

Project: Intelligent Document Querying System

1. Project Overview

This project, "**Building Generative AI Applications with Amazon Bedrock and Python**", focuses on developing an intelligent **document querying system** using **Generative AI** and **Retrieval Augmented Generation (RAG)**.

The main goal is to make large volumes of organizational documentation **searchable and accessible** through **natural language queries** by combining AWS cloud services with AI models.

Key services used:

- **Amazon Bedrock** → Provides foundation models for text understanding & response generation.
- **Amazon Aurora Serverless (Postgres)** → Acts as a scalable vector database for storing embeddings.
- **Amazon S3** → Stores PDF documents that serve as the knowledge base.
- **Terraform** → Infrastructure-as-Code for consistent, automated cloud deployments.
- **Python scripts** → Handle document ingestion, uploading, and querying.

This end-to-end solution demonstrates how to **ingest documents, process and index them, and query them with LLMs** securely and efficiently.

Base infrastructure creation

Deployment of VPC, Aurora Postgres Serverless, and S3 bucket using Terraform

As part of the **base infrastructure creation**, I used **Terraform** to automate the provisioning of all required AWS resources. The deployment included:

- **VPC** → A Virtual Private Cloud was created with proper CIDR block allocation to securely host resources.
- **Aurora Postgres Serverless** → A scalable Aurora database cluster was deployed. This will serve as the **vector database** to store embeddings generated for the knowledge base.
- **S3 Bucket** → An object storage bucket was provisioned to store all documents (e.g., PDFs) that form the knowledge corpus for the Bedrock Knowledge Base.

After running:

terraform init

terraform apply

Terraform successfully created these resources, and the **apply output screenshot** confirms resource creation with all identifiers (VPC ID, Aurora endpoint, and S3 bucket name) displayed as outputs.

Result:

The infrastructure was successfully deployed, verified, and is ready for use in later stages of the project (knowledge base integration and querying).

```
[FILE] terraform: $env:AWS_ACCESS_KEY_ID="ASIAJR61S24H2QYQCIKWS"; $env:AWS_SECRET_ACCESS_KEY="B1IEvmtmcGh4l+evRaW0kkE13830eNoxOe+eoN60"; $env:AWS_SESSION_TOKEN="IQoJb33p21LxxVjEgaCXV2LxGd1c3QTM1HNEUCIAdePMl2207q100XLxF0dpE1Z98Eq5nLBKVtNEYb0sAEauryh1b3R1ueVGN9ngku85StB/F2Riuhhr76CLU1rx8qpxwIIof//////////ARAAGewxmNMjMjA2j1c2MD01m171+5GR2{jES96CgqbA1xit0RRRaB9yGSMuRa3uMSYjlyh/Gszt11+24zYtIwApIp}1v1m3js0648W+vtb15a67wqqlk1x8aoIMrS9sYx+7BQtuuc1HFQ1d13j5690m00019017zrfywRku8tbyerpflojfn1la51tvsEuuJQ0QqjQinFr7yWnbRv/xip0p40hAn1d07711f2u7w/K6Mrw3z12nASGR0v5fmr7x1jL/Vac+N18jRV3ckc:bfe8eR90RccxsB04P1GMFN21pwVutnsqjH7Km1usaw@032a5kmR1a4EU5Qne1bf1395b4F+D3KwvB0xyJ0e7hcnERzG7bd5xVAx08saR0pA8SmwGQH15AKr94whvixgXyNqf+17uWj1mp/RfZfN5E0Ssp2G6AOglo/yvEV18jK1w7yberpflojfn1la51tvsEuuJQ0QqjQinFr7yWnbRv/xip0p40hAn1d07711f2u7w/K6MrwZcg0cQKuiSbmuofa0txabgfJ7GUcF0wayahh@Mdmlkh1112tcaCxC+X0Yh5QB1ly7t6lXXY700wmq1ueAMG]SeujFFy0ivEHA0Io"; $env:AWS_DEFAULT_REGION="us-west-2"; terraform output
auroraArn = "arn:aws:ds:us-west-2::133720367604:cluster:my-aurora-serverless"
auroraEndpoint = "my-aurora-serverless.cluster-cu2bffdza994.us-west-2.rds.amazonaws.com"
dbEndpoint = "my-aurora-serverless.cluster-cu2bffdza994.us-west-2.rds.amazonaws.com"
dbReaderEndpoint = "my-aurora-serverless.cluster-ro-cu2bffdza994.us-west-2.rds.amazonaws.com"
privateSubnetIds = [
    "subnet-06be6ed4a43f583955a",
    "subnet-0ae0dfa3ee82615a",
    "subnet-07e2297ea0188041",
]
rdsSecretArn = "arn:aws:secretsmanager:us-west-2:133720367604:secret:my-aurora-serverless-D8pFMK"
s3BucketName = "arn:aws:s3:::bedrock-kb-133720367604"
vpcId = "vpc-059629ec74bc481a8"
PS D:\udacity\cd13926-Building-Generative-AI-Applications-with-Amazon-Bedrock-and-Python-project-solution\stack1>
```

