NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

# Coaching - ECS

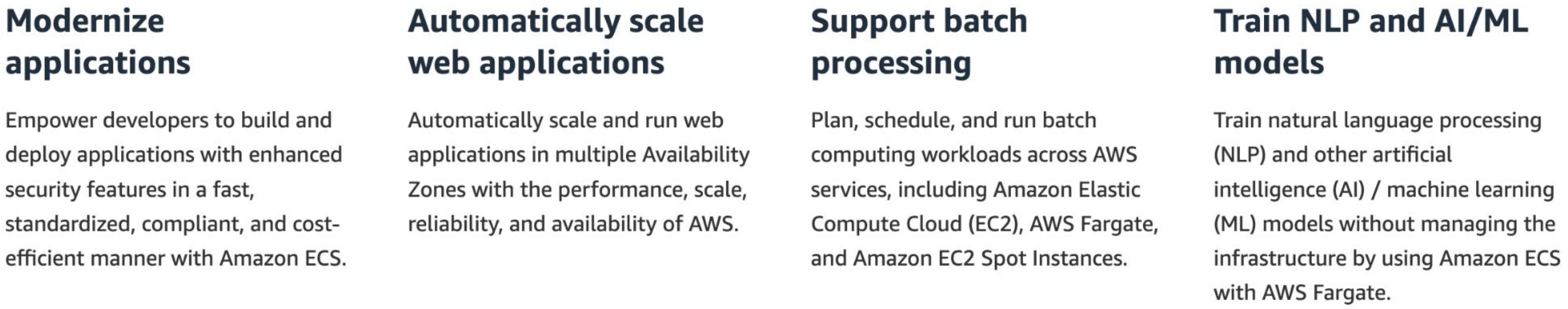
Cloud Infrastructure Engineering

**Nanyang Technological University & Skills Union - 2022/2023**

# ECS Overview

Amazon Elastic Container Service (Amazon ECS) is a fully managed container orchestration service that simplifies your deployment, management, and scaling of containerized applications.

# Use Cases



### Modernize applications

Empower developers to build and deploy applications with enhanced security features in a fast, standardized, compliant, and cost-efficient manner with Amazon ECS.

### Automatically scale web applications

Automatically scale and run web applications in multiple Availability Zones with the performance, scale, reliability, and availability of AWS.

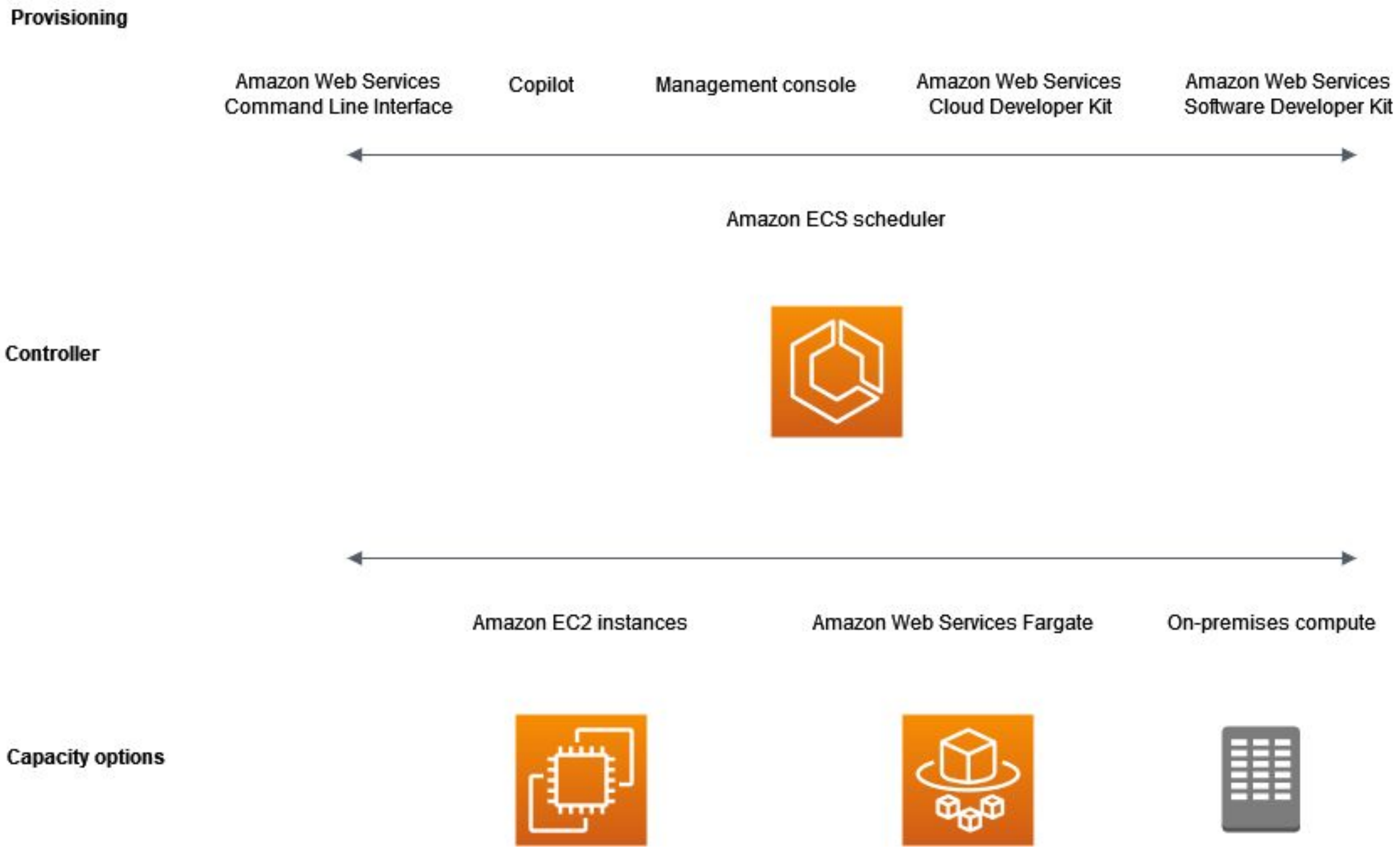
### Support batch processing

Plan, schedule, and run batch computing workloads across AWS services, including Amazon Elastic Compute Cloud (EC2), AWS Fargate, and Amazon EC2 Spot Instances.

### Train NLP and AI/ML models

Train natural language processing (NLP) and other artificial intelligence (AI) / machine learning (ML) models without managing the infrastructure by using Amazon ECS with AWS Fargate.

