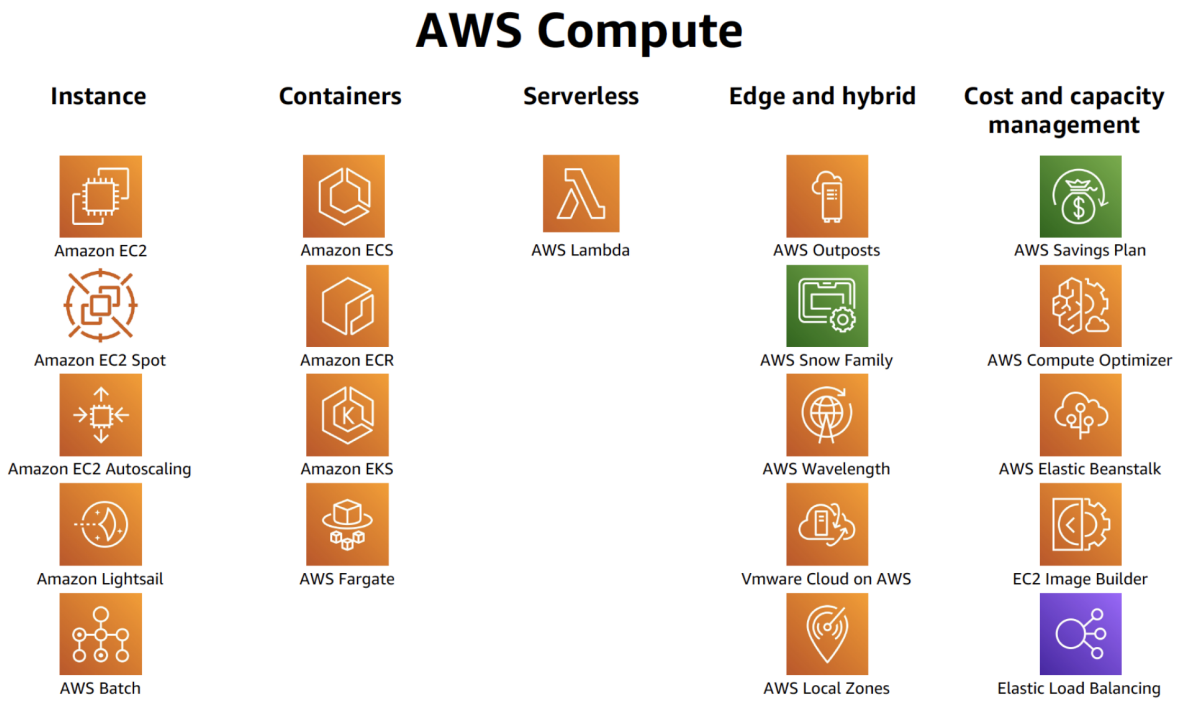
# HW1

## Compute - Amazon EC2




Building and running your organization starts with compute, whether you are building enterprise, cloud-native or mobile apps, or running massive clusters to drive analysis workloads. AWS offers a comprehensive portfolio of **compute services** allowing you to develop, deploy, run, and scale your applications and workloads in the world's most powerful, secure and innovative compute cloud.

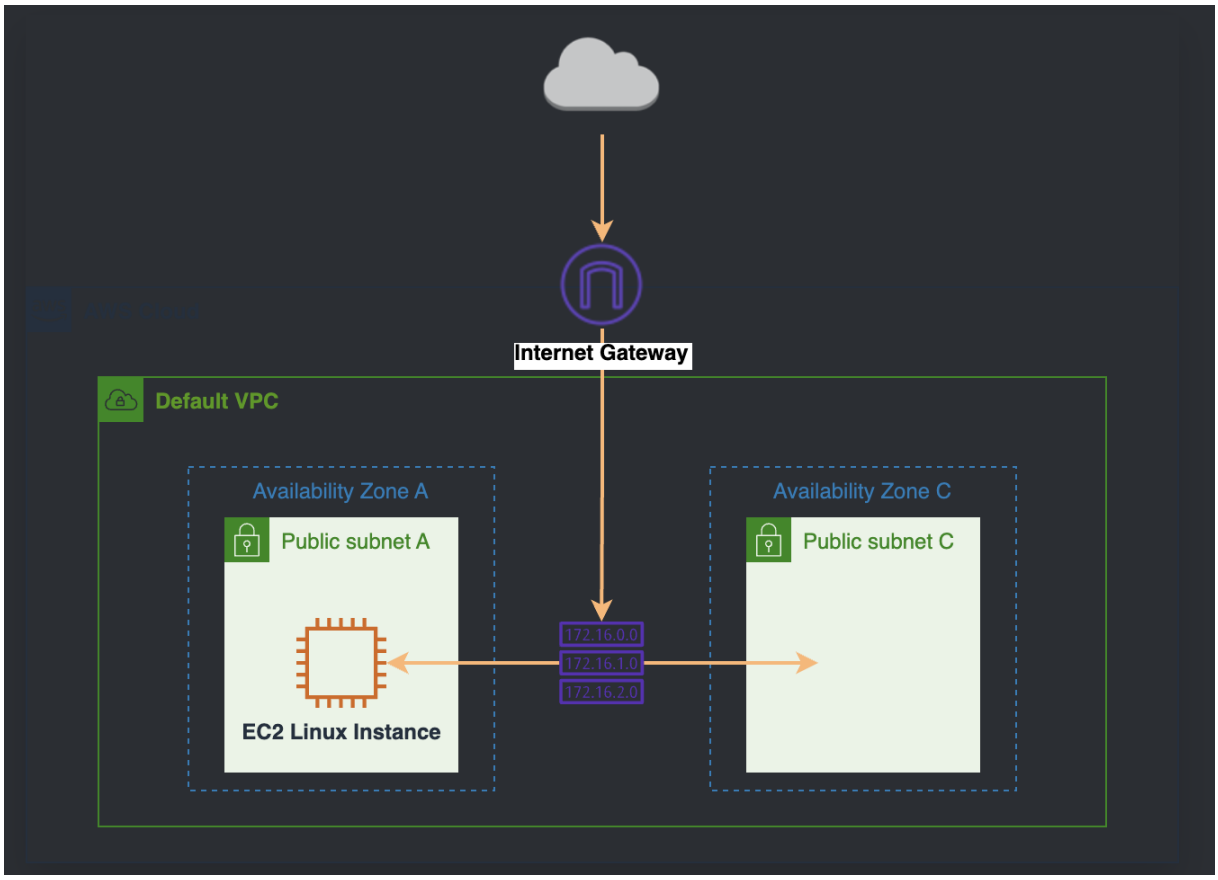
AWS computing services have the following characteristics:

- Right compute for your workloads
- Accelerate from idea to market
- Offer built in security
- Flexibility to optimize costs
- Provide compute resource where you need it

▼ **EC2 Linux Hands on Lab**

### Amazon EC2 Overview

Amazon EC2 provides scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.




Create your own web server by going through the labs in the order below:

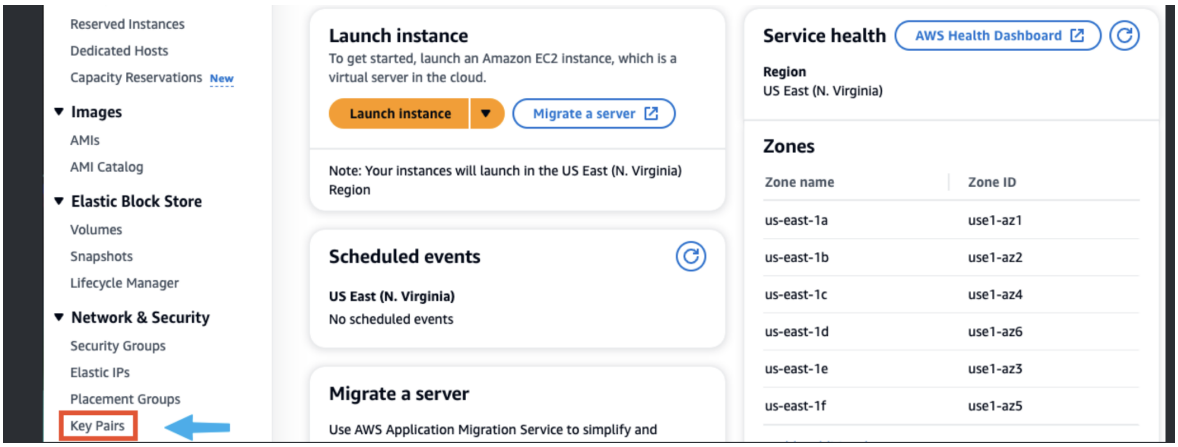
1. Create a new key pair

# Create a new Key Pair

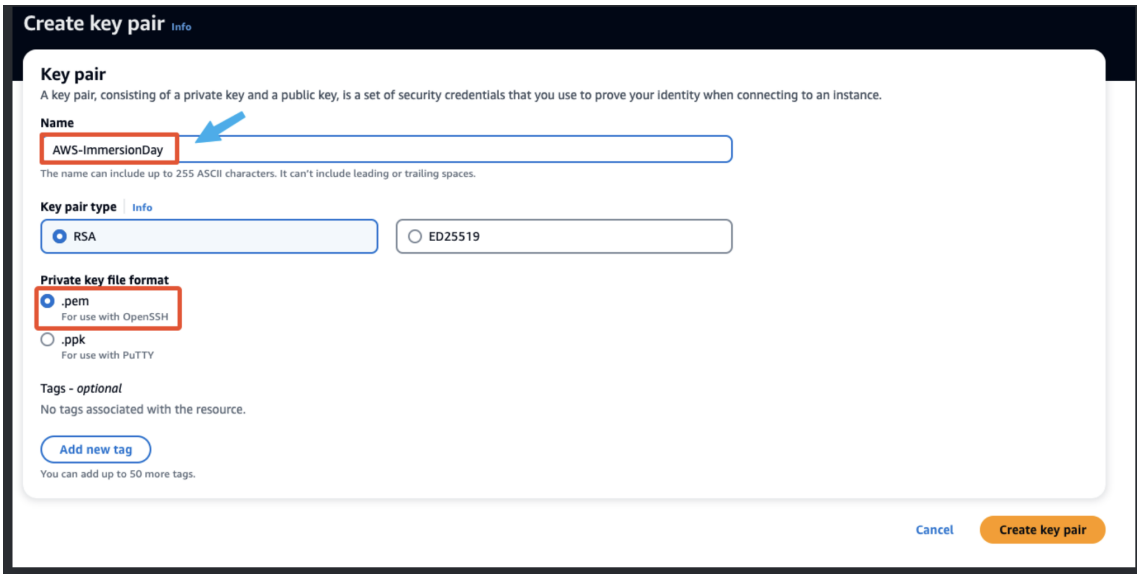
**Create a new Key Pair**

In this lab, you will need to create an EC2 instance using an SSH keypair. The following steps outline creating a unique SSH keypair for you to use in this lab.

1. Sign into the AWS Management Console and open the Amazon EC2 console . In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region.
2. Click on **Key Pairs** in the Network & Security section near the bottom of the leftmost menu. This will display a page to manage your SSH key pairs.




1. To create a new SSH key pair, click the **Create key pair** button at the top of the browser window.




2. Type **[Your Name]-ImmersionDay** into the Key Pair Name: text box and click **Create key pair** button. For Windows users, please select **ppk** for file format.
3. The page will download the file **[Your Name]-ImmersionDay.pem** to the local drive. Follow the browser instructions to save the file to the default download location. Remember the full path to the key pair file you just downloaded.

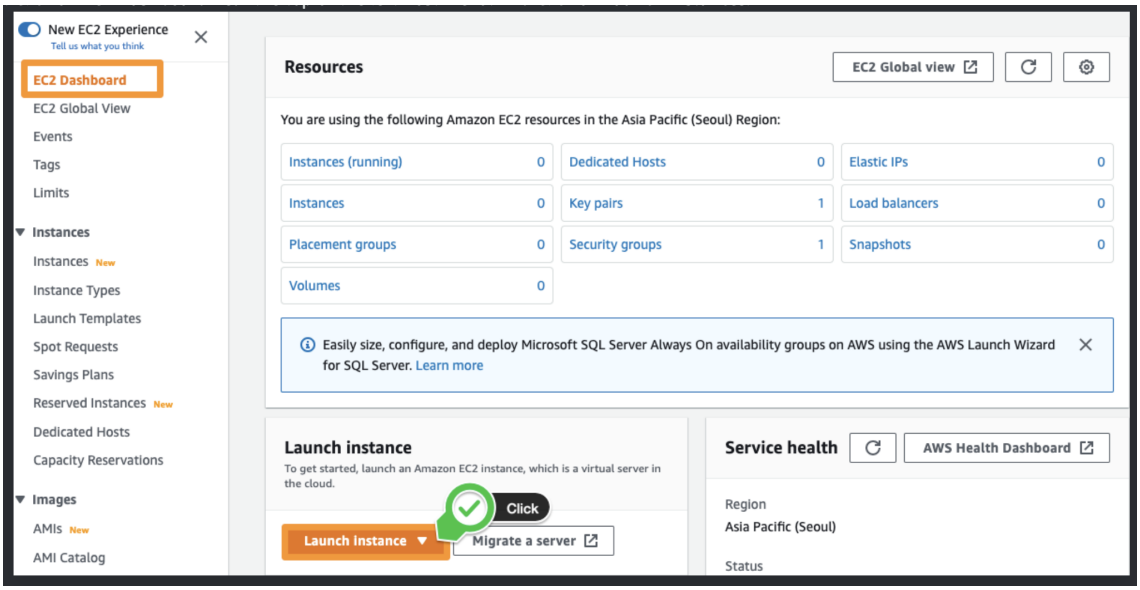
2. Launch a Web Server Instance

# Launch a Web Server Instance

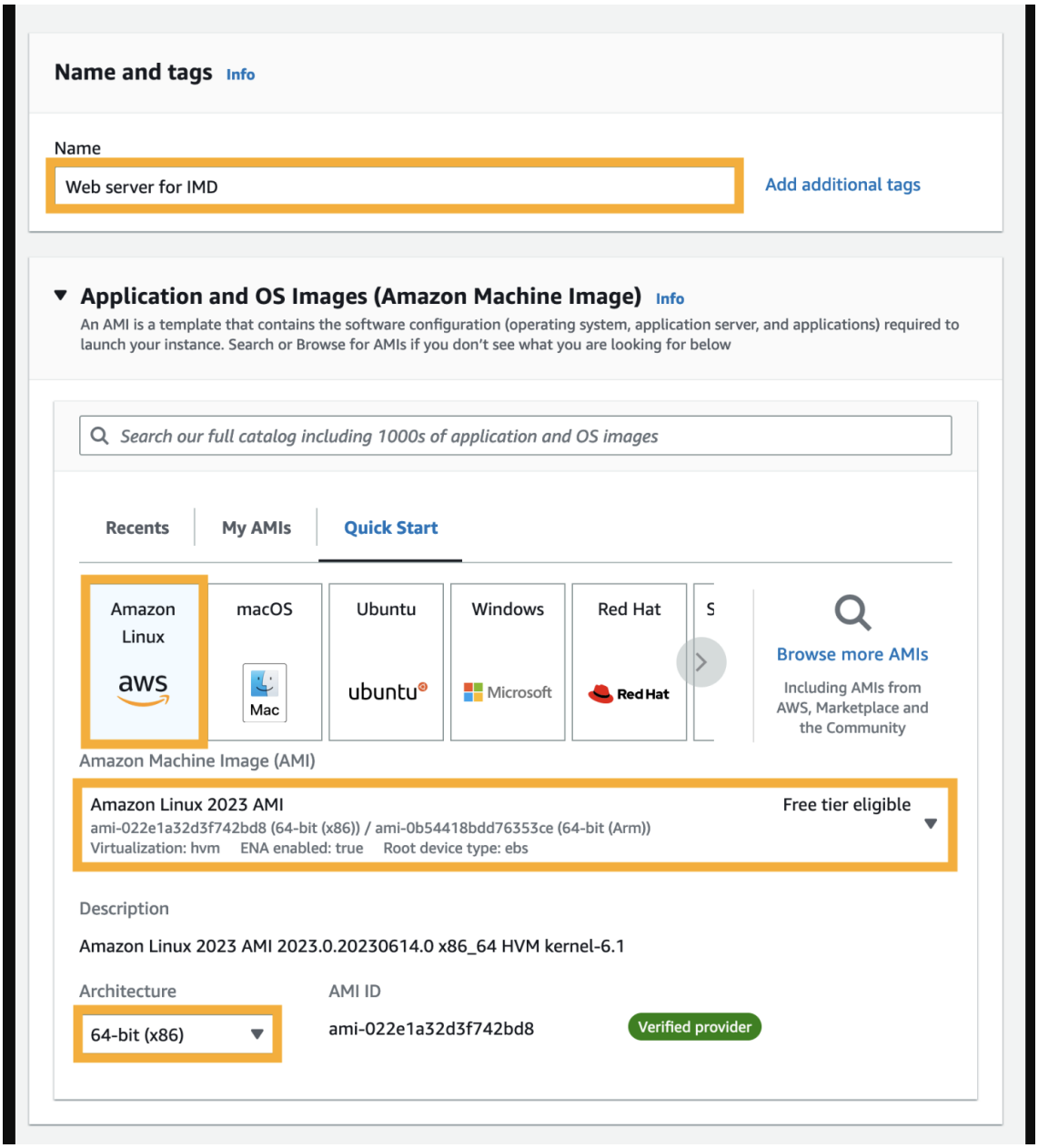
**Launch a Web Server Instance**

We will launch an Amazon Linux 2 instance, bootstrap Apache/PHP, and install a basic web page that will display information about our instance.

