2023年全国职业院校技能大赛

高职组

“区块链技术应用”

赛项赛卷（6卷）

任

务

书

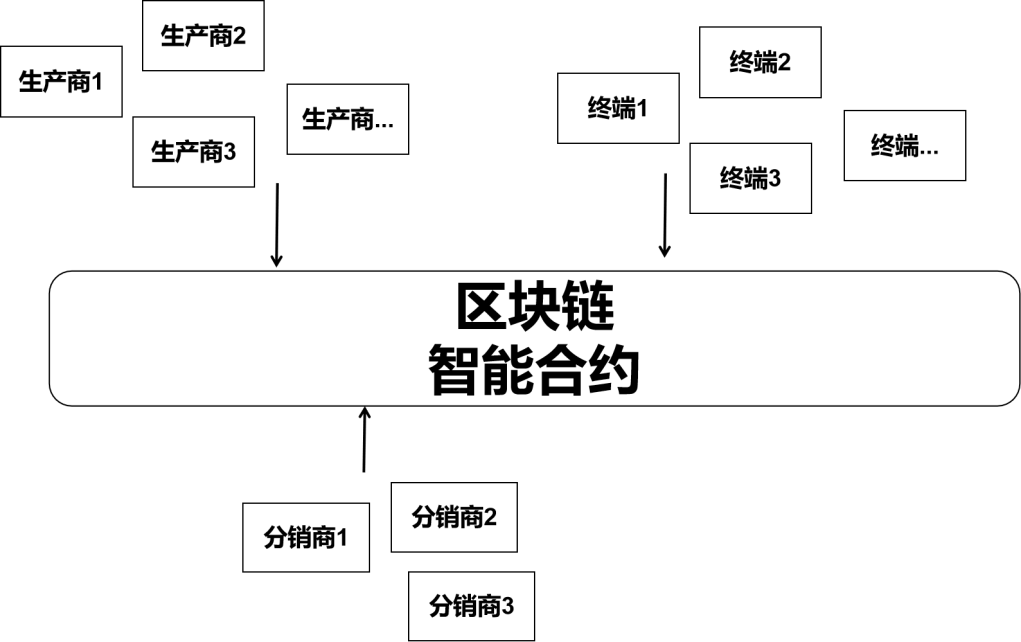
参赛队编号：

**背景描述**

近年来，食品安全问题层出不穷，涉及到各种食品类别，如肉类、水果、蔬菜等。食品安全事故不仅影响消费者的身体健康，还会导致社会恐慌，进而影响国家经济的稳定发展。随着全球化进程的加速，食品生产、加工、运输等环节变得更加复杂，食品安全溯源变得愈发困难。在这种背景下，采用先进的技术手段来保障食品安全，提高食品质量，已成为迫切的需求。

区块链技术为食品安全溯源提供了一种创新的解决方案。区块链技术能够实现数据的不可篡改、可追溯、透明共享等特性。这些特性使得区块链技术在食品安全溯源方面具有很大的潜力。

通过建立基于区块链技术的食品安全溯源平台，我们可以将食品安全溯源相关信息（如生产日期、产地、生产商和流通企业等）以去中心化的方式存储在区块链上，确保数据的真实性和不可篡改性。另外，利用区块链的智能合约技术，可以在保证数据安全的同时实现透明公开的食品安全溯源业务流程设计。在此基础上，引入监管机制有助于确保业务的健康发展。



## 模块一：区块链产品方案设计及系统运维（35分）

选手完成本模块的任务后，将任务中设计结果、运行代码、运行结果等截图粘贴至客户端桌面【区块链技术应用赛\重命名为工位号\模块一提交结果.docx】中对应的任务序号下。

