Configuration Manual

MSc Research Project
Cloud Computing

Hunaid Vekariya
Student ID: 23235951

School of Computing
National College of Ireland

Supervisor: Aqeel Kazmi

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Hunaid Vekariya
Student ID:	23235951
Programme:	Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	Aqeel Kazmi
Submission Due Date:	12/12/2024
Project Title:	Configuration Manual
Word Count:	XXX
Page Count:	7

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	*Hunaid Vekariya*
Date:	28th January 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Hunaid Vekariya
23235951

1 Introduction

This manual provides a comprehensive steps to perform and deploy the event driven architecture to retrieve and generate information from aws bedrock models with context to the knowledge base

2 System Requirements

2.1 Cloud Service Requirements

Access to AWS Bedrock, lambda, api gateway, cloudwatch, s3 bucket, IAM for permissions should be there. Opensearch to store the vector database

The aws bedrock models are not enabled by default. Needs to be requested from AWS in order to access the models.

3 Software Requirements

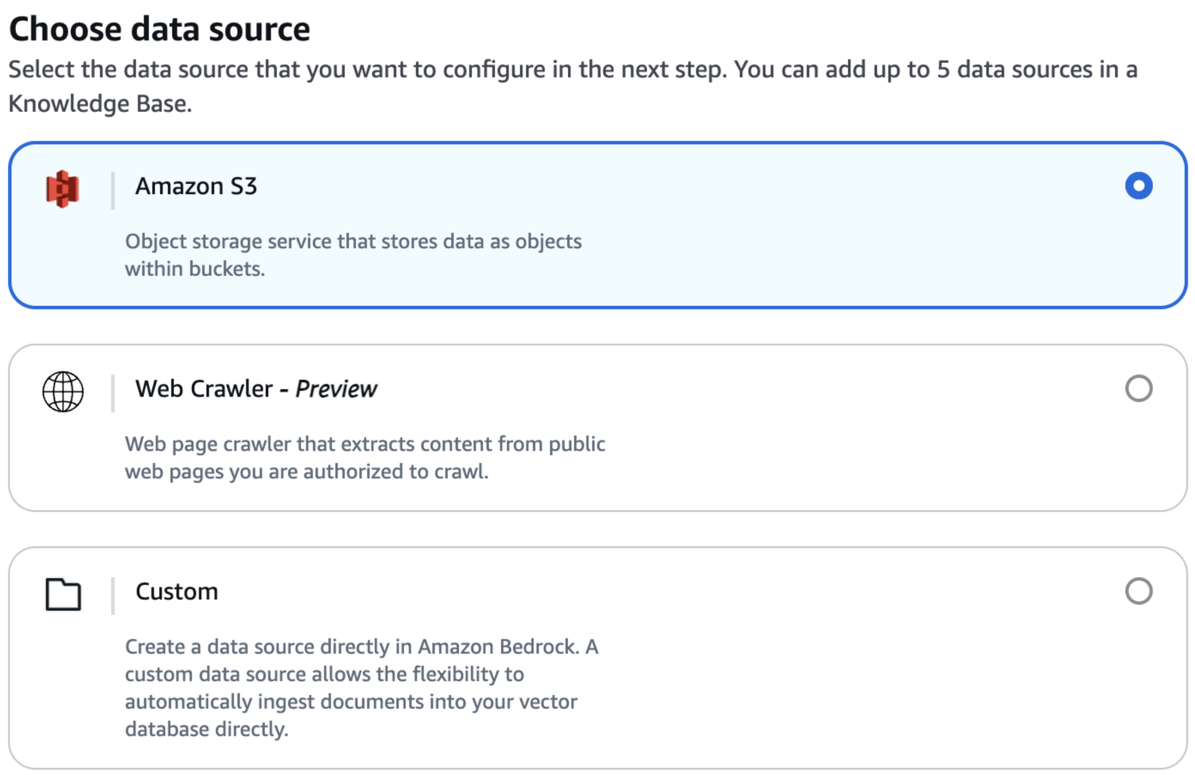
Python 3.10 libraries

1. All the interations and logic layer is being processed through python3.10 for this project. For development python3.10 must be used.
2. Boto3 - is used to interact with AWS services from Lambda which is AWS Bedrock single api.
3. json - the payload is in json format so it is used to read the data and save it to dynamodb
4. Datetime - To manage the date and time of the event happening

