

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 级	专业 (方向)	软件工程
学号	17343158	姓名	张泽琳
电话	15013041154	Email	457386641@qq.com
开始日期	2019.10.18	完成日期	2019.12.13

一、项目背景

1. 总体介绍

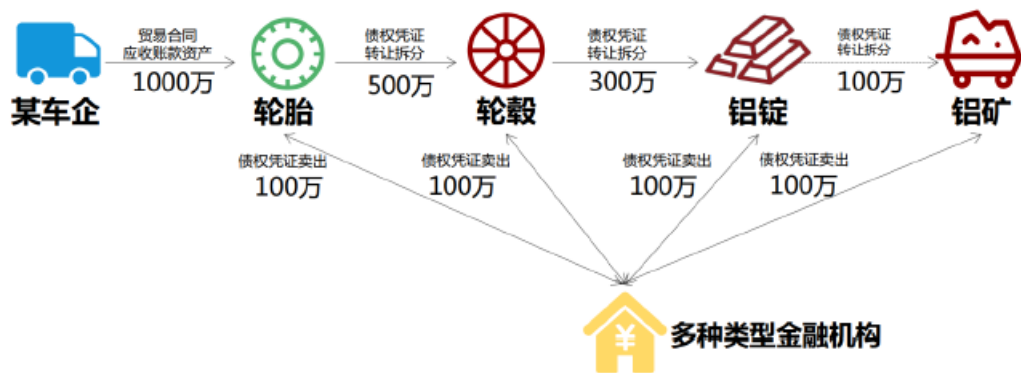
基于区块链、智能合约等，实现基于区块链的供应链金融平台。

2. 必要功能

基于已有的开源区块链系统 FISCO-BCOS (<http://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。在接下来的几个月里，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机构借款，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，让核心契合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欠款。

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. 存储设计

首先定义公司的结构体 `company` 来存储公司名、公司地址、公司类型（0 表示供应链中的企业、1 表示银行）、公司资产。

```
struct company {  
    string cname; // 公司名字  
    address address; // 公司地址  
    bool ctype; // 公司类型  
    uint cbalance; // 公司资产  
}
```

接着定义应收账款的结构体 `receipt` 来存储欠款公司名、收款公司名、欠款公司地址、收款公司地址、具体金额、还款期限以及账单是否有效（0 为无效，1 为有效）。

```
struct receipt {  
    string rfrom; // 欠款人  
    string rto; // 收款人  
    address rfromAddress; //欠款人地址  
    address rtoAddress; //收款人地址  
    uint amount; // 金额  
    uint endtime; // 还款期限  
    bool rstatus;  
}
```

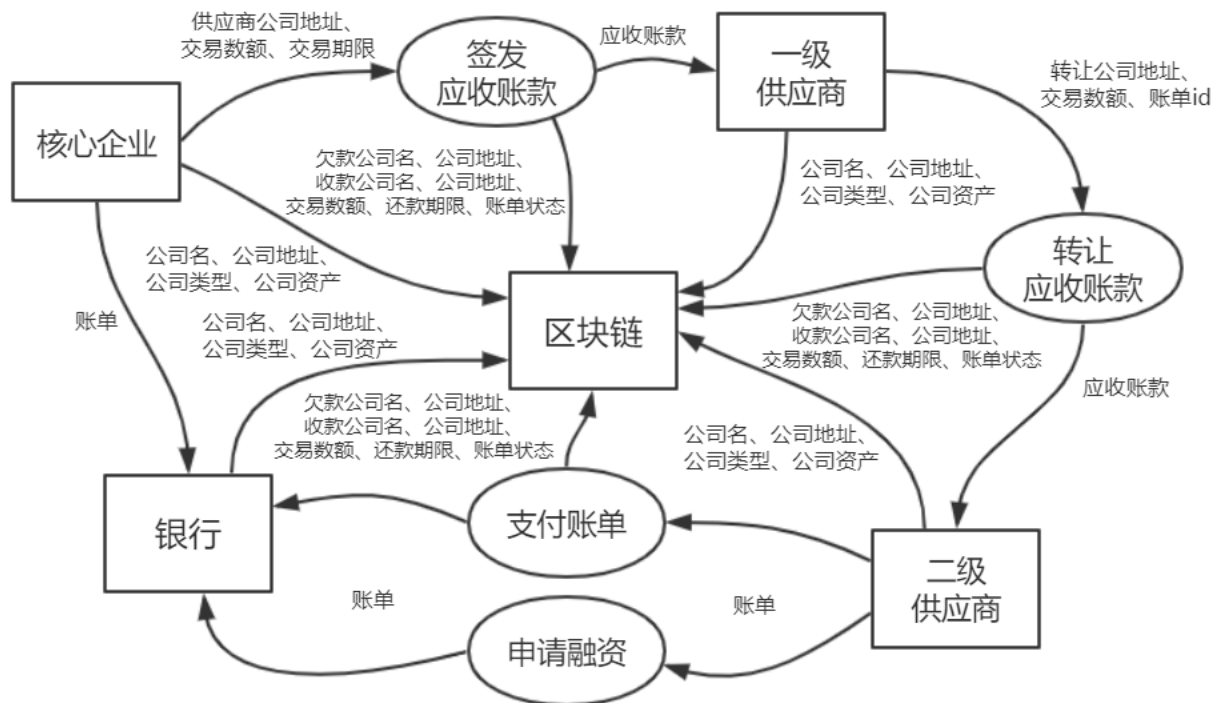
定义 `compantMap` 来存储公司地址和公司之间的映射关系，表示公司地址；`receiptMap` 来存储公司地址和应收账款的映射关系，表示公司当前的收据。

```
mapping(address => company) public companyMap;  
mapping(address => receipt[]) public receiptMap;
```

定义 companyList 来存储所有公司的信息。

```
company[] companyList;
```

2. 数据流图



3. 核心功能介绍

功能一：实现采购商品一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款单据。具体实现代码如下：

```
function signReceipt (address address, uint amount, uint time) public returns (bool, string memory) {
    address seller = msg.sender;
    address buyer = address;
    receiptMap[buyer].push(receipt({
        rfrom: companyMap[seller].cname,
        rto: companyMap[buyer].cname,
        rfromAddress: companyMap[seller].caddress,
        rtoAddress: companyMap[buyer].caddress,
        amount: amount,
        endtime: now + time,
        rstatus: true
    }));
    return (true, "Sign receipt successfully!");
}
```

该函数即为应收账款的签订产生函数，输入参数为卖家的公司地址、交易数额和还款期限，买家发起交易，然后新建一份应收账款，进行相关属性的赋值，并将其存储到 receiptMap 中，最后返回 true 和成功签订应收账款的提示。

