Notify

Real-Time Personalised Notification and Recommendation System

Objective:

To deliver highly personalised, optimised notifications using real-time events and machine learning.

Team Members:

- Pranshul
- Sejal
- Kedar
- Shambhavi
- Supriya

Problem Statement

- Users receive generic and poorly timed notifications
- No real-time personalisation
- Rule-based targeting does not scale
- No feedback-driven learning loop

Goal:

Build a **scalable, ML-driven, real-time recommendation and notification system** with optimised delivery.

Frontend & Event Ingestion

User interacts via Web/App frontend

Events generated:

- Orders
- Clicks
- Campaign views

Kafka Topics:

- Orders Topic
- Events Topic

Stored in:

- purchase_history
- notify_clickstream

Batch Ingestion

Target Tables

- **notify_purchase_history** – Historical transactions
- **notify_user_features** – Enriched user profiles & behaviour
- **notify_products** – Core product catalogue
- **notify_product_features** – Product performance metrics
- **notify_campaigns** – Campaign metadata
- **notify_user_product_matrix** – User–product interaction matrix

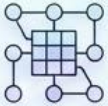
Tech Stack:

Amazon DynamoDB

Why DynamoDB:

- Low-latency at scale
- Serverless auto-scaling

Core & Feature Table Schemas



1. notify_user_product_matrix (Behavior Matrix)

user_id (PK)
product_id
last_purchase_date
purchased_before
times_purchased

Purpose: Captures user–product interaction strength for recommendation and similarity.

2. notify_purchase_history (Transactions)

purchase_id
user_id (PK)
product_id
purchase_date
price
quantity
total_amount

Purpose: Stores raw historical transactions for analytics and embedding generation.

3. notify_campaigns (Campaign Metadata)

campaign_id (PK)
created_at
name
description
notification_type
priority
start_date, end_date
status
template_short, template_long

Purpose: Defines all notification campaigns and delivery content.

4. notify_products (Core Product Catalog)

product_id (PK)
articleType
baseColour
gender
image_url
masterCategory
price
productDisplayName
season
subCategory
usage
year

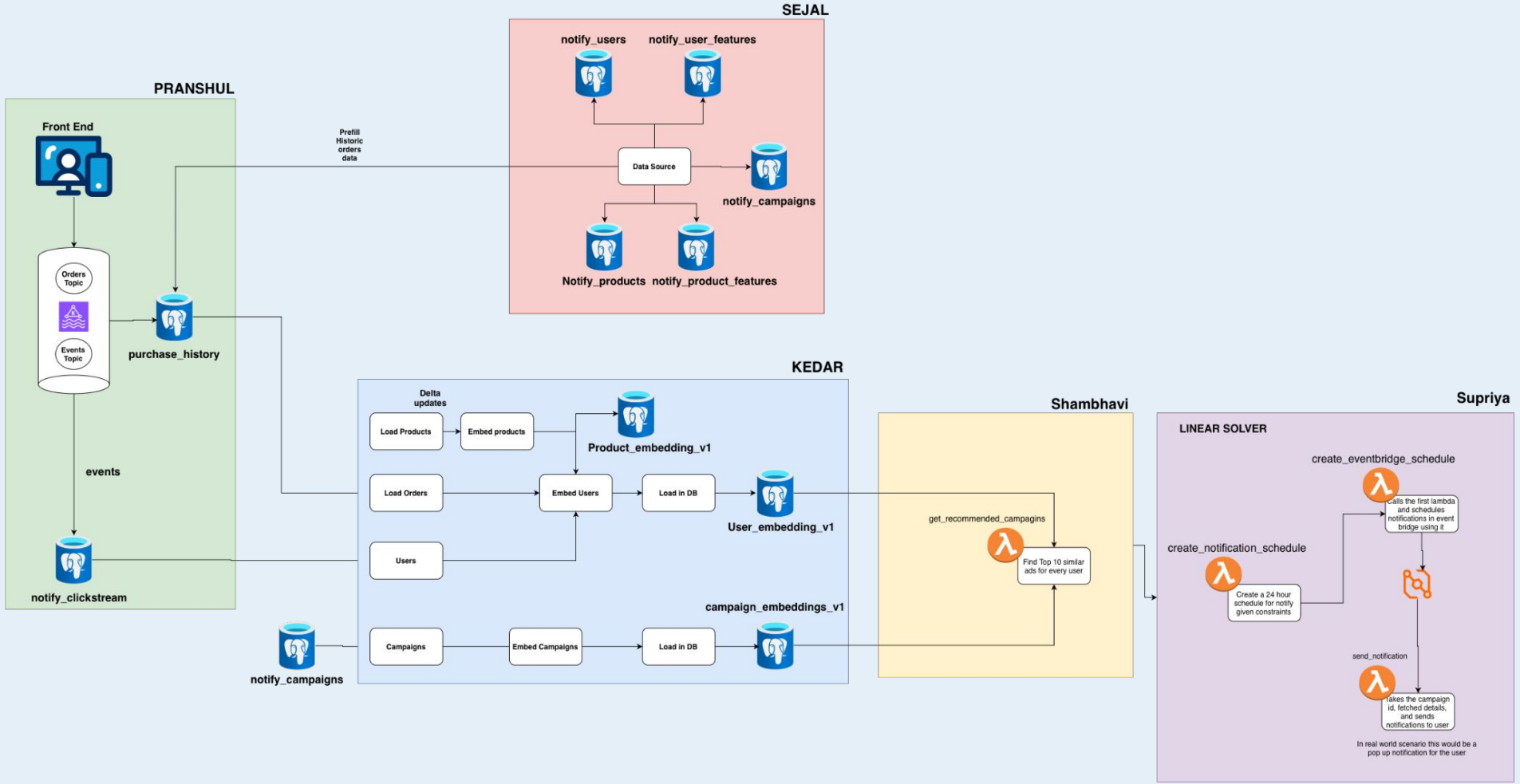
Purpose: Master product reference table for the system.

5. notify_product_features (Product Feature Store)

product_id (PK)
articleType
avg_price
baseColour
gender
image_url
last_sold_date
masterCategory
price
productDisplayName
revenue
season
subCategory
total_sales
usage
year

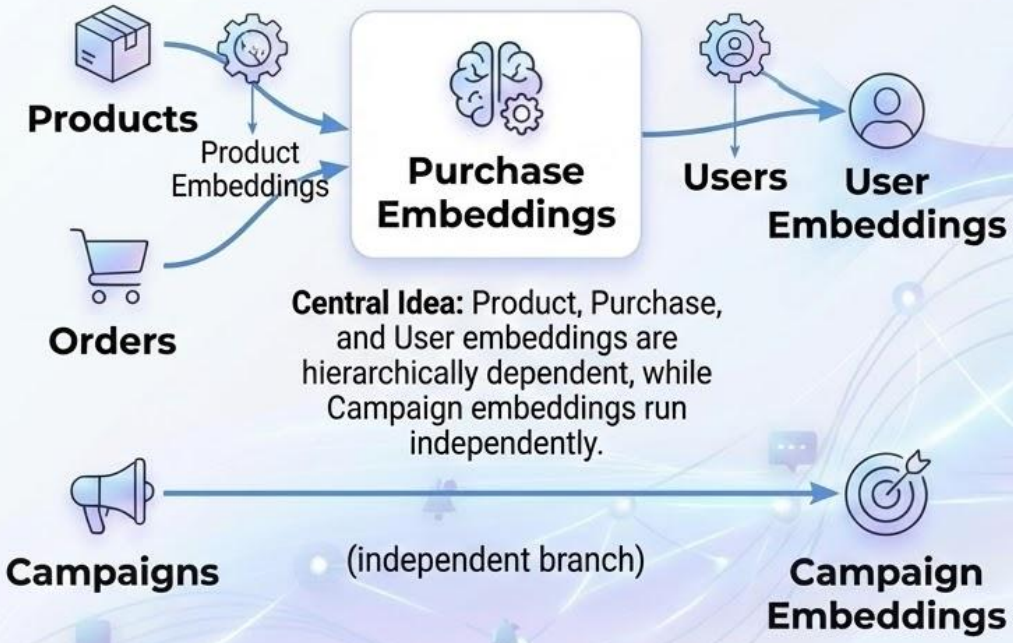
Purpose: Stores derived product performance features used for embeddings and targeting.

