

As we know for creating an ec2 instance we need four things.

1. Key
2. Subnet id
3. Security Group id
4. Amazon Machine Image ID.

Let us first create a key using amazon cli

```
aws ec2 create-key-pair --key-name ec2-key --query 'KeyMaterial' --output text > ec2-key.pem
```

We will be using the same key to create 5 EC2 instances.

The screenshot shows a Mac desktop with a Chrome browser window open. The browser's address bar displays the URL `awsacademy.instructure.com/courses/14374/modules/items/1212294`. The main content of the browser is a terminal window showing the following AWS CLI session:

```
ddd_v1_w_0sB_1130261@runweb53227:~$ aws ec2 create-key-pair --key-name ec2-key --query 'KeyMaterial' --output text > ec2-key.pem
ddd_v1_w_0sB_1130261@runweb53227:~$ aws ec2 describe-key-pairs
{
    "KeyPairs": [
        {
            "KeyPairId": "key-0e13647f141a2b241",
            "KeyFingerprint": "5ec2:f1:47:c5:78:72:68:39:22:54:69:52:db:4a:f2:84:65:42:c9",
            "KeyName": "ec2-key",
            "Tags": []
        },
        {
            "KeyPairId": "key-0e74d646a74a3b4a9",
            "KeyFingerprint": "c4:87:4e:06:a9:3c:88:85:a7:f8:8a:32:b2:6f:e8:26:09:d3:d9:59",
            "KeyName": "vockey",
            "Tags": []
        }
    ]
}
ddd_v1_w_0sB_1130261@runweb53227:~$
```

To the right of the terminal window, there is a sidebar with the title "Learner Lab - Foundational Level". It contains several hyperlinks related to the environment:

- Environment Overview
- Environment Navigation
- Access the AWS Management Console
- Region restriction
- Service usage and other restrictions
- Using the terminal in the browser
- Running AWS CLI commands

Here we can see we have created the key. Let us give read-only permission to this key.

The screenshot shows a Chrome browser window with multiple tabs open. The active tab is 'awsacademy.instructure.com/courses/14374/modules/items/1212294'. The page displays the AWS Learner Lab interface, specifically the 'Learner Lab - Foundational Services' module. On the left, a sidebar lists 'Account', 'Dashboard', 'Courses', 'Calendar', 'Inbox', 'History', and 'Help'. The main content area shows a terminal window with the following command and its output:

```
ddd_v1_w_0sB_1130261@runweb53227:~$ aws ec2 create-key-pair --key-name ec2-key --query 'KeyMaterial' --output text > ec2-key.pem
ddd_v1_w_0sB_1130261@runweb53227:~$ aws ec2 describe-key-pairs
{
  "KeyPairs": [
    {
      "KeyPairId": "key-0e13647f141a2b241",
      "KeyFingerprint": "5:e:c2:f1:47:c5:78:72:68:39:22:54:69:52:db:4a:f2:84:65:42:c9",
      "KeyName": "ec2-key",
      "Tags": []
    },
    {
      "KeyPairId": "key-0e74d646a74a3b4aa",
      "KeyFingerprint": "c4:87:4e:60:a9:3c:88:85:a7:f8:8a:32:b2:6f:e8:26:09:d3:d9:59",
      "KeyName": "vockey",
      "Tags": []
    }
  ]
}
ddd_v1_w_0sB_1130261@runweb53227:~$ chmod 400 ec2-key.pem
ddd_v1_w_0sB_1130261@runweb53227:~$
```

To the right of the terminal, there is a large box titled 'Learner Lab - Foundational Level' containing several links: 'Environment Overview', 'Environment Navigation', 'Access the AWS Management Console', 'Region restriction', 'Service usage and other restrictions', 'Using the terminal in the browser', and 'Running AWS CLI commands'. Below the terminal, there are 'Previous' and 'Next' navigation buttons.

2. We can create a new subnet and use that subnet id or we can use the existing subnets.
For this lab, we will use the existing subnets present in the us-east-1a zone.
subnet-0a1c2a85e00ef6e9c

The screenshot shows a Chrome browser window with multiple tabs open. The active tab is 'us-east-1.console.aws.amazon.com/vpc/home?region=us-east-1#subnets'. The page displays the AWS VPC Subnets management interface. On the left, a sidebar shows 'New VPC Experience' and lists 'VPC Dashboard', 'EC2 Global View New', 'Filter by VPC:', 'Select a VPC', 'VIRTUAL PRIVATE CLOUD', 'Your VPCs', 'Subnets' (which is selected), 'Route Tables', 'Internet Gateways', 'Egress Only Internet Gateways', 'Carrier Gateways', 'DHCP Options Sets', 'Elastic IPs', 'Managed Prefix Lists', 'Endpoints', 'Endpoint Services', and 'NAT Gateways'. The main content area shows a table of subnets:

Name	Subnet ID	State	VPC	IPv4 CIDR
subnet-02d6e208cf471bfd2	Available	vpc-0f19a97c7eacce15	172.31.0.0/20	
subnet-0b7f166dbb15b1487	Available	vpc-0f19a97c7eacce15	172.31.48.0/20	
subnet-0508013b0ae61df7d	Available	vpc-0f19a97c7eacce15	172.31.80.0/20	
subnet-0a1c2a85e00ef6e9c	Available	vpc-0f19a97c7eacce15	172.31.16.0/20	
subnet-02c7f55115eed90f	Available	vpc-0f19a97c7eacce15	172.31.32.0/20	
subnet-0f7281b43aed347a6	Available	vpc-0f19a97c7eacce15	172.31.64.0/20	

Below the table, there is a section for 'subnet-0a1c2a85e00ef6e9c' with tabs for 'Details', 'Flow logs', 'Route table', 'Network ACL', 'CIDR reservations', 'Sharing', and 'Tags'. At the bottom of the page, there are links for 'Privacy', 'Terms', and 'Cookie preferences'.

