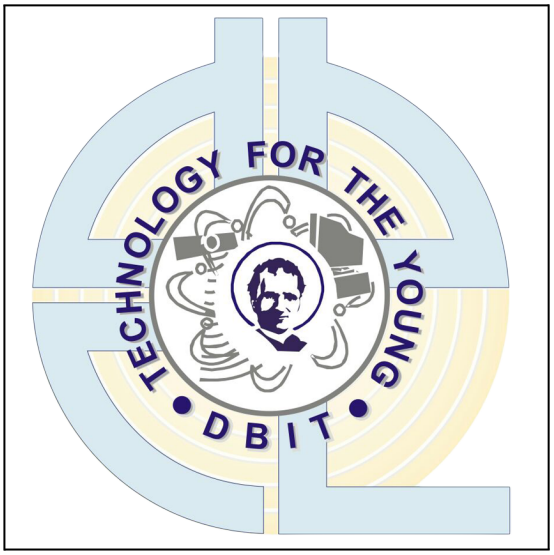
Don Bosco Institute of Technology(DBIT), Mumbai

Department of Information Technology



LAB JOURNAL

On

I.T/T.E/Sem V/ITL504 : AdvDevOps Lab

By

43 Ravi Pandey

Academic Year : Nov,2022

INDEX

Sr no	Exp No	Name of the experiment	Date
1.	1	To understand the benefits of Cloud Infrastructure and Setup AWS Cloud9 IDE, Launch AWS Cloud9 IDE and Perform Collaboration Demonstration	22/07/2022
2.	2	To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.	31/07/2022
3	5	To understand terraform lifecycle, core concepts/terminologies and install it on a Linux Machine.	20/08/2022
4	6	To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform.	27/09/2022
5	7	To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.	03/09/2022
6	8	Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application	10/09/2022
7	9	To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin	17/09/2022

		Executor) on Linux Machine	
8	10	To perform Port, Service monitoring, Windows/Linux server monitoring using Nagios	24/09/2022
9	11	To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs	01/10/2022
10	12	To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.	08/10/2022

AdvDevOps

Experiment No 1

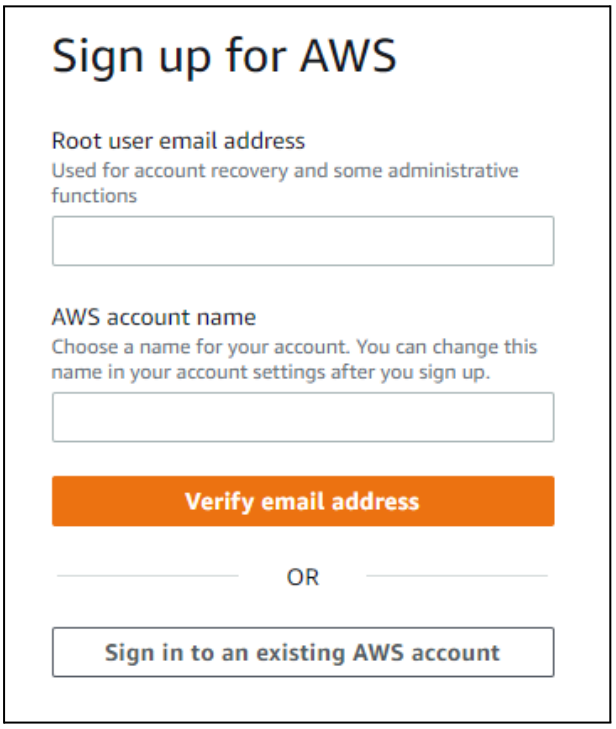
Name - Ravi Pandey
Roll no - 43
Batch - B
Date - 22/07/2022

Aim - To understand the benefits of Cloud Infrastructure and Setup AWS Cloud9 IDE, Launch AWS Cloud9 IDE and Perform Collaboration Demonstration

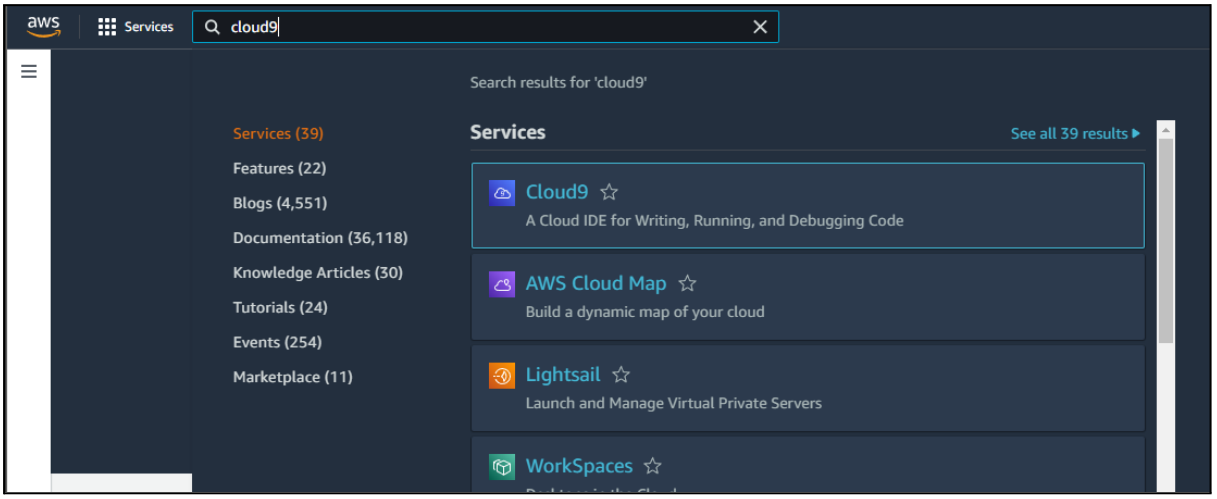
- **What is AWS?**

- Amazon Web Services, Inc. (AWS) is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis. These cloud computing web services provide distributed computing processing capacity and software tools via AWS server farms. One of these services is Amazon Elastic Compute Cloud (EC2), which allows users to have at their disposal a virtual cluster of computers, available all the time, through the Internet. AWS's virtual computers emulate most of the attributes of a real computer, including hardware central processing units (CPUs) and graphics processing units (GPUs) for processing; local/RAM memory; hard-disk/SSD storage; a choice of operating systems; networking; and pre-loaded application software such as web servers, databases, and customer relationship management (CRM).

