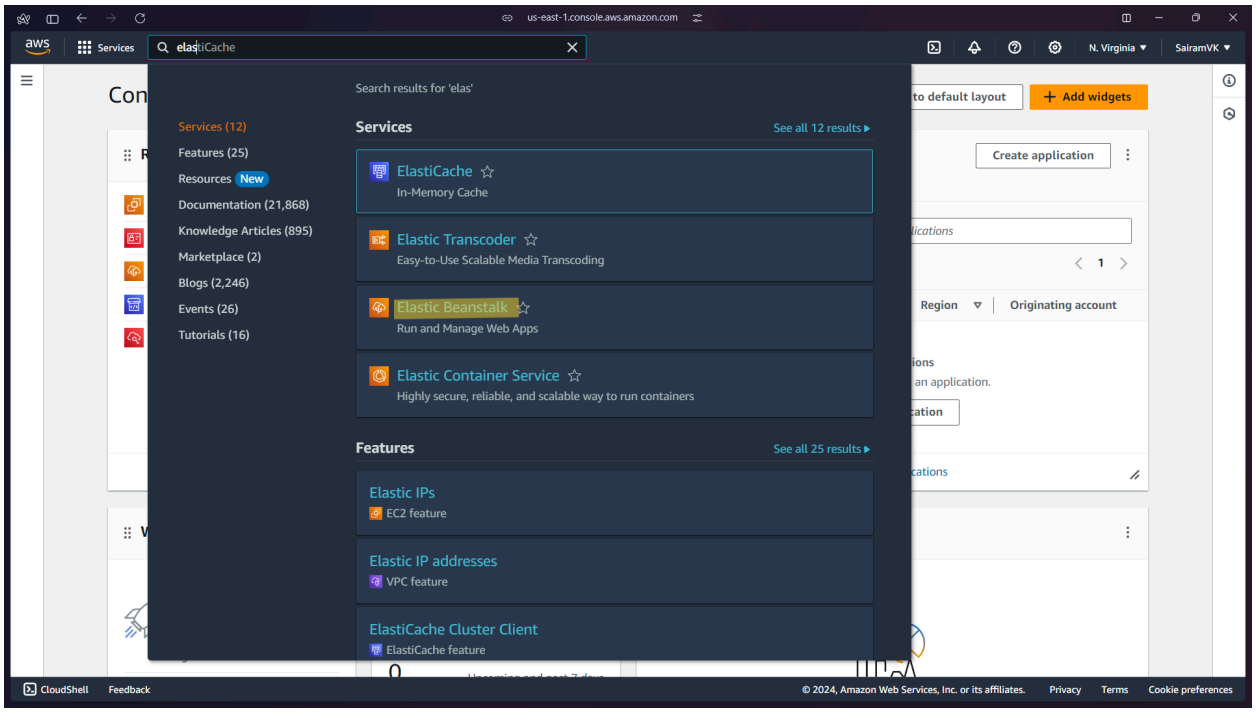
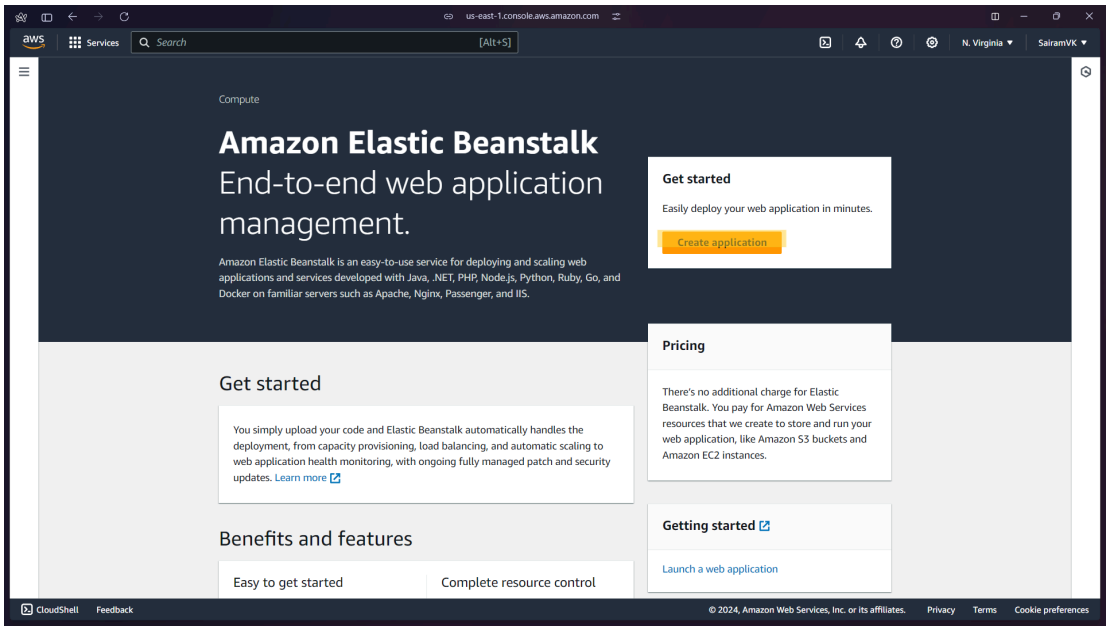
Sairam Konar D15C 27 2024-25

Aim: To Build Your Application using AWS CodeBuild and Deploy on S3/ SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

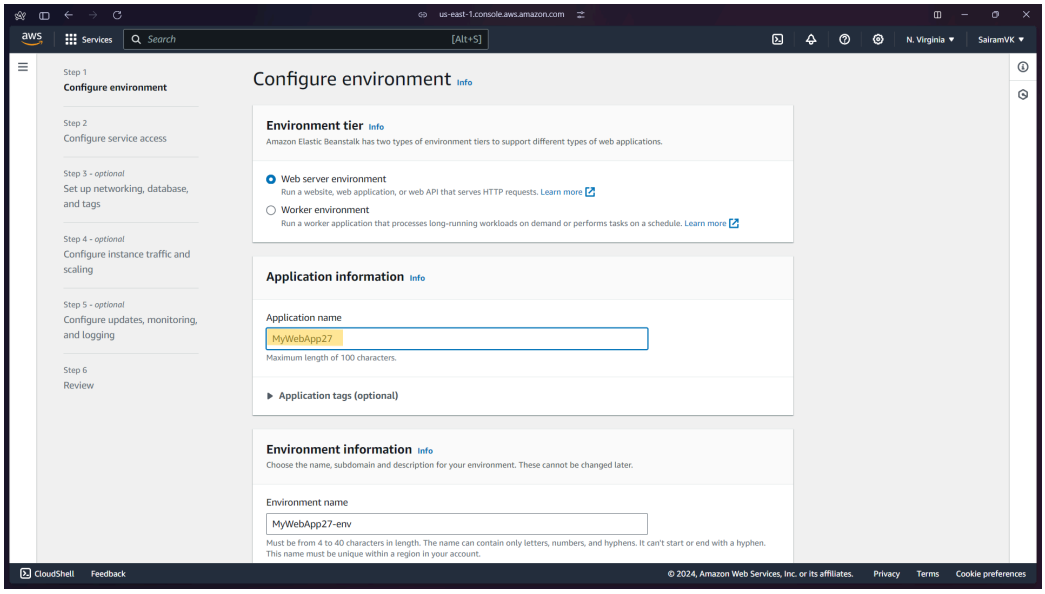
Step 1: Login to your AWS console. Search for Elastic Beanstalk in the searchbar near services.

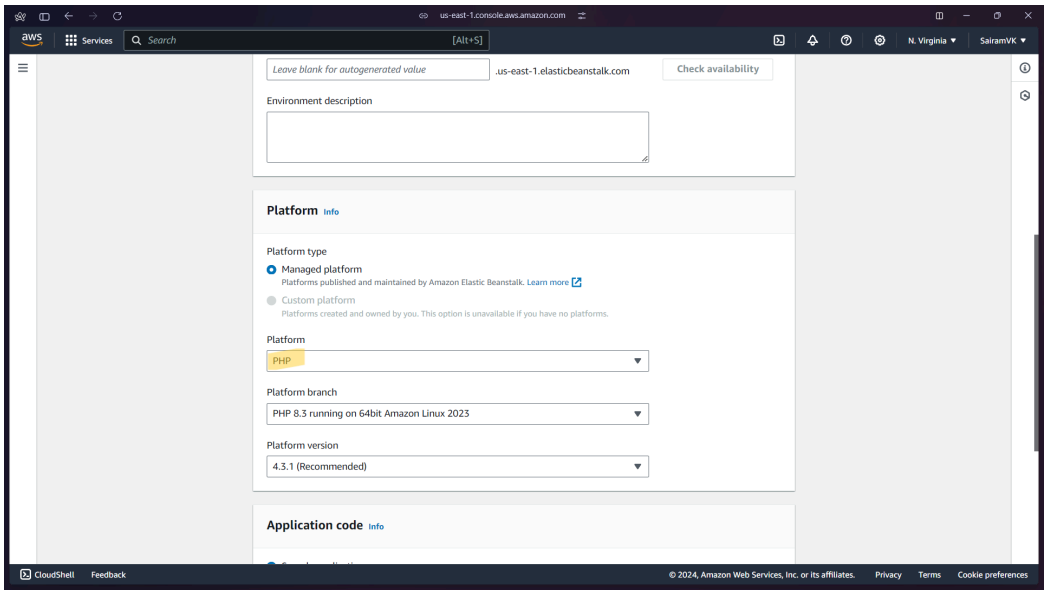


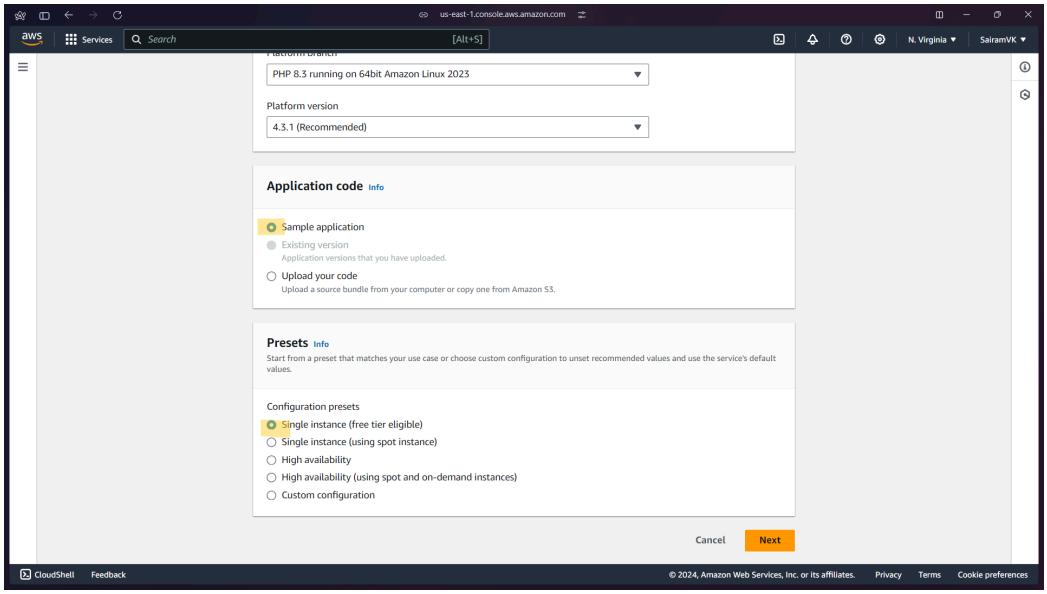
Step 2: Go to Elastic Beanstalk and click on Create Application



