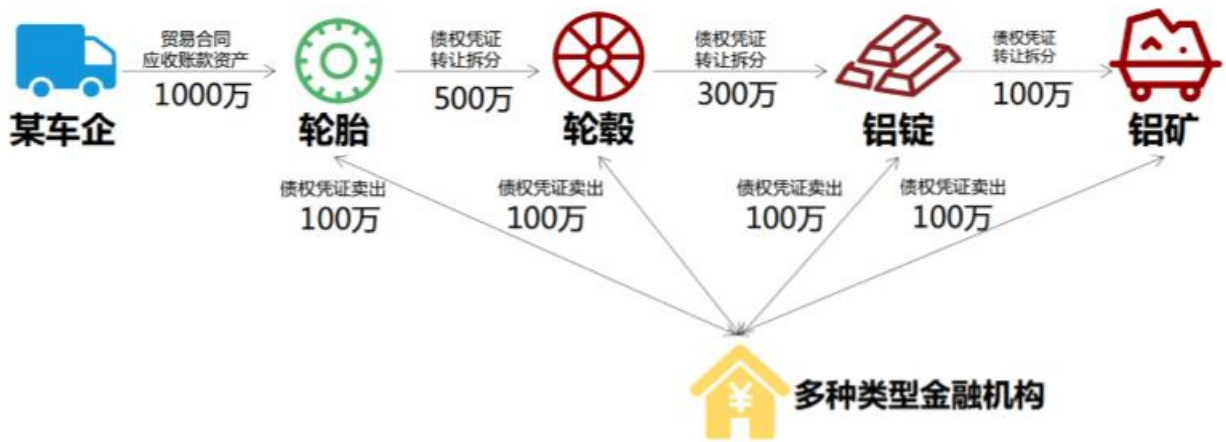
中山大学数据科学与计算机学院本科生实验报告

(2019 年秋季学期)

课程名称：区块链原理与技术　　任课教师：郑子彬

年级	2017	专业（方向）	软件工程
学号	17343030	姓名	高镇
电话	13573565303	Email	1127780786@qq.com
开始日期	2019.12.10	完成日期	2019.12.13

一、 项目背景



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。

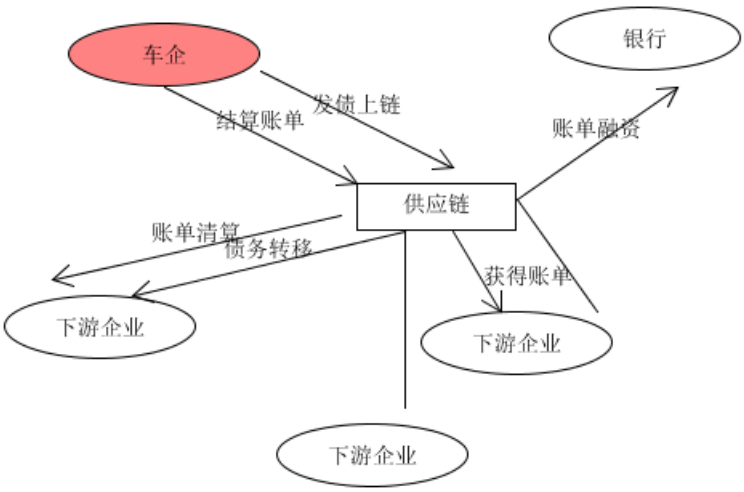
功能四：应收账款支付结算上链，应收账款单据到期时核心企业向下游企业支付相应的欠

款。

二、 方案设计

设计思想：

根据项目背景，区块链在传统企业中起到的创新作用主要在于它构建的信任机制，本次项目实现的主要思路为将企业的信用向下游传递，使得债务在商业运转中更加灵活，因此设计智能合约将车企作为主体，除此之外，银行作为传统背书机构使用债务进行融资，管理企业资产。



