International Journal of Advances in Engineering and Management (IJAEM)
Volume 8, Issue 5 May. 2026, pp: 69-79 www.ijaem.net ISSN: 2395-5252

NovaKart: An AI-Powered Agentic E-Commerce System

Abhishek Choudhary, NeerajRana, Shivam Sharma
Department of CSE-AIML,
Dr. A.P.J. Abdul Kalam Technical University (AKTU)
Sector11, Jankipuram Vistar Yojna, AKTU Campus, CDRI Road, Naya Khera,
Lucknow – 226 031, Uttar Pradesh, India

Date of Submission: 25-04-2026 Date of Acceptance: 04-05-2026

Abstract—
Traditional e-commerce systems rely heavily on static keyword search, rule-based recommendations, and manual customer support. These limitations reduce search relevance, affect user experience, and restrict intelligent product discovery. This paper proposes **NovaKart**, an **agentic-powered e-commerce platform** integrating Large Language Model (LLM) agents, semantic vector search, and multimodal retrieval to enable autonomous shopping assistance. NovaKart combines a React frontend, a Spring Boot backend, PostgreSQL with PGVector, CLIP embeddings, and Qwen models to support agent-driven product search, personalized recommendations, conversational buying, and image-based retrieval. The system uses multiple cooperating AI agents—planner, search agent, recommendation agent, and image understanding agent—to dynamically analyze user intent and perform tool-based reasoning. Experimental results show significantly improved search accuracy, reduced query latency, and enhanced user satisfaction. This research demonstrates how agentic AI can transform modern e-commerce into autonomous, multimodal, and intelligent digital shopping ecosystems.

Keywords—
Agentic AI, Vector Search, PGVector, CLIP Embeddings, Qwen LLM, Multimodal Search, Spring Boot, Autonomous Agents, E-commerce Intelligence

I. INTRODUCTION

E-commerce platforms traditionally depend on keyword-based search, category navigation, and static rule-based recommendation systems. These techniques often fail to understand user intent, leading to irrelevant search results, limited personalization, and poor engagement. As product catalogs expand, keyword search becomes insufficient because it cannot capture semantic relationships or natural-language queries. Additionally, traditional recommendation engines suffer from cold-start limitations, requiring large user histories before generating meaningful suggestions.

With the evolution of **Large Language Models (LLMs)** and **multimodal AI**, intelligent shopping experiences have become possible through conversational search, intent analysis, vector embeddings, and autonomous agent reasoning. Agentic AI systems can decompose complex tasks, call tools, interact with APIs, retrieve database information, and autonomously perform multi-step operations. These capabilities enable a new generation of intelligent e-commerce platforms.

To address the limitations of current systems, we present **NovaKart**, an **agentic-powered e-commerce system** using vector search, multimodal embeddings, and autonomous LLM agents. NovaKart is built using React.js (frontend), Spring Boot (backend), PostgreSQL with PGVector (database), CLIP embeddings for image-text similarity, and Qwen-based multimodal LLMs for conversational shopping. The system provides semantic search, image-based product discovery, intelligent recommendations, and AI-driven customer interaction.

The contributions of this work are as follows:

1. Design and implementation of a **multi-agent e-commerce architecture** using LLM-powered autonomous agents.
2. Integration of **PGVector** for high-accuracy semantic product retrieval.
3. Implementation of **multimodal product search** using CLIP image embeddings.
4. Agentic workflow enabling planning, tool calling, and reasoning for contextual shopping assistance.
5. A full-stack architecture enabling scalable and real-time AI-driven commerce.
6. Evaluation of vector search accuracy, multimodal matching, and agent response performance.

II. LITERATURE SURVEY

The development of intelligent e-commerce systems has been influenced by major advancements in

search technology, **representation learning**, and **Large Language Models (LLMs)**. Traditional search engines used in e-commerce platforms rely primarily on **keyword-based retrieval** using inverted indices, such as Elasticsearch or Lucene. These systems are effective only when users type exact or near-exact terms so they can match word tokens, but they fail at understanding semantics or user intent, especially in queries like *"warm clothes for snowfall"* which may not directly match product descriptions[13].

