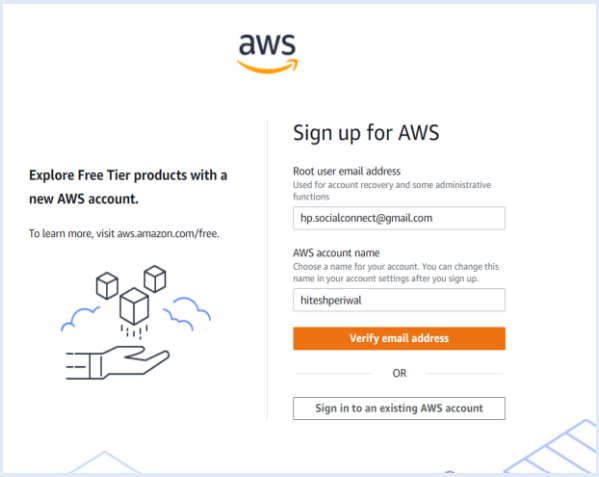
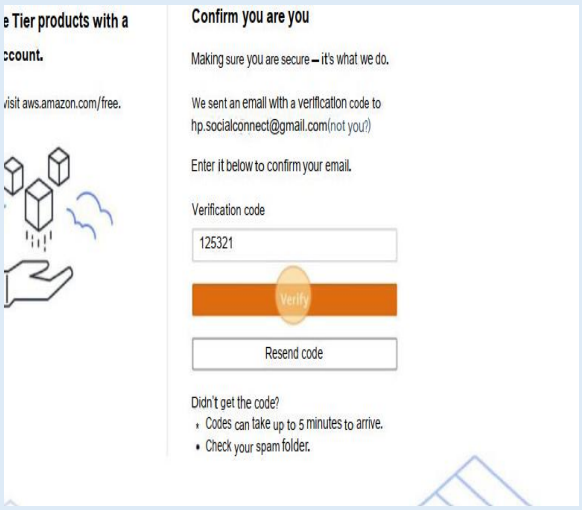
CREATING AN AWS ACCOUNT

1. **Visit aws.amazon.com and click on "Create an AWS Account"**

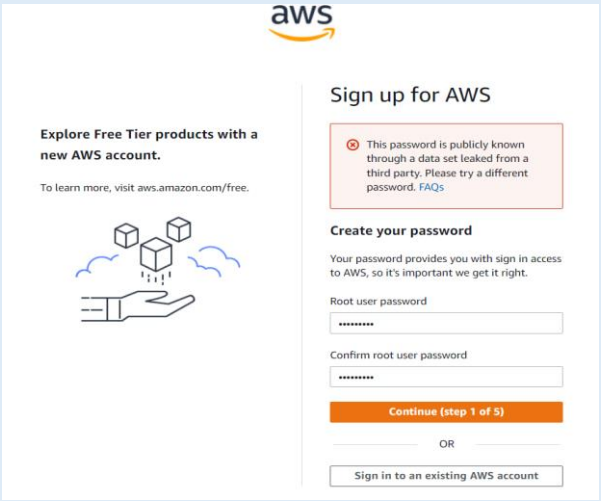
2. **Enter the root user email address ,AWS account name and click on "Verify email address"**



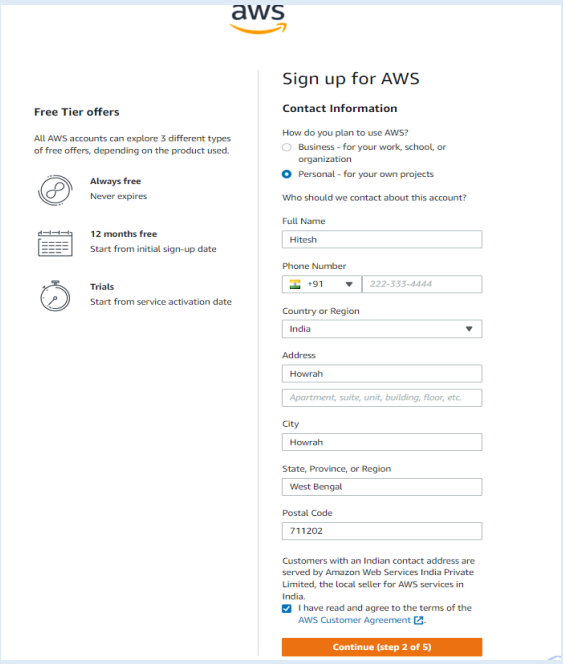
3. **Enter the verification code received in the given email address and click on "Verify"**



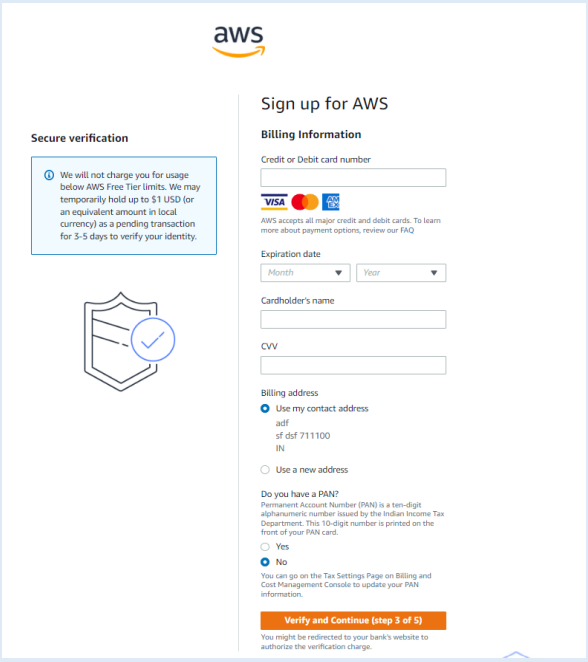
4. **Set your account's password**



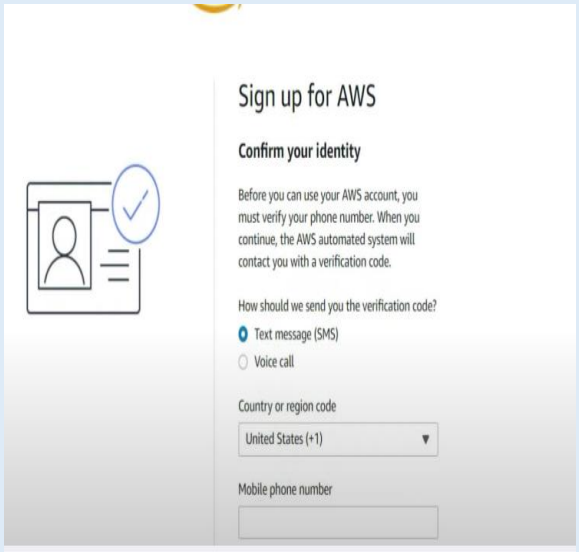
5. Fill in your contact details and click on continue



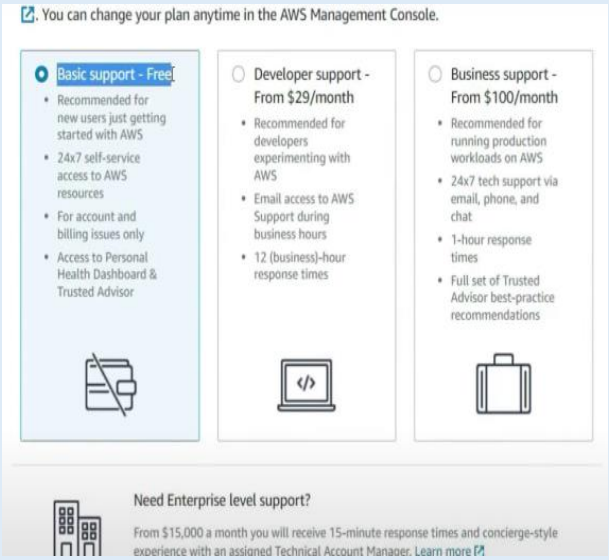
6. Enter your billing information and click on “Verify and Continue”



7. Confirm and verify your phone number



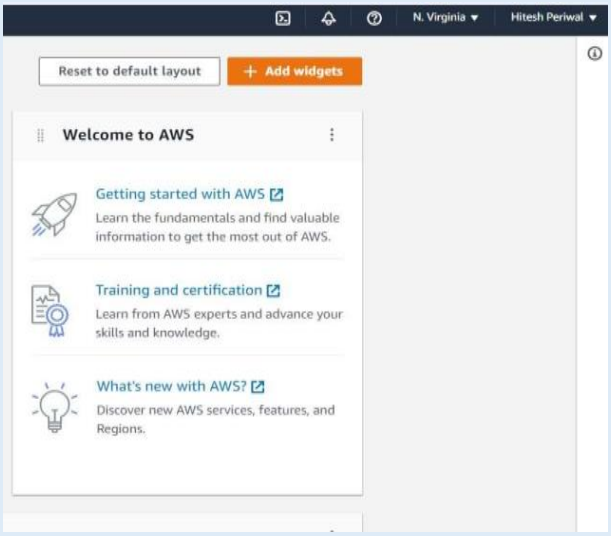
8. Choose Basic Support plan and hence your AWS account is created



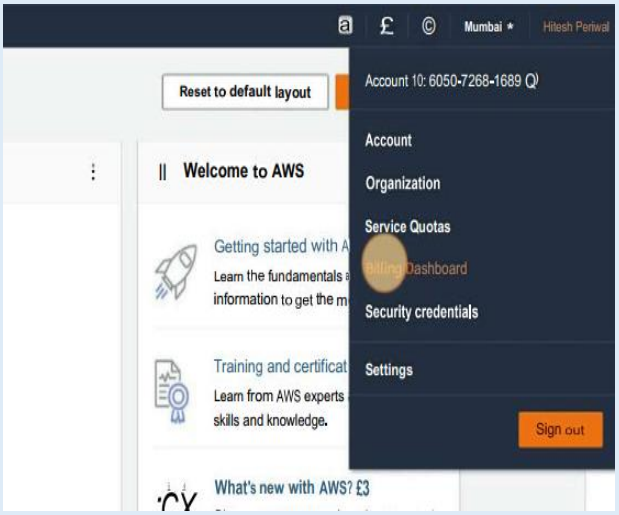
CONFIGURE A BUDGET AND SETTING A THRESHOLD VALUE

1. Visit aws.amazon.com and Sign in

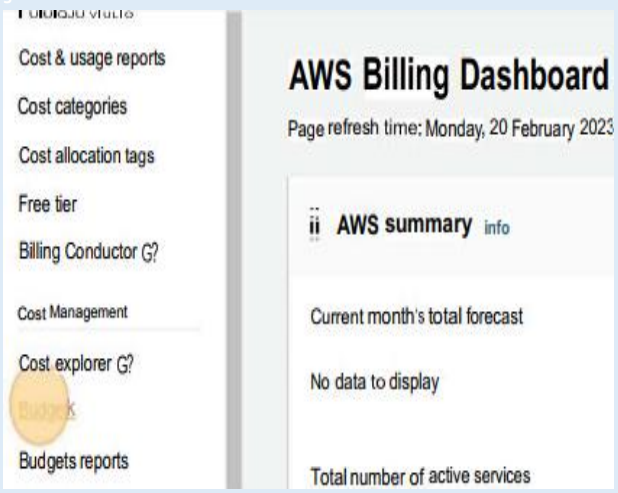
2. Click on "Hitesh Periwal"



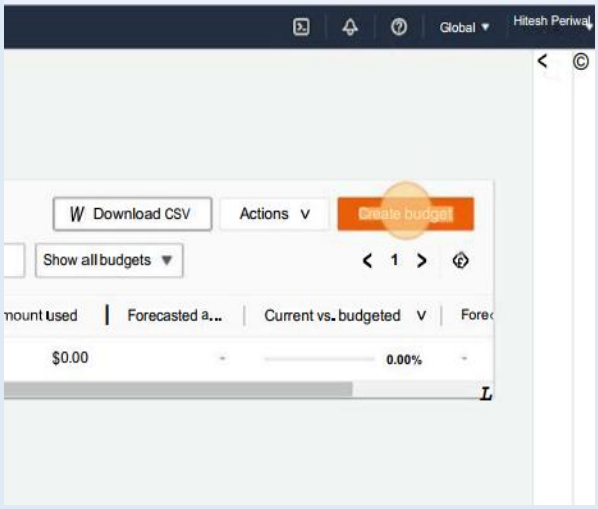
3. Click on "Billing Dashboard"



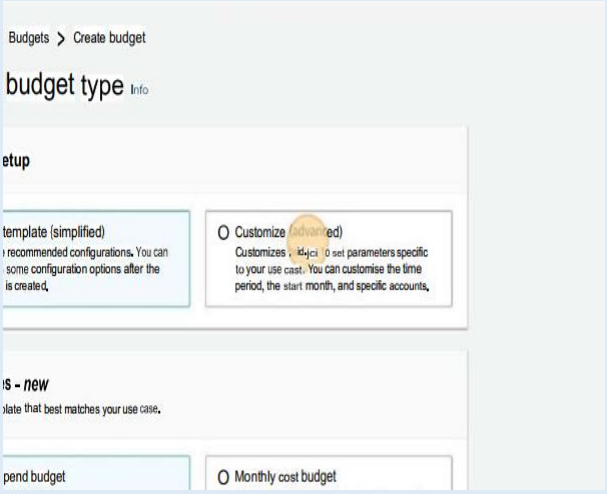
4. Click on "Budgets"



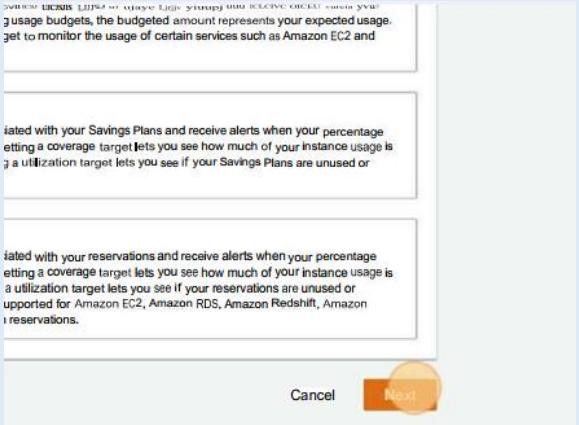
5. Click on “Create Budget”



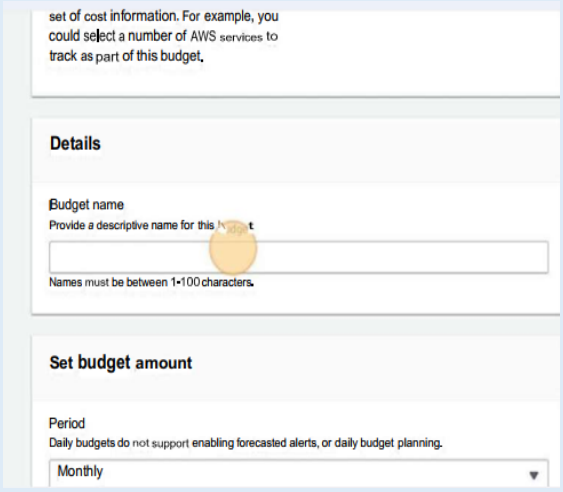
6. Select “Customize (advanced)



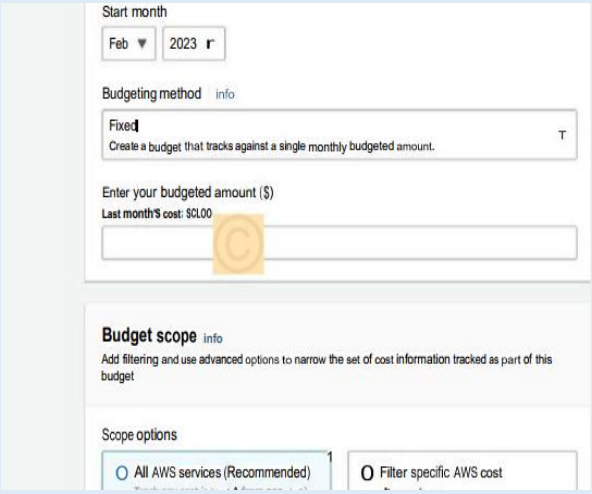
7. Click on “Next”



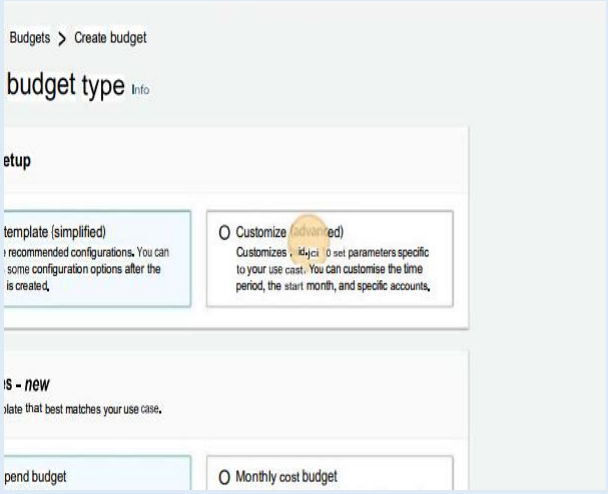
8. Give your budget a name



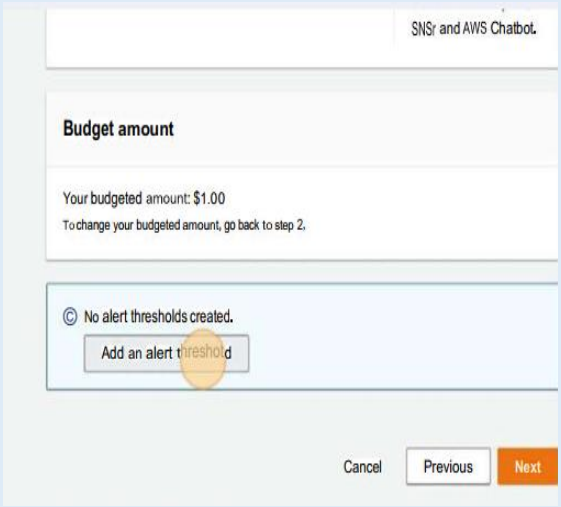
9. Enter the budgeted amount in $



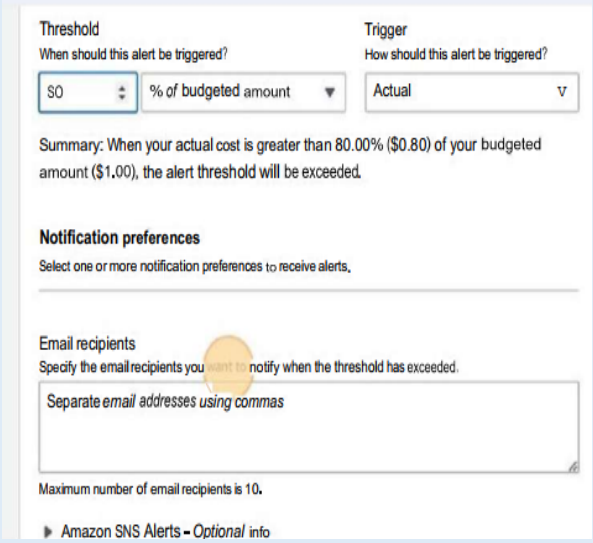
10. Click on "Next"



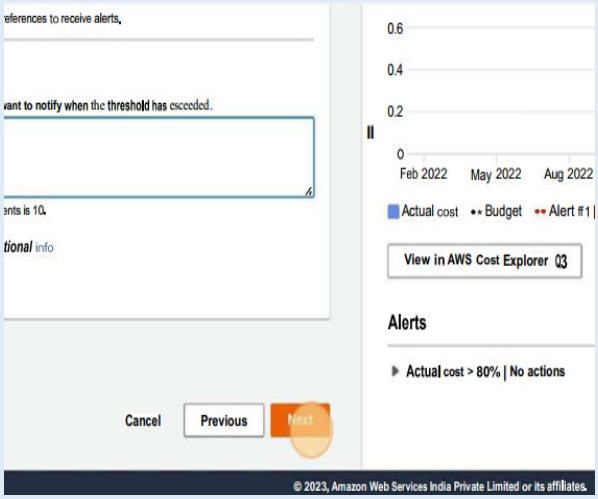
11. Click on "Add an alert threshold"



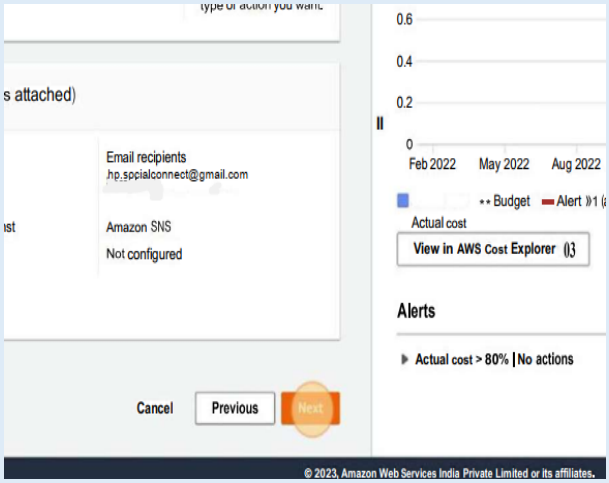
12. Set the threshold and then specify the email recipients you want to notify when threshold has exceeded



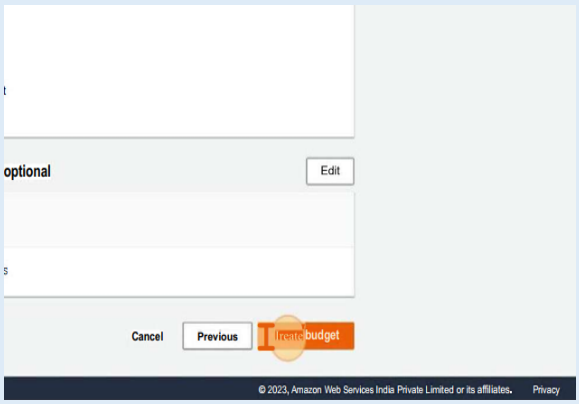
13. Click on “Create Budget”



14. Select “Customize (advanced)



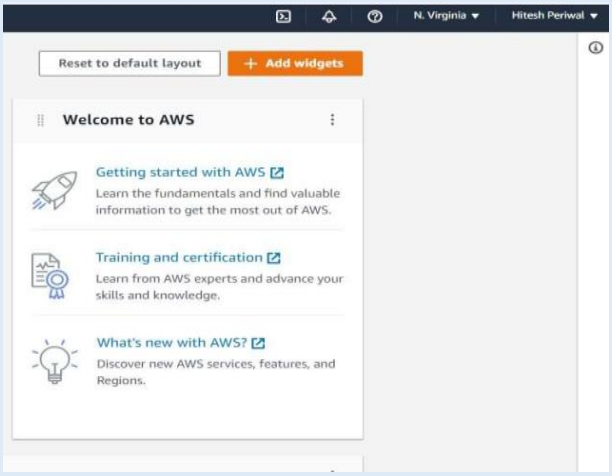
15. Click on “Create budget ” and hence the budget is created



