

软件设计说明书

**项目名称：** 基于区块链的校园交易平台

学生姓名 卢良强

学 号 202202532

学 院 区块链学院

年 级 2022级

专 业 区块链技术

指导教师 邹林薏

完成日期 2024年11月17日

目 录

[1 引言 1](#_Toc21636)

[1.1 开发背景及目的 1](#_Toc7829)

[1.2 命名规范 1](#_Toc20306)

[1.3 术语和缩写词 1](#_Toc14526)

[1.4 参考资料 1](#_Toc20549)

[1.5 版本信息 2](#_Toc31296)

[2 总体设计 2](#_Toc9931)

[2.1 硬件运行环境 2](#_Toc27940)

[2.2 软件运行环境 2](#_Toc5078)

[2.3 子系统清单 3](#_Toc22905)

[2.4 功能模块清单 3](#_Toc9878)

[3 数据库设计 3](#_Toc17313)

[3.1 数据库中表名列表 3](#_Toc1037)

[3.2 数据库中的表关系 4](#_Toc19990)

[3.3 数据库表的详细清单 5](#_Toc6212)

[4 典型的子系统设计 5](#_Toc8779)

[4.1 会员购书系统设计 5](#_Toc26157)

[4.2 书评管理系统设计 6](#_Toc24979)

[4.3 订单管理系统设计 7](#_Toc4973)

[5 界面设计 7](#_Toc26159)

[5.1 网站设计母板 7](#_Toc30629)

[5.2 主页: /index/ 9](#_Toc7676)

[6 接口设计 9](#_Toc29439)

[6.1 用户接口 9](#_Toc21329)

[6.2 外部接口 10](#_Toc11771)

[6.3 内部接口 10](#_Toc7693)

[7 角色授权设计 10](#_Toc13525)

[8 系统错误处理 10](#_Toc17902)

[8.1 出错信息 10](#_Toc13059)

[8.2 故障预防与补救 11](#_Toc26777)

[8.3 系统维护设计 11](#_Toc30604)

**基于区块链的校园交易平台软件设计说明书**

**1 引言**

## 1.1 开发背景及目的

随着校园二手商品交易需求的增长，传统的二手商品交易模式面临诸如交易安全性不足、信息不透明和流程复杂等问题，导致许多学生和教师难以高效、安全地进行物品交易。为了解决这些问题，本项目旨在开发一个基于区块链技术的校园交易平台。平台将利用区块链的去中心化、不可篡改、透明等特性，结合NFT（非同质化代币）技术，确保交易信息和商品的所有权得到确权，商品交易的每一个步骤透明可信。此外，通过智能合约，系统可以自动化处理商品交易的资金与物品转移，大幅提升交易的效率与安全性。

