击“批准”  操作成功，系统监听到免费访问批准事件 |
| 备选事件流1 | 用户点击“拒绝”  操作成功，系统监听到免费访问拒绝事件 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 用户名、用户积分、共享数据 |
| 用例相关人（参与讨论的人员） | 刘鼎 |

* 1. **数据大厅模块**
     1. **用例图**

****

* + 1. **用例描述**

**修改数据**

|  |  |
| --- | --- |
| 用例简要描述 | 用户对数据修改 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户登录系统 |
| 用例后置条件 | 数据修改成功 |
| 基本事件流 | 1. 用户进入数据修改页面 2. 输入框填写数据名、数据所需积分和数据描述等相关信息，上传共享数据文件 3. 点击确认   4、提示修改成功返回结果 |
| 备选事件流1 | 无 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 无 |
| 用例相关人（参与讨论的人员） | 李浩然 |
| 用例简要描述 | 用户根据关键词查询数据 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户登录系统 |
| 用例后置条件 | 数据修改成功 |
| 基本事件流 | 1. 用户输入数据名字 2. 点击搜索   3、返回此名字的数据 |
| 备选事件流1 | 无 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 无 |
| 用例相关人（参与讨论的人员） | 李浩然 |
| 用例简要描述 | 用户根据数据类型查询 |
| 用例角色 | 用户 |
| 用例前置条件 | 用户登录系统 |
| 用例后置条件 | 数据修改成功 |
| 基本事件流 | 1. 用户选择查询数据类型   2、返回此类型的所有数据 |
| 备选事件流1 | 1. 用户选择查询数据类型 2. 用户输入数据名字 3. 返回此名字的此类型的所有数据 |
| 用例非功能性需求 | 性能需求：  响应时间：小于0.1秒  可用性需求 ：  正常运行时间:99.9%。  并发需求：  能支持至少百个并发请求。  安全性需求： |
| 用例相关业务数据 | 无 |
| 用例相关人（参与讨论的人员） | 李浩然 |

1. **系统非功能需求**

|  |  |
| --- | --- |
| 性能需求 | 系统能够适应400个用户，平均每个会话持续8分钟；  生成的web页面，通过速率100KBps在10秒内可以全部下载 |
| 可用性需求 | 正常运行时间:99.99%。 |
| 并发需求 | 能支持至少百个并发请求。 |
| 安全性需求 | 所有涉及信息的操作必须登录后才能完成；  系统只允许用户查看自己的信息，而不能查看其他用户的资料。 |

1. **系统接口：外部系统接口**

|  |  |
| --- | --- |
| 用户界面 | 界面风格：本系统采用的是图形用户界面，界面简单且搭配合理。  界面布局：系统页面较为合理。  界面操作：页面上的按钮和文本框合理设计，让用户的使用更加方便快捷。  界面内容：界面设置导航，消息显示为弹出式。 |
| 硬件接口 | 软件之间交流的数据：用户账户上的信息，如已完成交易等。  通信协议：HTTP协议 |
| 软件接口 | 数据存储：采用FISCO BCOS 区块链  操作系统：开发系统为Ubuntu 20.02  工具：IDEA、VSCODE、Web Storm等  数据访问：Web3j SDK |
| 通信接口 | 通过浏览器访问 |
| 故障处理 | 登录失败或连接失败：尝试检查本地网络连接，并重启浏览器。 |

# 数据共享平台系统概要设计说明书

1. **概述**
   1. **系统简述**

目前数据共享的应用需求不断扩大，由于主流的数据共享系统采用集中式的服务模式，在没有权威机构背书的情况下，容易导致共享数据被滥用、用户隐私遭泄露的问题，同时缺少对用户数据共享行为的有效引导。区块链技术去中心化、数据不可篡改、交易可追溯的技术特性，为实现去中心化的数据共享系统，保障共享数据安全，丰富访问控制机制带来了新的希望。

本文提出基于区块链的数据共享系统访问控制机制，实现了基于区块链的数据共享系统访问控制部分。系统基于spring boot架构，使用web vue前端技术实现表示层，通过智能合约完成对区块链上访问控制数据的操作，可向用户提供用户信息管理、共享数据管理、共享数据发布、共享数据访问控制等功能，实现去中心化的、安全高效的数据共享服务。

本文提出的访问控制机制利用区块链技术特性，让访问控制过程公开透明，保证共享数据和用户信息的安全性。同时本文提出共享积分作为访问控制机制的用户激励方式，通过共享积分引导共享数据的流通，提高用户参与数据共享的积极性。另外，本文使用智能合约执行访问控制机制，有效保障机制运行效率，避免人为因素对机制造成的干扰。

* 1. **软件设计目标**

a) 节省开发费用和降低资源消耗。

b) 较高的可靠性。

c) 良好的可维护性。

* 1. **参考资料**

1. CSDN
2. 百度
   1. **修订版本记录**

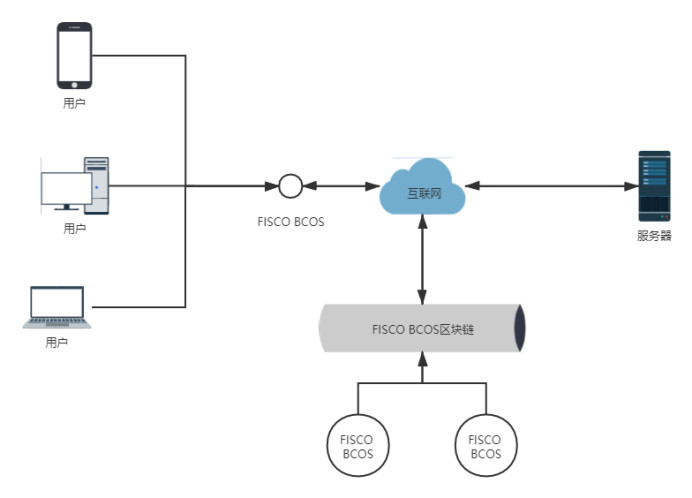
|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2021.6.18 | v1.0 | 完成概述部分及数据库表部分编写 | 刘鼎、李健、李浩然、刘泽辉 |
| 2021.7.3 | v2.0 | 添加区块链相关部分 |  |

1. **术语表**

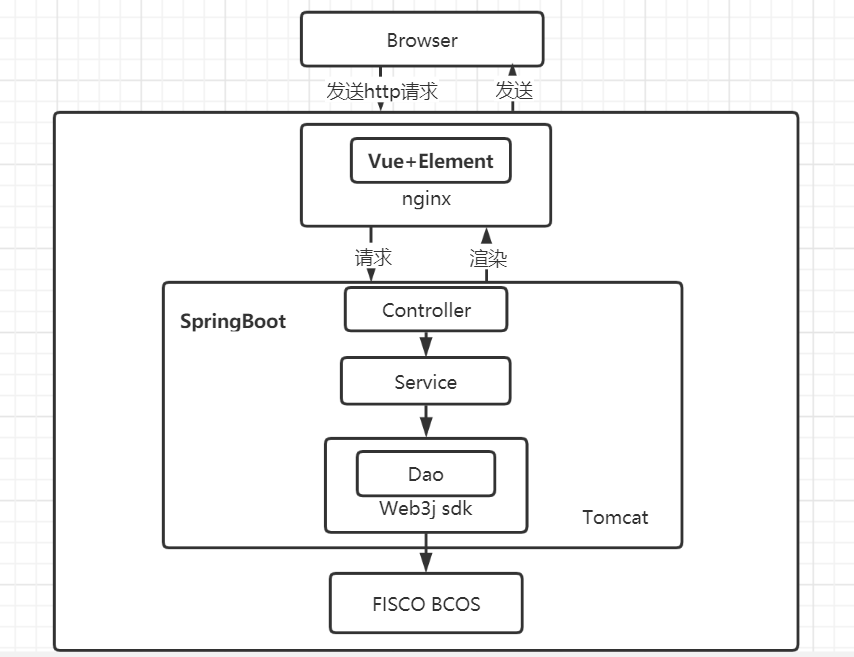
|  |  |  |
| --- | --- | --- |
| 术语 | 解释 |  |
| 区块链 | 区块链是一个信息技术领域的术语。从本质上讲，它是一个共享数据库，存储于其中的数据或信息，具有“不可伪造”“全程留痕”“可以追溯”“公开透明”“集体维护”等特征。基于这些特征，区块链技术奠定了坚实的“信任”基础，创造了可靠的“合作”机制，具有广阔的运用前景。 |  |
| 以太坊 | 以太坊（英文Ethereum）是一个开源的有智能合约功能的公共区块链平台，通过其专用加密货币以太币（Ether，简称“ETH”）提供去中心化的以太虚拟机（Ethereum Virtual Machine）来处理点对点合约。 |  |
| Solidity | Solidity 是 Ethereum 的一种契约型编程语言，其语法与 JavaScript 类似，并且旨在定位到以太坊虚拟机。  Solidity 是静态类型的，支持继承、库和用户自定义类型以及其他功能。可以用来创建投票、众筹、盲拍、多签名钱包等合同。 |  |
| spring boot | Spring Boot是由Pivotal团队提供的全新框架，其设计目的是用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。通过这种方式，Spring Boot致力于在蓬勃发展的快速应用开发领域(rapid application development)成为领导者。 |  |
| Vue | Vue是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与现代化的工具链以及各种支持类库结合使用时，Vue 也完全能够为复杂的单页应用提供驱动。 |  |
| Element | Element，一套为开发者、设计师和产品经理准备的基于 Vue 2.0 的组件库，提供了配套设计资源，帮助你的网站快速成型。由饿了么公司前端团队开源。 |  |

