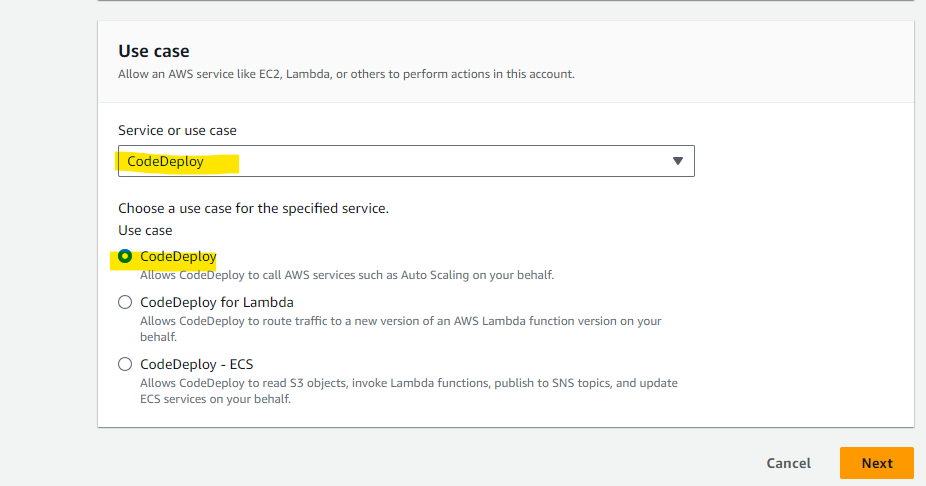
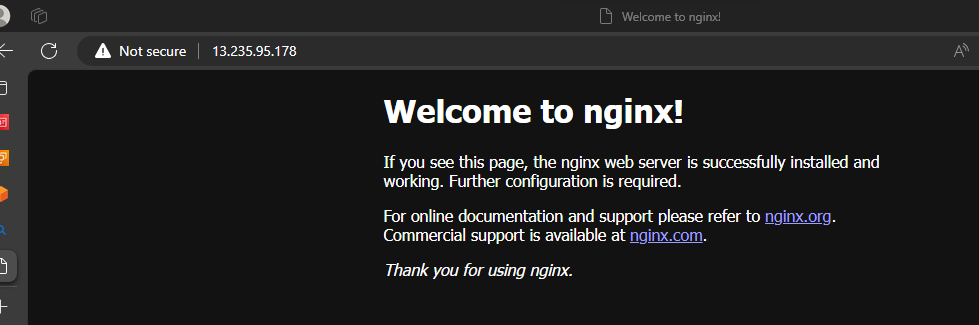
CI/CD Implementation  
  
Requirements

deploying code in local machine and upload it in Git hub repository.  
Once code uploaded in repository it should automatically trigger a pipeline that deploys application on an EC2 instance in AWS.

Steps to be followed.

Create an EC2 instance in public subnet, install nginx on that instance.  
Setup a GitHub repository where codes will be uploaded too.  
Now we have to setup CICD in such a way that it will automatically trigger by GitHub repo once code is uploaded and CICD will update the application page into ec2 instance.  
  
Let’s create role for EC2 instance and CodeDeploy   
  


Now we have to setup a server with nginx installed on ec2 instance.

User data used while launching instance,  
#!/bin/bash  
sudo yum update -y  
sudo yum install nginx -y  
sudo systemctl start nginx  
sudo systemctl enable nginx  
  
Once above steps is done, we can access site through public ip of instance.  
And also simultaneously we have to install and start codedeploy-agent on the instance.  
  
sudo yum install ruby -y  
cd /home/ec2-user  
wget <https://aws-codedeploy-us-east-1.s3.us-east-1.amazonaws.com/latest/install>  
chmod +x ./install  
sudo ./install auto  
sudo systemctl status codedeploy-agent  
sudo systemctl enable codedeploy-agent  
  


First we have to create Application and Deployment Groups (basically helps us to separate dev and prod environment) on CodeDeploy console.

