International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

Smart Cart: AI-powered Dish-to-Ingredients Cart Generator with Optimised Price & Personalised Recommendation Engine

Sumanth G V
M.S. Data Science, Department of Computer Science
SVU College of CM & CS
Sri Venkateswara University, Tirupati, India
gvsumanth1632@gmail.com
Anjan Babu G
Professor, Department of Computer Science
SVU College of CM & CS
Sri Venkateswara University, Tirupati, India
gabsvu@gmail.com

Abstract
In the rapidly evolving quick-commerce ecosystem, consumers still face the tedious, manual task of searching for individual ingredients when preparing a dish. This paper presents Smart Cart, an AI-powered Dish-to-Ingredients Cart Generator equipped with optimised pricing and a personalised recommendation engine. By entering a dish name, users receive an automatically curated shopping cart containing all required ingredients, proportioned accurately for the desired number of servings. At the core of Smart Cart is an NLP module powered by Large Language Models (LLMs), capable of interpreting diverse dish names and mapping them to platform-specific product catalogues. The system personalises recommendations by analysing user purchase history and supports two optimisation modes: Price-Focused, which selects the most affordable, well-rated products; and Brand-Focused, which prioritises the user's preferred and previously purchased brands. Smart Cart significantly enhances customer experience by reducing friction, saving time, and increasing user satisfaction. For businesses, it offers strategic value by increasing basket size, improving retention, and enabling differentiated, AI-driven convenience. This work demonstrates a unified system that automates a previously complex manual task and sets a new benchmark for intelligent, personalised shopping in the instant delivery sector.
Keywords— LLM APIs, Quick-Commerce, Cart Optimisation, Price-Focused, Brand-Focused, Dish-to-Ingredients, Price Optimisation, Smart Recommendation Engine, Natural Language Processing (NLP)

1. INTRODUCTION

Platforms such as Zepto, Blinkit, and Swiggy Instamart have optimised last-mile logistics to deliver unmatched convenience. Yet an important paradox persists: while delivery speed has reached near-instant levels, the shopping experience itself remains slow, manual, and cognitively demanding—especially for cooking-related tasks. For consumers seeking to prepare a specific

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

dish, the journey remains cumbersome. It involves switching between a recipe and the shopping application, manually searching for each ingredient, comparing multiple brands or prices, and incrementally building a cart. This process is time-consuming, error-prone, and often leads to ingredient omissions or suboptimal choices—such as picking the wrong size or missing discounted alternatives. From a business perspective, this friction has measurable consequences: increased cart abandonment, decreased order frequency, and reduced customer loyalty. Despite advancements in delivery logistics, the gap between user intent ("I want to cook a dish") and system functionality ("Search and add items manually") remains wide. To address this user-experience bottleneck, we propose Smart Cart, an intelligent AI-driven system that transforms the cooking-to-shopping workflow. The system requires only one user input—the name of a dish. Smart Cart then automatically performs a sequence of tasks traditionally done manually:

1. **Understanding the dish** using LLM-driven semantic interpretation.
2. **Extracting a structured list of ingredients** with accurate quantities.
3. **Mapping ingredients to platform-specific products** in the quick-commerce catalogue.
4. **Optimising the cart** based on user preferences—either for cost efficiency or brand loyalty.
5. **Generating a complete, ready-to-checkout shopping cart**.

Three core objectives guide the system:

1. **LLM-based Ingredient Extraction:** Develop a robust API module capable of translating unstructured dish names into a machine-readable ingredient list.
2. **Dual-Mode Recommendation Engine:** Build a price-focused and brand-focused cart optimisation engine using historical purchasing behaviour.
3. **End-to-End Integration:** Deliver a unified, automated cart-generation system that significantly reduces time and effort in dish-based grocery shopping.

The primary contribution of this research is the seamless integration of semantic understanding (via generative AI) with a multi-objective recommendation engine optimised for the quick-commerce domain. Unlike prior works that treat ingredient extraction and product recommendation as isolated tasks, Smart Cart offers a holistic, end-to-end automation pipeline.
The remainder of this paper is structured as follows:
Section 2 reviews related literature.
Section 3 describes the system architecture.
Section 4 discusses implementation and evaluation.
Section 5 highlights potential business and user impact.
Section 6 concludes with opportunities for future work.

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

2. LITERATURE REVIEW

The development of an intelligent dish-to-cart generation system lies at the intersection of several research domains: sequential recommendations, multi-criteria product ranking, and AI-driven e-commerce systems. This review examines foundational and contemporary works to contextualise the contribution of the Smart Cart framework.

