Creating a Highly Available Environment

Critical business systems should be deployed as Highly Available applications, meaning that they can remain operational even when some components fail. To achieve High Availability in AWS, it is recommended to **run services across multiple Availability Zones**.

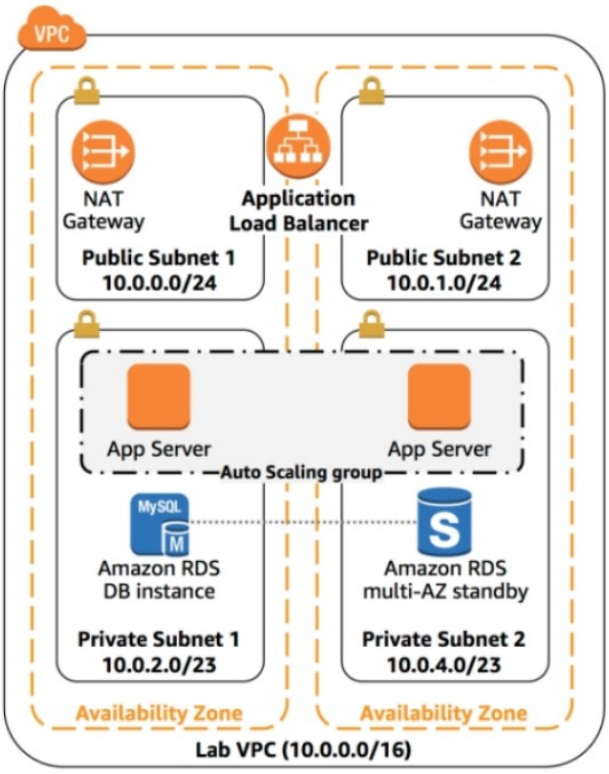
Many AWS services are inherently highly available, such as Load Balancers, or can be configured for high availability, such as deploying Amazon EC2 instances in multiple Availability Zones.

In this lab, you will start with an application running on a single Amazon EC2 instance and will then convert it to be Highly Available.

In this lab you will:

- Inspect a provided VPC
- Create an Application Load Balancer
- Create an Auto Scaling Group
- Test the application for High Availability

The final product of your lab will be this:



Duration

This lab will require approximately **40 minutes** to complete.

Accessing the AWS Management Console

⚠ **Windows Users**: Please use Chrome or Firefox as your web browser for this lab. The lab instructions are **not compatible with Internet Explorer** due to a difference in the Amazon RDS console.

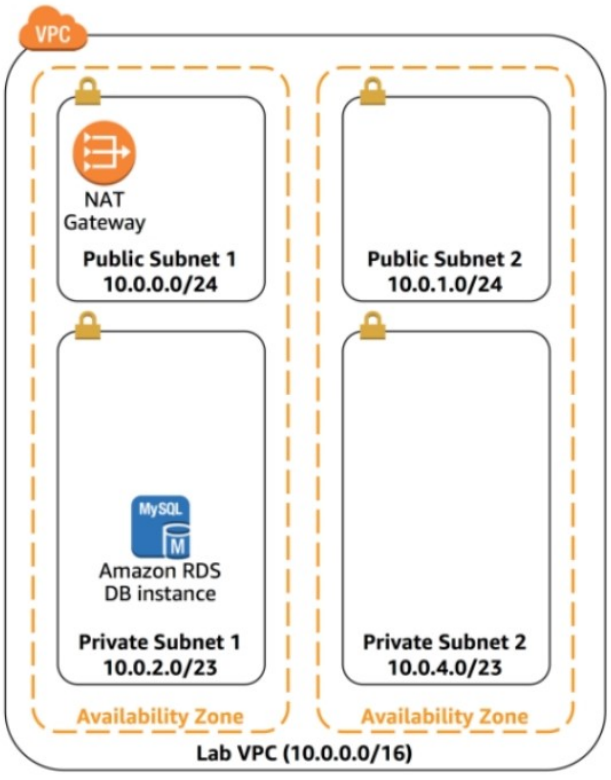
Sign to the AWS Management Console using your credentials.

⚠ Please do not change the Region during this lab.

Task 1: Inspect your VPC

This lab begins with an environment already deployed via AWS CloudFormation including:

- An Amazon VPC
- Public and Private Subnets in two Availability Zones
- An Internet Gateway (not shown) associated with the Public subnets
- A NAT Gateway in one of the Public subnets
- An Amazon RDS instance in one of the Private subnets



In this task, you will review the configuration of the VPC that has already been created.

