



Archetype AI

AWS Marketplace Deployment Guide

Version: 1.0.0-99

Executive summary

Archetype exists to unlock the potential of physical AI through Newton, a Large Behavior Model (LBM) that understands and reasons about the physical world using multimodal sensor data. Built on advanced machine learning techniques, Newton processes and interprets data from any sensor type while ensuring data privacy and enabling edge deployment. With a focus on the trillion-dollar sensor economy, Newton is positioned to transform industries such as construction and manufacturing by providing new insights into their physical operations.

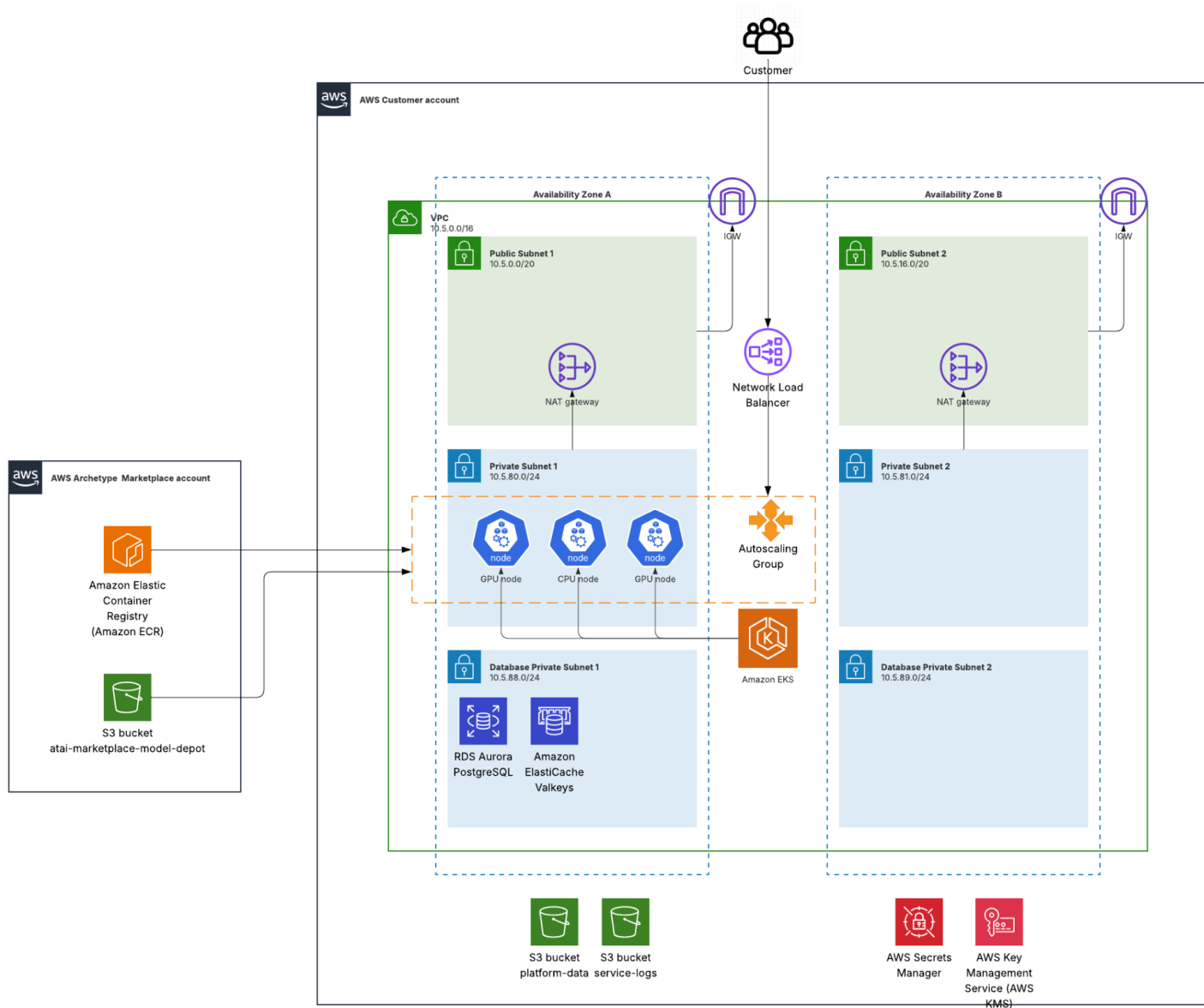
Table of contents

Executive summary	2
Table of contents	3
Infrastructure Overview	6
AWS License Manager setup	7
AWS infrastructure prerequisites	7
VPC configuration	7
Step 1: Create the VPC (Manual Configuration)	7
Step 1.1 Enable DNS Hostnames (if not already enabled)	8
Step 2: Create Internet Gateway	9
Step 3: Create Public Subnets (with correct CIDRs from the start)	10
3.1 Enable Auto-assign Public IPv4 for Public Subnets	11
Step 4: Create Private Subnets	12
Step 5: Create Database Subnets	13
Step 6: Allocate Elastic IP for NAT Gateway	14
Step 7: Create NAT Gateway	15
Step 8: Create and Configure Route Tables	16
Public Route Table:	16
Private Route Table:	18
Database Route Table:	20
Valkey clusters configuration	22
Prerequisites	22
Step 1: Create ElastiCache Subnet Group (if not existing)	22
Step 2: Create Security Group (if not existing)	25
Step 3: Create Parameter Group for Valkey 8.0	26
Step 3.1 After creation, edit the parameter group:	26
Step 4: Create Valkey Clusters	28
Cluster 1: registry	28
Cluster 2: access-manager	35
Cluster 3: api-events	36
Cluster 4: dfc (10 shards)	36
Cluster 5: file (10 shards)	37
Cluster 6: gpq (10 shards)	37
Cluster 7: health	38
Cluster 8: lens (10 shards)	39
Step 5: Store Auth Tokens in AWS Secrets Manager (Recommended)	40
Benefits of using Secrets Manager:	43
PostgreSQL database configuration	44
Prerequisites	44

Step 1: Create Database Subnet Group (if not already created)	44
Step 2: Create Security Group (if not existing)	45
Step 3: Create Aurora PostgreSQL Cluster	47
Step 4: Extra Database Configuration steps	54
Create databases	54
Create atai_dev database user	55
EKS cluster configuration	58
Prerequisites	58
Step 1: Create cluster IAM role	58
Step 2: Create cluster	60
Step 3: Update kubeconfig	66
Step 4: Get the default cluster security group	67
EKS managed node group configuration	68
Prerequisites	68
Step 1: Creating the Amazon EKS node IAM role	69
Step 2: Creating Node Security Group	71
Step 2.1 Add self rules to the node security group	72
Step 3: Launch templates	74
Step 3.1 CPU Node Group	74
Step 3.2 GPU Node Group	78
Step 4: Create CPU Node Group	82
Step 5: Create GPU Node Group	87
EKS configuration - Install the NVIDIA Device Plugin	92
Prerequisites	92
Step 1: Manual installation	92
S3 configuration	93
Step 1: Create the platform-data bucket	93
Step 2: Create the service logs bucket	96
Application Endpoints Configuration	98
Platform Architecture Overview	98
Required Public URLs	98
How Do I Expose My Services Publicly?	98
What is an Ingress?	98
How Do I Secure the Traffic?	99
How Do I Configure DNS?	99
Deployment Checklist	100
Before Helm Chart Installation:	100
Support	101
Prerequisites	102
Download the configuration files	102
Step 1: Kubernetes namespaces	103

Step 2: Kubernetes Service account for IAM roles (IRSA)	103
Step 3: Kubernetes secrets required for the atai-platform services	106
Step 3.1 Generate values for the IAM service secret	106
Step 3.2 Kubernetes secret generation	108
Helm chart installation	113
Prerequisites	113
Step 1: Installation	113
Getting Started with the Archetype Platform	115
Appendix	116
Bastion host configuration	116
Prerequisites	116
Step 1: Launch EC2 Instance (Bastion Host)	116
Step 2: Connect to your Bastion host	121
AWS Service Quotas	122
Running On-Demand G and VT instances	122
Running On-Demand Standard (A, C, D, H, I, M, R, T, Z) instances	124
AWS License Manager Setup	125
Get started with License Manager	125
EKS configuration - Install the Cert Manager	127
EKS configuration - Install the NGINX ingress controller	127

Infrastructure Overview



atai Infrastructure

AWS License Manager setup

This Marketplace product uses **AWS License Manager** for proper operation. You **must enable and configure AWS License Manager** in your account before deployment.

Failure to do so will prevent the product from functioning correctly. For more information on how to set up License Manager correctly, see [Appendix: License Manager Setup](#).

AWS infrastructure prerequisites

This document outlines the AWS infrastructure prerequisites that must be deployed before installing atai-platform helm chart.

Note: Before starting the creation of the following resources, create a **secrets.ini** file based on the [secrets.ini.template](#). This file will be required on [Step 3: Kubernetes secrets required for the atai-platform services](#).

VPC configuration

Create VPC with Public, Private, and Database Subnets

Step 1: Create the VPC (Manual Configuration)

1. Go to VPC Dashboard → Your VPCs → Create VPC
2. Select VPC only (not "VPC and more")
3. Configure:
 - a. Name tag: atai-platform-vpc
 - b. IPv4 CIDR block: 10.5.0.0/16
 - c. IPv6 CIDR block: No IPv6 CIDR block
 - d. Tenancy: Default
4. Click Create VPC

VPC > Your VPCs > Create VPC

CreateVpc

VPC settings

Resources to create [Info](#)
 Create only the VPC resource or the VPC and other networking resources.

VPC only VPC and more

Name tag - optional
 Creates a tag with a key of 'Name' and a value that you specify.

atai-platform-vpc

IPv4 CIDR block [Info](#)
 IPv4 CIDR manual input
 IPAM-allocated IPv4 CIDR block

IPv4 CIDR
 10.5.0.0/16
CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)
 No IPv6 CIDR block
 IPAM-allocated IPv6 CIDR block
 Amazon-provided IPv6 CIDR block
 IPv6 CIDR owned by me

Tenancy [Info](#)
 Default

Tags
 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 - optional**

You can add 49 more tags

[Cancel](#) [Preview code](#)

Step 1.1 Enable DNS Hostnames (if not already enabled)

1. Go to Your VPCs → select your VPC
2. Actions → Edit VPC settings

Your VPCs (1/1) [Info](#)

<input checked="" type="checkbox"/>	Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP opti...
<input checked="" type="checkbox"/>	atai-platform-vpc	vpc-0a79faee3e664a31d	Available	Off	10.5.0.0/16	-	dopt-03a8172d4e5b9cd3f

[Actions](#)

- Create default VPC
- Create flow log
- Edit VPC settings**
- Edit CIDRs
- Manage middlebox routes
- Manage tags
- Delete VPC

3. Enable DNS hostnames and DNS resolution
4. Save

VPC > Your VPCs > vpc-0a79faee3e664a31d > Edit VPC settings

Edit VPC settings

VPC details

VPC ID: vpc-0a79faee3e664a31d
 Name: atai-platform-vpc

DHCP settings
 DHCP option set: dopt-03a8172d4e5b9cd3f

DNS settings

Enable DNS resolution [Info](#)
 Enable DNS hostnames [Info](#)

Network Address Usage metrics settings
 Enable Network Address Usage metrics [Info](#)

[Cancel](#)

Step 2: Create Internet Gateway

1. Go to VPC Dashboard → Go to Internet Gateways → Create internet gateway
2. Name tag: atai-platform-vpc-igw
3. Click Create internet gateway

Create internet gateway Info

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

Internet gateway settings

Name tag
Creates a tag with a key of 'Name' and a value that you specify.

atai-platform-vpc-igw

Tags - optional
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 - optional**

Q Name X Q atai-platform-vpc-igw X Remove

Add new tag
You can add 49 more tags.

Cancel **Create internet gateway**

4. Select it → Actions → Attach to VPC

igw-05499c47a963a31f8

InternetGateway ig internet gateway was created: igw-05499c47a963a31f8 - atai-platform-vpc-igw. You can now attach to a VPC to enable the VPC to communicate with the internet. **Attach to a VPC** X

igw-05499c47a963a31f8 / atai-platform-vpc-igw

Details Info

Internet gateway ID
igw-05499c47a963a31f8

State
Detached

VPC ID
-

Owner
716124474177

Actions

Attach to VPC
Detach from VPC
Manage tags
Delete

Tags (1)

Search tags

Key	Value
Name	atai-platform-vpc-igw

Manage tags
< 1 > ⚙

5. Select your VPC → Attach internet gateway

Attach to VPC (igw-05499c47a963a31f8) Info

VPC
Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach below.

Available VPCs
Attach the internet gateway to this VPC.

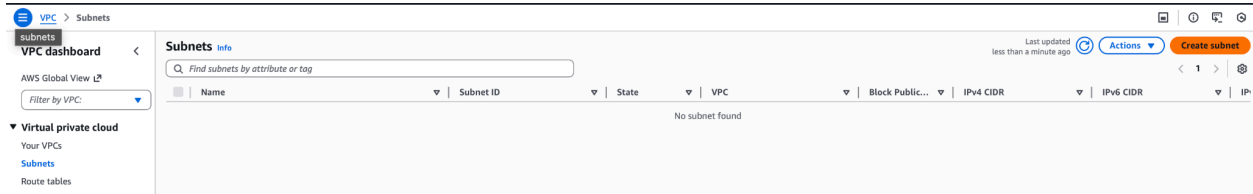
Q vpc-0a79faee3e664a31d X

► **AWS Command Line Interface command**

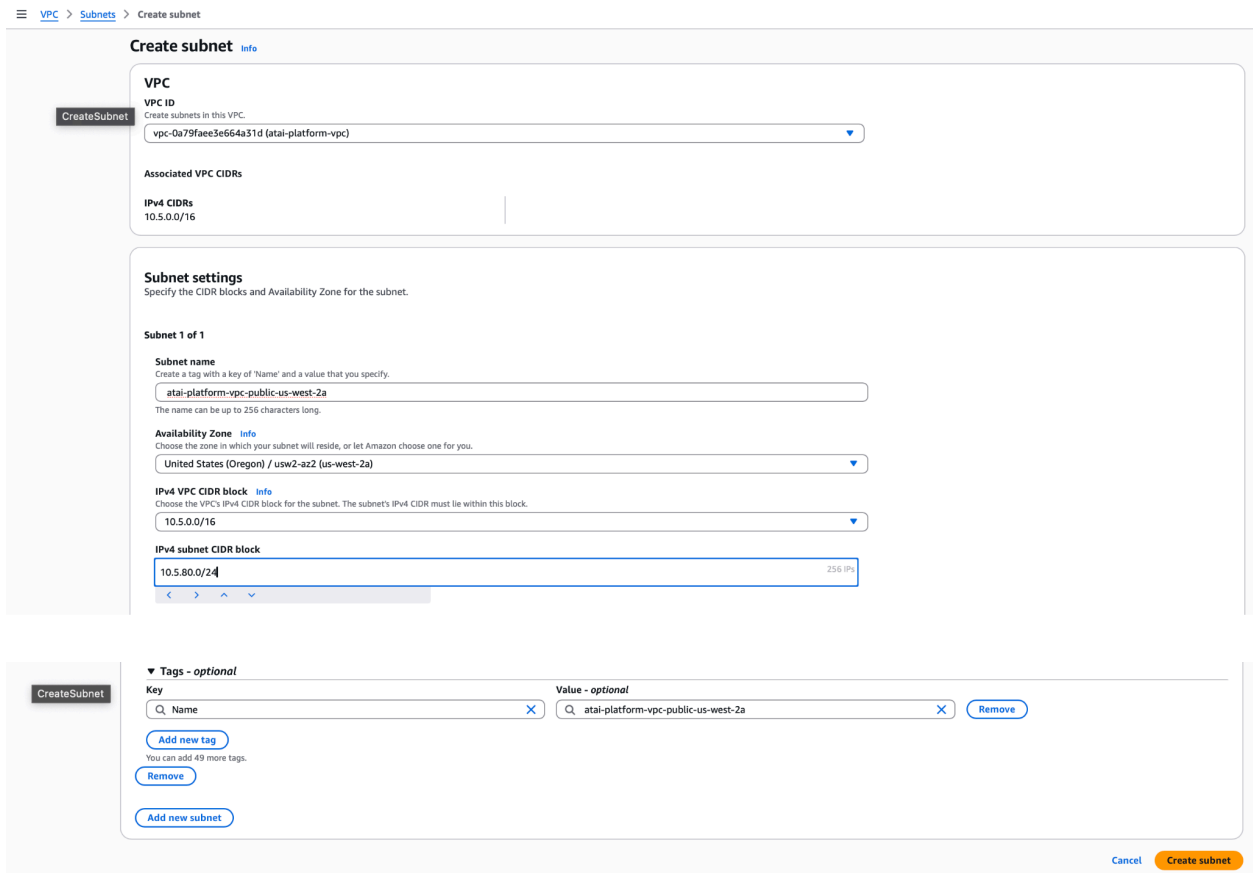
Cancel **Attach internet gateway**

Step 3: Create Public Subnets (with correct CIDRs from the start)

1. Go to VPC Dashboard → Go to Subnets → Create subnet



2. Subnet 1:
 - a. Select the VPC created in the Step 1
 - b. VPC: Select your VPC
 - c. Subnet name: atai-platform-vpc-public-us-west-2a
 - d. Availability Zone: us-west-2a (or your AZ1)
 - e. IPv4 CIDR block: 10.5.80.0/24
 - f. Click Create subnet

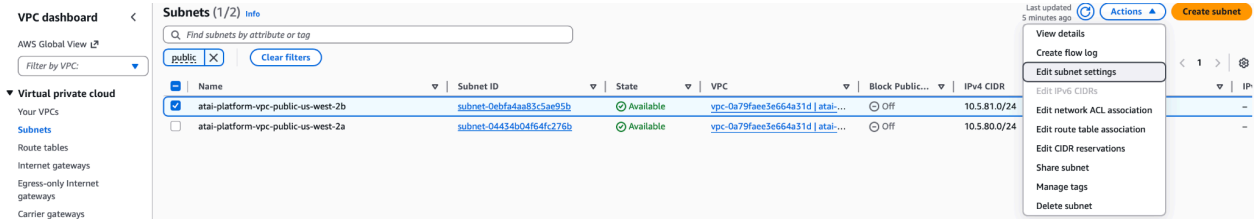


3. Subnet 2:
 - a. Select the VPC created in the Step 1
 - b. Subnet name: atai-platform-vpc-public-us-west-2b

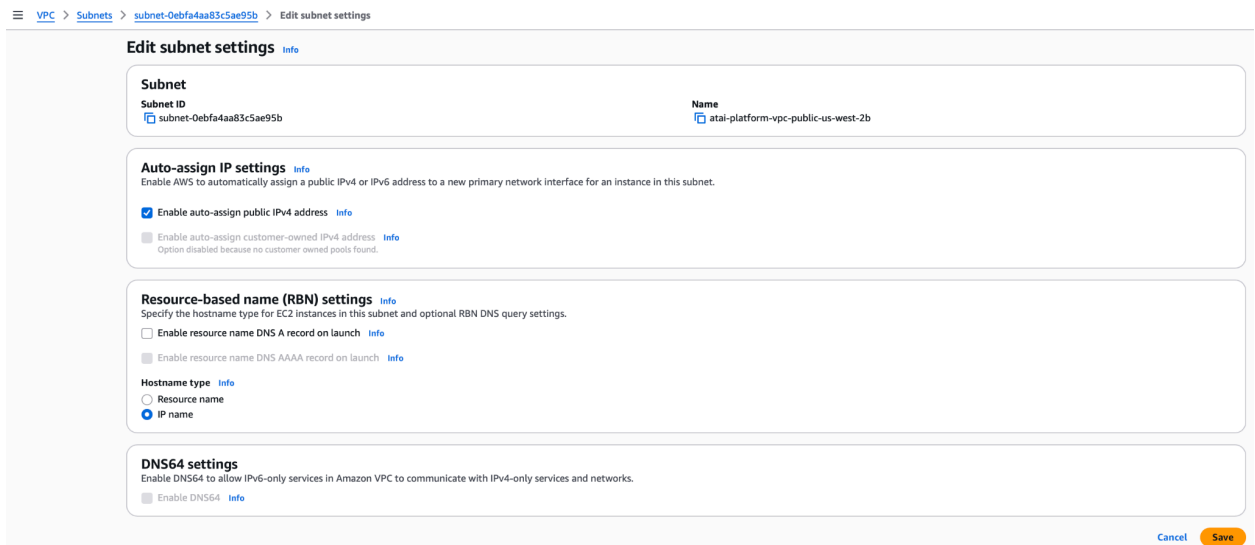
- c. Availability Zone: us-west-2b (or your AZ2)
- d. IPv4 CIDR block: 10.5.81.0/24
- e. Click Create subnet

3.1 Enable Auto-assign Public IPv4 for Public Subnets

1. Go to VPC Dashboard → Go to Subnets → select each public subnet
2. Actions → Edit subnet settings



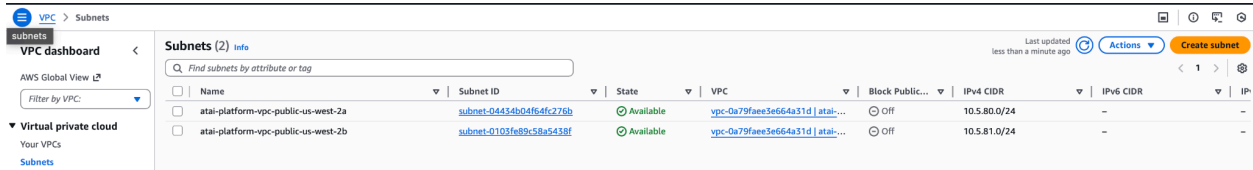
3. Check Enable auto-assign public IPv4 address
4. Save



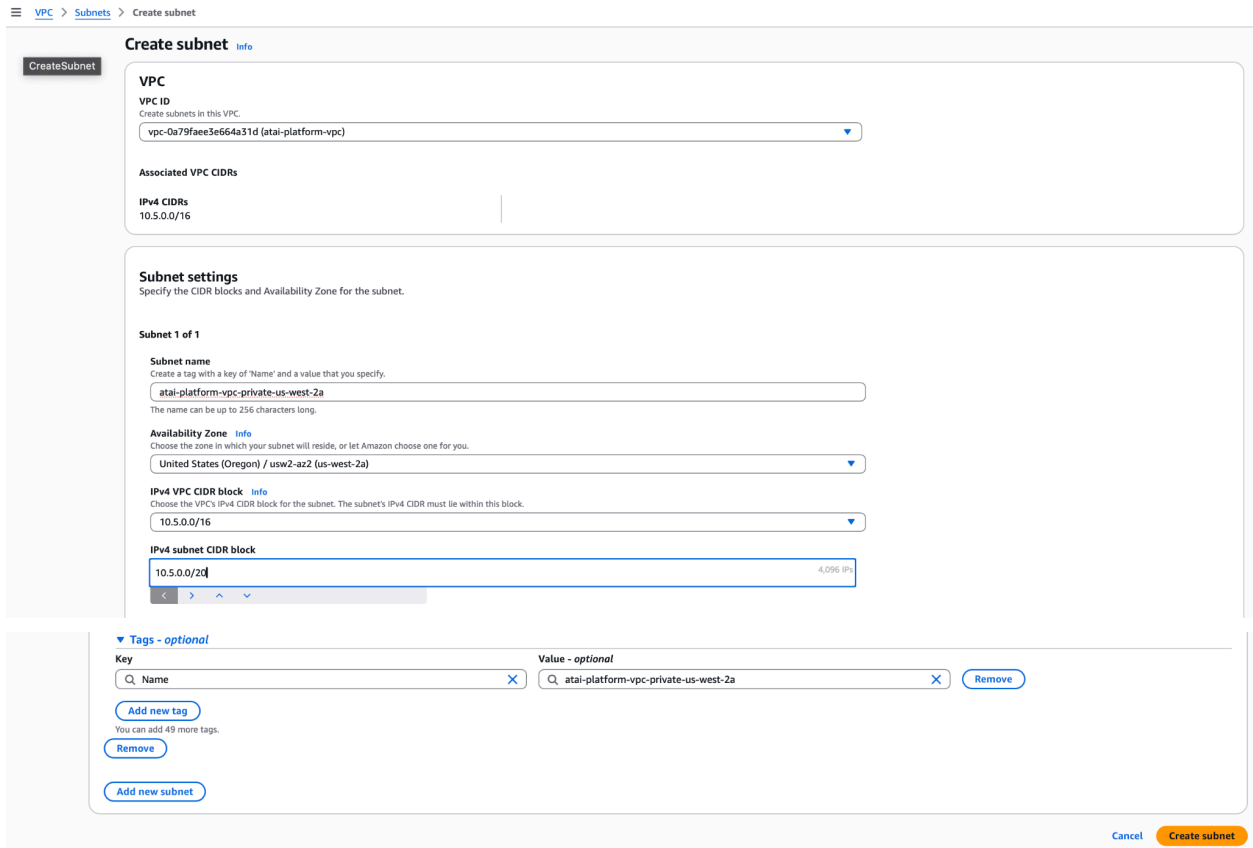
5. Repeat the process for both public subnets.

Step 4: Create Private Subnets

1. Go to VPC Dashboard → Go to Subnets → Create subnet



2. Subnet 1:
 - a. Select the VPC created in the Step 1
 - b. Name: atai-platform-vpc-private-us-west-2a
 - c. Availability Zone: us-west-2a (or your AZ1)
 - d. CIDR: 10.5.0.0/20
 - e. Click Create subnet

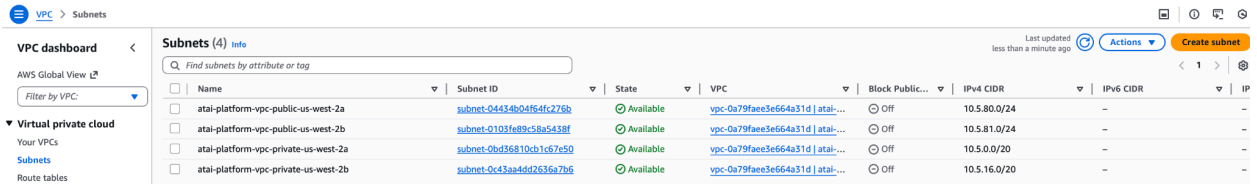


3. Subnet 2:
 - a. Select the VPC created in the Step 1
 - b. Name: atai-platform-vpc-private-us-west-2b
 - c. Availability Zone: us-west-2b (or your AZ2)
 - d. CIDR: 10.5.16.0/20

e. Click Create subnet

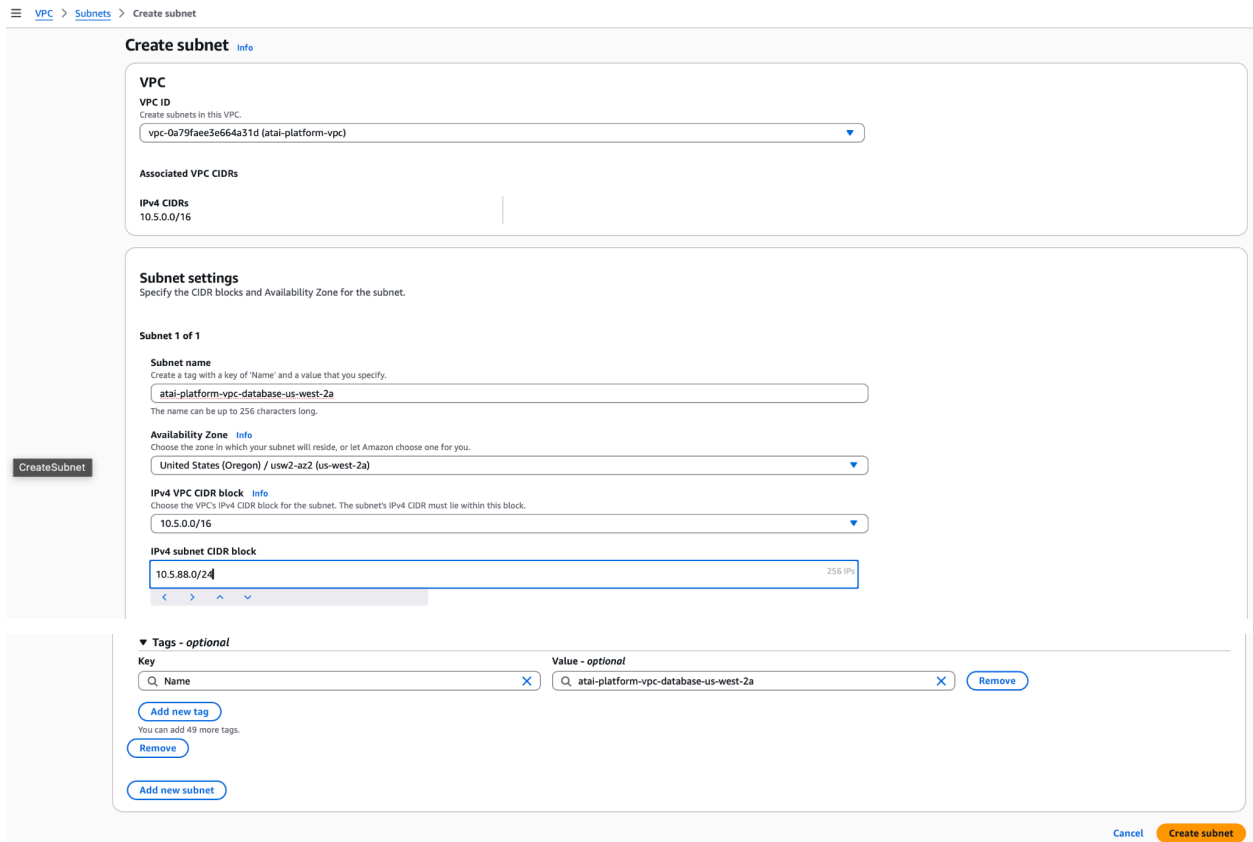
Step 5: Create Database Subnets

1. Go to VPC Dashboard → Go to Subnets → Create subnet



2. Subnet 1:

- Select the VPC created in the Step 1
- Name: atai-platform-vpc-database-us-west-2a
- Availability Zone: us-west-2a (or your AZ1)
- CIDR: 10.5.88.0/24
- Click Create subnet

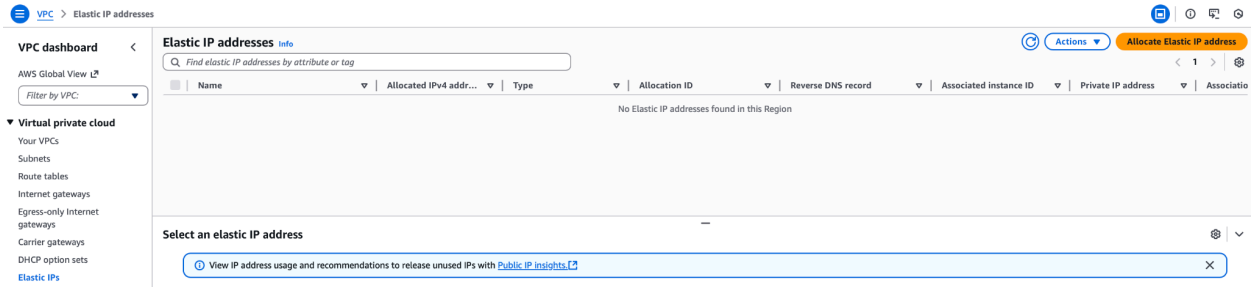


3. Subnet 2:

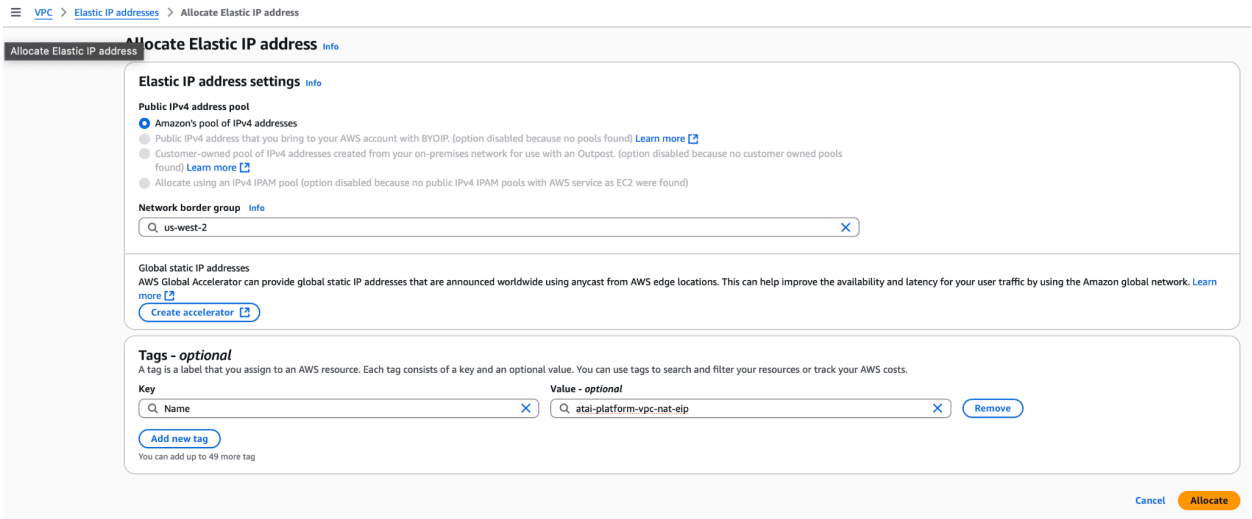
- Select the VPC created in the Step 1
- Name: atai-platform-vpc-database-us-west-2b
- Availability Zone: us-west-2b (or your AZ2)
- CIDR: 10.5.89.0/24
- Click Create subnet

Step 6: Allocate Elastic IP for NAT Gateway

- Go to VPC Dashboard → Go to Elastic IPs → Allocate Elastic IP address

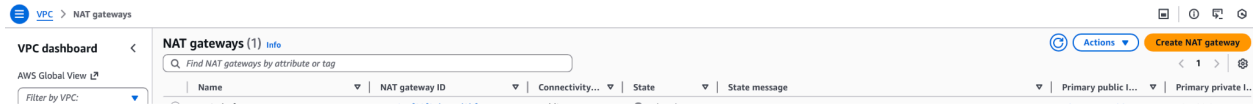


- Network border group: Select your region
- Public IPv4 address pool: Amazon's pool
- Add Name tag: atai-platform-vpc-nat-eip
- Click Allocate