High-Level System Overview



ML Pipeline Orchestration in Airflow

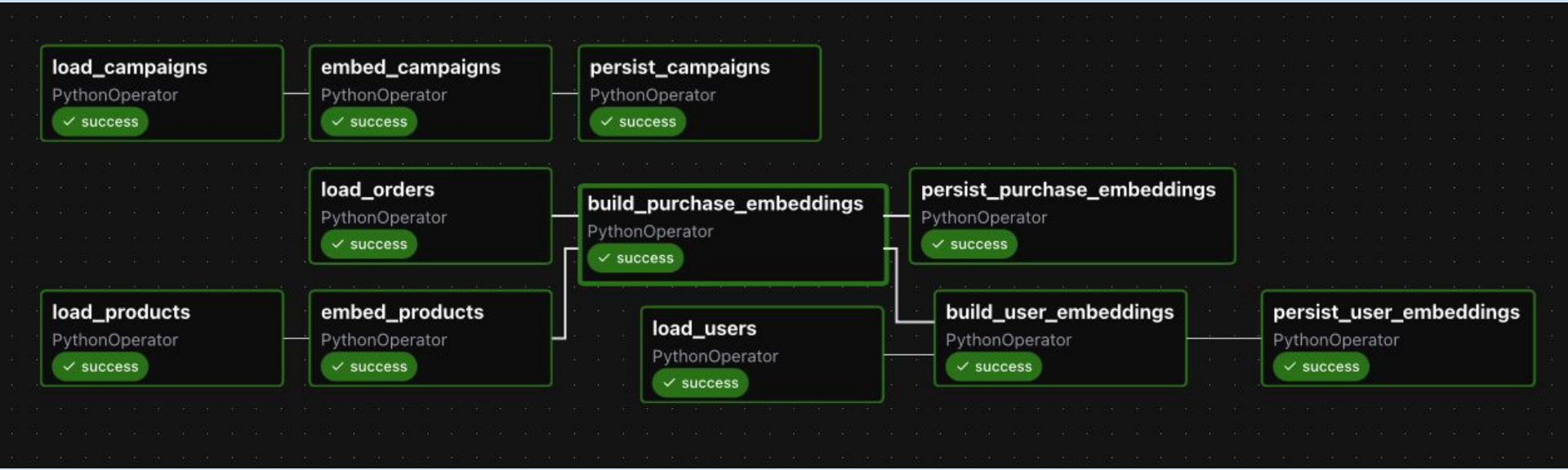
Flow Logic

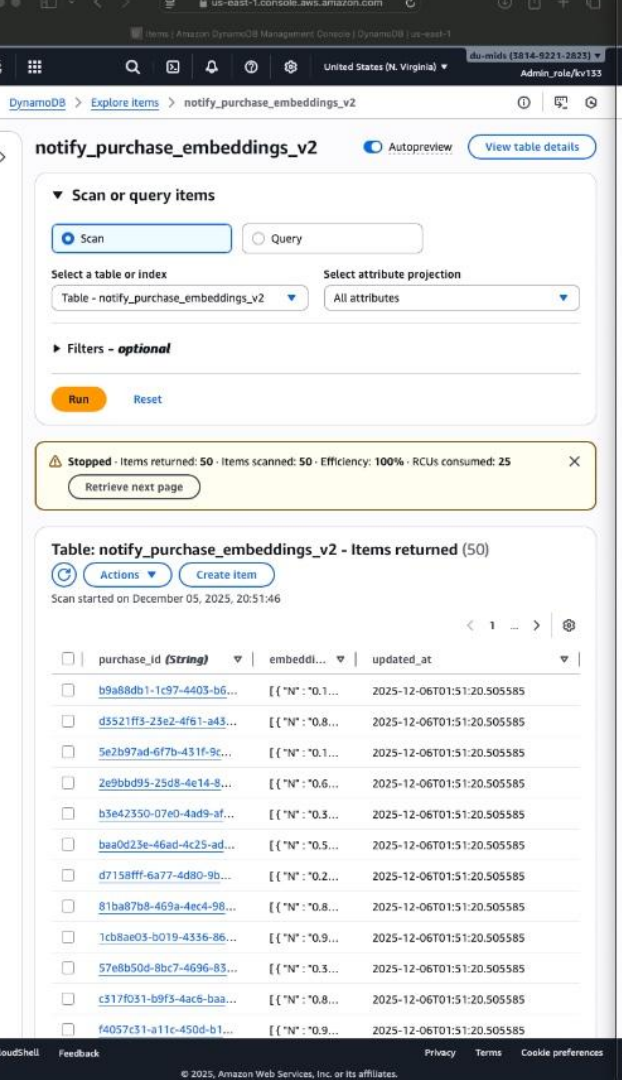


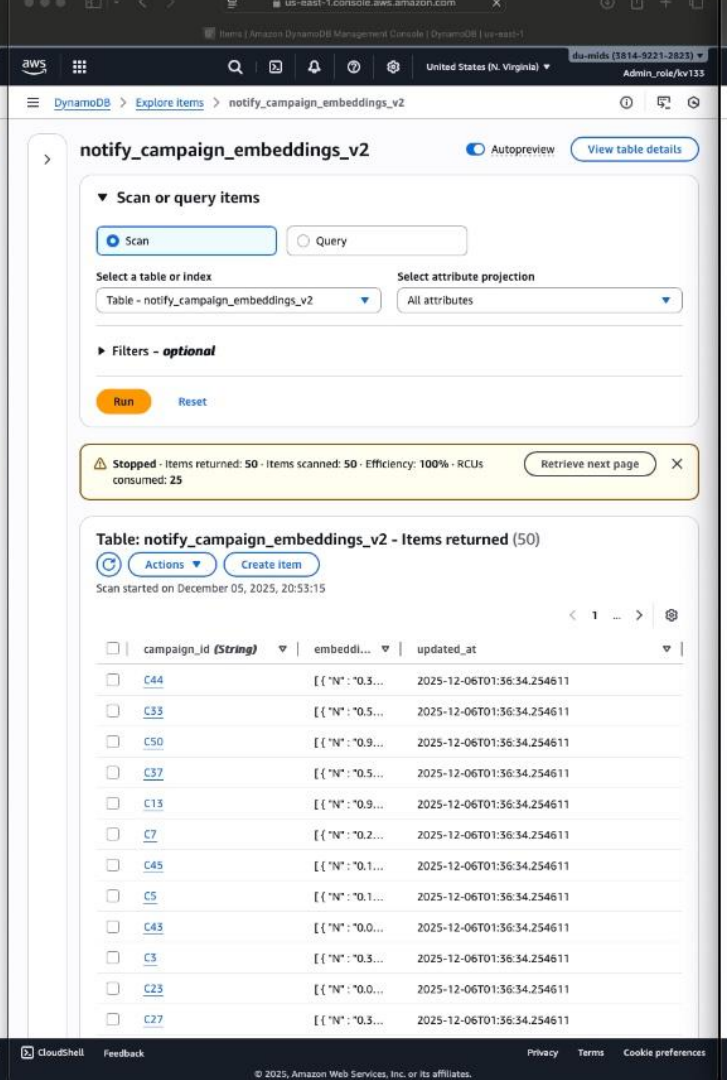
Execution Design

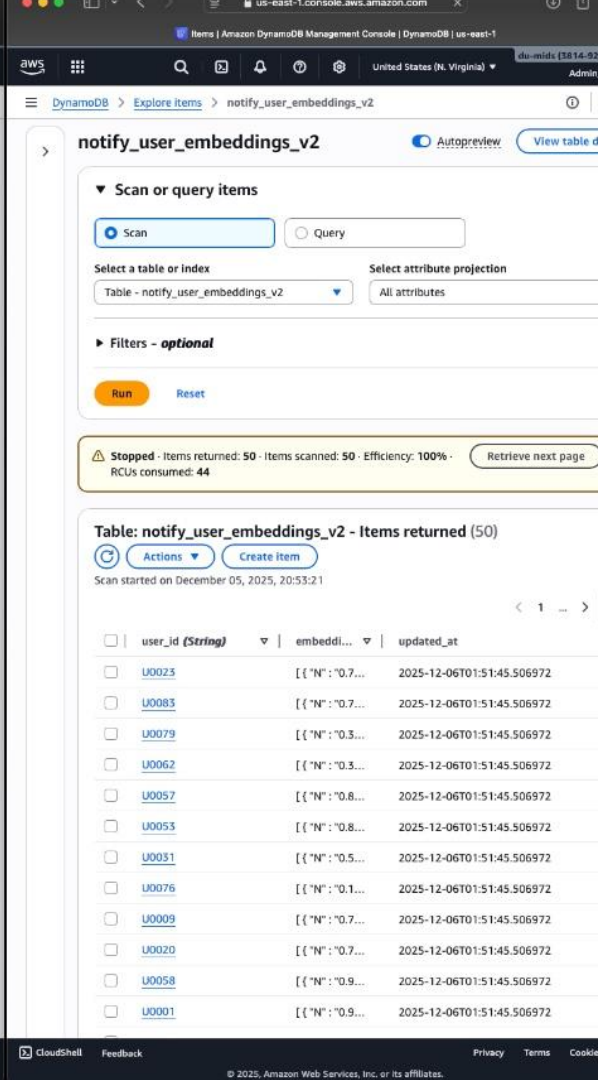
- Product and order pipelines converge at **Purchase Embeddings**
- User Embeddings are built downstream of **Purchase Embeddings**
- Product Embeddings are a first-class output
- Campaign embeddings run in **parallel**
- Ensures temporal consistency and maximum pipeline parallelism

