Lesson 03 Demo 04

Building and Deploying a React Application with AWS CodeBuild and S3

Objective: To automate the continuous integration process by implementing CodePipeline to perform build automation for React source code and deploying it to an S3 bucket

Tools required: AWS CodeBuild, AWS CodePipeline, and S3 bucket

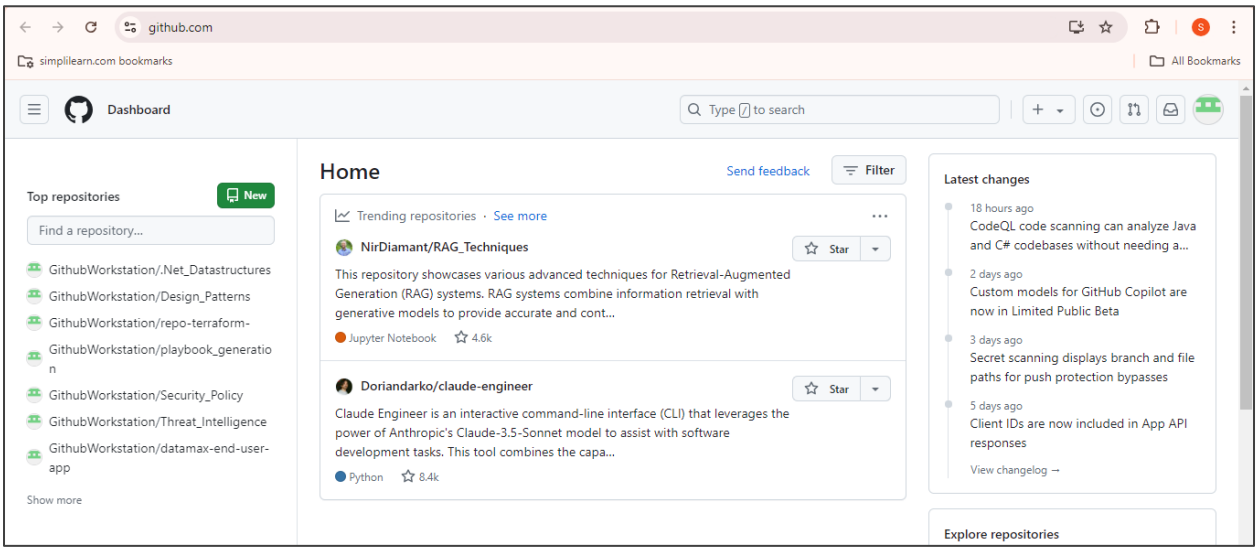
Prerequisites: EC2 instance creation

Steps to be followed:

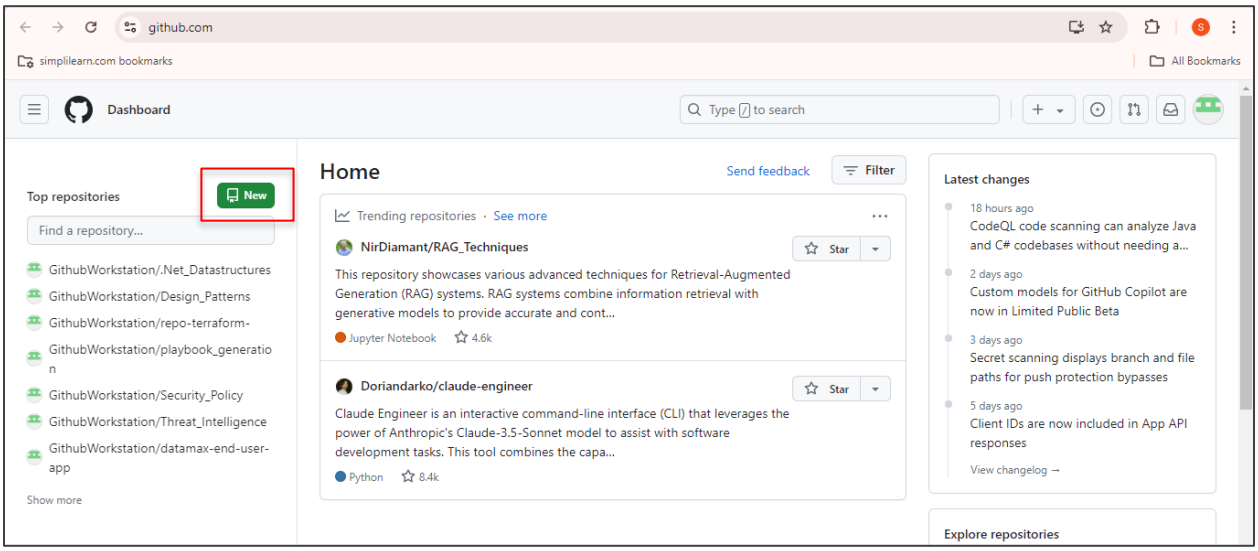
1. Create a React GitHub repository
2. Create an S3 bucket
3. Configure CodeBuild and CodePipeline to perform build and test automation

Step 1: Create a React GitHub repository

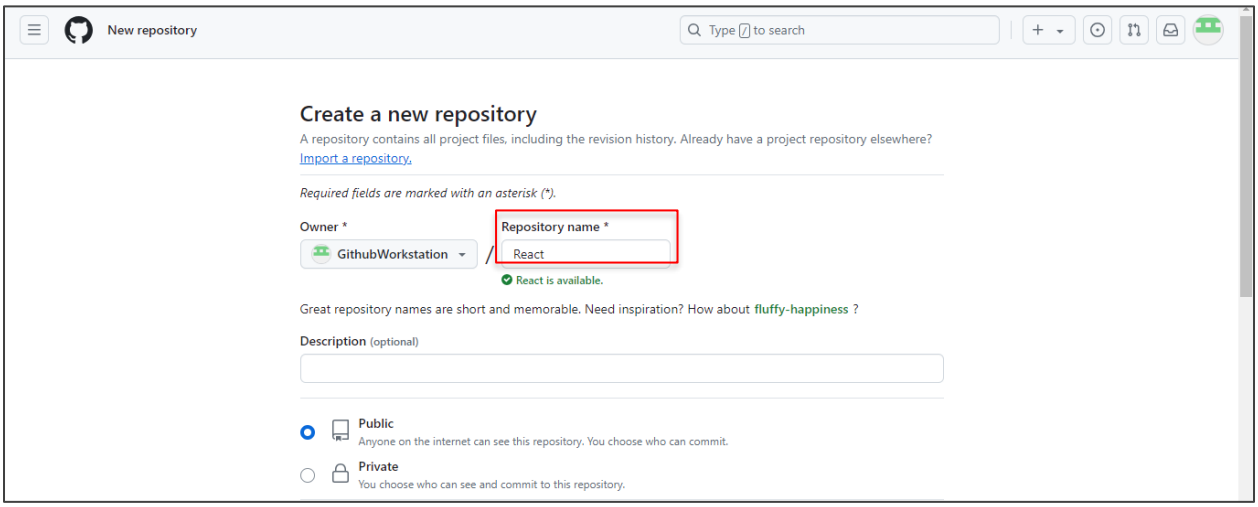
1.1 Go to the following URL and sign in to your GitHub account:
https://github.com/

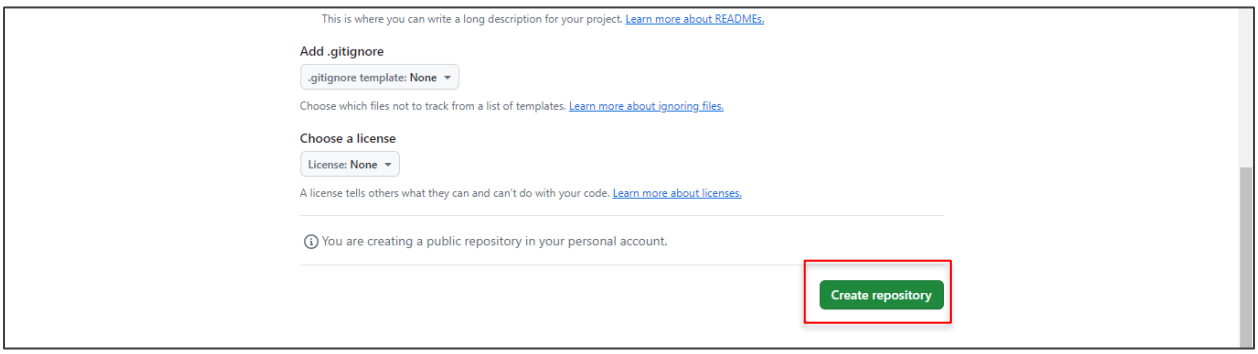


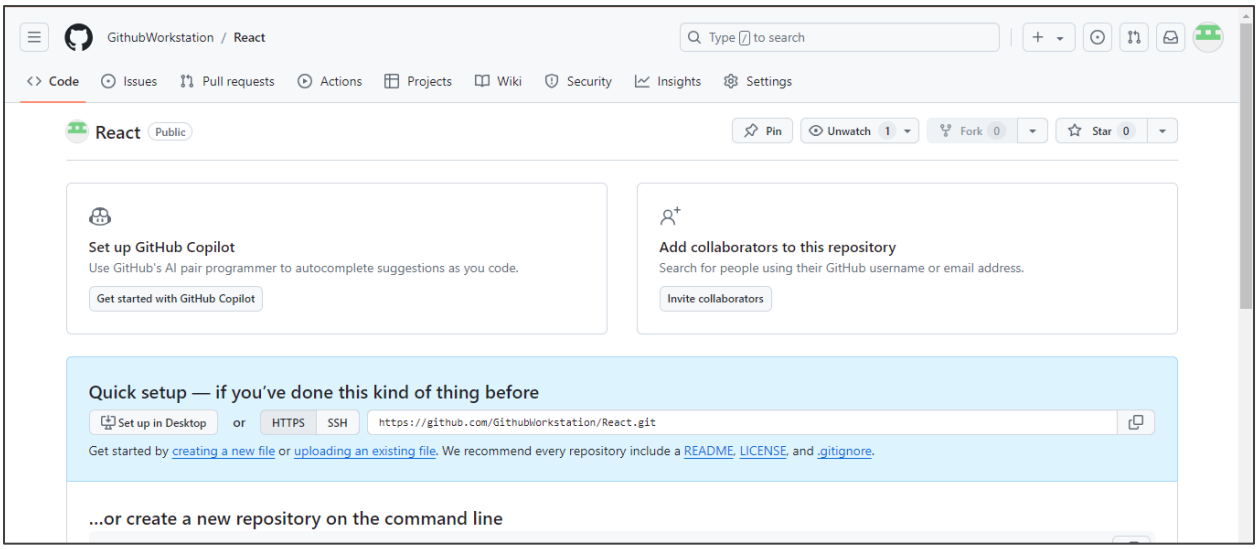
1.2 Click on the **New** button to create a new repository



1.3 Name the repository as **React**, scroll down, and click on **Create repository**

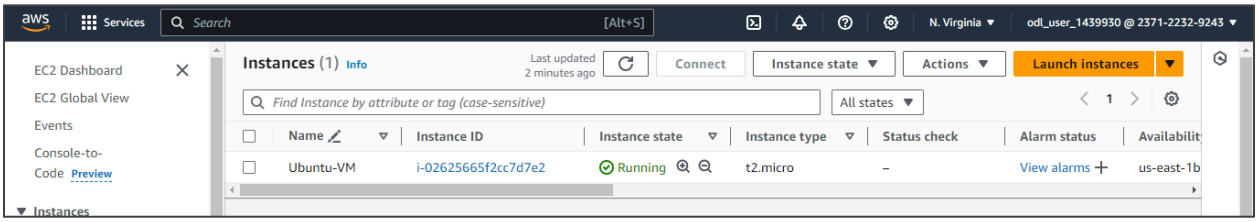






The repository is created successfully.