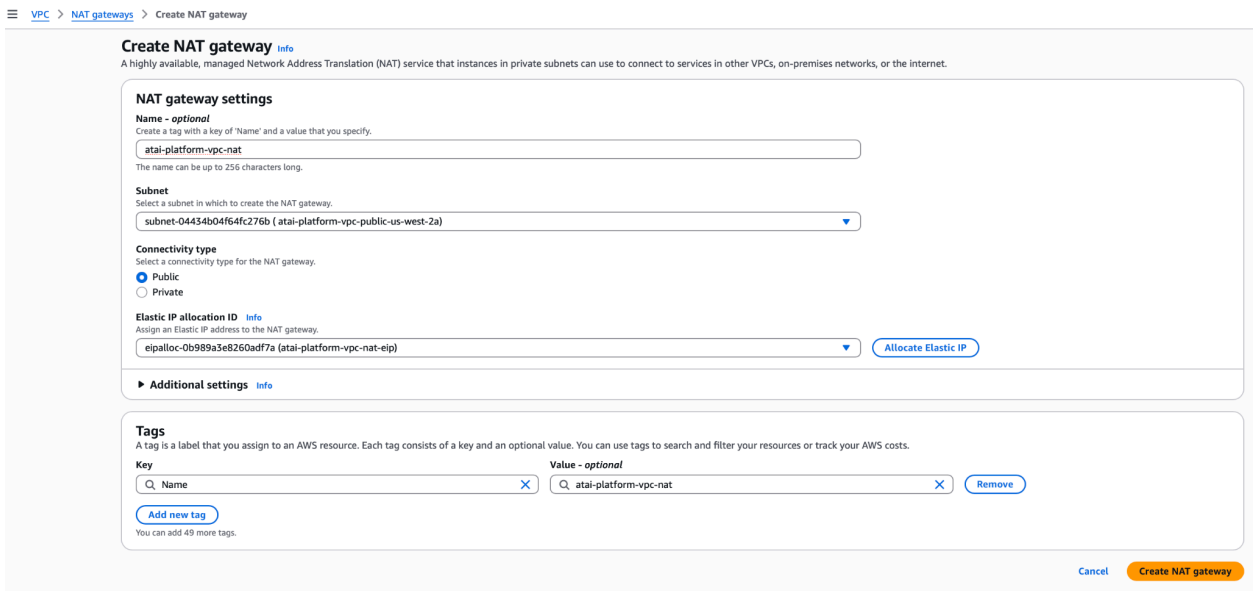


Step 7: Create NAT Gateway

1. Go to VPC Dashboard → Go to NAT Gateways → Create NAT gateway



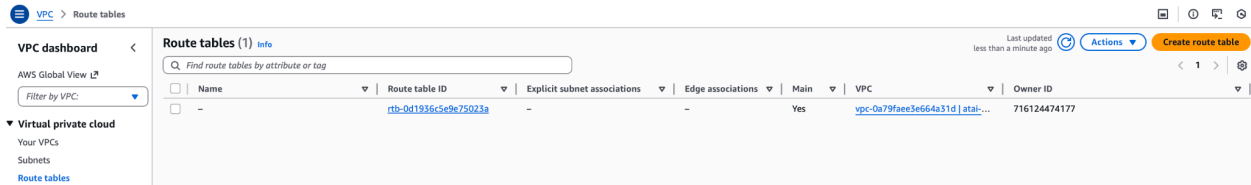
2. Name: atai-platform-vpc-nat
3. Subnet: Select first public subnet (10.5.80.0/24)
4. Elastic IP allocation ID: Select the EIP you just created in Step 6
5. Click Create NAT gateway (wait a few minutes)



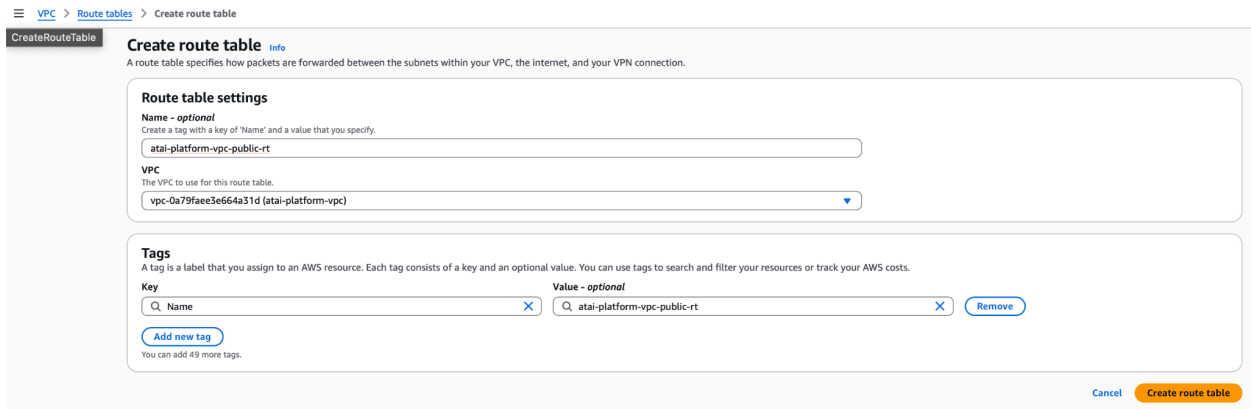
Step 8: Create and Configure Route Tables

Public Route Table:

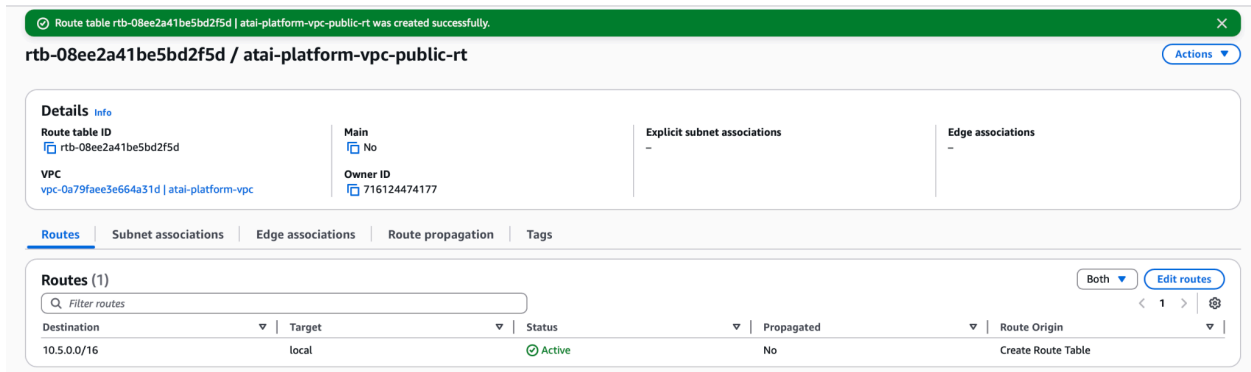
1. Go to VPC Dashboard → Route Tables → Create route table



2. Name: atai-platform-vpc-public-rt
3. VPC: Select your VPC from Step 1
4. Click Create route table



5. Routes → **Edit routes** → Add route:
 - a. Destination: 0.0.0.0/0
 - b. Target: Internet Gateway → select your IGW from Step 2
 - c. Save changes



[VPC](#) > [Route tables](#) > [rtb-08ee2a41be5bd2f5d](#) > Edit routes

Edit routes

Destination	Target	Status	Propagated	Route Origin	
10.5.0.0/16	local	Active	No	CreateRouteTable	
<input type="text" value="0.0.0.0/0"/>	<input type="text" value="local"/>				
	Internet Gateway		No	CreateRoute	<input type="button" value="Remove"/>
	<input type="text" value="igw-05499c47a963a31f8"/>				

6. Subnet associations → **Edit subnet associations:**
 - a. Select both public subnets → Save associations

[rtb-08ee2a41be5bd2f5d](#) / [atai-platform-vpc-public-rt](#)

Details Info

Route table ID rtb-08ee2a41be5bd2f5d	Main <input type="checkbox"/> No	Explicit subnet associations -	Edge associations -
VPC vpc-0a79faee3e664a31d atai-platform-vpc	Owner ID 716124474177		

[Routes](#) | [Subnet associations](#) | [Edge associations](#) | [Route propagation](#) | [Tags](#)

Explicit subnet associations (0)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
No subnet associations You do not have any subnet associations.			

[VPC](#) > [Route tables](#) > [rtb-08ee2a41be5bd2f5d](#) > Edit subnet associations

Edit subnet associations
Change which subnets are associated with this route table.

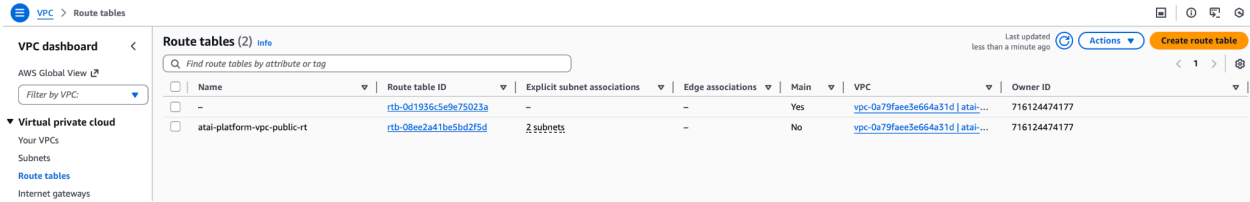
Available subnets (2/6)

<input type="checkbox"/>	Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route
<input checked="" type="checkbox"/>	atai-platform-vpc-public-us-west-2a	subnet-04434b04f64fc276b	10.5.80.0/24	-	Main
<input type="checkbox"/>	atai-platform-vpc-private-us-west-2a	subnet-0bd36810cb1c67e50	10.5.0.0/20	-	Main
<input type="checkbox"/>	atai-platform-vpc-private-us-west-2b	subnet-0c43aa4dd2636a7b6	10.5.16.0/20	-	Main
<input type="checkbox"/>	atai-platform-vpc-database-us-west-2a	subnet-06819ecc03094dea7	10.5.88.0/24	-	Main
<input type="checkbox"/>	atai-platform-vpc-database-us-west-2b	subnet-06bd3873d34243b83	10.5.89.0/24	-	Main
<input checked="" type="checkbox"/>	atai-platform-vpc-public-us-west-2b	subnet-0ebfa4a83c5ae95b	10.5.81.0/24	-	Main

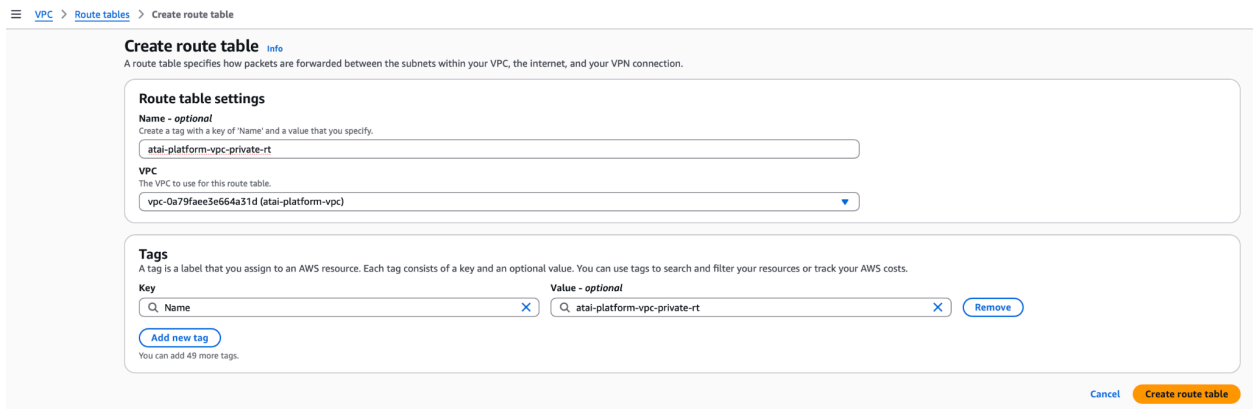
Selected subnets

Private Route Table:

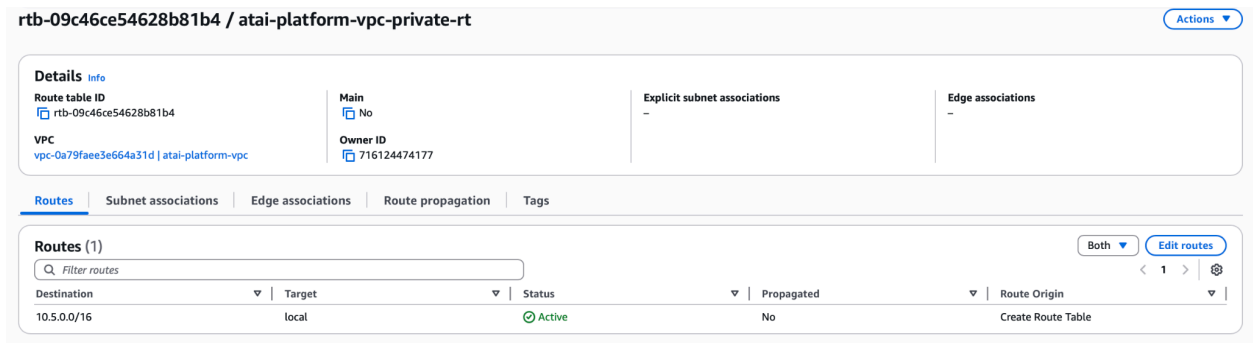
1. Go to VPC Dashboard → Route Tables → Create route table



2. Name: atai-platform-vpc-private-rt
3. VPC: Select your VPC from Step 1
4. Click Create route table



5. Routes → Edit routes → **Add route**:
 - a. Destination: 0.0.0.0/0
 - b. Target: NAT Gateway → select your NAT Gateway
 - c. Save changes



VPC > Route tables > rtb-09c46ce54628b81b4 > Edit routes

Edit routes

Destination	Target	Status	Propagated	Route Origin
10.5.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	NAT Gateway	-	No	CreateRoute

Buttons: Add route, Cancel, Preview, Save changes

6. Subnet associations → Edit subnet associations:
 - a. Select both private subnets → Save associations

rtb-09c46ce54628b81b4 / atai-platform-vpc-private-rt

RouteTableDetails

Details

Route table ID: rtb-09c46ce54628b81b4

VPC: vpc-0a79faee3e664a31d | atai-platform-vpc

Main: No

Owner ID: 716124474177

Explicit subnet associations: -

Edge associations: -

Routes | **Subnet associations** | Edge associations | Route propagation | Tags

Explicit subnet associations (0)

Find subnet association

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
No subnet associations You do not have any subnet associations.			

Buttons: Edit subnet associations

VPC > Route tables > rtb-09c46ce54628b81b4 > Edit subnet associations

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (2/6)

Filter subnet associations

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route
atai-platform-vpc-public-us-west-2a	subnet-04434b04f64fc276b	10.5.80.0/24	-	rtb-06
<input checked="" type="checkbox"/> atai-platform-vpc-private-us-west-2a	subnet-0bd36810cb1c67e50	10.5.0.0/20	-	Main
<input checked="" type="checkbox"/> atai-platform-vpc-private-us-west-2b	subnet-0c43aa4dd2636a7b6	10.5.16.0/20	-	Main
atai-platform-vpc-database-us-west-2a	subnet-06819ecd03094dea7	10.5.88.0/24	-	Main
atai-platform-vpc-database-us-west-2b	subnet-06bd3873d34243b83	10.5.89.0/24	-	Main
atai-platform-vpc-public-us-west-2b	subnet-0ebfa4aa83c5ae95b	10.5.81.0/24	-	rtb-06

Selected subnets

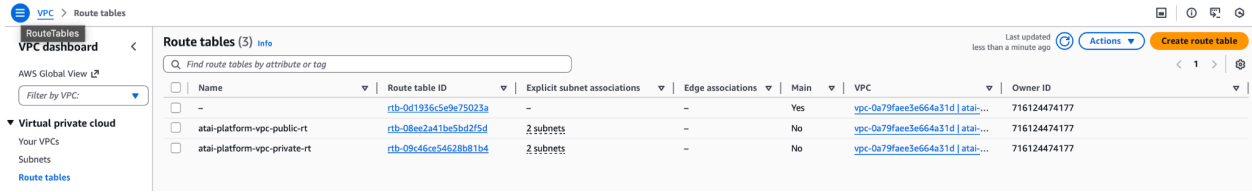
subnet-0bd36810cb1c67e50 / atai-platform-vpc-private-us-west-2a

subnet-0c43aa4dd2636a7b6 / atai-platform-vpc-private-us-west-2b

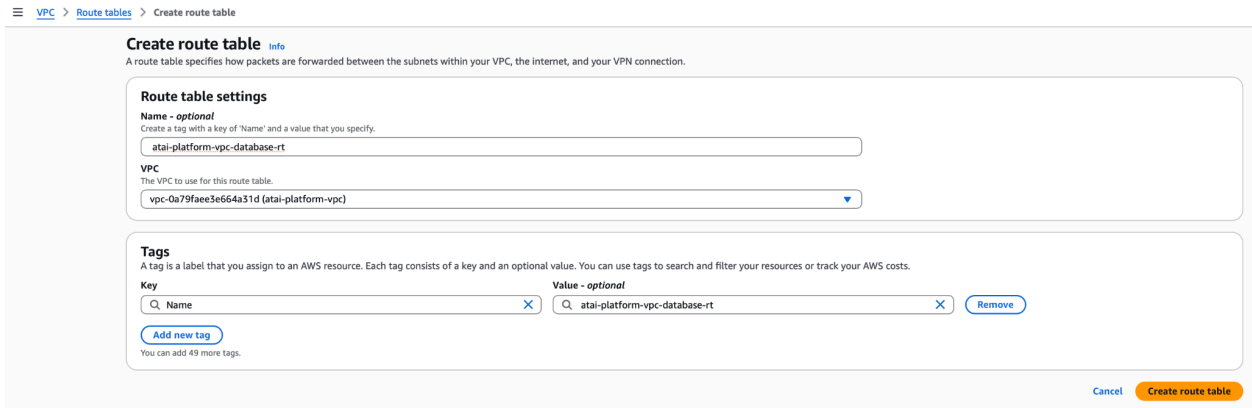
Buttons: Cancel, Save associations

Database Route Table:

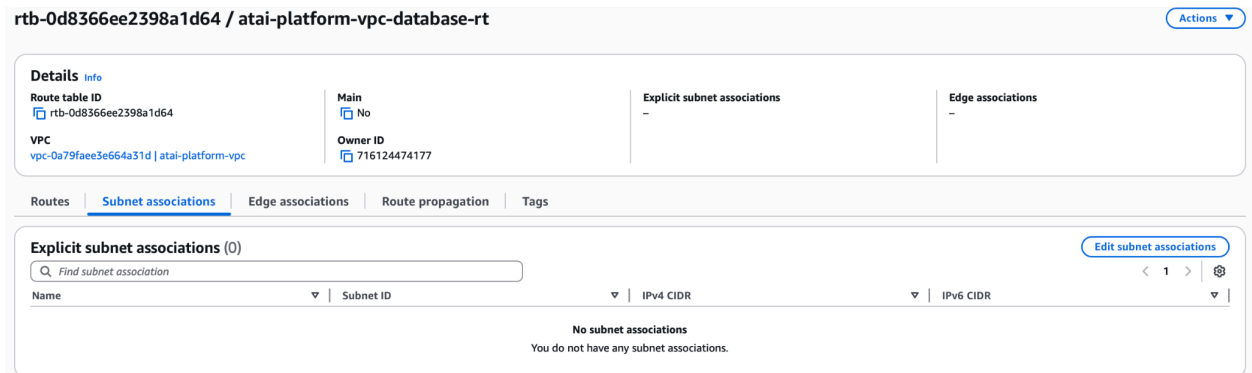
1. Go to VPC Dashboard → Route Tables → Create route table



2. Name: atai-platform-vpc-database-rt
3. VPC: Select your VPC from Step 1
4. Click Create route table



5. (No extra routes needed - isolated)
6. Subnet associations → Edit subnet associations:
 - a. Select both database subnets → Save associations



Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (2/6)

Filter subnet associations

<input type="checkbox"/>	Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route
<input type="checkbox"/>	atal-platform-vpc-public-us-west-2a	subnet-04454b04f64fc276b	10.5.80.0/24	-	rtb-06
<input type="checkbox"/>	atal-platform-vpc-private-us-west-2a	subnet-0bd36810cb1c67e50	10.5.0.0/20	-	rtb-05
<input type="checkbox"/>	atal-platform-vpc-private-us-west-2b	subnet-0c43aa4dd2636a7b6	10.5.16.0/20	-	rtb-05
<input checked="" type="checkbox"/>	atal-platform-vpc-database-us-west-2a	subnet-06819ecd03094dea7	10.5.88.0/24	-	Main (
<input checked="" type="checkbox"/>	atal-platform-vpc-database-us-west-2b	subnet-06bd3873d54243b83	10.5.89.0/24	-	Main (
<input type="checkbox"/>	atal-platform-vpc-public-us-west-2b	subnet-0ebfa4aa83c5ae95b	10.5.81.0/24	-	rtb-06

Selected subnets

subnet-06bd3873d54243b83 / atai-platform-vpc-database-us-west-2b X subnet-06819ecd03094dea7 / atai-platform-vpc-database-us-west-2a X

Cancel Save associations

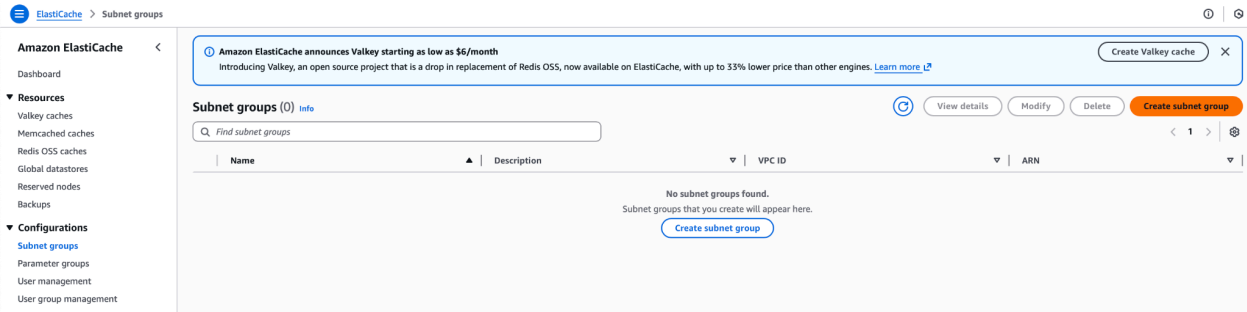
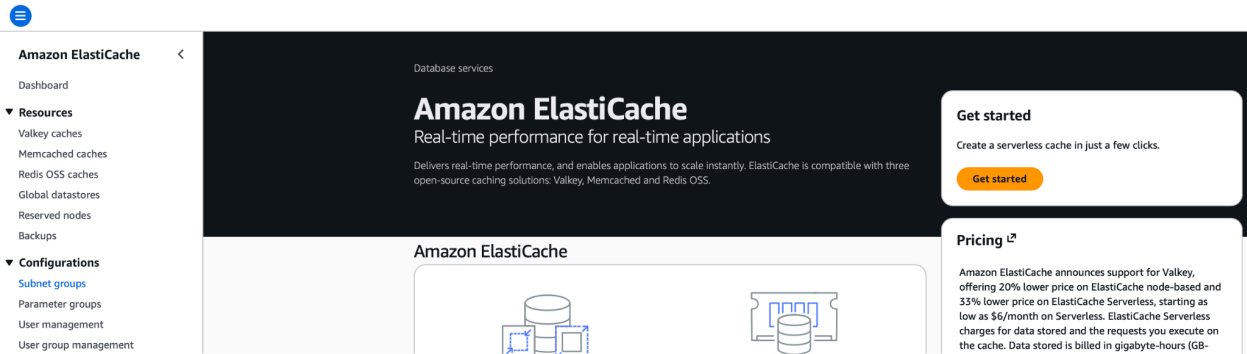
Valkey clusters configuration

Prerequisites

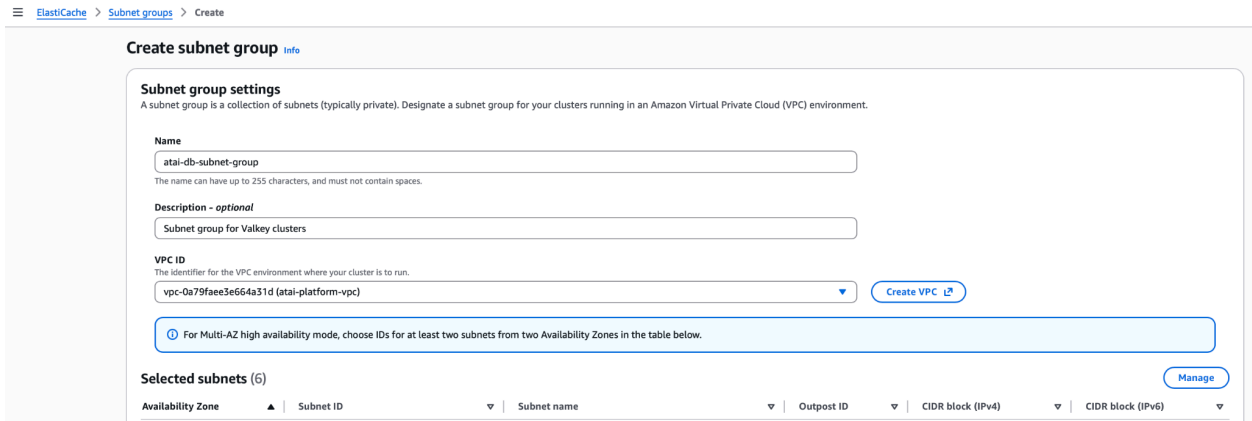
1. VPC with database subnets
2. (Optional) Security group allowing access from EKS pods
3. Subnet group for database subnets

Step 1: Create ElastiCache Subnet Group (if not existing)

1. Go to ElastiCache → Subnet groups → Create subnet group



2. Name: atai-db-subnet-group (or your name)
3. Description: Subnet group for Valkey clusters (single AZ for low latency)
4. VPC: Select your VPC



5. In the section **Selected subnets** click on **Manage**
 - a. Availability Zones: Select only the first AZ (e.g., us-west-2a) and click on **Choose**
 - i. Important: Use only one AZ to reduce latency
 - ii. Note: EKS Managed node groups will use a private subnet in the same AZ (different subnet CIDR)

Manage subnets ✕

Add or remove subnets from the table below.

Subnets (1/6) ↻

< 1 2 >

<input type="checkbox"/>	Availability Zone ▲	Subnet ID ▼	Subnet name ▼	Outp
<input type="checkbox"/>	us-west-2a	subnet-0bd36810cb1c67e50	atai-platform-vpc-private-us-west-2a	
<input type="checkbox"/>	us-west-2a	subnet-04434b04f64fc276b	atai-platform-vpc-public-us-west-2a	
<input checked="" type="checkbox"/>	us-west-2a	subnet-06819ecd03094dea7	atai-platform-vpc-database-us-west-2a	
<input type="checkbox"/>	us-west-2b	subnet-0ebfa4aa83c5ae95b	atai-platform-vpc-public-us-west-2b	
<input type="checkbox"/>	us-west-2b	subnet-06bd3873d34243b83	atai-platform-vpc-database-us-west-2b	

Cancel

Choose

6. Subnets: Select only the first database subnet (e.g., 10.5.88.0/24 in us-west-2a)
 - a. Do not select the second database subnet
7. Click Create

☰ [ElastiCache](#) > [Subnet groups](#) > Create

Subnet group settings

A subnet group is a collection of subnets (typically private). Designate a subnet group for your clusters running in an Amazon Virtual Private Cloud (VPC) environment.

Name

The name can have up to 255 characters, and must not contain spaces.

Description - optional

VPC ID
The identifier for the VPC environment where your cluster is to run.
 [Create VPC ↗](#)

ⓘ For Multi-AZ high availability mode, choose IDs for at least two subnets from two Availability Zones in the table below.

Selected subnets (1) [Manage](#)

Availability Zone ▲	Subnet ID ▼	Subnet name ▼	Outpost ID ▼	CIDR block (IPv4) ▼	CIDR block (IPv6) ▼
us-west-2a	subnet-06819ecd03094dea7	atai-platform-vpc-database-us-west-2a		10.5.88.0/24	-

Tags
You can use tags to search and filter your subnet groups, or track your AWS costs.
No tags associated with the subnet group.
[Add new tag](#)
You can add 50 more tags.

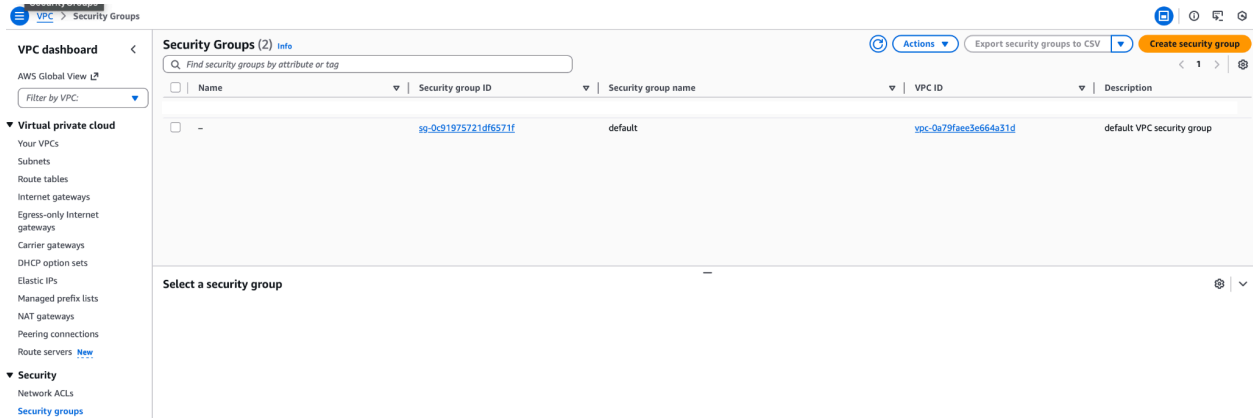
[Cancel](#) [Create](#)

Important Notes:

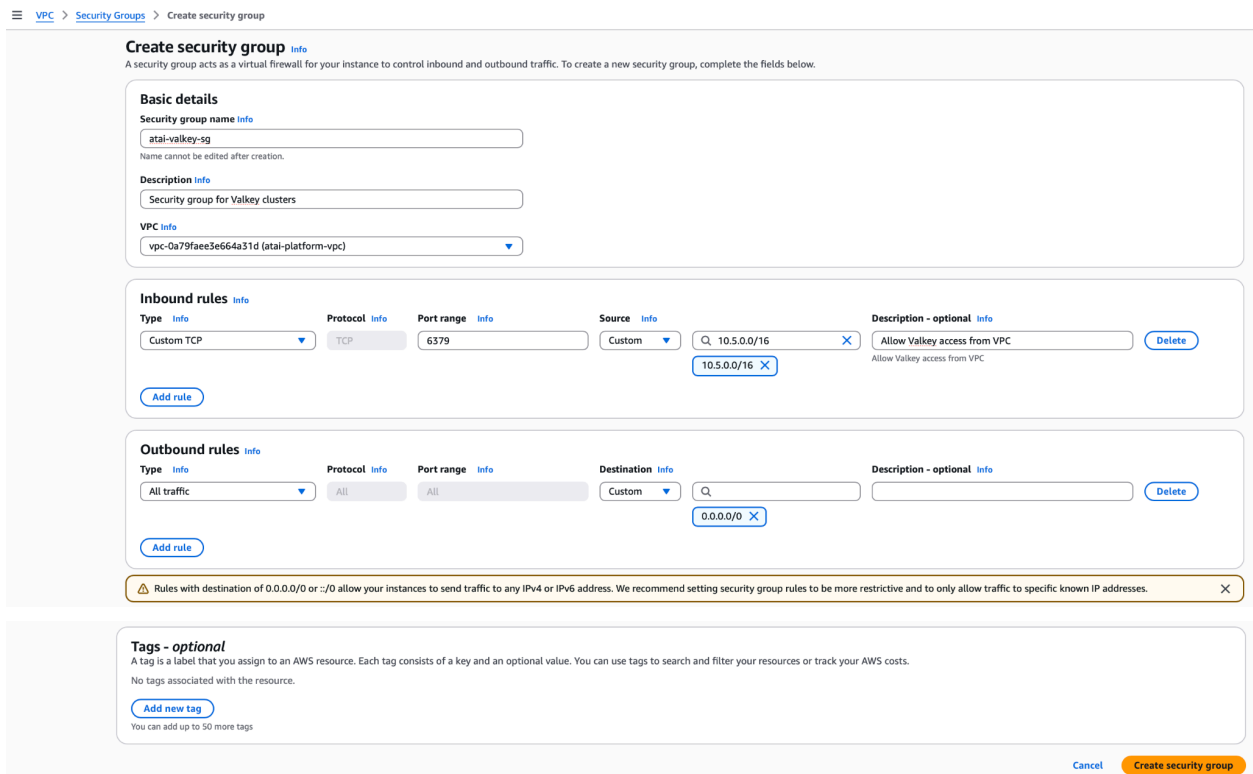
1. Single AZ deployment reduces cross-AZ network latency
2. All Valkey clusters will be created in this single AZ
3. EKS managed node groups should use a private subnet in the same AZ (e.g., 10.5.0.0/20 in us-west-2a) for optimal latency
4. Example: If you use us-west-2a for database subnet 10.5.88.0/24, use us-west-2a for private subnet 10.5.0.0/20 for node groups

Step 2: Create Security Group (if not existing)

1. Go to VPC → Security Groups → Create security group



2. Name: atai-valkey-sg (or your name)
3. Description: Security group for Valkey clusters
4. VPC: Select your VPC from section VPC configuration Step 1
5. Inbound rules: Add rule:
 - a. Type: Custom TCP
 - b. Port: 6379
 - c. Source: Custom → Enter your VPC CIDR (e.g., 10.5.0.0/16)
 - d. Description: Allow Valkey access from VPC

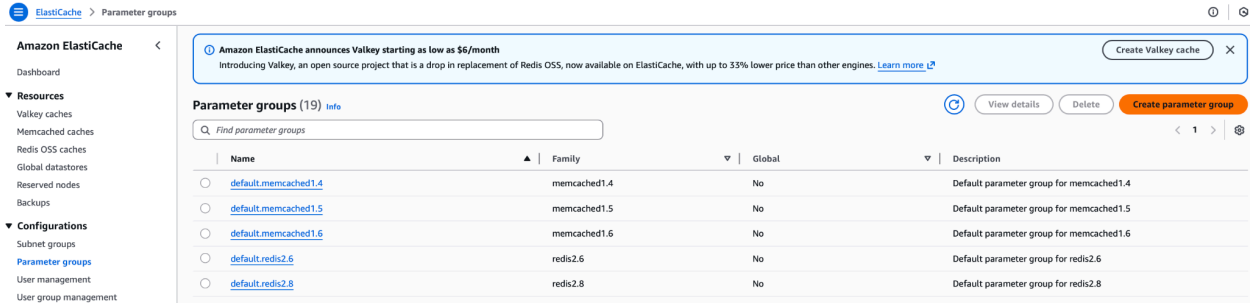


Note: For initial setup, opening to the VPC CIDR simplifies connectivity. Later, restrict to specific security groups (e.g., EKS node group security group) for tighter security.

