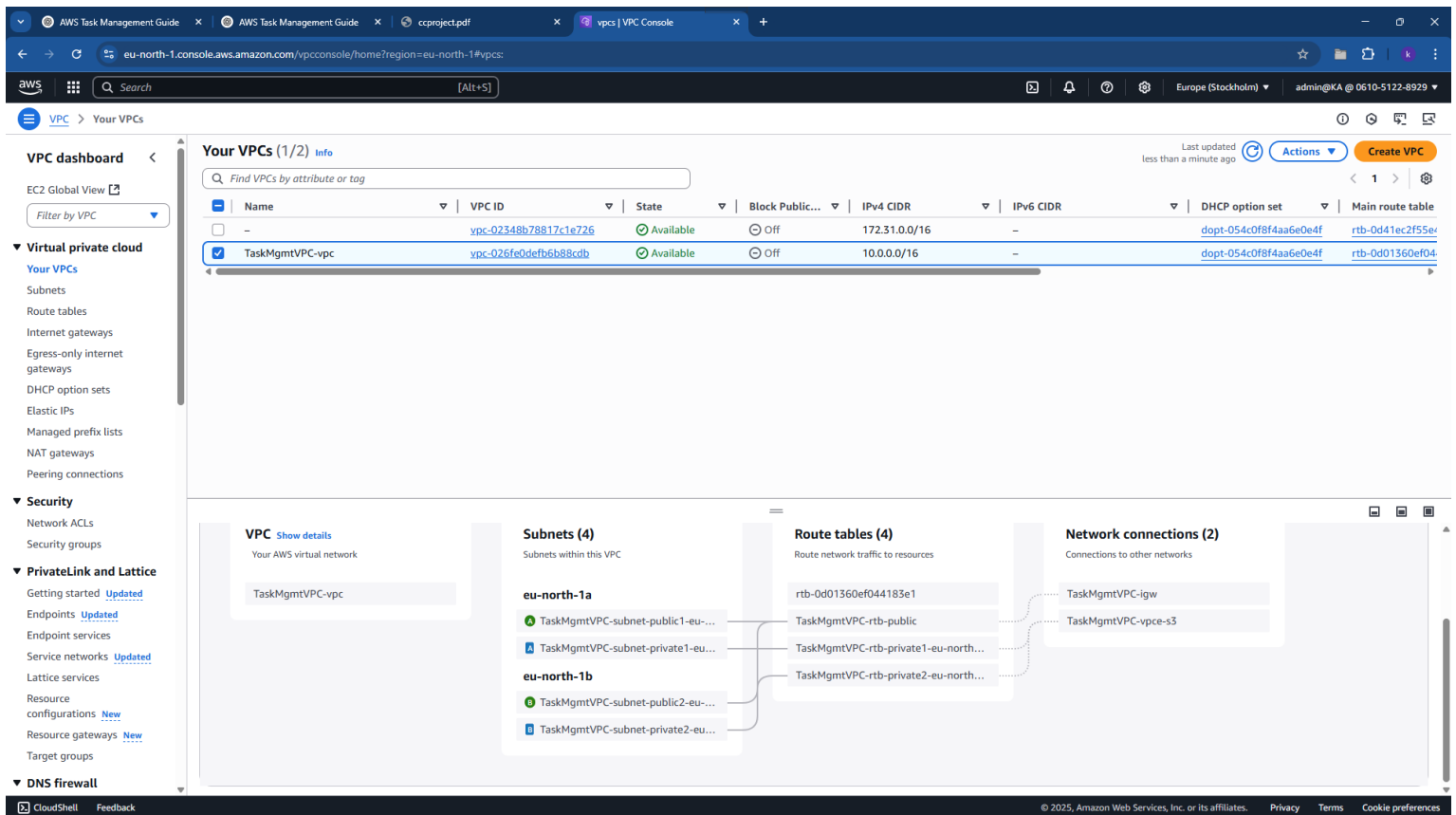


Phase 1: Core AWS Infrastructure Setup

Set up VPC

- **1 Public Subnet** (for EC2).
- **1 Private Subnet** (for databases).
- Configure routing tables and an Internet Gateway.

