Phase 1

0. Set Up

- I created an AWS account, enabled Multi-Factor Authentication (MFA), and set up an IAM user with AmazonEC2FullAccess permissions. I then saved the credentials securely for later use.
- Operating System: Linux (Ubuntu on WSL)

1. AWS CLI Configuration

- Command:

```
aws configure
```

  - Ran aws configure using the access key and secret key from my IAM user.
  - Set Default region: us-east-1
  - Set Output format: json
- Verified installation with:

```
aws --version
aws ec2 describe-regions
```

- The output shows:

```
aws-cli/2.31.3 Python/3.13.7 Linux/6.6.87.2-microsoft-standard-WSL2
exe/x86_64.ubuntu.24
```

- Followed by a list of AWS regions with OptInStatus, RegionName, and Endpoint.

2. Find Latest Amazon Linux AMI

- Command (Located recent Amazon Linux AMI):

```
aws ec2 describe-images \
   --owners amazon \
   --filters "Name=name,Values=al2023-ami-*"
"Name=architecture,Values=x86_64" \
   --query 'sort_by(Images, &CreationDate)[-1].[ImageId]' \
   --output text
```

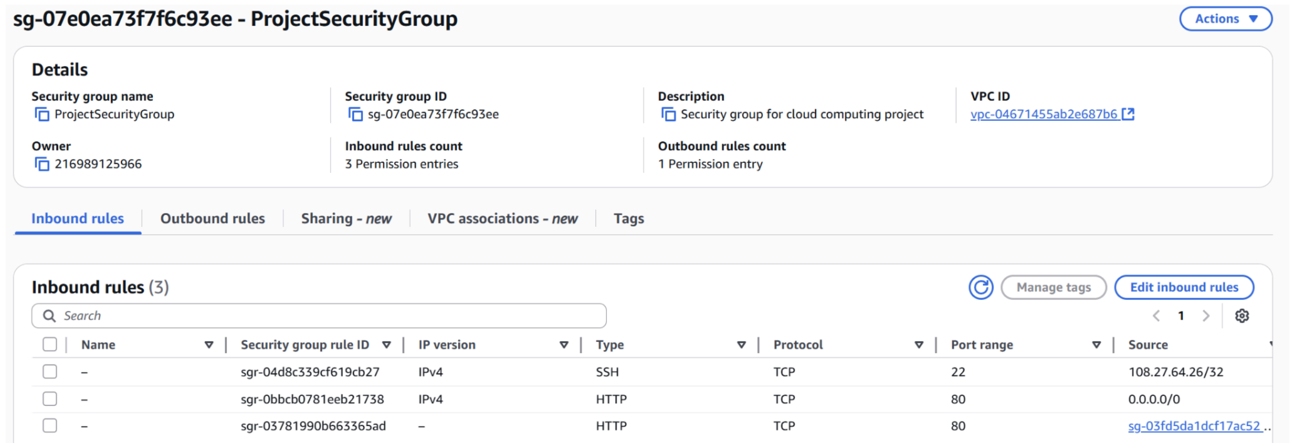
- Returned the AMI ID: ami-0b9755dd9758b73ce

3. Add Security groups

- Command (Create security group with basic setting):

```
aws ec2 create-security-group --group-name ProjectSecurityGroup \
                --description "Security group for cloud computing project"
        aws ec2 authorize-security-group-ingress \
                --group-name ProjectSecurityGroup --protocol tcp --port 22 \
                --cidr <your_ip>/32
        aws ec2 authorize-security-group-ingress \
                --group-name ProjectSecurityGroup --protocol tcp --port 80 \
                --cidr 0.0.0.0/0
```

Output: Security group created with ID: sg-07e0ea73f7f6c93ee.



sg-07e0ea73f7f6c93ee - ProjectSecurityGroup

Details

Security group name	ProjectSecurityGroup
Security group ID	sg-07e0ea73f7f6c93ee
Description	Security group for cloud computing project
VPC ID	vpc-04671455ab2e687b6
Owner	216989125966
Inbound rules count	3 Permission entries
Outbound rules count	1 Permission entry

Inbound rules (3)

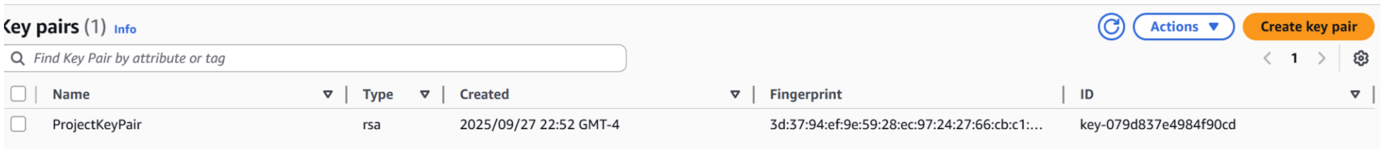
Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
–	sgr-04d8c339cf619cb27	IPv4	SSH	TCP	22	108.27.64.26/32
–	sgr-0bbcb0781eeb21738	IPv4	HTTP	TCP	80	0.0.0.0/0
–	sgr-03781990b663365ad	–	HTTP	TCP	80	sg-03fd5da1dcf17ac52 ..

4. **Created SSH Key Pair & Set Permissions on Key File**

- Command (Create a new key pair and save it locally with restricted permissions):

```
        aws ec2 create-key-pair \
           --key-name ProjectKeyPair \
           --query 'KeyMaterial' \
           --output text > ProjectKeyPair.pem

        chmod 400 ProjectKeyPair.pem
```

Result: A key pair named ProjectKeyPair was created in the EC2 Console.



Key pairs (1)

Name	Type	Created	Fingerprint	ID
ProjectKeyPair	rsa	2025/09/27 22:52 GMT-4	3d:37:94:ef:9e:59:28:ec:97:24:27:66:cb:c1:...	key-079d837e4984f90cd

5. **Created bootstrap.sh file and Launched EC2 Instance:**

- Created a file name "bootstrap.sh" in local project directory and fill in:

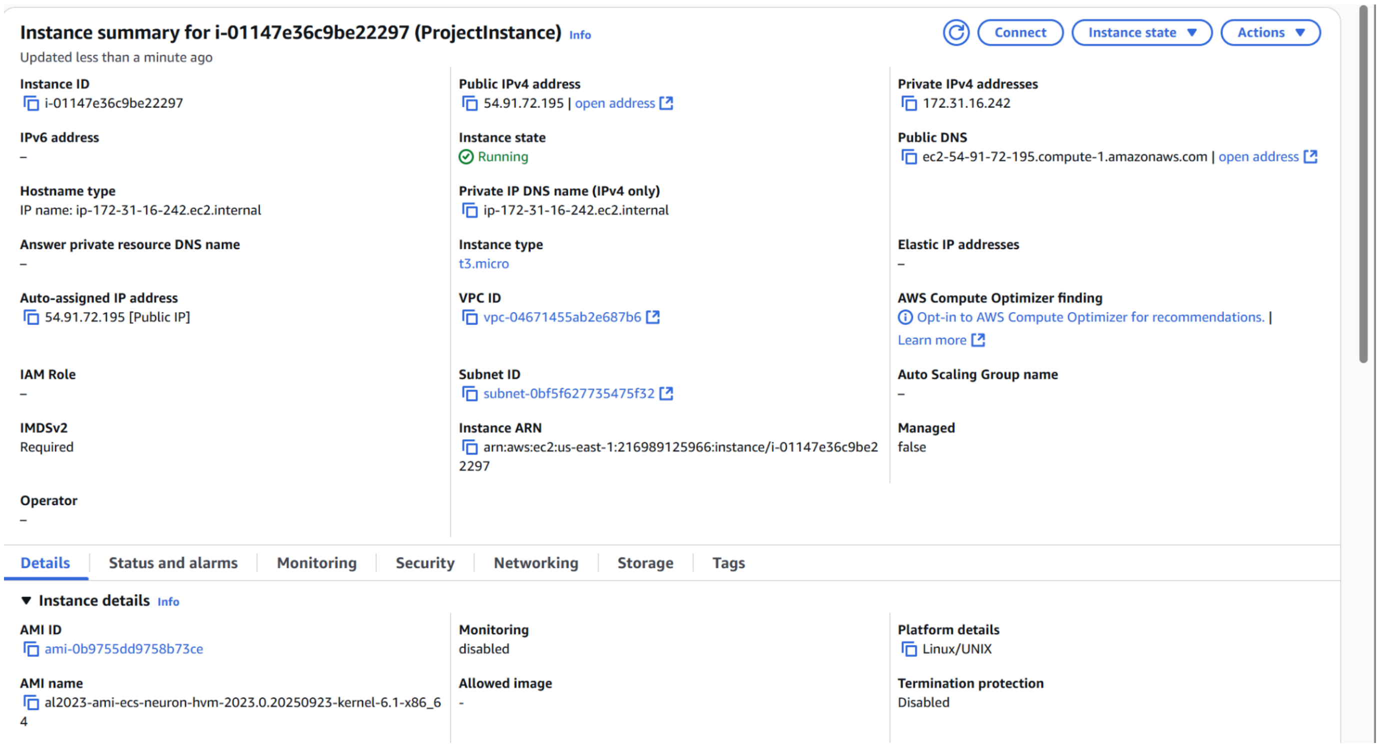
```
        #!/bin/bash
        yum update -y
        yum install -y httpd git python3
```

```
systemctl start httpd
systemctl enable httpd
```

- Command (Launch a new EC2 instance):

```
aws ec2 run-instances \
  --image-id ami-0b9755dd9758b73ce \
  --instance-type t3.micro \
  --key-name ProjectKeyPair \
  --security-group-ids sg-07e0ea73f7f6c93ee \

  --tag-specifications
'ResourceType=instance,Tags=[{Key=Name,Value=ProjectInstance}]' \
  --user-data file://bootstrap.sh
```

Output: An EC2 instance named ProjectInstance was created in the console.
Instance ID: i-01147e36c9be22297



Instance summary for i-01147e36c9be22297 (ProjectInstance)
Updated less than a minute ago

Instance ID: i-01147e36c9be22297
IPv6 address: –
Hostname type: IP name: ip-172-31-16-242.ec2.internal
Answer private resource DNS name: –
Auto-assigned IP address: 54.91.72.195 [Public IP]
IAM Role: –
IMDSv2: Required
Operator: –

Public IPv4 address: 54.91.72.195 | open address
Instance state: Running
Private IP DNS name (IPv4 only): ip-172-31-16-242.ec2.internal
Instance type: t3.micro
VPC ID: vpc-04671455ab2e687b6
Subnet ID: subnet-0bf5f627735475f32
Instance ARN: arn:aws:ec2:us-east-1:216989125966:instance/i-01147e36c9be22297

Private IPv4 addresses: 172.31.16.242
Public DNS: ec2-54-91-72-195.compute-1.amazonaws.com | open address
Elastic IP addresses: –
AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendations. | Learn more
Auto Scaling Group name: –
Managed: false

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

▼ Instance details Info

AMI ID: ami-0b9755dd9758b73ce
AMI name: al2023-ami-ecs-neuron-hvm-2023.0.20250923-kernel-6.1-x86_64
Monitoring: disabled
Allowed image: -
Platform details: Linux/UNIX
Termination protection: Disabled