With the rapid rise of **vector embeddings** and **semantic search**, modern platforms have begun shifting from keyword-based retrieval to **vector space retrieval**. Embedding-based search systems represent text or images as high-dimensional vectors, allowing similarity to be computed mathematically using cosine or Euclidean distance. Research conducted by Facebook AI on billion-scale similarity search demonstrated the effectiveness of vector retrieval even at extremely high scale[10]. Likewise, Pinecone and Milvus documented how vector databases outperform lexical search in tasks requiring contextual relevance[11][12].

The introduction of **PGVector**, a PostgreSQL extension for storing and querying vector embeddings, significantly simplified the integration of semantic search into traditional relational systems[9]. This hybrid approach enables e-commerce platforms to maintain relational integrity (orders, carts, users) while also supporting high-dimensional embedding similarity search[8].

In the domain of embeddings, **multimodal representation learning** gained traction with the introduction of **CLIP (Contrastive Language-Image Pre-training)** by OpenAI[2]. CLIP demonstrated that image and text embeddings can be placed in the same vector space, enabling cross-modal search (e.g., searching for products using a photo). This breakthrough influenced many modern visual search systems used in e-commerce, such as Amazon StyleSnap and Myntra Smart Lens.

Alongside CLIP, text embedding models like **Nomic Embed** introduced robust semantic representations for text queries and product descriptions, supporting high-accuracy retrieval in e-commerce scenarios[5]. The combination of textual and visual embeddings forms the foundation of **multimodal retrieval**, which NovaKart uses for image-based product recommendations.

Parallel to advancements in embeddings, the field of **Large Language Models (LLMs)** has transformed how applications interpret user intent. Models such as Qwen^[1], GPT-4, and LLaMA^[6] provide deep reasoning capabilities and can decompose complex natural language prompts into structured actions. These models introduced the possibility of "**agentic workflows**," where an LLM acts not only as a conversational engine but as a **decision-making system** that selects and executes tools autonomously[16].

Research on **AutoGen by Microsoft** explored how multiple AI agents can collaborate to complete complex tasks, a concept now widely adopted in LLM-based systems[16]. Google DeepMind further demonstrated how autonomous LLM agents can plan, reason, and perform multi-step operations in real-world environments[17]. Systems like LangChain simplified practical implementation of multi-agent orchestration patterns[18].

Retrieval-Augmented Generation (RAG) has become the gold standard for building context-aware applications. The foundational research paper by Lewis et al. introduced RAG as a method to combine external knowledge with LLM reasoning, producing highly factual responses compared to standalone LLMs[13]. Meta AI later published systematic improvements to RAG workflows, highlighting their importance in enterprise-grade search and decision-making systems[15].

In parallel, the backend infrastructure for AI-driven systems evolved significantly. Frameworks like **Spring Boot** provide enterprise-level scalability and make it easier to orchestrate API flows, agent actions, and database interactions within microservice-like monolithic architectures[19]. This backend orchestration becomes even more critical when integrating remote AI inference endpoints, such as those hosted on Kaggle GPUs.

In summary, the literature shows a clear shift from static keyword-based systems toward **intent-aware, vector-driven, agentic AI architectures**. NovaKart builds directly upon these advancements by combining vector embeddings, multimodal retrieval, a hybrid SQL–vector datastore, and multi-agent LLM reasoning to create a next-generation e-commerce experience.

III. RELATED WORK

A. Traditional Search Systems

Conventional e-commerce search engines rely on keyword matching, TF-IDF, and inverted indexing. These methods perform poorly under semantic, incomplete, or ambiguous queries. Studies indicate that keyword-based retrieval lacks the contextual understanding required for modern shopping experiences[1][2].

B. Recommendation Algorithms

Collaborative filtering, matrix factorization, and hybrid recommenders have been widely used but

International Journal of Advances in Engineering and Management (IJAEM)
Volume 8, Issue 5 May. 2026, pp: 69-79 www.ijaem.net ISSN: 2395-5252

face cold-start challenges and lack real-time personalization[3][4].

C. Vector Databases & Semantic Search

Recent research in vector search (FAISS, Pinecone, PGVector) emphasizes improved semantic retrieval. PGVector integrates embeddings directly into PostgreSQL, offering production-level performance[5][6]. However, prior systems rarely integrate **agentic orchestration with vector search**.

D. CLIP & Vision-Language Models

CLIP provides robust image-text alignment and is widely used in image retrieval systems[7][8]. However, CLIP has not been deeply combined with multi-agent conversational shopping workflows.

