04-11-2025 Task-2

Sri Perumbuduri D M S Satyanarayana

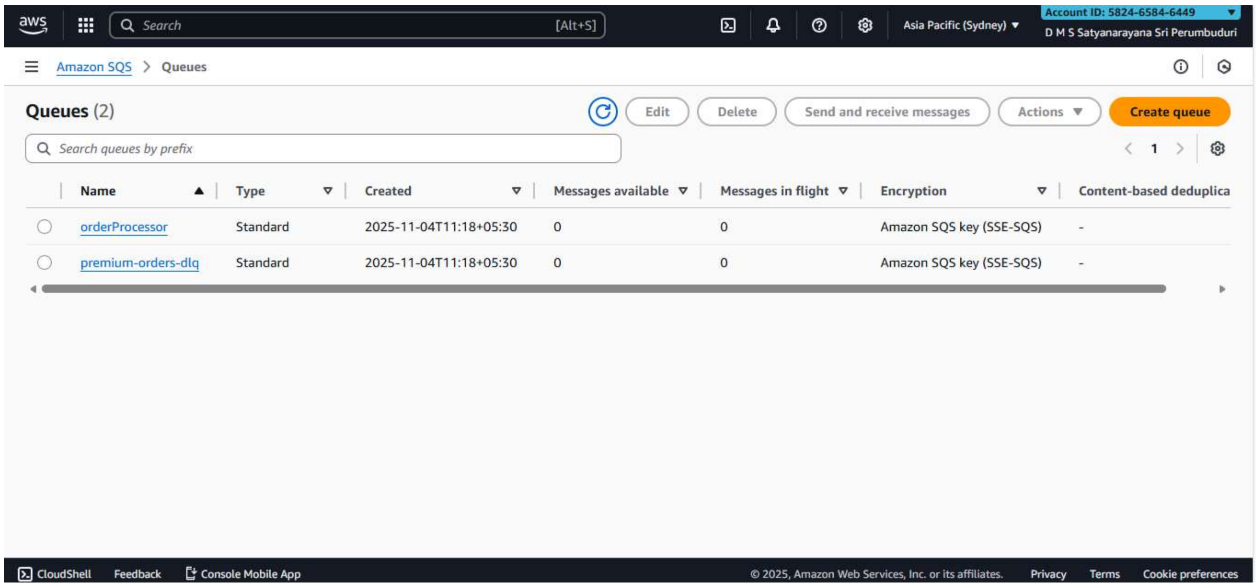
ENC\16800

Objective

Design and implement a real-time data processing pipeline for an e-commerce application that handles order events with the following requirements

