中山大学数据科学与计算机学院本科生实验报告

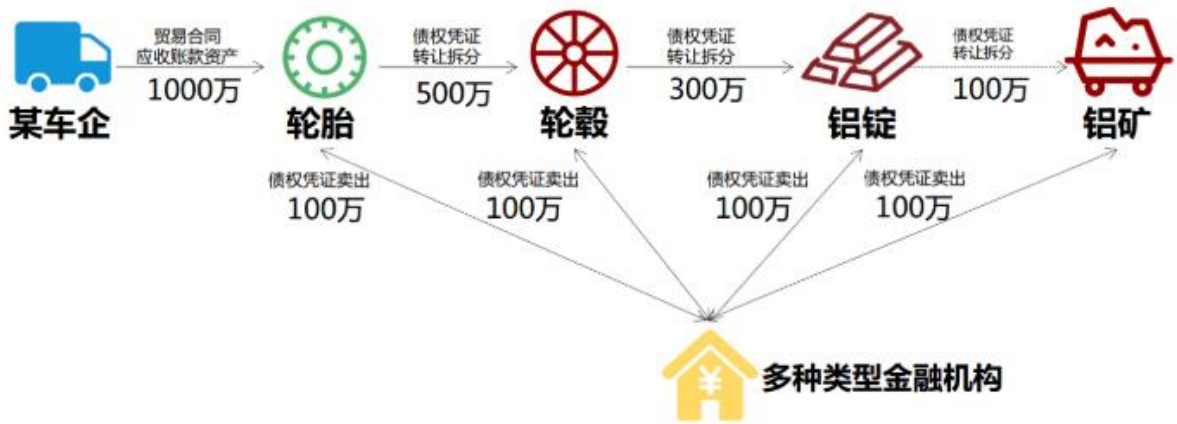
(2019 年秋季学期)

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	17343071	姓名	梁宇凌
电话	13242871016	Email	Liangyling3@163.com
开始日期	2019/11/7	完成日期	2019/12/13

项目地址：https://github.com/liangyling3/SupplyChain

一、 项目背景



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。

二、 方案设计

1. 存储设计

(1) 声明一个收据的数据结构。

```
//收据信息
struct Receipt {
    uint receiptID;          //单据ID
    address from;            //发起者地址
    address to;              //收款人地址
    uint mount;              //金额
    address prover;          //见证机构名称
    bool financinged;        //是否已融资
    bool settled;            //是否已结算（由核心公司）
    uint[] transferRecord;   //被转让记录，里面是用此receipt作为凭证的转账的
    receiptID
    uint leftValue;          //剩余金额，将收据用于融资、还款后的剩余金额
}
```

(2) 声明一个公司的数据结构，其中使用 mapping 表示公司与收据间的映射关系。

```
//公司信息
struct Company{
    address name;    //公司名称
    address addr;    //公司地址
    uint type;       //公司类型 0为核心企业 1为银行 2为中小企业
}
```