1. **设计概述**
   1. **系统结构设计**
      1. **系统物理架构**

用户可以使用电子设备作为以太坊节点，通过互联网向服务器获取系统的网页界面，通过网页与以太坊区块链间的交互完成数据共享操作。



* + 1. **系统逻辑架构**



* + 1. **系统数据模型**
       1. **数据字典**

|  |  |  |
| --- | --- | --- |
| 属性名 | 属性类型 | 属性含义 |
| userId | uint8 | 用户Id |
| credit | uint8 | 用户持有的共享积分 |
| userName | string | 用户名 |
| pwd | string | 用户密码 |
| userDescription | string | 用户描述 |
| userAddress | address | 用户的以太坊地址 |
| accessible | uint8[] | 可访问的共享数据列表 |
| msgList | uint8[] | 用户的消息列表 |

表1 用户表

|  |  |  |
| --- | --- | --- |
| 属性名 | 属性类型 | 属性含义 |
| dataId | uint8 | 共享数据Id |
| dataCredit | uint8 | 共享数据所需积分 |
| downloadTimes | uint8 | 共享数据被下载次数 |
| notAllowed | uint8[] | 不允许访问本数据的用户列表 |
| dataName | string | 共享数据名 |
| dataDescription | string | 共享数据描述 |
| dataAddress | string | 数据共享地址 |
| dataType | string | 共享数据类型 |
| userName | string | 数据拥有者名字 |

表2 共享数据表

|  |  |  |
| --- | --- | --- |
| 属性名 | 属性类型 | 属性含义 |
| reqId | uint8 | 消息发送方 |
| resId | uint8 | 消息接收方 |
| dataId | string | 涉及的共享数据Id |
| status | uint8 | 0为免费申请中，1为免费申请批准，2为免费申请拒绝 |
| text | string | 消息内容（申请原因） |

表3 消息表

* 1. **系统接口设计**

|  |  |  |
| --- | --- | --- |
| 所属 | 接口 | 功能 |
| 登录模块 | 登录 | 前端发送用户信息，后端执行登录 |
| 用户模块 | 注册 | 前端发送用户名、密码和以太坊地址，后端执行注册 |
| 修改个人信息 | 前端发送修改后的用户信息，后端执行修改 |
| 获取个人信息 | 前端发送请求，后端返回用户信息 |
| 查看用户数量 | 前端发送请求，后端返回注册用户数量 |
| 转移积分 | 前端发送转移积分数量与对象，后端执行积分转移 |
| 数据模块 | 发布数据 | 前端发送要发布的数据信息，后端执行数据发布 |
| 修改数据 | 前端发送修改后的数据信息，后端执行数据修改 |
| 获取所有数据 | 前端发送请求，后端返回所有数据信息 |
| 获取数据数量 | 前端发送请求，后端返回已发布数据数量 |
| 精确查询数据 | 前端发送数据类型与数据名称，后端返回符合的数据信息 |
| 请求模块 | 积分请求 | 前端发送常规访问，后端执行常规访问申请 |
| 免费请求 | 前端发送免费访问申请，后端执行免费访问申请 |
| 消息模块 | 获取所有消息 | 前端发送请求，后端返回所有已发布数据 |
| 获取消息数量 | 前端发送请求，后端返回已发布数据数量 |
| 同意免费访问请求 | 前端发送请求，后端执行免费访问控制（批准） |
| 拒绝免费访问请求 | 前端发送请求，后端执行免费访问控制（拒绝） |

* 1. **系统智能合约设计**
     1. **主要solidity函数及功能**

|  |  |
| --- | --- |
| **函数** | **函数功能** |
| register | 用户注册功能，向区块链添加用户数据 |
| login | 用户登录 |
| changeSelfInfo | 更改用户信息 |
| changeDataInfo | 更改共享数据信息 |
| isExistDataName | 共享数据名审核 |
| createData | 共享数据发布 |
| deleteData | 共享数据删除 |
| checkList | 检查不可访问清单 |
| checkCredit | 检查使用者积分 |
| shareRequest | 申请常规访问 |
| freeRequest | 申请免费访问 |
| freeApprove | 批准免费访问 |
| freeDeny | 拒绝免费访问 |

# 数据共享平台详细设计说明书

**1.      需求概述**

在大数据时代，各行各业对数据的需求量激增，而单一机构所能获取的数据资源有限，为避免造成信息孤岛问题，数据共享的需求不断扩大。传统的数据共享系统往往通过中心化的数据平台管理数据，这一模式需要具有强大公信力的机构进行担保，否则难以保证共享数据和用户信息的安全，并会带来主导权之争、隐私保护难、数据确权难、无激励机制等一系列问题，导致参与方缺乏主动共享数据的意愿，因此通过建设统一的数据平台实现数据共享的案例非常少。

区块链技术为解决上述问题提供了新的思路。通过引入区块链的分布式账本、密码学技术、智能合约及激励机制，可以有效保证数据的安全性，帮助用户达成共识，提高用户积极性。因此，基于区块链的数据共享访问控制平台有着很大的发展前景。

本数据共享平台可以提供一个供用户通过积分发布和下载数据的平台。用户在系统中可以进行注册用户，发布数据，下载数据，查看数据等操作。本系统后台是基于FISCO-BCOS的区块链系统，在保证稳定性和高效性的同时能够提供足够的安全性，可以保护用户的隐私和交易的安全性。

**2.      需求分析**

**2.1.  宏观分析**

**2.1.1.     政策利好推动区块链技术及其产业高速发展**

中国通信工业协会区块链专委会轮值主席于佳宁接受第一财经记者采访时表示，《指导意见》的出台，意味着国家对于区块链行业发展顶层设计已基本完成，同时明确了区块链行业未来10年的发展目标，进一步引领区块链产业走高质量发展之路，预示着区块链发展的“政策红利期”来临。 大数据时代和区块链时代都在来临，数据分享系统也在逐步构建和完善，整体而言，现在中国网民迫切需要一个C2C交易的数据交易平台。

**2.1.2.     中国消费者购买力正在增强，同时网民对线上数据的需求的在上涨**

近年来，中国网民对数据的消费不断上升，由于实体经济占比慢慢减少，线上资源数据越来越被人们需要，因此，给线上的数据共享平台的发展提供机会。

**2.1.3.     互联网渗透增大，物流设施趋于完善，大数据和云计算能力增强实现产业升级**

据网信部数据报告显示：截至2020年12月，我国网民规模达9.89亿，手机网民规模达9.86亿，互联网普及率达70.4%。其中，40岁以下网民超过50%，学生网民最多，占比为21.0%。说明网民对数据的需求很大，同时，大数据和云计算技术增强实现减轻了软件及应用开发与运维成本负担，随着人工智能的助力，算法推荐等为消费者提供精准推荐，使得数据共享平台的用户体验变好。

**2.2.  国内平台分析**

数据共享平台在中国乃至世界都是长期存在的，伴随近些年互联网技术的成熟以及人们对互联网的使用率在增加，国内的数据共享平台越来越多，比如百度文库、网上图书馆等等。 **2.3.  用户分析**

人们现在对精神物质的需求越来越多，人们越来越享受新知识或相关数据，线上数据越来越被人需要，催生出了巨大的线上共享数据市场。**2.4.  竞争态势分析**

1.       供应商：数据发布者

2.       购买者：数据需求者

3.       替代品：线下数据共享平台

4.       同行竞争：百度文库

**2.5.  市场分析**

**市场前景**

1、数据开放共享进程加速

近年来，诸如政务、电力等行业已经逐渐开始在数据共享方面进行有益的探索和尝试。未来，推动数据开放共享的政策举措将不断加强。此外，区块链、同态加密、差分隐私、多方安全计算、零知识证明等技术也将进一步取得突破，从而推进数据共享和流通进程。

2、数据应用在生活中

近年来，网上的资源有时候总是很难找，需要通过很多渠道才能找到，如果有一

个大的数据共享平台将会很好的解决这个问题。

**3.      现状总结**

**3.1.  行业可行性**

**去中心化云诞生的背景**

在2020年4月，国家发改委首次明确了新基建的范围，在信息基础建设这一项，把 5G、物联网、工业互联网、卫星互联网列为通信网络基础设施，把人工智能、云计算、区块链列为新技术基础设施。把数据中心、智能计算中心列为算力基础设施。其中区块链赫然在列。

这就是 Crust 的诞生背景。与其说这是选对了方向，我们更愿意认为这是在对的时间恰巧做了件对的事情。

在互联网浩浩汤汤发展了 50 年进入瓶颈期的时候，在 12 年前比特币诞生并不断掀起强风巨浪探索更广泛应用的这个时候，去中心化云存储的解决方案应运而生。

去中心化云存储是真实需求吗？我们有百度网盘和阿里云，也有 iCloud 和亚马逊云等各种成熟的云服务提供商，为什么要用去中心化的云？

讨论这个问题之前，先拿最近争议很大的丰巢收费来举例：丰巢引起争议的关键点在于，用户认为丰巢一边拿着物业给到优惠入场价格，一边收着快递员的使用费，最后还要收取超过 12 小时未取快递的用户费用，垄断了市场后开始了收割；而丰巢则解释其初衷是为了服务进一步的精细化、改善投递效率，同时也希望推动物流末端市场的迭代。