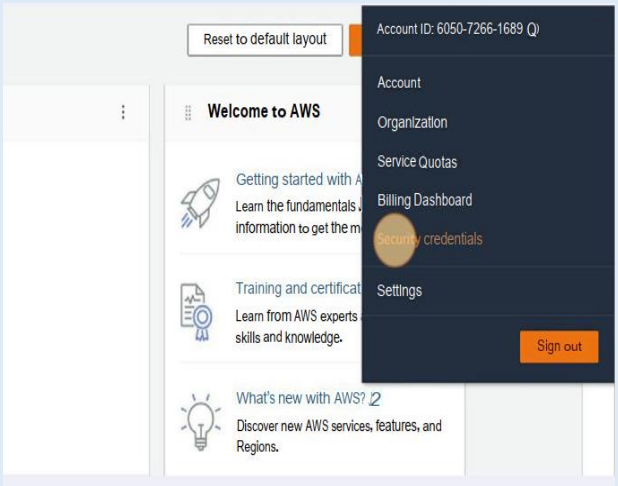
ACTIVATION OF MFA SECURITY

1. **Visit aws.amazon.com and Sign in**

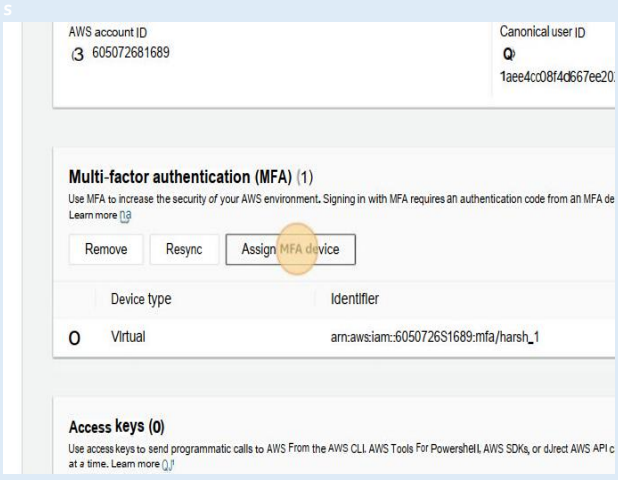
2. **Click on “Hitesh Periwal”**



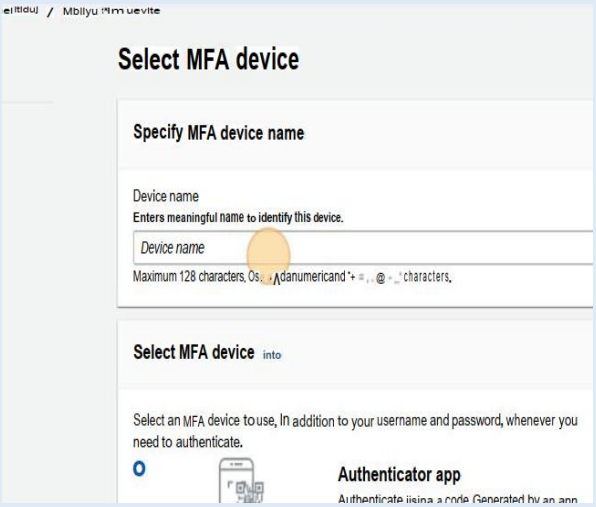
3. **Click on “Security credentials”**



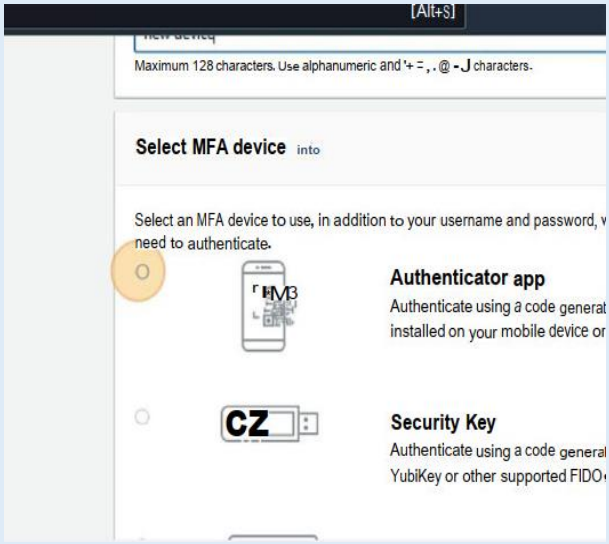
4. **Click on “Assign MFA device”**



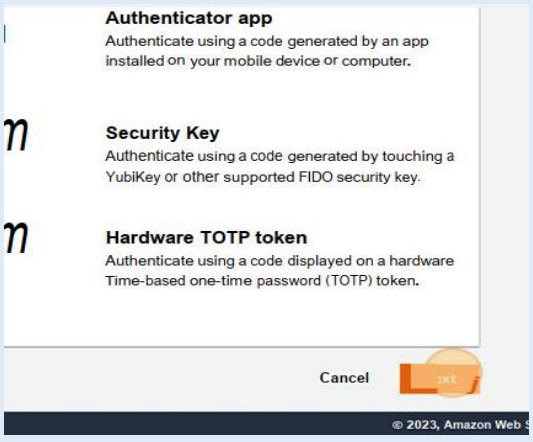
5. Enter the Device name



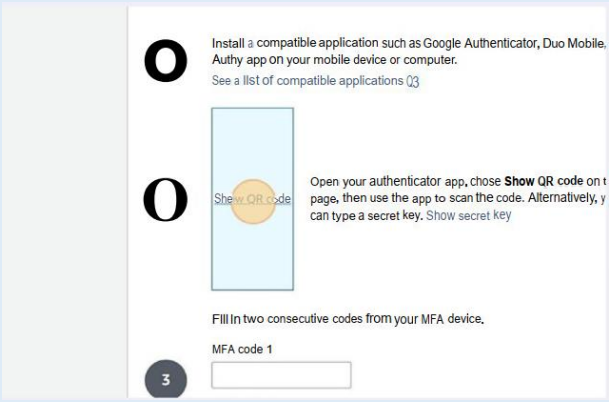
6. Click on the "Authenticator app" option



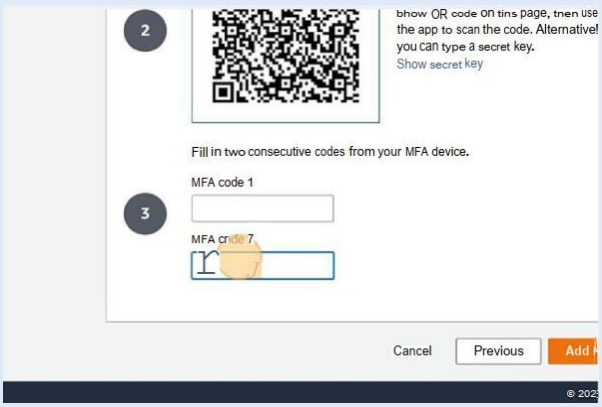
7. Click on “Next”



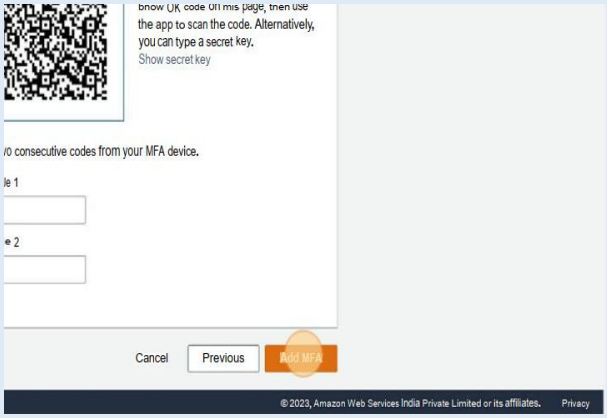
8. Download Authenticator from Google Play Store on your Smart Phone.Then click on PLUS button. Then click on "Scan a QR Code".Then come back to Desktop Screen.Click "Show QR code”



9. **Get two consecutive MFA code from smartphone and enter in the respective fields**



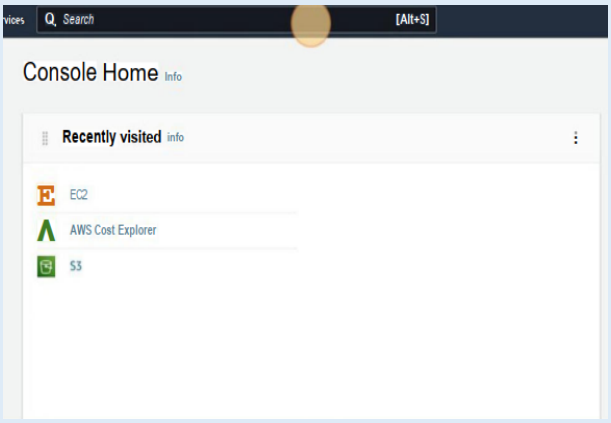
10. **Click on “Add MFA” and thus MFA has been added**



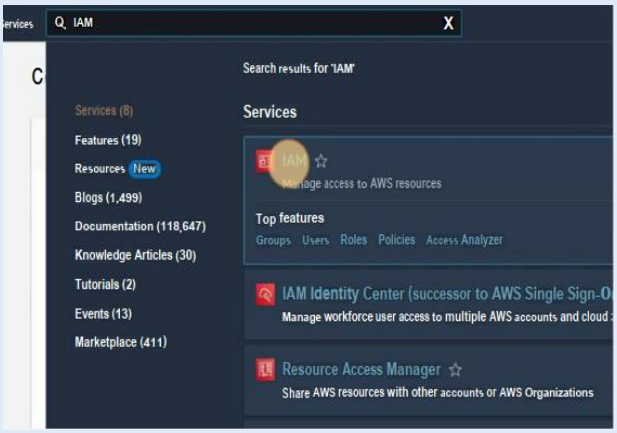
CREATE AN IAM USER GIVING FULL S3 ACCESS

1. **Visit aws.amazon.com and Sign in**

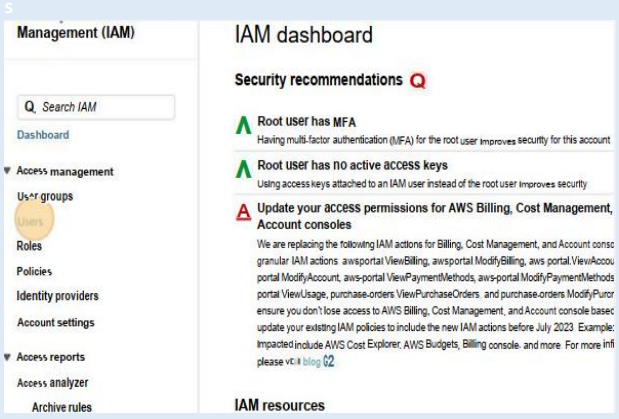
2. **Click on the search field and type "IAM" in the search field**



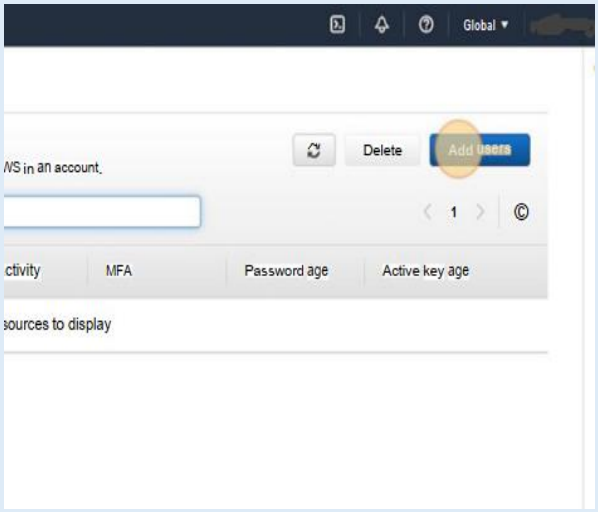
3. **Click on "IAM"**



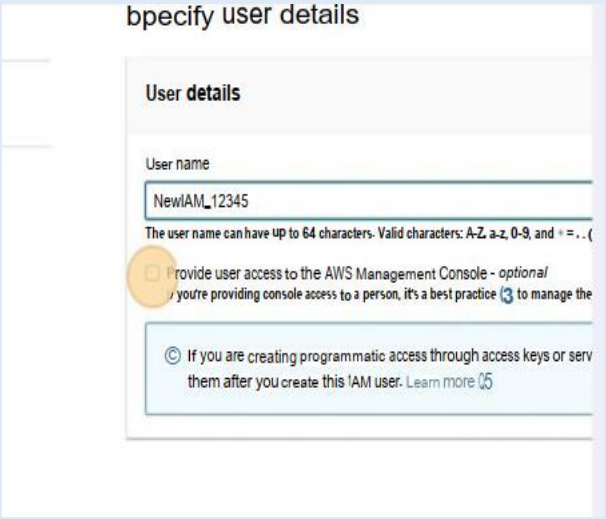
4. **Click on "Users"**



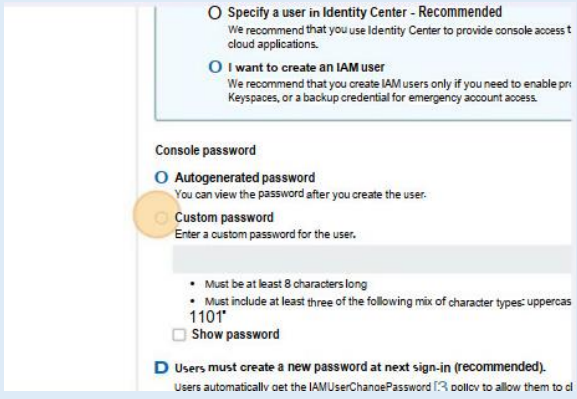
5. **Click on “Add users”**



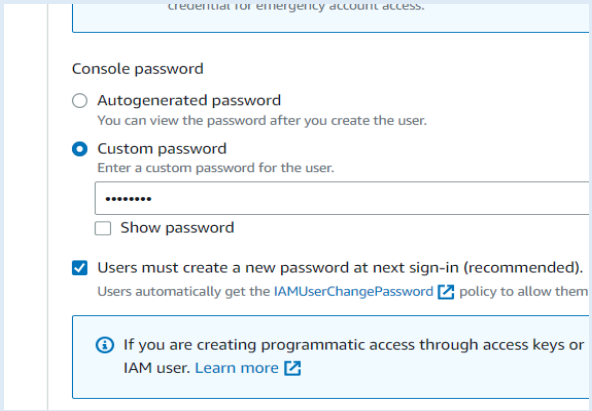
6. **Enter the user name and click on the checkbox to provide user access to AWS Management Console**



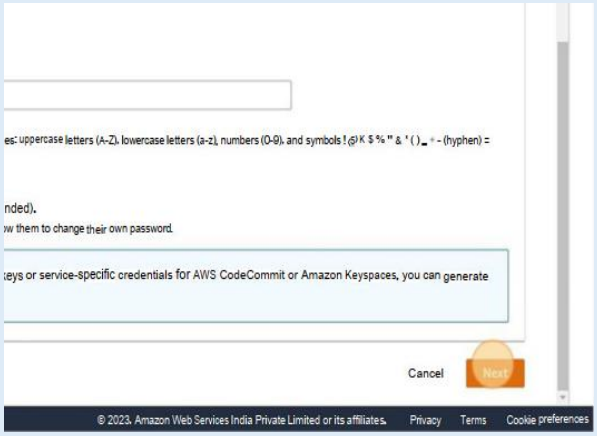
7. **Select “I want to create an IAM user” as well as “custom password”**



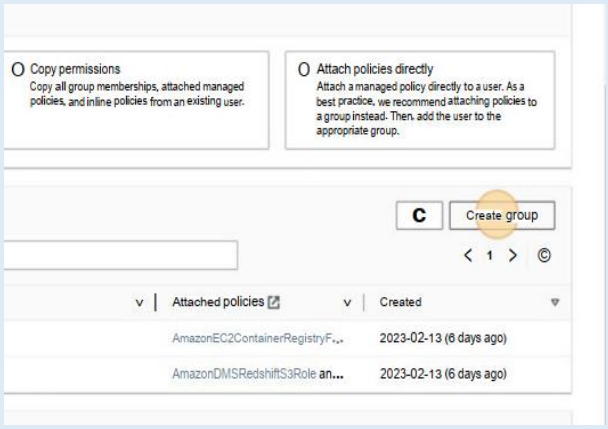
8. **Set the password and click the checkbox “Users must create a new password at next sign-in”**



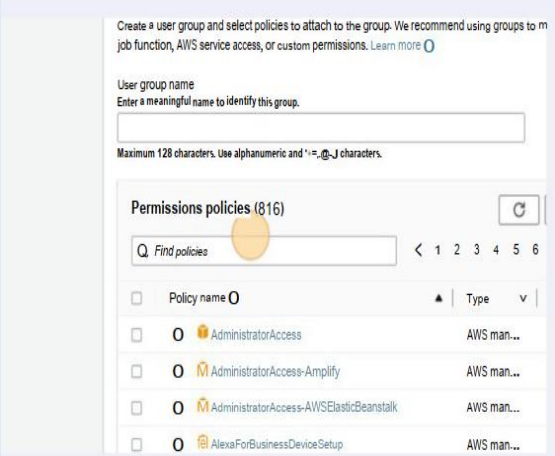
9. Click on “Next”



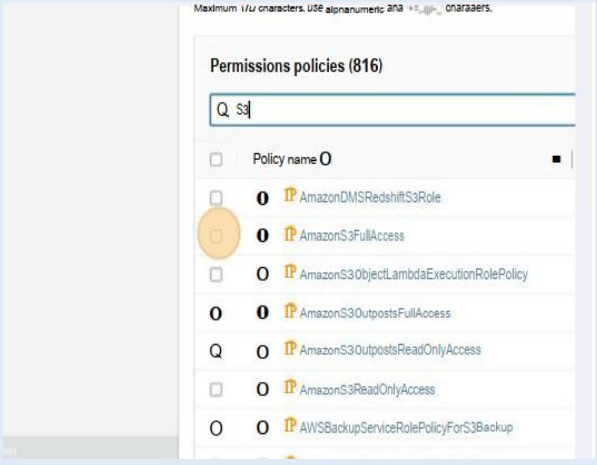
10. Click on “Create group”



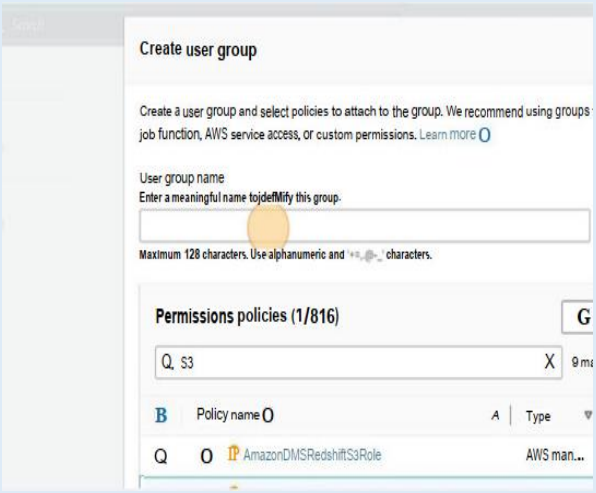
11. Click the “find policies” field and type “S3”



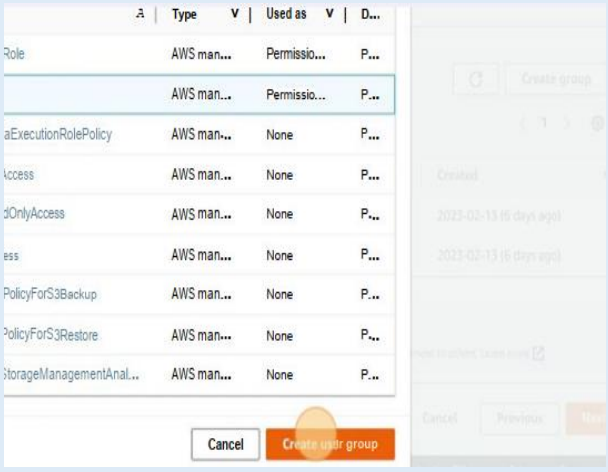
12. Click the checkbox “AmazonS3FullAccess”



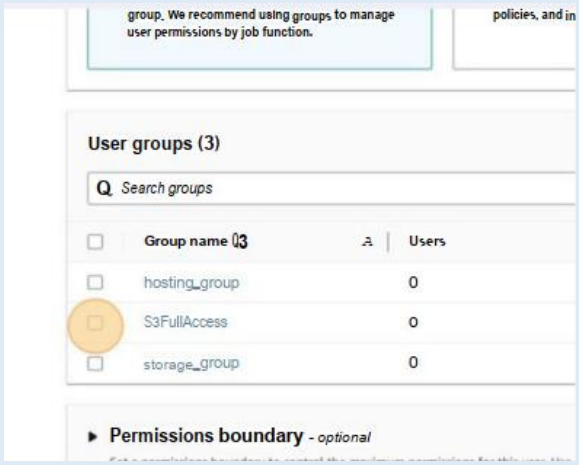
13. **Enter a user group name**



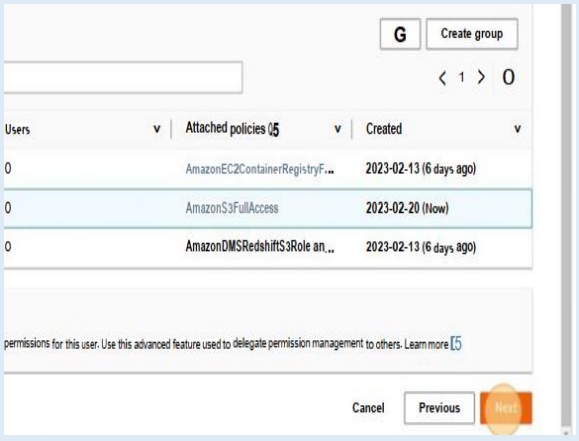
14. **Click on "Create user group"**



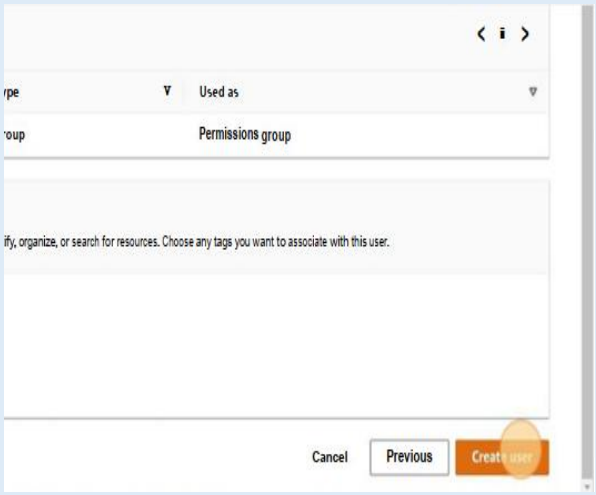
15. **Click on the checkbox "S3FullAccess"**



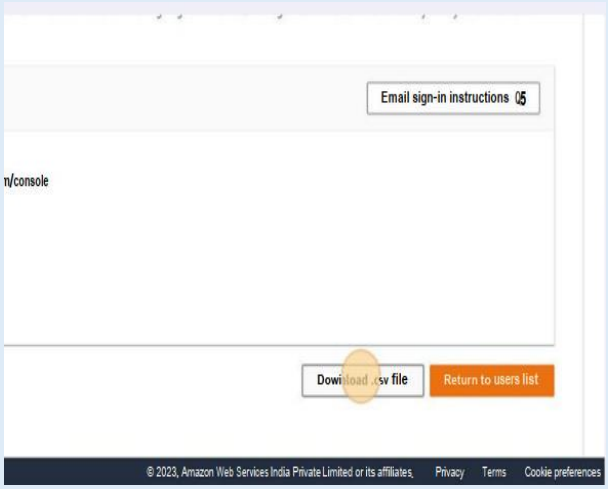
16. **Click on "Next"**



17. **Click on “Create user”**



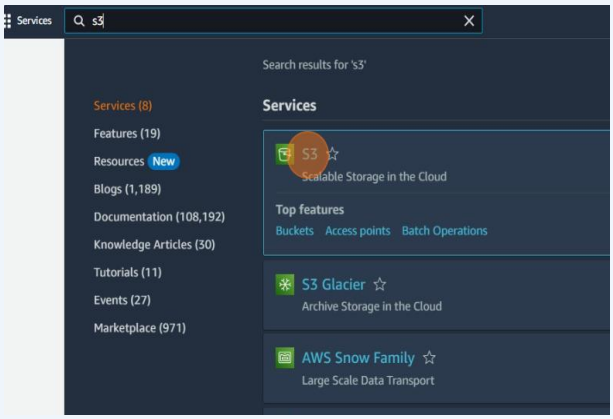
18. **Click on “Download .csv file”**



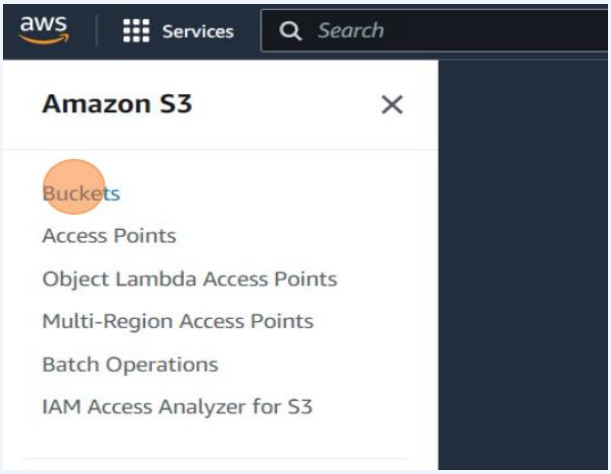
17. **IAM user is thus created**

CREATE A PRIVATE BUCKET IN AWS.UPLOAD A FILE AND CHECK BY PRESIGNED URL THAT YOU CAN ACCESS THE FILE OR NOT.