Step1: Create an SQS Queue

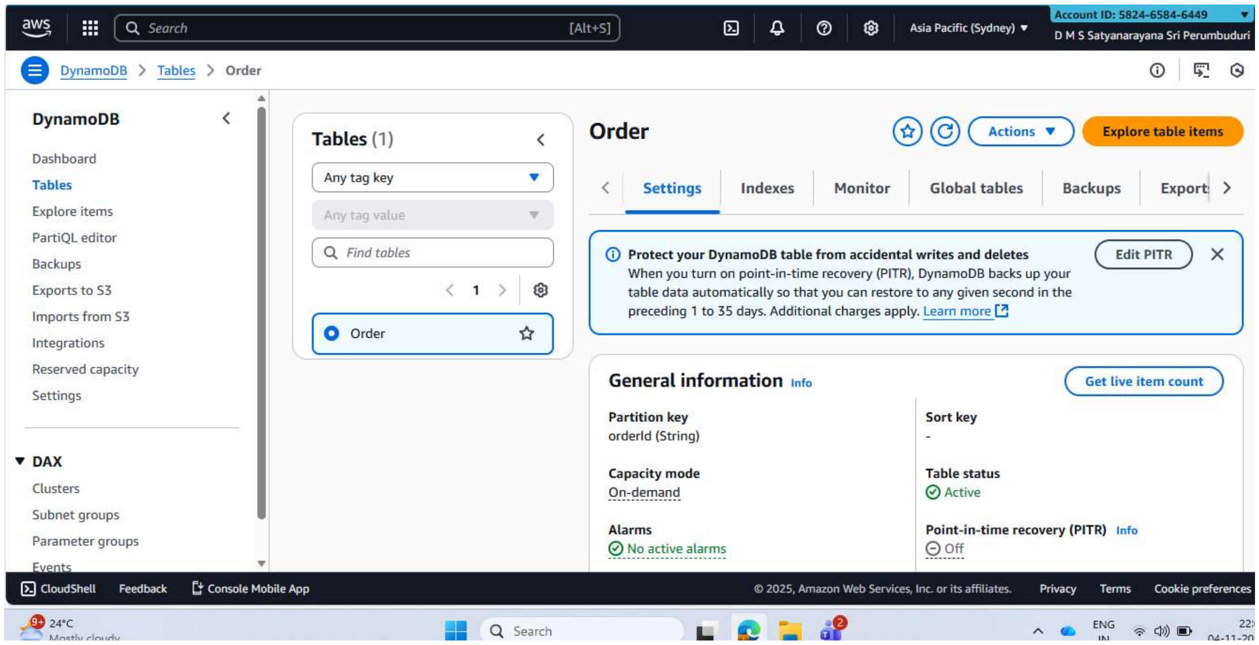
Use Amazon SQS to create a queue (standard or FIFO depending on message ordering requirements).Standard queues: high throughput, at-least-once delivery.FIFO queues: exactly-once processing, preserve message order.



Name	Type	Created	Messages available	Messages in flight	Encryption	Content-based deduplica
orderProcessor	Standard	2025-11-04T11:18+05:30	0	0	Amazon SQS key (SSE-SQS)	-
premium-orders-dlq	Standard	2025-11-04T11:18+05:30	0	0	Amazon SQS key (SSE-SQS)	-