# Amazon Elastic Container Service Layers



Provisioning

Amazon Web Services Command Line Interface

Copilot

Management console

Amazon Web Services Cloud Developer Kit

Amazon Web Services Software Developer Kit

Amazon ECS scheduler

Controller

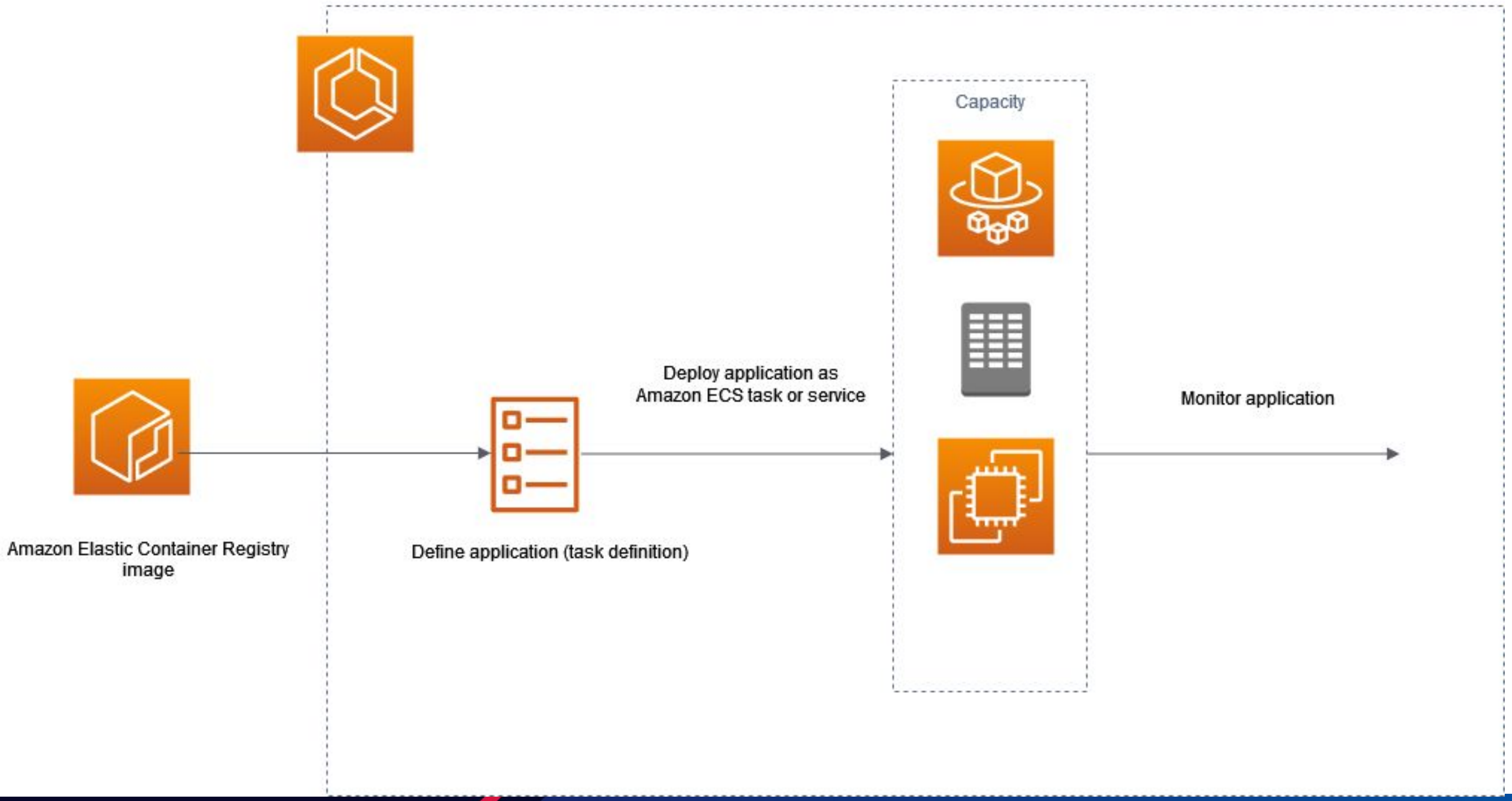
Amazon EC2 instances

Amazon Web Services Fargate

On-premises compute

Capacity options

# Amazon ECS Application Lifecycle



Capacity
Deploy application as Amazon ECS task or service
Monitor application
Amazon Elastic Container Registry image
Define application (task definition)

# ECS Terminologies

# ECS Cluster

An Amazon ECS cluster is a logical grouping of tasks or services. In addition to tasks and services, a cluster consists of the following resources:

- The infrastructure capacity which can be any of the following:
    - Amazon EC2 instances in the AWS cloud
    - Serverless (AWS Fargate (Fargate)) in the AWS cloud
    - On-premises virtual machines (VM) or servers (Using **ECS Anywhere:** Self-Managed Containers - Amazon ECS Anywhere - AWS)

# ECS Capacity Provider

**Fargate**: With Amazon ECS on AWS Fargate capacity providers, you can use both Fargate and Fargate Spot capacity with your Amazon ECS tasks. With Fargate Spot, you can run interruption tolerant Amazon ECS tasks at a rate that's discounted compared to the Fargate price. Fargate Spot runs tasks on spare compute capacity. When AWS needs the capacity back, your tasks are interrupted with a two-minute warning.

**EC2**: With AWS ECS on EC2 capacity providers, you can specify the auto scaling group and the setting required for the autoscaling group.

# AWS ECS Task Definition

A *task definition* is a blueprint for your application. It is a text file in JSON format that describes the parameters and one or more containers that form your application.

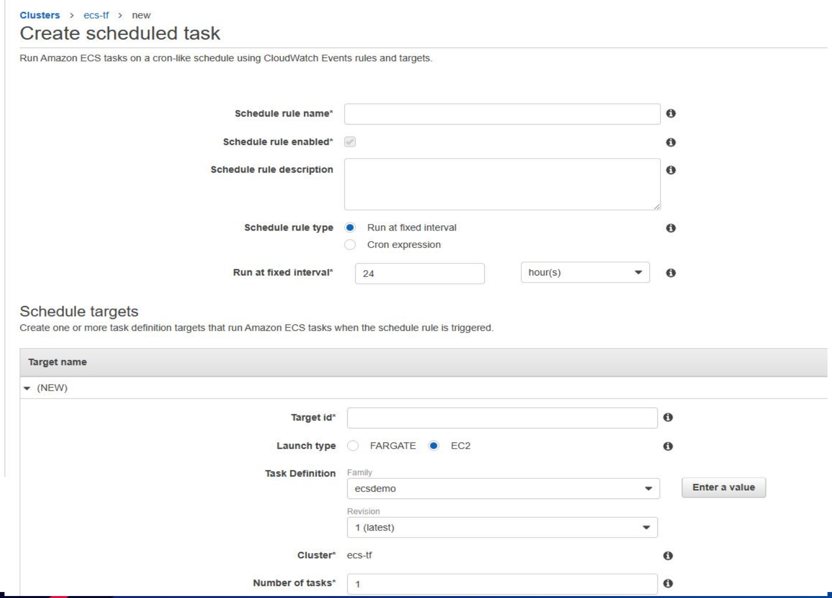
The following are some of the parameters that you can specify in a task definition:

- The Docker image to use with each container in your task
- How much CPU and memory to use with each task or each container within a task
- The launch type to use, which determines the infrastructure that your tasks are hosted on
- The Docker networking mode to use for the containers in your task
- The logging configuration to use for your tasks
- Whether the task continues to run if the container finishes or fails
- The command that the container runs when it's started
- Any data volumes that are used with the containers in the task
- The IAM role that your tasks use

# ECS Service vs Tasks

**Service** : Typically used for long-running tasks / containers that can be stopped and restarted. E.g. A web application

**Task**: A standalone task that is used to run a short running task. For e.g. A batch job, or a scheduled cron job to run at a specific time everyday and then exit



Clusters > ecs-tf > new

Create scheduled task

Run Amazon ECS tasks on a cron-like schedule using CloudWatch Events rules and targets.

