区块链茅台溯源DEMO说明文档

1. 需求分析

1.1 目标：

开发一个茅台酒溯源系统的演示版本，以确保产品从生产到销售的每个环节都能被追踪。

1.2 生产销售流程：

- 原料供给：确定茅台酒的原料来源，确保原料的质量与可追溯性。
- 产品生产：监控生产过程，记录关键生产环节和质量检验结果。
- 批发销售：追踪批发环节，包括批发商信息和产品流向。
- 零售：记录零售商信息和产品销售数据。
- 终端消费者：确保消费者能够验证产品的真伪和溯源信息。

1.3 系统设计：

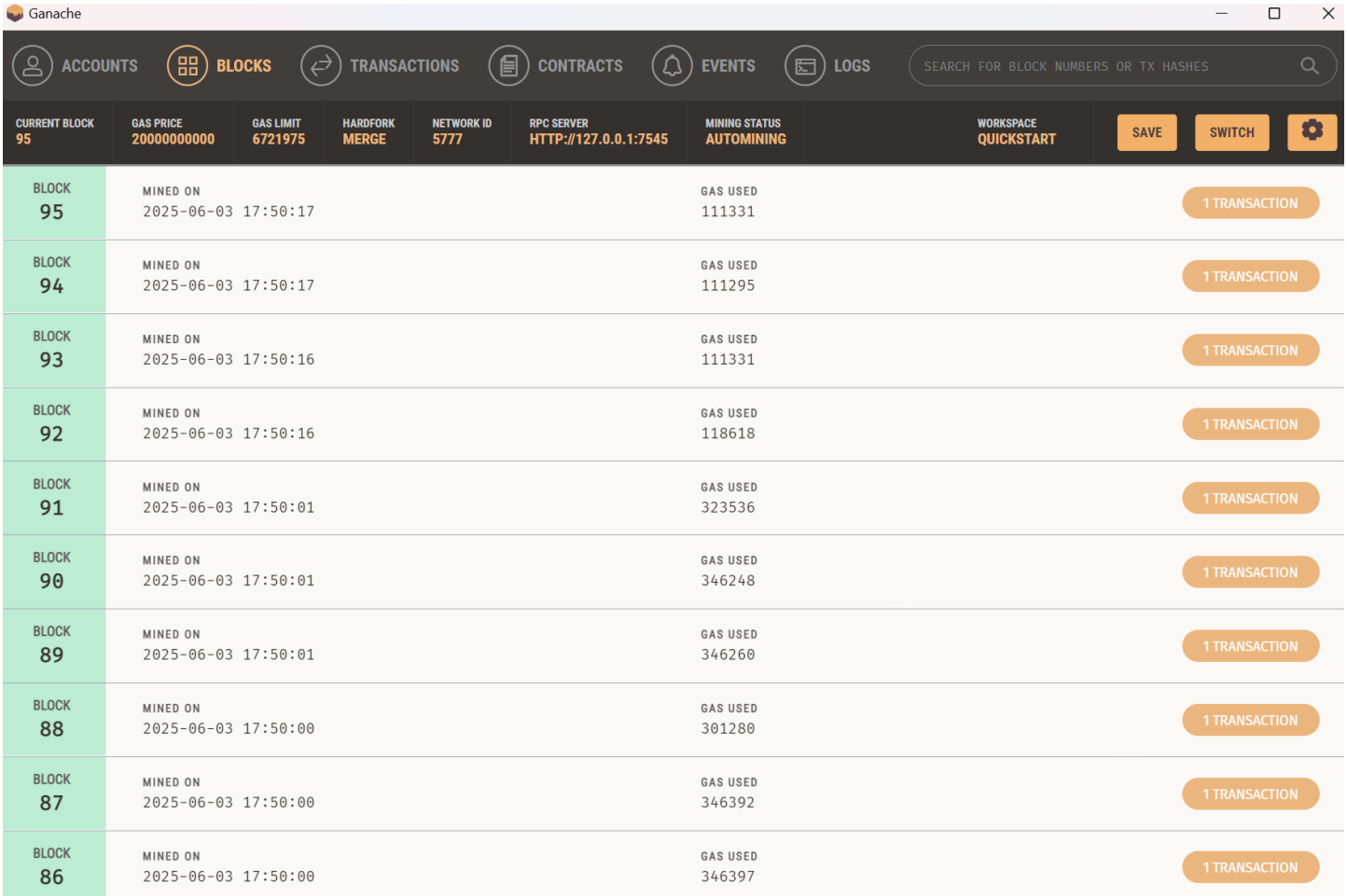
- 在本演示系统中，将模拟茅台酒的生产和销售流程可能包含四个阶段：原料供给、产品生产、批发销售和零售。
- 每个阶段的参与者将被视为一个独立的账户（节点），代表不同的厂家或供应商。
- 账户设置：
  - 原料供给：2个账户
  - 产品生产：2个账户
  - 批发销售：2个账户
  - 零售：4个账户
- 产品追踪：每瓶茅台酒将被赋予一个唯一的全程ID，以便在每个阶段记录关键信息，如时间戳、经手的账户（厂家信息）和验证人员信息。
- 账户信息：每个账户将关联详细的厂家信息，包括：厂家名称、联系方式、类别、店铺照片、地址和描述。

