

UNDERGRADUATE PROJECT REPORT

Project Title:	Design of a web-based price comparison system for an online shopping platform
Surname:	Zhu
First Name:	Xunran
Student Number:	201918010408
Supervisor Name:	Albert Xu
Module Code:	CHC 6096
Module Name:	Project
Date Submitted:	May 5, 2023

Chengdu University of Technology Oxford Brookes College

Chengdu University of Technology

BSc (Single Honours) Degree Project

Programme Name: **Computer Science**

Module No.: **CHC 6096**

Surname: Zhu

First Name: Xunran

Project Title: Design of a web-based price comparison system for an online shopping platform

Student No.: 201918010408

Supervisor: Albert Xu

2ND Supervisor (if applicable): **Not Applicable**

Date submitted: **May 5, 2023**

A report submitted as part of the requirements for the degree of BSc (Hons) in Computer Science

At

Chengdu University of Technology Oxford Brookes College

Declaration

Student Conduct Regulations:

Please ensure you are familiar with the regulations in relation to Academic Integrity. The University takes this issue very seriously and students have been expelled or had their degrees withheld for cheating in assessment. It is important that students having difficulties with their work should seek help from their tutors rather than be tempted to use unfair means to gain marks. Students should not risk losing their degree and undermining all the work they have done towards it. You are expected to have familiarised yourself with these regulations.

<https://www.brookes.ac.uk/regulations/current/appeals-complaints-and-conduct/c1-1/>

Guidance on the correct use of references can be found on www.brookes.ac.uk/services/library, and also in a handout in the Library.

The full regulations may be accessed online at

<https://www.brookes.ac.uk/students/sirt/student-conduct/>

If you do not understand what any of these terms mean, you should ask your Project Supervisor to clarify them for you.

I declare that I have read and understood Regulations C1.1.4 of the Regulations governing Academic Misconduct, and that the work I submit is fully in accordance with them.

Signature *Xunran Zhu* Date2023.5.5.....

REGULATIONS GOVERNING THE DEPOSIT AND USE OF OXFORD BROOKES UNIVERSITY MODULAR PROGRAMME PROJECTS AND DISSERTATIONS

Copies of projects/dissertations, submitted in fulfilment of Modular Programme requirements and achieving marks of 60% or above, shall normally be kept by the Oxford Brookes University Library.

I agree that this dissertation may be available for reading and photocopying in accordance with the Regulations governing the use of the Oxford Brookes University Library.

Signature *Xunran Zhu* Date2023.5.5.....

Acknowledgment

I have received a lot of help and support from many people in completing this report. I would like to express my sincere gratitude to those who helped and supported me in completing this report.

First of all, I would like to thank my supervisor Albert Xu, who provided me with a better direction and research idea. Despite his busy schedule, he still guided me in his free time, and his patient teaching has benefited me a lot in the writing process.

Secondly, I would like to thank my family. I would like to thank them for their support in material and spiritual culture. They provided me with a good study environment, and because of this environment, I was able to complete the thesis more successfully.

Finally, I would like to thank my classmates and friends. They accompanied me when I encountered difficulties and bottlenecks in my writing, and constantly encouraged and supported me so that I could adjust my mind faster and finish the report with confidence at all times.

I would like to express my most sincere thanks to all those who have helped and supported me!

Table of Contents

Declaration	ii
Acknowledgment.....	iii
Table of Contents	iv
Abstract	v
Abbreviations	vi
Glossary.....	vii
Chapter 1 Introduction.....	1
1.1 Background	1
1.2 Aim.....	1
1.3 Objectives	2
1.4 Project Overview	3
1.4.1 Scope.....	3
1.4.2 Audience.....	3
Chapter 2 Background Review	5
Chapter 3 Methodology	8
3.1 Approach.....	8
3.2 Technology	13
3.3 Project Version Management.....	14
Chapter 4 Results.....	16
Chapter 5 Professional Issues.....	43
5.1 Project Management	43
5.1.1 Activities	43
5.1.2 Schedule	44
5.1.3 Project Data Management	45
5.1.4 Project Deliverables.....	45
5.2 Risk Analysis.....	46
5.3 Professional Issues	49
Chapter 6 Conclusion	50
References	52
Appendices	54

Abstract

With the rapid development of network information technology, there are many online shopping platforms with a wide variety of products, so more and more people are willing to choose to shop on the Internet compared to offline shopping. With many online shopping platforms, when people want to buy a product, they will open multiple online shopping pages to search for the same product for price comparison, which will inevitably cause too many pages and affect the price comparison screening and shopping experience. In order to effectively solve the problem of opening multiple web pages when comparing prices, the authors now design a web-based online shopping platform price comparison system, which uses python crawler technology to crawl the information of the products that people need to compare prices from different websites, and then automatically import the data into the database after the crawl is completed, and finally display the data on the HTML page of the price comparison system for The data will be displayed on the HTML page of the price comparison system for people to compare and filter. Based on the above design idea, the authors have implemented a crawler technology that can successfully obtain web data and save it in the database, display the product information on the HTML interface and use fuzzy matching to compare the prices of products on different platforms. It helps people to compare prices and improve the shopping experience to a certain extent.

Keywords: Online shopping, Crawler technique, Python, HTML, Product Comparison

Abbreviations

CRUD: Create, Read, Update, Delete

DDL: Data Definition Language

DML: Data Manipulation Language

DCL: Data Control Language

TCL: Transaction Control Language

BLOB: Binary Large Objects

DBIM: Database Information Modeling

ADG: Attribute Dependency Graph

CPU: Central Processing Unit

GPU: Graphics Processing Unit

SOAP: Simple Object Access Protocol

Glossary

- 1) Bloom filter: A data structure that is mainly used to quickly determine whether an element belongs to a certain set.
- 2) Breadth-first strategy: A search algorithm, usually used for graph search or traversing data structures (such as trees or graphs).
- 3) Multi-threading technology: It refers to the technique of executing multiple threads simultaneously in the same process and can play an important role in handling multitasking and data sharing.
- 4) SOAP technology: It is an XML (Extensible Markup Language) based messaging protocol for communicating between applications in a distributed network. Interoperability can be achieved between different applications, operating systems and programming languages.
- 5) DBIM: It is a technique for visualizing and modeling data and information.
- 6) ADG: It is a data mining tool for visualizing the dependencies between attributes in data.
- 7) TPC-H test: It is a decision support benchmark test to measure the performance of a relational database management system (RDBMS).
- 8) Jinja2: It is a modern, beautifully designed, Python-based template engine that can be used in web development to dynamically generate HTML, XML and other formats of documents.
- 9) Urllib: It is a Python standard library for handling URL addresses, providing a series of modules and functions that can be used to send HTTP requests to web servers, fetch web page content, and other operations.

Chapter 1 Introduction

1.1 Background

In the 21st century, with the gradual rise of Internet technology, people's lifestyles and shopping habits have changed, and online shopping has become increasingly popular [1]. As a kind of e-commerce platform, online shopping platform enables consumers to purchase goods through the Internet using digital payment system. Compared to traditional brick-and-mortar stores, online shopping platforms have the advantages of convenience, selectivity, and freedom of shopping time, making it possible for consumers to make purchases wherever and whenever they have access to the Internet. In 2020, the sales of the Tmall shopping festival in China reached 498.2 billion RMB [2]. Today, due to different business models, these online shopping platforms often have different prices for the same item at the same time. For online shoppers, when they shop online, they want to get what they want at the lowest price. This requires consumers to compare prices of the same item across multiple platforms, and therefore requires opening multiple web interfaces. Obviously, although this method can achieve the purpose of price comparison, the operation is very troublesome and time-consuming, which not only increases the amount of page views for consumers, but also reduces the sense of shopping experience for consumers. It is difficult to compare prices of products from different merchants or different platforms, which makes it difficult for consumers to make informed shopping decisions. If there is a price comparison system that can display matching products from multiple shopping platforms after users enter relevant search terms, and display the product information and price comparison results, it will provide great convenience to users. Users can search for the products they want according to their needs and filter the search results to find the products they want to compare prices more quickly. The purpose of this paper is to study how to implement this price comparison system to improve the convenience of consumer shopping.

1.2 Aim

In this report, the data extraction method based on Web and web crawler technology are used to extract information from multiple online shopping platforms through the search function of the search bar of the home page of the system, and then the extracted information is integrated. Finally, the integrated results are displayed on the interface of the system, so that consumers can compare prices of similar products on each online shopping platform.

1.3 Objectives

(1) Define project requirements and objectives: Define the problem to be solved and the ultimate goal to be achieved.

(2) Background research and resource organization: Consult and collect related literature of the subject, and conduct background investigation of the subject. On this basis, the project is studied and researched, and the project conceptual model is proposed.

(3) Establish project development steps: Design the price comparison system, which is divided into crawler design, database design, front-end UI design and back-end functional code design.

(4) Database design: Including requirements analysis, conceptual structure design, logical design (ER modeling), physical design, successful database implementation, database testing, and database deployment and maintenance, which includes issues such as data security and data privacy.

(5) Crawler technology design: First determine the crawl target, develop crawl rules according to the target, verify and store the crawled data after realizing the crawler design function, then analyze and process the data, and finally update and maintain the data regularly to ensure the stability and reliability of the crawler.

(6) Front-end UI design: First determine the design requirements, followed by the development of a suitable design plan, including page layout and interaction design, etc., but also according to the project situation appropriate to add some creative design, the next is to determine the final design, including interface design, responsive design, etc., and finally is the test to ensure the reliability of the design and compatibility.

(7) Back-end code design: Determine the design requirements, develop the corresponding technical solutions according to the requirements, including the technologies and frameworks used, then write the code to achieve the core functions of the project and business requirements, followed by code debugging and testing, including performance testing, security testing, etc., and finally the overall integration and deployment of the code.

(8) Test project: Firstly, determine the test objectives, including the functions and requirements to be tested; secondly, develop the test plan, including the preparation of

test data and the construction of test environment; then execute test cases, including unit test, integration test, system test, etc.; next, find defects and fix them so that the project functions can operate normally; finally, prepare test reports and summaries.

(9) Interpret and deal with product instructions, hazard instructions and possible safety effects.

(10) Realize the interface showing the price comparison results of commodities on different platforms to users.

1.4 Project Overview

1.4.1 Scope

The price comparison system of online shopping platform is web-based, mainly providing a more convenient way for consumers to compare the price of the goods they need to buy, its main function is to integrate the price and information of the same goods from different platforms, without the need to open multiple web interfaces, directly on a web page to show the price comparison information, users can search for goods according to their needs, after which they will get the corresponding price comparison results, providing great convenience for users.

The price comparison system will first display an initial interface where you need to log in to perform product filtering and price comparison operations. Once you have successfully logged in by entering your account password, you will be redirected to the main screen. If you don't have an account, you need to go to the registration screen to register an account, and then return to the login screen to log in. The main screen in the login status shows some items of different categories, such as cell phones, computers, etc. Users can enter the product name to compare prices or filter the products in the main screen according to their needs. When the user performs the price comparison operation, the interface will display the corresponding product information and price comparison results to achieve the purpose of price comparison. After this series of operations, users can log out of their accounts for data security to prevent data leakage.

1.4.2 Audience

The primary audience for this project is consumers who prefer to shop and compare prices online. With the development of internet technology, more and more consumers prefer to shop online for what they like and need. Before buying a product, consumers compare and evaluate the price of the product they want to choose a better deal. This

web-based online shopping platform price comparison system provides consumers with more choices and greater convenience, and can help them make more appropriate shopping decisions to a certain extent.

Chapter 2 Background Review

Sharma and Gupta[3] introduced the importance of current web crawler technology, which is mainly reflected in the retrieval of data information and other aspects. This paper also discusses a series of problems to be solved in the construction of web crawler, including how to deal with the huge number of web pages and how to choose the appropriate crawling strategy. It also shows that web crawler technology is widely used in search engine, data mining, social media analysis and other fields. The emergence of crawlers can automatically collect and analyze large amounts of data, providing convenience for people. The article also mentioned the challenges faced by web crawler, how to deal with a large number of web pages and links, consumption of a large amount of memory in the process of use and network become the main challenges at present.

In Shi, Shi and Lin 's article [4] describes how to implement news page crawling based on incremental web crawlers. To achieve this, the authors detail the principle of Bloom filter, its implementation and how to use Bloom filter to achieve incremental crawling. This article provides a complete and clear solution to achieve efficient crawling of news pages using the Scrapy crawler framework and Bloom filter technology, while avoiding duplicate web links and handling differences between sites become the main challenges.

Vadivel et al. [5] introduced a novel component-based Web crawler and indexer method in this paper, which uses XML messages and Web services for communication. In this paper, the technical means used by this Web crawler and indexer are described in detail, and the advantages of these technical means are described in detail. For example, breadth-first strategy and multi-threading technology are adopted to improve system efficiency, SOAP technology is used to define message format more easily, and so on. It also explains some shortcomings of this method, for example, different algorithms may be needed when dealing with different types of web pages, and some performance problems may occur when dealing with large amounts of data.

Web crawlers can be broadly classified into four categories according to the system structure and implementation techniques, and the paper by Zhou and Wang [6] introduces the classification and design techniques of crawlers and mentions some common python crawler frameworks, including Scrapy, Pyspider, and Cola, and explains the advantages and disadvantages of these crawler frameworks. Compared to Pyspider

and Cola, Scrapy has more outstanding performance and is often used to handle large-scale projects, but is not as applicable for small-scale projects. Compared to Scrapy, Pyspider and Cola have the advantage of simplicity and ease of use.