Schedule rule name*

Schedule rule enabled* ☑

Schedule rule description

Schedule rule type ● Run at fixed interval ○ Cron expression

Run at fixed interval* 24 hour(s)

Schedule targets

Create one or more task definition targets that run Amazon ECS tasks when the schedule rule is triggered.

Target name

(NEW)

Target id*

Launch type ○ FARGATE ● EC2

Task Definition Family: ecsdemo Enter a value

Revision: 1 (latest)

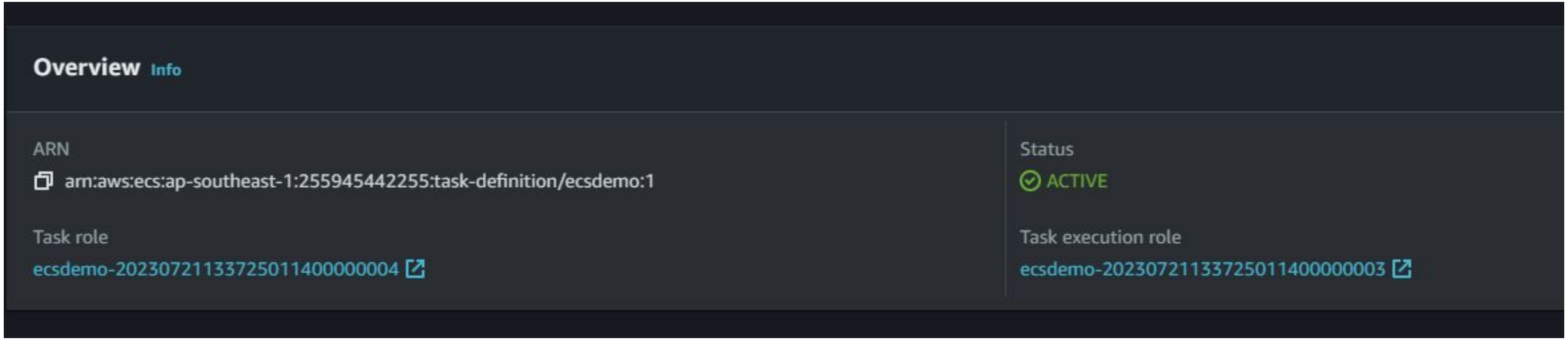
Cluster* ecs-tf

Number of tasks* 1

# ECS TaskExecutionRole vs TaskRole

**<u>TaskExecutionRole</u>**: IAM role that executes ECS actions such as pulling the image and storing the application logs in cloudwatch.

**<u>TaskRole</u>**: IAM role used by the task itself. For example, if your container wants to call other AWS services like S3, SQS, etc then those permissions would need to be covered by the TaskRole



Overview Info

ARN
arn:aws:ecs:ap-southeast-1:255945442255:task-definition/ecsdemo:1

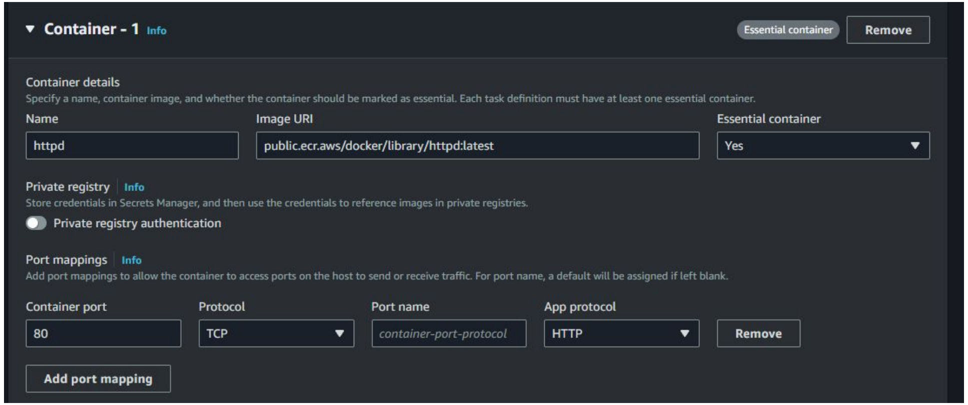
Status
ACTIVE

Task role
ecsdemo-20230721133725011400000004

Task execution role
ecsdemo-20230721133725011400000003

# Activity 1: Create ECS cluster and task definition using a default image

- Create a ECS cluster with Fargate type
- Create a task definition with the  following specs (leave all else as default other than the ones mentioned below):
  - Name: yourname-httpd
  - Under container-1
    - Name: httpd
    - image uri: `public.ecr.aws/docker/library/httpd:latest`



▼ Container - 1 Info

Container details

Specify a name, container image, and whether the container should be marked as essential. Each task definition must have at least one essential container.

| Name | Image URI | Essential container |
|---|---|---|
| httpd | public.ecr.aws/docker/library/httpd:latest | Yes ▼ |

Private registry Info

Store credentials in Secrets Manager, and then use the credentials to reference images in private registries.

