Project 1:

AWS CodePipeline is a fully managed continuous delivery service that helps you automate your release pipelines for fast and reliable application and infrastructure updates. CodePipeline automates the build, test, and deploy phases of your release process every time there is a code change, based on the release model you define. This enables you to rapidly and reliably deliver features and updates. You can easily integrate AWS CodePipeline with third-party services such as GitHub or with your own custom plugin. With AWS CodePipeline, you only pay for what you use. There are no upfront fees or long-term commitments

This activity guide cover steps for:

1. Create a CodeCommit repository
2. Add sample code to your CodeCommit repository
3. Create an EC2 Linux instance and install the CodeDeploy agent
4. To launch an instance
5. Create an application in CodeDeploy
6. Create your first pipeline in CodePipeline
7. To verify that your pipeline ran successfully
8. Modify code in your CodeCommit repository
9. To verify your pipeline ran successfully
10. Clean up resources

a.Unzip the files from https://docs.aws.amazon.com/codepipeline/latest/userguide/samples/SampleApp_Linux.zip into the local directory (for example, /tmp/MyDemoRepo or c:\temp\MyDemoRepo).

Be sure to place the files directly into your local repository. Do not include a SampleApp_Linux folder. On your local machine for example, your directory and file hierarchy should look like this:

b.Use git commands to upload the code in the local directory to public repository in Codecommit

|-- appspec.yml

|-- index.html

```
|-- LICENSE.txt
└-- scripts
    |-- install_dependencies
    |-- start_server
    └-- stop_server
```

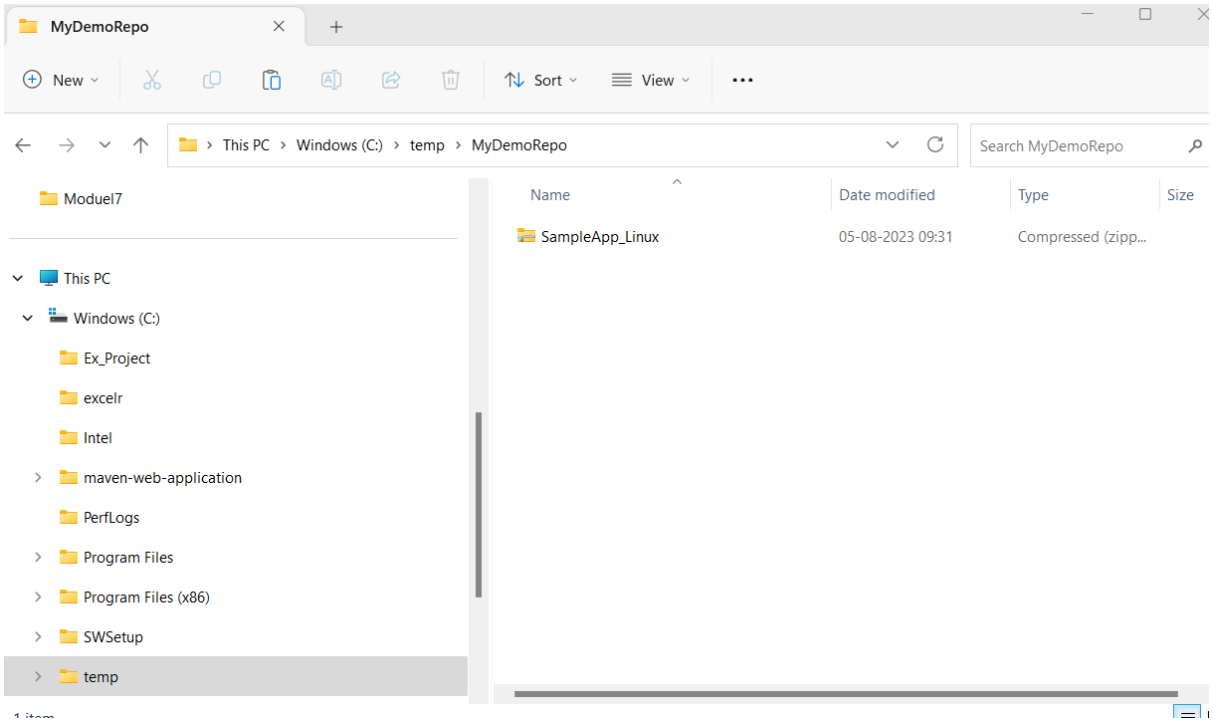
c.Create and configure Code Deploy (Amazon EC2 instance)

d. Create a pipeline using AWS CodePipeline , AWS Code Commit and Code Deploy to deploy index.html to Amazon Linux ec2 instance

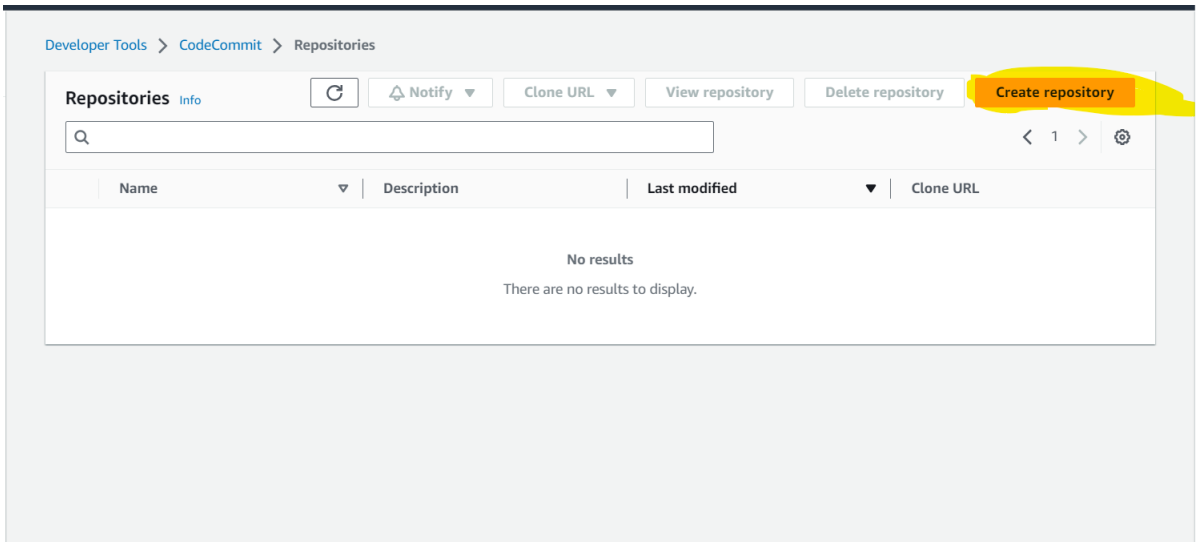
e.Check the o/p , by accessing the public ip of the ec2 instance . Contents of the index.html must be displayed

Download SampleApp_Linux file and unzip it.

