CI/CD flow for ANgular application using cross-account deployment

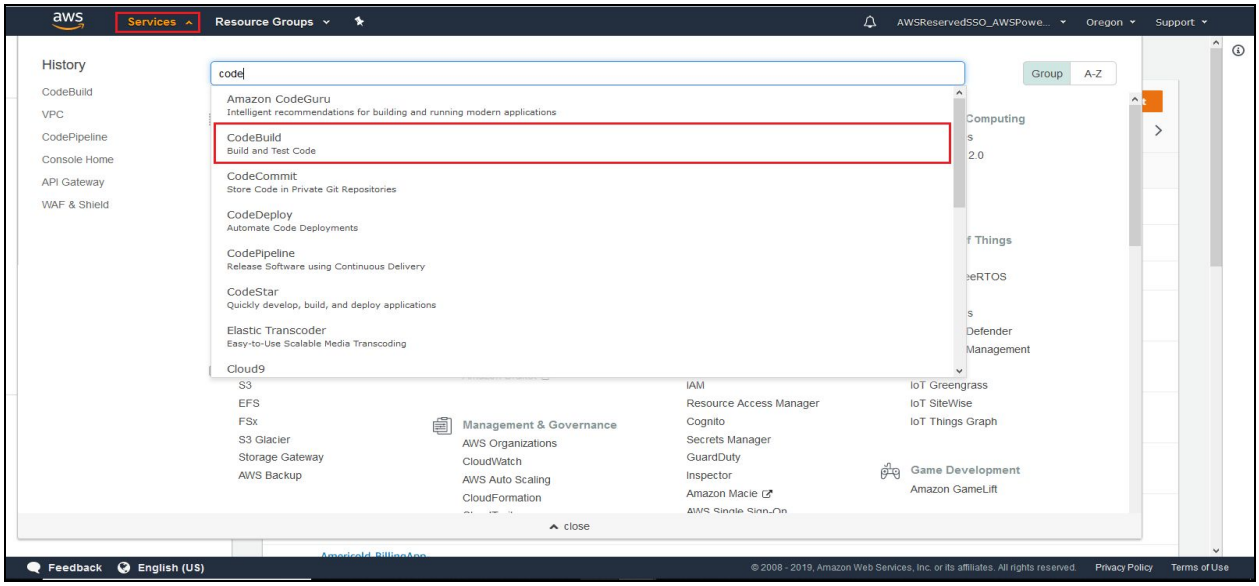
Agenda:

CodeBuild: AWS code build is a fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to deploy. With code build, you don't need to provision, manage, and scale your own build servers. code build scales continuously and processes multiple builds concurrently, so your builds are not left waiting in a queue. You can get started quickly by using prepackaged build environments, or you can create custom build environments that use your own build tools. With code build, you are charged by the minute for the compute resources you use.

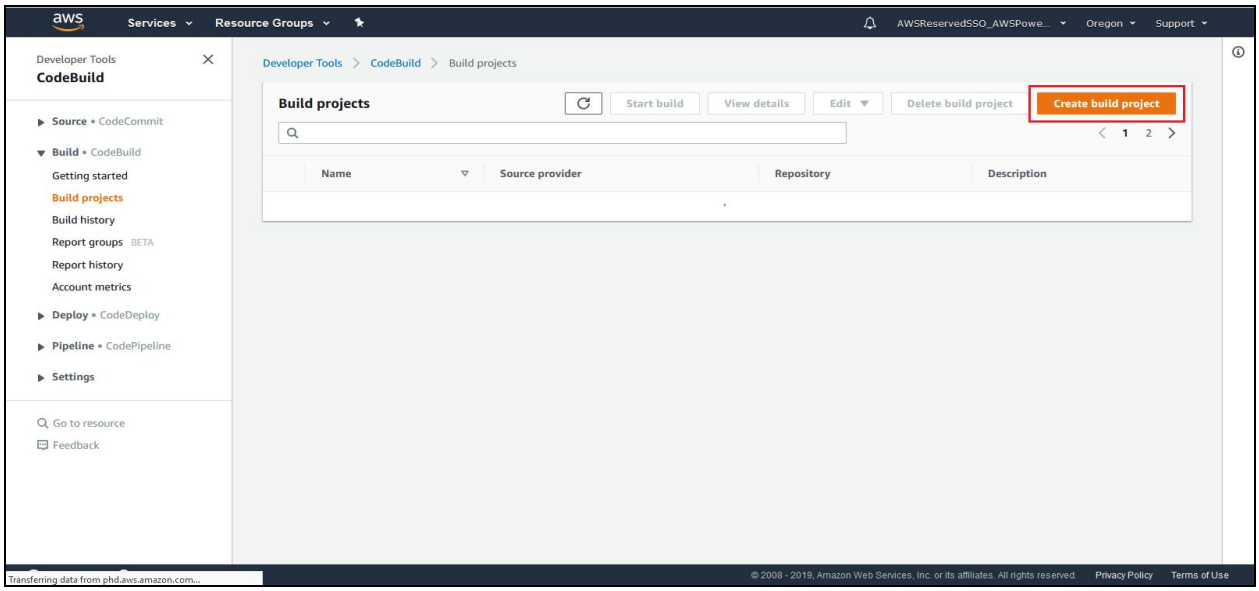
CodePipeline: AWS code pipeline is a fully managed continuous delivery service that helps you automate your release pipelines for fast and reliable application and infrastructure updates. code pipeline automates the build, test, and deploy phases of your release process every time there is a code change,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

By the end of the document, we learn how to create a CI-CD pipeline for Angular application using the cross-account.

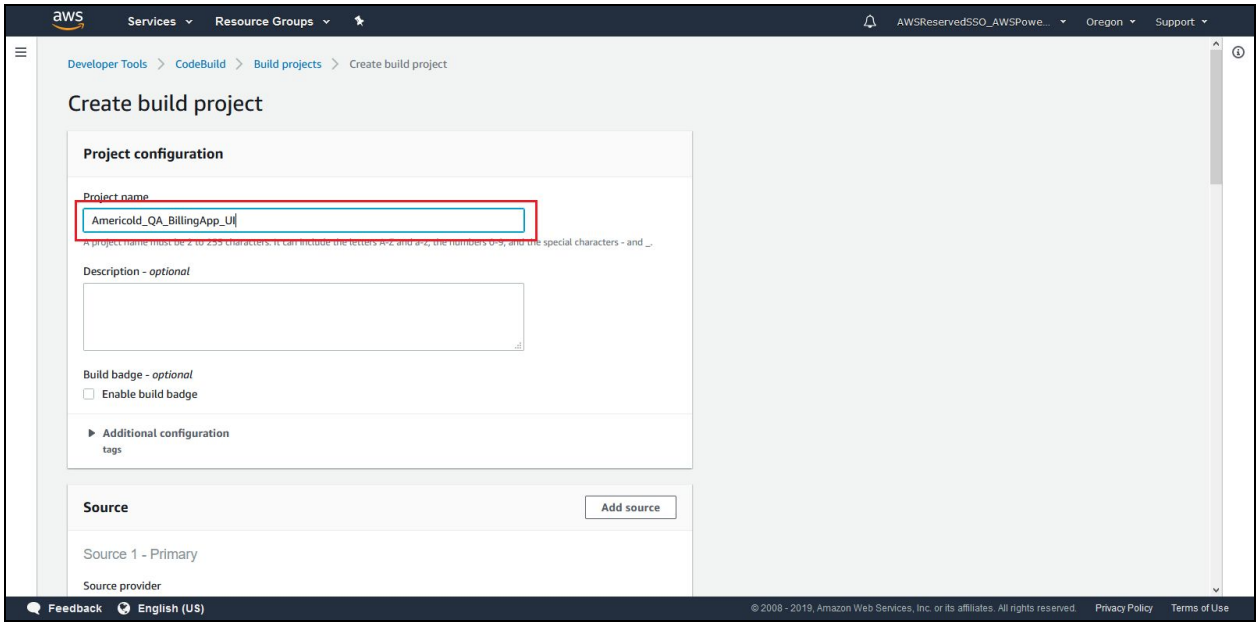
First, we need to login to the AWS console and search for **code build** service.



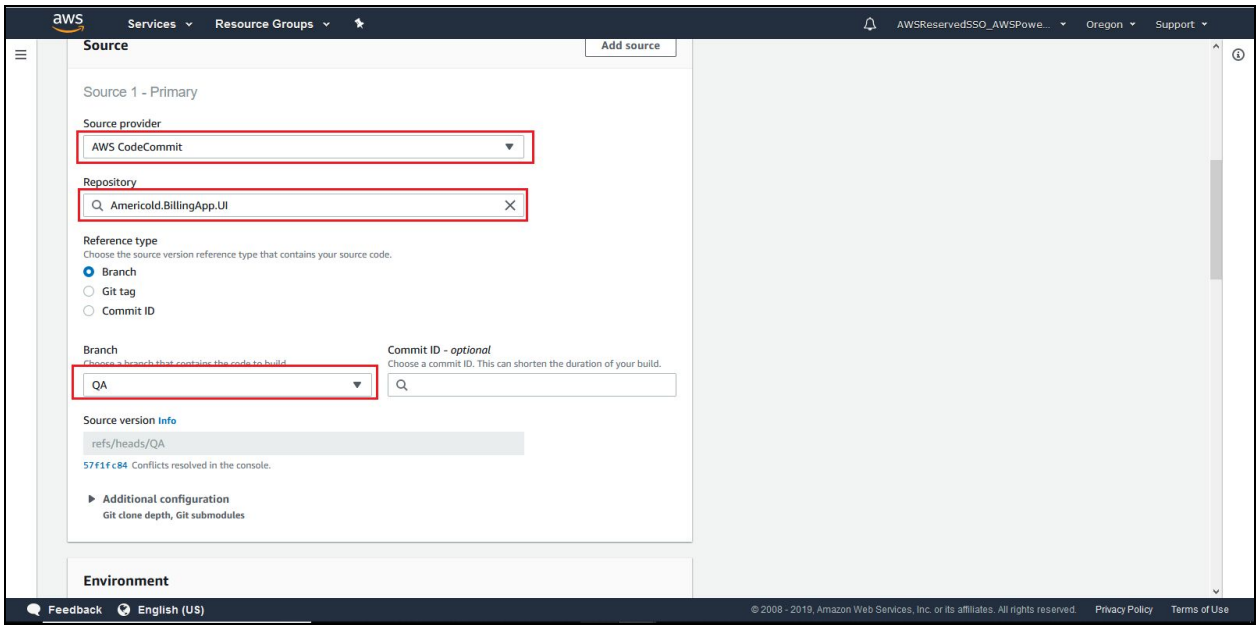
Go to build projects and click on **create build project** to create a new build project for integration**.**