6. Click Create security group

Step 3: Create Parameter Group for Valkey 8.0

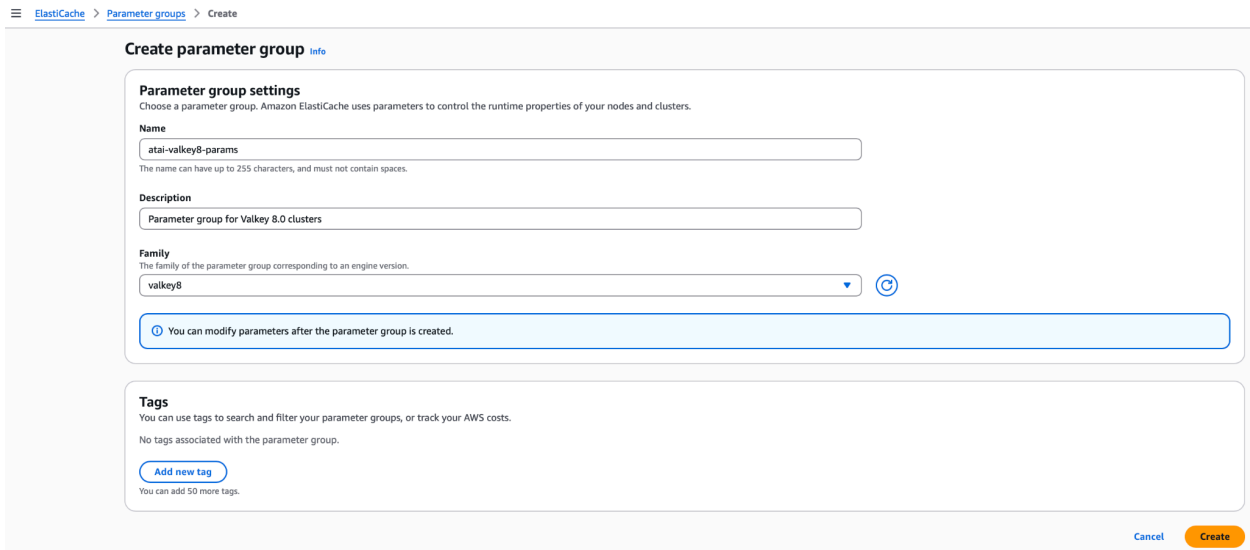
7. Go to ElastiCache → Parameter groups → Create parameter group



8. Group name: atai-valkey8-params

9. Description: Parameter group for Valkey 8.0 clusters

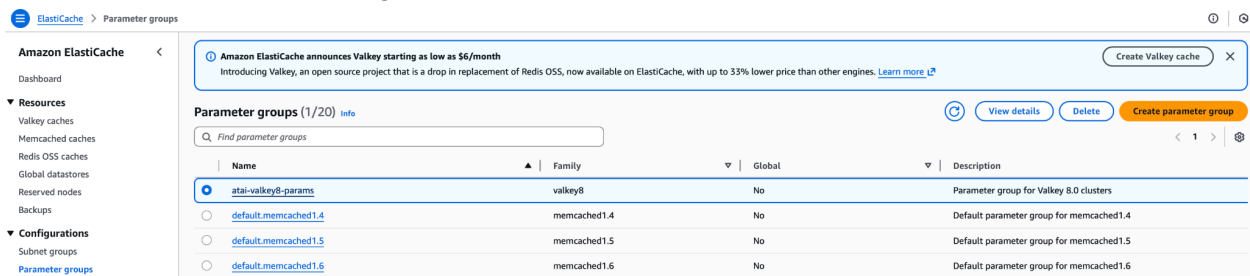
10. Parameter group family: valkey8



11. Click Create

Step 3.1 After creation, edit the parameter group:

1. Select the parameter group → Edit



2. Click on **Edit parameter values**

Amazon ElastiCache > Parameter groups > atai-valkey8-params

atai-valkey8-params [Info](#) [Delete](#)

Parameter group settings

Name atai-valkey8-params	Description Parameter group for Valkey 8.0 clusters	Family valkey8	Global No
ARN arn:aws:elasticache-west-2:716124474177:parametergroup:atai-valkey8-params			

Parameters (289) [Info](#) [Reset to defaults](#) [Edit parameter values](#)

Find parameters

Name	Allowed values	Is modifiable	Node type	Value	Source	Type	Change type	Description
act-pubsub-default	resetchannels,allch...	Yes	All	allchannels	system	string	immediate	Default pub...

3. In search bar type **cluster-enabled**

4. With the dropdown menu set the value to **yes**

atai-valkey8-params [Info](#) [Delete](#)

Parameter group settings

Name atai-valkey8-params	Description Parameter group for Valkey 8.0 clusters	Family valkey8	Global No
ARN arn:aws:elasticache-west-2:716124474177:parametergroup:atai-valkey8-params			

Parameters (1/59) [Info](#) [Cancel](#) [Preview changes](#) [Save changes](#)

Find parameters: cluster-enabled 1 match

Name	Allowed values	Is modifiable	Node type	Value	Source	Type	Change type	Description
cluster-enabled	yes,no	Yes	All	yes	system	string	requires-reboot	Enable cluster mode

5. Click Save changes

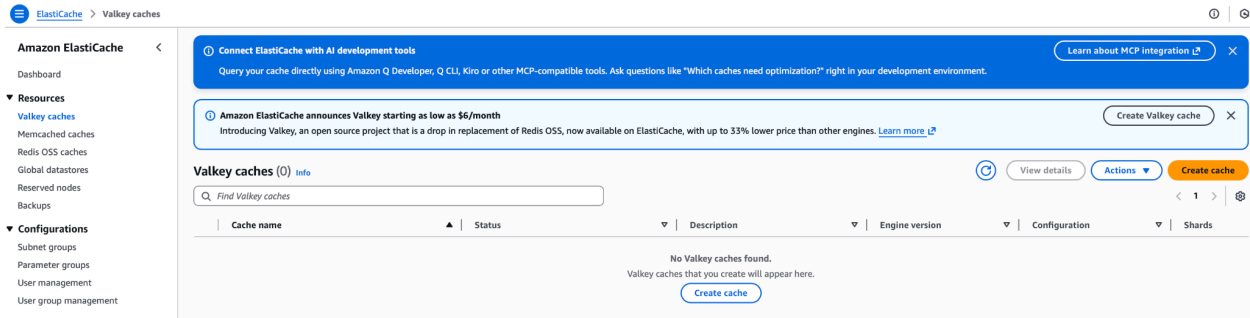
Important: **The cluster-enabled = yes parameter is required for cluster mode to work.** Without it, the clusters won't function properly in cluster mode, even if you enable cluster mode in the cluster settings.

Step 4: Create Valkey Clusters

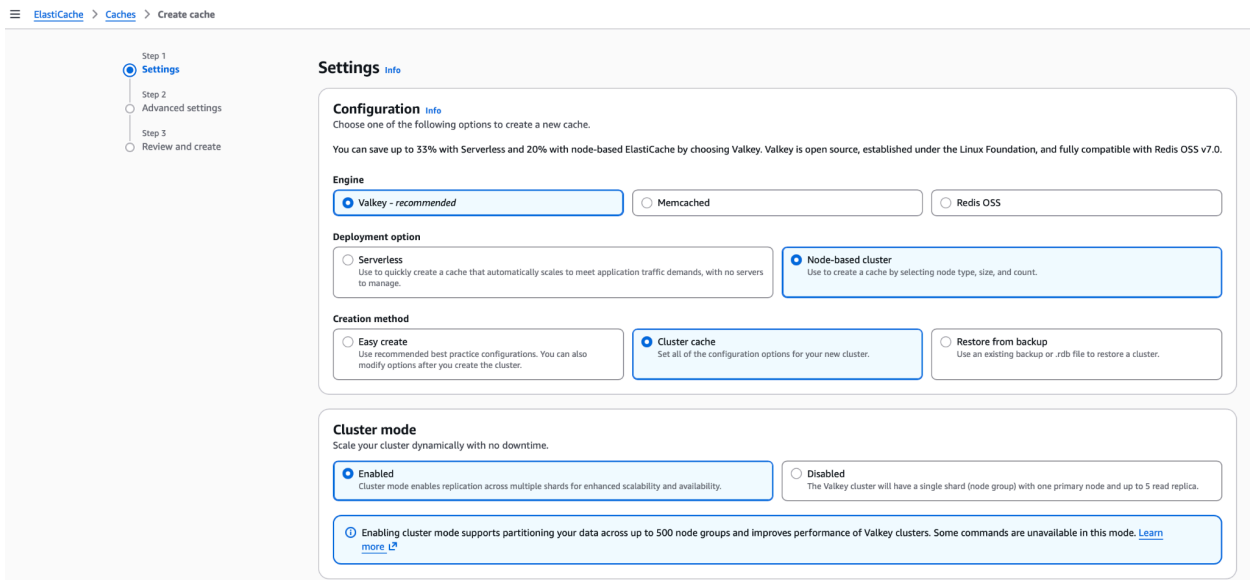
For each cluster, follow these steps.

Cluster 1: registry

1. ElastiCache → Valkey caches → **Create cache**



2. Engine: Valkey
3. Deployment option: Node-based cluster
4. Creation method: Cluster cache
5. Cluster mode: Enable



6. Cluster info
 - a. Name: atai-registry-service
 - b. Description: Registry Service Valkey cluster
7. Location
 - a. Location: AWS Cloud
 - b. Muti-AZ: Disable (to reduce latency)

Cluster info
Use the following options to configure the cluster.

Name
atai-access-manager-service
The name can have up to 40 characters, and must not contain spaces.

Description - optional
Access Manager Service Valkey cluster

Location
Choose whether to host the cluster in the AWS Cloud or on premises.

Location

AWS Cloud
Use the AWS Cloud for your ElastiCache instances.

On premises
Create your ElastiCache instances on an Outpost (through AWS Outposts). You need to create a subnet ID on an Outpost first.

Multi-AZ

Enable
Multi-AZ provides enhanced high availability through automatic failover to a read replica, cross AZs, in case of a primary node failure.

Auto-failover

Enable
ElastiCache Auto Failover provides enhanced high availability through automatic failover to a read replica in case of a primary node failure.

ⓘ Disabling ElastiCache Multi-AZ on your cluster reduces your fault tolerance. In the unlikely event of an Availability Zone failure or loss of network connectivity, your cluster will become unavailable. [Learn more](#)

8. Cache setting
 - a. Engine version: 8.0
 - b. Port: 6379
 - c. Parameter group: Select the parameter group created in the Step 3 of this section
 - d. Node type: cache.t4g.small
 - e. Number of shards: 1
 - f. Replicas per shard: 0

Cache settings
Use the following options to configure the cluster.

Engine version
Version compatibility of the Valkey engine that will run on your nodes.
8.0

Port
The port number that nodes accept connections on.
6379

Parameter groups
Parameter groups control the runtime properties of your nodes and clusters.
atai-valkey8-params

Node type
The type of node to be deployed and its associated memory size.
cache.t4g.small
1.37 GiB memory Up to 5 Gigabit network performance

Number of shards
Enter the number of shards in this cluster, from 1 to 500.
1

Replicas per shard
Enter the number of replicas for each shard, from 0 to 5.
0

ⓘ Multi-AZ can not be enabled when the number of replicas is set to 0. Select one or more replicas to enable Multi-AZ.

9. Connectivity:

a. Subnet groups: Select your database subnet group (single AZ) from Step 1

Connectivity

Choose the IP version(s) this cluster will support. Then select an existing subnet group or create a new one.

Network type
Choose between IPv4, dual stack and IPv6

IPv4
Your resources will communicate only over the IPv4 protocol.

Subnet groups

Choose existing subnet group Create a new subnet group

Subnet groups
A collection of subnets that you can designate for your clusters running in an Amazon VPC.

atal-db-subnet-group (vpc-0a79faee3e664a31d)

Associated subnets (1)

Availability Zone	Subnet ID	CIDR block (IPv4)
us-west-2a	subnet-06819ecd03094dea7	10.5.88.0/24

b. Use the default Availability Zone placements -> Next

Availability Zone placements

Use the following fields to configure placements for Availability Zones.

Slots and keyspaces
Distribution of the 16,384 cluster keyspaces slots across shards.

Equal distribution

Availability Zone placements
By locating nodes in different Availability Zones, you reduce the chance that a failure in one Availability Zone, such as a power outage, will cause your entire system to fail. Choose Specify Availability Zones if you want to specify Availability Zones for cluster nodes.

No preference

Shards	Slots/keyspaces	Primary
Shard 1	Equal distribution	No preference

Cancel **Next**

10. Enable encryption at rest and Encryption in transit

Step 1 Settings

Step 2 **Advanced settings**

Step 3 Review and create

Advanced settings [Info](#)

Security

Use the following section to configure network security and data security for your cluster.

Encryption at rest

Enable
Enables encryption of data stored on disk.

Encryption key
The master key that will be used to protect the key used to encrypt data at rest.

Default key
An AWS owned key will be used for encryption.

Customer managed CMK
Select a customer managed key.

Encryption in transit [Info](#)

Enable
Enables encryption of data that moves between the service and client.

Transit encryption mode
Required

In Required mode, the cluster will support only encrypted TLS connections. Transit encryption mode can be modified after the cluster has been created. [Learn more](#)

11. For access control choose **AUTH default user access**

- a. Important: Save this auth token securely. It is required later to communicate with your Valkey cluster.
- b. Security recommendation: Create a unique auth token per Valkey cluster (do not reuse the same token across clusters).
- c. Store each token in a secure location such as AWS Secrets Manager.

Access control

Provides the ability to configure authenticating and authorizing access.

AUTH default user access

AUTH token

The AUTH token used for the cluster.

..... Show token

At least 16 characters and a maximum of 128 characters, which can be any printable ASCII character except for ' (blank space), " (quotation mark), /, and @.

12. Security groups

- a. Click on **Manage**

Selected security groups (0)

A security group acts like a firewall that controls network access to your clusters.

Manage

Group ID [↗](#) | Name

No selected security groups
Add security groups by choosing the Manage button.

Manage

- b. Select the security group created in Step 2
- c. Click on **Choose**

Manage security groups



Security groups (1/2)



Find security groups

<input type="checkbox"/>	Security group ID ↗	Security group name	Description
<input checked="" type="checkbox"/>	sg-0a973800a58d354aa	atai-valkey-sg	Security group for Valkey clusters
<input type="checkbox"/>	sg-0c91975721df6571f	default	default VPC security group

Cancel

Choose

13. Use the default configuration for **Backups, Maintenance, Logs and Tags.**

Backup
You can use backups to restore a cluster or seed a new cluster. The backup consists of the cluster's metadata, along with all of the data in the cluster.

Enable automatic backups
ElastiCache will automatically create a daily backup of a set of replicas.

Backup retention period
The number of days for which automated backups are retained before they're automatically deleted.

1

Backup window
The daily time range during which automatic backups start if they're enabled.

No preference
 Specify backup window

Maintenance
Configure maintenance settings for the cluster.

Maintenance window
Specify the time range (UTC) for updates such as patching an operating system, updating drivers, and installing software or patches.

No preference
 Specify maintenance window

Auto upgrade minor versions
 Enable
Automatically schedule cluster upgrade to the latest minor version, once it becomes available. Cluster upgrade will only be scheduled during the maintenance window.

Topic for Amazon SNS notification
Choose an SNS topic from the list, or enter the Amazon Resource Name (ARN) for an existing topic. If no topic is chosen, no notifications are sent.

Disable notifications

Logs
Specify whether to provide the Valkey slow logs or engine logs.

Slow logs
 Enable
Provide the slow log for queries that exceed a specified runtime.

Engine logs
 Enable
Provide the engine log for the cluster.

Tags
You can use tags to search and filter your clusters, or track your AWS costs.

No tags associated with the cluster.

[Add new tag](#)
You can add 50 more tags.

Cancel [Previous](#) [Next](#)

14. Click Next

15. Review your configuration

ElastiCache > Caches > Create cache

Step 1 Settings
Step 2 Advanced settings
Step 3 Review and create

Review and create [info](#)

Step 1: Settings [Edit](#)

Cluster info
Use the following options to configure the cluster.

Name atai-access-manager-service	Description Access Manager Service Valkey cluster
--	---

Location
Choose whether to host the cluster in the AWS Cloud or on premises.

Location aws-cloud	Cluster mode Enabled
------------------------------	--------------------------------

Cache settings
Use the following options to configure the cluster.

Engine Valkey	Engine version 8.0	Port 6379
Parameter groups atai-valkey8-params	Node type cache.t4g.small	Number of shards 1
Replicas per shard 0		

Connectivity
Choose the IP version(s) this cluster will support. Then select an existing subnet group or create a new one.

Network type IPv4	Subnet group atai-db-subnet-group	Availability Zones us-west-2a
-----------------------------	---	---

Step 2: Advanced settings [Edit](#)

Security
Use the following section to configure network security and data security for your cluster.

Security groups sg-0a973800a58d354aa	Encryption at rest Enabled	Encryption key default-key
Encryption in transit Enabled	Access control redis-auth-token	AUTH token UQwFeJz2KNpodmMWG7]
Transit encryption mode Required		

Backup
You can use backups to restore a cluster or seed a new cluster. The backup consists of the cluster's metadata, along with all of the data in the cluster.

Automatic backups Enabled	Backup retention period 1 day	Backup window No preference
-------------------------------------	---	---------------------------------------

Maintenance
Specify the time range (UTC) for updates such as patching an operating system, updating drivers, and installing software or patches.

Maintenance window No preference	Auto upgrade minor versions Enabled	Topic for Amazon SNS notification SNS topic notifications disabled
--	---	--

Logs
Specify whether to provide the Valkey slow logs or engine logs.

Slow logs Disabled	Engine logs Disabled
------------------------------	--------------------------------

Tags
You can use tags to search and filter your clusters, or track your AWS costs.

Key	Value
No tags found. Tags that you create will appear here.	

[Cancel](#) [Previous](#) [Create](#)

16. Click on **Create**

⚠ Store your Valkey host, port, and auth token in a secure location. You will need them later in the *atai-platform* prerequisites, Step 3.2: Kubernetes Secrets Generation. These values will be referenced as:

None

```
[atai-platform-registry-service-backend]
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

Note: The **atai-registry-service** secrets will be required for following services:

- gpq-node-newton-model-c23
- gpq-node-newton-model-omega
- api-service-backend
- api-service-health-node
- lens-node-worker-node
- lens-node-service-backend
- lens-service-db-backend

These values will be referenced as:

None

```
[atai-platform-gpq-node-newton-model-c23]
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

```
[atai-platform-gpq-node-newton-model-omega]
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

```
[atai-platform-api-service-backend]
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

```
[atai-platform-api-service-health-node]
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

```
[atai-platform-lens-node-worker-node]
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```


```
[atai-platform-lens-node-service-backend]
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

```
[atai-platform-lens-service-db-backend]
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

Cluster 2: access-manager

Same as Cluster 1, but:

1. Name: `atai-access-manager-service`
2. Auth token: Generate a new unique token (different from previous clusters)

 Store your Valkey host, port, and token in a secure location. You will need them later in the `atai-platform` prerequisites, Step 3.2: Kubernetes Secrets Generation.


Note: The secrets from `atai-registry-service` will still be required for the following secret. These values will be referenced as:

```
None
[atai-platform-access-manager-service-backend]
ACCESS_MANAGER_SERVICE_IP_ADDRESS=clustercfg.atai-access-manager-service.<HASH>
.<AWS_REGION>.cache.amazonaws.com
ACCESS_MANAGER_SERVICE_PASSWORD=<ACCESS_MANAGER_SERVICE_PASSWORD>
ACCESS_MANAGER_SERVICE_PORT=6379
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

Cluster 3: api-events

Same as Cluster 1, but:

3. Name: atai-api-events-service
4. Auth token: Generate a new unique token (different from previous clusters)

 Store your Valkey host, port, and token in a secure location. You will need them later in the atai-platform prerequisites, Step 3.2: Kubernetes Secrets Generation.


Note: The secrets from **atai-registry-service** will still be required for the following secret. These values will be referenced as:

```
None
[atai-platform-api-events-service-backend]
API_EVENTS_SERVICE_IP_ADDRESS=clustercfg.atai-api-events-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
API_EVENTS_SERVICE_PASSWORD=<API_EVENTS_SERVICE_PASSWORD>
API_EVENTS_SERVICE_PORT=6379
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

Cluster 4: dfc (10 shards)

Same as Cluster 1, but:

1. Name: atai-dfc-service
2. Number of shards: 10 (instead of 1)
3. Replicas per shard: 0 (same for the other 10 shard caches)
4. Auth token: Generate a new unique token (different from previous clusters)

 Store your Valkey host, port, and token in a secure location. You will need them later in the atai-platform prerequisites, Step 3.2: Kubernetes Secrets Generation.

Note: The secrets from **atai-registry-service** will still be required for the following secret. These values will be referenced as:


```
None
[atai-platform-dfc-service-backend]
DFC_SERVICE_IP_ADDRESS=clustercfg.atai-dfc-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
DFC_SERVICE_PASSWORD=<DFC_SERVICE_PASSWORD>
DFC_SERVICE_PORT=6379
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-stage-registry-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
```

```
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

Cluster 5: file (10 shards)

Same as Cluster 3, but:

1. Name: atai-{environment}-file-service
2. Number of shards: 10 (instead of 1)
3. Replicas per shard: 0 (same for the other 10 shard caches)
4. Auth token: Generate a new unique token (different from previous clusters)

 Store your Valkey host, port, and token in a secure location. You will need them later in the atai-platform prerequisites, Step 3.2: Kubernetes Secrets Generation.


Note: The secrets from **atai-registry-service** will still be required for the following secret. These values will be referenced as:

```
None
[atai-platform-file-service-worker-node]
FILE_SERVICE_IP_ADDRESS=clustercfg.atai-file-service.<HASH>.<AWS_REGION_CODE>.c
ache.amazonaws.com
FILE_SERVICE_PASSWORD=<FILE_SERVICE_PASSWORD>
FILE_SERVICE_PORT=6379
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

Cluster 6: gpq (10 shards)

Same as Cluster 3, but:

1. Name: atai-gpq-service
2. Number of shards: 10 (instead of 1)
3. Replicas per shard: 0 (same for the other 10 shard caches)
4. Auth token: Generate a new unique token (different from previous clusters)

 Store your Valkey host, port, and token in a secure location. You will need them later in the atai-platform prerequisites, Step 3.2: Kubernetes Secrets Generation.

Note 1: The gpq secrets will be required for follow services:

- gpq-service-backend
- gpq-service-event-router

Note 2: The secrets from **atai-registry-service** will still be required for the following secret. These values will be referenced as:


```
None
[atai-platform-gpq-service-event-router]
GPQ_SERVICE_IP_ADDRESS=clustercfg.atai-gpq-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
GPQ_SERVICE_PASSWORD=<GPQ_SERVICE_PASSWORD>
GPQ_SERVICE_PORT=6379
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379

[atai-platform-gpq-service-backend]
GPQ_SERVICE_IP_ADDRESS=clustercfg.atai-gpq-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
GPQ_SERVICE_PASSWORD=<GPQ_SERVICE_PASSWORD>
GPQ_SERVICE_PORT=6379
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

Cluster 7: health

Same as Cluster 1, but:

1. Name: atai-health-service
2. Auth token: Generate a new unique token (different from previous clusters)

 Store your Valkey host, port, and token in a secure location. You will need them later in the atai-platform prerequisites, Step 3.2: Kubernetes Secrets Generation.

Note: The secrets from **atai-registry-service** will still be required for the following secret. These values will be referenced as:


```
None
[atai-platform-health-service-backend]
HEALTH_SERVICE_IP_ADDRESS=clustercfg.atai-health-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
HEALTH_SERVICE_PASSWORD=<HEALTH_SERVICE_PASSWORD>
HEALTH_SERVICE_PORT=6379
```

```
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

Cluster 8: lens (10 shards)

Same as Cluster 3, but:

1. Name: atai-lens-service
2. Number of shards: 10 (instead of 1)
3. Replicas per shard: 0 (same for the other 10 shard caches)
4. Auth token: Generate a new unique token (different from previous clusters)

 Store your Valkey host, port, and token in a secure location. You will need them later in the atai-platform prerequisites, Step 3.2: Kubernetes Secrets Generation.

Note: The secrets from **atai-registry-service** will still be required for the following secret. These values will be referenced as:

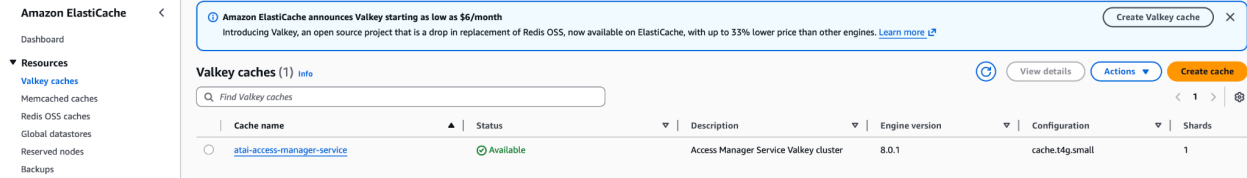
```
None
[atai-platform-lens-service-backend]
LENS_SERVICE_IP_ADDRESS=clustercfg.atai-lens-service.<HASH>.<AWS_REGION_CODE>.c
ache.amazonaws.com
LENS_SERVICE_PASSWORD=<LENS_SERVICE_PASSWORD>
LENS_SERVICE_PORT=6379
REGISTRY_SERVICE_IP_ADDRESS=clustercfg.atai-registry-service.<HASH>.<AWS_REGION
_CODE>.cache.amazonaws.com
REGISTRY_SERVICE_PASSWORD=<REGISTRY_SERVICE_PASSWORD>
REGISTRY_SERVICE_PORT=6379
```

Step 5: Store Auth Tokens in AWS Secrets Manager (Recommended)

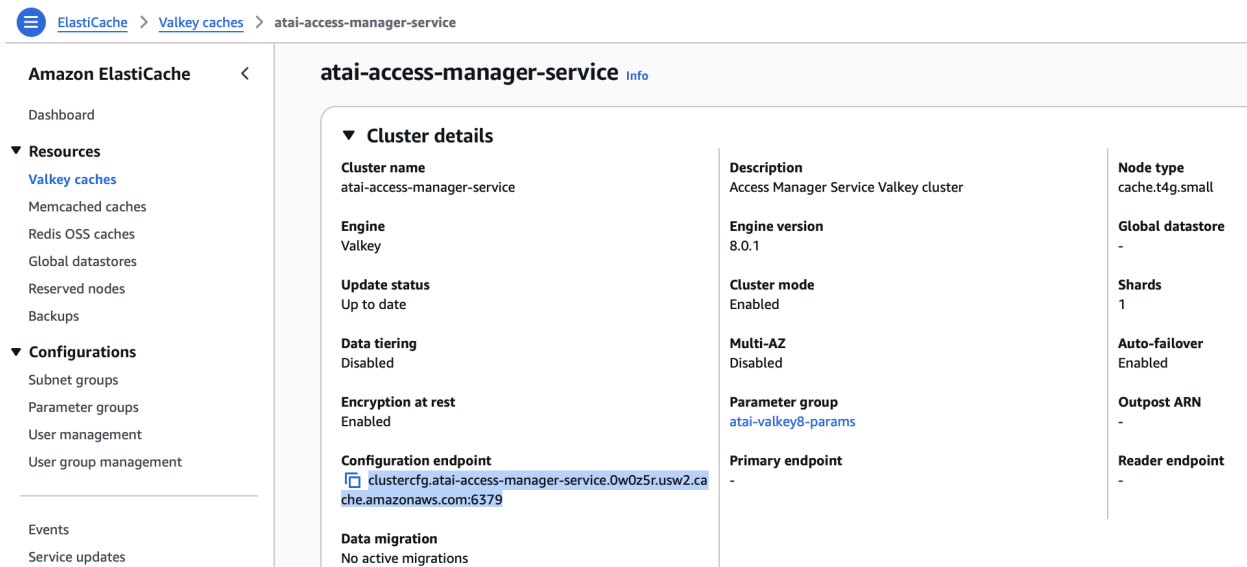
Note: This step is recommended but not mandatory. You can store credentials in Secrets Manager for easier access, better security, and integration with your applications. If you prefer to manage credentials differently, you can skip this step.

For each cluster, store the auth token and connection details:

1. Go to ElastiCache → Valkey caches → Click on your Valkey cache cluster



2. Copy the **Configuration endpoint**



3. Go to Secrets Manager → **Store a new secret**



4. Secret type: Other type of secret
5. Key/value pairs: Add:

- host: Configuration endpoint address (from cluster details, e.g., atai-access-manager-service.xxxxx.cache.amazonaws.com)
- port: 6379
- auth_token: The auth token you set for this cluster

☰ AWS Secrets Manager > Secrets > Store a new secret

Step 1
 Choose secret type

Step 2
 Configure secret

Step 3 - optional
 Configure rotation

Step 4
 Review

Choose secret type

Secret type Info

Credentials for Amazon RDS database
 Credentials for Amazon DocumentDB database
 Credentials for Amazon Redshift data warehouse

Credentials for other database
 Other type of secret
API key, OAuth token, other.

Key/value pairs Info

Key/value | Plaintext

host	clustercfg.atai-access-manager-service.0w0z5rusw2.cache.amazonaws.com	Remove
port	6379	Remove
auth	*****	Remove

[+ Add row](#)

Encryption key Info

You can encrypt using the KMS key that Secrets Manager creates or a customer-managed KMS key that you create.

aws/secretsmanager

[Add new key](#)

[Cancel](#) [Next](#)

6. Secret name: atai/valkey/{service-name}

Examples:

- atai/valkey/access-manager
- atai/valkey/api-events
- atai/valkey/dfc
- etc.

7. Encryption key: Use AWS managed key (default) or your KMS key

8. Click Next → Next

☰ AWS Secrets Manager > Secrets > Store a new secret

Step 1
 Choose secret type

Step 2
 Configure secret

Step 3 - optional
 Configure rotation

Step 4
 Review

Configure secret

Secret name and description Info

Secret name
 A descriptive name that helps you find your secret later.

atai/valkey/access-manager

Secret name must only contain alphanumeric characters and the characters /, +, @.

Description - optional

Credentials of the Access Manager Valkey

Maximum 250 characters.

Tags - optional

No tags associated with the secret.

[Add](#)

Resource permissions - optional Info [Edit permissions](#)

Add or edit a resource policy to access secrets across AWS accounts.

Replicate secret - optional

Create read-only replicas of your secret in other regions. Replica secrets incur a charge.

[Cancel](#) [Previous](#) [Next](#)

9. Do not configure rotation Click Next

AWS Secrets Manager > Secrets > Store a new secret

Step 1 Choose secret type
Step 2 Configure secret
Step 3 - optional Configure rotation
Step 4 Review

Configure rotation - optional

Configure automatic rotation [Info](#)
Configure AWS Secrets Manager to rotate this secret automatically.

Automatic rotation

Rotation schedule [Info](#)

Schedule expression builder schedule expression

Time unit Hours
Hours

Window duration - optional

Enter the time in hours.

Rotate immediately when the secret is stored. The next rotation will begin on your schedule.

Rotation function [Info](#)
Lambda rotation function [Info](#)
Choose a Lambda function that can rotate this secret.

[Create function](#)

10. Review and click on Store

AWS Secrets Manager > Secrets > Store a new secret

Step 1 Choose secret type
Step 2 Configure secret
Step 3 - optional Configure rotation
Step 4 Review

Review

Secret type

Secret type
Other type of secret

Encryption key
aws/secretsmanager

Secret configuration

Secret name
atai/valkey/access-manager

Description
Credentials of the Access Manager Valkey

Tags
-

Resource permissions
-

Secret replication
Disabled

Rotation schedule

Automatic rotation
Disabled

Rotation schedule
-

Rotation function

Lambda rotation function
-

Secret that performs rotation
-

Sample code
Use these code samples to retrieve the secret in your application.

Java | JavaScript | C# | Python3 | Ruby | Go | Rust | PHP

```
1 // Use this code snippet in your app.
2 // If you need more information about configurations or implementing the sample
3 // code, visit the AWS docs:
4 // https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/home.html
5
6 // Make sure to import the following packages in your code
7 // import software.amazon.awssdk.regions.Region;
8 // import software.amazon.awssdk.services.secretsmanager.SecretsManagerClient;
9 // import software.amazon.awssdk.services.secretsmanager.model.GetSecretValueRequest;
10 // import software.amazon.awssdk.services.secretsmanager.model.GetSecretValueResponse;
11
12 public static void getSecret() {
13     String secretName = "atal/valkey/access-manager";
14     Region region = Region.of("us-west-2");
15 }
```

Java Line 1, column 1 | Errors: 0 | Warnings: 0

[Download AWS SDK for Java](#)

Cancel Previous Store

11. Repeat for all 9 clusters with their respective:
 - a. Configuration endpoint (host)
 - b. Auth token
 - c. Service name in the secret path

Benefits of using Secrets Manager:

- Applications can retrieve credentials programmatically
- Credentials are encrypted at rest
- Access is logged for audit purposes
- No need to hardcode credentials in application code

Alternative: If you skip this step, ensure you have the auth tokens and configuration endpoints stored securely elsewhere, as they are required for applications to connect to the Valkey clusters.

⚠ Store your Valkey host, port, and token in a secure location. You will need them later in the *atal-platform* prerequisites, Step 3.2: Kubernetes Secrets Generation.

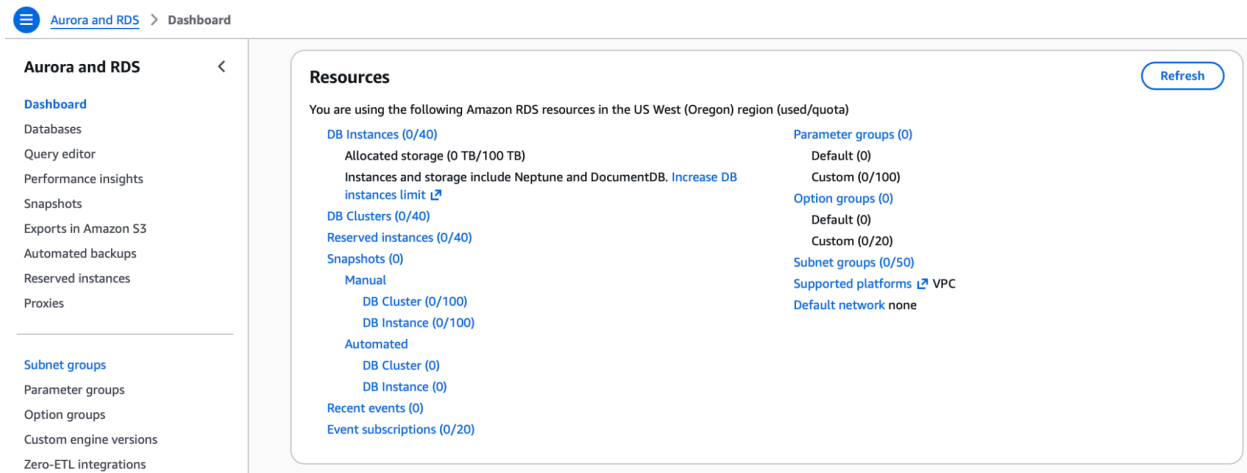
PostgreSQL database configuration

Prerequisites

1. VPC with database subnets (at least 2 AZs required for Aurora subnet group)
2. (Optional) Security group allowing access from EKS pods
3. Database subnet group

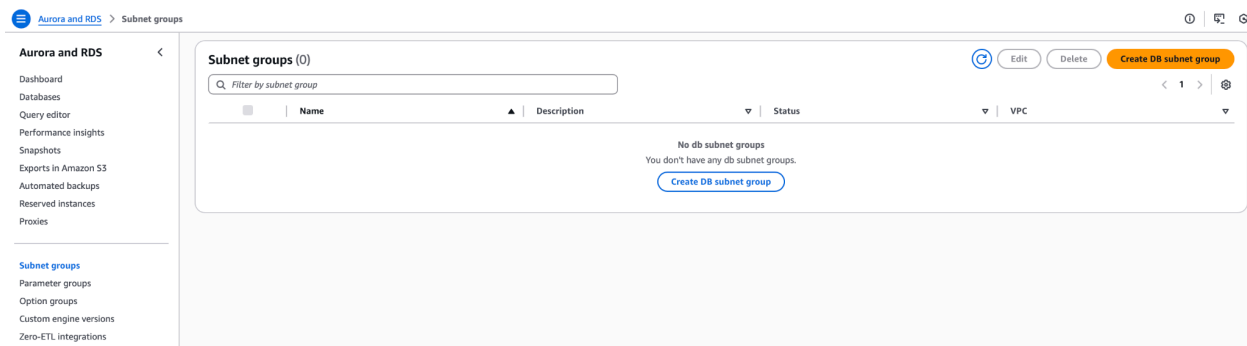
Step 1: Create Database Subnet Group (if not already created)

1. Go to RDS dashboard



The screenshot shows the 'Aurora and RDS' dashboard. The left sidebar contains navigation links for 'Dashboard', 'Databases', 'Query editor', 'Performance insights', 'Snapshots', 'Exports in Amazon S3', 'Automated backups', 'Reserved instances', 'Proxies', 'Subnet groups', 'Parameter groups', 'Option groups', 'Custom engine versions', and 'Zero-ETL integrations'. The main content area is titled 'Resources' and includes a 'Refresh' button. It displays resource usage for the US West (Oregon) region, listing categories such as DB Instances (0/40), Allocated storage (0 TB/100 TB), DB Clusters (0/40), Reserved instances (0/40), Snapshots (0), Manual (DB Cluster 0/100, DB Instance 0/100), Automated (DB Cluster 0, DB Instance 0), Recent events (0), Event subscriptions (0/20), Parameter groups (0), Default (0), Custom (0/100), Option groups (0), Default (0), Custom (0/20), Subnet groups (0/50), Supported platforms (VPC), and Default network (none).