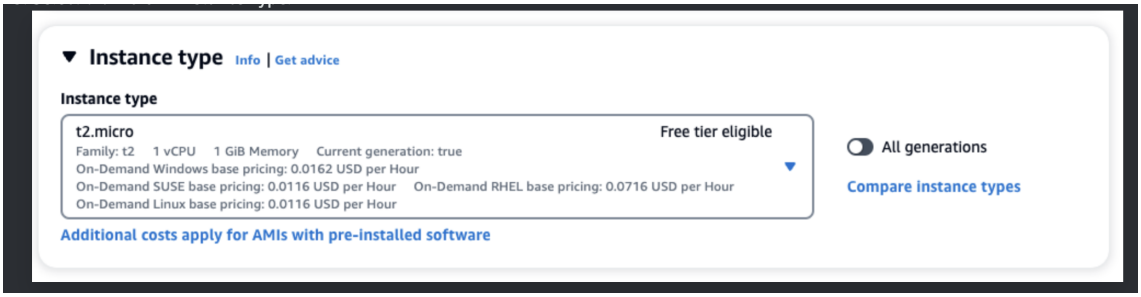
1. Click on **EC2 Dashboard** near the top of the leftmost menu. And Click on **Launch instances**.



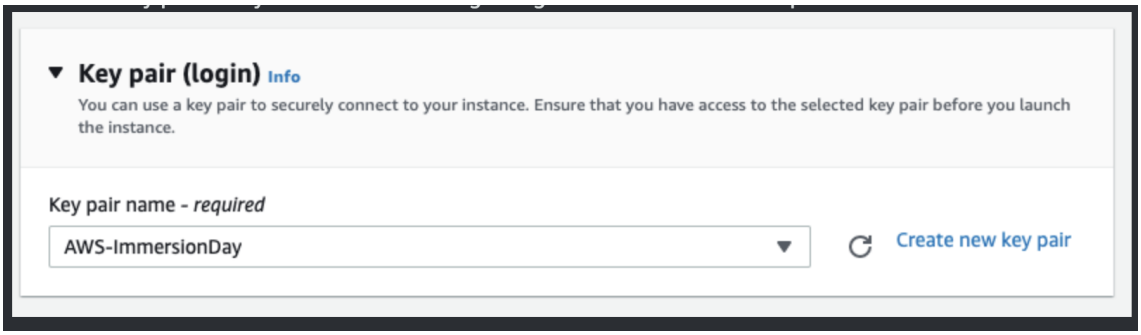

2. In **Name**, put the value **Web server for IMD**. And check the default setting for Amazon Machine Image below.



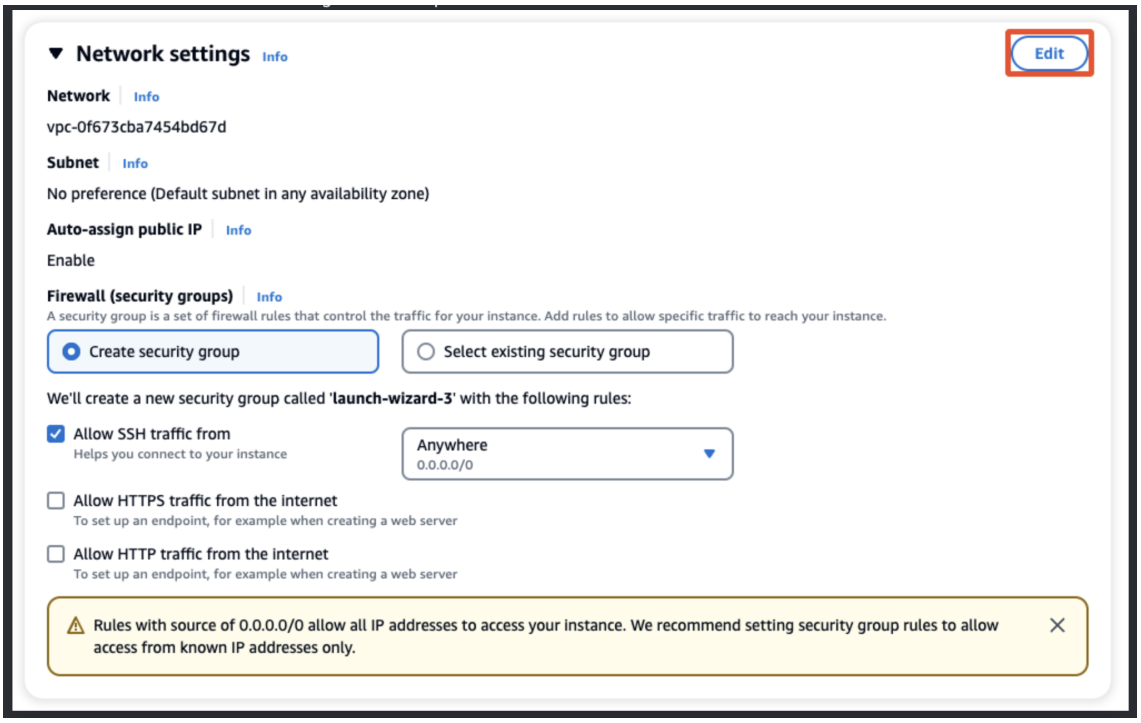

3. Select **t2.micro** in Instance Type.



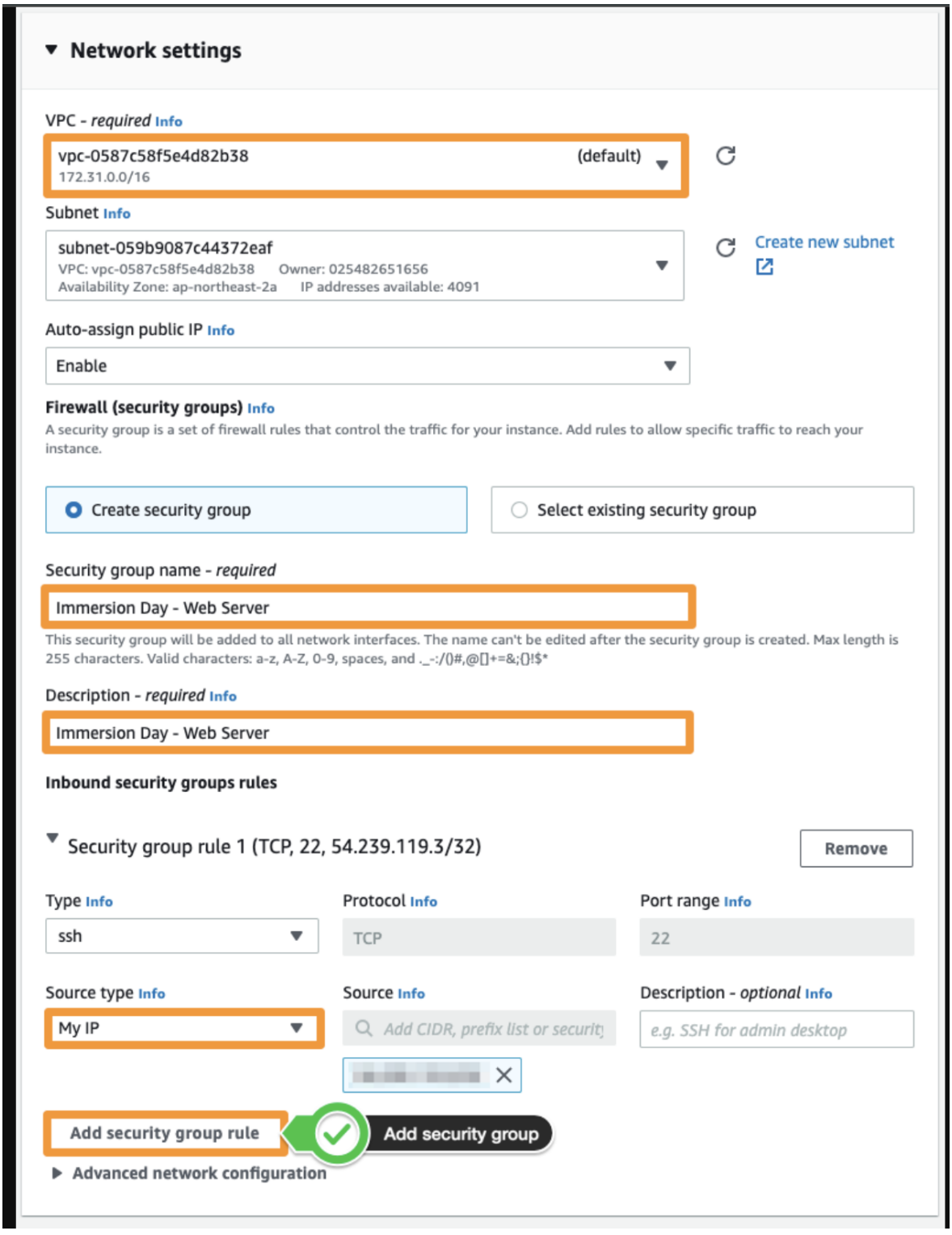

4. Select the key pair that you created in the beginning of this lab from the drop-down.



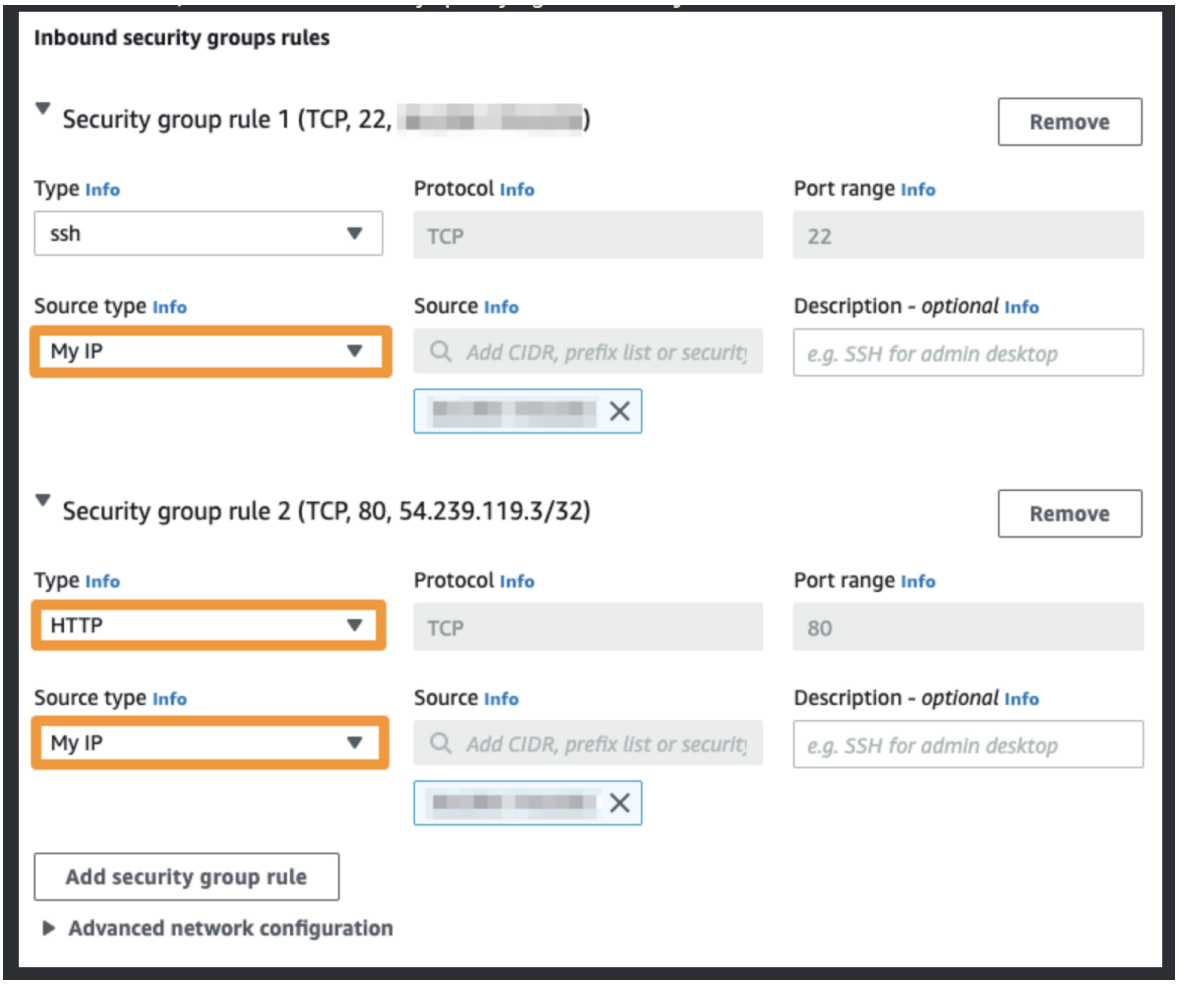

5. Click the Edit button in Network settings to set the space where EC2 will be located.



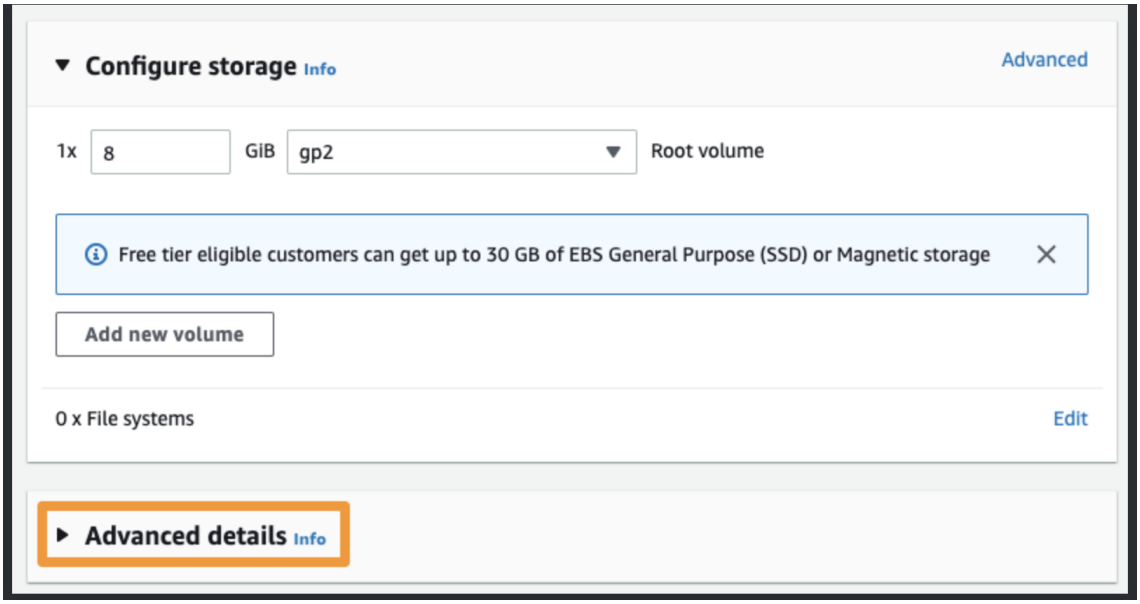

Check **default VPC** and **subnet**. **Auto-assign public IP** is set to **Enable**. Right below it, create **Security groups** to act as a network firewall. Security groups will specify the protocols and addresses you want to allow in your firewall policy. For the security group you are currently creating, this is the rule that applies to the EC2 that will be created. After entering *Immersion Day - Web Server* in Security group name and Description, select Add Security group rule and set HTTP to Type.




Also allow TCP/80 for Web Service by specifying it. Select **My IP** in the source.