### 任务1-1：区块链产品需求分析与方案设计（10分）

**子任务1-1-1 食品安全溯源区块链产品需求分析（6分）**

1. 根据项目背景描述，请设计一款基于区块链技术的食品溯源系统。请至少包括以下内容：（4分）

1. 用户群体及需求分析
2. 区块链技术分析
3. 需求总结和优先级排序

2. 根据设计的食品溯源区块链系统，进行功能性与非功能性需求分析（2分）

**子任务1-1-2 食品安全溯源区块链产品设计方案（4分）**

设计一个基于区块链技术的食品安全溯源系统，满足以下需求：

* 食品溯源：通过系统可以追溯到每一个食品的生产、加工、运输、销售等环节，并能够记录和管理每一环节的数据和信息。
* 数据共享：系统应支持数据共享，各参与方可以通过系统实现数据的实时共享，提高数据的透明度和可信度，加强食品安全监管。
* 用户权限管理：系统应支持用户权限管理，各参与方可以根据自身权限进行数据的访问和管理，保障数据的安全和隐私。
* 防篡改性：系统应具备数据防篡改性，防止数据被篡改和伪造，确保数据的真实性和可信度。

请你根据以上需求设计一个食品安全溯源区块链系统，并给出系统的架构、数据结构、功能模块和技术方案。

### 任务1-2：区块链系统部署与运维（15分）

围绕食品安全溯源区块链平台部署与运维需求，进行项目相关系统、节点以及管理工具的部署工作。通过监控工具完成对网络、节点服务的监控。最终利用业务需求规范，完成系统日志、网络参数、节点服务等系统结构的维护，具体要求如下：

1. 根据参数与端口设置要求，部署区块链系统并验证。
2. 根据参数与端口设置要求，部署区块链网络管理平台并验证。
3. 基于区块链系统相关管理平台，按照任务指南实施系统运维工作并验证。
4. 基于区块链系统相关监管工具，按照任务指南对区块链系统进行监管。

**子任务1-2-1： 搭建区块链系统并验证（4分）**

基于给定服务器环境以及软件（地址“/root/tools”），使用Docker配置单机4节点的区块链系统，具体要求如下：

|  |  |
| --- | --- |
| P2P起始端口 | 30500 |
| channel起始端口 | 20500 |
| JSONRPC | 8945 |

完成任务如下：

1. 完成系统搭建配置。

2. 完成单机4节点区块链平台搭建，成功运行区块链系统。

3. 使用基于Docker命令查看区块链系统状态。

4. 检查区块链系统节点node0连接状态输出。

**子任务1-2-2：搭建区块链系统管理平台并运维（4分）**

基于给定服务器环境以及软件（地址“/root/tools”），搭建区块链控制台并开展相关运维工作，具体工作内容如下：

1. 配置控制台，管理相关证书并启动。

2. 使用控制台查询区块链中区块高度。

3. 使用控制台查询区块链中创世区块信息。

4. 使用控制台查询区块链中共识状态。

**子任务1-2-3：区块链系统权限管理（4分）**

基于给定服务器环境以及软件（地址“/root/tools”），对区块链进行权限管理，具体工作如下：

1. 使用控制台，分别创建Account1、Account2、Account3用户并形成委员会。

2. 使用控制台，修改Account1的投票权重为2。

3. 使用控制台，将Account2从委员会中剔除，并赋予普通用户权限。

**子任务1-2-4：区块链系统监控（3分）**

基于搭建完成的区块链系统，编写监控脚本，检测区块链程序的运行情况，最后将执行结果截图保存。具体要求：

1. 编写脚本1，每隔1秒检查一次区块链节点进程数量，若为4则打印正常信息，否则打印错误信息。

2. 编写脚本2，每隔3秒检查一次区块链节点进程占用的端口数量，若为12则打印正常信息，否则打印错误信息。

### 任务1-3：区块链系统测试（10分）

设计对区块链系统的测试流程；结合实际业务需求，调用部署的智能合约中进行系统测试、性能测试等；根据业务需求，分析并且修复给定智能合约中的安全漏洞。利用模拟业务和测试工具完成区块链系统服务数据的测试。

1. 使用命令启动区块链系统可视化一体平台并验证启动情况（1分）

2. 通过可视化平台生成包括生产商(producer)、经销商(distributor)、零售商(retailer)账户，并将账户以p12加密形式导出后，导入指定的前置可视化平台，并验证地址一致性（1分）