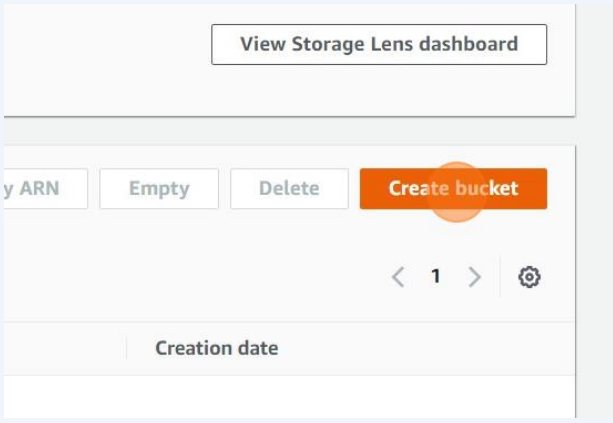
1. **Visit aws.amazon.com and Sign in.Click on the search field and type "s3" in the search field and select S3**



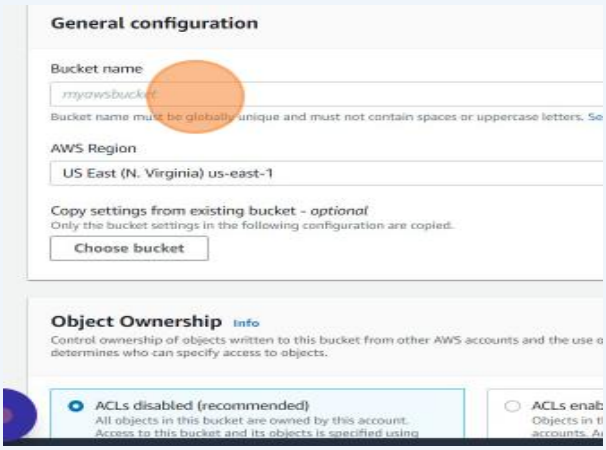
2. **Click on "Buckets"**



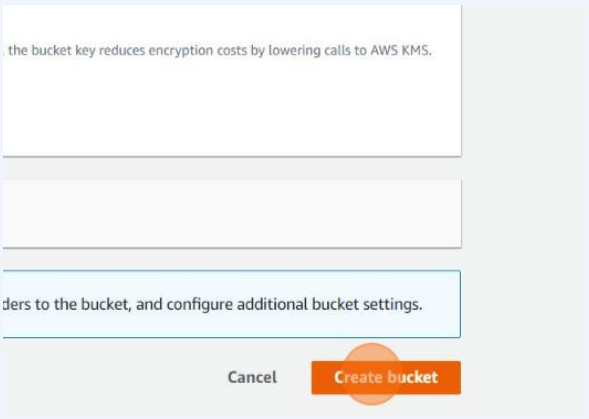
3. **Click on "Create Bucket"**



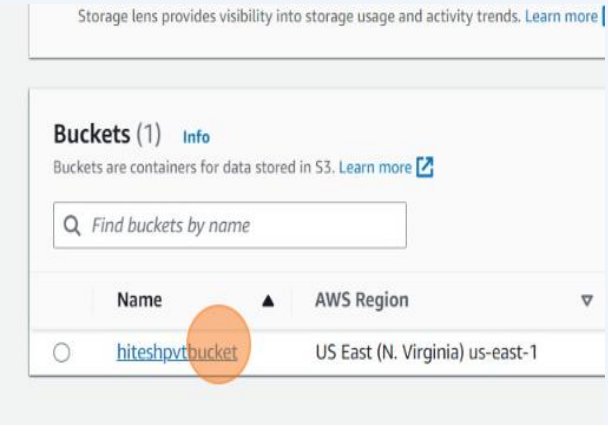
4. **Click the "Bucket name" field and type "hiteshpvtbucket" and keep all other options unchanged.**



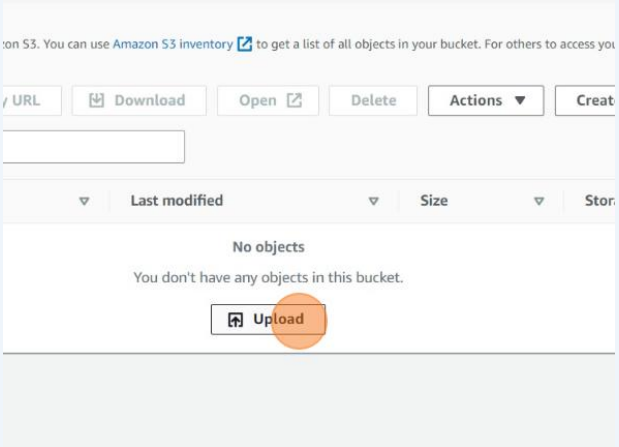
5. Click on “Create bucket”



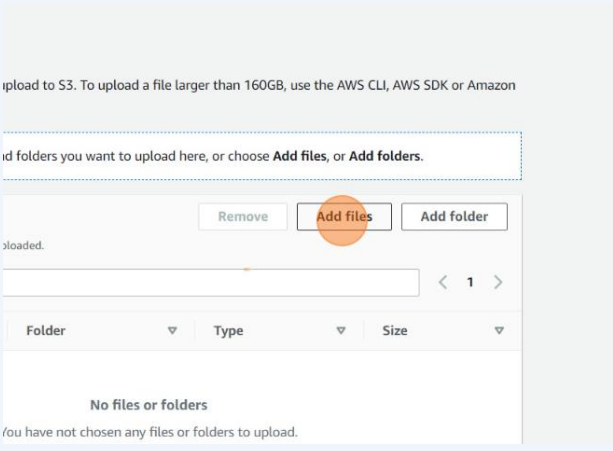
6. Click on “hiteshpvtbucket”



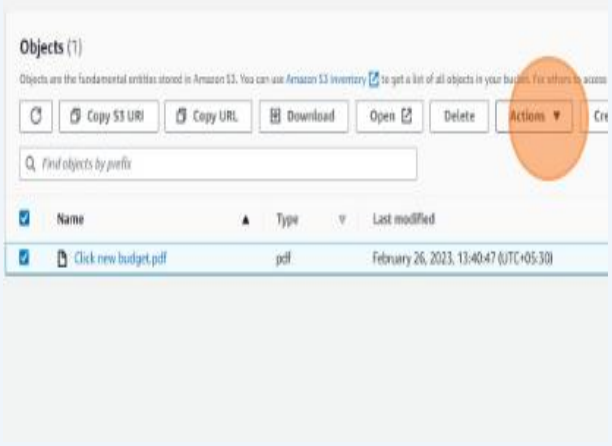
7. Click on “Upload”



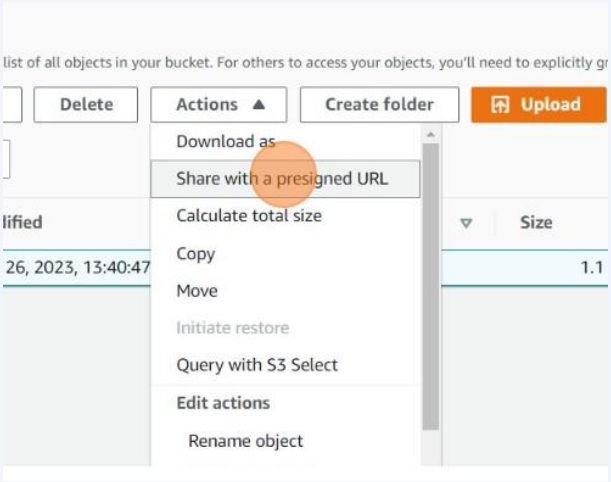
8. Add files and folders which you want to upload and click upload



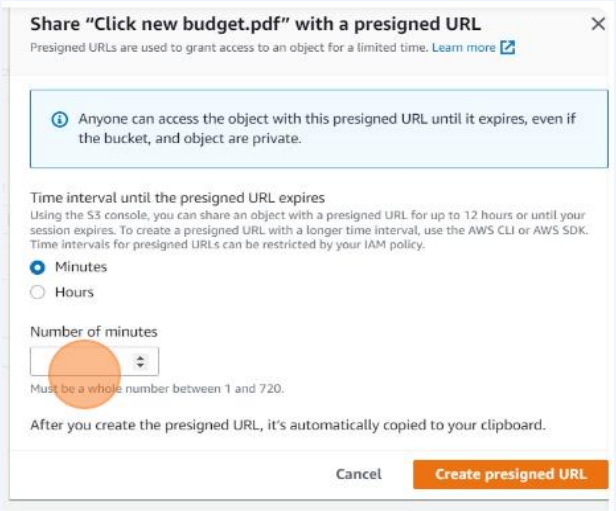
9. **Select the file for which you want to generate presigned url and click "Action"**



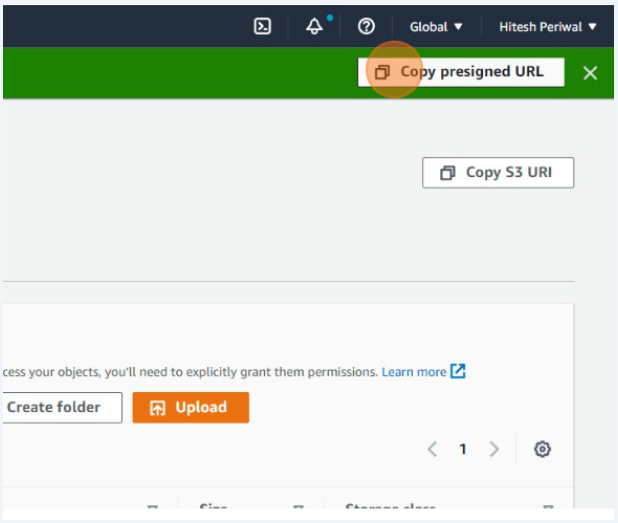
10. **Click on "Share with a presigned URL"**



11. **Set the time interval until which the presigned URL expires and click on "Create presigned URL"**



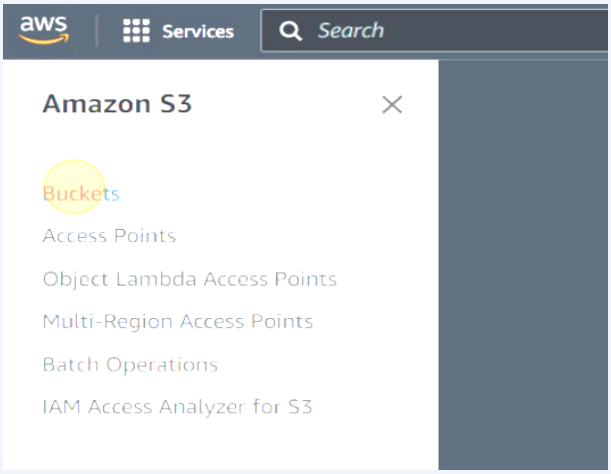
12. **Click on "Copy presigned URL" and you will be able to access the file using the URL for the set time period only.**



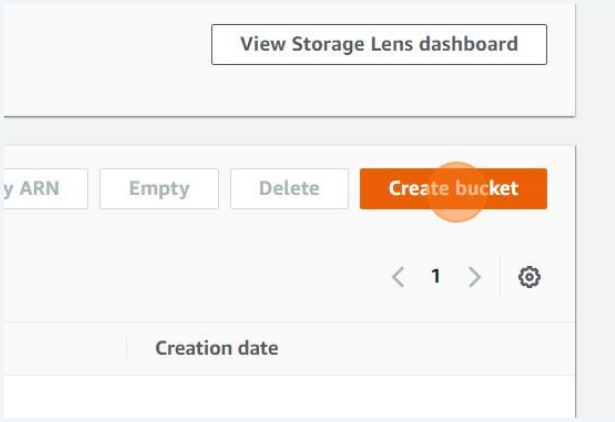
CREATE A PUBLIC BUCKET IN AWS.UPLOAD A FILE AND GIVE THE NECESSARY PERMISSION TO CHECK WHETHER THE FILE URL IS WORKING OR NOT.

1. **Visit aws.amazon.com and Sign in.Click on the search field and type "s3" in the search field and select S3**

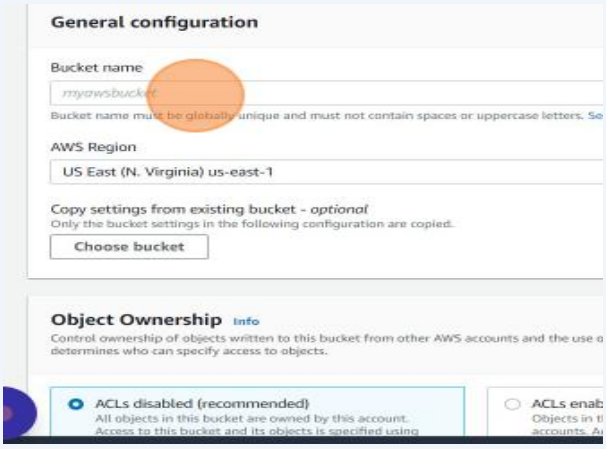
2. **Click on "Buckets"**



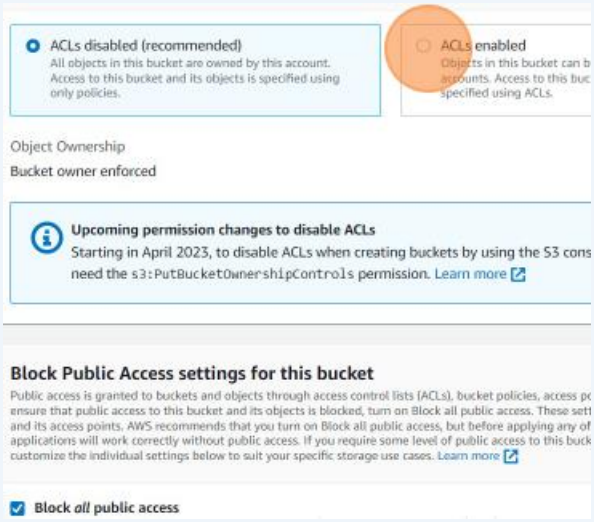
3. **Click on "Create Bucket"**



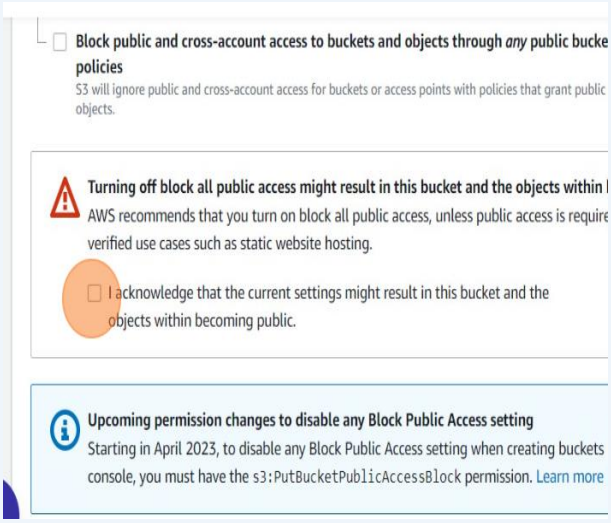
4. **Click the "Bucket name" field and type "hiteshpublicbucket".**



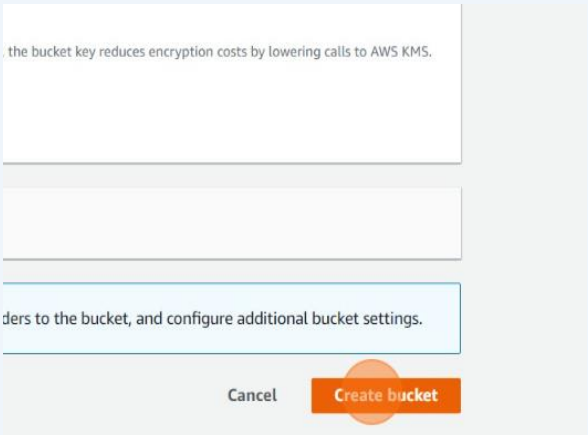
5. Select ACLs enabled and uncheck “Block all public access”



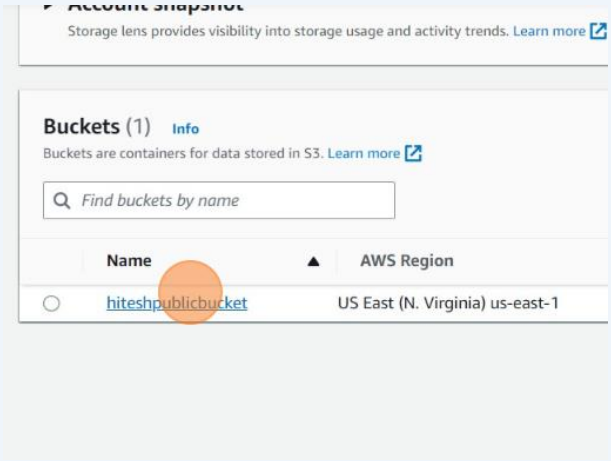
6. Click on this check box



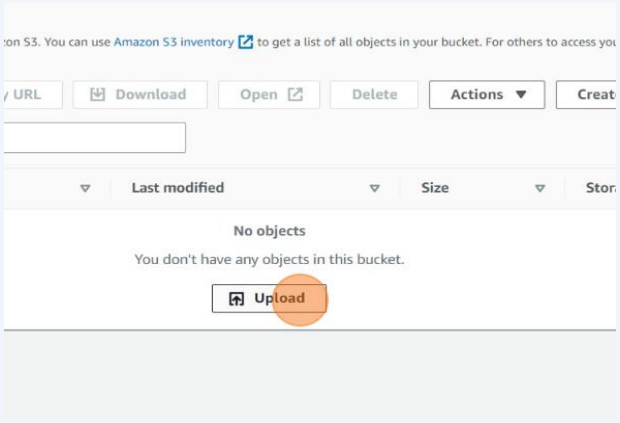
7. Click on “Create bucket”



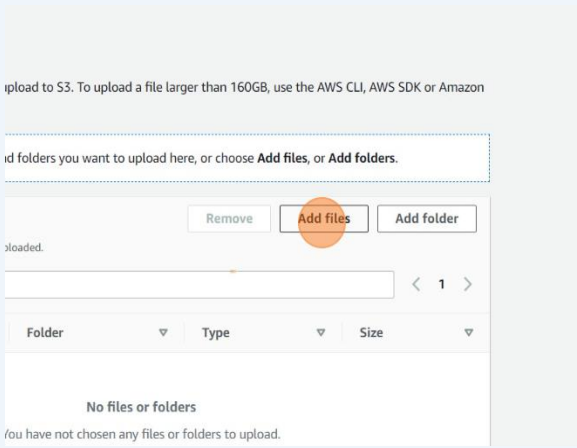
8. Click on “hiteshpublicbucket”



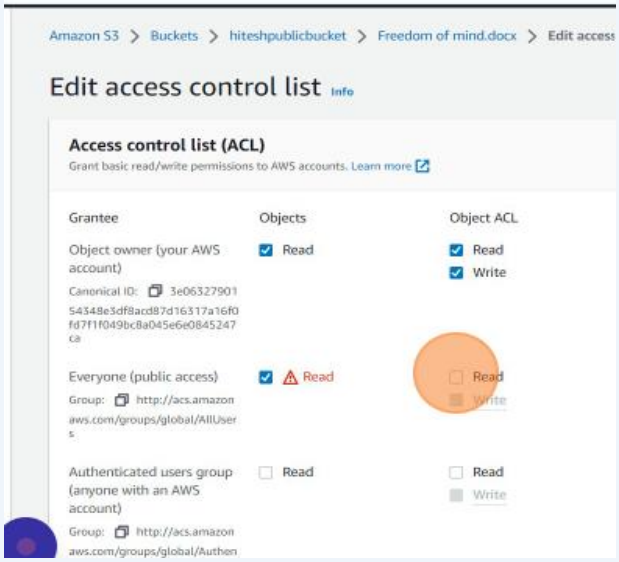
9. **Click on “Upload”**



10. **Add files and folders which you want to upload and click upload**

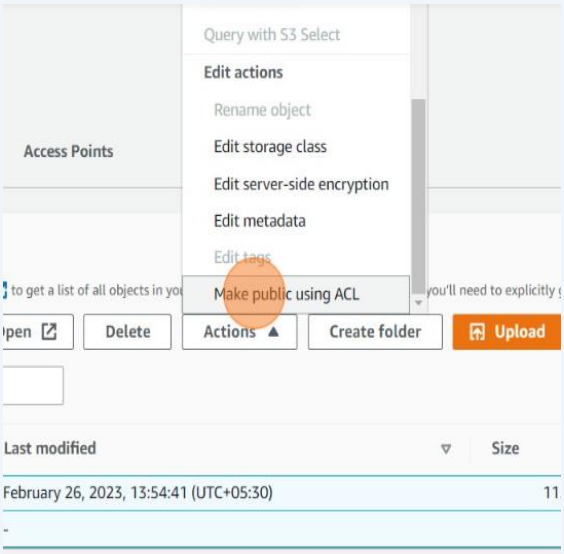


11. **You may click the filename and then go to permission and select edit and allow everyone to access the file. Acknowlegde and then save changes.**



or

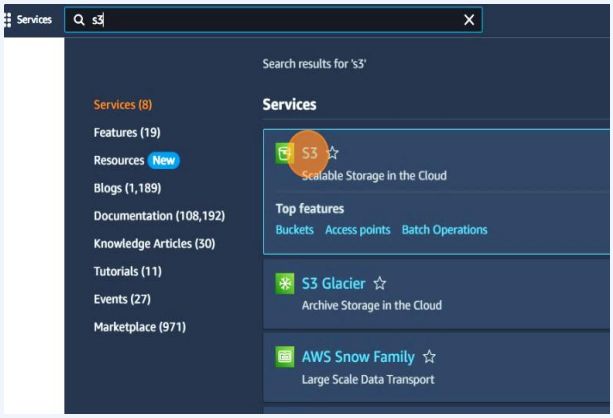
12. **Select the file click “Action”. Click on “Make public using ACL” and then click on make public.**



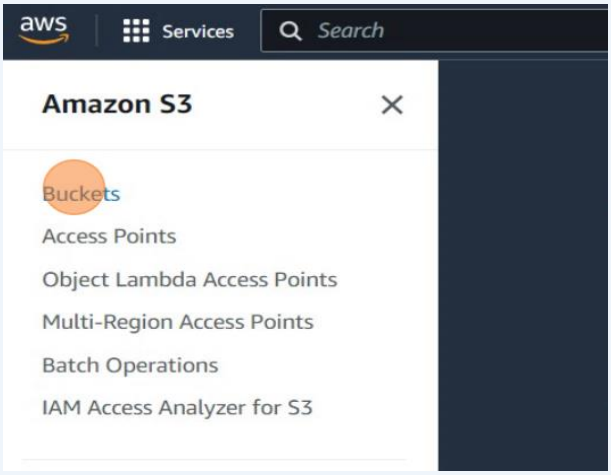
13. **Now the file is accessible through both the object URL and the presigned URL.**

UPLOAD A STATIC WEBSITE ON S3

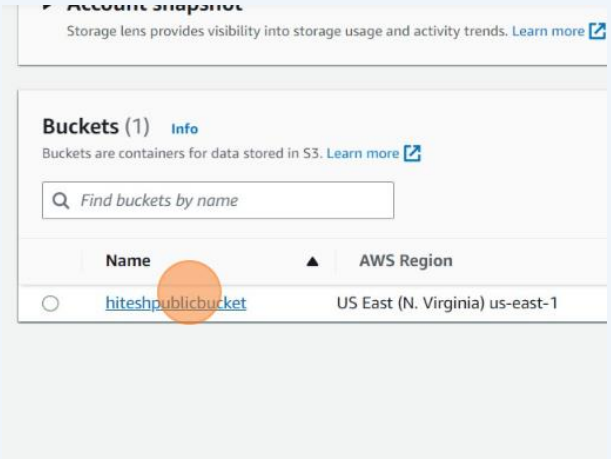
1. **Keep the html files ready.Visit aws.amazon.com and Sign in.Click on the search field and type “S3” in the search field and select “S3”**



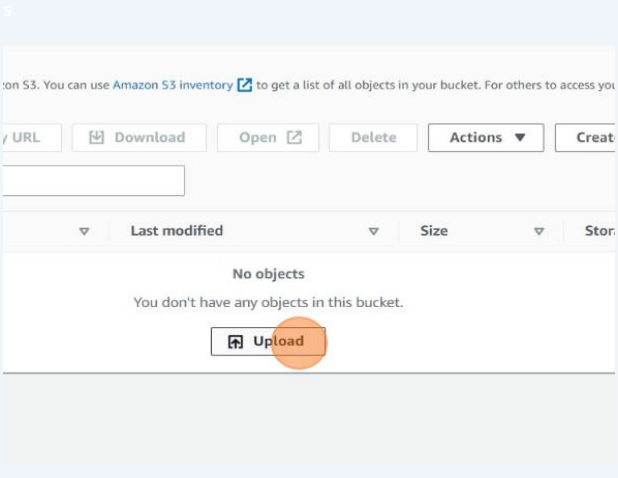
2. **Click on “Buckets”**



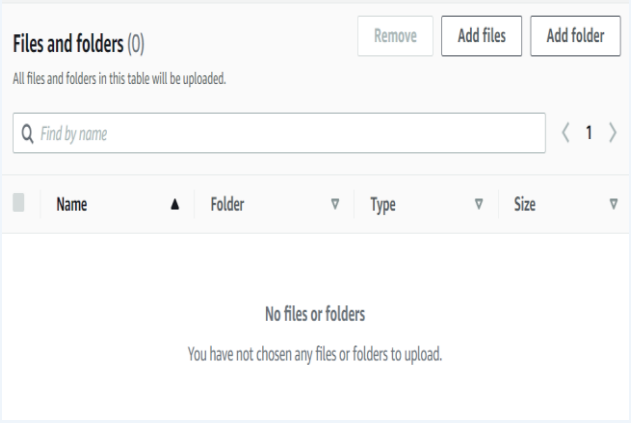
3. **Create a public bucket and click on it**