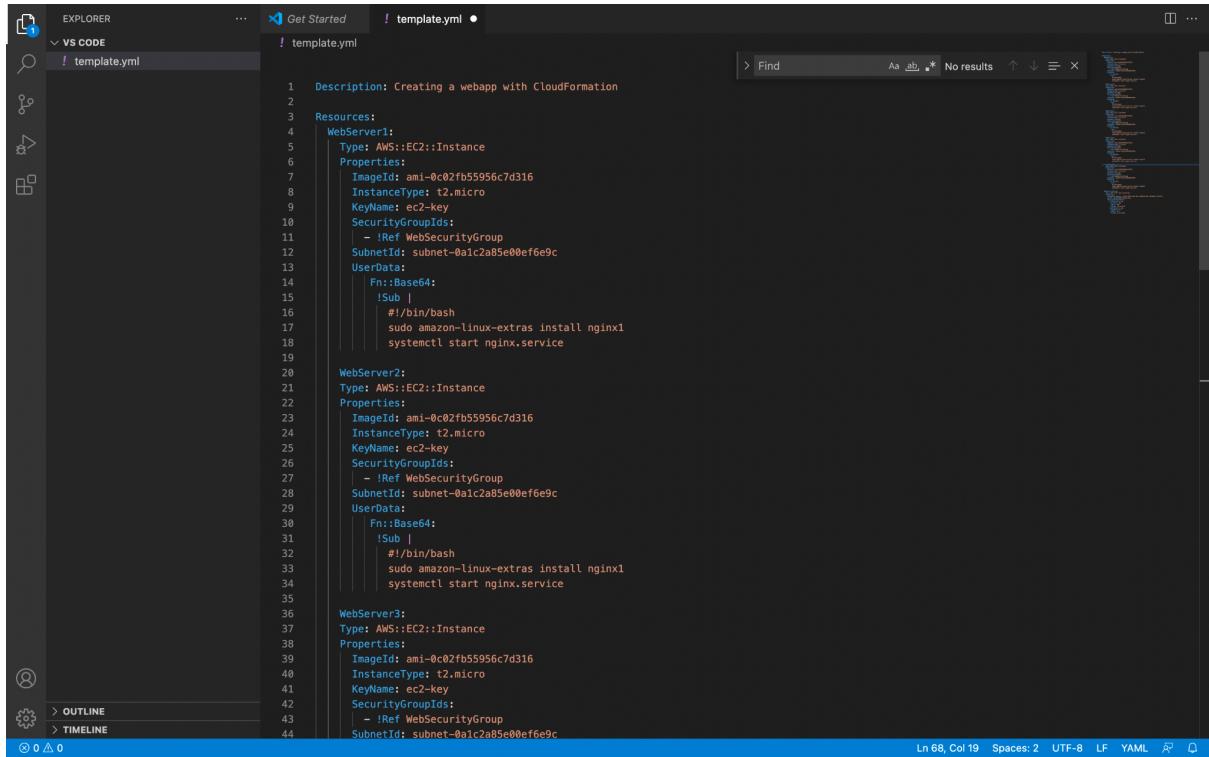
3. We will create a new security group using the cloud formation template so for the time being we will ignore it. We will continue this while creating a cloud formation template for ec2.

4. Amazon Machine Image Id: We will log in to the management console and copy the machine image id. For this lab, we will use ami-0c02fb55956c7d316 id.

Creating an EC2 instance and Security group using Cloud formation

Now we will create our ec2 template and security group using Cloud Formation Template. One interesting thing we are doing here is that we have mentioned the script that will be executed while installing the ec2 instance.

This script will install Nginx and start the server for us, so we don't need to log in to the server and do all these activities. We have to just update the Html pages so that they will be distinguishable.



The screenshot shows the Visual Studio Code interface with the 'template.yml' file open in the editor. The code defines three AWS::EC2::Instance resources: WebServer1, WebServer2, and WebServer3. Each instance has specific properties like ImageId, InstanceType, KeyName, and SecurityGroupIds. The UserData section for each instance contains a shell script that installs Nginx and starts the service. The script uses Fn::Base64 to encode the commands.

```
1  Description: Creating a webapp with CloudFormation
2
3  Resources:
4    WebServer1:
5      Type: AWS::EC2::Instance
6      Properties:
7        ImageId: ami-0c02fb55956c7d316
8        InstanceType: t2.micro
9        KeyName: ec2-key
10       SecurityGroupIds:
11         - !Ref WebSecurityGroup
12       SubnetId: subnet-0a1c2a85e00ef6e9c
13       UserData:
14         Fn::Base64:
15           !Sub |
16             #!/bin/bash
17             sudo amazon-linux-extras install nginx1
18             systemctl start nginx.service
19
20   WebServer2:
21     Type: AWS::EC2::Instance
22     Properties:
23       ImageId: ami-0c02fb55956c7d316
24       InstanceType: t2.micro
25       KeyName: ec2-key
26       SecurityGroupIds:
27         - !Ref WebSecurityGroup
28       SubnetId: subnet-0a1c2a85e00ef6e9c
29       UserData:
30         Fn::Base64:
31           !Sub |
32             #!/bin/bash
33             sudo amazon-linux-extras install nginx1
34             systemctl start nginx.service
35
36   WebServer3:
37     Type: AWS::EC2::Instance
38     Properties:
39       ImageId: ami-0c02fb55956c7d316
40       InstanceType: t2.micro
41       KeyName: ec2-key
42       SecurityGroupIds:
43         - !Ref WebSecurityGroup
44       SubnetId: subnet-0a1c2a85e00ef6e9c
```

