

**本科实验报告**

|  |  |
| --- | --- |
| 课程名称： | B/S体系软件设计 |
| 姓 名： | 马欣雨 |
| 学 院： | 计算机学院 |
| 系： | 计算机科学与技术 |
| 专 业： | 计算机科学与技术 |
| 学 号： | 3220102065 |
| 指导教师： | 胡晓军 |

2024年 11月 13日

**浙江大学实验报告**

课程名称： B/S体系软件设计 实验类型： 开发型

实验项目名称： 商品比价网站

学生姓名： 马欣雨 专业：计算机科学与技术 学号： 3220102065

同组学生姓名： 无 指导老师： 胡晓军

实验地点： 实验日期： 2024 年 11 月 13 日

目录

**[1. 项目概述 6](#_Toc5903)**

[1.1 编写目的 6](#_Toc32533)

[1.2 项目背景 6](#_Toc10323)

[1.3 目标用户 7](#_Toc26002)

**[2. 功能需求 7](#_Toc8731)**

[2.1 用户管理 7](#_Toc22008)

[2.2 商品比价查询 7](#_Toc24569)

[2.3 商品库管理 7](#_Toc22898)

[2.4 历史价格显示 7](#_Toc25338)

[2.5 降价提醒功能 7](#_Toc18247)

[2.6 手机适配 7](#_Toc4832)

**[3. 系统架构 8](#_Toc10136)**

[3.1 技术栈 8](#_Toc9491)

[3.2 系统架构图 8](#_Toc5015)

[3.3 系统模块设计 8](#_Toc27999)

**[4. 数据库设计 9](#_Toc11228)**

[4.1 数据库表设计 9](#_Toc4817)

**[5. 系统安全 10](#_Toc14302)**

**[6. 开发计划 10](#_Toc23910)**

**[7. 测试方案 10](#_Toc3618)**

**[8. 部署方案 10](#_Toc9132)**

1. 项目概述
   1. 编写目的

本文档描述软件产品功能设计说明书（SRS）的目的是：

1. 定义软件产品总体要求，作为用户和软件开发人员之间相互了解的基础；
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. 项目背景

随着电商平台的崛起，消费者需要比较不同平台的商品价格，以便做出更理智的购买决策。通过构建一个商品比价网站，用户可以方便地查询多个电商平台上同一商品的实时价格，并做出选择。通过该网站，用户可以实现价格追踪、历史价格查询等功能，提升购物体验。

* 1. 目标用户

电商平台用户、消费者，尤其是关注商品价格波动和比价的用户。

1. 功能需求
   1. 用户管理
   * 用户注册：用户需要提供用户名、密码、邮箱等信息。用户名需唯一，密码长度至少6个字符，邮箱格式需合法。
   * 用户登录：用户使用用户名和密码登录系统，支持登录状态保持（例如使用JWT或Session等技术）。
   * 用户信息管理：用户可以查看和更新个人信息（如邮箱、密码等）。
   1. 商品比价查询
   * 商品查询：通过商品名称进行商品搜索，系统会将商品名称进行分词处理以优化查询效果。支持从多个电商平台（如淘宝、京东等）抓取实时商品价格。
   * 多个查询结果处理：对于同一商品在不同平台的多个查询结果，系统需展示各平台的价格、优惠信息、商品详情等。
   * 电商平台登录验证：部分平台需要用户登录后才能查询价格信息，系统需要支持通过用户提供的账号信息获取商品价格。
   1. 商品库管理
   * 商品信息存储：将商品的详细信息（包括商品名称、多级品类、规格、条码、图片等）存储在数据库中，便于后续管理和查询。
   * 商品价格存储：将商品的历史价格存储在数据库中，供用户查看历史价格变化趋势。
   1. 历史价格显示

价格走势图：用户可以查看商品的历史价格走势，采用图表形式展示，以便用户了解商品价格的变化趋势。

* 1. 降价提醒功能
  + 定时查询价格：用户可以设置降价提醒功能，指定商品并设置期望价格。当价格低于或等于用户设定的提醒价时，系统通过邮件或推送通知用户。
  + 提醒方式：提醒可以通过邮件或APP推送等方式实现。
  1. 手机适配
  + 响应式设计：网站页面应支持响应式设计，能够在手机浏览器中良好展示。
  + 手机端支持：如果开发手机端应用，用户可以使用相机拍摄商品图片或扫码商品条码进行商品查询。

