

区块链期末项目报告

周朗 18342142 谢正雄 18342106 曾涛煜 18342007

一、项目背景

《基于区块链的供应链金融平台》

二、实现功能

基本功能：

功能一：实现采购商品—签发应收账款 交易上链。例如车企从轮胎公司购买一批轮胎并签订应收账款单据。

功能二：实现应收账款的转让上链。例如轮胎公司从轮毂公司购买一笔轮毂，便将于车企的应收账款单据部分转让给轮毂公司。轮毂公司可以利用这个新的单据去融资或者要求车企到期时归还钱款。

功能三：利用应收账款向银行融资上链，供应链上所有可以利用应收账款单据向银行申请融资。

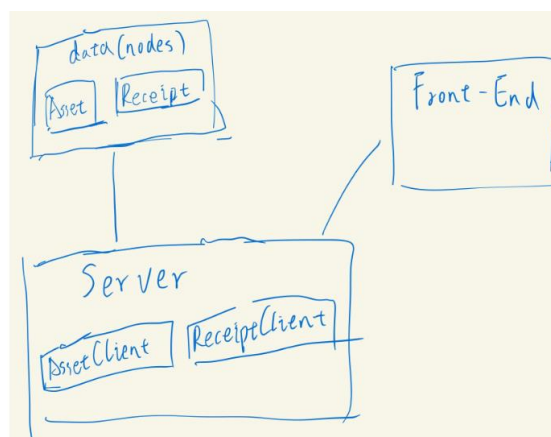
功能四：应收账款支付结算上链，应收账款单据到期时核心企业向下游企业支付相应的欠款。

额外加分项：

比较友好的用户界面。

三、实现设计

具体的债券转移、债券开具等的实现参考之前阶段二中的设计，阶段三主要完成的是前后端的设计、开发，完成内容大致如下：



使用 nodes 来存储数据，即债券的相关信息，front-end（前端）是用户界面，用户可以通过前端查询、修改相关信息，而 server（服务端）完成向 nodes 发送查询、修改请求，并且将从 nodes 中接收到的数据返回给 front-end，展示给用户。

项目主体是根据官网教程[开发第一个区块链应用](https://fisco-bcos-documentation.readthedocs.io/zh_CN/latest/docs/tutorial/sdk_application.html)编写，用到

的合约主要有自带的 Asset 以及二阶段中编写的 Receipt，API 文档也有提供。

前端设计：

前端就是给用户界面，负责将服务端传输的信息展示给用户查看，并且可以通过向后端发送用户的请求，从而实现对数据库中的数据进行增删查改，并且更新界面等功能，也正是由于前端的存在，将用户与服务端隔绝开，使用户无法直接对服务端进行操作。

本项目使用了 Angular 框架，提供了登录、注册和债券管理页面。Angular 是一款非常优秀的前端高级 JS 框架，这一类框架可以轻松构建 SPA 应用程序 通过指令扩展了 HTML，通过表达式绑定数据到 HTML。具有减少了页面上的 DOM 操作等有点。

前端界面会在后面的运行结果中展示。

服务端设计：

服务器使用 Java 编写，使用了 JDK 自带的轻量级服务器 `com.sun.net.httpserver.HttpServer`，虽说原来代码直接生成的是 Spring 以及 Bean 容器，使用 SpringMVC 可能会比较好交互，但因为不太熟悉该框架，增加学习成本，且并不需要复杂的功能，所以选择原生的服务器框架；

`Server.java` 负责整个服务器，包括路由以及处理函数，还整合上述两个对象来进行数据的操作；

`AssetClient.java` 和 `ReceiptClient.java` 负责与上述链端的两个对象进行交互，负责将对应数据的操作简化，给服务器提供更友好的接口。