3. 使用Postman对上述功能接口进行验证，并将验证结果截图提交工程文档。

* 使用Postman对食品溯源系统服务端“获取食品”(/food)功能接口进行验证，并将验证结果截图。（1分）

|  |  |
| --- | --- |
| 请求路由: | /food |
| 请求方法： | GET |
| 输入项说明： | |  |  |  | | --- | --- | --- | | **输入项** | **类型** | **说明** | | traceNumber | String | 食品编号 | |
| 输出项说明： | |  |  |  | | --- | --- | --- | | **输出项** | **类型** | **说明** | | current | String | 食品当前所处用户地址 | | name | String | 食品名称 | | produce | String | 食品生产者地址 | | timestamp | Timestamp | 食品生产时间戳 | | quality | Integer | 食品质量 | |

4.参照工程项目（地址：“/root/projects”）使用Caliper测试工具对食品安全溯源系统智能合约获取食品信息(getFood)功能进行压力测试。具体要求如下：

* 提供核心测试代码。（2分）
* 设置txNumber=10，tps=1，所有测试通过率为100%。（1分）

5.智能合约安全漏洞测试。（4分）

有如下问题智能合约：

pragma solidity ^0.7.6;

contract TimeLock {

mapping(address => uint) public balances;

mapping(address => uint) public lockTime;

function deposit() external payable {

balances[msg.sender] += msg.value;

lockTime[msg.sender] = block.timestamp + 1 weeks;

}

function increaseLockTime(uint \_secondsToIncrease) public {

lockTime[msg.sender] += \_secondsToIncrease;

}

function withdraw() public {

require(balances[msg.sender] > 0, "Insufficient funds");

require(block.timestamp > lockTime[msg.sender], "Lock time not expired");

uint amount = balances[msg.sender];

balances[msg.sender] = 0;

(bool sent, ) = msg.sender.call{value: amount}("");

require(sent, "Failed to send Ether");

}

}

contract Attack {

TimeLock timeLock;

constructor(TimeLock \_timeLock) {

timeLock = TimeLock(\_timeLock);

}

fallback() external payable {}

function attack() public payable {

timeLock.deposit{value: msg.value}();

timeLock.increaseLockTime(

type(uint).max + 1 - timeLock.lockTime(address(this))

);

timeLock.withdraw();

}

}

* 分析智能合约中存在问题，并说明危害。（1分）
* 根据测试工具中的代码文件，编写测试用例，复现智能合约中存在的漏洞。（1分）
* 创建新的智能合约，修复其中问题，说明修复内容并测试。（2分）

## 模块二：智能合约开发与测试（30分）

### 任务2-1：智能合约设计5分）

根据区块链食品溯源应用需求分析和方案设计文档的描述，完成以下工作要求：

1. 根据区块链食品溯源应用需求分析和方案设计文档的描述，设计智能合约功能。（2分）

2. 设计智能合约中各角色应具备的功能。（1分）

3. 设计智能合约中食品安全溯源功能。（2分）

### 任务2-2：智能合约开发（20分）

1. 食品信息（Food）、成员信息(Member)、生产订单信息(Productions)的结构体功能编码（6分）

（1）编写食品信息实体功能。（2分）

表2.2.1 Food实体说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| id | uint | 食品编号 |
| name | string | 食品名称 |
| foodType | string | 食品类型 |
| quality | string | 质量 |
| spec | string | 规格 |
| other | string | 其他信息 |
| updateTime | uint | 更新时间 |

struct Food{

//①食品编号id

//②食品名称name

//③食品类型foodType

//④质量quality

//⑤规格spec

//⑥其他信息other

//⑦更新时间updateTime

（2）完善智能合约中用户结构体内容（2分）

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| company | string | 公司名称 |
| location | string | 地址 |
| tel | string | 电话 |
| incharge | string | 负责人 |
| mainBusiness | string | 主营业务 |
| credit | uint | 信誉分 |
| businessType | uint | 成员类型(1:生产商 2:收购商 3:运输商 4:销售商) |
| userAddress | address | 创建用户地址 |
| updateTime | uint | 更新时间 |

struct Member {

//①公司名称

string location;//地址

//②电话

string incharge;//负责人

string mainBusiness;//主营业务

uint credit;//信誉值

//③成员类型(1:生产商 2:收购商 3:运输商 4:销售商)