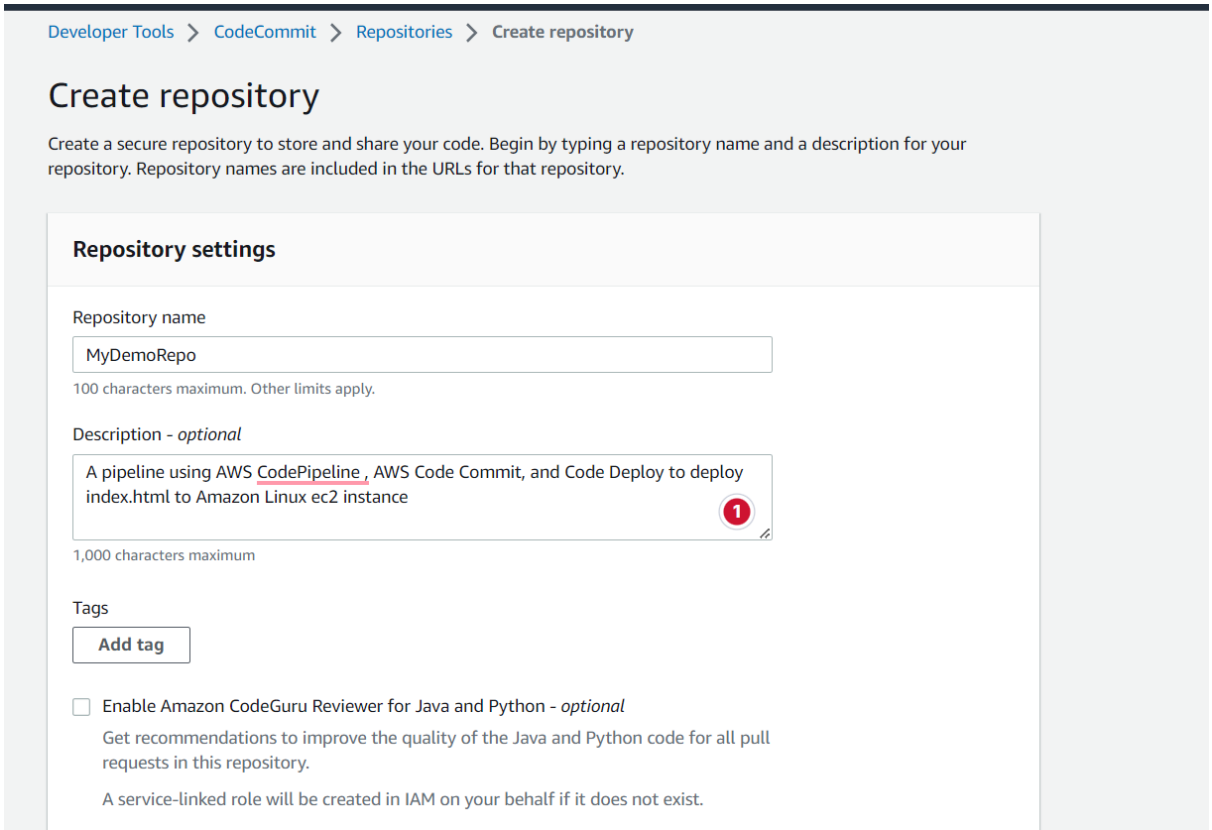
https://docs.aws.amazon.com/codepipeline/latest/userguide/samples/SampleApp_Linux.zip

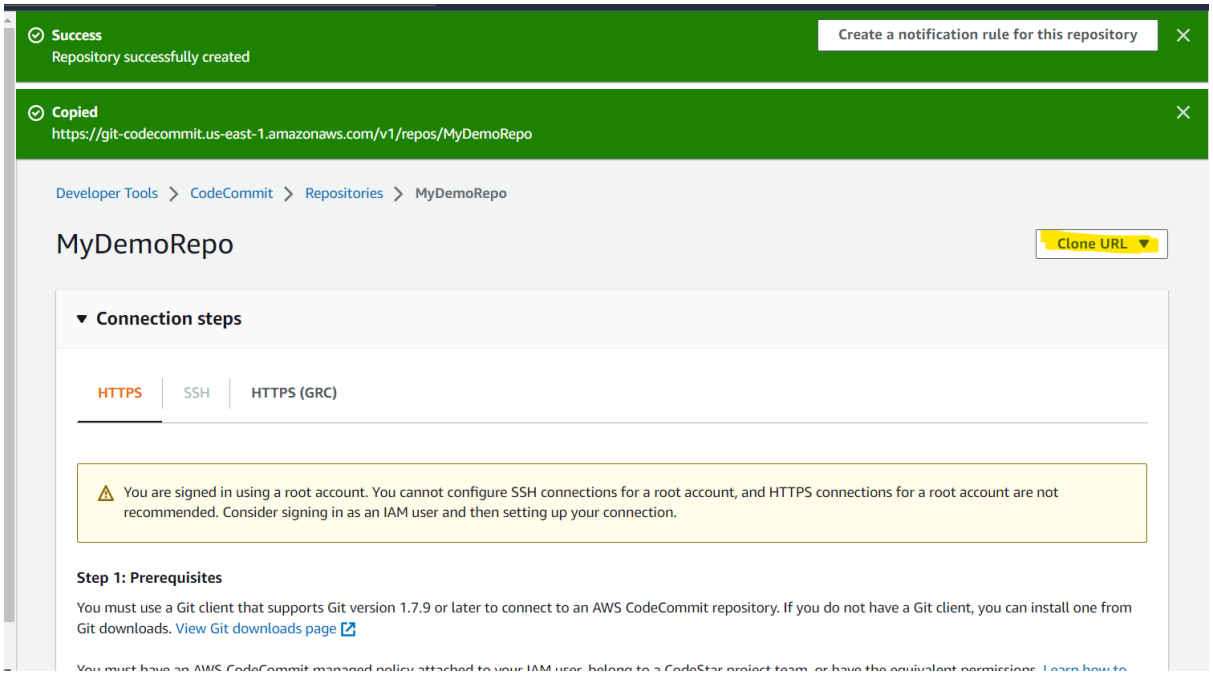


1. Navigate to CodeCommit and create a repository.



2. Enter the name and description and later, create the repository.



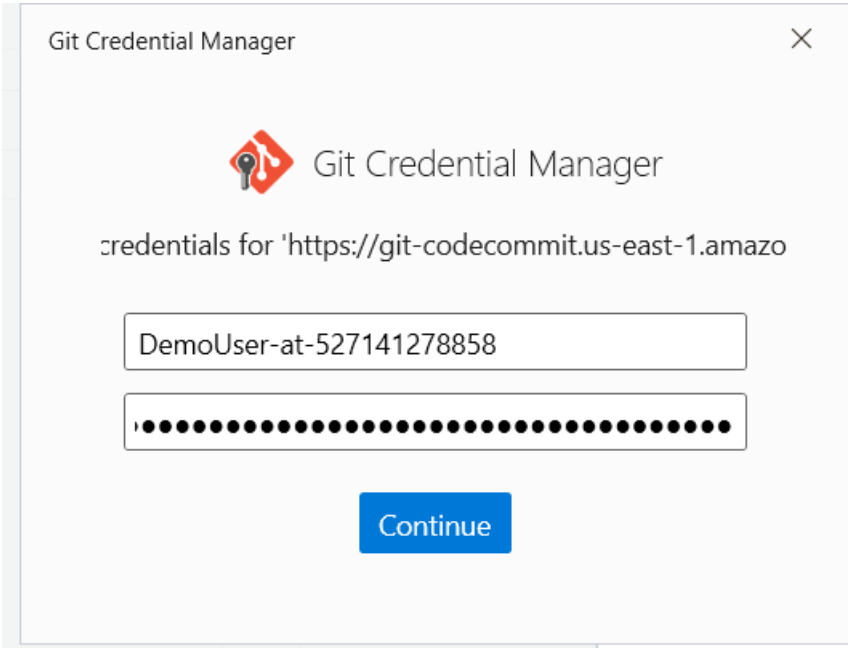


3. Now, you need to push the SampleApp_Linux file to the remote repository you have created. Copy the http link which you will use to clone the repository locally.

