

Lab 7: Capstone Lab - Build an AWS multi-tier architecture

© 2025 Amazon Web Services, Inc. or its affiliates. All rights reserved. This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited. All trademarks are the property of their owners.

Lab overview

You are tasked with applying your new knowledge to solve several architectural challenges within a specific business case. First, you are given a list of requirements related to the design. Then, you perform a series of actions to deploy and configure the services needed to meet the requirements.

During this capstone lab, you have access to the following:

- Downloadable AWS CloudFormation templates and other lab files
- Task scenarios, descriptions, and requirements
- Optional step-by-step instructions, if needed

The task scenarios provide relevant background and help you understand how the requirements solve a real-world business problem. Use the templates and the requirements list to complete all of the tasks in the capstone. Now that you are familiar with concepts and services, this lab solidifies your knowledge through practice. In the real world, you encounter problems that are not well-defined or sequenced logically. By the end of this capstone, you should have a better understanding of how you can apply architectural best practices to real-world problems.

Objectives

After completing this lab, you should be able to do the following:

- Deploy a virtual network spread across multiple Availability Zones in a Region using a CloudFormation template.
- Deploy a highly available and fully managed relational database across those Availability Zones (AZ) using Amazon Relational Database Service (Amazon RDS).
- Use Amazon Elastic File System (Amazon EFS) to provision a shared storage layer across multiple Availability Zones for the application tier, powered by Network File System (NFS).
- Create a group of web servers that automatically scales in response to load variations to complete your application tier.

Lab scenario

Example Corp. creates marketing campaigns for small-sized to medium-sized businesses. They recently hired you to work with the engineering teams to build out a proof of concept for their business. To date, they have hosted their client-facing application in an on-premises data center, but they recently decided to move their operations to the cloud in an effort to save money and transform their business with a cloud-first approach. Some members of their team have cloud experience and recommended the AWS Cloud services to build their solution.

In addition, they decided to redesign their web portal. Customers use the portal to access their accounts, create marketing plans, and run data analysis on their marketing campaigns. They would like to have a working prototype in two weeks. You must design an architecture to support this application. Your solution must be fast, durable, scalable, and more cost-effective than their existing on-premises infrastructure.

The following image shows the final architecture of the designed solution:

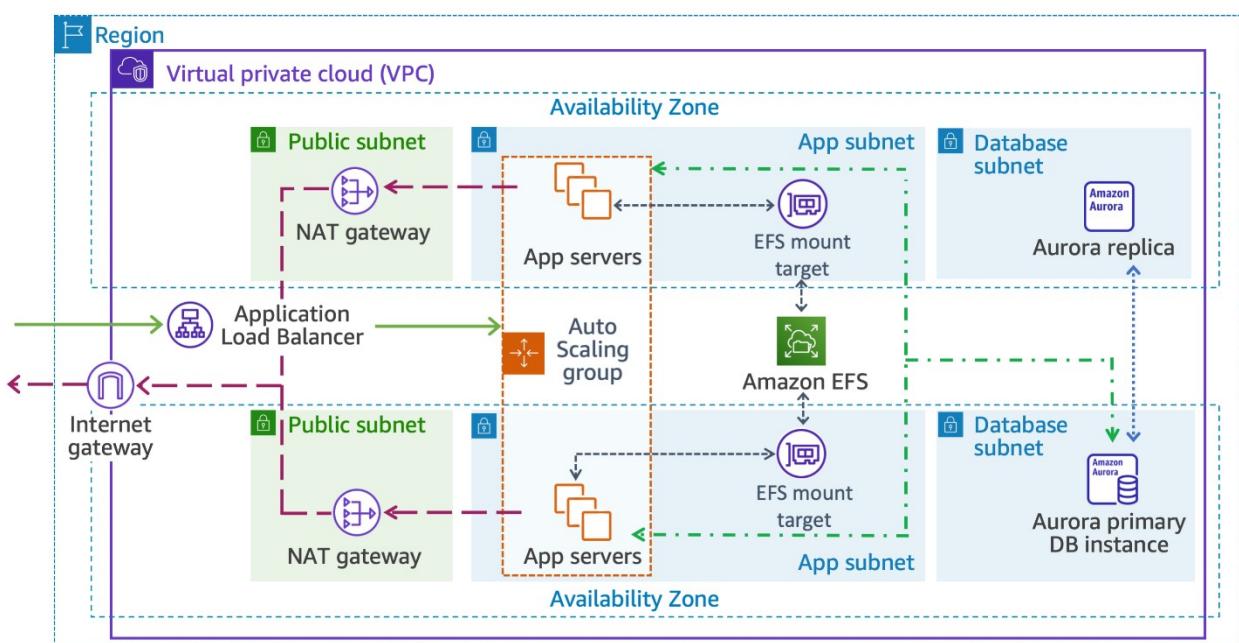


Image description: Preceding image depicts the entire architecture which will be created by the end of this lab. We have Amazon VPC to facilitate public and private networking using Public Subnets and Private Subnets. To ensure we are able to accept incoming internet traffic and allow outgoing traffic to the internet, we have Internet Gateway and NAT Gateways. In-order to facilitate high-availability (HA) for our architecture we have Application Load Balancer (ALB) and Auto-Scaling Group (ASG). There is also a fully scalable and elastic file server using Amazon EFS which can work across multiple availability zones (AZ)s. For database we are using

Amazon Aurora which works across multiple availability zones (AZ)s with primary in one availability zone (AZ) and a read-replica in another availability zone (AZ).

Task 1: Review and run a pre-configured CloudFormation template

In this task, you design and deploy a highly available and scalable deployment of the WordPress application stack. You use a CloudFormation template to deploy the networking resources required to support the application.

Task 1: Scenario

The company wants to deploy a new web application on WordPress on their existing web hosting account. The first step is to gather the requirements for the architecture design. To help you get started, the network team gave you a list of requirements to ensure there are no issues with the existing landscape. This includes the Classless Inter-Domain Routing (CIDR) range for the Amazon Virtual Private Cloud (Amazon VPC). Using this address range, build a VPC with two public subnets, two private application subnets, and two private database subnets. Make sure you attach an internet gateway to the VPC with a NAT gateway for routing traffic. They also requested two Elastic IP addresses. Ensure that you associate the following:

- The public subnet with the internet gateway
- The private application subnet and private database subnet with the NAT gateway

The network team also provided the route tables needed with their subnet associations.

Your cloud engineer has transferred the information request to a CloudFormation template and set up the security groups and outputs needed for future deployments. Please review the CloudFormation template with your cloud engineer, deploy the template, and check back with the network team to validate that the build meets all of their requirements.

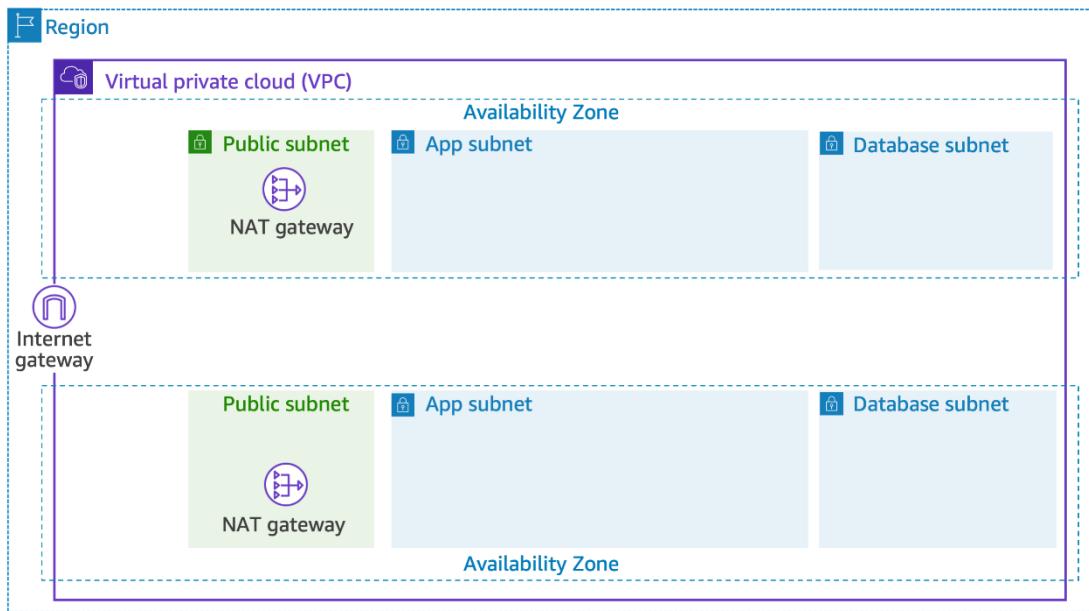


Image description: Preceding image depicts the entire architecture which will be created by the end of this lab. We have Amazon VPC to facilitate public and private networking using Public Subnets and Private Subnets. To ensure we are able to accept incoming internet traffic and allow outgoing traffic to the internet, we have Internet Gateway and NAT Gateways.

Task 1 instructions: Review and run a pre-configured CloudFormation template

Task 1.1: Navigate to the CloudFormation console

- At the top of the AWS Management Console, in the search box, search for and choose **CloudFormation**.

Stack name	Status	Created time	Description
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-8JacCtzJcCfMTMvBrEtkyJ-2-LabEnforce5QHfBLXAWF1H	CREATE_COMPLETE	2025-04-17 13:37:17 UTC+0530	-
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-8JacCtzJcCfMTMvBrEtkyJ-2-LabFusionGenerateRandomAlphaNumeri	CREATE_COMPLETE	2025-04-17 13:37:16 UTC+0530	-
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-8JacCtzJcCfMTMvBrEtkyJ-1	CREATE_COMPLETE	2025-04-17 13:37:14 UTC+0530	-
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-8JacCtzJcCfMTMvBrEtkyJ-2	CREATE_COMPLETE	2025-04-17 13:37:14 UTC+0530	-
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-8JacCtzJcCfMTMvBrEtkyJ-0	CREATE_COMPLETE	2025-04-17 13:37:14 UTC+0530	Lab7 environment template which builds some security enforcers

Task 1.2: Obtain and review the CloudFormation template

- Open the context (right-click) menu on this [Task1.yaml](#) link, and choose the option to save the CloudFormation template to your computer.
- Open the downloaded file in a text editor (not a word processor, preferably in Visual Studio Code).
- Review the CloudFormation template.
- Predict what resources are created by this template.

The Task1.yaml file is an AWS CloudFormation template that automatically sets up a complete virtual network environment in the AWS cloud. It contains all the instructions needed to create a Virtual Private Cloud (VPC) with both public and private subnets, allowing you to organize and secure your cloud resources. This includes components like an internet gateway for public internet access, NAT gateways for private subnet access to the internet, route tables to manage traffic flow, and security groups that act like firewalls to control access. In total, it sets up six subnets (two each for public, application, and database layers), elastic IP addresses, and connectivity rules—making it a solid foundation for deploying multi-tier applications securely in the cloud.

The Task1.yaml file consists of the following:

```
AWSTemplateFormatVersion: 2010-09-09 # Specifies the version of the AWS
CloudFormation template syntax.

Description: Lab7 Task 1 template which builds VPC, supporting
resources, a basic networking structure, and some Security groups for
use in later tasks. # A description of the template.

Parameters: # Defines input parameters that can be specified when the
stack is created or updated.

  VPCCIDR: # Parameter for the VPC CIDR block.
    Description: CIDR Block for VPC # Description of the VPCCIDR
parameter.
    Type: String # Specifies the data type of the parameter.
    Default: 10.0.0.0/16 # Default value for the VPCCIDR parameter.
    AllowedValues: # Restricts the values that can be specified for
this parameter.
      - 10.0.0.0/16 # Allowed value for the VPCCIDR parameter.

  PublicSubnet1Param: # Parameter for the Public Subnet 1 CIDR block.
    Description: Public Subnet 1 # Description of the
PublicSubnet1Param parameter.
    Type: String # Specifies the data type of the parameter.
    Default: 10.0.0.0/24 # Default value for the PublicSubnet1Param
parameter.
```

```
    AllowedValues: # Restricts the values that can be specified for
this parameter.
        - 10.0.0.0/24 # Allowed value for the PublicSubnet1Param
parameter.

    PublicSubnet2Param: # Parameter for the Public Subnet 2 CIDR block.
        Description: Public Subnet 2 # Description of the
PublicSubnet2Param parameter.
        Type: String # Specifies the data type of the parameter.
        Default: 10.0.1.0/24 # Default value for the PublicSubnet2Param
parameter.
        AllowedValues: # Restricts the values that can be specified for
this parameter.
        - 10.0.1.0/24 # Allowed value for the PublicSubnet2Param
parameter.

    AppSubnet1Param: # Parameter for the App Subnet 1 CIDR block.
        Description: App Subnet 1 # Description of the AppSubnet1Param
parameter.
        Type: String # Specifies the data type of the parameter.
        Default: 10.0.2.0/24 # Default value for the AppSubnet1Param
parameter.
        AllowedValues: # Restricts the values that can be specified for
this parameter.
        - 10.0.2.0/24 # Allowed value for the AppSubnet1Param parameter.

    AppSubnet2Param: # Parameter for the App Subnet 2 CIDR block.
        Description: App Subnet 2 # Description of the AppSubnet2Param
parameter.
        Type: String # Specifies the data type of the parameter.
        Default: 10.0.3.0/24 # Default value for the AppSubnet2Param
parameter.
        AllowedValues: # Restricts the values that can be specified for
this parameter.
        - 10.0.3.0/24 # Allowed value for the AppSubnet2Param parameter.

    DatabaseSubnet1Param: # Parameter for the Private Subnet 1 CIDR block
(intended for databases).
        Description: Private Subnet 1 # Description of the
DatabaseSubnet1Param parameter.
        Type: String # Specifies the data type of the parameter.
        Default: 10.0.4.0/24 # Default value for the DatabaseSubnet1Param
parameter.
        AllowedValues: # Restricts the values that can be specified for
this parameter.
        - 10.0.4.0/24 # Allowed value for the DatabaseSubnet1Param
parameter.
```

```
DatabaseSubnet2Param: # Parameter for the Private Subnet 2 CIDR block  
(intended for databases).  
  Description: Private Subnet 2 # Description of the  
DatabaseSubnet2Param parameter.  
  Type: String # Specifies the data type of the parameter.  
  Default: 10.0.5.0/24 # Default value for the DatabaseSubnet2Param  
parameter.  
  AllowedValues: # Restricts the values that can be specified for  
this parameter.  
    - 10.0.5.0/24 # Allowed value for the DatabaseSubnet2Param  
parameter.  
  
Resources: # Defines the AWS resources that will be created or updated  
by the template.  
#####  
# VPC and Network Structure  
#####  
LabVPC: # Logical ID of the VPC resource.  
  Type: 'AWS::EC2::VPC' # AWS resource type for creating a VPC.  
  Properties: # Properties that define the VPC.  
    CidrBlock: !Ref VPCCIDR # The IPv4 CIDR block for the VPC,  
referencing the VPCCIDR parameter.  
    EnableDnsSupport: True # Indicates whether DNS resolution is  
supported for the VPC.  
    EnableDnsHostnames: True # Indicates whether instances launched  
in the VPC get public DNS hostnames.  
    InstanceTenancy: 'default' # The tenancy options for instances  
launched in the VPC. 'default' means instances run on shared hardware.  
    Tags: # Tags to apply to the VPC.  
      - Key: Name # Key of the tag.  
      Value: LabVPC # Value of the tag.  
  
LabInternetGateway: # Logical ID of the Internet Gateway resource.  
  Type: 'AWS::EC2::InternetGateway' # AWS resource type for creating  
an Internet Gateway.  
  
AttachGateway: # Logical ID of the VPC Gateway Attachment resource.  
  Type: 'AWS::EC2::VPCGatewayAttachment' # AWS resource type for  
attaching the Internet Gateway to the VPC.  
  Properties: # Properties that define the VPC Gateway Attachment.  
    VpcId: !Ref LabVPC # The ID of the VPC to attach to, referencing  
the LabVPC resource.  
    InternetGatewayId: !Ref LabInternetGateway # The ID of the  
Internet Gateway to attach, referencing the LabInternetGateway  
resource.  
  
#NATs  
NATGateway1: # Logical ID of the first NAT Gateway resource.
```

```

Type: AWS::EC2::NatGateway # AWS resource type for creating a NAT Gateway.
Properties: # Properties that define the NAT Gateway.
AllocationId: !GetAtt ElasticIPAddress1.AllocationId # The allocation ID of the Elastic IP address to associate with the NAT Gateway.
SubnetId: !Ref PublicSubnet1 # The ID of the public subnet in which to create the NAT Gateway.
Tags: # Tags to apply to the NAT Gateway.
- Key: Name # Key of the tag.
Value: NATGateway1 # Value of the tag.

ElasticIPAddress1: # Logical ID of the first Elastic IP Address resource.
Type: AWS::EC2::EIP # AWS resource type for creating an Elastic IP address.
Properties: # Properties that define the Elastic IP address.
Domain: vpc # Specifies that the EIP is for use in a VPC.

NATGateway2: # Logical ID of the second NAT Gateway resource.
Type: AWS::EC2::NatGateway # AWS resource type for creating a NAT Gateway.
Properties: # Properties that define the NAT Gateway.
AllocationId: !GetAtt ElasticIPAddress2.AllocationId # The allocation ID of the Elastic IP address to associate with the NAT Gateway.
SubnetId: !Ref PublicSubnet2 # The ID of the public subnet in which to create the NAT Gateway.
Tags: # Tags to apply to the NAT Gateway.
- Key: Name # Key of the tag.
Value: NATGateway2 # Value of the tag.

ElasticIPAddress2: # Logical ID of the second Elastic IP Address resource.
Type: AWS::EC2::EIP # AWS resource type for creating an Elastic IP address.
Properties: # Properties that define the Elastic IP address.
Domain: vpc # Specifies that the EIP is for use in a VPC.

#Subnets
PublicSubnet1: # Logical ID of the first Public Subnet resource.
Type: 'AWS::EC2::Subnet' # AWS resource type for creating a Subnet.
Properties: # Properties that define the Subnet.
VpcId: !Ref LabVPC # The ID of the VPC in which to create the subnet, referencing the LabVPC resource.
CidrBlock: !Ref PublicSubnet1Param # The IPv4 CIDR block for the subnet, referencing the PublicSubnet1Param parameter.

```

```

        MapPublicIpOnLaunch: True # Indicates whether instances launched
in this subnet receive a public IPv4 address.
    AvailabilityZone: !Select # Selects an Availability Zone from a
list.
        - '0' # Index of the Availability Zone to select (the first
one).
        - !GetAZs '' # Intrinsic function to get a list of Availability
Zones in the current region.
    Tags: # Tags to apply to the Subnet.
        - Key: Name # Key of the tag.
        Value: PublicSubnet1 # Value of the tag.

PublicSubnet2: # Logical ID of the second Public Subnet resource.
    Type: 'AWS::EC2::Subnet' # AWS resource type for creating a Subnet.
    Properties: # Properties that define the Subnet.
        VpcId: !Ref LabVPC # The ID of the VPC in which to create the
subnet, referencing the LabVPC resource.
        CidrBlock: !Ref PublicSubnet2Param # The IPv4 CIDR block for the
subnet, referencing the PublicSubnet2Param parameter.
        MapPublicIpOnLaunch: True # Indicates whether instances launched
in this subnet receive a public IPv4 address.
    AvailabilityZone: !Select # Selects an Availability Zone from a
list.
        - '1' # Index of the Availability Zone to select (the second
one).
        - !GetAZs '' # Intrinsic function to get a list of Availability
Zones in the current region.
    Tags: # Tags to apply to the Subnet.
        - Key: Name # Key of the tag.
        Value: PublicSubnet2 # Value of the tag.

AppSubnet1: # Logical ID of the first Application Subnet resource.
    Type: 'AWS::EC2::Subnet' # AWS resource type for creating a Subnet.
    Properties: # Properties that define the Subnet.
        VpcId: !Ref LabVPC # The ID of the VPC in which to create the
subnet, referencing the LabVPC resource.
        CidrBlock: !Ref AppSubnet1Param # The IPv4 CIDR block for the
subnet, referencing the AppSubnet1Param parameter.
        MapPublicIpOnLaunch: False # Indicates whether instances launched
in this subnet receive a public IPv4 address (set to false for private
subnets).
    AvailabilityZone: !Select # Selects an Availability Zone from a
list.
        - '0' # Index of the Availability Zone to select (the first
one).
        - !GetAZs '' # Intrinsic function to get a list of Availability
Zones in the current region.
    Tags: # Tags to apply to the Subnet.

```

```

    - Key: Name # Key of the tag.
      Value: AppSubnet1 # Value of the tag.

AppSubnet2: # Logical ID of the second Application Subnet resource.
  Type: 'AWS::EC2::Subnet' # AWS resource type for creating a Subnet.
  Properties: # Properties that define the Subnet.
    VpcId: !Ref LabVPC # The ID of the VPC in which to create the
subnet, referencing the LabVPC resource.
    CidrBlock: !Ref AppSubnet2Param # The IPv4 CIDR block for the
subnet, referencing the AppSubnet2Param parameter.
    MapPublicIpOnLaunch: False # Indicates whether instances launched
in this subnet receive a public IPv4 address (set to false for private
subnets).
    AvailabilityZone: !Select # Selects an Availability Zone from a
list.
      - '1' # Index of the Availability Zone to select (the second
one).
      - !GetAZs '' # Intrinsic function to get a list of Availability
Zones in the current region.
    Tags: # Tags to apply to the Subnet.
      - Key: Name # Key of the tag.
        Value: AppSubnet2 # Value of the tag.

DatabaseSubnet1: # Logical ID of the first Database Subnet resource.
  Type: 'AWS::EC2::Subnet' # AWS resource type for creating a Subnet.
  Properties: # Properties that define the Subnet.
    VpcId: !Ref LabVPC # The ID of the VPC in which to create the
subnet, referencing the LabVPC resource.
    CidrBlock: !Ref DatabaseSubnet1Param # The IPv4 CIDR block for
the subnet, referencing the DatabaseSubnet1Param parameter.
    MapPublicIpOnLaunch: False # Indicates whether instances launched
in this subnet receive a public IPv4 address (set to false for private
subnets).
    AvailabilityZone: !Select # Selects an Availability Zone from a
list.
      - '0' # Index of the Availability Zone to select (the first
one).
      - !GetAZs '' # Intrinsic function to get a list of Availability
Zones in the current region.
    Tags: # Tags to apply to the Subnet.
      - Key: Name # Key of the tag.
        Value: DatabaseSubnet1 # Value of the tag.

DatabaseSubnet2: # Logical ID of the second Database Subnet resource.
  Type: 'AWS::EC2::Subnet' # AWS resource type for creating a Subnet.
  Properties: # Properties that define the Subnet.
    VpcId: !Ref LabVPC # The ID of the VPC in which to create the
subnet, referencing the LabVPC resource.