E. Agentic AI and Autonomous Reasoning

Research on AutoGPT, ReAct, and Toolformer demonstrates LLMs' ability for reasoning, tool usage, and autonomous task execution[9][10][11]. These systems primarily focus on general tasks, not domain-specific e-commerce intelligence.

F. Conversational Commerce

LLM-powered shopping assistants have emerged, yet most lack backend integration, vector search, multimodal capabilities, and autonomous decision-making[12][13].

G. Identified Gaps in Literature

Existing systems fail to integrate:

agentic reasoning
semantic + multimodal search
vector databases in operational commerce
LLM-based dynamic tool-using agents

NovaKart addresses these gaps using a unified agentic approach.

IV. SYSTEM ARCHITECTURE

The architecture of **NovaKart**, an agentic-powered e-commerce platform, is designed as a modular, scalable, and AI-centric system. It integrates **frontend UI**, **backend services**, **database with vector capabilities**, **multimodal AI models**, and an **agentic reasoning layer** that coordinates all intelligent features. The complete architecture is divided into five major layers:

A. Frontend Layer (React.js + AI Chat Interface)

The frontend is responsible for all user-facing interactions. It is built with **React.js**, Vite, TailwindCSS, and ShadCN UI.

Key Responsibilities:

1. **Conversational Shopping Interface**
   - Users can chat with the AI, upload images, or type natural language queries ("Find me shoes like this," "Suggest jackets under ₹1500").
2. **Real-Time Product Search & Filter UI**
   - Product listing, sorting, filters, categories, and offers.
3. **User Account Features**
   - Login/Signup, orders, cart, wishlist.
4. **API Communication**
   - Communicates with the backend through REST APIs.
5. **Visual Feedback for AI Operations**
   - Typing indicators, search progress, recommended results panel.

Why React?

Its fast rendering, component reusability, and compatibility with chatbot UI frameworks make it ideal for an interactive e-commerce platform.

B. Backend Layer (Spring Boot Microservices)

The backend is built using **Spring Boot**, which handles all business logic, validation, and integration with AI services.

Major Backend Components:

1. API Gateway

Routes all requests from frontend → appropriate microservice.
Manages authentication & JWT validation.

2. Product Management Service

Adds, updates, deletes products.
Handles images, descriptions, categories.

3. User & Order Service

Manages users, carts, orders, payments.

4. Agent Orchestration Service

Connects backend with the AI models (Qwen, CLIP).
Receives user queries → forwards to agentic layer.
Receives agent-generated tasks → runs tools (search, recommend, filter).

5. Embedding & Vector Processor

Generates text, description, and image embeddings.
Stores vectors into PGVector tables.
Offers REST endpoints for:
- Semantic search
- Vector similarity ranking
- Image-to-product retrieval

6. Security Module

Uses Spring Security + JWT
Prevents unauthorized access.

C. Database Layer (PostgreSQL + PGVector)

The database is the backbone of semantic and multimodal search. **PostgreSQL** is used with the **PGVector extension**,

International Journal of Advances in Engineering and Management (IJAEM)
Volume 8, Issue 5 May. 2026, pp: 69-79 www.ijaem.net ISSN: 2395-5252

enabling storage and querying of high-dimensional embeddings.

Key Tables:

1. **Products**
   `(id, name, description, price, category, image_url, embedding VECTOR(768), image_embeddingVECTOR(512))`
2. **Users**
3. **Orders**
4. **Embeddings**
5. **Chat History / Agent Memory (optional)**

Why PGVector?

Supports cosine similarity search natively.
More cost-efficient and locally deployable than Pinecone or Weaviate.
Integrates seamlessly with Spring Boot + PostgreSQL.

D. AI Models Layer (CLIP + Qwen + Nomic Embeddings)

This layer performs all AI-related computations, including embeddings, language understanding, and multimodal reasoning.

1. Embedding Models

CLIP for image embeddings (512-d)
Nomic / Instructor for text embeddings (768-d)

2. LLM Models

Qwen2/Qwen3 4B/7B for:

- Planning
- Reasoning
- Tool calling
- Natural language interpretation

Qwen-VL for:

- Image + text reasoning
- Understanding user-uploaded images
- Recommendations based on visual features

3. Model Hosting

Hosted on **Kaggle Notebooks**, **local GPU server**, or **cloud instance**.
Accessible via dedicated API endpoints (Flask / FastAPI / Spring Boot integration).

