RETAIL E-COMMERCE PRICE TRACKER

A Project Report

Submitted by

Ketan Kale	**112103064**
Nikhil Kokale	**112103072**
Atharva Lonhari	**112103079**

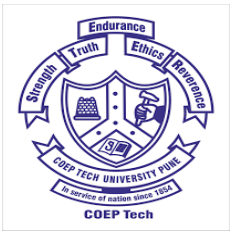
of

TY (Computer Engineering)

Under the guidance of

Dr. Tanuja R. Pattanshetti

COEP Technological University



DEPARTMENT OF COMPUTER ENGINEERING

COEP Technological University

April, 2024

DEPARTMENT OF COMPUTER ENGINEERING

COEP Technological University

CERTIFICATE

Certified that this project, titled "RETAIL E-COMMERCE PRICE TRACKER" has been successfully completed by

Ketan Kale	**112103064**
Nikhil Kokale	**112103072**
Atharva Lonhari	**112103079**

and is approved for the fulfilment of the requirements of "Software Engineering Mini Project- Stage II".

SIGNATURE

Dr. Tanuja R. Pattanshetti

Project Guide

Department of Computer Engineering

COEP Technological University,

Shivajinagar, Pune - 5.

Abstract

The Retail E-Commerce Price Tracker project aims to develop a comprehensive web application that empowers users to track and compare prices of electronic items across multiple online retailers. In a dynamic and competitive online marketplace, consumers often face challenges in finding the best deals and making informed purchasing decisions. The Retail E-Commerce Price Tracker addresses this need by leveraging web scraping techniques to collect real-time pricing data from various online retailers and presenting it through a user-friendly interface.

The project methodology involves thorough requirements analysis, technology selection, system design, implementation, and testing. Key features of the application include price comparison, price tracking with alerts, deal discovery, historical price analysis, product research, budget planning, and market analysis.

Through the implementation of the Retail E-Commerce Price Tracker application, users can efficiently navigate the electronic marketplace, compare prices, monitor price trends, and receive alerts on price drops, ultimately enabling them to make informed purchasing decisions and maximize savings. The project contributes to enhancing the overall shopping experience for consumers and provides valuable insights for retailers and industry analysts.

The outcomes of the project highlight the effectiveness of the Retail E-Commerce Price Tracker application in addressing the challenges faced by consumers in the electronic marketplace. By providing a centralized platform for price comparison and analysis, the application empowers users to make informed decisions and optimize their shopping experience.

Contents

List of Figures

Chapter 1

Synopsis

1.1 Project Title

RETAIL E-COMMERCE PRICE TRACKER

1.2 Internal Guide

Dr. Tanuja R. Pattanshetti

1.3 Problem Statement

The Electronic Price Tracker project addresses the need for a centralized platform to track and compare prices of electronic items across multiple online retailers. It aims to provide users with real-time pricing information and personalized alerts on price drops, enhancing their ability to make informed purchasing decisions and maximize savings. The project seeks to empower consumers in navigating the electronic marketplace efficiently and confidently, ensuring they secure the best possible prices for their desired products.

1.4 Plan of Project Execution

Task	Start Date	End Date	Duration	Resources
Problem statement finalization	08-Jan-24	15-Jan-24	8	Atharva,Ketan,Nikhil
Project Plan	16-Jan-24	31-Jan-24	16	Atharva
Requirement Analysis	01-Feb-24	05-Feb-24	5	Ketan,Atharva
Functional Flow and Screens Identification	06-Feb-24	09-Feb-24	4	Nikhil,Atharva
Creating Mocks for each of the Screens Identified	10-Feb-24	13-Feb-24	4	Ketan
Web Scraping POC	13-Feb-24	18-Feb-24	6	Nikhil
Database Modelling	19-Feb-24	29-Feb-24	10	Atharva
Data population using Web scraping	01-Mar-24	10-Mar-24	10	Atharva,Nikhil
Price comparision functionality	11-Mar-24	20-Mar-24	10	Atharva,Ketan,Nikhil
Frontend	21-Mar-24	31-Mar-24	10	Ketan,Nikhil
Testing	01-Apr-24	10-Apr-24	10	Atharva,Ketan,Nikhil
Documentation and project report	11-Apr-24	15-Apr-24	5	Atharva,Ketan,Nikhil

Chapter 2

Problem Definition and scope

2.1 Problem Definition

2.1.1 Goals and objectives

Goal and Objectives:

- Goals
  - Develop a software solution that empowers consumers to find the best product deals and save money on their online purchases.
- Objectives
  - **Aggregate product pricing data:** Collect and organize product pricing information from a wide range of online retailers in a timely and accurate manner.
  - **Enable comprehensive price comparison:**Develop algorithms to compare prices across retailers.
  - **Offer personalized deal recommendations:**Recommend the best deal to each user based on their customizable preferences and priorities.
  - **Provide price drop alerts:**Monitor price changes for specific products and send timely alerts to users when desired price drops occur. ‘

- **Deliver a user-friendly interface:**Create an intuitive and accessible user interface that facilitates easy product search, comparison, etc.

2.1.2 Statement of scope

- **Target Products:** Laptops, Smartphones, T.V.s, Refrigerators
- **Target Websites:** Vijay Sales, Croma, Reliance Digital, Flipkart
- **Target Audience:** This application is ideal for consumers who are looking to save money on their electronics purchases. It will be particularly beneficial for individuals who frequently buy or research electronic products

2.2 Software context

The Retail E-Commerce Price Tracker project operates within the context of web application development and data aggregation. It involves the integration of web scraping techniques to gather real-time pricing information from various online retailers. The project will utilize technologies such as HTML, CSS, and JavaScript for front-end development to create a user-friendly interface. On the back end, the application will be developed using programming languages like Python, along with frameworks such as Flask, to handle data processing, storage, and user authentication.

Additionally, the project may leverage libraries and tools specific to web scraping, such as BeautifulSoup, to extract data efficiently from retailer websites. Database management systems like MySQL may be employed to store and manage the collected data.

Overall, the software context of the Electronic Price Tracker project encompasses

web development, data aggregation, database management, and cloud computing technologies, all aimed at creating a robust and efficient platform for tracking and comparing electronic item prices across online retailers.

2.3 Major Constraints

- **Data acquisition**
  - **Website structure and its changes:** Adapting to variations in website layouts and data formats.
  - **Volume of data:** Handling large number of websites.
  - **Ethical Scraping:** The techniques must comply with website terms of service and keep in mind data privacy regulations.
- **Client-side constraints**
  - **Offline users:** Users not being logged in to the application.
  - **Synchronization of Data:** We need to ensure the consistency of client-side and server-side data and ensure they are synchronized.
- **Data processing**
  - **Optimization of performance:** We need to process large datasets effectively without compromising speed.

2.4 Outcome

The primary outcome of the Retail E-Commerce Price Tracker project is the successful development and deployment of a comprehensive web application that empowers users

to track and compare prices of electronic items across multiple online retailers. Key components of the outcome include:

- **User-Friendly Interface:** A user-friendly interface that enables users to easily navigate the application, search for electronic items, and compare prices from various online retailers. The interface is intuitive and visually appealing.
- **Real-Time Pricing Information:** The application provides users with up-to-date pricing information from multiple online retailers, ensuring accuracy and reliability. Data collected through web scraping techniques is regularly updated to reflect changes in pricing.
- **Personalized Alerts:** The ability for users to set up personalized alerts for price drops on specific electronic items of interest. These alerts are timely, notifying users via email when prices decrease on selected items.
- **Efficient Data Aggregation:** Efficient data aggregation and processing mechanisms to collect pricing information from various online retailers seamlessly.
- **Enhanced User Experience:** The outcome results in an enhanced user experience, enabling users to make informed purchasing decisions and maximize savings in the electronic marketplace.

2.5 Applications

- **Comparison Shopping:** The Retail E-Commerce Price Tracker application enables users to compare prices of electronic items across multiple online retailers, facilitating informed purchasing decisions and ensuring they obtain the best possible prices.

- **Price Tracking and Alerts:** Users can track the prices of specific electronic items over time and set up personalized alerts for price drops. This feature helps users stay informed about cost-saving opportunities and capitalize on favorable price changes.
- **Budget Planning:** The application can be used as a budget planning tool, allowing users to monitor prices of electronic items and plan their purchases accordingly. By tracking price trends and setting budget thresholds, users can optimize their spending and maximize savings.
- **Deal Hunting:** Users can leverage the application to hunt for deals and discounts on electronic items, taking advantage of price drops and promotional offers from various online retailers. This can lead to significant cost savings on desired products.
- **Market Analysis:** The aggregated pricing data collected by the application can be utilized for market analysis purposes. Retailers and industry analysts can analyze pricing trends, demand patterns, and competitive landscape to gain insights into market dynamics and consumer behavior.
- **Price Monitoring for Retailers:** Retailers can use the application to monitor prices of their own products as well as competitors' prices, enabling them to adjust pricing strategies and stay competitive in the market.
- **Educational Purposes:** The application can be used for educational purposes, such as teaching students about web scraping techniques, data aggregation, and market analysis in the context of e-commerce and online retailing.

2.6 Software Resources Required

1. **Database Management System (DBMS):** A relational database management system (RDBMS) such as MySQL is required to store and manage the application's

data, including pricing information and user accounts.

