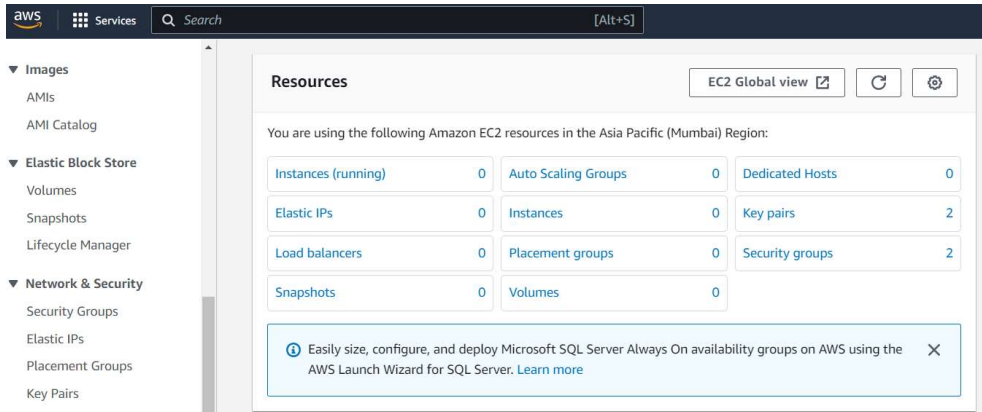


ASSIGNMENT – 10

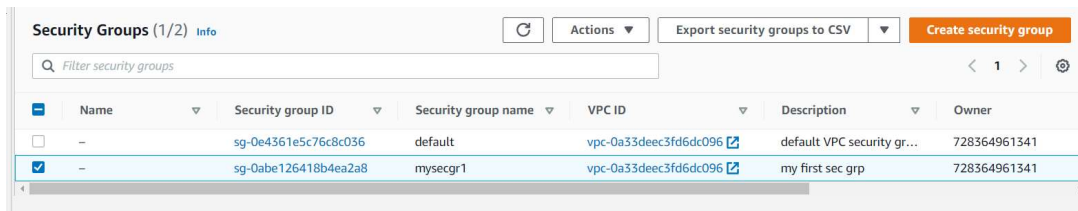
Problem Statement: Deploy project from GitHub to EC2 by creating new security group and user data.

Procedure:

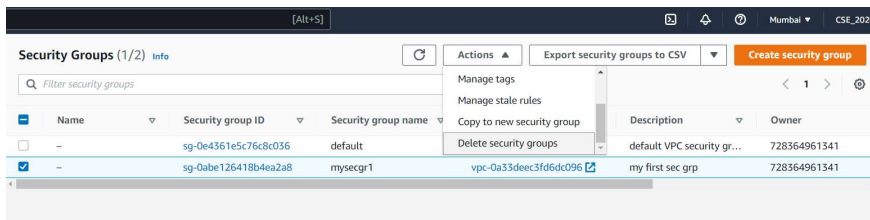
1. Sign in to your AWS account.
2. Go to your EC2 dashboard
3. Scroll down and Click on Security Groups option on the left side nav bar under Network & Security option.



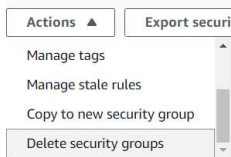
4. Select all the Security Groups other than the one named “default”.



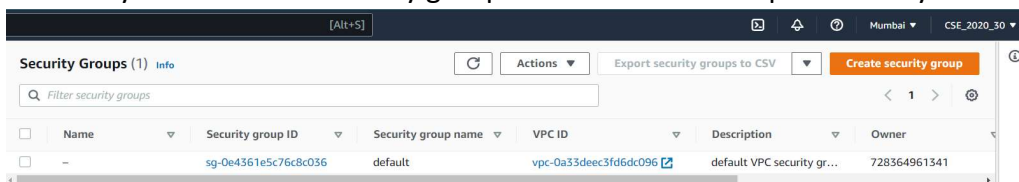
5. Then Click on the Actions button.



6. Scroll-Down the dropdown list until you find the “delete all security groups” option. Click on it.



7. Now only the “default” security group remains and we keep it that way.



8. Now click on the “Create Security Group” button.



9. Now start by giving a name to the security group and giving its description (anything).
Let the VPC remain unchanged.

EC2 > Security Groups > Create security group

Create security group [Info](#)

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [Info](#)
mysec1
Name cannot be edited after creation.

