

## **PROJECT 1 — 1. ECS with ALB, RDS, Route53, Acm, Iam and SecretsManager**

### **a. Infrastructure Requirements:**

#### **i. ECS**

**1. Create an ECS Cluster with the service running in Private Subnets.**

#### **2. Services**

**a. WordPress Docker Image**

**b. Custom microservice (a lightweight Node.js application responding with “Hello from Microservice”). Share the Node.js code and DockerFile too.**

**3. Setup auto scaling based on CPU and Memory.**

#### **ii. RDS**

**1. Choose the appropriate instance type to be used by WordPress as a database.**

**2. Create custom user and password in RDS to be used with Wordpress that do not auto rotate.**

**3. Configure automated backups.**

**4. RDS instance should be deployed in the Private Subnets.**

#### **iii. SecretsManager**

**1. Store the RDS database credentials as secrets.**

#### **iv. IAM and Security**

**1. Configure the ECS task definition to use the secrets stored in AWS SecretsManager, ensuring the WordPress securely connects to the RDS instance.**

**2. Use IAM roles to grant the ECS service the necessary permissions to access secrets from AWS SecretsManager.**

**3. Security Groups should be the least privilege.**

#### **v. Application Load Balancer (ALB) & Domain Mapping**

**1. Set up an Application Load Balancer (ALB) in Public Subnets to handle incoming HTTP/HTTPS traffic.**

2. Set up SSL certificate. The website should not open HTTP.
3. HTTP traffic should be redirected to HTTPS.
4. Configure the ALB to associate with a domain name.  
"wordpress.<domain-name>"  
"microservice.<domain-name>"

## **AWS Services Used**

### **Networking & DNS**

1. Amazon VPC – Custom VPC with public and private subnets
2. Amazon Subnets – Public (ALB) and Private (ECS, RDS) subnets
3. Internet Gateway (IGW) – Internet access for public subnets
4. NAT Gateway – Outbound internet access for ECS tasks in private subnets
5. Route Tables – Public, private, and database routing
6. Amazon Route 53 – DNS management and domain mapping

### **Load Balancing & Security**

7. Application Load Balancer (ALB) – Traffic routing and host-based routing
8. AWS Certificate Manager (ACM) – SSL/TLS certificates
9. Security Groups – Least-privilege network access control

## **Containers & Orchestration**

- 10. Amazon ECS (Fargate) – Container orchestration without EC2 management**
- 11. Amazon ECR – Docker image repository**

## **Database & Storage**

- 12. Amazon RDS (MySQL) – Managed relational database for WordPress**
- 13. RDS Subnet Group – Database isolation in private subnets**

## **Security & Identity**

- 14. AWS IAM – Roles and policies for ECS task execution**
- 15. AWS Secrets Manager – Secure storage of database credentials**

## **Monitoring & Logging**

- 16. Amazon CloudWatch – Logs and metrics for ECS tasks and ALB**

## **Automation & Scaling**

## 17. ECS Service Auto Scaling – CPU and memory-based auto scaling

Step 1) We will create Vpc with cidr block 172.16.0.0/16

CreateVpc | VPC Console

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateVpc:createMode=vpcOnly

aws Search [Alt+S]

VPC > Your VPCs > Create VPC

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

CUSTOM\_VPC

**IPv4 CIDR block** Info

☒ IPv4 CIDR manual input  
☐ IPAM-allocated IPv4 CIDR block

**IPv4 CIDR**

172.16.0.0/16

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

**IPv6 CIDR block** Info

☒ No IPv6 CIDR block  
☐ IPAM-allocated IPv6 CIDR block  
☐ Amazon-provided IPv6 CIDR block  
☐ IPv6 CIDR owned by me

**Tenancy** Info

Default

**VPC encryption control (\$)** - new Info

Monitor mode provides visibility into encryption status without blocking traffic. Enforce mode prevents unencrypted traffic. Additional charges apply

☒ None  
☐ Monitor mode  
See which resources in your VPC are unencrypted but allow the creation of unencrypted resources.

☐ Enforce mode  
Requires all resources, except exclusions, in your VPC to be encryption-capable and blocks creation of unencrypted resources.

Step 2) We will create subnets for Load balancer, App(ECS clusters), DB subnets.

a) Will create Alb private subnet 1 with cidr block 172.16.0.0/24



VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateSubnet:

Search [Alt+S]

VPC > Subnets > Create subnet

Associated VPC CIDRs

IPv4 CIDRs

172.16.0.0/16

**Subnet settings**

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

**Subnet name**

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

ALB\_PUBLIC\_SUBNET\_1

The name can be up to 256 characters long.

**Availability Zone** [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

Asia Pacific (Mumbai) / ap-south-1a

**IPv4 VPC CIDR block** [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

172.16.0.0/16

**IPv4 subnet CIDR block**

172.16.0.0/24 256 IPs

b) We will create alb public subnet 2 with cidr block 172.16.1.0/24

VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateSubnet:

Search [Alt+S]

VPC > Subnets > Create subnet

172.16.0.0/16

**Subnet settings**

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

**Subnet name**

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

ALB\_PUBLIC\_SUBNET\_2

The name can be up to 256 characters long.

**Availability Zone** [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

Asia Pacific (Mumbai) / ap-south-1b

**IPv4 VPC CIDR block** [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

172.16.0.0/16

**IPv4 subnet CIDR block**

172.16.1.0/24 256 IPs

▼ Tags - optional

Key	Value - optional
Name	ALB_PUBLIC_SUBNET_2

c) We will create App private subnet 1 with cidr block 172.16.2.0/24

VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpconsole/home?region=ap-south-1#CreateSubnet:

aws Search [Alt+S] Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

VPC > Subnets > Create subnet

IPv4 CIDRs  
172.16.0.0/16

**Subnet settings**  
Specify the CIDR blocks and Availability Zone for the subnet.

**Subnet 1 of 1**

**Subnet name**  
Create a tag with a key of 'Name' and a value that you specify.  
APP\_PRIVATE\_SUBNET\_1  
The name can be up to 256 characters long.

**Availability Zone** [Info](#)  
Choose the zone in which your subnet will reside, or let Amazon choose one for you.  
Asia Pacific (Mumbai) / ap-s1-az1 (ap-south-1a)

**IPv4 VPC CIDR block** [Info](#)  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.  
172.16.0.0/16

**IPv4 subnet CIDR block**  
172.16.2.0/24 256 IPs

Tags - optional

d) We will create app private subnet 2 with cidr block 172.16.3.0/24

VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpconsole/home?region=ap-south-1#CreateSubnet:

aws Search [Alt+S] Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

VPC > Subnets > Create subnet

Associated VPC CIDRs

IPv4 CIDRs  
172.16.0.0/16

**Subnet settings**  
Specify the CIDR blocks and Availability Zone for the subnet.

**Subnet 1 of 1**

**Subnet name**  
Create a tag with a key of 'Name' and a value that you specify.  
APP\_PRIVATE\_SUBNET\_2  
The name can be up to 256 characters long.

**Availability Zone** [Info](#)  
Choose the zone in which your subnet will reside, or let Amazon choose one for you.  
Asia Pacific (Mumbai) / ap-s1-az3 (ap-south-1b)

**IPv4 VPC CIDR block** [Info](#)  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.  
172.16.0.0/16

**IPv4 subnet CIDR block**  
172.16.3.0/24 256 IPs

e) We will create now DB subnet 1 for database with cidr block 172.16.4.0/24

VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateSubnet:

Search [Alt+S]

Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

VPC > Subnets > Create subnet

Associated VPC CIDRs

IPv4 CIDRs  
172.16.0.0/16

**Subnet settings**  
Specify the CIDR blocks and Availability Zone for the subnet.

**Subnet 1 of 1**

**Subnet name**  
Create a tag with a key of 'Name' and a value that you specify.  
DB\_PRIVATE\_SUBNET\_1  
The name can be up to 256 characters long.

**Availability Zone** [Info](#)  
Choose the zone in which your subnet will reside, or let Amazon choose one for you.  
Asia Pacific (Mumbai) / ap-south-1a

**IPv4 VPC CIDR block** [Info](#)  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.  
172.16.0.0/16

**IPv4 subnet CIDR block**  
172.16.4.0/24 256 IP

f) We will create DB subnet 2 with cidr block 172.16.5.0/24

VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateSubnet:

Search [Alt+S]

Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

VPC > Subnets > Create subnet

Associated VPC CIDRs

IPv4 CIDRs  
172.16.0.0/16

**Subnet settings**  
Specify the CIDR blocks and Availability Zone for the subnet.

**Subnet 1 of 1**

**Subnet name**  
Create a tag with a key of 'Name' and a value that you specify.  
DB\_PRIVATE-SUBNET\_2  
The name can be up to 256 characters long.

**Availability Zone** [Info](#)  
Choose the zone in which your subnet will reside, or let Amazon choose one for you.  
Asia Pacific (Mumbai) / ap-south-1b

**IPv4 VPC CIDR block** [Info](#)  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.  
172.16.0.0/16

**IPv4 subnet CIDR block**  
172.16.5.0/24 256 IP

Step 3) We will create internet gateway and attach it to Vpc to provide internet connectivity.

The screenshot shows the AWS Management Console interface for creating a new internet gateway. The breadcrumb navigation at the top indicates the path: VPC > Internet gateways > Create internet gateway. The main heading is 'Create internet gateway' with an 'Info' link. Below this, a descriptive sentence states: 'An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.' The 'Internet gateway settings' section contains a 'Name tag' field with the value 'IGW\_CUSTOM\_VPC'. The 'Tags - optional' section includes a table with one tag: Key 'Name' and Value 'IGW\_CUSTOM\_VPC'. There is an 'Add new tag' button and a note 'You can add 49 more tags.' At the bottom right, there are 'Cancel' and 'Create internet gateway' buttons.

Step 4) We will now attach the Internet gateway to the vpc

The screenshot shows the AWS Management Console interface for attaching an internet gateway to a VPC. The breadcrumb navigation at the top indicates the path: VPC > Internet gateways > Attach to VPC (igw-06414ce2905ebb304). A green notification banner at the top states: 'The following internet gateway was created: igw-06414ce2905ebb304 - IGW\_CUSTOM\_VPC. You can now attach to a VPC to enable the VPC to communicate with the internet.' The main heading is 'Attach to VPC (igw-06414ce2905ebb304)' with an 'Info' link. Below this, a descriptive sentence states: 'Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach below.' The 'Available VPCs' section includes a search bar with the value 'vpc-0cbb91d56e71b4fb7'. At the bottom right, there are 'Cancel' and 'Attach internet gateway' buttons.

Step 5) Now we'll add route to internet in alb route table to make alb subnets as public subnets.

ap-south-1 VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#RouteTableDetails:RouteTableId=rtb-0fed3cdf96c39f0f1

Search [Alt+S] Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

VPC > Route tables > rtb-0fed3cdf96c39f0f1

**VPC dashboard**

AWS Global View

Filter by VPC:

**Virtual private cloud**

Your VPCs

Subnets

**Route tables**

Internet gateways

Egress-only Internet gateways

DHCP option sets

Elastic IPs

Managed prefix lists

NAT gateways

**Details**

**Route table ID**  
rtb-0fed3cdf96c39f0f1

**Main**  
Yes

**Explicit subnet associations**  
2 subnets

**Edge associations**  
-

**VPC**  
vpc-0cbb91d56e71b4fb7 | CUSTOM\_VPC

**Owner ID**  
390503781838

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

**Routes (2)**

Filter routes

Both Edit routes

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-06414ce2905ebb...	Active	No	Create Route
172.16.0.0/16	local	Active	No	Create Route Table

Step 6) We now attach the alb subnets to this route table created above.

ap-south-1 VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#RouteTableDetails:RouteTableId=rtb-0fed3cdf96c39f0f1

Search [Alt+S] Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

VPC > Route tables > rtb-0fed3cdf96c39f0f1

**VPC dashboard**

AWS Global View

Filter by VPC:

**Virtual private cloud**

Your VPCs

Subnets

**Route tables**

Internet gateways

Egress-only Internet gateways

DHCP option sets

Elastic IPs

Managed prefix lists

NAT gateways

**Details**

**Route table ID**  
rtb-0fed3cdf96c39f0f1

**Main**  
Yes

**Explicit subnet associations**  
2 subnets

**Edge associations**  
-

**VPC**  
vpc-0cbb91d56e71b4fb7 | CUSTOM\_VPC

**Owner ID**  
390503781838

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

**Explicit subnet associations (2)**

Find subnet association

Edit subnet associations

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
ALB_PUBLIC_SUBNET_2	subnet-066347e4285a59a2a	172.16.1.0/24	-
ALB_PUBLIC_SUBNET_1	subnet-0a87fcb6fff130171	172.16.0.0/24	-

Step 7) We will now create private route table for app subnets and associate app/ecs subnets to this route table.

The screenshot shows the AWS VPC console for the region 'ap-south-1'. The breadcrumb navigation is 'VPC > Route tables > rtb-098a6e1f53002dc00'. A green notification banner at the top states: 'You have successfully updated subnet associations for rtb-098a6e1f53002dc00 / APP\_PRIVATE\_ROUTE\_TABLE.' The main heading is 'rtb-098a6e1f53002dc00 / APP\_PRIVATE\_ROUTE\_TABLE'. The 'Details' section shows the Route table ID, VPC, Main status, and Owner ID. The 'Subnet associations' tab is active, displaying a table with 2 explicit associations:

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
APP_PRIVATE_SUBNET_1	<a href="#">subnet-0eaf843ff693c4053</a>	172.16.2.0/24	-
APP_PRIVATE_SUBNET_2	<a href="#">subnet-001f00096d2d1f6a4</a>	172.16.3.0/24	-

Below this, it shows 'Subnets without explicit associations (2)' with a note that they are associated with the main route table.

Step 8) We will now create database route table and associate it with db subnets.

The screenshot shows the AWS VPC console for the region 'ap-south-1'. The breadcrumb navigation is 'VPC > Route tables > rtb-050f2db8207ab6f8e'. The main heading is 'rtb-050f2db8207ab6f8e / DB\_PRIVATE\_ROUTE\_TABLE'. The 'Details' section shows the Route table ID, VPC, Main status, and Owner ID. The 'Subnet associations' tab is active, displaying a table with 2 explicit associations:

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
DB_PRIVATE_SUBNET_1	<a href="#">subnet-0d8ff8fa7e92f2ed0</a>	172.16.4.0/24	-
DB_PRIVATE-SUBNET_2	<a href="#">subnet-0b15536b6a1070f4c</a>	172.16.5.0/24	-

Below this, it shows 'Subnets without explicit associations (0)' with a note that they are associated with the main route table.

Step 9) We can now verify all the resource created in vpc resource map .

The screenshot displays the AWS VPC console for the region 'ap-south-1'. The main view is the 'Resource map' for the VPC 'vpc-0cbb91d56e71b4fb7 / CUSTOM\_VPC'. The map shows the following resources:

- VPC:** vpc-0cbb91d56e71b4fb7 (CUSTOM\_VPC)
- Subnets (6):**
  - ap-south-1a: ALB\_PUBLIC\_SUBNET\_1, APP\_PRIVATE\_SUBNET\_1, DB\_PRIVATE\_SUBNET\_1
  - ap-south-1b: ALB\_PUBLIC\_SUBNET\_2, APP\_PRIVATE\_SUBNET\_2, DB\_PRIVATE\_SUBNET\_2
- Route tables (5):** APP\_PRIVATE\_ROUTE\_TABLE, ALB\_ROUTE\_TABLE, DB\_PRIVATE\_ROUTE\_TABLE, and two unassociated route tables.
- Network Connections (1):** icw\_CUSTOM\_VPC

The left sidebar shows the 'VPC dashboard' with various navigation options like Subnets, Route tables, Internet gateways, etc. The top navigation bar shows the account ID '3905-0378-1838' and the user 'Shubham Mishra'.