```

```
    CidrBlock: !Ref DatabaseSubnet2Param # The IPv4 CIDR block for
the subnet, referencing the DatabaseSubnet2Param parameter.
    MapPublicIpOnLaunch: False # Indicates whether instances launched
in this subnet receive a public IPv4 address (set to false for private
subnets).
    AvailabilityZone: !Select # Selects an Availability Zone from a
list.
        - '1' # Index of the Availability Zone to select (the second
one).
        - !GetAZs '' # Intrinsic function to get a list of Availability
Zones in the current region.
    Tags: # Tags to apply to the Subnet.
        - Key: Name # Key of the tag.
          Value: DatabaseSubnet2 # Value of the tag.

#Routing
#Route Tables
    PublicRouteTable: # Logical ID of the Public Route Table resource.
        Type: 'AWS::EC2::RouteTable' # AWS resource type for creating a
Route Table.
        Properties: # Properties that define the Route Table.
            VpcId: !Ref LabVPC # The ID of the VPC with which the route table
is associated, referencing the LabVPC resource.
        Tags: # Tags to apply to the Route Table.
            - Key: Name # Key of the tag.
              Value: PublicRouteTable # Value of the tag.

    PrivateRouteTableAZ1: # Logical ID of the Private Route Table for
Availability Zone 1.
        Type: 'AWS::EC2::RouteTable' # AWS resource type for creating a
Route Table.
        Properties: # Properties that define the Route Table.
            VpcId: !Ref LabVPC # The ID of the VPC with which the route table
is associated, referencing the LabVPC resource.
        Tags: # Tags to apply to the Route Table.
            - Key: Name # Key of the tag.
              Value: PrivateRouteTableAZ1 # Value of the tag.

    PrivateRouteTableAZ2: # Logical ID of the Private Route Table for
Availability Zone 2.
        Type: 'AWS::EC2::RouteTable' # AWS resource type for creating a
Route Table.
        Properties: # Properties that define the Route Table.
            VpcId: !Ref LabVPC # The ID of the VPC with which the route table
is associated, referencing the LabVPC resource.
        Tags: # Tags to apply to the Route Table.
            - Key: Name # Key of the tag.
              Value: PrivateRouteTableAZ2 # Value of the tag.
```

```
#Routes
  PublicRoute: # Logical ID of the Public Route resource.
    Type: 'AWS::EC2::Route' # AWS resource type for creating a Route.
    Properties: # Properties that define the Route.
      RouteTableId: !Ref PublicRouteTable # The ID of the route table
      to associate with the route, referencing the PublicRouteTable resource.
      DestinationCidrBlock: 0.0.0.0/0 # The destination IPv4 CIDR block
      for the route (all traffic).
      GatewayId: !Ref LabInternetGateway # The ID of an internet
      gateway or virtual private gateway for the route, referencing the
      LabInternetGateway resource.

  PrivateRouteAZ1: # Logical ID of the Private Route for Availability
  Zone 1.
    Type: 'AWS::EC2::Route' # AWS resource type for creating a Route.
    Properties: # Properties that define the Route.
      RouteTableId: !Ref PrivateRouteTableAZ1 # The ID of the route
      table to associate with the route, referencing the PrivateRouteTableAZ1
      resource.
      DestinationCidrBlock: 0.0.0.0/0 # The destination IPv4 CIDR block
      for the route (all traffic).
      NatGatewayId: !Ref NATGateway1 # The ID of a NAT gateway for the
      route, referencing the NATGateway1 resource.

  PrivateRouteAZ2: # Logical ID of the Private Route for Availability
  Zone 2.
    Type: 'AWS::EC2::Route' # AWS resource type for creating a Route.
    Properties: # Properties that define the Route.
      RouteTableId: !Ref PrivateRouteTableAZ2 # The ID of the route
      table to associate with the route, referencing the PrivateRouteTableAZ2
      resource.
      DestinationCidrBlock: 0.0.0.0/0 # The destination IPv4 CIDR block
      for the route (all traffic).
      NatGatewayId: !Ref NATGateway2 # The ID of a NAT gateway for the
      route, referencing the NATGateway2 resource.

#Subnet Associations
  PublicSubnet1RouteTableAssociation: # Logical ID for associating
  PublicSubnet1 with the PublicRouteTable.
    Type: 'AWS::EC2::SubnetRouteTableAssociation' # AWS resource type
    for associating a subnet with a route table.
    Properties: # Properties that define the subnet route table
    association.
      SubnetId: !Ref PublicSubnet1 # The ID of the subnet to associate,
      referencing the PublicSubnet1 resource.
      RouteTableId: !Ref PublicRouteTable # The ID of the route table
      to associate with the subnet, referencing the PublicRouteTable
      resource.
```

```
    PublicSubnet2RouteTableAssociation: # Logical ID for associating
    PublicSubnet2 with the PublicRouteTable.
        Type: 'AWS::EC2::SubnetRouteTableAssociation' # AWS resource type
        for associating a subnet with a route table.
        Properties: # Properties that define the subnet route table
        association.
            SubnetId: !Ref PublicSubnet2 # The ID of the subnet to associate,
            referencing the PublicSubnet2 resource.
            RouteTableId: !Ref PublicRouteTable # The ID of the route table
            to associate with the subnet, referencing the PublicRouteTable
            resource.

    AppSubnet1RouteTableAssociation: # Logical ID for associating
    AppSubnet1 with the PrivateRouteTableAZ1.
        Type: 'AWS::EC2::SubnetRouteTableAssociation' # AWS resource type
        for associating a subnet with a route table.
        Properties: # Properties that define the subnet route table
        association.
            SubnetId: !Ref AppSubnet1 # The ID of the subnet to associate,
            referencing the AppSubnet1 resource.
            RouteTableId: !Ref PrivateRouteTableAZ1 # The ID of the route
            table to associate with the subnet, referencing the
            PrivateRouteTableAZ1 resource.

    AppSubnet2RouteTableAssociation: # Logical ID for associating
    AppSubnet2 with the PrivateRouteTableAZ2.
        Type: 'AWS::EC2::SubnetRouteTableAssociation' # AWS resource type
        for associating a subnet with a route table.
        Properties: # Properties that define the subnet route table
        association.
            SubnetId: !Ref AppSubnet2 # The ID of the subnet to associate,
            referencing the AppSubnet2 resource.
            RouteTableId: !Ref PrivateRouteTableAZ2 # The ID of the route
            table to associate with the subnet, referencing the
            PrivateRouteTableAZ2 resource.

    DatabaseSubnet1RouteTableAssociation: # Logical ID for associating
    DatabaseSubnet1 with the PrivateRouteTableAZ1.
        Type: 'AWS::EC2::SubnetRouteTableAssociation' # AWS resource type
        for associating a subnet with a route table.
        Properties: # Properties that define the subnet route table
        association.
            SubnetId: !Ref DatabaseSubnet1 # The ID of the subnet to
            associate, referencing the DatabaseSubnet1 resource.
            RouteTableId: !Ref PrivateRouteTableAZ1 # The ID of the route
            table to associate with the subnet, referencing the
            PrivateRouteTableAZ1 resource.
```

```

DatabaseSubnet2RouteTableAssociation: # Logical ID for associating
DatabaseSubnet2 with the PrivateRouteTableAZ2.
  Type: 'AWS::EC2::SubnetRouteTableAssociation' # AWS resource type
for associating a subnet with a route table.
  Properties: # Properties that define the subnet route table
association.
    SubnetId: !Ref DatabaseSubnet2 # The ID of the subnet to
associate, referencing the DatabaseSubnet2 resource.
    RouteTableId: !Ref PrivateRouteTableAZ2 # The ID of the route
table to associate with the subnet, referencing the
PrivateRouteTableAZ2 resource.

#####
# Security Groups
#####
AppInstanceSecurityGroup: # Logical ID for the Application Instance
Security Group.
  Type: 'AWS::EC2::SecurityGroup' # AWS resource type for creating a
Security Group.
  Properties: # Properties that define the Security Group.
    GroupDescription: Security Group allowing HTTP traffic for lab
instances # A description of the security group.
    VpcId: !Ref LabVPC # The ID of the VPC to associate the security
group with, referencing the LabVPC resource.
    Tags: # Tags to apply to the Security Group.
      - Key: Name # Key of the tag.
        Value: AppInstanceSecurityGroup # Value of the tag.
    SecurityGroupIngress: # Defines the inbound rules for the
security group.
      - IpProtocol: tcp # Allows TCP traffic.
        FromPort: 80 # Allows traffic on port 80 (HTTP).
        ToPort: 80 # Allows traffic on port 80 (HTTP).
        CidrIp: 0.0.0.0/0 # Allows traffic from any IP address.

RDSSecurityGroup: # Logical ID for the RDS Security Group.
  Type: 'AWS::EC2::SecurityGroup' # AWS resource type for creating a
Security Group.
  Properties: # Properties that define the Security Group.
    GroupDescription: Security Group allowing RDS instances to have
internet traffic # A description of the security group.
    VpcId: !Ref LabVPC # The ID of the VPC to associate the security
group with, referencing the LabVPC resource.
    Tags: # Tags to apply to the Security Group.
      - Key: Name # Key of the tag.
        Value: RDSSecurityGroup # Value of the tag.

EFSMountTargetSecurityGroup: # Logical ID for the EFS Mount Target
Security Group.

```

```

Type: 'AWS::EC2::SecurityGroup' # AWS resource type for creating a
Security Group.
  Properties: # Properties that define the Security Group.
    GroupDescription: Security Group allowing traffic between EFS
Mount Targets and Amazon EC2 instances # A description of the security
group.
    VpcId: !Ref LabVPC # The ID of the VPC to associate the security
group with, referencing the LabVPC resource.
    Tags: # Tags to apply to the Security Group.
      - Key: Name # Key of the tag.
        Value: EFSMountTargetSecurityGroup # Value of the tag.
    SecurityGroupIngress: # Defines the inbound rules for the
security group.
      - IpProtocol: tcp # Allows TCP traffic.
        SourceSecurityGroupId: !Ref AppInstanceSecurityGroup # Allows
traffic from the AppInstanceSecurityGroup.
        FromPort: 80 # Allows traffic on port 80.
        ToPort: 80 # Allows traffic on port 80.

Outputs: # Defines values that are returned when the stack is created.

Region: # Logical ID of the Region output.
  Description: "Lab Region" # Description of the output value.
  Value: !Ref AWS::Region # Intrinsic function to get the AWS Region
in which the stack is being created.

DatabaseSubnet1CIDR: # Logical ID of the DatabaseSubnet1CIDR output.
  Description: "CIDR block for the DB Subnet in AZ a" # Description
of the output value.
  Value: !Ref DatabaseSubnet1Param # References the
DatabaseSubnet1Param parameter value.

DatabaseSubnet2CIDR: # Logical ID of the DatabaseSubnet2CIDR output.
  Description: "CIDR block for the DB Subnet in AZ b" # Description
of the output value.
  Value: !Ref DatabaseSubnet2Param # References the
DatabaseSubnet2Param parameter value.

DatabaseSubnet1ID: # Logical ID of the DatabaseSubnet1ID output.
  Description: "The Subnet ID for the DB Subnet in AZ a" #
Description of the output value.
  Value: !Ref DatabaseSubnet1 # References the physical ID of the
DatabaseSubnet1 resource.

  Export: # Defines the output as an export value for cross-stack
references.
    Name: "DatabaseSubnet1ID" # The name used to export this output
value.

```

```
DatabaseSubnet2ID: # Logical ID of the DatabaseSubnet2ID output.
  Description: "The Subnet ID for the DB Subnet in AZ b" #
Description of the output value.
  Value: !Ref DatabaseSubnet2 # References the physical ID of the
DatabaseSubnet2 resource.
  Export: # Defines the output as an export value for cross-stack
references.
    Name: "DatabaseSubnet2ID" # The name used to export this output
value.

  AppInstanceSecurityGroupID: # Logical ID of the
AppInstanceSecurityGroupID output.
  Description: "The Security Group ID for the Lab Instance Security
Group" # Description of the output value.
  Value: !Ref AppInstanceSecurityGroup # References the physical ID
of the AppInstanceSecurityGroup resource.
  Export: # Defines the output as an export value for cross-stack
references.
    Name: "AppInstanceSecurityGroupID" # The name used to export this
output value.

  EFSMountTargetSecurityGroupID: # Logical ID of the
EFSMountTargetSecurityGroupID output.
  Description: "The Security Group ID for the Lab EFS Mount Target" #
Description of the output value.
  Value: !Ref EFSMountTargetSecurityGroup # References the physical
ID of the EFSMountTargetSecurityGroup resource.
  Export: # Defines the output as an export value for cross-stack
references.
    Name: "EFSMountTargetSecurityGroupID" # The name used to export
this output value.

  RDSSecurityGroupID: # Logical ID of the RDSSecurityGroupID output.
  Description: "The Security Group ID for the Lab RDS cluster" #
Description of the output value.
  Value: !Ref RDSSecurityGroup # References the physical ID of the
RDSSecurityGroup resource.
  Export: # Defines the output as an export value for cross-stack
references.
    Name: "RDSSecurityGroupID" # The name used to export this output
value.

  VPCID: # Logical ID of the VPCID output.
  Description: "The VPC ID for the lab" # Description of the output
value.
  Value: !Ref LabVPC # References the physical ID of the LabVPC
resource.
```

```

Export: # Defines the output as an export value for cross-stack
references.
Name: "VPCID" # The name used to export this output value.

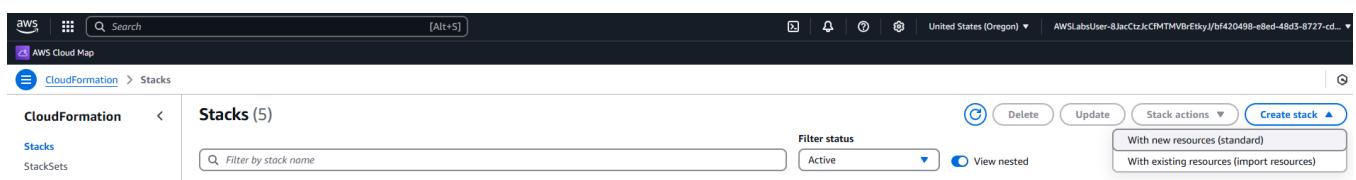
```

Task 1.3: Create the CloudFormation stack

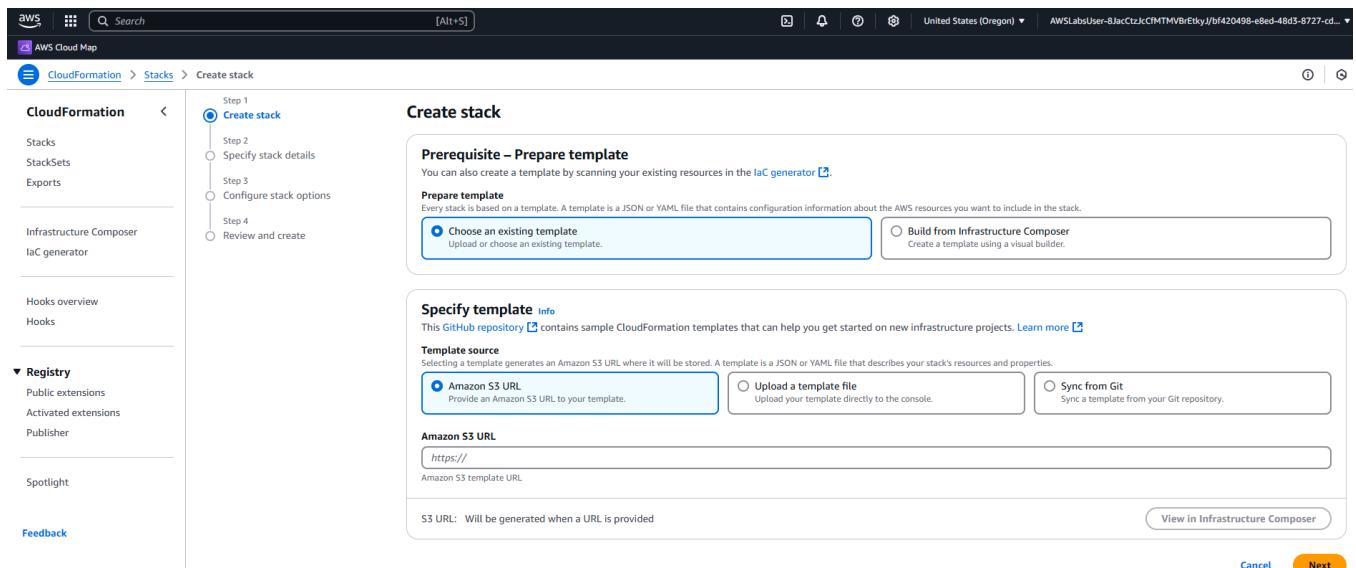
- Choose **Create stack**.

Note: If the console starts you on the Stacks page instead of the Amazon CloudFormation landing page, then you can get to the Create stack page in two steps.

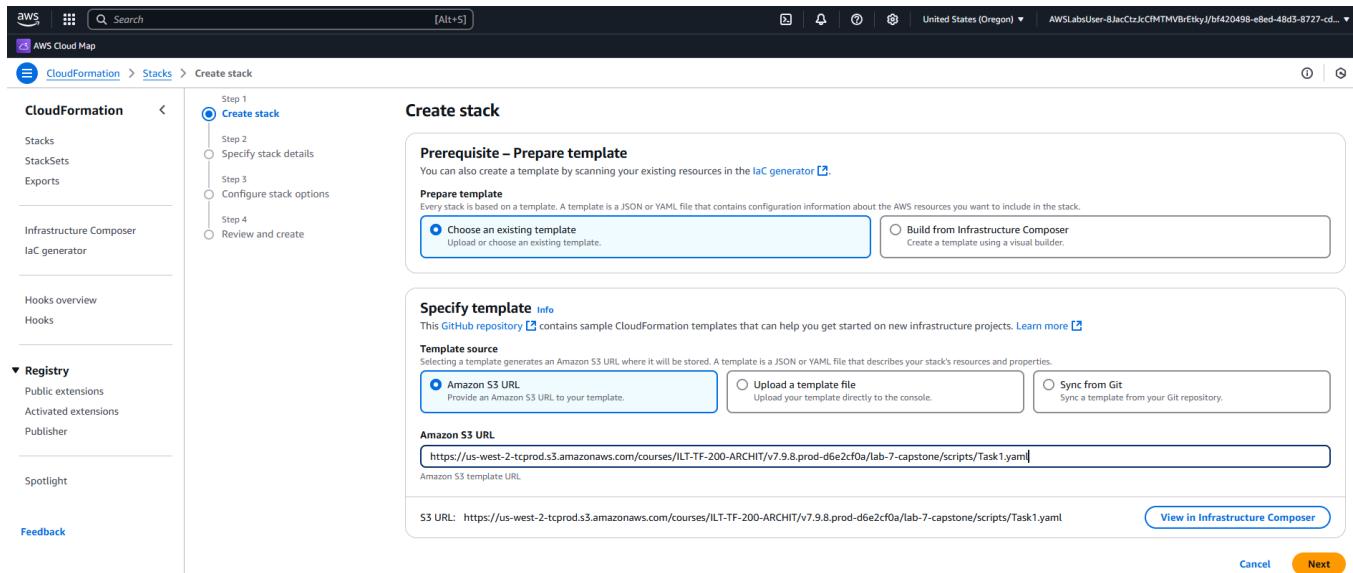
- Choose the **Create stack** dropdown menu.
- Choose **With new resources (standard)**.



The **Create stack** page is displayed.

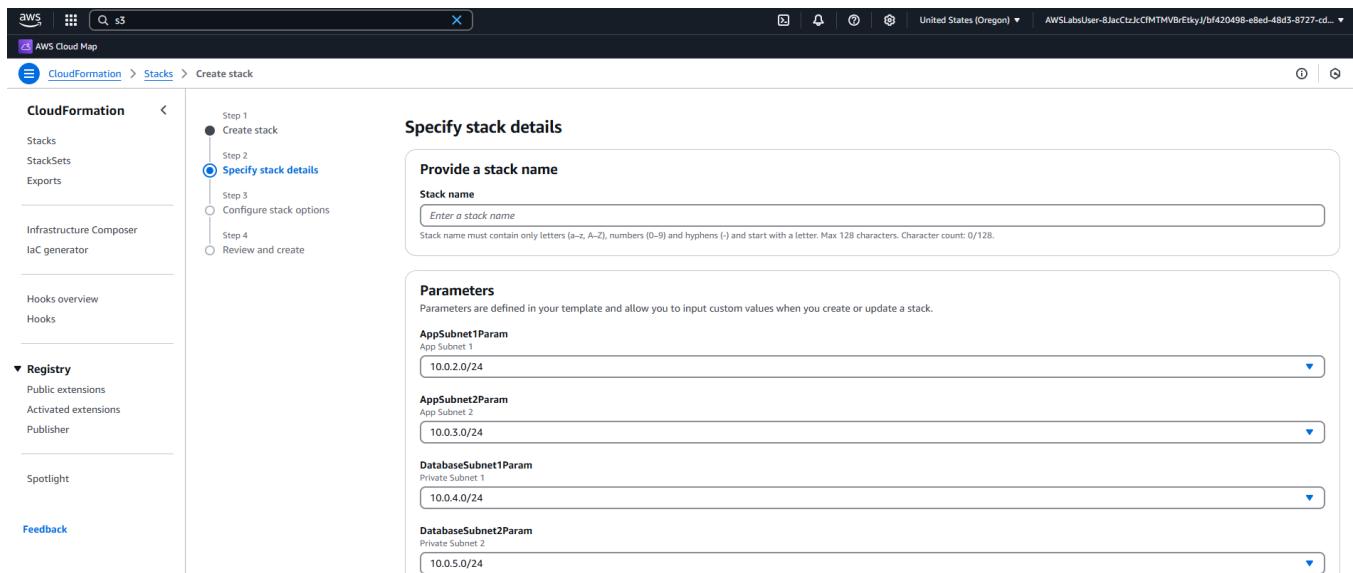


- Configure the following:
 - Select Choose an existing template.
 - Select Amazon S3 URL.
 - Copy the **Task1TemplateUrl** value (<https://us-west-2-tcprod.s3.amazonaws.com/courses/ILT-TF-200-ARCHIT/v7.9.8.prod-d6e2cf0a/lab-7-capstone/scripts/Task1.yaml>) and paste it in the **Amazon S3 URL** text box.



o Choose **Next**.

The **Specify stack details** page is displayed.



- **Stack name:** Enter **VPCStack**.
- **Parameters:** Keep the default values.

Specify stack details

Provide a stack name

Stack name

VPCStack

Stack name must contain only letters (a-z, A-Z), numbers (0-9) and hyphens (-) and start with a letter. Max 128 characters. Character count: 8/128.

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

AppSubnet1Param

App Subnet 1

10.0.2.0/24

AppSubnet2Param

App Subnet 2

10.0.3.0/24

DatabaseSubnet1Param

Private Subnet 1

10.0.4.0/24

DatabaseSubnet2Param

Private Subnet 2

10.0.5.0/24

PublicSubnet1Param

Public Subnet 1

10.0.0.0/24

PublicSubnet2Param

Public Subnet 2

10.0.1.0/24

VPCCIDR

CIDR Block for VPC

10.0.0.0/16

[Cancel](#)

[Previous](#)

[Next](#)

- Choose **Next**.

The **Configure stack options** page is displayed. You can use this page to specify additional parameters. You can browse the page, but leave settings at the default values.

Configure stack options

Tags - optional
Tags (key-value pairs) are used to apply metadata to AWS resources, which can help in organising, identifying and categorising those resources. You can add up to 50 unique tags for each stack.
No tags associated with the stack.

Add new tag
You can add 50 more tag(s)

Permissions - optional
Specify an existing AWS Identity and Access Management (IAM) service role that CloudFormation can assume.

IAM role - optional
Choose the IAM role for CloudFormation to use for all operations performed on the stack.

Stack failure options

Behaviour on provisioning failure
Specify the roll-back behaviour for a stack failure. [Learn more](#)

Roll back all stack resources
Roll back the stack to the last known stable state.

Preserve successfully provisioned resources
Preserves the state of successfully provisioned resources, while rolling back failed resources to the last known stable state. Resources without a last known stable state will be deleted upon the next stack operation.

Delete newly created resources during a rollback
Specify whether resources that were created during a failed operation should be deleted regardless of their deletion policy. [Learn more](#)

Use deletion policy
Retains or deletes created resources according to their attached deletion policy.

Delete all newly created resources
Deletes created resources during a rollback regardless of their attached deletion policy.

Additional settings
You can set additional options for your stack, like notification options and a stack policy. [Learn more](#)

Stack policy - optional
Defines the resources that you want to protect from unintentional updates during a stack update.

Rollback configuration - optional
Specify alarms for CloudFormation to monitor when creating and updating the stack. If the operation breaches an alarm threshold, CloudFormation rolls it back.

Notification options - optional
Specify a new or existing Amazon Simple Notification Service topic where notifications about stack events are sent.

Stack creation options - optional
Specify the timeout and termination protection options for stack creation.

- Choose **Next**.

The **Review and create** page is displayed. This page is a summary of all settings.

Review and create

Step 1: Specify template

Prerequisite – Prepare template

Template
Template URL: <https://us-west-2.t-prod.s3.amazonaws.com/courses/ILT-TF-200-ARCHIT/7.9.8.prod-d6e2cf0a/lab-7-capstone/scripts/Task1.yaml>

Stack description
Lab7 Task 1 template which builds VPC, supporting resources, a basic networking structure, and some Security groups for use in later tasks.

Step 2: Specify stack details

Provide a stack name
Stack name: VPCStack

The screenshots illustrate the 'Create stack' wizard in the AWS CloudFormation console, specifically the 'Step 3: Configure stack options' phase. The left sidebar shows navigation links for CloudFormation, Stacks, StackSets, Exports, Infrastructure Composer, Hooks, Registry, and Feedback.

Parameters (7)

Key	Value
AppSubnet1Param	10.0.2.0/24
AppSubnet2Param	10.0.3.0/24
DatabaseSubnet1Param	10.0.4.0/24
DatabaseSubnet2Param	10.0.5.0/24
PublicSubnet1Param	10.0.0.0/24
PublicSubnet2Param	10.0.1.0/24
VPCCIDR	10.0.0.0/16

Step 3: Configure stack options

Tags

Key	Value
No tags	

There are no tags defined for this stack.

Permissions

No permissions
There is no IAM role associated with this stack.

Stack failure options

Rollback on failure
Activated

Delete newly created resources during a rollback
Deactivated

Stack policy

No stack policy
There is no stack policy defined.

Rollback configuration

Monitoring time
-

CloudWatch alarm ARN
-

Notification options

SNS topic ARN
-

No notification options
There are no notification options defined.