2. **Web Framework:** A web framework is necessary for developing the backend logic of the Retail E-Commerce Price Tracker application.

3. **Web Scraping Tools:** Software tools and libraries for web scraping are required to collect real-time pricing data from various online retailers.

4. **Programming Languages:** Programming languages such as Python, JavaScript, HTML, CSS, and SQL are essential for developing different components of the Retail E-Commerce Price Tracker application, including backend logic, frontend interfaces, and database queries.

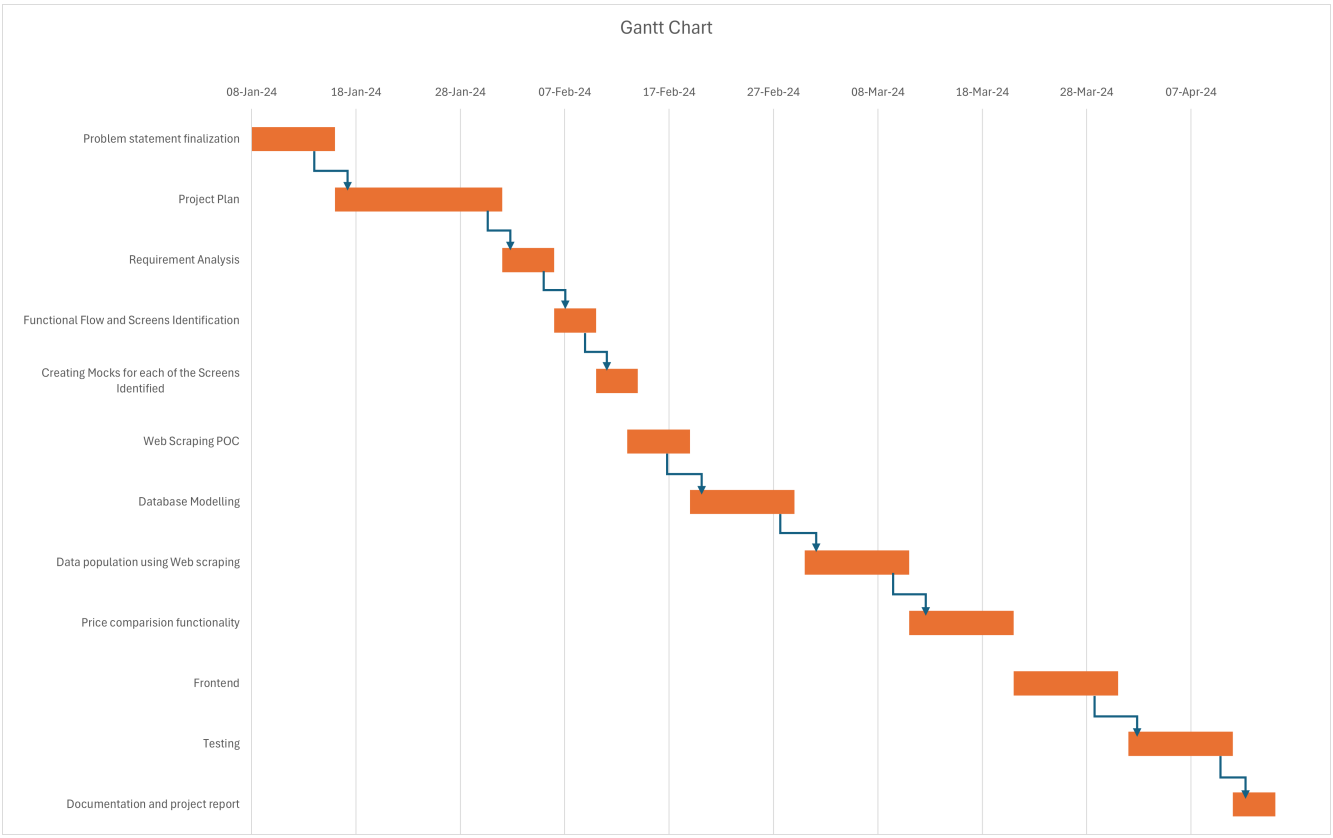
5. **Version Control System:** Version control software such as Git is necessary for managing the application's source code, tracking changes, and collaborating with team members.

Chapter 3

Project Plan

3.1 Project Schedule

3.1.1 Gantt Chart



Chapter 4

Software requirement specification

4.1 Introduction

4.1.1 Traceability Matrix

Project Name – Retail E-Commerce Price Tracker					
Business Requirements Document		Functional Requirements Document			Test Case Document
Business Requirement ID	Business Requirements/Business Use Case	Functional Requirement ID	Functional Requirement/Use Case	Priority	Test Case ID
BR_1	User Authentication	FR_1	User Registration/Signup	High	TC_1
		FR_2	Secure Login authentication	High	TC_2
BR_2	Comprehensive product search	FR_3	Search from dropdown	High	TC_3
		FR_4	Search by category	Medium	TC_4
BR_3	Security and Privacy	FR_5	Security measures to protect user information	High	TC_5
		FR_6	Ethical scraping practices	High	TC_6

Figure 4.1: Traceability1

BR_4	Data aggregation, comparison and population	FR_7	Product Information Storage (in a relational database)	High	TC_7
		FR_8	Data scraping from multiple websites (in suitable format)	High	TC_8
		FR_9	Dynamic Updating after some regular intervals	Mediu m	TC_9
BR_5	User-friendly web interface	FR_10	Dropdown for product listing	Mediu m	TC_10
		FR_11	Display of product information in an organized and appealing way	High	TC_11
BR_6	Showing Detailed Price Information	FR_12	Including all necessary information	High	TC_12
BR_7	Linking to Original Website	FR_13	Redirecting user to the particular website when clicked on that	High	TC_13

Figure 4.2: Traceability2

			particular component of that website		
		FR_14	Ensure lag-free transition	Low	TC_14
BR_8	Wishlist and price dropping alerts	FR_15	Enable users to select certain products they wish to have a reduced price	Low	TC_15
		FR_16	Send alert notification to their mails once price drops below certain threshold	Low	TC_16

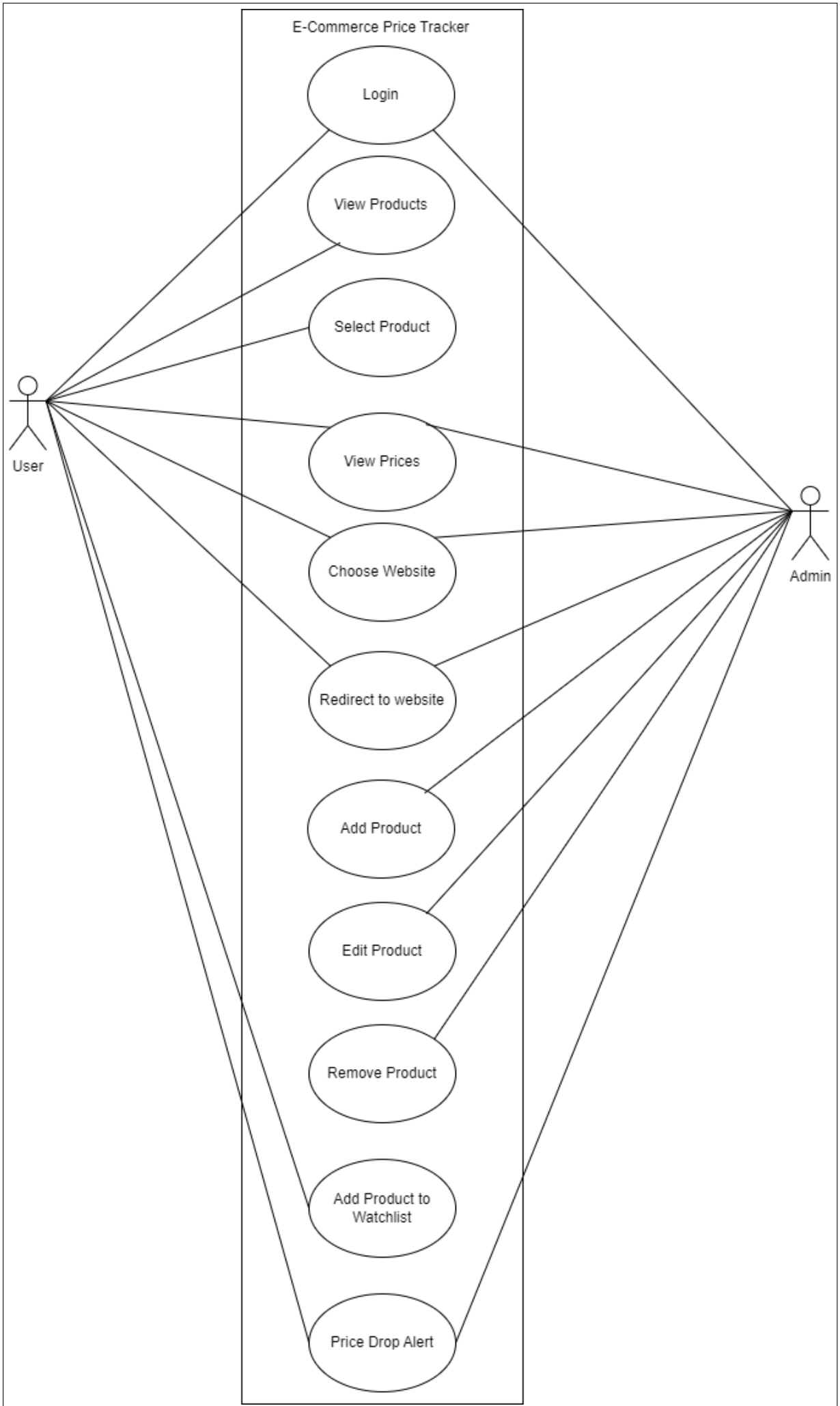
Figure 4.3: Traceability3

4.1.2 Use-cases

- **Price Comparison:** Users compare prices of electronic items from multiple retailers.
- **Price Tracking:** Users track item prices and receive alerts for drops.
- **Deal Discovery:** Users find discounts and promotions on electronics.
- **Product Research:** Users compare specs, reviews, and prices.
- **Budget Planning:** Users set budgets and monitor prices accordingly.
- **Market Analysis:** Retailers analyze pricing trends for competitiveness.