Step 10) We will create now security group for load balancer.

The screenshot shows the 'Create security group' page in the AWS console for the region 'ap-south-1'. The page is titled 'CreateSecurityGroup | EC2 | ap-south-1'. The 'Basic details' section is filled out as follows:

- Security group name:** ALB\_SG
- Description:** allows https request on alb
- VPC:** vpc-0cbb91d56e71b4fb7 (CUSTOM\_VPC)

The 'Inbound rules' section shows four rules being added:

Type	Protocol	Port range	Source	Description - optional	Action
HTTP	TCP	80	Anywhere...	0.0.0.0/0	Delete
HTTPS	TCP	443	Anywhere...	0.0.0.0/0	Delete
HTTP	TCP	80	Anywhere...	:::0	Delete
HTTPS	TCP	443	Anywhere...	:::0	Delete

An 'Add rule' button is visible at the bottom left of the rules section.

Step 11) We will create security group for ec2/ecs. Here we will allow http traffic from only loadbalancer sg and ssh only from my ip.

subnets | VPC Console x CreateSecurityGroup | EC2 | ap-south-1 x +

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#CreateSecurityGroup:

aws Elastic Container Service Account ID: 3905-0378-1838 Shubham Mishra

EC2 > Security Groups > Create security group

**Security group name** info

EC2\_SG

Name cannot be edited after creation.

**Description** info

allows access to ec2

**VPC** info

vpc-0cbb91d56e71b4fb7 (CUSTOM\_VPC)

**Inbound rules** info

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	Custom sg-088ebcd9afe76b8e9	
SSH	TCP	22	My IP 167.103.121.7/32	

Add rule

**Outbound rules** info

Step 12) We will create sg from database and allow port 3306 only from ec2/ecs security group.

Create task definition x CreateSecurityGroup x Secrets Manager | x Create database | x db\_subnets - SubnetDetails | VPC x +

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#CreateSecurityGroup:

aws Elastic Container Service Account ID: 3905-0378-1838 Shubham Mishra

EC2 > Security Groups > Create security group

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

**Basic details**

**Security group name** info

DB\_SG

Name cannot be edited after creation.

**Description** info

allows rds to be connected only through ec2

**VPC** info

vpc-0cbb91d56e71b4fb7 (CUSTOM\_VPC)

**Inbound rules** info

Type	Protocol	Port range	Source	Description - optional
MySQL/Aurora	TCP	3306	Custom sg-0c9111ea0f4bf55	

Add rule

**Outbound rules** info

Type	Protocol	Port range	Destination	Description - optional
All traffic	All	All	Custom 0.0.0.0/0	

Delete



Step 13) Also we will create NAT gateway to provide outbound internet access for resources in private subnet for ec2/ecs

NAT  
The name can be up to 256 characters long.

**Availability mode** Info  
Choose whether to deploy across all zones in the region or restrict to a single availability zone.

☒ **Regional - new**  
Scales automatically across all regional AZs, simplifying management for multi-AZ deployments.

☐ **Zonal**  
Provides granular control within a specific availability zone, adhering to subnet-level settings.

**VPC**  
Select a VPC in which to create the regional NAT gateway.

vpc-0cbb91d56e71b4fb7 (CUSTOM\_VPC)

**Connectivity type**  
Select a connectivity type for the NAT gateway.

☒ **Public**

☐ Private

**Method of Elastic IP (EIP) allocation** Info  
Choose how IP addresses are associated with NAT gateways.

☒ **Automatic**  
AWS automatically manages EIPs and AZ coverage for NAT gateways. This ensures easy scaling—adding AZs automatically allocates EIPs, simplifying management.

☐ **Manual**  
Manually assigns specific IP addresses for compliance or whitelisting. Note: Requires manual scaling to new AZs as workloads expand.

**Tags**  
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - optional
Q, Name	Q, NAT

[Add new tag](#) [Remove](#)

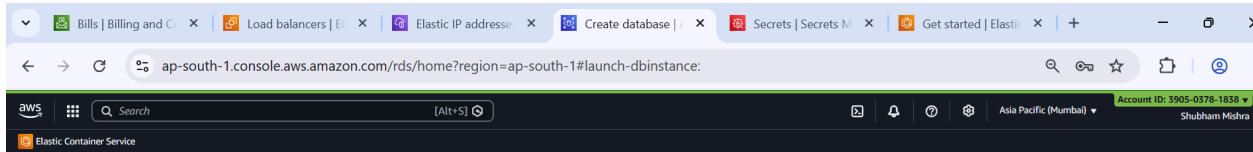
Step 14) We will add route to internet using nat gateway in private route table of app/ecs subnets.

**Edit routes**

Destination	Target	Status	Propagated	Route Origin
172.16.0.0/16	local	Active	No	CreateRouteTable
Q, 0.0.0.0/0	NAT Gateway	-	No	CreateRoute

[Add route](#) [Cancel](#) [Preview](#) [Save changes](#)

Step 15) Now we will create RDS database



## Create database [Info](#)

### Choose a database creation method

☒ Full configuration

You set all of the configuration options, including ones for availability, security, backups, and maintenance.

☐ Easy create

Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

### Engine options

#### Engine type [Info](#)

☐ Aurora (MySQL Compatible)



☐ Aurora (PostgreSQL Compatible)



☒ MySQL



☐ PostgreSQL



☐ MariaDB



☐ Oracle

ORACLE

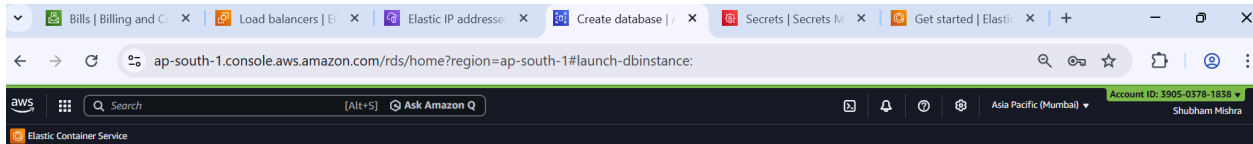
☐ Microsoft SQL Server



☐ IBM Db2

IBM Db2

Edition



### MySQL Community

#### Engine version [Info](#)

View the engine versions that support the following database features.

#### ▼ Hide filters

☐ Show only versions that support the Multi-AZ DB cluster [Info](#)

Create a Multi-AZ DB cluster with one primary DB instance and two readable standby DB instances. Multi-AZ DB clusters provide up to 2x faster transaction commit latency and automatic failover in typically under 35 seconds.

☐ Show only versions that support the Amazon RDS Optimized Writes [Info](#)

Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

#### Engine version

MySQL 8.0.43

☐ Enable RDS Extended Support [Info](#)

Amazon RDS Extended Support is a [paid offering](#). By selecting this option, you consent to being charged for this offering if you are running your database major version past the RDS end of standard support date for that version. Check the end of standard support date for your major version in the [RDS for MySQL documentation](#).

### Templates

Choose a sample template to meet your use case.

☐ Production

Use defaults for high availability and fast, consistent performance.

☐ Dev/Test

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

☒ Sandbox

To develop new applications, test existing applications, or gain hands-on experience with Amazon RDS.

ap-south-1.console.aws.amazon.com/rds/home?region=ap-south-1#launch-dbinstance:

aws

Search

[Alt+S] Ask Amazon Q

Asia Pacific (Mumbai)

Account ID: 3905-0378-1838

Shubham Mishra

Elastic Container Service

Aurora and RDS > Databases > Create database

Settings

DB instance identifier [Info](#)

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

cloudzenia-database

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

▼ Credentials Settings

Master username [Info](#)

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

admin

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

Credentials management

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

☐ Managed in AWS Secrets Manager - most secure

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

☒ Self managed

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

☐ Auto generate password

Amazon RDS can generate a password for you, or you can specify your own password.

Master password [Info](#)

ap-south-1.console.aws.amazon.com/rds/home?region=ap-south-1#launch-dbinstance:

aws

Search

[Alt+S] Ask Amazon Q

Asia Pacific (Mumbai)

Account ID: 3905-0378-1838

Shubham Mishra

Elastic Container Service

Aurora and RDS > Databases > Create database

DB instance class [Info](#)

▼ Hide filters

☐ Show instance classes that support Amazon RDS Optimized Writes [Info](#)

Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

☐ Include previous generation classes

☐ Standard classes (includes m classes)

☐ Memory optimized classes (includes r and x classes)

☒ Burstable classes (includes t classes)

db.t3.micro

2 vCPUs 1 GiB RAM EBS Bandwidth: Up to 2,085 Mbps Network: Up to 5 Gbps

Storage

Storage type [Info](#)

Provisioned IOPS SSD (io2) storage volumes are now available.

General Purpose SSD (gp2)

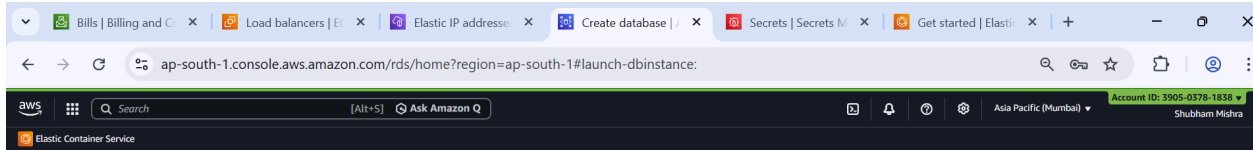
Baseline performance determined by volume size

Allocated storage [Info](#)

20

GiB

Allocated storage value must be 20 GiB to 6,144 GiB



**Connectivity Info**

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

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

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

**Virtual private cloud (VPC)**  
Choose the VPC. The VPC defines the virtual networking environment for this DB instance.

CUSTOM\_VPC (vpc-0cbb91d56e71b4fb7)  
6 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

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

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

db\_subnets  
2 Subnets, 2 Availability Zones

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

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

ap-south-1.console.aws.amazon.com/rds/home?region=ap-south-1#databases:

**Aurora and RDS**

**Databases (1)**

Filter by databases

cloudzenia-database

**Connection details to your database cloudzenia-database**

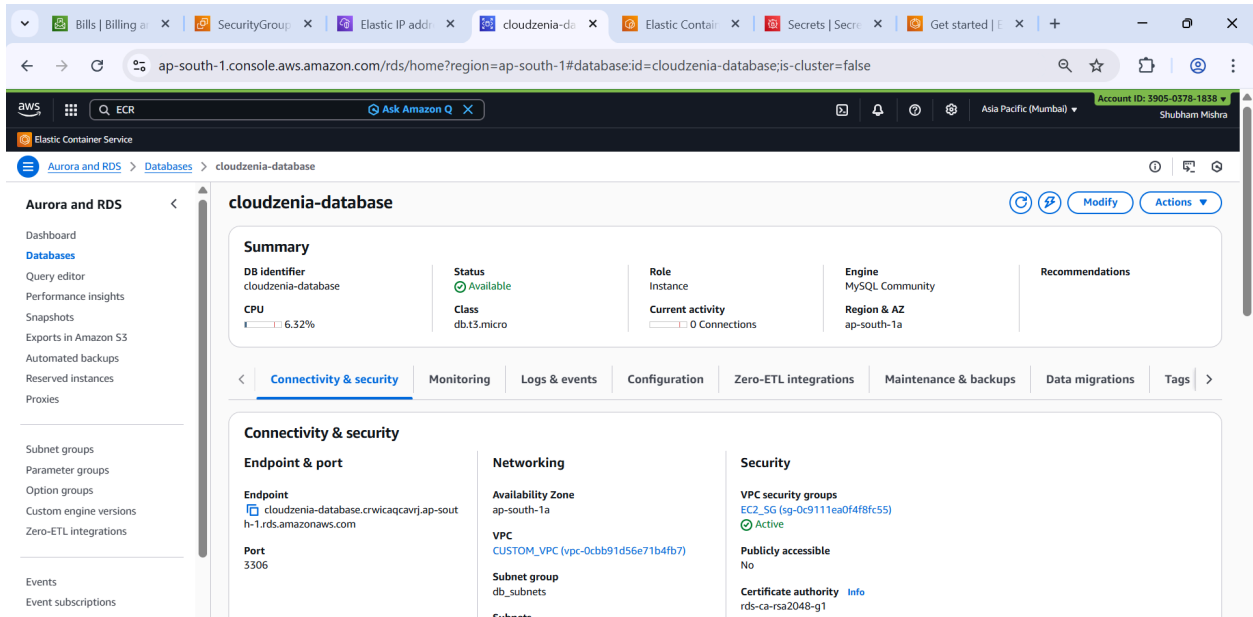
This is the only time you can view this password. Copy and save the password for your reference. If you lose the password, you must modify your database to change it. You can use a SQL client application or utility to connect to your database.

[Learn about connecting to your database](#)

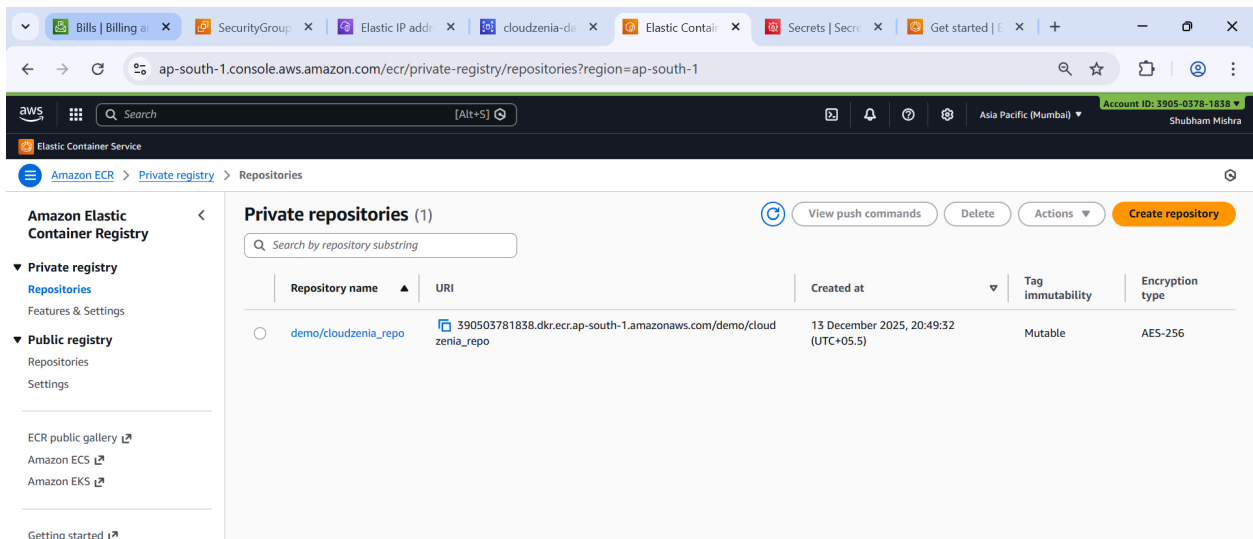
**Master username**  
admin

**Master password**  
\*\*\*\*\*

Close



Step 16) Now we will create ECR repository to store images and pull images in ECS



