Configuring and Deploying Amazon VPC for a 3-Tier Web App

**SPL-TF-200-NWCDVW-1 - Version 1.0.13**

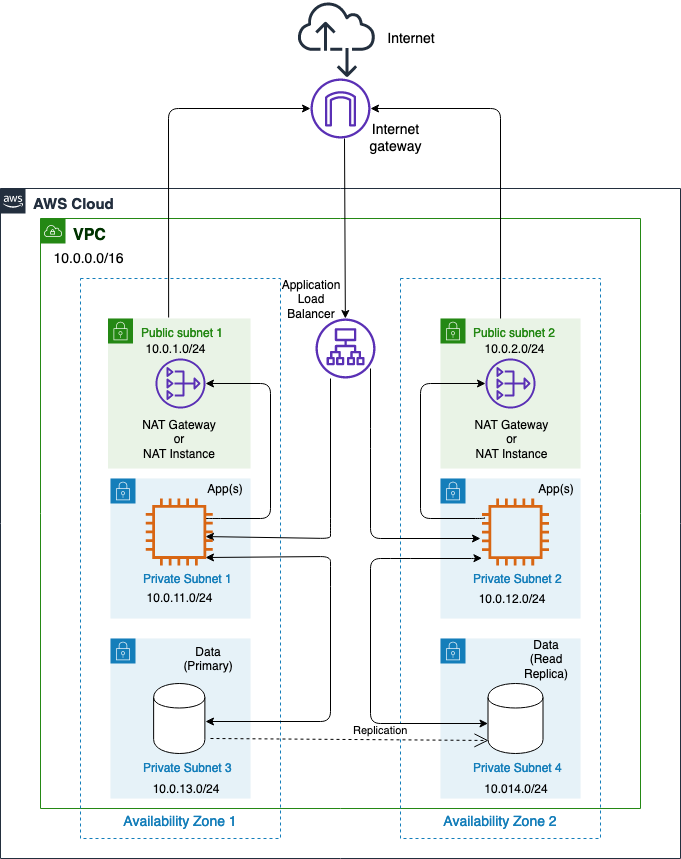
© 2024 Amazon Web Services, Inc. or its affiliates. All rights reserved. This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited. All trademarks are the property of their owners.

Note: Do not include any personal, identifying, or confidential information into the lab environment. Information entered may be visible to others.

Corrections, feedback, or other questions? Contact us at [*AWS Training and Certification*](https://support.aws.amazon.com/#/contacts/aws-training).

**Lab overview**

This lab demonstrates how to set up a VPC infrastructure to support a basic 3-tier web application in your AWS Cloud environment. The VPC will be built across multiple Availability Zones so that your application is highly available. Let’s get a quick overview of what a VPC is and how Elastic Load Balancing works.



**Amazon Virtual Private Cloud (Amazon VPC)** lets you provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define. You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways. You can use both IPv4 and IPv6 in your VPC for secure and easy access to resources and applications.

You can easily customize the network configuration of your Amazon VPC. For example, you can create a public-facing subnet for your web servers that have access to the internet. You can also place your backend systems, such as databases or application servers, in a private-facing subnet with no internet access. You can use multiple layers of security, including security groups and network access control lists, to help control access to Amazon EC2 instances in each subnet.

For more information about VPC you can view the AWS Documentation using the following link:

<https://docs.aws.amazon.com/vpc/latest/userguide/how-it-works.html>

TOPICS COVERED

* Build a VPC
* Create public and private subnets
* Configure the internet gateway and NAT gateway
* Ensure your resources are securely deployed within the VPC

TECHNICAL KNOWLEDGE PREREQUISITES

To successfully complete this lab, you should be familiar with basic navigation of the AWS Management Console and be comfortable editing scripts using a text editor.

**Start lab**

1. To launch the lab, at the top of the page, choose **Start lab**.

**Caution:** You must wait for the provisioned AWS services to be ready before you can continue.

1. To open the lab, choose **Open Console**.

You are automatically signed in to the AWS Management Console in a new web browser tab.

