Project 2:

**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)

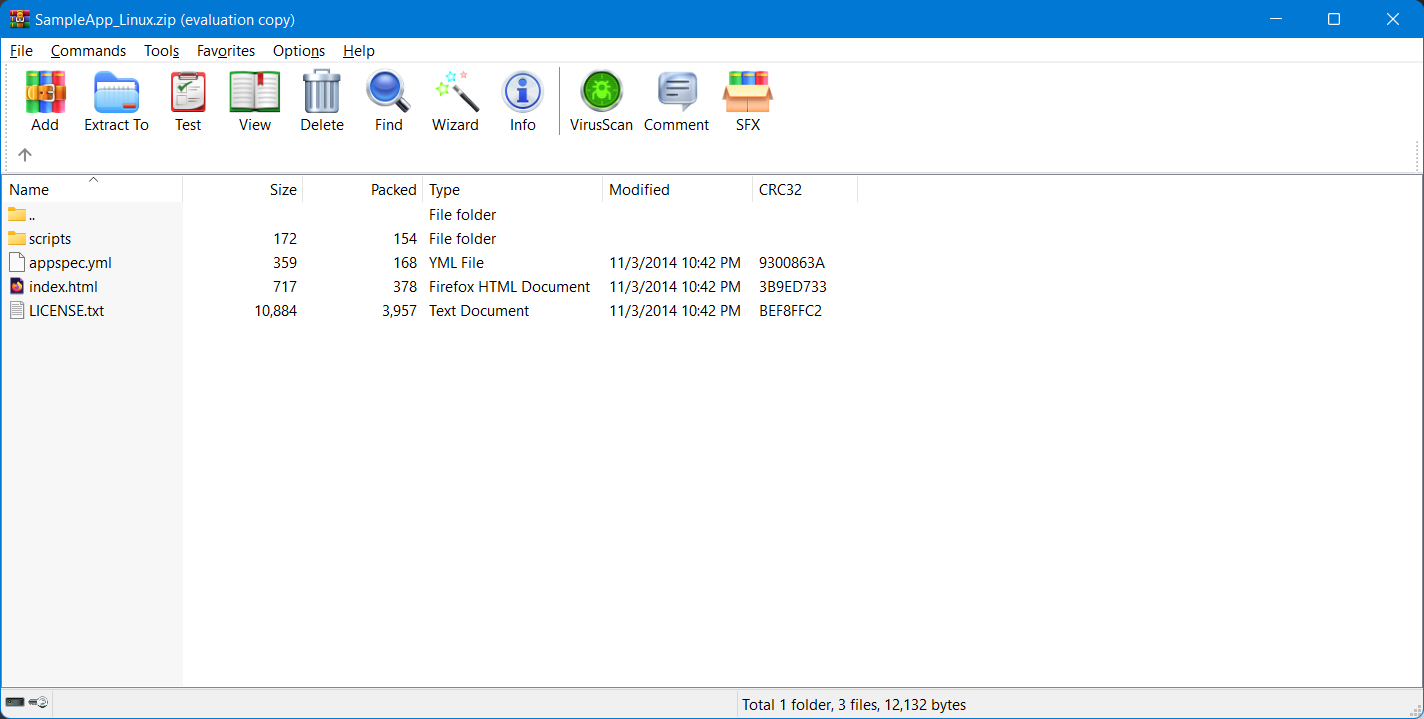
d. Create a pipeline using AWS CodePipeline , AWS Code Commit and Code Deploy to deploy index.html to Amazon Linux ec2 instance

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

--------------------------------------------------------------------------------------------------------

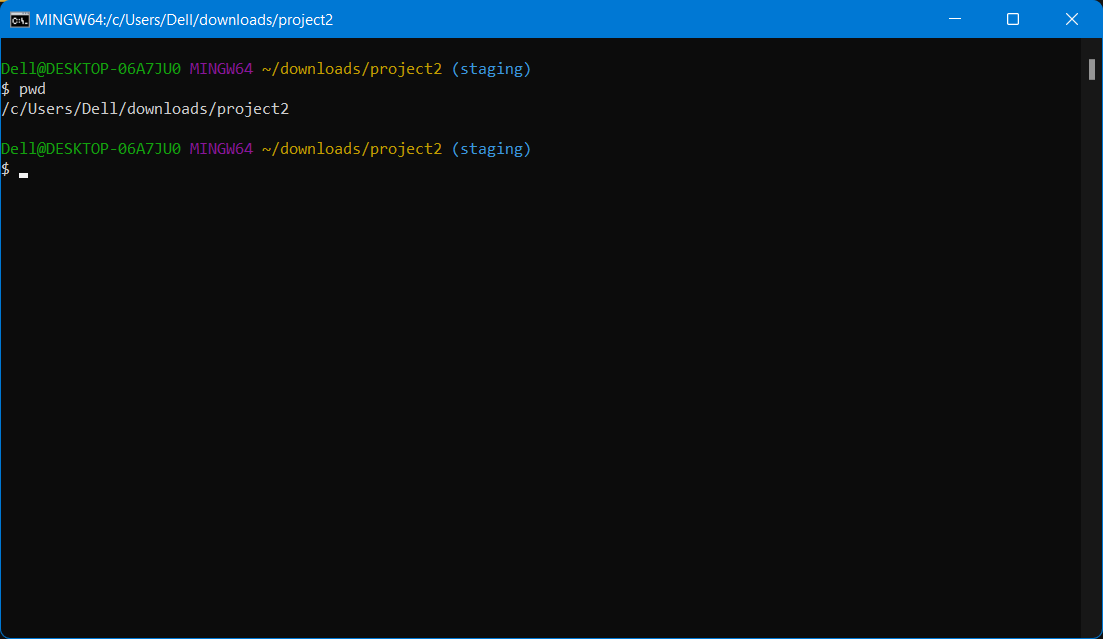
PART 1

STEP 1: First download and unzip the files into local repository.



STEP 2: Inside the folder of local repository right click and select open with Gitbash.

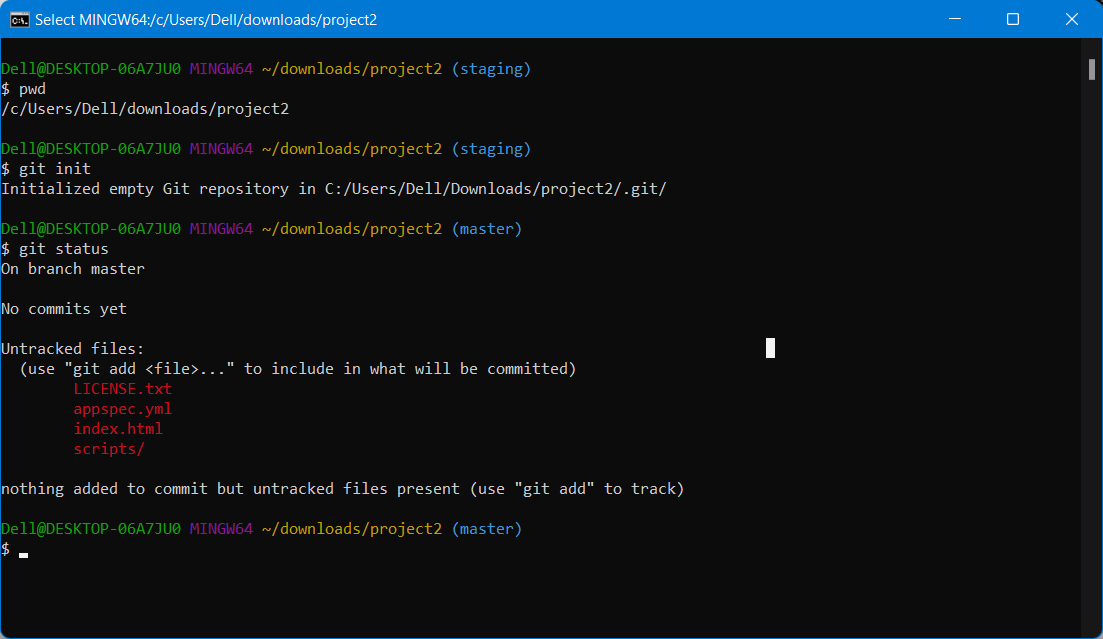
STEP 3: To verify the path of local repo execute the "pwd" in git CLI.



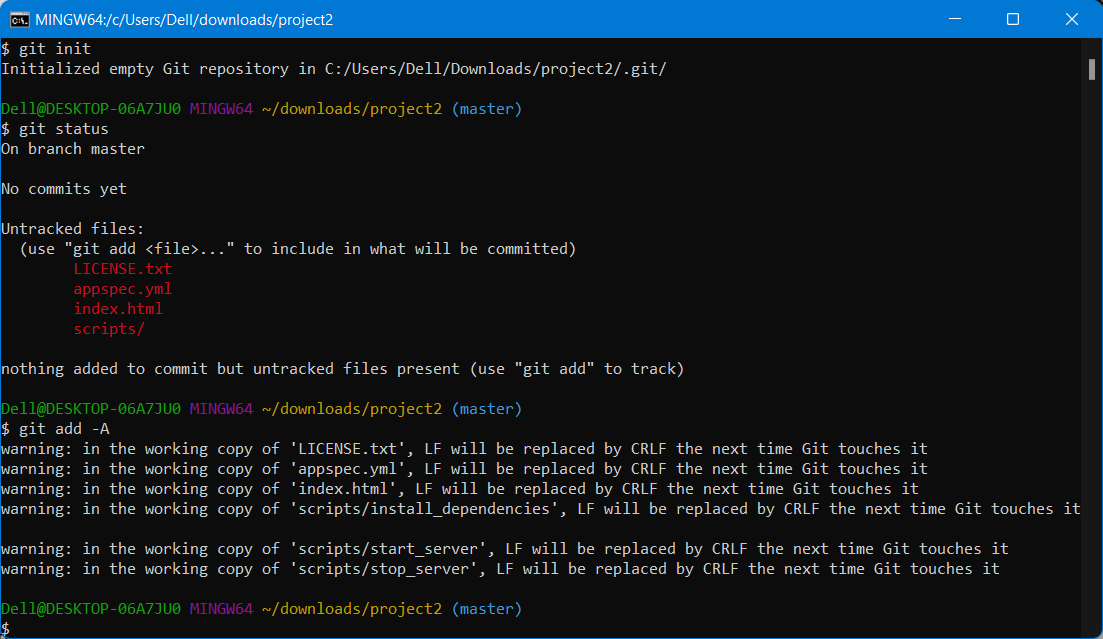
STEP 4: Next execute the "git init" command to initialize the current directory.



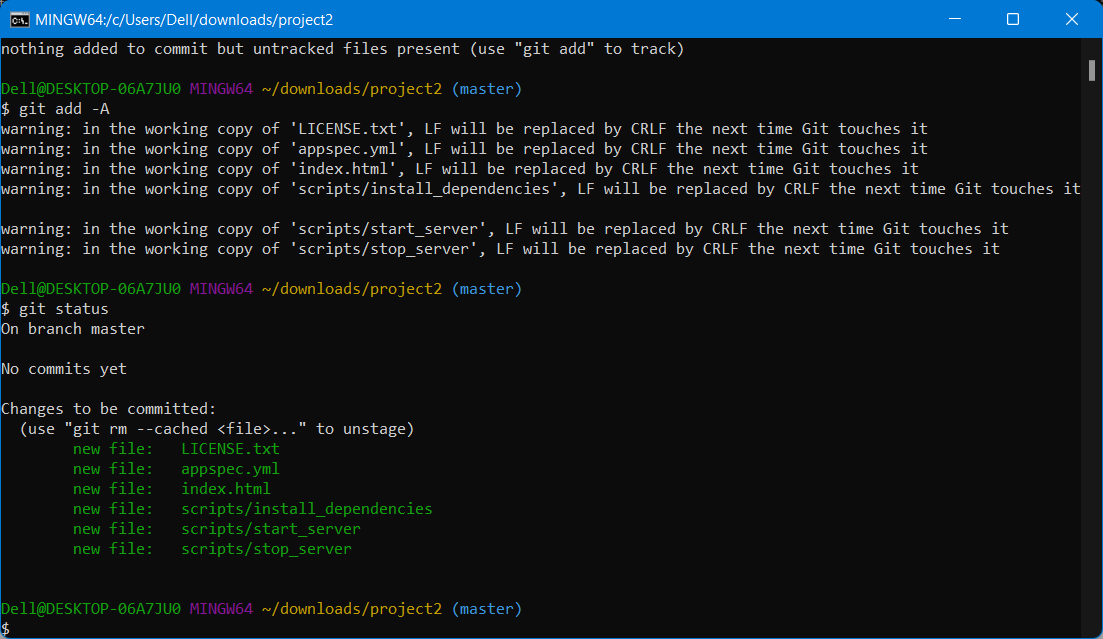
TEP 5: Next type "git status" cmd to check the files inside directory are tracked or untracked state.



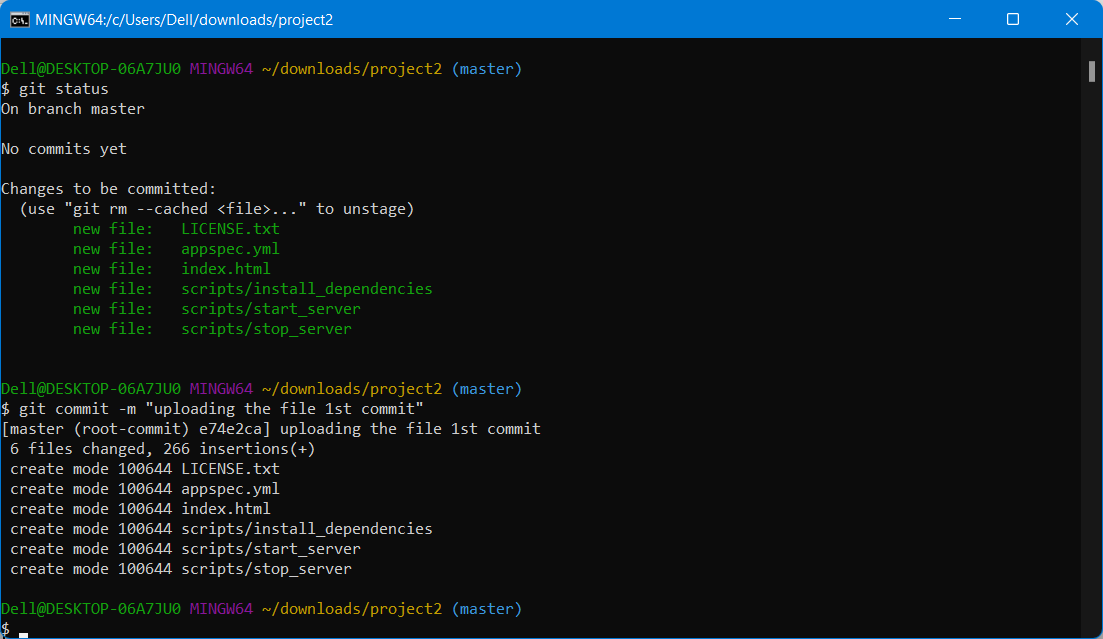
STEP 6: Now execute "git add -A" to make the files into tracked state.