Step 17) Now we will create files for docker image to store in ecr

a) package.json

```
{
  "name": "cloudzenia-microservice",
  "version": "1.0.0",
  "description": "CloudZenia ECS microservice",
  "main": "app.js",
  "scripts": {
    "start": "node app.js"
  },
  "dependencies": {
    "express": "^4.18.2"
  }
}
```

b) app.js

```
const express = require("express");
const app = express();

app.get("/", (req, res) => {
  res.send("Hello from Microservice");
});

app.get("/health", (req, res) => {
  res.status(200).send("OK");
});

const port = process.env.PORT || 3000;
app.listen(port, () => {
  console.log(`Microservice running on port ${port}`);
});
```

c) Dockerfile

```
# Use official lightweight Node image
FROM node:18-alpine

# Create app directory
WORKDIR /app

# Copy dependency files first (best practice)
COPY package.json ./

# Install dependencies
RUN npm install --only=production

# Copy application code
COPY app.js ./

# Expose application port
EXPOSE 3000

# Start the application
CMD ["npm", "start"]
```

```
root@shubham-Inspiron-14-3467:/home/shubham/microservice# ls
app.js  Dockerfile  package.json
root@shubham-Inspiron-14-3467:/home/shubham/microservice#
```

```
root@shubham-Inspiron-14-3467:/home/shubham/microservice# ls
app.js  Dockerfile  package.json
root@shubham-Inspiron-14-3467:/home/shubham/microservice# docker build -t cloudzenia-microservice .
```

```
root@shubham-Inspiron-14-3467:/home/shubham/microservice# docker image ls
REPOSITORY              TAG          IMAGE ID       CREATED        SIZE
cloudzenia-microservice latest       48b1d34b3eb9   About a minute ago  135MB
nginx                   latest       60adc2e137e7   3 weeks ago    152MB
node                    18-alpine   ee77c6cd7c18   8 months ago    127MB
root@shubham-Inspiron-14-3467:/home/shubham/microservice#
```

```
root@shubham-Inspiron-14-3467:/home/shubham/microservice# aws ecr get-login-password --region ap-south-1 \
| docker login --username AWS --password-stdin 390503781838.dkr.ecr.ap-south-1.amazonaws.com

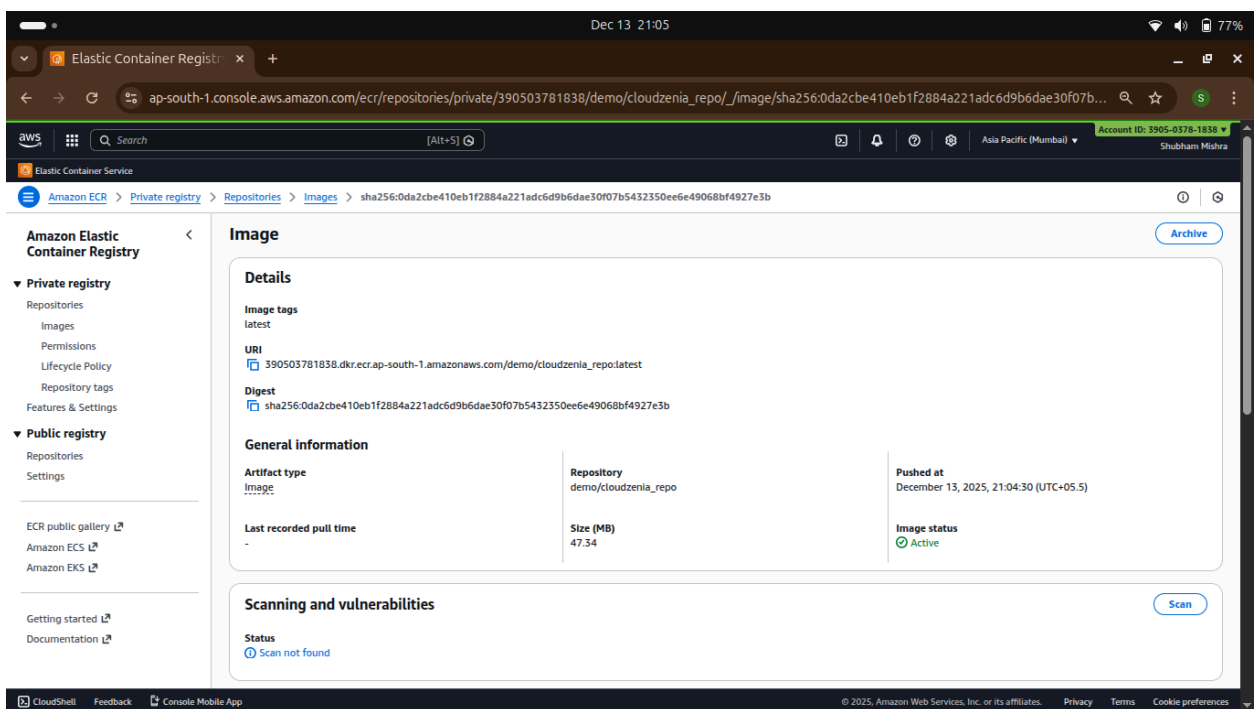
WARNING! Your credentials are stored unencrypted in '/root/.docker/config.json'.
Configure a credential helper to remove this warning. See
https://docs.docker.com/go/credential-store/

Login Succeeded
root@shubham-Inspiron-14-3467:/home/shubham/microservice#
```



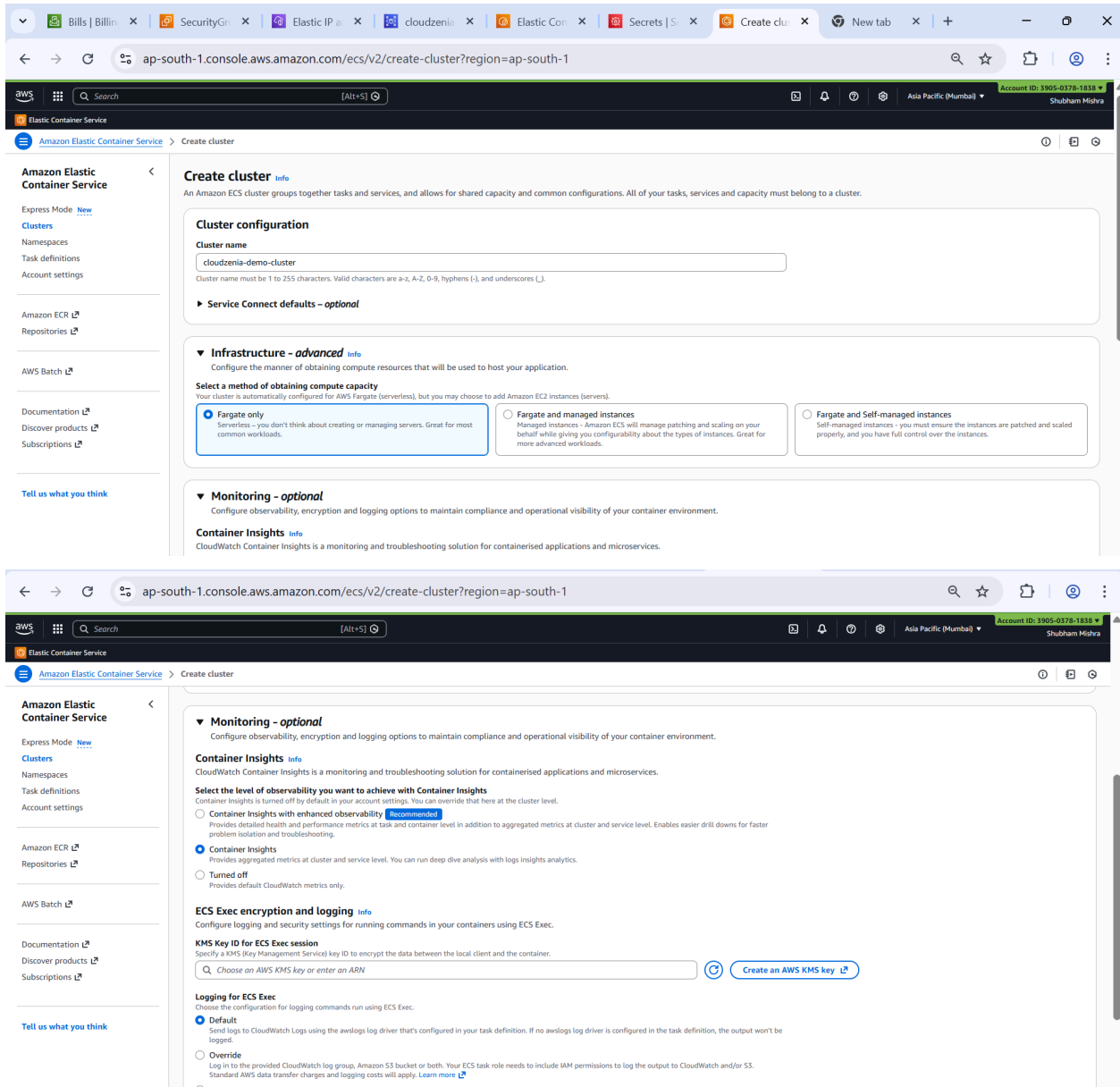
```
root@shubham-Inspiron-14-3467: /home/shubham/microservice# docker tag cloudzenia-microservice:latest 390503781838.dkr.ecr.ap-south-1.amazonaws.com/demo/cloudzenia_repo:latest
```

```
root@shubham-Inspiron-14-3467: /home/shubham/microservice# docker push 390503781838.dkr.ecr.ap-south-1.amazonaws.com/demo/cloudzenia_repo:latest
The push refers to repository [390503781838.dkr.ecr.ap-south-1.amazonaws.com/demo/cloudzenia_repo]
3285b338d2fd: Pushed
2aa3f572510a: Pushed
b19bb3e44a10: Pushed
a43e1cf4c16d: Pushed
02140d9a70a7: Pushed
f3b40b0cdb1c: Pushed
0b1f26057bd0: Pushed
08000c18d16d: Pushed
latest: digest: sha256:0da2cbe410eb1f2884a221adc6d9b6dae30f07b5432350ee6e49068bf4927e3b size: 1989
root@shubham-Inspiron-14-3467: /home/shubham/microservice#
```

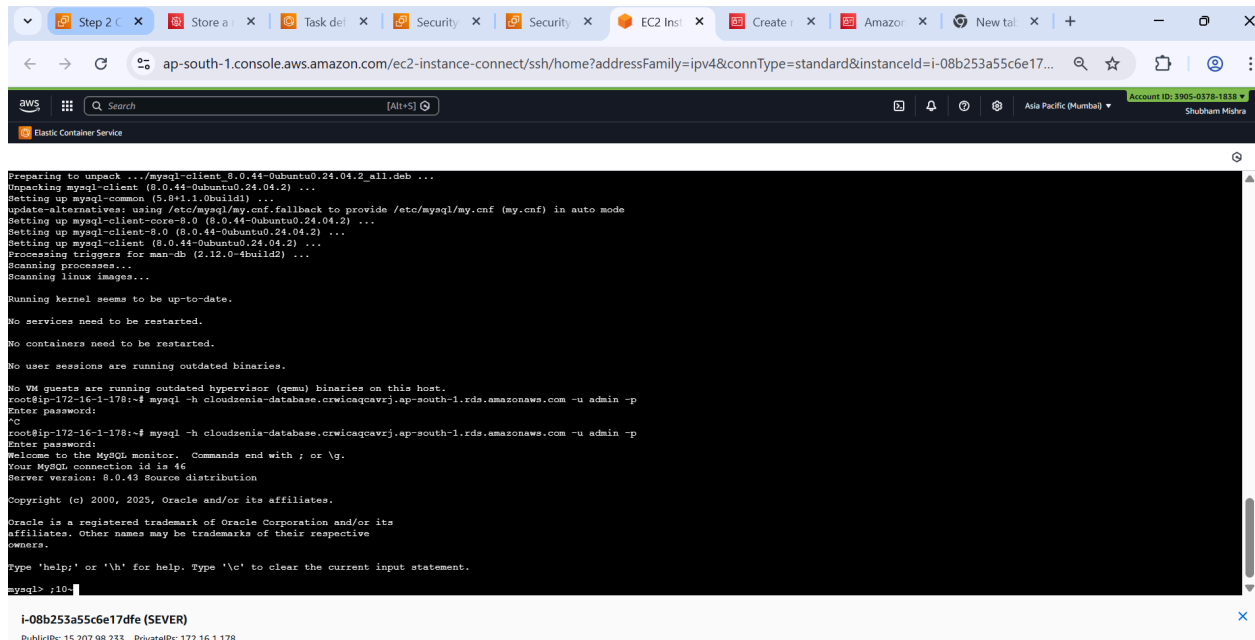


Microservice image in ECR

Step 18) Now we will create ECS cluster



**Step 19) Now we will launch any ec2 and install mysql so that we can create database for wordpress**



```
Preparing to unpack .../mysql-client-8.0.44-0ubuntu0.24.04.2_all.deb ...
Unpacking mysql-client (8.0.44-0ubuntu0.24.04.2) ...
Setting up mysql-common (5.8+1.1.0build1) ...
update-alternatives: using /etc/mysql/my.cnf.fallback to provide /etc/mysql/my.cnf (my.cnf) in auto mode
Setting up mysql-client-core-8.0 (8.0.44-0ubuntu0.24.04.2) ...
Setting up mysql-client-8.0 (8.0.44-0ubuntu0.24.04.2) ...
Setting up mysql-client (8.0.44-0ubuntu0.24.04.2) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-16-1-178:~# mysql -h cloudsenia-database.cwicaqevrj.ap-south-1.rds.amazonaws.com -u admin -p
Enter password:
^C
root@ip-172-16-1-178:~# mysql -h cloudsenia-database.cwicaqevrj.ap-south-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 46
Server version: 8.0.43 Source distribution

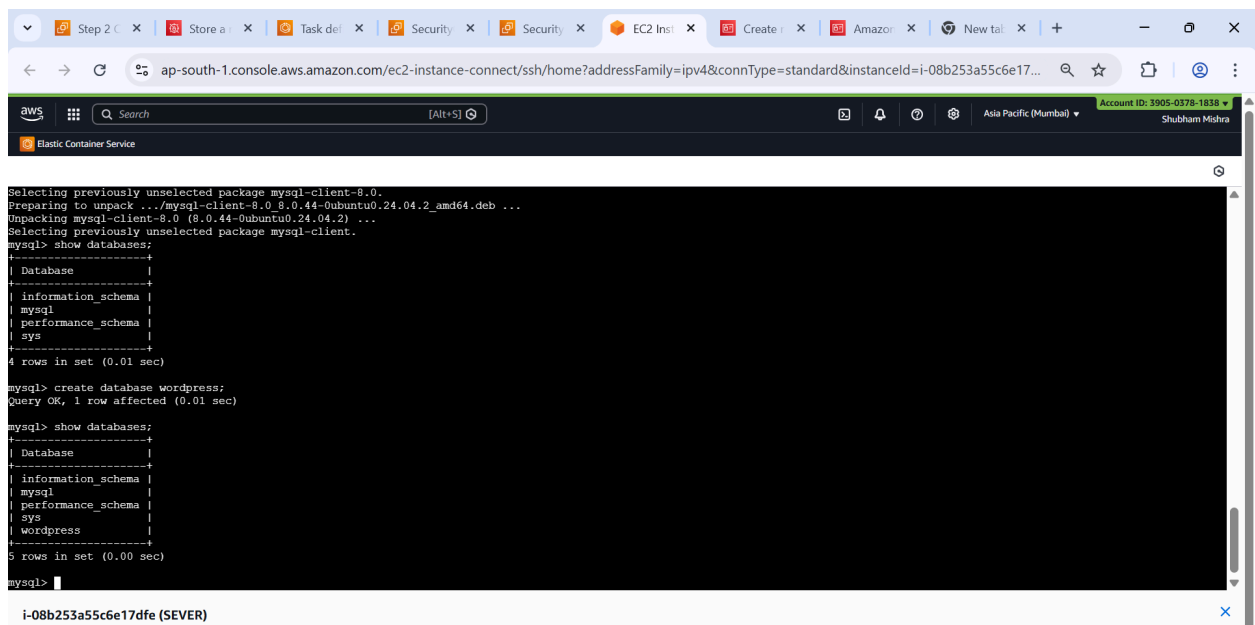
Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ;!0
```

i-08b253a55c6e17dfe (SEVER)



```
Selecting previously unselected package mysql-client-8.0.
Preparing to unpack .../mysql-client-8.0_8.0.44-0ubuntu0.24.04.2_amd64.deb ...
Unpacking mysql-client-8.0 (8.0.44-0ubuntu0.24.04.2) ...
Selecting previously unselected package mysql-client.
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.01 sec)

mysql> create database wordpress;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| wordpress |
+-----+
5 rows in set (0.00 sec)

mysql>
```

i-08b253a55c6e17dfe (SEVER)

**Step 20) Now we will store db secrets in secret manager**

Step 2 Create target group | EC2 | x Store a new secret | Secrets Manager | ap-south-1 | x Task definitions | Elastic Container Service | x SecurityGroup | EC2 | ap-south-1 | x +

ap-south-1.console.aws.amazon.com/secretsmanager/newsecret?region=ap-south-1

aws [Search] [Alt+S] Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

Elastic Container Service

AWS Secrets Manager > Secrets > Store a new secret

Step 1: Choose secret type

Step 2: Configure secret

Step 3 - optional: Configure rotation

Step 4: Review

### Choose secret type

**Secret type** [Info](#)

☐ Credentials for Amazon RDS database
 ☐ Credentials for Amazon DocumentDB database
 ☐ Credentials for Amazon Redshift data warehouse
 ☐ Credentials for other database
 ☐ Managed external secret  
Secrets vended by your third-party software vendor.
 ☒ **Other type of secret**  
API key, OAuth token, other.

