1. Problem Statement

Traditional keyword-based search fails to understand **user intent**, **context**, and **behavioral signals**. Modern search systems must combine **semantic understanding**, **user behavior analytics**, and **continuous learning** to deliver relevant results.

Your task is to design and implement an **AI-powered contextual search platform** for a product catalog that:

1. Understands **natural language queries**
2. Retrieves products based on **semantic relevance**
3. Continuously **learns from user behavior**
4. Improves ranking quality over time

This project is expected to demonstrate **backend engineering**, **data pipeline design**, **AI integration**, and **system architecture thinking**.

---

2. System Overview

The system must support:

- Product ingestion and indexing
- Semantic (context-aware) search
- Behavioral event tracking
- Learning-based ranking improvements
- Explainable AI-assisted search results

The solution must be **production-oriented**, not a prototype or a UI-only demo.

---

3. Functional Requirements

3.1 Product Ingestion Pipeline

You must build a **reusable ingestion pipeline** that:

- Accepts product data in JSON or CSV format

- Normalizes product fields such as:
    - Title
    - Description
    - Category
    - Attributes (brand, size, color, etc.)
- Generates **vector embeddings** for searchable text fields
- Stores product data in:
    - A **structured database**
    - A **vector database**

Hardcoded ingestion or one-time scripts will not be accepted.

3.2 Contextual Search Engine

The search system must:

- Accept **natural language queries**, for example:
    - *“Running shoes for flat feet under ₹5000”*
    - *“Lightweight laptop for coding and gaming”*
- Convert queries into embeddings
- Perform **semantic similarity search**
- Apply structured filters such as:
    - Price range
    - Category
    - Rating
- Rank and return relevant products

3.3 User Behavior Tracking & Analytics

The system must track **user interactions** including:

- Search queries
- Product clicks
- Add-to-cart events
- Purchases
- Time spent (dwell time)

All events must:

- Be captured asynchronously
- Flow through a **message queue or event pipeline**
- Be stored for analytical processing

The system must **not** rely on synchronous logging.

3.4 Learning from User Behavior (Mandatory)

The system must demonstrate **learning from user behavior** by improving search ranking using real usage data.

Examples:

- Boost products that receive frequent clicks for a query
- Penalize products with high bounce rate
- Adjust ranking based on conversion signals

A simple heuristic-based approach is acceptable, but it must be **clearly explained and implemented**.

3.5 AI Integration (Mandatory)

At least **one** of the following AI features must be implemented:

- Query expansion using an LLM
  (e.g., expanding “mobile” → smartphone, android phone, 5G phone)

- AI-based re-ranking of top-K search results

- Automatic attribute extraction from product descriptions

- AI-generated explanation:
  “Why was this product shown for this query?”

Pure keyword search without AI usage will result in **disqualification**.

---

4. Non-Functional Requirements

4.1 Architecture

- Clear separation of concerns
- Modular, maintainable code
- Proper layering (API, service, data, AI)

4.2 Scalability

- System should handle increasing product count
- Event ingestion must be asynchronous

-

4.3 Observability

- Logging
- Basic metrics (query latency, event counts)

---

5. Evaluation Criteria

Category	Weight
Search relevance & quality	25%
Data pipeline design	20%
Learning from behavior	20%
Code quality & modularity	20%
AI integration quality	15%

---

6. Important Guidelines

- UI quality will **not** be a primary evaluation factor
- Copying end-to-end tutorials will lead to penalties
- AI must assist reasoning, not replace system logic
  Partial but well-explained implementations are preferred over shallow completeness

---

7. Deliverables

Each team must submit:

1. Source code repository
2. Architecture diagram
3. README explaining:
    - System design
    - Data flow
    - AI usage
    - Learning logic

4. Sample dataset
5. Demo video (optional but recommended)

8. Bonus (Optional)

- Personalized search per user
- Query analytics dashboard
- Offline batch re-ranking
- Multi-language search