Experiment no. 2

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.

Theory :-

AWS CodeBuild is a fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to deploy. With CodeBuild, you don't need to provision, manage, and scale your own build servers. CodeBuild 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 CodeBuild, you are charged by the minute for the compute resources you use.

Benefits

Fully managed build service

AWS CodeBuild eliminates the need to set up, patch, update, and manage your own build servers and software. There is no software to install or manage.

Continuous scaling

AWS CodeBuild scales up and down automatically to meet your build volume. It immediately processes each build you submit and can run separate builds concurrently, which means your builds are not left waiting in a queue.

Pay as you go

With AWS CodeBuild, you are charged based on the number of minutes it takes to complete your build. This means you no longer have to worry about paying for idle build server capacity.

Extensible

You can bring your own build tools and programming runtimes to use with AWS CodeBuild by creating customized build environments in addition to the prepackaged build tools and runtimes supported by CodeBuild.

Enables continuous integration and delivery

AWS CodeBuild belongs to a family of AWS Code Services, which you can use to create complete, automated software release workflows for continuous integration and delivery (CI/CD). You can also integrate CodeBuild into your existing CI/CD workflow. For example, you can use CodeBuild as a worker node for your existing Jenkins server setup for distributed builds.

Secure

With AWS CodeBuild, your build artifacts are encrypted with customer-specific keys that are managed by the AWS Key Management Service (KMS). CodeBuild is integrated with AWS Identity and Access Management (IAM), so you can assign user-specific permissions to your build projects.

AWS CodePipeline

AWS CodePipeline is a continuous delivery service you can use to model, visualize, and automate the steps required to release your software. You can quickly model and configure the different stages of a software release process. CodePipeline automates the steps required to release your software changes continuously.

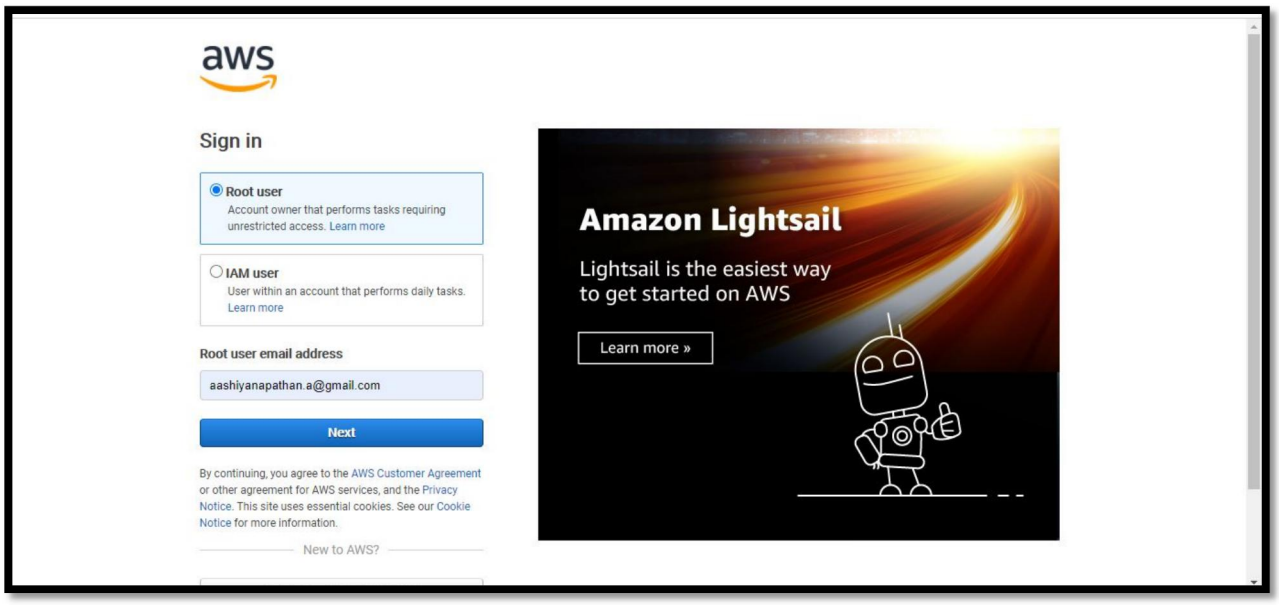
when developers commit changes to a source repository, CodePipeline automatically detects the changes. Those changes are built, and if any tests are configured, those tests are run. After the tests are complete, the built code is deployed to staging servers for testing. From the staging server, CodePipeline runs more tests, such as integration or load tests. Upon the successful completion of those tests, and after a manual approval action that was added to the pipeline is approved, CodePipeline deploys the tested and approved code to production instances.

CodePipeline can deploy applications to EC2 instances by using CodeDeploy, AWS Elastic Beanstalk, or AWS OpsWorks Stacks. CodePipeline can also deploy container-based applications to services by using Amazon ECS. Developers can also use the integration points provided with CodePipeline to plug in other tools or services, including build services, test providers, or other deployment targets or systems.

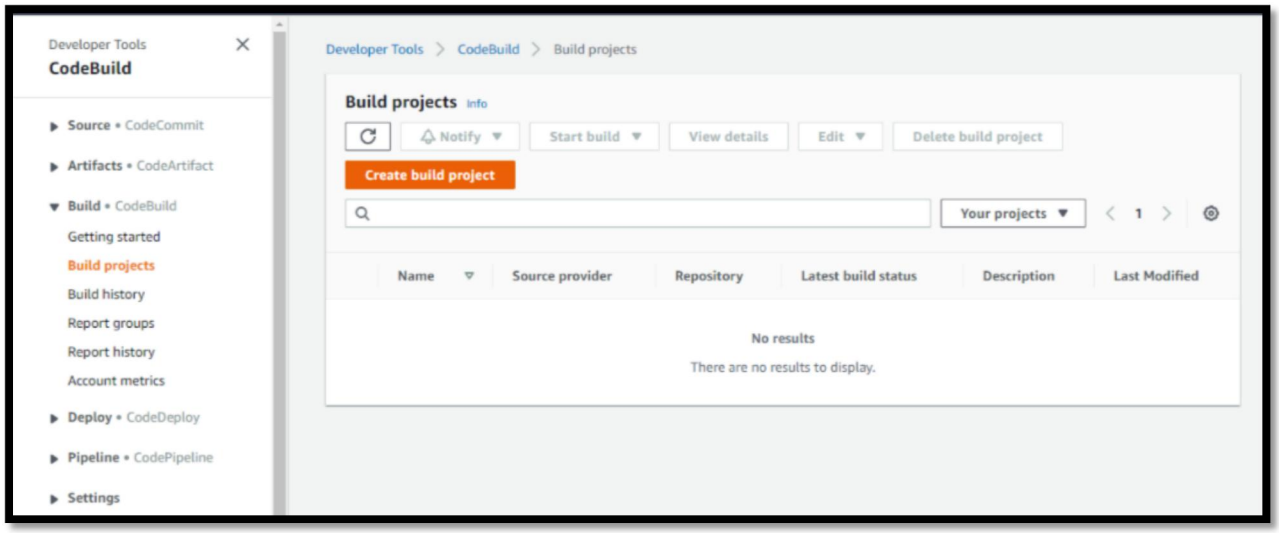
Steps:

1. Login with your AWS account.

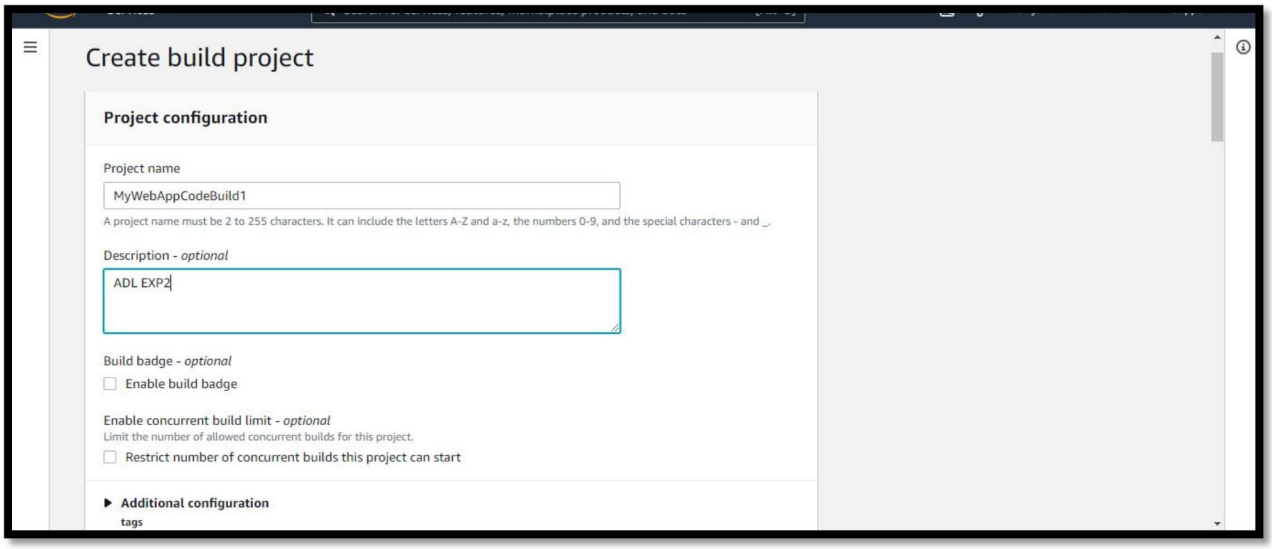
2. Navigate to CodeBuild service from Developer tools section as below:



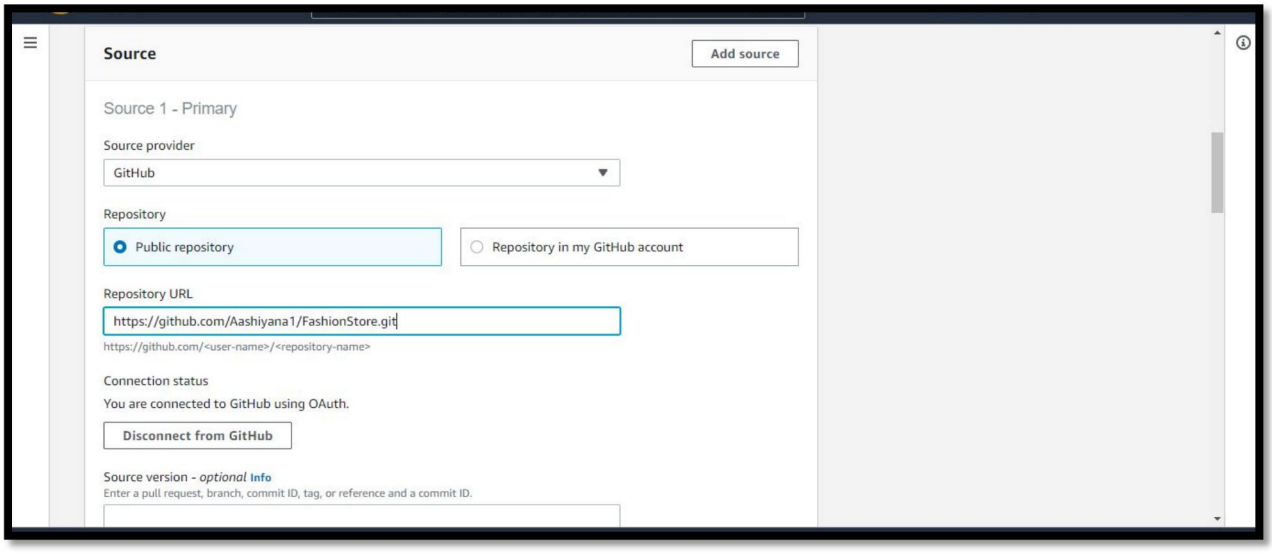
3. Click on Create build project :



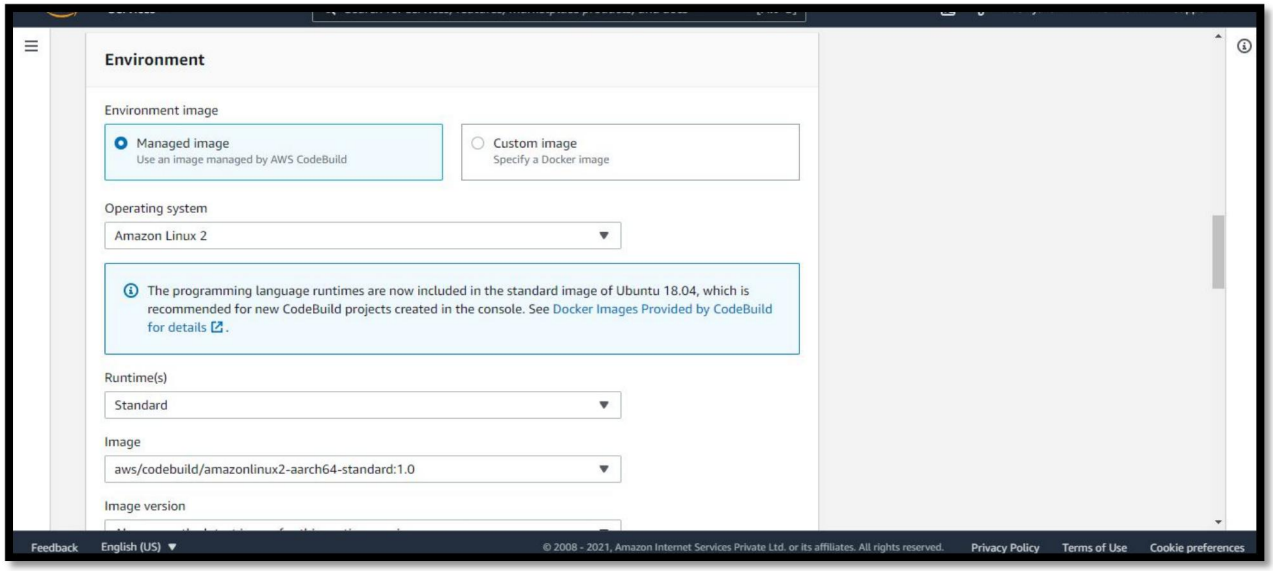
4. Provide the name for the build project(MyWebAppCodeBuild) and click on Next.

