一手认准:优库it资源网 www.ukoou.com

C2N launchpad （社区实践项目）

项目背景

基于社区发展和学员学习进阶需要，C2N社区推出启动平台项目。整体项目定位是社区基础项目实践平台。

项目除了满足学习用途，更加鼓励同学在平台贡献自己的智慧和代码。

项目发展一共分为三个阶段：

1. 第一阶段：学习和任务阶段，C2N技术团队和社区同学一起迭代该项目（4-5月份）
2. 第二阶段：社区内部项目孵化阶段，满足社区同学发挥团队的创造力（6月份开始）
3. 第三阶段：外部合作和开源发展阶段（待定）

项目地址

https://github.com/TechPlanB/C2N-Launchpad/

演示地址

https://c2-n-launchpad.vercel.app/

产品需求

> 内部版本没有kyc，注册流程

C2N launchpad是区块链上的一个去中心化发行平台，专注于启动和支持新项目。它提供了一个平台，允许新的和现有的项目通过代币销售为自己筹集资金，同时也为投资者提供了一个参与初期项目投资的机会。下面是C2N launchpad产品流程的大致分析：

1. 项目申请和审核

- 申请：项目方需要在C2N launchpad上提交自己项目的详细信息，包括项目介绍、团队背景、项目目标、路线图、以及如何使用筹集的资金等。
- 审核：C2N launchpad团队会对提交的项目进行审核，评估项目的可行性、团队背景、项目的创新性、以及社区的兴趣等。这一过程可能还包括与项目方的面对面或虚拟会议。

2. 准备代币销售

- 设置条款：一旦项目被接受，C2N launchpad和项目方将协商代币销售的具体条款，包括销售类型（如公开销售或种子轮）、价格、总供应量、销售时间等。

一手认准:优库it资源网 www.ukoou.com

- 准备市场：同时，项目方需要准备营销活动来吸引潜在的投资者。C2N launchpad也可能通过其平台和社区渠道为项目提供曝光。

3. KYC和白名单

- KYC验证：为了符合监管要求，参与代币销售的投资者需要完成Know Your Customer（KYC）验证过程。
- 白名单：完成KYC的投资者可能需要被添加到白名单中，才能在代币销售中购买代币。

4. 代币销售

- 销售开启：在预定时间，代币销售开始。根据销售条款，投资者可以购买项目方的代币。
- 销售结束：销售在达到硬顶或销售时间结束时关闭。

5. 代币分发

- 代币分发：销售结束后，购买的代币将根据约定的条款分发给投资者的钱包。

用户质押平台币，获得参与项目IDO的购买权重，后端配置项目信息并操作智能合约生成新的sale，用户在sale开始之后进行购买，项目结束后，用户进行claim

平台流程参考

https://medium.com/avalaunch/avalunch-tutorials-platform-overview-1675547b5aff
medium.com