4.1.3 Use Case View

Use Case Diagram:


E-Commerce Price Tracker
Login
View Products
Select Product
View Prices
Choose Website
Redirect to website
Add Product
Edit Product
Remove Product
Add Product to Watchlist
Price Drop Alert
User
Admin

4.2 Data Model and Description

4.2.1 Data objects and Relationships

- **User:**Represents a registered user of the Retail E-Commerce Price Tracker application. Users can log in, authenticate, and access personalized features such as price tracking and watchlists.
- **Product:**Represents an electronic item available for tracking and comparison within the application. Product details provide users with information about the items they are interested in purchasing.
- **Price:**Stores pricing information for different products from various online retailers. Prices are regularly updated through web scraping techniques to provide users with real-time pricing data.
- **Watchlist:**Allows users to monitor specific products for price drops. Users can add products to their watchlists and receive alerts when prices fall below their specified target prices.

Entity Relationship Diagram:

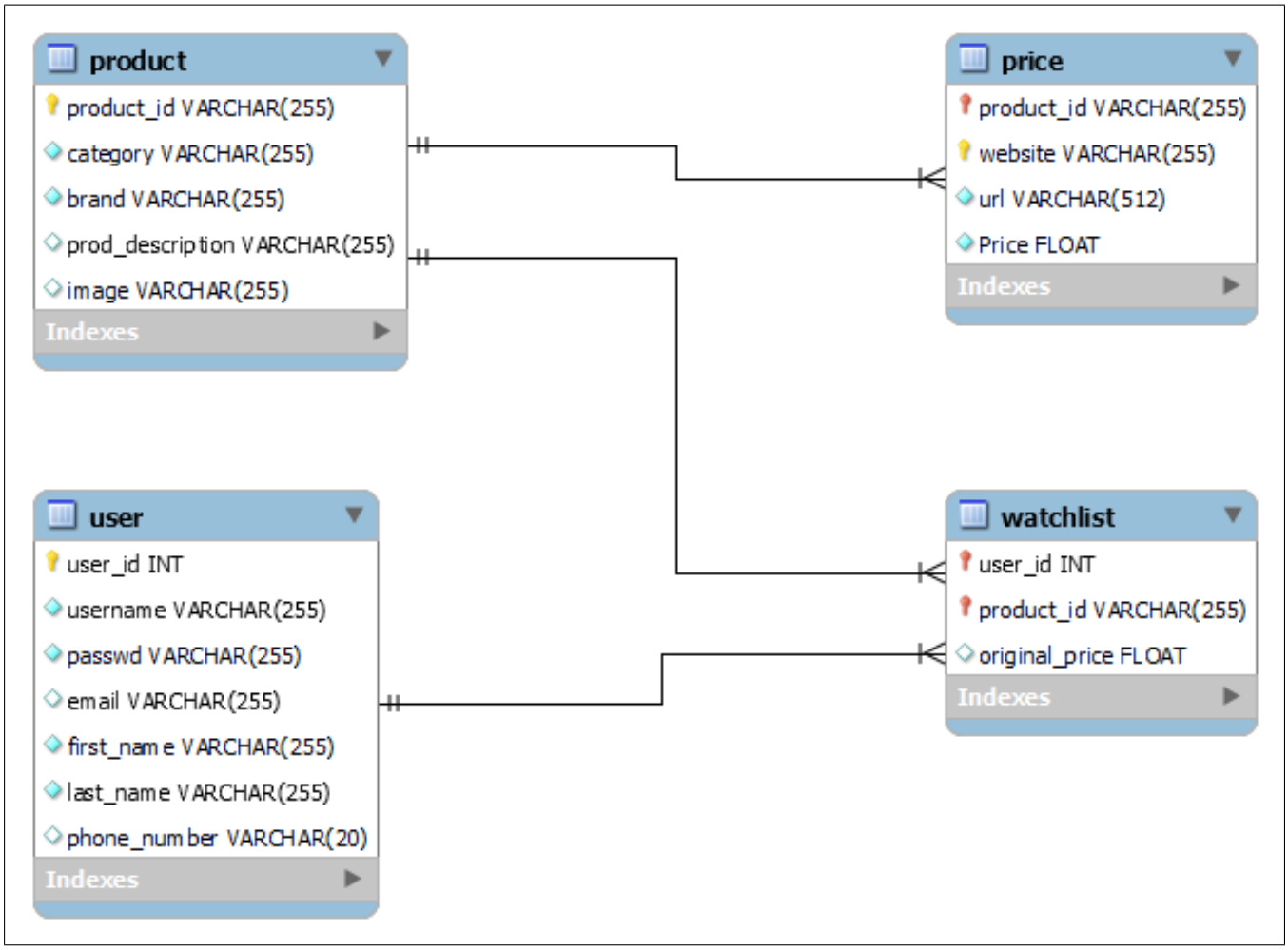


Figure 4.5: Entity Relationship diagram

4.3 Functional Model and Description

4.3.1 Data Flow Diagram

Level 0 Data Flow Diagram

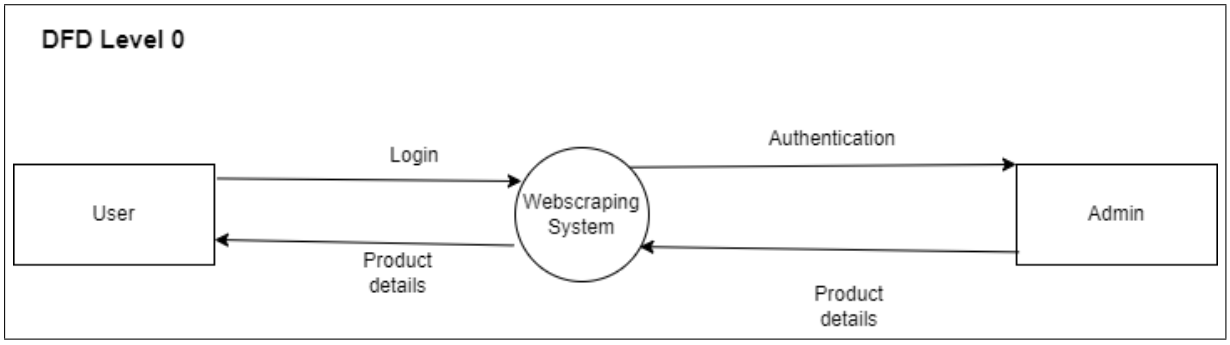


Figure 4.6: DFD Level0

Level 1 Data Flow Diagram

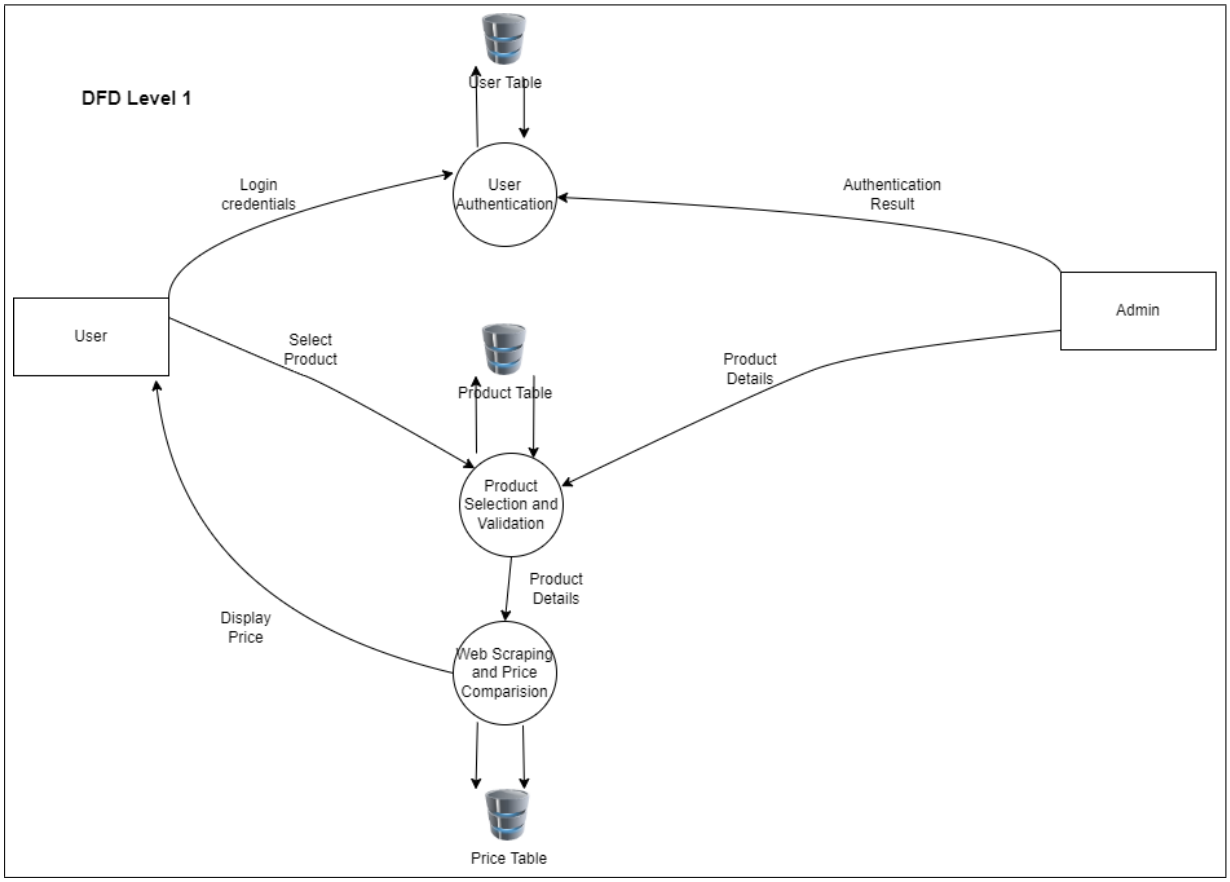


Figure 4.7: DFDLevel1

Level 2 Data Flow Diagram

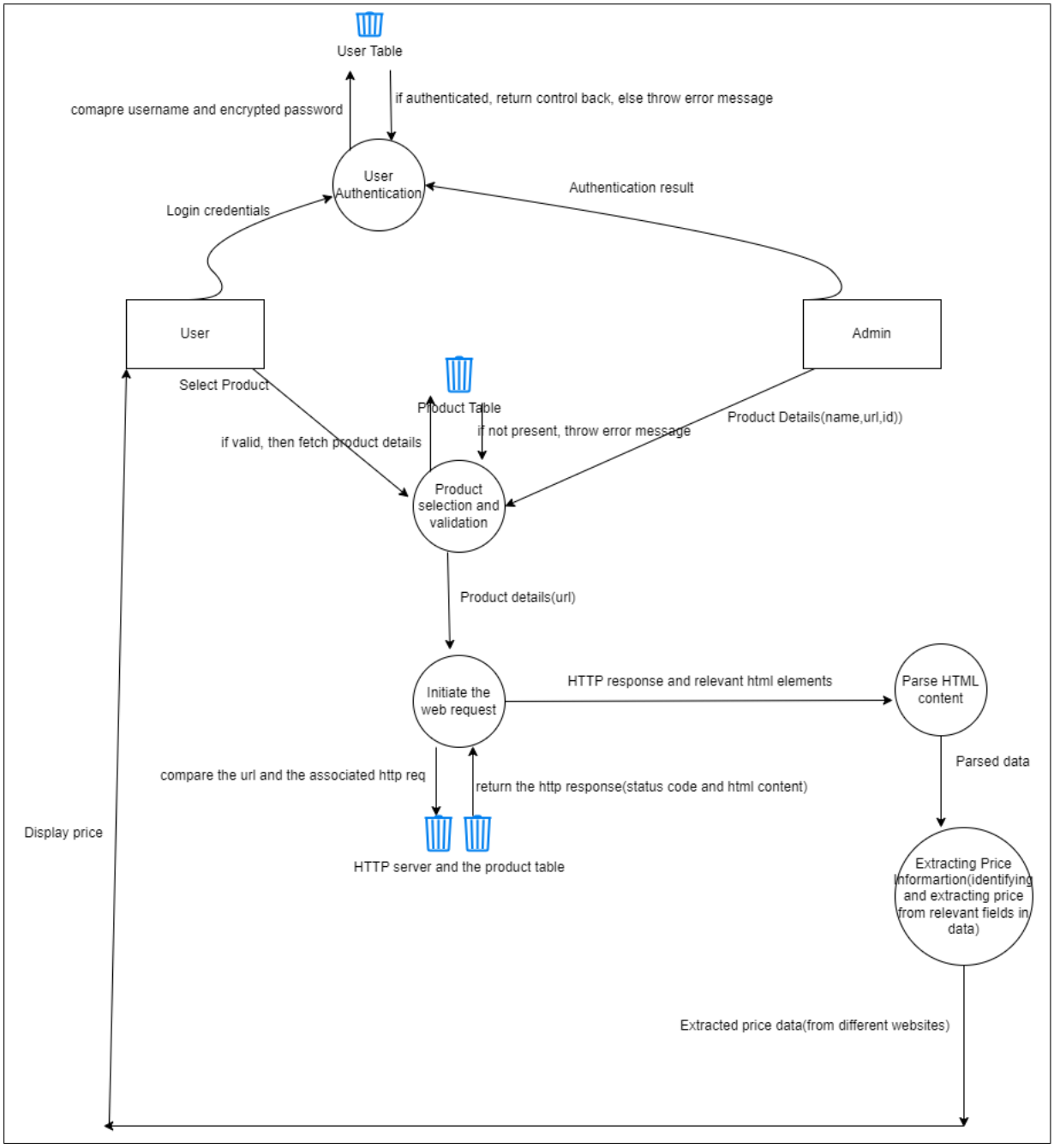


Figure 4.8: DFDLevel2

4.3.2 Description of functions

