Serverless CICD Project for Contact Form

Project Description:

We are about to create a serverless CI/CD process for a ***serverless contact form***. So whenever a user hits the website, the user can able to give their details and after clicking on submit the page will return response as "*Thank you*" . After completing the project we will do integration with ***some AWS services*** and achieve CI/CD for to enhance better automation and reduce repeated process and time consumption for developers.

Prerequites:

For this project you need to have the following files

- /
  - website/
    - index.html → For customer details
    - style.css → For the style of index file
    - script.js → For s3 website hosting code logic
  - lambda/
    - index.js → Lambda code to create aws resources and project logic
    - package.json → Dependencies of node modules

Serverless.yaml --→ for to create lambda, api gateway, dynamodb [after serverless deploy serverless will create a .serverless folder and within that it will create two files --→ ***cloudformation-createstack.json, cloudformation-updatestack.json***] {you need to explain those two files }

Procedure

- Create an EC2 instance and integrate it with vs code and aws configure with IAM user

Install

- ✓ ***git***
- ✓ ***aws cli***
- ✓ ***sudo yum install nodejs***
- ✓ ***sudo npm install -g serverless***

In the EC2 create a folder

- ***mkdir serverlesscicd***

  (WE USE SAM in aws and WE USE SERVERELESS FRAMEWORK (3RD Party tool - To provision the services)

- **vi *serverless.yaml* -**

```
service: serverlesscicd

frameworkVersion: '3'

provider:
  name: aws
  runtime: nodejs18.x

# you can overwrite defaults here
  stage: prod
  region: eu-west-1



functions:
  lambda:
    handler: index.handler ( OUR LAMDA CODE – WHAT GOING TO EXECUTE )
    event ( WE ARE GOING TO CREATE REST API )
      - http: ( REST API)
```

```
        path: /submit
        method: POST ( 2 METHODS – GET( Get lambda invoke) AND POST )
```

- create index.js (This is the lambda code)
- **vi Index.js**

```
const AWS = require('aws-sdk');
const dynamodb = new AWS.DynamoDB.DocumentClient();
const { v4: uuidv4 } = require('uuid');

exports.handler = async (event) => {
  try {
    console.log('Raw input data:', event); // Add this line to log the raw
input data

    const formData = {
      name: event.name,
      email: event.email,
      subject: event.subject,
      message: event.message,
    };

    const item = {
      SubmissionId: generateUUID(), // Generate a UUID
      ...formData, // Use the form data as attributes
    };

    // Store the form data in DynamoDB
    await storeFormData(item);

    return {
      statusCode: 200,
      body: JSON.stringify({ message: 'Form submitted successfully and this is
serverless cicd v1' }),
    };
  } catch (error) {
    console.error(error);
    return {
      statusCode: 500,
      body: JSON.stringify({ message: 'Error submitting the form' }),
    };
  }
```

```
};

// here we will provide the dynamo db table name
async function storeFormData(item) {
  const params = {
    TableName: 'ContactFormEntries',
    Item: item,
  };

  await dynamodb.put(params).promise();
}

function generateUUID() {
  return uuidv4();
}
```

- Run ***npm init*** --- > it will create a package.json – We are initiate the node js (we user this file to deploy – we command deploy ,it searches the package.json and deploy it , suppose it is not there it fails)

```
[ec2-user@ip-172-31-22-233 serverlesscicd]$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (serverlesscicd)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to /home/ec2-user/serverlesscicd/package.json:

{
  "name": "serverlesscicd",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}


Is this OK? (yes) yes
[ec2-user@ip-172-31-22-233 serverlesscicd]$ ll
total 4
-rw-r--r--. 1 ec2-user ec2-user 210 Nov 15 18:46 package.json
[ec2-user@ip-172-31-22-233 serverlesscicd]$
```

- As it is a serverless proj we require ***aws dependencies*** , I've added this one line of code by seeing the error in the lambda console , add one more line in the *package.json*

AWS dependencies

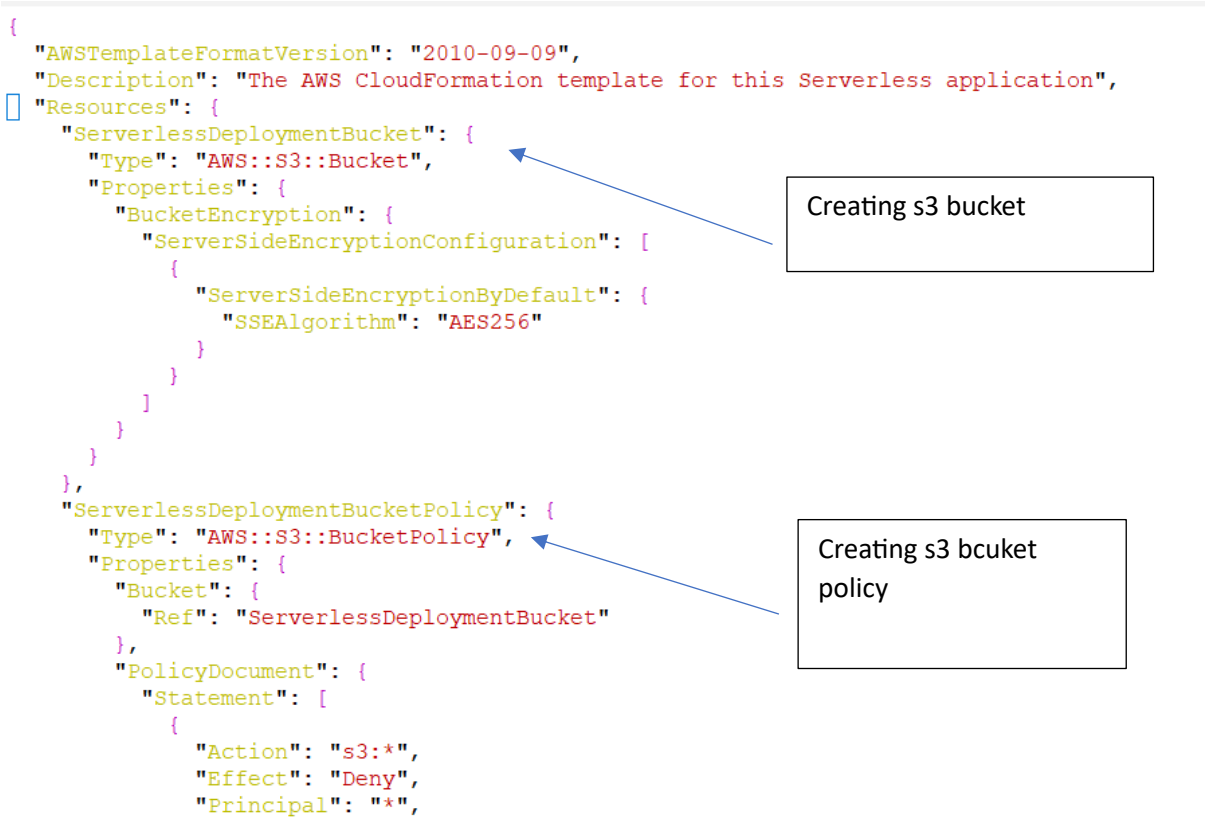
1. use module to use other services
2.

```
{
  "name": "serverlesscicd",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
        "aws-sdk": "^2.1386.0"
    }


}
~
~
~
```

- Here the index.js is the main lambda code to be executed in simple it is the name of the code , if you are changing the code or you want to run another code means you need to change the handler name for example in the ***serverless.yaml*** you need to provide the handler the code name here if the code name is testservice then you need to give as testservice.handler in the serverless.yaml

- **Run --- >** ***npm install*** --- > it will create a node modules folder and package-lock.json will be created (For what it is created? =
- **Run** -- > ***serverless deploy***
- **By default It** will create a lambda , rest api gateway based on the service+stage name+ function name – in the serverlessyaml file and also api gateway stage name plus service name
- If you again run the deploy command with the change in stage and lambda name it willcreate new lambda and api gateway
- The serverless deploy command will package all the code in to the lambda and it will create a .serverless folder within that two cloud formation files will be there , cloudformation-create-stack.json

- cloudformation-update-stack.json,, by default the serverless deploy command will upload our index.js code to s3 by as it creates two cloud formation files one to create s3 bucket and value of the output , that bucket will be imported in another update stack , and on this bucket only our index.js code and all our code will be uploaded , on the template you can find the s3 key , in that location the code will be there



```
                "aws:SecureTransport": false
              }
            }
          }
        ]
      }
    }
  },
  "Outputs": {
    "ServerlessDeploymentBucketName": {
      "Value": {
        "Ref": "ServerlessDeploymentBucket"
      }
    }
  }
}
```


Outputs the value of s3 bucket name


```
  },
  "LambdaLambdaFunction": {
    "Type": "AWS::Lambda::Function",
    "Properties": {
      "Code": {
        "S3Bucket": {
          "Ref": "ServerlessDeploymentBucket"
        },
        "S3Key": "serverless/serverlesscicd/prod/1700075330886-2023-11-15T19:08:50.886Z/serverlesscicd.zip"
      },
      "Handler": "index.handler",
      "Runtime": "nodejs18.x",
      "FunctionName": "serverlesscicd-prod-lambda",
      "MemorySize": 1024,
      "Timeout": 6,
      "Role": {
        "Fn::GetAtt": [
          "IamRoleLambdaExecution",
          "Arn"
        ]
      }
    },
    "DependsOn": [
      "LambdaLogGroup"
    ]
  },
  "LambdaLambdaVersionbOC9JI4KxCd0SUf1H5Co8IpYMruZ3SPjLb4WvvAE": {
    "Type": "AWS::Lambda::Version",
    "DeletionPolicy": "Retain",
    "Properties": {
      "FunctionName": {
        "Ref": "LambdaLambdaFunction"
      },
      "CodeSha256": "1KAz9hbXSvm7X3YJZLCxfyoBQiZ4GS3ALr8CdrbQUZ0="
    }
  },
  "ApiGatewayRestApi": {
    "Type": "AWS::ApiGateway::RestApi",
    "Properties": {
      "Name": "prod-serverlesscicd",
      "EndpointConfiguration": {
        "Types": [
          "EDGE"
        ]
```


Our code has been zipped here

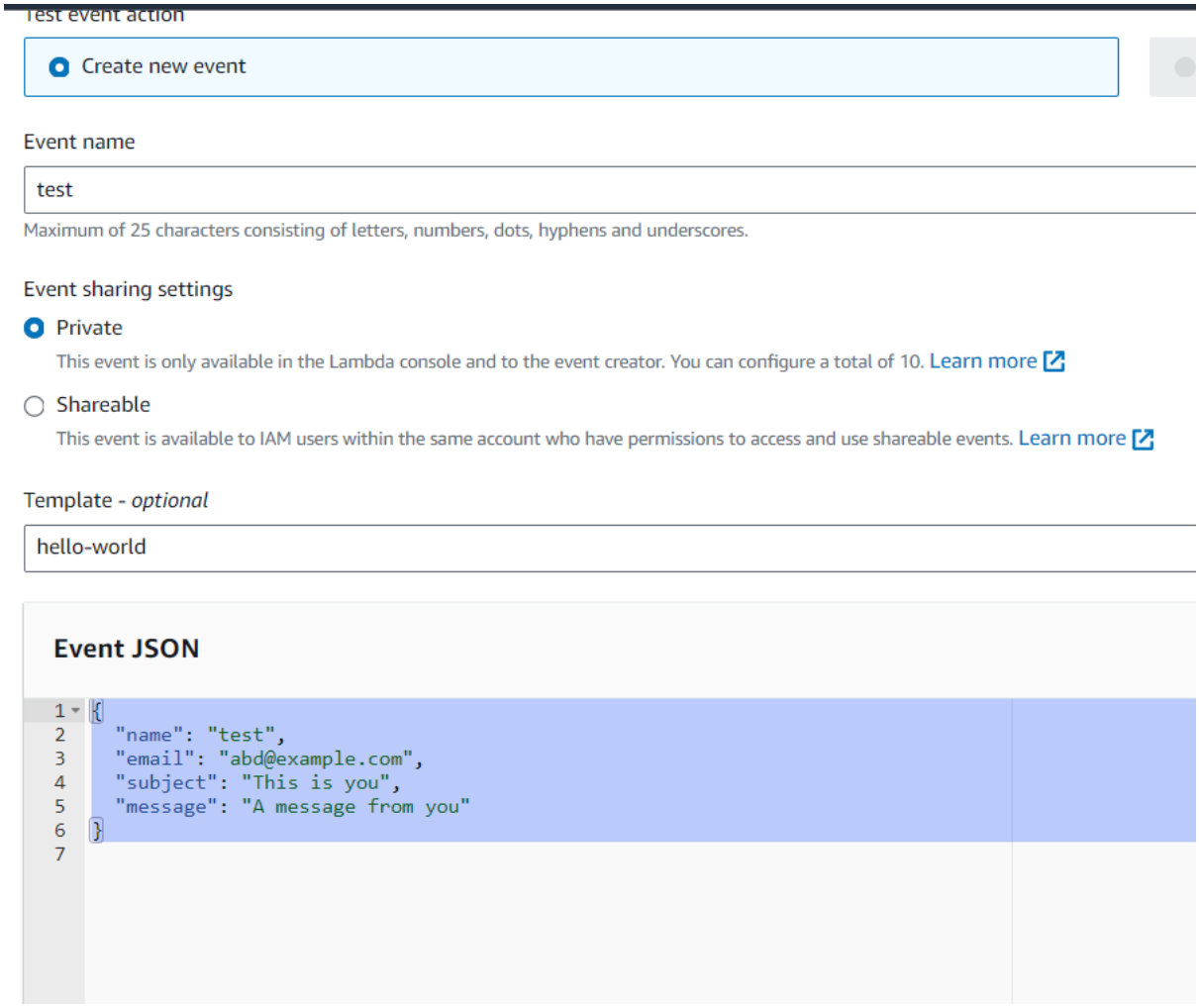
In ***serverless.yaml***

```
13
14
15  functions:
16    cicd:
17      handler: index.handler
18      events:
19        - http:
20            path: /submit
21            method: POST
22
```

- Even though we are giving http it will create rest api only , that is the latest update if you give rest instead of http , still it would rest api gateway but it will be older version of the rest api.
- So know about how to create http

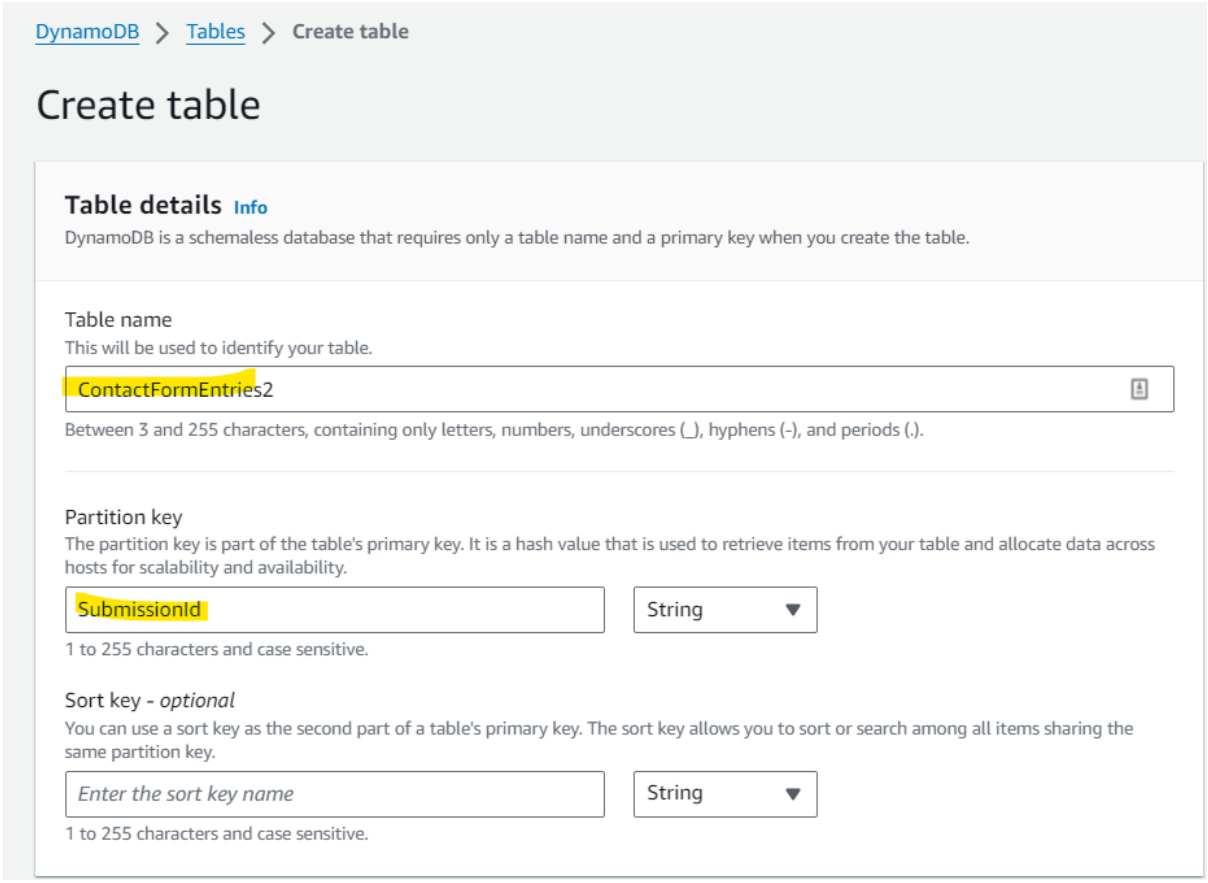
<u>Create events in lambda</u>

```
{
  "name": "test",
  "email": "abd@example.com",
  "subject": "This is you",
  "message": "A message from you"
}
```

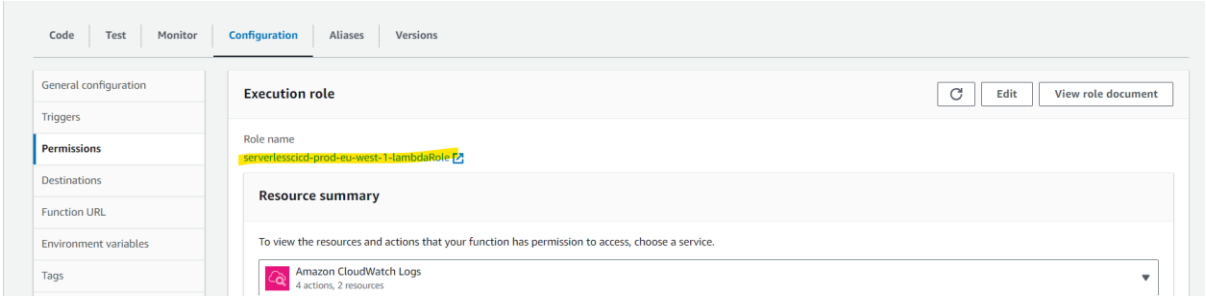


And click on ***save***

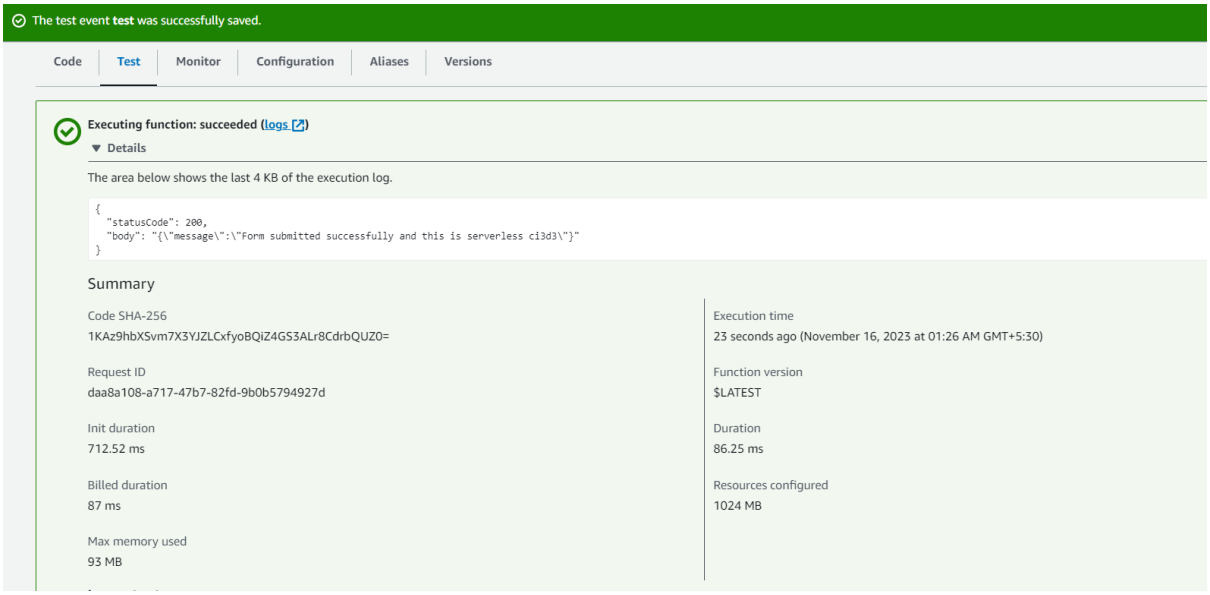
- Goto dynamo db and create a table as given in the same name in index.js