STEP 7: check the status by typing "git status"

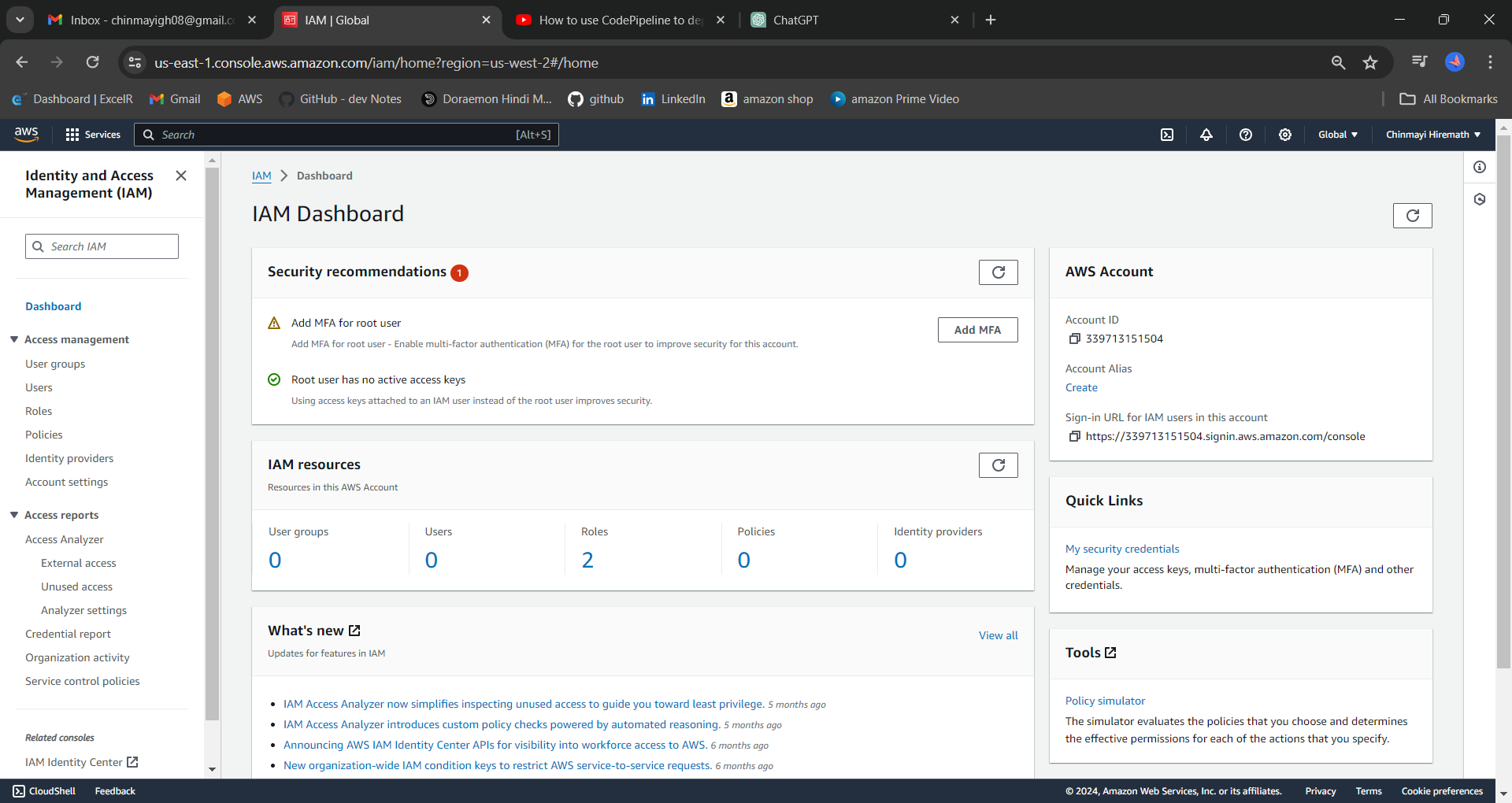


STEP 8: Now files are tracked but not committed so lets commit it by executing the git command git commit -m "commit messg"

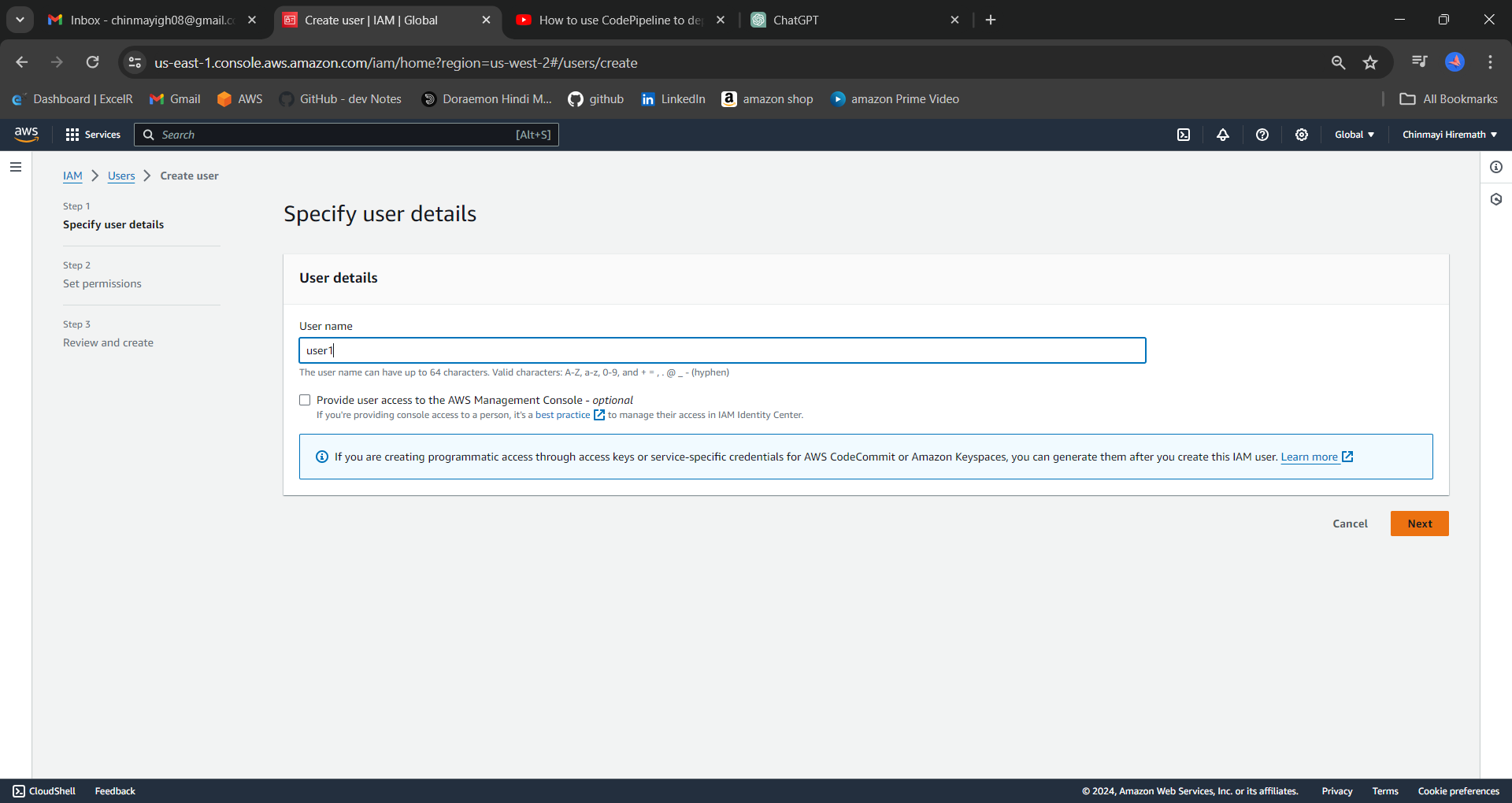


PART 2

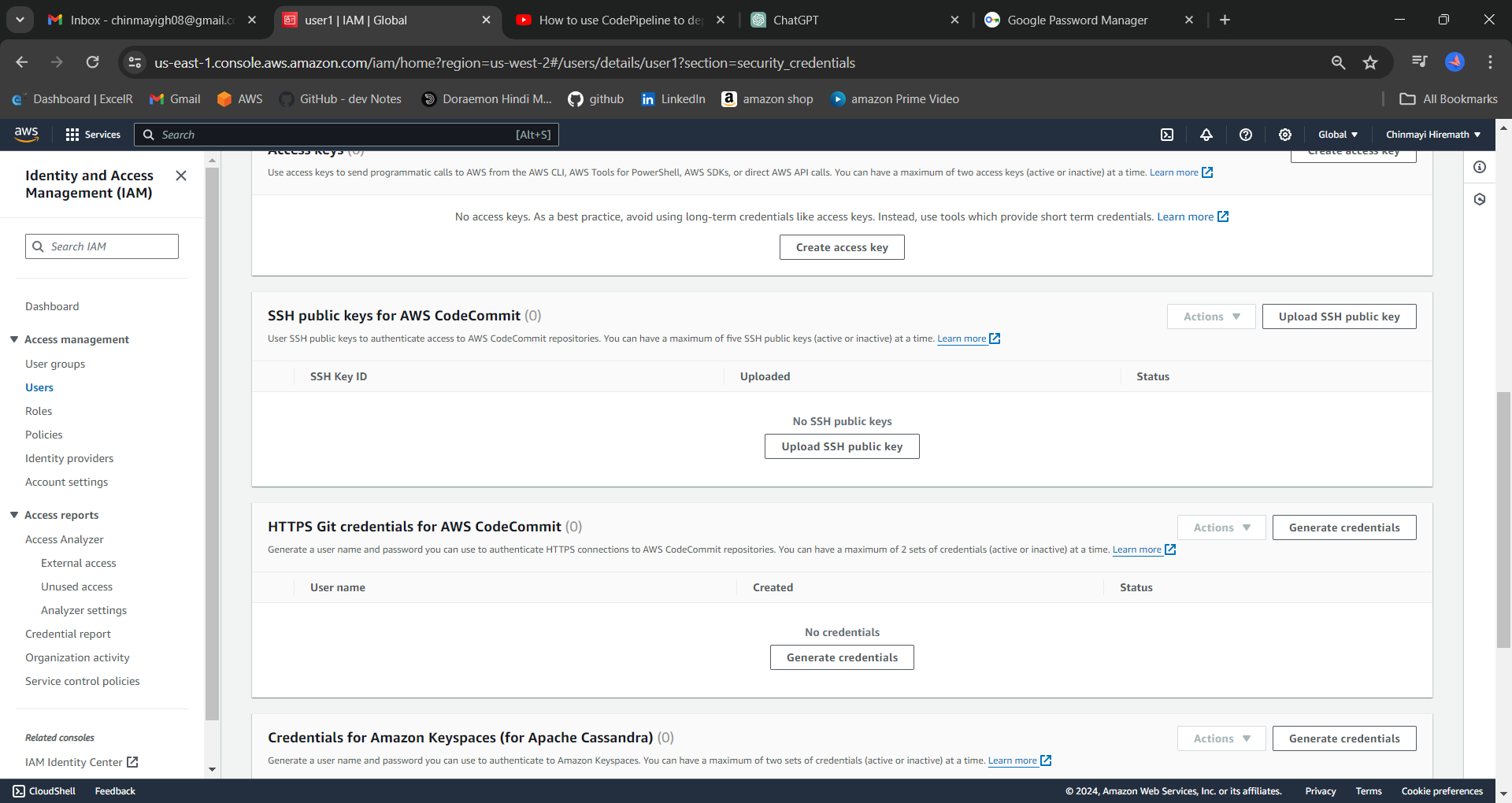
STEP 9: Now login to AWS console and open IAM service

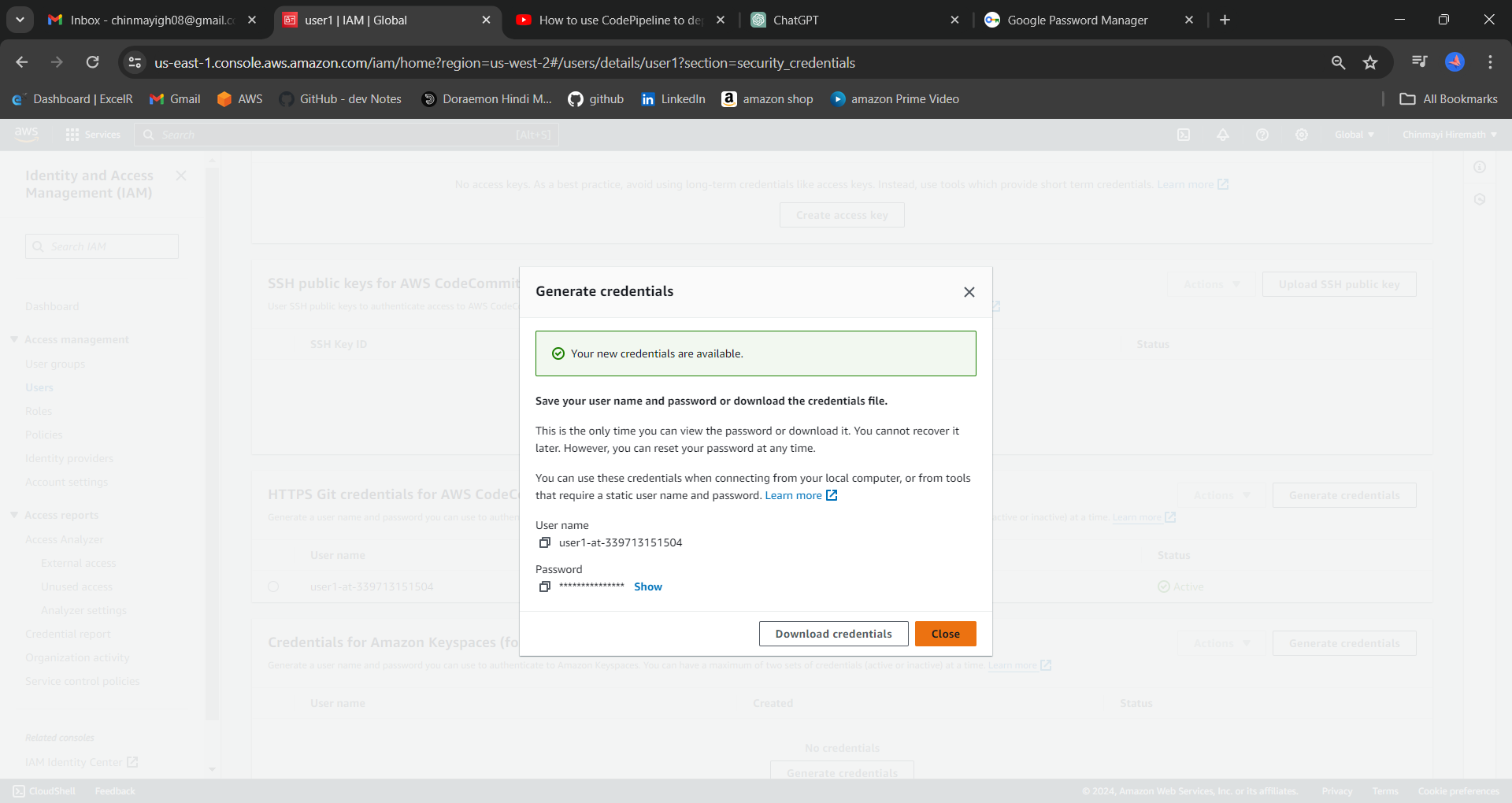


STEP 10: Create user and goup of IAM.