1. On the **AWS Management Console,** on the **Services** menu, click **VPC**.

2. In the left navigation pane, under **Filter by VPC**, click in the **Select a VPC box** and **select Lab VPC**.

This will limit the console to only show resources associated with the Lab VPC.

3. In the left navigation pane, click **Your VPCs**.

Here you can see the **Lab VPC** that has been created for you.

In the **CIDR** column, you will see a value of **10.0.0.0/20,** which means this VPC includes 4,096 IPs between 10.0.0.0 and 10.0.15.255 (with some reserved and unusable).

4. In the left navigation pane, click **Subnets**.

Here you can see **Public Subnet 1:**

- In the **VPC** column, you can see that this subnet exists inside of **Lab VPC**.
- In the **IPv4 CIDR** column, you can see a value of **10.0.0.0/24**, which means this subnet includes the 256 IPs (5 of which are reserved and unusable) between 10.0.0.0 and 10.0.0.255.
- In the Availability Zone column, you can see the Availability Zone in which this subnet resides.

5. Select ☑ **Public Subnet 1** to reveal more details at the bottom of the page.

Tip: You can draw the divider up and down to expand the lower window pane.

6. Click the **Route Table** tab in the lower half of the page. Here you can see details about the routing for this subnet:

- The first entry specifies that traffic destined within the VPC's CIDR range (**10.0.0.0/20**) will be 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-). This setting makes it a Public Subnet.

7. Click **the Network ACL tab**.

Here you can see the Network Access Control List (ACL) associated with the subnet. The rules currently permit ALL Traffic to flow in and out of the subnet, but they can be further restricted by using Security Groups.

8. In the left navigation pane, click **Internet Gateways.**

Notice that an Internet Gateway is already associated with Lab VPC.

9. In the left navigation pane, click **Security Groups.**

10. Select ☑ **Inventory DB.**

This is the security group used to control incoming traffic to the Database.

11. Click the **Inbound Rules** tab in the lower half of the page.

It is permitting inbound MySQL/Aurora traffic (port 3306) from anywhere within the VPC (*10.0.0.0/20*). You will later modify this to only accept traffic from the application servers.

12. Click the **Outbound Rules** tab.

By default, Security Groups allow all outbound traffic. However, the can be modified as necessary.

Task 2: Create an Application Load Balancer

To build a Highly Available application, it is best practice to launch resources **in multiple Availability Zones**. Availability Zones are physically separate data centers (or groups of data centers) within the same Region. Running your applications across multiple Availability Zones will provide greater availability in case of failure within a data center.

Given that the application is running on multiple application servers, you will need a way to distribute traffic amongst those servers. This can be accomplished by using a **Load Balancer,** which also performs health checks on instances and only sends requests to healthy instances.

13. On the **Services** menu, click **EC2**.

14. In the left navigation pane, click **Load Balancers** (you might need to scroll down to find it).

15. Click **Create Load Balancer**

Several types of Load Balancers are displayed. Read the descriptions of each type to understand their capabilities.

16. Under **Application Load Balancer**, click **Create**

17. For **Name**, enter: *Inventory-LB*

18. Scroll down to the **Availability Zones** section, then:

- For **VPC**, **select: Lab VPC**

You will now specify which subnets the Load Balancer should use. It will be an Internet-facing load balancer, so you will select both Public Subnets.

19. Click the **first** displayed Availability Zone, then click the **Public Subnet** displayed underneath.

20. Click the **second** displayed Availability Zone, then click the **Public Subnet** displayed underneath.

You should now have two subnets selected: **Public Subnet 1** and **Public Subnet 2.** (If not, go back and try the configuration again.)

21. Click **Next: Configure Security Settings**

A warning is displayed, which recommends using HTTPS for improved security. This is good advice, but is not necessary for this lab.

22. Click **Next: Configure Security Groups**

You will now create a security group that accepts all incoming HTTP and HTTPS traffic.

23. Select **Create a new security group**, then configure:

- ***Security group name:*** *Inventory-LB*
- ***Description:*** *Enable web access to Load Balancer*

24. Configure the existing rule (already shown on the page) as:

- **Type**: HTTP
- **Source**: Anywhere

25. Click **Add Rule** and configure:

- **Type**: HTTPS
- **Source**: Anywhere

These settings will accept all incoming HTTP and HTTPS requests.