Live System Validation: AWS MWAA

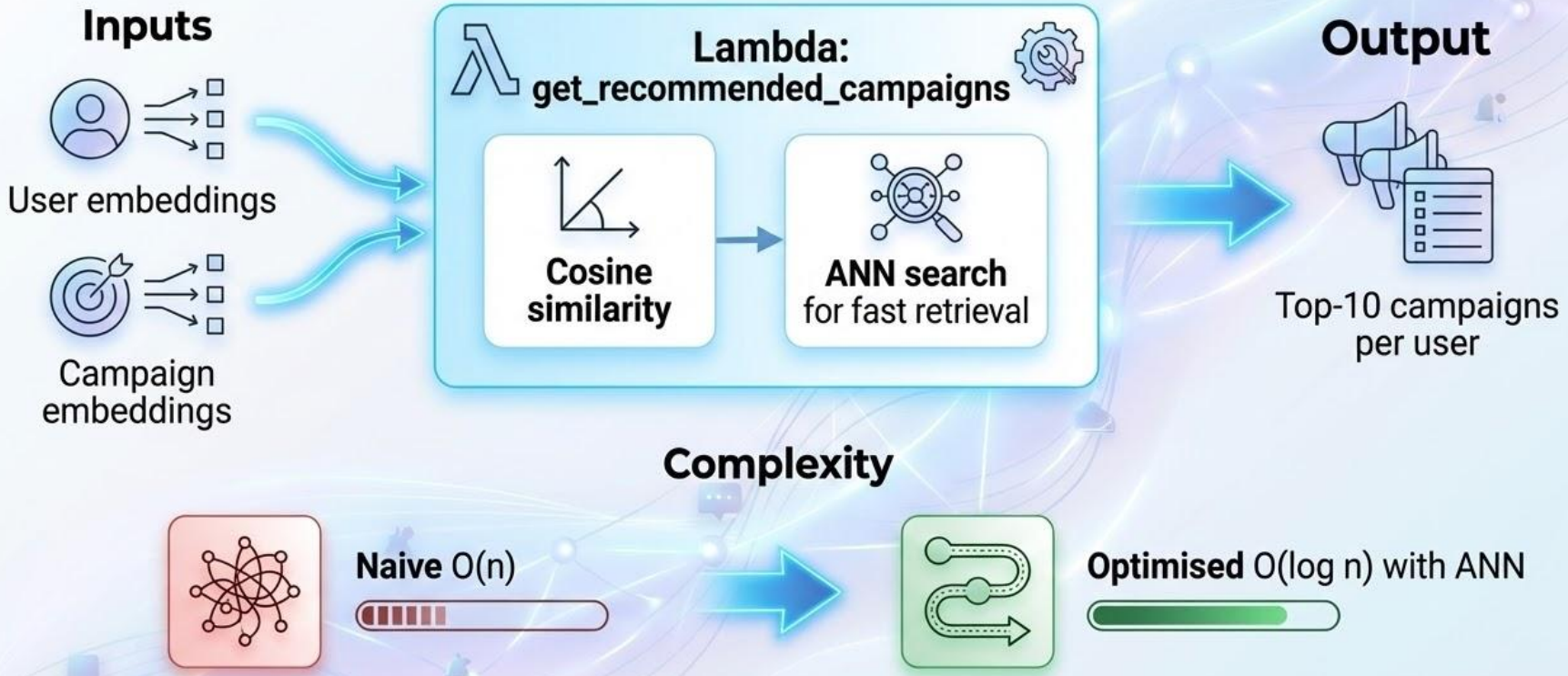


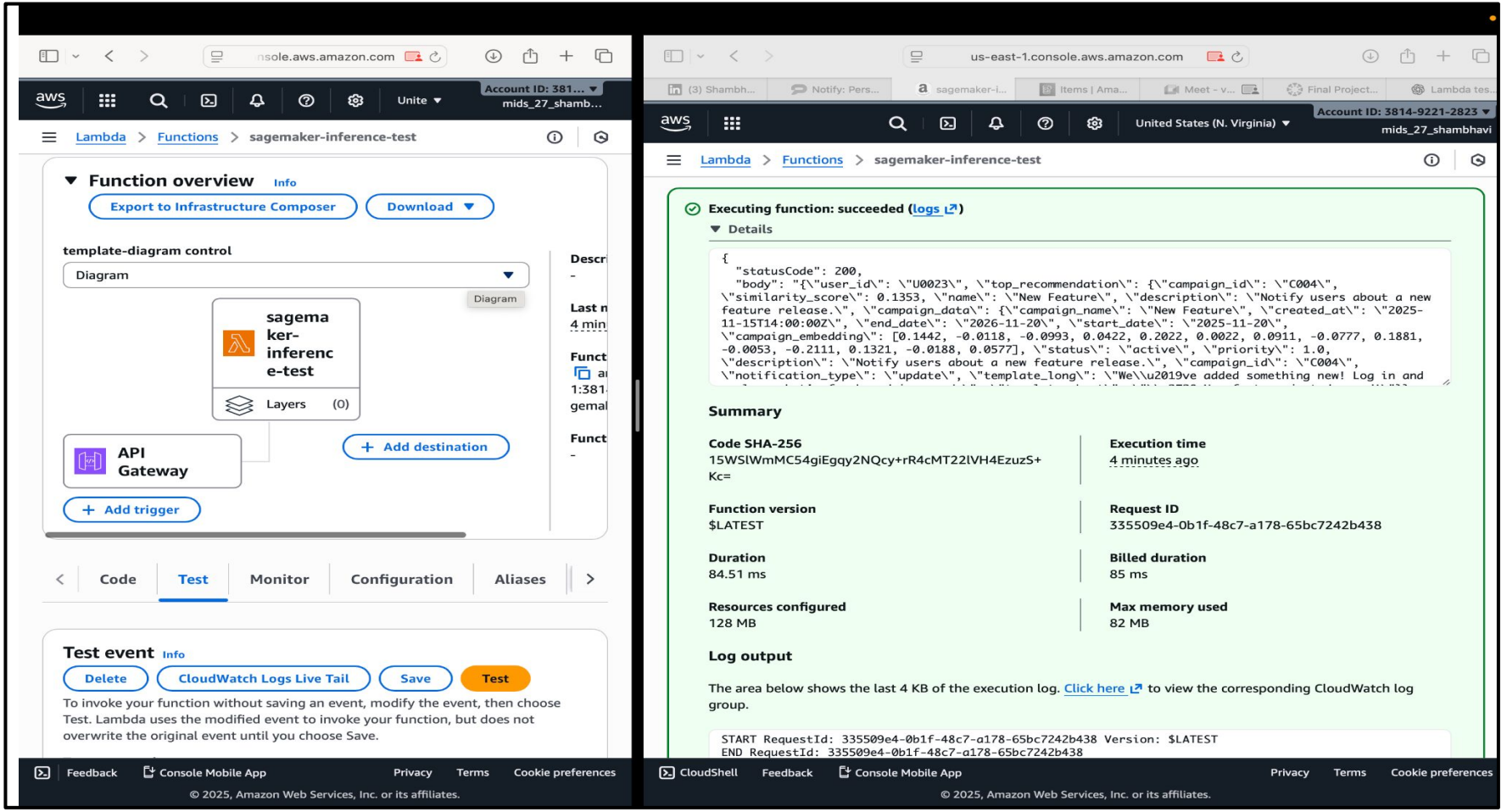

DynamoDB > Explore items > notify_purchase_embeddings_v2
notify_purchase_embeddings_v2
Autopreview
View table details
Scan or query items
Scan
Query
Select a table or index
Table - notify_purchase_embeddings_v2
Select attribute projection
All attributes
Filters - optional
Run
Reset
Stopped · Items returned: 50 · Items scanned: 50 · Efficiency: 100% · RCUs consumed: 25
Retrieve next page
Table: notify_purchase_embeddings_v2 - Items returned (50)
Actions
Create item
Scan started on December 05, 2025, 20:51:46
purchase_id (String)
embeddi...
updated_at
b9a88db1-1c97-4403-b6...
d3521ff3-23e2-4f61-a43...
5e2b97ad-6f7b-431f-9c...
2e9bbd95-25d8-4e14-8...
b3e42350-07e0-4ad9-af...
baa0d23e-46ad-4c25-ad...
d7158fff-6a77-4d80-9b...
81ba87b8-469a-4ec4-98...
1cb8ae03-b019-4336-86...
57e8b50d-8bc7-4696-83...
c317f031-b9f3-4ac6-baa...
f4057c31-a11c-450d-b1...
2025-12-06T01:51:20.505585
CloudShell
Feedback
Privacy
Terms
Cookie preferences
© 2025, Amazon Web Services, Inc. or its affiliates.