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。具体实现代码如下：

```
function transferReceipt (address caddress, uint amount, uint rid) public returns (bool, string memory) {
    address rf = msg.sender;
    address rt = caddress;

    receiptMap[rt].push(receipt({
        rfrom: receiptMap[rf][rid].rfrom,
        rto: companyMap[rt].cname,
        rfromAddress: receiptMap[rf][rid].rfromAddress,
        rtoAddress: companyMap[rt].caddress,
        amount: amount,
        endtime: receiptMap[rf][rid].endtime,
        rstatus: true
    }));
    receiptMap[rf][rid].amount -= amount;
    return (true, "Transfer receipt successfully!");
}
```

该函数即为应收账款的转让函数，输入参数为账单转入的公司地址、交易数额、账单 id，一级供应商发起交易，根据账单 id 从 receiptMap 上找到一级供应商相应的账单，新建一份应收账款，进行相关属性的赋值，并将其存储到 receiptMap 中，将一级供应商相应的账单交易数额减去此次转让交易数额，最后返回 true 和成功转让应收账款的提示，此时账单的状态为有效。

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。具体实现代码如下：

```
function financing (uint rid) public returns (bool, string memory) {
    receiptMap[BANK].push(receipt({
        rfrom: receiptMap[msg.sender][rid].rfrom,
        rto: companyMap[BANK].cname,
        rfromAddress: receiptMap[msg.sender][rid].rfromAddress,
        rtoAddress: companyMap[BANK].caddress,
        amount: receiptMap[msg.sender][rid].amount,
        endtime: receiptMap[msg.sender][rid].endtime,
        rstatus: true
    }));
    receiptMap[msg.sender][rid].rstatus = false;
    companyMap[BANK].cbalance -= receiptMap[msg.sender][rid].amount;
    companyMap[msg.sender].cbalance += receiptMap[msg.sender][rid].amount;
    return (true, "Financing successfully!");
}
```

该函数即为应收账款的融资函数，输入参数为账单的 id，二级供应商轮毂公司向银行发起融资，根据其账单 id 从 receiptMap 上找到二级供应商相应的账单，银行确认账单的有效性，向二级供应商提供资金，二级供应商的资产增加账单中的交易数额，银行的资产减

少相应的金额，最后返回 true 和成功融资的提示，此时账单的状态为无效。

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。具体实现如下：

```
function payDebt() public returns (bool, string memory) {
    for (uint i = 0; i < receiptMap[msg.sender].length; i++) {
        if (receiptMap[msg.sender][i].rstatus == true && receiptMap[msg.sender][i].rtoAddress == msg.sender) {
            companyMap[receiptMap[msg.sender][i].rfromAddress].cbalance -= receiptMap[msg.sender][i].amount;
            companyMap[msg.sender].cbalance += receiptMap[msg.sender][i].amount;
            receiptMap[msg.sender][i].rstatus = false;
        }
    }
    return (true, "Pay debt successfully!");
}
```

该函数即为应收账款的结算函数，二级供应商清算账单，银行清算账单，核心企业向二级供应商支付相应的欠款，二级供应商向银行支付相应的欠款，最后返回 true 和成功支付欠款的提示，此时账单的状态为无效。

4. 前端功能介绍

前端和链端的交互使用 web3 来完成。在编写好智能合约后，通过调用 web3 的相关接口来获得智能合约部署在链上的 abi 和合约地址，与智能合约进行绑定，进行相关函数的调用，将信息传递给网页，在友好高效的网页上实现与用户交互。

```
window.addEventListener("load", function() {

    if (typeof web3 !== 'undefined') {
        App.web3 = new Web3(web3.currentProvider);
    } else {
        App.web3 = new Web3(
            new Web3.providers.HttpProvider("http://127.0.0.1:8545"));
    }

    App.start();
});
```

切换用户，首先获取输入的用户 id，通过该用户 id 确定当前的公司地址，更新当前公司资产。

```
changeAccount: function() {
    const account_id = parseInt(document.getElementById('account-input').value);
    const addressElement = document.getElementById("account-address");
    this.account = accounts[account_id];
    addressElement.innerHTML = this.account;
    this.refreshName();
    this.refreshBalance();
},
```


添加公司，获取各个参数的值，通过 send 方法调用 addCompany 函数来进行添加。

```
addCompany: async function() {
  const cname = document.getElementById("add-cname").value;
  const caddress = document.getElementById("add-caddress").value;
  const ctype = document.getElementById("add-ctype").value;
  const cbalance = document.getElementById("add-cbalance").value;
  this.setStatus("Adding company...", "status0");
  const { addCompany } = this.meta.methods;
  await addCompany(cname, caddress, ctype, cbalance).send({ from: this.account, gas: 6721975 });
  this.setStatus("Add company successfully!", "status0");
},
```

签发应收账款，获取各个参数的值，通过 send 方法调用 signReceipt 函数来进行签订。

```
signReceipt: async function() {
  const caddress = document.getElementById("sign-caddress").value;
  const amount = parseInt(document.getElementById("sign-amount").value);
  const time = parseInt(document.getElementById("sign-time").value);
  this.setStatus("Signing receipt...", "status1");
  const { signReceipt } = this.meta.methods;
  await signReceipt(caddress, amount, time).send({ from: this.account, gas: 6721975 });
  this.setStatus("Sign receipt successfully!", "status1");
  this.refreshBalance();
},
```