**WARNING:** **Do not change the Region unless instructed.**

COMMON SIGN-IN ERRORS

**Error: You must first sign out**



If you see the message, **You must first log out before logging into a different AWS account:**

* Choose the **click here** link.
* Close your **Amazon Web Services Sign In** web browser tab and return to your initial lab page.
* Choose **Open Console** again.

**Error: Choosing Start Lab has no effect**

In some cases, certain pop-up or script blocker web browser extensions might prevent the **Start Lab** button from working as intended. If you experience an issue starting the lab:

* Add the lab domain name to your pop-up or script blocker’s allow list or turn it off.
* Refresh the page and try again.

**Task 1: Creating a VPC**

In this task, you create a new VPC in the AWS Cloud.

1. In the AWS Management Console, use the **AWS search bar** to search for

VPC

 and then choose the service from the list.

**Note**: You will see a default VPC (one is created whenever an AWS account is created). To learn more about default VPC, go to [Default VPC and default subnets](https://docs.aws.amazon.com/vpc/latest/userguide/default-vpc.html)

1. In the navigation pane, choose **Your VPCs**.
2. Choose **Create VPC**.
3. Under **Resources to create** choose **VPC only** and enter the following:

* **Name tag** *- optional*: Enter

LabVPC

* **IPv4 CIDR**: Enter

10.0.0.0/16

**Note**: This VPC will not have an IPv6 CIDR block, and we will leave it with default tenancy.

1. Choose **Create VPC**

A VPC with the specified CIDR block has been created. Now, let’s create the subnets.

**Task 2: Creating subnets**

A subnet is a range of IP addresses in your VPC. You can launch AWS resources into a specified subnet. Use a public subnet for resources that must be connected to the internet and a private subnet for resources that won’t be connected to the internet. To protect the AWS resources in each subnet, you can use multiple layers of security, including security groups and network access control lists (ACL).

In this task, you create two public subnets and four private subnets in the Lab VPC (as shown in the Lab Overview’s architecture diagram above).

CREATE PUBLIC SUBNETS

The public subnets will be for internet-facing resources.

1. In the left navigation pane, choose **Subnets**.

**Note**: You will see subnets for the default VPC. You can ignore them and go to the next step.

1. Choose **Create subnet** and configure it with the following details:

* **VPC ID**: Select

LabVPC

* **Subnet name**: Enter

PublicSubnet1

* **Availability Zone**: Select the first Availability Zone in the list
* **IPv4 subnet CIDR block**: Enter

10.0.1.0/24

1. Choose **Create subnet**

**Note**: When you create a VPC, you must specify an IPv4 CIDR block for the VPC. The allowed block size is between a /16 netmask (65,536 IP addresses) and /28 netmask (16 IP addresses). The CIDR block of a subnet can be the same as the CIDR block for the VPC (for a single subnet in the VPC) or a subset of the CIDR block for the VPC (for multiple subnets). If you create more than one subnet in a VPC, the CIDR blocks of the subnets cannot overlap.

Tools are available on the internet to help you calculate and create IPv4 subnet CIDR blocks; for example, [IPv4 Address Planner](https://network00.com/NetworkTools/IPv4AddressPlanner/).

One public subnet has been created. Now, create the other public subnet with the following details:

* **VPC ID**: Select

LabVPC

* **Subnet name**: Enter

PublicSubnet2

* **Availability Zone**: Select the second Availability Zone in the list
* **IPv4 subnet CIDR block**:

10.0.2.0/24

1. Choose **Create subnet**

**Note**: Even though we named these subnets PublicSubnet, they are not yet public. A public subnet must have an internet gateway, which you will create and attach later in the lab.

CREATE PRIVATE SUBNETS

The private subnets are for resources that remain isolated from the internet. In this lab, we create two private subnets for the EC2 instances and two private subnets for the RDS instances.

