SHAHEROZ ALAM KHAN

Aim

To automate the process of building, testing, and deploying applications using a CI/CD pipeline on AWS, leveraging Visual Studio Code (VS Code) as the primary development environment to ensure efficient and reliable delivery of software changes.

Objectives

1. **Integrate VS Code with AWS Services**: Use Visual Studio Code to manage source code and directly integrate with AWS services like CodeCommit
2. **Automate Build and Deployment Processes**: Set up a CI/CD pipeline that automatically builds and deploys code changes from within the VS Code environment to AWS.
3. **Continuous Integration and Deployment**: Implement a pipeline to continuously test and deploy changes, ensuring the application is always in a deployable state.
4. **Scalability and Flexibility**: Utilize AWS services that can scale with the application's needs and integrate with various development tools.

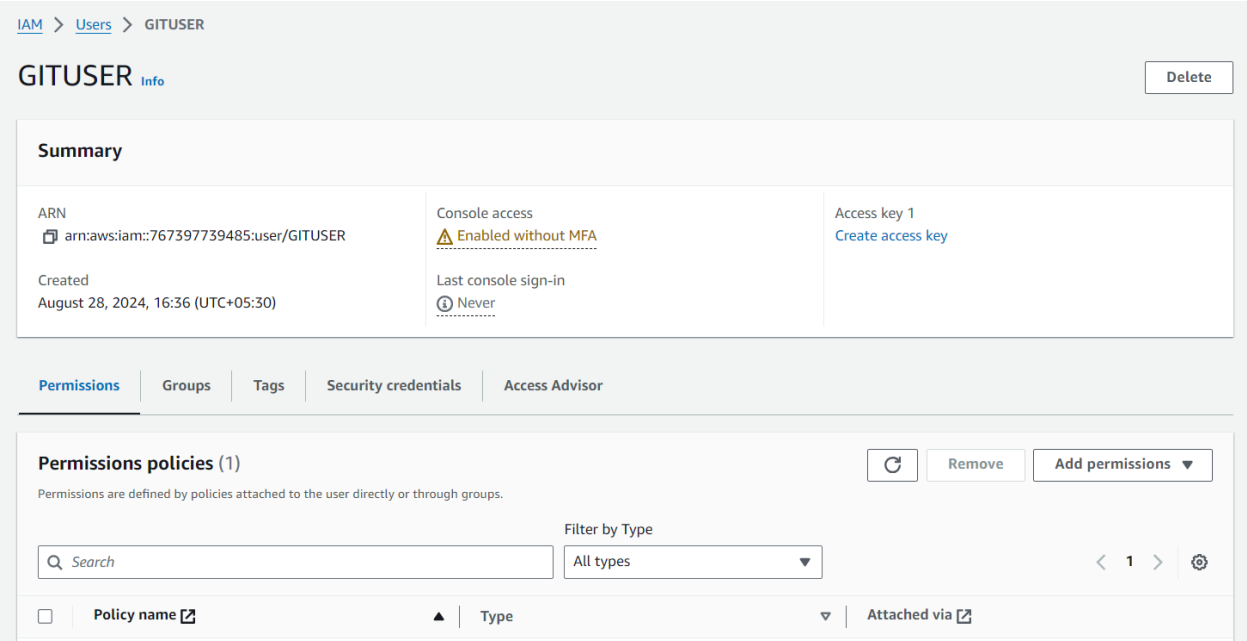
Prerequisites

1. **AWS Account**: A valid AWS account with appropriate permissions to create and manage AWS services.
2. **Visual Studio Code**: Installed VS Code with the AWS Toolkit extension for VS Code.
3. **AWS CLI**: Installed and configured AWS Command Line Interface (CLI) for managing AWS services from the terminal.
4. **AWS IAM Role**: An IAM user with programmatic access configured in AWS CLI, providing the necessary permissions for AWS services used in the pipeline.
5. **Git Installed**: Git must be installed and configured to manage version control in the VS Code environment.
6. **Code Repository**: A source code repository set up in AWS CodeCommit
7. **Build Specification File**: A `buildspec.yml` file or equivalent, which defines the build and test commands for AWS CodeBuild.

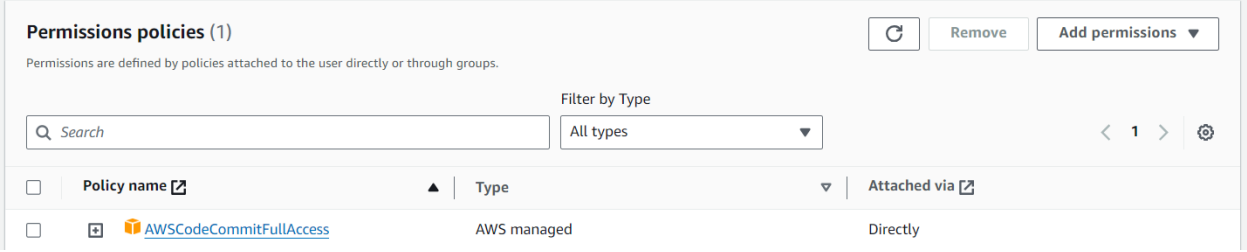
8. **Deployment Configuration**: Prepared deployment configurations for AWS CodeDeploy

Create and Configure IAM User:

- Set up an IAM user with appropriate permissions for AWS CodeCommit.

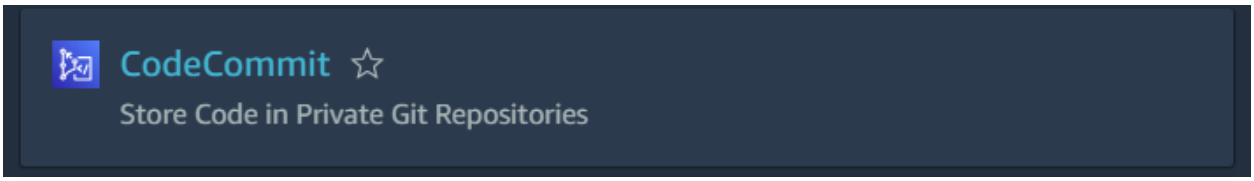


Attach policy: AWSCodeCommitFullAccess



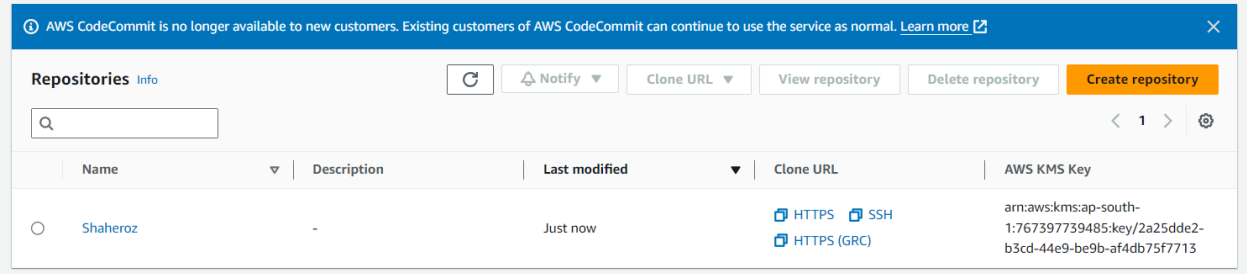
SHAHEROZ ALAM KHAN

Setup AWS CodeCommit Repository:



Create CodeCommit Repository:

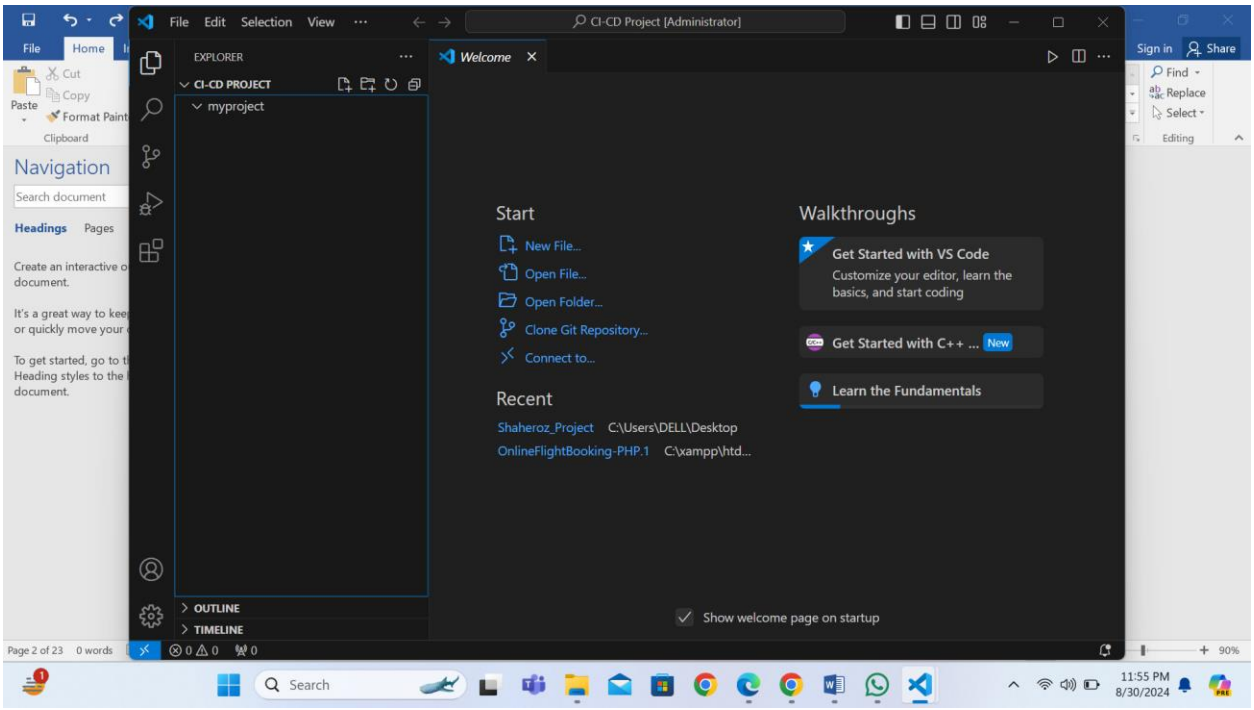
- In the AWS Management Console, navigate to **CodeCommit**.
- Click **Create repository**.
- Enter a repository name and optional description.
- Click **Create**.



Make a folder in your Local Machine

SHAHEROZ ALAM KHAN

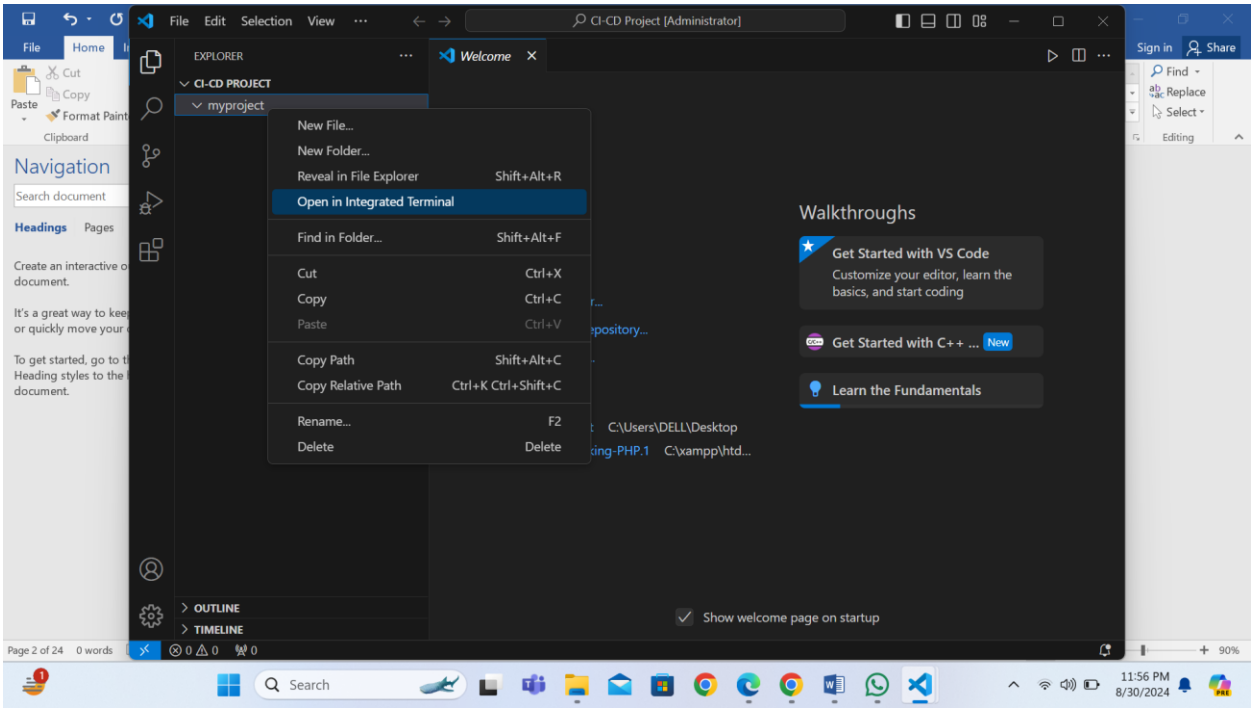
Open the folder in VS Code



Create another folder inside that folder i.e your project folder

Open the folder in interagted Terminal

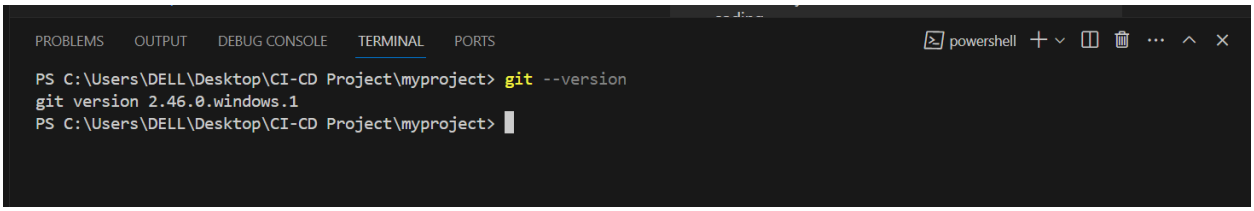
SHAHEROZ ALAM KHAN



Check git version

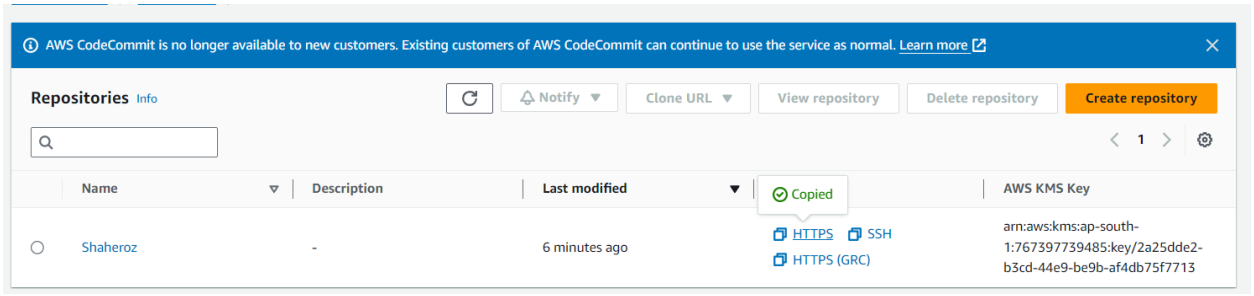
Git –version

If not downloaded make sure to download from official website



Clone Repository to Local Machine:

- In the CodeCommit console, open the repository you created.
- Click **Clone URL** and copy the HTTPS URL.



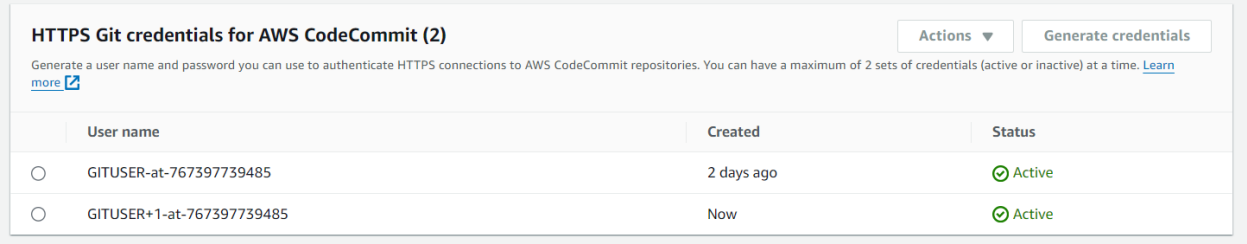
SHAHEROZ ALAM KHAN

Open Visual Studio Code terminal and run:

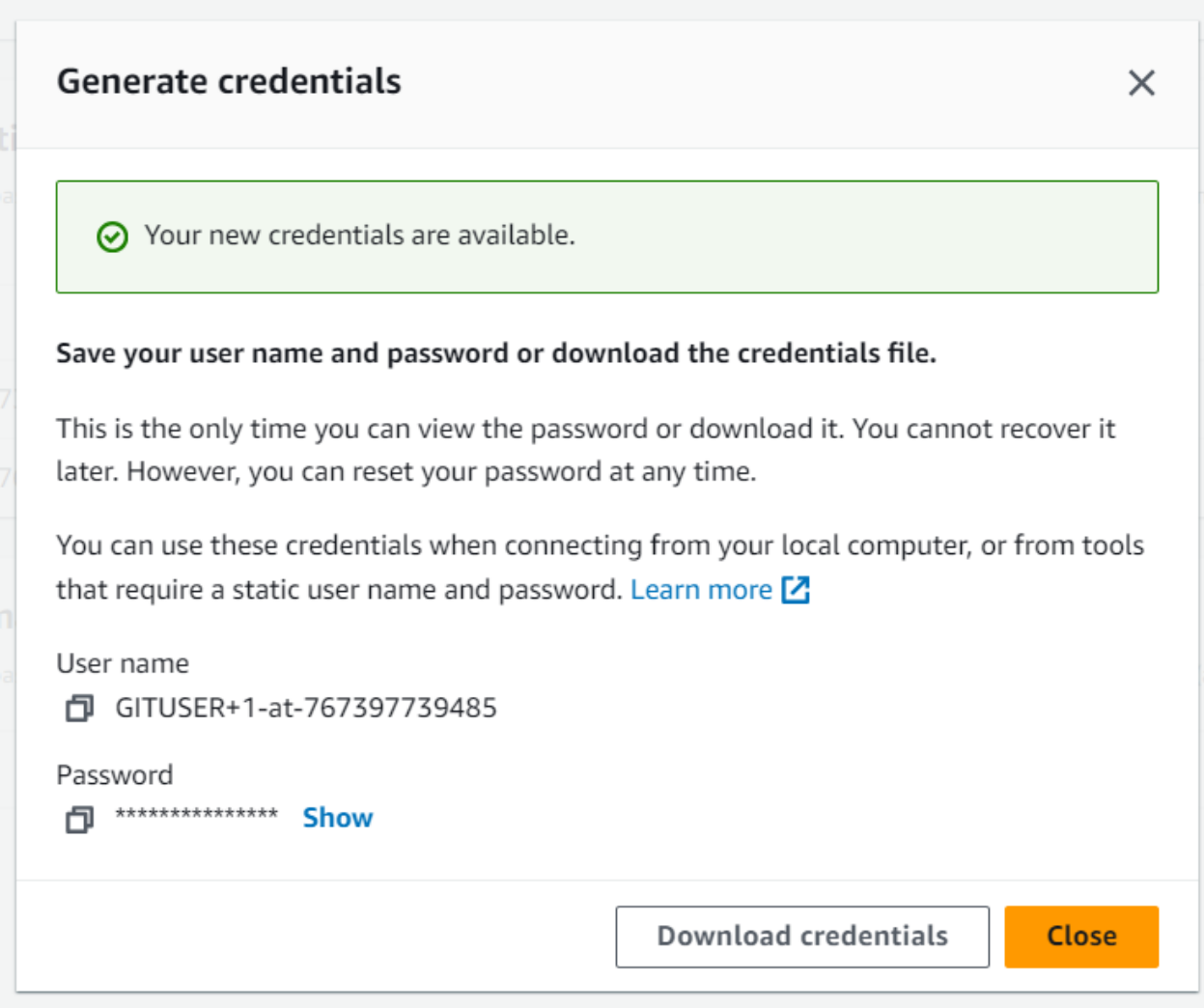
git clone <CodeCommit-Repository-URL>

```
PS C:\Users\DELL\Desktop\CI-CD Project\myproject> git clone https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/Shaheroz
Cloning into 'Shaheroz'...
warning: You appear to have cloned an empty repository.
PS C:\Users\DELL\Desktop\CI-CD Project\myproject> cd shaheroz
PS C:\Users\DELL\Desktop\CI-CD Project\myproject\shaheroz>
```

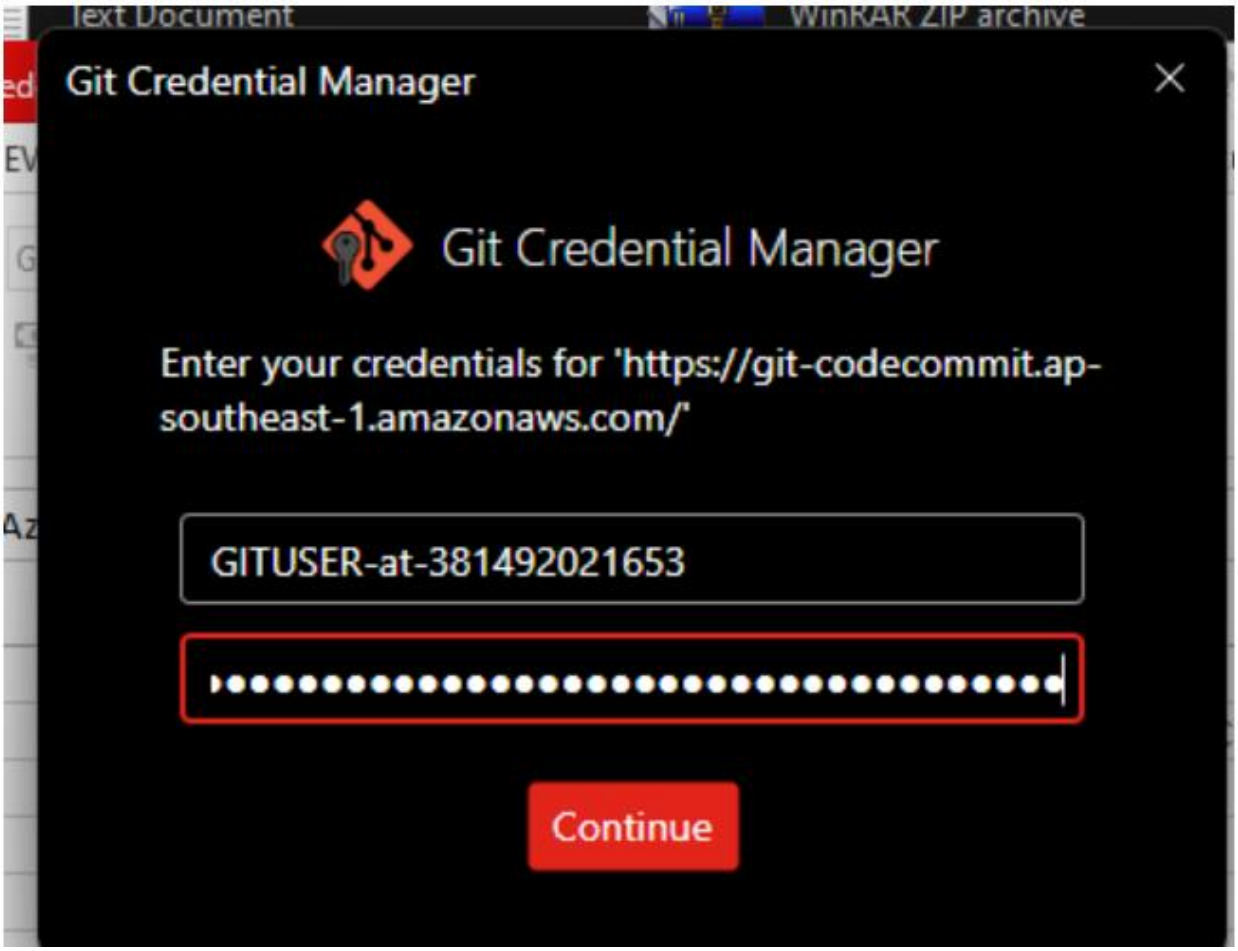
Enter your IAM user credentials when prompted.



SHAHEROZ ALAM KHAN



SHAHEROZ ALAM KHAN



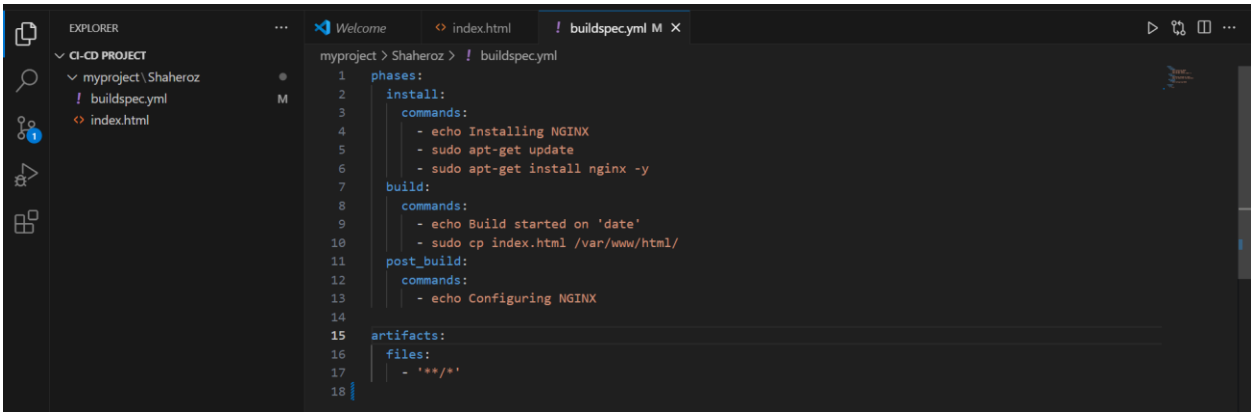
SHAHEROZ ALAM KHAN

Create a file in the project folder name

buildspec.yml

version: 0.2

```
phases:
  install:
    commands:
      - echo Installing NGINX
      - sudo apt-get update
      - sudo apt-get install nginx -y
  build:
    commands:
      - echo Build started on 'date'
      - sudo cp index.html /var/www/html/
  post_build:
    commands:
      - echo Configuring NGINX

artifacts:
  files:
    - '**/*'
```



NOW push the file in AWS codecommit

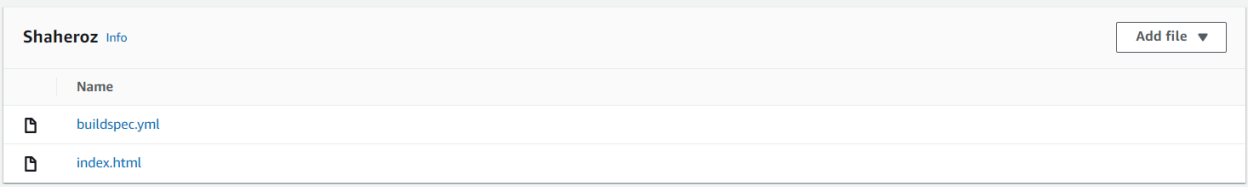
git add .

git commit –m “1st commit”

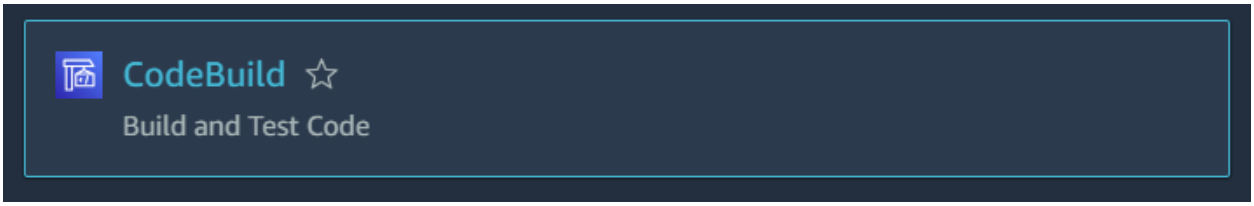
git push origin master

```
PS C:\Users\DELL\Desktop\CI-CD Project\myproject\shaheroz> git add .
PS C:\Users\DELL\Desktop\CI-CD Project\myproject\shaheroz> git commit -m "1st commit"
[master (root-commit) 53af46e] 1st commit
 2 files changed, 18 insertions(+)
 create mode 100644 buildspec.yml
 create mode 100644 index.html
PS C:\Users\DELL\Desktop\CI-CD Project\myproject\shaheroz> git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 455 bytes | 151.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Validating objects: 100%
To https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/Shaheroz
 * [new branch]      master -> master
PS C:\Users\DELL\Desktop\CI-CD Project\myproject\shaheroz>
```

Now the file is in your aws codecommit repositry

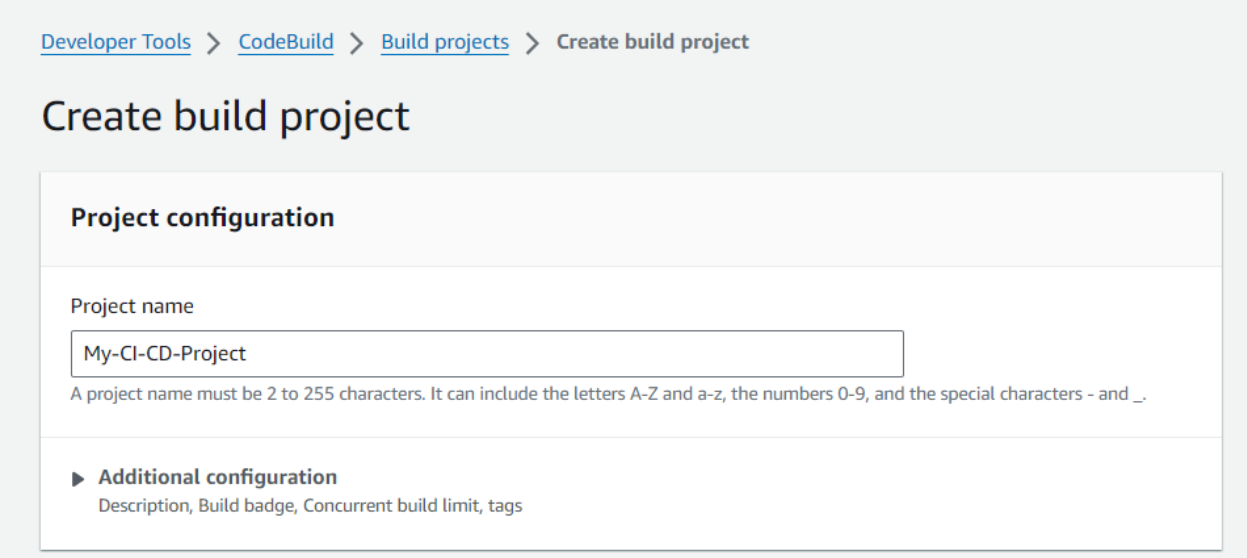


Configure AWS CodeBuild

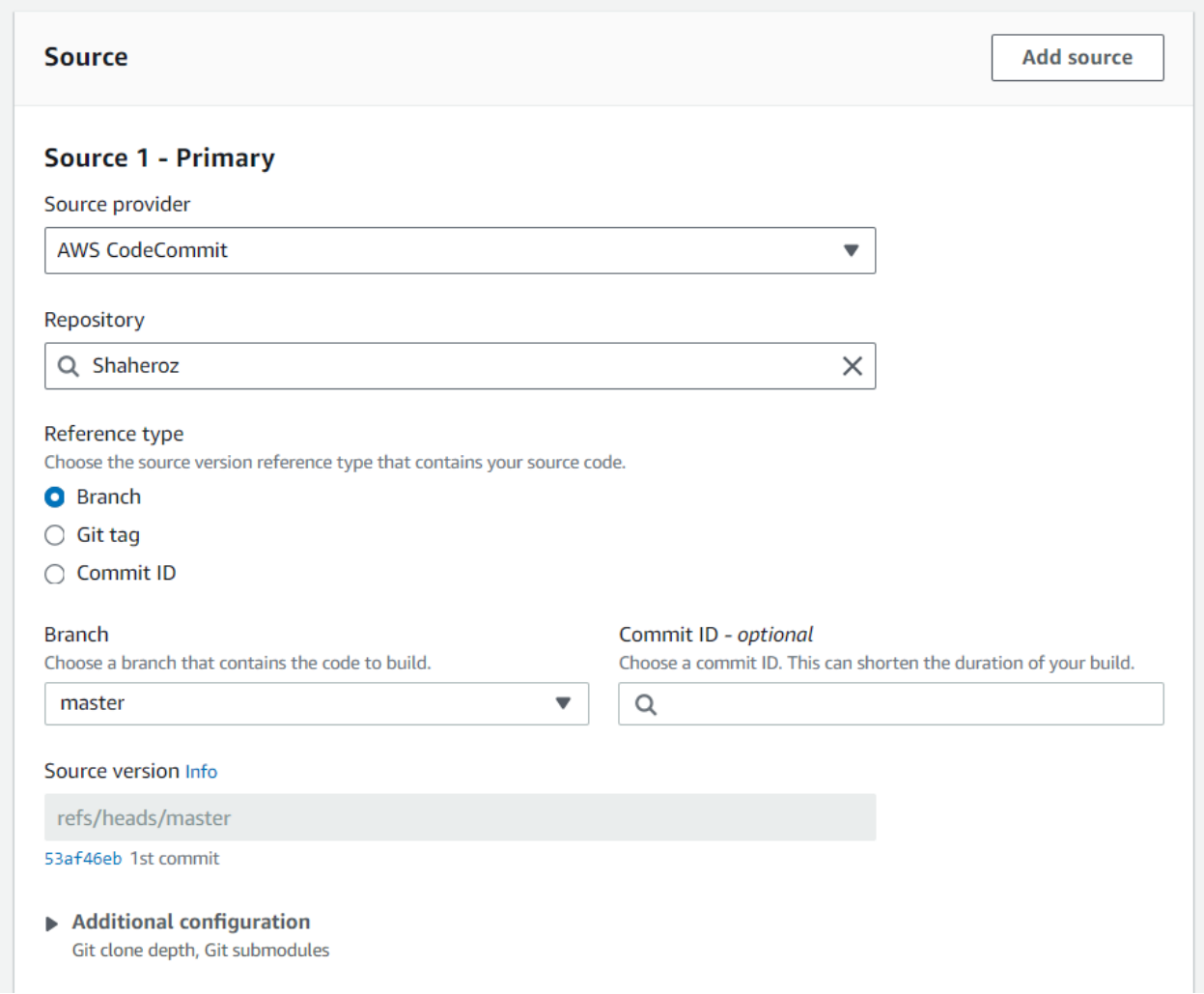


- In the AWS Management Console, go to **CodeBuild**.
- Click **Create build project**.

Enter a project name and description.