**Key/value pairs** [Info](#)

**Key/value** **Plaintext**

DB_HOST	cloudzenia-database.crwicqacavj.ap-south-1.rds.amazonaws.com	<a href="#">Remove</a>
DB_PORT	3306	<a href="#">Remove</a>
DB_NAME	wordpress	<a href="#">Remove</a>
DB_USER	admin	<a href="#">Remove</a>

## Step 21) Now we will create iam role for ecs task definition

a) ECS task execution role(used by ECS control plane)

Step 2 Create target group | EC2 | x Secrets | Secrets Manager | ap-south-1 | x Task definitions | Elastic Container Service | x ecsTaskExecutionRole | IAM | Global | x +

us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/roles/details/ecsTaskExecutionRole

aws [Search] [Alt+S] Global Account ID: 3905-0378-1838 Shubham Mishra

Elastic Container Service

IAM > Roles > ecsTaskExecutionRole

### ecsTaskExecutionRole

[Delete](#) [Edit](#)

**Summary**

Creation date: December 11, 2025, 12:27 (UTC+05:30)

Last activity: 2 days ago

ARN: [arn:aws:iam::390503781838:role/ecsTaskExecutionRole](#)

Maximum session duration: 1 hour

**Permissions** [Trust relationships](#) [Tags](#) [Last Accessed](#) [Revoke sessions](#)

**Permissions policies (1)** [Info](#) [Simulate](#) [Remove](#) [Add permissions](#)

You can attach up to 10 managed policies.

**Filter by Type** [All types](#)

Policy name	Type	Attached entities
<a href="#">AmazonECSTaskExecutionRolePolicy</a>	AWS managed	1

**AmazonECSTaskExecutionRolePolicy** [Copy JSON](#)

## b) ECS task role ( used by application inside containers)

The screenshot shows the AWS IAM console page for the **ecsTaskRole**. The left sidebar contains navigation links for Identity and Access Management (IAM), Access management, and Access reports. The main content area displays the role's summary, including its creation date (December 13, 2025), last activity, ARN, and maximum session duration (1 hour). Below the summary, there are tabs for Permissions, Trust relationships, Tags, Last Accessed, and Revoke sessions. The **Permissions** tab is active, showing a list of permissions policies. One policy, **AllowWordPressDBSecretRead**, is listed with a type of **Customer inline** and 0 attached entities. Buttons for **Simulate**, **Remove**, and **Add permissions** are visible.

## Step 22) Now we will create task definition for ecs cluster

### a) Task definition for wordpress

The screenshot shows the AWS ECS console page for creating a new task definition. The left sidebar contains navigation links for Amazon Elastic Container Service, Express Mode, Clusters, Namespaces, Task definitions, Account settings, Amazon ECR, Repositories, AWS Batch, Documentation, Discover products, and Subscriptions. The main content area displays the **Create new task definition** page. The **Task definition configuration** section shows the **Task definition family** as **wordpress\_task\_definition**. The **Infrastructure requirements** section shows the **Launch type** as **AWS Fargate** (selected). Below this, there are options for **Managed instances - new** and **Amazon EC2 instances**. The **OS Architecture** and **Network mode** sections are partially visible at the bottom.

Step 2 Create target group | EC2 | Secrets | Secrets Manager | ap-south-1 | Create task definition | Elastic Container Service | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ecs/v2/create-task-definition?region=ap-south-1

aws rds Ask Amazon Q

Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

Amazon Elastic Container Service

Create new task definition

Self-managed infrastructure using Amazon EC2 instances.

OS, Architecture, Network mode

Network mode is used for tasks and is dependent on the compute type selected.

Operating system/Architecture Linux/X86\_64

Network mode awsvpc

Task size

Specify the amount of CPU and memory to reserve for your task.

CPU 1 vCPU

Memory 2 GB

Task roles - conditional

Task role

A task IAM role allows containers in the task to make API requests to AWS services. You can create a task IAM role from the IAM console.

ecsTaskRole

Task execution role

A task execution IAM role is used by the container agent to make AWS API requests on your behalf. If you don't already have a task execution IAM role created, we can create one for you.

ecsTaskExecutionRole

Task placement - optional

Fault injection - optional

Step 2 Create target group | EC2 | Secrets | Secrets Manager | ap-south-1 | Create task definition | Elastic Container Service | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ecs/v2/create-task-definition?region=ap-south-1

aws rds Ask Amazon Q

Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

Amazon Elastic Container Service

Create new task definition

Fault injection - optional

Container - 1

Essential container

Remove

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 wordpress

Essential container Yes

Image URI wordpress:latest

Browse ECR images

Private registry

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

Private registry authentication

Port mappings

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 80

Protocol TCP

Port name container-port-protocol

App protocol HTTP

Remove

Step 2 Create target group | EC... | cloudzenia/wp/db | Secrets Ma... | Create task definition | Elastic C... | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ecs/v2/create-task-definition?region=ap-south-1

Amazon Elastic Container Service

Create new task definition

**Environment variables - optional**

Environment variables | Info

Add individually

Add environment variables using plain text values or secrets from AWS Secrets Manager or Parameter Store.

Key	Value type	Value	
WORDPRESS_DB_HOST	ValueFrom	arn:aws:secretsmanager:...	Remove
WORDPRESS_DB_NAME	ValueFrom	arn:aws:secretsmanager:...	Remove
WORDPRESS_DB_USER	ValueFrom	arn:aws:secretsmanager:...	Remove
WORDPRESS_DB_PASSW	ValueFrom	arn:aws:secretsmanager:...	Remove

[Add environment variable](#)

**Add from file**

Add environment variables in bulk by providing an environment file hosted on Amazon S3.

## b) Task definition for nodejs microservice

Step 2 Create target group | EC... | cloudzenia/wp/db | Secrets Ma... | Create task definition | Elastic C... | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ecs/v2/create-task-definition?region=ap-south-1

Amazon Elastic Container Service

Create new task definition

**Task definition configuration**

Task definition family | Info

Specify a unique task definition family name.

microservice\_task\_definition

Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

**Infrastructure requirements** | Info

Specify the infrastructure requirements for the task definition.

**Launch type** | Info

Selection of the launch type will change task definition parameters.

☒ **AWS Fargate**

Serverless compute for containers.

☐ **Managed instances - new**

Use if you have specific hardware constraints, such as GPU accelerators, CPU instruction sets, or network-optimized hardware (offloads scaling, patching, and instance management to AWS).

☐ **Amazon EC2 instances**

Self-managed infrastructure using Amazon EC2 instances.

**OS, Architecture, Network mode**

Network mode is used for tasks and is dependent on the compute type selected.

**Operating system/Architecture** | Info

Linux/x86\_64

**Network mode** | Info

awsipc

**Task size** | Info

Specify the amount of CPU and memory to reserve for your task.

**CPU**

1 vCPU

**Memory**

3 GB

Step 2 Create target group | EC2 | cloudzenia/wp/db | Secrets Manager | Create task definition | Elastic Container Service | IAM | Global

ap-south-1.console.aws.amazon.com/ecs/v2/create-task-definition?region=ap-south-1

Amazon Elastic Container Service

Express Mode [New](#)

Clusters

Namespaces

**Task definitions**

Account settings

Amazon ECR [↗](#)

Repositories [↗](#)

AWS Batch [↗](#)

Documentation [↗](#)

Discover products [↗](#)

Subscriptions [↗](#)

Tell us what you think

**Task execution role** [Info](#)

A task execution IAM role is used by the container agent to make AWS API requests on your behalf. If you don't already have a task execution IAM role created, we can create one for you.

ecsTaskExecutionRole

► Task placement - optional

► Fault injection - optional

▼ **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**

microservice

**Essential container**

Yes

Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

**Image URI**

390503781838.dkr.ecr.ap-south-1.amazonaws.com/demo/cloudzenia\_repo@sha256:0da2cbe410eb1f2884a221adc6d9b6dae30f07b5432350ee6e49068b

[Browse ECR images](#)

Up to 255 letters (uppercase and lowercase), numbers, hyphens, underscores, colons, periods, forward slashes, and number signs are allowed.

**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
3000	TCP	container-port-protocol	HTTP

[Remove](#)

[Add port mapping](#)

**Read-only root file system** [Info](#)

When this parameter is turned on, the container is given read-only access to its root file system.

☐ Read only

**Resource allocation limits - conditional** [Info](#)

Container-level CPU, GPU and memory limits are different from task-level values. They define how many resources are allocated for the container. If the container attempts to exceed the memory specified by the hard limit, the container is terminated.

CPU	GPU	Memory hard limit	Memory soft limit
0.5	1	1	1

in vCPU in GB in GB in GB

**Step 23) Now we will create two Target groups for our services like TG\_MICROSERVICE and TG\_WORDPRESS**



### Create target group

A target group can be made up of one or more targets. Your load balancer routes requests to the targets in a target group and performs health checks on the targets.

**Settings - immutable**  
Choose a target type and the load balancer and listener will route traffic to your target. These settings can't be modified after target group creation.

**Target type**  
Indicate what resource type you want to target. Only the selected resource type can be registered to this target group.

☐ **Instances**  
Supports load balancing to instances in a VPC. Integrate with Auto Scaling Groups or ECS services for automatic management.  
Suitable for: **ALB** **NLB** **GWLB**

☐ **Lambda function**  
Supports load balancing to a single Lambda function. ALB required as traffic source.  
Suitable for: **ALB**

☒ **IP addresses**  
Supports load balancing to VPC and on-premises resources. Facilitates routing to IP addresses and network interfaces on the same instance. Supports IPv6 targets.  
Suitable for: **ALB** **NLB** **GWLB**

☐ **Application Load Balancer**  
Allows use of static IP addresses and PrivateLink with an Application Load Balancer. NLB required as traffic source.  
Suitable for: **NLB**

**Target group name**  
Name must be unique per Region per AWS account.  
  
Accepts: a-z, A-Z, 0-9, and hyphen (-). Can't begin or end with hyphen. 1-32 total characters; Count: 15/32

**Target group name**  
Name must be unique per Region per AWS account.  
  
Accepts: a-z, A-Z, 0-9, and hyphen (-). Can't begin or end with hyphen. 1-32 total characters; Count: 15/32

**Protocol**  
Protocol for communication between the load balancer and targets.

**Port**  
Port number where targets receive traffic. Can be overridden for individual targets during registration.  
  
1-65535

**IP address type**  
Only targets with the indicated IP address type can be registered to this target group.  
☒ **IPv4**  
☐ **IPv6**

**VPC**  
Select the VPC that hosts the load balancer. Only VPCs that support the IP address type selected above are available in this list. On the **Register targets** page, you can register IP addresses from this VPC, or from private IP addresses located outside of this load balancer's VPC (such as a peered VPC, EC2-Classic, or on-premises targets that are reachable over Direct Connect or VPN).  
  
172.16.0.0/16

**Protocol version**  
☒ **HTTP1**  
Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.  
☐ **HTTP2**  
Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.  
☐ **gRPC**  
Send requests to targets using gRPC. Supported when the request protocol is gRPC.

[Create VPC](#)



Step 2 Create target group | EC2

cloudzenia/wp/db | Secrets Manager | Task definitions | Elastic Container Service | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#CreateTargetGroup:

Register targets - *recommended*

This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

**IP addresses**

**Step 1: Choose a network**

You can add IP addresses from the VPC selected for your target group or from outside the VPC. Note that you can assemble a mix of targets from multiple network sources by returning to this step and choosing another network.

**Network**

CUSTOM\_VPC  
vpc-0c2b91d56e71b4fb7  
IPv4 VPC CIDR: 172.16.0.0/16

**Step 2: Specify IPs and define ports**

You can manually enter IP addresses from the selected network.

Enter an IPv4 address from a VPC subnet.

[Add IPv4 address](#)

You can add up to 5 more IP addresses.

**Ports**

Ports for routing to this target.

1-65535 (separate multiple ports with comma)

[Include as pending below](#)

For ECS Fargate, targets are registered automatically by the ECS service, so the target group is initially empty.

## STEP 24) Now we will create application loadbalancer

Create application load balancer

cloudzenia/wp/db | Secrets Manager | Task definitions | Elastic Container Service | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#CreateALBWizard:

Create application load balancer

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.

CLOUDZENIA-ALB

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

**Scheme**

Internet-facing (selected)  
• Serves Internet-facing traffic.  
• Has public IP addresses.  
• DNS name resolves to public IP.  
• Requires a public subnet.

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

**Load balancer IP address type**

IPv4 (selected)  
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.