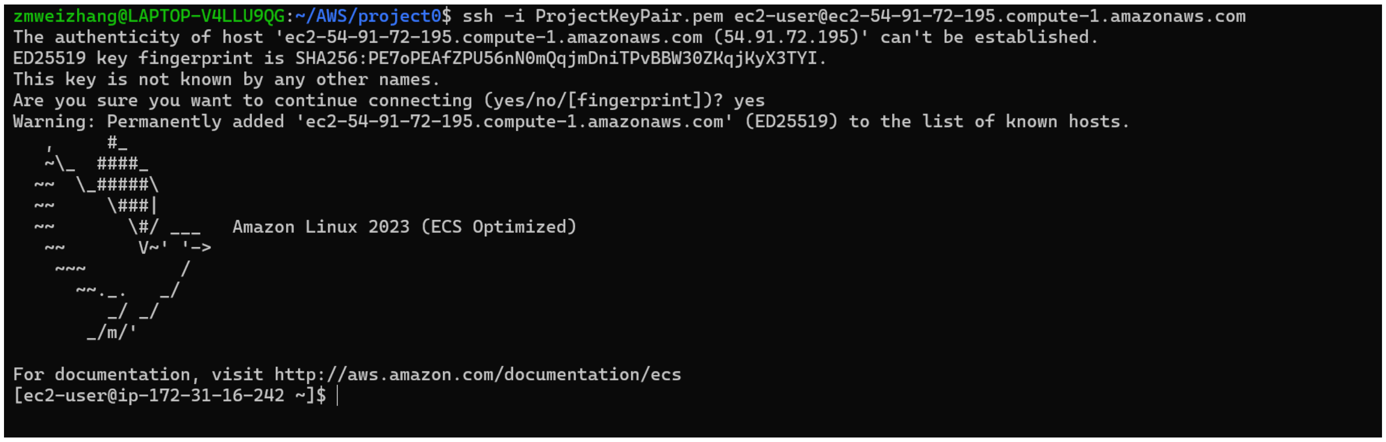
6. Connected EC2 Instance using SSH

- Command (Retrieve the public DNS name of the instance):

```
aws ec2 describe-instances \
   --filters "Name=tag:Name,Values=ProjectInstance" \
   --query 'Reservations[].Instances[].PublicDnsName' \
   --output text
```

Output: Provided instance DNS name: ec2-54-91-72-195.compute-1.amazonaws.com

- Command (Connect to the EC2 instance using the key pair and public DNS name):
  ssh -i ProjectKeyPair.pem ec2-user@ec2-54-91-72-195.compute-1.amazonaws.com
  Output: Showed the user logged into the EC2 instance, confirmed by the Amazon Linux 2023 welcome banner.



```
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$ ssh -i ProjectKeyPair.pem ec2-user@ec2-54-91-72-195.compute-1.amazonaws.com
The authenticity of host 'ec2-54-91-72-195.compute-1.amazonaws.com (54.91.72.195)' can't be established.
ED25519 key fingerprint is SHA256:PE7oPEAfZPU56nN0mQqjmDniTPvBBW30ZKqjKyX3TYI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-91-72-195.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   Amazon Linux 2023 (ECS Optimized)
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'

For documentation, visit http://aws.amazon.com/documentation/ecs
[ec2-user@ip-172-31-16-242 ~]$ |
```

7. **Configured Instances**

- Command (Update packages and install Apache, Git, Python3, and pip):

  ```
  sudo yum update -y
  sudo yum install -y httpd git python3 python3-pip
  ```

  Output: Displayed libxcrypt-compat-4.4.33-7.amzn2023.x86_64 and python3-pip-21.3.1-2.amzn2023.0.13.noarch are installed
- Command (Start and enable the Apache web server):

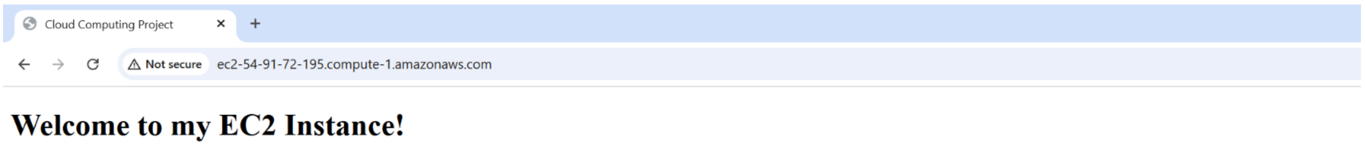
  ```
  sudo systemctl start httpd
  sudo systemctl enable httpd
  ```

- Then wrote:

  ```
  cat << 'EOF' | sudo tee /var/www/html/index.html
  <!DOCTYPE html>
  <html>
  <head>
     <title>Cloud Computing Project</title>
  </head>
  <body>
     <h1>Welcome to my EC2 Instance!</h1>
  </body>
  </html>
  EOF
  ```

  Result: The EC2 instance was configured as a web server, and the test page displayed “Welcome to my EC2 Instance!” when accessed through its public DNS in a browser.

The webpage can be seen through http://ec2-54-91-72-195.compute-1.amazonaws.com. The url was based on instance DNS name.



Cloud Computing Project

Not secure ec2-54-91-72-195.compute-1.amazonaws.com

Welcome to my EC2 Instance!

8. Created Customized AMI

- Command (Create a custom AMI from the running EC2 instance):

```
aws ec2 create-image \
    --instance-id i-01147e36c9be22297 \
    --name "ProjectWebServerAMI" \
    --description "Web server image for cloud computing project" \
    --no-reboot
```

Output: Displayed ImageId: ami-0e89041b5de66d237

- Command (Check the state of the new AMI):

```
aws ec2 describe-images \
   --image-ids ami-0e89041b5de66d237 \
   --query 'Images[].State'
```

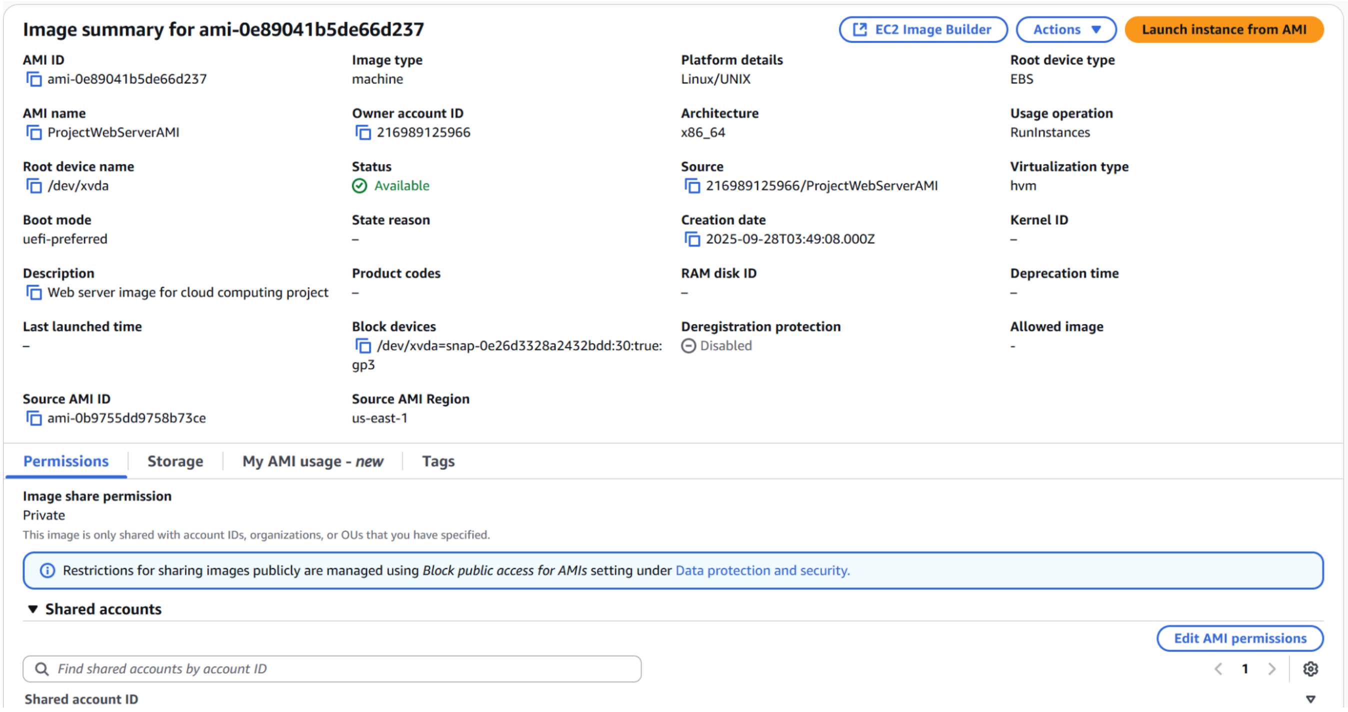


Image summary for ami-0e89041b5de66d237

AMI ID ami-0e89041b5de66d237	**Image type** machine	**Platform details** Linux/UNIX	**Root device type** EBS
AMI name ProjectWebServerAMI	**Owner account ID** 216989125966	**Architecture** x86_64	**Usage operation** RunInstances
Root device name /dev/xvda	**Status** Available	**Source** 216989125966/ProjectWebServerAMI	**Virtualization type** hvm
Boot mode uefi-preferred	**State reason** –	**Creation date** 2025-09-28T03:49:08.000Z	**Kernel ID** –
Description Web server image for cloud computing project	**Product codes** –	**RAM disk ID** –	**Deprecation time** –
Last launched time –	**Block devices** /dev/xvda=snap-0e26d3328a2432bdd:30:true:gp3	**Deregistration protection** Disabled	**Allowed image** -
Source AMI ID ami-0b9755dd9758b73ce	**Source AMI Region** us-east-1		

Image share permission
Private
This image is only shared with account IDs, organizations, or OUs that you have specified.

Restrictions for sharing images publicly are managed using *Block public access for AMIs* setting under Data protection and security.