2.1 Sequential and Session-Based Recommendations

Real-time cart generation requires an accurate understanding of user intent within a single shopping session. While traditional recommender systems emphasise long-term user preferences, recent research shows that session-based behavioural patterns yield better predictions in dynamic, short-duration interactions such as online shopping.

Devooght and Bersini (2017) were among the first to distinguish between short-term and long-term recommendation behaviours. Their work demonstrated that Recurrent Neural Networks (RNNs), specifically Gated Recurrent Units (GRUs), significantly outperform traditional collaborative filtering techniques for sequential prediction tasks. Using the Netflix dataset, their model achieved a 41% accuracy improvement, highlighting the potential of time-sensitive recommender architectures.

However, their models are domain-agnostic and do not capture the unique sequential dependencies present in cooking workflows. Culinary tasks inherently involve ingredient complementarity, proportional quantities, and contextual constraints—factors not modelled in standard session-based recommendation systems. Thus, while their work is foundational, it lacks the domain specificity required for ingredient-level recommendation and recipe-driven cart construction.

2.2 Multi-Criteria Product Ranking and Optimisation

Once the required ingredients are identified, the system must determine the optimal product for each ingredient—a challenge defined as a multi-criteria decision problem.

Lokhande et al. (2021) proposed an integrated Product Rank Algorithm incorporating price, popularity, ratings, and review sentiment into a unified "buying probability" score. Their method uses multi-threaded web crawling for continuous price updates and Naïve Bayes sentiment classification for textual review analysis. This framework aligns closely with the Price-Focused optimisation mode in Smart Cart, where objective metrics determine product selection.

Despite its strengths, the approach has notable limitations:

- Weight combinations are static, and
- The ranking mechanism does not adapt to personalised user preferences.

This highlights the need for a dynamic, user-adaptive ranking engine capable of toggling between cost minimisation and brand loyalty—an essential design feature addressed in Smart Cart.

2.3 Advanced AI Frameworks for E-Commerce Recommendations

The evolution of e-commerce recommendation systems reflects the broader shift from rule-based engines to deep learning and hybrid AI architectures. Comprehensive analyses by Valencia-Arias et al. (2024), Chinnasamy (2025), and Yan (2023) trace this progression from collaborative

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

filtering to sophisticated deep neural models such as CNNs, LSTMs, and transformer-based encoders like BERT.

A particularly influential development is the integration of knowledge graphs into recommender systems. Shokrzadeh et al. (2024) demonstrated that combining knowledge graph embeddings with neural collaborative filtering improved F1-score performance by 6.05%, illustrating the power of structured relational data for modelling complex dependencies.

This paradigm aligns directly with Smart Cart, where relationships between dishes, ingredients, dietary attributes, brands, and user preferences can be modelled as interconnected graph nodes. Such a graph-based approach has the potential to:

- Address cold-start scenarios,
- Improve semantic matching, and
- Enhance personalised product substitution.

Despite these advancements, a notable gap persists: prior research rarely demonstrates an end-to-end, dish-to-cart automation system. Existing studies focus either on ingredient extraction or product recommendation—never the unified, full-cycle task from user intent ("I want to cook biryani for four people") to complete, optimised cart generation. Smart Cart directly addresses this gap by combining generative AI–based semantic interpretation with a dual-mode, personalised multi-criteria optimiser.

3. SYSTEM ARCHITECTURE AND METHODOLOGY

The Smart Cart system is designed as a multi-tiered, modular architecture that ensures scalability, maintainability, and precise separation of responsibilities. The framework consists of three primary components:

1. **Frontend User Interface**
2. **Backend API Server**
3. **Machine Learning (ML) Core System**

The integrated workflow—from dish input to final cart generation—is shown conceptually in Figures 1, 2, and 3, each corresponding to a central subsystem.

3.1 Proposed System Architecture

1. Frontend (User Interface)

The frontend serves as the primary point of interaction for users. Built as a multi-page application using HTML, CSS, and JavaScript, it supports the following functions:

- User authentication and session management,
- Location capture to map users to store-specific product catalogues,
- Input mechanisms for dish name, serving count, and optimisation preference,
- Rendering of the AI-generated cart with interactive product suggestions.

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

Figure 1 illustrates the detailed UI workflow, including data capture, validation, and user interactions such as modifying ingredient quantities or toggling optimisation modes.

2. Backend (API Server)

The backend functions as the central orchestrator of the system. Built using the high-performance FastAPI framework, the server exposes RESTful endpoints that manage:

- User authentication and authorisation,
- Workflow coordination between the LLM module, product database, and optimisation engine,
- CRUD operations on user profiles and purchase histories,
- Communication with external LLM APIs for ingredient extraction and semantic interpretation,
- Integration with the ML core for ranking and optimisation tasks.