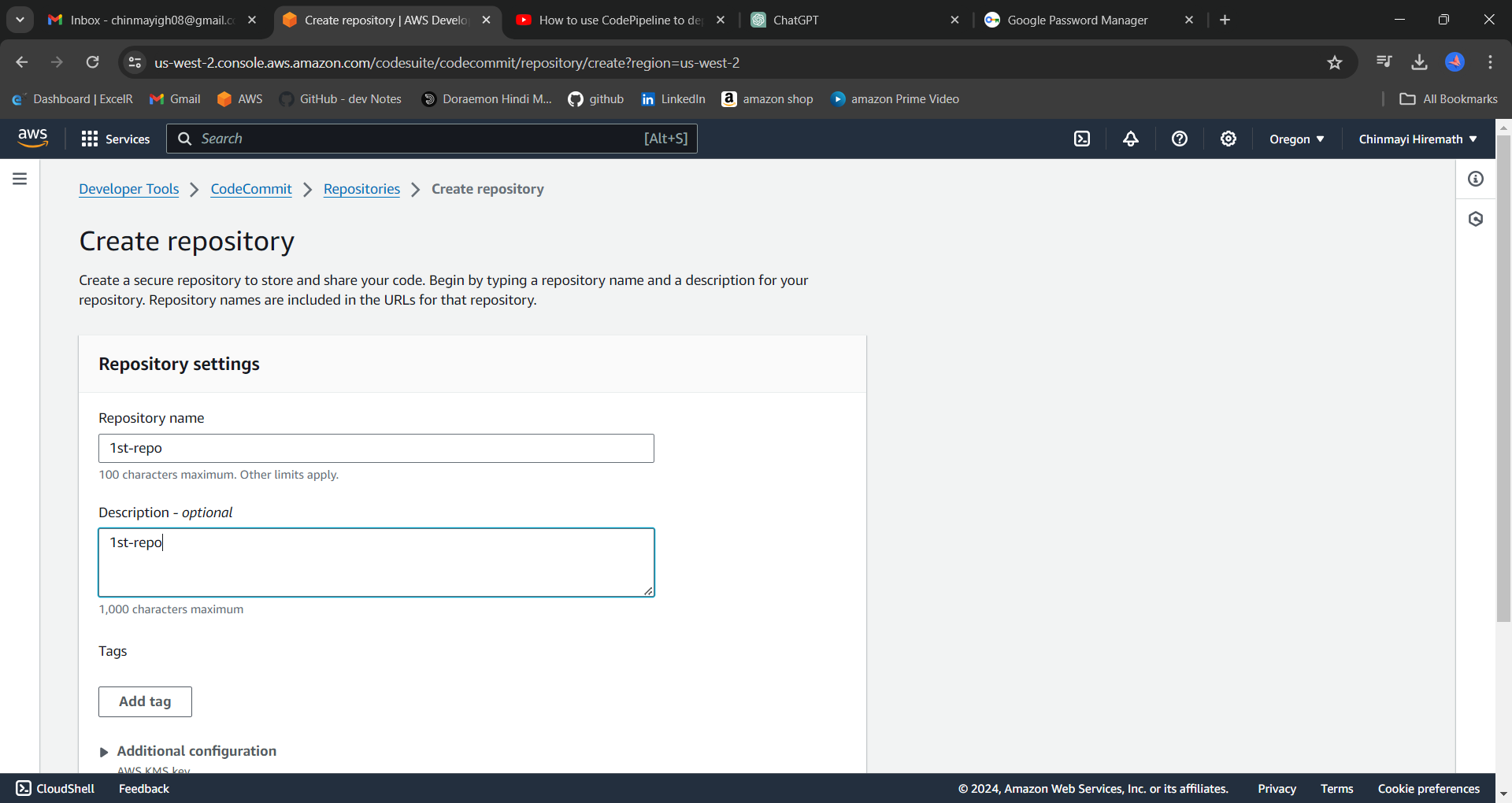


STEP 11: Now go to user "security credentials" to get the https git credentials

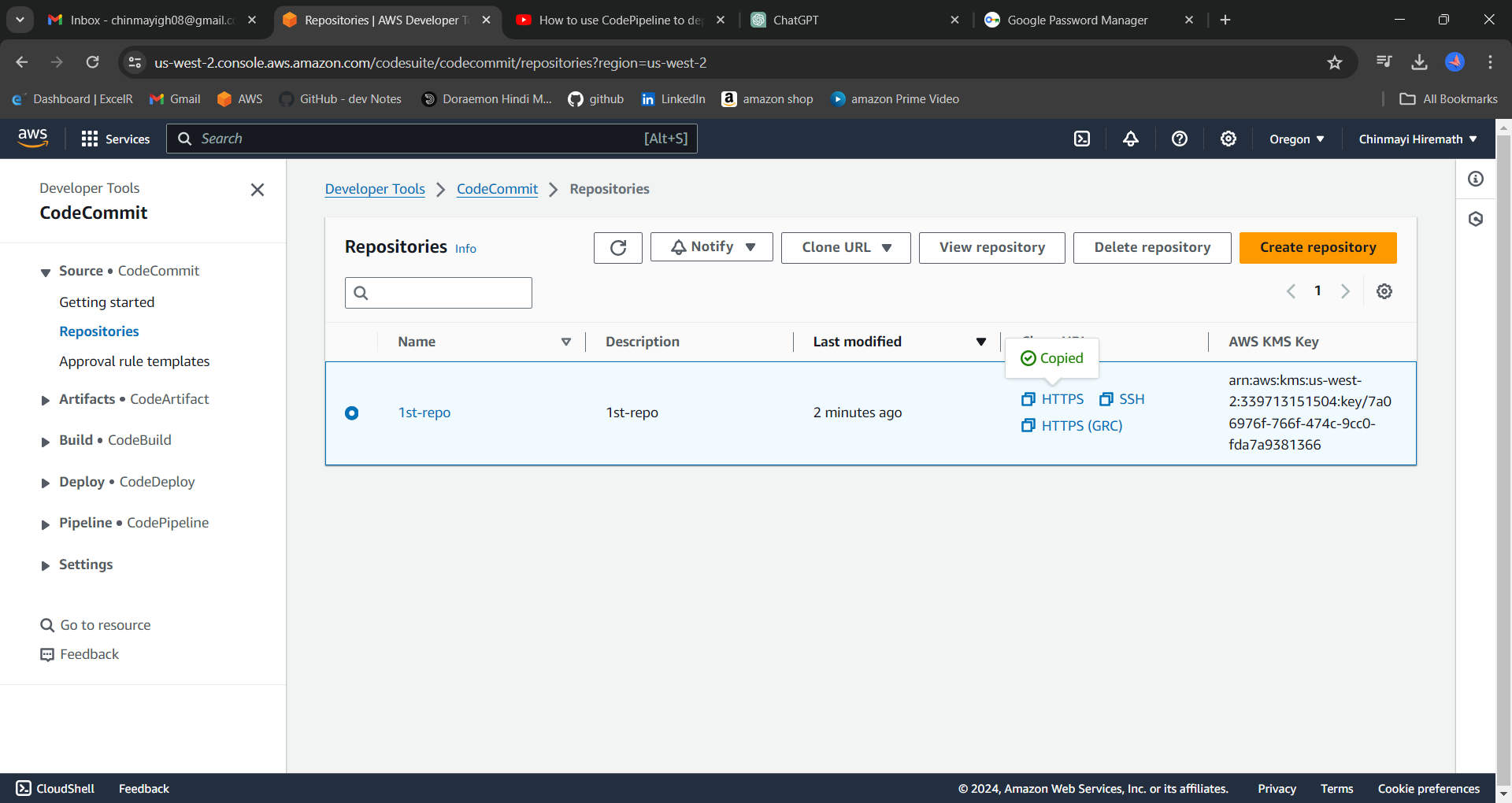




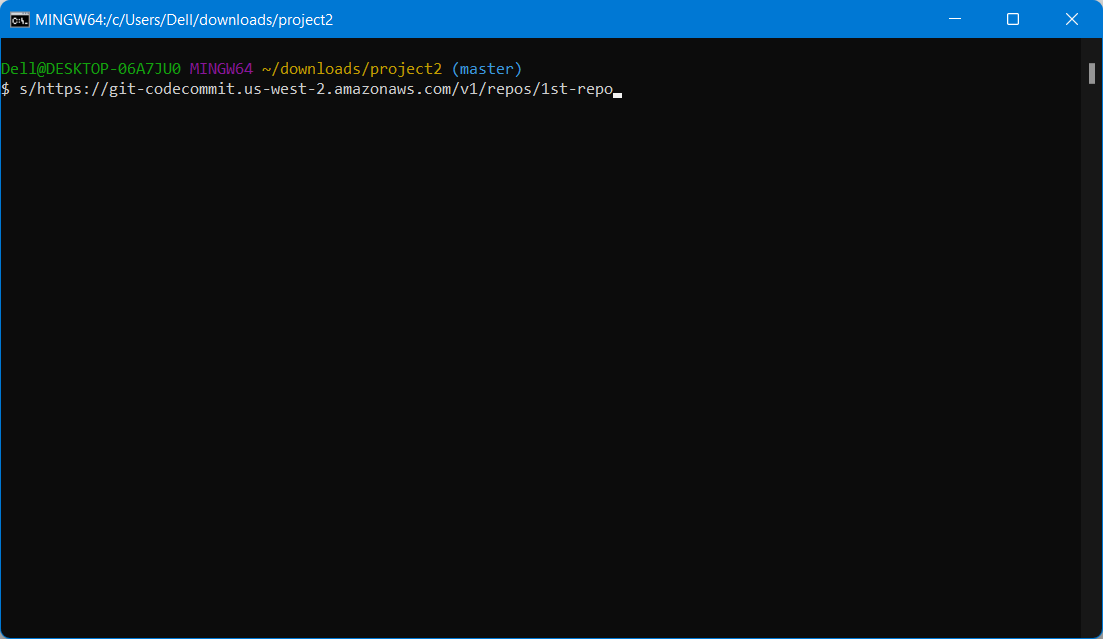
STEP 12: Go to code commit and create the new repository.

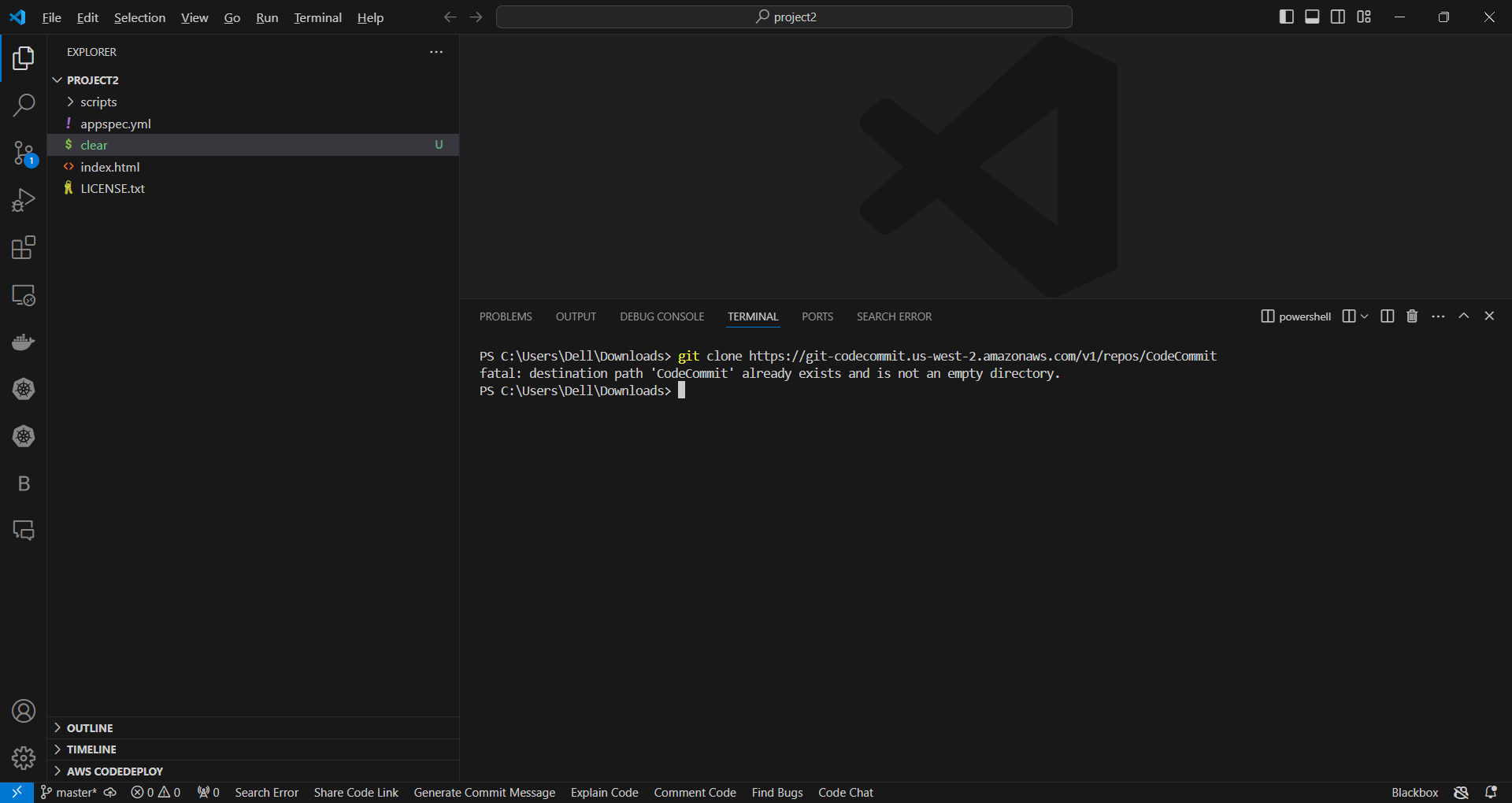


Now click on clone url and copy this url and paste this inside git bash console.



STEP 13: Inside git bash console execute the "git clone repoURL" cmd



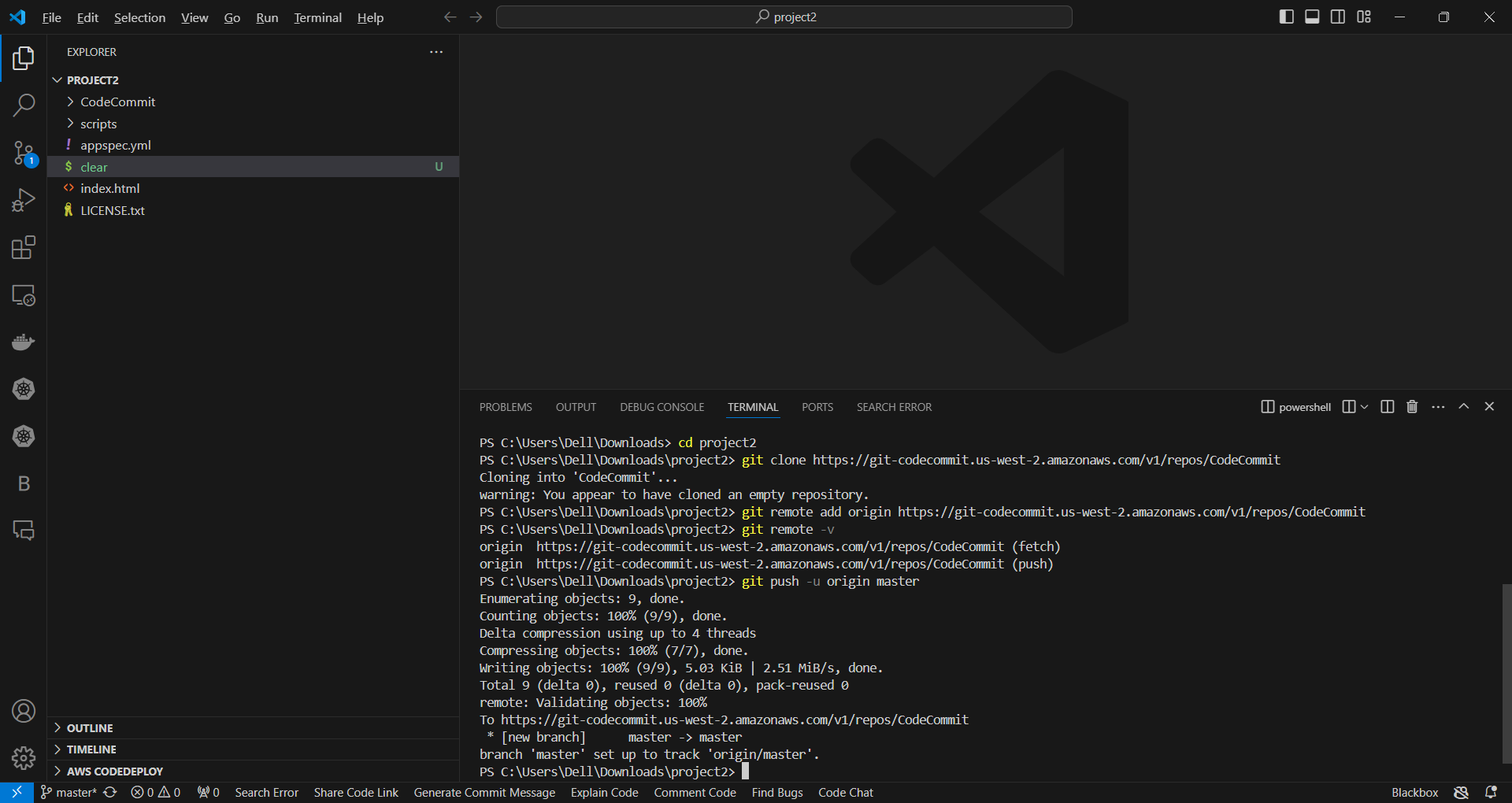


STEP 14: Now in git bash CLI please execute these 3 commands.