Shared accounts

Shared account ID

Phase 2

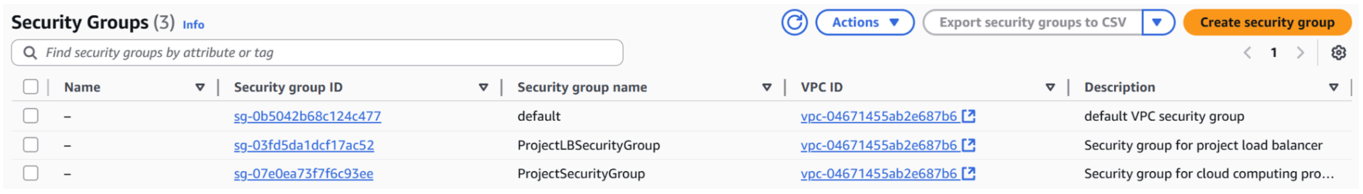
9. **Set Up Network Infrastructure**

- Command (Configure security groups for the load balancer and instances):

```
aws ec2 authorize-security-group-ingress \
   --group-name ProjectSecurityGroup \
   --protocol tcp --port 80 \
   --source-group ProjectLBSecurityGroup

aws ec2 create-security-group \
   --group-name ProjectLBSecurityGroup \
   --description "Security group for project load balancer"

aws ec2 authorize-security-group-ingress \
   --group-name ProjectLBSecurityGroup \
   --protocol tcp --port 80 \
   --cidr 0.0.0.0/0
```

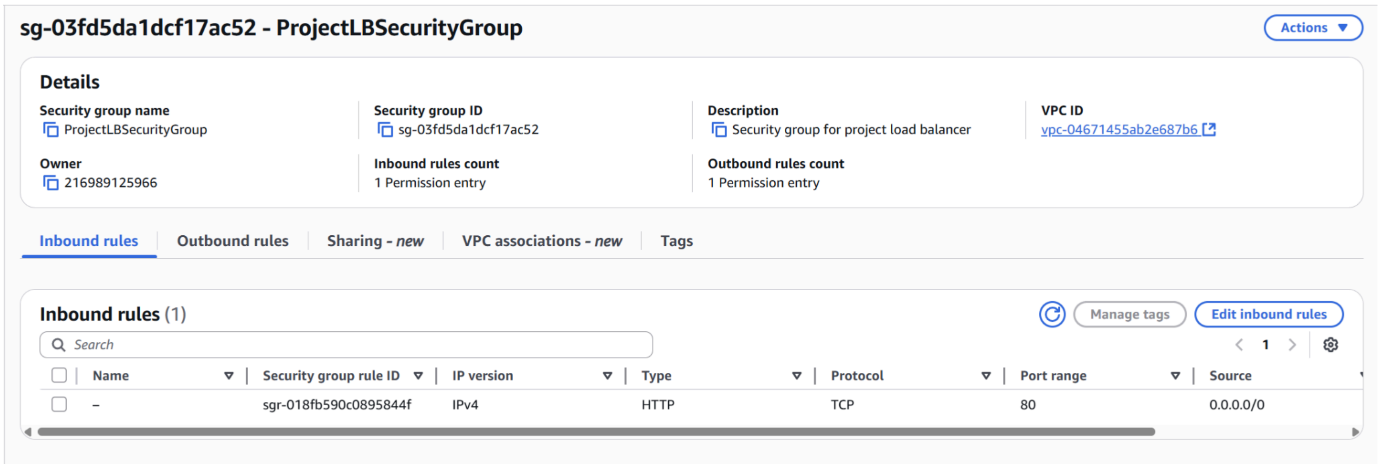
Result:

- Created additional security group rules to allow traffic from the load balancer to reach the EC2 instances.
- Created a new security group for the load balancer.
- Allowed incoming HTTP traffic to the load balancer.



Security Groups (3) Info

Name	Security group ID	Security group name	VPC ID	Description
–	sg-0b5042b68c124c477	default	vpc-04671455ab2e687b6	default VPC security group
–	sg-03fd5da1dcf17ac52	ProjectLBSecurityGroup	vpc-04671455ab2e687b6	Security group for project load balancer
–	sg-07e0ea73f7f6c93ee	ProjectSecurityGroup	vpc-04671455ab2e687b6	Security group for cloud computing pro…



sg-03fd5da1dcf17ac52 - ProjectLBSecurityGroup

Details

Security group name: ProjectLBSecurityGroup	Security group ID: sg-03fd5da1dcf17ac52	Description: Security group for project load balancer	VPC ID: vpc-04671455ab2e687b6
Owner: 216989125966	Inbound rules count: 1 Permission entry	Outbound rules count: 1 Permission entry	

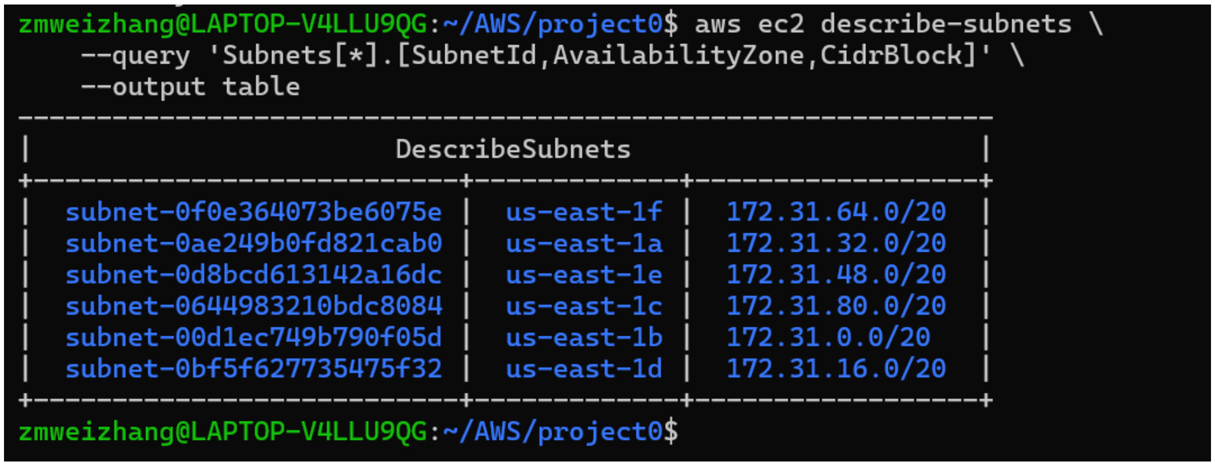
Inbound rules (1)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
–	sgr-018fb590c0895844f	IPv4	HTTP	TCP	80	0.0.0.0/0

- Command (To list available subnets with IDs, Availability Zones, and CIDR blocks):

```
aws ec2 describe-subnets \
  --query 'Subnets[*].[SubnetId,AvailabilityZone,CidrBlock]' \
  --output table
```

  Output: A table of subnets and their IDs was listed.



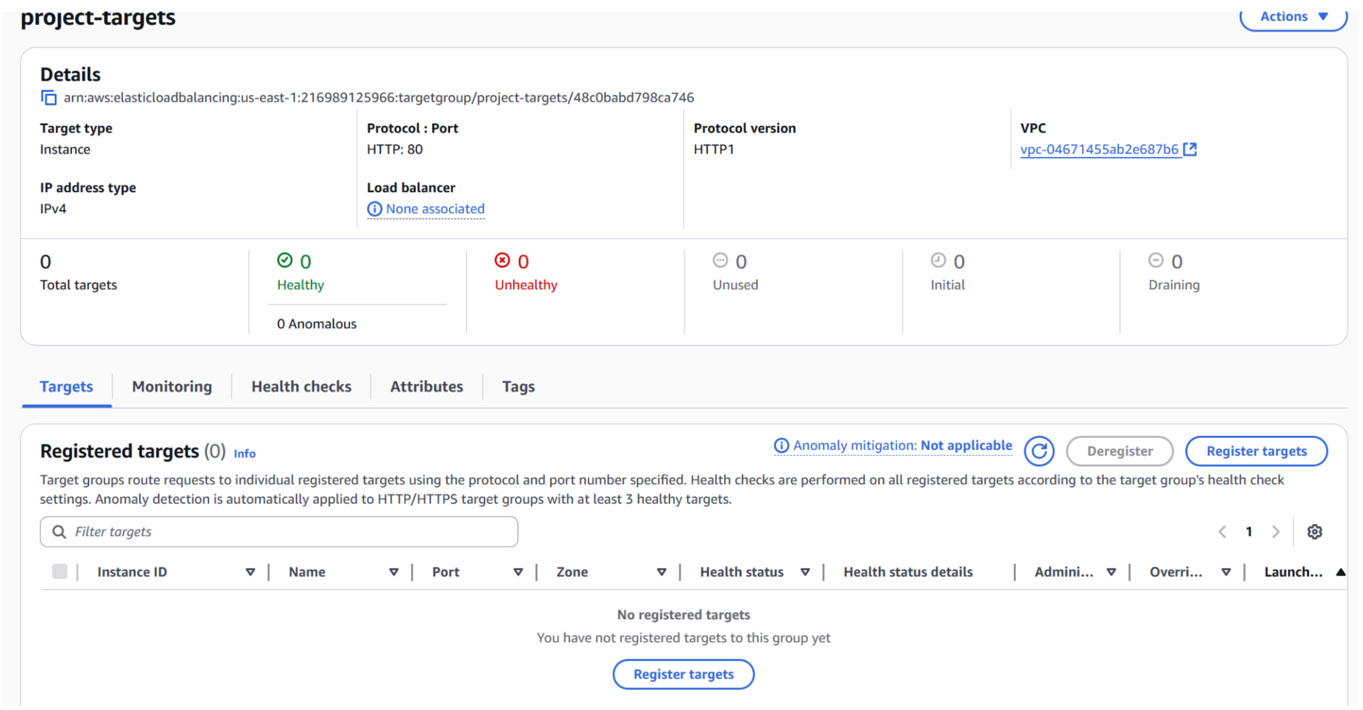
```
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$ aws ec2 describe-subnets \
    --query 'Subnets[*].[SubnetId,AvailabilityZone,CidrBlock]' \
    --output table
--------------------------------------------------------------
|                      DescribeSubnets                       |
+----------------------------+-------------+-----------------+
|  subnet-0f0e364073be6075e  |  us-east-1f |  172.31.64.0/20 |
|  subnet-0ae249b0fd821cab0  |  us-east-1a |  172.31.32.0/20 |
|  subnet-0d8bcd613142a16dc  |  us-east-1e |  172.31.48.0/20 |
|  subnet-0644983210bdc8084  |  us-east-1c |  172.31.80.0/20 |
|  subnet-00d1ec749b790f05d  |  us-east-1b |  172.31.0.0/20  |
|  subnet-0bf5f627735475f32  |  us-east-1d |  172.31.16.0/20 |
+----------------------------+-------------+-----------------+
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$
```

10. Load Balancer Setup

- Shown in AWS, default VPC: vpc-04671455ab2e687b6
- Command (Create a target group to define where the load balancer directs traffic):

```
aws elbv2 create-target-group \
  --name project-targets \
  --protocol HTTP \
  --port 80 \
  --vpc-id vpc-04671455ab2e687b6 \
  --health-check-protocol HTTP \
  --health-check-path "/index.html" \
  --health-check-interval-seconds 30 \
  --health-check-timeout-seconds 5 \
  --healthy-threshold-count 2 \
  --unhealthy-threshold-count 2 \
  --target-type instance
```

  Ouput arn:
  arn:aws:elasticloadbalancing:us-east-1:216989125966:targetgroup/project-targets/48c0babd798ca746



project-targets

Actions ▼

Details

arn:aws:elasticloadbalancing:us-east-1:216989125966:targetgroup/project-targets/48c0babd798ca746

Target type	Protocol : Port	Protocol version	VPC
Instance	HTTP: 80	HTTP1	vpc-04671455ab2e687b6
IP address type	**Load balancer**		
IPv4	None associated		

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

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (0) 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...	Launch...