1. Create the four private subnets with the following details:

* **PrivateSubnet1**:
  + **VPC ID**: Select

LabVPC

* + **Subnet name**: Enter

PrivateSubnet1

* + **Availability Zone**: Select the first Availability Zone in the list
  + **IPv4 subnet CIDR block**: Enter

10.0.11.0/24

* **PrivateSubnet2**:
  + **VPC ID**: Select

LabVPC

* + **Subnet name**: Enter

PrivateSubnet2

* + **Availability Zone**: Select the second Availability Zone in the list
  + **IPv4 subnet CIDR block**: Enter

10.0.12.0/24

* **PrivateSubnet3**:
  + **VPC ID**: Select

LabVPC

* + **Subnet name**: Enter

PrivateSubnet3

* + **Availability Zone**: Select the first Availability Zone in the list
  + **IPv4 subnet CIDR block**: Enter

10.0.13.0/24

* **PrivateSubnet4**:
  + **VPC ID**: Select

LabVPC

* + **Name tag**: Enter

PrivateSubnet4

* + **Availability Zone**: Select the second Availability Zone in the list
  + **IPv4 subnet CIDR block**: Enter

10.0.14.0/24

Your VPC now has six subnets in total: two public subnets for the Application Load Balancer in two different Availability Zones, two subnets for the EC2 instances in two different Availability Zones, and two subnets for the RDS instances in two different Availability Zones.

**Task 3: Creating an Internet gateway**

An internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in a VPC and the internet. An internet gateway does not impose availability risks or bandwidth constraints on network traffic.

An internet gateway serves two purposes:

* Provide a target in route tables to connect to the internet
* Perform network address translation (NAT) for instances that have been assigned public IPv4 addresses

In this task, you create an internet gateway so that internet traffic can access the public subnets.

1. In the left navigation pane, choose **Internet gateways**.

**Note**: A default internet gateway was created with the default VPC. You can ignore this and proceed with the next step.

1. Choose **Create internet gateway** and configure:

* **Name tag**: Enter

LabVPCInternetGateway

1. Choose **Create internet gateway**

Once it’s created, you need to attach the internet gateway to your Lab VPC.

1. Choose **Actions** > **Attach to VPC**.
2. For **VPC**, select

LabVPC

1. Choose **Attach internet gateway**

The internet gateway is now attached to your Lab VPC. You have created an internet gateway and attached it to your VPC, and you must now configure the route table of the public subnets to use the internet gateway. You’ll do that after you create your NAT gateway.

**Task 4: Creating a NAT gateway**

You can use a network address translation (NAT) gateway to enable instances in a private subnet to connect to the internet or other AWS services. The NAT gateway also prevents the internet from initiating a connection with those instances. For more information, see [NAT](https://docs.aws.amazon.com/vpc/latest/userguide/vpc-nat.html).

In this task, you create a NAT gateway so that the private subnets can connect to the internet.

1. In the left navigation pane, choose **NAT gateways**.
2. Choose **Create NAT gateway** and configure:

* **Name** *- optional*: Enter

LabVPCNATGateway

* **Subnet**: Select

PublicSubnet1

* **Connectivity type**: Select **Public**
* **Elastic IP allocation ID**: Choose **Allocate Elastic IP**

This will generate an Elastic IP address and will allocate it to the NAT gateway.

*An Elastic IP address is a public IPv4 address that is reachable from the internet. If your instance does not have a public IPv4 address, you can associate an Elastic IP address with your instance to enable communication with the internet. For example, this allows you to connect to your instance from your local computer.*

1. Choose **Create NAT gateway**

The NAT gateway has been created. You must now configure the route table of the private subnets to use the NAT gateway.

**Task 5: Configuring route tables**

A route table contains a set of rules, called *routes*, used to determine where network traffic is directed. Each subnet in a VPC must be associated with a route table, which controls the routing for the subnet. A subnet can only be associated with one route table at a time, but you can associate multiple subnets with the same route table.

In this task, you configure two route tables, one for the public subnets and one for the private subnets.