**Network mapping**

**VPC**

The load balancer will exist and scale within the selected VPC. The selected VPC is also where the load balancer targets must be hosted unless routing to Lambda or on-premises targets, or if using VPC peering. To confirm the VPC for your targets, view [target groups](#).

vpc-0c2b91d56e71b4fb7 (CUSTOM\_VPC)  
172.16.0.0/16

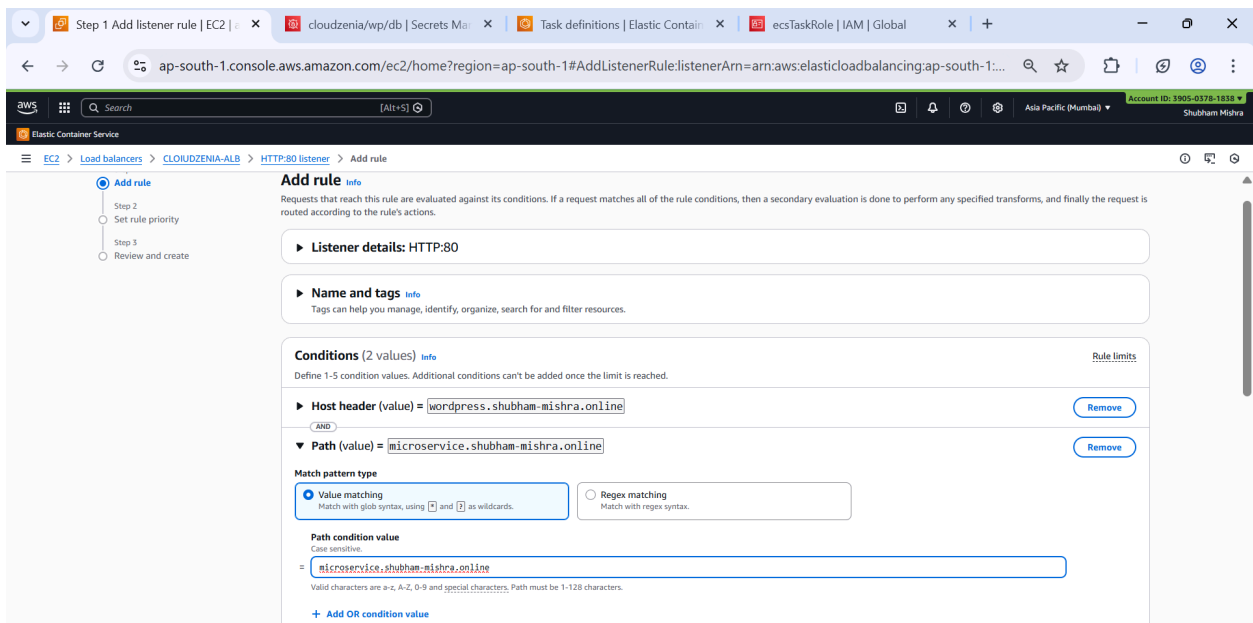
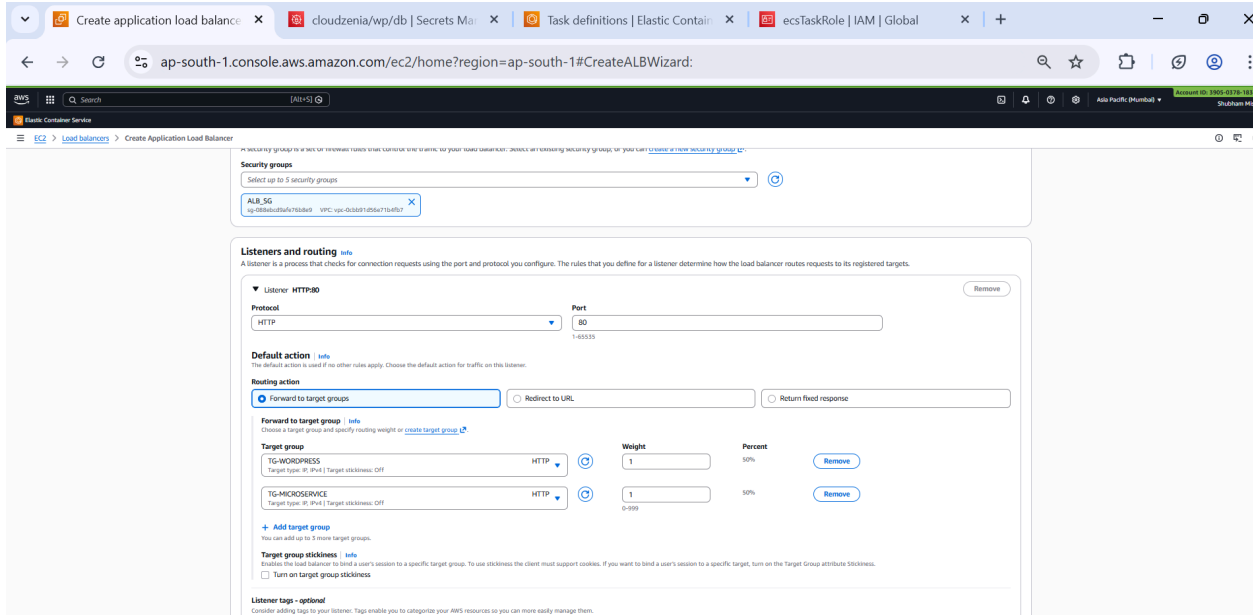
[Create VPC](#)

**IP pools**

You can optionally choose to configure an IPAM pool as the preferred source for your load balancers IP addresses. Create or view pools in the [Amazon VPC IP Address Manager console](#).

☐ Use IPAM pool for public IPv4 addresses  
The IPAM pool you choose will be the preferred source of public IPv4 addresses. If the pool is depleted IPv4 addresses will be assigned by AWS.

**Availability Zones and subnets**



Step 1 Add listener rule | EC2

cloudzenia/wp/db | Secrets Manager | Task definitions | Elastic Container Service | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#AddListenerRule:listenerArn=arn:aws:elasticloadbalancing:ap-south-1:...

Conditions (1 value)

Define 1-5 condition values. Additional conditions can't be added once the limit is reached.

Host header (value) = microservice.shubham-mishra.online

Match pattern type

Value matching (selected) Regex matching

Host header condition value

microservice.shubham-mishra.online

Transforms - optional (0)

Modify incoming request host headers and URLs before routing.

Actions

Requests matching all rule conditions route according to the rule actions.

Routing action

Forward to target groups (selected) Redirect to URL Return fixed response

Forward to target group

Target group

TG-MICROSERVICE

Weight

1

Percent

100%

**Step 25) lets create subdomain for our microservice and wordpress ecs tasks in different account as i already have domain over there**

shubham-mishra.online - add record

us-east-1.console.aws.amazon.com/route53/v2/hostedzones?region=ap-south-1#CreateRecordSet/Z00138753DLXEPY1BYND

Quick create record

Record 1

Record name

wordpress

Record type

CNAME - Routes traffic to another domain name and to some AWS resources

Value

CLOUDZENIA-ALB-451687647-ap-south-1.elb.amazonaws.com

TTL (seconds)

300

Routing policy

Simple routing

Record 2

Record name

microservice

Record type

CNAME - Routes traffic to another domain name and to some AWS resources

Value

CLOUDZENIA-ALB-451687647-ap-south-1.elb.amazonaws.com

TTL (seconds)

300

Routing policy

Simple routing

**Step 26) Now we will create service for our ECS cluster**

Listener details | EC2 | ap-south-1 | cloudzenia/wp/db | Secrets Manager | Create service | Elastic Container Service | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ecs/v2/clusters/cloudzenia-demo-cluster/create-service?region=ap-south-1

Account ID: 3905-0378-1838 | Asia Pacific (Mumbai) | Shubham Mishra

Amazon Elastic Container Service

Express Mode [New](#)

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR [↗](#)

Repositories [↗](#)

AWS Batch [↗](#)

Documentation [↗](#)

Discover products [↗](#)

Subscriptions [↗](#)

Tell us what you think

Create service [Info](#)

Service details

Task definition family

Select an existing task definition. To create a new task definition, go to [Task definitions](#).

microservice\_task\_definition

Task definition revision [Latest](#)

Select the task definition revision from the 100 most recent entries, or enter a revision. Leave the field blank to use the latest revision.

1

Service name

Assign a service name that is unique for this cluster.

microservice\_task\_definition-service-h9awfznj

Up to 255 letters (uppercase and lowercase), numbers, underscores and hyphens are allowed. Service names must be unique within a cluster.

Environment

Existing cluster

cloudzenia-demo-cluster

▼ Compute configuration - advanced

Compute options [Info](#)

To ensure task distribution across your compute types, use appropriate compute options.

☐ Capacity provider strategy

Specify a launch strategy to distribute your tasks across one or more capacity providers.

☒ Launch type

Launch tasks directly without the use of a capacity provider strategy.

Listener details | EC2 | ap-south-1 | cloudzenia/wp/db | Secrets Manager | Create service | Elastic Container Service | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ecs/v2/clusters/cloudzenia-demo-cluster/create-service?region=ap-south-1

Account ID: 3905-0378-1838 | Asia Pacific (Mumbai) | Shubham Mishra

Amazon Elastic Container Service

Express Mode [New](#)

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR [↗](#)

Repositories [↗](#)

AWS Batch [↗](#)

Documentation [↗](#)

Discover products [↗](#)

Subscriptions [↗](#)

Tell us what you think

Platform version [Info](#)

Specify the platform version on which to run your service.

LATEST

► Troubleshooting configuration - recommended

Deployment configuration

Scheduling strategy [Info](#)

☒ 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

Availability Zone re-balancing [Info](#)

☒ Turn on Availability Zone re-balancing

Amazon ECS automatically detects Availability Zone imbalances in task distributions across an ECS service, and evenly redistributes ECS service tasks across Availability Zones.

Health check grace period [Info](#)

seconds

60

► Deployment options

► Deployment failure detection [Info](#)

Update service | Elastic Container Service | ap-south-1.console.aws.amazon.com/ecs/v2/clusters/cloudzenia-demo-cluster/services/microservice\_task\_definition-service-h9awfznj/update?re...

Amazon Elastic Container Service

Express Mode [New](#)

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR [↗](#)

Repositories [↗](#)

AWS Batch [↗](#)

Documentation [↗](#)

Discover products [↗](#)

Subscriptions [↗](#)

Tell us what you think

You can now configure predictive scaling for your ECS services by using the service auto scaling section on the [Service detail page](#). This dedicated section enables you to configure all types of scaling policies, set up scheduled scaling actions, and track scaling activities. [Find out more](#)

☒ Use service auto scaling

Configure service auto scaling to adjust your service's desired count.

**Minimum number of tasks**

This lower boundary to which service auto scaling can adjust the desired count of the service.

1

**Maximum number of tasks**

This upper boundary to which service auto scaling can adjust the desired count of the service.

2

**Scaling policy**

[Scaling policy type](#) | [Info](#)

Create either a target tracking or step scaling policy.

☒ **Target tracking**

Increase or decrease the number of tasks that your service runs based on a target value for a specific metric.

☐ **Step scaling**

Increase or decrease the number of tasks that your service runs based on a set of existing adjustments, known as step adjustments, that vary based on the size of the alarm breach.

**Policy name**

cpu\_utilization

**ECS service metric**

ECSAverageCPUUtilization

**Target value**

70

**Scale-out cooldown period**

60

**Scale-in cooldown period**

180

☐ Turn off scale-in

[+ Add more scaling policies](#)

[Remove](#)

Listener details | EC2 | ap-south-1 | cloudzenia/wp/db | Secrets Manager | Create service | Elastic Container Service | ecsTaskRole | IAM | Global

ap-south-1.console.aws.amazon.com/ecs/v2/clusters/cloudzenia-demo-cluster/create-service?region=ap-south-1

Amazon Elastic Container Service

Express Mode [New](#)

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR [↗](#)

Repositories [↗](#)

AWS Batch [↗](#)

Documentation [↗](#)

Discover products [↗](#)

Subscriptions [↗](#)

Tell us what you think

**Networking**

**VPC** | [Info](#)

Select a VPC to use for your Amazon ECS resources.

vpc-0cbb91d56e71b4fb7

[Create a new VPC](#)

**Subnets**

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

Choose subnets

subnets-0eaf843f693c4053

subnets-001f00096d2d1f6a4

[Clear current selection](#)

**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 name**

Choose an existing security group.

Choose security groups

sg-072bef40da7aa4b4a

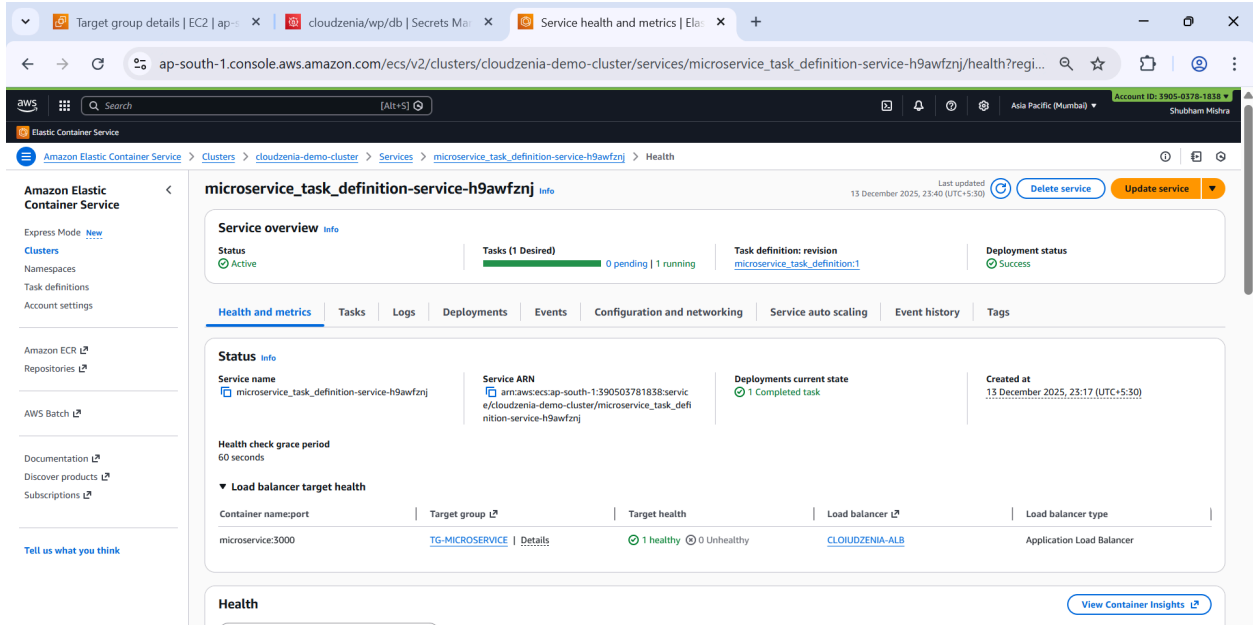
**Public IP** | [Info](#)

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

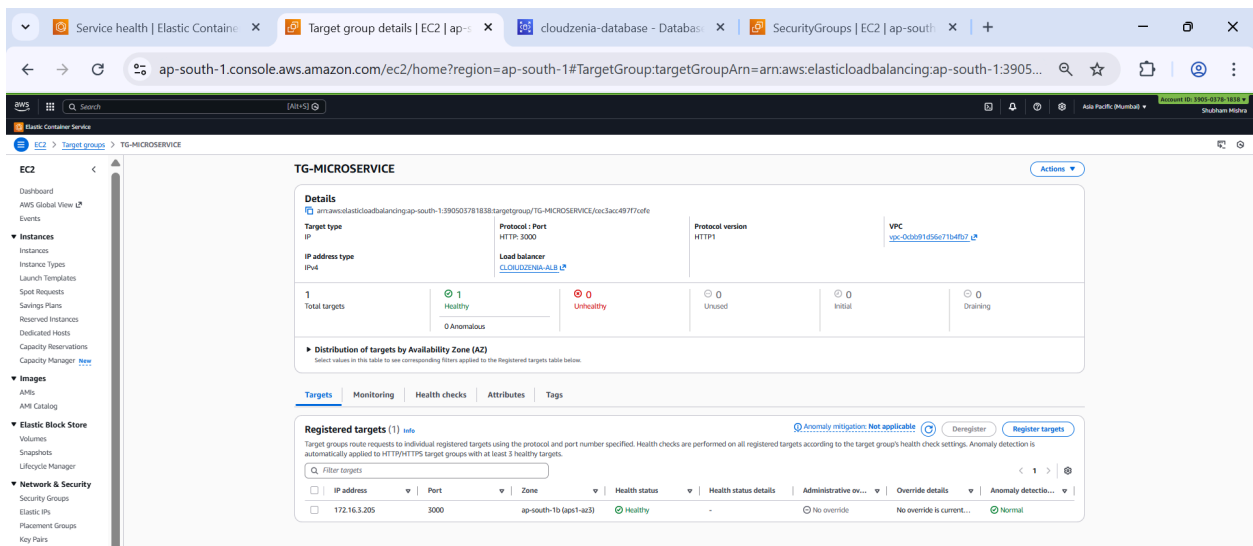
☒ Turned off

**Service Connect - optional** | [Info](#)

Service Connect allows for service-to-service communications with automatic discovery using short names and standard ports.