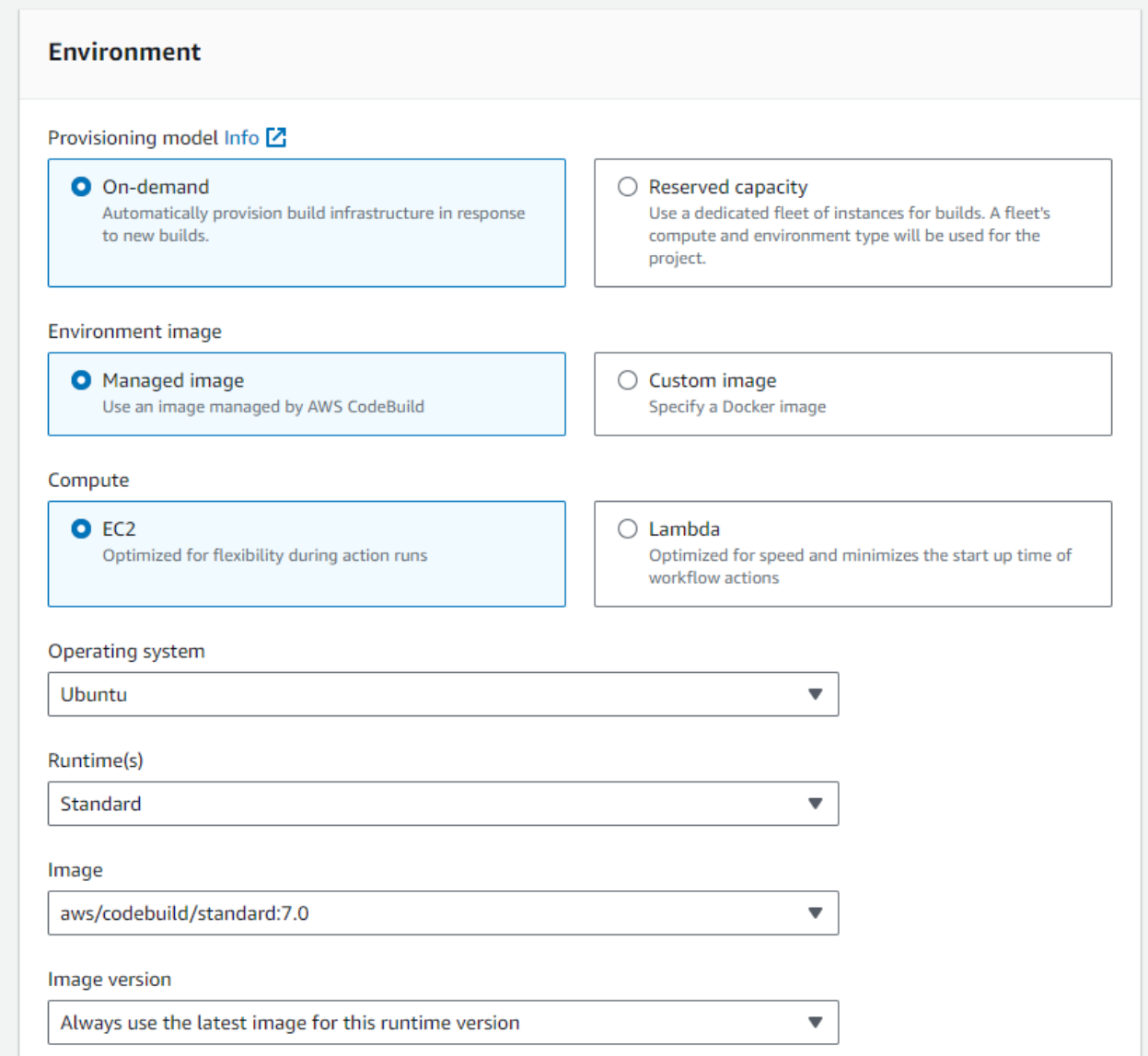


Choose **CodeCommit** as the source provider and select your repository and branch.

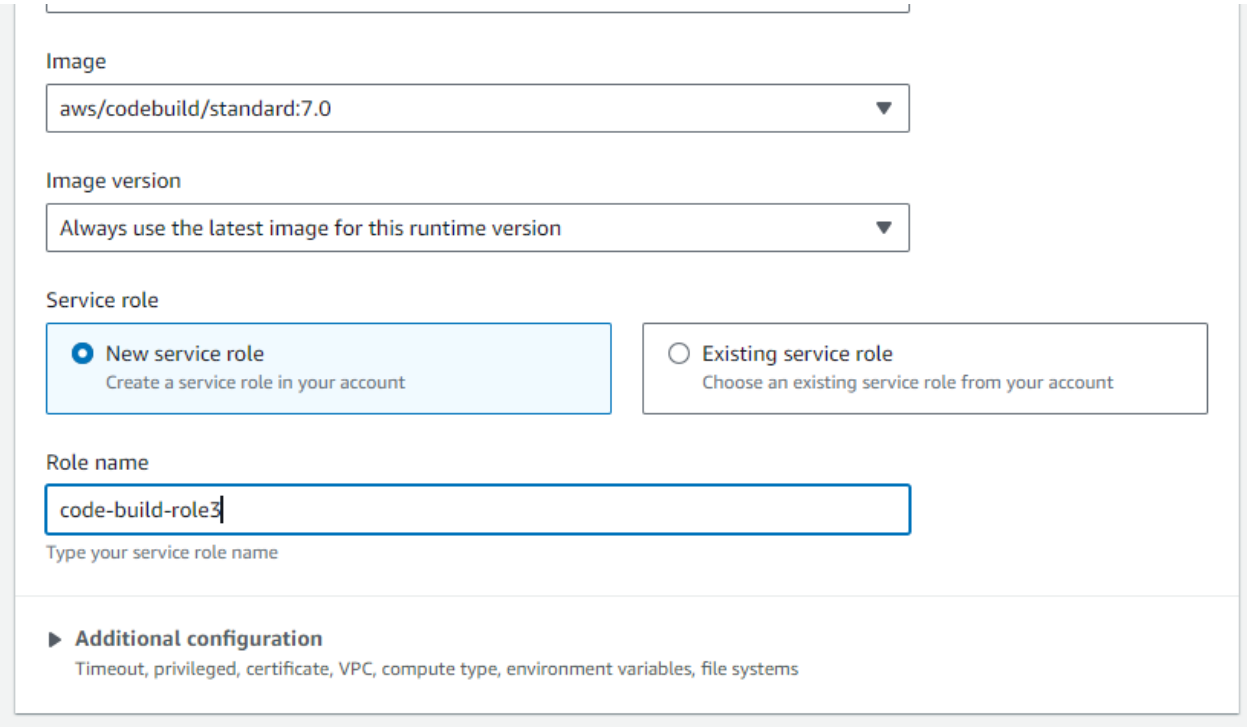


In environment go with by default

And choose operating system “**Ubuntu”**



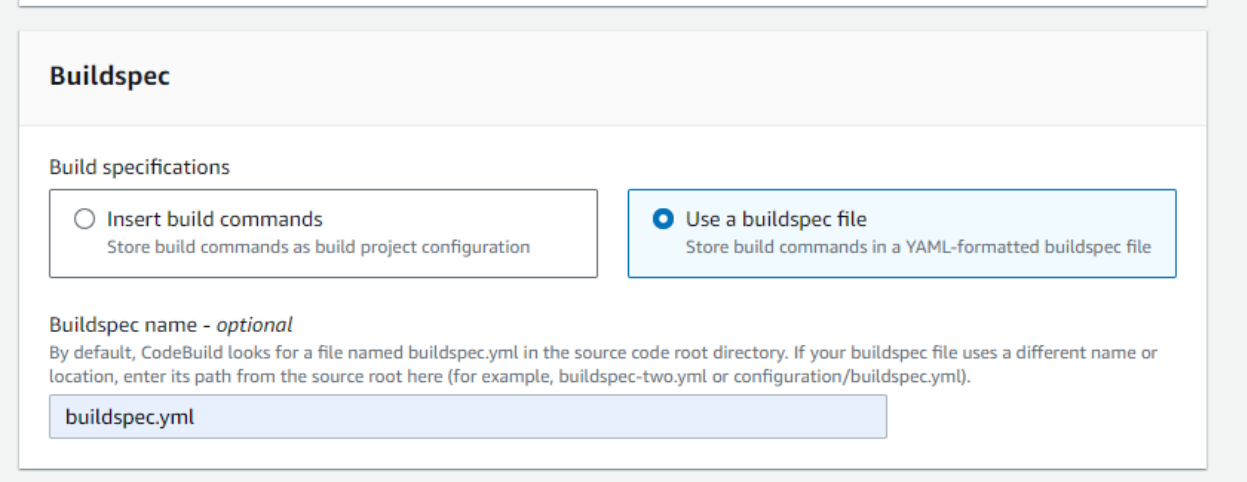
Only enter the role name aws create it by self with necessary policy



Buildspec

Choose : Use a buildspec file

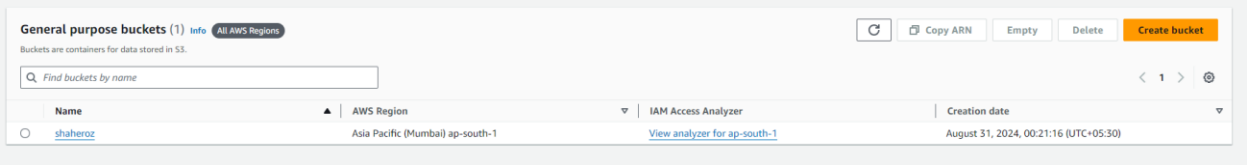
Enter the file name



SHAHEROZ ALAM KHAN

Create simple s3 bucket (eithout versioning and all)

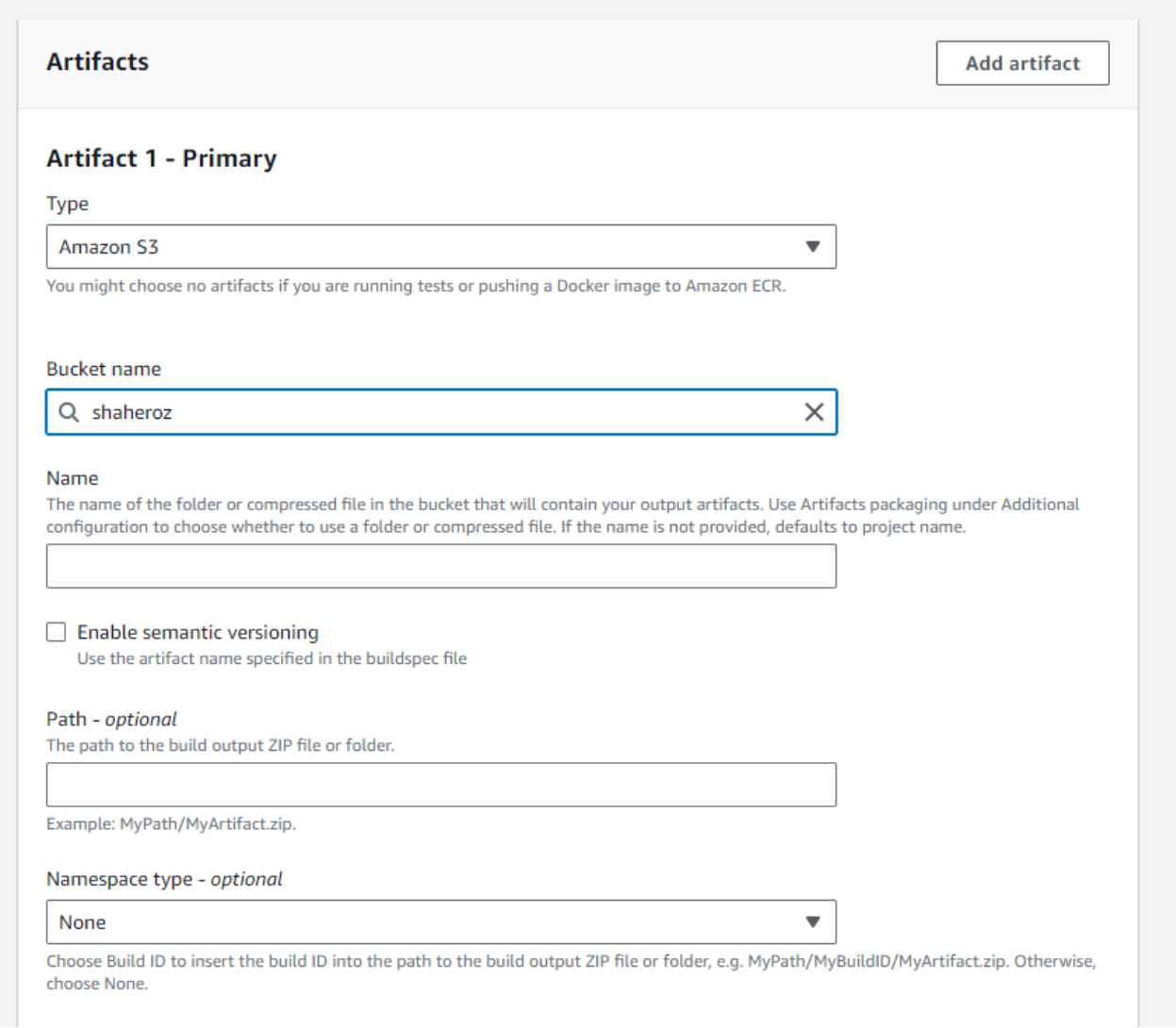
To save artifact



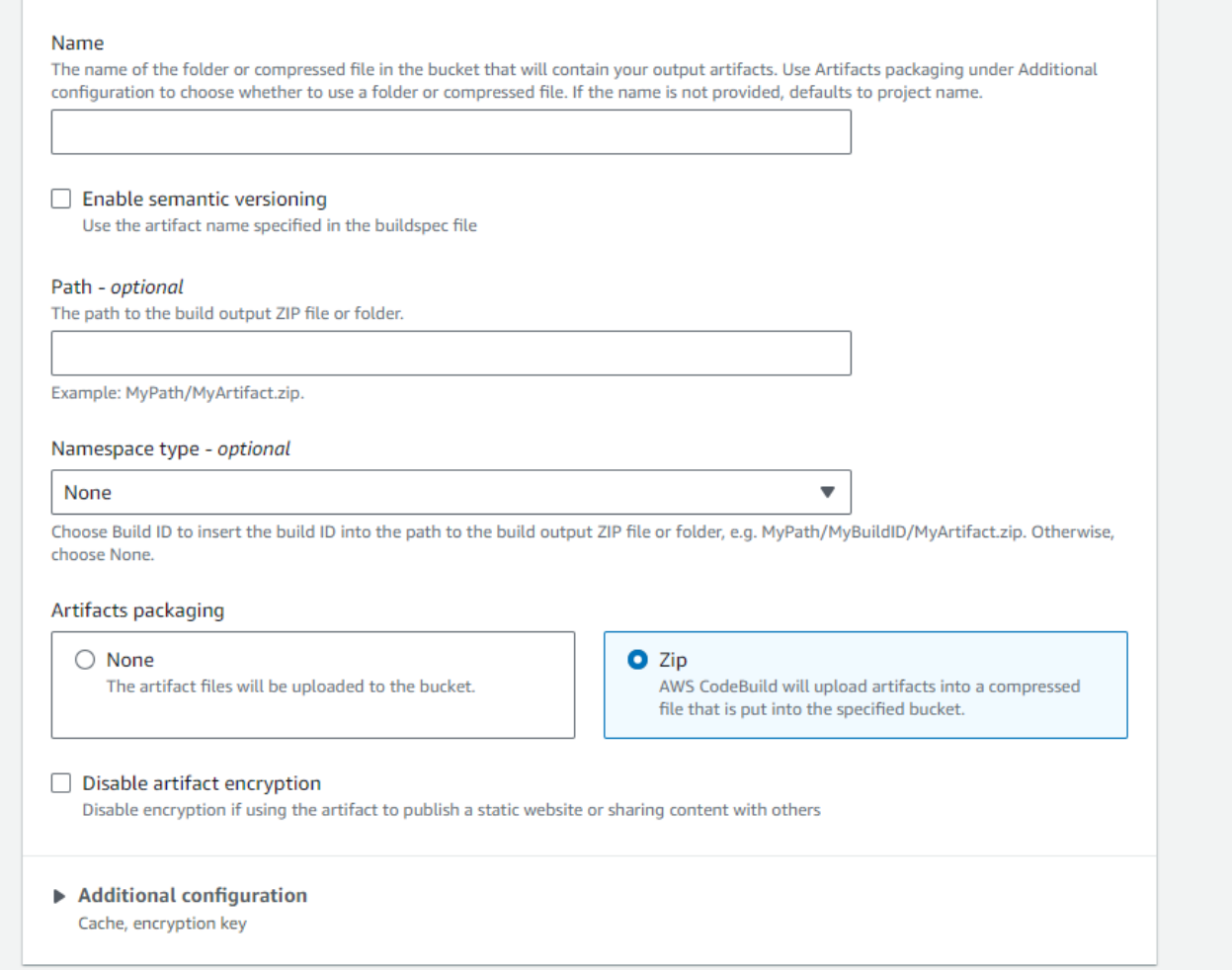
In artifact choose type Amazon S3

Seletect the bucket

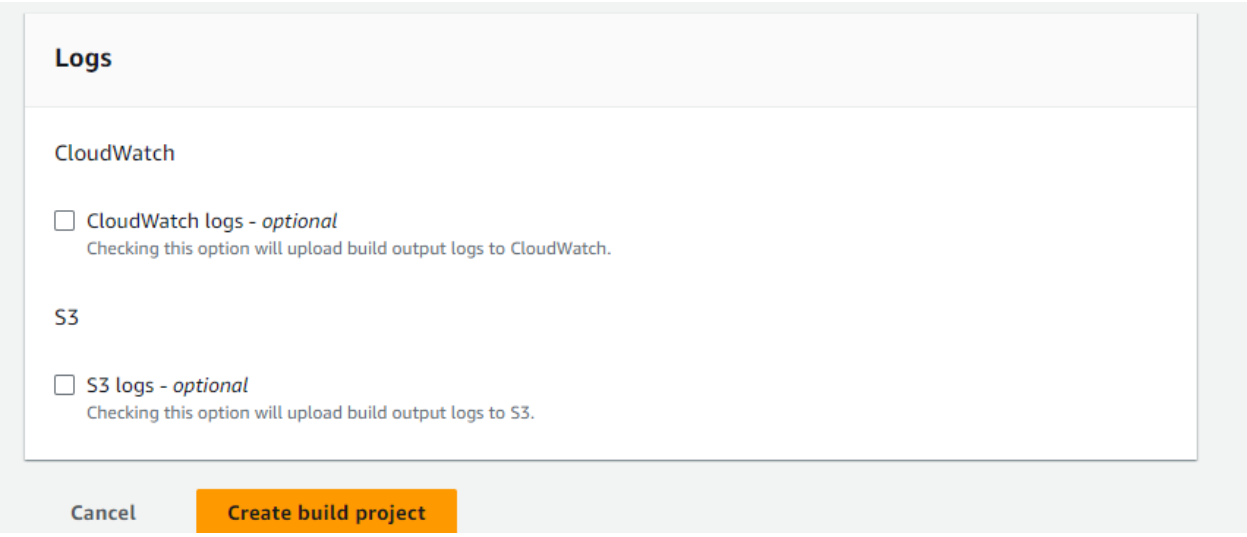
Remaining is by default



Select Artifact packing **.zip**

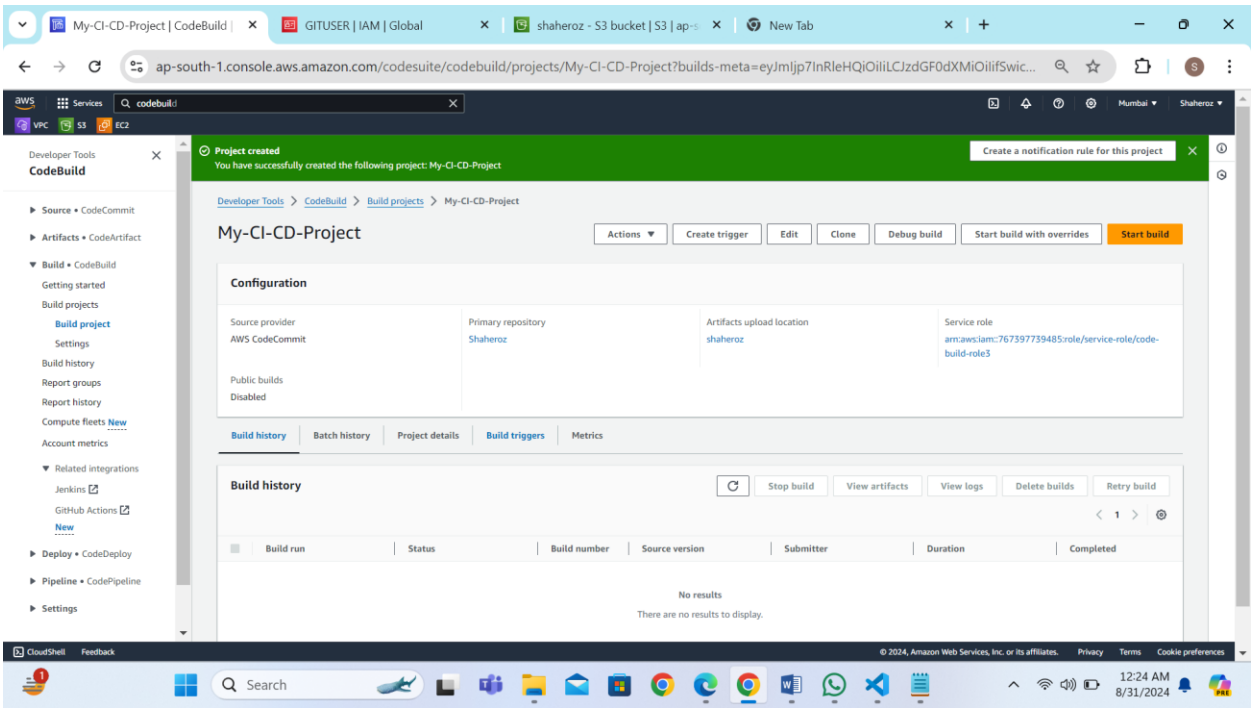


Click **Create build project**.

