

AWS LAB Autoscaling | Monarch Nigam

EXERCISE 11.1

Configure and Launch an Application Using Auto Scaling

1. From the EC2 Dashboard, create a launch configuration using the Ubuntu Server LTS AMI on the Quick Start tab and a t2.micro instance type.

The screenshot shows the AWS EC2 Launch Templates interface. On the left, there's a sidebar with 'Recent' and 'Quick Start' tabs, and a search bar. The main area has a heading 'Application and OS Images (Amazon Machine Image)'. It lists several AMI icons: Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Debian. Below this is a section for 'Amazon Machine Image (AMI)' with a detailed description of the selected Ubuntu Server 24.04 LTS (HVM) AMI. To the right is a 'Summary' panel containing sections for Software Image (AMI), Virtual server type (instance type), Firewall (security group), Storage (volumes), and a Free tier notice. At the bottom are 'Cancel' and 'Create launch template' buttons.

2. On the Create Launch Configuration page, name your configuration. You don't need to select an IAM role or enable monitoring for this exercise.

Save the following commands to a file on your local computer, name the file `start.sh`, expand the Advanced Details section, click the As File radio button, and select your `start.sh` script from your computer. This will install the Apache web server and create a simple `index.html` web page:

```
#!/bin/bash
```

```
apt-get update
apt-get install -y apache2
echo "Welcome to my website"> index.html
```

The screenshot shows the 'Create launch template' page in the AWS Management Console. In the 'User data - optional' section, the following command is entered:

```
#!/bin/bash
apt-get update
apt-get install -y apache2
echo "Welcome to MONARCH website"> /var/www/html/index.html
```

The 'Summary' panel on the right provides details about the instance configuration, including the AMI, instance type (t2.micro), and storage.

3. `cp index.html /var/www/html`

4. Choose (or create) a security group that will permit all HTTP traffic via port 80 and then create the configuration, making sure to select a valid key pair so that you'll be able to use SSH to log into the instance later if necessary.

The screenshot shows the 'Create launch template' page with two security group rules defined under 'Inbound Security Group Rules':

- Security group rule 1 (TCP, 22, 0.0.0.0/0)**: Type: ssh, Protocol: TCP, Port range: 22, Source type: Anywhere.
- Security group rule 2 (TCP, 80, 0.0.0.0/0)**: Type: HTTP, Protocol: TCP, Port range: 80, Source type: Anywhere.

The 'Summary' panel on the right provides details about the instance configuration, including the AMI, instance type (t2.micro), and storage.

The screenshot shows the AWS Launch Templates page. On the left, there's a navigation sidebar with sections like Instances, Images, and Launch Templates. The main area has a table titled "Launch Templates (1) Info". The table contains one row with the following data:

| Launch Template ID | Launch Template Name | Default Version | Latest Version | Create Time | Created By |
|----------------------|----------------------|-----------------|----------------|--------------------------|------------------------------|
| lt-0b15e26303fabc5d7 | Monarchtemplate | 1 | 1 | 2025-04-20T22:02:23.000Z | arn:aws:iam::160885292183:ro |

Below the table, there's a section titled "Select a launch template" with a dropdown menu.

5. Create an Auto Scaling group that will use your new launch configuration. Give your group a name and leave the Group Size value at 1 instance.

The screenshot shows the AWS Auto Scaling groups page. On the left, there's a navigation sidebar with the EC2 icon selected. The main area has a table titled "Auto Scaling groups (1) Info". The table contains one row with the following data:

| Name | Launch template/configuration | Instances | Status | Desired capacity | Min | Max | Availability Zones |
|---------------------------------|-----------------------------------|-----------|----------------------|------------------|-----|-----|--------------------|
| MonarchAutoscalingGroupdeleteit | Monarchtemplate Version Default | 0 | Updating capacity... | 1 | 1 | 1 | us-east-1a |

6. Select a virtual private cloud (VPC) into which your instances will launch, click once inside the Subnet field, and select the subnet where you'd like your instances to live. Assuming you're working with your default VPC, any one of the public subnets you'll be shown should be fine.
7. On the Configure Scaling Policies page, set Minimum Capacity to 1 and Maximum Capacity to 2.

Desired capacity type
Choose the unit of measurement for the desired capacity value. vCPUs and Memory(GiB) are only supported for mixed instances groups configured with a set of instance attributes.

Units (number of instances) ▾

Desired capacity

Specify your group size.

1

Scaling Info

You can resize your Auto Scaling group manually or automatically to meet changes in demand.

Scaling limits

Set limits on how much your desired capacity can be increased or decreased.

Min desired capacity

1

Equal or less than desired capacity

Max desired capacity

2

Equal or greater than desired capacity

Automatic scaling - optional

Choose whether to use a target tracking policy Info

You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

No scaling policies

Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

Target tracking scaling policy

Choose a CloudWatch metric and target value and let the scaling policy adjust the desired capacity in proportion to the metric's value.

8. Select Target Tracking Scaling Policy and then leave the Metric Type value as

Average CPU Utilization and edit its Target Value to 5 percent.

This unusually low value will make it easier to confirm that the autoscaling is

working—normally a value of 70 percent to 80 percent would make sense.

Scaling - Info
You can resize your Auto Scaling group manually or automatically to meet changes in demand.

Scaling limits
Set limits on how much your desired capacity can be increased or decreased.

| | |
|----------------------|----------------------|
| Min desired capacity | Max desired capacity |
| 1 | 2 |

Equal or less than desired capacity Equal or greater than desired capacity

Automatic scaling - optional
Choose whether to use a target tracking policy [Info](#)

You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

| | |
|--|--|
| <input type="radio"/> No scaling policies Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand. | <input checked="" type="radio"/> Target tracking scaling policy Choose a CloudWatch metric and target value and let the scaling policy adjust the desired capacity in proportion to the metric's value. |
|--|--|

Scaling policy name
Target Tracking Policy Monarch

Metric type [Info](#)
Monitored metric that determines if resource utilization is too low or high. If using EC2 metrics, consider enabling detailed monitoring for better scaling performance.

| |
|-------------------------|
| Average CPU utilization |
|-------------------------|

Target value
5

Instance warmup [Info](#)
300 seconds

Disable scale in to create only a scale-out policy

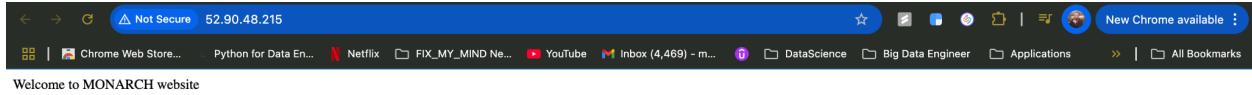
- For this exercise, you don't need to configure notifications or tags. Click Create to finish creating the group.