1.4 Navigate to your AWS Console, create an EC2 instance, and connect to it



1.5 Create a directory using the following command:
mkdir demo

```
ubuntu@ip-172-31-86-89:~$ mkdir demo
ubuntu@ip-172-31-86-89:~$
```

1.6 Navigate inside the created directory using the following command:
cd demo

```
ubuntu@ip-172-31-86-89:~$ cd demo
ubuntu@ip-172-31-86-89:~/demo$
```

1.7 Initialize Git using the following command:
git init

```
ubuntu@ip-172-31-86-89:~/demo$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/ubuntu/demo/.git/
ubuntu@ip-172-31-86-89:~/demo$
```

1.8 Clone the GitHub repository using the following command:
git clone --mirror https://github.com/anujdevopslearn/SonarQubeNodeJS

```
ubuntu@ip-172-31-86-89:~/demo$ git clone --mirror https://github.com/anujdevopslearn/SonarQubeNodeJS
Cloning into bare repository 'SonarQubeNodeJS.git'...
remote: Enumerating objects: 205, done.
remote: Counting objects: 100% (104/104), done.
remote: Compressing objects: 100% (53/53), done.
remote: Total 205 (delta 59), reused 69 (delta 31), pack-reused 101 (from 1)
Receiving objects: 100% (205/205), 212.16 KiB | 12.48 MiB/s, done.
Resolving deltas: 100% (67/67), done.
ubuntu@ip-172-31-86-89:~/demo$
```

1.9 Switch to the cloned repository using the following command:
cd SonarQubeNodeJS.git

```
ubuntu@ip-172-31-86-89:~/demo$ cd SonarQubeNodeJS.git
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$
```

1.10 List all files present in the repository using the following command:
ls -lart

```
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$ ls -lart
total 44
drwxrwxr-x 2 ubuntu ubuntu 4096 Aug 29 12:13 info
drwxrwxr-x 2 ubuntu ubuntu 4096 Aug 29 12:13 hooks
-rw-rw-r-- 1 ubuntu ubuntu   73 Aug 29 12:13 description
drwxrwxr-x 2 ubuntu ubuntu 4096 Aug 29 12:13 branches
drwxrwxr-x 4 ubuntu ubuntu 4096 Aug 29 12:13 ..
drwxrwxr-x 4 ubuntu ubuntu 4096 Aug 29 12:13 refs
drwxrwxr-x 4 ubuntu ubuntu 4096 Aug 29 12:13 objects
-rw-rw-r-- 1 ubuntu ubuntu  181 Aug 29 12:13 config
-rw-rw-r-- 1 ubuntu ubuntu  165 Aug 29 12:13 packed-refs
-rw-rw-r-- 1 ubuntu ubuntu   23 Aug 29 12:13 HEAD
drwxrwxr-x 7 ubuntu ubuntu 4096 Aug 29 12:13 .
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$
```

1.11 List all remote repositories using the following command:
git remote -v

```
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$ git remote -v
origin  https://github.com/anujdevopslearn/SonarQubeNodeJS (fetch)
origin  https://github.com/anujdevopslearn/SonarQubeNodeJS (push)
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$
```

1.12 Remove the existing remote configuration using the following command:
git remote remove origin

```
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$ git remote remove origin
Note: Some branches outside the refs/remotes/ hierarchy were not removed;
to delete them, use:
  git branch -d develop
  git branch -d master
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$
```

1.13 Add a new remote repository using the following command:
git remote add origin Repository-link

```
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$ git remote add origin https://github.com/GithubWorkstation/React.git
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$
```

Note: Replace **Repository-link** with the link to the repository you created in Step 1.3

1.14 Verify if the new remote repository is correctly configured by using the following command:
git remote -v

```
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$ git remote -v
origin  https://github.com/GithubWorkstation/React.git (fetch)
origin  https://github.com/GithubWorkstation/React.git (push)
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$
```

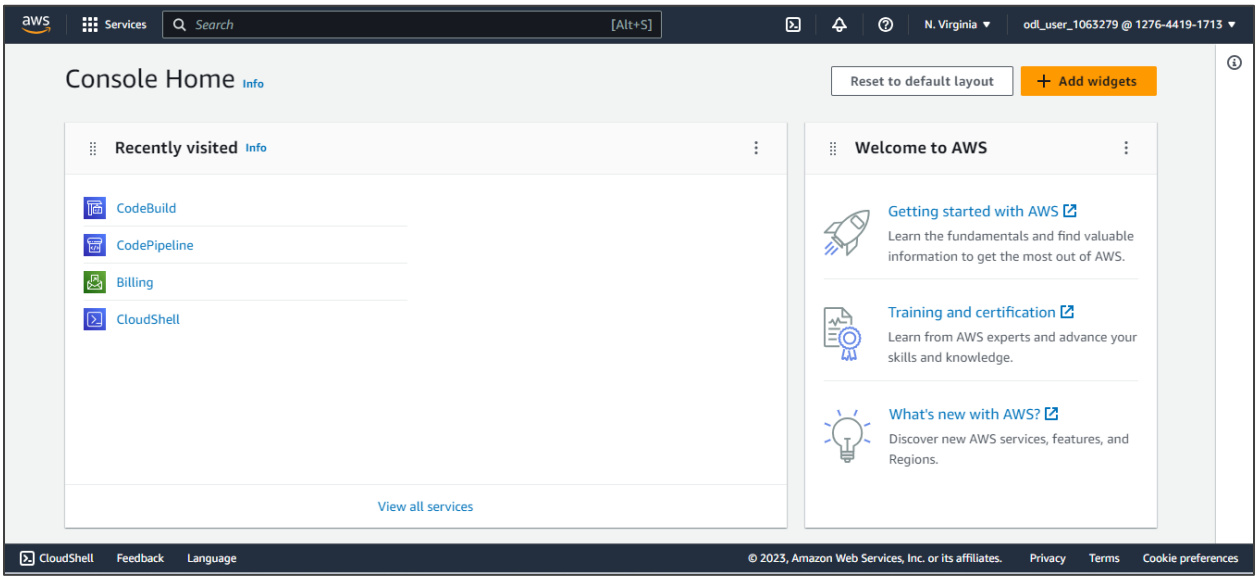
1.15 Upload your local repository content to the remote repository using the following command:
git push --all origin

```
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$ git push --all origin
Username for 'https://github.com': GithubWorkstation
Password for 'https://GithubWorkstation@github.com':
Everything up-to-date
ubuntu@ip-172-31-86-89:~/demo/SonarQubeNodeJS.git$
```

Note: When prompted, enter your GitHub credentials.