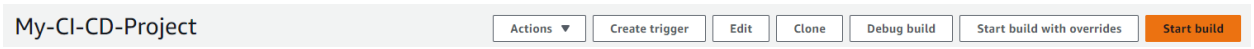


SHAHEROZ ALAM KHAN

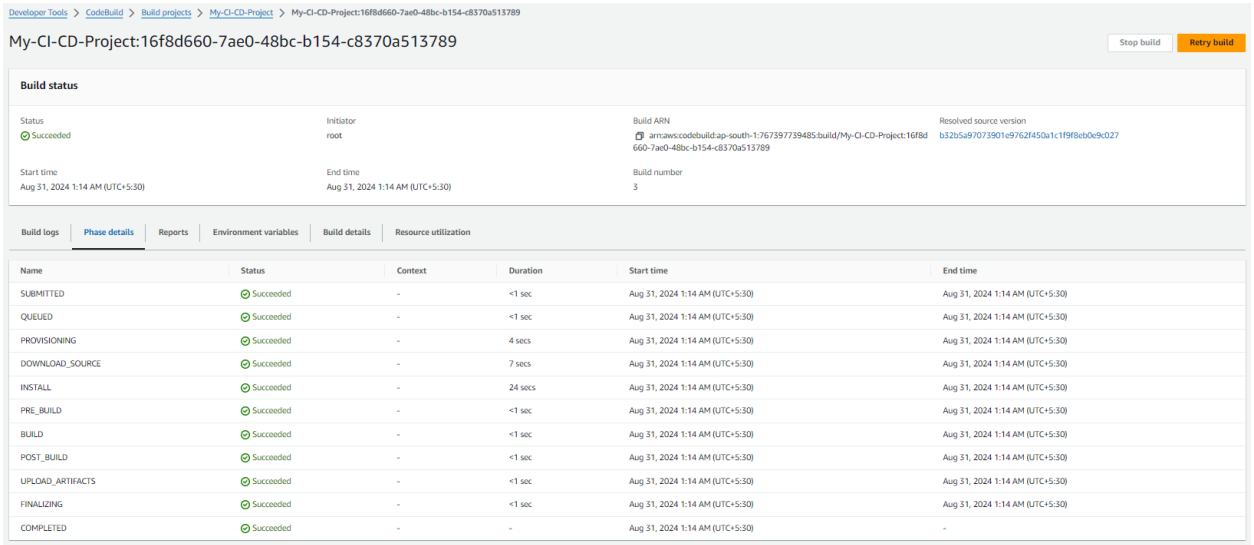
Now the code build is ready



Click on start build

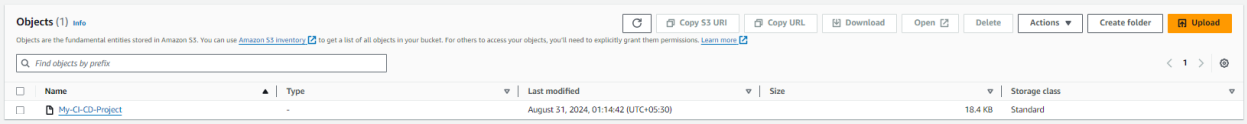


All the phases of code build must be succesfull

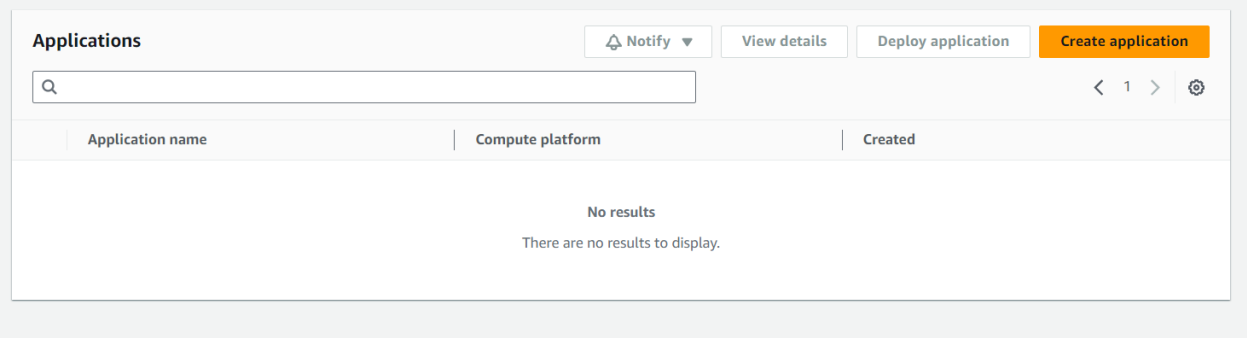


SHAHEROZ ALAM KHAN

Now the artifact file is stord in s3 bucket



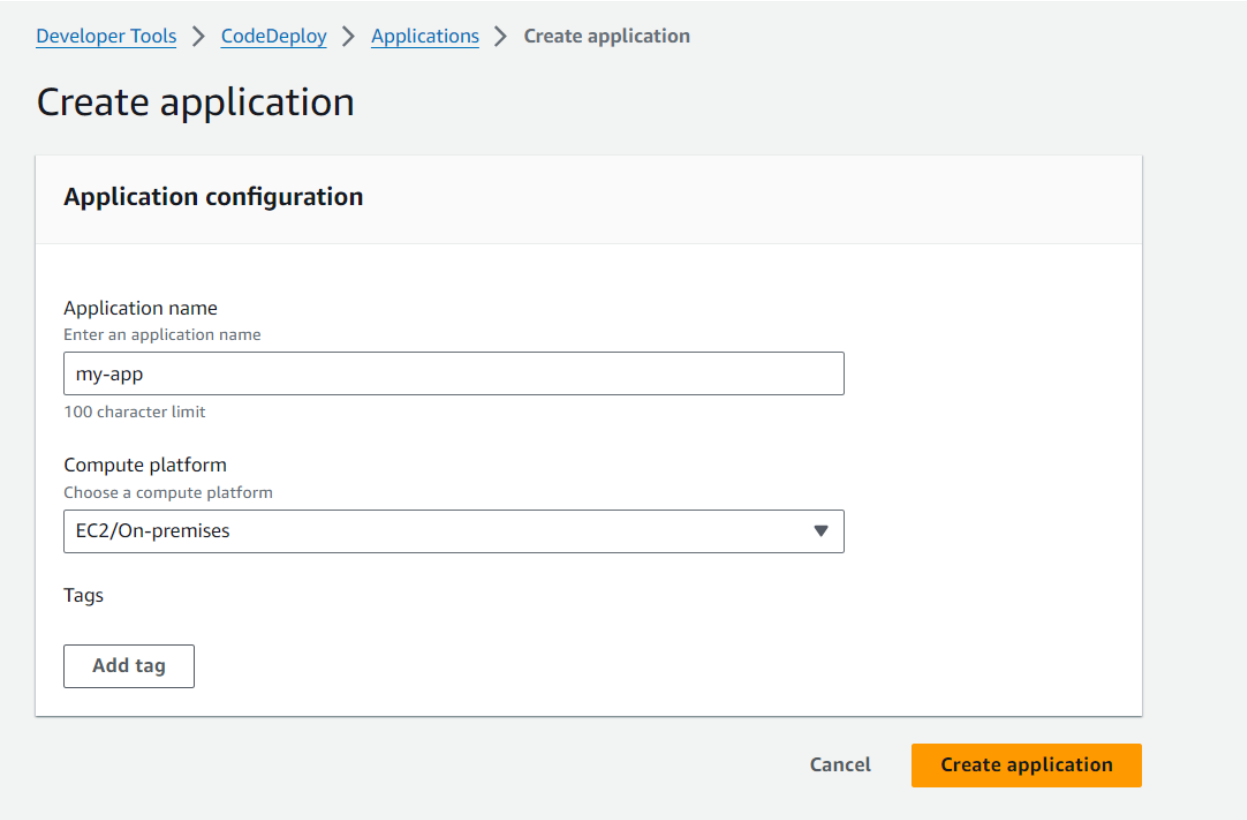
Set Up AWS CodeDeploy:



SHAHEROZ ALAM KHAN

Create an Application in CodeDeploy:

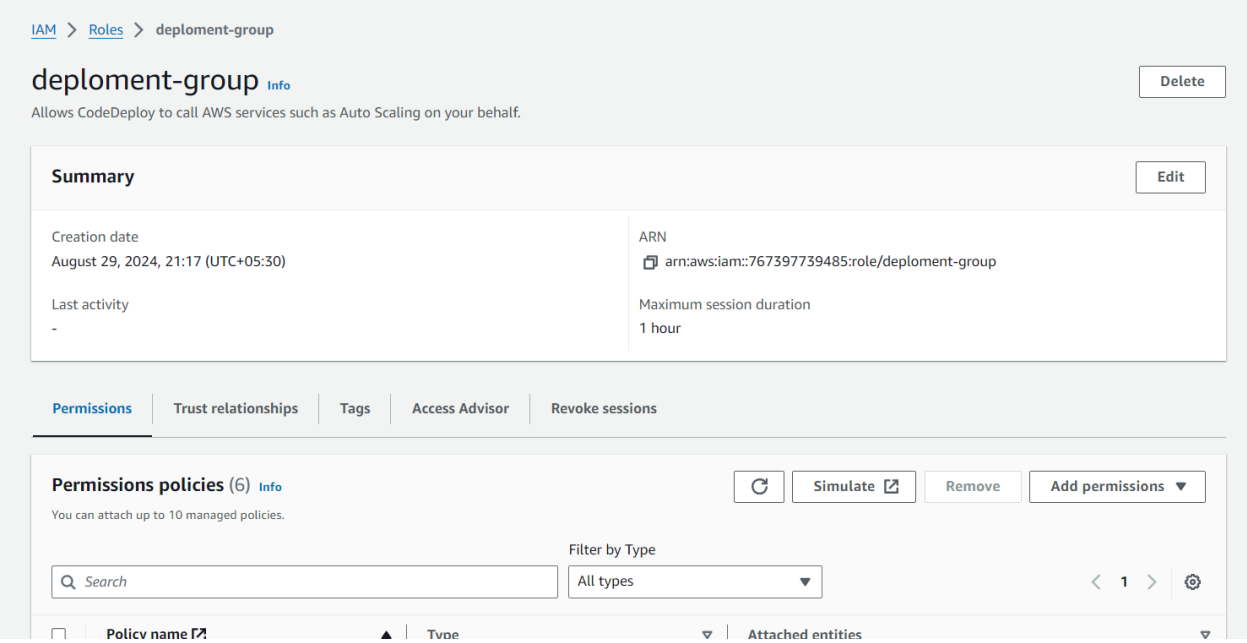
- Go to the AWS Management Console and open **CodeDeploy**.
- Click **Create application**.
- Enter an application name (e.g., `MyAppDeploy`).
- Choose the **Compute platform** (e.g., EC2/On-premises).



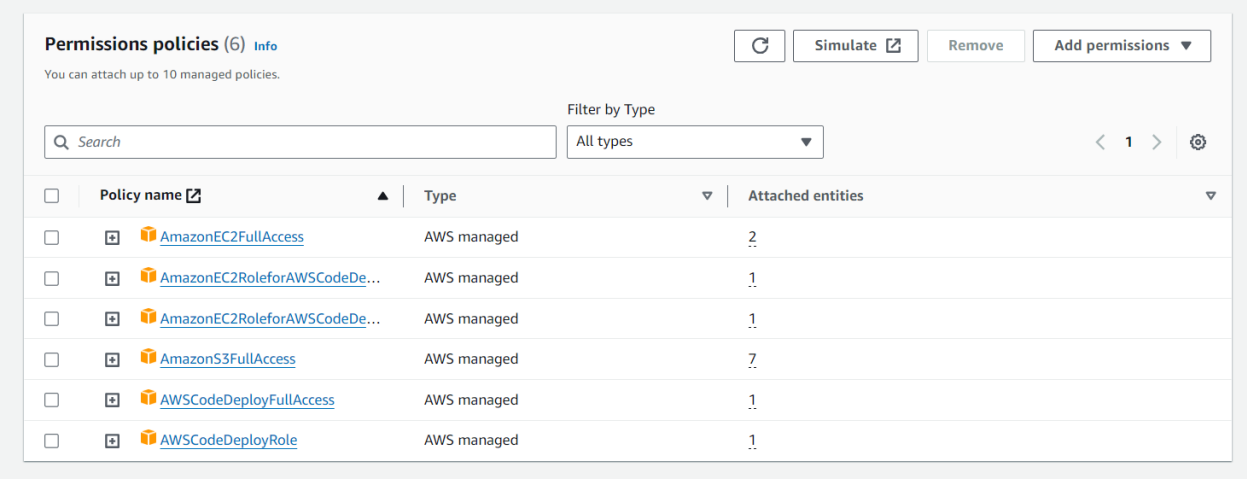
SHAHEROZ ALAM KHAN

Create IAM role deployment group

Aws service>codedeploy



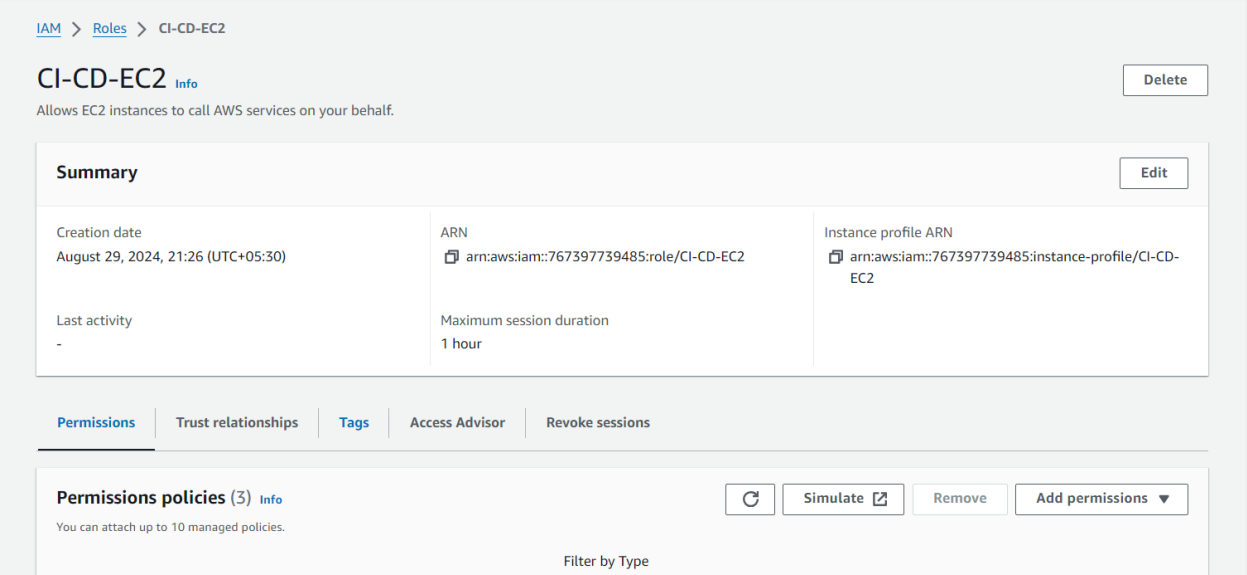
Attach these policies



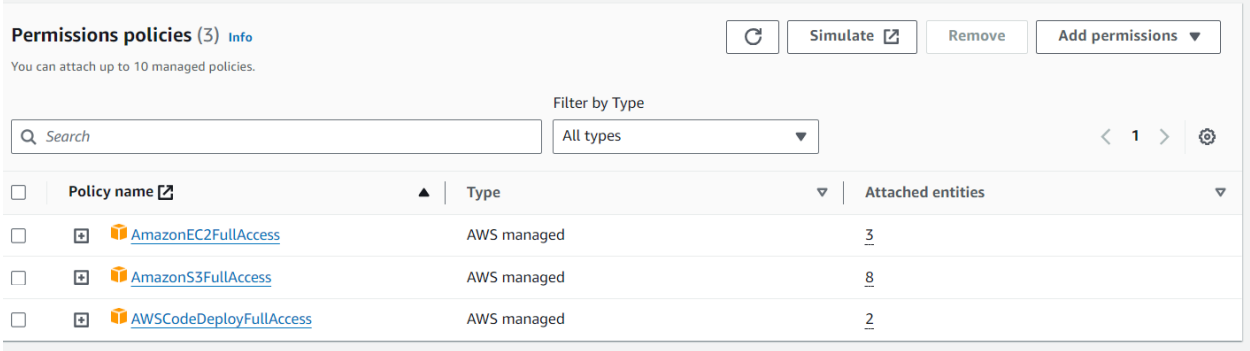
SHAHEROZ ALAM KHAN

Create IAM role for EC2 (server)