4.3.3 Activity Diagram:

- The Activity diagram represents the steps taken.

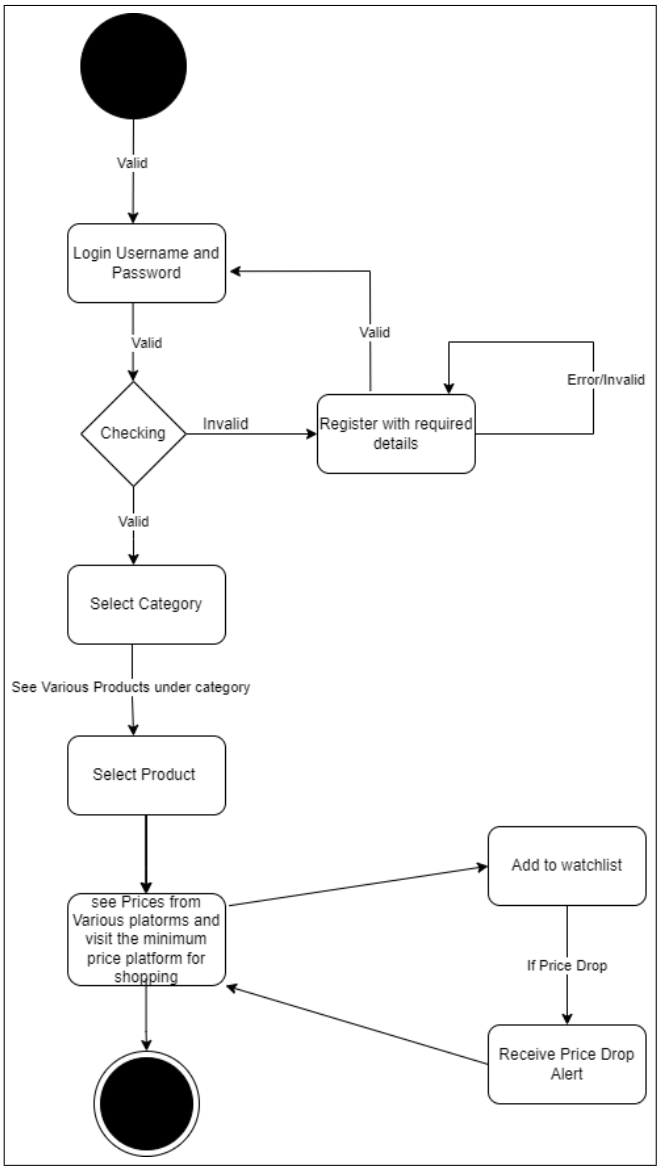


Figure 4.9: Activity diagram

4.3.4 Non Functional Requirements:

Performance Requirements

- **Response Time:** The system should respond to user actions (login, product selection, price comparison) within a reasonable time frame.
- **Scalability:** The system should be able to handle an increasing number of users and products without a significant decrease in performance. This involves efficient use of server resources and possibly implementing load balancing techniques.
- **Concurrency:** The system should support multiple users accessing and interacting with the application simultaneously without experiencing performance degradation. This includes handling concurrent requests for web scraping data from multiple websites.
- **Error Handling:** The system should handle errors gracefully, providing meaningful error messages to users and logging errors for system administrators to troubleshoot performance issues.
- **Database Performance:** Ensure efficient database performance for storing user information, product data, and price comparisons. This includes optimizing database queries and indexing for faster data retrieval.

Safety and Security Requirements:

- **Data Privacy:** Ensure that user login credentials, personal information, and transaction details are securely stored and encrypted to prevent unauthorized access or data breaches.
- **Input Validation:** Implement input validation ensure that user inputs are safe and free from malicious inputs.