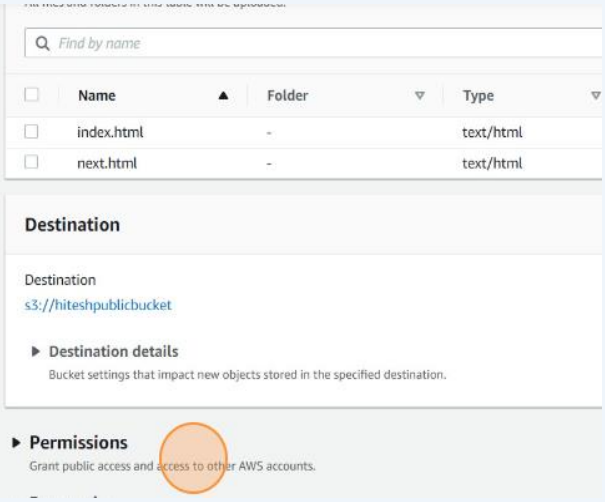
4. **Click on upload button**



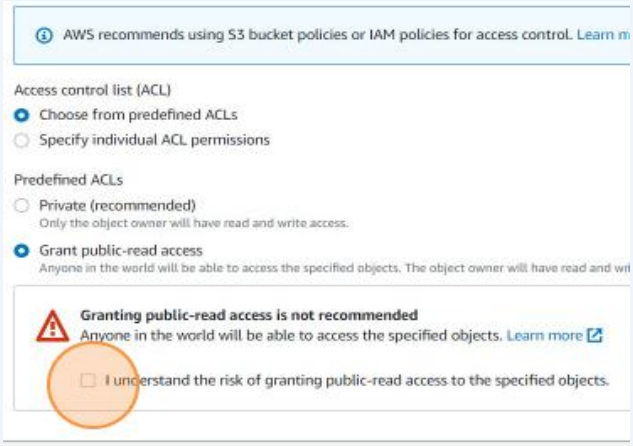
5. **Select the html files you want to upload by clicking on “Add files”**



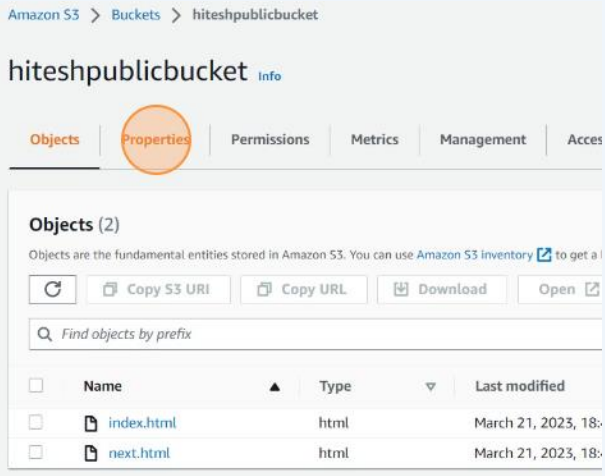
6. **Click on “Permissions”**



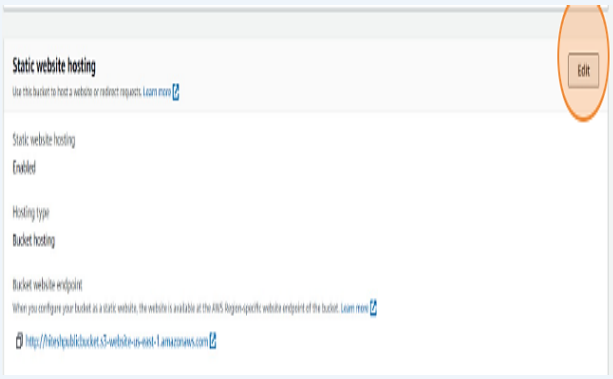
7. **Select the option “Choose from predefined ACLs”,”Grant public-read access”,check the checkbox “I understand the risk of …..” and then upload the file**



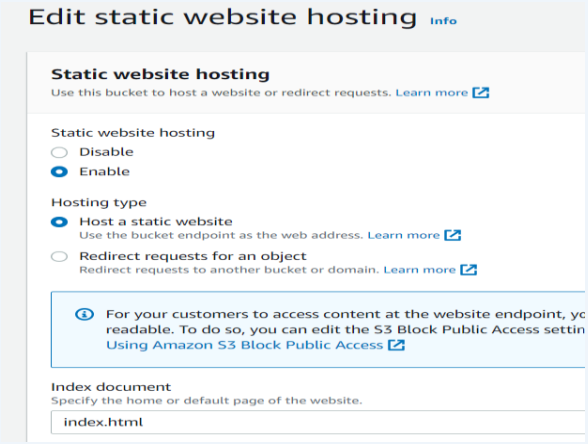
8. **Go to public bucket in which the website was uploaded.Select the properties tab**



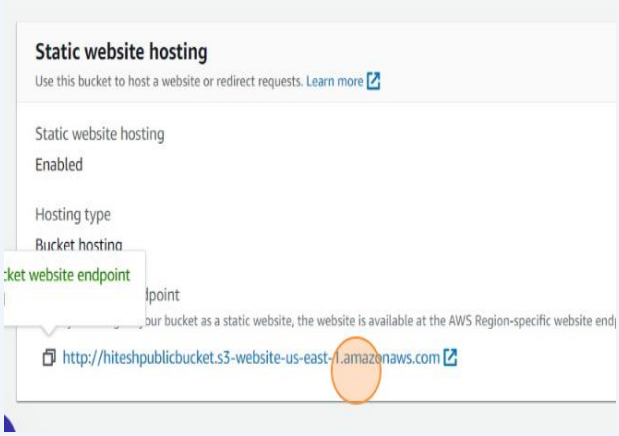
9. Go to static website section and click on “Edit”



10. Choose the option “Enable” for static website hosting,then choose “Host a static website” and specify the index document of the website.Finally click on save changes



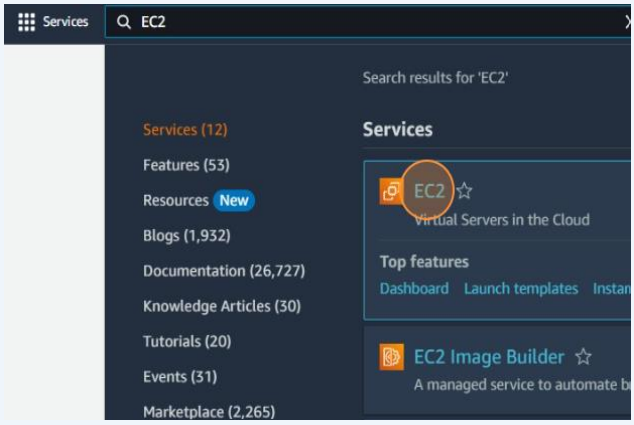
11. After saving changes go down and find the bucket website endpoint link.See whether the website is accessible or not across different devices



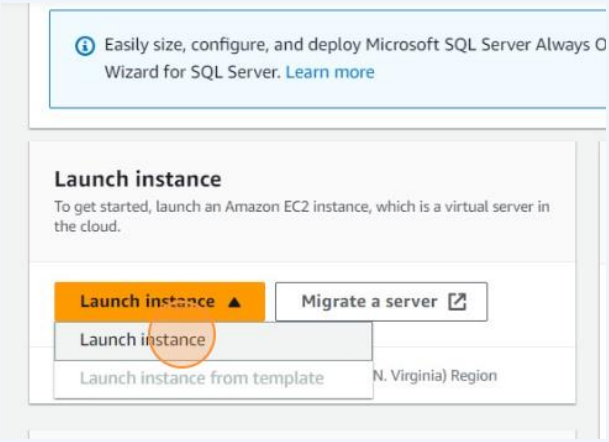
12. The website is accessible

UPLOAD A STATIC WEBSITE ON EC2 SERVER

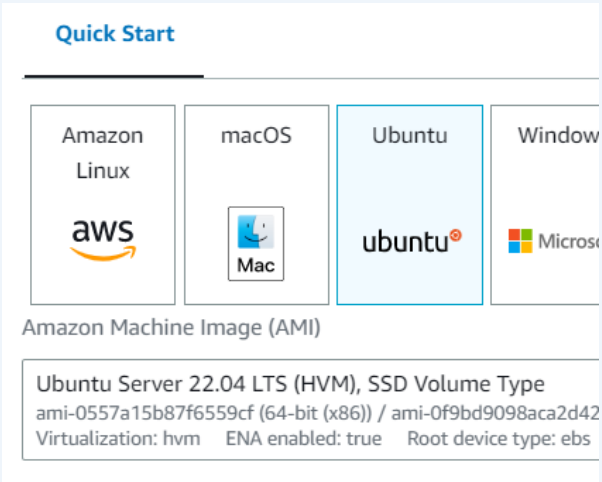
1. **Visit aws.amazon.com and Sign in.Click on the search field and type “EC2” in the search field and select EC2**



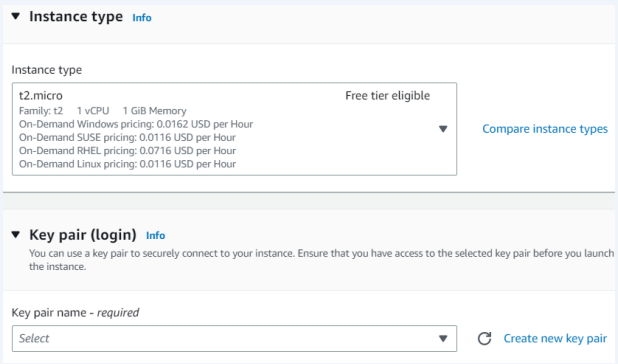
2. **Select “Launch instance” option**



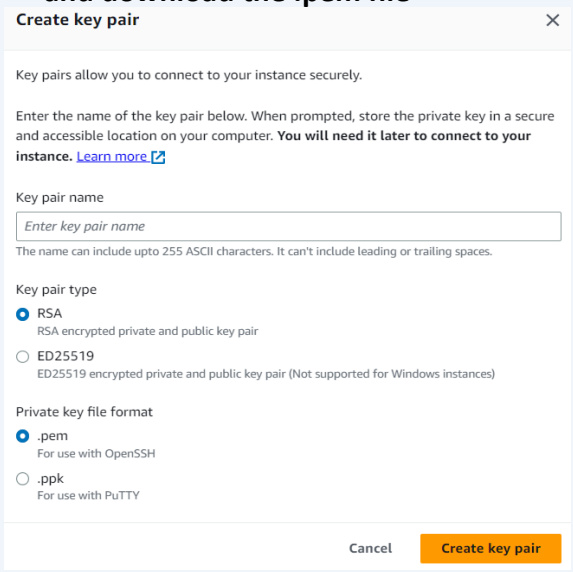
3. **Select Ubuntu**



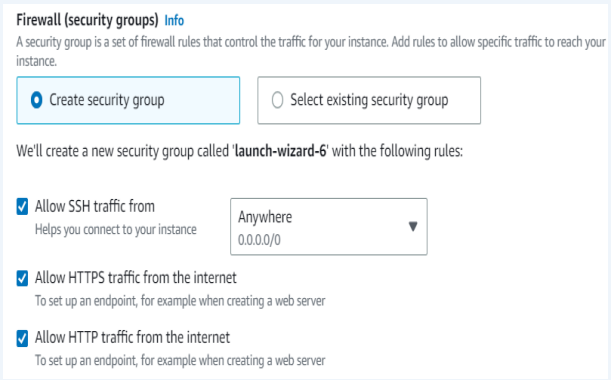
4. **Select instance type as “t2.micro” and and then “Click on create new key pair”**



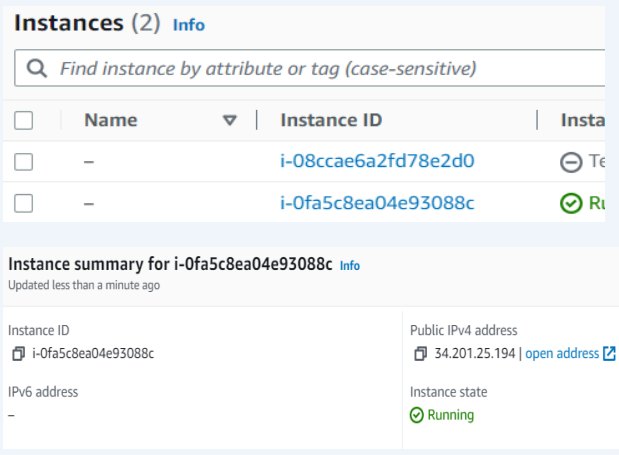
5. **Type the key pair name,Select “RSA” as key type and “.pem” as key file format.Then click on “Create key pair” and download the .pem file**



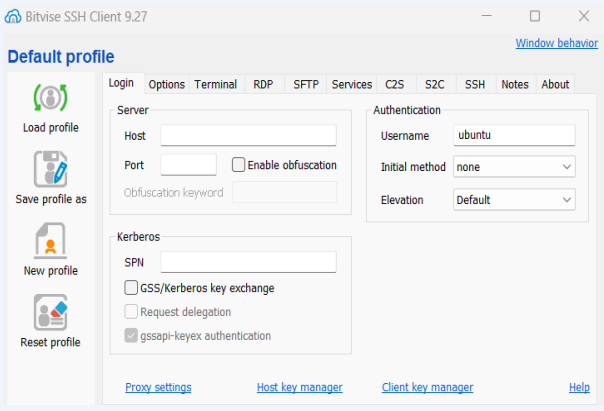
6. **Allow all the following three traffics and click on “Launch instance”**



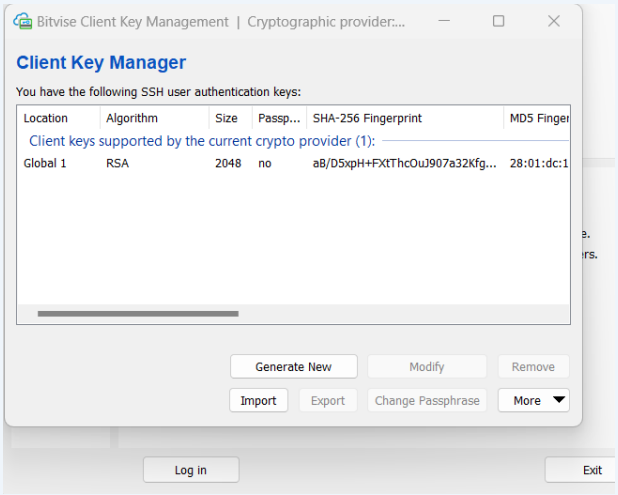
7. **On the instances page click on the instance id and then copy the public IPv4 address**



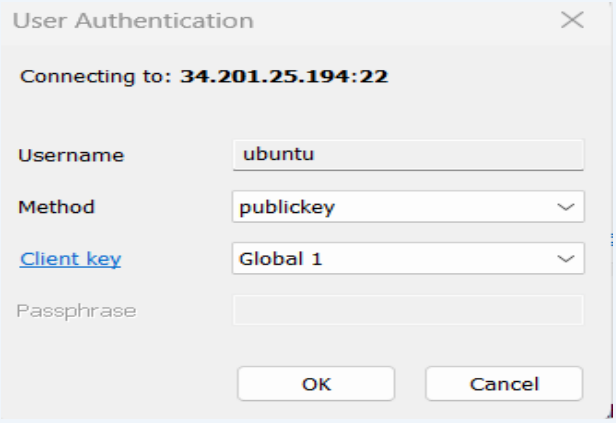
8. **Open Bitwise SSH client,paste the copied public IPv4 address in the host box.Select username as Ubuntu,initial method as none and elevation as default.Then click on client key manager**



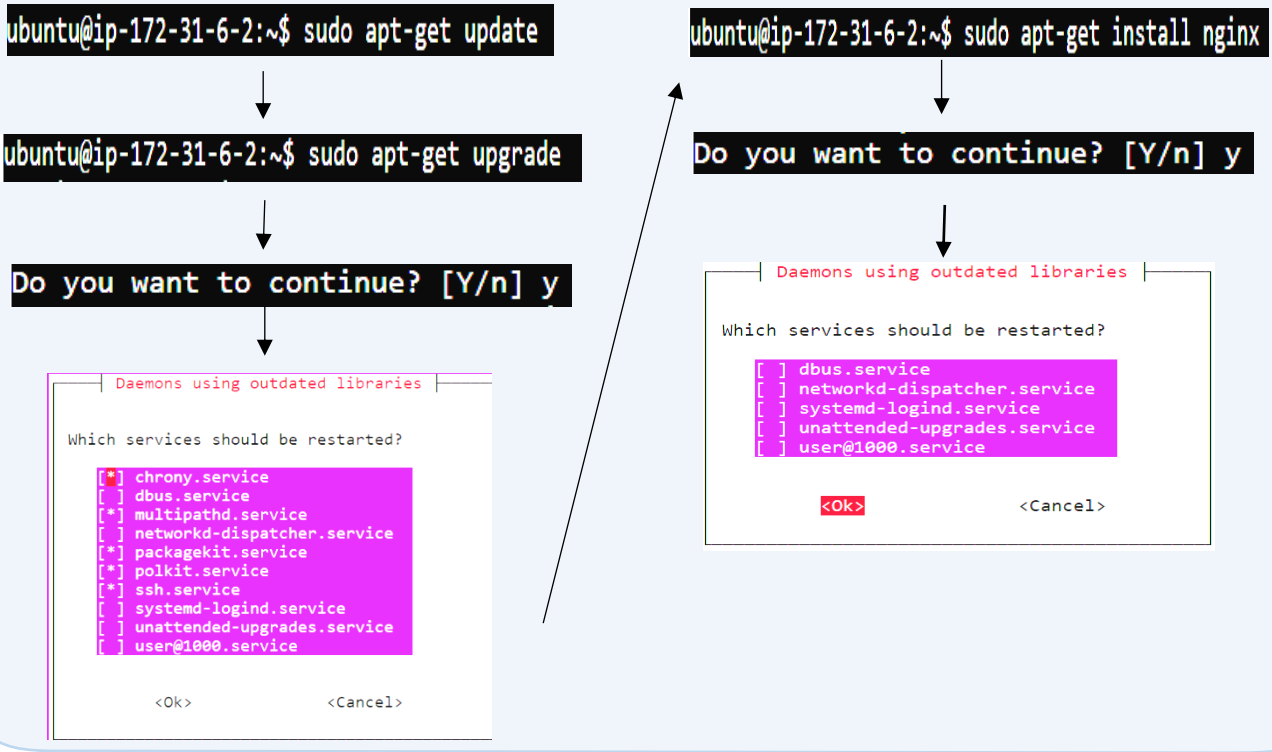
9. **Import the download .pem file and then click on login on the SSH client.**



10. **Select accept and continue and then type the username as Ubuntu,method as public key and client key as Global 1.Then click on "OK".**



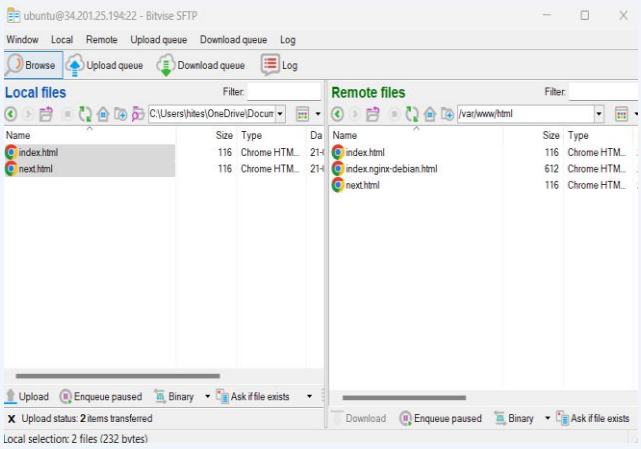
11. **Open the terminal console in the SSH client and type and run the following commands individually.If asked for confirmation click on yes and ok depending on the options available.**



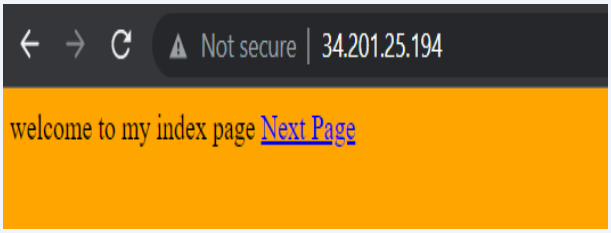
12. **Type the following commands in the terminal to change the working directory to the root and then move to var/www/ to give the permission add files to that folder.**

```
ubuntu@ip-172-31-6-2:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-6-2:~$ cd /
ubuntu@ip-172-31-6-2:/$ cd var/www/
ubuntu@ip-172-31-6-2:/var/www$ sudo chmod 777 html
```

13. **Open new SFTP window and move to the var/www/html directory in the server machine.Now drag and drop the required html files from the local machine to the server machine.**



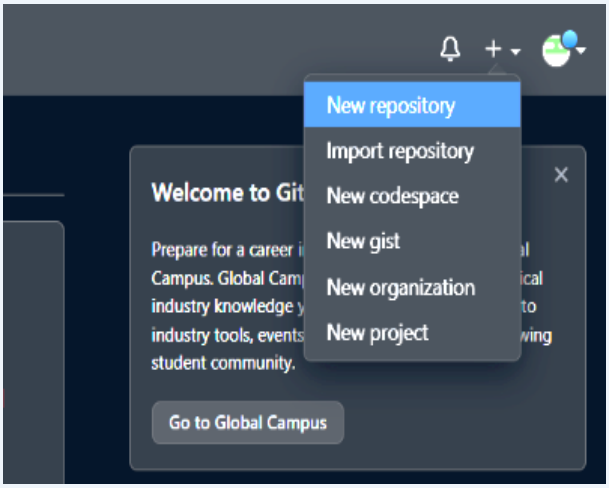
14. **Open the site using the public IPv4 address and it will be accessible**



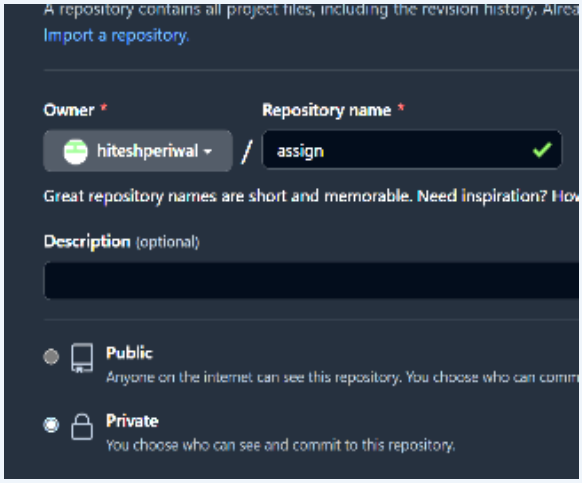
Assignment No. 8

DEPLOY A PROJECT FROM LOCAL MACHINE TO GITHUB AND VICE-VERSA

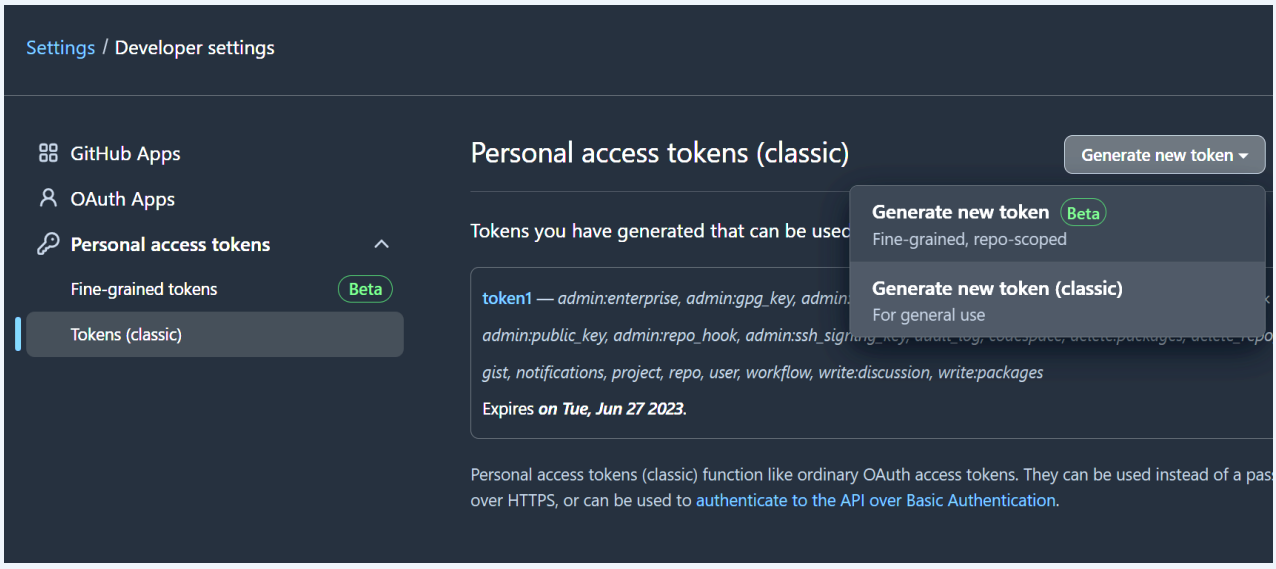
1. **Visit github.com,login in it and the click on “New repository”.**



