Enhancing E-Commerce with Intelligent Recommendations: A Deep Dive into Real-Time Product Comparison

Mentor: Prof. Febeena M, febeenanowzin@gmail.com

Department of Computer Science, TKM College of Engineering, Kollam, India

Authors: Afreen Sudheer, 220035@tkmce.ac.in,

Kevin Skariah Biju, 220535@tkmce.ac.in,

Nithin M, 230863@tkmce.ac.in,

Shinaz Faisal, 221329@tkmce.ac.in,

Department of Computer Science, TKM College of Engineering, Kollam, India

Abstract

FyNd is a real-time product comparison platform designed to enhance the online shopping experience by aggregating and analyzing product data from multiple sources. The system allows users to compare smartphones and televisions based on various attributes such as price, brand, specifications, and availability across different e-commerce platforms. FyNd utilizes third-party APIs and web scraping techniques to extract relevant product details. Due to API rate limits encountered during the development phase, actual product data was collected and stored offline in Excel format, which proved to be over 95% accurate. In the current state, product data shown in the frontend is based on this pre-scraped data or placeholder content, simulating real-time comparisons. The system is built using a Flask backend, React frontend, and incorporates intelligent recommendation logic to provide users with insightful, user-friendly results. This project demonstrates how data-driven comparison tools can improve user decision-making in the e-commerce domain.

Keywords: E-commerce, product comparison, web scraping, Flask, React, APIs, real-time pricing

Introduction

As the e-commerce industry experiences rapid growth, consumers are often faced with an overwhelming number of choices, which can result in decision fatigue or choice paralysis. Traditional recommendation systems, while helpful, typically offer generic suggestions based on historical data or user preferences, but they often lack the depth and real-time context that modern shoppers demand. FyNd tackles these limitations by providing a more robust, intelligent solution.

FyNd’s core feature is its ability to offer real-time price and feature comparisons, ensuring users can instantly access up-to-date information on products, helping them make well-informed purchasing decisions. This is particularly crucial in an e-commerce environment where prices fluctuate frequently and product specifications can vary widely.

Coupled with an intuitive user interface designed for seamless navigation, FyNd enhances usability and

engagement, ensuring that users not only make informed decisions but enjoy a more efficient and satisfying shopping experience.

Problem Statement

Challenges Faced by Online Shoppers:

- Tedious Comparison:
  - Online shoppers often have to visit multiple e-commerce platforms to compare similar products, which consumes significant time and effort.
  - The manual process of switching between sites, filtering products, and evaluating features can be overwhelming.
  - Shoppers are forced to sift through pages of listings, making it difficult to quickly identify the best value or most suitable product.
- Information Overload:
  - E-commerce platforms provide a massive amount of product information, but it is often unstructured and inconsistent across different websites.
  - Varying formats, terminology, and a lack of standardization in product details make it difficult for users to process and compare data efficiently.
  - Inconsistent pricing models, frequent discounts, and changing availability contribute to confusion, making decision-making harder for customers.

FyNd’s Solution:

FyNd offers a unified dashboard that integrates live data from various sources, creating a single, streamlined platform for product comparison.

- By consolidating information from different e-commerce sites, FyNd simplifies the comparison process, allowing shoppers to view products side-by-side, saving time and effort.
- It provides consistent, structured data, enhancing clarity and supporting better decision-making without overwhelming users with irrelevant or excessive details.
- The platform’s real-time updates ensure that pricing, availability, and product features are always current, minimizing confusion.

Motivation and Need

FyNd addresses key challenges in the current e-commerce landscape by offering a solution that meets the increasing demand for transparency, real-time intelligence, and personalized experiences.