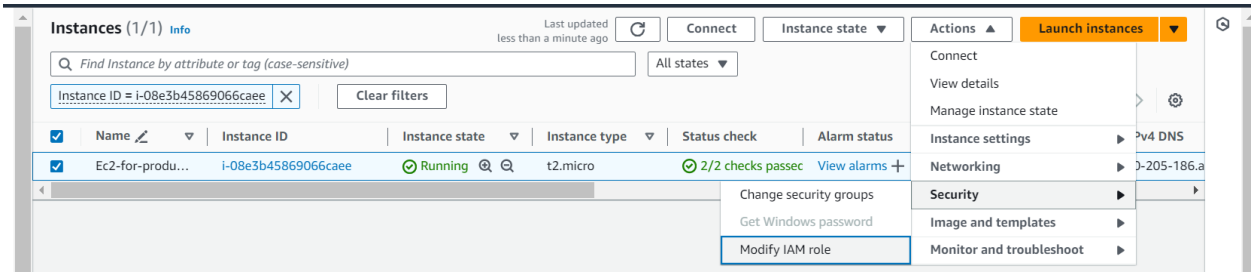
Aws service > EC2



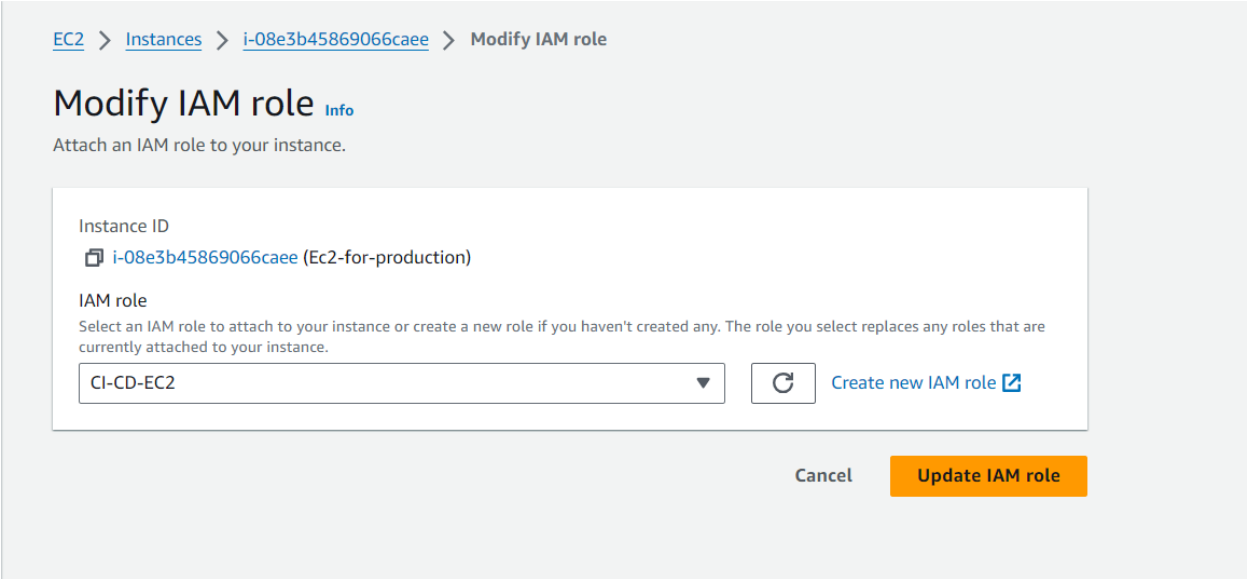
Attach these polices



Lauch a ec2 instace Ubuntu 22.04 (Allow SSH, HTTP AND HTTPS)



And update the IAM role



SHAHEROZ ALAM KHAN

Connect the ec2

And create a file with .sh extension

Paste the below content

#!/bin/bash

This installs the CodeDeploy agent and its prerequisites on Ubuntu 22.04.

sudo apt-get update

sudo apt-get install ruby-full ruby-webrick wget -y

cd /tmp

wget https://aws-codedeploy-ap-southeast-1.s3.ap-southeast-1.amazonaws.com/releases/codedeploy-agent_1.3.2-1902_all.deb

mkdir codedeploy-agent_1.3.2-1902_ubuntu22

dpkg-deb -R codedeploy-agent_1.3.2-1902_all.deb codedeploy-agent_1.3.2-1902_ubuntu22

sed 's/Depends:.*/Depends:ruby3.0/' -i ./codedeploy-agent_1.3.2-1902_ubuntu22/DEBIAN/control

dpkg-deb -b codedeploy-agent_1.3.2-1902_ubuntu22/

sudo dpkg -i codedeploy-agent_1.3.2-1902_ubuntu22.deb

systemctl list-units --type=service | grep codedeploy

sudo service codedeploy-agent status

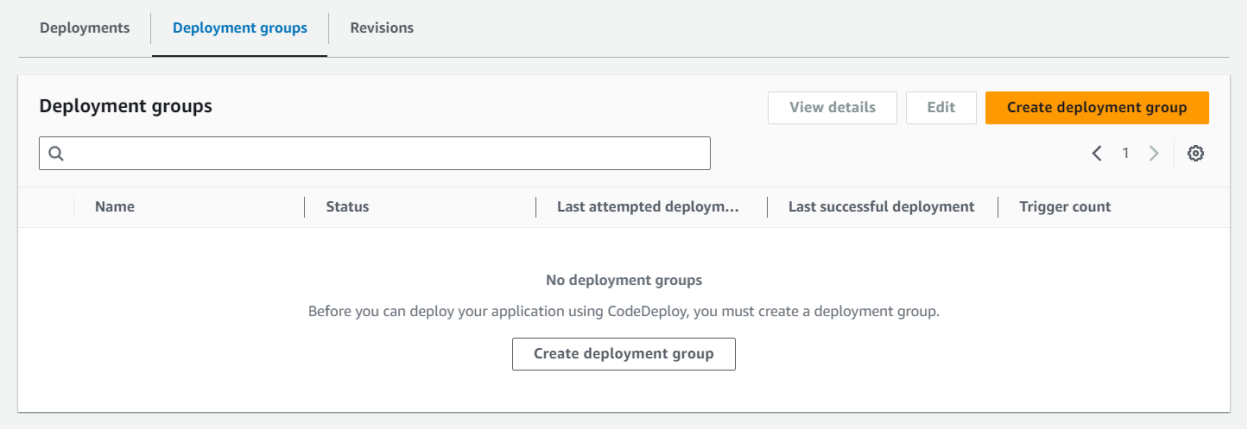
```
root@ip-172-31-46-226:/home/ubuntu# vi install.sh

root@ip-172-31-46-226:/home/ubuntu# cat install.sh
#!/bin/bash
# This installs the CodeDeploy agent and its prerequisites on Ubuntu 22.04.
sudo apt-get update
sudo apt-get install ruby-full ruby-webrick wget -y
cd /tmp
wget https://aws-codedeploy-ap-southeast-1.s3.ap-southeast-1.amazonaws.com/releases/codedeploy-agent_1.3.2-1902_all.deb
mkdir codedeploy-agent_1.3.2-1902_ubuntu22
dpkg-deb -R codedeploy-agent_1.3.2-1902_all.deb codedeploy-agent_1.3.2-1902_ubuntu22
sed 's/Depends:.*/Depends:ruby3.0/' -i ./codedeploy-agent_1.3.2-1902_ubuntu22/DEBIAN/control
dpkg-deb -b codedeploy-agent_1.3.2-1902_ubuntu22/
sudo dpkg -i codedeploy-agent_1.3.2-1902_ubuntu22.deb
systemctl list-units --type=service | grep codedeploy
sudo service codedeploy-agent status
root@ip-172-31-46-226:/home/ubuntu# bash install.sh
```

- Run that file” bash install.sh”

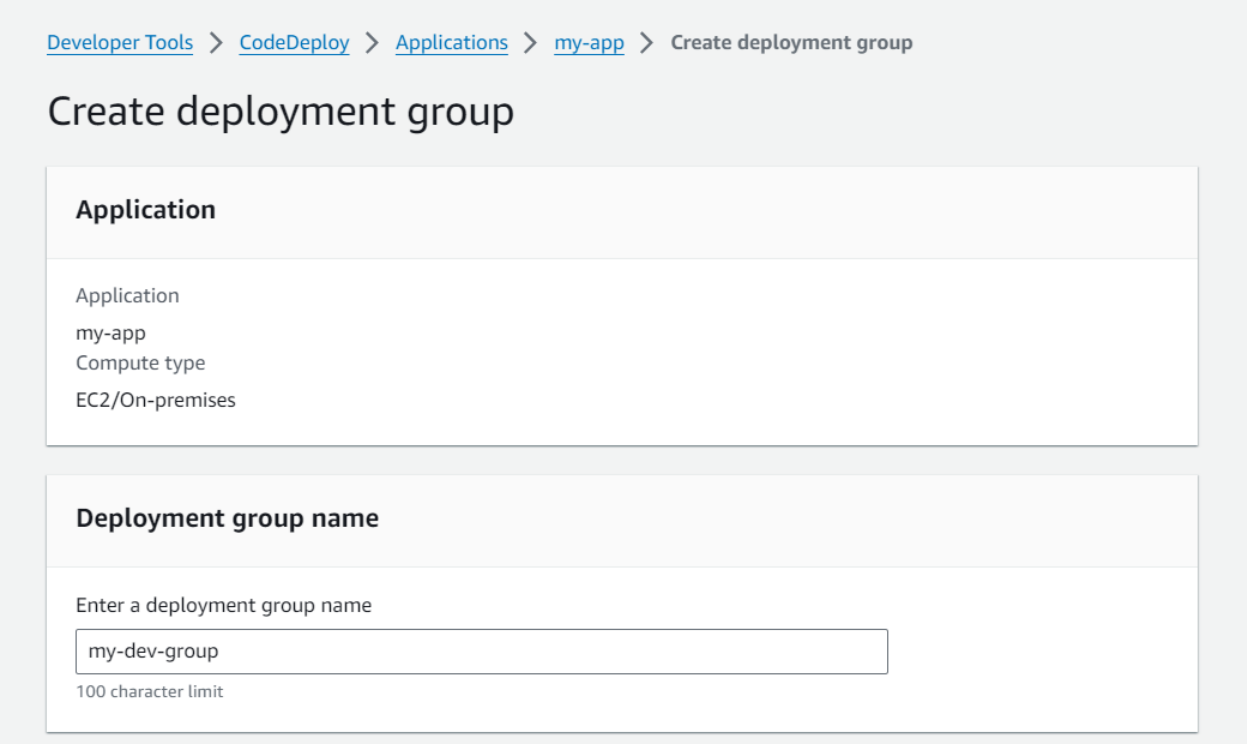
SHAHEROZ ALAM KHAN

Create a Deployment Group:

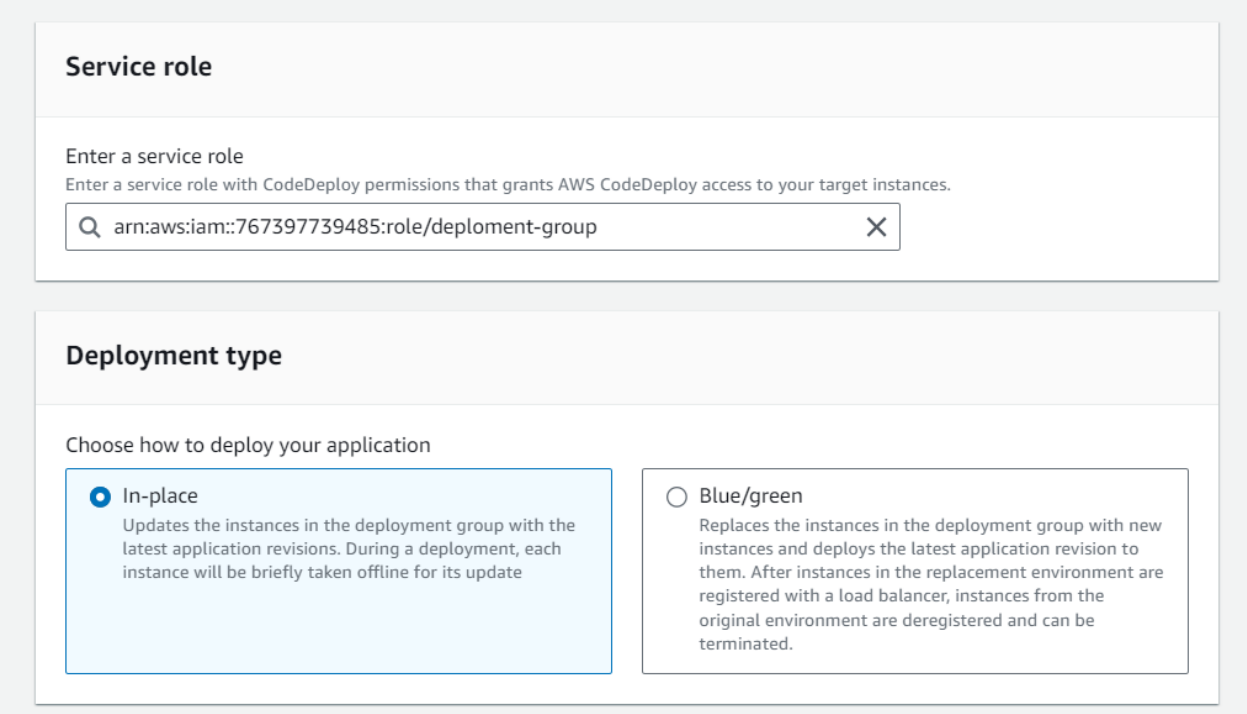
- Under your newly created application, click **Create deployment group**.



Enter a deployment group name (e.g., `MyAppDeployGroup`).

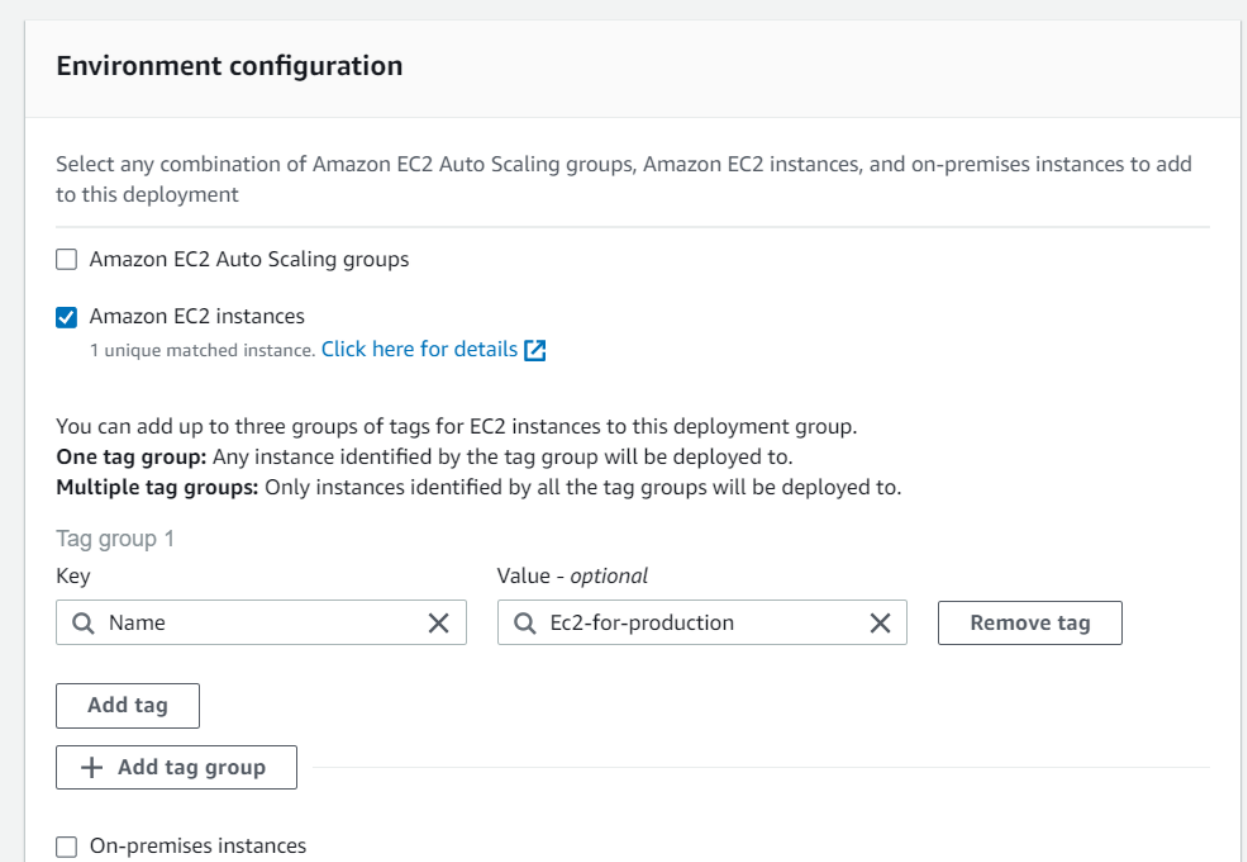


Service Role: Choose IAM role that created for deployment group



For EC2, specify the target EC2 instances by tags

Click **Create deployment group**.