The screenshot shows the VS Code interface with the 'VS CODE' tab selected in the Explorer sidebar. The main editor area displays a CloudFormation template named 'template.yml'. The template defines three AWS::EC2::Instance resources: 'WebServer3', 'WebServer4', and 'LoadBalancer'. Each instance has specific properties like ImageId, InstanceType, KeyName, SecurityGroupIds, and UserData. The 'UserData' block for each instance contains a shell script to install nginx and start the service. The 'LoadBalancer' resource is also defined with its own properties. The bottom status bar shows the file is at Line 68, Column 17, with 2 spaces, in UTF-8 encoding, and is a YAML file.

```
template.yml
36     WebServer3:
37       Type: AWS::EC2::Instance
38       Properties:
39         ImageId: ami-0c02fb55956c7d316
40         InstanceType: t2.micro
41         KeyName: ec2-key
42         SecurityGroupIds:
43           - !Ref WebSecurityGroup
44         SubnetId: subnet-0a1c2a85e00ef6e9c
45         UserData:
46           Fn::Base64:
47             !Sub |
48               #!/bin/bash
49               sudo amazon-linux-extras install nginx1
50               systemctl start nginx.service
51
52     WebServer4:
53       Type: AWS::EC2::Instance
54       Properties:
55         ImageId: ami-0c02fb55956c7d316
56         InstanceType: t2.micro
57         KeyName: ec2-key
58         SecurityGroupIds:
59           - !Ref WebSecurityGroup
60         SubnetId: subnet-0a1c2a85e00ef6e9c
61         UserData:
62           Fn::Base64:
63             !Sub |
64               #!/bin/bash
65               sudo amazon-linux-extras install nginx1
66               systemctl start nginx.service
67
68     LoadBalancer:
69       Type: AWS::EC2::Instance
70       Properties:
71         ImageId: ami-0c02fb55956c7d316
72         InstanceType: t2.micro
73         KeyName: ec2-key
74         SecurityGroupIds:
75           - !Ref WebSecurityGroup
76         SubnetId: subnet-0a1c2a85e00ef6e9c
77         UserData:
78           Fn::Base64:
79             !Sub |
80               #!/bin/bash
81               sudo amazon-linux-extras install nginx1
82               systemctl start nginx.service
83
84     WebSecurityGroup:
85       Type: AWS::EC2::SecurityGroup
86       Properties:
87         GroupDescription: "Allow HTTP and SSH inbound and outbound traffic"
88         VpcId: vpc-0f19a97c7eacce15
89         SecurityGroupIngress:
90           - IpProtocol: tcp
91             FromPort: 80
92             ToPort: 80
93             CidrIp: 0.0.0.0/0
94           - IpProtocol: tcp
95             FromPort: 22
96             ToPort: 22
97             CidrIp: 0.0.0.0/0
```

This screenshot shows the same VS Code environment as the previous one, but the CloudFormation template has been modified. The 'LoadBalancer' resource definition from the first screenshot has been removed. The rest of the template, including the three EC2 instances and the security group, remains the same. The bottom status bar indicates the file is at Line 68, Column 17, with 2 spaces, in UTF-8 encoding, and is a YAML file.

```
template.yml
66     |     systemctl start nginx.service
67
68     LoadBalancer:
69       Type: AWS::EC2::Instance
70       Properties:
71         ImageId: ami-0c02fb55956c7d316
72         InstanceType: t2.micro
73         KeyName: ec2-key
74         SecurityGroupIds:
75           - !Ref WebSecurityGroup
76         SubnetId: subnet-0a1c2a85e00ef6e9c
77         UserData:
78           Fn::Base64:
79             !Sub |
80               #!/bin/bash
81               sudo amazon-linux-extras install nginx1
82               systemctl start nginx.service
83
84     WebSecurityGroup:
85       Type: AWS::EC2::SecurityGroup
86       Properties:
87         GroupDescription: "Allow HTTP and SSH inbound and outbound traffic"
88         VpcId: vpc-0f19a97c7eacce15
89         SecurityGroupIngress:
90           - IpProtocol: tcp
91             FromPort: 80
92             ToPort: 80
93             CidrIp: 0.0.0.0/0
94           - IpProtocol: tcp
95             FromPort: 22
96             ToPort: 22
97             CidrIp: 0.0.0.0/0
```

Once the template is created we will log in to Amazon Management Console and go to CloudFormation and click on create stack button and select with new resources button.

