# Strategy

| **Optimization Area** | **Best Practice** |
| --- | --- |
| **RAG Data Quality** | **Preprocess retrieved documents: extract key metadata (product names, features, pricing tiers, compatibility). Use chunking with semantic grouping (per product or per customer use case).** |
| **Retrieval Strategy** | **Use hybrid search (vector + keyword). Prioritize recency and relevance scoring (e.g., via cosine similarity + metadata filters for customer industry or product category).** |
| **Context Structuring** | **Feed the model *structured JSON-like snippets* for customer environment and product data — it’s much easier for the model to reason over structured inputs than raw text.** |
| **Prompt Engineering** | **Include few-shot examples of “good recommendations” vs. “bad recommendations” to guide the model’s reasoning style.** |
| **Response Control** | **Request structured JSON output if you need to feed the results into downstream applications or dashboards.** |
| **Evaluation Loop** | **Implement human-in-the-loop review of top recommendations to create fine-tuned examples for reinforcement (can later be used to train custom embeddings or fine-tune a domain model).** |
| **Vendor Product Updates** | **Automatically refresh vendor product embeddings monthly to reflect catalog changes and new releases.** |
| **Customer Segmentation** | **Add metadata (industry, size, tech maturity) to retrieved context to make cross-sell suggestions more relevant.** |
| **Temperature Control** | **Keep temperature low (0.2–0.4) for factual, data-grounded output; slightly higher (0.6–0.7) for creative sales positioning text.** |

Questions:

1. Should I use multiple Retrievals? (Yes, implement this as Inference V2)

Example:

Retrieval 1 is used for asking the LLM what Products it would like to recommend (ill refer to as **recs**) based on the User’s Prompt and Context retrieved from the Vector Store (expected to only contain Customer’s Existing Environment)

So the Prompt to the 1st LLM will look like:

Based on the Company’s Information : {context\_1}

Can you generate me a List of Products which you want to recommend based on the given question?

Question: {question}

The Response will look like:

* Cloud Storage
* App Engine
* Vertex AI

This list is stored as **recs**

Retrieval 2 is performed based on the Output of the LLM from Retrieval 1, where it will search for Product information from **recs**. The Product information is then sent to another LLM as Context, and its job is to explain why does it recommend **recs**.

So the Prompt to the 2nd LLM will look like:

Based on the Company’s Information : {context\_1}

You previously mentioned you would recommend: {**recs**}

Can you explain why would you recommend those based on the Product Information given: {context\_2}

Then the Response would be what’s given to the User

# Plans

## Inference V2

1. Implement Multi-Prompt approach

Example:

In the first Inference, we:

* Retrieve the Context based on the User Prompt
* Have the LLM generate:
  + A List of Products used by the Customer
  + A List of Products to Recommend to the Custmer

The output is then used by the 2nd Inference, where we’ll:

* Retrieve the Context based on the List of Products owned by the Customer and List of Proucts to Recommend.
* Have the LLM generate:
  + The Reasoning for Recommending those Products

## Inference V3

1. Implement Prompt Bridges

Example:

There is a chance that a User doesn’t want to use the Chatbot to perform Cross Sell, but we only have a Cross Sell RAG Pipeline

Our Application may support other Pipelines (Example: “QnA about Customer” RAG Pipelines)

This feature will help the application determine which RAG Pipeline to use for processing the Response.

# TODO

1. Implement Inference V2
2. Rename each Uploaded File by adding their ID as Suffix
3. Create a Generate Random Number Function with a Parameter for a Fixed Length