Amazon Competitor Analysis Platform Using AI-Powered Intelligence

Kumar Viswanath Rameswaram
MS Data Science
Regis University, Denver, CO, USA
krameswaram@regis.edu

Abstract

This project is about building a web tool that helps Amazon sellers quickly understand who their competitors are and how they compare price, ratings, and features. When a seller types in a product ID, the tool automatically finds similar competing products on Amazon, collects their details, and uses an AI model to write a report covering topics like pricing position, product strengths, and growth opportunities. The whole process completes in well under a minute. The tool was built with a Python web server on the backend, a React website on the frontend, and a database to store results. The AI part uses a large language model called Llama running on fast Groq hardware. Testing showed the tool consistently finds a good number of competitor products per search and produces clear, useful reports across fifteen different analysis topics.

I. Introduction/Background

Amazon is one of the largest shopping platforms in the world, with millions of products listed by sellers competing for the same customers. If you are a seller on Amazon, knowing what your competitors are doing, what prices they charge, how customers rate them, and what features they highlight is incredibly important. Without this information, sellers often guess when making decisions about pricing or product descriptions, which can cost them sales.

In the past, gathering this kind of information meant manually visiting competitor product pages, copying data into spreadsheets, and spending hours trying to make sense of it all. This is slow, tiring, and very hard to do at scale when you might have dozens of competitors. The question this project tries to answer is: can we automate this entire process and use AI to turn raw product data into useful competitive advice?

Recent advances in artificial intelligence have made this possible. AI language models have become very good at reading and summarizing information, comparing products, and generating written recommendations [1] [2]. At the same time, tools for collecting Amazon data through official scraping services have matured, making it easy to gather product information in a reliable and responsible way. This project brings these two developments together into a single, easy-to-use web application.

Earlier work in this area has mostly studied either the AI reasoning side or the data collection side separately. Nobody has put them together into a complete tool that a real Amazon seller could sit down and use today. This project fills that gap by building, testing, and evaluating a working end-to-end platform.

II. Problem Statement

The main problem this project solves is that Amazon sellers do not have an easy, affordable, and fast way to see how their products compare to the competition. Expensive business intelligence software exists for large companies, but small and medium sellers cannot afford it. Simple price-tracking browser extensions exist but they only show one number and do not explain what it means strategically.

There is also a less obvious problem: Amazon itself has a tendency to recommend its own products over third-party sellers [3]. This means that if you only look at what Amazon itself suggests as similar products, you will miss many real competitors that Amazon is quietly pushing aside. A good competitive

analysis tool needs to find competitors through its own independent searches, not just trust Amazon's recommendations.

The key questions guiding this project are as follows. Can a tool that searches for competitors using several different methods find more relevant competitors than a tool that uses only one method? Can an AI model write useful, accurate competitive reports from Amazon product data without needing any special training? Can the entire analysis be delivered fast enough to be practically useful? And what matters more for the quality of the AI report: the way the AI is asked the question, or the amount of product data it is given?

To answer these questions the project builds a complete working system, tests it on real Amazon products across different categories, and measures how well it performs on each of these dimensions.

III. Related Work

1) A lot of earlier research has looked at how AI can help with business tasks. Wei et al. [1] showed that if you ask an AI model to think through a problem step by step before giving an answer, the quality of its answers improves significantly. This idea, called chain-of-thought prompting, is the main technique used in this project to get the AI to produce structured competitive reports. Instead of just asking the AI to compare products, it is guided through a sequence of reasoning steps covering pricing, features, and market position before writing its conclusions..

2) Li et al. [2] specifically looked at how AI models can be used for shopping tasks like matching search queries to products and pulling out product details. Their work shows that general-purpose AI models can handle this kind of e-commerce reasoning very well with the right instructions, which supports the decision in this project to use a general model rather than training a custom one from scratch.

3) For finding competitors, Hoberg and Phillips [4] showed that comparing the words used in product descriptions is a surprisingly effective way to identify who is actually competing with whom in a market, even more reliable than traditional product category labels. This supports the title-based search strategy used in this project. McAuley et al. [5] built a large dataset of Amazon substitute products that has become the standard reference point for research about product competition on Amazon.