E. Agentic Layer (Autonomous Multi-Agent System)

This is the most innovative part of NovaKart. The agentic system consists of multiple cooperating agents that mimic human-like decision-making.

1. Planner Agent

First agent to analyze every user message.
Extracts intent:

- "Find shoes under ₹2000"
- "Show me products similar to this image"

Breaks task into smaller subtasks.

2. Search Agent

Performs semantic / vector / keyword search.
Calls PGVector similarity API.
Filters results based on user constraints (price, color, category).

3. Recommendation Agent

Uses embeddings + user history to suggest items.
Learns preferences from chat and click patterns.

4. Image Understanding Agent

Uses CLIP/QWEN-VL to process uploaded images.
Extracts visual features → returns similar products.

5. Tool Execution Agent

Executes functions:

- `searchProducts()`
- `getProductByImage()`
- `recommendProducts()`
- `getCategories()`

6. Agent Reasoning Loop

Each agent uses **ReAct + Toolformer + AutoGPT-like techniques**:

1. Think → 2. Plan → 3. Act (tool) → 4. Observe → 5. Refine → 6. Respond

This creates a powerful autonomous system.

F. Data Flow Summary

1. **User interacts via chat or image upload.**
2. Query sent to **Spring Boot backend**.
3. Backend sends message to **Planner Agent**.
4. Planner creates a reasoning plan → assigns subtasks.
5. Agents perform semantic search, vector search, recommendations, etc.
6. Results returned to backend → formatted → sent to frontend.
7. Frontend renders:
   - Product lists
   - Recommendations
   - Explanations ("Here are items similar to your image")

G. System Qualities & Advantages

1. Scalability

Microservice design
Distributed agent layer
Independent model hosting

2. High Performance

Vector queries < 40ms

International Journal of Advances in Engineering and Management (IJAEM)
Volume 8, Issue 5 May. 2026, pp: 69-79 www.ijaem.net ISSN: 2395-5252

Fast React rendering
Caching of frequent searches

3. Flexibility

Models can be swapped (Qwen → LLaMA → GPT)
Agent count can be expanded

4. Multimodal Intelligence

Image + text + database + agent reasoning combined

V. METHODOLOGY

The methodology of NovaKart is designed to integrate **semantic vector search**, **multimodal AI models**, and **LLM-based agentic reasoning** to enable intelligent, autonomous e-commerce operations. The system follows a structured pipeline consisting of five primary stages:

1. **Data Preprocessing & Embedding Generation**
2. **Semantic Vector Search using PGVector**
3. **Multimodal Image-Based Retrieval (CLIP)**
4. **Agentic Workflow & Tool-Based Reasoning**
5. **Recommendation Engine and Personalization**

Each component is described in detail below.

A. Data Preprocessing and Embedding Generation

1. Product Data Normalization

All product information passes through a preprocessing pipeline:

Removal of stopwords
Tokenization
Lowercasing
Punctuation removal
Attribute extraction (brand, category, color, material, price range)

This ensures consistency across text fields.

Example:

Product Title → "Nike Black Running Shoes for Men"
Processed Text → "nike black running shoes men"

2. Text Embeddings Generation

We generate **semantic embeddings** using:

Nomic Embeddings (768-d)
InstructorXL (optional, 1024-d)

Each product has:

`title_embedding`
`description_embedding`
`combined_embedding` (weighted)

Weighted Embedding Formula

$E_{combined} = 0.6E_{title} + 0.4E_{description}$

This improves retrieval accuracy since titles are usually more expressive.

3. Image Embeddings Generation (CLIP)

Every product image is converted into a **512-dimensional CLIP vector**:

$E_{image} = CLIP(Image)$

These embeddings support:

image-to-image search
image-to-text search
product similarity clustering

4. Storing Embeddings in PGVector

PGVector stores embeddings in columns defined as:

```
embedding VECTOR(768)
image_embeddingVECTOR(512)
```

Indexes are created using:

```
CREATE INDEX product_embedding_idx
ON products USING ivfflat (embedding vector_cosine_ops)
WITH (lists = 100);
```

This provides **fast top-k vector queries**.

B. Semantic Vector Search

Vector search replaces traditional keyword queries.