CONFIGURE PUBLIC ROUTE TABLE

To use an internet gateway, a subnet’s route table must contain a route that directs internet-bound traffic to the internet gateway. This subnet is called a *public subnet*.

You will:

* Create a public route table for internet-bound traffic
* Add a route to the route table to direct internet-bound traffic to the internet gateway
* Associate the public subnets with the new route table

1. In the left navigation pane, choose **Route tables**.

Several route tables are displayed, but there is only one route table associated with the Lab VPC. This is the main route table.

1. Choose **Create route table** and configure:

* **Name** *- optional*: Enter

PublicRouteTable

* **VPC**: Select

LabVPC

1. Choose **Create route table**
2. Check if **Routes** tab for **PublicRouteTable** is selected.
3. Choose **Edit routes**

Now, add a route to direct internet-bound traffic (0.0.0.0/0) to the internet gateway.

1. Choose **Add route** and configure:

* **Destination**: Enter

0.0.0.0/0

* **Target**: Select **Internet Gateway** from the list and then select **LabVPCInternetGateway**.

1. Choose **Save changes**

The last step is to associate this new route table with the public subnets.

1. Choose the **Subnet associations** tab.
2. Choose **Edit subnet associations**
3. On the Edit subnet associations page, select the rows with **PublicSubnet1** and **PublicSubnet2**.
4. Choose **Save associations**

The PublicSubnet is now public because it has a route table entry that sends traffic to the internet via the internet gateway.

CONFIGURE PRIVATE ROUTE TABLE

To use a NAT gateway, you must update the route table associated with one or more of your private subnets to point internet-bound traffic to the NAT gateway. This enables instances in your private subnets to communicate with the internet, mainly for any upgrades that needs to be installed.

You will:

* Create a private route table for private subnets
* Add a route to the route table for the NAT gateway
* Associate the private subnets with the new route table

1. Choose **Create route table** and configure it with the following details:

* **Name** *- optional*: Enter

PrivateRouteTable

* **VPC**: Select

LabVPC

1. Choose **Create route table**
2. Check if **Routes** tab for **PrivateRouteTable** is selected.
3. Choose **Edit routes**

In the next step, you add a route to direct internet-bound traffic (0.0.0.0/0) to the NAT gateway.

1. Choose **Add route** and configure:

* **Destination**: Enter

0.0.0.0/0

* **Target**: Select **NAT Gateway** from the list and then select **LabVPCNATGateway**.

1. Choose **Save changes**

The last step is to associate this new route table with the private subnets.

1. Choose the **Subnet associations** tab.
2. Choose **Edit subnet associations**
3. On the Edit subnet associations page, select all of the rows with *PrivateSubnet*.

**Note**: You may need to expand the Subnet ID column to see their full names.

1. Choose **Save associations**

You have successfully created a route table and associated the private subnets to the NAT gateway in the public subnet so that the incoming traffic can reach the private subnets.

**Note**: If you have resources in multiple Availability Zones that share one NAT gateway, and if the NAT gateway’s Availability Zone is down, resources in the other Availability Zones lose internet access. To create an AZ-independent architecture, create a NAT gateway in at least two Availability Zones that Amazon CloudWatch can monitor and configure your routing to ensure resources use the NAT gateway in the same Availability Zone.

**Task 6: Creating security groups**

A security group acts as a virtual firewall for instances, controlling inbound and outbound traffic. Security groups operate at the instance network interface level, not the subnet level. Therefore, each instance can have its own firewall that controls traffic. If you do not specify a particular security group at launch time, the instance is automatically assigned to the default security group for the VPC.