No registered targets

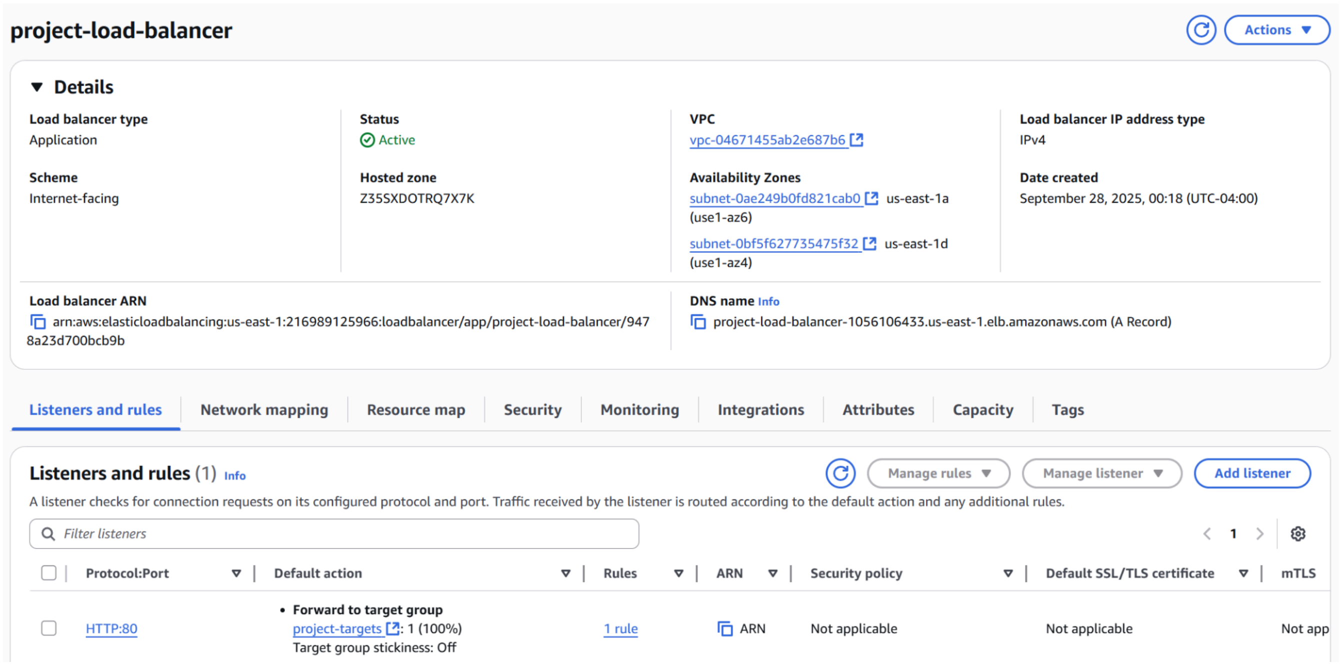
You have not registered targets to this group yet

Register targets

- Command (Create an Application Load Balancer across two subnets):

```
        aws elbv2 create-load-balancer \
          --name project-load-balancer \
          --subnets subnet-0ae249b0fd821cab0 subnet-0bf5f627735475f32 \
          --security-groups sg-03fd5da1dcf17ac52 \
          --scheme internet-facing \
          --type application \
          --tags Key=Project,Value=CloudComputing
```

  Ouput arn:
  arn:aws:elasticloadbalancing:us-east-1:216989125966:loadbalancer/app/project-load-balancer/9478a23d700bcb9b \



project-load-balancer

Actions

Details

Load balancer type Application	**Status** Active	**VPC** vpc-04671455ab2e687b6	**Load balancer IP address type** IPv4
Scheme Internet-facing	**Hosted zone** Z35SXDOTRQ7X7K	**Availability Zones** subnet-0ae249b0fd821cab0 us-east-1a (use1-az6) subnet-0bf5f627735475f32 us-east-1d (use1-az4)	**Date created** September 28, 2025, 00:18 (UTC-04:00)

Load balancer ARN
arn:aws:elasticloadbalancing:us-east-1:216989125966:loadbalancer/app/project-load-balancer/9478a23d700bcb9b

DNS name Info
project-load-balancer-1056106433.us-east-1.elb.amazonaws.com (A Record)

Listeners and rules (1) Info

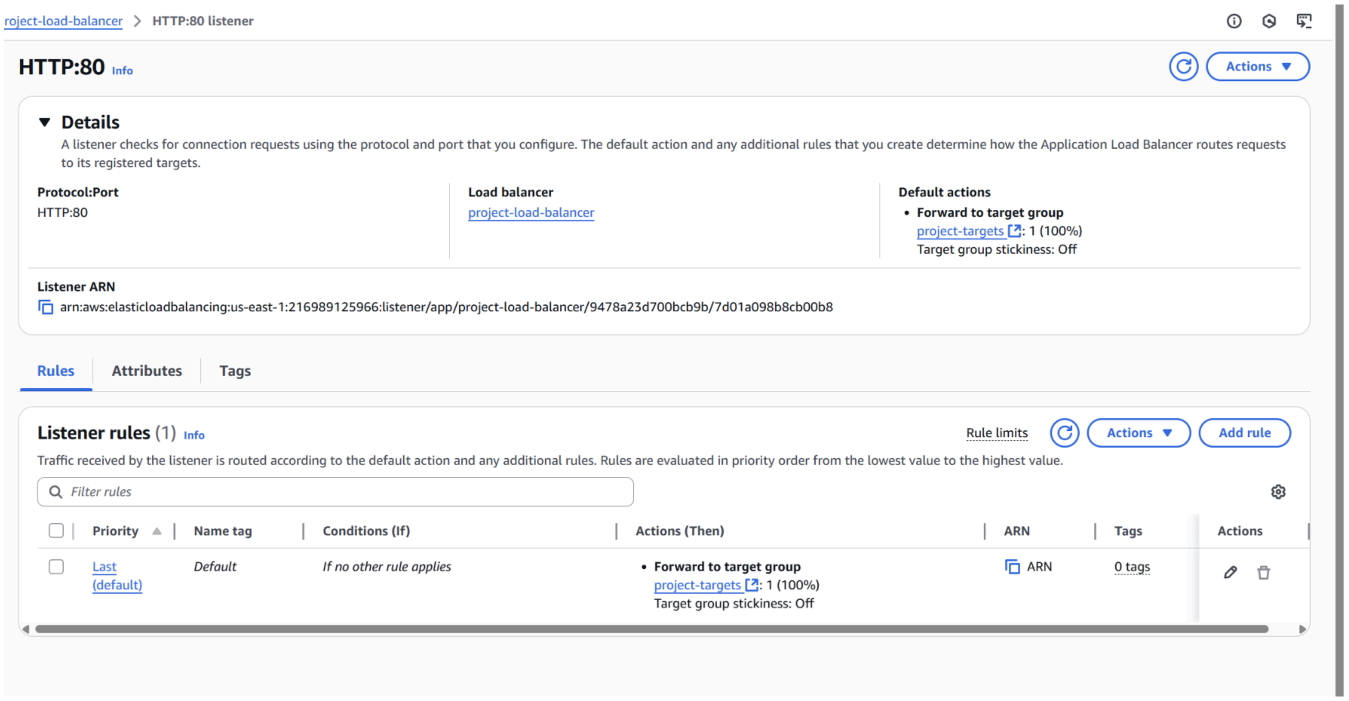
A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS
HTTP:80	• Forward to target group project-targets: 1 (100%) Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not app

- Command (Created Listener for the load balancer):

```
aws elbv2 create-listener \
  --load-balancer-arn arn:aws:elasticloadbalancing:us-east-1:216989125966:loadbalancer/app/project-load-balancer/9478a23d700bcb9b \
  --protocol HTTP \
  --port 80 \
  --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-east-1:216989125966:targetgroup/project-targets/48c0babd798ca746
```

Ouput arn:
arn:aws:elasticloadbalancing:us-east-1:216989125966:listener/app/project-load-balancer/9478a23d700bcb9b/7d01a098b8cb00b8



roject-load-balancer > HTTP:80 listener

HTTP:80 Info

Actions ▼

▼ Details

A listener checks for connection requests using the protocol and port that you configure. The default action and any additional rules that you create determine how the Application Load Balancer routes requests to its registered targets.

Protocol:Port	Load balancer	Default actions
HTTP:80	project-load-balancer	• Forward to target group project-targets: 1 (100%) Target group stickiness: Off

Listener ARN

arn:aws:elasticloadbalancing:us-east-1:216989125966:listener/app/project-load-balancer/9478a23d700bcb9b/7d01a098b8cb00b8

Rules | Attributes | Tags

Listener rules (1) Info

Rule limits | Actions ▼ | Add rule

Traffic received by the listener is routed according to the default action and any additional rules. Rules are evaluated in priority order from the lowest value to the highest value.

Filter rules

Priority	Name tag	Conditions (If)	Actions (Then)	ARN	Tags	Actions
Last (default)	Default	If no other rule applies	• Forward to target group project-targets: 1 (100%) Target group stickiness: Off	ARN	0 tags	

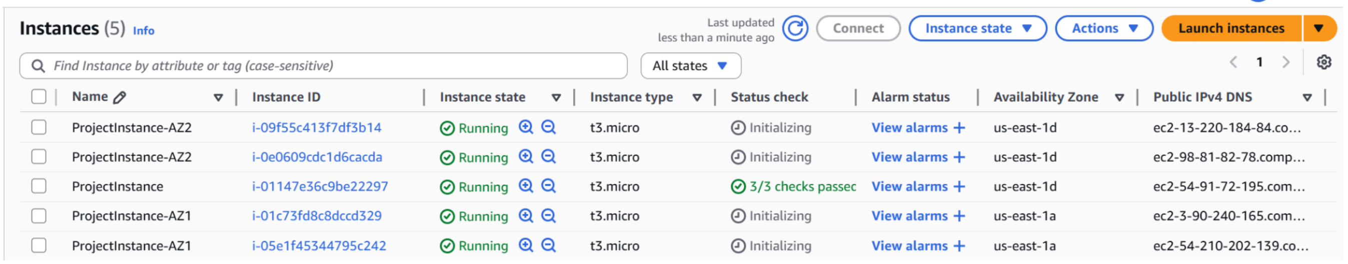
11. Set Up Multi-AZ Infrastructure Deployment

- Command (Deploy multiple EC2 instances across two Availability Zones for high availability):

```
# Launch instances in first AZ (us-east-1a)
aws ec2 run-instances \
  --image-id ami-0e89041b5de66d237 \
  --instance-type t3.micro \
  --key-name ProjectKeyPair \
  --security-group-ids sg-07e0ea73f7f6c93ee \
  --subnet-id subnet-0ae249b0fd821cab0 \
  --count 2 \
  --tag-specifications
'ResourceType=instance,Tags=[{Key=Name,Value=ProjectInstance-AZ1}]' \
  --user-data file://bootstrap.sh

# Launch instances in second AZ (us-east-1d)
aws ec2 run-instances \
  --image-id ami-0e89041b5de66d237 \
  --instance-type t3.micro \
  --key-name ProjectKeyPair \
  --security-group-ids sg-07e0ea73f7f6c93ee \
  --subnet-id subnet-0bf5f627735475f32 \
  --count 2 \
  --tag-specifications
'ResourceType=instance,Tags=[{Key=Name,Value=ProjectInstance-AZ2}]' \
  --user-data file://bootstrap.sh
```

Result: Multiple duplicate instances are created in AWS EC2 console instance list.