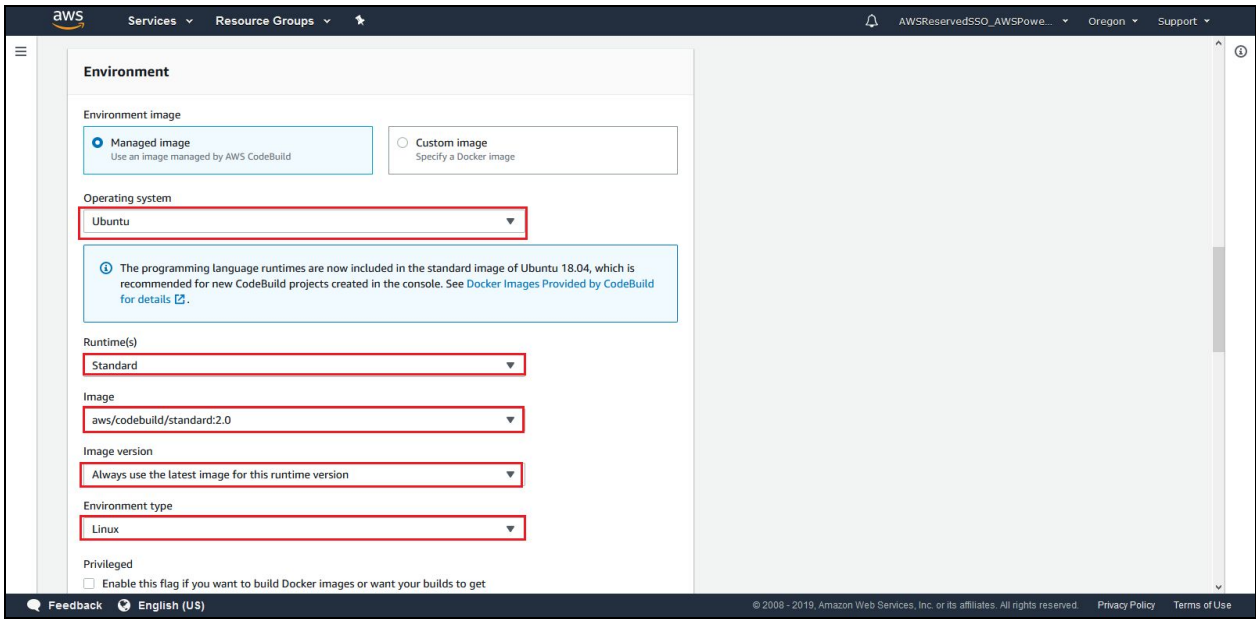
Enter the name of the project**.**

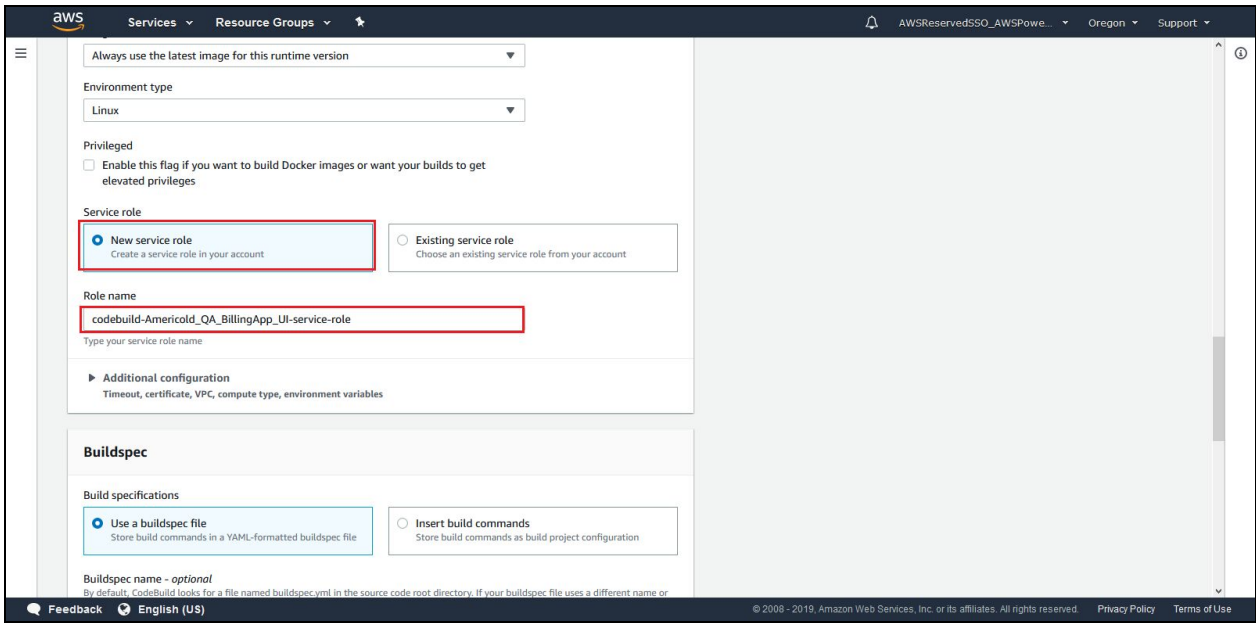


Under the source section, select the source provider as **AWS code commit**, and select the repository and branch as shown below.

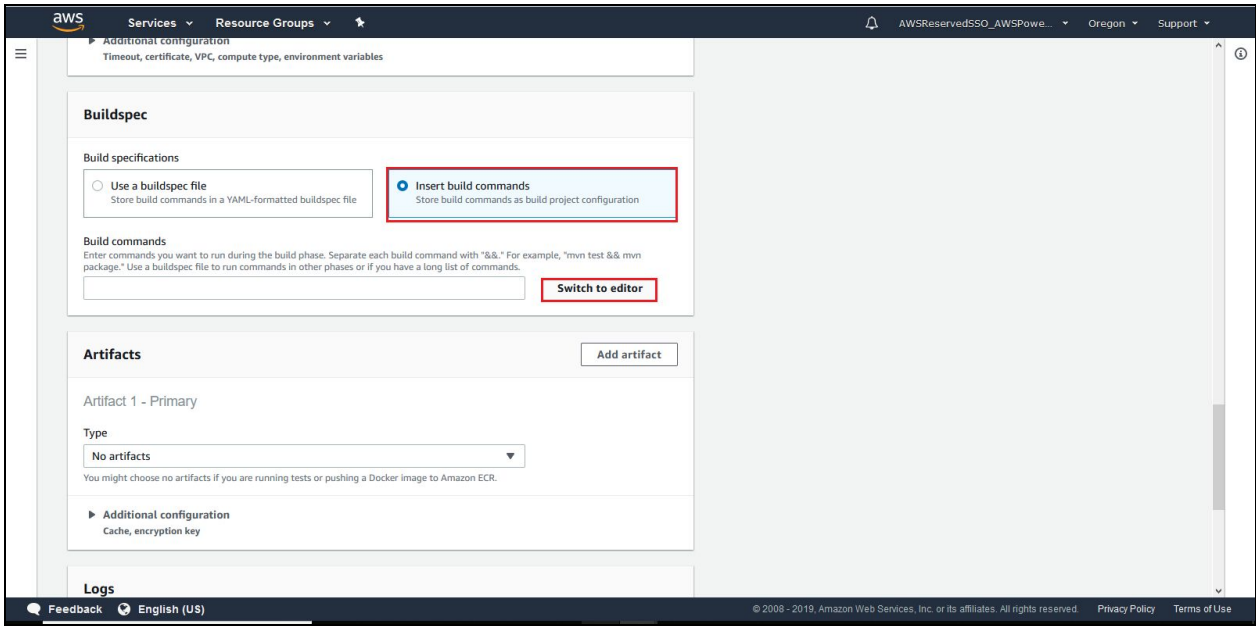


Under the environment section, we took the managed image that is provided by AWS, operating system as Ubuntu, runtime as standard, image as standard 2.0 version, image version as always latest, environment type as Linux and created new service role for the code build.

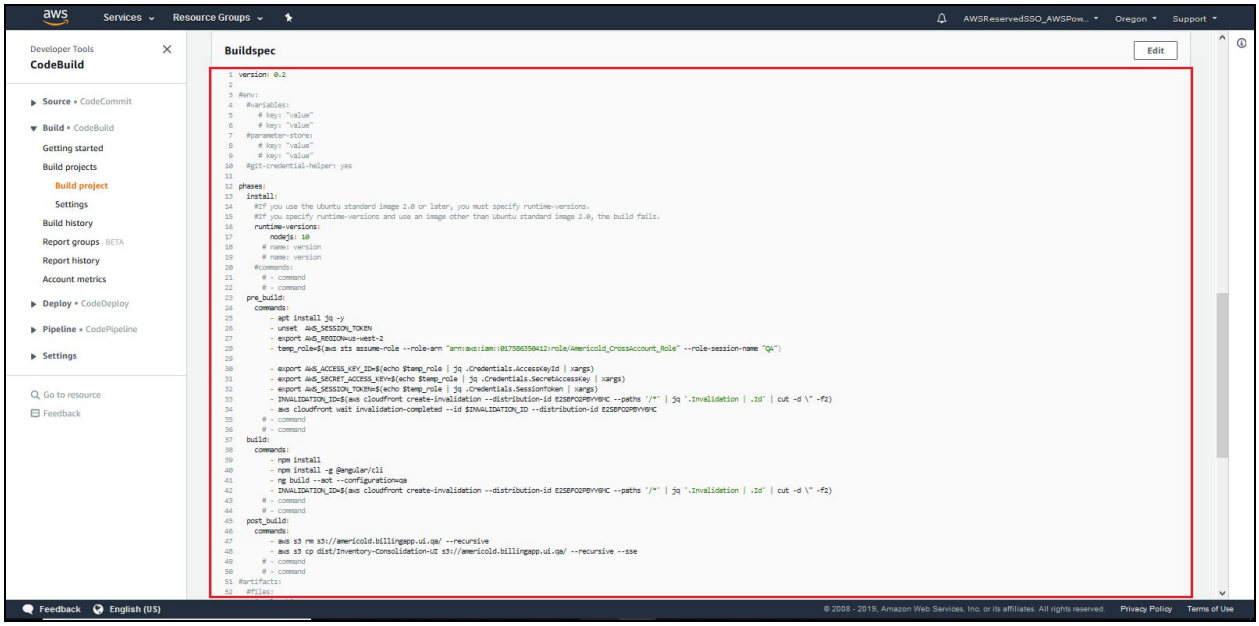




Under the build spec section, choose an option as **insert build commands** to edit the **buildspec.yml** file from console. buildspec.yml file has a set of commands for integration and deployment.



Configure the respective stages as per requirement in the **buildspec.yml file** as shown below.



Below is the step by step explanation for the above script:

Install stage:

- **runtime-versions:**

  nodejs: 10: It defines the runtime environment and version for the application

Build Stage:

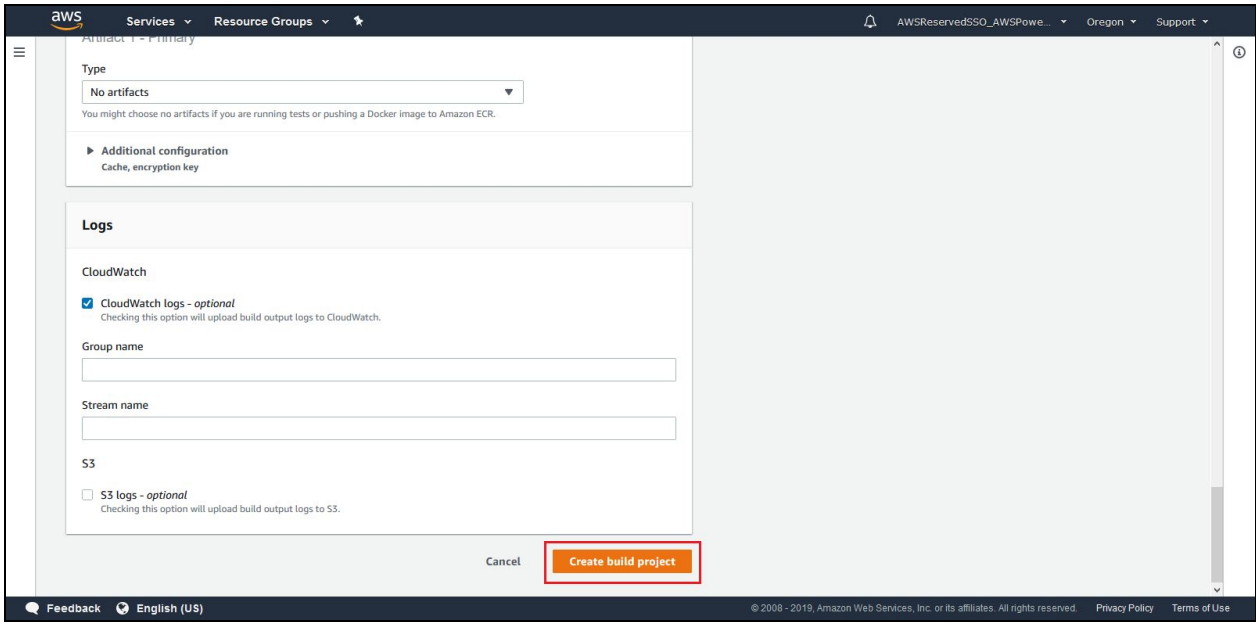
- **apt install jq -y:** Install JQ modules using the following command
- **unset AWS_SESSION_TOKEN:** Clear the AWS session token
- **export AWS_REGION=us-east-1:** Set the AWS region
- **temp_role=$(aws sts assume-role --role-arn "arn:aws:iam::XXXXXXXXXX:role/" --role-session-name "QA"):** Create IAM role for cross account in destination account and give the role ARN as temp_role.
- **export AWS_ACCESS_KEY_ID=$(echo $temp_role | jq.Credentials.AccessKeyId | xargs)**
  export AWS_SECRET_ACCESS_KEY=$(echo $temp_role | jq.Credentials.SecretAccessKey | xargs)
  export AWS_SESSION_TOKEN=$(echo $temp_role | jq .Credentials.SessionToken | xargs) :
  Set the AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY, AWS_SESSION_TOKEN variables with their respective values using jq tool as shown below
- **INVALIDATION_ID=$(aws cloudfront create-invalidation --distribution-id E2SBFO2PBYY6MC --paths '/*' | jq '.Invalidation | .Id' | cut -d \" -f2) :**
  Create invalidation for cloudfront distribution to clear cache
- **aws cloudfront wait invalidation-completed --id $INVALIDATION_ID --distribution-id E2SBFO2PBYY6MC:** Wait for invalidation to be completed
- **npm install:** Install node modules using the following command
- **npm install -g @angular/cli:** Install angular cli using the following command

- **ng build --aot --configuration=qa:** Generate build artifact using the following command
- **INVALIDATION_ID=$(aws cloudfront create-invalidation --distribution-id E3GILM5AHJJTX5 --paths '/*' | jq '.Invalidation | .Id' | cut -d \" -f2)**
- **--distribution-id E3GILM5AHJJTX5:** Create invalidation for cloudfront distribution to clear cache

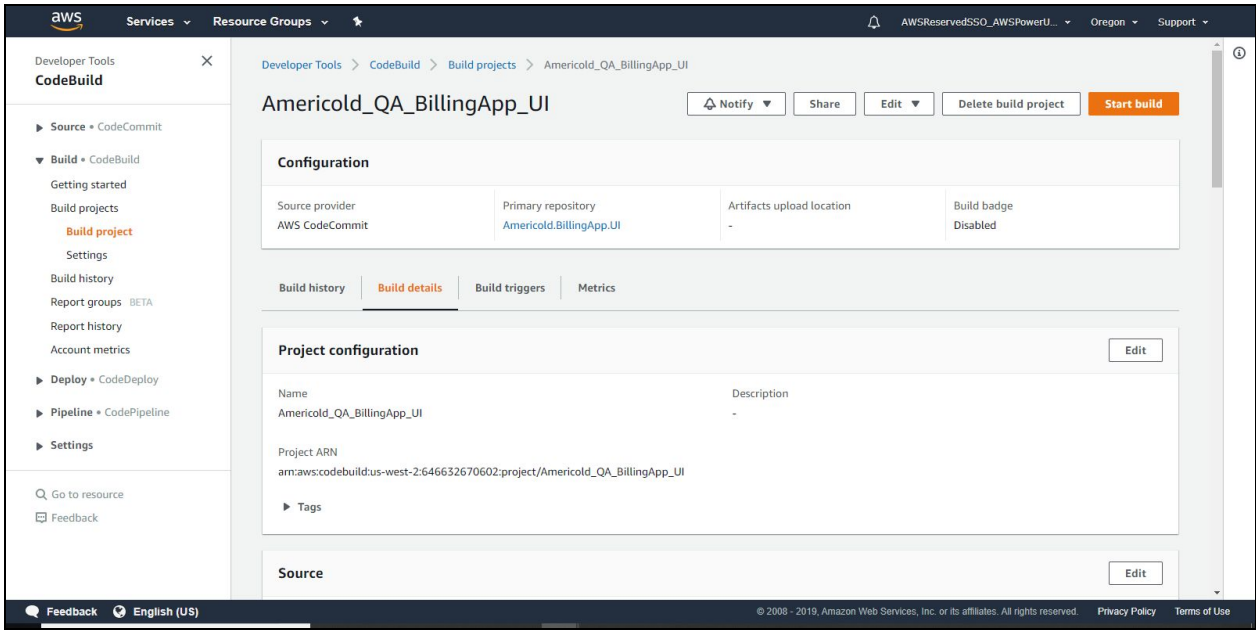
Post Build:

- **aws s3 rm s3://americold.billingapp.ui.qa/ --recursive:** To remove the billing app UI artifacts in S3
- **aws s3 cp dist/Inventory-Consolidation-UI s3://americold.billingapp.ui.qa/ --recursive --sse:** To copy dist folder to S3 bucket

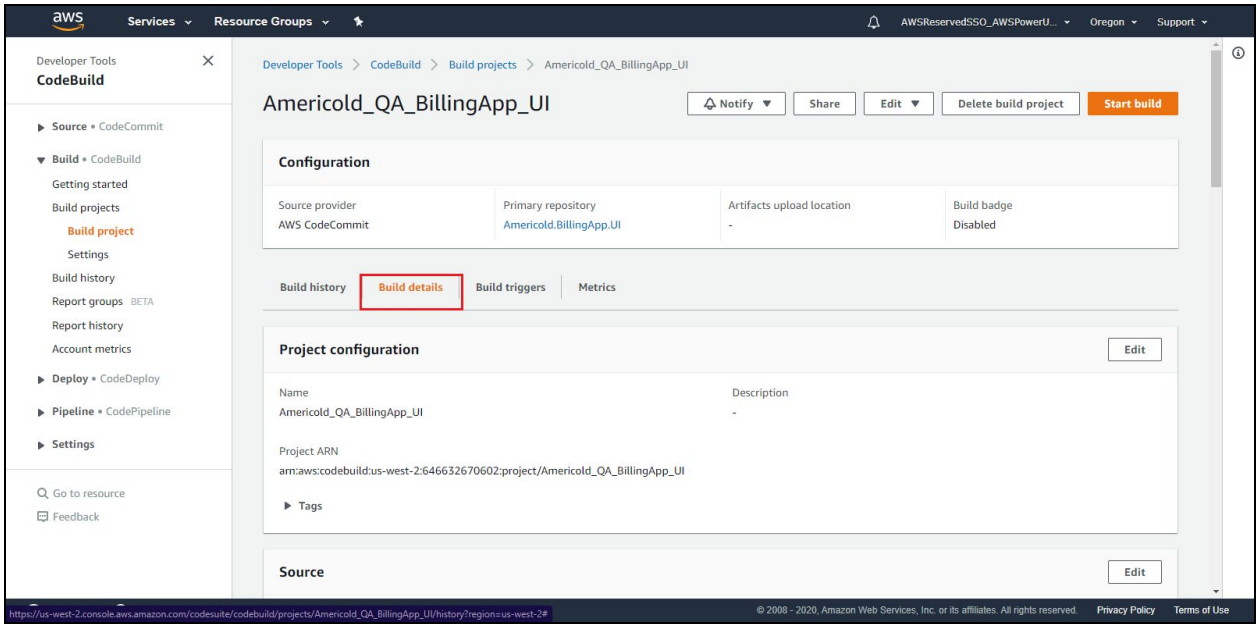
We are enabling cloud watch logs to check the build logs. Once all the configurations were done, click on **create build project.**



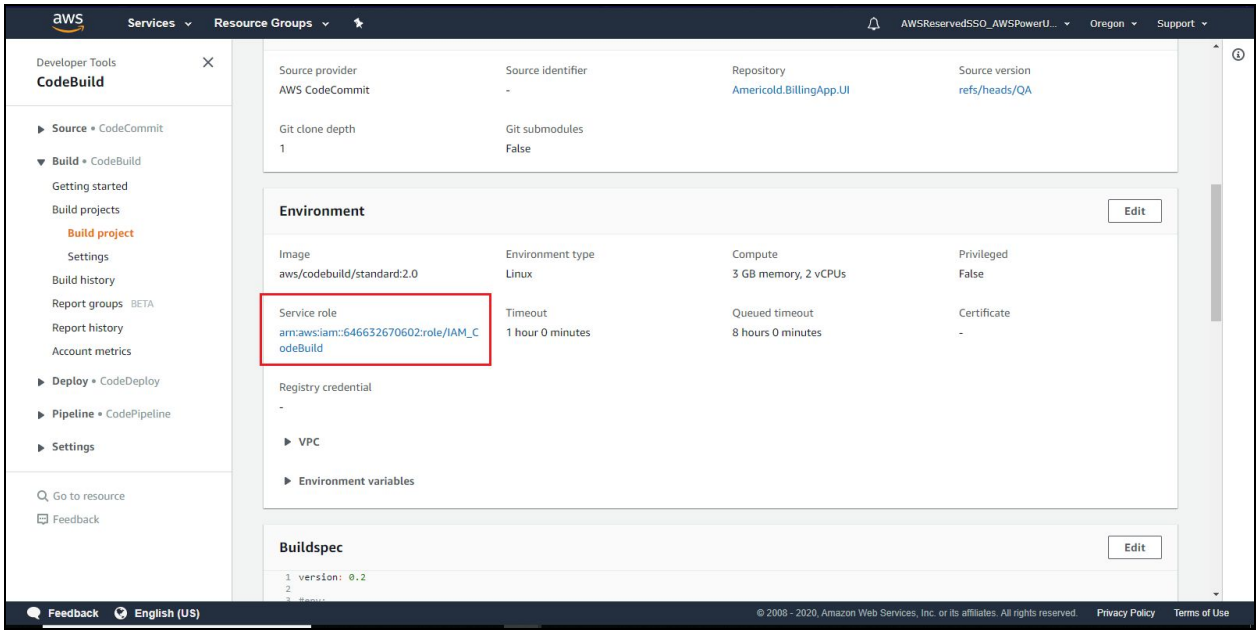
The new build project has been created and the dashboard is shown below.