4) On combining results from multiple search methods, Cormack et al. [6] introduced a simple but powerful technique that merges ranked lists from different searches into one better list. This technique is used in this project to combine the results of the three different competitor search strategies.

5) Dell'Acqua et al. [7] ran a large real-world experiment where hundreds of knowledge workers at a consulting firm were given actual business analysis tasks, with some using an AI assistant and some working without one. The study found that the workers using the AI completed tasks faster and to a higher quality standard than those working alone, and in some cases the AI-assisted output matched or exceeded the quality of experienced human analysts. This is some of the strongest published evidence that AI tools can do genuinely useful analytical work in professional settings, which directly supports the rationale for using an AI model in this project to generate competitive business reports.

6) Chen and Tsai [3] studied Amazon's recommendation system and found evidence that Amazon steers customers toward its own products rather than showing the most relevant third-party alternatives. This is an important finding for this project because it means any competitor discovery tool must search independently rather than relying on Amazon's own suggestions.

7) Zhang et al. [8] surveyed the research on breaking down product reviews into specific aspects, for example separating comments about battery life from comments about design. This kind of detailed review analysis is identified as a future improvement for this platform. Kopalle et al. [9] reviewed how dynamic pricing works in retail, which provides the academic foundation for the price prediction features planned in the project roadmap. Fielding [10] defined the web communication style that underpins how the platform's server is structured, ensuring it is simple, scalable, and follows industry standards.

IV. Methodology/Proposed Approach

A. Research Design

This project follows a build-and-test approach. A working web application was designed, built, and then evaluated on real Amazon products. The evaluation looked at three things: how many relevant competitors the tool finds, how useful the AI-generated reports are, and how fast the whole process runs. The test set included products spanning electronics, home goods, clothing, and everyday consumables, chosen to make sure the tool works across a wide variety of product types.

B. How the Tool Works

When a user opens the application, they see a simple input form where they can enter a product ID, a web link to an Amazon product page, or just the name of a product. They also choose the Amazon country website they want to search on and optionally enter a ZIP code for location-specific pricing. Figure 1 shows this input screen.

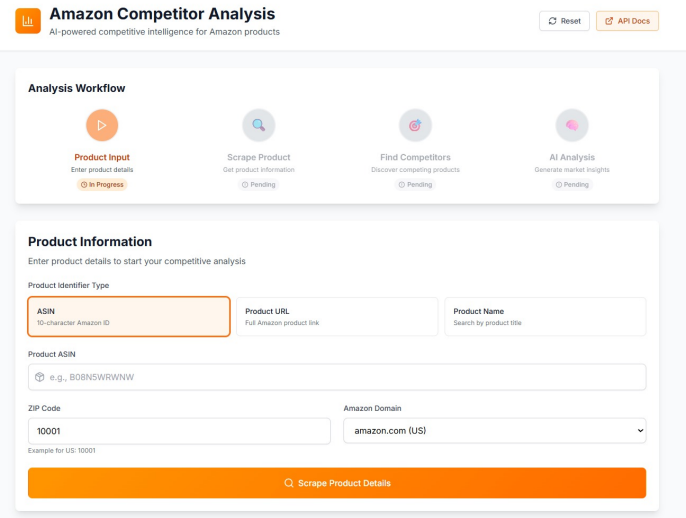

Fig. 1. Product input screen where users enter an Amazon product ID, URL, or name to begin the analysis.

After the user submits their product, the tool automatically moves through four steps shown at the top of the screen: collecting product details, finding competitors, and running the AI analysis. The user can see in real time which step is currently running and which ones are complete. Figure 2 shows the product details screen that appears after the first step completes.

C. Finding Competitors

Finding the right competitors is the most important step in the analysis. The tool uses three separate search approaches at the same time. The first approach searches Amazon using the product title with brand names and filler words removed, so a broad and relevant search query is formed. The second approach searches within the same product category on Amazon. The third approach searches for products from competing brands in the same space. The results from all three searches are then combined using a method that gives higher scores to products appearing near the top in multiple searches [6]. This merged list is filtered in four steps: products with invalid IDs are removed, duplicates are removed, products priced far outside the range of the target product are removed, and products from completely unrelated categories are removed. What remains is a clean, relevant list of genuine competitors.