本项目的目的在于：

1）提供一个基于区块链的安全、高效、透明的校园交易平台；

2）实现商品的数字化确权，确保商品的真实性和唯一性；

3）通过去中心化的身份认证和支付系统，简化注册与支付流程；

4）通过自动化智能合约，提高交易效率，减少纠纷。

## 1.2 命名规范

**1.2.1 数据库命名规范**

表命名： 大写开头的英文单词组合，例如 User, ProductNFT。

字段命名： 大写开头的驼峰命名法，例如 UserID, ProductName。

约束命名：主键：PK\_表名，如 PK\_User。

外键：FK\_表名\_关联表名，如 FK\_Product\_User。

**1.2.2 代码命名规范**

变量命名：

全局变量：全大写，用下划线分隔，如 APP\_VERSION。

私有变量：以“\_”开头，小写开头驼峰命名，如 \_password。

公有变量：大写开头驼峰命名，如 UserName。

类命名：

大写开头驼峰命名，如 UserManager, TransactionProcessor。

方法命名：

大写开头驼峰命名，如 RegisterUser, ProcessOrder。

**1.2.3 文件和项目结构命名规范**

文件夹命名： 全小写，用下划线分隔，如 transaction\_manager。

文件命名： 全小写，用下划线分隔，如 user\_controller.py。

**1.2.4 前端命名规范**

组件命名： 大写开头驼峰命名，如 UserProfile, ProductList。

CSS类名： 小写，用连字符分隔，如 user-card, form-control。

**1.2.5 智能合约命名规范**

合约命名： 大写开头驼峰命名，如 MarketplaceContract。

函数命名： 大写开头驼峰命名，如 MintToken, TransferOwnership。

变量命名：公有变量：驼峰命名，如 tokenID, ownerAddress。

私有变量：以“\_”开头，如 \_balances, \_contractOwner。

## 1.3 术语和缩写词

（1）NFT: Non-Fungible Token，即非同质化代币，代表商品的唯一性，应用于数字资产的确权与交易。

（2）智能合约: 通过区块链自动执行的合约，保证交易的安全性和透明性。

（3）区块链: 一种去中心化的分布式账本技术，记录不可篡改的交易历史，确保交易的透明性和安全性。

（4）IPFS: InterPlanetary File System，分布式文件存储系统，用于存储NFT元数据。

（5）MetaMask: 一个基于浏览器的区块链钱包插件，用户可以通过它进行区块链交易和身份验证。

（6）ERC-721: 以太坊区块链上的标准，用于NFT的实现。

（7）ETH: 以太坊区块链的原生加密货币，用于支付和交易。

## 1.4 参考资料

列出有关资料的作者、标题、编号、发表日期、出版单位或资料来源，可包括：1）项目经核准的计划任务书、合同或上级机关的批文；2）与项目有关的已发表的资料；3）文档中所引用的资料，所采用的软件标准或规范。

[1]吴熙,钱佳琪,屠月海.基于区块链技术的海量数据关联存储系统设计[J].机械设计与制造工程,2024,53(11):95-99.

[2]张军玲,周梦婷.区块链技术赋能数字版权交易机制[J].常州工学院学报,2024,37(05):68-74.

[3]布赫.区块链技术对数字经济中数据交易的影响[J].中国市场,2024,(29):199-202.DOI:10.13939/j.cnki.zgsc.2024.29.050.

## 1.5 版本信息

版本更新信息如表1-1所示。

表 1-1 版本更新表

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 创建人 | 创建日期 | 更新纪要 |
| V1.0 | 卢良强 | 2023-11-17 | 文档创建 |
| ... | ... | ... | ... |

**2 总体设计**

## 2.1 硬件运行环境

服务器的配置与型号如下表2-1所示。

表 2-1 配置与型号

|  |  |
| --- | --- |
| 配置 | 型号 |
| CPU | AMD Ryzen 7 7840HS w/ Radeon 780M Graphics 3.80 GHz 以上 |
| 内存 | 16GB |
| 网络配置 | 100M网卡及以上 |

## 2.2 软件运行环境

服务器的操作系统和型号如下表2-2所示。

表 2-2 配置与型号

|  |  |
| --- | --- |
| 配置 | 型号 |
| 操作系统 | Windows 11 版本:23H2 |
| 数据库 | MySQL-5.7.0 |

## 2.3 子系统清单

子系统清单如表2-3所示。

表 2-3 子系统功能表

|  |  |
| --- | --- |
| 子系统名称 | 子系统功能描述 |
| 用户管理系统 | 实现用户注册、登录、权限管理，支持不同用户角色（学生、教师、管理员）的权限分配与管理。 |
| 区块链网络模块 | 处理商品交易上链、NFT信息存储、交易验证等核心区块链功能，确保交易透明、安全。 |
| 商品发布与交易系统 | 支持商品发布、生成NFT、查看商品和NFT详情、购买商品和NFT、交易记录查询等功能。 |
| 平台部署与运维系统 | 负责服务器环境配置、代码与服务部署、数据库初始化与数据存储监控，支持系统稳定运行。 |
| 平台管理系统 | 提供系统监控、性能预警、日志分析、安全更新、系统版本更新与备份恢复测试功能. |
| 平台功能验收系统 | 涉及功能测试、性能测试、安全测试以及教师用户的功能验收测试，保障系统符合需求并运行稳定。 |

## 2.4 功能模块清单

功能模块清单如表2-4所示。

表 2-4 功能模块表

|  |  |
| --- | --- |
| 名称 | 模块功能描述 |
| 注册账号 | 用户通过填写信息并验证手机号/邮箱，从游客注册为正式用户。 |
| 登录系统 | 已注册用户通过区块链钱包（如MetaMask）或账号密码登录系统。 |
| 找回密码 | 会员通过验证邮箱或手机号，重新设置并获取登录密码。 |
| 查看个人信息 | 用户查看并管理个人信息，包括用户名、邮箱、手机号等信息。 |
| 修改个人信息 | 用户修改个人资料，如用户名、联系方式等，同时支持更换区块链钱包绑定。 |