链端设计：

链端储存着 Asset 和 Receipt 两张表，然后调用阶段二编写的 `Receipt.sol` 和 `Finance.sol` 对其进行增删查改，起到数据库的作用。

```
//Asset 表结构
// 资产管理表, key : account, field : asset_value
// | 资产账户(主键) | 资产金额 |
// |-----|-----|
// | account | asset_value |
// |-----|-----|
//

//Receipt 表结构
// 收据管理表, key : id, field : debtor_account debtee_account amount
// | 单号(主键) | 债务人 | 债主 | 金额 |
// |-----|-----|-----|-----|
// | id | debtor_account | debtee_account | amount |
// |-----|-----|-----|-----|
```

部分代码如下，具体实现请查看源代码。

```
返回值:
参数一：成功返回0，账户不存在返回-1
参数二：第一个参数为0时有效，资产金额

*/
function select(string memory account) public view returns(int256, uint256) {
    // 打开表
    TableFactory tf = TableFactory(0x1001);
    Table table = tf.openTable("t_asset");
    // 查询
    Entries entries = table.select(account, table.newCondition());
    uint256 asset_value = 0;
    if (0 == uint256(entries.size())) {
        return (-1, asset_value);
    } else {
        Entry entry = entries.get(0);
        return (0, uint256(entry.getInt("asset_value")));
    }
}
```

四、 运行方式

前端运行方式：

在相应文件夹中 npm install，获取相关依赖文件，然后 npm start 运行；

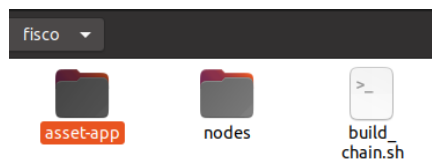
链端运行方式：

在 fisco 文件夹中运行 bash ./nodes/127.0.0.1/start_all.sh (由于链端过大, 因此没有 push 到 GitHub 上, 因此请根据

https://fisco-bcos-documentation.readthedocs.io/zh_CN/latest/docs/installation.html 中的教程的前四步构建)；

服务端运行方式：(好像路径有中文会报错)

在链端运行之后，将 asset-app 移动到与链端相同的文件夹，如下：



执行：

```
# 假设我们将asset-app放在~/fisco目录下 进入~/fisco目录
$ cd ~/fisco
# 创建放置证书的文件夹
$ mkdir -p asset-app/src/test/resources/conf
# 拷贝节点证书到项目的资源目录
$ cp -r nodes/127.0.0.1/sdk/* asset-app/src/test/resources/conf
# 若在IDE直接运行，拷贝证书到resources路径
$ mkdir -p asset-app/src/main/resources/conf
$ cp -r nodes/127.0.0.1/sdk/* asset-app/src/main/resources/conf
```

将证书等拷贝到项目中，然后输入 gradle build 命令编译项目：

```
transfer receipt account success => from: xxx, to: y, value: 3
handx@handx-VirtualBox:~/fisco/asset-app$ gradle build
:compileJava UP-TO-DATE
:processResources UP-TO-DATE
:classes UP-TO-DATE
:jar UP-TO-DATE
:assemble UP-TO-DATE
:compileTestJava NO-SOURCE
:processTestResources UP-TO-DATE
:testClasses UP-TO-DATE
:test NO-SOURCE
:check UP-TO-DATE
:build UP-TO-DATE

BUILD SUCCESSFUL in 1s
4 actionable tasks: 4 up-to-date
handx@handx-VirtualBox:~/fisco/asset-app$
```

然后输入 bash runserver.sh 运行：

```
handx@handx-VirtualBox:~/fisco/asset-app$ bash runserver.sh
deploy Asset success, contract address is 0x0f1f86589ed1d66de30827e32c5a4775899
914e5
deploy Receipt success, contract address is 0x6379f8e2e8f41dbf3063e5c6ae64da9a5
941d1bc
```