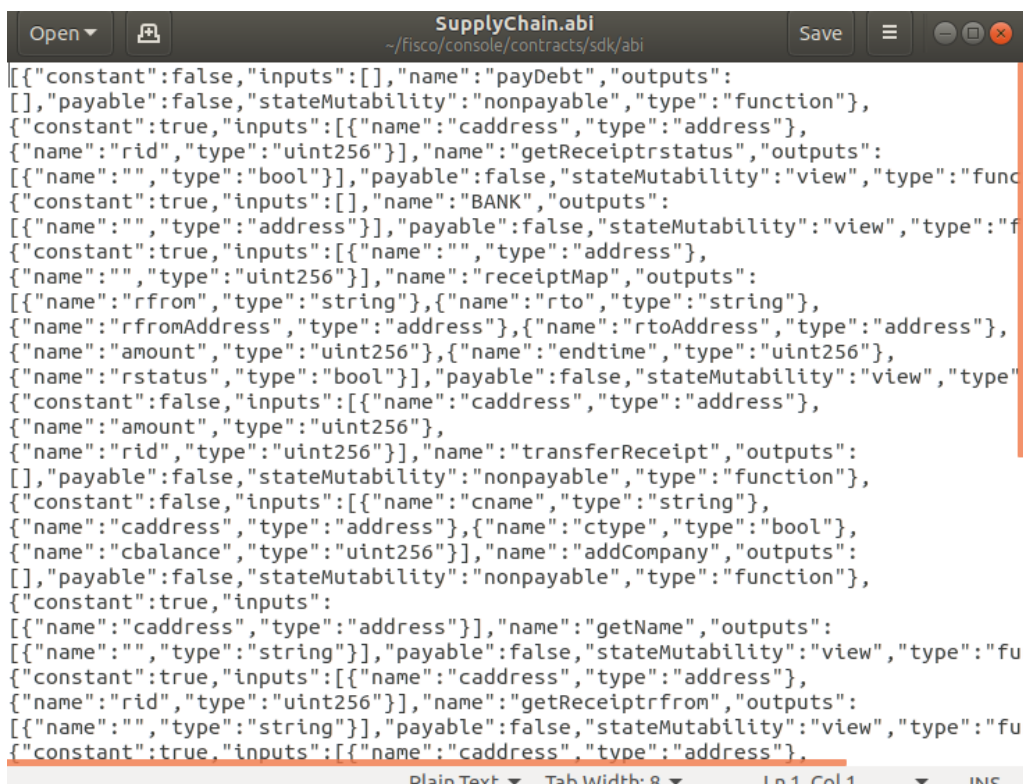
转让应收账款，获取各个参数的值，通过 send 方法调用 transferReceipt 函数进行转让。

```
transferReceipt: async function() {
  const caddress = document.getElementById("transfer-caddress").value;
  const amount = parseInt(document.getElementById("transfer-amount").value);
  const rid = parseInt(document.getElementById("transfer-rid").value);
  this.setStatus('Transferring receipt...', "status2");
  const { transferReceipt } = this.meta.methods;
  await transferReceipt(caddress, amount, rid).send({ from: this.account, gas: 6721975 });
  this.refreshBalance();
  this.setStatus('Transfer receipt successfully!', "status2");
},
```

融资，获取参数的值，通过 send 方法调用 financing 函数来进行融资。

```
financing: async function() {
  const rid = parseInt(document.getElementById("financing-rid").value);
  this.setStatus('Financing...', "status3");
  const { financing } = this.meta.methods;
  await financing(rid).send({ from: this.account, gas: 6721975 });
  this.refreshBalance();
  this.setStatus('Financing successfully!', "status3");
},
```

支付欠款，通过 send 方法调用 payDebt 函数来进行支付。



```
[{"constant":false,"inputs":[],"name":"payDebt","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":true,"inputs":[{"name":"caddress","type":"address"},
{"name":"rid","type":"uint256"}],"name":"getReceiptrstatus","outputs":
[{"name":"","type":"bool"}],"payable":false,"stateMutability":"view","type":"func
{"constant":true,"inputs":[],"name":"BANK","outputs":
[{"name":"","type":"address"}],"payable":false,"stateMutability":"view","type":"f
{"constant":true,"inputs":[{"name":"","type":"address"},
{"name":"","type":"uint256"}],"name":"receiptMap","outputs":
[{"name":"rfrom","type":"string"},{"name":"rto","type":"string"},
{"name":"rfromAddress","type":"address"},{"name":"rtoAddress","type":"address"},
{"name":"amount","type":"uint256"},{"name":"endtime","type":"uint256"},
{"name":"rstatus","type":"bool"}],"payable":false,"stateMutability":"view","type"
{"constant":false,"inputs":[{"name":"caddress","type":"address"},
{"name":"amount","type":"uint256"},
{"name":"rid","type":"uint256"}],"name":"transferReceipt","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":false,"inputs":[{"name":"cname","type":"string"},
{"name":"caddress","type":"address"},{"name":"ctype","type":"bool"},
{"name":"cbalance","type":"uint256"}],"name":"addCompany","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":true,"inputs":
[{"name":"caddress","type":"address"}],"name":"getName","outputs":
[{"name":"","type":"string"}],"payable":false,"stateMutability":"view","type":"fu
{"constant":true,"inputs":[{"name":"caddress","type":"address"},
{"name":"rid","type":"uint256"}],"name":"getReceiptrfrom","outputs":
[{"name":"","type":"string"}],"payable":false,"stateMutability":"view","type":"fu
{"constant":true,"inputs":[{"name":"caddress","type":"address"}].
```

启动客户端，在终端输入 `npm install` 下载需要的依赖，后输入 `npm run dev` 启动，通过浏览器访问 <http://localhost:8080> 即可看到用户界面。

三、 功能测试

1. 添加公司

选择菜单栏中 Add company 进行添加，该交易发起者为用户 0 即银行，填入公司名、公司地址、公司类型、公司资产，点击添加按钮。

Supplychain Finance

[Add company](#) [Sign receipt](#) [Transfer receipt](#) [Financing](#) [Pay debt](#)

Account ID:

Name: BANK

Address: 0xdBD30b47CDEB6F785FB80C39cb7f3e2cd7De1228

Balance: 10000

Change account

Add company

cname:

caddress:

ctype:

cbalance:

Add company

添加宝马公司：

Supplychain Finance

[Add company](#) [Sign receipt](#) [Transfer receipt](#) [Financing](#) [Pay debt](#)

Account ID:

Name: BANK

Address: 0xdBD30b47CDEB6F785FB80C39cb7f3e2cd7De1228

Balance: 10000

Change account

Add company

cname:

caddress:

ctype:

cbalance:

Add company successfully!

Add company

添加轮胎公司：

Supplychain Finance

[Add company](#) [Sign receipt](#) [Transfer receipt](#) [Financing](#) [Pay debt](#)

Account ID:

Name: BANK

Address: 0xdBD30b47CDEB6F785FB80C39cb7f3e2cd7De1228

Balance: 10000

Change account

Add company

cname:

caddress:

ctype:

cbalance:

Add company successfully!

Add company

添加轮毂公司：

Supplychain Finance

[Add company](#) [Sign receipt](#) [Transfer receipt](#) [Financing](#) [Pay debt](#)

Account ID:

Name: BANK

Address: 0xdBD30b47CDEB6F785FB80C39cb7f3e2cd7De1228

Balance: 10000

Change account

Add company

cname:

caddress:

ctype:

cbalance:

Add company successfully!

