**ACIT 3640 - Cloud Computing**

**(Lab 4)**

**Creating a Website Using Load Balancing in AWS**

Create a Website using LAMP

In this lab, you will create a LAMP (**L**inux, **A**pache, **M**ySQL, **P**HP) stack instance. You have two options for building a LAMP stack. You can build a LAMP stack yourself or you can launch a prebuilt LAMP stack instance. In this lab, you will create both types of instances.

Part 1: Build Your Own LAMP Stack

1. Get a list of Ubuntu AMIs by going to this website: <http://cloud-images.ubuntu.com/locator/ec2/>
2. Now, go to the bottom of the screen. This list contains about approximately 965 entries. Use the dropdown menu boxes at the bottom of each column to filter and find this particular image.

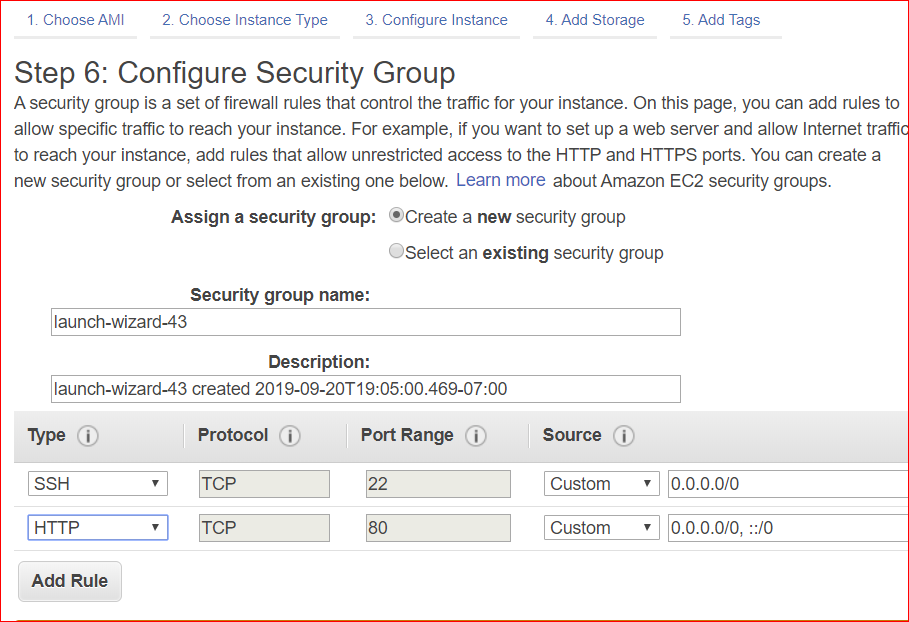
Zone: us-west-2 [Oregon]

Version: 16.04 LTS [Ubuntu server]

Arch: amd64 [64-bit AMD processor]

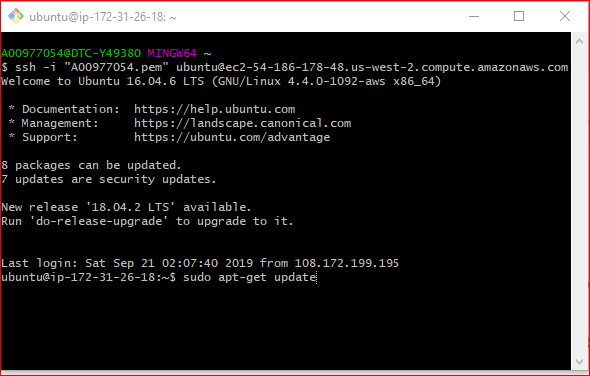
Instance type: hvm: ebs-ssd [Hardware Virtual Machine, Elastic Block Storage, Solid State Drive]

1. This should leave you with one choice. Now, click on the AMI-ID number under the AMI-ID column. This will take you straight into the AWS console.
2. When selecting the security group, make sure your security group allows both SSH and HTTP access.