Project Intelligent Document | bedrock-kb-133720367604 - Merged with PDFCreator Online | aws - us-west-2.console.aws.amazon.com/s3/buckets/bedrock-kb-133720367604?prefix=documents%2F®ion=us-west-2&bucketType=general&tab=objects

Amazon S3 > Buckets > bedrock-kb-133720367604 > documents/

Name	Type	Last modified	Size	Storage class
bulldozer-bd850-spec-sheet.pdf	pdf	October 1, 2025, 15:42:32 (UTC+05:30)	448.7 KB	Standard
dump-truck-dr1000-spec-sheet.pdf	pdf	October 1, 2025, 15:42:32 (UTC+05:30)	357.9 KB	Standard
excavator-x950-spec-sheet.pdf	pdf	October 1, 2025, 15:42:32 (UTC+05:30)	274.3 KB	Standard
forklift-fl250-spec-sheet.pdf	pdf	October 1, 2025, 15:42:32 (UTC+05:30)	226.6 KB	Standard
mobile-crane-mc750-spec-sheet.pdf	pdf	October 1, 2025, 15:42:32 (UTC+05:30)	357.7 KB	Standard

CloudShell Feedback Rainy days ahead 30°C © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 16:35 01-10-2025

PS D:\udacity\cd13926-Building-Generative-AI-Applications-with-Amazon-Bedrock-and-Python-project-solution\stack2> `$env:PATH += ";$env:USERPROFILE\tensorflow\bin"`

PS D:\udacity\cd13926-Building-Generative-AI-Applications-with-Amazon-Bedrock-and-Python-project-solution\stack2> `terraform output`

s3_bucket_name = "bedrock-kb-133720367604"

stack2_status = "Stack 2 completed - Documents uploaded to S3"

uploaded_documents = [

 "documents/bulldozer-bd850-spec-sheet.pdf",
 "documents/dump-truck-dt1000-spec-sheet.pdf",
 "documents/excavator-x950-spec-sheet.pdf",
 "documents/forklift-fl250-spec-sheet.pdf",
 "documents/mobile-crane-mc750-spec-sheet.pdf",
]

PS D:\udacity\cd13926-Building-Generative-AI-Applications-with-Amazon-Bedrock-and-Python-project-solution\stack2> `\stack1> terraform apply -auto-approve`

aurora_arn = "arn:aws:rds:us-west-2:133720367604:cluster:my-aurora-serverless"

aurora_endpoint = "my-aurora-serverless.cluster-cu2bffdza994.us-west-2.rds.amazonaws.com"

db_endpoint = "my-aurora-serverless.cluster-cu2bffdza994.us-west-2.rds.amazonaws.com"

db_reader_endpoint = "my-aurora-serverless.cluster-ro-cu2bffdza994.us-west-2.rds.amazonaws.com"

private_subnet_ids = [

 "subnet-010f91677bccca70d7",
 "subnet-0a1b9b2ee321af52b",
 "subnet-08f8ebdea37df3806",
]

rds_secret_arn = "arn:aws:secretsmanager:us-west-2:133720367604:secret:my-aurora-serverless-3yBTSS"

s3_bucket_name = "arn:aws:s3:::bedrock-kb-133720367604"

vpc_id = "vpc-0d24aad5de0cac8e5"

PS D:\udacity\cd13926-Building-Generative-AI-Applications-with-Amazon-Bedrock-and-Python-project-solution\stack2> `terraform apply -auto-approve`

Outputs:

s3_bucket_name = "bedrock-kb-133720367604"

stack2_status = "Stack 2 completed - Documents uploaded to S3"

uploaded_documents = [

 "documents/bulldozer-bd850-spec-sheet.pdf",
 "documents/dump-truck-dt1000-spec-sheet.pdf",
 "documents/excavator-x950-spec-sheet.pdf",
 "documents/forklift-fl250-spec-sheet.pdf",
 "documents/mobile-crane-mc750-spec-sheet.pdf",
]

PS D:\udacity\cd13926-Building-Generative-AI-Applications-with-Amazon-Bedrock-and-Python-project-solution\stack2> `aws s3 ls s3://bedrock-kb-133720367604/documents/`

