中山大學
SUN YAT-SEN UNIVERSITY

区块链大作业
实验报告

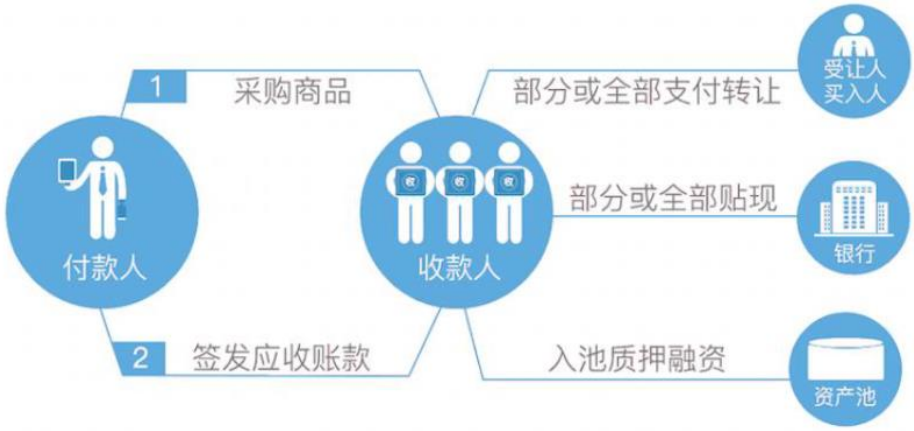
学院名称	：	计算机学院
专业班级	：	18 级软件工程 2 班
小组成员	：	吴振华 18342099 孙浩男 18342087 孙广海 18342086

2021 年 1 月 27 日

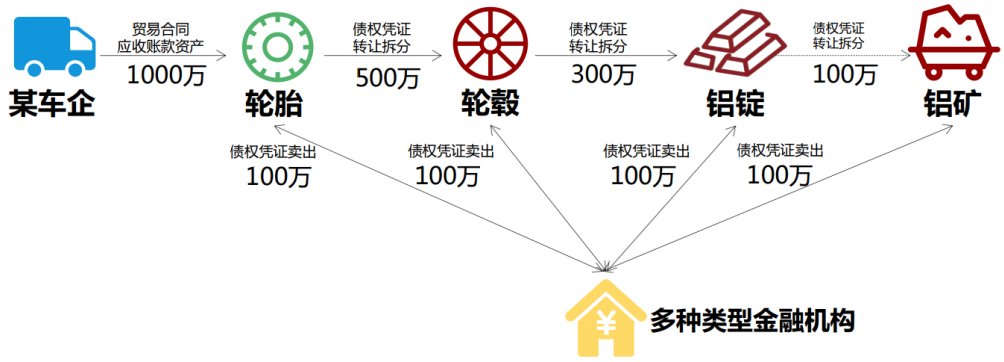
目录

一.项目背景

基于已有的开源区块链系统 FISCO-BCOS（https://github.com/FISCO-BCOS/FISCO-BCOS），以联盟链为主，开发基于区块链或区块链智能合约的供应链金融平台，实现供应链应收账款资产的溯源、流转。



场景介绍：



传统供应链金融：

某车企（宝马）因为其造车技术特别牛，消费者口碑好，所以其在同行业中占据绝对优势地位。因此，在金融机构（银行）对该车企的信用评级将很高，认为他有很大的风险承担的能力。在某次交易中，该车企从轮胎公司购买了一批轮胎，但由于资金暂时短缺向轮胎公司签订了 1000 万的应收账款单据，承诺 1 年后归还轮胎公司 1000 万。这个过程可以拉上金融机构例如银行来对这笔交易作见证，确认这笔交易的真实性。在接下里的几个月里，轮胎公司因为资金短缺需要融资，这个时候它可以凭借跟某车企签订的应收账款单据向金融结构借款，金融机构认可该车企（核心企业）的还款能力，因此愿意借款给轮胎公司。但是，这样的信任关系并不会往下游传递。在某个交易中，轮胎公司从轮毂公司购买了一批轮毂，但由于租金暂时短缺向轮胎公司签订了 500 万的应收账款单据，承诺 1 年后归还轮胎公司 500 万。当轮毂公司想利用这个应收账款单据向金融机构借款融资的时候，金融机构因为不认可轮胎公司的还款能力，需要对轮胎公司进行详细的信用分析以评估其还款能力同时验证应收账款单据的真实性，才能决定是否借款给轮毂公司。这个过程将增加很多经济成本，而这个问题主要是由于该车企的信用无法在整个供应链中传递以及交易信息不透明化所导致的。