26. Click **Next: Configure Routing**

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 upon the URL of the incoming request, such as having requests from mobile apps going to a different set of servers. Your web application will use only one Target Group.

27. For **Name**, enter: *Inventory-App*

28. Expand ▶ **Advanced health check settings.**

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 will make them slightly faster for use in this lab.

29. Configure these values:

- **Healthy threshold**: 2
- **Interval**: 10

This means that the Health Check will be performed every 10 seconds and if the instance responds correctly twice in a row, it will be considered healthy.

30. Click **Next: Register Targets**

Targets are the individual instances that will respond to requests from the Load Balancer. You do not have any web application instances yet, so you can skip this step.

31. Click **Next: Review**

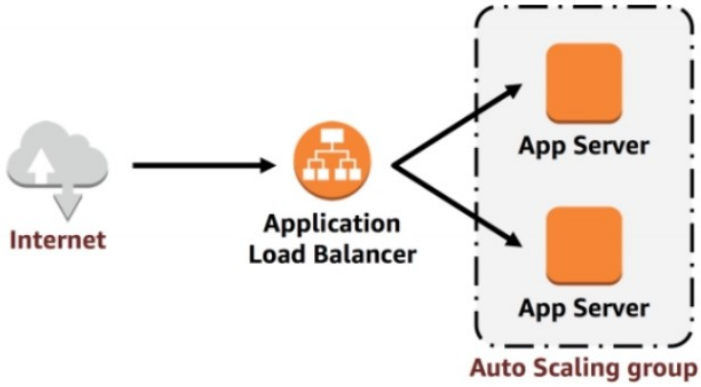
32. Review the settings and click **Create** then **Close**

Your Load Balancer will now be provisioned in the background. There is no need to wait.

Task 3: Create an Auto Scaling Group

Amazon EC2 Auto Scaling is a service designed to launch or terminate Amazon 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 will create an Auto Scaling group that deploys Amazon EC2 instances across your Private Subnets. This is best practice security for deploying applications because instances in a private subnet cannot be accessed from the Internet. Instead, users will send requests to the Load Balancer, which will forward the requests to Amazon EC2 instances in the private subnets.



33. In the left navigation pane, click **Auto Scaling Groups.**

34. Click **Create Auto Scaling group** then click **Get started**

You will first create a Launch Configuration, which defines what type of instances should be launched by Auto Scaling. The interface looks similar to launching an Amazon EC2 instance, but rather than launching an instance it simply stores the configuration for later use.

35. Configure the following:

- Step 1 (Choose AMI):
  - **AMI**: Amazon Linux 2 AMI (at the top)
- Step 2 (Choose Instance Type):
  - **Instance** Type: t2.micro
- Step 3 (Configure details):
  - **Name**: *Inventory-LC*
  - **IAM Role**: Inventory-App-Role
  - **User Data** (Under ▶ **Advanced Details**):

```
#!/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://us-west-2-tcprod.s3.amazonaws.com/courses/ILT-TF-100-ARCHIT/v6.4.0/lab-2-webapp/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
```

- Step 4 (Add Storage)
  - Use default settings (no changes)
- Step 5 (Configure Security Group):
  - **Select an existing security group***: Inventory-App*

You will receive a warning that You will not able able to connect to the instance. This is acceptable because you will not be connecting to the instance. All configuration is done via the User Data script.

- Step 6 (Review):
  - **Create launch configuration**
  - Select ☑ **I acknowledge...**
  - **Create launch configuration**

You will immediately be taken to the **Create Auto Scaling group** configuration screen.

While the Launch Configuration defined **what to launch**, the Auto Scaling group defines **where to launch** the resources.

36. Configure the following:

- Step 1 (Configure Auto Scaling group details):
  - **Group name:** Inventory-ASG (ASG = Auto Scaling Group)
  - **Group size:** 2 instances
  - **Network:** Lab VPC
  - **Subnet:** Select **Private Subnet 1** AND **Private Subnet 2**

You can ignore the warning that says *No public IP addresses will be assigned*. The Amazon EC2 instances will be launched in a private subnet, so they do not require public IP addresses.