2. Subnet groups → Create DB subnet group



The screenshot shows the 'Subnet groups' page in the Amazon RDS console. The left sidebar is the same as in the previous screenshot. The main content area is titled 'Subnet groups (0)' and includes a search filter, a table with columns for Name, Description, Status, and VPC, and a 'Create DB subnet group' button. A message states: 'No db subnet groups. You don't have any db subnet groups.' The 'Create DB subnet group' button is highlighted in orange.

3. Name: atai-db-subnet-group (or your name)
4. Description: Subnet group for Aurora PostgreSQL cluster
5. VPC: Select your VPC

[Aurora and RDS](#) > [Subnet groups](#) > Create DB subnet group

Create DB subnet group

To create a new subnet group, give it a name and a description, and choose an existing VPC. You will then be able to add subnets related to that VPC.

Subnet group details

Name
You won't be able to modify the name after your subnet group has been created.

Must contain from 1 to 255 characters. Alphanumeric characters, spaces, hyphens, underscores, and periods are allowed.

Description

VPC
Choose a VPC identifier that corresponds to the subnets you want to use for your DB subnet group. You won't be able to choose a different VPC identifier after your subnet group has been created.

6 Subnets, 2 Availability Zones

6. Availability Zones: Select at least 2 AZs (AWS requirement for Aurora)
 - a. Note: Aurora requires at least 2 AZs for the subnet group, even if you deploy a single instance
 - b. Example: Select us-west-2a and us-west-2b
7. Subnets: Select your database subnets atai-platform-vpc-database-us-west-2a and atai-platform-vpc-database-us-west-2b (e.g., 10.5.88.0/24 in us-west-2a and 10.5.89.0/24 in us-west-2b)

Add subnets

Availability Zones
Choose the Availability Zones that include the subnets you want to add.

Subnets
Choose the subnets that you want to add. The list includes the subnets in the selected Availability Zones.

For Multi-AZ DB clusters, you must select 3 subnets in 3 different Availability Zones.

Subnets selected (2)

Availability zone	Subnet name	Subnet ID	CIDR block
us-west-2b	atai-platform-vpc-database-us-west-2b	subnet-06bd3873d34243b83	10.5.89.0/24
us-west-2a	atai-platform-vpc-database-us-west-2a	subnet-06819ecd03094dea7	10.5.88.0/24

8. Click Create

Step 2: Create Security Group (if not existing)

1. Go to VPC → Security Groups → Create security group

[VPC](#) > Security Groups

Security Groups (2) info

Find security groups by attribute or tag

Name	Security group ID	Security group name	VPC ID	Description
-	sg-0c91975721df6571f	default	vpc-0a79faee3e664a31d	default VPC security group

Select a security group

2. Name: atai-rds-sg (or your name)
3. Description: Security group for Aurora PostgreSQL cluster
4. VPC: Select your VPC from section VPC configuration Step 1
5. Inbound rules: Add rule:
 - a. Type: PostgreSQL
 - b. Port: 5432
 - c. Source: Custom → Enter your VPC CIDR (e.g., 10.5.0.0/16)
 - d. Description: Allow PostgreSQL access from VPC

☰ VPC > Security Groups > Create security group

Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [Info](#)

Name cannot be edited after creation.

Description [Info](#)

VPC [Info](#)

Inbound rules Info

<small>Type Info</small> PostgreSQL	<small>Protocol Info</small> TCP	<small>Port range Info</small> 5432	<small>Source Info</small> Custom	<input type="text" value="10.5.0.0/16"/> <input type="button" value="X"/> <input type="button" value="10.5.0.0/16"/> <input type="button" value="X"/>	<small>Description - optional Info</small> <input type="text" value="Allow PostgreSQL access from VPC"/> <small>Allow PostgreSQL access from VPC</small>	<input type="button" value="Delete"/>
--	---	--	--	--	--	---------------------------------------

Outbound rules Info

<small>Type Info</small> All traffic	<small>Protocol Info</small> All	<small>Port range Info</small> All	<small>Destination Info</small> Custom	<input type="text" value="0.0.0.0/0"/> <input type="button" value="X"/> <input type="button" value="0.0.0.0/0"/> <input type="button" value="X"/>	<small>Description - optional Info</small> <input type="text"/>	<input type="button" value="Delete"/>
---	---	---	---	--	--	---------------------------------------

⚠ Rules with destination of 0.0.0.0/0 or :::0 allow your instances to send traffic to any IPv4 or IPv6 address. We recommend setting security group rules to be more restrictive and to only allow traffic to specific known IP addresses.

Tags - optional

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.

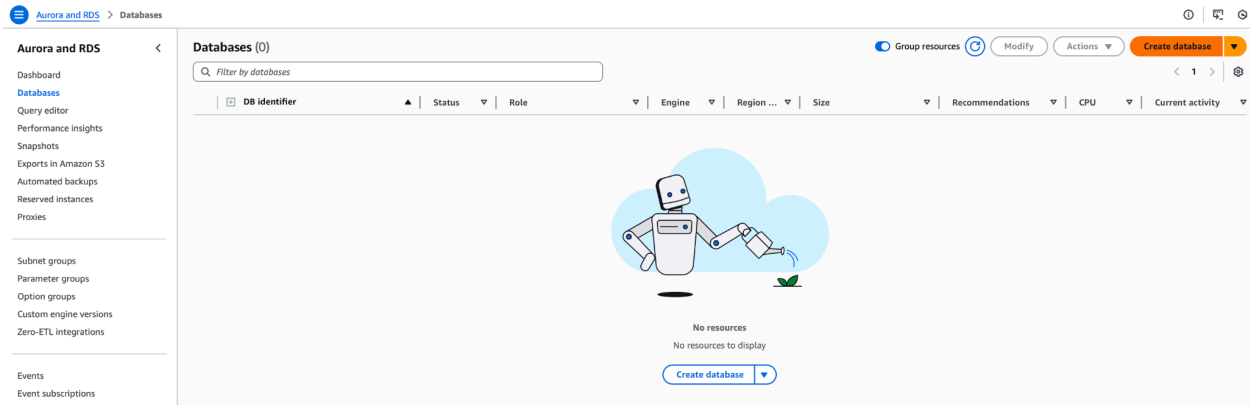
No tags associated with the resource.

You can add up to 50 more tags

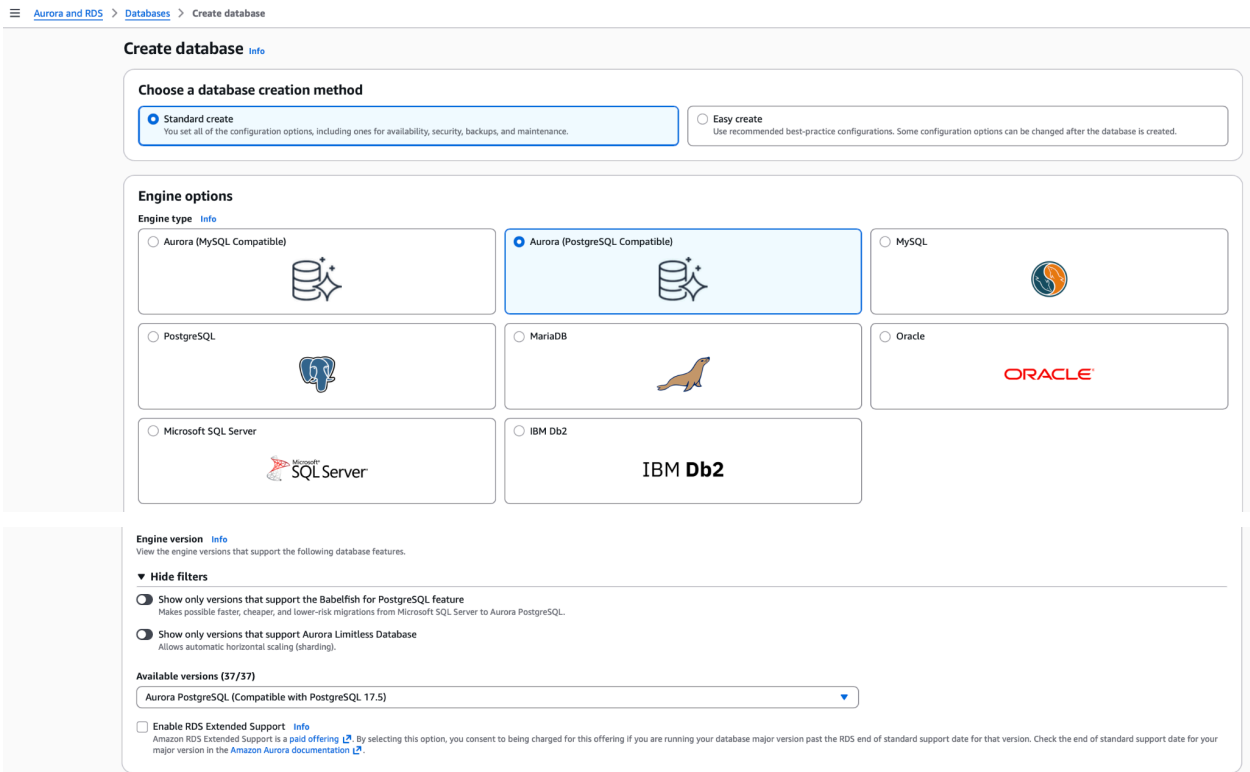
Note: For initial setup, opening to the VPC CIDR simplifies connectivity. Later, restrict to specific security groups (e.g., EKS node group security group) for tighter security.
 Click Create security group

Step 3: Create Aurora PostgreSQL Cluster

1. Go to RDS → Databases → Create database



2. Database creation method: Standard create
3. Engine options:
4. Engine type: Amazon Aurora
 - a. Edition: Amazon Aurora PostgreSQL-Compatible Edition
 - b. Available versions: Select Aurora PostgreSQL 17.5
 - c. Templates: Production (or Dev/Test for non-production)



Templates

Choose a sample template to meet your use case.

Production

Use defaults for high availability and fast, consistent performance.

Dev/Test

This instance is intended for development use outside of a production environment.

5. Settings:

- DB cluster identifier: atai-platform
- Master username: postgres (or your preferred username)
- Credentials management: Managed in AWS secrets manager

Settings

DB cluster identifier [Info](#)

Type a name for your DB cluster. The name must be unique across all DB clusters owned by your AWS account in the current AWS Region.

atai-platform

The DB cluster identifier is case-insensitive, but is stored as all lowercase (as in "mydbcluster"). Constraints: 1 to 63 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

▼ Credentials Settings

Master username [Info](#)

Type a login ID for the master user of your DB instance.

postgres

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management

You can use AWS Secrets Manager or manage your master user credentials.

Managed in AWS Secrets Manager - most secure

RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

Self managed

Create your own password or have RDS create a password that you manage.

If you manage the master user credentials in AWS Secrets Manager, additional charges apply. See [AWS Secrets Manager pricing](#). Additionally, some RDS features aren't supported. See [limitations here](#).

Select the encryption key [Info](#)

You can encrypt using the KMS key that Secrets Manager creates or a customer managed KMS key that you create.

aws/secretsmanager (default)

[Add new key](#)

6. Cluster storage configuration

- Storage type: Aurora Standard

Cluster storage configuration [Info](#)

Choose the storage configuration for the Aurora DB cluster that best fits your application's price predictability and price performance needs.

Configuration options

Database instance, storage, and I/O charges vary depending on the configuration. [Learn more](#)

Aurora I/O-Optimized

- Predictable pricing for all applications. Improved price performance for I/O-intensive applications (I/O costs >25% of total database costs).
- No additional charges for read/write I/O operations. DB instance and storage prices include I/O usage.

Aurora Standard

- Cost-effective pricing for many applications with moderate I/O usage (I/O costs <25% of total database costs).
- Pay-per-request I/O charges apply. DB instance and storage prices don't include I/O usage.

7. Instance configuration

- DB instance class: Select your instance class (e.g., db.t4g.medium for dev, db.r7g.large for production)

Instance configuration

The DB instance configuration options below are limited to those supported by the engine that you selected above.

DB instance class [Info](#)

▼ Hide filters

Include previous generation classes

Serverless v2

Memory optimized classes (includes r classes)

Burstable classes (includes t classes)

Optimized Reads classes

db.r7g.large

2 vCPUs 16 GiB RAM EBS Bandwidth: Up to 10,000 Mbps Network: Up to 12.5 Gbps

8. For Availability and Durability select Don't create an Aurora Replica

Availability & durability

Multi-AZ deployment [Info](#)

Create an Aurora Replica or Reader node in a different AZ (recommended for scaled availability)
Creates an Aurora Replica for fast failover and high availability.

Don't create an Aurora Replica

9. Connectivity:

- Virtual private cloud (VPC): Select your VPC
- DB subnet group: Select your database subnet group (created in Step 1)
- Publicly accessible: No (should be in private subnets)

Connectivity [Info](#)

Compute resource

Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.

Don't connect to an EC2 compute resource
Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.

Connect to an EC2 compute resource
Set up a connection to an EC2 compute resource for this database.

Network type [Info](#)

To use dual-stack mode, make sure that you associate an IPv6 CIDR block with a subnet in the VPC you specify.

IPv4
Your resources can communicate only over the IPv4 addressing protocol.

Dual-stack mode
Your resources can communicate over IPv4, IPv6, or both.

Virtual private cloud (VPC) [Info](#)

Choose the VPC. The VPC defines the virtual networking environment for this DB cluster.

atai-platform-vpc (vpc-0a79faee3e664a31d)
6 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

After a database is created, you can't change its VPC.

DB subnet group [Info](#)

Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB cluster can use in the VPC that you selected.

atai-db-subnet-group
2 Subnets, 2 Availability Zones

Public access [Info](#)

Yes
RDS assigns a public IP address to the cluster. Amazon EC2 instances and other resources outside of the VPC can connect to your cluster. Resources inside the VPC can also connect to the cluster. Choose one or more VPC security groups that specify which resources can connect to the cluster.

No
RDS doesn't assign a public IP address to the cluster. Only Amazon EC2 instances and other resources inside the VPC can connect to your cluster. Choose one or more VPC security groups that specify which resources can connect to the cluster.

- VPC security group: Choose existing → Select atai-rds-sg
- Availability Zone: No preference (AWS will choose) or select your preferred AZ for the primary instance

VPC security group (firewall) [Info](#)

Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

Choose existing
Choose existing VPC security groups

Create new
Create new VPC security group

Existing VPC security groups

Choose one or more options

atai-rds-sg × default ×

Certificate authority - optional [Info](#)

Using a server certificate provides an extra layer of security by validating that the connection is being made to an Amazon database. It does so by checking the server certificate that is automatically installed on all databases that you provision.

rds-ca-rsa2048-g1 (default)
Expiry: May 24, 2061

If you don't select a certificate authority, RDS chooses one for you.

RDS Data API

Enable the RDS Data API [Info](#)
Enable the SQL HTTP endpoint for the Data API. With this endpoint enabled, you can run SQL queries against this database over HTTP. You can do so by using the CLI, an AWS SDK, or the RDS query editor. For information about pricing, see [Amazon RDS pricing](#).

Note: For low latency, you can select the same AZ as your EKS node groups (e.g., us-west-2a)

10. Leave the default values for Read replica write forwarding, Tags, Babelfish settings and Database authentication.

Read replica write forwarding

Turn on local write forwarding [Info](#)
Issues write operations from reader DB instances within the same DB cluster.

Tags - optional

A tag consists of a case-sensitive key-value pair.
No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tags.

Babelfish settings [Info](#)

Turn on Babelfish
Makes possible faster, cheaper, and lower-risk migrations from Microsoft SQL Server to Aurora PostgreSQL.

Database authentication [Info](#)

Password authentication is always active for your database engine. You can also turn on additional authentication methods for your database below.

IAM database authentication
Authenticates using IAM database authentication.

Kerberos authentication
Authenticates using Kerberos authentication through an AWS Directory Service for Microsoft Active Directory.

11. Monitoring

a. Select Database insight - Standard

Monitoring [Info](#)

Choose monitoring tools for this database. Database Insights provides a combined view of Performance Insights and Enhanced Monitoring for your fleet of databases. Database Insights pricing is separate from RDS monthly estimates. See [Amazon CloudWatch pricing](#).

Database Insights - Advanced

- Retains 15 months of performance history
- Fleet-level monitoring
- Integration with CloudWatch Application Signals

Database Insights - Standard

- Retains 7 days of performance history, with the option to pay for the retention of up to 24 months of performance history

Performance Insights

Enable Performance Insights
With Performance Insights dashboard, you can visualize the database load on your Amazon RDS DB Instance load and filter the load by waits, SQL statements, hosts, or users.

Retention period

7 days

AWS KMS key [Info](#)

(default) aws/rds

b. Disable Enhanced monitoring

c. Select PostgreSQL log under the Logs exports section

d. You can turn off DevOps Guru to avoid extra costs

Additional monitoring settings

Enhanced Monitoring, CloudWatch Logs and DevOps Guru

Enhanced Monitoring

Enable Enhanced monitoring
Enabling Enhanced Monitoring metrics are useful when you want to see how different processes or threads use the CPU.

Log exports

Select the log types to publish to Amazon CloudWatch Logs

iam-db-auth-error log

instance log

PostgreSQL log

IAM role

The following service-linked role is used for publishing logs to CloudWatch Logs.

RDS service-linked role

DevOps Guru

Turn on DevOps Guru [Info](#)
DevOps Guru for RDS automatically detects performance anomalies for DB instances and provides recommendations.

12. Additional configuration (optional — defaults are fine, customize if needed):
 - a. Initial database name: Leave blank (or enter a name if you want to create an initial database)
 - b. DB cluster parameter group: default.aurora-postgresql17 (or create custom if needed)
 - c. DB parameter group: default.aurora-postgresql17 (or create custom if needed)
 - d. Backup retention period: 7 days (default, or your preference)
 - e. Backup window: No preference (default, or set a specific window)
 - f. Enable encryption: Enable encryption (it's enabled by default)
 - g. Encryption key: Use AWS managed key (default) or your KMS key

▼ Additional configuration
Database options, encryption turned on, failover, backup turned on, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned on.

Database options

Initial database name [Info](#)

If you do not specify a database name, Amazon RDS does not create a database.

DB cluster parameter group [Info](#)

DB parameter group [Info](#)

Option group [Info](#)

Failover priority

Backup

Backup retention period [Info](#)
The number of days (1-35) for which automatic backups are kept.
 days

Copy tags to snapshots

Enable encryption
Choose to encrypt the given instance. Master key IDs and aliases appear in the list after they have been created using the AWS Key Management Service console. [Info](#)

AWS KMS key [Info](#)

Maintenance

Auto minor version upgrade [Info](#)

Enable auto minor version upgrade
Enabling auto minor version upgrade will automatically upgrade your database minor version. For limitations and more details, see [Automatically upgrading the minor engine version documentation](#).

Maintenance window [Info](#)
Select the period you want pending modifications or maintenance applied to the database by Amazon RDS.

Choose a window

No preference

Enable deletion protection
Protects the database from being deleted accidentally. While this option is enabled, you can't delete the database.

Estimated monthly costs

DB instance	402.96 USD
Storage	0.10 USD
Total	403.06 USD

This billing estimate is based on on-demand usage as described in [Amazon Aurora Pricing](#). Estimate does not consider reserved instance benefits and costs for instance storage, IOs, or data transfer.
 Estimate your monthly costs for the DB Instance using the [AWS Simple Monthly Calculator](#).

ⓘ You are responsible for ensuring that you have all of the necessary rights for any third-party products or services that you use with AWS services.

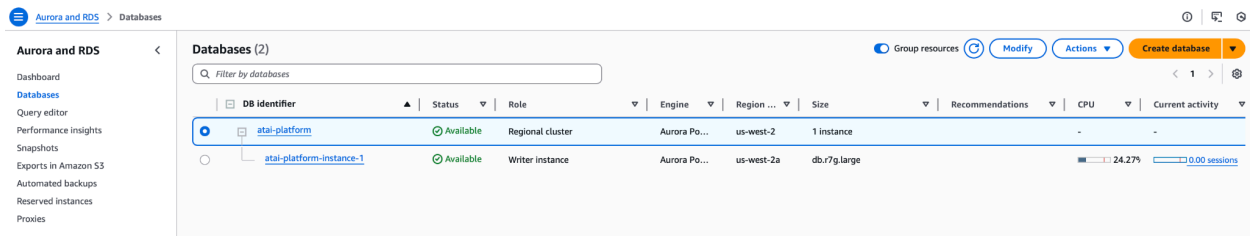
[Cancel](#) [Create database](#)

Note: No additional configurations are required — defaults work. You can customize backup retention, monitoring, and other settings if needed.

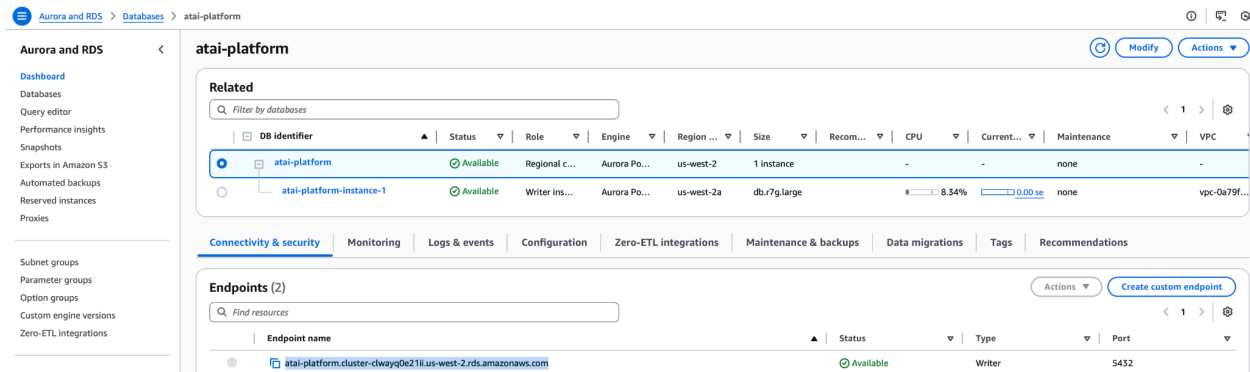
13. Click on **Create database**

14. Get my database hostname

- After you create your database go to the RDS dashboard → **Databases**
- Select your database cluster



c. Copy the **Cluster endpoint**

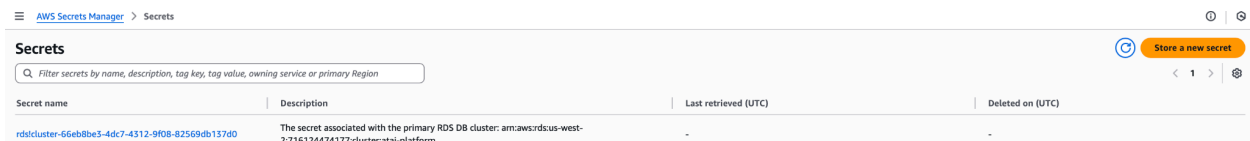


15. Get my database credentials

- Go to Secrets Manager → **Secrets**



b. Click on the new RDS secret



- Click on **Get secret value**. You'll be able to see the user and password to connect to your RDS postgresSQL.

rdscluster-66eb8be3-4dc7-4312-9f08-82569db137d0

This secret was created by Amazon RDS (rds). Because this secret is managed by Amazon RDS (rds), you will not be able to modify the secret value. However, the secret may be modified in any other manner. [Learn more](#)

Secret details

Encryption key

aws/secretsmanager

Secret name

rdscluster-66eb8be3-4dc7-4312-9f08-82569db137d0

Secret ARN

arn:aws:secretsmanager:us-west-2:716124474177:secret:rdscluster-66eb8be3-4dc7-4312-9f08-82569db137d0-z0QsWh

Secret description

The secret associated with the primary RDS DB cluster: am:aws:rds:us-west-2:716124474177:cluster:atai-platform

Secret type

-

Actions

Overview | Rotation | Versions | Replication | Tags

Secret value

Retrieve and view the secret value.

Retrieve secret value

Step 4: Extra Database Configuration steps

Connect to your RDS, in our case we are using bastion host machine:

None

```
psql -h your-rds-endpoint.region.rds.amazonaws.com -U master_username -d postgres
```

Create databases

None

```
CREATE DATABASE experiment_logs_db;  
CREATE DATABASE iam_db;  
CREATE DATABASE platform_api_events_db;  
CREATE DATABASE platform_eval_db;  
CREATE DATABASE platform_lens_db;  
CREATE DATABASE platform_logs_db;  
CREATE DATABASE lens_db;
```

```
postgres=> \l
```

Name	Owner	Encoding	Locale Provider	Collate	Ctype	Locale	ICU Rules	Access privileges
experiment_logs_db	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=Tc/postgres + postgres=Ctc/postgres+ atai_dev=Ctc/postgres
iam_db	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=Tc/postgres + postgres=Ctc/postgres+ atai_dev=Ctc/postgres
lens_db	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=Tc/postgres + postgres=Ctc/postgres+ atai_dev=Ctc/postgres
platform_api_events_db	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=Tc/postgres + postgres=Ctc/postgres+ atai_dev=Ctc/postgres
platform_eval_db	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=Tc/postgres + postgres=Ctc/postgres+ atai_dev=Ctc/postgres
platform_lens_db	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=Tc/postgres + postgres=Ctc/postgres+ atai_dev=Ctc/postgres
platform_logs_db	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=Tc/postgres + postgres=Ctc/postgres+ atai_dev=Ctc/postgres
postgres	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=Tc/postgres + postgres=Ctc/postgres+ atai_dev=Ctc/postgres
rdsadmin	rdsadmin	UTF8	libc	en_US.UTF-8	en_US.UTF-8			rdsadmin=Ctc/rdsadmin
template0	rdsadmin	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=c/rdsadmin + rdsadmin=Ctc/rdsadmin
template1	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=c/postgres + postgres=Ctc/postgres

(11 rows)
postgres=>

Create atai_dev database user

None


```
CREATE USER atai_dev WITH PASSWORD 'your_secure_password';
```

```
(10 rows)
postgres=> \du

```

Role name	Attributes
atai_dev	
postgres	Create role, Create DB Password valid until infinity +
rds_ad	Cannot login
rds_extension	No inheritance, Cannot login
rds_iam	Cannot login
rds_password	Cannot login
rds_replication	Cannot login
rds_superuser	Cannot login
rdsadmin	Superuser, Create role, Create DB, Replication, Bypass RLS+ Password valid until infinity
rdswriteforwarduser	No inheritance

```
postgres=> █
```

 Store your atai_dev user and password in a secure location. You will need them later in the *atai-platform* prerequisites, Step 3.2: Kubernetes Secrets Generation.

Note 1: The **database** secrets will be required for following services:

- lens-service-db-backend
- api-events-service-backend

Note 2: Add the secrets in the same section as the services mentioned in **Note 1**. These values will be referenced as:

None

```
[atai-platform-api-events-service-backend]
```

```
...
```

```
API_EVENTS_SERVICE_DB_HOST=<DB_NAME>.<CLUSTER_HASH>.<AWS_REGION>.rds.amazonaws.com
```

```
API_EVENTS_SERVICE_DB_PASS=<API_EVENTS_SERVICE_DB_PASS>
```

```
API_EVENTS_SERVICE_DB_PORT=5432
```

```
API_EVENTS_SERVICE_DB_USER=atai_dev
```

```
[atai-platform-lens-service-db-backend]
...
POSTGRES_HOST=<DB_NAME>.<CLUSTER_HASH>.<AWS_REGION>.rds.amazonaws.com

POSTGRES_PASSWORD=<POSTGRES_PASSWORD>
POSTGRES_PORT=5432
POSTGRES_USER=atai_dev
```

Then assign proper permissions to the new PostgreSQL user:

```
None

GRANT ALL PRIVILEGES ON DATABASE experiment_logs_db TO atai_dev;
GRANT ALL PRIVILEGES ON DATABASE iam_db TO atai_dev;
GRANT ALL PRIVILEGES ON DATABASE platform_api_events_db TO atai_dev;
GRANT ALL PRIVILEGES ON DATABASE platform_eval_db TO atai_dev;
GRANT ALL PRIVILEGES ON DATABASE platform_lens_db TO atai_dev;
GRANT ALL PRIVILEGES ON DATABASE platform_logs_db TO atai_dev;
GRANT ALL PRIVILEGES ON DATABASE lens_db TO atai_dev;
GRANT ALL PRIVILEGES ON DATABASE postgres TO atai_dev;
```

Additional permissions required:

```
None

\c database_name (this should be done on all DBs)

GRANT USAGE ON SCHEMA public TO atai_dev;
GRANT CREATE ON SCHEMA public TO atai_dev;
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO atai_dev;
GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO atai_dev; ALTER
DEFAULT PRIVILEGES IN SCHEMA public GRANT ALL ON TABLES TO atai_dev;
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT ALL ON SEQUENCES TO atai_dev;
GRANT ALL PRIVILEGES ON ALL FUNCTIONS IN SCHEMA public TO atai_dev; ALTER
DEFAULT PRIVILEGES IN SCHEMA public GRANT ALL ON FUNCTIONS TO atai_dev;
```

Solve a common permission issue:

None

```
-- Issue:psycopg2.errors.InsufficientPrivilege:  
-- permission denied for schema public  
--- LINE 1: CREATE TABLE IF NOT EXISTS platform_lens_db(  
-- Fix: on that specific DB
```

```
GRANT CREATE ON SCHEMA public TO atai_dev;  
GRANT USAGE ON SCHEMA public TO atai_dev;
```

EKS cluster configuration

Prerequisites

1. An existing VPC and subnets that meet Amazon EKS requirements
2. At least two private subnets in your VPC:
 - a. Recommended naming:
 - i. atai-platform-vpc-private-us-west-2a,
 - ii. atai-platform-vpc-private-us-west-2b (or similar)
 - b. Recommended CIDR blocks: /20 per subnet
 - i. Example: 10.5.0.0/20 (subnet 1)
 - ii. Example: 10.5.16.0/20 (subnet 2)
 - c. Subnets must be in different Availability Zones
 - d. Subnets must have routes to NAT Gateway or Internet Gateway for outbound internet access
3. The kubectl command line tool is required. The version can be the same as or up to one minor version earlier or later than the Kubernetes version of your cluster.
4. Version 2.12.3 or later or version 1.27.160 or later of the AWS Command Line Interface (AWS CLI) installed and configured on your device.
5. An IAM principal with permissions to create and describe an Amazon EKS cluster

Step 1: Create cluster IAM role

1. If you already have a cluster IAM role, or you're going to create your cluster with eksctl, then you can skip this step. By default, eksctl creates a role for you.
2. Run the following command to create an IAM trust policy JSON file.

```
None
cat > eks-cluster-role-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

3. Create the Amazon EKS cluster IAM role. If necessary, preface `eks-cluster-role-trust-policy.json` with the path on your computer that you wrote the file to in the previous step. The command associates the trust policy that you created in the previous step to the role. To create an IAM role, the IAM principal that is creating the role must be assigned the `iam:CreateRole` action (permission).

None

```
aws iam create-role --role-name atai-platform-eks-cluster-role
--assume-role-policy-document file://"eks-cluster-role-trust-policy.json"
```

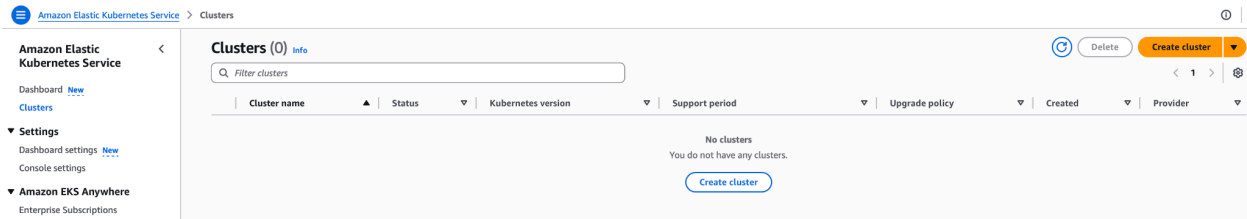
4. You can assign either the Amazon EKS managed policy or create your own custom policy. For the minimum permissions that you must use in your custom policy, see [Amazon EKS cluster IAM role](#). Attach the Amazon EKS managed policy named [AmazonEKSClusterPolicy](#) to the role. To attach an IAM policy to an IAM principal, the principal that is attaching the policy must be assigned one of the following IAM actions (permissions): `iam:AttachUserPolicy` or `iam:AttachRolePolicy`.

None