- **Content Integrity:** Verify the integrity of scraped content to ensure that prices and product information obtained from external websites are accurate and reliable, minimizing the risk of misleading or fraudulent data.

4.3.5 Design Constraints

- **Data acquisition**
  - **Website structure and its changes:** Adapting to variations in website layouts and data formats.
  - **Volume of data:** Handling large number of websites.
  - **Ethical Scraping:** The techniques must comply with website terms of service and keep in mind data privacy regulations.
- **Client-side constraints**
  - **Offline users:** Users not being logged in to the application.
  - **Synchronization of Data:** We need to ensure the consistency of client-side and server-side data and ensure they are synchronized.
- **Data processing**
  - **Optimization of performance:** We need to process large datasets effectively without compromising speed.

Chapter 5

Detailed Design Document

5.1 Component Design

5.1.1 Class Diagram

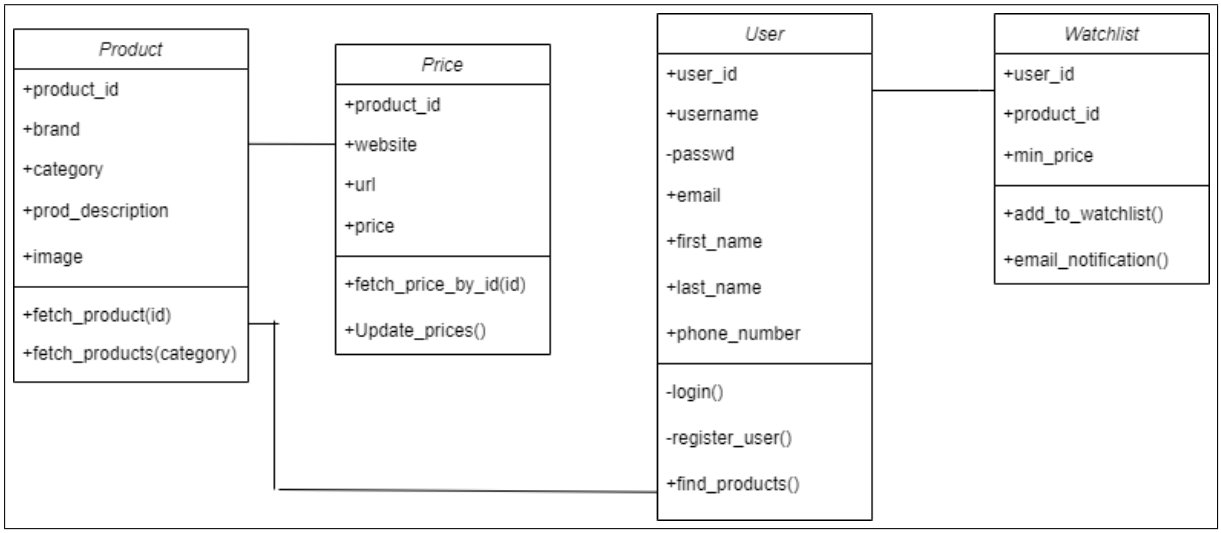


Figure 5.1: Class Diagram

5.1.2 Sequence Diagram

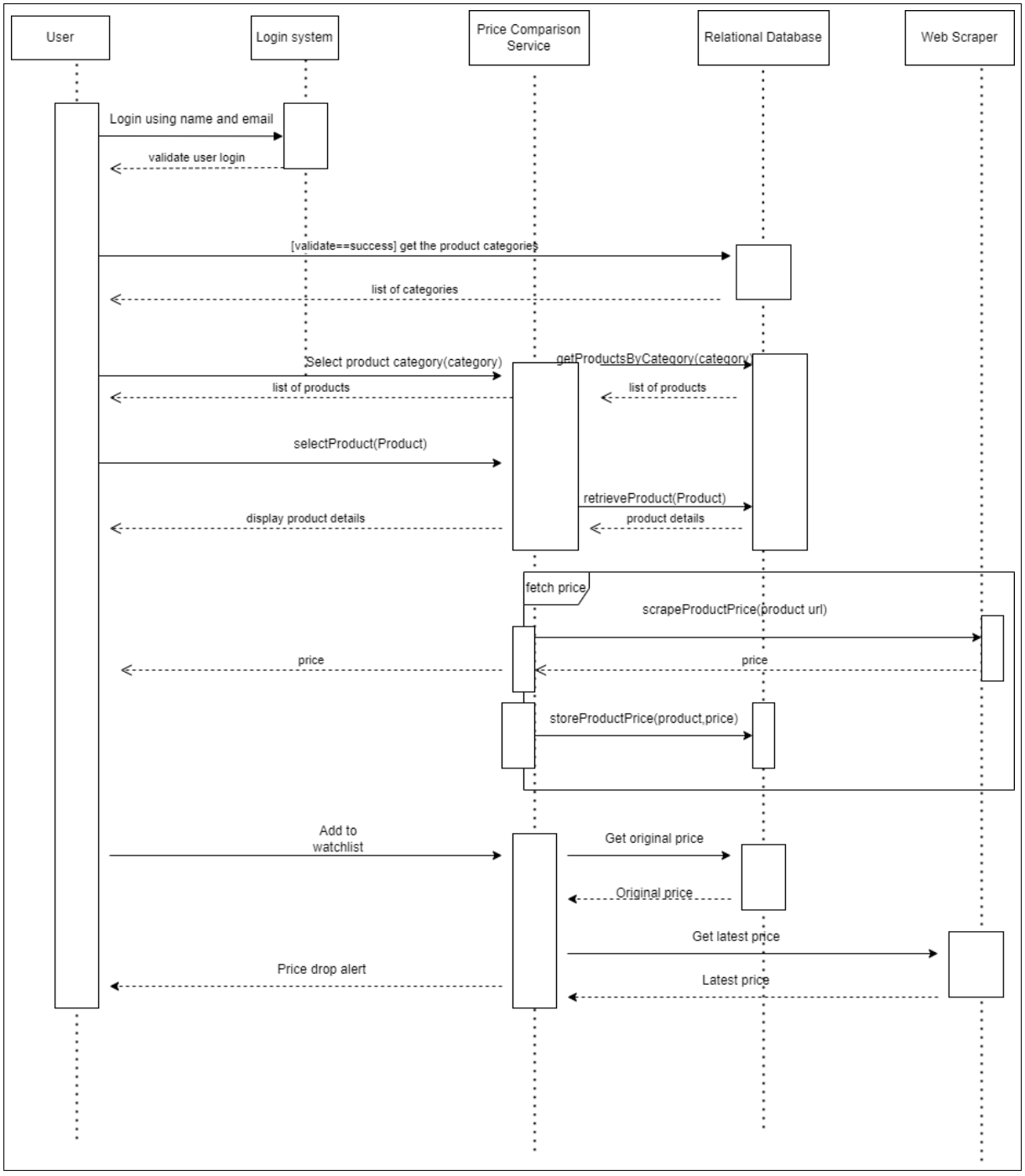


Figure 5.2: Sequence Diagram

5.1.3 Component Diagram

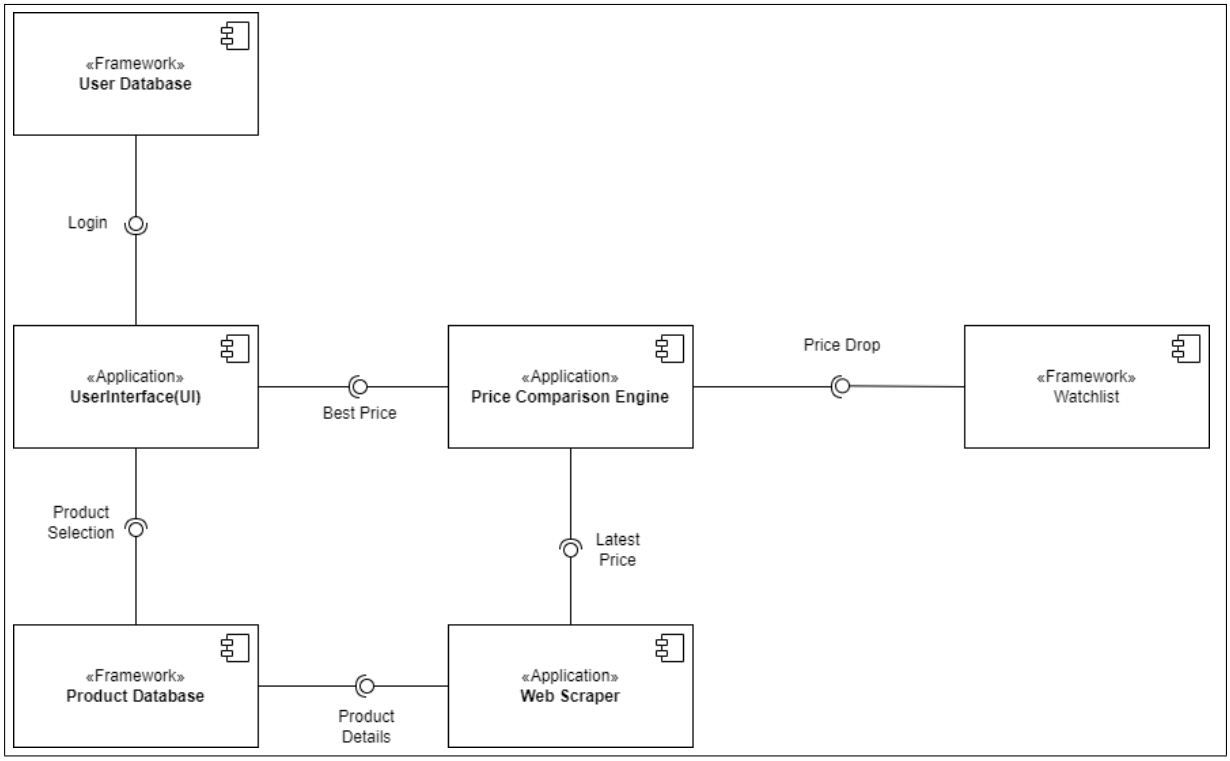


Figure 5.3: Component Diagram

5.1.4 Deployment Diagram

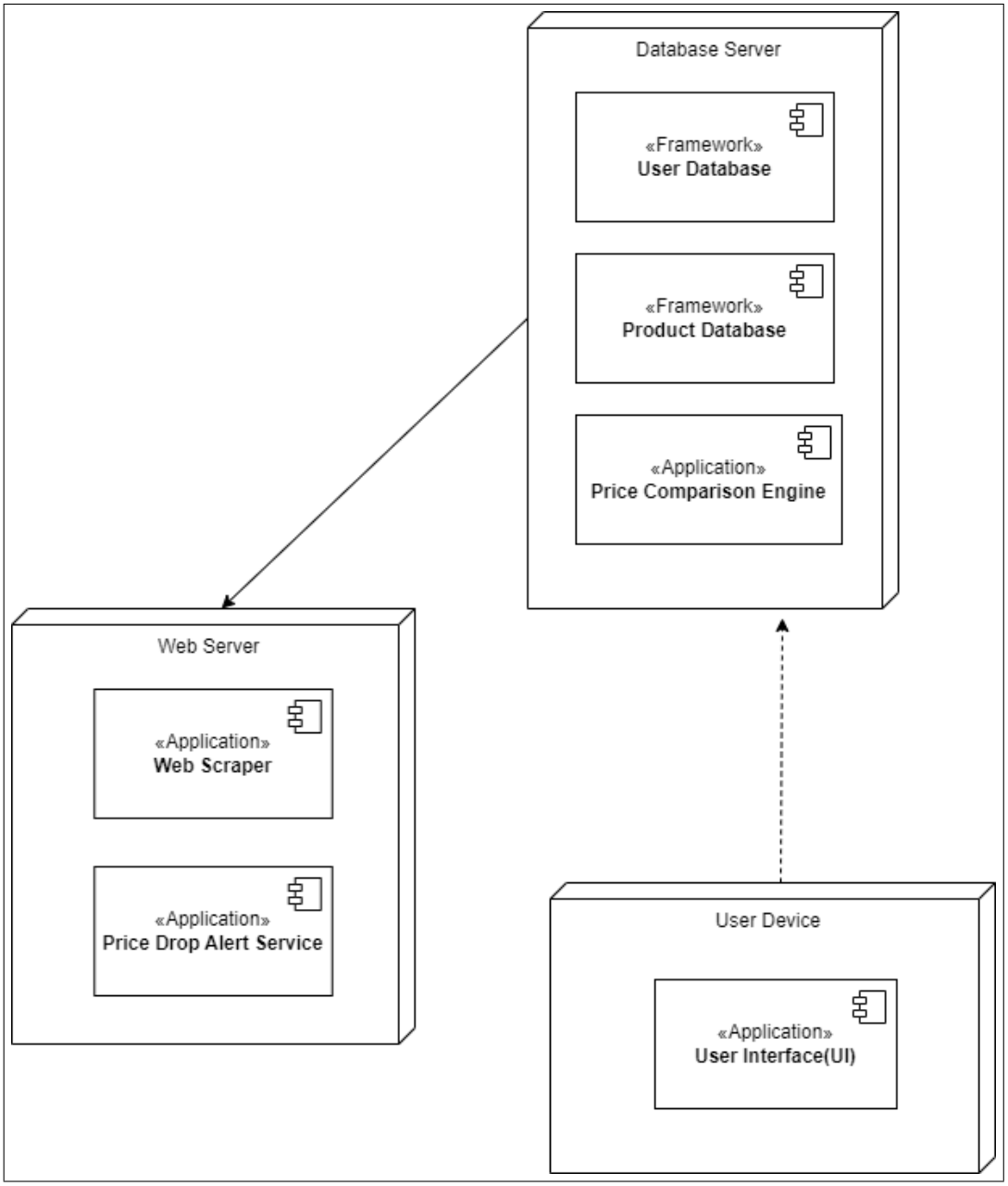


Figure 5.4: Deployment Diagram

5.2 Navigation Flow

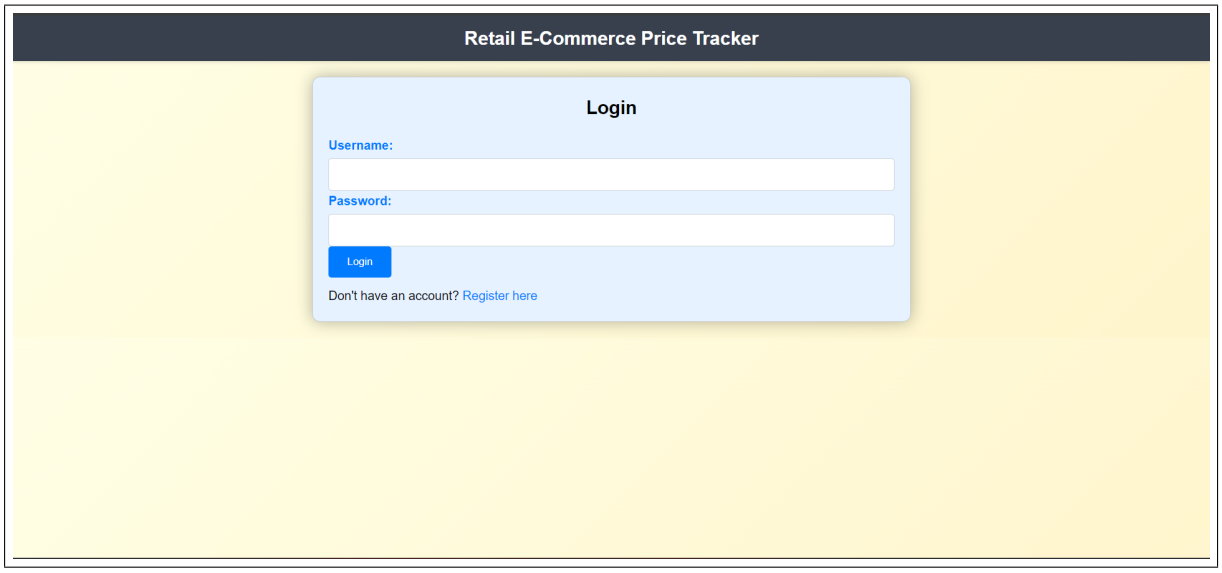


Figure 5.5: Login

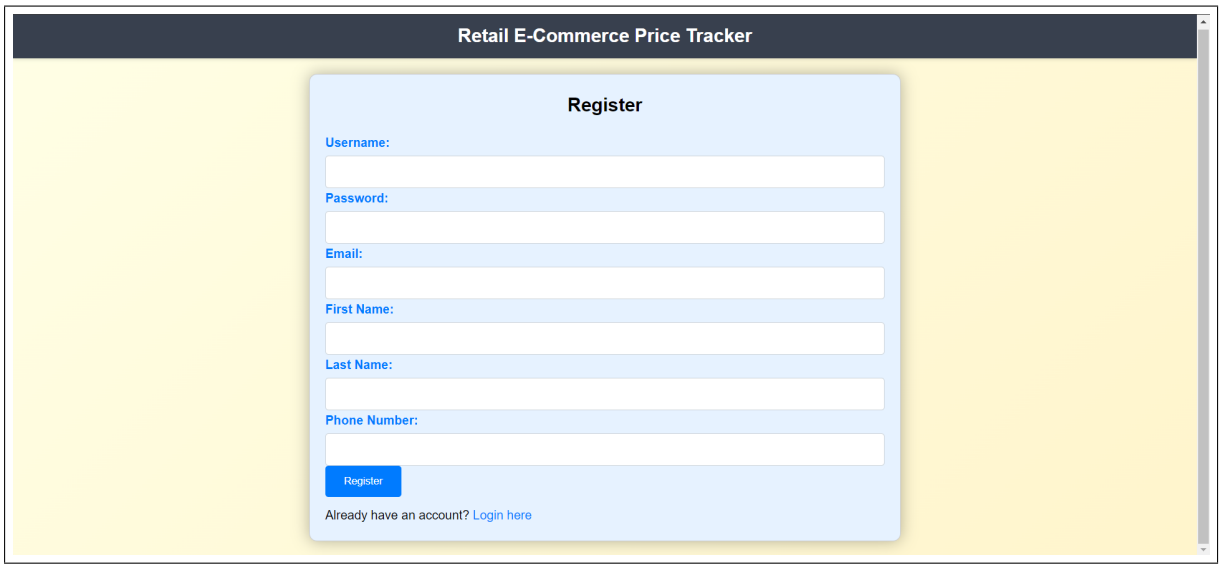


Figure 5.6: Registration

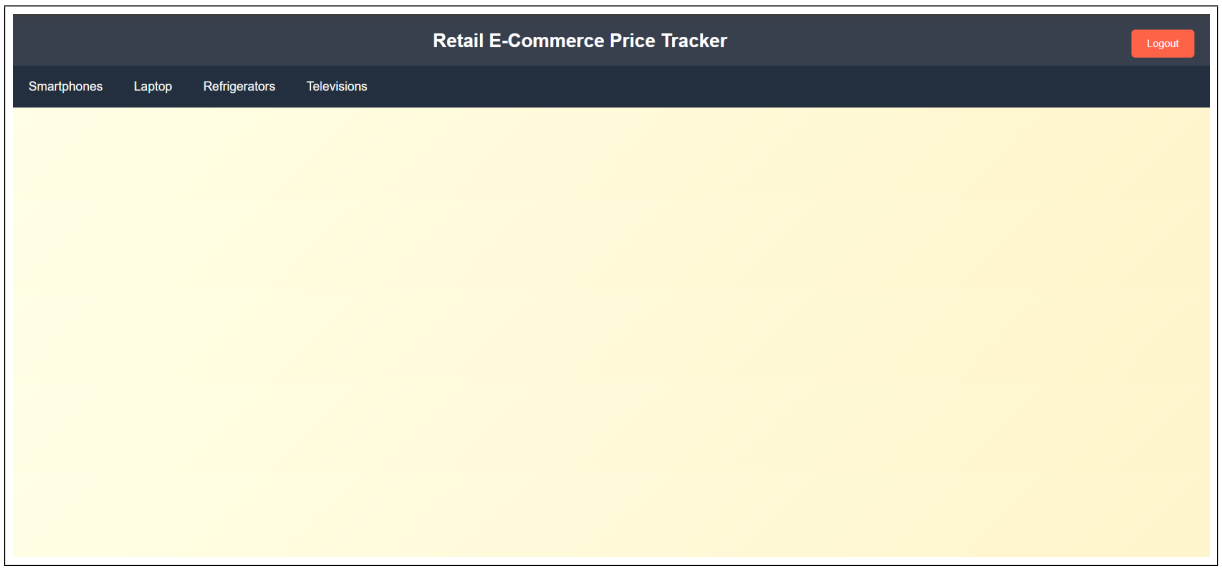


Figure 5.7: main1

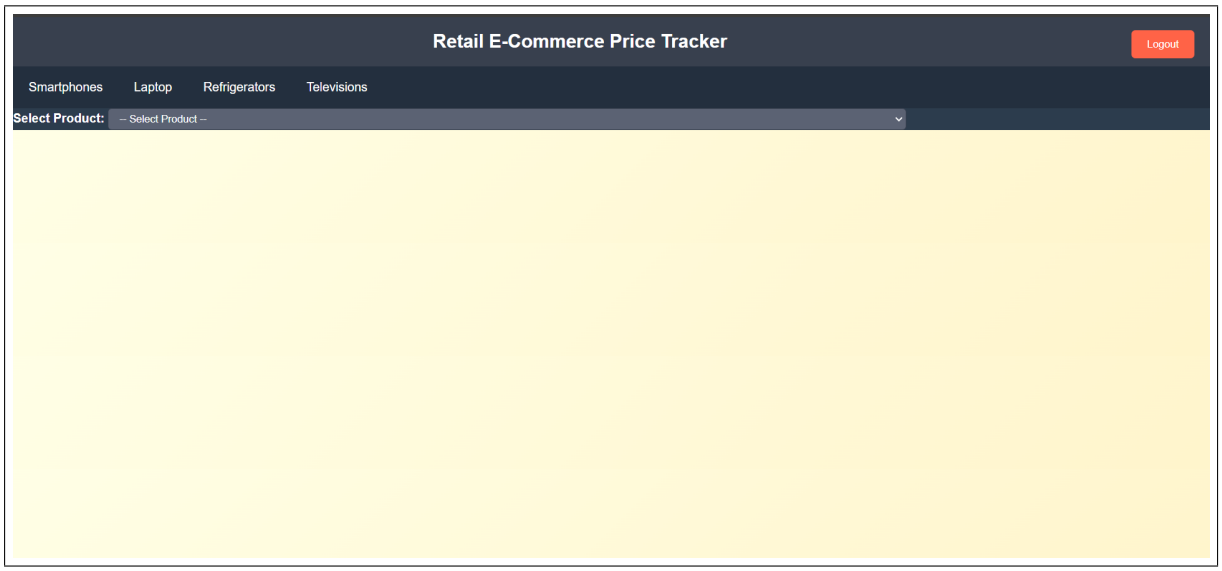


Figure 5.8: main2

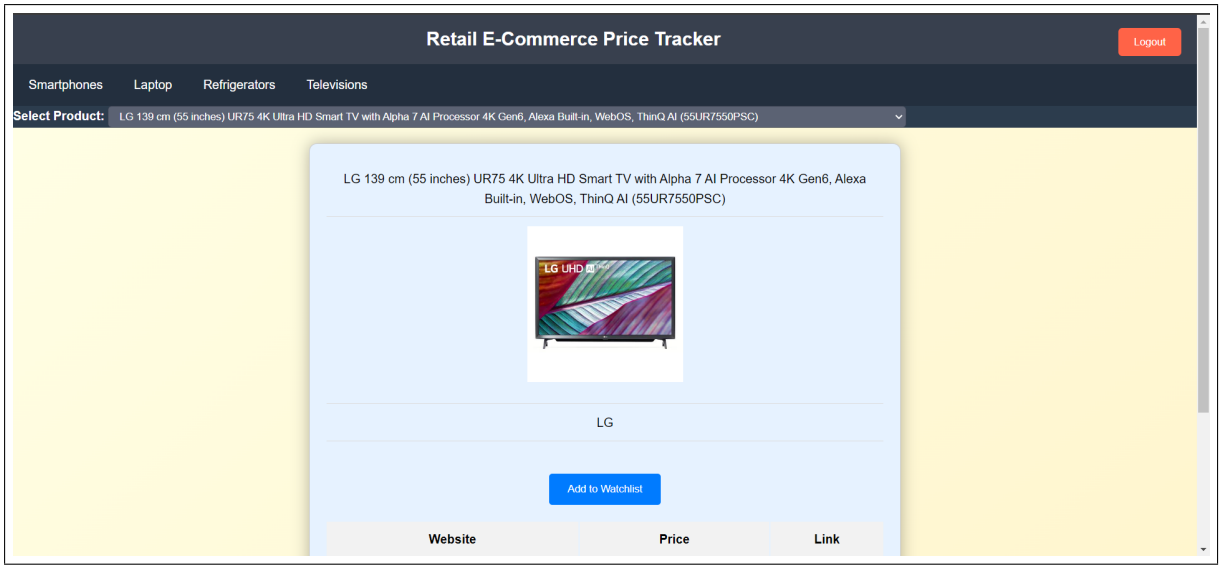


Figure 5.9: main3

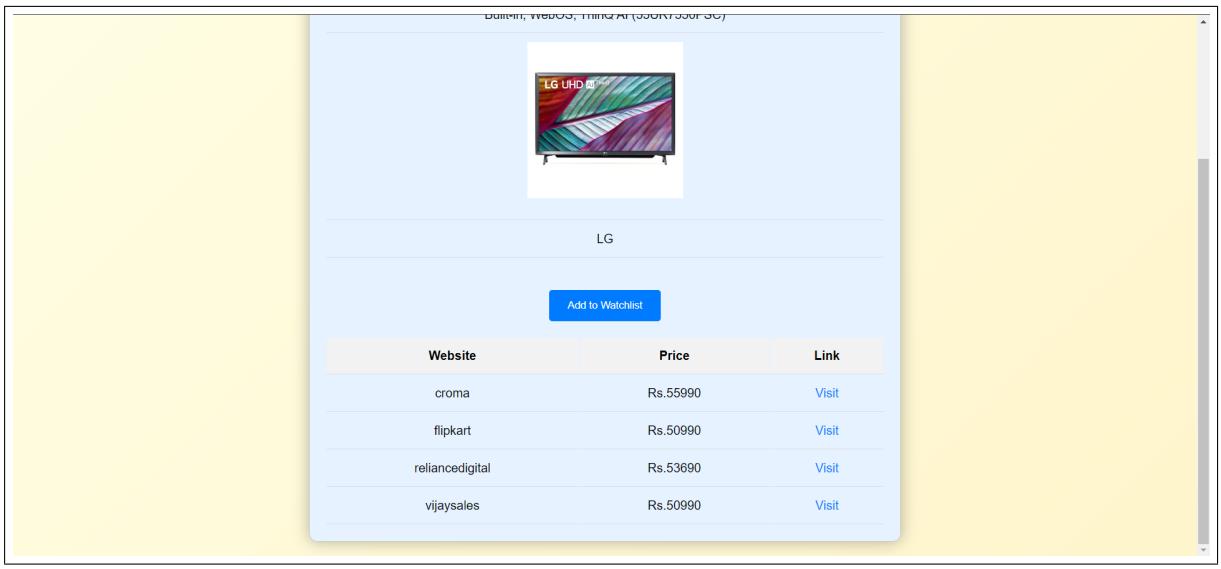


Figure 5.10: main4

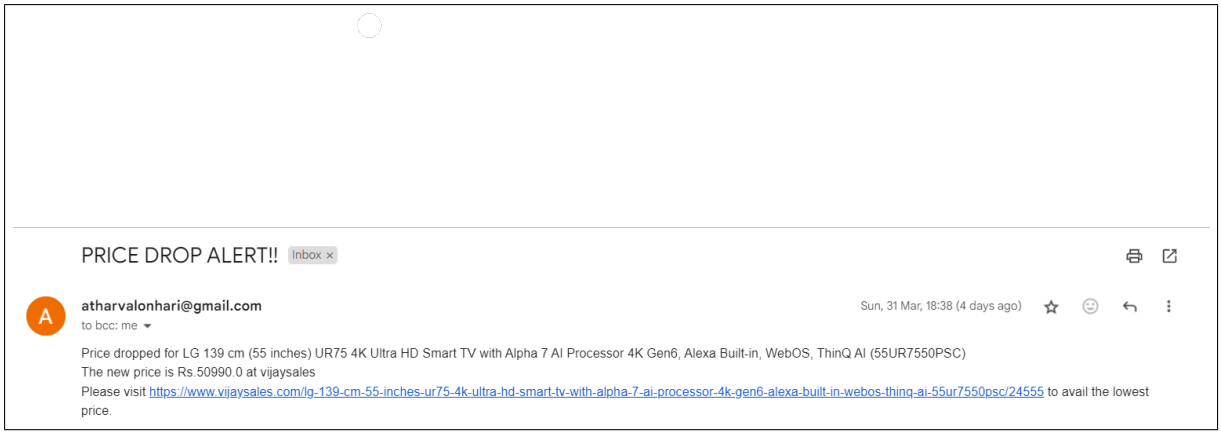


Figure 5.11: pricedrop email

5.3 Code Snippets

```python
@app.route('/index')
def index():
    if 'username' in session:
        return render_template('index.html')
    else:
        return redirect(url_for('login'))

@app.route('/products/<category>')
def get_products(category):
    products = fetch_products(category)
    return jsonify(products)

@app.route('/product/<product_id>')
def get_product(product_id):
    product = fetch_product_by_id(product_id)
    if product:
        return jsonify(product)
    else:
        return jsonify({'error': 'Product not found'}), 404

@app.route('/price/<product_id>')
def get_prices(product_id):
    prices = fetch_prices_by_product_id(product_id)
    return jsonify(prices)
```

Figure 5.12: Code1