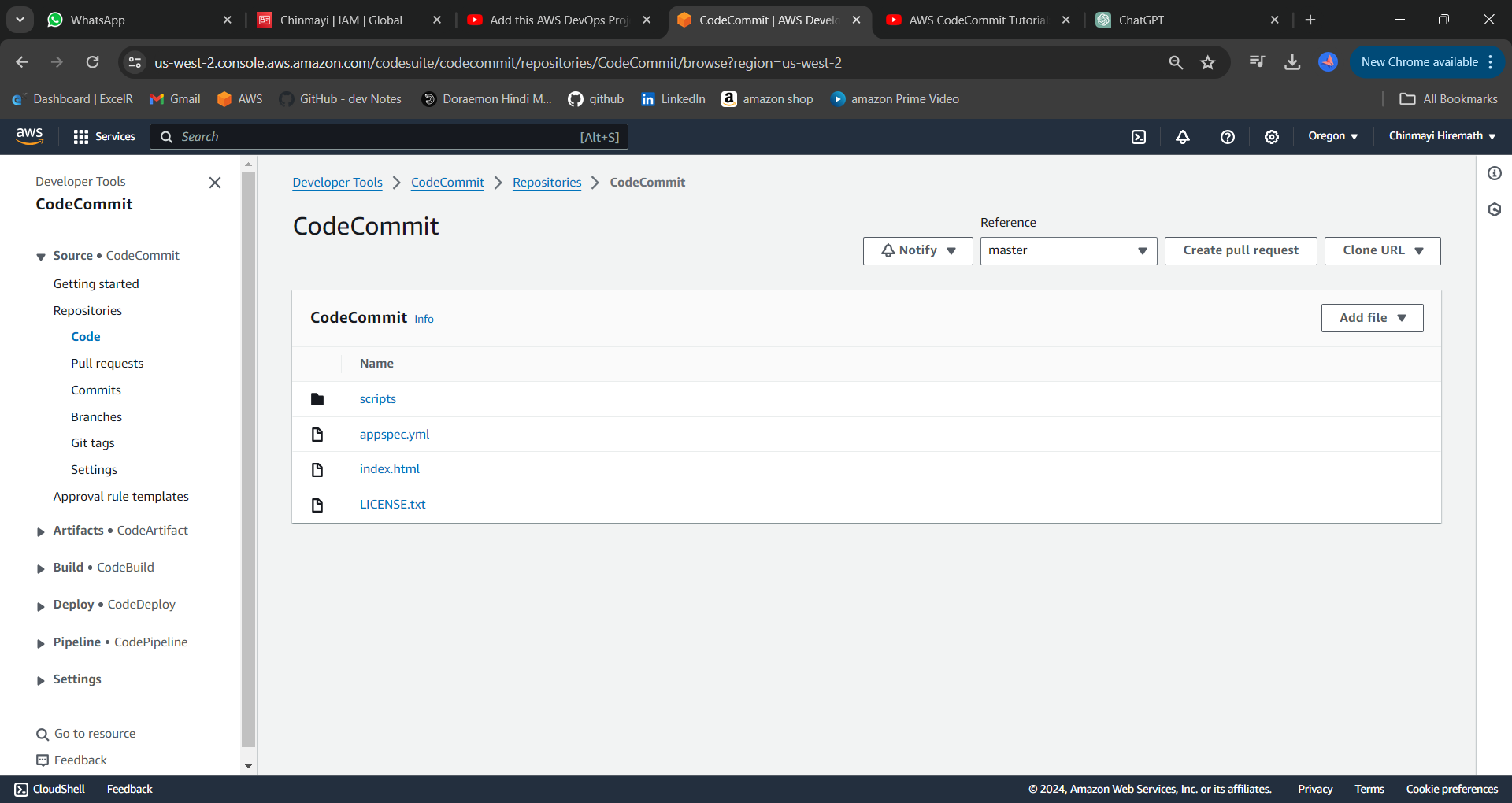
# git remote add origin repourl

# git remote -v

# git push -u origin master



STEP 15: Go inside the remote repository and check all the files have been pushed.



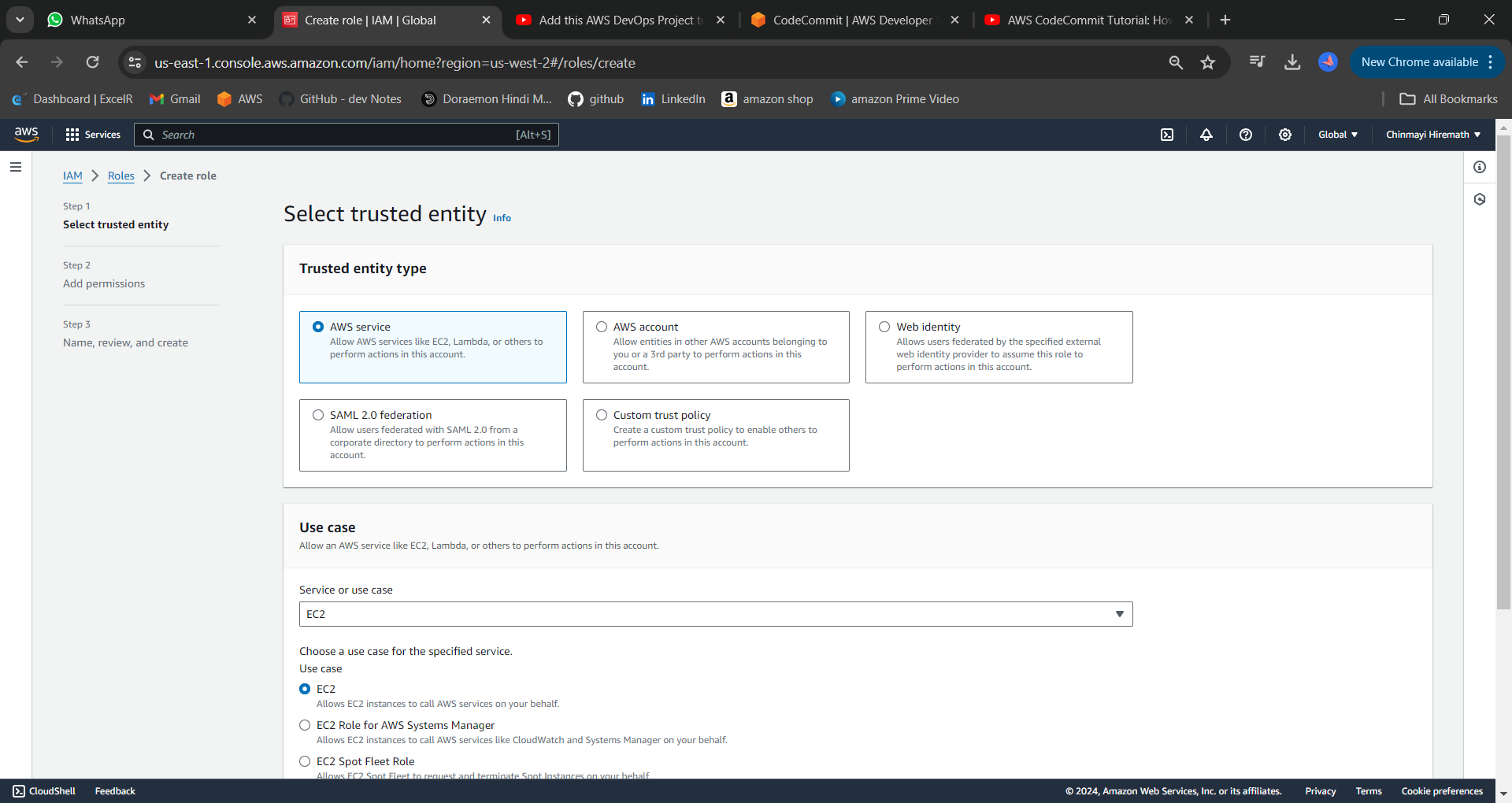
PART 3:

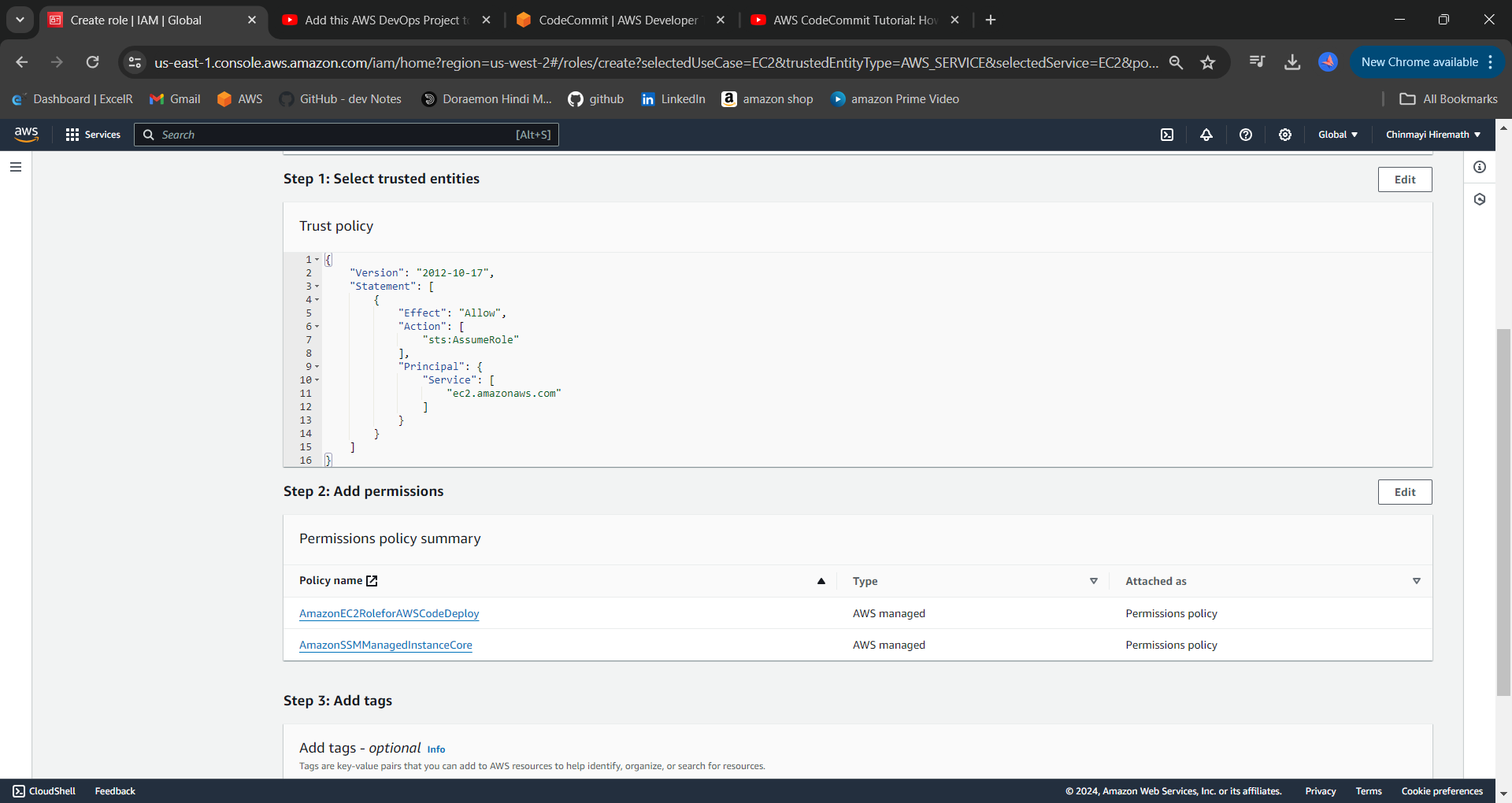
Create an Amazon EC2 Linux instance and install the CodeDeploy agent.

STEP 16: Now navigate to IAM service in order to create the roles for EC2 service.

Create Roles in that Chinmayi User. Name it EC2instancerole and give it permission of

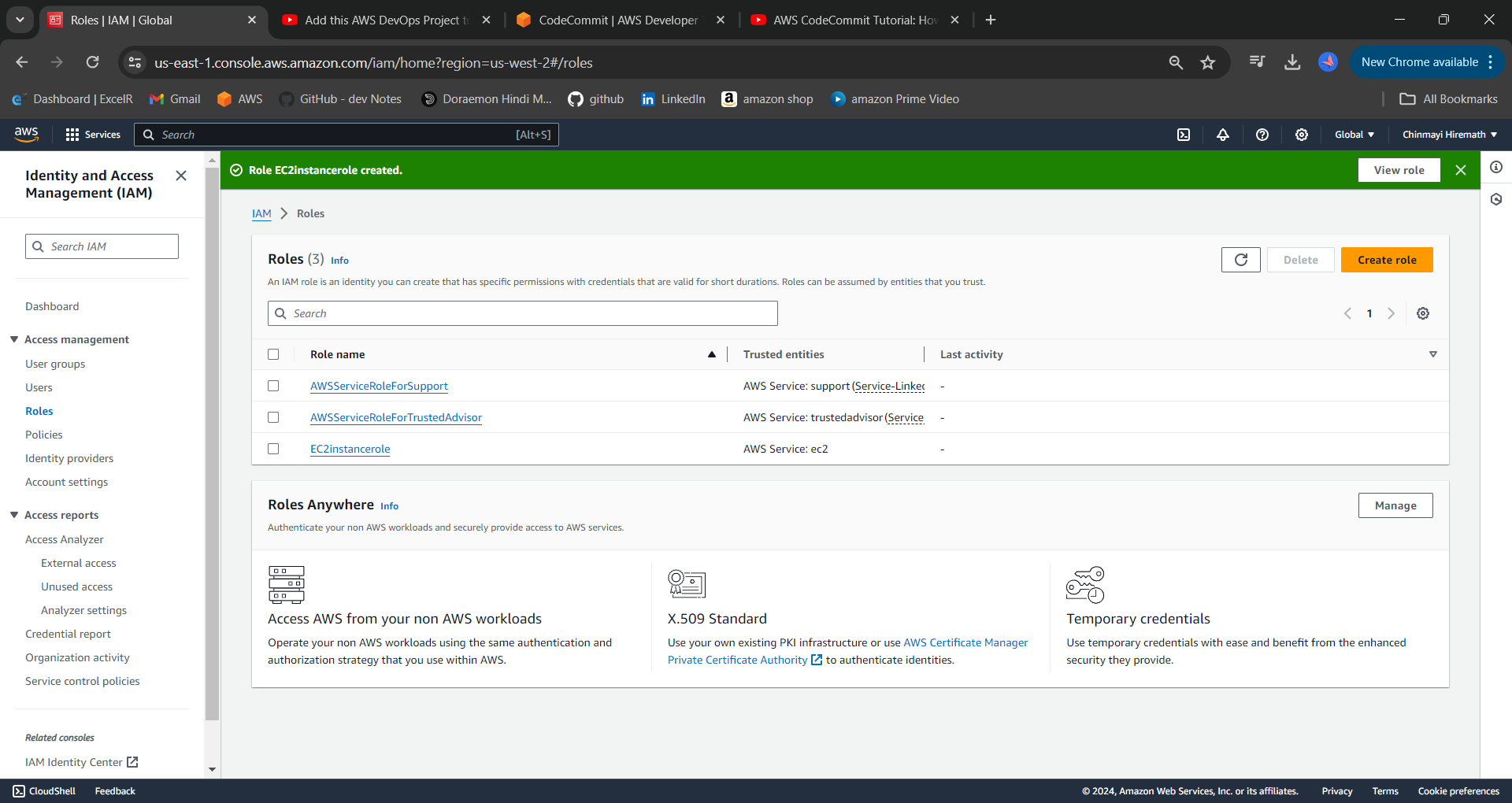
AmazonEC2RoleforAWSCodeDeploy ,AmazonSSMManagedInstanceCore





This will permit to automate the task in ec2 and the codedeploy will start working automatically.