续表 2-4 功能模块表

|  |  |
| --- | --- |
| 发布商品 | 用户填写商品信息、上传图片及描述，设置价格和库存，并选择是否生成NFT后发布商品。 |
| 生成NFT | 用户将已发布的商品铸造成NFT，生成元数据并上链，确保商品的数字化确权和唯一性。 |
| 查看商品和NFT | 用户浏览商品和NFT列表，支持关键字搜索、分类筛选，以及查看商品/NFT的详细信息页面。 |
| 购买商品和NFT | 用户将商品/NFT添加至购物车，选择支付方式，通过智能合约完成支付与交易确认，并生成订单。 |
| 查询交易记录 | 用户查询自己完成的交易记录，支持按时间或状态筛选，并提供导出交易记录报告和实时交易状态跟踪。 |
| 确认订单并付款 | 会员检查购物车中的商品或NFT订单，确认订单并通过区块链钱包完成付款。 |
| 系统监控与维护 | 管理员监控系统运行状态，包括性能预警、日志收集分析、服务器重启及备份恢复等功能。 |
| 教师功能验收 | 教师用户对平台功能进行测试与反馈，确保系统满足需求并符合实际应用场景。 |
| 平台测试 | 包括功能测试、性能测试和安全测试，生成测试报告并为系统上线提供依据。 |
| 平台部署 | 完成服务器环境配置、服务部署、数据库初始化和域名配置，确保系统能够稳定运行并提供服务。 |
| 存储NFT信息 | 将NFT元数据存储到去中心化存储系统（如IPFS），确保数据的安全性和不可篡改性。 |
| 验证交易 | 区块链验证交易签名，检查交易记录，处理交易冲突并反馈验证结果。 |

**3 数据库设计**

## 3.1 数据库中表名列表

数据库中表名如表3-1所示。

表 3-1 数据库列表

|  |  |  |
| --- | --- | --- |
| 编号 | 表名 | 用途 |
| 1 | User | 用户信息表，存储用户的基本信息及角色权限。 |
| 2 | Product | 商品信息表，记录商品的基本信息及所属用户。 |
| 3 | NFTMetadata | NFT元数据表，记录生成的NFT的详细信息及关联商品 |
| 4 | Transaction | 交易记录表，存储所有的商品和NFT交易记录 |
| 5 | Role | 用户角色表，用于管理用户角色权限信息（学生、教师、管理员等） |
| 6 | ShoppingCart | 购物车表，记录用户添加到购物车中的商品或NFT信息 |
| 7 | Order | 订单表，存储用户生成的订单信息及状态 |
| 8 | Feedback | 用户反馈表，存储用户书评及功能验收反馈等信息 |