区块链+供应链金融：

将供应链上的每一笔交易和应收账款单据上链，同时引入第三方可信机构来确认这些信息的交易，例如银行，物流公司等，确保交易和单据的真实性。同时，支持应收账款的转让，融资，清算等，让核心企业的信用可以传递到供应链的下游企业，减小中小企业的融资难度。

实现功能：

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，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。

二.方案设计

（1）表单设计：

我们的方案需要两个数据库表单，分别存储机构信息以及账单情况，机构信息中使用机构名称 org_name 作为主键，表单中记录密码余额等信息；账单中记录账单的收付款两方，账单金额等信息

```
function createTable() private{
    TableFactory tf = TableFactory(0x1001);
    tf.createTable("Organization", "org_name", "org_pass,org_type,org_tolmoney");
    tf.createTable("Receipt", "money_sender", "money_receiver, status, money, start_data,end_data, info");
}
```

（2） 用户登录：

一个组织（公司/银行）只有完成了组织注册，才能使用系统进行交易、融资等操作。在这个函数中，首先检查输入的合法性，接着检查该组织名是否已被注册，若输入合法且组织名未被注册，则将该组织加入到组织列表中，并返回组织编号。每个组织初始化有 1000 的总资产。

```
function register(string memory name, string memory pass , string memory  org_type) public returns(uint){
    int256 ret=0;
    int256 money=1000;
    bool type;
   if(isEqual(name,"")  ||  isEqual(pass,"") ||  isEqual(org_type, "")     )
       ret=-2;
   if(NameExist(name)==-1)
       ret=-1;
   else{
       Table table = openOrgTable();
       Entry entry = table.newEntry();
       entry.set("org_name", name);
       entry.set("org_pass", pass);
       if(isEqual(org_type,"1")) {type = true;}
       else {type = false;}
       entry.set("org_type", type);
       entry.set("org_tolmoney", money);
       int count = table.insert(name, entry);
       if (count==1) {ret = 0;}
       else {ret=-2;}
   }

   emit RegisterEvent(ret, name, pass, org_type);
   return ret;

}
```

（3）账单转移

首先我们对于账单的收发两方的合法性进行判断，之后查询数据库中的数据，获取用户额度，如果用户的额度以及账单满足条件，则进行账单的更新，否则返回错误，同时更新账单的同时，加入一个账单转移产生的新账单