Add company

添加成功后出现 Add company successfully! 的提示信息。此时查看宝马公司、轮胎公司、轮毂公司的具体信息如下：

Account ID: 1

Name: BAOMA

Address: 0xcbcbeB8dADF2a2151636caAd4A58668D24531018

Balance: 1000

Account ID: 2

Name: LUNTAI

Address: 0x16F4FB5Ca2aF0D1c6DF0B5b857Dc05F40F3C777e

Balance: 0

Account ID: 3

Name: LUNGU

Address: 0xe59b2df47906E954bC2E2403707F050dFf0D0c76

Balance: 0

2. 签发应收账款

选择菜单栏中 Sign receipt 进行签订，更改用户 id，宝马公司为交易发起者，填入卖家公司地址、交易数额、交易时间，点击签订按钮。

The screenshot displays the 'Supplychain Finance' application interface. At the top, a navigation bar includes links for 'Add company', 'Sign receipt', 'Transfer receipt', 'Financing', and 'Pay debt'. The 'Sign receipt' option is currently selected. Below the navigation bar, the account details for 'BAOMA' (Account ID: 1) are shown, including its address and a balance of 1000. A 'Change account' button is positioned below these details. The main section is titled 'Sign receipt' and contains three input fields labeled 'address:', 'amount:', and 'time:'. A 'Sign receipt' button is located at the bottom of this section.

Supplychain Finance

Add company Sign receipt Transfer receipt Financing Pay debt

Account ID: 1

Name: BAOMA

Address: 0xcbcbeB8dADF2a2151636caAd4A58668D24531018

Balance: 1000

Change account

Sign receipt

caddress: 0x16F4FB5Ca2aF0D1c6DF0B5b857Dc05F40F3C777e

amount: 1000

time: 50

Sign receipt successfully!

Sign receipt

签订成功后出现 Sign receipt successfully! 的提示信息。

3. 转让应收账款

选择菜单栏中 Transfer receipt 进行签订，更改用户 id，轮胎公司为交易发起者，填入账单转入公司地址、交易数额、账单 id，点击转让按钮。

Supplychain Finance

Add company Sign receipt Transfer receipt Financing Pay debt

Account ID: 2

Name: LUNTAI

Address: 0x16F4FB5Ca2aF0D1c6DF0B5b857Dc05F40F3C777e

Balance: 0

Change account

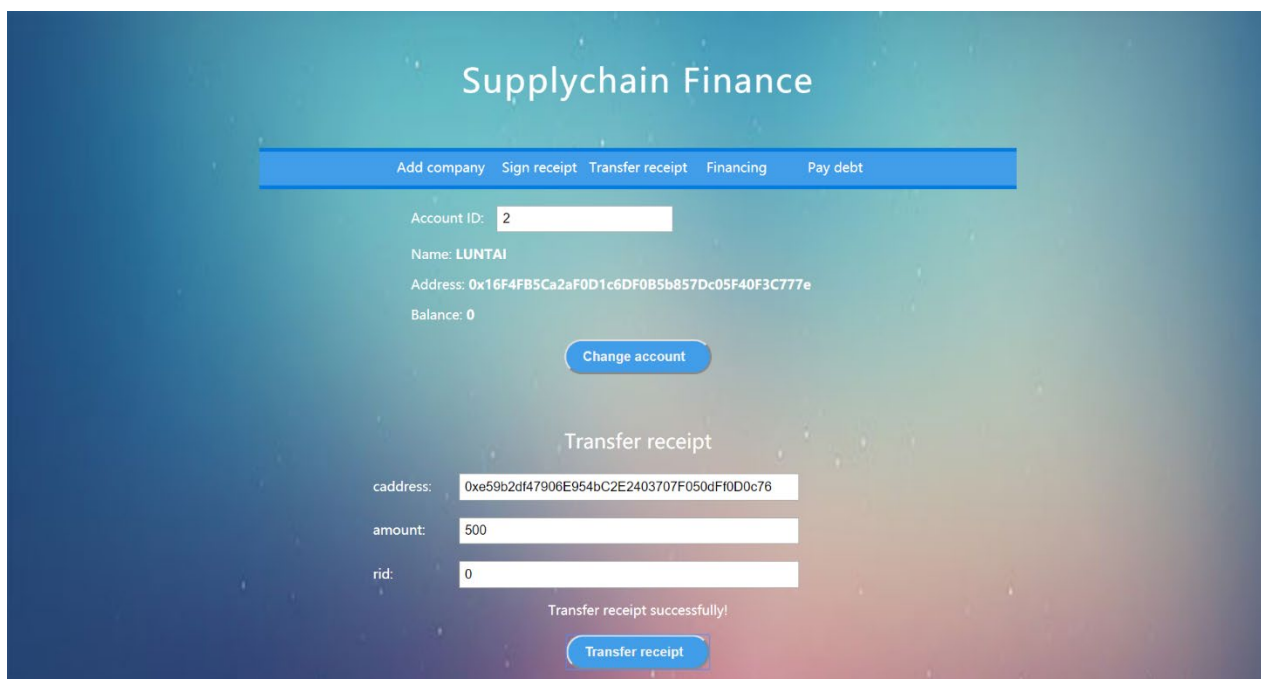
Transfer receipt

caddress:

amount:

rid:

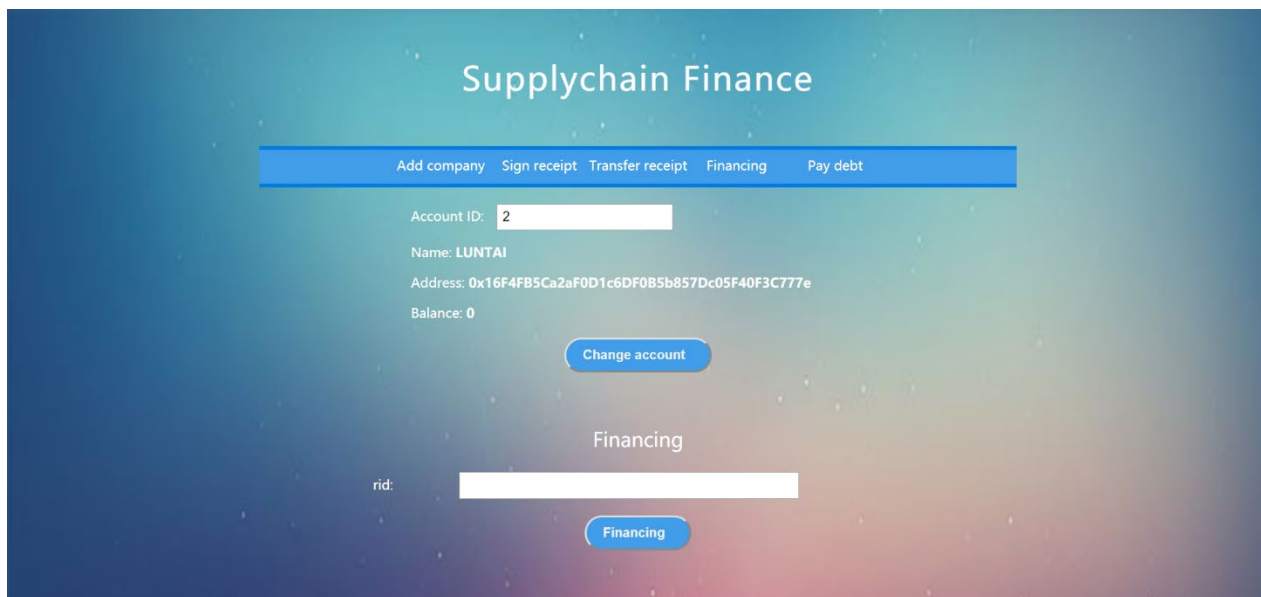
Transfer receipt

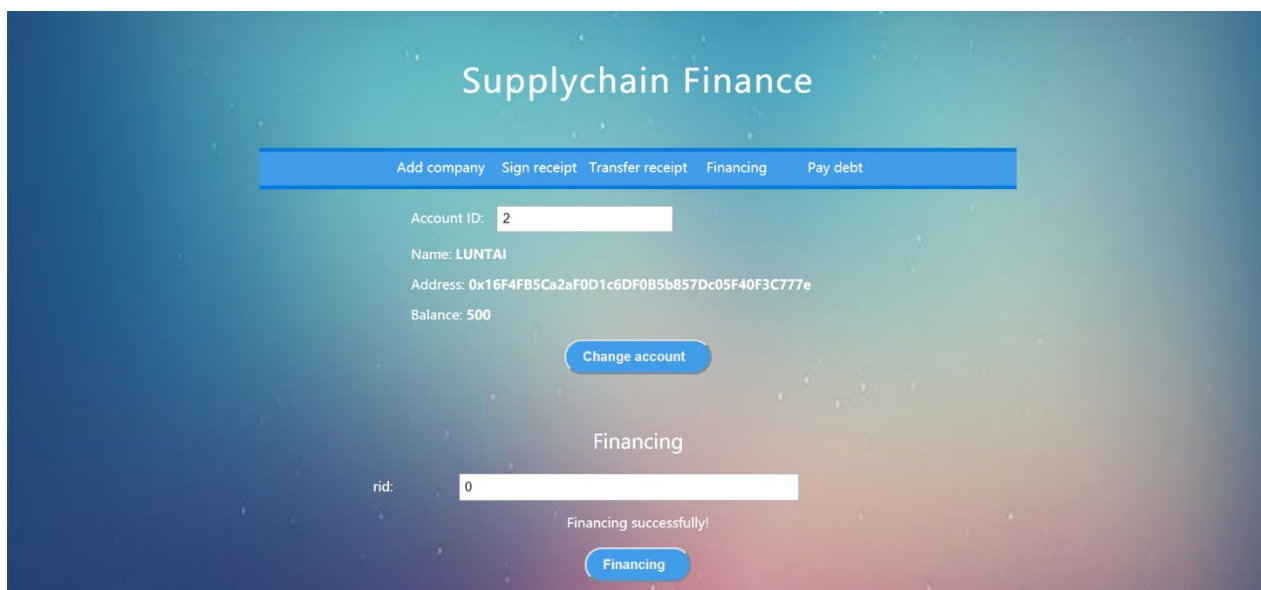


转让成功后出现 Transfer receipt successfully! 的提示信息。

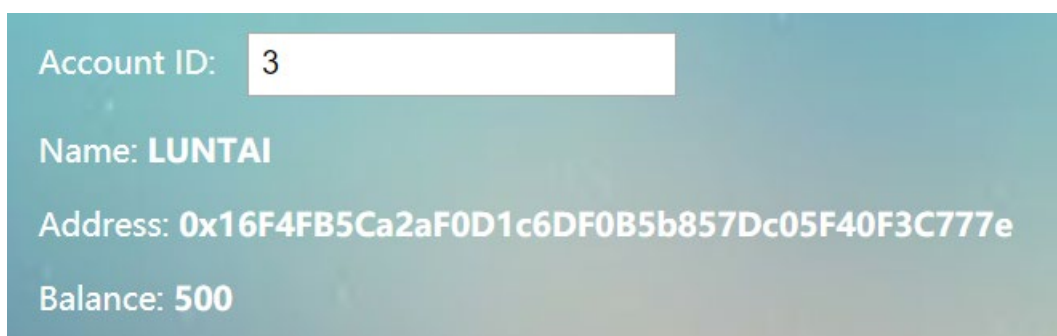
4. 申请融资

选择菜单栏中 Financing 进行融资，更改用户 id，轮毂公司向银行发起融资，填入具体账单 id，点击融资按钮。



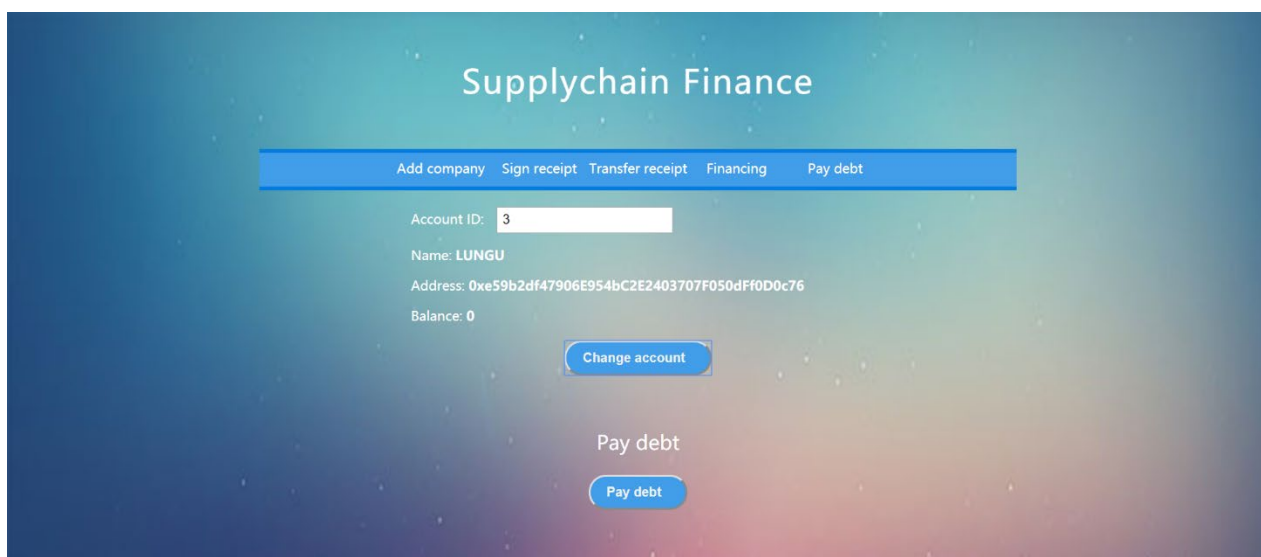


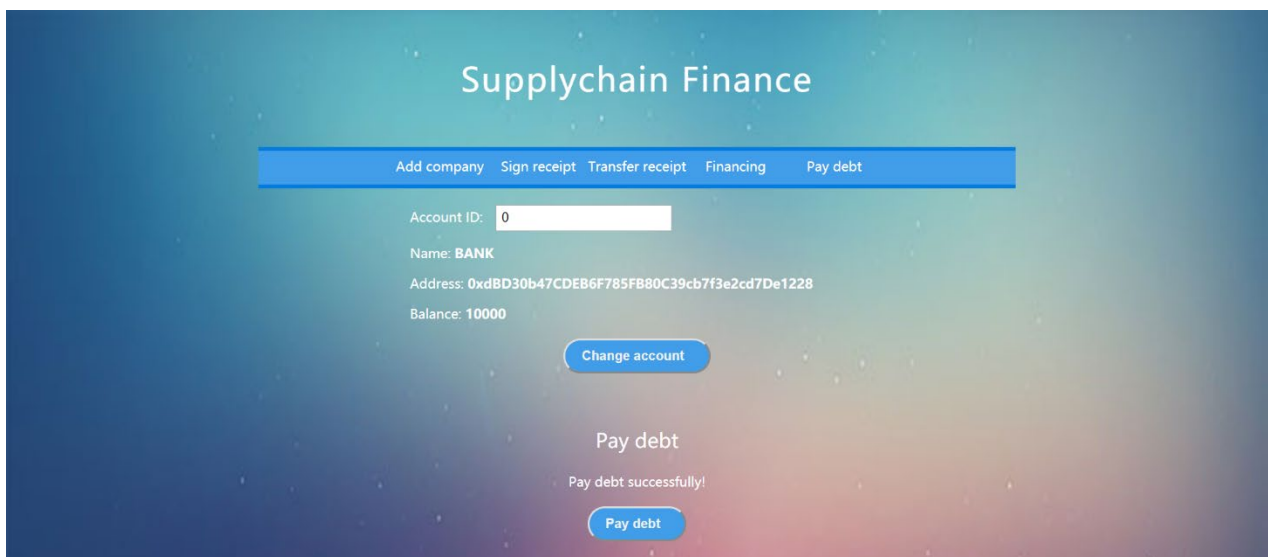
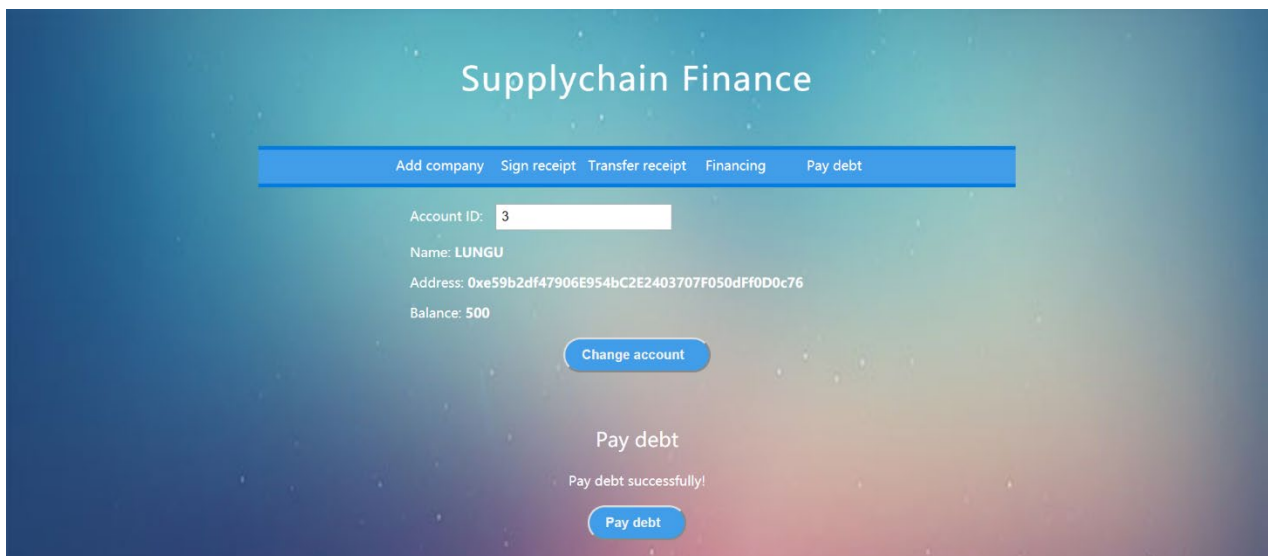
融资成功后出现 Financing successfully! 的提示信息。此时查看轮毂公司的具体信息，其中公司资产增加为 500。