4 Knowledge Base installation

To create the knowledge base select the option "Create knowledge base with vector Store

Fig 1 Shows about sourcing the data from three different methods using Amazon S3 where you can add csv, documents, structured and unstructured data. Through webcrawler with providing the urls of websites you want to embedd and customer sources directly into aws bedrock



Choose data source

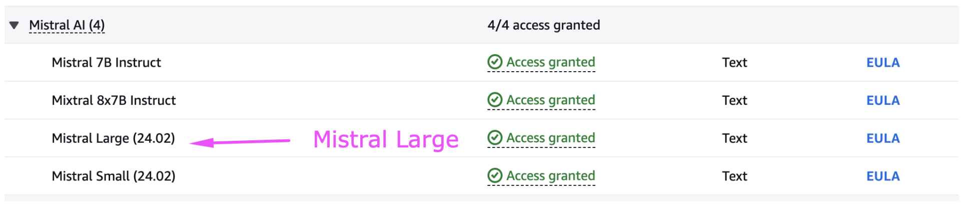
Select the data source that you want to configure in the next step. You can add up to 5 data sources in a Knowledge Base.

[✓ Amazon S3] Object storage service that stores data as objects within buckets.

Web Crawler - *Preview* Web page crawler that extracts content from public web pages you are authorized to crawl.

Custom Create a data source directly in Amazon Bedrock. A custom data source allows the flexibility to automatically ingest documents into your vector database directly.

Figure 1: Enter Caption



Mistral AI (4)	4/4 access granted		
Mistral 7B Instruct	Access granted	Text	EULA
Mixtral 8x7B Instruct	Access granted	Text	EULA
Mistral Large (24.02)	Access granted	Text	EULA
Mistral Small (24.02)	Access granted	Text	EULA

Figure 2: Data Parsing and Chunking

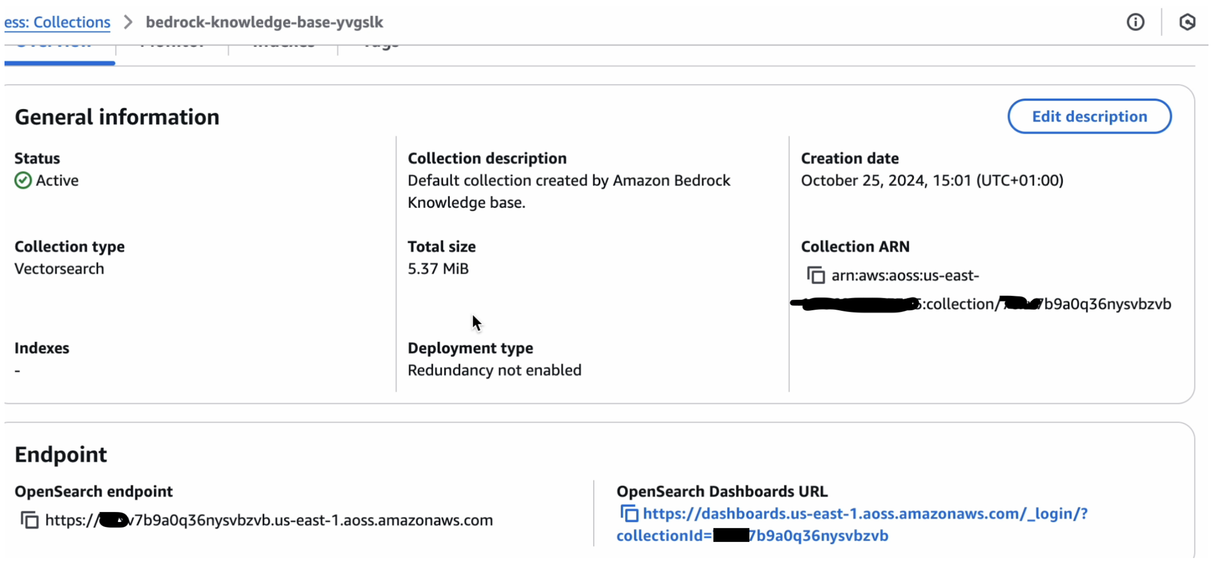
Fig 2 shows about the data parsing and chunking which can optimize the opensearch vector store even further. Using the provided tokens and overlap percentage it would be a best strategy if the data is heterogeneous.