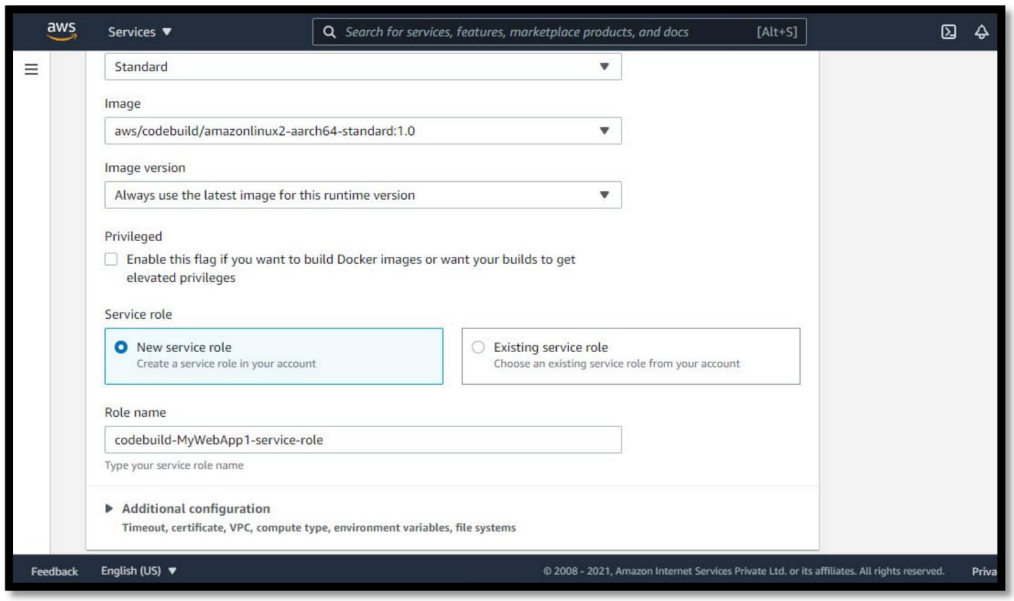


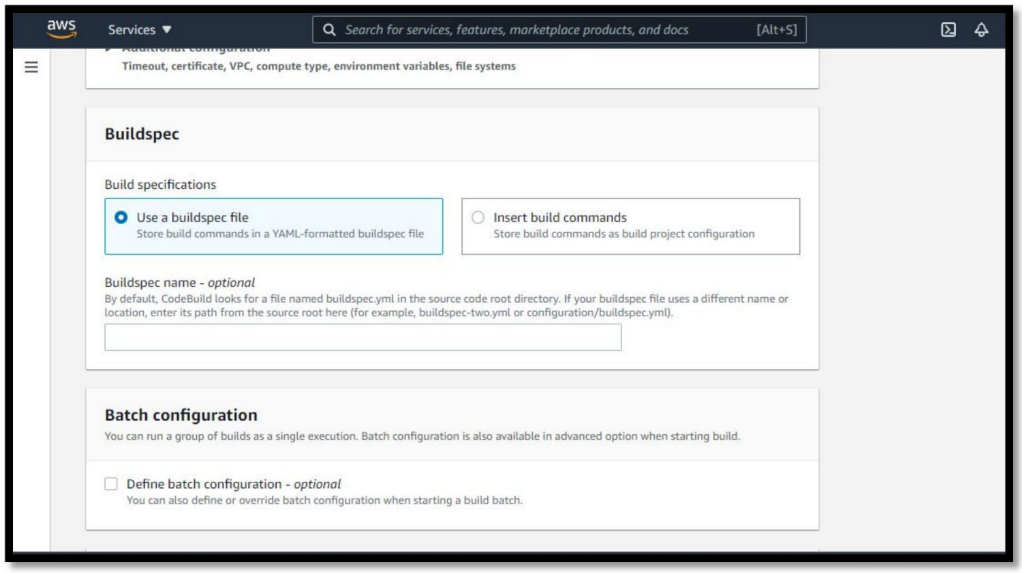
5. Add Source

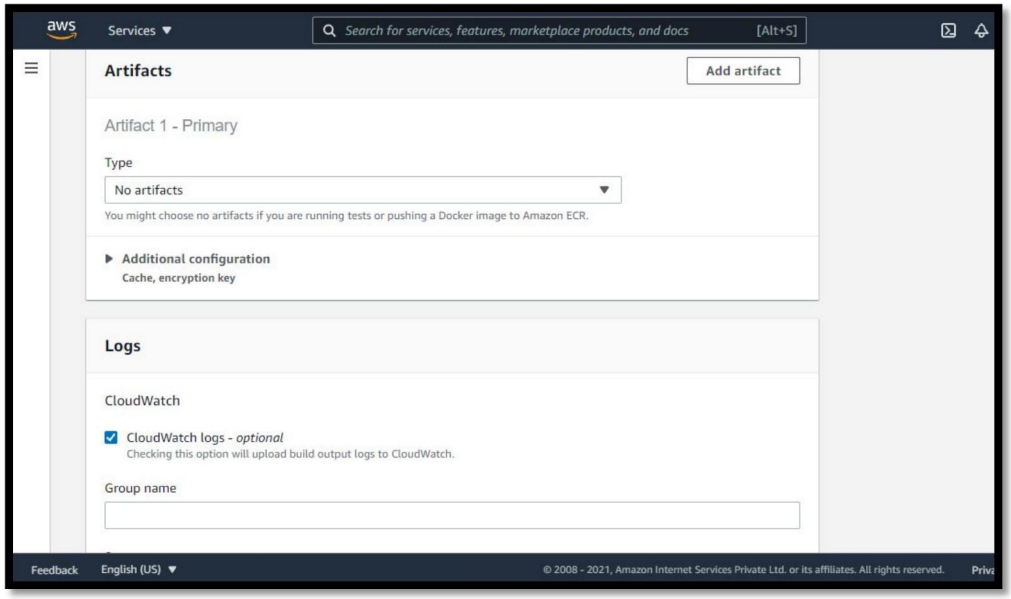


5. Add Environment image

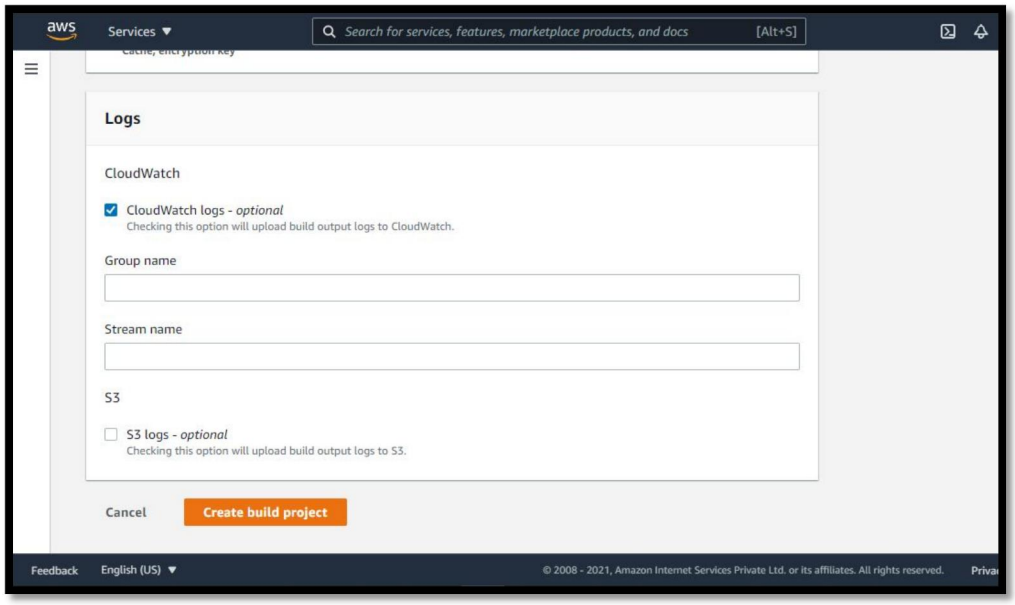

Environment
Environment image
Managed image
Use an image managed by AWS CodeBuild
Custom image
Specify a Docker image
Operating system
Amazon Linux 2
The programming language runtimes are now included in the standard image of Ubuntu 18.04, which is recommended for new CodeBuild projects created in the console. See Docker Images Provided by CodeBuild for details.
Runtime(s)
Standard
Image
aws/codebuild/amazonlinux2-aarch64-standard:1.0
Image version
Feedback
English (US)
© 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

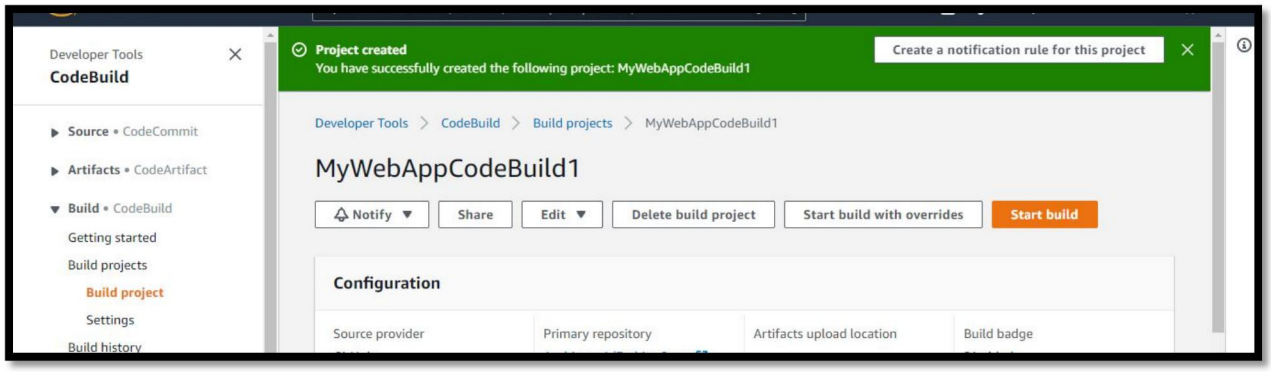

aws
Services
Search for services, features, marketplace products, and docs
[Alt+S]
Standard
Image
aws/codebuild/amazonlinux2-aarch64-standard:1.0
Image version
Always use the latest image for this runtime version
Privileged
Enable this flag if you want to build Docker images or want your builds to get elevated privileges
Service role
New service role
Create a service role in your account
Existing service role
Choose an existing service role from your account
Role name
codebuild-MyWebApp1-service-role
Type your service role name
Additional configuration
Timeout, certificate, VPC, compute type, environment variables, file systems
Feedback
English (US)
© 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.



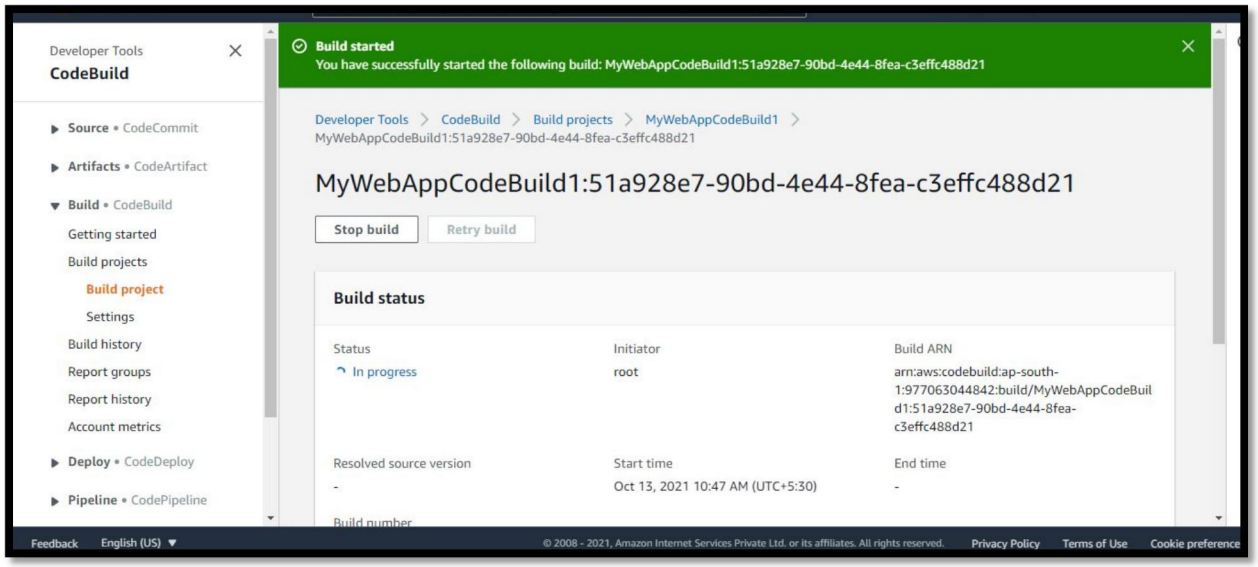


7. Click on Create build project





8. Click on Start build

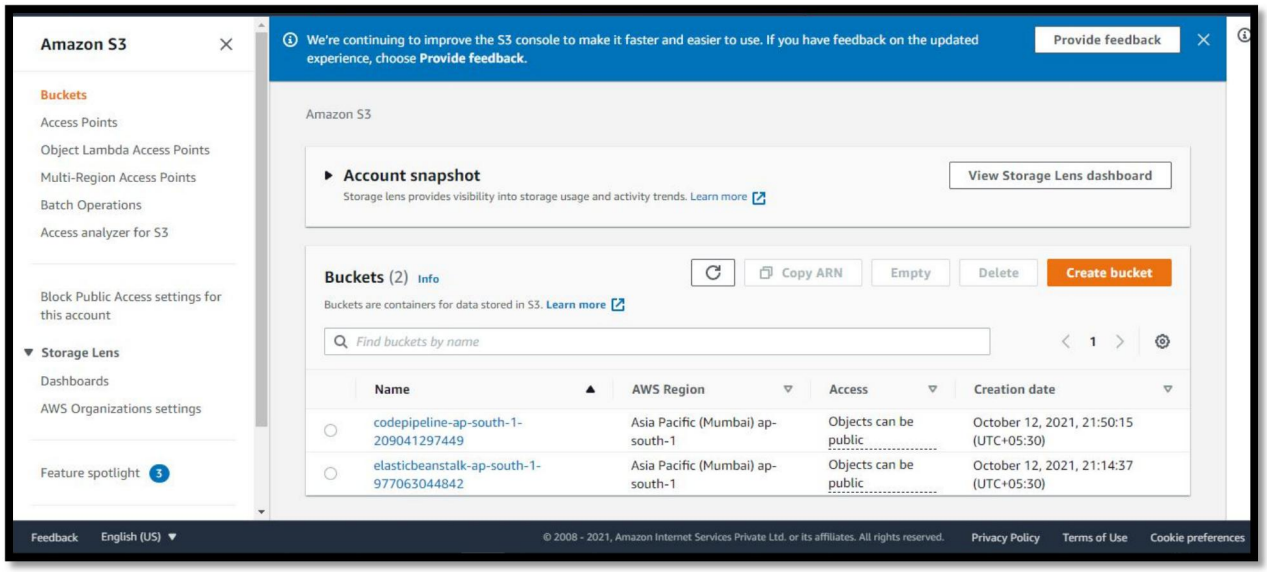


Deploy on S3 / SEBS using AWS CodePipeline

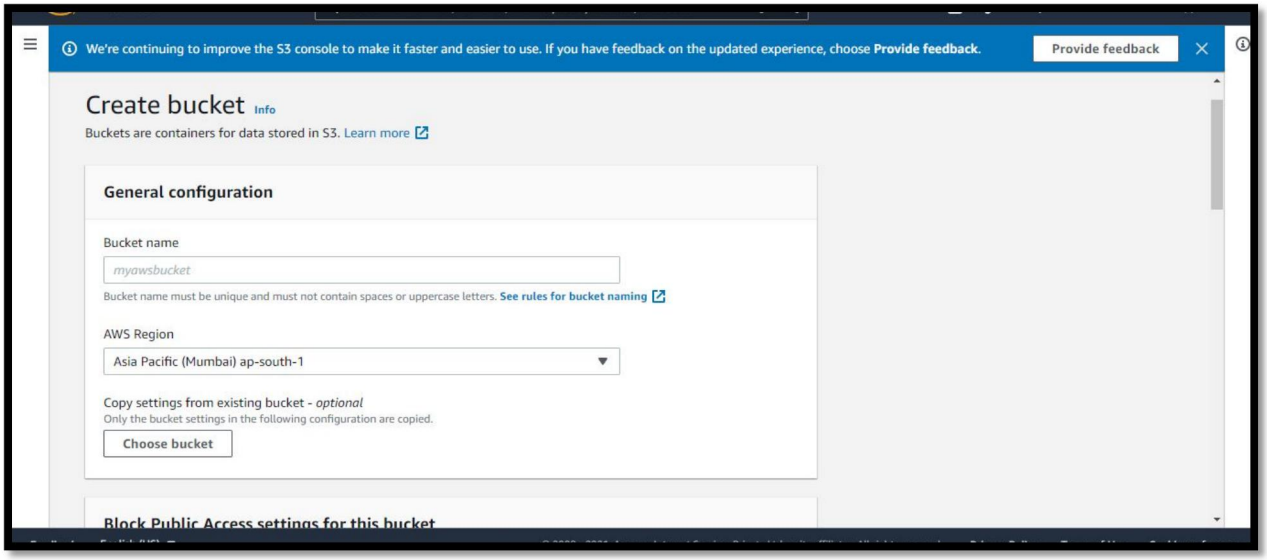
Create an S3 bucket for your application

To create an S3 bucket

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.

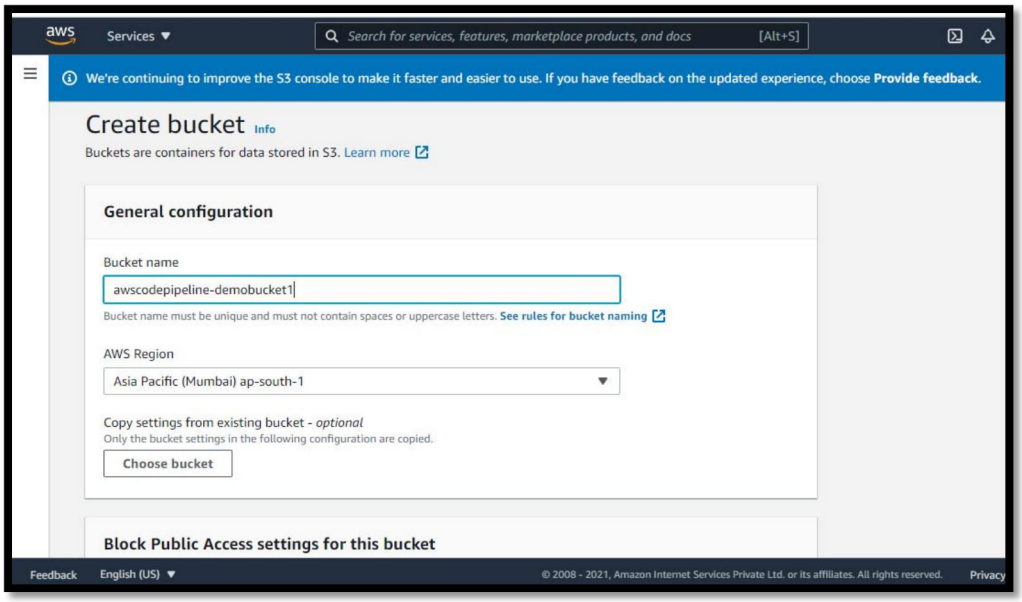


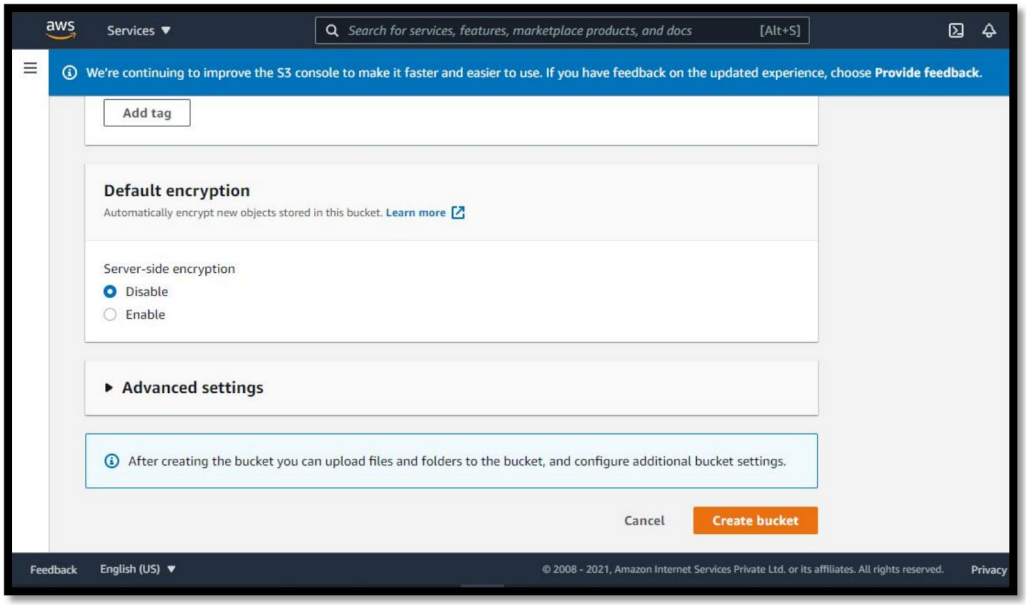
2. Choose Create bucket.