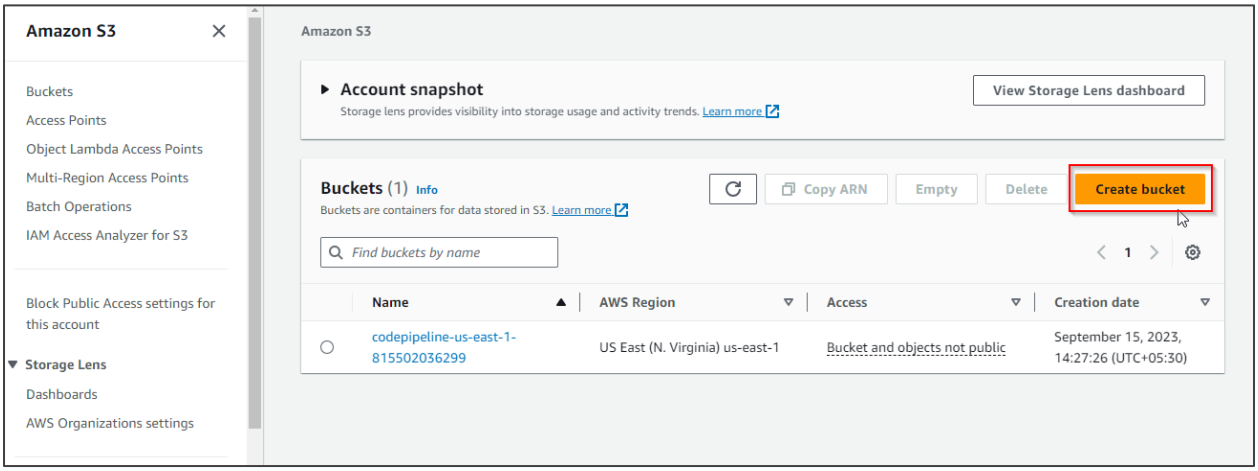
Step 2: Create an S3 bucket

2.1 Go to the AWS Console

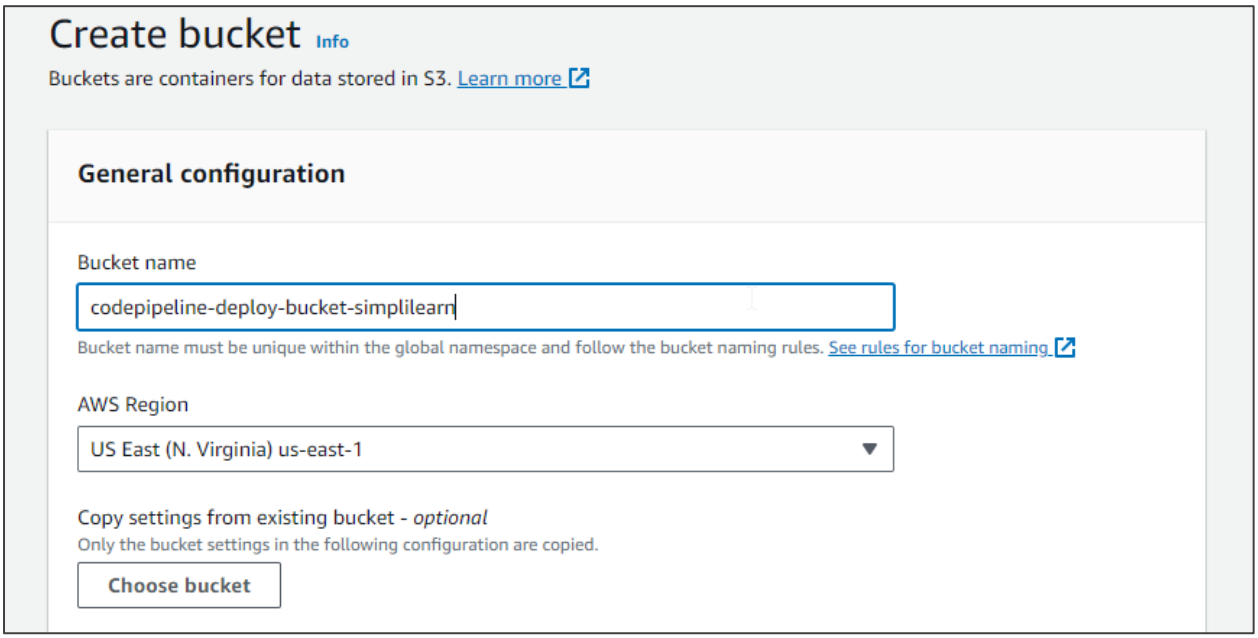


2.2 Search for **S3** and click on it

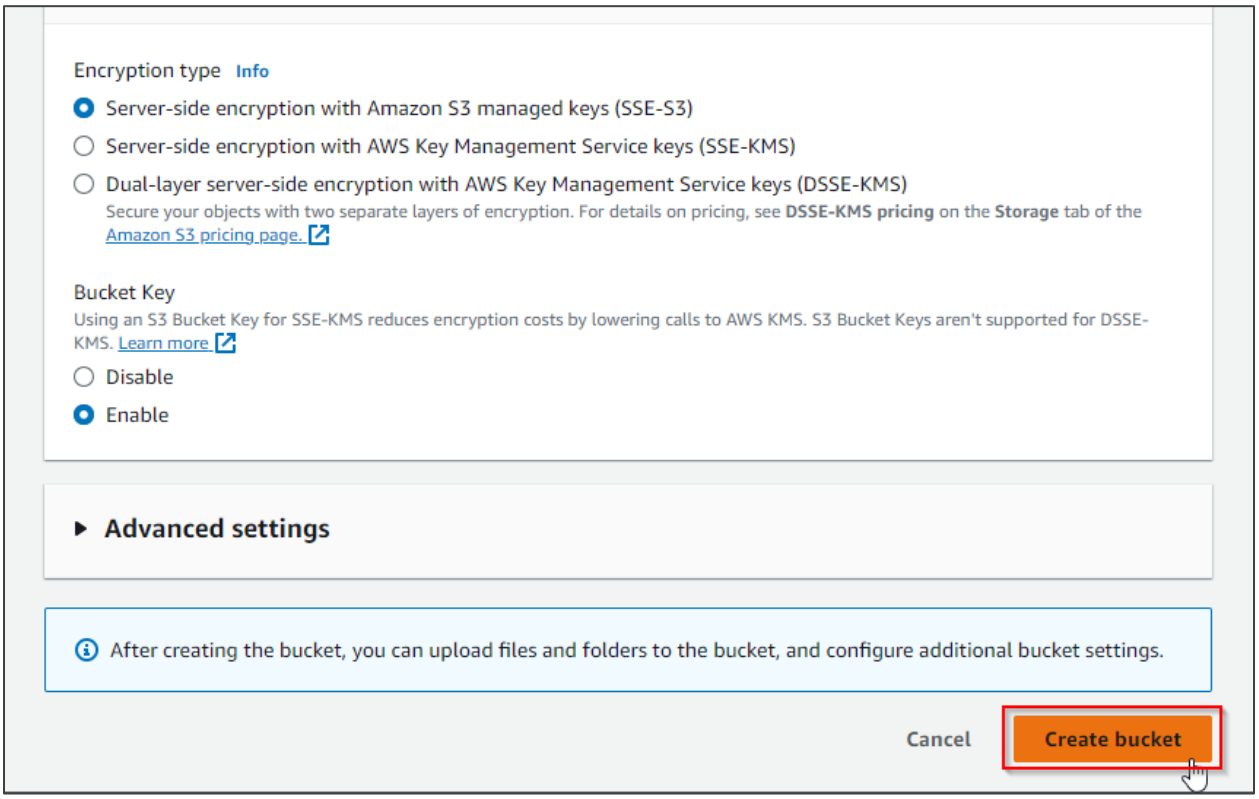
2.3 Click on the **Create bucket** button

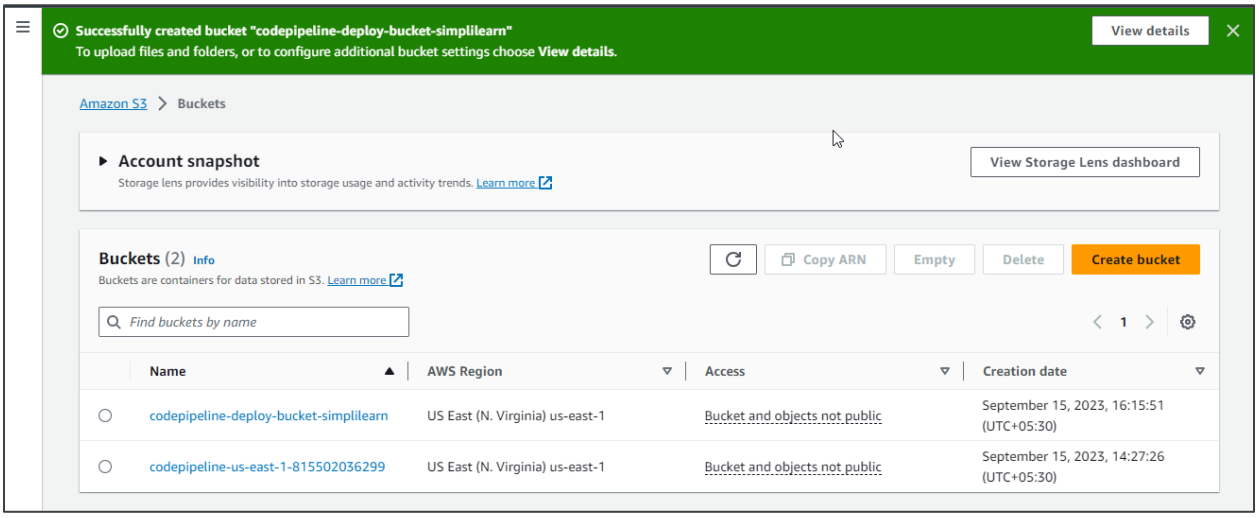


2.4 Enter the name of the bucket



2.5 Scroll down and click on the **Create bucket** button

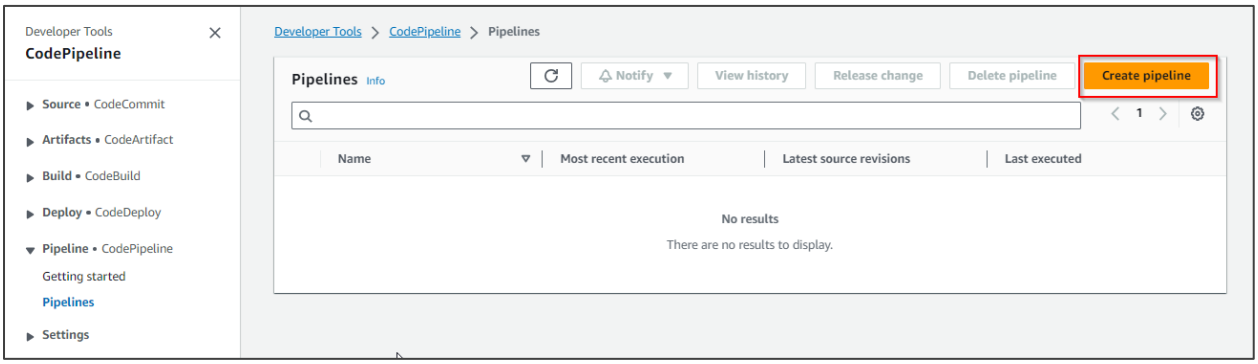




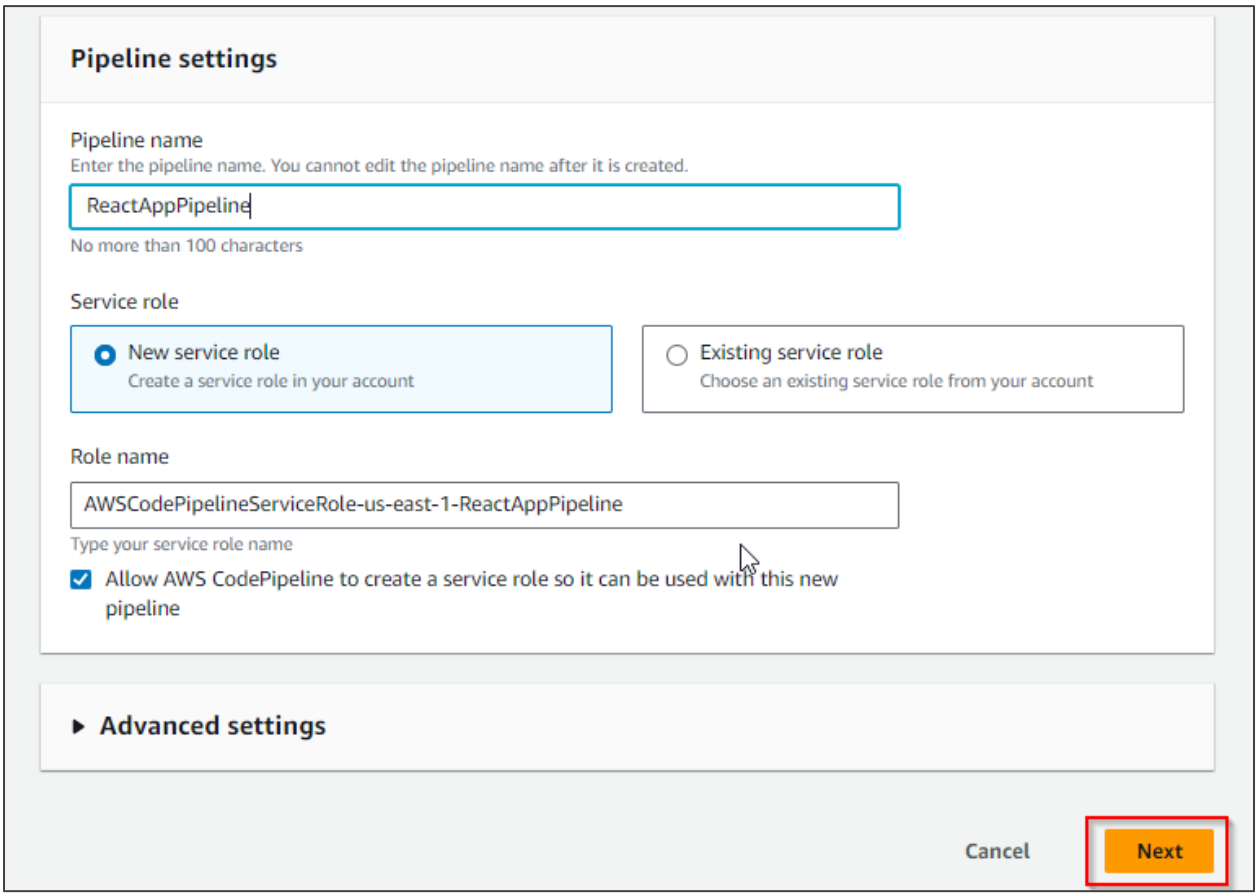
The S3 bucket is created successfully.

Step 3: Configure CodeBuild and CodePipeline to perform build and test automation

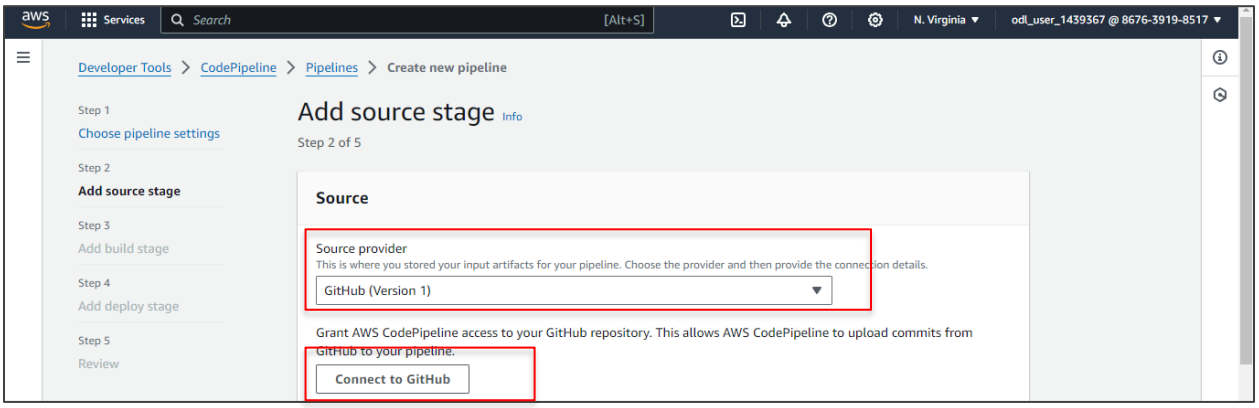
3.1 Click on the **Create pipeline** button