The screenshot shows the AWS CloudFormation console. On the left, there's a sidebar with options like Stacks, StackSets, Exports, Designer, Registry (Public extensions, Activated extensions, Publisher), and Feedback. The main area is titled 'CloudFormation > Stacks'. It displays a table with one row:

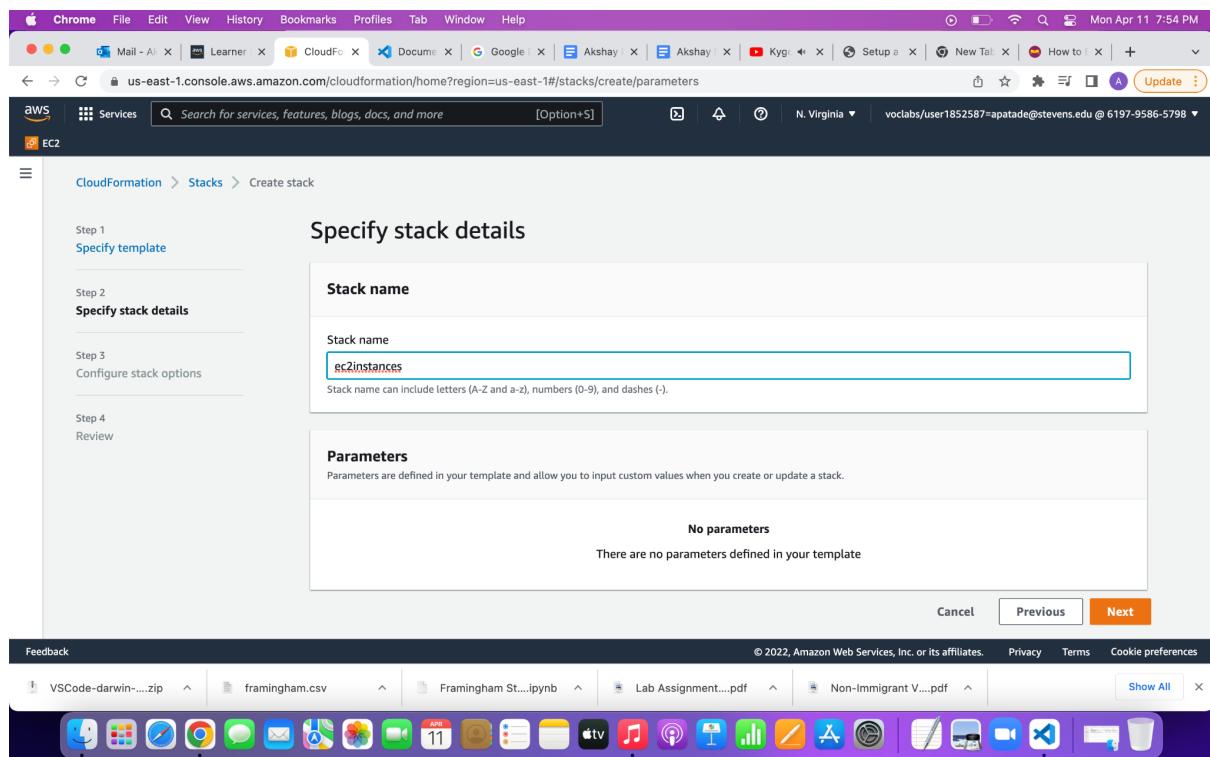
Stack name	Status	Created time	Description
c48997a7248241794860t1w619795865798	CREATE_COMPLETE	2022-03-29 15:29:48 UTC-0400	foundational Learner Lab template

At the bottom of the page, there's a feedback section and a standard browser footer.

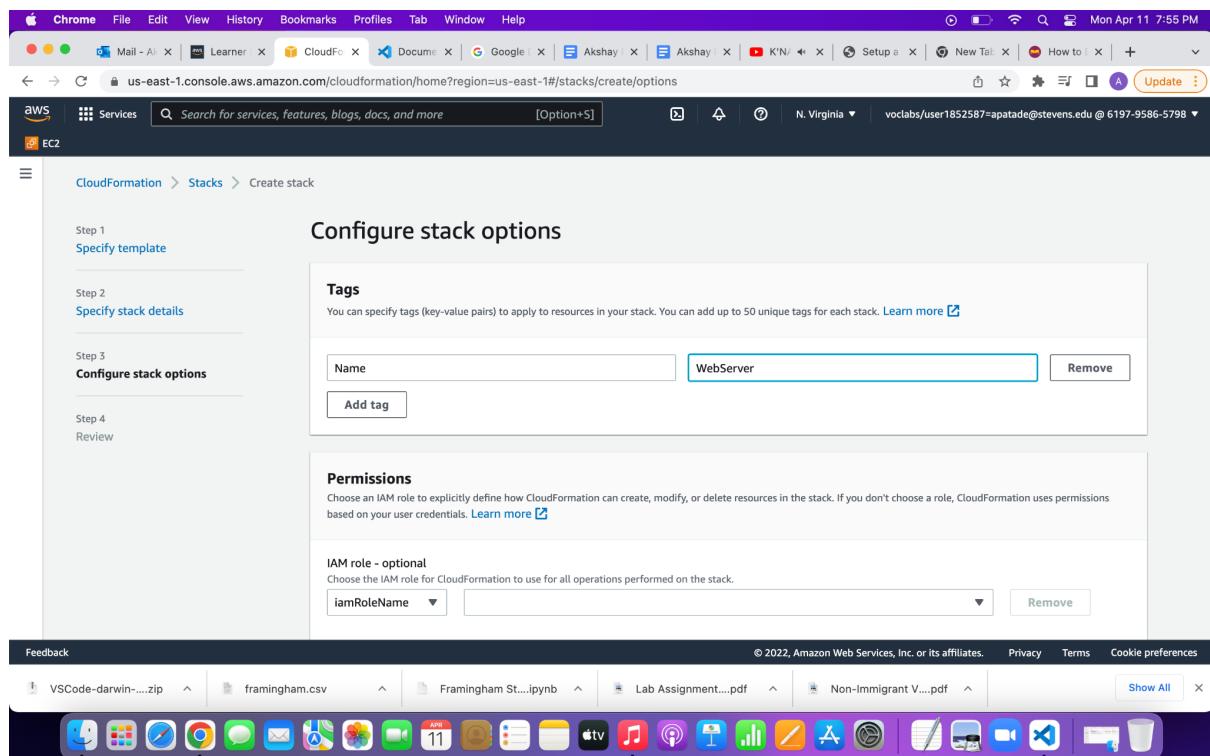
Next, click on the upload a template file radio button and then upload your template file, and then click on the upload button.