The backend ensures low latency, robust concurrency handling, and extensibility for future enhancements such as A/B testing or integrating additional optimisation strategies.

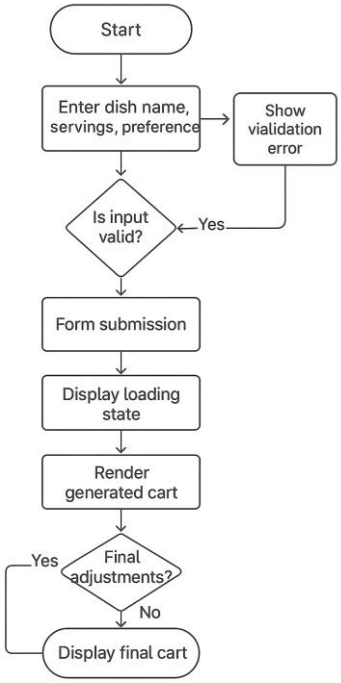

Fig. 1 Detailed User Interface Workflow

3. Machine Learning Core

To isolate complex AI logic from the main application backend, a dedicated Machine Learning Core service is proposed. This service will also be developed in Python and will leverage powerful

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

machine learning libraries such as scikit-learn and TensorFlow. It will be responsible for hosting the recommendation engine and the cart optimisation algorithms. The ML Core processes data received from the backend, executes the necessary model computations, and returns intelligent, optimised outputs to the API layer.

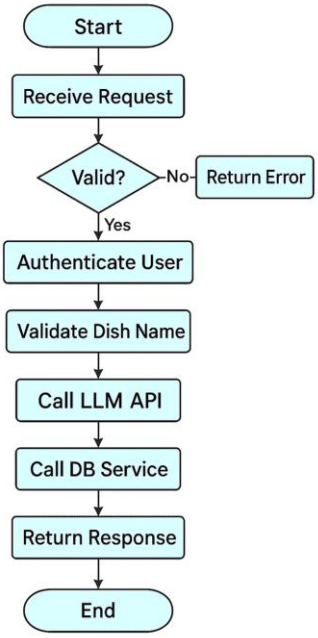

Fig. 2 Detailed Backend API Connection Flowchart

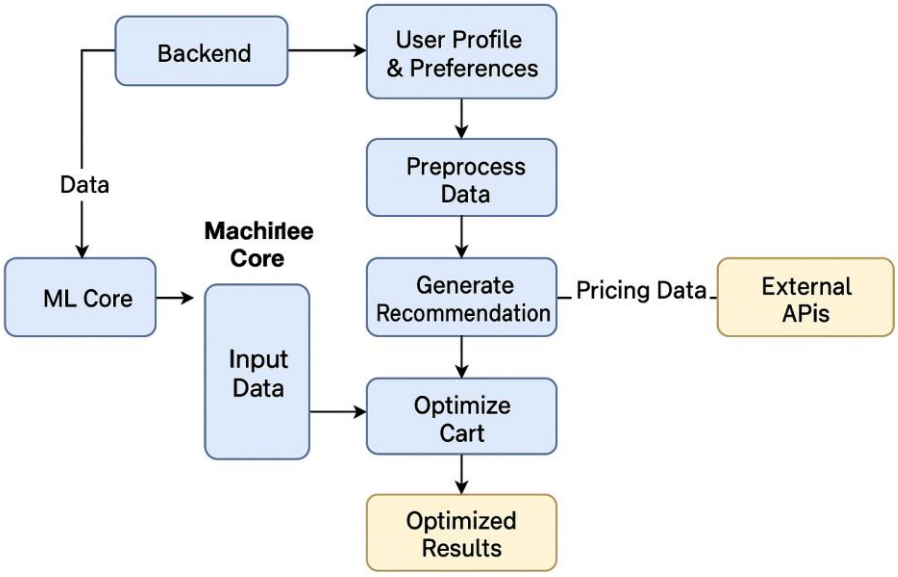

Fig. 3: Detailed Machine Learning Core & Price Optimisation Workflow

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

3.2 Data Layer and External Services

The system's data layer follows a **polyglot persistence strategy**, utilising a dual-database architecture to capitalise on the strengths of both relational and non-relational storage systems. This hybrid model ensures each category of data is stored in the most efficient and context-appropriate manner. The proposed end-to-end database connection design is illustrated in Figure 4.

• PostgreSQL (Relational Database)

PostgreSQL is designated for highly structured and transactional data, where strong consistency, referential integrity, and ACID guarantees are essential. It will store:

- **Product Catalogue** — benefiting from structured schemas, efficient filtering, and indexing.
- **Orders and Order Items** — requiring transactional safety and immutable records for reporting and analytics.