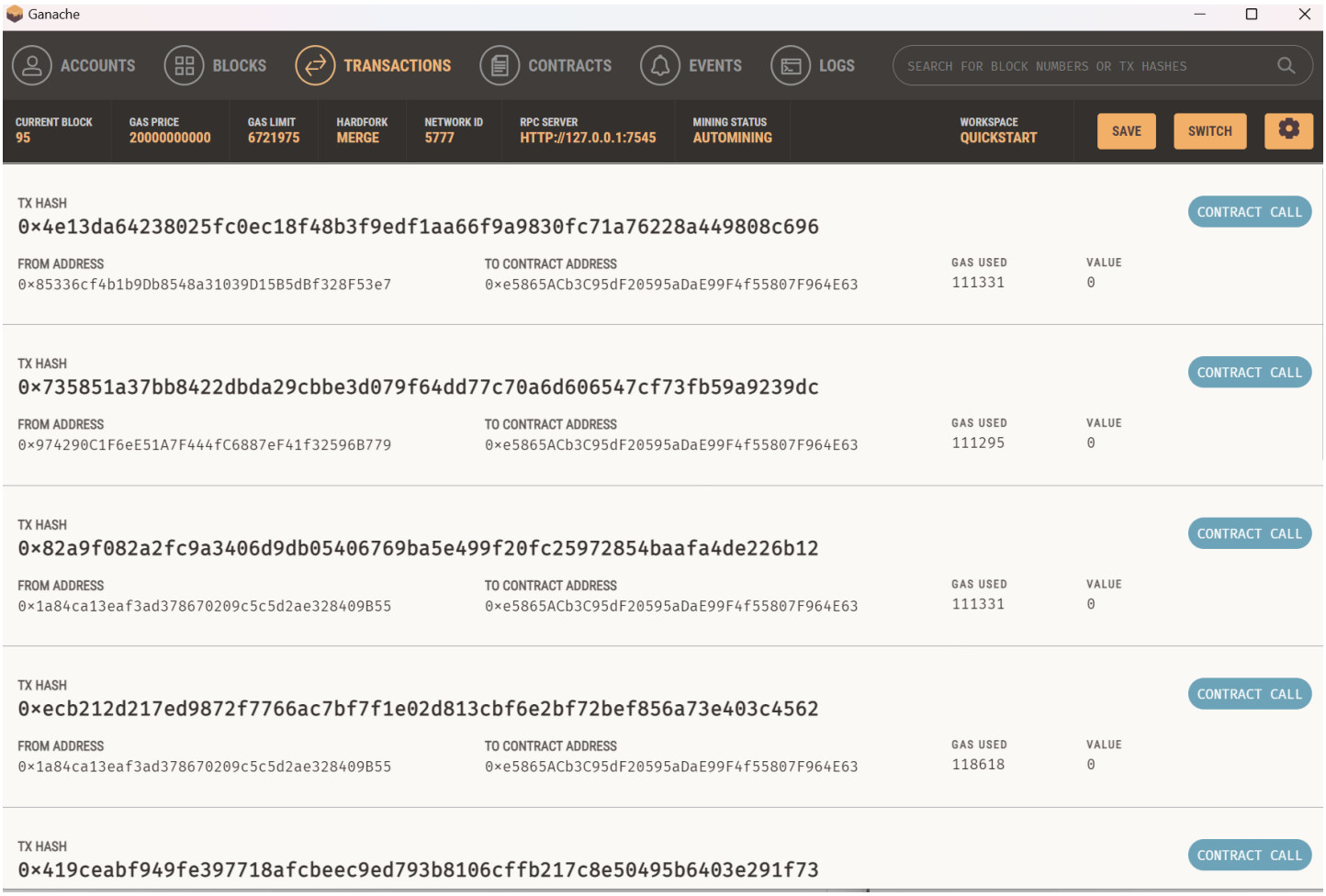
2. 以太坊区块链平台

本项目选择以太坊作为基础平台，并使用以太坊的智能合约功能实现自动化交易和可信的记录保持。

2.1 本地环境模拟：