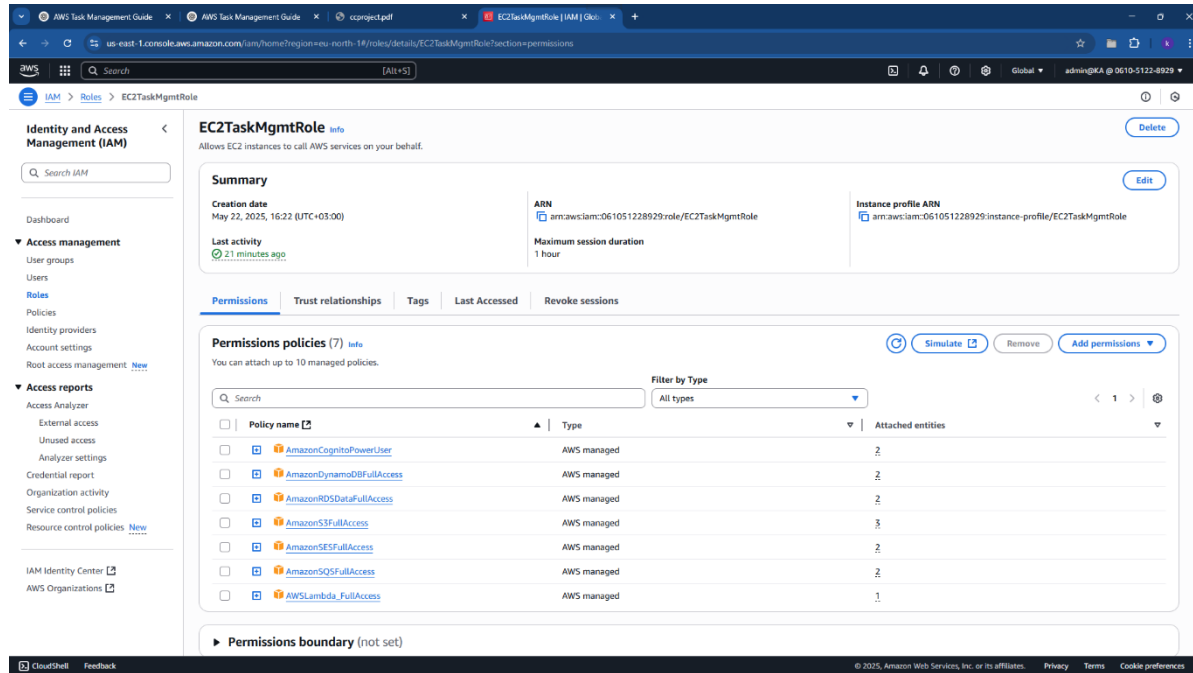


The screenshot shows the AWS VPC console interface. The top navigation bar includes the AWS logo, a search bar, and the user's profile. The main content area is titled 'Your VPCs (1/2)' and displays a table of VPCs. The table has columns for Name, VPC ID, State, Block Public..., IPv4 CIDR, IPv6 CIDR, DHCP option set, and Main route table. The first VPC listed is 'TaskMgmtVPC-vpc' with VPC ID 'vpc-026fe0defb6b88c0b' and state 'Available'. Below the table, there is a diagram showing the VPC structure. The diagram includes a VPC box labeled 'TaskMgmtVPC-vpc', which contains four subnets: 'TaskMgmtVPC-subnet-public1-eu-north-1a', 'TaskMgmtVPC-subnet-private1-eu-north-1a', 'TaskMgmtVPC-subnet-public2-eu-north-1b', and 'TaskMgmtVPC-subnet-private2-eu-north-1b'. The subnets are connected to four route tables: 'rtb-0d01360ef044183e1' (public), 'rtb-0d01360ef044183e1' (private), 'rtb-0d01360ef044183e1' (private), and 'rtb-0d01360ef044183e1' (private). The diagram also shows connections to an Internet Gateway and a VPC peering connection.

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP option set	Main route table
-	vpc-02348b78817c1e726	Available	Off	172.31.0.0/16	-	dopt-054c0f8f4aa6e0e4f	rtb-0d41ec2f55e...
TaskMgmtVPC-vpc	vpc-026fe0defb6b88c0b	Available	Off	10.0.0.0/16	-	dopt-054c0f8f4aa6e0e4f	rtb-0d01360ef04...

Set up IAM Roles/Policies

EC2 Role:



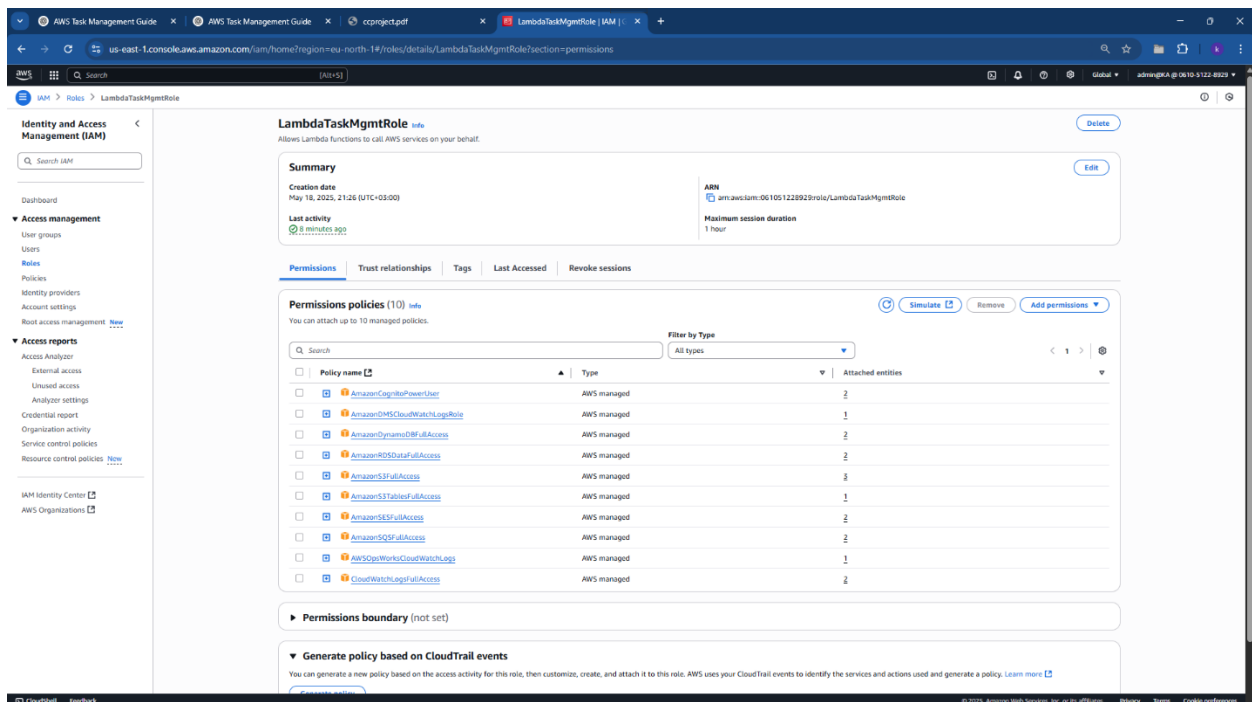
The screenshot shows the AWS IAM console page for the **EC2TaskMgmtRole**. The page is titled "EC2TaskMgmtRole" and includes a description: "Allows EC2 instances to call AWS services on your behalf." The page is divided into several sections:

- Summary:** Displays the role's creation date (May 22, 2025, 16:22 UTC+03:00), last activity (21 minutes ago), ARN (arn:aws:iam:061051228929:role/EC2TaskMgmtRole), and maximum session duration (1 hour).
- Permissions:** A tab showing 7 permissions policies. The table lists the following policies and their attached entities:

Policy name	Type	Attached entities
AmazonCognitoPowerUser	AWS managed	2
AmazonDynamoDBFullAccess	AWS managed	2
AmazonRDSDataFullAccess	AWS managed	2
AmazonS3FullAccess	AWS managed	3
AmazonSESFullAccess	AWS managed	2
AmazonSQSFullAccess	AWS managed	2
AWSLambda_FullAccess	AWS managed	1

Below the table, there is a section for "Permissions boundary" which is currently not set.

Lambda Role:



The screenshot shows the AWS IAM console page for the **LambdaTaskMgmtRole**. The page is titled "LambdaTaskMgmtRole" and includes a description: "Allows Lambda functions to call AWS services on your behalf." The page is divided into several sections:

- Summary:** Displays the role's creation date (May 18, 2025, 21:26 UTC+03:00), last activity (8 minutes ago), ARN (arn:aws:iam:061051228929:role/LambdaTaskMgmtRole), and maximum session duration (1 hour).
- Permissions:** A tab showing 10 permissions policies. The table lists the following policies and their attached entities:

Policy name	Type	Attached entities
AmazonCognitoPowerUser	AWS managed	2
AmazonDMSCloudWatchLogsRole	AWS managed	1
AmazonDynamoDBFullAccess	AWS managed	2
AmazonRDSDataFullAccess	AWS managed	2
AmazonS3FullAccess	AWS managed	2
AmazonS3OutpostsFullAccess	AWS managed	1
AmazonSESFullAccess	AWS managed	2
AmazonSQSFullAccess	AWS managed	2
AWSOpsWorksCloudWatchLogs	AWS managed	1
CloudWatchLogsFullAccess	AWS managed	2

Below the table, there is a section for "Permissions boundary" which is currently not set.