Stack creation options

Timeout
-

Termination protection
Deactivated

Quick-create link

Use quick-create links to get stacks up and running quickly from the AWS CloudFormation console with the same basic configuration as this stack. Copy the URL on the link to share. [Learn more](#)

[Open quick-create link](#)

[Create changeset](#)

[Cancel](#) [Previous](#) [Submit](#)

- At the bottom of the page, choose **Submit**.

The **stack details** page is displayed.

The stack enters the **CREATE_IN_PROGRESS** status.

The screenshot shows the AWS CloudFormation console with the navigation path: CloudFormation > Stacks > VPCStack. On the left, the sidebar includes sections for Stack details (Drifts, StackSets, Exports), Infrastructure Composer (IaC generator), Hooks overview, Registry (Public extensions, Activated extensions, Publisher), Spotlight, and Feedback. The main area displays a list of stacks under the heading "Stacks (6)". The "VPCStack" entry is highlighted and shows its status as "CREATE_IN_PROGRESS". Below it, two nested stacks are listed: "NESTED LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-JaCccJcCMTMVBreRkyJ-Z-LabEnforcer-SQHFLXAWFT1H" and "NESTED LabStack-bf420498-e8ed-48d3-8727-cd8a9792f11-LabFusionGenerateRandomAlphaNumeriPasswor-QFB6KL1YMRX". The "Events" tab is selected, showing one event from April 17, 2025, at 14:45:57 UTC+0530, which is "CREATE_IN_PROGRESS" and "User Initiated".

- Choose the **Stack info** tab.
- Occasionally choose the **Overview** refresh .
- Wait for the status to change to **CREATE_COMPLETE**.

Note: This stack can take up to 5 minutes to deploy the resources.

The screenshot shows the AWS CloudFormation console with the same navigation path and sidebar as the previous image. The "VPCStack" entry is now shown as "CREATE_COMPLETE". The "Events" tab still shows the initial "CREATE_IN_PROGRESS" event, but the "Stack info" tab provides more detailed information about the stack's creation. The "Overview" section includes fields for Stack ID (arn:aws:cloudformation:us-west-2:070991923640:stack/VPCStack/921f5430-1b6c-11f0-a1d0-068258314c15), Status (CREATE_COMPLETE), and a Description: "Lab7 Task 1 template which builds VPC, supporting resources, a basic networking structure, and some Security groups for use in later tasks". Other fields include Detailed status, Status reason, Parent stack, Created time (2025-04-17 14:45:57 UTC+0530), Updated time, Deleted time, Last drift check time, Drift status (NOT_CHECKED), IAM role, Termination protection (Deactivated), and Root stack.

Task 1.4: View created resources from the console

- Choose the **Resources** tab.

The list shows the resources that are being created. CloudFormation determines the optimal order for resources to be created, such as creating the VPC before the subnet.

The screenshots show the AWS CloudFormation console interface across three different stages of resource creation:

Screenshot 1 (Top): The "Resources" tab is selected for the "VPCStack" stack. The table lists 28 resources, all of which are in the "CREATE_COMPLETE" status. These include various EC2 components like security groups, subnets, route tables, and network interfaces, along with other VPC-related resources like an internet gateway and a private route table.

Logical ID	Physical ID	Type	Status
AppInstanceSecurityGroup	sg-05ddaa622830ed0182	AWS::EC2::SecurityGroup	CREATE_COMPLETE
AppSubnet1	subnet-0663bd2f7c9c2d8b	AWS::EC2::Subnet	CREATE_COMPLETE
AppSubnet1RouteTableAssociation	rtbassoc-0dd86f6406a53290e	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE
AppSubnet2	subnet-0e53af5b5e9bcd2a01	AWS::EC2::Subnet	CREATE_COMPLETE
AppSubnet2RouteTableAssociation	rtbassoc-0cf560ad6de67794	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE
AttachGateway	IGW/vpc-061c4c3574f72bc96	AWS::EC2::VPCGatewayAttachment	CREATE_COMPLETE
DatabaseSubnet1	subnet-06a75a5165431c8d	AWS::EC2::Subnet	CREATE_COMPLETE

Screenshot 2 (Middle): The "Resources" tab is selected for the "VPCStack" stack. The table lists 28 resources, all of which are in the "CREATE_COMPLETE" status. This list includes the same set of resources as in Screenshot 1, indicating that the creation process has completed.

Logical ID	Physical ID	Type	Status
DatabaseSubnet1RouteTableAssociation	rtbassoc-08e11f7555c57289	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE
DatabaseSubnet2	subnet-09acb46503a10f3be	AWS::EC2::Subnet	CREATE_COMPLETE
DatabaseSubnet2RouteTableAssociation	rtbassoc-04b895f74c24358	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE
EFSMountTargetSecurityGroup	sg-0770dee5367cd5988	AWS::EC2::SecurityGroup	CREATE_COMPLETE
ElasticIPAddress1	54.71.1.163	AWS::EC2::EIP	CREATE_COMPLETE
ElasticIPAddress2	44.232.124.12	AWS::EC2::EIP	CREATE_COMPLETE
LabInternetGateway	igw-0ecea1e807ec0ff05	AWS::EC2::InternetGateway	CREATE_COMPLETE
LabVPC	vpc-061c4c3574f72bc96	AWS::EC2::VPC	CREATE_COMPLETE
NATGateway1	nat-031e39b18b2e848ec	AWS::EC2::NatGateway	CREATE_COMPLETE
NATGateway2	nat-01f2028f1a12d1da9	AWS::EC2::NatGateway	CREATE_COMPLETE
PrivateRouteAZ1	rtb-0afe65c65445c5d0f0j0.0.0/0	AWS::EC2::Route	CREATE_COMPLETE

Screenshot 3 (Bottom): The "Resources" tab is selected for the "VPCStack" stack. The table lists 28 resources, all of which are in the "CREATE_COMPLETE" status. This list includes the same set of resources as in Screenshot 1, indicating that the creation process has completed.

Logical ID	Physical ID	Type	Status
PrivateRouteAZ2	rtb-0fe56dc27b15a9fd5j0.0.0/0	AWS::EC2::Route	CREATE_COMPLETE
PrivateRouteTableAZ1	rtb-0afe65c65445c870f	AWS::EC2::RouteTable	CREATE_COMPLETE
PrivateRouteTableAZ2	rtb-0fe56dc27b15a9fd5	AWS::EC2::RouteTable	CREATE_COMPLETE
PublicRoute	rtb-039406706f61ca15j0.0.0/0	AWS::EC2::Route	CREATE_COMPLETE
PublicRouteTable	rtb-039406706f691ca15	AWS::EC2::RouteTable	CREATE_COMPLETE
PublicSubnet1	subnet-02d449a3de8b55a89	AWS::EC2::Subnet	CREATE_COMPLETE
PublicSubnet1RouteTableAssociation	rtbassoc-03b83652d93fc392e	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE
PublicSubnet2	subnet-0e9145ec31433c8dd	AWS::EC2::Subnet	CREATE_COMPLETE
PublicSubnet2RouteTableAssociation	rtbassoc-0a69ff9f39c532687	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE
RDSecurityGroup	sg-0417cba3d31fc4a88	AWS::EC2::SecurityGroup	CREATE_COMPLETE

- Review the resources that were deployed in the stack.
- Choose the **Events** tab and scroll through the list.

The list shows (in reverse order) the activities performed by CloudFormation, such as starting to create a resource and then completing the resource creation. Any errors encountered during the creation of the stack are listed in this tab.

VPCStack - Events (95)

Timestamp	Logical ID	Status	Detailed status	Status reason
2025-04-17 14:48:08 UTC+0530	VPCStack	CREATE_COMPLETE	-	-
2025-04-17 14:48:07 UTC+0530	PrivateRouteAZ2	CREATE_COMPLETE	-	-
2025-04-17 14:48:06 UTC+0530	PrivateRouteAZ2	CREATE_IN_PROGRESS	-	Resource creation Initiated
2025-04-17 14:48:05 UTC+0530	PrivateRouteAZ2	CREATE_IN_PROGRESS	-	-
2025-04-17 14:48:04 UTC+0530	NATGateway2	CREATE_COMPLETE	-	-

- Choose the **Outputs** tab.

Review the key-value pairs in the Outputs section. These values might be useful in later lab tasks.

VPCStack - Outputs (9)

Key	Value	Description	Export name
AppInstanceSecurityGroupID	sg-05dda622830ed0182	The Security Group ID for the Lab Instance Security Group	AppInstanceSecurityGroupID
DatabaseSubnet1CIDR	10.0.4.0/24	CIDR block for the DB Subnet in AZ a	-
DatabaseSubnet1ID	subnet-06a75a55165431c8d	The Subnet ID for the DB Subnet in AZ a	DatabaseSubnet1ID
DatabaseSubnet2CIDR	10.0.5.0/24	CIDR block for the DB Subnet in AZ b	-
DatabaseSubnet2ID	subnet-09acb46503a10f3be	The Subnet ID for the DB Subnet in AZ b	DatabaseSubnet2ID
EFSMountTargetSecurityGroupID	sg-0770dee5367cd5988	The Security Group ID for the Lab EFS Mount Target	EFSMountTargetSecurityGroupID
RDSSecurityGroupID	sg-0417cba3d31fc4a88	The Security Group ID for the Lab RDS cluster	RDSSecurityGroupID
Region	us-west-2	Lab Region	-
VPCID	vpc-061c4c3374f72bc96	The VPC ID for the lab	VPCID

You have learned to configure the stack and created all of the resources using the provided CloudFormation template.

Task 2: Create an Amazon RDS database

In this task, you create an Amazon Aurora DB instance and the subnet group to support it.

Task 2: Scenario

The network team verified the previous deployment requirements and everything looks good. The next phase of the project is to establish a secure backend database to give the database administrator (DBA) access for any migration operations before go-live.

The current database requires a lot of administrative overhead and the business has agreed to move to a managed database service. They want their architecture to be highly available, so you recommend that they set up a Multi-AZ RDS Aurora database layer. After reviewing the proposed design, the DBA has outlined the database requirements. Because of previous performance issues, the database does not require encryption, and you agree that this is the best choice.

The existing monitoring solution polls data every 10 minutes. The engineering team doesn't have room in the budget for additional features; therefore, enhanced monitoring is not required.

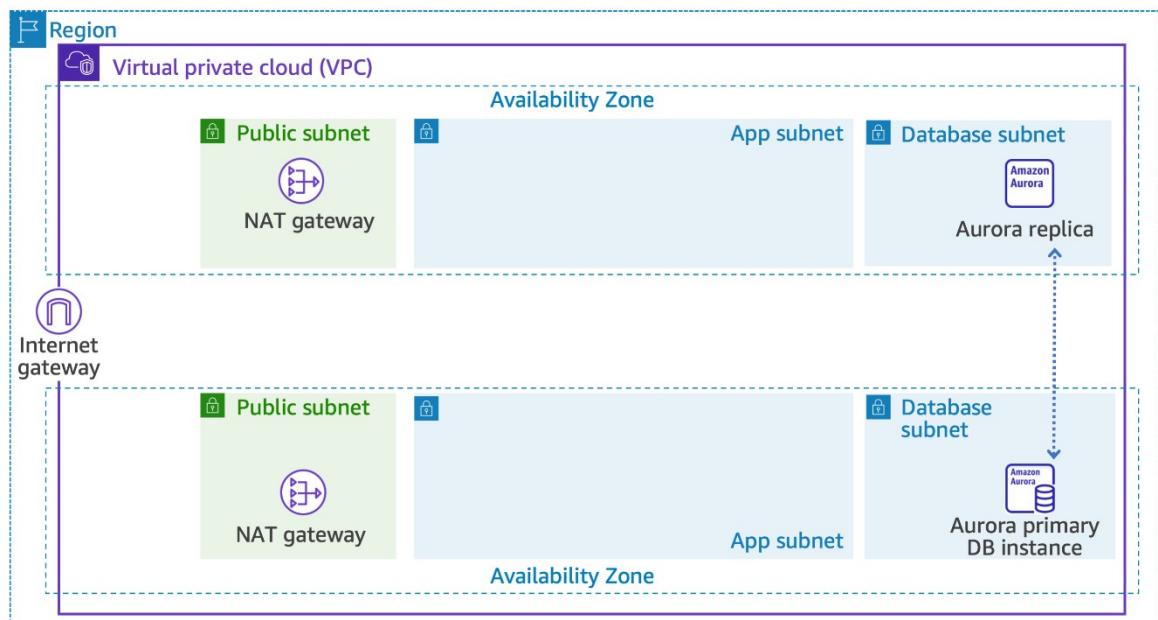


Image description: Preceding image depicts the entire architecture which will be created by the end of this lab. We have Amazon VPC to facilitate public and private networking using Public Subnets and Private Subnets. To ensure we are able to accept incoming internet traffic and allow outgoing traffic to the internet, we have Internet Gateway and NAT Gateways. For database we are using Amazon Aurora

which works across multiple availability zones (AZ)s with primary in one availability zone (AZ) and a read-replica in another availability zone (AZ).

Task 2 instructions: Create an Amazon RDS database

Task 2.1: Navigate to the Amazon RDS console

- At the top of the AWS Management Console, in the search box, search for and choose **RDS**.

The **Amazon RDS console** page is displayed.

The screenshot shows the AWS Management Console with the RDS dashboard selected. The left sidebar lists options like Databases, Query Editor, and Subnet groups. The main panel displays 'Resources' with counts for DB Instances (0/40), DB Clusters (0/40), and Parameter groups (2). It also shows allocated storage (0 TB/100 TB) and supported platforms (VPC).

Task 2.2: Create a new DB subnet group

- In the left navigation pane, choose **Subnet groups**.

The screenshot shows the 'Subnet groups' page with a count of 0. It features a 'Create DB subnet group' button in orange. A message indicates 'No db subnet groups' and 'You don't have any db subnet groups.' The left sidebar includes options like Dashboard, Databases, and Query Editor.

- Choose **Create DB subnet group**.

The **Create DB subnet group** page is displayed.

A screenshot of the AWS Cloud Map interface. At the top, there's a search bar with the placeholder "[Alt+S]" and a "Search" button. To the right of the search bar are several icons: a magnifying glass, a bell, a question mark, and a gear. Next to these are dropdown menus for "United States (Oregon)" and the user "AWSLabsUser-8JacCtzJcCfMTMVBrEtkyJ/bf420498-e8ed-48d3-8727-cd...". Below the header, a purple bar contains the text "AWS Cloud Map". Underneath the bar, the navigation path is shown: "Aurora and RDS > Subnet groups > Create DB subnet group".

Create DB subnet group

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

Subnet group details

Name

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

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

Description

(Optional)

VPC

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

Choose a VPC

- In the **Subnet group details** section, configure the following:

- Name:** Enter **AuroraSubnetGroup**.
- Description:** Enter **A 2 AZ subnet group for my database**.
- VPC:** Select **LabVPC** from the dropdown menu.

A screenshot of the AWS Cloud Map interface, identical to the one above but with a different VPC selection. The "VPC" dropdown now shows "LabVPC (vpc-061c4c3374f72bc96) 6 Subnets, 2 Availability Zones".

Create DB subnet group

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

Subnet group details

Name

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

AuroraSubnetGroup

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

Description

A 2 AZ subnet group for my database

VPC

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

LabVPC (vpc-061c4c3374f72bc96)
6 Subnets, 2 Availability Zones

- In the **Add subnets** section, configure the following:

- From the **Availability Zones** dropdown menu:
 - Select the Availability Zone ending in **a**.
 - Select the Availability Zone ending in **b**.
- From the **Subnets** dropdown menu:
 - Select the subnet with the CIDR block **10.0.4.0/24** from the Availability Zone ending in **a**.
 - Select the subnet with the CIDR block **10.0.5.0/24** from the Availability Zone ending in **b**.

Screenshot of the AWS Aurora and RDS Subnet groups creation interface.

Add subnets

Availability Zones
Choose the Availability Zones that include the subnets you want to add.
Choose an availability zone
us-west-2a X us-west-2b X

Subnets
Choose the subnets that you want to add. The list includes the subnets in the selected Availability Zones.
Select subnets
DatabaseSubnet1 Subnet ID: subnet-06a75a55165431c8d CIDR: 10.0.4.0/24 X DatabaseSubnet2 Subnet ID: subnet-09acb46505a10f3be CIDR: 10.0.5.0/24 X

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

Subnets selected (2)

Availability zone	Subnet name	Subnet ID	CIDR block
us-west-2a	DatabaseSubnet1	subnet-06a75a55165431c8d	10.0.4.0/24
us-west-2b	DatabaseSubnet2	subnet-09acb46505a10f3be	10.0.5.0/24

Create

- Choose **Create**.

A **Successfully created AuroraSubnetGroup. View subnet group** message is displayed on top of the screen.

Screenshot of the AWS Aurora and RDS Subnet groups list interface.

Aurora and RDS

- Dashboard
- Databases
- Query Editor
- Performance insights
- Snapshots
- Exports in Amazon S3
- Automated backups
- Reserved instances
- Proxies

Subnet groups

Subnet groups (1)

Successfully created AuroraSubnetGroup. [View subnet group](#)

Name	Description	Status	VPC
aurorasubnetgroup	A 2 AZ subnet group for my database	Complete	vpc-061c4c3374f72bc96

Task 2.3: Create a new Amazon Aurora database

- In the left navigation pane, choose **Databases**.

The screenshot shows the 'Aurora and RDS' section of the AWS Management Console. In the left sidebar, 'Databases' is selected. The main area is titled 'Databases (0)' with a search bar and filtering options. A message at the bottom states 'No instances found'.

- Choose **Create database**.

The **Create database** page is displayed.

- In the **Choose a database creation method** section, select **Standard create**.
- In the **Engine options** section, configure the following:
 - In **Engine type**, select **Aurora(MySQL Compatible)**.

The screenshot shows the 'Create database' configuration page. Under 'Choose a database creation method', 'Standard create' is selected. In the 'Engine options' section, 'Aurora (MySQL Compatible)' is selected. On the right side, there is a detailed description of 'Aurora MySQL-Compatible Edition' and its features.

- In the **Templates** section, select **Production**.

The screenshot shows the 'Templates' section. 'Production' is selected, with a note that it uses defaults for high availability and fast, consistent performance. 'Dev/Test' is also listed as an option.

- In the **Settings** section, configure the following:
 - **DB cluster identifier:** Enter **MyDBCluster**.
 - **Master username:** Enter **admin**.
 - **Credentials management:** Select **Self managed**
 - **Master password:** Paste the **LabPassword** value(1QaDM4T5Cxny)
 - **Confirm master password:** Paste the **LabPassword** (1QaDM4T5Cxny).

Aurora and RDS > Create database

Settings

DB cluster identifier [Info](#)
Enter a name for your DB cluster. The name must be unique across all DB clusters owned by your AWS account in the current AWS Region.

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

Credentials Settings

Master username [Info](#)
Type a login ID for the master user of your DB instance.

1 to 32 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](#)

Password strength **Very strong**

Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / \ * @

Confirm master password [Info](#)

Note: Remember your credentials.

- In the **Instance configuration** section, configure the following:
 - **DB instance class:** Select **Burstable classes (includes t classes)**.
 - From the **instance type** dropdown menu, choose **db.t3.medium**.

Instance configuration

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

DB instance class [Info](#)

Hide filters

Include previous generation classes

Serverless v2

Memory optimized classes (includes r classes)

Burstable classes (includes t classes)

db.t3.medium
2 vCPUs 4 GiB RAM Network: Up to 2,085 Mbps

- In the **Availability & durability** section, for **Multi-AZ deployment**, select **Create an Aurora Replica or Reader node in a different AZ (recommended for scaled availability)**.

Availability & durability

Multi-AZ deployment [Info](#)

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

Don't create an Aurora Replica

- In the **Connectivity** section, configure the following:

- o **Virtual private cloud (VPC):** Select **LabVPC** from the dropdown menu.
- o **DB subnet group:** Select **aurorasubnetgroup** from the dropdown menu.
- o **Public access:** Select **No**.
- o **VPC security group (firewall):** Select **Choose existing**.
- o **Existing VPC security groups:**
 - Select only the **xxxxx-RDSSecurityGroup-xxxxx** group from the dropdown menu.
 - To remove the default security group, choose the **X**.
- o Expand the **Additional configuration** section and configure the following:
 - **Database port:** Leave the configuration at the default value.

Connectivity Info

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

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

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

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

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

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

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

LabVPC (vpc-061c43374f72b96)
8 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

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

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

aurorasubnetgroup
2 Subnets, 2 Availability Zones

Public access Info

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

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

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

Choose existing
Choose existing VPC security groups

Create new
Create new VPC security group

Existing VPC security groups

Choose one or more options

VPCStack-RDSSecurityGroup-XLpPKSFjvRG X

RDS Proxy
RDS Proxy is a fully managed, highly available database proxy that improves application scalability, resiliency, and security.

Create an RDS Proxy Info
RDS automatically creates an IAM role and a Secrets Manager secret for the proxy. RDS Proxy has additional costs. For more information, see [Amazon RDS Proxy pricing](#).

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

rds-ca-rsa2048-g1 (default)
Expires: May 25, 2061

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

▼ Additional configuration

Database port Info
TCP/IP port that the database will use for application connections.

3306

- In the **Monitoring** section, clear **Enable Enhanced monitoring**.

Monitoring [Info](#)

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

Database Insights - Advanced

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

Database Insights - Standard

▼ Additional monitoring settings
Enhanced Monitoring, CloudWatch Logs and DevOps Guru

Enhanced Monitoring

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

Log exports
Select the log types to publish to Amazon CloudWatch Logs

Audit log
 Error log
 General log
 iam-db-auth-error log
 Instance log
 Slow query log

- Scroll to the bottom of the page and expand the main **Additional configuration** section.
 - o In the **Database options** section, configure the following:
 - **Initial database name:** Enter **WPDatabase**.

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

Database options

Initial database name [Info](#)

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

DB cluster parameter group [Info](#)

DB parameter group [Info](#)

Option group [Info](#)

Failover priority

- o In the **Encryption** section, clear **Enable encryption**.
- o In the **Maintenance** section, clear **Enable auto minor version upgrade**.
- o In the **Deletion protection** section, clear **Enable deletion protection**.

Encryption

Enable encryption

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

Backtrack

Backtrack lets you quickly rewind the DB cluster to a specific point in time, without having to create another DB cluster. [Info](#)

Enable Backtrack

Enabling Backtrack will charge you for storing the changes you make for backtracking.

Maintenance

Auto minor version upgrade [Info](#)

Enable auto minor version upgrade

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

Maintenance window [Info](#)

Select the period you want pending modifications or maintenance applied to the database by Amazon RDS.

Choose a window

No preference

Deletion protection

Enable deletion protection

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

- Scroll to the bottom of the screen and choose **Create database**.

Aurora MySQL-Compatible Edition

Aurora MySQL is Amazon's enterprise-class MySQL-compatible database.

Aurora MySQL offers:

- Up to five times the throughput of MySQL Community Edition
- Up to 128 TB of autoscaling SSD storage
- Six-way replication across three Availability Zones
- Up to 15 read replicas with replicas lag under 10-ms
- Automatic monitoring with failover

Maintenance

Auto minor version upgrade [Info](#)

Enable auto minor version upgrade

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

Maintenance window [Info](#)

Select the period you want pending modifications or maintenance applied to the database by Amazon RDS.

Choose a window

No preference

Deletion protection

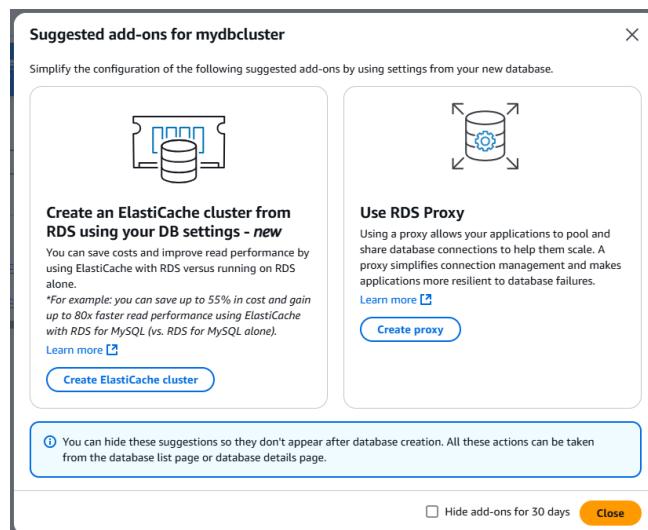
Enable deletion protection

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

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

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

- On the **Suggested add-ons for mydbcluster** pop-up window, choose **Close**.