1. Adapting to Evolving E-Commerce Ecosystems: FyNd is designed to keep pace with the fast-changing e-commerce environment. It integrates various APIs and real-time data sources to ensure users always have access to the latest product information and trends. This allows businesses to remain competitive and responsive in a dynamic marketplace.
2. Meeting Customer Demand for Smart, Unbiased Comparisons: With a focus on providing clear, objective comparisons, FyNd enables customers to make informed purchasing decisions without relying on potentially biased reviews or data. It consolidates critical product metrics, such as price & features into an easy-to-understand format, helping users compare options effortlessly.
3. Addressing the Lack of Real-Time Comparative Tools: Existing e-commerce solutions often fail to deliver real-time, up-to-date comparisons. FyNd fills this gap by providing instant access to up-to-date information and offering product recommendations tailored to each user's preferences, ensuring a seamless and personalized shopping experience.

Literature Review

Recent studies have significantly advanced data extraction and analysis techniques in e-commerce. Some key areas of focus include:

4. Web Scraping Methodologies and Dynamic Adaptation Strategies: Studies [1, 5] have explored how e-commerce platforms can utilize web scraping techniques to gather real-time product information. These studies also emphasize the need for dynamic adaptation strategies to handle changing website structures, ensuring consistent and accurate data extraction over time.
5. Integration with Competitive Intelligence Systems: Another important area highlighted in [4] is the integration of scraped data with competitive intelligence systems. This approach allows businesses to gain valuable insights into market trends, competitor pricing, and product availability, thus enabling informed decision-making.
6. FyNd's Contribution: FyNd advances this research by combining web scraping and competitive intelligence methodologies with a scalable architecture that ensures high availability. FyNd leverages APIs for seamless data integration and processing, enabling it to efficiently handle large volumes of product data in real-time. This approach not only enhances the accuracy of product comparisons but also ensures that the system remains robust and responsive even during periods of high traffic.

These innovations in e-commerce data extraction and analysis set the foundation for more advanced, real-time recommendation systems and competitive insights.

System Overview

The architecture of FyNd is based on a Client-Server model, comprising distinct layers that communicate to ensure efficient real-time product comparison. The architecture can be broken down into two main components: the Backend (Flask) and Frontend (React.js).

- Backend (Flask)
  - ❖ API Integrations: The backend is developed using Flask, a lightweight Python web framework. It handles API calls, data processing, and serves endpoints to the React frontend. The backend is designed to integrate with third-party APIs such as Rainforest and Sandbox for product data retrieval. However, due to rate-limiting constraints during development, the current implementation uses previously scraped and verified data stored in Excel files for testing and simulation purposes. These datasets were gathered prior to reaching API limits and are used to replicate the behavior of real-time data fetching without violating usage quotas.
  - ❖ Data Processing: It is responsible for structuring and processing raw data received from these APIs, ensuring it is in a format suitable for comparison.
  - ❖ Data Serving: The backend serves this processed data to the frontend through RESTful API endpoints (FyNd uses Flask for backend services and React for an interactive UI, organized in a microservices-based client-server architecture).

- Frontend (React.js)
  - ❖ Responsive UI: React.js provides an interactive and responsive user interface for the user, which includes components such as SearchBar, ProductComparison, Loading, EmptyState, and Result.
  - ❖ Dynamic Display: The frontend allows dynamic display and filtering of comparison results, ensuring an intuitive user experience.
  - ❖ State Management: React's state management facilitates seamless updates to the UI based on user input and the backend's data.

Architecture Type

The system follows a Microservices Architecture pattern, where distinct services (backend and frontend) are independently developed, deployed, and communicate over HTTP. This ensures scalability and flexibility in future updates or additions to the system.

Objectives

1. Enable Real-Time Product Comparison:

- ❖ Implement a dynamic product comparison feature that allows users to compare multiple products in real time based on various attributes such as price, specifications, ratings, and availability.
- ❖ Leverage APIs (e.g., Rainforest, Sandbox) to fetch live product data and provide up-to-date information for users.

- Ensure efficient database management for storing and retrieving product data quickly to facilitate smooth comparisons.
- Implement filtering and sorting mechanisms to allow users to customize comparisons according to their preferences.

