☁️ AWS CloudGuide – Your Cloud Architecture Assistant

DATA 6312: Natural Language Processing for Data

Team members:
Mohammed Ismail Sarfaraz Shaik
Harshith Maddala
Jasreen Kaur Mehta

PROBLEM STATEMENT

AWS technical documentation is enormous, highly fragmented, and constantly evolving. Engineers often spend too much time searching through thousands of pages to find specific configuration details or architectural guidance.

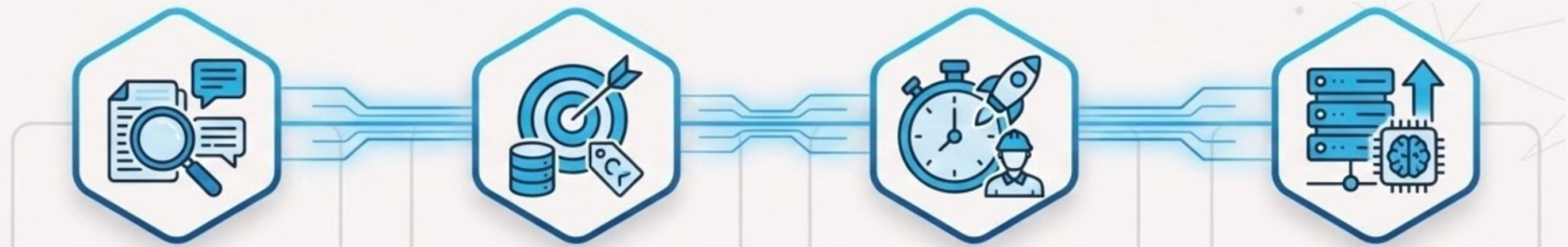
The system is designed for cloud engineers, solution architects, DevOps teams, and students learning AWS. These users need fast, dependable access to official AWS guidance without manually digging through multiple manuals.

A retrieval-augmented conversational AI assistant that provides accurate, citation-backed responses directly from AWS documentation. Instead of manually searching, users can ask questions in plain English and instantly retrieve reliable, official AWS information.

WHY NOT?

- Most cloud engineers or Users rely on tools like Google Search, Stack Overflow, or general-purpose chatbots.
- But these sources often return outdated, inconsistent, or hallucinated information.
- The built-in AWS search engine struggles with long-form queries and often surfaces irrelevant pages.
- General LLMs like ChatGPT are not trained specifically on AWS manuals, so their responses may miss critical configuration details or violate best practices.
- Existing solutions also lack citation transparency, making it hard to verify correctness.

OBJECTIVES



Improve Accessibility

Make AWS technical documentation easier to search, understand, and navigate through a conversational AI interface.

Enhance Accuracy

Provide precise, citation-backed answers using hybrid retrieval and grounded LLM responses.

Boost Productivity

Reduce time cloud engineers spend manually searching PDFs, forums, or documentation pages.

Infrastructure

Develop a robust RAG backend that can scale to larger datasets, additional cloud platforms, and future model upgrades.

TABLE OF CONTENTS

Dataset

The AWS PDF Dataset contains text extracted from all publicly available AWS technical PDFs, including user guides, architecture best practices, and service documentation.

- Dataset of more than 100k pages
- Covers a wide range of AWS services, architectures, and operational concepts

Preprocessing & Standardization

Dataset Download & Ingestion

- The data ingestion pipeline begins by downloading the **semihk1/aws-public-pdf-chunked-dataset**, which contains high-quality text chunks extracted from the official 2025 AWS User Guide PDFs.

Quality Filtering

- Removes low-value content such as Table of Contents pages, copyright notices, and extremely short or noisy chunks.

Metadata Structuring

Each chunk is standardized into a JSON object containing:

- text
- source
- chunk_id
- length

Precision Chunking

- Large documents were split using a recursive text splitter into **1000-character chunks with 200 overlap** to maintain contextual continuity.

Normalization & Cleaning

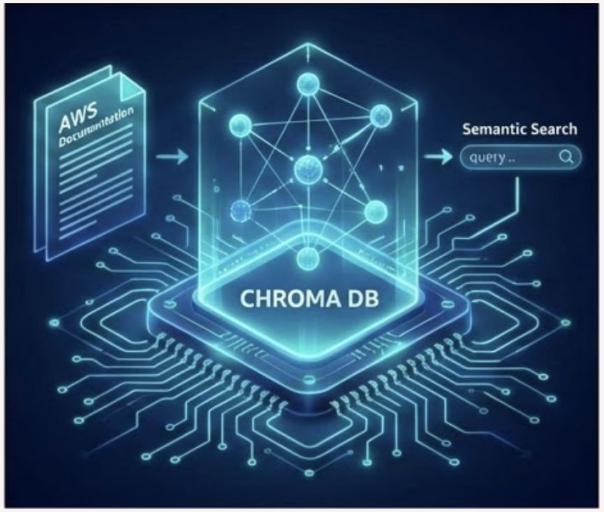
- Ensures uniform formatting, proper whitespace, and consistent text fields for both semantic and keyword retrieval engines.

