智能合约说明文档

赵嘉铖 191250202

目标合约以及功能实现

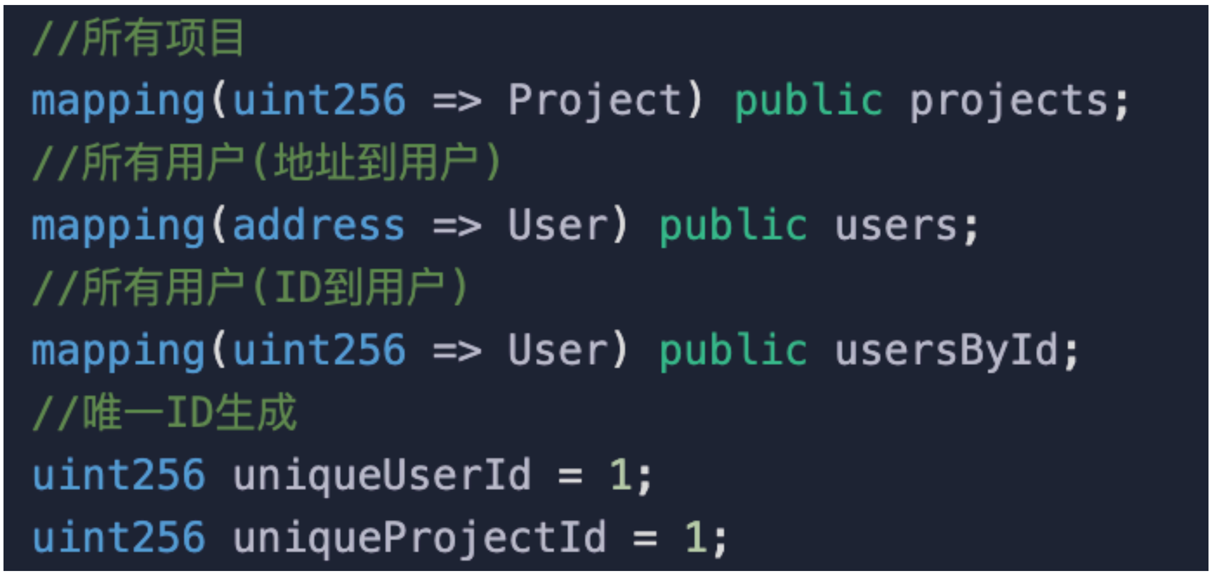
- 获得贡献度模块的用户信息存储合约
- 编写了基础数据类型和框架，编写了用户注册、项目创建、加入项目三个主要功能函数
- 在此基础上编写了便于查询的获取信誉分、贡献度函数