据公开资料显示，丰巢 2020 年一季度未经审计的营业收入 3.34 亿元，亏损 2.45 亿元，去年营收 16.14 亿元，亏损 7.81 亿元；自 2015 年 6 月成立，截至去年底，亏损合计约 20 亿元。

虽然行业增长迅速，丰巢的市场规模也在不断上升，然而这不意味着丰巢已经实现盈利。

在被国家邮政局约谈后，最终丰巢决定将免费保存时间提高为18小时，这场争议才逐渐趋于平息。如果用 Crust 的方式构建快递柜，只要小区业主贡献出闲置的仓库或储存柜，只对快递公司和过度超时用户收费（比如超过 7 天），所得费用扣除管理快递柜的人员和水电等成本后，再按资源贡献比例进行分配。这样的设计相比较于丰巢模式主要有两点优势：

提高资源利用率

降低用户使用成本

这种自给自足的方式不需要丰巢这样的第三方来提供服务，自然也就不会存在上面的争议。以上构想虽然偏向理想化，在可行性上并不具备通用性，一方面因为是潜在的管理成本较高，一方面仓库和储存柜这样的资源介质没有量化标准。但是在去中心化云存储这个领域，这样的设计就天然契合。“Code Is Low"的范式下，统一管理，公平量化。公开透明，降本增效。相比较于当前的商业模式，Crust 做的事情就是去掉中间商，平衡服务的价格，最大限度保障用户，快递员和仓储等资源提供方的各自利益。

去中心化云存储，基于闲置存储资源的充分利用，能够降本；分布式的存储节点，就近提供最近端服务，这是增效。

**3.2.  未来发展方向**

**去中心化云的优势**

再回到上面的问题，去中心化云是不是个伪需求？首先云存储是毋庸置疑的一个需求，相比较于中心化云存储，去中心化云存储有如下几点优势：

1. 成本更低：区块链的加密算法对数据去重率的问题有良好的解决能力，同时也能降低数据冗余率。另外区块链的边缘节点架构，对硬件的要求较低，也无需专业的制冷系统，从而降低整个系统的成本，也降低分摊到用户身上的成本。

2. 可用性更高：负载分散到各个节点上，将整体网络的可用性和效率提高了一个档次。

3. 安全性更高：存储数据分布式地存储在各个节点上，相比较于中心化服务器，具备天然的数据可靠性、容灾、抗 DdoS 等安全特性。

4. 隐私保护：存储数据是被切割成多个部分存储在众多节点上，并且进行了加密处理，从机制上保证了数据安全和隐私。

从去中心化存储的场景切入，基于区块链技术，创新的结合可信执行环境技术，以此打造可信、可靠、高效的去中心化云服务生态。

**4.      用户角色分析**

**4.1.  角色**

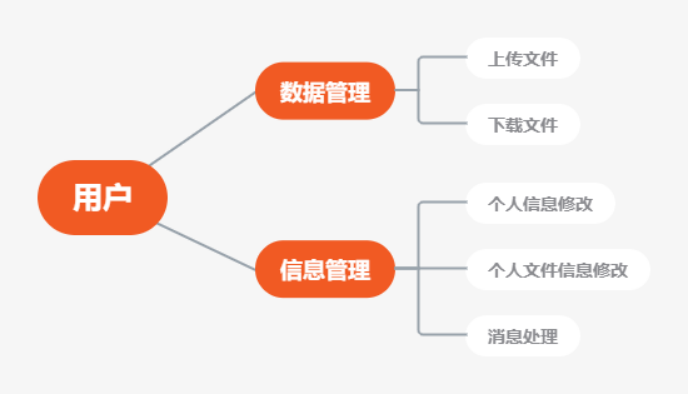
用户角色从整体来看，只有一种普通用户。

**4.1.1.     普通用户**

如从调研来看，用户的主要需求在于资源数据获取。用户能够在我们的平台上传数据，能够在数据共享平台上浏览上传的共享数据、进行访问下载。

**4.2.  参与业务**

普通用户：上传共享数据、浏览所有共享数据、下载共享数据、申请免费访问共享数据、管理个人信息、查看或者更改个人数据信息、处理数据申请的消息。



**5.      功能性需求**

|  |  |  |  |
| --- | --- | --- | --- |
| 功能名称 | 用户角色 | 描述 | 备注 |
| 账号注册 | 普通用户 | 用户输入用户名，密码，地址进行账户注册 |  |
| 登录 | 普通用户 | 各类用户使用自己的用户名和密码进行登录。 |  |
| 用户信息大厅详情 | 普通用户 | 用户查看所有共享数据信息，根据关键词搜索或者分类搜索查看特定文件信息。 |  |
| 用户上传共享文件及相关信息 | 普通用户 | 用户输入共享数据信息及共享文件，在区块链上创建共享信息，并向用户发放奖励。 |  |
| 用户常规访问 | 普通用户 | 用户在数据大厅申请常规访问，系统检查用户是否具备访问该数据的资格，若具备则向用户返回共享数据，扣减用户相应共享积分，向数据拥有者发放数据共享奖励 |  |
| 用户免费访问 | 普通用户 | 用户在数据大厅申请免费访问，检查用户是否具备访问该数据的资格，若具备则向数据拥有者发送免费共享申请，若数据拥有者选择同意，则向用户返回共享数据。 |  |
| 用户个人信息查看 | 普通用户 | 用户进入个人信息中心，用户可查看个人详细信息，同时可修改自身的用户详细信息。 |  |
| 用户查看文件信息 | 普通用户 | 用户进入个人信息中心，用户可查看个人详细信息，同时可修改自身的用户详细信息。。 |  |
| 用户查看消息 | 普通用户 | 用户进入个人中心，查看消息列表，对免费访问请求进行处理。 |  |
| 积分转移 | 普通用户 | 用户选择对象地址，转移积分。 |  |

**6.      非功能性需求**

|  |  |
| --- | --- |
| 性能需求 | 1.系统能够适应400个用户，平均每个会话持续8分钟；  2.生成的web页面，通过速率100KBps在10秒内可以全部下载 |
| 可用性需求 | 正常运行时间:99.9%。 |
| 并发需求 | 能支持至少百个并发请求。 |
| 安全性需求 | 1.所有涉及信息的操作必须登录后才能完成；  2.系统只允许用户查看自己的信息，而不能查看其他用户的资料； |

**7.      系统设计**

**7.1.  基本架构设计**

**7.1.1.     用户注册模块设计**

用户注册模块的实现主要包括AccessControl.sol智能合约。后端负责接受前端输入并调用AccessControl.sol智能合约，AccessControl.sol智能合约审查输入参数的正确性，若正确则在区块链上创建用户，并向前端返回结果。

用户注册模块的智能合约关键函数下表所示。

|  |  |  |
| --- | --- | --- |
| **函数功能** | **函数声明** | **函数参数** |
| **用户名审核** | isExistUserName(string, uint) | 用户名；审核方式 |
| **用户地址审核** | isExistUserAddress(string) | 用户地址 |
| **创建用户** | createUser(string memory, uint8, address) public | 用户名；用户Id；用户地址 |
| **用户名与用户Id映射** | (string => uint8) nameToid | 用户名；用户Id |
| **用户地址与**  **用户Id映射** | (address => uint8) adToid | 用户地址；用户Id |
| **新用户注册**  **事件** | NewUser(uint8, string, address) | 用户Id；用户名；用户地址 |

用户进入用户注册页面后，根据输入框填写用户名、用户密码和用户地址相关信息，点击“注册”向智能合约传递输入参数，并调用相关函数，获得返回结果。

AccessControl.sol智能合约在接受到前端传递的参数后，调用函数isExistUserName()和isExistUserAddress()，核查用户名和用户地址等信息，审核通过后调用createUser()函数在区块链上的用户列表中创建用户，为用户生成用户Id，将用户名与用户Id、用户地址与用户Id对应储存于映射关系nameToid[]和adToid[]中，并触发新用户注册事件NewUser。后端向前端返回调用智能合约的结果，并通过对新用户注册事件的监听获知用户是否成功创建于区块链上。

**7.1.2.     用户登录模块设计**

用户登录模块的实现主要包括Login.js类以及AccessControl.sol智能合约。后端负责接受前端输入并调用AccessControl.sol智能合约，AccessControl.sol智能合约审查输入参数的正确性，并向前端返回结果。

用户注册模块的智能合约关键函数下表所示。

|  |  |  |
| --- | --- | --- |
| **函数功能** | **函数声明** | **函数参数** |
| **用户名与用户Id映射** | (string => uint8) nameToid | 用户名；用户Id |
| **获取用户密码** | users(uint) | 用户Id |

用户进入用户登录页面后，根据输入框填写用户名、用户密码相关信息，点击“登录”向智能合约传递输入参数，并调用相关函数，获得返回结果，若结果正确则跳转至数据大厅页面。

AccessControl.sol智能合约在接受到前端传递的参数后，调用映射关系nameToid，核查用户是否存在，若存在则返回用户Id，并调用users()核查用户密码是否正确，返回调用结果。后端向前端返回调用智能合约的结果，若用户密码正确则跳转至数据大厅界面。

**7.2.  功能实现函数设计**

**7.2.1.     用户详情模块主要功能函数设计**

用户详情模块的实现主要包括AccessControl.sol智能合约。后端根据前端参数调用AccessControl.sol智能合约，AccessControl.sol智能合约调用用户列表及相关函数，并向前端返回结果。