○ Private registry authentication

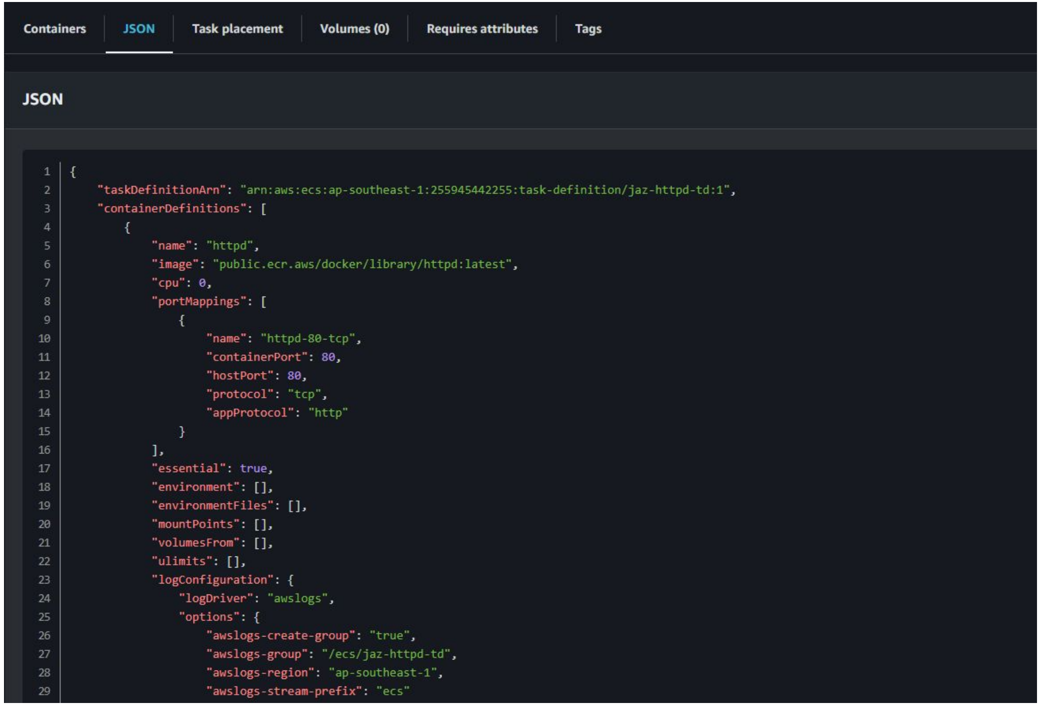
Port mappings Info

Add port mappings to allow the container to access ports on the host to send or receive traffic. For port name, a default will be assigned if left blank.

| Container port | Protocol | Port name | App protocol |
|---|---|---|---|
| 80 | TCP ▼ | container-port-protocol | HTTP ▼ |

# Task Definition JSON

- You can view this after you've finished creating your task definition. (Task definitions -> Click on yours -> JSON)

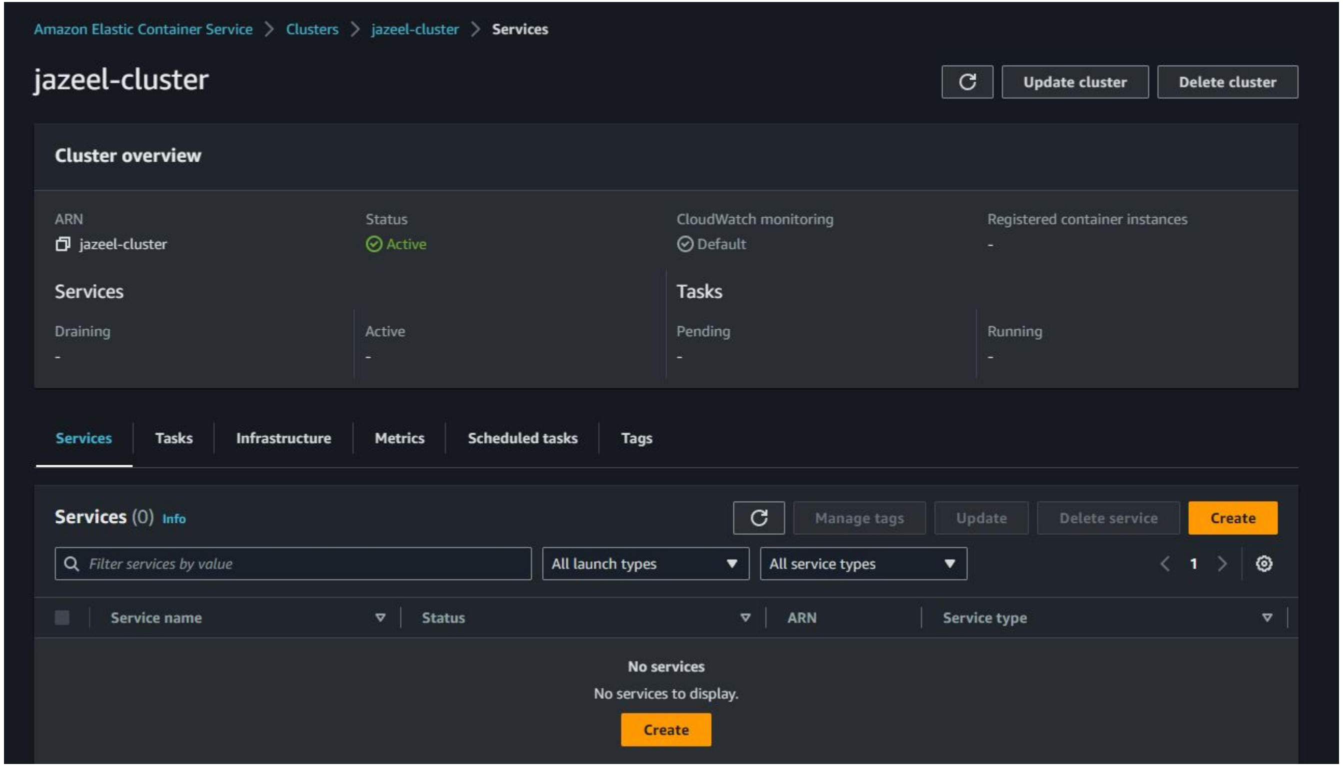


```
{
    "taskDefinitionArn": "arn:aws:ecs:ap-southeast-1:255945442255:task-definition/jaz-httpd-td:1",
    "containerDefinitions": [
        {
            "name": "httpd",
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "cpu": 0,
            "portMappings": [
                {
                    "name": "httpd-80-tcp",
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp",
                    "appProtocol": "http"
                }
            ],
            "essential": true,
            "environment": [],
            "environmentFiles": [],
            "mountPoints": [],
            "volumesFrom": [],
            "ulimits": [],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-create-group": "true",
                    "awslogs-group": "/ecs/jaz-httpd-td",
                    "awslogs-region": "ap-southeast-1",
                    "awslogs-stream-prefix": "ecs"
```

# Activity 1: Create ECS service

- Go to the ECS cluster that you've created and go to the **Services** tab.
- Click on **Create (Can refer to next slide)**
- Under Deployment Configuration
  - Family: Specify the task definition which you created in the previous slide
  - Service name: httpd
- Under Networking
  - Create your own security group with inbound port 80
  - For VPC: Feel free to use any VPC, but remember to use a public subnet
  - Public IP: Turned on

