Workbook

Designing and Justifying an Industry-Grade AI Commerce Assistant

(RAG + Tools + LCEL + VectorDB + OpenAI)

Purpose of This Workbook

This workbook is designed to guide teams through structured decision-making before building a production-grade AI Commerce Assistant.

By the end, teams should be able to:

- Define a clear problem statement
- Select appropriate LLMs
- Choose embedding models
- Select a vector database

- Design a RAG pipeline
- Justify tool choices (Weather, Serper, etc.)
- Decide on reranking strategy
- Define evaluation metrics
- Explain architectural trade-offs

This is not a coding exercise.

It is a system design and justification exercise.

Section 1 — Problem Definition

1.1 Define the Business Objective

Describe the core problem your AI system is solving.

- What customer pain point are you addressing?
- Is this replacing search, recommendation, support, or all three?

- What measurable business impact is expected?

Write your objective clearly:

Business Objective:

1.2 Define the User Persona

Who will use the system?

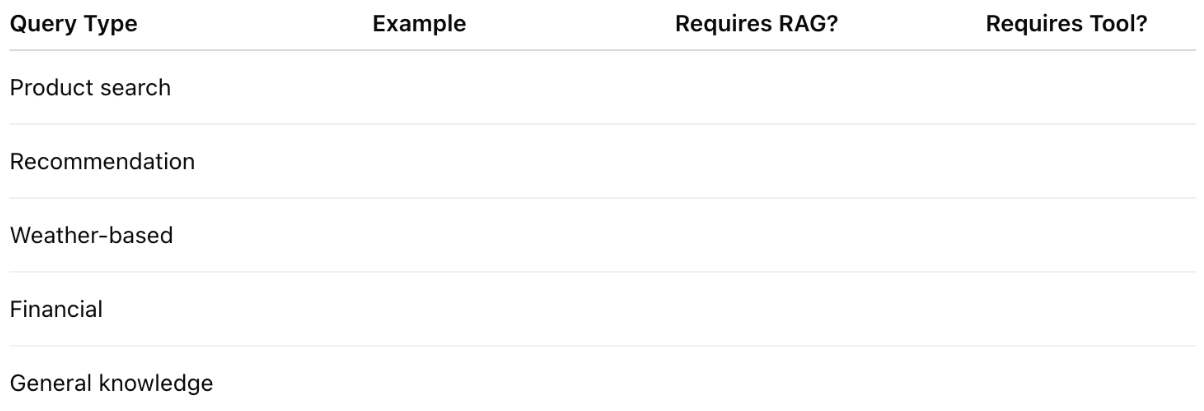
- New customer?
- Returning user?
- High-value buyer?
- Enterprise buyer?

Primary User Persona:

Key Expectations:

1.3 Define Query Categories

List the expected types of user queries:



Query Type	Example	Requires RAG?	Requires Tool?
Product search			
Recommendation			
Weather-based			
Financial			
General knowledge			

Section 2 — LLM Selection Decision

2.1 Model Requirements

What do you need from your LLM?

- Deterministic behavior?
- Tool calling?

- Large context window?
- Low latency?
- Cost optimization?

List requirements:

2.2 Model Choice

You must choose:

- GPT-4o-mini
- GPT-4o
- Other OpenAI model

Justify your selection:

Chosen Model:

Justification:

- Reasoning capability:
- Cost:
- Latency:
- Tool calling support:
- Context length:

Trade-offs:

Section 3 — Embedding Model Selection

3.1 Requirements

- Dimensionality?
- Cost constraints?
- Performance requirements?

- Multilingual support?

3.2 Embedding Model Decision

Options:

- text-embedding-3-small
- text-embedding-3-large

Chosen Model:

Justification:

- Cost per 1M tokens:
- Retrieval accuracy:
- Expected catalog size:
- Latency:

Section 4 — Vector Database Decision

4.1 Scale Estimation

Estimated product catalog size:

Expected growth rate:

Concurrent users:

4.2 Database Options

DB	Pros	Cons	Use Case Fit
Chroma			
Pinecone			
Weaviate			
Qdrant			

4.3 Decision

Selected Vector DB:

Justification:

- Scalability:
- Persistence:
- Metadata filtering:
- Cost:
- Operational complexity:

Section 5 — Chunking Strategy

5.1 Content Type

What are you chunking?

- Product descriptions?

- Reviews?
- FAQs?
- Specifications?

5.2 Chunk Size Decision

Chosen chunk size:

Overlap:

Why?

