

Load balancer:

Task 1. Launch an EC2 instance

In this task, you launch an Amazon Elastic Compute Cloud (Amazon EC2) instance as you have in previous labs.

3. In the search box to the right of **Services**, search for and choose **EC2** to open the EC2 console.
4. In the left navigation pane, choose **EC2 Dashboard** to ensure you are on the dashboard page.
5. Choose the **Launch instance** button in the middle of the page, then select **Launch instance** from the dropdown menu.
6. In the *Name and tags* panel:
 - o For **Name** enter Web Server 1
7. In the *Application and OS Images* panel:
 - o For **Quick Start**, keep the default **Amazon Linux** chosen
8. In the *Instance type* panel:
 - o Keep the default instance type, **t2.micro**.
9. In the *Key pair (login)* panel:
 - o From the **Key pair name - required** dropdown list, choose **vockey**.

10. In the *Network settings* section, choose **Edit**.


- o From the **Subnet** dropdown list, choose the existing subnet in **Availability Zone us-east-1a**.
- o For **Security group name - required**, enter Web Server security group
- o For **Description - required**, enter Security group for my web server
- o In the **Inbound security groups rules** section, Select **Remove** to remove the default rule.
- o Choose **Add security group rule** to configure new rule as below
 - **Type** : HTTP
 - **Source type** : Anywhere

11. In the *Configure storage* panel:

- o Keep the default storage configuration.

12. Scroll down, and expand **Advanced Details** Panel, then configure:

- o Scroll down to the field **User data**.
- o Copy the following code and paste it into the **User data** field.



```
#!/bin/bash
yum update -y
yum -y install httpd
systemctl enable httpd
systemctl start httpd
echo '<html><h1>Hello World! This is server 1.</h1></html>' >
/var/www/html/index.html
```

This script does the following:

- Updates the server
- Installs an Apache web server (httpd)
- Configures the web server to automatically start on boot
- Starts the web server
- Creates a simple webpage

13. Choose **Launch instance** .

14. On the next screen, Choose **View all instances**.

15. Before you continue, wait for your instance to display the following:

- o **Instance state:** Running
- o **Status check:** 2/2 checks passed

Tip: To refresh the instance information, choose the refresh icon.

Task 2. Access your EC2 instance's website

In this task, you will access the content from the web server on the EC2 instance that you just created.

18. Select the **Web Server 1** instance that you created earlier in this lab.

19. In the **Details** tab copy the **Public IPv4 address** of your instance, then open a new tab in your web browser and paste in and load the address.

It should display the web server page with the message *Hello World! This is server 1.**

Note: If web page is not displayed, ensure that you are accessing page using http:// (not https://).

Task 3. Create a second EC2 instance for load balancing

In this task, you will create a second EC2 instance so that you will later be able to configure load balancing between the two instances.

20. Return to the **EC2 Management Console** browser tab.

21. Select the **Web Server 1** instance.

22. From the **Actions** menu, choose **Images and templates**, then choose **Launch more like this**

A **Launch an instance** page opens.

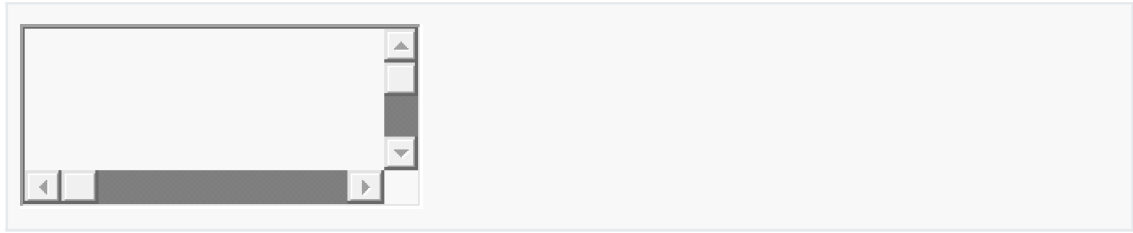
23. In the **Name and tags** pane, change the Name to **Web Server 2**.