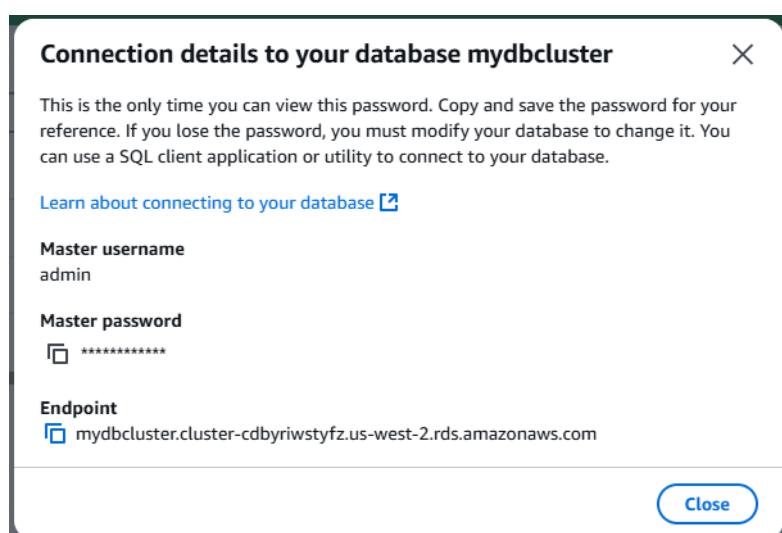
Note: Your Aurora MySQL DB cluster is in the process of launching. The cluster you configured consists of two instances, each in a different Availability Zone. The Amazon Aurora DB cluster can take up to 5 minutes to launch. Wait for the mydbcluster status to change to *Available*. You do not have to wait for the availability of the instances to continue.

A **Successfully created database mydbcluster** message is displayed on top of the screen.

The screenshot shows the AWS Aurora and RDS Databases page. A green success message at the top states: "Successfully created database mydbcluster. You can use settings from mydbcluster to simplify configuration of suggested database add-ons while we finish creating your DB for you." Below this, a table lists three databases:

DB identifier	Status	Role	Engine	Region ...	Size	Recommendations	CPU	Current activity	Last modified
mydbcluster	Available	Regional cluster	Aurora MySQL	us-west-2	2 instances	-	-	non	2023-09-12 10:45:00
mydbcluster-instance-1	Creating	Writer instance	Aurora MySQL	us-west-2b	db.t3.medium	-	-	non	2023-09-12 10:45:00
mydbcluster-instance-1-us-west-2a	Creating	Reader instance	Aurora MySQL	us-west-2a	db.t3.medium	-	-	non	2023-09-12 10:45:00

- Choose **View connection details** displayed on the success message border to save the connection details of your **mydbcluster** database to a text editor. Once the connection details are saved on a notepad, choose **Close**.



Much of this information can also be found in the next task.

Note If you notice the error “Failed to turn on enhanced monitoring for database **mydbcluster** because of missing permissions” you can safely ignore the error.

Task 2.4: Copy database metadata

- In the left navigation pane, choose **Databases**.

The screenshot shows the AWS Aurora and RDS Databases page. The left navigation pane has "Databases" selected. In the main table, the first row for **mydbcluster** is expanded, showing its status as **Available**, role as **Regional cluster**, engine as **Aurora MySQL**, region as **us-west-2**, size as **2 instances**, and two performance metrics: **10.52%** and **35.42%**. The **mydbcluster** link is highlighted in blue.

- Choose the **mydbcluster** link.

- Choose the **Connectivity & security** tab.
- Copy the **endpoint** value for the **Writer** instance to a text editor.

The screenshot shows the AWS RDS Aurora console with the 'mydbcluster' database selected. The left navigation pane is visible with options like Dashboard, Databases, Query Editor, etc. The main area shows the database details with three instances: mydbcluster-instance-1 (Writer), mydbcluster-instance-1-us-west-2a (Reader), and mydbcluster-instance-1-us-west-2b (Reader). The 'Connectivity & security' tab is active, showing the endpoints. One endpoint, 'mydbcluster.cluster-cdbyrwstyfz.us-west-2.rds.amazonaws.com', is highlighted with a blue border and has a tooltip 'Endpoint copied'.

- Choose the **Configuration** tab.

The screenshot shows the AWS RDS Aurora console with the 'mydbcluster' database selected. The left navigation pane is visible with options like Dashboard, Databases, Query Editor, etc. The main area shows the database details with three instances: mydbcluster-instance-1 (Writer), mydbcluster-instance-1-us-west-2a (Reader), and mydbcluster-instance-1-us-west-2b (Reader). The 'Configuration' tab is active, showing detailed configuration parameters like DB cluster role (Regional cluster), Engine version (8.0.mysql.Aurora.3.05.2), Resource ID (cluster-CG7FMYSZ4LQG3PUSC1LM7), and Amazon Resource Name (ARN) (arn:aws:rds:us-west-2:07091923640:cluster:mydbcluster).

- Copy the **Master username** value to a text editor.
- For **Master password**: Use the **LabPassword** value (1QaDM4T5Cxny).
- In the left navigation pane, choose **Databases**.

The screenshot shows the AWS RDS Aurora console with the 'mydbcluster' database selected. The left navigation pane is visible with options like Dashboard, Databases, Query Editor, etc. The main area shows the database details with three instances: mydbcluster-instance-1 (Writer), mydbcluster-instance-1-us-west-2a (Reader), and mydbcluster-instance-1-us-west-2b (Reader). The 'Databases' tab is active, showing the list of instances.

- Choose the **mydbcluster-instance-1** writer instance link.
- Choose the **Configuration** tab.

The screenshot shows the AWS RDS Configuration page for a database cluster named 'mydbcluster'. Under the 'Configuration' tab, the 'DB name' field is highlighted with a red box. Other visible fields include 'DB instance ID' (mydbcluster-instance-1), 'Engine version' (8.0.mysql_Aurora.3.0.52), 'RDS Extended Support' (Disabled), and 'Option groups' (default.aurora-mysql-8-0). The 'Instance class' is db.t3.medium, with 2 vCPU and 4 GB RAM. The 'Storage' section shows 'Encryption' as 'Not enabled' and 'Storage type' as '-'. The 'Monitoring' section includes 'Database Insights - Standard', 'Performance Insights' (Disabled), 'Enhanced Monitoring' (Disabled), and 'DevOps Guru' (Disabled). The 'Availability' section shows a failover priority of 1.

- Copy the **DB name** value to a text editor.

You have created the Amazon Aurora database.

Task 3: Create an Amazon EFS file system

In this task, you create and configure a custom Amazon EFS file system.

Task 3: Scenario

Example Corp. is having issues with the lead time on ordering new hardware. This is slowing down their ability to onboard new customers and expand the business. The SysOps team has a request for a storage solution built for the cloud. They need to be able to confirm that the backup policies and encryption settings meet their internal and regulatory compliance requirements. Managing time, cost, and compliance gives Example Corp. a competitive advantage.

You recommend Amazon EFS (Amazon Elastic File System (EFS) is a **fully managed, scalable, elastic, and serverless network file system** provided by Amazon Web Services (AWS). It's designed to be easy to use and provides shared file storage for Linux, macOS, and Windows-based workloads in the AWS Cloud and on-premises.) to the business as a simple, serverless, set-and-forget, elastic file system. With Amazon EFS, they can share data securely, without provisioning or managing storage.

Your task is to create an Amazon EFS file system that meets the SysOps team's requirements.

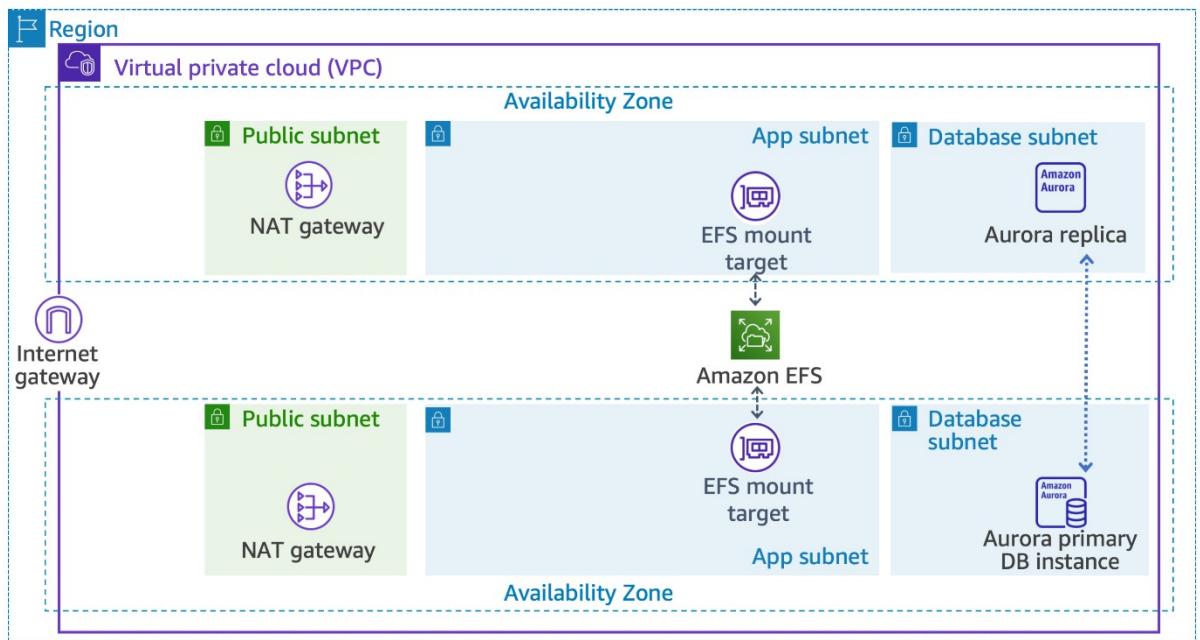


Image description: Preceding image depicts the entire architecture which will be created by the end of this lab. We have Amazon VPC to facilitate public and private networking using Public Subnets and Private Subnets. To ensure we are able to accept incoming internet traffic and allow outgoing traffic to the internet, we have Internet Gateway and NAT Gateways. There is also a fully scalable and elastic file server using Amazon EFS which can work across multiple availability zones (AZ)s. For database we are using Amazon Aurora which works across multiple availability zones (AZ)s with primary in one availability zone (AZ) and a read-replica in another availability zone (AZ).

Task 3 instructions: Creating an Amazon EFS file system

Task 3.1: Navigate to the Amazon EFS console

- At the top of the AWS Management Console, in the search box, search for and choose **EFS**.

The **Amazon Elastic File System console** page is displayed.

The screenshot shows the AWS Management Console with the EFS service selected. The left sidebar has a navigation menu with options like 'File systems' and 'Access points'. The main content area is titled 'Amazon Elastic File System' and describes it as a 'Scalable, elastic, cloud-native NFS file system'. Below this, there is a brief description of Amazon EFS and a prominent orange 'Create file system' button.

Task 3.2: Create a new file system

- Choose **Create file system**.

The **Create file system** page is displayed.

The screenshot shows the 'Create file system' wizard. Step 1 is titled 'Recommended settings'. It includes fields for 'Name - optional' (with placeholder 'Optional. Apply a name to your file system'), 'Virtual Private Cloud (VPC)' (with dropdown showing 'vpc-00b2a8e49e31269c3' and 'default'), and a note about recommended settings and pricing. At the bottom are 'Cancel', 'Customize' (highlighted in blue), and 'Create file system' buttons.

Create a file system with the recommended settings shown below by choosing Create file system. To view all settings or to customize your file system, choose Customize. [Learn more](#)

Name - *optional*
Name your file system.
Optional. Apply a name to your file system

Name can include letters, numbers, and +-=_:* symbols, up to 256 characters.

Virtual Private Cloud (VPC)
Choose the VPC where you want EC2 instances to connect to your file system.

vpc-00b2a8e49e31269c3
default

Recommended settings

Your file system is created with the following recommended settings unless you choose to customize the file system. You will be charged for storage and throughput. We recommend reviewing pricing for these features using the [AWS Pricing Calculator](#).

Cancel Customize **Create file system**

- Choose **Customize**.

The **File system settings** page is displayed.

The screenshot shows the 'File system settings' page for Amazon EFS. On the left, a sidebar lists steps: 'Step 1 File system settings' (selected), 'Step 2 Network access', 'Step 3 - optional File system policy', and 'Step 4 Review and create'. The main area is titled 'File system settings' and contains a 'General' section. In 'General', there's a 'Name - optional' field (placeholder 'Optional. Apply a name to your file system') and a note about regional vs. single zone storage. Two radio buttons are shown: 'Regional' (selected) and 'One Zone'. Both options have detailed descriptions. At the bottom right is a 'Next Step' button.

Step 1
File system settings

Step 2
Network access

Step 3 - optional
File system policy

Step 4
Review and create

File system settings

General

Name - *optional*
Name your file system.
Optional. Apply a name to your file system

File system type
Choose to either store data across multiple Availability Zones or within a single Availability Zone. [Learn more](#)

Regional
Offers the highest levels of availability and durability by storing file system data across multiple Availability Zones within an AWS Region.

One Zone
Provides continuous availability to data within a single Availability Zone within an AWS Region.

Next Step

- In the **General** section, configure the following:

- o **Name - optional:** Enter **myWPEFS**.
- o Clear **Enable automatic backups.**

File system settings

General

Name - optional

Name your file system.

File system type

Choose to either store data across multiple Availability Zones or within a single Availability Zone. [Learn more](#)

Regional

Offers the highest levels of availability and durability by storing file system data across multiple Availability Zones within an AWS Region.

One Zone

Provides continuous availability to data within a single Availability Zone within an AWS Region.

Automatic backups

Automatically backup your file system data with AWS Backup using recommended settings. Additional pricing applies. [Learn more](#)

Enable automatic backups

⚠ We recommend that you create a backup policy for your file system

- o Under **Lifecycle Management set Transition into Infrequent Access (IA) as None**
- o Under **Lifecycle Management set Transition into Archive as None**
- o Clear **Enable encryption of data at rest.**

Lifecycle management

Automatically save money as access patterns change by moving files into the Infrequent Access (IA) or Archive storage class. [Learn more](#)

Transition into Infrequent Access (IA)

Transition files to IA based on the time since they were last accessed in Standard storage.

Transition into Archive

Transition files to Archive based on the time since they were last accessed in Standard storage.

Transition into Standard

Transition files back to Standard storage based on when they are first accessed in IA or Archive storage.

Valid only when you have enabled lifecycle management to transition files into IA or Archive storage.

Encryption

Choose to enable encryption of your file system's data at rest. Uses the AWS KMS service key (aws/elasticfilesystem) by default. [Learn more](#)

Enable encryption of data at rest

Performance Settings

- o Set **Throughput mode to Bursting**
- o Expand the **Additional settings** section, configure **Performance mode to General Purpose (Recommended)**

Performance settings

Throughput mode

Choose a method for your file system's throughput limits. [Learn more](#)

Enhanced

Provides more flexibility and higher throughput levels for workloads with a range of performance requirements.

Bursting

Provides throughput that scales with the amount of storage for workloads with basic performance requirements.

▼ Additional settings

Performance mode

Set your file system's performance mode based on IOPS required. [Learn more](#)

General Purpose (Recommended)

Ideal for a variety of diverse workloads, including high performance and latency-sensitive applications

Max I/O

Designed for highly parallelized workloads that can tolerate higher latencies

- o Expand the **Tags - optional** section, configure the following:

- **Tag key:** Enter **Name**.
- **Tag value – optional:** Enter **myWPEFS**.
- Leave all other settings at their default value.

▼ Tags optional

Add tags to associate key-value pairs to your resource. [Learn more](#)

Tag key

Name

Tag value - optional

myWPEFS

[Remove tag](#)

[Add tag](#)

You can add 49 more tag(s)

[Cancel](#)

[Next](#)

- Choose **Next**.

The **Network access** page is displayed.

The screenshot shows the 'Network access' step of the EFS creation wizard. On the left, a sidebar lists steps: Step 1 File system settings, Step 2 Network access (which is selected and highlighted in blue), Step 3 - optional File system policy, and Step 4 Review and create. The main area is titled 'Network' and contains a note about enabling an EFS service-linked role. Below this, a dropdown menu is set to 'vpc-00b2a8e49e31269c3'. Under 'Mount targets', there is a table with four columns: Availability zone (set to 'us-west-2a'), Subnet ID (set to 'subnet-01437b5bdb...'), IP address (set to 'Automatic'), and Security groups. A single security group, 'sg-0a07a7c1b0405fe67', is listed under 'Security groups'.

- From the **Virtual Private Cloud (VPC)** dropdown menu, select **LabVPC**.
- For **Mount targets**, configure the following:
 - Availability zone**: Select the **Availability Zone ending in “a”** from the dropdown menu.
 - Subnet ID**: Select **AppSubnet1** from the dropdown menu.
 - Security groups**: Select **EFSMountTargetSecurityGroup** from the dropdown menu.
 - To remove the **default** Security group, choose the **X**.
 - Availability zone**: Select **Availability Zone ending in “b”** from the dropdown menu.
 - Subnet ID**: Select **AppSubnet2** from the dropdown menu.
 - Security groups**: Select **EFSMountTargetSecurityGroup** from the dropdown menu.
 - To remove the **default** Security group, choose the **X**.

Network access

Network

Virtual Private Cloud (VPC) [Learn more](#)

Choose the VPC where you want EC2 instances to connect to your file system.

vpc-061c4c3374f72bc96
LabVPC

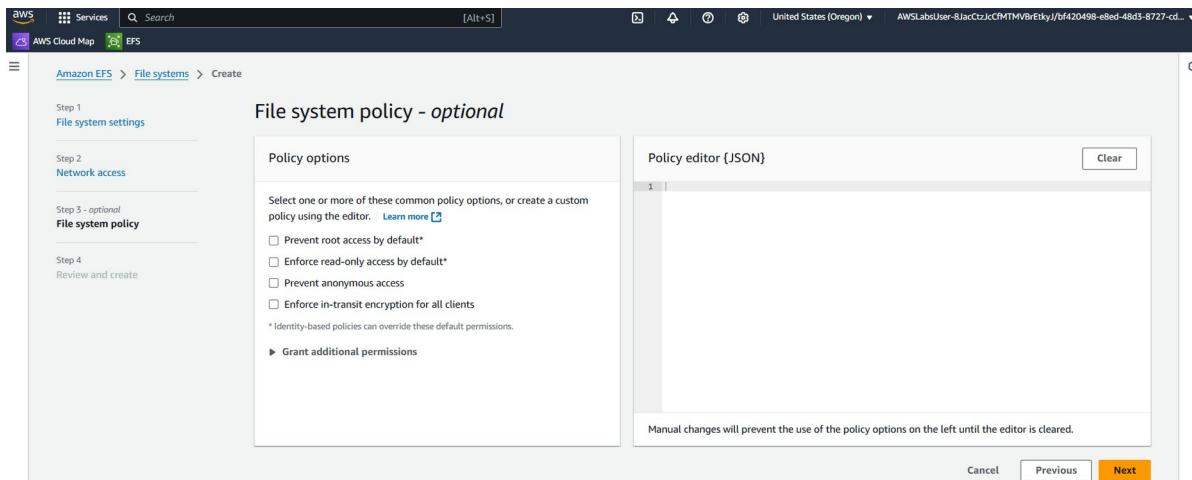
Mount targets

A mount target provides an NFSv4 endpoint at which you can mount an Amazon EFS file system. We recommend creating one mount target per Availability Zone. [Learn more](#)

Availability zone	Subnet ID	IP address	Security groups
us-west-2a	subnet-0663bdf2f7c9c2d8b	Automatic	<input type="button" value="Choose security groups"/> sg-0770dee5367cd5988 VPCstack- EFSMountTargetSecurityGroup- qPf1PlqH83ZB
us-west-2b	subnet-0e53afb5e9bcd2a01	Automatic	<input type="button" value="Choose security groups"/> sg-0770dee5367cd5988 VPCstack- EFSMountTargetSecurityGroup- qPf1PlqH83ZB

- Choose **Next**.

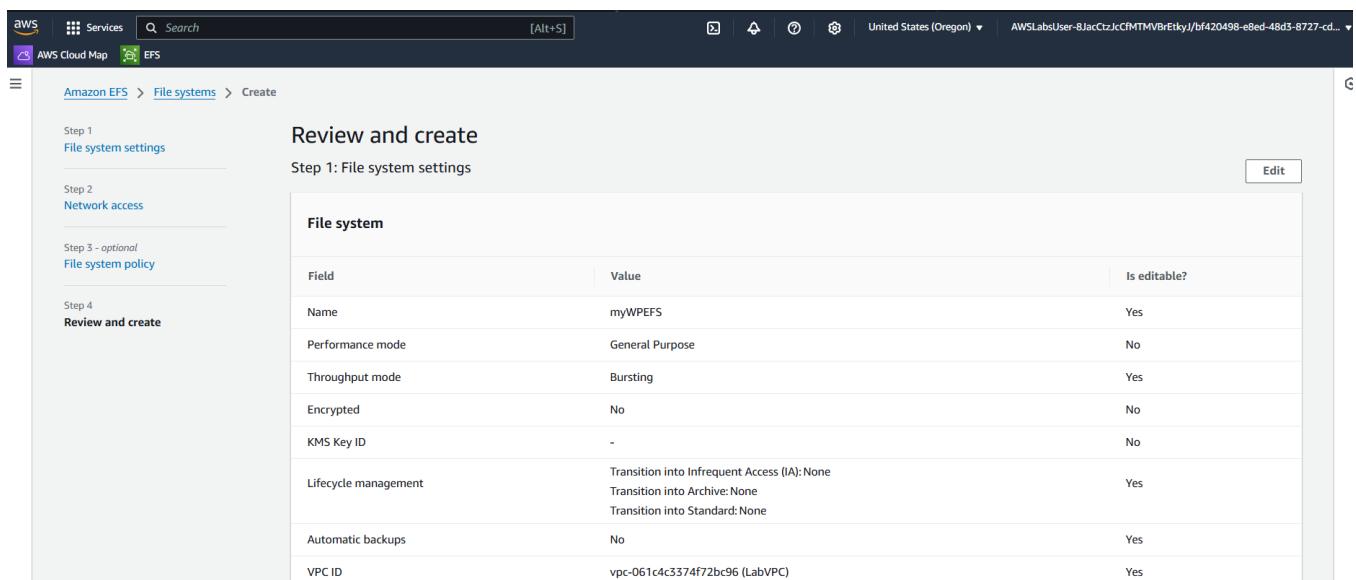
The **File system policy – optional** page is displayed.



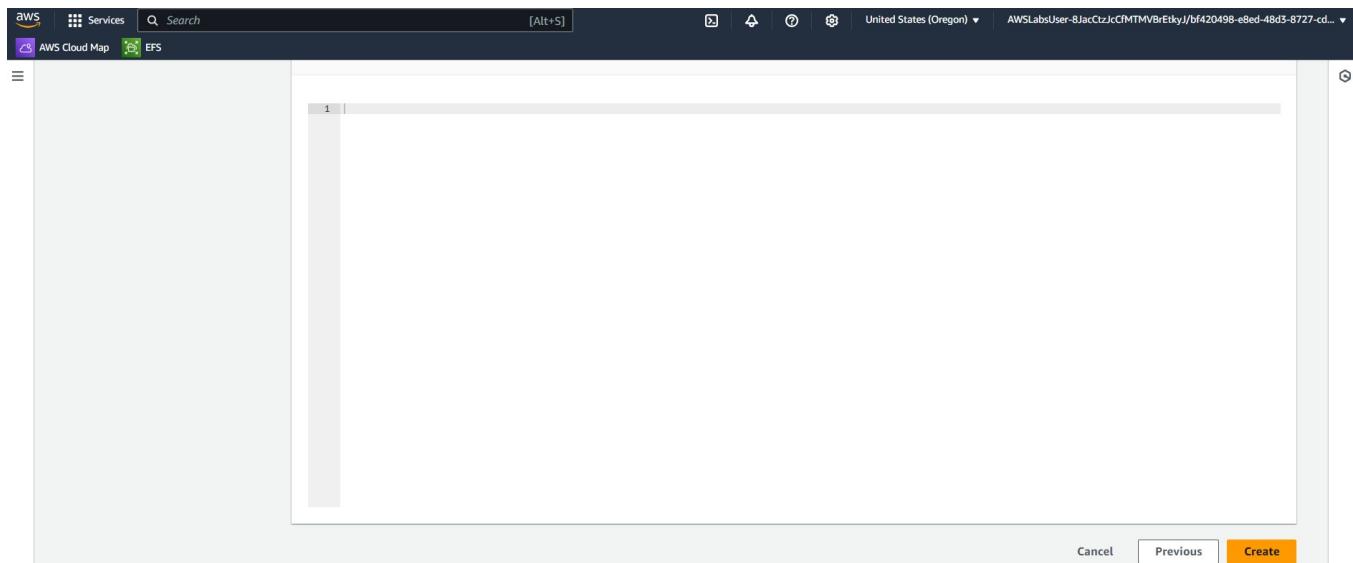
Note: Configuring this page is not necessary in this lab.

- Choose **Next**.

The **Review and create** page is displayed.



- Scroll to the bottom of the page, and choose **Create**.



A **Success!** File system (fs-xxxxxxx) is available. message is displayed on top of the screen.

The file system state displays as *Available* after several minutes.