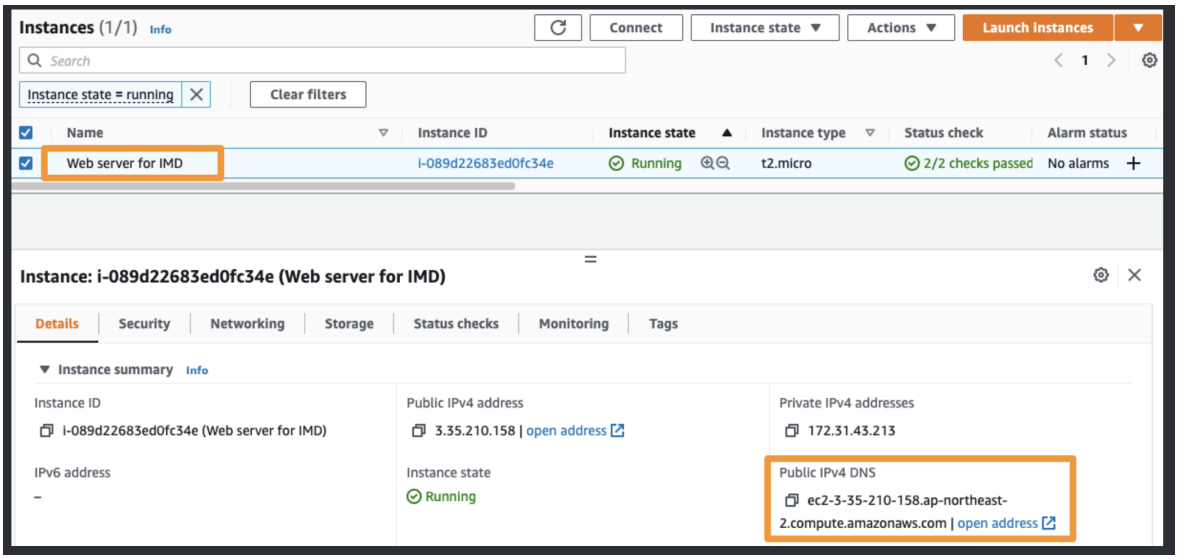

1. All other values accept the default values, expand by clicking on the **Advanced Details** tab at the bottom of the screen.



Click the Meta Data version dropdown and select V2 only (token required) Enter the following values in the User data field and select Launch instance.

```
#!/bin/sh

#Install a LAMP stack
dnf install -y httpd wget php-fpm php-mysqli php-json php php-devel
dnf install -y mariadb105-server
dnf install -y httpd php-mbstring

#Start the web server
chkconfig httpd on
systemctl start httpd

#Install the web pages for our lab
if [ ! -f /var/www/html/immersion-day-app-php7.zip ]; then
cd /var/www/html
wget -O 'immersion-day-app-php7.zip' '
https://static.us-east-1.prod.workshops.aws/9c1aea7d-0009-44d3-875b-c0fa6583c6bd/assets/immersion-day-app-php7.zip?Key-Pair-Id=K36Q2WVO3JP7QD&Policy=eyJTdGF0ZW1lbnQiOlt7IlJlc291cmNlIjoiaHR0cHM6Ly9zdGF0aWMudXMtZWFzdC1HG9csbBXufeimzCdYMNEpD1HUsOnwWh2ALltEAcj714EqDw7swM-tttwIBxv3pVCMq1iZI2BRnZ42LHdE9MGWvuqJ~6HtySrKAIV~ozrnypw4nPXCCAP0RvMbeY7ifCyiPq3Vy4CL7dCUvfFkxz56kNx7HBO8pEbzeUlgSXZlPr6KX65UrrCQHIEpAmKVYfwp3PZqRJlP5eAh2rNqu2NilCw8iibViofIUvZ0Hv6Uv~pHvypp6Pqrce4OLfUEudy8WMvyRjSjBaagHhjYQ__'
unzip immersion-day-app-php7.zip
fi

#Install the AWS SDK for PHP
if [ ! -f /var/www/html/aws.zip ]; then
cd /var/www/html
mkdir vendor
cd vendor
wget
https://docs.aws.amazon.com/aws-sdk-php/v3/download/aws.zip
unzip aws.zip
fi

# Update existing packages

dnf update -y
```

1. Click the **View Instances** button in the lower right hand portion of the screen to view the list of EC2 instances. Once your instance has launched, you will see your Web Server as well as the Availability Zone the instance is in, and the publicly routable **DNS name**. Click the checkbox next to your web server to view details about this EC2 instance.




**Browse the Web Server**

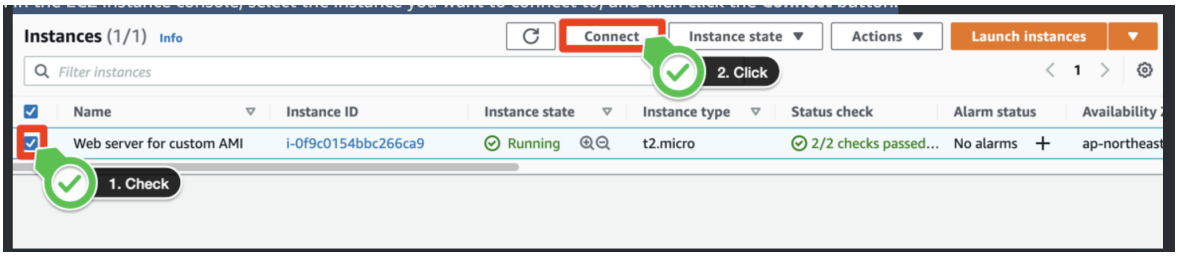
Wait for the instance to pass the Status Checks to finish loading. Open a new browser tab and browse the Web Server by entering the EC2 instance's **Public DNS name** into the browser. The EC2 instance's Public DNS name can be found in the console by reviewing the **Public IPv4 DNS** name line highlighted above. You should see a website that looks like the following.

If you are using the Chrome web browser, when you attach the **Public IPv4 DNS** value to the web browser, if it does not run, https may be automatically added in front of the DNS value, so it may not run. Therefore, it is recommended to enter http://.

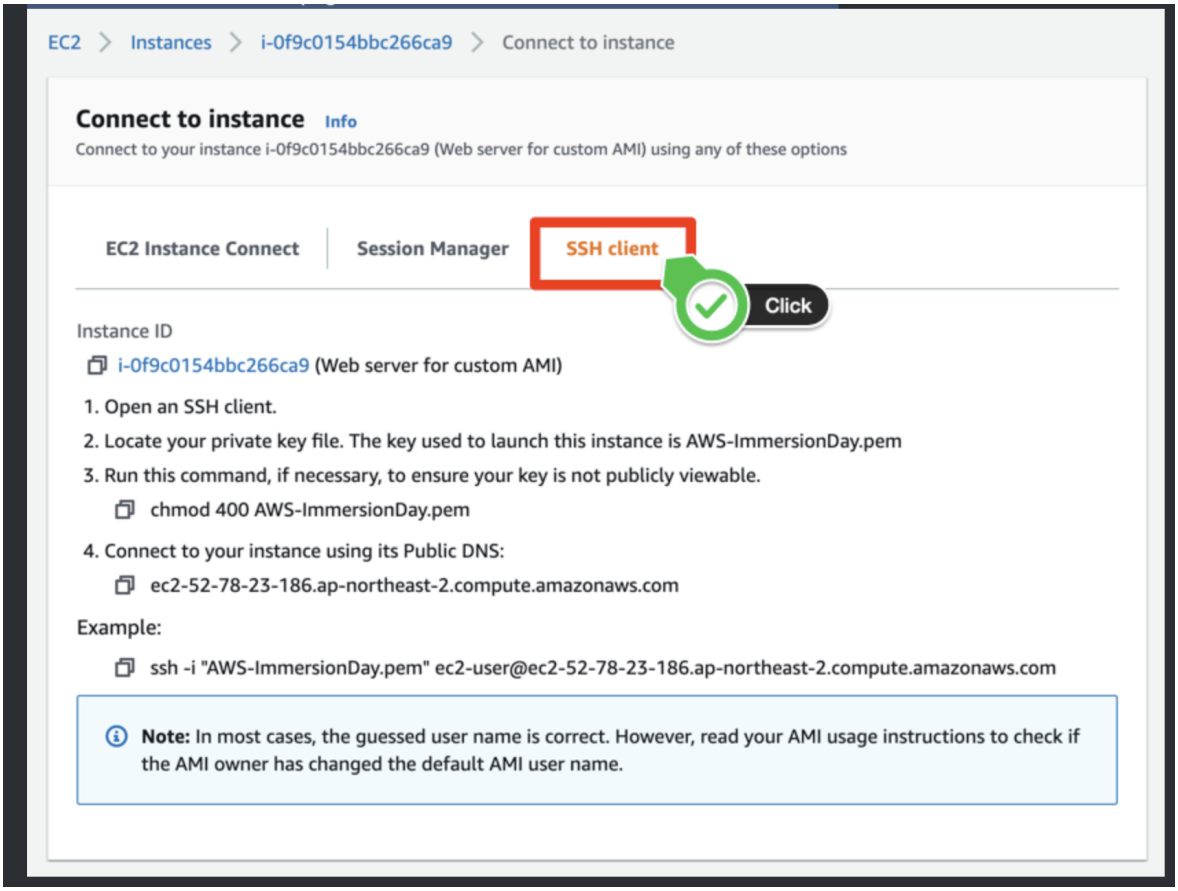

**Great Job! You have deployed a server and launched a web site in a matter of minutes!**

3. Connect to your linux instance

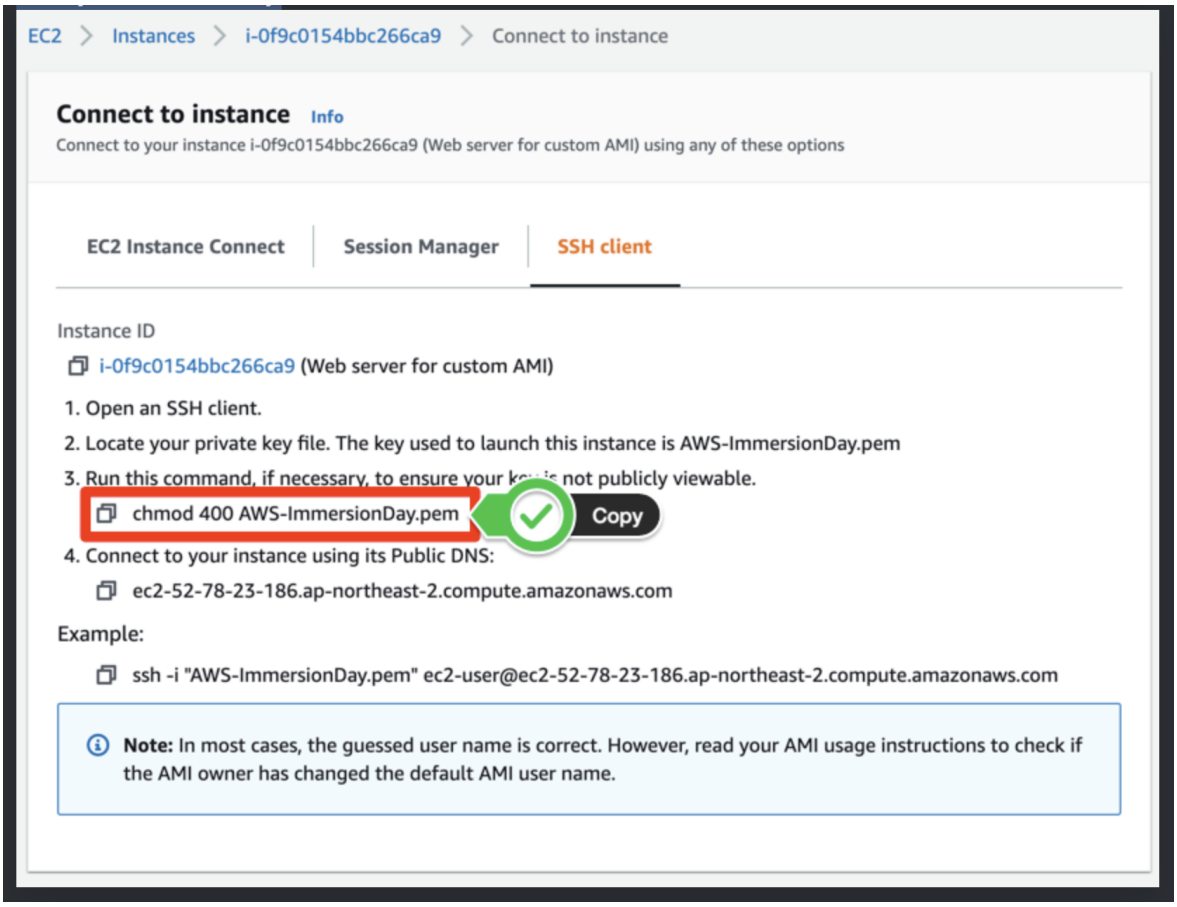
   In the EC2 instance console, select the instance you want to connect to, and then click the **Connect** button.




In the **Connect to instance** page, select **SSH client**. Follow the instructions below.

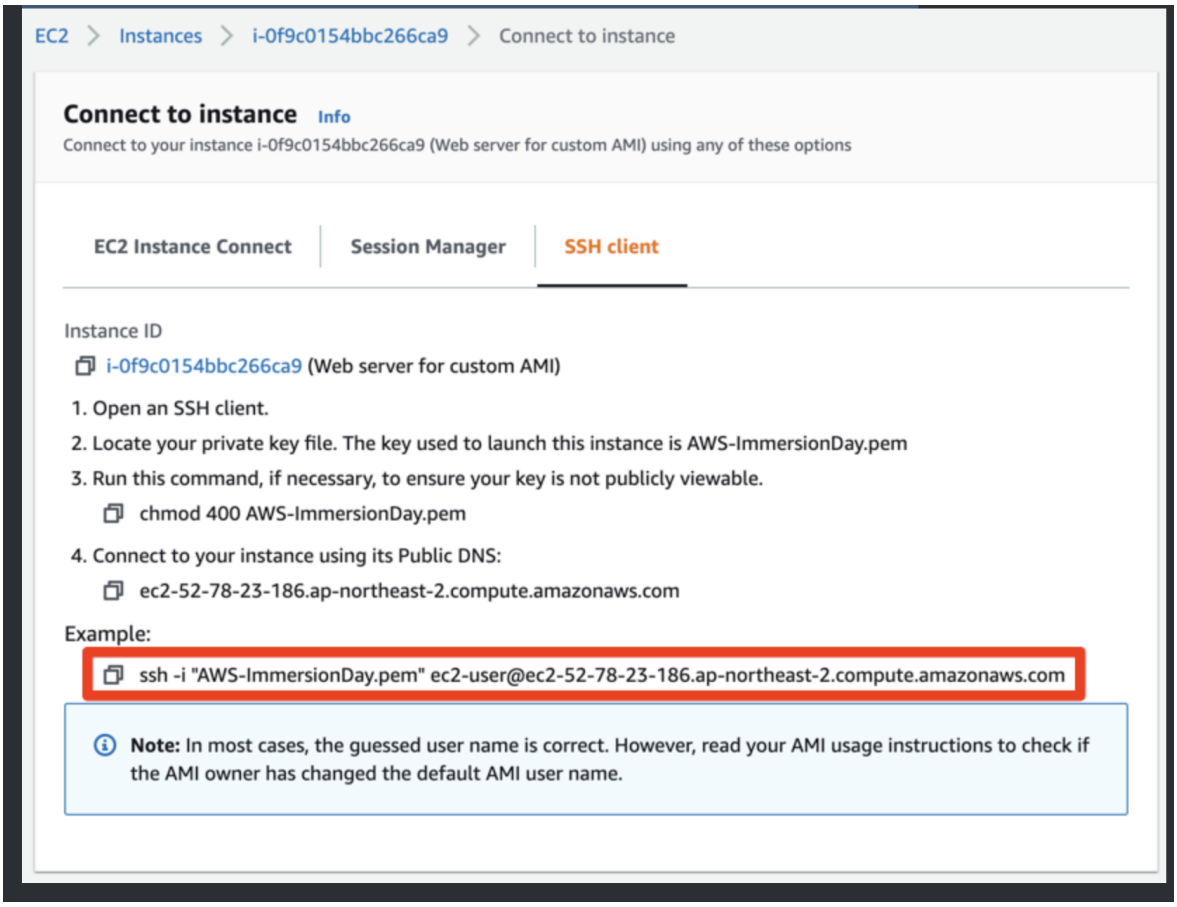



Navigate to where your private key is located and enter the following command. Replace **[Your Name]** with the name you specified prior when you created the key.