At the bottom of the page, there is a section titled "Generate policy based on CloudTrail events" which provides instructions on how to generate a new policy based on the access activity for this role.

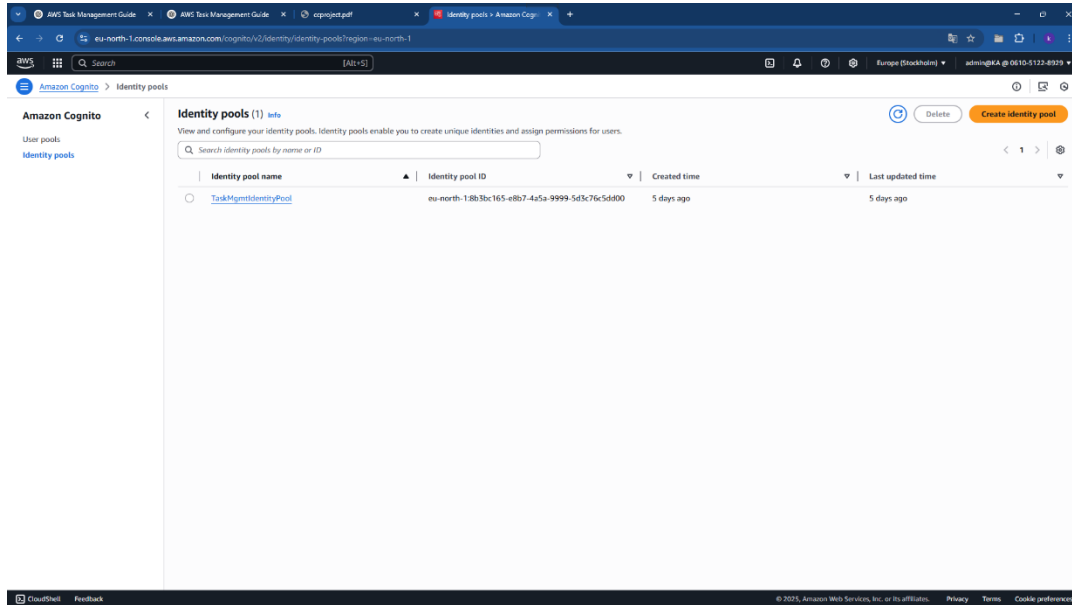


Phase 2: Authentication and Authorization

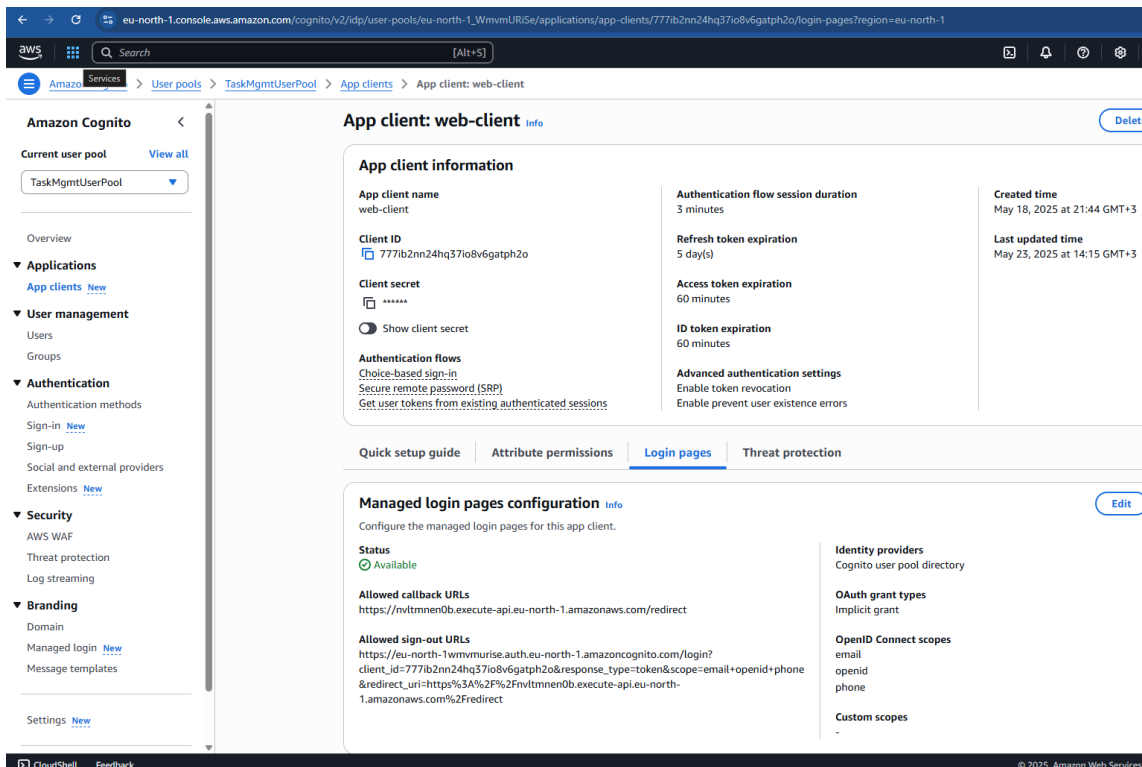


Cognito for User Management

-create an identity pool to give temporary AWS credentials to users



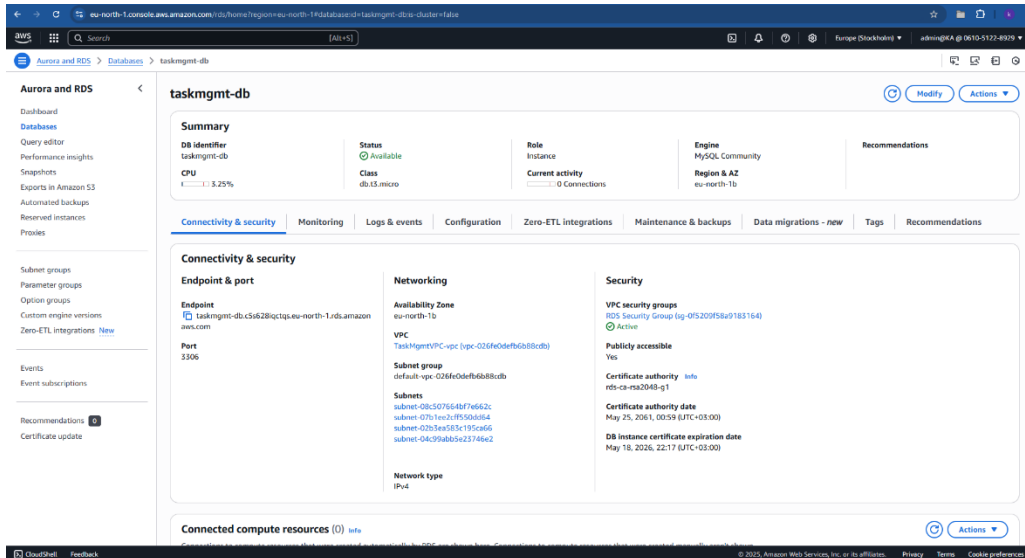
-create a user pool with a hosted UI and an app client