SHAHEROZ ALAM KHAN

Create a Deployment Configuration File

Create a folder name script inside colone repository

And make two in script folder name install_nginx.sh and start_nginx.sh

And create a appspec.yml file in direct colone repository

Content of script folder:

under scripts

start_nginx.sh

```
#!/bin/bash

sudo service nginx start
```

---

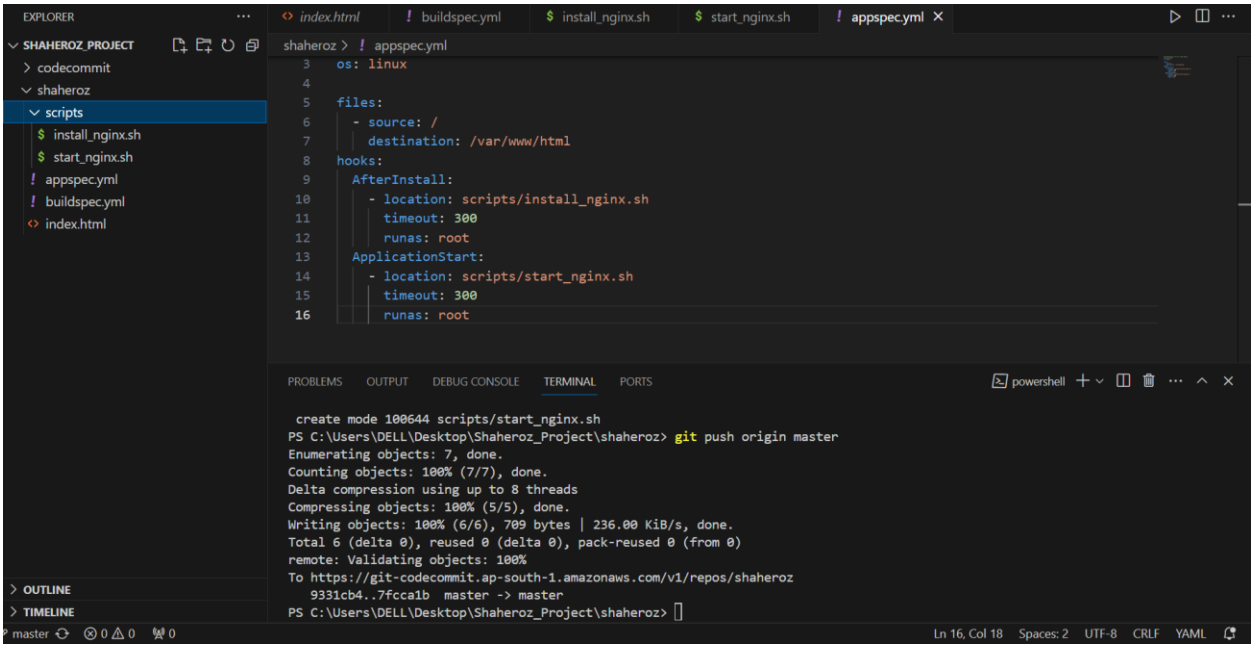
install_nginx.sh

```
#!/bin/bash

sudo apt-get update
sudo apt-get install -y nginx
```

Content of appspec.yml file

```
#configuration file for code deploy
version: 0.0
os: linux

files:
  - source: /
    destination: /var/www/html
hooks:
  AfterInstall:
    - location: scripts/install_nginx.sh
      timeout: 300
```

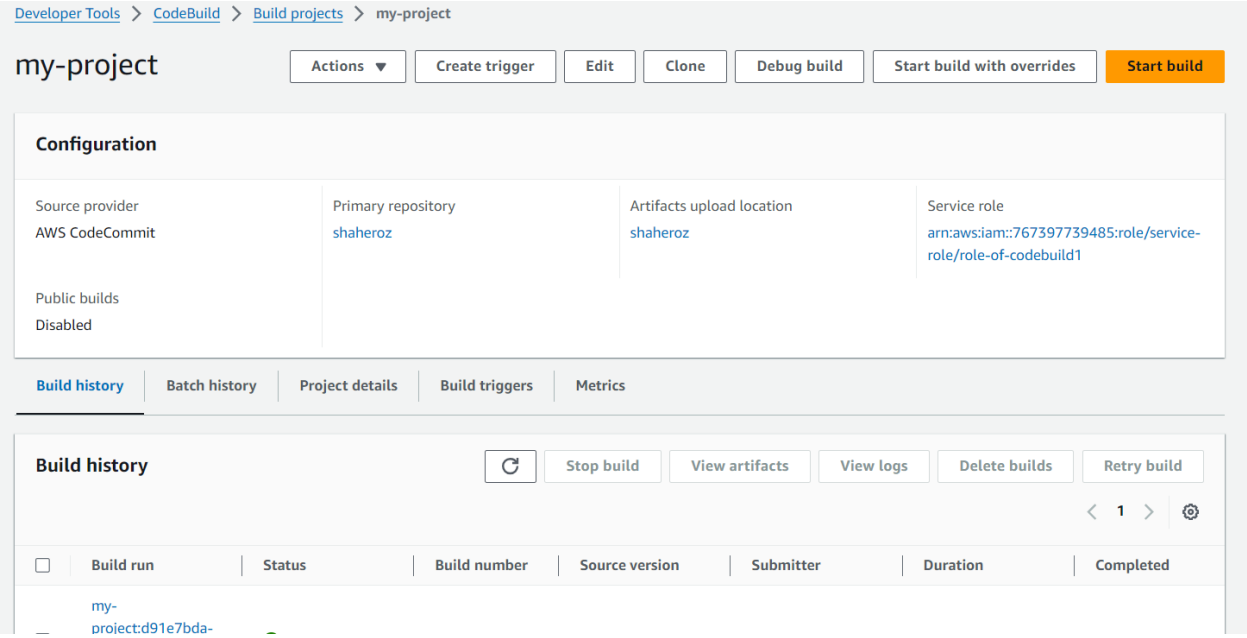
```
    runas: root
ApplicationStart:
  - location: scripts/start_nginx.sh
    timeout: 300
    runas: root
```

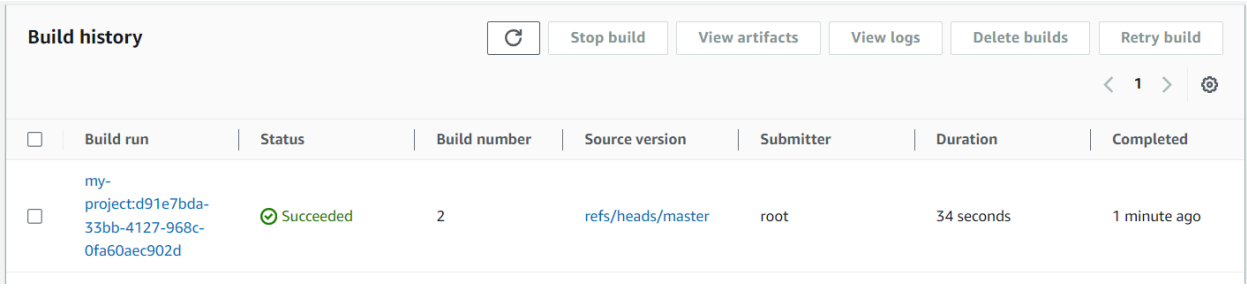


SHAHEROZ ALAM KHAN

Now click on start build in codebuild>aplliaction

All the phases should be successfull

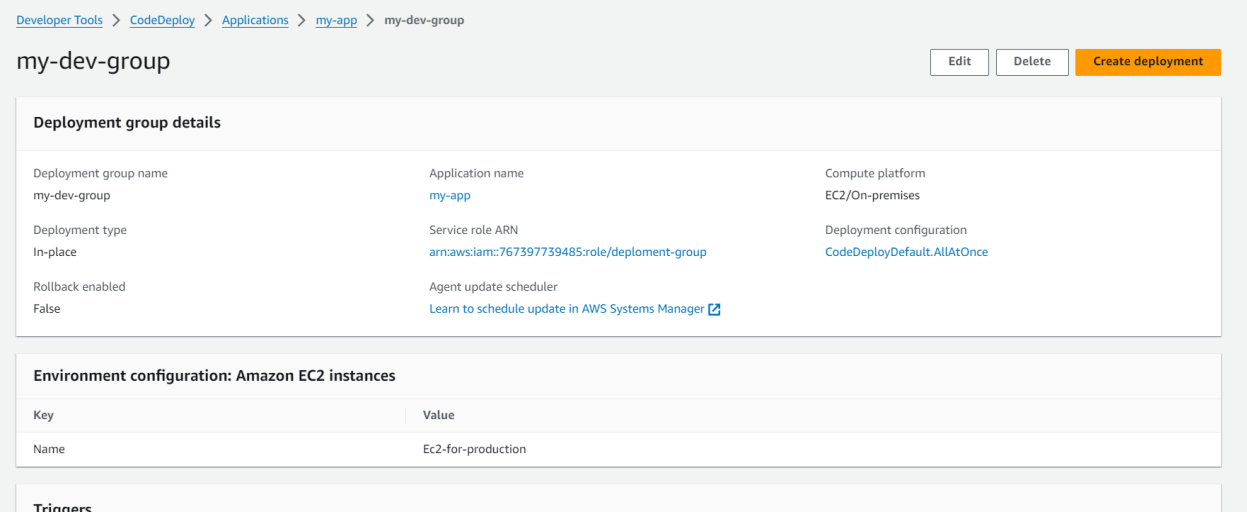




SHAHEROZ ALAM KHAN

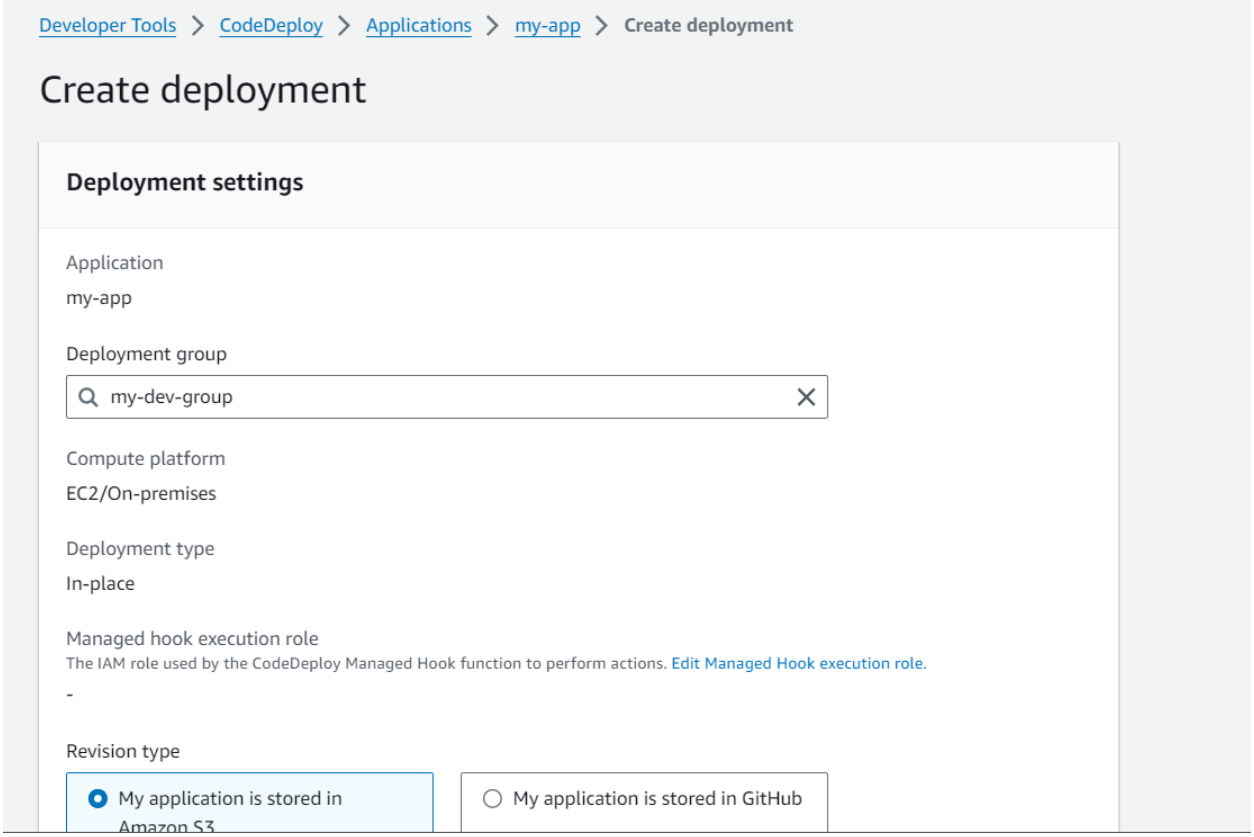
Create deployment inside the deployment group of codebuild application

Click on create deployment



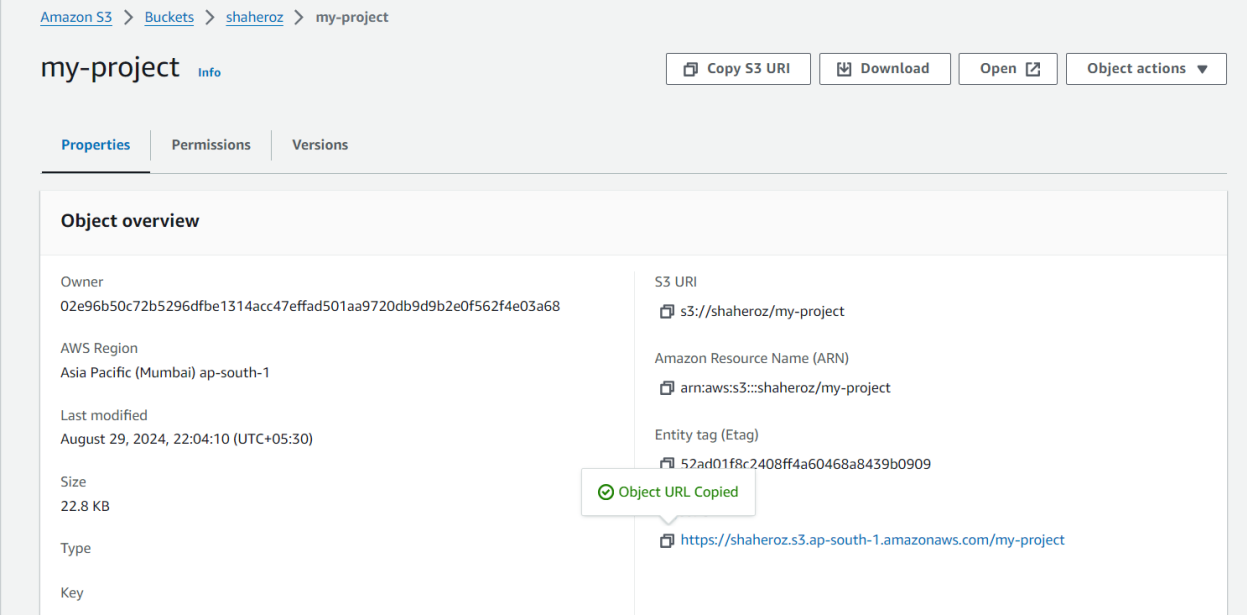
Enter the for deployment

Choose s3 in revision type



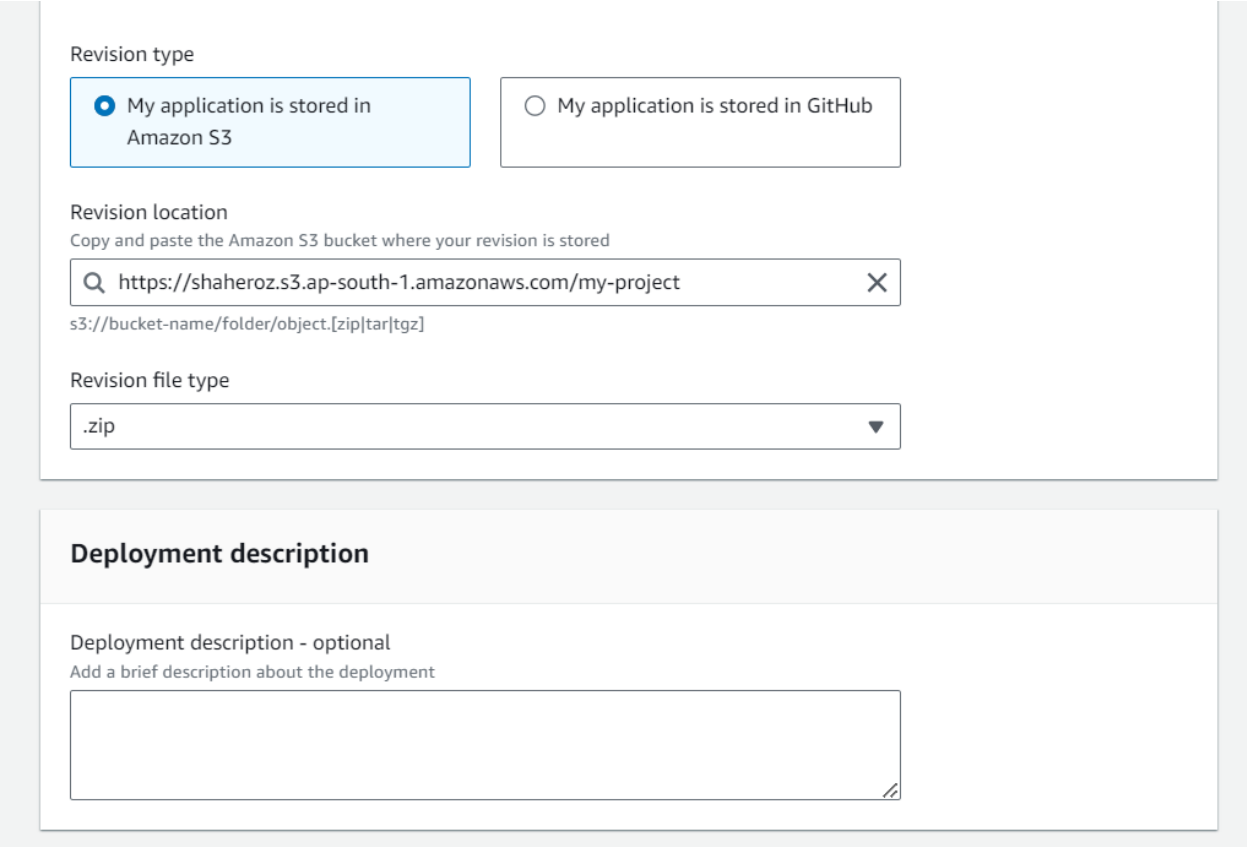
SHAHEROZ ALAM KHAN

Copy the uri of artifact (object)