Trade-offs:

- Larger chunks:
- Smaller chunks:

Section 6 — Retrieval Strategy

6.1 Similarity Search

- Top K value:
- Similarity metric (cosine, dot product):
- Threshold filtering:

Justify:

6.2 Metadata Filtering

Will you filter by:

- Price?
- Category?
- Availability?

- Rating?

Explain logic:

Section 7 — Reranking Strategy

7.1 Do You Need Reranking?

When would reranking be necessary?

- High catalog volume?
- Low precision in top-K?
- Need higher recommendation accuracy?

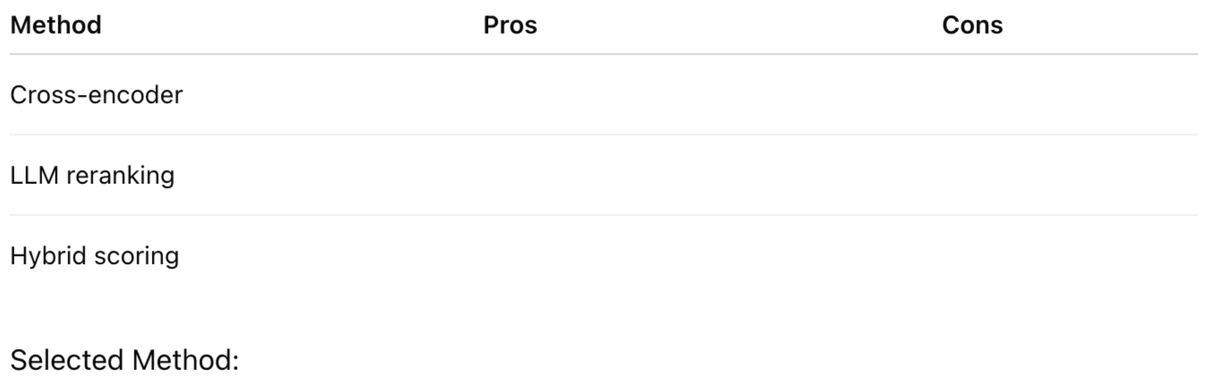
Decision:

Yes / No

Justify:

7.2 Reranking Options

Selected Method:



Method	Pros	Cons
Cross-encoder		
LLM reranking		
Hybrid scoring		

Selected Method:

Justification:

- Latency impact:
- Accuracy gain:
- Cost impact:

Section 8 — Tool Selection Strategy

8.1 Required Tools

List all tools required:

- Weather
- Web Search (Serper)
- Inventory
- Calculator
- Interest
- Shipping estimator

Why is each tool necessary?

Tool:

____

Justification:

____

____

8.2 Tool Routing Strategy

Will routing be:

- Intent classifier-based?
- Fully LLM autonomous?
- Rule-based first, LLM second?

Selected approach:

____

Justification:

____

____

Section 9 — Fallback Strategy

If:

- No product found
- Low similarity score
- Out-of-stock product

What happens?

Fallback approach:

Why?

Section 10 — LCEL vs Direct API

Will you:

- Use LCEL pipelines?
- Use LangGraph?
- Use direct OpenAI APIs?

Decision:

Justification:

- Debugging clarity:
- Control:

- Maintainability:
- Team expertise:

Section 11 — Personalization

Will you:

- Use user embeddings?
- Use past purchases?
- Use session memory?

Explain personalization strategy:

Section 12 — Evaluation Strategy

12.1 Offline Evaluation

Metrics:

- Retrieval precision
- Recall
- MRR
- NDCG

Selected metrics:

12.2 Online Evaluation

- CTR improvement
- Conversion rate
- Bounce rate
- Tool usage frequency

Define KPIs:

Section 13 — Cost Analysis

Estimate:

- LLM cost per query
- Embedding cost
- Vector DB cost
- Tool API cost

Projected monthly cost:

Cost optimization strategies:

Section 14 — Risk Assessment

Identify risks:

- Hallucination
- Tool misuse
- Latency spikes
- API downtime
- Injection attacks

Mitigation strategy:

Section 15 — Final Architecture Justification

Summarize your final decisions:

LLM:

Embedding Model:

Vector DB:

Chunking Strategy:

Retrieval Strategy:

Reranking:

Tools:

Fallback:

Why this architecture is production-ready:

Final Reflection

Explain:

1. Why your LLM choice is appropriate.
2. Why your vector database scales.
3. Why reranking is necessary or not.
4. Why your tool routing strategy is safe.
5. How you would scale to 10x traffic.

Write a structured justification:

Outcome

After completing this workbook, a You should be able to:

- Defend every architectural choice
- Explain trade-offs clearly

- Estimate cost and latency
- Design a scalable RAG + Tool system
- Justify reranking decisions
- Move confidently into implementation