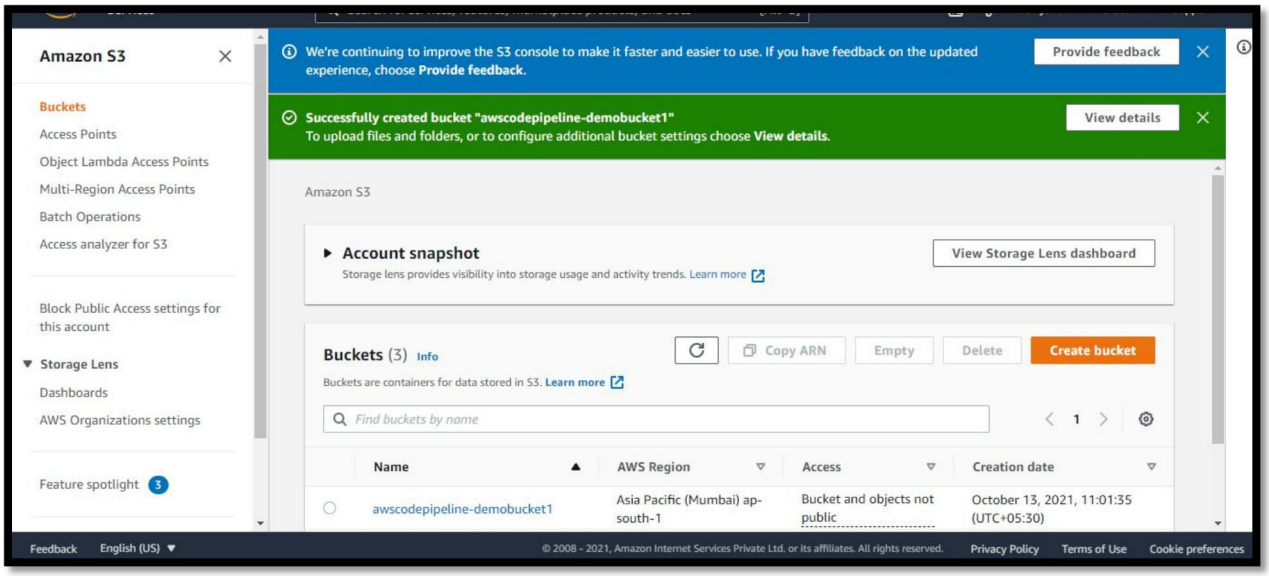
3. In Bucket name, enter a name for your bucket (for example, awscodepipeline-demobucket-example-date).

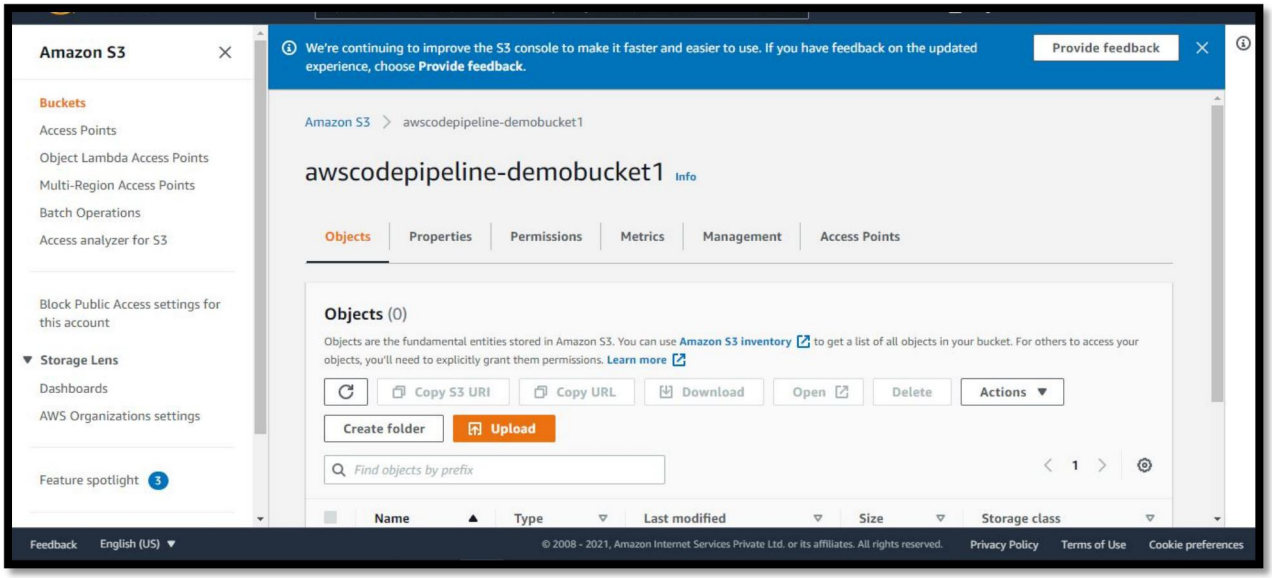
In Region, choose the Region where you intend to create your pipeline, such as Asia Pacific (Mumbai), and then choose Create bucket.





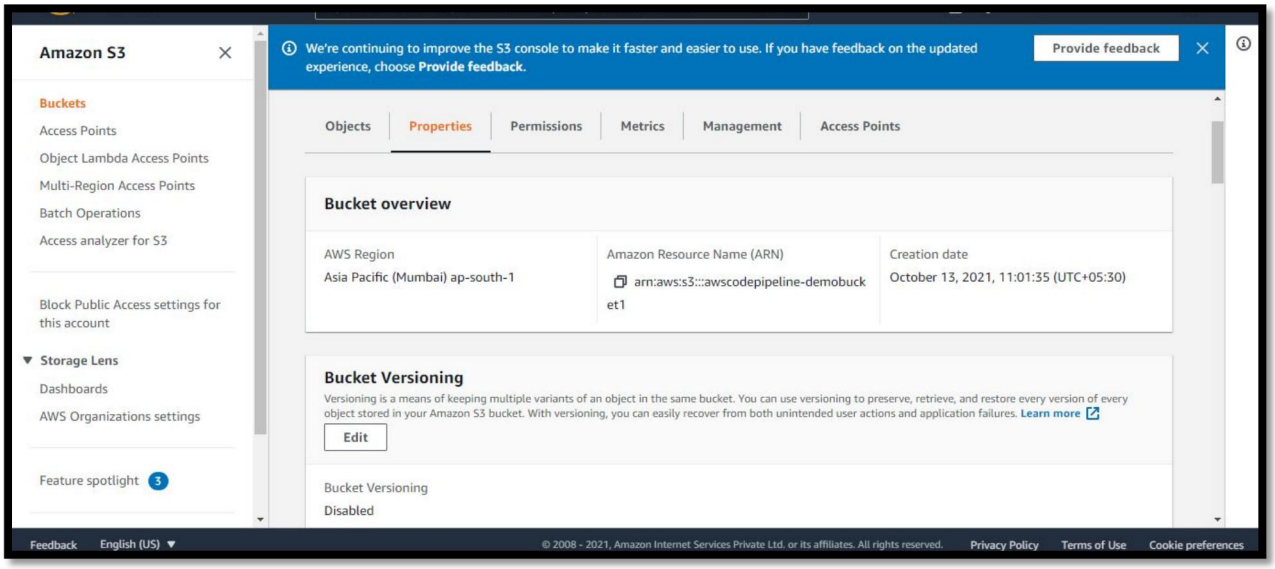
4. After the bucket is created, a success banner displays. Choose Go to bucket details.

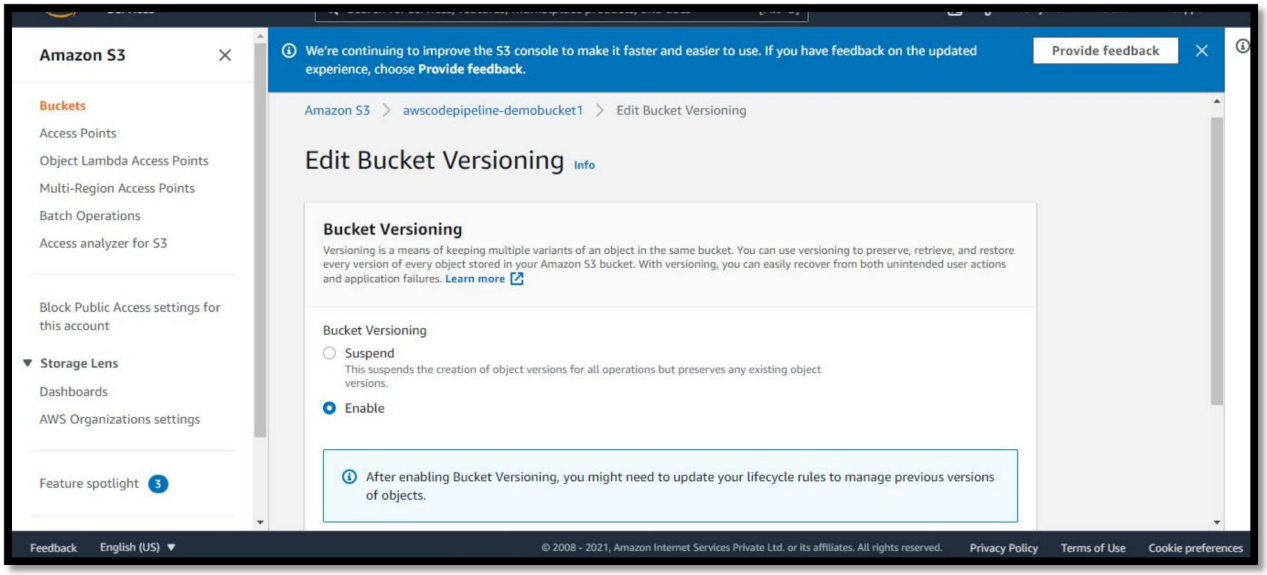




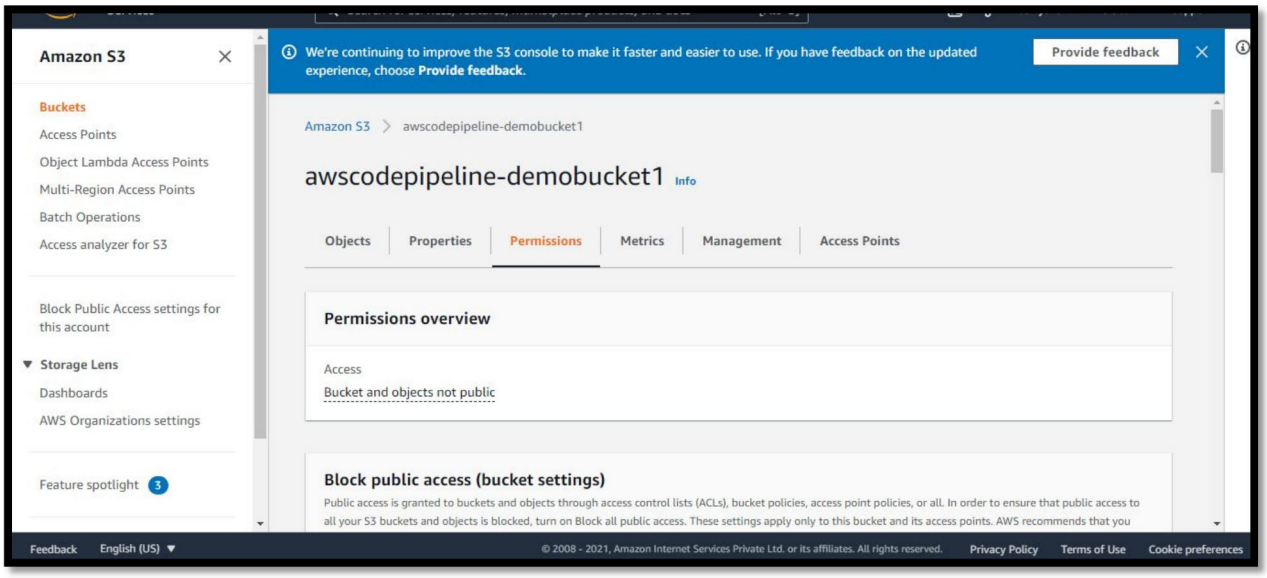
5. On the Properties tab, choose Versioning. Choose Enable versioning, and

then choose Save.
When versioning is enabled, Amazon S3 saves every version of every object in the bucket.





6. On the Permissions tab, leave the defaults. For more information about S3 bucket and object permissions, see Specifying Permissions in a Policy.



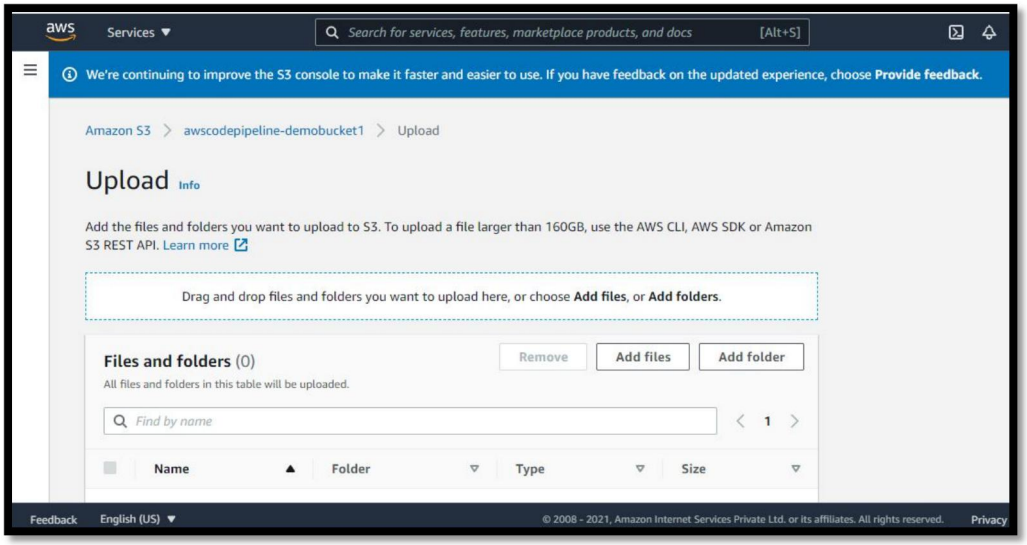
7. Next, download a sample and save it into a folder or directory on your local computer.
    a. Choose one of the following. Choose SampleApp_Windows.zip if you want to follow the steps in this tutorial for Windows Server instances.

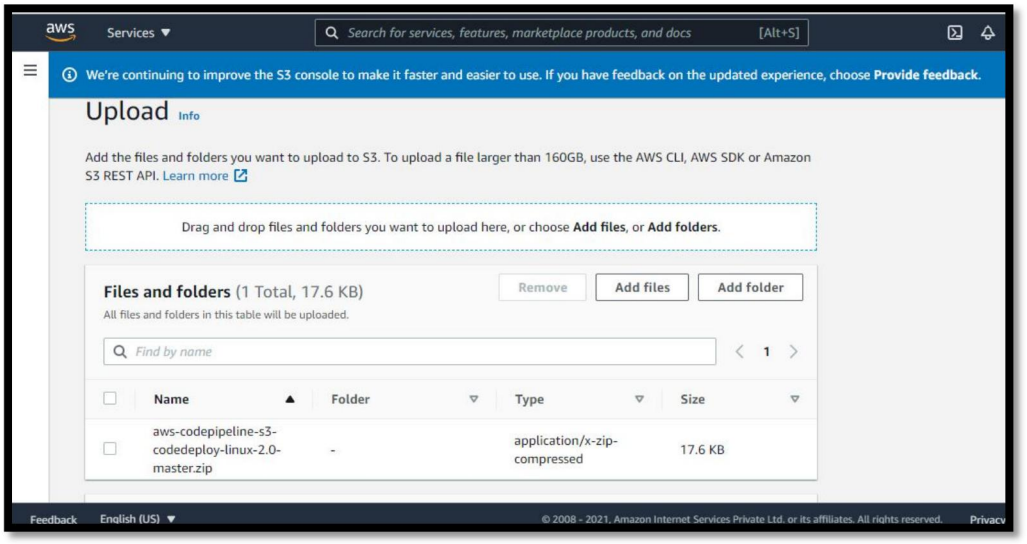
- If you want to deploy to Amazon Linux instances using CodeDeploy, download the sample application here: SampleApp_Linux.zip.
- If you want to deploy to Windows Server instances using CodeDeploy, download the sample application here: SampleApp_Windows.zip.
    b. Download the compressed (zipped) file. Do not unzip the file.