# Service Creation



## jazeel-cluster

### Cluster overview

ARN
jazeel-cluster

Status
Active

CloudWatch monitoring
Default

Registered container instances
-

**Services**

Draining
-

Active
-

**Tasks**

Pending
-

Running
-

**Services (0)**

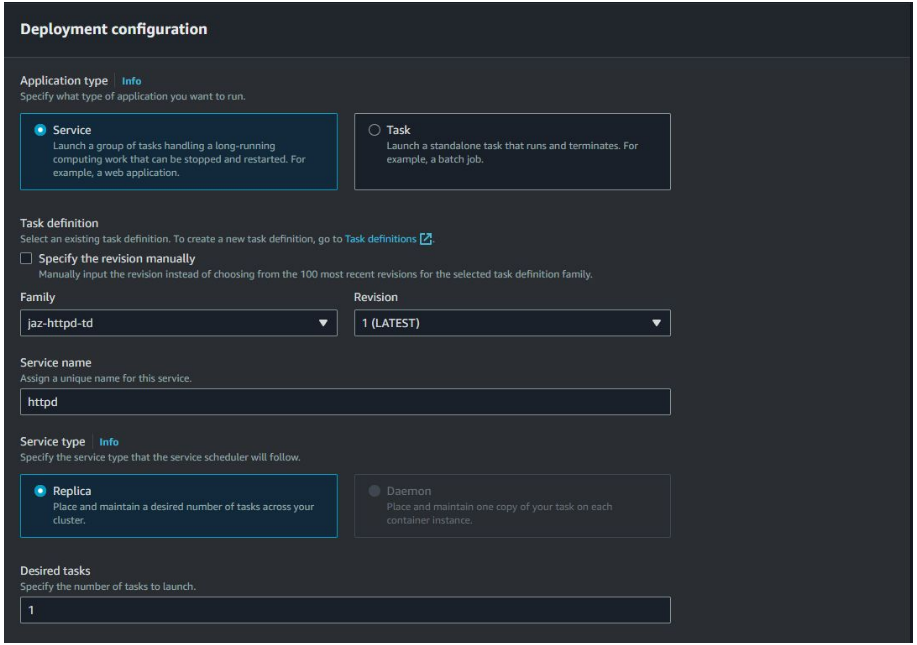
| | Service name | Status | ARN | Service type |
|---|---|---|---|---|

**No services**

No services to display.

# Activity 1: Create ECS cluster and service using a default image

- Service configurations:



**Deployment configuration**

Application type | Info

Specify what type of application you want to run.

● Service
Launch a group of tasks handling a long-running computing work that can be stopped and restarted. For example, a web application.

○ Task
Launch a standalone task that runs and terminates. For example, a batch job.

Task definition

Select an existing task definition. To create a new task definition, go to Task definitions.

□ Specify the revision manually
Manually input the revision instead of choosing from the 100 most recent revisions for the selected task definition family.

Family: jaz-httpd-td

Revision: 1 (LATEST)

Service name

Assign a unique name for this service.

httpd

Service type | Info

Specify the service type that the service scheduler will follow.

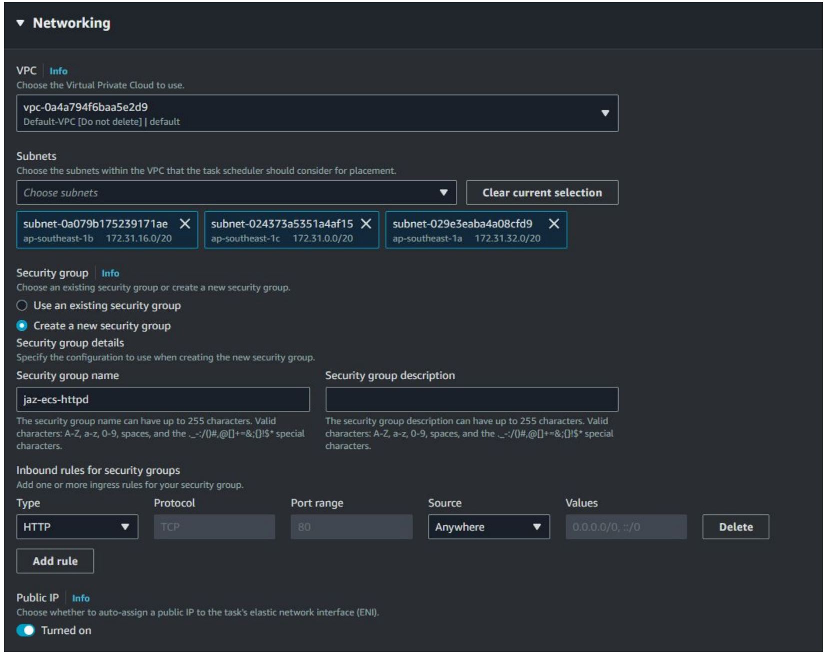
● Replica
Place and maintain a desired number of tasks across your cluster.

○ Daemon
Place and maintain one copy of your task on each container instance.

Desired tasks

Specify the number of tasks to launch.

1



**Networking**

VPC | Info

Choose the Virtual Private Cloud to use.

vpc-0a4a794f6baa5e2d9
Default-VPC [Do not delete] | default

Subnets

Choose the subnets within the VPC that the task scheduler should consider for placement.

*Choose subnets*

subnet-0a079b175239171ae ✕
ap-southeast-1b 172.31.16.0/20

subnet-024373a5351a4af15 ✕
ap-southeast-1c 172.31.0.0/20

subnet-029e3eaba4a08cfd9 ✕
ap-southeast-1a 172.31.32.0/20

Security group | Info

Choose an existing security group or create a new security group.

○ Use an existing security group

● Create a new security group

Security group details

Specify the configuration to use when creating the new security group.

Security group name: jaz-ecs-httpd

The security group name can have up to 255 characters. Valid characters: A-Z, a-z, 0-9, spaces, and the ._-:/()#,@[]+=&;{}!$* special characters.

Security group description:

The security group description can have up to 255 characters. Valid characters: A-Z, a-z, 0-9, spaces, and the ._-:/()#,@[]+=&;{}!$* special characters.

Inbound rules for security groups

Add one or more ingress rules for your security group.

| Type | Protocol | Port range | Source | Values |
|---|---|---|---|---|
| HTTP | TCP | 80 | Anywhere | 0.0.0.0/0, ::/0 |

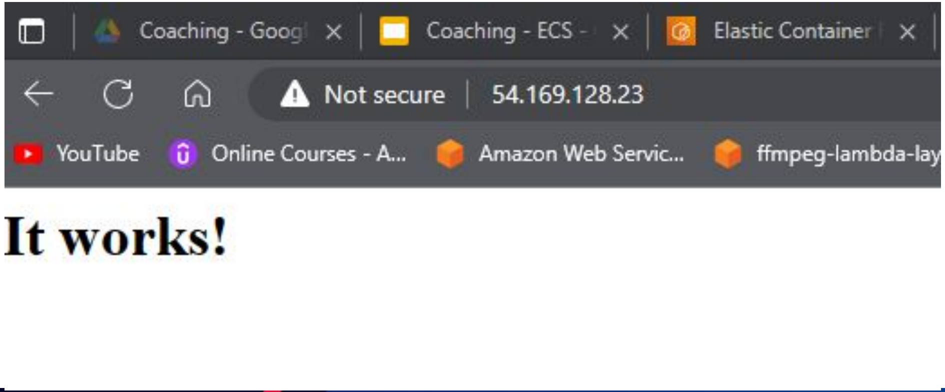
Public IP | Info

Choose whether to auto-assign a public IP to the task's elastic network interface (ENI).