智能合约函数如下

|  |  |  |
| --- | --- | --- |
| **函数功能** | **函数声明** | **函数参数** |
| **获取用户信息** | users(uint) | 用户Id |
| **用户名审核** | isExistUserName(string, uint) | 用户名；审核方式 |
| **用户地址审核** | isExistUserAddress(string) | 用户地址 |
| **修改用户信息** | alterUser(string memory string memory, uint8, address) | 用户名；用户Id；用户地址 |
| **用户名与用户Id映射** | (string => uint8) nameToid | 用户名；用户Id |
| **用户地址与**  **用户Id映射** | (address => uint8) adToid | 用户地址；用户Id |
| **用户信息修改事件** | AlterUser(uint8, string, address) | 用户Id；用户名；用户地址 |

用户详情模块可分为两部分：

（1）用户管理

用户进入用户管理页面后，调用智能合约查看用户详细信息、消息列表。

AccessControl.sol智能合约在接受到前端传递的参数后，调用users()获取用户信息，返回调用结果。后端向前端返回调用智能合约的结果，用户点击按钮时调用相关函数跳转相关页面。

（2）用户修改

用户打开用户修改页面后，根据输入框填写用户名、用户密码和用户地址、用户描述等相关信息，点击“确认”向智能合约传递输入参数，并调用相关函数，获得返回结果，若结果正确将关闭用户修改页面。

AccessControl.sol智能合约在接受到前端传递的参数后，调用函数isExistUserName()和isExistUserAddresss()，核查用户名、用户地址是否与其他用户重复，审核通过后调用alterUser()函数在区块链上的用户列表中修改用户，将用户名与用户Id、用户地址与用户Id对应映射关系nameToid[]和adToid[]更新，并触发用户信息修改事件AlterUser。UserDetail.js类向前端返回调用智能合约的结果，并通过对用户信息修改事件的监听获知用户信息是否在区块链上成功修改。

**7.2.2.     共享数据发布模块主要功能函数设计**

共享数据发布模块的实现主要包括AccessControl.sol智能合约。后端负责接受前端输入并调用AccessControl.sol智能合约，AccessControl.sol智能合约根据输入参数调用相关函数，修改共享数据列表，并向前端返回结果。

共享数据发布模块关键函数如下。

|  |  |  |
| --- | --- | --- |
| **函数功能** | **函数声明** | **函数参数** |
| **共享数据名审核** | isExistDataName(string) | 共享数据名 |
| **创建共享数据** | createData(uint8, uint8, uint8 [] memory, string memory, string memory, string memory, string memory) | 共享数据Id；共享数据所需积分；共享数据不允许访问用户Id集合；共享数据名；共享数据描述；共享数据存储地址 |
| **共享数据名与共享数据Id映射** | (string => uint8) nameToid | 共享数据名；共享数据Id |
| **新数据发布事件** | NewData(uint8, string) | 共享数据Id；共享数据名 |

AccessControl.sol智能合约在接受到前端传递的参数后，调用函数isExistDataName()，核查数据名是否与其他数据重复，审核通过后调用createData()函数在区块链上的共享数据列表中创建共享数据，为共享数据生成共享数据Id，将数据名与数据Id对应储存于映射关系nameToid[]中，并触发新数据发布事件NewData。后端向前端返回调用智能合约的结果，并通过监听共享数据发布事件获知共享数据是否在区块链上成功创建。

**7.2.3.     共享数据详情模块主要功能函数设计**

共享数据详情模块的实现主要包括AccessControl.sol智能合约。后端根据前端参数调用AccessControl.sol智能合约，AccessControl.sol智能合约根据参数调用共享数据列表及相关函数，并向前端返回结果。

共享数据详情模块关键函数

|  |  |  |
| --- | --- | --- |
| **函数功能** | **函数声明** | **函数参数** |
| **获取数据信息** | datas(uint) | 共享数据Id |
| **数据名审核** | isExistDataName(uint) | 共享数据Id |
| **修改数据信息** | alterData(uint8, uint8, uint8 [] memory, string memory, string memory, string memory, string memory) | 共享数据Id；共享数据所需积分；共享数据不允许访问用户Id集合；共享数据名；共享数据描述；共享数据存储地址 |
| **数据名与数据Id映射** | (string => uint8) nameToid | 共享数据名；共享数据Id |
| **数据信息修改事件** | AlterData(uint8, string) | 共享数据Id；共享数据名 |

共享数据详情模块可分为三部分：

（1）数据大厅

AccessControl.sol智能合约在接受到前端传递的参数后，调用users()获取用户信息，返回调用结果。后端向前端返回调用智能合约的结果，用户点击按钮时调用相关函数跳转相关页面。

（2）数据详情

AccessControl.sol智能合约在接受到前端传递的参数后，调用datas()获取共享数据信息，返回调用结果。后端向前端返回调用智能合约的结果，用户点击按钮时调用相关函数跳转相关页面。

（3）数据修改

AccessControl.sol智能合约在接受到前端传递的参数后，调用函数isExistDataName()，核查数据名是否与其他数据重复，审核通过后调用alterData()函数在区块链上的共享数据列表中修改共享数据，将数据名与数据Id对应映射关系nameToid[]更新，并触发用户信息修改事件AlterData。后端向前端返回调用智能合约的结果，并通过监听共享数据信息修改事件获知共享数据信息是否在区块链上成功修改。

**7.2.4.     常规访问模块主要功能函数设计**

常规访问模块的实现主要包括AccessControl.sol智能合约。后端负责根据前端参数调用AccessControl.sol智能合约，AccessControl.sol智能合约调用共享数据列表及相关函数，并向前端返回结果。

常规访问详情模块关键函数

|  |  |  |
| --- | --- | --- |
| **函数功能** | **函数声明** | **函数参数** |
| **获取数据信息** | datas(uint) | 共享数据Id |
| **检查不可访问清单** | checkList(uint8, uint8) | 用户Id；共享数据Id |
| **检查使用者积分** | checkCredit(uint8, uint8) | 用户Id；共享数据Id |
| **申请常规访问** | shareRequest(uint8, uint8) | 用户Id；共享数据Id |
| **常规访问成功事件** | RequestApproved(uint8；uint8) | 用户Id；共享数据Id |

用户打开访问申请页面后，点击“常规访问”调用智能合约，检查用户是否访问资格，若具备则进行常规访问申请，调用相关函数，扣减用户共享积分，向数据拥有者发放数据共享奖励，返回共享数据文件，同时获取智能合约端处理结果。

AccessControl.sol智能合约在接受到前端传递的参数后，调用checkList()函数检查用户是否具备访问资格，若具备则调用checkCredit()函数检查用户积分是否足够，若足够则调用shareRequest()函数执行相关操作，返回共享数据文件，并触发常规访问成功事件RequestApproved。后端向前端返回调用智能合约的结果，并通过监听常规访问成功事件获知常规访问控制信息是否写入到区块链上。

**7.2.5.     免费访问模块主要功能函数设计**

免费访问模块的实现主要包括AccessControl.sol智能合约。后端负责接受前端输入并调用AccessControl.sol智能合约，AccessControl.sol智能合约根据输入参数调用消息列表，并向前端返回结果。

免费访问详情模块关键函数

|  |  |  |
| --- | --- | --- |
| **函数功能** | **函数声明** | **函数参数** |
| **获取消息信息** | msg(uint) | 消息编号 |
| **申请免费访问** | freeRequest(uint8, uint8, uint) | 用户Id；共享数据Id；消息编号 |
| **批准免费访问** | freeApproved(uint8, uint8,uint) | 用户Id；共享数据Id；消息编号 |
| **拒绝免费访问** | freeDenied(uint8, uint8, uint) | 用户Id；共享数据Id；消息编号 |
| **免费访问申请事件** | FreeRequest(uint) | 消息编号 |
| **免费访问批准事件** | FreeApproved(uint) | 消息编号 |
| **免费访问拒绝事件** | FreeDenied(uint) | 消息编号 |

免费访问模块由两部分组成：

（1）免费访问申请

用户打开访问申请页面后，点击“免费访问”调用智能合约，检查用户是否访问资格，若具备则向数据拥有者发送免费访问请求，修改消息列表，并获取智能合约端处理结果。

AccessControl.sol智能合约在接受到前端传递的参数后，调用checkList()函数检查用户是否具备访问资格，若具备则调用freeRequest()函数执行相关操作，并触发免费访问申请事件FreeRequest。后端向前端返回调用智能合约的结果，并通过监听免费访问申请事件获知免费访问信息是否成功写入到区块链上。

（2）免费访问控制

用户打开消息处理页面后，调用智能合约查看消息详细信息，点击“批准”可批准免费访问申请，点击“拒绝”可拒绝免费访问申请，并获取智能合约端处理结果。

AccessControl.sol智能合约在接受到前端传递的参数后，调用msg()获取消息详细信息，“批准”按钮可调用freeApproved()函数/“拒绝”按钮可调用freeDenied()函数更新消息列表、用户列表，并触发免费访问批准/免费访问拒绝事件。后端向前端返回调用智能合约的结果，并通过监听免费访问批准事件/免费访问拒绝事件获知免费访问控制信息是否成功写入到区块链上。

**7.3.  区块链设计**

**7.3.1.     区块链结构**

本系统采用单链两节点单群组的结构，使用pbft共识机制。

用户向节点发起RPC请求以发起交易，节点收到交易后将交易附加到交易池中，打包器不断从交易池中取出交易并通过一定条件触发将取出交易打包为区块。生成区块后，由共识引擎进行验证及共识，验证区块无误且节点间达成共识后，将区块上链。当节点通过同步模块从其他节点处下载缺失的区块时，会同样对区块进行执行及验证。