The screenshot shows the AWS Management Console with the EFS service selected. A green banner at the top indicates success: "Success! File system (fs-03ca5f65ad019c390) is available." Below the banner, the "File systems (1)" table is displayed. The table has columns for Name, File system ID, Encrypted, Total size, Size in Standard, Size in IA, Size in Archive, Provisioned Throughput (MiB/s), File system state, and Create time. The single row shows: Name: myWPEFS, File system ID: fs-03ca5f65ad019c390, Encrypted: Unencrypted, Total size: 6.00 KB, Size in Standard: 6.00 KB, Size in IA: 0 Bytes, Size in Archive: 0 Bytes, Provisioned Throughput (MiB/s): -, File system state: Available, Create time: Thu, 2025 11:00 GMT.

Task 3.3: Copy Amazon EFS metadata

- In the left navigation pane, select **File systems**.
- Copy the **File system ID** generated for *myWPEFS* to a text editor. It has a format like *fs-a1234567*.

Congratulations, you have created the Amazon EFS.

Task 4 instructions: Create an Application Load Balancer

Task 4.1: Navigate to the Amazon EC2 console

- At the top of the AWS Management Console, in the search box, search for and choose **EC2**.

The screenshot shows the AWS EC2 Dashboard. On the left, a navigation pane lists EC2 services: Dashboard, Instances (with sub-options like Instances, Instance Types, Launch Templates, etc.), Images, and Elastic Block Store. The main content area displays the following sections:

- Resources**: A summary of Amazon EC2 resources in the United States (Oregon) Region, including Instances (running), Auto Scaling Groups, Capacity Reservations, Dedicated Hosts, Elastic IPs, Key pairs, Load balancers, Security groups, Snapshots, Instances, Placement groups, and Volumes.
- Launch instance**: Options to Launch instance or Migrate a server. Note: Your instances will launch in the United States (Oregon) Region.
- Service health**: Shows the Region as United States (Oregon) and the Status as "This service is operating normally".
- Zones**: Lists Zone name (us-west-2a) and Zone ID (usw2-az1).
- Account attributes**: Lists Default VPC (vpc-00b2a8e49e31269c3), Settings (Data protection and security, Allowed AMIs, Zones, EC2 Serial Console, Default credit specification, EC2 console preferences), and Explore AWS (Amazon GuardDuty Malware Protection, Save up to 90% on EC2 with Spot Instances, Enable Best Price-Performance with AWS Graviton2).

Task 4.2: Create a Target group

- In the left navigation pane, select **Target Group**.

The screenshot shows the AWS Lambda Target groups page. The left navigation pane includes categories like Images, Elastic Block Store, Network & Security, and Load Balancing, with Target Groups selected under Load Balancing. The main content area shows the following:

- Target groups**: A table with columns for Name, ARN, Port, Protocol, Target type, Load balancer, and VPC ID. A message states: "No target groups" and "You don't have any target groups in us-west-2". A prominent orange "Create target group" button is at the bottom.
- 0 target groups selected**: A note: "Select a target group above."

- Choose **Create target group**.

The **Specify group details** page is displayed.

Step 1
Specify group details
Step 2
Register targets

Specify group details
Your load balancer routes requests to the targets in a target group and performs health checks on the targets.

Basic configuration
Settings in this section can't be changed after the target group is created.

Choose a target type

Instances

- Supports load balancing to instances within a specific VPC.
- Facilitates the use of [Amazon EC2 Auto Scaling](#) to manage and scale your EC2 capacity.

IP addresses

- Supports load balancing to VPC and on-premises resources.
- Facilitates routing to multiple IP addresses and network interfaces on the same instance.
- Offers flexibility with microservice based architectures, simplifying inter-application communication.
- Supports IPv6 targets, enabling end-to-end IPv6 communication, and IPv4-to-IPv6 NAT.

Lambda function

- Facilitates routing to a single Lambda function.
- Accessible to Application Load Balancers only.

- In the **Basic configuration** section, configure the following:
 - For **Choose a target type**: Select **Instances**.

Specify group details

Your load balancer routes requests to the targets in a target group and performs health checks on the targets.

Basic configuration

Settings in this section can't be changed after the target group is created.

Choose a target type

Instances

- Supports load balancing to instances within a specific VPC.
- Facilitates the use of [Amazon EC2 Auto Scaling](#) to manage and scale your EC2 capacity.

- For **Target group name**: Enter **myWPTargetGroup**.

Target group name

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

Protocol : Port

Choose a protocol for your target group that corresponds to the Load Balancer type that will route traffic to it. Some protocols now include anomaly detection for the targets and you can set mitigation options once your target group is created. This choice cannot be changed after creation

1-65535

IP address type

Only targets with the indicated IP address type can be registered to this target group.

IPv4

Each instance has a default network interface (eth0) that is assigned the primary private IPv4 address. The instance's primary private IPv4 address is the one that will be applied to the target.

IPv6

Each instance you register must have an assigned primary IPv6 address. This is configured on the instance's default network interface (eth0). [Learn more](#)

- For **VPC**: Select **LabVPC** from the dropdown menu.

VPC

Select the VPC with the instances that you want to include in the target group. Only VPCs that support the IP address type selected above are available in this list.

LabVPC

vpc-061c4c3374f72bc96
IPv4 VPC CIDR: 10.0.0.0/16



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.

- In the **Health checks** section, configure the following:
 - o For **Health check path**: Enter `/wp-login.php`.

Health checks

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

Health check protocol

Health check path

Use the default path of "/" to perform health checks on the root, or specify a custom path if preferred.

Up to 1024 characters allowed.

- o Expand the **Advanced health check settings** section and configure the following:
 - **Healthy threshold**: Enter `2`.
 - **Unhealthy threshold**: Enter `10`.
 - **Timeout**: Enter `50`.
 - **Interval**: Enter `60`.

Leave the remaining settings on the page at their default values.

▼ Advanced health check settings

Health check port
The port the load balancer uses when performing health checks on targets. By default, the health check port is the same as the target group's traffic port. However, you can specify a different port as an override.

Traffic port
 Override

Healthy threshold
The number of consecutive health check successes required before considering an unhealthy target healthy.

2-10

Unhealthy threshold
The number of consecutive health check failures required before considering a target unhealthy.

2-10

Timeout
The amount of time, in seconds, during which no response means a failed health check.

seconds
2-120

Interval
The approximate amount of time between health checks of an individual target.

seconds
5-300

Success codes
The HTTP codes to use when checking for a successful response from a target. You can specify multiple values (for example, "200,202") or a range of values (for example, "200-299").

Attributes

i Certain default attributes will be applied to your target group. You can view and edit them after creating the target group.

► Tags - optional
Consider adding tags to your target group. Tags enable you to categorize your AWS resources so you can more easily manage them.

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

- Choose **Next**.

The **Register targets** page is displayed. There are no targets to register currently.

Name	Value
LAMBDA_TASK_ROOT	/var/task
AWS_LAMBDA_FUNCTION_NAME	lambda_function
AWS_LAMBDA_FUNCTION_MEMORY_SIZE	128

This function has no code. It will always return a 200 OK response.

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

- Scroll to the bottom of the page and choose **Create target group**.

A Successfully created target group: myWPTargetGroup message is displayed on top of the screen.

The screenshot shows the AWS EC2 Target Groups page. In the left navigation pane, 'Target groups' is selected under 'Load Balancing'. The main content area displays the details of the 'myWPTargetGroup'. Key information includes:

- arn:aws:elasticloadbalancing:us-west-2:070991923640:targetgroup/myWPTargetGroup/112b727be10d88d4**
- Protocol : Port**: HTTP: 80
- Protocol version**: HTTP1
- VPC**: vpc-061c4c3374f72bc96
- Targets**: 0 Total targets, 0 Healthy, 0 Unhealthy, 0 Unused, 0 Initial, 0 Draining
- Health checks**: 0 Anomalous

Below the table, there are tabs for Targets, Monitoring, Health checks, Attributes, and Tags. A note at the bottom states: "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."

Task 4.3: Create an Application Load Balancer

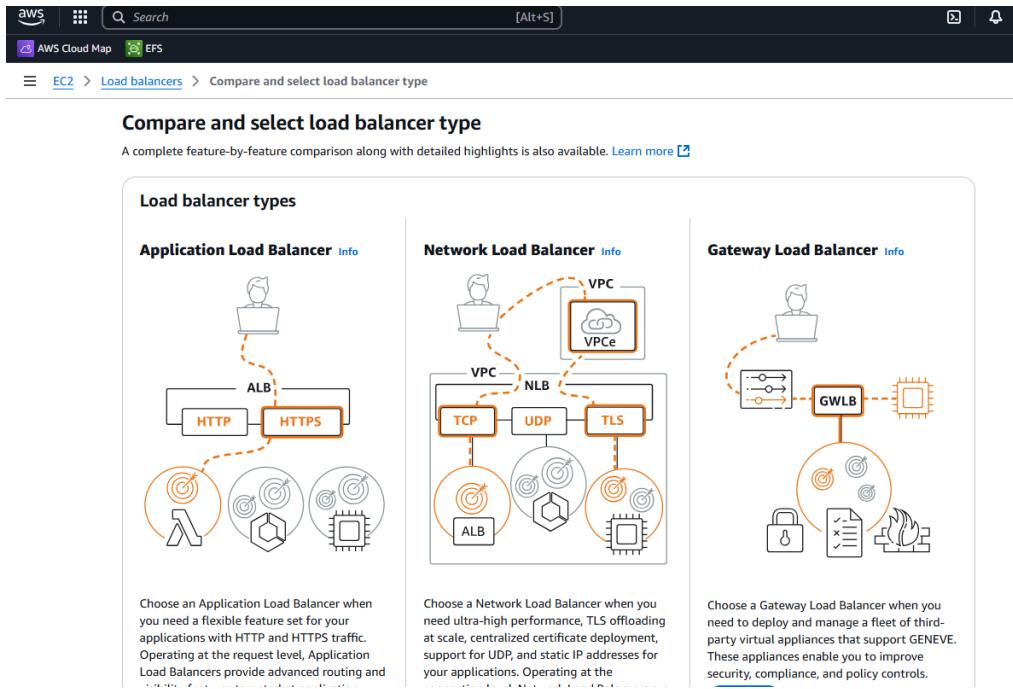
- In the left navigation pane, choose **Load Balancers**.

The screenshot shows the AWS EC2 Load Balancers page. In the left navigation pane, 'Load balancers' is selected under 'Load Balancing'. The main content area displays the 'Load balancers' section, which includes:

- Elastic Load Balancing**: Describes how it scales your load balancer capacity automatically in response to changes in incoming traffic.
- Actions** and **Create load balancer** buttons.
- A table header for 'Load balancers' with columns: Name, DNS name, State, VPC ID, Availability Zones, Type, and Date created.
- A progress bar indicating 'Loading load balancers'.

- Choose **Create load balancer**.

The **Compare and select load balancer type** page is displayed.



- In the **Load balancer types** section, for **Application Load Balancer**, choose **Create**.

The **Create Application Load Balancer** page is displayed.

The screenshot shows the "Create Application Load Balancer" page. At the top, there is a message: "Application Load Balancers now support public IPv4 IP Address Management (IPAM). You can get started with this feature by configuring IP pools in the Network mapping section." Below this, the title is "Create Application Load Balancer".

Basic configuration

- Load balancer name**: A text input field where the user has entered "myWPAAppALB".
- Scheme**: A dropdown menu with two options: "Internet-facing" (selected) and "Internal".
- Load balancer IP address type**: A dropdown menu with two options: "IPv4" (selected) and "IPv6".

- In the **Basic configuration** section, configure the following:
 - For **Load balancer name**: Enter **myWPAAppALB**.

The screenshot shows the "Create Application Load Balancer" page again, focusing on the "Basic configuration" section. The "Load balancer name" field contains "myWPAAppALB". The "Scheme" dropdown is set to "Internet-facing". The "Load balancer IP address type" dropdown is set to "IPv4".

- In the **Network mapping** section, configure the following:
 - **VPC:** Select **LabVPC** from the dropdown menu.
 - **Mappings:**
 - Select the first Availability Zone listed, and select **PublicSubnet1** from the Subnet dropdown menu.
 - Select the second Availability Zone listed, and select **PublicSubnet2** from the Subnet dropdown menu.

Network mapping [Info](#)

The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.

VPC | [Info](#)
 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](#). For a new VPC, create a VPC.

LabVPC
 vpc-061c4c3374f72bc96
 IPv4 VPC CIDR: 10.0.0.0/16

IP pools - new | [Info](#)
 You can optionally choose to configure an IPAM pool as the preferred source for your load balancers IP addresses. Create or view Pools in [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 | [Info](#)
 Select at least two Availability Zones and a subnet for each zone. A load balancer node will be placed in each selected zone and will automatically scale in response to traffic. The load balancer routes traffic to targets in the selected Availability Zones only.

us-west-2a (usw2-az1)
 Subnet
 Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.
 subnet-02d4499de8b55a89
 IPv4 subnet CIDR: 10.0.0.0/24 PublicSubnet1

us-west-2b (usw2-az2)
 Subnet
 Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.
 subnet-0c9145ec51435c8dd
 IPv4 subnet CIDR: 10.0.1.0/24 PublicSubnet2

- In the **Security groups** section, configure the following:
 - From the **Security groups** dropdown menu, select **ApplInstanceSecurityGroup**.
 - To remove the **default** security group, choose the **X**.

Security groups [Info](#)

A security group is a set of firewall rules that control the traffic to your load balancer. Select an existing security group, or you can [create a new security group](#).

Security groups
 Select up to 5 security groups

VPCStack-AppInstanceSecurityGroup-6zM299OntcbC X
 sg-05ddaa622830ed0182 VPC: vpc-061c4c3374f72bc96

- In the **Listeners and routing** section, configure the following:
 - For Listener **HTTP:80**: from the **Default action** dropdown menu, select **myWPTargetGroup**.

Listeners and routing [Info](#)

A listener is a process that checks for connection requests using the port and protocol you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets.

Listener HTTP:80
 Protocol: **HTTP** Port: **80** 1-65535

Default action | [Info](#)
 Forward to: **myWPTargetGroup** Target type: Instance, IPv4
[Create target group](#)

Listener tags - optional
 Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.

Add listener tag
 You can add up to 50 more tags.

Add listener

- Scroll to the bottom of the page and choose **Create load balancer**.

The screenshot shows the 'Create Application Load Balancer' wizard. In the 'Basic configuration' section, the load balancer is named 'myWPAppALB' and has an 'Internet-facing' scheme with an IPv4 IP address type. In the 'Network mapping' section, it's assigned to a VPC ('vpc-061c4c3574f72bc96') with a Public IPv4 IPAM pool. The 'Availability Zones and subnets' section lists two zones: 'us-west-2a' and 'us-west-2b', each with a specific subnet. The 'Listeners and routing' section shows a single listener for port 80 targeting the 'myWPTargetGroup'. The 'Service integrations' section includes CloudFront and WAF configurations. The 'Attributes' section contains a note about default attributes. The 'Creation workflow and status' section shows a step for 'Server-side tasks and status' with a note that tasks will become available after creation. At the bottom right are 'Cancel' and 'Create load balancer' buttons.

A **Successfully created load balancer: myWPAppALB** message is displayed on top of the screen.

The screenshot shows the 'myWPAppALB' load balancer details page. The left sidebar includes sections for Images, Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling. The main content area displays the load balancer's configuration: Type is Application, Scheme is Internet-facing, Status is Provisioning, Hosted zone is Z1H1FLSHABSFS, VPC is vpc-061c4c3574f72bc96, Availability Zones include us-west-2b and us-west-2a, and Load balancer ARN is arn:aws:elasticloadbalancing:us-west-2:070991923640:loadbalancer/app/myWPAppALB/5cd61d5b02143f8a. The DNS name is myWPAppALB-1778473209.us-west-2.elb.amazonaws.com (A Record). Below this, the 'Listeners and rules' tab is selected, showing one listener rule. The 'Actions' button is visible at the top right.

The load balancer is in the *Provisioning* state for a few minutes and then changes to *Active*.

The screenshot shows the 'Load balancers (1)' list page. The left sidebar includes EC2, Instances, and other EC2-related options. The main table lists the single load balancer 'myWPAppALB' with its details: Name (myWPAppALB), DNS name (myWPAppALB-1778473209...), State (Provisioning, highlighted with a red box), VPC ID (vpc-061c4c3574f72bc96), Availability Zones (2 Availability Zones), Type (application), and Date created (April 17, 2025, 23:22 (UTC+05:30)). The 'Actions' and 'Create load balancer' buttons are at the top right.

Load balancers (1/1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Name	DNS name	Status	VPC ID	Availability Zones	Type	Date created
myWPAAppALB	myWPAAppALB-1778473209...	Active	vpc-061c4c3374f72bc96	2 Availability Zones	application	April 17, 2025, 23:22 (UTC+05:30)

Load balancer: myWPAAppALB

Details

Load balancer type	Status	Load balancer IP address type
Application	Active	IPv4
Scheme	Hosted zone	Date created
Internet-facing	Z1H1FL5HABSF5	April 17, 2025, 23:22 (UTC+05:30)
Load balancer ARN	VPC	DNS name Info
arn:aws:elasticloadbalancing:us-west-2:070991923640:loadbalancer/app/myWPAAppALB/5cd61d5b02143fb8a	vpc-061c4c3374f72bc96	myWPAAppALB-1778473209.us-west-2.elb.amazonaws.com (A Record)
	Availability Zones	
	subnet-0c9145ec31433c8dd	us-west-2b (usw2-az2)
	subnet-02d449a9de8b55a89	us-west-2a (usw2-az1)

- Copy the **myWPAAppALB** load balancer **DNS name** to a text editor.

Load balancers (1/1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Name	DNS name	Status	VPC ID	Availability Zones	Type	Date created
myWPAAppALB	myWPAAppALB-1778473209...	Active	vpc-061c4c3374f72bc96	2 Availability Zones	application	April 17, 2025, 23:22 (UTC+05:30)

Load balancer: myWPAAppALB

Details

Load balancer type	Status	Load balancer IP address type
Application	Active	IPv4
Scheme	Hosted zone	Date created
Internet-facing	Z1H1FL5HABSF5	April 17, 2025, 23:22 (UTC+05:30)
Load balancer ARN	VPC	DNS name Info
arn:aws:elasticloadbalancing:us-west-2:070991923640:loadbalancer/app/myWPAAppALB/5cd61d5b02143fb8a	vpc-061c4c3374f72bc96	myWPAAppALB-1778473209.us-west-2.elb.amazonaws.com (A Record)
	Availability Zones	
	subnet-0c9145ec31433c8dd	us-west-2b (usw2-az2)
	subnet-02d449a9de8b55a89	us-west-2a (usw2-az1)

You have created the target group and an Application Load Balancer.

Task 5: Create a launch template using CloudFormation

In this task, you use a CloudFormation template to deploy the WordPress user data within an Amazon Elastic Compute Cloud (Amazon EC2) Auto Scaling launch template. The template includes the Amazon EFS mount points and the Aurora configurations.

Task 5: Scenario

Up to this point, you created the underlying network resources, the database, an Amazon EFS file system, and an Application Load Balancer. It's time to put it all together. Example Corp. already has a WordPress account, so the cloud engineer uses the settings and configuration from their existing environment to create a

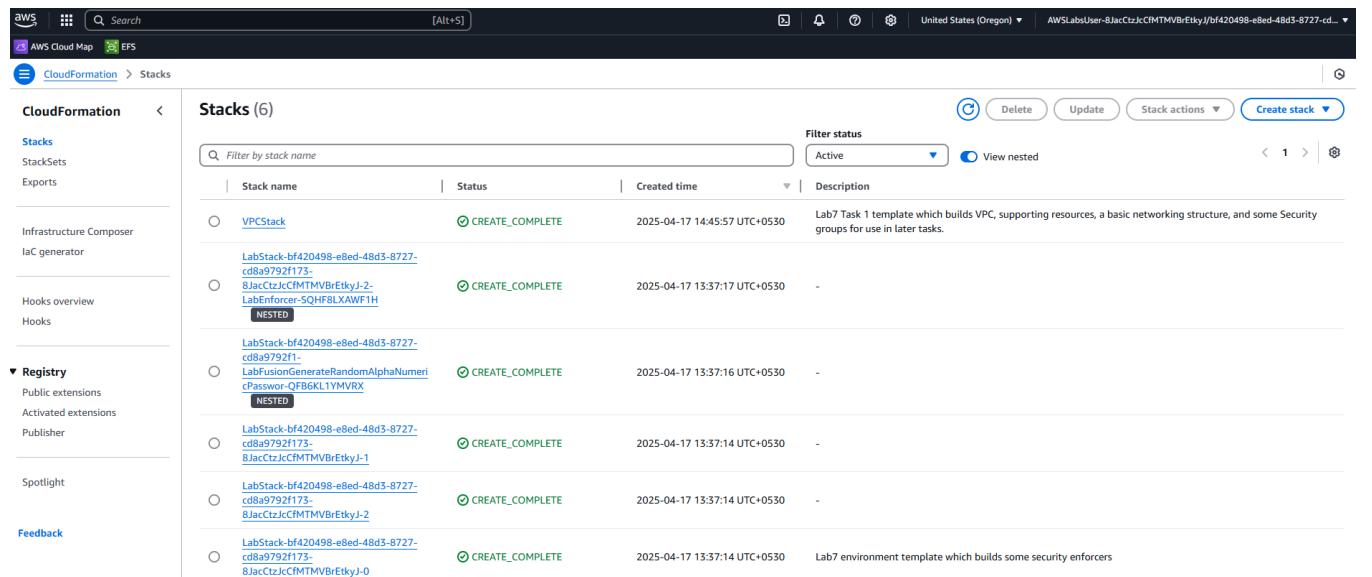
CloudFormation template. This includes all of the new resources you provisioned to set up a launch template.

Review the CloudFormation template and create all of the necessary resources. Pay special attention to the user data script. Check for any errors in the deployment and address those, if needed. After that is completed, you can create the application servers to support traffic to the new environment.

Task 5 instructions: Creating a launch template using CloudFormation

Task 5.1: Navigate to the CloudFormation console

- At the top of the AWS Management Console, in the search box, search for and choose **CloudFormation**.



The screenshot shows the AWS CloudFormation console with the 'Stacks' tab selected. The left sidebar has sections for 'Stacks', 'StackSets', 'Exports', 'Infrastructure Composer', 'Hooks', 'Registry', 'Publisher', 'Spotlight', and 'Feedback'. The main area displays a table titled 'Stacks (6)'. The columns are 'Stack name', 'Status', 'Created time', and 'Description'. The first stack, 'VPCStack', is described as 'Lab7 Task 1 template which builds VPC, supporting resources, a basic networking structure, and some Security groups for use in later tasks.' The other five stacks are nested stacks under 'LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-BJacCtzJcCMFTMvBrEtkyj-2'. The last stack, 'LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-BJacCtzJcCMFTMvBrEtkyj-0', is described as 'Lab7 environment template which builds some security enforcers'.

Stack name	Status	Created time	Description
VPCStack	CREATE_COMPLETE	2025-04-17 14:45:57 UTC+0530	Lab7 Task 1 template which builds VPC, supporting resources, a basic networking structure, and some Security groups for use in later tasks.
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-BJacCtzJcCMFTMvBrEtkyj-2	CREATE_COMPLETE	2025-04-17 13:37:17 UTC+0530	-
LabEnforcer-SQHfBLXAWF1H	NESTED		
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-BJacCtzJcCMFTMvBrEtkyj-1	CREATE_COMPLETE	2025-04-17 13:37:16 UTC+0530	-
LabFusionGenerateRandomAlphaNumeri-cPasswOr-QFB6KL1YMVRK	NESTED		
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-BJacCtzJcCMFTMvBrEtkyj-1	CREATE_COMPLETE	2025-04-17 13:37:14 UTC+0530	-
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-BJacCtzJcCMFTMvBrEtkyj-2	CREATE_COMPLETE	2025-04-17 13:37:14 UTC+0530	-
LabStack-bf420498-e8ed-48d3-8727-cd8a9792f173-BJacCtzJcCMFTMvBrEtkyj-0	CREATE_COMPLETE	2025-04-17 13:37:14 UTC+0530	Lab7 environment template which builds some security enforcers

Task 5.2: Obtain and review the CloudFormation template

- Open the context (right-click) menu on this [Task5.yaml](#) link, and choose the option to save the CloudFormation template to your computer.
- Open the downloaded file in a text editor (not a word processor, preferably in Visual Studio Code).
- Review the CloudFormation template.
- Predict what resources are created by this template.

The Task5.yaml file is an AWS CloudFormation template designed to automate the deployment of WordPress EC2 servers. It defines a stack that includes parameters for configuring the database, WordPress admin credentials, EC2 instance type, and networking components like the Application Load Balancer and Elastic File System. The template provisions key resources such as a security group for WordPress web servers, security group rules for EFS and RDS access, and a launch template that installs and configures a WordPress environment on Amazon Linux 2023. This setup

includes installing Apache, PHP, and necessary extensions, mounting an EFS volume, downloading WordPress, setting up configuration files, and enabling Opcache. The result is a ready-to-use WordPress instance hosted on AWS infrastructure.