4.1 Opensearch

Opensearch is created automatically from AWS Bedrock knowledge base *AWS Bedrock Knowledge Base RAG Workshop* (n.d.) console once the dataset is added. It should look like below

5 Json payload

The schema is very important to trigger the whole architecture otherwise it cannot be accomplished. Please find the json schema for the payload in the artifacts.


ess: Collections > bedrock-knowledge-base-yvgslk

General information

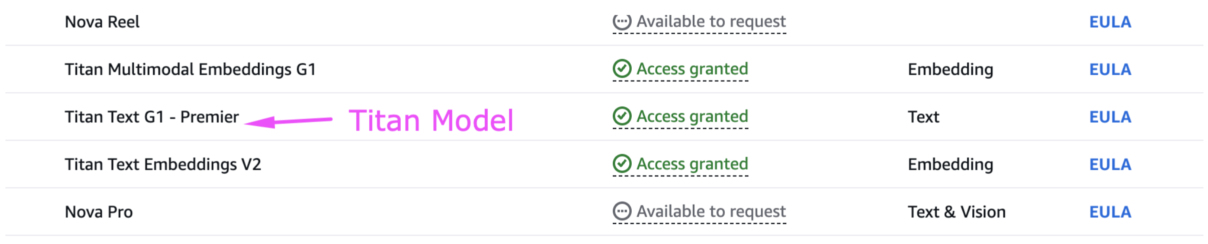
Edit description

Status Active	**Collection description** Default collection created by Amazon Bedrock Knowledge base.	**Creation date** October 25, 2024, 15:01 (UTC+01:00)
Collection type Vectorsearch	**Total size** 5.37 MiB	**Collection ARN** arn:aws:aoss:us-east-[redacted]:collection/[redacted]7b9a0q36nysvbzvb
Indexes -	**Deployment type** Redundancy not enabled	

Endpoint

OpenSearch endpoint https://[redacted]v7b9a0q36nysvbzvb.us-east-1.aoss.amazonaws.com	**OpenSearch Dashboards URL** https://dashboards.us-east-1.aoss.amazonaws.com/_login/?collectionId=[redacted]7b9a0q36nysvbzvb

Figure 3: Opensearch



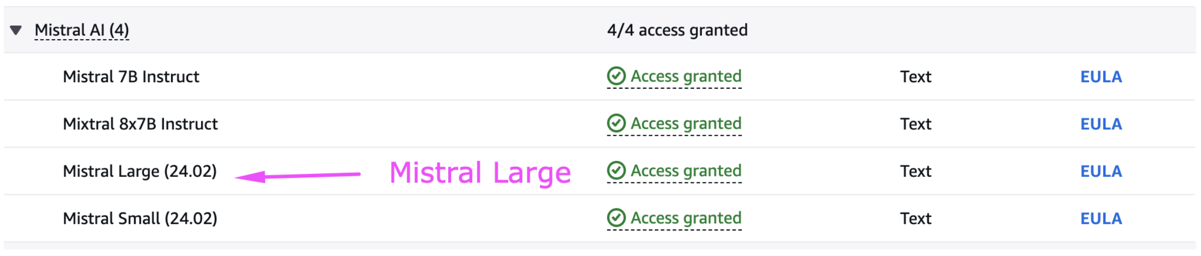
Nova Reel	Available to request		EULA
Titan Multimodal Embeddings G1	Access granted	Embedding	EULA
Titan Text G1 - Premier	Access granted	Text	EULA
Titan Text Embeddings V2	Access granted	Embedding	EULA
Nova Pro	Available to request	Text & Vision	EULA