DynamoDB > Explore items > notify_campaign_embeddings_v2
notify_campaign_embeddings_v2
Autopreview
View table details
Scan or query items
Scan
Query
Select a table or index
Table - notify_campaign_embeddings_v2
Select attribute projection
All attributes
Filters - optional
Run
Reset
Stopped · Items returned: 50 · Items scanned: 50 · Efficiency: 100% · RCUs consumed: 25
Retrieve next page
Table: notify_campaign_embeddings_v2 - Items returned (50)
Actions
Create item
Scan started on December 05, 2025, 20:53:15
campaign_id (String)
embeddi...
updated_at
C44
C33
C50
C37
C13
C7
C45
C5
C43
C3
C23
C27
2025-12-06T01:36:34.254611
CloudShell
Feedback
Privacy
Terms
Cookie preferences
© 2025, Amazon Web Services, Inc. or its affiliates.


DynamoDB > Explore items > notify_user_embeddings_v2
notify_user_embeddings_v2
Autopreview
Scan or query items
Scan
Query
Select a table or index
Table - notify_user_embeddings_v2
Select attribute projection
All attributes
Filters - optional
Run
Reset
Stopped · Items returned: 50 · Items scanned: 50 · Efficiency: 100% · RCUs consumed: 44
Retrieve next page
Table: notify_user_embeddings_v2 - Items returned (50)
Actions
Create item
Scan started on December 05, 2025, 20:53:21
user_id (String)
embeddi...
updated_at
U0023
U0083
U0079
U0062
U0057
U0053
U0031
U0076
U0009
U0020
U0058
U0001
2025-12-06T01:51:45.506972
CloudShell
Feedback
Privacy
Terms
© 2025, Amazon Web Services, Inc. or its affiliates.

Real-Time Recommendation Engine




Lambda > Functions > sagemaker-inference-test
Function overview Info
Export to Infrastructure Composer
Download
template-diagram control
Diagram
sagemaker-inference-test
Layers (0)
API Gateway
Add destination
Add trigger
Code
Test
Monitor
Configuration
Aliases
Test event Info
Delete
CloudWatch Logs Live Tail
Save
Test
To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save.
Feedback
Console Mobile App
Privacy
Terms
Cookie preferences
© 2025, Amazon Web Services, Inc. or its affiliates.
us-east-1.console.aws.amazon.com
United States (N. Virginia)
Account ID: 3814-9221-2823
mids_27_shambhavi
Lambda > Functions > sagemaker-inference-test
Executing function: succeeded (logs)
Details
{
"statusCode": 200,
"body": "{\"user_id\": \"U0023\", \"top_recommendation\": {\"campaign_id\": \"C004\", \"similarity_score\": 0.1353, \"name\": \"New Feature\", \"description\": \"Notify users about a new feature release.\", \"campaign_data\": {\"campaign_name\": \"New Feature\", \"created_at\": \"2025-11-15T14:00:00Z\", \"end_date\": \"2026-11-20\", \"start_date\": \"2025-11-20\", \"campaign_embedding\": [0.1442, -0.0118, -0.0993, 0.0422, 0.2022, 0.0022, 0.0911, -0.0777, 0.1881, -0.0053, -0.2111, 0.1321, -0.0188, 0.0577], \"status\": \"active\", \"priority\": 1.0, \"description\": \"Notify users about a new feature release.\", \"campaign_id\": \"C004\", \"notification_type\": \"update\", \"template_long\": \"We\\u2019ve added something new! Log in and
Summary
Code SHA-256
15WSlWmMC54giEgqy2NQcy+rR4cMT22lVH4EzuzS+Kc=
Execution time
4 minutes ago
Function version
$LATEST
Request ID
335509e4-0b1f-48c7-a178-65bc7242b438
Duration
84.51 ms
Billed duration
85 ms
Resources configured
128 MB
Max memory used
82 MB
Log output
The area below shows the last 4 KB of the execution log. Click here to view the corresponding CloudWatch log group.
START RequestId: 335509e4-0b1f-48c7-a178-65bc7242b438 Version: $LATEST
END RequestId: 335509e4-0b1f-48c7-a178-65bc7242b438
CloudShell
Feedback
Console Mobile App
Privacy
Terms
Cookie preferences
© 2025, Amazon Web Services, Inc. or its affiliates.

Scheduling & Optimisation Layer

Key Lambdas

`create_eventbridge_schedule`

`create_notification_schedule`

Constraints Handled

User fatigue

Campaign priority

Time windows

Engagement optimisation

Output

24-hour optimised notification plan.