- ProjectInstance-AZ2 (i-09f55c413f7df3b14)
- ProjectInstance-AZ2 (i-0e0609cdc1d6cacda)
- ProjectInstance-AZ1 (i-01c73fd8c8dccd329)
- ProjectInstance-AZ1 (i-05e1f45344795c242)

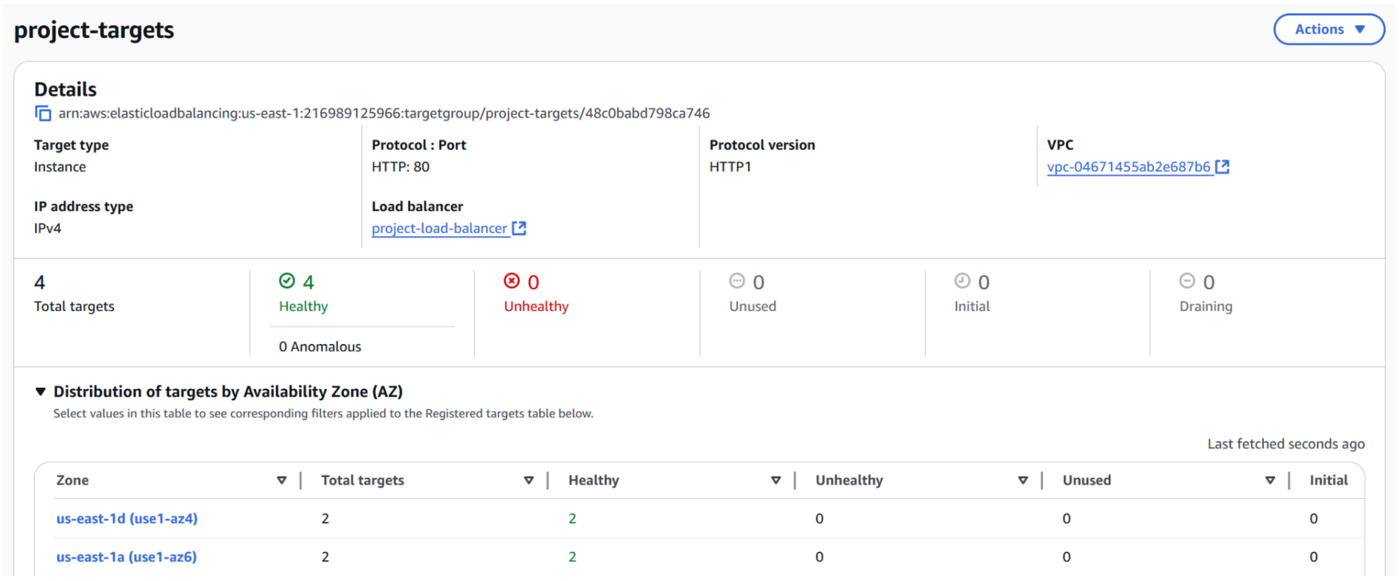


Instances (5) Info — Last updated less than a minute ago — Find Instance by attribute or tag (case-sensitive) — All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
ProjectInstance-AZ2	i-09f55c413f7df3b14	Running	t3.micro	Initializing	View alarms +	us-east-1d	ec2-13-220-184-84.co...
ProjectInstance-AZ2	i-0e0609cdc1d6cacda	Running	t3.micro	Initializing	View alarms +	us-east-1d	ec2-98-81-82-78.comp...
ProjectInstance	i-01147e36c9be22297	Running	t3.micro	3/3 checks passec	View alarms +	us-east-1d	ec2-54-91-72-195.com...
ProjectInstance-AZ1	i-01c73fd8c8dccd329	Running	t3.micro	Initializing	View alarms +	us-east-1a	ec2-3-90-240-165.com...
ProjectInstance-AZ1	i-05e1f45344795c242	Running	t3.micro	Initializing	View alarms +	us-east-1a	ec2-54-210-202-139.co...

- Command (Registered the instances into target group):

```
aws elbv2 register-targets \
    --target-group-arn
arn:aws:elasticloadbalancing:us-east-1:216989125966:targetgroup/project-targets/
48c0babd798ca746 \
    --targets Id=i-09f55c413f7df3b14 Id=i-0e0609cdc1d6cacda
Id=i-01c73fd8c8dccd329 Id=i-05e1f45344795c242
```



project-targets

Actions

Details

arn:aws:elasticloadbalancing:us-east-1:216989125966:targetgroup/project-targets/48c0babd798ca746

Target type Instance	**Protocol : Port** HTTP: 80	**Protocol version** HTTP1	**VPC** vpc-04671455ab2e687b6
IP address type IPv4	**Load balancer** project-load-balancer		

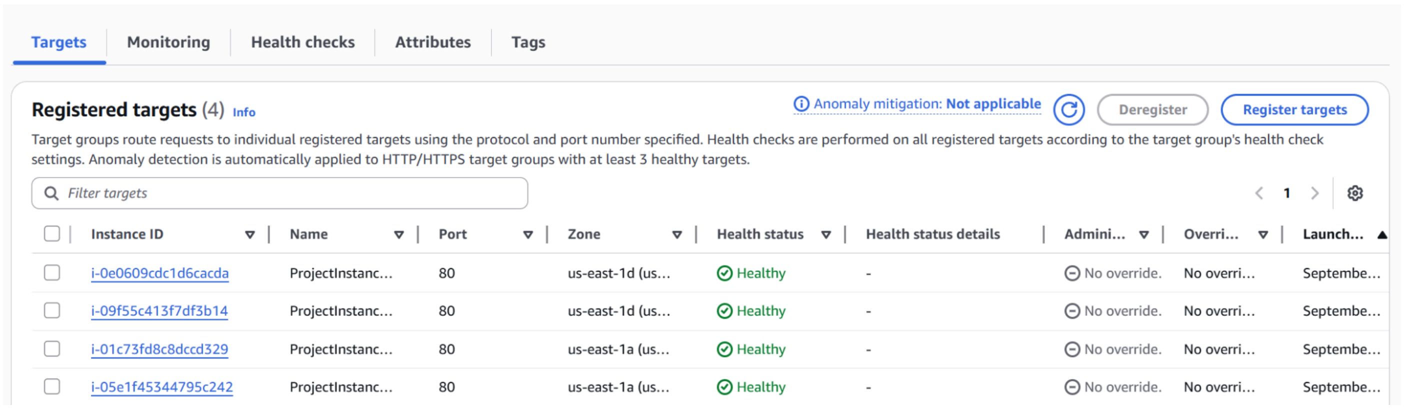
4 Total targets	✓ 4 Healthy; 0 Anomalous	0 Unhealthy	0 Unused	0 Initial	0 Draining

Distribution of targets by Availability Zone (AZ)

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

Last fetched seconds ago

Zone	Total targets	Healthy	Unhealthy	Unused	Initial
us-east-1d (use1-az4)	2	2	0	0	0
us-east-1a (use1-az6)	2	2	0	0	0



Registered targets (4) 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

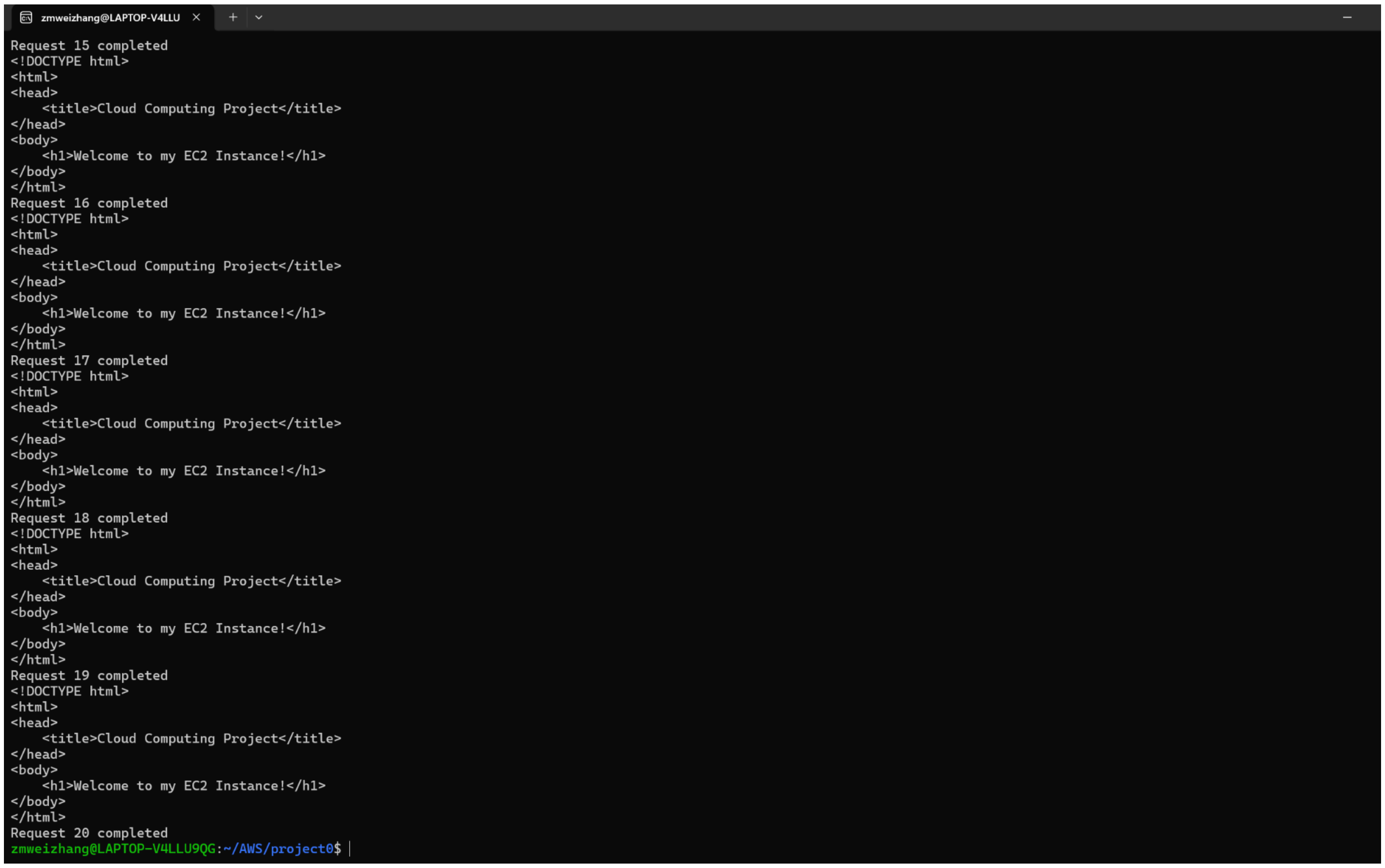
Instance ID	Name	Port	Zone	Health status	Health status details	Admini...	Overri...	Launch...
i-0e0609cdc1d6cacda	ProjectInstanc...	80	us-east-1d (us...	✓ Healthy	-	No override.	No overri...	Septembe...
i-09f55c413f7df3b14	ProjectInstanc...	80	us-east-1d (us...	✓ Healthy	-	No override.	No overri...	Septembe...
i-01c73fd8c8dccd329	ProjectInstanc...	80	us-east-1a (us...	✓ Healthy	-	No override.	No overri...	Septembe...
i-05e1f45344795c242	ProjectInstanc...	80	us-east-1a (us...	✓ Healthy	-	No override.	No overri...	Septembe...

12. Tested High Availability Architecture

- Command (Create test script to verify distribution, make it executable, and run it):

```
cat << 'EOF' > test-distribution.sh
#!/bin/bash
for i in {1..20}; do
  curl -s http://project-load-balancer-1056106433.us-east-1.elb.amazonaws.com
  echo "Request $i completed"
  sleep 1
done
EOF

chmod +x test-distribution.sh

./test-distribution.sh
```

Output: The test script sent 20 HTTP requests to the load balancer DNS. Each response displayed the custom web page "Welcome to my EC2 Instance!", confirming that the load balancer routed traffic to the EC2 instances across multiple Availability Zones.