## 3.2 数据库中的表关系

数据库中的表关系如图3-1所示。

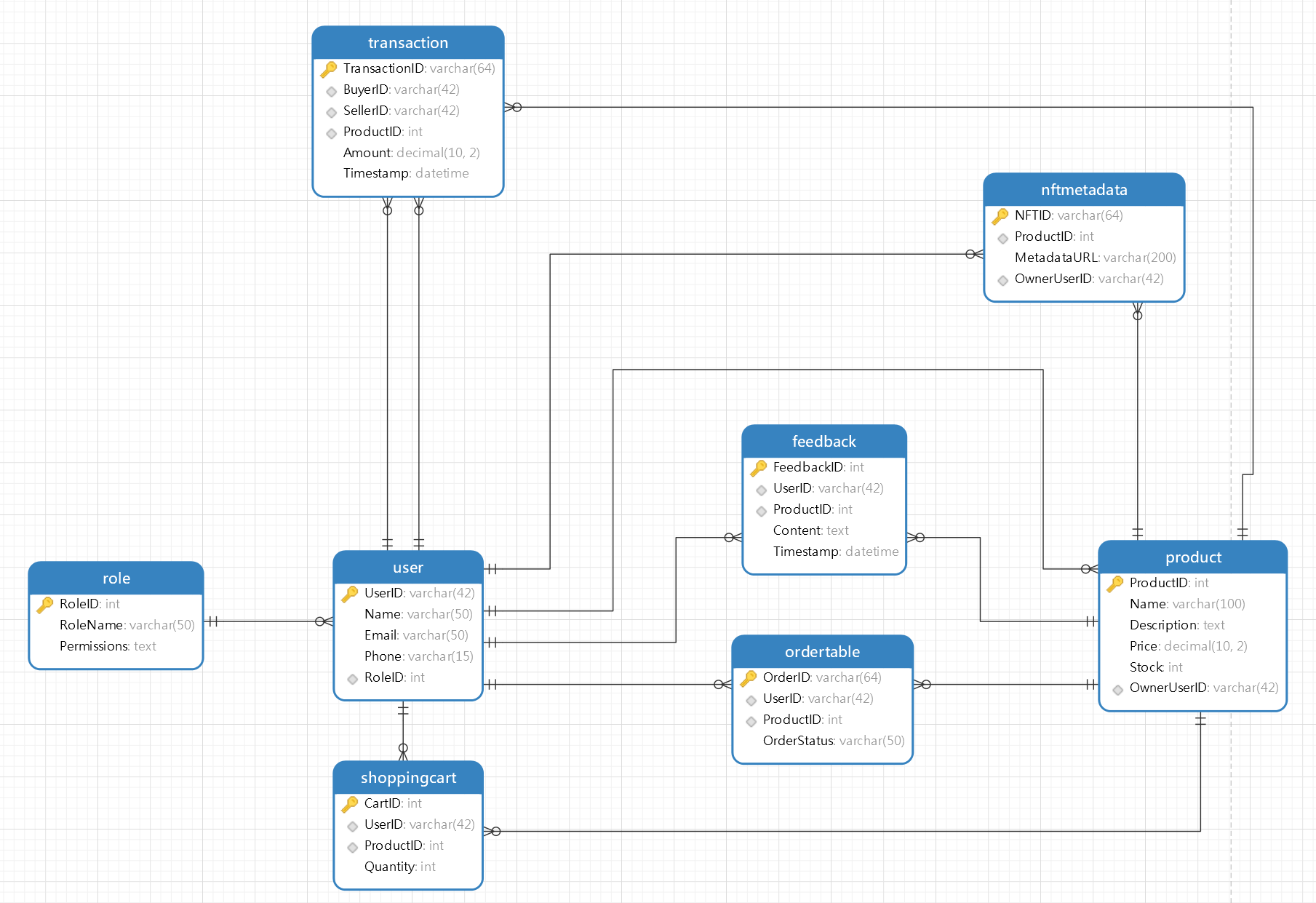


图 3-1 数据库表关系图

## 3.3 数据库表的详细清单

**3.3.1 用户信息表**

用户信息如表3-2所示。

表 3-2 用户信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 英文字段名 | 中文字段名 | 数据类型 | 是否允许为空 | 主键/外键 |
| UserID | 用户ID | varchar(42) | 否 | 主键 |
| Name | 用户名 | varchar(50) | 是 | / |
| Email | 邮箱 | varchar(50) | 是 | / |
| Phone | 手机号 | varchar(15) | 是 | / |
| RoleID | 角色ID | int | 否 | 外键 |

**3.3.2 商品信息表**

商品信息如表3-3所示。

表 3-3 商品信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 英文字段名 | 中文字段名 | 数据类型 | 是否允许为空 | 主键/外键 |
| ProductID | 商品ID | int | 否 | 主键 |
| Name | 商品名称 | varchar(100) | 否 | / |
| Description | 商品描述 | text | 否 | / |
| Price | 商品价格 | decimal(10,2) | 否 | / |
| Stock | 库存数量 | int | 否 | / |
| OwnerUserID | 发布者ID | varchar(42) | 否 | 外键 |

**3.3.3 NFT元数据信息表**

NFT元数据信息如表3-4所示。

表 3-4 NFT元数据信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 英文字段名 | 中文字段名 | 数据类型 | 是否允许为空 | 主键/外键 |
| NFTID | NFT编号 | varchar(64) | 否 | 主键 |
| ProductID | 商品ID | int | 否 | 外键 |
| MetadataURL | 元数据URL | varchar(200) | 否 | / |
| OwnerUserID | 当前所有者ID | varchar(42) | 否 | 外键 |

**3.3.4 图书信息表**

图书信息如表3-5所示。

表 3-5 图书信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 英文字段名 | 中文字段名 | 数据类型 | 是否允许为空 | 主键/外键 |
| TransactionID | 交易ID | varchar(64) | 否 | 主键 |
| BuyerID | 买家ID | varchar(42) | 否 | 外键 |
| SellerID | 卖家ID | varchar(42) | 否 | 外键 |
| ProductID | 商品ID | int | 否 | 外键 |

续表 3-5 图书信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Amount | 交易金额 | decimal(10,2) | 否 | / |
| Timestamp | 时间戳 | datetime | 否 | / |

**3.3.5 图书信息表**

图书信息如表3-6所示。

表 3-6 图书信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 英文字段名 | 中文字段名 | 数据类型 | 是否允许为空 | 主键/外键 |
| RoleID | 角色ID | int | 否 | 主键 |
| RoleName | 角色名称 | varchar(50) | 否 | / |
| Permissions | 权限信息 | text | 否 | / |