- Expand ▶ **Advanced Details**
  - Select ☑ **Load Balancing**
- **Target Groups:** Inventory-App

This tells the Auto Scaling group to register new EC2 instances as part of the Inventory-App target group that you created earlier. The Load Balancer will send traffic to instances that are in this target group.

- Step 2 (Configure scaling policies):
  - **Keep this group at its initial size**

For this lab, you will maintain two instances at all times to ensure High Availability. If the application is expected to receive varying loads of traffic, it is also possible to create scaling policies that define when to launch/terminate instances. However, this is not necessary for the Inventory application in this lab.

- Step 3 (Configure Notifications):
  - (Nothing to configure on this step)
- Step 4 (Configure Tags):
  - **Key**: *Name*
  - **Value**: *Inventory-App*
  - Select ☑ **Tag New Instances**

This will tag the Auto Scaling group with a Name, which will also appear on the EC2 instances launched by the Auto Scaling group. This makes it easier to identify which Amazon EC2 instances are associated with which application. You could also add tags such as Cost Center to make it easier to assign application costs in the billing files.

- Step 5 (Review): o
  - Create Auto Scaling group

The *Inventory-ASG* will appear in the console:

Name	Launch Configuration /	Instances	Desired	Min	Max
Inventory-ASG	Inventory-LC	0	2	2	2

This shows that:

- The group currently has **no instances**, but the i information icon indicates that instances are being launched. (Hover over the icon for more details.)
- The **Desired** quantity is 2 instances, so Auto Scaling will attempt to launch two instances to reach the desired quantity
- The **Min** and **Max** are also set to 2 instances, so Auto Scaling will try to always provide two instances, even in the event of failure.

Your application will soon be running across two Availability Zones and Auto Scaling will maintain that configuration even if an instance or Availability Zone fails.

After a minute, click Refresh to update the display. It should **show 2 instances** running.

Task 4: Update Security Groups

The application you have deployed is a 3-Tier architecture. You will now configure the Security Groups to enforce these tiers:

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 will automatically add those instances to the Target Group.

Application Security Group

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

37. In the left navigation pane, click **Security Groups.**

38. Select ☑ **Inventory-App.**

39. Click the **Inbound** tab in the lower half of the page.

The security group is currently empty. You will now add a rule to accept incoming HTTP traffic from the load balancer. There is no need to configure HTTPS traffic because the load balancer has been configured to forward HTTPS requests via HTTP. This offloads security to the load balancer, reducing the amount of work required by the individual application servers.

40. Click **Edit** and configure:

- **Type**: HTTP
- **Source**:
    - Click in the box beside **Custom**
    - Delete the current contents

    - Type sg
    - Select **Inventory-LB** from the list that appears
- **Description**: Traffic from Load Balancer
- Click **Save**

The application servers can now receive traffic from the Load Balancer. This includes Health Checks that are performed automatically by the Load Balancer.

Database Security Group

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

41. Select ☑ **Inventory-DB** (and make sure no other security groups are selected).

The existing rule permits traffic on port 3306 (used by MySQL) from any IP address within the VPC. This is a good rule, but the security can be tightened further.

42. In the Inbound tab, click Edit and configure:

- Click in the box beside **Custom**
- Delete the current contents
- Type sg
- Select **Inventory-App** from the list that appears
- **Description**: *Traffic from App servers*
- Click **Save**

You have now configured *3-Tier Security*, with each element in the tier only accepting traffic from the tier above.

In addition, the use of private subnets means that there are two security barriers between the Internet and your application resources. This matches the best practice of applying multiple layers of security.

Task 5: Test the Application - Your application is now ready for testing!

In this task, you will confirm that your web application is running and you will test that it is Highly Available.

43. In the left navigation pane, click Target Groups. The Inventory-App group of instances will be displayed.

44. Click the Targets tab in the lower half of the page.

You should see two Registered targets. The Status column shows the results of the Load Balancer Health Check that is performed against the instances.

45. Occasionally click Refresh in the top-right until the Status for both instances appears as healthy.

If the status does not eventually change to healthy, ask your instructor for assistance in diagnosing the configuration. Hovering over the i icon in the Status column will provide more information about the status.

You will be testing the application by connecting to the Load Balancer, which will then send your request to one of the Amazon EC2 instances. You will first need to retrieve the DNS Name of the Load Balancer.