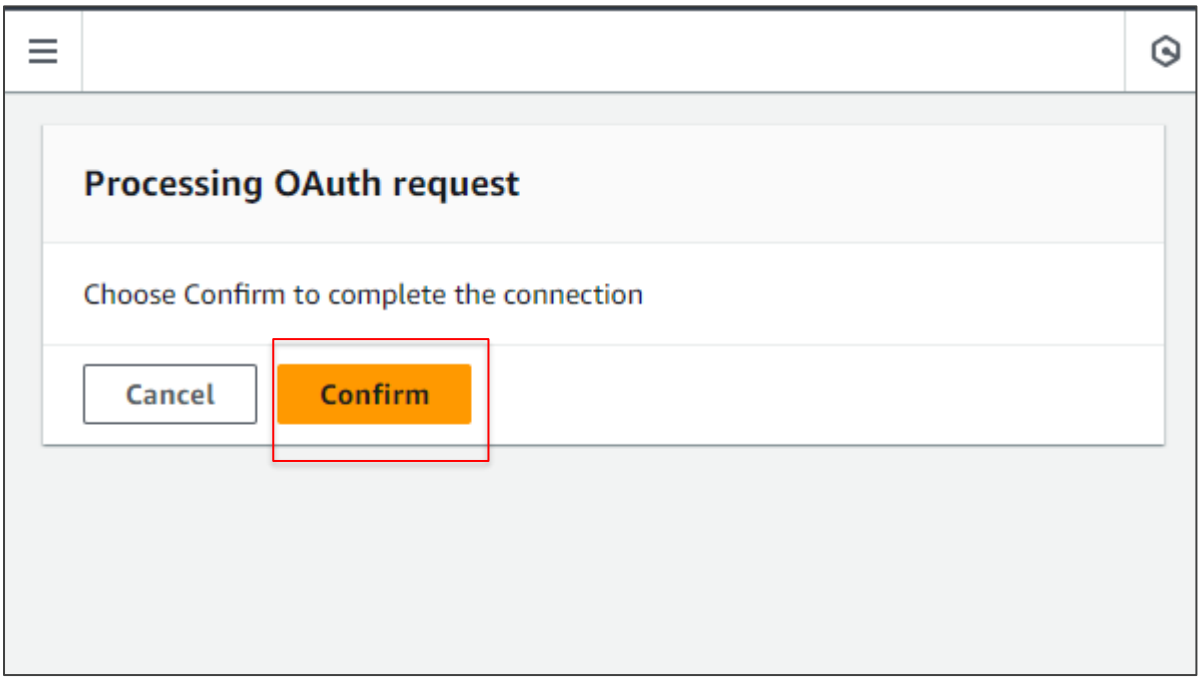
3.2 Enter the **Pipeline name**, and click the **Next** button

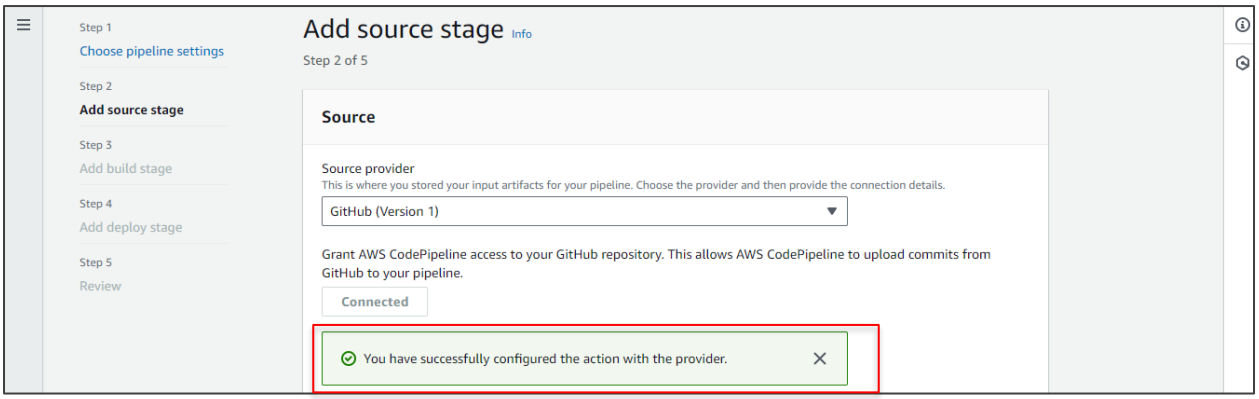


3.3 Select **GitHub (Version 1)** as the **Source provider** and click on **Connect to GitHub**



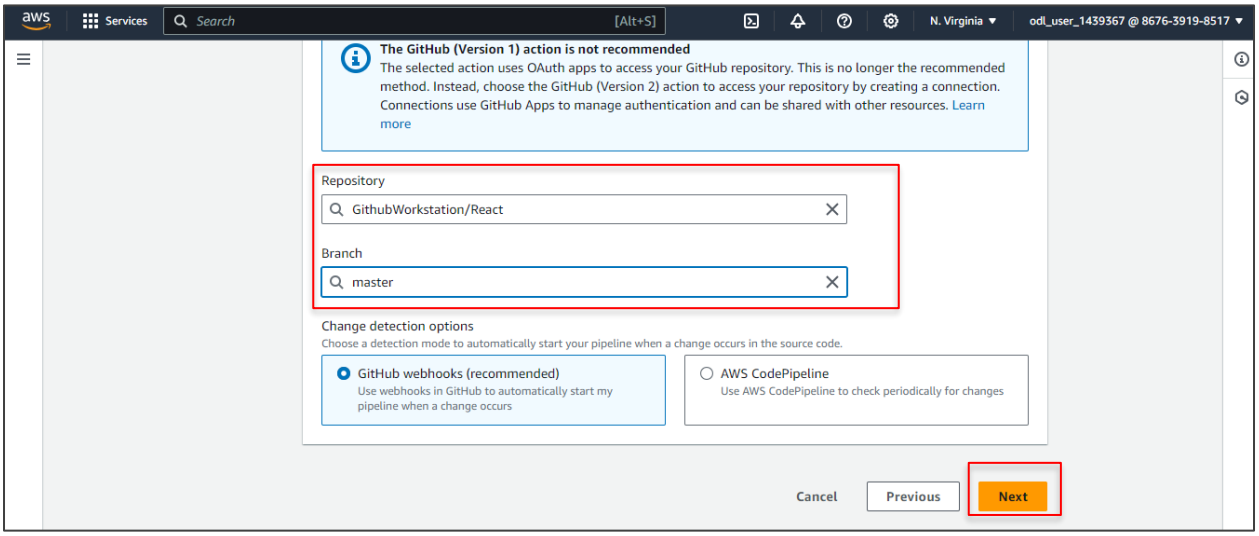
3.4 When prompted, click on **Confirm**



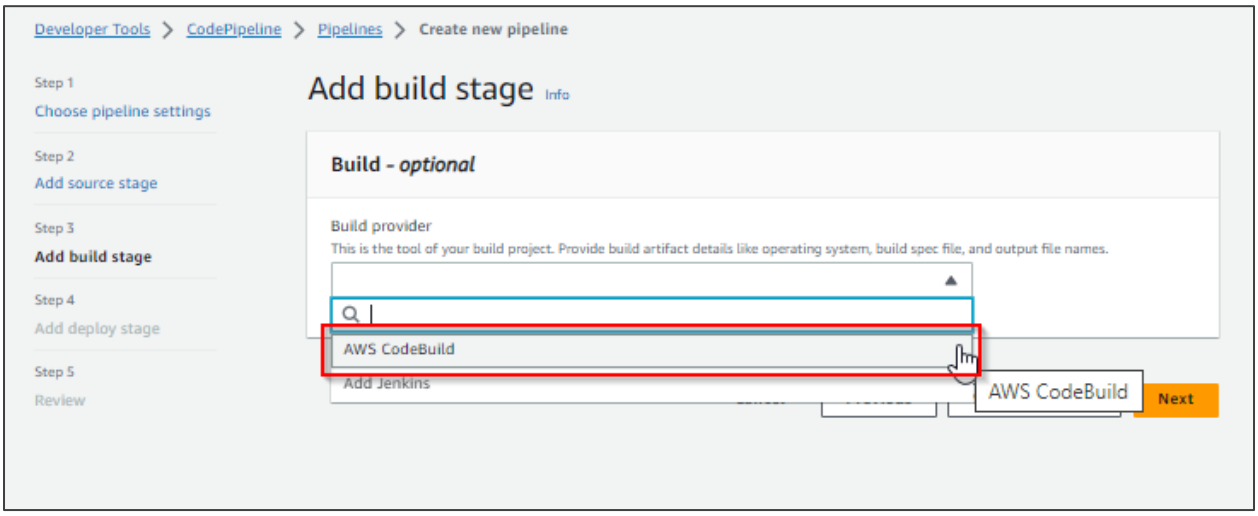


GitHub is configured successfully.

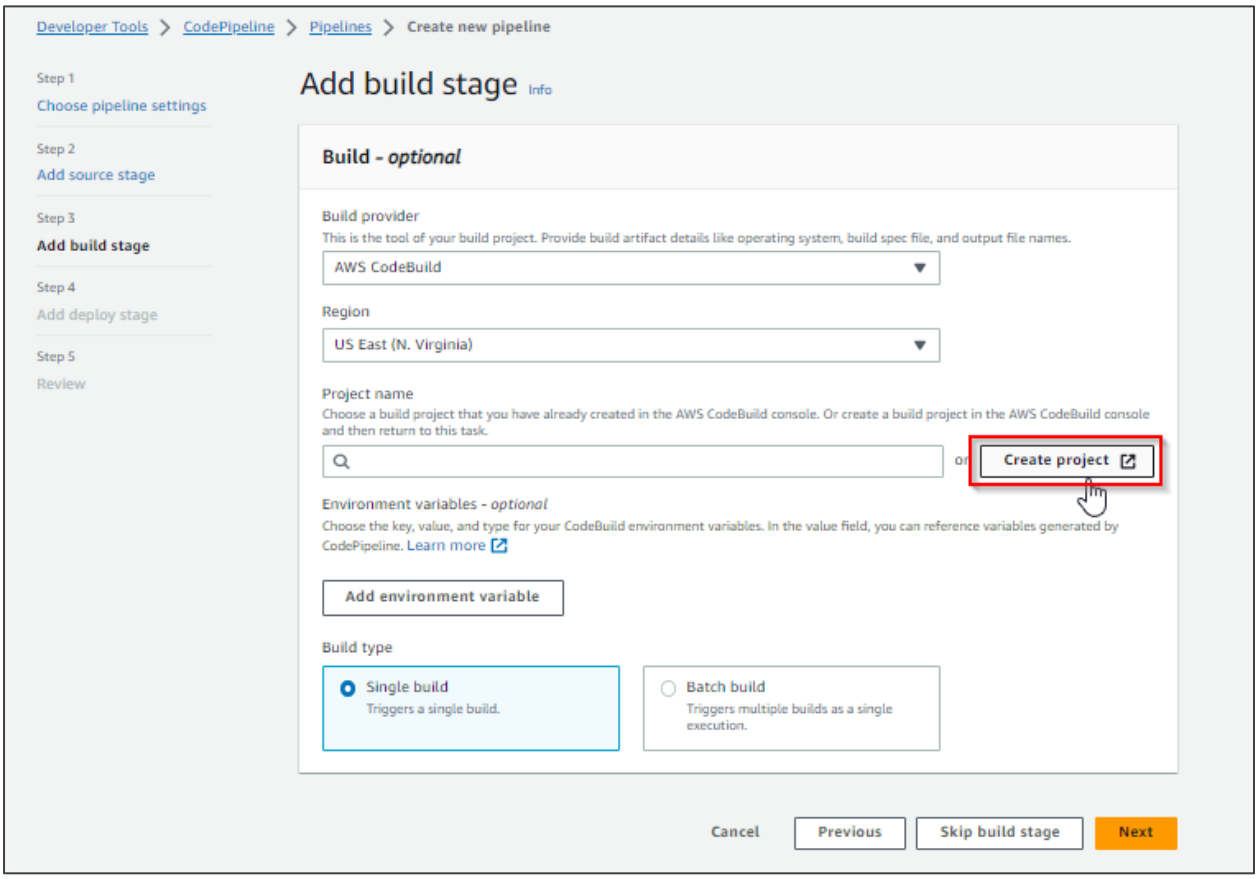
3.5 Enter the created repository name, select **master** under the branch section, and click on **Next**



