江西軟件職業技術大學
JIANGXI UNIVERSITY OF SOFTWARE PROFESSIONAL TECHNOLOGY

软件设计说明书

项目名称： 基于以太坊智能合约+NestJS微服务的游戏社区与任务市场系统

学生姓名 闵俊涛

学　　号 202202083

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12日

目 录

基于以太坊智能合约+NestJS微服务的游戏社区与任务市场系统软件设计说明书

1 引言

1.1 开发背景及目的

随着区块链技术的不断发展，游戏社区与任务市场的需求日益增长。传统游戏社区存在诸多问题，如中心化管理导致的信任问题、交易不透明等。为解决这些问题，本项目构建基于以太坊智能合约和NestJS微服务架构的游戏社区与任务市场系统，实现去中心化的任务管理，提高交易透明度和安全性，同时利用微服务架构提升系统可扩展性与维护性，为用户提供优质的游戏社区体验。

1.2 命名规范

(1) 数据库表名：采用大写字母开头的驼峰命名法，如 “UserTable” 表示用户表，“TaskList” 表示任务列表表，表名应简洁明了，准确反映表的内容。

(2) 数据库字段名：全部使用小写字母，单词之间用下划线 “_” 分隔，如 “user_name” 表示用户名，“task_description” 表示任务描述，字段名应具有可读性，易于理解。

(3) 变量名

局部变量：采用小写字母开头的驼峰命名法，如 “userData” 表示用户数据，“taskStatus” 表示任务状态，变量名应尽量简短且表意明确。

全局变量：在变量名前加上 “g_” 前缀，采用大写字母开头的驼峰命名法，如 “g_SystemConfig” 表示系统配置全局变量，以便与局部变量区分。

(4) 函数名：使用动词开头的大写字母开头的驼峰命名法，如 “GetUserInfo ()” 表示获取用户信息函数，“UpdateTaskStatus ()” 表示更新任务状态函数，函数名应准确描述函数的功能。

(5) 类名：采用大写字母开头的驼峰命名法，如 “UserManager” 表示用户管理类，“TaskExecutor” 表示任务执行类，类名应体现类的职责和功能。

(6) 接口名：在接口名前加上大写字母 “I” 前缀，采用大写字母开头的驼峰

命名法，如 “IUserService” 表示用户服务接口，“ITaskRepository” 表示任务数据访问接口，以明确表示其为接口类型。

(7) 常量名：全部使用大写字母，单词之间用下划线 “_” 分隔，如 “MAX_TASK_COUNT” 表示最大任务数量，“DEFAULT_PAGE_SIZE” 表示默认页面大小，常量名应具有明确的含义和固定的值。

1.3 术语和缩写词

(1) 以太坊智能合约：基于以太坊区块链的智能合约技术，实现去中心化交易与业务逻辑。

(2) NestJS：用于构建高效可扩展Node.js服务器端应用的框架。

(3) gRPC：高性能开源远程过程调用框架。

(4) Kafka：高吞吐量分布式发布订阅消息系统。

(5) TDD：测试驱动开发方法。

(6) CI/CD：持续集成/持续部署实践。

(7) 社区积分：系统内虚拟货币，通过任务、活动获取，用于交易或兑换。

(8) 任务发布者：在系统中发布任务的用户。

(9) 任务领取者：领取并完成任务的用户。

1.4 参考资料

[1] 以太坊基金会.以太坊官方文档[OL].https://ethereum.org/en/developers/docs/,2024

[2] NestJS团队.NestJS官方文档[OL].https://docs.nestjs.com/,2024

[3] gRPC团队.gRPC官方文档[OL].https://grpc.io/docs/,2024

[4] Apache Kafka团队.Kafka官方文档[OL].https://kafka.apache.org/documentation/,2024

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	闵俊涛	2024-11-12	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	Intel Xeon E5-2680 2.50GHz 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	Linux（Ubuntu 22.4.0 以上）
数据库	MySQL-8.0.15