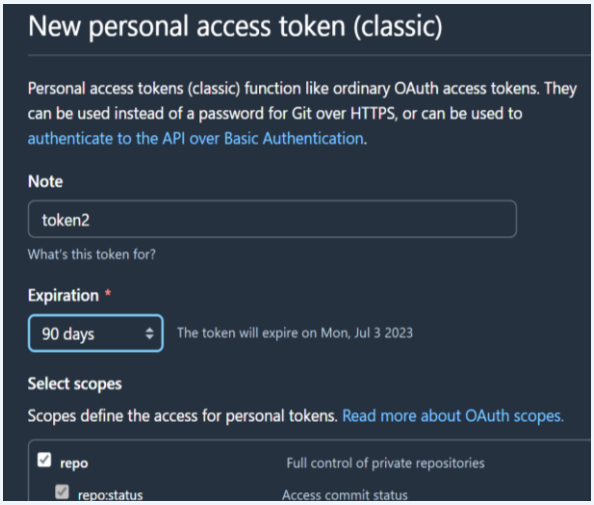
2. **Give a Repository name and set it as private.Then click on “Create repository”.**



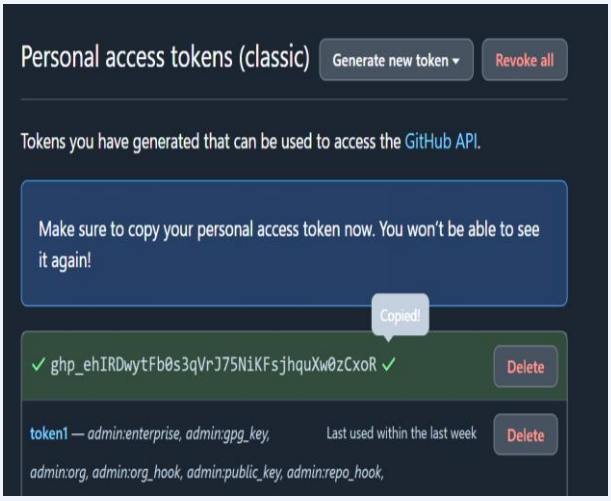
3. **Go to Settings,on the bottom of the left panel click on “developer settings”.Then go to “Personal access tokens” ,choose “Tokens(classic)”.Now on the right side select “Generate new token” and then “Generate new token(classic).**



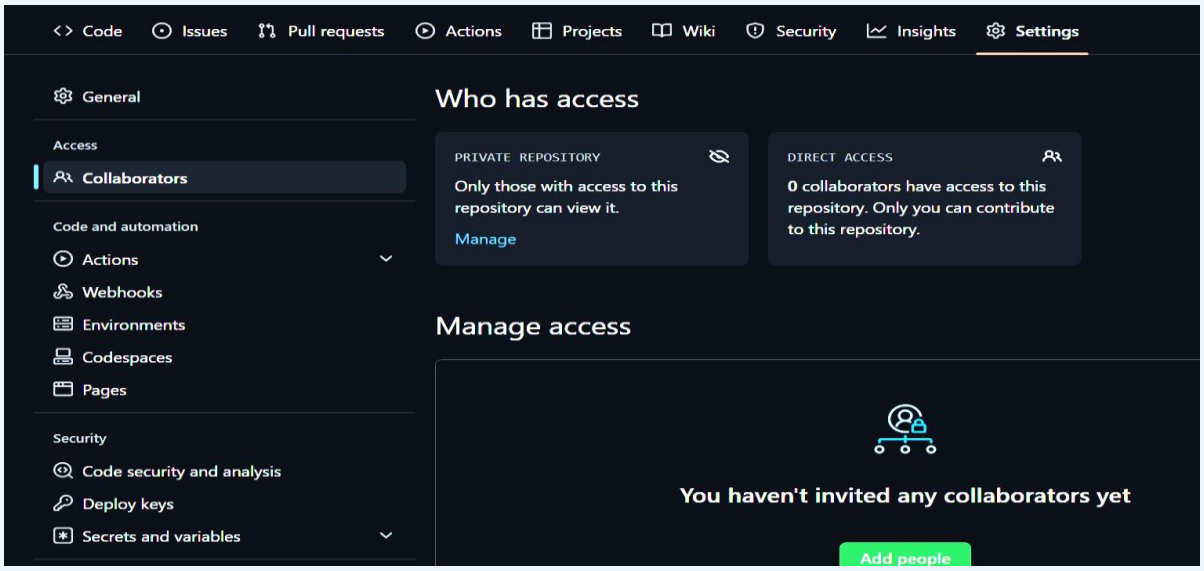
4. **Now give this token a name,select expiration duration as 90 days.Click on all the checkbox listed under scopes to get the full control.Then click on “Generate token”.**



5. **Copy and save the generated token for future use.**

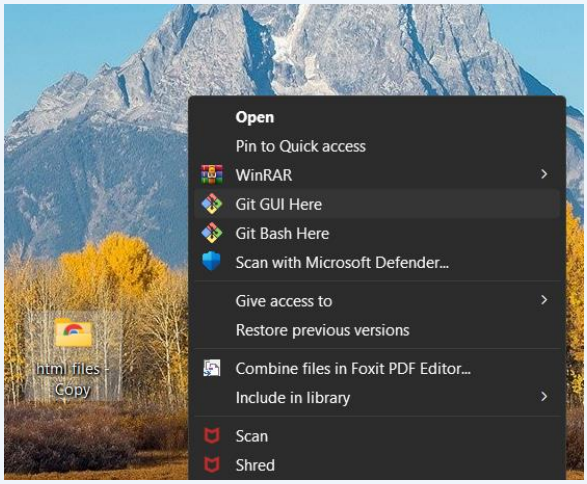


6. **Go to settings option inside “assign” repository.Click on “Collaborators” and “add people” with whom you want to collab this project.Invitation can be sent to the respective person using the github username or email id with which his/her github account is associated.The invitation can be accepted using the link that the person will receive on the registered email or by navigating to the notification section of his/her github account.Upon acceptance of the invitation , the private repository will be accessible to that collaborator.**



7. Copy the https link of the repository which will be required in the upcoming steps.

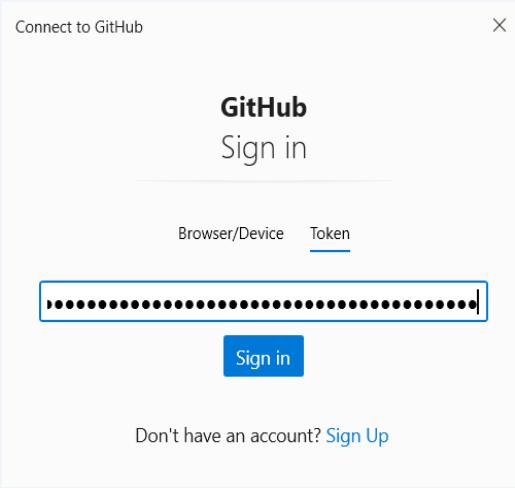
8. Right click on the folder whose file you wish to deploy on github and click on "Git Bash here".



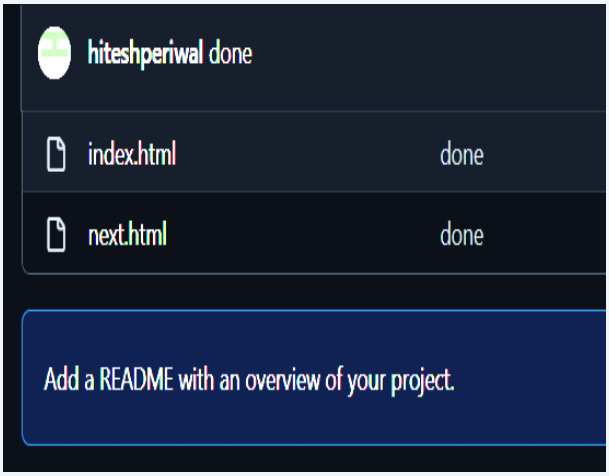
9. Type the following commands in the git bash to deploy a project from local machine to Github

```
hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files -
 Copy
$ git init
Initialized empty Git repository in C:/Users/hites/OneDrive/Documents/onedrive/Desktop/html files

hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files - Copy (master)
$ ls -A
.git/  index.html  next.html

hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files - Copy (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        index.html
        next.html

nothing added to commit but untracked files present (use "git add" to track)

hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files - Copy (master)
$ git add .

hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files - Copy (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html
        new file:   next.html
hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files - Copy (master)
$ git config --global user.email hp.socialconnect@gmail.com

hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files - Copy (master)
$ git config --global user.name hiteshperiwal

hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files - Copy (master)
$ git commit -m "done"
hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files - Copy (master)
$ git remote add origin https://github.com/hiteshperiwal/assign.git

hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/html files - Copy (master)
$ git push -u origin master
```

10. **In the window which poped up,select sign in using token,paste the token saved previously and click “Sign in”.**



11. **The project is thus deployed from local machine to github”.**



12. **To deploy a project from Github to local machine :-**

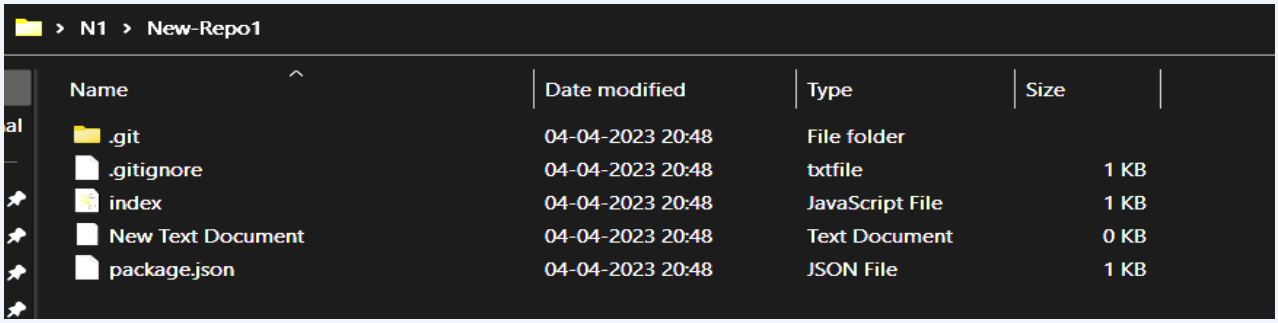
a. Copy the link of the repository which you want to deploy.

b. Create a new folder named “N1” and open gitbash there.

c. Type the following commands in the git bash (use the link of the repository in the command).

d. Thus,the task is complete

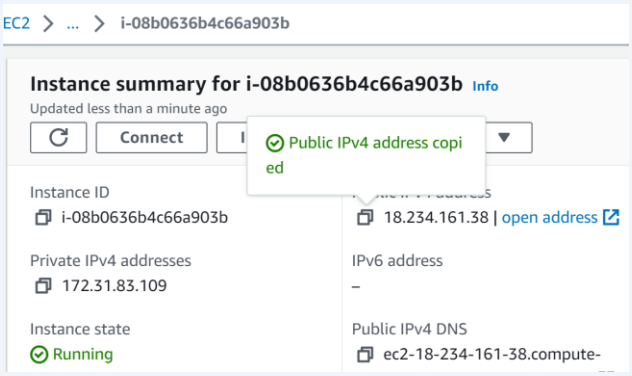
```
hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/N1
$ git init
Initialized empty Git repository in C:/Users/hites/OneDrive/Documents/onedrive/Desktop/N1/.git/

hites@LAPTOP-Q12SOP5J MINGW64 ~/OneDrive/Documents/onedrive/Desktop/N1 (master)
$ git clone https://github.com/sudip7407/New-Repo1.git
Cloning into 'New-Repo1'...
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 15 (delta 6), reused 4 (delta 0), pack-reused 0
Receiving objects: 100% (15/15), done.
Resolving deltas: 100% (6/6), done.
```



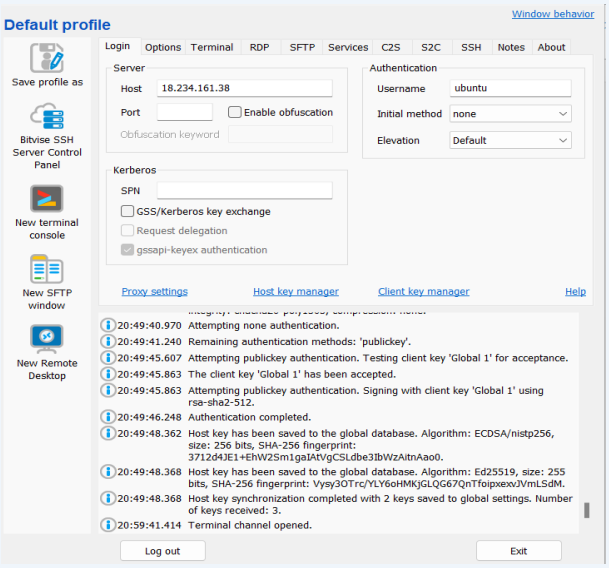
Assignment No. 9

DEPLOY A PROJECT FROM GITHUB TO EC2

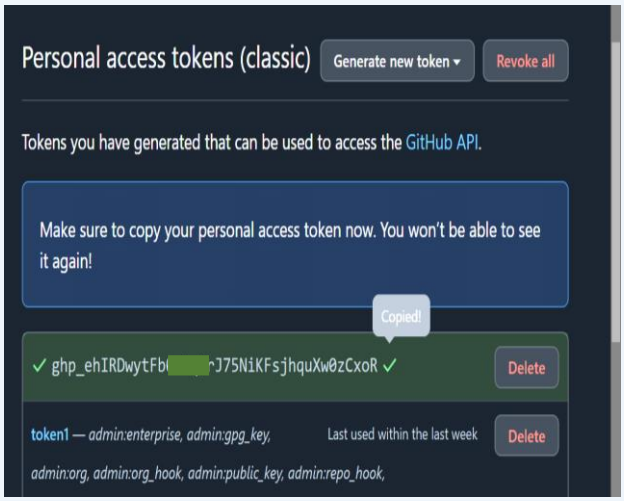
1. **Visit aws.amazon.com and Sign in.Create an EC2 server and copy its IPv4 address.**



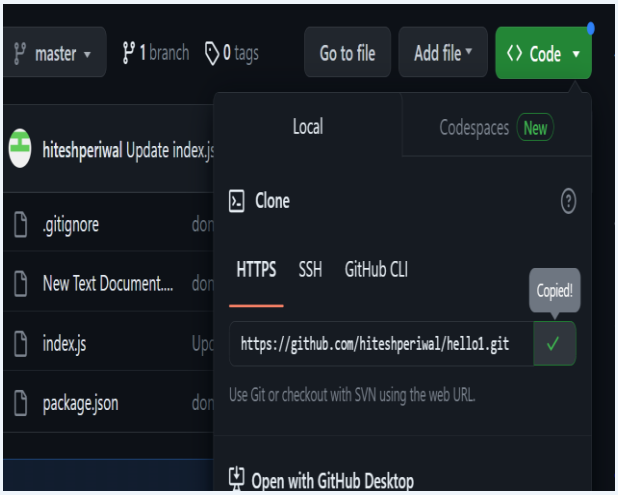
2. **Login to Bitwise SSH client using the copied IPv4 address as host address.**



3. **Create a personal access token in the Github and save it for future reference**



4. **Create a repository containing the project files in the Github and copy and save its HTTPS address for future reference**



5. **Open a "New terminal console" in the Bitwise SSH client and type the following commands in it to install nodejs:-**

pwd *(to check if it is in server)*
sudo apt-get update
sudo apt-get upgrade
sudo apt install nginx
nginx –v *(to check nginx version)*
curl –s1 https://deb.nodesource.com/setup_18.x |sudo -E bash -
sudo apt install nodejs
node –v *(to check nodejs version)*

```
ubuntu@ip-172-31-83-109:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-83-109:~$ sudo apt-get update
```

```
ubuntu@ip-172-31-83-109:~$ sudo apt-get upgrade
```

```
ubuntu@ip-172-31-83-109:~$ sudo apt install nginx
```

```
ubuntu@ip-172-31-83-109:~$ nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
ubuntu@ip-172-31-83-109:~$ curl -s1 https://deb.nodesource.com/setup_18.x |sudo -E bash -
```

```
ubuntu@ip-172-31-83-109:~$ sudo apt install nodejs
```

```
ubuntu@ip-172-31-83-109:~$ node -v
v18.15.0
```

6. **To upload the project on EC2 server type the following commands:-**

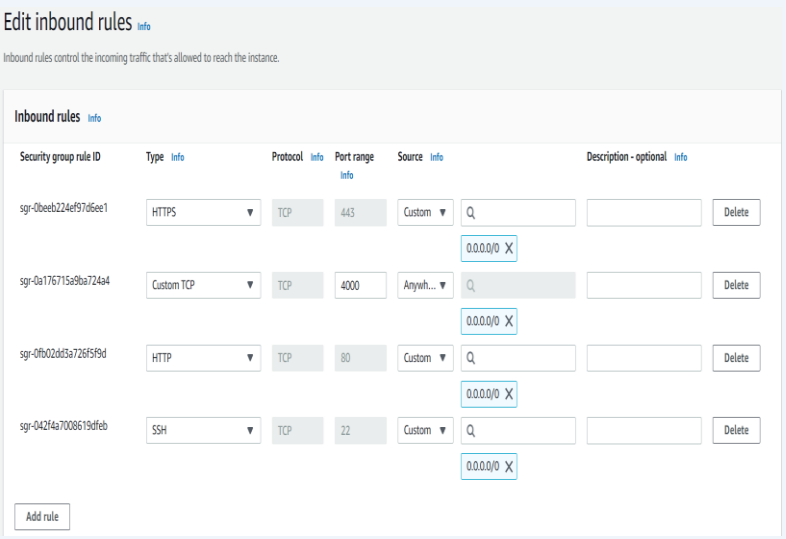
git clone ***https address of the github hub repository where project is uploaded***
(For username type your github username and for password enter the personal access token)
dir *(to check the directory of the project)*
cd hello1/ (to enter into repository)
npm install *(node package manager installation to run node commands)*
node index.js *(as project is uploaded start server)*

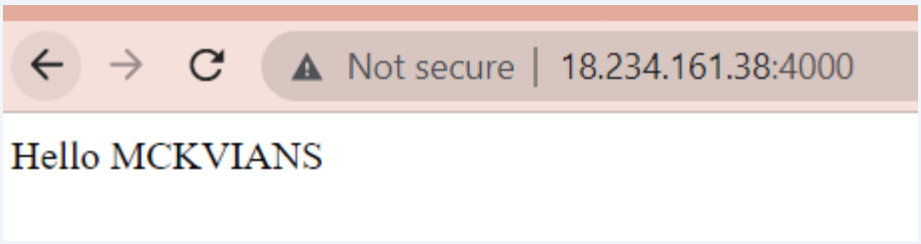
```
ubuntu@ip-172-31-82-194:~$ git clone https://github.com/hiteshperiwal/hello1.git
Cloning into 'hello1'...
Username for 'https://github.com': hiteshperiwal
Password for 'https://hiteshperiwal@github.com':
```

```
ubuntu@ip-172-31-83-109:~$ dir
hello1
ubuntu@ip-172-31-83-109:~$ cd hello1/
ubuntu@ip-172-31-83-109:~/hello1$ npm install
```

```
ubuntu@ip-172-31-83-109:~/hello1$ node index.js
Started server
```

7. **On pasting the public Pv4 address address of EC2 server ,the project wont load because only through specific ports it is accessible.Go to git hub and check for the ports permitted for the project by opening the index.js file.**
Now to address this issue go to instance(in aws) -> security -> security groups -> edit inbound rules -> Add rules -> set type as custom tcp , port range as specified in index.js file, source as anywhere-IPv4 and 0.0.0.0/0 ->save rules.Now type the port number after colon in IPv4 address and the website will be accessible.
<u>Hence the project is successfully deployed from github to EC2.</u>

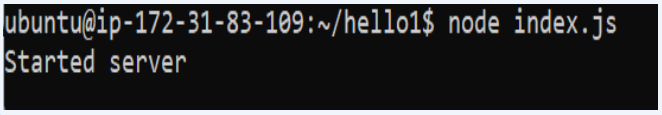


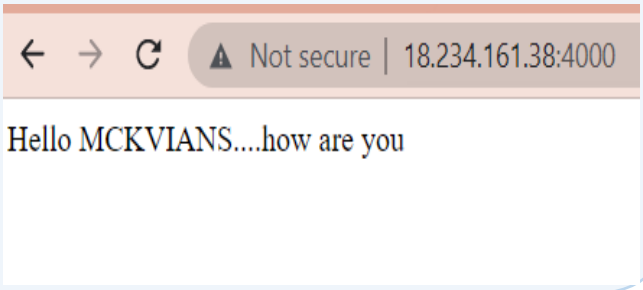


8. **If there is some editing done in the project it will not be reflected in the website.For that we need to terminate the server using ctrl+c and then type the follow command:-**

git pull
(enter the github username and its token as password)
node index.js

Now,refresh the browser to see the changes reflected on it.