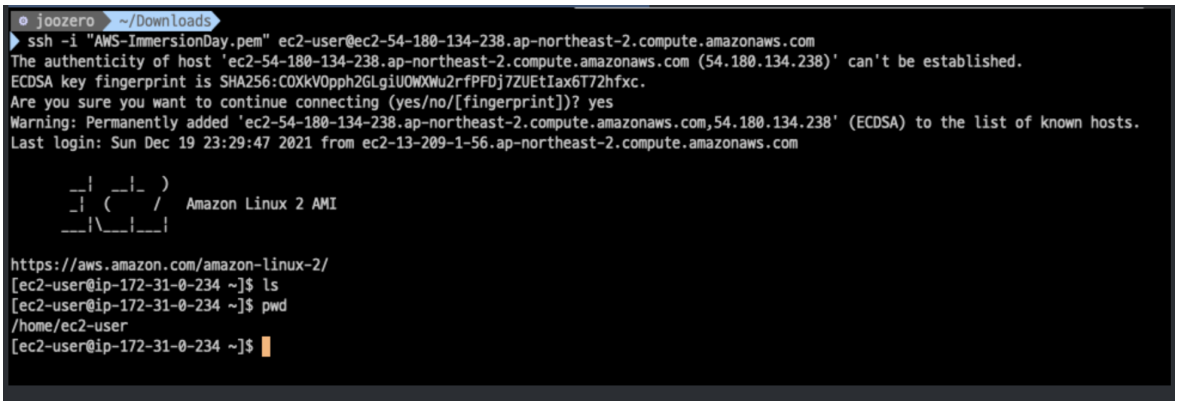

`chmod 400 [Your Name]-ImmersionDay.pem`

Next, enter the following command in your SSH client to connect to your Linux instance.




`ssh -i "[Your Name]-ImmersionDay.pem" ec2-user@<Public IPv4 DNS>`

After connecting continue query to yes, You can check the result as below.



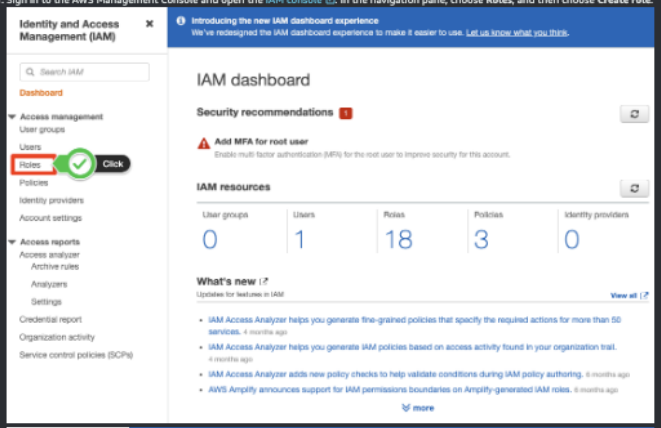

4. Connect to your Linux instance using Session Manager (Optional)

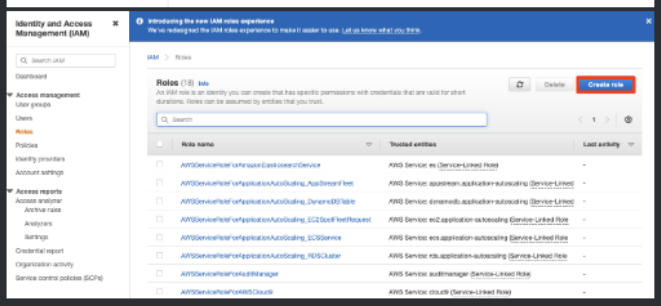
## Connect to your Linux instance using Session Manager (Optional)

Session Manager is a fully managed AWS Systems Manager capability that lets you manage your Amazon EC2 instances through an interactive one-click browser-based shell or through the AWS CLI. You can use Session Manager to start a session with an instance in your account. After the session is started, you can run bash commands as you would through any other connection type.

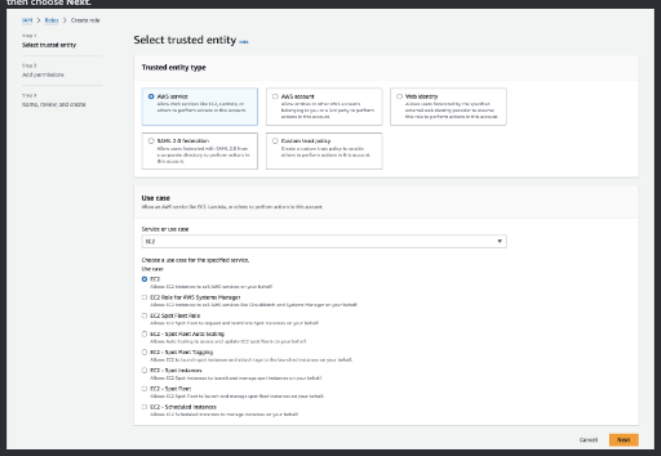
### Create an IAM instance profile for Systems Manager

1. Sign in to the AWS Management Console and open the IAM console. In the navigation pane, choose **Roles**, and then choose **Create role**.

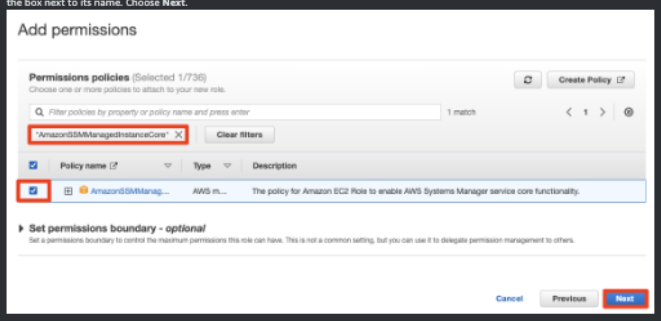



2. Under **Select type of trusted entity**, choose **AWS service**. Immediately under **Choose the service that will use this role**, choose **EC2**, and then choose **Next**.

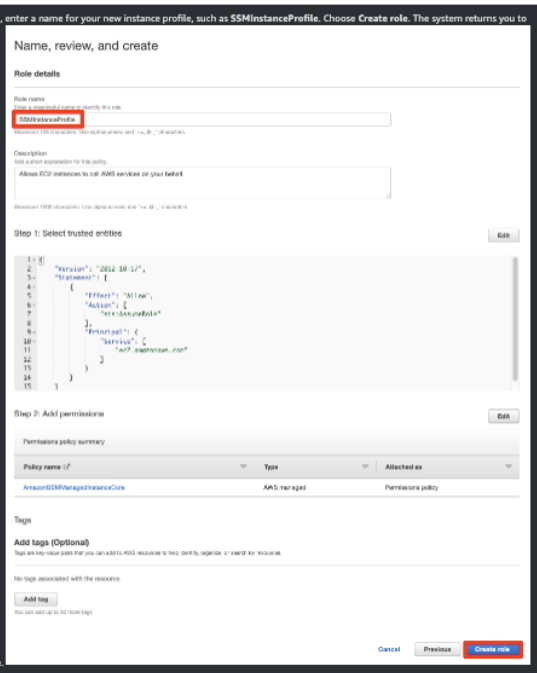



3. On the **Attach permissions policies page**, do the following: Use the **Search** field to locate the **AmazonSSMManagedInstanceCore**. Select the box next to its name. Choose **Next**.




4. For **Role name**, enter a name for your new instance profile, such as **SSMInstanceProfile**. Choose **Create role**. The system returns you to

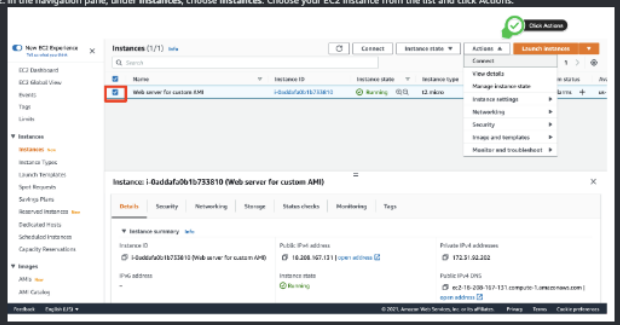
4. For **Role name**, enter a name for your new instance profile, such as **SSMInstanceProfile**. Choose **Create role**. The system returns you to the **Roles** page.



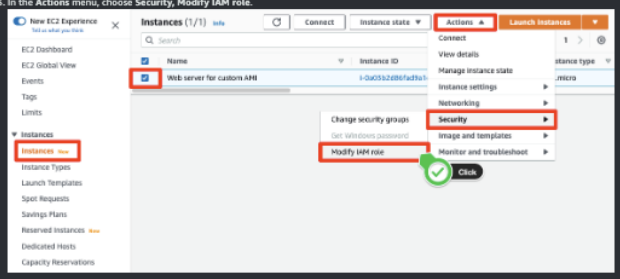

Make a note of the role name. You will choose this role when you create new instances that you want to manage by using Systems Manager.

## Attach the Systems Manager instance profile to an existing instance (console)

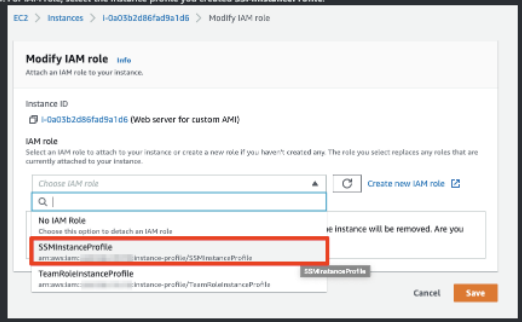
1. Sign in to the AWS Management Console and open the Amazon EC2 console at Amazon EC2 console.
2. In the navigation pane, under **Instances**, choose **Instances**. Choose your EC2 instance from the list and click Actions.



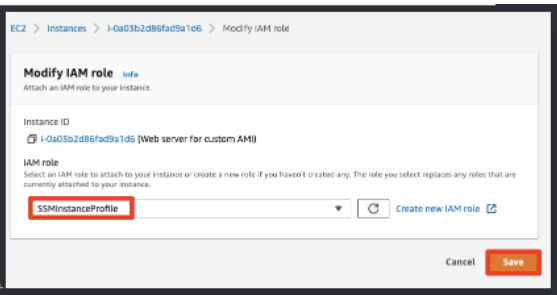

3. In the **Actions** menu, choose **Security, Modify IAM role.**




4. For IAM role, select the instance profile you created **SSMInstanceProfile.**

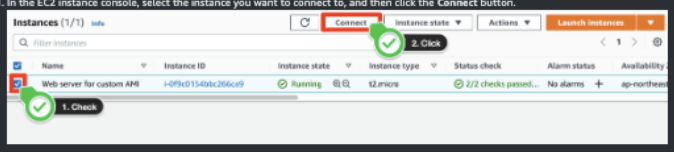


5. Choose **Save**.

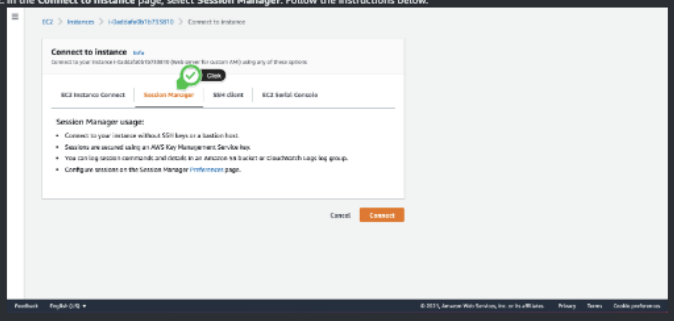



## Connect to your Linux instance using Session Manager

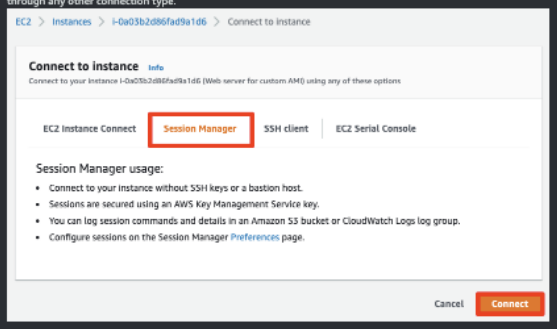
1. In the EC2 instance console, select the instance you want to connect to, and then click the **Connect** button.



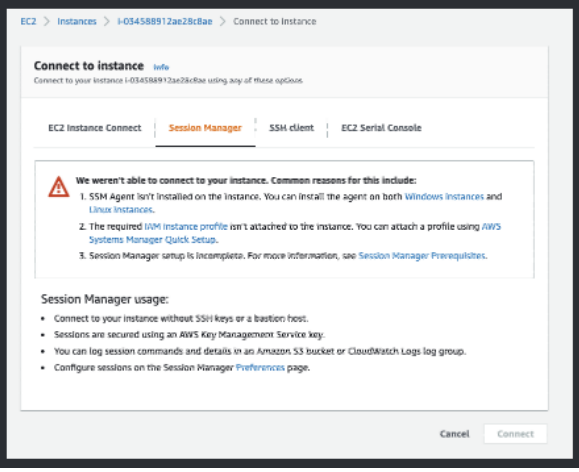

2. In the **Connect to instance** page, select **Session Manager**. Follow the instructions below.



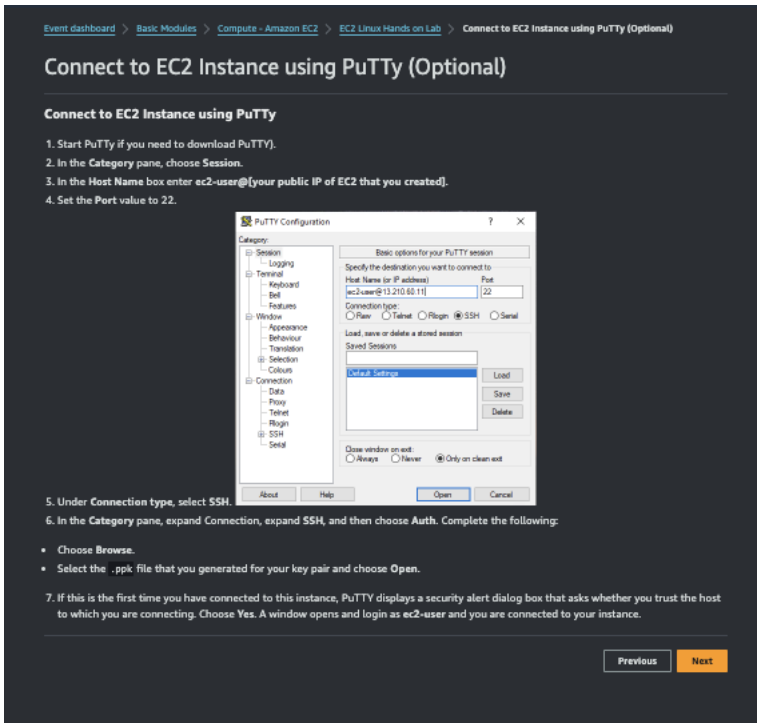

3. Review the **Session Manager usage section** for advantages of using Session Manager.
4. Choose **Connect**. A new session will be started in a new tab. After the session is started, you can run bash commands as you would through any other connection type.

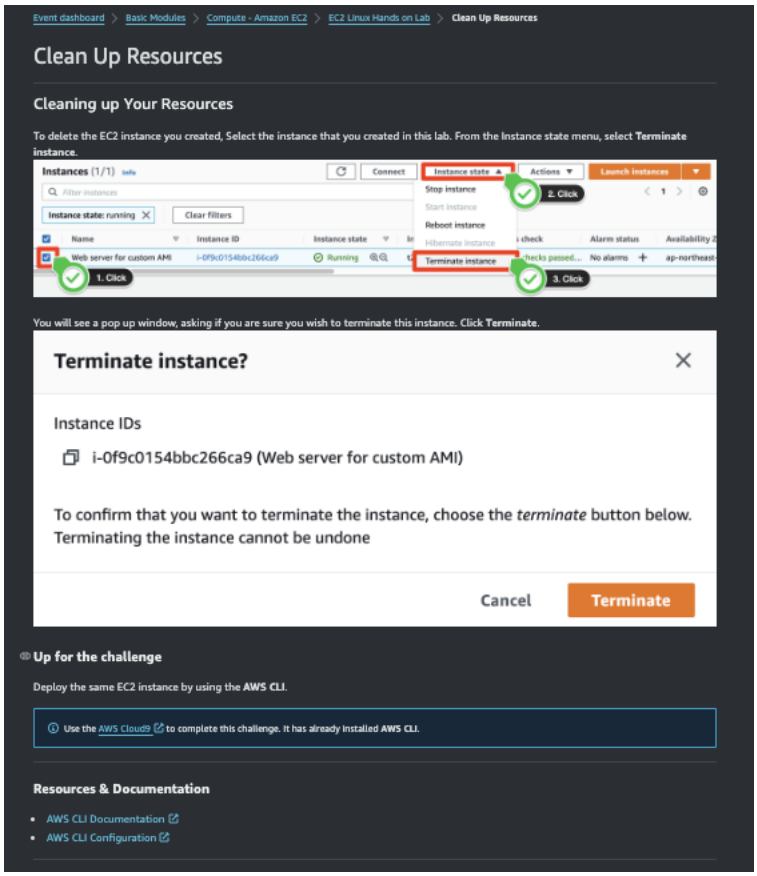



⚠ If you receive an error like shown below, wait for few seconds and refresh your browser. Behind the scenes the EC2 instance is being setup for use with Session Manager



5. Connect to EC2 Instance using PuTTy (Optional)






▼ **Auto Scaling on AWS**

AWS Auto Scaling monitors your applications and automatically adjusts capacity to maintain steady, predictable performance at the lowest possible cost. Using AWS Auto Scaling, it's easy to setup application scaling for multiple resources across multiple services in minutes. The service provides a simple, powerful user interface that lets you build scaling plans for resources including Amazon EC2 instances and Spot Fleets, Amazon ECS tasks, Amazon DynamoDB tables and indexes, and Amazon Aurora Replicas.