As one of the cores of front-end technology stack, HTML plays an important role in front-end development. In Zhao's article [7], HTML5 technology and its application in mobile Web front-end design are mainly introduced. The author explains the advantages of HTML5 over traditional information technology. According to the development steps of the Web, the author analyzes the structure layer, the presentation layer and the behavior layer of the mobile Web front end, and designs the corresponding document specification according to these layers. In addition, it also introduces how to effectively improve the performance of the Web front end by using HTML5 technology. In this article, JavaScript is used to write interactive functions, local storage technology is used to cache data, and responsive design is used to adapt to different device requirements.

Yan et al. [8] introduced the application of HTML5 in the direction of multimedia, and detailed the working principle of HTML5, its advantages and disadvantages in use and its future development direction. Compared with other technologies, such as HTML4, HTML5 has more syntax support, provides better readability, and adds more applications such as local database on the basis of the original, enriching the interactive content. Compared with traditional web design technologies, HTML5 has the disadvantage of high learning cost, and compatibility is also one of the challenges HTML5 faces.

In Das and Devgan's [9] article on how to use MySQL database server, it is mentioned that MySQL is a threaded database server that is easy to use, robust and fast, and it stores large video files through BLOB, but at the same time, MySQL has the shortage of support for handling unstructured data and is not suitable for large and demanding scenarios. As mentioned in Ling [10], MySQL operates through the SQL language, which has the advantages of scalability and high performance, but the disadvantage of not supporting distributed transaction processing cannot be ignored. Oracle and MySQL are two different database management systems. In Pendse et al. [11] article describes how to use Oracle Database to extend the DBIM advantages to ADG architecture. The article details the advantages of Oracle such as high performance and high security. Compared to MySQL, Oracle's high availability and disaster recovery capabilities far exceed those of MySQL, and Oracle is generally used to handle large enterprise-class

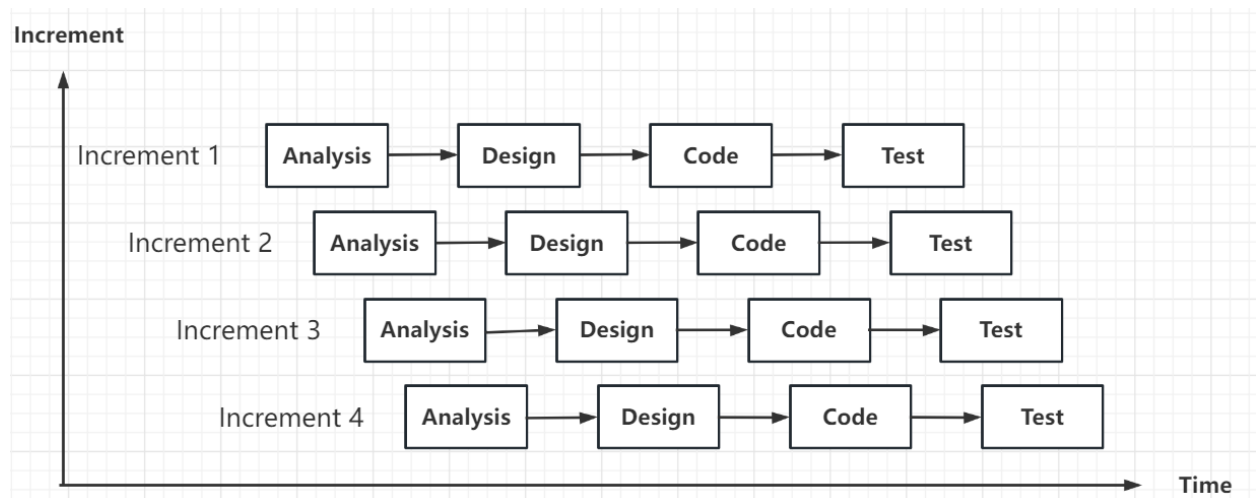
applications. Based on these functional characteristics, complexity, resource consumption, etc. become the main problems of Oracle. SQL Server, as another relational database management system different from Oracle and MySQL, also has its own advantages and disadvantages. In the article by Vershinin and Mustafina [12] the system performance of PostgreSQL, MySQL and Microsoft SQL Server was tested and compared based on TPC-H test, mainly using the DbVisualizer Free program for this operation. In the article, SQL Server and MySQL are compared and analyzed, where SQL Server supports custom data types while MySQL does not. not only that, SQL Server has a database size limit compared to MySQL and PostgreSQL. SQL Server has a stronger database management capability, so unlike MySQL, which is more suitable for large enterprise applications, is similar to Oracle in this respect.

Chapter 3 Methodology

3.1 Approach

This part is mainly divided into three parts, development model diagram, requirements analysis and technical framework.

3.1.1.1 Development model diagram



(Figure 1: Incremental model diagram)

The development model of the project is an incremental model, which divides the development into several small incremental phases, and is a step-by-step development model. Each of these phases can be completed independently, and as each increment is completed, the functionality and performance of the software will be gradually enhanced, and finally integrated into a complete system to meet the final project development requirements. The incremental model development steps are divided into the following phases:

- 1) Requirements analysis phase. The developer needs to specify a plan and schedule to determine the development goals for each increment.
- 2) Module Design Phase. The developer designs the functionality, interface, data structure, etc. based on the project requirements.
- 3) Development phase. Developers code through the requirements design.
- 4) Testing phase. Unit tests and system tests are performed, and the code is continuously corrected according to the test results.

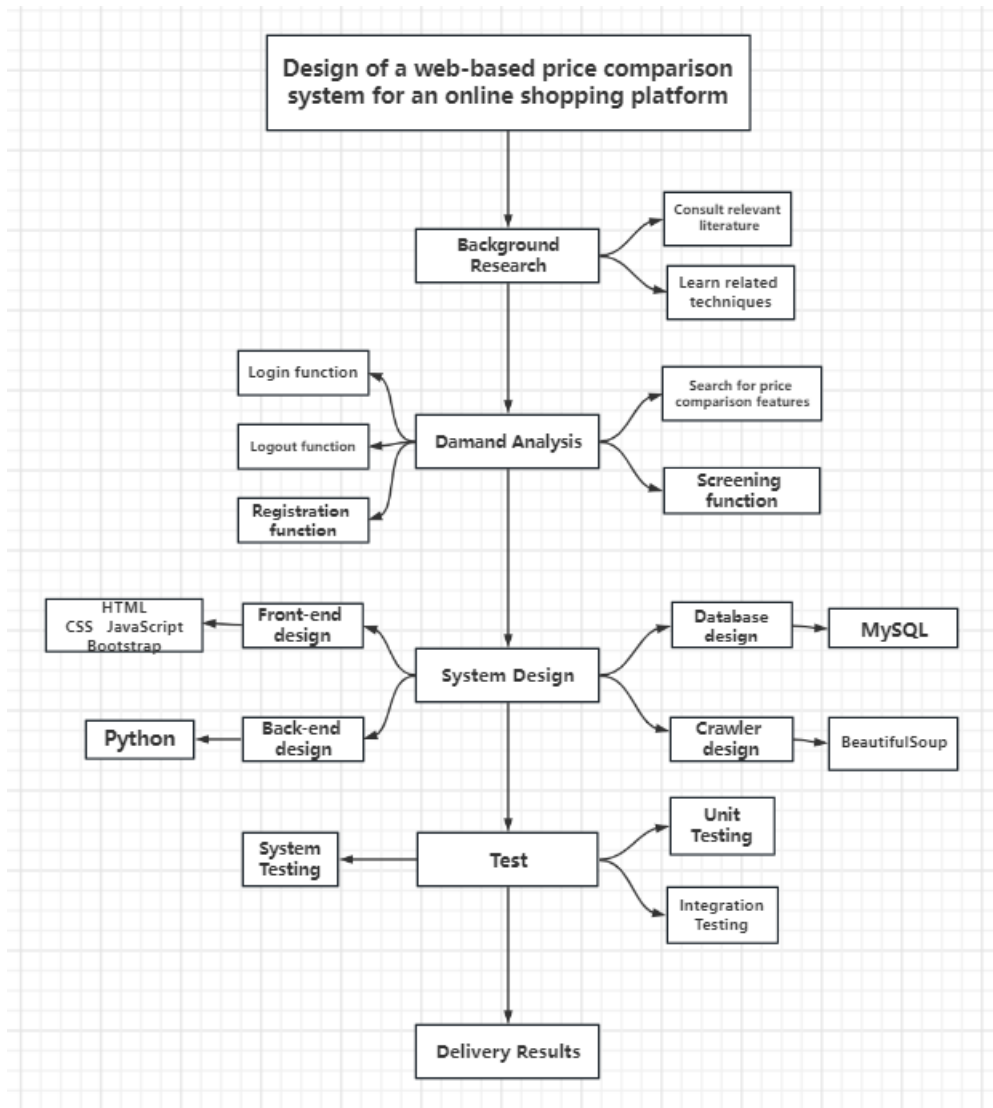
5) Integration phase. The developer integrates all the different modules, tests the whole system and user acceptance tests to ensure that the system meets the design requirements.

6) Release and maintenance phase. After releasing the project system, the developer needs to perform regular maintenance work on the system.

3.1.1.2 Requirements analysis

The product price comparison system is designed to help consumers compare prices across multiple online stores, bringing together multiple web interfaces through one platform so that consumers can choose the item that best suits their needs. Based on this requirement, first of all, the project needs to have user registration and login functions. When users use the system for the first time, they need to perform registration operation and then login operation after the operation is completed. After login, the system will save the user's personal information and display some products (e.g., cell phones, computers, etc.) on the main interface. Secondly, it should also have a filtering function. In the main interface, there will be a category selection bar, which can filter the given products, mainly divided into four categories: cell phones, computers, home appliances and books. Next, as a price comparison system, the search and price comparison function are essential. When users want to compare the price of a product on different platforms, they can enter the keywords of the product they want to compare in this search field in the main interface (e.g., iPhone 11, etc.), and the system will respond by displaying the price comparison results on the product comparison screen, including the product name, model, price and platform. When the user has completed these operations and needs to leave the price comparison system, logging out of the account is not a good option, this operation can make the user's personal information to be effectively protected, as well as to improve the security and privacy of the data. Therefore, the system also needs a logout function to achieve this purpose.

3.1.1.3 Technical framework



(Figure 2: Technical circuit diagram)

Based on the above technical framework, to achieve this project, you need to develop and design according to the needs of the project, in this project, the front-end technology using HTML-based Bootstrap framework, the back-end code is the implementation of the Python language to complete the database using MySQL, crawler design using Beautiful Soup to complete.

Bootstrap as a HTML, CSS and JavaScript-based language written in open-source framework. It has a large number of component styles and JS plug-ins. Components constitute an independent structural unit in the page, is a simple encapsulation of data and methods, with the characteristics of reusability. Based

on these features, it is often used to develop responsive layouts and mobile devices first Web project. HTML is mainly used in Bootstrap to define the page structure and content. Defining different text styles can be achieved using different tags, such as text size, boldness, etc. CSS is mainly used for style design and layout; it is a style sheet language used to control the layout and appearance of the page. In Bootstrap, CSS predefines a series of style classes that developers can apply to HTML elements to achieve style and layout effects. For example, CSS can be used to finely control elements such as text, images, borders, backgrounds, etc. It can also be used to achieve interactive effects such as mouse hover, click, etc. through animations and pseudo-classes to make web pages more readable and user-friendly. JavaScript plug-ins, as a program that extends the functionality of the browser, can be implemented through external libraries or code written by the developer himself, and can add new functionality to the existing one, thus improving the performance of the system website. jQuery, Angular, etc. are common JavaScript plug-ins that people can use to achieve their needs according to their purpose.

Python, a cross-platform computer programming language, is a high-level, interpretative, object-oriented programming language that is easy to learn because of its relatively few keywords and simple structure. It is also extensible. In addition, it is easy to read, easy to maintain, and portable, which are among the many advantages of python. In Web development, Python can implement web back-end in many ways, based on the fact that Python has many web frameworks, such as Django, Flask, Pyramid and so on. In this project, the developers chose Flask as the main framework for the project development. Flask only relies on a small number of external libraries, does not take up too many resources, and is a lightweight web application framework, which has the characteristics of easy integration and simplicity. As mentioned in Vogel et al.'s article [13], Flask has a rich ecosystem of extensions, so much so that it can be used as a development framework for countless applications and is popular and used all over the world. To implement Flask in Python based language, first install Flask framework in PyCharm compiler using pip, create Flask application, use "`@app.route('/')`" to define the route, use "`def()` function" to handle the corresponding request, so as to get a local-based virtual URL, by accessing this virtual URL to achieve the

purpose of accessing the Flask application, according to the needs of the project, add templates, in Flask default use Jinja2 as the template engine, using 'render_template' to render HTML templates in the templates directory, and 'app.config[]' to connect the database to the project back-end. Flask-Script is used to provide some command line tools for managing Flask applications, while Flask-Login is used to help developers manage user authentication and authorization. Based on the various extensions and libraries provided in Flask, the project can be designed according to the developer's needs.

MySQL is a relational database system that stores data in different tables, a feature that makes it faster and more flexible. In addition, MySQL is cost effective, secure, and compatible. SQL is a language used to manipulate databases as a structured query language that can interact with many different RDBMS (relational database management systems) such as MySQL, Oracle, SQL Server, PostgreSQL, etc. In the SQL language, there are many commands and syntax that allow you to operate on relational databases (such as add, delete, modify, and query). SQL statements can be classified into the categories of DDL, DML, DCL and TCL. Among them, DDL is used to define the structure of the database (such as creating tables), DML is used to manipulate the data in the database (such as querying, inserting), DCL is used to control the data access rights in the database (such as granting or revoking user access rights to database objects), and TCL is used to control the processing of things in the database (such as committing or rolling back transactions). SQL statements consist of keywords and operators. Developers can use the keyword SELECT to select data from the database, use the SELECT DISTINCT statement to return uniquely different values, use the WHERE clause to filter records and extract those that meet the specified conditions, etc. Developers can use different SQL statements to operate on the database according to their project needs and purposes.