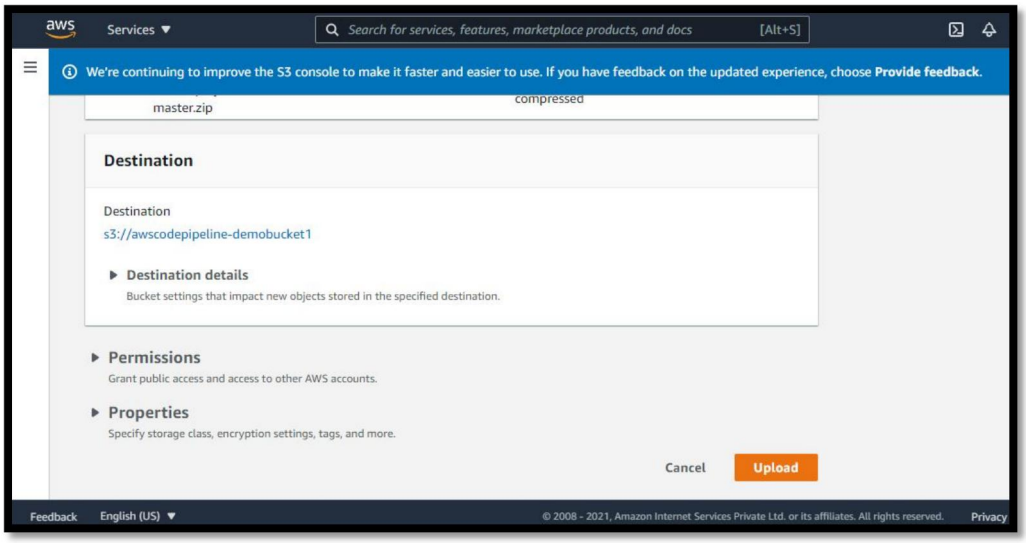
SampleApp_Windows.zip

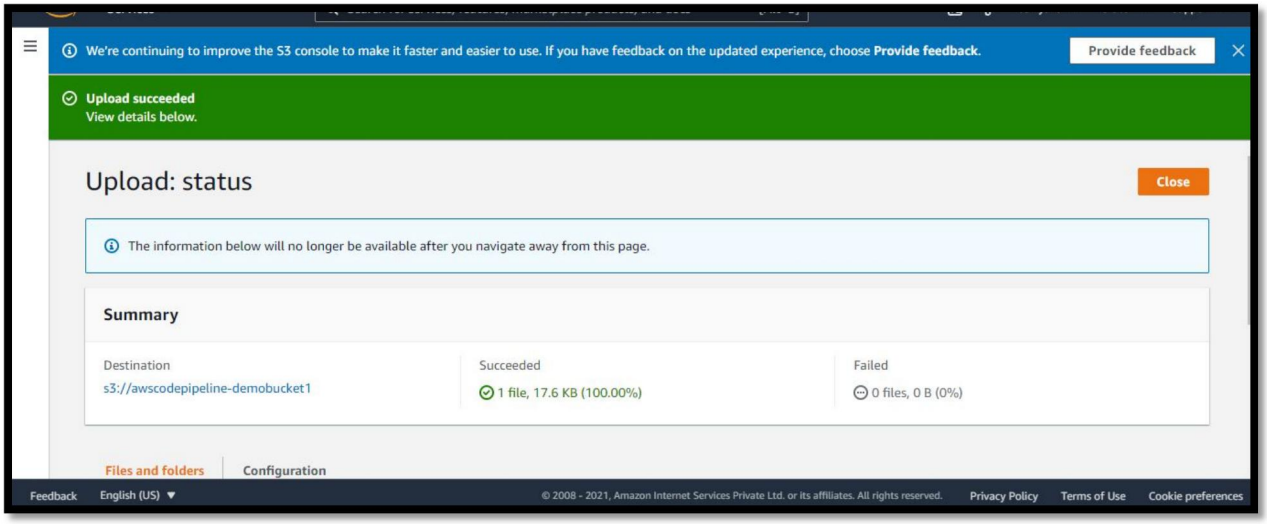
8. In the Amazon S3 console, for your bucket, upload the file:
    1. Choose **Upload**.

2. Drag and drop the file or choose **Add files** and browse for the file.
3. Choose **Upload**.





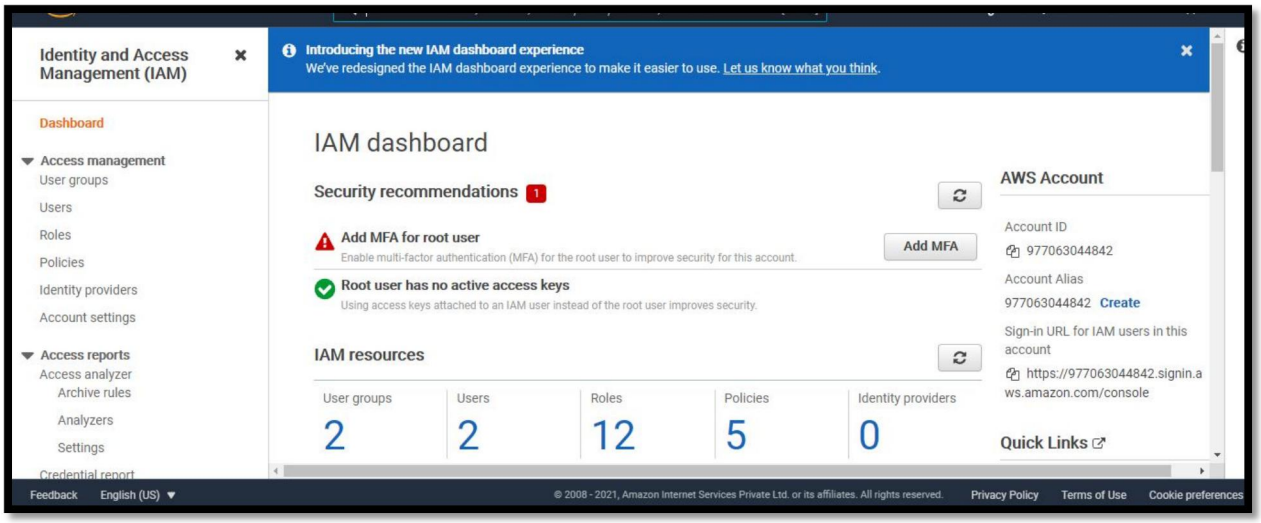




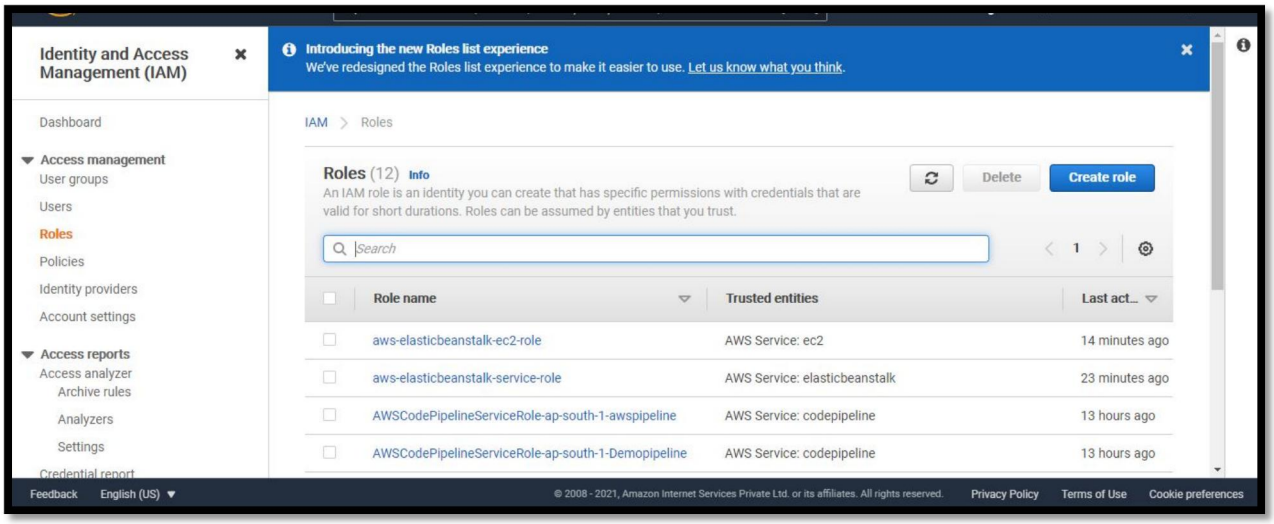
Deploy Sample Application on EC2 instance using AWS CodeDeploy

To create an instance role

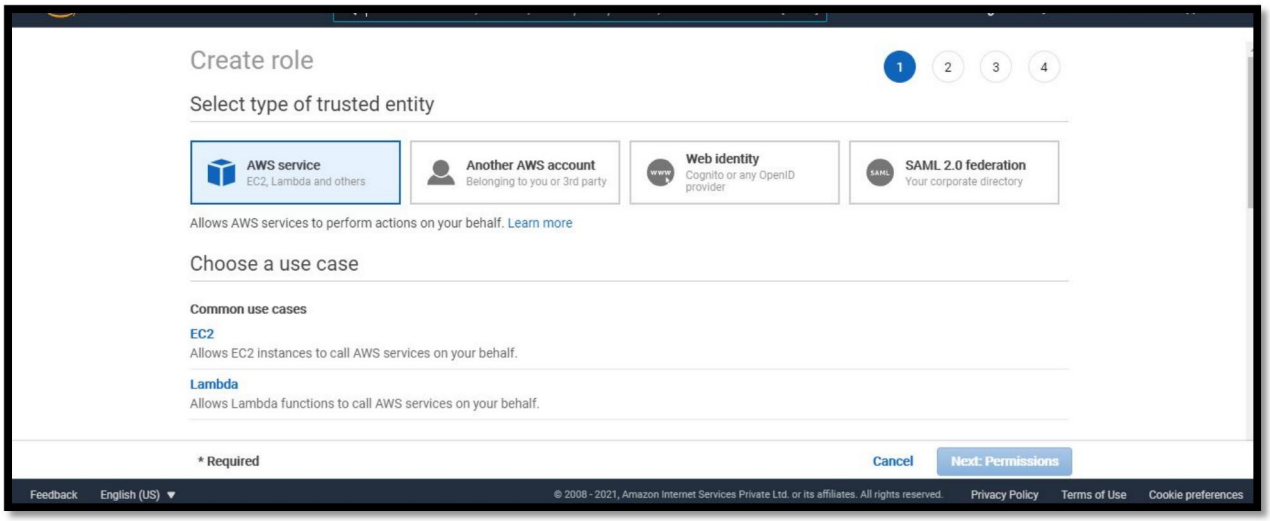
1. Open the IAM console at (https://console.aws.amazon.com/iam/).



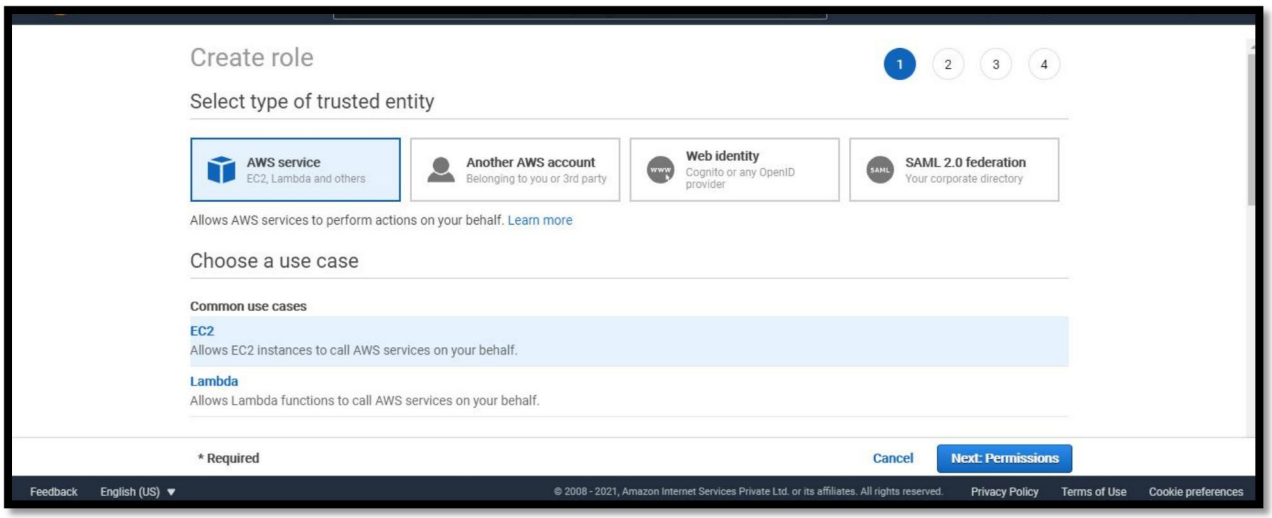
2. From the console dashboard, choose Roles.



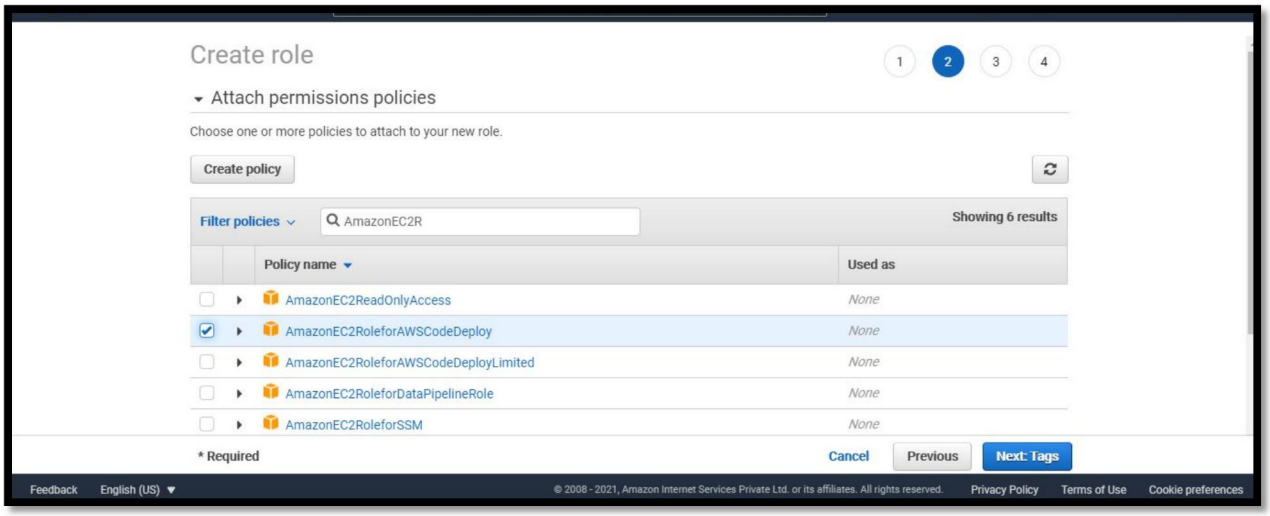
3. Choose Create role.



4. Under Select type of trusted entity, select AWS service. Under Choose a use case, select EC2, and then choose Next: Permissions.

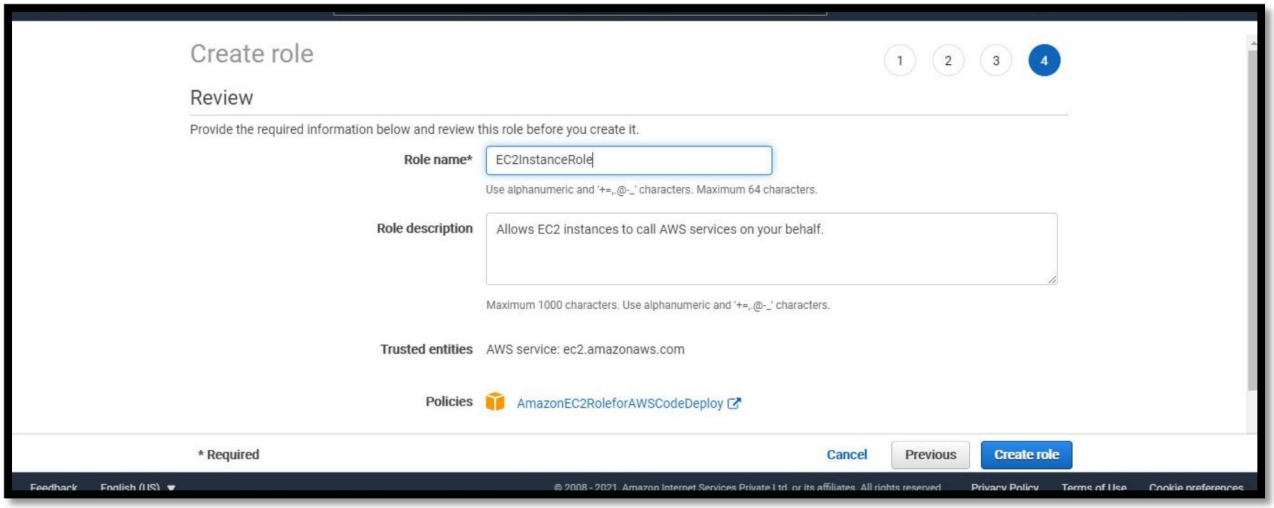


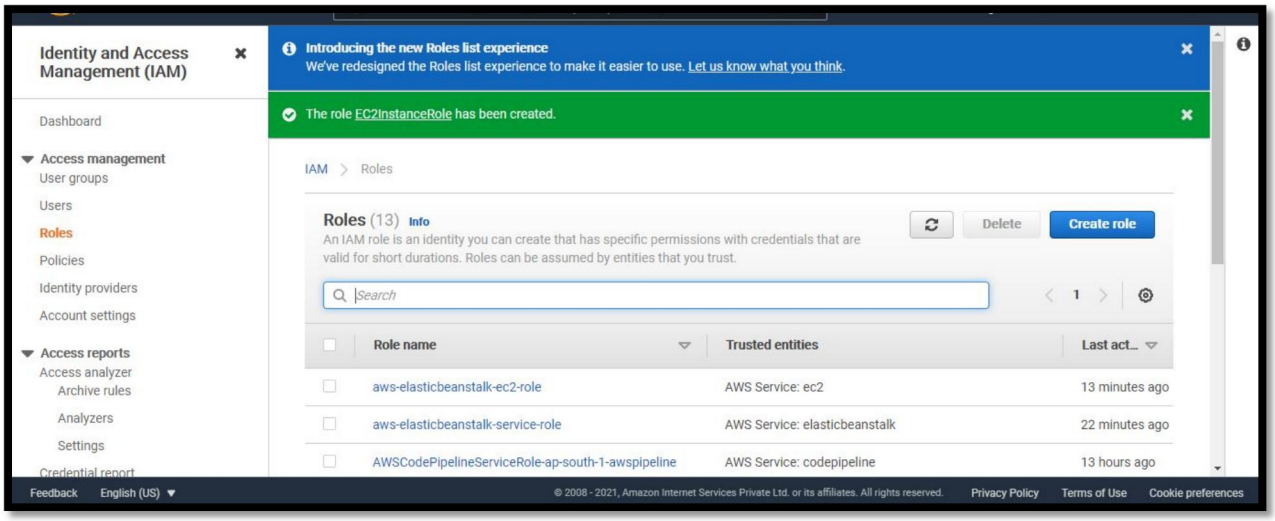
5. Search for and select the policy named AmazonEC2RoleforAWSCodeDeploy, and then choose Next: Tags.