实现步骤与说明

- 创建用户和项目两个结构体

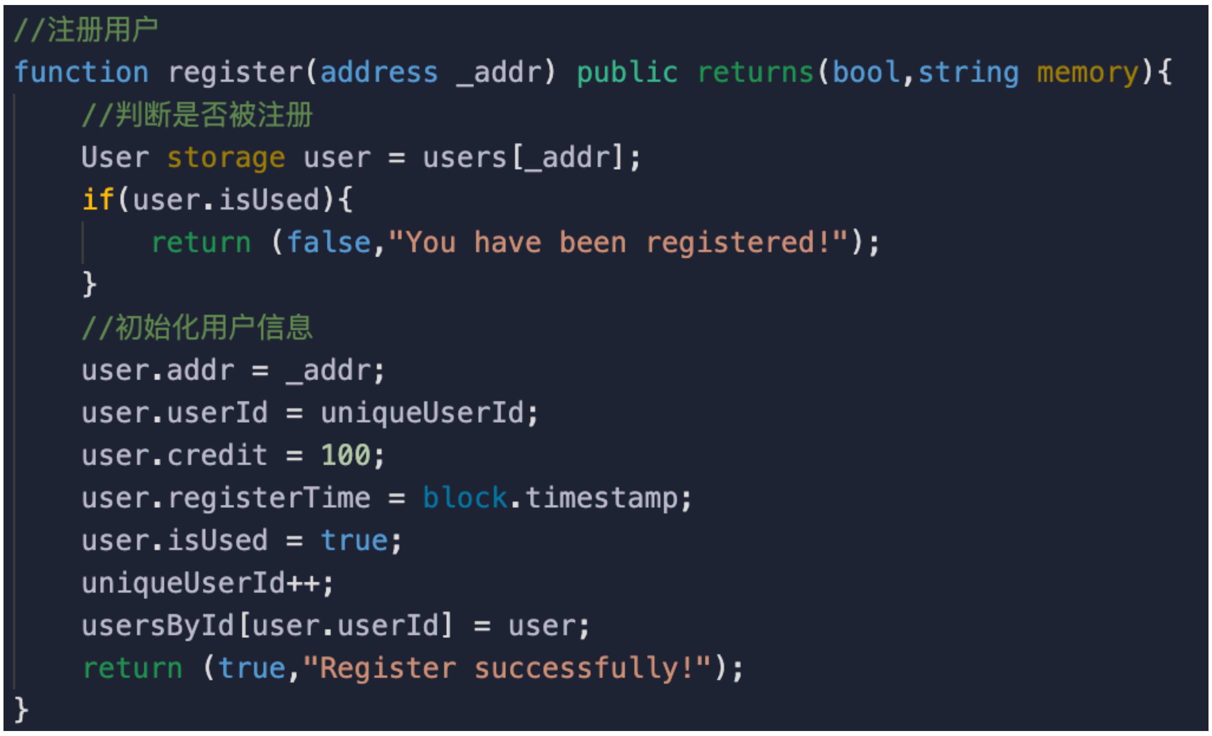
```
//用户
struct User {
    //用户地址、用户ID、唯一信誉分、注册时间、判断引用、所有创建的项目、所有参加的项目
    address addr;
    uint256 userId;
    uint256 credit;
    uint256 registerTime;
    bool isUsed;
    uint256[] myProjects;
    uint256[] joinProjects;
}
//项目
struct Project {
    //项目ID、创建者、创建时间、判断引用、项目贡献者与贡献度映射
    uint256 projectId;
    address creator;
    uint256 createTime;
    bool isCreated;
    mapping(uint256 => uint256) contributors;
}
```

- 创建存储所有用户以及所有项目的映射(为了方便读取，创建了一个从用户ID到用户的映射)