- **Git clone "url_link"**

```
C:\Users\Manjunath S>git clone https://git-codecommit.us-east-1.amazonaws.com/v1/repos/MyDemoRepo
Cloning into 'MyDemoRepo'...
warning: You appear to have cloned an empty repository.
```

- It will ask for the credentials, you need to download the credentials from IAM->Git credentials.



4. Initialize and check the git status. You can the files but it is not yet committed.

```
C:\Users\Manjunath S>cd C:\temp\MyDemoRepo\SampleApp_Linux

C:\temp\MyDemoRepo\SampleApp_Linux>git init
Initialized empty Git repository in C:/temp/MyDemoRepo/SampleApp_Linux/.git/

C:\temp\MyDemoRepo\SampleApp_Linux>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        LICENSE.txt
        appspec.yml
        index.html
        scripts/

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

C:\temp\MyDemoRepo\SampleApp_Linux>
```

5. Add and commit the changes later, check the status.

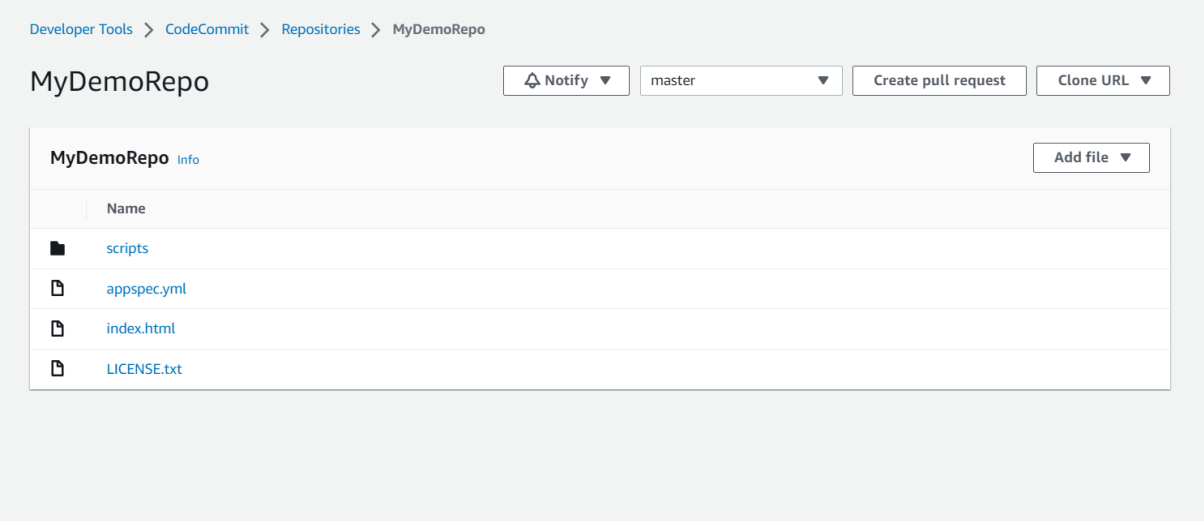
```
C:\temp\MyDemoRepo\SampleApp_Linux>git add -A
warning: in the working copy of 'LICENSE.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'appspec.yml', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'scripts/install_dependencies', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'scripts/start_server', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'scripts/stop_server', LF will be replaced by CRLF the next time Git touches it

C:\temp\MyDemoRepo\SampleApp_Linux>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   LICENSE.txt
        new file:   appspec.yml
        new file:   index.html
        new file:   scripts/install_dependencies
        new file:   scripts/start_server
        new file:   scripts/stop_server
```

```
C:\temp\MyDemoRepo\SampleApp_Linux>git commit -m "uploading the contents"
[master (root-commit) 9699651] uploading the contents
 6 files changed, 266 insertions(+)
 create mode 100644 LICENSE.txt
 create mode 100644 appspec.yml
 create mode 100644 index.html
 create mode 100644 scripts/install_dependencies
 create mode 100644 scripts/start_server
 create mode 100644 scripts/stop_server

C:\temp\MyDemoRepo\SampleApp_Linux>git push
```

6. And next, push the changes into the remote repository.

```
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 12 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 5.01 KiB | 1.00 MiB/s, done.
Total 9 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Validating objects: 100%
To https://git-codecommit.us-east-1.amazonaws.com/v1/repos/MyDemoRepo
 * [new branch]      master -> master
branch 'master' set up to track 'MyDemoRepo/master'.

C:\temp\MyDemoRepo\SampleApp_Linux>
```

7. Verify whether you can see the files in the remote repository.



8. The next step is to launch EC2 Instance. Select the AMI image and required credentials.

- And use the below commands to run the agent in your instance.

sudo apt-get update

sudo apt-get install -y awscli

sudo apt-get install -y ruby

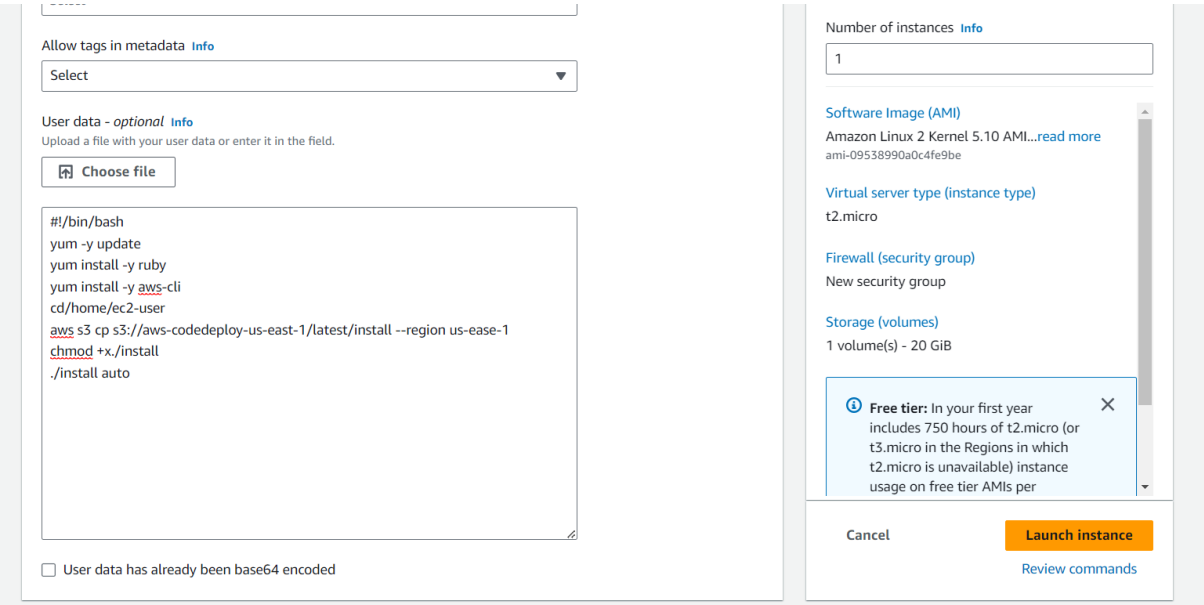
wget https://aws-codedeploy-us-east-1.s3.us-east-1.amazonaws.com/latest/install