Now it’s time to create an instance and use that role to permit it so that codedeploy can automatically deploy all the code.



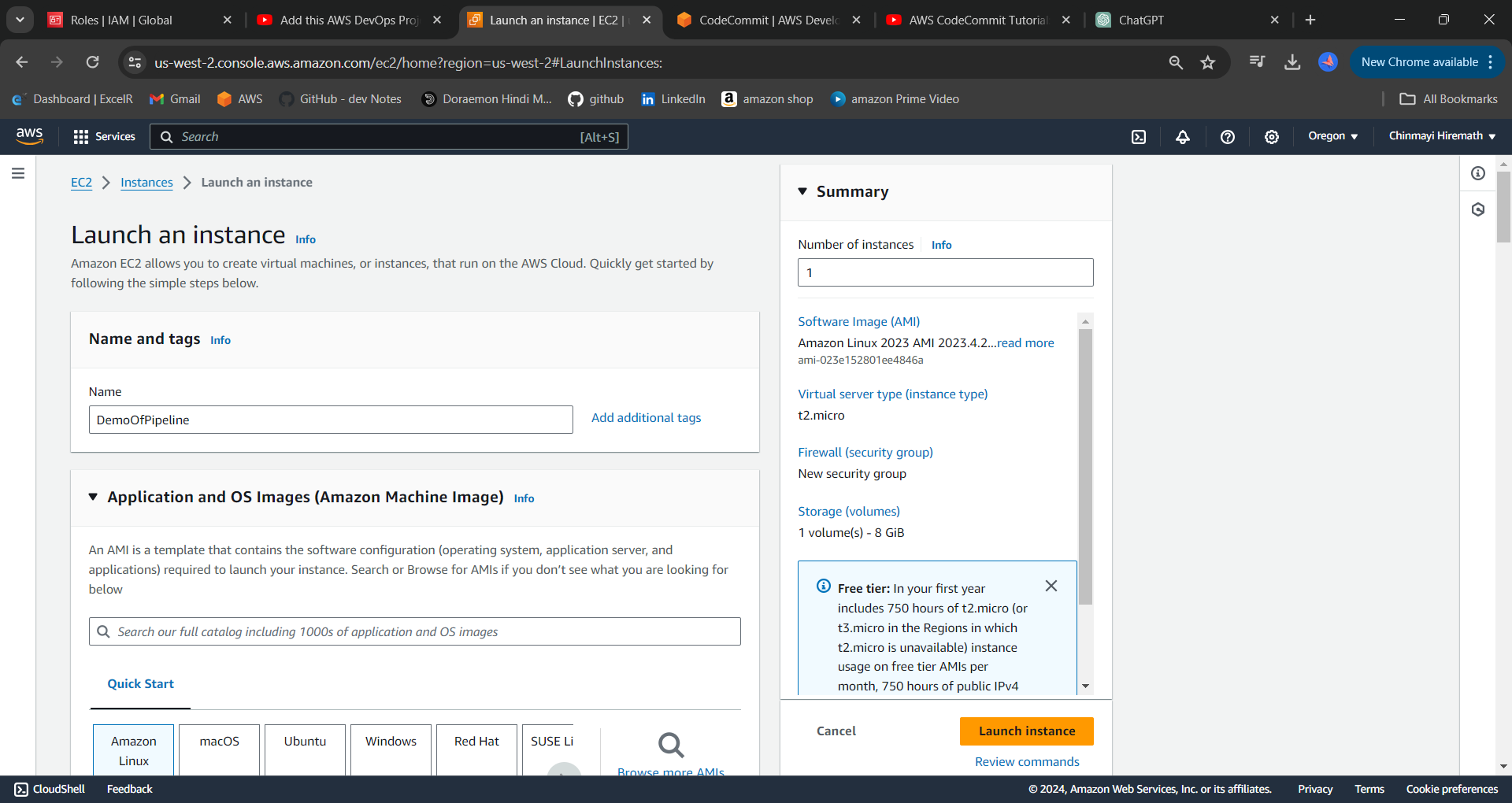
STEP 17: Next navigate to EC2 service in order to launch the linux server and there add the role in the advanced settings which has been created in IAM

Name its instance DemoOfPipeline.

Inbound rules must be:

SSH – 22

HTTP – MyIP / Anywhere

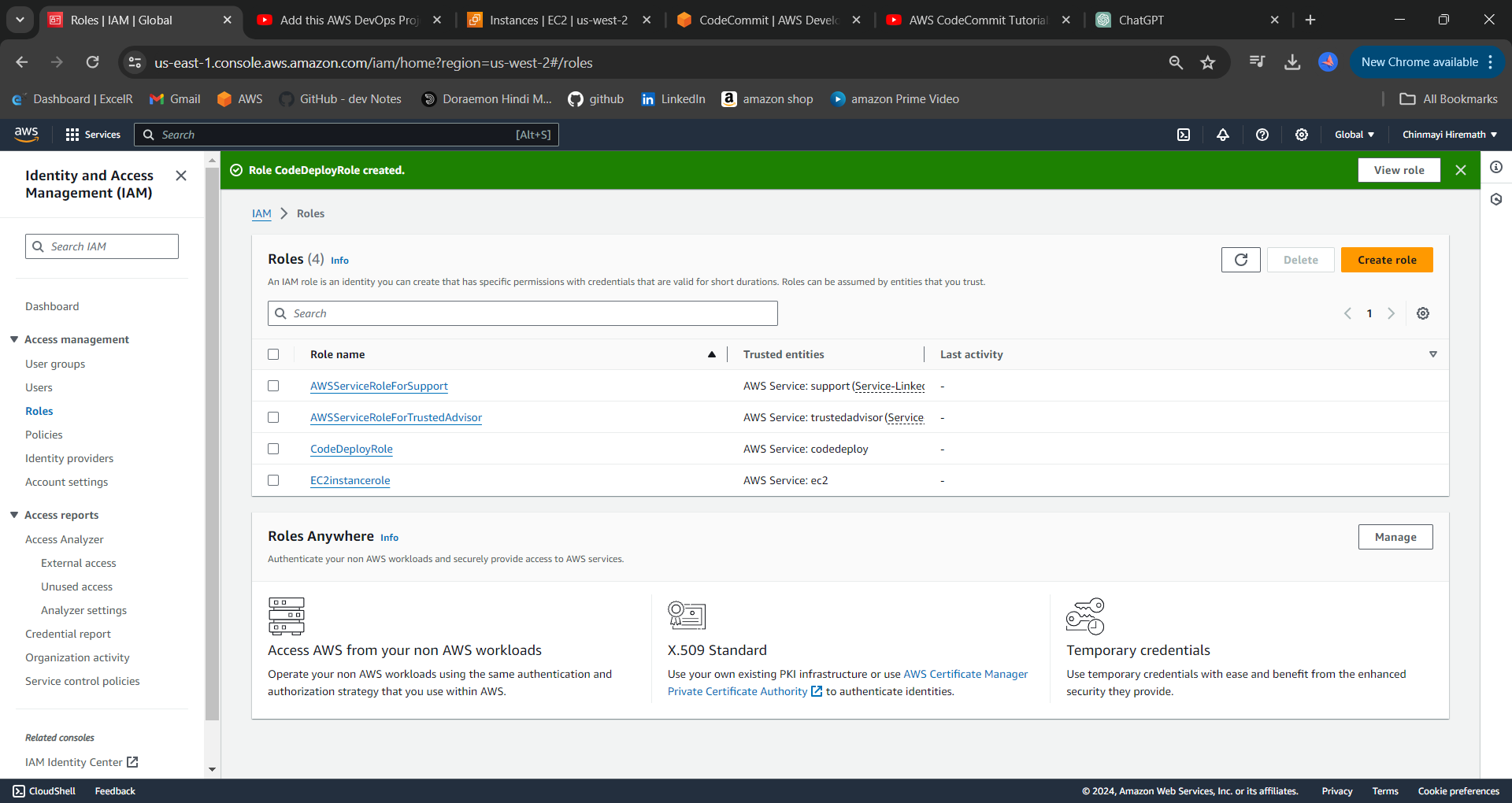


PART 4 : CODE DEPLOY SERVICE

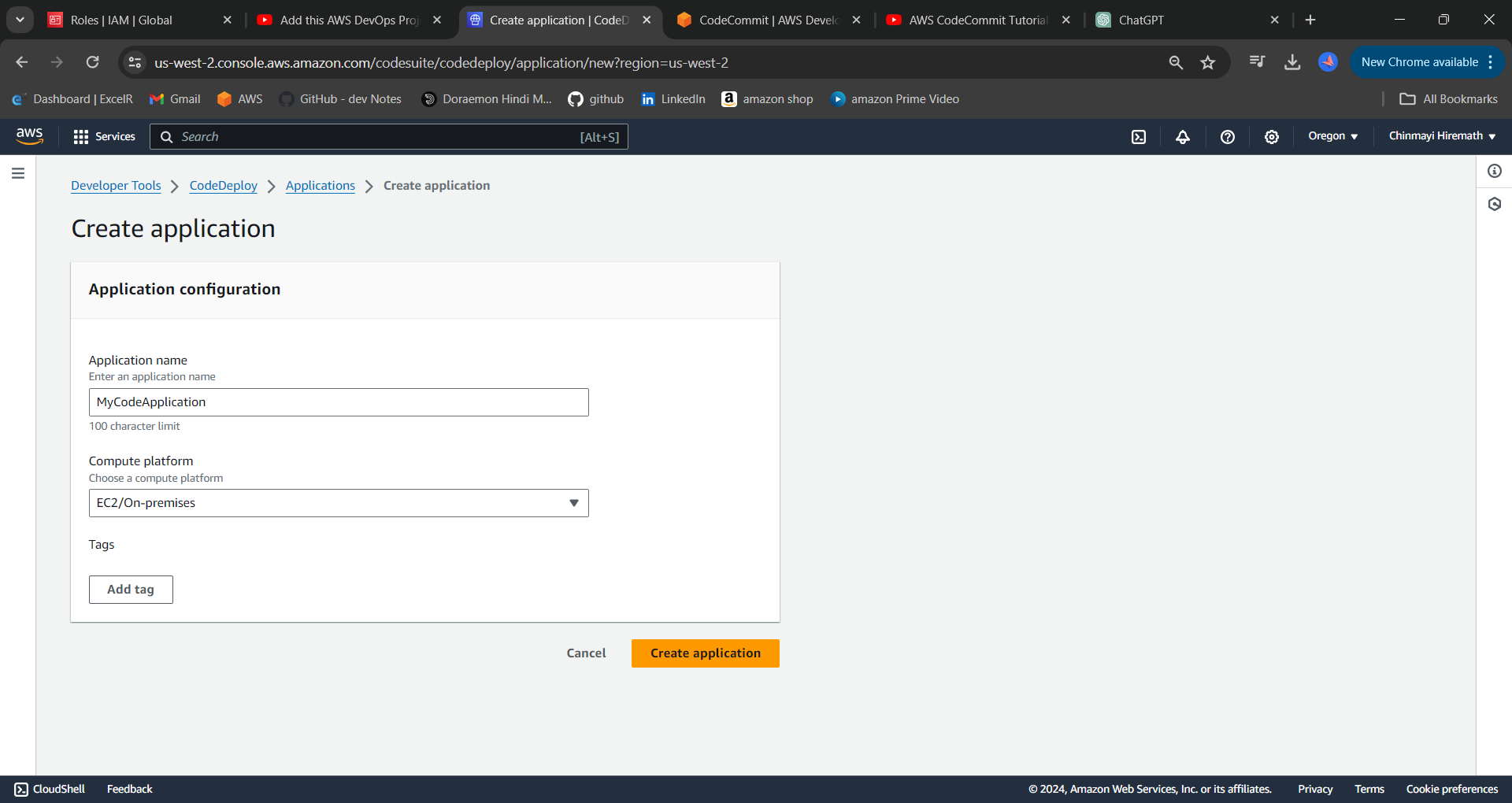
STEP 18: Now go to IAM service and create the role for code deploy service which can be used while creating the application in code deploy.

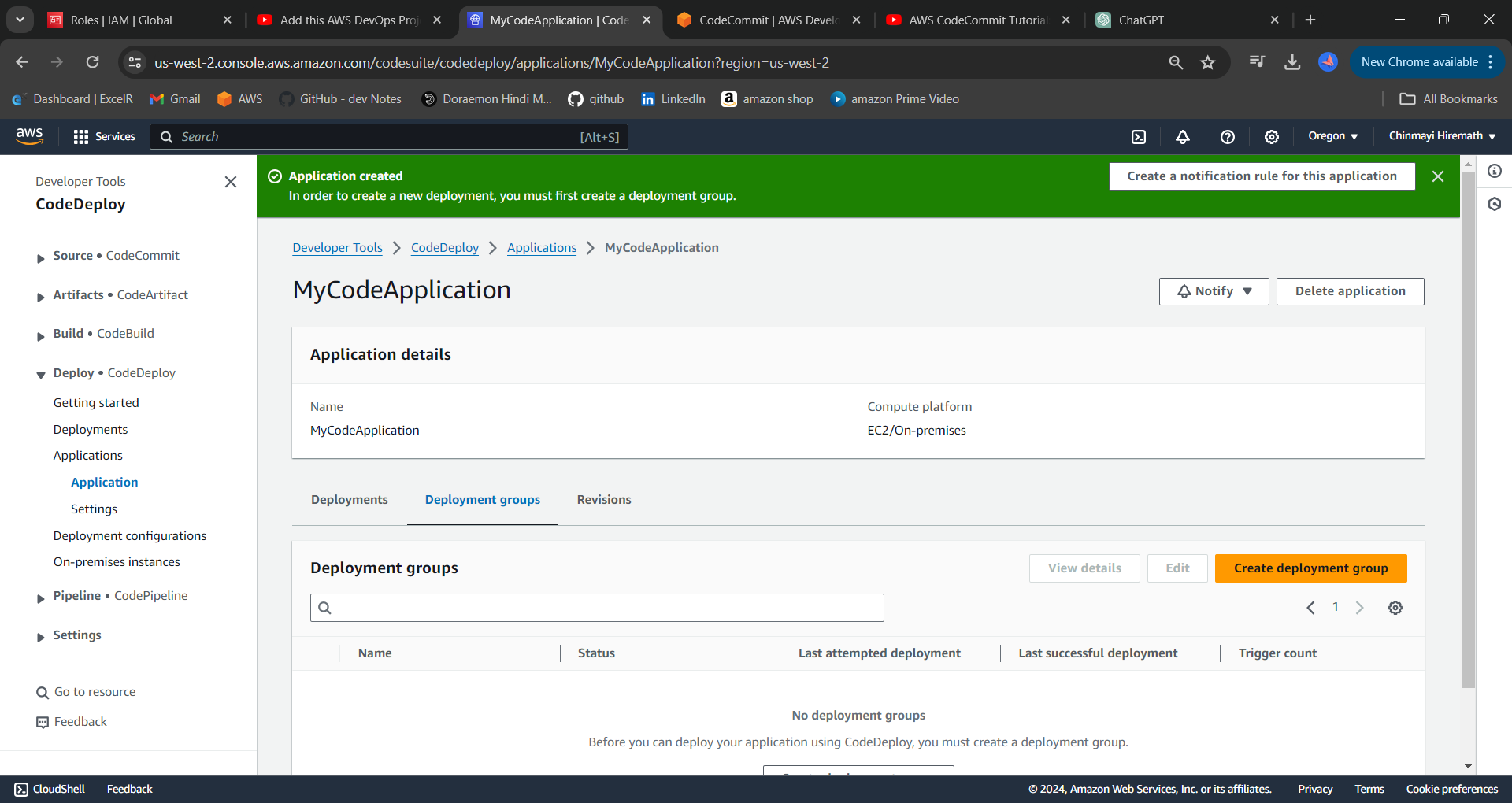
Application in codedeploy contains software that we want to deploy.