```
aws iam attach-role-policy --policy-arn
arn:aws:iam::aws:policy/AmazonEKSClusterPolicy --role-name
atai-platform-eks-cluster-role
```

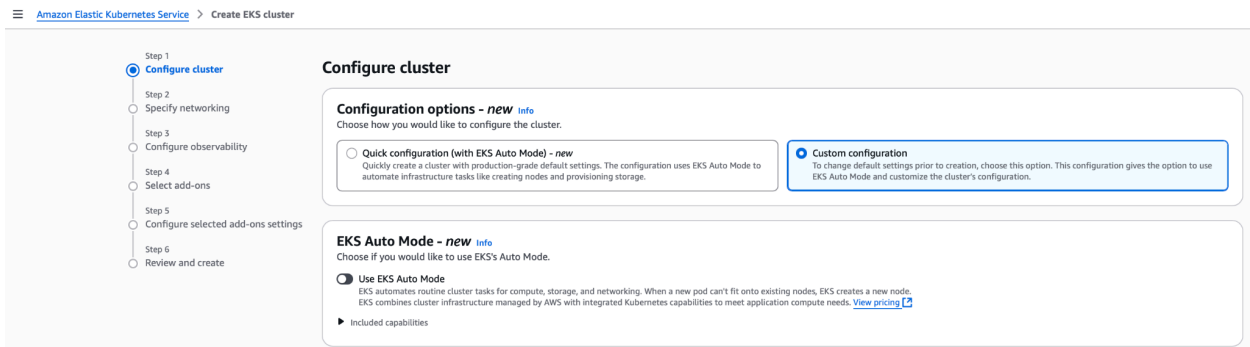
Step 2: Create cluster

1. Open EKS Console → Click on **Create cluster**



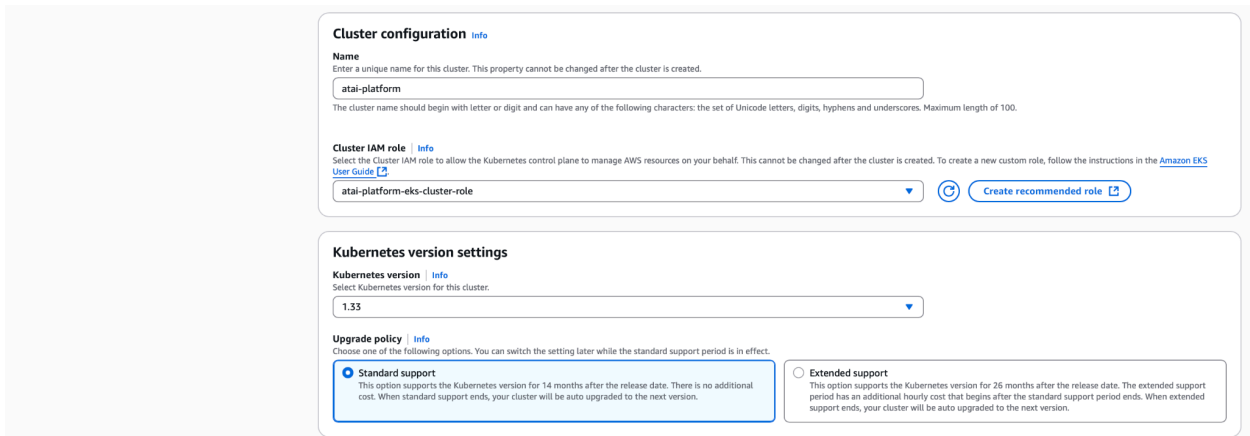
2. Under Configuration options select Custom configuration

3. Under EKS Auto Mode, toggle Use EKS Auto Mode off.



4. On the Configure cluster page, enter the following fields:

- Name: atai-platform
- Cluster IAM role – Choose the Amazon EKS cluster IAM role that you created in the Step 1 to allow the Kubernetes control plane to manage AWS resources on your behalf.
- Kubernetes version: 1.33
- Support type: Standard support



5. Cluster access

- a. Select Allow cluster administrator access
- b. Cluster authentication mode: EKS API and ConfigMap

Auto Mode Compute - new [info](#)
Configure node management for your EKS cluster. EKS offers four compute options: EKS Auto Mode, EC2 Managed Node Groups, Fargate, and hybrid nodes. Node groups, Fargate profiles, and hybrid nodes are configured after cluster creation. You can also create self-managed nodes.

Compute configuration
If EKS Auto Mode is not managing compute resources, you need to create compute resources once the cluster is ready. We recommend creating a node group after cluster creation. [View documentation](#)

Cluster access [info](#)
Control how IAM principals can access this cluster.

Bootstrap cluster administrator access [info](#)
Choose whether the IAM principal creating the cluster has Kubernetes cluster administrator access.

Allow cluster administrator access
Allow cluster administrator access for your IAM principal.

Disallow cluster administrator access
Disallow cluster administrator access for your IAM principal.

Cluster authentication mode [info](#)
Configure which source the cluster will use for authenticated IAM principals.

EKS API
The cluster will source authenticated IAM principals only from EKS access entry APIs.

EKS API and ConfigMap
The cluster will source authenticated IAM principals from both EKS access entry APIs and the aws-auth ConfigMap.

6. Envelope encryption

- a. By default, AWS implements envelope encryption using an AWS owned key. Alternatively, you can setup your own customer managed key (CMK) and link this key by providing the CMK ARN when configuring your EKS cluster.

Envelope encryption [info](#)
Envelope encryption is applied to all Kubernetes API data.

By default, AWS implements envelope encryption using an AWS owned key. Alternatively, you can setup your own customer managed key (CMK) and link this key by providing the CMK ARN when configuring your EKS cluster.

Use your own AWS KMS key
After a cluster is created, you can migrate from using an AWS owned key to a customer managed key (CMK), but not vice versa.

7. Use the default configuration for ARC Zonal Shift (disabled by default).
(Optional) Enable Deletion protection
(Optional) Add tags
Click on **Next**

ARC Zonal shift [info](#)
Shift application traffic away from an impaired Availability Zone (AZ) in your EKS cluster. You can change this later.

Enabled
EKS will register your cluster with ARC zonal shift to enable you to use zonal shift to shift application traffic away from an AZ.

Disabled
EKS will not register your cluster with ARC zonal shift.

Before you start a zonal shift, you need to setup your cluster environment to be resilient to an AZ failure beforehand. [Learn more](#)

Deletion protection
Deletion protection must be turned off to be able to delete a cluster. It can be turned on and off after the cluster is created.

Turn on deletion protection
Deletion protection provides additional security against accidental cluster deletion.

Tags (0) [info](#)
No tags associated with the resource.

[Add new tag](#)
You can add up to 50 tags.

[Cancel](#) [Next](#)

8. Networking

- a. VPC: Select your VPC
- b. Subnets: Select at least two private subnets:
 - i. atai-platform-vpc-private-us-west-2a (10.5.0.0/20)
 - ii. atai-platform-vpc-private-us-west-2b (10.5.16.0/20)
- c. (Optional) Security groups: EKS automatically creates a cluster security group on cluster creation to facilitate communication between worker nodes and control plane

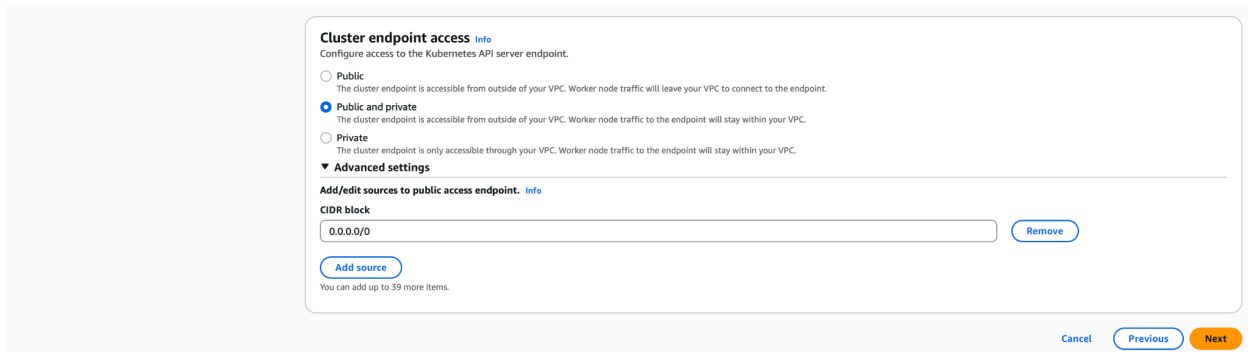
After the cluster is created, you must **retrieve the security group ID** assigned by AWS. This ID is required when creating Launch Templates for your Managed Node Groups, to ensure proper networking and access between the control plane and the worker nodes. To learn how to get the security group automatically created by EKS go to Step 4: Get the default cluster security group

- d. Cluster IP address family: IPv4

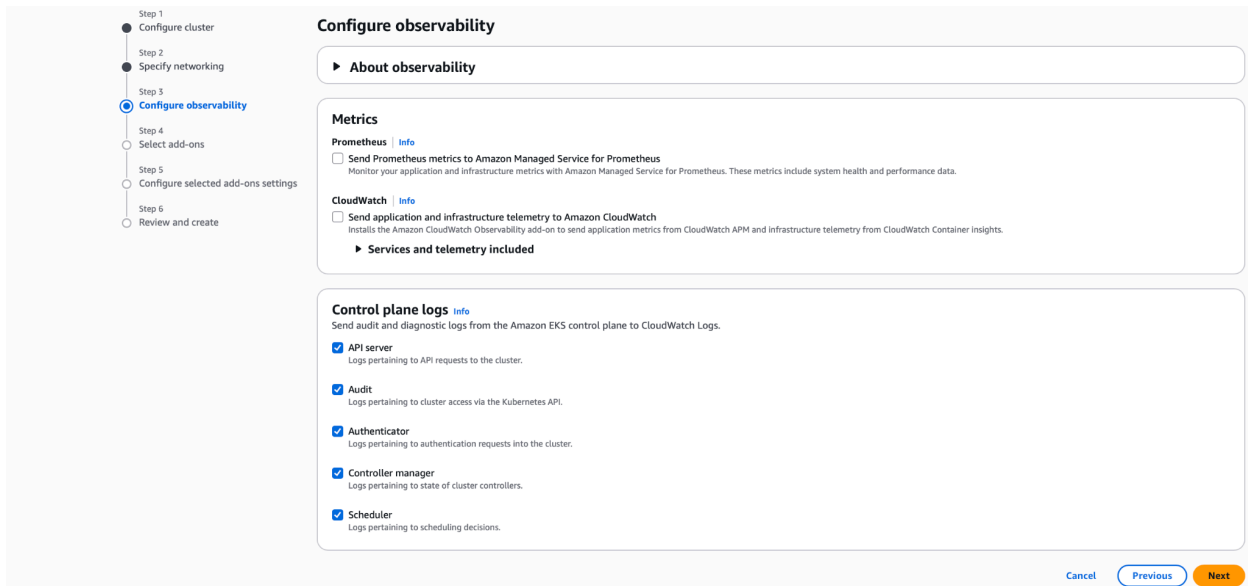
The screenshot shows the 'Specify networking' step in the AWS IAM console. On the left, a progress bar indicates the current step is 'Specify networking'. The main content area is titled 'Specify networking' and contains several sections:

- Networking Info:** A warning that the IP address family and service IP address range cannot be changed after cluster creation.
- VPC:** A dropdown menu showing 'vpc-0a79faee2e664a31d | atai-platform-vpc'.
- Subnets:** A dropdown menu showing two selected subnets: 'subnet-0bd36810cb1c67e50 | atai-platform-vpc-private-us-west-2a' and 'subnet-0c43aa4dd2636a7b6 | atai-platform-vpc-private-us-west-2b'. A 'Clear selected subnets' button is also present.
- Additional security groups:** A dropdown menu for selecting security groups.
- Choose cluster IP address family:** Radio buttons for 'IPv4' (selected) and 'IPv6'.
- Configure Kubernetes service IP address block:** A radio button for 'Specify the range from which cluster services will receive IP addresses'.
- Configure remote networks to enable hybrid nodes:** A radio button for 'Specify the CIDR blocks for your on-premises environments that you will use for hybrid nodes'.

- 9. Cluster endpoint access
 - a. Public and private
 - b. Click on **Next**



- 10. Configure observability page:
 - a. Control plane logging: Enable all:
 - i. API server
 - ii. Audit
 - iii. Authenticator
 - iv. Controller manager
 - v. Scheduler
 - b. Prometheus metrics: Optional



11. Select add-ons: Select these add-ons:

- a. Amazon VPC CNI (required)
- b. CoreDNS (required)
- c. kube-proxy (required)
- d. (Optional) EKS Pod Identity Agent (for Pod Identity)

Click Next

Select add-ons
Review the add-ons from multiple categories, then select add-ons to enhance your cluster.

AWS add-ons (19) [Info](#)

- Node monitoring agent** [Info](#)
Enable automatic detection of node health issues.
Category observability
Compatible compute EC2, Hybrid Nodes
- CoreDNS** [Info](#)
Enable service discovery within your cluster.
Category networking
Compatible compute EC2, Hybrid Nodes, Fargate, EKS Auto Mode
- Amazon VPC CNI** [Info](#)
Enable pod networking within your cluster.
Category networking
Compatible compute EC2
- kube-proxy** [Info](#)
Enable service networking within your cluster.
Category networking
Compatible compute EC2, Hybrid Nodes
- SR-IOV Network Metrics Exporter**
Install SR-IOV Network Metrics Exporter to generate Prometheus metrics about SR-IOV (Single-Root I/O Virtualization) network devices in EKS bare metal environments.
Category observability
Compatible compute EC2
- Amazon FSx CSI driver**
Enable Amazon FSx for Lustre within your cluster.
Category storage
Compatible compute EC2, EKS Auto Mode
- Amazon EKS Pod Identity Agent** [Info](#)
Install EKS Pod Identity Agent to use EKS Pod Identity to grant AWS IAM permissions to pods through Kubernetes service accounts.
Category security
Compatible compute EC2, Hybrid Nodes
- Amazon CloudWatch Observability** [Info](#)
Install CloudWatch Agent and enable Container Insights and Application Signals within your cluster.
Category observability
Compatible compute EC2, Hybrid Nodes, EKS Auto Mode
- Mountpoint for Amazon S3 CSI Driver** [Info](#)
Enable Mountpoint for Amazon Simple Storage Service (S3) within your cluster.
Category storage
Compatible compute EC2, EKS Auto Mode
⚠ Add-on does not support EKS Pod Identity at this time. Please use IAM roles for service accounts (IRSA) with this add-on after the cluster is created.
- Amazon SageMaker HyperPod task governance**
Prioritize tasks, allocate compute resources, and maximize utilization.
Category policy-management
Compatible compute HyperPod

12. Configure selected add-ons settings

- VPC CNI: v1.20.4-eksbuild.2 or Default/Current version
- CoreDNS: v1.12.1-eksbuild.2 or Default/Current version
- kube-proxy: v1.33.3-eksbuild.4 or Default/Current version
- EKS Pod Identity Agent: v0.1.35 or Default/Current version

13. Review your cluster configuration and click on **Create**

Step 1
● Configure cluster

Step 2
● Specify networking

Step 3
● Configure observability

Step 4
● Select add-ons

Step 5
● Configure selected add-ons settings

Step 6
● **Review and create**

Review and create

Step 1: Cluster Edit

Cluster configuration

Name atai-platform	Kubernetes version 1.33
EKS Auto Mode Disabled	Upgrade policy Standard support
Cluster IAM role arn:aws:iam::716124474177:role/atai-platform-eks-cluster-role	Kubernetes cluster administrator access Allow cluster administrator access
Authentication mode EKS API and ConfigMap	

ARC Zonal shift

ARC Zonal shift
Disabled

Deletion protection

Deletion protection
Disabled

Tags (0)
Tags that you've added. Each tag consists of a key and an optional value.

< 1 >

Key	Value
No tags This cluster does not have any tags.	

Step 2: Networking Edit

Networking
These properties cannot be changed after the cluster is created.

VPC vpc-0a79faee3e664a31d	Subnets subnet-0bd36810cb1c67e50 subnet-0c43aa4dd2636a7b6
Cluster IP address family IPv4	

Cluster endpoint access

API server endpoint access Public and private	Public access source allowlist 0.0.0.0/0
---	--

Step 3: Observability Edit

Control plane logs

API server on	Authenticator on	Scheduler on
Audit on	Controller manager on	

Step 4: Add-ons Edit

Selected add-ons (4)

Find add-on

Add-on name	Type	Status
coredns	networking	Ready to install
eks-pod-identity-agent	security	Ready to install
kube-proxy	networking	Ready to install
vpc-cni	networking	Ready to install

Step 5: Versions Edit

Selected add-ons version (4)

Find add-on versions

Add-on name	Version
coredns	v1.12.1-eksbuild.2
eks-pod-identity-agent	v1.3.9-eksbuild.5
kube-proxy	v1.33.3-eksbuild.4
vpc-cni	v1.20.4-eksbuild.2

EKS Pod Identity (1)

Add-on name	IAM role	Service account
vpc-cni	Not set	aws-node

Cancel Previous Create

Step 3: Update kubeconfig

1. Enable kubectl to communicate with your cluster by adding a new context to the kubectl config file.

None

```
aws eks update-kubeconfig --region region-code --name atai-platform
```

An example output is as follows.

None

```
Added new context arn:aws:eks:region-code:111122223333:cluster/atai-platform to
/home/username/.kube/config
```

2. Confirm communication with your cluster by running the following command.

```
None  
kubectl get svc
```

An example output is as follows.

```
None  
NAME          TYPE          CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE  
kubernetes    ClusterIP     10.100.0.1   <none>        443/TCP    28h
```

Step 4: Get the default cluster security group

After the cluster is created, you must **retrieve the security group ID** assigned by AWS. This ID is required when creating Launch Templates for your Managed Node Groups, to ensure proper networking and access between the control plane and the worker nodes.

You can determine the ID of your cluster security group in the AWS Management Console under the cluster's Networking section. Or, you can do so by running the following AWS CLI command.

```
None  
aws eks describe-cluster --name atai-platform --query  
cluster.resourcesVpcConfig.clusterSecurityGroupId
```

EKS managed node group configuration

Prerequisites

1. EKS cluster is ACTIVE
2. IAM role for node groups created (see Step 1 below)
3. Security group for nodes (see Step 2 below)
4. An existing VPC and subnets that meet Amazon EKS requirements
5. One private subnets in your VPC:
 - a. Recommended naming:
 - i. atai-platform-vpc-private-us-west-2a,
 - b. Recommended CIDR blocks: /20 per subnet
 - i. Example: 10.5.0.0/20 (subnet 1)
 - c. Subnets must have routes to NAT Gateway or Internet Gateway for outbound internet access
6. The kubectl command line tool is required. The version can be the same as or up to one minor version earlier or later than the Kubernetes version of your cluster.
7. Version 2.12.3 or later or version 1.27.160 or later of the AWS Command Line Interface (AWS CLI) installed and configured on your device.
8. An IAM principal with permissions to create and describe an Amazon EKS cluster, and create Managed Node Groups.

Step 1: Creating the Amazon EKS node IAM role

1. Run the following command to create an IAM trust policy JSON file.

```
None
cat > node-role-trust-relationship.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole"
      ],
      "Principal": {
        "Service": [
          "ec2.amazonaws.com"
        ]
      }
    }
  ]
}
EOF
```

2. Create the IAM role.

```
None
aws iam create-role \
  --role-name atai-platform-eks-node-role \
  --assume-role-policy-document file://"node-role-trust-relationship.json"
```

3. Attach two required IAM managed policies to the IAM role.

```
None
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name atai-platform-eks-node-role

aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryPullOnly \
  --role-name atai-platform-eks-node-role
```

```
aws iam attach-role-policy \  
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \  
  --role-name atai-platform-eks-node-role
```

4. Attach one of the following IAM policies to the IAM role depending on which IP family you created your cluster with. In this manual, we created the cluster with IPv4, therefore run the following command:

None

```
aws iam attach-role-policy \  
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \  
  --role-name atai-platform-eks-node-role
```

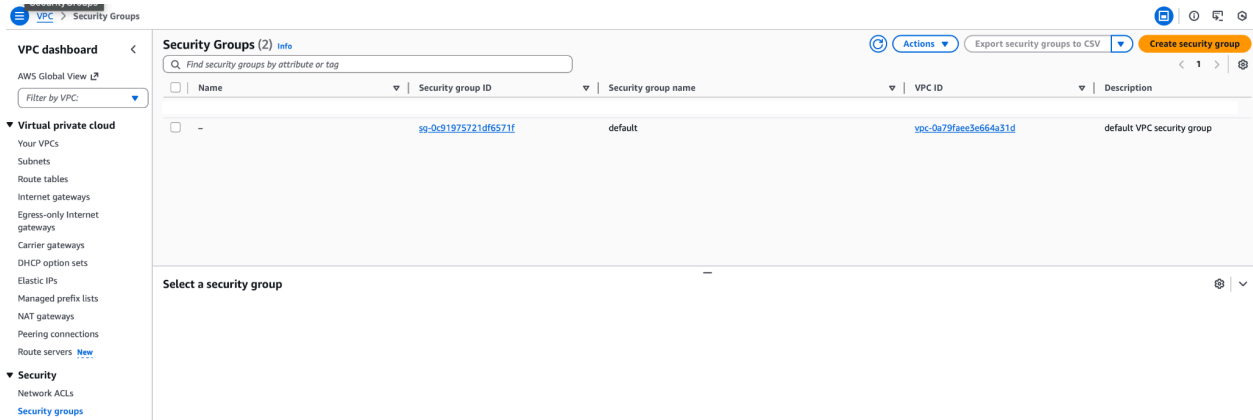
5. Attach the following IAM policy to the IAM role to allow SSH access to the nodes through AWS Session Manager (SSM) for debugging purposes.

None