Indexing: Vector & Keyword Search Engines

To enable high-accuracy retrieval from AWS documentation, the system builds two complementary search indexes: a semantic vector index and a sparse keyword index.

Vector Index (Semantic Search)

- **Model:** Built using **BAAI/bge-m3** embeddings, accelerated on GPU.
- **Storage:** Persisted in **ChromaDB** for low-latency retrieval.
- **Function:** Encodes text into **1,024-dimensional vectors** to capture conceptual meaning.
- **Advantage:** Enables natural language queries (e.g., "How do I save money?" matches "Cost Optimization") that keyword search would miss.



BM25 Index (Keyword Search)

- Created using **BM250kapi** from the **rank_bm25** library.
- Tokenizes each text chunk and ranks documents by keyword relevance.
- Excels at precise term matching (e.g., “S3 bucket policy,” “IAM trust relationship”).
- Stored as a serialized object along with the raw documents.

Both indexes allow the system to retrieve the most relevant sections of AWS documentation using complementary strategies — semantic understanding plus exact keyword matching.

Hybrid Retrieval Engine

Resource Loading & Initialization

- Loads the semantic embedding model **BAAI/bge-m3** (GPU-accelerated if available).
- Loads the **Chroma vector database** and restores all the 800k+ stored embeddings.
- Loads the **BM25 keyword index** and its associated document list.
- Loads **Qwen2.5-7B-Instruct** in **Native FP16 (Half-Precision)** along with tokenizer and assembles a HuggingFace text-generation pipeline.

Hybrid Retrieval Engine (Vector + BM25)

- Performs semantic similarity search using **ChromaDB**.
- Performs keyword relevance search using **BM25**.
- Tokenizes queries for **BM25** using simple lowercase splitting.
- Retrieves top-k documents from both engines simultaneously for fusion.

Reciprocal Rank Fusion (RRF)

- Combines **vector** and **BM25** results into a single ranked list.
- Assigns fusion scores based on document **rank** positions.
- Produces a unified ordering of chunks to maximize relevance.
- Eliminates duplicates by merging identical content found by both search engines.

LLM Answer Generation Pipeline

- **Constructs Prompt:** Builds a structured system + user prompt using a chat template.
- **Injects Context:** Embeds retrieved AWS documentation directly into the prompt.
- **Enforces Grounding:** Instructs model to answer strictly from provided text to prevent hallucinations.
- **Generates Answer:** Runs inference via Qwen2.5-7B pipeline.
- **Formats Output:** Cleans and structures the final response, returning it with source citations.

Streamlit Interface

Interactive Chat Interface

Users can ask questions in natural language and view responses formatted as conversational messages.

Evaluation & Metrics

COSINE SIMILARITY

Cosine similarity measures how similar two text embeddings are by calculating the cosine of the angle between them. A score closer to **1** means the embeddings — and therefore the meanings — are highly aligned.

Q1. "What is the maximum execution time for an AWS Lambda function?"

Q2. "What is the difference between a Scan and a Query in DynamoDB?"

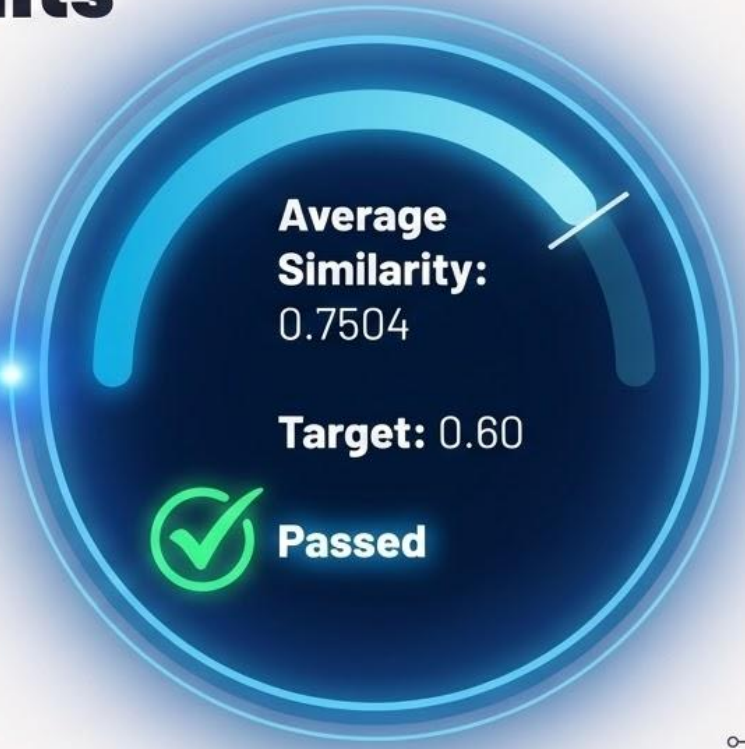
Q3. "How do I create an S3 bucket?"

Q4. "What is the difference between On-Demand and Spot Instances?"