6. Choose Next: Review. Enter a name for the role (for example, EC2InstanceRole).

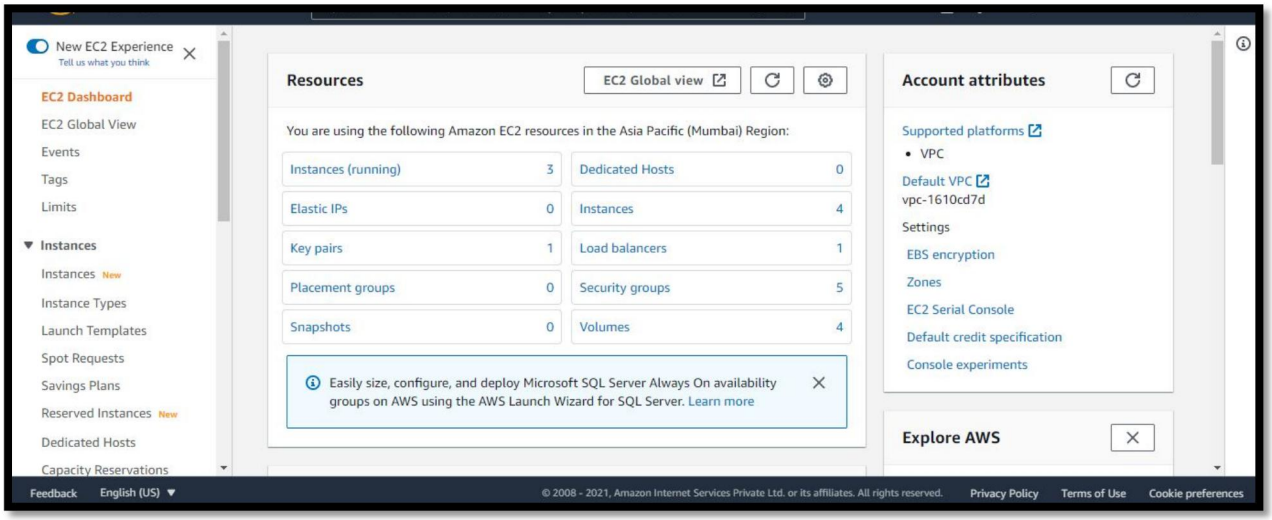
Choose Create role.



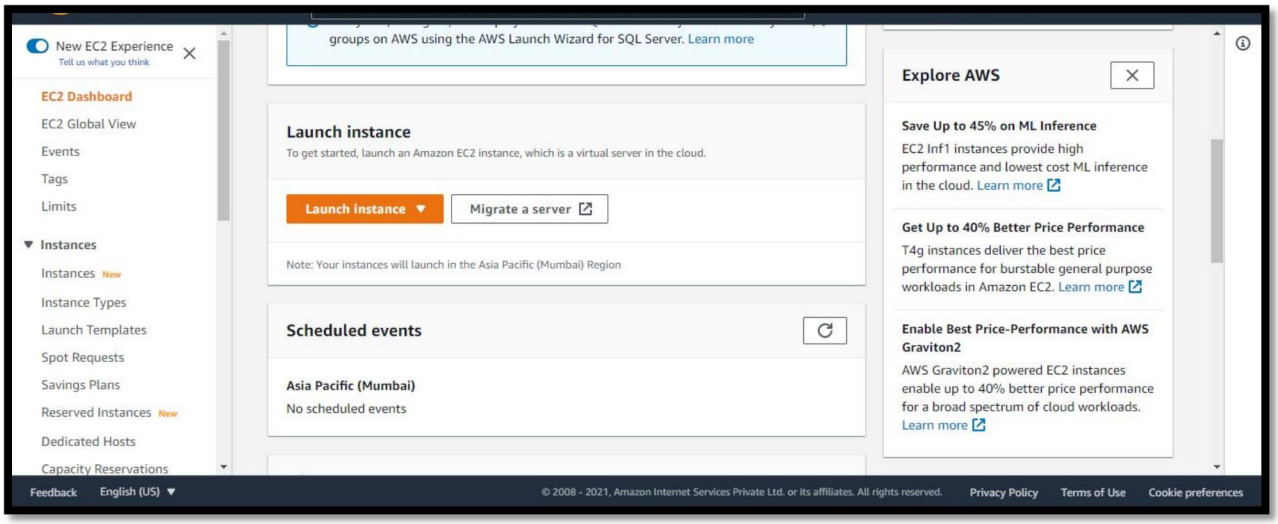


To launch instances

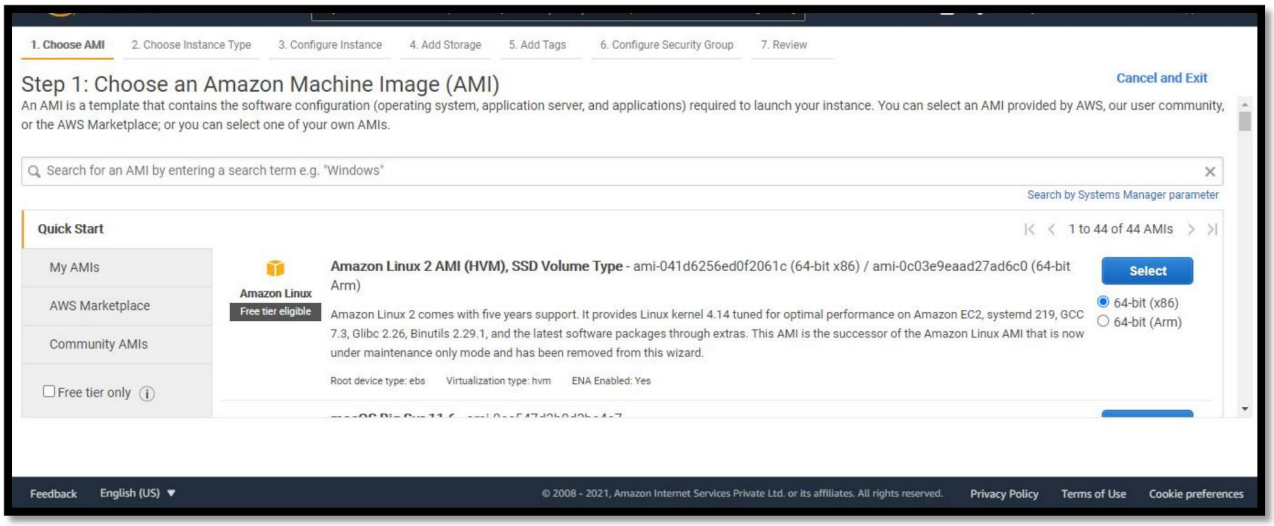
1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

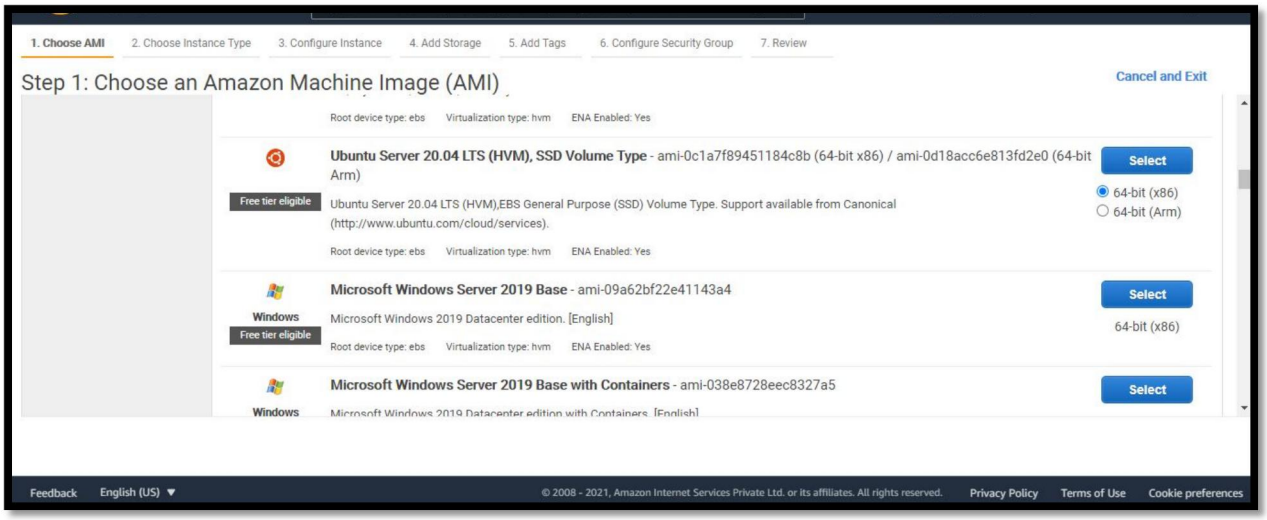


2. From the console dashboard, choose Launch instance, and select Launch instance from the options that pop up.

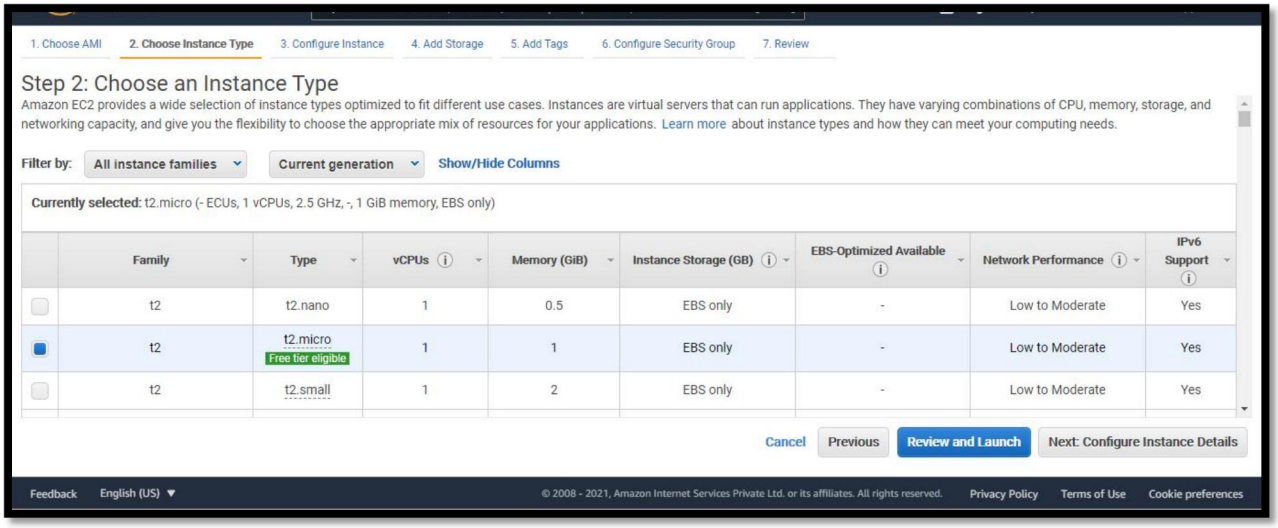


3. On the Step 1: Choose an Amazon Machine Image (AMI) page, locate the Microsoft Windows Server 2019 Base option, and then choose Select. (This AMI is labeled "Free tier eligible" and can be found at the top of the list.)





4. On the Step 2: Choose an Instance Type page, choose the free tier eligible t2.micro type as the hardware configuration for your instance, and then choose Next: Configure Instance Details.

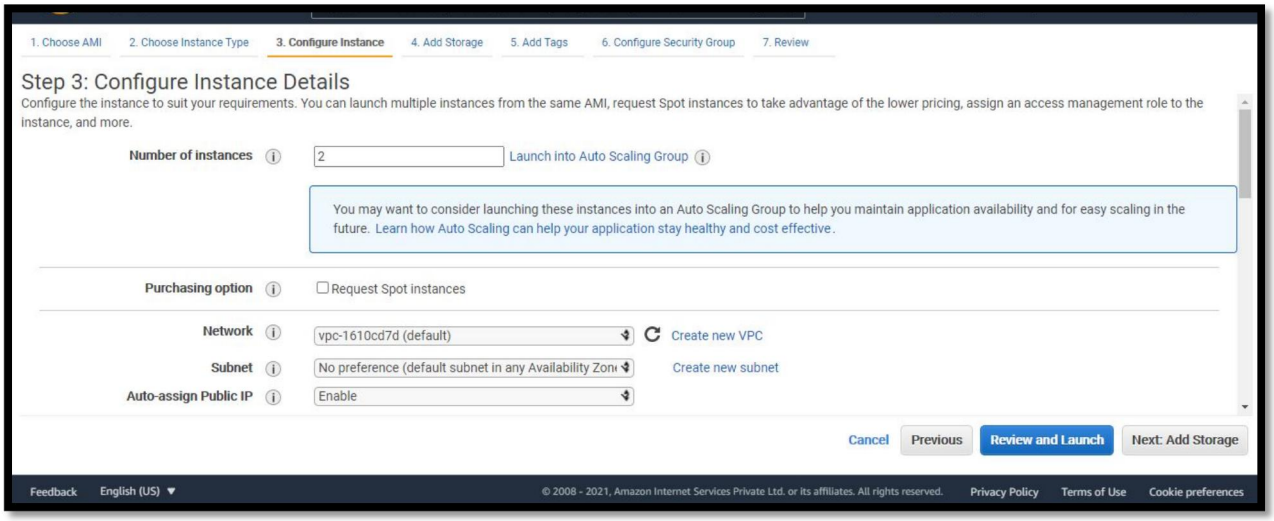


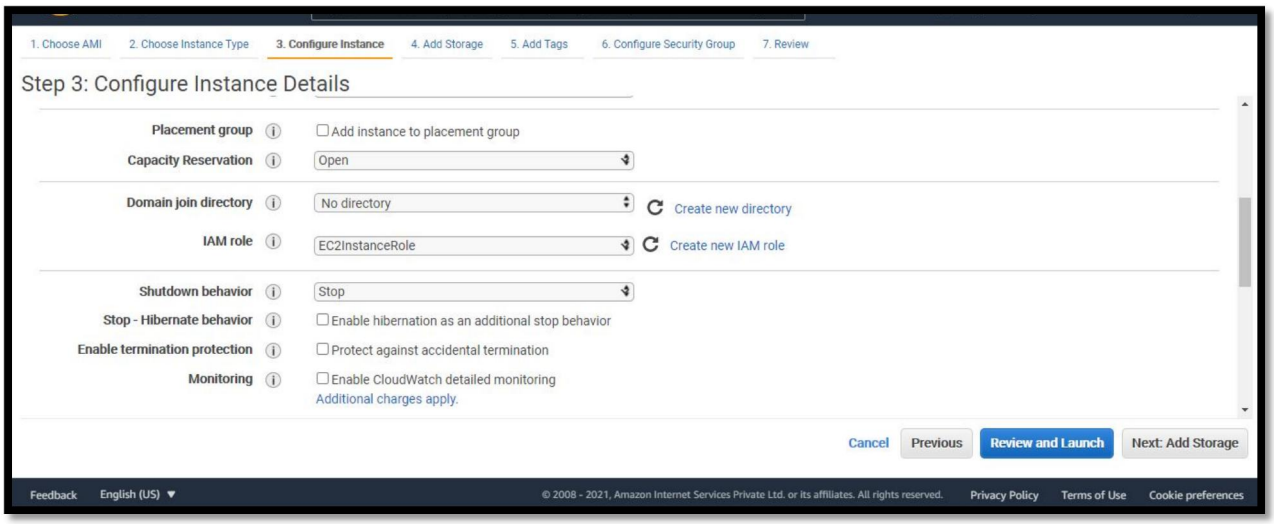
5. On the Step 3: Configure Instance Details page, do the following:
   - In Number of instances, enter 2.
   - In Auto-assign Public IP, choose Enable.
   - In IAM role, choose the IAM role you created in the previous procedure (for example, EC2InstanceRole).
   - Expand Advanced Details, and in User data, with As text selected, enter the following:

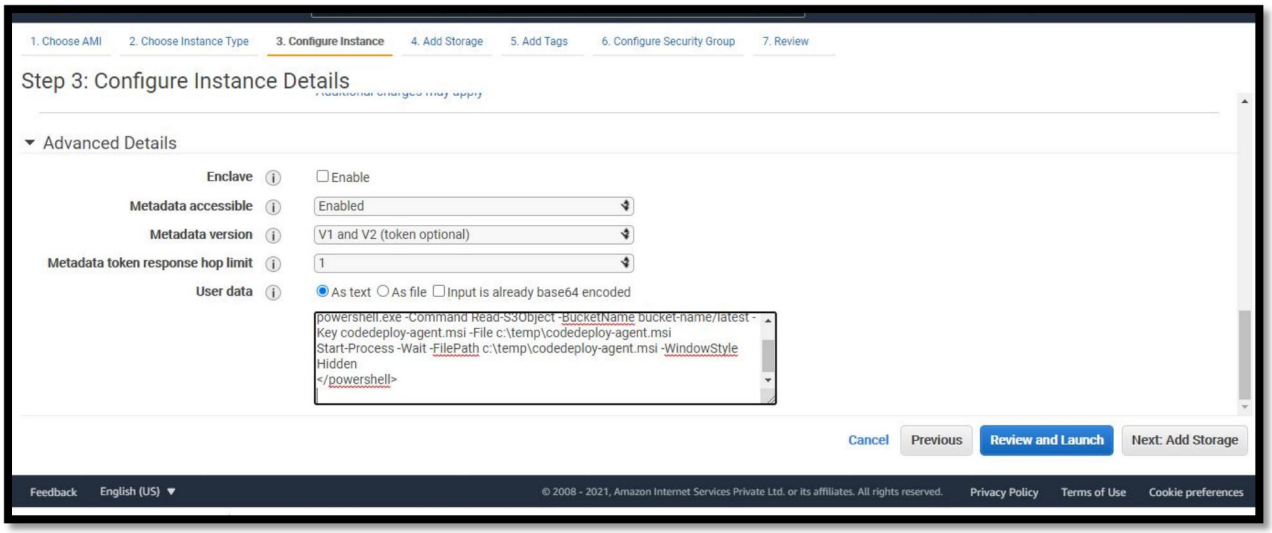
```
<powershell>
New-Item -Path c:\temp -ItemType "directory" -Force
powershell.exe -Command Read-S3Object -BucketName bucket-name/latest -Key codedeploy-agent.msi -File c:\temp\codedeploy-agent.msi
Start-Process -Wait -FilePath c:\temp\codedeploy-agent.msi -WindowStyle Hidden
</powershell>
```

   - *bucket-name* is the name of the S3 bucket that contains the CodeDeploy Resource Kit files for your Region. For example, for the US West (Oregon) Region, replace *bucket-name* with aws-codedeploy-us-west-2. For a list of bucket names, see Resource Kit Bucket Names by Region.
     This code installs the CodeDeploy agent on your instance as it is created. This script is written for Windows instances only.