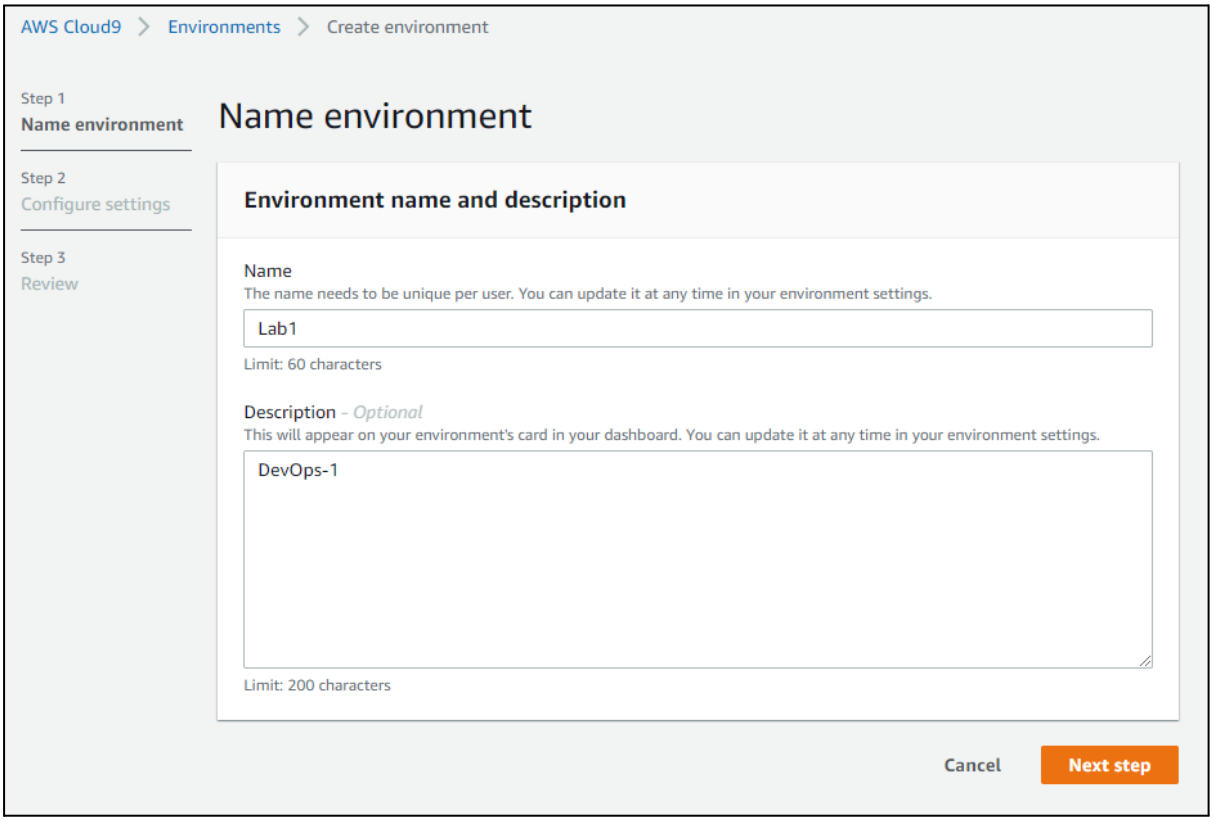
- **Step - 1**
  - **Create a free account on AWS**



- **Step -2**
  -Search for cloud9



- **Step-3**
  - **Create environment and write sample code**



Environment name and settings

Name

Lab1

Description

DevOps-1

Environment type

EC2

Instance type

t2.micro

Subnet

Platform

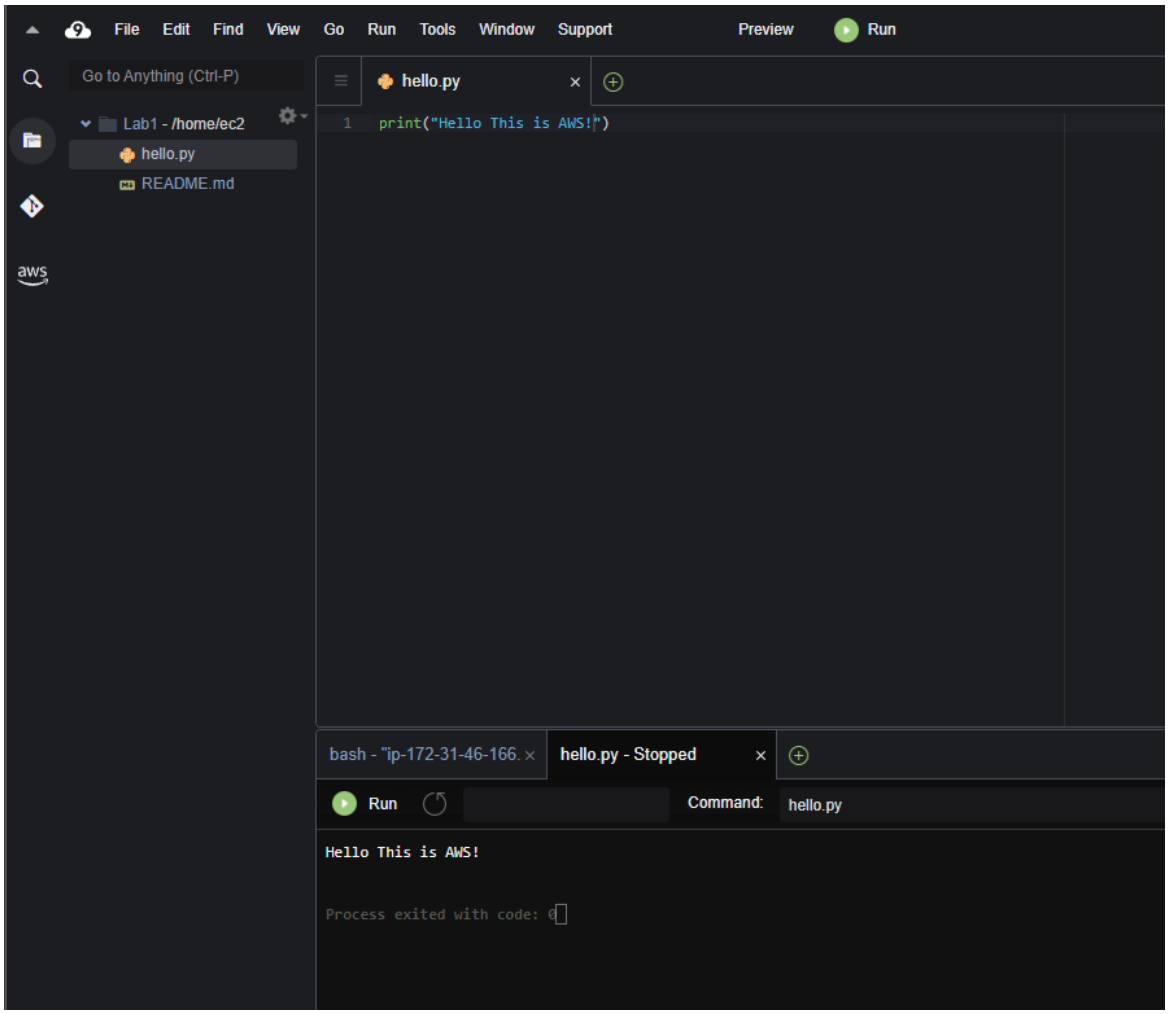
Amazon Linux 2 (recommended)