1. Query Embedding

When a user types a query such as:
"Show me affordable running shoes under ₹2000"
The LLM agent extracts intent:

category: running shoes
price constraint: < ₹2000

The text query is embedded:

$E_q = Embed(\text{"affordable running shoes"})$

2. Vector Similarity Search

PGVector performs cosine similarity:

$Similarity(u,v) =$

Retrieval:

```
SELECT * FROM products
ORDER BY embedding <=>E_q
LIMIT 10;
```

3. Hybrid Search (Metadata + Vector Search)

The system blends:

cosine similarity

International Journal of Advances in Engineering and Management (IJAEM)
Volume 8, Issue 5 May. 2026, pp: 69-79 www.ijaem.net ISSN: 2395-5252

price filters
category filters

This is called **hybrid retrieval**.

C. Multimodal Search (Image-Based Retrieval)

The system allows users to upload an image and find visually similar products.

1. Image → Embedding

User uploads an image:

$$E_{query\_img} = CLIP(Image_{user})$$

2. Vector Search with PGVector

Nearest neighbors retrieved using:

```
SELECT * FROM products
ORDER BY
image_embedding<=>E_query_img
LIMIT 20;
```

3. Cross-Modal Retrieval

CLIP enables:

image-to-text
text-to-image

Cross-modal similarity:

$$Score =$$

Thus, a user can:

describe a product in words
upload a reference image
and the system matches both

D. Agentic Workflow and Tool-Based Reasoning

Agentic AI is the **heart** of NovaKart.

The system uses a **multi-agent architecture**, where each agent performs a specialized task.

1. Planner Agent (Primary Controller)

Responsibilities:

Understands user intent
Breaks the request into sub-tasks
Chooses relevant agents/tools
Orchestrates multi-step reasoning

Example Input:

"Find stylish jackets under ₹1500 similar to this image."

Planner Output (Sample):

```
TASK 1: Extract features from user
image (Image Agent)
TASK 2: Perform vector search
(Search Agent)
TASK 3: Filter by price (Backend
Tool)
TASK 4: Rank by style similarity
(Recommendation Agent)
```

The Planner uses **ReAct** reasoning:

Think
Act (call tool)
Observe
Think again
Act again

2. Search Agent

Roles:

Executes semantic vector search
Uses PGVector for ranking
Applies filters (price, category, brand)

The search agent is called by the Planner when textual or multimodal queries are needed.

3. Image Agent

Roles:

Converts user-uploaded image → CLIP embedding
Sends the vector to the Search Agent
Extracts color, category, patterns

4. Recommendation Agent

Uses:

collaborative filtering
embedding similarity
popularity scores
personalized user vectors

User Preference Vector

$$E_{user} = mean(E_{product\_purchased} + E_{product\_viewed})$$

Recommendation Score

$$Score = similarity(E_{user}, E_{product})$$

Used to:

rank search results
provide suggestions
generate a personalized homepage

5. Tool Invocation System

The Planner Agent calls tools such as:

`searchProducts()`
`filterByPrice()`
`imageEmbedding()`
`recommendProducts()`
`getProductDetails()`

This makes the LLM operate like an **autonomous agent** capable of interacting with the entire e-commerce backend.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 8, Issue 5 May. 2026, pp: 69-79 www.ijaem.net ISSN: 2395-5252

E. Recommendation Engine (Advanced)

The recommendation engine operates on vector embeddings, allowing deep personalization.

1. Content-Based Filtering

Matches product embeddings with user embeddings.

2. Collaborative Filtering

Similar users → similar product preferences.

3. Cross-Modal Recommendations

Image preference → product suggestions.

4. Real-Time Adaptation

The system updates user vector after every:

- click
- view
- purchase
- wishlist action

User Vector Update Rule

Where:

$$E^{new}_{user} = \alpha E^{old}_{user}(1-\alpha)E_{product}$$

α = 0.7 ensures stability

(1-α) allows recent interactions to influence recommendations

F. End-to-End Flow of a User Query

Example Query:

"Find me black sneakers under ₹3000 similar to this image."

Step-by-step Flow:

Step	Component	Description
1	Frontend	User uploads image + query
2	Planner Agent	Detects intent, creates task list
3	Image Agent	Generates CLIP embedding
4	Search Agent	Performs vector search (text + image)
5	Backend	Applies price + category filters
6	Recommendation Agent	Ranks results based on user preferences
7	Qwen LLM	Formats readable response
8	Frontend	Displays final product list

Table no : 1

This pipeline ensures:

- accuracy
- personalization
- multimodal understanding
- intelligent reasoning

G. Performance Optimization Techniques

To ensure real-time search performance:

1.FFlat Indexing

1. Improves PGVector speed by 10x.

2.ching Layer

2. Redis stores:

frequent queries

popular embeddings

3.rallel Agent Execution

3. Search Agent + Recommendation Agent run in parallel threads.

4. Batching Embedding Requests

Groups multiple embedding processes to reduce compute time.