Web crawler is a technology that automatically collects data from the Internet, which follows certain rules to obtain data information from websites and then saves this data information locally or uploads it to the server. As an important part of the search engine, it can collect a large amount of automated data from the Internet, including text, images, videos and so on. Scrapy, BeautifulSoup, PyQuery,

Selenium, Pyspider are some of the commonly used crawler frameworks. In this project, BeautifulSoup is used to implement web crawling techniques. BeautifulSoup is a Python library for extracting data from HTML or XML. It can parse different types of documents through different parsers to get a BeautifulSoup object, and access the tags, attributes, and content of the document according to the various methods and properties provided in BeautifulSoup. In GOEL et al.'s article [14], it is stated that BeautifulSoup is a web data crawling tool that helps developers organize and parse the data from the crawled pages. To implement a web crawler, the developer first needs to determine a suitable crawling strategy, and information such as the URL of the website and the page structure are part of determining the crawling strategy. The next step is to write the code for the crawling strategy, mainly using the Python language to achieve the development. Then the data cleaning is performed, the unwanted information is filtered, and the required information is sorted. In Wang's article [15], it is mentioned that after receiving the data from the web page parsing, the data needs to be cleaned and sorted, because the obtained data is basically in disorder, and the data cleaning method is used to sort the obtained data according to a certain requirement format, and finally it is put into the database to achieve data crawling. Requests library is also an important part of the web crawler technology implementation, it is a HTTP library, it has the unique advantage of being written closer to the URL access process, it is based on urllib, but in the data transfer is more convenient and faster than urllib, Requests library can also automatically parse JSON and XML, support SSL and other features The Requests library also automatically parses JSON and XML, supports SSL, and other features. In addition, Requests library also provides a number of HTTP methods, including GET, POST, PUT, etc., based on these advantages, so it is widely used in web crawler technology.

3.2 Technology

Software: PyCharm 2021.2.3 , MySQL Workbench 8.0 CE, MySQL 8.0 Command Line Client

Hardware: CPU: Intel(R) Core (TM) i7-8565U CPU @ 1.80GHz 1.99 GHz

GPU: Intel(R) UHD Graphics 620, NVIDIA GeForce MX250

Programming language: Python3.9, SQL, HTML

Browser: Microsoft Edge

Database: MySQL Database

Frame: Flask

3.3 Project Version Management

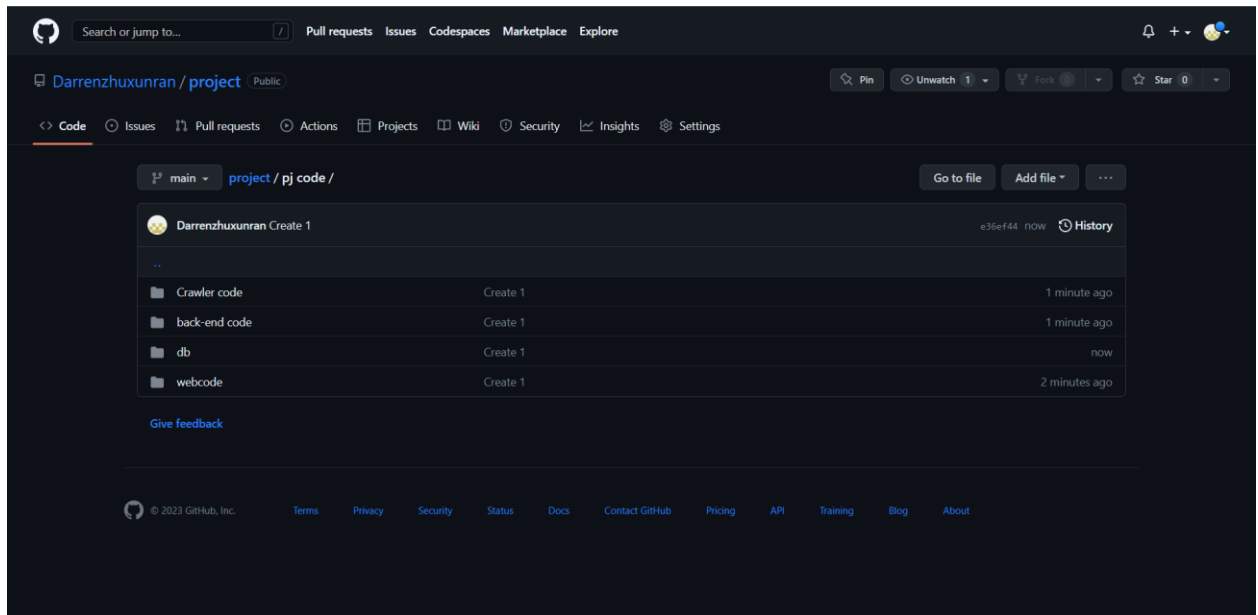
The project will use GitHub as a version management tool to manage the source code.

The steps to do this are shown below:

1. Create a separate folder called project in GitHub's Repositories for adding and storing code and files.
2. Create four subfolders in the project folder to categorize each part of the program code: crawler code, back-end code, front-end code, and database code.
3. According to the classification of the subfolders, put the corresponding code of the program into the package and then upload it.
4. After uploading the files, you will be able to find the corresponding code in GitHub's Repositories as needed to manage the project version.

The version management link is as follows:

<https://github.com/Darrenzhuxunran/project/tree/main/pi%20code>



(Figure 3: Version Management)

Chapter 4 Results

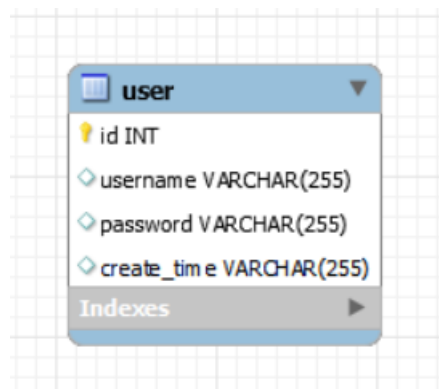
This section is a detailed description and demonstration of the project's design implementation and completed functionality. It is divided into two main sections: design and implementation and functional testing.

4.1 Design and implementation

4.1.1 Database design

According to the requirement analysis, the most basic prerequisite for a user to compare prices in the price comparison system is that the user has already performed a login operation in the system. When a user has an account, he or she can log in by entering the password directly on the login screen. If you do not have an account, you will need to register first and then continue with the login process after you have successfully registered. Therefore, you need to create a table named "user" in MySQL to store the user's account and password and the time of registration. In addition, as a price comparison system, the database should also store the information of the products to be compared, so it is necessary to create a table named "commodity" to store the product information, including product name, corresponding pictures, product description, price, etc.

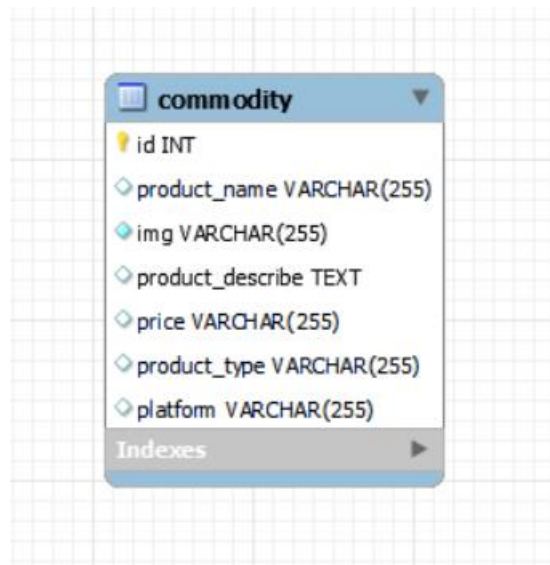
The table in the database corresponds to the ER diagram shown below:



(Figure 4: ER diagram of user table)

The 'user' table is used to store user accounts and passwords. When a user registers an account for login, the corresponding account and password will be stored in the 'user' table of the database after successful registration. In this table, the id is used as the primary key of the table and is of type INT, which is used to uniquely identify each record

and ensure the accuracy, security and integrity of the data. The username and password are stored in VACHAR, and the maximum number of characters it can store is set to 255. VACHAR is a variable-length string type, and unlike the fixed-length string type CHAR, VARCHAR only takes up the necessary storage space. In addition, when storing each piece of data, using the 'DATETIME' type will also store the creation time in the corresponding location in the table.



(Figure 5: ER diagram of commodity table)

The 'commodity' table is used to store product information. When users want to compare prices of products in this system, they first need to use crawlers to crawl and collect information on the required websites, then automatically import the 'commodity' table into the database after crawling and collecting, and finally realize the price comparison of products according to the product information in the table. Therefore, in this table, id is used as the primary key to uniquely identify each record. product_name, img, product_type, platform, and price are stored in VACHAR type, with a character limit of 255, and product_describe is stored in TEXE type, which is used to store long text data, such as article content, comments, etc. Unlike the VACHAR type, the TEXE type does not have a length limit, so it is more convenient for storing long text data.

The following is a demonstration of the data stored in the 'user' table and 'commodity' table in the database:

id	username	password	create_time
1	123456	123456	2023-01-07 20:39:40
2	root	root	2023-01-11 23:48:02
3	qqqqqq	111111	2023-04-22 22:35:39
4	darre	1233211	2023-04-22 22:36:58
5	darre	1234321	2023-04-22 22:48:19
6	darren	12344321	2023-04-22 22:50:08
7	darren1	1234567	2023-04-22 22:58:20
8	qqwwee	123321	2023-04-22 23:00:33
9	gmq123	123321	2023-04-29 14:01:11

(Figure 6: 'user' table data display)

Figure 6 details the information stored in the 'user' table, including id, username, password, and create time.

id	product_name	img	product_describe	price	product_type	platform
7	联想拯救者Y9000P	https://img12.360buyimg.com/n0/jfs/t1/18563...	啊啊啊啊啊啊	8999	电脑	京东
8	联想	https://img.alicdn.com/imgextra/i1/68910596/O...	啊啊大大大大	8999	电脑	淘宝
9	华为	https://img12.360buyimg.com/n0/jfs/t1/19767...	啊啊大大	5299	电脑	京东
10	小米13Ultra 徕卡光学全焦段四摄 第二代骁龙...	http://img11.360buyimg.com/n7/jfs/t1/168768/...	xx	5999.00	手机	京东
11	OPPO K9x 8GB+128GB 银紫超梦 天玑810 50...	http://img13.360buyimg.com/n7/jfs/t1/5010/23...	xx	1199.00	手机	京东
12	Redmi Note 11 5G 天玑810 33W Pro快充 500...	http://img12.360buyimg.com/n7/jfs/t1/175598/...	xx	999.00	手机	京东
13	华为智选 Hi nova 9z 5G全网通手机 6.67英...	http://img11.360buyimg.com/n7/jfs/t1/82881/1...	xx	1099.00	手机	京东
14	OPPO A36 6GB+128GB 晴川蓝 高通骁龙680 ...	http://img10.360buyimg.com/n7/jfs/t1/103237/...	xx	899.00	手机	京东
15	荣耀X40 120Hz OLED硬核曲面 5100mAh 快...	http://img10.360buyimg.com/n7/jfs/t1/7169/20...	xx	1599.00	手机	京东
16	Note 12 Turbo 第二代骁龙7+ 6400万像素	http://img13.360buyimg.com/n7/jfs/t1/200197/...	xx	2099.00	手机	京东
17	OPPO K10x 极光 8GB+128GB 67W超级快充 ...	http://img12.360buyimg.com/n7/jfs/t1/143292/...	xx	1299.00	手机	京东
18	荣耀80 1.6亿像素超清主摄 AI Vlog视频大师...	http://img10.360buyimg.com/n7/jfs/t1/202290/...	xx	2399.00	手机	京东

(Figure 7: 'commodity' table data display)

Figure 7 shows in detail the information stored in the 'commodity' table, including id, product name type, picture information, product description, price, product type and the platform to which the product belongs.

4.1.2 Web crawler design

Crawler design needs to meet the requirements of crawling the website platform and automatically importing the information of the website into the database table after parsing. Web crawler needs to send HTTP request first, and then use HTML parser to parse and process the HTML after the server returns the webpage content, and then extract the required data. In this project, the required data mainly includes product name, picture information source, product description, price, product type and the platform to which the product belongs. Finally, the crawler automatically imports the extracted data into the 'commodity' table in the database for storage and completes a data crawl.