Figure 4: Enable Titan Text G1 - Premier

6 Bedrock Models Intialization

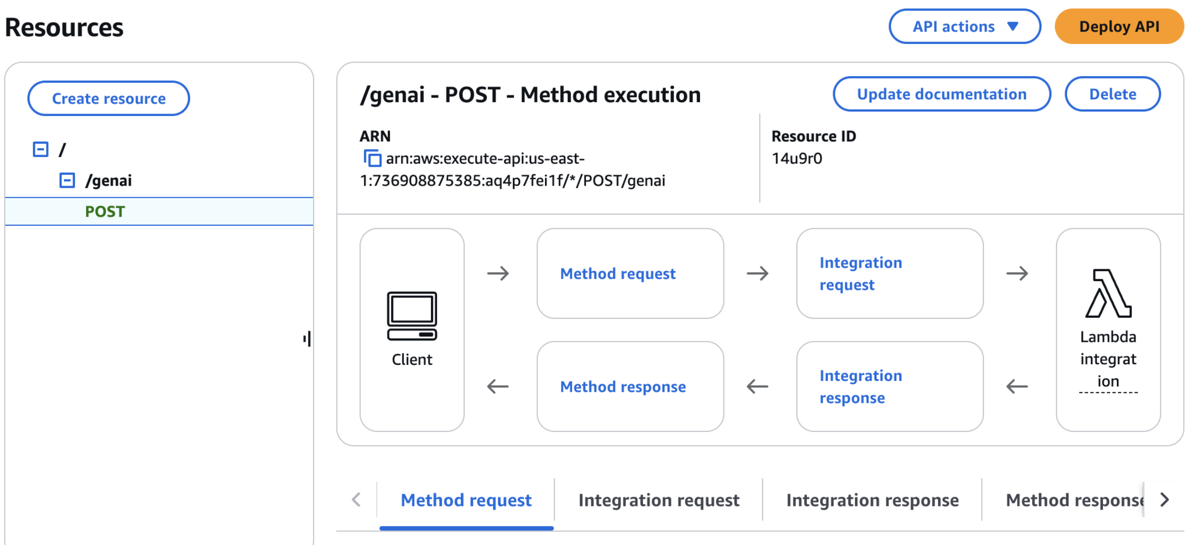
The aws bedrock provides a single api to connect with all the mdoels specifying the model id and knowledge base Id if available. The manual uses two Models as listed below which needs to be enabled prior using them

1. Titan Text G1 - Premier *Amazon Bedrock Titan Models - User Guide* (n.d.) 2. Mistral Large (24.02) *Amazon Bedrock - Mistral* (n.d.)



Mistral AI (4)	4/4 access granted		
Mistral 7B Instruct	Access granted	Text	EULA
Mixtral 8x7B Instruct	Access granted	Text	EULA
Mistral Large (24.02)	Access granted	Text	EULA
Mistral Small (24.02)	Access granted	Text	EULA

Figure 5: Enable Mistral Large



ARN
arn:aws:execute-api:us-east-1:736908875385:aq4p7fei1f/*/POST/genai

Resource ID
14u9r0

Figure 6: API Gateway Configuration

7 Serverless Architecture

7.0.1 Api Gateway

API Gateway to consume webhooks. Create an api gateway with integration to Lamda function. The api gateway must be POST to receive the webhook and process it to Lambda. The lambda processed the event and send back 200 to API gateway which in return is sent to the sender of webhook. Webhook can be triggered from Postman or python cli.