//④创建用户地址;

uint updateTime;

}

（3）编写生产订单(Productions)结构体信息。（2分）

表2.2.2 生产订单Productios结构体信息

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| orderNo | uint | 溯源订单号 |
| produceNo | uint | 生产订单号 |
| orderType | uint | 订单类型(1: 直接付款 2:凭证付款) |
| foodId | uint | 食品编号 |
| price | uint | 单价 |
| num | uint | 生产数量 |
| updateTime | uint | 更新时间 |
| createUser | address | 订单创建人 |

struct Productions {

//①总订单号

//②食品订单号

uint orderType;//订单类型(1: 直接付款 2:凭证)

uint foodId;//产品编号

uint price;//单价

uint num;//生产数量

//③更新时间;

//④订单创建人

}

2.食品溯源(Trace)的接口编码（6分）

（1）根据食品信息结构体，完成食品信息添加相应功能（2分）

function createFood(①, ②, ③, ④, ⑤, string memory other) public returns(⑥) {

foodMap[⑦] = Food(foodId, name, foodType, quality, spec, other, ⑧);

emit CreateFood(foodId, name);

return foodId;

}

（2）编写食品溯源收购商创建收购订单功能。

function Buy(uint orderNo, ①, uint orderType,uint foodId, uint price, uint num) public returns(②) {

require(userRoleMap[msg.sender] == ③ || msg.sender == owner);

buyMap[buyNo] = Buys(orderNo,buyNo, orderType, foodId, price, num, now, ④);

return buyNo;

}

（3）编写食品溯源创建运输订单功能。

function Transport(uint orderNo, ①, uint orderType, uint num, string memory from\_place, string memory to\_place) public returns(②) {

require(userRoleMap[msg.sender] == ③ || msg.sender == owner);

transportMap[transportNo] = Transports(orderNo, transportNo, orderType, num, from\_place, to\_place, now，④ );

return transportNo;

}

3.角色（Role）管理的接口编码（4分）

（1）编写食品溯源增加角色接口，实现添加角色的功能。（1分）

function addRole(address userAddress, uint role) public ①{

userRoleMap[②] = role;

}

（2）编写食品溯源获取角色功能。（1分）

function getUserRole(address userAddress) public view returns(①) {

uint role = userRoleMap[②];

return role;

}

（3）编写食品溯源修改角色功能。（2分）

function changeRole(address userAddress, uint newrole) public ①{

userRoleMap[userAddress] = newrole;

Member storage ②= memberMap[userAddress];

member.businessType = ③;

memberMap[userAddress] = ④;

}

4.合约编译、部署和调用（4分）

（1）解决代码错误和警告，正确编译并部署合约，成功获取部署的合约地址和abi。（1分）

（2）调用食品溯源智能合约的接口，完整验证业务流程。（3分）

任务2-3：智能合约测试（5分）

1. 配置区块链网络（1分）

启动区块链网络，创建新的Workspace，配置对外访问的RPC接口为7545，配置项目的配置文件config.js实现与新建Workspace的连接。

1. 设置producerId和sellederId两个变量（1分）

基于VSCODE加载的测试项目，补全位于test文件夹中foodTraceNew.js文件预操作的方法。在测试文件中添加预定义的方法（在其他方法启动前使用），在方法中分别设置producerId和sellederId两个变量，具体要求如下：

（1）producerId设置为1

（2）sellderId设置为4

1. 补全createMember和getMember方法（1分）

基于VSCODE加载的测试项目，补全位于test文件夹中foodTraceNew.js文件，添加测试用例，测试智能合约的createMember和getMember方法。