The relevant design code is as follows:

```

if __name__ == '__main__':

    keyword = input('Please enter keywords:')

    page_num = eval(input('Please enter the number of pages to crawl:'))

    for page in range(1, page_num+1):

        url = "https://search.jd.com/Search?keyword=%s&enc=utf-8&page=%s"%(keyword,page*2-1)
        main(keyword, page, url)
        if page % 2 == 0:
            time.sleep(2) # Stay for 10 seconds every 2 pages

```

(Figure 8: Crawl operation)

Figure 8 shows the crawling operation of web crawler. According to the needs of users, by inputting keywords and the number of pages of the website to be crawled, crawling can be realized on the given website and corresponding data information can be obtained.

```

def get_one_html(url): # Get the html page of a page and return
    try:

        headers = {
            'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/5
        time.sleep(5)
        r = requests.get(url, headers=headers)
        # print(r.text)
        if r.status_code == 200:
            r.encoding = r.apparent_encoding
            return r.text
    except Exception as er:
        print(er)

```

(Figure 9: Gets and returns HTML pages)

Figure 9 shows the code implementation of the interface for getting and returning HTML. headers are a part of the HTTP request head, including User-Agent. Different User-agents are set to simulate different browsers to access. Setting appropriate headers can simulate browser behavior in order to prevent anti-crawler recognition and interception.

```

def get_pic_url(html): # Regularly extract key information of each page and return
    soup=bs4.BeautifulSoup(html,'html.parser')

    pic_urls = soup.find_all(attrs={'class': 'p-img'})

    # pic_urls = re.findall('"pic_url": "(.*?)"', html, re.S)
    img_url = [] # Create an empty list with links to all pictures on each page
    for one_pic_url in pic_urls:
        k=one_pic_url.find("a")
        img=one_pic_url.find("img")
        href= img.get('data-lazy-img')
        img_url.append('http:' + href)
    return img_url # Returns a list of links to pictures

```

(Figure 10: Obtain picture information)

Figure 10 shows the code implementation to get the image information. After analyzing the page structure of the target website, we determine the tags and attributes of the HTML where the images are located, and use the Requests and BeautifulSoup libraries in python to obtain the URLs of the images so as to collect the image data. Requests library can send a request to the target website to get the HTML code, and BeautifulSoup library as a tool to parse the HTML, can easily extract the tags and attributes of the image. Finally, we use python's regular expressions to get the URL of the image from the tag, and finally put the address into the corresponding location in the database to achieve the image extraction operation.

```

def get_name(html): # Regularly extract key information of each page and return
    soup=bs4.BeautifulSoup(html,'html.parser')
    pic_urls = soup.find_all(attrs={'class': 'p-name'})

    # pic_urls = re.findall('"pic_url": "(.*?)"', html, re.S)
    names = [] # Create an empty list with links to all pictures on each page
    for one_pic_url in pic_urls:
        k=one_pic_url.find("a")
        # name=one_pic_url.find("a").get("title")
        name=one_pic_url.text
        names.append(name)
    return names

```

(Figure 11: Get product name)

Figure 11 shows that the crawler finds the label of the item name by using the 'find ()' method in BeautifulSoup, and uses the 'text' attribute to retrieve the text content of the label.

```
def get_price(html): # Regularly extract key information of each page and return
    soup=bs4.BeautifulSoup(html,'html.parser')
    pic_urls = soup.find_all(attrs={'class': 'p-price'})
    prices = []
    for one_pic_url in pic_urls:
        price = one_pic_url.find("i").contents
        prices.append(price)
    return prices
```

(Figure 12: Obtaining product price)

Figure 12 shows the crawler looking for the tag of the price of the item by using the 'find ()' method in BeautifulSoup, looking for the first 'i' tag, 'contents' for all the children of the i tag and returning a list.

```
def write_to_file(page, img_urls, keyword): # Write file (download)
    i = page # Use page numbers to prevent subsequent writes from overwriting previous ones
    n = 0
    rootpath='./picture/1/'
    for pic_url in img_urls:
        pic = requests.get(pic_url)
        if not os.path.exists(rootpath):
            os.makedirs(rootpath)
            print("Directory created successfully!")
        with open(rootpath + str(i) + "-" + str(n) + '.jpg', 'wb') as f:
            f.write(pic.content)
        print('---Page {} Image {} downloaded successfully---'.format(str(i), str(n)))
        n += 1
```

(Figure 13: Data write file)

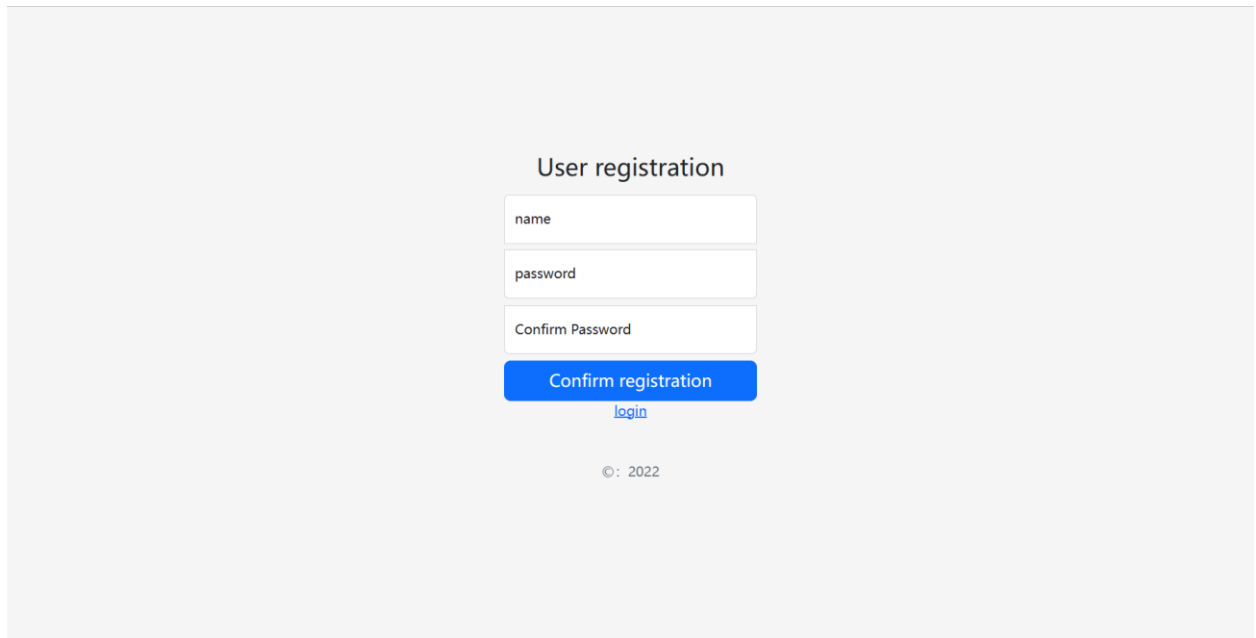
Figure 13 shows the code to implement the data write operation. Here, the 'os' library of python is used to provide an interface to access system functions, and 'f.write()' is used to write data to a file, creating a new file when it does not exist. When the file exists, the file is opened directly and the contents are added to the end of the file. The 'with open' statement is used to open the file, assign the file object to the variable "f", then call the

'f.write()' method to write the required string to the file, and finally, the 'with open' statement is used to automatically close the file to achieve successful writing of the file.

4.1.3 UI front-end design

According to the requirement analysis of the project, the front-end interface design mainly includes the registration interface, the login interface, the main product interface and the price comparison interface. In the registration interface, there should be three input boxes, namely username box, password box and password confirmation box, under which there should be a registration button. When the user clicks on the registration button after entering the user's name and the same password twice, the page should jump to the login interface. In the login interface, there should be two input boxes, one for username and one for password, and a login button at the bottom of the input box. After the user enters the username and password and clicks the login button, the page should jump to the main interface for operation. In the main product interface, there should be a function bar, and two links should be added in the function bar, one for returning to the main interface, and the other for entering the product price comparison link. At the bottom of the function bar, a filter bar should be designed to categorize the products on the main page, mainly into the categories of cell phones, computers, books and home appliances, and a button named 'select' should be set to filter the operation. In the price comparison interface, a search box needs to be designed to crawl and collect data from a given website through web crawler technology, and then click the 'search' button behind the search box after inputting the products that the user wants to compare prices on to achieve the operation of comparing prices on products.

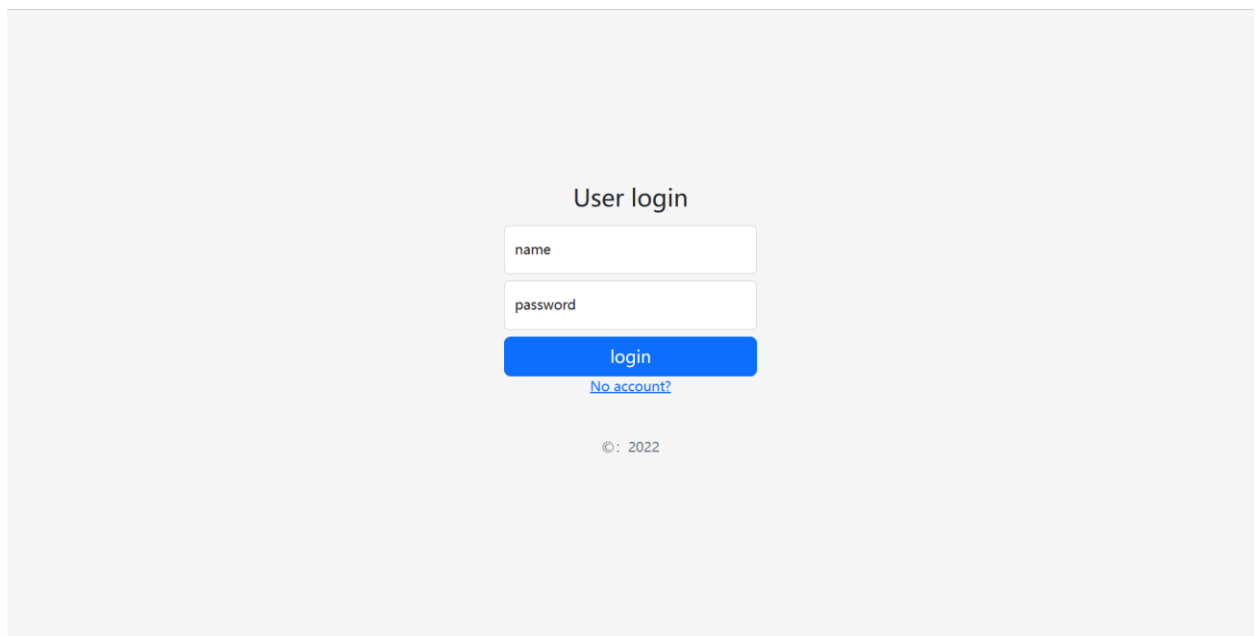
The front-end interface design is shown as follows:



The image shows a user registration form centered on a light gray background. The form is titled "User registration" in a bold, black font. Below the title are three input fields: "name", "password", and "Confirm Password". Each field has a light gray border and a small "x" icon in the top right corner. Below the "Confirm Password" field is a blue button with the text "Confirm registration" in white. Below the button is a blue link labeled "login". At the bottom of the form is a copyright notice "©: 2022".

(Figure 14: Registration interface)

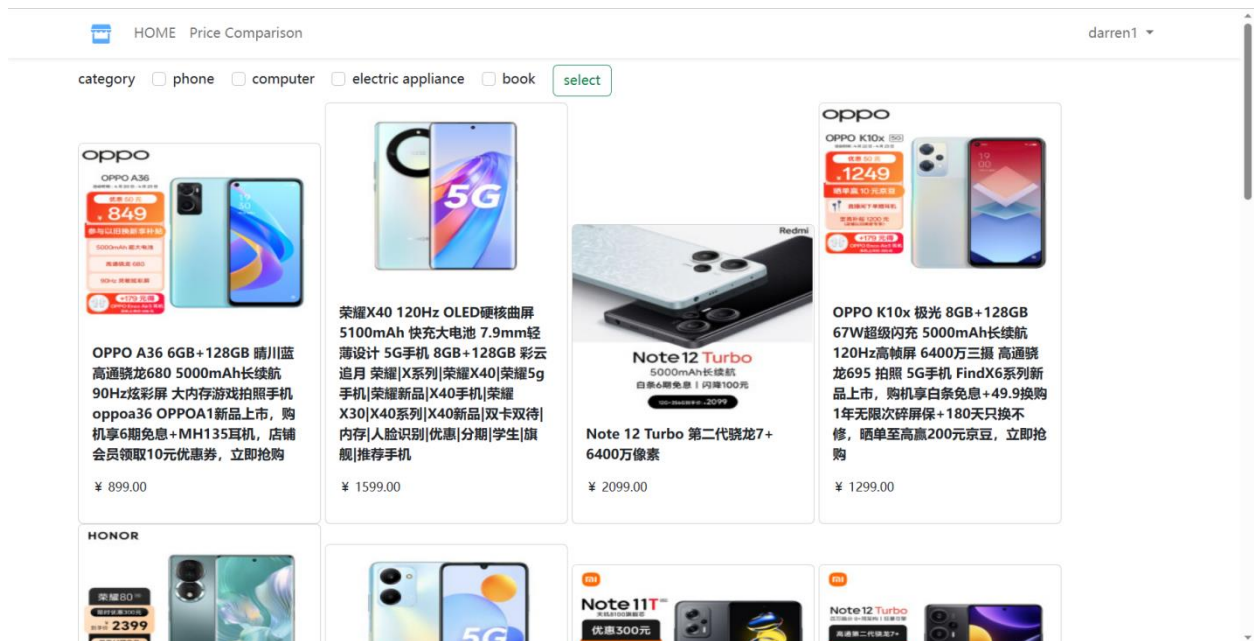
Figure 14 shows the registration interface. The user enters the user's name and password as required and then clicks 'Confirm registration' button after re-entering the password, so as to realize the registration of the user.