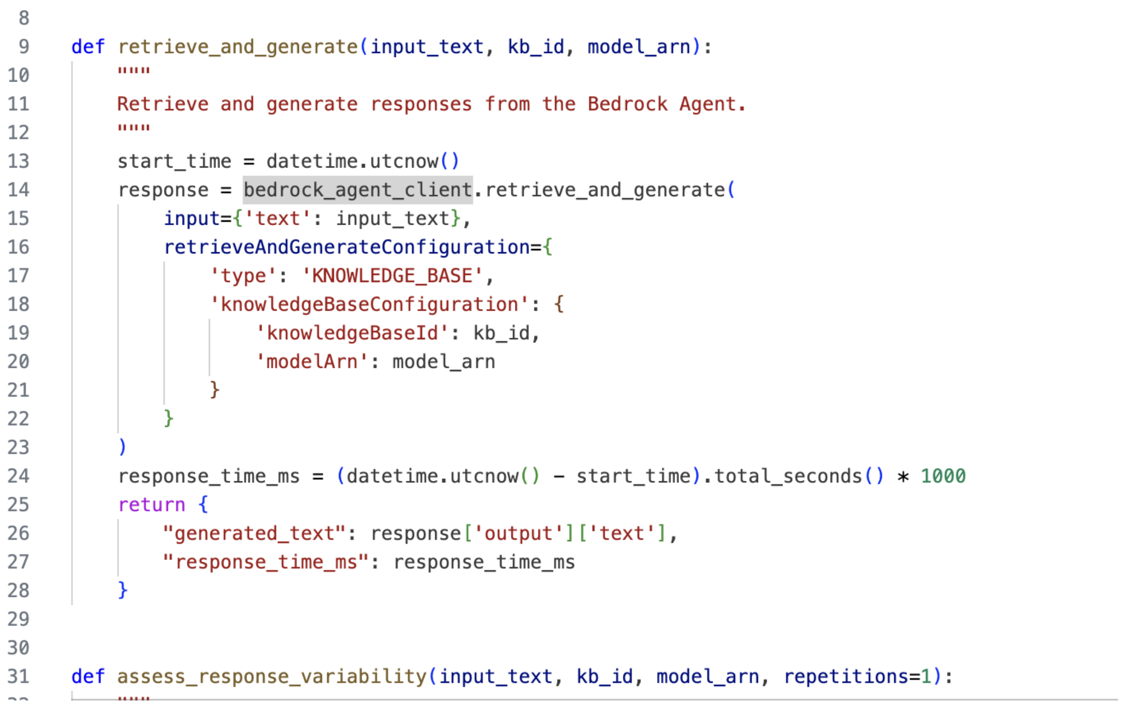
7.0.2 Lambda

Three lambda function are needed to be created for this architecture

- **First Lambda Function** - To pass the webhook to Dynamodb form API Gateway.
-
- **Second Lambda Function** - To consume the new json body from Dynamodb and pre-process the data further and convert it into a prompt.
-
- **Third Lambda Function** - To consume the prompt from the dynamodb table and use the retrieve api with aws bedrock knowledge base to generate the responses.

7.0.3 DynamoDb

Two tables are created for DynamoDb, the partition key are as described in the Fig-6. The tables are enabled with Dynamodb Streams which triggers other services like lambda, kinesis when a new row or event is inserted, deleted, updated or replaced in the table. In our case we only handle new inserted items which will trigger lambda function.



```python

def retrieve_and_generate(input_text, kb_id, model_arn):
    """
    Retrieve and generate responses from the Bedrock Agent.
    """
    start_time = datetime.utcnow()
    response = bedrock_agent_client.retrieve_and_generate(
        input={'text': input_text},
        retrieveAndGenerateConfiguration={
            'type': 'KNOWLEDGE_BASE',
            'knowledgeBaseConfiguration': {
                'knowledgeBaseId': kb_id,
                'modelArn': model_arn
            }
        }
    )
    response_time_ms = (datetime.utcnow() - start_time).total_seconds() * 1000
    return {
        "generated_text": response['output']['text'],
        "response_time_ms": response_time_ms
    }


def assess_response_variability(input_text, kb_id, model_arn, repetitions=1):
    """
```

