

Guided Lab: Creating a Highly Available Environment

Lab overview and objectives

Critical business systems should be deployed as highly available applications; that is, applications remain operational even when some components fail. To achieve high availability in Amazon Web Services (AWS), you run services across multiple Availability Zones.

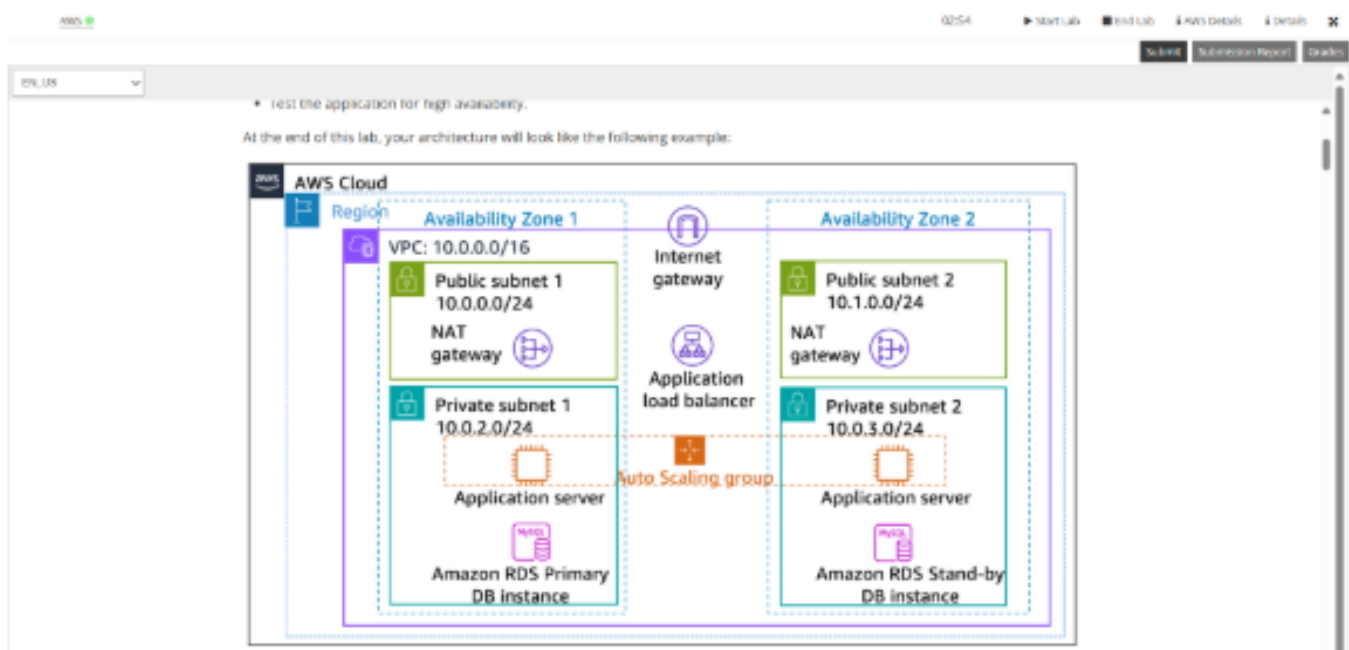
Many AWS services are inherently highly available, such as load balancers. Many AWS services can also be configured for high availability, such as deploying Amazon Elastic Compute Cloud (Amazon EC2) instances in multiple Availability Zones.

In this lab, you start with an application that runs on a single EC2 instance and then make it highly available.

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

- Inspect a provided virtual private cloud (VPC).
- Create an Application Load Balancer.
- Create an Auto Scaling group.
- Test the application for high availability.

At the end of this lab, your architecture will look like the following example:



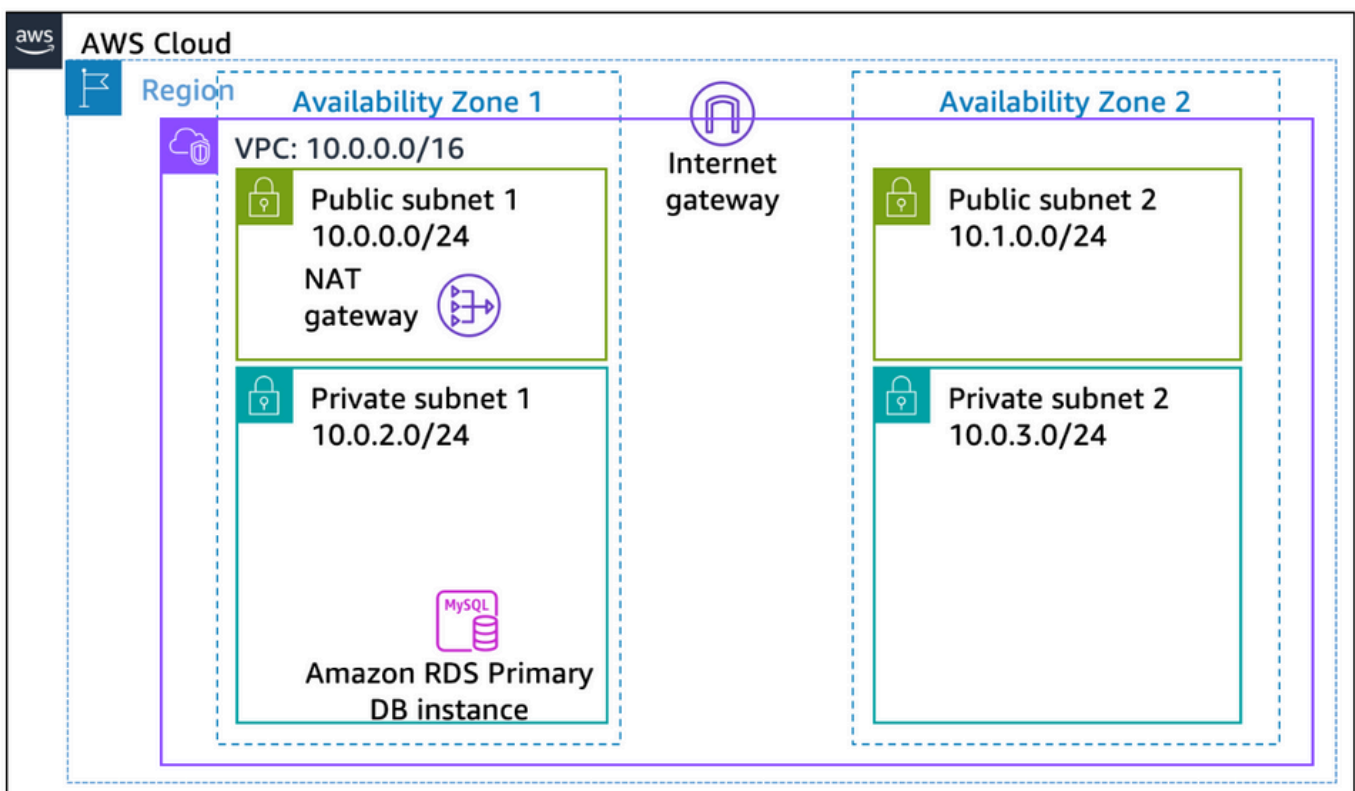
Duration

The lab requires approximately **40 minutes** to complete.