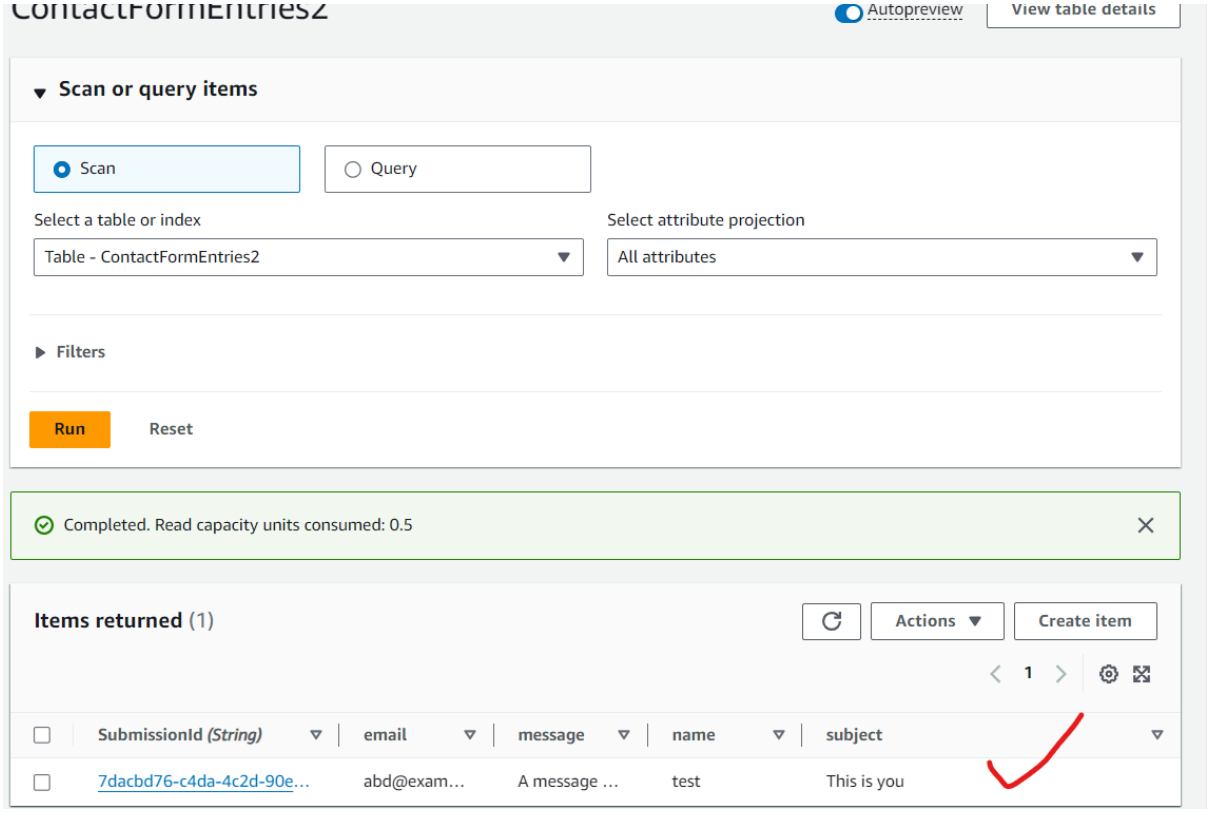
- Edit the existing lambda role to have access to dynamo db , if you didn't attach the role then the permission error will be visible in cloud watch logs
- Under *configuration -- > permission --- > execution role -- > add permission -- > attach policy --- > dynamo db full access/ required access.*



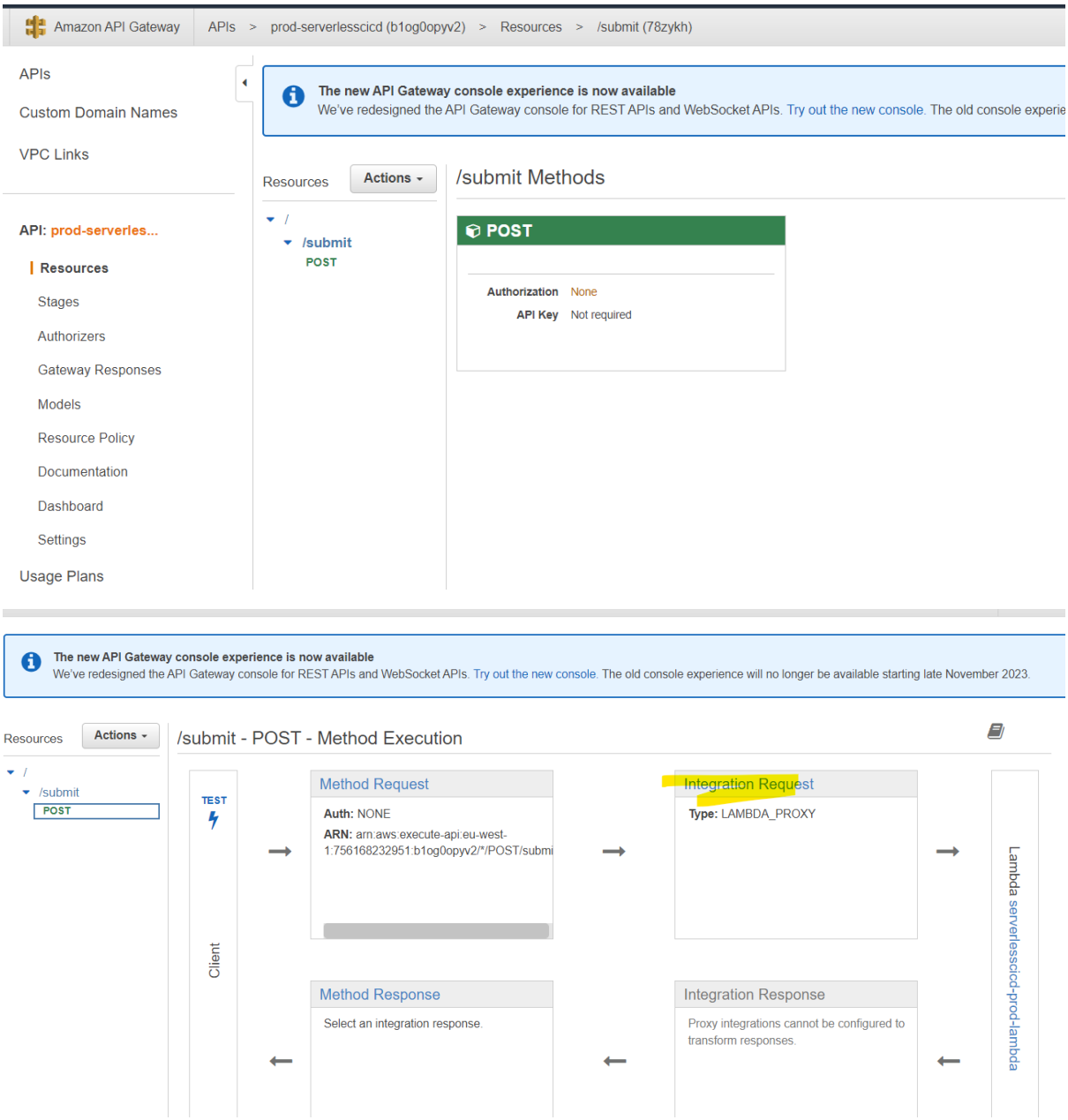
➢ *Now run the test event, you should see*



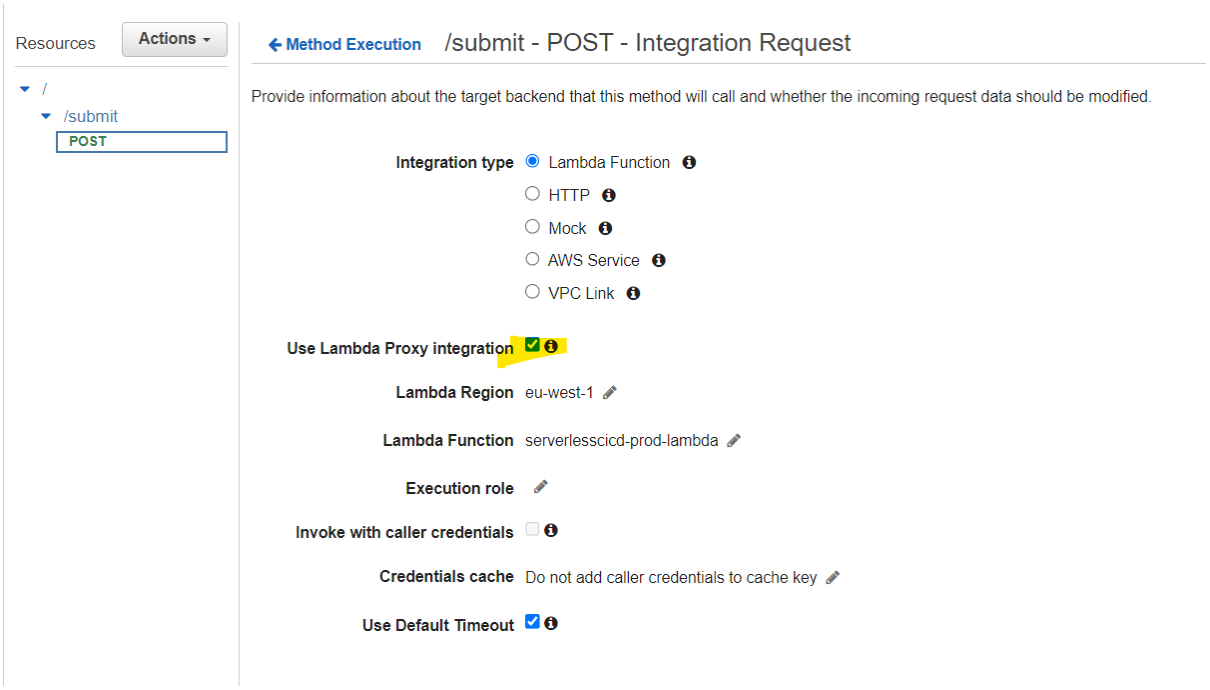
➢ *In the dynamodb you should see the content*



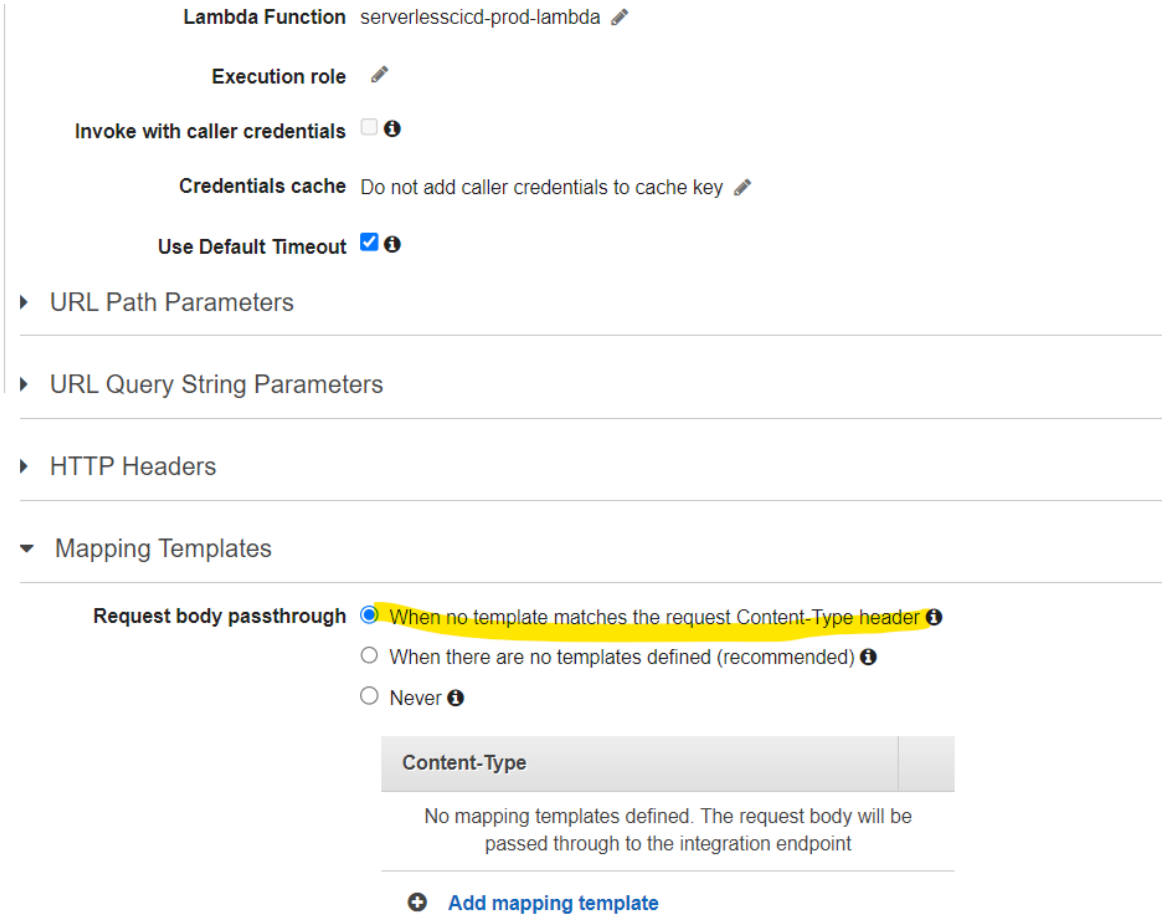
- *Go to api gateway*
- *Click on Post--- > Integration request*



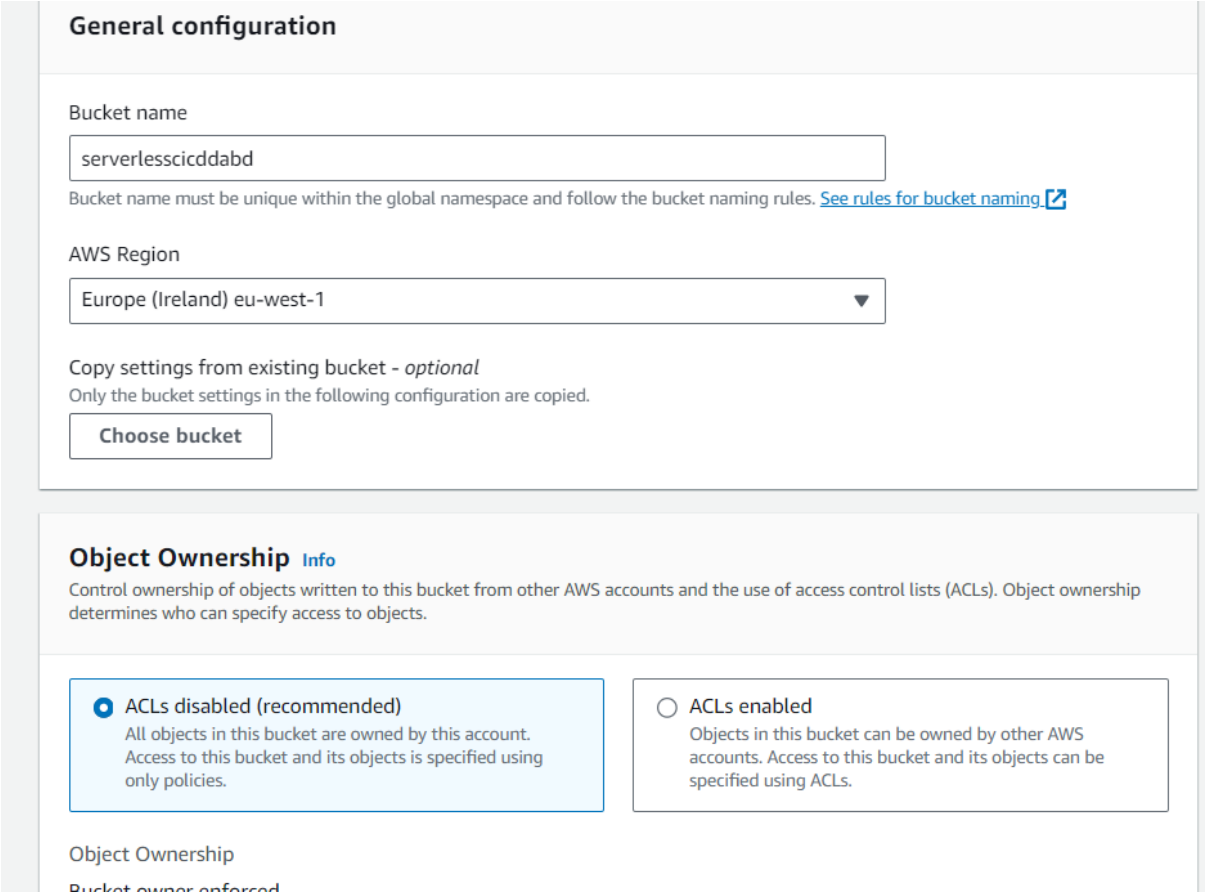
➢ *Uncheck the lambda proxy integration*

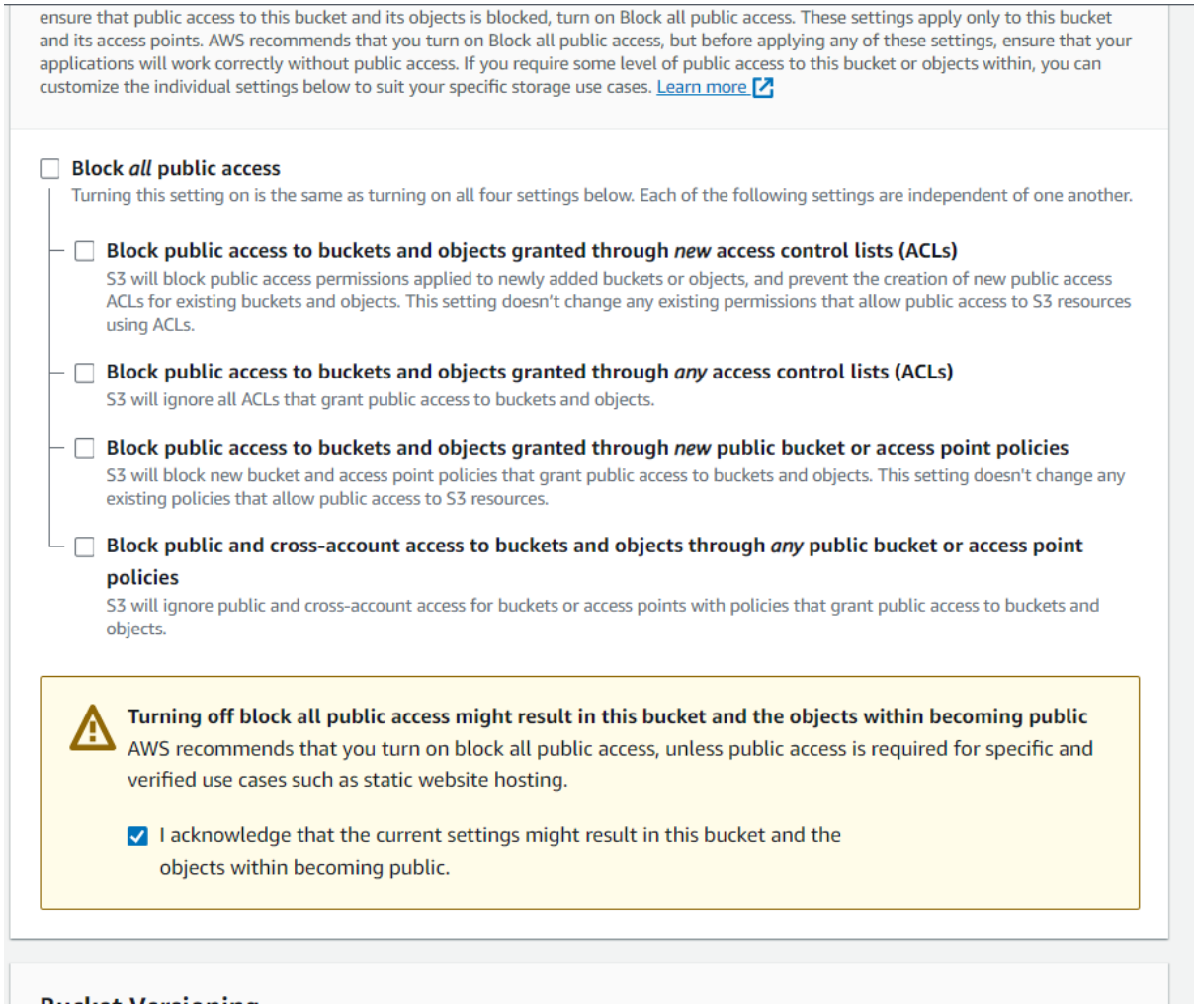


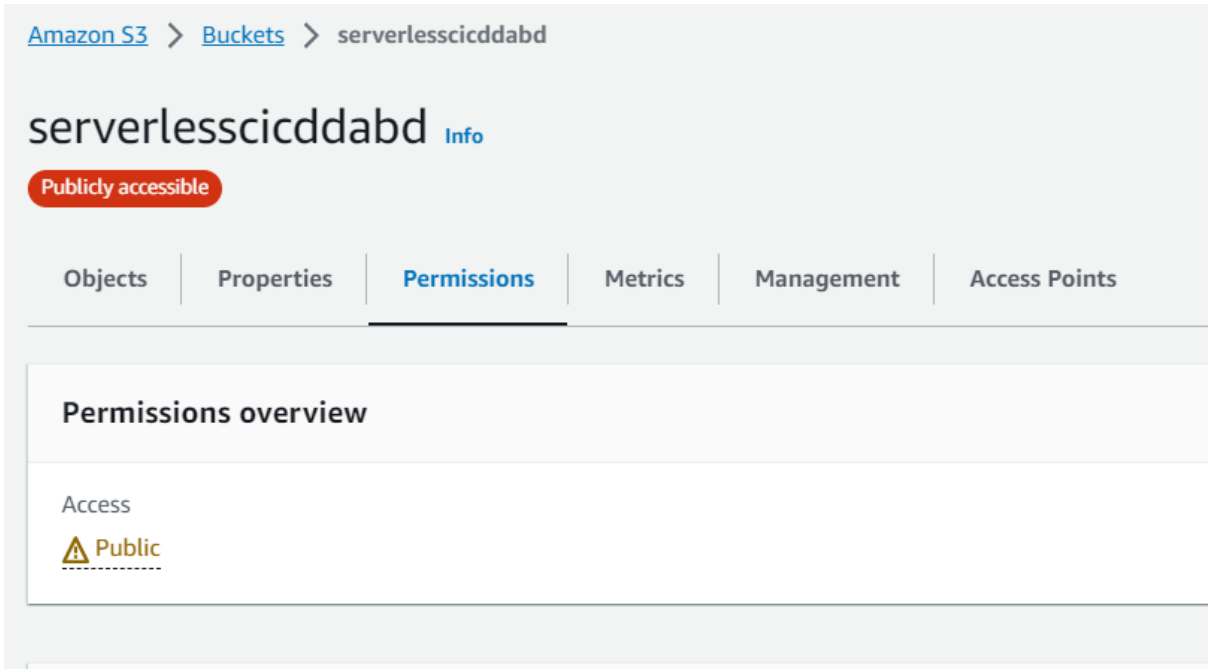
➢ *Scroll down and click on mapping template and verify like configured below*