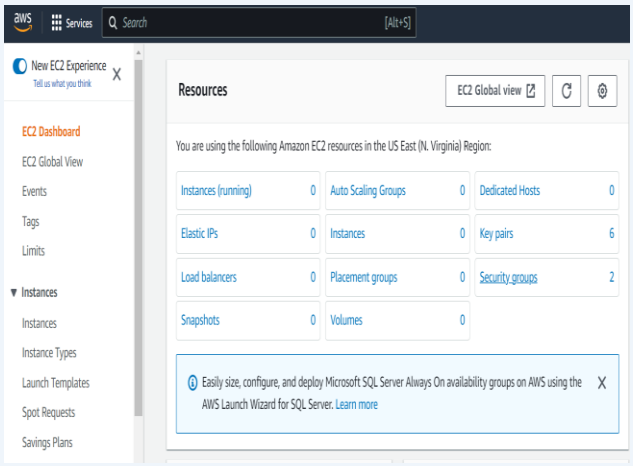




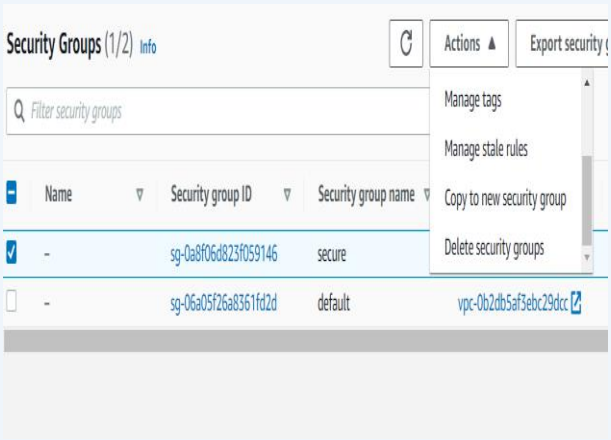
Assignment No. 9

DEPLOY A PROJECT FROM GITHUB TO EC2 BY CREATING NEW SECURITY GROUP AND USER DATA

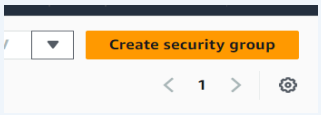
1. **Visit aws.amazon.com and Sign in.Go to EC2 Service and click on “Security Groups”**

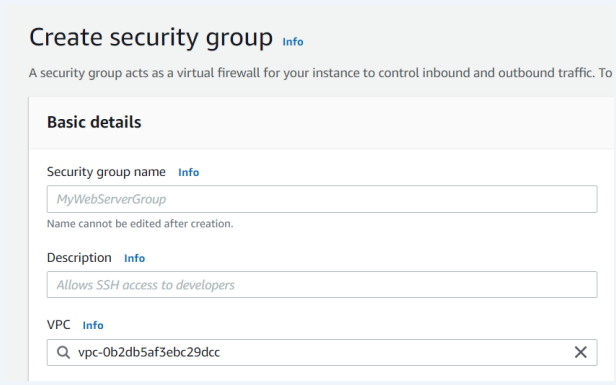


2. **Select all security groups except the default and go to Actions dropdown menu and choose “Delete security groups”**

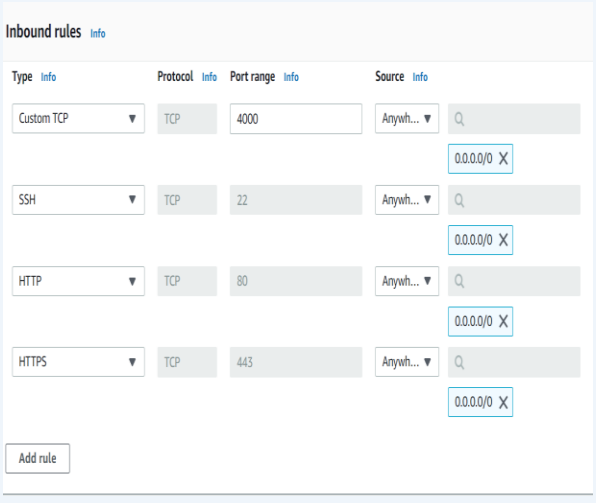


3. **Click on “Create security group”.Enter the “Security group name” and “Description”**

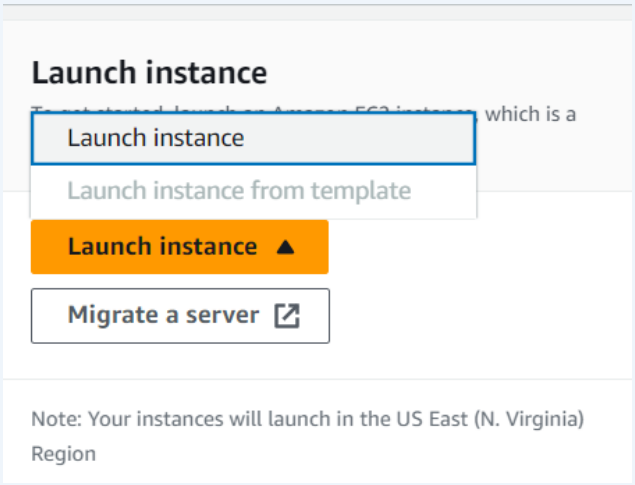




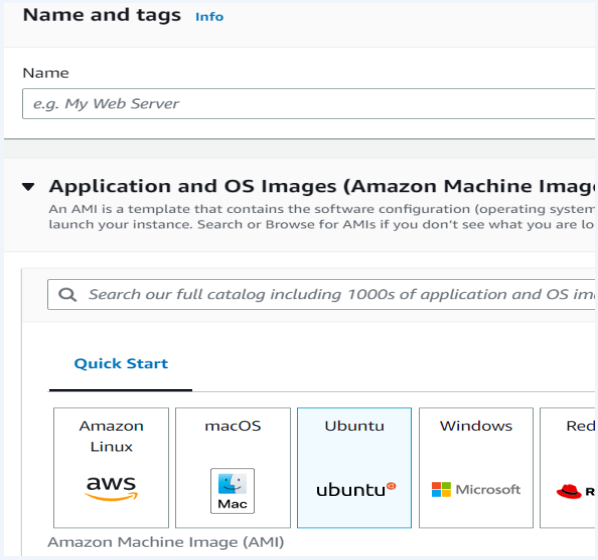
4. **Add the Inbound rules with details as shown in the figure.After entering the details click on “Create security group”**



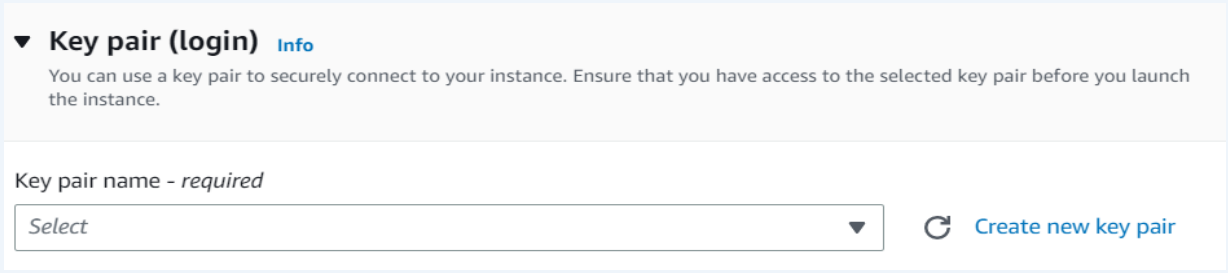
5. **Go to EC2 dashboard and Click on "Launch Instance"**



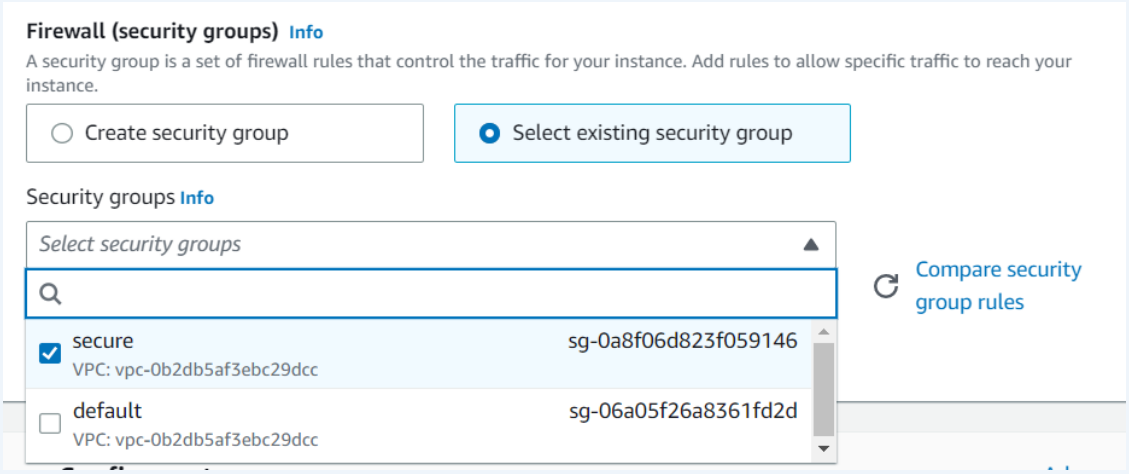
6. **Enter the instance name and select Ubuntu as Amazon Machine Image**



7. **Select key pair**



8. **Under network settings ,click on "Select existing security group" and in the security group dropdown select the security group which you just created**



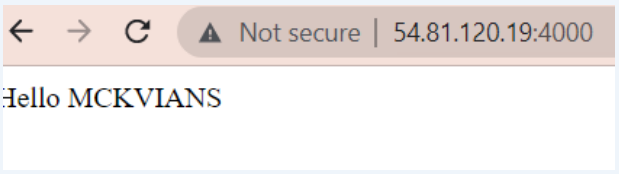
9. **Type the following user data under advanced details section.Then click on create instance.**

User data - *optional* Info

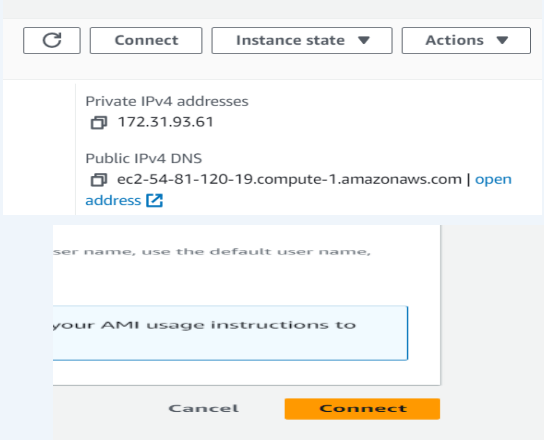
Enter user data in the field.

```
#!/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
git clone https://github.com/hiteshperiwal/hello1.git
cd ...
npm install
node index.js
```

10. **Now using the public ip address of the instance and the corresponding port number of the project ,we can access the project.**



11. **However for certain machines the project will not be accessible directly.To access the project,go to instance and click on connect and again click on connect**



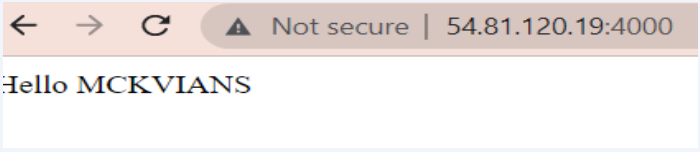
12. **Now type the following commands in the terminal.Keep the repository url and the github personal access token ready as they will be required in these commands**

```
ubuntu@ip-172-31-93-61:~$ nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
ubuntu@ip-172-31-93-61:~$ npm -v
9.5.1
ubuntu@ip-172-31-93-61:~$ node -v
v18.16.0
ubuntu@ip-172-31-93-61:~$ git clone https://github.com/hiteshperiwal/hello1.git
Cloning into 'hello1'...
Username for 'https://github.com': hiteshperiwal
Password for 'https://hiteshperiwal@github.com':
```

```
ubuntu@ip-172-31-93-61:~$ ls
hello1
ubuntu@ip-172-31-93-61:~$ cd hello1/
ubuntu@ip-172-31-93-61:~/hello1$ npm install
```

```
ubuntu@ip-172-31-93-61:~/hello1$ node index.js
Started server
```

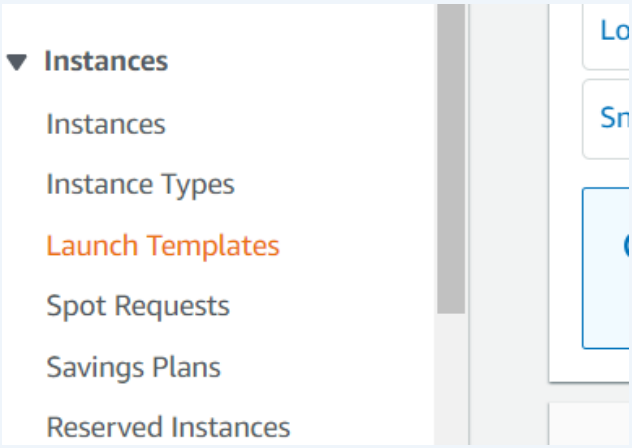
13. **Now using the public ip address of the instance and the corresponding port number of the project ,we can access the project.Hence the project is successfully deployed from github to EC2 using new security group and user data**



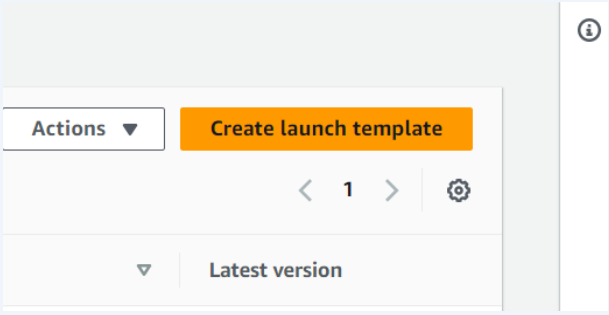
Assignment no. 11

Build scaling plans in AWS that balances load on different EC2 instances

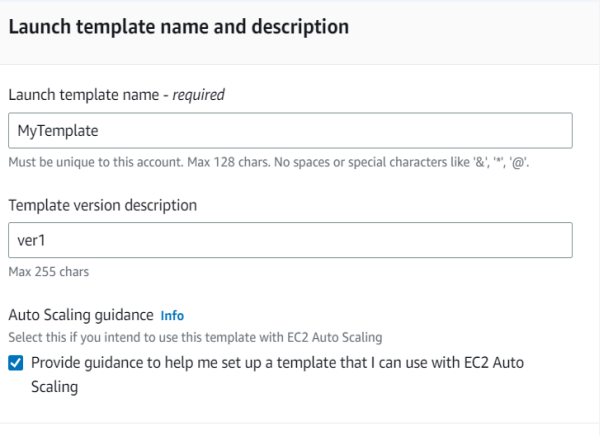
1. Visit aws.amazon.com and Sign in.Go to EC2 dashboard and click “Launch Template”



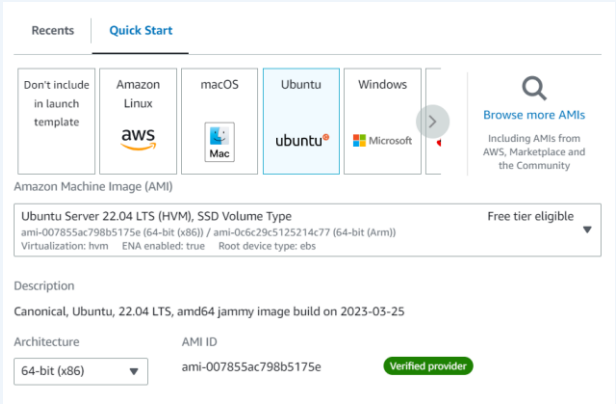
2. Click on “Create launch instance” option



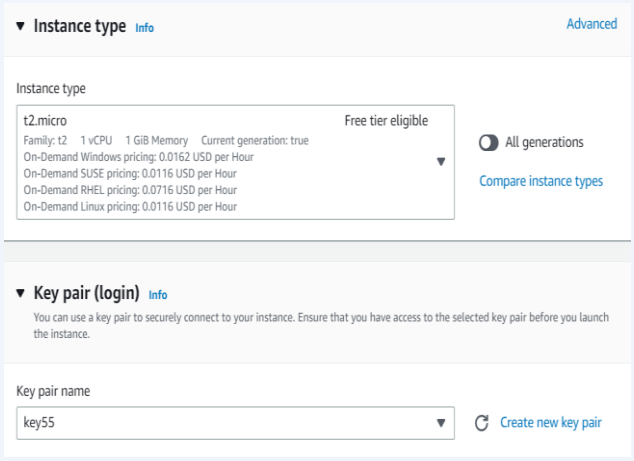
3. Give template name,template version description and select autoscaling guidance



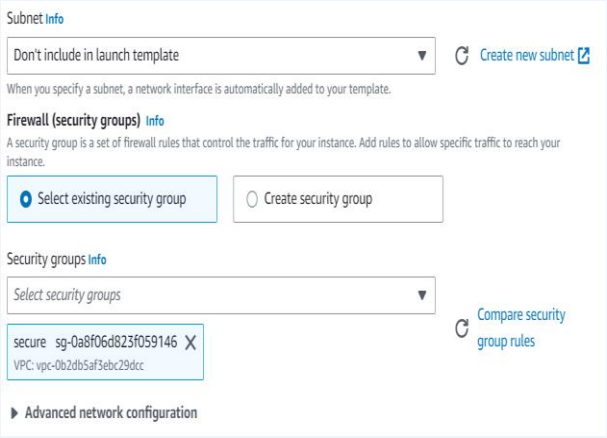
4. Select Ubuntu as AMI



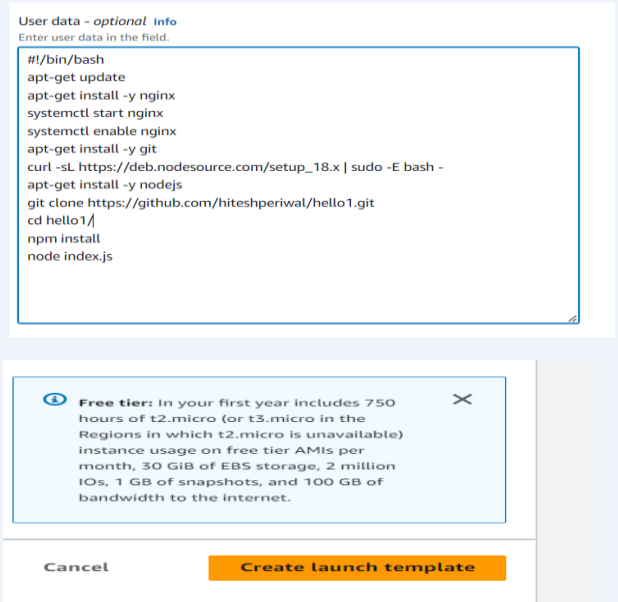
5. Select “t2.micro” as instance type and key pair for login



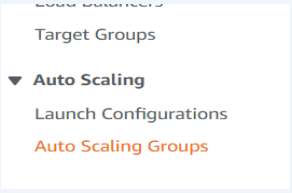
6. Now select the existing security group which you created for the project

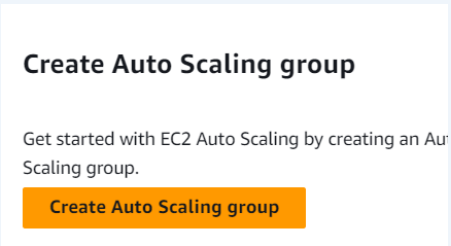


7. Type the following user data and click on create launch template

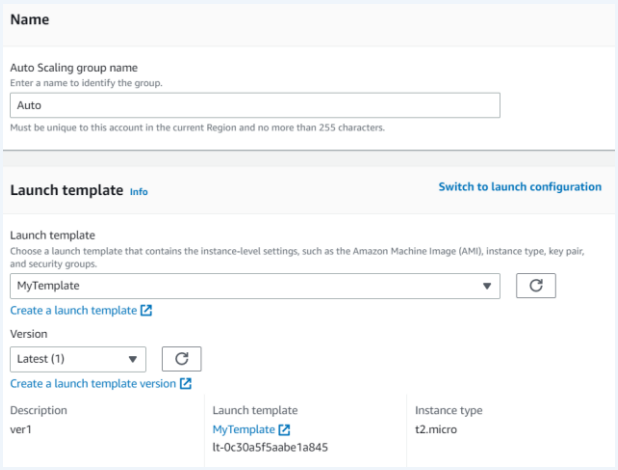


8. In the EC2 dashboard click on “Auto scaling groups and then click on create Auto Scaling groups”

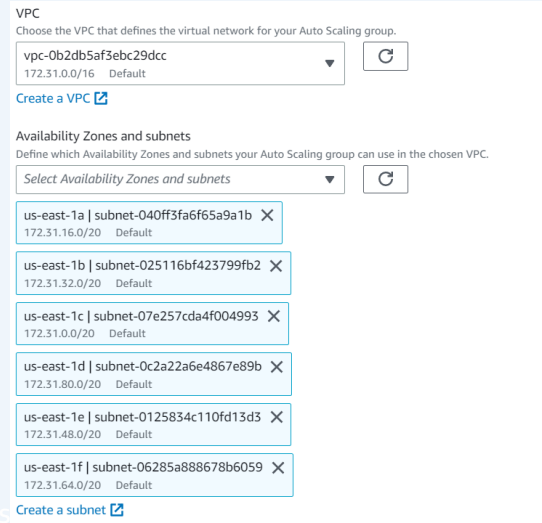




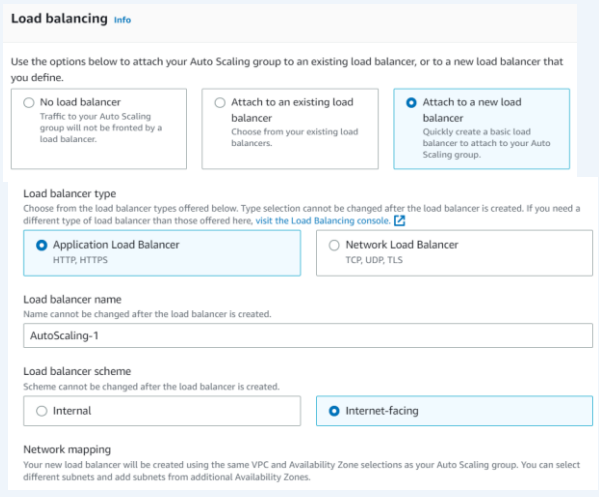
9. **Enter the Auto Scaling group name.Select the template created by you and its version as latest.Click on “Next”**



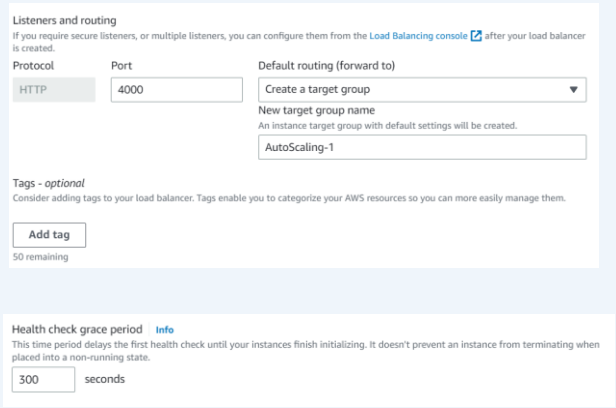
10. **Select all the Availability zones and subnets and click on “Next”**



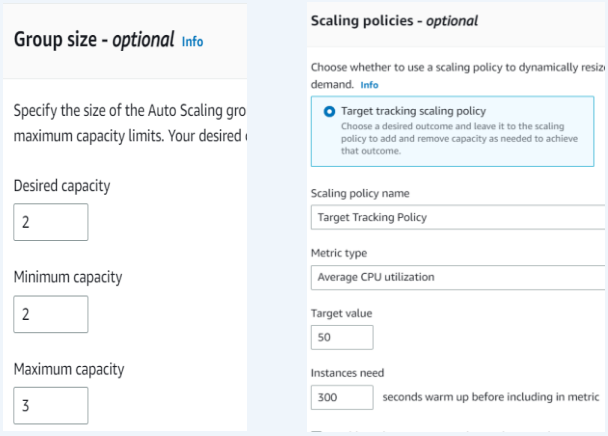
11. **In the load balancing ,select “Attach to a new load balancer”.In the load balancer type select “Application load balancer”.Give the load balancer name and select the load balancer scheme as internet facing**



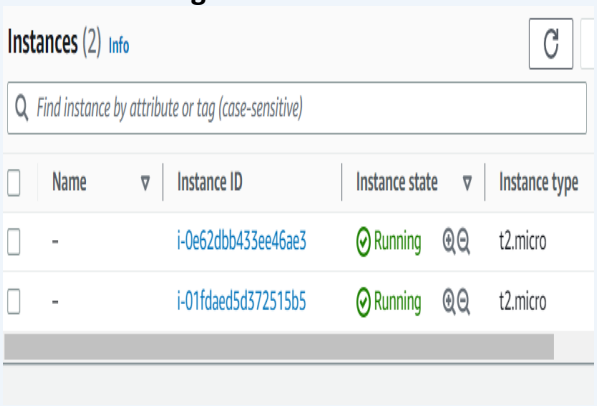
12. **In the listeners section, enter the port as mentioned in your project and health check grace period as**



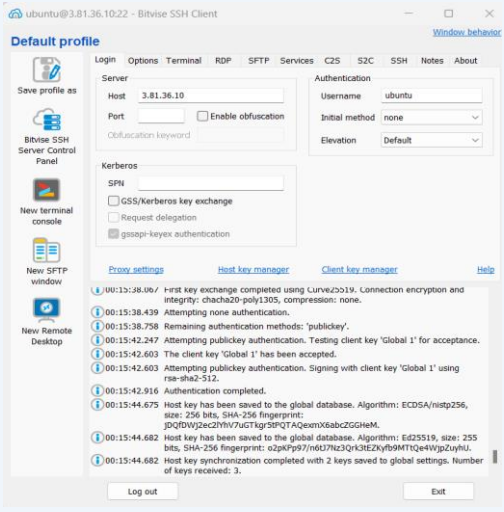
13. Enter the desired capacity as 2,minimum capacity as 2 and maximum capacity as 3.Select target tracking scaling policy,give it a name,target value 50 and 300 seconds as warm up .Click on “Next”,”Next”,”Next” and then “Create auto scaling group”.



14. Now two instances are running.Overload one of them to see whether auto scaling functionality is working or not.



15. Login in Bitwise SSH client using one of the public ip address of the instances created and the key pair used in those instance.