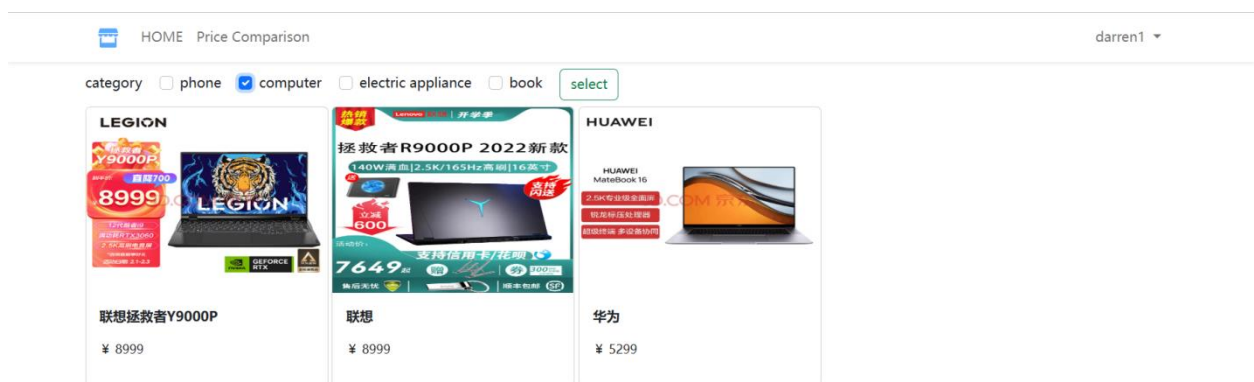
The image shows a user login form centered on a light gray background. The form is titled "User login" in a bold, black font. Below the title are two input fields: "name" and "password". Each field has a light gray border and a small "x" icon in the top right corner. Below the "password" field is a blue button with the text "login" in white. Below the button is a blue link labeled "No account?". At the bottom of the form is a copyright notice "©: 2022".

(Figure 15: Login interface)

Figure 15 shows the login interface. After the user enters the registered user name and password, click the "login" button to realize the login of the user.



(Figure 16: Main interface 1)



(Figure 17: Main interface 2)

Figures 16 and 17 are the main interface. In Figure 16, the main interface shows all the commodities in the database, including mobile phones, computers, etc., without category

screening. In Figure 17, select "computer" as the category. After filtering operation, all the computers in the database are displayed in the main interface. Not only that, in both interfaces, the name and price of the product are displayed.



(Figure 18: Price comparison interface 1)



(Figure 19: Price comparison interface 2)

Figure 18 and 19 show the interface of commodity price comparison. In Figure 18, the link from the function bar of the main interface is used to enter the price comparison interface, which is the initial price comparison interface. Users can see that there is a search box on the interface, in which the label '`<label></label>`' is used to provide label text for the input field, which can improve the readability of the form. When the user wants to compare the price of goods, input in the input box and click the "Search" button after the input box to compare the price. In Figure 19, enter the name of the commodity to be compared in the input box and click the "Search" button. The interface will respond according to the commodity information in the database and display the commodity price to be compared.

4.1.4 Back-end code design

The back-end code design is particularly important to achieve the proper operation of the project. In the back-end code implementation, the database information needs to be configured, including the address, database user name, database password and database name. To import the database driver, MySQL needs to install MySQL-connector-python, and then use the connect () function provided by the database driver to connect to the database for the purpose of operating on the database at the back end (CRUD). Connecting to the database is to meet the needs of the front-end and users, allowing data to be passed and interacted with each other between the back-end and front-end, and realizing the sharing of data between the front and back-end. After a successful database connection, you need to use 'def()' to declare several functions and define their parameters and functions, in order to create reusable blocks of code that can be called in other parts of the python program. In the Flask framework, it is also necessary to use '@app.route()' to set up the route for the purpose of letting the Flask framework know how to distribute the HTTP request to the corresponding view function.

The relevant back-end design implementation code is shown below:

```

import os
import sys
from flask import Flask

app = Flask(__name__)
sys.path.append(os.path.join(os.path.dirname(__file__)))
app.config['SECRET_KEY'] = 'AADKFADadhcadcca1231293hrn1j23rd132'

from views.index import *

if __name__ == '__main__':
    app.run(debug=True)

```

(Figure 20: Main Program)

Figure 20 shows the main program of the project, where 'if __name__ == "__main__"' is a special conditional statement that checks whether the current script is being executed as the main program and can perform actions such as test code while the script is being run as the main program.

```

# Database Configuration
HOST = 'localhost'      # Database Address
USERNAME = 'root'       # Database Username
PASSWORD = 'zxr57344'   # Database Password
DATABASE = 'compare'    # Database Name

```

(Figure 21: Database Configuration)

Figure 21 shows the information configuration code for the database. Include 'HOST', 'USERNAME', 'PASSWORD', and 'DATABASE' sections.

```

# Link Database
def dbLink():
    db = pymysql.connect(
        host=config.HOST,
        user=config.USERNAME,
        password=config.PASSWORD,
        database=config.DATABASE,
        port=3306
    )
    cursor = db.cursor(DictCursor) # Create cursor objects using db's methods
    return cursor,db

# Direct sql statements
def executionSql(sql,isSubmit=False):
    cursor,db = dbLink()
    if isSubmit:
        cursor.execute(sql) # Execute sql commands with cursor object execute() method
        db.commit() # Submit to database for execution
    else:
        cursor.execute(sql)
    res = cursor.fetchall()
    cursor.close()
    db.close()
    return res

```

(Figure 22: Database connection)

Figure 22 defines a back-end connection to the database as well as a way to execute SQL statements directly. The database connection is implemented by importing the database configuration module using the 'import' statement. In the defined method of executing SQL statements, use the cursor object 'Execute()' method to execute the SQL command, submit it to the database for execution after execution, use the 'cursor.fetchall()' statement used to get all the query results, and finally close the cursor and the database connection to free up database resources to avoid waste.

```

def CreateSql(sql):
    cursor,db = dbLink()
    cursor.execute(sql)
    db.commit()
    res = cursor.rowcount
    cursor.close()
    db.close()
    return res

def deleteSql(sql):
    cursor,db = dbLink()
    cursor.execute(sql)
    db.commit()
    res = cursor.rowcount
    cursor.close()
    db.close()
    return res

def updateSql(sql):
    cursor,db = dbLink()
    cursor.execute(sql)
    db.commit()
    res = cursor.rowcount
    cursor.close()
    db.close()
    return res

```

(Figure 23: Define database methods)

Figure 23 defines some methods in the database, mainly Create, Delete, and Update. First, a function named 'dbLink ' needs to be called to return a tuple, which is mainly used to establish a database connection and get a cursor object, then execute SQL statements through the cursor object 'cursor ' to submit the results to the database 'db', and finally, close the cursor object and the database connection.

```

import random
import time
from .db import executionSql, CreateSql

def queryTypeRandomData(s, r):
    list = ['手机', '电脑', '图书', '家电']
    dt = []
    for i in list:
        sql = f"SELECT * FROM commodity WHERE product_type = '{i}' LIMIT {s}, {r}"
        data = executionSql(sql)
        dt.extend(data)
    dts = []
    for i in list:
        sql = f"SELECT * FROM commodity WHERE product_type = '{i}'"
        data = executionSql(sql)
        dts.extend(data)
    return dt, len(dts)

def queryTypeData(ds, s=0, r=10):
    dt = []
    for i in ds:
        sql = f"SELECT * FROM commodity WHERE product_type = '{i}'"
        data = executionSql(sql)
        dt.extend(data)
    dts = []
    for i in ds:
        sql = f"SELECT * FROM commodity WHERE product_type = '{i}' LIMIT {s}, {r}"
        data = executionSql(sql)
        dts.extend(data)
    return dt, len(dts)

```

(Figure 24: Define the method of querying data)

Figure 24 defines two methods, 'queryTypeRandomData()' and 'queryTypeData()', and then creates the empty list. Next, a for loop is used to iterate through the elements of the list, a SQL query statement is spliced and 'f-string' is used mainly to format the string, the 'executeSql()' function is called to and finally use the 'extend' method, which adds the values from another sequence to the end of the original list in one go, adding the list of query results to the empty list created earlier.

```

def queryData(name,ds):
    dt = []
    for i in ds:
        sql = f"SELECT * FROM `commodity` WHERE product_name LIKE '{name}%' AND platform = '{i}' LIMIT 1"
        data = executionSql(sql)
        dt.extend(data)
    return dt

def inquire_user(username,password):
    sql = f"SELECT * FROM user WHERE username='{username}' and password = '{password}'"
    data = executionSql(sql)
    return data

def register(username,password):
    create_time = time.strftime('%Y-%m-%d %H:%M:%S')
    sql = f"INSERT INTO user(username,password,create_time) VALUES ('{username}','{password}','{create_time}')"
    CreateSql(sql)

```

(Figure 25: Define methods for inquiring and registering)

Figure 25 defines three function methods, namely 'queryData()', 'inquire_user()' and 'register()'. The 'queryData()' function is used to query the product data of a product name on a specified platform, with the parameters 'name' as the product name and 'ds' as the platform list. And, the function uses SQL statements for querying data, 'LIKE' for fuzzy matching queries, and finally returns all the results of the query in the form of a list. The 'inquire_user()' function queries the database for the existence of a user record matching the specified username and password, and returns the query result. The 'register()' function is used to implement the user registration function, while using the 'time' module to get the current time as the user registration time, call the 'CreateSql()' function to insert the registration information and time together in the database.

```

@app.route('/', methods=['GET', 'POST'])
def index():
    pagesize = 10
    data, countdata = queryTypeRandomData(0, 3)
    p = request.args.get('p')
    page = 0
    if session.get('username') != None:
        if p == None:
            data, countdata = queryTypeRandomData(0, 10)
            page = math.ceil(countdata / pagesize)
        elif p != None:
            data, countdata = queryTypeRandomData((int(p) - 1) * pagesize, int(p) * pagesize)
            page = math.ceil(countdata / pagesize)
    if request.method == 'POST':
        if session.get('username') == None:
            return redirect(url_for('Login'))
        types = request.form.getlist('type')
        data, countdata = queryTypeData(types)
        page = math.ceil(countdata / pagesize)
    return render_template('index.html', data=data, page=page)

```

(Figure 26: Set route 1)

```

@app.route('/parity', methods=['GET', 'POST'])
def parity():
    data = []
    if request.method == 'POST':
        if session.get('username') == None:
            return redirect(url_for('Login'))
        name = request.form.get('name')
        data = queryData(name, ['京东', '淘宝'])
    return render_template('parity.html', data=data)

```

(Figure 27: Set route 2)


```

@app.route('/Login', methods=['GET', 'POST'])
def Login():
    error = None
    if request.method == 'POST':
        username = request.form.get('username')
        password = request.form.get('password')
        if len(username) < 6 or len(password) < 6:
            error = 'User name or password length less than 6 bits'
        if inquire_user(username, password):
            session['username'] = username
            return redirect(url_for('index'))
        else:
            error = 'Incorrect username or password'
    return render_template('Login.html', error=error)

```

(Figure 28: Set route 3)

```

@app.route('/Register', methods=['GET', 'POST'])
def Register():
    error = None

    if request.method == 'POST':
        username = request.form.get('username')
        password = request.form.get('password')
        repassword = request.form.get('repassword')
        if len(username) < 6 or len(password) < 6:
            error = 'User name or password length less than 6 bits'
        elif password != repassword:
            error = 'Two password outputs do not match'
        elif inquire_user(username, password):
            error = 'User presence'
        elif error is None:
            register(username, password)
            return redirect(url_for('Login'))
    return render_template('Register.html', error=error)

```

(Figure 29: Set route 4)

```

@app.route('/Logout')
def Logout():
    session.clear()
    return redirect(url_for('Login'))

```

(Figure 30: Set route 5)

Figures 26, 27, 28, 29, 30, show the back-end routing functions built using the Flask framework. In Figure 26, the 'index' function is defined for coming out with routed 'GET' and 'POST' requests, and calling the 'queryTypeData()' function for querying the corresponding type of product data. In Figure 27, the main role of this part is to implement the price comparison function for the products and call the 'queryData()' for querying the price of the products on the platform. Figures 28 and 29 correspond to the user login and registration functions, respectively, with some restrictions on the characters and numbers that can be entered during login and registration, so that login and registration can only be performed when the restrictions are met. Figure 30 shows the implementation code of the user logout operation.

```

conn = pymysql.connect(host='127.0.0.1',
                        user='root',
                        password='zxr57344',
                        database='compare',
                        port=3306,
                        charset='utf8')
print("Successful connection")

cursor = conn.cursor()

def insertdata1(names,imgs,prices,num):
    print("tet")
    for i in range(num):
        name = names[i]
        img = imgs[i]
        price = prices[i]
        cursor.execute('insert into commodity (product_name, img, product_describe,price,product_type,platform) values (%s,%s,%s,%s,%s,%s)' % (name,img,name,price,"手机","京东"))

    conn.commit()
    # closeconn()

def closeconn():
    cursor.close()
    conn.close()

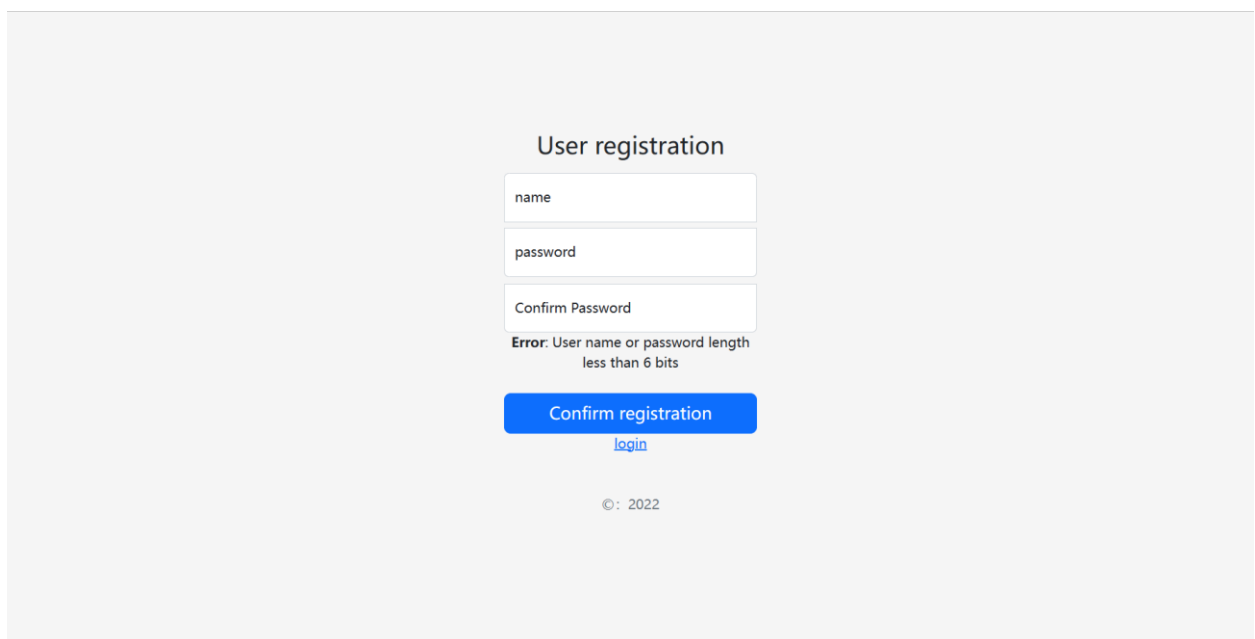
```

(Figure 31: Crawl data into the database)

Figure 31 shows the implementation code for importing the data crawled by the crawler into the database. After connecting to the database, a function named 'insertdata1()' is defined to add the data to the database, and the cursor object and database connection are closed after the operation is completed.