Notification Delivery System

Lambda Function

`send_notification`

`send_notification` campaign

Delivery Process

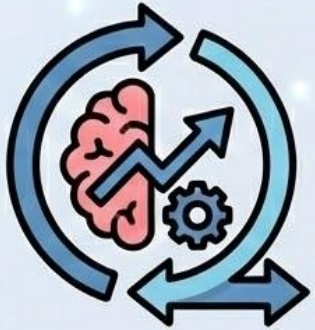
Pulls scheduled campaign

Sends notification

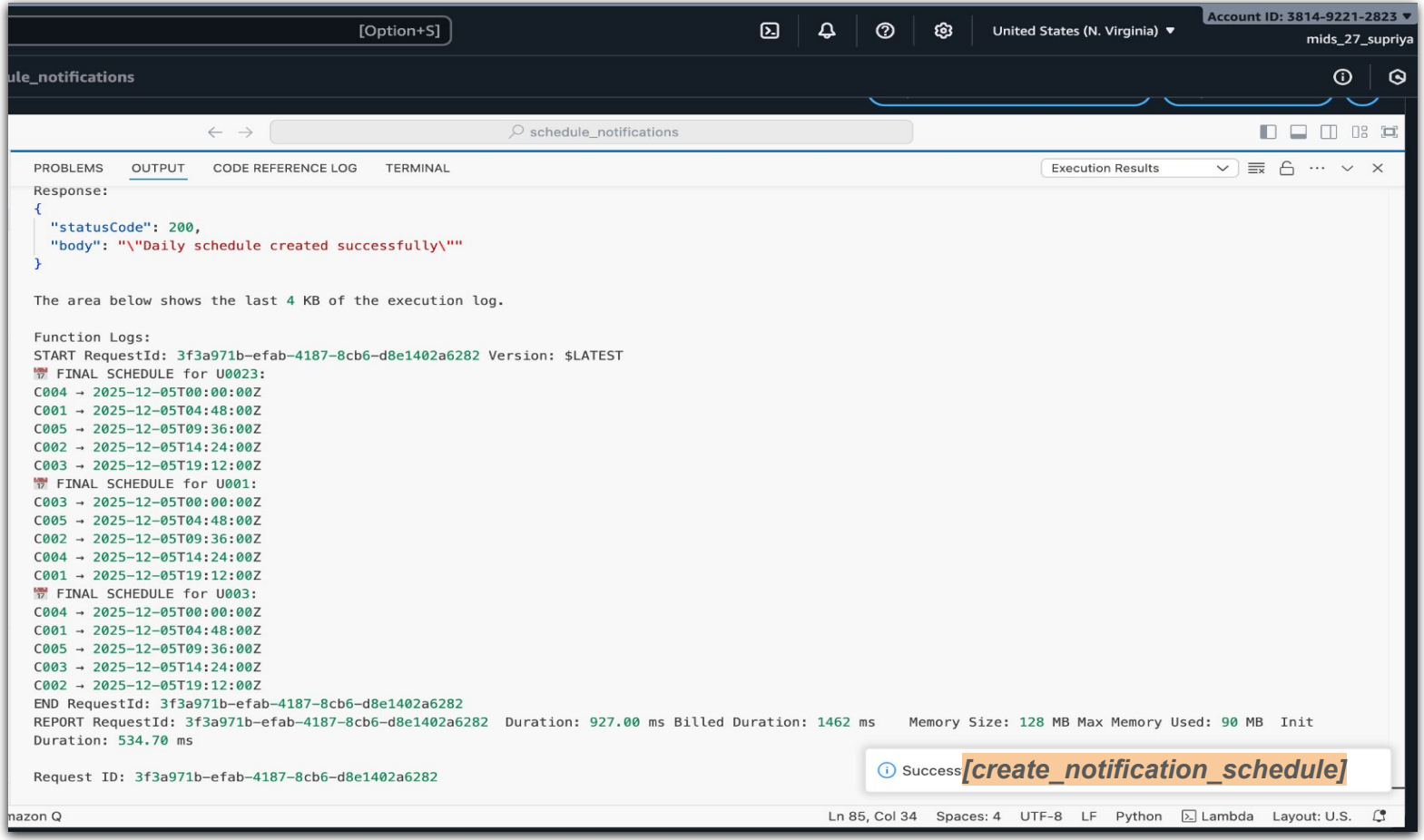
Fetches user context

User interaction flows back to clickstream

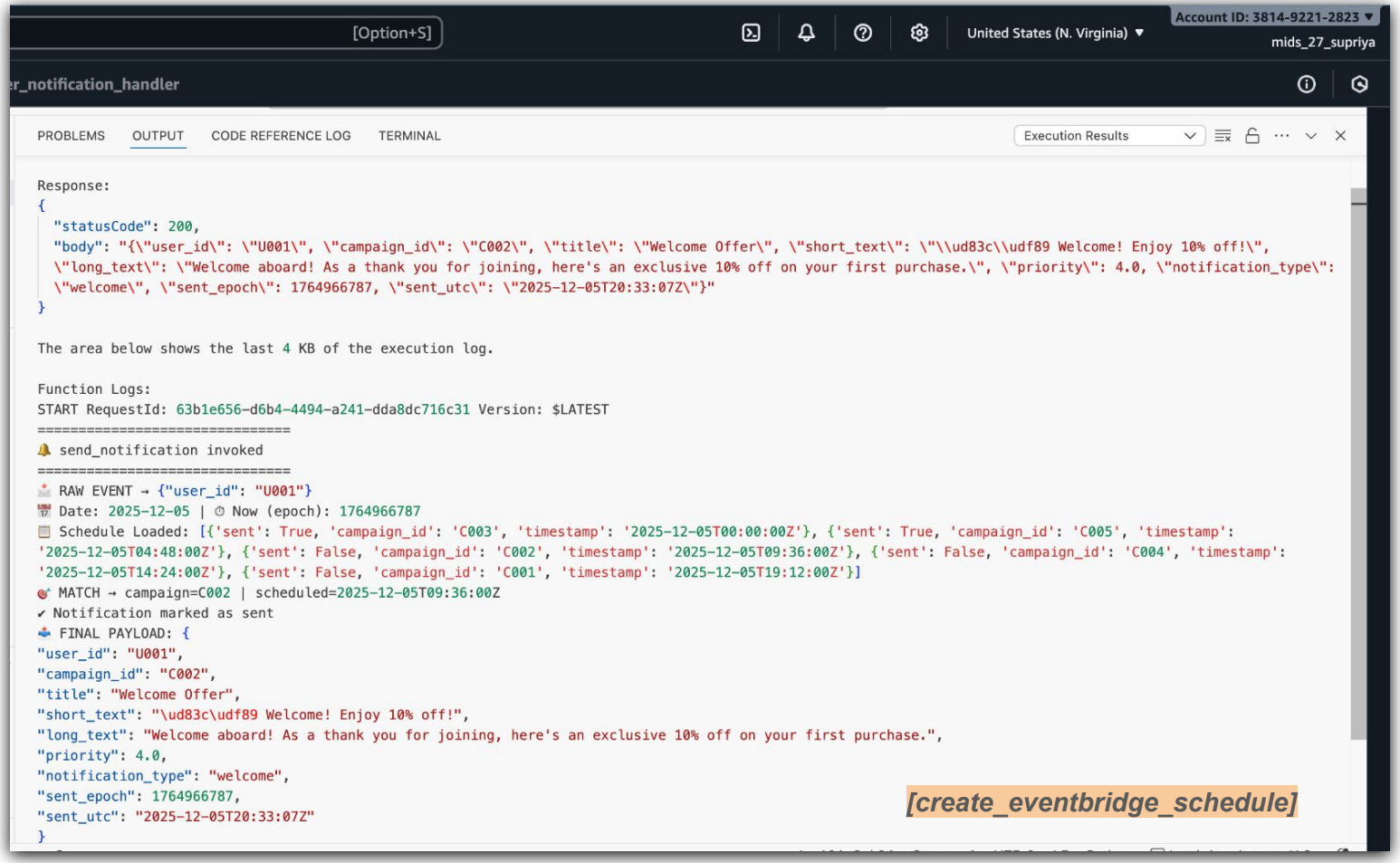
Result



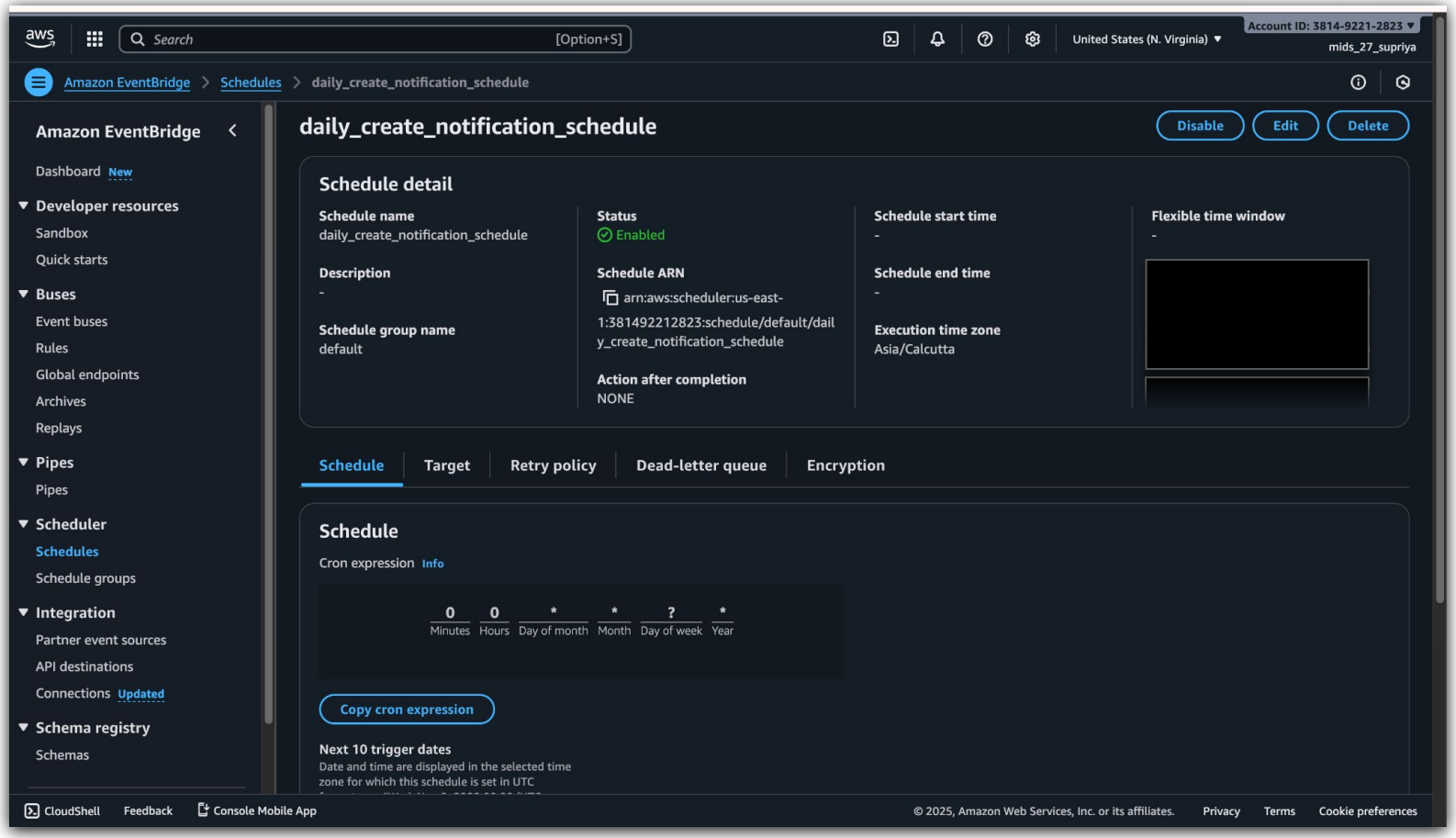
Closed-loop learning system.



