# 附件3 电子商务平台详细设计

**目录**

[附件3 电子商务平台详细设计 1](#_Toc138966651)

[1.导言 2](#_Toc138966652)

[1.1 目的 2](#_Toc138966653)

[1.2 范围 2](#_Toc138966654)

[1.3 编写说明 2](#_Toc138966655)

[1.4 术语定义 2](#_Toc138966656)

[1.5 引用标准 2](#_Toc138966657)

[1.6 参考资料 2](#_Toc138966658)

[1.7 版本更新信息 2](#_Toc138966659)

[2.系统设计概述 2](#_Toc138966660)

[3. 详细设计概述 3](#_Toc138966661)

[4. 用户登录模块的详细设计 4](#_Toc138966662)

[5. 订单管理模块的详细设计 4](#_Toc138966663)

[6. 供应商管理模块的详细设计 5](#_Toc138966664)

[7. 商品管理模块的详细设计 6](#_Toc138966665)

[8. 最小模块组件的伪代码 7](#_Toc138966666)

[8.1 用户注册功能： 7](#_Toc138966667)

[8.2 用户登录功能： 7](#_Toc138966668)

[8.3 商品列表展示功能： 7](#_Toc138966669)

[8.4 添加商品到购物车功能： 7](#_Toc138966670)

[8.5从购物车移除商品功能： 8](#_Toc138966671)

[8.6 创建订单功能： 8](#_Toc138966672)

[8.7 支付功能： 8](#_Toc138966673)

[8.8 订单查询功能： 8](#_Toc138966674)

[8.9 用户信息查询功能： 8](#_Toc138966675)

[8.10 用户信息更新功能： 8](#_Toc138966676)

[8.11 商品详情查询功能： 9](#_Toc138966677)

[8.12 商品分类查询功能： 9](#_Toc138966678)

[8.13 购物车总价计算功能： 9](#_Toc138966679)

[8.14 收货地址管理功能： 9](#_Toc138966680)

[8.15 发布商品功能： 9](#_Toc138966681)

[8.16 商品库存更新功能： 10](#_Toc138966682)

[8.17 数据统计分析功能： 10](#_Toc138966683)

**1.导言**

## 1.1 目的

该文档的目的是描述《电子商务平台》项目的详细设计，其主要内容包括：

·系统功能简介；

·系统详细设计简述；

·各个模块的三层划分；

·最小模块组件的伪代码。

本文档的预期的读者是：

·开发人员；

·项目管理人员；

·测试人员。

## 1.2 范围

该文档定义了系统的各个模块和模块接口，但未确定单位的具体实现，这部分内容将在程序实现中确定。

## 1.3 编写说明

MVC：Model-View-Control（模式—视图—控制）的缩写，表示一个三层的结构体系。

## 1.4 术语定义

无

## 1.5 引用标准

无

## 1.6 参考资料

无

## 1.7 版本更新信息

本文档版更新记录如表C—1所列。

**表C-1 版本更新记录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 修改编号 | 修改日期 | 修改后版本 | 修改位置 | 修改内容概述 |
| 000 | 2023.6.30 | 1.0 | 全部 | 初始发布版本 |

# 2.系统设计概述

根据《电子商务平台》的概要设计，本系统按照功能分解，可以分为管理端子系统和客户端子系统。根据页面流的设计，管理端系统分为登录管理，订单管理，商品管理，供货商管理等模块，客户端系统分为商品，订单，购物车等模块，它们的关系如图C-1所示，以下将对各个部分分别进行详细设计。

**图C-1 模块设计**

# 3. 详细设计概述

由于本系统采用了 FastAPI 和 Vue 的组合，结合了后端和前端的开发，采用 MVC（Model-View-Controller）的设计模式。整个开发过程中，将尽可能遵循代码复用的原则，例如使用标准库、统一数据库的基本操作和结果显示等。