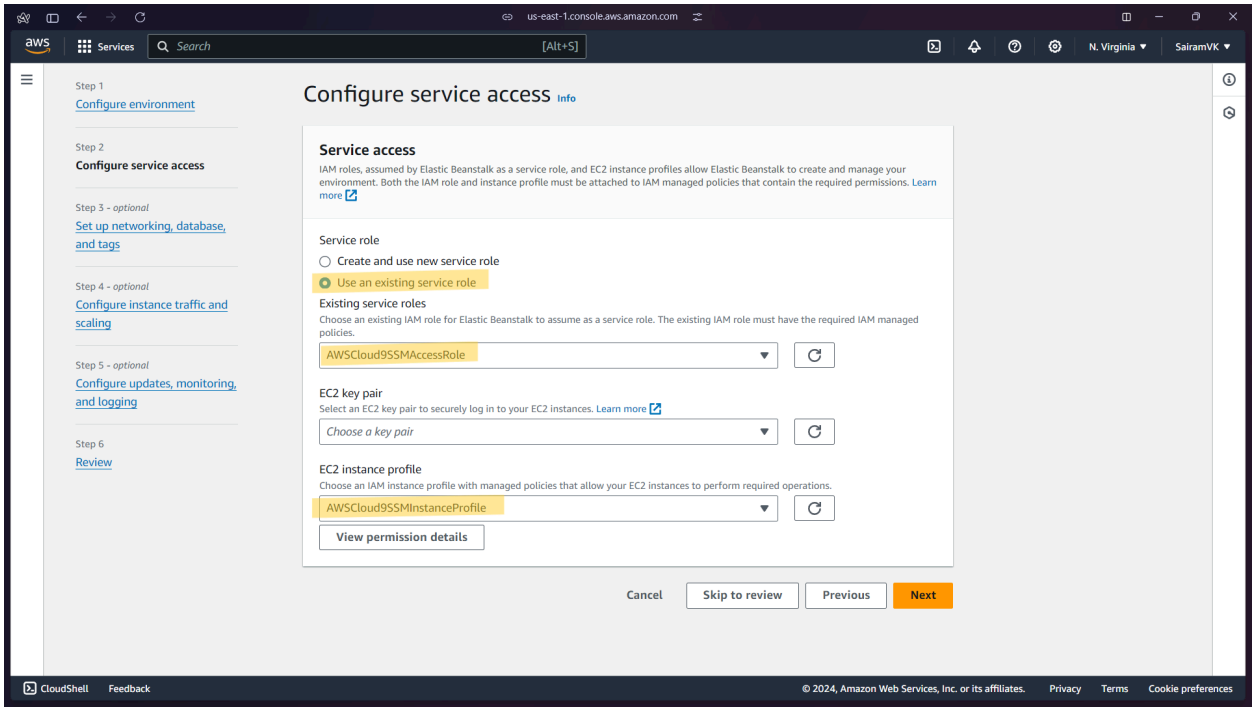
Step 3: Enter the name of your application. Scroll down and in the platform, select platform as PHP. Keep the application code as Sample Application. Set the instance to single instance. Click on NEXT.



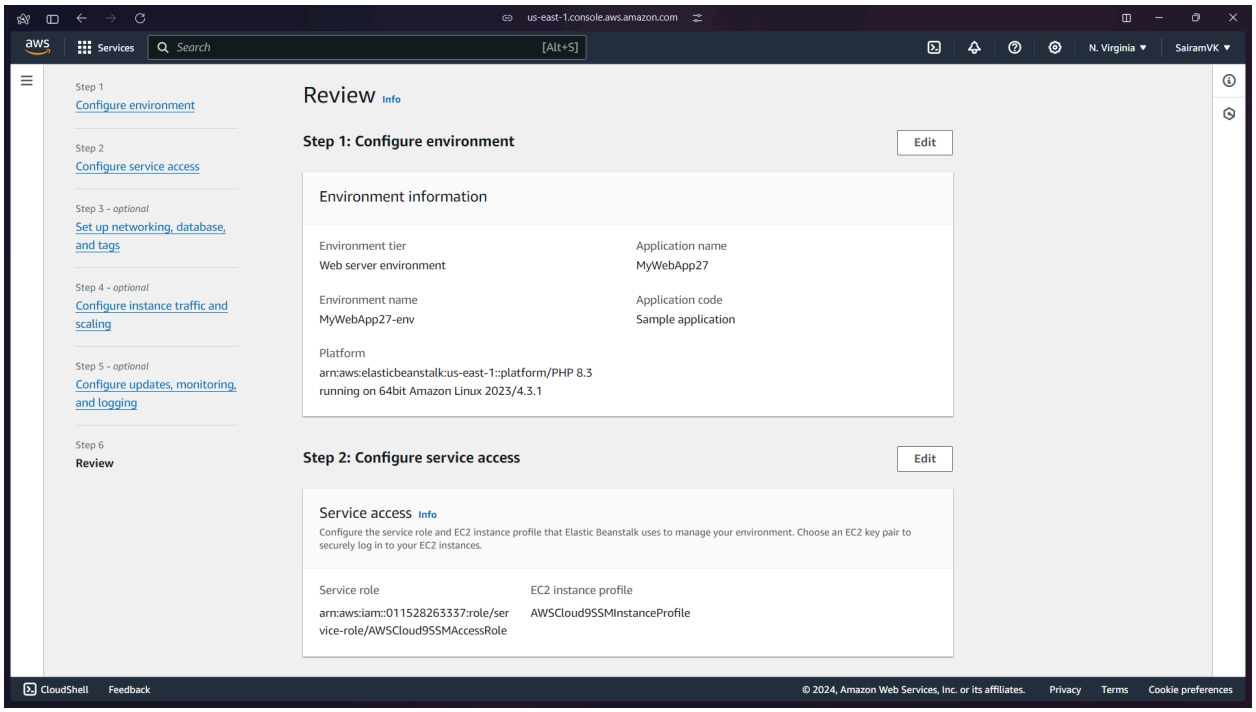


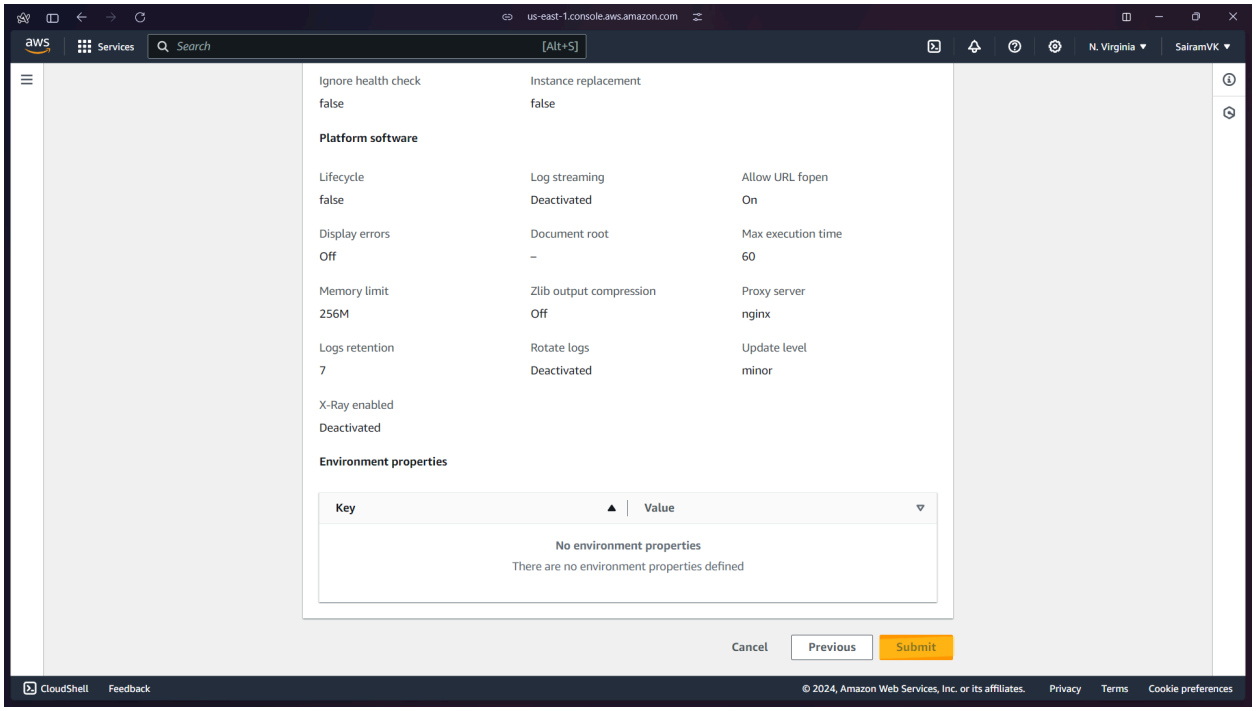


Step 4: Use an existing service role and choose whatever service role is available on your account.