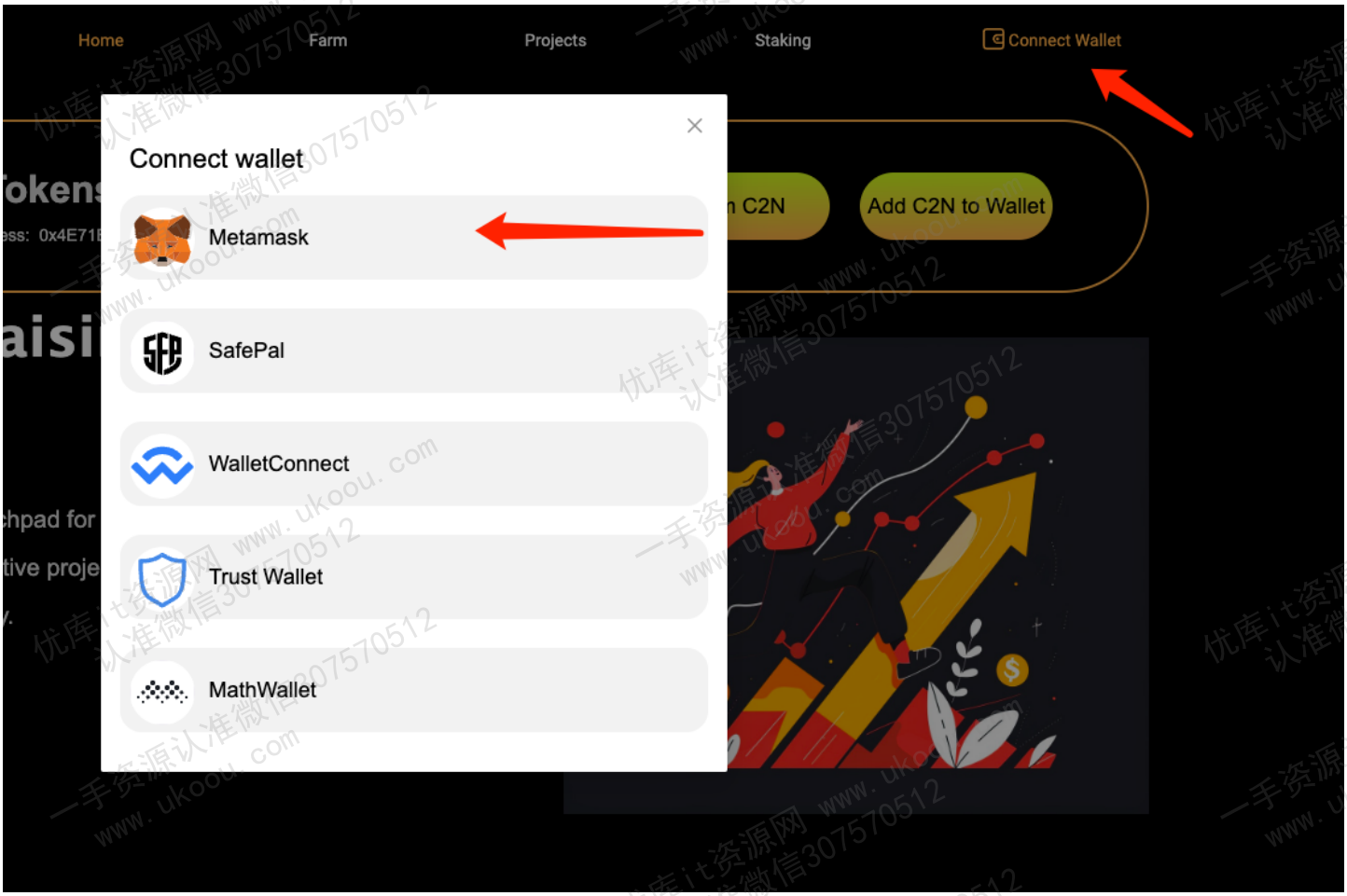
功能操作

初始化

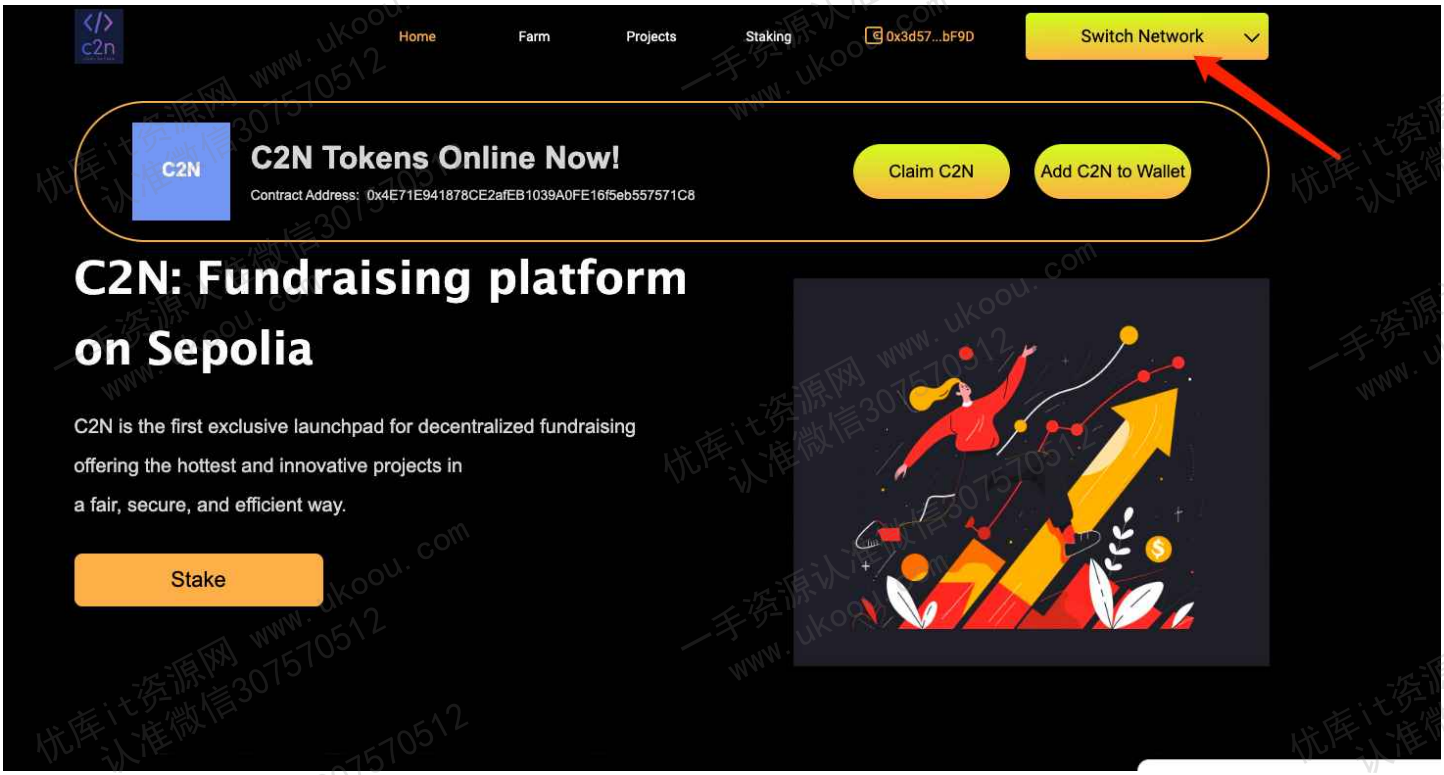
要进行下面的流程，需要提前准备sepolia的测试代币作为gas

1. 连接钱包(推荐metamask)