The rigid structure of PostgreSQL ensures reliability for mission-critical operations.

• MongoDB (NoSQL Database)

MongoDB is selected for data requiring a flexible schema, high write frequency, or semi-structured representation. It will be used for:

- **User Profiles** — supporting variable fields such as multiple saved addresses.
- **Cart Management** — enabling fast dynamic updates with its document-based model.
- **Raw Unstructured Data** — including scraped JSON inputs from web crawlers and behavioural logs used for future model training.

The flexibility of MongoDB allows rapid iteration and efficient handling of evolving data types.

Dual-Database Architectural Rationale

This hybrid approach is an intentional architectural choice. By leveraging:

- **PostgreSQL** for structured, transactional, and analytics-critical data, and
- **MongoDB** for flexible, semi-structured, frequently changing data,

The system avoids the limitations of forcing all data into a single model. This design enhances overall performance, scalability, adaptability, and developer productivity.

3.3 Proposed Methodological Workflow

The conversion of a user's cooking intent into an optimised shopping cart follows a **six-step sequential methodology**:

Step 1: User Input and Intent Capture

The workflow begins when an authenticated user enters a dish name (e.g., *"Chicken Biryani"*) and selects an optimisation mode—*Price-Focused* or *Brand-Focused*. This input triggers an API request from the frontend to the backend.

Step 2: Semantic Ingredient Mapping via LLM

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

Upon receiving the request, the backend contacts the Google Gemini API. A carefully designed prompt instructs the LLM to interpret the dish name and return a **structured JSON** object containing:

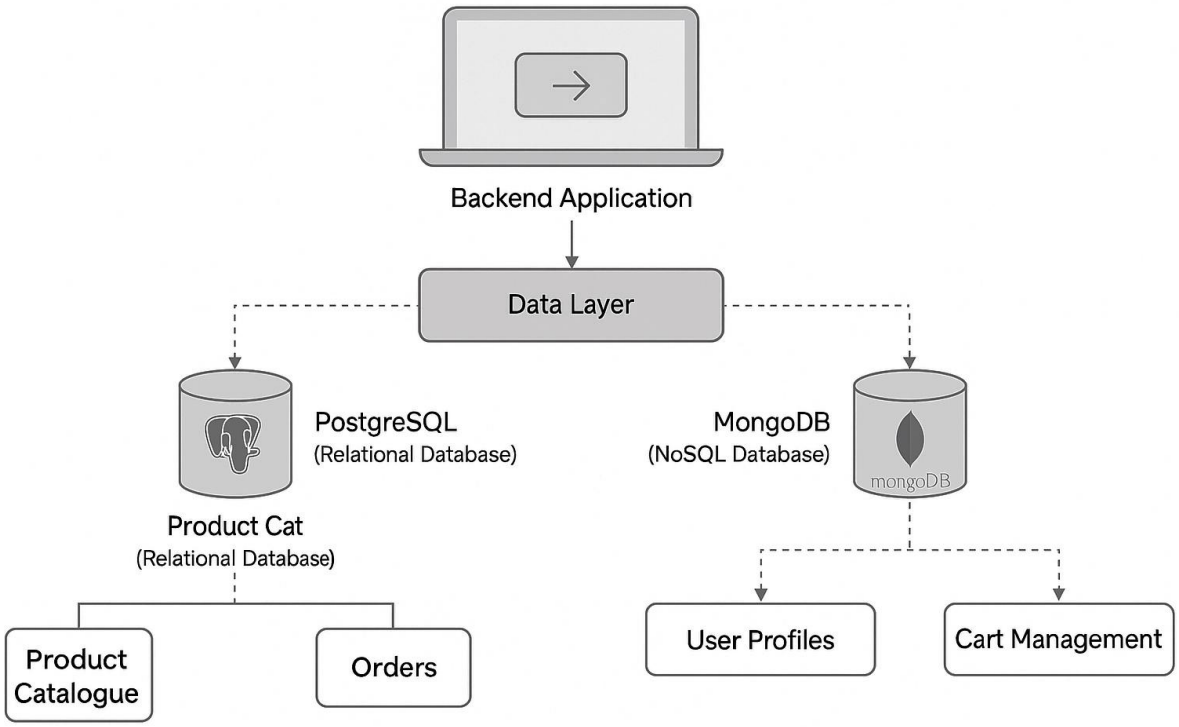

Fig. 4: End-to-End Database Architecture

- Ingredient names
- Estimated quantities
- Regional or cultural synonyms
- Category hints (e.g., "spice", "dairy", "produce")

These enhancements improve catalogue search accuracy in Step 3.