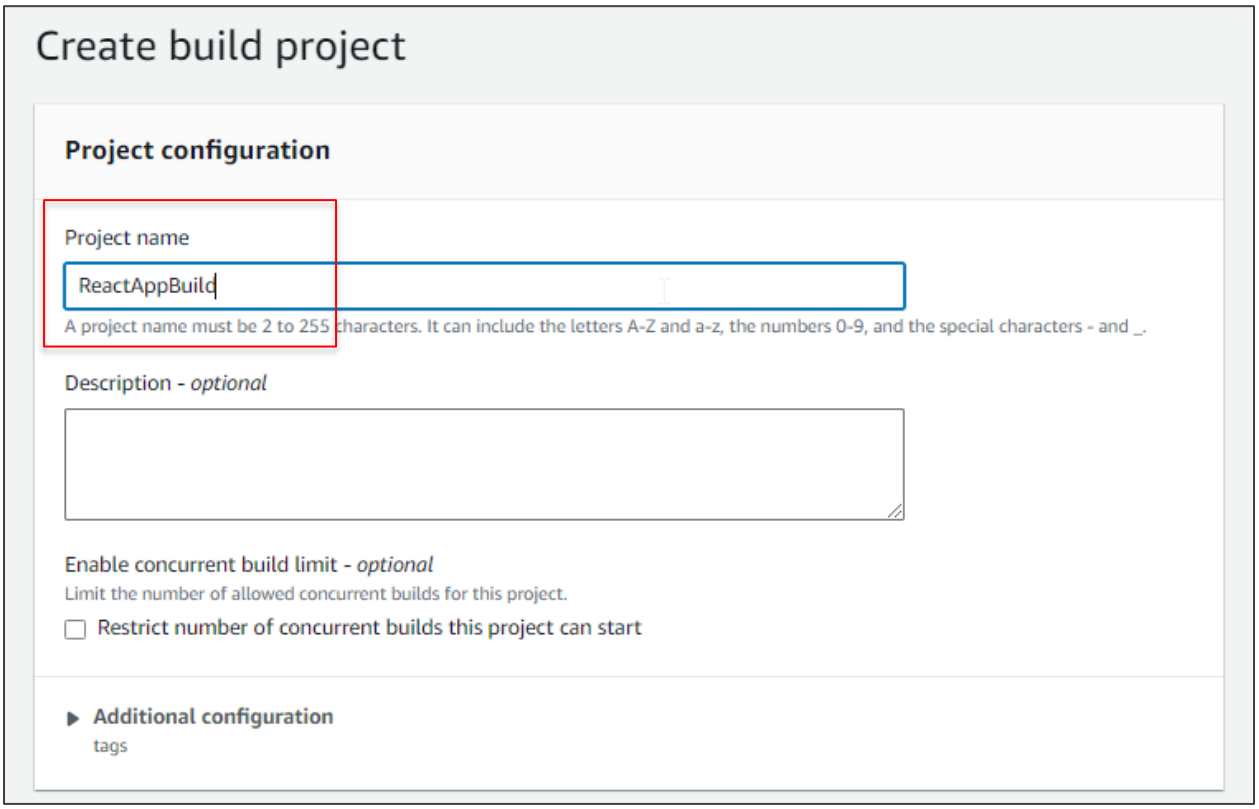
3.6 Select **AWS CodeBuild** as the **Build provider** for building and automating tests



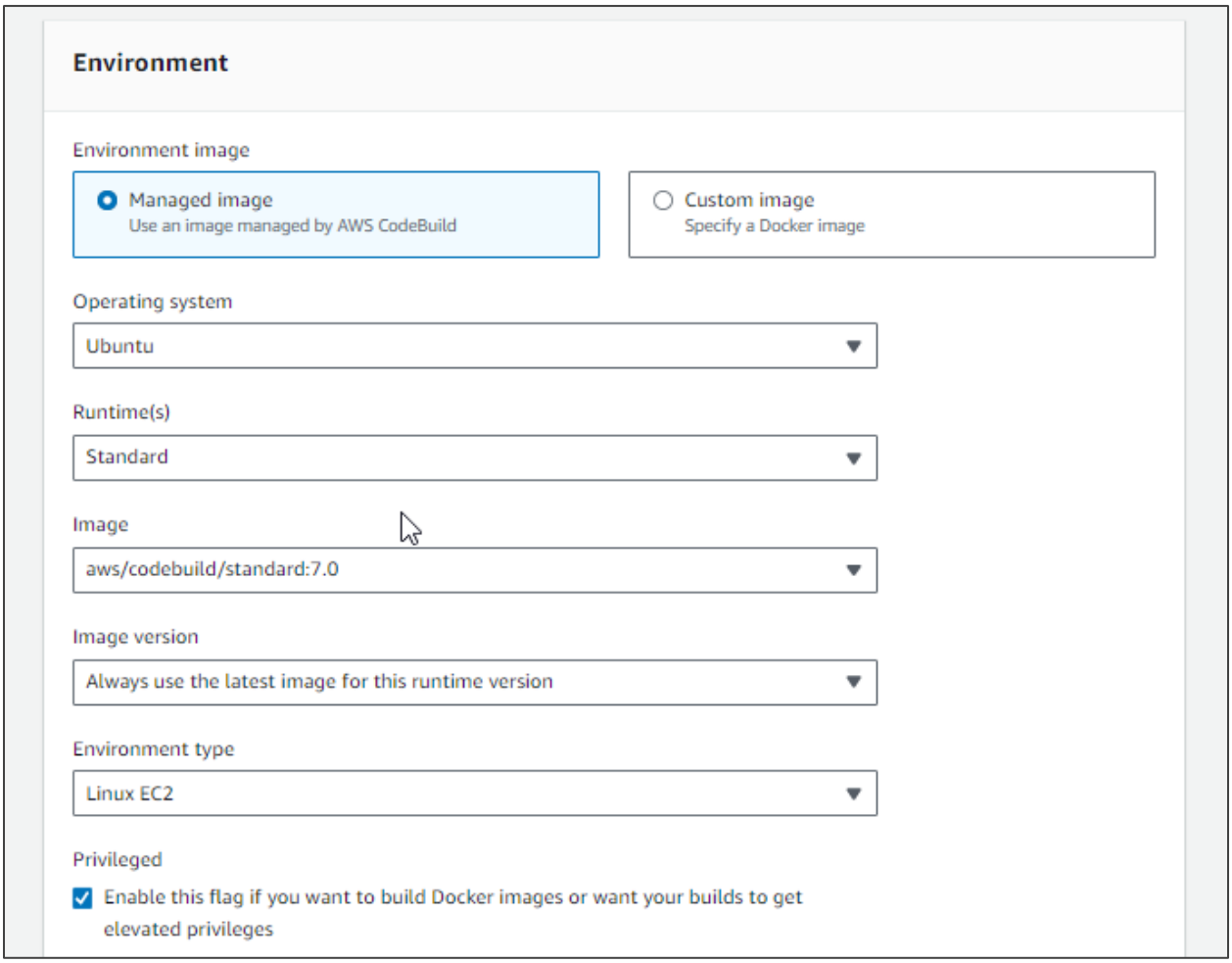
3.7 Then, click on the **Create project** button, and a new pop-up window will appear