```
def populate_data(product_id,website):                          #populate data using webscraping into the database
    print("Product ID: {}".format(product_id))

    url_vijaysales = get_google_search_results(product_id,"vijaysales")
    if(url_vijaysales):
        r = requests.get(url_vijaysales)
        soup = BeautifulSoup(r.content, 'html.parser')
        key = soup.find_all(class_='cls-ty sptyp')
        value = soup.find_all(class_='cls-vl spval')
        brand = None
        category = None
        for i in range(0, len(key)):
            if(key[i].get_text().strip()=="BRAND"):
                brand = value[i].text.strip()
            if(key[i].get_text().strip()=="Generic Name" or key[i].get_text().strip()=="Type" ):
                category = value[i].text.strip()
        print(brand)
        print(category)
        prod_description = soup.find(class_='pdpinfor').text.strip().replace('Specifications Of ','')
        print(prod_description)
        img = get_img_src(url_vijaysales)
        print(img)
        Price_vijaysales = soup.find_all(class_='clsSpecPrc clsWithVSP')
        if(Price_vijaysales):
            Price_vijaysales = int(Price_vijaysales[0].text.strip().replace("Offer Price₹","").replace(",",""))
        else:
            Price_vijaysales = int(soup.find_all(class_='priceMRP')[0].text[4:].split("M")[0].replace(',',""))
        print("Vijaysales: ₹{}".format(Price_vijaysales))
        query = "INSERT IGNORE INTO product(product_id,category,brand,prod_description,image) VALUES ('{}','{}','{}','{}','{}');".format(product_id,
        mycursor.execute(query)
        mydb.commit()
```

Figure 5.13: Code2

```
query = "SELECT * FROM watchlist"
mycursor.execute(query)
list = mycursor.fetchall()
for i in range(len(list)):
    user_id = list[i][0]
    product_id = list[i][1]
    original_price = list[i][2]
    lowest_price,website,url = update_prices(product_id)
    if(lowest_price < original_price):
        query = "UPDATE watchlist SET original_price = {} WHERE product_id = '{}' AND user_id = {}".format(lowest_price,product_id,user_id)
        mycursor.execute(query)
        mydb.commit()
        subject = "PRICE DROP ALERT!!"
        query = "SELECT prod_description FROM product WHERE product_id = '{}'".format(product_id)
        mycursor.execute(query)
        product = mycursor.fetchone()
        body = "Price dropped for {}\nThe new price is Rs.{} at {}\nPlease visit {} to avail the lowest price.".format(product[0],
                                                                                                                        lowest_price,website,url)