**3.3.6 图书信息表**

图书信息如表3-7所示。

表 3-7 图书信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 英文字段名 | 中文字段名 | 数据类型 | 是否允许为空 | 主键/外键 |
| CartID | 购物车ID | int | 否 | 主键 |
| UserID | 用户ID | varchar(42) | 否 | 外键 |
| ProductID | 商品ID | int | 否 | 外键 |
| Quantity | 商品数量 | int | 否 | / |

**3.3.7 图书信息表**

图书信息如表3-8所示。

表 3-8 图书信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 英文字段名 | 中文字段名 | 数据类型 | 是否允许为空 | 主键/外键 |
| OrderID | 订单ID | int | 否 | 主键 |
| UserID | 用户ID | int | 否 | 外键 |
| ProductID | 商品ID | int | 否 | 外键 |
| OrderStatus | 订单状态 | varchar(50) | 否 | / |

**3.3.8 图书信息表**

图书信息如表3-9所示。

表 3-9 图书信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 英文字段名 | 中文字段名 | 数据类型 | 是否允许为空 | 主键/外键 |
| FeedbackID | 反馈ID | int | 否 | 主键 |
| UserID | 用户ID | int | 否 | 外键 |
| ProductID | 商品ID | int | 否 | 外键 |
| Content | 内容 | text | 否 | / |
| Timestamp | 提交时间 | datetime | 否 | / |

**4 典型的子系统设计**

## 4.1 会员购书系统设计

**4.1.1 类设计**

该子系统实现用户注册、登录、权限管理和个人信息维护等功能

**User 类**  
表示用户的基本信息和相关操作

属性：

userID（String）：用户ID（区块链钱包地址）

userName（String）：用户名

password（String）：用户密码（加密存储）

email（String）：用户邮箱

phoneNumber（String）：用户联系电话

registrationDate（Date）：注册日期

status（String）：用户状态（如 "Active", "Inactive"）

方法：

register()：用户注册

login(String userName, String password)：用户登录

updateInfo()：更新用户的个人信息

resetPassword()：重置密码

viewProfile()：查看用户个人信息

**Role 类**  
表示用户角色与权限。

属性：

roleID（String）：角色ID

roleName（String）：角色名称（如 "Student", "Teacher", "Admin"）

permissions（List<String>）：角色权限

方法：

assignRole(String userID, String roleID)：分配角色

removeRole(String userID)：移除角色

viewRolePermissions(String roleID)：查看角色权限

## 4.2 商品管理系统设计

**4.2.1 类设计**

该子系统支持商品的发布、浏览、编辑和删除操作。

**Product 类**  
用于存储商品的基本信息。

属性：

productID（String）：商品ID

productName（String）：商品名称

description（String）：商品描述

price（Decimal）：商品价格

stock（int）：库存数量

ownerID（String）：商品发布者ID

方法：

addProduct(String ownerID, String productInfo)：添加商品

updateProduct(String productID, String newInfo)：更新商品信息

deleteProduct(String productID)：删除商品

viewProductDetails(String productID)：查看商品详情

## 4.3 NFT管理系统设计

**4.3.1 类设计**

该子系统负责商品NFT的生成、元数据存储和查询。

**NFT 类**

用于管理NFT元数据及相关操作。

属性：

nftID（String）：NFT ID

productID（String）：关联商品ID

metadataURL（String）：元数据存储地址

ownerID（String）：NFT所有者ID

方法：

mintNFT(String productID)：生成NFT

storeMetadata(String metadata)：存储NFT元数据

viewNFTDetails(String nftID)：查看NFT详情

**4.4 交易管理系统设计**

**4.4.1 类设计**

该子系统负责商品/NFT的交易，包括订单生成、支付处理和状态更新。

**Transaction 类**

表示交易的详细信息。

属性：

transactionID（String）：交易ID

buyerID（String）：买家ID

sellerID（String）：卖家ID

productID（String）：商品ID

amount（Decimal）：交易金额

status（String）：交易状态（如 "Pending", "Completed"）

timestamp（Date）：交易时间

方法：

createTransaction(String buyerID, String sellerID, String productID, Decimal amount)：创建交易