In this task, you create three security groups for this lab: one for the RDS instances, one for the EC2 instances, and one for the Application Load Balancer (ALB). *The ALBSecurityGroup allows the application to be accessed from the internet.*

1. In the left navigation pane, choose **Security groups**.
2. Choose **Create security group** and configure it with the following details:

* **Security group name**: Enter

LabVPCALBSG

* **Description**: Enter

Allows web access

* **VPC**: Select *LabVPC*

1. For **Inbound rules**, choose **Add rule** and configure it with the following details:

* **Type** : Select *All traffic*
* **Source type**: Select *Anywhere-IPv4*

1. For **Tags - optional**, choose **Add new tag** and configure it with the following details:

* **Key**: Enter

Name

* **Value** *- optional*: Enter

LabVPCALBSG

**Note**: By adding tags, it will be easy to identify the subnets in the subnet list.

1. At the bottom of the page, choose **Create security group**

The inbound rules determine what traffic is permitted to reach the instance. You have configured it to permit HTTP (port 80) traffic coming from anywhere on the internet (0.0.0.0/0).

You will create the security group for the EC2 instances. *Only the ALBSecurityGroup will be allowed to talk to the EC2 instances.*

1. In the left navigation pane, choose **Security groups**.
2. Create another new security group with the following details:

* **Security group name**: Enter

LabVPCEC2SG

* **Description**: Enter

Allows ALB to access the EC2 instances

* **VPC**: Select *LabVPC*

1. For **Inbound rules**, choose **Add rule** and configure it with the following details:

* **Type**: Select *Custom TCP*
* **Port range**: Enter

80

* **Source**: Select *Custom* and choose

LabVPCALBSG

1. Create another inbound rule with the following details:

* **Type**: Select *Custom TCP*
* **Port range**: Enter

8443

* **Source**: Select *Custom* and choose

LabVPCALBSG

1. Create a new tag with the following details:

* **Key**: Enter

Name

* **Value** *- optional*: Enter

LabVPCEC2SG

1. Choose **Create security group**

You have configured the inbound rules to permit ALB (ports 80 and 8443) traffic to the EC2 instances.

Now, you will create an RDSSecurityGroup so the EC2 instances can communicate to the RDS instances.

1. In the left navigation pane, choose **Security groups**.
2. Create the last security group with the following details:

* **Security group name**: Enter

LabVPCRDSSG

* **Description**: Enter

Allows application to access the RDS instances

* **VPC**: Select *LabVPC*

1. Add the following inbound rule:

* **Type**: Select *MYSQL/Aurora*
* **Port range**: *3306*
* **Source**: Select *Custom* and choose

LabVPCEC2SG

1. Add the following tag:

* **Key**: Enter

Name

* **Value** *- optional*: Enter

LabVPCRDSSG

1. Choose **Create security group**

This will allow the EC2 instances to communicate with the RDS instances on port 3306.

**Note**: You can grant access to a specific CIDR range, or to another security group in your VPC or in a peer VPC (requires a VPC peering connection). When you specify a security group as the source for a rule, traffic is allowed from the network interfaces that are associated with the source security group for the specified protocol and port. Incoming traffic is allowed based on the private IP addresses of the network interfaces that are associated with the source security group (and not the public IP or Elastic IP addresses). Adding a security group as a source does not add rules from the source security group.

**Task 7: Launch web app instances and database resources, and deploy the application**

In this task, you provision the RDS and EC2 instances in the private subnets and configure an Application Load Balancer in the public subnets. To test the VPC, you also deploy a web application to read a population census.

CREATE THE DATABASE IN THE PRIVATE SUBNETS

Create a DB (database) subnet group so that the RDS will be deployed within the subnets you want to use.

1. In the AWS Management Console, use the **AWS search bar** to search for

RDS

 and then choose the service from the list.

1. In the left navigation pane, choose **Subnet groups**.
2. Choose **Create DB subnet group** and configure it with the following details:

* **Name**: Enter

LabVPCRDSsubnetgroup

* **Description**: Enter

Subnet group for RDS

* **VPC**: Select *LabVPC*
* **Availability Zones**: Select the two Availability Zones you used to create the subnets (they should be the first two listed).
* **Subnets**: Choose *10.0.13.0/24* and *10.0.14.0/24*

1. Choose **Create**

The DB subnet group has been created successfully. Now let’s create the database.