24. In the **Key pair (login)** section, from the **Key pair name - required** dropdown list, choose **vockey**.

25. From the **Subnet** dropdown list, choose the existing subnet in **Availability Zone us-east-1b**.

26. Scroll down, and expand the **Advanced Details** section, then scroll down to the **User data** field.

27. Use copy and paste to replace the existing code with the code shown below.



```
#!/bin/bash
yum update -y
yum -y install httpd
systemctl enable httpd
systemctl start httpd
echo '<html><h1>Hello World! This is server 2.</h1></html>' > /var/www/html/index.html
```

Note: This script is almost the same as the one that you used for the first instance. However, notice that it says *This is server 2*. The text that displays when you access Web Server 2 will be different than the text of Web Server 1. When you access the instances through the load balancer, this difference in text is how you will know which instance is displayed.

28. Choose **Launch instance** .

29. On the next screen, Choose **View all instances**.

30. Before you continue, wait for your instance to display the following:

Instance state: Running

Status check: 2/2 checks passed

Tip: To refresh the instance information, choose the refresh icon.

Task 4. Access the website on the second EC2 instance

31. Select the **Web Server 2** instance.

32. In the **Details** tab copy the **Public IPv4 address** of your instance, then open a new tab in your web browser and paste in and load the address.

33. This will open a new tab in your web browser and display the web server page with the message *Hello World! This is server 2.*

Make a note of the *Availability Zones* where the **Web Server 1** and **Web Server 2** instances are running. For example, **us-east-1a** and **us-east-1b**. You will need this information in the next task.

Task 5. Create a load balancer

34. Back in the EC2 console, in the left navigation pane, under **Load Balancing**, choose **Load Balancers**.

35. Select **Create Load Balancer**.

Tip: An *Application Load Balancer* has many features and can be used for HTTP and HTTPS traffic.

36. Under **Application Load Balancer**, choose **Create**.

37. In the *Basic Configuration* panel:

- o For **Name**, enter myloadbalancer.

38. In the *Network mapping* panel:

- o Under **Mappings**, select the Availability Zones that you created the two instances in.
For example, **us-east-1a** and **us-east-1b**.

Note: The Subnet to use in each selected Availability Zone will be automatically populated.

39.

40. In the *Security Groups* panel:

- o Choose **Web Server security group** from the drop down menu.
- o After you close the drop down menu, choose the **X** next to the **default** security group to remove it.

41. In the *Listeners and routing* panel:

- o Choose **Create target group**.

This will open a new tab in your browser.

42.

43. In the *Basic Configuration* panel:

- o Keep the target type set to **Instances**.
- o For **Target group name** enter myalbTG

44. In the *Health checks* panel:

- o For **Health check path**, enter index.html after the forward slash (/)
- o The path should look like the following: /index.html

45. Choose **Next**.

46. In the **Register targets** page, in the **Available instances** panel, check the boxes next to the **Web Server 1** and **Web Server 2** instances that you created in this lab.

47. Choose **Include as pending below**.

Verify that both instances now appear in the **Targets** list below.

48. Choose **Create target group**.

A banner displays the message that the target group was successfully created.

49. Return to the **Load Balancers** console tab in the browser.

50. In the **Listeners and routing** section, under **Listener** choose the refresh icon.

51. From the dropdown, choose the **myalbTG** target group you created.

52. Scroll down and choose **Create load balancer**.

When the load balancer is created, a *Successfully created load balancer* message displays.

53. Choose **View load balancer**.

Before continuing, ensure that the **State** of the load balancer that you just created changes to *Active*.

It can take a few minutes for it to become active.

Tip: To refresh the load balancer information, choose the refresh icon.

Task 6. Test the load balancer

In this task, you will test the load balancer that you just created.

50. Select the load balancer that you just created, and expand the **Details** section.

51. Under **Details**, copy the **DNS name** value to your clipboard.

52. Open a new tab in your web browser, paste the DNS name that you just copied, and press **Enter**.

If your load balancer is working, the *Hello World!* message displays. Notice whether the message says *This is server 1* or *This is server 2*.

53. Refresh the browser tab a few times.

Notice when the message changes between *This is server 1* and *This is server 2*. When the message changes, it means that the load balancer has directed you to the web server on the other EC2 instance that you created in this lab.

elastic beanstalk

Task 1. Deploy an application using Elastic Beanstalk

4. Choose the **Services** menu, locate the **Compute** services, and choose **Elastic Beanstalk**.
5. Choose **Create Application**.
6. For **Application name**, enter a name for your application; for example, `MyLabApp`
7. For **Platform**, select **PHP**.
8. For **Application code**, select **Sample application**.
9. Choose **Next**.