五、 运行截图

前端外观：

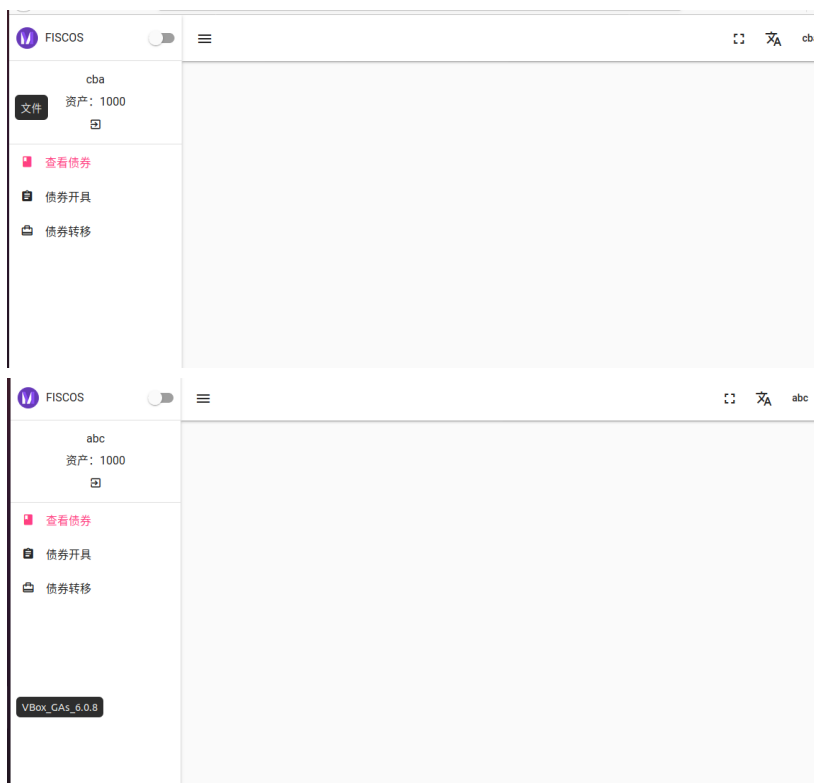


由于没有设置初始金额，因此还需要使用控制台给一开始的账户赋初始资产，创建用户 abc、cba：

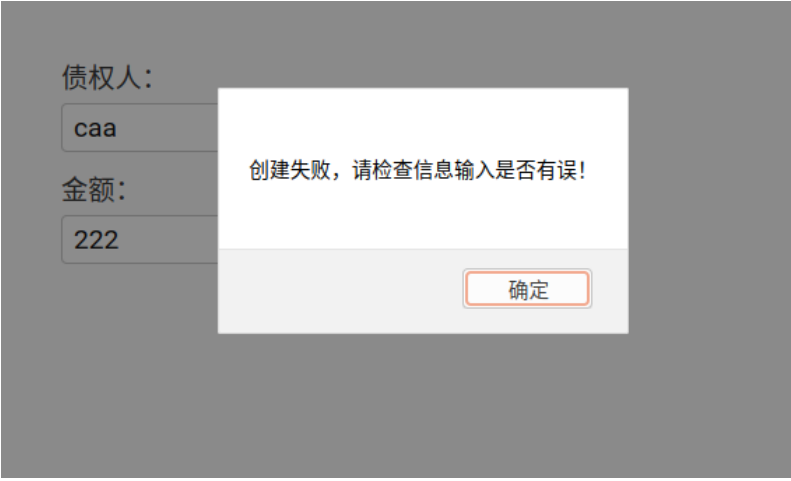
```
logs
{"RegisterEvent": [[0, "cba", 1000]]}

logs
{"RegisterEvent": [[0, "abc", 1000]]}
```

在前端查看：



债券开具：



(能够检查数据)



再创建用户 aaa，并转移 100 元的债券给他：



FISCOS

aaa

资产: 0

查看债券

债券开具

债券转移

单号: 2

债务人: abc 债权人: aaa

金额: 100

(可以看到, 转移成功)