Daily schedule successfully generated with timestamps — capped at 5 notifications per user.



Lambda identifies the next pending notification and prepares the delivery payload.



EventBridge automation triggers notification delivery every minute — fully serverless and hands-free.

Technology Stack

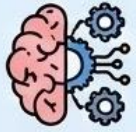
Frontend
Python

Real Time Data
Kafka

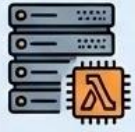
Batch Ingestion
Glue

Databases
DynamoDB

Orchestration
Amazon Managed Airflow



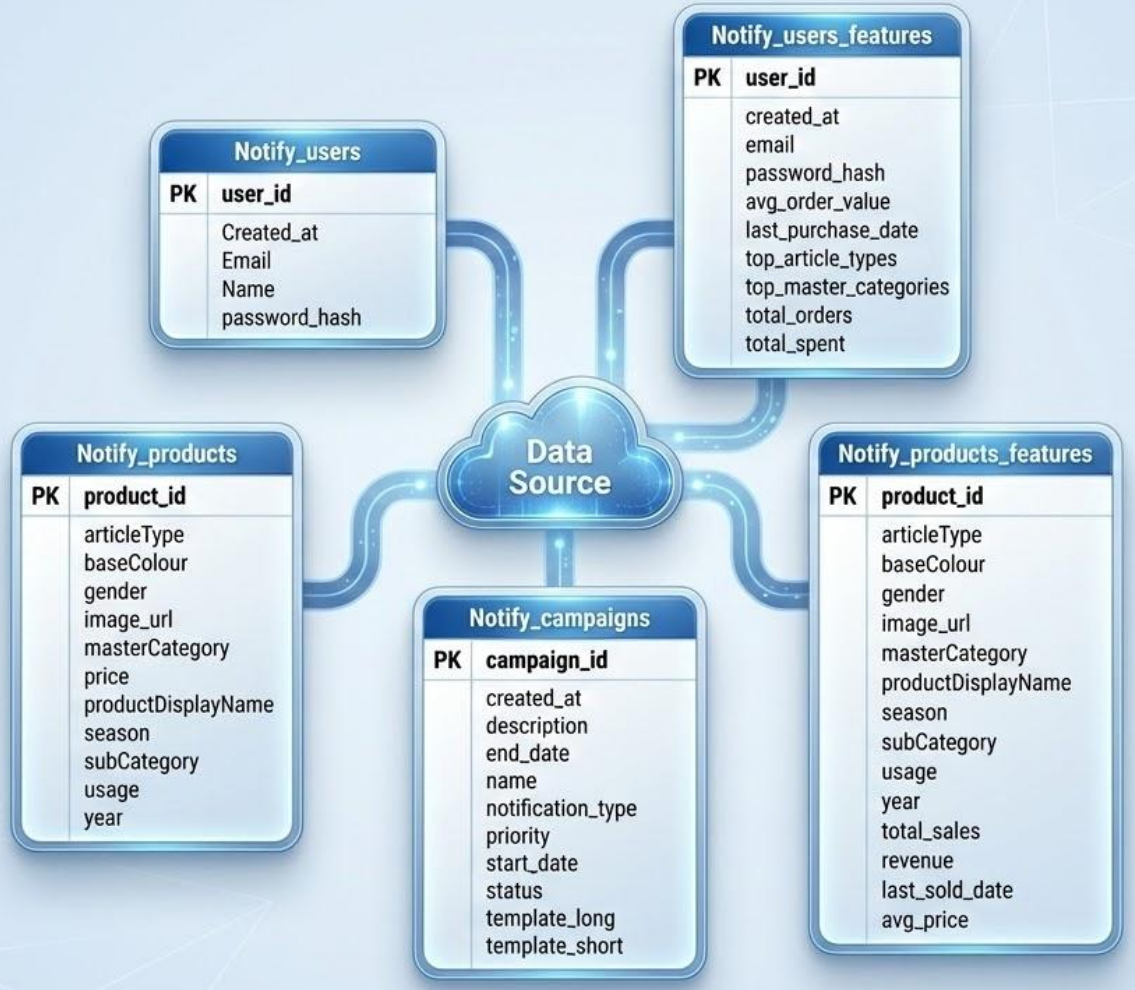
ML
BERT Embeddings, ANN Search



Compute
Lambda, EC2

Scheduling
EventBridge

THANK YOU!


Notify_users_features
PK
user_id
created_at
email
password_hash
avg_order_value
last_purchase_date
top_article_types
top_master_categories
total_orders
total_spent
Notify_users
PK
user_id
Created_at
Email
Name
password_hash
Data Source
Notify_products
PK
product_id
articleType
baseColour
gender
image_url
masterCategory
price
productDisplayName
season
subCategory
usage
year
Notify_products_features
PK
product_id
articleType
baseColour
gender
image_url
masterCategory
productDisplayName
season
subCategory
usage
year
total_sales
revenue
last_sold_date
avg_price
Notify_campaigns
PK
campaign_id
created_at
description
end_date
name
notification_type
priority
start_date
status
template_long
template_short

Batch Ingestion

`Target Tables`

- `purchase_history` – Historical orders
- `notify_users` – User profiles
- `notify_products` – Product catalogue
- `notify_campaigns` – Campaign metadata

Tech Stack:

- **Amazon DynamoDB**

Why DynamoDB:

- **Low-latency at scale**
- **Serverless auto-scaling**

Outcome:
Stable historical backbone for all ML pipelines.

Core & Feature Table Schemas

1. `notify_user_product_matrix` **(Behavior Matrix)**

- `user_id` (PK)
- `product_id`
- `last_purchase_date`
- `purchased_before`
- `times_purchased`

Purpose: Captures **user–product interaction strength** for recommendation and similarity.

2. `notify_purchase_history` **(Transactions)**

- `purchase_id`
- `user_id` (PK)
- `product_id`
- `purchase_date`
- `price`
- `quantity`
- `total_amount`

Purpose: Stores **raw historical transactions** for analytics and embedding generation.

3. `notify_campaigns` **(Campaign Metadata)**

- `campaign_id` (PK)
- `created_at`
- `name`
- `description`
- `notification_type`
- `priority`
- `start_date, end_date`
- `status`
- `template_short, template_long`

Purpose: Defines all **notification campaigns and delivery content**.

4. `notify_products` **(Core Product Catalog)**

- `product_id` (PK)
- `articleType`
- `baseColour`
- `gender`
- `image_url`
- `masterCategory`
- `price`
- `productDisplayName`
- `season`
- `subCategory`
- `usage`
- `year`

Batch Ingestion

Target Tables

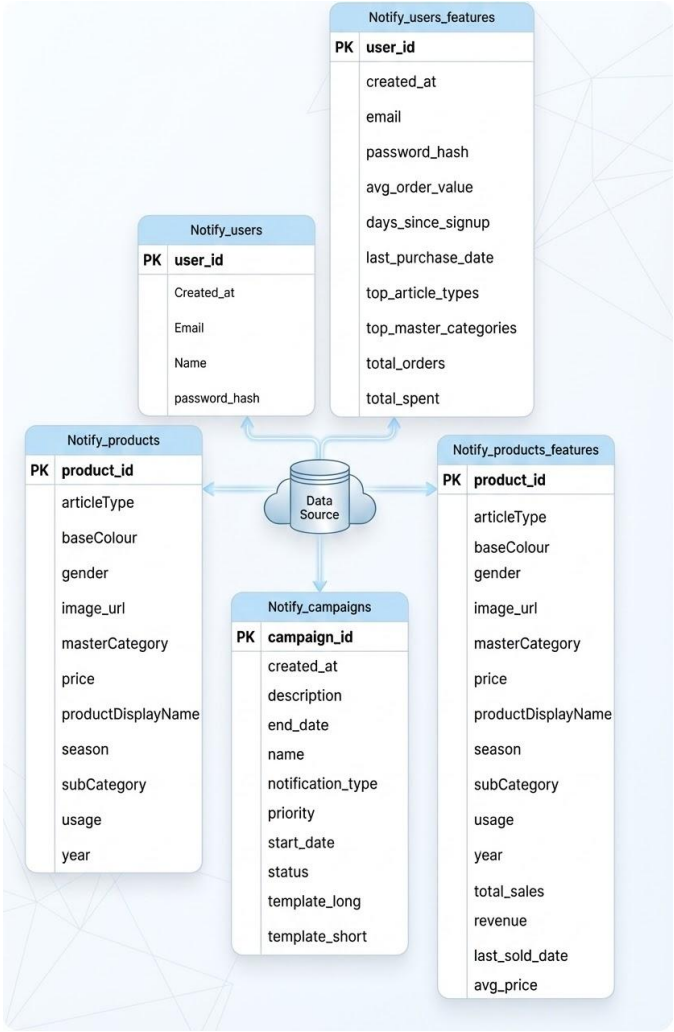
🧾 **notify_purchase_history** – Historical transactions
👤 **notify_user_features** – Enriched user profiles & behaviour
🛍️ **notify_products** – Core product catalogue
📊 **notify_product_features** – Product performance metrics
📣 **notify_campaigns** – Campaign metadata
🔗 **notify_user_product_matrix** – User–product interaction matrix