chmod +x ./install

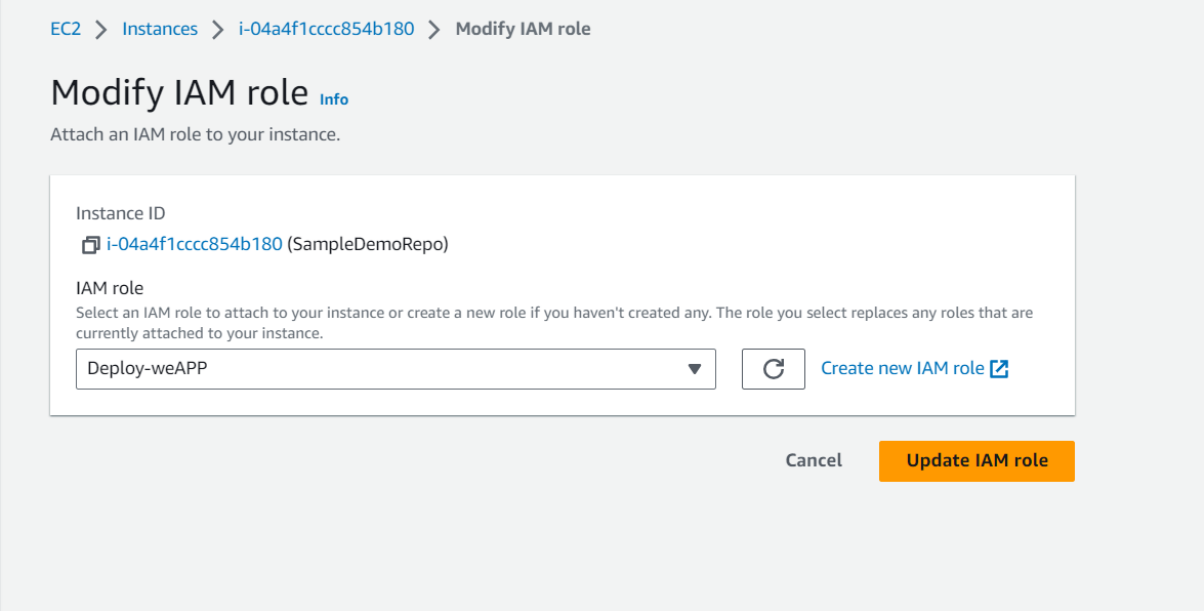
sudo ./install auto

sudo service codedeploy-agent start

sudo service codedeploy-agent status

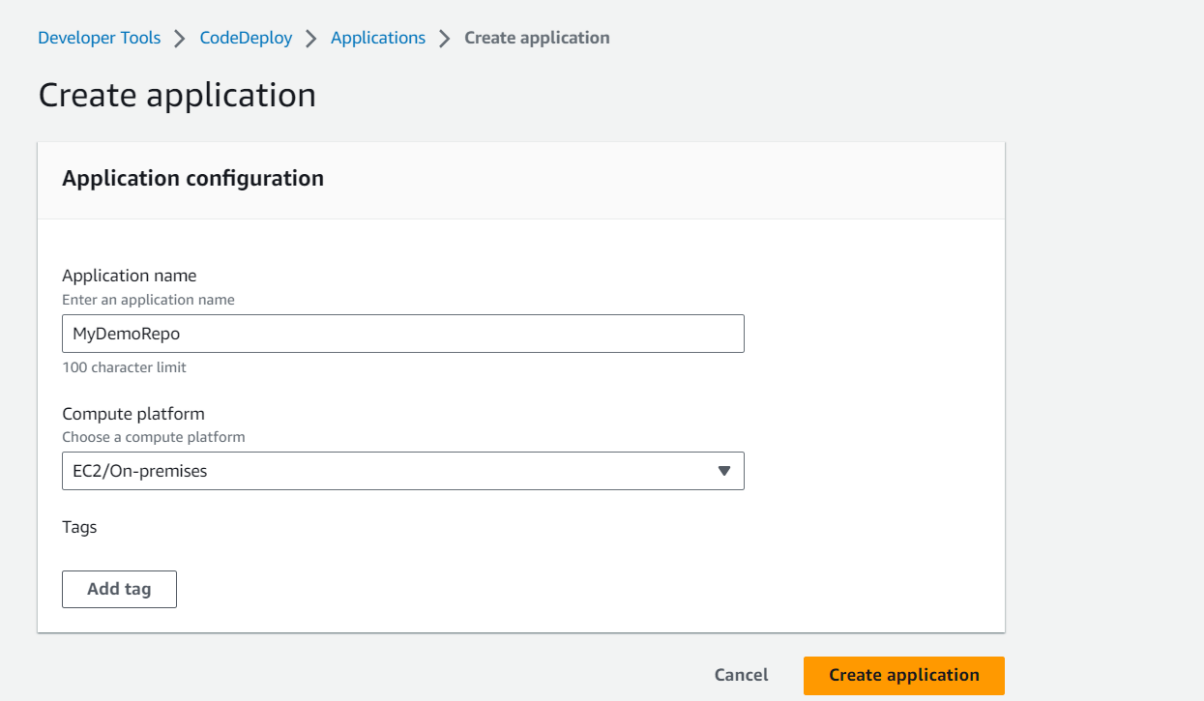


9. And in your instance, modify the IAM role and attach the role.

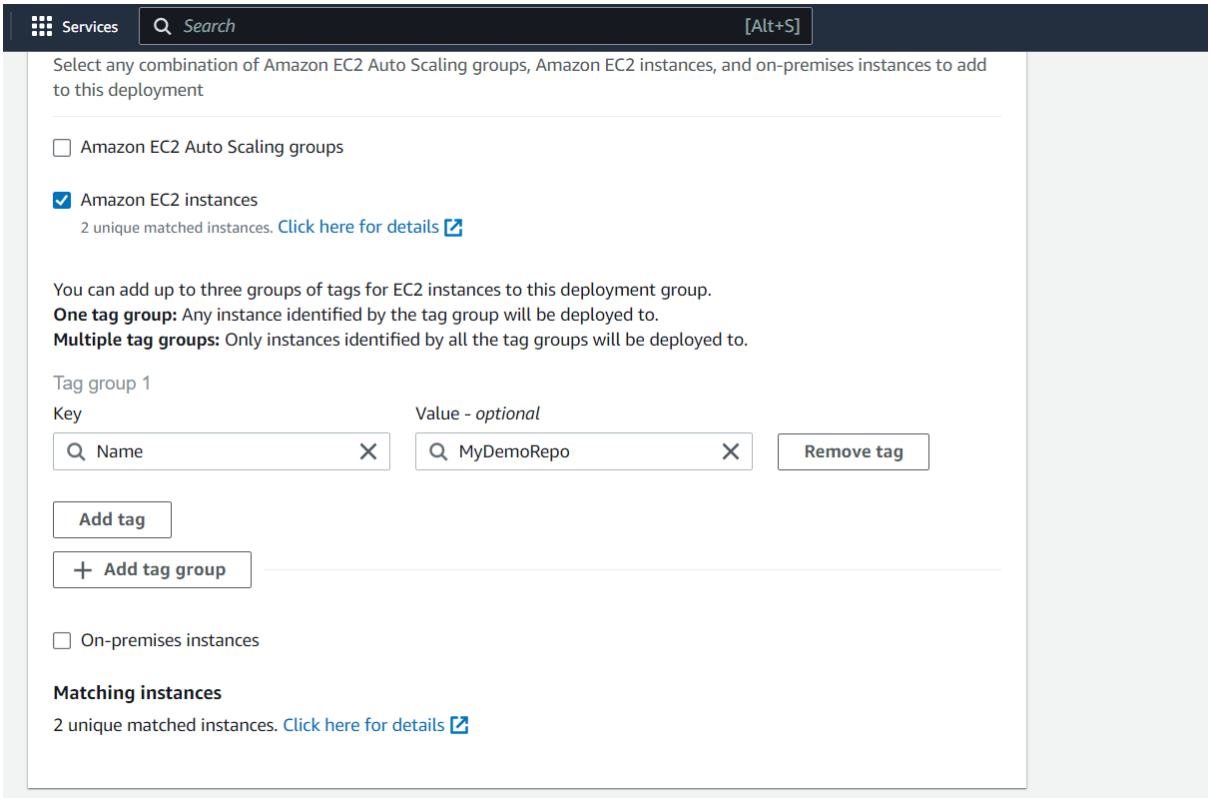


10. The next step is to create an application, Navigate Codedeploy > Applications > Create applications.

