

RAG-based Chatbot with Bedrock



Dineshraj Dhanapathy

What is NextWork?

NextWork is an organization that provides projects and resources for learning and development, with a goal of helping everyone find a job they love. [\[1\]](#)

Show details >

Enter your message here

Run

Introducing Today's Project!

RAG (Retrieval Augmented Generation) is a method that combines retrieved data with AI to produce informed answers. In this project, I will demonstrate RAG by merging live search results with AI output.

Tools and concepts

Services I used were Amazon Bedrock, S3, and OpenSearch Serverless. Key concepts I learnt include chunking, embeddings, vector stores, and the importance of syncing data for effective AI-powered

Project reflection

This project took me approximately 2 hours. The most challenging part was setting up the Knowledge Base and troubleshooting model responses. It was most rewarding to see the chatbot provide accurate, real-time answers.

I did this project today to deepen my understanding of AI models and enhance my skills in integrating knowledge bases. The project met my goals by successfully building a functional, responsive AI chatbot.

Understanding Amazon Bedrock

Amazon Bedrock is a managed service offering scalable foundation models and tools to build generative AI applications. I'm using Bedrock in this project to integrate and deploy AI-powered solutions.

My Knowledge Base is connected to S3 because S3 is a secure, scalable, and durable storage solution that hosts our data and enables efficient retrieval and processing for AI models for fast access.

In an S3 bucket, I uploaded documents including FAQs, manuals, and guides for our chatbot's Knowledge Base. My S3 bucket is in the same region as my KB because I'm using Ohio East 2 for low latency

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	Automate Your Browser with AI Agents.pdf	pdf	February 13, 2025, 02:48:12 (UTC+05:30)	17.3 MB	Standard
<input type="checkbox"/>	Build a Three-Tier Web App.pdf	pdf	February 13, 2025, 02:48:12 (UTC+05:30)	16.6 MB	Standard
<input type="checkbox"/>	Building an AI Workflow.pdf	pdf	February 13, 2025, 02:48:43 (UTC+05:30)	16.4 MB	Standard
<input type="checkbox"/>	Create S3 Buckets with Terraform.pdf	pdf	February 13, 2025, 02:48:12 (UTC+05:30)	16.5 MB	Standard
<input type="checkbox"/>	Deploy Backend with Kubernetes.pdf	pdf	February 13, 2025, 03:04:20 (UTC+05:30)	15.3 MB	Standard

My Knowledge Base Setup

My Knowledge Base uses a vector store, which means it efficiently stores and retrieves embeddings for fast AI responses. When I query my Knowledge Base, OpenSearch will quickly find relevant document chunks

Embeddings are numerical representations of text that help AI understand context and meaning. The embedding model I'm using is Titan Text Embeddings v2 because it provides accurate, efficient semantic search.

Chunking is the process of breaking large text into smaller parts for efficient AI processing. In my Knowledge Base, chunks are set to 300 tokens, ensuring the chatbot retrieves precise and relevant info.

The screenshot shows the NextWork Knowledge Base setup interface. It consists of two main sections: 'Step 1: Provide details' and 'Step 2: Setup up data source'.

Step 1: Provide details

Knowledge Base details		
Knowledge Base name nextwork-rag-documentation	Knowledge Base description This Knowledge base stores all documentation at NextWork.	Service role AmazonBedrockExecutionRoleForKnowledgeBase_st1hm
Knowledge base type Knowledge base use vector store	Data source type S3	Log Deliveries —

Step 2: Setup up data source

Data source: s3-bucket-nextwork-rag-bedrock		
Data source name s3-bucket-nextwork-rag-bedrock	Account ID 466742534146 (this account)	S3 URI s3://nextwork-rag-bedrock-dd

AI Models

AI models are important for my chatbot because they generate intelligent, context-aware responses. Without AI models, my chatbot would only retrieve static text without understanding user intent.

To get access to AI models in Bedrock, I had to request model access in the AWS console. AWS needs explicit access because it ensures compliance, security, and proper usage of foundation models.

Models	Access status	Modality	EULA
▼ Amazon (4)	1/4 access granted		
Titan Text Embeddings V2	Access granted	Embedding	EULA
Nova Pro Cross-region inference	Available to request	Text & Vision	EULA
Nova Lite Cross-region inference	Available to request	Text & Vision	EULA
Nova Micro Cross-region inference	Available to request	Text	EULA
▼ Anthropic (4)	0/4 access granted		
Claude 3.5 Haiku Cross-region inference	Available to request	Text	EULA
Claude 3.5 Sonnet V2 Cross-region inference	Available to request	Text & Vision	EULA
Claude 3.5 Sonnet Cross-region inference	Available to request	Text & Vision	EULA
Claude 3 Haiku Cross-region inference	Available to request	Text & Vision	EULA
▼ Meta (8)	2/8 access granted		
Llama 3.3 70B Instruct	Access granted	Text	EULA
Llama 3.2 1B Instruct Cross-region inference	Available to request	Text	EULA
Llama 3.2 3B Instruct Cross-region inference	Available to request	Text	EULA
Llama 3.2 11B Vision Instruct Cross-region inference	Available to request	Text & Vision	EULA
Llama 3.2 90B Vision Instruct Cross-region inference	Available to request	Text & Vision	EULA
Llama 3.1 405B Instruct Cross-region inference	Available to request	Text	EULA
Llama 3.1 70B Instruct Cross-region inference	Available to request	Text	EULA
Llama 3.1 88B Instruct Cross-region inference	Access granted	Text	EULA

Syncing the Knowledge Base

Even though I already connected my S3 bucket when creating the Knowledge Base, I still need to sync because it ensures the latest data is indexed, updated, and ready for accurate retrieval by the AI.