First, to access codedeploy we need to permit it by making CodeDeployRole. Give it the name CodeDeployRole.



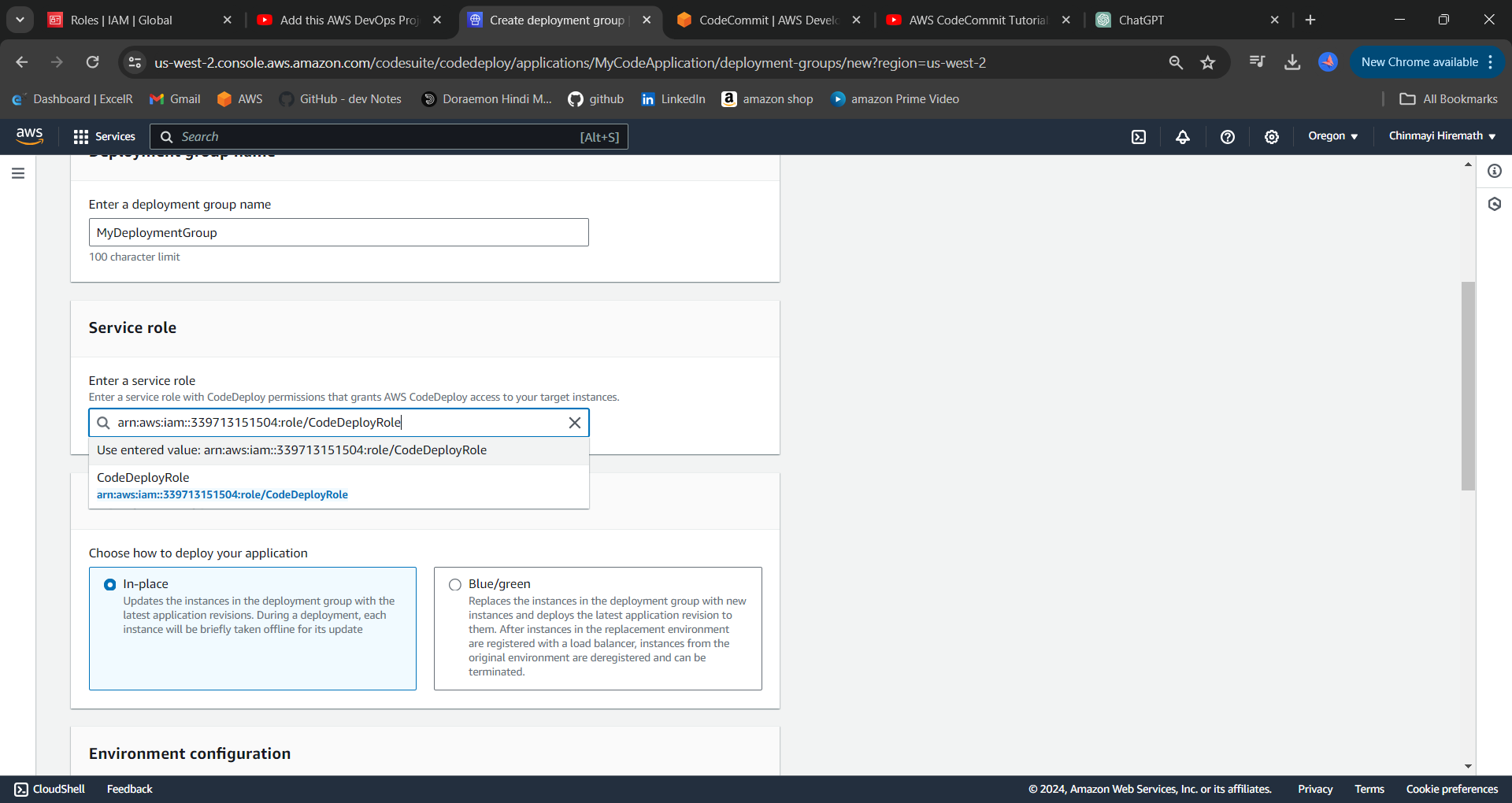
STEP 19: Now navigate to Code deploy and create the application.

Then we are going to create codedeploy Application with the name MyCodeApplication. Then we need to send him into a deployment group we will create a Deployment group with the name MyDeploymentGroup. In that deployment group we will give CodeDeployRole to them.



STEP 20: Now create the deployment group.

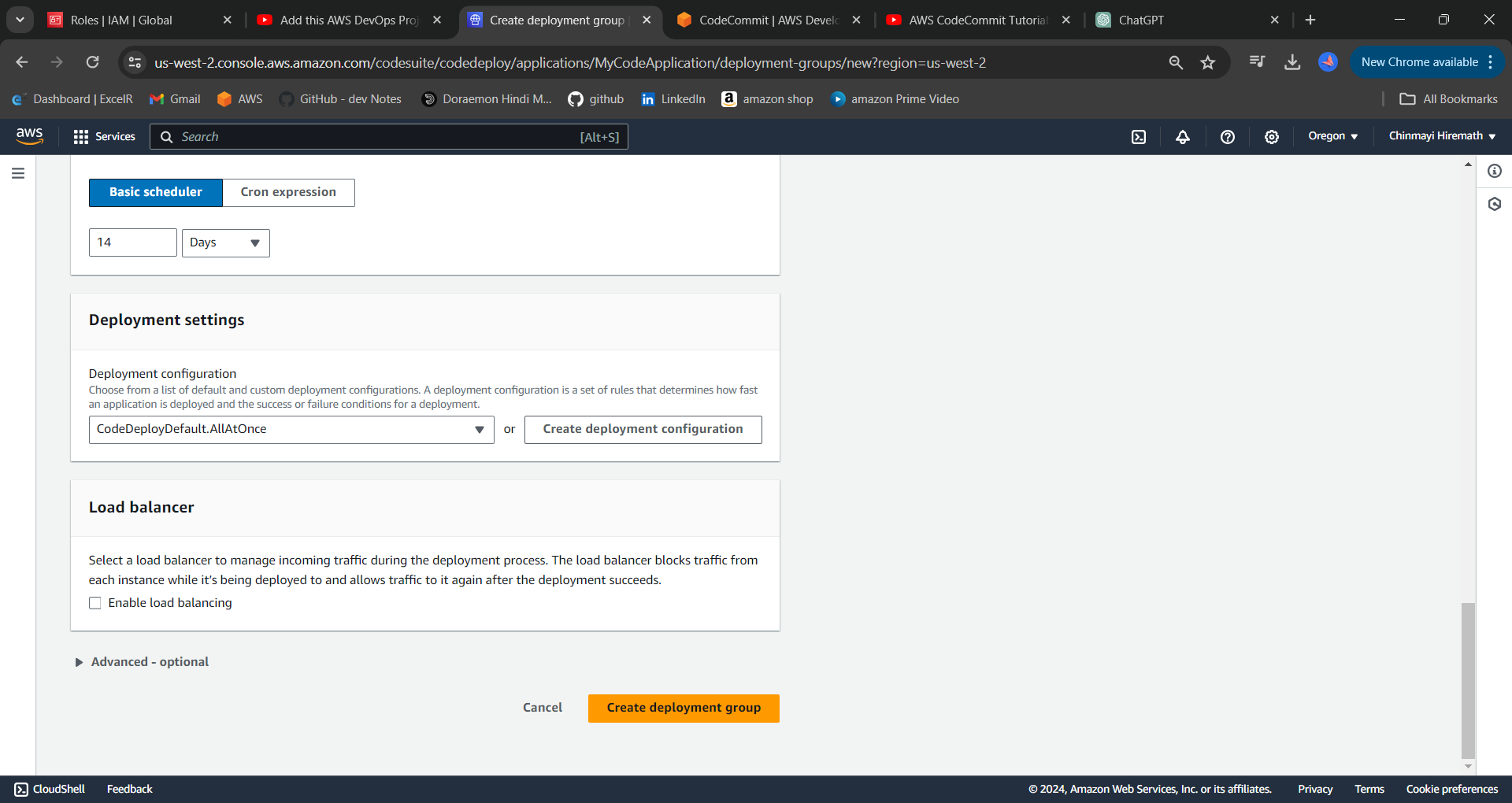
Now give a name to deployment group and select the role which was created for code deploy.



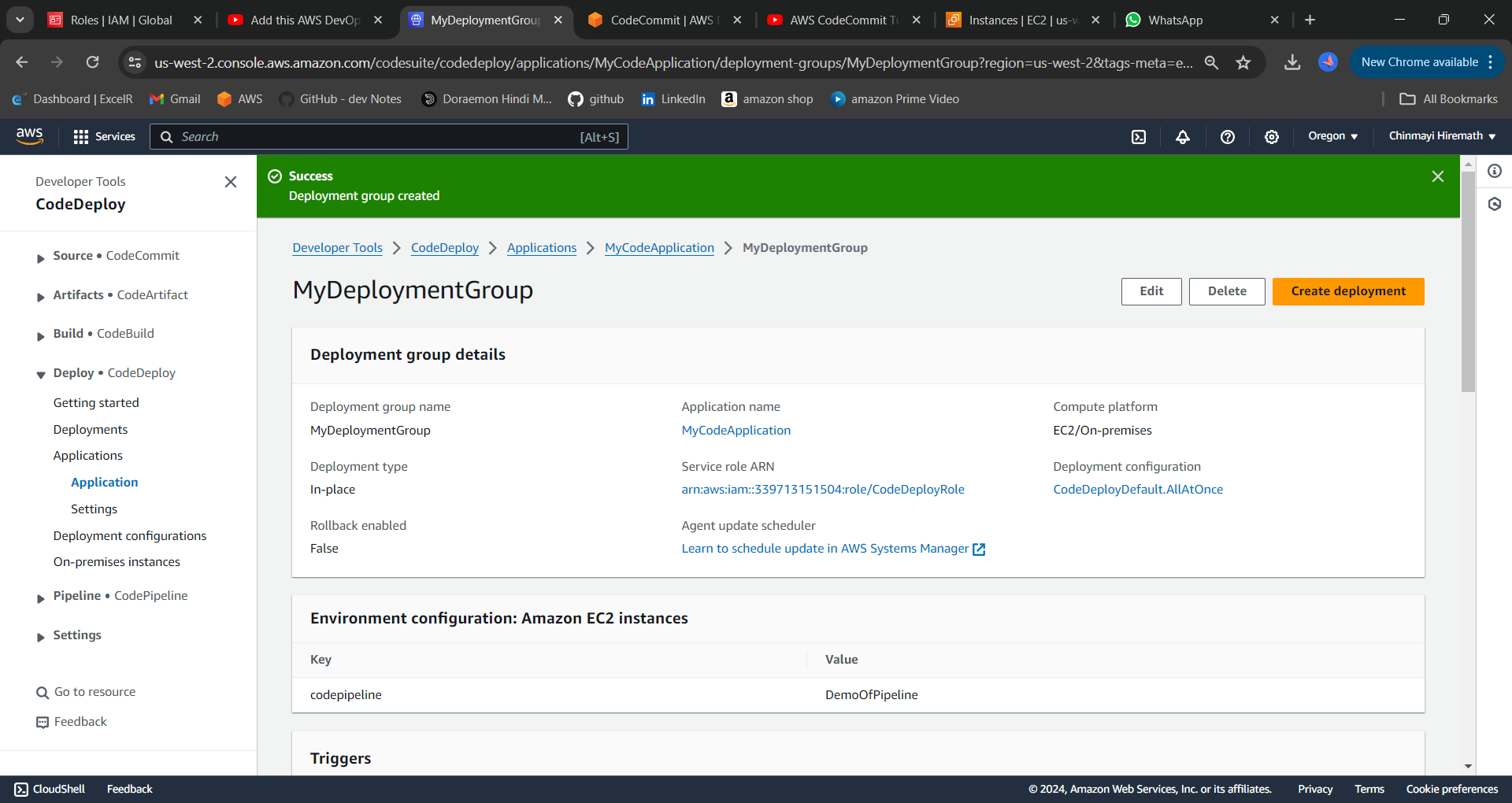
STEP 21: Now select deployment type as 'in place" and environment as "Amazon EC2Instances"



Now choose "Now and schedule updates"

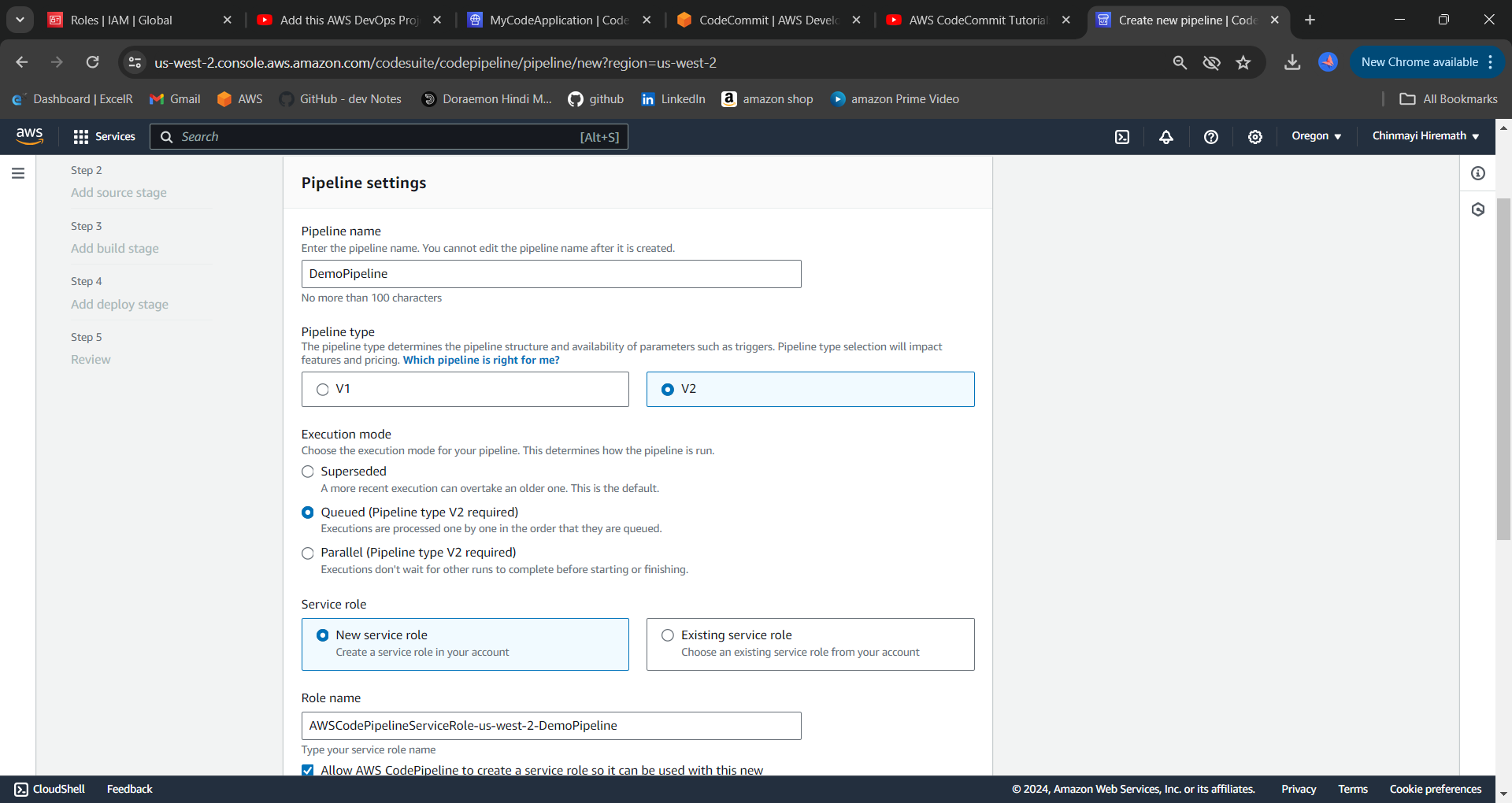


STEP 22: Now enable the load balancer option and select "create deployment group"