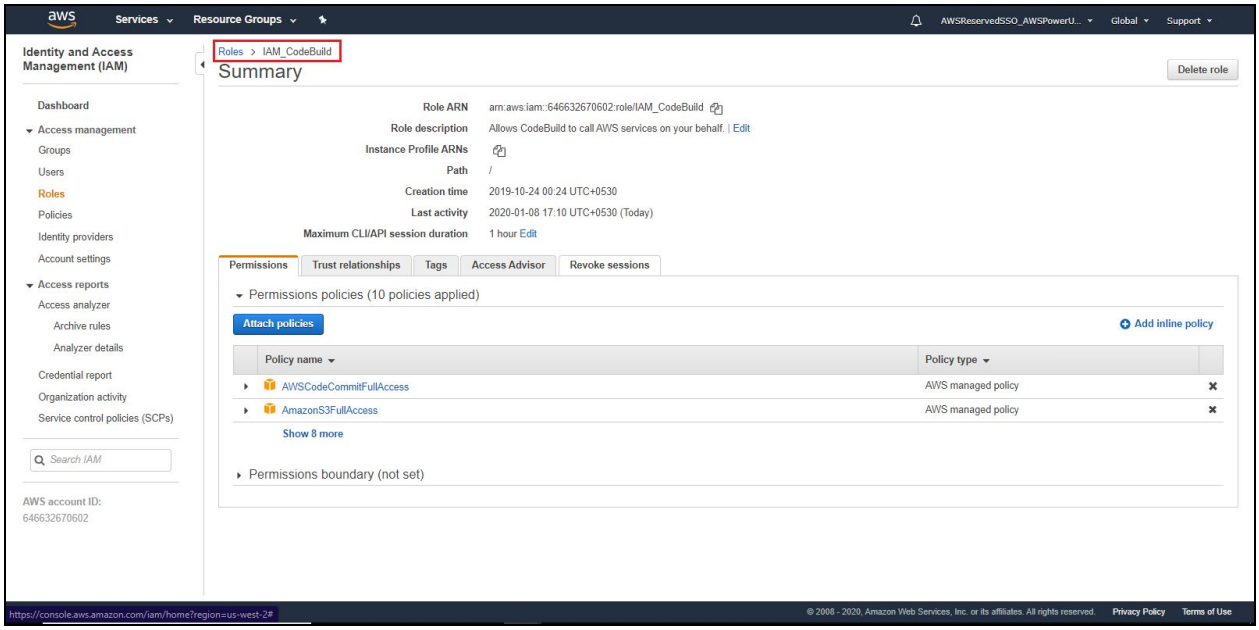
Now goto details and goto **environment section** in build details.



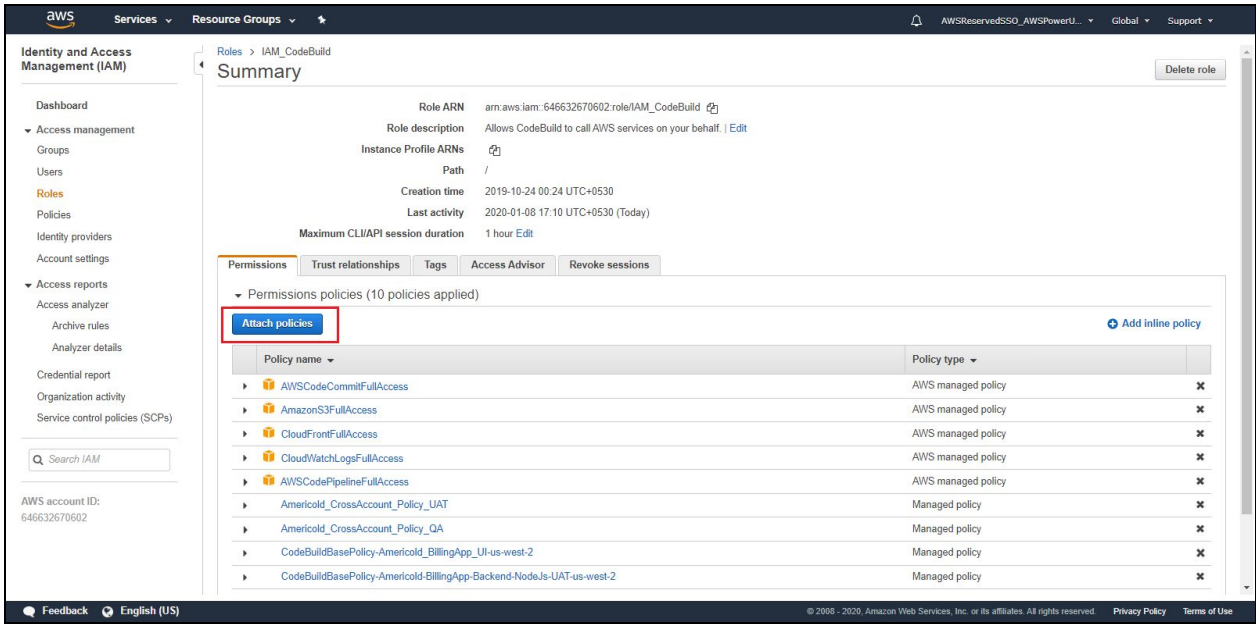
Now click on service role which is a new service role created while configuring code build. It will redirect us to the **IAM console**.



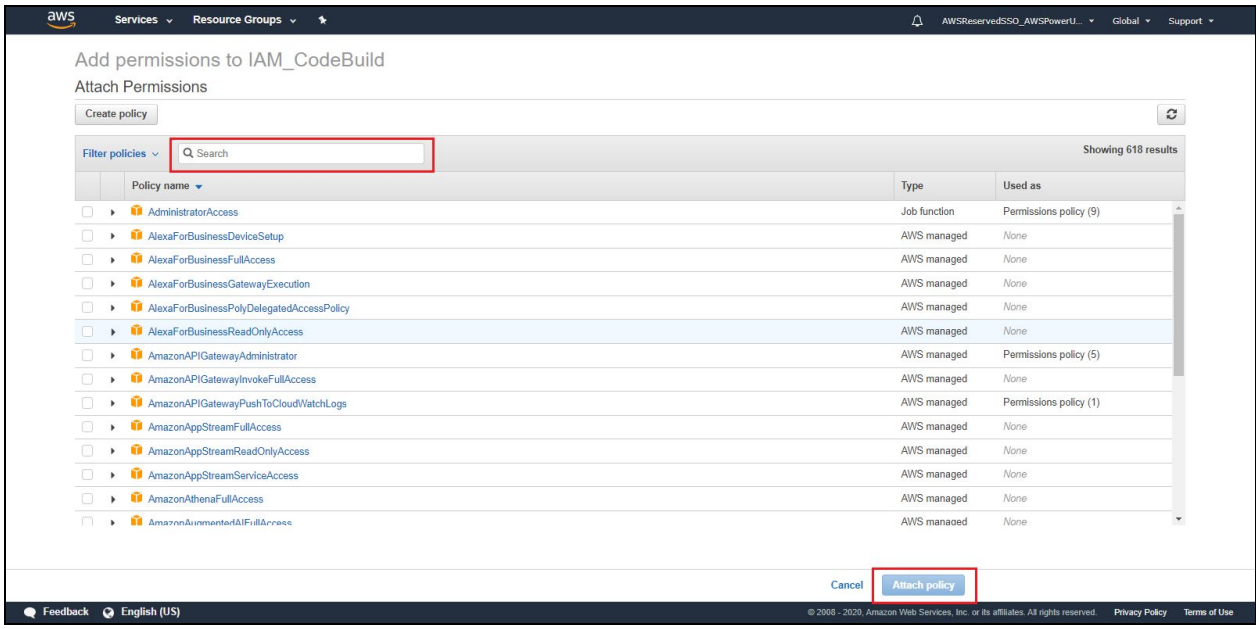
Now the IAM role dashboard will be as shown below and we need to add the required policies to access the AWS services for application deployment through code build.



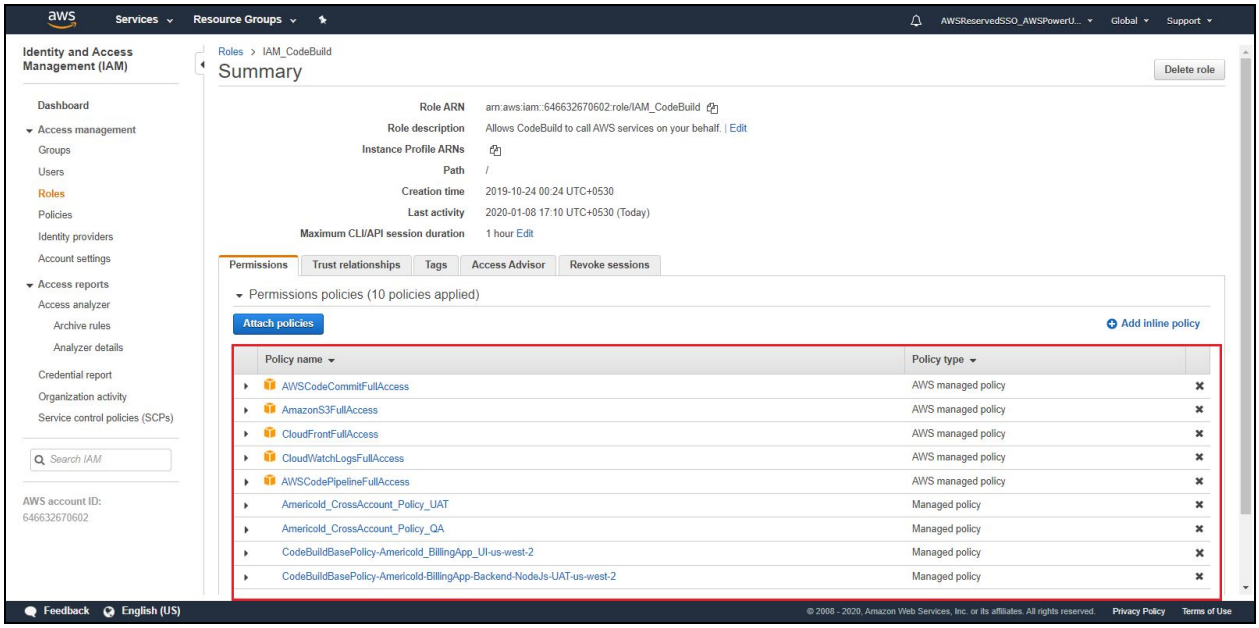
Now click on **Attach policies** to add the policies for the IAM Role.



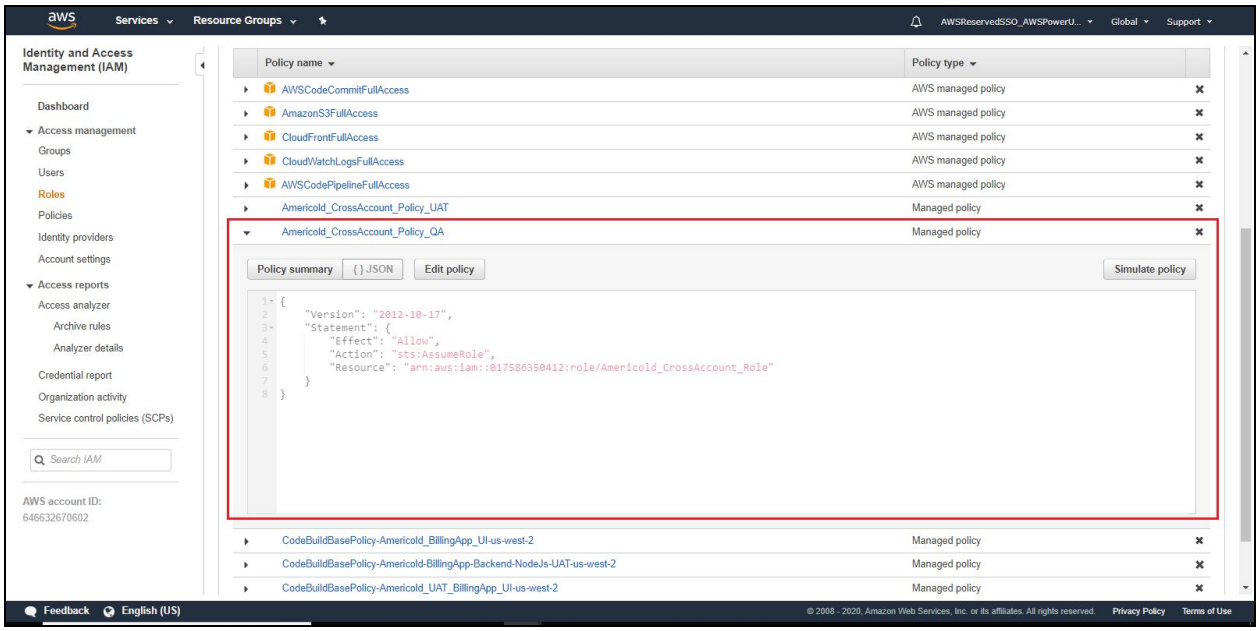
Now search for the required policy and select the check boxes and click on Attach Policies.