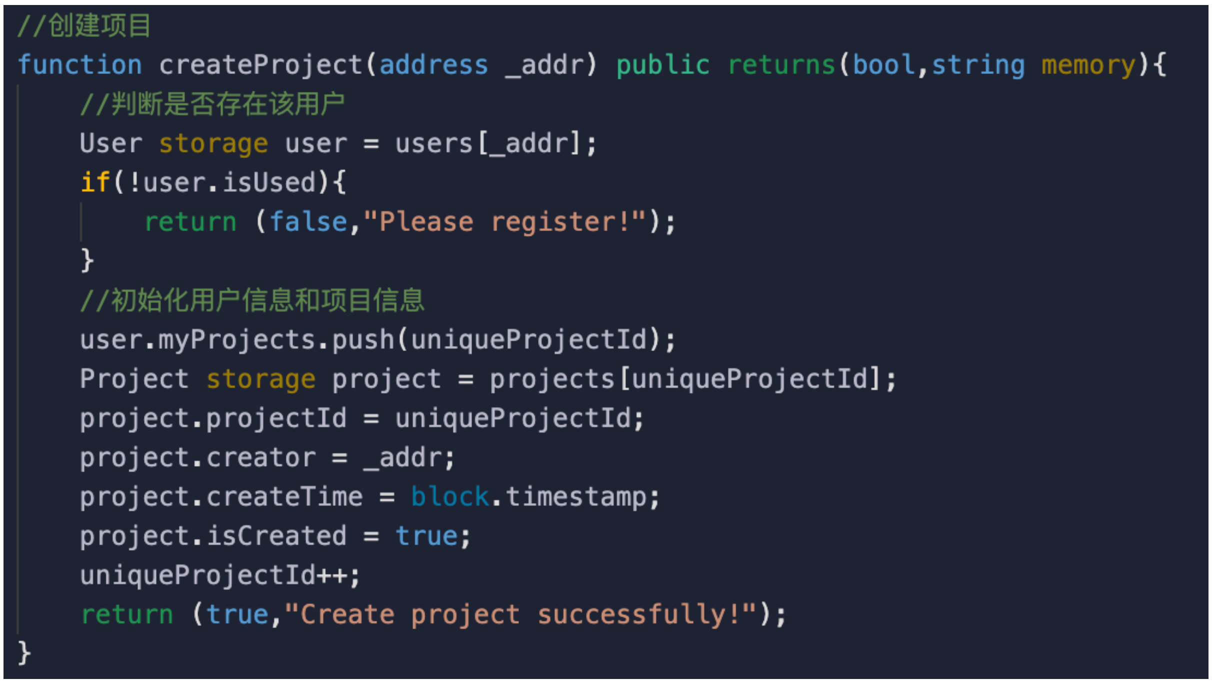
```
//所有项目
mapping(uint256 => Project) public projects;
//所有用户(地址到用户)
mapping(address => User) public users;
//所有用户(ID到用户)
mapping(uint256 => User) public usersById;
//唯一ID生成
uint256 uniqueUserId = 1;
uint256 uniqueProjectId = 1;
```

- 实现注册用户



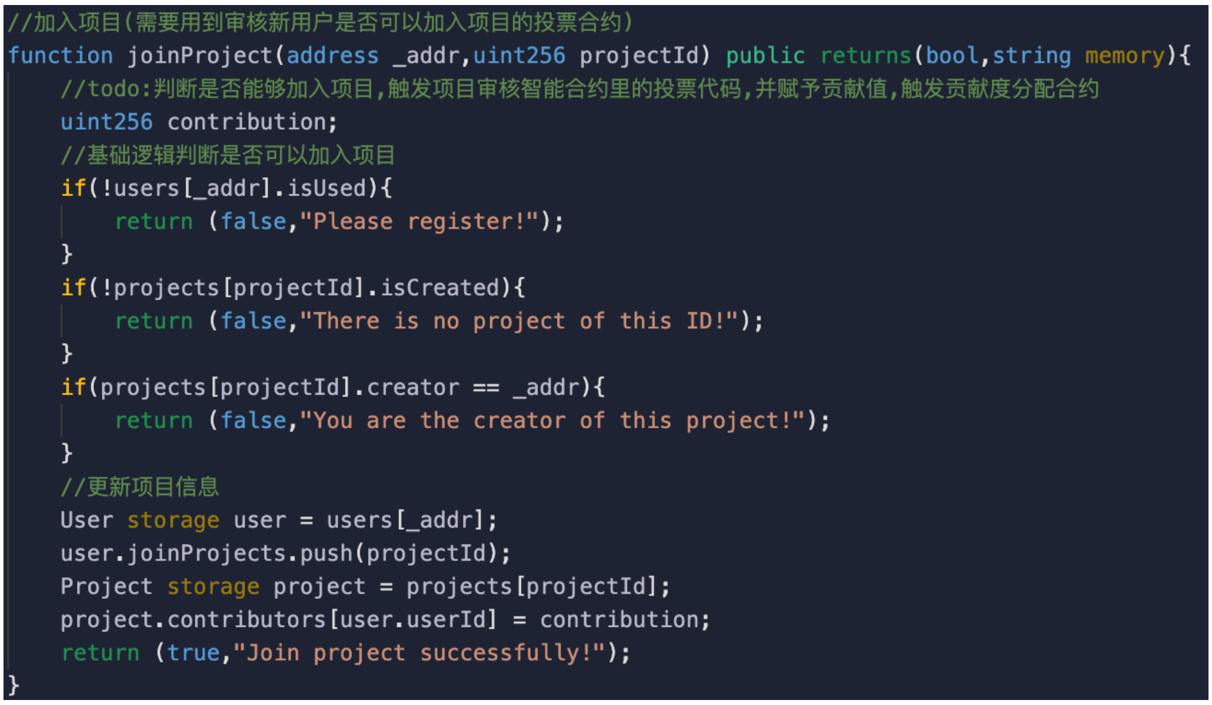
```
//注册用户
function register(address _addr) public returns(bool,string memory){
    //判断是否被注册
    User storage user = users[_addr];
    if(user.isUsed){
        return (false,"You have been registered!");
    }
    //初始化用户信息
    user.addr = _addr;
    user.userId = uniqueUserId;
    user.credit = 100;
    user.registerTime = block.timestamp;
    user.isUsed = true;
    uniqueUserId++;
    usersById[user.userId] = user;
    return (true,"Register successfully!");
}
```