Auto Scaling groups (1) [Info](#)

| Name | Launch template/configuration | Instances | Status | Desired capacity | Min | Max | Availability Zones |
|-------------------------|-----------------------------------|-----------|----------------------|------------------|-----|-----|--------------------|
| monarchautoscalinggroup | Monarchtemplate Version Default | 0 | Updating capacity... | 1 | 1 | 2 | us-east-1 |

- Your group will immediately try to start the first instance. It may take a few minutes for everything to load. When it does, point your browser to the IP address associated with the instance (you can retrieve that from the EC2 Instances Dashboard).

Considering how small the `index.html` page is, it might be hard to get the Auto Scaling tool to launch a second instance. You'll probably need to "cheat" by running a busywork command such as the following on the server command line to raise your CPU level high enough:



EXERCISE 11.2

Sync Two S3 Buckets as Cross-Region Replicas

1. Use the S3 Console to create two buckets in two separate regions. One will be your source and the other your destination.

A screenshot of the AWS S3 console. The top navigation bar includes the AWS logo, search bar, and account information ('United States (N. Virginia) monarch.smiclass.2025'). The main content area shows a success message: 'Successfully created bucket "monarch-source-bucket". To upload files and folders, or to configure additional bucket settings, choose View details.' Below this, there's an 'Account snapshot' section with an update frequency of 'updated every 24 hours' and a note about storage lens visibility. At the bottom, there are tabs for 'General purpose buckets' (selected) and 'Directory buckets', along with a 'Create bucket' button. A table lists the single bucket: 'monarch-source-bucket' (Name), 'US East (N. Virginia) us-east-1' (AWS Region), 'View analyzer for us-east-1' (IAM Access Analyzer), and 'April 20, 2025, 18:42:45 (UTC-05:00)' (Creation date).

Successfully created bucket "monarch-destination-bucket"
To upload files and folders, or to configure additional bucket settings, choose [View details](#).

Account snapshot - updated every 24 hours [All AWS Regions]

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

[View Storage Lens dashboard](#)

[General purpose buckets](#) [Directory buckets](#)

General purpose buckets (2) [Info] All AWS Regions

Buckets are containers for data stored in S3.

| Name | AWS Region | IAM Access Analyzer | Creation date |
|--|-----------------------------------|---|--------------------------------------|
| monarch-destination-bucket | US West (N. California) us-west-1 | View analyzer for us-west-1 | April 20, 2025, 18:43:41 (UTC-05:00) |
| monarch-source-bucket | US East (N. Virginia) us-east-1 | View analyzer for us-east-1 | April 20, 2025, 18:42:45 (UTC-05:00) |

[Copy ARN](#) [Empty](#) [Delete](#) [Create bucket](#)

- Click the name of your source bucket in the S3 Console, then click the Management tab, then the Create Replication Rule button. If versioning wasn't already enabled for the bucket, you can click the Enable Versioning button to set it.

Successfully edited Bucket Versioning
To transition, archive, or delete older object versions, configure lifecycle rules for this bucket.

monarch-source-bucket [Info]

Objects | Metadata | **Properties** | Permissions | Metrics | Management | Access Points

Bucket overview

AWS Region: US East (N. Virginia) us-east-1

Amazon Resource Name (ARN): arn:aws:s3:::monarch-source-bucket

Creation date: April 20, 2025, 18:42:45 (UTC-05:00)

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Bucket Versioning: Enabled

Multi-factor authentication (MFA) delete: Disabled

- Define the objects you want replicated from your source bucket. Choose Apply To All Objects for this exercise.

4. In the Destination Definition section, select the name of the second bucket you created as the destination.

The screenshot shows the AWS S3 Replication Rules configuration interface. At the top, the navigation path is: Amazon S3 > Buckets > monarch-source-bucket > Replication rules > Create replication rule. The main form is titled "Source bucket" and contains the following details:

- Source bucket:** monarch-source-bucket
- Source Region:** US East (N. Virginia) us-east-1
- Choose a rule scope:** Apply to all objects in the bucket

Below this is the "Destination" section:

- Destination:** monarch-destination-bucket
- Bucket name:** monarch-destination-bucket
- Object versioning enabled:** (checkbox checked) This bucket now has object versioning enabled. If you need to suspend versioning you can do so in [Bucket properties](#) and you will no longer be able to use it as a destination bucket for this rule.
- Destination Region:** US West (N. California) us-west-1

5. Choose From Existing IAM Roles and select Create New Role from the drop-down menu. This will create the permissions necessary to move assets between buckets. You can then review your settings and save the rule.

Screenshot of the AWS S3 Replication rules configuration page for the 'monarch-source-bucket'.

Replication rules [Info](#)

Replication enables automatic and asynchronous copying of objects across buckets in the same or different AWS Regions. A replication configuration is a set of rules that define what options should be applied to a group of objects during replication.

Replication configuration settings

Configuration settings affect all replication rules in the bucket.

| | | | |
|---------------|---------------------------------|----------|---|
| Source bucket | monarch-source-bucket | IAM role | s3cr_role_for_monarch-source-bucket |
| Source Region | US East (N. Virginia) us-east-1 | | |

Actions [Create replication job](#) [Edit](#)

Replication rules (1)

Use replication rules to define options you want Amazon S3 to apply during replication such as server-side encryption, replica ownership, transitioning replicas to another storage class, and more. [Learn more](#)

| Replication rule name | Status | Destination bucket | Destination Region | Priority | Scope | Storage class | Replica owner | Replication Time Control | KMS-encrypted objects (SSE-KMS or DSSE-KMS) | Replica modification sync |
|-----------------------|---------|---------------------------------|-----------------------------------|----------|---------------|----------------|----------------|--------------------------|---|---------------------------|
| replicate-all | Enabled | s3://monarch-destination-bucket | US West (N. California) us-west-1 | 0 | Entire bucket | Same as source | Same as source | Disabled | Do not replicate | Disabled |

- Save your rule and confirm the replication is working by uploading a file to the source bucket and then checking to see that a copy has been created in the destination bucket.