Turned on

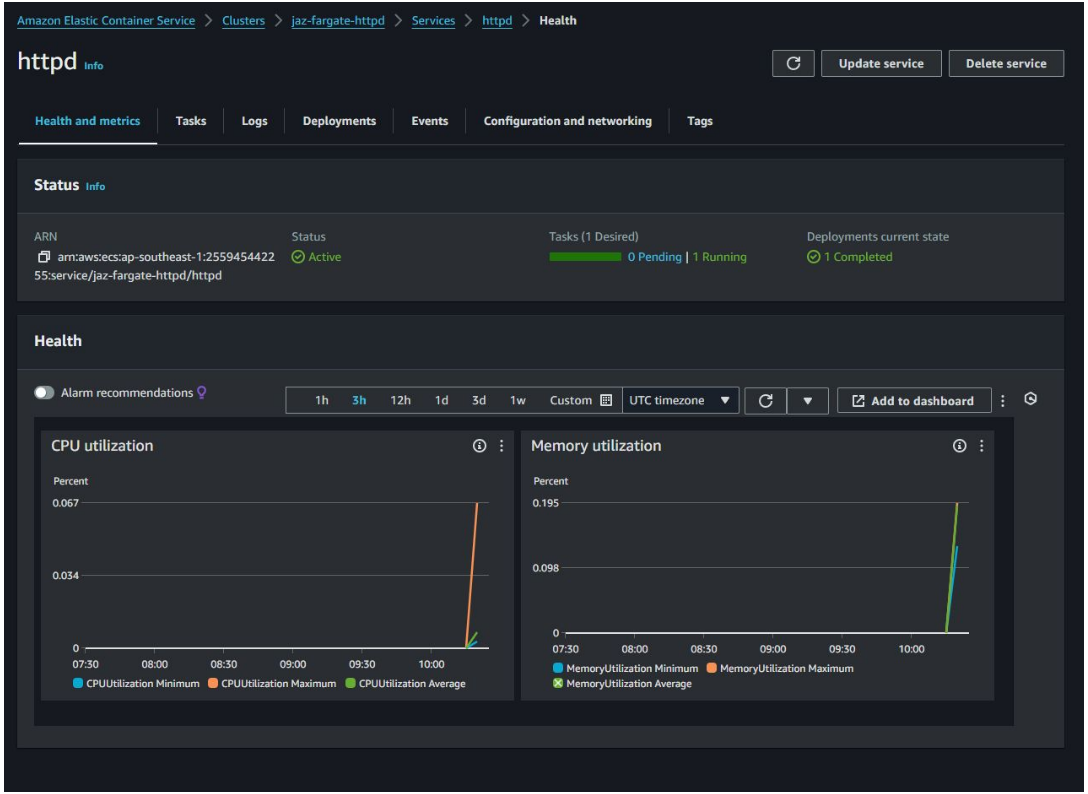
# Activity

1) Access the cluster you have created
2) Go to the **tasks** tab
3) Click on the task
4) Obtain the public IP and try to access it in a browser tab
5) Delete the service that you've created.
6) Retain the ECS cluster for the next activity



It works!

# Service - Health and metrics



Amazon Elastic Container Service > Clusters > jaz-fargate-httpd > Services > httpd > Health

httpd Info

**Status** Info

ARN: arn:aws:ecs:ap-southeast-1:255945442255:service/jaz-fargate-httpd/httpd

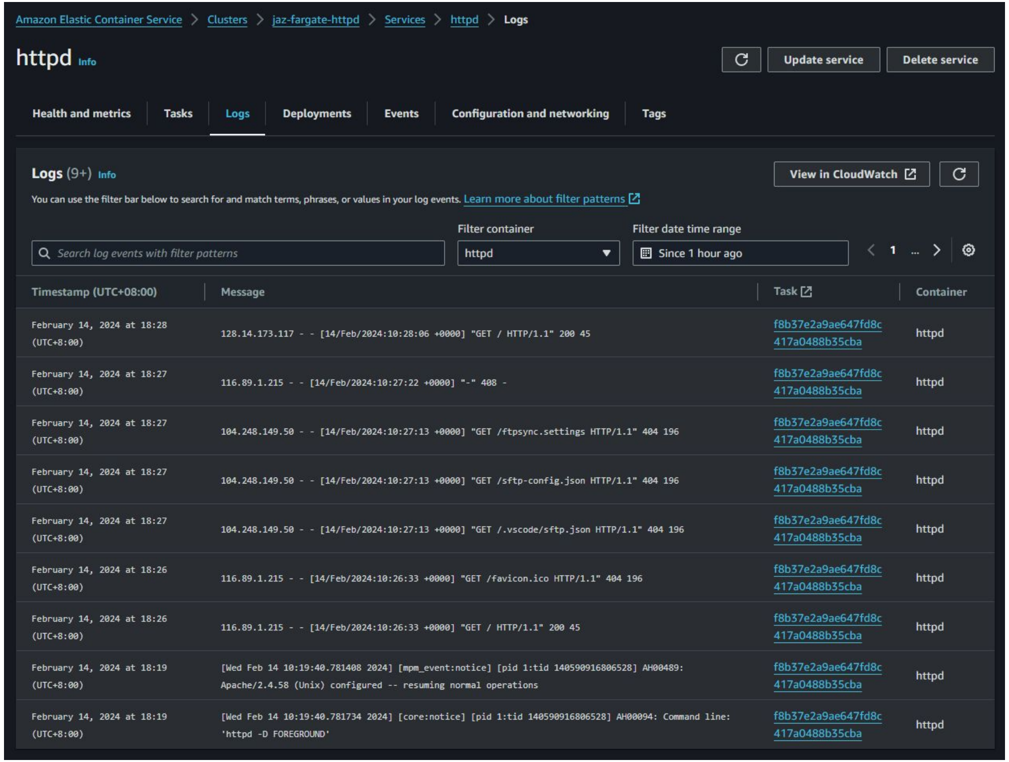
Status: Active

Tasks (1 Desired): 0 Pending | 1 Running

Deployments current state: 1 Completed

**Health**

# Service - Logs


Amazon Elastic Container Service > Clusters > jaz-fargate-httpd > Services > httpd > Logs
httpd Info
Update service
Delete service
Health and metrics
Tasks
Logs
Deployments
Events
Configuration and networking
Tags
View in CloudWatch
1 ...