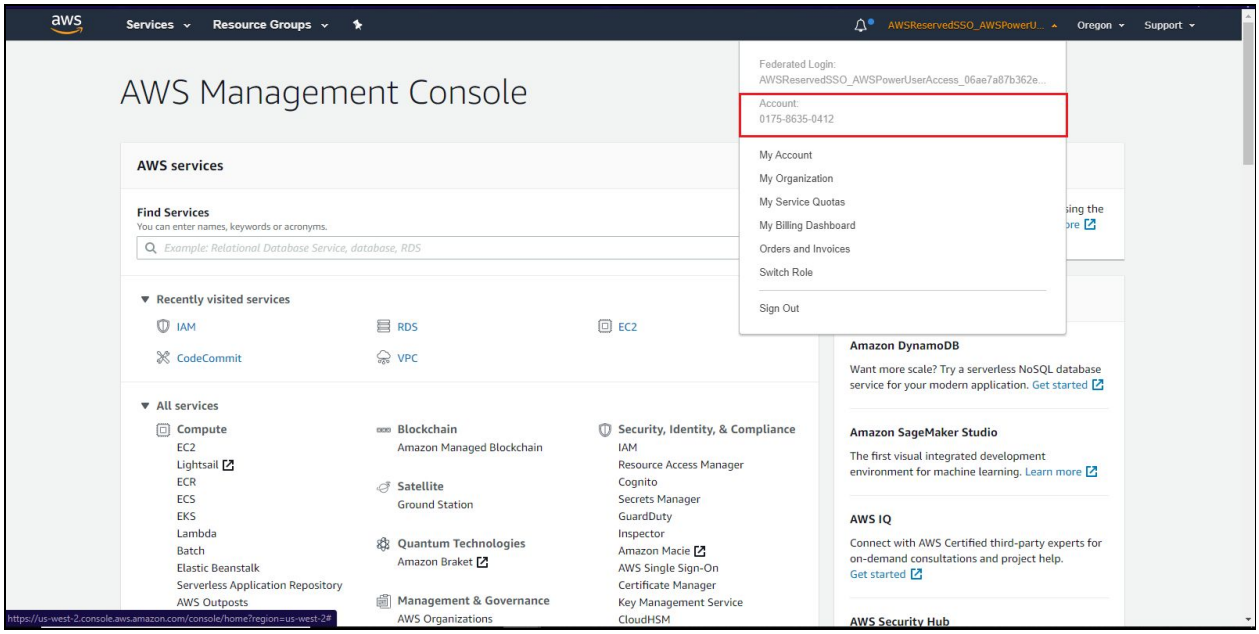
Here we have attached the policies for **Code commit, S3, cloud front, cloud watch logs, code pipeline, and Cross account policy.**



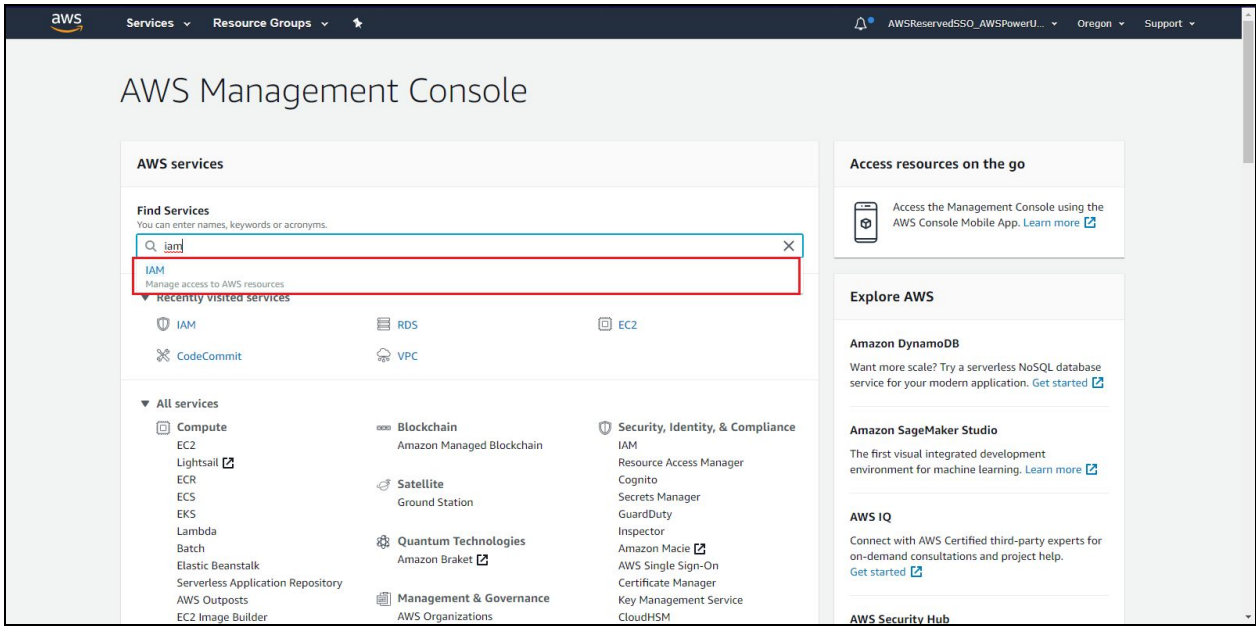
Here we have attached the QA environment account assume role policy to the code build role to provide the necessary permissions to deploy the code into the QA environment.



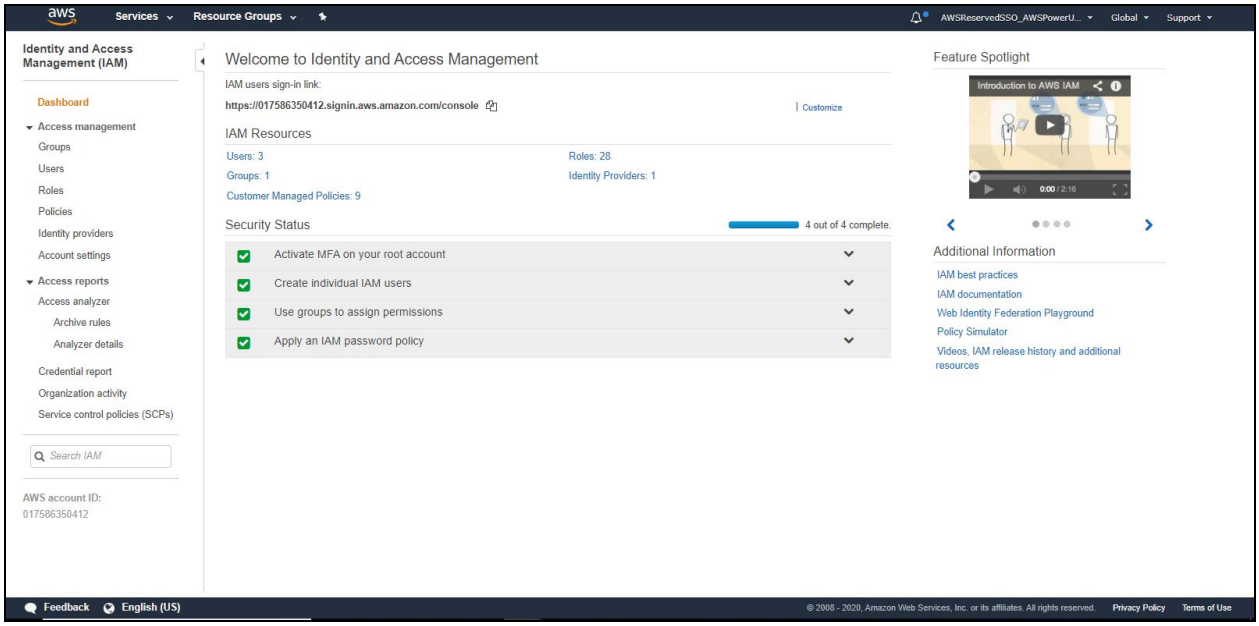
Login into the AWS QA environment account and we can verify the account number of QA environment with the policy attached in the dev environment.



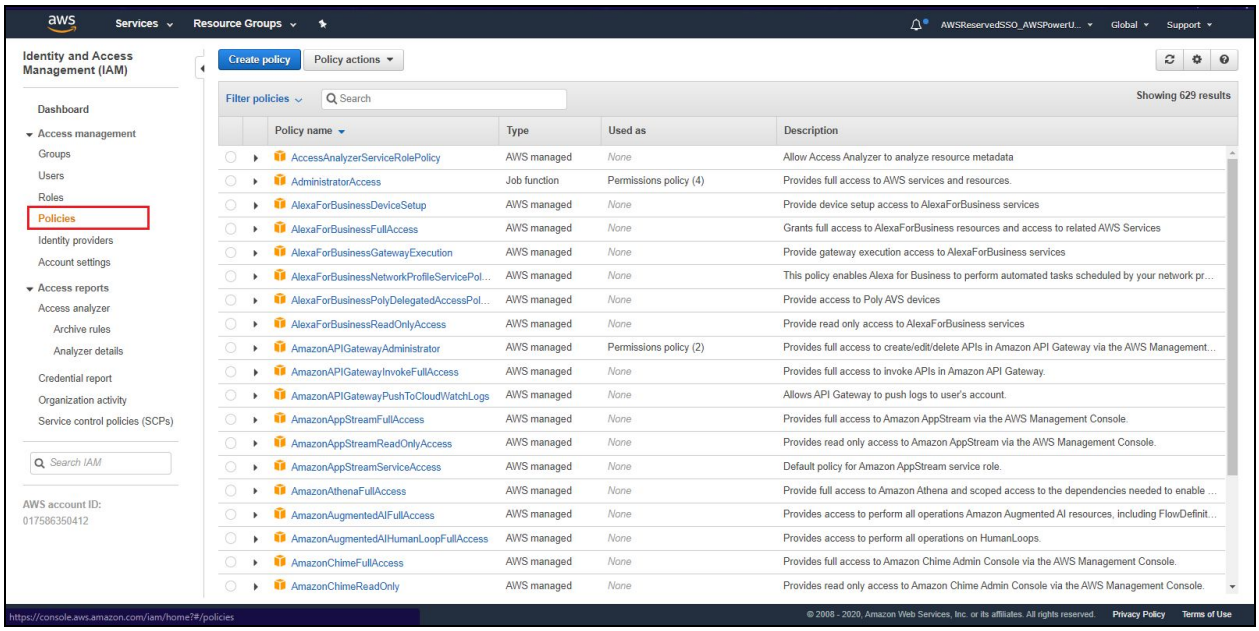
After login into the QA environment goto AWS IAM service