H. Evaluation Criteria

Metrics Used:

- Vector search accuracy
- Latency of search operations
- LLM agent response time
- Multimodal retrieval precision
- User satisfaction score
- Top-10 precision for recommendations

VI. RESULTS AND EVALUATION

The performance of the proposed **Agentic E-Commerce System (NovaKart)** was evaluated across multiple dimensions including **search accuracy, recommendation quality, agent response time, model efficiency, user satisfaction, and system scalability**. The evaluation was performed on a dataset of **5,000+ e-commerce product entries**, including text descriptions and images, embedded using **CLIP**, **Nomic**, and stored in **PGVector**. The Qwen-3 LLM family was used for reasoning and multi-agent orchestration.

A. Quantitative Evaluation

1) Vector Search Performance (PGVector + CLIP/Nomic)

To evaluate the accuracy of semantic product search, we used **Recall@K**, a common information-retrieval metric.

Metric	Value
Recall@5	**87.3%**
Recall@10	**93.4%**
Median Latency	**112 ms**
Peak Latency	**191 ms**

Table no :2

Interpretation:

The system retrieves highly relevant vector-based

International Journal of Advances in Engineering and Management (IJAEM)
Volume 8, Issue 5 May. 2026, pp: 69-79 www.ijaem.net ISSN: 2395-5252

search results with low latency suitable for real-time shopping applications.

2) Agent Response Time Analysis
We measured the execution time of the three core agents:

Agent	Average Response Time
Planning Agent	**0.82 s**
Search Agent	**0.29 s**
Image Understanding Agent (CLIP/Qwen-VL)	**1.12 s**

Table no :3

Interpretation:
All agents respond within acceptable limits for an AI-powered shopping assistant. The image model is slower due to heavy visual encoding.

3) Recommendation Engine Evaluation
We evaluated the AI recommendation pipeline using **Mean Reciprocal Rank (MRR)** and **User Success Rate (USR)**.

Metric	Score
MRR	**0.81**
USR	**92%**
Top-N Recommendation Accuracy	**88.5%**

Table no :4

Interpretation:
The agentic recommendation system consistently ranks relevant products in the top suggestions.

B. Qualitative Evaluation
1) User Study (Human Evaluation)
A user study was conducted with **30 participants**, who performed common shopping tasks such as:

Finding products via natural language
Uploading an image to find similar items
Asking multi-step queries ("Find me a ₹1500 black backpack under 25L capacity.")
Generating personalized recommendations

User satisfaction was measured on a 5-point Likert scale.

Category	Score (1–5)
Search Understanding	**4.6**
Recommendation Quality	**4.7**
Multi-Agent Task Handling	**4.5**
Image-Based Product Matching	**4.3**
Overall Experience	**4.6**

Table no :5

Observations:
Participants appreciated natural language flexibility and multi-agent reasoning. Image search accuracy received slightly lower scores due to occasional mismatches.

C. Agent-Orchestration Success Rate
To measure the stability of the agent workflow, 200 multi-agent tasks were executed.

Workflow Type	Success Rate
Single-Agent Tasks	**98%**
Multi-Agent Planning → Search → Recommendation	**94%**
Image → CLIP → Search → Response	**91%**

Table no :6

Interpretation:
Most tasks succeed end-to-end. Failures primarily occur due to ambiguous input text or missing metadata.

D. System Scalability Testing
Stress tests were performed with increasing numbers of simultaneous requests (using JMeter).

Concurrent Users	Avg. Latency	Throughput
50	210 ms	420 req/min
100	315 ms	780 req/min
250	612 ms	1,320 req/min
500	940 ms	1,870 req/min

Table no :7

Interpretation:
NovaKart scales effectively up to **500 concurrent users** before latency crosses the 1-second threshold. Caching and load-balancing improvements can further enhance scalability.

E. Comparative Study with Existing Systems
We compared NovaKart against traditional e-commerce search engines (keyword-based) and modern LLM-enhanced systems.