        msg = f"Subject:{subject}\n\n{body}"
        query = "SELECT email FROM user WHERE user_id = {}".format(user_id)
        mycursor.execute(query)
        email = mycursor.fetchone()[0]
        server.sendmail(EMAIL_ID, email, msg)
        mycursor.close()
        mydb.close()
        server.quit()
time.sleep(86400)
```

Figure 5.14: Code3

```html
<body>
    <div class="header">
        <h1>Retail E-Commerce Price Tracker</h1>
        <form action="/logout" method="post">
            <button type="submit" class="logout-button">Logout</button>
        </form>
    </div>

    <div class="categories">
        <a href="#" onclick="fetchProducts('SmartPhones')" value="SmartPhones">Smartphones</a>
        <a href="#" onclick="fetchProducts('Laptop')" value="Laptop">Laptop</a>
        <a href="#" onclick="fetchProducts('Refrigerators')" value="Refrigerators">Refrigerators</a>
        <a href="#" onclick="fetchProducts('Televisions')" value="Televisions">Televisions</a>
    </div>

    <div class="dropdown" style="display: none;">
        <div class="form-group">
            <label for="productSelect" style="font-weight: bold; color: white;">Select Product:</label>
            <select class="form-control" id="productSelect">
                <option value="0">-- Select Product --</option>
            </select>
        </div>
    </div>
```

Figure 5.15: Code4

```css
button {
    display: inline-block;
    background-color: var(--primary-color);
    color: #fff;
    border: none;
    border-radius: 0.25rem;
    padding: 0.75rem 1.5rem;
    cursor: pointer;
}

button:hover {
    background-color: #0056b3;
}
```

Figure 5.16: Code5

```
document.addEventListener("DOMContentLoaded", function() {
    // Hide product details initially
    document.querySelector('.product-details').style.display = "none";
    document.querySelector('.price-details').style.display = "none";
    document.querySelector('.card').style.display = "none";


    // Add event listener to navbar items
    var navbarItems = document.querySelectorAll('.categories a');
    navbarItems.forEach(function(item) {
        item.addEventListener("click", function() {
            var category = item.getAttribute("value");
            console.log("Category selected:", category);
            fetchProducts(category);

            // Show the select product part
            document.querySelector('.dropdown').style.display = "block";
        });
    });

    // Add event listener to product select dropdown