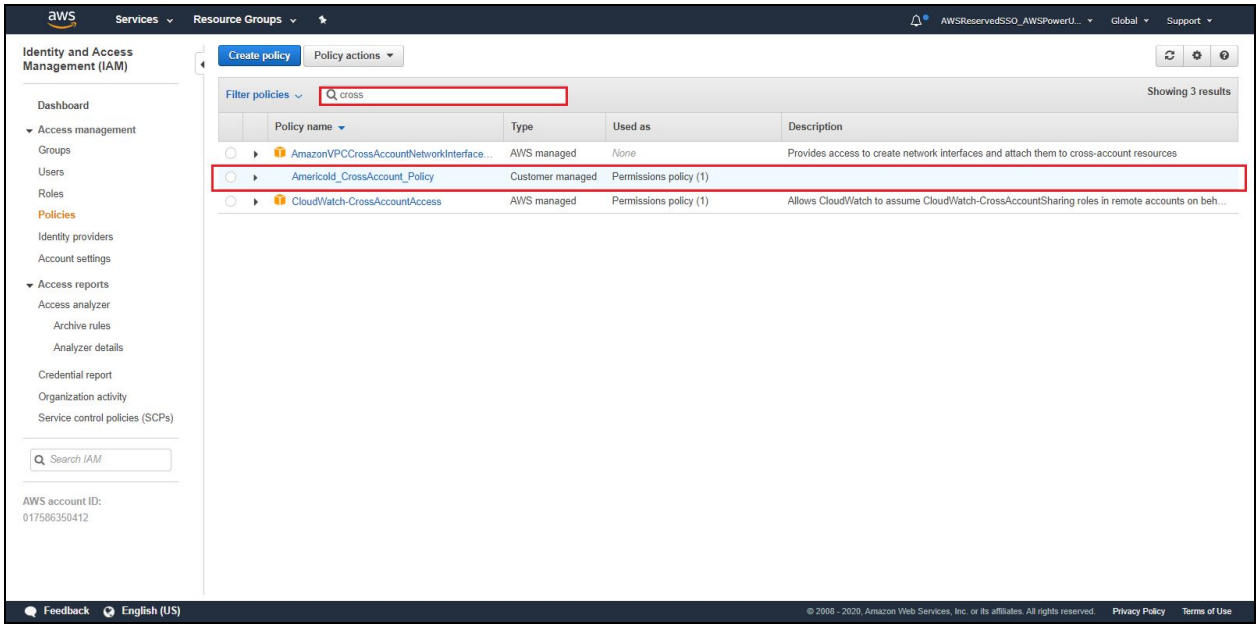
The dashboard of the AWS IAM service will be as shown below



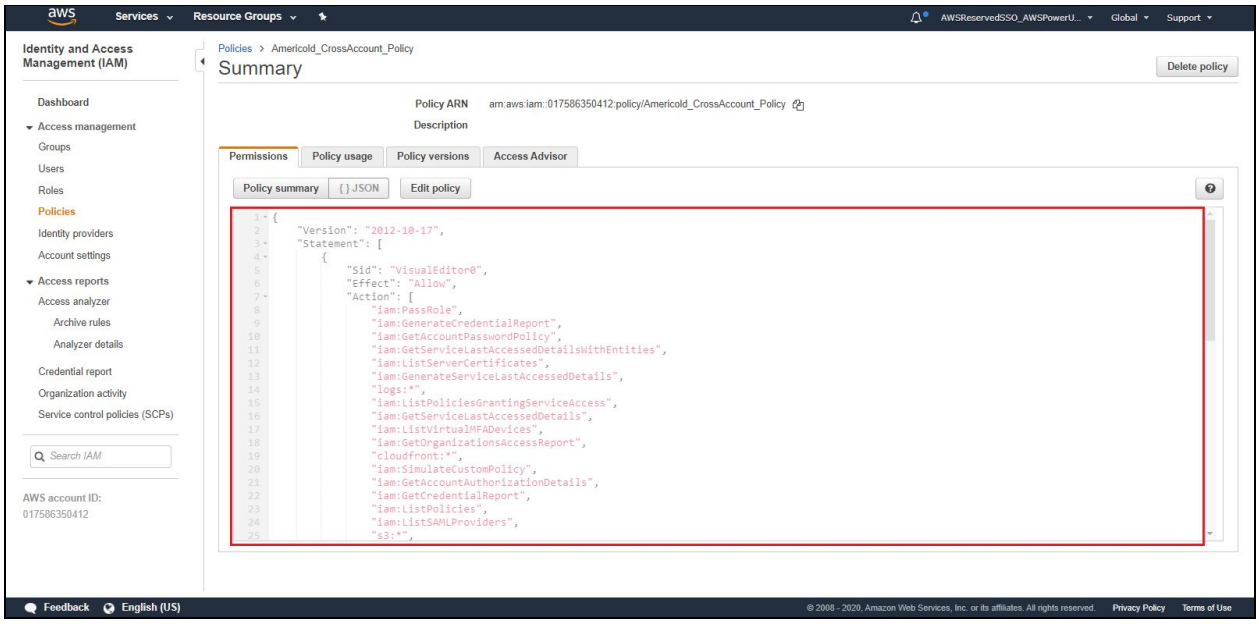
Goto policies section



Here we can see the cross-account access policy which is created previously.

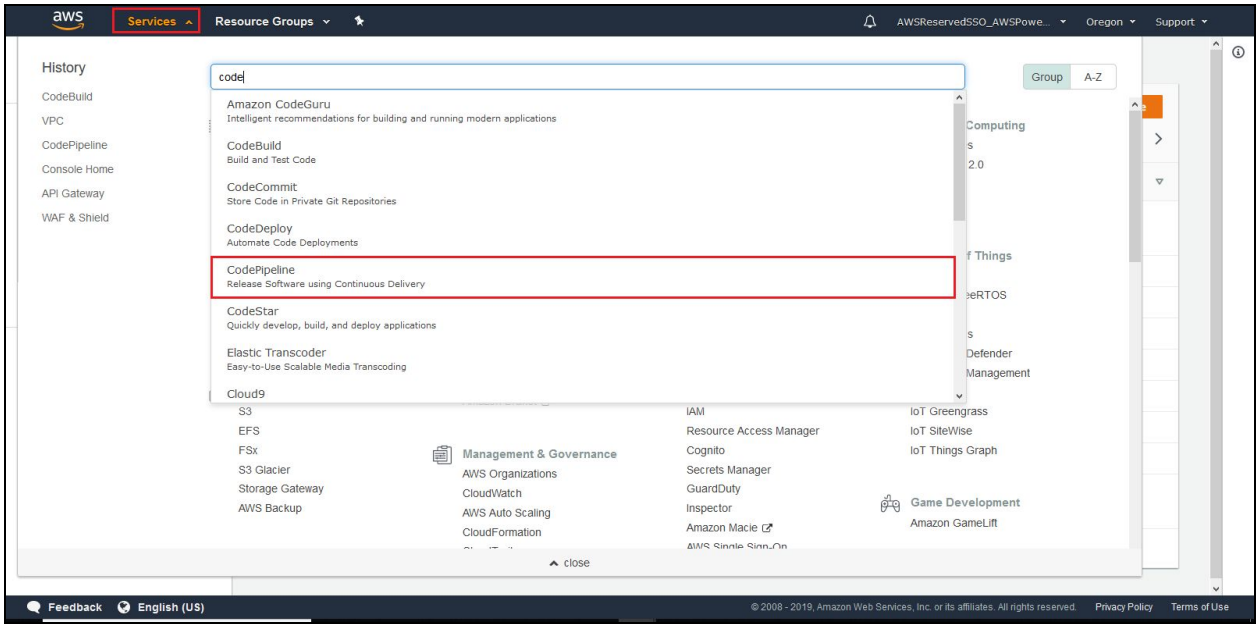


Here we can check the permissions given to the policy which are required for the deployment

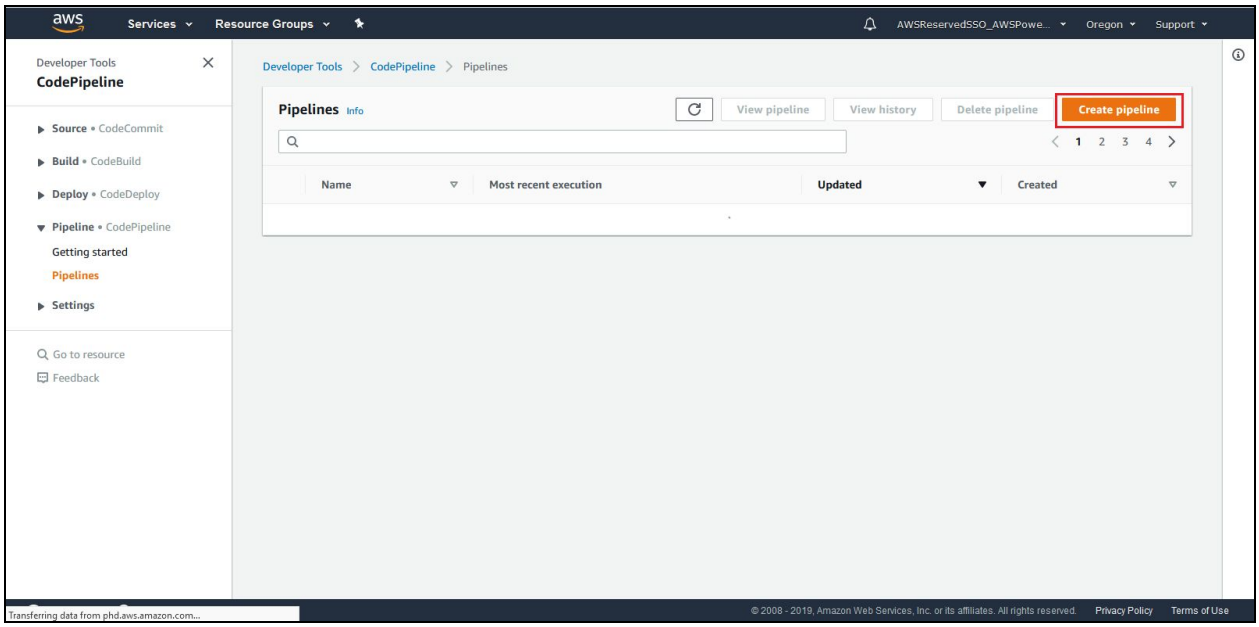


We are going to automate the process for integration and deployment using the code pipeline.

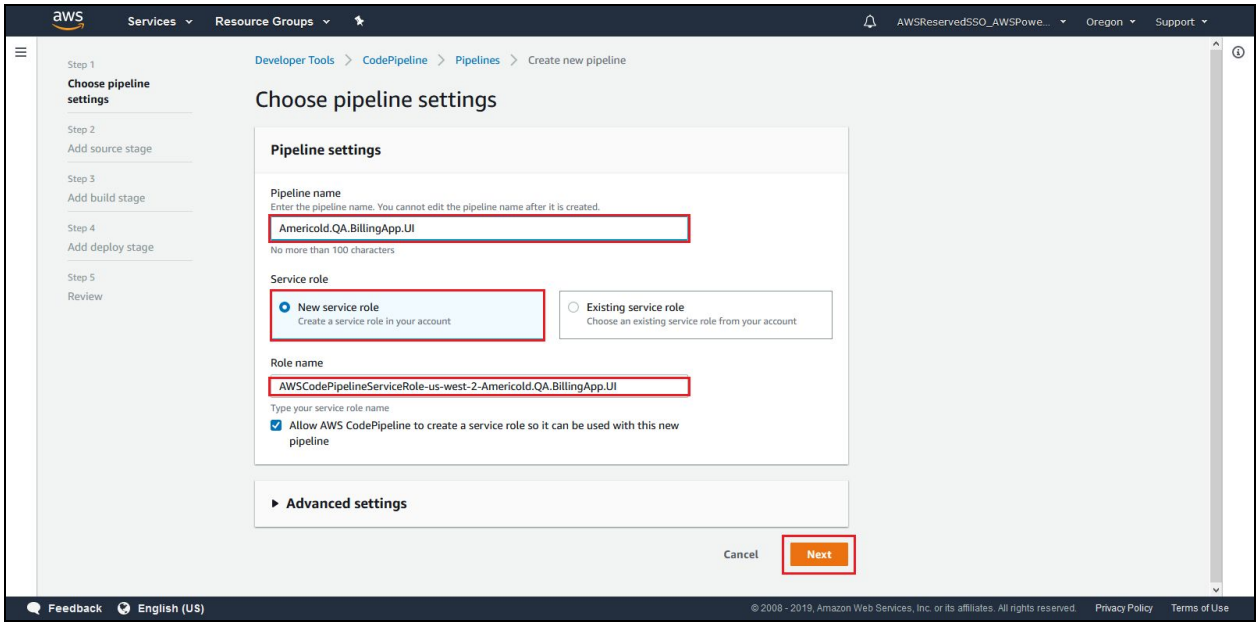
Open AWS console and select code pipeline service.