Screenshot of the AWS S3 Upload status page for the 'monarch-source-bucket'.

Upload: status

Upload succeeded
For more information, see the [Files and folders](#) table.

Upload: status

Summary

Destination: s3://monarch-source-bucket

| Succeeded | Failed |
|---------------------------|-------------------|
| 1 file, 47.0 KB (100.00%) | 0 files, 0 B (0%) |

Files and folders [Configuration](#)

Files and folders (1 total, 47.0 KB)

| Name | Folder | Type | Size | Status | Error |
|----------------------------------|--------|------------|---------|-----------|-------|
| 706d25e7-2c52-482b-865b-1cdf2... | - | image/jpeg | 47.0 KB | Succeeded | - |

Amazon S3

General purpose buckets

Directory buckets

Table buckets

Access Grants

Access Points

Object Lambda Access Points

Multi-Region Access Points

Batch Operations

IAM Access Analyzer for S3

Block Public Access settings for this account

monarch-destination-bucket [Info](#)

Objects Properties Permissions Metrics Management Access Points

Objects (1)

Actions [Upload](#)

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

| <input type="checkbox"/> | Name | Type | Last modified | Size | Storage class |
|--------------------------|--|------|--------------------------------------|---------|---------------|
| <input type="checkbox"/> | 706d25e7-2c52-482b-865b-1cdf2dc4f326.jpg | jpg | April 20, 2025, 19:16:26 (UTC-05:00) | 47.0 KB | Standard |

As we can see the uploaded file(image), has automatically replicated in the destination bucket

EXERCISE 11.3

1. Upload to an S3 Bucket Using Transfer Acceleration
Enable S3 Transfer Acceleration on an existing bucket (substitute the name of your bucket for my-bucket-name).
\$ aws s3api put-bucket-accelerate-configuration \ --bucket my-bucket-name \ --accelerate-configuration Status=Enabled

In below screenshot we can see commands successful run

```
Last login: Sun Apr  6 08:52:24 on ttys000
[monarchnigam@Monarchs-Air ~ % aws configure
AWS Access Key ID [*****P3VQ]: AKIASK5MCXSLZ35TQ2WT
AWS Secret Access Key [*****BwC0]: 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
monarchnigam@Monarchs-Air ~ % aws s3api put-bucket-accelerate-configuration \
[ --bucket monarch-source-bucket \
  --accelerate-configuration Status=Enabled
monarchnigam@Monarchs-Air ~ % aws s3api get-bucket-accelerate-configuration \
[ --bucket monarch-source-bucket

{
    "Status": "Enabled"
}
monarchnigam@Monarchs-Air ~ %
```

2. Transfer a file by specifying the s3-accelerate endpoint. Make sure you're using the correct file and bucket names and region. \$ aws s3 cp filename.mp4 s3://my-bucket-name \ \backslash --region us-east-1 \ \backslash --endpoint-url [http://s3-accelerate.amazonaws.c](https://s3-accelerate.amazonaws.com)

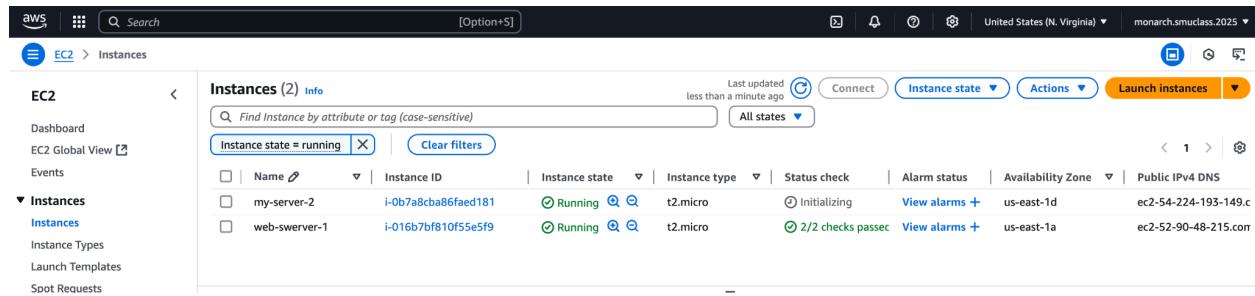
```
monarchnigam@Monarchs-Air ~ % aws s3 cp /Users/monarchnigam/Downloads/706d25e7-2c52-482b-865b-1cdf2dc4f326.jpg \
[ s3://monarch-source-bucket/ \
  --region us-east-1 \
  --endpoint-url https://s3-accelerate.amazonaws.com

upload: Downloads/706d25e7-2c52-482b-865b-1cdf2dc4f326.jpg to s3://monarch-source-bucket/706d25e7-2c52-482b-865b-1cdf2dc4f326.jpg
monarchnigam@Monarchs-Air ~ %
```

EXERCISE 11.4

Create and Deploy an EC2 Load Balancer

1. Create one instance in each of two subnets. You select the subnets on the Configure Instance Details page. Make a note of the subnet designations you choose



The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with 'EC2' selected. The main area has a heading 'Instances (2) info'. Below it is a search bar with 'Find Instance by attribute or tag (case-sensitive)' and a 'Clear filters' button. A filter 'Instance state = running' is applied. The main table lists two instances:

| | Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability Zone | Public IPv4 DNS |
|--------------------------|---------------|---------------------|--------------------------|---------------|----------------------------------|----------------------------|-------------------|----------------------|
| <input type="checkbox"/> | my-server-2 | i-0b7a8cba86faed181 | Running ⓘ ⓘ | t2.micro | Initializing ⓘ | View alarms + | us-east-1d | ec2-54-224-193-149.c |
| <input type="checkbox"/> | web-swerver-1 | i-016b7bf810f55e5f9 | Running ⓘ ⓘ | t2.micro | 2/2 checks passed ⓘ | View alarms + | us-east-1a | ec2-52-90-48-215.cor |