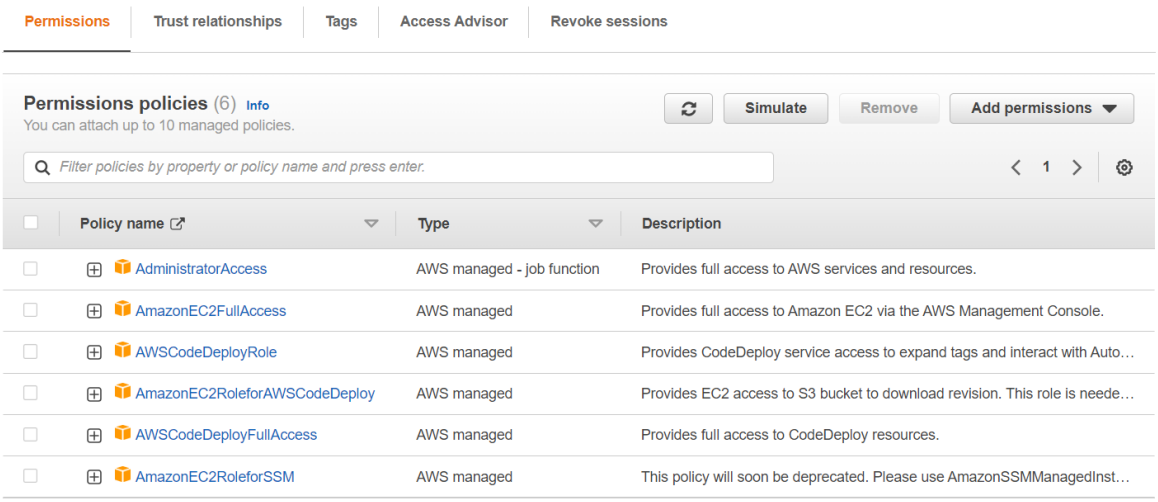
Here, you need to enter the name of the application and select EC2/On-premises.



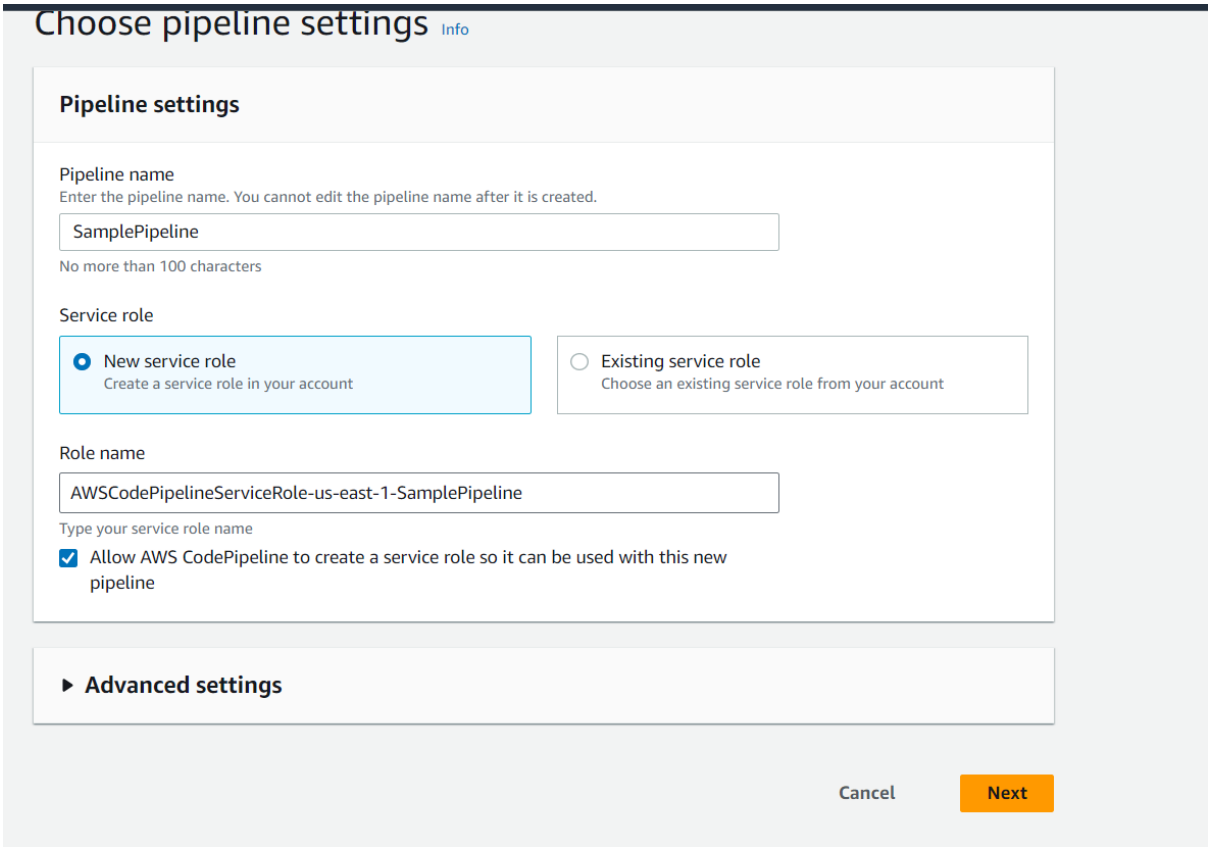
11. After creating the application, create code-deploy group, you need to enter the service role and select Amazon EC2 instance. Select the key, the key will be your instance name. Next, unselect the load balance.