- Create an *s3 bucket* for website hosting and make the ***bucket public***

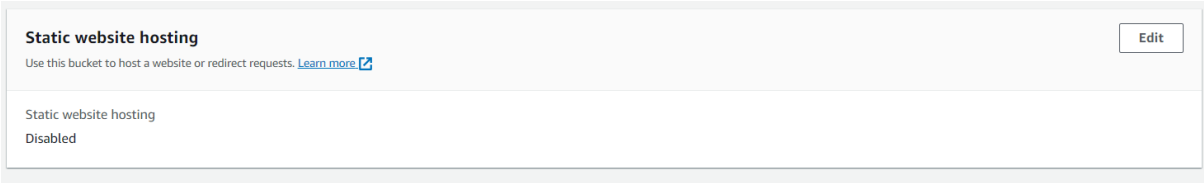




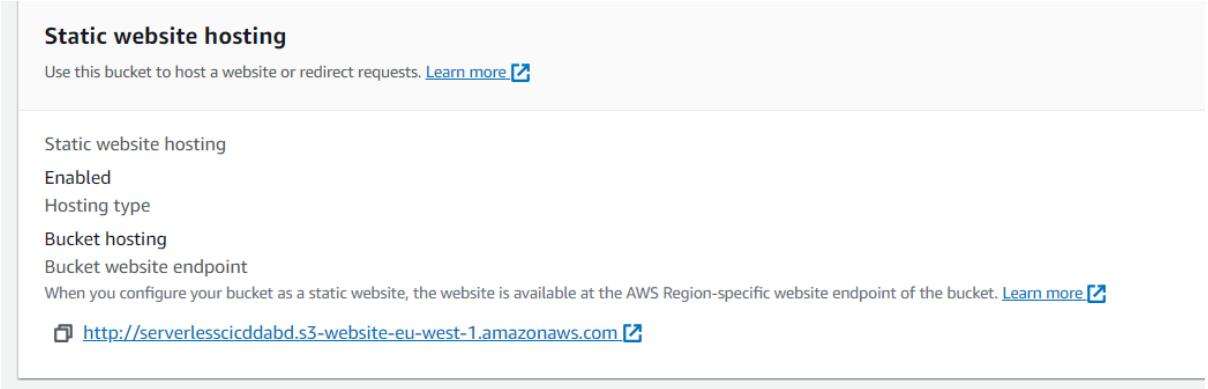


by using policy generator

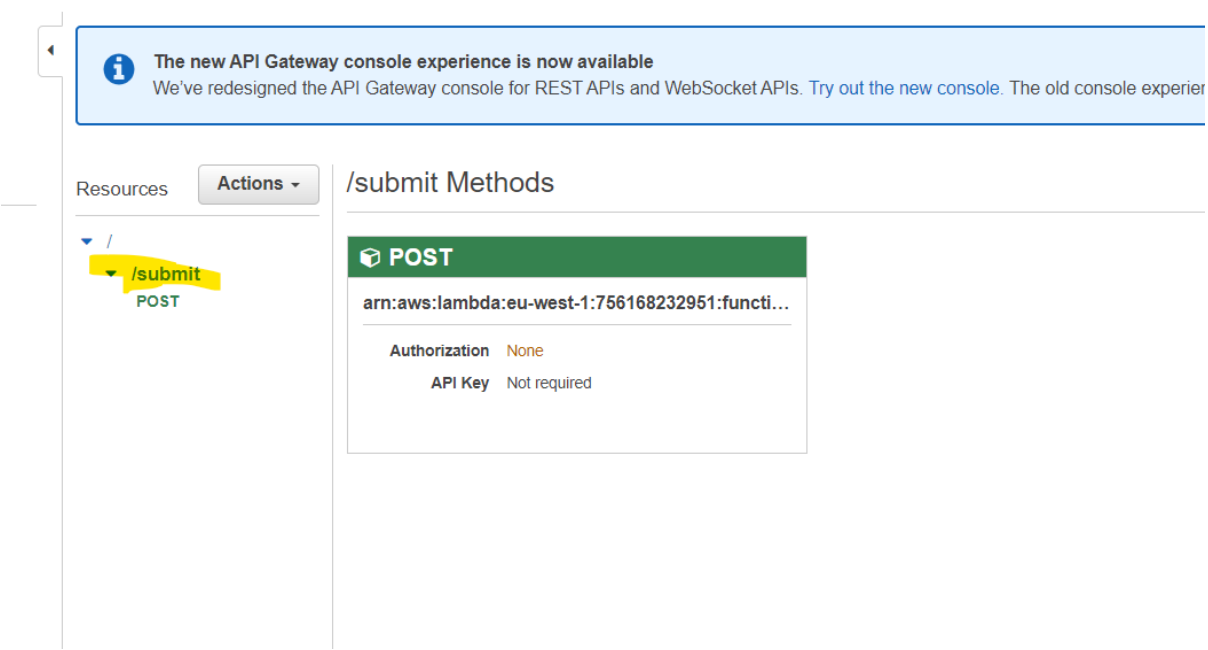
➢ ***Enable website hosting with index.html***



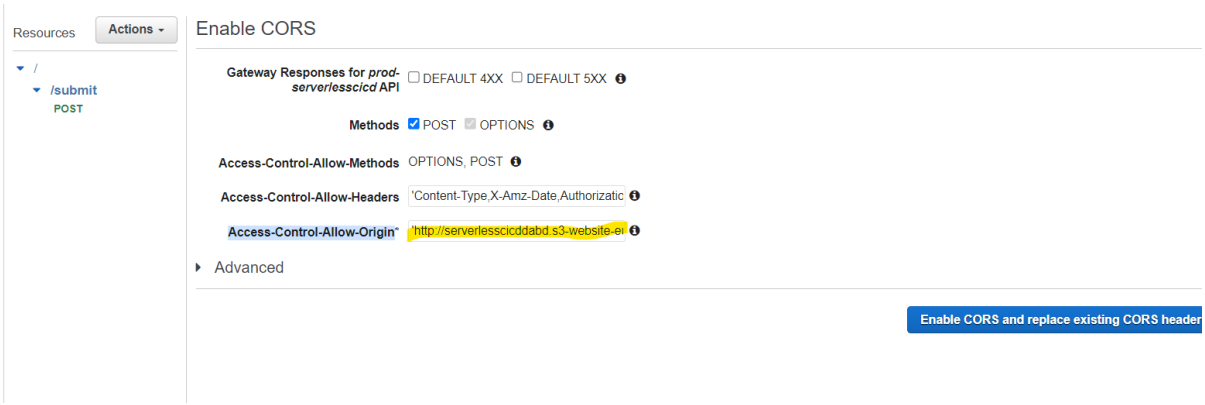
➢ *Copy the s3website link*



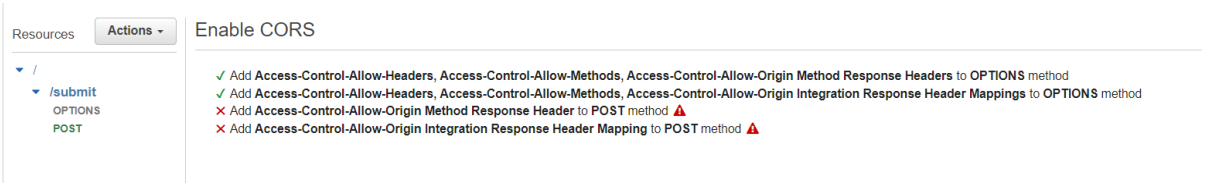
➢ Go to api gateway click on ***submit*** resource



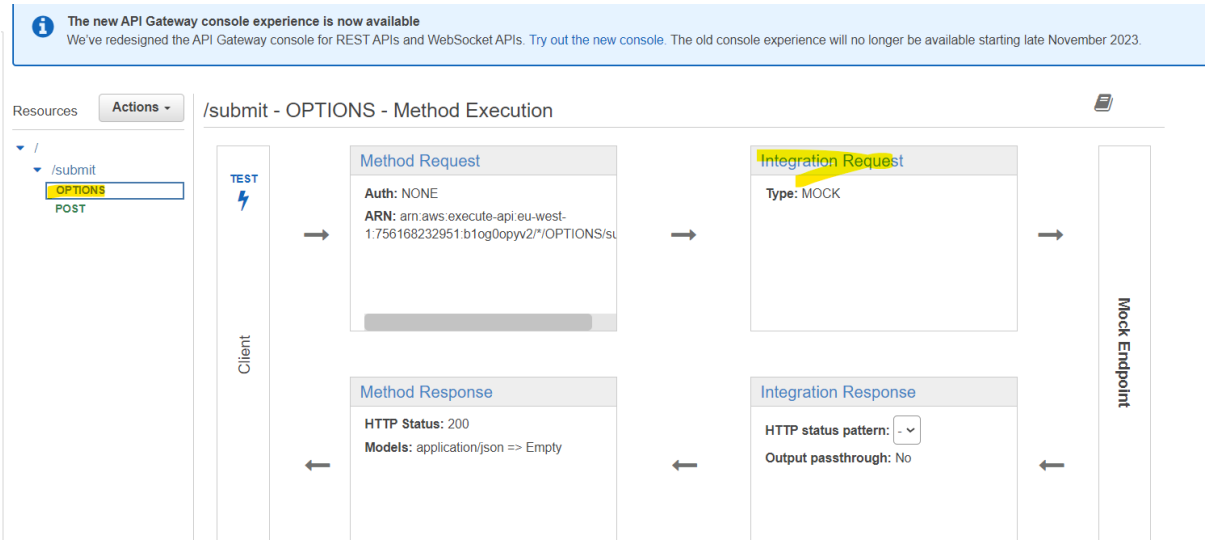
- *Actions -- > enable cors*
- *On that Allow access origin paste the s3 static website url*



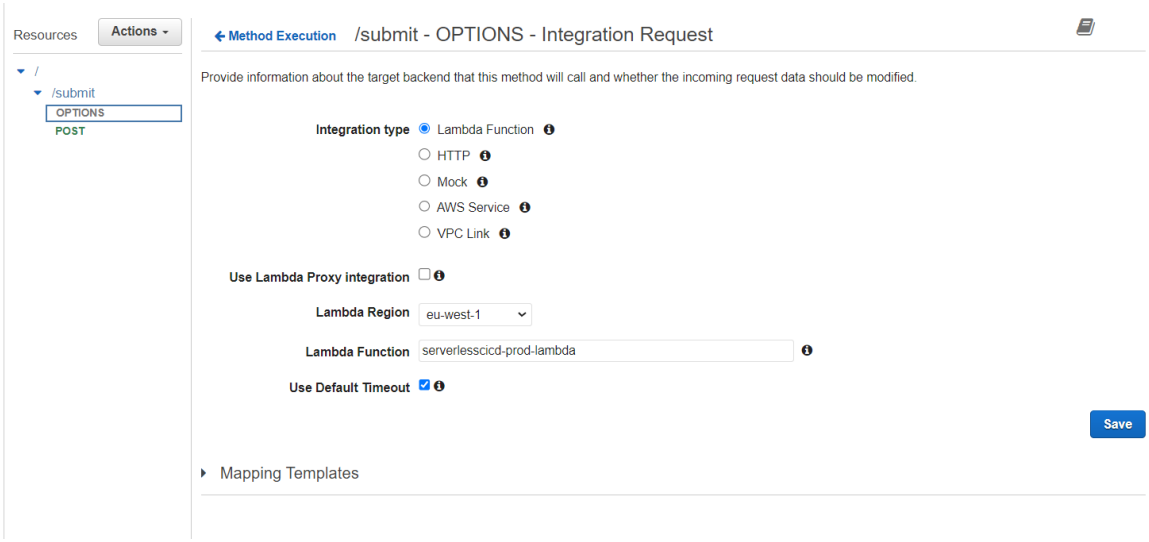
- It would throw a error



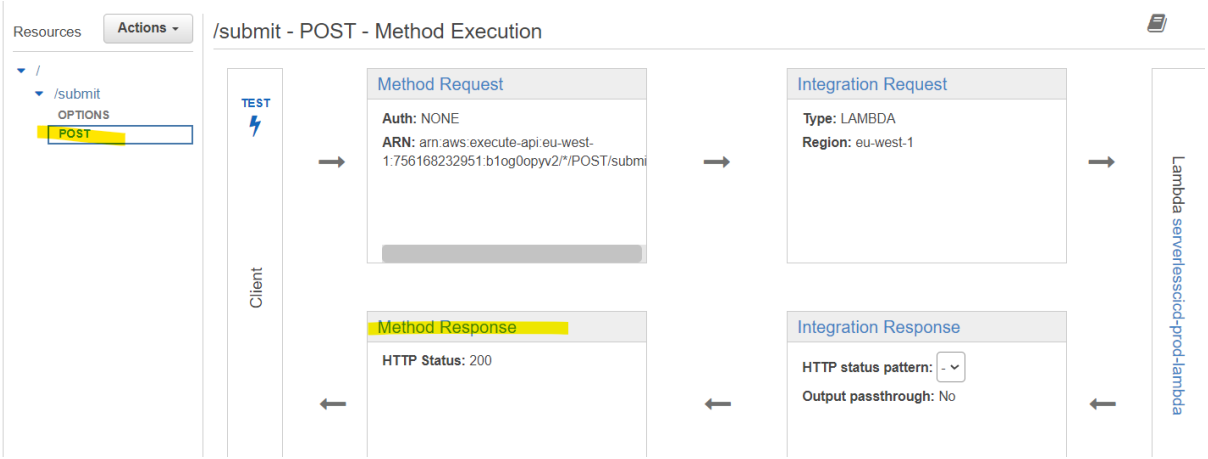
- *Now click on options--- > integration request*



➢ *Configure like below*

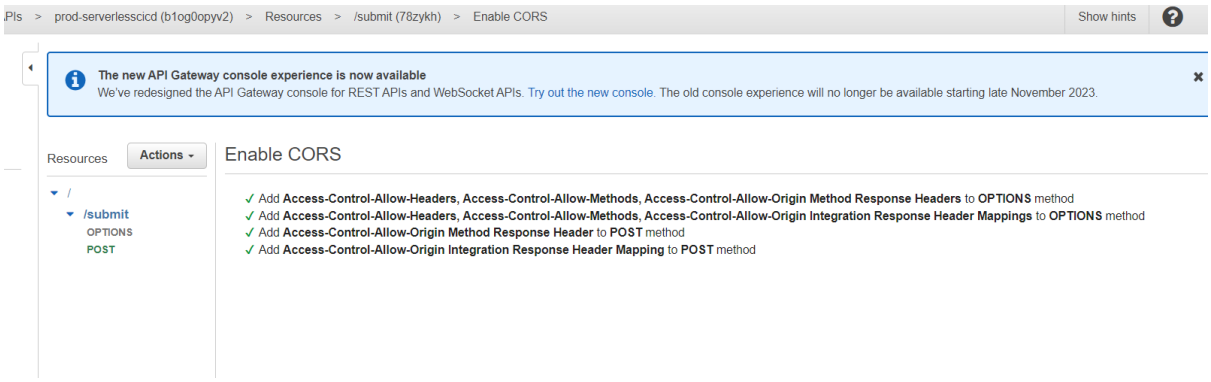


➢ *Click on POST---- > method response*

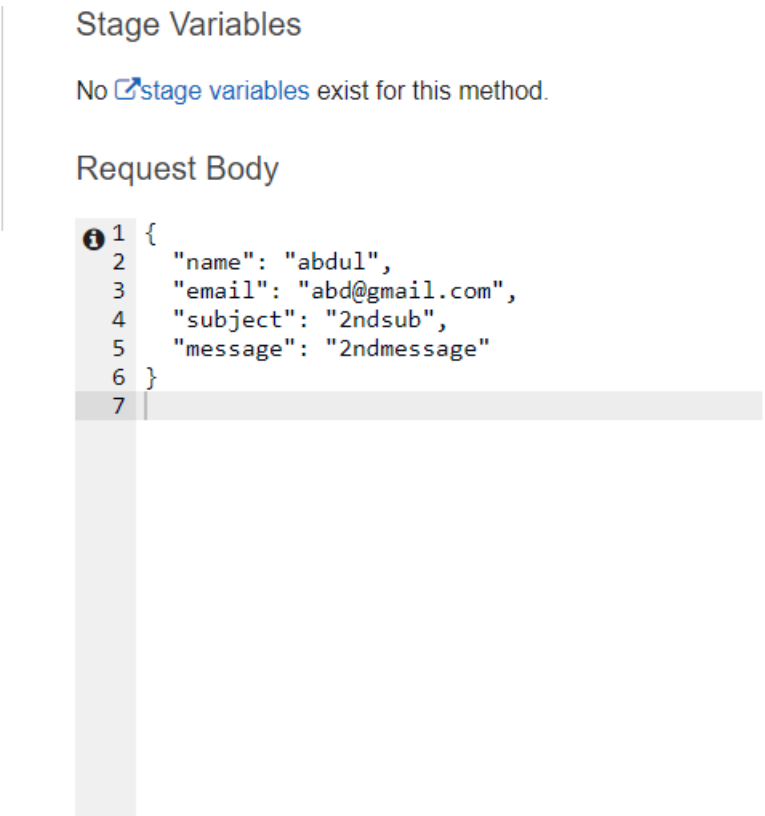


➢ *Type 200*

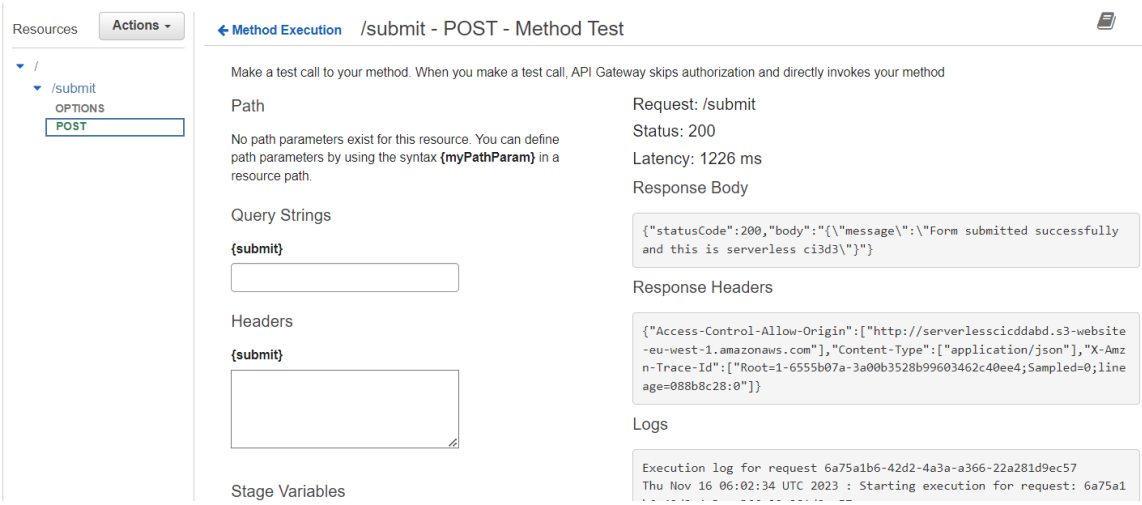
- *Now again enable CORS*



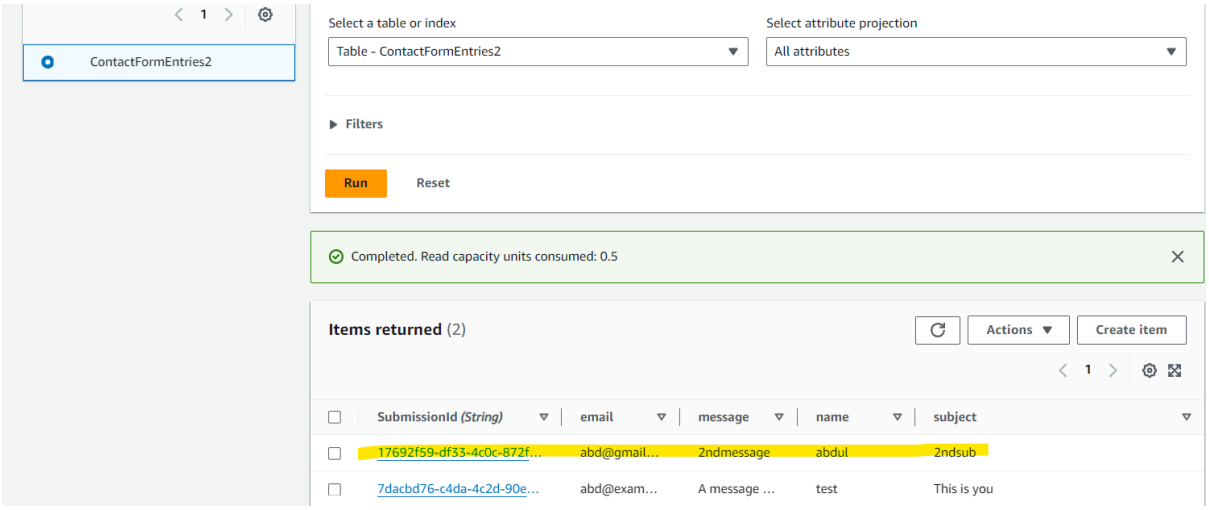
- *Now **Deploy API** under prod stage and copy the api url*
- ***Goto resources -- > test***
- *Under request body*