Step2: Implement DynamoDB Table

Enable DynamoDB streams

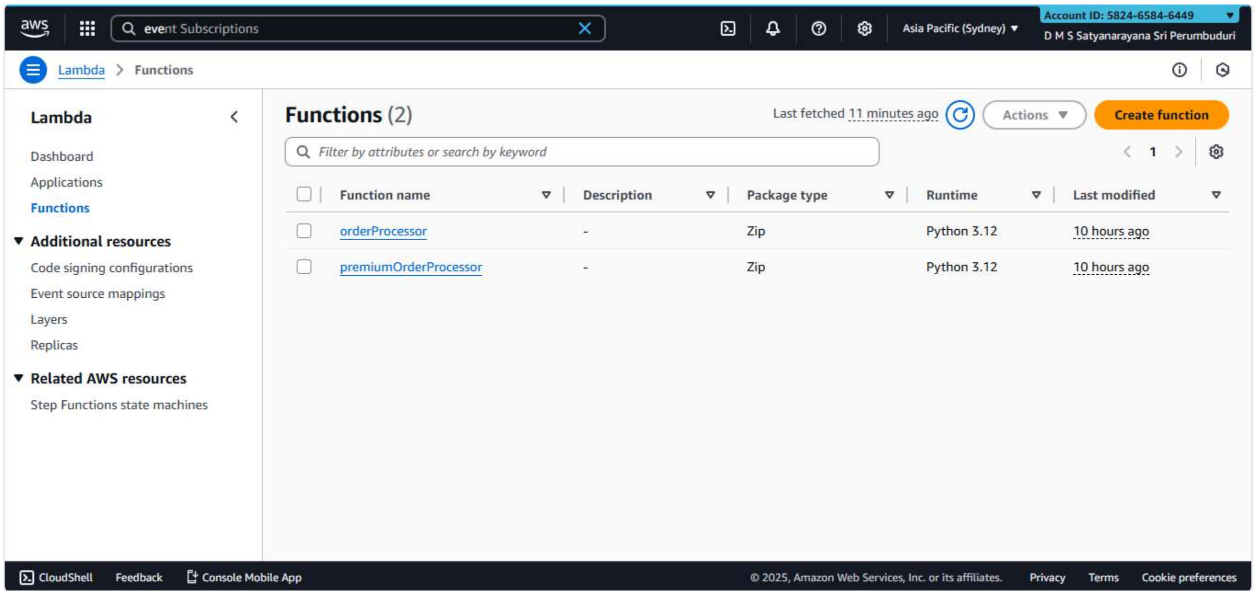

Order

Protect your DynamoDB table from accidental writes and deletes
When you turn on point-in-time recovery (PITR), DynamoDB backs up your table data automatically so that you can restore to any given second in the preceding 1 to 35 days. Additional charges apply. Learn more

General information

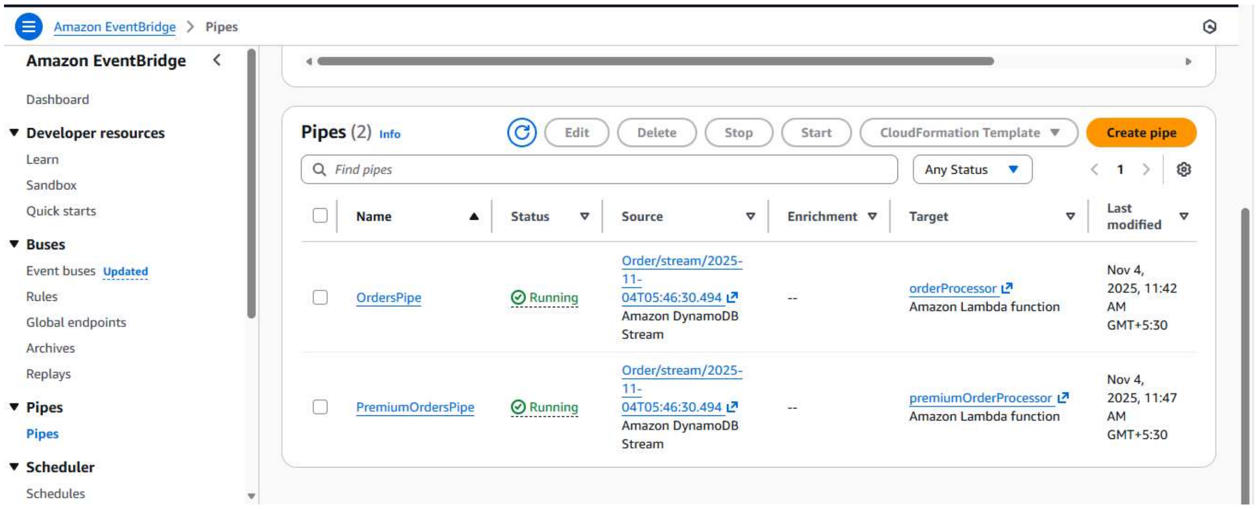
Partition key: orderId (String)	Sort key: -
Capacity mode: On-demand	Table status: Active
Alarms: No active alarms	Point-in-time recovery (PITR): Off

Step3: Lamda Function to process message body


Functions (2)