1. Using Git Bash, SSH into your instance using your .pem file.
2. The username is **ubuntu**, *not* ec2-user.
3. Update this image using apt-get.

sudo apt-get update



1. Install the Apache2 webserver:

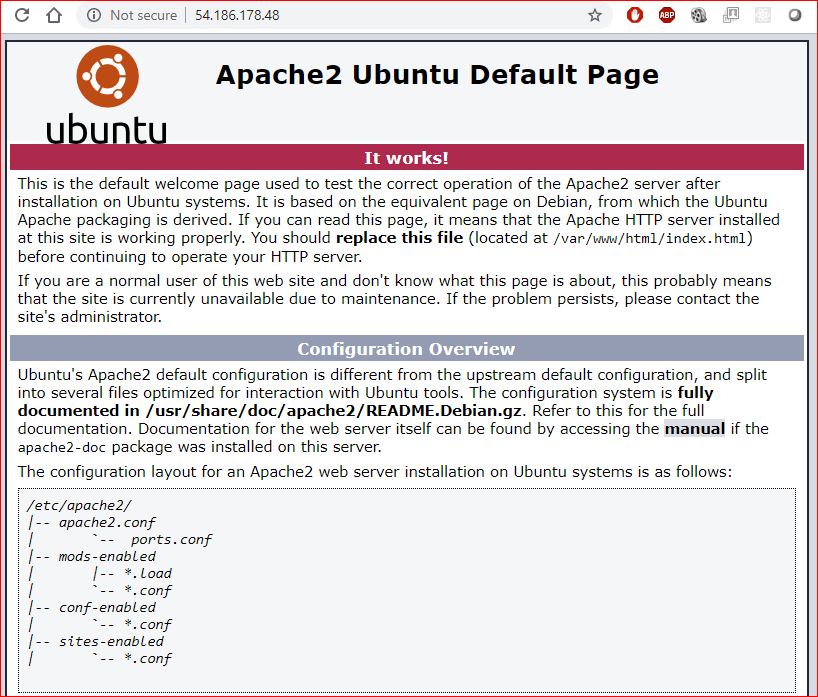
sudo apt-get install apache2

1. Enable and start Apache:

sudo systemctl enable apache2

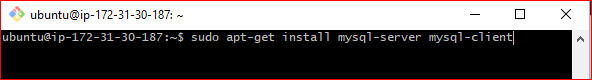
sudo systemctl start apache2

1. Launch a web browser from the lab workstation or your laptop and enter the instance Public IP address as the URL. The Apache2 Ubuntu Default Page should appear in your browser.



1. Install the MySQL database:

sudo apt-get install mysql-server mysql-client



1. Install PHP:

sudo apt-get install php libapache2-mod-php php-mcrypt php-mysql

1. Test the PHP version:

php –v

1. Update everything and restart Apache:

sudo apt-get update

sudo systemctl restart apache2

1. Create a PHP web page:

cd /var/www/html

sudo nano index.php

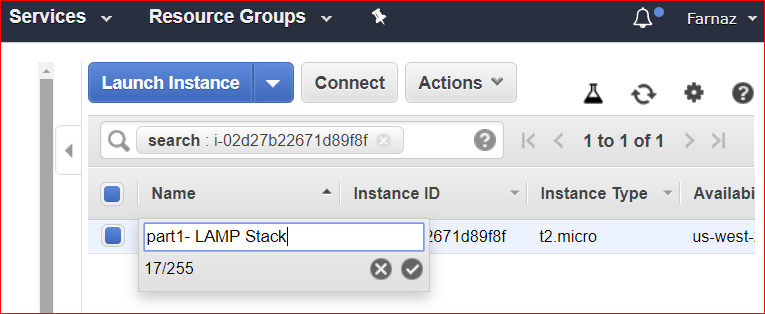
<?php  
 $url = "http://169.254.169.254/latest/meta-data/instance-id";  
 $instance\_id = file\_get\_contents($url);  
 echo "Instance ID: <b>" . $instance\_id . "</b><br/>";

$url = "http://169.254.169.254/latest/meta-data/public-ipv4";  
 $public\_ip = file\_get\_contents($url);  
 echo "Public IP: <b>" . $public\_ip . "</b><br/>";  
?>

1. Enter the Public IP address of the instance, followed by /index.php, in your browser. What should be displayed?

Note: this URL: <http://169.254.169.254/latest/meta-data/> in the above php code, is a link to view all categories of instance metadata from within a running instance.