The Task5.yaml file consists of the following:

```
AWSTemplateFormatVersion: 2010-09-09 # Specifies the version of the AWS CloudFormation template.

Description: Stack to create a launch configuration for wordpress EC2 servers. # A description of the CloudFormation stack.

Metadata: # Optional section for adding metadata to the template.
  AWS::CloudFormation::Interface: # Specifies the user interface for CloudFormation parameters in the AWS console.
    ParameterGroups: # Defines logical groupings of parameters for better user experience.
      - Label: # Label for the first parameter group.
        default: Database Parameters # The default label for this group.
        Parameters: # List of parameter names belonging to this group.
          - DatabaseName
          - DatabaseHostName
          - DatabaseUsername
          - DatabasePassword
      - Label: # Label for the second parameter group.
        default: WordPress Parameters # The default label for this group.
        Parameters: # List of parameter names belonging to this group.
          - Username
          - Password
          - Email
      - Label: # Label for the third parameter group.
        default: Other Parameters # The default label for this group.
    Parameters: # List of parameter names belonging to this group.
      - EC2ServerInstanceType

  ParameterLabels: # Provides user-friendly labels for individual parameters.
    DatabaseName:
      default: DB name # Default label for the DatabaseName parameter.
    DatabaseHostName:
      default: Database endpoint # Default label for the DatabaseHostName parameter.
    DatabaseUsername:
      default: Database User Name # Default label for the DatabaseUsername parameter.
```

```
DatabasePassword:  
    default: Database Password # Default label for the  
DatabasePassword parameter.  
EC2ServerInstanceType:  
    default: Instance Type # Default label for the  
EC2ServerInstanceType parameter.  
Username:  
    default: WordPress admin username # Default label for the  
Username parameter.  
Password:  
    default: WordPress admin password # Default label for the  
Password parameter.  
Email:  
    default: WordPress admin email address # Default label for the  
Email parameter.  
  
Parameters: # Defines the input parameters for the CloudFormation  
stack.  
  
LatestAL2023AmiId: # Parameter to store the latest Amazon Linux 2023  
AMI ID.  
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id> # Specifies  
that the parameter value is fetched from AWS Systems Manager Parameter  
Store as an EC2 Image ID.  
    Default: /aws/service/ami-amazon-linux-latest/al2023-ami-kernel-  
default-x86_64 # The default SSM parameter path to get the latest  
AL2023 AMI.  
  
DatabaseName: # Parameter for the WordPress database name.  
    AllowedPattern: ^([a-zA-Z0-9]*$) # Regular expression to allow only  
alphanumeric characters.  
    Description: The Amazon RDS database name. # Description of the  
parameter.  
    Type: String # Specifies the parameter data type as a string.  
    Default: WPDatabase # The default value for the database name.  
  
DatabaseHostName: # Parameter for the Amazon RDS database endpoint.  
    Description: Amazon RDS DB Endpoint # Description of the parameter.  
    Type: String # Specifies the parameter data type as a string.  
  
DatabaseUsername: # Parameter for the Amazon RDS username.  
    AllowedPattern: ^([a-zA-Z0-9]*$) # Regular expression to allow only  
alphanumeric characters.  
    Description: The Amazon RDS username. # Description of the  
parameter.  
    ConstraintDescription: Must contain only alphanumeric characters  
(minimum 8; maximum 16). # User-friendly message for the allowed  
pattern.
```

```
  MaxLength: 16 # Maximum length of the username.
  MinLength: 5 # Minimum length of the username.
  Type: String # Specifies the parameter data type as a string.
  Default: admin # The default value for the database username.

  DatabasePassword: # Parameter for the Amazon RDS password.
    Description: The Amazon RDS password. # Description of the
parameter.
    MaxLength: 41 # Maximum length of the password.
    MinLength: 6 # Minimum length of the password.
    NoEcho: true # Specifies that the password value should not be
displayed in the AWS console.
    Type: String # Specifies the parameter data type as a string.

  Username: # Parameter for the WordPress admin username.
    AllowedPattern: ^([a-zA-Z0-9]*$) # Regular expression to allow only
alphanumeric characters.
    Description: WordPress admin username. # Description of the
parameter.
    ConstraintDescription: Must contain only alphanumeric characters
(minimum 8; maximum 16). # User-friendly message for the allowed
pattern.
    MaxLength: 16 # Maximum length of the username.
    MinLength: 5 # Minimum length of the username.
    Type: String # Specifies the parameter data type as a string.
    Default: wpadmin # The default value for the WordPress admin
username.

  Password: # Parameter for the WordPress admin password.
    Description: WordPress admin password. # Description of the
parameter.
    MaxLength: 41 # Maximum length of the password.
    MinLength: 6 # Minimum length of the password.
    NoEcho: true # Specifies that the password value should not be
displayed in the AWS console.
    Type: String # Specifies the parameter data type as a string.

  Email: # Parameter for the WordPress admin email address.
    Type: String # Specifies the parameter data type as a string.
    Description: Email address for WordPress notifications # # Description of the parameter.

  EC2ServerInstanceType: # Parameter for the EC2 instance type for the
WordPress servers.
    Description: Amazon EC2 Instance Type # Description of the
parameter.
    Type: String # Specifies the parameter data type as a string.
    Default: t3.medium # The default EC2 instance type.
```

```
    AllowedValues: # List of allowed values for the instance type.
      - t3.small
      - t3.medium

    ALBDnsName: # Parameter for the DNS name of the Application Load
    Balancer.
      Description: Application Load Balancer DNS. # Description of the
    parameter.
      Type: String # Specifies the parameter data type as a string.

    WPElasticFileSystemID: # Parameter for the ID of the Elastic File
    System.
      Description: Elastic File System ID. # Description of the
    parameter.
      Type: String # Specifies the parameter data type as a string.

Resources: # Defines the AWS resources that will be created by the
stack.

  WebTierSecurityGroup: # Defines a security group for the web
  instances.
    Type: AWS::EC2::SecurityGroup # Specifies the resource type as an
  EC2 security group.
    Properties: # Defines the properties of the security group.
      GroupDescription: Security group for web instances # A
  description for the security group.
      GroupName: Wordpress Servers Security Group # The name of the
  security group.
      SecurityGroupIngress: # Defines the inbound rules for the
  security group.
        - IpProtocol: tcp # Allows TCP traffic.
          FromPort: 80 # Allows traffic on port 80 (HTTP).
          ToPort: 80 # Allows traffic on port 80 (HTTP).
          SourceSecurityGroupId: !ImportValue
AppInstanceSecurityGroupID # Allows traffic from the security group of
the application load balancer (imported from another stack).
  VpcId: # Specifies the VPC ID for the security group.
    !ImportValue VPCID # Imports the VPC ID from another stack.

  EFSSecurityGroupInboundRule: # Defines an inbound rule for the EFS
  security group.
    Type: AWS::EC2::SecurityGroupIngress # Specifies the resource type
  as an EC2 security group inbound rule.
    Properties: # Defines the properties of the inbound rule.
      IpProtocol: tcp # Allows TCP traffic.
      FromPort: 2049 # Allows traffic on port 2049 (NFS).
      ToPort: 2049 # Allows traffic on port 2049 (NFS).
```

```
    SourceSecurityGroupId: !Ref WebTierSecurityGroup # Allows traffic
from the web tier security group defined in this stack.
    GroupId: !ImportValue EFSMountTargetSecurityGroupID # Specifies
the ID of the EFS mount target security group (imported from another
stack).

    RDSSecurityGroupInboundRule: # Defines an inbound rule for the RDS
security group.
        Type: AWS::EC2::SecurityGroupIngress # Specifies the resource type
as an EC2 security group inbound rule.
        Properties: # Defines the properties of the inbound rule.
            IpProtocol: tcp # Allows TCP traffic.
            FromPort: 3306 # Allows traffic on port 3306 (MySQL/MariaDB).
            ToPort: 3306 # Allows traffic on port 3306 (MySQL/MariaDB).
            SourceSecurityGroupId: !Ref WebTierSecurityGroup # Allows traffic
from the web tier security group defined in this stack.
        GroupId: !ImportValue RDSSecurityGroupID # Specifies the ID of
the RDS security group (imported from another stack).

# Lab Launch Template

    LabLaunchTemplate: # Defines an EC2 launch template named
'LabLaunchTemplate'.
        Type: AWS::EC2::LaunchTemplate # Specifies the resource type as an
EC2 Launch Template.
        Properties: # Defines the properties of the launch template.
            LaunchTemplateName: LabLaunchTemplate # Sets the name of the
launch template.
            LaunchTemplateData: # Defines the configuration data used to
launch EC2 instances.
                UserData: # Specifies the user data script to be executed on
instance launch.
                    "Fn::Base64": # Encodes the following multi-line string in
Base64 format.
                        !Sub
                            - | # Substitutes variables within the following multi-line
string.
                                #!/bin/bash -xe # Shebang line to execute the script with
error checking (exit on error).
                                DB_NAME=${DatabaseName} # Sets a shell variable DB_NAME
to the value of the CloudFormation parameter DatabaseName.
                                DB_HOSTNAME=${DatabaseHostName} # Sets a shell variable
DB_HOSTNAME to the value of the CloudFormation parameter
DatabaseHostName.
                                DB_USERNAME="${DatabaseUsername}" # Sets a shell variable
DB_USERNAME to the value of the CloudFormation parameter
DatabaseUsername.
```

```
        DB_PASSWORD="${DatabasePassword}" # Sets a shell variable  
DB_PASSWORD to the value of the CloudFormation parameter  
DatabasePassword.  
        WP_ADMIN=${Username} # Sets a shell variable WP_ADMIN to  
the value of the CloudFormation parameter Username.  
        WP_PASSWORD=${Password} # Sets a shell variable  
WP_PASSWORD to the value of the CloudFormation parameter Password.  
        WP_EMAIL=${Email} # Sets a shell variable WP_EMAIL to the  
value of the CloudFormation parameter Email.  
        LB_HOSTNAME=${ALBDnsName} # Sets a shell variable  
LB_HOSTNAME to the value of the CloudFormation parameter ALBDnsName.  
  
        #package installation  
        dnf update -y # Updates all installed packages on the  
Amazon Linux system without prompting for confirmation.  
        dnf install httpd gcc-c++ php wget php-  
{pear,devel,fpm,mysqli,cgi,common,curl,mbstring,gd,mysqlnd,gettext,bcma-  
th,json,xml,fpm,intl,zip,devel,opcache} -y # Installs the Apache web  
server (httpd), C++ compiler (gcc-c++), PHP, wget, and various  
essential PHP extensions without prompting for confirmation.  
  
        systemctl enable nfs-server.service # Enables the Network  
File System (NFS) server service to start automatically on boot.  
        systemctl start nfs-server.service # Starts the NFS  
server service immediately.  
  
        mkdir -p /var/www/wordpress # Creates the directory  
/var/www/wordpress if it doesn't exist, including any necessary parent  
directories.  
        mount -t nfs4 -o  
nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2 $  
{ElasticFileSystem}.efs.${AWS::Region}.amazonaws.com:/  
/var/www/wordpress # Mounts the specified Elastic File System volume to  
the /var/www/wordpress directory using NFS version 4.1 with specific  
mount options.  
  
        ## create site config  
        cat <<EOF >/etc/httpd/conf.d/wordpress.conf # Creates a  
new Apache configuration file named wordpress.conf in the  
/etc/httpd/conf.d/ directory.  
        ServerName 127.0.0.1:80 # Sets the server name for the  
virtual host to localhost on port 80.  
        DocumentRoot /var/www/wordpress # Sets the document root  
for the virtual host to the /var/www/wordpress directory.  
        <Directory /var/www/wordpress> # Defines the directory  
directives for the /var/www/wordpress directory.  
        Options Indexes FollowSymLinks # Allows directory  
indexing and following of symbolic links within the directory.
```

```
        AllowOverride All # Allows .htaccess files to override
server configurations within the directory.
        Require all granted # Grants access to all clients for
this directory.
    </Directory>
EOF

        pecl install igbinary # Installs the igbinary PHP
extension, which provides a faster PHP serializer.
        cd /tmp # Changes the current working directory to /tmp.

## install WordPress and WP CLI
curl -o /bin/wp https://raw.githubusercontent.com/wp-
cli/builds/gh-pages/phar/wp-cli.phar # Downloads the WordPress Command-
Line Interface (WP-CLI) as an executable phar file and saves it as
/bin/wp.
        chmod +x /bin/wp # Makes the /bin/wp file executable.
        wget -P /tmp/ https://wordpress.org/latest.tar.gz #
Downloads the latest WordPress archive to the /tmp/ directory.
        tar -vxzf latest.tar.gz -C /var/www/ # Extracts the
contents of the WordPress archive to the /var/www/ directory.
        cp /var/www/wordpress/wp-config-sample.php
/var/www/wordpress/wp-config.php # Copies the sample WordPress
configuration file to the active configuration file.
        cd /var/www/wordpress/ # Changes the current working
directory to the WordPress installation directory.
        sed -i 's/database_name_here/'"${DB_NAME}"'/' wp-config.php
# Replaces the placeholder 'database_name_here' in wp-config.php with
the actual database name.
        sed -i 's/username_here/'"${DB_USERNAME}"'/' wp-config.php
# Replaces the placeholder 'username_here' in wp-config.php with the
database username.
        sed -i 's/password_here/'"${DB_PASSWORD}"'/' wp-config.php
# Replaces the placeholder 'password_here' in wp-config.php with the
database password.
        sed -i 's/localhost/'"${DB_HOSTNAME}"'/' wp-config.php #
Replaces 'localhost' in wp-config.php with the actual database
hostname.

# install WordPress if not installed
# use public alb host name if wp domain name was empty
if ! $(wp core is-installed --allow-root); then # Checks
if WordPress is already installed using WP-CLI.
        wp core install --url="http://${LB_HOSTNAME}" --
title='Wordpress on AWS' --admin_user="$WP_ADMIN" --
admin_password="$WP_PASSWORD" --admin_email="$WP_EMAIL" --allow-root #
Installs WordPress using WP-CLI with the provided database and admin
credentials.
```

```
wp plugin install w3-total-cache --allow-root #
Installs the W3 Total Cache plugin using WP-CLI.
        chown -R apache:apache /var/www/wordpress # Recursively
changes the ownership of the /var/www/wordpress directory and its
contents to the apache user and group.
        chmod u+wrx /var/www/wordpress/wp-content/* # Adds
read, write, and execute permissions for the owner to all files and
directories within the wp-content directory.
        if [ ! -f /var/www/wordpress/opcache-instanceid.php ];
then # Checks if the opcache-instanceid.php file does not exist.
        wget -P /var/www/wordpress/
https://s3.amazonaws.com/aws-refarch/wordpress/latest/bits/opcache-
instanceid.php # Downloads the opcache-instanceid.php script to the
WordPress directory.
        fi
        fi
RESULT=$? # Captures the exit code of the last executed
command.
        echo $RESULT # Prints the exit code of the last command.
        if [ $RESULT -eq 0 ]; then # Checks if the exit code of
the last command was 0 (success).
        touch /var/www/wordpress/wordpress.initialized #
Creates an empty file named wordpress.initialized in the WordPress
directory to indicate successful initialization.
        else # If the last command failed.
        touch /var/www/wordpress/wordpress.failed # Creates an
empty file named wordpress.failed in the WordPress directory to
indicate a failed initialization.
        fi
        fi
## install opcache
# create hidden opcache directory locally & change owner
to apache

mkdir -p /var/www/.opcache # Creates a hidden directory
named .opcache within /var/www/.
# enable opcache in /etc/php-7.0.d/10-opcache.ini
sed -i
's/;opcache.file_cache=.*/opcache.file_cache=\var\www\/.opcache/'
/etc/php.d/10-opcache.ini # Modifies the PHP-FPM opcache configuration
file to set the opcache file cache directory.
        sed -i
's/opcache.memory_consumption=.*/opcache.memory_consumption=512/'
/etc/php.d/10-opcache.ini # Modifies the PHP-FPM opcache configuration
file to set the opcache memory consumption to 512MB.
# download opcache-instance.php to verify opcache status
if [ ! -f /var/www/wordpress/opcache-instanceid.php ];
then # Checks if the opcache-instanceid.php file does not exist.
```

```

        wget -P /var/www/wordpress/
https://s3.amazonaws.com/aws-refarch/wordpress/latest/bits/opcache-
instanceid.php # Downloads the opcache-instanceid.php script to the
WordPress directory.
    fi

        # Turn on web server
        systemctl enable httpd # Enables the Apache web server
service to start automatically on boot.
        systemctl start httpd # Starts the Apache web server
service immediately.
        - ElasticFileSystem: !Ref WPElasticFileSystemID #
Substitutes the value of the WPElasticFileSystemID parameter.
        ALBDNSName: !Ref ALBDnsName # Substitutes the value of
the ALBDnsName parameter.
        ImageId: !Ref LatestAL2023AmiId # Specifies the AMI ID to be
used for launching instances, referencing the LatestAL2023AmiId
parameter.
        SecurityGroupIds: # Specifies the security groups to associate
with the launched EC2 instances.
        - !Ref WebTierSecurityGroup # References the
WebTierSecurityGroup resource defined in this template.
        InstanceType: !Ref EC2ServerInstanceType # Specifies the EC2
instance type to be used, referencing the EC2ServerInstanceType
parameter.

Outputs: # Defines the output values of the CloudFormation stack.
        WPLaunchTemplate: # Defines an output named WPLaunchTemplate.
        Description: 'Launch Template for WordPress' # A description for
the output.
        Value: # The value of the output.
        !Ref LabLaunchTemplate # The physical ID of the LabLaunchTemplate
resource.

```

Task 5.3: Create the CloudFormation stack

- Choose **Create stack**.

Note: If the console starts you on the Stacks page instead of the Amazon CloudFormation landing page, then you can get to the Create stack page in two steps.

- Choose the **Create stack** dropdown menu.
- Choose **With new resources (standard)**.

The screenshot shows the AWS CloudFormation Stacks page. The left sidebar has sections for CloudFormation (Stacks, StackSets, Exports), Infrastructure Composer, Hooks overview, Registry (Public extensions, Activated extensions, Publisher), Spotlight, and Feedback. The main area displays a table titled 'Stacks (6)' with one item: 'VPCStack' (Status: CREATE_COMPLETE, Created time: 2025-04-17 14:45:57 UTC+0530). A description below the table states: 'Lab7 Task 1 template which builds VPC, supporting resources, a basic networking structure, and some Security groups for use in later tasks.' Action buttons at the top right include Delete, Update, Stack actions, Create stack, With new resources (standard), and With existing resources (import resources).

The **Create stack** page is displayed.

The screenshot shows the 'Create stack' wizard, Step 1: Prerequisite – Prepare template. The left sidebar is identical to the previous screenshot. The main area shows a step navigation: Step 1 (Create stack) is selected, followed by Step 2 (Specify stack details), Step 3 (Configure stack options), and Step 4 (Review and create). The 'Prerequisite – Prepare template' section contains two options: 'Choose an existing template' (selected) and 'Build from Infrastructure Composer'. Below this is the 'Specify template' section, which includes an 'Amazon S3 URL' input field containing 'https://'. A note says 'S3 URL: Will be generated when a URL is provided'. At the bottom right are 'Cancel' and 'Next' buttons.

- Configure the following:
 - Select **Choose an existing template**.
 - Select **Amazon S3 URL**.
 - Copy the **Task5TemplateUrl** value (<https://us-west-2-tcprod.s3.amazonaws.com/courses/ILT-TF-200-ARCHIT/v7.9.8.prod-d6e2cf0a/lab-7-capstone/scripts/Task5.yaml>) and paste it in the **Amazon S3 URL** text box.
 - Choose **Next**.

The screenshot shows the 'Create stack' wizard, Step 1: Prerequisite – Prepare template. The left sidebar is identical to the previous screenshots. The main area shows the 'Specify template' section with the 'Amazon S3 URL' input field now containing the value 'https://us-west-2-tcprod.s3.amazonaws.com/courses/ILT-TF-200-ARCHIT/v7.9.8.prod-d6e2cf0a/lab-7-capstone/scripts/Task5.yaml'. A note says 'S3 URL: https://us-west-2-tcprod.s3.amazonaws.com/courses/ILT-TF-200-ARCHIT/v7.9.8.prod-d6e2cf0a/lab-7-capstone/scripts/Task5.yaml'. At the bottom right are 'Cancel' and 'Next' buttons.

The **Specify stack details** page is displayed.

CloudFormation <
Stacks
StackSets
Exports

Infrastructure Composer
IaC generator

Hooks overview
Hooks

Registry
Public extensions
Activated extensions
Publisher

Spotlight

Feedback

Specify stack details

Provide a stack name

Stack name

Enter a stack name

Stack name must contain only letters (a-z, A-Z), numbers (0-9) and hyphens (-) and start with a letter. Max 128 characters. Character count: 0/128.

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Database Parameters

DB name

The Amazon RDS database name.

WPDatabase

Database endpoint

Amazon RDS DB Endpoint

Enter String

Database User Name

The Amazon RDS username.

admin

Database Password

The Amazon RDS password.

- Set the **Stack name** as **WPLaunchConfigStack**.

Specify stack details

Provide a stack name

Stack name

WPLaunchConfigStack

Stack name must contain only letters (a-z, A-Z), numbers (0-9) and hyphens (-) and start with a letter. Max 128 characters. Character count: 19/128.

- Configure the following **parameters**:

- DB name:** Paste the **initial database name** you copied in Task 2 (WPDatabase)

Note: Make sure that you paste the *initial database name*, not the cluster name.

- Database endpoint:** Paste the **writer endpoint** you copied in Task 2. (mydbcluster.cluster-cdbbyriwstyfz.us-west-2.rds.amazonaws.com)
- Database User Name:** Paste the **Master username** you copied in Task 2. (admin)
- Database Password:** Paste the **Master password** you copied in Task 2. (1QaDM4T5Cxny)
- WordPress admin username:** Defaults to **wpadmin**.
- WordPress admin password:** Paste the **LabPassword** value (1QaDM4T5Cxny)

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Database Parameters

DB name

The Amazon RDS database name.

Database endpoint

Amazon RDS DB Endpoint

Database User Name

The Amazon RDS username.

Database Password

The Amazon RDS password.

WordPress Parameters

WordPress admin username

WordPress admin username.

WordPress admin password

WordPress admin password.

- o **WordPress admin email address:** Input a valid email address. (email of your choice)
- o **Instance Type:** Leave the default value of **t3.medium**.
- o **ALBDnsName:** Paste the **DNS name** value you copied in Task 4. (myWPApplB-1778473209.us-west-2.elb.amazonaws.com)
- o **LatestAL2023Amild:** Leave the default value.
- o **WPElasticFileSystemID:** Paste the **File system ID** value you copied in Task 3. (fs-03ca5f65ad019c390)

WordPress admin email address

Email address for WordPress notifications

Other Parameters

Instance Type

Amazon EC2 Instance Type



ALBDnsName

Application Load Balancer DNS.

LatestAL2023Amild

WPElasticFileSystemID

Elastic File System ID.

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

- Choose **Next**.

The **Configure stack options** page is displayed. You can use this page to specify additional parameters. You can browse the page, but leave settings at their default values.

Configure stack options

Tags - optional
Tags (key-value pairs) are used to apply metadata to AWS resources, which can help in organising, identifying and categorising those resources. You can add up to 50 unique tags for each stack.
No tags associated with the stack.

Add new tag
You can add 50 more tag(s)

Permissions - optional
Specify an existing AWS Identity and Access Management (IAM) service role that CloudFormation can assume.

IAM role - optional
Choose the IAM role for CloudFormation to use for all operations performed on the stack.

Stack failure options

Behaviour on provisioning failure
Specify the roll-back behaviour for a stack failure. [Learn more](#)

Roll back all stack resources
Roll back the stack to the last known stable state.

Preserve successfully provisioned resources
Preserves the state of successfully provisioned resources, while rolling back failed resources to the last known stable state. Resources without a last known stable state will be deleted upon the next stack operation.

Delete newly created resources during a rollback
Specify whether resources that were created during a failed operation should be deleted regardless of their deletion policy. [Learn more](#)

Use deletion policy
Retains or deletes created resources according to their attached deletion policy.

Delete all newly created resources
Deletes created resources during a rollback regardless of their attached deletion policy.

Additional settings
You can set additional options for your stack, like notification options and a stack policy. [Learn more](#)

Stack policy - optional
Defines the resources that you want to protect from unintentional updates during a stack update.

Rollback configuration - optional
Specify alarms for CloudFormation to monitor when creating and updating the stack. If the operation breaches an alarm threshold, CloudFormation rolls it back.