✓ *You should see the below response*



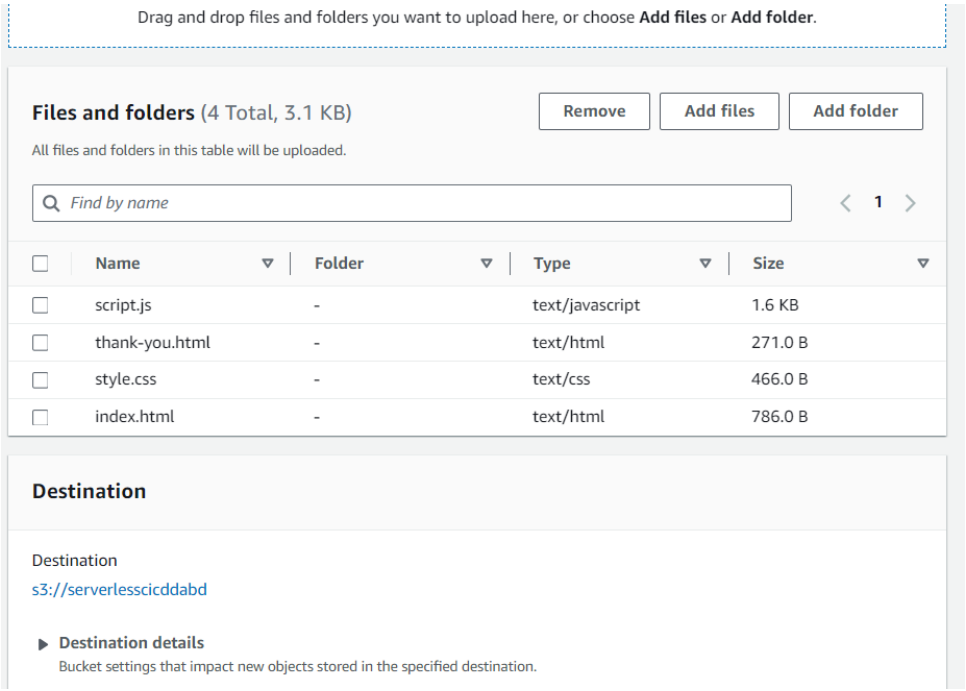
✓ *In dynamo db you should see the below response*



➢ *In the script.js give the api gateway url*

```
});

function submitForm(formData) {
  // Make an API request to the backend (API Gateway) for form submission
  fetch('https://b1og0opyv2.execute-api.eu-west-1.amazonaws.com/prod/submit', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json'
    },

    body: JSON.stringify(formData)
  })
  .then(function(response) {
    if (response.ok) {
      // Redirect to the thank you page
      window.location.href = 'thank-you.html';
    } else {
      throw new Error('Form submission failed.');
    }
  })
  .catch(function(error) {
    console.error(error);
    alert('Form submission failed. Please try again later.');
  });
}
```

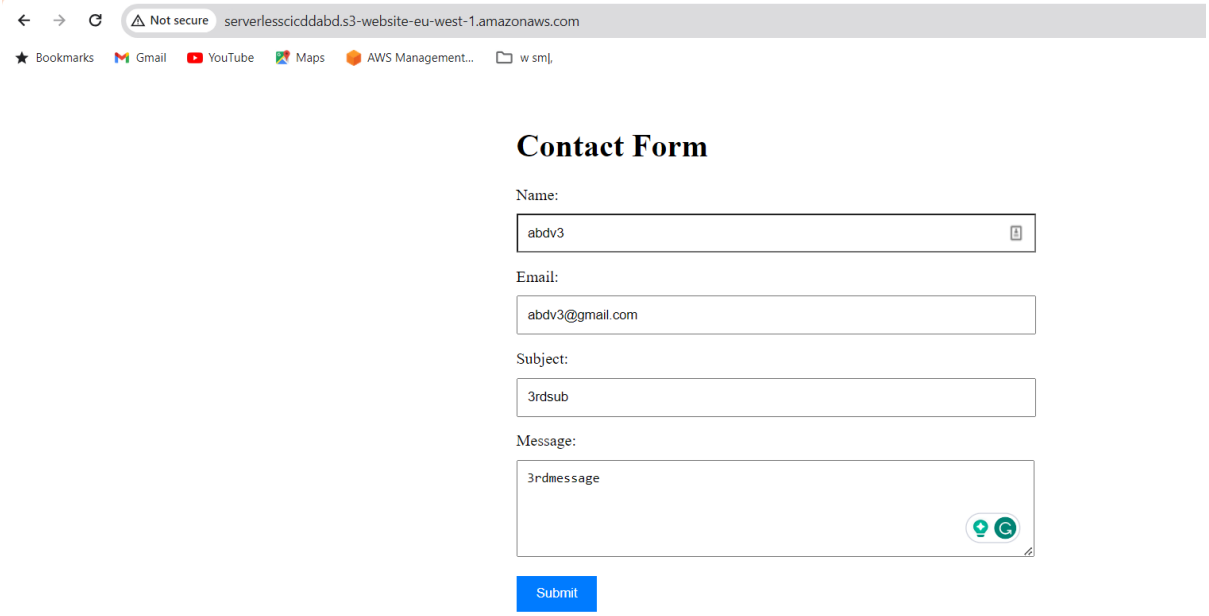
➢ *Go to s3 and upload the below files*

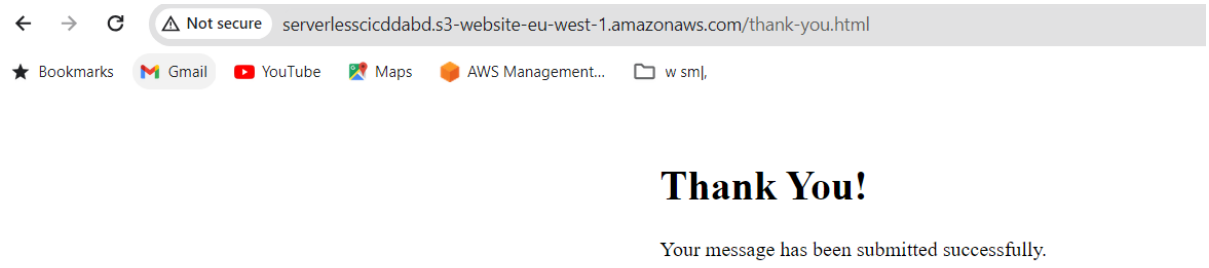


➢ *In s3 paste the following cors under cors configuration*

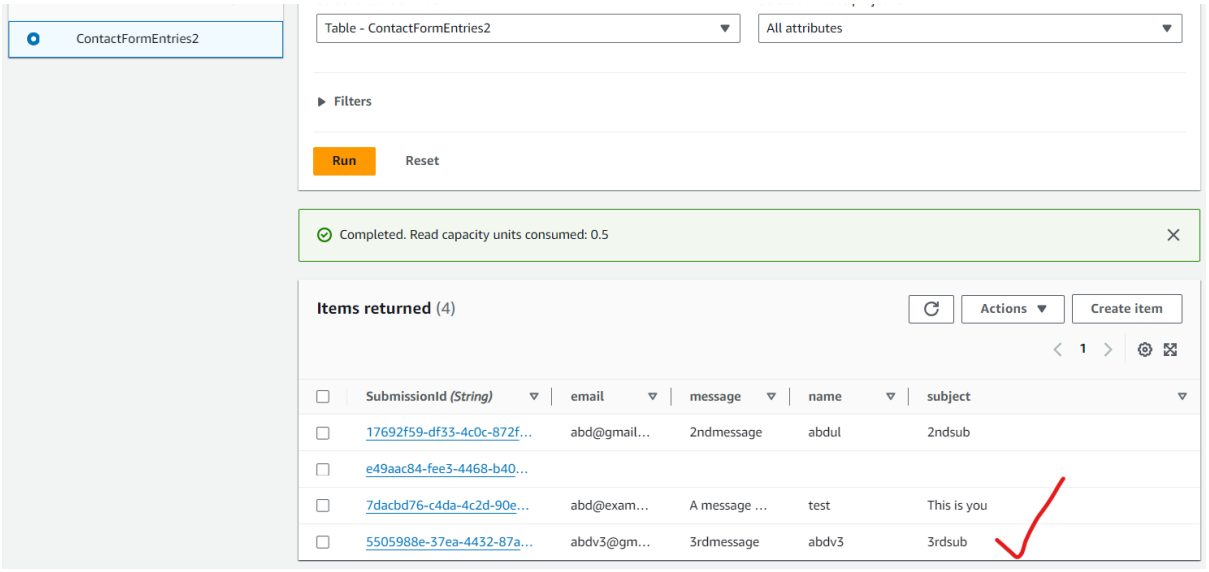
```
[
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "POST"
        ],
        "AllowedOrigins": [
            "your-static-website-url"
        ],
        "ExposeHeaders": []
    }
]
```

- Hit the S3 website url under properties
- *You should see the below success response*



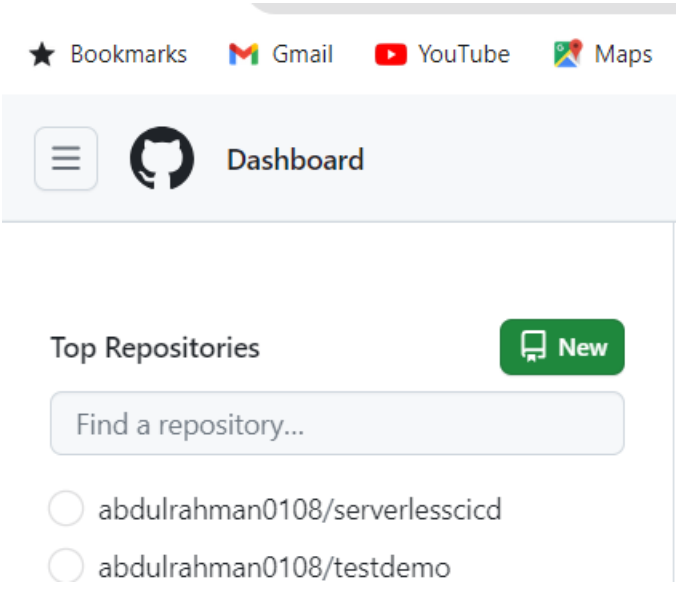


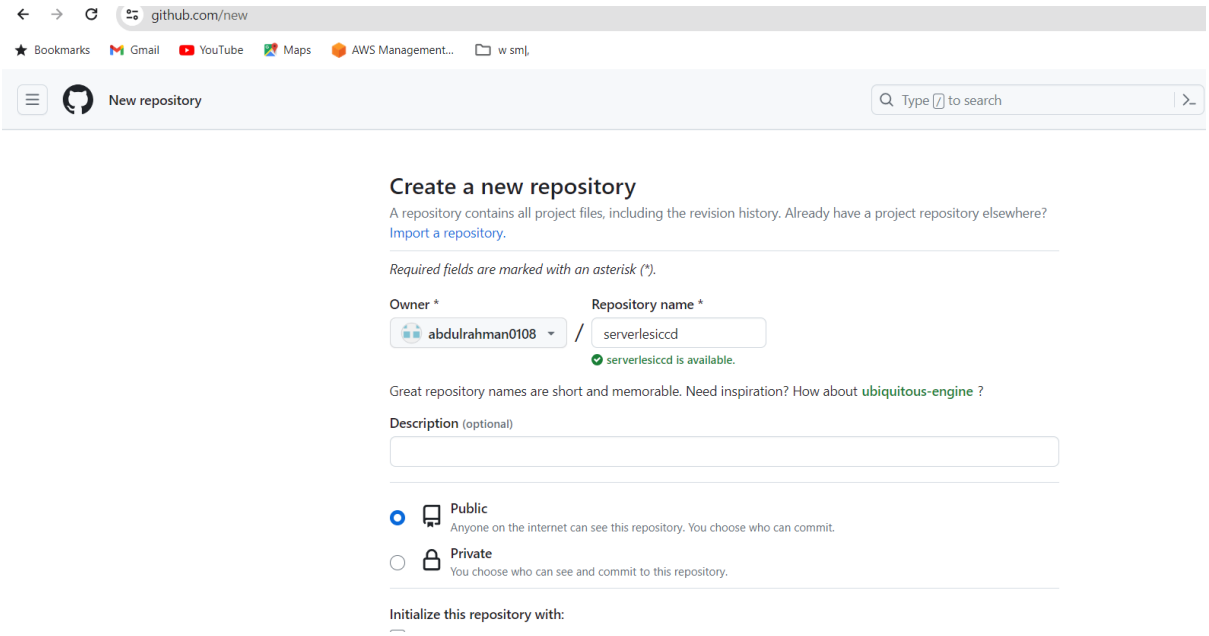
✓ *In dynamo db you should see the below response*



✓ Thus we have integrated *LAMBDA-- > API GATEWAY -- > DYNAMODB -- > S3*

- Now we are about to initiate CI/CD
- Initially we need to integrate our working repo with github
- Create a new repository in ***git hub***





- cd serverelesscidcd/
- ll
- Initiate Git --- > ***git init*** --- in the working repository
- git add*
- git config username and email
- git commit –m “first commit”
- ***git remote add abd*** ***https://github.com/abdulrahman0108/serverlesiccd.git***

- run -- > ***ssh-keygen -t ed25519 -C your_email@example.com***

- ***Press two enter***

- cd
- cd .ssh
- ll
- it displays the all keys
- cat id_pub – copy the key
- Run -- > *eval "$(ssh-agent -s)"*

```
git@github.com: Permission denied (publickey).
[ec2-user@ip-172-31-22-233 serverlesscicd]$ ssh-keygen -t ed25519 -C "abdul.ib96@gmail.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ec2-user/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ec2-user/.ssh/id_ed25519
Your public key has been saved in /home/ec2-user/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:23WPaLDqA7AkGt0DlXNFgWM4MEW1BvM9fz9nD9+kpRo abdul.ib96@gmail.com
The key's randomart image is:
+--[ED25519 256]--+
|  o=*+.++.       |
|  ..=+=o         |
| . o =+.o        |
|. o =.   o       |
| o o +  S o o .  |
|.   . .  o = + o |
|       .. o E = *|
|        .. . . @o|
|       .o.  ..o +|
+----[SHA256]-----+
[ec2-user@ip-172-31-22-233 serverlesscicd]$ eval "$(ssh-agent -s)"
Agent pid 32495
[ec2-user@ip-172-31-22-233 serverlesscicd]$ ssh-add ~/.ssh/id_ed25519
Identity added: /home/ec2-user/.ssh/id_ed25519 (abdul.ib96@gmail.com)
[ec2-user@ip-172-31-22-233 serverlesscicd]$ cd
[ec2-user@ip-172-31-22-233 ~]$ cd .ssh/
[ec2-user@ip-172-31-22-233 .ssh]$ ll
total 16
-rw-------. 1 ec2-user ec2-user 385 Nov 12 19:18 authorized_keys
-rw-------. 1 ec2-user ec2-user 411 Nov 16 10:25 id_ed25519
-rw-r--r--. 1 ec2-user ec2-user 102 Nov 16 10:25 id_ed25519.pub
-rw-r--r--. 1 ec2-user ec2-user  92 Nov 16 10:24 known_hosts
[ec2-user@ip-172-31-22-233 .ssh]$ cat id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIM0SLbtj2qoUS/j1lBUuE2xH/1P2JiNrqo9M3QxwgQuc abdul.ib96@gmail.com
[ec2-user@ip-172-31-22-233 .ssh]$ ^C
[ec2-user@ip-172-31-22-233 .ssh]$ ssh -v git@github.com
OpenSSH_8.7p1, OpenSSL 3.0.8 7 Feb 2023
debug1: Reading configuration data /etc/ssh/ssh_config
```

- **ssh-add ~/.ssh/id_ed25519**
- **Go to Git hub and add the public key in the ssh keys and add it**
- **ssh –v git@github.com**
- **hi 7598273777 – it shows**

```
total 44
-rw-r--r--.  1 ec2-user ec2-user  1170 Nov 15 19:01 index.js
drwxr-xr-x. 37 ec2-user ec2-user 16384 Nov 15 19:00 node_modules
-rw-r--r--.  1 ec2-user ec2-user 14158 Nov 15 19:00 package-lock.json
-rw-r--r--.  1 ec2-user ec2-user   270 Nov 15 18:51 package.json
-rw-r--r--.  1 ec2-user ec2-user   287 Nov 15 19:08 serverless.yaml
[ec2-user@ip-172-31-22-233 serverlesscicd]$ git push -u abd master
Username for 'https://github.com': abdulrahman0108
Password for 'https://abdulrahman0108@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/en/get-started/getting-started-w
tication.
fatal: Authentication failed for 'https://github.com/abdulrahman0108/serve
[ec2-user@ip-172-31-22-233 serverlesscicd]$ ^C
[ec2-user@ip-172-31-22-233 serverlesscicd]$ git push -u abd master
Username for 'https://github.com': abdulrahman0108
Password for 'https://abdulrahman0108@github.com':
Enumerating objects: 2283, done.
Counting objects: 100% (2283/2283), done.
Compressing objects: 100% (2231/2231), done.
Writing objects: 100% (2283/2283), 11.39 MiB | 3.14 MiB/s, done.
Total 2283 (delta 1178), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1178/1178), done.
To https://github.com/abdulrahman0108/serverlesiccd.git
 * [new branch]      master -> master
branch 'master' set up to track 'abd/master'.
[ec2-user@ip-172-31-22-233 serverlesscicd]$
```

This creates a new SSH key, using the provided email as a label.

```
> Generating public/private ALGORITHM key pair.
```

When you're prompted to "Enter a file in which to save the key", you can press **Enter** to accept the default file location. Please note that if you created SSH keys previously, ssh-keygen may ask you to rewrite another key, in which case we recommend creating a custom-named SSH key. To do so, type the default file location and replace id_ALGORITHM with your custom key name.

```
> Enter a file in which to save the key (/home/YOU/.ssh/ALGORITHM):[Press enter]
```