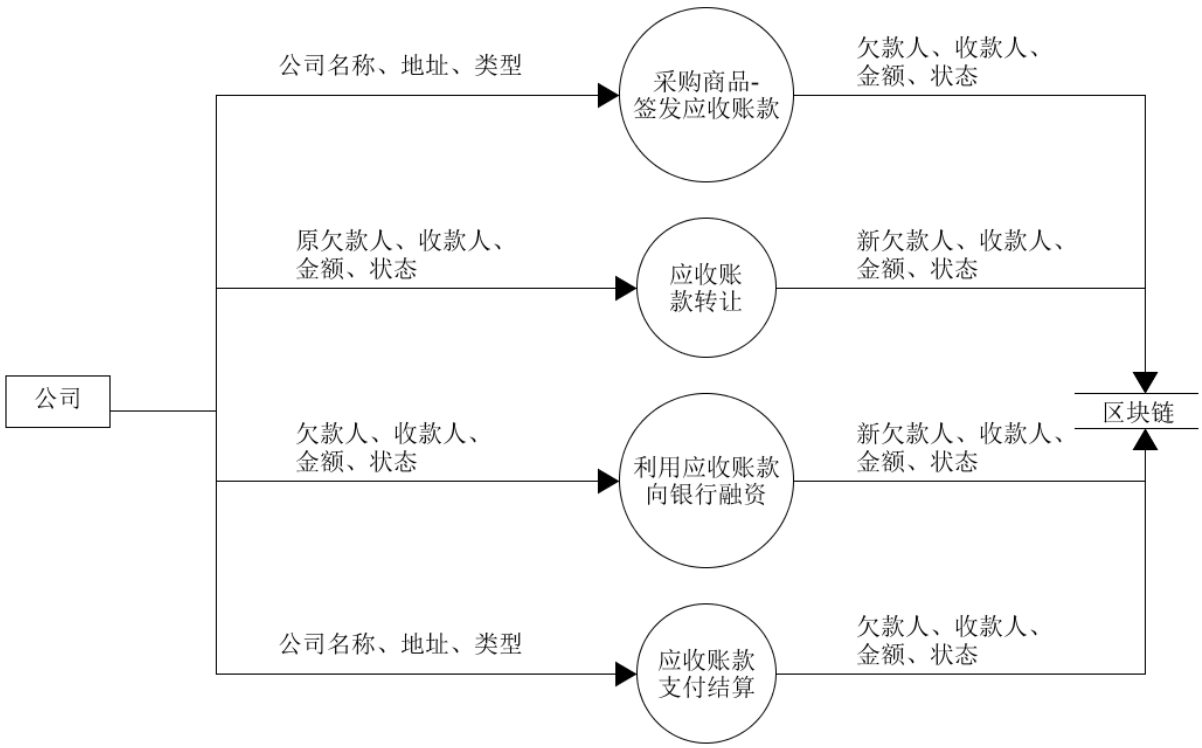
(3) 声明一个 mapping 存储现有公司。

```
mapping(address => Company) public companies;
```

(4) 数组存储链上的公司、收据。

```
Company[] public companiesInChain;
Receipt[] public receipts;
```

2. 数据流图



3. 核心功能介绍（文字+代码）

功能一 ：实现采购商品—签发应收账款交易上链。就是把谁因为什么东西欠了谁多少钱记录到链上，这个交易过程需要被见证（通常为银行）。

```
//功能一：实现采购商品—签发应收账款交易上链。
function generateReceipt(address fromer, address receiver, address prover,
uint amount) {
    //核心机构入链
    companiesInChain.push(Company(fromer, 0));
    companies[fromer] = companiesInChain[0];
    //见证方入链
    companiesInChain.push(Company(prover, 1));
    companies[prover] = companiesInChain[1];
    //收款方入链
    companiesInChain.push(Company(receiver, 2));
    companies[receiver] = companiesInChain[2];

    //单据入链
    receipts.length ++;
    uint index = receipts.length -1;
    receipts[index].receiptID = receipts.length;
    receipts[index].from = fromer;
    receipts[index].to = receiver;
    receipts[index].mount = amount;
    receipts[index].financinged = false;
    receipts[index].settled = false;
    receipts[index].prover = prover;
    receipts[index].leftValue = amount;

    //打印收据
    emit NewReceipt("new", receipts[index].receiptID, receipts[index].from,
    receipts[index].to, receipts[index].mount, receipts[index].prover,
    receipts[index].financinged, receipts[index].settled, receipts[index]
    .transferRecord, receipts[index].leftValue);

}
```

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。就是把车企欠轮胎 1000，轮胎欠轮毂 500，那么可以等价于车企欠轮胎和轮毂各自 500，要把这些应收账款的转让交易上链。

```
//功能二：实现应收账款的转让上链
function Transfer(address fromer, address receiver, uint amount, uint
receiptID) {
    //检查收据没有被用来融资
    require(
        !receipts[receiptID].financinged,
        "This receipt has been used to financing."
    );
    //检查收据金额足够
    require(
        receipts[receiptID].leftValue >= amount,
        "This receipt's leftValue is not enough now."
    );

    //收款方入链
    if (!InChain(receiver)) {
        companiesInChain.push(Company(receiver, 2));
        companies[receiver] = companiesInChain[companiesInChain.length - 1];
    }

    //新收据入链
    receipts.length ++;
    uint index = receipts.length-1;
    receipts[index].receiptID = receipts.length;
    receipts[index].from = receipts[receiptID].to;
    receipts[index].to = receiver;
    receipts[index].mount = amount;
    receipts[index].financinged = false;
    receipts[index].settled = false;
    receipts[index].leftValue = amount;
    //打印新收据
    emit NewReceipt("new", receipts[index].receiptID, receipts[index].from,
    receipts[index].to, receipts[index].mount, receipts[index].prover,
    receipts[index].financinged, receipts[index].settled, receipts[index]
    .transferRecord, receipts[index].leftValue);
```

```
    //更新旧收据
    receipts[receiptID].transferRecord.push(receipts.length - 1);
    receipts[receiptID].leftValue -= amount;
    //打印旧收据
    emit NewReceipt("fresh", receiptID, receipts[receiptID].from, receipts
    [receiptID].to, receipts[receiptID].mount, receipts[receiptID].prover,
    receipts[receiptID].financinged, receipts[receiptID].settled, receipts
    [receiptID].transferRecord, receipts[receiptID].leftValue);

}
```