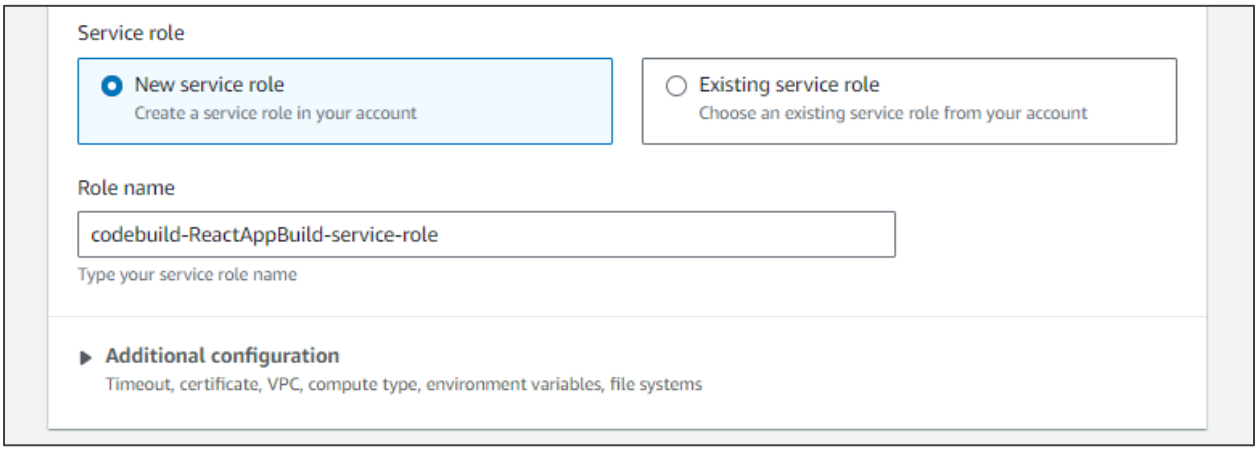


3.8 In the new window, enter a name for the project

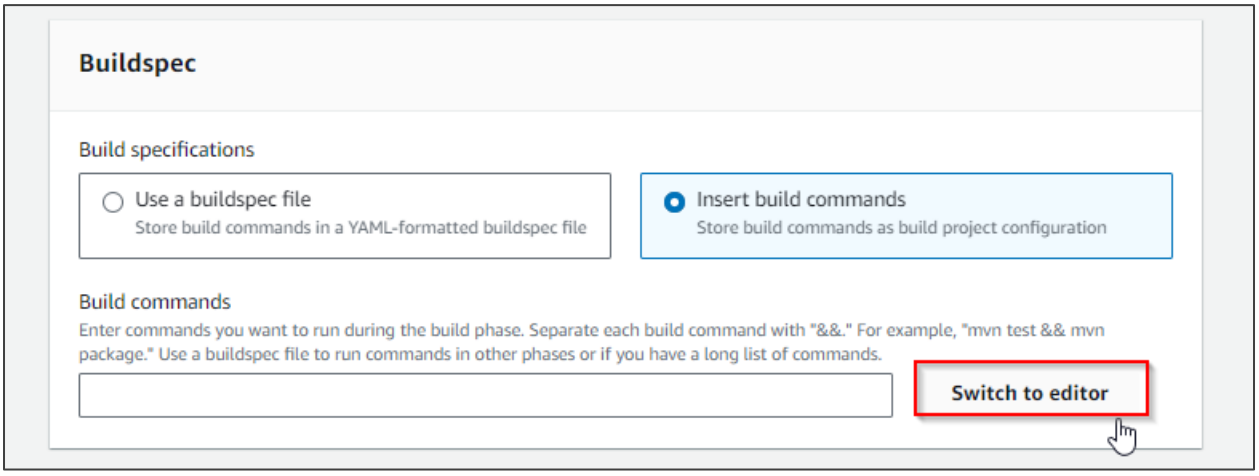


3.9 In the **Environment** section, add the details as shown in the following screenshots:



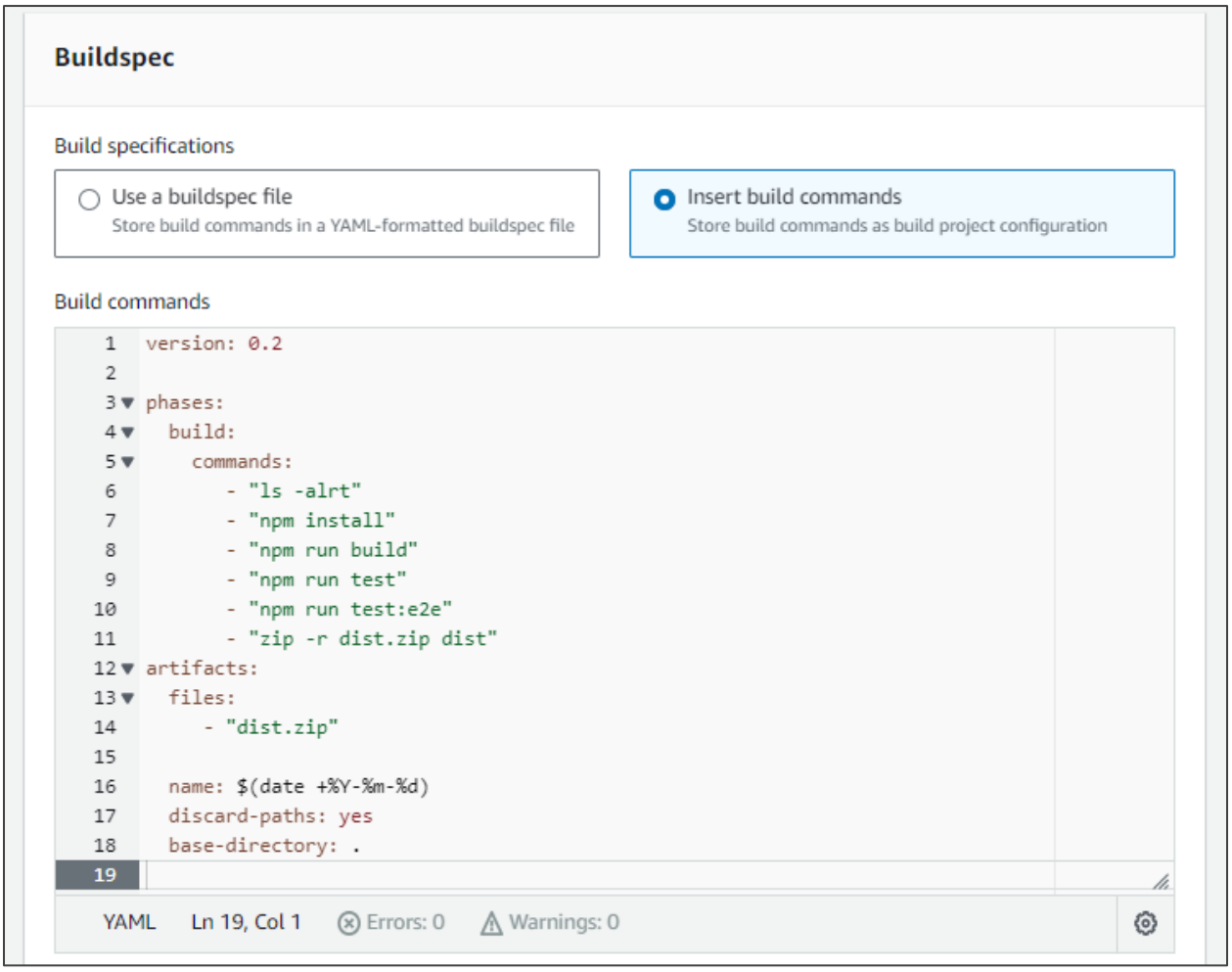


3.10 In the **Buildspec** section, select the **Insert build commands** option and click on the **Switch to editor** button

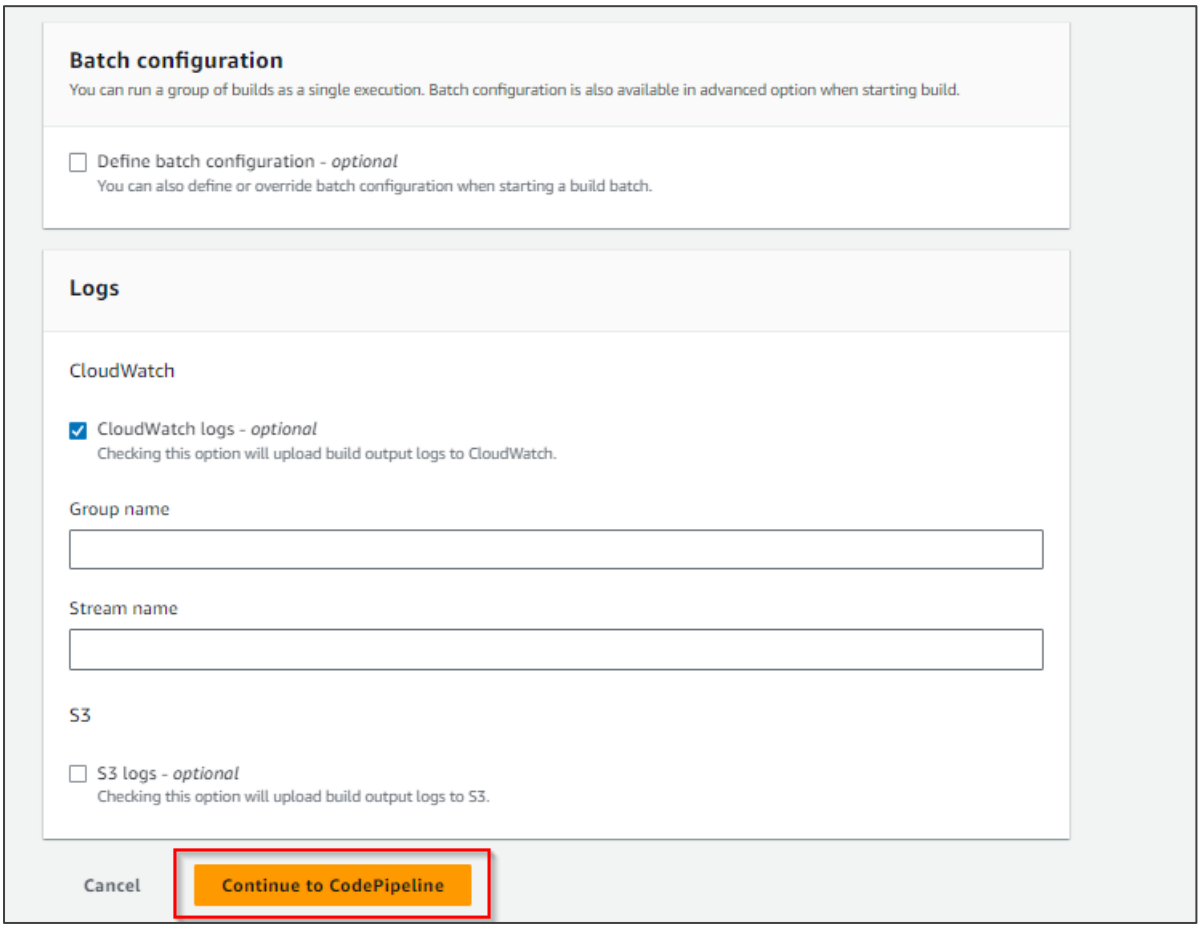


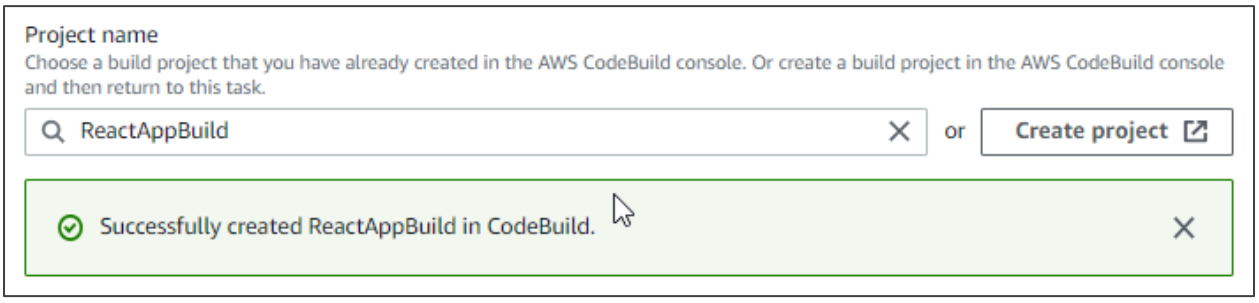
3.11 Remove the existing build commands from the editor and enter the following YAML code:

```
version: 0.2

phases:
  build:
    commands:
      - "ls -alrt"
      - "npm install"
      - "npm run build"
      - "npm run test"
      - "npm run test:e2e"
      - "zip -r dist.zip dist"
artifacts:
  files:
    - "dist.zip"

  name: $(date +%Y-%m-%d)
  discard-paths: yes
  base-directory: .
```


Buildspec
Build specifications
Use a buildspec file
Store build commands in a YAML-formatted buildspec file
Insert build commands
Store build commands as build project configuration
Build commands
1 version: 0.2
2
3 phases:
4 build:
5 commands:
6 - "ls -alrt"
7 - "npm install"
8 - "npm run build"
9 - "npm run test"
10 - "npm run test:e2e"
11 - "zip -r dist.zip dist"
12 artifacts:
13 files:
14 - "dist.zip"
15
16 name: $(date +%Y-%m-%d)
17 discard-paths: yes
18 base-directory: .
19
YAML
Ln 19, Col 1
Errors: 0
Warnings: 0

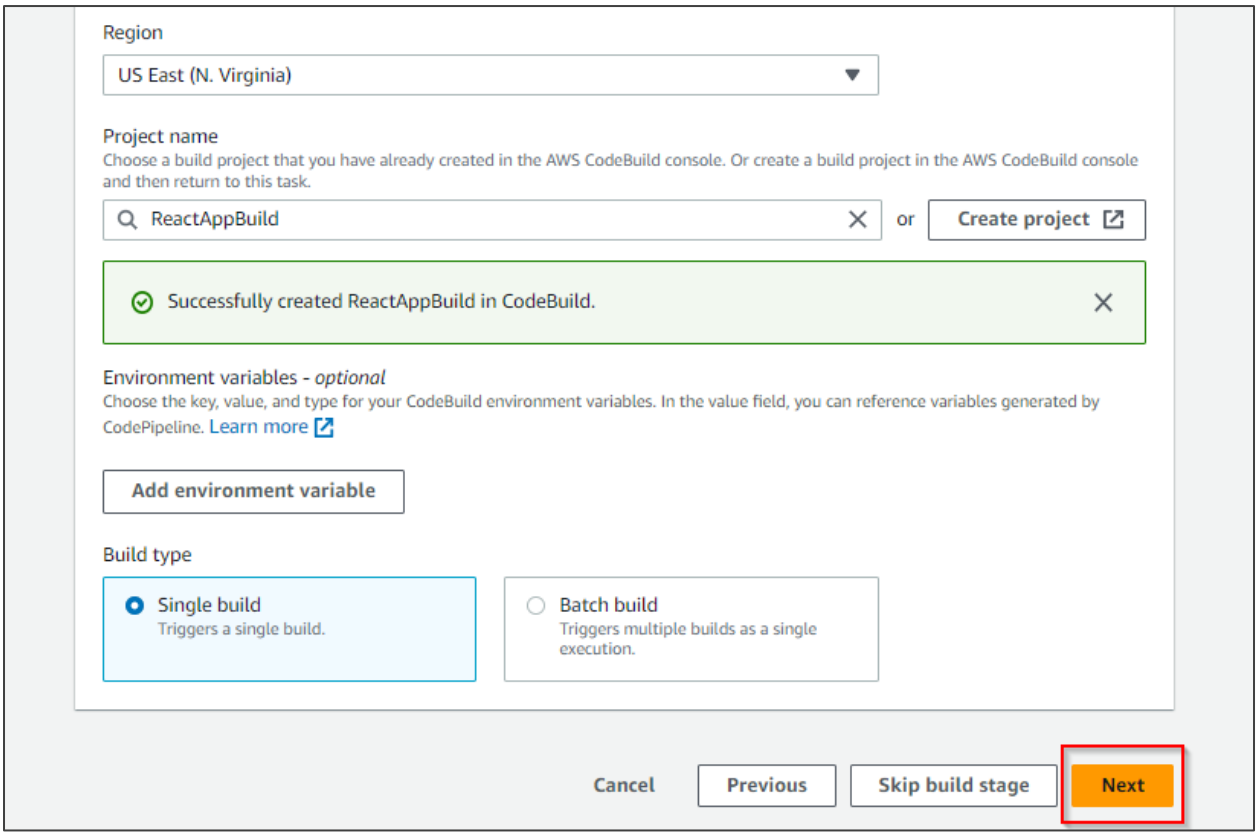
3.12 Scroll down to the bottom of the page and click on the **Continue to CodePipeline** button