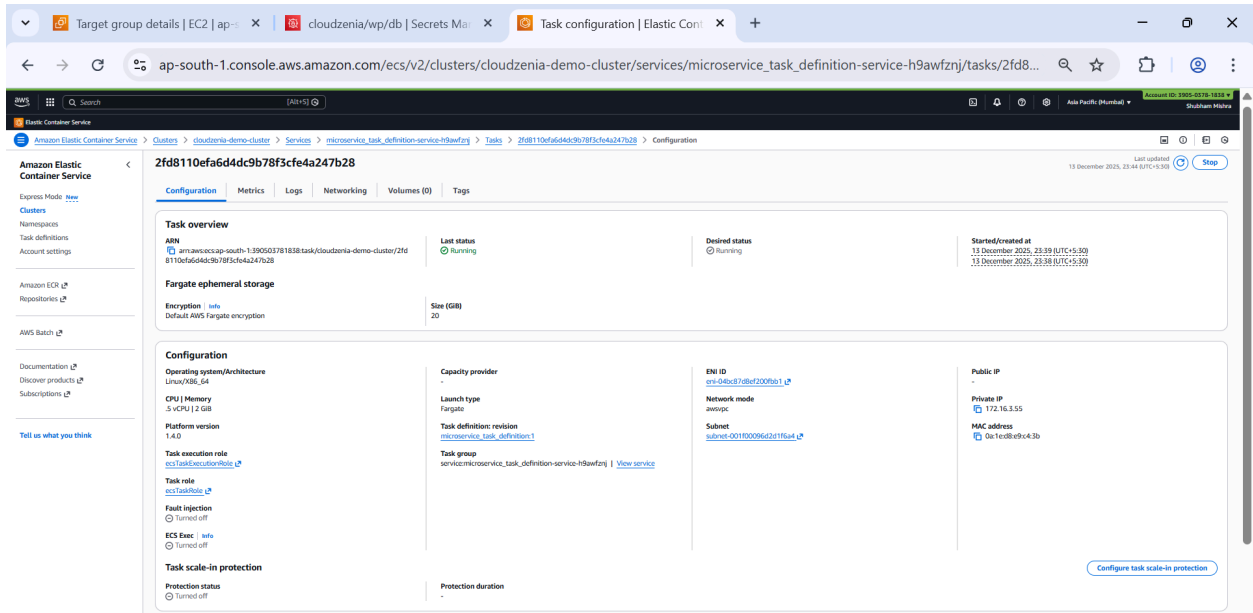
a) Service for microservice task created successfully.



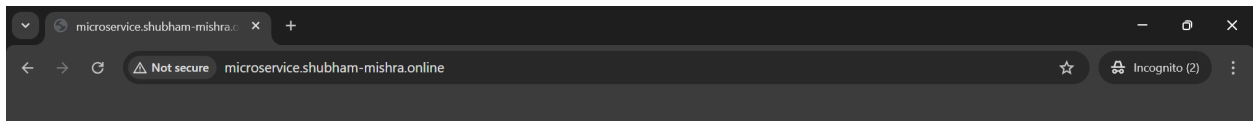
TG\_MICROSERVICE registered target

STEP 27) Once service created we can see task(container is running ) created.





## Step 28) Now lets verify accessibility from subdomain



Hello from Microservice



This confirms our microservice ecs task is successfully created and it is responding over alb tg and route53 .

## Step 29) Now lets create another service in another task in ecs cluster for wordpress task definition

Target group details | EC2 | ap-south-1

Create service | Elastic Container Service

ap-south-1.console.aws.amazon.com/ecs/v2/clusters/cloudzenia-demo-cluster/create-service?region=ap-south-1

Search [Alt+S]

Asia Pacific (Mumbai) Account ID: 3905-0378-1038 Shubham Mishra

Amazon Elastic Container Service

Amazon Elastic Container Service

Express Mode [New](#)

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR [↗](#)

Repositories [↗](#)

AWS Batch [↗](#)

Documentation [↗](#)

Discover products [↗](#)

Subscriptions [↗](#)

Create service [info](#)

Service details

Task definition family

Select an existing task definition. To create a new task definition, go to [Task definitions](#).

wordpress\_task\_definition

Task definition revision [Latest](#)

Select the task definition revision from the 100 most recent entries, or enter a revision. Leave the field blank to use the latest revision.

3

Service name

Assign a service name that is unique for this cluster.

wordpress\_task\_definition-service-0439vbe6

Up to 255 letters (uppercase and lowercase), numbers, underscores and hyphens are allowed. Service names must be unique within a cluster.

Environment

Existing cluster

AWS Fargate

Update service | Elastic Container Service

Certificate details | a8c9f034-ff1

Listener details | EC2 | ap-south-1

ap-south-1.console.aws.amazon.com/ecs/v2/clusters/cloudzenia-demo-cluster/services/wordpress\_task\_definition-service-0439vbe6/update?region=ap-south-1

Search [Alt+S] Ask Amazon Q

Asia Pacific (Mumbai) Account ID: 3905-0378-1038 Shubham Mishra

Amazon Elastic Container Service

Amazon Elastic Container Service

Express Mode [New](#)

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR [↗](#)

Repositories [↗](#)

AWS Batch [↗](#)

Documentation [↗](#)

Discover products [↗](#)

Subscriptions [↗](#)

Tell us what you think

Update

You can now configure predictive scaling for your ECS services by using the service auto scaling section on the [Service detail page](#). This dedicated section enables you to configure all types of scaling policies, set up scheduled scaling actions, and track scaling activities. [Find out more](#)

☒ Use service auto scaling

Configure service auto scaling to adjust your service's desired count.

Minimum number of tasks

The lower boundary to which service auto scaling can adjust the desired count of the service.

1

Maximum number of tasks

The upper boundary to which service auto scaling can adjust the desired count of the service.

2

Scaling policy

Scaling policy type [info](#)

Create either a target tracking or step scaling policy.

☒ Target tracking

Increase or decrease the number of tasks that your service runs based on a target value for a specific metric.

☐ Step scaling

Increase or decrease the number of tasks that your service runs based on a set of scaling adjustments, known as step adjustments, that vary based on the size of the alarm breach.

Policy name

cpu\_utilization

ECS service metric

ECSAverageCPUUtilization

Target value

70

Scale-out cooldown period

60

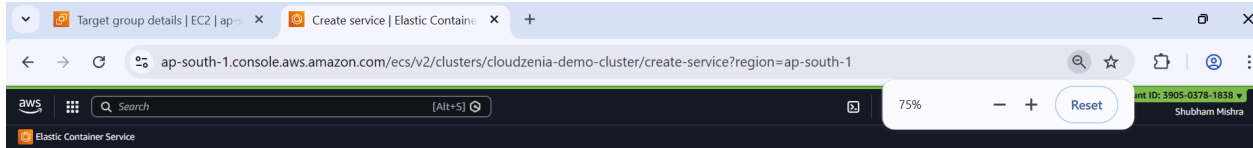
Scale-in cooldown period

180

☐ Turn off scale-in

[+ Add more scaling policies](#)

[Remove](#)



Amazon Elastic Container Service

Express Mode New

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR ↗

Repositories ↗

AWS Batch ↗

Documentation ↗

Discover products ↗

Subscriptions ↗

Tell us what you think

FARGATE

Platform version Info  
Specify the platform version on which to run your service.  
LATEST

Troubleshooting configuration - recommended

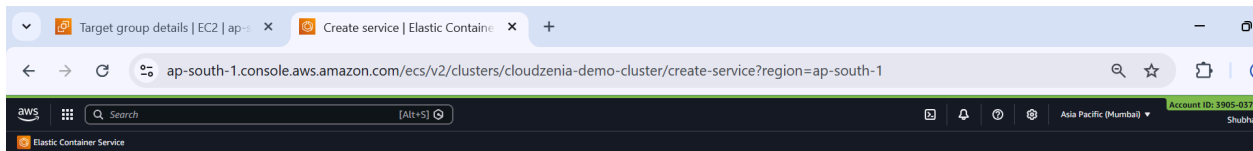
Deployment configuration

Scheduling strategy Info  
☒ 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

Availability Zone re-balancing Info  
☒ Turn on Availability Zone re-balancing  
Amazon ECS automatically detects Availability Zone imbalances in task distributions across an ECS service, and evenly redistributes ECS service tasks across Availability Zones.

Health check grace period Info  
60  
seconds



Amazon Elastic Container Service

Express Mode New

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR ↗

Repositories ↗

AWS Batch ↗

Documentation ↗

Discover products ↗

Subscriptions ↗

Tell us what you think

▼ Networking

VPC Info  
Select a VPC to use for your Amazon ECS resources.  
vpc-0cbb91d56e71b4fb7  
CUSTOM\_VPC  
Create a new VPC ↗

Subnets  
Choose the subnets within the VPC that the task scheduler should consider for placement.  
Choose subnets  
subnet-0eaf843ff693c4053  
APP\_PRIVATE\_SUBNET\_1  
ap-south-1a 172.16.2.0/24  
subnet-0d8ff8fa7e92f2ed0  
DB\_PRIVATE\_SUBNET\_1  
ap-south-1a 172.16.4.0/24  
Clear current selection

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 name  
Choose an existing security group.  
Choose security groups  
sg-078bef40da7aa4b4a  
ec2-sg\_1\_EC2\_SG

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

Target group details | EC2 | ap-south-1 | Create service | Elastic Container Service

ap-south-1.console.aws.amazon.com/ecs/v2/clusters/cloudzenia-demo-cluster/create-service?region=ap-south-1

Amazon Elastic Container Service

Express Mode [New](#)

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR [🔗](#)

Repositories [🔗](#)

AWS Batch [🔗](#)

Documentation [🔗](#)

Discover products [🔗](#)

Subscriptions [🔗](#)

Tell us what you think

**Load balancer type** [info](#)

Specify the load balancer type to distribute incoming traffic across the tasks running in your service.

☒ **Application Load Balancer**  
An Application Load Balancer makes routing decisions at the application layer (HTTP/HTTPS), supports path-based routing, and can route requests to one or more ports.

☐ **Network Load Balancer**  
A Network Load Balancer makes routing decisions at the transport layer (TCP/UDP).

**Container** [info](#)

The container and port to load balance the incoming traffic to.

wordpress:80-80

Host port/Container port

**Application Load Balancer** [info](#)

Specify whether to create a new load balancer or choose an existing one.

☐ Create a new load balancer

☒ Use an existing load balancer

**Load balancer** [info](#)

Choose an existing load balancer to distribute traffic. View existing load balancers and create new one in [EC2 Console](#).

CLOUDZENIA-ALB

Internet-facing

**Listener** [info](#)

Specify the port and protocol that the load balancer will listen for connection requests on.

☐ Create new listener

☒ Use an existing listener

HTTP:80

**Listener rules for 80:HTTP** [🔗](#) (0)

Traffic received by the listener is routed according to its rules. Rules are evaluated in priority order, from the lowest value to the highest value. The default rule is evaluated last.

Priority	Rule path	Target group
No rules are configured for this listener.		

**Target group** [info](#)

Specify whether to create a new target group or choose an existing one that the load balancer will use to route requests to the tasks in your service.

☐ Create new target group

☒ Use an existing target group

Target group name

TG-WORDPRESS

Health check path

/health

Service created for wordpress task.

Step 30) Now we can see two task for task definition ie Microservice and wordpress created successfully.

Target group details | EC2 | ap-south-1 | Cluster services | Elastic Container Service

ap-south-1.console.aws.amazon.com/ecs/v2/clusters/cloudzenia-demo-cluster/services?region=ap-south-1

Amazon Elastic Container Service

Express Mode [New](#)

Clusters

Namespaces

Task definitions

Account settings

Amazon ECR [🔗](#)

Repositories [🔗](#)

AWS Batch [🔗](#)

Documentation [🔗](#)

Discover products [🔗](#)

Subscriptions [🔗](#)

Tell us what you think

**cloudzenia-demo-cluster**

Cluster overview

ARN: arn:aws:ecs:ap-south-1:390503781838:cluster/cloudzenia-demo-cluster

Status: Active

CloudWatch monitoring: Container Insights [View in CloudWatch](#)

Registered container instances: -

**Services**

Draining: -

Active: 2

**Tasks**

Pending: 1

Running: 1

**Services (2)** [info](#)

Filter services by value

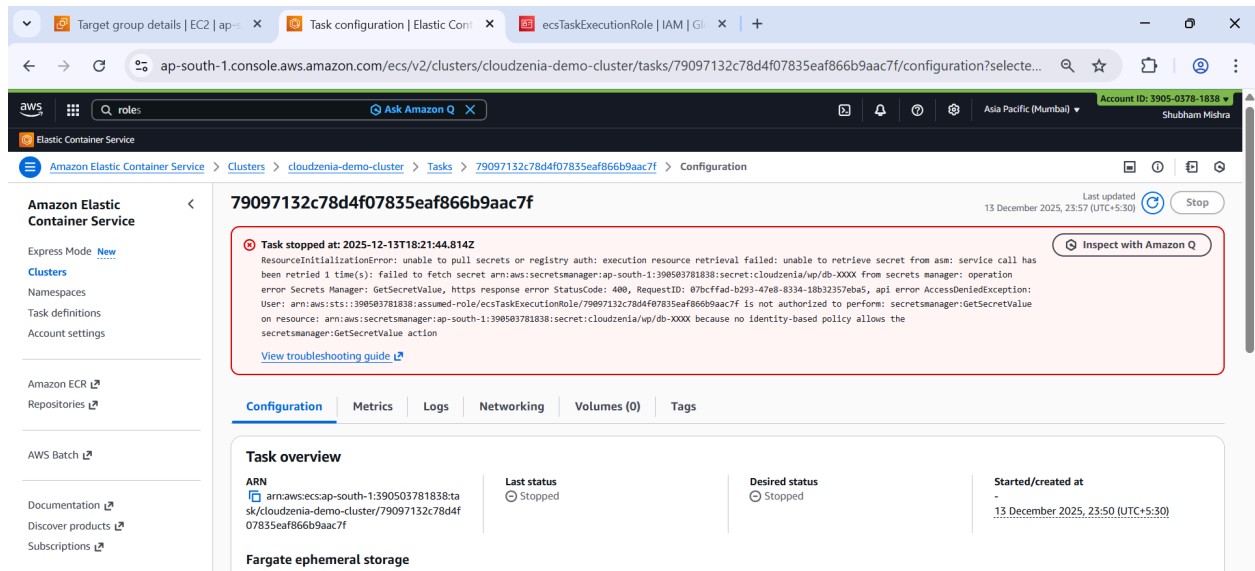
Filter launch type: Any launch type

Filter scheduling strategy: Any scheduling strategy

Filter resource management type: Any resource management type

Service name	ARN	Status	Replicas	Launch type	Task definition	Deployments and tasks	Last deployment	Created at
<a href="#">microservice_task_definition-service-043</a>	arn:aws:ecs:ap-south-1:390503781838:task-definition/microservice-task-definition:1	Active	1	FARGATE	<a href="#">microservice_task_definition:1</a>	1/1 tasks running	Completed	34 minutes ago
<a href="#">wordpress_task_definition-service-043</a>	arn:aws:ecs:ap-south-1:390503781838:task-definition/wordpress-task-definition:1	Active	1	FARGATE	<a href="#">wordpress_task_definition:1</a>	0/1 tasks running	In progress	49 seconds ago

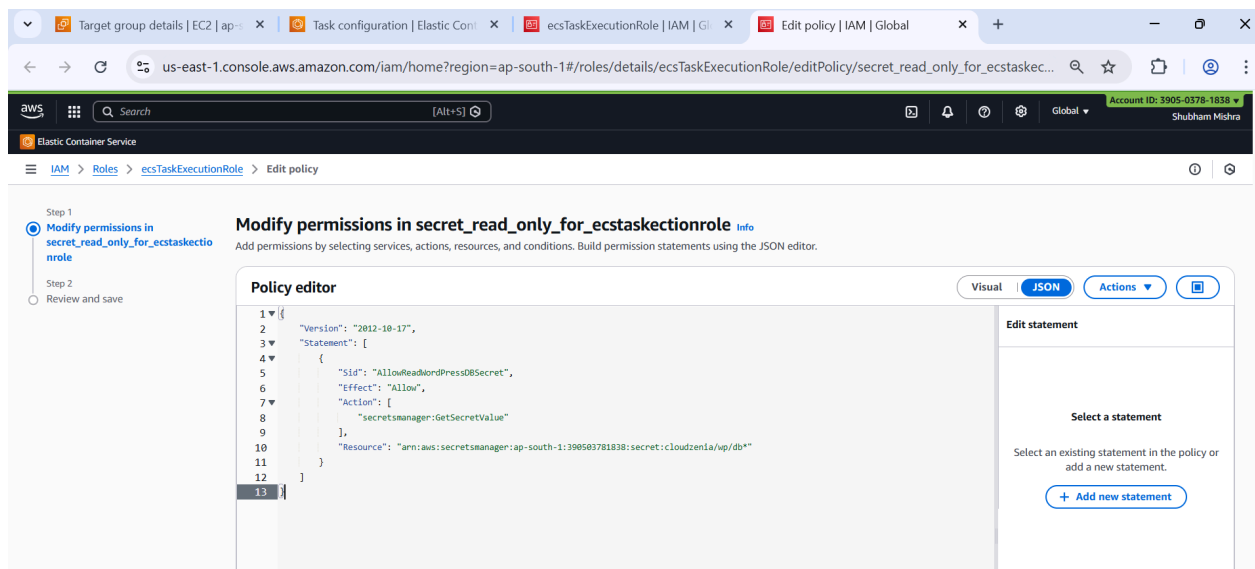
## Step 31) Now our task for wordpress is failing due to missing permission of ecstaskexecutionrole.



