## 1. Model Choice

The project uses Gemini-2.0 Pro Experimental (02-05) as the primary large language model (LLM), paired with the text-embedding-004 model for retrieval. Gemini-2.0 Pro Exp was selected for its state-of-the-art performance on complex queries and factual tasks. It is described as having the strongest capability for handling complex prompts with superior understanding and reasoning over world knowledge, surpassing any prior model release, and it offers an unprecedented 2 million-token context window​ [blog.google](https://blog.google/technology/google-deepmind/gemini-model-updates-february-2025/#:~:text=to%20that%20feedback,to%20comprehensively%20analyze%20and%20understand)

## 2. System Prompts

We implemented a custom system prompt to guide the chatbot’s behavior. The prompt explicitly instructs the model to use only the retrieved content to answer the user’s question (avoid any outside knowledge or fabrication) It also encourages the chatbot to deduce the answer with retrieved information.

"””You are a helpful assistant that retrieves relevant information from provided documents to accurately answer user questions.

Your response must be based only on the given context and document content.

Instructions:

1. Use only the provided documents as your source of information.

2. Do not fabricate information or speculate beyond what is present in the documents.

3. If a direct answer is not found, provide the closest relevant information or logically deduce an answer using the document content.

4. Clearly indicate when an answer is based on deduction rather than explicit information.

User Question: {user\_input}”””

## 3. RAG

## 1) Chunking

For the method, we used the TokenTextSplitter recommended for Gemini API as it has strict token limits. Also, more uniform chunk sizes can improve retrieval embedding consistency. Main drawback is it fails to maintain semantic coherence like RecursiveCharacterTextSplitter does, which brings higher risk of hallucination. Our solution to this problem is increasing “k” in search\_kwargs: the number of most relevant chunks returned as final results. This helps to ensure enough information is fetched even when there is context break.

## 2) Embedding

For the retrieval component, we utilize text-embedding-004 as the embedding model. This model converts text into high-dimensional vector representations that enable semantic search for relevant documents. We chose text-embedding-004 because of its strong performance on retrieval benchmarks – it outperforms previous embedding models of similar size on standard tests like MTEB, achieving state-of-the-art semantic similarity results​ [ai.google.dev](https://ai.google.dev/gemini-api/docs/models/gemini#:~:text=%60text,the%20standard%20MTEB%20embedding%20benchmarks)

## 4. Insights from QA Testing

We tested the chatbot on Q&A pairs derived from Alphabet’s 2024 10-K, examining its ability to handle real-world financial information.

**Strengths:**

Demonstrated strong factual accuracy by providing exact figures. (e.g., R&D spending)

Clearly indicating when data was missing, minimizing hallucinations.

Good at basic reasoning, performing good calculations like free cash flows.

Answers were generally concise, well-structured and easy to understand..

**Weaknesses:**

Very occasional verbosity or formatting error.

Inability to provide conclusive answers for unaddressed topics.

## 5. Model Evaluation & Potential Improvements

We tested the Gemini-2.0 Pro Exp-based chatbot on a QA dataset from Alphabet’s 10-K. The system had high factual accuracy: it retrieved and conveyed relevant information correctly, rarely misinterpreting the source with an accuracy of about 94.7%. The main drawback is computational complexity, but query responses remained sufficiently quick, aided by fast embedding lookups. The large context window also reduces retrieval calls, contributing to efficient knowledge extraction.

From a user perspective, answers were concise, professional, and generally easy to understand. In the future, we could cite exact document sections, refine prompts for brevity, and standardize how unanswerable queries are handled (e.g. using a consistent format or explicit apology). We may also introduce re-ranking or smaller, faster models for simpler queries, especially under heavier usage. Additional caching could further enhance speed and lower costs.

Continuous evaluation through accuracy metrics, such as precision/recall and user feedback, will identify potential gaps. This iterative approach will guide fine-tuning and refine the system’s reliability. By integrating user feedback loops, we aim to optimize prompts, retrieval strategies, and final answers to consistently meet evolving demands. Currently, the chatbot meets our accuracy targets, so future efforts will focus on boosting efficiency, maintaining consistent performance, and ensuring a polished user experience.

Accuracy Report: [Alphabet 10-K Evaluation Report (2024) G7 - Google Docs](https://docs.google.com/document/d/15AczlyUs0EmsCH5ncpNu6CNun998f7LDiddALk7oLFg/edit?tab=t.xay9481hmv4j)

## 6 Reference

[1][blog.google](https://blog.google/technology/google-deepmind/gemini-model-updates-february-2025/#:~:text=to%20that%20feedback,to%20comprehensively%20analyze%20and%20understand)

[2][community.openai.com](https://community.openai.com/t/prompt-engineering-for-rag/621495#:~:text=DOCUMENT%3A%20)

[3][ai.google.dev](https://ai.google.dev/gemini-api/docs/models/gemini#:~:text=%60text,the%20standard%20MTEB%20embedding%20benchmarks)