

中山大学数据科学与计算机学院 本科生实验报告

课程名称： 区块链原理与技术

任课教师： 郑子彬

年级	大四	专业（方向）	计应
学号	16340249	姓名	向泳丽
电话	15989181701	Email	741563052@qq.com
开始日期	2019/10	完成日期	2020/1

中山大学数据科学与计算机学院本科生实验报告

一、项目背景

二、方案设计

2.1 程序概览图

2.2 类示意图

2.3 设计流程

2.4 设计思路

2.4.1 链端

2.4.2 后端

2.4.3 前端

2.5 核心代码

三、功能测试

3.1 签发账单

3.1.1 创建银行并认证

3.1.2 账单签发

3.2 转让账单

3.2.1 查询现有账单

3.2.2 转让账单并确认

3.3 融资

3.4 支付结算

四、界面展示

4.1 登录界面

4.2 主界面

4.3 存款

4.4 存款结果

4.5 取款

- 4.6 取款结果
- 4.7 创建账单
- 4.8 创建账单后
- 4.9 转移账单
- 4.10 转移结果
- 4.11 融资
- 4.12 融资结果
- 4.13 支付欠款
- 4.14 支付后
- 4.15 信用审核

五、体会心得

一、项目背景

本次项目实现了以下功能：

汽车制造商可以将账单发送给银行或轮胎制造商，以记录未付的欠款。

通过汽车公司的现有存款银行确定是否信任账单。如果汽车公司的保证金超过10,000元，则可以在公司之间传递账单，并且银行可以由下游公司提供资金。

如果汽车公司的存款少于10,000元人民币，银行将不信任它，也将不允许转让账单

如果汽车制造商开具轮胎制造商账单并且银行认可了该账单，则轮胎制造商可以将账单用于融资或者向下游公司付款。比如轮胎制造商可以使用该账单购买轮毂，并将对应额度的账单转移给轮毂制造商。轮毂制造商也可以转移账单或向银行融资。

当资金允许时，汽车制造商可以使用其存款支付账单，从而消除欠款。

所有账单的发行，转移和确认都必须以event的形式链接在一起，以确保资源和信息的公开和无法更改，并且每个更改请求和交易都将在链的末尾详细记录，以确保在交易圈内的信誉。