2.3 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	用户从游客注册为会员
登录系统	已注册用户登录系统
找回密码	会员丢失密码后，通过审核重新获取
查看个人信息	会员查看个人信息
修改个人信息	会员登录后对资料进行管理
发布帖子	玩家进入社区发布帖子，填写标题、内容等信息
浏览帖子	进入社区浏览其他人的帖子
点赞评论	对喜欢的帖子点赞、评论
创建社区	认证玩家创建游戏社区，填写名称、主题等信息
创建社区积分	社区积攒一定人气后可以创建积分
任务发布	任务发布者填写任务详情、定价并提交发布申请
任务浏览	任务浏览、查询
任务执行	玩家选择任务，按要求执行并提交完成申请
积分交易	玩家选择交易对象与方式，输入交易信息提交申请

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	存储用户的基本信息及认证状态等

2	Task	记录任务相关详情

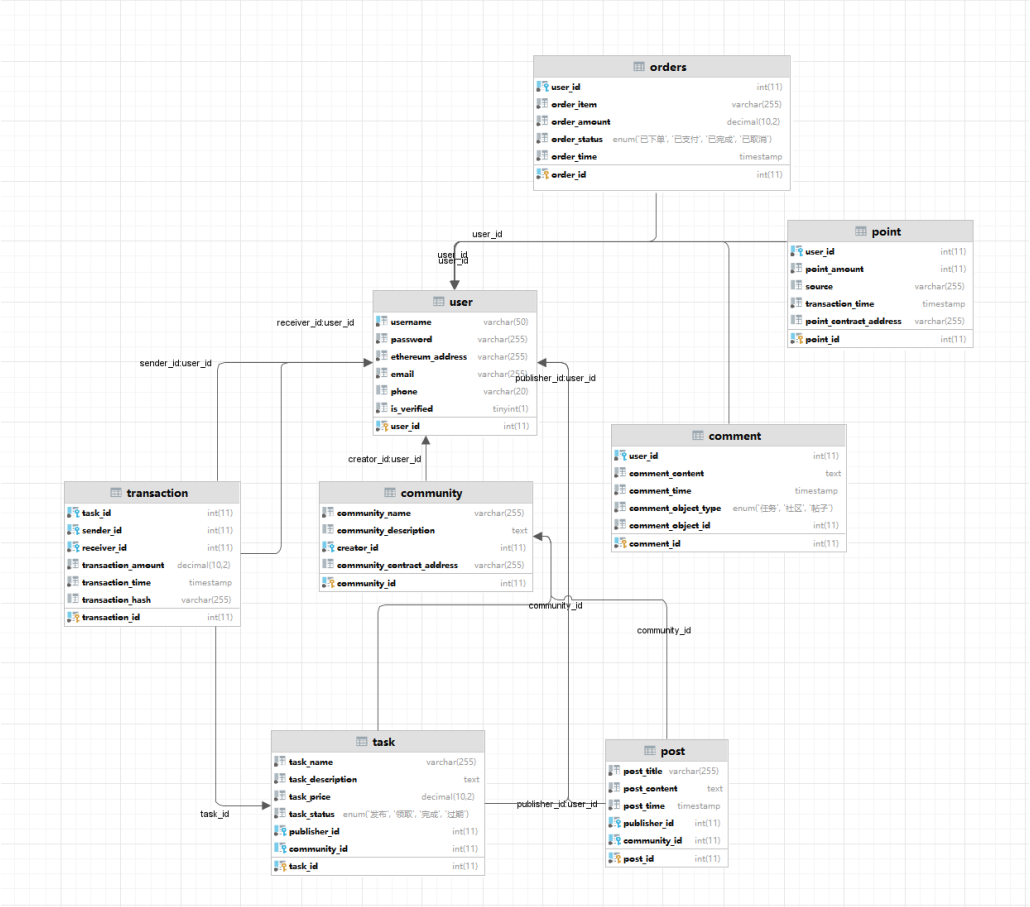
续表 3-1 数据库列表

3	Community	存放社区的基本信息
4	Post	存储帖子的具体内容
5	Transaction	存储任务相关的交易记录
6	Point	记录用户积分的获取
7	Comment	评论内容及相关信息
8	Order	产生的订单详情

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 User表

User表如表3-2所示。

表 3-2 User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
user_id	用户ID	INT	否	主键
username	用户名	VARCHAR(50)	否	唯一约束
password	密码	VARCHAR(255)	否	/
ethereum_address	以太坊地址	VARCHAR(255)	否	/
email	邮箱	VARCHAR(255)	否	/
phone	手机号码	VARCHAR(20)	是	/
is_verified	是否认证	TINYINT(1)	否	/