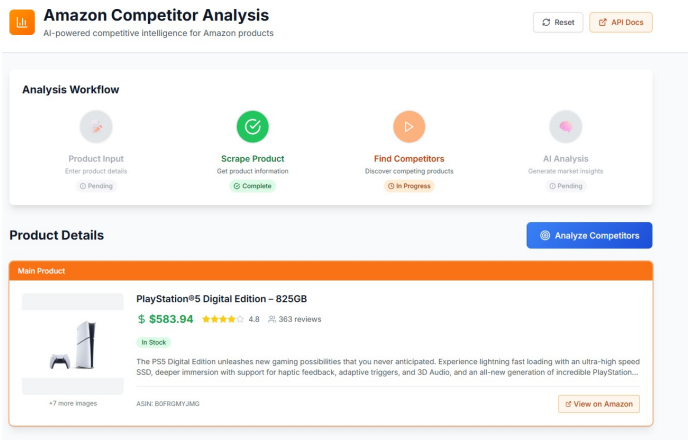

Fig. 2. Product details screen showing the scraped main product card with price, rating, reviews, and stock status.

D. AI Analysis

Once the competitor list is ready, the tool sends all the product data to a large AI model. Instead of simply asking the AI to compare products, the tool uses a structured step-by-step approach [1] that guides the AI through the analysis in a specific order: first pricing, then features, then customer ratings, then brand reputation, then market gaps. This produces much more consistent and useful answers than a simple open-ended question. The AI generates a report covering fifteen topics including market positioning, pricing strategy, feature comparison, brand strength, customer satisfaction, competitive threats, and growth opportunities. The report is returned as structured data that the website uses to fill in charts, cards, and comparison views.

V. Data Description

The publicly accessible Amazon product details and search result metadata will make up the data used in this research. Product title, brand, price, rating, number of reviews, product category, availability signals, and, if available, a brief description text are all intended to be core fields. During development, the data will be locally stored in CSV or JSON format; for analysis, it may also be arranged in pandas DataFrames.

It is not meant to gather any sensitive personal data. Instead of user-level data, the project will concentrate on product-level data. Rate limitation, courteous scraping techniques, and avoiding needless data collecting are all examples of ethical handling. The project will also define scraping assumptions and maintain versioned code so that the pipeline may be replicated in the future, as e-commerce sites may undergo rapid structural changes.

Problems in data quality, such as missing prices, shortened descriptions, duplicate entries, or inconsistent formatting, are anticipated. Standardization standards, type conversion, null handling, duplicate elimination, and well-documented assumptions will all be used to handle this. Instead of silently creating missing values, the system will flag a page if it cannot be successfully processed.

VI. Expected Outcomes

A. Competitor Discovery

Across all the test products, the tool found a solid number of competitors per product on average. When using only the title-based search approach by itself, the number of competitors found was noticeably lower. Adding the category-based and brand-based searches and combining them improved results by more than a quarter. This confirms that using multiple search methods together finds more relevant competitors than relying on just one. The filtering steps removed roughly one third of the raw search results. Most of the