二、方案设计

2.1 程序概览图

Car

Wheel hub Company

Wheel hub Company

Balance:10000

Deposit

Withdraw

Create bills

Transfer bills

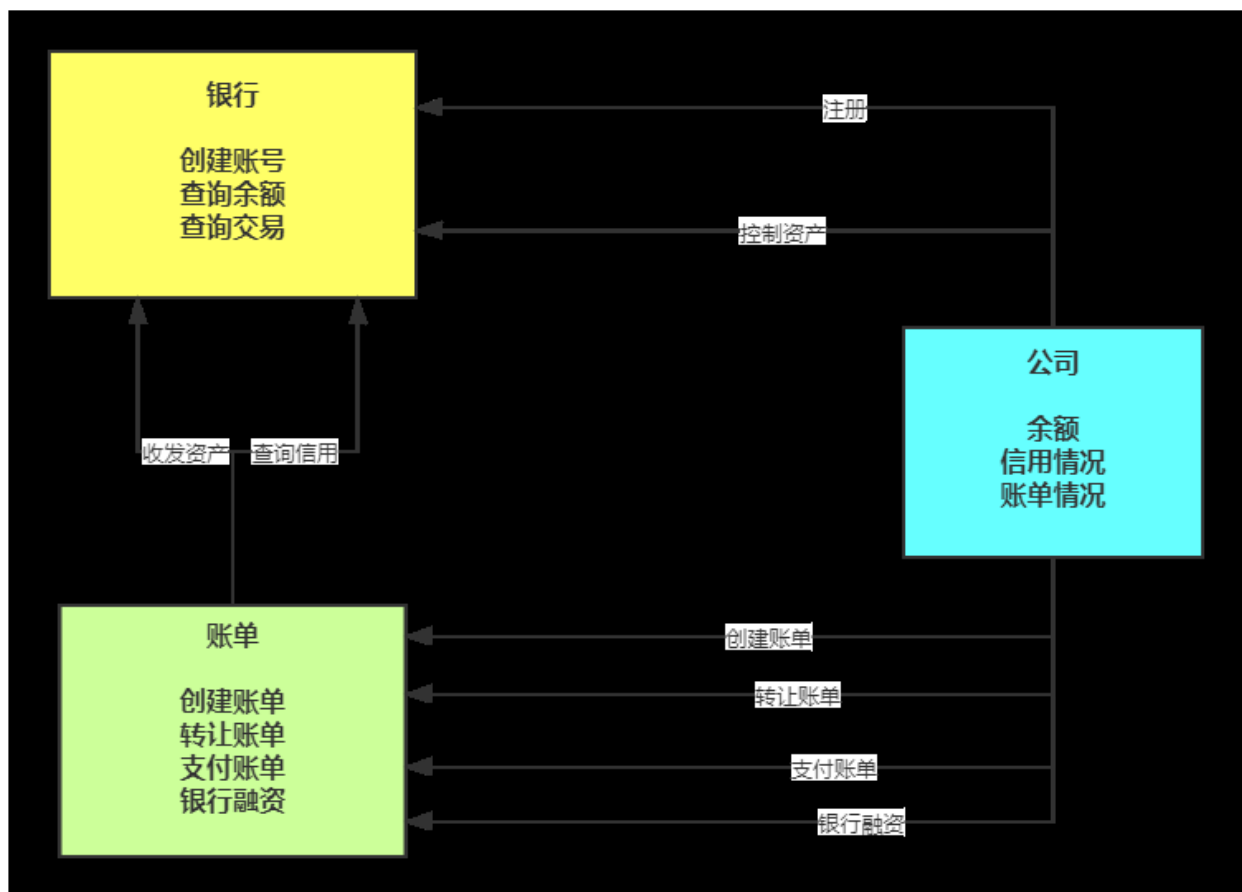
Finacing

Payment

























Creditor Bill

Arrears Bill

2.2 类示意图



2.3 设计流程

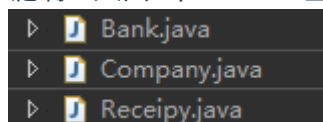
Commits on Jan 16, 2020		
final code pygmelion committed 15 minutes ago	 f2cab0b	
upload remaining code pygmelion committed 35 minutes ago	 c91a1de	
Commits on Jan 15, 2020		
checker pygmelion committed 16 hours ago	 d1f4daf	
update mainpage && add deposit && add withdraw pygmelion committed 16 hours ago	 84c0965	
fix a bug && upload Mainpage pygmelion committed 18 hours ago	 b34f1ac	
rebuild highlight pygmelion committed 18 hours ago	 21fc6bc	
update account && upload highlight pygmelion committed 18 hours ago	 1de156c	
add account system pygmelion committed 18 hours ago	 efaf6f8	
add backend framework pygmelion committed 18 hours ago	 bc1f9ef	
upload all .sol pygmelion committed 18 hours ago	 abf262f	
upload Recipy.sol w21180239 committed 18 hours ago	 cb7bc47	
Initial commit pygmelion committed 18 hours ago	Verified  1b2f91b	

PS:第一次commit使用了服务器电脑未切换git账号

2.4 设计思路

2.4.1 链端

链端一共由3个contract组成，如下：



在一台Ubuntu机器下（这次用的是虚拟机）部署完合约之后，便可以在程序中使用SDK提供的接口与对应的IP地址（192.168.100.2:20200）进行交互了。

2.4.2 后端

在main.java中调用fisco的SDK链接到链端并创建web3j对象

```

public static void main(String args[]) throws Exception {

    ApplicationContext context = new ClassPathXmlApplicationContext("classpath:app
    Service service = context.getBean(Service.class);
    service.run();

    ChannelEthereumService channelEthereumService = new ChannelEthereumService();
    channelEthereumService.setChannelService(service);
    credentials = Credentials.create(Keys.createEcKeyPair());

    web3j = Web3j.build(channelEthereumService, service.getGroupId());

    startLoginPage();
}