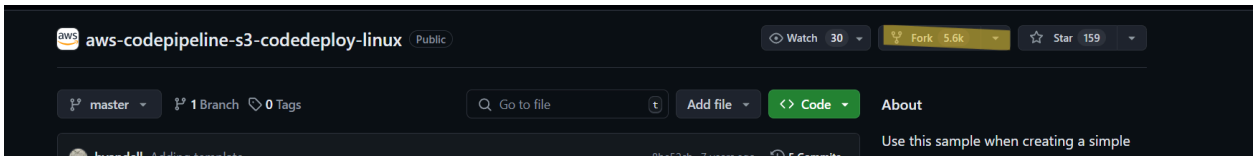
Step 5: Review the settings that you have set up for your application and submit your application.

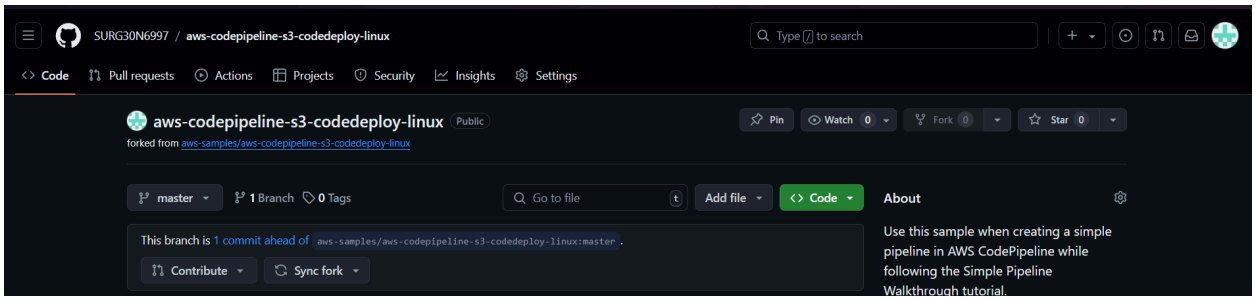




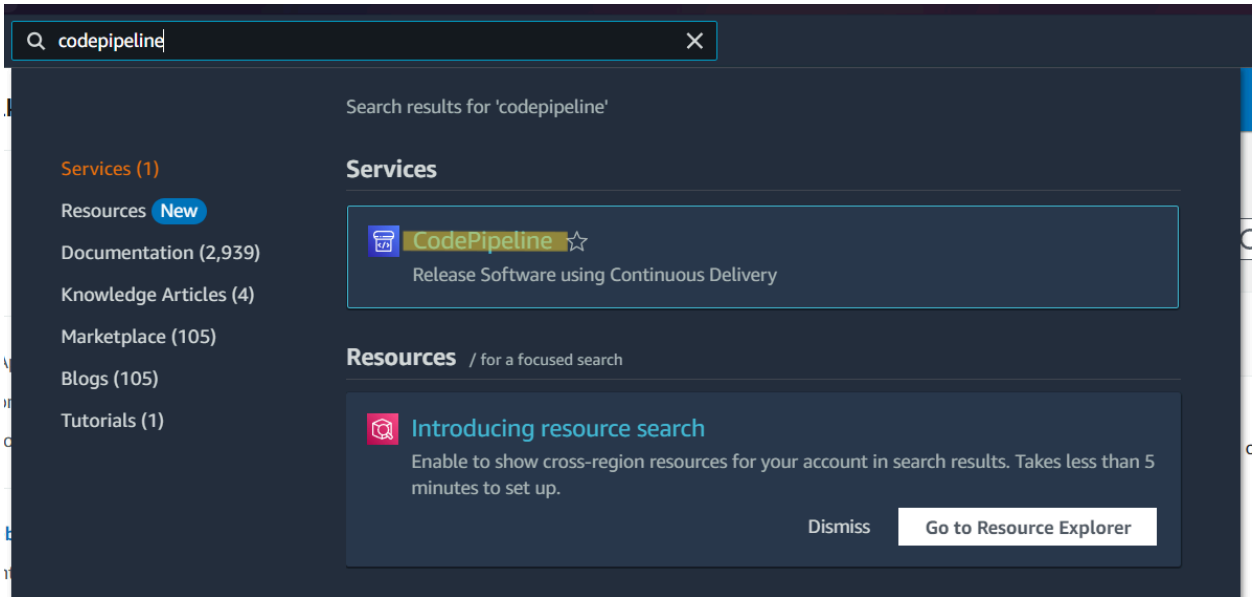
Sairam Konar D15C 27 2024-25

Step 6: Go to the github link below. This is a github with a sample code for deploying a file on AWS CodePipeline. Fork this repository into your personal github.
https://github.com/aws-samples/aws-codepipeline-s3-codedeploy-linux

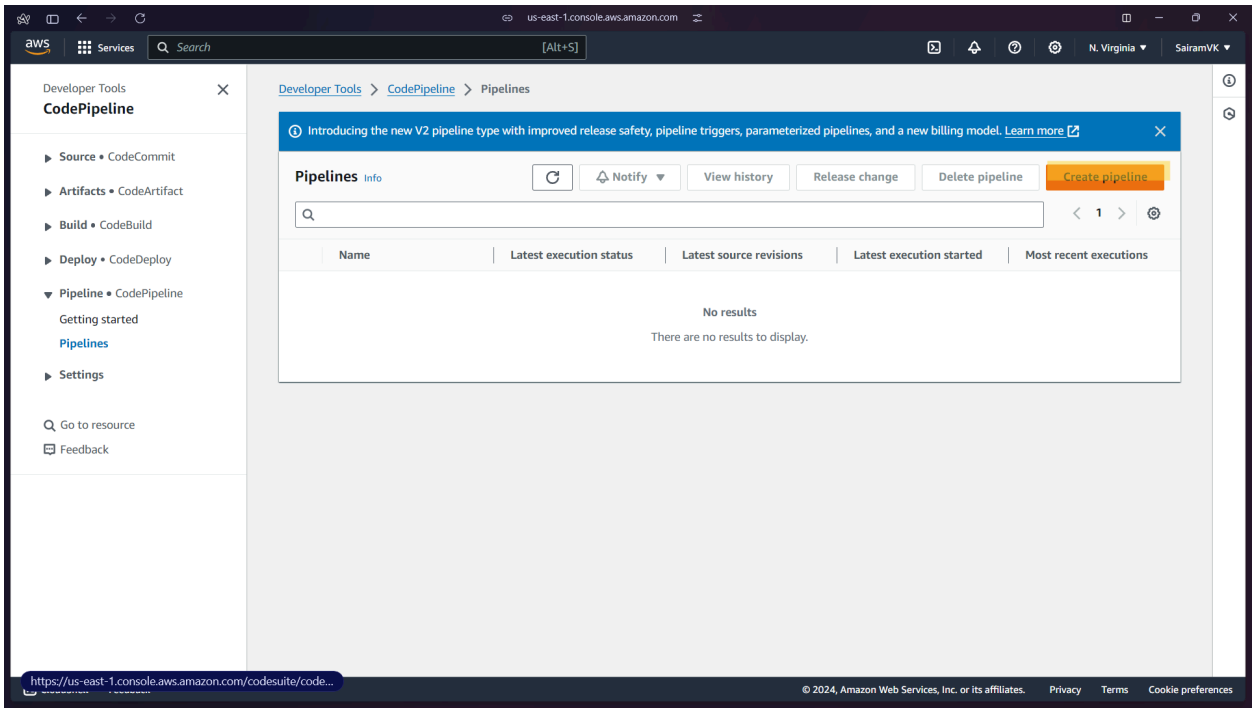




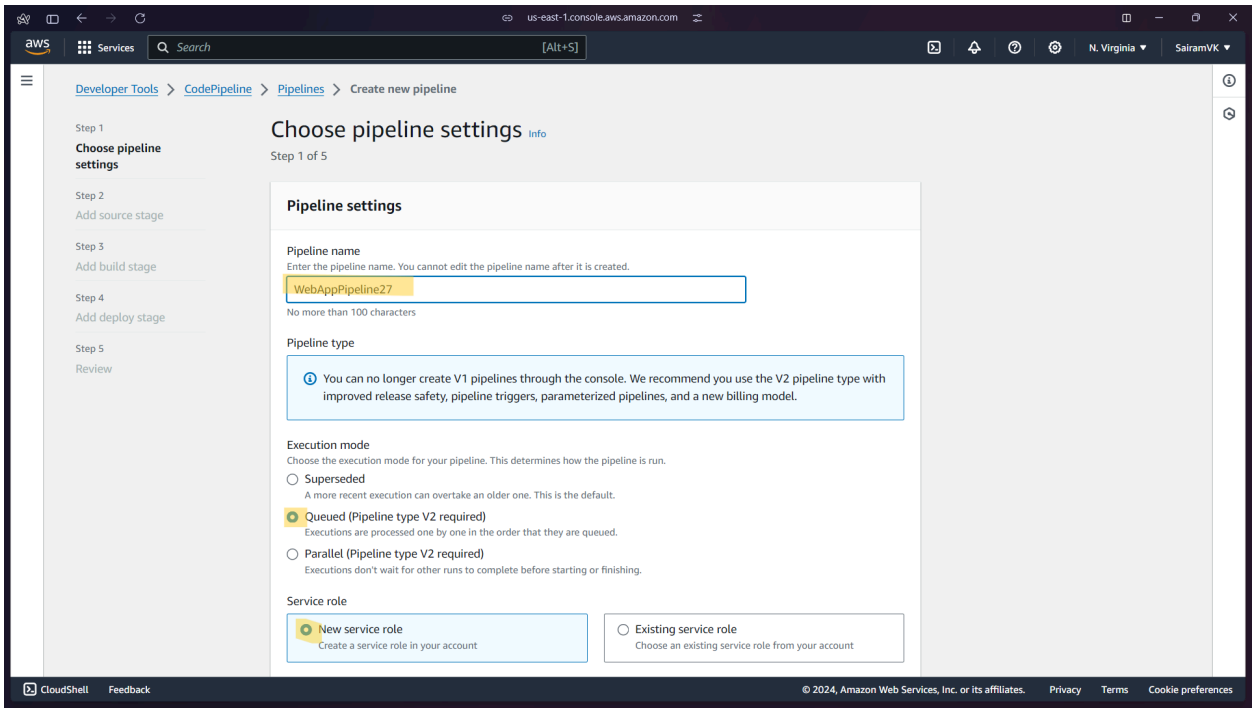
Step 7: Search CodePipeline in the services tab and click on it.