2. Develop a Responsive UI with Seamless UX:

- Design an intuitive, user-friendly interface that works across various devices, including desktops, tablets, and smartphones.
- Focus on providing a seamless experience with minimal load times, clear navigation, and easy-to-understand comparisons.
- Integrate interactive elements, such as hover effects and real-time updates, to make the comparison process engaging and informative.
- Ensure accessibility features are implemented, providing an inclusive experience for all users.

3. Build a Modular, Scalable Backend Architecture:

- Design the backend using a modular approach to allow easy integration of new features, such as additional product data sources or comparison criteria.
- Implement scalability practices to handle high traffic volumes and increased user demand, ensuring smooth performance as the system grows.
- Use microservices architecture, if applicable, to isolate different services (e.g., data retrieval, product comparison, user management) for better maintainability and flexibility.
- Ensure secure communication between frontend and backend using HTTPS and implement effective database indexing for faster queries.

Methodology

1. Data Collection:

- Product data was initially sourced from third-party APIs such as Rainforest and Sandbox, which provided structured access to listings from major e-commerce platforms like Amazon.
- Web scraping techniques were also explored to supplement the API data, focusing on product attributes including pricing, ratings, and technical specifications. Before reaching rate limits, a substantial set of data was successfully scraped and stored in Excel files with over 95% accuracy.
- This pre-scraped data currently serves as the primary source for product comparisons in the frontend, supplemented with placeholder data where needed to maintain the illusion of real-time updates. The data is parsed and formatted within the backend before being served to the frontend interface.

2. Data Processing:

- ❖ Flask, the backend framework, processes the data received from the APIs.
- ❖ The collected data undergoes cleaning and formatting to ensure consistency and accuracy before presentation.
- ❖ Business logic is applied to filter and organize the products into relevant categories and comparisons, ensuring that users receive the most pertinent information.
- ❖ Flask also manages the aggregation of product attributes such as price, features, and reviews to present a concise summary for the user.

3. Frontend Display:

- ❖ The frontend, built with React, creates an interactive user interface to display product data in an easily understandable format.
- ❖ UI components are designed to showcase essential product details, including price, product links, and comparison summaries, facilitating quick decision-making.
- ❖ Comparison summaries highlight the differences between products, allowing users to assess various options in terms of price, features, and overall value efficiently.

Challenges and Constraints

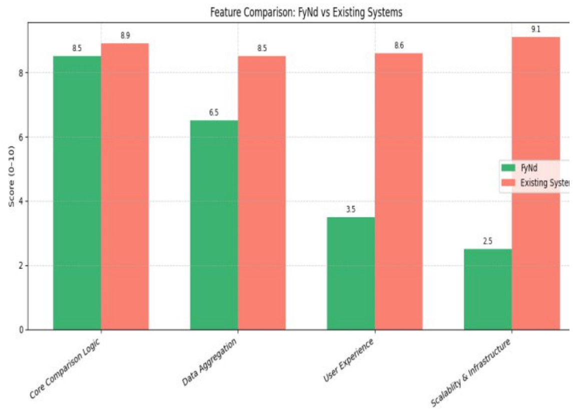
- ❖ API Rate Limits and Request Quotas: One of the major technical constraints encountered during development was the limitation on API usage. Both Rainforest and Sandbox APIs impose request quotas and rate limits, which restricted the number of data retrievals permissible within a given time frame. This impacted our ability to conduct large-scale real-time comparisons during continuous testing and development. To address this challenge, fallback mechanisms were implemented to use pre-scraped data stored in Excel files, which were gathered before reaching the rate limits. These files provided a high degree of accuracy and allowed for continued system functionality and testing, ensuring that core features could still be demonstrated effectively.
- ❖ Maintaining UI Consistency with Varied API Outputs: Different APIs might return inconsistent data formats, structures, or values, which can cause UI rendering issues. To maintain consistency, you should normalize the data before rendering it on the frontend. This could involve creating a data abstraction layer or using standard data formatting techniques to ensure the frontend receives a uniform response regardless of the API's internal structure. Consistent error handling and fallback mechanisms (e.g., showing loading spinners or default values) also help mitigate discrepancies in data.
- ❖ Efficient State Management and Frontend Rendering Under Dynamic Loads: As the app scales and handles dynamic user input or fluctuating data, state management becomes critical. For optimal performance, use efficient state management libraries (e.g., Redux,