- 实现创建用户



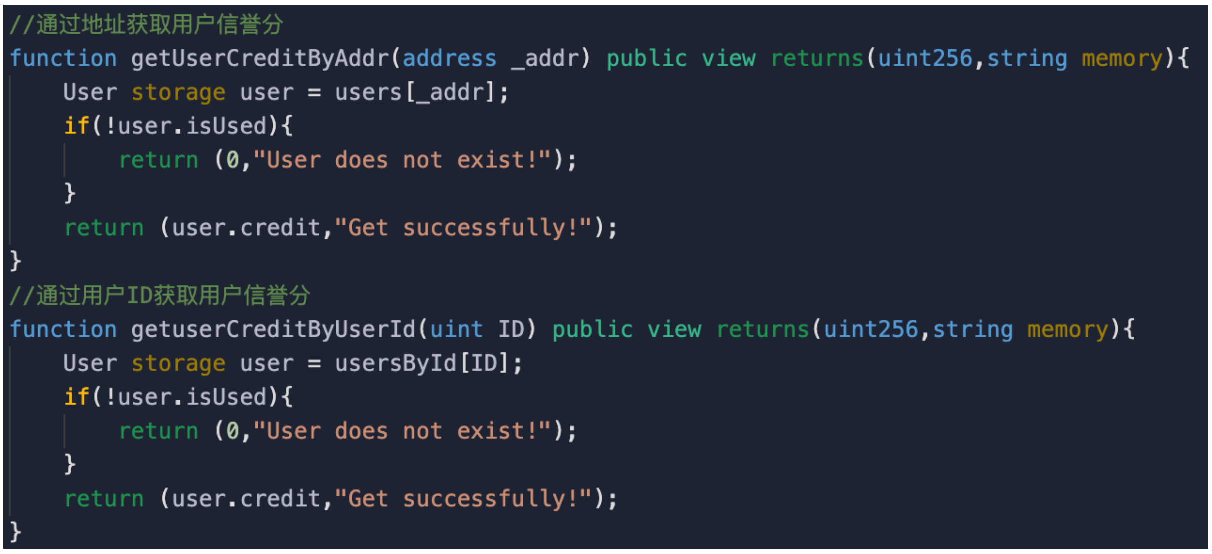
```
//创建项目
function createProject(address _addr) public returns(bool,string memory){
    //判断是否存在该用户
    User storage user = users[_addr];
    if(!user.isUsed){
        return (false,"Please register!");
    }
    //初始化用户信息和项目信息
    user.myProjects.push(uniqueProjectId);
    Project storage project = projects[uniqueProjectId];
    project.projectId = uniqueProjectId;
    project.creator = _addr;
    project.createTime = block.timestamp;
    project.isCreated = true;
    uniqueProjectId++;
    return (true,"Create project successfully!");
}
```

- 实现加入项目函数部分内容(加入项目应该需要用到审核新用户是否可以加入项目的投票合约)