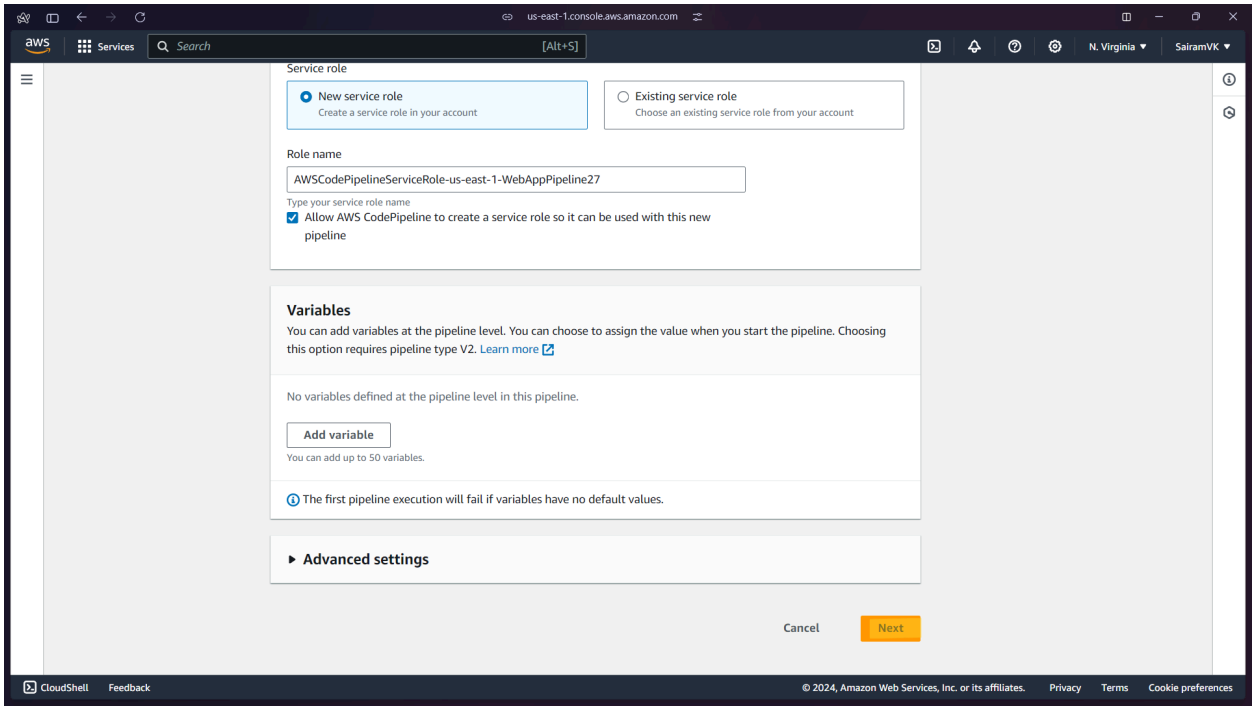


Step 8: Click on Create Pipeline.

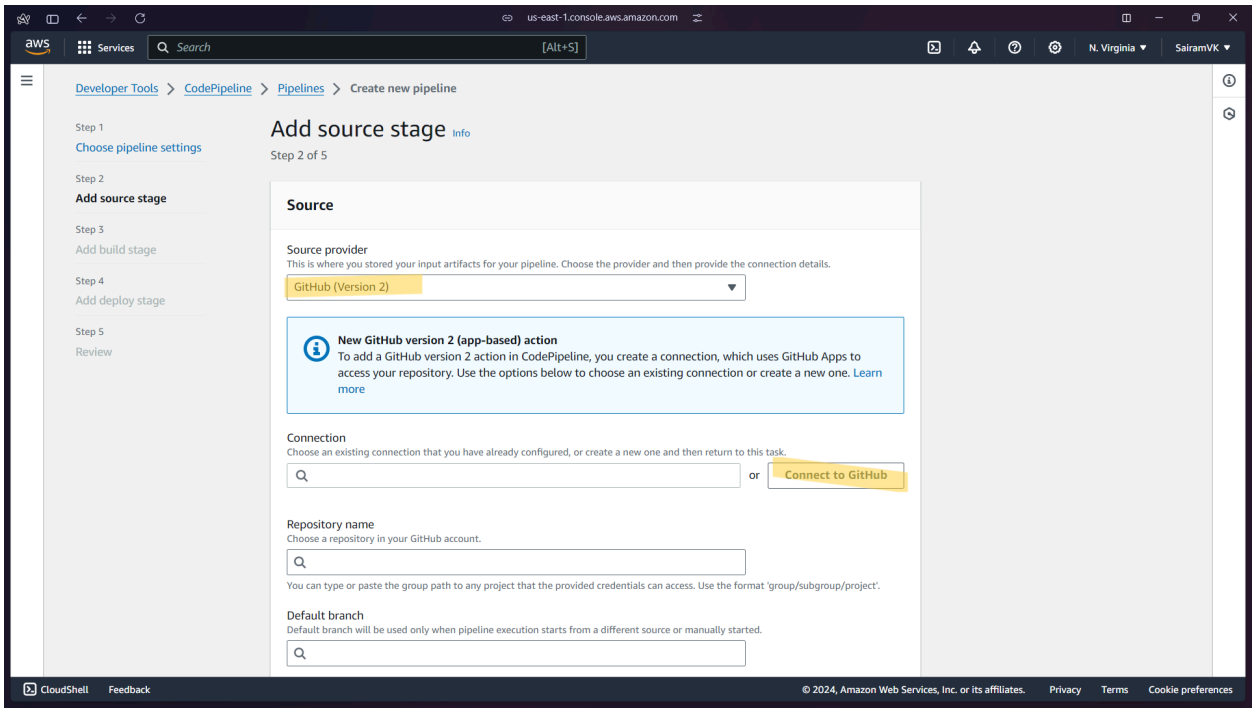


Step 9: Give a name to your Pipeline. A new service role would be created with the name of the Pipeline.