The screenshot shows the 'Create Stack' wizard. The first step is 'Specify template'. On the left, there are tabs for Step 1 (Specify template), Step 2 (Specify stack details), Step 3 (Configure stack options), and Step 4 (Review). The 'Specify template' tab is active. The main area has two sections: 'Prerequisite - Prepare template' and 'Specify template'. In the 'Prerequisite - Prepare template' section, there are three radio buttons: 'Template is ready' (selected), 'Use a sample template', and 'Create template in Designer'. In the 'Specify template' section, there's a note about what a template is, a 'Template source' dropdown (set to 'Amazon S3 URL'), and a 'Upload a template file' section with a 'Choose file' button and a file input field containing 'template.yml'. At the bottom, there's a 'Next Step' button.

Next, write a name for your stack and then click on next. I have mentioned ec2instances.



You can configure your stack. For this lab work, we will just hit next.



Review your stack and hit create stack button.

Now go to ec2 instance and security group to check whether the instances and security group are successfully created or not.

Instances (7) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm stat
WebServer1	i-097402187bee11acc	Running	t2.micro	2/2 checks passed	No alarms
WebServer2	i-08472b5024bb4b57e	Running	t2.micro	2/2 checks passed	No alarms
WebServer3	i-0648a360e08a139f7	Running	t2.micro	2/2 checks passed	No alarms
WebServer4	i-0e60b12a7464cb29a	Running	t2.micro	2/2 checks passed	No alarms
LoadBalancer	i-0f10f5c547215ffdb	Running	t2.micro	2/2 checks passed	No alarms

Select an instance

Security Groups (1/2) Info

Name	Security group ID	Security group name	VPC ID	Description	Owner
sg-0df9a4c5626a20ff	default	vpc-0f19a97c7eacce15	default VPC security gr...	61979586	
Web App Securi...	sg-0c0dfc2efb4f6501d	ec2instances-WebSecu...	vpc-0f19a97c7eacce15	Allow HTTP and SSH i...	61979586

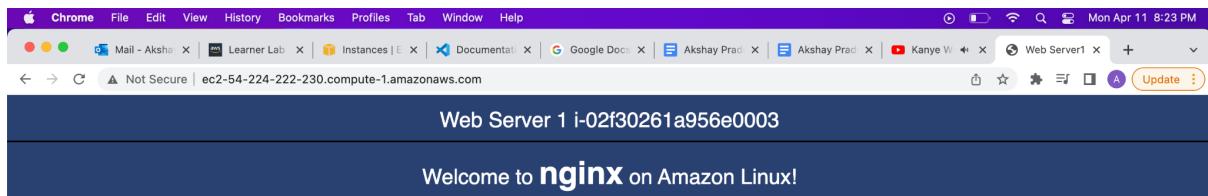
sg-0c0dfc2efb4f6501d - ec2instances-WebSecurityGroup-D1D736MB580K

Details **Inbound rules** **Outbound rules** **Tags**

You can now check network connectivity with Reachability Analyzer

Run Reachability Analyzer

Here we can see that our instances and the security group are successfully created. Now we update the HTML of each server so that it is distinguishable and we will check whether we can access the server or not.



This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

Website Administrator

This is the default `index.html` page that is distributed with **nginx** on Amazon Linux. It is located in `/usr/share/nginx/html`.

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

NGINX



Name	Instance ID	Instance state	Instance type	Status check	Alarm stat
WebServer1	i-097402187bee11acc	Running	t2.micro	2/2 checks passed	No alarms
WebServer2	i-08472b5024bb4b57e	Running	t2.micro	2/2 checks passed	No alarms
WebServer3	i-0648a360e08a139f7	Running	t2.micro	2/2 checks passed	No alarms
WebServer4	i-0e60b12a7464cb29a	Running	t2.micro	2/2 checks passed	No alarms
LoadBalancer	i-0f0f5c547215ff0db	Running	t2.micro	2/2 checks passed	No alarms

Web Server 2 i-08472b5024bb4b57e

Welcome to nginx on Amazon Linux!

This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

Website Administrator

This is the default `index.html` page that is distributed with **nginx** on Amazon Linux. It is located in `/usr/share/nginx/html`.

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

NGINX



Instances (1/7) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm stat
WebServer1	i-097402187hee11acc	Running	t2.micro	2/2 checks passed	No alarms
WebServer2	i-08472b5024bb4b57e	Running	t2.micro	2/2 checks passed	No alarms
WebServer3	i-0648a360e08a139f7	Running	t2.micro	2/2 checks passed	No alarms
WebServer4	i-0e60b12a7464cb29a	Running	t2.micro	2/2 checks passed	No alarms
LoadBalancer	i-0f10f5c547215ffdb	Running	t2.micro	2/2 checks passed	No alarms