4.2 Functional testing

4.2.1 Registration function



The image shows a web form titled "User registration". It contains three input fields: "name", "password", and "Confirm Password". Below the "password" field, there is an error message: "Error: User name or password length less than 6 bits". At the bottom of the form, there is a blue button labeled "Confirm registration" and a link labeled "login". The footer of the page shows "©: 2022".

(Figure 32: Registration Test 1)

In Figure 32, enter the user name "darre" and enter the same password twice with "001122" to test the registration, it will cause the registration to fail with 'User name or password length less than 6 bits' character. Because "darre" only has 5 characters at this time, it does not meet the registration requirements. When the user's name is "darrenZ" and the password is the same twice and "00112" for registration test, the situation is the same as above.

The image shows a web form titled "User registration". It contains three input fields: "name", "password", and "Confirm Password". Below the "Confirm Password" field, there is an error message: "Error: Two password outputs do not match". At the bottom of the form, there is a blue button labeled "Confirm registration" and a link labeled "login". The footer of the page says "©: 2022".

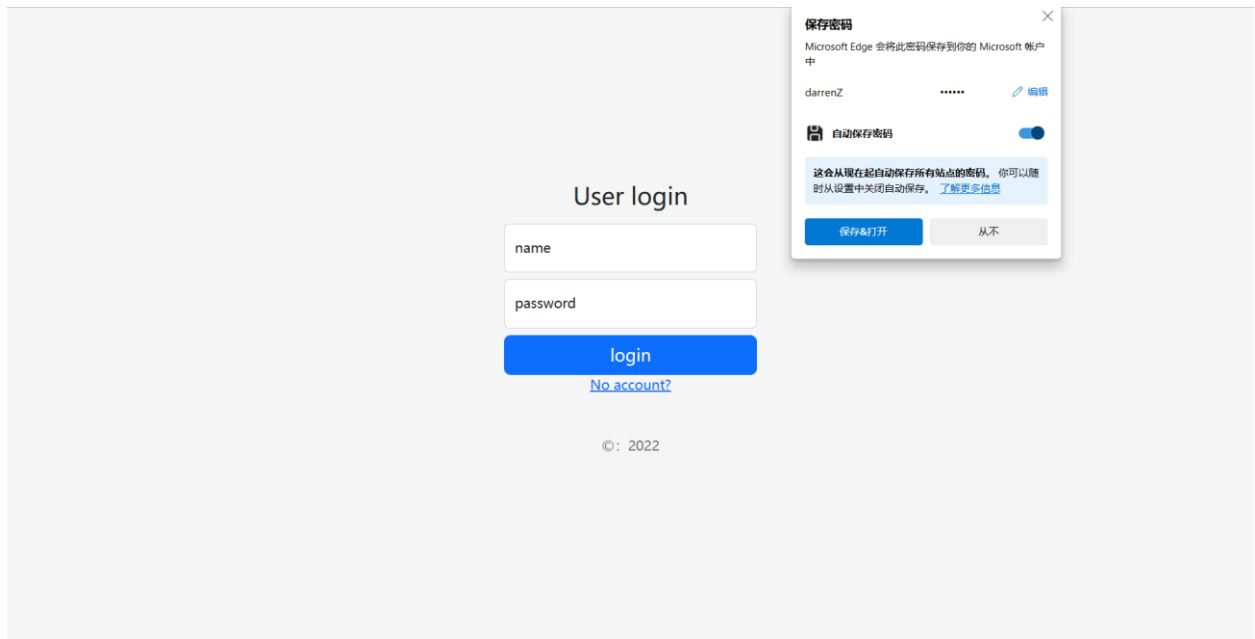
(Figure 33: Registration Test 2)

In Figure 33, enter the user name "darrenZ", enter the first password as "001122", enter the second password as "220011" or any other password different from the first time, click "Confirm registration" button to register, the registration will fail. The string 'Two password outputs do not match' appears. This is caused by the inconsistency between the two passwords entered.

The image shows a web form titled "User registration". It contains three input fields: "name", "password", and "Confirm Password". Below the "Confirm Password" field, there is an error message: "Error: User presence". At the bottom of the form, there is a blue button labeled "Confirm registration" and a link labeled "login". The footer of the page says "©: 2022".

(Figure 34: Registration Test 3)

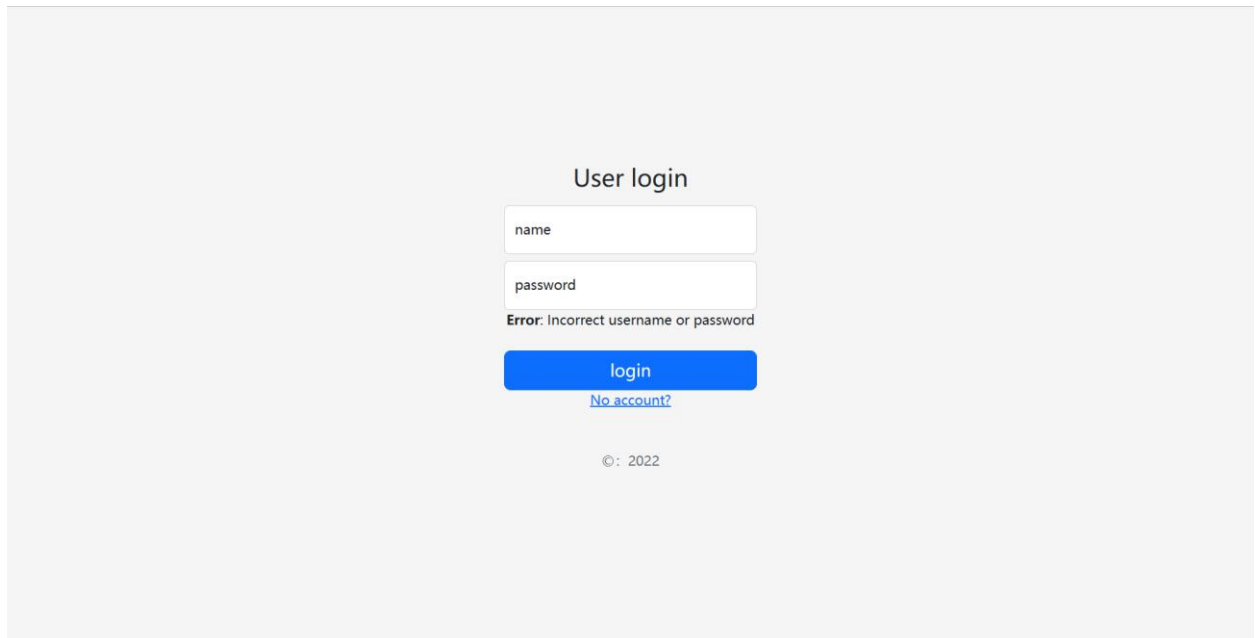
In Figure 34, use SQL statement to insert an account in MySQL Command Line Client, the account number is "qqqqqqq" and the password is "111111". After this, enter the registration screen, enter the account number as "qqqqqqq" in the username input box, enter the same password twice as "111111", click "Confirm registration" button to register, it will lead to registration failure, and display 'User presence' character prompt. This is because the account password has already been registered, cannot be registered again.



(Figure 35: Registration Test 4)

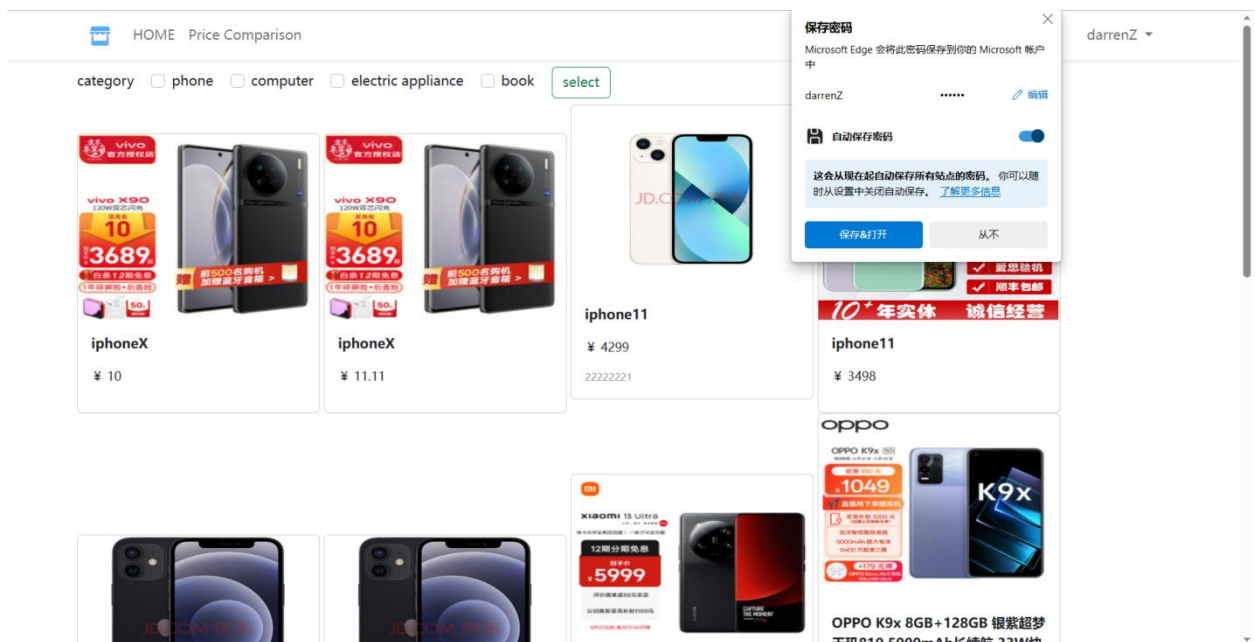
In Figure 35, according to the account registration requirements, the user's name must be at least 6 digits long and the two passwords must be the same and at least 6 digits long. According to this principle, enter the user name "darrenZ", enter the password twice as "001122", and then click the "Confirm registration "button, this will jump from the registration screen to the login screen, which means the account registration is successful.

4.2.2 Login function



(Figure 36: Login Test 1)

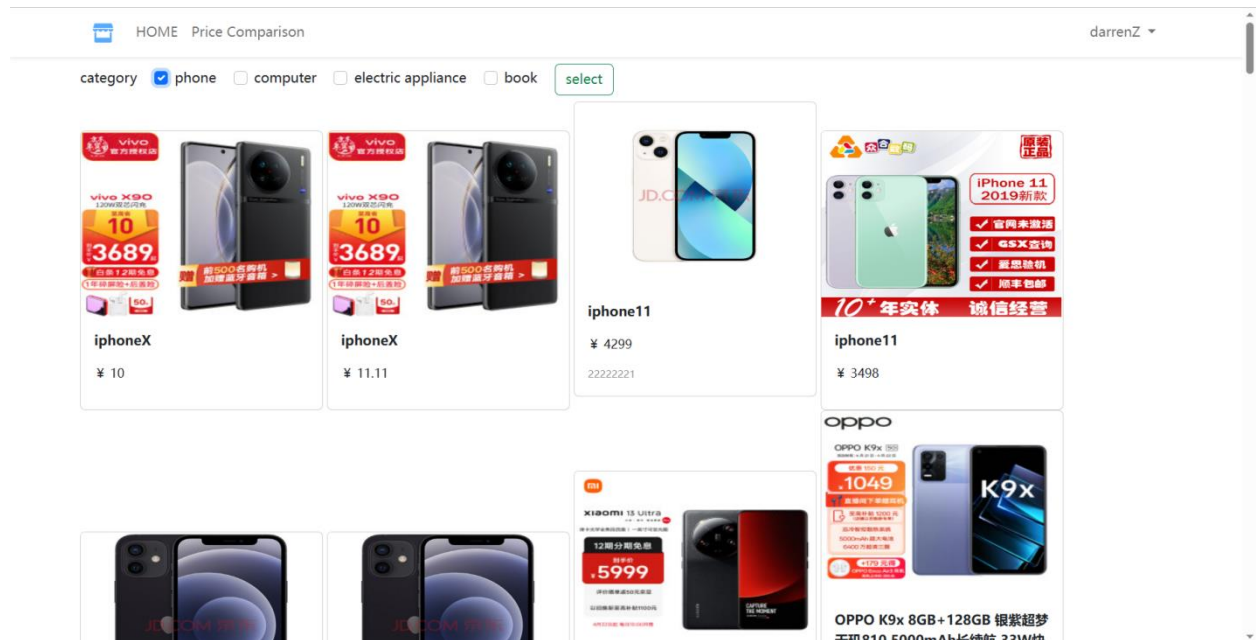
In Figure 36, the tester entered two user names and passwords, one with the username "darrenqqq" and then the password "00112233" and the other with the username "darren123", and then the password "123321". Obviously, neither of them can complete the login operation, and the word 'Incorrect username or password' is displayed at the bottom of the input box for reminder. This is because the account and password have not been registered beforehand.