Notification options - optional
Specify a new or existing Amazon Simple Notification Service topic where notifications about stack events are sent.

Stack creation options - optional
Specify the timeout and termination protection options for stack creation.

- Choose **Next**.

The **Review and create** page is displayed. This page is a summary of all settings.

Review and create

Step 1: Specify template

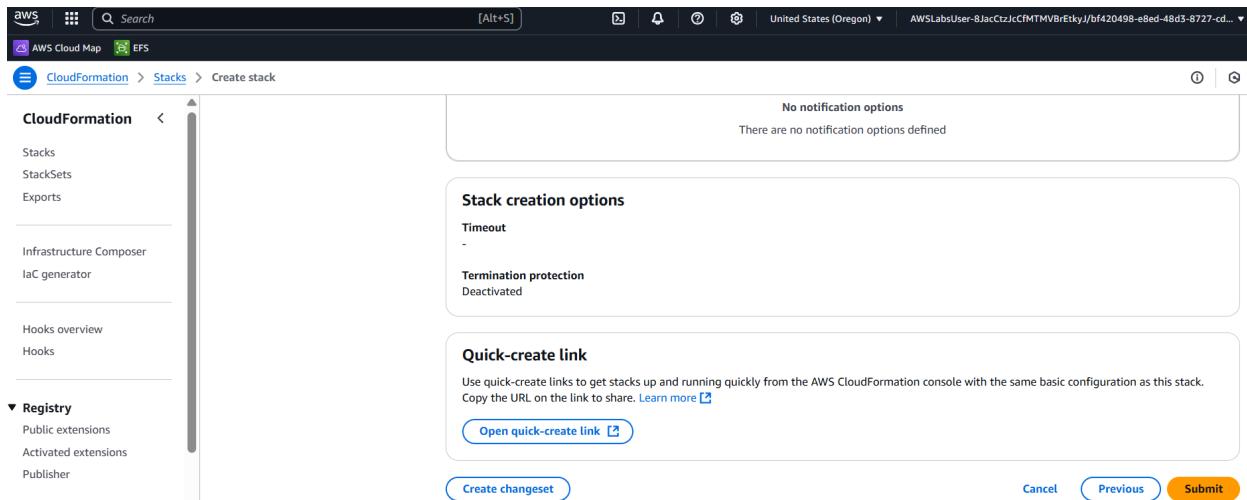
Prerequisite – Prepare template

Template
Template is ready

Step 2: Specify stack details

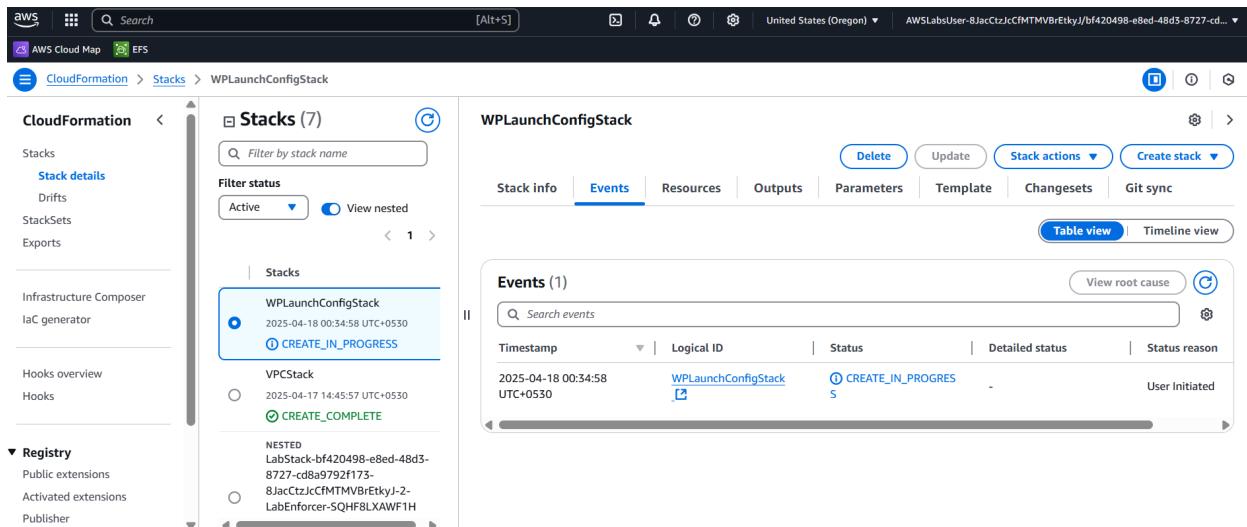
Provide a stack name

- Scroll to the bottom of the page and choose **Submit**.



The **stack details** page is displayed.

The stack enters the **CREATE_IN_PROGRESS** status.



- Choose the **Stack info** tab.
- Occasionally choose the **Overview** refresh .
- Wait for the stack status to change to **CREATE_COMPLETE**.

CloudFormation > **Stacks** > **WPLaunchConfigStack**

Stacks (7)

WPLaunchConfigStack

Stack info | Events | Resources | Outputs | Parameters | Template | Changesets | Git sync

Overview

Stack ID: arn:aws:cloudformation:us-west-2:070991923640:stack/WPLaunchConfigStack/db39cd60-1bbe-11f0-999c-068bd58d0f11

Description: Stack to create a launch configuration for wordpress EC2 servers.

Status: CREATE_COMPLETE

Detailed status: -

Status reason: -

Root stack: -

Parent stack: -

Created time: 2025-04-18 00:34:58 UTC+0530

Updated time: -

Note: This stack can take up to 5 minutes to deploy the resources.

Task 5.4: View created resources from the console

- Choose the **Resources** tab.

The list shows the resources that are created.

CloudFormation > **Stacks** > **WPLaunchConfigStack**

Stacks (7)

WPLaunchConfigStack

Resources (4)

Logical ID	Physical ID	Type	Status	Module
EFSSecurityGroupInboundRule	sgr-0ff13ad3102271317	AWS::EC2::SecurityGroup Ingress	CREATE_COMPLETE	-
LabLaunchTemplate	lt-0698813684aae3213	AWS::EC2::LaunchTemplate	CREATE_COMPLETE	-
RDSSecurityGroupInboundRule	sgr-0732ad99cdfe69504	AWS::EC2::SecurityGroup Ingress	CREATE_COMPLETE	-
WebTierSecurityGroup	sg-0db537674ed7aebe	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-

You have created the stack using the provided CloudFormation template.

Task 6: Create the application servers by configuring an Auto Scaling group and a scaling policy

In this task, you create the WordPress application servers by configuring an Auto Scaling group and a scaling policy.

Task 6: Scenario

Now that your template has been deployed, it's time to create the application servers and auto scaling mechanism. In the previous task, you chose to implement auto scaling for the app servers to meet the scaling requirements of the project plan.

Create an Auto Scaling group and scaling policy. Verify that the instance status is healthy, and test the load balancer availability. Hand the environment over to the engineering teams to validate full functionality when the unit test is complete, and ask them for feedback.

When the team is satisfied, ask them to migrate the WordPress site to the AWS environment and test the app functionality. A common practice is to introduce examples of failure. If the app is working as intended, you would introduce some examples of failure, for example, delete an app server or roll back the database to a recent backup. This step is not a part of this lab.

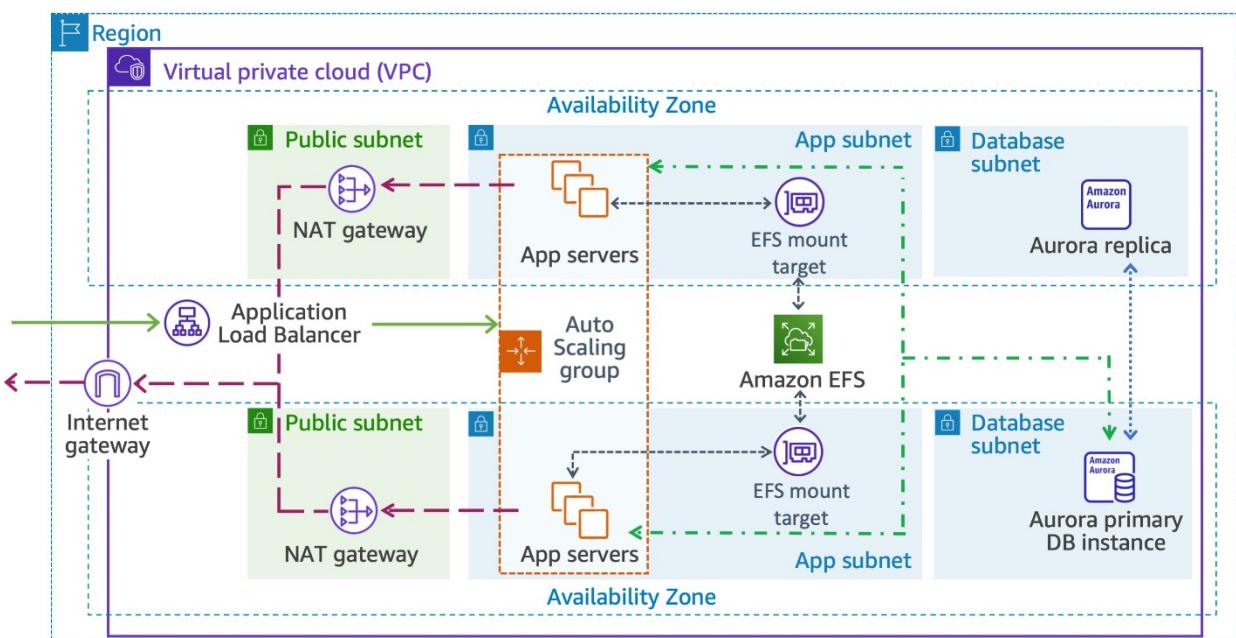


Image description: Preceding image depicts the entire architecture which will be created by the end of this lab. We have Amazon VPC to facilitate public and private networking using Public Subnets and Private Subnets. To ensure we are able to accept incoming internet traffic and allow outgoing traffic to the internet, we have

Internet Gateway and NAT Gateways. In-order to facilitate high-availability (HA) for our architecture we have Application Load Balancer (ALB) and Auto-Scaling Group (ASG). There is also a fully scalable and elastic file server using Amazon EFS which can work across multiple availability zones (AZ)s. For database we are using Amazon Aurora which works across multiple availability zones (AZ)s with primary in one availability zone (AZ) and a read-replica in another availability zone (AZ).

Task 6 instructions: Create the application servers by configuring an Auto Scaling group and a scaling policy

Task 6.1: Create an Auto Scaling group

- At the top of the AWS Management Console, in the search box, search for and choose **EC2**.

The screenshot shows the AWS EC2 Dashboard. The left sidebar has sections for Dashboard, Instances, Images, and Elastic Block Store. The main area shows 'Resources' with metrics: Instances (running) 0, Auto Scaling Groups 0, Capacity Reservations 0, Dedicated Hosts 0, Elastic IPs 2, Instances 0, Key pairs 0, Load balancers 1, Placement groups 0, Security groups 6, Snapshots 0, and Volumes 0. Below this are sections for 'Launch instance' (with 'Launch instance' and 'Migrate a server' buttons) and 'Service health' (Region: United States (Oregon), Status: This service is operating normally). To the right is the 'Account attributes' panel with settings for Default VPC, Settings (Data protection and security, Allowed AMIs, Zones, EC2 Serial Console, Default credit specification, EC2 console preferences), and an 'Explore AWS' section.

- In the left navigation pane, under the **Auto Scaling** section, choose **Auto Scaling Groups**.

The screenshot shows the 'Auto Scaling groups' page. The left sidebar has sections for Volumes, Snapshots, Lifecycle Manager, Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces), Load Balancing (Load Balancers, Target Groups, Trust Stores), and Auto Scaling (Auto Scaling Groups). The main area has a central banner: 'Amazon EC2 Auto Scaling helps maintain the availability of your applications'. Below it is a 'Create Auto Scaling group' button. To the right are sections for 'How it works' (diagram showing a queue and processing steps) and 'Pricing' (text stating no additional fees beyond service fees for EC2, CloudWatch, and other resources).

- Choose **Create Auto Scaling group**.

The **Choose launch template or configuration** page is displayed.

This screenshot shows the 'Choose launch template or configuration' step of the 'Create Auto Scaling group' wizard. On the left, a sidebar lists steps from 1 to 7. Step 1 is selected, showing 'Choose launch template or configuration'. The main area contains two sections: 'Name' and 'Launch template'. In the 'Name' section, there is a field for 'Auto Scaling group name' with placeholder text 'Enter a name to identify the group.' Below it is a note: 'Must be unique to this account in the current Region and no more than 255 characters.' In the 'Launch template' section, there is a dropdown menu labeled 'Select a launch template' and a link 'Create a launch template'. At the bottom right are 'Cancel' and 'Next' buttons.

- Configure the following:
 - Auto Scaling group name:** Enter **WP-ASG**.
 - Launch template:** Select the launch template that you created in Task 5. (**LabLaunchTemplate**)

This screenshot shows the 'Launch template' configuration page. It includes sections for 'Launch template', 'Version', 'Description', 'AMI ID', 'Key pair name', 'Additional details', and 'Storage (volumes)'. The 'Launch template' section shows 'LabLaunchTemplate' selected. The 'Version' section shows 'Default (1)'. The 'Description' section shows 'LabLaunchTemplate'. The 'AMI ID' section shows 'ami-05572e392e80aeee89'. The 'Key pair name' section shows '-' (empty). The 'Additional details' section shows 'Date created: Fri Apr 18 2025 00:35:11 GMT+0530 (India Standard Time)'. At the bottom right are 'Cancel' and 'Next' buttons.

- Choose **Next**.

The **Choose instance launch options** page is displayed.

Step 1

- Choose launch template or configuration

Step 2

Choose instance launch options

Step 3 - optional

- Integrate with other services

Step 4 - optional

- Configure group size and scaling

Step 5 - optional

- Add notifications

Step 6 - optional

- Add tags

Step 7

Choose instance launch options Info

Choose the VPC network environment that your instances are launched into, and customize the instance types and purchase options.

Instance type requirements Info

You can keep the same instance attributes or instance type from your launch template, or you can choose to override the launch template by specifying different instance attributes or manually adding instance types.

Launch template	Version	Description
LabLaunchTemplate [edit] lt-0698813684aae3213	Default	-

Instance type
t3.medium

Network Info

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC
Choose the VPC that defines the virtual network for your Auto Scaling group.

vpc-00b2a8e49e31269c3
172.31.0.0/16 Default

[Create a VPC](#) [\[edit\]](#)

- In the **Network** section, configure the following:
 - VPC:** Select **LabVPC** from the dropdown menu.
 - Availability Zones and subnets:** Select **AppSubnet1** and **AppSubnet2** from the dropdown menu.

Network Info

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC
Choose the VPC that defines the virtual network for your Auto Scaling group.

vpc-061c4c3374f72bc96 (LabVPC)
10.0.0.0/16

[Create a VPC](#) [\[edit\]](#)

Availability Zones and subnets
Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.

Select Availability Zones and subnets

us-west-2a | subnet-0663bdf2f7c9c2d8b (AppSubnet1) [X](#)
10.0.2.0/24

us-west-2b | subnet-0e53afb5e9bcd2a01 (AppSubnet2) [X](#)
10.0.3.0/24

[Create a subnet](#) [\[edit\]](#)

Availability Zone distribution - new
Auto Scaling automatically balances instances across Availability Zones. If launch failures occur in a zone, select a strategy.

Balanced best effort
If launches fail in one Availability Zone, Auto Scaling will attempt to launch in another healthy Availability Zone.

Balanced only
If launches fail in one Availability Zone, Auto Scaling will continue to attempt to launch in the unhealthy Availability Zone to preserve balanced distribution.

[Cancel](#) [Skip to review](#) [Previous](#) [Next](#)

- Choose **Next**.

The **Integrate with other services - optional** page is displayed.

Step 1
 Choose launch template or configuration

Step 2
 Choose instance launch options

Step 3 - optional
 Integrate with other services

Step 4 - optional
 Configure group size and scaling

Step 5 - optional
 Add notifications

Step 6 - optional
 Add tags

Step 7
 Review

Integrate with other services - optional Info

Use a load balancer to distribute network traffic across multiple servers. Enable service-to-service communications with VPC Lattice. Shift resources away from impaired Availability Zones with zonal shift. You can also customize health check replacements and monitoring.

Load balancing Info

Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

No load balancer
Traffic to your Auto Scaling group will not be fronted by a load balancer.

Attach to an existing load balancer
Choose from your existing load balancers.

Attach to a new load balancer
Quickly create a basic load balancer to attach to your Auto Scaling group.

VPC Lattice integration options Info

To improve networking capabilities and scalability, integrate your Auto Scaling group with VPC Lattice. VPC Lattice facilitates communications between AWS services and helps you connect and manage your applications across compute services in AWS.

Select VPC Lattice service to attach

No VPC Lattice service
VPC Lattice will not manage your Auto Scaling group's network access and connectivity with other services.

Attach to VPC Lattice service
Incoming requests associated with specified VPC Lattice target groups will be routed to your Auto Scaling group.

Create new VPC Lattice service 

Application Recovery Controller (ARC) zonal shift - new Info

During an Availability Zone impairment, target instance launches towards other healthy Availability Zones.

Enable zonal shift
New instance launches will be retargeted towards healthy Availability Zones until the zonal shift is canceled.

Health checks

Health checks increase availability by replacing unhealthy instances. When you use multiple health checks, all are evaluated, and if at least one fails, instance replacement occurs.

EC2 health checks

Always enabled

Additional health check types - optional Info

Turn on Elastic Load Balancing health checks
Elastic Load Balancing monitors whether instances are available to handle requests. When it reports an unhealthy instance, EC2 Auto Scaling can replace it on its next periodic check.

Turn on VPC Lattice health checks
VPC Lattice can monitor whether instances are available to handle requests. If it considers a target as failed a health check, EC2 Auto Scaling replaces it after its next periodic check.

Turn on Amazon EBS health checks
EBS monitors whether an instance's root volume or attached volume is still. When it reports an unhealthy volume, EC2 Auto Scaling can replace the instance on its next periodic health check.

Health check grace period Info

This time period delays the first health check until your instances finish initializing. It doesn't prevent an instance from terminating when placed into a non-running state.

300 seconds

Cancel Skip to review Previous Next

- On the **Integrate with other services - optional** page, configure the following:
 - Select **Attach to an existing load balancer**.
 - Select **Choose from your load balancer target groups**.
 - From the **Existing load balancer target groups** dropdown menu, select **myWPTargetGroup | HTTP**.

Integrate with other services - optional Info

Use a load balancer to distribute network traffic across multiple servers. Enable service-to-service communications with VPC Lattice. Shift resources away from impaired Availability Zones with zonal shift. You can also customize health check replacements and monitoring.

Load balancing Info

Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

No load balancer
Traffic to your Auto Scaling group will not be fronted by a load balancer.

Attach to an existing load balancer
Choose from your existing load balancers.

Attach to a new load balancer
Quickly create a basic load balancer to attach to your Auto Scaling group.

Attach to an existing load balancer

Select the load balancers that you want to attach to your Auto Scaling group.

Choose from your load balancer target groups

This option allows you to attach Application, Network, or Gateway Load Balancers.

Choose from Classic Load Balancers

Existing load balancer target groups

Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.

Select target groups 



myWPTargetGroup | HTTP

Application Load Balancer: myWPAppALB 

- For **Additional health check types - optional**: Select **Turn on Elastic Load Balancing health checks**.
- **Health check grace period**: Leave at the default value of 300 or more.

Health checks

Health checks increase availability by replacing unhealthy instances. When you use multiple health checks, all are evaluated, and if at least one fails, instance replacement occurs.

EC2 health checks

Always enabled

Additional health check types - optional | [Info](#)

Turn on Elastic Load Balancing health checks Recommended

Elastic Load Balancing monitors whether instances are available to handle requests. When it reports an unhealthy instance, EC2 Auto Scaling can replace it on its next periodic check.

EC2 Auto Scaling will start to detect and act on health checks performed by Elastic Load Balancing. To avoid unexpected terminations, first verify the settings of these health checks in the [Load Balancer console](#) X

Turn on VPC Lattice health checks
VPC Lattice can monitor whether instances are available to handle requests. If it considers a target as failed a health check, EC2 Auto Scaling replaces it after its next periodic check.

Turn on Amazon EBS health checks
EBS monitors whether an instance's root volume or attached volume stalls. When it reports an unhealthy volume, EC2 Auto Scaling can replace the instance on its next periodic health check.

Health check grace period | [Info](#)

This time period delays the first health check until your instances finish initializing. It doesn't prevent an instance from terminating when placed into a non-running state.

seconds

[Cancel](#) [Skip to review](#) [Previous](#) **Next**

- Choose **Next**.

The **Configure group size and scaling - optional** page is displayed.

aws [Alt+S] Search

AWS Cloud Map EFS EC2

United States (Oregon) AWSLambdaUser-8JacCtzJcCMTMVBrEtkyJ/bf420498-e8ed-48d3-8727-cd... ▾

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1 Choose launch template or configuration

Step 2 Choose instance launch options

Step 3 - optional Integrate with other services

Step 4 - optional **Configure group size and scaling**

Step 5 - optional Add notifications

Step 6 - optional Add tags

Step 7 Review

Configure group size and scaling - optional [Info](#)

Define your group's desired capacity and scaling limits. You can optionally add automatic scaling to adjust the size of your group.

Group size [Info](#)

Set the initial size of the Auto Scaling group. After creating the group, you can change its size to meet demand, either manually or by using automatic scaling.

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

Units (number of instances)

Desired capacity
Specify your group size.

Scaling [Info](#)

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

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

Min desired capacity	Max desired capacity
<input type="text" value="1"/>	<input type="text" value="1"/>
Equal or less than desired capacity Equal or greater than desired capacity	

Automatic scaling - optional
[Info](#)

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

- On the **Configure group size and scaling - optional** page, configure the following:
 - o In the **Group size** section:
 - **Desired capacity**: Enter **2**.
 - o In the **Scaling** section:
 - **Min desired capacity**: Enter **2**
 - **Max desired capacity**: Enter **4**
- In the **Automatic scaling - optional** section, configure the following:
 - o Select **Target tracking scaling policy**.

Group size Set the initial size of the Auto Scaling group. After creating the group, you can change its size to meet demand, either manually or by using automatic scaling.

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

Desired capacity Specify your group size.

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

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

Min desired capacity 2 Equal or less than desired capacity

Max desired capacity 4 Equal or greater than desired capacity

Automatic scaling - optional

Choose whether to use a target tracking policy You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

No scaling policies Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

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

The remaining settings on this section can be left at their default values.

Scaling policy name Target Tracking Policy

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

Average CPU Utilization

Target value 50

Instance warmup 300 seconds

Disable scale in to create only a scale-out policy

Instance maintenance policy Control your Auto Scaling group's availability during instance replacement events. This includes health checks, instance refreshes, maximum instance lifetime features and events that happen automatically to keep your group balanced, called rebalancing events.

Choose a replacement behavior depending on your availability requirements

- Mixed behavior** No policy For replacing events, new instances will launch before terminating others. For all other events, instances terminate and launch at the same time.
- Prioritize availability** Launch before terminating Launch new instances and wait for them to be ready before terminating others. This allows you to go above your desired capacity by a given percentage and may temporarily increase costs.
- Control costs** Terminate and launch Terminates and launches instances at the same time. This allows you to go below your desired capacity by a given percentage and may temporarily reduce availability.
- Flexible** Custom behavior Set custom values for the minimum and maximum amount of available capacity. This gives you greater flexibility in setting how far below and over your desired capacity EC2 Auto Scaling goes when replacing instances.

- **Monitoring:** Select **Enable group metrics collection within CloudWatch**.

Additional capacity settings

Capacity Reservation preference Select whether you want Auto Scaling to launch instances into an existing Capacity Reservation or Capacity Reservation resource group.

Default Auto Scaling uses the Capacity Reservation preference from your launch template.

None Instances will not be launched into a Capacity Reservation.

Capacity Reservations only Instances will only be launched into a Capacity Reservation. If capacity isn't available, the instances fail to launch.

Capacity Reservations first Instances will attempt to launch into a Capacity Reservation first. If capacity isn't available, instances will run in On-Demand capacity.

Additional settings

Instance scale-in protection If protect from scale in is enabled, newly launched instances will be protected from scale in by default.

Enable instance scale-in protection

Monitoring Enable group metrics collection within CloudWatch

Default instance warmup The amount of time that CloudWatch metrics for new instances do not contribute to the group's aggregated instance metrics, as their usage data is not reliable yet.

Enable default instance warmup

Cancel **Skip to review** **Previous** **Next**

- Choose **Next**.

The **Add notifications - optional** page is displayed.

- Choose **Next**.

The **Add tags - optional** page is displayed.

- Choose **Add tag** and then configure the following:
 - o **Key:** Enter **Name**.
 - o **Value - optional:** Enter **WP-App**.

- Choose **Next**.

The **Review** page is displayed.