3. At the prompt, type a secure passphrase. For more information, see "Working with SSH key passphrases."

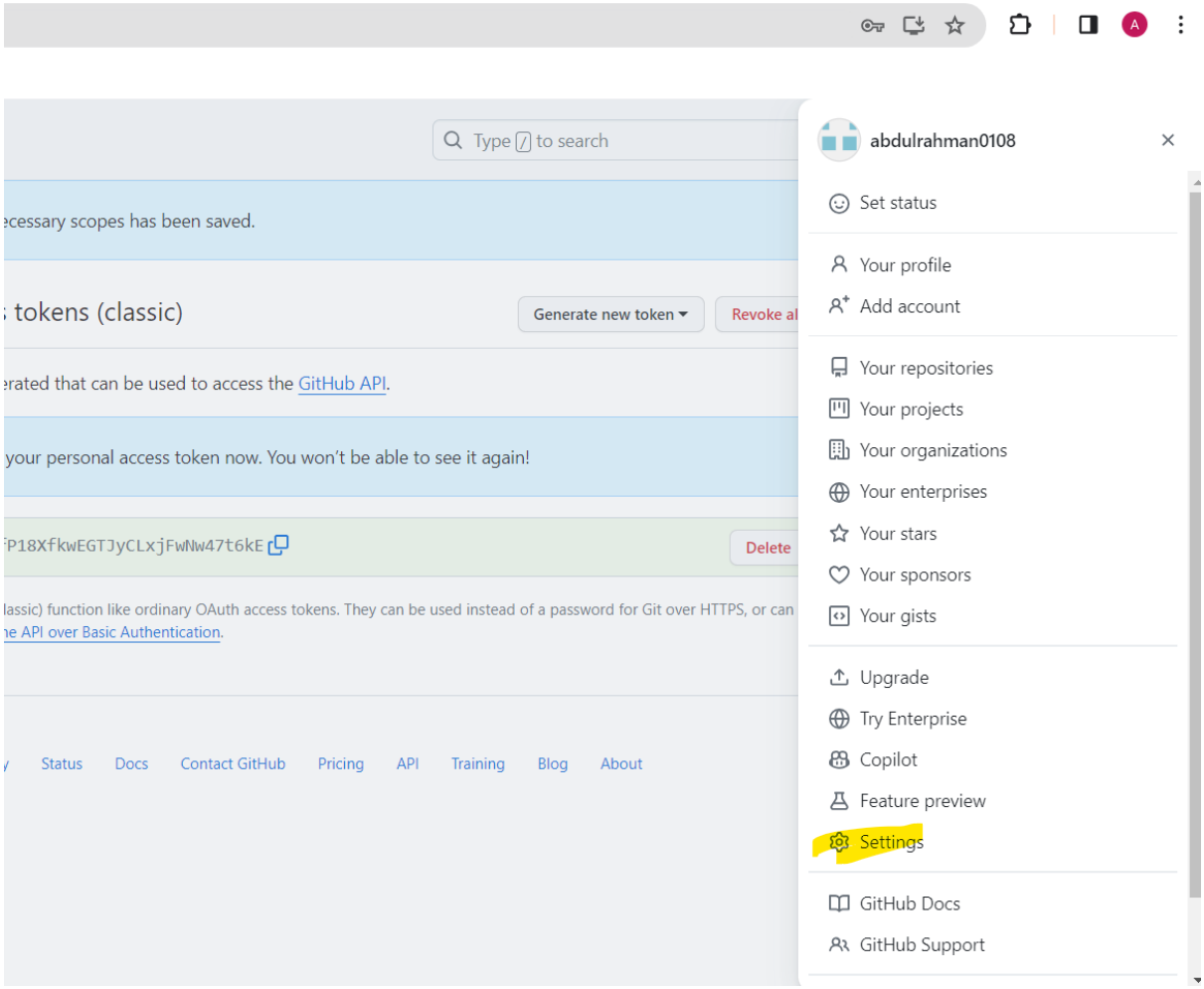
```
> Enter passphrase (empty for no passphrase): [Type a passphrase]
> Enter same passphrase again: [Type passphrase again]
```

- *for more clarity on above github steps follow the url*

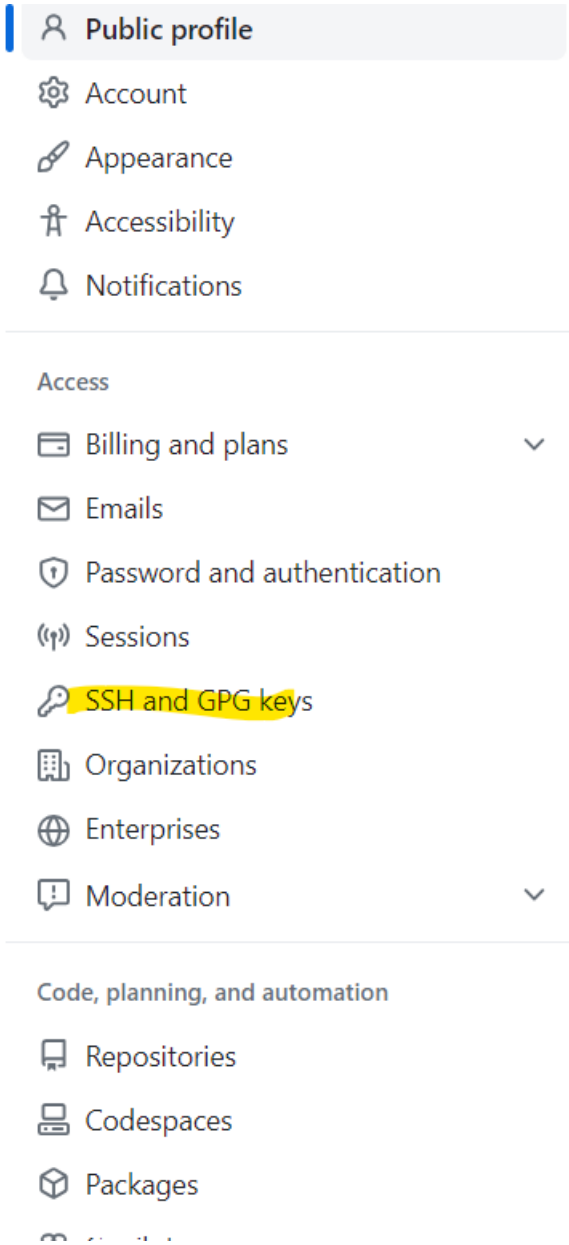
[https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent?platform=linux]

https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account]

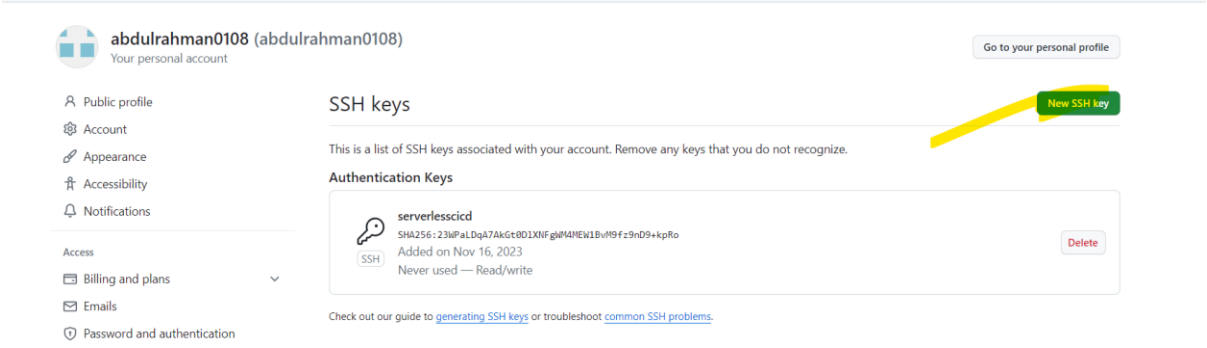
- Now we have created the ssh public key, now we need to add the public key in github
- ***Goto settings in github***



- ***Click on add ssh and gpg keys***



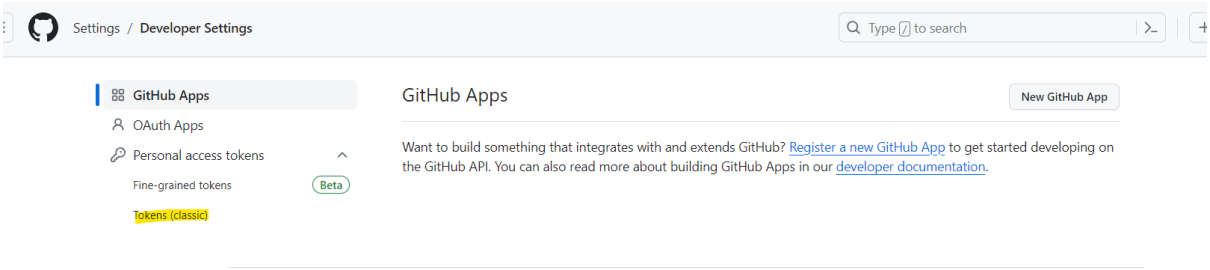
- *Copy the public key from .ssh/*
- *Run -- >* ***cat id_ed25519.pub***
- ***Copy the ssh and paste on the github ssh key***



Now run -- > ***ssh -v git@github.com*** ,, you should see

```
debug1: SSH2_MSG_EXT_INFO received
debug1: kex_input_ext_info: server-sig-algs=<ssh-ed25519-cert-v01@openssh.com,ecdsa-sha2-nistp521-cert-v01@openssh.com,ecdsa-sha2-nistp384-cert
openssh.com,sk-ssh-ed25519-cert-v01@openssh.com,sk-ecdsa-sha2-nistp256-cert-v01@openssh.com,rsa-sha2-512-cert-v01@openssh.com,rsa-sha2-256-cert
sk-ssh-ed25519@openssh.com,sk-ecdsa-sha2-nistp256@openssh.com,ssh-ed25519,ecdsa-sha2-nistp521,ecdsa-sha2-nistp384,ecdsa-sha2-nistp256,rsa-sha2-
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey
debug1: Next authentication method: publickey
debug1: Offering public key: /home/ec2-user/.ssh/id_ed25519 ED25519 SHA256:23WPaLDqA7AkGt0DlXNFgWM4MEW1BvM9fz9nD9+kpRo agent
debug1: Server accepts key: /home/ec2-user/.ssh/id_ed25519 ED25519 SHA256:23WPaLDqA7AkGt0DlXNFgWM4MEW1BvM9fz9nD9+kpRo agent
Authenticated to github.com ([140.82.121.4]:22) using "publickey".
debug1: pkcs11_del_provider: called, provider_id = (null)
debug1: channel 0: new [client-session]
debug1: Entering interactive session.
debug1: pledge: filesystem full
debug1: client_input_global_request: rtype hostkeys-00@openssh.com want_reply 0
debug1: client_input_hostkeys: searching /home/ec2-user/.ssh/known_hosts for github.com / (none)
debug1: client_input_hostkeys: searching /home/ec2-user/.ssh/known_hosts2 for github.com / (none)
debug1: client_input_hostkeys: hostkeys file /home/ec2-user/.ssh/known_hosts2 does not exist
Learned new hostkey: RSA SHA256:uNiVztksCsDhcc0u9e8BujQXVUpKZIDTMczCvj3tD2s
Learned new hostkey: ECDSA SHA256:p2QAMXNIC1TJYWeIOttrVc98/R1BUFWu3/LiyKgUfQM
Adding new key for github.com to /home/ec2-user/.ssh/known_hosts: ssh-rsa SHA256:uNiVztksCsDhcc0u9e8BujQXVUpKZIDTMczCvj3tD2s
Adding new key for github.com to /home/ec2-user/.ssh/known_hosts: ecdsa-sha2-nistp256 SHA256:p2QAMXNIC1TJYWeIOttrVc98/R1BUFWu3/LiyKgUfQM
debug1: update_known_hosts: known hosts file /home/ec2-user/.ssh/known_hosts2 does not exist
PTY allocation request failed on channel 0
Hi abdulrahman0108! You've successfully authenticated, but GitHub does not provide shell access.
debug1: client_input_channel_req: channel 0 rtype exit-status reply 0
debug1: channel 0: free: client-session, nchannels 1
Connection to github.com closed.
Transferred: sent 2928, received 3228 bytes, in 0.2 seconds
Bytes per second: sent 12356.2, received 13622.2
debug1: Exit status 1
[ec2-user@ip-172-31-22-233 .ssh]$ ll
total 20
```

- Again go to github clikc on ***settings -- > developer settings-- > generate classic tokens***



- Give all the access and generate token and set expiration as **never expire** and copy the token id

- Run -- > ***git push -u jegan master***
- username : jegan
- Password : Token of git

Given the commands to push to github for reference

git init

35 git add *

36 git config --global user.name "abdulrahman0108"

37 git config --global user.mail "abdul.ib96@gmail.com"

38 git commit -m "1st commit"

39 ll

40 git remote add abd https://github.com/abdulrahman0108/serverlesiccd.git

44 ssh -v git@github.com

45 ssh-keygen -t ed25519 -C "abdul.ib96@gmail.com"

46 eval "$(ssh-agent -s)"

47 ssh-add ~/.ssh/id_ed25519

49 cd .ssh/

51 cat id_ed25519.pub

52 ssh -v git@github.com

54 ssh-keygen -E md5 -lf ~/.ssh/id_ed25519.pub

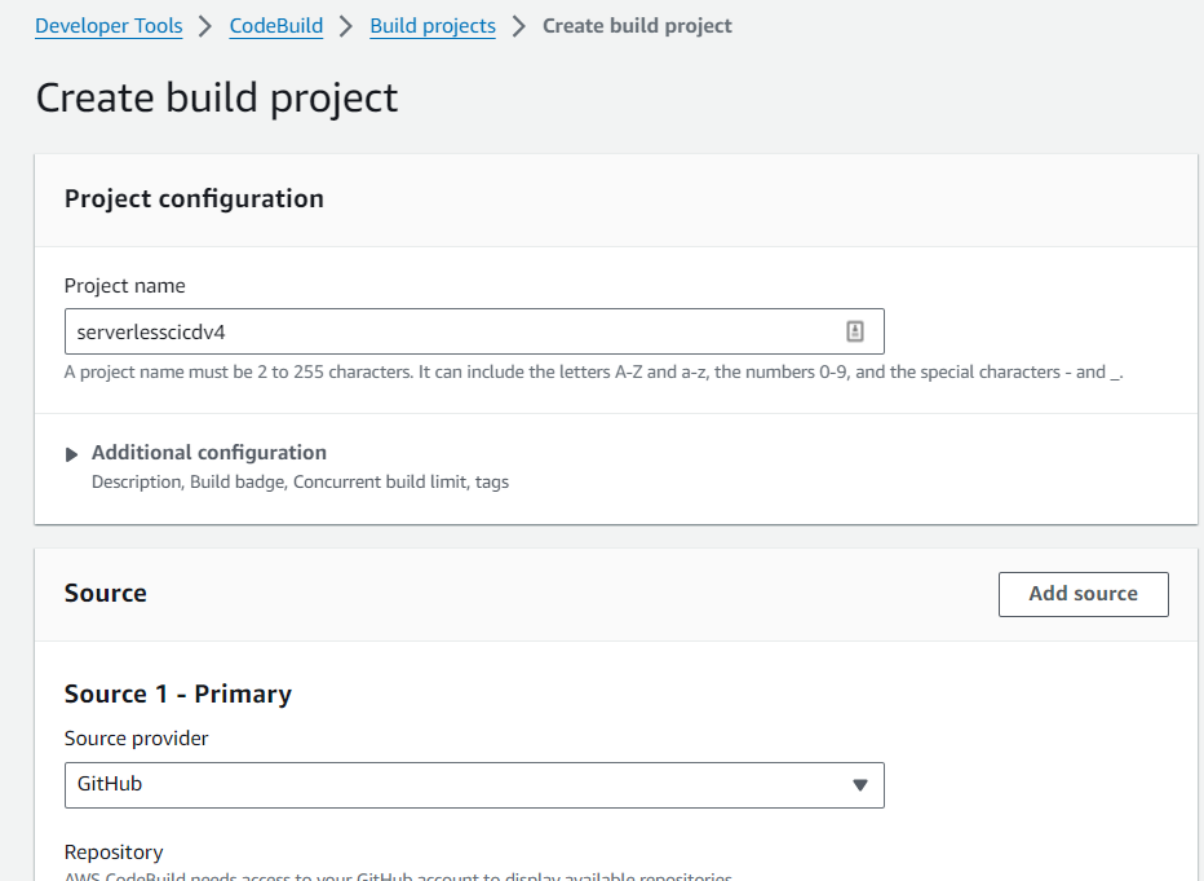
56 cd serverlesscicd/

58 git push -u abd master

✓ *You should able to push from local to github*

```
[ec2-user@ip-172-31-22-233 serverlesscicd]$ git push -u abd master
Username for 'https://github.com': abdulrahman0108
Password for 'https://abdulrahman0108@github.com':
Enumerating objects: 2283, done.
Counting objects: 100% (2283/2283), done.
Compressing objects: 100% (2231/2231), done.
Writing objects: 100% (2283/2283), 11.39 MiB | 3.14 MiB/s, done.
Total 2283 (delta 1178), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1178/1178), done.
To https://github.com/abdulrahman0108/serverlesiccd.git
 * [new branch]      master -> master
branch 'master' set up to track 'abd/master'.
[ec2-user@ip-172-31-22-233 serverlesscicd]$ history
```

- *Now go to **code build***
- ***Build the project***



- ***Save Token***

Source 1 - Primary

Source provider

GitHub

Repository

AWS CodeBuild needs access to your GitHub account to display available repositories.

- Connect using OAuth
- Connect with a GitHub personal access token

GitHub personal access token

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

Save token

Source version - *optional* Info

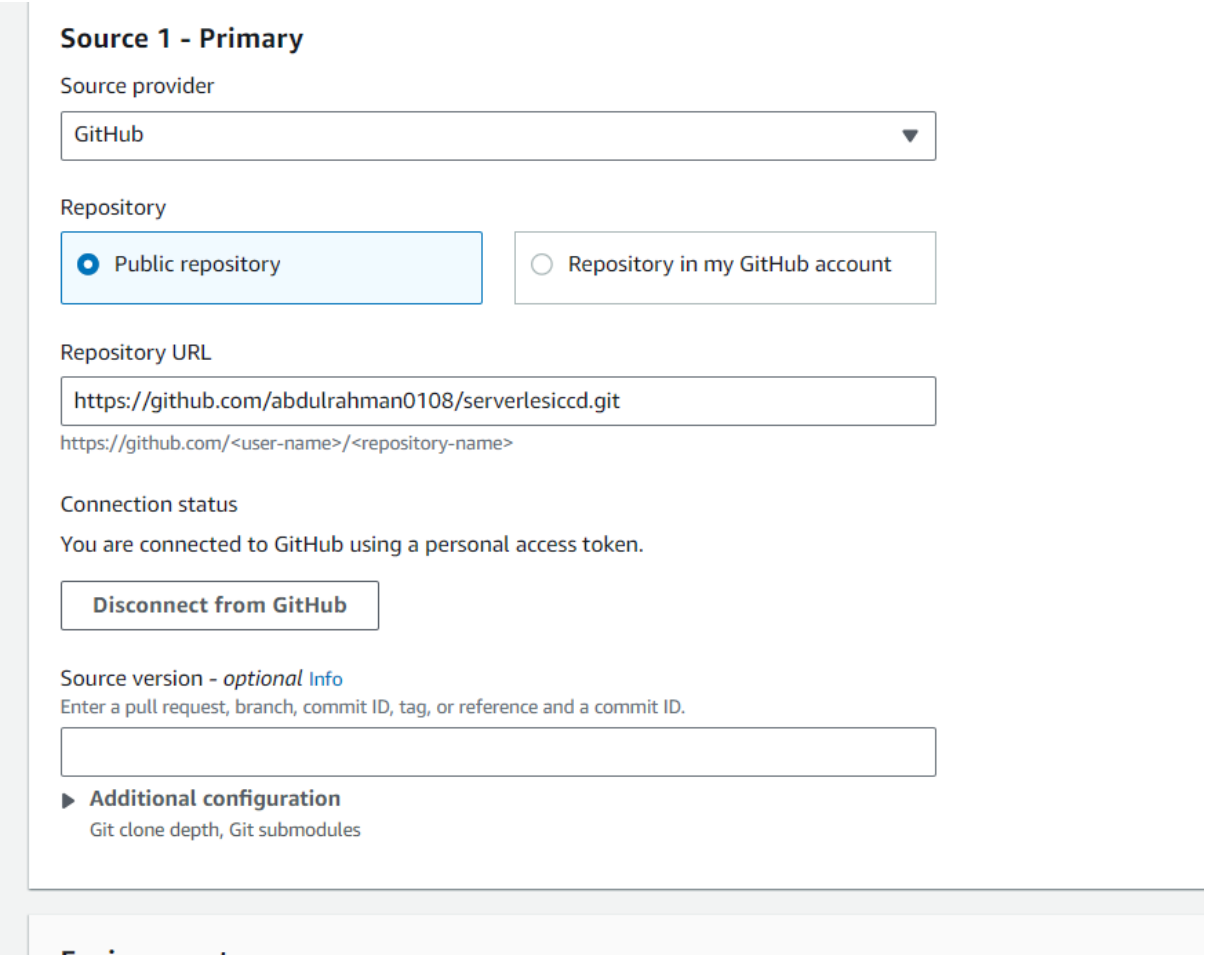
Enter a pull request, branch, commit ID, tag, or reference and a commit ID.