10. Under **Existing service roles** choose the drop-down and select **EMR_EC2_DefaultRole**.
11. Under **EC2 key pair** choose the drop-down and select **vockey**.
12. Under **EC2 instance profile** choose the drop-down and select **EMR_EC2_DefaultRole**.
13. Choose **Next**.
14. Under **VPC** select the available VPC.
15. Under **Public IP address** select **Activated**.
16. Under **Instance subnets** select at least two.
17. Choose **Next**.
18. Under **EC2 Security groups** select the **default**.
19. Choose **Next**.
20. Under **Health reporting** choose **Basic**.
21. Under **Managed updates** de-select **Activated**.
22. Choose **Next**.
23. Choose **Submit**.

Watch the console as Elastic Beanstalk creates and runs the necessary resources to run the application. It takes 5-10 minutes for the process to complete.

Elastic Beanstalk creates an Amazon Simple Storage Service (Amazon S3) storage bucket and a security group, launches an Amazon Elastic Compute Cloud (Amazon EC2) instance, and runs the code.

When complete, the screen changes to show the newly created environment. It is ready for you to upload a PHP application.

10. Open a new browser tab or window. Navigate to the [AWS Tutorials and Samples web page](#).
11. On the page, in the second list of downloads, find **PHP – php.zip**. Download the sample **PHP** application to your computer.

You should now have a file called *php.zip*.

12. Return to the Elastic Beanstalk console tab.
13. Choose **Upload and deploy**.
14. Choose **Choose file**, navigate to and select the *php.zip* file that you downloaded, and choose **Open**.
15. Choose **Deploy**.

The application deploys to the environment using all of the cloud resources Elastic Beanstalk provisioned.

16. To see your PHP website, in the left navigation pane, choose **Go to environment**.

The web application opens in a new tab.

Task 2. Deploy an application using CloudFormation

17. Return to the AWS Management Console tab.
18. Choose the **Services** menu, locate the **Compute** services, and choose **EC2**.
19. In the left navigation pane, under **Network & Security**, choose **Key Pairs**.
20. Choose **Create key pair**.
21. For **Name**, enter `CFLearner`
22. Choose **Create key pair**.
23. When the download window opens, choose **Cancel**. You do not need to download the file.
24. Choose the **Services** menu, locate the **Management & Governance** services, and choose **CloudFormation**.

The **Stacks** list appears and displays the CloudFormation stacks that were created in your lab environment. Notice the stack whose **Description** says *AWS Elastic Beanstalk environment*. This stack was created automatically when you deployed your application through Elastic Beanstalk.

This demonstrates how the two services work together to create an environment to run your code.

25. Choose **Create stack**, and then choose **With new resources (standard)**.
26. In the **Prerequisite** section, choose **Use a sample template**.
27. In the **Select a sample template** section, select **WordPress blog**.

WordPress is web software that you can use to create a website or blog. This template installs a highly available and scalable WordPress deployment using an Amazon Relational Database Service (Amazon RDS) database instance for storage.

28. Select **Next**.
29. For **Stack name**, enter `WordPressStack`
30. In the **Parameters** section, configure the following:

- **DBPassword:** Enter Testing1
- **DBUser:** Enter testadmin
- **KeyName:** Choose the **CFLearner** key pair

1. Choose **Next**, and then choose **Next** again.

You will use the default stack options.

2. Review the stack configuration, and then choose **Submit**.

This initiates the deployment of the resources. You can watch the CloudFormation stack events on the **Events** tab as a complete WordPress site is created.

S3

Task 1. Create an S3 bucket

4. Choose the **Services** menu, locate the **Storage** services, and select **S3**.
5. Select **Create bucket** on the right side of the page.
6. For **Bucket name**, enter a unique Domain Name System (DNS)-compliant name for your new bucket.

Follow these naming guidelines:

- o The name must be unique across all existing bucket names in Amazon S3.
 - o The name must only contain lowercase characters.
 - o The name must start with a letter or number.
 - o The name must be between 3 and 63 characters long.
 - o After you create the bucket, you cannot change the name, so choose wisely.
 - o Choose a bucket name that reflects the objects in the bucket. This is because the bucket name is visible in the URL that points to the objects that you're going to put in your bucket.
7. For **Region**, choose the AWS Region where you want the bucket to reside.