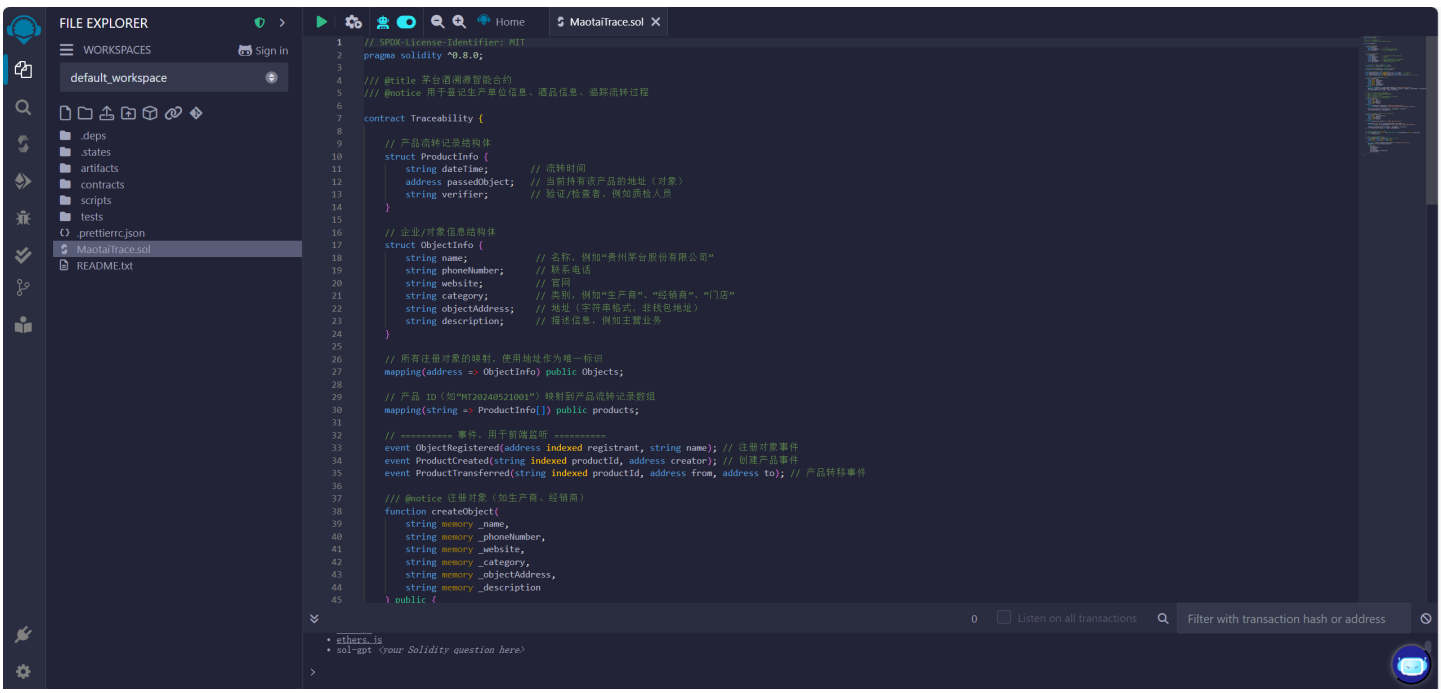
- 本项目使用Ganache作为以太坊区块链，一个用于在本地开发环境中部署合约、开发应用程序和运行测试。
- Ganache提供了一个直观的界面，可以清晰地展示区块和交易信息，使得开发和调试过程更加高效。