2. Use SSH to access your instances, install the Apache web server, and create unique `index.html` files containing a short note describing the server—perhaps including the server's IP address. This will make it easier to know which instance your load balancer has loaded when you test everything out later.

```
monarchnigam — ubuntu@ip-172-31-34-36: ~ — ssh -i ~/Downloads/monarch_keypair.pem ubuntu@52....
```

```
[ubuntu@ip-172-31-34-36:~$ sudo yum update -y
sudo: yum: command not found
[ubuntu@ip-172-31-34-36:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Fetched 126 kB in 1s (233 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
63 packages can be upgraded. Run 'apt list --upgradable' to see them.
[ubuntu@ip-172-31-34-36:~$ sudo apt install -y apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.58-1ubuntu8.6).
0 upgraded, 0 newly installed, 0 to remove and 63 not upgraded.
[ubuntu@ip-172-31-34-36:~$ sudo systemctl start apache2
[ubuntu@ip-172-31-34-36:~$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
ubuntu@ip-172-31-34-36:~$
```



This is Web Server 1 - 172.31.34.36

```
monarchnigam — ubuntu@ip-172-31-16-253: ~ — ssh -i ~/Downloads/monarch_keypair.pem ubuntu@54...
```

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
[ubuntu@ip-172-31-16-253:~$ sudo systemctl start apache2
[ubuntu@ip-172-31-16-253:~$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
[ubuntu@ip-172-31-16-253:~$ echo "This is Web Server 2 in different AZ - $(hostname -I)" | sudo tee /var/www/html/index.html
This is Web Server 2 in different AZ - 172.31.16.253
ubuntu@ip-172-31-16-253:~$
```



This is Web Server 2 in different AZ - 172.31.16.253

In the above screenshots I have shown that two EC2 instances are running in two Different AZ

3. From the EC2 Dashboard, create a load balancer. Select an application load balancer, give it a name, and specify the “Internet-facing” scheme, IPv4 for the IP address type, and the default HTTP and port 80 values for the listener. Make sure you're using the right VPC (the default should be fine) and select (and note) two availability zones.
- For this exercise, you can ignore the warning about using an unencrypted listener.

Create Application Load Balancer Info

The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and containers, based on request attributes. When the load balancer receives a connection request, it evaluates the listener rules in priority order to determine which rule to apply, and if applicable, it selects a target from the target group for the rule action.

► How Application Load Balancers work

Basic configuration

Load balancer name
Name must be unique within your AWS account and can't be changed after the load balancer is created.
 A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme Info
Scheme can't be changed after the load balancer is created.

Internet-facing
• Serves internet-facing traffic.
• Has public IP addresses.
• DNS name resolves to public IPs.
• Requires a public subnet.

Internal
• Serves internal traffic.
• Has private IP addresses.
• DNS name resolves to private IPs.
• Compatible with the **IPv4** and **Dualstack** IP address types.

Load balancer IP address type Info
Select the front-end IP address type to assign to the load balancer. The VPC and subnets mapped to this load balancer must include the selected IP address types. Public IPv4 addresses have an additional cost.

IPv4
Includes only IPv4 addresses.

Dualstack
Includes IPv4 and IPv6 addresses.

Dualstack without public IPv4
Includes a public IPv6 address, and private IPv4 and IPv6 addresses. Compatible with **internet-facing** load balancers only.

- Select (or create) a security group that allows access to port 80.

The screenshot shows the AWS CloudFormation Create Stack Wizard Step 3: Set Configuration. The 'Template' tab is selected. The template body contains the following CloudFormation snippet:

```

Resources:
  MyLambdaFunction:
    Type: AWS::Lambda::Function
    Properties:
      Handler: index.handler
      Runtime: nodejs12.x
      Role: !GetAtt LambdaExecutionRole.Arn
      Code:
        ZipFile:
          Fn::Base64: !File "lambda_function.zip"
      Timeout: 10
      MemorySize: 128
      Environment:
        Variables:
          API_ID: !Ref MyAPIGateway
  MyAPIGateway:
    Type: AWS::Serverless::Api
    Properties:
      StageName: dev
      Auth:
        Default:
          Type: None
      Domains:
        Items:
          - DomainName: myapi.dev
            Route: "/{proxy+}"
            HostnameType: IGD
            CertificateArn: !GetAtt MyLambdaFunction.CertificateArn
      TracingEnabled: true
  MyVPC:
    Type: AWS::VPC::InterfaceEndpoint
    Properties:
      ServiceName: com.amazonaws.us-east-1.execute-api
      VpcId: !Ref MyVPC
      SubnetIds:
        - !Ref MySubnet
      EndpointType: Interface
      VpcEndpointType: Interface
      VpcPeeringConnectionId: !Ref MyVPCPeeringConnection
  MyVPCPeeringConnection:
    Type: AWS::VPC::PeeringConnection
    Properties:
      PeerVpcId: !Ref MyPeerVPC
      PeerOwnerId: !Ref MyPeerOwner
      PeerRegion: us-east-1
      PeerSubnetIds:
        - !Ref MyPeerSubnet
      VpcId: !Ref MyVPC
  MyPeerVPC:
    Type: AWS::CloudFormation::Interface
    Properties:
      Parameters:
        MyPeerSubnet:
          Type: String
        MyPeerOwner:
          Type: String
  MyPeerSubnet:
    Type: AWS::VPC::Subnet
    Properties:
      VpcId: !Ref MyVPC
      CidrBlock: 10.0.1.0/24
      AvailabilityZone: !Ref MyAvailabilityZone
  MyPeerOwner:
    Type: AWS::CloudFormation::Interface
    Properties:
      Parameters:
        MyPeerSubnet:
          Type: String
  MyAvailabilityZone:
    Type: AWS::CloudFormation::Interface
    Properties:
      Parameters:
        MyPeerSubnet:
          Type: String

```

The outputs section shows the ARN of the Lambda function:

```

Outputs:
  MyLambdaFunctionArn:
    Description: The ARN of the Lambda function
    Value: !GetAtt MyLambdaFunction.Arn

```

- Click the Create Target Group link and change the Target Type value to IP and add a path to a file the health check can use. For this exercise, enter `/index.html`.

demotargetgroup

Details

arn:aws:elasticloadbalancing:us-east-1:160885292183:targetgroup/demotargetgroup/a64a962a2302aca4

| Target type | Protocol : Port | Protocol version | VPC |
|-----------------|--|------------------|-----------------------|
| Instance | HTTP: 80 | HTTP1 | vpc-0e92ca1c7be9b22ad |
| IP address type | Load balancer monarch-app-load-balancer | | |
| IPv4 | | | |

2 Total targets | 0 Healthy | 0 Unhealthy | 0 Anomalous | 0 Unused | 2 Initial | 0 Draining

Distribution of targets by Availability Zone (AZ)

Select values in this table to see corresponding filters applied to the Registered targets table below.

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (2) Info

Anomaly mitigation: Not applicable | Deregister | Register targets

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

| Filter targets | Instance ID | Name | Port | Zone | Health status | Health status details | Admini... | Overri... | Last... |
|--------------------------|---------------------|----------------|------|--------------------|----------------------|-----------------------------|-----------------------------------|------------------------------------|---------|
| <input type="checkbox"/> | i-0b7a8cba86faed181 | my-server-2 | 80 | us-east-1d (us...) | Initial | Target registration is i... | <input type="radio"/> No override | <input type="radio"/> No overri... | A |
| <input type="checkbox"/> | i-01c17lkf010ffccf0 | web_instance_1 | 80 | us-east-1a (us...) | Initial | Target registration is i... | <input type="radio"/> No override | <input type="radio"/> No overri... | A |

- On the Register Targets page, register your two EC2 instances as targets to the group using their private IPs. You can retrieve those private IP addresses from the EC2 Instances console or from the command prompt of the instance itself using the `ip addr` command. - Depicted in the above screenshot
- Once your balancer is active (which can take a couple of minutes), get the DNS name of your balancer from the Load Balancers console and paste it into your browser. The `index.html` page of one of your instances should load. Refresh the browser a few times so that you can confirm that both instances are being accessed.

http://monarch-app-load-balancer-418774742.us-east-1.elb.amazonaws.com/

This is Web Server 2 in different AZ - 172.31.16.253

http://monarch-app-load-balancer-418774742.us-east-1.elb.amazonaws.com/

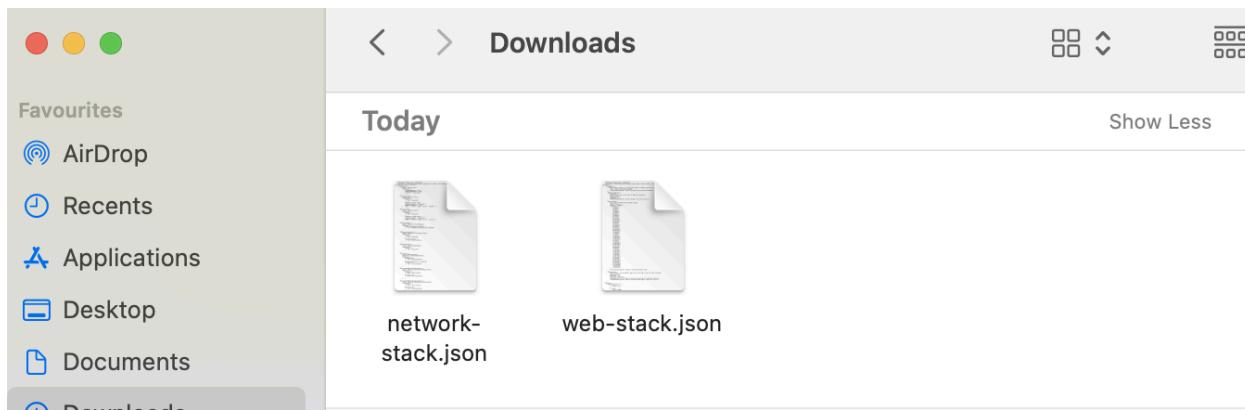
This is Web Server 1 - 172.31.34.36

EXERCISE 11.5

Create a Nested Stack

In this exercise, you'll create a nested stack that creates an EC2 Auto Scaling group and the supporting network infrastructure. To complete this exercise, you'll need to know the name of a Secure Shell (SSH) key pair in the region you're using.

1. Visit <http://awscsa.github.io> and click Chapter 14. Download the `web-stack.json` and `network-stack.json` CloudFormation templates.

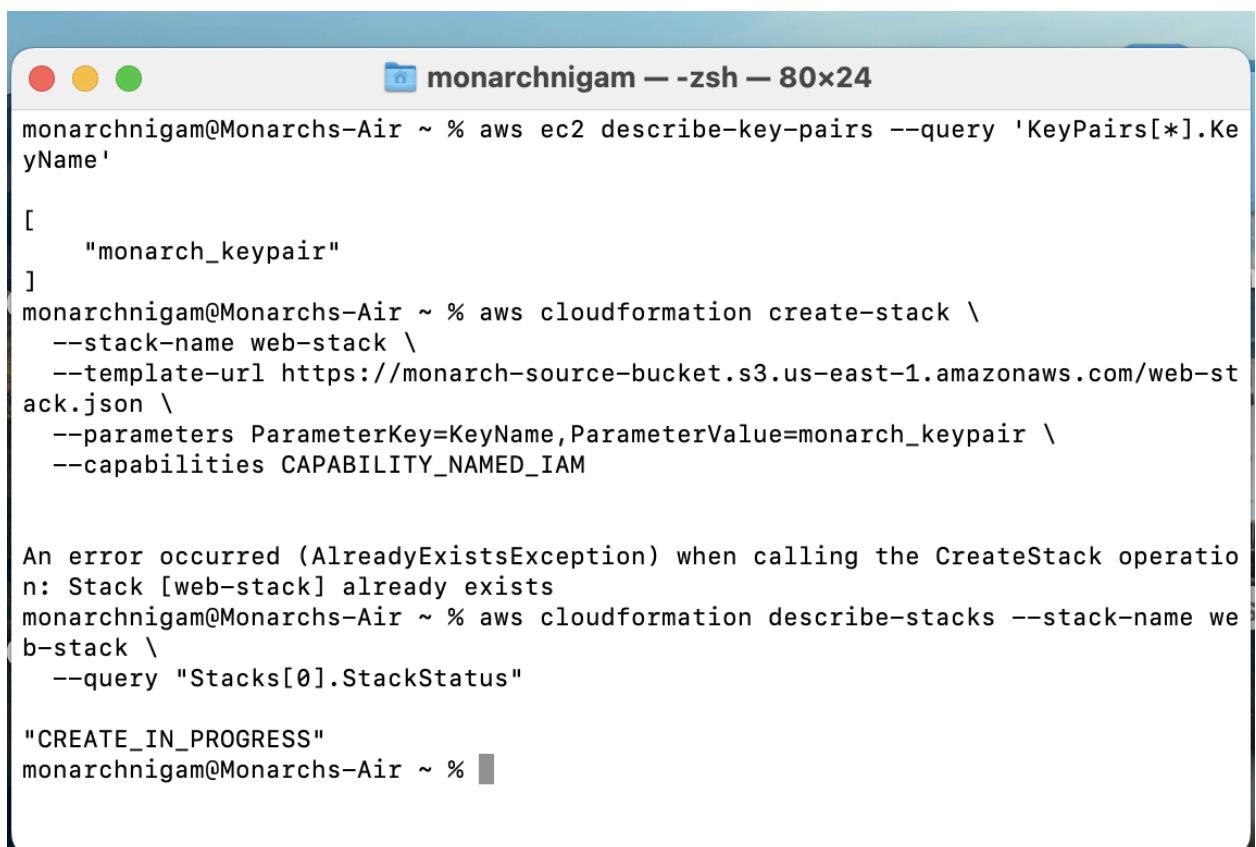


2. Create an S3 bucket and upload the `network-stack.json` template to the bucket.
3. Edit the `web-stack.json` template and locate the `Network-Stack` resource. Change the `TemplateURL` value to the URL of the `network-stack.json` template in S3.