Task 1: Inspecting your VPC

This lab begins with an environment that is already deployed through AWS CloudFormation. This environment includes the following:

- A VPC
- Public and private subnets in two Availability Zones
- An internet gateway that is associated with the public subnets
- A NAT gateway in one of the public subnets
- An Amazon Relational Database Service (Amazon RDS) instance in one of the private subnets



In this task, you review the configuration of the VPC that was created for this lab.

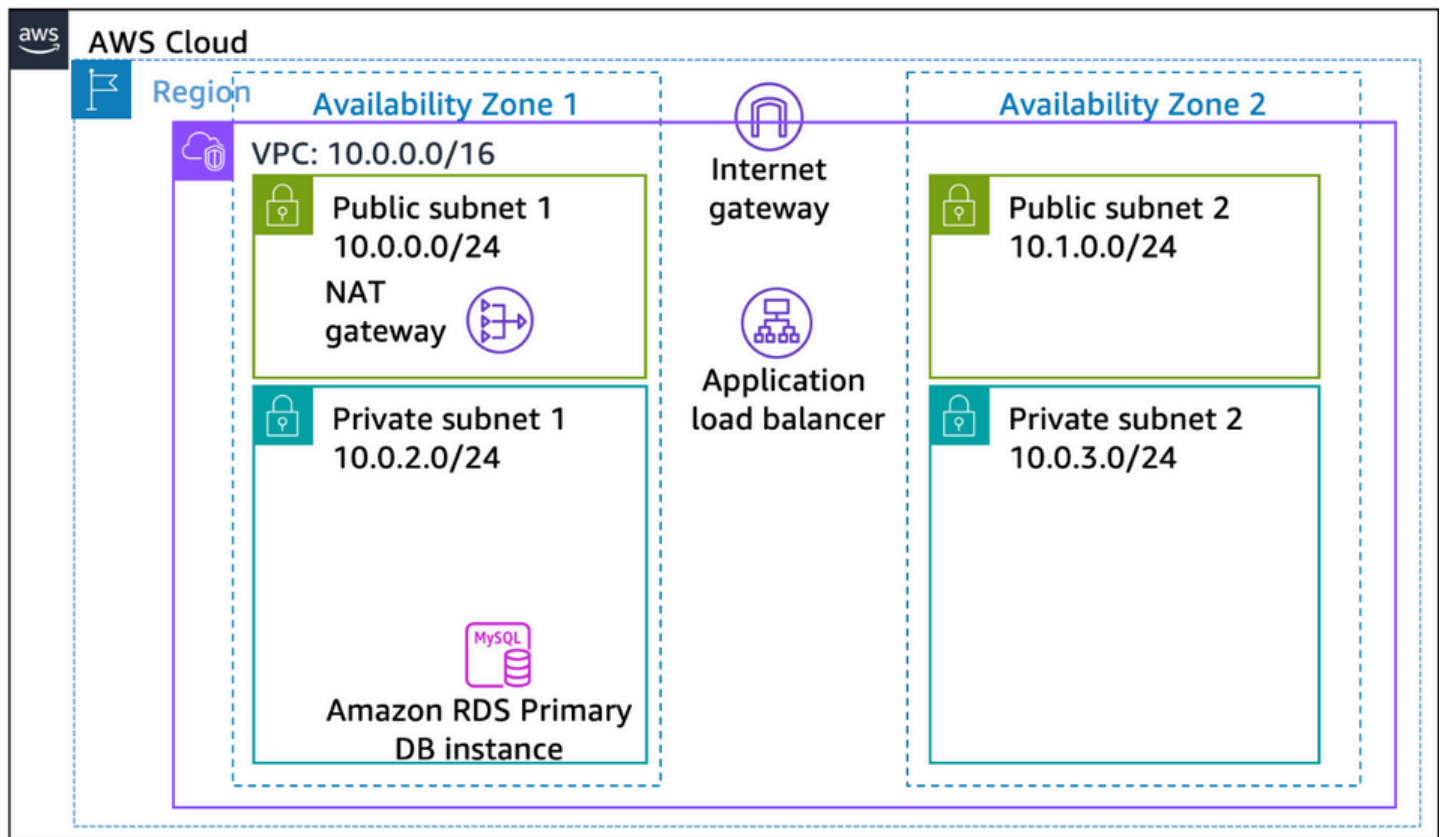
1. On the AWS Management Console, in the search box, enter and choose VPC to open the Amazon VPC console.
2. In the left navigation pane, from the **Filter by VPC** dropdown list, choose **Lab VPC**.
3. This setting limits the console to show only resources that are associated with the Lab VPC.
4. In the left navigation pane, choose **Your VPCs**.

5. Here, you can access information about the **Lab VPC** that was created for you.
6. Select **Lab VPC**.
7. In the **Details** tab, notice that the **IPv4 CIDR** field has a value of **10.0.0.0/16**, which means that this VPC includes all IP addresses that start with 10.0.x.x.
8. In the left navigation pane, choose **Subnets**.
9. Here, you can access information about **Public Subnet 1**. In the list of subnets, notice the following:
 - The **VPC** column for **Public Subnet 1** shows that this subnet exists inside the **Lab VPC**.
 - The **IPv4 CIDR** column has a value of **10.0.0.0/24**, which means that this subnet includes the 256 IP addresses between 10.0.0.0 and 10.0.0.255. Five of these addresses are reserved and unusable.
 - The **Availability Zone** column lists the Availability Zone where this subnet resides.
10. To reveal more details, select **Public Subnet 1**.
11. **Tip:** To adjust the size of the lower window pane, you can drag the divider.
12. In the lower half of the page, choose the **Route table** tab.
13. This tab includes details about the routing for this subnet:
 - The first entry specifies that traffic destined within the Classless Inter-Domain Routing (CIDR) range for the VPC (**10.0.0.0/16**) is routed within the VPC (**local**).
 - The second entry specifies that any traffic destined for the internet (**0.0.0.0/0**) is routed to the internet gateway (**igw-**) that exists in Lab VPC. This setting makes the subnet a *public subnet*.
14. Choose the **Network ACL** tab.
15. This tab has information about the network access control list (network ACL) that is associated with the subnet. The rules are currently set to the default settings. They permit all traffic to flow in and out of the subnet, but you can further restrict the rules by using security groups.
16. In the left navigation pane, choose **Internet gateways**.
17. Notice that an internet gateway with the name **Lab IG** is already attached to the **Lab VPC**.
18. In the left navigation pane, choose **Security groups**.
19. Select **Inventory-DB**.
20. This security group controls incoming traffic to the database.
21. In the lower half of the page, choose the **Inbound rules** tab.
22. The rule defined in this security group permits inbound MySQL or Aurora traffic (port 3306) from anywhere in the VPC (**10.0.0.0/16**). You later modify this setting so that it accepts traffic from only the application servers.
23. Choose the **Outbound rules** tab.
24. By default, security groups allow all outbound traffic. However, you can modify these settings as needed.