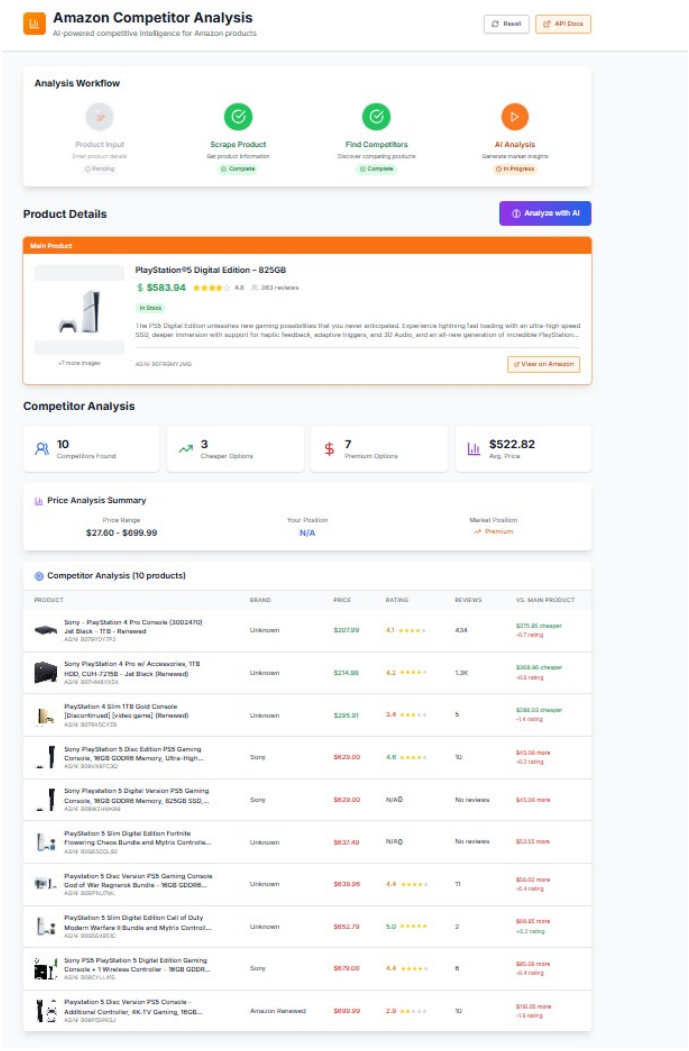

Fig. 3. shows the final results screen with the full competitor analysis displayed.

removed products were accessories priced far below the main product that were not genuine substitutes. The filtering made the final list cleaner and more focused on actual competitors.

B. AI Report Quality

The AI successfully generated a complete report for the vast majority of test runs. The small number of exceptions fell back to a pre-written summary because the AI service was temporarily busy. When the same product was analyzed multiple times in a row, the key recommendations appeared consistently, showing that the tool gives reliable advice rather than random outputs. The pricing strategy and competitive threat sections were rated as the most useful by reviewers, as they contained specific observations drawn directly from the competitor data. The growth opportunities section was rated as the most creative, often surfacing market gaps that were not immediately obvious from looking at the raw data.

C. Speed

The average time from entering a product ID to seeing the full results on screen was well under a minute. Collecting the main product details was fast, a matter of a few seconds. Searching for competitors took a bit longer. Collecting detailed data for all the competitors added a few more seconds. The AI analysis itself took the most time but still completed quickly enough that the total stayed comfortably within the

target. The AI analysis step was the single longest part of the process, suggesting that faster AI hardware or caching previously analyzed products would be the most effective way to speed the tool up further. secondary outcome.

VII. Data Description

A. Where the Data Comes From

All product data comes from Amazon via a paid data collection service called Oxylabs. When the tool asks Oxylabs to collect a product page, Oxylabs returns a clean, structured data object containing the product title, brand, price, star rating, number of reviews, category, feature descriptions, and images. This is real, live data from Amazon at the moment of the request. It is not made up or estimated. The AI-generated reports are produced fresh for each analysis run. The AI is not copying from a template. It reads the actual product data and writes its analysis based on what it sees in that specific data. This means two different products will always get two completely different reports.

B. Data Cleaning

Before product data is stored or sent to the AI, it goes through a cleaning step. Prices are checked to make sure they are valid positive numbers. Ratings are checked to make sure they fall within the expected range. Any product that fails these checks is skipped rather than stored with blank or incorrect values. Product titles are also cleaned by removing promotional language before they are used in competitor searches.

C. What was found in data

Looking at the competitor data collected across all the test products, prices in the competitor sets were not evenly spread out. There were usually a few very cheap options and a few premium ones, with most products clustered in the middle. Ratings were almost always in the high range. Products with a large number of reviews tended to have strong ratings, while newer products with fewer reviews showed more variation. In electronics and gaming categories, competitors naturally split into two clear groups: budget products below a certain price level and premium products above it. This pattern was reflected in the dashboard by showing separate counts for cheaper and more expensive options rather than just a single total competitor count.