**7.3.2.     数据传输**

一笔交易（tx1），从客户端上发往某个节点，节点在接收到交易后，会将交易放入自身的交易池（Tx Pool）中供共识去打包。与此同时，节点会将交易广播给其它的节点，其它节点收到交易后，也会将交易放到自身的交易池中。交易在发送的过程中，会有丢失的情况，为了能让交易尽可能的到达所有的节点，收到广播过来交易的节点，会根据一定的策略，选择其它的节点，再进行一次广播。

**8.      后台设计**

**8.1.  环境设置**

Maven：4.0.0

Jdk：OpenJdkpenJdk13

IDE : IntelliJ IDEA

**8.2.  后端架构**

后端的整体架构大约分为3层：数据接口、服务层和Controller层，结构如上图所示。

1、数据接口：在智能合约转化成java文件以后，对其暴露的函数接口进行封装，使接口更加美观，方便服务层进行调用。

2、 服务层：利用数据接口实现需求，使其可以被Controller层所调用

3、 Controller：响应前端的请求，并返回所需的数据

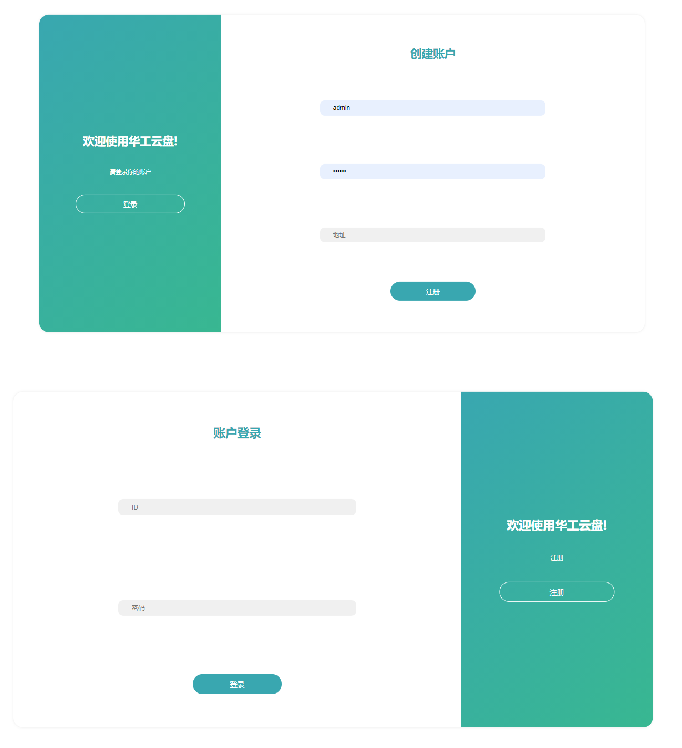
4、Pojo：定义后端的数据类型，方便Controller传参。

5、Result：定义返回给前端的Json结果。

6、Cconfig：定义后端相关配置。

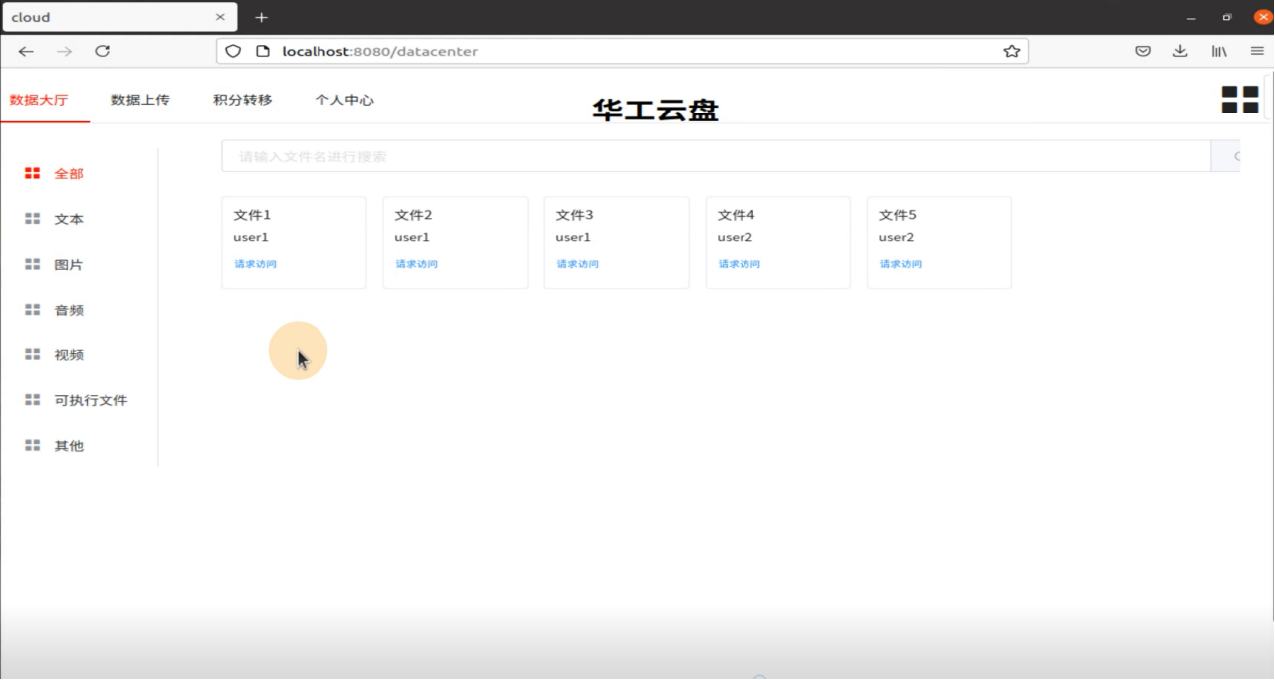
**9.      界面设计**

**登录注册界面**

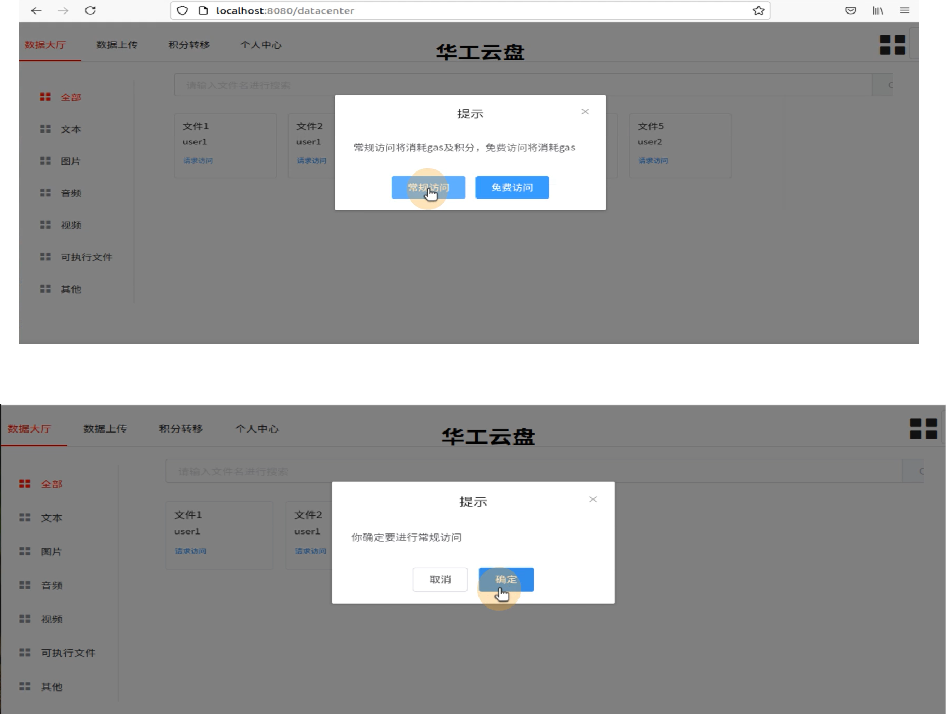


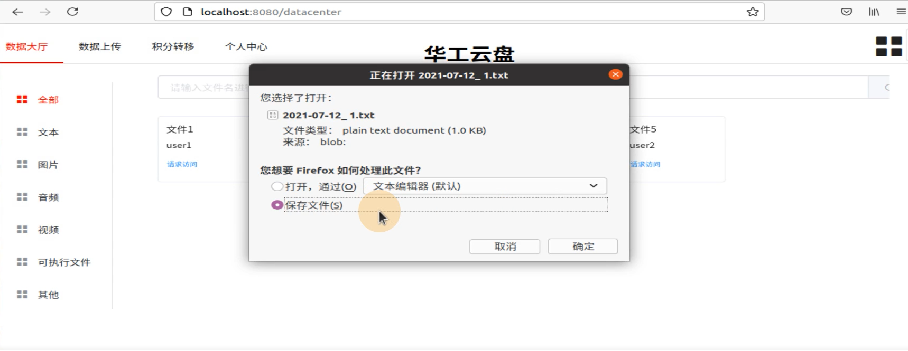
数据大厅界面











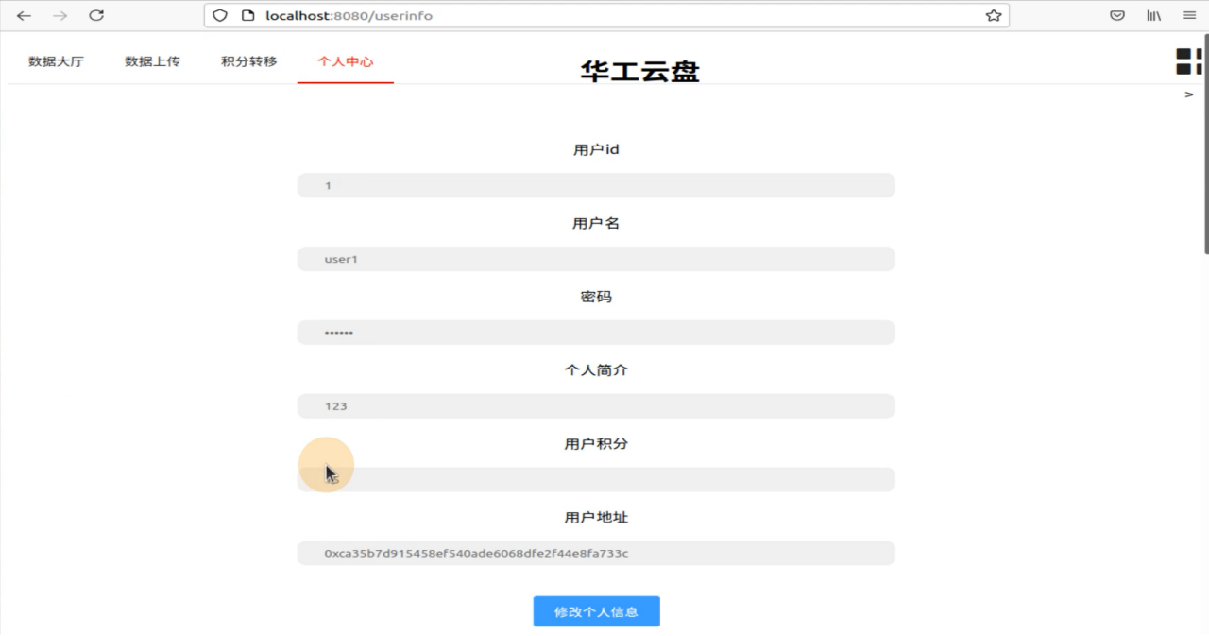
积分转移界面

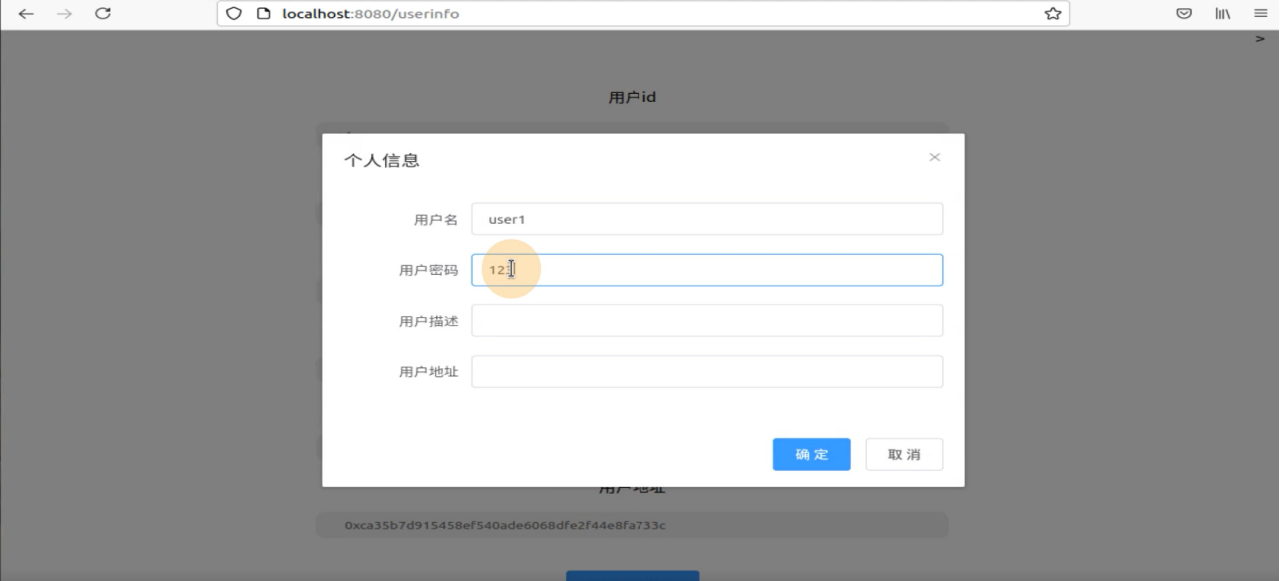


上传文件界面



用户信息管理界面







# 数据共享平台测试报告

**1.      引言**

**1.1.编写目的**

该报告用于对数据共享平台的功能进行性能和压力测试。

**1.2.项目背景与系统简介**

本次测试实验的内容是二手物品交易市场，包含基本的用户注册、用户登录、用户发布新的交易、用户登出等功能。

**1.3.术语**

无

**2.      参考资料**

《LoadRunner使用教程》

开发人员GitHub网站及文档

https://github.com/withstars/Books-Management-System

**3.      测试概要**

**3.1.测试的概要介绍**

测试范围为二手物品交易市场部分功能的稳定性测试和破坏性测试。

**3.2.  用例设计方法**

压力测试用例法

**3.3.  测试环境与配置**

a) 测试机2台 :安装Windows 7 系统， LoadRunner 11.

b) 被测服务器1台：

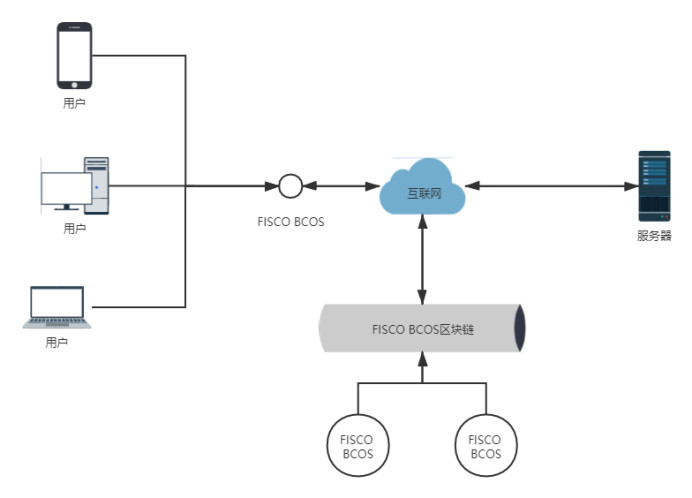
                     CPU：单核，2.4Ghz

                     内存：2G