```

之后再在MainPage.java中将Company实例化，之后，便可以使用对象调用contract了

2.4.3 前端

前端部分使用简单的java UI进行设计，代码与后端合并

2.5 核心代码

- 创建银行账号

```

public void create_bank_account(BigInteger initMoney) throws Exception {
    TransactionReceipt rec = com.CreateBankAccount(this.__user_name__, this.pulic_user_key, initMoney).send();
    List<Log> logs = rec.getLogs();
    TransactionDecoder td = TransactionDecoderFactory.buildTransactionDecoder(this.ABI, "");
    Map<String, List<List<EventResultEntity>>> map_1_1 = td.decodeEventReturnObject(logs);
    System.out.println(map_1_1);
    set_bank_account_page();
}

```

账号密码错误则要求重输，正确则进入下一步

- 信用测试

```

public RemoteCall<TransactionReceipt> judgeIfBelieve(String cadd) {
    final Function function = new Function(
        FUNC_JUDGEIFBELIEVE,
        Arrays.<Type>asList(new org.fisco.bcos.web3j.abi.datatypes.Address(cadd)),
        Collections.<TypeReference?>>emptyList());
    return executeRemoteCallTransaction(function);
}

```

在sol中进行测试

- 余额

```

public DeclineMoney(String _userkey, String abi, Company _company, MainPage _mpt) {
    this.company = _company;
    this.ABI = abi;
    this.PubUserKey = _userkey;
    this.mpt = _mpt;
    initialize();
}

```

在Declinemoney中进行操作

- 创建账单

```
public void build_receipy(BigInteger _amount) throws Exception {
    String selectedStr = (String) comboBox.getSelectedItemAt();
    String targetPubUserKey = main.getPubUserKey(selectedStr);
    TransactionReceipt receipt = this.maaainpage.company.creatReceipt(maaainpage.PubUserKey, targetPubUserKey, _amount).send();
    List<Log> logs = receipt.getLogs();
    TransactionDecoder td = TransactionDecoderFactory.buildTransactionDecoder(this.maaainpage.receipyABI, "");
    Map<String, List<List<EventResultEntity>>> map_list_list = td.decodeEventReturnObject(logs);
    System.out.println(map_list_list);
    this.maaainpage.SetReceipyData();
}

public BuildReceipy(MainPage _mp) {
    this.maaainpage = _mp;
    initialize();
}
```

在BuildRecipy中调用接口进行创建

- 转移账单

```
public void TransferReceipy(BigInteger _amount) throws Exception {
    String selectedStr = (String) com_box.getSelectedItemAt();
    String alreadyTargetPubUserKey = main.getPubUserKey(selectedStr);
    String _to = (String) com_box_1.getSelectedItemAt();
    String targetPubUserKey = main.getPubUserKey(_to);
    TransactionReceipt receipt = this.mp.com.transferReceipy(alreadyTargetPubUserKey, mp.pulic_user_key, targetPubUserKey, _amount).send();
    List<Log> logs = receipt.getLogs();
    TransactionDecoder td = TransactionDecoderFactory.buildTransactionDecoder(this.mp.receipyABI, "");
    Map<String, List<List<EventResultEntity>>> map_list_list = td.decodeEventReturnObject(logs);
    System.out.println("iningienigeignei");
    System.out.println(map_list_list);
    if (map_list_list.get("transferReceipyFailedEvent(string)") != null) {
        String data = (String) map_list_list.get("transferReceipyFailedEvent(string)").get(0).get(0).getData();
        if (data.equals("not believed error")){
            Warning.ShowWarning("Denied! ");
        }
    }
    this.mp.set_rec_data();
}

public TransferReceipy(MainPage _mp) {
    this.mp = _mp;
    initialize();
}
```

进行验证地址有效，查询地址间账单后进转移

- 缴清账单

```
public void Pay_Bill(BigInteger _amount) throws Exception {
    String selectedStr = (String) com_box.getSelectedItemAt();
    String targetPubUserKey = main.getPubUserKey(selectedStr);
    TransactionReceipt receipt = this.mp.com.AskMoneyFromReceipy(mp.pulic_user_key, targetPubUserKey, _amount).send();
    List<Log> logs = receipt.getLogs();
    TransactionDecoder td = TransactionDecoderFactory.buildTransactionDecoder(this.mp.receipyABI, "");
    Map<String, List<List<EventResultEntity>>> map_list_list = td.decodeEventReturnObject(logs);
    System.out.println(map_list_list);
    this.mp.set_rec_data();
    this.mp.set_bank_account_page();
}
```