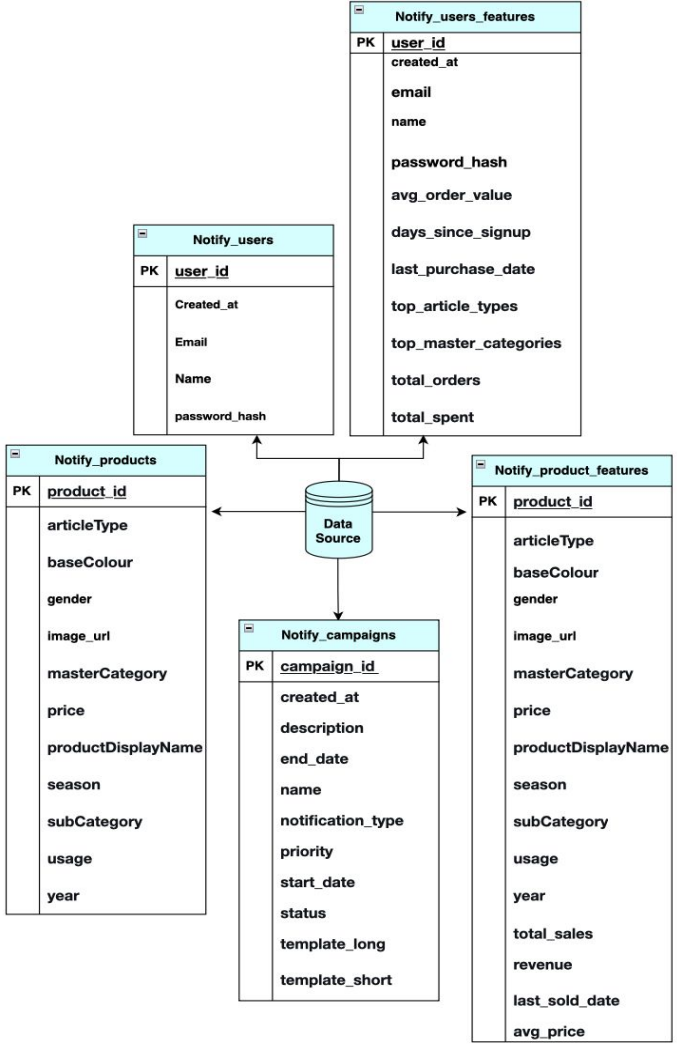
Tech Stack

Amazon DynamoDB

Why DynamoDB

⏱️ Low-latency at scale
☁️ Serverless auto-scaling


Notify_users_features
PK
user_id
created_at
email
password_hash
avg_order_value
days_since_signup
last_purchase_date
top_article_types
top_master_categories
total_orders
total_spent
Notify_users
PK
user_id
Created_at
Email
Name
password_hash
Notify_products
PK
product_id
articleType
baseColour
gender
image_url
masterCategory
price
productDisplayName
season
subCategory
usage
year
Data Source
Notify_products_features
PK
product_id
articleType
baseColour
gender
image_url
masterCategory
price
productDisplayName
season
subCategory
usage
year
total_sales
revenue
last_sold_date
avg_price
Notify_campaigns
PK
campaign_id
created_at
description
end_date
name
notification_type
priority
start_date
status
template_long
template_short


Notify_users_features
PK
user_id
created_at
email
name
password_hash
avg_order_value
days_since_signup
last_purchase_date
top_article_types
top_master_categories
total_orders
total_spent
Notify_users
PK
user_id
Created_at
Email
Name
password_hash
Notify_products
PK
product_id
articleType
baseColour
gender
image_url
masterCategory
price
productDisplayName
season
subCategory
usage
year
Data Source
Notify_campaigns
PK
campaign_id
created_at
description
end_date
name
notification_type
priority
start_date
status
template_long
template_short
Notify_product_features
PK
product_id
articleType
baseColour
gender
image_url
masterCategory
price
productDisplayName
season
subCategory
usage
year
total_sales
revenue
last_sold_date
avg_price

Notification Delivery System

Lambda: `send_notification`

- Pulls scheduled campaign
- Fetches user context
- Sends notification
- User interaction flows back to clickstream

Result: Closed-loop learning system.

✅ Notify_users (Core User Table)

Primary Key: `user_id`

Fields:

- `created_at` – account creation timestamp
- `email` – user email
- `name` – user name
- `password_hash` – encrypted password

Purpose:
Stores **core authentication and identity information** for each user.

✅ Notify_users_features (User Feature Store)

Primary Key: `user_id`

Fields:

- `created_at`
- `email`
- `password_hash`
- `avg_order_value`
- `days_since_signup`
- `last_purchase_date`
- `top_article_types`
- `top_master_categories`
- `total_orders`
- `total_spent`

Purpose:
Stores **derived behavioural features** used for:

- User profiling
- Embedding generation
- Recommendation logic

✅ Notify_products (Core Product Table)

Primary Key: `product_id`

Fields:

Scheduling & Optimisation Layer

Key Lambdas:

- `create_eventbridge_schedule`
- `create_notification_schedule`

Constraints handled:

- User fatigue
- Time windows
- Campaign priority
- Engagement optimisation

Output: 24-hour optimised notification plan.

Real-Time Recommendation Engine

Lambda: `get_recommended_campaigns`

- Inputs:
  - User embeddings
  - Campaign embeddings
- Processing:
  - Cosine similarity
  - ANN search for fast retrieval
- Output:
  - Top-10 campaigns per user

Complexity:
Naive `O(n)` → Optimised `O(log n)` with ANN

Technology Stack

Frontend: Python

Real Time Data: Kafka

Batch Ingestion: Glue

Databases: DynamoDB

Orchestration: Amazon Managed Airflow

ML: BERT Embeddings, ANN Search

Compute: Lambda, EC2

Scheduling: EventBridge

ML Pipeline Orchestration in Airflow

Central Idea

Product, Purchase, and User embeddings are **hierarchically dependent**, while Campaign embeddings run independently.