一手认准:优库it资源网 www.ukoou.com



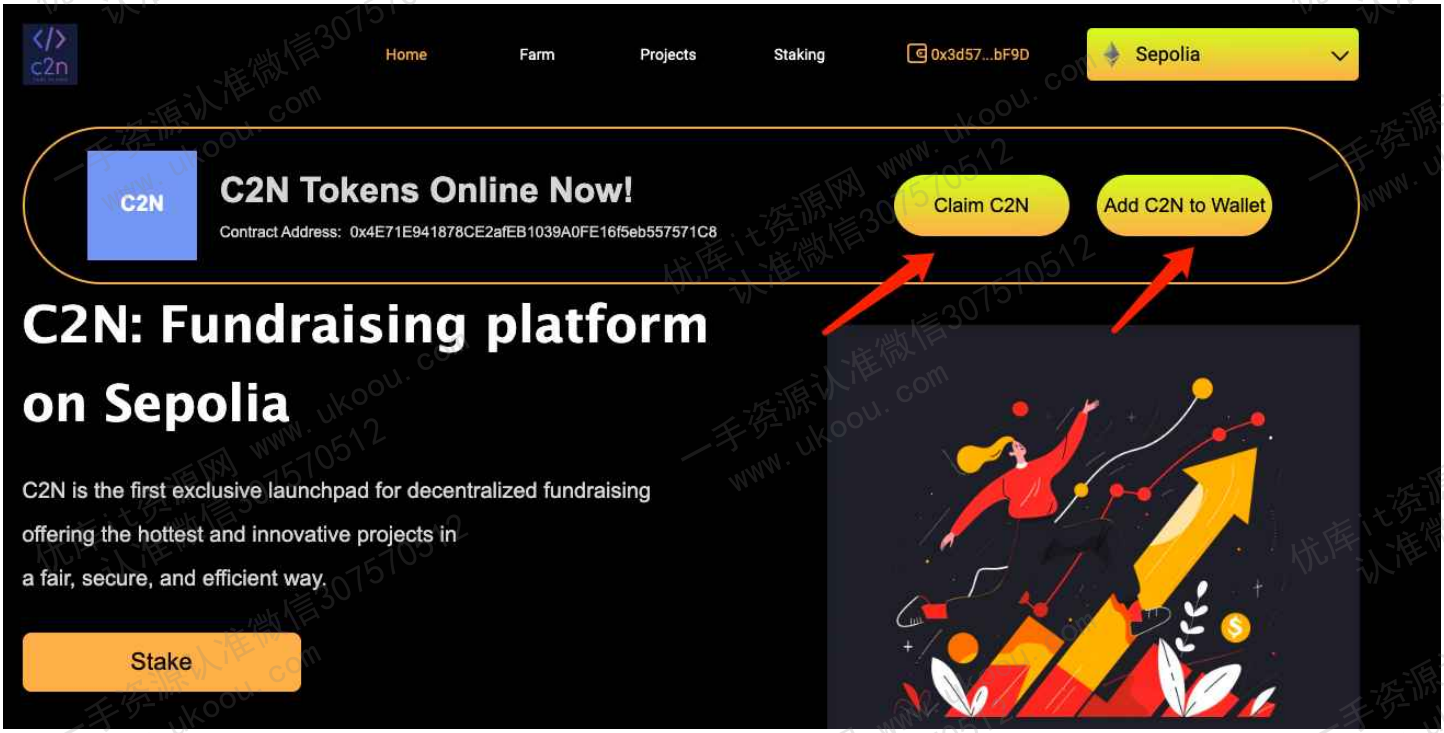
2. 切换网络到sepolia，或者可以直接在钱包里面进行切换



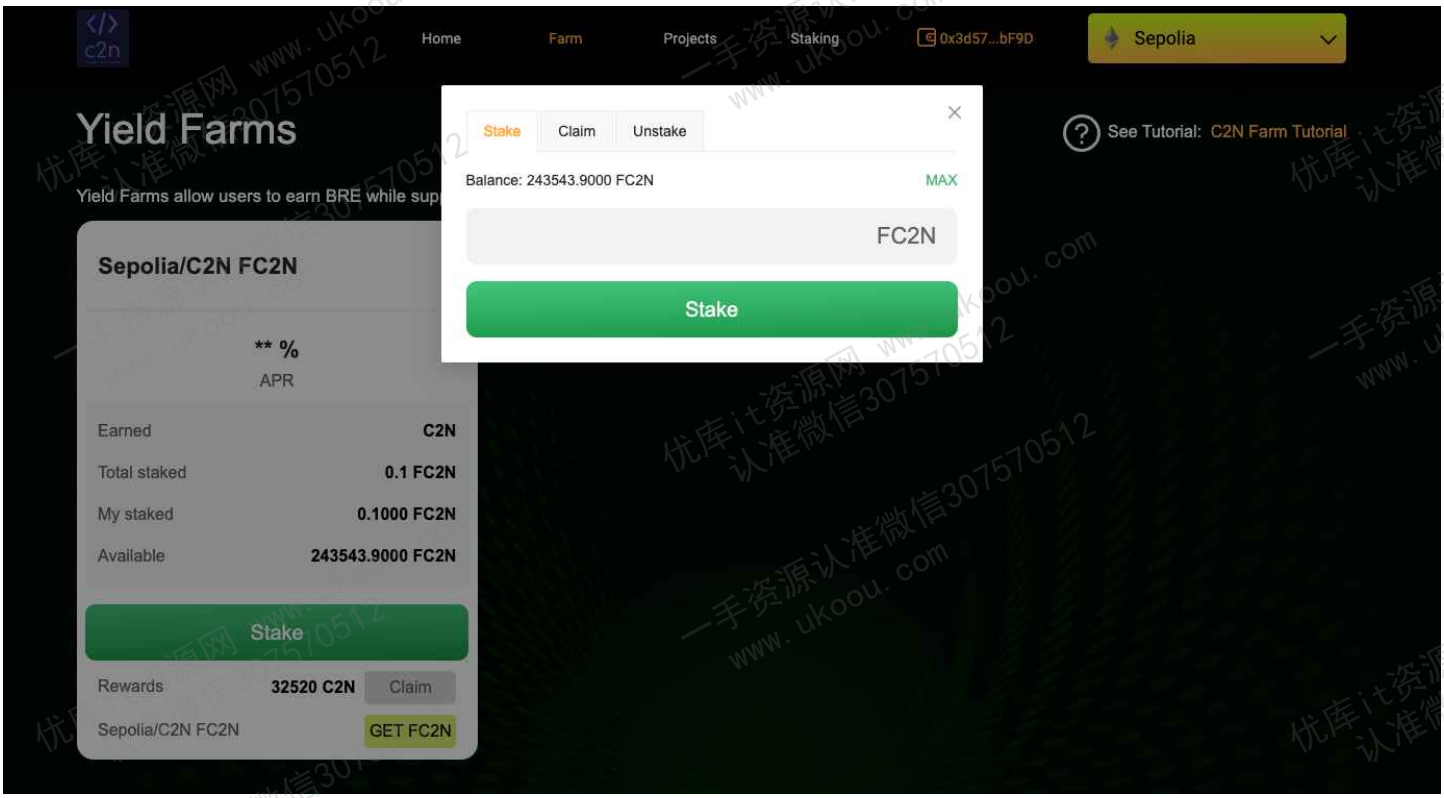
Farm 流程

一手认准:优库it资源网 www.ukoou.com

1. Farm 流程需要用到我们的Erc20测试代币C2N, 可以在首页领取C2N(一个账户只能领取一次),并且添加到我们metamask，添加之后我们可以在metamask 看到我们领取的C2N 代币