使用Pay_Bill进行缴款

三、功能测试

首先构造5个用户：

用户名称	用户ID	用户描述
LuntaiFactor	 700006	
LunguFactory	 700005	
CarFactory	 700004	
Receipy	 700003	
Bank	 700002	


其中LuntaiFactor、CarFactor和LunguiFactor分别代表轮胎公司、汽车公司、轮毂公司。测试过程基于他们三所公司的账户转移进行实现

3.1 签发账单

3.1.1 创建银行并认证

如果想使用银行账单功能，必须首先注册银行账户，调用CreateBankAccount功能，创建银行账户

合约地址:



用户:

CarFactory

方法:

function

CreateBankA

参数:

_companyName

CarFactor

_userAdd

0xc5f8508f5eade1

可看到注册信息已经写入区块链中

eventName : AddAccount(bool status,address companyAdd,string companyType,uint256 money)

data:

companyAdd	0xC5F8508F5eadEFbF5F375bf2d0..
companyName	CarFactor
companyType	1
money	200000

3.1.2 账单签发

汽车公司向轮胎公司购买了14000元轮胎，并签发了14000元账单给轮胎公司。
调用createReceipy功能函数。

合约地址:	0x56121f4cdbb19b39767bd746fa		
用户:	CarFactory		
方法:	function	▼	creatReceipy
参数:	_from	f2d0e656a94f5e931c	
	_to	b1350a20f5cdf4970bb	
	_amount	14000	

由于账单的发起公司是汽车公司，汽车公司在银行的现有资产是两万元，超过银行的信任阈值一万元，所以可以信任，在judgeBelief 事件中可以看到是通过信任测试的，只有通过信任测试的账单，才能向下游传递。

eventName : refreshReceipyEvent(bool ifBelief,address fromAdd
mount)

data:

ifBelief	 true
fromAdd	 0xC5F8508F5eadEFbF5F375bf2d
toAdd	 0xE8673Fd98d0E55f92012b1350a
amount	 14000

至此，建立了一笔额度为14000元的，从汽车工厂到轮胎工厂的账单

3.2 转让账单

3.2.1 查询现有账单

调用logAllReceipy接口函数可以查询到所有账单，包括欠款和债权两种有关账单。

eventName : logReceipyEvent(bool ifBelief,address fromAdd,address toAdd,uint256 amount)

data:

ifBelief	 true
fromAdd	 0xC5F8508F5eadEFbF5F375bf2d0..
toAdd	 0xE8673Fd98d0E55f92012b1350a2..
amount	 14000

3.2.2 转让账单并确认

调用账单转移接口transferReceipy进行已有账单的转移。

用户: LuntaiFactor ▼

方法: function ▼ transferRece ▼

参数:

_fromUserAddr	a94f5e931e
_to	!b1350a20f5cdf4970bb6
_toUserAddr	l3c89fd70b828
_amount	4000

eventName : transferReceipyEvent(bool ifBelief,address oriFromAdd,address toAdd,uint256 amount,bool isOriDeleted)

data:

oriToAdd	0xE8673Fd98d0E55f92012b1350a2.
toAdd	0x8Bb8Ce67E28Aa3dFE71C63a60..
amount	4000
isOriDeleted	false

确认查询:





eventName : logReceipyEvent(bool ifBelief,address fromAdd,address toAnt)

data:

ifBelief	true
fromAdd	0xC5F8508F5eadEFbF5F375bf2d0..
toAdd	0xE8673Fd98d0E55f92012b1350a2.
amount	10000

address: 0x49b2f7c6b1405013a696c08aa70a267a7c798f1f


eventName : logReceipyEvent(bool ifBelief,address fromAdd,address toAdd,nt)

data:	
ifBelief	 true
fromAdd	 0xC5F8508F5eadEFbF5F375bf2d0..
toAdd	 0x8Bb8Ce67E28Aa3dFE71C63a60..
amount	 4000