2.2 智能合约开发：

本项目使用Remix IDE进行合约代码的编写和测试，这是一个强大的开源工具，支持浏览器在线从编写到部署智能合约的整个生命周期。



3. 智能合约设计

为实现产品从原料供应到零售环节的全流程可信追溯，本系统基于以太坊平台设计并部署了名为 `Traceability` 的智能合约。该合约涵盖对象注册、产品创建、流转记录、查询接口等核心功能。以下详细说明合约的结构、数据模型、主要功能与安全控制。

3.1 合约结构

智能合约命名为 `Traceability`，以太坊 Solidity 语言编写，版本为 `^0.8.0`。合约代码由以下几部分组成：

- 数据结构定义（Structs）
- 状态变量（Mappings）
- 核心业务函数（Functions）
- 事件定义（Events）

3.2 数据模型设计

3.2.1 对象信息 `ObjectInfo`

用于注册系统中的各类实体，如原料供应商、生产商、批发商、零售商等。字段包括：

字段名	类型	描述
name	string	厂家或对象名称
phoneNumber	string	联系电话
website	string	官网链接
category	string	类别（如供应商、零售商等）
objectAddress	string	实体地址（物理地址）
description	string	简介描述

所有对象注册信息由 `Objects` 映射进行管理，键为对象地址（以太坊账户地址），值为其详细信息。

3.2.2 产品流转信息 `ProductInfo`

用于追踪每一个产品在生命周期中的流转记录，字段包括：

字段名	类型	描述
dateTime	string	当前流转记录的时间戳

passedObject	address	当前持有该产品的实体地址
verifier	string	验证人员或质检信息

所有产品追踪记录由 `products` 映射管理，键为产品唯一编号（如 `MT20240521001`），值为该产品的所有流转记录组成的数组。

3.3 核心功能模块

3.3.1 注册对象

代码块

```
function createObject(
        string memory _name,
        string memory _phoneNumber,
        string memory _website,
        string memory _category,
        string memory _objectAddress,
        string memory _description
    ) public
```

- 限制：每个地址只能注册一次
- 输入：包含对象名称、电话、网址、类别、地址和描述信息
- 输出：写入 `Objects` 映射，并触发 `ObjectRegistered` 事件

3.3.2 创建产品

代码块

```
function createProduct(
        string memory _id,
        string memory _dateTime,
        address _passedObject,
        string memory _verifier
    ) public
```

- 限制：
  - 产品 ID 必须唯一
  - 必须由产品的初始持有者发起（即 `msg.sender == _passedObject`）
- 输入：产品编号、创建时间、初始地址、验证人员

- 输出：初始化流转记录数组，写入第一条记录并触发 `ProductCreated` 事件

3.3.3 转移产品

代码块

```
function transferProduct(
        string memory _id,
        string memory _dateTime,
        address _passedObject,
        string memory _verifier
    ) public
```

- 限制：
  - 产品必须存在
  - 只能由当前持有者发起（`msg.sender` 为上一条记录的 `passedObject`）
- 输入：产品编号、转移时间、接收方地址、验证人员
- 输出：追加一条新的流转记录，并触发 `ProductTransferred` 事件

3.3.4 查询接口

- `getProduct`：根据产品编号获取完整的流转记录数组
- `getObject`：根据地址返回注册对象的所有基本信息

3.4 事件机制

为便于前端或监听程序捕捉状态变化，合约定义了如下事件：

事件名	描述
ObjectRegistered	成功注册对象时触发
ProductCreated	创建产品时触发
ProductTransferred	产品发生流转时触发