3.3.2 Task表

Task表如表3-3所示。

表 3-3 Task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
task_id	任务ID	INT	否	主键
task_name	任务名称	VARCHAR(255)	否	/
task_description	任务描述	TEXT	否	/
task_price	任务价格	DECIMAL(10, 2)	否	/
task_status	任务状态	ENUM('发布', '领取', '完成', '过期')	否	/

3.3.3 Community表

Community表如表3-4所示。

表 3-4 Community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
community_id	社区ID	INT	否	主键
community_name	社区名称	VARCHAR(255)	否	/
community_description	社区描述	TEXT	是	/
creator_id	创建者ID	INT	否	外键，关联User表的user_id
community_contract_address	社区合约地址	VARCHAR(255)	是	/

3.3.4 Post表

Post表如表3-5所示。

表 3-5 Post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
post_id	帖子ID	INT	否	主键
post_title	帖子标题	VARCHAR(255)	否	/
post_content	帖子内容	TEXT	否	/
post_time	发布时间	TIMESTAMP	否	/
publisher_id	发布者ID	INT	否	外键，关联User表的user_id

3.3.5 Point表

Point表如表3-6所示。

表 3-6 Point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
point_id	积分记录ID	INT	否	主键
user_id	用户ID	INT	否	外键，关联User表的user_id
point_amount	积分数量	INT	否	/
source	积分来源	VARCHAR(255)	是	/
transaction_time	交易时间	TIMESTAMP	否	/

3.3.6 Transaction表

Transaction表如表3-7所示。

表 3-7 Transaction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
transaction_id	交易ID	INT	否	主键
task_id	任务ID	INT	否	外键，关联Task表的task_id
sender_id	发送方ID	INT	否	外键，关联User表的user_id
receiver_id	接收方ID	INT	否	外键，关联User表的user_id
transaction_amount	交易金额	DECIMAL(10, 2)	否	/

3.3.7 Comment表

Comment表如表3-8所示。

表 3-8 Comment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
comment_id	评论ID	INT	否	主键
				外键，关联User表的
user_id	评论者ID	INT	否	user_id
comment_content	评论内容	TEXT	否	/
comment_time	评论时间	TIMESTAMP	否	/
comment_object_type	评论对象类型	ENUM('任务','社区','帖子')	否	/

3.3.8 Orders表

Orders表如表3-9所示。

表 3-9 Orders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
order_id	订单ID	INT	否	主键
				外键，关联User表的
user_id	用户ID	INT	否	user_id
order_item	订单物品	VARCHAR(255)	否	/
order_amount	订单金额	DECIMAL(10, 2)	否	/
order_status	订单状态	ENUM('已下单','已支付','已完成','已取消')	否	/

4 界面设计

本系统由于使用 Vue框架，而在 Vue 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。

4.1 网站设计母板

4.1.1 页面布局

头部

顶部区域：展示欢迎语，对游客显示登录 / 注册入口，对已登录用户显示用户名及注销链接。欢迎语亲切友好，登录 / 注册链接醒目易操作。

中部区域：网站 Logo 位于左侧，点击可返回主页；中部为全局搜索框，支持模糊搜索任务、社区、帖子等内容；右侧依次为个人账户按钮和查看购物车按钮，按钮样式简洁直观，方便用户快速访问个人信息和购物车。

底部导航栏：涵盖多个导航按钮，包括 “首页”“任务市场”“游戏社区”“积分商城”“个人中心” 等核心模块，部分按钮带有下拉菜单，如 “个人中心” 下拉包含 “个人信息”“我的订单”“设置” 等选项，各按钮功能明确，布局合理。

主体内容区：此区域为各页面的核心展示部分，根据不同功能模块动态加载相应组件，如任务列表、社区帖子列表、积分交易记录等，内容展示清晰有序，方便用户浏览与操作。

尾部：包含版权声明、使用条款、隐私政策、联系我们、帮助中心等文字链接，链接样式简洁统一，提供用户获取更多信息和寻求帮助的途径。

4.1.2 交互逻辑

登录与注册流程

游客点击头部登录 / 注册链接，页面平滑跳转到 /login/ 或 /register/ 页面，登录页面直接使用用户账户签名登录，注册页面引导用户填写详细信息并进行验证。

