Test questions:

Percentages of the preference for type of engine in next vehicle?

Which type is preferred the most?

What was the sample size of this survey?

What was my first question?

Most important factors driving the choice of brand for the next vehicle?

Which of these is preferred by most of the Japanese?

What was my second question?

Link to the PDF document I used:

https://www.deloitte.com/in/en/Industries/automotive/perspectives/global-automotive-consumer-study.html

Processing Pipeline
o **Chunking:** sensible PDF text extraction; state your chunking strategy.
o **Embeddings:** any model (OpenAI, HF, BGE, etc.). Note model name and dimensions in README.
o **Vector Store:** any vector database (Chroma, FAISS, LanceDB, Qdrant, etc) of your choice.
o **Retrieval:** mention the retrieval strategy and the reason.

Based on our implementation, here are the answers:

Chunking Strategy

Approach: Hybrid chunking with structured data preservation

Implementation Details:

- **Chunk Size:** 800-1000 tokens with 150-200 token overlap
- **Smart Detection:** Identifies structured data (tables, charts, percentages, lists)
- **Dual Mode:**
    - **Regular text:** Sliding window chunking with overlap

    - **Structured data:** Kept intact in dedicated chunks
- **Page Preservation:** Maintains page metadata for accurate citations

Key Features:

- Preserves tabular data and charts as single units
- Uses regex patterns to detect structured content (\d+%, tables, etc.)
- Maintains context through overlapping chunks for continuous text
- Page-level tracking for precise source referencing

Embeddings Model

Primary Model: Google Gemini Embedding-001

- **Dimensions:** 768 dimensions
- **API:** Google Generative AI API
- **Task Type:** retrieval_document for optimal document retrieval

Fallback Model: SentenceTransformer all-MiniLM-L6-v2

- **Dimensions:** 384 dimensions
- **Usage:** Local fallback when Gemini API is unavailable
- **Purpose:** Ensures application works without API dependencies

Embedding Strategy:

- Automatic fallback to local model if Gemini fails
- Dimension compatibility checks between index and embeddings
- Support for both single text and batch processing

Vector Store

Choice: FAISS (Facebook AI Similarity Search)

- **Type:** FAISS CPU version 1.7.4
- **Index Type:** IndexFlatL2 (Euclidean distance)
- **Persistence:** Local disk storage (storage/faiss_index)

Reasons for Choosing FAISS:

1. **Local Operation:** No external dependencies or servers required
2. **Windows Compatibility:** Works seamlessly on Windows platforms
3. **Lightweight:** Minimal resource requirements
4. **Fast Retrieval:** Optimized for similarity search
5. **Easy Persistence:** Simple file-based storage

6. **Python Integration:** Excellent library support

Storage Structure:

- FAISS index: storage/faiss_index
- Metadata JSON: storage/metadata.json (chunk text + page info)

Retrieval Strategy

Approach: Dense Passage Retrieval (DPR) with Query Awareness

Retrieval Process:

1. **Query Embedding:** Convert user query to same embedding space
2. **Similarity Search:** FAISS L2 distance search (Euclidean)
3. **Top-k Retrieval:** Return top 5 most similar chunks
4. **Re-ranking:** Sort by similarity score (1/(1 + distance))

Smart Retrieval Features:

- **Context-Aware:** Different strategies for different query types:
    - **Data queries** (percentages, charts): Prioritize structured chunks
    - **Conversation queries:** Skip document search, use chat history
    - **General queries:** Standard semantic search

Retrieval Parameters:

- k=5 chunks retrieved per query
- Similarity thresholding based on distance scores
- Page-based source attribution for citations

Why This Strategy:

1. **Accuracy:** Dense embeddings capture semantic meaning better than keyword search
2. **Speed:** FAISS provides millisecond-level retrieval
3. **Context:** Overlapping chunks maintain contextual information
4. **Precision:** Structured data preservation ensures complete information retrieval
5. **Flexibility:** Handles diverse query types appropriately