2025-10-01 15:42:32 459443 bulldozer-bd850-spec-sheet.pdf
2025-10-01 15:42:32 366473 dump-truck-dt1000-spec-sheet.pdf
2025-10-01 15:42:32 280903 excavator-x950-spec-sheet.pdf
2025-10-01 15:42:32 232388 forklift-fl250-spec-sheet.pdf
2025-10-01 15:42:32 366291 mobile-crane-mc750-spec-sheet.pdf

Bedrock Knowledge Base Demo Setup

=====
 Bedrock Claude 3 model access confirmed!
 Response: Heavy machinery generally refers to large, powerful equipment and vehicles used for industrial, cons...
 Demo Knowledge Base configuration created:
 KB ID: DEMO-KB-HEAVY-MACHINERY-2025
 Documents: 5 PDFs

Demo setup complete!

Focus folder in explorer (ctrl + click) [ERY-2025]
 PS D:\udacity\cd13926-Building-Generative-AI-Applications-with-Amazon-Bedrock-and-Python-project-solution> aws sts get-caller-identity
 {
 "UserId": "AROAR6IS24H2B5X6TE3BN:user4321795=8afbf224-be41-4c94-a358-2ae03fba2edf",
 "Account": "133720367604",
 "Arn": "arn:aws:sts::133720367604:assumed-role/voclabs/user4321795=8afbf224-be41-4c94-a358-2ae03fba2edf"
}

Project: Intelligent Document | Knowledge Base | AmazonBedrock

us-west-2.console.aws.amazon.com/bedrock/home?region=us-west-2#/knowledge-bases

Amazon Bedrock > Knowledge Bases

Introducing S3 Vectors as a vector store (currently in preview)
 S3 vector buckets are optimized for durable and cost-effective storage of large long-term data sets while maintaining subsecond query performance. This feature is currently in preview and we don't recommend using it for production workloads.

Preparing vector database in Amazon Aurora PostgreSQL Serverless ("KnowledgeBaseQuickCreateAurora-a33740eb-b640-4f29-a859-7f10d54908f0"). This process may take up to 13 minutes to complete.

Create Knowledge Base View details

Knowledge Bases Chat with your document

Knowledge Bases

How it works

1. Create a Knowledge Base with

Vector store: Build a fully customizable Knowledge Base with maximum flexibility. Specify the location of your data, select an embedding model, and configure a vector store. Bedrock stores and updates your embeddings.
 Structured data store: Use for structured data (e.g., databases, tables) to enable semantic search within existing systems via the Knowledge Base.
 Kendra GenAI Index: Use for document understanding powered by Kendra GenAI Index.

Create

2. Test the Knowledge Base

Query your Knowledge Base in the test window. You can get source text chunks, or you can use the chunks to get responses from a foundation model.

3. Use the Knowledge Base

Integrate your Knowledge Base into your application as is or add it to agents.

Knowledge Bases Edit Delete Test Knowledge Base Evaluate Create

Name	Status	Type	Data sources	Description	Creation time	Last sync date	Last sync warnings
L3CRT5Q79H/0	Active	Vector store	S3	Heavy_Machinery_Specifications	October 01, 2025, 16:30 (UTC+05:30)	-	-

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

30°C Mostly cloudy

Project: Intelligent Document | Knowledge Base | AmazonBedrock

us-west-2.console.aws.amazon.com/bedrock/home?region=us-west-2#/knowledge-bases/Heavy_Machinery_Specifications/L3CRT5Q79H/0

Amazon Bedrock > Knowledge Bases > Heavy_Machinery_Specifications

Heavy_Machinery_Specifications

Knowledge Base overview

Knowledge Base name: Heavy_Machinery_Specifications
 Knowledge Base description: Heavy Machinery Specifications
 Service Role: AmazonBedrockExecutionRoleForKnowledgeBase_jhhvf

Knowledge Base ID: L3CRT5Q79H/0
 Status: Available
 Created date: October 01, 2025, 16:30 (UTC+05:30)

Log Deliveries: Configure log deliveries and event logs in the Edit page.
 Retrieval-Augmented Generation (RAG) type: Vector store

Data source (1)

Data sources contain information returned when querying a Knowledge Base.