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。根据应收账款的额度向银行借钱，比如车企欠你 500，你可以用这个凭证向银行借 500，也需要把这种交易上链。

```
//功能三：  利用应收账款向银行融资上链
function Financing(uint receiptID) {
    //检查收据没有被用来融资
    require(
        !receipts[receiptID].financinged,
        "This receipt has been used to financing."
    );
    // 更新收据
    receipts[receiptID].financinged = true;
    // 打印收据
    emit NewReceipt("fresh", receiptID, receipts[receiptID].from, receipts
    [receiptID].to, receipts[receiptID].mount, receipts[receiptID].prover,
    receipts[receiptID].financinged, receipts[receiptID].settled, receipts
    [receiptID].transferRecord, receipts[receiptID].leftValue);

}
```

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。到期还钱，销毁在链上对应的应收账款，表明这笔账结了。

```
//功能四：应收账款支付结算上链
function Settle(uint receiptID) {
    // 更新收据
    receipts[receiptID].settled = true;
    // 打印收据
    emit NewReceipt("fresh", receiptID, receipts[receiptID].from, receipts
    [receiptID].to, receipts[receiptID].mount, receipts[receiptID].prover,
    receipts[receiptID].financinged, receipts[receiptID].settled, receipts
    [receiptID].transferRecord, receipts[receiptID].leftValue);

}
```

路由控制供应链部分代码（./iview-admin/src/router/routers.js）

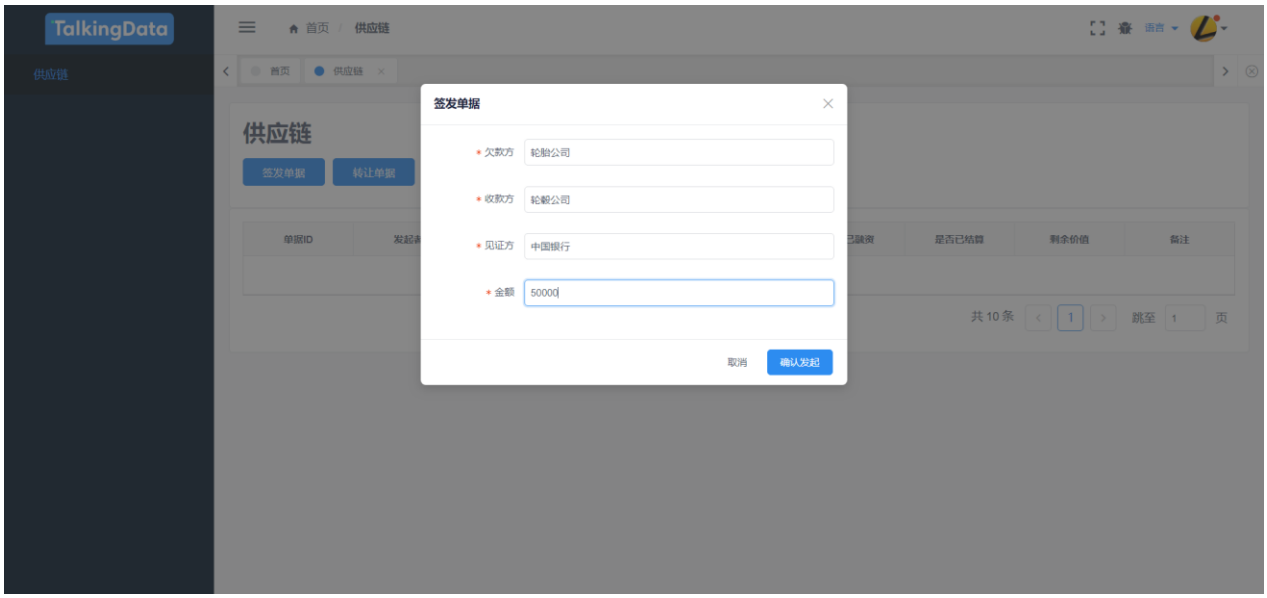
```
{
  path: '/SupplyChain',
  name: 'SupplyChain',
  component: Main,
  meta: {
    hideInBread: true
  },
  children: [
    {
      path: '/SupplyChain',
      name: '供应链',
      meta: {
        title: '供应链'
      },
      component: () => import('@/view/SupplyChain.vue')
    }
  ]
},
```