(Figure 37: Login Test 2)

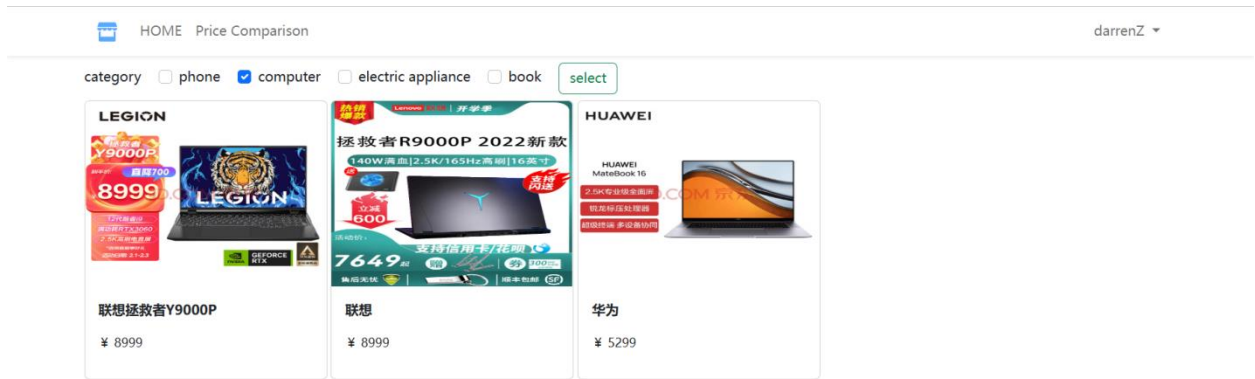
In Figure 37, the account "darrenZ" and password "001122" registered in Figure 35 are used as the input of the account and password. After the input is finished, click "login" button to jump to the main interface. If "darrenZ" is displayed in the upper right corner, the login is successful.

4.2.3 Filter function



(Figure 38: Screening Test 1)

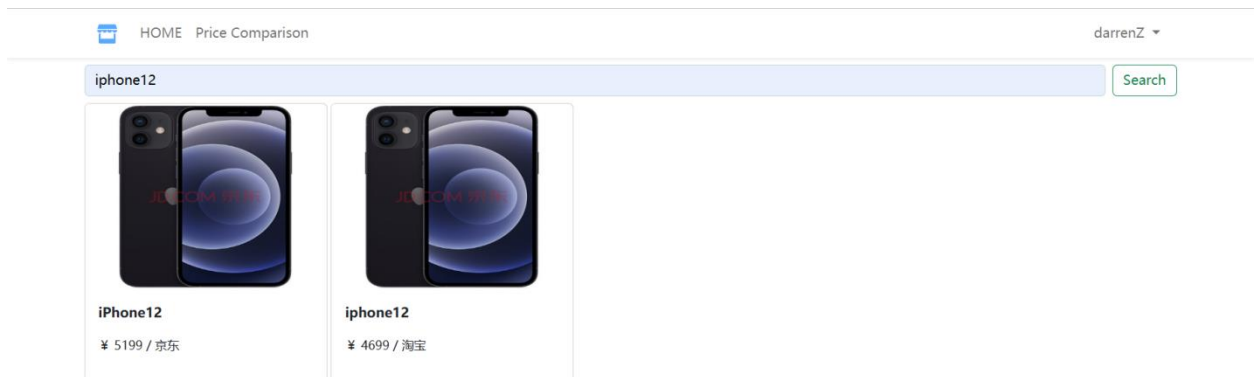
In Figure 38, after successfully logging in and entering the main interface, the testers filtered the product categories by clicking on "phone" and then clicking on the "select" button, and the system will filter the products according to the selected category.



(Figure 39: Screening Test 2)

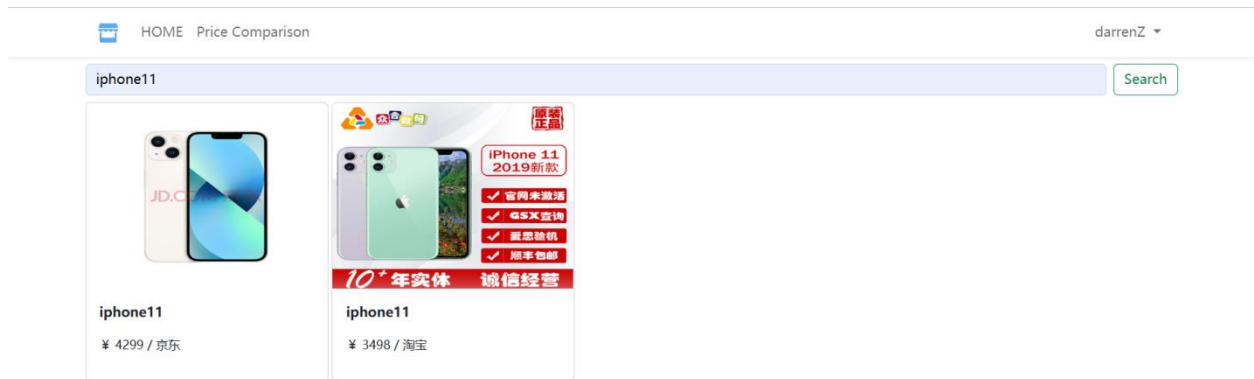
In Figure 39, the tester clicked on "computer" and then clicked on the "select" button, the system will filter the products according to the selected category, and all the products displayed here are computers.

4.2.4 Price comparison function



(Figure 40: Comparison Test 1)

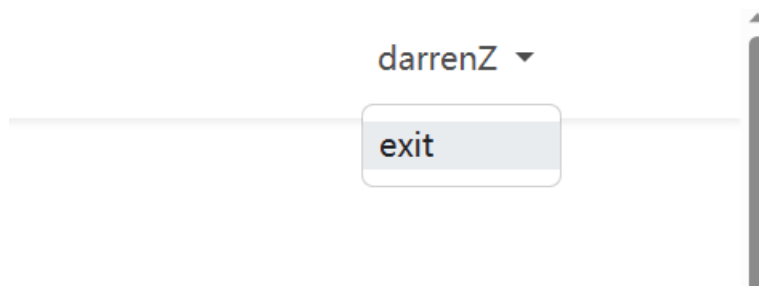
In Figure 40, click the "Price Comparison" link on the main interface to enter the price comparison interface. Enter the name of the product you want to compare in the price comparison search box, in this case "iPhone 12", and then click the 'Search' button at the end of the search box to compare prices. Finally, the interface shows the price of the same product on two different platforms, Jingdong and Taobao, and the price comparison operation is realized.



(Figure 41: Comparison Test 2)

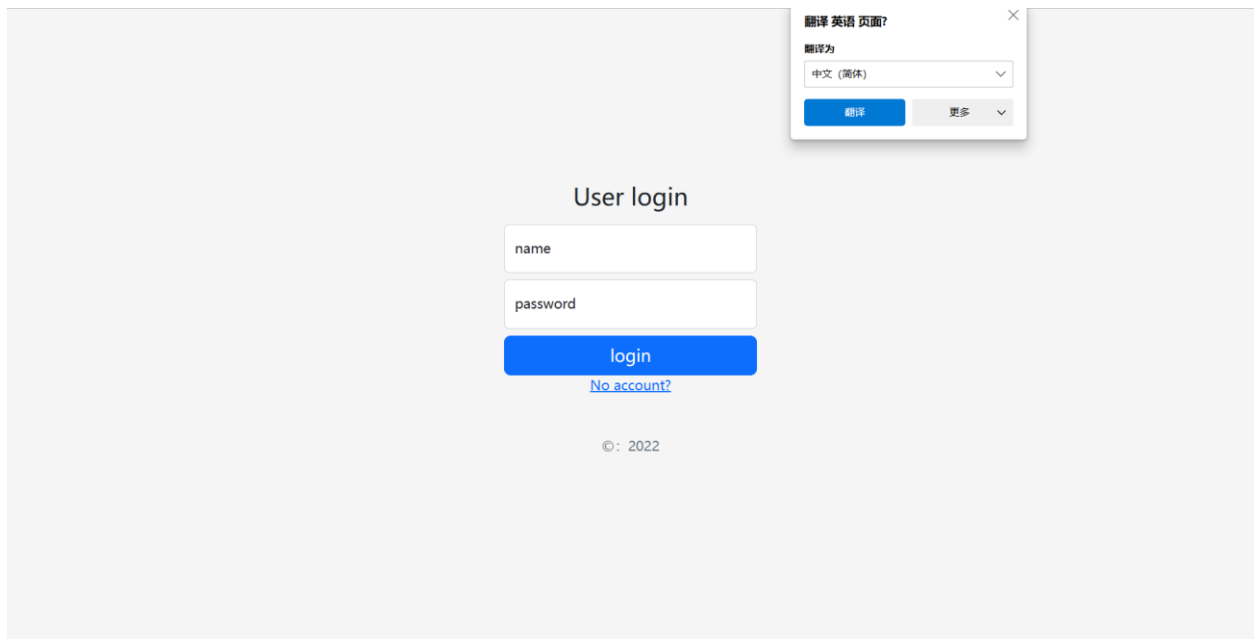
In Figure 41, enter "iPhone11" and click the 'Search' button at the end of the search box to compare prices. Finally, the price of the same product on two different platforms, Jingdong and Taobao, is displayed in the interface to achieve the price comparison operation.

4.2.5 Logout function



(Figure 42: Logout Test 1)

In Figure 42, clicking on the login account named "darrenZ" will bring up an "exit" hyperlink, which when clicked will trigger the "Logout()" function in Flask to clear the user session and redirect the user to the "Login" page.



(Figure 43: Logout Test 2)

In Figure 43, when the tester clicks on the "exit" hyperlink, it triggers the "Logout()" function in Flask, which clears the user session and redirects the user to the "Login" page, enabling the user to logout.

Chapter 5 Professional Issues

5.1 Project Management

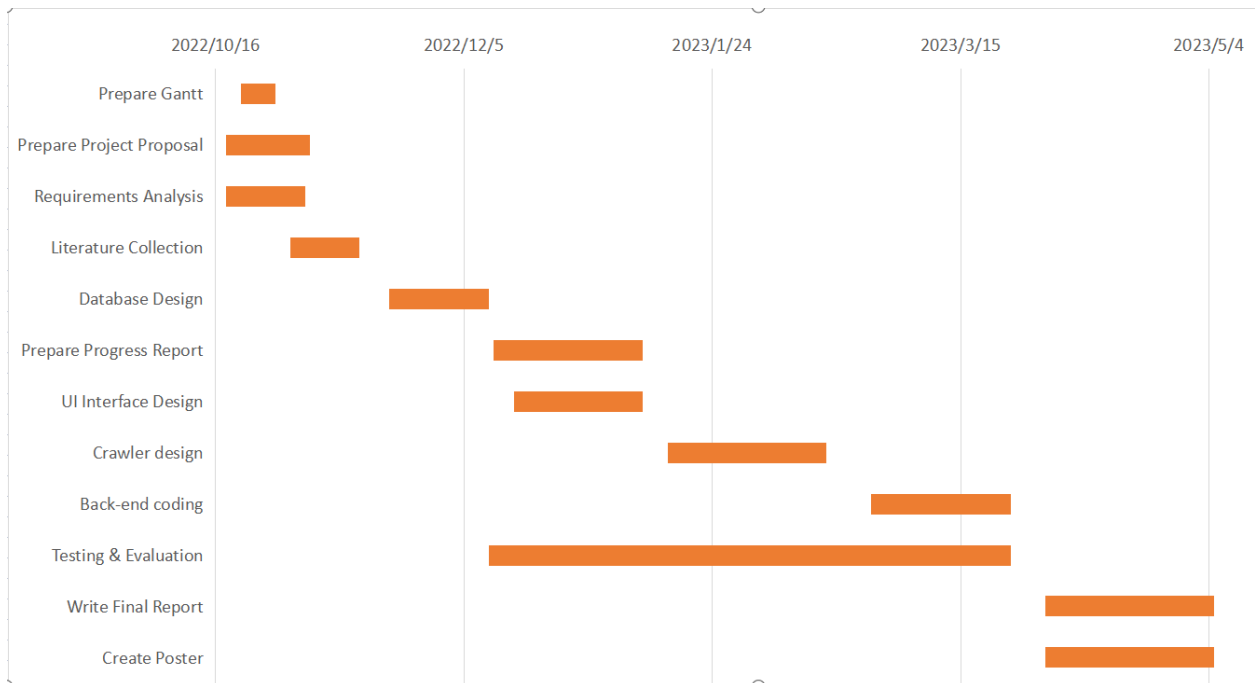
5.1.1 Activities

Objectives	Activities	Completion status
1) Background research and resource organization	Find relevant literature.	Completed
	Analyze and learn the methodological techniques used in the literature.	Completed
2) Establish project development steps	Split the project into database design, crawler, UI front-end design and back-end code design.	Completed
3) Database design	Demand Analysis.	Completed
	Conceptual structure design.	Completed
	Database implementation.	Completed
	Database testing and maintenance.	Completed
4) Design of the crawler technology	Identify crawler targets.	Completed
	Develop crawl plan and rules.	Completed
	Implementing crawling techniques.	Completed
	Testing crawler functionality and maintenance.	Completed
5) UI front-end design	Determine the design requirements.	Completed
	Develop a suitable design plan.	Completed
	Realization of the design.	Completed