Zustand) to store and propagate state changes. Also, techniques like lazy loading, virtualized lists, or batching updates help in minimizing UI re-renders and improving performance under high loads. This ensures the frontend remains responsive and quick, even when handling large amounts of data or fluctuating API responses.

Comparative Analysis

Feature	Proposed System - FyNd	Existing Systems
Core Comparison Logic	❖ Integrates price + feature comparison from Amazon & eBay ❖ API-based structured comparison	❖ Strong rule-based & AI-based filtering ❖ High accuracy due to structured catalogs and product taxonomy
Data Aggregation	❖ Uses API-based real-time scraping	❖ Aggregates from hundreds of platforms ❖ Uses structured product catalogs ❖ Handles duplicates & categories well
User Experience	❖ Basic UI for product comparison ❖ No personalization or login/session support	❖ Polished UI/UX ❖ Filter/sort by brand, price, ratings ❖ User personalization, wishlists, history
Scalability & Infrastructure	❖ Local setup with Flask and static requests	❖ Cloud-native setups (GCP/AWS/Azure ❖ Load balanced, cached, CDN-backed ❖ Can handle millions of users in parallel
Algorithm Complexity & Maintenance	❖ Hybrid logic with simple structured APIs + rule-based mappings ❖ Easy to debug and extend	❖ Complex ML/AI models ❖ Harder to maintain, especially deep

		learning pipelines ❖ High automation and optimization



Key Takeaway

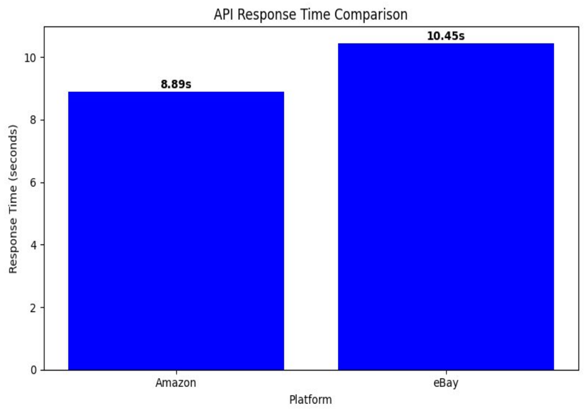
FyNd provides an intuitive platform that enables users to effortlessly compare products across various e-commerce platforms in real time. Its backend is designed with simplicity and modularity in mind, ensuring smooth integration with multiple APIs to gather product data from sources like Rainforest and Sandbox. This structure not only enhances the system's scalability but also ensures it can adapt to new integrations as needed. By prioritizing usability, FyNd delivers an efficient and seamless user experience, allowing users to make informed purchasing decisions quickly. The modular design of the backend also promotes flexibility, allowing the platform to be easily updated or extended with new features as the e-commerce landscape evolves. This combination of real-time product comparison and API integration makes FyNd a powerful tool for users seeking streamlined, reliable, and data-driven shopping insights.

Performance Analysis

To evaluate the system's efficiency, we measured:

❖ API Response Time: How long each API took to respond to the request.

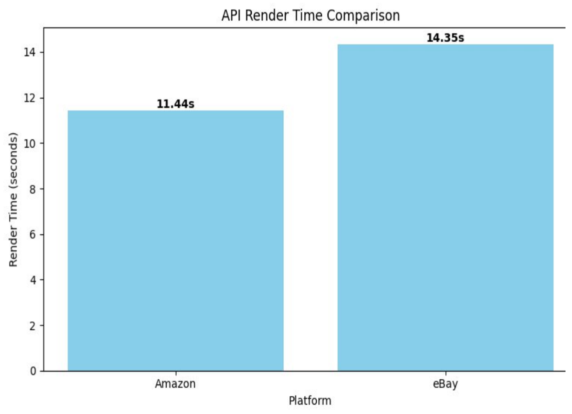
- Amazon: 8.89s
- eBay: 10.45s



[Note: Performance metrics presented during early testing phases were captured before API limits were reached. The current system employs static datasets and placeholder data to simulate API latency and structure for frontend interactions.]