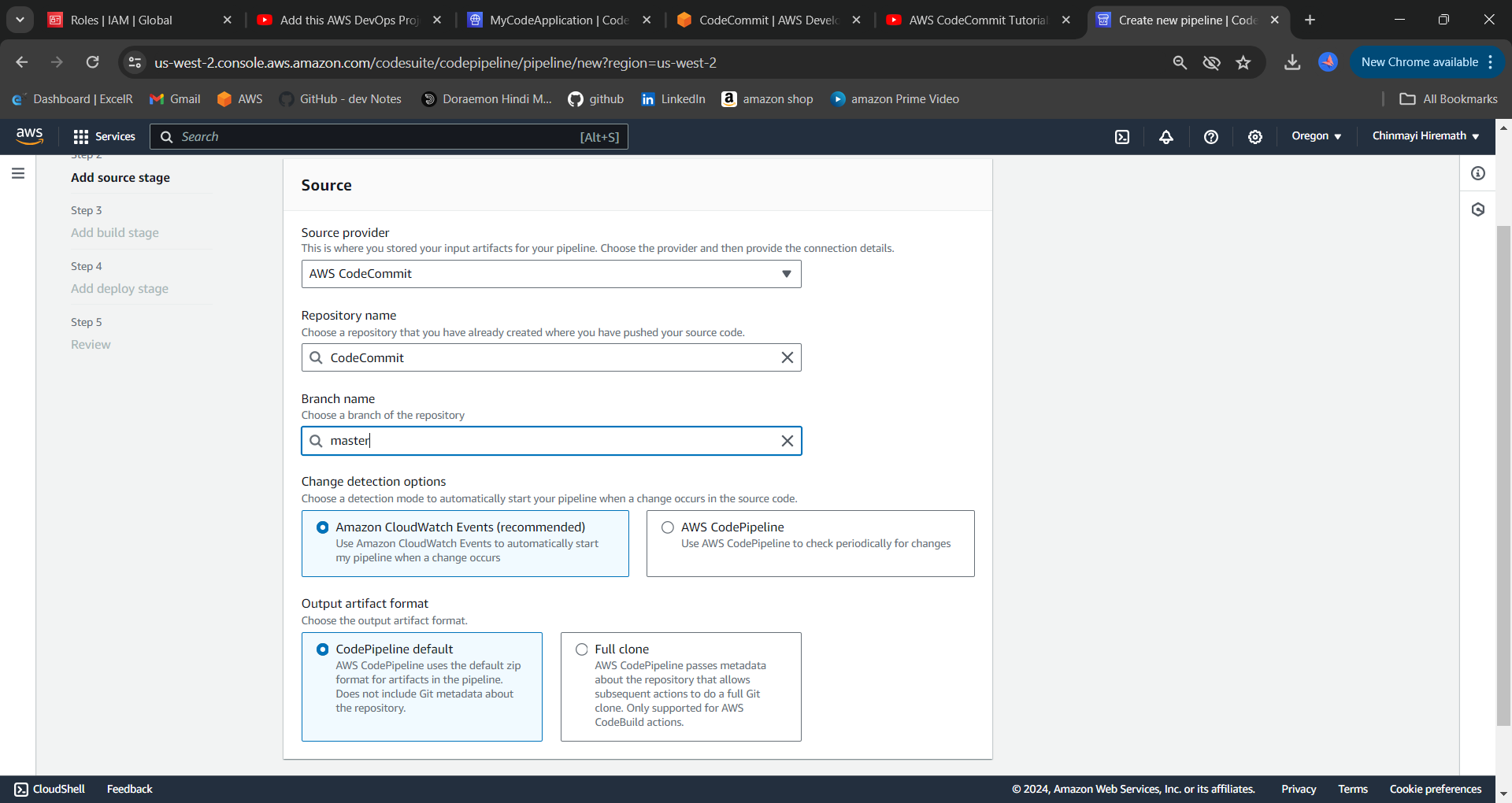


PART 5 CODE PIPELINE

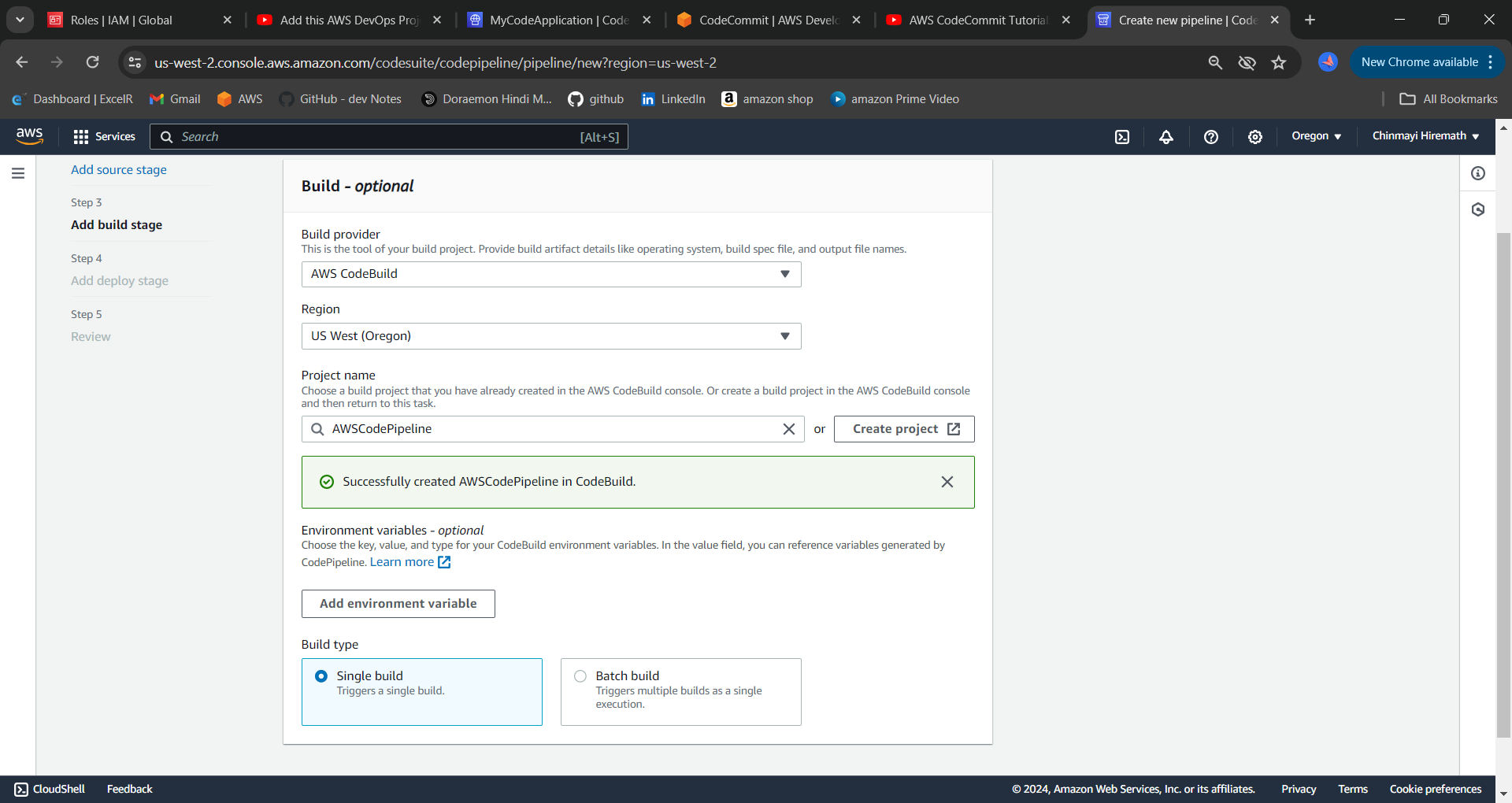
STEP 23: Now open the code pipeline service and create the pipeline and configure it.



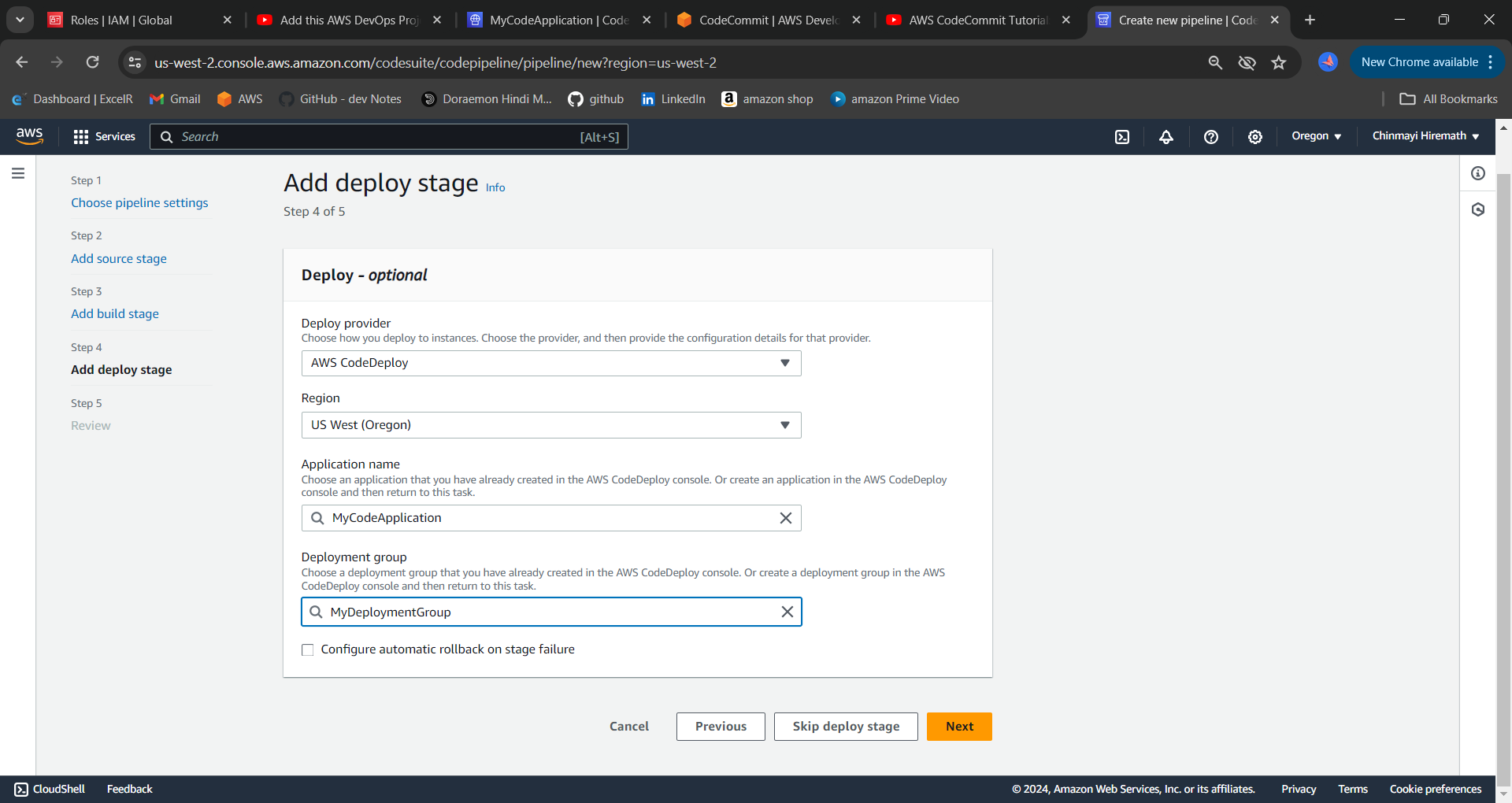
STEP 24: Now Customize as Below



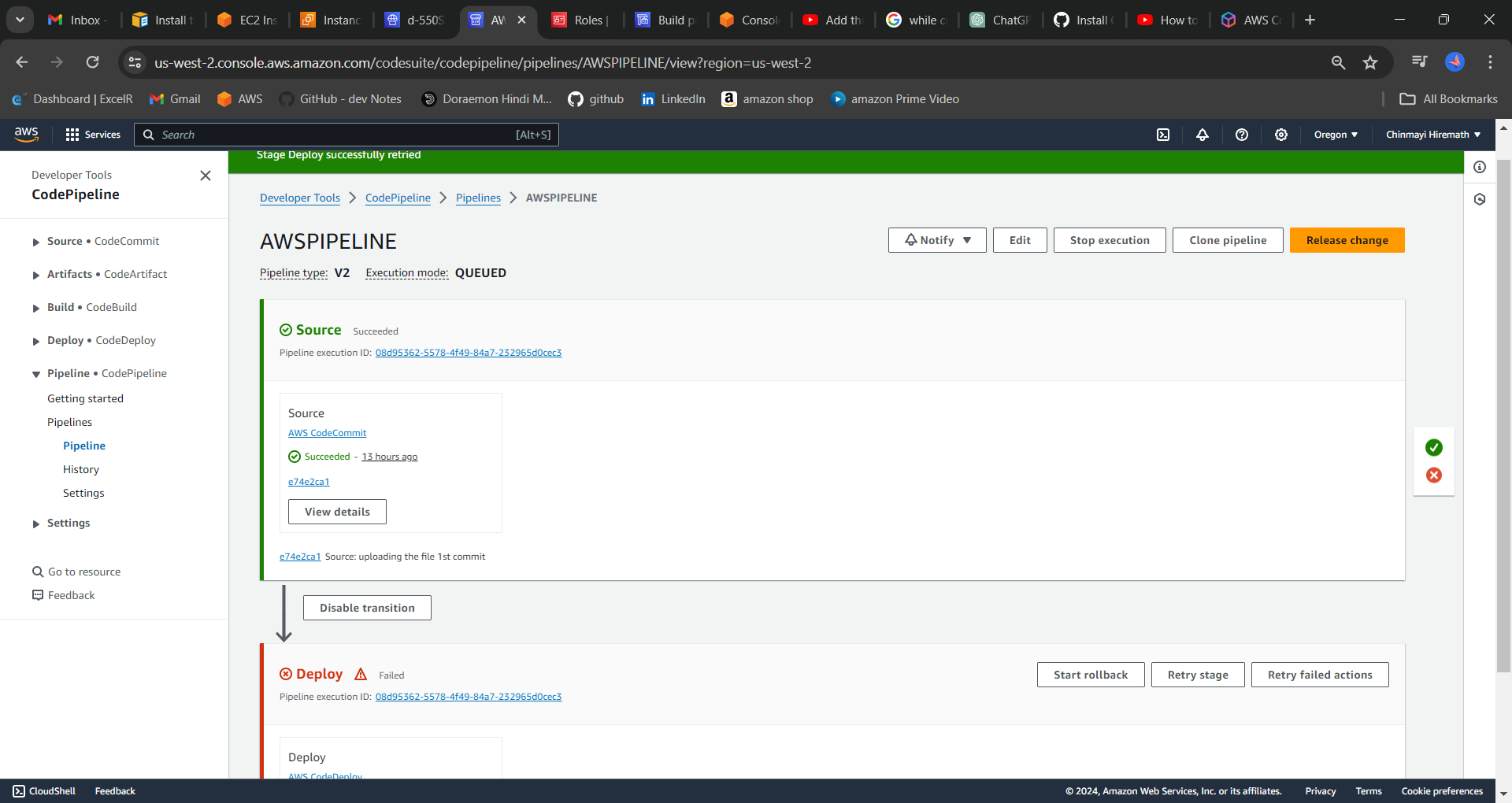
STEP 25: Next add the build step and specify the build file name as it is in the code commit repository.