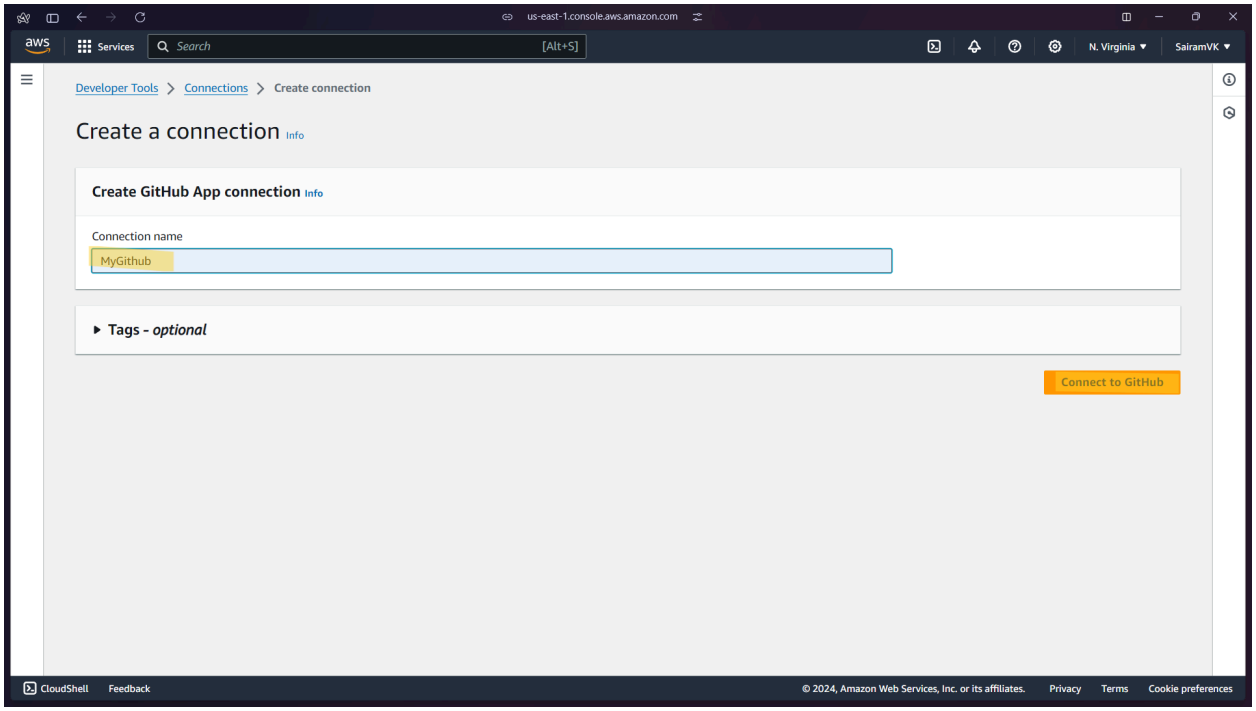


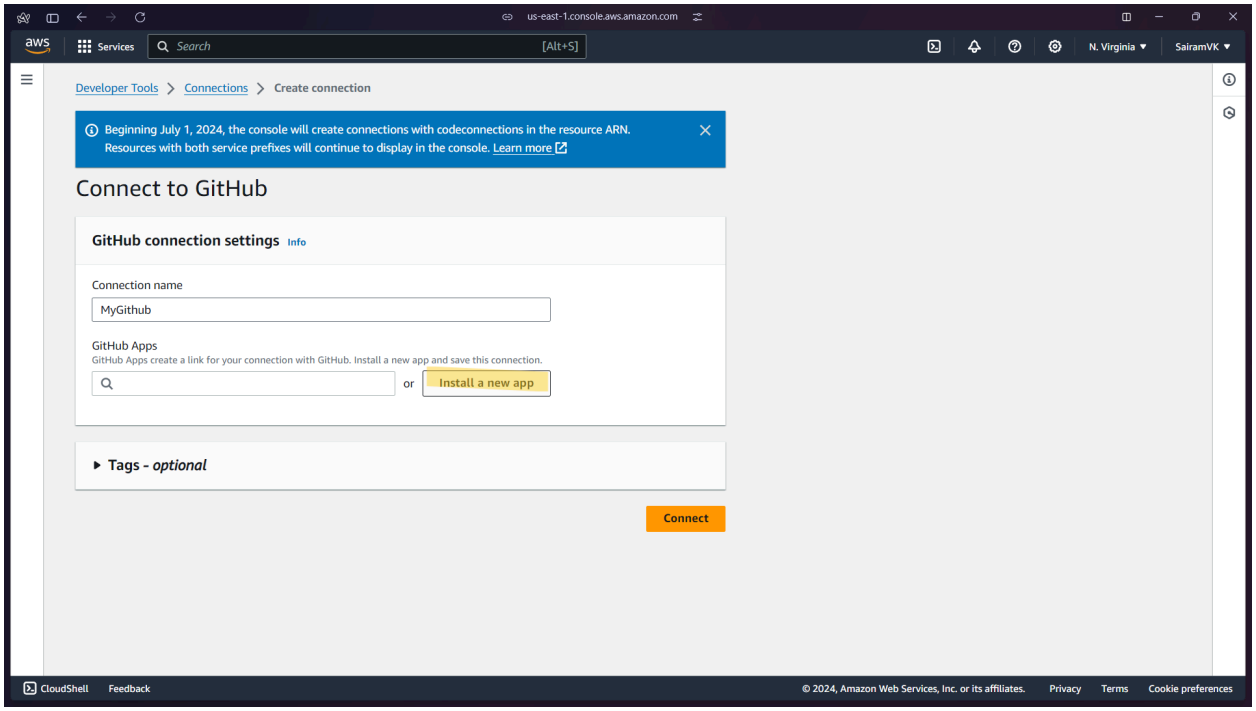


Step 10: Select a source provider (as Github (Version 2)). Click on Connect to Github to connect your github.

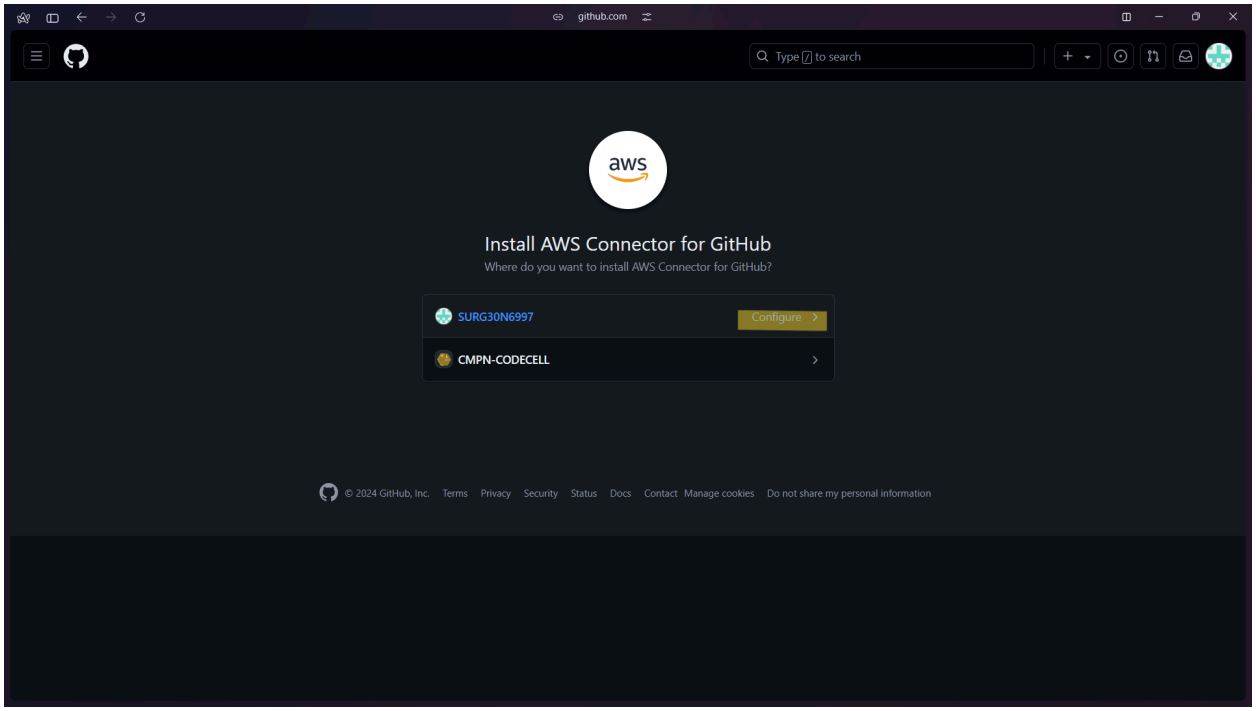


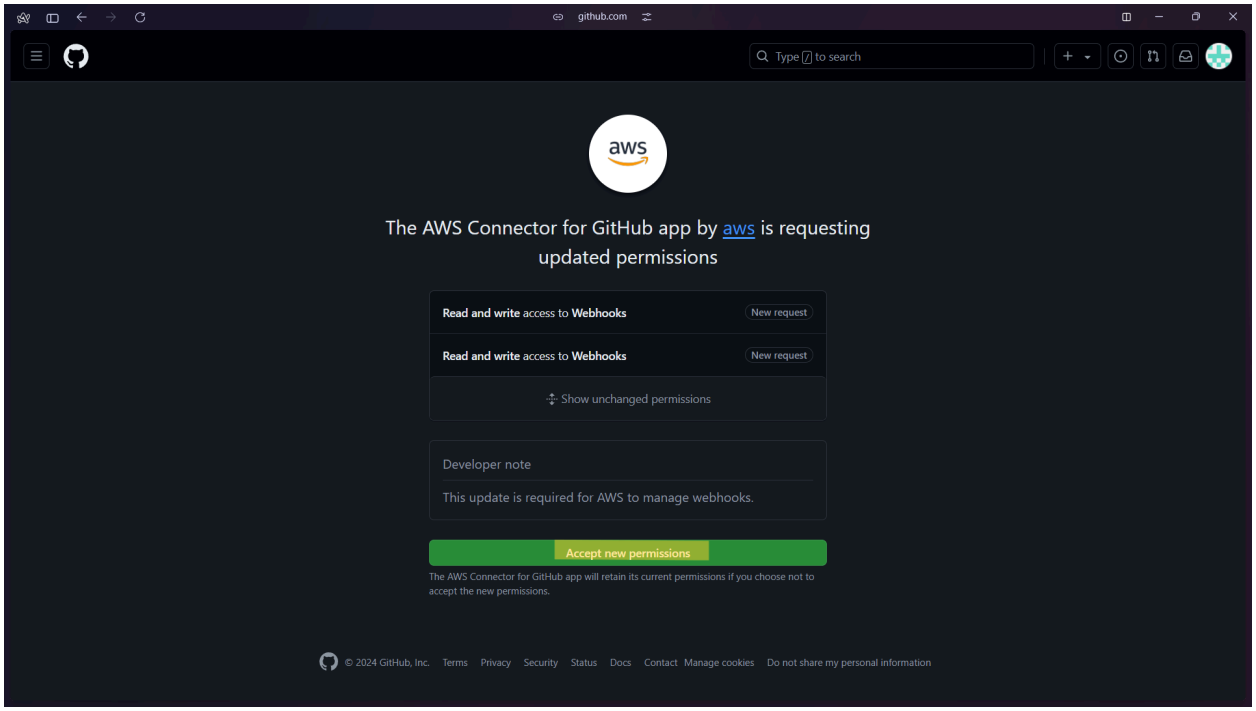
Step 11: Give a name to your GitHub app Connection and click on Connect. This will give you a prompt to either to select a GitHub app or install a new app. If this is your first time, click on Install a new app.