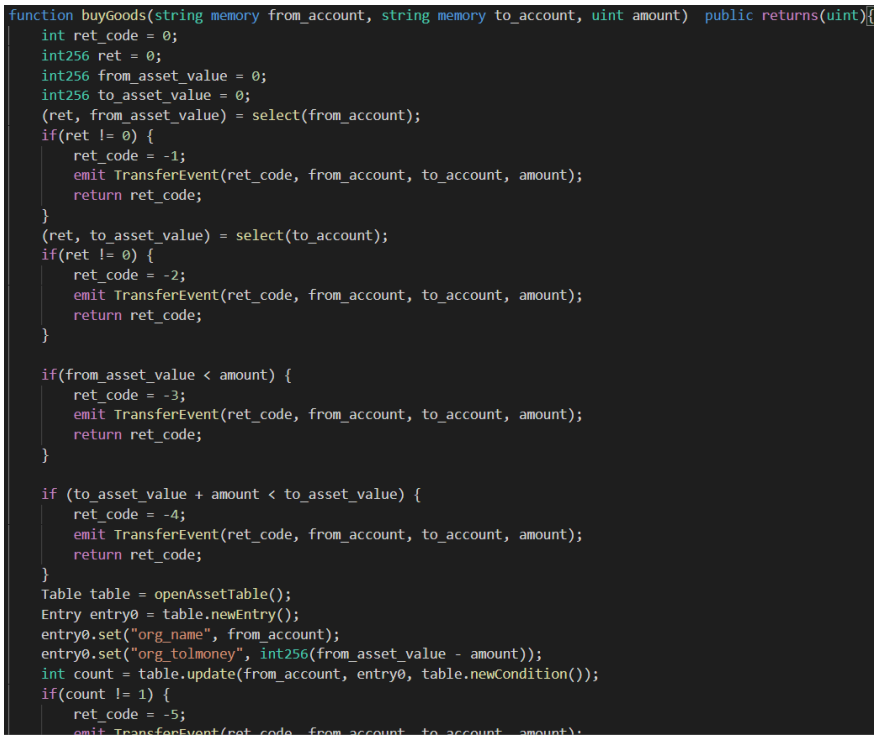
```
function Receipt_trans(string memory company1, string memory company2, uint money, uint id)  public returns(uint, uint){
   //check if its enough
   uint sum = 0;
   uint ret1,ret2;
   Table table1 = openRcpTable();
   Entries entry1 = table1.select(id, table1.newCondition());

   if(isEqual(entry1.getByte32("money_receiver"),company1) && !isEqual(entry1.getByte32("money_receiver"),company2) && (e
       if(entry1.getInt("money") == money){
           Entry entry0 = table1.newEntry();
           entry0.set("money_receiver",company2);
           int count1 = table1.update(id, entry0, table1.newCondition());
           sum += money;
           emit Receipt_transEvent(uint sum, uint id,string  company1, string  company2, uint money);
           return (sum, id);

       }
       else if(entry1.getInt("money") > money){
           int newmoney = entry1[i].getInt("money")-money;
           string memory temp = entry1[i].getByte32("money_sender");
           Entry entry2 = table1.newEntry();
           uint newid = id+1000;
           entry2.set("id", newid);
           entry2.set("money_sender", temp);
           entry2.set("money_receiver", company2);
           entry2.set("status", 0);
           entry2.set("money", newmoney);
           entry2.set("start_data", entry1.getInt("start_data"));
           entry2.set("end_data", entry1.getInt("end_data"));
           entry2.set("info", entry1.getInt("info"));
           int count2 = table.insert(newid, entry2);
           sum += money;
           emit Receipt_transEvent(uint sum, uint newid,string  company1, string  company2, uint money);
           return (sum, newid);
       }
       else {
           uint newid = 0000000;
```

（4）发起交易

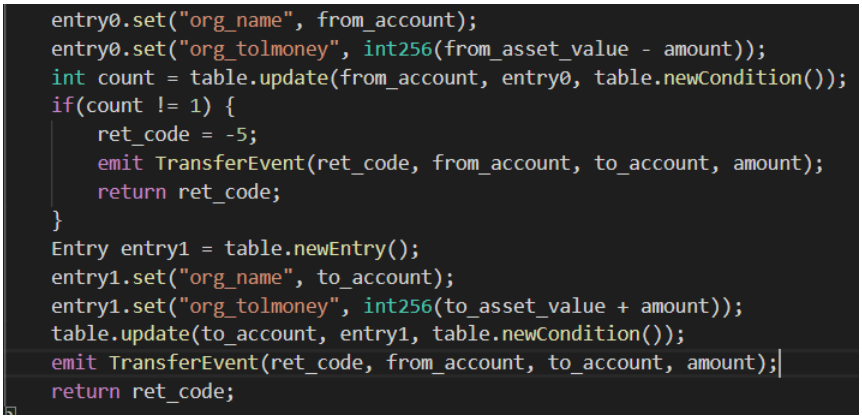
对于账单编号的合法性进行确认，解决重复账单问题，之后对于账单数额以及用户的信用额度进行判断，满足额度条件情况下，新建一条账单信息，加入账单数据库表单中



```
function buyGoods(string memory from_account, string memory to_account, uint amount)  public returns(uint){
    int ret_code = 0;
    int256 ret = 0;
    int256 from_asset_value = 0;
    int256 to_asset_value = 0;
    (ret, from_asset_value) = select(from_account);
    if(ret != 0) {
        ret_code = -1;
        emit TransferEvent(ret_code, from_account, to_account, amount);
        return ret_code;
    }
    (ret, to_asset_value) = select(to_account);
    if(ret != 0) {
        ret_code = -2;
        emit TransferEvent(ret_code, from_account, to_account, amount);
        return ret_code;
    }

    if(from_asset_value < amount) {
        ret_code = -3;
        emit TransferEvent(ret_code, from_account, to_account, amount);
        return ret_code;
    }

    if (to_asset_value + amount < to_asset_value) {
        ret_code = -4;
        emit TransferEvent(ret_code, from_account, to_account, amount);
        return ret_code;
    }
    Table table = openAssetTable();
    Entry entry0 = table.newEntry();
    entry0.set("org_name", from_account);
    entry0.set("org_tolmoney", int256(from_asset_value - amount));
    int count = table.update(from_account, entry0, table.newCondition());
    if(count != 1) {
        ret_code = -5;
        emit TransferEvent(ret_code, from_account, to_account, amount);
```



```
    entry0.set("org_name", from_account);
    entry0.set("org_tolmoney", int256(from_asset_value - amount));
    int count = table.update(from_account, entry0, table.newCondition());
    if(count != 1) {
        ret_code = -5;
        emit TransferEvent(ret_code, from_account, to_account, amount);
        return ret_code;
    }
    Entry entry1 = table.newEntry();
    entry1.set("org_name", to_account);
    entry1.set("org_tolmoney", int256(to_asset_value + amount));
    table.update(to_account, entry1, table.newCondition());
    emit TransferEvent(ret_code, from_account, to_account, amount);
    return ret_code;
```