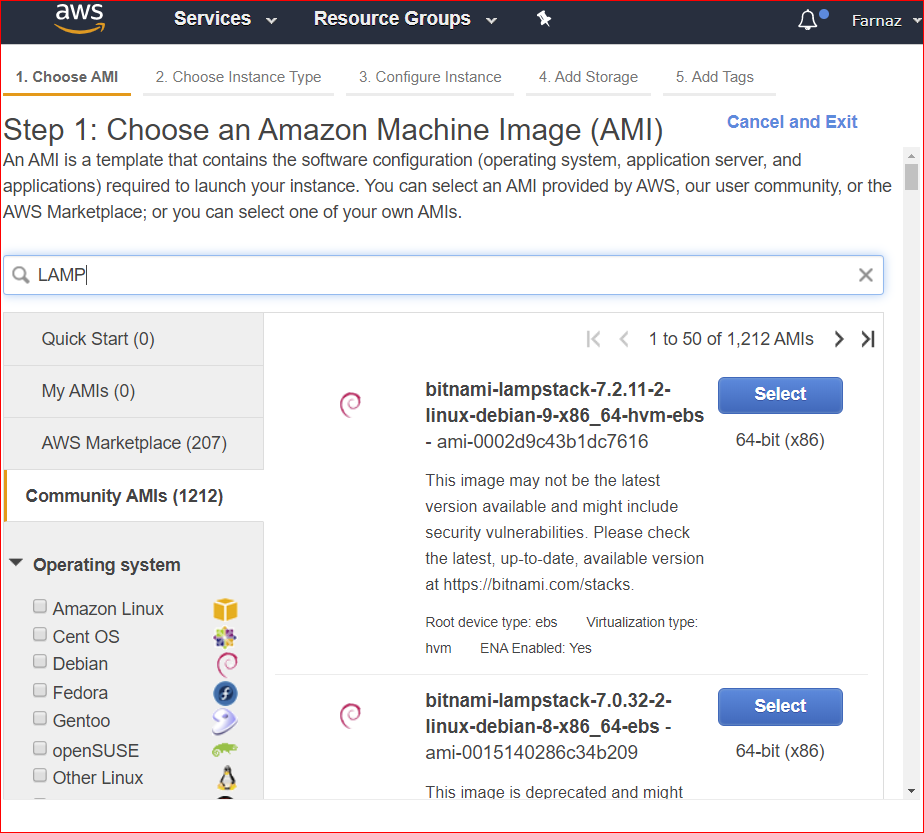
1. Give a name to your running instance as”part1- LAMP Stack”.



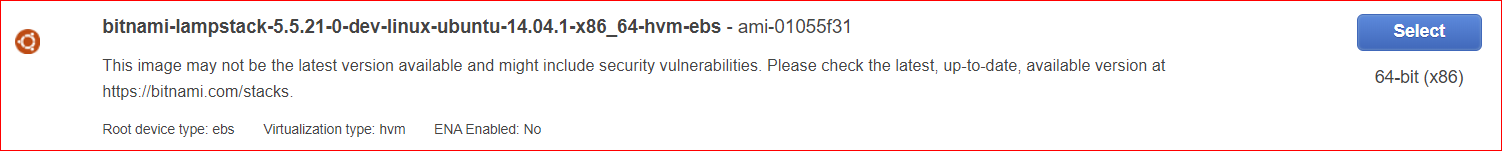
Don’t terminate your instance and follow the next section.

Part2: Use a Prebuilt LAMP Stack Instance

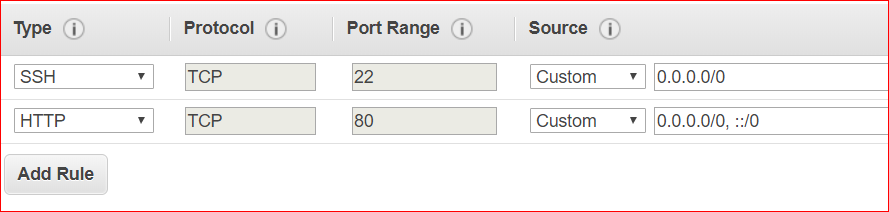
1. Login to the AWS console, choose EC2, Launch Instance, and click Community AMIs on the left side navigation bar. Search for LAMP.