▶ **Additional configuration**

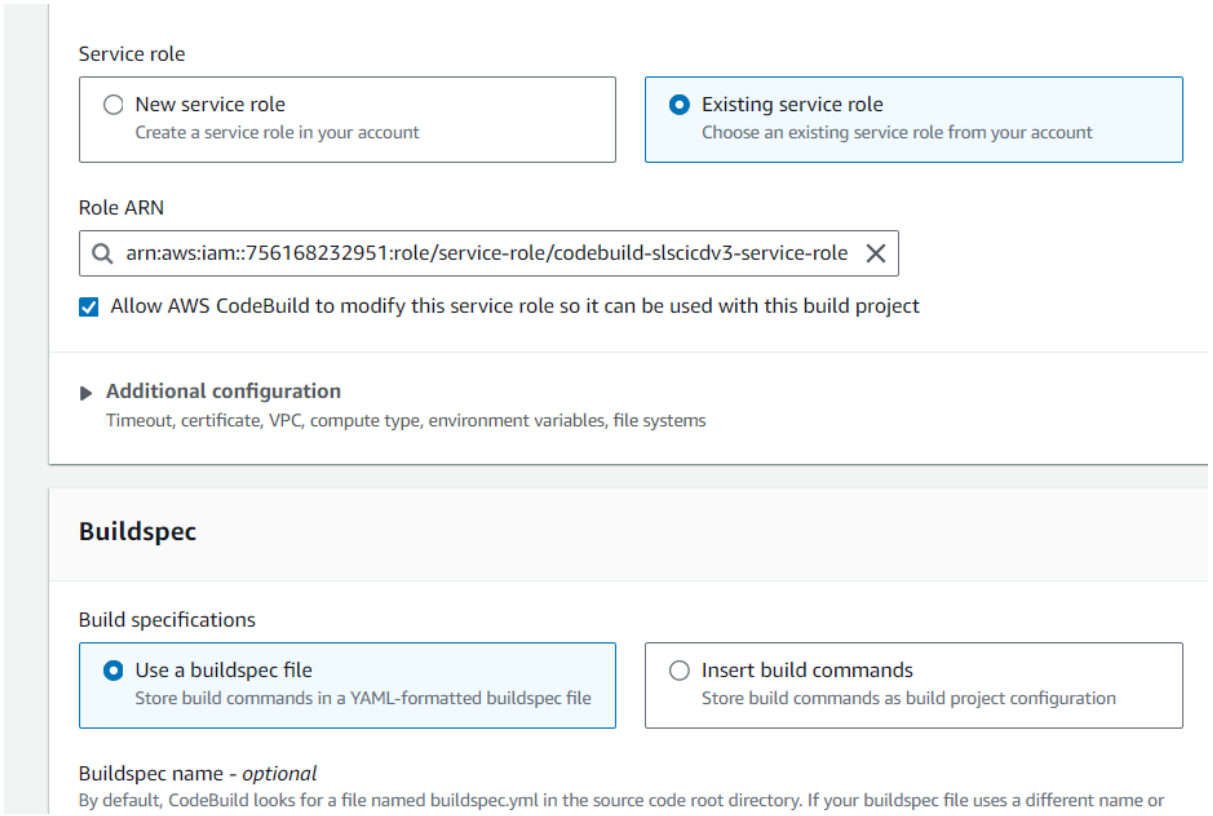
Git clone depth, Git submodules

Environment

- ***Give the repository url***



- ***Attach the existing role or create the role with the following permissions***



Docker images provided by CodeBuild - AWS CodeBuild (amazon.com)

- ***Configure the environment like below***

Environment image

- (•) Managed image
  Use an image managed by AWS CodeBuild
- ( ) Custom image
  Specify a Docker image

Compute

- (•) EC2
  Optimized for flexibility during action runs
- ( ) Lambda
  Optimized for speed and minimizes the start up time of workflow actions

Operating system

Amazon Linux ▼

Runtime(s)

Standard ▼

Image

aws/codebuild/amazonlinux2-x86_64-standard:5.0 ▼

Image version

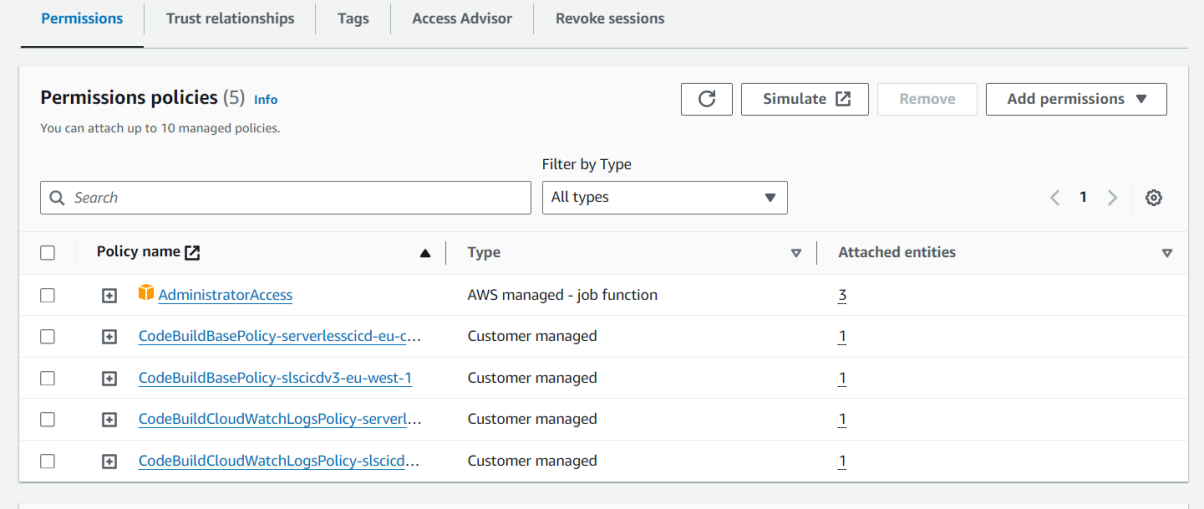
Always use the latest image for this runtime version ▼

- [ ] Use GPU-enhanced compute

Privileged

- [ ] Enable this flag if you want to build Docker images or want your builds to get elevated privileges

- ***Cause the code build to avoid permission issues attach the following permission***



- *Give the **buildspec.yaml** name*

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

Buildspec name - *optional*

By default, CodeBuild looks for a file named buildspec.yml in the source code root directory. If your buildspec file uses a different name or location, enter its path from the source root here (for example, buildspec-two.yml or configuration/buildspec.yml).

buildspec.yaml

Batch configuration

You can run a group of builds as a single execution. Batch configuration is also available in advanced option when starting build.

- [ ] Define batch configuration - *optional*
  You can also define or override batch configuration when starting a build batch.

Artifacts

Add artifact

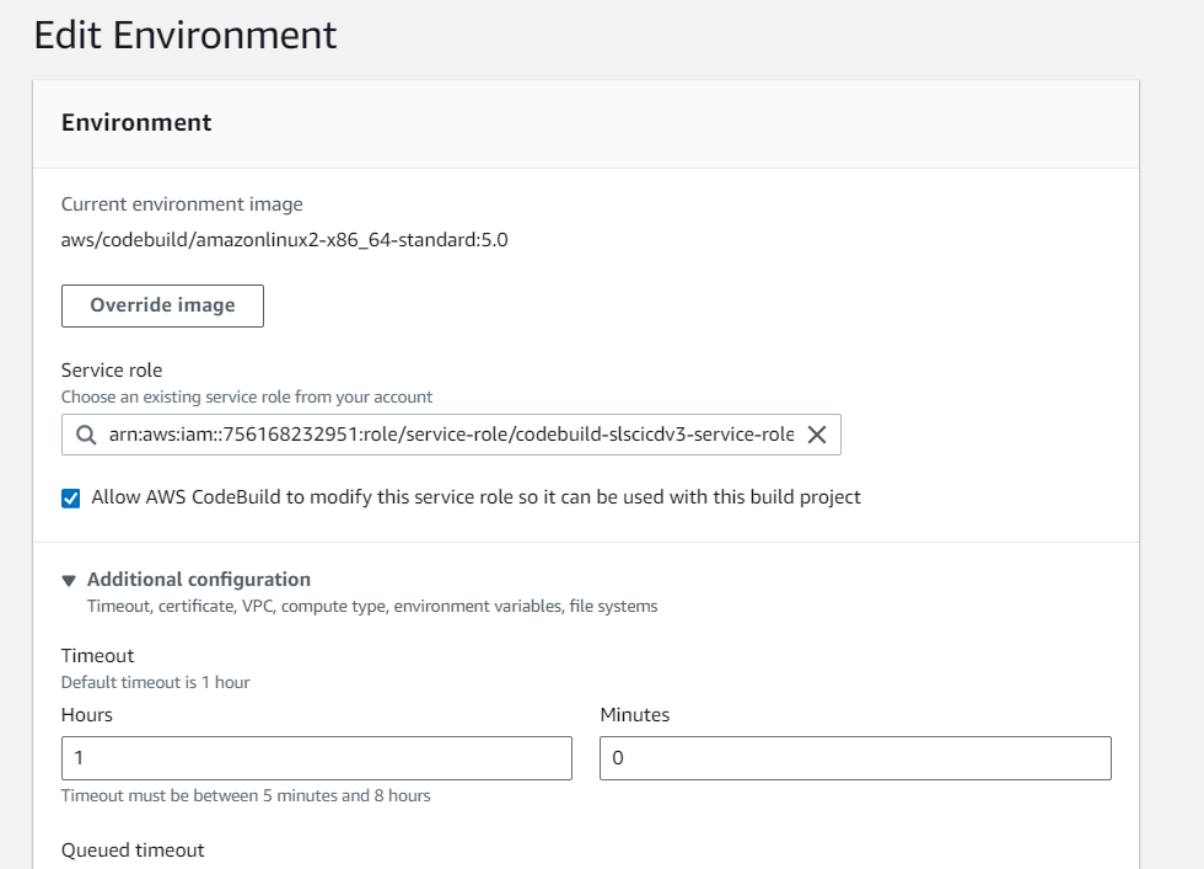
Artifact 1 - Primary

Type

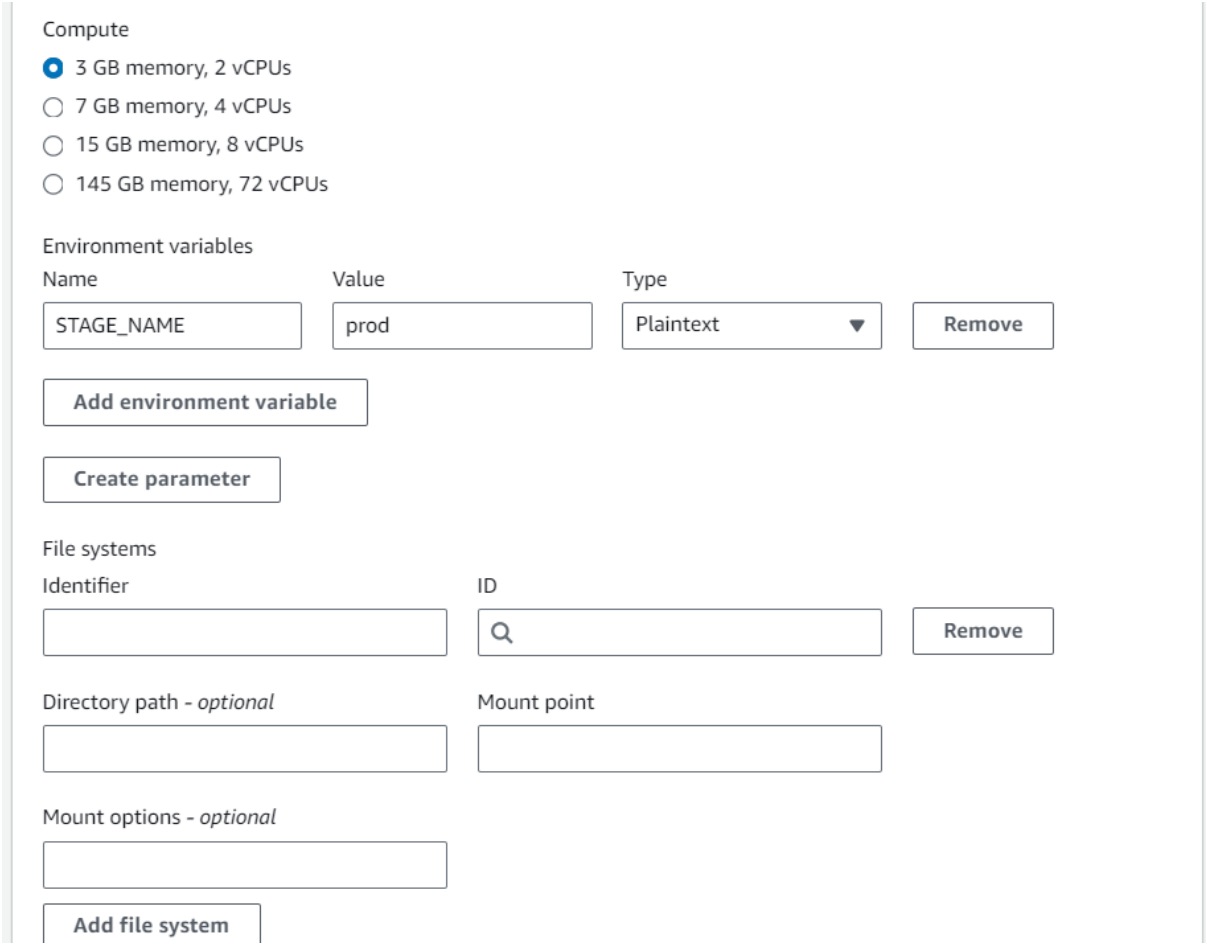
No artifacts

You might choose no artifacts if you are running tests or pushing a Docker image to Amazon ECR.

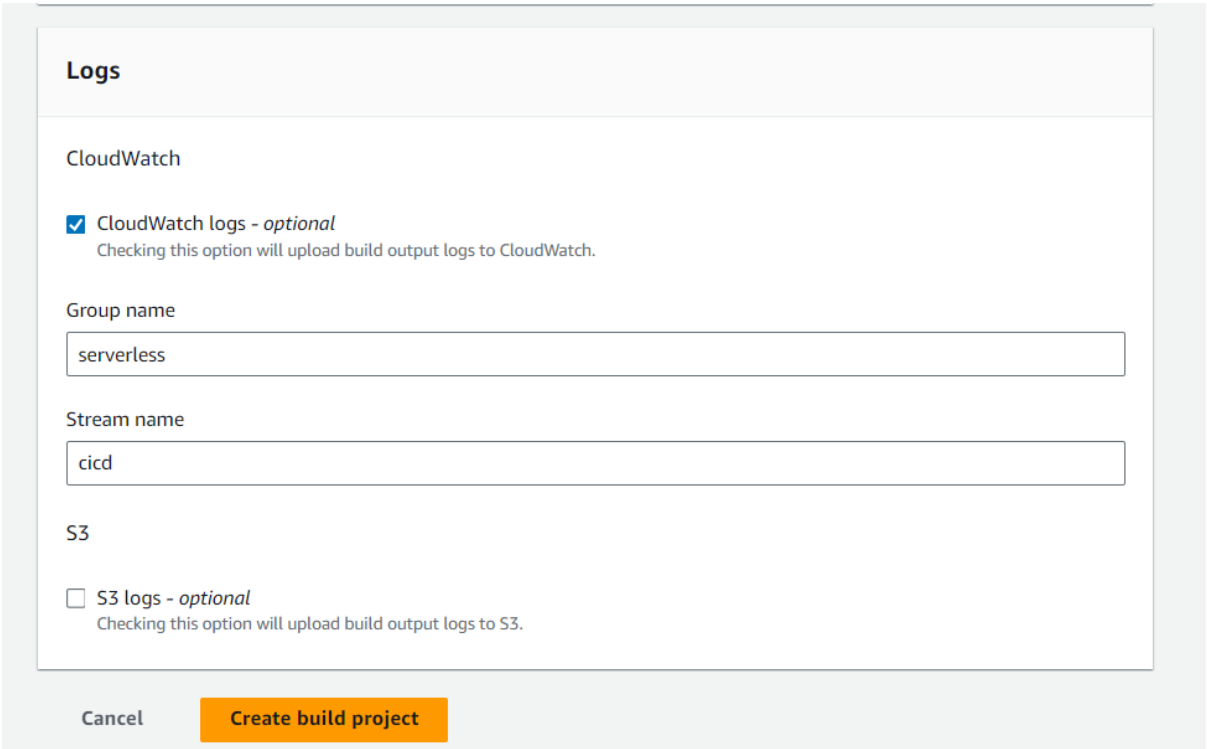
- *Under additional configuration*



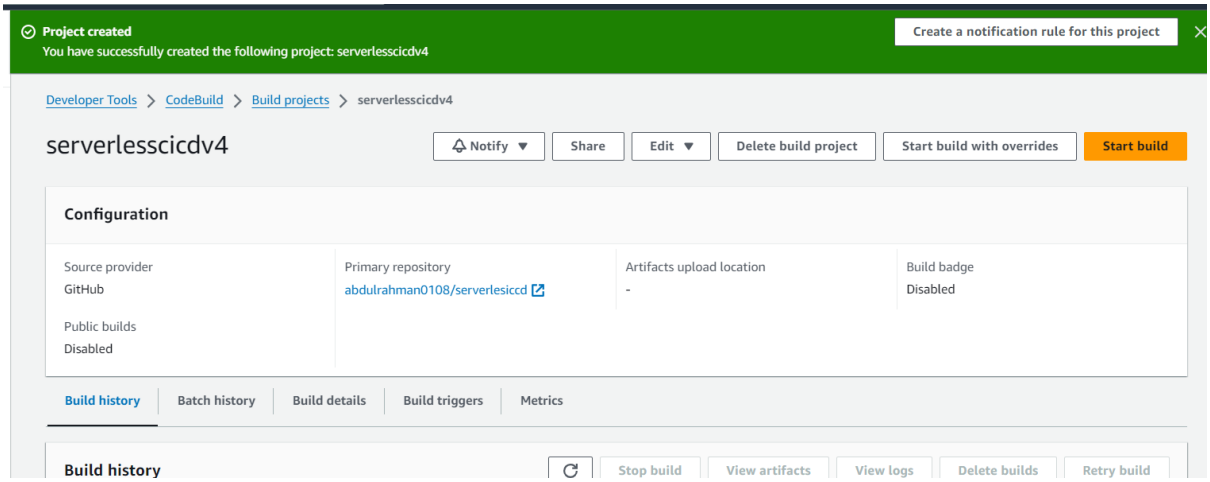
- ***Set the stage variable as given below***



➢ *Export the logs to cloudwatch if require s3 also and create build project*



➢ *Start the build and check for the logs*



- *The build will be expected to be failure cause we haven't push the buildspec.yaml*
- *Create **buildspec.yaml** in the working repository inside the project folder*

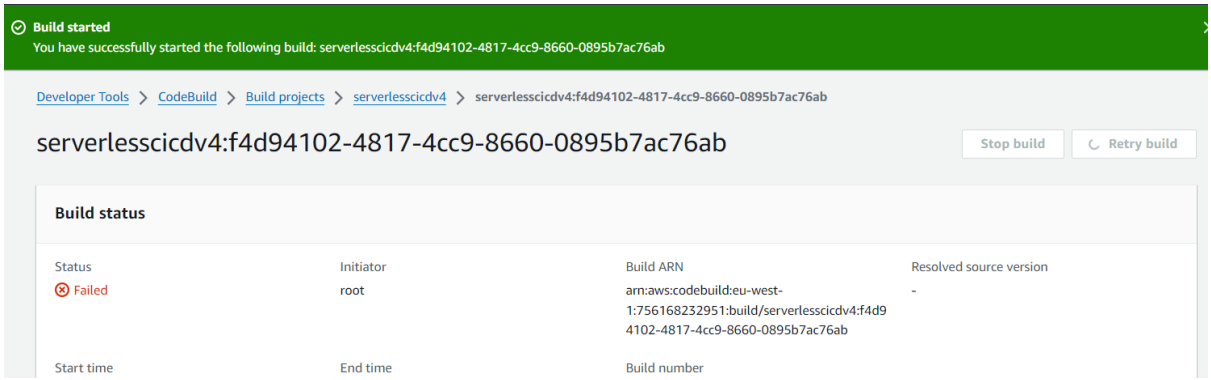
```yaml
version: 0.2
run-as: root

phases:
   install:
       runtime-versions:
          nodejs: 18
       commands:
          - npm install -g serverless
          - npm install
   build:
       commands:
          - serverless deploy --stage ${STAGE_NAME}

cache:
   paths:
      - node_modules
```