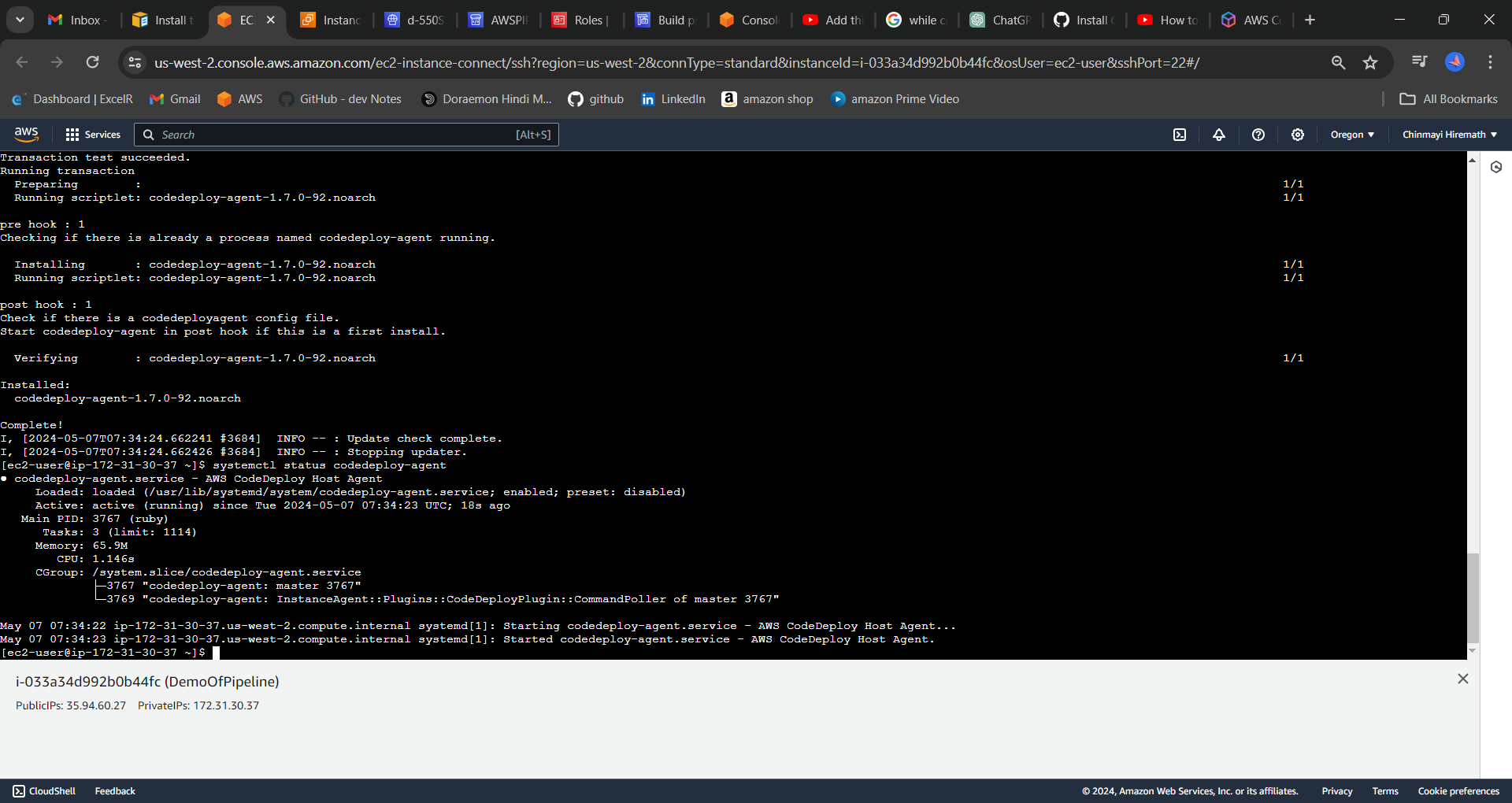
STEP 26: Now click next and it will land on deploy stage there please select codedeploy service .



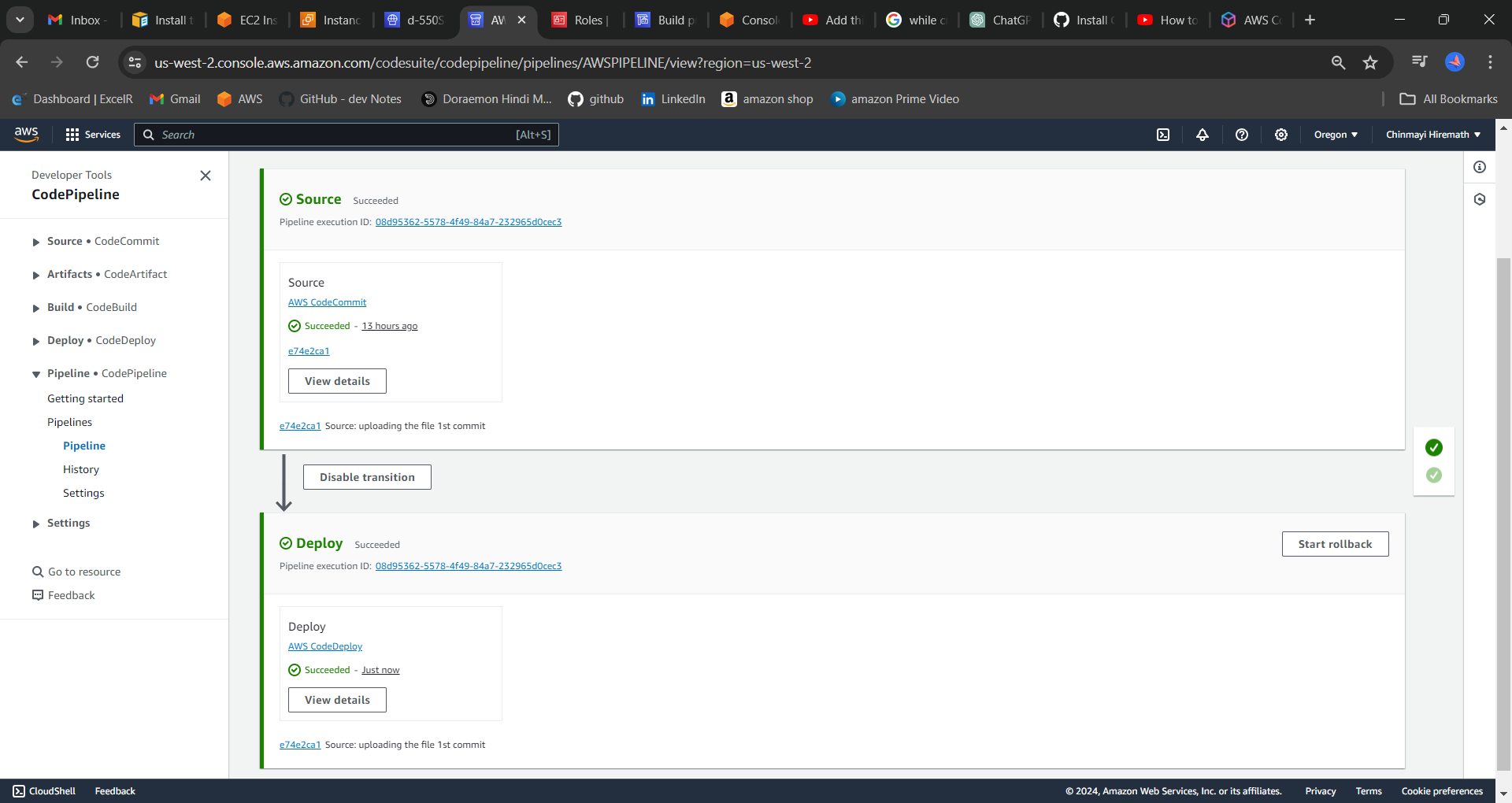
STEP 27: Now review all the stages which were configured and click on create pipeline.

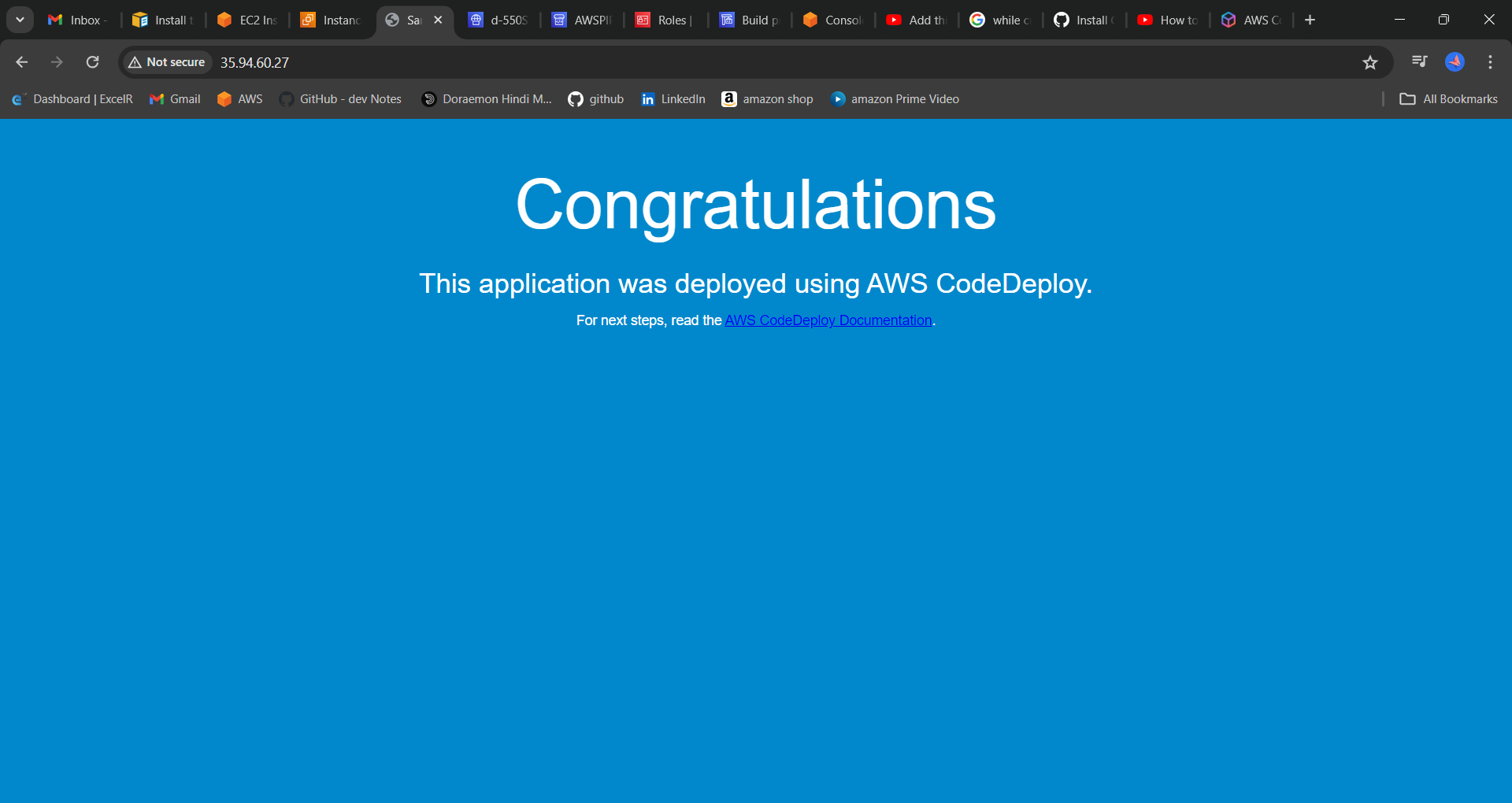


The Deploy will fail here ,to resolve this we have to install CodeDeploy agent on our instance

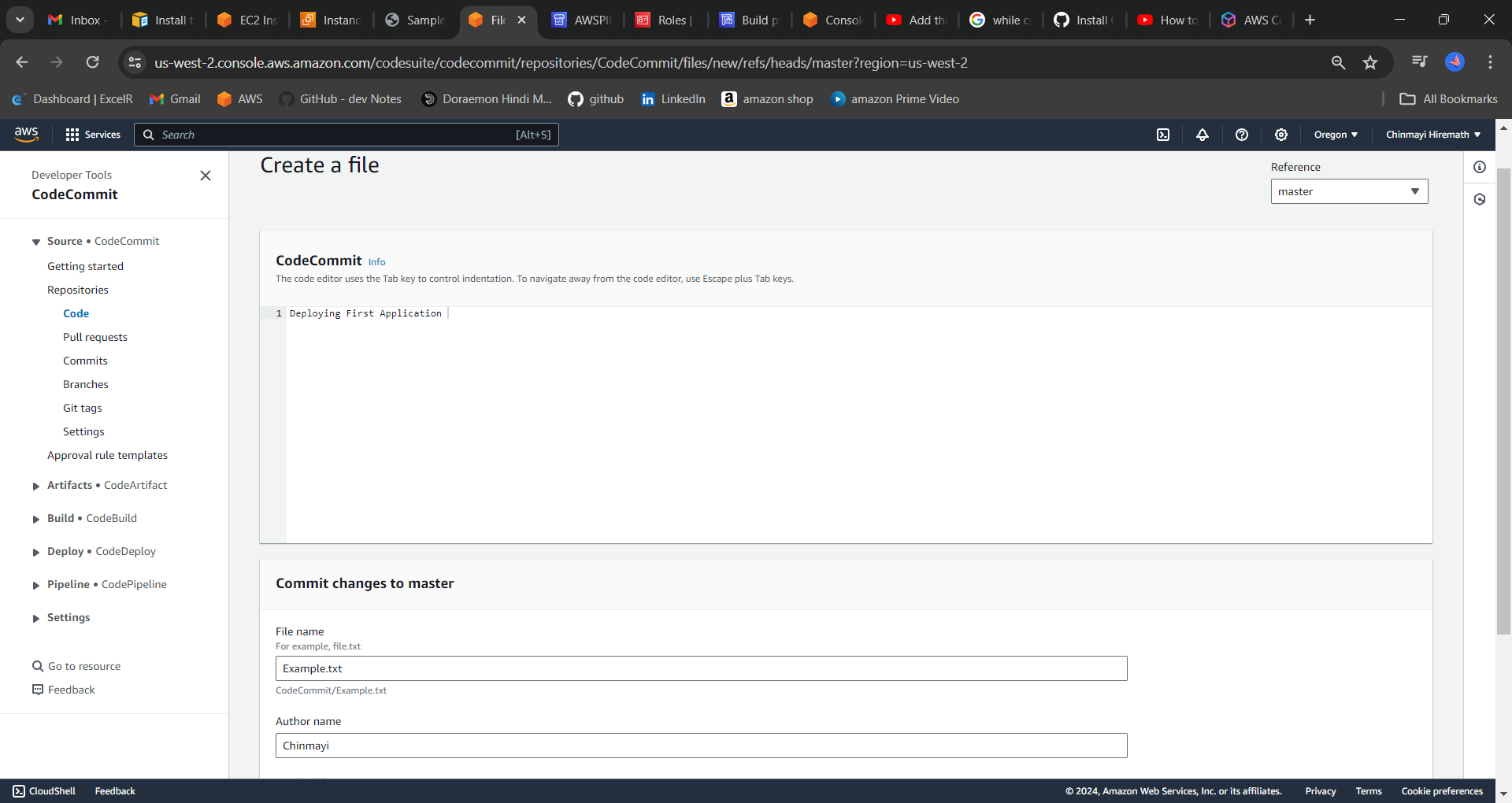


STEP 28: Check all the stages ran successfully.

  
STEP 29: Now navigate to the EC2 service and copy the public IP of the created instance and paste it in the new tab.



STEP 30: Now lets modify the code commit repo and pipeline should automatically trigger itself post committing it.



See pipeline automatically triggered itself