Task 2: Creating an Application Load Balancer

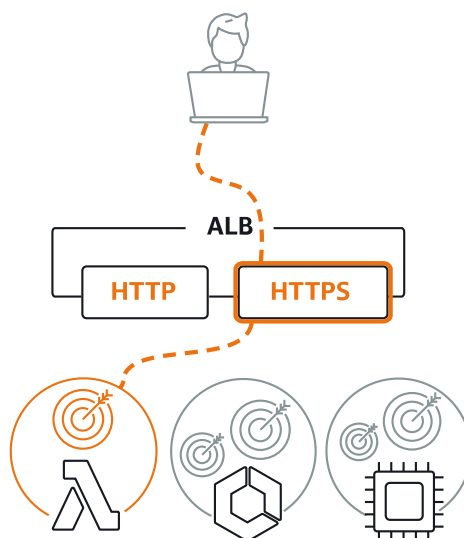
To build a highly available application, it is a best practice to launch resources in multiple Availability Zones. Availability Zones are physically separate data centers (or groups of data centers) in the same Region. If you run your applications across multiple Availability Zones, you can provide greater availability if a data center experiences a failure.

Because the application runs on multiple application servers, you need a way to distribute traffic among those servers. You can accomplish this goal by using a *load balancer*. This load balancer also performs health checks on instances and sends requests to only healthy instances.



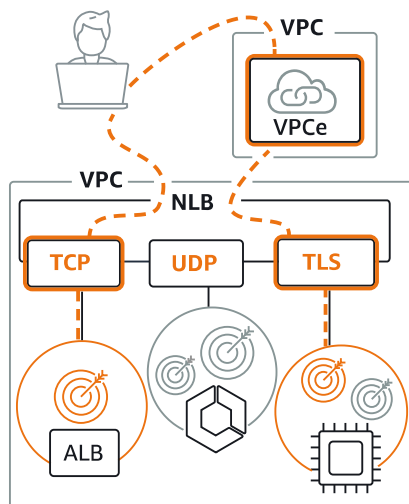
Load balancer types

- Application Load Balancer



- Choose an Application Load Balancer when you need a flexible feature set for your applications with HTTP and HTTPS traffic. Operating at the request level, Application Load Balancers provide advanced routing and visibility features targeted at application architectures, including microservices and containers.

- **Network Load Balancer**



- Choose a Network Load Balancer when you need ultra-high performance, TLS offloading at scale, centralized certificate deployment, support for UDP, and static IP addresses for your applications. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second securely while maintaining ultra-low latencies.

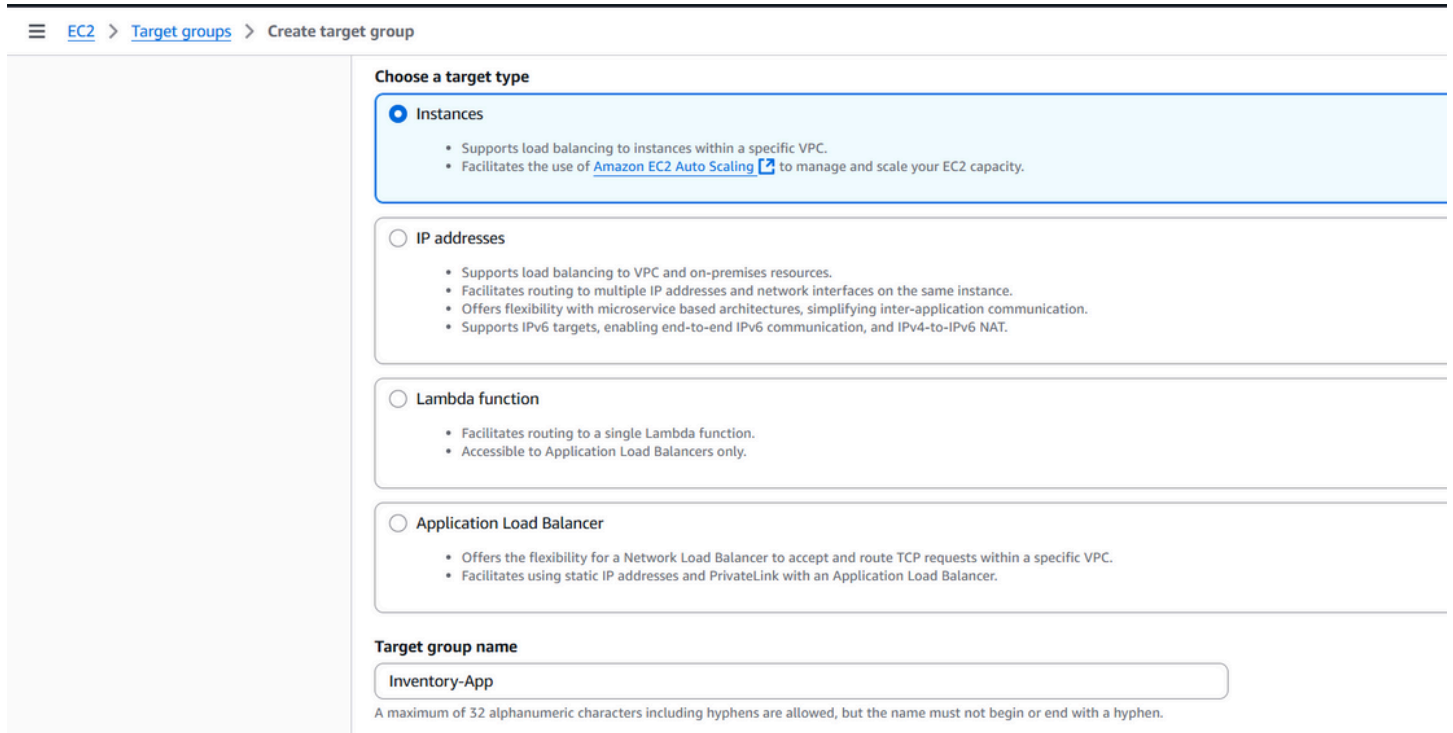
- **Gateway Load Balancer**



- Choose a Gateway Load Balancer when you need to deploy and manage a fleet of third-party virtual appliances that support GENEVE. These appliances enable you to improve security, compliance, and policy controls.

1. On the AWS Management Console, in the search box, enter and choose EC2 to open the Amazon EC2 console.
2. In the left navigation pane, choose **Load Balancers**. (You might need to scroll down to find it.)
3. Choose **Create load balancer**.
4. Several types of load balancers are displayed. Read the descriptions of each type to understand their capabilities.
5. For **Application Load Balancer**, choose **Create**.
6. In the **Basic configuration** section, for **Load balancer name**, enter Inventory-LB.
7. In the **Network mapping** section, configure the following options:
 - For **VPC**, select **Lab VPC**.
8. **Important:** Be sure to choose **Lab VPC**. It is likely not the default selection.
9. You now specify which subnets the load balancer should use. It will be a public load balancer, so you select both public subnets.
 - For **Mappings**, choose the first Availability Zone, and then choose the **Public Subnet** that displays.
 - for **Mappings**, also choose the second Availability Zone, and then choose the **Public Subnet** that displays.
10. You should now have selected two subnets: **Public Subnet 1** and **Public Subnet 2**. If not, go back and try the configuration again.
11. In the **Security groups** section, select the **create a new security group** hyperlink. This link opens a new browser tab.
12. On the **Create security group** page, in the **Basic details** section, configure the following options to create the new security group:
 - **Security group name:** Enter Inventory-LB.
 - **Description:** Enter Enable web access to load balancer.
 - **VPC:** From the dropdown list, select **Lab VPC**.
13. In the **Inbound rules** section, choose **Add rule**, and configure the following options:
 - **Type:** HTTP
 - **Source:** Anywhere-IPv4
14. For **Inbound rules**, choose **Add rule** again, and configure the following options:
 - **Type:** HTTPS
 - **Source:** Anywhere-IPv4
15. Choose **Create security group**.
16. Next, you assign the security group to the load balancer.
17. Return to the browser tab where you are still configuring the load balancer, and configure the following options:
 - In the **Security groups** section, choose the refresh icon.