**Logs** (9+) Info

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

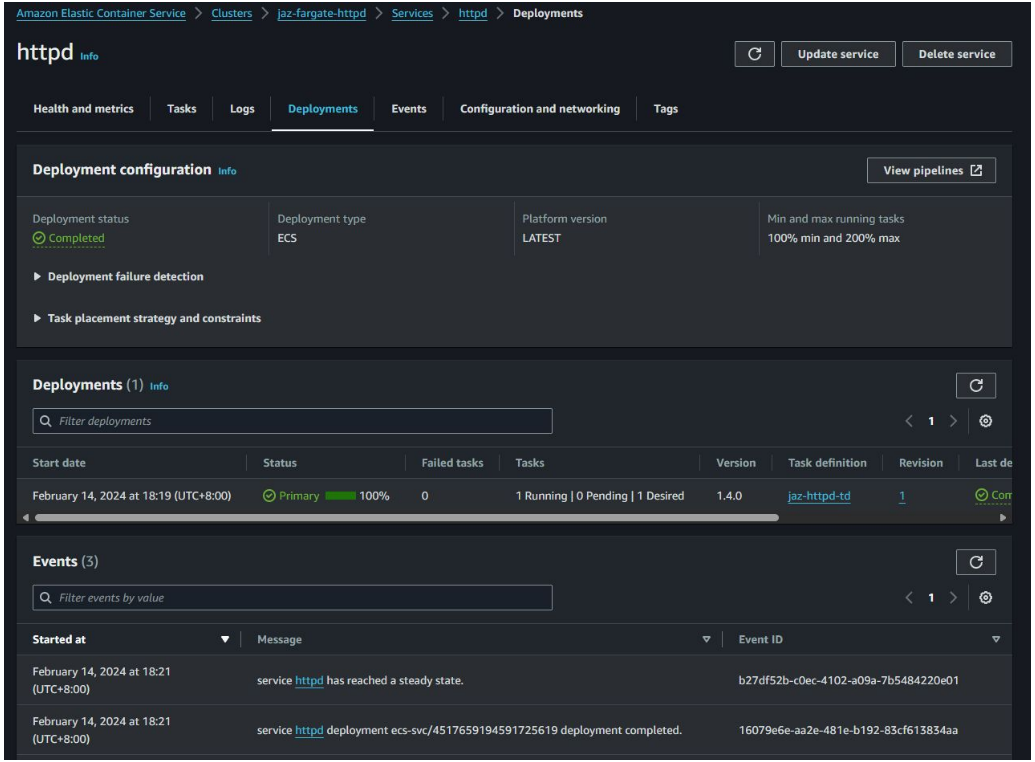
Search log events with filter patterns

Filter container: httpd

Filter date time range: Since 1 hour ago

| Timestamp (UTC+08:00) | Message | Task | Container |
|---|---|---|---|
| February 14, 2024 at 18:28 (UTC+8:00) | 128.14.173.117 - - [14/Feb/2024:10:28:06 +0000] "GET / HTTP/1.1" 200 45 | f8b37e2a9ae647fd8c417a0488b35cba | httpd |
| February 14, 2024 at 18:27 (UTC+8:00) | 116.89.1.215 - - [14/Feb/2024:10:27:22 +0000] "-" 408 - | f8b37e2a9ae647fd8c417a0488b35cba | httpd |
| February 14, 2024 at 18:27 (UTC+8:00) | 104.248.149.50 - - [14/Feb/2024:10:27:13 +0000] "GET /ftpsync.settings HTTP/1.1" 404 196 | f8b37e2a9ae647fd8c417a0488b35cba | httpd |
| February 14, 2024 at 18:27 (UTC+8:00) | 104.248.149.50 - - [14/Feb/2024:10:27:13 +0000] "GET /sftp-config.json HTTP/1.1" 404 196 | f8b37e2a9ae647fd8c417a0488b35cba | httpd |
| February 14, 2024 at 18:27 (UTC+8:00) | 104.248.149.50 - - [14/Feb/2024:10:27:13 +0000] "GET /.vscode/sftp.json HTTP/1.1" 404 196 | f8b37e2a9ae647fd8c417a0488b35cba | httpd |
| February 14, 2024 at 18:26 (UTC+8:00) | 116.89.1.215 - - [14/Feb/2024:10:26:33 +0000] "GET /favicon.ico HTTP/1.1" 404 196 | f8b37e2a9ae647fd8c417a0488b35cba | httpd |
| February 14, 2024 at 18:26 (UTC+8:00) | 116.89.1.215 - - [14/Feb/2024:10:26:33 +0000] "GET / HTTP/1.1" 200 45 | f8b37e2a9ae647fd8c417a0488b35cba | httpd |
| February 14, 2024 at 18:19 (UTC+8:00) | [Wed Feb 14 10:19:40.781408 2024] [mpm_event:notice] [pid 1:tid 140590916806528] AH00489: Apache/2.4.58 (Unix) configured -- resuming normal operations | f8b37e2a9ae647fd8c417a0488b35cba | httpd |
| February 14, 2024 at 18:19 (UTC+8:00) | [Wed Feb 14 10:19:40.781734 2024] [core:notice] [pid 1:tid 140590916806528] AH00094: Command line: 'httpd -D FOREGROUND' | f8b37e2a9ae647fd8c417a0488b35cba | httpd |

# Service - Deployment



Amazon Elastic Container Service > Clusters > jaz-fargate-httpd > Services > httpd > Deployments
httpd Info
Update service
Delete service
Health and metrics
Tasks
Logs
Deployments
Events
Configuration and networking
Tags

**Deployment configuration** Info

| Deployment status | Deployment type | Platform version | Min and max running tasks |
|---|---|---|---|
| ✓ Completed | ECS | LATEST | 100% min and 200% max |