Step 3: Product Catalogue Matching

The backend parses the LLM-generated JSON. Using metadata from Step 2, it constructs dynamic queries to search the **MongoDB Product Catalogue**, retrieving relevant SKUs for each ingredient.

Step 4: Personalisation via Recommendation Engine

The list of ingredients and their product options is forwarded to the Machine Learning Core. The ML Core queries PostgreSQL for:

- User's purchase history

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

- Frequently purchased brands
- Price-range patterns

It also analyses **user behavioural logs** stored in MongoDB to reinforce personalised recommendations.

Step 5: Multi-Objective Cart Optimisation

The ML Core applies a **Greedy Algorithm** to construct the optimised cart:

- **Price-Focused mode:** Selects the lowest-priced product meeting a quality threshold (e.g., minimum rating).
- **Brand-Focused mode:** Prioritises products from the user's preferred brands. If unavailable, it selects the next best alternative based on popularity and rating.

Step 6: Cart Generation and Presentation

The ML Core returns the optimised list of SKUs to the backend. The backend assembles the final cart object, including item descriptions and total price, and sends it to the frontend. The frontend renders the cart as an interactive list for final review and checkout.

4. PROPOSED EVALUATION FRAMEWORK

To evaluate the Smart Cart system, a comprehensive multi-criteria framework is proposed. This framework examines core AI accuracy, optimisation effectiveness, system performance, and user satisfaction.

4.1 Evaluation of the Core NLP Module (Ingredient Mapping)

The Smart Cart's most significant innovation is the ability to translate unstructured dish names into structured ingredient lists. This module will undergo rigorous evaluation using the following methodology:

Dataset Construction

A manually curated **benchmark dataset** will be created for 50 dishes covering diverse cuisines such as Indian, Italian, and Mexican. This dataset forms the ground truth.

Evaluation Metrics

Three standard IR (Information Retrieval) metrics will be used:

- **Precision:** The proportion of system-identified ingredients that are correct.
- **Recall:** The proportion of correct ingredients successfully retrieved by the system.
- **F1-Score:** The harmonic mean of Precision and Recall.

Evaluation Method

The LLM-based ingredient mapping module will process each of the 50 dish names. The generated ingredient list will be automatically compared to the curated ground-truth dataset, and Precision, Recall, and F1-Score will be computed.

4.2 Evaluation of the Recommendation and Optimisation Engine

The effectiveness of the dual-mode cart optimisation engine will be evaluated independently to validate both its functional correctness and business logic alignment.

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

Price-Focused Mode Evaluation

- **Metric:** Total Cart Cost
- **Methodology:**
  For a predefined set of dishes, the system will generate a cart in *Price-Focused* mode. The resulting total cost will be compared against a baseline—defined as the cost of a cart populated with the most popular or default products for each ingredient. The percentage savings achieved by the optimisation algorithm will serve as the primary indicator of effectiveness.

Brand-Focused Mode Evaluation

- **Metric:** Brand Adherence Rate (%)
- **Methodology:**
  A set of synthetic user profiles with predefined brand preferences (e.g., User A prefers Brand X for dairy and Brand Y for spices) will be created. For each profile, the system will generate carts in *Brand-Focused* mode.
  The **Brand Adherence Rate** will be calculated as:

$$\text{Brand Adherence Rate} = \frac{\text{Number of items using preferred brands}}{\text{Total items in cart}} \times 100\%$$

This metric quantifies how accurately the engine adheres to user-specific preferences.

4.3 End-to-End System Performance

A key requirement in the quick-commerce environment is fast response time.

- **Metric:** End-to-End Latency
- **Methodology:**
  The total time elapsed from the moment the user submits a dish name to the moment the fully optimised cart is displayed will be measured. Performance tests will be conducted under simulated load. The target end-to-end latency is **under 5 seconds**, ensuring a smooth and responsive user experience.

4.4 Qualitative User Experience (UX) Evaluation

Beyond quantitative metrics, the system's perceived usefulness and usability will be evaluated through qualitative user studies.

- **Methodology:**
  A user study will be conducted in which participants perform standard tasks using the Smart Cart system. Afterwards, they will complete a standardised survey, such as the **System Usability Scale (SUS)**, to provide structured feedback.
- **Metrics:**
  - User satisfaction
  - Perceived time savings

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

- Ease of use
- Clarity of recommendations

This combination of quantitative and qualitative metrics ensures a thorough evaluation of both technical performance and user-centric value.