注册成功后自动登录并跳转至首页，登录失败提示错误原因，用户可根据

提示重新登录或找回密码。

页面导航

点击头部 Logo 或导航栏 “首页” 按钮，始终跳转至 /index/ 页面，确保用户无论身处何页面都能快速返回主页。

点击 “任务市场”“游戏社区”“积分商城” 等按钮，分别跳转到对应功能模块页面，如 /task-market/、/game-community/、/point-mall/，页面切换流畅，过渡效果自然。

点击 “个人中心” 下拉菜单中的选项，如 “个人信息” 跳转到 /user-info/ 页面，“我的订单” 跳转到 /my-orders/ 页面，方便用户管理个人资料和订单。

功能模块跳转

点击搜索框，自动获取焦点，用户输入关键词后实时显示搜索结果，可通过回车键或点击搜索按钮进行搜索，搜索结果展示清晰，可快速定位目标内容。

点击个人账户按钮，跳转到个人中心页面，展示用户详细信息、积分余额、历史记录等，用户可在此进行信息修改、密码重置等操作，操作流程简单易懂。

点击查看购物车按钮，跳转到 /shopping-cart/ 页面，展示购物车中商品列表、数量、总价等信息，用户可在此进行商品管理、结算等操作，购物车功能操作便捷，支持全选、单选、数量调整、删除商品等操作。

4.2 主页：/index/

4.2.1 参数

本页面无特定参数传递，主要展示系统的核心信息和热门内容，以吸引用户并引导其进行进一步操作。

4.2.2 调用背景

系统默认访问入口，用户在浏览器输入网址后直接加载此页面，展示系统的整体形象和主要功能模块，让用户快速了解系统的用途和特色。

用户登录成功后，作为登录后的默认跳转页面，确保用户能够及时获取系统的最新信息和推荐内容，如热门任务、社区活动等。

在系统内部的任何页面，用户点击左上角的网站 Logo 或导航栏中的 “首页” 按钮，均可返回至该页面，方便用户在不同功能模块间快速切换和导航。

4.2.3 页面结构

轮播图区域

位于页面顶部，占据较大视觉空间，自动循环播放三张高质量图片，每张图片展示时间约为 10 秒，切换效果流畅自然，吸引用户注意力。

图片内容包括系统特色功能介绍、热门任务推荐、社区精彩活动预览、限时促销信息等，图片设计精美，文字说明简洁明了，引导用户点击查看详情。

促销或推荐商品展示区

以卡片式布局展示商品信息，每行展示三到四张卡片，卡片之间间距适中，布局整齐美观。

每张卡片包含商品图片、名称、简要描述、价格、评分等关键信息，图片清晰展示商品外观，名称突出显示，描述简洁准确，价格醒目，评分以星级形式展示，方便用户快速了解商品价值。

书店信息与 “关于我们” 区域

位于页面底部，展示书店的基本信息，如书店名称、成立时间、经营理念等，增强用户对书店的了解和信任。

“关于我们” 部分提供链接，用户点击可查看更详细的公司介绍、团队成员、联系方式等信息，提升用户对系统背后团队的认知度。

4.2.4 交互操作

轮播图交互

图片自动切换过程中，用户可随时点击暂停按钮停止轮播，点击图片或左右箭头按钮可手动切换图片，查看感兴趣的内容。

轮播图上可设置热点区域，点击热点区域可直接跳转到相应的任务详情、社区活动页面或促销活动页面，方便用户快速参与感兴趣的活动。

商品卡片交互

鼠标悬停在商品卡片上时，卡片背景颜色变淡或出现阴影效果，突出显示商品，同时显示 “查看详情” 和 “加入购物车” 按钮，按钮样式明显，易于操作。

点击 “查看详情” 按钮，页面跳转到 /product-page/?productId=xxx

（其中 xxx 为商品的唯一标识符），展示商品的详细信息，包括详细描述、参数规格、用户评价等，用户可在此页面进行购买、收藏等操作。

点击 “加入购物车” 按钮，商品以动画效果平滑添加到购物车中，同时购物车图标上显示当前购物车中的商品数量，实时更新，方便用户查看。用户可在购物车页面进一步管理商品，如调整数量、删除商品或进行结算。