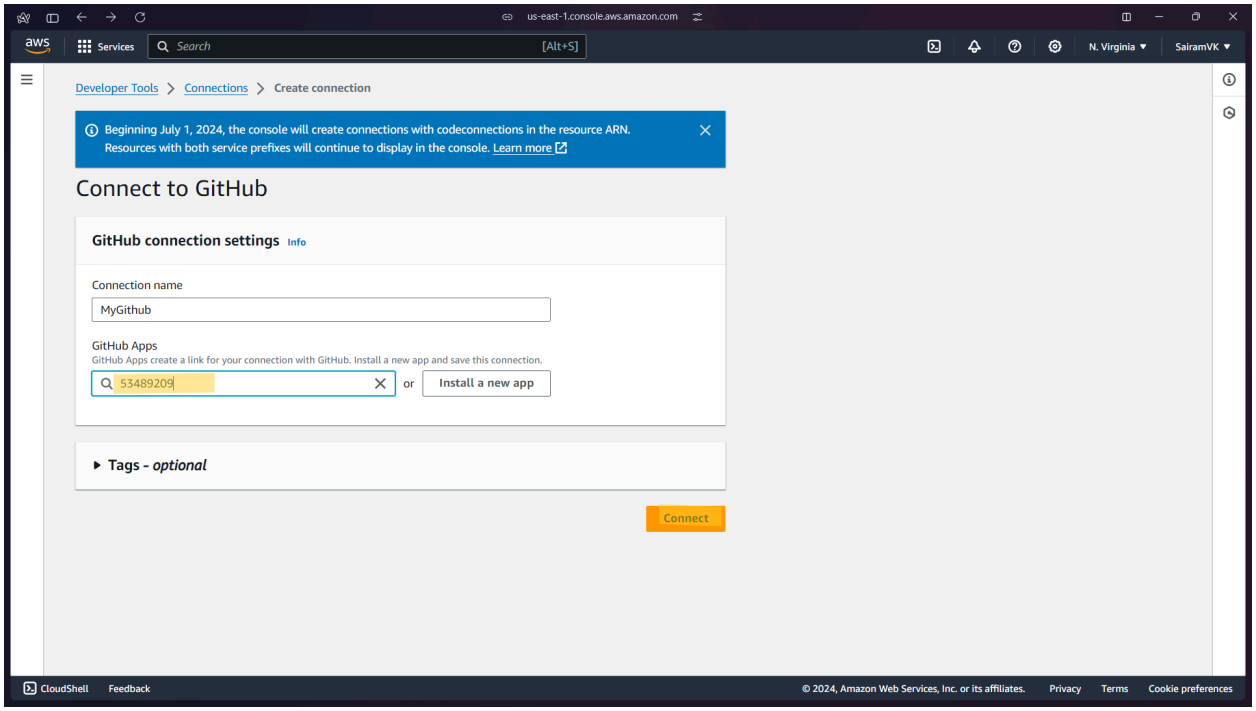


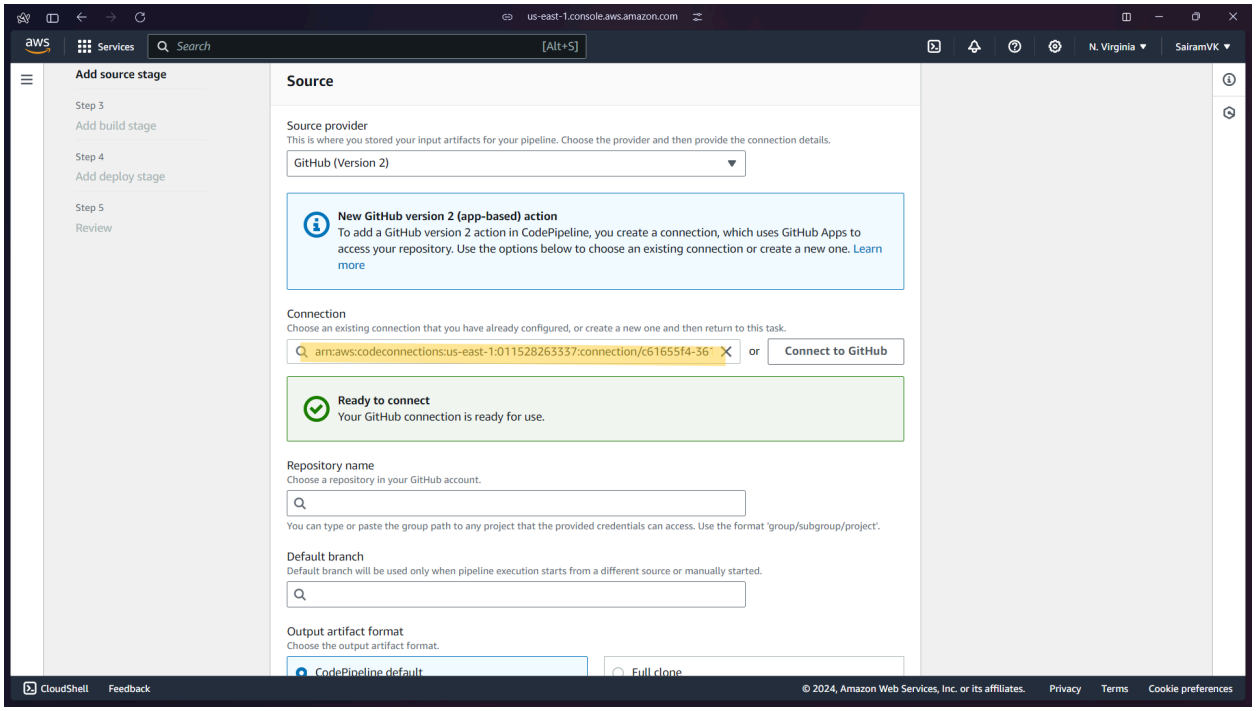
Step 12: This will direct you to install AWS Connector on your GitHub. Install it to your account and give it its permissions.



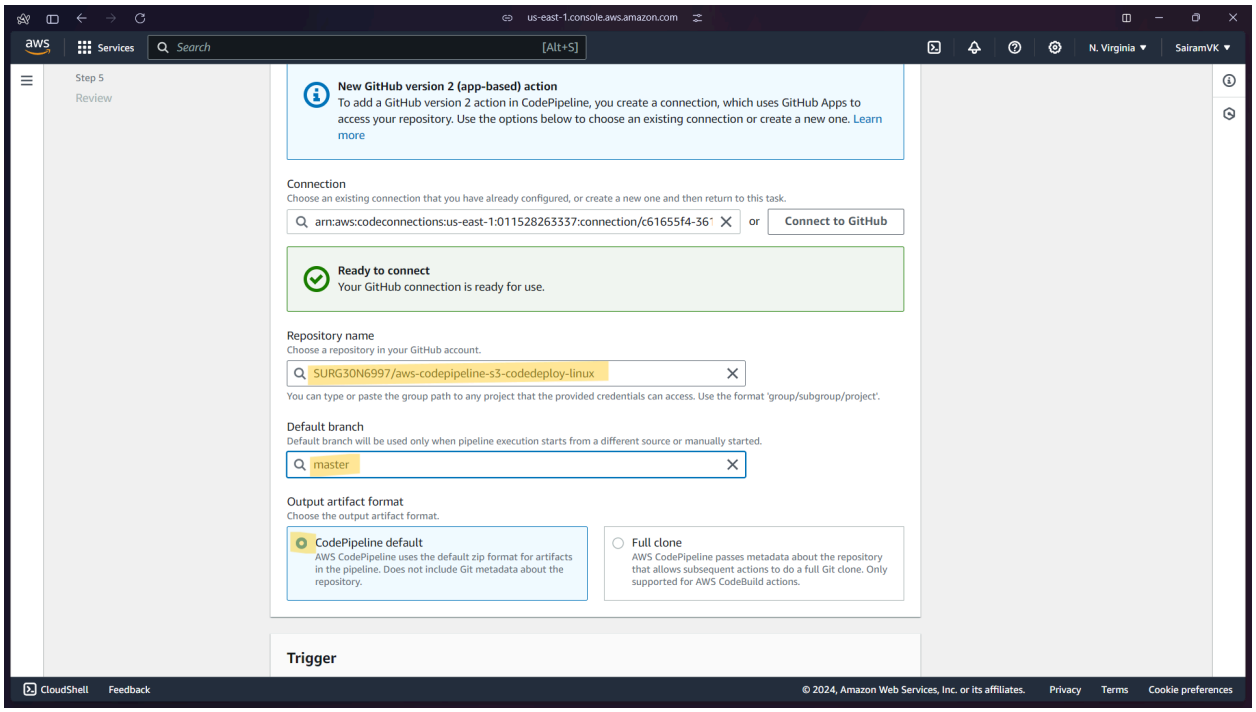


Step 13: After the app is set up, it gives the number in the text field. Click on Connect. After clicking on connect, the link is shown in the connection field and AWS shows that GitHub connection is ready to use.