1. 系统架构
   1. 技术栈
      * 前端技术：

HTML5、CSS3、JavaScript:用于构建网页。

Vue.js或React:前端框架，用于构建用户交互界面。

Axios:用于与后端进行API通信。

* + - 后端技术：

Python或Node.js:用于开发后端服务。

Django（如果使用Python）或Express.js（如果使用Node.js）：作为Web框架。

Redis：用于缓存商品价格数据，减少对电商平台的频繁请求。

* + - 数据库：

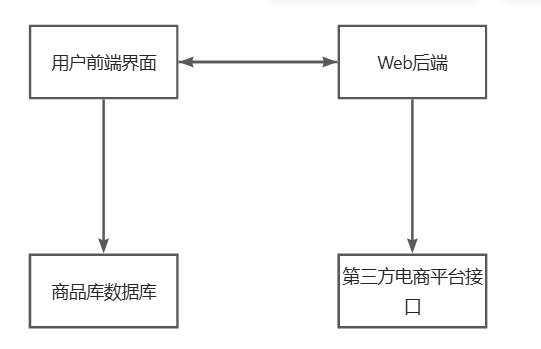
MySQL：用于存储商品信息和历史价格。

* + - 第三方服务：

邮件推送服务（如SendGrid）：用于降价提醒。

推送通知服务（如Firebase等）：用于App推送。

* 1. 系统架构图



* 1. 系统模块设计
     1. 用户模块
  + 注册、登录、修改个人信息
  + 密码加密与存储
    1. 商品比价模块
  + 商品查询功能
  + 电商平台接口集成
  + 商品信息展示与多平台价格对比
    1. 价格趋势模块
  + 图表展示商品历史价格趋势
    1. 降价提醒模块
  + 定时查询商品价格
  + 发送邮件或推送通知提醒
    1. 手机端适配模块
  + 响应式设计
  + 条形码扫描与商品拍照识别

1. 数据库设计
   1. 数据库表设计
      1. 用户表（users）

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| user\_id | INT(PK) | 用户ID（主键） |
| username | VARCHAR(255) | 用户名（唯一） |
| password | VARCHAR(255) | 用户密码（加密） |
| email | VARCHAR(255) | 用户邮箱（唯一） |
| create\_at | DATETIME | 注册时间 |

* + 1. 商品表（products）

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| product\_id | INT(PK) | 商品ID（主键） |
| name | VARCHAR(255) | 商品名称 |
| category | VARCHAR(255) | 商品品类 |
| specs | TEXT | 商品规格 |
| barcode | VARCHAR(255) | 商品条码 |
| Image\_url | VARCHAR(255) | 商品图片URL |

* + 1. 价格表（product\_prices）

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| price\_id | INT(PK) | 价格ID（主键） |
| product\_id | INT(FK) | 商品ID（外键） |
| platform | VARCHAR(255) | 电商平台名称 |
| price | DECIMAL(10,2) | 商品价格 |
| timestamp | DATETIME | 记录时间 |

* + 1. 降价提醒表（price\_alerts）

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| alert\_id | INT(PK) | 提醒ID（主键） |
| user\_id | INT(FK) | 用户ID（外键） |
| product\_id | INT(FK) | 商品ID（外键） |
| target\_price | DECIMAL(10,2) | 目标价格 |
| alert\_time | DATETIME | 提醒设置时间 |

用户与商品、价格提醒表通过外键关联。

商品与价格表通过product\_id关联。

1. 系统安全

* 密码加密：使用加密算法（如bcrypt）存储用户密码。
* API安全：使用JWT或OAuth2实现API的安全认证和授权。
* 数据隐私：通过HTTPS确保数据传输的安全性，避免数据泄露。

1. 开发计划

* 阶段一：完成基本的用户注册、登录模块和商品查询功能。
* 阶段二：实现电商平台API接口集成，并支持商品价格比对。
* 阶段三：完成商品价格历史显示与价格提醒功能。
* 阶段四：进行系统优化、性能调优，并进行系统测试。

1. 测试方案

* 单元测试：对核心功能模块进行单元测试。
* 集成测试：测试各个模块之间的协作，如前后端数据交互。
* 系统测试：进行系统的整体测试，包括功能测试、性能测试等。

1. 部署方案

* Docker容器化：通过Docker将项目容器化，确保项目在不同环境中能够一直运行。
* CI/CD：使用GitHub进行代码的持续集和持续部署。