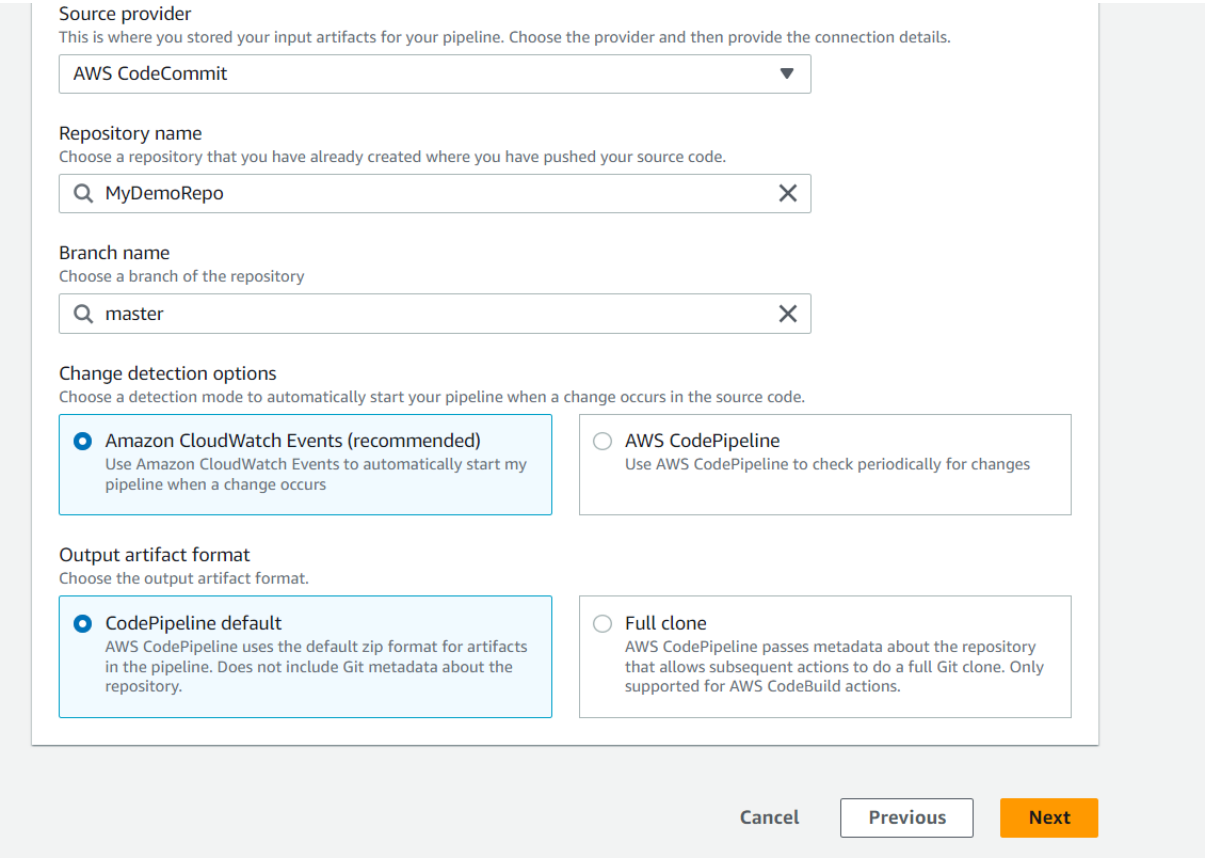
Below are the sample permission for the role.



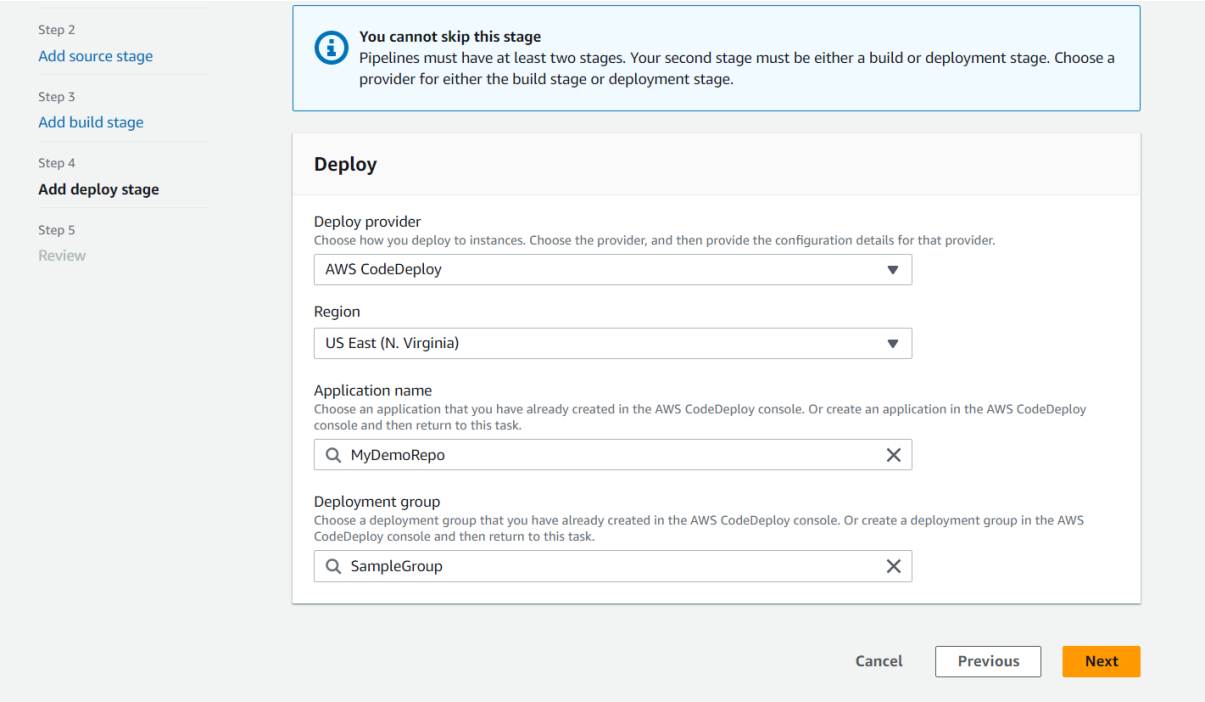
12. Now, navigate to the pipeline in codedeploy and create a new pipeline. Enter the name of the pipeline and select next.