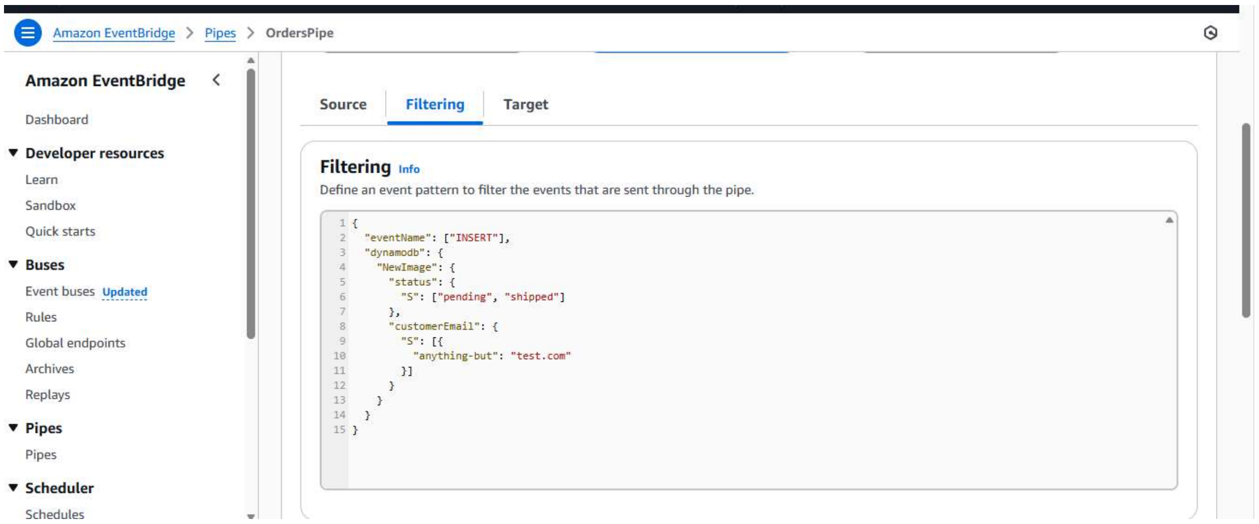
Function name	Description	Package type	Runtime	Last modified
orderProcessor	-	Zip	Python 3.12	10 hours ago
premiumOrderProcessor	-	Zip	Python 3.12	10 hours ago

Step4 : Create Event Bridge Pipes



Name	Status	Source	Enrichment	Target	Last modified
OrdersPipe	Running	Order/stream/2025-11-04T05:46:30.494 Amazon DynamoDB Stream	--	orderProcessor Amazon Lambda function	Nov 4, 2025, 11:42 AM GMT+5:30
PremiumOrdersPipe	Running	Order/stream/2025-11-04T05:46:30.494 Amazon DynamoDB Stream	--	premiumOrderProcessor Amazon Lambda function	Nov 4, 2025, 11:47 AM GMT+5:30

Filters in Event Bridge pipes

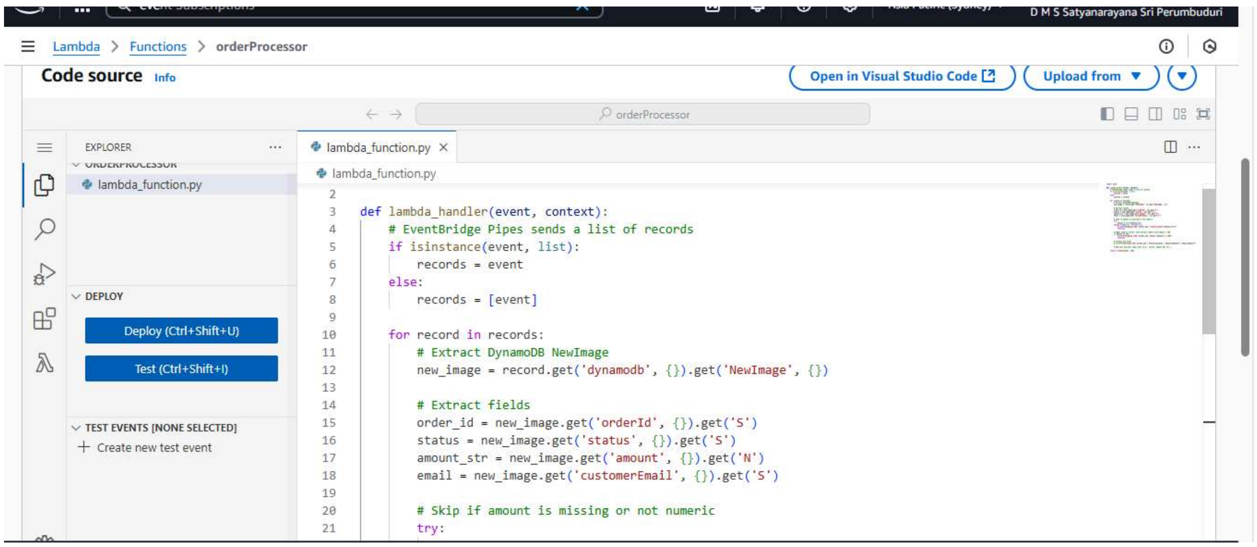


Filtering Info

Define an event pattern to filter the events that are sent through the pipe.