```
5     },
6   },
7   "Resources": [
8     "NetworkStack" : {
9       "Type" : "AWS::CloudFormation::Stack",
10      "Properties" : {
11        "TemplateURL" : "https://monarch-source-bucket.s3.us-east-1.amazonaws.com/network-stack.json"
12      }
13    }
14  ],
15 }
```

4. Upload the `web-stack.json` template to the same S3 bucket you used in step 2.
5. CloudFormation will begin to create the `Web` stack and the nested network stack.

Enter the following command to view the stacks:



```
monarchnigam@Monarchs-Air ~ % aws ec2 describe-key-pairs --query 'KeyPairs[*].Ke
yName'

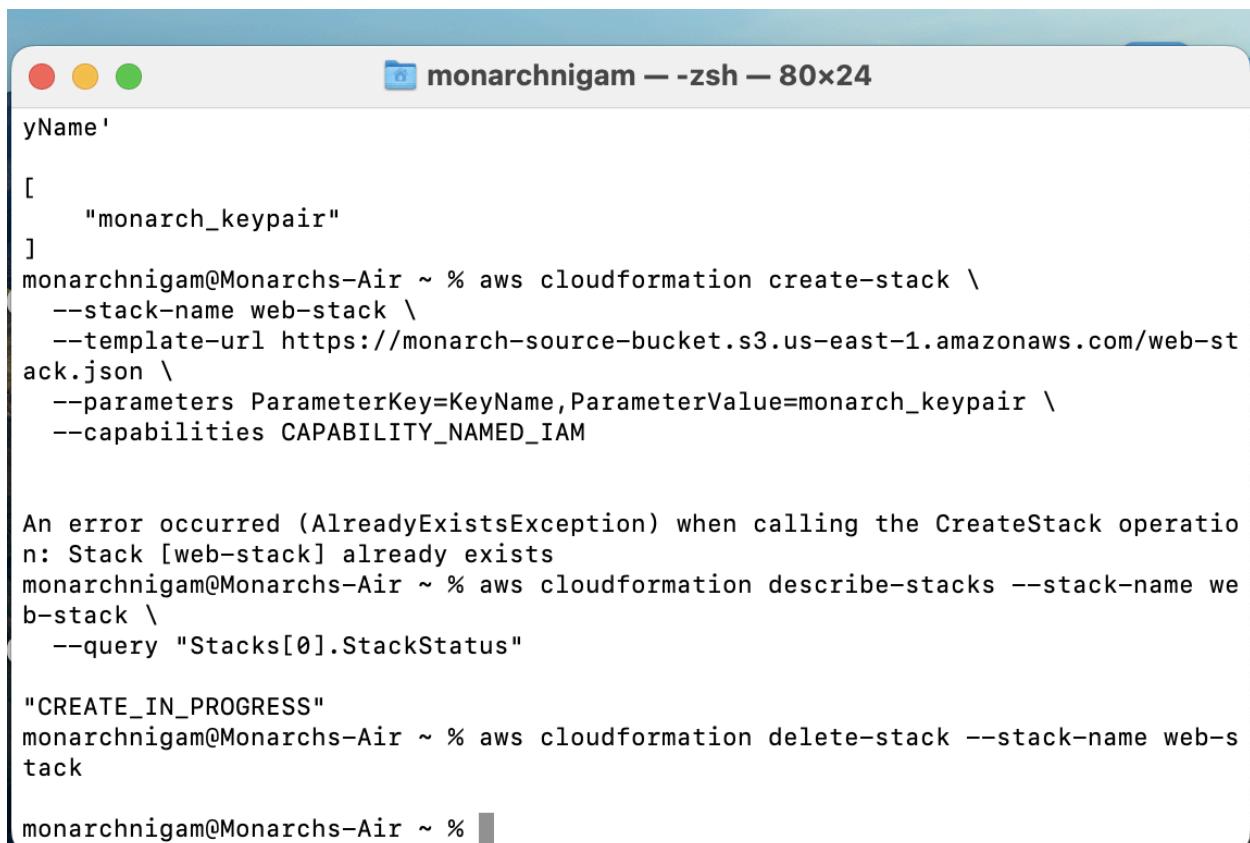
[
  "monarch_keypair"
]
monarchnigam@Monarchs-Air ~ % aws cloudformation create-stack \
--stack-name web-stack \
--template-url https://monarch-source-bucket.s3.us-east-1.amazonaws.com/web-st
ack.json \
--parameters ParameterKey=KeyName,ParameterValue=monarch_keypair \
--capabilities CAPABILITY_NAMED_IAM