1. 测试createOrder和getOrder方法（1分）

基于VSCODE加载的测试项目，补全位于test文件夹中foodTraceNew.js文件，添加测试用例，测试智能合约的createOrder和getOrder方法。

1. 测试createFood和getFood方法（1分）

基于VSCODE加载的测试项目，补全位于test文件夹中foodTraceNew.js文件，添加测试用例，测试智能合约的createFood和getFood方法。

## 模块三：区块链应用系统开发（30分）

### 任务3-1：前端功能开发（10分）

1.请基于前端系统的开发模板，在组件main.js中添加对应的逻辑代码，实现角色生产商的部分功能，并测试功能完整性。（3分）

示例页面：







本任务具体要求如下：

1. 生产商首页展示已有的食品列表，若没有食品则显示暂无数据，展示列有溯源码、食品名称、生产商、生产时间；
2. 生产商首页有“退出”按钮，可以退出登录返回首页；
3. 生产商页面可以新建食品，弹出框中需要用户输入溯源码、视频名称、生产商、质检情况等，单击取消关闭弹出框，单击确定则根据操作反馈进行提示；
4. 生产商首页展示的食品列表可根据生产时间排序。

Main.js

代码片段1：

<!-- 创建蔬菜按钮 -->

<el-button type="primary" @click="popup = true;">*选手填写部分*</el-button>

<!-- 退出按钮 -->

<el-button type="danger" @click="$emit(*选手填写部分*, false)" class="button-logout">*选手填写部分*</el-button>

代码片段2：

<!-- 食品信息列表 -->

<el-table

:data="renderList"

style="width: 100%"

:default-sort =*选手填写部分*

>

<el-table-column

prop="traceNumber"

label=*选手填写部分*

>

</el-table-column>

<el-table-column

prop="foodName"

label=*选手填写部分*

>

</el-table-column>

<el-table-column

prop="traceName"

label=*选手填写部分*

>

</el-table-column>

<el-table-column

prop="date"

label=*选手填写部分*

*选手填写部分*

>

</el-table-column>

</el-table>

代码片段3：

.then(ret => {

if (ret.data.ret !== 1) {

// 已有溯源码校验

if (ret.data.ret === 0 && ret.data.msg === 'traceNumber already exist') {

this.$message({

message: '该溯源码已存在，请重新创建',

type: *选手填写部分*,

center: true

});

} else {

this.$message({

message: *选手填写部分*,

type: *选手填写部分*,

center: true

});

}

return

}

this.$message({

message: *选手填写部分*,

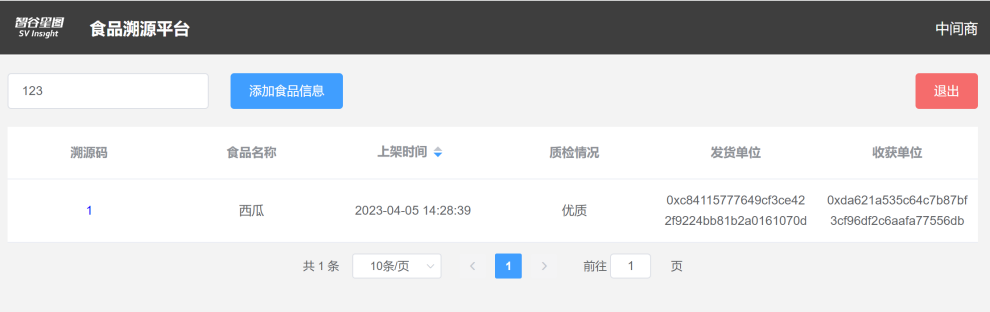
type: *选手填写部分*,

center: true

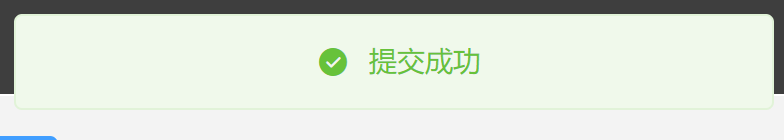
});