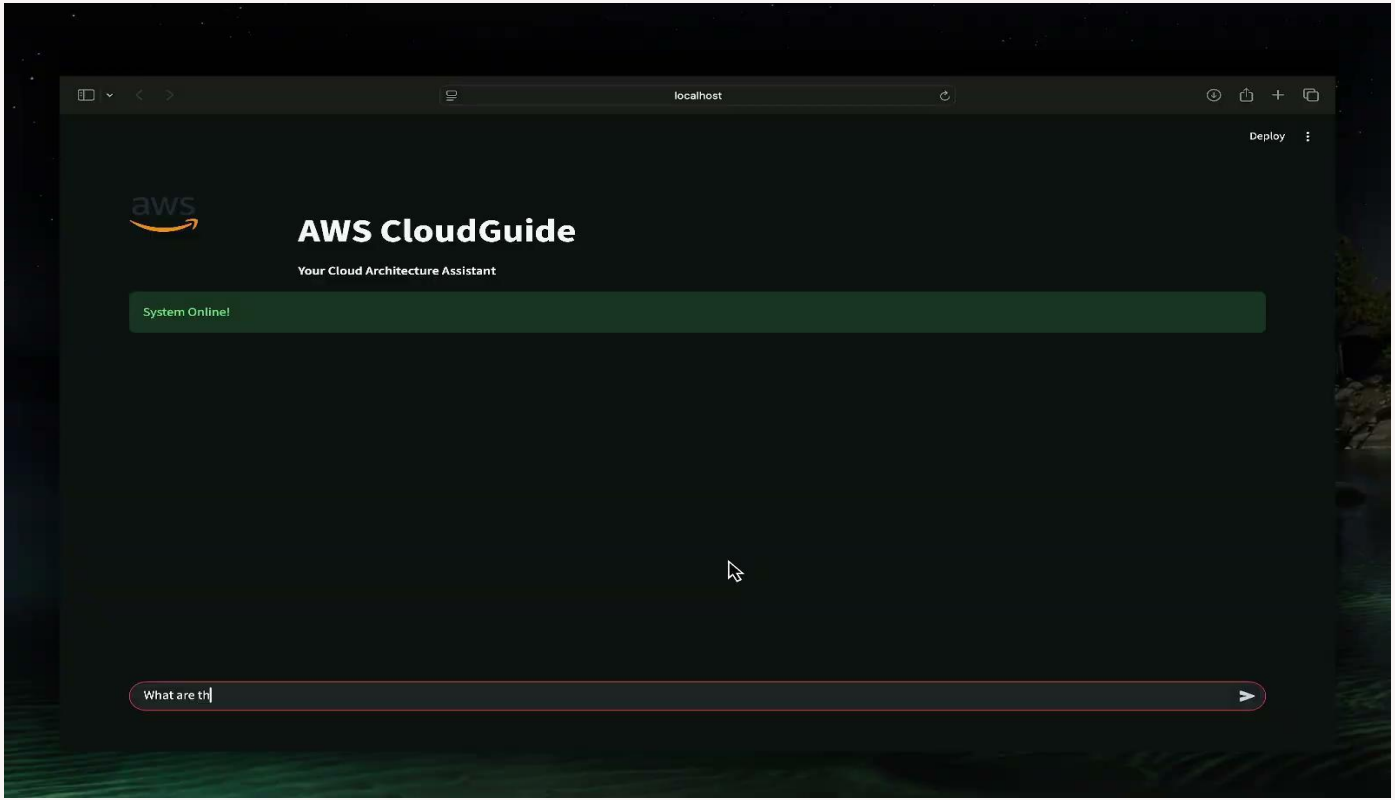
Q5."What is the AWS root user?"

Results

- **Q1:** 0.7707
- **Q2:** 0.6745
- **Q3:** 0.7777
- **Q4:** 0.7575
- **Q5:** 0.7715



Demo


localhost
Deploy
aws
AWS CloudGuide
Your Cloud Architecture Assistant
System Online!
What are th

Pros and Cons

Pros

- **High Retrieval Accuracy:** Hybrid RAG (Vectors + BM25 + RRF) provides reliable, context-rich results.
- **Grounded, Citation-Backed Answers:** Responses are tied directly to AWS documentation, reducing hallucinations.
- **Scalable Architecture:** Works with large technical corpora and can expand to other cloud platforms.
- **User-Friendly Interface:** Streamlit app offers a clean, intuitive conversational experience.

Cons

- **GPU Dependency:** Embedding generation and model inference require a capable GPU for reasonable performance.
- **Ingestion Fragility:** Single-GPU processing of 100k+ chunks leads to Out-of-Memory (OOM) risks.
- **Initial Setup is Heavy:** Building embeddings and indexes takes time and resources.

Future Work

Expand to Multi-Cloud Support: Integrate Azure, GCP, and Kubernetes documentation to create a unified cloud assistant.

Add Code Snippet Retrieval: Surface official AWS CLI, SDK, and CloudFormation examples directly in responses.

LLM Upgrade Options: Explore Qwen-14B or higher param models for deeper reasoning and more detailed answers.

Voice Interface & Chatbot API: Enable voice queries and integrate the assistant into existing DevOps workflows.

THANKS!

Do you have any questions?