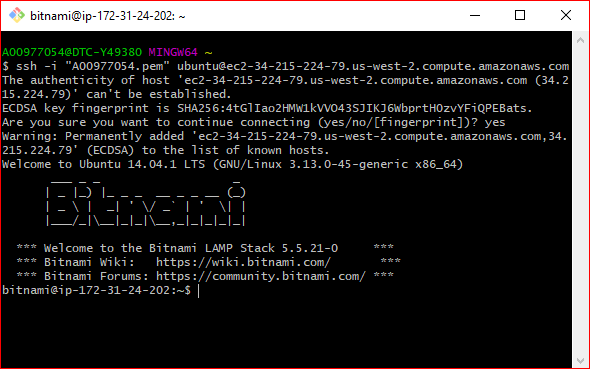
1. You will see a long list of bitnami AMIs (Amazon Machine Images). Scroll down until you find a *non-deprecated* Ubuntu AMI.



1. Select the instance. When selecting the security group, make sure your security group allows both SSH and HTTP access.



1. SSH into the instance.



1. Navigate to this directory: /opt/bitnami/apache2/htdocs. This is the root directory of the Apache web server.



1. Create a PHP web page in this directory:

sudo nano index.php

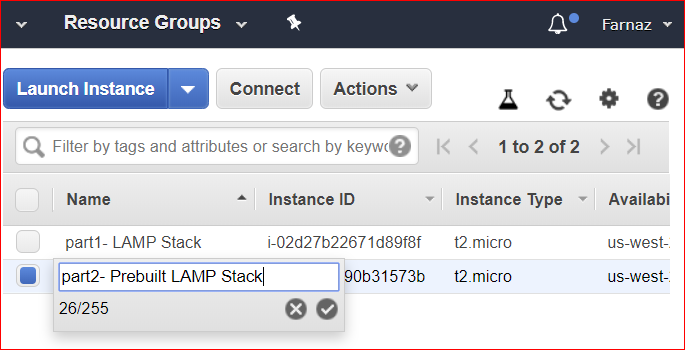
<?php  
 $url = "http://169.254.169.254/latest/meta-data/instance-id";  
 $instance\_id = file\_get\_contents($url);  
 echo "Instance ID: <b>" . $instance\_id . "</b><br/>";

$url = "http://169.254.169.254/latest/meta-data/public-ipv4";  
 $public\_ip = file\_get\_contents($url);  
 echo "Public IP: <b>" . $public\_ip . "</b><br/>";  
?>