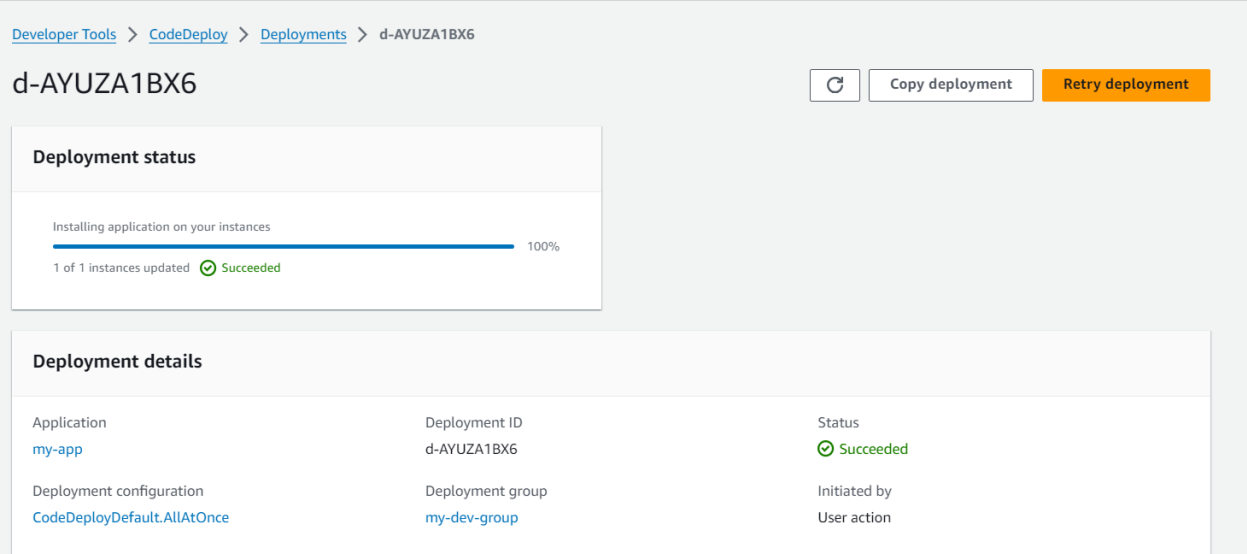
In revision location enter the uri of artifact

And chose file type .zip



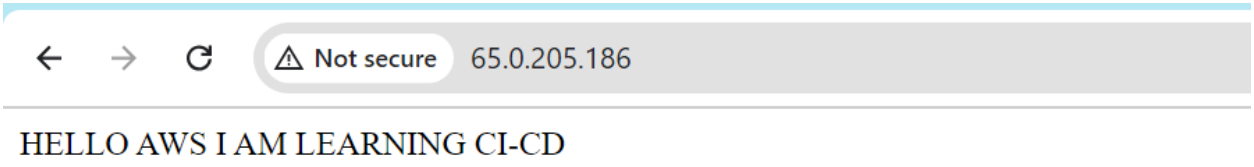
SHAHEROZ ALAM KHAN

Deployment status should 1 of 1 updated



SHAHEROZ ALAM KHAN

Now check with the public of ec2 that you application deployed

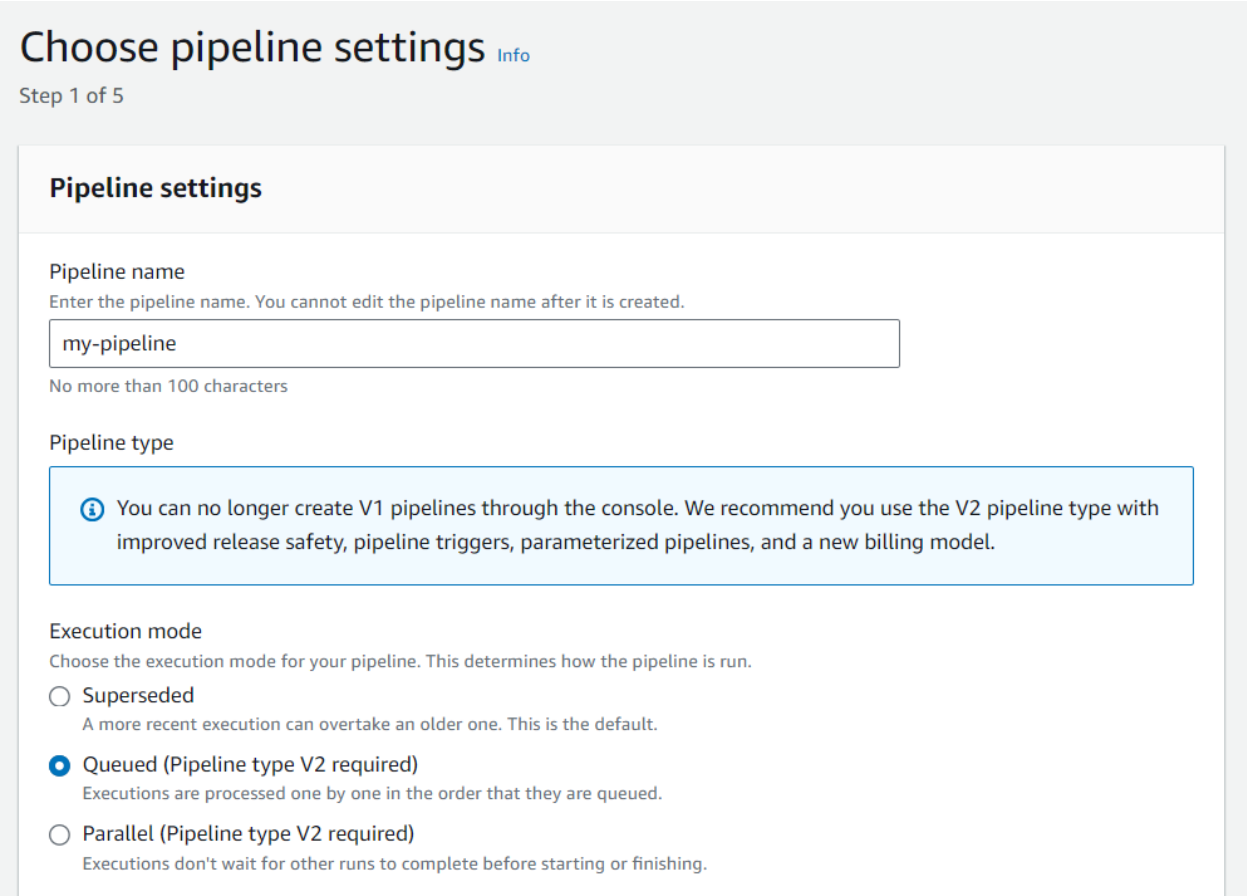


HELLO AWS I AM LEARNING CI-CD

SHAHEROZ ALAM KHAN

Create a CI/CD Pipeline in AWS CodePipeline

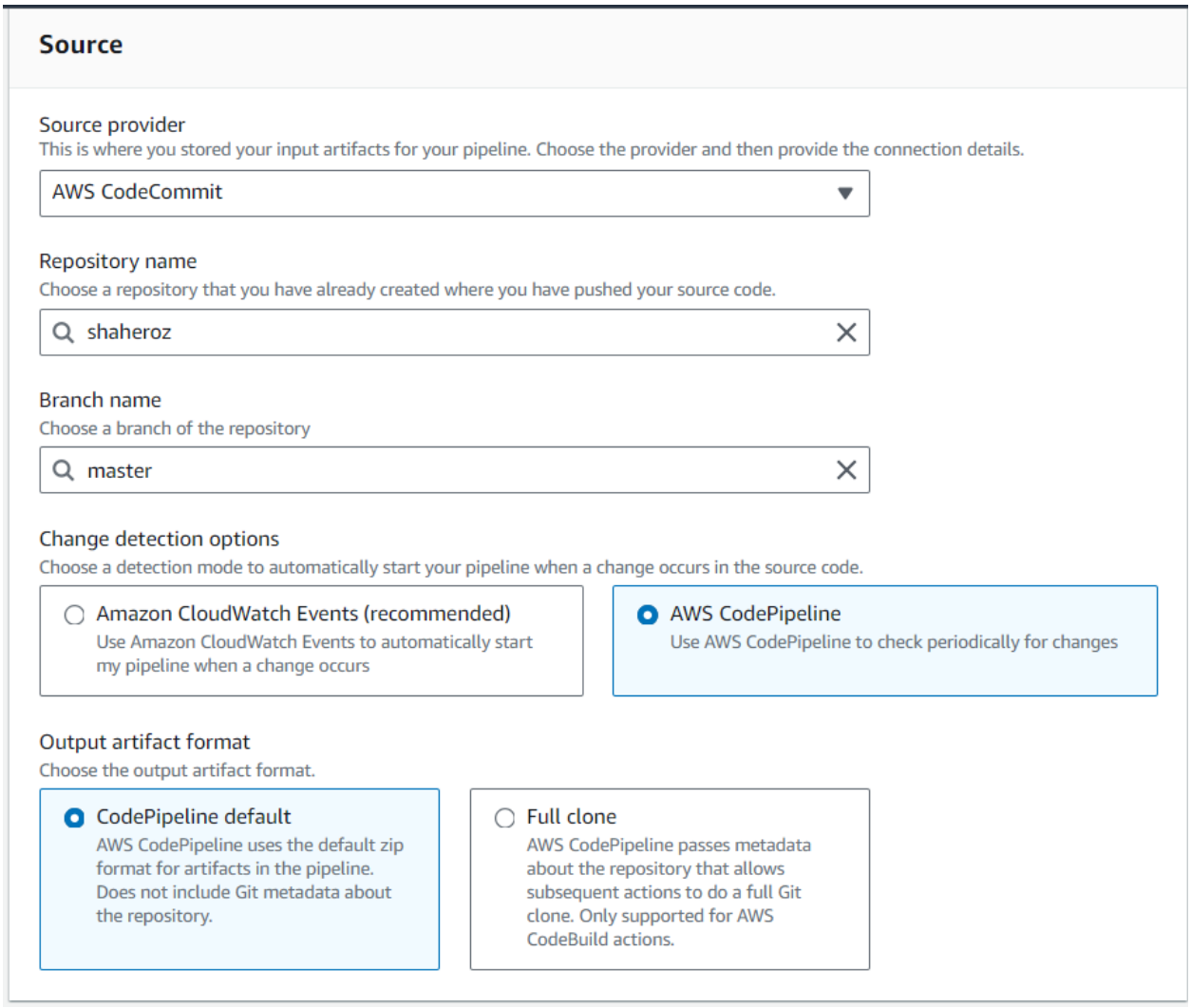
1. **Create a New Pipeline**:
    - Go to the AWS Management Console and open **CodePipeline**.
    - Click **Create pipeline**.
    - Enter a pipeline name (e.g., `MyAppPipeline`).
    - Select execution mode (queued (pipeline type v2 required))



SHAHEROZ ALAM KHAN

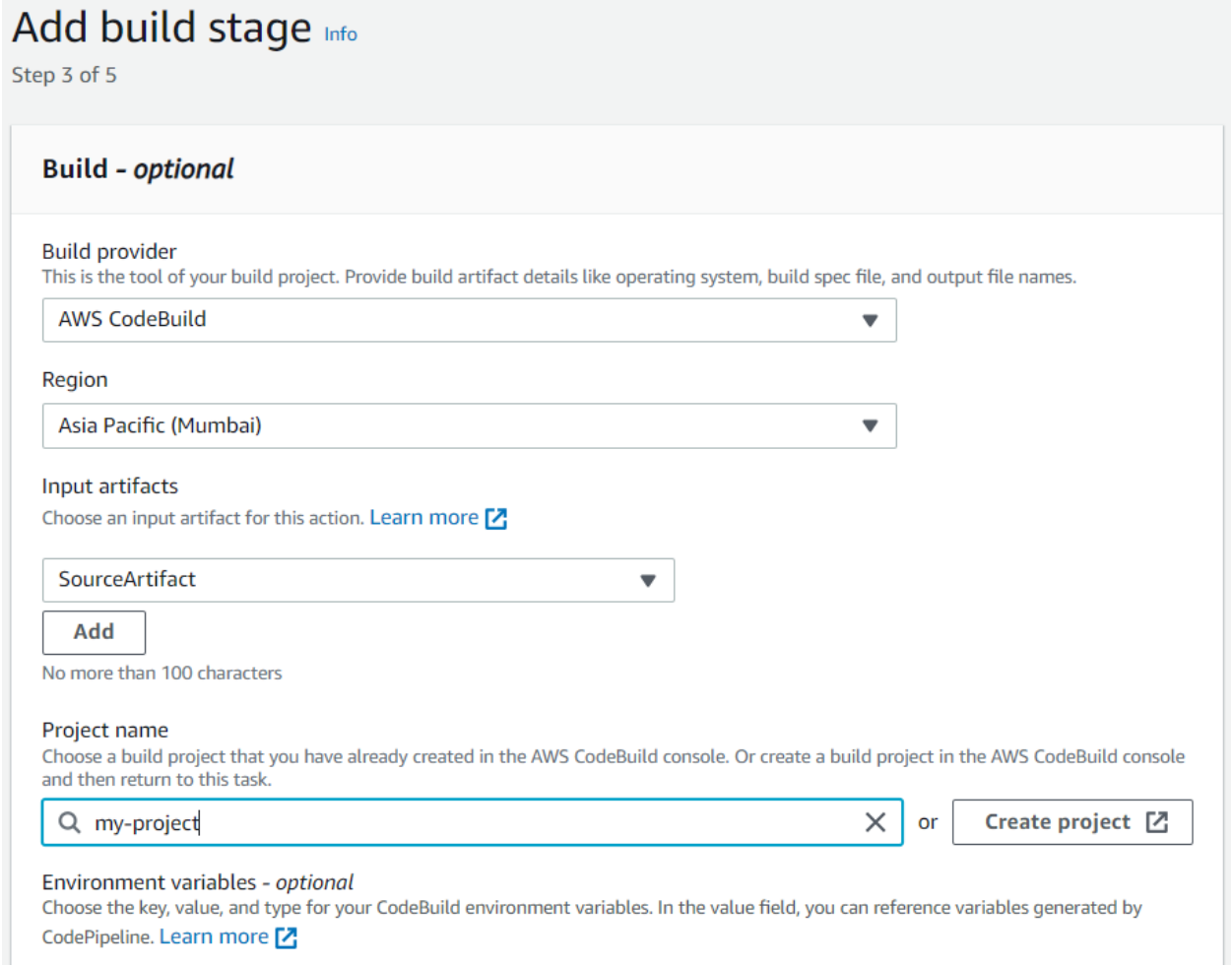
Source Stage:

- Select the source provider CodeCommit
- Choose the repository and branch.
- Detection option AWS CodePipeline



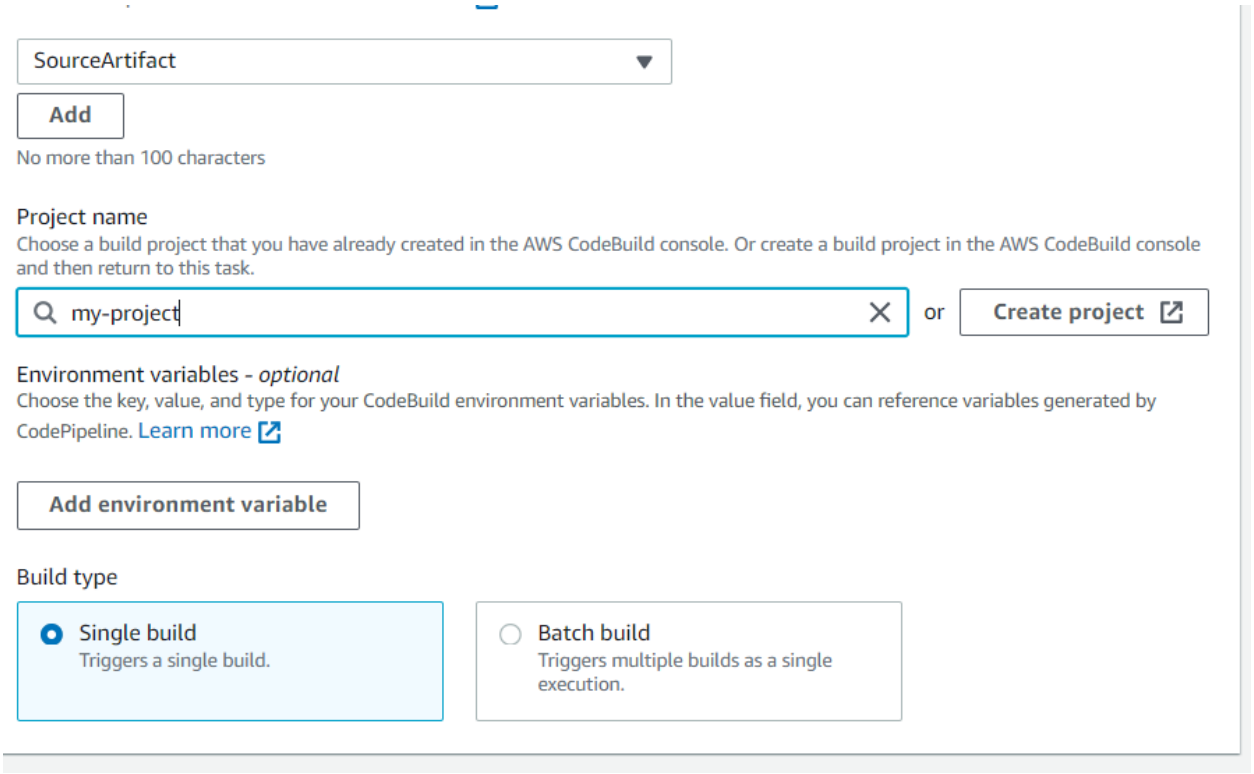
Build Stage:

- Select **AWS CodeBuild**.
- Input artifact (source artifact)
- Choose the build project created earlier (`MyAppBuild`)