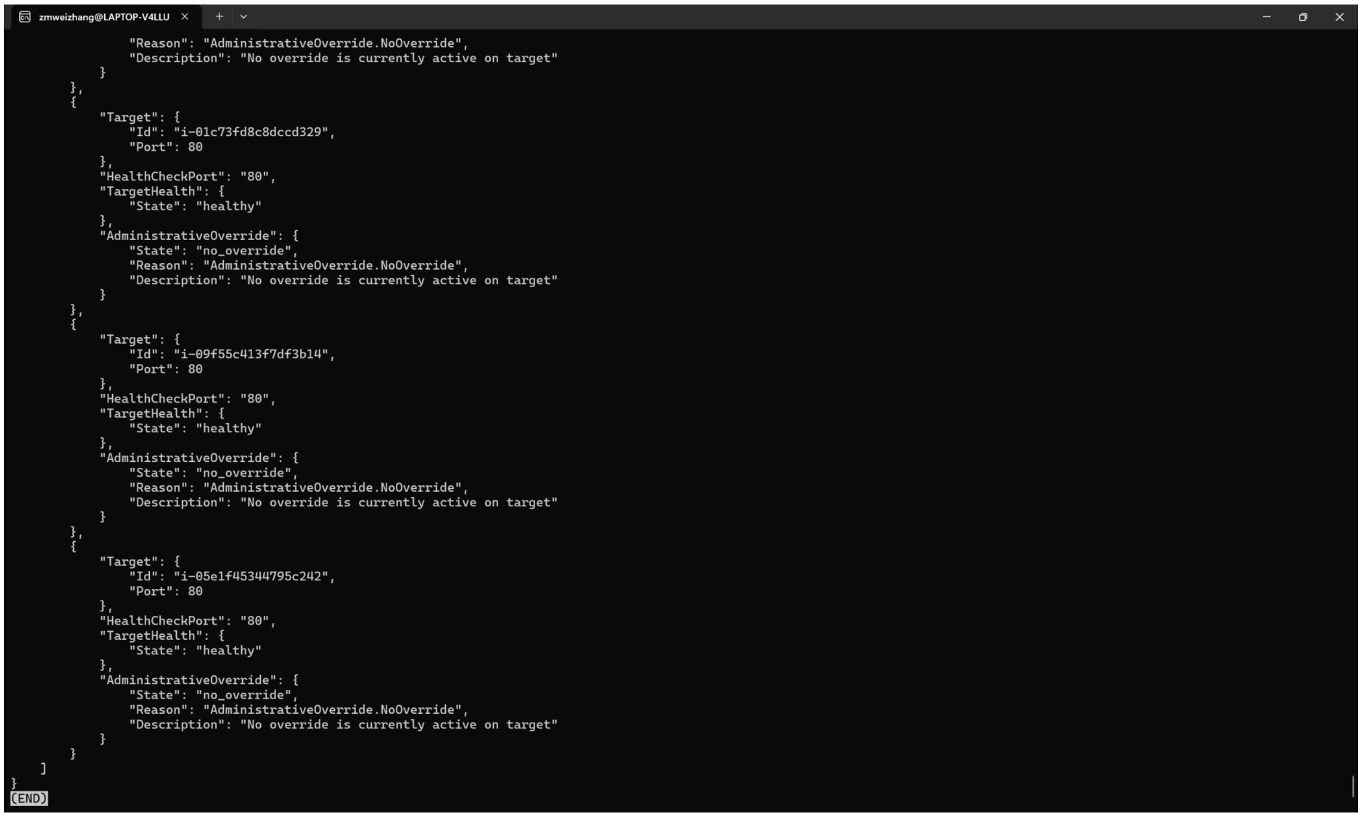


```
Request 15 completed
<!DOCTYPE html>
<html>
<head>
    <title>Cloud Computing Project</title>
</head>
<body>
    <h1>Welcome to my EC2 Instance!</h1>
</body>
</html>
Request 16 completed
<!DOCTYPE html>
<html>
<head>
    <title>Cloud Computing Project</title>
</head>
<body>
    <h1>Welcome to my EC2 Instance!</h1>
</body>
</html>
Request 17 completed
<!DOCTYPE html>
<html>
<head>
    <title>Cloud Computing Project</title>
</head>
<body>
    <h1>Welcome to my EC2 Instance!</h1>
</body>
</html>
Request 18 completed
<!DOCTYPE html>
<html>
<head>
    <title>Cloud Computing Project</title>
</head>
<body>
    <h1>Welcome to my EC2 Instance!</h1>
</body>
</html>
Request 19 completed
<!DOCTYPE html>
<html>
<head>
    <title>Cloud Computing Project</title>
</head>
<body>
    <h1>Welcome to my EC2 Instance!</h1>
</body>
</html>
Request 20 completed
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$
```

- Command (Test Failover by terminate 1 instance monitor health checks)

```
aws ec2 terminate-instances --instance-ids i-09f55c413f7df3b14

aws elbv2 describe-target-health \
    --target-group-arn
arn:aws:elasticloadbalancing:us-east-1:216989125966:targetgroup/project-targets/
48c0babd798ca746
```

Output: The describe-target-health command showed that the registered instances (i-01c73fd8c8dccd329, i-09f55c413f7df3b14, i-05e1f45344795c242) all passed health checks with State: healthy, which the load balancer is actively routing traffic to them.



```
zmweizhang@LAPTOP-V4LLU
                "Reason": "AdministrativeOverride.NoOverride",
                "Description": "No override is currently active on target"
            }
        },
        {
            "Target": {
                "Id": "i-01c73fd8c8dccd329",
                "Port": 80
            },
            "HealthCheckPort": "80",
            "TargetHealth": {
                "State": "healthy"
            },
            "AdministrativeOverride": {
                "State": "no_override",
                "Reason": "AdministrativeOverride.NoOverride",
                "Description": "No override is currently active on target"
            }
        },
        {
            "Target": {
                "Id": "i-09f55c413f7df3b14",
                "Port": 80
            },
            "HealthCheckPort": "80",
            "TargetHealth": {
                "State": "healthy"
            },
            "AdministrativeOverride": {
                "State": "no_override",
                "Reason": "AdministrativeOverride.NoOverride",
                "Description": "No override is currently active on target"
            }
        },
        {
            "Target": {
                "Id": "i-05e1f45344795c242",
                "Port": 80
            },
            "HealthCheckPort": "80",
            "TargetHealth": {
                "State": "healthy"
            },
            "AdministrativeOverride": {
                "State": "no_override",
                "Reason": "AdministrativeOverride.NoOverride",
                "Description": "No override is currently active on target"
            }
        }
    ]
}
(END)
```

13. Created Dashboard for CloudWatch

- Command (create a file named "dashboard.json" and fill in):

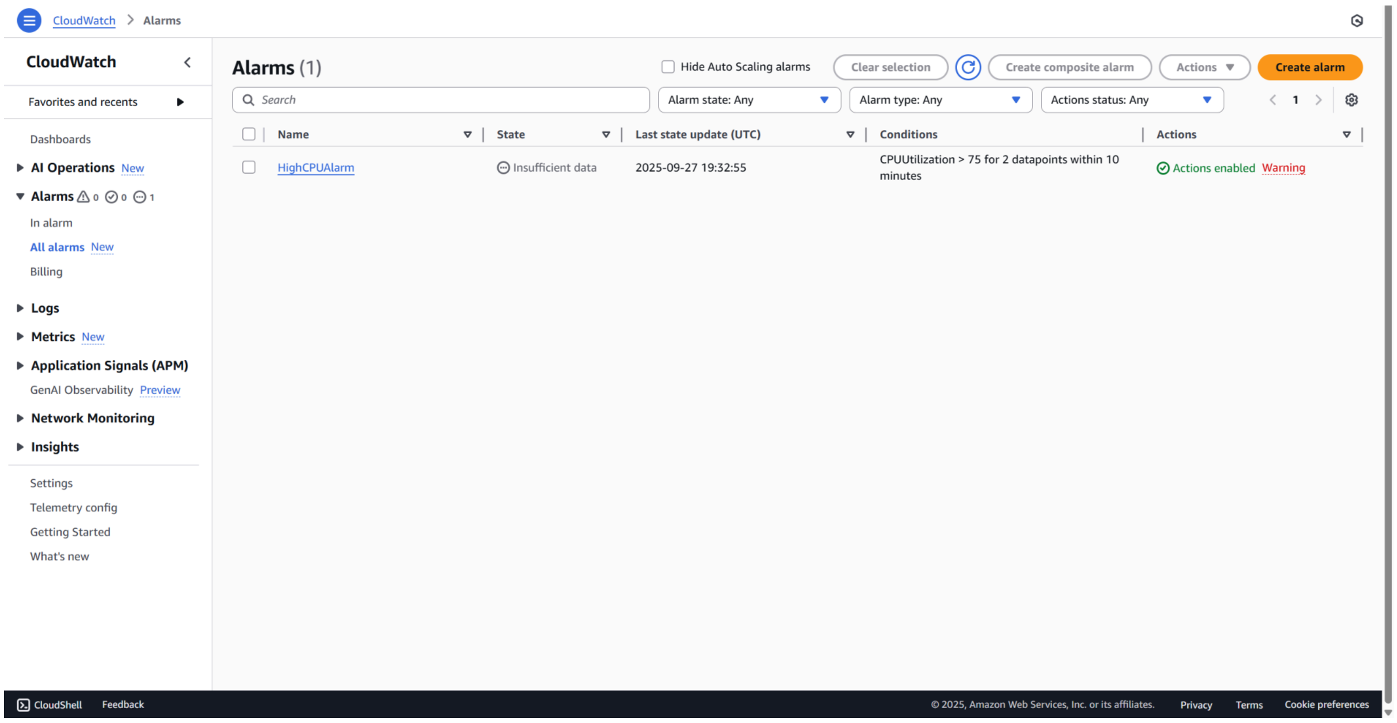
```
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/ApplicationELB", "RequestCount", "LoadBalancer",
"app/project-load-balancer/9478a23d700bcb9b", { "stat": "Sum", "period": 300 }
]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "us-east-1",
            "title": "Load Balancer Request Count",
            "period": 300
          }
        },
        {
          "type": "metric",
          "x": 12,
          "y": 0,
          "width": 12,
          "height": 6,
          "properties": {
            "metrics": [
              [ "AWS/ApplicationELB", "TargetResponseTime", "LoadBalancer",
"app/project-load-balancer/9478a23d700bcb9b", { "stat": "Average", "period":
300 } ]
            ],
            "view": "timeSeries",
            "stacked": false,
            "region": "us-east-1",
            "title": "Target Response Times",
            "period": 300
          }
        },
        {
          "type": "metric",
          "x": 0,
          "y": 6,
          "width": 12,
          "height": 6,
          "properties": {
            "metrics": [
              [ "AWS/EC2", "CPUUtilization", "InstanceId",
"i-01c73fd8c8dccd329", { "stat": "Average", "period": 300 } ],
              [ "AWS/EC2", "CPUUtilization", "InstanceId",
"i-05e1f45344795c242", { "stat": "Average", "period": 300 } ],
              [ "AWS/EC2", "CPUUtilization", "InstanceId",
"i-0e0609cdc1d6cacda", { "stat": "Average", "period": 300 } ]
            ],
            "view": "timeSeries",
            "stacked": false,
                "region": "us-east-1",
                "title": "Instance CPU Utilization",
                "period": 300
            }
        },
        {
            "type": "metric",
            "x": 12,
            "y": 6,
            "width": 12,
            "height": 6,
            "properties": {
                "metrics": [
                    [ "AWS/EC2", "NetworkIn", "InstanceId", "i-01c73fd8c8dccd329", {
"stat": "Sum", "period": 300 } ],
                    [ "AWS/EC2", "NetworkIn", "InstanceId", "i-05e1f45344795c242", {
"stat": "Sum", "period": 300 } ],
                    [ "AWS/EC2", "NetworkIn", "InstanceId", "i-0e0609cdc1d6cacda", {
"stat": "Sum", "period": 300 } ],
                    [ "AWS/EC2", "NetworkOut", "InstanceId", "i-01c73fd8c8dccd329",
{ "stat": "Sum", "period": 300 } ],
                    [ "AWS/EC2", "NetworkOut", "InstanceId", "i-05e1f45344795c242",
{ "stat": "Sum", "period": 300 } ],
                    [ "AWS/EC2", "NetworkOut", "InstanceId", "i-0e0609cdc1d6cacda",
{ "stat": "Sum", "period": 300 } ]
                ],
                "view": "timeSeries",
                "stacked": false,
                "region": "us-east-1",
                "title": "Instance Network Traffic",
                "period": 300
            }
        }
    ]
}
```