An error occurred (AlreadyExistsException) when calling the CreateStack operatio
n: Stack [web-stack] already exists
monarchnigam@Monarchs-Air ~ % aws cloudformation describe-stacks --stack-name we
b-stack \
--query "Stacks[0].StackStatus"

"CREATE_IN_PROGRESS"
monarchnigam@Monarchs-Air ~ %
```

6. CloudFormation will begin to create the Web stack and the nested network stack.

Enter the following command to view the stacks:



```
yName'

[
    "monarch_keypair"
]
monarchnigam@Monarchs-Air ~ % aws cloudformation create-stack \
--stack-name web-stack \
--template-url https://monarch-source-bucket.s3.us-east-1.amazonaws.com/web-st
ack.json \
--parameters ParameterKey=KeyName,ParameterValue=monarch_keypair \
--capabilities CAPABILITY_NAMED_IAM

An error occurred (AlreadyExistsException) when calling the CreateStack operatio
n: Stack [web-stack] already exists
monarchnigam@Monarchs-Air ~ % aws cloudformation describe-stacks --stack-name we
b-stack \
--query "Stacks[0].StackStatus"

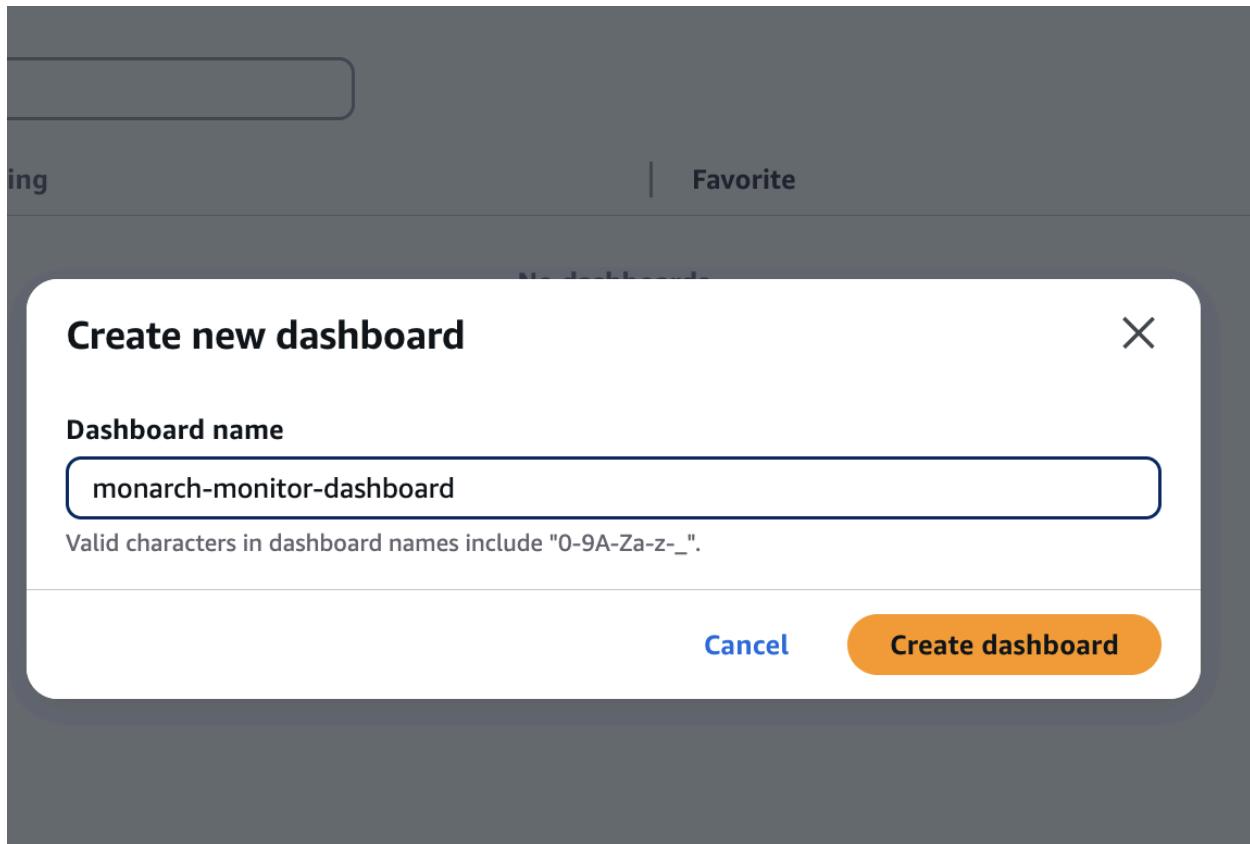
"CREATE_IN_PROGRESS"
monarchnigam@Monarchs-Air ~ % aws cloudformation delete-stack --stack-name web-s
tack