实现方案：

主要使用了 express 框架实现前端，后端通过 webase 的节点前置服务提供的接口实现与链端的互动。

参考地址：https://webasedoc.readthedocs.io/zh_CN/latest/docs/WeBASE-

Front/interface.html#id246

存储设计：

```
struct Receipt {
    uint id;
    address from;           //borrower
    address to;             //receiver
    uint mount;
    uint startDate;
    uint endDate;
    bool isPay;
    bool isLoan;
}
```

其中 id 为收据的唯一编号，from，to 分别为开具账单方与收账方，value 为收据的面值，还有账单签发与应收日期，以及是否归还，是否已向银行用于抵押贷款。

```
uint public receiptNum;
address public automobileCompany;
address private bankingHouse;
mapping(address => string)public compNames;
mapping(address => uint)public balances;
mapping(address => int32)public credit;
Receipt[] public receipts;
```

设置变量存储当前所有的收据数目，存储车企和银行的公钥地址，设置 3 个映射，分别为公司名、账面资金、公司信用。

```
//time calculate in days
function SignReceipt(address receiver, uint amount, uint timeLeft)public {

    if(msg.sender == automobileCompany){

        uint timeTemp = now + timeLeft * 1 days;

        receipts.push(Receipt({
            id: receiptNum,
            from: msg.sender,
            to: receiver,
            value: amount,
            startDate: now,
            endDate: timeTemp,
            isPay:false,
            isLoan:false
        }));

        receiptNum ++;
        credit[msg.sender] -= int32(amount);
        credit[receiver] += int32(amount);
    }
}
```

车企可以向下游公司开具具有信用的账单。

```
function receipt_trans(address giveto, uint amount) public { //债务转让
        for(uint i=0; i < r_receive[msg.sender].length; i++) {
            address temp = r_receive[msg.sender][i].to;

            if(r_receive[msg.sender][i].amount > amount) {
                for(uint j=0; j < r_owe[temp].length; j++) {
                    if(r_owe[temp][j].pid == r_receive[msg.sender][i].pid)
                        r_owe[temp][j].amount -= amount;
                        break;
                }
                r_receive[msg.sender][i].amount -= amount;
                r_receive[giveto].push(receipt(giveto,temp,amount,r_receive[msg.sender][i].time,r_receive[ms
                r_owe[temp].push(receipt(giveto,temp,amount,r_receive[msg.sender][i].time,r_receive[msg.send
                break;
            }
            else{
                for(uint j=0; j < r_owe[temp].length; j++) {
                    if(r_owe[temp][j].pid == r_receive[msg.sender][i].pid){
                        delete r_owe[temp][j];
                        for(uint k = j; k < r_owe[temp].length - 1; k++)
                            r_owe[temp][k] = r_owe[temp][k+1];
                        delete r_owe[temp][r_owe[temp].length - 1];
                        r_owe[temp].length -= 1;
                        break;
                    }
                }

                amount -= r_receive[msg.sender][i].amount;


                for(uint k = i; k < r_receive[msg.sender].length - 1; k++)
                    r_receive[msg.sender][k] = r_receive[msg.sender][k + 1];
                delete r_receive[msg.sender][r_receive[msg.sender].length - 1];
                r_receive[msg.sender].length -= 1;//?????
                i --;
                r_owe[temp].push(receipt(giveto,temp,amount,r_receive[msg.sender][i].time,r_receive[msg.send
                r_receive[giveto].push(receipt(giveto,temp,amount,r_receive[msg.sender][i].time,r_receive[ms
            }
```