Phase 3: Data Management

Amazon RDS

Create a MySQL database that stores user profiles and task relationships and security group rules for allowing inbound connection

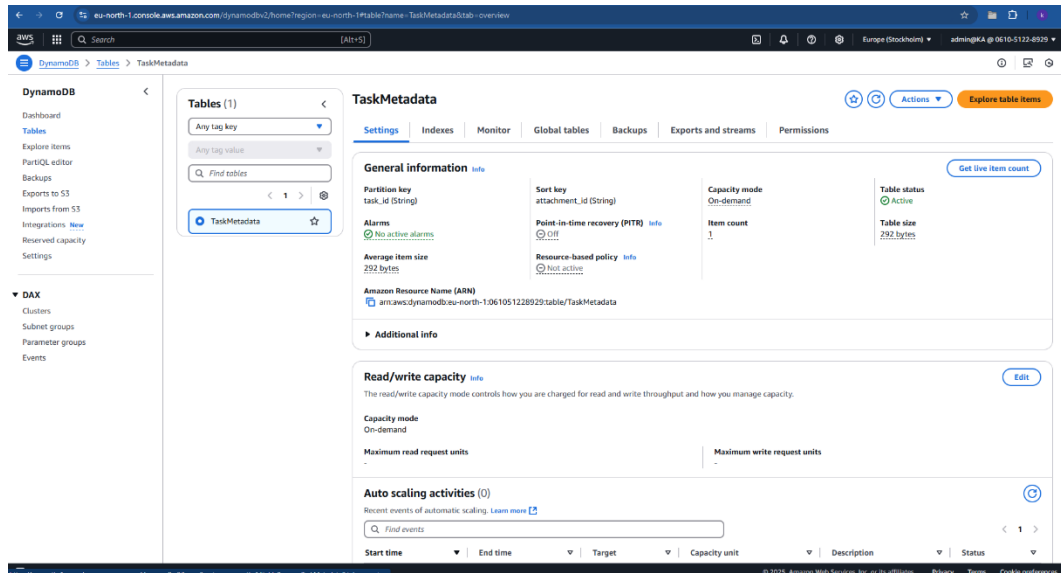


The screenshot displays the AWS Management Console for the Amazon RDS instance 'taskmgmt-db'. The console shows the instance is in an 'Available' state. Key configuration details include:

- Summary:** DB identifier 'taskmgmt-db', CPU 'db.t3.micro', Engine 'MySQL Community', and Region & AZ 'eu-north-1a'.
- Connectivity & security:** Endpoint 'taskmgmt-db.c5s82lqtpg.eu-north-1.rds.amazonaws.com', Port '3306', and VPC 'TaskMgmtVPC-vpc'.
- Security:** VPC security groups 'rds-ca-2048-g1' and 'rds-ca-rs2048-g1', and a publicly accessible instance.

Dynamo DB

Store task metadata including attachment's s3 key and name for retrieving them from the s3 bucket they are stored in

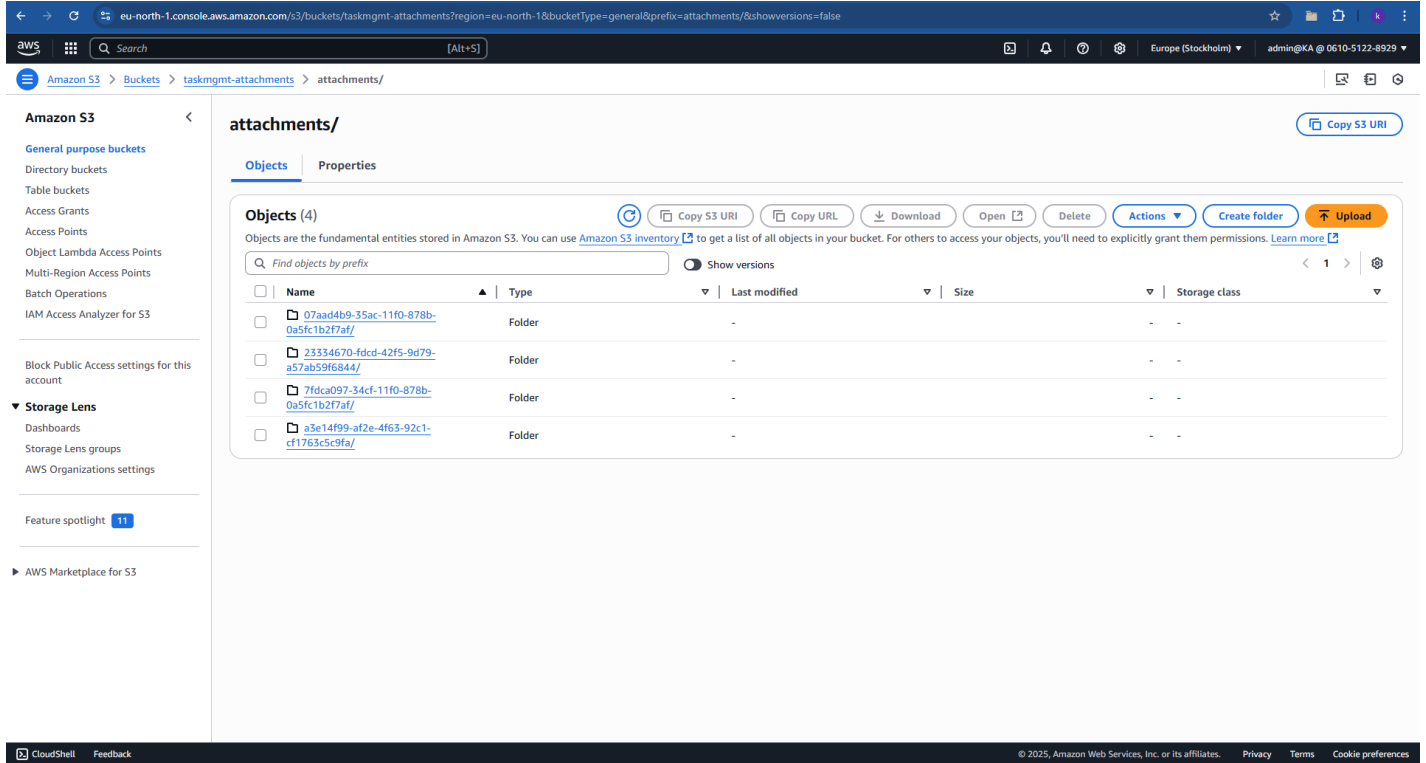


The screenshot displays the AWS Management Console for the DynamoDB table 'TaskMetadata'. The console shows the table is in an 'Active' state. Key configuration details include:

- General information:** Partition key 'task_id (String)', Sort key 'attachment_id (String)', Capacity mode 'On-demand', and Item count '1'.
- Read/write capacity:** Capacity mode 'On-demand', Maximum read request units '-', and Maximum write request units '-'.
- Auto scaling activities:** Recent events of automatic scaling.

Amazon S3

Create a bucket for file attachments.



The screenshot shows the Amazon S3 console interface. The main content area displays the 'attachments/' bucket with a list of four folders. The left sidebar contains navigation links for various S3 features. The top navigation bar shows the AWS logo, search bar, and user information.

Amazon S3

General purpose buckets

- Directory buckets
- Table buckets
- Access Grants
- Access Points
- Object Lambda Access Points
- Multi-Region Access Points
- Batch Operations
- IAM Access Analyzer for S3

Storage Lens

- Dashboards
- Storage Lens groups
- AWS Organizations settings

Feature spotlight 11

AWS Marketplace for S3

attachments/

Objects (4)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

☒ Show versions

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	07aad4b9-35ac-11f0-878b-0a5fc1b2f7af/	Folder	-	-	-
<input type="checkbox"/>	23334670-fdc4-42f5-9d79-a57ab59f6844/	Folder	-	-	-
<input type="checkbox"/>	7fdca097-34cf-11f0-878b-0a5fc1b2f7af/	Folder	-	-	-
<input type="checkbox"/>	a3e14f99-af2e-4f63-92c1-cf1763c5c9fa/	Folder	-	-	-

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)



Phase 4: Backend Logic



API Gateway


Create an Api gateway with each Api configured with authorization (Cognito authorizer) and integrations to trigger lambda functions when Api is called.

Configure CORS: Access-Control-Allow-Origin, Access-Control-Allow-Headers, Access-Control-Allow-Methods

The screenshot shows the AWS API Gateway console for the 'TaskMgmtAPI (w2vsfcgub7)'. The 'Authorization' tab is selected, showing 'Routes for TaskMgmtAPI' on the left and 'Attach authorizers to routes' on the right. The routes listed are: /delete-account (DELETE, JWT Auth), /delete-attachment (DELETE, JWT Auth), /tasks (POST, GET, PUT, DELETE, JWT Auth), and /upload-attachment (POST, JWT Auth). The 'Attach authorizers to routes' section shows the 'Authorize for route DELETE /delete-account' configuration. The 'Authorizer name' is 'CognitoAuth', the 'Authorizer type' is 'JWT', and the 'Authorizer ID' is 'c12bg3'. The 'Identity source' is 'request.header.Authorization'. The 'Issuer' is 'https://cognito-idp.eu-north-1.amazonaws.com/eu-north-1_WmvmURI5e'. The 'Audience' is '777b2m24hq37o8v6gath2a'. The 'Authorization scopes' section is empty, with an 'Add scope' button. A 'Save' button is at the bottom right.

The screenshot shows the AWS API Gateway console for the 'TaskMgmtAPI (w2vsfcgub7)'. The 'Integrations' tab is selected, showing 'Routes for TaskMgmtAPI' on the left and 'Attach integrations to routes' on the right. The routes listed are: /delete-account (DELETE, AWS Lambda), /delete-attachment (DELETE, AWS Lambda), /tasks (POST, GET, PUT, DELETE, AWS Lambda), and /upload-attachment (POST, AWS Lambda). The 'Attach integrations to routes' section shows the 'Integration details for route DELETE /delete-account (2mykhg5)'. The 'Lambda function' is 'delete_user (eu-north-1)'. The 'Integration ID' is 'qt45t4c'. The 'Description' is empty. The 'Payload format version' is '2.0 (interpreted response format)'. The 'Invoke permissions' section is empty. The 'Example policy statement' is empty. The 'Timeout' is '30000'. The 'Request parameter mapping' is 'Not configured'. The 'Response parameter mappings' are 'Not configured'. A 'Save' button is at the bottom right.

Lambda Functions

 [Lambda](#) > [Functions](#) 🔍 📄 🔄

Lambda

Dashboard

Applications

Functions

▼ **Additional resources**

Code signing configurations

Event source mappings

Layers

Replicas



▼ **Related AWS resources**


Step Functions state machines


Functions (13)

🔍 *Filter by attributes or search by keyword*

<input type="checkbox"/>	Function name	Description	Package type	Runtime	Last modified
<input type="checkbox"/>	GetTasksWithAttachments	-	Zip	Node.js 22.x	4 days ago
<input type="checkbox"/>	TaskNotificationProcessor	-	Zip	Node.js 22.x	2 days ago
<input type="checkbox"/>	updateTaskFunction	-	Zip	Node.js 22.x	2 days ago
<input type="checkbox"/>	storeUserInMySQL	-	Zip	Node.js 22.x	2 days ago
<input type="checkbox"/>	getAltTaskFunction	-	Zip	Node.js 22.x	4 days ago
<input type="checkbox"/>	upload_attachment	-	Zip	Node.js 22.x	2 days ago
<input type="checkbox"/>	CognitoRedirectHandler	-	Zip	Node.js 22.x	16 hours ago
<input type="checkbox"/>	delete_attachment	-	Zip	Node.js 22.x	2 days ago
<input type="checkbox"/>	getTaskByIdFunction	-	Zip	Node.js 22.x	3 days ago
<input type="checkbox"/>	addUserIdentity	-	Zip	Node.js 22.x	2 days ago
<input type="checkbox"/>	deleteTaskFunction	-	Zip	Node.js 22.x	2 days ago
<input type="checkbox"/>	createTaskFunction	-	Zip	Node.js 22.x	2 days ago
<input type="checkbox"/>	delete_user	-	Zip	Node.js 22.x	2 days ago

Last fetched 0 seconds ago  Actions  [Create function](#)

< 1 > 

 CloudShell [Feedback](#) © 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

- Task operations (CRUD).
- Handle RDS/DynamoDB queries via AWS SDK.
- Handle S3 file uploads/deletions.
- Publish messages to SQS on task updates.
- Manage Notifications triggering.
- handle syncing data between all services.