（5）更新账单

当发生账单分割操作等是，需要更新账单信息，通过读取原有账单信息将最新账单信息写入数据库

```
function updateTransaction(string id, int256 money) public returns(int256, string[]){
    int256 ret_code = 0;
    bytes32[] memory str_list = new bytes32[](2);
    int256[] memory int_list = new int256[](3);
    string[] memory acc_list = new string[](2);
    (int_list, str_list) = select_transaction(id);
    acc_list[0] = byte32ToString(str_list[0]);
    acc_list[1] = byte32ToString(str_list[1]);

    if(int_list[0] == 0) {
        if(int_list[2] < money){
            ret_code = -2;
            emit UpdateTransactionEvent(ret_code, id, money);
            return (ret_code, acc_list);
        }
        Table table = openTransactionTable();
        Entry entry0 = table.newEntry();
        entry0.set("id", id);
        entry0.set("acc1", byte32ToString(str_list[0]));
        entry0.set("acc2", byte32ToString(str_list[1]));
        entry0.set("money", int_list[1]);
        entry0.set("status", (int_list[2] - money));
        int count = table.update(id, entry0, table.newCondition());
        if(count != 1) {
            ret_code = -3;
            emit UpdateTransactionEvent(ret_code, id, money);
            return (ret_code,acc_list);
        }

        int256 temp = transfer(byte32ToString(str_list[0]),byte32ToString(str_list[1]),money);
        if(temp != 0){
            ret_code = -4 * 10 + temp;
            emit UpdateTransactionEvent(ret_code, id, money);
            return (ret_code,acc_list);
        }
```