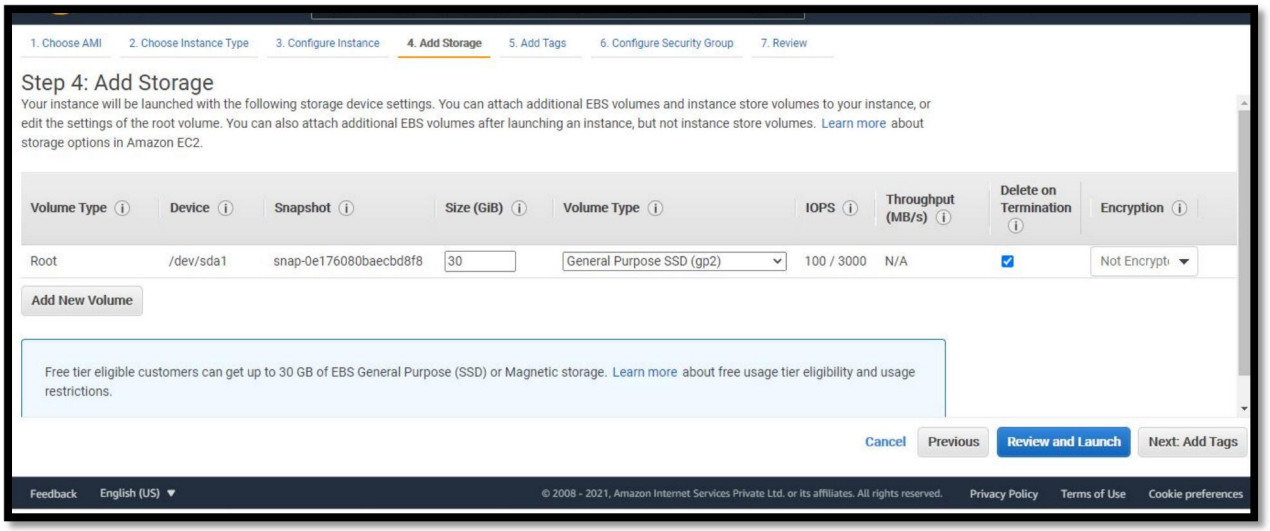
- Leave the rest of the items on the Step 3: Configure Instance Details page unchanged. Choose Next: Add Storage.



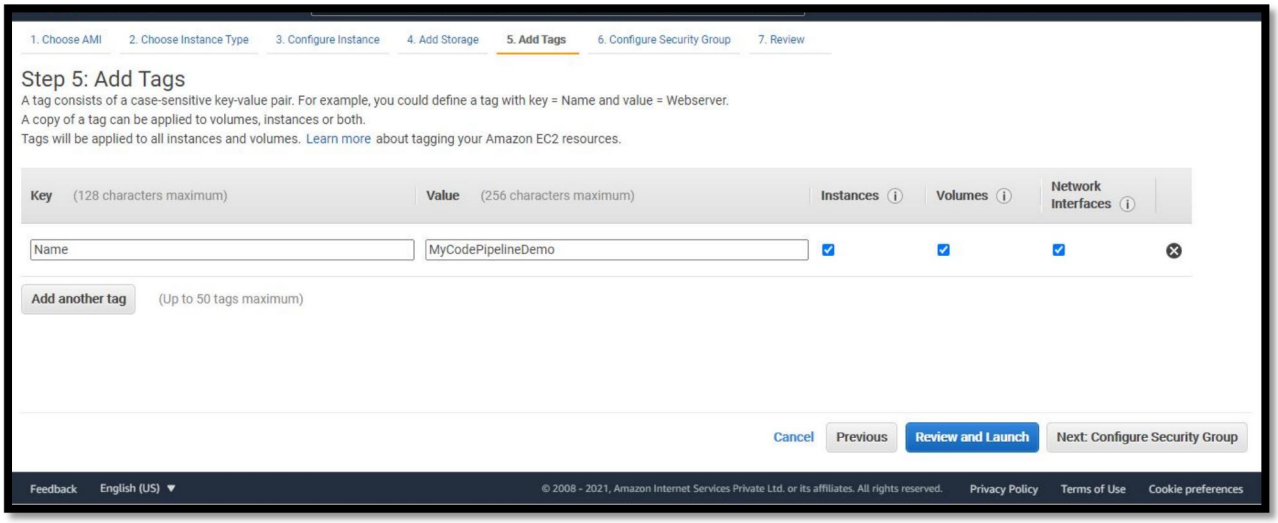




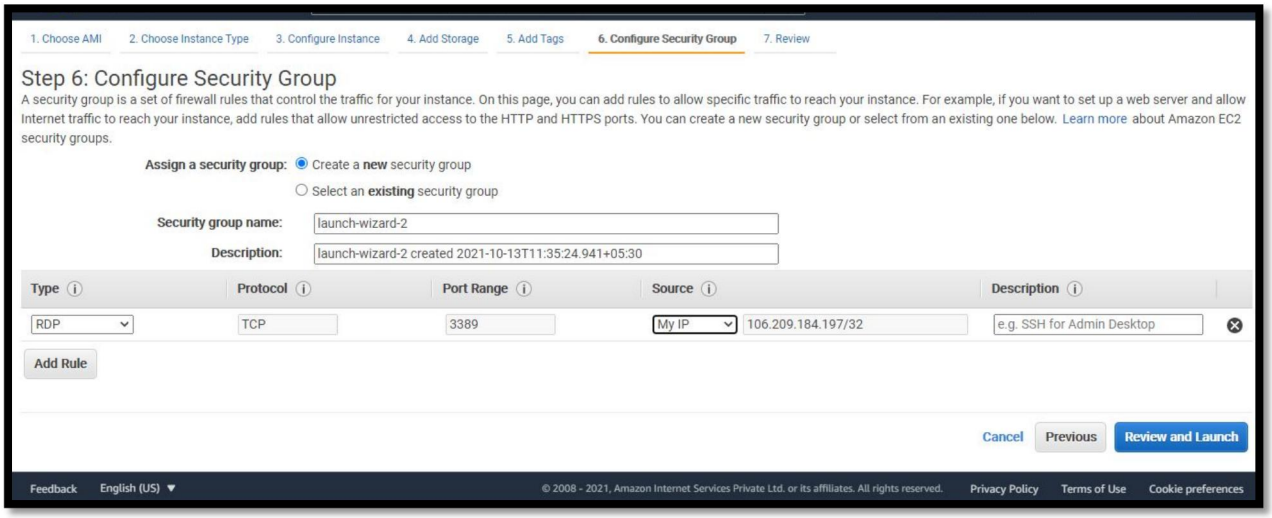
6. Leave the Step 4: Add Storage page unchanged, and then choose Next: Add Tags.



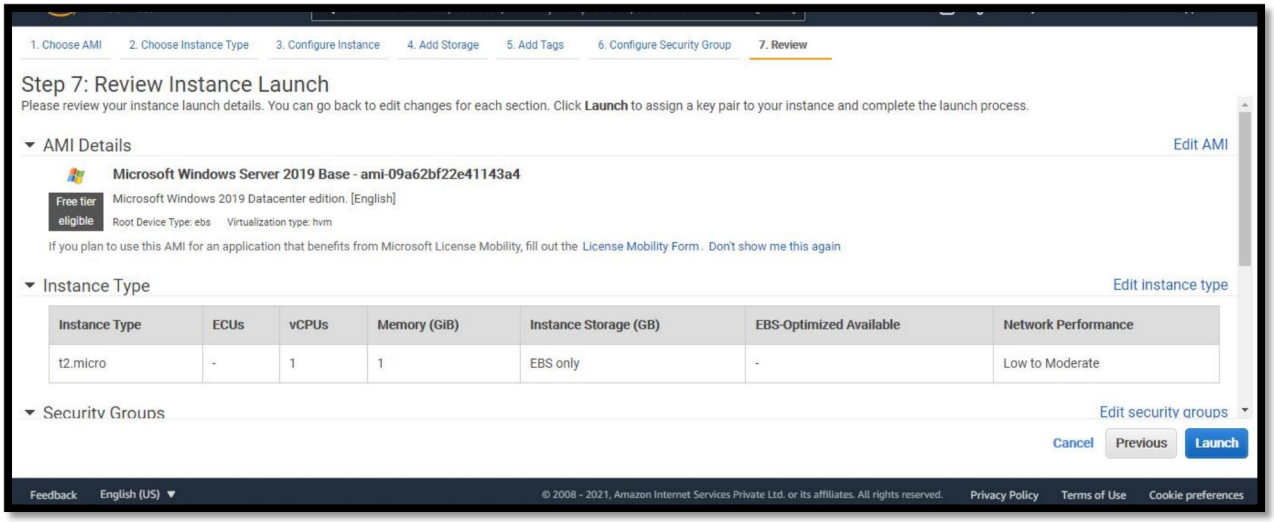
7. On the Add Tags page, choose Add Tag. Enter Name in the Key field, enter MyCodePipelineDemo in the Value field, and then choose Next: Configure Security Group.



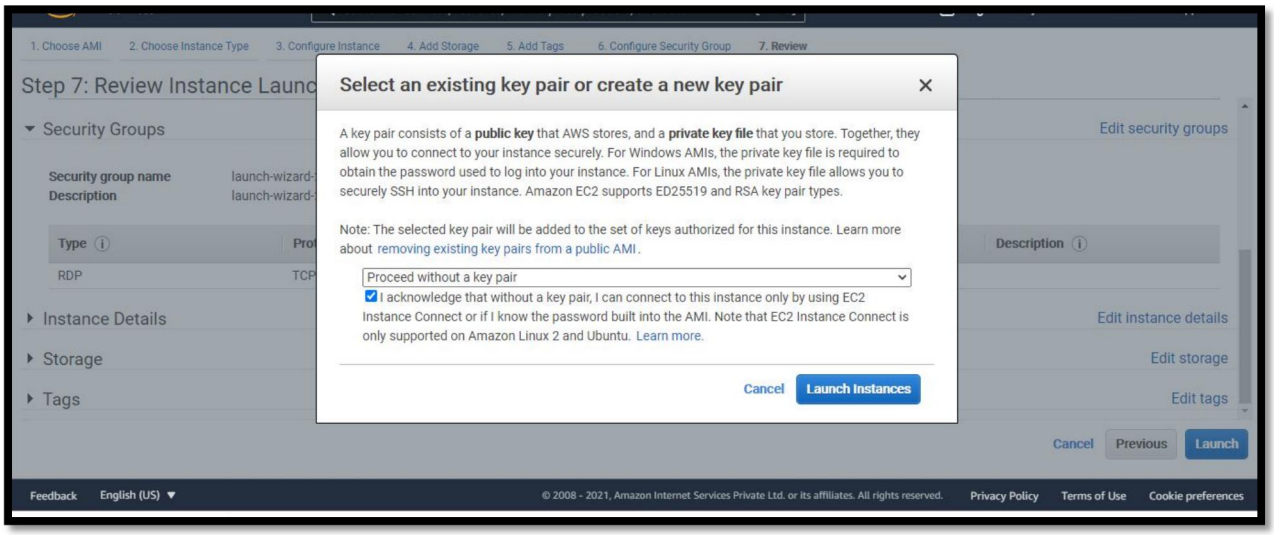
8. On the Configure Security Group page, allow port 80 communication so you can access the public instance endpoint.

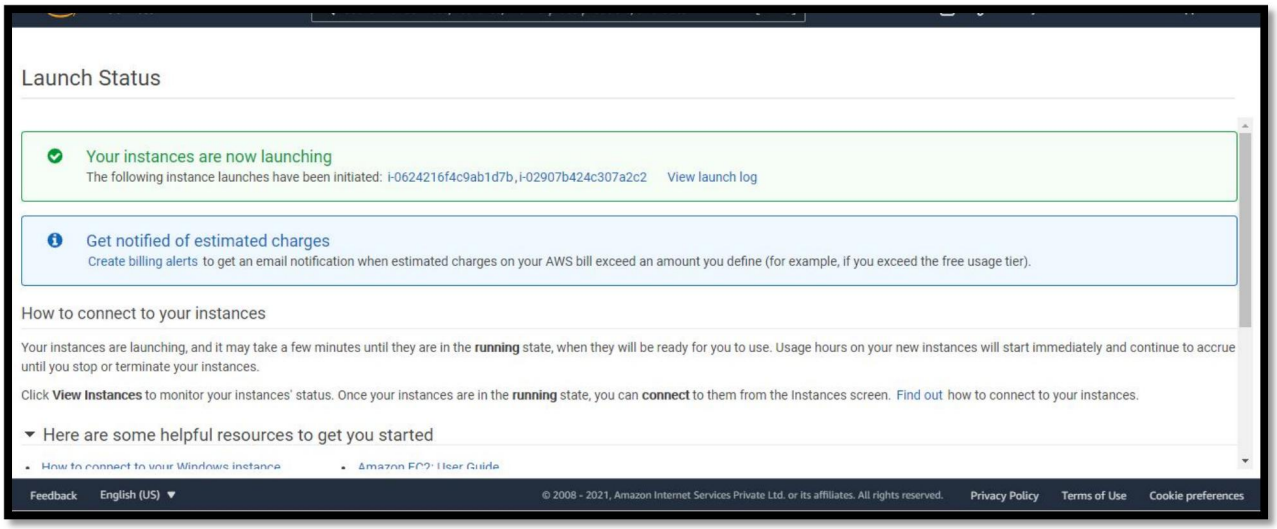


9. Choose Review and Launch.

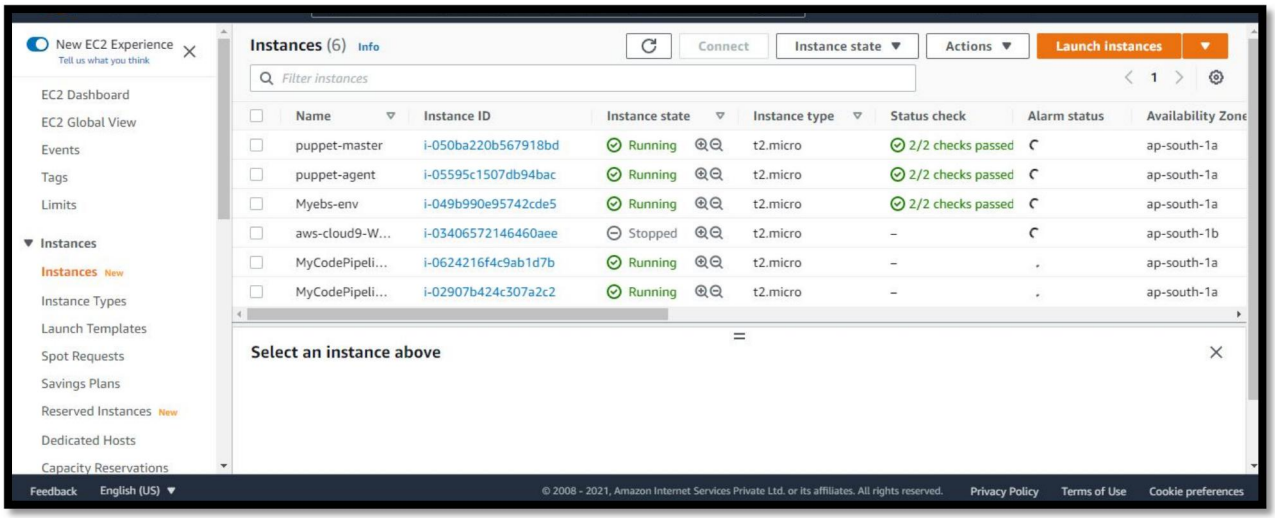


10.On the Review Instance Launch page, choose Launch. When prompted for a key pair, choose Proceed without a key pair.
When you are ready, select the acknowledgment check box, and then choose Launch Instances.