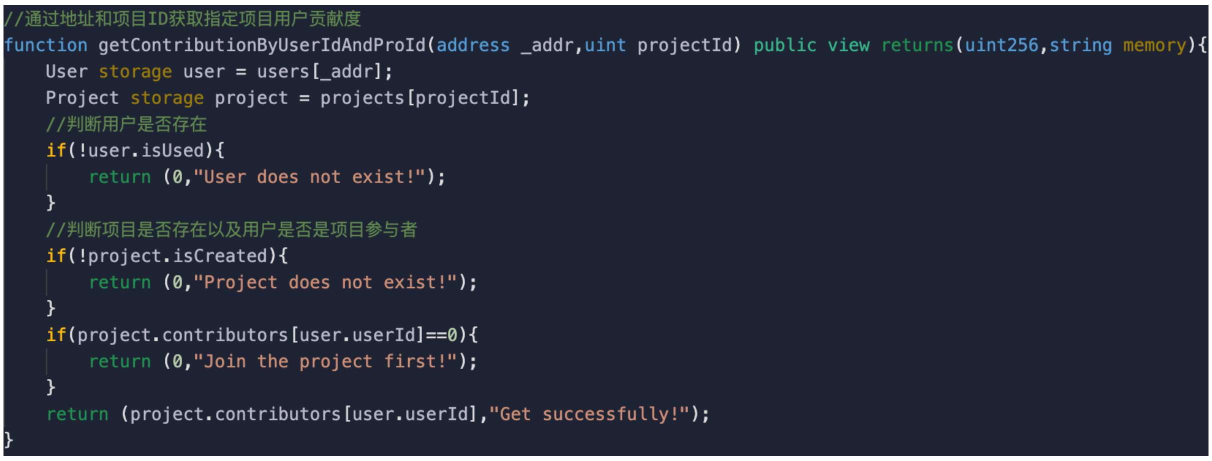
```
//加入项目(需要用到审核新用户是否可以加入项目的投票合约)
function joinProject(address _addr,uint256 projectId) public returns(bool,string memory){
    //todo:判断是否能够加入项目,触发项目审核智能合约里的投票代码,并赋予贡献值,触发贡献度分配合约
    uint256 contribution;
    //基础逻辑判断是否可以加入项目
    if(!users[_addr].isUsed){
        return (false,"Please register!");
    }
    if(!projects[projectId].isCreated){
        return (false,"There is no project of this ID!");
    }
    if(projects[projectId].creator == _addr){
        return (false,"You are the creator of this project!");
    }
    //更新项目信息
    User storage user = users[_addr];
    user.joinProjects.push(projectId);
    Project storage project = projects[projectId];
    project.contributors[user.userId] = contribution;
    return (true,"Join project successfully!");
}
```

- 实现信誉分获取函数(通过地址或者用户ID)



```
//通过地址获取用户信誉分
function getUserCreditByAddr(address _addr) public view returns(uint256,string memory){
    User storage user = users[_addr];
    if(!user.isUsed){
        return (0,"User does not exist!");
    }
    return (user.credit,"Get successfully!");
}
//通过用户ID获取用户信誉分
function getuserCreditByUserId(uint ID) public view returns(uint256,string memory){
    User storage user = usersById[ID];
    if(!user.isUsed){
        return (0,"User does not exist!");
    }
    return (user.credit,"Get successfully!");
}
```

- 实现贡献度获取函数



```
//通过地址和项目ID获取指定项目用户贡献度
function getContributionByUserIdAndProId(address _addr,uint projectId) public view returns(uint256,string memory){
    User storage user = users[_addr];
    Project storage project = projects[projectId];
    //判断用户是否存在
    if(!user.isUsed){
        return (0,"User does not exist!");
    }
    //判断项目是否存在以及用户是否是项目参与者
    if(!project.isCreated){
        return (0,"Project does not exist!");
    }
    if(project.contributors[user.userId]==0){
        return (0,"Join the project first!");
    }
    return (project.contributors[user.userId],"Get successfully!");
}
```