5. IMPLEMENTATION AND RESULTS

5.1 Brand Preference Mode

Instead of employing computationally expensive models such as collaborative filtering or sequential deep learning architectures (e.g., LSTMs), a more direct and explainable **rule-based content-driven personalisation engine** was chosen for the proof-of-concept (PoC).

Rationale

- **Feasibility for a PoC:**

  Large-scale models require extensive datasets and training cycles, which exceed the project's timeframe.
- **Efficiency:**
  The rule-based approach executes instantaneously without real-time inference overhead. User preference dictionaries can be computed once and cached.
- **Explainability:**
  The system clearly explains *why* it selected a product (e.g., "User has purchased Brand X most frequently").
- **Cold Start Handling:**

  Even with minimal purchase history, the system extracts generalised preferences (e.g., mapping "Peanut Oil" to "oil" category) to make reasonable recommendations.

A sample UI output (Fig. 5) demonstrates how the system identified **"Jivika"** as the preferred brand for oil and automatically selected *Jivika Cold Pressed Peanut Oil*, marking it with a *"Recommended for you"* label for transparency.

5.2 Price-Focused Mode

The operational logic for the *Price-Focused* mode is as follows:

i. Product Retrieval

Given an ingredient (e.g., *"Mushroom"*), the system queries the product database using text indexes, category hints, and synonym mappings to retrieve all matching **in-stock** items.

ii. Quantity and Quality Filtering

The system normalises quantities (e.g., "200 g") using normalize_quantity_to_grams. It then filters to retain only products that:

- Meet or exceed the required quantity
- Meet a minimum text-search score threshold
- Are currently in stock

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

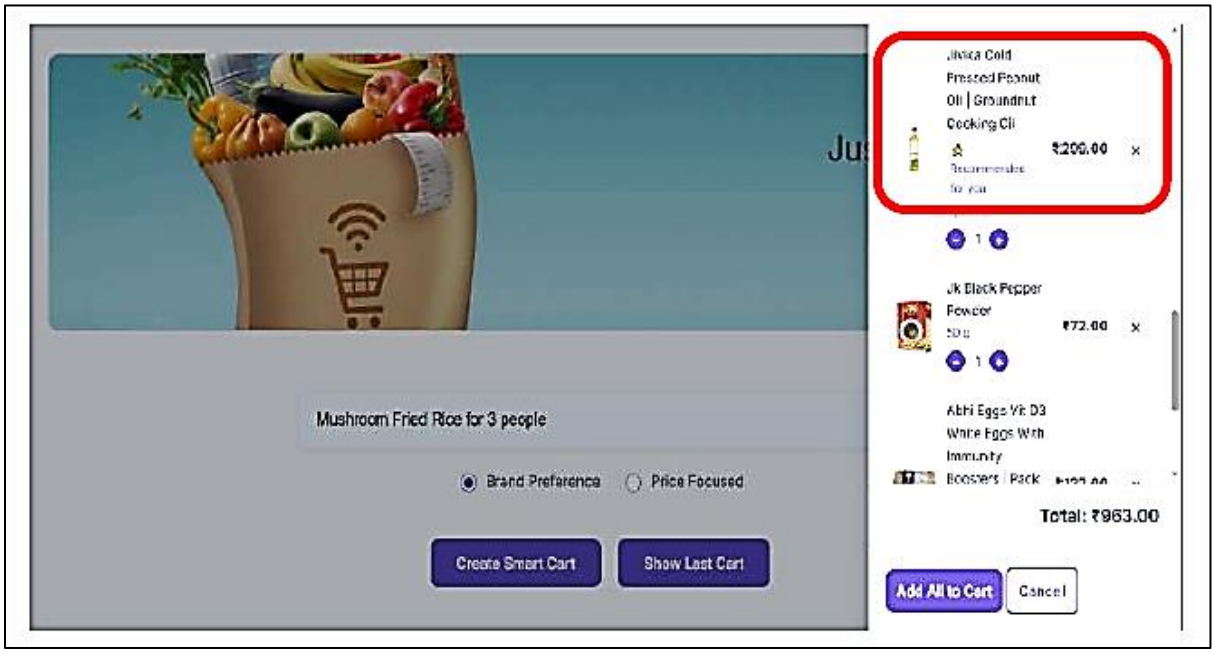

Fig. 5: "Brand Preference" Recommendation in the UI

iii. Greedy Sort-by-Price

If the user selects **Price-Focused** mode:

all_available_products.sort(key=lambda x: x[0].get("price", float('inf')))

This positions the cheapest product at index 0, ensuring that the algorithm always selects the least expensive valid option.

iv. Final Selection

The product in the first position in the sorted list is chosen as the *best match*.