	Test UI interface.	Completed
6) Back-end code design	Define design requirements.	Completed
	Develop the corresponding technical solutions, including the technologies and frameworks used.	Completed
	Write code to implement the function.	Completed
	Test code, including unit tests, system tests.	Completed
7) Realization of website development	Test that all functions work properly.	Completed
	Realization of website design and development.	Completed

(Table 1: Activities)

5.1.2 Schedule

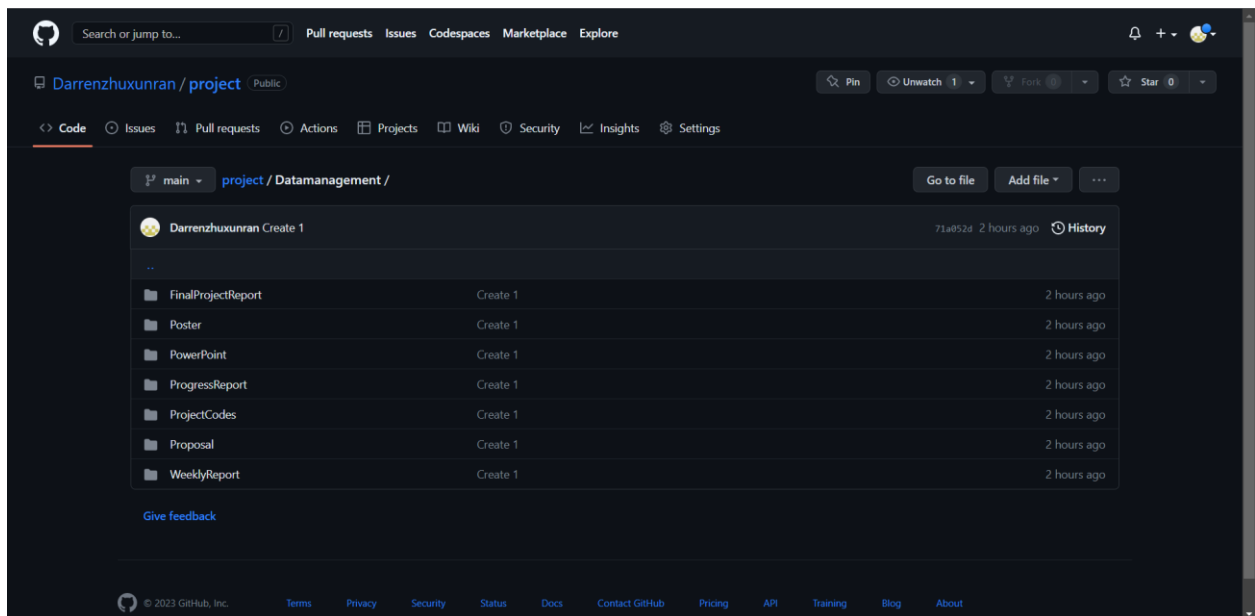


(Figure 44: Gantt Chart)

5.1.3 Project Data Management

The project uses GitHub for data storage. Create a folder named 'project' on GitHub, create a subfolder named 'Datamanagement' in that folder, and in the 'Datamanagement' folder, then create several subfolders for storing various project data, including weekly report, project proposal, progress report, project code, final report, project presentation PPT and project poster.

As shown in Figure 45 below:



(Figure 45: Project Data Management)

The project data management link is below:

<https://github.com/Darrenzhuxunran/project/tree/main/Datamanagement>

5.1.4 Project Deliverables

- 1) Project Proposal
- 2) Project Weekly Report
- 3) Project Progress Report
- 4) Project Progress Video

- 5) Project Final Report
- 6) Project Code
- 7) Project presentation PPT
- 8) Project Poster

5.2 Risk Analysis

This section focuses on the analysis of the potential risks of the project. It includes the potential causes of project risks, the severity of the risk, the likelihood of the risk occurring, the impact of the risk on the project, and the appropriate opportunities or plans to be taken to address or reduce the corresponding risk. It also presents the possible risks in the form of a table.

Potential Risk	Potential Causes	Severity	Likelihood	Risk	Mitigation
Loss of data	Loss of relevant documents	2	3	6	Look for project related literature again, and store and manage it after collection, and make cloud backup.
	Related code loss	4	3	12	Re-design the coding, after each part of the coding is completed in the local data storage and management

					and cloud backup.
Software bugs	Incomplete test plans	3	3	9	Develop a comprehensive test plan
	Technical defect	4	4	16	Look up materials, learn coding techniques, and make touch-up changes to the code when the basic functionality is complete.
Feature creep	The expectations for the project were large and wanted to implement many unique features.	1	2	2	Define the core features and requirements of the product during project development, strictly control the addition of new features, and conduct regular product reviews and optimization.

(Table 2: Risk Analysis)

Likelihood

Severity		1	2	3	4	5
	1	1	2	3	4	5
	2	2	4	6	8	10
	3	3	6	9	12	15
	4	4	8	12	16	20
	5	5	10	15	20	25

(Table 3: Risk Severity Matrix)

KEY	Meaning	Description
Green	Low Risk	There is little impact on the development of the project, and even if it happens, these risks can be easily addressed.
Yellow	Medium Risk	The impact on the development of the project is more obvious, but not devastating, and requires some time and effort to correct and recover.
Red	High Risk	There is a serious threat to the development of the project, which can sometimes be fatal and irreversible. It takes a long time to recover or is impossible to recover.

(Table 4: Risk KEY)

5.3 Professional Issues

This project is a web-based online shopping platform price comparison system. In this system, although it is necessary to use web crawler technology to crawl a given website to obtain data, after crawling, it will enter a sleep state and will not affect the normal access speed of other users due to the overload of the website's server caused by the crawling operation. After users register their accounts and passwords, they will only be used for the login operation of the project and will not obtain information about other platforms for profit based on the user's registration information. The social issue of this project is mainly the protection and handling of user data. The leaked data may be used to give "interested parties" access to more information. In terms of ethics, the project refers to a professional code of conduct that meets the requirements and does not create any legal, social, ethical or environmental problems.

Chapter 6 Conclusion

In this section, there are two main parts, one is a summary of this project, and the other is to present the problems of the project and the prospect of the future.

6.1 Summary

This project is a web-based online shopping platform price comparison system. In this project, Python is used as the development language, BeautifulSoup as the crawler framework, MySQL as the database, Flask as the back-end framework, and Bootstrap based on HTML, CSS and JavaScript as the front-end framework. The various parts are linked through the back-end code on the PyCharm compiler to finally form a complete price comparison system. The main function of this price comparison system is to compare the price of the same product from two different platforms through fuzzy matching, in order to facilitate consumers to make a more correct decision based on the price comparison results when shopping online. The project begins with a description of the background and audience of the price comparison system in the first chapter. In Chapter 2, the authors analyze and compare the various technologies used to implement the price comparison system, and briefly summarize the advantages and disadvantages of each technology. The Chapter 3 provides a detailed description of the design and development of the price comparison system, including the methods and techniques used, and provides a project versioning scheme. In Chapter 4, the completed design is tested and the corresponding test diagrams are attached after the test results are completed, in which the system achieves all the functions well and realizes the operation of price comparison of goods. In Chapter 5, it focuses on the success of the development project, provides a Gantt chart of the entire project development, and explains and analyzes the potential risks in the development as well as social ethics and morality. Overall, the project features have all been implemented to meet the development design requirements and achieve the developer's intended purpose.

6.2 Prospect of the future

Although the basic functions of the price comparison system have been realized, there are still some deficiencies that need to be improved. In this system, the ranking of goods is not handled efficiently. When web crawler technology is used to crawl web pages, the obtained data can only be displayed on the front-end interface after being imported into

the database. However, the ascending and descending order of price and sales volume cannot be realized on the front-end interface. Moreover, the lack of price comparison platforms is one of the drawbacks of the system. There are only two platforms for commodity price comparison in this project, which may lead to the selection of relatively cheap goods on these two platforms, but not the lowest price among all online shopping platforms. Therefore, based on these defects, we should continue to develop and upgrade towards these goals in the future, so that the function of the price comparison system becomes more perfect and powerful. At the same time, we should also devote ourselves to the development of mobile client in the future. With the continuous development of mobile Internet, the development of price comparison system in mobile client is particularly important. Data security and privacy protection should also be emphasized, and developers should take effective measures to protect the security and privacy of user data.

References

- [1] S. Y. Chung and C. Park, *The 11th International Conference on Advanced Communication Technology : ubiquitous ICT convergence makes life better! : ICACT 2009 : Phoenix Park, Korea, Feb. 15-18, 2009 : proceedings.*
- [2] Q. Guo, W. Yu, Z. Ma, and J. Kong, "User Behavior Monitoring Mechanism of Online Shopping Processes based on Hierarchical Colored Petri Net," in *Proceeding - 2021 China Automation Congress, CAC 2021*, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 6118–6122. doi: 10.1109/CAC53003.2021.9728033.
- [3] S. Sharma and P. Gupta, "The anatomy of web crawlers," in *International Conference on Computing, Communication and Automation, ICCCA 2015*, Institute of Electrical and Electronics Engineers Inc., Jul. 2015, pp. 849–853. doi: 10.1109/CCAA.2015.7148493.
- [4] Z. Shi, M. Shi, and W. Lin, *ACIT-CSII-BCD 2016 4th International Conference on Applied Computing and Information Technology (ACIT 2016), 3rd International Conference on Computational Science/Intelligence and Applied Informatics (CSII 2016), 1st International Conference on Big Data, Cloud Computing, Data Science & Engineering (BCD 2016) : 12-14 December 2016, Las Vegas, Nevada, USA : proceedings.*
- [5] A. Vadivel, S. S.G, R. D. Mahalakshmi, and J. Karthika, *2012 International Conference on Advances in Engineering, Science and Management.* IEEE, 2012.
- [6] F. Zhou and Y. Wang, "Exploring The Role of Web Crawler and Anti-Crawler Technology in Big Data Era," in *Proceedings - 2022 11th International Conference of Information and Communication Technology, ICTech 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 316–319. doi: 10.1109/ICTech55460.2022.00070.
- [7] Z. Haitao, "Research on Mobile Web Front-end Design Based on HTML5 Technology," in *Proceedings - 2021 6th International Symposium on Computer and Information Processing Technology, ISCIPT 2021*, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 290–293. doi: 10.1109/ISCIPT53667.2021.00065.
- [8] X. Yan, L. Yang, S. Lan, and X. Tong, "Application of HTML5 multimedia," in *Proceedings - 2012 International Conference on Computer Science and Information Processing, CSIP 2012*, 2012, pp. 871–874. doi: 10.1109/CSIP.2012.6308992.
- [9] A. Das and S. S. Devgan, "Implementation of streaming audio and video for online web based courses using mysql database server and PHP," in *Conference Proceedings - IEEE SOUTHEASTCON*, IEEE, 2000, pp. 523–526. doi: 10.1109/secon.2000.845625.
- [10] L. Ling, "The Design of Multifunctional Online Travel Service Management Platform and the Implementation of MySQL," in *4th International Conference on Inventive Research in Computing Applications, ICIRCA 2022 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 1293–1296. doi: 10.1109/ICIRCA54612.2022.9985607.
- [11] S. Pendse *et al.*, "Oracle database in-memory on active data guard: Real-time analytics on a standby database," in *Proceedings - International Conference on Data Engineering*, IEEE Computer Society, Apr. 2020, pp. 1570–1578. doi: 10.1109/ICDE48307.2020.00139.

- [12] I. S. Vershinin and A. R. Mustafina, "Performance Analysis of PostgreSQL, MySQL, Microsoft SQL Server Systems Based on TPC-H Tests," in *Proceedings - 2021 International Russian Automation Conference, RusAutoCon 2021*, Institute of Electrical and Electronics Engineers Inc., Sep. 2021, pp. 683–687. doi: 10.1109/RusAutoCon52004.2021.9537400.
- [13] P. Vogel, T. Klooster, V. Andrikopoulos, and M. Lungu, "A Low-Effort Analytics Platform for Visualizing Evolving Flask-Based Python Web Services," in *Proceedings - 2017 IEEE Working Conference on Software Visualization, VISSOFT 2017*, Institute of Electrical and Electronics Engineers Inc., Oct. 2017, pp. 109–113. doi: 10.1109/VISSOFT.2017.13.
- [14] S. Goel, M. Bansal, A. K. Srivastava, and N. Arora, *Proceedings of the Third International Conference on Electronics, Communication and Aerospace Technology (ICECA 2019) : 12-14, June 2019*.
- [15] Y. Wang, "Research on Python Crawler Search System Based on Computer Big Data," in *2023 IEEE 3rd International Conference on Power, Electronics and Computer Applications (ICPECA)*, IEEE, Jan. 2023, pp. 1179–1183. doi: 10.1109/ICPECA56706.2023.10075835.

Appendices