2.请基于前端系统的开发模板，在组件components.js中添加对应的逻辑代码，实现角色中间商的部分功能，并测试功能完整性。（3分）

示例页面：









本任务具体要求如下：

1. 中间商首页展示已有的食品列表，若没有食品则显示暂无数据，展示列有溯源码、食品名称、上架时间、质检情况、发货单位、收货单位；
2. 中间商首页有“退出”按钮，可以退出登录返回首页；
3. 中间商页面可以输入溯源码添加食品信息，输入正确的溯源码后在弹出框中展示相关溯源信息，并在弹出框中可以填写质检情况和收货单位，点击取消弹窗消失，点击确定根据反馈进行提示；
4. 中间商首页展示的食品列表可根据上架时间排序；
5. 点击溯源码可显示食品详细信息。

components.js

代码片段1：

<div>

<el-input

placeholder="*选手填写部分*"

v-model.number="traceNumber"

clearable>

</el-input>

<el-button type="primary" @click=*选手填写部分*>添加食品信息</el-button>

<el-button type="danger" @click="$emit(*选手填写部分*, false)" class="button-logout">*选手填写部分*</el-button>

代码片段2：

<!-- 食品流转信息列表 -->

<el-table

:data="renderList"

style="width: 100%"

:default-sort = "{prop: 'date', order: 'descending'}"

>

<el-table-column

prop="traceNumber"

label=*选手填写部分*

>

<template slot-scope="scope">

<!-- 鼠标点击食品细节弹窗信息 -->

<el-popover trigger="click" placement="top" @show="handlePopover(scope.row)" :width="400">

<FoodDetail :food=*选手填写部分* />

<div slot="reference">

<span style="color: blue; cursor: pointer;">{{scope.row.traceNumber}}</span>

</div>

</el-popover>

</template>

代码片段3：

<el-table-column

prop="name"

*选手填写部分*

>

</el-table-column>

<el-table-column

prop="date"

*选手填写部分* *选手填写部分*

>

</el-table-column>

<el-table-column

prop="quality"

*选手填写部分*

>

</el-table-column>

<el-table-column

prop="from"

*选手填写部分*

>

</el-table-column>

<el-table-column

prop="to"

*选手填写部分*

>

</el-table-column>

代码片段4：

template:`

<div>

<el-dialog title="*选手填写部分*" :visible.sync="dialogFormVisible" :show-close="false" width="40%" :center="true">

<el-form :model="form" ref="form" label-position="left" :label-width="formLabelWidth">

<el-form-item

label=*选手填写部分*

prop="quality"

:rules="[

{ required: true, message: *选手填写部分*, trigger: 'change' }

]">

<el-radio-group v-model="form.quality">

<el-radio label="0">优质</el-radio>

<el-radio label="1">合格</el-radio>

<el-radio label="2">不合格</el-radio>

</el-radio-group>

</el-form-item>

<el-form-item

label=*选手填写部分*

prop="traceName"

:rules="[

{ required: true, message: '请输入收货单位' }

]">

<el-input v-model="form.traceName" autocomplete="off"></el-input>

</el-form-item>

</el-form>

<div>

<el-button @click="$emit(*选手填写部分*, false)">取 消</el-button>

<el-button type="primary" *选手填写部分*>确 定</el-button>

</div>

<el-divider></el-divider>

<!-- 食品详情信息 -->

<TraceDetail :food-detail=*选手填写部分*:user=*选手填写部分*/>

3.请基于前端系统的开发模板，在组件components.js中添加对应的逻辑代码，实现角色消费者的部分功能，并测试功能完整性。（4分）

示例页面：



本任务具体要求如下：

1. 消费者页面左侧可进行溯源码搜索；
2. 消费者页面右侧可进行食品基本信息和食品流通信息展示。

main.js

代码片段1：

template:`

<el-row style="height: 100%;" class="consumer">

<!-- 左边为溯源查询 -->

<el-col :span="7" style="height: 100%;">

<div class="grid-content bg-purple">

<el-form :model="form" ref="form">

<h4>*选手填写部分*</h4>

<el-divider></el-divider>

<el-form-item

label=*选手填写部分*

prop="traceNumber"