Cost-saving settings

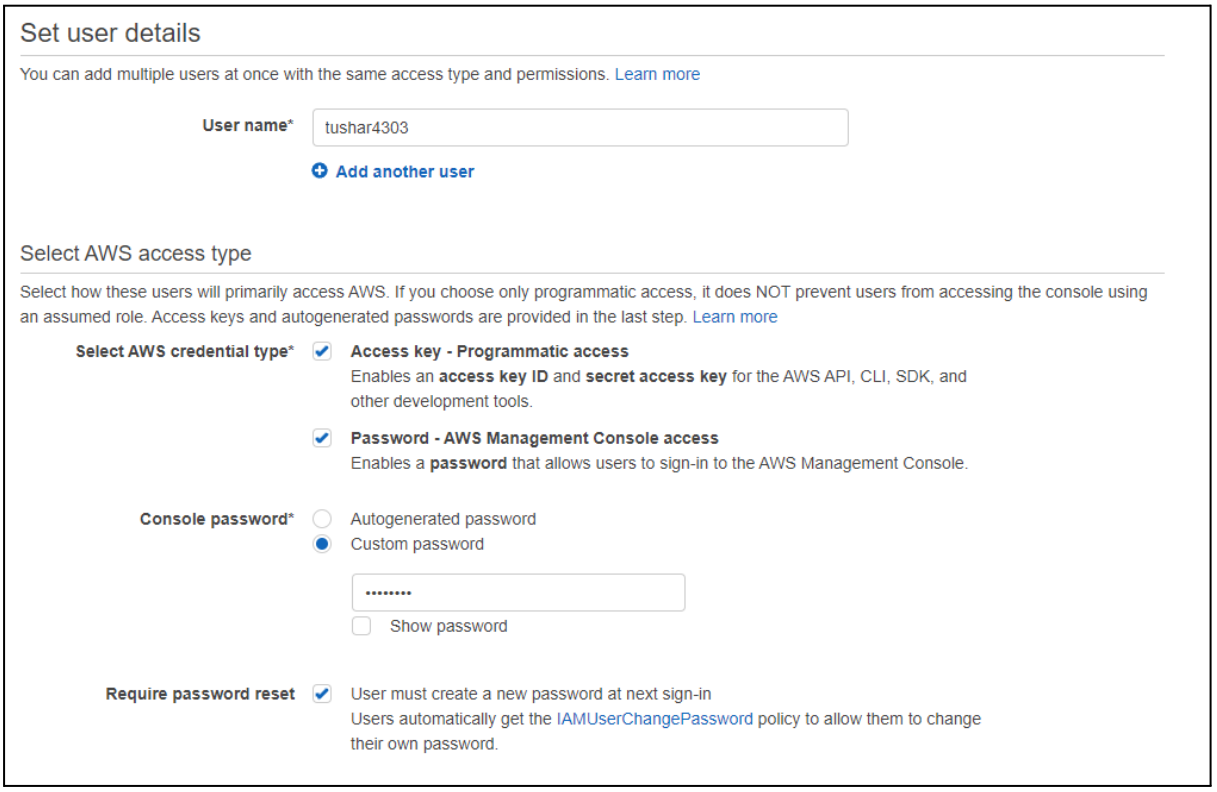
After 30 minutes (default)

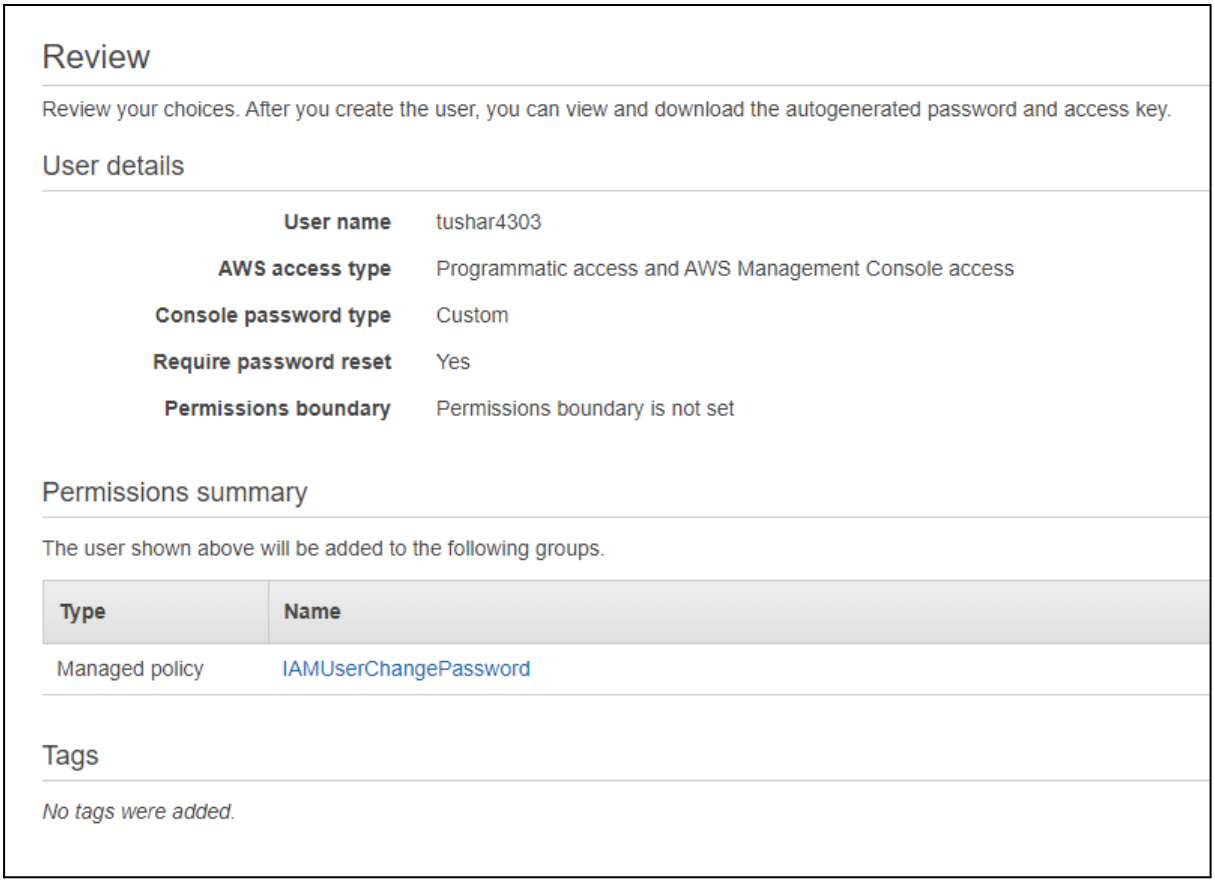
IAM role

AWSServiceRoleForAWSCloud9 (generated)