The sync process involves three steps: Ingesting, where Bedrock retrieves data from S3; Processing, where it chunks and embeds the data; and Storing, where Bedrock stores the processed data in OpenSearch Serverless.

The screenshot shows the AWS Bedrock Knowledge Base configuration interface. On the left, the 'Knowledge Base overview' section displays the following details:

- Knowledge Base name: nextwork-rag-documentation
- Knowledge Base ID: 2LFWWYKBN
- Status: Available
- Created date: February 13, 2025, 03:25 (UTC+05:30)
- Log Deliveries: Configure log deliveries and event logs in the [Edit](#) page.
- Retrieval-Augmented Generation (RAG) type: Vector store

The 'Data source (1)' section shows a single data source connected to an S3 bucket named 's3-bucket...'. The 'Tags' section indicates there are no tags assigned to the knowledge base.

On the right, the 'Test Knowledge Base' interface is open, showing the following configuration:

- Generate responses: Enabled
- Model: Llama 3.1 8B In...
- Configurations: Configure your retrieval and responses
- Description: To customize the search strategy for your Knowledge Base, select the configurations icon
- Test message: Test your Knowledge Base by running a query to generate responses. To disable response generation and only see retrieved information stored from your vector store, turn off Generate responses above.
- Message input field: Enter your message here
- Run button: Run

DI

Dineshraj Dhanapathy
NextWork Student

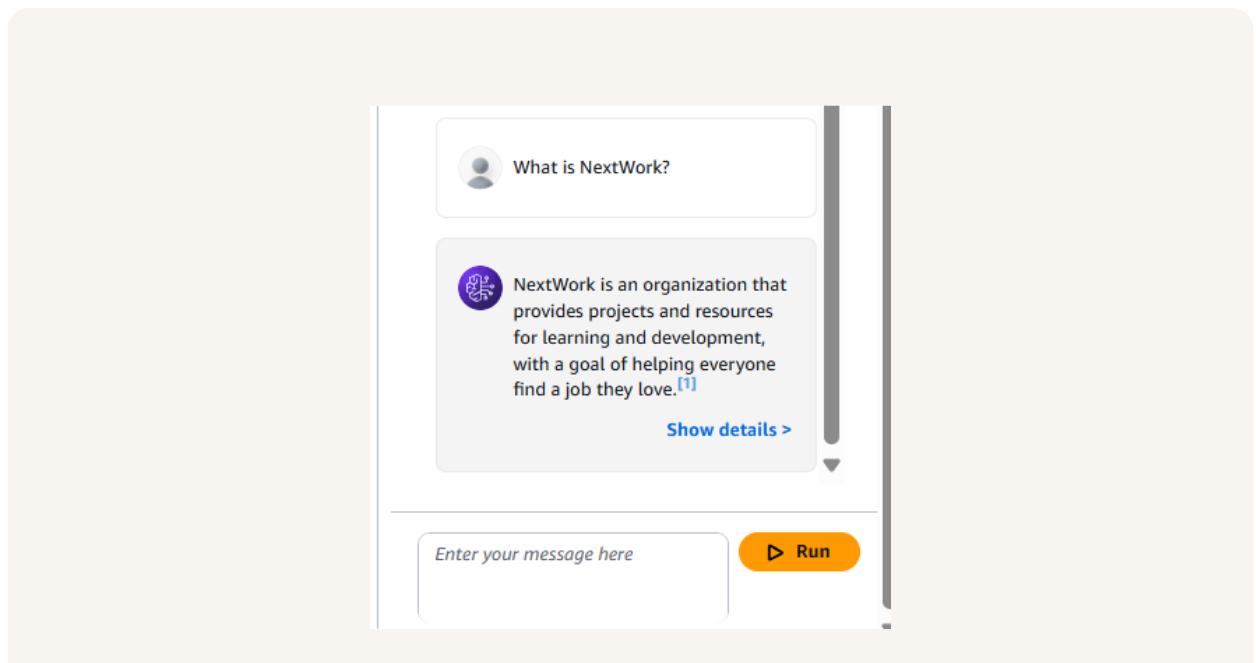
NextWork.org

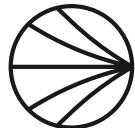
Testing My Chatbot

I initially tried to test my chatbot using Llama 3.1 8B as the AI model, but it lacked the depth needed for complex queries. I had to switch to Llama 3.3 70B because it provides more accurate, nuanced responses.

When I asked about topics unrelated to my data, my chatbot struggled to provide relevant answers. This proves that the chatbot relies on the knowledge base, and without related data, its accuracy drops.

You can also turn off the Generate Responses setting to prevent the AI from automatically generating answers. This allows you to manually adjust or review the responses before they are delivered, giving more control over the interaction.





NextWork.org

Everyone should be in a job they love.

Check out nextwork.org for
more projects