1. In the left navigation pane, choose **Databases**.
2. Choose **Create database** and configure it with the following details:

* **Choose a database creation method**: *Standard create*
* **Engine Options**:
  + **Engine type**: Select *Aurora (MySQL-Compatible)*
  + **Available version**: Select *Aurora (MYSQL 5.7) 2.11.5*
* **Templates**: Select *Production*
* **Settings**:
  + **DB cluster identifier**: Enter

LabVPCDBCluster

* + **Master username**: Leave the default as

admin

* + **Credentials management**: Choose **Self managed** option
  + **Master password**: Enter

testingrdscluster

* + **Confirm password**: Enter

testingrdscluster

* **DB instance class**: Select **Burstable classes (includes t classes)** and select **db.t3.small**
* **Availability & durability**: *Create an Aurora Replica or Reader node in a different AZ*

1. In the **Connectivity** section configure following

* **Virtual private cloud(VPC)**: Select *LabVPC*
* **DB subnet group**: Select *labvpcrdssubnetgroup*
* **Public access**: Select *No*
* **VPC security group (firewall)**: Select *Choose existing*
* **Existing VPC security groups**: Select

LabVPCRDSSG

 and remove *default*

* **Additional Configuration and For Database port**: Select *3306*

1. Uncheck *Enable Enhanced monitoring* in the **Monitoring** section.
2. Choose  **Additional Configuration**

* **Initial database name**: Enter

Population

The other values will be left to the default values selected.

1. Choose **Create database**

This will create a writer instance in one Availability Zone and a reader instance in the other Availability Zone. It will take few minutes to complete. Choose the refresh button to see the status updated. You can proceed once you receive the *Successfully created database* message.

1. Choose *LabVPCDBCluster* under **DB identifier** and then copy the *Endpoint name* under **Endpoints** in Connectivity & security section, for the *Writer* type and the *Reader* type. These will be needed when the application is deployed on the EC2 instances.

**Note**: The database may initially show two *Reader* types, but it will show the *Writer* type once the creation is complete.

CONFIGURE THE EC2 INSTANCES

The next step is to create the EC2 instances and deploy the application code. If you want the application to be highly available in different Availability Zones, it’s a good practice to create a *launch template* and reuse it for deploying the EC2 instance instead of creating from scratch every time.

You will create the launch template first.

1. In the AWS Management Console, use the **AWS search bar** to search for

EC2

 and then choose the service from the list.

1. In the left navigation pane, choose **Launch Templates**.
2. Choose **Create launch template** and configure it with the following details:

* **Launch template name and description**:
  + **Launch template name** *- required*: Enter

LabVPCEC2Template

* + **Template version description**: Enter

Template to launch EC2 instance and deploy the application

* **Application and OS Images (Amazon Machine Image)**:
  + **AMI**: Select *Amazon Linux 2 AMI (HVM), SSD Volume Type*
* **Instance type**:
  + **Instance type**: Select *t2.small*
* **Network settings**:
  + **Subnet**: Select *PrivateSubnet1*
  + **Firewall (security groups)**: Choose *Select existing security group*
  + **Security groups**: From the drop down, choose *LabVPCEC2SG*

1. Choose  **Advanced details**

* **IAM instance profile**: Choose the entry that includes

IAMInstanceProfileRole

 in the name.

*This profile role was created in the lab startup process and will allow the EC2 instance to access the S3 bucket.*

* **User data**: Copy and paste the following script into the empty field.

#!/bin/bash

yum update -y

yum install -y python3

aws s3 cp s3://us-west-2-aws-training/courses/SPL-TF-200-NWCDVW/v1.0.13.prod-b203ed8c/scripts/vpcapp.zip .

unzip vpcapp.zip

cd vpcapp

pip3 install -r requirements.txt

export DATABASE\_HOST=${RDSClusterEndPoint}

export DATABASE\_USER=admin

export DATABASE\_PASSWORD=testingrdscluster

export DATABASE\_DB\_NAME=Population

cd loaddatabase

python3 database\_populate.py

cd ..

python3 application.py

1. Replace all of the text after **export DATABASE\_HOST=** with the *Writer* DB *Endpoint name* that you copied earlier. When you do so, be sure to replace all of the text after =, including the $ and { }.