16. Open the terminal console,create a shell executables file,type the following commands and execute the file to overload the server

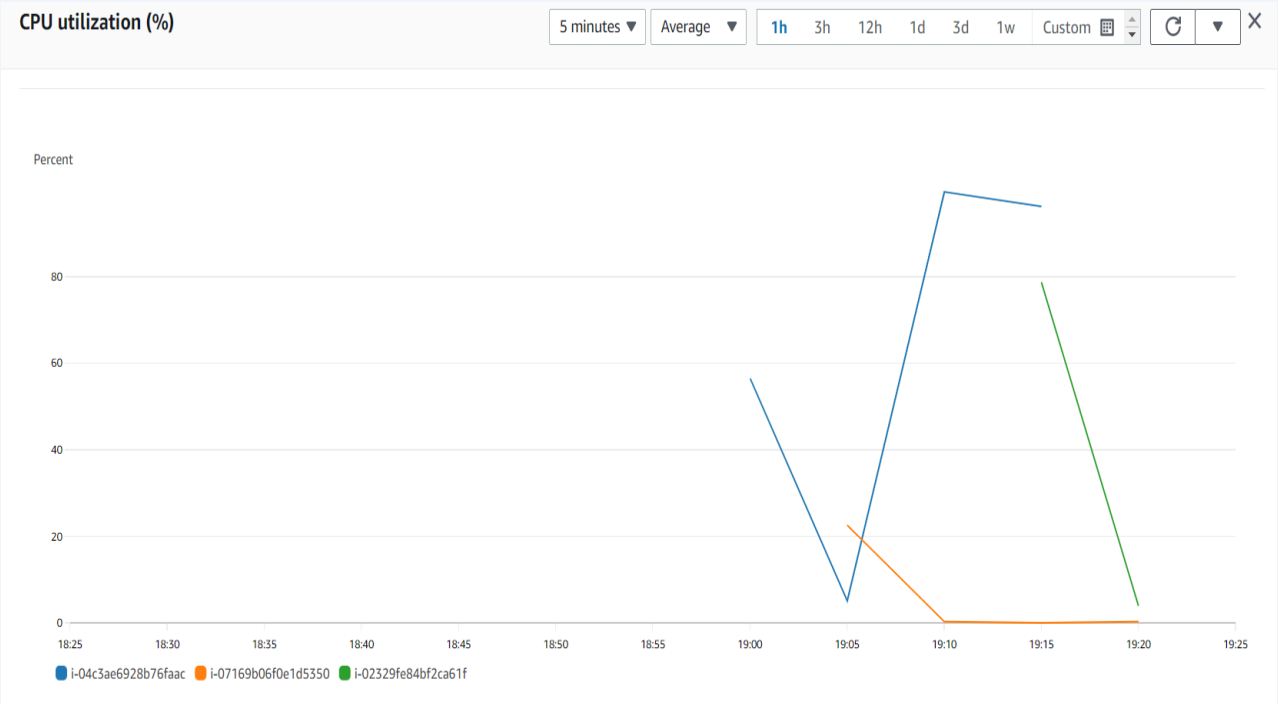
```
ubuntu@ip-172-31-68-249:~$ vi inf1.sh
```

```
#!/bin/bash
while true
do
        echo "Looping forever"
            # Add other commands to run in the loop here
done
~
~
```

```
ubuntu@ip-172-31-68-249:~$ chmod +x inf1.sh
ubuntu@ip-172-31-68-249:~$ ./inf1.sh
```

```
Looping forever
Looping forever
Looping forever
Looping forever
Looping forever
Looping forever
Looping forever
Looping forever
```

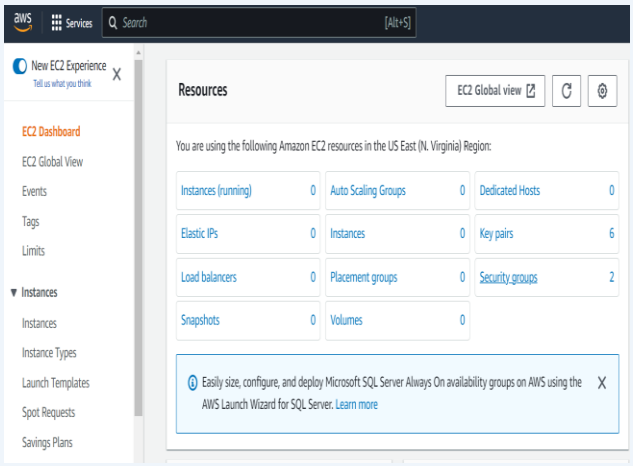
17. **Select the running instances and click on monitoring.You will observe creation of new instance during the 300 seconds warm up when the server is overloaded.Hence load is balanced on different EC2 instances**



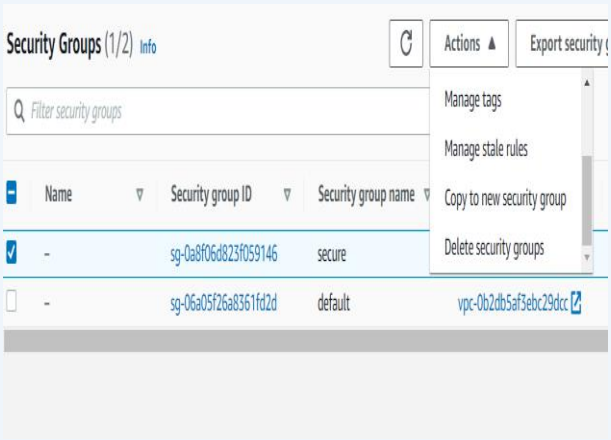
Assignment No. 12

Deploy and run project in aws without using port

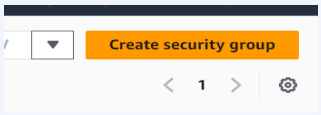
1. **Visit aws.amazon.com and Sign in.Go to EC2 Service and click on “Security Groups”**

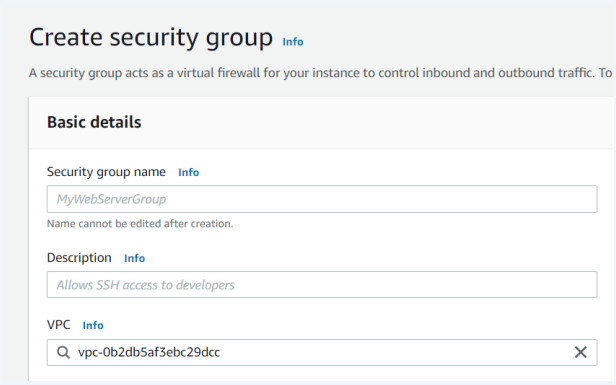


2. **Select all security groups except the default and go to Actions dropdown menu and choose “Delete security groups”**

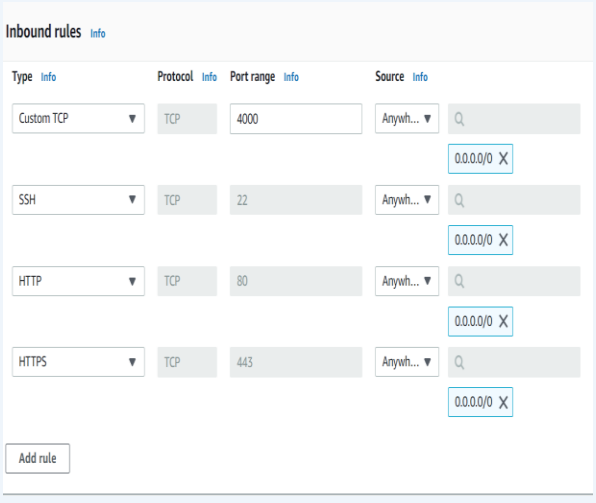


3. **Click on “Create security group”.Enter the “Security group name” and “Description”**

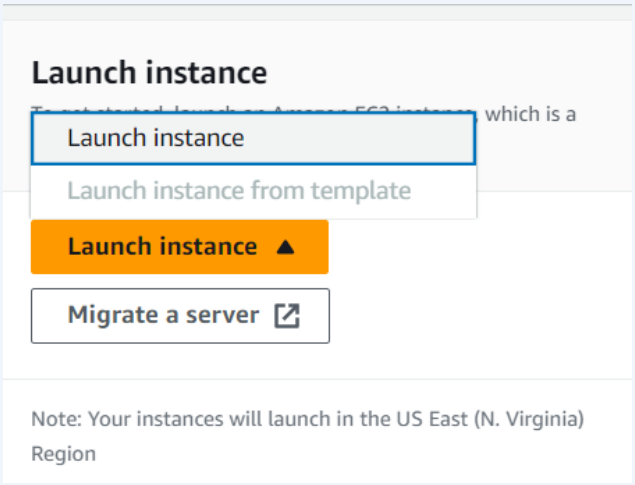




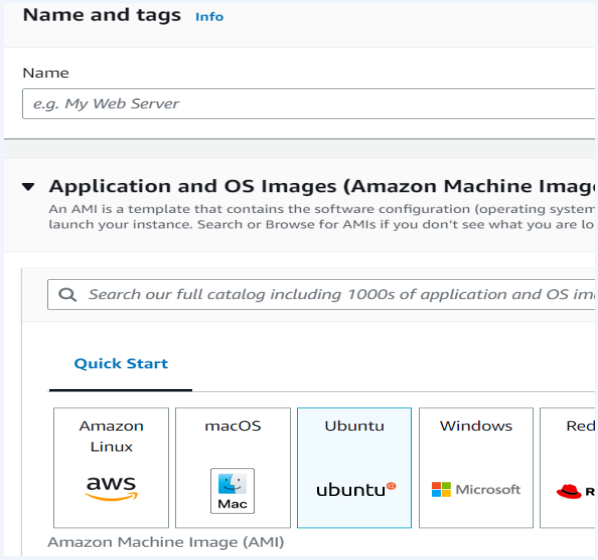
4. **Add the Inbound rules with details as shown in the figure.After entering the details click on “Create security group”**



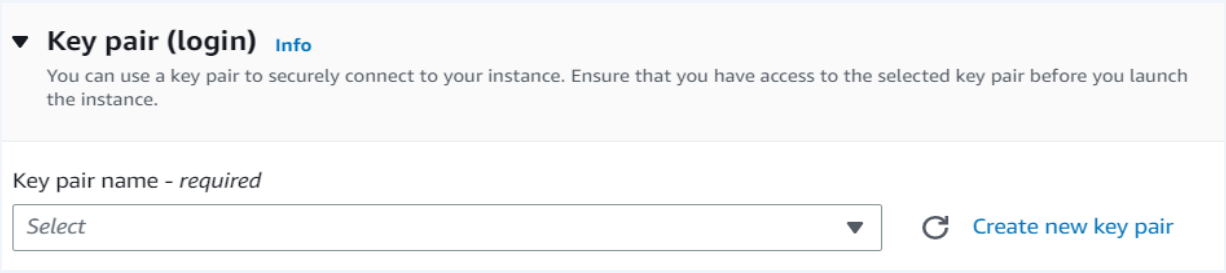
5. **Go to EC2 dashboard and Click on "Launch Instance"**



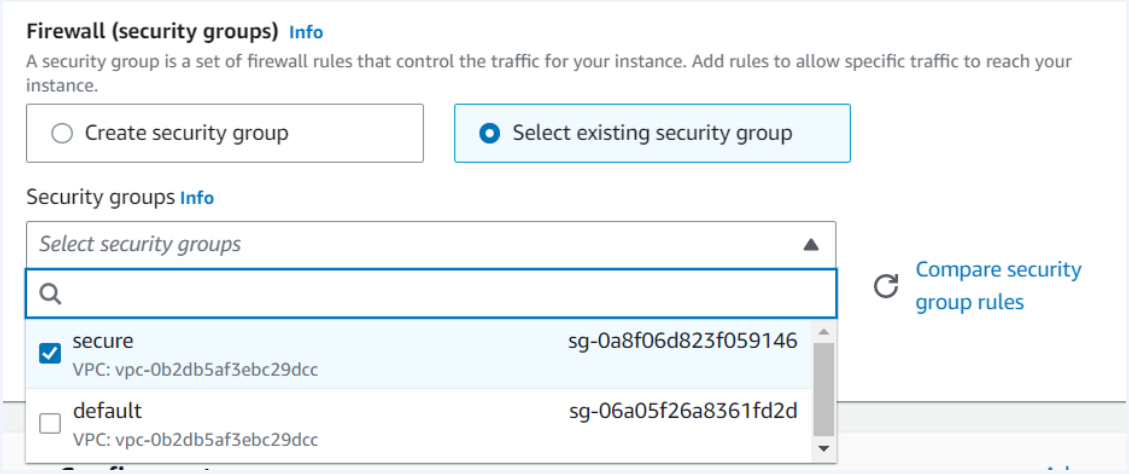
6. **Enter the instance name and select Ubuntu as Amazon Machine Image**



7. **Select key pair**



8. **Under network settings ,click on "Select existing security group" and in the security group dropdown select the security group which you just created**



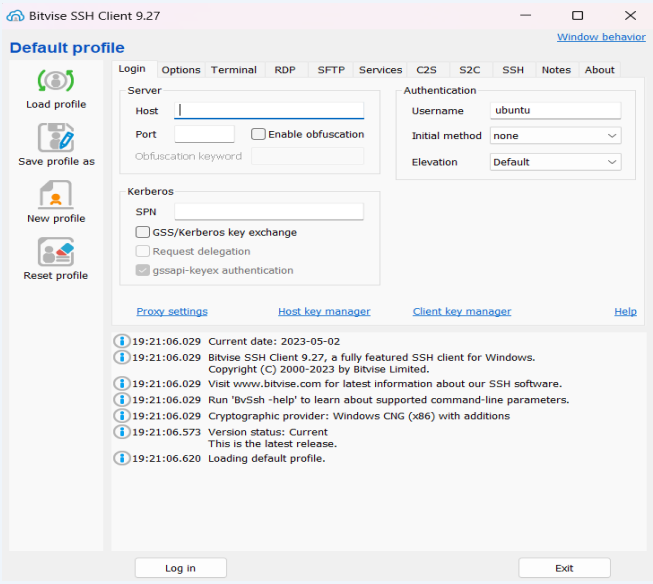
9. **Type the following user data under advanced details section.Then click on create instance.**

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
git clone https://github.com/hiteshperiwal/hello1.git
cd ...
npm install
node index.js
```

10. **Login to Bitwise SSH client using the public ip address.**



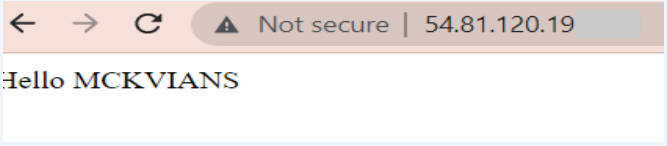
11. **Now login in Bitwise SSH client using the public ip address.Open the terminal and type the following commands and then open nano editor.In the nano editor after commenting the previous location type the new location as given below and then save and exit.**

```
ubuntu@ip-172-31-87-57:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-87-57:~$ cd /
ubuntu@ip-172-31-87-57:/$ pwd
/
ubuntu@ip-172-31-87-57:/$ cd /etc/nginx/sites-available/
ubuntu@ip-172-31-87-57:/etc/nginx/sites-available$ sudo nano default
```