- From the **Security groups** dropdown list, select the **Inventory-LB** security group that you just created.
 - Next, choose the **X** for the default security group so that **Inventory-LB** is now the only security group chosen.
18. In the **Listeners and routing** section, choose the **Create target group** link.
19. A new browser tab opens.
20. **Analysis:** *Target groups* define where to send traffic that comes into the load balancer. The Application Load Balancer can send traffic to multiple target groups based on the URL of the incoming request, such as having requests from mobile apps going to a different set of servers. Your web application uses only one target group.
21. For **Step 1: Specify group details**, configure the following options:



EC2 > Target groups > Create target group

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.
- ☐ **Application Load Balancer**
 - Offers the flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC.
 - Facilitates using static IP addresses and PrivateLink with an Application Load Balancer.

Target group name

Inventory-App

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

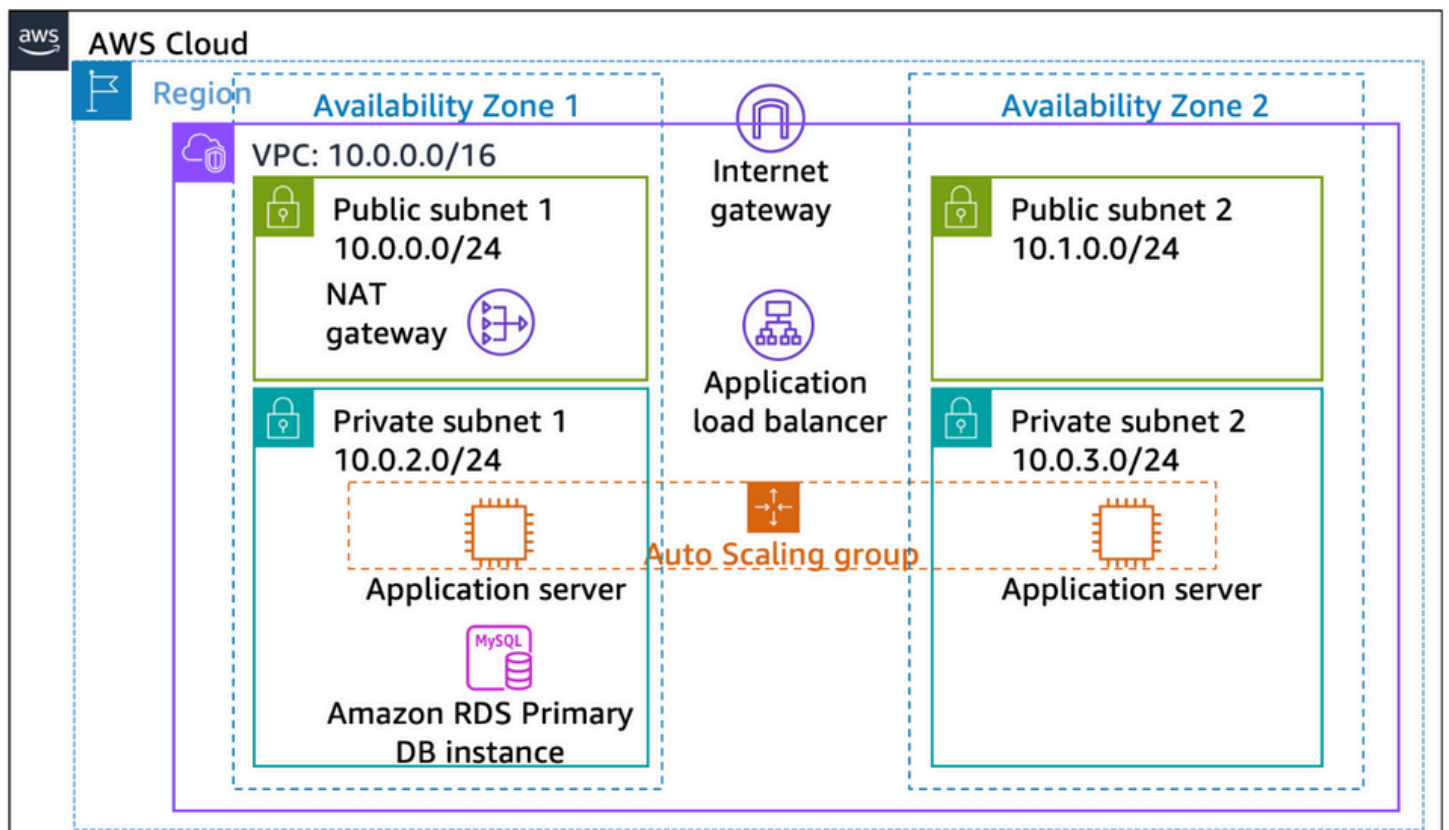
- In the **Basic configuration** section, configure the following options:
 - **Choose a target type:** Choose **Instances**.
 - **Target group name:** Enter Inventory-App.
 - **VPC:** Ensure that **Lab VPC** is chosen.
 - In the **Health checks** section, expand **Advanced health check settings**, and configure the following options:
 - **Note:** The Application Load Balancer automatically performs *health checks* on all instances to ensure that they are responding to requests. The default settings are recommended, but you make them slightly faster for use in this lab.
 - **Healthy threshold:** Enter 2.
 - **Interval:** Enter 10 (seconds).
1. The configurations you have chosen results in the health check being performed every 10 seconds. If the instance responds correctly twice in a row, it is considered healthy.
 2. Choose **Next**.
 3. The **Step 2: Register targets** screen appears.

4. **Note:** *Targets* are the individual instances that respond to requests from the load balancer.
5. You do not have any web application instances yet, so you can skip this step.
6. Review the settings, and choose **Create target group**.
7. A *Successfully created the target group* message displays.
8. Return to the browser tab where you already started defining the load balancer.
9. In the **Listeners and routing** section, choose the refresh icon.
10. From the **Default action** dropdown list, choose the **Inventory-App** target group that you just created.
11. Scroll to the bottom of the page, and choose **Create load balancer**.
12. The load balancer is successfully created.
13. Choose **View load balancer**.

Task 3: Creating an Auto Scaling group

Amazon EC2 Auto Scaling is a service designed to launch or terminate EC2 instances automatically based on user-defined policies, schedules, and health checks. It also automatically distributes instances across multiple Availability Zones to make applications highly available.

In this task, you create an Auto Scaling group that deploys EC2 instances across your private subnets, which is a security best practice for application deployment. Instances in a private subnet cannot be accessed from the internet. Instead, users send requests to the load balancer, which forwards the requests to EC2 instances in the private subnets.



Task 3.1: Creating an AMI for Auto Scaling

You create an Amazon Machine Image (AMI) from the existing Web Server 1. This saves the contents of the root volume of the web server so that new instances can be launched with an identically configured guest operating system.