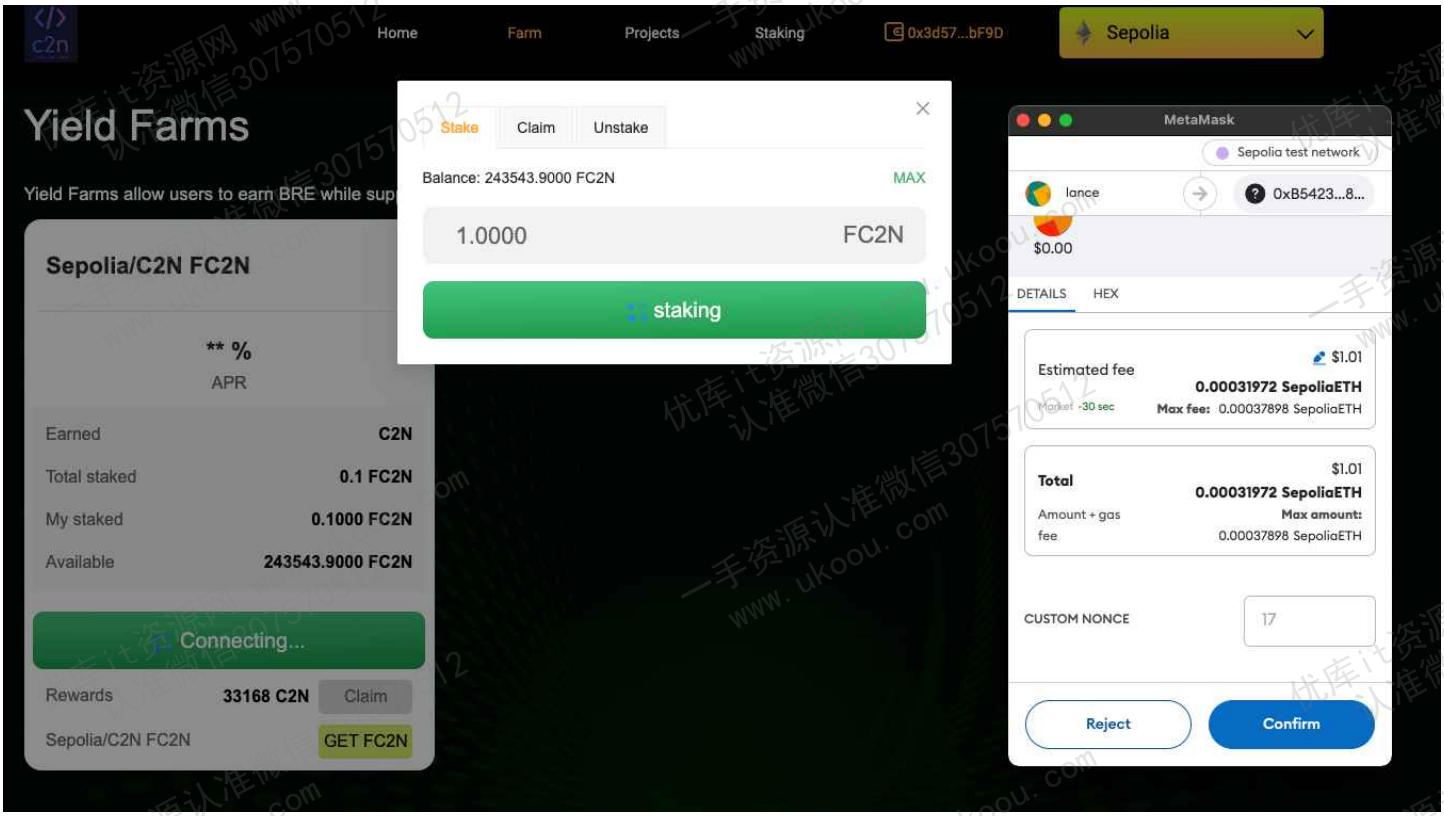
2. 在我们farm界面，我们可以质押fc2n 代币获取c2n, (方便大家操作，我们的测试网fc2n，c2n 是在上一步中领取的同一代币)，在这里我们有三个操作，stake:质押，unstake(witthdraw):撤回质押， 以及 claim:领取奖励;



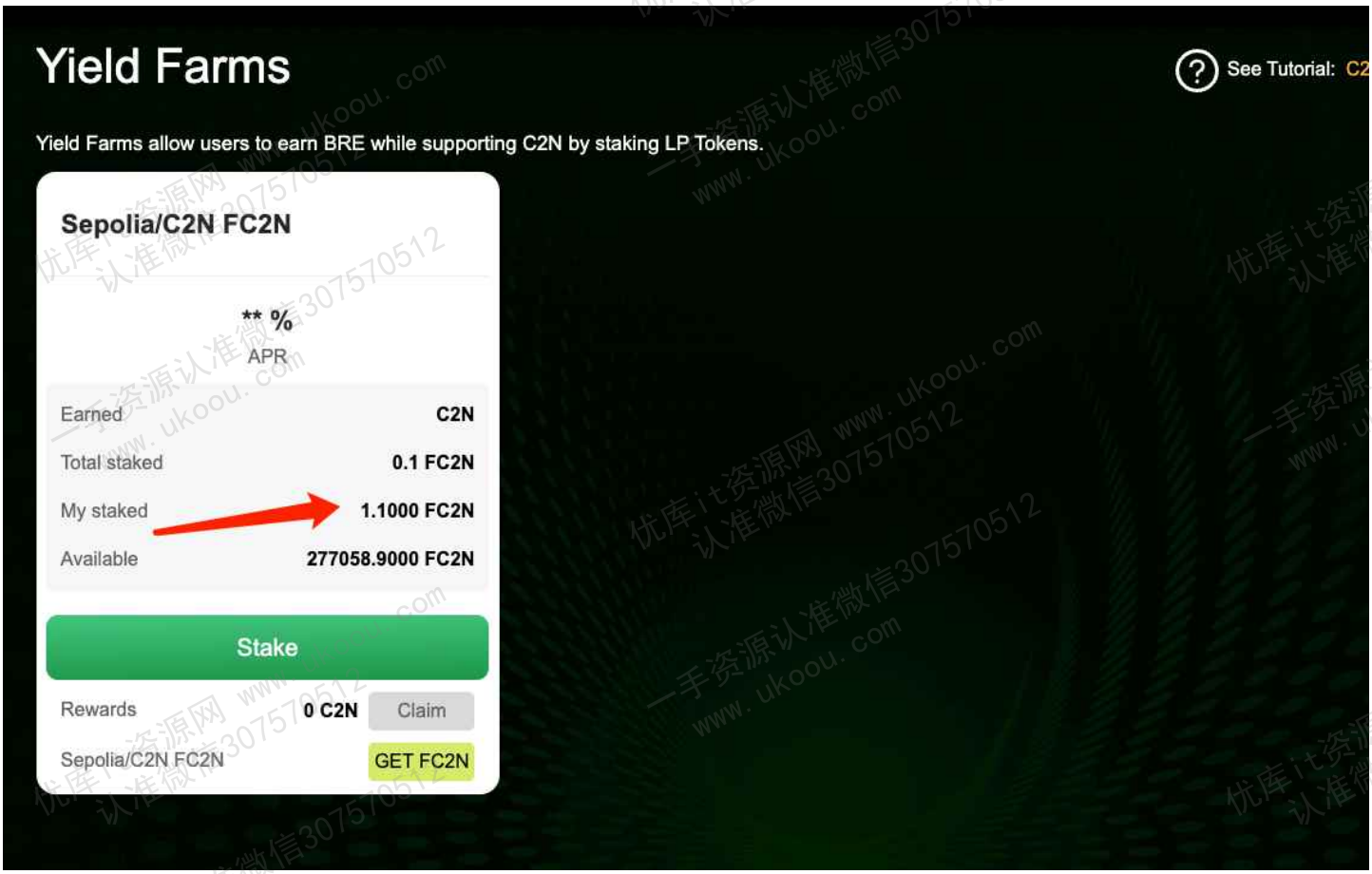
点击stake 或者claim 进入对应的弹窗，切换tab可以进行对应的操作;