                     系统：Windows 7

                     应用：二手物品交易市场

物理架构图：



**4.      测试结果与缺陷分析**

**4.1.测试组织**

主要测试人员：刘鼎、刘泽辉、李健、李浩然

**4.2.  测试时间**

2021.6.28 测试项目开始

2021.6.28 执行测试，编写测试报告

**4.3.  测试版本**

最终版

**4.4.  覆盖分析**

**4.4.1.黑盒测试设计**

1. 注册测试

1、基本事件流

（1）输入用户名、密码和地址

（2）确认注册

（3）注册提示成功

2、备选事件流

备选事件流1：

（1）输入的用户名、密码和地址某几项为空

（2）提示输入不能为空

1、 设计场景：通过需求用例的主事件流和备选事件流的组合给出不同的场景

|  |  |  |
| --- | --- | --- |
| 场景编号 | 场景名称 | 场景描述 |
| 1 | 注册成功 | 基本事件流 |
| 2 | 注册失败 | 基本事件流-备选流1 |

2、 设计测试用例标准覆盖场景

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 注册信息是否为空 | 预期结果 |
| FISCO01 | 1 | V | 成功注册 |
| FISCO02 | 2 | I | 提示输入不能为空 |

3、 根据测试用例标准给出具体的测试数据

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 注册信息是否为空 | 预期结果 |
| FISCO01 | 1 | 是 | 成功注册 |
| FISCO02 | 2 | 否 | 提示输入不能为空 |

1. 登录测试

1、基本事件流

（1）输入用户名、密码

（2）确认登录

（3）登录提示成功

2、备选事件流

备选事件流1：

（1）输入的用户名或密码为空

（2）提示输入不能为空

备选事件流2：

（3）输入的用户名不存在

（4）提示用户名不存在

备选事件流3：

（5）输入的密码错误

（6）提示密码错误

1、 设计场景：通过需求用例的主事件流和备选事件流的组合给出不同的场景

|  |  |  |
| --- | --- | --- |
| 场景编号 | 场景名称 | 场景描述 |
| 1 | 登录成功 | 基本事件流 |
| 2 | 登录失败 | 基本事件流-备选流1 |
| 3 | 登录失败 | 基本事件流-备选流2 |
| 4 | 登录失败 | 基本事件流-备选流3 |

2、 设计测试用例标准覆盖场景

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 输入信息为空 | 用户名是否存在 | 密码是否正确 | 预期结果 |
| FISCO11 | 1 | V | V | V | 成功注册 |
| FISCO12 | 2 | I |  |  | 提示输入不能为空 |
| FISCO13 | 3 | V | I |  | 提示用户不存在 |
| FISCO14 | 4 | V | V | I | 提示密码错误 |