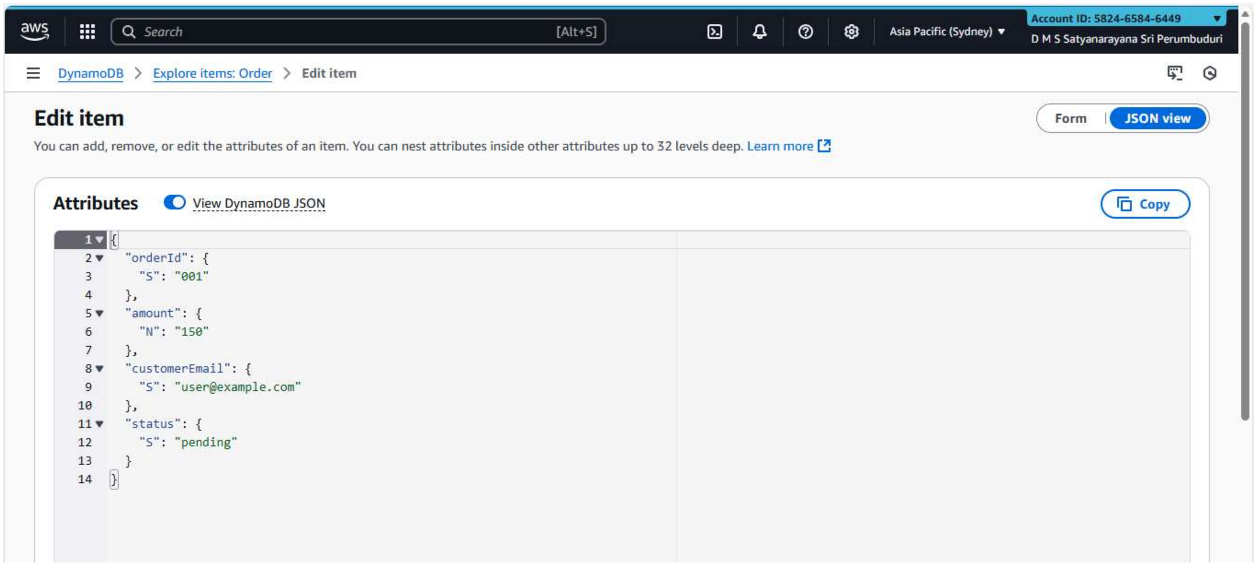
```
{
  "eventName": ["INSERT"],
  "dynamodb": {
    "NewImage": {
      "status": {
        "S": ["pending", "shipped"]
      },
      "customerEmail": {
        "S": [{
          "anything-but": "test.com"
        }]
      }
    }
  }
}
```

Lamda Function for processing messages


D M S Satyanarayana Sri Perumbuduri
Lambda > Functions > orderProcessor
Code source Info
Open in Visual Studio Code
Upload from
EXPLORER
lambda_function.py
DEPLOY
Deploy (Ctrl+Shift+U)
Test (Ctrl+Shift+I)
TEST EVENTS [NONE SELECTED]
Create new test event

```python

def lambda_handler(event, context):
    # EventBridge Pipes sends a list of records
    if isinstance(event, list):
        records = event
    else:
        records = [event]

    for record in records:
        # Extract DynamoDB NewImage
        new_image = record.get('dynamodb', {}).get('NewImage', {})

        # Extract fields
        order_id = new_image.get('orderId', {}).get('S')
        status = new_image.get('status', {}).get('S')
        amount_str = new_image.get('amount', {}).get('N')
        email = new_image.get('customerEmail', {}).get('S')

        # Skip if amount is missing or not numeric
        try:
```