*This script deploys the application on the EC2 instance along with the EC2 launch process.*

1. De-select  **User data has already been base64 encoded**.
2. Choose **Create launch template**
3. At the bottom of the page, choose **View launch templates**
4. In the left navigation pane, choose **Instances** from Instance menu.
5. Choose **Launch instance**
6. Select **Launch instance from template** and configure it with the following details:

* **Choose a launch template**:
  + **Source template**: Select *LabVPCEC2Template*
  + Under Key pair (login) choose **Proceed without a key pair(Not recommended)**
* **Network settings**:
  + **Subnet**: Select *PrivateSubnet1*

1. In the **Resource tags** section, choose **Add new tag** and configure:

* **Key**: Enter

Name

* **Value**: Enter

ApplicationServer1

1. Choose **Launch instance**
2. Choose **Instances**, and you can see the new EC2 instance **ApplicationServer1** being launched.
3. Wait until **ApplicationServer1** shows *2/2 checks passed* in the **Status Checks** column.

* This will take 3-5 minutes. Press the refresh icon **!** in the upper-right pane to check for updates.

The EC2 instance along with the application code has been deployed successfully.

CREATE AN APPLICATION LOAD BALANCER

You will create an Application Load Balancer in the public subnets to access the application from a browser.

1. In the left navigation pane, choose **Load Balancers**.
2. Choose **Create Load Balancer**
3. Choose **Create** in the **Applicaton Load Balancer** column.
4. Configure the **Create Application Load Balancer** with the following details:

* **Basic configuration**:
  + **Load balancer name**: Enter

LabVPCALB

* **Network mapping**:
  + **VPC**: Select

LabVPC

* + **Mappings**:
    - Select the check box for the first Availability Zone listed, and select

PublicSubnet1

 from the subnet list.

* + - Select the check box for the second Availability Zone listed, and select

PublicSubnet2

 from the subnet list.

1. In **Security Groups** section, select

LabVPCALBSG

 from the dropdown.

**Note**: If *default* is selected by default, unselect it.

1. In **Listeners and routing** section:

* Click **Create target group** link.

1. In **Specify group details** page:

* **Target group name** :

LabVPCALBTargetGroup

* Click **Next**

1. In **Register Targets**:

* Under **Available instances** select the EC2 instance that has been deployed in the previous steps.
* Change the value of port to

8443

.

* Click **Include as pending below**

The instance will now appear under the **Review targets**.

1. Choose **Create target group**
2. Come back to the browser tab with **Create Application Load Balancer** and in **Listeners and routing** section:

* Select

LabVPCALBTargetGroup

 from the dropdown

1. Choose **Create load balancer**
2. Choose **View load balancer**

**Note**: The load balancer will appear to be in the *provisioning* state for few minutes and will then change to *active*.

1. Select the check box for the load balancer

LabVPCALB

 if not selected by default.

1. From the **Description** tab, copy the **DNS Name** and paste the value in a new browser to invoke the load balancer.

You should see the application **Population Facts** displayed with two fields: **Choose year** and **Choose Country**. Press the **Submit** button to test the application. You should see the results related to the census for that particular country and year. To keep testing, use the back button to go back and select other values.

**Challenge task**

**Note**: This challenge task is optional and is provided in case you have lab time remaining. Or, skip to the end of the lab [here](https://labs.skillbuilder.aws/sa/lab/arn%3Aaws%3Alearningcontent%3Aus-east-1%3A470679935125%3Ablueprintversion%2FSPL-TF-200-NWCDVW-1%3A1.0.13-e3fc53ac/en-US#conclusion).

Deploy an EC2 instance in the private subnet of the second Availability Zone, connect to the read replica of the Aurora database, and register the EC2 instance in the Application Load Balancer.