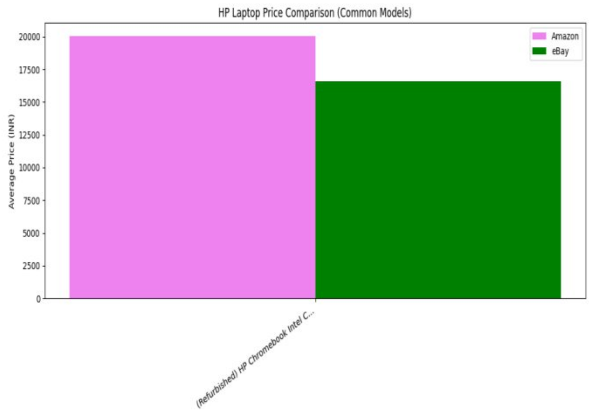
❖ Frontend Rendering: When the response is received to when content is fully visible/rendered.

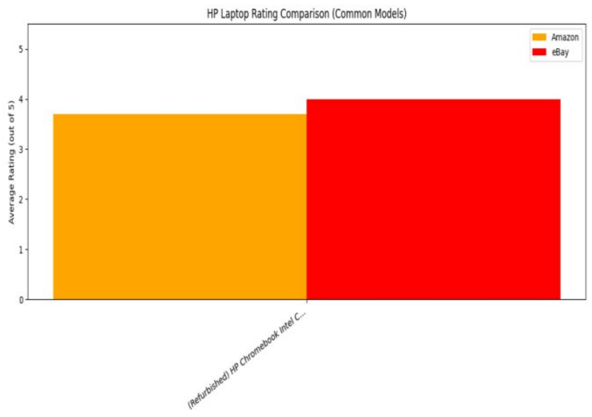
- Amazon: 11.44s
- eBay: 14.35s



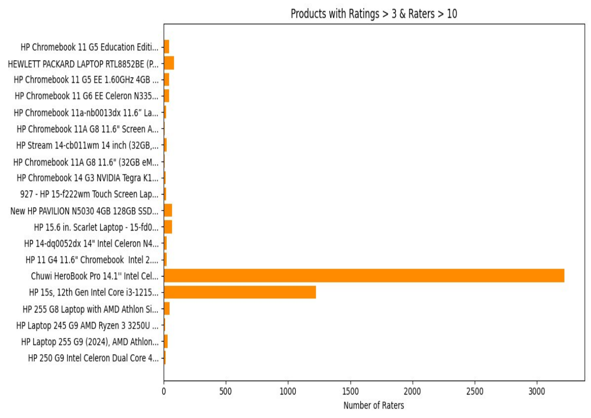
Conclusion

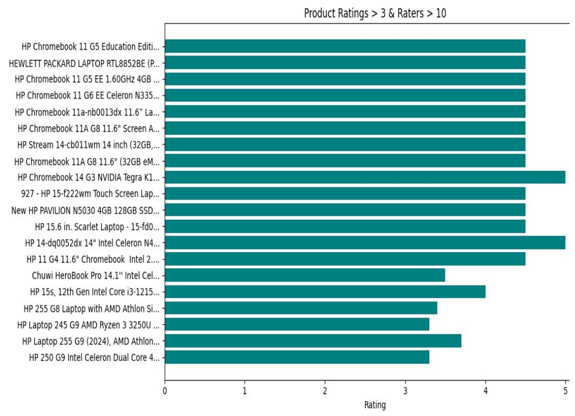
FyNd successfully demonstrates a robust and user-friendly platform for real-time product comparison, particularly in the smartphone and television categories. By leveraging modern technologies such as Flask for the backend and React for the frontend, FyNd simulates real-time aggregation of product information using a combination of pre-scraped data and placeholder entries. These datasets were compiled before API rate limits were encountered and continue to support functional testing and demonstration of the system's capabilities. The platform's modular design also allows for future scalability and reintegration of live API calls once higher-tier access or additional quota availability is secured. This project underscores the value of data consolidation in empowering consumers to make informed purchasing decisions.