在本文档的详细设计中，将按照 MVC 架构的三个层次（模型、视图、控制器）编写各个模块的伪代码。伪代码是一种算法描述语言，用于描述算法和程序逻辑，具有结构清晰、代码简单、易于理解的特点。它不依赖于特定的编程语言，因此可以方便地转换为实际编码。

本系统的分层结构如下：

1. 模型层（Model）：

· 定义数据模型和数据结构，包括数据库表结构、数据验证规则等。

· 实现数据库的访问和操作，包括数据的增删改查等基本操作。

· 处理业务逻辑和数据处理的方法，例如数据的处理、计算和验证。

2. 视图层（View）：

· 设计后端 API 接口，用于接收前端的请求并进行处理。

· 解析请求参数，验证输入数据的合法性，并调用相应的控制器方法进行业务处理。

· 根据业务需求生成响应数据，并返回给前端。

3. 控制器层（Controller）：

· 实现后端 API 接口对应的具体处理函数，负责处理来自视图层的请求。

· 调用模型层的方法进行业务逻辑处理，包括数据的获取、处理和操作。

* 根据业务需求生成相应的结果数据，并返回给视图层。

在整个设计过程中，应注意模块之间的接口定义和数据传递方式，确保数据的准确性和一致性。同时，为了提高代码的可维护性和可扩展性，可以采用模块化的设计思想，将功能相关的代码组织在一起，实现代码的复用和解耦。

# 4. 用户登录模块的详细设计

用户登录管理主要实现了对用户登录和验证和权限管理功能，用户登录管理的视图层、控制层和模型层三个层次的模块如表C-2所示。

表C-2 用户登录管理的三层模块

|  |  |  |
| --- | --- | --- |
| 层级 | 功能 | 说明 |
| 模型层 | 用户类 | 包含用户的属性（例如用户名、密码等）和相关方法 |
|  |  | 处理用户的保存、查询和密码验证等操作 |
| 视图层 | 登录页面 | 提供用户登录的界面，接收用户输入的用户名和密码 |
|  | 注册页面 | 提供用户注册的界面，接收用户输入的用户名和密码 |
|  |  | 将用户输入的数据传递给控制器层进行处理 |
| 控制器层 | 用户登录功能 | 负责处理用户登录的业务逻辑 |
|  |  | 接收视图层传递的用户名和密码 |
|  |  | 调用模型层的方法进行用户名和密码的验证 |
|  |  | 返回登录结果给视图层 |
|  | 用户注册功能 | 负责处理用户注册的业务逻辑 |
|  |  | 接收视图层传递的用户名和密码 |
|  |  | 创建用户对象并调用模型层的方法将用户信息保存到数据库 |
|  |  | 返回注册结果给视图层 |

# 5. 订单管理模块的详细设计

订单管理主要实现了对订单的查询/创建/删除功能，订单管理的视图层、控制层和模型层三个层次的模块如表C-3所示。

表C-3 订单管理的三层模块

|  |  |  |
| --- | --- | --- |
| 层级 | 功能 | 说明 |
| 模型层 | 订单类 | 包含订单的属性（例如订单号、商品信息、价格等）和相关方法 |
|  |  | 处理订单的创建、查询和删除等操作 |
|  |  | 处理订单状态的变更和计算订单金额等业务逻辑 |
| 视图层 | 订单列表页面 | 显示订单列表，包括订单号、商品信息、价格和状态等 |
|  | 订单详情页面 | 显示特定订单的详细信息，包括订单号、商品信息、价格、状态和配送信息等 |
| 控制器层 | 订单查询功能 | 负责处理订单查询的业务逻辑 |
|  |  | 接收视图层传递的查询条件（例如订单号、时间范围、状态等） |
|  |  | 调用模型层的方法根据查询条件从数据库中获取符合条件的订单列表 |
|  |  | 返回订单列表给视图层进行显示 |
|  | 订单创建功能 | 负责处理订单创建的业务逻辑 |
|  |  | 接收视图层传递的订单信息（例如商品信息、价格、配送信息等） |
|  |  | 调用模型层的方法创建订单对象并保存到数据库 |
|  |  | 返回订单创建结果给视图层 |
|  | 订单删除功能 | 负责处理订单删除的业务逻辑 |
|  |  | 接收视图层传递的订单号 |
|  |  | 调用模型层的方法根据订单号从数据库中删除订单 |
|  |  | 返回订单删除结果给视图层 |