5 接口设计

5.1 用户接口

5.1.1 前端技术选型与架构

本项目前端运用 Vue3 框架，并结合 Element-plus 组件库进行开发。通过 Vue-CLI4.5.0 脚手架工具搭建项目架构，实现界面路由管理。在设计上，遵循简洁、美观、易用原则，注重用户体验。

5.1.2 页面设计与交互

登录与注册

登录页面提供多种登录方式，如用户名 / 邮箱 / 手机号码 + 密码，方便用户操作。注册页面支持邮箱和手机号码注册，输入框旁实时显示格式要求及验证提示，确保信息准确。注册成功后，自动跳转至首页并显示欢迎信息。

任务管理

任务列表以直观卡片形式展示，包含任务名称、发布者、奖励、状态等关键信息，支持按状态筛选排序。点击任务卡片进入详情页，展示详细描述、要求和时间等，已登录用户根据身份显示相应操作按钮，如发布者可管理任务进度，领取者可提交完成申请。

社区管理

社区列表展示社区名称、主题和成员数，搜索框方便用户查找。社区详情页有公告、帖子列表和成员列表等模块，用户可发布帖子、参与讨论和查看成员信息。管理员拥有社区设置权限，如修改信息、管理成员和置顶帖子。

积分管理

积分详情页呈现积分余额、获取和交易记录，获取记录详细说明来源，交易记录展示交易明细。积分商城页面列出可兑换商品或服务，点击兑换后系统

处理并提示结果，确保积分扣除和兑换操作准确无误。

个人中心

个人信息页面展示用户名、邮箱、电话和以太坊地址等，用户可编辑修改。订单管理页面列出所有订单，显示编号、时间、状态和金额，用户可查看详情和操作订单（如取消）。设置页面提供密码修改、通知和隐私设置等个性化功能。

5.1.3 响应式设计

运用弹性布局和媒体查询技术，确保前端页面在桌面电脑、平板电脑和手机等不同设备上自适应显示。元素大小、位置和排版根据屏幕尺寸动态调整，保证内容清晰可读，操作便捷，为用户提供一致体验。

5.2 外部接口

5.2.1 后端技术选型与架构

后端采用 Nest.js 框架搭建，利用其模块化、依赖注入等特性构建高效稳定的服务端应用。结合 TypeScript 语言，增强代码的可读性和可维护性，便于团队协作开发。

5.2.2 数据存储与交互

与 MySQL 数据库交互

通过 TypeORM 实现与 MySQL Server 8.0.23 数据库的连接和操作。定义数据实体类与数据库表结构映射，使用 Repository 模式进行数据的增删改查。例如，在用户模块中，通过 UserRepository 处理用户注册、登录和信息管理等操作，确保数据操作的准确性和高效性。

邮件发送接口

借助 nodemailer 模块实现邮件发送功能，配置 163 邮箱服务器参数，包括服务器地址、端口、用户名和密码等。用于发送注册验证邮件、密码找回链接等，确保邮件内容准确、格式规范，能及时送达用户邮箱。

5.2.3 接口安全与性能优化

接口安全

实施安全认证和授权机制，采用 JSON Web Tokens（JWT）技术验证用户身

份。用户登录成功后获取 JWT 令牌，后续请求携带该令牌，后端验证其有效性以确保合法访问。对敏感接口进行权限控制，依据用户角色和权限限制操作，防止非法请求和数据泄露。

性能优化

优化数据库查询，合理创建索引，避免全表扫描，提高数据检索速度。例如，针对任务查询接口，根据任务名称、发布者等常用查询条件建立索引。采用连接池管理数据库连接，减少连接创建和销毁开销，提升系统并发处理能力。对频繁访问的数据进行缓存，如热门任务和社区信息，使用 Redis 等缓存工具降低数据库压力，加快接口响应速度。

5.3 内部接口

5.3.1 模块间通信机制

各模块借助 Nest.js 的依赖注入机制实现解耦和通信。通过定义清晰的服务接口和模块间的依赖关系，确保数据在不同模块间准确传递和共享，维持系统的一致性和稳定性。

5.3.2 参数传递方式

Session 传递用户信息

用户登录成功后，用户信息存储在 Session 中，在整个会话期间有效。其他模块从 Session 中获取用户 ID 等信息，用于关联操作记录和验证用户权限，如任务提交、社区发帖等操作均依赖 Session 中的用户信息。