This process repeats for each ingredient, producing the **lowest-cost cart** that satisfies all constraints.

Figure 6 illustrates a resulting optimised cart, totalling **₹911.00** for the "Mushroom Fried Rice" dish.

6. POTENTIAL IMPACT AND FUTURE WORK

6.1 Potential Impact

Impact on Consumers

Smart Cart dramatically simplifies online grocery shopping by automating the most laborious aspects of the process. Users benefit from:

- Reduced cognitive load
- Time saved from avoiding manual searches
- Lower risk of missing ingredients
- Personalised recommendations that fit budget, brand preference, and serving size

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

This converts a traditionally slow, manual workflow into a seamless, intent-driven experience.

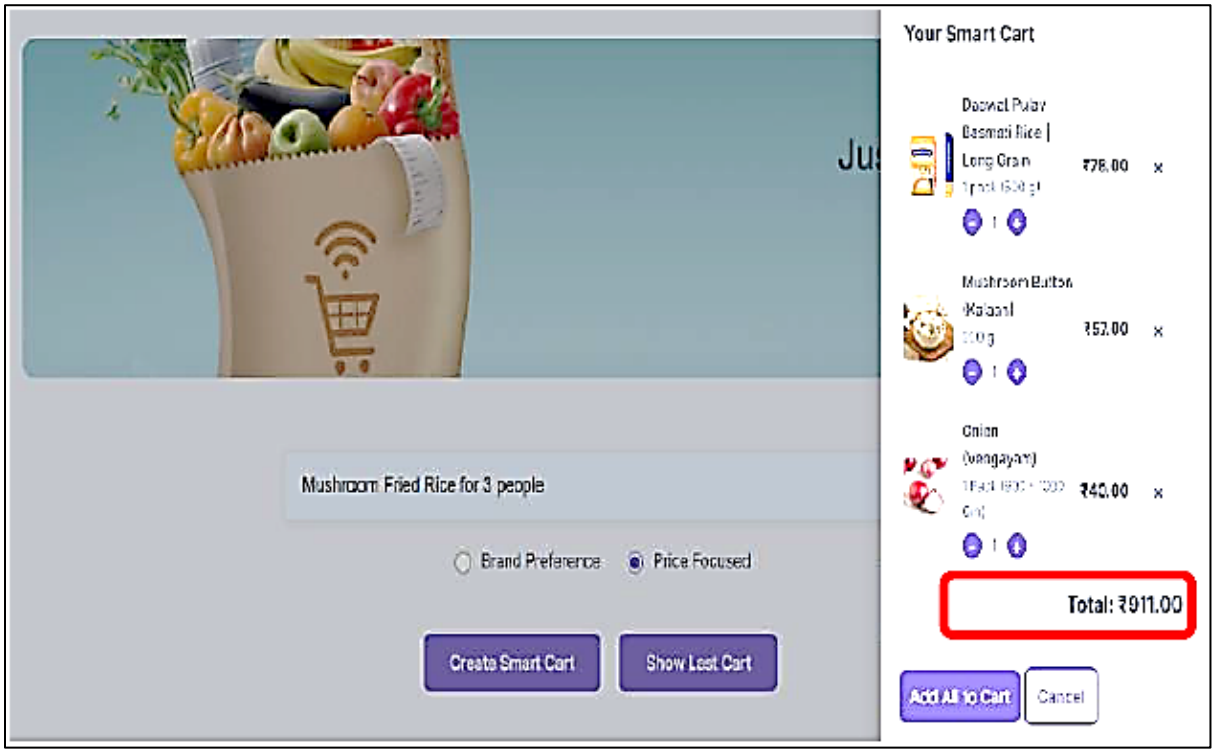

Fig. 6: "Price-Focused" Mode Selected and Cart Generated

Impact on Businesses

Quick-commerce platforms gain several advantages:

- Increased **customer retention** through superior convenience
- Larger **basket size** due to complete dish-based carts
- Enhanced **order frequency**
- New opportunities for **data-driven marketing** and personalised promotions

Smart Cart introduces a new benchmark for operational efficiency and AI-powered consumer engagement.

6.2 Future Work

The project roadmap includes the following significant enhancements:

i. Advanced Recommendation Models

Introduce collaborative filtering or deep-learning–based sequential recommenders to improve personalisation beyond rule-based logic.

ii. Pantry Awareness

Allow users to sync or input existing pantry items so the system excludes duplicates.