```
location / {
        proxy_pass http://localhost:4000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'Upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
```

12. **After exiting the nano editor,type the following command in the terminal to restart the nginx web server.After restarting the nginx web server you will be able to access the website without the port number . Hence we are running project in aws without using port number.**

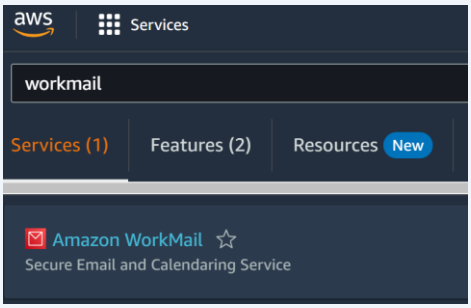
```
ubuntu@ip-172-31-87-57:/etc/nginx/sites-available$ sudo systemctl restart nginx
ubuntu@ip-172-31-87-57:/etc/nginx/sites-available$
```

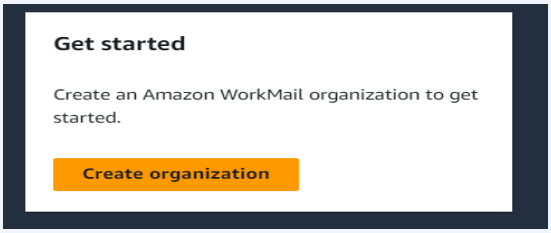


Assignment No. 13

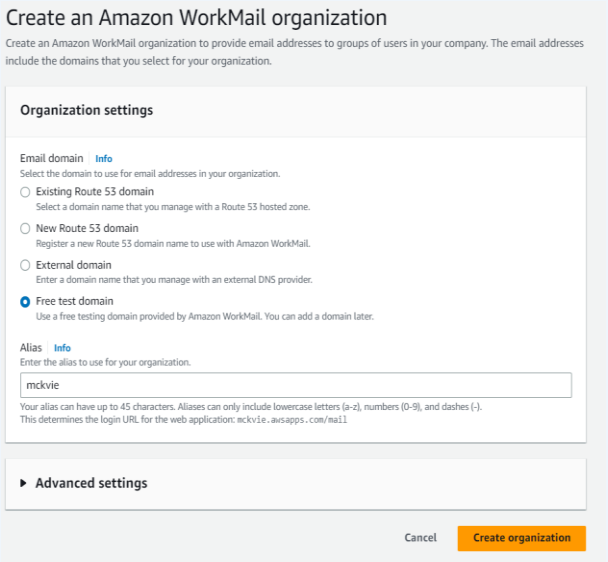
Create a workmail for your organization

1. **Visit aws.amazon.com and Sign in.Click on the search field and type “workmail” in the search field and select Amazon WorkMail. Click on “Create organization”**

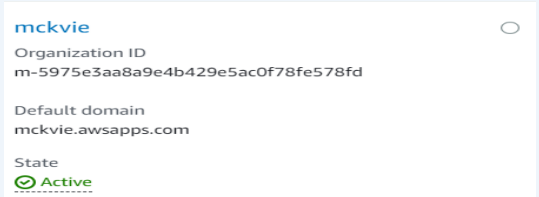


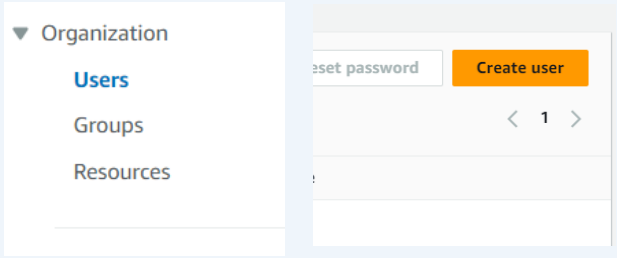


2. **Select “Free test domain” in the organization settings and click on “Create organization”**

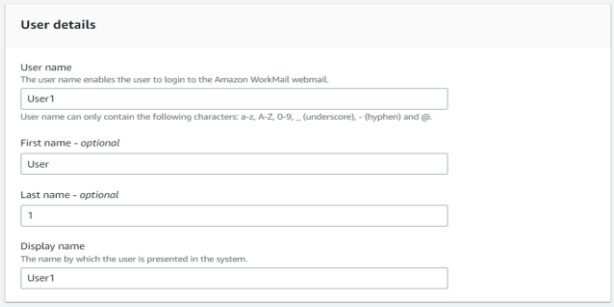


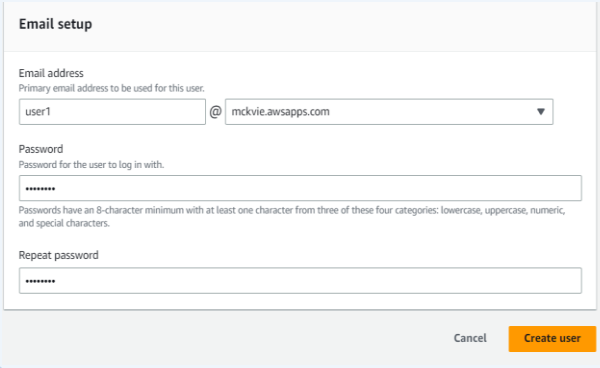
3. **Click on the organization which was just created.On the left panel select users and then click on “Create user”**



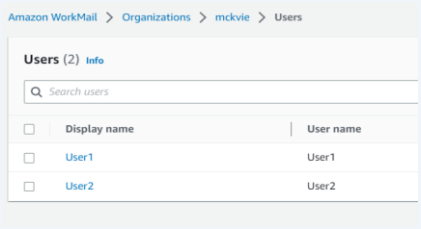


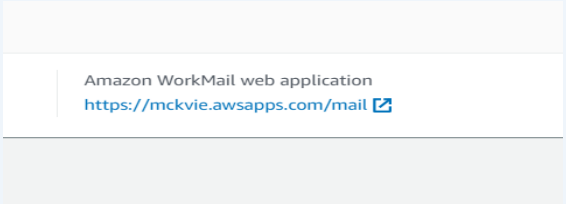
4. **Setup the user email and click on “Create user”**



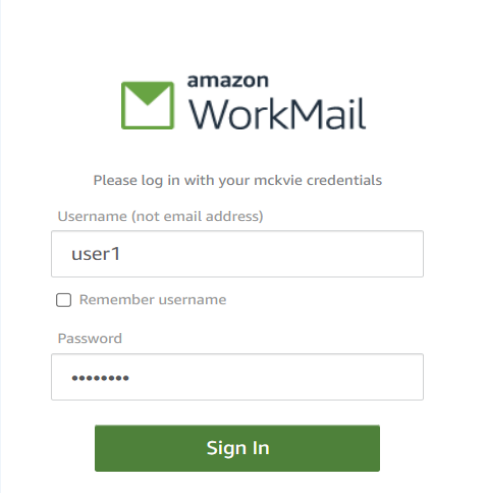


6. **Create two such users.Go back to organization dashboard and click on the organization created by you. Now click on the Amazon WorkMail web application link.**

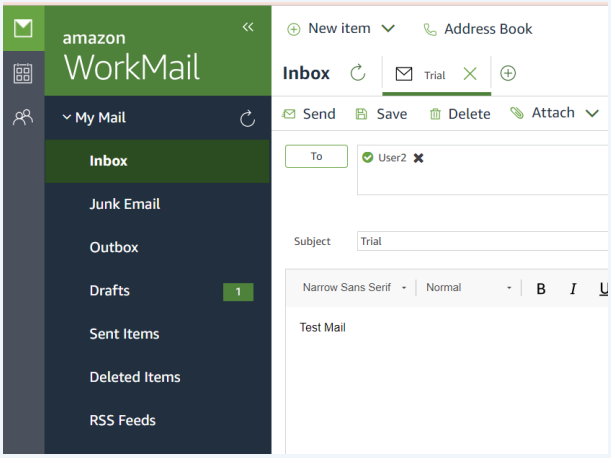




7. **Login to the amazon workmail by any of the user of your organization's workmail**

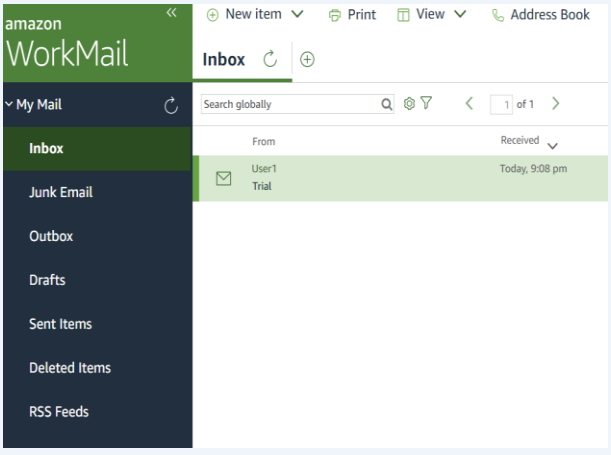


7. **Send an email from one user account of the organization to the another.**



8. **Check if the email is received by the other user of the same organization.**

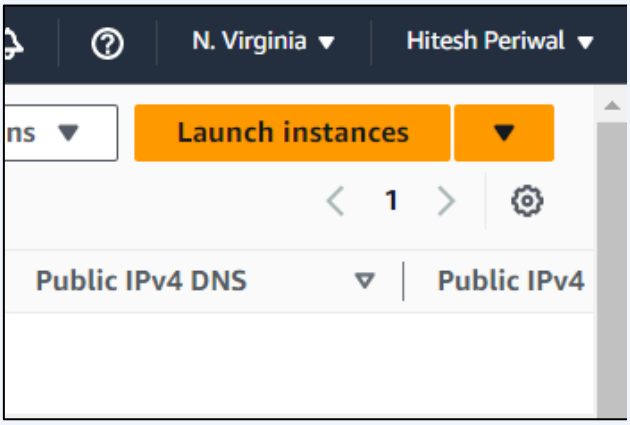
Hence organization's workmail is setup successfully.



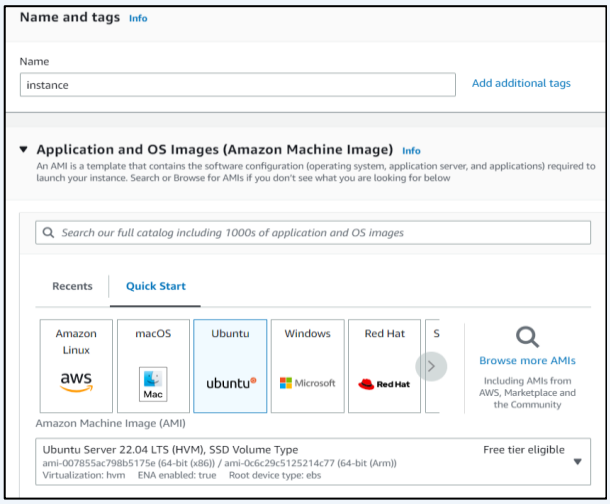
Assignment No. 14

Create an elastic IP for an instance.

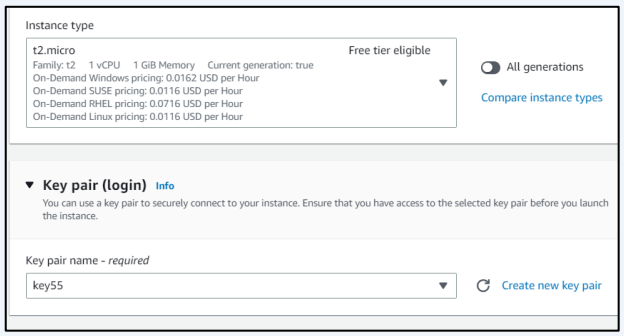
1. Visit aws.amazon.com and Sign in.Go to EC2 dashboard and click “Instance” and then click on “Launch instances”



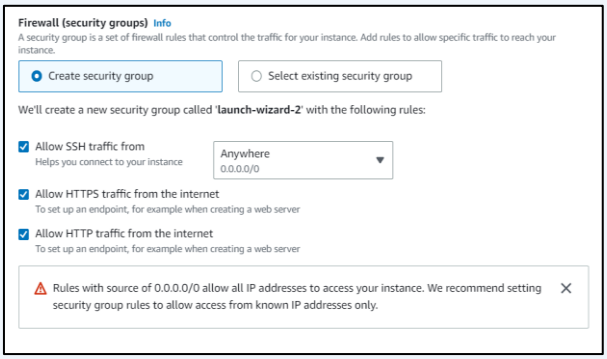
2. Enter instance name and select Ubuntu as AMI

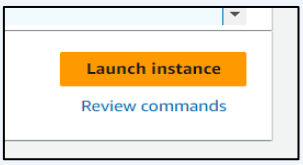


3. Select t2.micro as instance type and a keypair

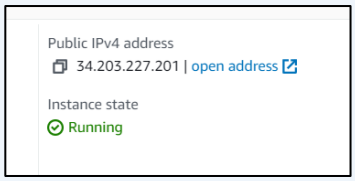


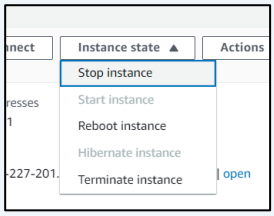
4. Under firewall select create security group and select all the three traffic options

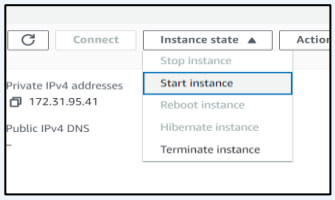


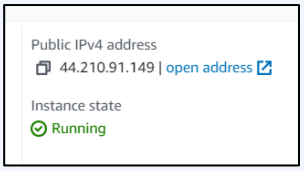


5. Now go to the instance created .Change its state to stop.After its successfully stopped ,again start it.It will be observed that its public IPv4 will change.

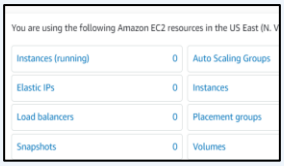


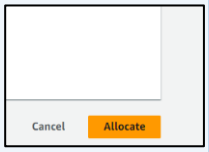




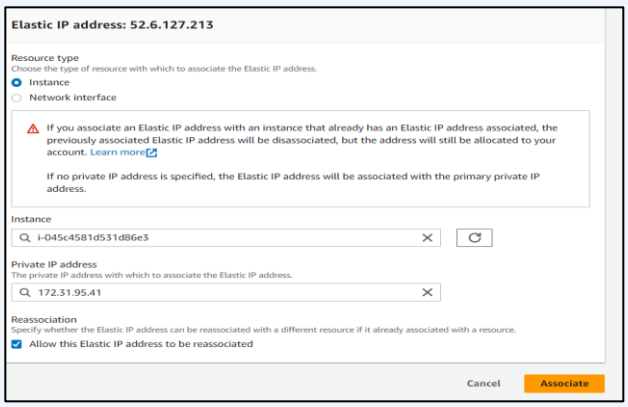


6. Go to EC2 dashboard and select “Elastic IPs”.Then click on “Allocate Elastic IP address” and “Allocate”. Select the elastic ip address,go to “Actions” dropdown and select “Associate Elastic IP addresses”.

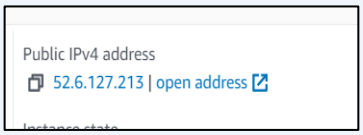


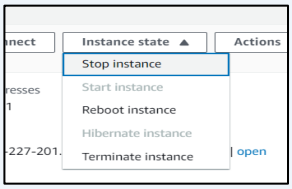


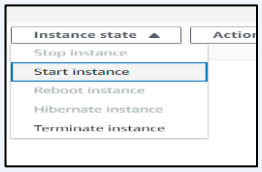
7. Now select the instance and its private IP address.Click on checkbox which reads “Allow this Elastic IP address to be reassociated”.Then click on “Associate”

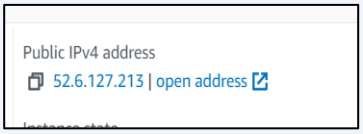


8. Now go to the instance.Change its state to stop.After its successfully stopped ,again start it.It will be observed that its public IPv4 will remain same.Hence Elastic IP address is successfully associated with this instance.

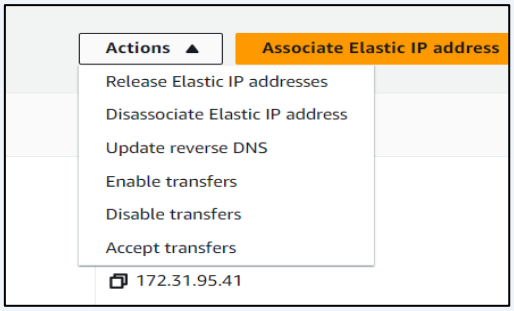








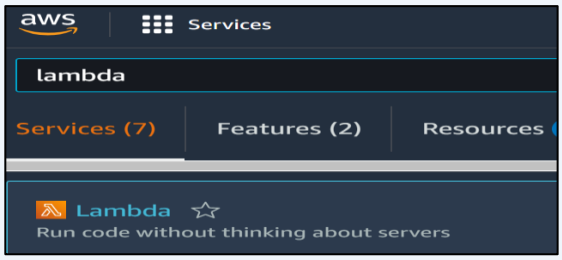
9. To remove this Elastic IP address we need to first disassociate it and then release it.



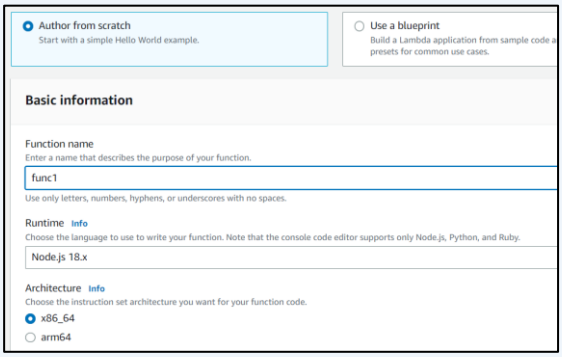
Assignment No. 15

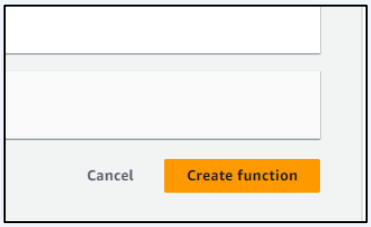
Create serverless computing service

1. Visit aws.amazon.com and Sign in.Go to Search box,type lambda and click on “Lambda”.Now click on “Create function”

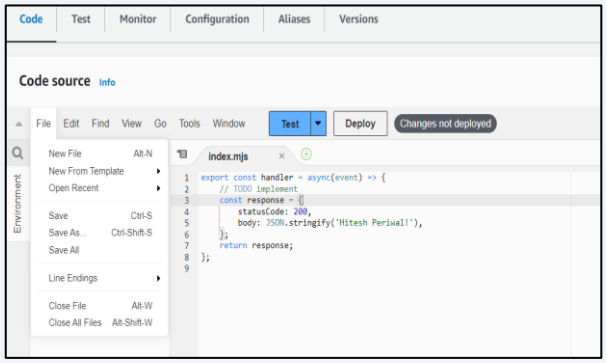


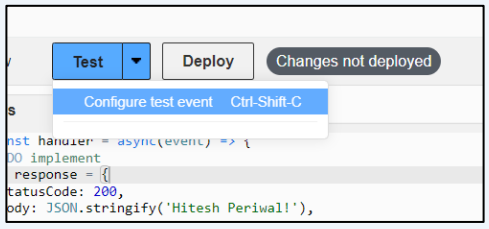
2. Type funtion name,select Runtime as “Node.js 18.x” and architecture as “x86_64”.Click on “Create function”



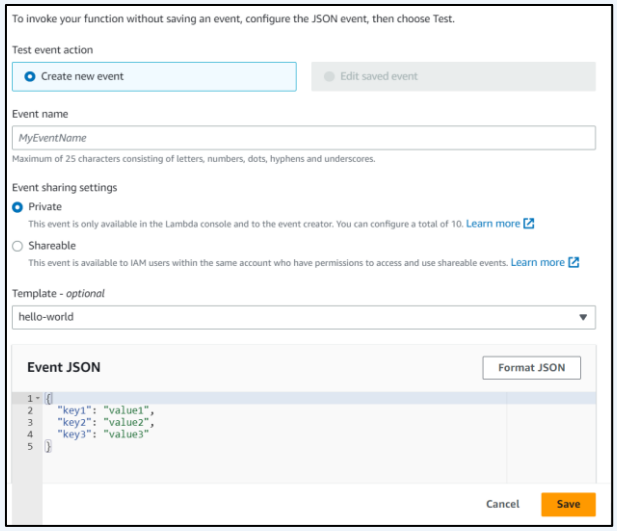


3. Go to Code section.You may edit the code and save it.Click on Test dropdown and select “Configure test event”

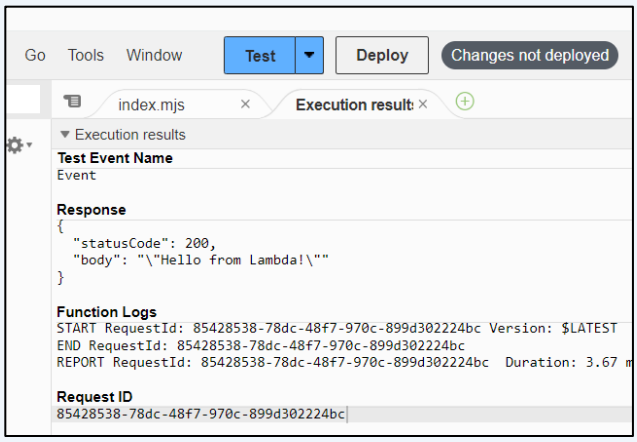




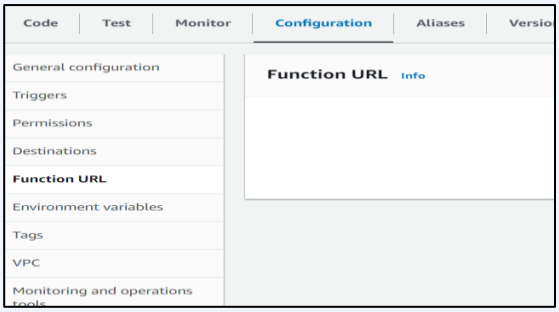
4. Select “Create new event”,give event name,event sharing as private,select any template(optional) and click on “Save”

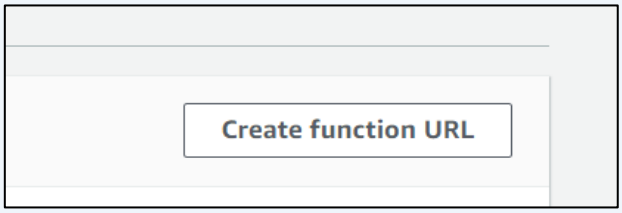


5. Now test the event by clicking on “Test” and deploy it by clicking on “Deploy”

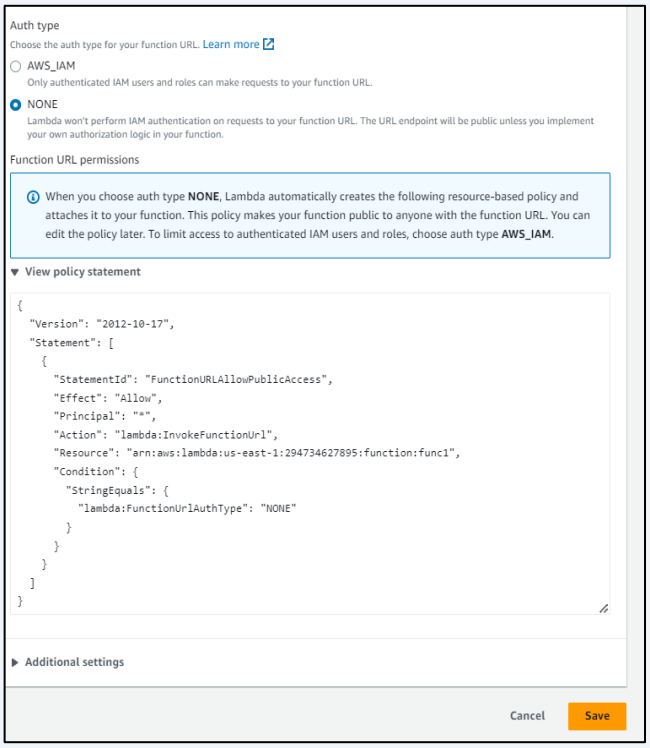


6. Go to configuration ,select Function URL and click on “Create function URL”

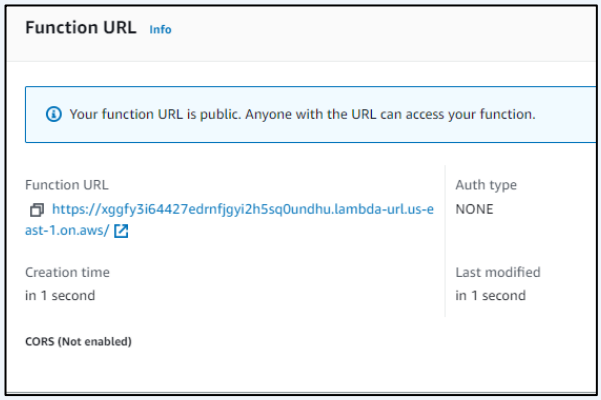


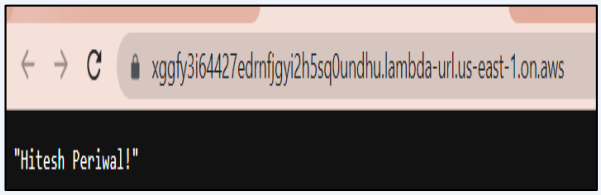


7. Select Authorization type as “NONE” and Click on“Save”



8. Function URL is thus generated.You can use it access and view your deployed project.

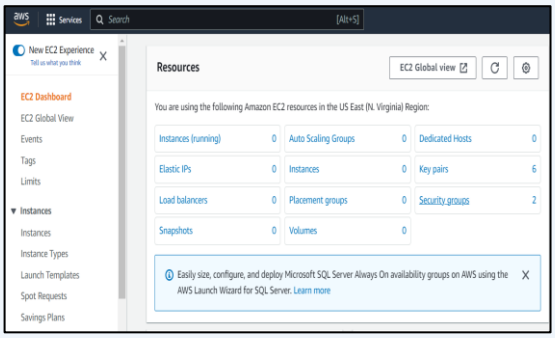


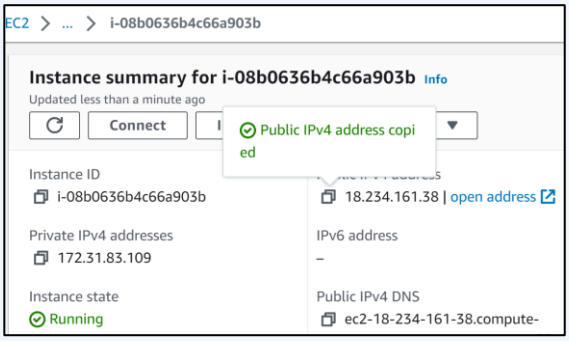


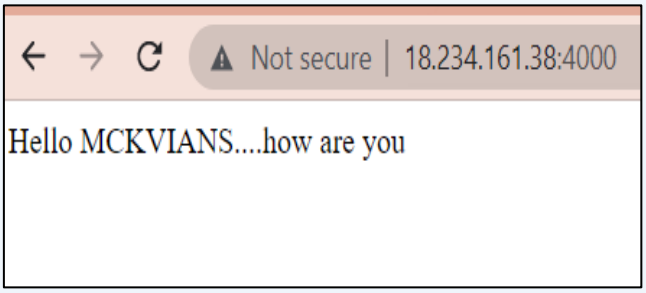
Assignment No. 16

Manage Amazon DNS service and run a project using domain-name and URL

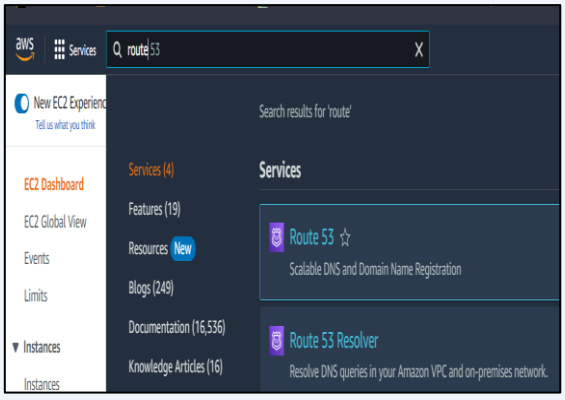
1. **Visit aws.amazon.com and Sign in.**
2. **Go to EC2 Service**
3. **Create an instance with custom security group and user data.(As done in Assignment 10)**
4. **Click on the instance and copy the public IPv4 address and paste in the browser.**
5. **Check accessibility of project by appending port number after the public IPv4 address.**





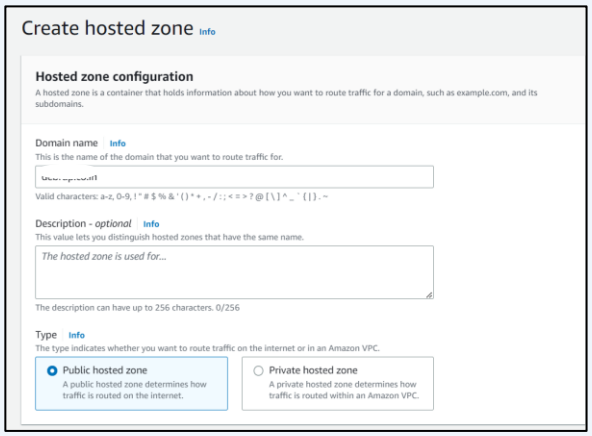


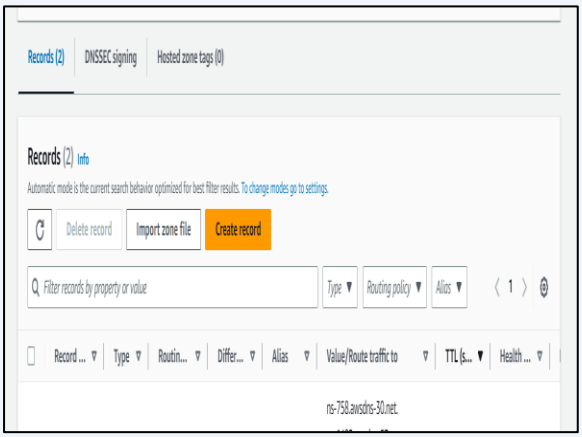
6. Our EC2 instance works as intended. However, to access the webpage one always requires the public IPv4 address of the server instance which is very complicated/less accessible for end-users of our webpage/web application. To make it easier for our end-users, we need to bind a domain name to the server instance so that anyone can use the domain name and the URL to access the project.
Now Search "Route 53" in the search bar of AWS console and click on "Route 53"



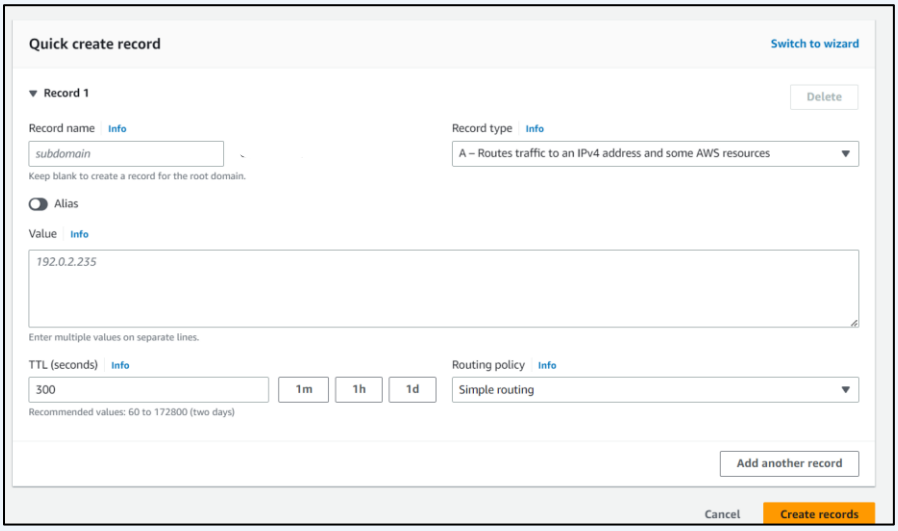
7. We require a registered Domain name for this assignment. So, after obtaining one (free or paid) go to the Webpage of your Domain provider and log-in to your account where you can find all the details of your purchased Domains.
This may vary from site to site, so you will have to do this based on what site you are using. We (for now) will be using GoDaddy.com, because we have purchased a Domain from them.

8. In the Route 53 dashboard "Click on "Create Hosted Zone button".(Alternatively, you can go to hosted zones from the left-side bar and then select create hosted zone option.)
9. Now, copy your Domain name from your Domain providers website. Here we used GoDaddy.com. Paste the domain name in the given field in Hosted Zone configuration page.
10. Click on "Create Hosted Zone".
11. Click on "Create a record button".

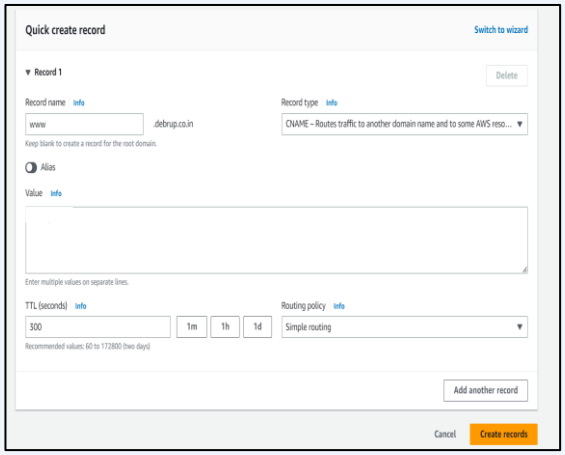


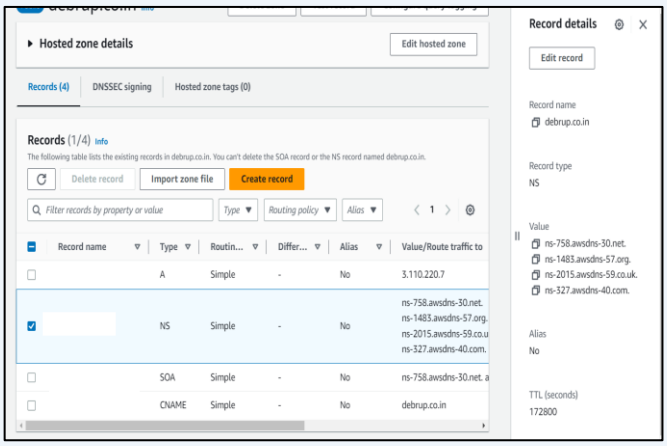


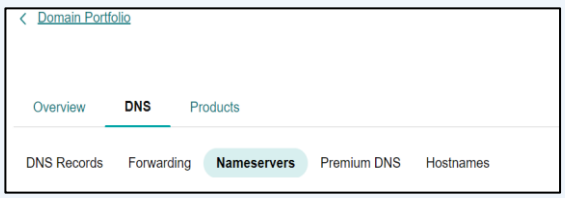
12. Need not to give any name. Keep the record name blank and record type as it is.
13. Under the value, copy and paste your server instance public IPv4 address which you want to route to using your DNS.Then click on create records button



14. **Click on the Create Record button again.But this time give the record name as “www”. Select Record type as CNAME. In the text box under value, write the full domain-name there. (For example: example.com). Click on “create records”.**
15. **Now select the record with type nameserver (NS) .(The values seen on the right-hand side are required for the next steps.)**
16. **Now go to your Domain providers webpage. Go to your purchased Domains settings.**
17. **Click on DNS section. (This may vary from provider to provider)**
18. **Click on the nameservers option.**
19. **Click on the Change nameservers and add here all the values opened in the Route 53 page.Select use my own nameservers option.Add nameservers.Then click on “Save”.**







20. **Search from any browser using your domain name with www.**

21. **Append port no. to access the project.**

Thus,we have successfully run our project using our custom domain-name and URL.