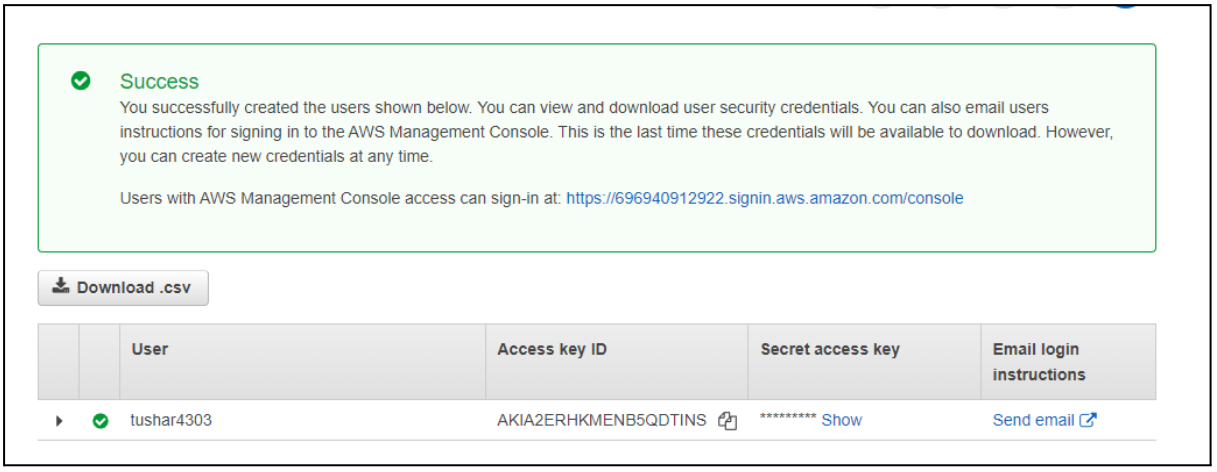


- **Step - 4**
  - **Add new IAM User**



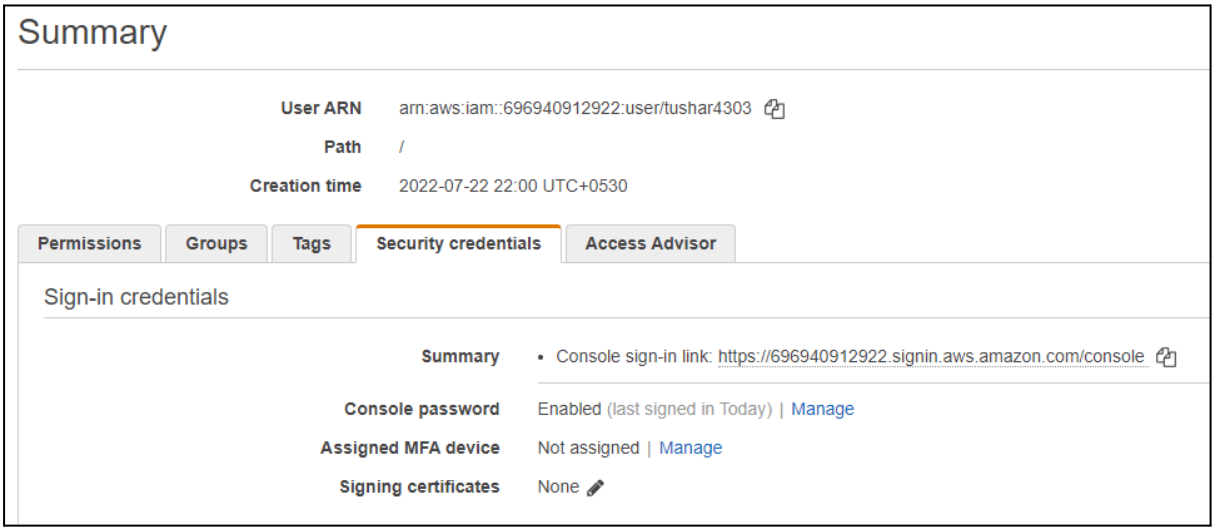


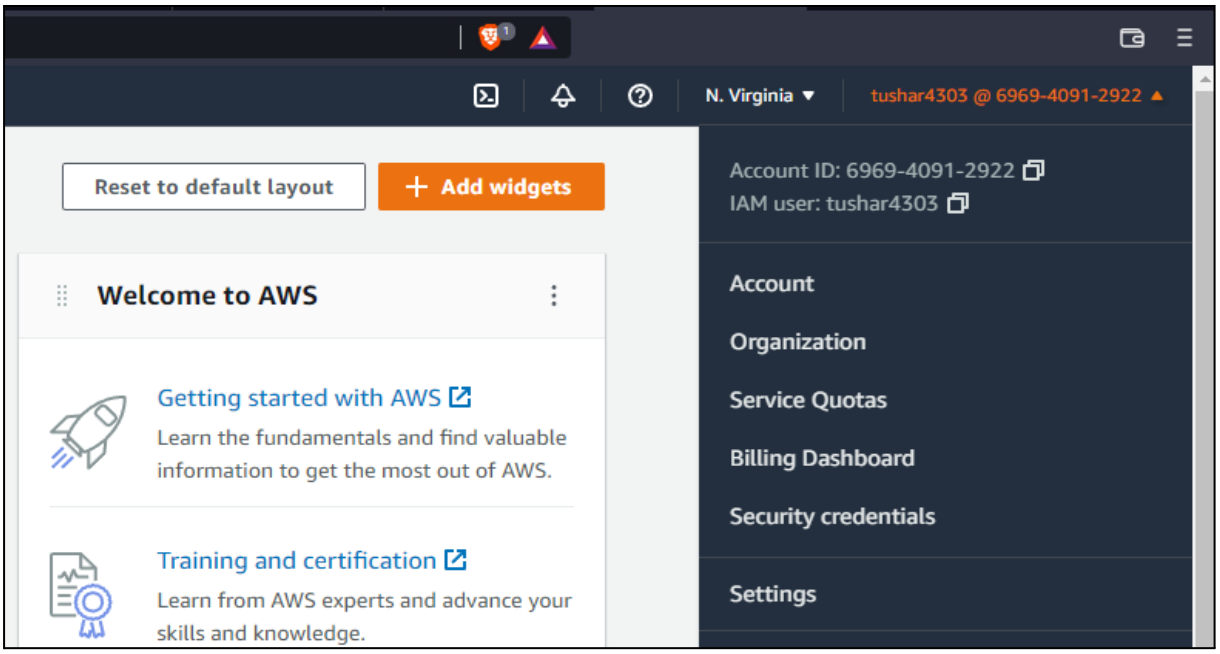
- **Step-5**
  - **Download the credentials**