1. On the AWS Management Console, in the search box, enter and choose EC2 to open the Amazon EC2 console.
2. In the left navigation pane, choose **Instances**.
3. First, you confirm that the **Web Server 1** instance created for you in this lab is running.
4. Wait until the **Status check** for **Web Server 1** displays *2/2 checks passed*. Choose refresh to update.
5. You now create an AMI based on this instance.
6. Select **Web Server 1**.
7. From the **Actions** dropdown list, choose **Image and templates > Create image**.
8. On the **Create image** page, configure the following options:
 - **Image name:** Enter Web Server AMI.
 - **Image description:** Enter Lab AMI for Web Server.
9. Choose **Create image**.
10. A banner at the top of the screen displays the **AMI ID** for your new AMI.
11. You use this AMI ID when creating the launch template to launch the Auto Scaling group later in the lab.

Task 3.2: Creating a launch template and an Auto Scaling group

You first create a *launch template*. A launch template is a template that an Auto Scaling group uses to launch EC2 instances. When you create a launch template, you specify information for the instances such as the AMI, the instance type, a key pair, and security group.

1. In the left navigation pane, choose **Launch Templates**.
2. Choose **Create launch template**.
3. To create the launch template, configure the following options:
 - In the **Launch template name and description** section, configure the following options:
 - For **Launch template name**, enter Inventory-LT.
 - For **Auto Scaling guidance**, select **Provide guidance to help me set up a template that I can use with EC2 Auto Scaling**.
 - In the **Application and OS Images (Amazon Machine Image)** section, configure the following options:
 - Choose **My AMIs**.
 - For **Amazon Machine Image (AMI)**, choose **Web Server AMI**.
 - In the **Instance type** section, choose **t2.micro**.
 - In the **Key pair (login)** section, for **Key pair name**, choose **vockey**.
 - In the **Network settings** section, configure the following options:

- For **Firewall (security groups)**, choose **Select existing security group**.
- For **Security groups**, choose **Inventory-App**.
- Expand the **Advanced details** section, and configure the following options:
 - For **IAM instance profile**, choose **Inventory-App-Role**.
 - For **Detailed CloudWatch monitoring**, choose **Enable**.
 - **Note:** This option allows Auto Scaling to react quickly to changing utilization.
 - In the **User data** box, enter the following script:
 - `#!/bin/bash`
 - `# Install Apache Web Server and PHP`
 - `yum install -y httpd mysql`
 - `amazon-linux-extras install -y php7.2`
 - `# Download Lab files`
 - `wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACACAD-3-113230/12-lab-mod10-guided-Scaling/s3/scripts/inventory-app.zip`
 - `unzip inventory-app.zip -d /var/www/html/`
 - `# Download and install the AWS SDK for PHP`
 - `wget https://github.com/aws/aws-sdk-php/releases/download/3.62.3/aws.zip`
 - `unzip aws -d /var/www/html`
 - `# Turn on web server`
 - `chkconfig httpd on`
 - `service httpd start`
- 4. Choose **Create launch template**.
- 5. Next, you create an Auto Scaling group that uses this launch template. The Auto Scaling group defines where to launch the EC2 instances.
- 6. In the success message, choose the link for the **Inventory-LT** launch template.
- 7. From the **Actions** dropdown list, choose **Create Auto Scaling group**.
- 8. For **Step 1: Choose launch template or configuration**, configure the following options:
 - **Auto Scaling group name:** Enter **Inventory-ASG** (**ASG** stands for Auto Scaling group).
 - **Launch template:** Confirm that the **Inventory-LT** template that you just created is selected.
- 9. Choose **Next**.
- 10. For **Step 2: Choose instance launch options**, configure the following options:
 - **VPC:** Choose **Lab VPC**.
 - **Availability Zones and subnets:** Choose **Private Subnet 1**, and then choose **Private Subnet 2**.
- 11. These settings launch EC2 instances in private subnets across both Availability Zones.
- 12. Choose **Next**.
- 13. For **Step 3: Configure advanced options**, configure the following options:
 - In the **Load balancing** section, configure the following options:
 - Choose **Attach to an existing load balancer**.
 - From the **Existing load balancer target groups** dropdown list, choose **Inventory-App**.

Step 1 Choose launch template

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.

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

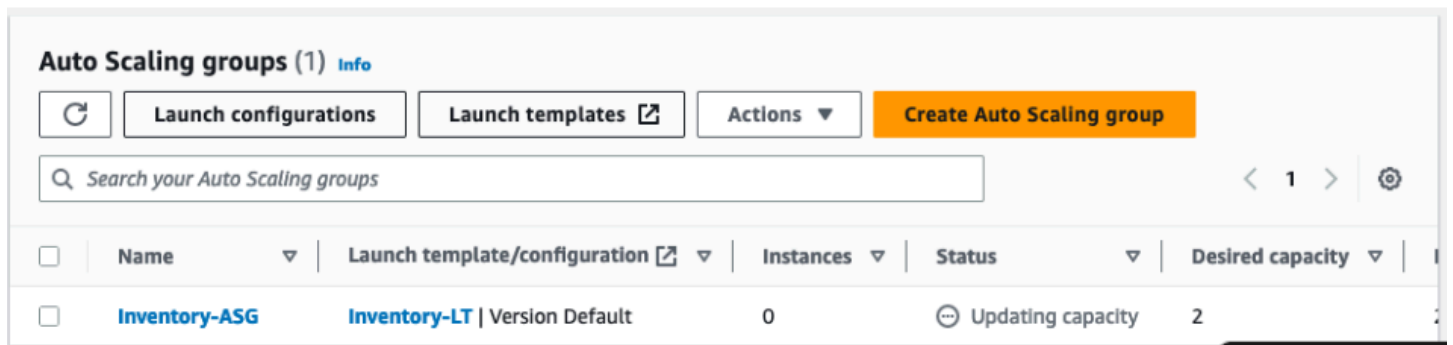
Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.

Select target groups

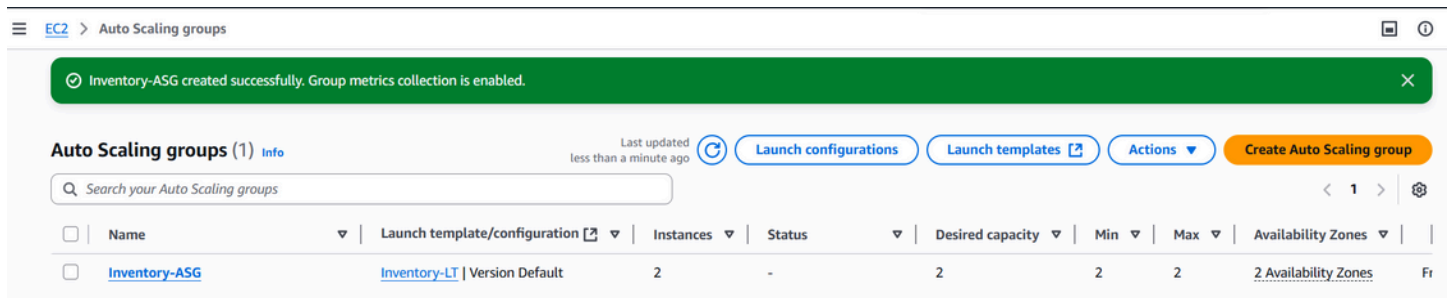
Inventory-App | HTTP
Application Load Balancer: Inventory-LB