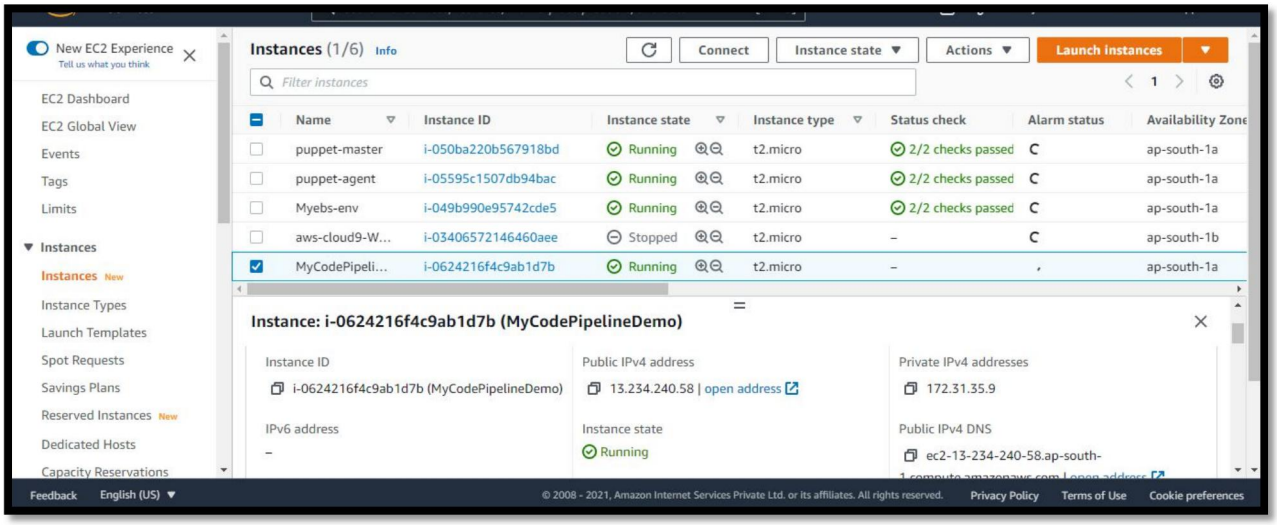


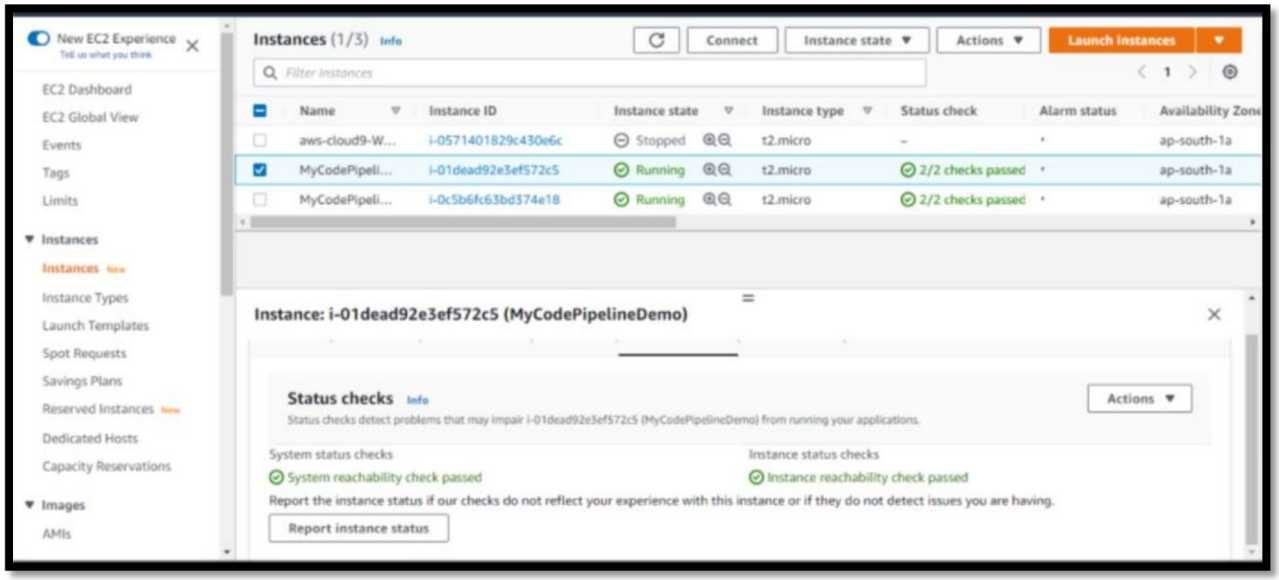


11. Choose View Instances to close the confirmation page and return to the console.
12. You can view the status of the launch on the Instances page. When you launch an instance, its initial state is pending. After the instance starts, its state changes to running, and it receives a public DNS name. (If the Public DNS column is not displayed, choose the Show/Hide icon, and then select Public DNS.)



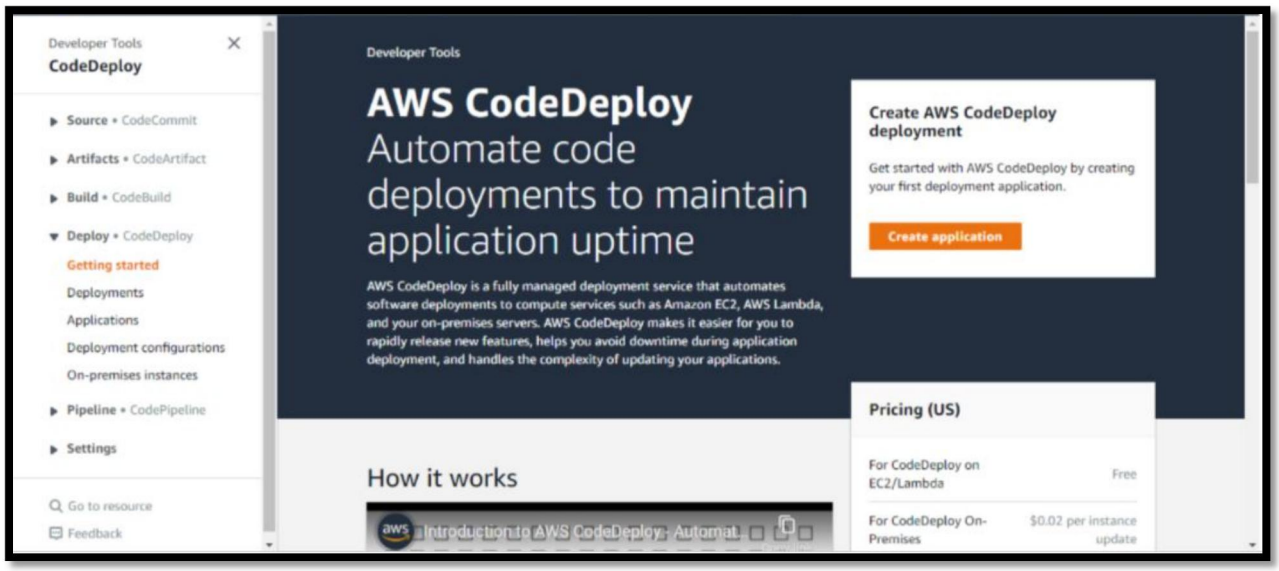
13. It can take a few minutes for the instance to be ready for you to connect to it. Check that your instance has passed its status checks. You can view this information in the Status Checks column.



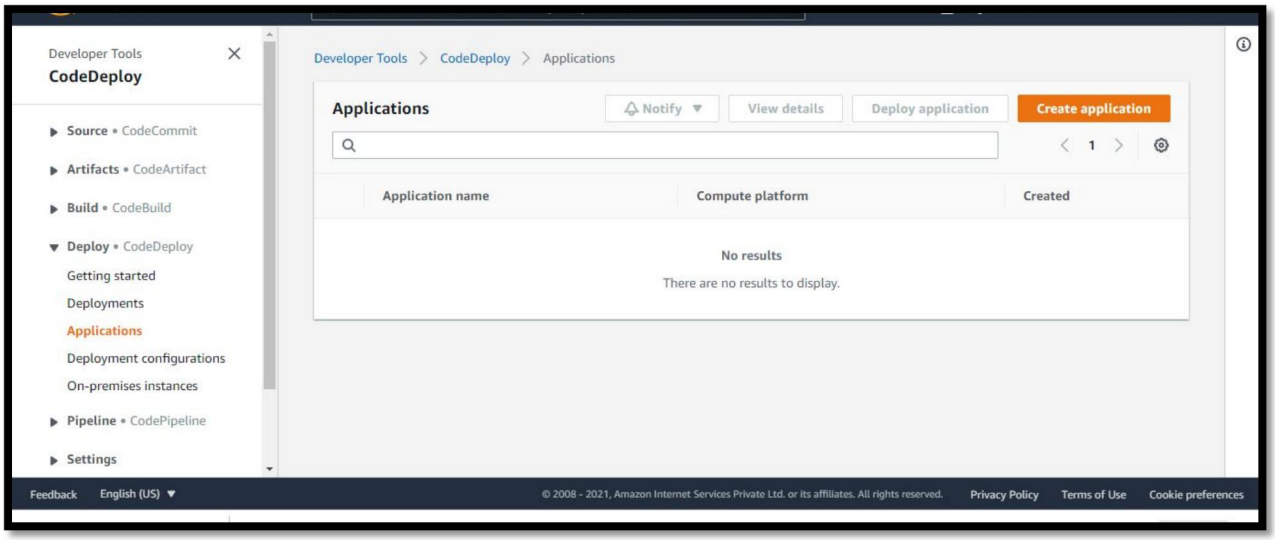


To create an application in CodeDeploy

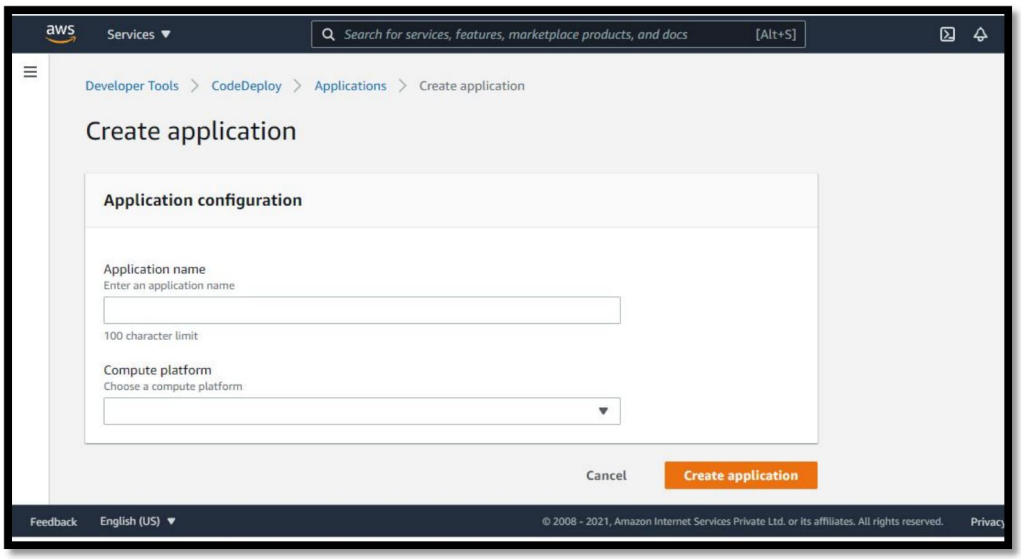
1. Open the CodeDeploy console at https://console.aws.amazon.com/codedeploy.



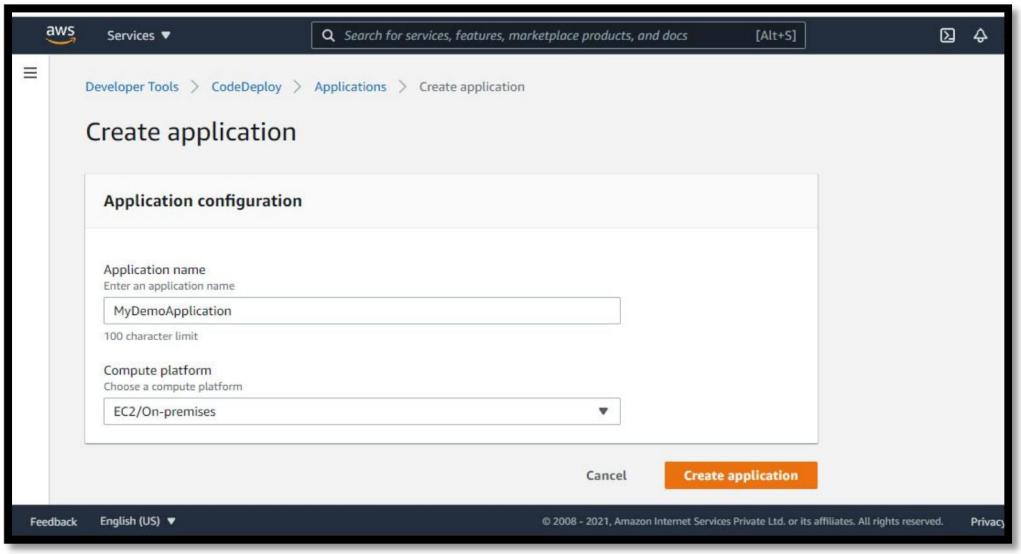
2. If the Applications page does not appear, on the AWS CodeDeploy menu, choose Applications.



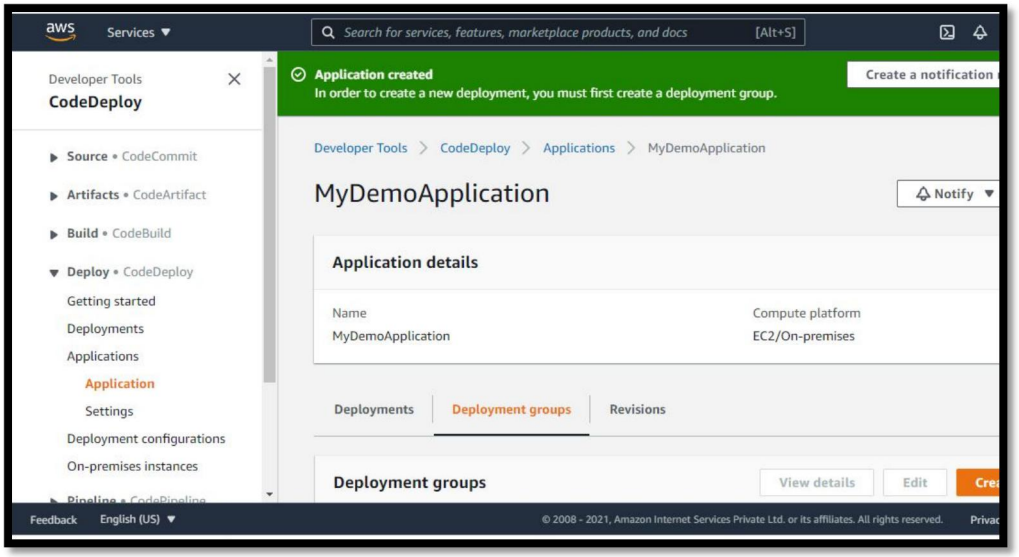
3. Choose Create application.