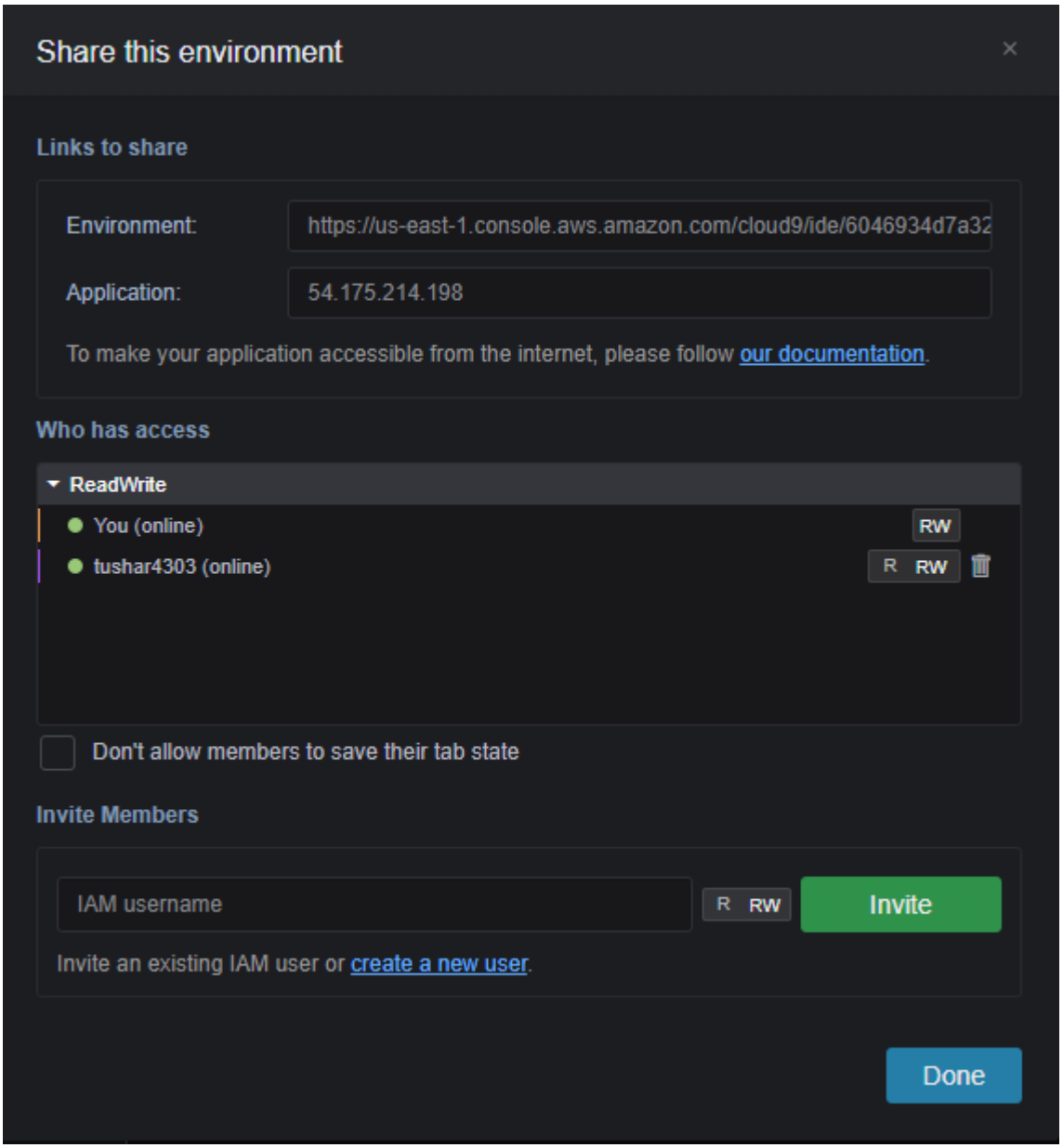
- **Step-6**

  -Copy the console link and open in other browser

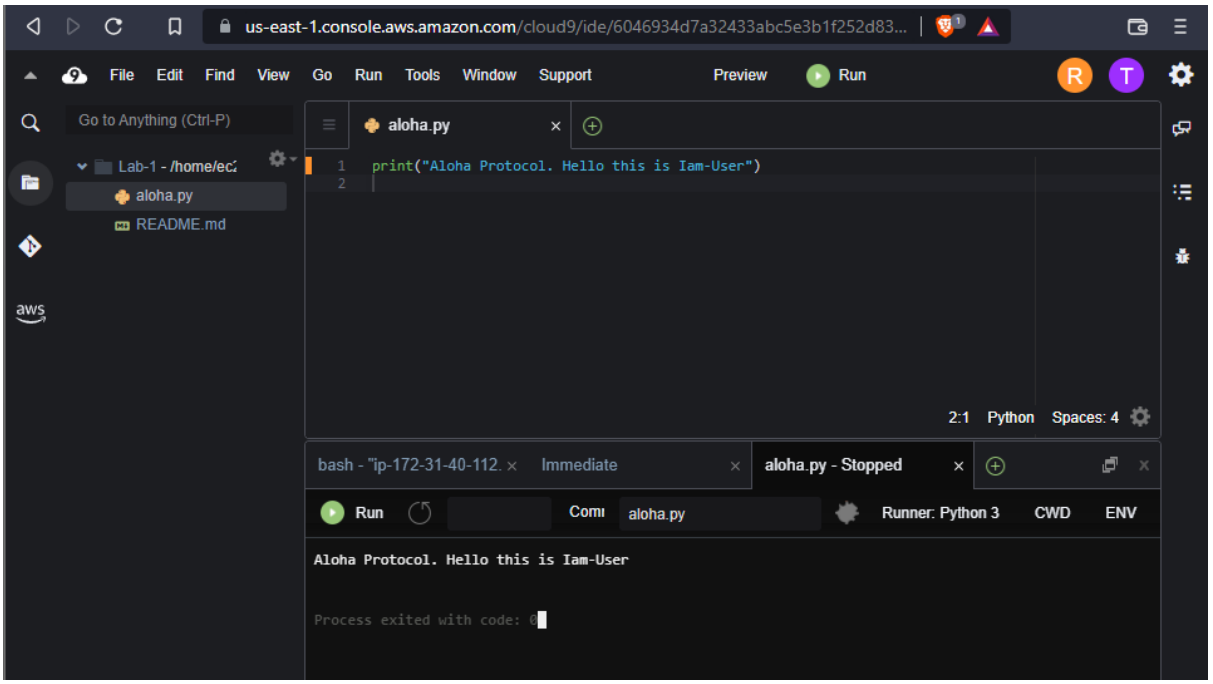




- **Step-7**
  -Copy the environment ID



- **Step-8**
  -Paste the environment ID in other browser and make some changes to verify the connection