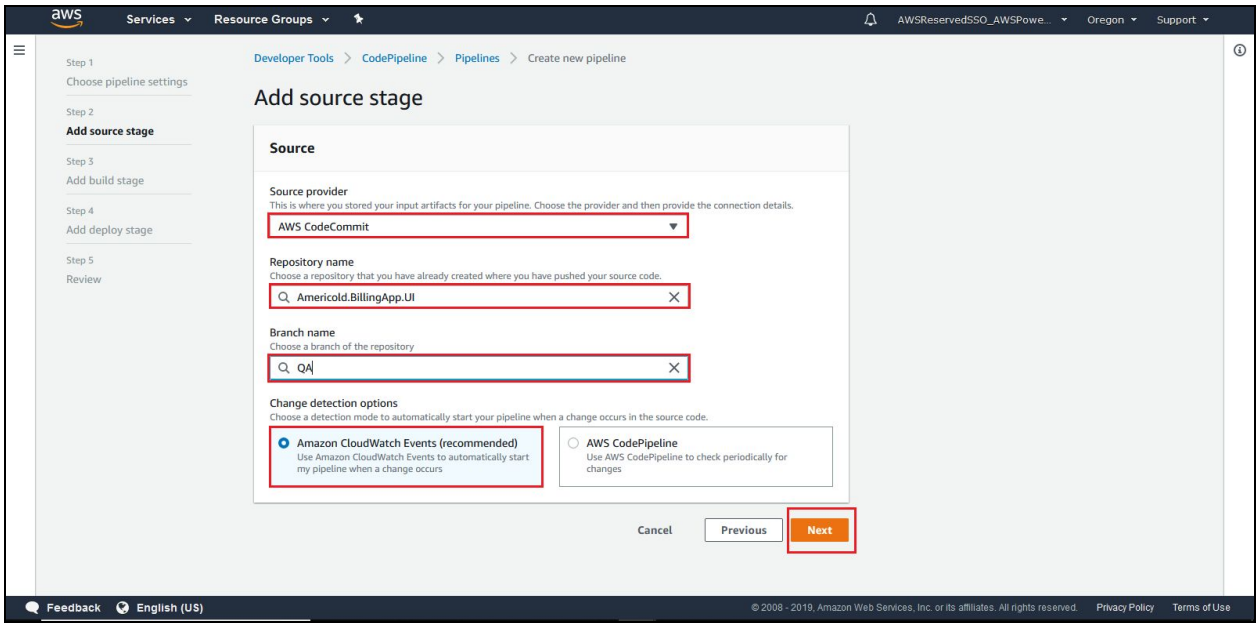
In the code pipeline dashboard, Goto pipelines and click on **create pipelines**.



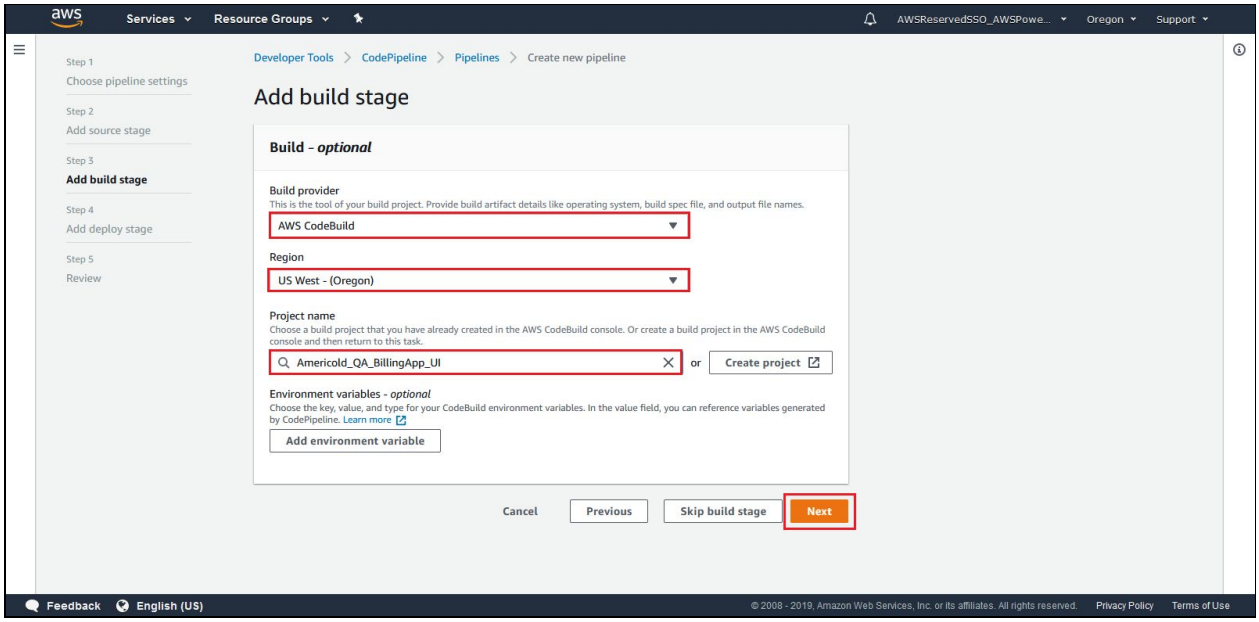
Enter the name of the pipeline and choose the service role as a new service role and click on **Next** as shown below.



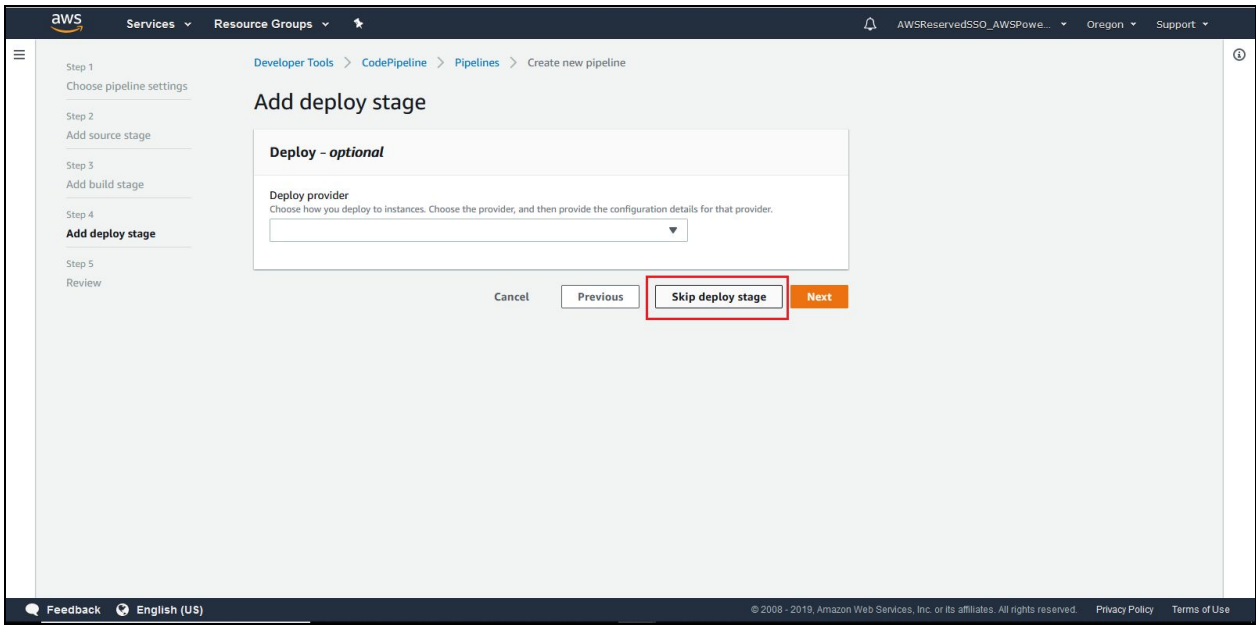
Under add source stage, select the source provider as **AWS CodeCommit,** select the name of the Repository and Branch name. Choose the change detection option as amazon cloudwatch events to enable cloudwatch logs for the pipeline and click on **Next**.



Under add build stage, select the build provider as **AWS code build**, region as **US West – (Oregon),** select the build project name that we created in the above steps and click on **Next** as shown below.

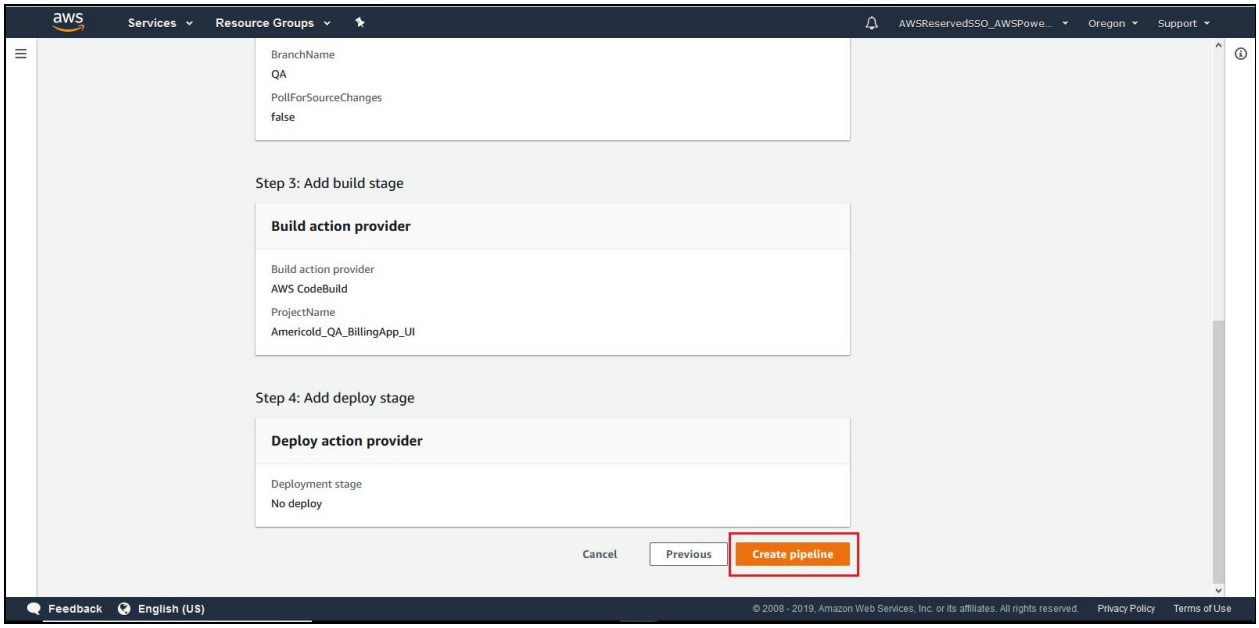


We are deploying through the commands. So, we can skip the deploy stage.