    document.getElementById("productSelect").addEventListener("change", function() {
        var productId = this.value;
        if (productId !== "0") {
            document.querySelector('.card').style.display = "block";

            fetchProductDetails(productId);
            // Show product details after selecting a product
            document.querySelector('.product-details').style.display = "block";
            fetchPriceDetails(productId);
```

Figure 5.17: Code6

Chapter 6

Summary and Conclusion

The Retail E-Commerce Price Tracker project aims to develop a comprehensive web application that empowers users to track and compare prices of electronic items across multiple online retailers. By leveraging web scraping techniques, the application collects real-time pricing data and presents it through a user-friendly interface, enabling users to make informed purchasing decisions and maximize savings. Key features include price comparison, price tracking with alerts, deal discovery, product research, budget planning, and market analysis.

Throughout the development process, careful consideration is given to factors such as data accuracy, legal compliance, user privacy, and system performance. Responsibilities of developers include requirements analysis, system design, coding, testing, documentation, and collaboration with stakeholders. The project estimates include time, cost, resource allocation, risk assessment, and contingency planning to ensure successful project execution.

Use cases demonstrate the diverse applications of the Retail E-Commerce Price Tracker application for both consumers and businesses in navigating the electronic marketplace

effectively. Data object descriptions outline the key entities involved in the application, including users, products, prices, and watchlists. These entities facilitate the management of user accounts, product details, pricing information, and price tracking functionality within the application.

In conclusion, the Retail E-Commerce Price Tracker project aims to address the challenges faced by consumers in the electronic marketplace by providing a centralized platform for price comparison, tracking, and analysis. By empowering users with real-time pricing data and personalized tools, the application facilitates informed decision-making and enhances the overall shopping experience. Through continuous improvement and adaptation to user feedback, the Electronic Price Tracker application strives to remain a valuable resource for consumers and businesses alike in the dynamic landscape of online retail.