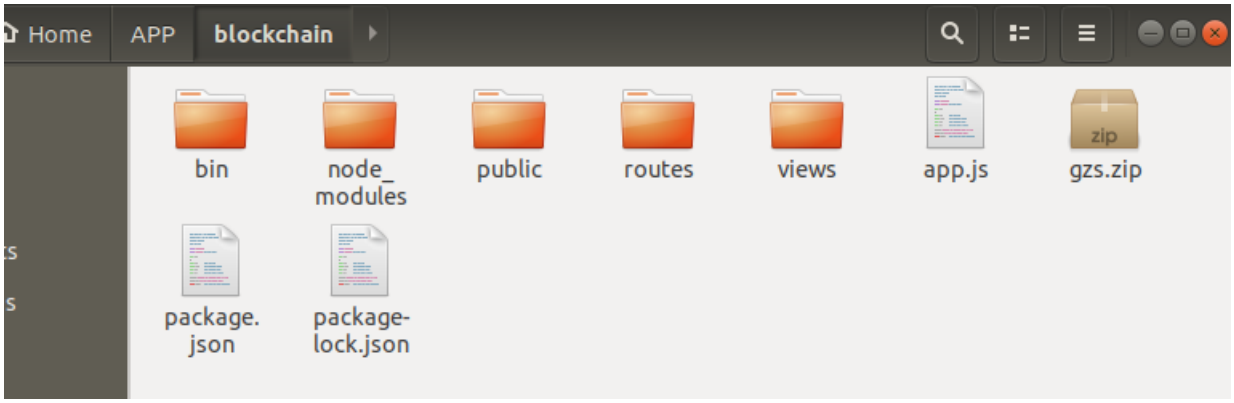
持有的债务可以向下游转移，先遍历所有的债务，找到所属，再生成新债务，可以实现多张债务转移（当某张债务不够转移数目时）。在最终项目中对该函数代码有所简化。

```
function Loan(address from, address to)public{

    if(msg.sender != bankingHouse){
        revert("no have the right to Loan");
    }
    else{
        for(uint i = 0; i < receipts.length; i ++){
            if(receipts[i].from == from && receipts[i].to == to && receipts[i].isPay == false && receipts
                receipts[i].to = bankingHouse;
                receipts[i].isLoan == true;
                credit[to] -= int32(receipts[i].value);
                credit[bankingHouse] += int32(receipts[i].value);
                balances[to] += receipts[i].value;
                balances[bankingHouse] -= receipts[i].value;
            }
        }
    }
}
```

银行放贷在最初的实现为抵押债务增加余额，最终考虑到银行放贷往往不会超过资产价值，因此改为将债务转移给银行，银行发放余额。

```
function PayDebt()public{
    if(msg.sender == automobileCompany){
        for(uint i = 0; i < receipts.length; i ++){
            if(now >= receipts[i].endDate){
                if(receipts[i].isPay == false){
                    balances[receipts[i].to] += receipts[i].value;
                    credit[receipts[i].to] -= int32(receipts[i].value);
                    balances[receipts[i].from] -= receipts[i].value;
                    credit[receipts[i].from] += int32(receipts[i].value);
                    receipts[i].isPay = true;

                }
            }
        }
    }
    else{
        revert("no have the right to pay debt");
    }
}
```

结算只有车企可调用，还清已到期的账单，已偿还的账单不必再偿还。



使用 express 框架搭建前后端，端口默认运行在 3000，view 文件夹下存放视图文件（.ejs），router 下存放路由文件（.js），public 下主要是一些格式文件。

```
router.post('/sign', function (req, res) {
  var amount = req.body.amount,
    receiver = req.body.receiver,
    time = req.body.time;

  var http=require('http');
  var arr = [];
  arr.push(receiver);
  arr.push(amount);
  arr.push(time);
 var post_data = {
    "useAes":false,
    "user":account,
    "contractName":"blockchain",
    "contractAddress":caddr,
    "funcName":"createContract",
    "funcParam":arr,
    "groupId" :"1"
};

var content=JSON.stringify(post_data);

 var options = {
    hostname: '127.0.0.1',
    port: 5002,
    path: '/WeBASE-Front/trans/handle',
    method: 'POST',
    headers:{"Content-type":"application/json"}
  };
console.log("post options:\n",options);
console.log("content:",content);
console.log("\n");

var req = http.request(options, function(res) {

  console.log("statusCode: ", res.statusCode);
  console.log("headers: ", res.headers);

  var _data='';

  res.on('data', function(chunk){
     _data += chunk;
  });
```