Description [Info](#)
mysec1

VPC [Info](#)
vpc-0a33deec3fd6dc096

10. Next, we will add Inbound Rules. Start adding by clicking the Add rule button. These include:

a) SSH

Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info	
SSH	TCP	22	Anywh... 0.0.0.0/0		Delete

b) HTTP

HTTP	TCP	80	Anywh... 0.0.0.0/0		Delete
------	-----	----	-----------------------	--	--------

c) HTTPS

HTTPS	TCP	443	Anywh... 0.0.0.0/0		Delete
-------	-----	-----	-----------------------	--	--------

d) Custom TCP

Custom TCP	TCP	4000	Anywh... 0.0.0.0/0		Delete
------------	-----	------	-----------------------	--	--------

The last one with custom TCP has a specific port range that we require to connect to our project. It has been specified in our index.js file (refer Ass9).

Now the final Inbound Rules section should look like this.

Inbound rules [Info](#)

Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info	
SSH	TCP	22	Anywh... 0.0.0.0/0		Delete
HTTP	TCP	80	Anywh... 0.0.0.0/0		Delete
HTTPS	TCP	443	Anywh... 0.0.0.0/0		Delete
Custom TCP	TCP	4000	Anywh... 0.0.0.0/0		Delete

Add rule

11. Next outbound rules and all other sections remain unchanged. Now Click on the create security group button.

Outbound rules [Info](#)

Type Info	Protocol Info	Port range Info	Destination Info	Description - optional Info	
All traffic	All	All	Custom 0.0.0.0/0		Delete

Add rule

Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

Add new tag
You can add up to 50 more tags

Cancel Create security group

12. Now go back to the security groups list and click on the security group ID of the newly created Security Group.

The screenshot shows the AWS Management Console for Security Groups. At the top, there's a header 'Security Groups (2)' with an 'Info' link and a refresh button. Below is a search bar 'Filter security groups'. A table lists two security groups: 'mysec1' with ID 'sg-0493398d43b761e55' and 'default' with ID 'sg-0e4361e5c76c8c036'. Below the table, a detailed view for 'mysec1' shows its ID, description, VPC ID, owner, and rule counts. The 'Inbound rules' tab is selected, showing a list of four rules: HTTPS (port 443), SSH (port 22), HTTP (port 80), and Custom TCP (port 4000). A 'Run Reachability Analyzer' button is visible at the top right of the rules section.

After clicking we can view the inbound rules that we added during its creation.

13. Now we go to the instances section from the left side nav bar.

14. Now we Create a new EC2 instance. Click on the Launch Instance button.

The screenshot shows the 'Launch instances' page in the AWS EC2 console. It features a search bar 'Find instance by attribute or tag (case-sensitive)', a table with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4 DNS. Below the table, a message states 'No instances' and 'You do not have any instances in this region'. At the top right, there are buttons for 'Connect', 'Instance state', 'Actions', and a prominent orange 'Launch instances' button.

Now,

a) Give the name

This is the first step of the 'Launch an instance' wizard. It has a title 'Launch an instance' and a sub-header 'Name and tags'. A text input field contains 'debserver1'. To the right of the field is a link 'Add additional tags'.

b) Select Ubuntu as OS.

This screenshot shows the 'Application and OS Images' section. It includes a search bar 'Search our full catalog including 1000s of application and OS images'. Below is a 'Quick Start' section with a row of image tiles for Amazon Linux, macOS, Ubuntu, Windows, Red Hat, and SUSE. The Ubuntu tile is highlighted. To the right is a link 'Browse more AMIs'.

c) Select a keypair or generate a new one if none is available.

This screenshot shows the 'Key pair (login)' section. It has a sub-header 'Key pair (login)'. A text input field for 'Key pair name - required' contains 'debkey2'. To the right of the field is a link 'Create new key pair'.

d) Then under Network settings select the Select Existing Security Group option.

This screenshot shows the 'Network settings' section. It has a sub-header 'Network settings'. Under 'Network', it shows 'vpc-0a33deec3fd6dc096'. Under 'Subnet', it shows 'No preference (Default subnet in any availability zone)'. Under 'Auto-assign public IP', it shows 'Enable'. Under 'Firewall (security groups)', there are two radio buttons: 'Create security group' (unselected) and 'Select existing security group' (selected). Below these is a dropdown menu 'Select security groups' and a link 'Compare security group rules'.