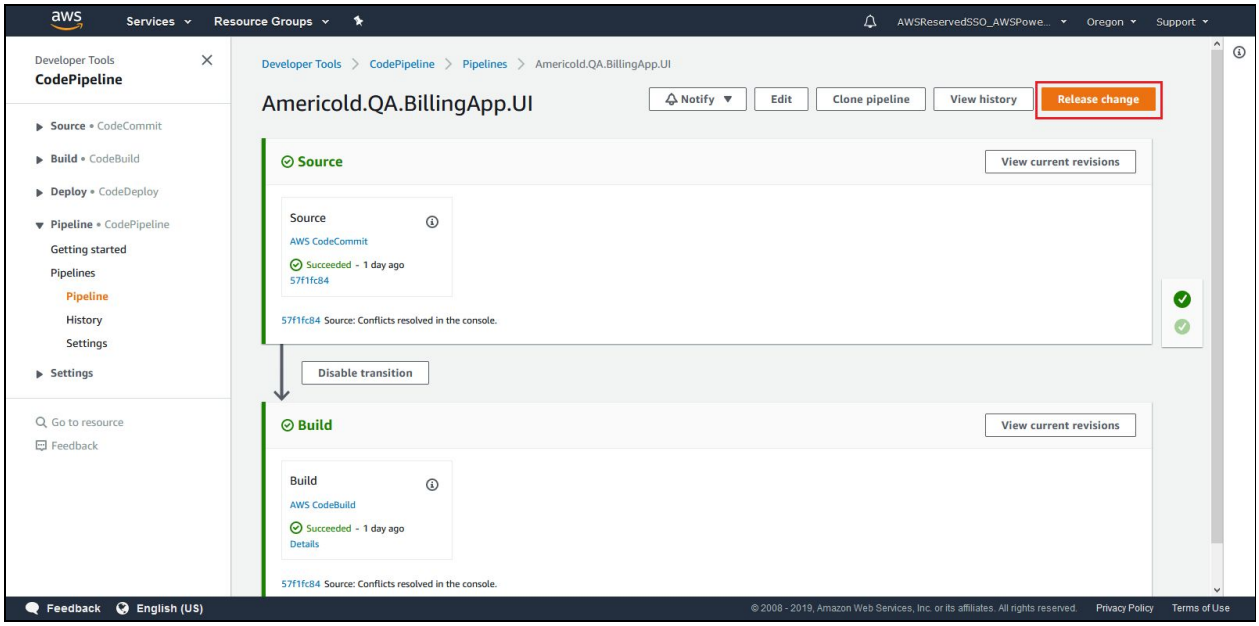


Give the Confirmation as **Skip**

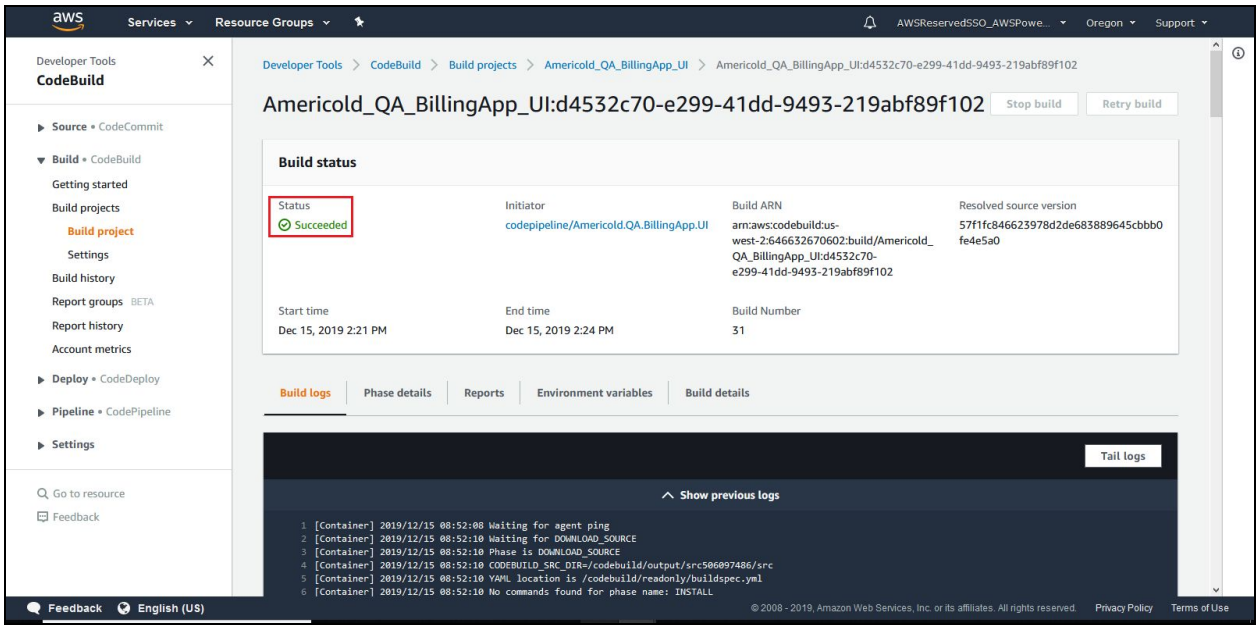
Review the pipeline configuration and click on create pipelines.

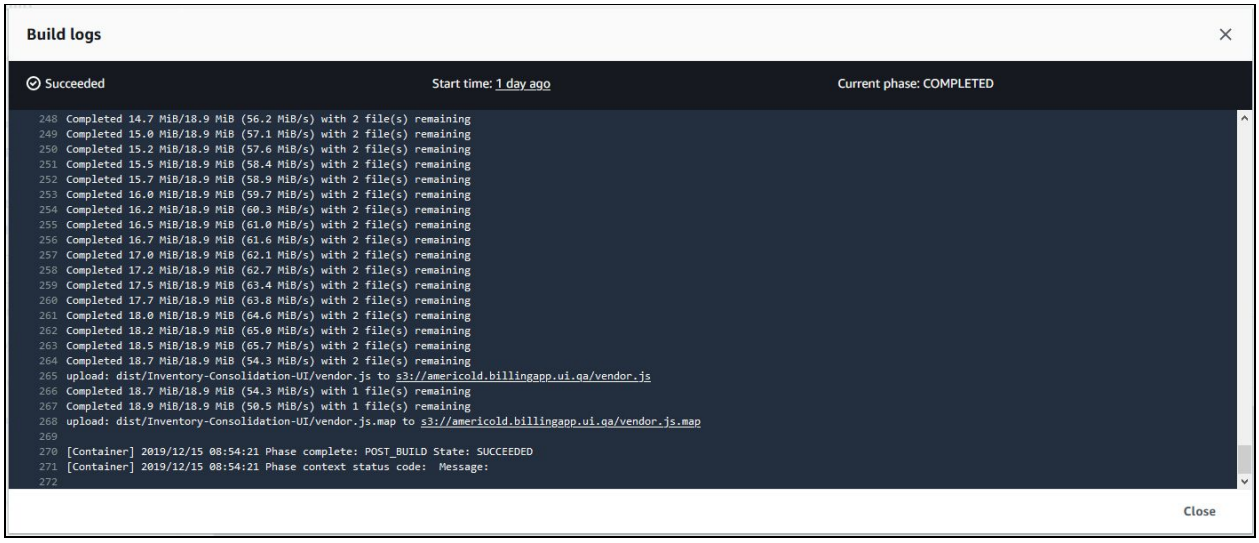


We can see the process is automated for CI/CD. We can check the stage logs by clicking on the details. The pipeline will trigger automatically for the first time to release changes and later it will be triggered when the new changes are committed to the repository.

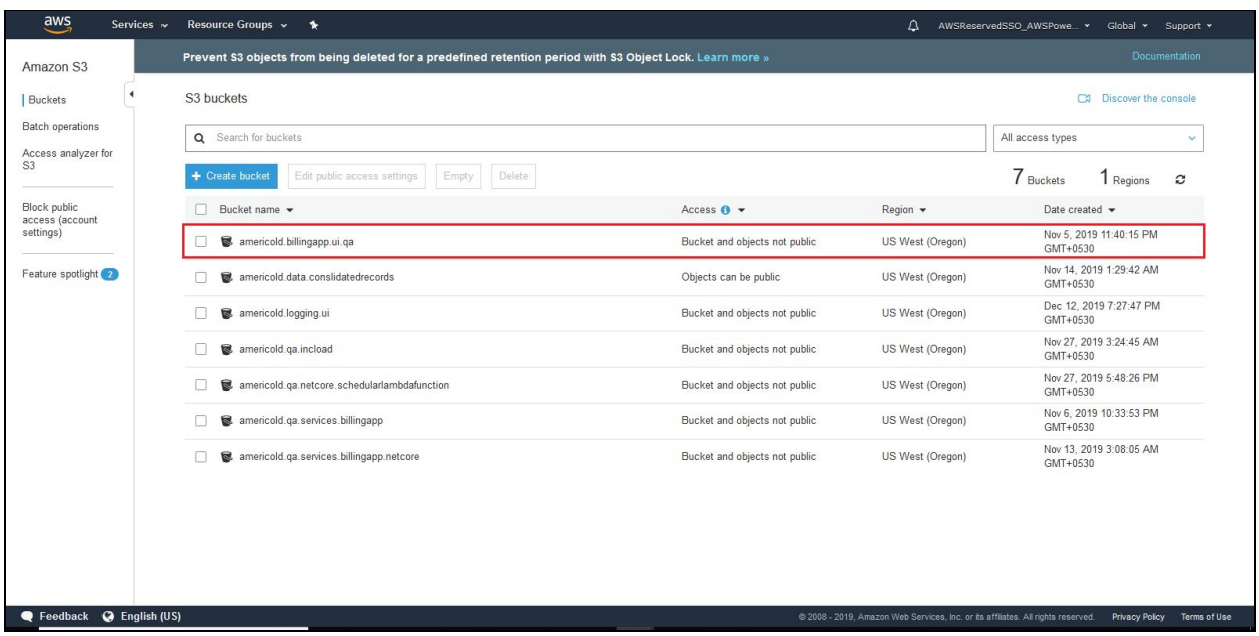


We can check the build logs here.





After completing the CI/CD we can see the updated data in the S3 bucket.



aws Services Resource Groups AWSReservedSSO_AWSPowe... Global Support

Amazon S3 > americold.billingapp.ui.qa

americold.billingapp.ui.qa

Overview | Properties | Permissions | Management | Access points

Type a prefix and press Enter to search. Press ESC to clear.

Upload | Create folder | Download | Actions

US West (Oregon)

Viewing 1 to 14

Name	Last modified	Size	Storage class
assets	--	--	--
index.html	Dec 18, 2019 5:06:48 AM GMT+0530	1.2 KB	Standard
main.js	Dec 18, 2019 5:06:48 AM GMT+0530	1.0 MB	Standard
main.js.map	Dec 18, 2019 5:06:48 AM GMT+0530	259.5 KB	Standard
polyfills-es5.js	Dec 18, 2019 5:06:48 AM GMT+0530	434.2 KB	Standard
polyfills-es5.js.map	Dec 18, 2019 5:06:48 AM GMT+0530	275.3 KB	Standard
polyfills.js	Dec 18, 2019 5:06:48 AM GMT+0530	424.3 KB	Standard
polyfills.js.map	Dec 18, 2019 5:06:48 AM GMT+0530	368.0 KB	Standard
runtime.js	Dec 18, 2019 5:06:48 AM GMT+0530	6.1 KB	Standard
runtime.js.map	Dec 18, 2019 5:06:48 AM GMT+0530	6.1 KB	Standard

Feedback English (US)

 Privacy Policy Terms of Use