B/S体系软件设计 商品比价网站 设计报告

1. 文档介绍

1.1. 编写目的

本文档描述软件产品功能设计说明书目的是：

1. 定义软件总体要求，作为用户和软件开发人员相互了解的基础。
2. 提供性能要求、初步设计和用户影响的信息，作为软件人员进行软件结构设计和编码的基础。
3. 作为软件总体测试的依据。

1.2. 文档范围

该设计报告主要是对商品比价网站中功能及功能操作方式进行基本的描述，总体设计，详细设计，用户界面，数据库设计，运行设计等。

1.3. 读者对象

编写详细设计人员及程序开发人员。

B/S体系软件设计课程的助教和授课老师。

1.4. 术语与缩写解释

术语	解释
订阅	选择特定商品，在该商品降价时能够给予用户提醒

2. 项目介绍

2.1. 项目说明

产品名称：商品比价网站

开发者：沈聪愉

用户群：注册用户、管理员

2.2. 项目背景

实现一个商品价格比较的网站，可以实现用户注册、登录，查询主流电商平台商品价格，建立商品库，设置消息提醒等功能。

2.3. 需求概述

2.3.1. 功能需求

用户模块

功能名称	功能编号
注册	u001
登录	u002
密码修改	u003
订阅商品	u004

商品模块

功能名称	功能编号
查询	p001
保存信息	p002
历史价格展示	p003

2.3.2. 性能需求

1. 用户模块的功能和商品模块的功能能够支持500个用户并发访问。
2. 系统要具有良好的反应速度，在正常的网络情况瞎，需要满足以下要求：Web响应用户动作时间小于1秒；查询并返回商品结果时间小于5秒；
3. 该系统应实现多Web浏览器支持，在大多数流行的Web浏览器中正确显示和执行，包括FireFox、Chrome、Edge等，且在手机浏览器或微信、QQ等应用内置浏览器中友好显示。

2.3.3. 安全需求

1. 用于身份验证的用户名和密码应防止未经授权的用户访问系统。应构建访问控制以防止合法用户非法使用系统资源。某些敏感数据（如用户名，密码）在交换时应加密。密码在存储之前应加密。
2. 在用户登录期间，应该防止SQL注入，密码强制破解和伪造会话入侵。
3. 为数据库加上一定的约束，对关键性操作如删除、修改进行限制，并对用户进行警示。
4. 着重账户信息安全性设计，做到外界人员无法入侵到系统本身。内部人员操作需要留下操作痕迹，
   使用权管理层可以定期或不定期地维护系统。

2.4. 条件与限制

2.4.1. 服务器配置要求

支持500人并发访问；CPU4核2.6G；内存8.0G；硬盘7200转

2.4.2. 客户端配置要求

CPU不小于2.0Ghz；内存不小于2.0GB

2.4.3. 软件依赖

操作系统：Windows 10/11，MacOS

MySQL管理软件：MySQL WorkBench等开发工具

项目管理：git

3. 总体设计

3.1. 基本设计概念和流程处理

前端：基于 HTML，CSS，JavaScript 网络前端，并且采用 Vue 框架来实现交易客户端各种页面的构建。

后端：基于 Python 的 Django 框架

数据库：数据库采用 MySQL，通过 promise 来实现异步查询

客户端：考虑到本系统所使用的框架较新，推荐使用 Chrome、Firefox、Safari 或 Microsoft Edge 浏览器。不推荐使用任何一个版本的 IE 浏览器，如需使用，版本须在 11 及以上。

3.2. 技术介绍

将采用基于HTML，CSS，JavaScript网络前端，并且采用Vue框架来实现交易客户端各种页面的构建。Vue是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue被设计为可以自底向上逐层应用。Vue的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与现代化的工具链以及各种支持类库结合使用时，Vue也完全能够为复杂的单页应用提供驱动。

Django是一个基于Python的高级Web开发框架，它采用了MVC（模型-视图-控制器）设计模式，通过强大的ORM工具与数据库交互，简化了开发过程。同时，Django还提供了自动生成的管理界面，方便对数据模型进行管理和编辑。通过路由系统，我们可以定义URL模式并将其映射到相应的视图函数，实现用户请求的处理。另外，Django的表单处理功能使得创建和验证表单变得简单，并且模板引擎能够灵活地分离HTML和Python代码。为了保证应用程序的安全性，Django内置了多种安全功能，并支持国际化。此外，Django拥有丰富的插件生态系统，可以为应用程序增加额外的功能和集成。综上所述，使用Django可以快速构建高质量的Web应用程序，在各类项目中得到广泛应用，如社交媒体平台、电子商务网站以及内容管理系统等。

4. 详细设计

4.1. 数据库设计

basic_user(普通用户)

字段名称	数据类型	是否为主键	是否为外键	是否为空	字段解释
user_id	int	Y	N	N	标识用户的唯一自增ID
user_name	varchar(20)	N	N	N	可用于登录的唯一用户名
phone	varchar(20)	N	N	N	可用于登录的唯一手机号
email	varchar(50)	N	N	N	可用于登录的唯一邮箱

字段名称	数据类型	是否为主键	是否为外键	是否为空	字段解释
password_hash	varchar(64)	N	N	N	经过SHA-256算法加密后的密码哈希值

platforms(电商平台)

字段名称	数据类型	是否为主键	是否为外键	是否为空	字段解释
platform_id	int	Y	N	N	标识电商平台的唯一自增ID
platform_name	varchar(50)	N	N	N	电商平台的唯一名称

products(商品)