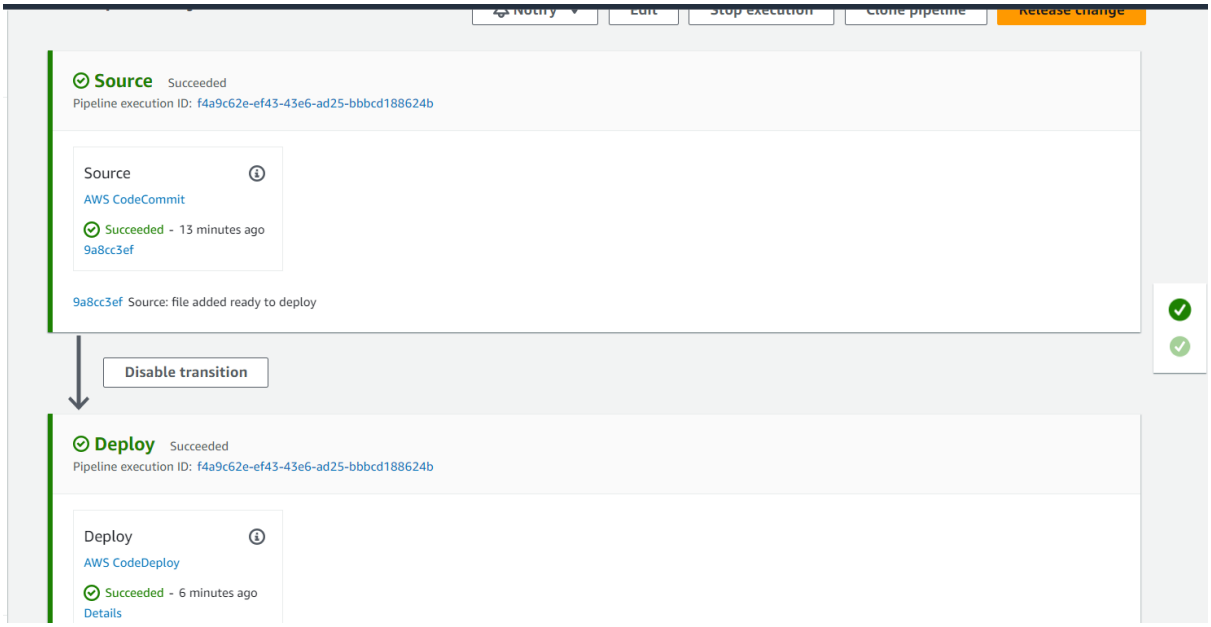
13. The source provider will be AWS Codecommit, select the repository and the branch that you have created.

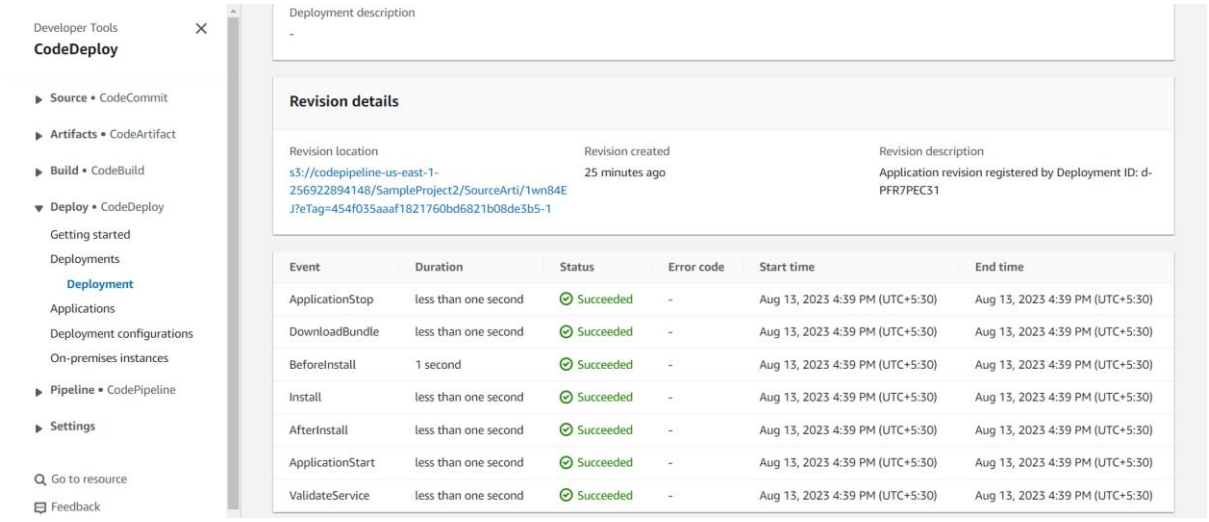


14. You can skip the build-stage. In the Deploy stage, the deploy provider will be AWS codedeploy. Select the region. Select the application and deployment group that you have created.

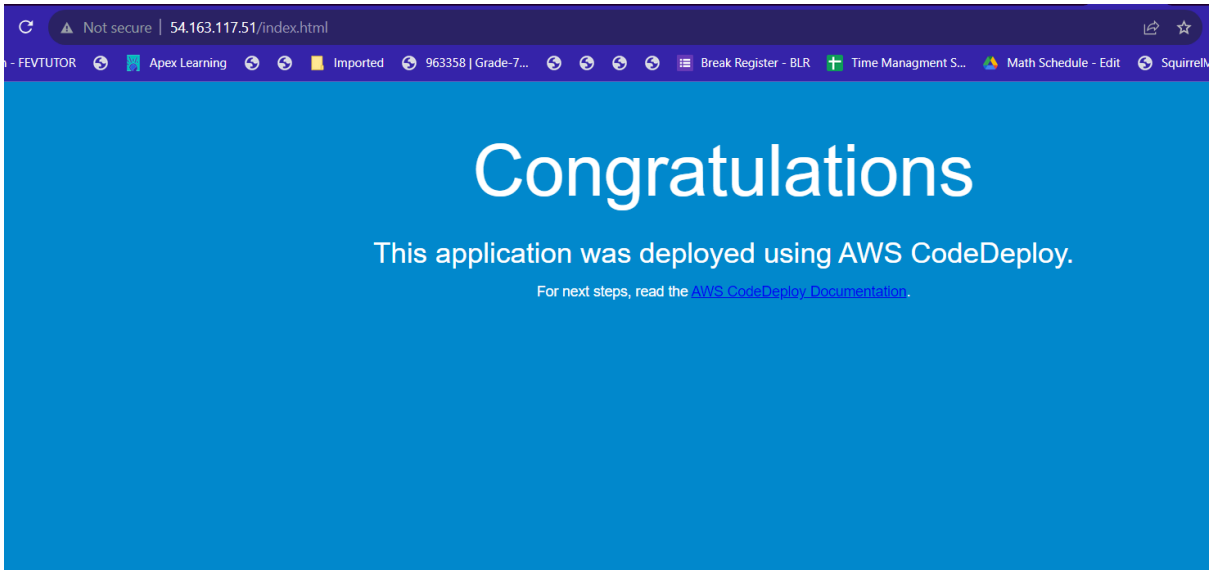


Click on next, review the pipeline, and create the pipeline. Monitor the results of the pipeline.