updateTransactionStatus(String transactionID, String status)：更新交易状态

viewTransactionDetails(String transactionID)：查看交易详情

**4.5 系统监控与维护系统设计**

**4.5.1 类设计**

该子系统负责监控系统运行状态并维护平台的安全性和稳定性。

**SystemMonitor 类**

用于监控系统运行状态和日志收集。

属性：

logID（String）：日志ID

event（String）：事件信息

timestamp（Date）：记录时间

方法：

logEvent(String event)：记录系统事件

generateReport()：生成系统运行报告

**MaintenanceManager 类**

用于平台的备份和维护操作。

属性：

maintenanceID（String）：维护ID

backupSchedule（String）：备份计划

lastBackupDate（Date）：上次备份日期

方法：

backupDatabase()：备份数据库

restoreDatabase(String backupID)：恢复数据库

updateSystem()：系统更新维护

**5 界面设计**

本系统由于使用 Django 框架，而在 Django 中，url 是实现页面跳转和其他用户交互功能的核心，所以本节的功能模块设计介绍以 url 为主线。

## 5.1 网站设计母板

本系统在设计前端页面时使用的“母版”的设计思想，即系统绝大多数页面的布局都继承自该母版页面，也就是说，系统的每个页面的头部和尾部的布局和功能完全相同，只是中间部分会有布局和功能上的变化。如图5-1所示。