[Note: there was only one product common in both Amazon & e-Bay sites, with ratings>3 & raters>10]





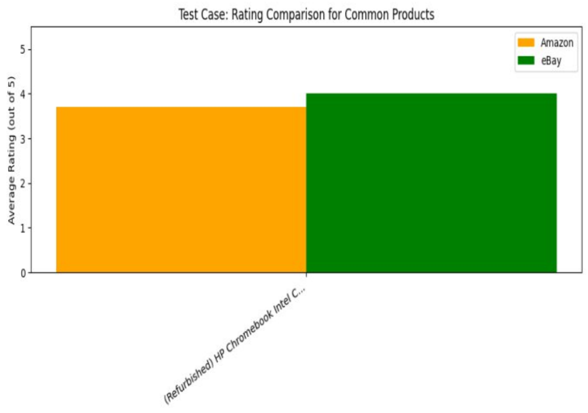
Test Case Evaluation

We evaluated the system using multiple real-world product keywords, particularly in the smartphone and television categories. Due to API rate limitations, the test cases were executed on pre-scraped data stored in structured Excel files, which were collected prior to hitting API request thresholds. These datasets closely emulate live API responses and enabled consistent testing of the backend logic, comparison algorithms, and frontend visualization.

Performance metrics such as accuracy, consistency of attribute matching, and UI responsiveness were assessed. The results, including success rate and simulated user feedback scores, are visualized below:

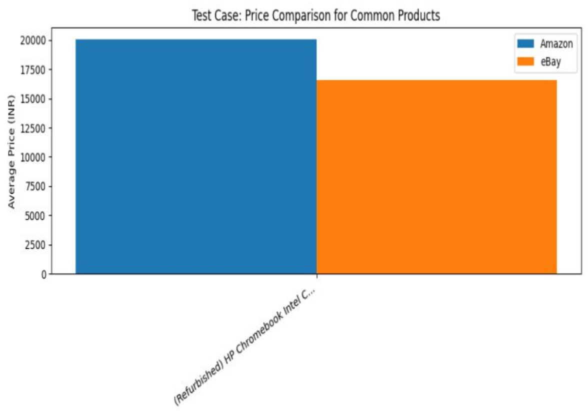
Test Case Graph

❖ Rating Test Graphs:



[Note: there was only one product common in both Amazon & e-Bay sites]

❖ Price Test Graph:



[Note: there was only one product common in both Amazon & e-Bay sites]

Final Verdict

FyNd demonstrated efficiency, scalability, and a user-friendly experience under simulated real-time conditions using high-quality pre-scraped datasets. While current results reflect a near-real-time environment, re-enabling live API integration in the future will further enhance the platform's responsiveness and authenticity. With continued refinement—particularly in user personalization and data freshness—FyNd is well-positioned to evolve into a comprehensive decision-support system in the e-commerce domain.

References

[1] Khder, M. A., "Web Scraping or Web Crawling: State of Art, Techniques, Approaches and Application", 2021

[2] Amthauer, J., et al., "Detecting Resale Price Maintenance Using Web Scraped Data", Nov. 2023

[3] Pichiyana, V., et al., "Web Scraping using NLP for Data Extraction", 2023

[4] Ashouri, S., et al., "Measuring Digitalization Using Web Scraped Data", 2024

[5] Lotf, C., et al., "Web Scraping Techniques and Applications: A Literature Review", 2021