1. Enter the Public IP address of the instance, followed by /index.php, in your browser. What should be displayed?

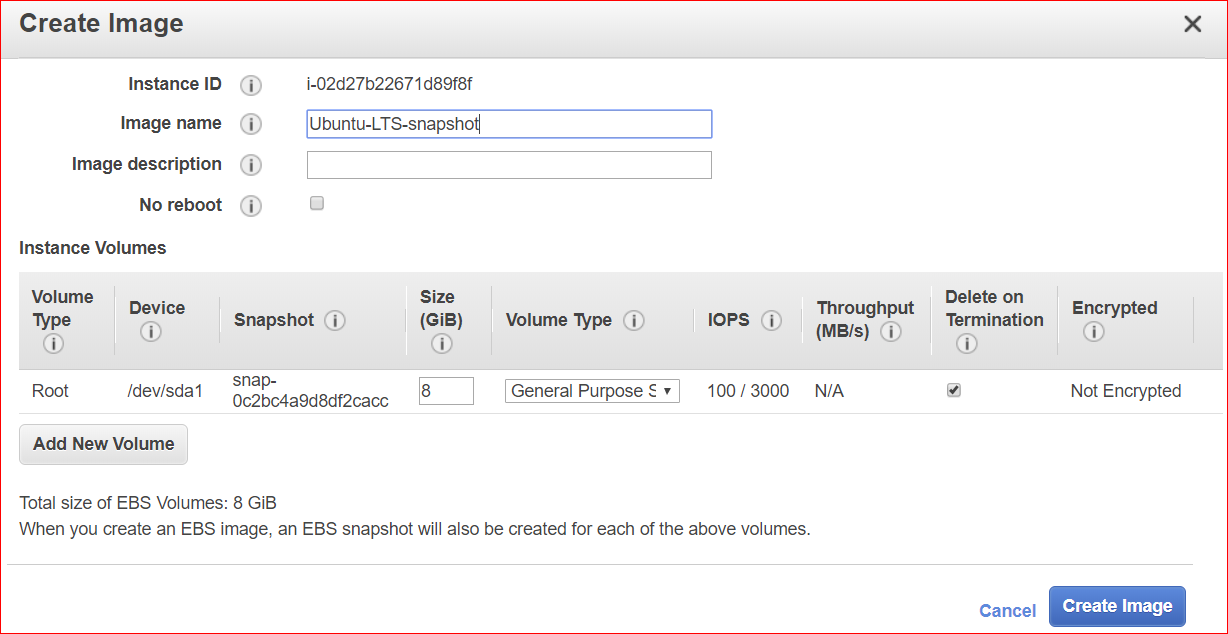
This web page displays metadata about the instance, specifically the Instance ID and the Public IP address. For more information on retrieving AWS Instance metadata and user data, see: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html>.

1. Give a name to your running instance as”part2- Prebuilt LAMP Stack”. Leave your Ubuntu instance running.

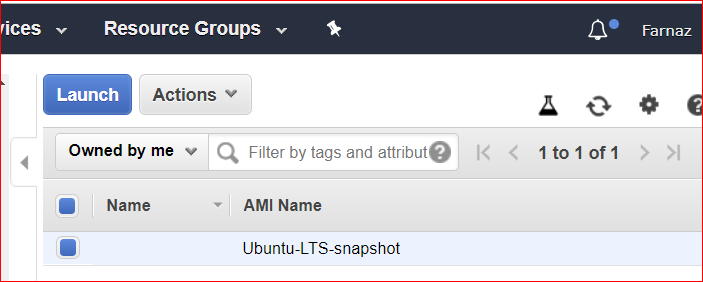


Part3: Clone the Original AMI

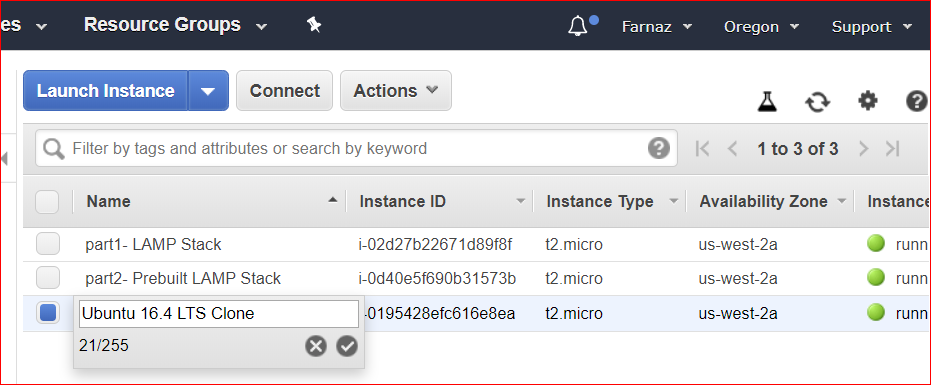
1. You are going to take a snapshot of Ubuntu LAMP stack instance that you built yourself (Instance for part 1). Make sure this instance is running.
2. Right click on the running instance and select **Image -> Create Image.**
3. Write down or record the Instance ID. Add an image name that consists of the Key Name and the word “snapshot.” Example: Ubuntu-LTS-snapshot.



1. Click Create Image.
2. If you click on Snapshots on the left side of your screen, you will not see any snapshots – at this time. Instead, click on AMIs, under the Images area. You will notice that your snapshot is being first built as an AMI image. Refresh this screen until the status changes from “pending” to “available.”
3. Go back to the AMI image screen. Select the snapshot of the new AMI, right click and **Launch**.