>

<el-input v-model.number="form.traceNumber" type="textarea" :rows="3" autocomplete="off" @clear="onSearch = false;"></el-input>

</el-form-item>

<el-button type="primary" *选手填写部分* >查询</el-button>

</el-form>

</div>

</el-col>

<!-- 右边为溯源查询结果 -->

<el-col :span="16">

<div class="grid-content bg-purple-light">

<div v-if ="!onSearch" class="consumer-tip">请在左侧查询栏中输入溯源码进行查询</div>

<div v-if ="onSearch && *选手填写部分*" class="consumer-tip">该溯源码无对应信息，请确认后重新查询</div>

<!-- 溯源查询成功后才展示 -->

<div v-if ="onSearch && foodDetail.length">

<!-- 溯源详情组件， user: 3 为消费者 -->

<TraceDetail :food-detail="*选手填写部分*" :user="*选手填写部分*" />

</div>

</div>

</el-col>

<el-col :span="1">

<!--退出按钮 -->

<el-button type="danger" @click=*选手填写部分*class="button-logout">退出</el-button>

</el-col>

</el-row>

代码片段2：

// 处理按溯源码查询

onSubmit() {

if (!this.form.traceNumber) {

this.$message.error(*选手填写部分*);

return;

}

this.onSearch = true;

axios({

method: 'get',

url: `/trace?traceNumber=${this.form.traceNumber}`

})

.then(ret => {

*选手填写部分*

})

.catch(err => {

console.log(err);

})

}

### 任务3-2：区块链应用后端功能开发（20分）

1.请基于后端开发模板，开发完善IndexController类，编写超市添加食品流转信息的方法，实现超市添加食品流转信息的功能，并测试功能完整性。（6分）

本任务具体要求如下：

1. 开发文件IndexController.java中的add\_trace\_by\_retailer方法，请求接口为/addretail；
2. 开发文件IndexController.java中的add\_trace\_by\_retailer方法，要求对前端传入的参数进行二次验证；
3. 开发文件IndexController.java中的add\_trace\_by\_retailer方法，要求封装返回值为String，但不返回视图页面。

IndexController.java

代码片段1：

*选手填写部分* @PostMapping(*选手填写部分*, produces=MediaType.APPLICATION\_JSON\_VALUE)

public String add\_trace\_by\_retailer(@RequestBody JSONObject jsonParam) {

//声明返回对象

JSONObject \_outPutObj = new JSONObject();

*选手填写部分*String trace\_number = *选手填写部分*

String trace\_name = *选手填写部分*

int quality = *选手填写部分*

JSONArray params = JSONArray.parseArray("["+trace\_number+",\""+trace\_name+"\","+quality+"]");

JSONObject \_jsonObj = new JSONObject();

\_jsonObj.put("contractName",CONTRACT\_NAME);

\_jsonObj.put("contractAddress",CONTRACT\_ADDRESS);

\_jsonObj.put("contractAbi",JSONArray.parseArray(CONTRACT\_ABI));

\_jsonObj.put("user",RETAILER\_ADDRESS);

\_jsonObj.put("funcName",*选手填写部分*);

\_jsonObj.put("funcParam",*选手填写部分*);

String responseStr = httpPost(URL,*选手填写部分*);

JSONObject responseJsonObj = JSON.parseObject(responseStr);

String msg = responseJsonObj.getString("message");

if (msg.equals("Success")){

\_outPutObj.put("ret",*选手填写部分*);

\_outPutObj.put("msg",msg);

}else{

\_outPutObj.put("ret",*选手填写部分*);

\_outPutObj.put("msg",msg);