3. Stake ，输入要质押的FC2N代币数量，点击stake 会唤起钱包，在钱包中confirm，然后等待交易完成;

一手认准:优库it资源网 www.ukoou.com

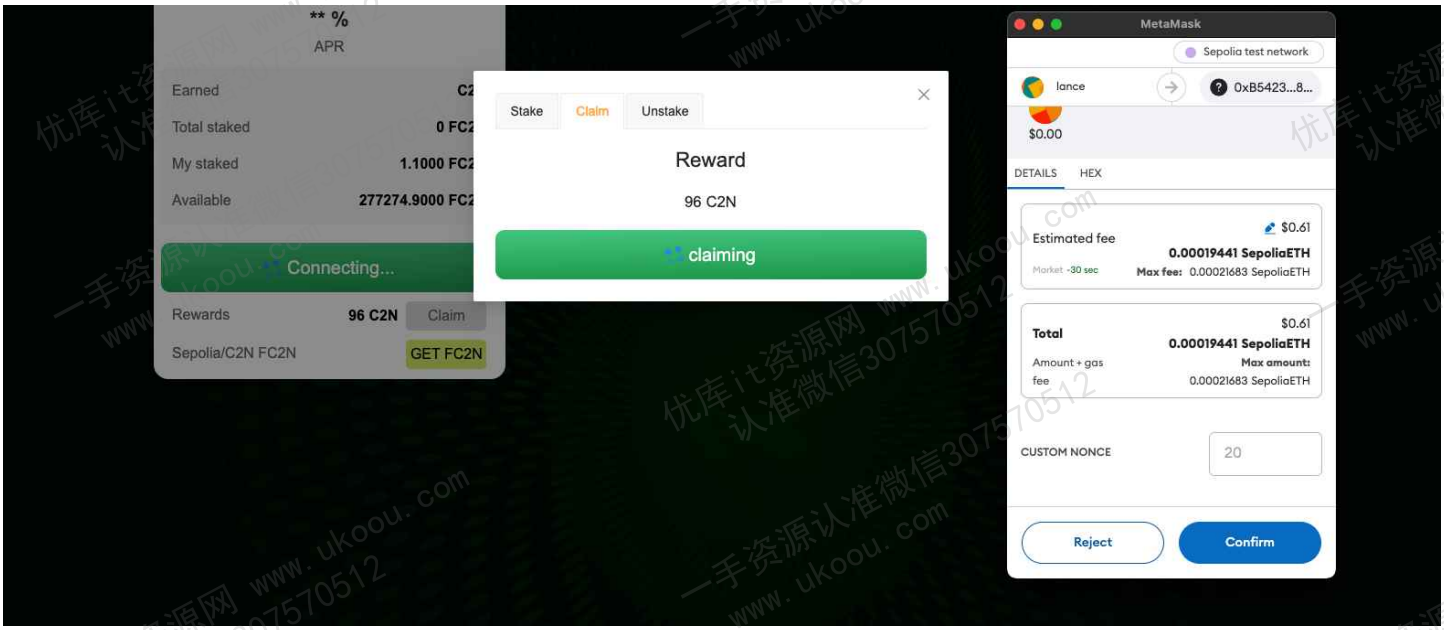


我们新增质押了1FC2N,交易完成之后我们会看到，My staked 从0.1 变成1.1;
Total staked 的更新是一个定时任务，我们需要等待一小段时间之后才能看到更新