1. Before clicking on **Review and Launch,** make sure to:
   1. Use the same Security Group (SSH and HTTP) as you used for the first instance.
   2. Note 2: Make sure you launch this instance in the same Availability Zone as the first instance (Oregon).
2. On the EC2 Dashboard, add a Name for your new instance, e.g. “Ubuntu 16.4 LTS Clone.”



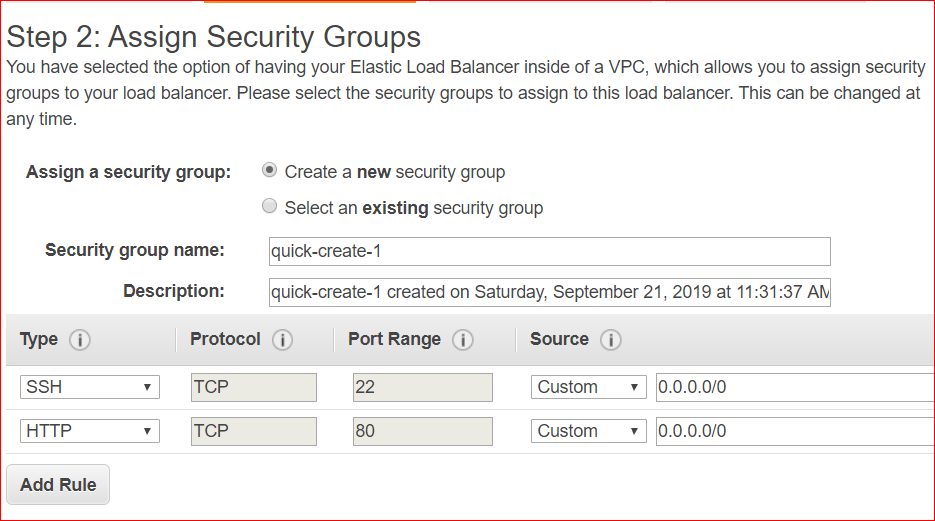
1. Enter the IP address of your new instance (Ubuntu 16.4 LTS Clone), followed by index.php. It should display the Instance ID and Public IP address of the instance.
2. Do not do part 4 (creating the load balancer) unless index.php program works on this instance! (Did you remember to open port 80 HTTP on the security group for the new instance?)
3. Write in the Instance ID and Public IP address of each instance (Enter public IP address of each instance followed by index.php in your browser)
   1. Instance ID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   2. Public IP address: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   3. Instance ID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   4. Public IP address: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   5. Instance ID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   6. Public IP address: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Show above information on the three open tabs in your browser to your instructor.

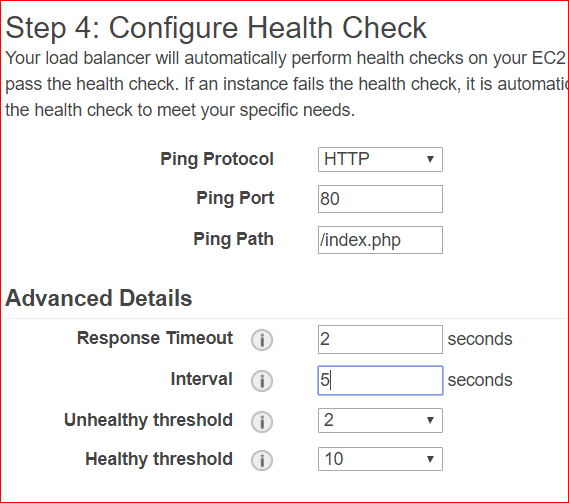
Part4: Create a Load Balancer

You are now going to create a load balancer to manage the traffic to your website.