46. In the left navigation pane, click Load Balancers.

47. In the Description tab in the lower half of the window, copy the DNS Name to your clipboard.

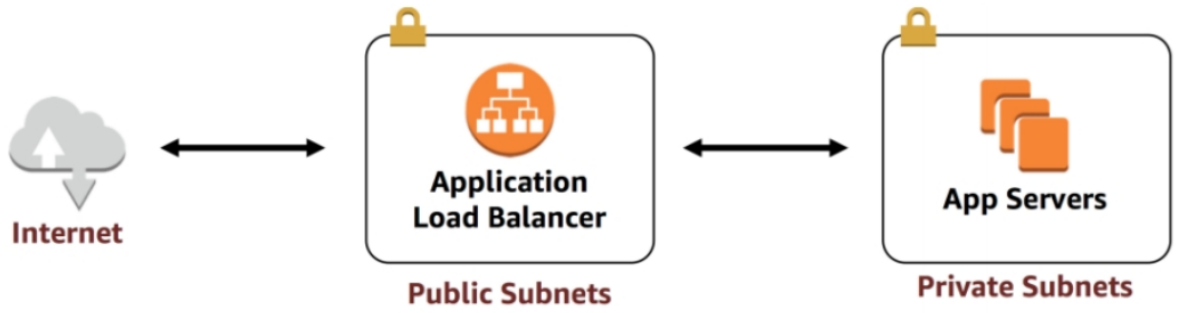
It should be similar to: inventory-LB-xxxx.elb.amazonaws.com

48. Open a new web browser tab, paste the DNS Name from your clipboard and hit Enter.

The Load Balancer forwarded your request to one of the Amazon EC2 instances. The Instance ID and Availability Zone are shown at the bottom of the web page.

49. Reload the page in your web browser. You should notice that the Instance ID and Availability Zone sometimes changes between the two instances.

The flow of information when displaying this web application is:



- You sent the request to the **Load Balancer**, which resides in the public subnets that are connected to the Internet.
- The Load Balancer chose one of the **Amazon EC2 instances** that reside in the private subnets and forwarded the request to it.
- The Amazon EC2 instance then returned the web page to the Load Balancer, which returned it to your web browser.

Task 6: Test High Availability

Your application has been configured to be Highly Available. This can be proven by terminating one of the Amazon EC2 instances.

50. Return to the EC2 Management Console tab in your web browser (but do not close the web application tab — you will return to it soon).

51. In the left navigation pane, click Instances. You will now terminate one of the Web application instances to simulate a failure.

52. Select ☑ one of the Inventory-App instances (it does not matter which one you select).

53. Click Actions then Instance State > Terminate.

In a short time, the Load Balancer health checks will notice that the instance is not responding and will automatically route all requests to the remaining instance.

54. Return to the Web application tab in your web browser and reload the page several times.

You should notice that the Availability Zone shown at the bottom of the page stays the same. Even though an instance has failed, your application remains available!

After a few minutes, Auto Scaling will also notice the instance failure. It has been configured to keep two instances running, so Auto Scaling will automatically launch a replacement instance.

55. Return to the EC2 Management Console tab in your web browser. Click refresh in the top-right every 30 seconds until a new Amazon EC2 instance appears.

After a few minutes, the Health Check for the new instance should become healthy and the Load Balancer will continue sending traffic between two Availability Zones. You can reload your Web application tab to see this happening.

This demonstrates that your application is now Highly Available.

Challenge: Make the Database Highly Available

This challenge is optional and is provided in case you still have lab time remaining.

The application architecture is now Highly Available. However, the Amazon RDS database is still operating from only one database instance.

In this task, you will make the database Highly Available by configuring it to run across multiple Availability Zones ("multi-AZ").

56. On the Services menu, click RDS.

57. In the left navigation pane, click Databases.

58. Click inventory-db. Feel free to peruse the information displayed about the database.

59. Click Modify

60. For Multi-AZ deployment, select Yes.

This is the only step necessary to convert the database to run across multiple data centers (Availability Zones).

Note that this does not mean that the database is distributed across multiple instances. Rather, one instance is the Master, which is handling all requests. Another instance will be launched as the Secondary instance, which will take over in case the Master fails. Your application continues to use the same DNS Name for the database, but the connections will automatically redirect to the currently active database server.

Just as it is possible to scale an Amazon EC2 instance by changing attributes, it is also possible to scale an Amazon RDS database. You will now scale-up the database.