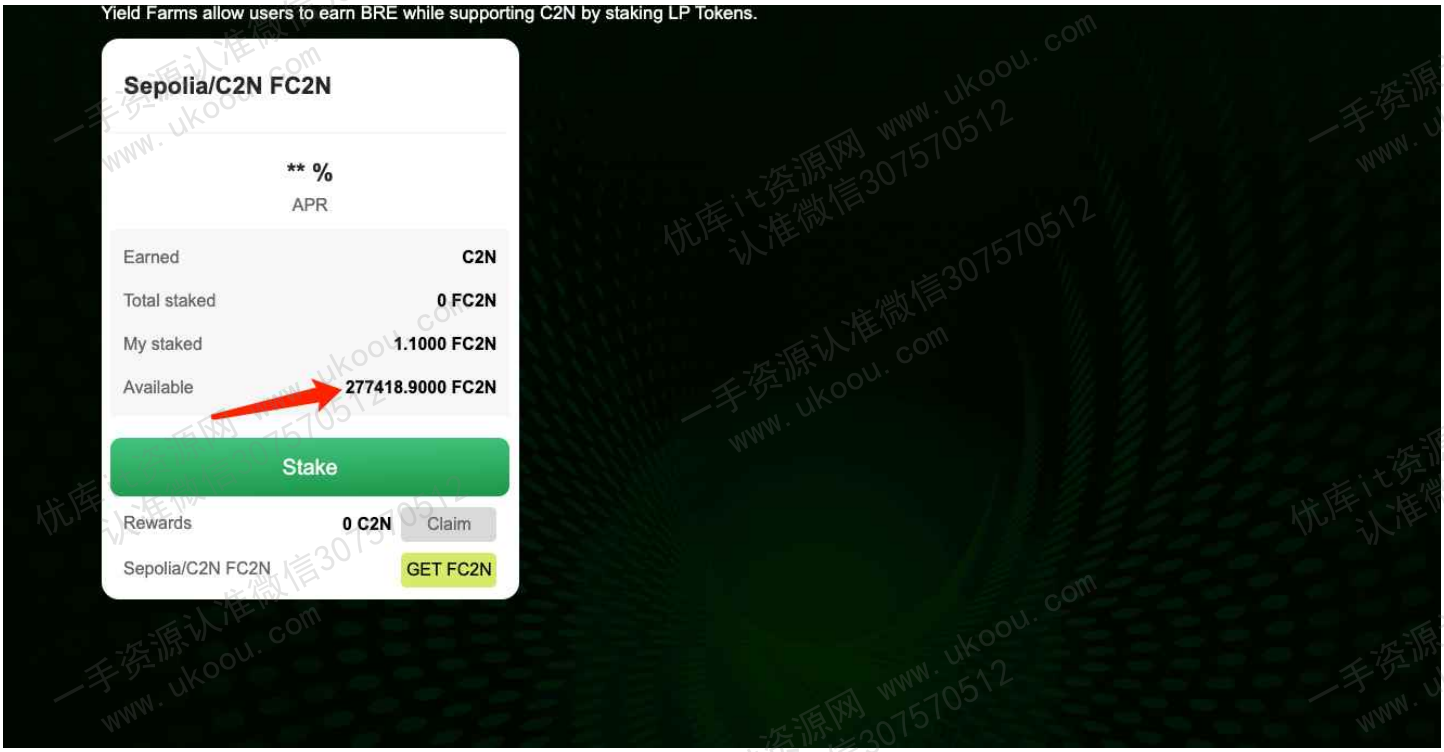


3. Claim 领取质押奖励的C2N,点击claim 并且在钱包确认

一手认准:优库it资源网 www.ukoou.com

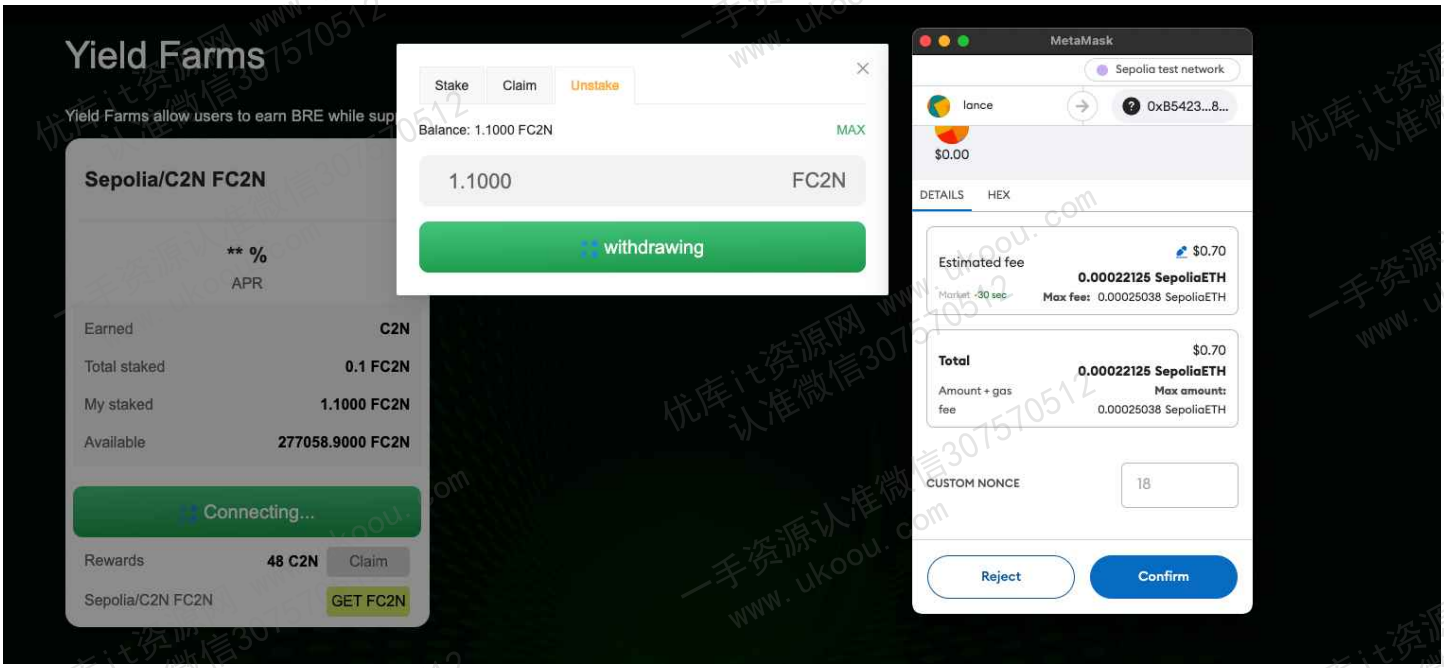


交易完成后我们会看到Available的FC2N数量增加了96，钱包里面C2N的代币数量同样增加了96



4. Unstake(withdraw),输入需要撤回的FC2N 数量(小于已经质押的Balance)，点击withdraw，并且在钱包确认交易

一手认准:优库it资源网 www.ukoou.com



unstake 完成后我们可以看到my staked 的数量变为0

技术文档

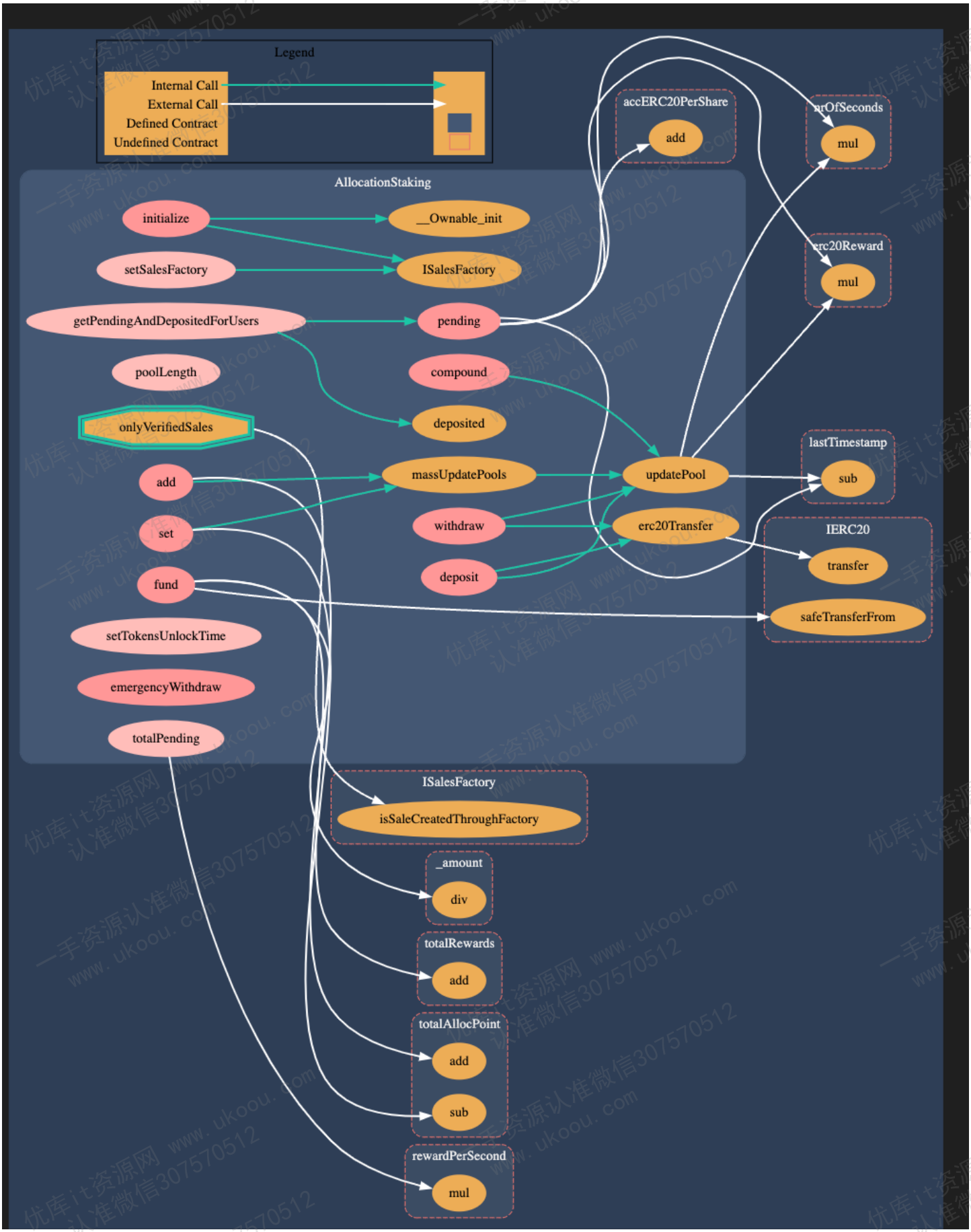
合约开发

项目核心由两个合约组成，以下列出需要实现的函数功能

AllocationStaking.sol

关系调用

一手认准:优库it资源网 www.ukoou.com



函数说明

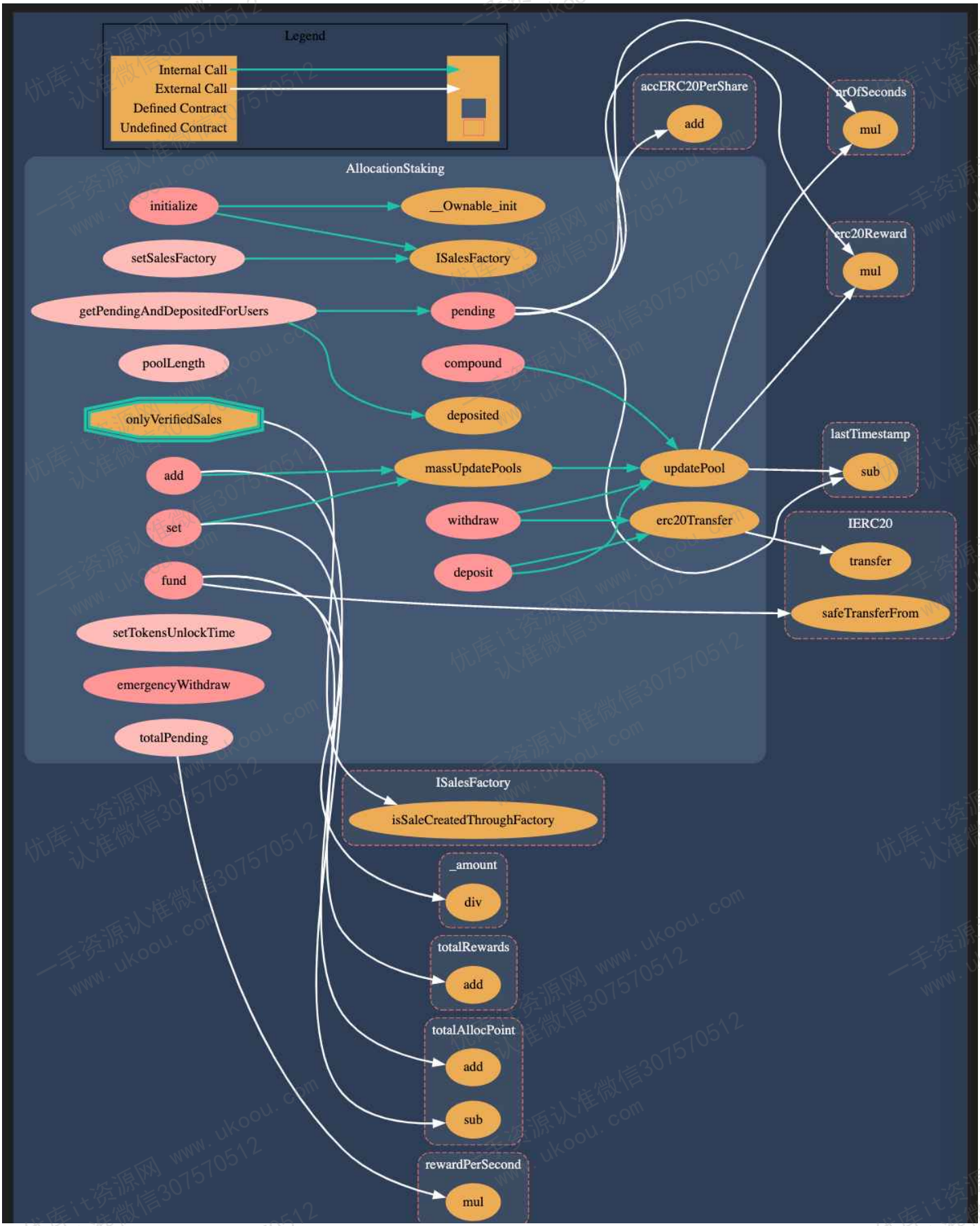
一手认准:优库it资源网 www.ukoou.com

功能	描述