以签发为例，通过访问 webase 提供的接口，发送 post 请求，对链端部署的合约进行操作。

序号	中文	参数名	类型	最大长度	必填	说明
1	用户编号	user	String		是	用户编号或者用
2	合约名称	contractName	String		是	
3	合约地址	contractAddress	String		是	
4	方法名	funcName	String		是	
5	方法参数	funcParam	List			JSONArray，对
6	群组ID	groupId	int			
7	是否是加密私钥	useAes	boolean		否	

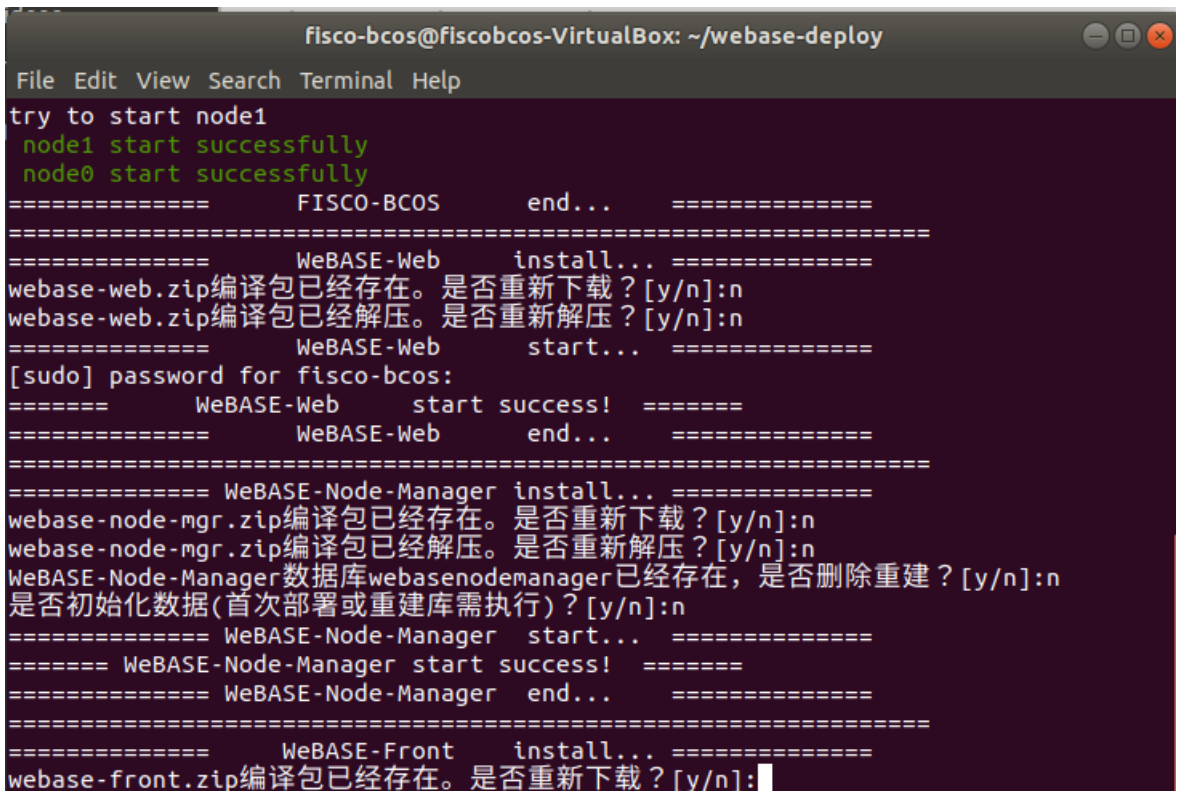
2）数据格式

```
{
    "useAes ":false,
    "user":"700001",
    "contractName":"HelloWorld",
    "contractAddress":"dasdfav23rf213vbcdvadf3bcdf2fc23rqde",
    "funcName":"set",
    "funcParam":["Hi,Welcome!"],
    "groupId" :"1"
}
```