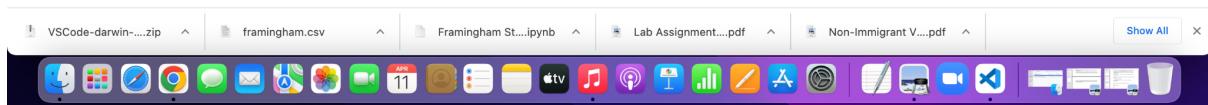
Instance: i-08472b5024bb4b57e (WebServer2)

Instance ID i-08472b5024bb4b57e (WebServer2)	Public IPv4 address 52.205.255.222 open address	Private IPv4 addresses 172.31.20.132
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-52-205-255-222.compute-1.amazonaws.com open address
Hostname type IP name: ip-172-31-20-132.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-20-132.ec2.internal	Answer private resource DNS name -

This screenshot shows a Chrome browser window with multiple tabs open. The active tab displays the message "Welcome to nginx on Amazon Linux!" on a dark blue header bar. Below the header, a paragraph of text reads: "This page is used to test the proper operation of the nginx HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly." A "Website Administrator" section follows, containing instructions about the default index.html page and how to edit the configuration file.

Website Administrator

This is the default `index.html` page that is distributed with nginx on Amazon Linux. It is located in `/usr/share/nginx/html`. You should now put your content in a location of your choice and edit the `root` configuration directive in the nginx configuration file `/etc/nginx/nginx.conf`.



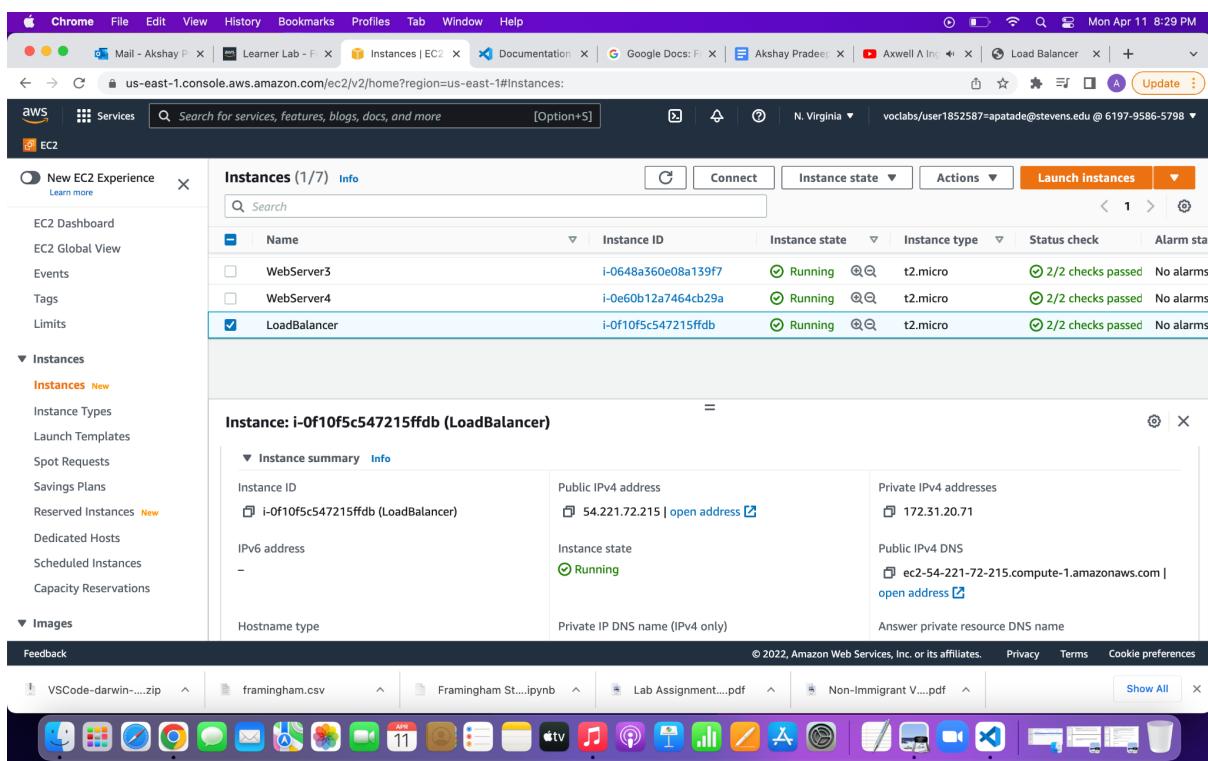
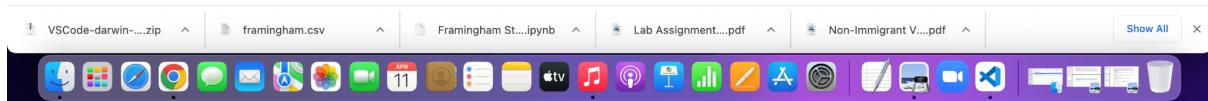
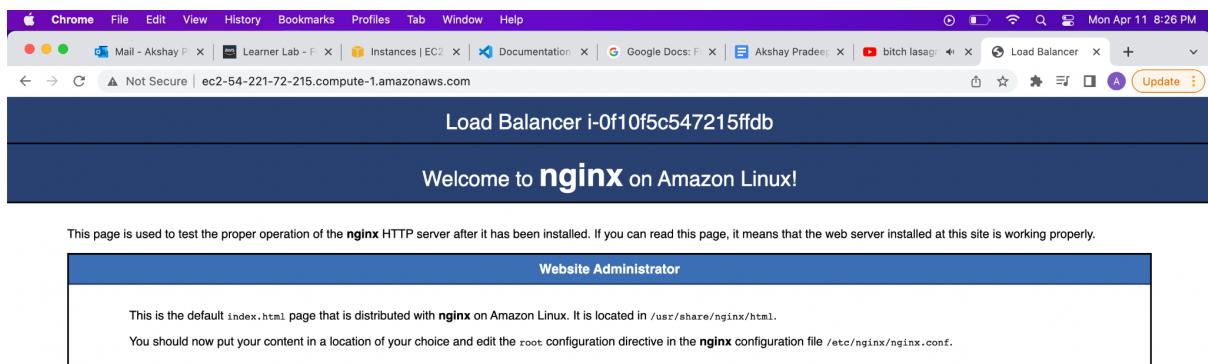
This screenshot shows the AWS EC2 Instances page in a browser. The left sidebar lists navigation options like EC2 Dashboard, EC2 Global View, Events, Tags, Limits, Instances (selected), and Images. The main content area shows a table titled "Instances (1/7) Info" with one item: "WebServer3". The instance details include its ID (i-0648a360e08a139f7), state (Running), type (t2.micro), and status (2/2 checks passed). The public IPv4 address is listed as 34.227.61.217. The right side of the page includes links for "Private IPv4 addresses" (172.31.24.180), "Public IPv4 DNS" (ec2-34-227-61-217.compute-1.amazonaws.com), and "Answer private resource DNS name" (IP address).