3、 根据测试用例标准给出具体的测试数据

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 输入信息为空 | 用户名是否存在 | 密码是否正确 | 结果 |
| FISCO11 | 1 | 否 | 是 | 是 | 成功注册 |
| FISCO12 | 2 | 是 |  |  | 提示输入不能为空 |
| FISCO13 | 3 | 否 | 否 |  | 提示用户不存在 |
| FISCO14 | 4 | 否 | 是 | 否 | 提示密码错误 |

1. 上传文件测试

1、基本事件流

（1）输入文件信息

（2）选择文件

（3）上传文件

（4）确认上传

（5）提示上传成功

2、备选事件流

备选事件流1：

（1）输入的文件信息某几项为空

（2）提示输入不能为空

备选事件流2：

（3）为上传文件

（4）提示上传文件不能为空

1、 设计场景：通过需求用例的主事件流和备选事件流的组合给出不同的场景

|  |  |  |
| --- | --- | --- |
| 场景编号 | 场景名称 | 场景描述 |
| 1 | 上传成功 | 基本事件流 |
| 2 | 上传失败 | 基本事件流-备选流1 |
| 3 | 上传失败 | 基本事件流-备选流2 |

2、 设计测试用例标准覆盖场景

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 上传信息是否存在空 | 文件是否为空 | 预期结果 |
| FISCO21 | 1 | V | V | 成功上传 |
| FISCO22 | 2 | I |  | 提示输入不能为空 |
| FISCO23 | 3 | V | I | 提示文件不能为空 |

3、 根据测试用例标准给出具体的测试数据

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 上传信息是否存在空 | 文件是否为空 | 预期结果 |
| FISCO21 | 1 | 否 | 否 | 成功上传 |
| FISCO22 | 2 | 是 |  | 提示输入不能为空 |
| FISCO23 | 3 | 否 | 是 | 提示文件不能为空 |

1. 用户信息修改测试

1、基本事件流

（1）点击修改用户信息按钮

（2）输入用户修改信息

（3）确认修改

（4）注册提示成功

2、备选事件流

备选事件流1：

（1）输入的用户名、密码和地址某几项为空

（2）提示输入不能为空

1、 设计场景：通过需求用例的主事件流和备选事件流的组合给出不同的场景

|  |  |  |
| --- | --- | --- |
| 场景编号 | 场景名称 | 场景描述 |
| 1 | 修改成功 | 基本事件流 |
| 2 | 修改失败 | 基本事件流-备选流1 |

2、 设计测试用例标准覆盖场景

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 修改信息是否为空 | 预期结果 |
| FISCO01 | 1 | V | 成功修改 |
| FISCO02 | 2 | I | 提示输入不能为空 |

3、 根据测试用例标准给出具体的测试数据

|  |  |  |  |
| --- | --- | --- | --- |
|  | 覆盖的场景 | 修改信息是否为空 | 预期结果 |
| FISCO01 | 1 | 是 | 成功修改 |
| FISCO02 | 2 | 否 | 提示输入不能为空 |

1. 文件信息修改测试

1、基本事件流

（1）点击修改文件信息按钮

（2）输入文件修改信息

（3）确认修改

（4）提示修改成功

2、备选事件流

备选事件流1：

（1）输入的文件信息某几项为空

（2）提示输入不能为空

1、 设计场景：通过需求用例的主事件流和备选事件流的组合给出不同的场景

|  |  |  |
| --- | --- | --- |
| 场景编号 | 场景名称 | 场景描述 |
| 1 | 修改成功 | 基本事件流 |
| 2 | 修改失败 | 基本事件流-备选流1 |

2、 设计测试用例标准覆盖场景

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 修改文件信息是否为空 | 预期结果 |
| FISCO01 | 1 | V | 成功修改 |
| FISCO02 | 2 | I | 提示输入不能为空 |

3、 根据测试用例标准给出具体的测试数据

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 修改文件信息是否为空 | 预期结果 |
| FISCO01 | 1 | 是 | 成功修改 |
| FISCO02 | 2 | 否 | 提示输入不能为空 |

1. 消息处理测试

1、基本事件流

（1）点击同意免费申请按钮

（2）提示同意成功

2、备选事件流

备选事件流1：

（1）点击拒绝免费申请按钮

（2）提示拒绝成功

1、 设计场景：通过需求用例的主事件流和备选事件流的组合给出不同的场景

|  |  |  |
| --- | --- | --- |
| 场景编号 | 场景名称 | 场景描述 |
| 1 | 同意成功 | 基本事件流 |
| 2 | 拒绝成功 | 基本事件流-备选流1 |

2、 设计测试用例标准覆盖场景

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 是否同意 | 预期结果 |
| FISCO01 | 1 | V | 同意请求成功 |
| FISCO02 | 2 | I | 拒绝请求成功 |

3、 根据测试用例标准给出具体的测试数据

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 是否同意 | 预期结果 |
| FISCO01 | 1 | 是 | 同意请求成功 |
| FISCO02 | 2 | 否 | 拒绝请求成功 |

1. 常规访问测试

1、基本事件流

（1）点击常规访问按钮

（2）提示是否下载

（3）点击确定下载

2、备选事件流

备选事件流1：

（1）点击常规访问按钮

（2）提示用户积分不足

1、 设计场景：通过需求用例的主事件流和备选事件流的组合给出不同的场景

|  |  |  |
| --- | --- | --- |
| 场景编号 | 场景名称 | 场景描述 |
| 1 | 访问成功 | 基本事件流 |
| 2 | 访问失败 | 基本事件流-备选流1 |

2、 设计测试用例标准覆盖场景

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 积分是否足够 | 预期结果 |
| FISCO01 | 1 | V | 访问成功 |
| FISCO02 | 2 | I | 访问失败 |

3、 根据测试用例标准给出具体的测试数据

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 积分是否足够 | 预期结果 |
| FISCO01 | 1 | 是 | 访问成功 |
| FISCO02 | 2 | 否 | 访问失败 |

1. 免费访问测试

1、基本事件流

（1）点击免费访问按钮

（2）提示输入申请理由

（3）点击确定键

（4）对方同意

（5）进行常规访问并下载文件

2、备选事件流

备选事件流1：

（1）点击免费访问按钮

（2）提示输入申请理由

（3）点击确定键

（4）对方拒绝

（5）进行常规访问并提示积分不足

1、 设计场景：通过需求用例的主事件流和备选事件流的组合给出不同的场景

|  |  |  |
| --- | --- | --- |
| 场景编号 | 场景名称 | 场景描述 |
| 1 | 免费访问成功 | 基本事件流 |
| 2 | 免费访问失败 | 基本事件流-备选流1 |

2、 设计测试用例标准覆盖场景

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 对方是否通过 | 预期结果 |
| FISCO01 | 1 | V | 免费访问成功 |
| FISCO02 | 2 | I | 免费访问失败 |

3、 根据测试用例标准给出具体的测试数据

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 覆盖的场景 | 对方是否通过 | 预期结果 |
| FISCO01 | 1 | 是 | 免费访问成功 |
| FISCO02 | 2 | 否 | 免费访问失败 |

**4.4.2.**性能和压力**测试**

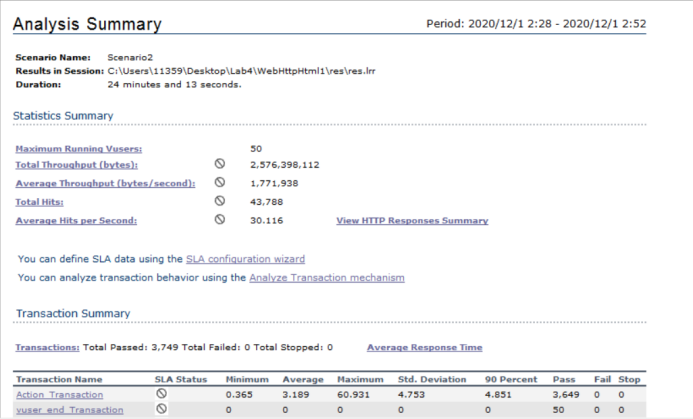
|  |  |  |  |
| --- | --- | --- | --- |
| 测试编号 | 功能/性能项 | 测试案例编号 | 测试通过 |
| 1 | 用户数量50，持续时间3分钟 | 1 | 通过 |

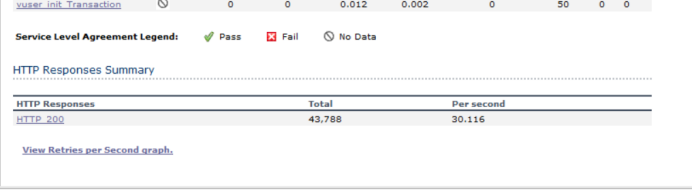
**稳定性测试-文件共享系统**

**测试执行情况与记录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **用例名称** | **用例中事务** | **并发用户数** | **用户并发情况** | **持续时间** |
| **稳定性测试** | **登录-上传文件-登出** | **50** | **初始20个用户，每30s增加10个用户，结束时每30s减少10个用户** | **20min** |
| **平均cpu占有率要求** | **最高cpu占有率要求** | **事务通过率要求** | **平均事务响应时间要求** | **内存使用要求** |
| **60%以下** | **80%**  **以下** | **90%以上** | **8s以下** | **不超过35%** |

LoadRunner进行50个用户场景模拟测试结果收集后，显示的该结果的一个摘要信息，如图所示。概要中列出了场景执行情况。”Statistics Summary”（统计信息摘要）、”Transaction Summary”（事务摘要）以及”HTTP Responses Summary”（HTTP响应摘要）等。以简要的信息列出本次测试结果。