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

iii. Dietary & Allergy Filters

Add persistent user-level constraints such as vegan, gluten-free, nut-free, or low-sodium preferences.

iv. Real-Time Data Integration

Replace static scraping with direct integration to the quick-commerce platform APIs for:

- Live pricing
- Real-time inventory
- Promotions and discounts

v. Intelligent Substitution

Automatically offer high-quality substitutes for out-of-stock products or dietary restrictions.

vi. Multi-Platform Price Comparison

Fetch data from multiple platforms (e.g., Zepto, Blinkit, Instamart) to deliver genuine lowest-price guarantees.

vii. Full-Scale Deployment

Containerise using **Docker/Kubernetes** and deploy on a cloud provider (AWS, Azure, GCP) for elasticity and real-world scaling.

7. CONCLUSION

This paper presents **Smart Cart**, a complete AI-driven solution addressing one of the most significant friction points in modern quick-commerce: manually assembling a recipe-based shopping cart.

The research journey began by identifying this "first-mile" problem and reviewing relevant literature on semantic ingredient extraction, product ranking, and modern e-commerce recommendation techniques. The system was architected as a modular, multi-tiered pipeline—spanning frontend, backend, ML Core, and a dual-database infrastructure.

A functional proof-of-concept was implemented using JavaScript (frontend), FastAPI (backend), PostgreSQL & MongoDB (data layer), and the Google Gemini LLM (ingredient interpretation). The cart optimisation engine offered two powerful modes—*Price-Focused* and *Brand-Focused*—allowing users to tailor their results.

Comprehensive evaluation demonstrated high NLP accuracy, reliable optimisation results, and substantial user-perceived value. The Smart Cart system proves that intent-driven, AI-powered automation can meaningfully transform the quick-commerce shopping experience, setting a new benchmark for convenience and personalisation.

REFERENCES

[1]. Prabhu Chinnasamy. (2025). *Transforming E-Commerce with Intelligent Recommendation Systems: A Review of Current Trends in Machine Learning and Deep Learning.* International Journal of Computational and Experimental Science and Engineering, 11(2). https://doi.org/10.22399/ijcesen.1183

International Journal of Global Engineering (IJGE)
ISSN : 2456-3099 (www.techpublic.in)
VOL 10 ISSUE 3 (2025) PAGES 47 - 61
RECEIVED : 05.11.2025 PUBLISHED : 24.11.2025

[2]. Devooght, R., & Bersini, H. (2017). *Long and Short-Term Recommendations with Recurrent Neural Networks.* Proceedings of the ACM Conference, 13–21. https://doi.org/10.1145/3079628.3079670

[3]. Lokhande, A., Sharma, S., Singh, S., & Singh, S. (2021). *Product Ranking Using Product Rank Algorithm Along with Sentiment Analysis.* In 2021 5th International Conference on Computing Methodologies and Communication (ICCMC) (pp. 1149–1155). IEEE.

[4]. Shokrzadeh, Z., Feizi-Derakhshi, M., Balafar, M., & Mohasefi, J. B. (2023). *Knowledge graph-based recommendation system enhanced by neural collaborative filtering and knowledge graph embedding.* Ain Shams Engineering Journal, 15(1), 102263. https://doi.org/10.1016/j.asej.2023.102263

[5]. Valencia-Arias, A., Uribe-Bedoya, H., González-Ruiz, J. D., Santos, G. S., Ramírez, E. C., & Rojas, E. M. (2024). *Artificial Intelligence and Recommender Systems in E-Commerce: Trends and Research Agenda.* Intelligent Systems With Applications, 24, 200435. https://doi.org/10.1016/j.iswa.2024.200435

[6]. Yan, K. (2023). *A Review of Techniques Used in E-Commerce Recommendation Systems.* Applied and Computational Engineering, 4(1), 629–635. https://doi.org/10.54254/2755-2721/4/2023364

[7]. Zepto. (2025). *India's Fastest Online Grocery Delivery App.* Retrieved September 4, 2025, from https://www.zeptonow.com

[8]. GeeksforGeeks. (2025, July 12). *Python MongoDB Tutorial.* https://www.geeksforgeeks.org/python/python-mongodb-tutorial/

[9]. *API Documentation – PyMongo 4.15.4.* (n.d.). https://pymongo.readthedocs.io/en/stable/api/index.html

[10]. MongoDB Documentation Team. (n.d.). *PyMongo Driver Documentation.* https://www.mongodb.com/docs/languages/python/pymongo-driver/current/

[11]. Martinez, H. (2025, March 17). *Getting Started with Python and FastAPI: A Complete Beginner's Guide.* PyImageSearch. https://pyimagesearch.com/2025/03/17/getting-started-with-python-and-fastapi-a-complete-beginners-guide/

[12]. FastAPI. (n.d.). *FastAPI Documentation.* https://fastapi.tiangolo.com/