- In the **Health checks** section, configure the following options:
 - Select **Turn on Elastic Load Balancing health checks**.
 - For **Health check grace period**, enter 90 seconds.
 - In the **Additional settings** section, select **Enable group metrics collection within CloudWatch**.
 - This setting captures metrics at 1-minute intervals, which allows Auto Scaling to react quickly to changing usage patterns.
1. Choose **Next**.
 2. For **Step 4: Configure group size and scaling policies**, configure the following options:
 - For **Group size**, for **Desired capacity**, enter 2.
 - For **Scaling**, configure the following options:
 - For **Min desired capacity**, enter 2.
 - For **Max desired capacity**, enter 2.
 - These settings allow Auto Scaling to automatically add or remove instances, always keeping 2–6 instances running.
 - For **Automatic scaling - optional**, choose **No scaling policies**.
 - For this lab, you maintain two instances at all times to ensure high availability. If the application is expected to receive varying loads of traffic, you can also create *scaling policies* that define when to launch or terminate instances. However, you do not need to create scaling policies for the inventory application in this lab.
 3. Choose **Next**.
 4. For **Step 5: Add notifications**, you do not need to configure any settings. Choose **Next**.
 5. For **Step 6: Add tags**, choose **Add tag**, and configure the following options:
 - **Key:** Enter Name.
 - **Value:** Enter Inventory-App.
 6. These settings tag the Auto Scaling group with a name, which also appears on the EC2 instances that are launched by the Auto Scaling group. You can use tags to identify which EC2 instances are associated with which application. You could also add tags such as **Cost Center** to assign application costs in the billing files.
 7. Choose **Next**.

8. For **Step 7: Review**, review the details of your Auto Scaling group, and then choose **Create Auto Scaling group**.
9. The **Inventory-ASG** Auto Scaling group appears in the console. Your Auto Scaling group initially shows an instance count of zero, but new instances are launched to reach the desired count of two instances.

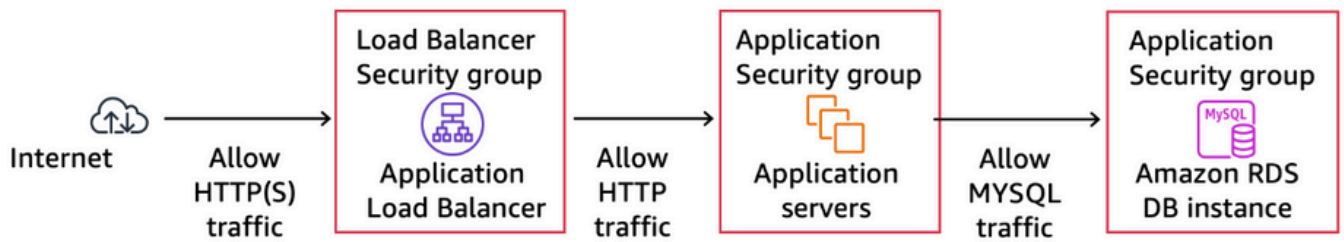


1. The columns show the following information:
 - The group currently has no **Instances**, but the **Status** column indicates *Updating capacity*.
 - The **Desired capacity** is **2** instances. Amazon EC2 Auto Scaling attempts to launch two instances to reach the desired quantity.
 - The **Min** and **Max** are also set to **2** instances. Amazon EC2 Auto Scaling tries to always provide two instances, even if failure occurs.
 - Your application soon runs across two Availability Zones. Amazon EC2 Auto Scaling maintains that configuration even if an instance or Availability Zone fails.
2. After a minute, choose the refresh icon to update the display. It should show that **2** instances are running.



Task 4: Updating security groups

The application you deployed is a three-tier architecture. You now configure the security groups to enforce these tiers:



Task 4.1: Configuring the load balancer security group

You already configured the load balancer security group when you created the load balancer. It accepts all incoming HTTP and HTTPS traffic.

The load balancer has been configured to forward incoming requests to a target group. When Auto Scaling launches new instances, it automatically adds those instances to the target group.

Task 4.2: Configuring the application security group

The application security group was provided as part of the lab setup. You now configure it to accept only incoming traffic from the load balancer.

1. In the left navigation pane, choose **Security Groups**.
2. Select **Inventory-App**.
3. Choose the **Inbound rules** tab.

The security group is currently empty. You now add a rule to accept incoming HTTP traffic from the load balancer. You do not need to configure HTTPS traffic because the load balancer was configured to forward HTTPS requests through HTTP. This practice offloads security to the load balancer, reducing the amount of work that is required by the individual application servers.

1. Choose **Edit inbound rules**.
2. On the **Edit inbound rules** page, choose **Add rule**, and configure the following options:
 - For **Type**, choose **HTTP**.
 - For **Port**, enter 80.
 - For **Source**, configure the following options:
 - Choose the search box to the right of **Custom**.
 - Delete the current contents.

- Enter **sg**.
 - From the list that appears, select **Inventory-LB**.
 - For **Description**, enter Traffic from load balancer.
3. Choose **Save rules**.
 4. The application servers can now receive traffic from the load balancer. This includes health checks that the load balancer performs automatically.

Task 4.3: Configuring the database security group

You now configure the database security group to accept only incoming traffic from the application servers.

1. In the **Security groups** list, select **Inventory-DB** and make sure that no other security groups are selected.
2. The existing rule permits traffic on port 3306 (used by MySQL) from any IP address within the VPC. This is a good rule, but security can be restricted further.
3. In the **Inbound rules** tab, choose **Edit inbound rules**, and configure the following options:
 - For the existing rule, choose **Delete**.
 - Choose **Add rule**.
 - For **Type**, choose **MYSQL/Aurora**.
 - For **Source**, configure the following options:
 - Choose the search box to the right of **Custom**.
 - Enter **sg**.
 - From the dropdown list, select **Inventory-App**.
 - For **Description**, enter Traffic from application servers.
4. Choose **Save rules**.
5. You have now configured three-tier security. Each element in the tier accepts traffic from only the tier above.
6. In addition, the use of private subnets means that you have two security barriers between the internet and your application resources. This architecture follows the best practice of applying multiple layers of security.

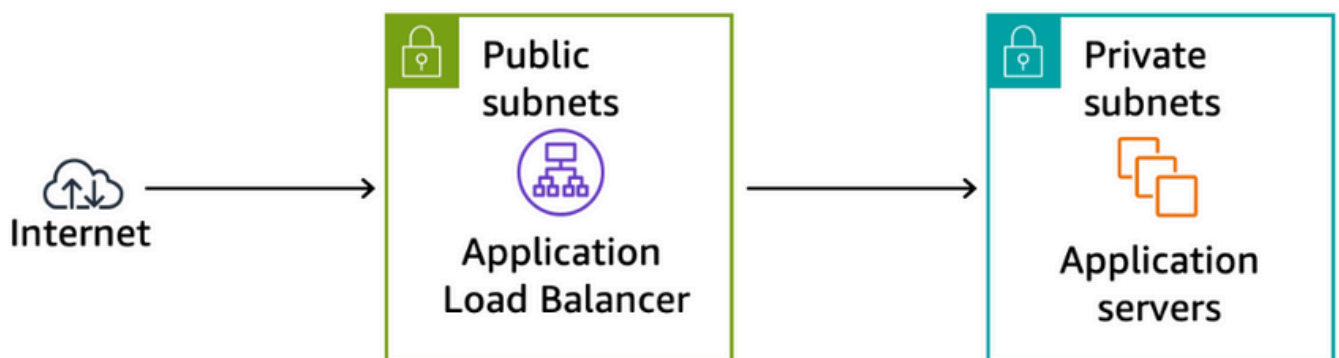
Task 5: Testing the application

Your application is now ready for testing.