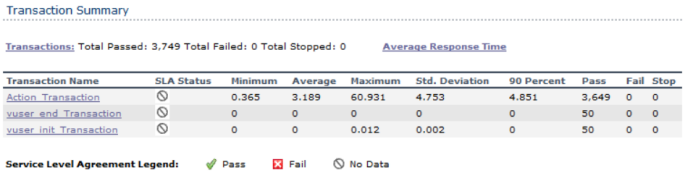
该部分给出了本次测试场景的名称、结果存放路径及场景的持续时间，如上图所示。从该图可知，本次测试从2：28开始，到2：52结束，共历时24min13s。

统计信息摘要



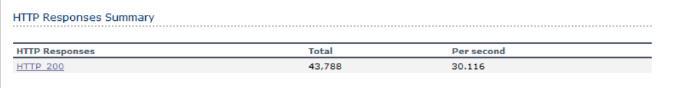
该部分给出了场景执行结束后并发数、总吞吐量、平均每秒吞吐量、总请求数、平均每秒请求数的统计值。由图可知，本次测试运行的最大并发数为50，总吞吐量为2,576,398,112字节，平均每秒的吞吐量为1,771,938字节，总的请求数为43,788，平均每秒的请求为30.116。

事务摘要



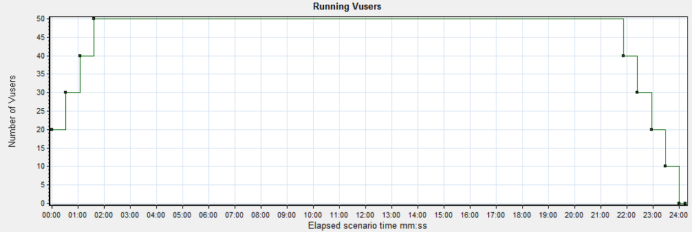
该部分给出了场景执行结束后相关 Action 的平均响应时间、通过率等情况，由图可得本次测试事务通过率为100%。

HTTP响应摘要



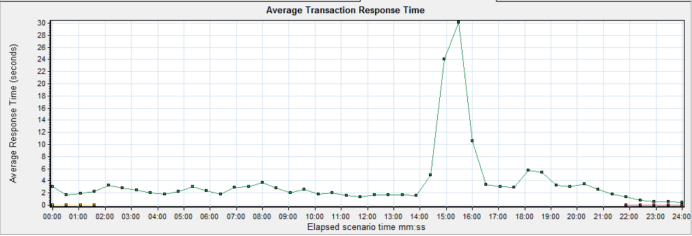
该部分显示在场景执行过程中， 每次 HTTP 请求发出去的状态。从图中可以看到，在本次测试过程中 LoadRunner 共模拟发出了43,788次请求（与“统计信息摘要”中的“总点击次数”一致），其中“ HTTP 200 ”的是 43,788 次，说明在本次过程中，经过发出的请求全部分都能正确响应了（“ HTTP 200 ”表示 请求被正确响应）。

\* 运行Vuser



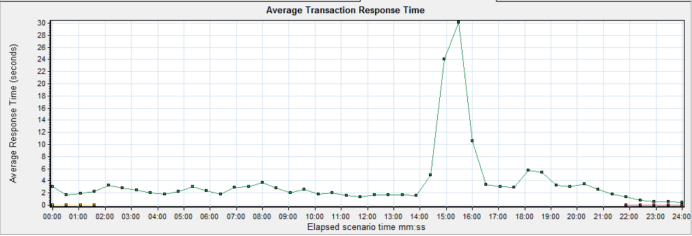
该图反应系统形成负载过程，随着时间的推移，虚拟用户逐渐增加，在1：30分时，达到了负载峰值 50 个虚拟用户， 负载生成大约是每30分钟增加 10个用户， 峰值负载持续20分钟。

\*　平均事务响应时间



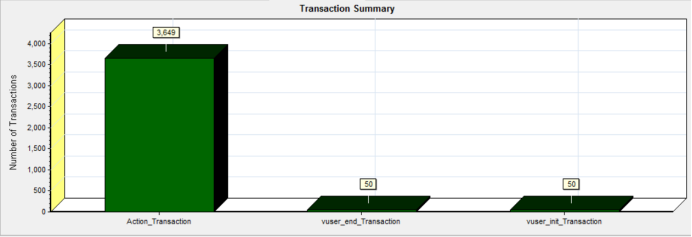
由图可知， Action的平均时间是3.189秒，最大响应时间是60.931次/秒。

\* 每秒点击数



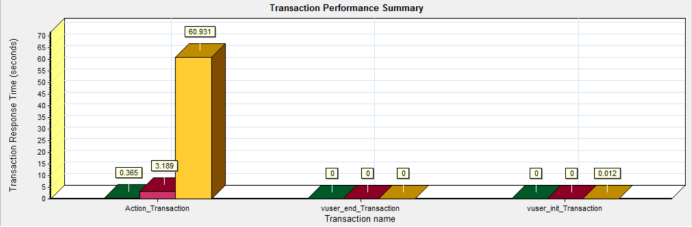
每秒点击数提供当前负载中对系统所产生的点击量记录，每一次点击相当于对服务器发出一 次请求，该数据越大越好，从图中可以看出，在15min左右每秒点击数突变，其他时间较为平稳，最高为37.063次/s，平均为30.116次/s。

\* 事务摘要



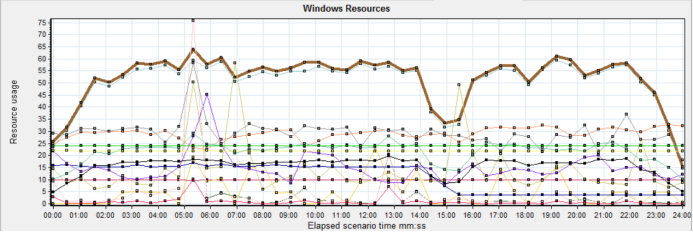
从图中可以看出， Action 操作一共有 3649 次成功，0 次失败。

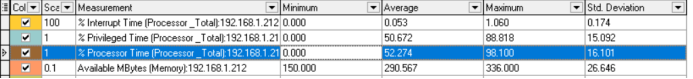
\* 事务性能摘要



由图可知， Action 事务最大值是 60.931s，平均值是 3.189s，最小值是 0.365s，相差大，说明此时系统不够稳定。

\* Windows资源





从图中可以看出， CPU 使用率（Processor Time (Processor \_Total）的平均值为： 52.274% 、 根据本次性能测试要求的： CPU 使用率不超过 80%，CPU平均使用率不超过60%，内存剩余不低于20%。可以看出CPU最高使用率不满足要求，其他指标满足要求。

**5.      测试建议与结论**

**5.1.  建议**

根据以上测试数据，系统在50个用户并发情况下，系统平均CPU占有率，事务通过率，事务平均响应时间，内存使用率均满足用例要求，最高CPU占有率不满足用例要求，本次测试用例不通过。由于系统平均CPU占有率较高，最高CPU占有率不满足要求，CPU系统资源成为系统比较明显的瓶颈，应升级CPU并提供CPU平台，以提升CPU性能。

**5.2.  测试结论**

测试完成目标，可以结束。

# 数据共享平台系统项目总结

本次区块链项目实训中，我们组的课题为基于FISCO-BCOS的数据共享平台。这个项目与之前做过项目的最大区别便是在后端需要将数据发送到FISCO-BCOS节点处理与储存。由于大家都是在实训过程中才开始接触并学习使用FISCO区块链系统，对Web3j SDK也并不熟悉，因此在项目刚开题时我们进度缓慢。在老师相继介绍了Java SDK的特性和用法，我们才确定了项目的基本框架：前端使用Vue+Element开发网页，后端采用Springboot开发工具，以合约和SDK相结合的方式与FISCO的交互。

我们此次实现的是一种基于区块链技术的数据共享系统，将用户信息及共享数据信息以交易数据的形式储存于区块链中，保障了数据的可追溯性和不可篡改性，可有效提高用户对基于区块链的数据共享系统的信任度。同时，项目利用了共享积分的概念，通过共享积分引导共享数据的访问控制，用以进行数据共享系统内的用户激励，鼓励用户发布更多的优质数据资源。在此基础上我们设计了基于智能合约的访问控制机制，通过智能合约完成七个功能模块的逻辑执行，从而提供了一个去中心化的具备用户激励功能的数据共享环境。最后我们实现了完整的基于区块链的数据共享系统访问控制机制，并对各功能模块进行了测试，结果表明实现成果达到了设计要求。

这次实训时间较为紧迫，我们人员也不算充足，但在大家的努力下我们仍完成了项目，虽然功能可能还不够丰富，但基本达到我们的设定需求。

我们这次项目针对数据共享系统中的中心化服务模式、访问控制机制设缺少对用户的引导激励等问题的解决做出了贡献，但由于自身水平、研究时间所限，在设计解决方案的过程中，仍然遗留了一些问题暂未解决，这些问题也是未来的改进和优化的着力点，共包括以下几个方面：

（1）访问控制机制中共享积分的获取方式和使用规则还不够完善，而现实中一些知识共享社区已经对具有类似功能的激励积分设计了较为成熟的使用规则，如何在区块链环境下更好的发挥共享积分的用户激励功能仍需要进一步的探索。

（2）对共享数据本身传输过程的设计较为简单，如何结合数据共享系统存储部分给出更加安全高效的解决方法仍需进一步的研究。