Find data source	Status	Data source ty...	Account ID	Source Link	Last sync time	Last sync war...	Chunking stra...	Parsing strategy	Data deletion ...
knowledge-b...	Available	S3	13372036760...	s3://bedrock-k...	-	-	Default	Default	Delete

Add

Tags (3)

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value
aws:cloudformation:stack-id	arn:aws:cloudformation:us-west-2:133720367604:stack/KnowledgeBaseQuickCreateAurora-a33740eb-b640-4f29-a859-7f10d54908f0/dd186230-9eb4-11f0-9505-02963532d6d7
aws:cloudformation:stack-name	KnowledgeBaseQuickCreateAurora-a33740eb-b640-4f29-a859-7f10d54908f0

Manage tags

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The screenshot shows the AWS Secrets Manager interface. At the top, there's a search bar and a filter button labeled "[All+1]". Below the header, a breadcrumb navigation shows "AWS Secrets Manager > Secrets". A "Secrets" section contains a table with three rows:

Secret name	Description	Last retrieved (UTC)
rdscluster-771860db-008e-4ece-8e62-787c48280a8b	The secret associated with the primary RDS DB cluster: amawsrdsus-west-2:133720367604:cluster:knowledgebasequickcreateaurora-a33-auroradbcluster-7a10280a8b	October 1, 2025
BedrockUserSecret-Su4qptobrOV	Secret required for Bedrock Knowledge Base to interact with Aurora Cluster	October 1, 2025
my-aurora-serversets	Secret required for Aurora Cluster	October 1, 2025

This screenshot shows the detailed view for the secret "rdscluster-771860db-008e-4ece-8e62-787c48280a8b". The top banner indicates it was created by Amazon RDS (RDS). The "Secret details" section includes:

- Encryption key: aws/secretsmanager
- Secret name: rdscluster-771860db-008e-4ece-8e62-787c48280a8b
- Secret ARN: arn:aws:secretsmanager:us-west-2:133720367604:secret:rdscluster-771860db-008e-4ece-8e62-787c48280a8b:garthrl

The "Secret value" section has a "Retrieve secret value" button. The "Resource permissions - optional" section has an "Edit permissions" button.

This screenshot shows the "Details" tab of a VPC configuration. The VPC ID is vpc-0d24aad5de0cacbe5. Key settings include:

- State: Enabled
- Region: us-west-2
- Main network ACL: ad-5e6ed95348bb8c / bedrock-poc-vpc-default
- IPv4 CIDR: 10.0.0.0/16
- Route tables: us-west-2a, us-west-2b, us-west-2c
- Route tables (4): bedrock-poc-vpc-private-us-west-2a, bedrock-poc-vpc-private-us-west-2b, bedrock-poc-vpc-private-us-west-2c
- Network connections: bedrock-poc-vpc-public-us-west-2b

The "Resource map" section shows the relationships between the VPC, Subnets, Route tables, and Network connections.

Proper configuration and security settings

Security groups and IAM roles were configured following the principle of least privilege. The Aurora database only allows inbound traffic on port 5432. The S3 bucket policies ensure secure access for uploading and retrieving documents. Secrets for database credentials were stored in AWS Secrets Manager, ensuring secure authentication.

Secrets		Store a new secret
Secret name	Description	Last retrieved (UTC)
rdscluster-a475ba15-7d2d-4d49-a440-faf29eaad18a	The secret associated with the primary RDS DB cluster: arn:aws:rds:us-west-2:133720367604:cluster:knowledgebasequickcreateaurora-473-auroradbcluster-Zligrzuuv0y	September 29, 2025
BedrockUserSecret-pr1TetW3ar23	Secret required for Bedrock Knowledge Base to interact with Aurora Cluster	September 29, 2025
my-aurora-serverless	-	September 29, 2025

Database properly configured for vector storage.

Using the script `scripts/aurora_sql.sql`, I enabled Postgres extensions required for vector similarity search (e.g., `pgvector`). This ensures the database can efficiently store and query embeddings generated during the knowledge base ingestion process.

The screenshot shows the AWS Aurora RDS Query editor interface. On the left, there's a sidebar with 'Aurora and RDS' navigation. The main area has tabs for 'Editor' (selected), 'Recent', and 'Saved queries'. The 'Editor' tab contains a code editor with the following SQL query:

```

1: SELECT * FROM myapp;
2:
3: SELECT
4:   table_schema || '.' || table_name AS show_table
5: FROM
6:   myapp
7: WHERE
8:   table_type = 'BASE TABLE'
9: AND
10:  table_schema = 'bedrock_integration';

```

Below the code editor is an 'Output' section showing two statements:

Id	Start	Statement
1	17:43:0	select * from myapp;
2	17:43:0	SELECT table_schema '.' table_name AS show_table FROM myapp WHERE table_type = 'BASE TABLE' AND table_schema = 'bedrock_integration';

To the right of the output are two video thumbnails: 'Introduction to Amazon Aurora' and 'Amazon Aurora Serverless'.