In this task, you confirm that your web application is running. You also test that it is highly available.

1. In the left navigation pane, choose **Target Groups**.
2. Select **Inventory-App**.
3. Choose the **Targets** tab.
4. This tab should show two **Registered targets**. The **Health status** column shows the results of the load balancer health check that is performed against the instances.

5. In the **Registered targets** area, choose the refresh icon until the **Status** for both instances appears as *healthy*.
6. If the status does not eventually change to *healthy*, ask your educator for help with diagnosing the configuration.
7. You test the application by connecting to the load balancer, which then sends your request to one of the EC2 instances. You first need to retrieve the DNS name of the load balancer.
8. In the left navigation pane, choose **Load Balancers**, and then choose **Inventory-LB**.
9. In the **Details** tab in the lower half of the window, copy the **DNS name** to your clipboard.
10. It should be similar to **Inventory-LB-xxxx.elb.amazonaws.com**.
11. In a new web browser tab, paste the DNS name from your clipboard, and press Enter.
12. The load balancer forwarded your request to one of the EC2 instances. The instance ID and Availability Zone are shown at the bottom of the webpage.
13. Reload the page in your web browser. You should notice that the instance ID and Availability Zone sometimes toggle between the two instances.
14. When this web application displays, the following is the flow of data over the network:
15.
 - You send the request to the load balancer, which resides in the public subnets that are connected to the internet.
 - The load balancer chooses one of the EC2 instances that resides in the private subnets and forwards the request to it.
 - The EC2 instance then returns the webpage content to the load balancer, which returns it to your web browser.



Task 6: Testing high availability

Your application is configured to be highly available. You can prove the application's high availability by terminating one of the EC2 instances.

1. Return to the **EC2 console** tab in your web browser. Do not close the web application tab. You return to it soon.
2. In the left navigation pane, choose **Instances**.
3. You now terminate one of the web application instances to simulate a failure.
4. Select one of the **Inventory-App** instances. It does not matter which one you select.

5. Choose **Instance state > Terminate instance**.
6. In the **Terminate instance?** window, choose **Terminate**.
7. In a short time, the load balancer health checks notice that the instance is not responding. The load balancer automatically routes all requests to the remaining instance.
8. Return to the web application tab in your web browser, and reload the page several times.
9. You should notice that the **Availability Zone** that is shown at the bottom of the page stays the same. Although an instance failed, your application remains available.
10. After a few minutes, Amazon EC2 Auto Scaling also notices the instance failure. It is configured to keep two instances running, so Amazon EC2 Auto Scaling automatically launches a replacement instance.
11. Return to the Amazon EC2 console tab where you have the instances list displayed. In the upper-right area, choose the refresh icon every 30 seconds until a new EC2 instance appears.
12. After a few minutes, the health check for the new instance should become healthy. The load balancer resumes sending traffic between the two Availability Zones. You can reload your web application tab to see this happen.
13. This task demonstrates that your application is now highly available.

Successfully initiated termination (deletion) of i-0bce33d8abb72f9d6

Instances (6) Info

Find Instance by attribute or tag (case-sensitive)

| | Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability Zone | Public IPv4 |
|--------------------------|---------------|---------------------|----------------|---------------|-------------------|---------------|-------------------|-------------|
| <input type="checkbox"/> | Inventory-App | i-0bce33d8abb72f9d6 | Terminated | t2.micro | - | View alarms + | us-east-1b | - |
| <input type="checkbox"/> | Inventory-App | i-04404af224eea7e6e | Running | t2.micro | Initializing | View alarms + | us-east-1b | - |
| <input type="checkbox"/> | Inventory-App | i-0c3623b282de771d5 | Terminated | t2.micro | - | View alarms + | us-east-1b | - |
| <input type="checkbox"/> | Inventory-App | i-0f3e1fdc3ab1b3903 | Running | t2.micro | 2/2 checks passed | View alarms + | us-east-1a | - |
| <input type="checkbox"/> | Inventory-App | i-0963a8992d2b52154 | Terminated | t2.micro | - | View alarms + | us-east-1a | - |

Successfully initiated termination (deletion) of i-0bce33d8abb72f9d6

Instances (6) Info

Find Instance by attribute or tag (case-sensitive)

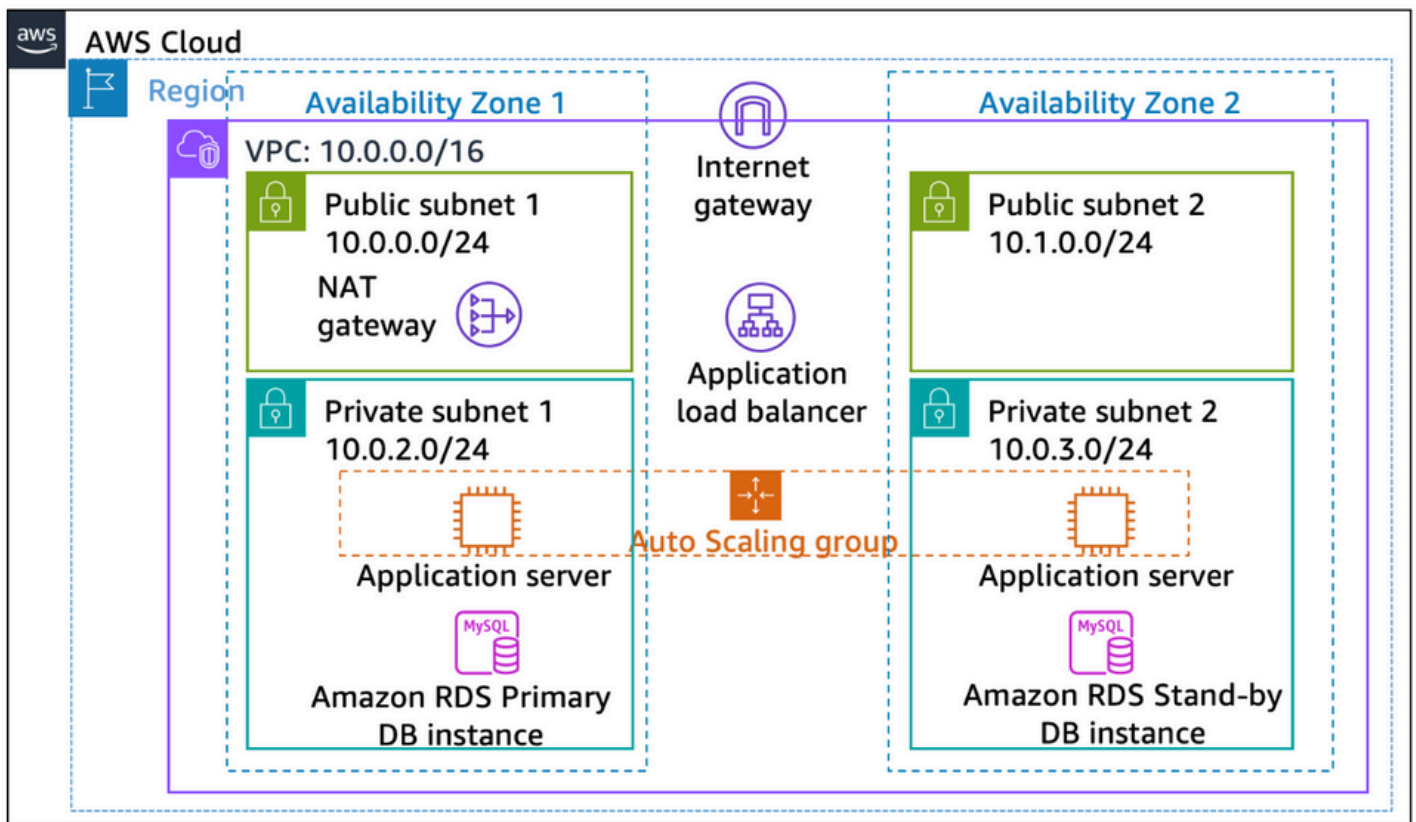
| | Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability Zone | Public IPv4 |
|--------------------------|---------------|---------------------|----------------|---------------|-------------------|---------------|-------------------|-------------|
| <input type="checkbox"/> | Inventory-App | i-0bce33d8abb72f9d6 | Terminated | t2.micro | - | View alarms + | us-east-1b | - |
| <input type="checkbox"/> | Inventory-App | i-04404af224eea7e6e | Running | t2.micro | 2/2 checks passed | View alarms + | us-east-1b | - |
| <input type="checkbox"/> | Inventory-App | i-0c3623b282de771d5 | Terminated | t2.micro | - | View alarms + | us-east-1b | - |
| <input type="checkbox"/> | Inventory-App | i-0f3e1fdc3ab1b3903 | Running | t2.micro | 2/2 checks passed | View alarms + | us-east-1a | - |
| <input type="checkbox"/> | Inventory-App | i-0963a8992d2b52154 | Terminated | t2.micro | - | View alarms + | us-east-1a | - |