其中所传输的 json 格式按照接口提供的参数组织。

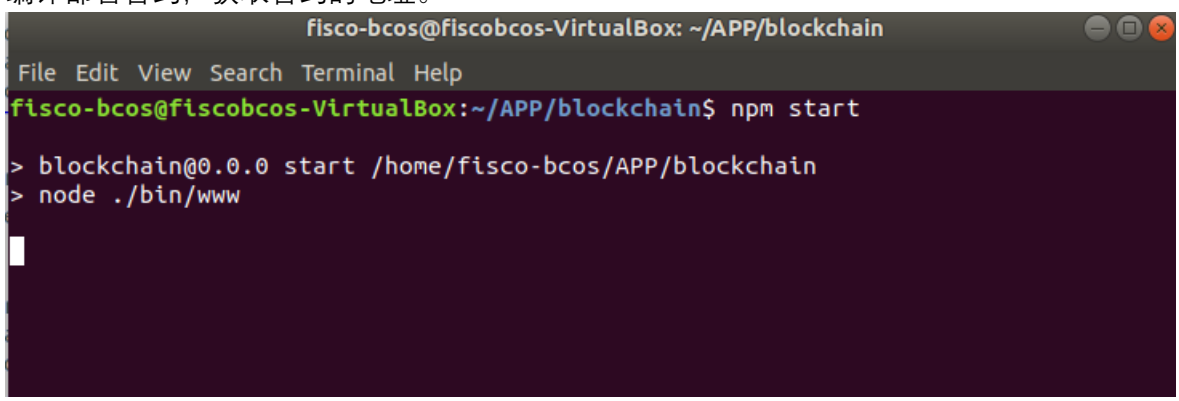
数据流图示例：

三、 功能测试

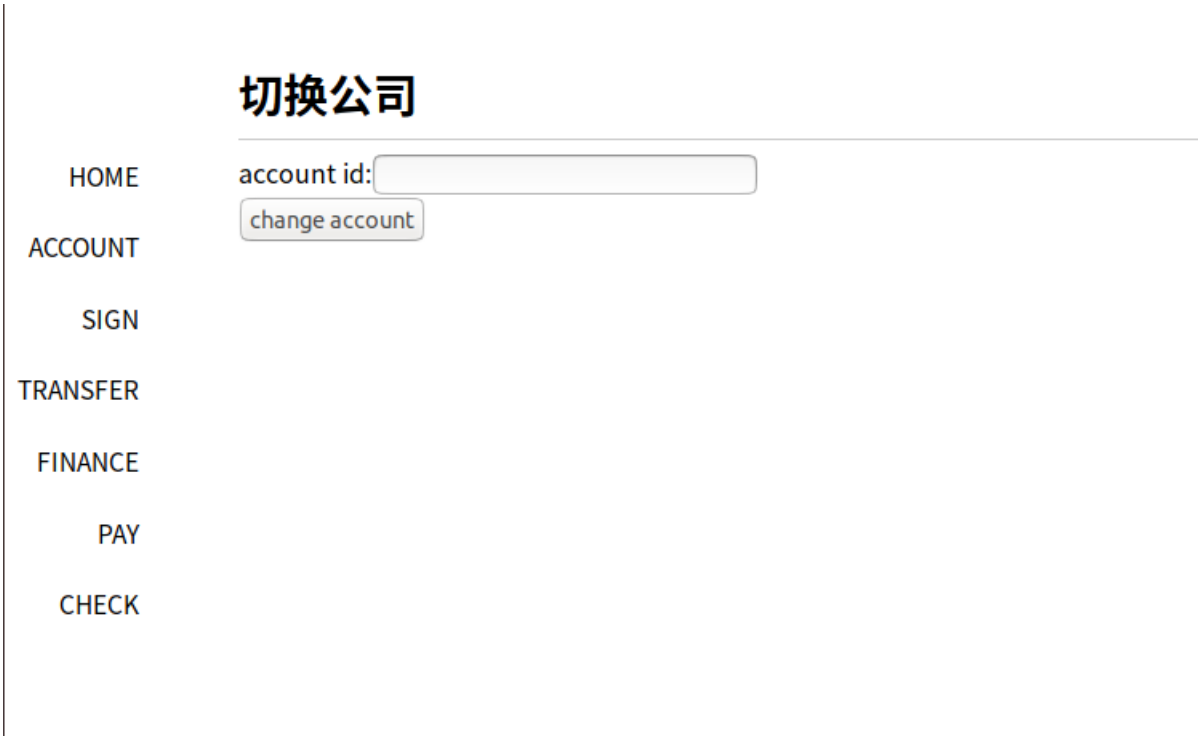


一键部署 webase。

编译部署合约，获取合约的地址。



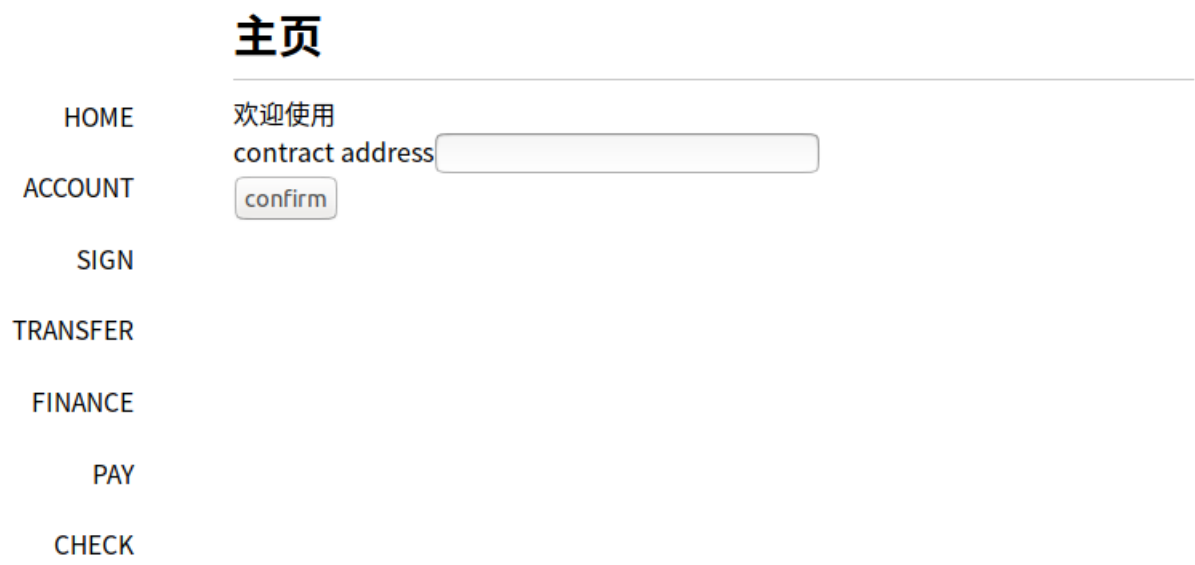
进入项目目录下，启动服务器。在主页输入合约地址。



输入合约调用者的公钥。

车企向下游公司签发账单。更多功能演示详见目录下演示视频。

四、 界面展示



五、 心得体会

车企向下游公司签发账单。更多功能演示详见目录下演示视频。区块链作为当下的热门技术拥有着较高的普及度和良好的前景，特别是在金融行业的应用前景，以其独特的去中心化和不可篡改等特性带动了经济的新增长点。在本学期的学习过程中，从最开始的初步了解，到逐步上手开发，我体会到相关知识灵活且丰富，需要多锻炼。虽然有时会有所困，但解决问题后能力得到很多提升。在课程结束之后，我对区块链知识产生了浓厚兴趣，我想我也会继续学习相关知识。