# 6. 供应商管理模块的详细设计

表C-4 供应商管理的三层模块

|  |  |  |
| --- | --- | --- |
| 层级 | 功能 | 说明 |
| 模型层 | 供应商类 | 包含供应商的属性（例如供应商编号、名称、联系信息等）和相关方法 |
|  |  | 处理供应商的创建、查询、修改和删除等操作 |
|  |  | 处理供应商信息的验证和业务逻辑 |
| 视图层 | 供应商列表页面 | 显示供应商列表，包括供应商编号、名称、联系信息等 |
|  | 供应商详情页面 | 显示特定供应商的详细信息，包括供应商编号、名称、联系信息等 |
| 控制器层 | 供应商查询功能 | 负责处理供应商查询的业务逻辑 |
|  |  | 接收视图层传递的查询条件（例如供应商名称、联系人等） |
|  |  | 调用模型层的方法根据查询条件从数据库中获取符合条件的供应商列表 |
|  |  | 返回供应商列表给视图层进行显示 |
|  | 供应商创建功能 | 负责处理供应商创建的业务逻辑 |
|  |  | 接收视图层传递的供应商信息（例如名称、联系信息等） |
|  |  | 调用模型层的方法创建供应商对象并保存到数据库 |
|  |  | 返回供应商创建结果给视图层 |
|  | 供应商删除功能 | 负责处理供应商删除的业务逻辑 |
|  |  | 接收视图层传递的供应商编号 |
|  |  | 调用模型层的方法根据供应商编号从数据库中删除供应商 |
|  |  | 返回供应商删除结果给视图层 |

供应商管理模块的三层结构如表C-4所示，其实现了以下功能：

1. 供应商查询：用户可以根据供应商名称、联系人等条件进行供应商查询，获取满足条件的供应商列表。
2. 供应商创建：用户可以创建新的供应商，输入供应商名称、联系信息等供应商详情，并将供应商保存到数据库中。
3. 供应商删除：用户可以根据供应商编号选择要删除的供应商，并从数据库中删除相应的供应商记录。
4. 供应商列表展示：系统会显示供应商列表，包括供应商编号、名称、联系信息等关键信息，以便用户浏览和查看。
5. 供应商详情查看：用户可以查看特定供应商的详细信息，包括供应商编号、名称、联系信息等。

# 7. 商品管理模块的详细设计

表C-5 商品管理的三层模块

|  |  |  |
| --- | --- | --- |
| 层级 | 功能 | 说明 |
| 模型层 | 商品类 | 包含商品的属性（例如商品编号、名称、价格等）和相关方法 |
|  |  | 处理商品的创建、查询、修改和删除等操作 |
|  |  | 处理商品信息的验证和业务逻辑 |
| 视图层 | 商品列表页面 | 显示商品列表，包括商品编号、名称、价格等 |
|  | 商品详情页面 | 显示特定商品的详细信息，包括商品编号、名称、价格等 |
| 控制器层 | 商品查询功能 | 负责处理商品查询的业务逻辑 |
|  |  | 接收视图层传递的查询条件（例如商品名称、价格范围等） |
|  |  | 调用模型层的方法根据查询条件从数据库中获取符合条件的商品列表 |
|  |  | 返回商品列表给视图层进行显示 |
|  | 商品创建功能 | 负责处理商品创建的业务逻辑 |
|  |  | 接收视图层传递的商品信息（例如名称、价格等） |
|  |  | 调用模型层的方法创建商品对象并保存到数据库 |
|  |  | 返回商品创建结果给视图层 |
|  | 商品删除功能 | 负责处理商品删除的业务逻辑 |
|  |  | 接收视图层传递的商品编号 |
|  |  | 调用模型层的方法根据商品编号从数据库中删除商品 |
|  |  | 返回商品删除结果给视图层 |
|  | 商品修改功能 | 负责处理商品修改的业务逻辑 |
|  |  | 接收视图层传递的商品编号和修改内容 |
|  |  | 调用模型层的方法根据商品编号从数据库中查询商品并更新商品信息 |
|  |  | 返回商品修改结果给视图层 |