3.3 融资

调用融资接口 transferReceipy

eventName : transferReceipyEvent(bool ifBelief,address oriFromAdd, address toAdd,uint256 amount,bool isOriDeleted)

data:	
oriToAdd	 0x8Bb8Ce67E28Aa3dFE71C63a60..
toAdd	 0xa3499AAdb7894ba292fa18c201C..
amount	 2000
isOriDeleted	 false

可以得到银行融资记录，设置的融资金额为2000元


由于当前账单为4000元，所以融资之后原始账单依然存在，没有被删除，原始汽车公司到轮毂的账单应 该变为2000元

完成之后查询各个账单和余额：

轮毂公司：

eventName : SearchMoneyEvent(uint256 money)


data:

name	data
money	 12000

银行:

eventName : SearchMoneyEvent(uint256 money)

data:

name	data
money	 998000

欠轮胎公司的账单:

eventName : logReceipyEvent(bool ifBelief,address fromAdd,address toAnt)

data:

ifBelief	 true
fromAdd	 0xC5F8508F5eadEFbF5F375bf2d0..
toAdd	 0xE8673Fd98d0E55f92012b1350a2.
amount	 10000






欠轮胎公司的账单:

eventName : logReceipyEvent(bool ifBelief,address fromAdd,address toAdd)

data:		^
ifBelief	 true	
fromAdd	 0xC5F8508F5eadEFbF5F375bf2d0..	
toAdd	 0x8Bb8Ce67E28Aa3dFE71C63a60..	
amount	 2000	
		▼

欠银行的账单:

eventName : logReceipyEvent(bool ifBelief,address fromAdd,address toAdd)

data:		^
ifBelief	 true	
fromAdd	 0xC5F8508F5eadEFbF5F375bf2d0..	
toAdd	 0xa3499AAdb7894ba292fa18c201C..	
amount	 2000	
		▼

3.4 支付结算

支付结算功能是由欠款方给债权方一定数额的金额，用于支付之前的账单，账单全部支付完成后账单则作废。

现在我让汽车公司给轮胎公司支付之前的10000元账单。调用AskMoneyFromReceipy接口进行支付结算。

结算前:

轮胎公司为15000元

汽车公司为200000元

eventName : SearchMoneyEvent(uint256 money)

data:

name	data
money	📄 15000

eventName : SearchMoneyEvent(uint256 money)

data:

name	data
money	📄 200000

结算后:

轮胎公司的金额增加了10000元




汽车公司金额减少10000元

eventName : MoneyChange(bool isIncrease,uint256 amount,uint256 a

data:

name	data
isIncrease	📄 true
amount	📄 10000
amount_after	📄 25000

eventName : MoneyChange(bool isIncrease,uint256 amount,uint256
data:

name	data
isIncrease	 false
amount	 10000
amount_after	 190000

清算数据上链:

eventName : payForReceiptEvent(bool isPayAll,address fromAdd,address
amount)

data:

isPayAll	 true
fromAdd	 0xC5F8508F5eadEFbF5F375bf2d0..
toAdd	 0xE8673Fd98d0E55f92012b1350a2.
amount	 10000

四、界面展示

4.1 登录界面

Login

输入账号密码登录

4.2 主界面

Car

Wheel hub Company

Wheel hub Company

Balance:10000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Creditor Bill

Arrears Bill

整个程序的主界面

4.3 存款

Car

Wheel hub Company

Wheel hub Company

Balance:10000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Deposit

Input how much you want yo input

取消

好

存入1000

4.4 存款结果

Car

Wheel hub Company

Wheel hub Company

Balance:11000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Creditor Bill

Arrears Bill

可见余额变为11000

4.5 取款

Car

Wheel hub Company

Wheel hub Company

Balance:11000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Deposit

1000

取消 好

取出1000

4.6 取款结果

Car

Wheel hub Company

Wheel hub Company

Balance:10000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Creditor Bill

Arrears Bill

可见余额变为10000

4.7 创建账单

Car

Wheel Company

Wheel hub Company

Balance:10000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Create Bill

1000

取消

好

创建一个账单

Owe wheel company 1000 yuan

关闭

收到创建成功提示

4.8 创建账单后

Car

Wheel Company

Wheel hub Company

Balance:10000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Creditor Bill

Arrears Bill

Owe: wheel company 1000 yuan

可以看到界面上出现了欠款账单

4.9 转移账单

Wheel

Car Company

Wheel hub Company

Balance:2000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Have Bill:

Transfer bills

600

取消

好

将账单转移给另一个公司

4.10 转移结果

Wheel

Car Company

Wheel hub Company

Balance:2000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Arrears Bill

Have Bill: car company 400 yuan

Arrears Bill

轮胎公司的账单变为400

Wheelhub

Wheel hub Company

Wheel hub Company

Balance:2000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Creditor Bill

Have Bill: car company 600 yuan

Arrears Bill

轮毂公司的账单变为600

4.11 融资

Wheelhub

Wheel Company

Wheel hub Company

Balance:2000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Have Bill:

Finacing

400

取消

好

轮毂公司使用来自汽车公司的账单向银行融资

4.12 融资结果

Wheelhub

Wheel Company

Wheel hub Company

Balance:2400

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Creditor Bill

Have Bill: car company 200 yuan

Arrears Bill

轮毂公司存款增加，账单减少

Car

Wheel hub Company

Wheel hub Company

Balance:10000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Arrears Bill

Owe:wheel 400 company yuan

Creditor Bill

Owe:wheelhub company 200 yuan

Owe:bank 400 yuan

汽车公司可以看到账单明细

4.13 支付欠款

Car

Wheel CompanyWheel hub Company

Balance:10000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Payment

400

取消好

汽车公司支付一笔欠款

4.14 支付后

Car

Wheel CompanyWheel hub Company

Balance:9600

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Creditor Bill

Arrears Bill

Owe:wheelhub company 200 yuan

Owe:bank 400 yuan

存款减少，小于10000

4.15 信用审核

Car

Wheel Company

Wheel hub Company

Balance:9600

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Creditor Bill

Arrears Bill

Owe:wheelhub company 200 yuan

Owe:bank 400 yuan

Owe: wheel company 1000 yuan

汽车公司再创建一笔1000的给轮胎公司的账单

Wheel

Car Company

Wheel hub Company

Balance:2000

Deposit

Withdraw

Create bills

Transfer bills

Finacing

Payment

Have Bill:

该用户存款少于10000，不可信赖暂时无法转移账单。

关闭

由于汽车公司存款小于10000，轮胎公司尝试将账单转移时失败

五、体会心得

这次的大作业可谓是收获良多，串通了我一整个学期学到的关于区块链的知识，并督促我将这些知识付诸实践。在此写下一些感想。

首先是关于区块链这个领域，之前只闻其名却未真正学习，一直以为是一种很高大上的，只能用在比特币上的技术。通过这学期老师的讲解和阅读论文我才知道，原来区块链也是一种非常数学的理论，但是并不黑盒，了解了相关原理之后就有一种豁然开朗的感觉（虽然我还是觉得很难）。同时也知道了区块链技术在金融领域有着广泛的运用空间，而不仅仅局限在比特币上，比如这次的作业的信任链问题就是一个非常传统的问题，通过使用区块链技术可以大大的减少时间和人力成本，可谓新时代的必然选择。

说会这次项目，由于这学期比较忙，关于实习，实验室以及一些其他的事情，分散了一部分精力，导致本次项目没能按照预定的时间提交，但是后来我还是决定好好的面对这本科最后一个大项目，在请教了同学和老师之后，又自己对相关的fisco啊之类的知识进行了紧张的学习之后开始正式写这个项目，最后总算是ddl临近之前肝出来了。

其实我之前一直以为这次可以使用第二次作业的Webase，我还想着挺简单，后来才知道不能直接用他的接口，要用SDK进行链端函数的交互，在折腾了一番之后也总算是实现了。不过这导致的就是我的.sol相比于第二次的时候有很大的变化，添加了一些功能，写起来也经历了一些问题。

后来又急急忙忙写了前端（虽然有点丑），但是也是我在有限的时间内能拿出的最好的成果了，希望TA不要嫌弃~

在新年前总算是完成了这次的项目和报告，也在这里提前预祝祝TA新年快乐，过个好年！