Table contents


aws
Search [Alt+S]
Asia Pacific (Sydney)
Account ID: 5824-6584-6449
D M S Satyanarayana Sri Perumbuduri
DynamoDB > Explore items: Order > Edit item
Form
JSON view
Copy

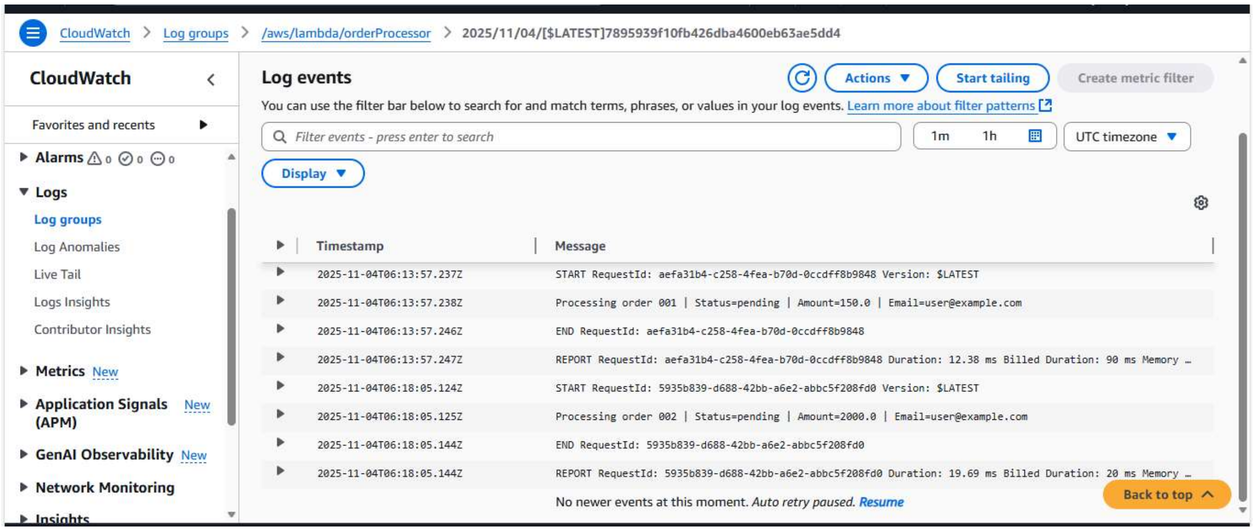
Edit item

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. Learn more

Attributes — View DynamoDB JSON

```json
{
  "orderId": {
    "S": "001"
  },
  "amount": {
    "N": "150"
  },
  "customerEmail": {
    "S": "user@example.com"
  },
  "status": {
    "S": "pending"
  }
}
```

Cloud Watch Logs



You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                  Message
2025-11-04T06:13:57.237Z   START RequestId: aefa31b4-c258-4fea-b70d-0ccdff8b9848 Version: $LATEST
2025-11-04T06:13:57.238Z   Processing order 001 | Status=pending | Amount=150.0 | Email=user@example.com
2025-11-04T06:13:57.246Z   END RequestId: aefa31b4-c258-4fea-b70d-0ccdff8b9848
2025-11-04T06:13:57.247Z   REPORT RequestId: aefa31b4-c258-4fea-b70d-0ccdff8b9848 Duration: 12.38 ms Billed Duration: 90 ms Memory …
2025-11-04T06:18:05.124Z   START RequestId: 5935b839-d688-42bb-a6e2-abbc5f208fd0 Version: $LATEST
2025-11-04T06:18:05.125Z   Processing order 002 | Status=pending | Amount=2000.0 | Email=user@example.com
2025-11-04T06:18:05.144Z   END RequestId: 5935b839-d688-42bb-a6e2-abbc5f208fd0
2025-11-04T06:18:05.144Z   REPORT RequestId: 5935b839-d688-42bb-a6e2-abbc5f208fd0 Duration: 19.69 ms Billed Duration: 20 ms Memory …
```

No newer events at this moment. *Auto retry paused.* Resume