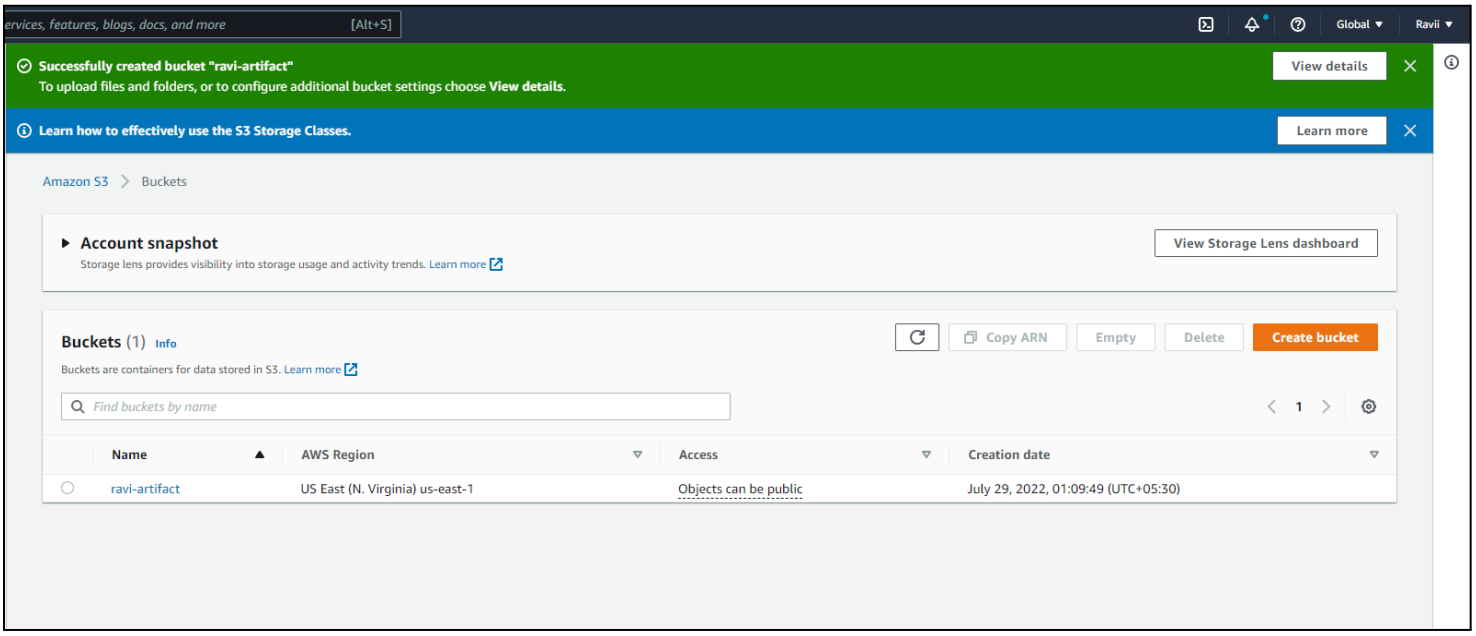
ADVANCE DEVOPS LAB

Name : Ravi Pandey
Roll No. - 43
Experiment No.: 02

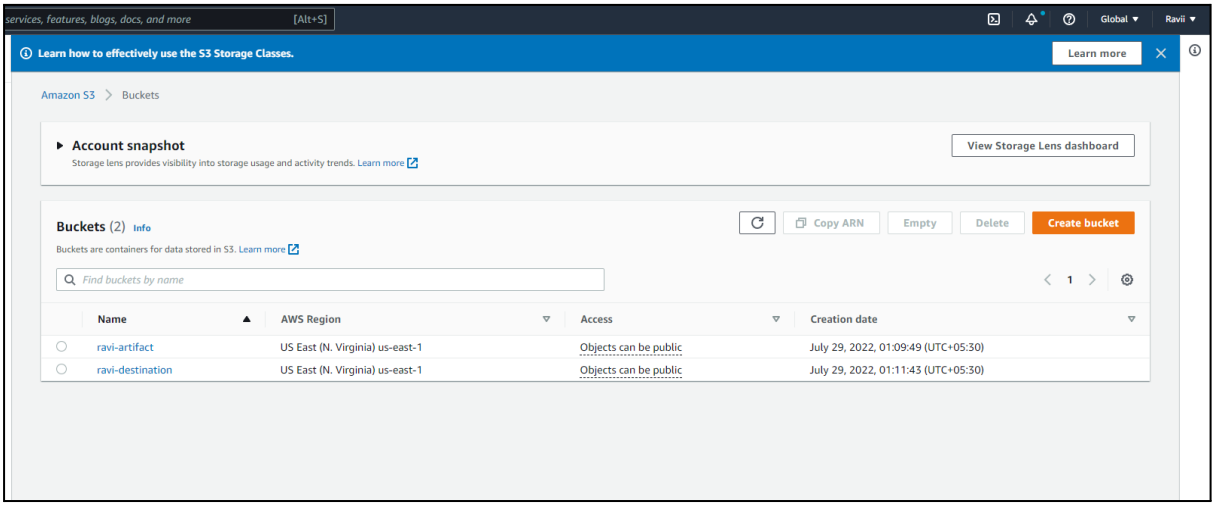
Aim :To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

OutPut :

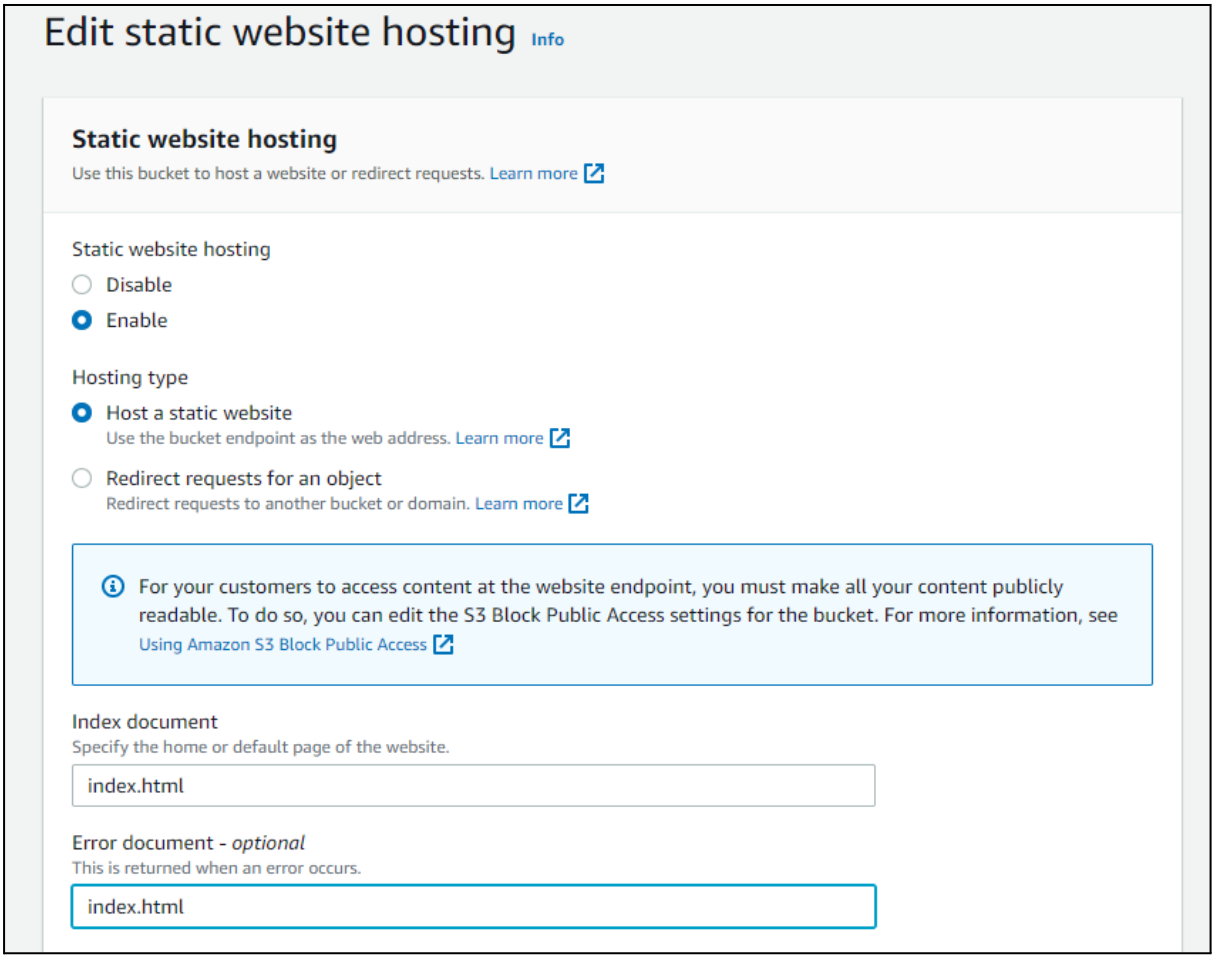
STEP 1 :