Optional task 1: Making the database highly available

This task is optional. You can work on this task if you have remaining lab time.

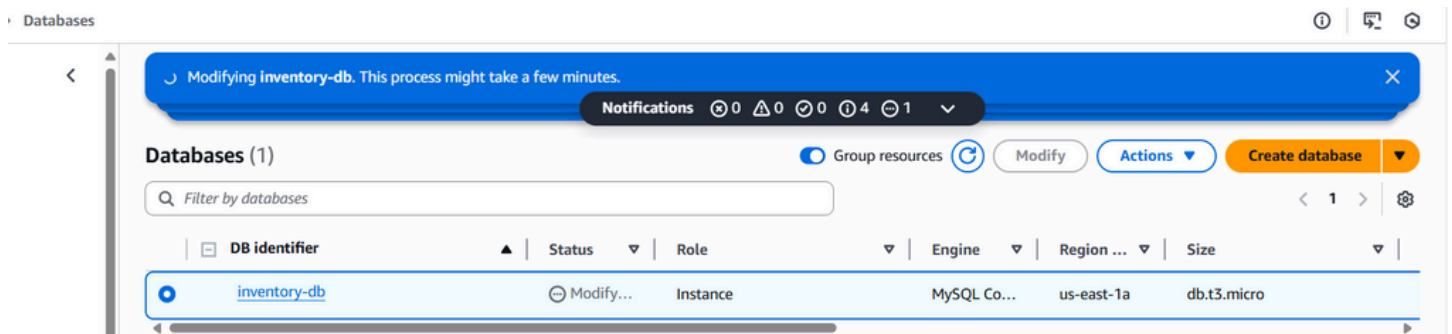
The application architecture is now highly available. However, the Amazon RDS database operates from only one database instance.

In this optional task, you make the database highly available by configuring it to run across multiple Availability Zones (that is, in a Multi-AZ deployment).



1. On the AWS Management Console, in the search box, enter and choose RDS to open the Amazon RDS console.
2. In the left navigation pane, choose **Databases**.
3. Choose the link for the name of the **inventory-db** instance.
4. Explore the information about the database.
5. Choose Modify.
6. In the **Availability & durability** section, for **Multi-AZ deployment**, choose **Create a standby instance**.
7. **Analysis:** You need to reconfigure this one setting to convert the database to run across multiple data centers (Availability Zones).
8. This option does not mean that the database is distributed across multiple instances. Instead, one instance is the primary instance, which handles all requests. Another instance is launched as the standby instance, which takes over if the primary instance fails. Your application continues to use the same DNS name for the database. However, the connections automatically redirect to the currently active database server.
9. You can scale an EC2 instance by changing attributes, and you can also scale an Amazon RDS database this way. You now scale up the database.
10. In the **Instance configuration** section, for **DB instance class**, choose **db.t3.small**.
11. This action doubles the size of the instance.
12. In the **Storage** section, for **Allocated storage**, enter 20.

13. You can explore the other options on the page, but do not change any other settings.
14. At the bottom of the page, choose **Continue**.
15. This change impacts database performance. Therefore, these changes can be scheduled during a defined maintenance window, or they can be run immediately.
16. For **Schedule modifications**, choose **Apply immediately**.
17. Choose **Modify DB instance**.
18. The **Status** of the database is *Modifying* while it applies the changes. You do not need to wait for it to complete.



Optional task 2: Configuring a highly available NAT gateway

This task is optional. You can work on this task if you have remaining lab time.

The application servers run in a private subnet. If the servers must access the internet (for example, to download data), the requests must be redirected through a NAT gateway. (The NAT gateway must be located in a public subnet.)

The current architecture has only one NAT gateway in public subnet 1. Thus, if Availability Zone 1 fails, the application servers are not be able to communicate with the internet.

In this optional task, you make the NAT gateway highly available by launching another NAT gateway in the other Availability Zone. The resulting architecture will be highly available:

1. On the AWS Management Console, in the search box, enter and choose VPC to open the Amazon VPC console.
2. In the left navigation pane, choose **NAT gateways**.
3. The existing NAT gateway displays. You now create a NAT gateway for the other Availability Zone.
4. Choose **Create NAT gateway**.
5. On the **Create NAT gateway** page, configure the following options:
 - For **Name -optional**, enter NatGateway2.
 - For **Subnet**, choose **Public Subnet 2**.
 - Choose **Allocate Elastic IP**.
6. Choose **Create NAT gateway**.
7. You now create a new route table for private subnet 2. This route table redirects traffic to the new NAT gateway.

8. In the left navigation pane, choose **Route tables**.
9. Choose **Create route table**.
10. On the **Create route table** page, configure the following options:
 - **Name:** Private Route Table 2
 - **VPC:** Lab VPC
11. Choose **Create route table**.
12. Choose the **Routes** tab, and observe the settings.
13. Currently, one route directs all traffic locally. You now add a route to send internet-bound traffic through the new NAT gateway.
14. Choose **Edit routes**.
15. On the **Edit routes** page, configure the following options:
 - Choose **Add route**.
 - For **Destination**, enter 0.0.0.0/0.
 - For **Target**, choose **NAT Gateway**, and then choose **NatGateway2**.
16. Choose **Save changes**.
17. Choose the **Subnet associations** tab.
18. Choose **Edit subnet associations**.
19. Select **Private Subnet 2**.
20. Choose **Save associations**.
21. This action now sends internet-bound traffic from private subnet 2 to the NAT gateway that is in the same Availability Zone.
22. Your NAT gateways are now highly available. A failure in one Availability Zone does not impact traffic in the other Availability Zone.

Conclusion

Congratulations! You now have successfully done the following:

- Inspected a provided VPC
- Created an Application Load Balancer
- Created an Auto Scaling group
- Tested the application for high availability