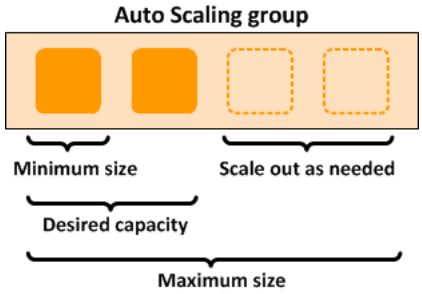
AWS Auto Scaling makes scaling simple with recommendations that allow you to optimize performance, costs, or balance between them. If you're already using Amazon EC2 Auto Scaling to dynamically scale your Amazon EC2 instances, you can now combine it with AWS Auto Scaling to scale additional resources for other AWS services. With AWS Auto Scaling, your applications always have the right resources at the right time.

**EC2 Auto Scaling Lab Overview**

**What is Amazon EC2 Auto Scaling?**

Amazon EC2 Auto Scaling helps you ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application. You create collections of EC2 instances, called *Auto Scaling groups*. You can specify the minimum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes below this size. You can specify the maximum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes above this size. If you specify the desired capacity, either when you create the group or at any time thereafter, Amazon EC2 Auto Scaling ensures that your group has this many instances. If you specify scaling policies, then Amazon EC2 Auto Scaling can launch or terminate instances as demand on your application increases or decreases.

For example, the following Auto Scaling group has a minimum size of one instance, a desired capacity of two instances, and a maximum size of four instances. The scaling policies that you define adjust the number of instances, within your minimum and maximum number of instances, based on the criteria that you specify.




**Features of Amazon EC2 Auto Scaling**

With Amazon EC2 Auto Scaling, your EC2 instances are organized into Auto Scaling groups so that they can be treated as a logical unit for the purposes of scaling and management. Auto Scaling groups use launch templates (or launch configurations) as configuration templates for their EC2 instances.

The following are key features of Amazon EC2 Auto Scaling:

**Monitoring the health of running instances** Amazon EC2 Auto Scaling automatically monitors the health and availability of your instances using EC2 health checks and replaces terminated or impaired instances to maintain your desired capacity.

**Custom health checks** In addition to the built-in health checks, you can define custom health checks that are specific to your application to verify that it's responding as expected. If an instance fails your custom health check, it's automatically replaced to maintain your desired capacity.

**Balancing capacity across Availability Zones** You can specify multiple Availability Zones for your Auto Scaling group, and Amazon EC2 Auto Scaling balances your instances evenly across the Availability Zones as the group scales. This provides high availability and resiliency by protecting your applications from failures in a single location.

**Multiple instance types and purchase options** Within a single Auto Scaling group, you can launch multiple instance types and purchase options (Spot and On-Demand Instances), allowing you to optimize costs through Spot Instance usage. You can also take advantage of Reserved Instance and Savings Plan discounts by using them in conjunction with On-Demand Instances in the group.

**Automated replacement of Spot Instances** If your group includes Spot Instances, Amazon EC2 Auto Scaling can automatically request replacement Spot capacity if your Spot Instances are interrupted. Through Capacity Rebalancing, Amazon EC2 Auto Scaling can also monitor and proactively replace your Spot Instances that are at an elevated risk of interruption.

**Load balancing** You can use Elastic Load Balancing or VPC Lattice load balancing and health checks to ensure an even distribution of application traffic to your healthy instances. Whenever instances are launched or terminated, Amazon EC2 Auto Scaling automatically registers and deregisters the instances from the load balancer.**Scalability**Amazon EC2 Auto Scaling also provides several ways for you to scale your Auto Scaling groups. Using auto scaling allows you to maintain application availability and reduce costs by adding capacity to handle peak loads and removing capacity when demand is lower. You can also manually adjust the size of your Auto Scaling group as needed.

**Instance refresh** The instance refresh feature provides a mechanism to update instances in a rolling fashion when you update your AMI or launch template. You can also use a phased approach, known as a canary deployment, to test a new AMI or launch template on a small set of instances before rolling it out to the whole group.

**Lifecycle hooks** Lifecycle hooks are useful for defining custom actions that are invoked as new instances launch or before instances are terminated. This feature is particularly useful for building event-driven architectures, but it also helps you manage instances through their lifecycle.

**Support for stateful workloads** Lifecycle hooks also offer a mechanism for persisting state on shut down. To ensure continuity for stateful applications, you can also use scale-in protection or custom termination policies to prevent instances with long-running processes from terminating early.

For more information about the benefits of Amazon EC2 Auto Scaling, see Amazon EC2 Auto Scaling benefits.

**Work with Auto Scaling groups**

You can create, access, and manage your Auto Scaling groups using any of the following interfaces:

- **AWS Management Console** – Provides a web interface that you can use to access your Auto Scaling groups. If you've signed up for an AWS account, you can access your Auto Scaling groups by signing into the AWS Management Console, using the search box on the navigation bar to search for **Auto Scaling groups**, and then choosing **Auto Scaling groups**.
- **AWS Command Line Interface (AWS CLI)** – Provides commands for a broad set of AWS services, and is supported on Windows, macOS, and Linux. To get started, see Prepare to use the AWS CLI. For more information, see autoscaling in the *AWS CLI Command Reference*.
- **AWS Tools for Windows PowerShell** – Provides commands for a broad set of AWS products for those who script in the PowerShell environment. To get started, see the AWS Tools for Windows PowerShell User Guide. For more information, see the AWS Tools for PowerShell Cmdlet Reference.
- **AWS SDKs** – Provides language-specific API operations and takes care of many of the connection details, such as calculating signatures, handling request retries, and handling errors. For more information, see AWS SDKs.
- **Query API** – Provides low-level API actions that you call using HTTPS requests. Using the Query API is the most direct way to access AWS services. However, it requires your application to handle low-level details such as generating the hash to sign the request, and handling errors. For more information, see the Amazon EC2 Auto Scaling API Reference.
- **AWS CloudFormation** – Supports creating Auto Scaling groups using CloudFormation templates. For more information, see Create Auto Scaling groups with AWS CloudFormation.

To connect programmatically to an AWS service, you use an endpoint. For information about endpoints for calls to Amazon EC2 Auto Scaling, see Amazon EC2 Auto Scaling endpoints and quotas in the *AWS General Reference*.

An overview of auto scaling on AWS can be found here. This lab will first walk you through the process of creating an Amazon Machine Image (AMI) from a web host created with CloudFormation. Then it will walk you through creating a launch template and setting up the web host within an auto scaling group behind an Appliction Load Balancer (ALB). The end result will be an auto scaling group behind a load balancer that scales based on CPU utilization of the hosts.

**Services and concepts covered in this lab:**

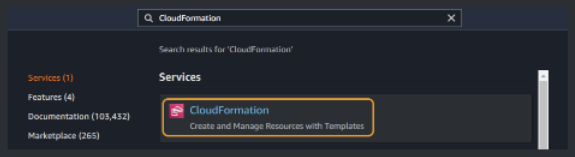
- Primary: Auto Scaling, Launch Templates, Creation and configuration of Security Groups
- Secondary: AMIs, Application Load Balancers

## Lab Prerequisites

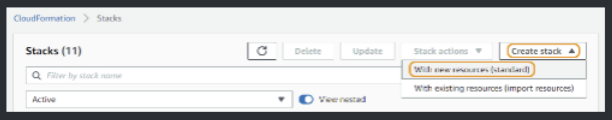
**To create an AMI (Amazon Machine Image) for our Auto-Scaling group we will first need to setup a web host. We will generate an AMI from the instance and then auto-scale the instance behind a load balancer. Click on the template link button below to build the web host in EC2 using CloudFormation.**

### Download and launch the CloudFormation template

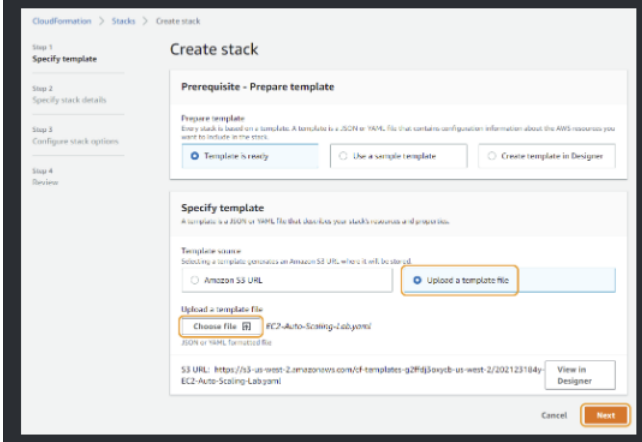
1. Download the "EC2-Auto-Scaling-Lab.yaml" CloudFormation template by **right-clicking** on this link and save it to your local hard drive.
2. In the AWS Console search for `CloudFormation` or select the **Services** menu and click on **CloudFormation** under "Management & Governance".




3. In the CloudFormation console select the **Create stack** button and then select **With new resources (standard)**.




4. Under "Template source" select **Upload a template file** and then select the **Choose file** button. Select the "EC2-Auto-Scaling-Lab.yaml" template file you downloaded in the first step. Once you have selected the template file click on the **Next** button.




5. On the Specify stack details page, fill in the following fields:

a. Under "Stack name" name your stack `[Your Initials]-EC2-Web-Host` -

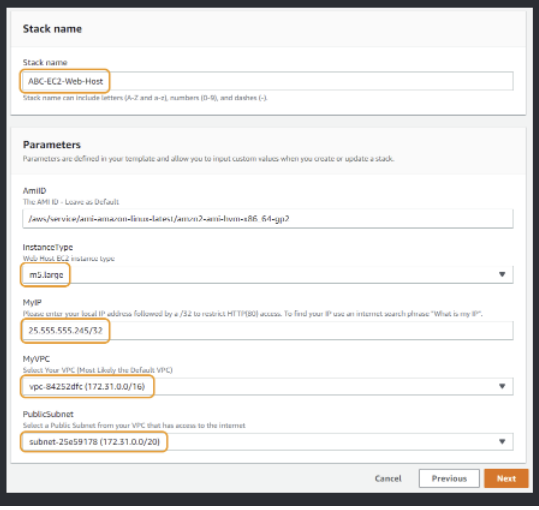
b. You can leave "AmiID" as the default, resulting in the use the most recent version of this AMI.

c. Under "InstanceType" **select** the **m5.large** or the t2.micro. It is recommended that you use m5.large size instances to demonstrate real world performance as t2 types instances are not recommended for production workloads. If you have an issue with the m5 instances for any reason, load the CloudFormation template again and select the t2.micro, it is sufficient enough for this lab.

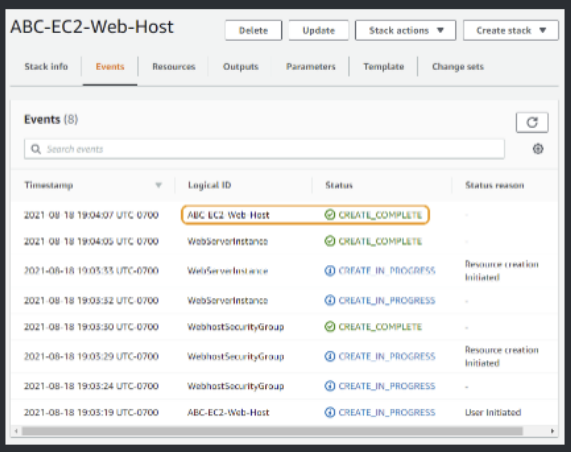
d. Under "MyIP" **input** the IP address of your local machine followed by a `/32` . This will lock down HTTP port 80 to your machine. You can find you local IP by searching What is my IP.

e. Under "MyVPC" **select** the VPC you want to use to setup the instance. In most accounts the default VPC will be a good choice and in new AWS accounts it will be the only choice.

f. Under "PublicSubnet" **select** a subnet within your VPC that has internet access. A public subnet is defined by a subnet having a route to the internet gateway within it's route table. By default, the Default VPC subnets are all public.



6. Once you are done entering the details above, click on **Next**. On the next page, "Configure stack options", you can leave "Tags", "Permissions", and "Advanced options" as default and select **Next**.
7. On the Review [Your Initials]-EC2-Web-Host page, review your settings and click on **Create stack** to start building your web server.
8. Wait till the "Logical ID" "[Your Initials]-EC2-Web-Host" shows a status of "CREATE_COMPLETE".

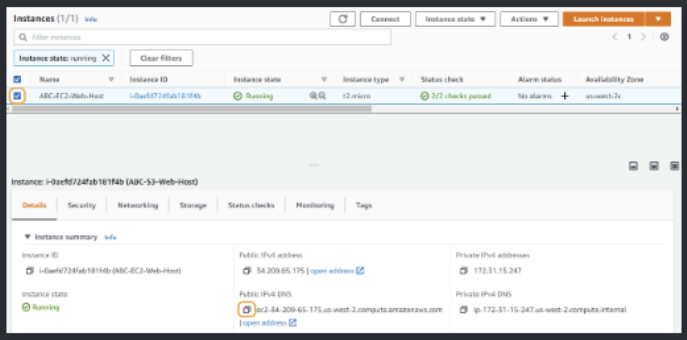



The CloudFormation stack creation should be completed in about 3 minutes.

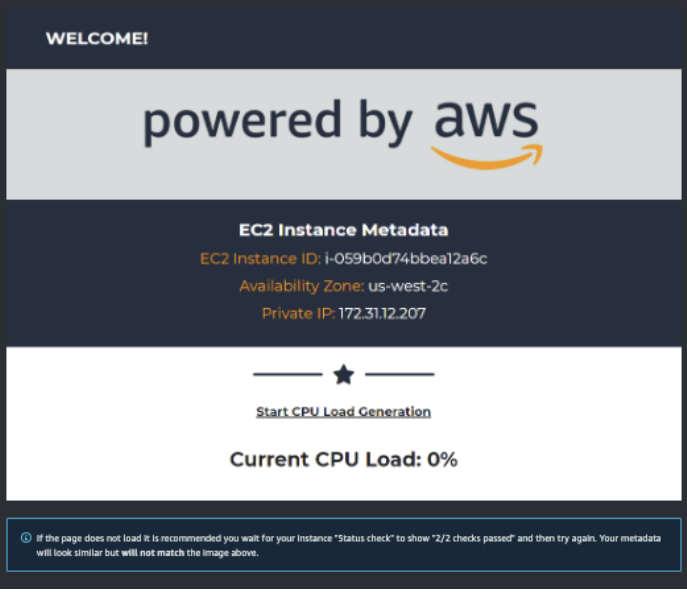
## Confirm the successful setup of your instance:

1. Navigate to the EC2 service page by selecting **Services** and then **EC2** or search for EC2 in the search bar.
2. Select **Instances** from the left hand menu. On the "Instances" page, select your instance "[Your Initials]-EC2-Web-Host" and copy the "Public IPv4 DNS" address into your clipboard by clicking on the image of two overlapping squares to the left of the Public IPv4 DNS Address. Paste this address into a new tab on your web browser.

Clicking on the open address link under "Public IPv4 DNS" heading may result in you not being able to see your website. The "open address" link uses https:// instead of http://, which will result with an error because our web host has not been setup with an SSL cert.