4. In Application name, enter MyDemoApplication.
5. In Compute Platform, choose EC2/On-premises.

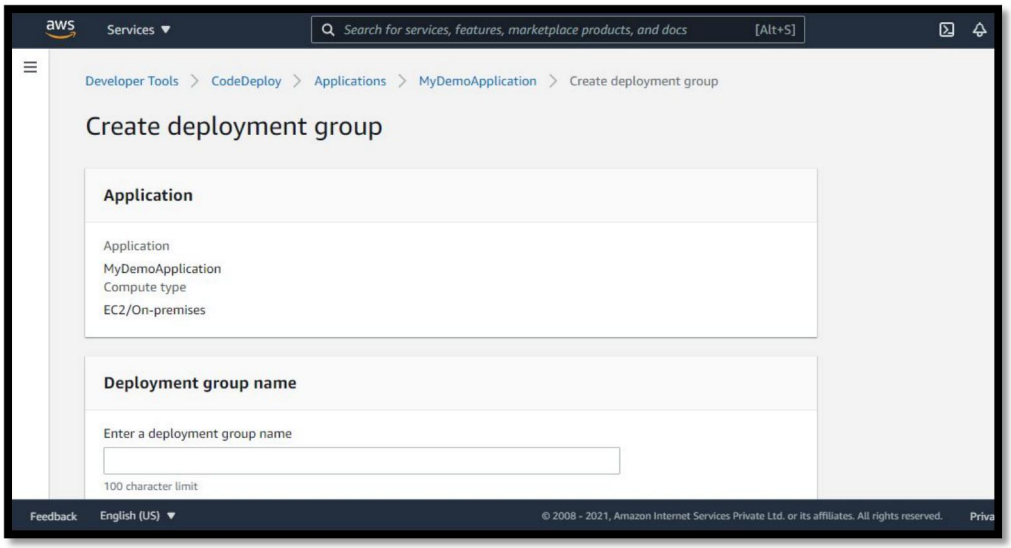


6. Choose Create application.

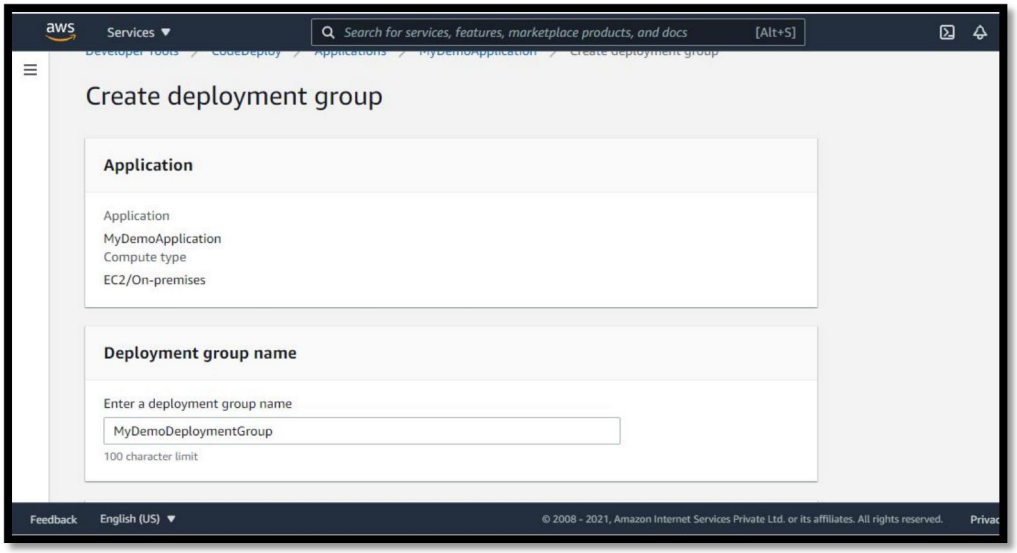


To create a deployment group in CodeDeploy

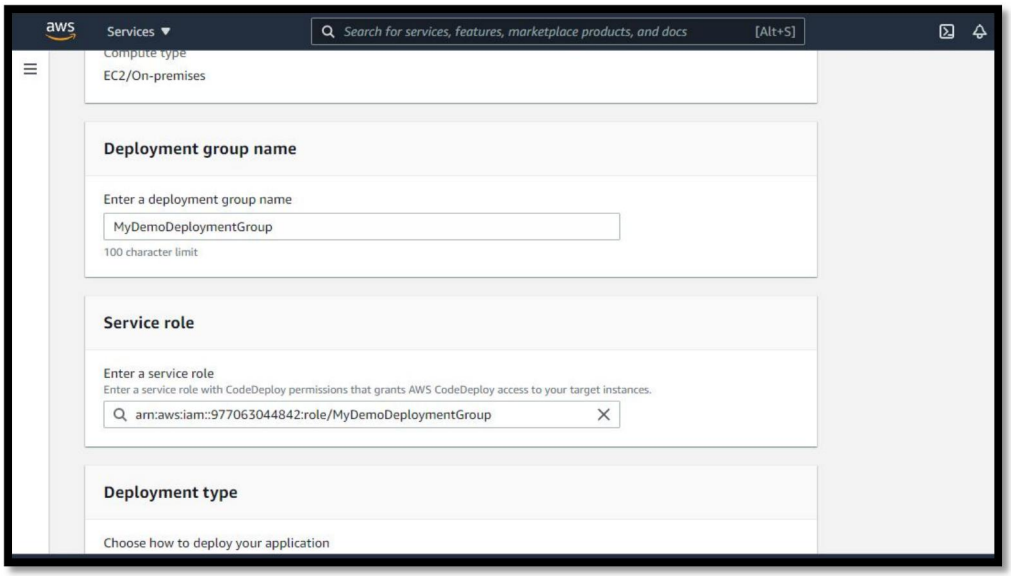
1. On the page that displays your application, choose Create deployment group.



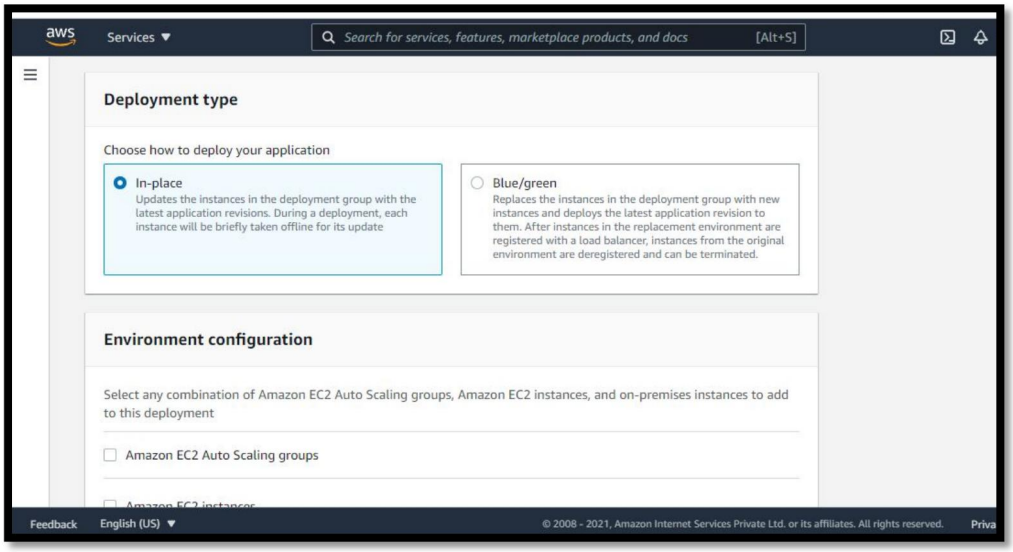
2. In Deployment group name, enter MyDemoDeploymentGroup.



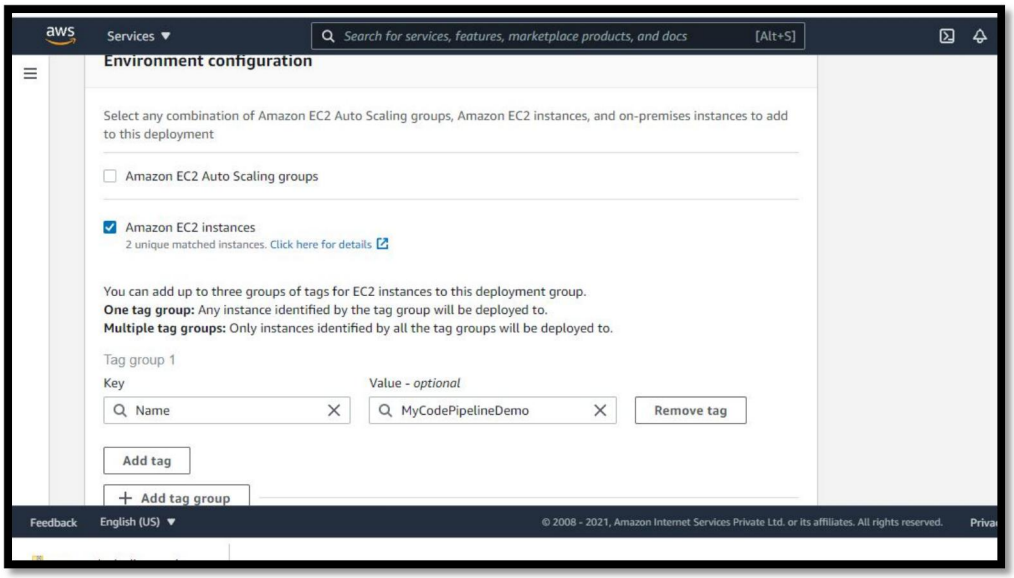
3. In Service Role, choose a service role that trusts AWS CodeDeploy with, at minimum, the trust and permissions described in Create a Service Role for CodeDeploy. To get the service role ARN, see Get the Service Role ARN (Console).



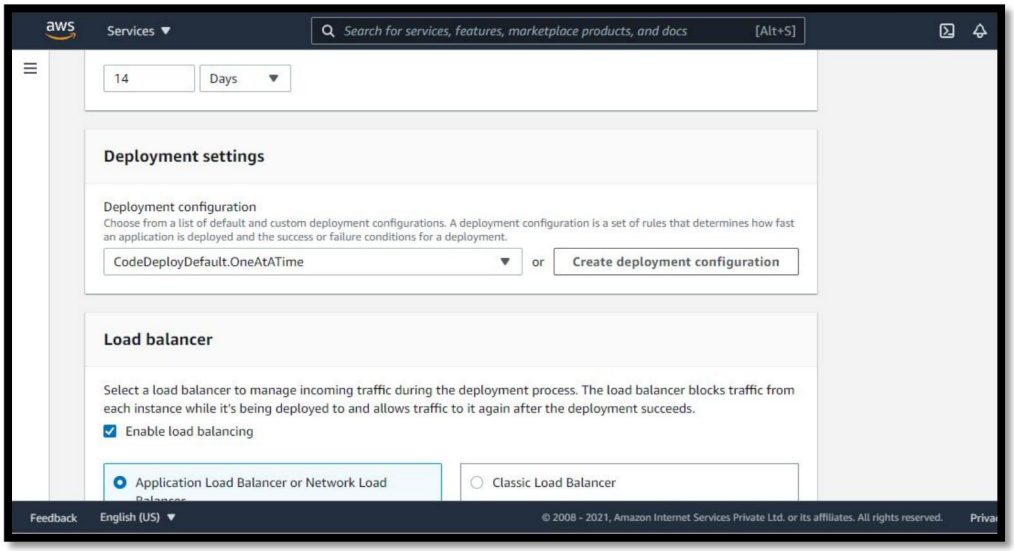
4. Under Deployment type, choose In-place.



5. Under Environment configuration, choose Amazon EC2 Instances. Choose Name in the Key field, and in the Value field, enter MyCodePipelineDemo.

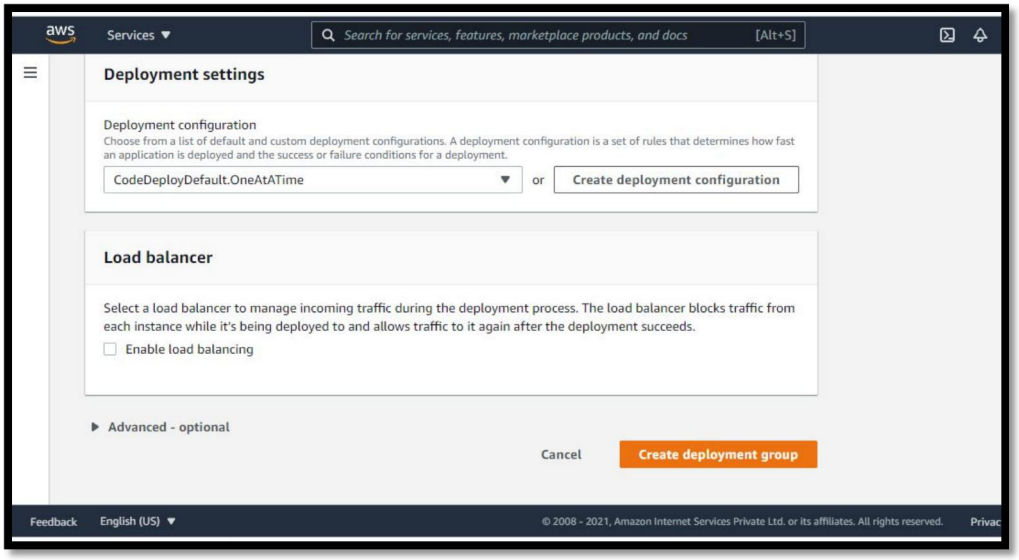


6. Under Deployment settings, choose CodeDeployDefault.OneAtaTime.

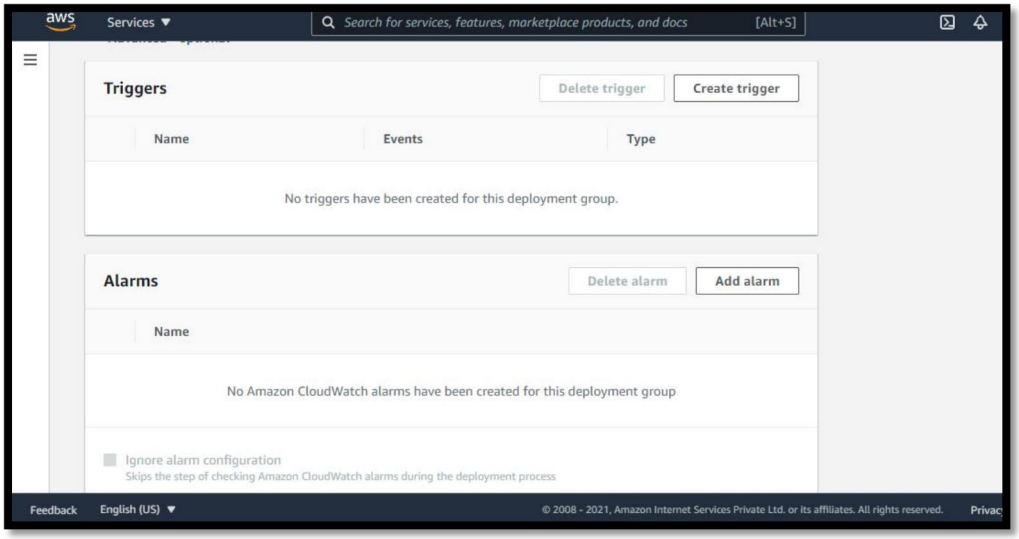


7. Under Load Balancer, make sure the Enable load balancing box is not selected. You do not need to set up a load balancer or choose a target group

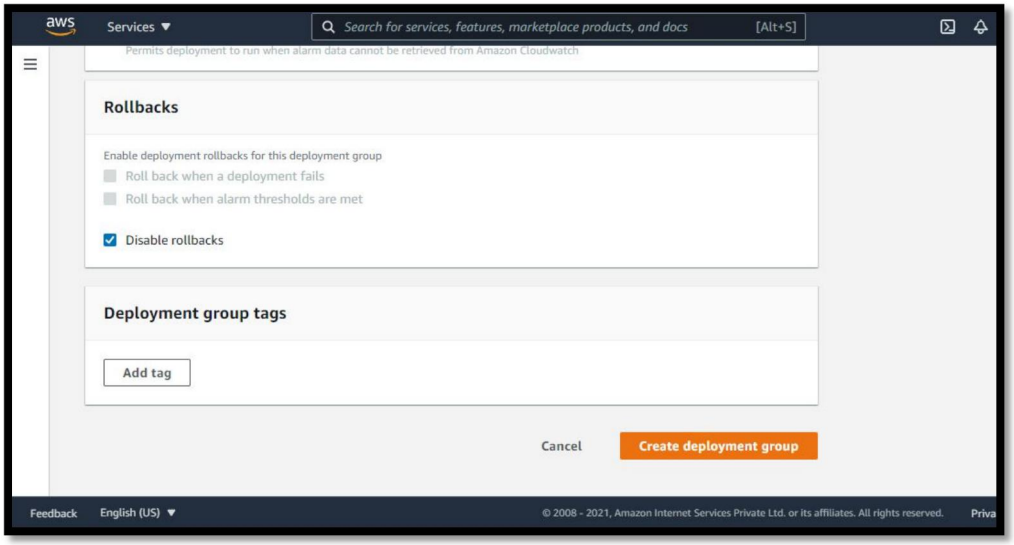
for this example. After you deselect the checkbox, the load balancer options do not display.

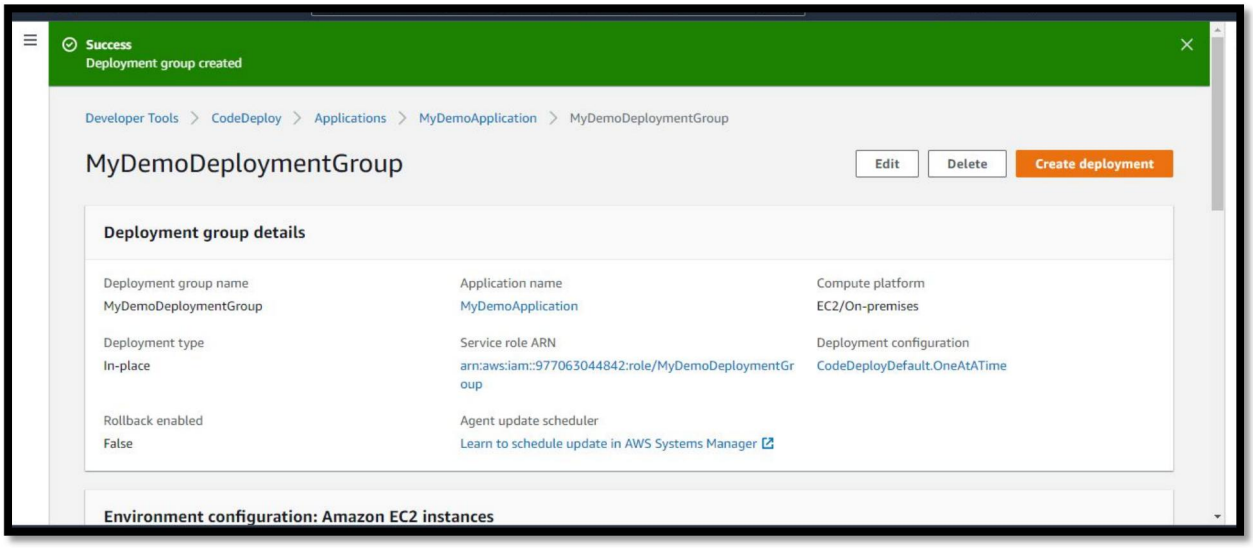


8. In the Advanced section, leave the defaults.



9. Choose Create deployment group.





Conclusion :-

We have successfully Build an Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.