- ***Push the buildspec.yaml to github***

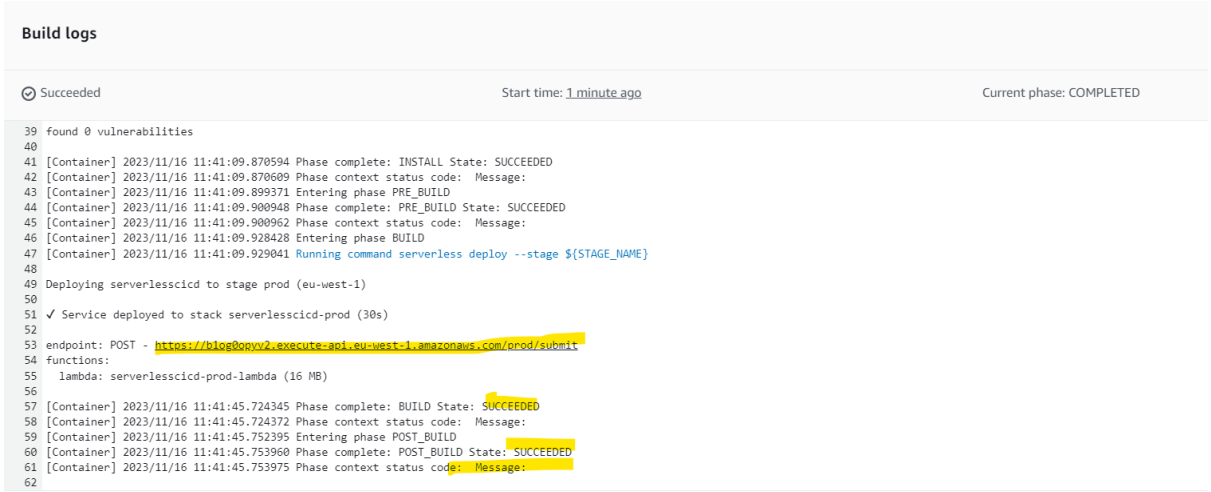
```
[ec2-user@ip-172-31-22-233 serverlesscicd]$ git push -u abd master
Username for 'https://github.com': abdulrahman0108
Password for 'https://abdulrahman0108@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 459 bytes | 459.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/abdulrahman0108/serverlesiccd.git
   6c60b7e..0235237  master -> master
branch 'master' set up to track 'abd/master'.
[ec2-user@ip-172-31-22-233 serverlesscicd]$
```

Retry the build

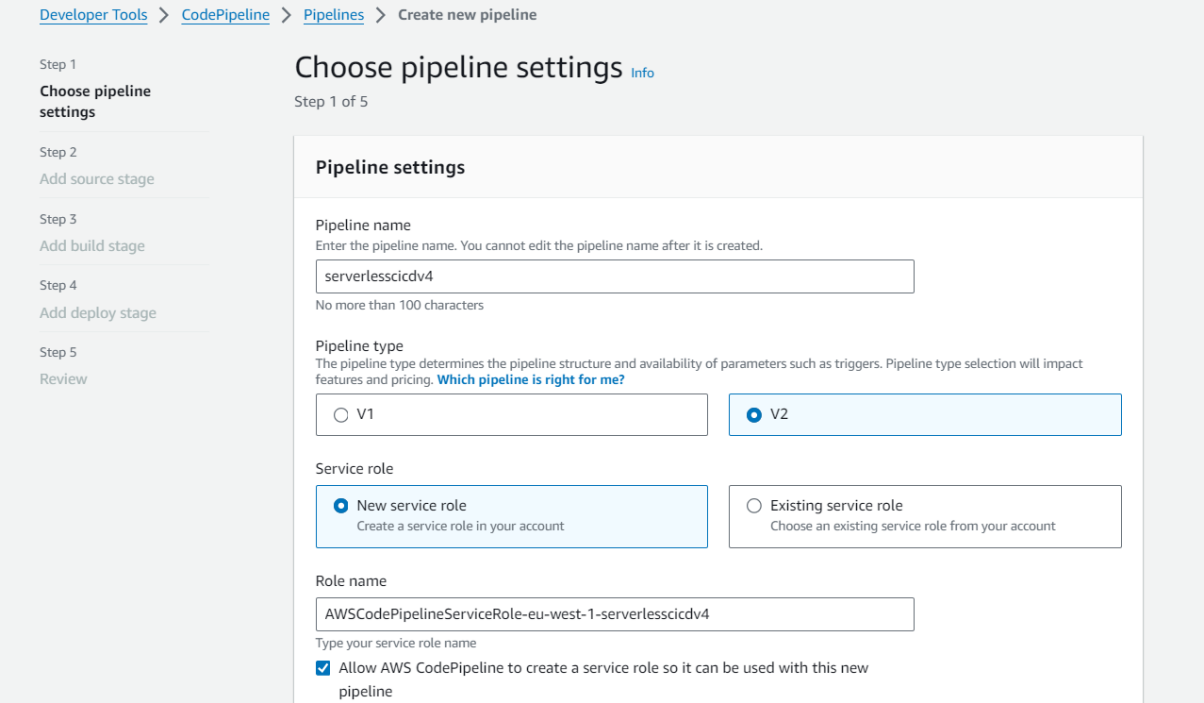
If error – Give the Roles full Admin Access



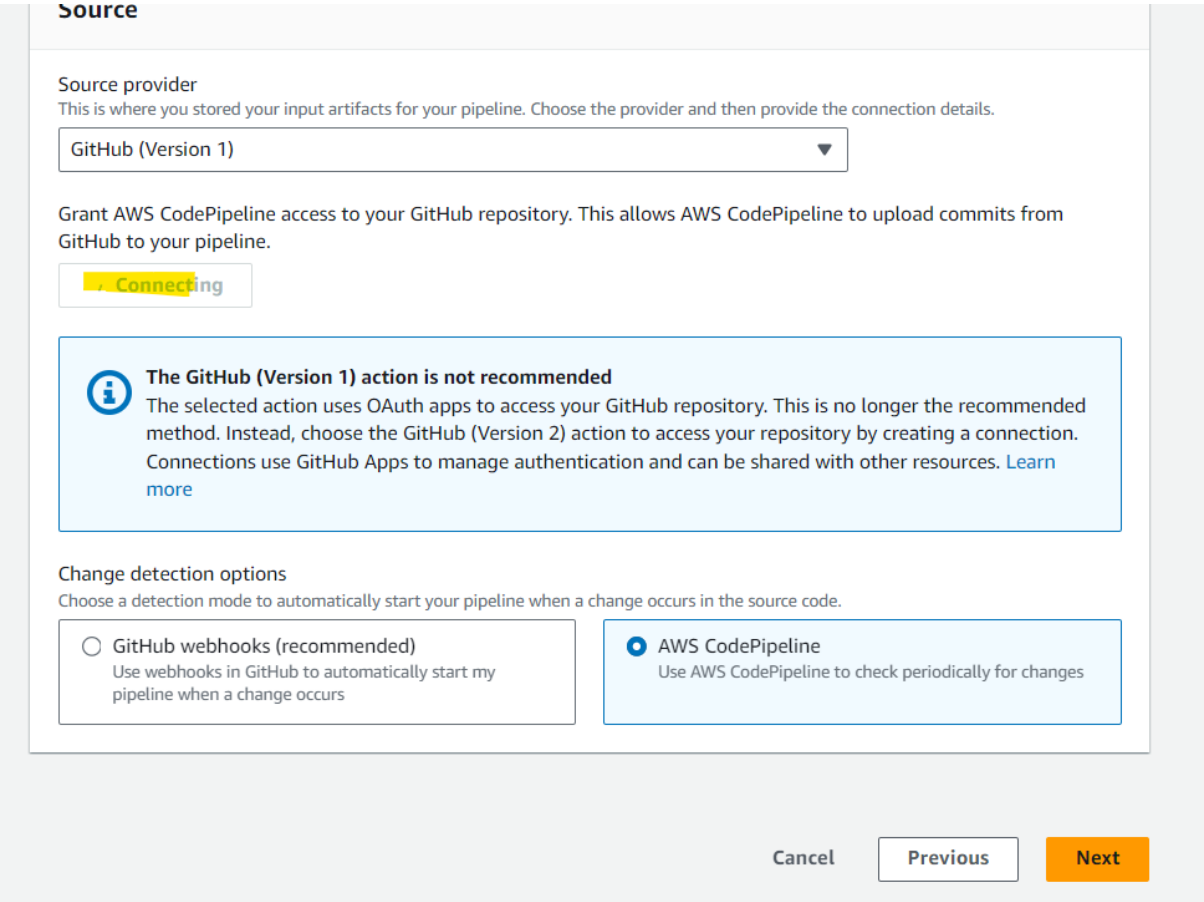
- ***You should the build succeeded state along with the api gateway url***
- ***BuiLd gives the POST – url and it cant be browsed***



- ***Now integrate the project with code pipeline***
- ***Create the code pipeline like below***



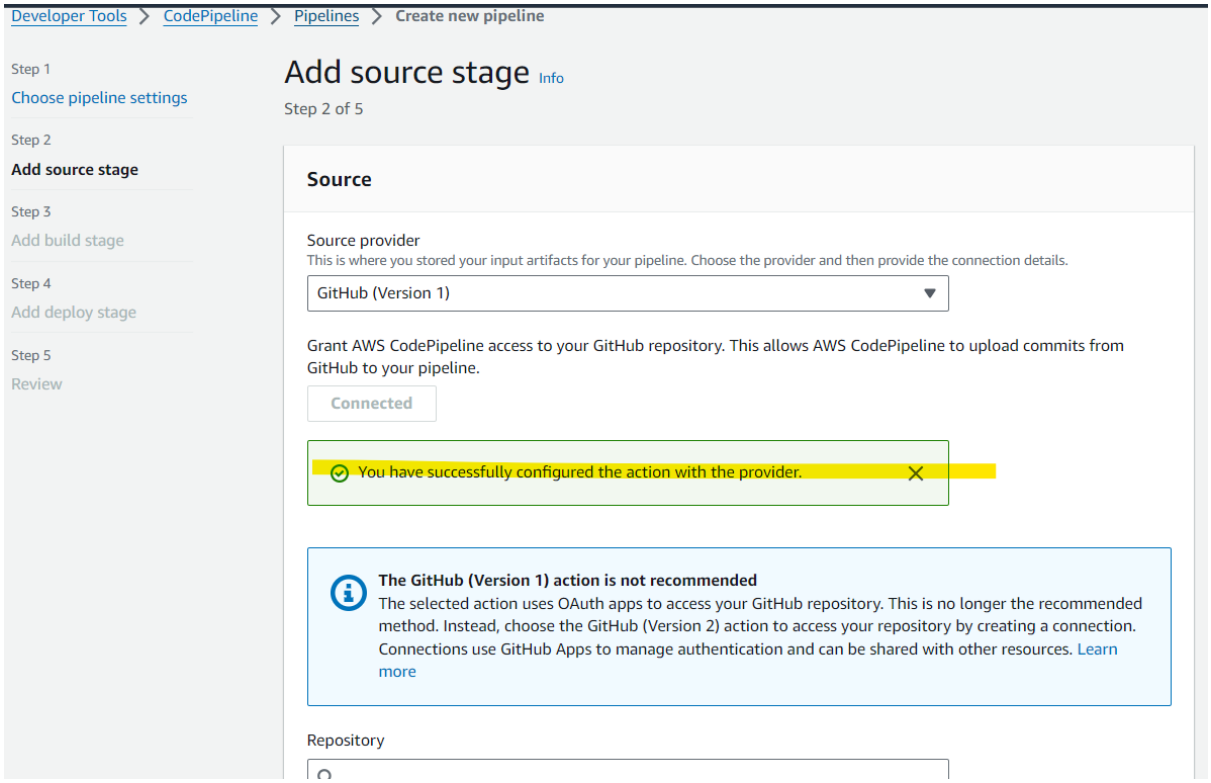
- ***Choose the source provider as github v1 and connect to github***



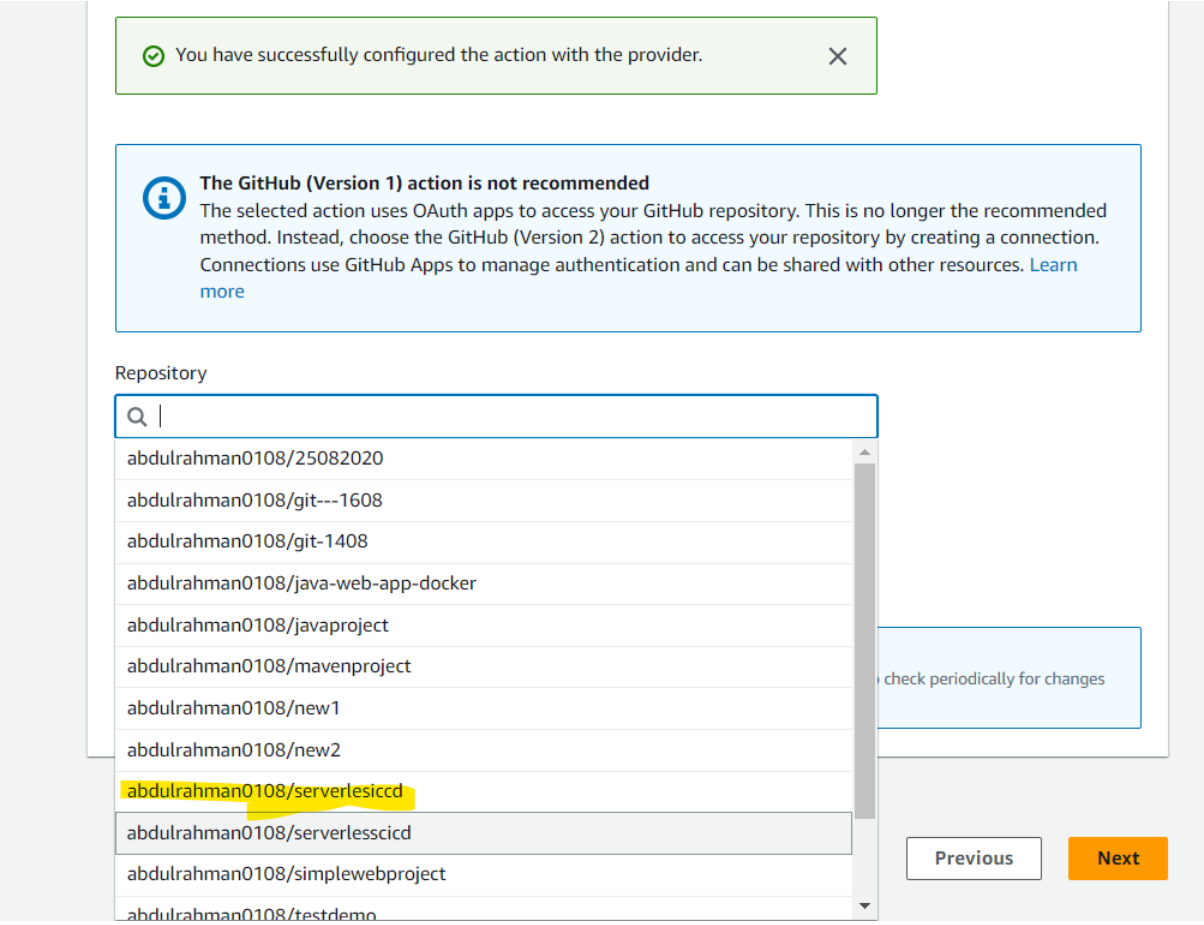
Correction : Give webhook not aws codepipeline

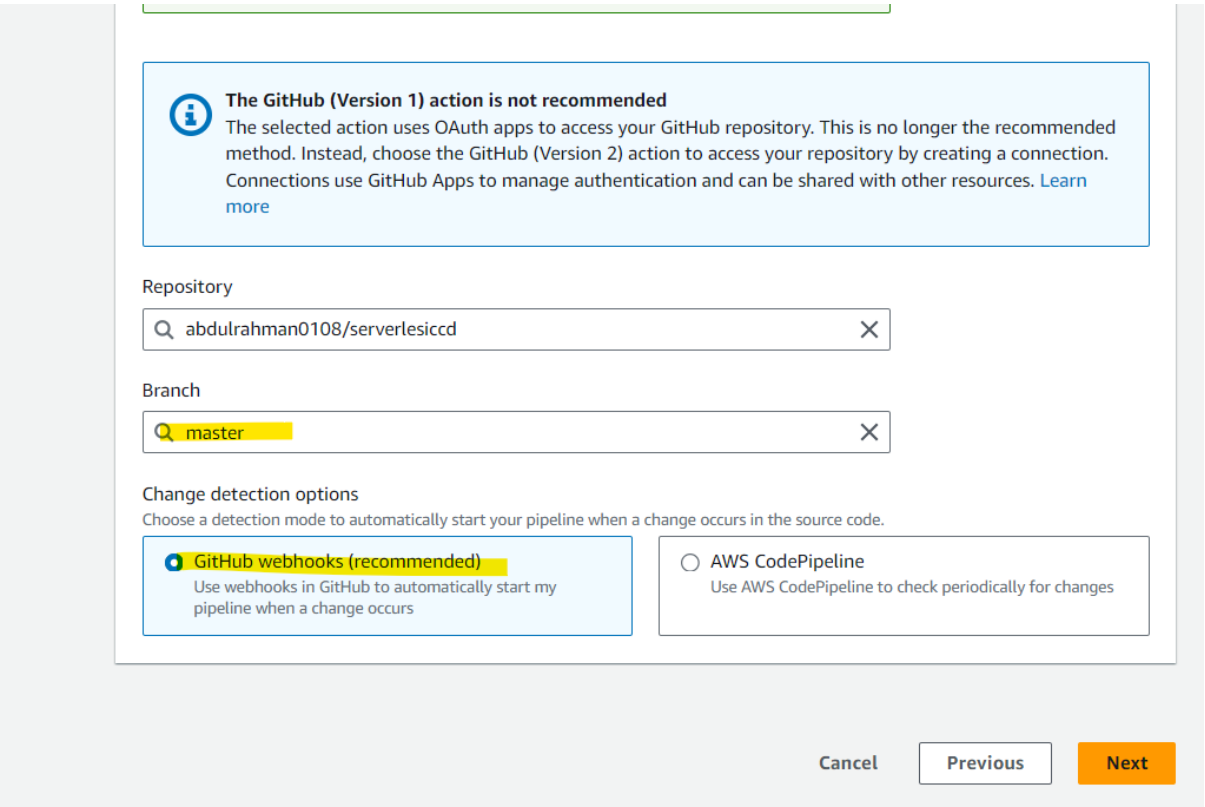
- ***Give a connection name***

- ***Click authorize aws-codesuite***
- **It will prompt to authorize – click on authorize**