**Task stopped at: 2025-12-13T18:21:44.814Z**  
ResourceInitializationError: unable to pull secrets or registry auth: execution resource retrieval failed: unable to retrieve secret from asm: service call has been retried 1 time(s): failed to fetch secret arn:aws:secretsmanager:ap-south-1:390503781838:secret:cloudzenia/wp/db-XXXX from secrets manager: operation error Secrets Manager: GetSecretValue, https response error StatusCode: 400, RequestID: 970c7ffad-b293-4768-8334-18832357eb45, api error AccessDeniedException: User: arn:aws:sts::390503781838:assumed-role/ecstaskexecutionrole/79097132c78d4f07835eaf866b9aac7f is not authorized to perform: secretsmanager:GetSecretValue on resource: arn:aws:secretsmanager:ap-south-1:390503781838:secret:cloudzenia/wp/db-XXXX because no identity-based policy allows the secretsmanager:GetSecretValue action  
[View troubleshooting guide](#)

**Task overview**  
ARN: arn:aws:ecs:ap-south-1:390503781838:task/cloudzenia-demo-cluster/79097132c78d4f07835eaf866b9aac7f  
Last status: Stopped  
Desired status: Stopped  
Started/created at: 13 December 2025, 23:50 (UTC+5:30)

## Step 31) Now we will create inpolicy to allow read secrets in secrets manager and attach it to ecstaskexecutionrole.



**Modify permissions in secret\_read\_only\_for\_ecstaskexecutionrole**  
Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