GET 请求传递参数

在页面跳转时，通过 GET 请求的 URL 参数传递简单查询参数或状态标识。如从任务列表跳转到详情页时，将任务 ID 作为参数传递，详情页依据此 ID 从数据库获取并展示任务详细信息。

表单提交传递数据

用户进行数据提交操作（如注册、任务发布、帖子发布）时，通过 HTML 表单将数据发送至后端。后端使用 Nest.js 的请求处理机制将表单数据映射到相应的 DTO（Data Transfer Object）对象，进行数据验证和业务处理，确保数据完整性和准确性。

5.3.3 接口规范与文档

制定统一的内部接口规范，明确接口功能、参数、返回值和异常处理等内容。编写详细接口文档，涵盖接口名称、地址、请求方法、参数说明、响应结构和示例代码等，方便开发人员理解和使用接口，提升团队协作效率。同时，对接口进行版本管理，确保系统升级和迭代过程中接口的兼容性和稳定性。

6 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6-1所示。

表 6-1 角色授权设计表

角色	任务发布	任务领取	社区创建	帖子发布	积分交易	用户管理	系统设置
游客	×	×	×	×	×	×	×
普通玩家	×	√	×	√	√	×	×
任务发布者	√	×	×	√	√	×	×
社区管理员	×	×	√	√	√	×	√
系统管理员	√	×	√	√	√	√	√

7 系统错误处理

7.1 出错信息

(1) 用户输入错误：如注册时用户名已存在、密码不符合要求等，在页面提示具体错误信息，引导用户修改。

(2) 任务操作错误：如任务已过期不能领取、任务完成不符合要求等，提示错误原因，用户可根据提示调整操作。

(3) 系统内部错误：如数据库连接失败、智能合约执行出错等，显示友好的系统错误提示，记录错误日志，通知系统管理员处理。

7.2 故障预防与补救

(1) 数据备份与恢复

定期备份数据库，包括用户数据、任务数据、社区数据等，备份频率为每天一次，存储在异地服务器。

当出现数据丢失或损坏时，可使用备份数据恢复系统。

(2) 智能合约安全

在部署前进行严格的代码审查和安全测试，采用专业的区块链安全工具进行漏洞扫描，确保合约代码逻辑正确且无安全隐患。

建立智能合约监控机制，实时监测合约执行情况，一旦发现异常交易或潜在风险，及时暂停合约执行并进行调查处理。

(3) 系统监控与预警

部署系统监控工具，对服务器资源（CPU、内存、磁盘、网络等）进行实时监控，设置阈值，当资源使用率超过阈值时发出预警。

监控系统运行状态，包括服务的可用性、响应时间等，当系统出现故障或性能下降时，及时通知系统管理员进行处理。

(4) 安全防护措施

采用防火墙、入侵检测系统（IDS）、入侵防御系统（IPS）等网络安全设备，防止外部非法攻击，保护系统安全。

对用户输入数据进行严格的验证和过滤，防止 SQL 注入、跨站脚本攻击（XSS）等常见安全漏洞，确保系统输入安全。

7.3 系统维护设计

(1) 系统更新与升级

定期评估系统性能和功能需求，根据用户反馈和业务发展需要，制定系统更新计划，确保系统持续优化。

升级系统时，进行充分的测试，包括功能测试、性能测试、兼容性测试等，确保升级后的系统稳定可靠，不影响用户正常使用。

(2) 日志管理

记录系统运行过程中的各类操作日志，包括用户操作、任务执行、交易记录、系统错误等，便于故障排查和系统审计。

定期清理过期日志，确保日志文件不会占用过多存储空间，影响系统性能。

(3) 性能优化

持续关注系统性能指标，如响应时间、并发处理能力等，通过优化数据

库查询、调整服务器配置、优化代码算法等方式，不断提升系统性能。

定期进行性能测试，模拟高并发场景，找出性能瓶颈并进行针对性优化。

(4) 技术支持与培训

为用户提供技术支持渠道，如在线客服、电子邮件、电话等，及时解答用户在使用过程中遇到的问题。

针对系统管理员和关键用户，定期开展系统操作和维护培训，提高用户对系统的熟悉程度和操作技能，确保系统正常运行和有效维护。