- Command (Run the AWS CLI command to create the CloudWatch dashboard):

```
aws cloudwatch put-dashboard \
    --dashboard-name "ProjectDashboard" \
    --dashboard-body file://dashboard.json
```

- Command (Create CloudWatch Alarms):

```
aws cloudwatch put-metric-alarm \
  --alarm-name HighCPUAlarm \
  --alarm-description "Alarm when CPU exceeds 75%" \
  --metric-name CPUUtilization \
  --namespace AWS/EC2 \
  --statistic Average \
  --period 300 \
  --threshold 75 \
  --comparison-operator GreaterThanThreshold \
  --evaluation-periods 2 \
  --alarm-actions arn:aws:sns:us-east-1:216989125966:ProjectAlerts \
  --dimensions Name=InstanceId,Value=i-01c73fd8c8dccd329
```

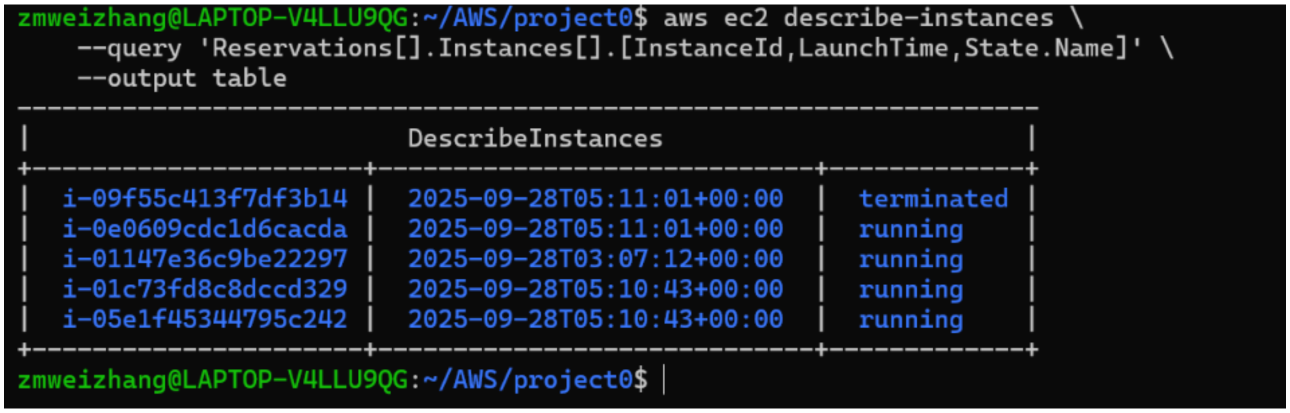


Name	State	Last state update (UTC)	Conditions	Actions
HighCPUAlarm	Insufficient data	2025-09-27 19:32:55	CPUUtilization > 75 for 2 datapoints within 10 minutes	Actions enabled Warning

14. Cost Analysis and Resource Optimization

- Command (List running instances with launch times and states):

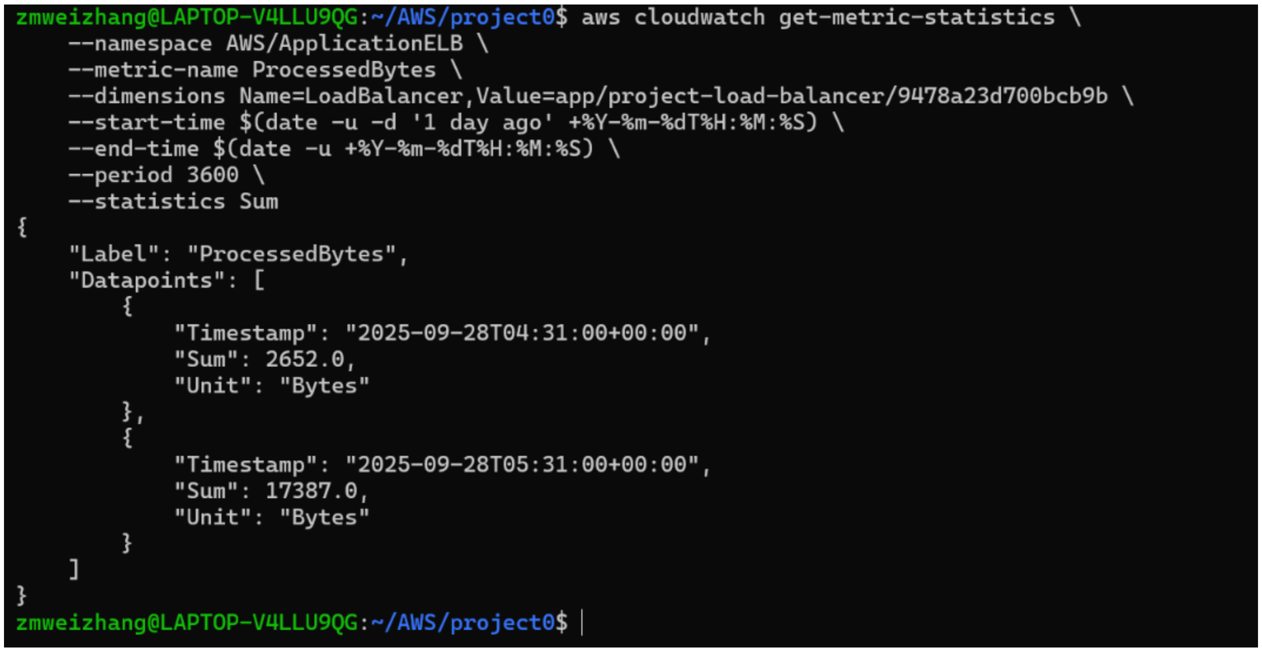
```
aws ec2 describe-instances \
  --query 'Reservations[].Instances[].[InstanceId,LaunchTime,State.Name]' \
  --output table
```



```
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$ aws ec2 describe-instances \
    --query 'Reservations[].Instances[].[InstanceId,LaunchTime,State.Name]' \
    --output table
--------------------------------------------------------------------------
|                           DescribeInstances                            |
+------------------------+-------------------------------+--------------+
|  i-09f55c413f7df3b14   |  2025-09-28T05:11:01+00:00    |  terminated  |
|  i-0e0609cdc1d6cacda   |  2025-09-28T05:11:01+00:00    |  running     |
|  i-01147e36c9be22297   |  2025-09-28T03:07:12+00:00    |  running     |
|  i-01c73fd8c8dccd329   |  2025-09-28T05:10:43+00:00    |  running     |
|  i-05e1f45344795c242   |  2025-09-28T05:10:43+00:00    |  running     |
+------------------------+-------------------------------+--------------+
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$
```

- Command (Get load balancer processed bytes in the last 24 hours):

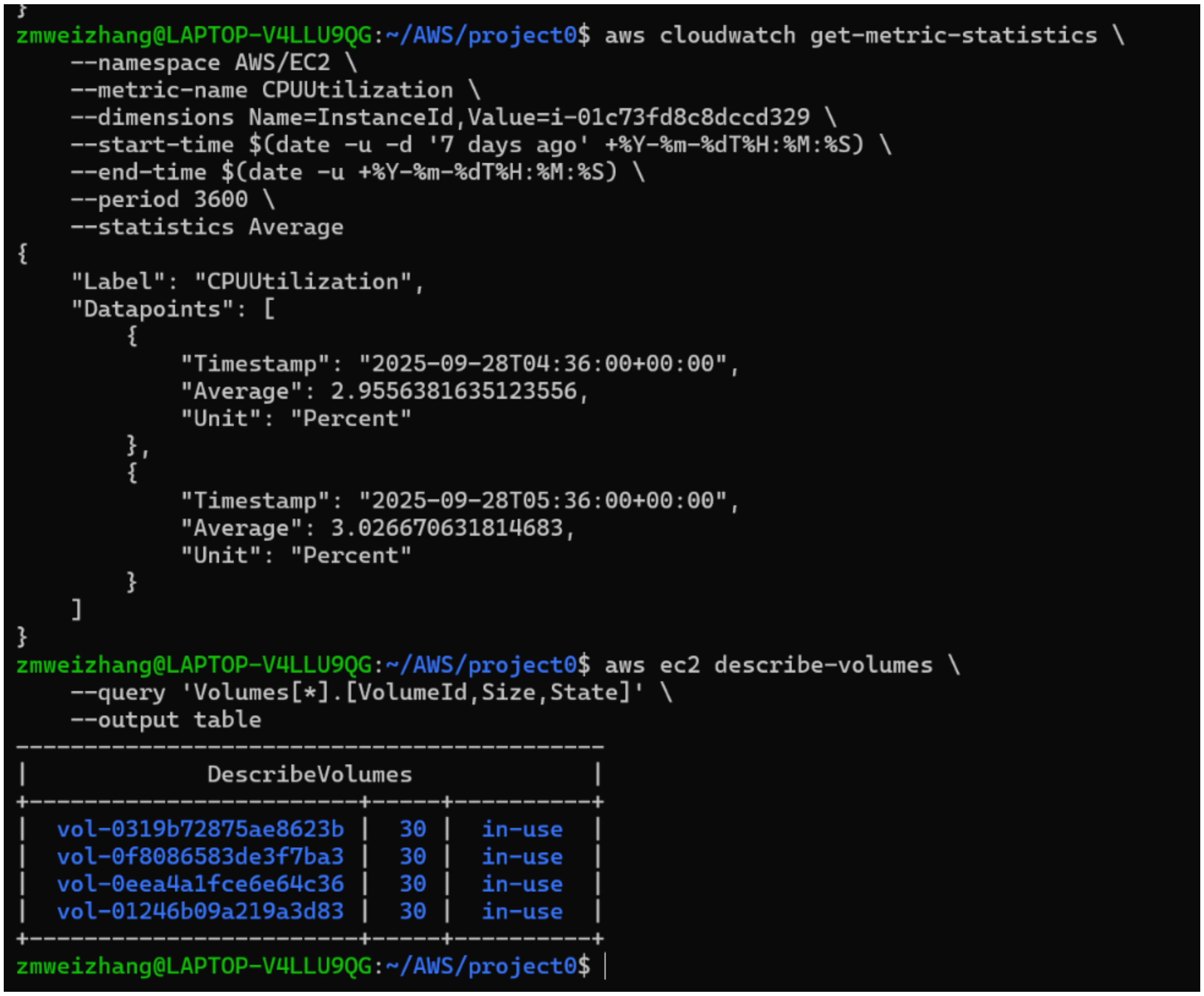
```
aws cloudwatch get-metric-statistics \
  --namespace AWS/ApplicationELB \
  --metric-name ProcessedBytes \
  --dimensions
Name=LoadBalancer,Value=app/project-load-balancer/9478a23d700bcb9b \
  --start-time $(date -u -d '1 day ago' +%Y-%m-%dT%H:%M:%S) \
  --end-time $(date -u +%Y-%m-%dT%H:%M:%S) \
  --period 3600 \
  --statistics Sum
```



```
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$ aws cloudwatch get-metric-statistics \
    --namespace AWS/ApplicationELB \
    --metric-name ProcessedBytes \
    --dimensions Name=LoadBalancer,Value=app/project-load-balancer/9478a23d700bcb9b \
    --start-time $(date -u -d '1 day ago' +%Y-%m-%dT%H:%M:%S) \
    --end-time $(date -u +%Y-%m-%dT%H:%M:%S) \
    --period 3600 \
    --statistics Sum
{
    "Label": "ProcessedBytes",
    "Datapoints": [
        {
            "Timestamp": "2025-09-28T04:31:00+00:00",
            "Sum": 2652.0,
            "Unit": "Bytes"
        },
        {
            "Timestamp": "2025-09-28T05:31:00+00:00",
            "Sum": 17387.0,
            "Unit": "Bytes"
        }
    ]
}
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$
```