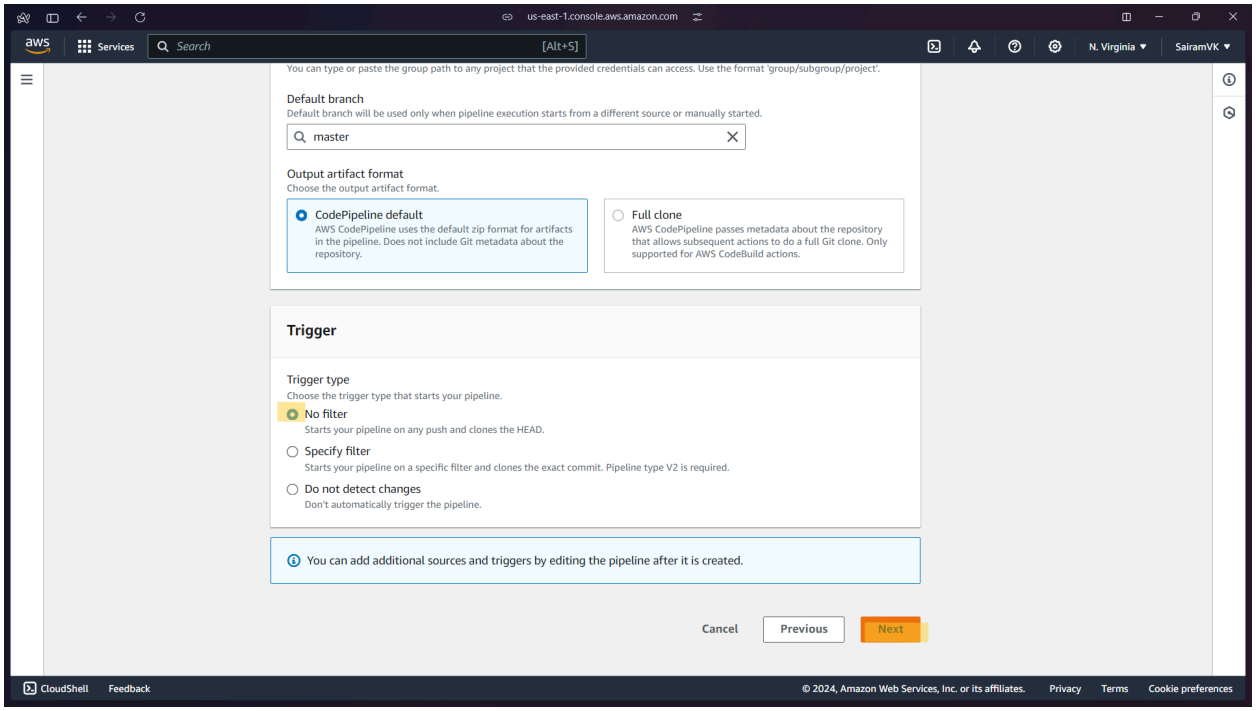




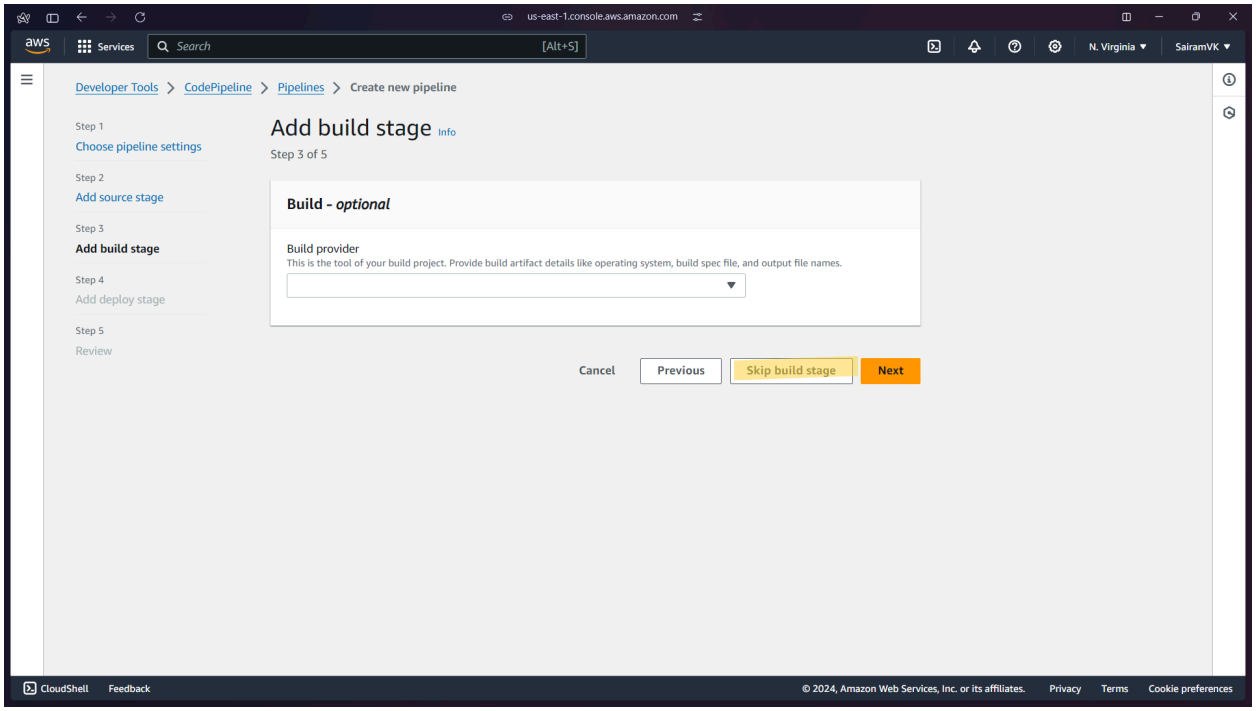
Step 14: Select the repository that you had forked to your GitHub. After that select the branch on which the files are present (default is Master).

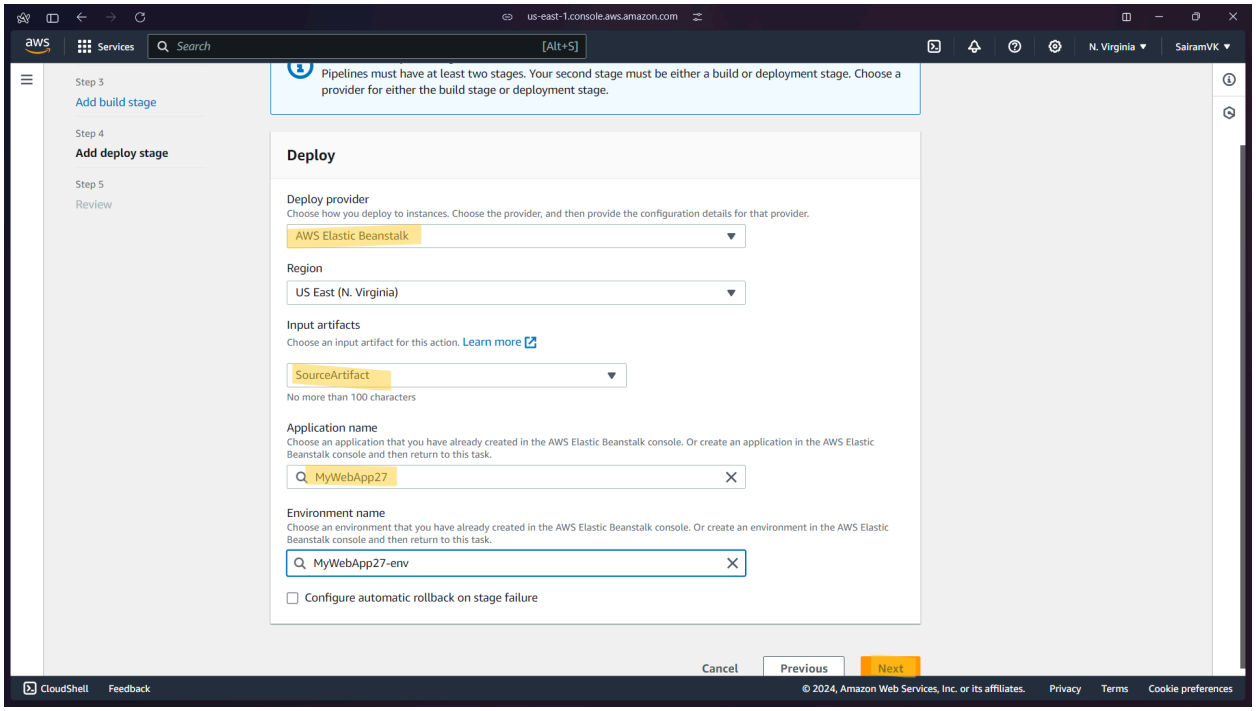


Step 15: Set the Trigger type as no filter. This would allow it to the website to update as soon as some change is made in the github.

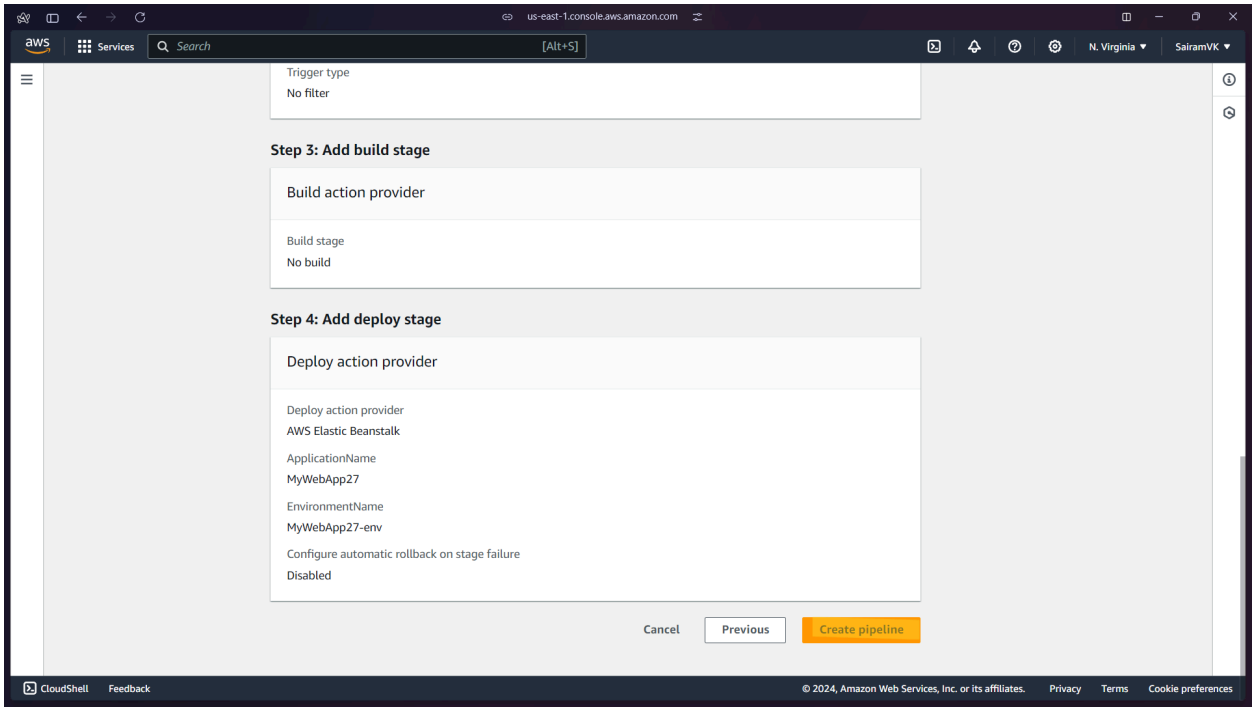


Step 16: Skip the build stage and go to Deploy. Select the deploy provider as AWS Elastic Beanstalk and Input Artifact as SourceArtifact. The application name would be the name of your Elastic Beanstalk. Then click on next.