Name	Instance ID	Instance state	Instance type	Status check	Alarm stat
WebServer1	i-097402187bee11acc	Running	t2.micro	2/2 checks passed	No alarms
WebServer2	i-08472b5024bb4b57e	Running	t2.micro	2/2 checks passed	No alarms
<input checked="" type="checkbox"/> WebServer3	i-0648a360e08a139f7	Running	t2.micro	2/2 checks passed	No alarms
WebServer4	i-0e60b12a7464cb29a	Running	t2.micro	2/2 checks passed	No alarms
LoadBalancer	i-0f10f5c547215ffdb	Running	t2.micro	2/2 checks passed	No alarms

The screenshot shows a Chrome browser window with the address bar displaying "Not Secure | ec2-3-91-189-71.compute-1.amazonaws.com". The main content area shows the "Web Server 4 i-0e60b12a7464cb29a" page with the heading "Welcome to nginx on Amazon Linux!". Below the heading, a message states: "This page is used to test the proper operation of the nginx HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly." A "Website Administrator" section contains instructions: "This is the default index.html page that is distributed with nginx on Amazon Linux. It is located in /usr/share/nginx/html. You should now put your content in a location of your choice and edit the root configuration directive in the nginx configuration file /etc/nginx/nginx.conf." At the bottom center of the page is the "NGINX" logo.



The screenshot shows the AWS EC2 Instances page. The left sidebar includes options like "New EC2 Experience", "EC2 Dashboard", "EC2 Global View", "Events", "Tags", "Limits", "Instances" (selected), "Images", and "Feedback". The main content area displays a table titled "Instances (1/7) Info" with one item: "WebServer4" (Instance ID: i-0e60b12a7464cb29a, State: Running, Type: t2.micro). Below this, a detailed view for "Instance: i-0e60b12a7464cb29a (WebServer4)" is shown, including sections for "Instance summary" (with fields for Instance ID, Public IPv4 address, Private IPv4 addresses, and Public IPv4 DNS) and "Hostname type" (with a field for "Private IP DNS name (IPv4 only)"). The browser address bar shows "us-east-1.console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:".



We are able to access all the webservers via ssh and HTTP.

Now we will log in to the load balancer and change the configure file.

The screenshot shows a Mac desktop with a Chrome browser window open to an AWS Academy learner lab. The lab is titled "Learner Lab - Foundational Level". The main content area displays an Nginx configuration file with code like:

```
worker_processes auto;
error_log /var/log/nginx/error.log;
pid /run/nginx.pid;
...
upstream myapp {
    ...
}
server {
    listen 80;
    server_name myapp.com;
    location / {
        proxy_pass http://myapp;
    }
}
nginx.conf" 31L, 629B
```

The right sidebar provides a summary of the environment and links to AWS Management Console, Region restriction, Service usage and other restrictions, Using the terminal in the browser, and Running AWS CLI commands.

Now that we have made changes in our configuration file, we will restart the nginx and check whether the load balancing is working properly or not.

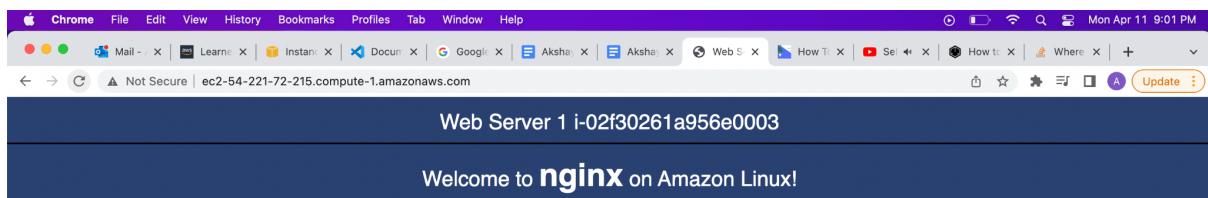
The screenshot shows the same AWS Academy learner lab after performing a system restart. The terminal output now shows the command:

```
[ec2-user@ip-172-31-20-71 nginx]$ sudo service nginx restart
```

The output indicates that the service was successfully restarted:

```
Redirecting to /bin/systemctl restart nginx.service
Job for nginx.service failed because the control process exited with error code. See "systemctl status nginx.service" and "journalctl -xe" for details.
```

The right sidebar continues to provide links to AWS resources.