▶ Deployment failure detection

▶ Task placement strategy and constraints

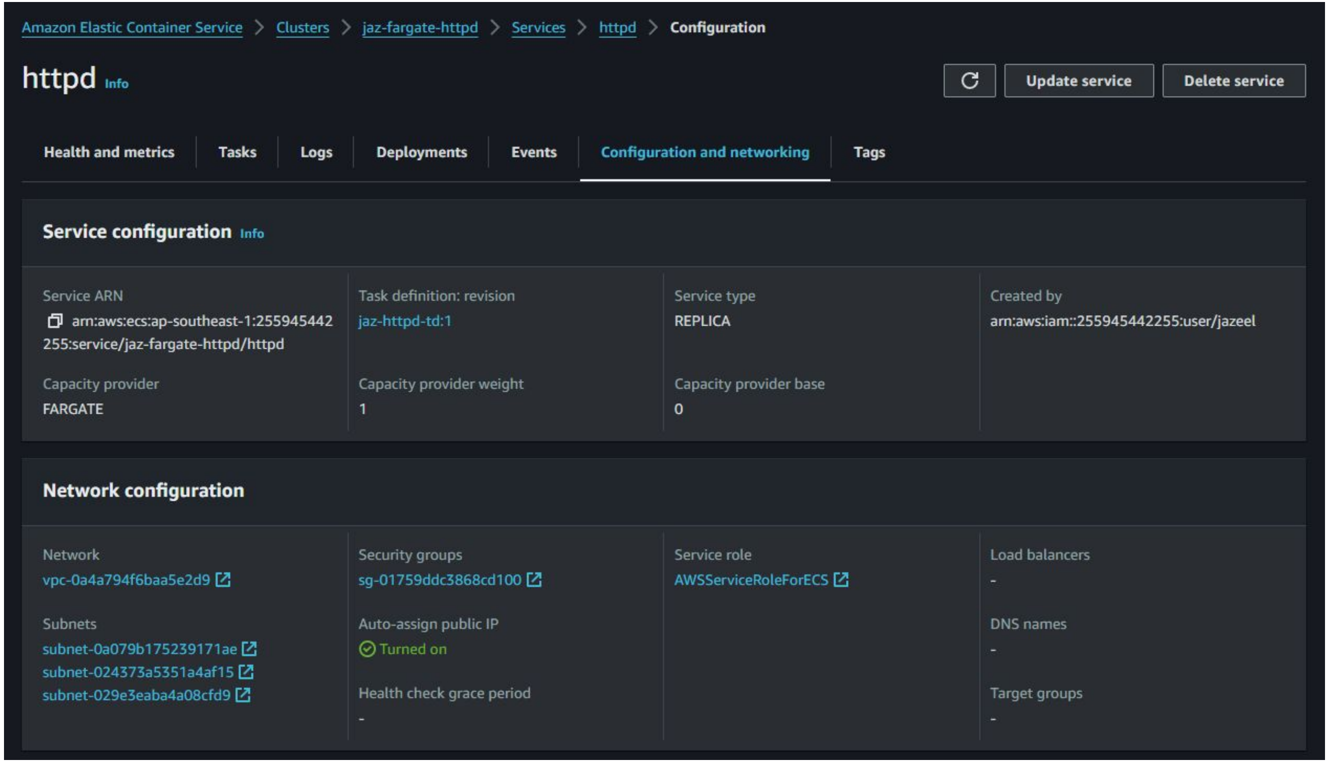
**Deployments** (1) Info

| Start date | Status | Failed tasks | Tasks | Version | Task definition | Revision | Last de |
|---|---|---|---|---|---|---|---|
| February 14, 2024 at 18:19 (UTC+8:00) | ✓ Primary 🟩 100% | 0 | 1 Running \| 0 Pending \| 1 Desired | 1.4.0 | jaz-httpd-td | 1 | ✓ Com |

**Events** (3)

| Started at | Message | Event ID |
|---|---|---|
| February 14, 2024 at 18:21 (UTC+8:00) | service httpd has reached a steady state. | b27df52b-c0ec-4102-a09a-7b5484220e01 |
| February 14, 2024 at 18:21 (UTC+8:00) | service httpd deployment ecs-svc/4517659194591725619 deployment completed. | 16079e6e-aa2e-481e-b192-83cf613834aa |

# Service - Networking


Amazon Elastic Container Service > Clusters > jaz-fargate-httpd > Services > httpd > Configuration
Update service
Delete service
Health and metrics
Tasks
Logs
Deployments
Events
Configuration and networking
Tags

httpd Info

**Service configuration** Info

Service ARN
arn:aws:ecs:ap-southeast-1:255945442255:service/jaz-fargate-httpd/httpd

Task definition: revision
jaz-httpd-td:1

Service type
REPLICA

Created by
arn:aws:iam::255945442255:user/jazeel

Capacity provider
FARGATE

Capacity provider weight
1

Capacity provider base
0

**Network configuration**

Network
vpc-0a4a794f6baa5e2d9

Security groups
sg-01759ddc3868cd100

Service role
AWSServiceRoleForECS

Load balancers
-

Subnets
subnet-0a079b175239171ae
subnet-024373a5351a4af15
subnet-029e3eaba4a08cfd9

Auto-assign public IP
✓ Turned on

DNS names
-

Health check grace period
-

Target groups
-

# Post-Activity for Activity 1

1) Delete the service that you’ve created.
2) Retain the ECS cluster for the next activity

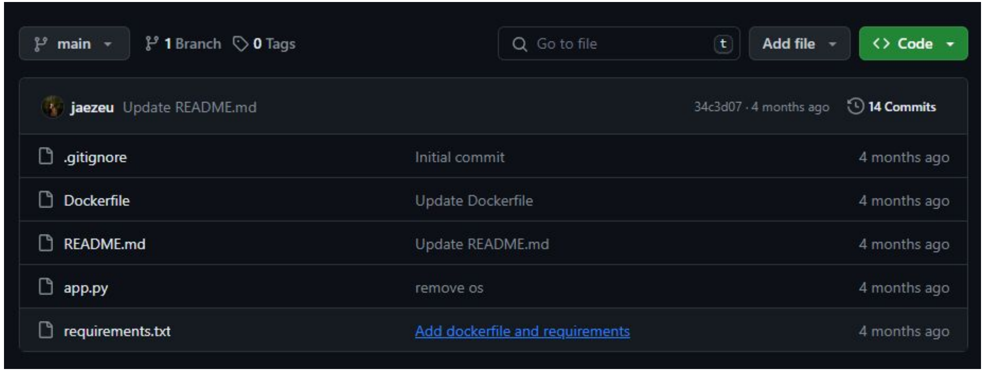
# Activity 2: Using custom image in ECS

- Create a new private repository in AWS ECR
- Using the sample code in the next page, build your own docker image and push it to that ECR (Feel free to also try with other docker images or code if you have)
- Create a new ECS task definition using the custom image you’ve  pushed to your ECR (You can obtain the image URI from your repo). Remember to also update the container port to be the same port as your docker image
- Create a new service in your ECS cluster with this new task definition, with the respective security group rules required.

# Activity 2: Using custom image in ECS

- app.py - hello-flask/app.py at main · jaezeu/hello-flask · GitHub
- requirements.txt - hello-flask/requirements.txt at main · jaezeu/hello-flask · GitHub
- Dockerfile - hello-flask/Dockerfile at main · jaezeu/hello-flask · GitHub

If you do not already have, Create your own github repository with the files above. Files you should have in your git repo:



jaezeu Update README.md 34c3d07 · 4 months ago

| | | |
|---|---|---|
| .gitignore | Initial commit | 4 months ago |
| Dockerfile | Update Dockerfile | 4 months ago |
| README.md | Update README.md | 4 months ago |
| app.py | remove os | 4 months ago |
| requirements.txt | Add dockerfile and requirements | 4 months ago |

# Post-Activity for Activity 2

1) Delete all the resources you’ve created in today’s activity:
    a) Security groups
    b) ECS cluster + Task definition + Service
    c) ECR repository

# Post-Activity

Some resources you can refer to (which would help you write lesser code), in order to create all of our today’s infra:

ECS cluster + Task definition + Service: terraform-aws-modules/terraform-aws-ecs: Terraform module to create AWS ECS resources 🇺🇦 (github.com)

ECR Repository: aws_ecr_repository | Resources | hashicorp/aws | Terraform | Terraform Registry

ECS security group: terraform-aws-modules/terraform-aws-security-group: Terraform module to create AWS Security Group resources 🇺🇦 (github.com)