Feature	Traditional Search	LLM Chatbot	NovaKart (Ours)
Semantic Understanding	Low	High	**Very High**
Multi-Agent Processing	No	Limited	**Advanced**
Image-Based Search	No	Partial	**Full CLIP Integration**
Personalization	Rule-Based	Moderate	**AI-Driven**

International Journal of Advances in Engineering and Management (IJAEM)
Volume 8, Issue 5 May. 2026, pp: 69-79 www.ijaem.net ISSN: 2395-5252

Latency	Low	High	**Medium (Optimized)**
Vector Database	No	Rare	**Core Component**

Table no :8

Conclusion:
NovaKart outperforms classical systems in intelligence and reasoning, while maintaining better performance than traditional LLM-only chatbots due to agent modularity.

F. Error Analysis
The most common failure cases included:

- Misclassification of visually similar products (e.g., backpacks vs laptop bags)
- Ambiguous long prompts requiring deeper reasoning
- Missing product images leading to CLIP retrieval issues
- Complex queries requiring external knowledge not available in the database

The overall error rate across all evaluations was measured at **7.2%**, which is within acceptable limits for modern AI-driven systems.

G. Summary of Evaluation
The evaluation demonstrates that:

- ✔ Vector search is highly accurate
- ✔ Agents respond quickly and reliably
- ✔ Multi-agent reasoning improves correctness
- ✔ Users rate the system highly
- ✔ System scales efficiently for medium-sized e-commerce platforms

Overall, NovaKart meets the performance expectations required for a **production-ready agentic e-commerce application**.

VII. DISCUSSION

The proposed agentic-powered e-commerce platform, **NovaKart**, demonstrates that the integration of **large language models, multi-agent reasoning, multimodal embeddings, and vector databases** significantly enhances user interaction and search relevance compared to traditional e-commerce systems.
The results highlight three major insights:

A. Effectiveness of Multi-Agent Reasoning
The use of multiple specialized agents—Planner, Search Agent, Recommendation Agent, and Image Agent—created a dynamic workflow that was more accurate than single-model chatbots. The **Planner Agent** helped break down complex queries, while the **Search and Recommendation Agents** produced contextually rich suggestions based on embeddings.
This multi-step reasoning workflow led to:

- Better alignment with user intent
- More accurate semantic retrieval
- High user satisfaction (4.6/5)

However, multi-agent systems occasionally produced inconsistent responses when subtasks lacked sufficient metadata, which indicates the need for better feedback loops and improved agent coordination.

B. Impact of Multimodal Understanding
The inclusion of **CLIP** and **Qwen-VL** models enabled the platform to interpret user-uploaded product images and retrieve visually similar items. This feature differentiates NovaKart from standard platforms.
Key observations:

- Users found image-based search intuitive.
- Visual-semantic similarity worked well on clear images.
- Performance degraded slightly on low-resolution or cluttered images.

This suggests that future versions should incorporate preprocessing or fine-tuned models for more robust visual matching.

C. Vector Search and Embeddings Performance
PGVector provided fast similarity search, allowing real-time semantic retrieval with **median latency of 112 ms**. The combination of **CLIP image embeddings** and **Nomic text embeddings** resulted in high Recall@10 scores above 93%.
Despite its overall efficiency, vector search performance varied by product category—items with visually diverse designs (e.g., fashion wear) sometimes had lower similarity scores. This behavior can be mitigated by hybrid search (keyword + vector + metadata).

D. System Scalability and Practical Deployment Insights
Stress testing showed that NovaKart scales effectively up to **500 concurrent users**, making it viable for medium-sized e-commerce companies. The microservice architecture allowed the application to remain stable under peak load.
However, LLM inference remained the main computational bottleneck, especially for image-based reasoning tasks. Deployment strategies such as model quantization, batching, caching, and distributed inference may significantly improve performance

International Journal of Advances in Engineering and Management (IJAEM)
Volume 8, Issue 5 May. 2026, pp: 69-79 www.ijaem.net ISSN: 2395-5252

VIII. CONCLUSION

This research presents *NovaKart*, an agentic-powered e-commerce system that integrates **LLMs, vector databases, multimodal embeddings, and autonomous agents** to create an intelligent shopping experience. The system bridges the gap between traditional keyword-based e-commerce search and modern AI-driven conversational commerce.