monarchnigam@Monarchs-Air ~ %
```

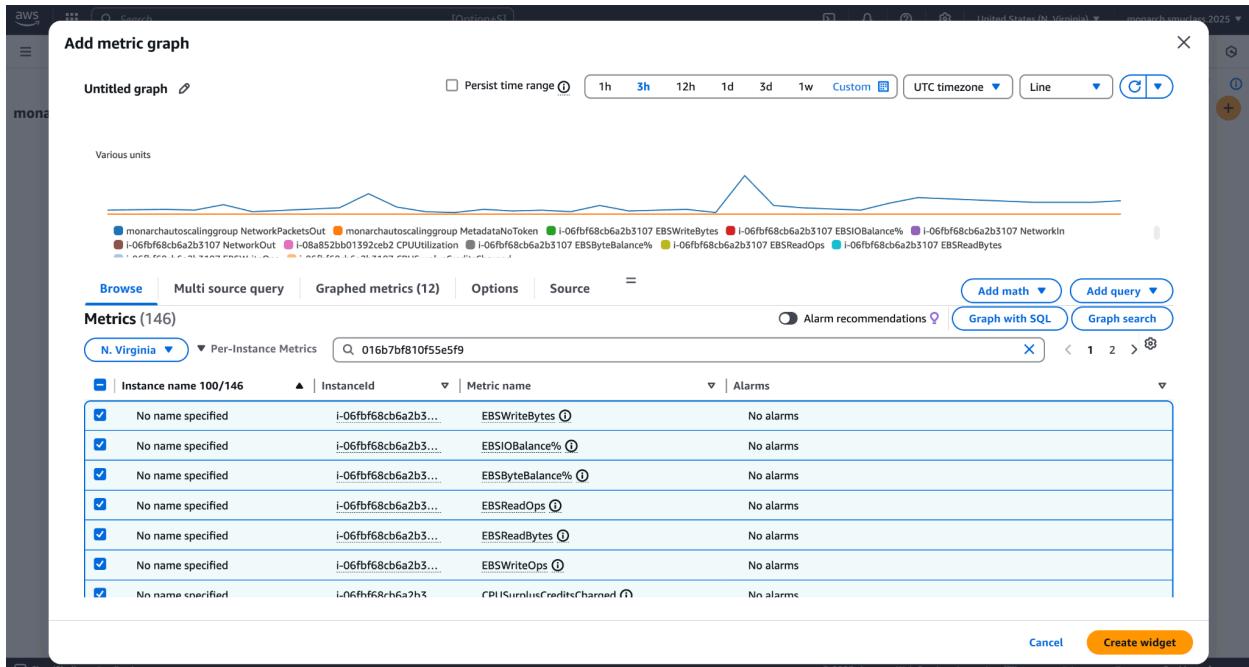
EXERCISE 11.6

Create a CloudWatch Dashboard

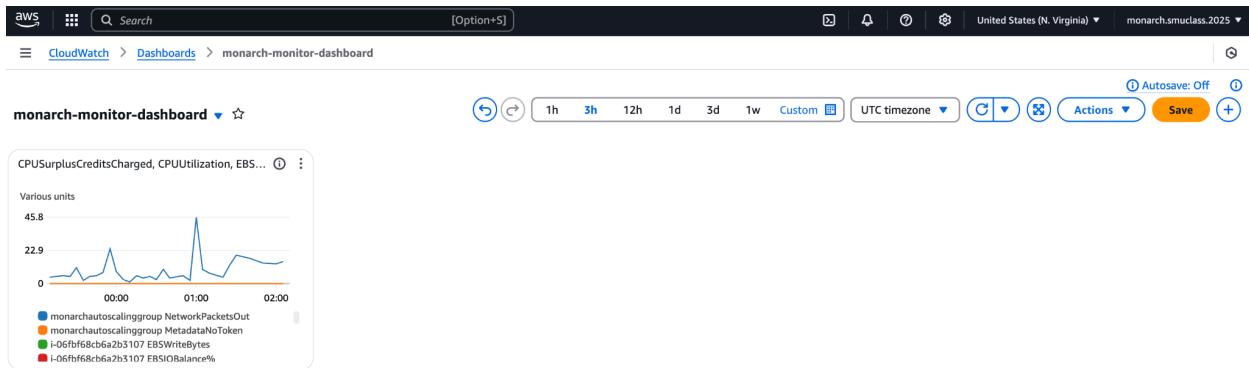
1. So that you'll get a better idea of how things will look, make sure you have at least one EC2 instance running and at least one S3 bucket with some files in it.
2. From the CloudWatch console, click Dashboards, click Create Dashboard, and give your dashboard a name. On the Add Widget page, select the Line metric. Then select the Metrics option.



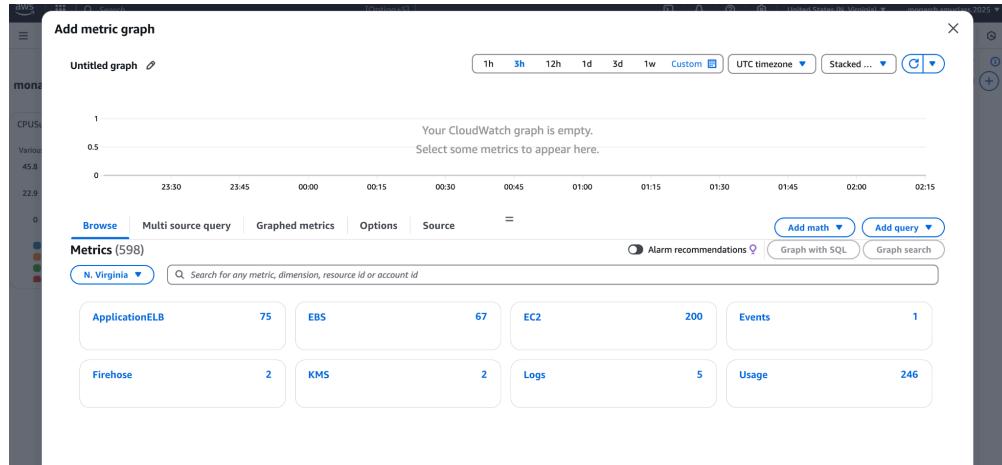
3. From the Metrics displayed in the bottom half of the screen, select the appropriate AWS region and click EC2 and then click Per-Instance Metrics. Check the boxes for a few metrics associated with your running instance. StatusCheckFailed, NetworkOut, NetworkIn, and CPUUtilization would make for a nice sample. Note how those metrics are now represented in the graph at the top of the screen. When you're satisfied with your selections, click Create Widget.



4. Your first widget will now appear in your dashboard. You can click the widget to display it full-screen. Selecting a particular metric from the legend at the bottom of the graph will show you only that metric, making it easier to understand what the values represent.



- If your widget is still full-screen, click Close to return to the dashboard.
- Click Add Widget to create a second widget. This time, select Stacked Area and then Storage Metrics. Select S3 metrics and then the NumberOfObjects and BucketSizeBytes metrics for your bucket. Create the widget.



In the above screenshot we can't find S3 although the S3 bucket is created with the stack files uploaded in the last exercise. Hence we will create another dashboard with services other than S3, which are visible in the above screenshot

- Once you've added all the widgets and metrics you'll need to monitor, click Save Dashboard.