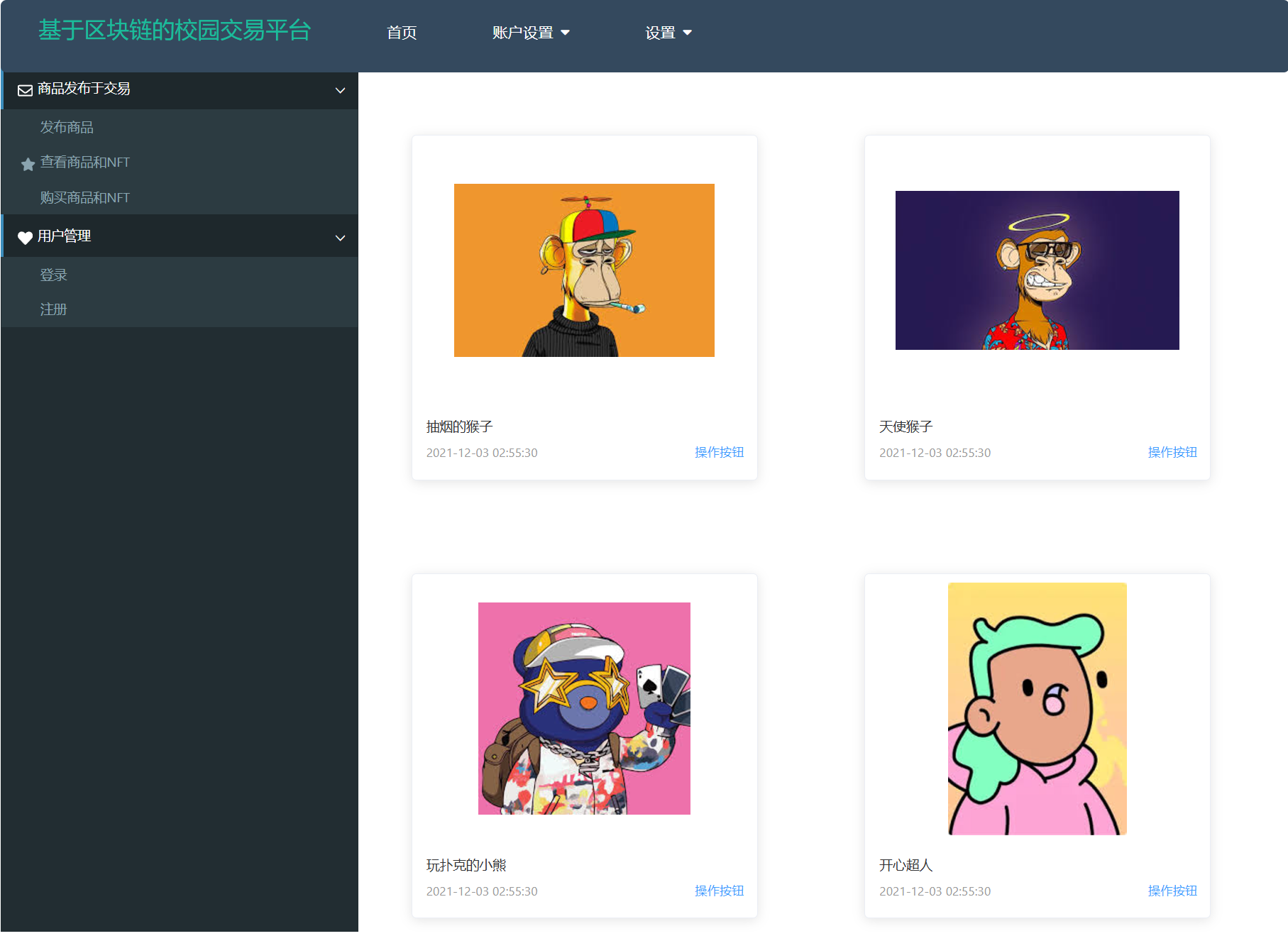


图 5-1 网站设计母版图

因此，本节之后其他功能模块的设计介绍将不包括头部和尾部的设计功能。

**5.1.1 页面组成**

(1)头部：包括平台名称“基于区块链的校园交易平台”，右侧是功能菜单，包含“首页”、“账户设置”、“设置”等选项。根据用户状态，显示登录/注册超链接（游客可见）或注销按钮（已登录用户可见）。功能菜单下方是具体功能的分类导航，包括“商品发布与交易”和“用户管理”两大模块，分别展示相关功能入口。头部支持点击平台Logo，跳转到系统主页

(2)左侧导航栏：按功能分类固定展示，主要包含两部分：“商品发布与交易”模块和“用户管理”模块。用户可以通过展开相应功能，跳转到“发布商品”、“查看商品和NFT”、“购买商品和NFT”等具体页面；同时，用户管理部分提供登录、注册等功能跳转入口。

(3)中部商品展示区：采用网格布局展示商品和NFT信息。每个商品以卡片形式排列，卡片显示商品图片、标题、发布时间，以及操作按钮。操作按钮提供如“查看详情”、“编辑商品”或“删除商品”等功能入口，点击后跳转到对应的功能页面。商品卡片布局动态响应屏幕大小，以适配不同设备

(4)尾部：展示版权信息和一些功能模块跳转链接，例如“帮助中心”、“关于我们”等，便于用户快速导航。

**5.1.2 调用描述**

(1)游客状态下，点击头部的“登录”或“注册”按钮，系统会分别跳转到登录页面 /login/ 或注册页面 /register/。用户完成登录后，头部菜单会显示注销按钮，点击该按钮系统会注销用户，并弹出提示“已注销”，页面恢复为游客状态。

(2)点击导航栏中的“发布商品”跳转到 /publish\_product/ 页面，用户可以上传商品信息进行发布；点击“查看商品和NFT”跳转到 /view\_products/ 页面，展示用户发布的所有商品信息；点击“购买商品和NFT”跳转到 /purchase\_products/ 页面，提供商品购买流程。点击左侧导航栏的任何功能模块，页面都会根据路径跳转到相应的功能页面。

(3)在商品展示区，点击商品卡片的图片或标题，会跳转到 /product\_detail/?id=商品ID 页面，查看商品详细信息；点击“操作按钮”会弹出具体操作选项（如查看详情、编辑、删除），选择后跳转到对应的功能页面。

(4)点击账户设置中的“个人信息”选项，会跳转到 /user\_info/，用户可以查看和修改个人信息；点击“交易记录”选项跳转到 /transaction\_history/，用户可以查看和导出交易记录。尾部功能跳转链接，如“帮助中心”，点击后跳转到 /help\_center/ 页面，提供相关的帮助文档。

## 5.2 主页: /index/

**5.2.1 参数**

无。

**5.2.2 调用背景**

主页作为本系统的主要访问入口，具有以下特点：

(1)用户在浏览器输入网址，按下回车后会默认跳转至该页面。

(2)用户登录验证通过后，系统会将用户重定向至主页。

(3)在系统的任何页面，点击网页左上角的平台Logo或导航条中的“主页”超链接，将跳转至该页面。

**5.2.3 页面组成**

主页采用上下分层的布局结构，具体内容如下：

(1)轮播图区域：主页顶部显示循环播放的轮播图，包含三张图片，循环播放，每张图片的展示时间为10秒。轮播图用于展示平台特色、新品推荐和促销信息等，提升用户的访问体验。

(2)商品推荐区：轮播图下方为推荐商品展示区。商品信息以卡片形式排列，卡片包含商品图片、名称、发布时间以及操作按钮。布局采用响应式表格式设计，卡片数量根据屏幕宽度动态调整。