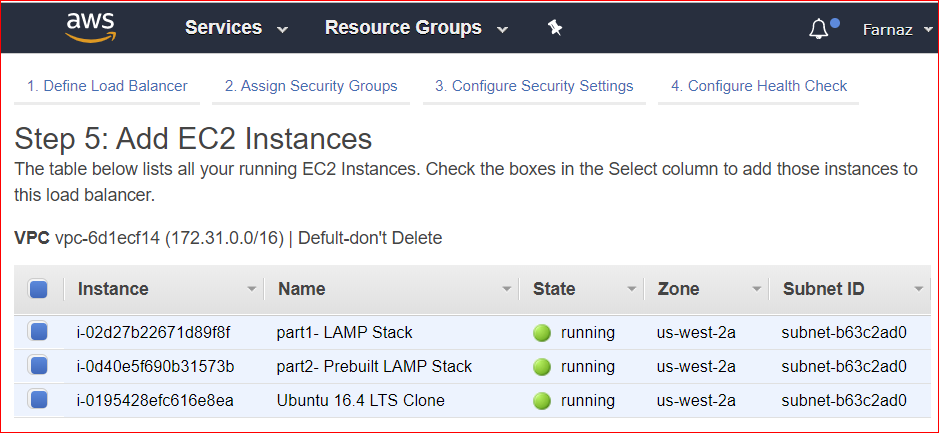
1. On the left navigation bar, find and select “Load Balancers.” Click **Create Load Balancer.**
2. Select Classic Load Balancer and Continue.
3. Enter a name like “A0012345-load-balancer” and accept the default values on the remaining fields.
4. On the next screen, create a new security group and add Rule for SSH and HTTP access.



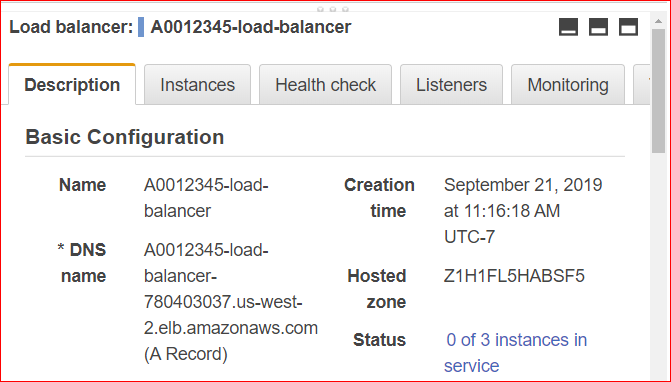
1. On Step 4: Configure Health Check, go to the Ping Path field. Remove “/index.html” and enter “/index.php” Set the Response Timeout to 2 and the Health Check Interval to 5.



1. On Step 6, select the three instances running your website.



1. Continue through the remaining steps and create your load balancer.
2. Select Load Balancers on the left menu. Notice that your load balancer is listed.
3. Click the Description tab on your load balancer. Note the **Status: 0 of 3 instances in service.**



1. Click on the hyperlink. You will see your three instances listed. The status of the instances is probably “OutOfService.” You can hover over the status with your mouse.
2. Click on the Description tab again. When the status reads **Status: 3 of 3 instances in service**, your load balancer is running.
   1. Note: Make sure the Load Balancer is in the same Availability Zone as the three instances.
3. Now, you want to access the websites running on the three instances. Instead of using the Public DNS name for each instance, use the DNS Name for the load balancer. It is on the Description tab. So, copy that name – without the “A Record” -- and use it as the URL in your web browser. Then, add “/index.php” at the end of the URL. Click the Refresh button on your browser. Which instance was accessed? Refresh your browser again. Did it use the second or third instance? Ask your neighbor to access your website.
4. Go to the EC2 Dashboard. Stop one of your instances. Refresh the tab on your browser containing the URL of the load balancer.
5. Go to the EC2 Dashboard. Restart the instance that was stopped. Refresh the tab on your browser containing the URL of the load balancer. Are the Instance IDs the same as you recorded in Step 10, above? Are the Public IP addresses the same as you recorded in Step 10, above?

Show your work to your instructor:

Enter DNS Name for your load balancer followed by index.php in your browser. By refreshing browser you should show that load balancer is using different instances.

Clean Up

1. Delete the load balancer.
2. Deregister the AMI.
3. Delete your snapshot.
4. Terminate all of your running instances.