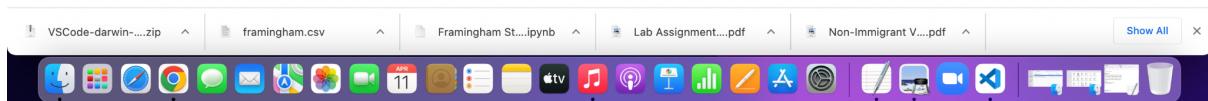
This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

Website Administrator

This is the default `index.html` page that is distributed with **nginx** on Amazon Linux. It is located in `/usr/share/nginx/html`.

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

NGINX



This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

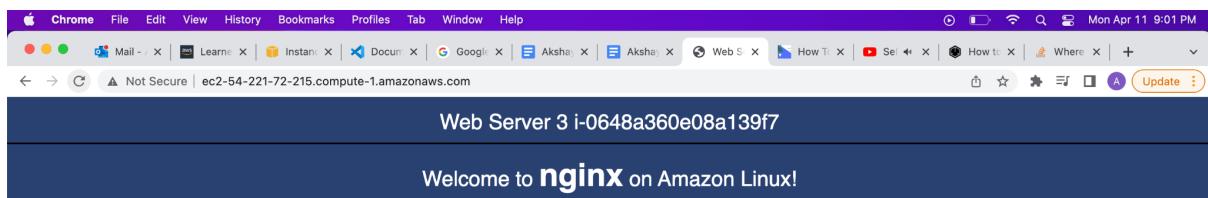
Website Administrator

This is the default `index.html` page that is distributed with **nginx** on Amazon Linux. It is located in `/usr/share/nginx/html`.

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

NGINX





This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

Website Administrator

This is the default `index.html` page that is distributed with **nginx** on Amazon Linux. It is located in `/usr/share/nginx/html`.

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

NGINX



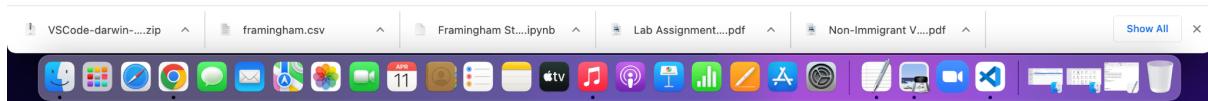
This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

Website Administrator

This is the default `index.html` page that is distributed with **nginx** on Amazon Linux. It is located in `/usr/share/nginx/html`.

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

NGINX



Template of Cloud Formation

Template.yml

Description: Creating a webapp with CloudFormation

Resources:

WebServer1:

Type: AWS::EC2::Instance

Properties:

ImageId: ami-0c02fb55956c7d316

InstanceType: t2.micro

KeyName: ec2-key

SecurityGroupIds:

- !Ref WebSecurityGroup

SubnetId: subnet-0a1c2a85e00ef6e9c

UserData:

Fn::Base64:

!Sub |

#!/bin/bash

sudo amazon-linux-extras install nginx1

systemctl start nginx.service

WebServer2:

Type: AWS::EC2::Instance

Properties:

ImageId: ami-0c02fb55956c7d316

InstanceType: t2.micro

KeyName: ec2-key

SecurityGroupIds:

- !Ref WebSecurityGroup

SubnetId: subnet-0a1c2a85e00ef6e9c

UserData:

Fn::Base64:

!Sub |

#!/bin/bash

sudo amazon-linux-extras install nginx1

systemctl start nginx.service

WebServer3:

Type: AWS::EC2::Instance

Properties:

ImageId: ami-0c02fb55956c7d316

InstanceType: t2.micro

KeyName: ec2-key

SecurityGroupIds:

- !Ref WebSecurityGroup

SubnetId: subnet-0a1c2a85e00ef6e9c

UserData:

Fn::Base64:

!Sub |

#!/bin/bash

sudo amazon-linux-extras install nginx1

systemctl start nginx.service

WebServer4:

Type: AWS::EC2::Instance

Properties:

ImageId: ami-0c02fb55956c7d316

InstanceType: t2.micro

KeyName: ec2-key

SecurityGroupIds:

- !Ref WebSecurityGroup

SubnetId: subnet-0a1c2a85e00ef6e9c

UserData:

```
Fn::Base64:
```

```
!Sub |
```

```
#!/bin/bash
```

```
sudo amazon-linux-extras install nginx1
```

```
systemctl start nginx.service
```

LoadBalancer:

Type: AWS::EC2::Instance

Properties:

ImageId: ami-0c02fb55956c7d316

InstanceType: t2.micro

KeyName: ec2-key

SecurityGroupIds:

- !Ref WebSecurityGroup

SubnetId: subnet-0a1c2a85e00ef6e9c

UserData:

```
Fn::Base64:
```

```
!Sub |
```

```
#!/bin/bash
```

```
sudo amazon-linux-extras install nginx1
```

```
systemctl start nginx.service
```

WebSecurityGroup:

Type: AWS::EC2::SecurityGroup

Properties:

GroupDescription: "Allow HTTP and SSH inbound and outbound traffic"

VpcId: vpc-0f19a97c7eaccce15

SecurityGroupIngress:

- IpProtocol: tcp

FromPort: 80

ToPort: 80

CidrIp: 0.0.0.0/0

- IpProtocol: tcp

FromPort: 22

ToPort: 22

CidrIp: 0.0.0.0/0