(3)页面底部：底部区域用于展示平台相关信息，包括版权声明、帮助文档链接及“关于我们”模块。

**5.2.4 调用描述**

(1)轮播图区域：用户可点击轮播图中显示的图片，跳转到相关商品页面或促销活动页面。

(2)推荐商品展示区：商品信息以卡片形式呈现，用户可以：点击商品图片或名称： 跳转到 /product\_page/?productID=?，查看该商品的详细信息。点击操作按钮： 执行与商品相关的具体操作，如“添加到购物车”或“立即购买”。

(3)底部区域：点击“关于我们”跳转到 /about\_us/ 页面。点击“帮助文档”跳转到 /help\_center/ 页面。

**6 接口设计**

## 6.1 用户接口

基于 Vue3 和 Element-Plus 组件库构建，配合 Vue-CLI 工具搭建前端页面。通过精心设计的交互和动态渲染，用户可快速完成商品发布、查看和交易等操作。所有数据均通过 API 调用与后端同步，确保前端的实时更新。

## 6.2 外部接口

采用 RESTful API 标准，结合 SpringBoot 2.4.5 框架与 MySQL Server 8.0.23 数据库进行交互。外部接口主要用于数据存储和处理，例如通过 /api/products 获取商品信息，或通过 /api/transactions 提交交易记录。系统的邮件服务基于 SMTP 协议，利用 163 邮箱服务器完成验证邮件或通知邮件的发送，例如调用 /api/email/sendVerification 发送注册验证码。

## 6.3 内部接口

通过 ORM（对象关系映射）模块实现与数据库的交互，确保数据的高效读写和一致性。页面跳转时，通过 Session 和 GET 请求进行参数传递。例如，用户登录后，其信息会存储在 Session 中，用于维护用户会话状态；商品详情页面则通过 URL 参数（如 /product\_page/?productID=123）获取所需数据。内部接口的设计以数据为核心，结合模块化的架构设计，实现各功能模块之间的高效协作。整个接口设计确保了系统的稳定性和可扩展性，为后续开发提供了良好的基础。

**7 角色授权设计**

本项目的使用角色有三类：游客、普通用户、管理员，角色授权设计如表7-1所示。

表 7-1 角色授权设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 子系统名称 | 管理员 | 会员 | 游客 |
| 个人信息管理系统 | ● | ● | ○ |
| 商品浏览系统 |  | ● | ● |
| 商品购买系统 |  | ● |  |
| 订单管理系统 |  | ● |  |
| 商品管理系统 | ● |  |  |
| 用户反馈管理系统 | ● | ● | ○ |
| ●表示有全部权限，○表示享有权限但并非全部权限。 | | | |

**8 系统错误处理**

## 8.1 出错信息

(1)用户输入检查：针对游客和会员输入的内容，系统进行严格的有效性和安全性检查，避免无效输入和数据注入攻击。提示用户明确的错误信息，例如“请输入有效的邮箱地址”或“密码必须包含至少8个字符”。

(2)异常捕获：系统在程序运行时捕获所有可能的异常，并以统一的格式向用户提供错误提示信息。例如，当数据库连接异常时，系统显示“服务器连接失败，请稍后重试”。

(3)权限错误处理：当用户尝试访问超出其权限范围的功能或页面时，系统自动重定向到权限错误提示页面，显示“您无权访问此页面”。

## 8.2 故障预防与补救

(1)权限控制：系统通过统一的权限控制机制，确保所有模块操作均经过权限验证，防止未授权访问。敏感数据（如密码、交易记录）采用加密存储，确保即使数据库被攻击，也无法轻易窃取信息。

(2)数据备份：系统定期执行全量备份和增量备份，确保在发生数据丢失或损坏时能够迅速恢复。备份策略包括：每日增量备份：仅备份当天新增或修改的数据。每周全量备份：备份所有数据的完整副本。

## 8.3 系统维护设计

(1)模块化和分层设计：系统采用模块化开发方式，将功能模块化、分层化，增强代码的内聚性和降低模块间耦合度。各模块独立运行，便于开发和维护。例如，用户管理模块独立于商品交易模块。

(2)面向对象思想：模块划分和设计均遵循面向对象的原则，确保数据与操作的封装性。例如，商品类包含商品的属性和操作方法，订单类包含订单的属性和管理逻辑。

(3)编码规范与注释：代码编写严格遵守命名规范和结构化风格，所有函数、类和变量名称均具有明确的语义。在复杂的逻辑实现部分添加注释，便于后续开发人员理解和维护。通过以上设计，系统能有效减少错误发生的概率，提升故障恢复能力，并确保后期维护工作高效进。