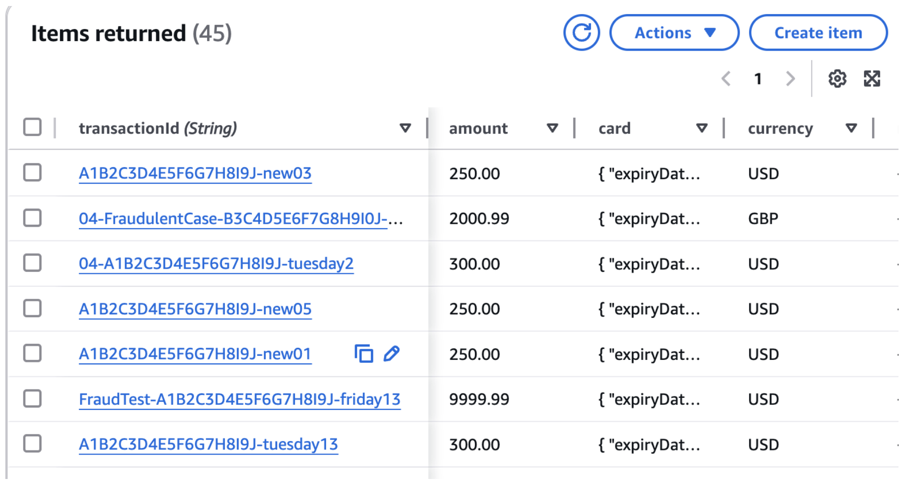
Figure 7: Lambda Code for Bedrock

7.1 Cloudwatch Monitoring

Cloudwatch is an important managed service which store all the logs generated by the lambda function in order to debug and understand the responses. The responses are stored in cloudwatch once generated by the LLM and RAG. The log group would be automatically created on creation of lambda function. One of the example outputs of logs for bedrock is mentioned below in Fig 9

References

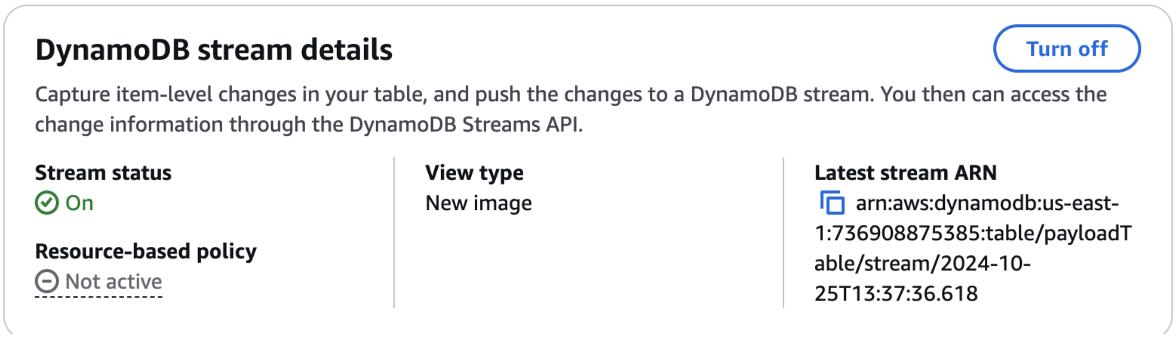
Amazon Bedrock - Mistral (n.d.). `https://aws.amazon.com/bedrock/mistral/`. Ac-



Items returned (45)

transactionId *(String)*	amount	card	currency
A1B2C3D4E5F6G7H8I9J-new03	250.00	{ "expiryDat...	USD
04-FraudulentCase-B3C4D5E6F7G8H9I0J-...	2000.99	{ "expiryDat...	GBP
04-A1B2C3D4E5F6G7H8I9J-tuesday2	300.00	{ "expiryDat...	USD
A1B2C3D4E5F6G7H8I9J-new05	250.00	{ "expiryDat...	USD
A1B2C3D4E5F6G7H8I9J-new01	250.00	{ "expiryDat...	USD
FraudTest-A1B2C3D4E5F6G7H8I9J-friday13	9999.99	{ "expiryDat...	USD
A1B2C3D4E5F6G7H8I9J-tuesday13	300.00	{ "expiryDat...	USD