Let’s create and Application onCodeDeploy Console with deployment group

A screenshot of a computer

Description automatically generated

Next Click on deployment group

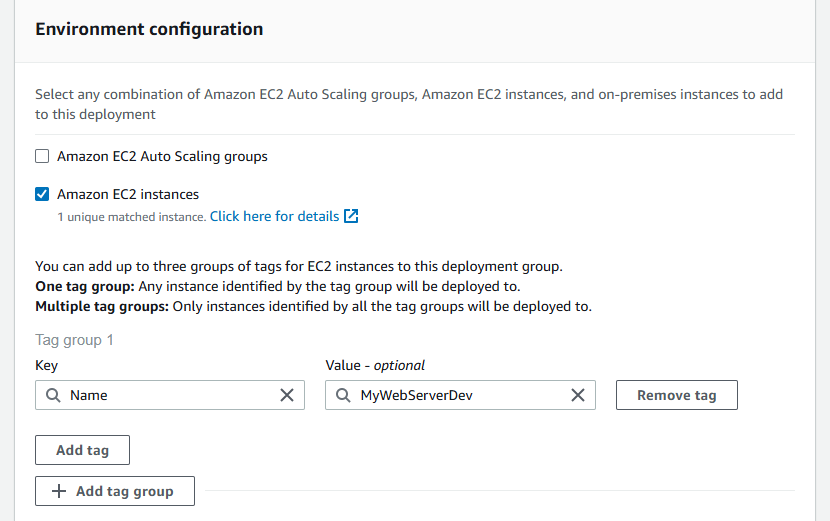
A screenshot of a computer

Description automatically generated

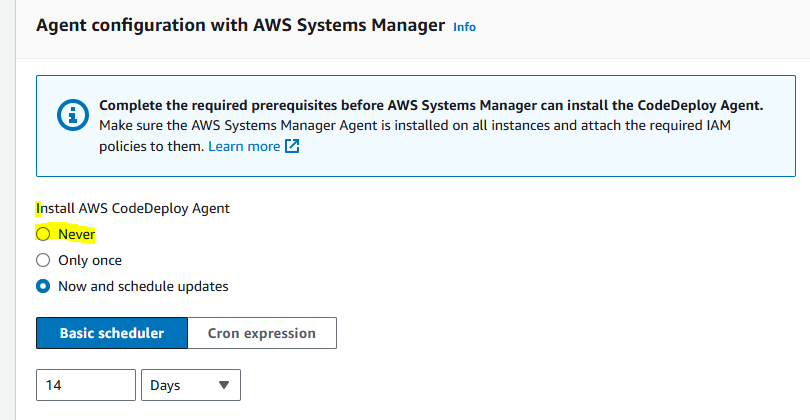
A screenshot of a computer

Description automatically generated

select the instance



As of now we have installed agent manually on instance, we can schedule installation and updates of our choice.



Once deployement group is created, now we can proceed with pipeline creation  
  
A screenshot of a computer

Description automatically generated

Next, let’s create the CodePipeline, with selecting source as Github (selecting respective repository and branch where site is present).

A screenshot of a computer

Description automatically generated

once clicked on next step, unde source page we can selct GitHub version 2 -> click on connect-> we will get a new window -> just give any name for your connection ci-cd and click on connect -> now click on install app (it will help us to autheticate our github account and repository - > next we can click on connect.  
  
A screenshot of a computer

Description automatically generated

GitHub repo is connected as expected

A screenshot of a computer

Description automatically generated

Under depoy page,as we have CodeDeploy for deployment we can select respective application and deployment group created on previous steps.  
  
A screenshot of a computer

Description automatically generated

Now click on create pipeline

A screenshot of a computer

Description automatically generated

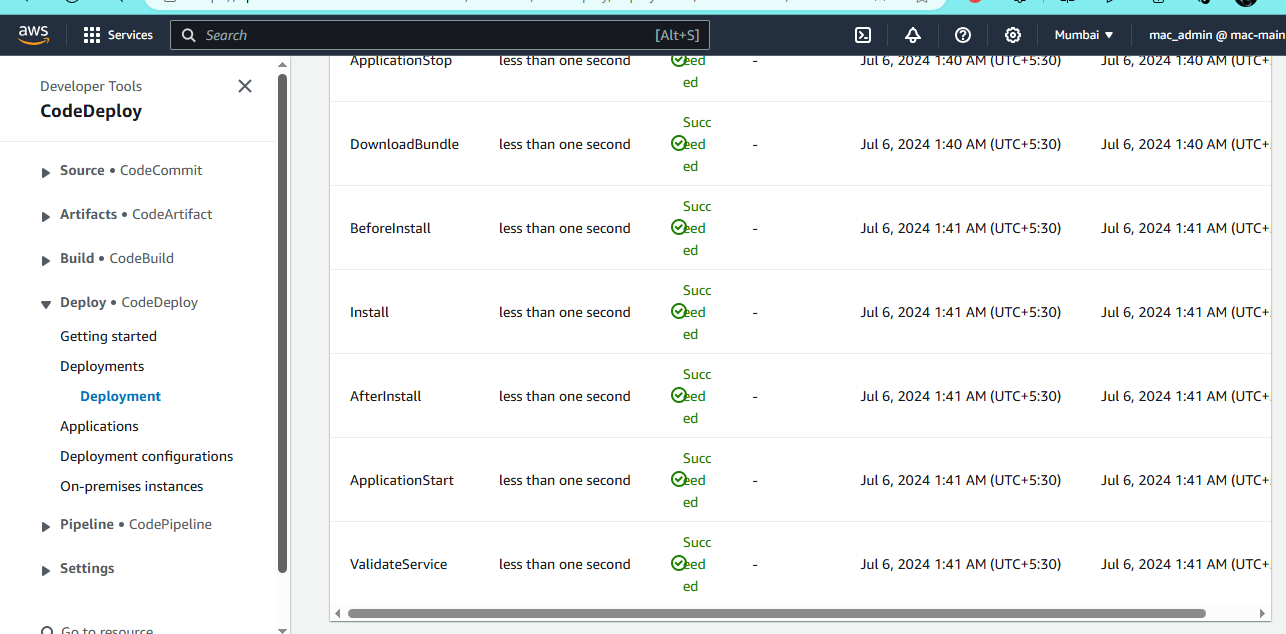
A screenshot of a computer

Description automatically generated

My script failed in while starting the service, as I was using nginx server not httpd.  
   
A screenshot of a computer program

Description automatically generated

Once I changed the script with stop and start of nginx service, the deployment got succeeded.



A screenshot of a computer

Description automatically generated

Now when I tried to access the deployed page through instance public ip it was accessible (was landed on nginx default page).

But this was not the once which I was looking for.  
  
A screenshot of a computer

Description automatically generated

To resolve this issue, I went to the appspec.yml and changed the destination to /usr/share/nginx/html/index (here I have created a new directory with name **index**)

A screenshot of a computer program

Description automatically generated

Finally, we landed to the required page  
  
A screenshot of a computer

Description automatically generated

If we wan't to access the final page with public IP of instace directly instead of using /index.html.  
We need to make changes in /etc/nginx/nginx.conf file

A red circle with white text

Description automatically generated

We can give **/usr/share/nginx/html/index**Now, let's create commit changes into html page and wait for the changes to trigger automatically in our page.

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

Here it's done :)

A screen shot of a computer

Description automatically generated