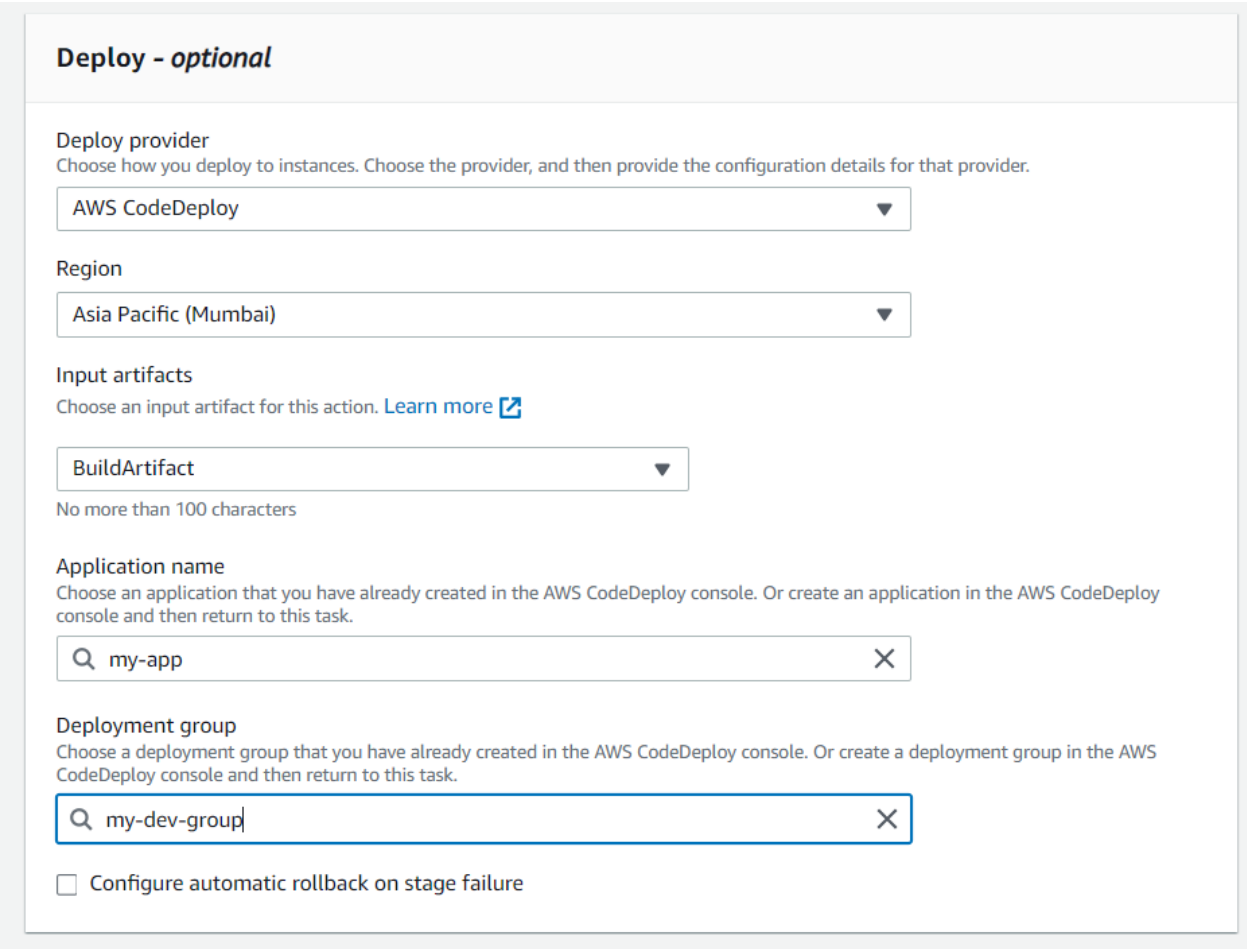
SHAHEROZ ALAM KHAN

Build type (Single build)

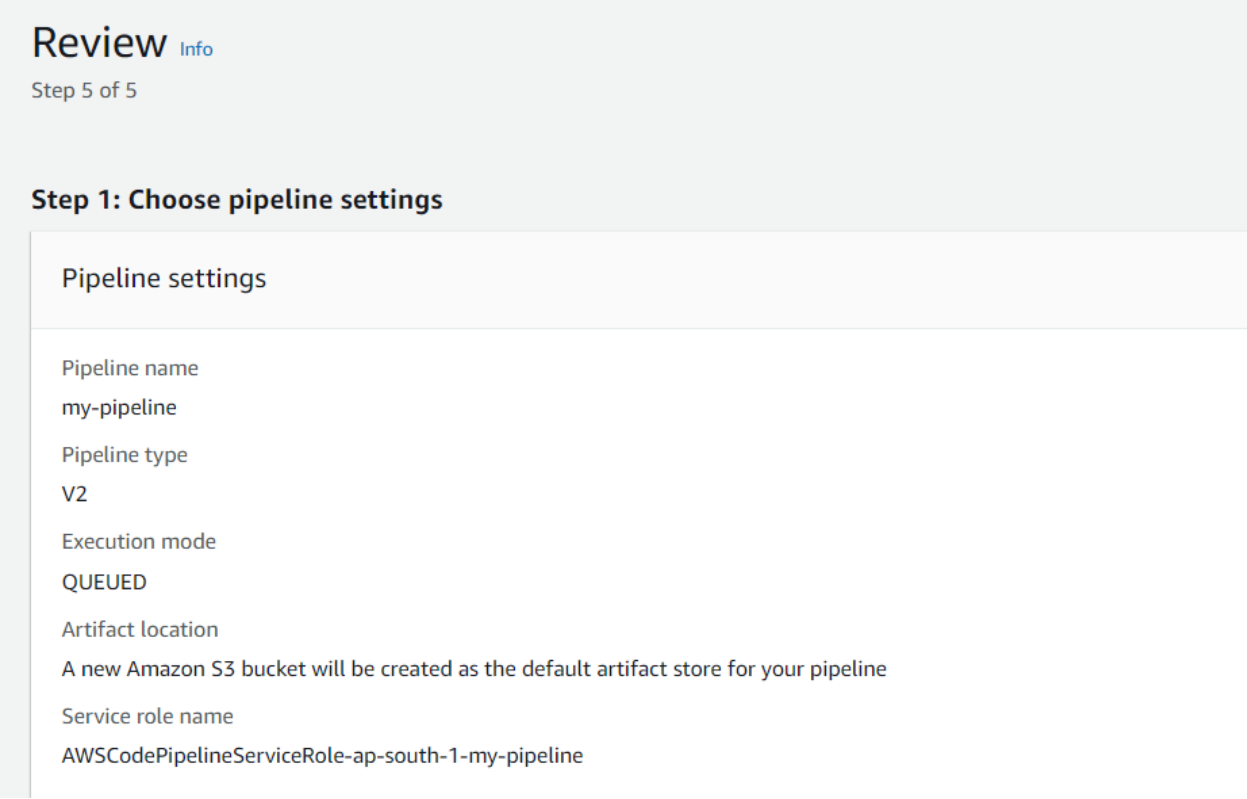


Deploy Stage:

- If using CodeDeploy, select **AWS CodeDeploy**.
- Choose the application and deployment group created earlier

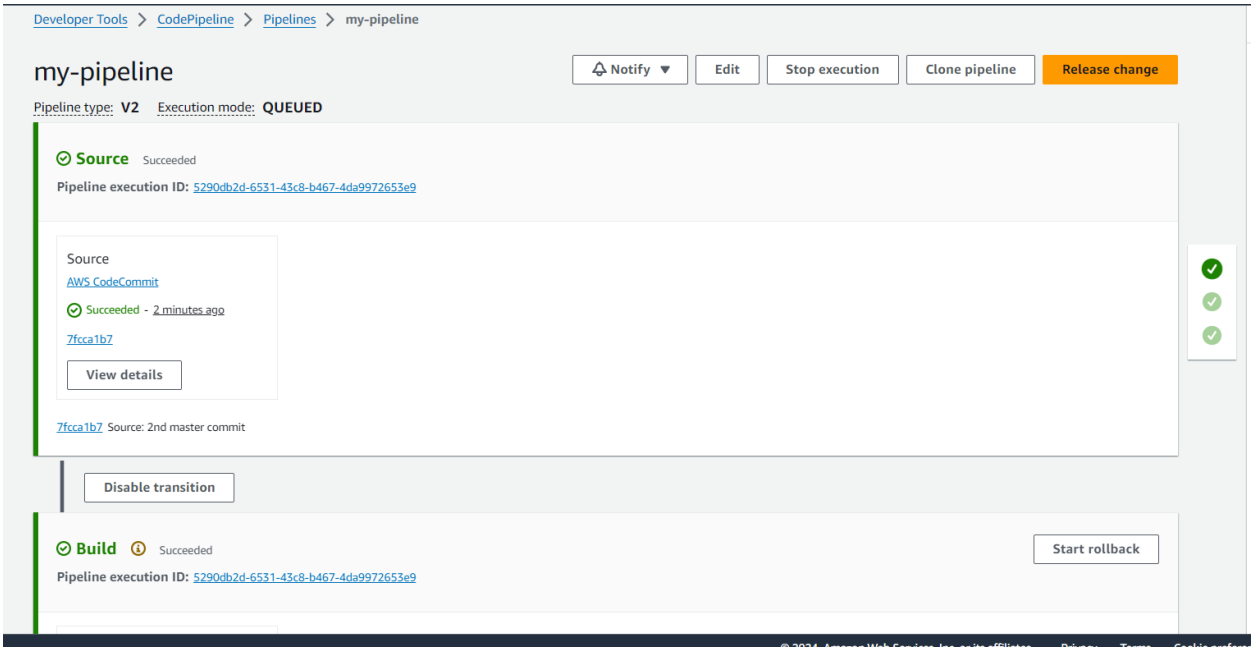


Click **Next** and then **Create pipeline**.

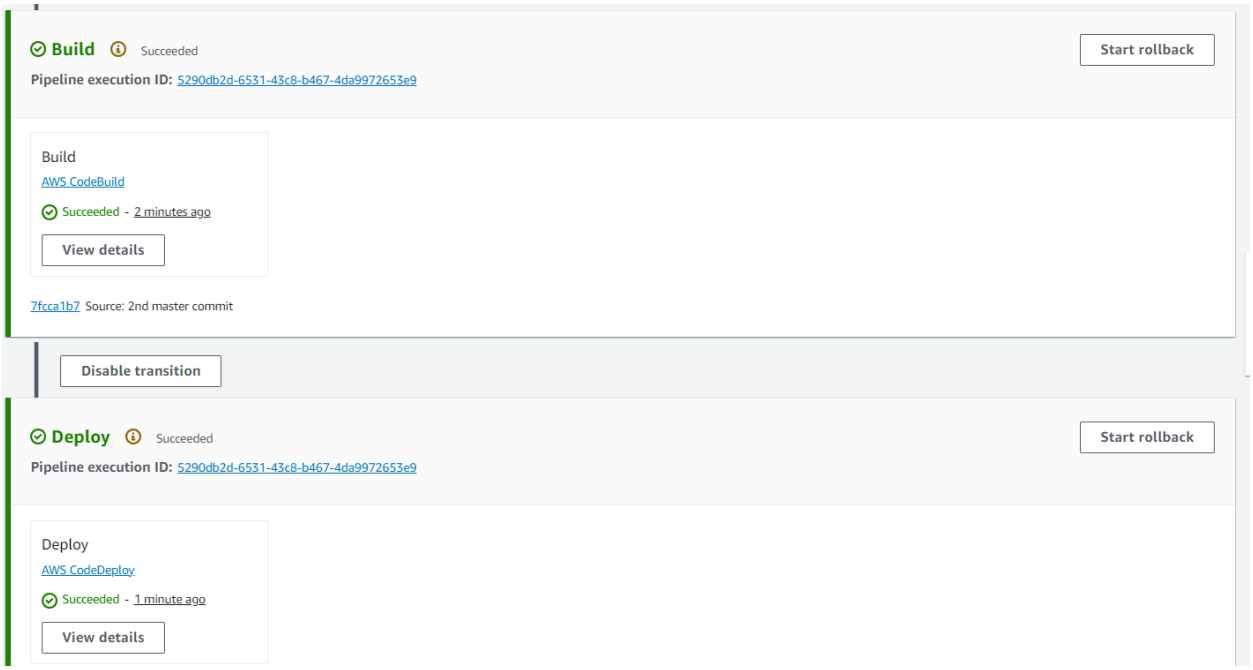


Now the pipeline stages in progress

After succeesfull of all stages.



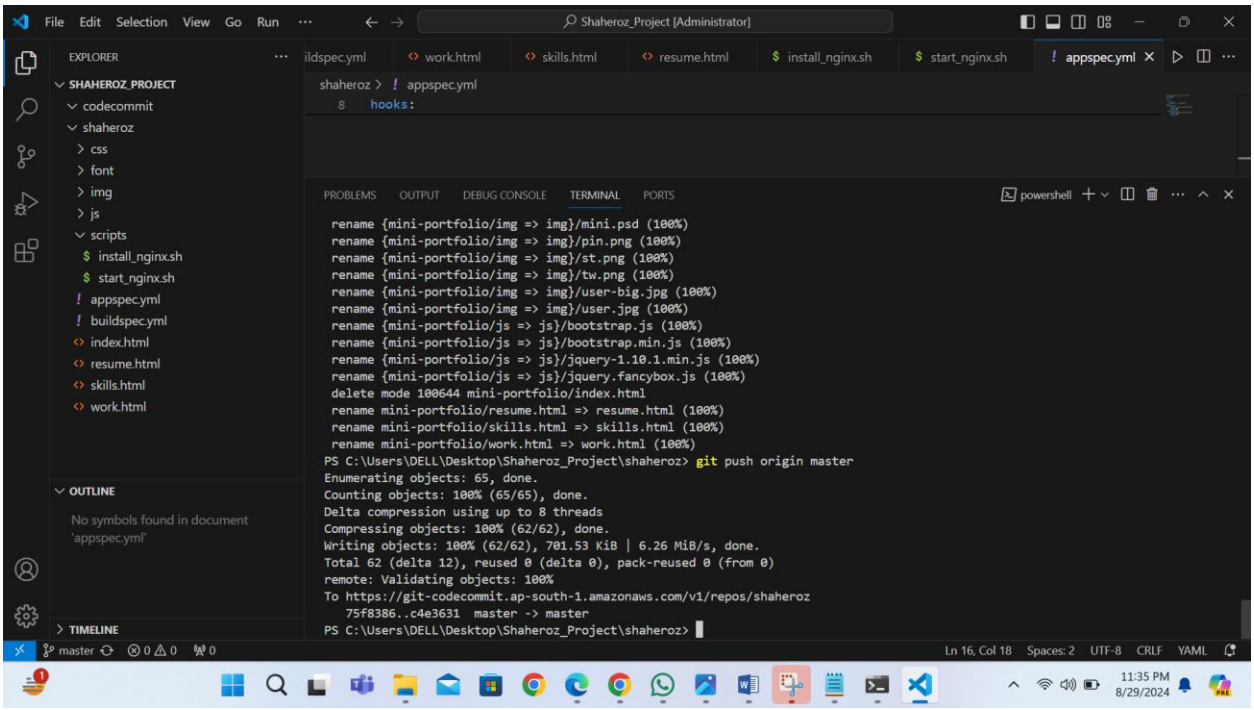
SHAHEROZ ALAM KHAN



SHAHEROZ ALAM KHAN

Test and Validate the Pipeline

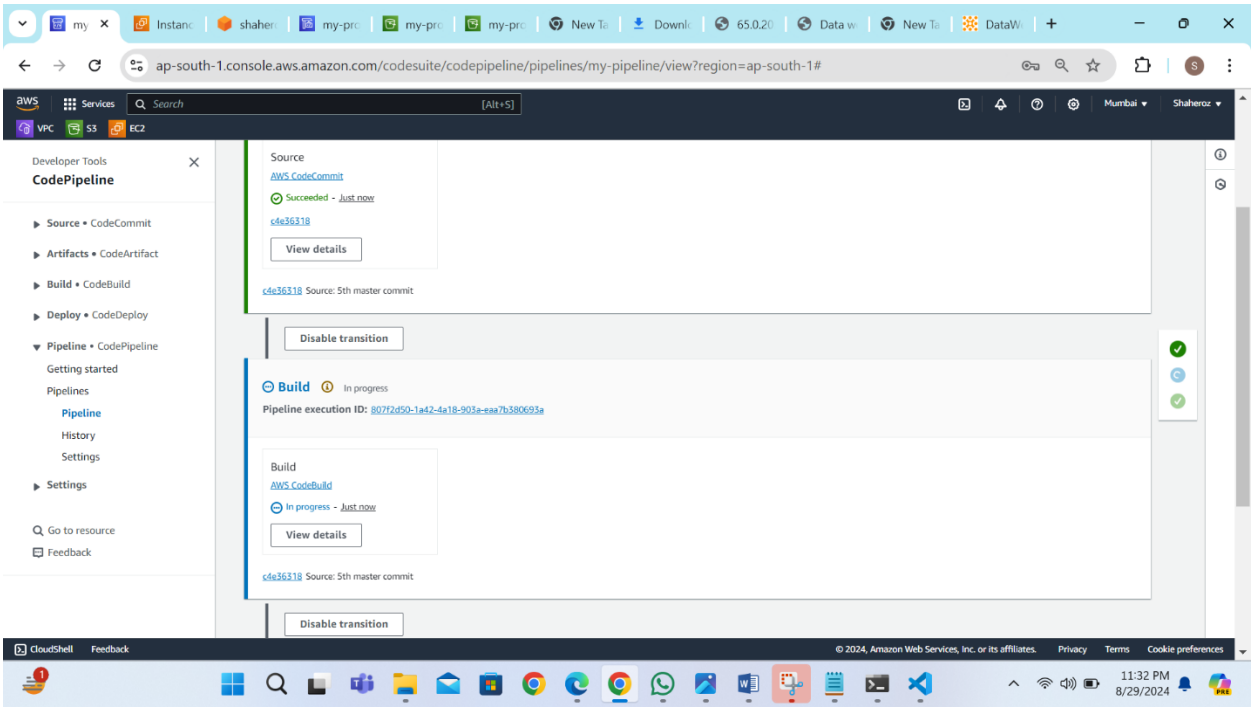
1. **Make a Code Change in VS Code**:
    - Edit a file or make a code change in your project.
    - Commit and push the changes to your repository.



SHAHEROZ ALAM KHAN

Monitor Pipeline Execution:

- Watch the pipeline execution in **AWS CodePipeline** from the AWS Management Console



SHAHEROZ ALAM KHAN

Ensure the entire CI/CD pipeline runs smoothly and deploys the changes to the production environment without errors.

SHAHEROZ ALAM KHAN