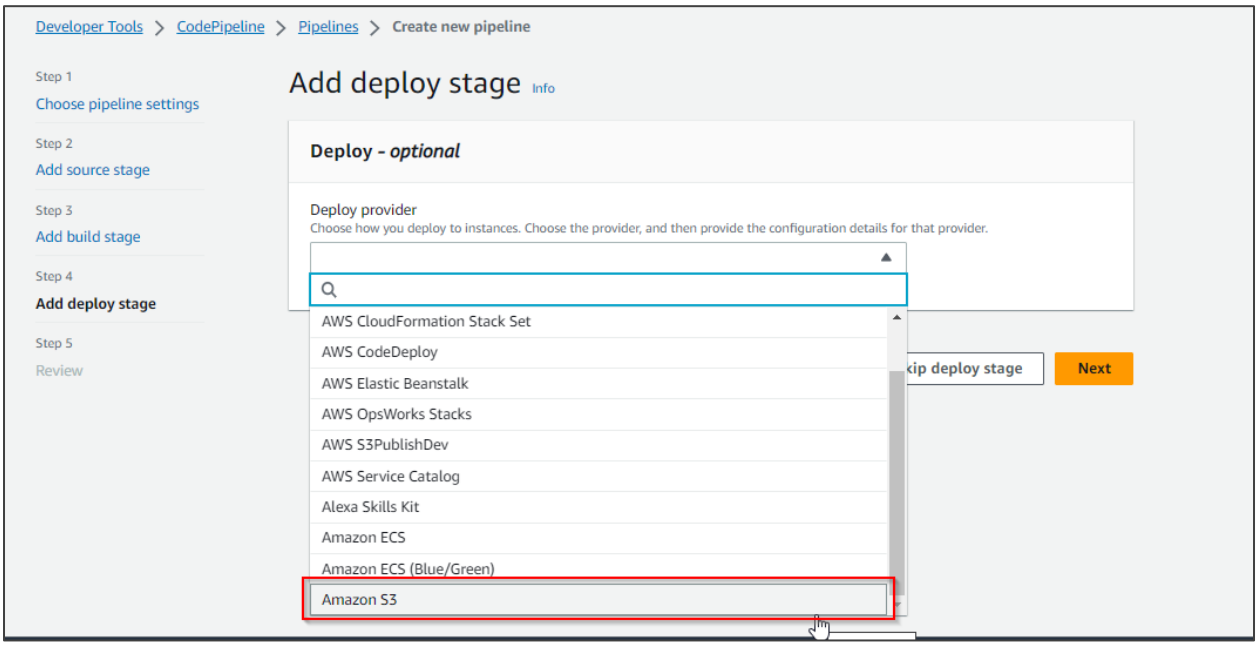


The project is successfully created.

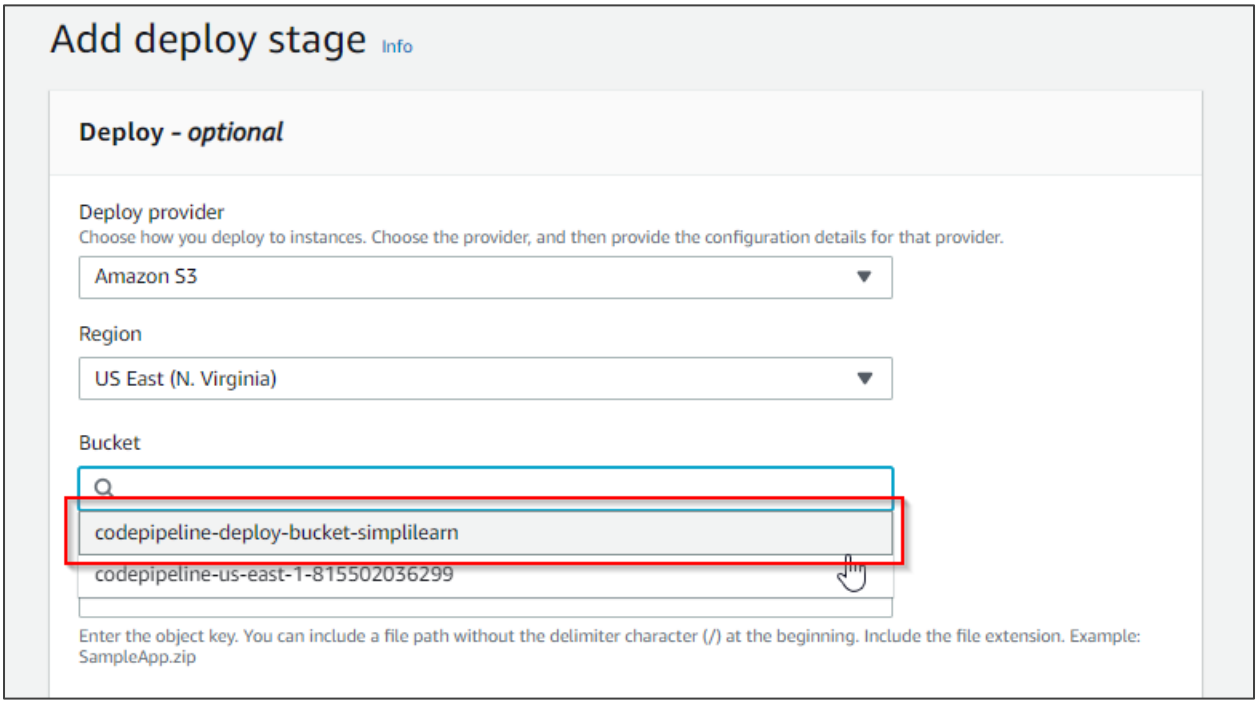
3.13 Now, click the **Next** button



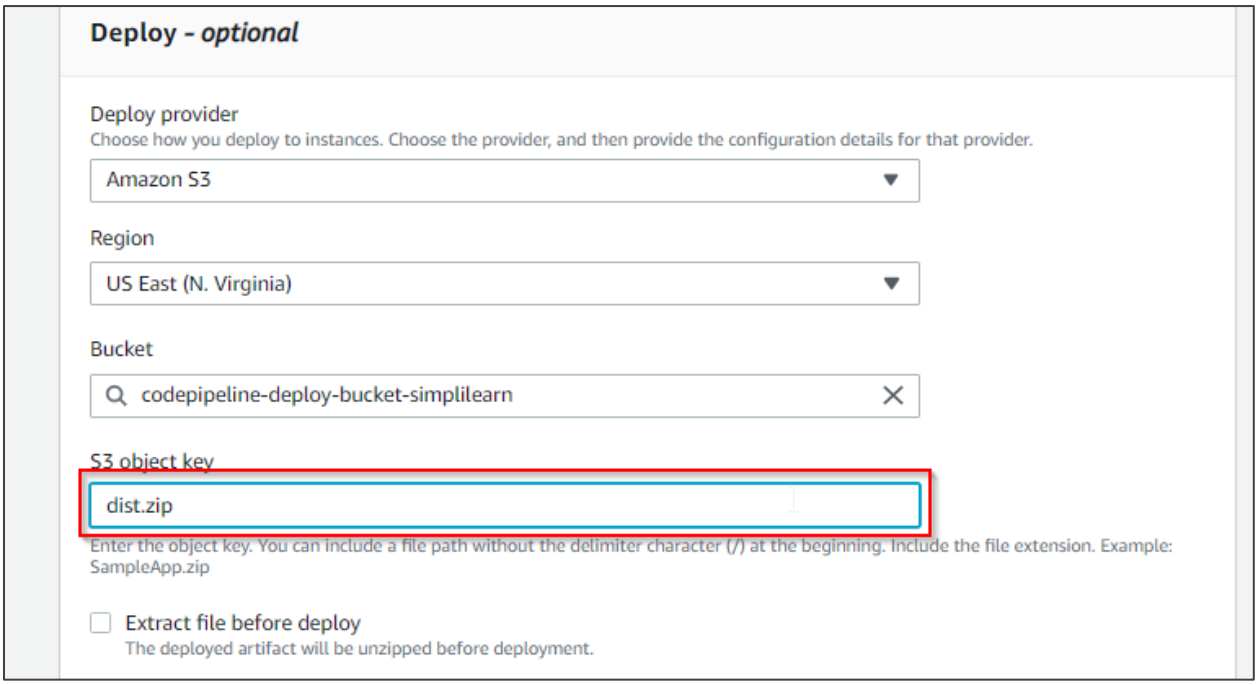
3.14 In the deploy stage, select the **Amazon S3** option



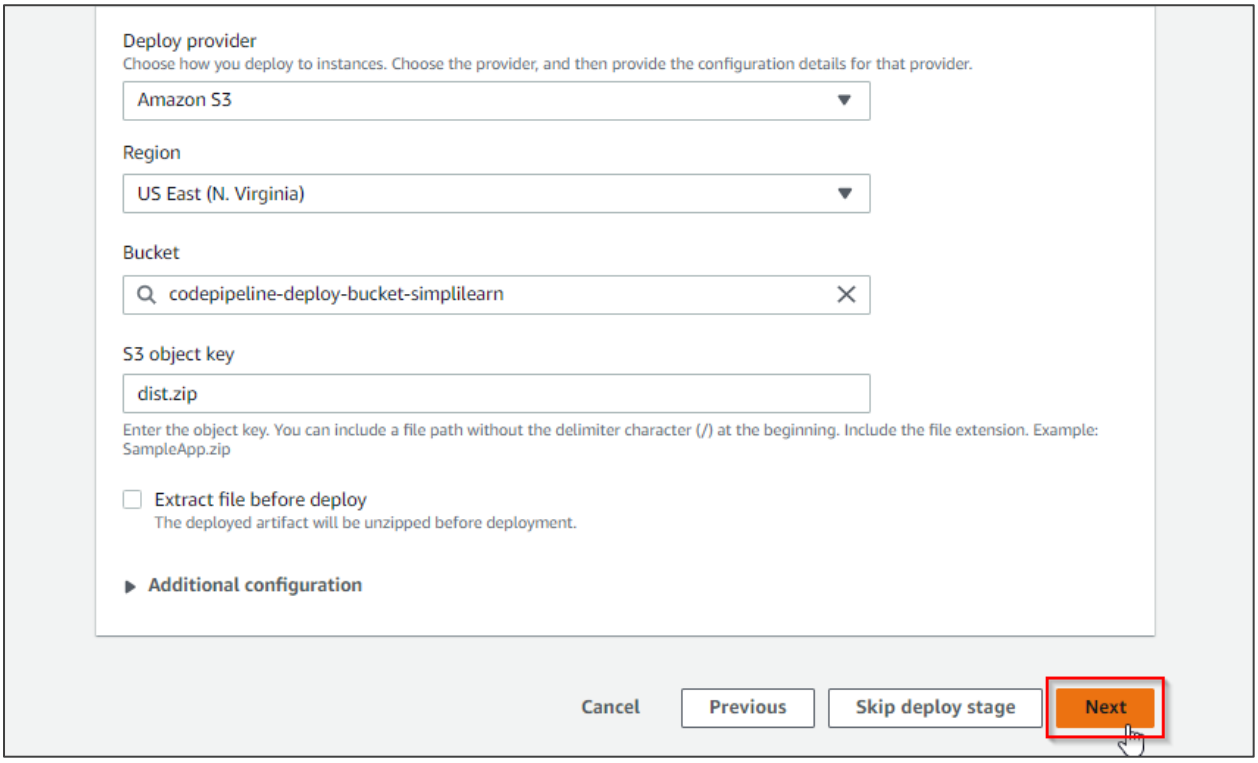
3.15 Select the **codepipeline-deploy-bucket-simplilearn** option under **Bucket**