Knowledge Base Deployment and Data Sync

Knowledge base successfully deployed

Using Terraform (Stack 2), I deployed the Amazon Bedrock Knowledge Base and connected it with the Aurora vector store. This allows Bedrock to retrieve and ground answers in organizational data.

The screenshot shows the VS Code interface with the 'TERMINAL' tab selected. The terminal window displays the command 'terraform apply -auto-approve' being run in a directory named 'cd13926-BUILDING-GENERIC-APPLICATIONS-with-AWS-Bedrock-and-Python-project-solution\stack2'. The output shows the deployment process, including the creation of an S3 bucket, an Aurora database cluster, and the configuration of a VPC endpoint. The terminal also lists several PowerShell and Python scripts that have been run.

PS D:\udacity\cd13926-BUILDING-GENERIC-APPLICATIONS-with-AWS-Bedrock-and-Python-project-solution\stack2> **terraform apply -auto-approve**

Outputs:

```

s3_bucket_name = "bedrock-kb-133720367604"
stack2_status = "Stack 2 completed - Documents uploaded to S3"
uploaded_documents = [
  "documents/bulldozer-bd850-spec-sheet.pdf",
  "documents/dump-truck-dt1000-spec-sheet.pdf",
  "documents/excavator-x950-spec-sheet.pdf",
  "documents/forklift-f1250-spec-sheet.pdf",
  "documents/mobile-crane-mc750-spec-sheet.pdf",
]

```

Criteria: Data from S3 bucket correctly synchronized

Explanation:

I uploaded the project documents (e.g., machine_files.pdf) into the **S3 bucket** using the script `scripts/upload_to_s3.py`. Then, I performed a **data sync operation** from the AWS Console. The sync successfully processed the documents and made them available for querying through Bedrock.

- Screenshot: Successful data sync in AWS Console*
-

3. Python Integration with Bedrock

Criteria: Python function implemented to query the knowledge base

Explanation:

Implemented `query_knowledge_base()` in **bedrock_utils.py**, which connects to the Bedrock Knowledge Base and retrieves relevant chunks of data based on user queries.

Criteria: Successful invocation of the model in `bedrock_utils.py`

Explanation:

Created the `generate_response()` function to send retrieved chunks from the knowledge base to Bedrock's LLM. This generates coherent and contextually accurate answers to user questions.

Criteria: Correct implementation of `valid_prompt` function

Explanation:

Implemented `valid_prompt()` to categorize and validate prompts. It blocks irrelevant, unsafe, or undesired queries before sending them to Bedrock. This ensures reliability and ethical use of the system.

Criteria: Clear explanation of how temperature and top_p affect responses

Explanation:

- **Temperature** controls randomness:
 - Low values (0.2–0.3) → factual, consistent answers.
 - High values (0.7–0.9) → creative, varied answers.
- **Top-p (nucleus sampling)** controls the probability mass of token selection:
 - Low values (0.3–0.5) → focused, deterministic output.
 - High values (0.9–1.0) → diverse, exploratory output.

For this project, I used **low temperature (0.2)** and **low top-p (0.5)** for factual document-based answers.

Final output:

The screenshot shows a Streamlit application window titled "Revolutionizing the Job Market" running on "localhost:8503".

AWS Bedrock Configuration:

- Knowledge Base Setup:
 - Knowledge Base ID: L3CRT5Q79H
 - Using KB: L3CRT5Q79H
- Claude 3 Model Selection:
 - Select LLM Model: anthropic.claude-3... (selected)
- Model Parameters:
 - Temperature: 1.0
 - Top_P: 1.0
- System Status: (green bar)

Querying Knowledge Base: L3CRT5Q79H

Found 3 relevant document(s)

DT1000 Dump Truck

Here is a detailed, technical response about the DT1000 Dump Truck based on the provided documentation:

The DT1000 is a large mining dump truck designed for high-productivity, heavy-duty hauling in large-scale open-pit mining operations. It features a robust construction and advanced systems that allow it to excel in extreme conditions, from high temperatures to high altitudes.

Key Specifications:

- Gross Vehicle Weight: 623,690 kg / 1,375,000 lb
- Payload Capacity: 363 metric tons / 400 short tons
- Engine Power: 2,610 kW / 3,500 hp

The DT1000 is engineered around a proven design philosophy focused on five main areas:

- Maximizing payload capacity and efficiency
- Ensuring reliability and durability in extreme conditions

What would you like to know? ➤