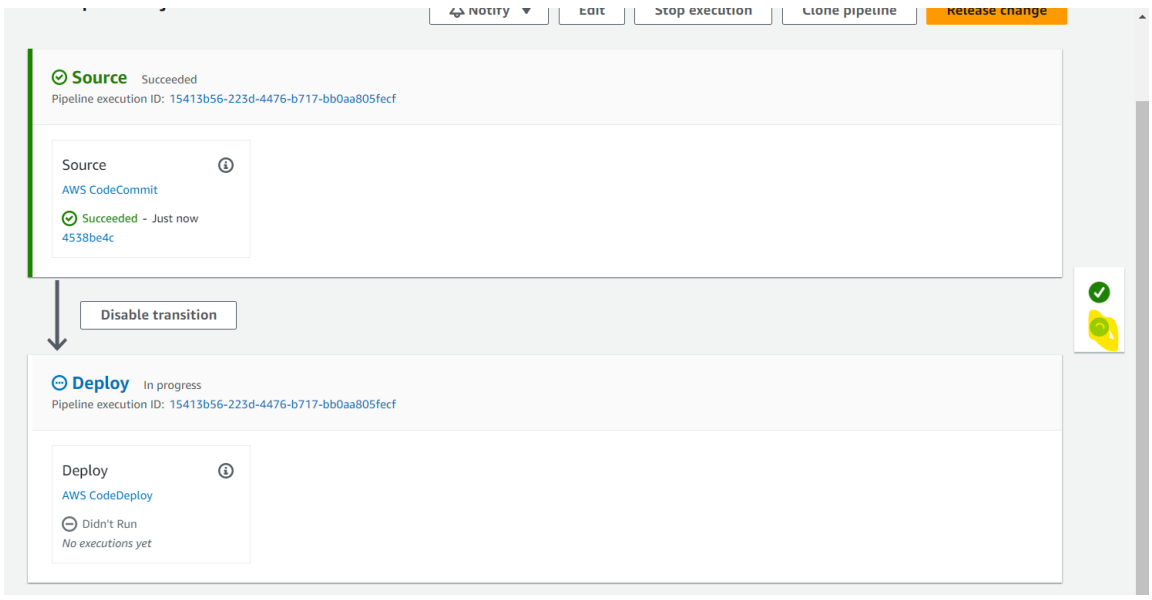


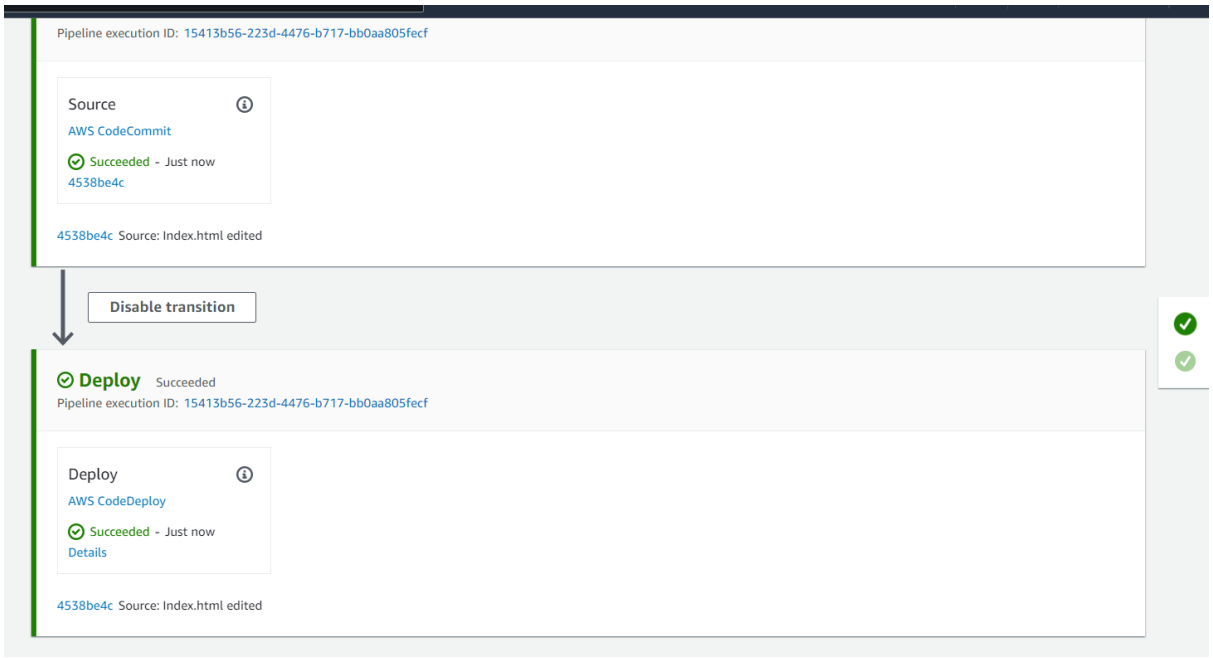
To see the output, copy the public IP address of your instance and browse it.



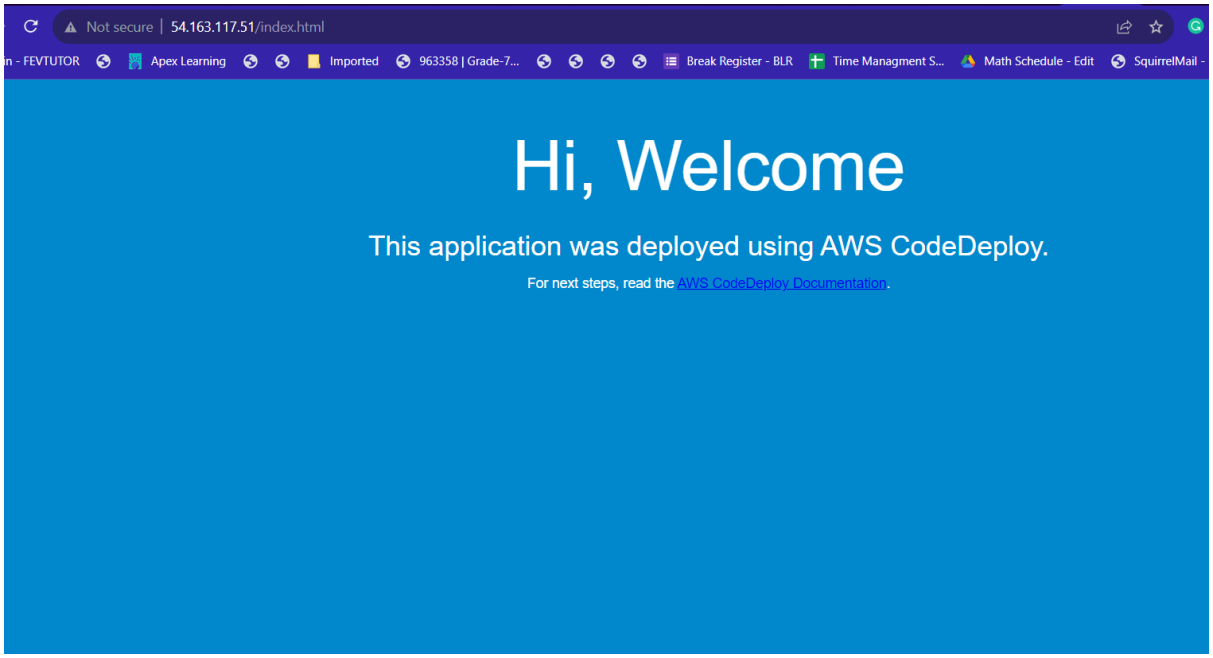
15. Modify code in your CodeCommit repository.

To verify your pipeline ran successfully.





And check the result on your browser once again.



Later, clean-up the resources if it is not needed anymore.