3. You should now see the from page titled "EC2 Instance Metadata".

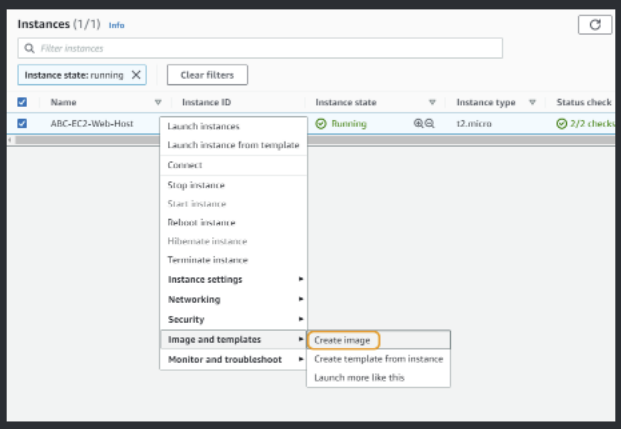



If the page does not load it is recommended you wait for your Instance "Status check" to show "2/2 checks passed" and then try again. Your metadata will look similar but **will not match** the image above.

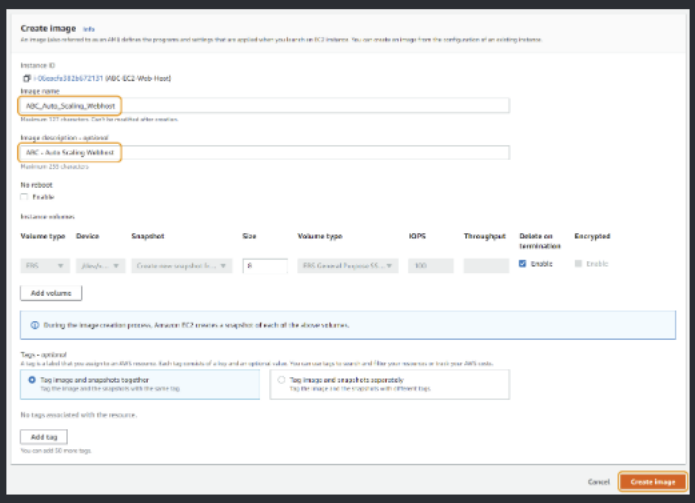
## Generate a custom AMI of the web server created in the EC2 - Linux Lab:

Now that we have our instance setup to host our website, we will generate a custom machine image for our auto scaling group. This will create an image of our web host that will be used by our Auto Scaling group to spin up multiple instances based on server load.

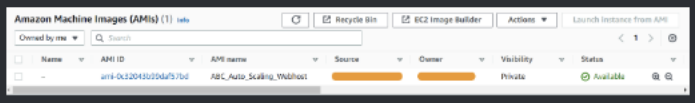
1. In the EC2 Console under Instances, you can create Amazon Machine Images (AMIs) from either running or stopped instances. Return to or open the EC2 console.
2. **Right-click** your webhost instance named "[Your Initials] - Web Server" and under "Image and templates" choose **Create image** from the context menu. (This can also be done by selecting the instance and clicking on the Actions menu in the upper right hand corner)




3. On the "Create Image" page, put in the Image name `[Your Initials]_Auto_Scaling_Webhost` and a description. You can leave the Instance volumes as default and then choose **Create Image**.




4. It may take a few minutes for the AMI to be created. In the EC2 console under "Images" in the left hand menu select **AMIs**. You should see the AMI you just created, it may be in a pending state, but after a few moments it will transition to an available state.

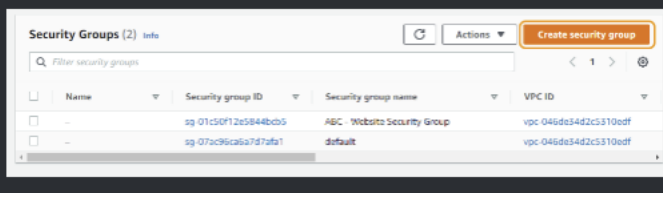



We are done with our new Amazon Machine Image for now, we can now move on to setting up our auto scaling security group.

## Create a new Security Group for our Auto Scaling Group:

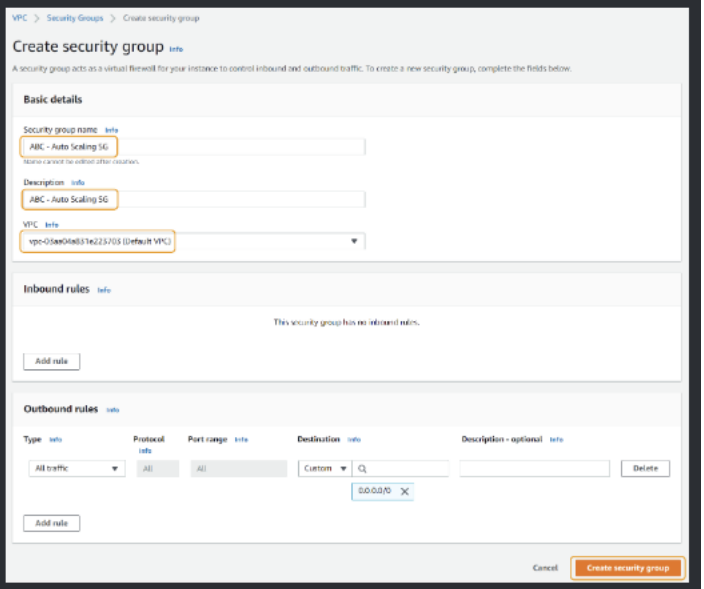
Before we get into setting up our Launch Template we will need to setup a special Security Group for our Auto Scaling Group. A security group provides instance (virtual machine) level protection. If you want to know more about security groups, check out this page.

1. Within the console Under "Services" select **EC2** or search `EC2` in the search bar. On the EC2 page under the "Network & Security" heading in the left-hand menu select **Security Groups**. You should see other security groups, including the security group for your web server named "[Your Initials]-EC2-Web-Host - Website Security Group". To start the creation of a new security group, click on the **Create security group** button.



2. Name your security group `[Your Initials] - Auto Scaling SG` and you can use the same name for the description as well and make sure you have the correct VPC select. (Most likely the **Default** VPC unless you setup a new one for this lab)
3. Under "Inbound rules" there currently are not any rules created, we will leave it empty for now. We will be creating a rule later in this lab but we need our load balancer security group to exist first.
4. "Outbound rules" currently allow all traffic out so there is no need for any additional rules.

Now click on **Create security group.**

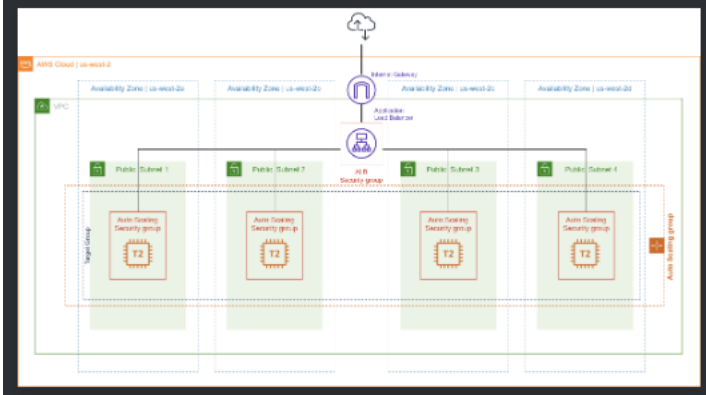



**You are now finished with the prerequisites and ready to move onto the next step**

# Creating a Launch Template

## Auto Scaling Lab Architecture:

Below is a diagram of our end state architecture. Let's get building!



## There are three main components to EC2 Auto Scaling on AWS

**1. Launch Template:** A Launch Template is a feature of EC2 Auto Scaling that allows a way to templatize your launch requests. It enables you to store launch parameters so that you do not have to specify them every time you launch an instance. For example, a launch template can contain a specific Amazon Machine Image, instance type, storage, and networking settings that you typically use to launch instances. For each Launch Template, you can create one or more numbered Launch Template Versions. Each version can have different launch parameters.

**2. Auto Scaling Groups:** For auto scaling your EC2 instances are organized into groups so that they can be treated as a logical unit for the purposes of scaling and management. When you create a group, you can specify its minimum, maximum, and desired number of EC2 instances.

**3. Scaling Policies:** A Scaling Policy tells Auto Scaling when and how to scale. Scaling can occur manually, on a schedule, on demand, or you can use Auto Scaling to maintain a specific number of instances.

Auto Scaling is well suited for applications that have unpredictable demand patterns that can experience hourly, daily, or weekly variability in usage. This helps you to manage your cost and eliminate over-provisioning of capacity during times when it is not needed. Auto Scaling can also find an unhealthy instance, terminate that instance, and launch a new one based on the scaling plan.

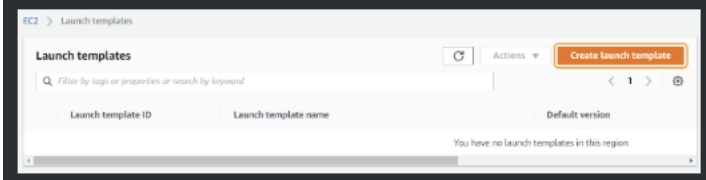
The number of EC2 instances can be scaled in or out as Auto Scaling responds to the metrics you define when creating these groups.

- You can specify the minimum number of instances in each Auto Scaling Group, so that your group never goes below this size. (Even if the instances are determined to be unhealthy)
- You can specify the maximum number of instances in each Auto Scaling Group, so that your group never goes above this size.
- You can specify a desired capacity to specify the number of healthy instances your auto scaling group should have at all times. More information can be found here
- You can specify scaling policies so that Auto Scaling will modify the desired target capacity mentioned in the previous point. It will launch or terminate instances as demand on your application increases or decreases.

## Creating a Launch Template

When you create an Auto Scaling Group, you must specify a Launch Template. The first step in this lab is to create the Launch Template for an EC2 Auto Scaling Group.

1. Under the "Services" select **EC2**.
2. In the left navigation pane, find "Instances" and select **Launch Templates**.
3. Now select **Create launch template**.




4. This takes you to the "Create launch template" page, starting with the "Launch template name and description":

   a. *Launch template name*: `[Your Initials]-scaling-template`

   b. *Template version description*: This is optional

   c. *Auto scaling guidance*: **Check the box** to provide guidance

5. "Launch Template Contents" defines the parameters for the instances in the Auto Scaling group:

   a. *Amazon machine image (AMI)*: Select "My AMIs" and "Owned by me". In the drop down search by typing your initials and select the custom AMI you just created **[Your Initials]_Auto_Scaling_Webhost**. (The new AMI you just created may already be selected)

   b. *Instance type*: **t2.micro**

   c. *Key pair (Login)*: **Select** the Key Pair you created in the first lab, it is most likely call [Your Initials]-KeyPair

   d. *Networking Settings*:

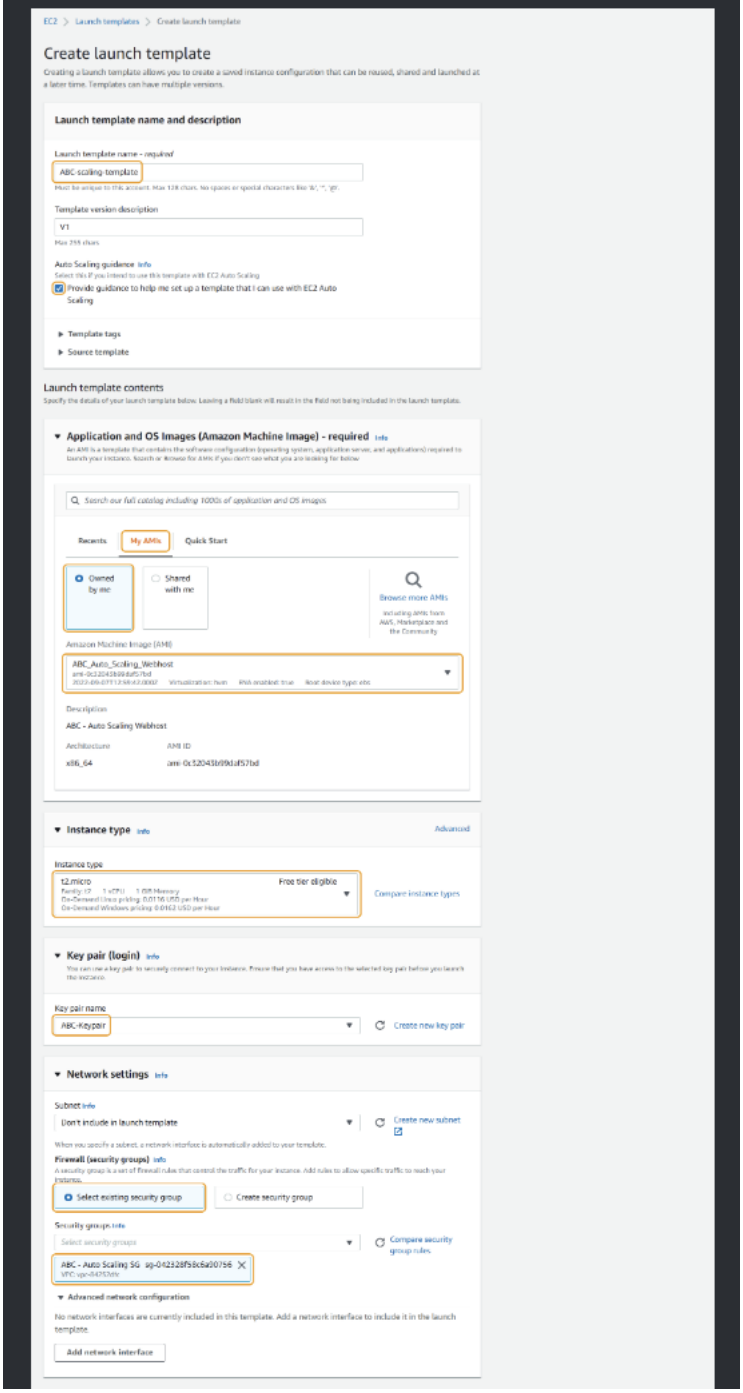
   - Subnet: **Don't include in launch template**
   - Firewall (security groups): Select "Select existing security group" and then select the security group you created in the first part of this lab named **[Your Initials] - Auto Scaling SG**

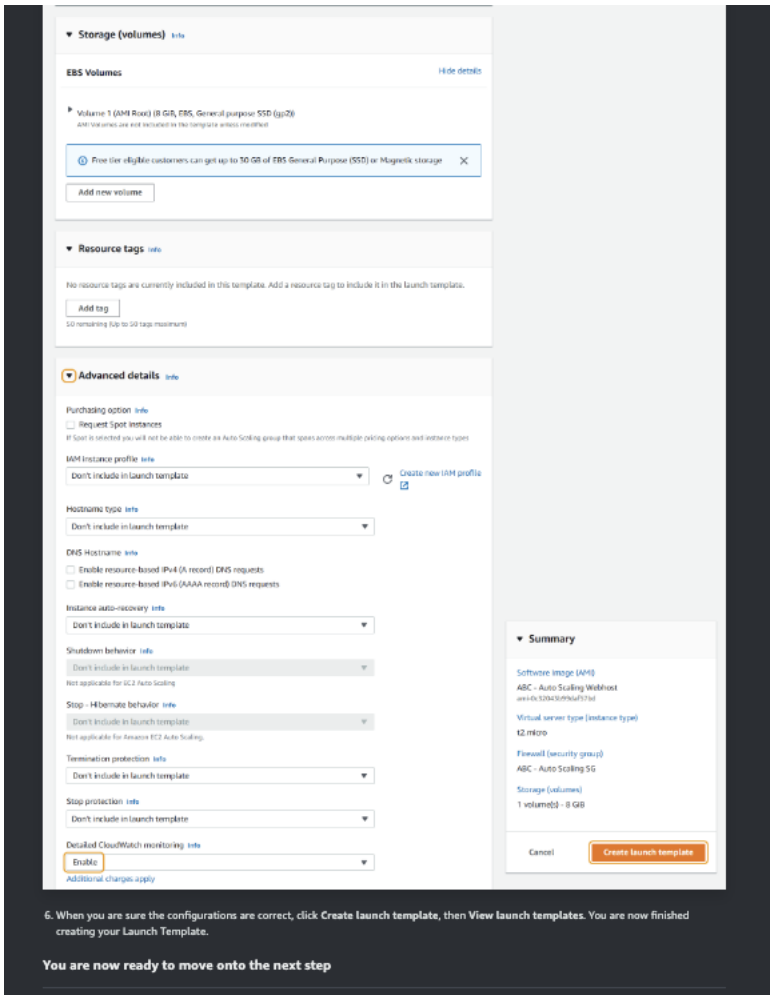
   e. *Configure storage*: Leave as default

   f. *Resource tags*: None

   g. *Advanced Details*: **IMPORTANT**: Select the arrow to expand "Advanced details" and under "Detailed Cloudwatch monitoring" select **Enable**. Leave everything else as the default.