Phase 5: Notifications and Asynchronous Processing



Amazon SQS

Create a queue for task update notifications, Lambda function reads messages and sends emails (via SES).

The screenshot shows the Amazon SQS console for the 'TaskEventsQueue'. The queue is of type 'Standard' and is encrypted with an Amazon SQS key (SSE-SQS). The URL is 'https://sqs.eu-north-1.amazonaws.com/061051228929/TaskEventsQueue'. The queue has one Lambda trigger named 'TaskNotificationProcessor' which is enabled. The console also shows tabs for 'SNS subscriptions', 'EventBridge Pipes', 'Dead-letter queue', 'Monitoring', 'Tagging', 'Queue policies', 'Encryption', and 'Dead-letter queue redrive tasks'.



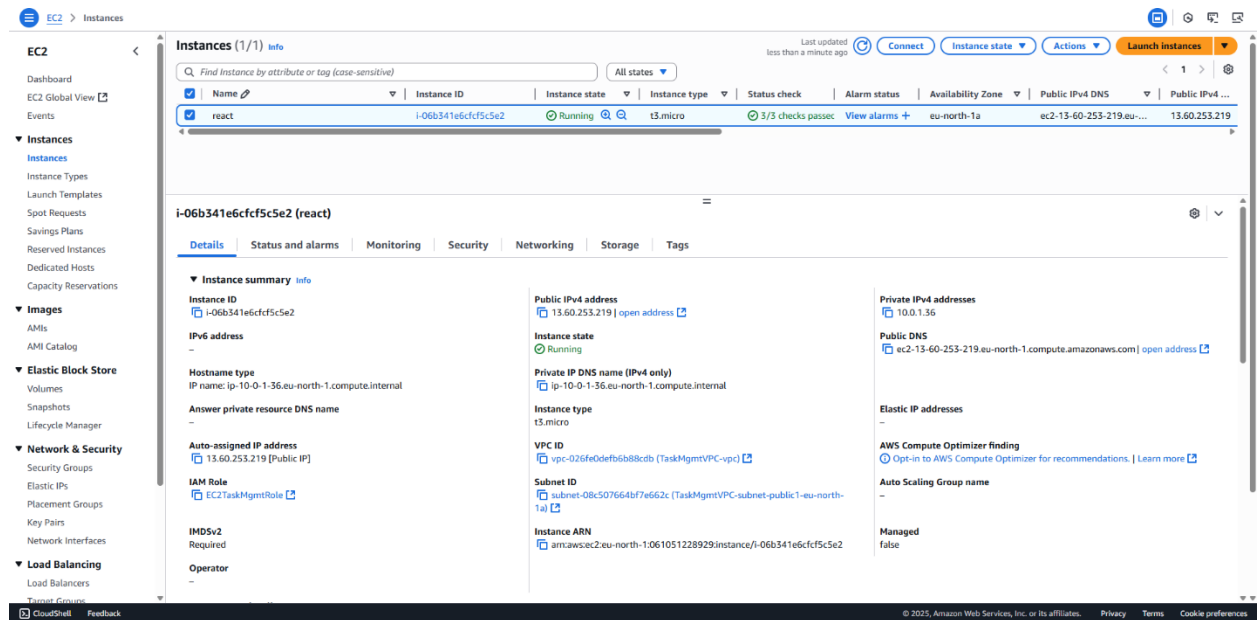
Amazon SES

create SES identity to send emails to users

The screenshot shows the Amazon SES console for the 'Identities' configuration. It lists two identities: 'karimam55522@gmail.com' and 'karimam80@gmail.com', both of which are verified. The console also shows a 'Recommendations' section with a message: 'No recommendations found. Select up to 10 verified domain identities and select Check for recommendations on the identities table above.' The console also includes a sidebar with navigation links for 'Amazon SES', 'Configuration', 'Virtual Deliverability Manager', and 'Mail Manager'.

Phase 6: Frontend and Hosting

EC2 instance



The screenshot displays the AWS Management Console interface for the EC2 service. The left sidebar shows the navigation menu with categories like EC2, Images, Elastic Block Store, Network & Security, and Load Balancing. The main content area is titled 'Instances (1/1)' and shows a table with one instance named 'react'. Below the table, the 'Details' tab is selected, showing various instance attributes:

- Instance ID:** i-06b341e6cfc5c5e2
- Instance state:** Running
- Instance type:** t3.micro
- Status check:** 3/3 checks passed
- Alarm status:** View alarms
- Availability Zone:** eu-north-1a
- Public IPv4 DNS:** ec2-13-60-253-219.eu-north-1.compute.amazonaws.com
- Public IPv4 address:** 13.60.253.219

The 'Instance summary' section provides a detailed overview of the instance's configuration, including its hostname type, IP addresses, VPC ID, subnet ID, and IAM role.

- created an ec2 instance to host the react frontend I created
- github containing the frontend code: <https://github.com/KarimAmr15/AWS-TaskManagement>

after connecting to the ec2 instance using SSH:

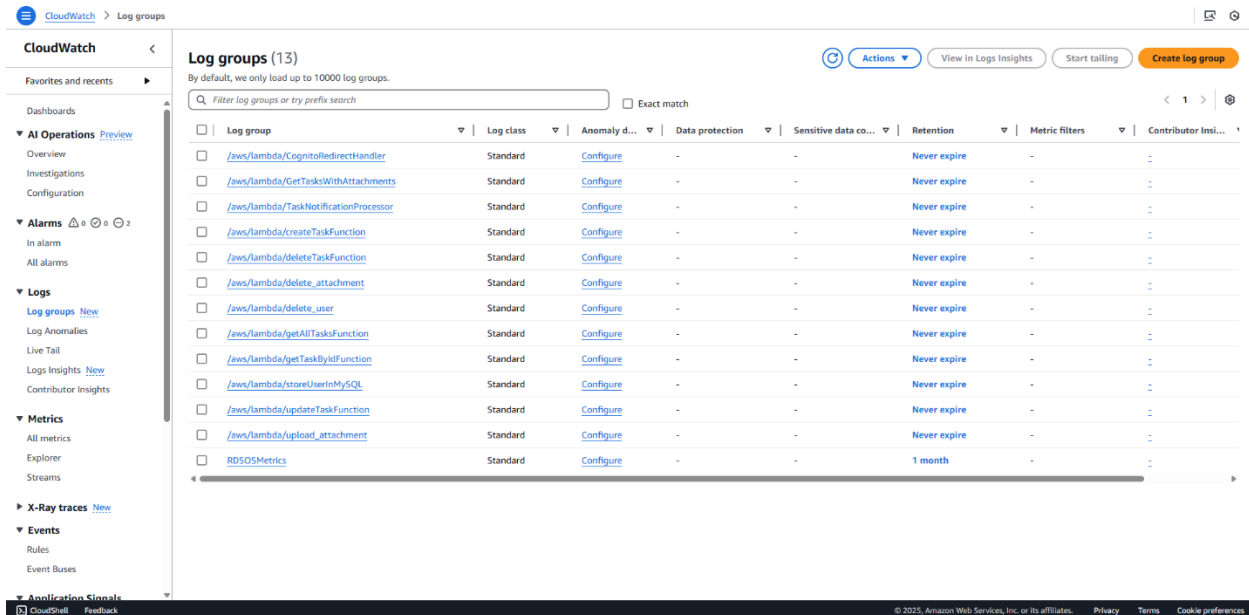
1. Prepare React App (run build)
2. Install NGINX
3. Move files to web root
4. Edit NGINX config (/etc/nginx/nginx.conf) to handle the react app routing correctly
5. Restart NGINX