Flow Logic

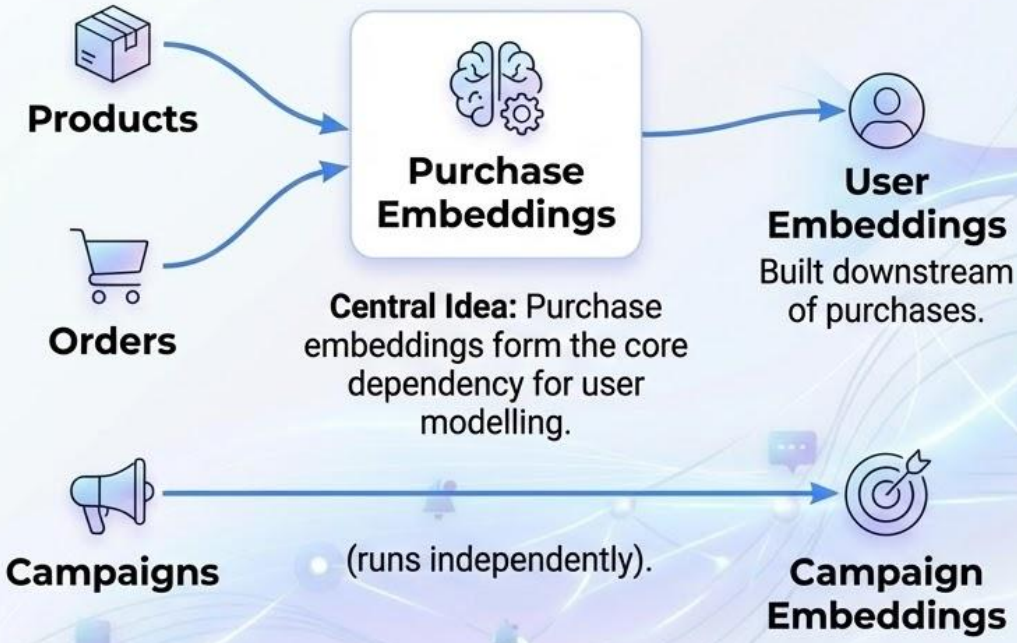
- Products → **Product Embeddings**
- Orders + Product Embeddings → **Purchase Embeddings**
- Users + Purchase Embeddings → **User Embeddings**
- Campaigns → **Campaign Embeddings** (independent branch)

Execution Design

- Product and order pipelines converge at **Purchase Embeddings**
- **User Embeddings are built downstream of Purchase Embeddings**
- **Product Embeddings are a first-class output**
- Campaign embeddings run in **parallel**
- Ensures **temporal consistency and maximum pipeline parallelism**

ML Pipeline Orchestration in Airflow

Flow Logic



Execution Design

- Product and order pipelines converge at purchase embeddings
- User embeddings are built downstream of purchases
- Campaign embeddings run in parallel
- Ensures temporal consistency and maximum pipeline parallelism

Frontend & Event Ingestion

User interacts via Web/App frontend

Events generated:

- Orders
- Clicks
- Campaign views

Kafka Topics:

- Orders Topic
- Events Topic

Stored in:

- `purchase_history`
- `notify_clickstream`

Objective:
To deliver highly personalised, optimised notifications using real-time events and machine learning.

Notify

Real-Time Personalised Notification and Recommendation System

Team Members:
Pranshul, Sejal, Kedar, Shambhavi, Supriya

Problem Statement

Users receive generic and poorly timed notifications

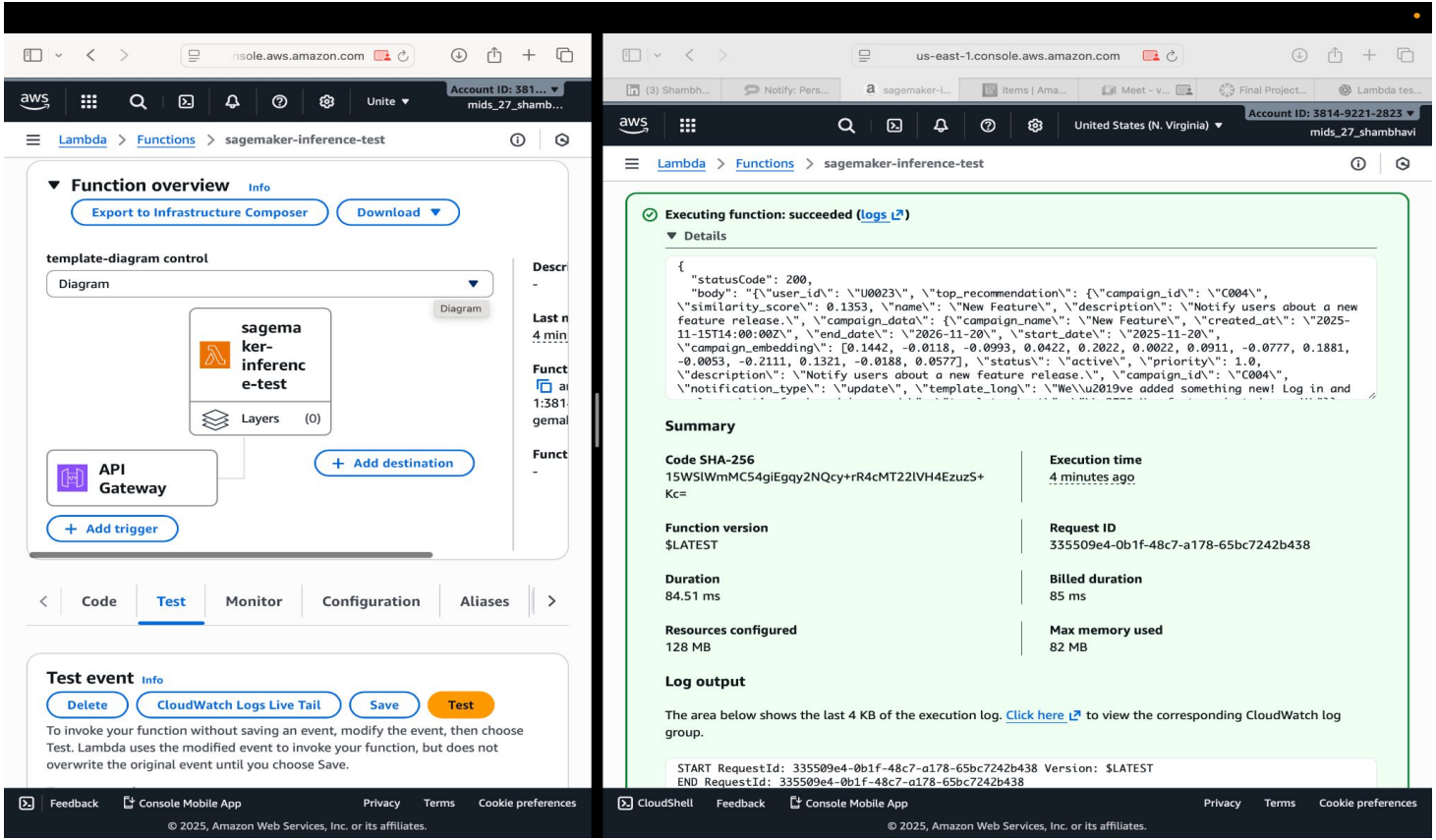
No real-time personalisation

Rule-based targeting does not scale

No feedback-driven learning loop

Goal:

Build a **scalable, ML-driven, real-time recommendation and notification system** with optimised delivery.


Lambda > Functions > sagemaker-inference-test
Function overview Info
Export to Infrastructure Composer
Download
template-diagram control
Diagram
sagemaker-inferenc e-test
Layers (0)
API Gateway
Add destination
Add trigger
Code
Test
Monitor
Configuration
Aliases
Test event Info
Delete
CloudWatch Logs Live Tail
Save
Test
To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save.
Feedback
Console Mobile App
Privacy
Terms
Cookie preferences
© 2025, Amazon Web Services, Inc. or its affiliates.
Executing function: succeeded (logs)
Details
{
"statusCode": 200,
"body": "{\"user_id\": \"U0023\", \"top_recommendation\": {\"campaign_id\": \"C004\", \"similarity_score\": 0.1353, \"name\": \"New Feature\", \"description\": \"Notify users about a new feature release.\", \"campaign_data\": {\"campaign_name\": \"New Feature\", \"created_at\": \"2025-11-15T14:00:00Z\", \"end_date\": \"2026-11-20\", \"start_date\": \"2025-11-20\", \"campaign_embedding\": [0.1442, -0.0118, -0.0993, 0.0422, 0.2022, 0.0022, 0.0911, -0.0777, 0.1881, -0.0053, -0.2111, 0.1321, -0.0188, 0.0577], \"status\": \"active\", \"priority\": 1.0, \"description\": \"Notify users about a new feature release.\", \"campaign_id\": \"C004\", \"notification_type\": \"update\", \"template_long\": \"We\\u2019ve added something new! Log in and
Summary
Code SHA-256
15WSlWmMC54giEgqy2NQcy+rR4cMT22lVH4EzuzS+Kc=
Execution time
4 minutes ago
Function version
$LATEST
Request ID
335509e4-0b1f-48c7-a178-65bc7242b438
Duration
84.51 ms
Billed duration
85 ms
Resources configured
128 MB
Max memory used
82 MB
Log output
The area below shows the last 4 KB of the execution log. Click here to view the corresponding CloudWatch log group.
START RequestId: 335509e4-0b1f-48c7-a178-65bc7242b438 Version: $LATEST
END RequestId: 335509e4-0b1f-48c7-a178-65bc7242b438
CloudShell
Feedback
Console Mobile App
Privacy
Terms
Cookie preferences
© 2025, Amazon Web Services, Inc. or its affiliates.