- Command (Check instance CPU utilization over last 7 days):

```
aws cloudwatch get-metric-statistics \
    --namespace AWS/EC2 \
    --metric-name CPUUtilization \
    --dimensions Name=InstanceId,Value=i-01c73fd8c8dccd329 \
    --start-time $(date -u -d '7 days ago' +%Y-%m-%dT%H:%M:%S) \
    --end-time $(date -u +%Y-%m-%dT%H:%M:%S) \
    --period 3600 \
    --statistics Average

aws ec2 describe-volumes \
    --query 'Volumes[*].[VolumeId,Size,State]' \
    --output table
```



```
}
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$ aws cloudwatch get-metric-statistics \
    --namespace AWS/EC2 \
    --metric-name CPUUtilization \
    --dimensions Name=InstanceId,Value=i-01c73fd8c8dccd329 \
    --start-time $(date -u -d '7 days ago' +%Y-%m-%dT%H:%M:%S) \
    --end-time $(date -u +%Y-%m-%dT%H:%M:%S) \
    --period 3600 \
    --statistics Average
{
    "Label": "CPUUtilization",
    "Datapoints": [
        {
            "Timestamp": "2025-09-28T04:36:00+00:00",
            "Average": 2.9556381635123556,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2025-09-28T05:36:00+00:00",
            "Average": 3.026670631814683,
            "Unit": "Percent"
        }
    ]
}
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$ aws ec2 describe-volumes \
    --query 'Volumes[*].[VolumeId,Size,State]' \
    --output table
-------------------------------------------
|             DescribeVolumes             |
+-------------------------+-----+---------+
|  vol-0319b72875ae8623b  |  30 |  in-use |
|  vol-0f8086583de3f7ba3  |  30 |  in-use |
|  vol-0eea4a1fce6e64c36  |  30 |  in-use |
|  vol-01246b09a219a3d83  |  30 |  in-use |
+-------------------------+-----+---------+
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$
```

15. Clean Up:

- Command (Deleted the listener, load balancer, and target group):

```
aws elbv2 delete-listener \
   --listener-arn
arn:aws:elasticloadbalancing:us-east-1:216989125966:listener/app/project-load-b
alancer/9478a23d700bcb9b/7d01a098b8cb00b8

aws elbv2 delete-load-balancer \
   --load-balancer-arn
arn:aws:elasticloadbalancing:us-east-1:216989125966:loadbalancer/app/project-l
oad-balancer/9478a23d700bcb9b

aws elbv2 delete-target-group \
   --target-group-arn
arn:aws:elasticloadbalancing:us-east-1:216989125966:targetgroup/project-target
s/48c0babd798ca746
```

- Command (Terminated all instances, 5 instances total):

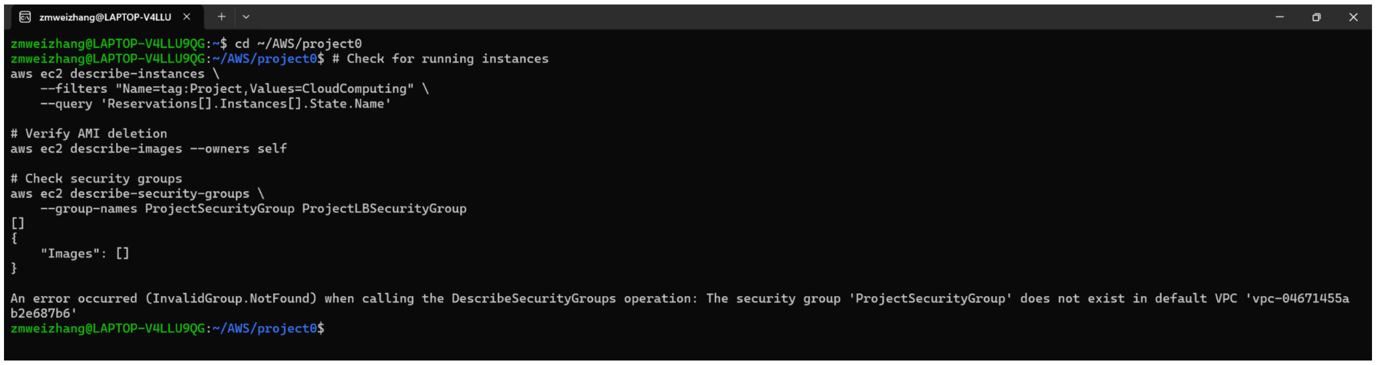
```
aws ec2 terminate-instances \
   --instance-ids i-09f55c413f7df3b14 i-0e0609cdc1d6cacda
i-01c73fd8c8dccd329 i-05e1f45344795c242 i-01147e36c9be22297
```

- Command (Deleted supporting resources, such as AMI, security groups, key pair):

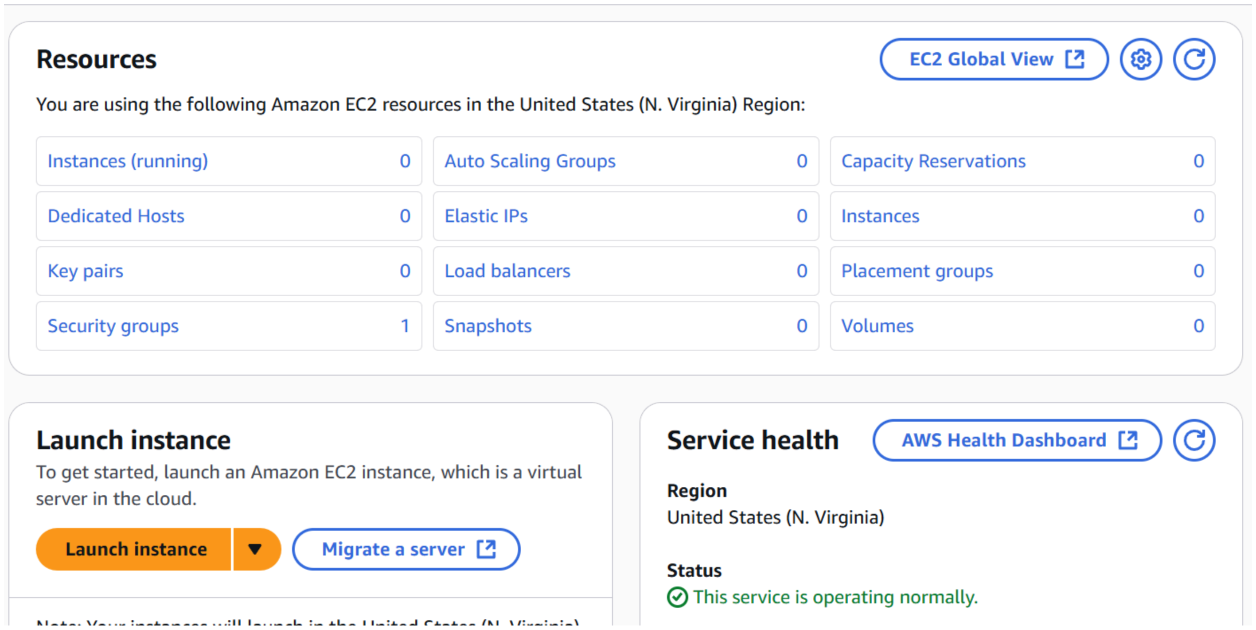
```
aws ec2 deregister-image --image-id ami-0e89041b5de66d237

aws ec2 delete-security-group --group-name ProjectSecurityGroup
aws ec2 delete-security-group --group-name ProjectLBSecurityGroup

aws ec2 delete-key-pair --key-name ProjectKeyPair
```

- Command (Verified clean up):

```
aws ec2 describe-instances \
   --filters "Name=tag:Project,Values=CloudComputing" \
   --query 'Reservations[].Instances[].State.Name'

aws ec2 describe-images --owners self

aws ec2 describe-security-groups \
   --group-names ProjectSecurityGroup ProjectLBSecurityGroup
```



```
zmweizhang@LAPTOP-V4LLU9QG:~$ cd ~/AWS/project0
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$ # Check for running instances
aws ec2 describe-instances \
    --filters "Name=tag:Project,Values=CloudComputing" \
    --query 'Reservations[].Instances[].State.Name'

# Verify AMI deletion
aws ec2 describe-images --owners self

# Check security groups
aws ec2 describe-security-groups \
    --group-names ProjectSecurityGroup ProjectLBSecurityGroup
[]
{
    "Images": []
}

An error occurred (InvalidGroup.NotFound) when calling the DescribeSecurityGroups operation: The security group 'ProjectSecurityGroup' does not exist in default VPC 'vpc-04671455a
b2e687b6'
zmweizhang@LAPTOP-V4LLU9QG:~/AWS/project0$
```



Resources

You are using the following Amazon EC2 resources in the United States (N. Virginia) Region:

Instances (running)	0	Auto Scaling Groups	0	Capacity Reservations	0
Dedicated Hosts	0	Elastic IPs	0	Instances	0
Key pairs	0	Load balancers	0	Placement groups	0
Security groups	1	Snapshots	0	Volumes	0

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Service health

Region
United States (N. Virginia)

Status
This service is operating normally.

(Security groups displayed as 1 refers to the default VPC security group AWS provided by default)

Output: Instances, AMIs, and security groups are all shown empty.