商品管理模块的三层模型如表C-5所示，其实现了以下功能：

1. 商品查询：用户可以根据商品名称、价格范围等条件进行商品查询，获取满足条件的商品列表。
2. 商品创建：用户可以创建新的商品，输入商品名称、价格等商品详情，并将商品保存到数据库中。
3. 商品删除：用户可以根据商品编号选择要删除的商品，并从数据库中删除相应的商品记录。
4. 商品列表展示：系统会显示商品列表，包括商品编号、名称、价格等关键信息，以便用户浏览和查看。
5. 商品详情查看：用户可以查看特定商品的详细信息，包括商品编号、名称、价格等。
6. 商品修改：用户可以根据商品编号选择要修改的商品，并更新商品的相关信息。

# 8. 最小模块组件的伪代码

以下是电子商务平台的8个最小模块组件的伪代码示例：

## 8.1 用户注册功能：

```python

def register(username, password):

# 实现用户注册逻辑

# ...

return result

```

## 8.2 用户登录功能：

```python

def login(username, password):

# 实现用户登录逻辑

# ...

return result

```

## 8.3 商品列表展示功能：

```python

def get\_product\_list():

# 实现获取商品列表逻辑

# ...

return product\_list

```

## 8.4 添加商品到购物车功能：

```python

def add\_to\_cart(user\_id, product\_id, quantity):

# 实现添加商品到购物车逻辑

# ...

return result

```

## 8.5从购物车移除商品功能：

```python

def remove\_from\_cart(user\_id, product\_id):

# 实现从购物车移除商品逻辑

# ...

return result

```

## 8.6 创建订单功能：

```python

def create\_order(user\_id, cart\_items):

# 实现创建订单逻辑

# ...

return result

```

## 8.7 支付功能：

```python

def make\_payment(order\_id, payment\_info):

# 实现支付逻辑

# ...

return result

```

## 8.8 订单查询功能：

```python

def get\_order\_details(user\_id, order\_id):

# 实现订单查询逻辑

# ...

return order\_details

```

## 8.9 用户信息查询功能：

```python

def get\_user\_info(user\_id):

# 实现查询用户信息逻辑

# ...

return user\_info

```

## 8.10 用户信息更新功能：

```python

def update\_user\_info(user\_id, new\_info):

# 实现更新用户信息逻辑

# ...

return result

```

## 8.11 商品详情查询功能：

```python

def get\_product\_details(product\_id):

# 实现查询商品详情逻辑

# ...

return product\_details

```

## 8.12 商品分类查询功能：

```python

def get\_category\_list():

# 实现查询商品分类列表逻辑

# ...

return category\_list

```

## 8.13 购物车总价计算功能：

```python

def calculate\_cart\_total(user\_id):

# 实现计算购物车总价逻辑

# ...

return total\_price

```

## 8.14 收货地址管理功能：

```python

def add\_shipping\_address(user\_id, address\_info):

# 实现添加收货地址逻辑

# ...

return result

def update\_shipping\_address(user\_id, address\_id, new\_info):

# 实现更新收货地址逻辑

# ...

return result

def delete\_shipping\_address(user\_id, address\_id):

# 实现删除收货地址逻辑

# ...

return result

```

## 8.15 发布商品功能：

```python

def publish\_product(product\_info):

# 实现发布商品逻辑

# ...

return result

```

## 8.16 商品库存更新功能：

```python

def update\_product\_stock(product\_id, quantity):

# 实现更新商品库存逻辑

# ...

return result

```

## 8.17 数据统计分析功能：

```python

def generate\_sales\_report(start\_date, end\_date):

# 实现生成销售报告逻辑

# ...

return report\_data

```