前端视图部分代码（./iview-admin/src/view/SupplyChain.vue）

```
<Modal v-model="isGenerateReceipt" title="签发单据">
    <!-- 3.0.1 GenerateReceiptForm 是一个数组 3.0.2
    GenerateReceiptRules -->
    <Form :model="GenerateReceiptForm" ref="GenerateReceiptForm"
    :label-width="110" :rules="GenerateReceiptRules">
        <FormItem label="欠款方" prop="fromAccount" >
            <Input  clearable
            v-model="GenerateReceiptForm.fromAccount" placeholder = "
            请输入新合同收款方"/>
        </FormItem>
        <FormItem label="收款方" prop="toAccount" >
            <Input  clearable v-model="GenerateReceiptForm.toAccount"
            placeholder = "请输入新合同收款方"/>
        </FormItem>
        <FormItem label="见证方" prop="prover" >
            <Input  clearable v-model="GenerateReceiptForm.prover"
            placeholder = "请输入见证该合同方"/>
        </FormItem>
        <FormItem label="金额" prop="amount" >
            <Input  clearable v-model="GenerateReceiptForm.amount"
            placeholder = "请输入转让金额"/>
        </FormItem>
    </Form>
    <div slot="footer">
      <!-- 3.0.3 点击取消 到cancelGenerateReceipt函数 -->
      <Button type="text" @click="cancelGenerateReceipt">取消</Button>
    <!-- 4.0 提交发起-->
    <Button type="primary" @click="doGenerateReceipt">确认发起</
    Button>
</div>
</Modal>
```