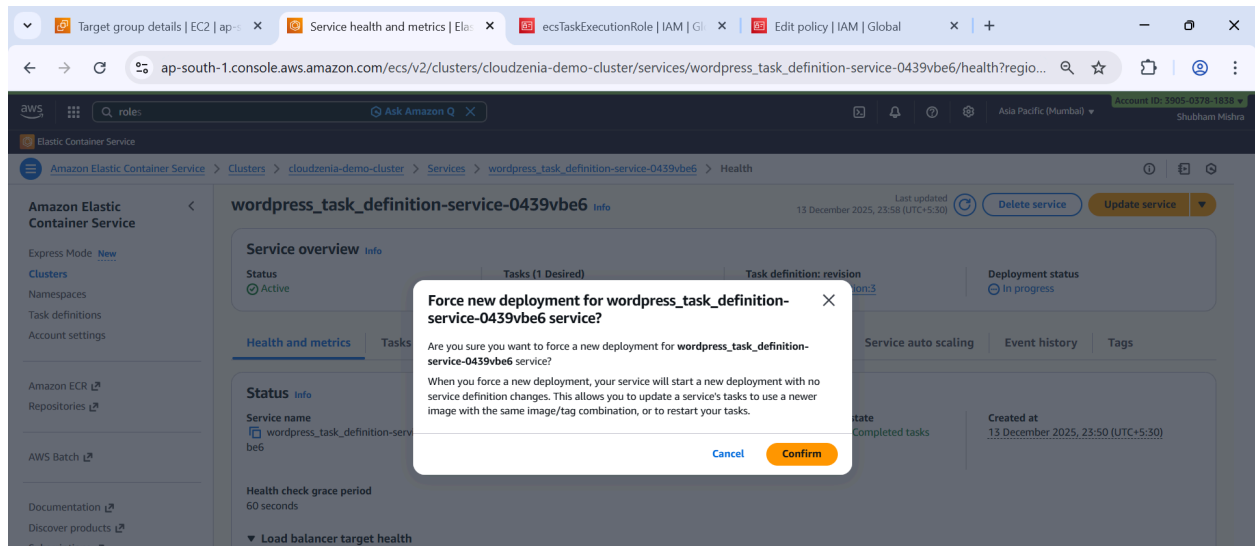
**Policy editor**

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Sid": "AllowReadWordPressDBSecret",  
6       "Effect": "Allow",  
7       "Action": [  
8         "secretsmanager:GetSecretValue"  
9       ],  
10      "Resource": "arn:aws:secretsmanager:ap-south-1:390503781838:secret:cloudzenia/wp/db-XXXX"  
11    }  
12  ]  
13 }
```

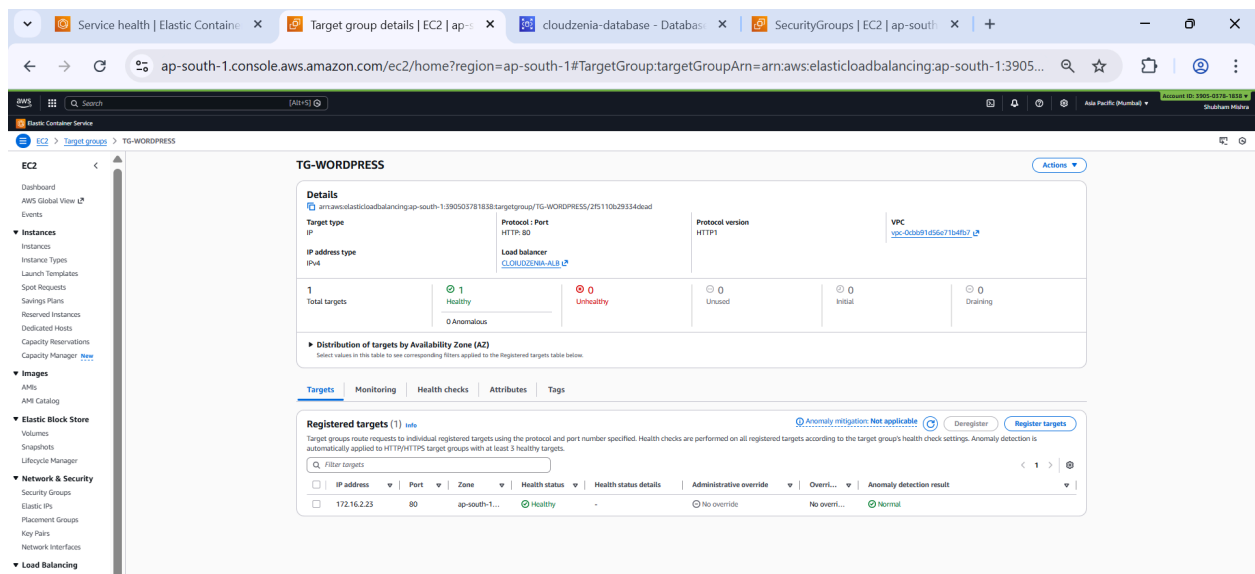
**Edit statement**

Select a statement  
Select an existing statement in the policy or add a new statement.  
[+ Add new statement](#)

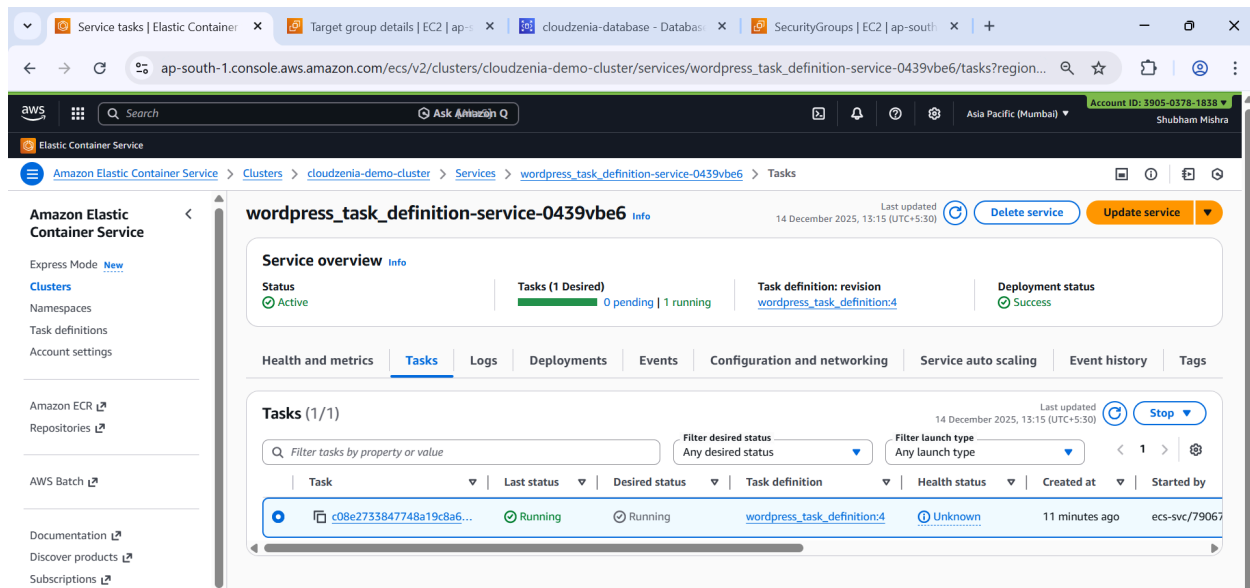
## Step 32) Now we will force new deployment after updating role for wordpress task definition.



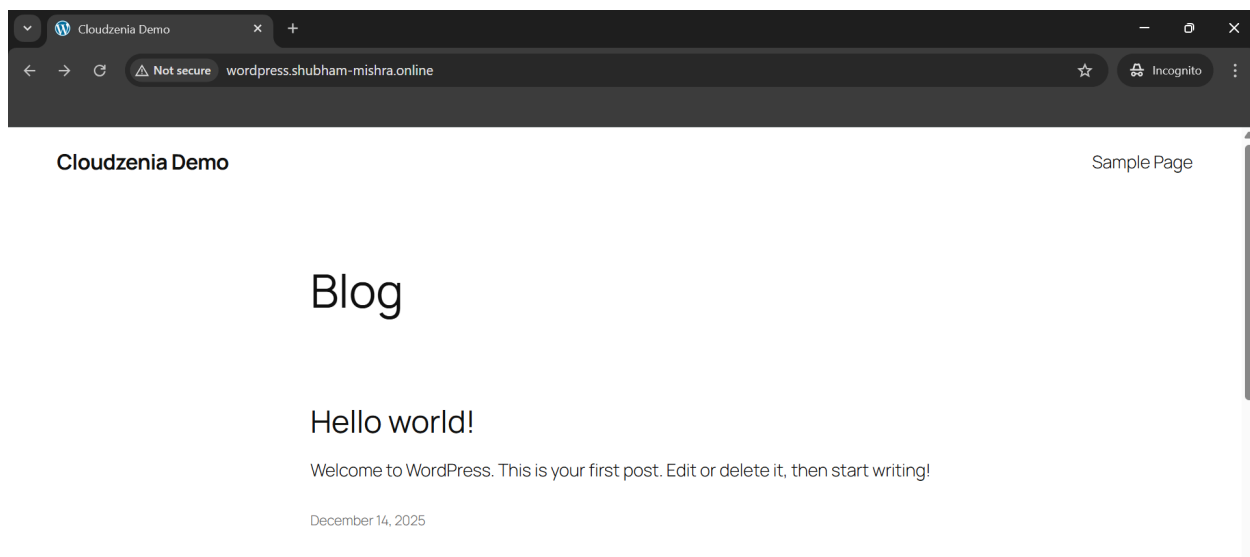
## Step 33) Now we can see the target group for worpdress is in healthy state



## Step 34) Now we can see healthy container task for wordpress

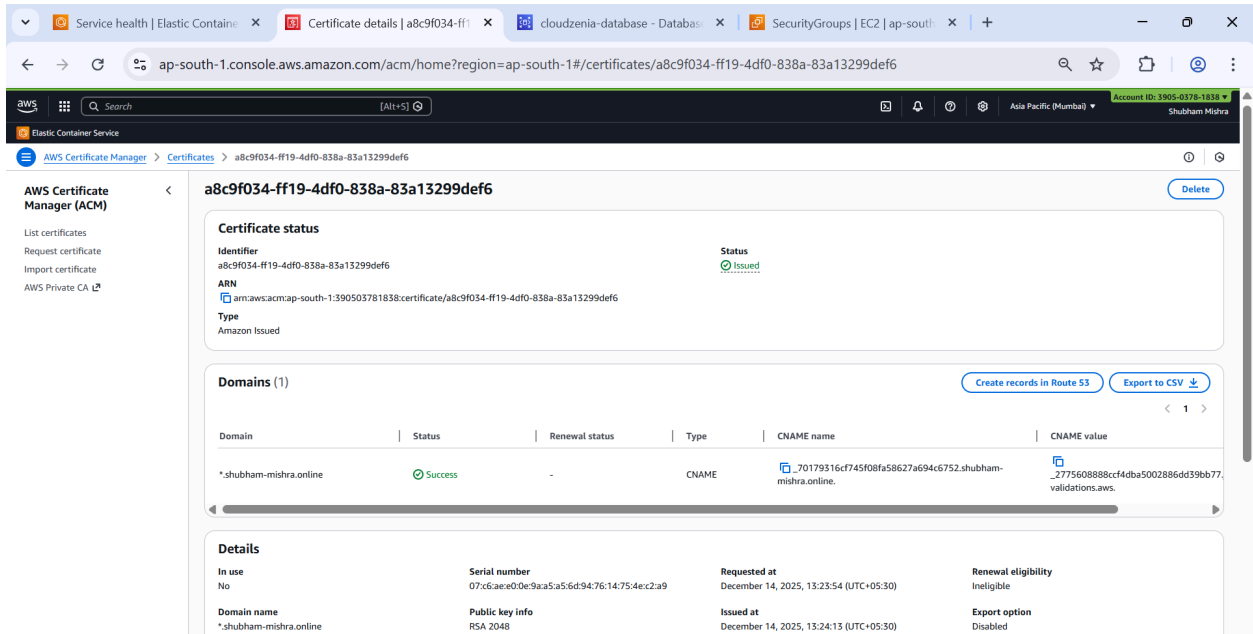


## Step 35) Now lets verify worpress.shubham-mishra.online for another task definition



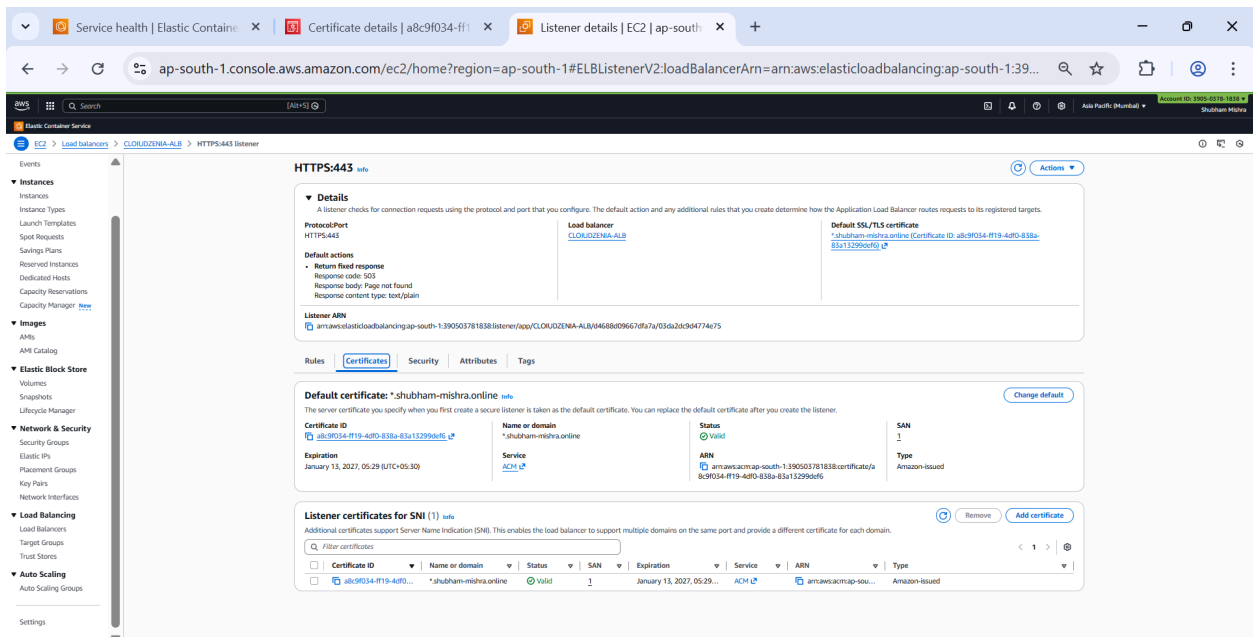
This confirms that wordpress container is running and is accessible by alb through route53 domain.

## Step 31) Now Lets add SSL certificate to our domains and make sure only it resolves over https.



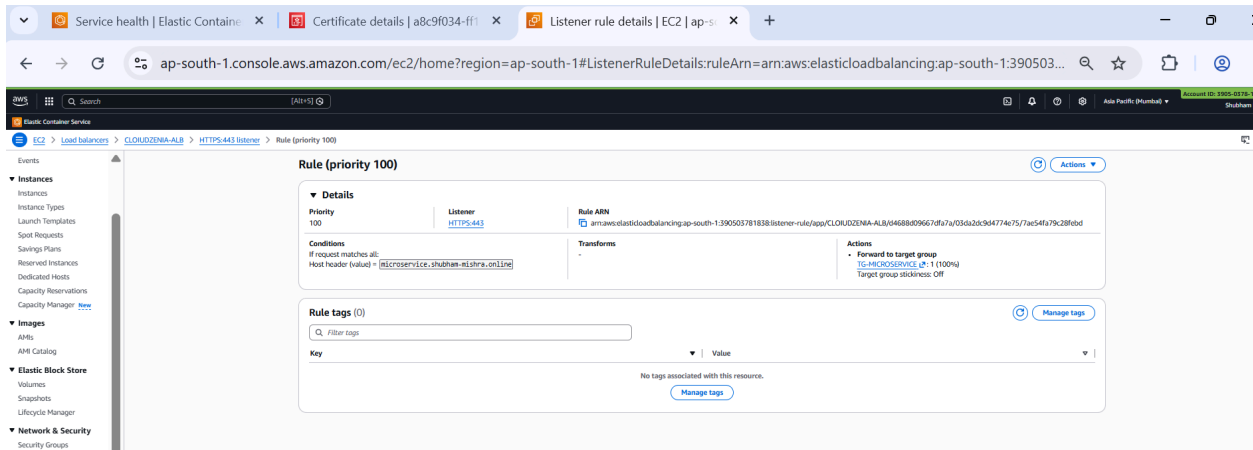
We have successfully added ssl certificate in ACM and configured into Route53 of another account.

Step 32) Now we will add 443 listener in alb to use SSL certificate.

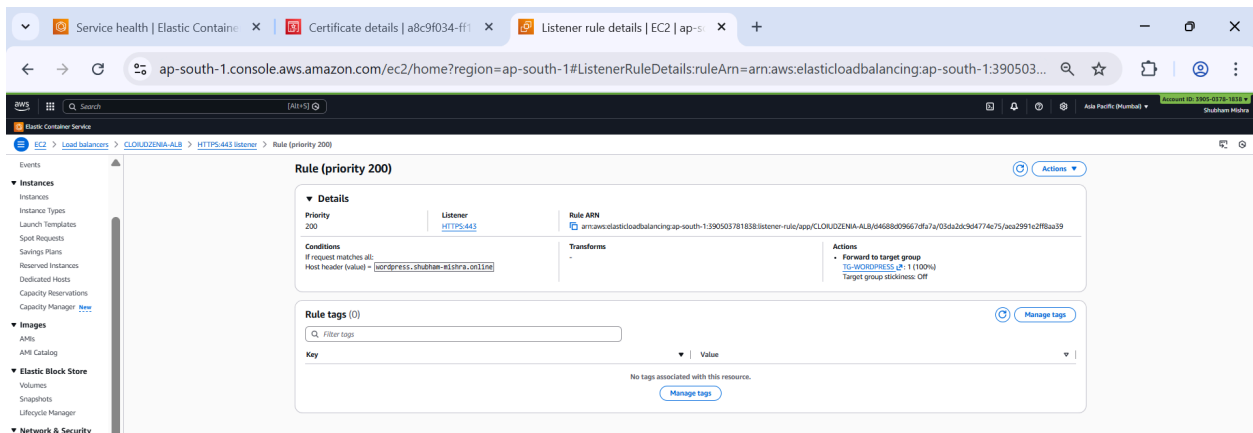


Step 33) Now we will add rule for TG-MICROSERVICE target group if host headed is microservice.shubham-mishra.online then forward to TG-MICROSERVICE.

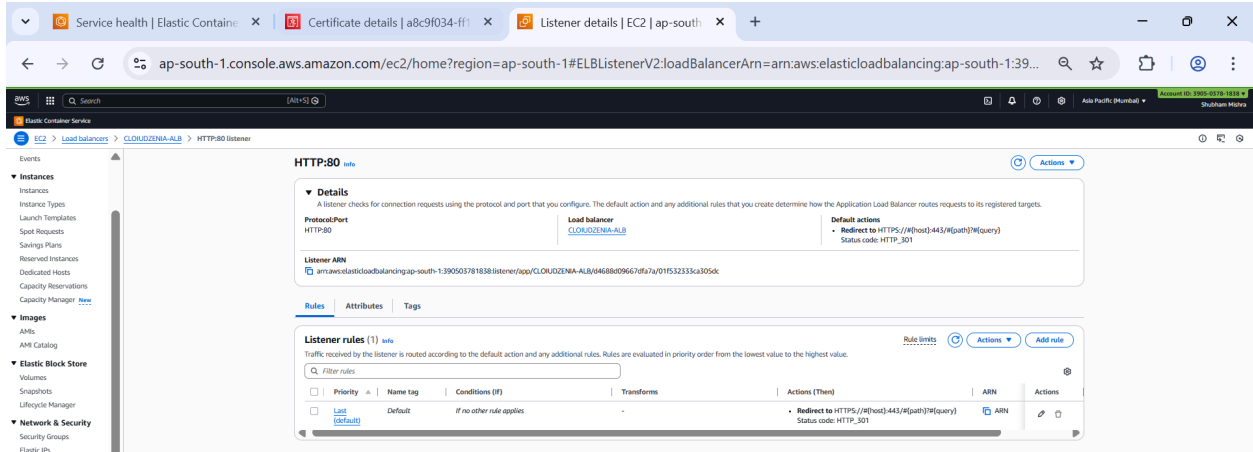




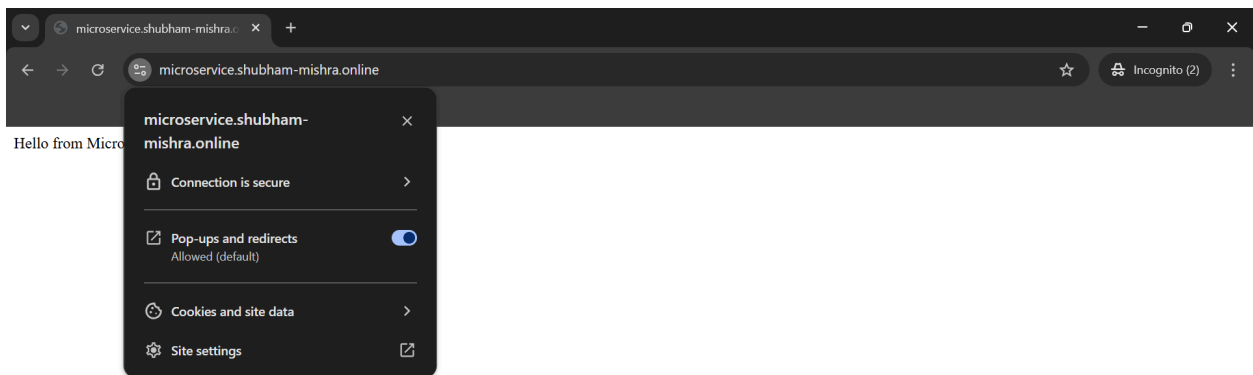
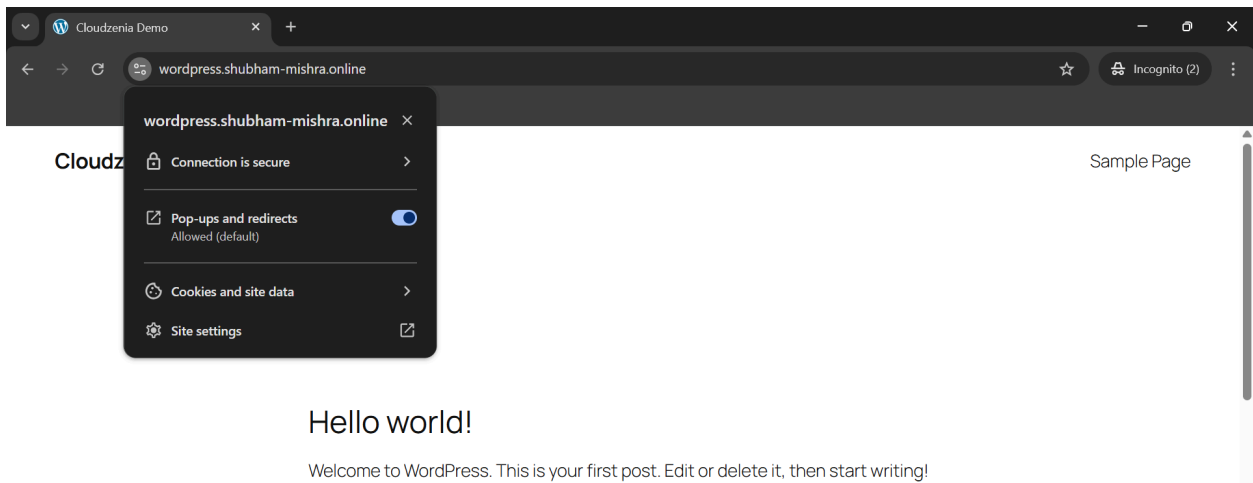
**Step 34) Now we will add rule for TG-WORDPRESS target group if host headed is wordpress.shubham-mishra.online then forward to TG-WORDPRESS.**



**Step 35) Now we will add redirect rule in http if any request on http then redirect to https 443**



**Step 36) Now we can see ssl certificate is working on our domain**



**Step 37) Now we can see that even http request is being redirected to https.**

```
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>curl http://microservice.shubham-mishra.online
<html>
<head><title>301 Moved Permanently</title></head>
<body>
<center><h1>301 Moved Permanently</h1></center>
</body>
</html>

C:\Users\UIPL-Admin>curl http://wordpress.shubham-mishra.online
<html>
<head><title>301 Moved Permanently</title></head>
<body>
<center><h1>301 Moved Permanently</h1></center>
</body>
</html>

C:\Users\UIPL-Admin>
```

```
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>
C:\Users\UIPL-Admin>curl -L https://microservice.shubham-mishra.online
Hello from Microservice
C:\Users\UIPL-Admin>
```

```
C:\Users\UIPL-Admin>curl https://wordpress.shubham-mishra.online
<!DOCTYPE html>
<html lang="en-US">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <meta name='robots' content='noindex, nofollow' />
  <title>Cloudzenia Demo</title>
  <link rel="alternate" type="application/rss+xml" title="Cloudzenia Demo &raquo; Feed" href="https://wordpress.shubham-mishra.online/feed/" />
  <link rel="alternate" type="application/rss+xml" title="Cloudzenia Demo &raquo; Comments Feed" href="https://wordpress.shubham-mishra.online/comments/feed/" />

```

## Project Outcome

- **Successfully designed and implemented a production-ready AWS architecture using ECS Fargate, Application Load Balancer,**

**RDS, Secrets Manager, IAM, ACM, and Route 53.**

- **Deployed two containerized services:**
  - **WordPress application backed by Amazon RDS (MySQL) running in private subnets.**
  - **Custom Node.js microservice responding with “*Hello from Microservice*”, fully containerized using Docker.**
- **Ensured secure architecture by:**
  - **Running ECS tasks and RDS instances in private subnets.**
  - **Restricting database access using least-privilege security groups.**
  - **Storing and consuming database credentials securely via AWS Secrets Manager.**
  - **Granting ECS access through IAM task roles without hard-coding secrets.**
- **Implemented high availability and scalability:**
  - **Application Load Balancer deployed in public subnets.**
  - **Traffic routed using host-based routing to separate target groups.**
  - **Enabled ECS Service Auto Scaling based on CPU and memory utilization.**
- **Configured secure domain access:**

- **Integrated custom subdomains:**
  - **wordpress.<domain>**
  - **microservice.<domain>**
- **Implemented HTTPS using AWS Certificate Manager (ACM).**
- **Enforced HTTP → HTTPS redirection at the ALB level.**
- **Achieved cost-optimized design:**
  - **ECS services can scale down to zero tasks when not in use.**
  - **NAT Gateway and ECS services can be safely stopped and re-enabled without configuration changes.**
  - **RDS endpoint remains unchanged during stop/start operations.**
- **Validated successful deployment by:**
  - **Accessing both applications through custom HTTPS domains.**
  - **Verifying healthy target group status and successful ALB routing.**
  - **Confirming secure database connectivity from WordPress via Secrets Manager.**