Here, you are enabling CloudWatch Detailed monitoring. By default, your instance has basic monitoring in 5-minute intervals for the instances. After you enable detailed monitoring CloudWatch will monitor the instances in your auto scaling group in 1-minute intervals. This will allow the auto scaling group to respond quicker to changes in the group.

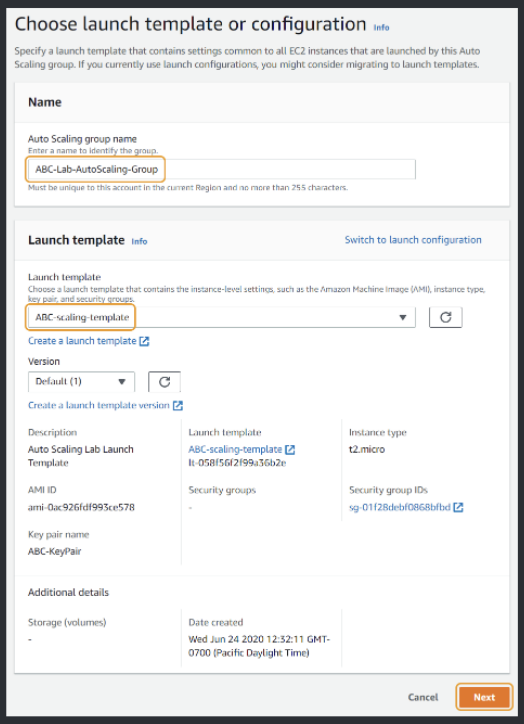





**Create Auto Scaling Group**

You have created a Launch Template, which defines the parameters of the instances launched. Now we will create an Auto Scaling Group so that you can define how many EC2 instances should be launched and where to launch them.

1. Make sure you are on the EC2 Service page.
2. In the left navigation pane, find "Auto Scaling" and Select **Auto Scaling Groups**.
3. Click **Create an Auto Scaling group.**
4. Give the Auto Scaling group a name: `[Your Initials]-Lab-AutoScaling-Group`
5. From the Launch Template drop down choose the launch template named **[Your Initials]-scaling-template** you created in the previous section and select **Next**.

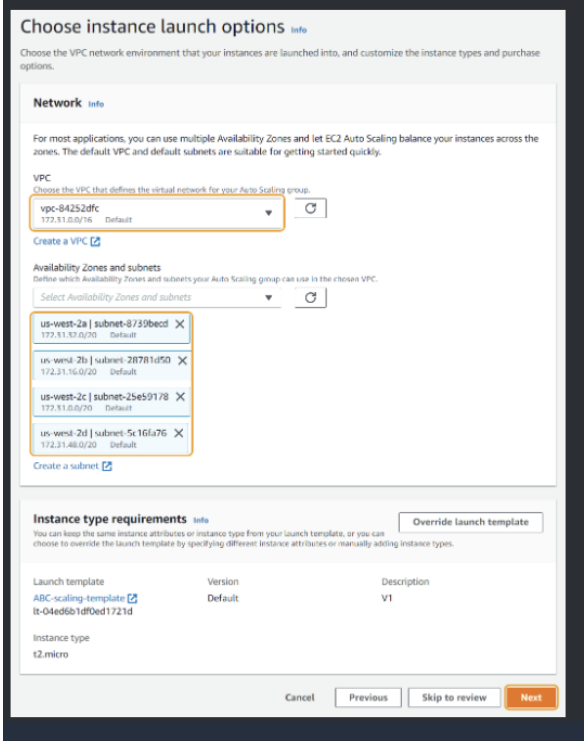



6. Configure settings page, configure the following and select **Next**:

***Network***:

- VPC: **Select** your VPC (most likely Default)
- Subnets: **Select** the subnets where you would like the auto scaling group to use when spinning up the hosts. (If you are using the default VPC, this will most likely be four subnets, as shown below)

A best practice for your Auto Scaling Group would be to select only private subnets. The instances will be sitting behind a load balancer and will not need public IP addresses. For the sake of this lab, they could be either private or public subnets.



7. Specify load balancing and health checks:

a. *Load balancing*: **Attach to a new load balancer**

b. *Load balancer type*: **Application Load Balancer**

c. *Load balancer name*: `[Your Initials]-Application-Load-Balancer`

d. *Load balancer scheme*: **Internet-facing**

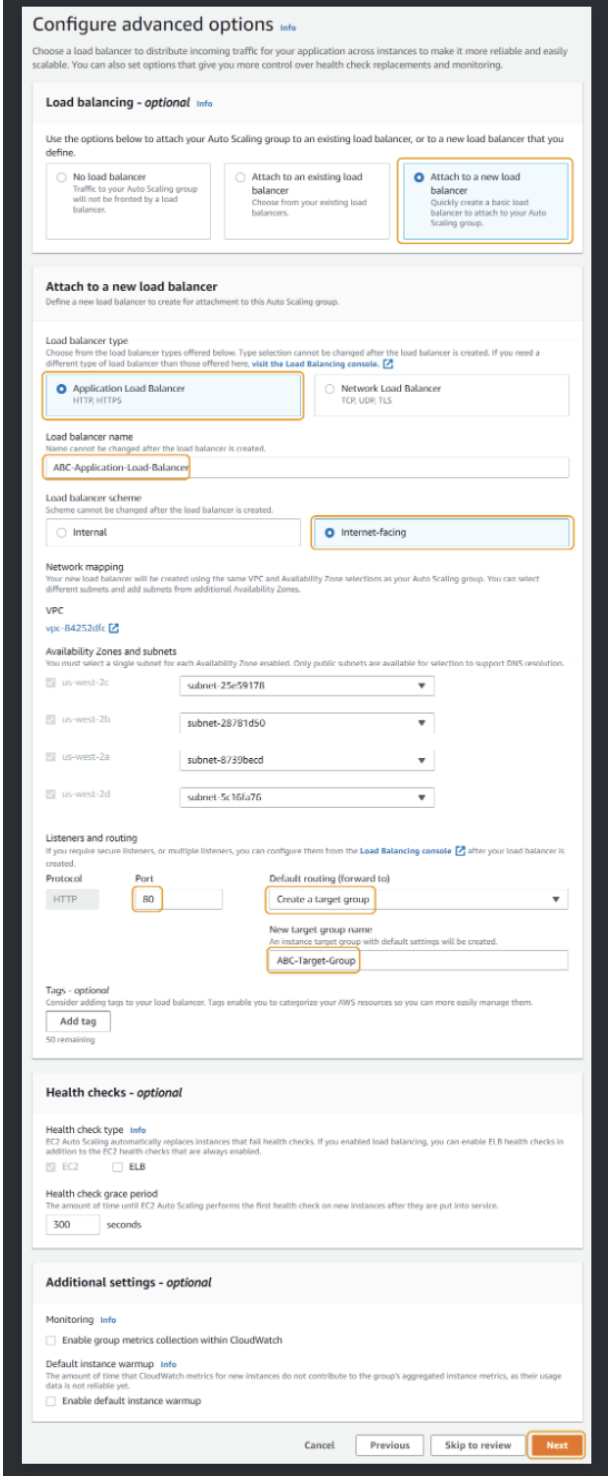
e. *Networking mapping*: You should see all the Availability Zones and subnets you selected in the previous step. (If you had multiple subnets per AZ, this were it would let you choose between them)

f. *Listeners and routing*: Keep the Port as 80 and select **Create a target group** from the "Default routing (forward to)" dropdown.

g. *New target group name*: `[Your Initials]-Target-Group`

- The target group is where your load balancer is going to look for instances to distribute traffic. We are setting our auto scaling group to automatically register instances into this group and it will also be associated to our load balancer.

h. *Health checks & Additional settings*: Leave as default and select **Next**.



8. Configure the group size and scaling policies below and then select **Next**.

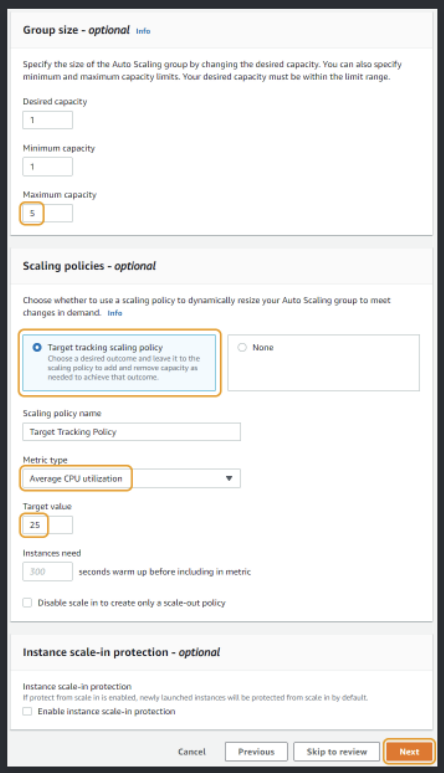
   a. *Group Size*: The settings below will keep our group size to one EC2 instance unless a scaling policy is triggered.

   - Desired capacity: 1
   - Minimum capacity: 1
   - Maximum Capacity: 5

   b. *Scaling policies*: Select **Target tracking scaling policy**

   - Metric type: **Average CPU utilization**
   - Target Value: 25

We are going to set our target CPU utilization low to speed up the lab.




9. Add Notifications:

   You can configure your Auto Scaling Group to send notifications to an endpoint that you choose, such as an email address. You can receive notifications whenever a specified event takes place, including the successful launch of an instance, failed instance launch, instance termination, and failed instance termination.

   For now, you are going to skip this step, select **Next**.

10. Add Tags: Add a single tag and then select **Next**.

    a. Select the **Add tag** button and configure the following:

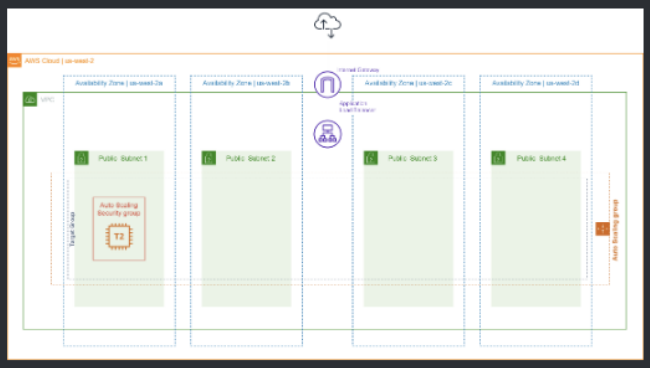
    - Key: `Name`
    - Value: `[Your Initials] - Auto Scaling Group`

11. Review you settings and then select **Create Auto Scaling group**. You have now created your Auto Scaling Group, target group and load balancer.

    a. You will soon see a new instance created by the Auto Scaling group in the EC2 console with the name tag "[Your Initials] - Auto Scaling Group". (You may need to refresh the screen to see the instance)

    b. If you select **Load Balancers** under "Load Balancing" in the left hand menu, you will see your load balancer provisioning.

In the next step we will create an additional security group and update the security settings to allow traffic to flow between the ALB and our web hosts.




**You are now ready to move onto the next step**

**Configuring Security Groups**

## Creating a Load Balancer Security Group

When our load balancer was provisioned it was setup with the default security group in our VPC. To allow access to the load balancer via the public DNS, we will need to create and attach a security group to allow inbound traffic on port 80 from the internet.

We will also create an outbound rule that allows outgoing traffic from the load balancer to only be sent to hosts within the Auto Scaling Security Group.

Within the console Under "Services" search and select **EC2**. On the EC2 page under the "Network & Security" heading in the left-hand menu select **Security Groups**. You should see other security groups, including the security group for your web server named [Your Initials]-EC2-Web-Host - Website Security Group. Click on the **Create security group** button.

1. Basic details:

   a. *Security group name*: `[Your Initials]-SG-Load-Balancer`

   b. *Description*: `[Your Initials]-SG-Load-Balancer`

   c. *VPC*: Select your VPC (Most likely the Default VPC)

2. Inbound rules:

   a. Click on the **Add rule** button

   b. *Type*: `HTTP`

   c. *Source*: Custom: `[Input your public IP address followed by a /32]` (You can find your local IP by searching What is my IP.)

3. Outbound rules:

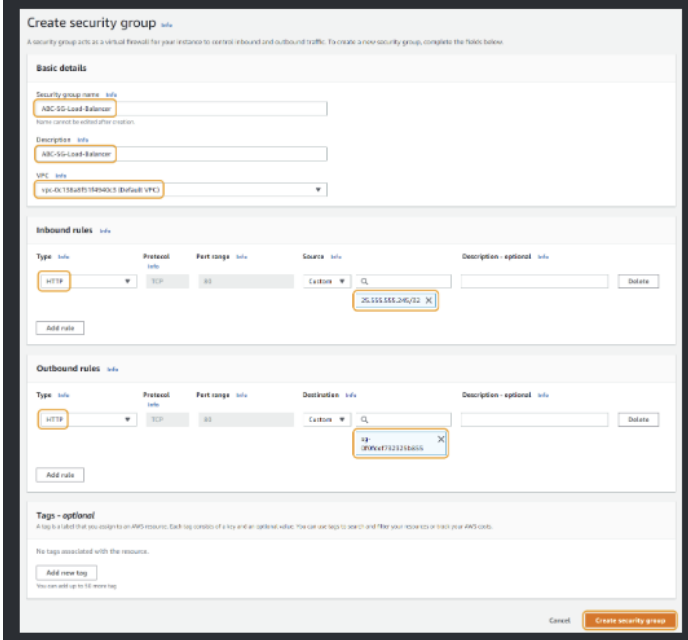
   a. Find the "All traffic" rule and click on **Delete** to remove the rule. (All Outbound rules should now be removed)

   b. Click on the **Add rule** button

   c. *Type*: HTTP

   d. Under "Destination" select **Custom** and in the field select your **[Your Initials]-Auto Scaling SG** as the "Destination". Hint: start by typing `sg` to get the Security Group list.

   f. You security group configuration should look similar to the image below. Select **Create security group** when finished.



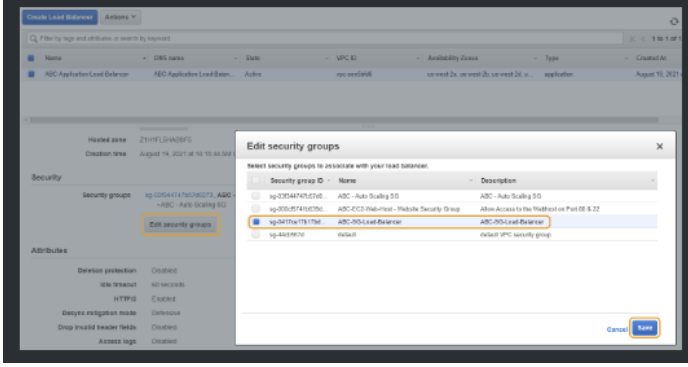

4. Attach your new Load Balancer Security group to your Load Balancer:

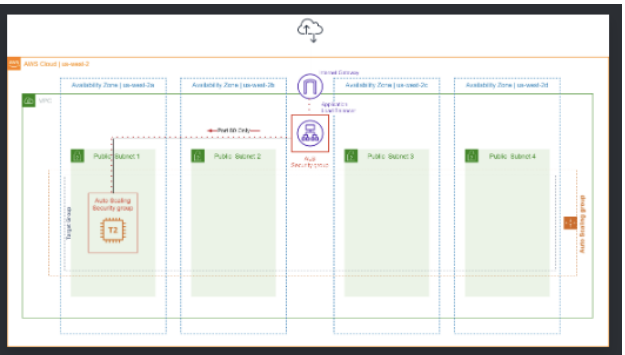
   a. On the EC2 service page left side menu find "Load Balancing" and select **Load Balancers**. Select the load balancer you created. Make sure the State is "Active".

   b. Under the "Description" tab scroll down to the "Security" section and click on **Edit security groups**.

   c. Select the box to the left of your new load balancer sg named `[Your Initials]-SG-Load-Balancer`.

   d. Make sure you also **un-select** any other security group and then click on the **Save** button.





## Task 4: Add Inbound Rule to the Auto Scaling Security Group

1. We will need to setup a rule to only allow traffic from the new Load Balancer Security Group to the Auto Scaling Security Group. This will be one of the layers of protection that will prevent our webhosts from being directly accessed from the internet.