Go to [challenge solution](https://labs.skillbuilder.aws/sa/lab/arn%3Aaws%3Alearningcontent%3Aus-east-1%3A470679935125%3Ablueprintversion%2FSPL-TF-200-NWCDVW-1%3A1.0.13-e3fc53ac/en-US#challenge-solution) if you have trouble.

**Conclusion**

 Congratulations! You now have successfully:

* Created a VPC
* Created public and private subnets
* Created an internet gateway and a NAT gateway
* Configured route tables and associated them to subnets
* Deployed a 3-tier web application to the VPC

**End lab**

Follow these steps to close the console and end your lab.

1. Return to the **AWS Management Console**.
2. At the upper-right corner of the page, choose **AWSLabsUser**, and then choose **Sign out**.
3. Choose **End lab** and then confirm that you want to end your lab.

**Additional resources**

For more information about AWS Training and Certification, see [*https://aws.amazon.com/training/*](https://aws.amazon.com/training/).

*Your feedback is welcome and appreciated.*  
If you would like to share any feedback, suggestions, or corrections, please provide the details in our [*AWS Training and Certification Contact Form*](https://support.aws.amazon.com/#/contacts/aws-training).

**Appendix**

**Challenge solution**

This solution guides you on how to create the second EC2 instance in a different Availability Zone and, also, how to configure the instance in the Application Load Balancer.

1. In the left navigation pane, choose **Instances** from Instances menu.
2. Choose **Launch instance**
3. Select **Launch instance from template** and configure it with the following details:

* **Choose a launch template**:
  + **Source template**: Select *LabVPCEC2Template*
  + Under Key pair (login) choose **Proceed without a key pair(Not recommended)**
* **Network settings**:
  + **Subnet**: Select *PrivateSubnet2*

1. In the **Resource tags** section, choose **Add new tag** and configure:

* **Key**: Enter

Name

* **Value**: Enter

ApplicationServer2

1. Choose  **Advanced details**

* **IAM instance profile**: Choose the entry that includes *IAMInstanceProfileRole* in the name.
* **User data**: Copy and paste the following script.

#!/bin/bash

yum update -y

yum install -y python3

aws s3 cp s3://us-west-2-aws-training/courses/SPL-TF-200-NWCDVW/v1.0.13.prod-b203ed8c/scripts/vpcapp.zip .

unzip vpcapp.zip

cd vpcapp

pip3 install -r requirements.txt

export DATABASE\_HOST=${RDSClusterEndPoint}

export DATABASE\_USER=admin

export DATABASE\_PASSWORD=testingrdscluster

export DATABASE\_DB\_NAME=Population

python3 application.py

1. Replace all of the text after **export DATABASE\_HOST=** with the *Reader* DB *Endpoint name* that you copied earlier. When you do so, be sure to replace all of the text after =, including the $ and { }.
2. Choose **Launch instance**.
3. Choose **View launch template**.
4. Choose **Running instances**, and you can see the new EC2 instance **ApplicationServer2** being launched.
5. Choose  button to see the new instance in the list.
6. Wait until **ApplicationServer2** shows *2/2 checks passed* in the **Status Checks** column.

* This will take 3-5 minutes. Press the refresh icon **!** in the upper-right pane to check for updates.

Next step will be to configure the above EC2 instance in the load balancer.

1. In the left navigation pane, choose **Target Groups**.
2. Choose the **LabVPCALBTargetGroup**.
3. Choose the **Targets** tab.
4. Choose **Register Targets**.
5. Select the check box for the

ApplicationServer2

.

1. Change the port value to

8443

 in the **Ports for the selected instances** text box.

1. Click **Include as pending below**

The selected instance will appear in the bottom panel of the registered targets.

1. Choose **Register pending targets**

Wait until the **Status** changes its value from *initial* to *healthy*.

Now the load balancer is configured to talk to both EC2 instances. Try running the application from the browser with the load balancer *DNS Name*. The Availability Zone the data is being read from will appear on the page.