Phase 7: Monitoring and Logging

□ Amazon CloudWatch

- Enable logging for:
 - Lambda functions.
 - API Gateway access logs.
- Set up metrics dashboards.
- Create CloudWatch Alarms for errors or latency spikes.

LOG GROUPS



CloudWatch > Log groups

CloudWatch <

Log groups (13)

By default, we only load up to 10000 log groups.

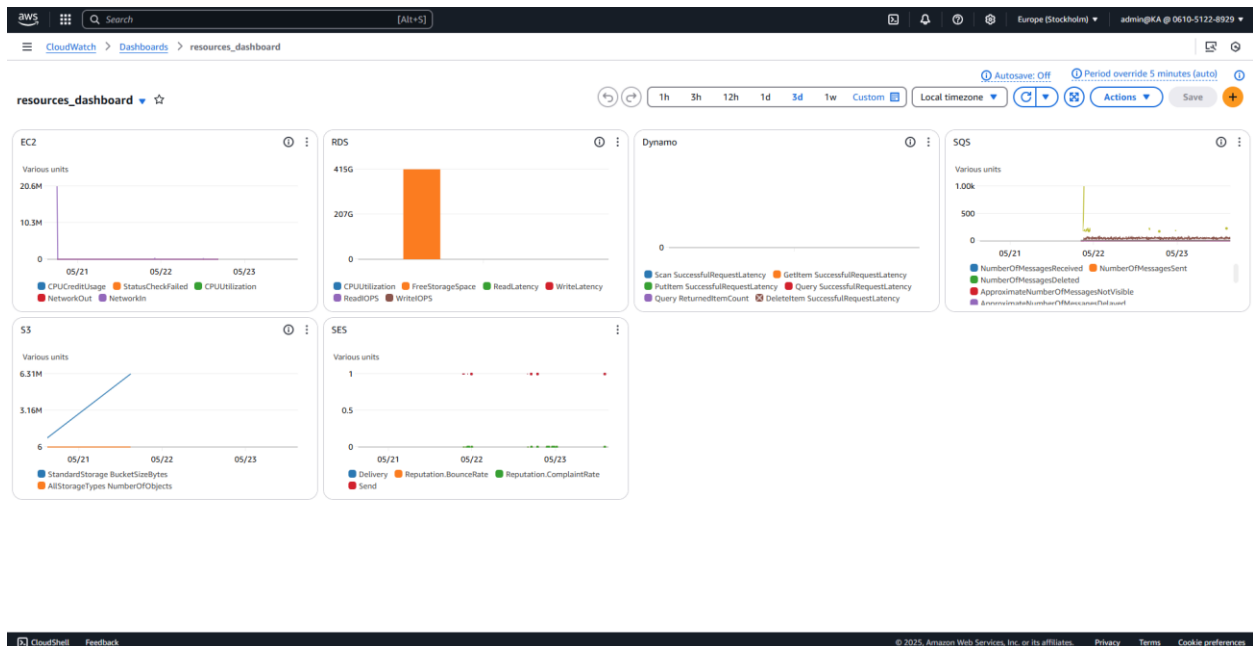
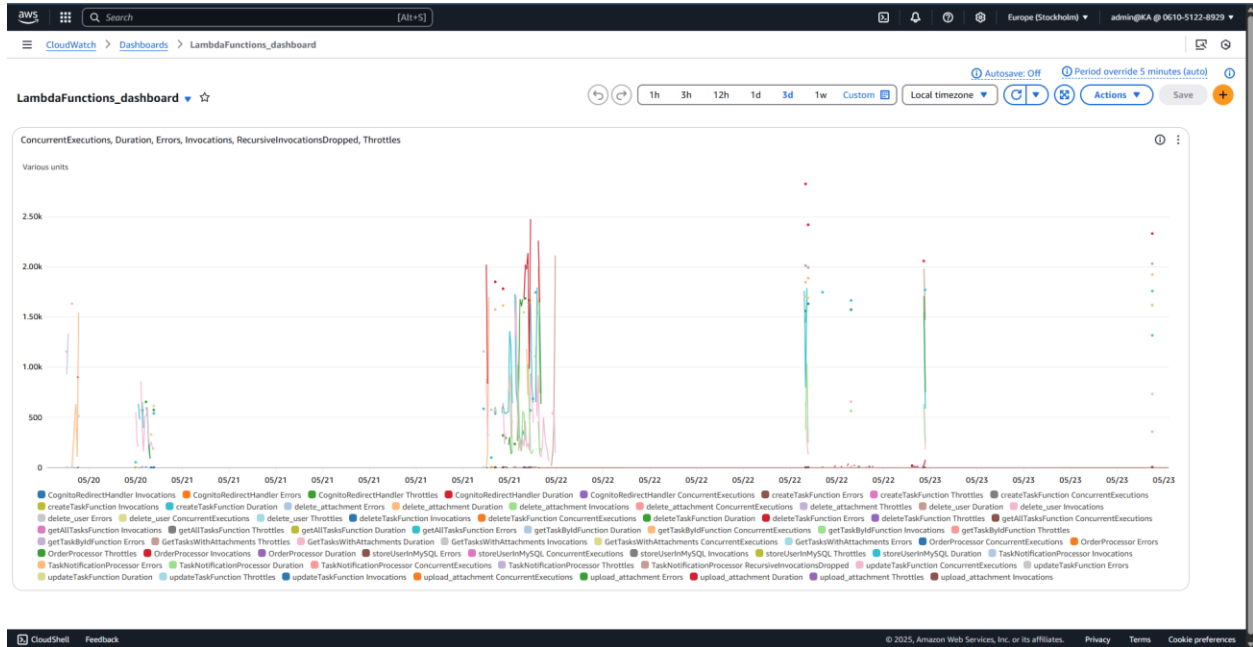
Filter log groups or try prefix search