4. 后端设计

本系统的后端采用 Python 语言，结合 Flask 框架与 Web3.py 库，实现了对以太坊智能合约的调用与封装。整体设计关注模块化、可维护性与对前端的良好支持。

4.1 技术栈与依赖

技术	描述
Flask	提供轻量级 RESTful API 接口
Web3.py	与以太坊智能合约进行交互
Ganache	本地以太坊私链环境，模拟部署和交易
uuid + datetime	用于生成唯一产品 ID 与模拟时间戳
JSON	数据格式标准，前后端交互通用

后端通过读取智能合约的 ABI 和部署地址（以 JSON 文件形式提供），初始化 `Web3` 实例并完成合约实例绑定。

```
contract = w3.eth.contract(address=contract_address, abi=contract_abi)
```

4.2 系统初始化与合约部署对接

在系统启动时，后端通过 `/init` 接口完成如下功能：

- 设置默认账户（用于交易签名）
- 预注册多个溯源对象分类（如原料商、生产商、批发商等）
- 储存对象地址与名称的映射关系，供后续查询与链路展示使用

4.3 API 模块功能详解

4.3.1 对象管理接口

- **POST /objects**
  接收对象名并调用 `createObject()` 向链上注册新角色
- **GET /objects**
  获取所有对象的地址和注册信息，封装为前端可识别的 JSON 格式返回

4.3.2 产品生命周期接口

- **POST /products**
  自动生成一个产品 ID，并模拟从“原料商 → 生产商 → 批发商”的流转过程。每一步调用 `transferProduct()`，数据链被记录在合约中。
- **POST /products/transfer**
  提供任意对象间的产品转移接口，参数包含产品 ID、接收方地址、时间、验证人地址等
- **GET /products/<id>**
  查询产品对应的转移链路，解析为 JSON 格式，结构如下：

4.3.3 辅助与扩展接口

- **GET /products/detail/<id>**
  在基本链路上，进一步返回对象对应的实际名称和注册信息，方便用户界面展示
- **GET /products**
  查询当前系统中所有已记录的产品 ID（如合约端支持列表返回）

5. 前端设计

本系统前端采用 **Gradio** 框架开发，构建了一个简洁直观的 Web 应用界面，实现了对区块链上产品信息和溯源路径的可视化查询。Gradio 作为 Python 生态内的轻量级 UI 框架，能够直接与后端函数绑定，省去了传统前端开发中复杂的 HTML、CSS 和前后端接口对接过程，非常适合原型设计与教学演示类区块链应用。

5.1 架构设计

系统前端基于 Gradio 的 `Blocks`、`Row`、`Column`、`Tab` 等组件构建，整体分为两个主要功能区域：

功能模块	说明
茅台酒产品溯源链展示	用户点击按钮后生成一瓶茅台酒产品，并展示其区块链上的流转路径
溯源地址信息查询	用户输入地址后查询该节点的详细信息（如商家名称、地址等）

5.2 功能实现细节

5.2.1 茅台酒产品溯源链展示

当用户点击“随机生成产品”按钮时，将调用后端 `backend.new_product()` 创建一个新的产品，并通过 `backend.get_product()` 获取该产品的流转链路信息。通过 `generate_list()` 函数对链路进行格式化展示，每个流转节点包括：

- 时间戳（格式化为年月日时分）
- 溯源地址（Object Address）
- 审查人（Verifier）

5.2.2 溯源地址信息查询

用户在文本框中输入任意节点地址，点击“查询溯源信息”按钮后，系统将调用后端 `backend.get_object(address)` 方法获取对应厂家/商家的详细资料，并通过 Markdown 富文本格式展示。

展示内容包括：

- 厂家名称、联系方式、地址、类型、描述
- 节点代表图片（由 URL 加载）

5.3 界面展示

2. 溯源信息查询

2.溯源信息查询

输入左侧的溯源地址可查询该节点的详细信息

溯源地址

0x021F54EdF417f41b6fC224A4330468f3E0636d92

厂家名称：茅台酒体验店
联系方式：0851-90123456
类别：零售销售
地址：贵州省贵阳市观山湖区金阳大道3号
描述：茅台酒体验店，提供品鉴服务

查询溯源信息