Figure 8: Enter Caption



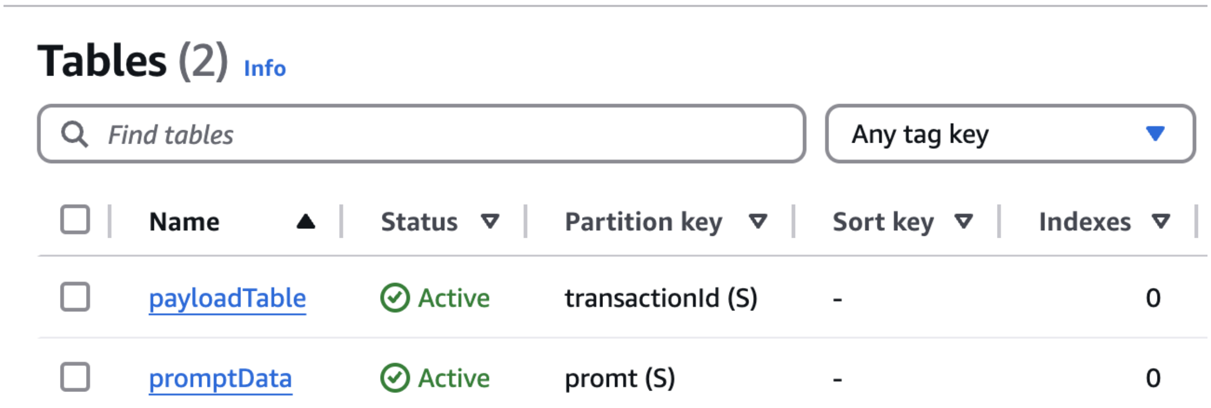
DynamoDB stream details

Capture item-level changes in your table, and push the changes to a DynamoDB stream. You then can access the change information through the DynamoDB Streams API.

Stream status	View type	Latest stream ARN
✓ On	New image	arn:aws:dynamodb:us-east-1:736908875385:table/payloadTable/stream/2024-10-25T13:37:36.618

Resource-based policy
Not active

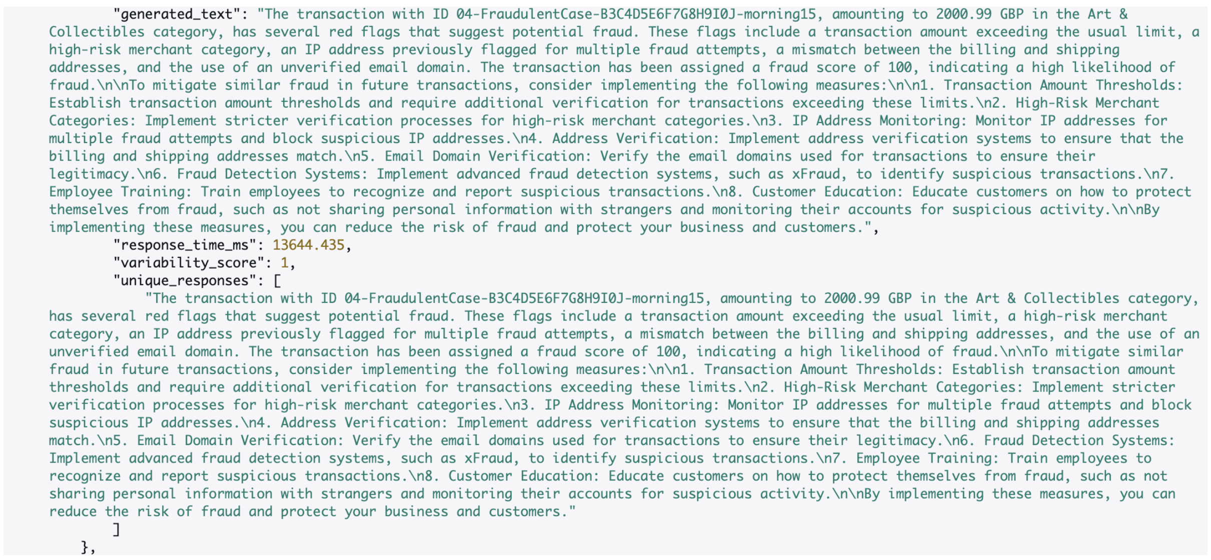
Figure 9: Enter Caption



Tables (2) Info

	Name	Status	Partition key	Sort key	Indexes
☐	payloadTable	✓ Active	transactionId (S)	-	0
☐	promptData	✓ Active	promt (S)	-	0