```
aws iam attach-role-policy \  
  --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore \  
  --role-name atai-platform-eks-node-role
```

Step 2: Creating Node Security Group

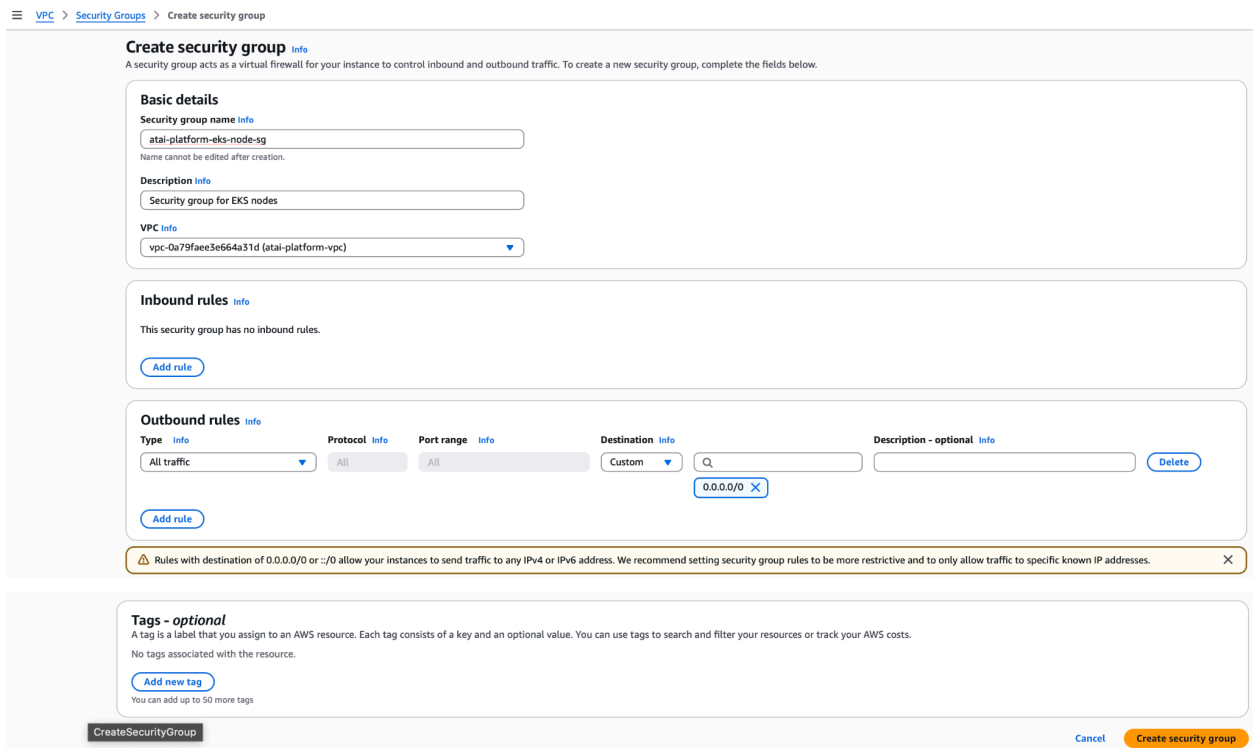
6. Go to VPC → Security Groups → Create security group



7. Name: atai-platform-eks-node-sg (or your name)

8. Description: Security group for EKS nodes

9. VPC: Select your VPC from section VPC configuration Step 1



10. Click on **Create security group**

Step 2.1 Add self rules to the node security group

A "self rule" in a security group allows traffic from the same security group.

1. Select your security group configured in Step 2 and click on **Edit inbound rules**

The screenshot shows the AWS IAM console interface for a security group. At the top, a green notification bar states: "Security group (sg-049eb8e8d3019b45f | atai-platform-eks-node-sg) was created successfully". Below this, the security group name "sg-049eb8e8d3019b45f - atai-platform-eks-node-sg" is displayed. The "Details" section provides the following information:

Security group name atai-platform-eks-node-sg	Security group ID sg-049eb8e8d3019b45f	Description Security group for EKS nodes	VPC ID vpc-0a79faee3e664a31d
Owner 716124474177	Inbound rules count 0 Permission entries	Outbound rules count 1 Permission entry	

The "Inbound rules" tab is active, showing a search bar and a table with the following columns: Name, Security group rule ID, IP version, Type, Protocol, Port range, Source, and Description. The table is currently empty, displaying the message "No security group rules found".

2. Inbound rules: Add rule:

- a. Rule 1: All TCP
 - i. Type: All TCP
 - ii. Source: Custom
 1. In the Source field, select Custom (not CIDR blocks or IP addresses).
 2. Click on the blank textbox next to Source
 3. Navigate to Security groups.
 4. Select the same security group you're currently configuring.
 - iii. Description: Allow all TCP traffic between nodes
- b. Rule 2: All UDP
 - i. Type: All UDP
 - ii. Source: Custom
 1. In the Source field, select Custom (not CIDR blocks or IP addresses).
 2. Click on the blank textbox next to Source
 3. Navigate to Security groups.
 4. Select the same security group you're currently configuring.
 - iii. Description: Allow all UDP traffic between nodes
- c. Rule 3: ICMP
 - i. Type: All ICMP - IPv4
 - ii. Port: None
 - iii. Source: Custom
 1. In the Source field, select Custom (not CIDR blocks or IP addresses).
 2. Click on the blank textbox next to Source
 3. Navigate to Security groups.
 4. Select the same security group you're currently configuring.
 - iv. Description: Allow all ICMP traffic between nodes

3. Click on **Save rules**

☰ VPC > Security Groups > sg-049eb8e8d3019b45f - atai-platform-eks-node-sg > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
-	Custom TCP	TCP	0 - 65535	Custom	sg-049eb8e8d3019b45f Allow all TCP traffic between node	Delete
-	Custom UDP	UDP	0 - 65535	Custom	sg-049eb8e8d3019b45f Allow all UDP traffic between node	Delete
-	All ICMP - IPv4	ICMP	All	Custom	sg-049eb8e8d3019b45f Allow all ICMP traffic between node	Delete

[Add rule](#)

[View changes](#) [Save rules](#)

Source dropdown menu:

- 0.0.0.0/24
- 0.0.0.0/32
- ::/0
- ::/16
- ::/32
- ::/48
- ::/64
- Security Groups**
- atai-rds-sg | sg-0463527653dd99e61
- atai-valkey-sg | sg-0a973800e58d354aa
- eks-cluster-sg-atai-platform-122654107 | sg-013f3e0903384fbee
- eks-cluster-sg-atai-platform-122654107
- default | sg-0c91975721df6571f
- launch-wizard-1 | sg-0bbfdd5da98f78fe
- atai-platform-eks-node-sg | sg-049eb8e8d3019b45f
- Prefix lists

Note: Creating Self-Referencing Security Group Rules

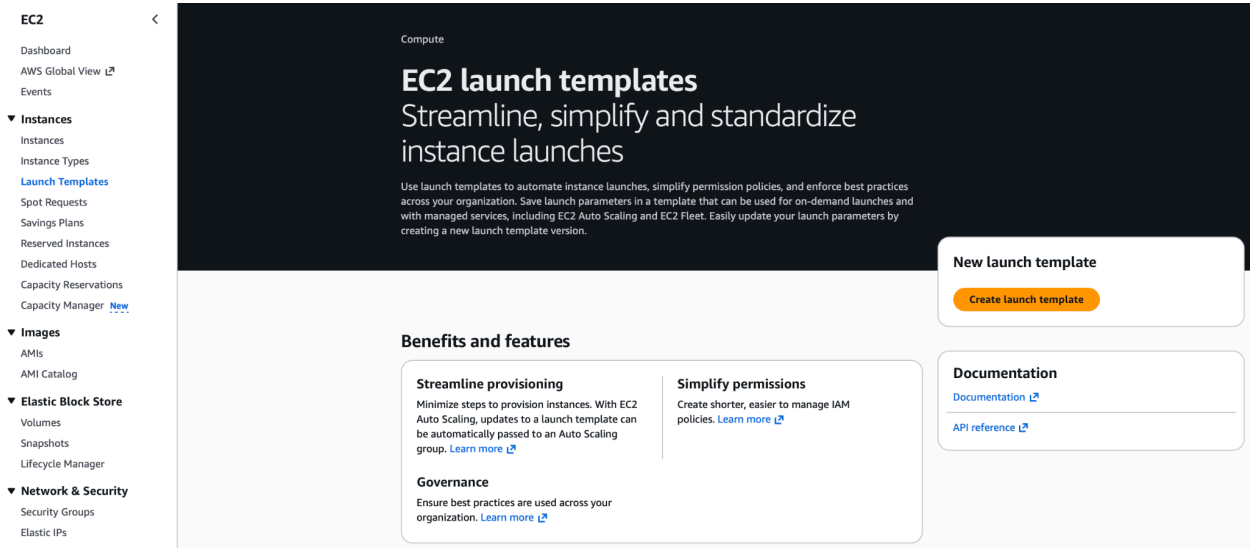
When adding ingress rules that allow traffic from the same security group (self-rule):

1. In the Source field, select Custom (not CIDR blocks or IP addresses).
2. Open the dropdown menu.
3. Navigate to Security groups.
4. Select the same security group you're currently configuring.

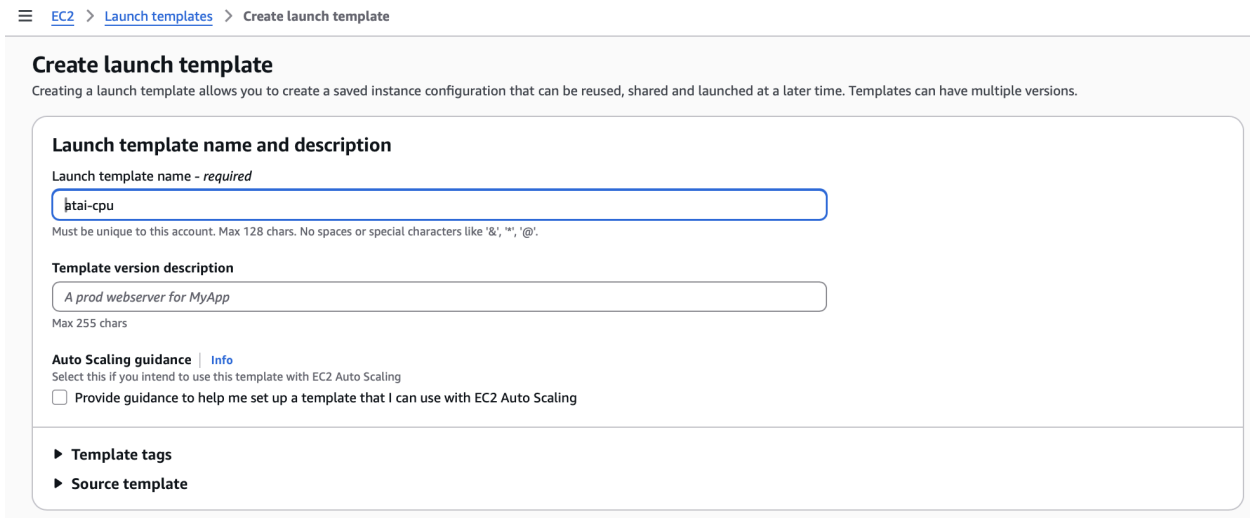
Step 3: Launch templates

Step 3.1 CPU Node Group

1. Go to EC2 → Launch template → **Create Launch Template**



2. Name: atai-cpu



Launch template contents

Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose **Browse more AMIs**.

Q Search our full catalog including 1000s of application and OS images

Recents | Quick Start

Don't include in launch template Recently launched Currently in use

[Browse more AMIs](#)
Including AMIs from AWS, Marketplace and the Community

▼ Instance type [Info](#) | [Get advice](#) Advanced

Instance type

Don't include in launch template

All generations [Compare instance types](#)

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name

Don't include in launch template [Create new key pair](#)

3. Network settings

- a. Select the security group created in the Step 2
- b. Select the default cluster security group that you get from the section **EKS cluster configuration - Step 4: Get the default cluster security group.**

▼ Network settings [Info](#)

Subnet [Info](#)

Don't include in launch template [Create new subnet](#)

When you specify a subnet, a network interface is automatically added to your template.

Availability Zone [Info](#)

Don't include in launch template [Enable additional zones](#)

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Select existing security group Create security group

Security groups [Info](#)

Select security groups

atai-platform-eks-node-sg sg-049eb8e8d3019b45f
VPC: vpc-0a79faee3e664a31d

eks-cluster-sg-atai-platform-122654107 sg-013f3e0903384fba
VPC: vpc-0a79faee3e664a31d

Hide all selected

► Advanced network configuration

4. Storage (Volumes) - Click on **Add new volume**

▼ Storage (volumes) [Info](#)

No volume details are currently included in this template. Add a new volume to include it in the launch template

[Add new volume](#)

a. Volume 1 configuration

- i. Size: 100
- ii. Volume type: gp3
- iii. Device name:
 1. Open the dropdown menu and click on **Specify a custom value**
 2. Value: /dev/xvda

Specify a Device name value ✕

Specifying a custom value allows you to create a template that can be used in other accounts

Device name

[Cancel](#)

[Save](#)

- iv. IOPS: 3000
- v. Throughput: 125

▼ Storage (volumes) [Info](#)

EBS Volumes

[Hide details](#)

▼ Volume 1 (Custom) [Remove](#)

Storage type [Info](#)

EBS

Device name - *required* [Info](#)

/dev/xvda

Snapshot [Info](#)

Don't include in launch template

Size (GiB) [Info](#)

100

Volume type [Info](#)

gp3

IOPS [Info](#)

3000

Delete on termination [Info](#)

Yes

Encrypted [Info](#)

Don't include in launch template

KMS key [Info](#)

Don't include in launch template

KMS keys are only applicable when encryption is set on this volume.

Throughput [Info](#)

125

Volume initialization rate - *new, optional* [Info](#)

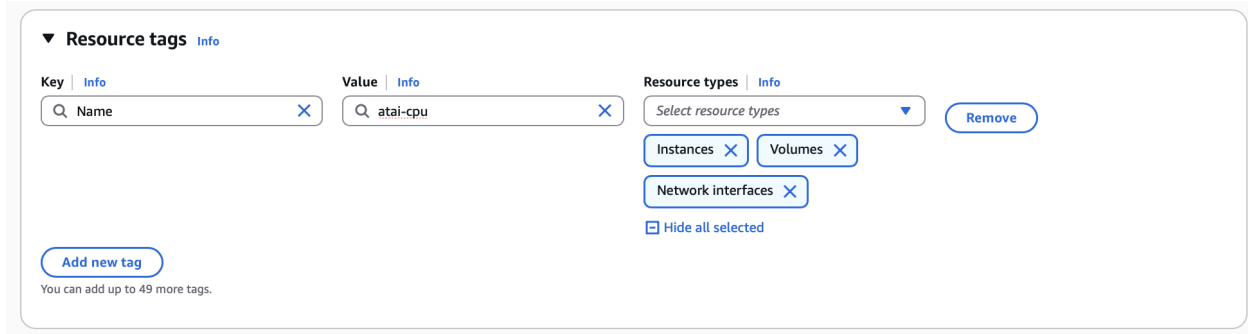
Enter a value

Min: 100 MiB/s, Max: 300 MiB/s. Additional charges apply [L2](#)

[i](#) Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage ✕

[Add new volume](#)

5. Resource tags
 - a. Key: Name
 - b. Value: atai-cpu
 - c. Resource types
 - i. Instances
 - ii. Volumes
 - iii. Network Interfaces



▼ Resource tags [Info](#)

Key [Info](#) Value [Info](#) Resource types [Info](#)

Q Name X Q atai-cpu X Select resource types

Remove

Instances X Volumes X

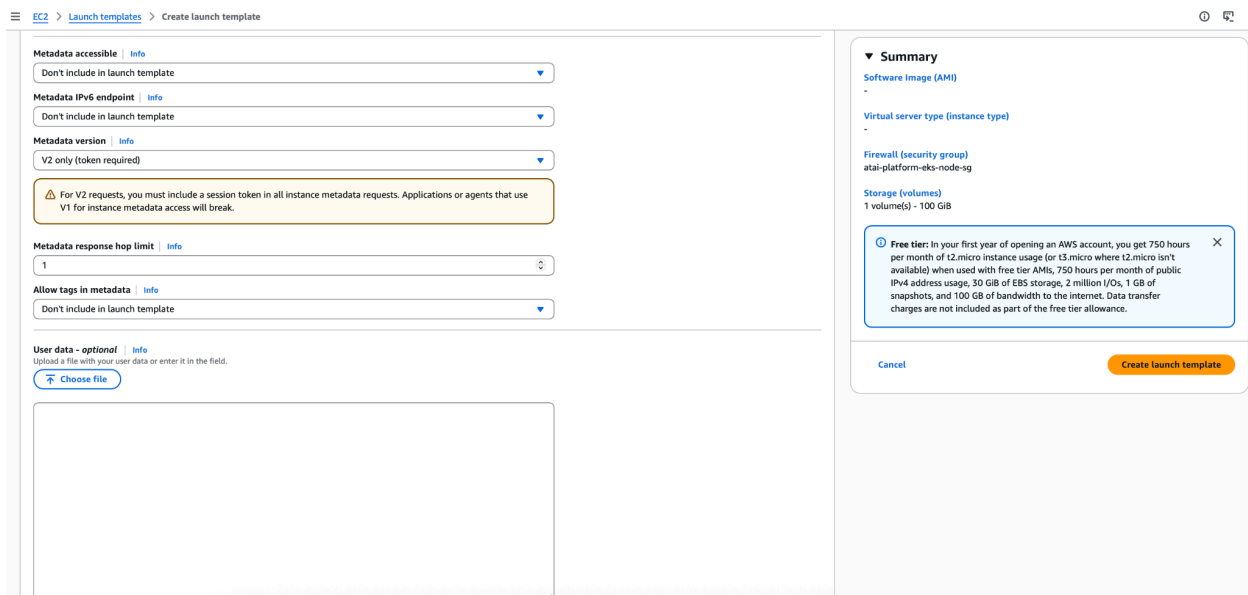
Network interfaces X

Hide all selected

Add new tag

You can add up to 49 more tags.

6. Advanced details
 - a. Metadata version: V2
 - b. Metadata response hop limit: 1



EC2 > Launch templates > Create launch template

Metadata accessible [Info](#)

Don't include in launch template

Metadata IPv6 endpoint [Info](#)

Don't include in launch template

Metadata version [Info](#)

V2 only (token required)

⚠ For V2 requests, you must include a session token in all instance metadata requests. Applications or agents that use V1 for instance metadata access will break.

Metadata response hop limit [Info](#)

1

Allow tags in metadata [Info](#)

Don't include in launch template

User data - optional [Info](#)

Upload a file with your user data or enter it in the field.

Choose file

▼ Summary

Software Image (AMI)

-

Virtual server type (instance type)

-

Firewall (security group)

atai-platform-eks-node-ig

Storage (volumes)

1 volume(s) - 100 GiB

📄 Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GiB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included as part of the free tier allowance.

Cancel Create launch template

7. Click on **Create Launch Template**

Step 3.2 GPU Node Group

8. Go to EC2 → Launch template → Create Launch Template

The screenshot shows the AWS Management Console page for 'EC2 launch templates'. The left-hand navigation pane is visible, showing categories like 'Instances', 'Images', 'Elastic Block Store', and 'Network & Security'. The main content area has a dark header with the text 'EC2 launch templates Streamline, simplify and standardize instance launches'. Below this, there are three columns of information: 'Benefits and features' (with sub-sections for Streamline provisioning, Simplify permissions, and Governance), 'New launch template' (with a 'Create launch template' button), and 'Documentation' (with links for 'Documentation' and 'API reference').

9. Name: atai-gpu

The screenshot shows the 'Create launch template' form in the AWS Management Console. The breadcrumb trail at the top reads 'EC2 > Launch templates > Create launch template'. The main heading is 'Create launch template', followed by a sub-heading 'Launch template name and description'. The form contains several input fields and options:

- Launch template name - required:** A text input field containing 'atai-gpu'. Below it, a note states: 'Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.'
- Template version description:** A text input field containing 'A prod webserver for MyApp'. Below it, a note states: 'Max 255 chars'.
- Auto Scaling guidance:** A section with an 'Info' link and a checkbox labeled 'Provide guidance to help me set up a template that I can use with EC2 Auto Scaling'. The checkbox is currently unchecked.
- Template tags:** A section with a right-pointing arrow and the text 'Template tags'.
- Source template:** A section with a right-pointing arrow and the text 'Source template'.

Launch template contents

Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose **Browse more AMIs**.

Q Search our full catalog including 1000s of application and OS images

Recents | Quick Start

Don't include in launch template

Recently launched

Currently in use

[Browse more AMIs](#)
Including AMIs from
AWS, Marketplace and
the Community

▼ Instance type [Info](#) | [Get advice](#)

[Advanced](#)

Instance type

Don't include in launch template

All generations

[Compare instance types](#)

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name

Don't include in launch template

[Create new key pair](#)

10. Network settings

- Select the security group created in the Step 2
- Select the default cluster security group that you get from the section EKS cluster configuration Step 4: Get the default cluster security group.

▼ Network settings [Info](#)

Subnet [Info](#)

Don't include in launch template

[Create new subnet](#)

When you specify a subnet, a network interface is automatically added to your template.

Availability Zone [Info](#)

Don't include in launch template

[Enable additional zones](#)

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Select existing security group

Create security group

Security groups [Info](#)

Select security groups

atai-platform-eks-node-sg sg-049eb8e8d3019b45f [×](#)
VPC: vpc-0a79faee3e664a31d

eks-cluster-sg-atai-platform-122654107 sg-013f3e0903384fbea [×](#)
VPC: vpc-0a79faee3e664a31d

[Compare security group rules](#)

Hide all selected

► Advanced network configuration

11. Storage (Volumes) - Click on **Add new volume**

▼ Storage (volumes) [Info](#)

No volume details are currently included in this template. Add a new volume to include it in the launch template

[Add new volume](#)

a. Volume 1 configuration

- i. Size: 100
- ii. Volume type: gp3
- iii. Device name:
 1. Open the dropdown menu and click on **Specify a custom value**
 2. Value: /dev/xvda

Specify a Device name value ✕

Specifying a custom value allows you to create a template that can be used in other accounts

Device name

[Cancel](#)

[Save](#)

- iv. IOPS: 3000
- v. Throughput: 125

▼ Storage (volumes) [Info](#)

EBS Volumes

[Hide details](#)

▼ Volume 1 (Custom) [Remove](#)

Storage type [Info](#)

EBS

Device name - *required* [Info](#)

/dev/xvda

Snapshot [Info](#)

Don't include in launch template

Size (GiB) [Info](#)

100

Volume type [Info](#)

gp3

IOPS [Info](#)

3000

Delete on termination [Info](#)

Yes

Encrypted [Info](#)

Don't include in launch template

KMS key [Info](#)

Don't include in launch template

KMS keys are only applicable when encryption is set on this volume.

Throughput [Info](#)

125

Volume initialization rate - *new, optional* [Info](#)

Enter a value

Min: 100 MIB/s, Max: 300 MIB/s. [Additional charges apply](#)

[i](#) Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage ✕

[Add new volume](#)

12. Resource tags

- a. Key: Name
- b. Value: atai-gpu
- c. Resource types
 - i. Instances
 - ii. Volumes
 - iii. Network Interfaces

▼ Resource tags [Info](#)

Key [Info](#) Value [Info](#) Resource types [Info](#)

Q Name X Q atai-gpu X Select resource types Remove

Instances X Volumes X

Network interfaces X

Hide all selected

Add new tag

You can add up to 49 more tags.

13. Advanced details

- a. Metadata version: V2
- b. Metadata response hop limit: 1

EC2 > Launch templates > Create launch template

Metadata accessible [Info](#)

Don't include in launch template

Metadata IPv6 endpoint [Info](#)

Don't include in launch template

Metadata version [Info](#)

V2 only (token required)

⚠ For V2 requests, you must include a session token in all instance metadata requests. Applications or agents that use V1 for instance metadata access will break.

Metadata response hop limit [Info](#)

1

Allow tags in metadata [Info](#)

Don't include in launch template

User data - optional [Info](#)

Upload a file with your user data or enter it in the field.

Choose file

▼ Summary

Software Image (AMI)

-

Virtual server type (instance type)

-

Firewall (security group)

atai-platform-eks-node-sg

Storage (volumes)

1 volume(s) - 100 GiB

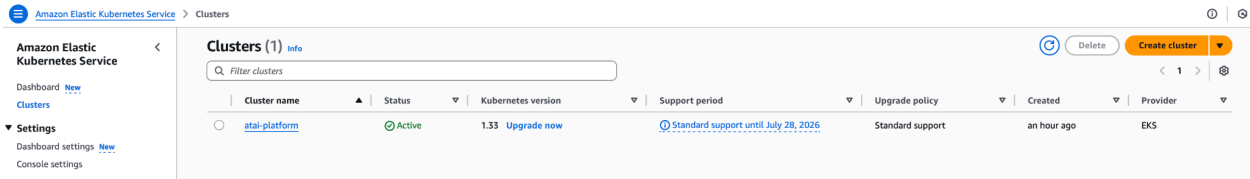
📄 Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included as part of the free tier allowance.

Cancel Create launch template

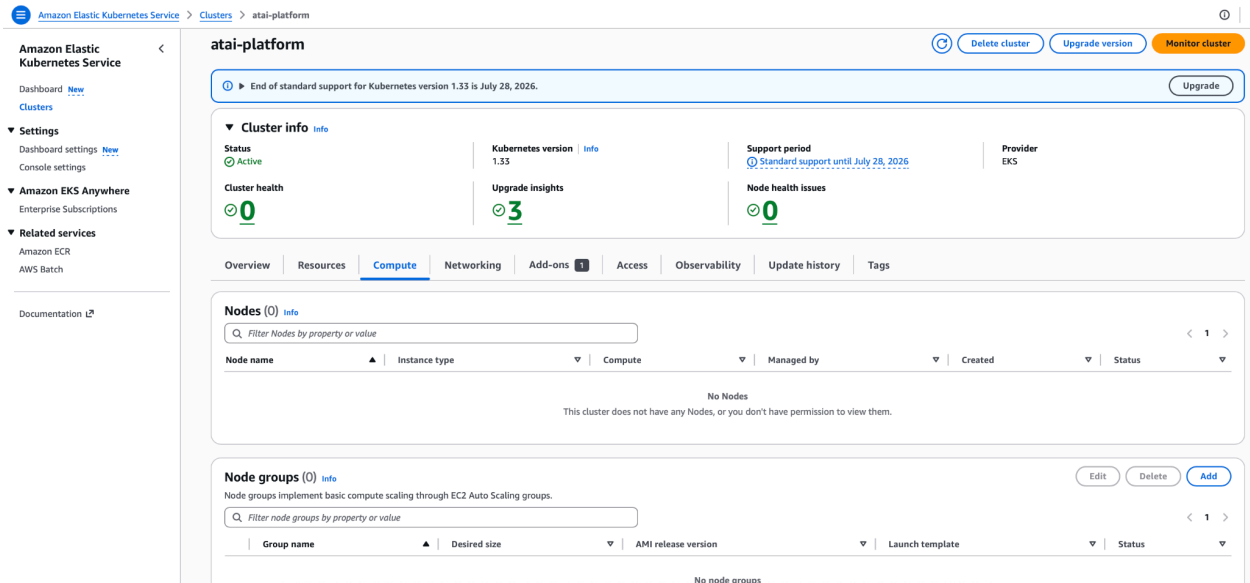
14. Click on **Create Launch Template**

Step 4: Create CPU Node Group

1. EKS Console → Your cluster



2. Select Compute tab → Node groups section → Add node group



3. Configure node group:

- a. Node group name: atai-cpu
- b. Node IAM role: Select the role from Step 1
- c. Launch template: Select the Launch template created in the Step 3.1

Amazon Elastic Kubernetes Service > Clusters > atai-platform > Add node group

Step 1
Configure node group
Step 2
Set compute and scaling configuration
Step 3
Specify networking
Step 4
Review and create

Configure node group [info](#)

A node group is a group of EC2 instances that supply compute capacity to your Amazon EKS cluster. You can add multiple node groups to your cluster.

Node group configuration

These properties cannot be changed after the node group is created.

Name
Assign a unique name for this node group.

The node group name should begin with letter or digit and can have any of the following characters: the set of Unicode letters, digits, hyphens and underscores. Maximum length of 63.

Node IAM role [info](#)
Select the IAM role that will be used by the nodes. To create a new role, go to the [IAM console](#).

[Create recommended role](#)

The selected role must not be used by a self-managed node group as this could lead to a service interruption upon managed node group deletion. [Learn more](#)

Launch template [info](#)

These properties cannot be changed after the node group is created.

Use launch template
Configure this node group using an EC2 launch template.

Launch Template Name
To create a new launch template, go to the corresponding page in the [EC2 console](#).

[Create launch template](#)

Launch template version
Select the launch template version.

[Create launch template](#)

4. Configure Kubernetes labels and taints:

- a. Labels: Add label:
 - i. Key: archetypeai.io
 - ii. Value: cpu
- b. Taints: None (leave empty)

Kubernetes labels [info](#)

Key	Value	
<input type="text" value="archetypeai.io"/>	<input type="text" value="cpu"/>	Remove label

[Add label](#)

Remaining labels available to add: 49

Kubernetes taints [info](#)

This node group does not have any taints.

[Add taint](#)

Remaining taints available to add: 50

Tags [info](#)

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 tags.

[Cancel](#) [Next](#)

5. Node group compute configuration
 - a. AMI type: Amazon Linux 2023 (AL2023_x86_64_STANDARD)
 - b. Capacity type: On-Demand
 - c. Instance types: m6i.4xlarge
 - d. Disk size: Specified in the Launch template

The screenshot shows the 'Set compute and scaling configuration' step in the AWS console. On the left, a progress bar indicates four steps: Step 1 (Configure node group), Step 2 (Set compute and scaling configuration - currently active), Step 3 (Specify networking), and Step 4 (Review and create). The main content area is titled 'Set compute and scaling configuration' and contains the following sections:

- Node group compute configuration:** A note states 'These properties cannot be changed after the node group is created.'
- AMI type:** A dropdown menu is set to 'Amazon Linux 2023 (x86_64) Standard (AL2023_x86_64_STANDARD)'.
- Capacity type:** A dropdown menu is set to 'On-Demand'.
- Instance types:** A search box contains 'm6i.4xlarge'. Below it, a card displays the instance type details: 'm6i.4xlarge', 'vCPU: 16 vCPUs', 'Memory: 64 GiB', 'Network: Up to 12.5 Gigabit', 'Max ENI: 8', and 'Max IPs: 240'.
- Disk size:** A dropdown menu is set to 'Specified in launch template'.

6. Node group scaling configuration:
 - a. Desired size: 1 (or 2 for production)
 - b. Minimum size: 1
 - c. Maximum size: 5

The screenshot shows the 'Node group scaling configuration' section of the AWS console. It contains three input fields for node counts:

- Desired size:** A numeric input field is set to '1'. Below it, it says 'nodes' and 'Desired node size must be greater than or equal to 0'.
- Minimum size:** A numeric input field is set to '1'. Below it, it says 'nodes' and 'Minimum node size must be greater than or equal to 0'.
- Maximum size:** A numeric input field is set to '5'. Below it, it says 'nodes' and 'Maximum node size must be greater than or equal to 1 and cannot be lower than the minimum size'.

7. Node group update configuration:
 - a. Maximum unavailable: Percentage
 - b. Value: 33%
 - c. Update strategy: Default

The screenshot shows the 'Node group update configuration' section of the AWS console. It contains the following settings:

- Maximum unavailable:** Two radio buttons are present: 'Number' (unselected) and 'Percentage' (selected). Below the 'Percentage' button, it says 'Specify a percentage'.
- Value:** A numeric input field is set to '33' with a '%' symbol to its right. Below it, it says 'Percentage must be between 1 to 100'.
- Update strategy:** Two radio buttons are present: 'Default' (selected) and 'Minimal' (unselected).

8. Node group auto repair configuration: Disable

Node auto repair configuration info

When node auto repair is enabled, Amazon EKS continuously monitors the health of the nodes within a managed node group. This feature automatically detects and replaces nodes when issues occur.

The node auto repair feature reacts to the Ready condition of the kubelet and any node object manual deletions. It can detect more node conditions for repair when the node monitoring agent is also installed. [Go to Add-ons](#)

Enable node auto repair

Cancel Previous **Next**

9. Node group network configuration

a. Subnets: Select your private subnets:

i. atai-platform-vpc-private-us-west-2a (10.5.0.0/20)

It's important to select the private subnet in the same AZs as your Valkey clusters and RDS PostgreSQL cluster.

Step 1 Configure node group

Step 2 Set compute and scaling configuration

Step 3 **Specify networking**

Step 4 Review and create

Specify networking

Node group network configuration

These properties cannot be changed after the node group is created.

Subnets info

Specify the subnets in your VPC where your nodes will run. To create a new subnet, go to the corresponding page in the [VPC console](#).

Select subnets

subnet-0bd36810cb1c67e50 | atai-platform-vpc-private-us-west-2a us-west-2a 10.5.0.0/20

Clear selected subnets

EC2 Key Pair

Select an EC2 key pair to allow secure remote access to your nodes. To create a new EC2 key pair, go to the corresponding page in the [EC2 console](#).

Not specified in launch template

Without a key pair you will not be able to directly connect to nodes after they are created.

Cancel Previous **Next**

10. Review and click on **Create**

Wait for node group to be **ACTIVE** (5-10 minutes)

Amazon Elastic Kubernetes Service > Clusters > atai-platform > Add node group

Step 1: Configure node group

Step 2: Set compute and scaling configuration

Step 3: Specify networking

Step 4: Review and create

Review and create

Step 1: Node group

Node group configuration

Name atai-cpu	Node IAM role arn:aws:iam::716124474177:role/atai-platform-eks-node-role
------------------	---

Kubernetes labels (1)

Key	Value
archetypeai.io	cpu

Kubernetes taints (0)

Filter by key, value or effect

Key	Value	Effect
No taints This node group does not have any Kubernetes taints.		

Tags (0)

Tags that you've added. Each tag consists of a key and an optional value.

Key	Value
No tags This node group does not have any tags.	

Step 2: Compute and scaling configuration

Node group compute configuration

Capacity type On-Demand	Instance types m6i.4xlarge	Disk size Specified in launch template
AMI type Amazon Linux 2023 (x86_64) Standard (AL2023_x86_64_STANDARD)		

Node group scaling configuration

Desired size 1 node	Minimum size 1 node	Maximum size 5 nodes
------------------------	------------------------	-------------------------

Node group update configuration

Maximum unavailable 35 %	Update strategy Default
-----------------------------	----------------------------

Node auto repair configuration

Node auto repair Disabled

Step 3: Networking

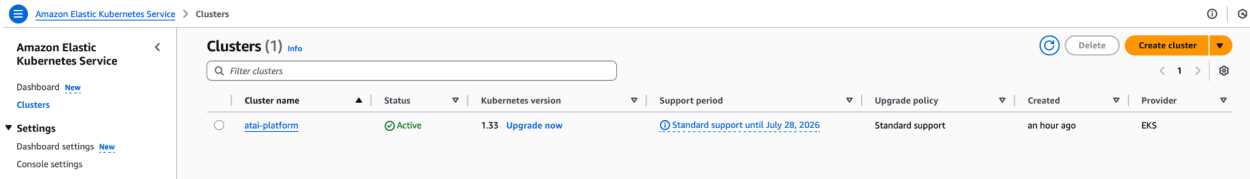
Node group network configuration

Subnets subnet-0bd36810cb1c67e50	Configure remote access to nodes off
-------------------------------------	---

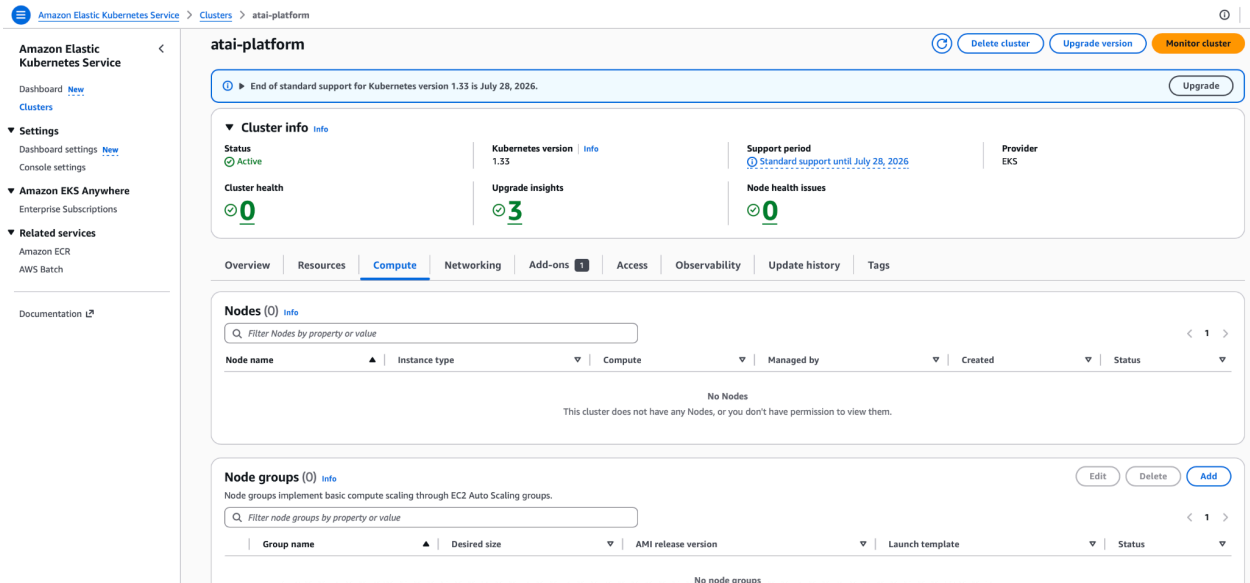
Cancel Previous **Create**

Step 5: Create GPU Node Group

1. EKS Console → Your cluster



2. Select Compute tab → Node groups section → Add node group



3. Configure node group:

- a. Node group name: atai-gpu
- b. Node IAM role: Select the role from Step 1
- c. Launch template: Select the Launch template created in the Step 3.2

Amazon Elastic Kubernetes Service > Clusters > atai-platform > Add node group

Step 1 **Configure node group**

Step 2 Set compute and scaling configuration

Step 3 Specify networking

Step 4 Review and create

Configure node group Info

A node group is a group of EC2 instances that supply compute capacity to your Amazon EKS cluster. You can add multiple node groups to your cluster.

Node group configuration

These properties cannot be changed after the node group is created.

Name
Assign a unique name for this node group.

The node group name should begin with letter or digit and can have any of the following characters: the set of Unicode letters, digits, hyphens and underscores. Maximum length of 63.

Node IAM role Info
Select the IAM role that will be used by the nodes. To create a new role, go to the [IAM console](#).

[Create recommended role](#)

The selected role must not be used by a self-managed node group as this could lead to a service interruption upon managed node group deletion.
[Learn more](#)

Launch template

These properties cannot be changed after the node group is created.

Use launch template
Configure this node group using an EC2 launch template.

Launch Template Name
To create a new launch template, go to the corresponding page in the [EC2 console](#).

[Create launch template](#)

Launch template version
Select the launch template version.

[Refresh](#)

4. Configure Kubernetes labels and taints:

- a. Labels: Add label:
 - i. Key: archetypeai.io
 - ii. Value: gpu
- b. Taints:
 - i. Key: nvidia.com/gpu
 - ii. Value: present
 - iii. Effect: NoSchedule

Kubernetes labels Info

Key	Value	
<input type="text" value="archetypeai.io"/>	<input type="text" value="gpu"/>	Remove label

[Add label](#)

Remaining labels available to add: 49

Kubernetes taints Info

Key	Value	Effect	
<input type="text" value="nvidia.com/gpu"/>	<input type="text" value="present"/>	<input type="text" value="NoSchedule"/>	Remove taint

[Add taint](#)

Remaining taints available to add: 49

Tags Info

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 tags.

[Cancel](#) [Next](#)

5. Node group compute configuration

- a. AMI type: Amazon Linux 2023 (x86_64) Nvidia (AL2023_x86_64_NVIDIA)
- b. Capacity type: On-Demand
- c. Instance types: g6e.2xlarge
- d. Disk size: Specified in the Launch template

The screenshot shows the 'Set compute and scaling configuration' step in the AWS IAM console. On the left, a progress bar indicates four steps: 'Configure node group', 'Set compute and scaling configuration' (selected), 'Specify networking', and 'Review and create'. The main content area is titled 'Set compute and scaling configuration' and contains the following sections:

- Node group compute configuration:** A note states 'These properties cannot be changed after the node group is created.'
- AMI type:** A dropdown menu is set to 'Amazon Linux 2023 (x86_64) Nvidia (AL2023_x86_64_NVIDIA)'. A small 'Info' link is present.
- Capacity type:** A dropdown menu is set to 'On-Demand'.
- Instance types:** A search box contains 'g6e.2xlarge'. A tooltip is visible showing details: 'g6e.2xlarge', 'vCPU: 8 vCPUs', 'Memory: 64 GiB', 'Network: Up to 20 Gigabit', 'Max ENI: 4', and 'Max IPs: 60'.
- Disk size:** A text field contains 'Specified in launch template'.

6. Node group scaling configuration:

- a. Desired size: 4
- b. Minimum size: 4
- c. Maximum size: 10

The screenshot shows the 'Node group scaling configuration' step in the AWS IAM console. The main content area is titled 'Node group scaling configuration' and contains the following sections:

- Desired size:** A spinner box is set to '4 nodes'. A note below reads 'Desired node size must be greater than or equal to 0'.
- Minimum size:** A spinner box is set to '4 nodes'. A note below reads 'Minimum node size must be greater than or equal to 0'.
- Maximum size:** A spinner box is set to '20 nodes'. A note below reads 'Maximum node size must be greater than or equal to 1 and cannot be lower than the minimum size'.

7. Node group update configuration:

- a. Maximum unavailable: Percentage
- b. Value: 33%
- c. Update strategy: Default

The screenshot shows the 'Node group update configuration' step in the AWS IAM console. The main content area is titled 'Node group update configuration' and contains the following sections:

- Maximum unavailable:** A note states 'Set the maximum number or percentage of unavailable nodes to be tolerated during the node group version update.' There are two radio buttons: 'Number' (unselected) and 'Percentage' (selected).
- Value:** A spinner box is set to '33 %'. A note below reads 'Percentage must be between 1 to 100.'
- Update strategy:** There are two radio buttons: 'Default' (selected) and 'Minimal' (unselected).

8. Node group auto repair configuration: Disable

Node auto repair configuration Info

When node auto repair is enabled, Amazon EKS continuously monitors the health of the nodes within a managed node group. This feature automatically detects and replaces nodes when issues occur.

The node auto repair feature reacts to the Ready condition of the kubelet and any node object manual deletions. It can detect more node conditions for repair when the node monitoring agent is also installed. [Go to Add-ons](#)

Enable node auto repair

Cancel Previous Next

9. Node group network configuration

a. Subnets: Select your private subnets:

i. atai-platform-vpc-private-us-west-2a (10.5.0.0/20)

It's important to select the private subnet in the same AZs as your Valkey clusters and RDS PostgreSQL cluster.

Step 1
● Configure node group

Step 2
● Set compute and scaling configuration

Step 3
● **Specify networking**

Step 4
○ Review and create

Specify networking

Node group network configuration

These properties cannot be changed after the node group is created.

Subnets Info

Specify the subnets in your VPC where your nodes will run. To create a new subnet, go to the corresponding page in the [VPC console](#).

Select subnets

subnet-0bd36810cb1c67e50 | atai-platform-vpc-private-us-west-2a
us-west-2a 10.5.0.0/20

Clear selected subnets

EC2 Key Pair

Select an EC2 key pair to allow secure remote access to your nodes. To create a new EC2 key pair, go to the corresponding page in the [EC2 console](#).

Not specified in launch template

Without a key pair you will not be able to directly connect to nodes after they are created.

Cancel Previous Next

10. Review and click on **Create**

Wait for node group to be **ACTIVE** (5-10 minutes)

Amazon Elastic Kubernetes Service > Clusters > atai-platform > Add node group

Step 1: Configure node group

Step 2: Set compute and scaling configuration

Step 3: Specify networking

Step 4: Review and create

Review and create

Step 1: Node group

Node group configuration

Name	atai-gpu	Node IAM role	amaws:iam::716124474177:role/atai-platform-eks-node-role
------	----------	---------------	--

Kubernetes labels (1)

Key	Value
archetypeai.io	gpu

Kubernetes taints (1)

Filter by key, value or effect

Key	Value	Effect
nvidia.com/gpu	present	NoSchedule

Tags (0)

Tags that you've added. Each tag consists of a key and an optional value.

Key	Value
No tags	
This node group does not have any tags.	

Step 2: Compute and scaling configuration

Node group compute configuration

Capacity type	On-Demand	Instance types	g6e.zxlarge	Disk size	Specified in launch template
AMI type	Amazon Linux 2023 (x86_64) Nvidia (AL2023_x86_64_NVIDIA)				

Node group scaling configuration

Desired size	4 nodes	Minimum size	4 nodes	Maximum size	20 nodes
--------------	---------	--------------	---------	--------------	----------

Node group update configuration

Maximum unavailable	33 %	Update strategy	Default
---------------------	------	-----------------	---------

Node auto repair configuration

Node auto repair	Disabled
------------------	----------

Step 3: Networking

Node group network configuration

Subnets	subnet-0bd36810cb1c67e50	Configure remote access to nodes	off
---------	--------------------------	----------------------------------	-----

Cancel Previous Create

EKS configuration - Install the NVIDIA Device Plugin

The NVIDIA device plugin DaemonSet to enable GPU resource scheduling in Kubernetes.

Prerequisites

1. EKS cluster is ACTIVE
2. The kubectl command line tool is required. The version can be the same as or up to one minor version earlier or later than the Kubernetes version of your cluster.
3. An IAM principal with permissions to create and describe an Amazon EKS cluster

Step 1: Manual installation

1. Direct download from GitHub:

None

```
curl -L -o nvidia-device-plugin-v0.18.0.yml  
https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/v0.18.0/deployments/  
static/nvidia-device-plugin.yml
```

2. Then apply the Kubernetes manifest:

None

```
kubectl apply -f nvidia-device-plugin-v0.18.0.yml
```

3. Verify nodes are annotated with the nvidia label:

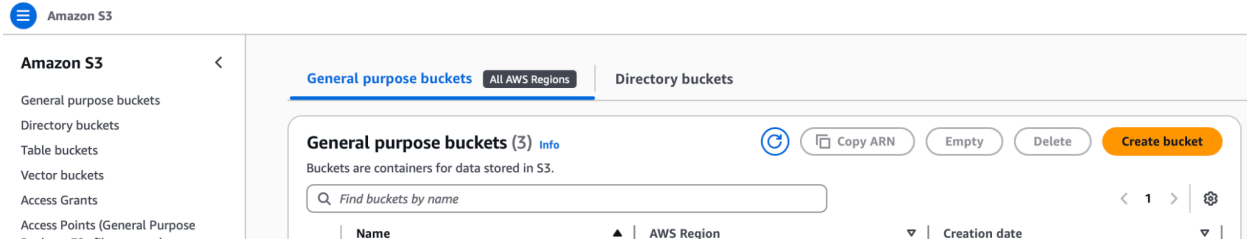
None

```
kubectl get nodes -o  
custom-columns="NAME:.metadata.name,GPU:.status.allocatable.nvidia\.com/gpu"
```

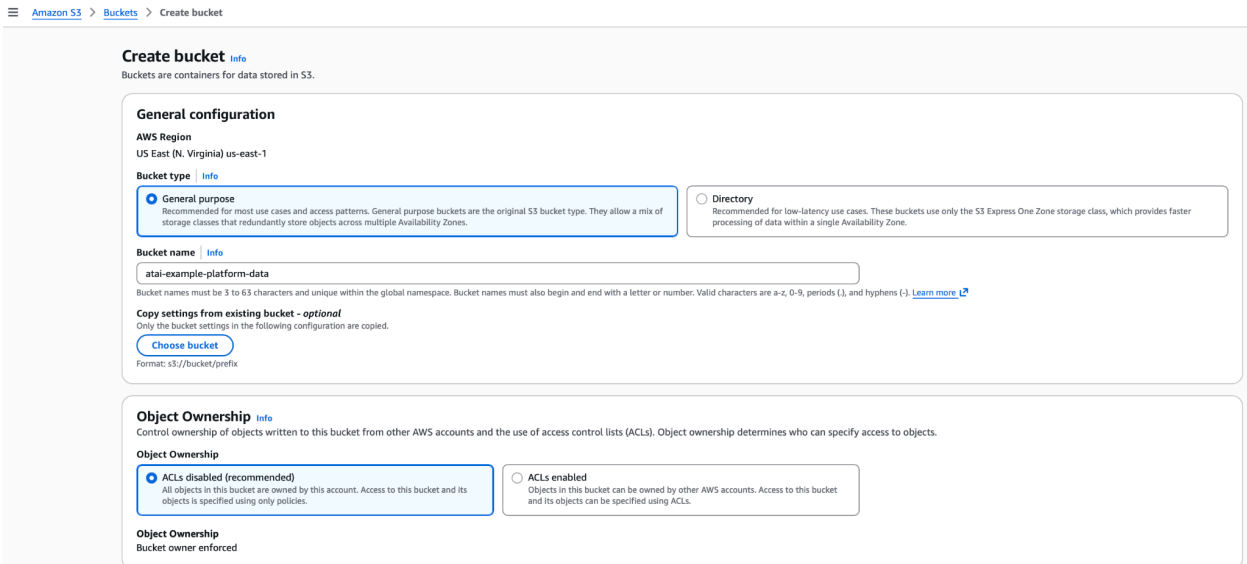
S3 configuration

Step 1: Create the platform-data bucket

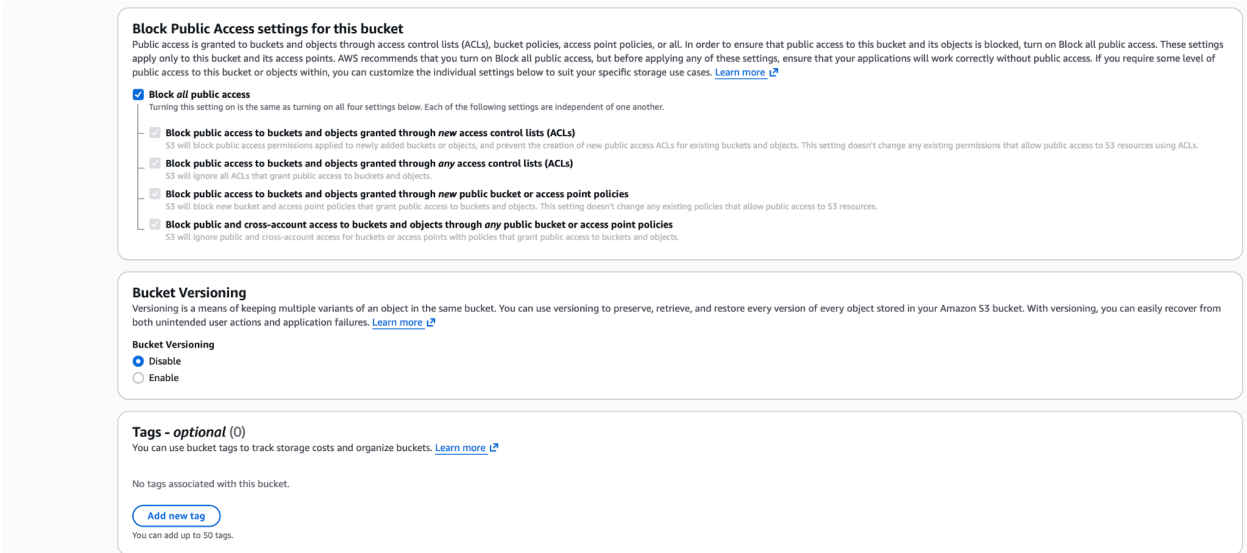
1. Go to the S3 dashboard and click on **Create Bucket**
2. Make sure, you are located in your home AWS region



3. Bucket name: atai-<CUSTOMER UNIQUE ID>-platform-data



4. Keep the selection in the **Block Public Access** setting for the bucket



5. Use the default Encryption configuration

Default encryption Info
Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type Info
Secure your objects with two separate layers of encryption. For details on pricing, see DSSE-KMS pricing on the Storage tab of the [Amazon S3 pricing page](#).

Server-side encryption with Amazon S3 managed keys (SSE-S3)
 Server-side encryption with AWS Key Management Service keys (SSE-KMS)
 Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)

Bucket Key
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

Disable
 Enable

► **Advanced settings**

ⓘ After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

Cancel **Create bucket**

6. Click on **Create bucket**

⚠ Store your bucket endpoint in a secure location. You will need them later in the *atai-platform* prerequisites, Step 3.2: Kubernetes Secrets Generation.

Note: The **platform-data** bucket will be required for following services:

- access-manager-service-backend
- api-service-health-node
- api-service-backend
- dfc-service-backend
- file-service-worker-node
- file-service-backend
- lens-node-worker-node
- lens-node-service-backend
- lens-service-backend

Note 2: The expected format for `<PLATFORM_S3_BUCKET>` is `s3://<BUCKET_NAME>`. These values will be referenced as:

```
None
[atai-platform-access-manager-service-backend]
...
PLATFORM_ASSETS_DIR=s3://<BUCKET_NAME>

[atai-platform-api-service-health-node]
...
PLATFORM_ASSETS_DIR=s3://<BUCKET_NAME>

[atai-platform-api-service-backend]
...
```

```
PLATFORM_ASSETS_DIR=s3://<BUCKET_NAME>

[atai-platform-dfc-service-backend]
...
COLD_STORAGE_ROOT_PATH=s3://<BUCKET_NAME>/dfc_service

[atai-platform-file-service-worker-node]
...
PLATFORM_ASSETS_DIR=s3://<BUCKET_NAME>

[atai-platform-file-service-backend]
...
PLATFORM_ASSETS_DIR=s3://<BUCKET_NAME>

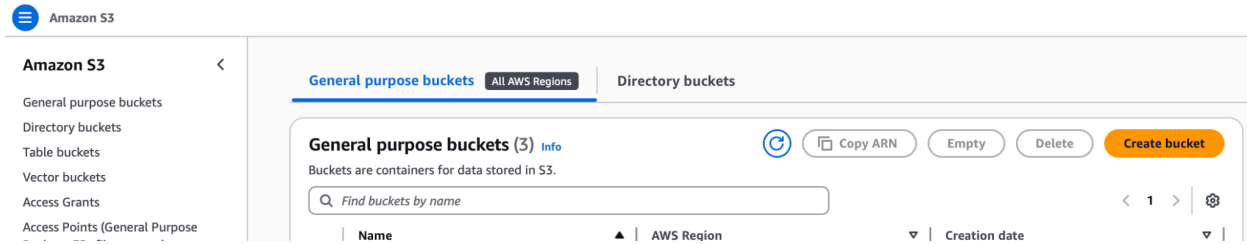
[atai-platform-lens-node-worker-node]
...
PLATFORM_ASSETS_DIR=s3://<BUCKET_NAME>

[atai-platform-lens-node-service-backend]
...
PLATFORM_ASSETS_DIR=s3://<BUCKET_NAME>

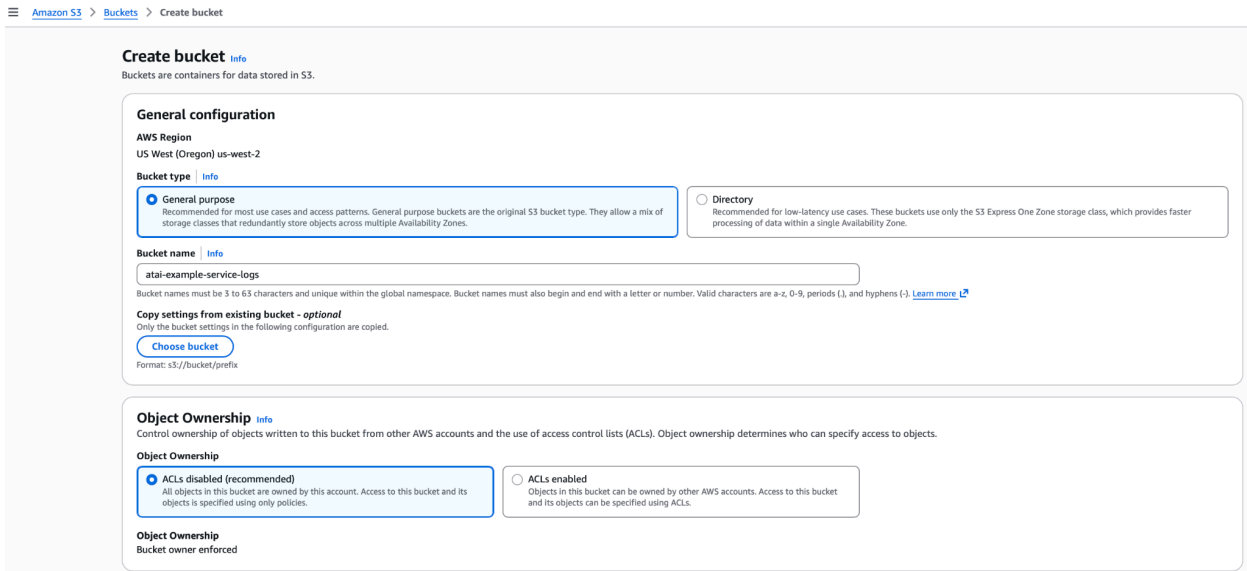
[atai-platform-lens-service-backend]
...
PLATFORM_ASSETS_DIR=s3://<BUCKET_NAME>
```

Step 2: Create the service logs bucket

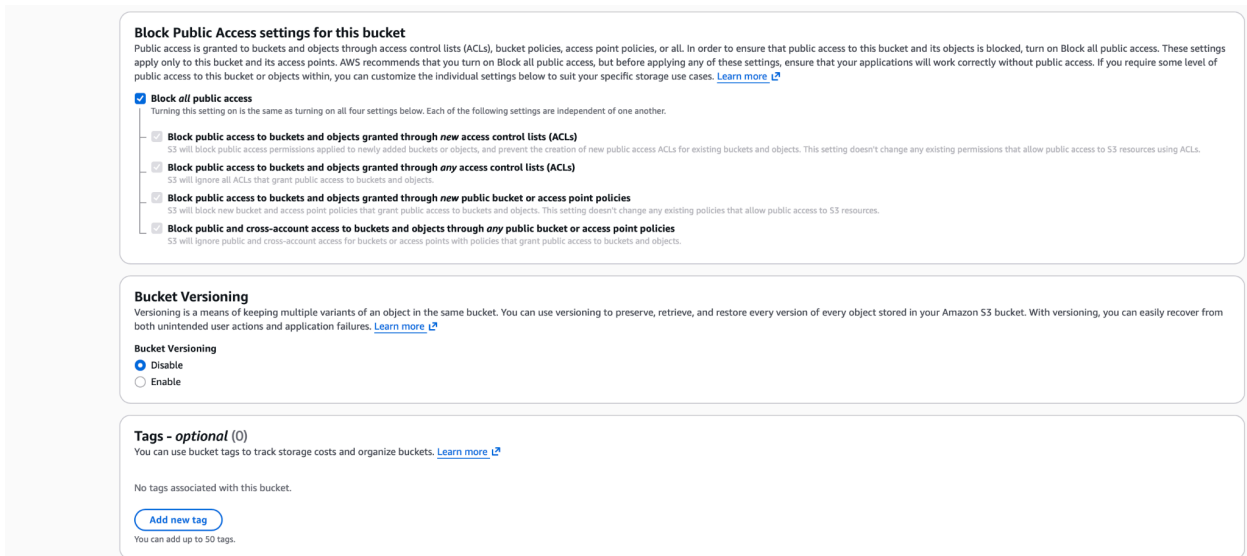
1. Go to the S3 dashboard and click on **Create Bucket**
2. Make sure, you are located in your home AWS region



3. Bucket name: atai-<CUSTOMER UNIQUE ID>-service-logs



4. Keep the selection in the **Block Public Access** setting for the bucket



5. Use the default Encryption configuration

Default encryption [Info](#)
Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type [Info](#)
Secure your objects with two separate layers of encryption. For details on pricing, see DSSE-KMS pricing on the Storage tab of the [Amazon S3 pricing page](#).
 Server-side encryption with Amazon S3 managed keys (SSE-S3)
 Server-side encryption with AWS Key Management Service keys (SSE-KMS)
 Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)

Bucket Key
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)
 Disable
 Enable

► **Advanced settings**

ⓘ After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

[Cancel](#) [Create bucket](#)

6. Click on **Create bucket**

⚠ Store your bucket endpoint in a secure location. You will need them later in the *atai-platform* prerequisites, Step 3.2: Kubernetes Secrets Generation.

Note: The **service-logs** bucket will be required for following services:

- access-manager-service-backend

Note 2: The expected format for `<SERVICE_LOGS_S3_BUCKET>` is `s3://<BUCKET_NAME>`. These values will be referenced as:

```
None
[atai-platform-access-manager-service-backend]
...
PLATFORM_LOGS_BUCKET=s3://<BUCKET_NAME>
```

Application Endpoints Configuration

Platform Architecture Overview

The Archetype platform consists of two main entry points that must be publicly accessible:

1. **Console:** The web-based user interface where users interact with the platform
2. **API:** The backend service that handles business logic and data processing

For proper operation, the API must be publicly accessible so the Console can communicate with it, and vice versa. This is a standard configuration for web applications where the frontend (Console) and backend (API) need to exchange data.

Required Public URLs

Once configured, the following URLs must be publicly accessible and secured:

1. **console-archetype.<your-domain>** - Console web application
2. **api-archetype.<your-domain>** - API backend service

Both URLs must be accessible over HTTPS with valid SSL certificates.

How Do I Expose My Services Publicly?

To make your Console and API services publicly accessible, you need to use a Kubernetes Ingress.

What is an Ingress?

An Ingress is a Kubernetes resource that defines routing rules for external traffic. It specifies which domain names should route to which services and on which paths. However, an Ingress resource by itself doesn't handle traffic, you need an Ingress Controller to process these rules and route the actual traffic.

An Ingress Controller acts as a reverse proxy that:

- Receives external HTTP/HTTPS traffic
- Reads Ingress resource rules
- Routes traffic to the appropriate backend services based on domain names and paths

Common Ingress Controllers: Popular Ingress Controller options include:

- AWS Load Balancer Controller - Creates AWS Application Load Balancers (ALB) or Network Load Balancers (NLB)
- NGINX Ingress Controller - A widely-used, feature-rich Ingress Controller
- Traefik - Another popular option

The Archetype platform uses a combination of AWS Load Balancer Controller and NGINX Ingress Controller to provide both AWS-native load balancer management and flexible traffic routing capabilities.

If you don't have an Ingress Controller configured: See Appendix [EKS configuration - Install the AWS Load Balancer controller](#) and then [EKS configuration - Install the NGINX ingress controller](#).

How Do I Secure the Traffic?

Once your services are exposed via Ingress, it's important to secure the traffic with SSL/TLS certificates. This ensures all communications are encrypted and secure.

Common Certificate Solutions: Ingress Controllers typically integrate with certificate management solutions:

- Let's Encrypt through Cert-Manager - Automated, free SSL certificates that are automatically renewed
- AWS Certificate Manager (ACM) - Native AWS-managed certificates

Cert-Manager is a Kubernetes add-on that automatically provisions, renews, and manages SSL certificates. It can integrate with Let's Encrypt to obtain free certificates automatically, or work with AWS ACM for AWS-managed certificates.

If you need help configuring certificates: See Appendix: [EKS configuration - Configure Cert-Manager and Let's Encrypt](#)

How Do I Configure DNS?

After your Ingress Controller is configured, it typically creates an AWS Network Load Balancer (NLB) or Application Load Balancer (ALB) as the entry point for your services.

DNS Configuration Steps:

1. Your Ingress Controller creates an AWS load balancer (NLB or ALB)
 - a. The load balancer receives a public DNS name or IP address
2. You need to add DNS records that point your domain names to this load balancer

Required DNS Records:

1. Add the following DNS records (A records or CNAME records) pointing to the load balancer created by your Ingress solution:
 - a. **console-archetype.<your-domain>** → Points to the load balancer
 - b. **api-archetype.<your-domain>** → Points to the load balancer

The exact DNS configuration depends on your DNS provider and whether your load balancer provides a DNS name (use CNAME) or an IP address (use A record).

Deployment Checklist

Before Helm Chart Installation:

- VPC and subnets** deployed with proper CIDR blocks
- 8 Valkey instances** created with correct names and versions
- 1 RDS PostgreSQL** instance created with Aurora engine
- EKS cluster** deployed with Kubernetes 1.33
- 1 CPU node group** deployed with m6i.4xlarge instances
- 4 GPU node group** deployed with g6e.2xlarge instances and taints
- S3 bucket** atai-{customer-prefix}-platform-data and atai-{customer-prefix}-service-logs created
- Network connectivity** verified between pods and databases
- k8s Ingress** solution installed in the EKS cluster

Support

For questions about infrastructure requirements, contact the Archetype team (support@archetypeai.dev) for validation of instance types and scaling configurations.

atai-platform

Prerequisites

1. The kubectl command line tool is required. The version can be the same as or up to one minor version earlier or later than the Kubernetes version of your cluster.
2. The eksctl command line tool is required. For more information visit [Installation options for Eksctl](#).
3. Version 2.12.3 or later or version 1.27.160 or later of the AWS Command Line Interface (AWS CLI) installed and configured on your device.
4. An IAM principal with permissions to create and describe an Amazon EKS cluster

Download the configuration files

1. Run the commands below to get the configuration files and script required for the atai-platform configuration:

```
None
mkdir -p scripts/policies && \
curl -o scripts/6.create-irsa.sh
https://archetypeai-marketplace-assets.s3.us-west-2.amazonaws.com/scripts/6.create-irsa.sh && \
curl -o scripts/7.create-k8s-secrets.sh
https://archetypeai-marketplace-assets.s3.us-west-2.amazonaws.com/scripts/7.create-k8s-secrets.sh && \
curl -o scripts/secrets.ini.template
https://archetypeai-marketplace-assets.s3.us-west-2.amazonaws.com/scripts/secrets.ini.template && \
curl -o scripts/policies/platform-data-access.json.tpl
https://archetypeai-marketplace-assets.s3.us-west-2.amazonaws.com/scripts/policies/platform-data-access.json.tpl && \
curl -o scripts/policies/service-logs-access.json.tpl
https://archetypeai-marketplace-assets.s3.us-west-2.amazonaws.com/scripts/policies/service-logs-access.json.tpl && \
curl -o scripts/policies/model-depot-access.json
https://archetypeai-marketplace-assets.s3.us-west-2.amazonaws.com/scripts/policies/model-depot-access.json
```

2. Make the scripts executable:

```
None
chmod +x scripts/6.create-irsa.sh scripts/7.create-k8s-secrets.sh
```

Step 1: Kubernetes namespaces

1. Create the namespace to install all the components of the atai-platform

None

```
$ kubectl create namespace atai-platform
namespace/atai-platform created
```

Step 2: Kubernetes Service account for IAM roles (IRSA)

1. Associated an IAM OIDC provider

None

```
$ eksctl utils associate-iam-oidc-provider \
  --region <AWS_REGION> \
  --cluster atai-platform \
  --approve
```

```
2025-11-07 21:47:54 [i] will create IAM Open ID Connect provider for cluster
"atai-platform" in "us-west-2"
```

```
2025-11-07 21:47:55 [✓] created IAM Open ID Connect provider for cluster
"atai-platform" in "us-west-2"
```

2. Run the script 6.create-irsa.sh

None

```
$ ./6.create-irsa.sh \
  --region us-west-2 \
  --customer-name example \
  --cluster-name atai-platform \
  --platform-data-bucket "atai-example-platform-data" \
  --service-logs-bucket "atai-example-service-logs"
```

```
[INFO] Checking required commands...
```

```
[INFO] Getting AWS account ID...
```

```
[INFO] AWS Account ID: 123456789123
```

```
[INFO] Verifying cluster atai-platform exists...
```

```
[INFO] Associating IAM OIDC provider...
```

```
2025-11-09 09:51:36 [i] IAM Open ID Connect provider is already associated
with cluster "atai-platform" in "us-west-2"
```

```
[INFO] Verifying S3 buckets exist...
```

```

{
  "BucketRegion": "us-west-2",
  "AccessPointAlias": false
}
{
  "BucketRegion": "us-west-2",
  "AccessPointAlias": false
}
[INFO] Verified buckets exist: atai-example-platform-data,
atai-example-service-logs
[INFO] Creating IAM policies...
[INFO] Creating policy: atai-example-platform-data-access
[INFO] Created policy:
arn:aws:iam::123456789123:policy/atai-example-platform-data-access
[INFO] Creating policy: atai-example-service-logs-access
[INFO] Created policy:
arn:aws:iam::123456789123:policy/atai-example-service-logs-access
[INFO] Creating policy: atai-example-model-depot-access
[INFO] Created policy:
arn:aws:iam::123456789123:policy/atai-example-model-depot-access
[INFO] Creating namespace: atai-platform
Warning: resource namespaces/atai-platform is missing the
kubectl.kubernetes.io/last-applied-configuration annotation which is required
by kubectl apply. kubectl apply should only be used on resources created
declaratively by either kubectl create --save-config or kubectl apply. The
missing annotation will be patched automatically.
namespace/atai-platform configured
[INFO] Creating IAM role and service account: atai-platform-sa
2025-11-09 09:51:52 [i] 1 iamserviceaccount (atai-platform/atai-platform-sa)
was included (based on the include/exclude rules)
2025-11-09 09:51:52 [!] metadata of serviceaccounts that exist in Kubernetes
will be updated, as --override-existing-serviceaccounts was set
2025-11-09 09:51:52 [i] 1 task: {
  2 sequential sub-tasks: {
    create IAM role for serviceaccount "atai-platform/atai-platform-sa",
    create serviceaccount "atai-platform/atai-platform-sa",
  } }2025-11-09 09:51:52 [i] building iamserviceaccount stack
"eksctl-atai-platform-addon-iamserviceaccount-atai-platform-atai-platform-sa"
2025-11-09 09:51:53 [i] deploying stack
"eksctl-atai-platform-addon-iamserviceaccount-atai-platform-atai-platform-sa"
2025-11-09 09:51:53 [i] waiting for CloudFormation stack
"eksctl-atai-platform-addon-iamserviceaccount-atai-platform-atai-platform-sa"
2025-11-09 09:52:23 [i] waiting for CloudFormation stack
"eksctl-atai-platform-addon-iamserviceaccount-atai-platform-atai-platform-sa"

```

```
2025-11-09 09:52:26 [i] created serviceaccount
"atai-platform/atai-platform-sa"
[INFO] Created IAM role and service account successfully
[INFO] Setup completed successfully!
[INFO]
[INFO] Summary:
[INFO] - Created namespace: atai-platform
[INFO] - Created 3 IAM policies
[INFO] - Using S3 buckets:
[INFO] * Platform Data: atai-example-platform-data
[INFO] * Service Logs: atai-example-service-logs
[INFO] - Created IAM role: atai-example-platform-role
[INFO] - Created service account: atai-platform-sa
[INFO]
[INFO] Policy ARNs:
[INFO] Platform Data:
arn:aws:iam::123456789123:policy/atai-example-platform-data-access
[INFO] Service Logs:
arn:aws:iam::123456789123:policy/atai-example-service-logs-access
[INFO] Model Depot:
arn:aws:iam::123456789123:policy/atai-example-model-depot-access
[INFO]
[INFO] Bucket ARNs:
[INFO] Platform Data: arn:aws:s3:::atai-example-platform-data
[INFO] Service Logs: arn:aws:s3:::atai-example-service-logs
[INFO] Model Depot: arn:aws:s3:::atai-marketplace-model-depot
(base) ccastellanos@Cristians-MacBook-Pro scripts %
```

Step 3: Kubernetes secrets required for the atai-platform services

Step 3.1 Generate values for the IAM service secret

Master Key (**IAM_MASTER_KEY**)

Used to authenticate **administrative operations**, such as creating and managing organizations and keys.

Purpose:

- Solves the *bootstrapping problem* — allows you to populate an empty database.
- Can be **changed at any time**, since it's only stored in runtime.
- Gets **hashed with Argon2** before being stored in memory.

To generate:

```
None  
openssl rand -base64 32
```

After generation stores the secret in a secure place, you will need it for Step 3.2 Kubernetes secret generation.

Server Salt (**IAM_SERVER_SALT**)

Used as the **salt input for Argon2** when hashing API keys.

Key Points:

- Must remain **persistent** for the lifetime of the service.
(Changing it invalidates all existing keys.)
- Leaking it doesn't immediately compromise security **if keys are service-generated** (not manually uploaded).

To generate:

```
None  
openssl rand -base64 48
```

After generation stores the secret in a secure place, you will need it for Step 3.2 Kubernetes secret generation.

IAM db name (IAM_DB_NAME)

Postgres database that was previously created called iam_db in the section PostgreSQL database configuration Step 4: Extra Database Configuration steps. You will need it for Step 3.2 Kubernetes secret generation.

IAM db user (IAM_DB_USER)

atai_dev postgres user that was created in the PostgreSQL database configuration Step 4: Extra Database Configuration steps. You will need it for Step 3.2 Kubernetes secret generation.

IAM db password (IAM_DB_PASSWORD)

atai_dev postgres user password that was created in the PostgreSQL database configuration Step 4: Extra Database Configuration steps. You will need it for Step 3.2 Kubernetes secret generation.

IAM db port (IAM_DB_PORT)

Port of your PostgreSQL instance

IAM db password (IAM_DB_HOST)

Host of your PostgreSQL instance

Step 3.2 Kubernetes secret generation

1. Create a **secrets.ini** file using the **secrets.ini.template** file as a base. Below you will find a guide on how to replace the placeholder values in the **secrets.ini.template** file.

Placeholder	Description
<PLATFORM_ENVIRONMENT_TAG>	Deployment environment identifier (e.g., atai-sandbox, customer-name, etc,.) Example: atai-sandbox
<PLATFORM_API_ENDPOINT>	Public API endpoint URL for the platform. <ol style="list-style-type: none">1. Identify your custom domain (e.g., example.com, customer.com)2. Replace {your-domain} in the format with your actual domain: https://api.archetype.{your-domain}/v0.5
<CONSOLE_ENDPOINT>	Public endpoint URL for console/web interface <ol style="list-style-type: none">1. Identify your custom domain (e.g., example.com, customer.com)2. Replace {your-domain} in the format with your actual domain: https://console.archetype.{your-domain}
<LENS_EXTERNAL_SESSION_ENDPOINT>	External endpoint URL for lens session management <ol style="list-style-type: none">1. Identify your custom domain (e.g., example.com, customer.com)2. Replace {your-domain} in the format with your actual domain: wss://api.sandbox.{your-domain}/v0.5

Note 1: the items above will be required for following services:

- <PLATFORM_ENVIRONMENT_TAG>
 - All services
- <PLATFORM_API_ENDPOINT>
 - api-service-health-node
 - api-service-backend
 - console-2-service-frontend
 - lens-node-service-backend
- <CONSOLE_ENDPOINT>
 - api-service-health-node
 - api-service-backend
 - console-2-service-frontend
- <LENS_EXTERNAL_SESSION_ENDPOINT>
 - api-service-health-node
 - api-service-backend

Note 2: Some values are hardcoded and must be exactly as the template referenced below.

- <PLATFORM_BOT_API_KEY>
 - lens-service-backend
- <MODEL_DEPOT_URI>
 - gpq-node-newton-model-c23
 - gpq-node-newton-model-omega
- <FLAGS_SECRET>
 - console-2-service-frontend

These values will be referenced as:

```
Shell
[atai-platform-access-manager-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>

[atai-platform-api-events-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>

[atai-platform-api-service-health-node]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>
PLATFORM_API_ENDPOINT=<PLATFORM_API_ENDPOINT>
CONSOLE_ENDPOINT=<CONSOLE_ENDPOINT>
LENS_EXTERNAL_SESSION_ENDPOINT=<LENS_EXTERNAL_SESSION_ENDPOINT>
```

```
[atai-platform-api-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>
PLATFORM_API_ENDPOINT=<PLATFORM_API_ENDPOINT>
CONSOLE_ENDPOINT=<CONSOLE_ENDPOINT>
LENS_EXTERNAL_SESSION_ENDPOINT=<LENS_EXTERNAL_SESSION_ENDPOINT>

[atai-platform-console-2-service-frontend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>
PUBLIC_CONSOLE_API_URL=<PLATFORM_API_ENDPOINT>
CONSOLE_ENDPOINT=<CONSOLE_ENDPOINT>
CONSOLE_API_URL=<PLATFORM_API_ENDPOINT>
FLAGS_SECRET=abc123ABC132

[atai-platform-dfc-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>

[atai-platform-file-service-worker-node]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>

[atai-platform-file-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>

[atai-platform-gpq-node-newton-model-c23]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>
MODEL_DEPOT_URI=s3://atai-marketplace-model-depot

[atai-platform-gpq-node-newton-model-omega]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>
MODEL_DEPOT_URI=s3://atai-marketplace-model-depot

[atai-platform-gpq-service-event-router]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>

[atai-platform-gpq-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>
```

```
[atai-platform-health-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>

[atai-platform-lens-node-worker-node]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>

[atai-platform-lens-node-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>
PLATFORM_API_ENDPOINT=<PLATFORM_API_ENDPOINT>

[atai-platform-lens-service-db-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>

[atai-platform-lens-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>
PLATFORM_BOT_API_KEY=abc123ABC132

[atai-platform-registry-service-backend]
...
PLATFORM_ENVIRONMENT_TAG=<PLATFORM_ENVIRONMENT_TAG>
```

2. Run the script 7.create-k8s-secrets.sh

```
Shell
$ ./7.create-k8s-secrets.sh -n atai-platform -f secrets.ini

secret/atai-platform-access-manager-service-backend created
secret/atai-platform-api-events-service-backend created
secret/atai-platform-api-service-health-node created
secret/atai-platform-api-service-backend created
secret/atai-platform-console-2-service-frontend created
secret/atai-platform-dfc-service-backend created
secret/atai-platform-file-service-worker-node created
secret/atai-platform-file-service-backend created
secret/atai-platform-gpq-node-newton-model-c23 created
```

```
secret/atai-platform-gpq-node-newton-model-omega created
secret/atai-platform-gpq-service-event-router created
secret/atai-platform-gpq-service-backend created
secret/atai-platform-health-service-backend created
secret/atai-platform-lens-node-worker-node created
secret/atai-platform-lens-node-service-backend created
secret/atai-platform-lens-service-db-backend created
secret/atai-platform-lens-service-backend created
secret/atai-platform-registry-service-backend created
```

Helm chart installation

Prerequisites

1. The kubectl command line tool is required. The version can be the same as or up to one minor version earlier or later than the Kubernetes version of your cluster.
2. Version 2.12.3 or later or version 1.27.160 or later of the AWS Command Line Interface (AWS CLI) installed and configured on your device.
3. Helm 3.19.0. Learn more about [Helm installation here](#).

Step 1: Installation

1. Retrieve an authentication token and authenticate your clients. Enter the AWS CLI:

None

```
aws ecr get-login-password \  
  --region us-east-1 | helm registry login \  
  --username AWS \  
  --password-stdin 709825985650.dkr.ecr.us-east-1.amazonaws.com
```

2. Create a tmp folder:

None

```
mkdir awsmc-chart && cd awsmc-chart
```

3. Pull the atai-platform helm chart:

None

```
$ helm pull \  
oci://709825985650.dkr.ecr.us-east-1.amazonaws.com/archetype-ai/atai_core/helm/  
atai-platform --version 1.0.0-99
```

Pulled:

```
709825985650.dkr.ecr.us-east-1.amazonaws.com/archetype-ai/atai_core/helm/atai-p  
latform:0.1.0  
Digest: <sha256:...>  
709825985650.dkr.ecr.us-east-1.amazonaws.com/archetype-ai/atai_core/helm/atai-p  
latform:0.1.0 contains an underscore.  
OCI artifact references (e.g. tags) do not support the plus sign (+). To  
support
```

storing semantic versions, Helm adopts the convention of changing plus (+) to an underscore (_) in chart version tags when pushing to a registry and back to a plus (+) when pulling from a registry.

4. Create a **values.yaml** file using the following base structure:

```
Go
api-service-backend:
  ingress:
    enabled: true
    className: nginx
    annotations:
      nginx.ingress.kubernetes.io/ssl-redirect: "true"
      nginx.ingress.kubernetes.io/backend-protocol: "HTTP"
      nginx.ingress.kubernetes.io/force-ssl-redirect: "true"
    host: <PLATFORM_API_ENDPOINT>
    path: /
    pathType: Prefix
    tls:
      enabled: true
      clusterIssuer: letsencrypt-stage
console-2-service-frontend:
  ingress:
    enabled: true
    className: nginx
    annotations:
      nginx.ingress.kubernetes.io/ssl-redirect: "true"
      nginx.ingress.kubernetes.io/backend-protocol: "HTTP"
      nginx.ingress.kubernetes.io/force-ssl-redirect: "true"
    host: <CONSOLE_ENDPOINT>
    path: /
    pathType: Prefix
    tls:
      enabled: true
      clusterIssuer: letsencrypt-stage
```

5. Install the atai-platform helm chart:

Shell

```
$ helm upgrade atai-platform atai-platform-1.0.0-99.tgz \  
--namespace atai-platform \  
--values values.yaml \  
--install
```

```
Release "atai-platform" does not exist. Installing it now.  
I1111 11:06:28.291728 35840 warnings.go:110] "Warning:  
spec.template.spec.containers[0].env[9]: hides previous definition of  
\"REDIS_USE_INSECURE_TLS\", which may be dropped when using apply"  
NAME: atai-platform  
LAST DEPLOYED: Tue Nov 11 11:06:23 2025  
NAMESPACE: atai-platform  
STATUS: deployed  
REVISION: 1  
TEST SUITE: None
```

Getting Started with the Archetype Platform

To get started with the Archetype Platform, please visit following documentation page:
<https://docs.archetypeai.app/overview/introduction>

Appendix

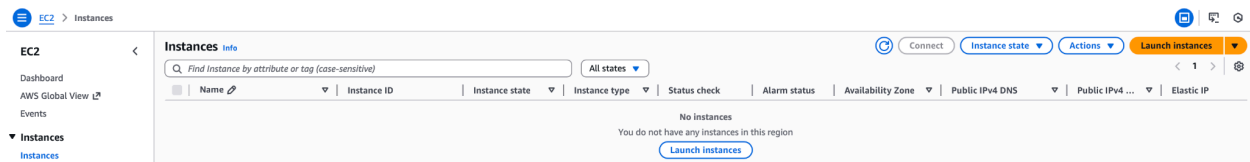
Bastion host configuration

Prerequisites

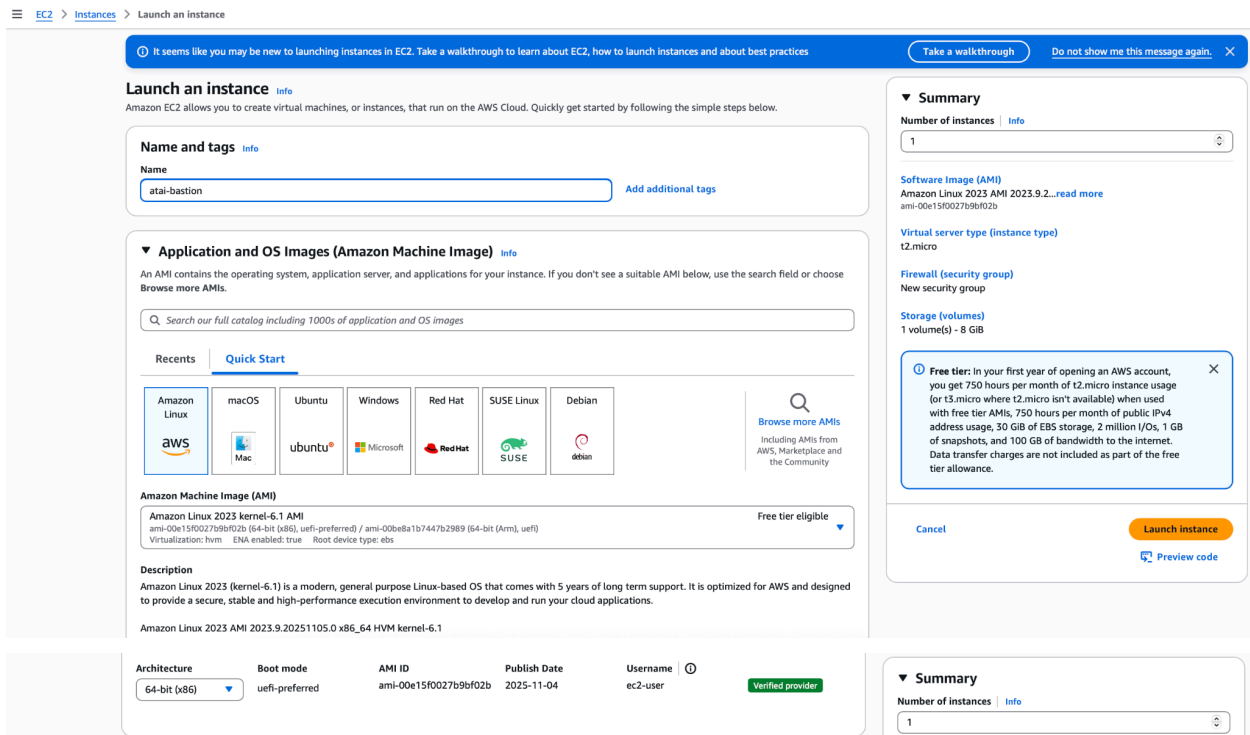
1. VPC with public subnet
2. Internet Gateway attached to VPC
3. Route table configured for public subnet

Step 1: Launch EC2 Instance (Bastion Host)

1. Go to EC2 → Instances → Launch instance



2. Name: atai-bastion
3. Application and OS Images (Amazon Machine Image):
 - a. Search for: Amazon Linux 2023 AMI
 - b. Select: Amazon Linux 2023 AMI (x86_64, HVM, kernel 6.1)



4. Instance type: Select your instance type (e.g., t3.medium for dev, t3.large for production)

Launch an instance

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t3.large
Family: t3 2 vCPU 8 GiB Memory Current generation: true On-Demand RHEL base pricing: 0.112 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0867 USD per Hour On-Demand Linux base pricing: 0.0832 USD per Hour
On-Demand SUSE base pricing: 0.1395 USD per Hour On-Demand Windows base pricing: 0.1108 USD per Hour

All generations [Compare instance types](#)

[Additional costs apply for AMIs with pre-installed software](#)

5. Select an existing key pair or click on **Create a new key pair**

Launch an instance

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

Select

6. Assign a new atai-bastion-key, then click on **Create key pair**

Create key pair ✕

Key pair name

Key pairs allow you to connect to your instance securely.

atai-bastion-key

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type



RSA

RSA encrypted private and public key pair



ED25519

ED25519 encrypted private and public key pair

Private key file format



.pem

For use with OpenSSH



.ppk

For use with PuTTY



When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#)

[Cancel](#)

[Create key pair](#)

7. Network settings:

- a. VPC: Select your VPC
- b. Subnet: Select a public subnet (e.g., atai-platform-vpc-public-us-west-2a)
- c. Auto-assign public IP: Enable (or use Elastic IP from Step 4)

Launch an instance

Network settings [Info](#)

VPC - *required* [Info](#)

vpc-0a79faee3e664a31d (atai-platform-vpc)
10.5.0.0/16

Subnet [Info](#)

subnet-04434b04f64fc276b atai-platform-vpc-public-us-west-2a
VPC: vpc-0a79faee3e664a31d Owner: 716124474177 Availability Zone: us-west-2a (usw2-az2)
Zone type: Availability Zone IP addresses available: 250 CIDR: 10.5.80.0/24

Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of free tier allowance

- d. Firewall (security groups): Select **Create security group**
 - i. By default AWS will add an SSH rule

Launch an instance

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

Security group name - *required*

launch-wizard-1

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and .-:/!@#%&*~`|_|'+=&:[]\$*

Description - *required* [Info](#)

launch-wizard-1 created 2025-11-07T19:48:21.016Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0) Remove

Type Info	Protocol Info	Port range Info
ssh	TCP	22
Source type Info	Source Info	Description - <i>optional</i> Info
Anywhere	<input type="text" value="0.0.0.0/0"/> <input type="text" value="0.0.0.0/0"/>	e.g. SSH for admin desktop

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

[Add security group rule](#)

► **Advanced network configuration**

8. Configure storage

- a. Default (8 GiB gp3)
- b. Recommended 25 GB, but you can adjust based on your requirements

▼ Configure storage [Info](#) Advanced

1x GIB Root volume, 3000 IOPS, Not encrypted

[Free tier eligible customers can get up to 30 GB of EBS General Purpose \(SSD\) or Magnetic storage](#) ✕

[Add new volume](#)

[Click refresh to view backup information](#) ↻

The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

0 x File systems Edit

9. Advanced details (expand):

- a. Metadata accessible: Enable
- b. Metadata version: V2 only (token required) (IMDSv2)
- c. Metadata token response hop limit: 2

▼ Advanced details [Info](#)

Domain join directory | [Info](#)

[Create new directory](#)

IAM instance profile | [Info](#)

[Create new IAM profile](#)

Metadata accessible | [Info](#)

Metadata IPv6 endpoint | [Info](#)

Metadata version | [Info](#)

⚠ For V2 requests, you must include a session token in all instance metadata requests. Applications or agents that use V1 for instance metadata access will break.

Metadata response hop limit | [Info](#)

Allow tags in metadata | [Info](#)

d. User data: Paste the following script:

```
None
#!/bin/bash

# Update system packages
sudo dnf update -y

# Install essential packages
sudo dnf install -y \
    htop \
    vim \
    git \
    wget \
    unzip \
    jq \
    postgresql17 \
    redis6

# Install AWS CLI v2
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
rm -rf aws awscliv2.zip
```

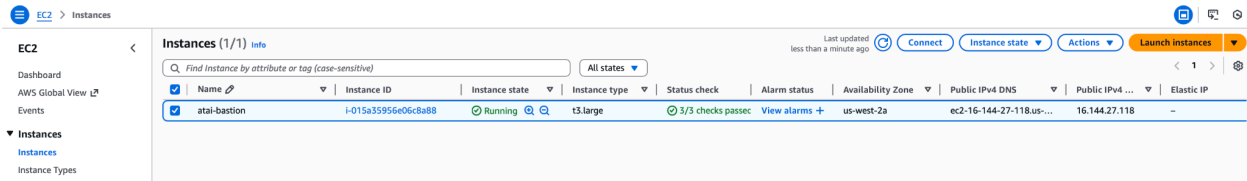
The screenshot shows the 'Launch instance' dialog in the AWS console. The 'User data' field is populated with the script provided in the previous block. A 'Free tier' notification is visible on the right side of the dialog, stating: 'Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included as part of the free tier allowance.' The 'Launch instance' button is highlighted in orange.

10. Click on **Launch instance**.

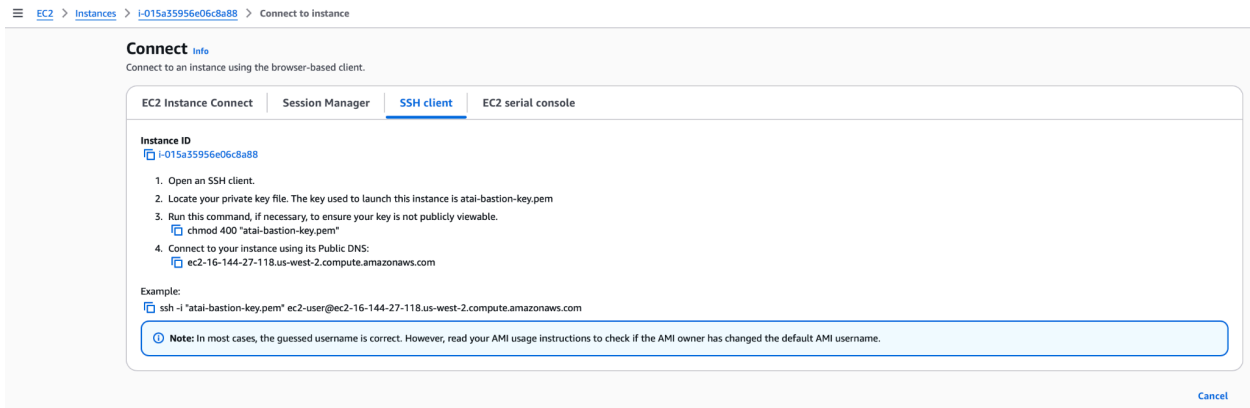
The screenshot shows the AWS console 'Launch instance' page. The breadcrumb navigation is 'EC2 > Instances > Launch an instance'. A green success message is displayed: 'Success Successfully initiated launch of instance (i-015a3595e06c8a88)'. Below the message is a 'Launch log' link.

Step 2: Connect to your Bastion host

1. Go to EC2 → Instances → Select your atai-bastion instance and click on **Connect**



2. Click on the tab **SSH client**



- a. Open an SSH client.
- b. Locate your private key file. The key used to launch this instance is `atai-bastion-key.pem`
- c. Run this command, if necessary, to ensure your key is not publicly viewable.

None

```
chmod 400 "atai-bastion-key.pem"
```

- d. Connect to your instance using its Public DNS e.g.:

None

```
ec2-16-144-27-118.us-west-2.compute.amazonaws.com
```

Example

None

```
ssh -i "atai-bastion-key.pem"  
ec2-user@ec2-16-144-27-118.us-west-2.compute.amazonaws.com
```

AWS Service Quotas

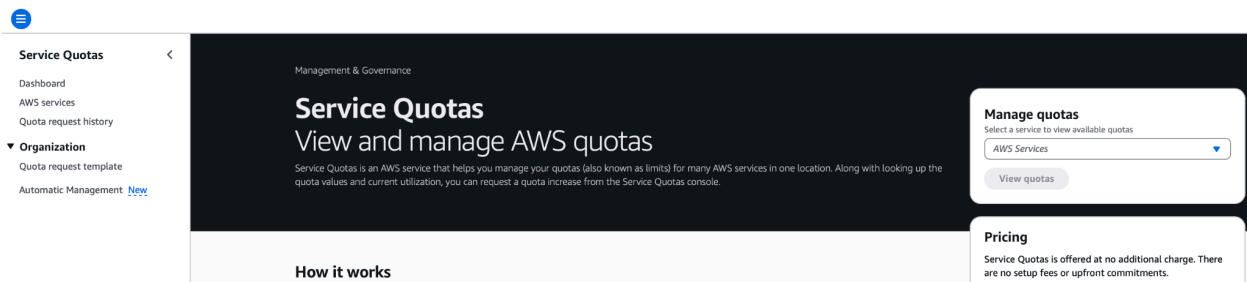
Running On-Demand G and VT instances

Requirements:

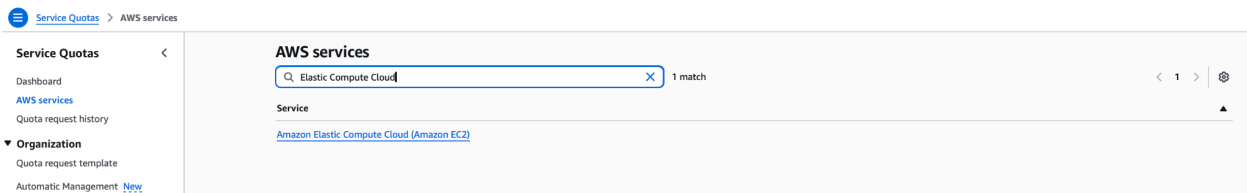
- Instance Type: g6e.2xlarge
- vCPUs per instance: 8 vCPUs
- Minimum instances required: 10
- Total vCPUs needed: $10 \times 8 = 80$ vCPUs

Service Quota to check:

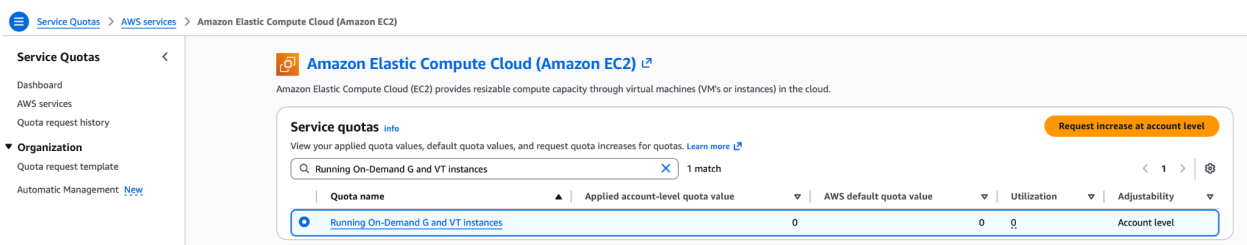
3. Go to AWS Service Quotas → In the left panel click on **AWS Services**



4. In the search bar type AWS Elastic Compute Cloud



5. In the search bar type **Running On-Demand G and VT instances**, select the quota and click on **Request increase at account level**.



6. Set the **Increase account value** to 80 vCPUs and click on **Request**

Request quota increase: Running On-Demand G and VT instances ✕

Description

Maximum number of vCPUs assigned to the Running On-Demand G and VT instances.

Requested for

Account (716124474177)

Region

United States (Oregon) us-west-2

Increase quota value

Enter in the total amount that you want the quota to be.

Must be a number greater than your current quota value of 0

Utilization

0

Approvals: For some services, smaller increases are automatically approved, while larger requests are submitted to AWS Support.

Approval timeline: AWS Support can approve, deny, or partially approve your requests. Larger increase requests take more time to process and assess while we work with the service team.

[Cancel](#)

[View quota details](#)

[Request](#)

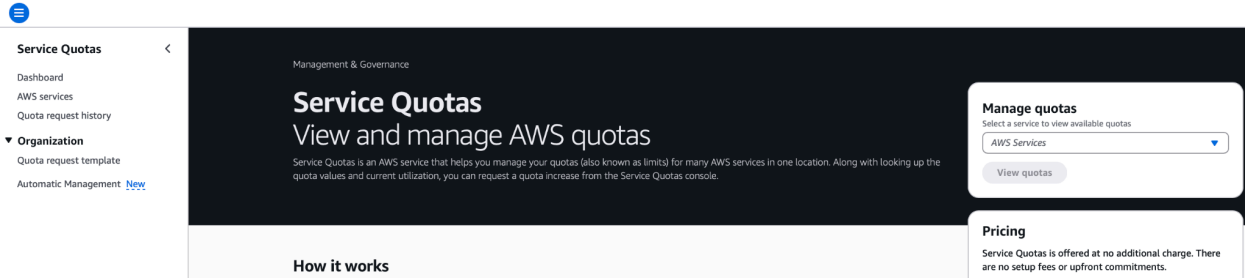
Running On-Demand Standard (A, C, D, H, I, M, R, T, Z) instances

Requirements:

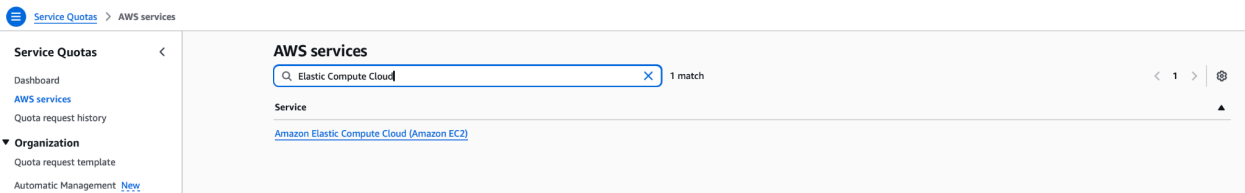
- Instance Type: m6i.4xlarge
- vCPUs per instance: 16 vCPUs
- Minimum instances required: 6
- Total vCPUs needed: $6 \times 16 = 96$ vCPUs

Service Quota to check:

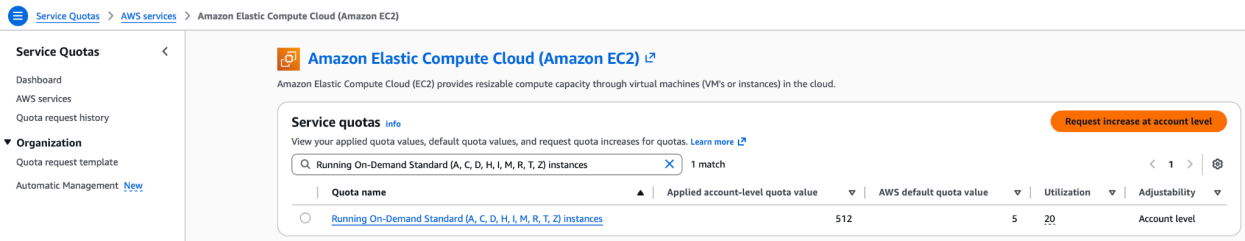
1. Go to AWS Service Quotas → In the left panel click on **AWS Services**



2. In the search bar type AWS Elastic Compute Cloud



3. In the search bar type **Running On-Demand Standard (A, C, D, H, I, M, R, T, Z) instances**, select the quota and click on **Request increase at account level**.



4. Set the **Increase account value** to 96 vCPUs and click on **Request**

Note: If your existing quota for Standard Instances (A, C, D, H, I, M, R, T, Z family) already exceeds 96 vCPUs, no increase is needed for this family.

AWS License Manager Setup

All atai-platform services are integrated with AWS License Manager and require it to be enabled and configured. Without proper License Manager setup, the product will not function. If you see the error bellow in the logs of the atai_core pods:

None

```
Error could not call LM checkout api **An error occurred  
(AccessDeniedException) when calling the CheckoutLicense operation: Service  
role not found. Consult setup procedures in License Manager Us
```

Follow the steps below to configure it and prevent service outages. To use AWS License Manager, you must first complete onboarding steps. The following procedure walks you through the onboarding steps in the AWS Management Console.

Get started with License Manager

1. Open the License Manager console at <https://console.aws.amazon.com/license-manager/>.

The screenshot shows the AWS License Manager console interface. At the top, there's a search bar and navigation icons. The main heading is "AWS License Manager" with the subtext "Manage, discover, and report software license usage". Below this, a brief description states: "AWS License Manager offers multiple ways to track license usage across your environments. Get started with user-based licenses, granted licenses, self managed licenses, or seller issued licenses."

On the right side, there's a "Get started" section with a button labeled "Start using AWS License Manager". Below that is a "Pricing" section with a note: "There is no additional charge for AWS License Manager." and links to various pricing pages: "Amazon pricing", "Amazon EC2 pricing", "Amazon EBS pricing", "Amazon Systems Manager pricing", and "Amazon SNS pricing".

The "How it works" section features a diagram illustrating the workflow for self-managed licenses:

- Define rules for your licensed software**: Represented by a laptop icon with code symbols.
- Attach licensing rules during launch and proactively control usage**: Represented by a cloud icon with code symbols.
- Search inventory and track licenses brought in post-launch**: Represented by a cloud icon with a magnifying glass.
- Use alerts to control and centrally manage licenses across all AWS accounts and on-premises**: Represented by a monitor icon with charts.

The diagram is titled "Self managed licenses" and is enclosed in a box with the AWS License Manager logo.

- You are prompted to configure permissions for License Manager and its supporting services. Follow the directions to configure the required permissions.

IAM permissions (one-time setup)



AWS License Manager requires permissions to manage licenses used by resources.

I grant AWS License Manager the required permissions

[View details](#)

Cancel

Grant permissions

- With the initial setup complete, you can proceed with your installation.

The screenshot shows the AWS License Manager dashboard. The left sidebar contains navigation links: Dashboard, Usage reports, Inventory search, Self-managed licenses, Host resource groups, License type conversion, Granted licenses, and Seller issued licenses. The main content area is titled 'Overview' and displays three metrics: 'Granted licenses' with a value of 2, 'Self-managed licenses' with a value of 0, and 'Seller issued licenses' with a value of 0. Below this is a section for 'Granted license entitlements (4)', which includes a table of license entitlements.

Product name	Entitlement	Usage
Archetype AI	ArchetypePlatform	Enabled
Archetype AI	AWS:Marketplace:Usage	Enabled

EKS configuration - Install the AWS Load Balancer controller

Prerequisites

Before starting this tutorial, you must complete the following steps:

1. Create an Amazon EKS cluster. To create one, see [Get started with Amazon EKS](#)
2. Install [Helm](#) on your local machine. Helm installed - Version 3.x or later
3. Make sure that your Amazon VPC CNI plugin for Kubernetes, kube-proxy, and CoreDNS add-ons are at the minimum versions listed in [Service account tokens](#)
4. The eksctl command line tool is required. For more information visit [Installation options for Eksctl](#).
5. Version 2.12.3 or later or version 1.27.160 or later of the AWS Command Line Interface (AWS CLI) installed and configured on your device.
6. Learn about AWS Elastic Load Balancing concepts. For more information, see the [Elastic Load Balancing User Guide](#).
7. Learn about Kubernetes [service](#) and [ingress](#) resources

Step 1: Create IAM Role using eksctl

1. Create IAM OIDC provider


None

```
eksctl utils associate-iam-oidc-provider \
  --region <region-code> \
  --cluster <your-cluster-name> \
  --approve
```

2. Download an IAM policy for the AWS Load Balancer Controller that allows it to make calls to AWS APIs on your behalf.

None

```
curl -O
https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/
v2.14.1/docs/install/iam_policy.json
```

 If you are a non-standard AWS partition, such as a Government or China region, review the policies on GitHub and download the appropriate policy for your region.

1. Create an IAM policy using the policy downloaded in the previous step.

None

```
aws iam create-policy \  
  --policy-name AWSLoadBalancerControllerIAMPolicy \  
  --policy-document file://iam_policy.json
```

2. Replace the values for cluster name, region code, and account ID

None

```
eksctl create iamserviceaccount \  
  --cluster=<cluster-name> \  
  --namespace=kube-system \  
  --name=aws-load-balancer-controller \  
  
  --attach-policy-arn=arn:aws:iam::<AWS_ACCOUNT_ID>:policy/AWSLoadBalancerControl  
lerIAMPolicy \  
  --override-existing-serviceaccounts \  
  --region <aws-region-code> \  
  --approve
```

Step 2: Install AWS Load Balancer Controller

1. Add the eks-charts Helm chart repository. AWS maintains this repository on GitHub.

None

```
helm repo add eks-charts https://aws.github.io/eks-charts
```

2. Update your local repo to make sure that you have the most recent charts.

None

```
helm repo update eks-charts
```

3. Install the AWS Load Balancer Controller.

Add the following flags to the helm command that follows:

- a. `--set region=region-code`
- b. `--set vpcId=vpc-xxxxxxx`

Replace my-cluster with the name of your cluster. In the following command, aws-load-balancer-controller is the Kubernetes service account that you created in a previous step. For more information about configuring the helm chart, see values.yaml on GitHub.

None

```
helm install aws-load-balancer-controller  
eks-charts/aws-load-balancer-controller \  
  -n kube-system \  
  --set clusterName=atai-platform \  
  --set region=region-code \  
  --set vpcId=vpc-xxxxxxx \  
  --set serviceAccount.create=false \  
  --set serviceAccount.name=aws-load-balancer-controller \  
  --version 1.14.0
```

4. The helm install command automatically installs the custom resource definitions (CRDs) for the controller. The helm upgrade command does not. If you use helm upgrade, you must manually install the CRDs. Run the following command to install the CRDs:

None

```
wget  
https://raw.githubusercontent.com/aws/eks-charts/master/stable/aws-load-balancer-controller/crds/crds.yaml  
kubectl apply -f crds.yaml
```

Step 3: Verify that the controller is installed

1. Verify that the controller is installed.

None

```
kubectl get deployment -n kube-system aws-load-balancer-controller
```

An example output is as follows.

None

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
aws-load-balancer-controller	2/2	2	2	76s

EKS configuration - Install the NGINX ingress controller

Why We Use Both AWS Load Balancer Controller and NGINX Ingress Controller

Our platform uses both components because they serve different roles and work together.

1. **AWS Load Balancer Controller** - Handles AWS infrastructure provisioning
 - a. Automatically creates AWS Network Load Balancers (NLB) or Application Load Balancers (ALB) when Kubernetes Services of type LoadBalancer are created
 - b. Manages AWS-specific configurations (security groups, subnets, tags, target groups)
 - c. Provides native AWS integration and control
2. **NGINX Ingress Controller** - Handles advanced traffic routing and request processing
 - a. Provides capabilities beyond what AWS Load Balancer Controller offers for routing

In Summary

1. **AWS Load Balancer Controller** = Creates and manages the AWS infrastructure (the "front door")
2. **NGINX Ingress Controller** = Provides the routing intelligence (the "traffic director")
3. Together = Automated AWS infrastructure + powerful routing capabilities

This combination is a common pattern for production Kubernetes workloads on AWS, providing both infrastructure automation and flexible request handling.

Prerequisites

Before installing NGINX Ingress Controller, ensure you have:

1. EKS Cluster - Your Kubernetes cluster must be running and accessible
2. kubectl configured - You must be able to connect to your cluster (kubectl get nodes should work). The version can be the same as or up to one minor version earlier or later than the Kubernetes version of your cluster.
3. Install [Helm](#) on your local machine. Helm installed - Version 3.x or later
4. Public Subnets - You need the subnet IDs where the load balancer will be created
5. VPC ID - The VPC ID where your cluster is running

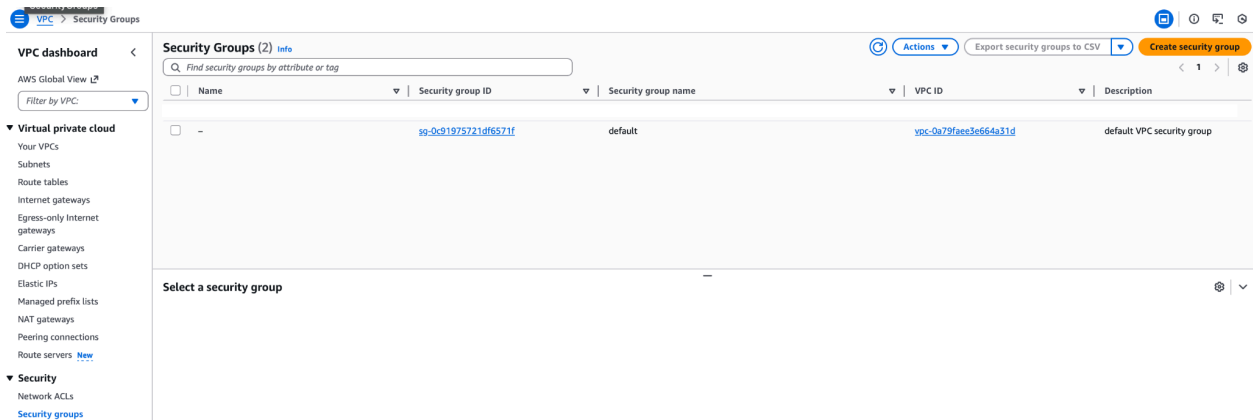
Step 1: Create Security Group for NGINX Ingress Controller

The NGINX Ingress Controller needs a security group that allows:

- Inbound traffic on port 80 (HTTP)
- Inbound traffic on port 443 (HTTPS)
- Outbound traffic to anywhere

To create a new security group from the AWS console use the following steps:

1. Go to VPC → Security Groups → Create security group



12. Name: atai-nginx-ingress-controller-sg (or your name)
13. Description: Security group for the Load Balancer created by the NGINX Ingress controller
14. VPC: Select your VPC from section VPC configuration Step 1
15. Inbound rules: Add rule:
 - a. HTTP
 - i. Type: HTTP
 - ii. Port: 80
 - iii. Source: Custom → Enter 0.0.0.0/0
 - b. HTTPS
 - i. Type: HTTPS
 - ii. Port: 443
 - iii. Source: Custom → Enter 0.0.0.0/0

Create security group info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name info

atal-nginx-ingress-controller-sg

Name cannot be edited after creation.

Description info

Security group for the Load Balancer created by the NGINX Ingress

VPC info

vpc-0a79fae3e664a31d (atal-platform-vpc)

Inbound rules info

Type <small>info</small>	Protocol <small>info</small>	Port range <small>info</small>	Source <small>info</small>	Description - optional <small>info</small>	
HTTP	TCP	80	Anywhere... 0.0.0.0/0		Delete
HTTPS	TCP	443	Anywhere... 0.0.0.0/0		Delete

Add rule

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Outbound rules info

Type <small>info</small>	Protocol <small>info</small>	Port range <small>info</small>	Destination <small>info</small>	Description - optional <small>info</small>	
All traffic	All	All	Custom 0.0.0.0/0		Delete

Add rule

Rules with destination of 0.0.0.0/0 or ::/0 allow your instances to send traffic to any IPv4 or IPv6 address. We recommend setting security group rules to be more restrictive and to only allow traffic to specific known IP addresses.

Tags - optional

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.

No tags associated with the resource.

Add new tag
You can add up to 50 more tags

Cancel Create security group

Step 2: Add helm repository

1. Add the NGINX Ingress Controller Helm repository:

None

```
$ helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
$ helm repo update
```

```
"ingress-nginx" has been added to your repositories
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "ingress-nginx" chart repository
Update Complete. ✨Happy Helming!✨
```

2. Verify the repository was added:

None

```
$ helm repo list
NAME          URL
ingress-nginx https://kubernetes.github.io/ingress-nginx
```

Step 3: Prepare configuration values

1. Create a file named `nginx-ingress-values.yaml` with the following content:

```
None
cat << EOF > nginx-ingress-values.yaml
controller:
  # Use Deployment for standard setup, or DaemonSet for high availability
  kind: Deployment

  service:
    # Use LoadBalancer type to create an AWS load balancer
    type: LoadBalancer
    # Use AWS Load Balancer Controller to manage the load balancer
    loadBalancerClass: "service.k8s.aws/nlb"
    # External traffic policy: "Cluster" for standard, "Local" for HA
    # (preserves source IP)
    externalTrafficPolicy: Cluster
    annotations:
      # Create a Network Load Balancer
      service.beta.kubernetes.io/aws-load-balancer-type: "nlb"
      # Make it internet-facing
      service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
      # Use the security group you created in Step 1
      service.beta.kubernetes.io/aws-load-balancer-security-groups:
"sg-xxxxxxxx"
      # Allow AWS Load Balancer Controller to manage security group rules

service.beta.kubernetes.io/aws-load-balancer-manage-backend-security-group-rules: "true"
  # Specify the public subnets where the load balancer will be created
  # Replace with your actual public subnet IDs, comma-separated
  service.beta.kubernetes.io/aws-load-balancer-subnets:
"subnet-xxxxxxxx,subnet-yyyyyyyyy"
  # Optional: Name for the load balancer target group
  service.beta.kubernetes.io/aws-load-balancer-group-name:
"nginx-external-group"
  # Allow large file uploads (up to 500MB, default is 1MB)
  config:
    proxy-body-size: "500m"
EOF
```

 Important: Replace the following values in the file:

- `sg-xxxxxxxx` - Your security group ID from Step 1
- `subnet-xxxxxxxx,subnet-yyyyyyyyy` - Your public subnet IDs of your public subnets such as `atai-platform-vpc-public-us-west-2a` and `atai-platform-vpc-public-us-west-2b` (comma-separated, no spaces)

Step 2.1 High Availability Configuration (Optional)

If you want high availability with source IP preservation, change these values in the YAML file:

```
None
controller:
  kind: DaemonSet # Changed from Deployment
  service:
    externalTrafficPolicy: Local # Changed from Cluster
```

Step 4: Install NGINX Ingress Controller

1. Install using helm

```
None
helm install ingress-nginx-controller-external ingress-nginx/ingress-nginx \
  --namespace ingress-nginx-controller \
  --create-namespace \
  --version 4.13.2 \
  --values nginx-ingress-values.yaml
```

Step 5: Verify installation

Step 5.1. Check Pod Status

1. Wait a few moments, then check if the pods are running:

```
None
$ kubectl get pods -n ingress-nginx-controller
```

2. You should see the NGINX Ingress Controller pod(s) in Running status:

```
None
NAME                                READY   STATUS    RESTARTS   AGE
ingress-nginx-controller-external-controller-xxxxxxx  1/1     Running   0          2m
```

Step 5.2 Check Service Status

1. Check the LoadBalancer service:


None

```
$ kubectl get svc -n ingress-nginx-controller
```

You should see a service of type LoadBalancer:

None

```
NAME                                                    TYPE
CLUSTER-IP      EXTERNAL-IP
PORT(S)          AGE
ingress-nginx-controller-external-controller          LoadBalancer
172.20.133.73   a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6.elb.<region>.amazonaws.com
80:30495/TCP,443:31274/TCP   57m
```

 Note the EXTERNAL-IP - This is the DNS name of your AWS Network Load Balancer. You'll need this for DNS configuration.

DNS configuration

After your Ingress Controller is configured, it typically creates an AWS Network Load Balancer (NLB) or Application Load Balancer (ALB) as the entry point for your services.

DNS Configuration Steps:

1. Your Ingress Controller creates an AWS load balancer (NLB or ALB)
 - a. The load balancer receives a public DNS name or IP address

If you installed the NGINX ingress controller following the instruction from Appendix [EKS configuration - Install the NGINX ingress controller](#), you can get the Network Load Balancer associated with your NGINX ingress controller with the command below:

```
None
$ LB_DNS=$(kubectl get svc ingress-nginx-controller-external-controller -n
ingress-nginx-controller -o
jsonpath='{.status.loadBalancer.ingress[0].hostname}')
$ echo "Load Balancer DNS: $LB_DNS"

Load Balancer DNS: a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6.elb.<region>.amazonaws.com
```

2. You need to add DNS records that point your domain names to this load balancer

Required DNS Records:

1. Add the following DNS records (A records or CNAME records) pointing to the load balancer created by your Ingress solution:
 - a. **console-archetype.<your-domain>** → Points to the load balancer
 - b. **api-archetype.<your-domain>** → Points to the load balancer

EKS configuration - Configure Cert-Manager and Let's Encrypt

Prerequisites

1. EKS Cluster - Your Kubernetes cluster must be running and accessible
2. kubectl configured - You must be able to connect to your cluster (kubectl get nodes should work). The version can be the same as or up to one minor version earlier or later than the Kubernetes version of your cluster.
3. Install [Helm](#) on your local machine. Helm installed - Version 3.x or later

Step 1: Add helm repository

1. Add the Cert-manager Helm repository:

None

```
helm repo add jetstack https://charts.jetstack.io
helm repo update
```

2. Verify the repository was added:

None

```
helm repo list
```

Step 2: Install Cert-manager

1. Install Cert-Manager using Helm. The chart will automatically install the required CRDs:

None

```
helm install cert-manager jetstack/cert-manager \
  --namespace cert-manager \
  --create-namespace \
  --version 1.18.2 \
  --set installCRDs=true \
  --set serviceAccount.create=true
```

Step 3: Verify installation

Step 3.1 Check Pod Status

1. Wait a few moments (30-60 seconds), then check if the pods are running:

```
None
kubectl get pods -n cert-manager
```

You'll see three pods in running status

```
None
NAME                                READY   STATUS    RESTARTS   AGE
cert-manager-xxxxxxxx-xxxxx         1/1     Running   0           2m
cert-manager-cainjector-xxxxxxxx-xxxxx 1/1     Running   0           2m
cert-manager-webhook-xxxxxxxx-xxxxx   1/1     Running   0           2m
```

Step 3.2. Check Deployment Status

1. Verify the deployments are ready:

```
None
kubectl get deployments -n cert-manager
```

All deployments should show READY 1/1:

```
None
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
cert-manager                        1/1     1             1           6m33s
cert-manager-cainjector             1/1     1             1           6m33s
cert-manager-webhook                1/1     1             1           6m33s
```

Step 3.3 Verify Cert-Manager is Working

1. Test that Cert-Manager can create resources:

```
None
# Check if ClusterIssuer CRD is available
kubectl api-resources | grep cert-manager
# You should see resources like:
# certificates, certificaterequests, challenges, clusterissuers, issuers,...
```

Step 4: Create Cluster Issuer

Understanding Issuers and ClusterIssuers

Cert-Manager supports two types of certificate issuers:

- **Issuer** - Scoped to a specific namespace. Can only issue certificates for resources in that namespace.
- **ClusterIssuer** - Cluster-wide. Can issue certificates for any namespace in the cluster.

For most use cases, use a ClusterIssuer so you can issue certificates across all namespaces without creating multiple issuers.

Understanding Certificate Challenges

To obtain SSL certificates from Let's Encrypt, Cert-Manager must prove domain ownership.

There are two challenge types:

- **HTTP-01 Challenge:**
 - Let's Encrypt makes an HTTP request to your domain
 - Cert-Manager creates a temporary file that Let's Encrypt can access
 - Requires your domain to be publicly accessible and point to your load balancer
 - Simpler to set up
 - Requires DNS to be configured first (domain must point to your load balancer). You must complete the [DNS configuration](#) before creating the ClusterIssuer.
- **DNS-01 Challenge:**
 - Let's Encrypt verifies domain ownership via DNS records
 - Cert-Manager creates a TXT record in your DNS provider
 - Works even if your domain isn't fully accessible yet
 - More complex (requires API access to your DNS provider)
 - Better for wildcard certificates

For simplicity, we'll use HTTP-01 challenge, which requires:

1. Your domain names (console.archetype.<your-domain> and api.archetype.<your-domain>) must have DNS records pointing to your NGINX Ingress Controller's load balancer. For more information about the DNS configuration visit the Appendix [DNS configuration](#) before creating the ClusterIssuer.
2. The load balancer must be publicly accessible
3. Port 80 (HTTP) must be open for Let's Encrypt validation

Let's Encrypt Servers

Let's Encrypt provides two ACME servers:

- **Staging Server** (<https://acme-staging-v02.api.letsencrypt.org/directory>)
 - For testing
 - Higher rate limits (good for testing)
 - Certificates are not trusted by browsers (you'll see a warning)
 - Use this first to test your configuration

- **Production Server** (<https://acme-v02.api.letsencrypt.org/directory>)
 - For production use
 - Lower rate limits (50 certificates per registered domain per week)
 - Certificates are trusted by browsers
 - Use this after testing with staging
 - Recommendation: Start with the staging server to test, then switch to production once everything works.

Step 4.1. DNS configuration

You must add the following DNS records (A records or CNAME records) pointing to the load balancer created by your Ingress solution:

- c. **console-archetype.<your-domain>** → Points to the load balancer
- d. **api-archetype.<your-domain>** → Points to the load balance

For more information visit the Appendix [DNS configuration](#) before creating the ClusterIssuer.

Step 4.2 Create the Cluster Issuer

1. Create a ClusterIssuer YAML file. For HTTP-01 challenge, create cluster-issuer-http.yaml:

```
None
cat << EOF > cluster-issuer-http.yaml
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-stage
  namespace: cert-manager
spec:
  acme:
    # Let's Encrypt staging server (use for testing)
    server: https://acme-staging-v02.api.letsencrypt.org/directory
    # Your email address (Let's Encrypt will send expiration notices here)
    email: your-email@example.com
    # Private key secret name (Cert-Manager will create this)
    privateKeySecretRef:
      name: letsencrypt-stage
    # HTTP-01 challenge solver
    solvers:
    - http01:
        ingress:
          class: nginx
EOF
```

⚠ Important: Replace the following values in the file:

- your-email@example.com - Your DNS administrator email address

2. Apply the cluster issuer to the cluster

None

```
kubectl apply -f cluster-issuer-http.yaml
```

Step 4.3 Verify the cluster issuer

1. Check that the ClusterIssuer was created:

None

```
kubectl get clusterissuer -n cert-manager
```

You should see:

None

NAME	READY	AGE
letsencrypt-stage	True	4m47s

2. Check details status

None

```
kubectl describe clusterissuer letsencrypt-stage -n cert-manager
```