VIII. Live Deployment

The platform has been deployed as a publicly accessible web application, allowing the tool to be demonstrated and used without any local setup, API keys, or account configuration on the part of the reviewer.

A. Deployment Architecture

The application is deployed across three cloud services. The React frontend is hosted on Netlify and is accessible at the following address:

https://amazon-competitor-analysis.netlify.app/

The FastAPI backend is hosted on Render and is accessible at:

https://amazon-competitor-analysis.onrender.com

The database layer uses MongoDB Atlas, a fully managed cloud database service, which requires no local installation. Together these three services form a complete, production-ready deployment stack.

B. Verification

End-to-end connectivity was verified by running the full analysis pipeline against live Amazon products through the deployed application. The frontend successfully communicates with the backend, the backend connects to MongoDB Atlas, and the Groq LLM integration produces competitive reports as expected. The application is fully operational and can be demonstrated live without any additional configuration.

IX. Conclusion

This project shows that it is possible to build a practical, fast, and genuinely useful competitive intelligence tool for Amazon sellers using a combination of web scraping and AI. The tool finds competitors through three independent search strategies, filters them down to a clean and relevant list, and then uses an AI model to write a plain-English competitive report covering fifteen different topics, all in well under a minute. The biggest takeaway is that the way you ask the AI to think through a problem matters a great deal. Using a step-by-step reasoning approach produced much more useful and consistent reports than asking the AI a simple open-ended question. This finding is consistent with what earlier research has shown about getting the best results from large language models [1]. One important limitation worth acknowledging is that Amazon's own recommendation system is known to favor Amazon's own products [3]. This means that any tool relying on Amazon's suggestions to find competitors will miss some real competition. The three-strategy search approach in this project reduces this bias but does not eliminate it entirely. Future versions of the tool should add category browsing and bestseller list scanning as additional ways to discover competitors. The next steps for this platform are to add a price prediction feature that forecasts how competitor prices are likely to change over time, to add a review analysis feature that breaks down what customers like and dislike about each product in detail, and to make the tool available as an online service that multiple sellers can use at the same time. These additions would transform the current project into a production-ready competitive intelligence service.

References

[1] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Advances in Neural Information Processing Systems*, 2022.

[2] Y. Li, S. Ma, X. Wang, S. Huang, C. Jiang, H.-T. Zheng, P. Xie, F. Huang, and Y. Jiang, "EcomGPT: Instruction-tuning large language models with chain-of-task tasks for e-commerce," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

[3] N. Chen and H.-T. Tsai, "Steering via algorithmic recommendations," *RAND Journal of Economics*, vol. 55, no. 4, pp. 501–518, 2024.

[4] G. Hoberg and G. Phillips, "Text-based network industries and endogenous product differentiation," *Journal of Political Economy*, vol. 124, no. 5, pp. 1423–1465, 2016.

[5] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel, "Image-based recommendations on styles and substitutes," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2015.

[6] G. V. Cormack, C. L. A. Clarke, and S. Büttcher, "Reciprocal rank fusion outperforms Condorcet and individual rank learning methods," in *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2009.

[7] F. Dell'Acqua, E. McFowland III, E. Mollick, H. Lifshitz-Assaf, K. C. Kellogg, S. Rajendran, L. Krayer, F. Candelon, and K. R. Lakhani, "Navigating the jagged technological frontier: Field experimental evidence of the effects of Artificial Intelligence on knowledge worker productivity and quality," *Organization Science*, 2026.

[8] W. Zhang, X. Li, Y. Deng, L. Bing, and W. Lam, "A survey on aspect-based sentiment analysis: Tasks, methods, and challenges," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 11, pp. 11 019–11 038, 2023.

[9] P. K. Kopalle, K. Pauwels, L. Y. Akella, and M. Gangwar, "Dynamic pricing: Definition, implications for managers, and future research directions," *Journal of Retailing*, vol. 99, no. 4, pp. 580–593, 2023.

[10] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, 2000.