61. For DB instance class, select db.t2.small. This will double the size of the instance.

62. For Storage type, select Provisioned IOPS (SSD).

This is a faster form of storage that runs on Solid State Disks. The speed of the disk can also be configured by changing the Provisioned IOPS ("Input/Output Per Second") to a desired performance level.

The size of the disk can also be increased. Amazon RDS will automatically allocate more disk space to the database.

63. For Allocated storage, enter: 200 This will double the amount of space allocated to the database.

Feel free to peruse the other options on the page, but do not change any of the values.

64. Click Continue at the bottom of the page.

Database performance will be impacted by these changes. Therefore, they can be scheduled during a defined maintenance window, or run immediately.

65. Under Scheduling of Modifications, select Apply immediately.

66. Click Modify DB instance

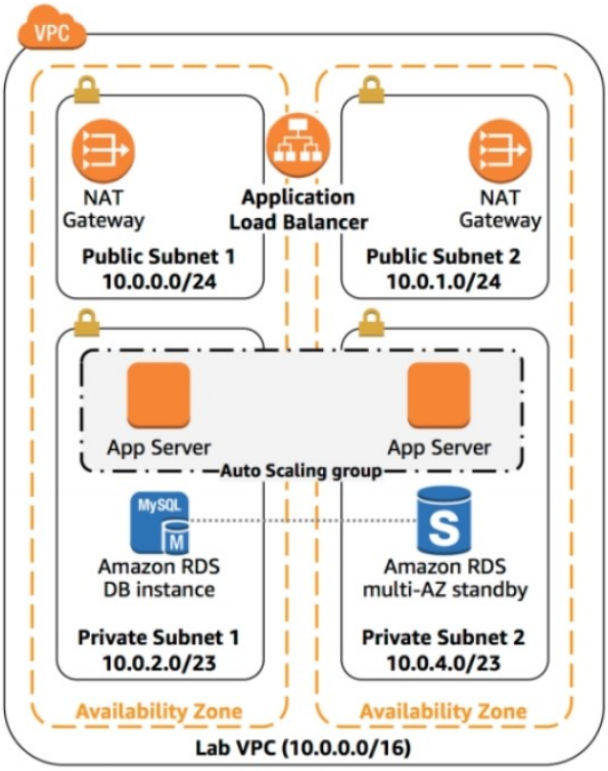
The database will enter a modifying state while changes are applied. There is no need to wait for it to complete.

Challenge: High Availability NAT Gateway

The application servers are running in a private subnet. If they need to access the internet (eg to download data), the requests must be redirected through a NAT Gateway (located in a public subnet).

The current architecture has only one NAT Gateway in Public Subnet 1. This means that if Availability Zone 1 failed, the application servers would not be able to communicate with the internet.

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



67. On the Services menu, click VPC.

68. In the left navigation pane, click NAT Gateways. The existing NAT Gateway is displayed. You will now create one for the other Availability Zone.

69. Click Create NAT Gateway and configure:

- Subnet: Select the subnet that is shown to the left of these instructions as PublicSubnet2
- Click Create New EIP
- Click Create a NAT Gateway
- Click Edit route tables

You will now create a new Route Table for Private Subnet 2 that redirects traffic to the new NAT Gateway.

70. Click Create route table and configure:

- Name tag: Private Route Table 2
- VPC: Lab VPC
- Click Create then click Close

71. Select ☑ Private Route Table 2, ensuring that it is the only route table selected.

72. Click the Routes tab. There is currently one route that directs all traffic locally. You will now add a route to send internet-bound traffic through the new NAT Gateway.

73. Click Edit routes then:

- Click Add route
- Destination: 0.0.0.0/0
- Target: Select NAT Gateway then select the nat-entry that is not the one shown to the left of these instructions
- Click Save routes then click Close

The NAT Gateway shown to the left is for Public Subnet 1. You are configuring the route table to use the other NAT Gateway.

74. Click the Subnet Associations tab.

75. Click Edit subnet associations

76. Select ☑ Private Subnet 2.

77. Click Save

This will now send internet-bound traffic from Private Subnet 2 to the NAT Gateway that is in the same availability zone.

Your NAT Gateways are now highly available. A failure in one AZ will not impact traffic in the other AZ.

Lab Complete

Congratulations! You have completed the lab. Click End Lab at the top of this page to clean up your lab environment.