字段名称	数据类型	是否为主键	是否为外键	是否为空	字段解释
product_id	int	Y	N	N	标识商品的唯一自增ID
product_name	varchar(100)	N	N	N	商品名（与平台ID构成唯一约束）
platform_id	int	N	Y	N	商品来自的电商平台ID（与商品名构成唯一约束）
deal_num	int	N	N	N	商品成交数
shop_name	char(80)	N	N	N	店铺名称
location	char(40)	N	N	N	店铺地址
text	char(500)	N	N	C	商品描述
img	mediumblob	N	N	N	商品图片
web	char(200)	N	N	N	商品链接
is_valid	bool	N	N	N	商品存在为真，商品下架为否

price_history(价格更新历史记录)

字段名称	数据类型	是否为主键	是否为外键	是否为空	字段解释
price_history_id	int	Y	N	N	商品价格变化历史记录的唯一自增ID
product_id	int	N	Y	N	商品ID
price	double	N	N	N	商品此次更新记录的价格

字段名称	数据类型	是否为主键	是否为外键	是否为空	字段解释
update_date	date	N	N	N	更新价格的日期（年月日）
update_time	time	N	N	N	更新价格的时间（时分秒）

sub_product(商品订阅)

字段名称	数据类型	是否为主键	是否为外键	是否为空	字段解释
sub_product_id	int	Y	N	N	用户商品订阅的唯一自增ID
product_id	int	N	Y	N	商品ID（与用户ID构成唯一约束）
user_id	int	N	Y	N	用户ID（与商品ID构成唯一约束）

4.2. 接口设计

4.2.1. 内部接口

1. 哈希加密模块

`SHA256` : pwd_hash(pwd: string) -> string

输入用户密码，加密后返回哈希值。

`SHA256` : is_equal(pwd: string, user_name: string) -> bool

输入用户名和密码，返回密码是否匹配的布尔值。

2. 商品模块

`ProductManager` : daily_update() -> none

服务器端每日更新数据库中商品价格和商品信息，如果无法获取则默认商品下架。

`ProductManager` : get_products(search_key_words: string[], platform_id: int) -> int

用户输入查询的商品关键字和电商平台ID，通过爬虫爬取相应商品信息，返回一共获取的商品数目。

3. 订阅模块

`SubManager` : daily_update() -> int

服务器端每日检查用户订阅的商品价格是否降低，如果降低则发送邮件，返回发送邮件数目。

4. 邮件模块

`EmailManager` : send_email(addr: string, msg: string) -> bool

向指定邮箱发送邮件，返回是否发送成功。

4.2.2. 外部接口

前端请求返回信息中`state`为**1**表示正确处理，`state`为**0**表示发生错误；`msg`表示返回描述信息。

1. 注册用户

`/user/add`

Method: POST

Request Body:

```
{
    "user_name": string,
    "phone": string,
    "email": string,
    "password_hash": string,
    "code": string
}
```

Response:

```
{
    "state": int,
    "msg": string
}
```

2. 修改密码

`/user/pwd/modify`

Method: POST

Request Body:

```
{
    "user_id": int,
    "origin_pwd_hash": string,
    "new_pwd_hash": string
}
```

Response:

```
{
    "state": int,
    "msg": string
}
```

3. 用户登录

`/user/login`

Method: POST

Request Body:

```
{
    "user_name": string,
    "pwd_hash": string
}
```

Response:

```
{
    "state": int,
    "msg": string,
    "data":
    {
        "user_id": int
    }
}
```

4. 邮箱验证

`/email/confirm`

Method: POST

Request Body:

```
{
    "email": string
}
```

Response:

```
{
    "state": int,
    "msg": string
}
```

5. 查询商品

`/product/search`

Method: GET

Path Parameter:

- `page` (int, 可选) - 页码，默认值为1
- `limit` (int, 可选) - 每页显示的商品数量，默认值为10
- `search_key_word` (string, 必选) - 用户查询商品关键词
- `platform_id` (int, 必选) - 查询商品的电商平台

Response:

```
{
    "state": int,
    "msg": string,
    "data":
```

```
    [
        {
            "product_id": int,
            "product_name": string,
            "platform_id": int,
            "deal_num": int,
            "shop_name": string,
            "location": string,
            "img": string,
            "text": string,
            "price": double
        },
        { ... }
    ]
}
```

6. 订阅商品

`/sub/product/add`

Method: POST

Request Body:

```
{
    "product_id": int,
    "user_id": int
}
```

Response:

```
{
    "state": int,
    "msg": string
}
```

7. 取消订阅商品

`/sub/product/cancel`

Method: POST

Request Body:

```
{
    "product_id": int,
    "user_id": int
}
```

Response:

```
{
    "state": int,
    "msg": string
}
```

8. 获取订阅商品列表

`/sub/product/get`

Method: GET

Path Parameter:

- `page` (int, 可选) - 页码，默认值为1
- `limit` (int, 可选) - 每页显示的商品数量，默认值为10
- `user_id` (int, 必选) - 用户唯一标识

Response:

```
{
    "state": int,
    "msg": string,
    "data":
    [
        {
            "product_id": int
        },
        { ... }
    ]
}
```

9. 获取指定ID商品

`/product/get`

Method: GET

Path Parameter:

- `product_id` (int, 必选) - 商品唯一标识

Response:

```
{
    "state": int,
    "msg": string,
    "data":
    {
        "product_name": string,
        "platform_id": int,
        "deal_num": int,
        "shop_name": string,
        "location": string,
        "img": string,
        "text": string,
        "price": double
    }
}
```

10. 查看商品最近x天价格

`/product/price/get`

Method: GET

Path Parameter:

- `product_id` (int, 必选) - 商品唯一标识
- `day` (int, 可选) - 获取最近多少天的价格变化，默认值为30

Response:

```
{
    "state": int,
    "msg": string,
    "data":
    [
        {
            "price": double
        },
        { ... }
    ]
}
```

4.3. 界面设计

使用简单搜索引擎模式，界面类似常见搜索引擎。