初始化合约	设置 ERC20 代币、每秒奖励、开始时间戳和销售工厂地址。
设置销售工厂地址	更新销售工厂的地址。
获取 LP 池的数量	查看当前所有的 LP 池数量。
为 Farm 提供资金	增加结束区块，为 Farm 提供更多资金。
向池中添加新的 LP 代币	只有合约所有者可以调用此功能。
更新给定池的 ERC20 分配点	只有合约所有者可以调用此功能。
查看用户已存入的 LP	查看特定用户的已存储 LP 代币数量。
查看用户待处理的 ERC20	查询用户待领取的 ERC20 奖励数量。
查看 Farm 尚未支付的总奖励	查看所有池中尚未支付的 ERC20 奖励总额。
更新所有池的奖励变量	为所有 LP 池更新奖励计算变量。
设置用户的代币解锁时间	为用户设置代币的解锁时间。
向 Farm 存入 LP 代币以获取 ERC20 分配	用户可以存入 LP 代币以换取相应的 ERC20 代币奖励。
从 Farm 中提取 LP 代币	用户可以提取他们存入的 LP 代币。
将收益复合到存款中	用户可以选择将收益重新投资到 LP 代币中。
在紧急情况下提取	允许用户在没有考虑奖励的情况下紧急提取他们的 LP 代币。
转移 ERC20 并更新支付所有奖励的所需 ERC20	查询多个用户的存款和收益情况。
获取多个用户的存款和收益	查询多个用户的存款和收益情况。