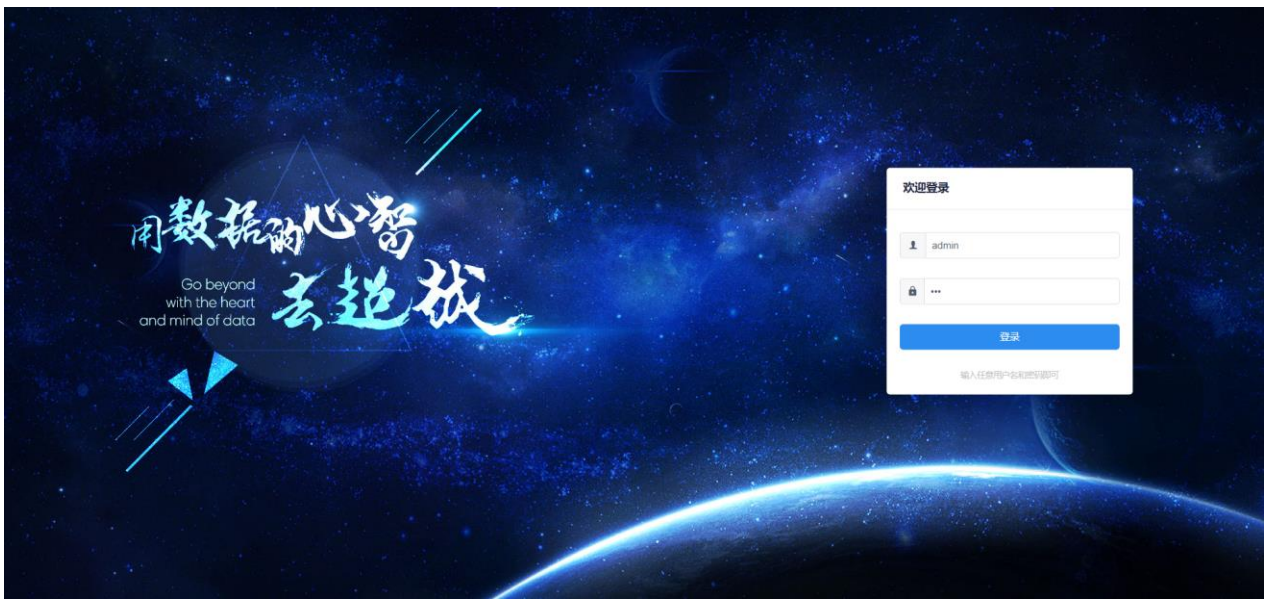
三、 功能测试

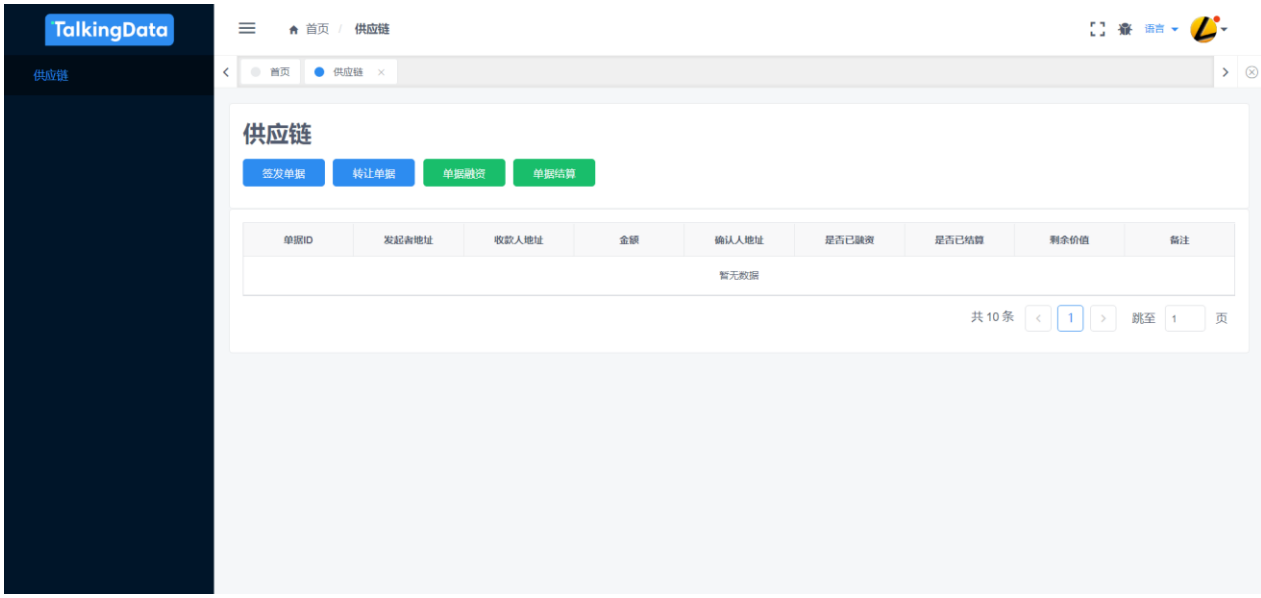
1. 功能一：签发单据



2. 功能二：转让单据
3. 功能三：单据融资
4. 功能四：单据结算

四、 界面展示





五、 心得体会

这次大作业耗费的精力较大。从最终效果来说其实不算特别复杂，但是一开始完全没有思路。在群里 TA 提出思路之后才能够写出大概的智能合约。除此以外，前端以及如何将前端、后端、链端连接起来也是难点之一。了解到 iview-admin 框架可以用于本作业项目，这样一来前端的实现就比较顺利了。将智能合约放入服务端，即可将前后端连接起来，并使用合约功能。

总体来说，这次大作业不仅囊括了区块链的相关知识，还涉及编程的整个方方面面。对我来说，不仅对于区块链有了进一步的理解，也加强了我智能合约的编程能力。除此以外，我还了解并使用了前端 vue 框架，并能够搭建出一个初步的完整项目。因此，收获颇丰。

作业要求：

1. 命名要求：学号_姓名，例如 16340000_林 XX。
2. 实验报告提交格式为 pdf。
3. 实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。