AWS Cloud Map EFS EC2

EC2 > Auto Scaling groups > Create Auto Scaling group

- Step 1
 - Choose launch template or configuration
 - Choose instance launch options
 - Integrate with other services
 - Configure group size and scaling
 - Add notifications
 - Add tags
- Step 2
- Step 3 - optional
- Step 4 - optional
- Step 5 - optional
- Step 6 - optional
- Step 7

Review Info

Step 1: Choose launch template or configuration

Group details

Auto Scaling group name
WP-ASG

Launch template

Launch template
[LabLaunchTemplate](#)
lt-0698813684aae5213

Version
Default

Description

[Edit](#)

Step 2: Choose instance launch options

Network

VPC
[vpc-061c4c3374f72bc96](#)

Availability Zones and subnets

Availability Zone	Subnet	Subnet CIDR range
us-west-2a	subnet-0663bdf2f7c9c2d8b	10.0.2.0/24
us-west-2b	subnet-0e53afb5e9bcd2a01	10.0.3.0/24

[Edit](#)

AWS Cloud Map EFS EC2

EC2 > Auto Scaling groups > Create Auto Scaling group

Availability Zone distribution

Balanced best effort

Instance type requirements

This Auto Scaling group will adhere to the launch template.

Step 3: Integrate with other services

Load balancing

Load balancer 1

Name
[myWPApplALB](#)

Type
Application/HTTP

Target group
[myWPTargetGroup](#)

[Edit](#)

VPC Lattice integration options

VPC Lattice target groups

Application Recovery Controller (ARC) zonal shift

ARC zonal shift
Disabled

AWS Cloud Map EFS EC2

EC2 > Auto Scaling groups > Create Auto Scaling group

Health checks

Health check type
EC2, ELB

Health check grace period
300 seconds

Step 4: Configure group size and scaling policies

Group size

Desired capacity
2

Desired capacity type
Units (number of instances)

Scaling

Minimum desired capacity
2

Maximum desired capacity
4

Target tracking policy
Policy type
Target tracking scaling

Scaling policy name
Target Tracking Policy

Execute policy when
As required to maintain Average CPU utilization at 50

Take the action
Add or remove capacity units as required

Instances need
300 seconds to warm up before including in metric

Scale in
Enabled

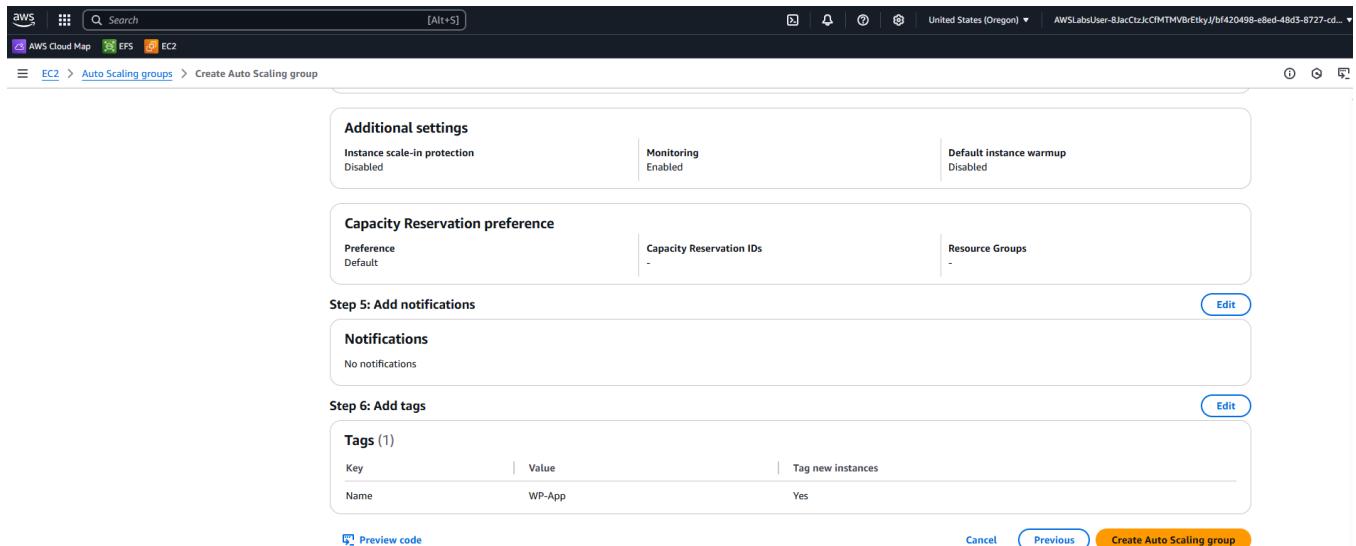
Instance maintenance policy

Replacement behavior
No policy

Min healthy percentage
-

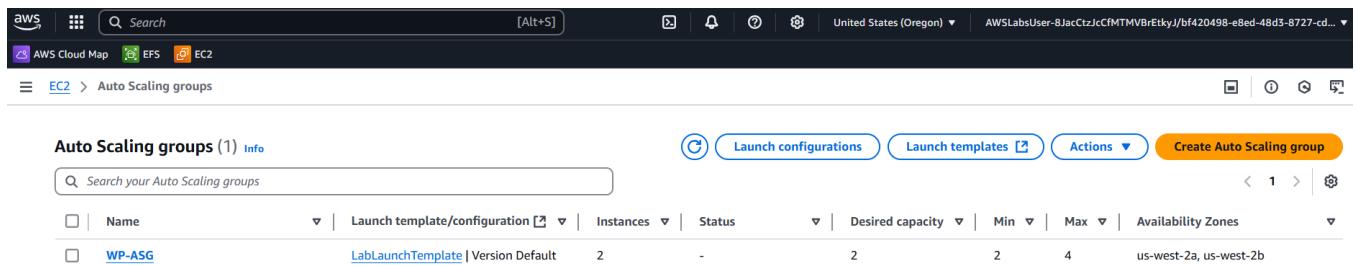
Max healthy percentage
-

1



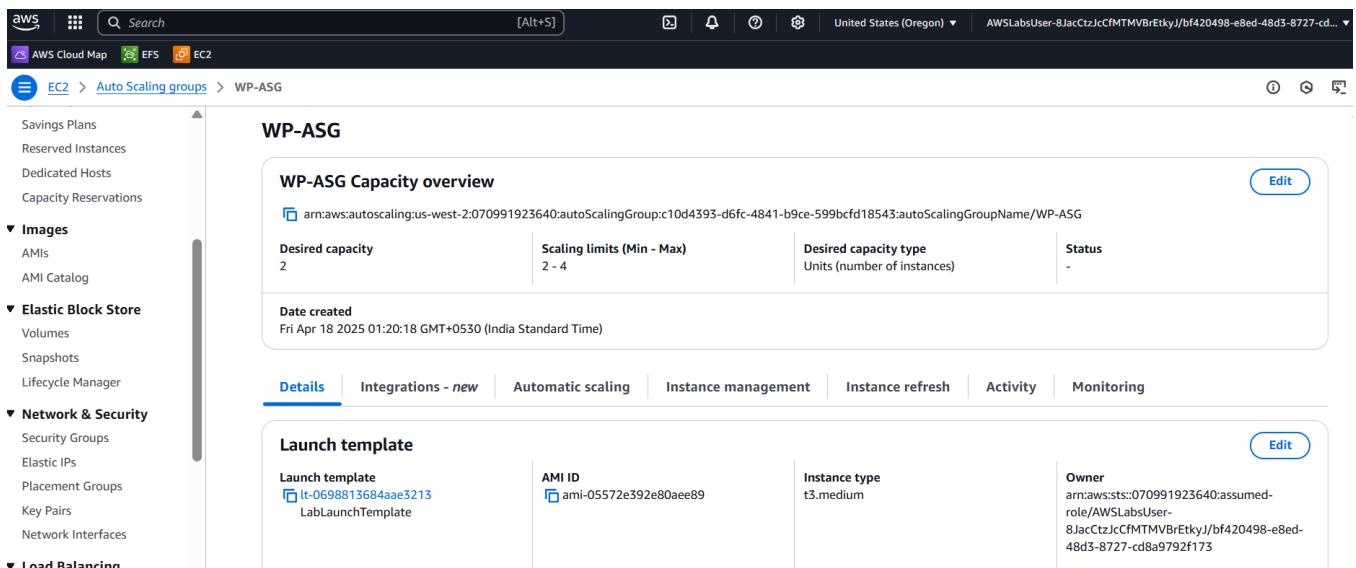
- Review the Auto Scaling group configuration for accuracy, and then at the bottom of the page, choose **Create Auto Scaling group**.

The **Auto Scaling groups** page is displayed.



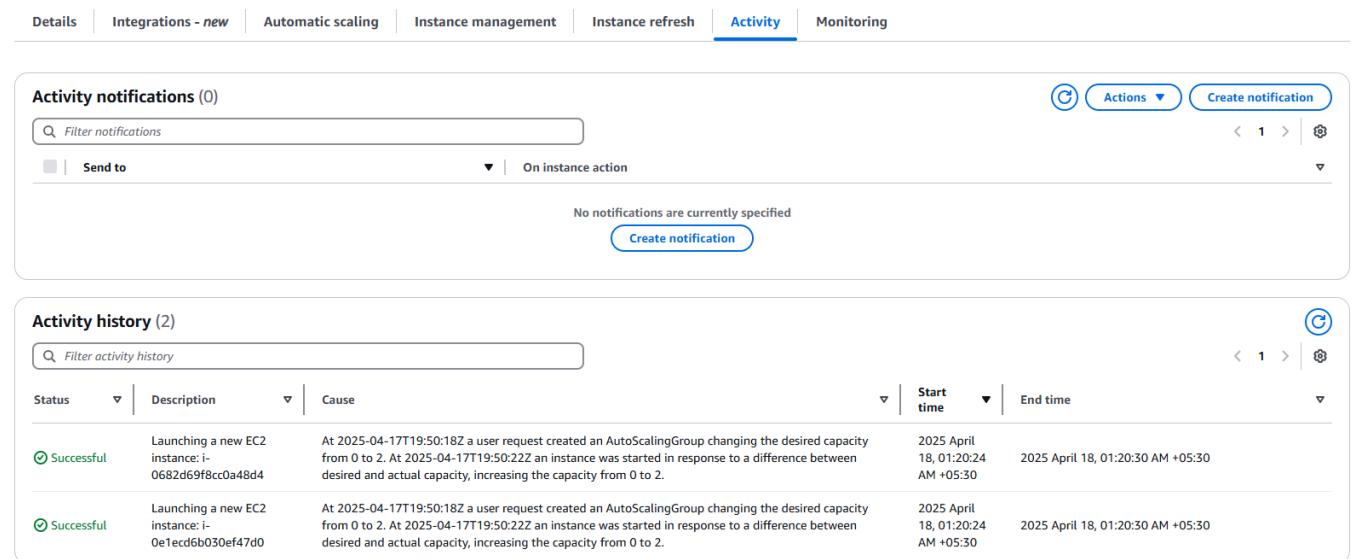
Now that you have created your Auto Scaling group, you can verify that the group has launched your EC2 instances.

- Choose the Auto Scaling group **WP-ASG** link.



- Choose the **Activity** tab.

The **Activity history** section maintains a record of events that have occurred in your Auto Scaling group. The Status column contains the current status of your instances. When your instances are launching, the status column shows *PreInService*. The status changes to *Successful* after an instance is launched.

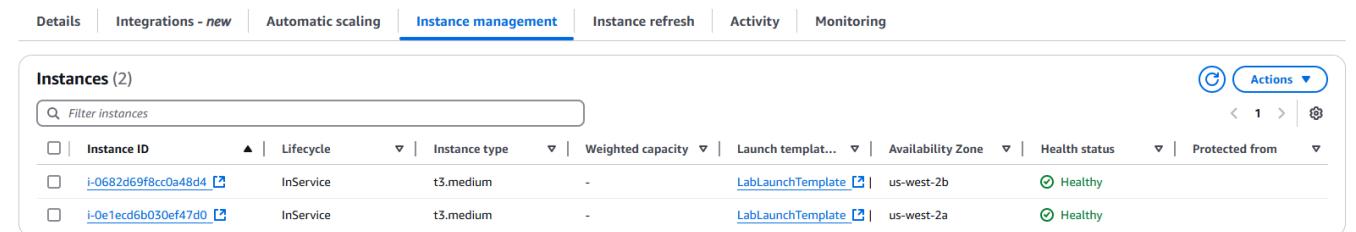


The screenshot shows the 'Activity' tab selected in the navigation bar. The 'Activity notifications' section is empty. The 'Activity history' section shows two entries:

Status	Description	Cause	Start time	End time
Successful	Launching a new EC2 instance: i-0682d69f8cc0a48d4	At 2025-04-17T19:50:18Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 2. At 2025-04-17T19:50:22Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 2.	2025 April 18, 01:20:24 AM +05:30	2025 April 18, 01:20:30 AM +05:30
Successful	Launching a new EC2 instance: i-0e1ecdb030ef47d0	At 2025-04-17T19:50:18Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 2. At 2025-04-17T19:50:22Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 2.	2025 April 18, 01:20:24 AM +05:30	2025 April 18, 01:20:30 AM +05:30

- Choose the **Instance management** tab.

Your Auto Scaling group has launched two Amazon EC2 instances and they are in the *InService* lifecycle state. The Health Status column shows the result of the Amazon EC2 instance health check on your instances.



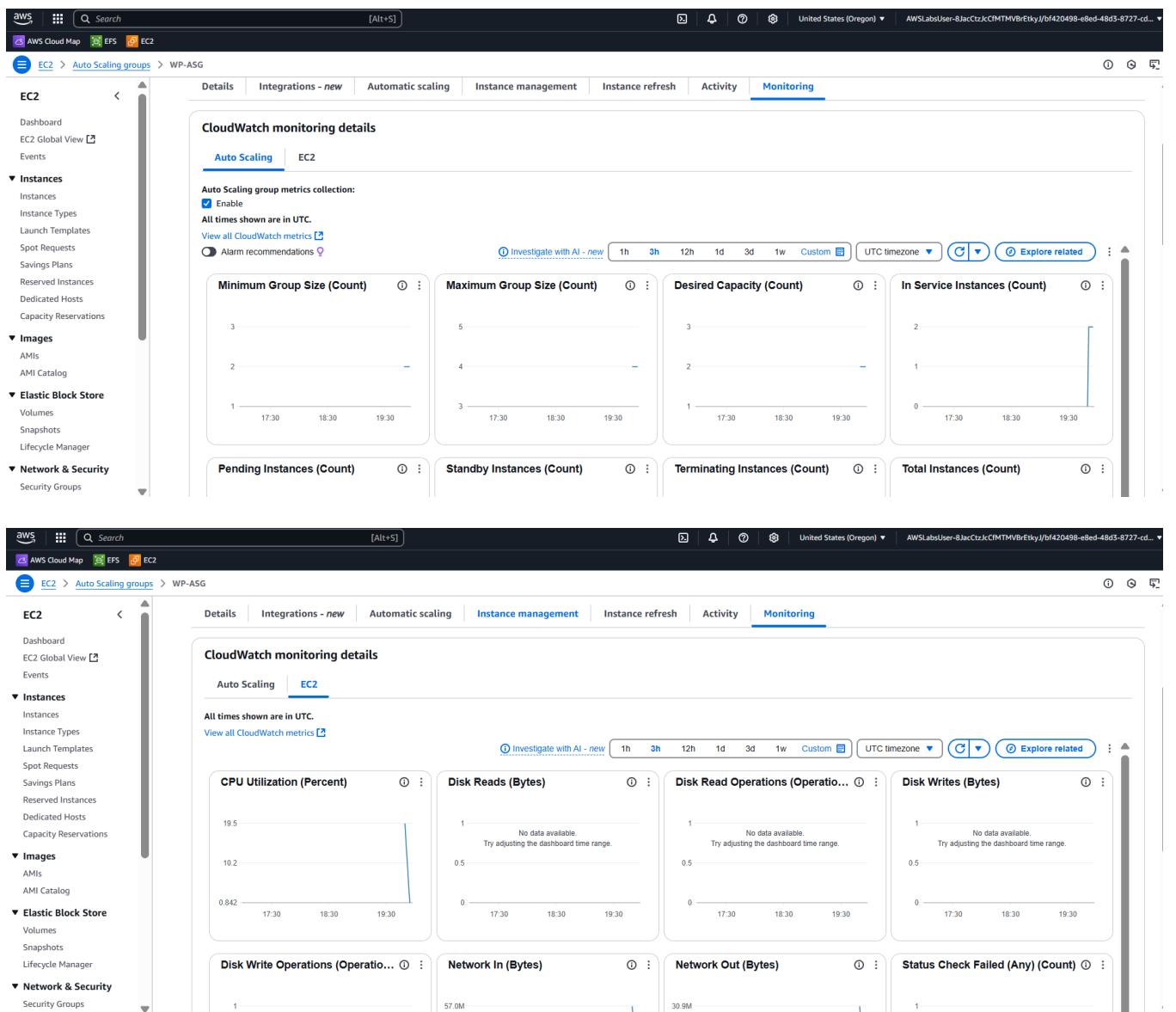
The screenshot shows the 'Instance management' tab selected in the navigation bar. The 'Instances' section shows two instances:

Instance ID	Lifecycle	Instance type	Weighted capacity	Launch templ...	Availability Zone	Health status	Protected from
i-0682d69f8cc0a48d4	InService	t3.medium	-	LabLaunchTemplate	us-west-2b	Healthy	
i-0e1ecdb030ef47d0	InService	t3.medium	-	LabLaunchTemplate	us-west-2a	Healthy	

If your instances have not reached the *InService* state yet, you need to wait a few minutes. You can choose the refresh button to retrieve the current lifecycle state of your instances.

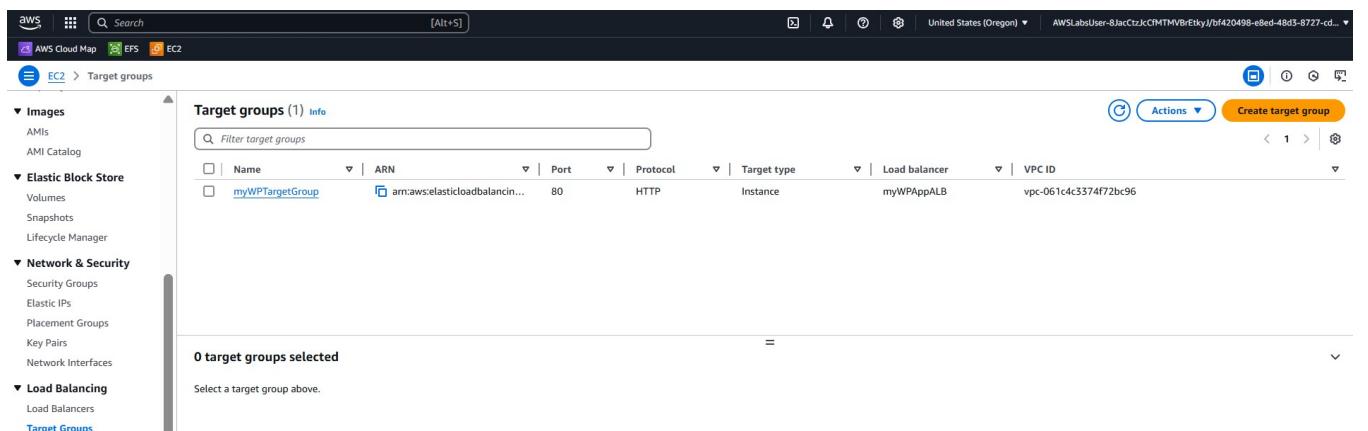
- Choose the **Monitoring** tab. Here, you can review monitoring-related information for your Auto Scaling group.

This page provides information about activity in your Auto Scaling group, as well as the usage and health status of your instances. The **Auto Scaling** tab displays Amazon CloudWatch metrics about your Auto Scaling group, while the **EC2** tab displays metrics for the Amazon EC2 instances managed by the Auto Scaling group.



Task 6.2: Verify the target groups are healthy

- In the left navigation pane on the left, choose **Target Groups**.



- Choose the **myWPTargetGroup** link.

myWPTargetGroup

Targets

Instance ID	Name	Port	Zone	Health status	Health status details	Administrative ...	Override details	Launch...	Anomaly d...
i-0e1ec0f6b030ef47d0	WP-App	80	us-west-2a (us...)	Healthy	-	No override	No override is cur...	April 18, 2...	Normal
i-0682d69fbcc0a4bd4	WP-App	80	us-west-2b (us...)	Healthy	-	No override	No override is cur...	April 18, 2...	Normal

- In the **Targets** tab, wait until the instance Health status is displayed as *healthy*.

Note: It can take up to 5 minutes for the health checks to show as healthy. Wait for the **Health status** to display *healthy* before continuing.

Task 6.3: Log in to the WordPress web application

- In the left navigation pane, choose **Load Balancers**.

Load balancers (1)

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
myWPApplB	myWPApplB-98622990.us-west-2.elb.amazonaws.com	Active	vpc-01c715619a04cb7bd	2 Availability Zones	application	April 18, 2025, 11:57 (UTC+05:30)

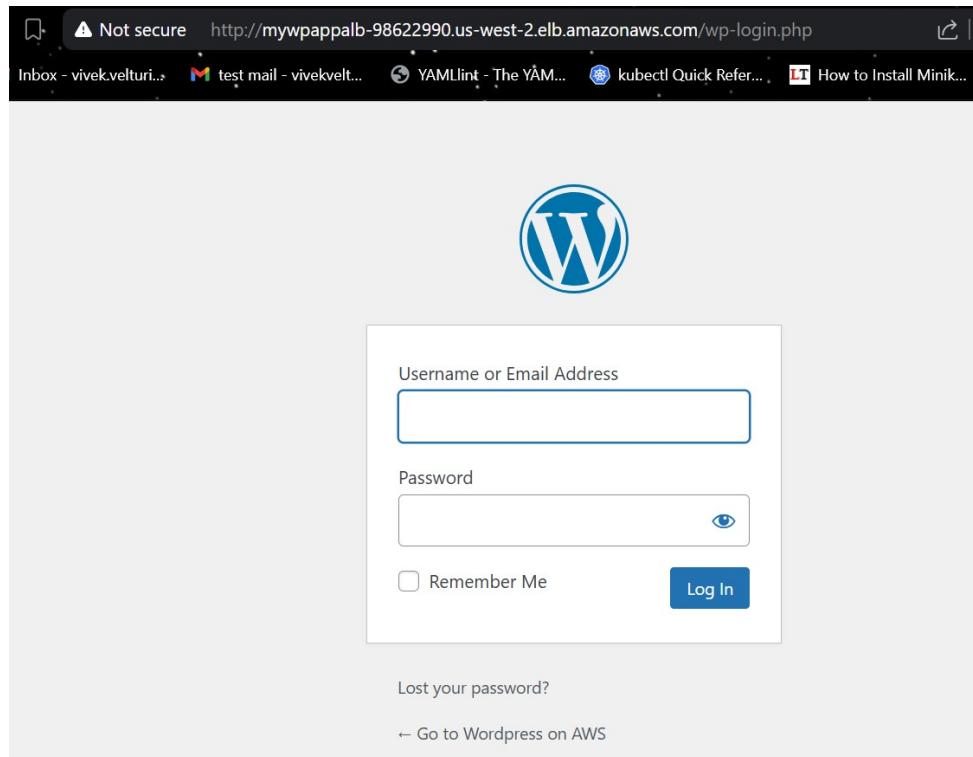
- Copy the **myWPApplB** load balancer **DNS name** (`myWPApplB-98622990.us-west-2.elb.amazonaws.com`) to a text editor and append the value `/wp-login.php` to the end of the DNS name to complete your WordPress application URL.

Expected output: Example of a completed WordPress application URL:

`myWPApplB-4e009e86b4f704cc.elb.us-west-2.amazonaws.com/wp-login.php`

- Paste the WordPress application URL value into a new browser tab.

The **WordPress login** page is displayed.



- Enter the following:
 - o Username or Email Address: Enter **wpadmin** .
 - o Password: Paste the **LabPassword** value(1QaDM4T5Cxny) .
- Choose the **Log in** button.

Welcome to WordPress!

Learn more about the 6.8 version.

Author rich content with blocks and patterns

Customize your entire site with block themes

Switch up your site's look & feel with Styles

Image description: Preceding image depicts the final state when user navigates to the /wp-login.php page and inputs the login credentials.

You have created the AWS Auto Scaling group and successfully launched the WordPress application.

Conclusion

You now have successfully:

- Deployed a virtual network spread across multiple Availability Zones in a Region using a CloudFormation template.
- Deployed a highly available and fully managed relational database across those Availability Zones using Amazon RDS.
- Used Amazon EFS to provision a shared storage layer across multiple Availability Zones for the application tier, powered by NFS.
- Created a group of web servers that automatically scales in response to load variations to complete your application tier.