三.功能测试

- 启动界面

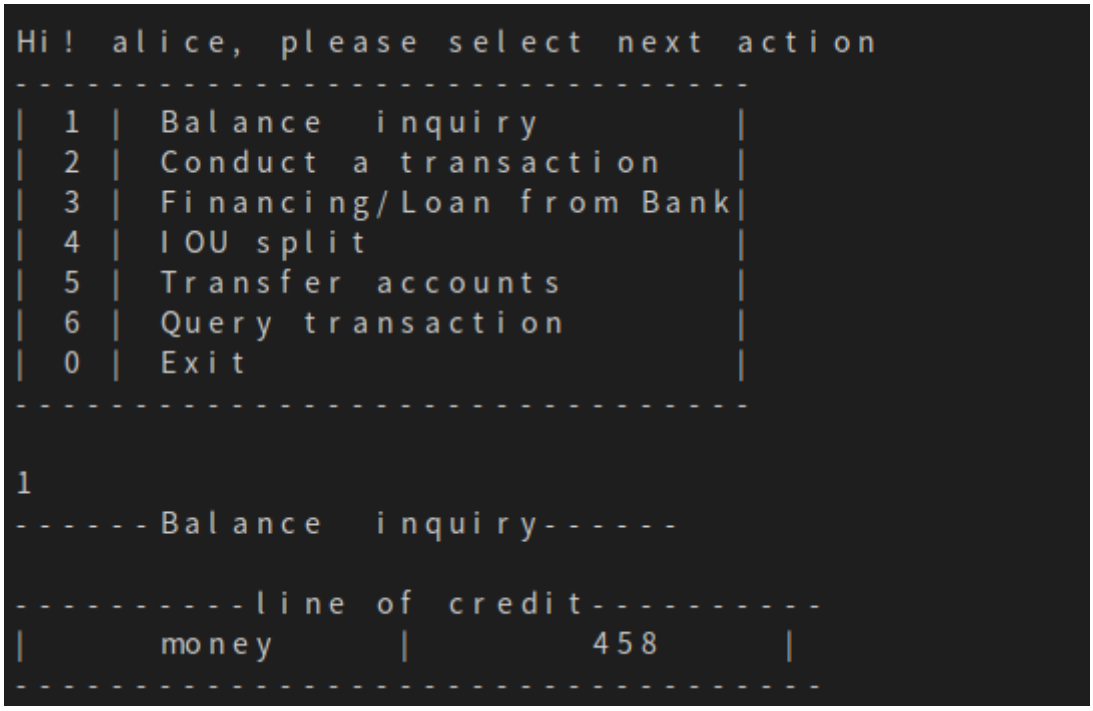
```
--------------------------------------------------------------------------------
#     # ######  #         ####    ####   #     # ######
#     # #       #        #     # #     # ##   ## #
#     # #####   #        #       #     # # ## # #####
# ##  # #       #        #       #     # #    # #
##   ## #       #        #     # #     # #    # #
#     # ######  ######   ####    ####   #     # ######
--------------------------------------------------------------------------------

Please enter number to select function
--------------------------------------------------------------------------------
|        0:LOG IN     |          1:REGISTER         |          2:QUIT             |
--------------------------------------------------------------------------------
```

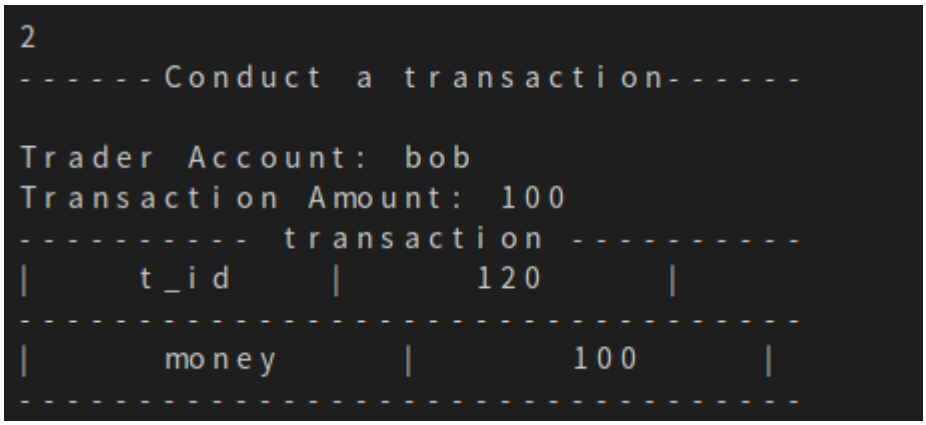
- 注册用户，密码需要二次确认。

```
Please enter number to select function
--------------------------------------------------------------------------------
|        0:LOG IN     |          1:REGISTER         |          2:QUIT             |
--------------------------------------------------------------------------------
1
------REGISTER------
Please enter your ID: alice
Please enter your password:
Please enter again:
```

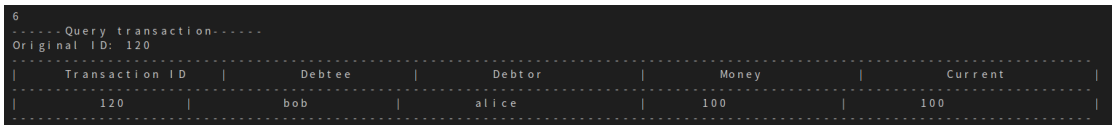
- 登录 alice 账户，查询额度。



- alice 和 bob 完成一笔 100 的交易。



- 查询这笔交易。



- alice 查询额度。

```
1
------Balance  inquiry------

----------line of credit----------
|      money      |        358      |
-------------------------------------
```