Figure 10: DynamoDB



```
        "generated_text": "The transaction with ID 04-FraudulentCase-B3C4D5E6F7G8H9I0J-morning15, amounting to 2000.99 GBP in the Art &
Collectibles category, has several red flags that suggest potential fraud. These flags include a transaction amount exceeding the usual limit, a
high-risk merchant category, an IP address previously flagged for multiple fraud attempts, a mismatch between the billing and shipping
addresses, and the use of an unverified email domain. The transaction has been assigned a fraud score of 100, indicating a high likelihood of
fraud.\n\nTo mitigate similar fraud in future transactions, consider implementing the following measures:\n\n1. Transaction Amount Thresholds:
Establish transaction amount thresholds and require additional verification for transactions exceeding these limits.\n2. High-Risk Merchant
Categories: Implement stricter verification processes for high-risk merchant categories.\n3. IP Address Monitoring: Monitor IP addresses for
multiple fraud attempts and block suspicious IP addresses.\n4. Address Verification: Implement address verification systems to ensure that the
billing and shipping addresses match.\n5. Email Domain Verification: Verify the email domains used for transactions to ensure their
legitimacy.\n6. Fraud Detection Systems: Implement advanced fraud detection systems, such as xFraud, to identify suspicious transactions.\n7.
Employee Training: Train employees to recognize and report suspicious transactions.\n8. Customer Education: Educate customers on how to protect
themselves from fraud, such as not sharing personal information with strangers and monitoring their accounts for suspicious activity.\n\nBy
implementing these measures, you can reduce the risk of fraud and protect your business and customers.",
        "response_time_ms": 13644.435,
        "variability_score": 1,
        "unique_responses": [
            "The transaction with ID 04-FraudulentCase-B3C4D5E6F7G8H9I0J-morning15, amounting to 2000.99 GBP in the Art & Collectibles category,
has several red flags that suggest potential fraud. These flags include a transaction amount exceeding the usual limit, a high-risk merchant
category, an IP address previously flagged for multiple fraud attempts, a mismatch between the billing and shipping addresses, and the use of an
unverified email domain. The transaction has been assigned a fraud score of 100, indicating a high likelihood of fraud.\n\nTo mitigate similar
fraud in future transactions, consider implementing the following measures:\n\n1. Transaction Amount Thresholds: Establish transaction amount
thresholds and require additional verification for transactions exceeding these limits.\n2. High-Risk Merchant Categories: Implement stricter
verification processes for high-risk merchant categories.\n3. IP Address Monitoring: Monitor IP addresses for multiple fraud attempts and block
suspicious IP addresses.\n4. Address Verification: Implement address verification systems to ensure that the billing and shipping addresses
match.\n5. Email Domain Verification: Verify the email domains used for transactions to ensure their legitimacy.\n6. Fraud Detection Systems:
Implement advanced fraud detection systems, such as xFraud, to identify suspicious transactions.\n7. Employee Training: Train employees to
recognize and report suspicious transactions.\n8. Customer Education: Educate customers on how to protect themselves from fraud, such as not
sharing personal information with strangers and monitoring their accounts for suspicious activity.\n\nBy implementing these measures, you can
reduce the risk of fraud and protect your business and customers."
        ]
    },
```

Figure 11: Response Example

cessed: 2024-12-12.

Amazon Bedrock Titan Models - User Guide (n.d.). `https://docs.aws.amazon.com/bedrock/latest/userguide/titan-models.html`. Accessed: 2024-12-12.

AWS Bedrock Knowledge Base RAG Workshop (n.d.). `https://github.com/aws-samples/bedrock-kb-rag-workshop`. Accessed: 2024-12-12.