}

return *选手填写部分*

}

2.请基于已有的项目，开发完善IndexController类，编写获取所有食物信息的方法，实现获取所有食物信息的功能，并测试功能完整性。（6分）

本任务具体要求如下：

1. 开发文件IndexController.java中的getlist方法，请求接口为/foodlist，该接口调用私有方法get\_food\_list，不直接与合约交互，提高系统的安全性；
2. 开发文件IndexController.java中的get\_food\_list方法，要求通过合约进行食品列表的查询，且外部无法直接调用；
3. 开发文件IndexController.java中的getlist方法，要求封装返回值为String，但不返回视图页面。

IndexController.java

代码片段1：

*选手填写部分* @GetMapping(*选手填写部分*, produces=MediaType.APPLICATION\_JSON\_VALUE)

public String getlist(){

JSONArray num\_list = get\_food\_list();

JSONArray num\_list2 = num\_list.getJSONArray(0);

JSONArray resList = new JSONArray();

for (int i=0;i<num\_list2.size();i++){

String food = get\_food(*选手填写部分*);

resList.add(food);

}

return *选手填写部分*

}

代码片段2：

*选手填写部分* JSONArray get\_food\_list(){

JSONObject \_jsonObj = new JSONObject();

\_jsonObj.put("contractName",CONTRACT\_NAME);

\_jsonObj.put("contractAddress",CONTRACT\_ADDRESS);

\_jsonObj.put("contractAbi",JSONArray.parseArray(CONTRACT\_ABI));

\_jsonObj.put("user","");

\_jsonObj.put("funcName",*选手填写部分*);

String responseStr = httpPost(URL,*选手填写部分*);

JSONArray responseJsonObj = JSON.parseArray(responseStr);

return responseJsonObj;

}

3.请基于已有的项目，开发完善IndexController类，编写获取食品详细信息的相关方法，并测试功能完整性。（8分）

本题目的具体要求如下：

1. 开发文件IndexController.java中的food方法，请求接口为/food，该接口调用私有方法get\_food，不直接与合约交互，提高系统的安全性；
2. 开发文件IndexController.java中的food方法，对传入数据进行二次验证；
3. 开发文件IndexController.java中的get\_food方法，要求通过合约进行溯源信息的查询，且外部无法直接调用；
4. 开发文件IndexController.java中的food方法，要求封装返回值为String，但不返回视图页面。

IndexController.java

代码片段1：

/\*\*

\* 获取某个食品的当前信息

\* @param traceNumber 食品溯源id，食品溯源过程中的标识符

\* @return 对应食品的当前信息

\*/

@ResponseBody

@GetMapping(path="/food", produces=MediaType.APPLICATION\_JSON\_VALUE)

public String food(String traceNumber){

JSONObject \_outPut = new JSONObject();

if (Integer.parseInt(*选色填写部分*) <= 0){

\_outPut.put(*选色填写部分*,"invalid parameter");

return \_outPut.toJSONString();

}

String res = get\_food(*选色填写部分*);

return res;

}

代码片段2：

/\*\*

\* 从链上获取某个食品的基本信息

\* @param traceNumber: 食品溯源id，食品溯源过程中的标识符

\* @return 对应食品的信息

\*/

private String get\_food(String traceNumber){

JSONArray params = JSONArray.parseArray("["+traceNumber+"]"); ;

JSONObject \_jsonObj = new JSONObject();

\_jsonObj.put("contractName",*选色填写部分*);

\_jsonObj.put("contractAddress",*选色填写部分*);

\_jsonObj.put("contractAbi",JSONArray.parseArray(CONTRACT\_ABI));

\_jsonObj.put("user","");

\_jsonObj.put("funcName",*选色填写部分*;

\_jsonObj.put("funcParam",*选色填写部分*);

String responseStr = httpPost(URL,*选色填写部分*);

JSONArray food = JSON.parseArray(responseStr);

JSONObject \_outPut = new JSONObject();

\_outPut.put(*选色填写部分*);

\_outPut.put(*选色填写部分*);

\_outPut.put(*选色填写部分*);

\_outPut.put(*选色填写部分*);

\_outPut.put(*选色填写部分*);

\_outPut.put(*选色填写部分*);

return *选色填写部分*;

}