- 登录 bob，查询 bob 的额度。

```
Hi! bob, please select next action
--------------------------------
| 1 | Balance  inquiry          |
| 2 | Conduct a transaction     |
| 3 | Financing/Loan from Bank|
| 4 | IOU split                 |
| 5 | Transfer accounts         |
| 6 | Query transaction         |
| 0 | Exit                      |
--------------------------------

1
------Balance  inquiry------

----------line of credit----------
|      money      |        687      |
-------------------------------------
```

- bob 将之前的账单拆分，分给 cart40。

```
4
------IOU Spilt------

Beneficiary: cart
Original ID: 120
Transfer amount: 40
 Split transaction success!
---------- IOU Split ----------
|    old bill  id  |       120     |
-----------------------------------
|    new bill  id  |       121     |
-----------------------------------
|    to user     |       cart     |
-----------------------------------
|     money      |       40     |
-----------------------------------
```

- 查询这两笔交易。

```
6
------Query transaction------
Original ID: 120
-----------------------------------------------------------------------------------------------------
|     Transaction ID     |        Debtee        |        Debtor        |        Money        |        Current        |
-----------------------------------------------------------------------------------------------------
|          120           |          bob         |        alice         |        100          |        60             |
-----------------------------------------------------------------------------------------------------
```

```
6
------Query transaction------
Original ID: 121
-----------------------------------------------------------------------------------------------------
|     Transaction ID     |        Debtee        |        Debtor        |        Money        |        Current        |
-----------------------------------------------------------------------------------------------------
|          121           |          cart        |        alice         |        40           |        40             |
-----------------------------------------------------------------------------------------------------
```

- 登录 cart，向银行融资。

```
3
------Financing/Loan from Bank------

Financing/Loan Amount: 30
---------- transaction ----------
|     t_id     |      122      |
-------------------------------------
|      money      |       30      |
-------------------------------------
Hi! cart, please select next action
---------------------------------
| 1 | Balance inquiry           |
| 2 | Conduct a transaction     |
| 3 | Financing/Loan from Bank  |
| 4 | IOU split                 |
| 5 | Transfer accounts         |
| 6 | Query transaction         |
| 0 | Exit                      |
---------------------------------
```

- 登录回 alice，偿还额度。

```
5
------Transfer/Loan Repayment------

Trade ID: 120
Transfer Amount: 30
 Update transaction success.
Hi! alice, please select next action
--------------------------------
| 1 | Balance inquiry          |
| 2 | Conduct a transaction    |
| 3 | Financing/Loan from Bank|
| 4 | IOU split                |
| 5 | Transfer accounts        |
| 6 | Query transaction        |
| 0 | Exit                     |
--------------------------------
```

- 查询 alice 额度。

```
1
------Balance inquiry------

----------line of credit----------
|      money      |        388      |
------------------------------------
```

- 退出系统。

```
----------------------------------------------------------------------
 ####   ####   ####  #####  #####  #   # ######
#    # #    # #    # #    # #    #  # #  #
#      #    # #    # #    # #####    #   #####
#  ### #    # #    # #    # #    #   #   #
#    # #    # #    # #    # #    #   #   #
 ####   ####   ####  #####  #####    #   ######
----------------------------------------------------------------------
```

四.实验心得

本次项目的过程十分曲折却也是收获了许多，首先整个项目的前提工作是配置环境，在虚拟机中配置 java SDK，gradle 环境，我们小组多次因为 java 版本和 gradle 版本不对而多次心态出现些许问题，所以，一定要查看清楚版本及其对应关系，然后进行环境配置。第二个是进行项目的创建，由于网络上对这类项目的教程极度匮乏，缺少许多交流者和学习者对相关的讨论和材料，只有官方文档的指导，值得令人疑惑的是，官方文档给出的教程是有错

误的，完全按照教程学习是无法让你的项目顺利 build 起来，建议可以投诉和建议官方修改。然后下一个方面是对合约的修改，需要用到 solidity 对数据库进行查询和更新等相应操作，这个在网上是完全没有教程的，就靠自己摸索，所以学习起来比较慢和繁琐，只能一点一点试错，建议老师以后课程可以教学语言类的知识。最后我们小组对完成整个项目的总体体会是体会到区块链（联盟链）有更深的认识，可以用程序来对合约进行设计，希望我们小组会在今后更深入的学区块链，继续努力。