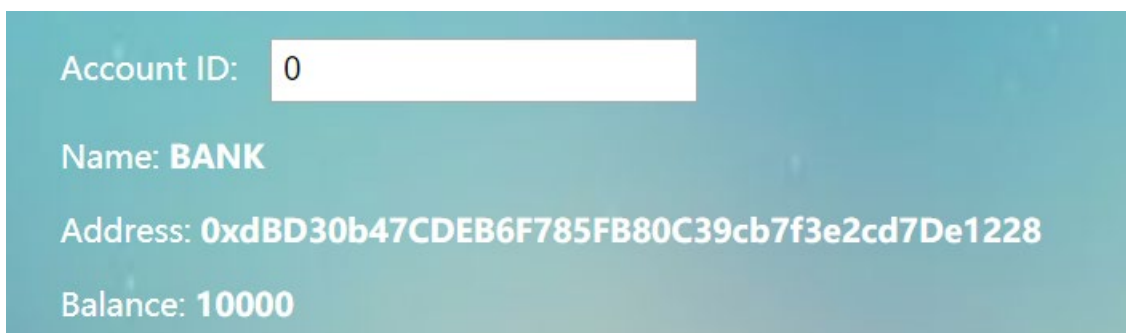
5. 支付账单

选择菜单栏中 Pay debt 进行签订，更改用户 id，轮毂公司和银行清算账单，点击支付按钮。





支付成功后出现 Pay debt successfully! 的提示信息。此时查看银行、宝马公司、轮胎公司和轮毂公司的具体信息，其中银行资产保持不变为 10000，宝马公司资产为 0，轮胎公司和轮毂公司的资产增加为 500。



Account ID:

1

Name: **BAOMA**

Address: **0xcbcbeB8dADF2a2151636caAd4A58668D24531018**

Balance: **0**

Account ID:

2

Name: **LUNTAI**

Address: **0x16F4FB5Ca2aF0D1c6DF0B5b857Dc05F40F3C777e**

Balance: **500**

Account ID:

3

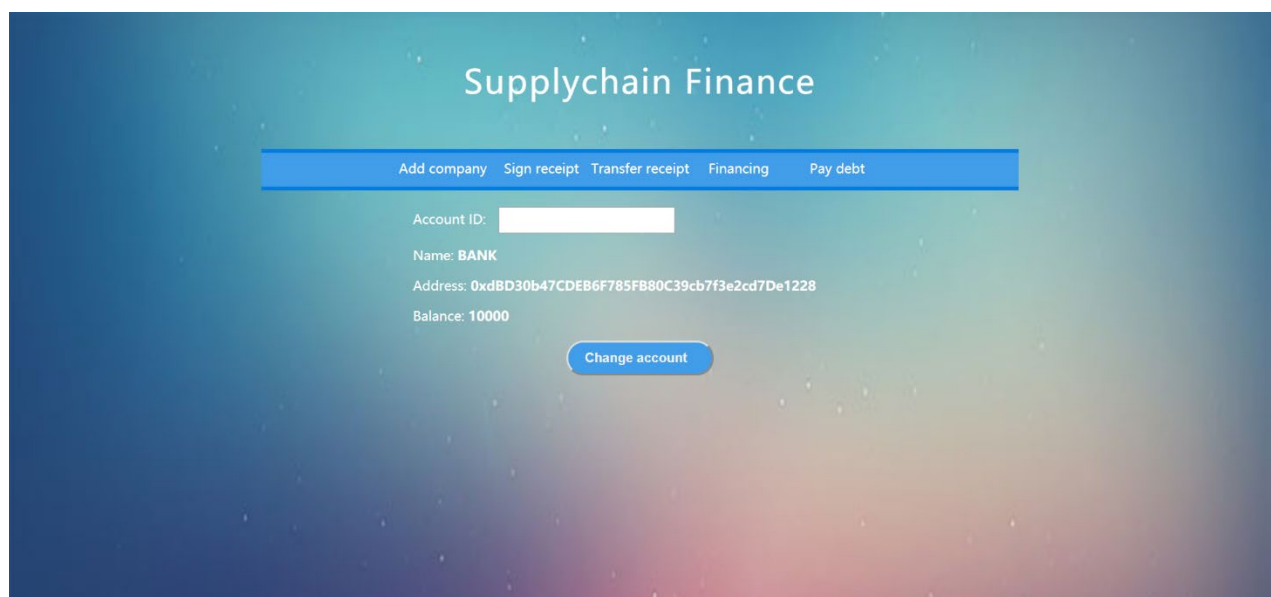
Name: **LUNTAI**

Address: **0x16F4FB5Ca2aF0D1c6DF0B5b857Dc05F40F3C777e**

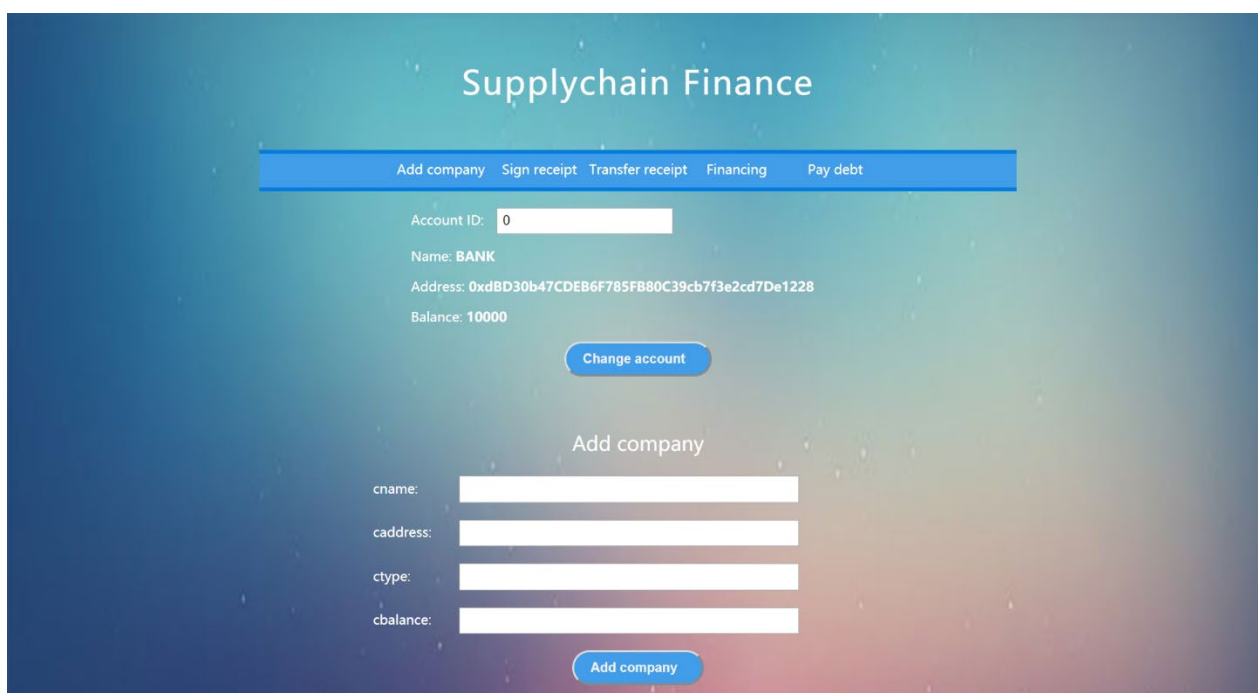
Balance: **500**

四、 界面展示

1. 初始界面:



2. 添加公司界面:



Supplychain Finance

Add company Sign receipt Transfer receipt Financing Pay debt

Account ID: 0

Name: BANK

Address: 0xdBD30b47CDEB6F785FB80C39cb7f3e2cd7De1228

Balance: 10000

Change account

Add company

cname:

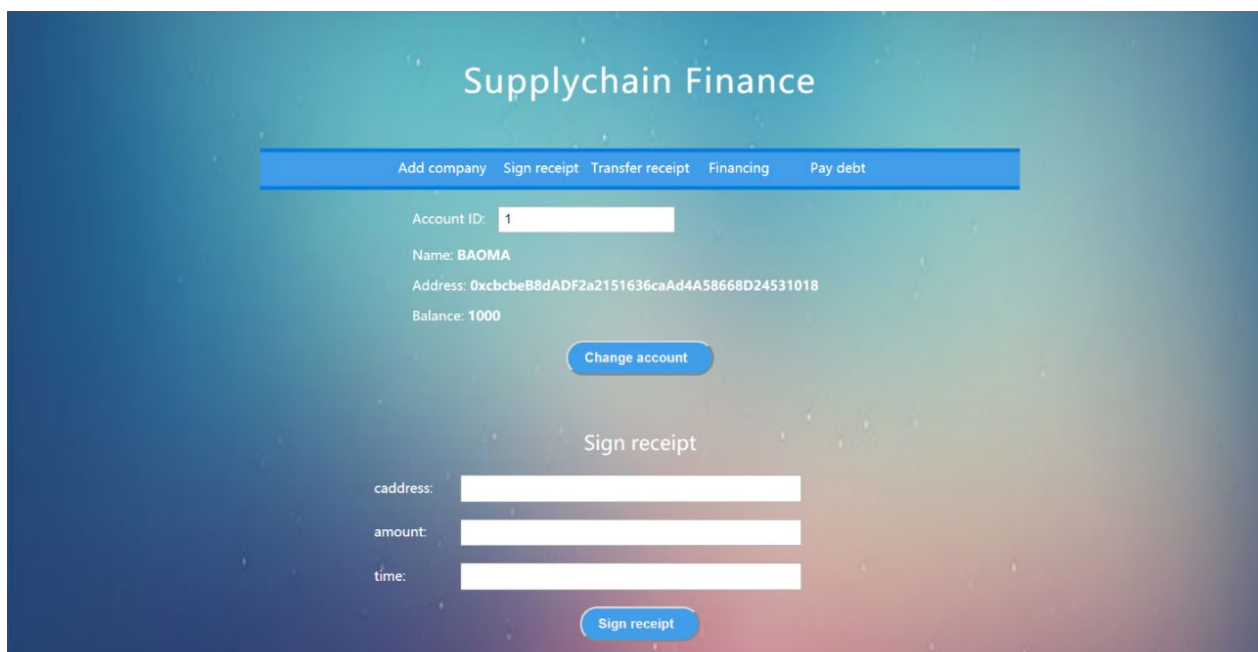
caddress:

ctype:

cbalance:

Add company

3. 签发应收账款界面:



Supplychain Finance

Add company Sign receipt Transfer receipt Financing Pay debt

Account ID: 1

Name: BAOMA

Address: 0xcbcbeB8dADF2a2151636caAd4A58668D24531018

Balance: 1000

Change account

Sign receipt

caddress:

amount:

time:

Sign receipt

4. 转让应收账款界面:

Supplychain Finance

Add company

Sign receipt

Transfer receipt

Financing

Pay debt

Account ID:

2

Name:

LUNTAI

Address:

0x16F4FB5Ca2aF0D1c6DF0B5b857Dc05F40F3C777e

Balance:

0

Change account

Transfer receipt

caddress:

amount:

rid:

Transfer receipt

5. 申请融资界面:

Supplychain Finance

Add company

Sign receipt

Transfer receipt

Financing

Pay debt

Account ID:

2

Name:

LUNTAI

Address:

0x16F4FB5Ca2aF0D1c6DF0B5b857Dc05F40F3C777e

Balance:

0

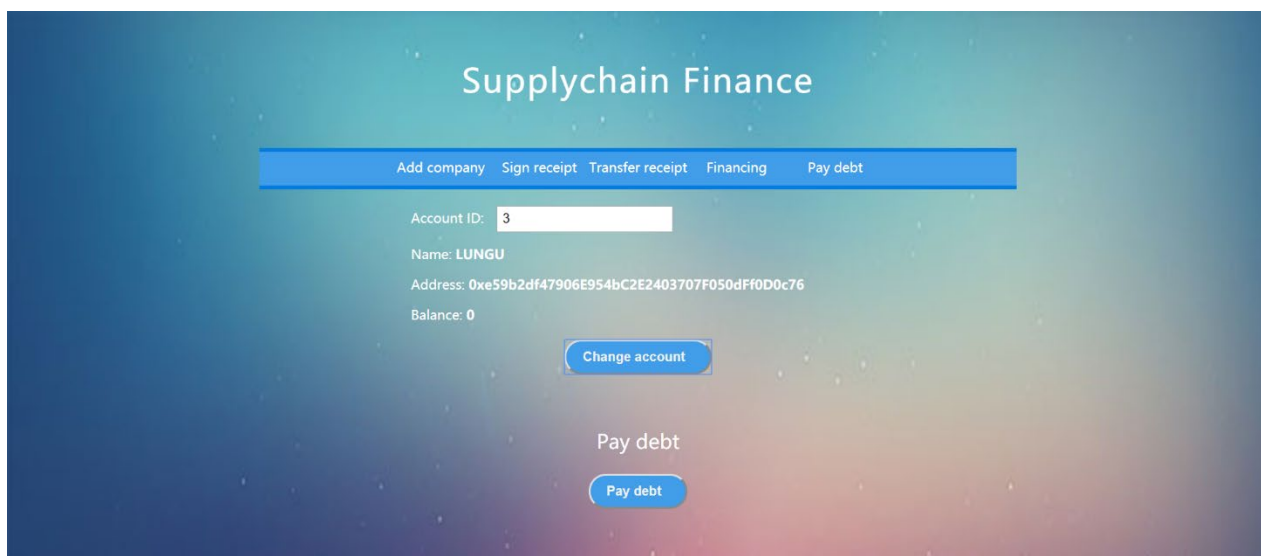
Change account

Financing

rid:

Financing

6. 支付账单界面:



五、 心得体会

通过本次实验，我学习到了许多关于区块链的新知识，包括如何使用 solidity 语言编写智能合约，编写完智能合约后如何将智能合约部署到链上，对链端和前端后端的数据交互方式也有了一定的了解，能够通过链端和前端后端相结合来做出一个基本完整的区块链应用，对区块链的去中心化结构也有了更加深入的了解。虽然这个应用还有许多的缺陷和不足，许多功能仍然需要优化和完善，但这也激励我在接下来区块链的学习中更加认真，弥补之前的不足。

在这次实验过程中我也遇到了许多困难，比如一开始对 solidity 语言不熟悉，花了许多时间去学习它的语法，但在编写和部署过程中还是出现了许多问题，通过向其他同学询问后才得以解决，微众银行的老师们也在群里积极地为我们解答问题。在大作业第三阶段编写前端代码过程中，我对如何将前端与链端结合起来感到十分困难，后来通过使用 web3 来完成。在编写好智能合约后，通过调用 web3 的相关接口来获得智能合约部署在链上的 abi 和合约地址，与智能合约进行绑定，进行相关函数的调用，将信息传递给网页，就能在友好高效的网页上实现与用户交互。

区块链在当今社会中的热度越来越高，通过这次实验，我更加深入的学习了区块链，知道了其为何能这么备受关注。完成这次实验虽然花费了不少时间，但我也从中收获了许多，希望在今后对区块链的学习中能够更进一步。