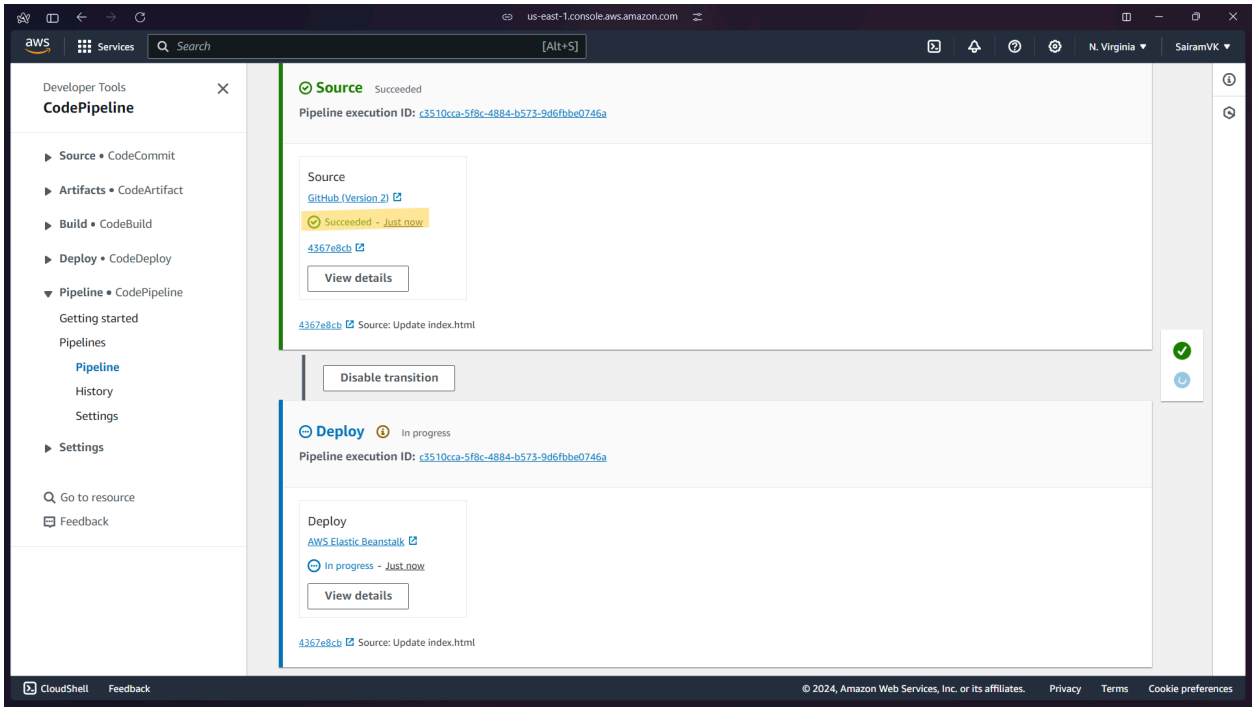




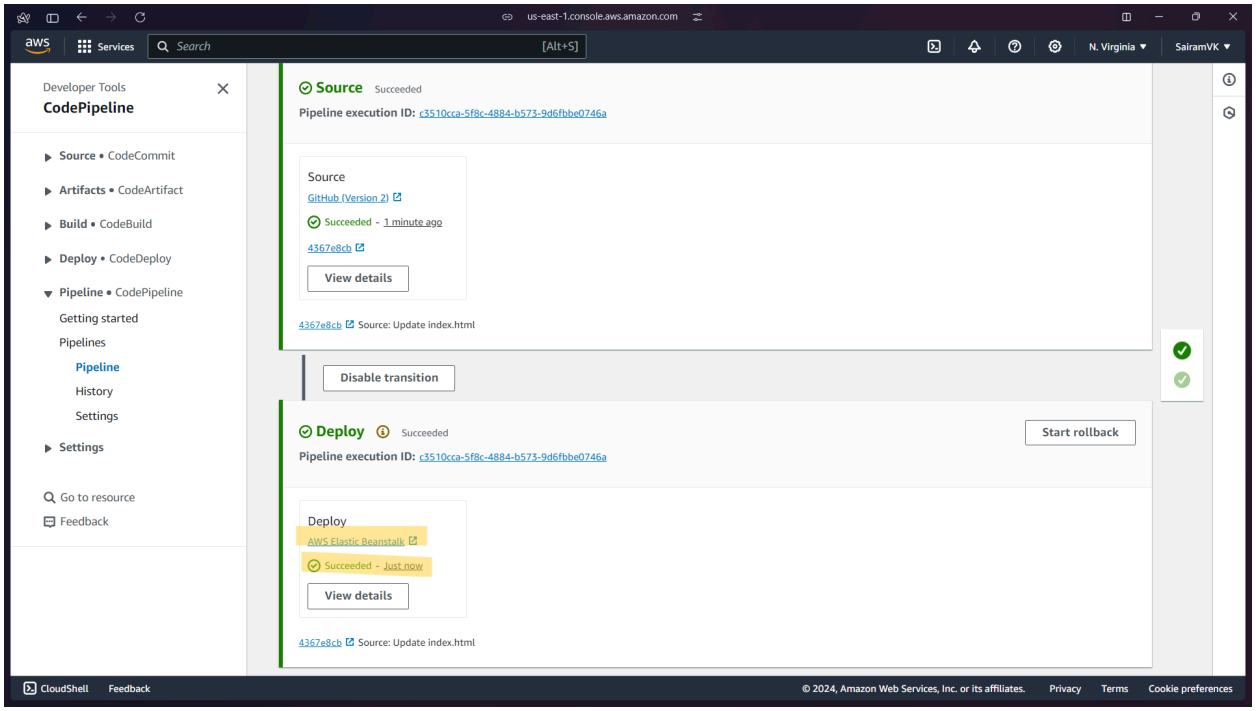
Step 17: Check all the information and click on create Pipeline.



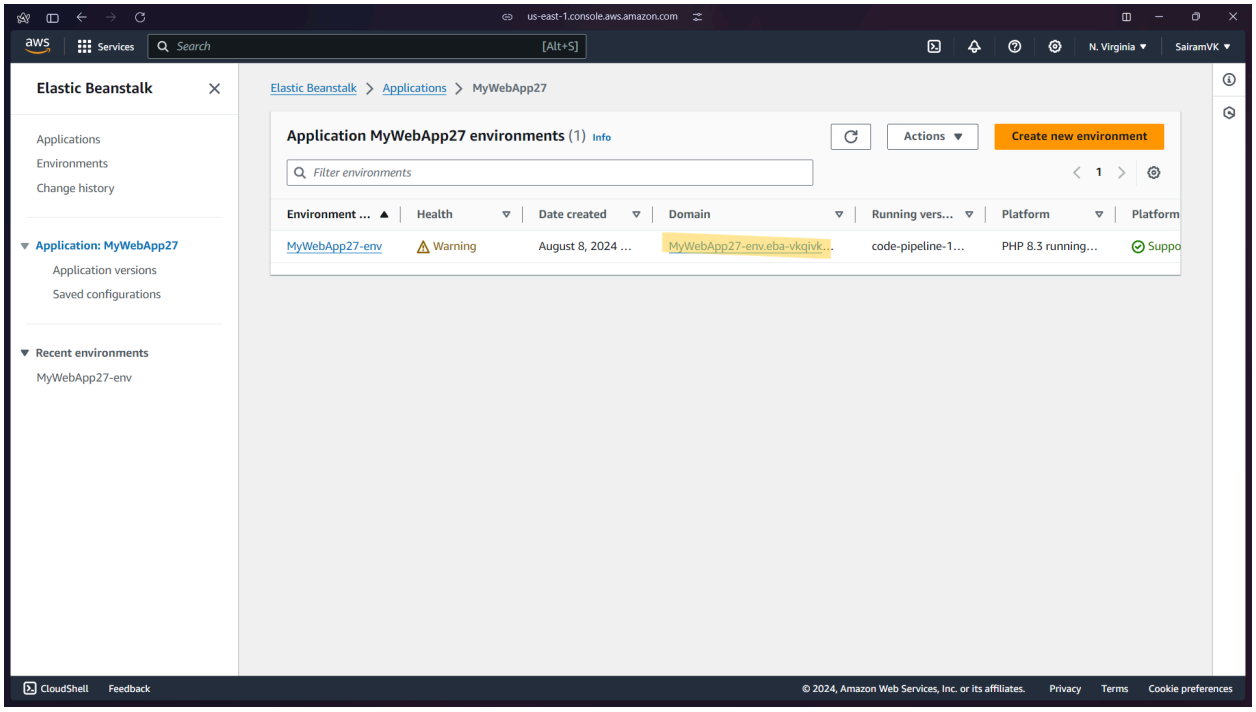
Step 18: If the pipeline is successfully deployed, this screen comes up where the source is set up and then it is transitioned to deploy.



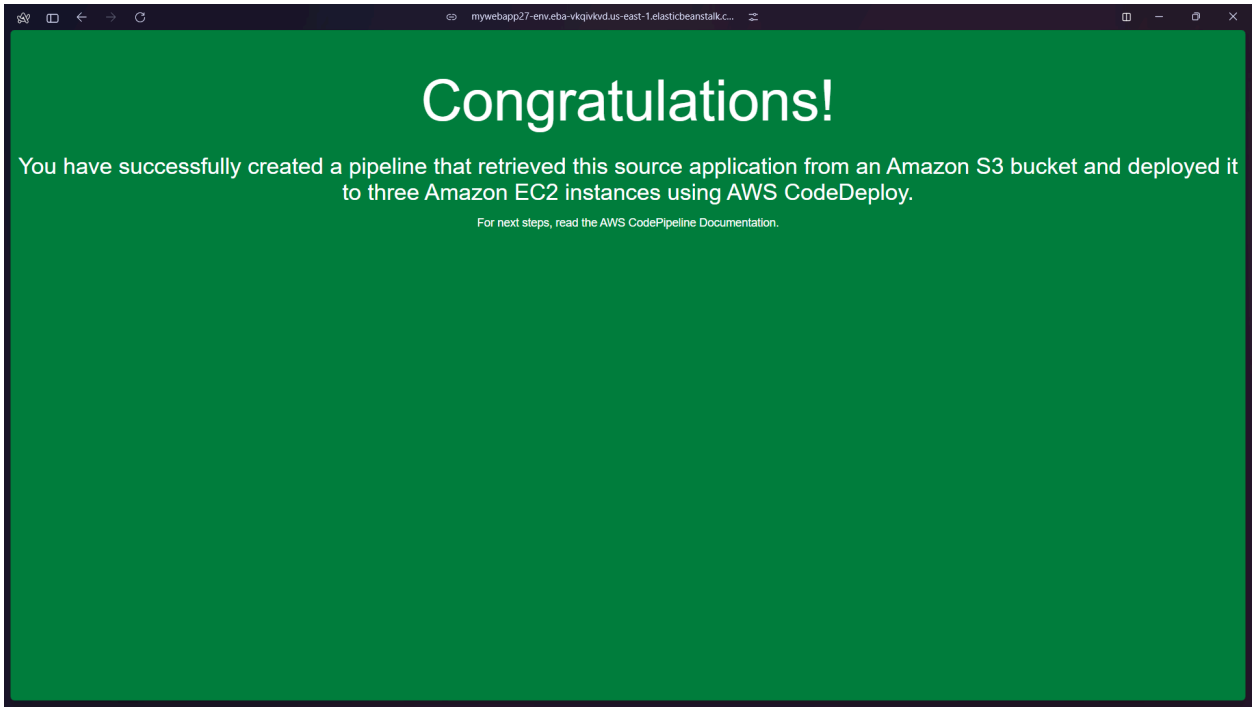
Step 19: Once the deployment is complete, click on the AWS Elastic Beanstalk under Deploy.



Step 20: This will redirect you to the application screen of Elastic Beanstalk. Click on the link shown under Domain.



Step 21: This will successfully show the sample website hosted.



Step 22: Now, we make some changes to the index.html file in the github.
For eg: If you make some changes to the <h2> tag.
Once the changes are committed, when the website is refreshed, the changes can be seen.