The implemented multi-agent architecture demonstrates:

- High-quality semantic search and retrieval
- Strong recommendation accuracy (MRR = 0.81)
- Effective image-product matching
- Natural conversational flow through LLM reasoning
- Scalability for real-world use

Overall, NovaKart significantly enhances user engagement, search relevance, and platform intelligence. The system proves that **agentic workflows combined with multimodal AI models can redefine online shopping** by delivering personalized, context-aware, and autonomous assistance.

IX. FUTURE WORK

While NovaKart achieves strong performance, several opportunities exist to enhance the system further.

A. Fine-Tuning Domain-Specific Models

The current deployment uses general-purpose CLIP and Qwen models.

Future work may involve:

- Fine-tuning embeddings on product-specific datasets
- Training category-specific encoders (fashion, electronics, groceries)
- Adaptive learning from user interactions in real time

This can significantly improve similarity retrieval and personalized recommendations.

B. Reinforcement Learning for Agent Optimization

Currently, the Planner and Search Agents rely on heuristic reasoning.

Future iterations can leverage:

- **Reinforcement Learning from Human Feedback (RLHF)**
- **Self-evaluating agents** that refine reasoning steps
- **Long-term user preference modeling**

This will enable agents to improve autonomously without manual rule updates.

C. Integration of Graph Databases

E-commerce products often have relationships (brand → category → material → style). Using a **graph database (e.g., Neo4j)** can help:

- Improve complex query reasoning
- Enable "explainable recommendations"
- Support knowledge graph–assisted search

This complements vector search with structured semantic knowledge.

D. Real-Time Inventory & Dynamic Pricing Intelligence

Incorporating business intelligence features such as:

- Predictive inventory forecasting
- AI-driven price optimization
- Demand trend analysis
- Personalized discounting

would move NovaKart from an e-commerce *platform* to a complete AI-powered *retail intelligence ecosystem*.

E. On-Device or Edge AI Models

To reduce server load and improve latency, future versions may run lightweight models directly on the client (browser/mobile):

- Fast embedding generation
- Quick preference classification
- Offline recommendations

This can significantly reduce API costs and improve scalability.

F. Enhancing Visual Understanding

To strengthen image search:

- Add a preprocessing pipeline for noise/lighting correction
- Use segmentation-based models for product isolation
- Replace CLIP with more advanced models (e.g., SigLIP, EVA-CLIP, Bi-VL models)

These upgrades would provide more consistent visual similarity.

G. Multi-Language Support and Globalization

Adding multilingual capabilities using Qwen's multilingual LLM would enable users to shop using Hindi, Tamil, Bengali, and other languages, enhancing accessibility for Indian e-commerce users.

H. Real-World Deployment and A/B Testing

A large-scale production deployment would enable:

- A/B testing different agent strategies

Measuring conversion rate impact
Optimizing recommendation pipelines
Studying long-term personalization effects

This is necessary for real-world commercial adoption.

REFERENCES

[1] Salton, G., et al., "Vector Space Model for Document Retrieval," 1975.
[2] Manning, C., "Introduction to Information Retrieval," Cambridge, 2008.
[3] Koren, Y., "Matrix Factorization Techniques for Recommender Systems," 2009.
[4] Ricci, F., "Recommender Systems Handbook," 2015.
[5] PGVector Documentation, "PostgreSQL Vector Extension," 2023.
[6] Johnson, J., "Billion-Scale Vector Search," FAISS, 2019.
[7] Radford, A., "Learning Transferable Visual Models Using Natural Language Supervision," CLIP, 2021.
[8] OpenAI Research Report on Multimodal Models, 2022.
[9] Yao, S., "ReAct: Synergizing Reasoning and Acting in LLMs," 2023.
[10] Schick, T., "Toolformer: Language Models Can Use Tools," 2023.
[11] AutoGPT Research Report, 2023.
[12] Google, "LLM-Powered Conversational Agents for Commerce," 2023.
[13] Meta AI, "LLaMA for Commerce Applications," 2024.
[14] Pinecone Research Papers, 2023.
[15] Nomic Embeddings Whitepaper, 2023.
[16] Qwen Technical Report, Alibaba, 2024.
[17] PGVector Benchmarks, 2024.
[18] Springer, "Multimodal Information Retrieval Techniques," 2021.
[19] IEEE Transactions on AI Agents, 2024.
[20] Amazon AI Labs, "Neural E-Commerce Search," 2022.