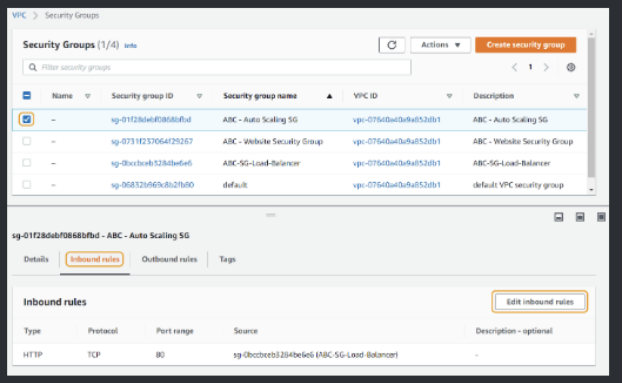
   a. On the EC2 service page left side menu under "Network & Security" select **Security Groups**.

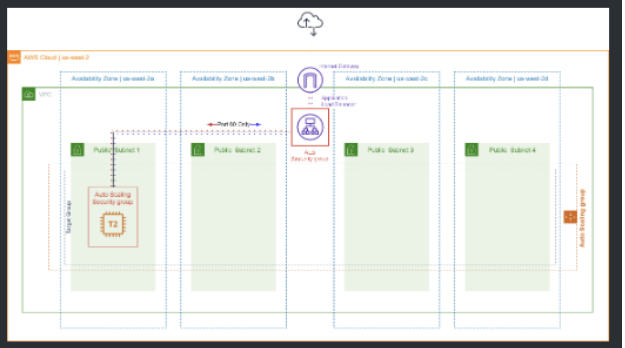
   b. Select your Auto Scaling Security Group: **[Your Initials] - Auto Scaling SG**

   c. Select the **Inbound Rules** tab and click on the **Edit inbound rules** button and then the **Add rule** button.

   d. From the "Type" drop down select **HTTP**. Under "Source" select **Custom** and in the field specify your **[Your Initials]-SG-Load-Balancer** as the "Source". Hint: start by typing `sg` to get the Security Group list. Now click on **Save rules**.

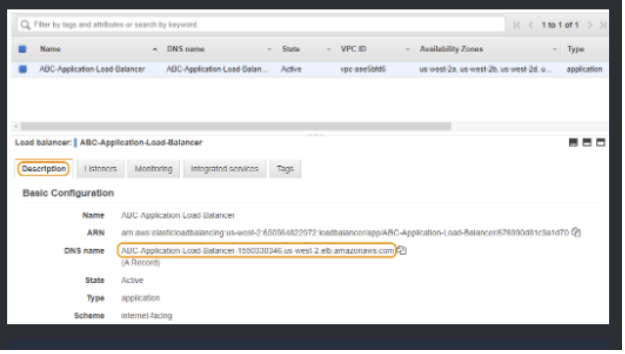
Your rule should now look similar to the image below.

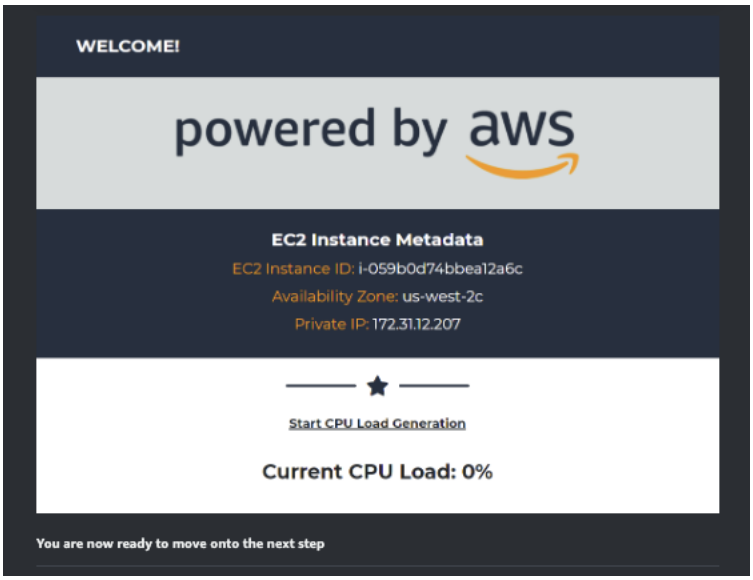


2. We will now test to make sure your load balancer is working. There is currently only one instance (or target) running in the auto scaling group, but you should be able to access the website.

   Return to your the load balancers page by selecting **Load Balancers** from the left hand menu. Under the "Description" tab **copy** the DNS name and **paste** it into a web browser. You should now see the website being loaded from your auto scaling group. Leave this page open, you will need it in the next step.



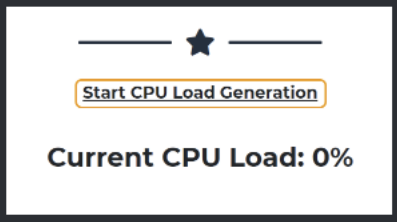



**Testing the Auto Scaling Group**

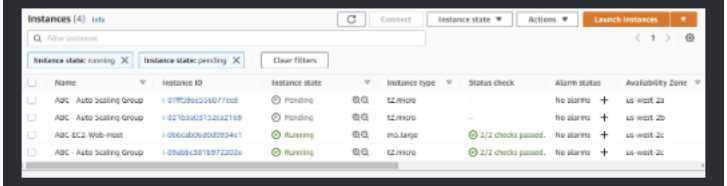
## Testing the Auto Scaling Group

Now that you have created your Auto Scaling Group and load balancer, let's test it to ensure that everything is working correctly.

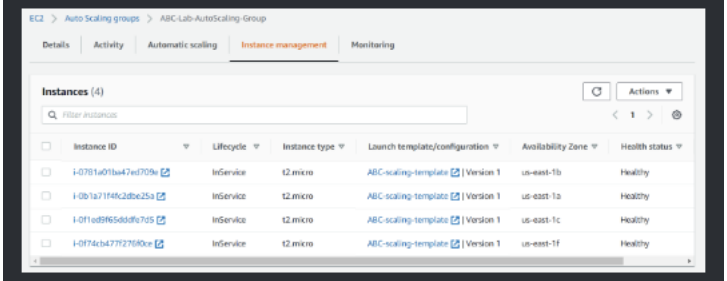
1. Make sure you are on the website accessed through the Load Balancer DNS address in the previous step.
2. At the bottom of the front page click on the **Start CPU Load Generation** link: Once the CPU load goes above 25% for a sustained period the Auto Scaling policy will begin spinning up the instances specified in the launch template to meet demand. *(You may have to do this twice if the first time doesn't generate enough load)*



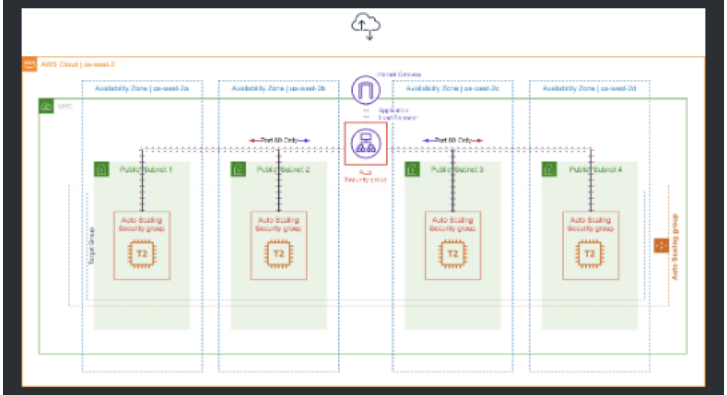

3. In the "Instances" section of the EC2 Console you can watch for the new instances created by Auto Scaling, this might take a couple of minutes. Refresh the EC2 instances page and you should soon see a new instance spinning up automatically. You can select the instance named [Your Initials] - Auto Scaling Group and click on the **Monitoring** tab below to keep an eye on the "CPU Utilization".




4. You can also see this by going to the Auto Scaling Groups page. https://console.aws.amazon.com/ec2autoscaling Then select your auto scaling group **[Your Initials]-Lab-AutoScaling-Group**. If you look at the details under the Instance management tab, you can see if new instances are spinning up. You can look at the Instance management tab to see how many instances there are in your group currently. The monitoring tab shows you different metrics like group size, pending instances, total instances, and much more.




## Your architecture now looks like this:




5. Once a number of new instances have successfully started (probably 3 or 4), repeatedly refresh your web-browser on you web host. You should now see the Instance ID, Availability Zone and Private IP change as the load balancer distributes the requests across the Auto Scaling group.


Congratulations! Your have successfully created an EC2 Auto Scaling Group behind an Application Load Balancer.

Lab Author: Cy Hopkins

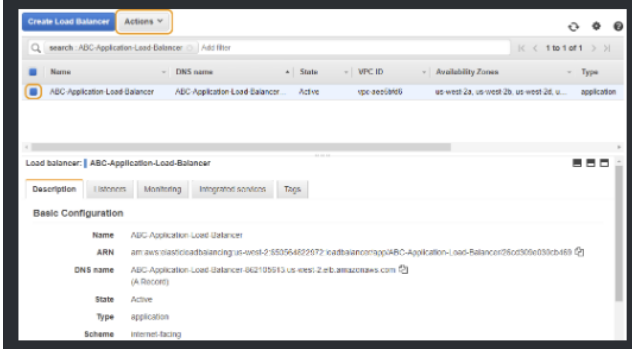
**If you need to cleanup this lab, move on to the next section**

**Auto Scaling Lab Teardown**

It's important that you delete the resources created in this lab in the order below.

## Deleting your Load Balancer

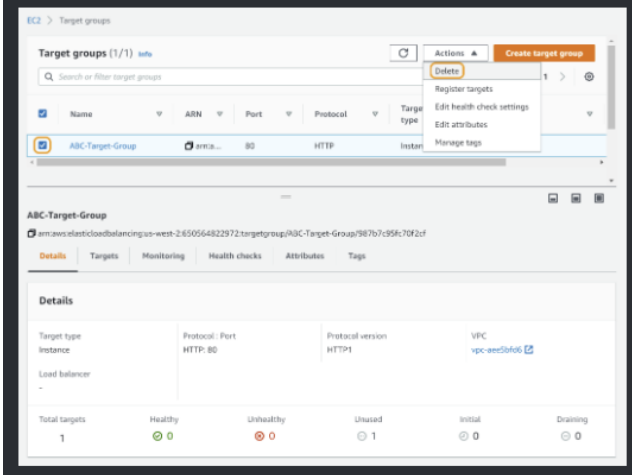
1. In the console navigate to the "EC2" service. From the left menu under "Load Balancing" select **Load Balancers**.
2. Select the load balancer named **[Your Initials]-Application-Load-Balancer**.
3. Click on the **Actions** drop down at the top of the page and select **Delete**. In the following popup window select the **Yes, Delete** button.




4. Your load balancer should be gone immediately if your delete was successful.

## Deleting your Target Group

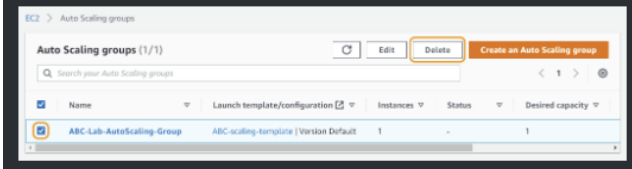
1. In the console navigate to the "EC2" service. From the left menu under "Load Balancing" select **Target Groups**.
2. Select the target group named **[Your Initials]-Target-Group**
3. Click on the **Actions** drop down at the top of the page and select **Delete**. In the following popup window select the **Yes, Delete** button.




4. Your target group should now be deleted.

## Deleting your Auto Scaling Group

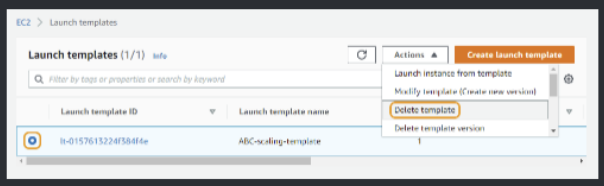
1. In the console navigate to the "EC2" service. From the left menu under "Auto Scaling" select **Auto Scaling Groups**.
2. Select the auto scaling group named **[Your Initials]-Lab-AutoScaling-Group**
3. Click on the **Delete** button at the top of the page. In the following popup window type `delete` in the text field and select the **Delete** button.




4. All of the instances in your auto scaling group will now be terminated, it may take a few minutes to complete.

## Deleting your Launch Template

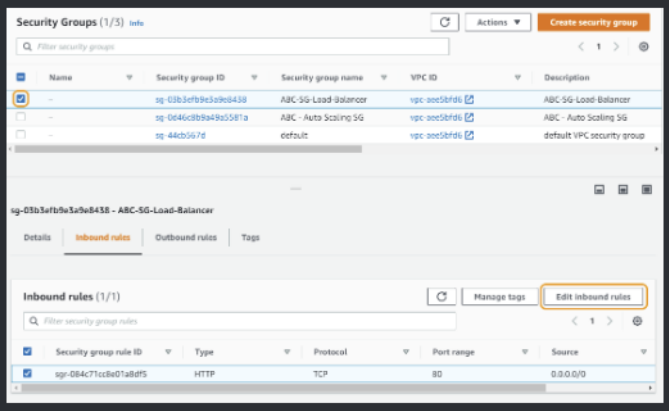
1. In the console navigate to the "EC2" service. From the left menu under "Instances" select **Launch Templates**.
2. Select the launch template named **[Your Initials]-scaling-template**.
3. Click on the **Actions** drop down at the top of the page and select **Delete template**. In the following popup window type `delete` in the text field and select the **Delete** button.



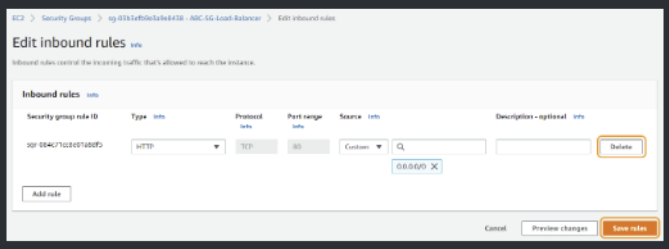

4. Your launch template should now be deleted.

## Deleting your Security Groups

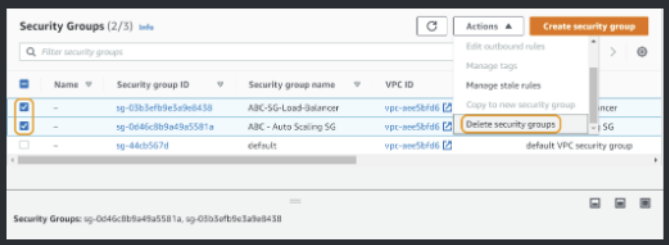
1. In the console navigate to the "EC2" service. From the left menu under "Network & Security" select **Security Groups**.
2. Select the security group named **[Your Initials]-SG-Load-Balancer** and click on the **Edit inbound rules** button.




3. On the "Edit inbound rules" page, click on the **Delete** button to delete the one rule and then select the **Save rules** button.



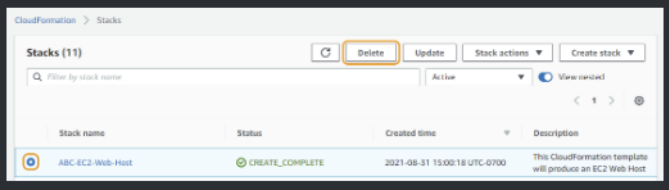

4. Now select the **Outbound rules** tab and select the **Edit outbound rules** button.
5. On the "Edit outbound rules" page, click on the **Delete** button to delete the one rule and then select the **Save rules** button.
6. Now only select the security group name **[Your Initials] - Auto Scaling SG** and repeat steps 2 & 3. (No need to remove the outbound rule on this security group)
7. Now select the two security groups named **[Your Initials]-SG-Load-Balancer** and **[Your Initials] - Auto Scaling SG**.
8. Click on the **Actions** drop down at the top of the page and select **Delete security groups**. (You may have to scroll down in the Actions menu) In the following popup window type `delete` in the text field and select the **Delete** button.



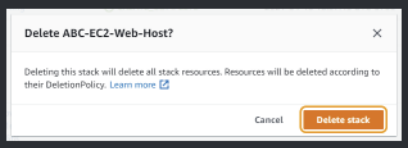

9. Your security groups should now be deleted. If you were not able to remove the security groups you may not have removed all the rules successfully.

## Deleting your CloudFormation Stack

1. In the console open CloudFormation under services or with search.
2. Select the stack named **[Your Initials]-EC2-Web-Host** and then the **Delete** button.




3. In the popup select **Delete stack.**




4. The stack will take a few minutes to delete, select the refresh button to see the updated status. The stack will no longer be visible when it is deleted.

## Congratulations! You have completed the Auto Scaling Lab!