BrewerySale.sol 功能

关系调用

一手认准:优库it资源网 www.ukoou.com



函数调用

一手认准:优库it资源网 www.ukoou.com

功能	描述
设置销售参数	包括销售代币、销售所有者、代币的 ETH 价格、代币数量、结束时间、解锁时间、最大参与度。
设置注册时间	确定用户注册销售的时间窗口。
设置销售开始时间	确定销售活动开始的具体时间。
存入代币	销售前只能调用一次，用于存入要销售的代币。
参与销售	用户在注册后，按时参与代币购买。
用户领取他们的参与份额	用户可以在销售后根据规则领取自己的代币份额。
一次提取多个解锁的部分	用户可以提取多个阶段解锁的代币份额。
提取销售的收益和剩余部分	用户可以提取他们因参与销售而获得的收益和未售出的代币。
只提取收益	用户可以选择只提取销售活动中产生的收益。
只提取剩余部分	用户可以选择只提取销售结束后剩余的代币部分。
获取用户的参与信息	

技术依赖

OpenZeppelin

OpenZeppelin库提供了一些安全的合约实现，如ERC20、SafeMath等。

前端开发

WIP

后端开发

数据库输入项目信息，配合合约sale显示项目进度和用户购买信息

学员任务

为了帮助学员逐步完成以太坊智能合约C2N Launchpad开发的学习任务，下面我将根据合约代码，拆分出一系列循序渐进的开发任务，并提供详细的文档。这将帮助学员理解并实践如何构建一个基于以太坊的农场合约（Farming contract），用于分配基于用户质押的流动性证明（LP tokens）的ERC20代币奖励。

开发任务拆分

任务一：了解基础合约和库的使用

1. 阅读和理解OpenZeppelin库的文档：熟悉 `IERC20` 、 `SafeERC20` 、 `SafeMath` 、 `Ownable` 这些库的功能和用途。

一手认准:优库it资源网 www.ukoou.com

2. 创建基础智能合约结构：根据给定的智能合约代码，理解合约结构和主要变量的定义。

任务二：用户和池子信息结构定义

1. 定义用户信息结构（`UserInfo`）：
   - 学习如何在Solidity中定义结构体。
   - 添加注释，解释每个字段的含义。
2. 定义池子信息结构（`PoolInfo`）：
   - 理解并定义池子信息，包括LP代币地址、分配点、最后奖励时间戳等。

任务三：合约构造函数和池子管理

1. 编写合约的构造函数：
   - 初始化ERC20代币地址、奖励生成速率和起始时间戳。
2. 实现添加新的LP池子的功能（`add`函数）：
   - 学习权限管理，确保只有合约拥有者可以添加池子。

任务四：奖励机制的实现

1. 编写更新单个池子奖励的函数（`updatePool`）：
   - 理解如何计算每个池子的累计ERC20代币每股份额。
2. 实现用户存入和提取LP代币的功能（`deposit`和`withdraw`函数）：
   - 理解如何更新用户的`amount`和`rewardDebt`。

任务五：紧急提款和奖励分配

1. 实现紧急提款功能（`emergencyWithdraw`函数）：
   - 让用户在紧急情况下提取他们的LP代币，但不获取奖励。
2. 实现ERC20代币转移的内部函数（`erc20Transfer`）：
   - 确保奖励正确支付给用户。

任务六：合约测试和部署

1. 编写测试用例：
   - 使用如Truffle或Hardhat的框架进行合约测试。
2. 部署合约到测试网络（如Ropsten或Rinkeby）：
   - 学习如何在公共测试网络上部署和管理智能合约。

任务七：前端集成和交互

1. 开发一个简单的前端应用：

一手认准:优库it资源网 www.ukoou.com

- 使用Web3.js或Ethers.js与智能合约交互。

2. 实现用户界面：
   - 允许用户通过网页界面存入、提取LP代币，查看待领取奖励。

任务重难点分析

在上述的智能合约代码中，奖励机制的核心功能围绕着分配ERC20代币给在不同流动性提供池（LP pools）中质押LP代币的用户。这个过程涉及多个关键步骤和计算，用以确保每个用户根据其质押的LP代币数量公平地获得ERC20代币奖励。下面将详细解释这个奖励机制的实现过程。

奖励计算原理

1. 用户信息（`UserInfo`）和池子信息（`PoolInfo`）：
   - `UserInfo` 结构存储了用户在特定池子中质押的LP代币数量（`amount`）和奖励债务（`rewardDebt`）。奖励债务表示在最后一次处理后，用户已经计算过但尚未领取的奖励数量。
   - `PoolInfo` 结构包含了该池子的信息，如LP代币地址、分配点（用于计算该池子在总奖励中的比例）、最后一次奖励时间戳、累计每股分配的ERC20代币数（`accERC20PerShare`）等。
2. 累计每股分配的ERC20代币（`accERC20PerShare`）的计算：
   - 当一个池子接收到新的存款、提款或奖励分配请求时，系统首先调用 `updatePool` 函数来更新该池子的奖励变量。
   - 计算从上一次奖励到现在的时间内，该池子应分配的ERC20代币总量。这个总量是基于时间差、池子的分配点和每秒产生的奖励量来计算的。
   - 将计算出的奖励按照池子中总LP代币数量平分，更新 `accERC20PerShare`，确保每股的奖励反映了新加入的奖励。
3. 用户奖励的计算：
   - 当用户调用 `deposit` 或 `withdraw` 函数时，合约首先计算用户在这次操作前的待领取奖励。
   - 待领取奖励是通过将用户质押的LP代币数量乘以池子的 `accERC20PerShare`，然后减去用户的 `rewardDebt` 来计算的。这样可以得到自上次用户更新以来所产生的新奖励。
   - 用户完成操作后，其 `amount`（如果是存款则增加，如果是提款则减少）和 `rewardDebt` 都将更新。新的 `rewardDebt` 是用户更新后的LP代币数量乘以最新的 `accERC20PerShare`。

奖励发放

一手认准:优库it资源网 www.ukoou.com

- 在用户进行提款（`withdraw`）操作时，计算的待领取奖励会通过`erc20Transfer`函数直接发送到用户的地址。
- 这种奖励分配机制确保了用户每次质押状态变更时，都会根据其质押的时间和数量公平地获得相应的ERC20代币奖励。

通过这种设计，智能合约能够高效且公平地管理多个LP池子中的奖励分配，使得用户对质押LP代币和领取奖励的过程感到透明和公正。