- e) Now under the security groups dropdown menu select the one we just created.

Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☐ Create security group ☒ Select existing security group

Security groups Info
Select security groups

mysec1 sg-0493398d43b761e55
VPC: vpc-0a33deec3fd6dc096

default sg-0e4361e5c76c8c036
VPC: vpc-0a33deec3fd6dc096

Compare security group rules

Advanced

It should look like this.....

Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☐ Create security group ☒ Select existing security group

Security groups Info
Select security groups

mysec1 sg-0493398d43b761e55 X
VPC: vpc-0a33deec3fd6dc096

Compare security group rules

Advanced

- f) Now scroll down and click on the Advanced Details option.

► Advanced details Info

- g) Now again scroll-down to the newly appeared sub-sections until you find User Data section.

User data - optional Info
Enter user data in the field.

- h) Write the following commands in the given box. Remember this user data is given to execute the given commands once the server starts. So essentially, we can provide all commands that we entered in our Assignment 9 previously and execute them without connecting to our server itself!! They will be executed sequentially.

Within the User data field, enter the following codes.

```
#!/bin/bash
apt-get update
apt-get install -y nginx
systemctl start nginx
systemctl enable nginx
apt-get install -y git
curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -
apt-get install -y nodejs
cd /home/ubuntu
git clone https://github.com/exd35-avi/Awsproject2.git
cd Awsproject2
npm install
nodeindex.js
```

- i) Now we click on the launch instance button.

User data - optional [Info](#)

Enter user data in the field.

```
#!/bin/bash
apt-get update
apt-get install -y nginx
systemctl start nginx
systemctl enable nginx
apt-get install -y git
curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -
apt-get install -y nodejs
cd /home/ubuntu
git clone https://github.com/sohail3080/Awsproject2.git
cd Awsproject2
npm install
node index.js
```

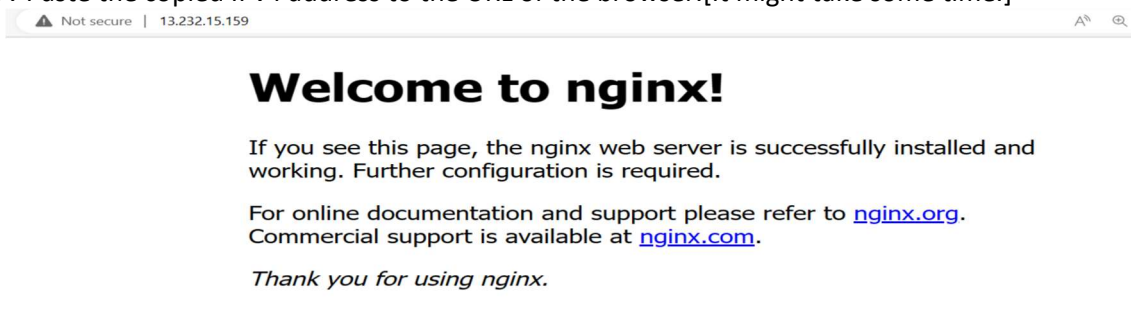
15. Now we Click on the 'Instance Id' link of our newly created server in our Instances list.

Instances (1) Info							
<input type="text" value="Find instance by attribute or tag (case-sensitive)"/>							
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input type="checkbox"/>	debserver1	i-0a6ab24417f81fffb	Running	t2.micro	Initializing	No alarms	ap-south-1a

16. Copy the IPv4 address.

Instance summary for i-0a6ab24417f81fffb (debserver1) Info		
Updated less than a minute ago		
Instance ID i-0a6ab24417f81fffb (debserver1)	Public IPv4 address 3.110.134.34 open address	Private IPv4 addresses 172.31.41.246
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-3-110-134-34.ap-south-1.compute.amazonaws.com open address

17. Paste the copied IPv4 address to the URL of the browser.[It might take some time.]



18. Add the Port number to the URL. Here, we have taken 4000 as port number. [It might take some time.]



We have successfully Deployed a project from GitHub to EC2 by creating a new Security group and User Data.