Choose a Region close to you to minimize latency and costs, or to address regulatory requirements. Objects stored in a Region never leave that Region unless you explicitly transfer them to another Region.

8. Uncheck the **Block all public access** box because you want to be able to test if the website is working.

A warning message similar to **Turning off block all public access might result in this bucket and the objects within becoming public** appears below the security setting you deselected.

9. Below the warning, check the box next to **I acknowledge that....**
10. Scroll to the bottom of the page, and select **Create bucket**.

Your new bucket appears in the **Buckets** list.

Task 2. Add a bucket policy to make the content publicly available

11. Choose the link for your bucket's name, and then select the **Permissions** tab.
12. In the **Bucket policy** section, choose **Edit**.
13. To grant public read access for your website, copy the following bucket policy, and paste it in the policy editor.



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::example-bucket/*"
      ]
    }
  ]
}
```

14. In the policy, replace **example-bucket** with the name of your bucket.
15. Select **Save changes**.

Task 3. Upload an HTML document

In this task, you upload an HTML document to your new bucket.

16. Open the context menu (right-click) for the following link, and then choose **Save link as:** [index.html](#)
17. Save the index.html file to your local computer.
18. In the console, choose the **Objects** tab.
19. Upload the index.html file to your bucket.
 - o Choose **Upload**.
 - o Drag and drop the index.html file onto the upload page.
 - o As an alternative, choose **Add files**, navigate to the file, and choose **Open**.
20. Expand the **Properties** section.

This section lists the storage classes that are available in Amazon S3. You will learn more about storage classes later, but take a minute to review them now.

Ensure that the **Standard** storage class is selected.

21. At the bottom of the page, choose **Upload**.
22. Choose **Close**.

The index.html file appears in the **Objects** list.

Task 4. Test your website

26. Select the **Properties** tab, and scroll down to the **Static website hosting** section.
27. Choose **Edit**.
28. Select **Enable**.
29. In the **Index document** text box, enter index.html
30. Select **Save changes**.
31. Scroll down to the **Static website hosting** section again, and copy the **Bucket website endpoint** URL to your clipboard.
32. Open a new tab in your web browser, paste the URL you just copied, and press **Enter**.

The **Hello World** webpage should display. You have successfully hosted a static website using an S3 bucket!

EC2

Task 1. Start creating the instance and assign a name

4. Choose the **Services** menu, locate the **Compute** services, and select **EC2**.
5. Choose the **Launch instance** button in the middle of the page, and then select **Launch instance** from the dropdown menu.
6. Name the instance:
 - o Give it the name `Web Server 1`

Tags help you categorize your AWS resources in different ways; for example, by purpose, owner, or environment. This is useful when you have many resources of the same type—you can quickly identify a specific resource based on the tags you have assigned to it. Each tag consists of a *key* and a *value*, which you define.

Note: *Name* is simply another tag. The *key* for this tag is `Name`, and the *value* is `Web Server 1`.

Task 2. Application and OS Images

7. Choose an AMI from which to create the instance:
 - o In the list of available *Quick Start* AMIs, keep the default **Amazon Linux** AMI selected.
 - o Also keep the default **Amazon Linux 2023 AMI x86_64 (HVM)** selected.

The type of *Amazon Machine Image (AMI)* you choose determines the Operating System (OS) that will run on the EC2 instance that you launch. In this case, you have chosen Amazon Linux 2023 as the guest OS.

8.

Task 3. Choose an instance type

8. Specify an Instance type:

- o In the *Instance type* panel, keep the default **t2.micro** selected.

The *Instance Type* defines the hardware resources assigned to the instance. This instance type has 1 virtual central processing unit (CPU) and 1 GiB of memory.

Task 4. Choose a key pair

9. Select the key pair to associate with the instance:
 - o From the **Key pair name** menu, select **vockey**.

The *vockey* key pair you selected will allow you to connect to this instance via SSH after it has launched. Although you will not need to do that in this lab, it is still required to identify an existing key pair, or create a new one, when you launch an instance.

Task 5. Network settings

10. Next to Network settings, choose **Edit**.

11. Keep the default *VPC* and *subnet* settings. Also keep the **Auto-assign public IP** setting set to **Enable**.

The Network indicates the virtual private cloud (VPC) you want to launch the instance into. You can have multiple networks; for example, one for *development*, a second for *testing*, and a third for *production*.

12. Under *Firewall (security groups)*, keep the default **Create security group** option chosen.

13. Configure a new security group:
 - o Keep the default selection **Create a new security group**.
 - o **Security group name:** Clear the text and enter `Web Server`

- o **Description:** Clear the text and enter `Security group for my web server`
- o Choose **Remove** to remove the default SSH inbound rule.

Note: You will configure a different inbound rule later in this lab.

A *security group* acts as a virtual firewall that controls the traffic for one or more instances. When you launch an instance, you associate one or more security groups with the instance. You add rules to each security group that allow traffic to or from its associated instances. You can modify the rules for a security group at any time. The new rules are automatically applied to all instances that are associated with the security group.

Task 6. Configure storage

14. In the *Configure storage* section, keep the default settings.

You will launch the Amazon EC2 instance using a default Elastic Block Store (EBS) disk volume. This will be your root volume (also known as a *boot volume*) which will host the Amazon Linux 2023 guest operating system that you specified earlier. It will run on a general purpose SSD (*gp2*) hard drive that is 8 GiB in size. You could alternatively add more storage volumes, however that is not needed in this lab.

Task 7. Advanced details

15. Configure a script to run on the instance when it launches:
- o Expand the **Advanced details** panel.
 - o Scroll to the bottom of the page and then copy and paste the code shown below into the **User data** box:



```
#!/bin/bash  
yum update -y
```

```
yum -y install httpd
systemctl enable httpd
systemctl start httpd
echo '<html><h1>Hello World!</h1></html>' > /var/www/html/index.html
```

This bash script will run with root user permissions on the guest OS of the instance. It will run automatically when the instance launches for the first time. This script does the following:

- Updates the server
- Installs an Apache web server (httpd)
- Configures the web server to automatically start on boot
- Activates the web server
- Creates a simple webpage

Task 8. Review the instance and launch

16. At the bottom of the **Summary** panel on the right side of the screen choose **Launch instance**

You will see a Success message.

17. Choose **View all instances**

The instance will first appear in the *Pending* state, which means it is being launched. The state will then change to *Running*, which indicates that the instance has started booting. It takes a few minutes for the instance to boot.

18. Select the **Web Server 1** instance, and review the information in the **Details** tab that displays in the lower pane.

Notice that the instance has a **Public IPv4 address**. You can use this IP address to communicate with the instance from the internet.

19. Before you continue, wait for your instance to display the following:
- o **Instance state:** *Running*

- o **Status check:** 2/2 checks passed

This may take a few minutes. Choose the refresh icon at the top of the page every 30 seconds or so to more quickly become aware of the latest status of the instance.

Task 9. Access your EC2 instance

When you launched your EC2 instance, you provided a script that installed a web server and created a simple webpage. In this task, you will try to access the content from the web server.

20. From the **Details** tab, copy the **Public IPv4 address** value of your instance to your clipboard.

Note: A *public* address means that the instance can be reached from the internet. Each instance that receives a public IP address is also given an external DNS hostname; for example, `ec2-xxx-xxx-xxx-xxx.compute-1.amazonaws.com`. AWS resolves an external DNS hostname to the *public* IP address of the instance when communication comes from outside its VPC. When communication comes from inside its VPC, the DNS hostname is resolved to the *private* IPv4 address.

21. Open a new tab in your web browser, paste the public IP address you just copied, and press **Enter**.

The webpage does not load. You must update the security group to be able to access the page.

Task 10. Update the security group

You are not able to access your web server because the security group is not permitting inbound traffic on port 80, which is used for HTTP web requests. In this task, you update the security group.

22. Return to the **EC2 Management Console** browser tab.

23. In the left navigation pane, under **Network & Security**, choose **Security Groups**.

24. Select the **Web Server** security group, which you created when launching your EC2 instance.

25. In the lower pane, choose the **Inbound rules** tab.

Task 11. Create an inbound rule

26. Choose **Edit inbound rules**, and then choose **Add rule**.

27. Configure the following:

- o **Type:** HTTP
- o **Source:** Anywhere-IPv4
- o Choose **Save rules**

The new inbound HTTP rule creates an entry for IPv4 IP (0.0.0.0/0) and IPv6 IP addresses (::/0).

Task 12. Test the rule

28. Return to the tab that you used to try to connect to the web server.

29. Refresh the page.

The page should display the message *Hello World!*