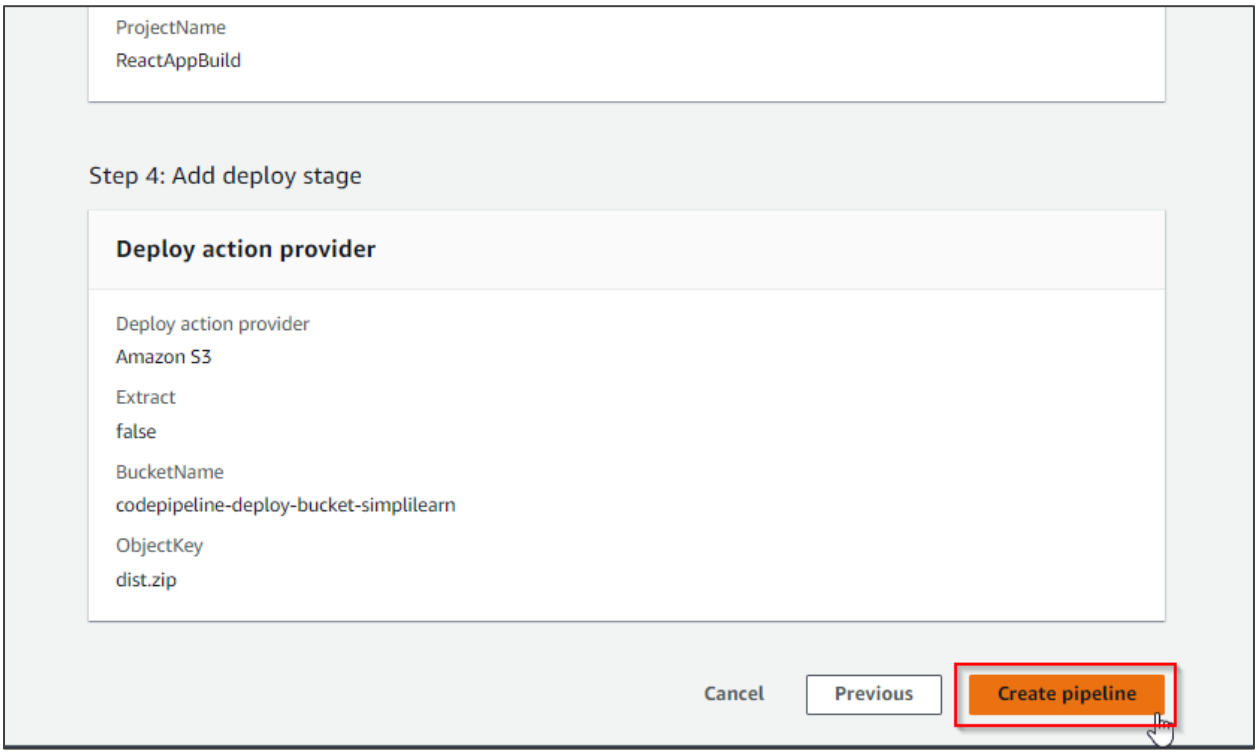
3.16 Enter **dist.zip** in the **S3 object key** section



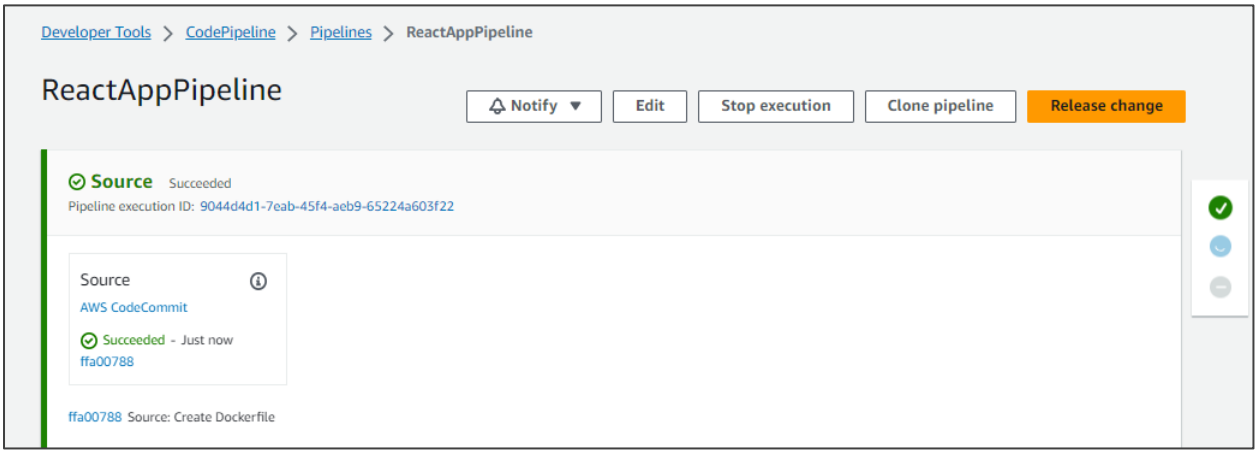
3.17 Click on the **Next** button



3.18 Scroll to the bottom of the page and click on the **Create pipeline** button

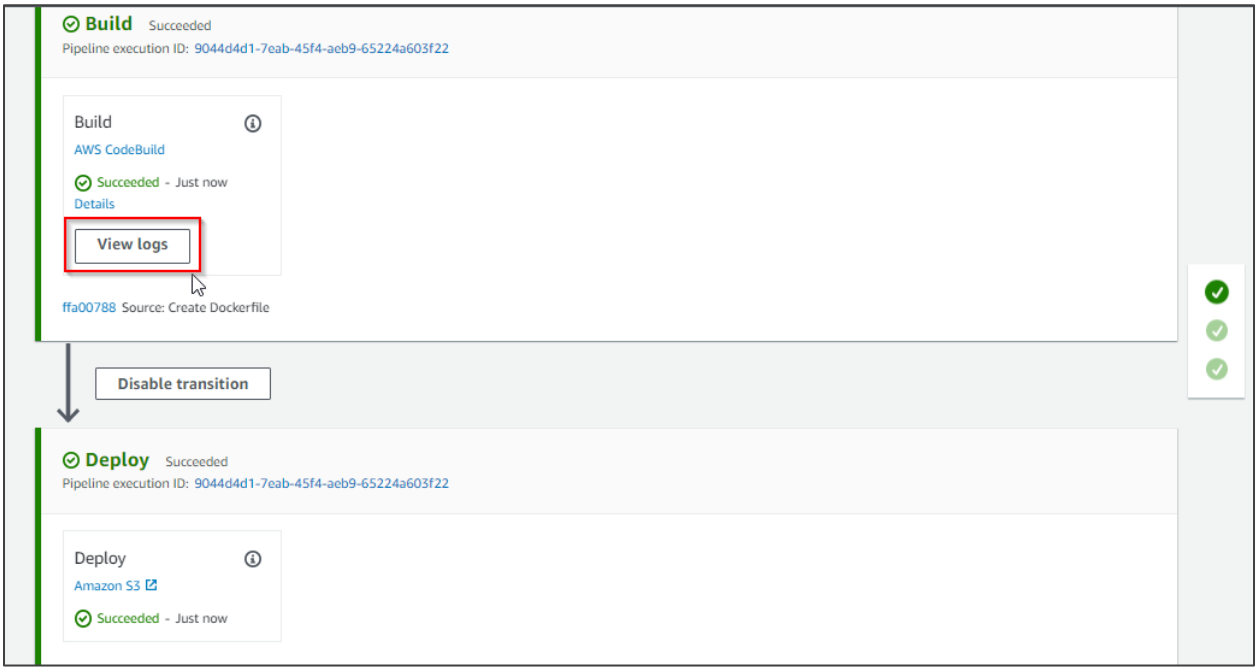


After creating the pipeline, the execution will start.



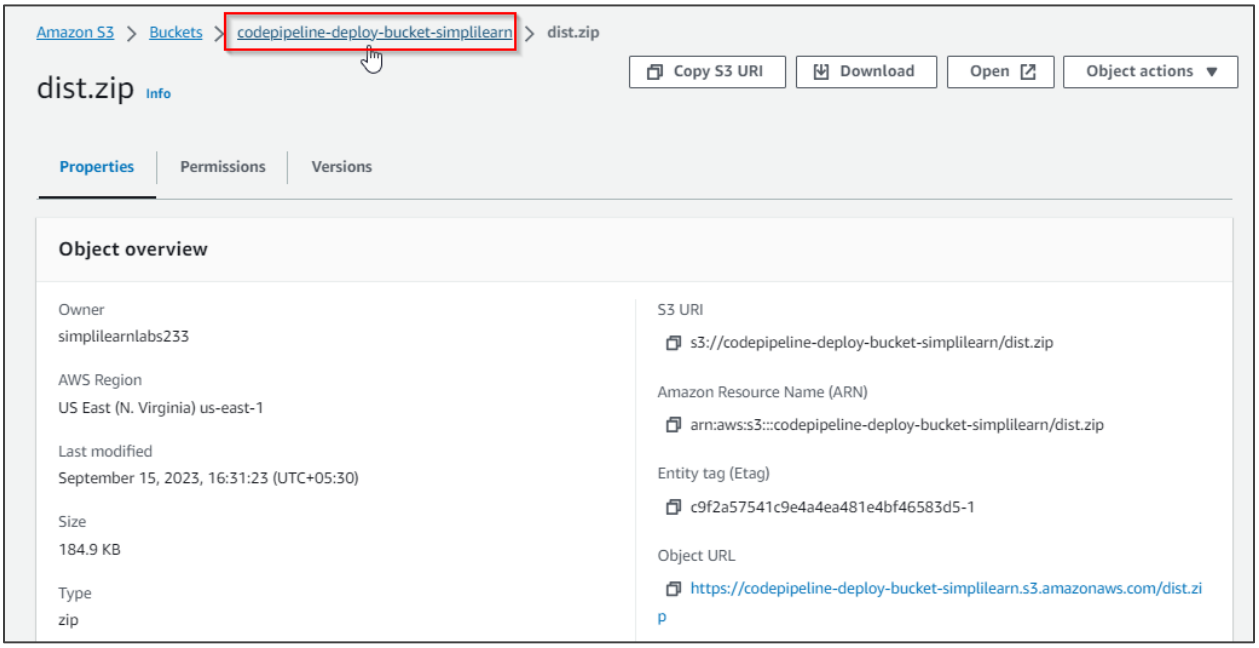

Build
In progress
Pipeline execution ID: 9044d4d1-7eab-45f4-aeb9-65224a603f22
Build
AWS CodeBuild
In progress - 1 minute ago
Details
View logs
ffa00788 Source: Create Dockerfile
Disable transition
Deploy
Didn't Run
Deploy
Amazon S3
Didn't Run
No executions yet

3.19 Once the build stage is complete, click on the **View logs** button in the **Build** section to validate the process

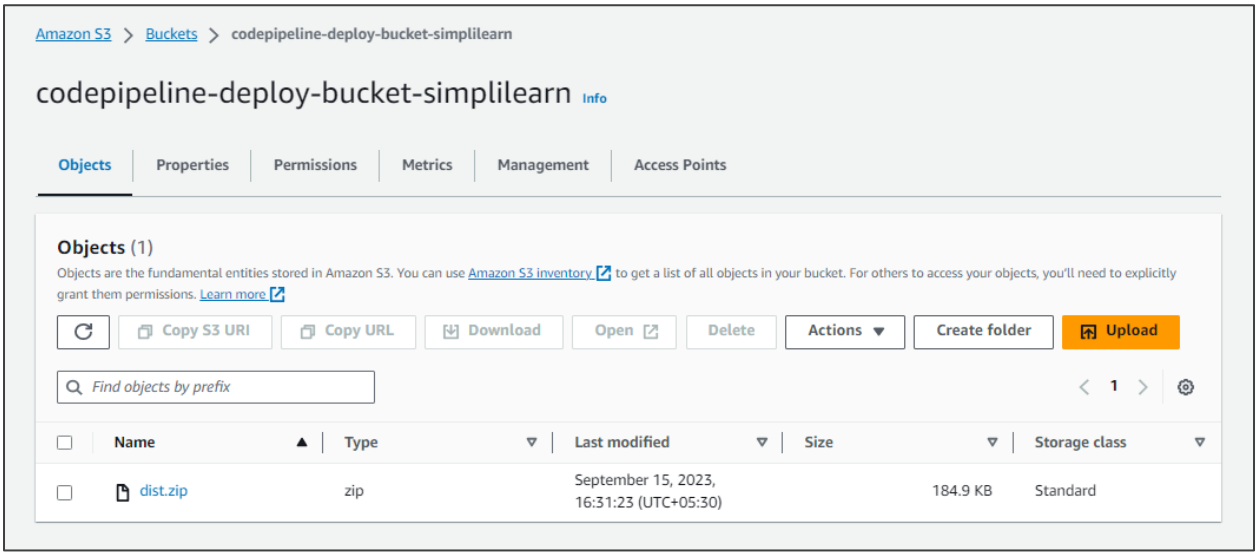


3.20 Once the **Build** and **Deploy** stages are complete, validate whether the artifact was deployed to the S3 bucket by clicking on **Amazon S3**

3.21 Now, click on the bucket name **codepipeline-deploy-bucket-simplilearn**



The following interface appears after successfully building and deploying a React application with AWS CodeBuild and S3:



By following these steps, you have successfully completed the process of building and deploying a React application using AWS services, including CodeBuild and S3.