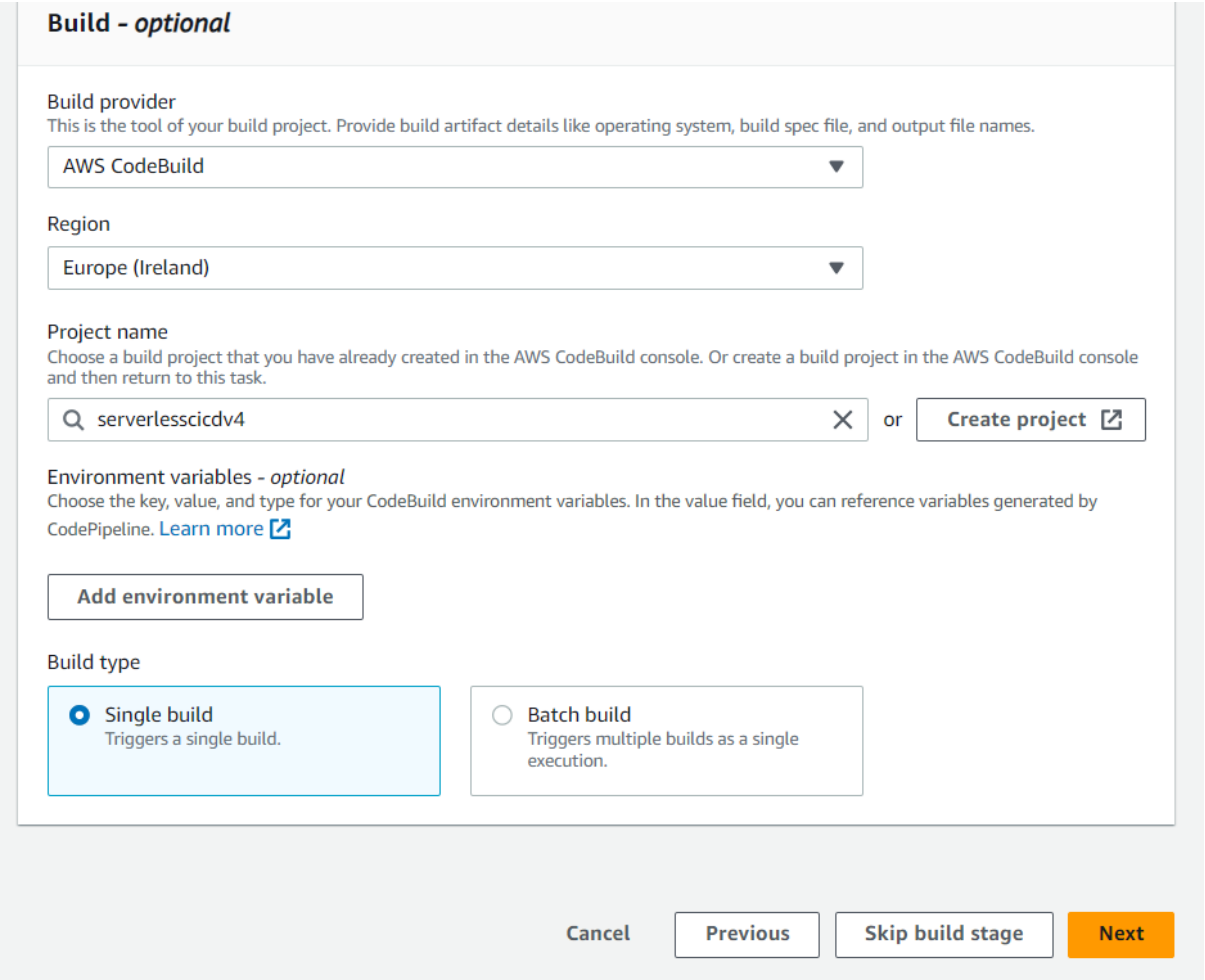


➢ ***Choose your repository with master branch***

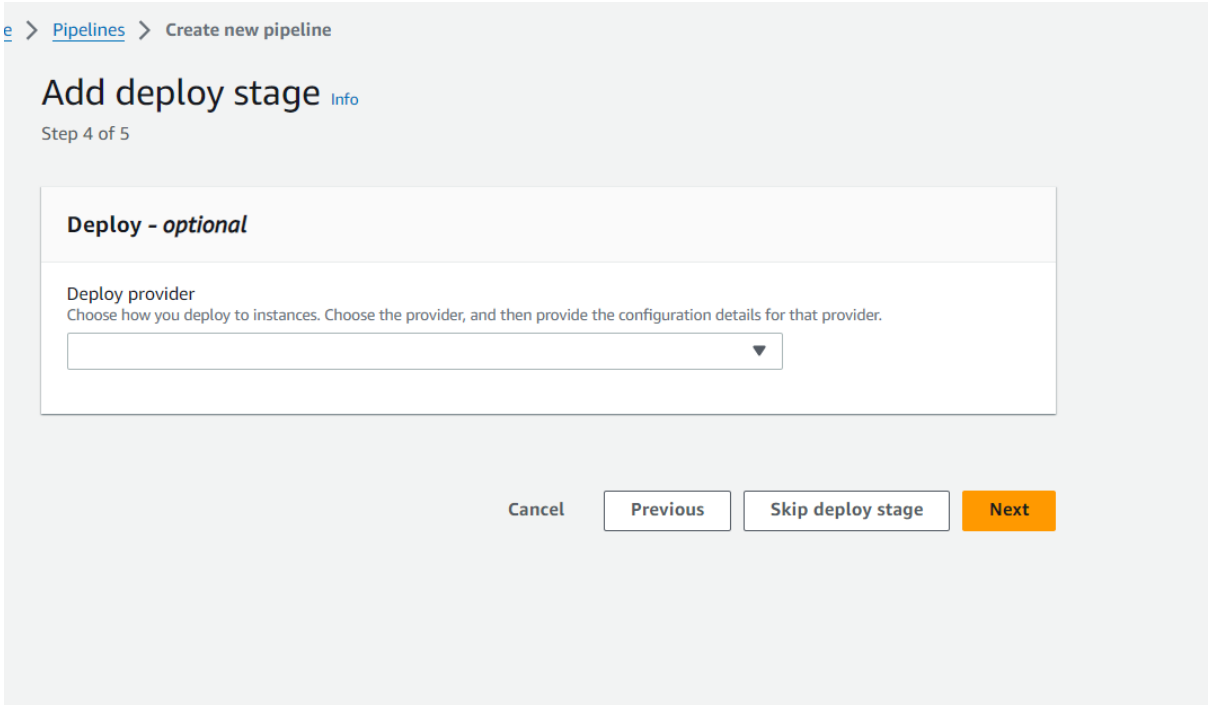




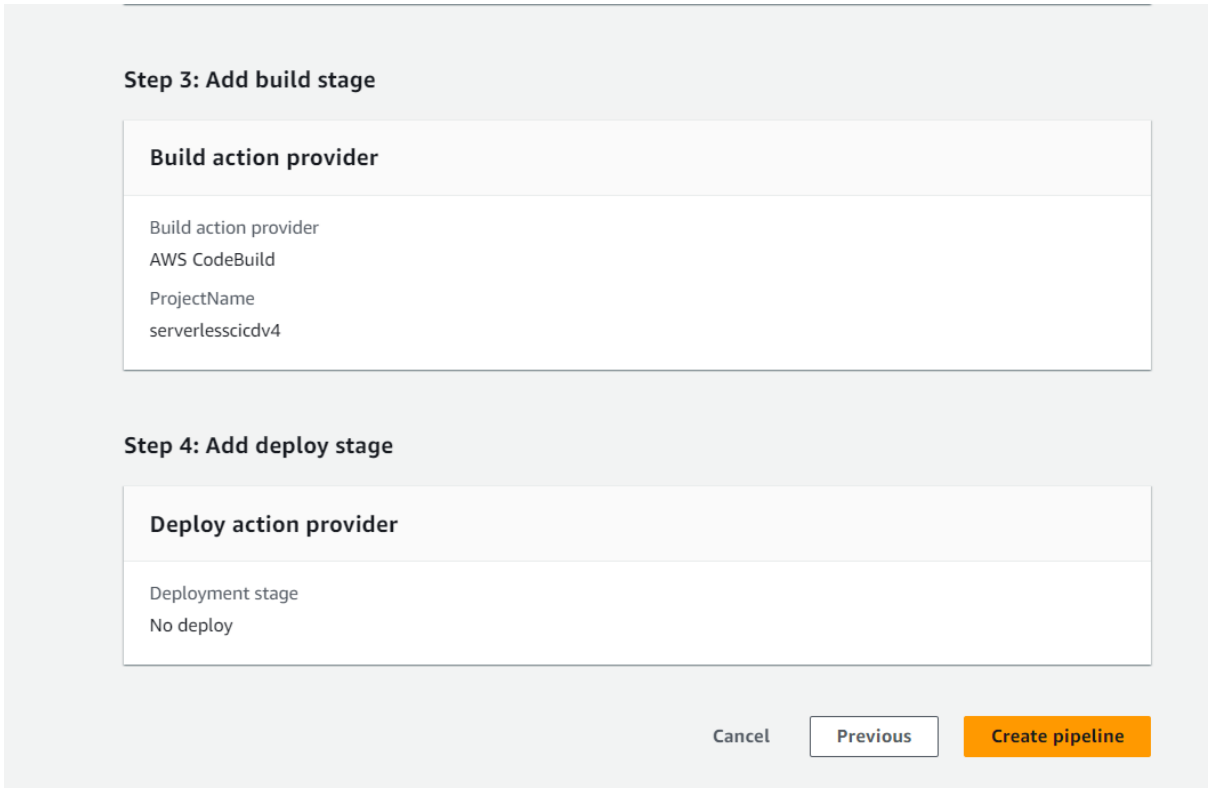
- ***Give the build provider name***



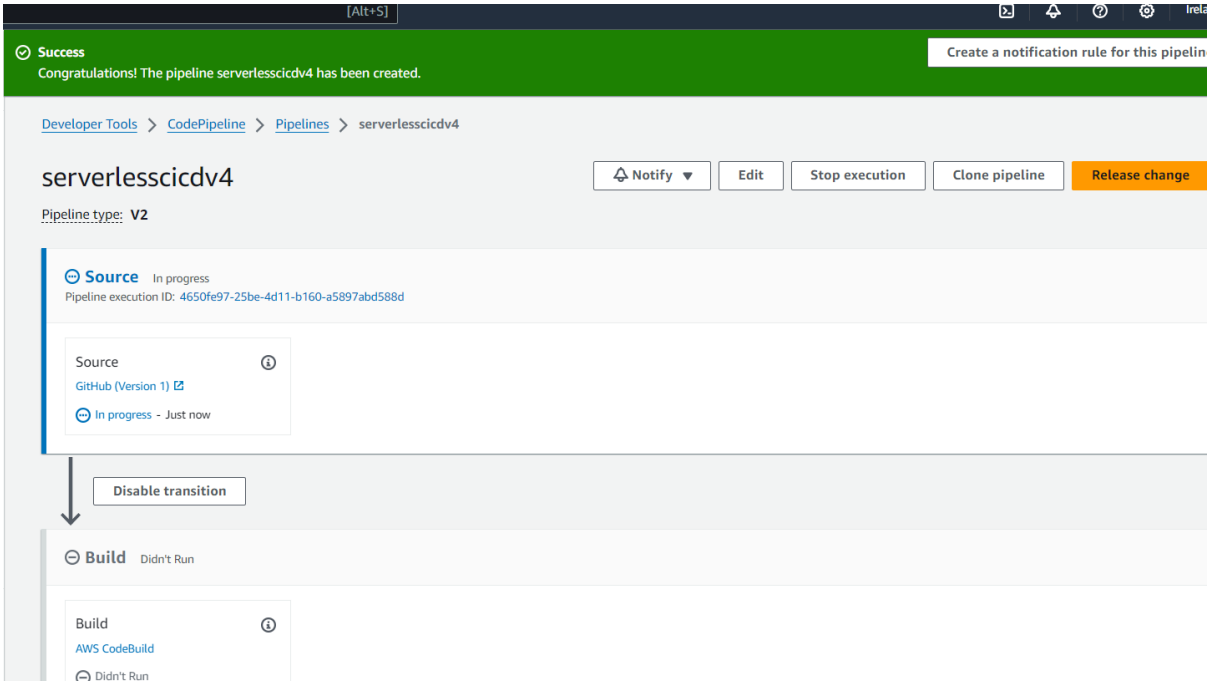
- ***As we have given the deploy in the buildspec yaml itself we can skip the integration with code deploy***



- ***Create pipeline***



- ***After creating the pipeline , the pipeline will run a dry build***.



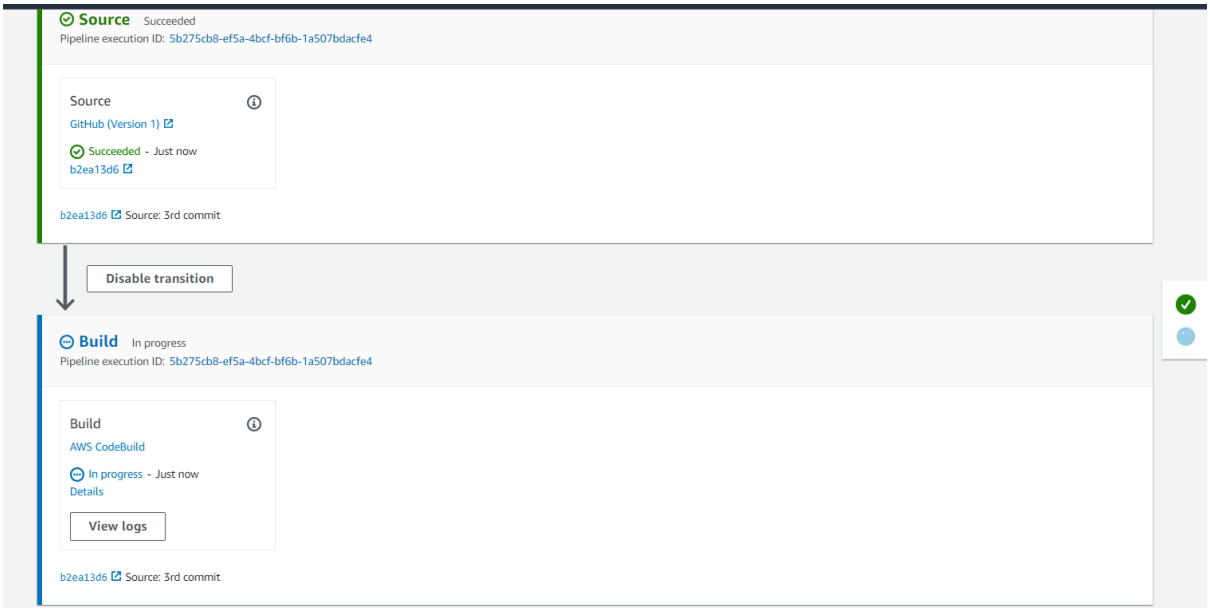
- ***Thus we have successfully integrated the project with lambda, api gateway, dynamodb,s3, github,codebuild, code pipeline***
- ***Open the index.js in the local repository and edit the lambda code***

```
  try {
    console.log('Raw input data:', event); // Add this line to log the raw input data

    const formData = {
      name: event.name,
      email: event.email,
      subject: event.subject,
      message: event.message,
    };

    const item = {
      SubmissionId: generateUUID(), // Generate a UUID
      ...formData, // Use the form data as attributes
    };

    // Store the form data in DynamoDB
    await storeFormData(item);

    return {
      statusCode: 200,
      body: JSON.stringify({ message: 'Form submitted successfully and this is serverless cicdv5' }),
    };
  } catch (error) {
    console.error(error);
    return {
      statusCode: 500,
      body: JSON.stringify({ message: 'Error submitting the form' }),
    };
  }
};
// here we will provide the dynamodb table name
async function storeFormData(item) {
  const params = {
    TableName: 'ContactFormEntries2',
    Item: item,
  };

  await dynamodb.put(params).promise();
}

function generateUUID() {
  return uuidv4();
```

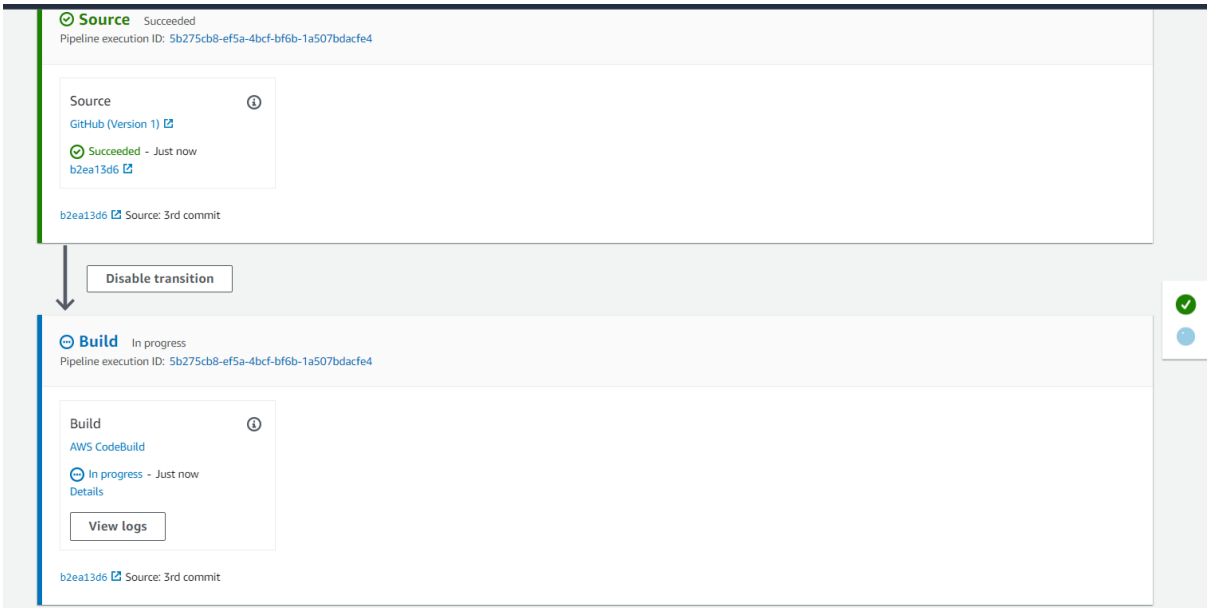
- ***When you push the changes***

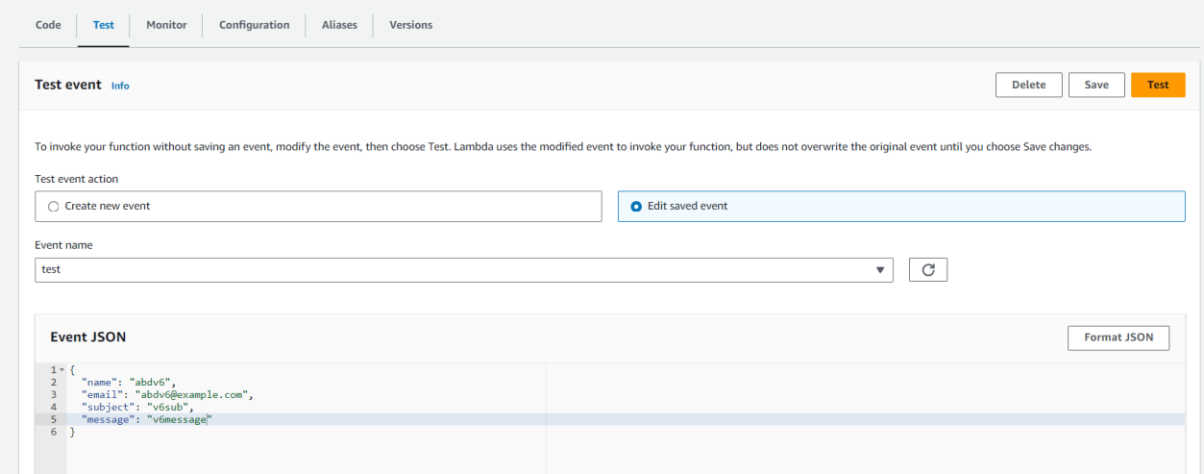
```
[ec2-user@ip-172-31-22-233 serverlesscicd]$ git push -u abd master
Username for 'https://github.com': abdulrahman0108
Password for 'https://abdulrahman0108@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 316 bytes | 316.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/abdulrahman0108/serverlesiccd.git
   0235237..b2ea13d  master -> master
branch 'master' set up to track 'abd/master'.
[ec2-user@ip-172-31-22-233 serverlesscicd]$
```

- The build will be started again

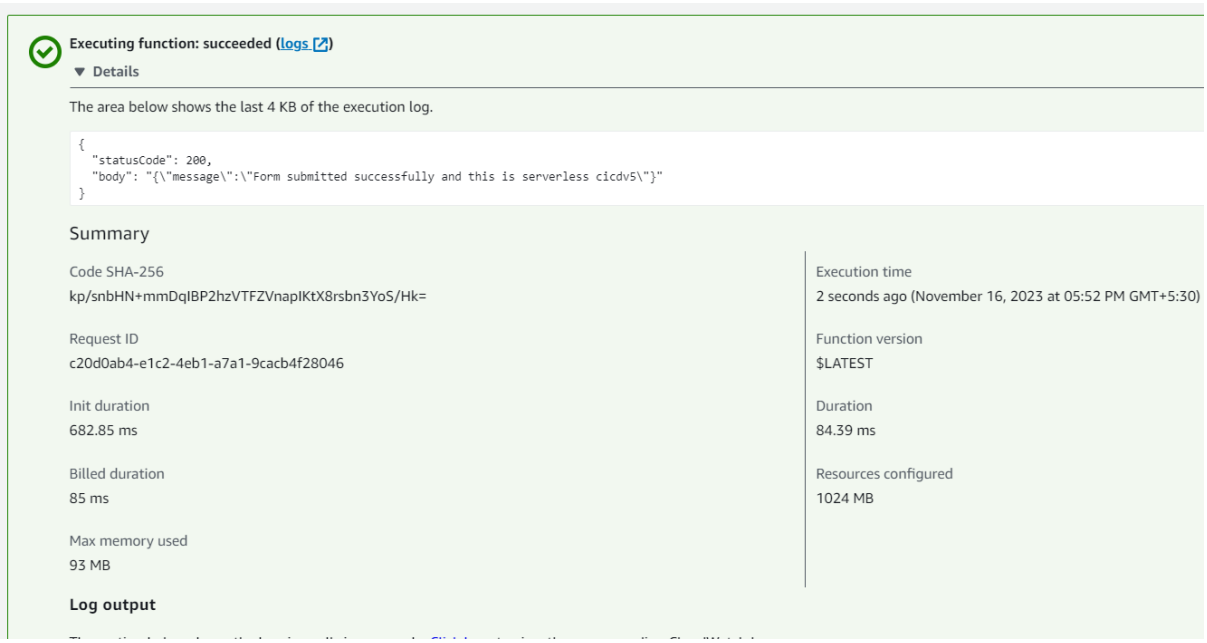


- ***Go to lambda and edit the saved test event***





- ***Test it***
- **Here the lambda response code is changed to v6**



- ✓ *Thus if any changes in github repository will identified by githook in the code pipeline and it trigger the code build to build the project from the github repository and deploy the changes in the code.* ***Thus we have achieved CI/CD integration.***

Interview

Frame the questions with integrating with services

1. **ECS - deploy and usage , procedure**
2. **IAM Roles**
3. ***List the aws services u know***
4. ***say something about hobby and yourself***
5. ***How do you know about AWS and why do u choose it***
6. ***How to intregrate ECS and DynamoDb -***
7. ***Aws SDK***
8. ***What are services and Database U worked with list ? ---? ?***
9. ***How to install postgre in EC2 (Browse)***
10. ***How to change website http to https***
11. ***400 – Client request error***
12. ***503 error – file not inside so check the file/permission/handler***
13. ***200 – success***