Exact match

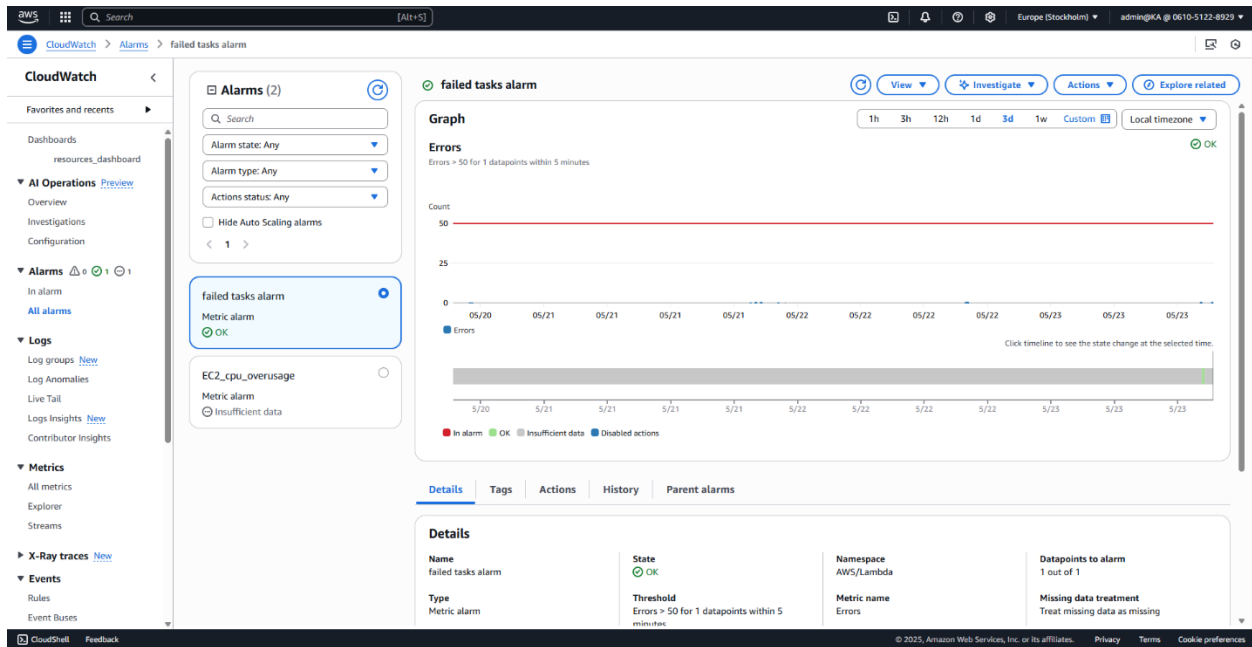
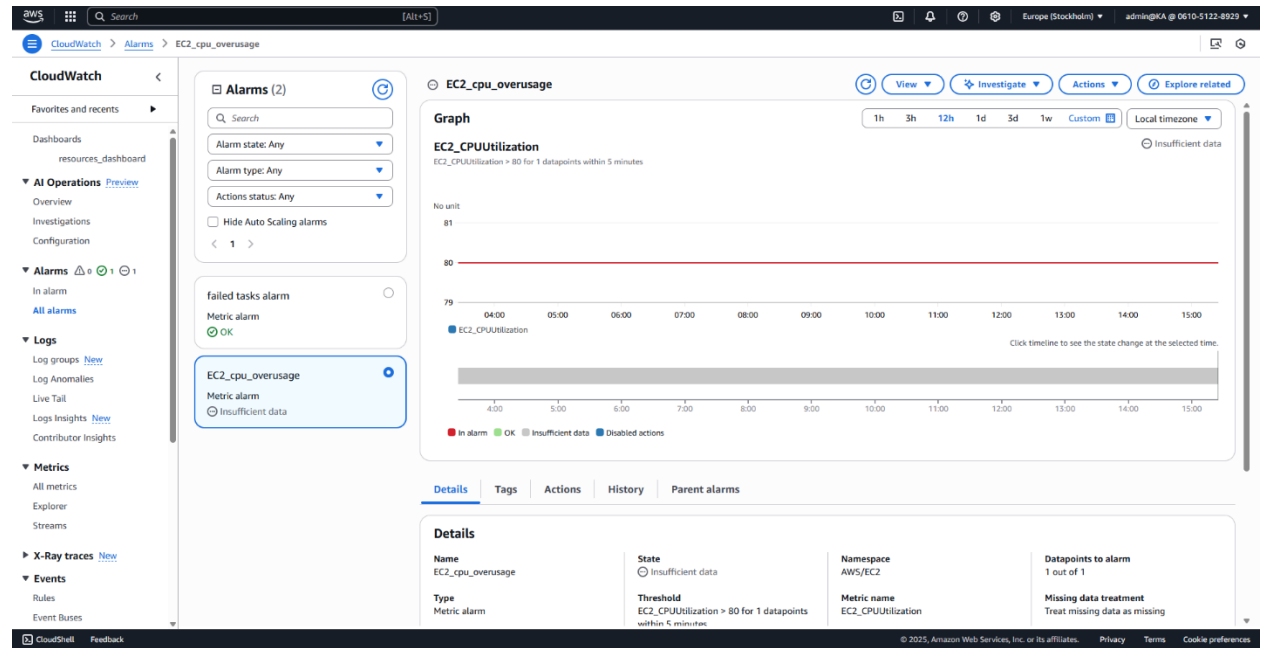
<input type="checkbox"/>	Log group	Log class	Anomaly d...	Data protection	Sensitive data co...	Retention	Metric filters	Contributor Insi...
<input type="checkbox"/>	/aws/lambda/CognitoRedirectHandler	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/GetTaskWithAttachments	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/TaskNotificationProcessor	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/createTaskFunction	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/deleteTaskFunction	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/delete_attachment	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/delete_user	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/getAllTaskFunction	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/getTaskByIdFunction	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/storeUserInMySQL	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/updateTaskFunction	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	/aws/lambda/upload_attachment	Standard	Configure	-	-	Never expire	-	⋮
<input type="checkbox"/>	RDSOSMetrics	Standard	Configure	-	-	1 month	-	⋮

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

DASHBOARDS



ALARMS



Architecture