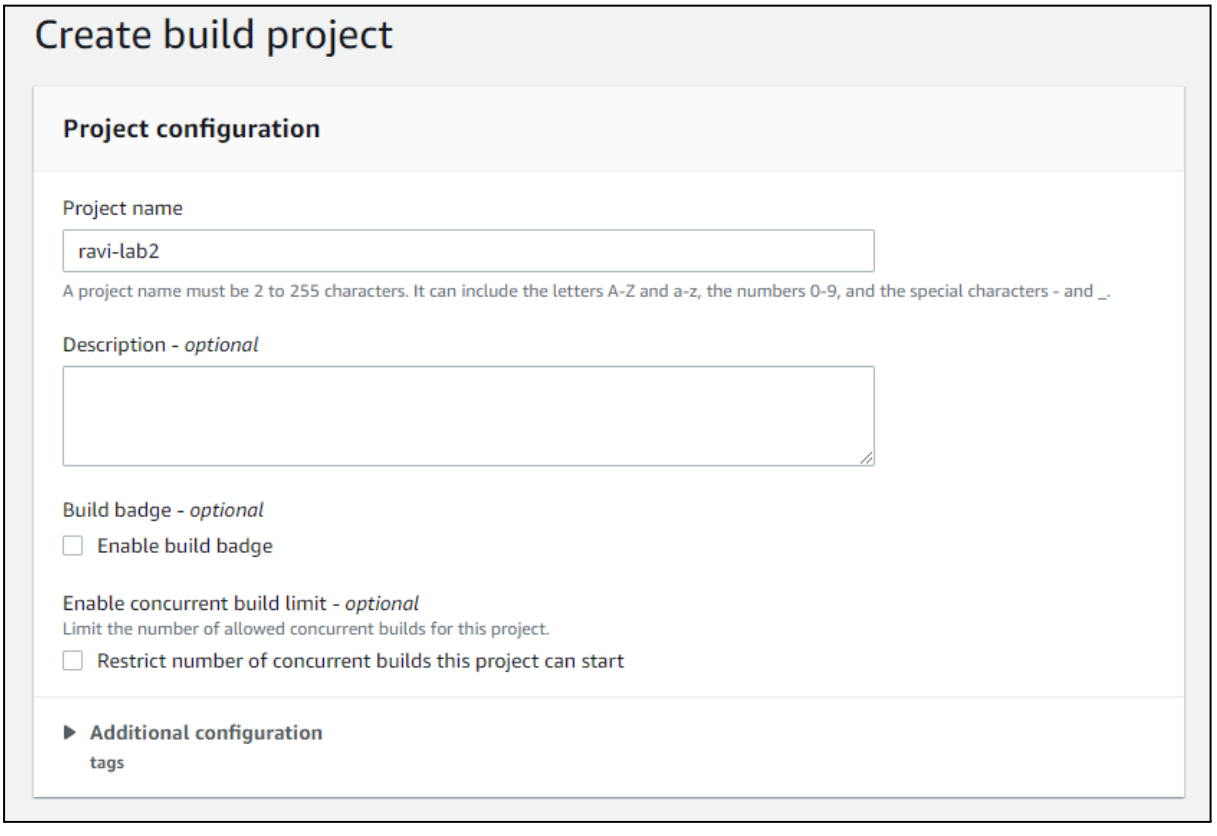
STEP 2



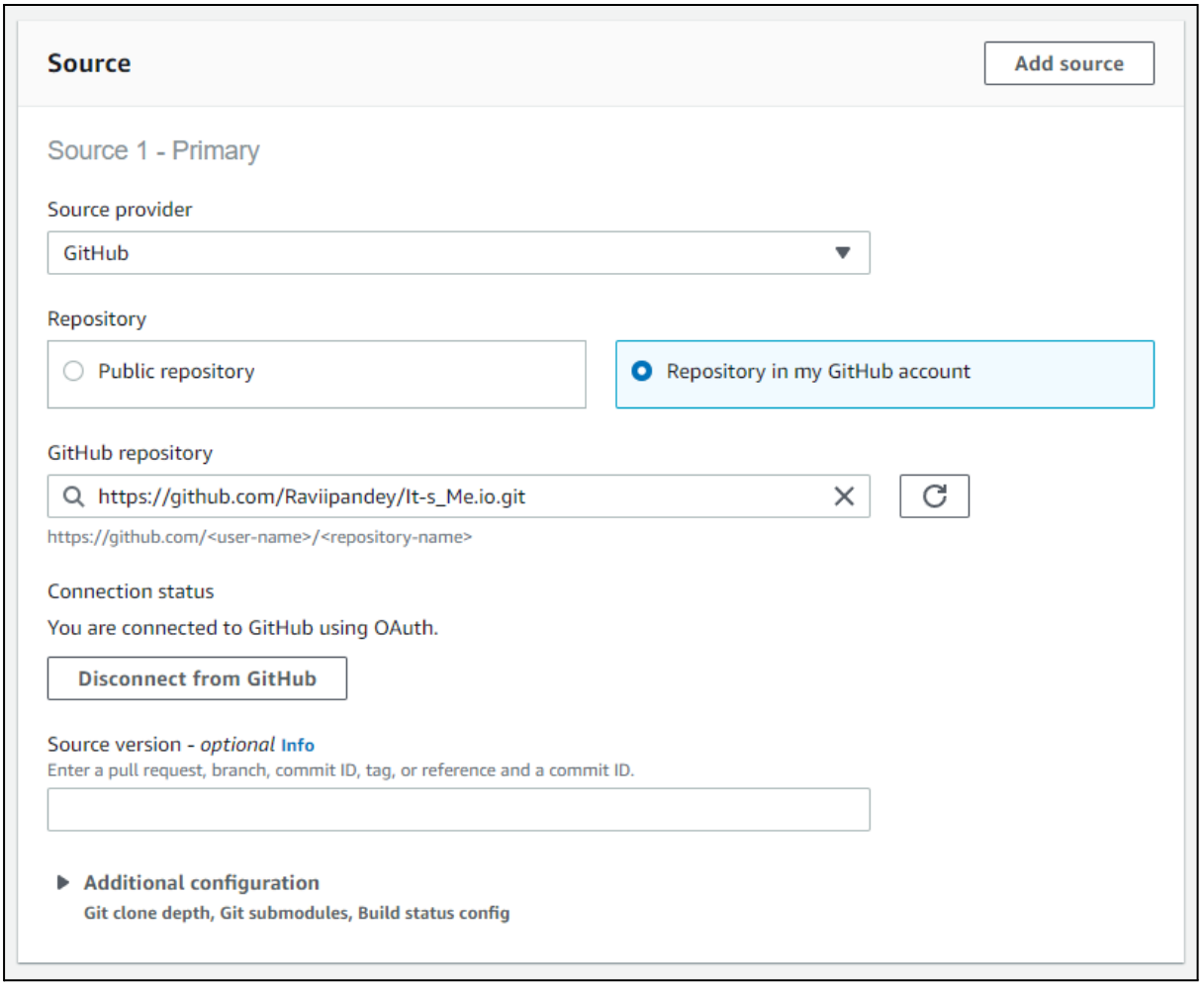
STEP 3:


Edit static website hosting Info
Static website hosting
Use this bucket to host a website or redirect requests. Learn more
Static website hosting
Disable
Enable
Hosting type
Host a static website
Use the bucket endpoint as the web address. Learn more
Redirect requests for an object
Redirect requests to another bucket or domain. Learn more
For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see
Using Amazon S3 Block Public Access
Index document
Specify the home or default page of the website.
index.html
Error document - optional
This is returned when an error occurs.
index.html

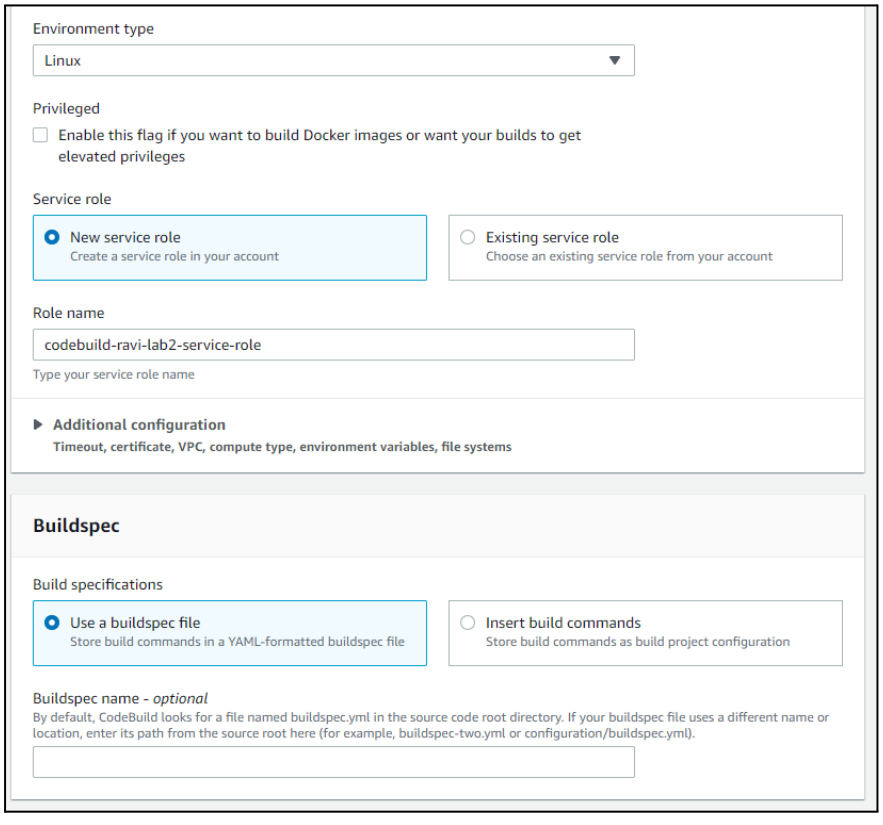
STEP 4:

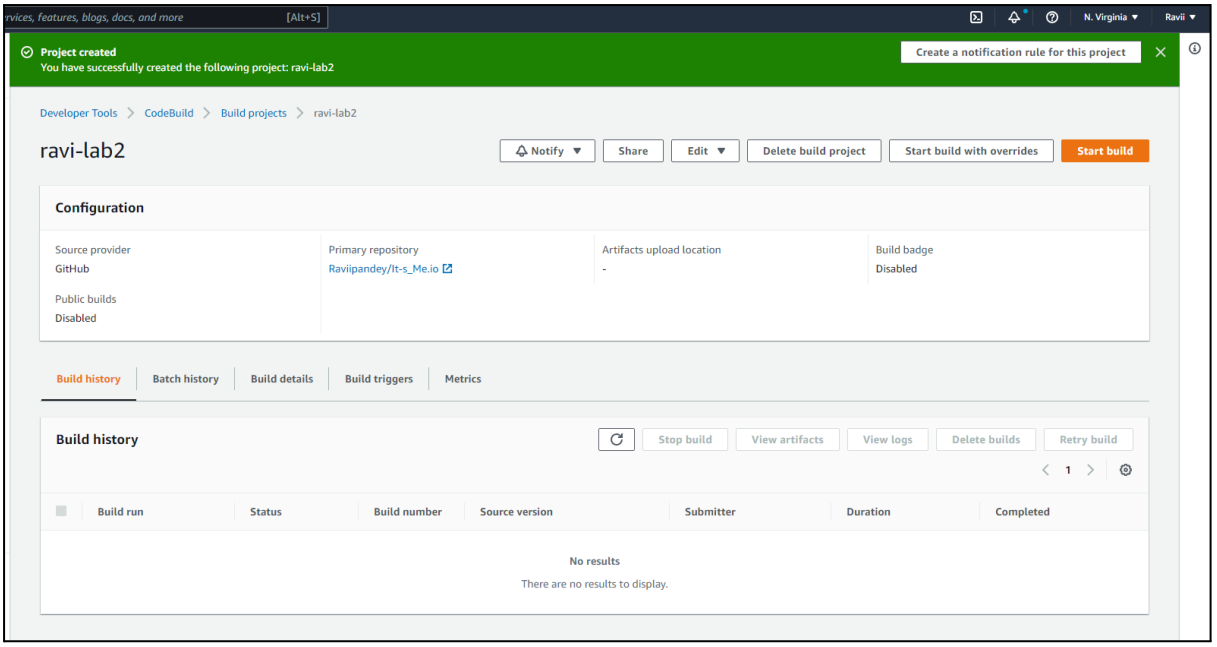

Create build project
Project configuration
Project name
ravi-lab2
A project name must be 2 to 255 characters. It can include the letters A-Z and a-z, the numbers 0-9, and the special characters - and _.
Description - optional
Build badge - optional
Enable build badge
Enable concurrent build limit - optional
Limit the number of allowed concurrent builds for this project.
Restrict number of concurrent builds this project can start
Additional configuration
tags

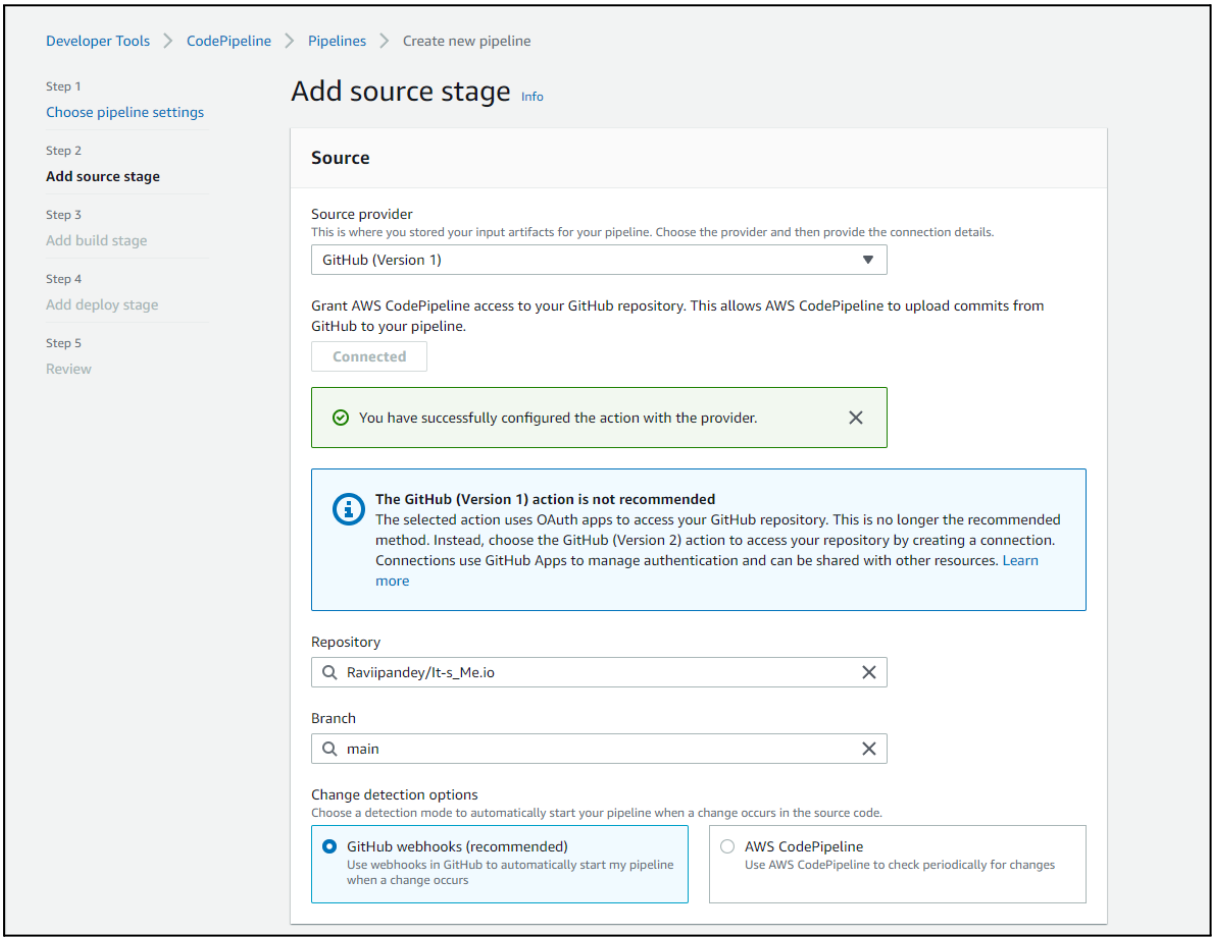
STEP 5


Source
Add source
Source 1 - Primary
Source provider
GitHub
Repository
Public repository
Repository in my GitHub account
GitHub repository
https://github.com/Raviipandey/It-s_Me.io.git
https://github.com/<user-name>/<repository-name>
Connection status
You are connected to GitHub using OAuth.
Disconnect from GitHub
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, Build status config

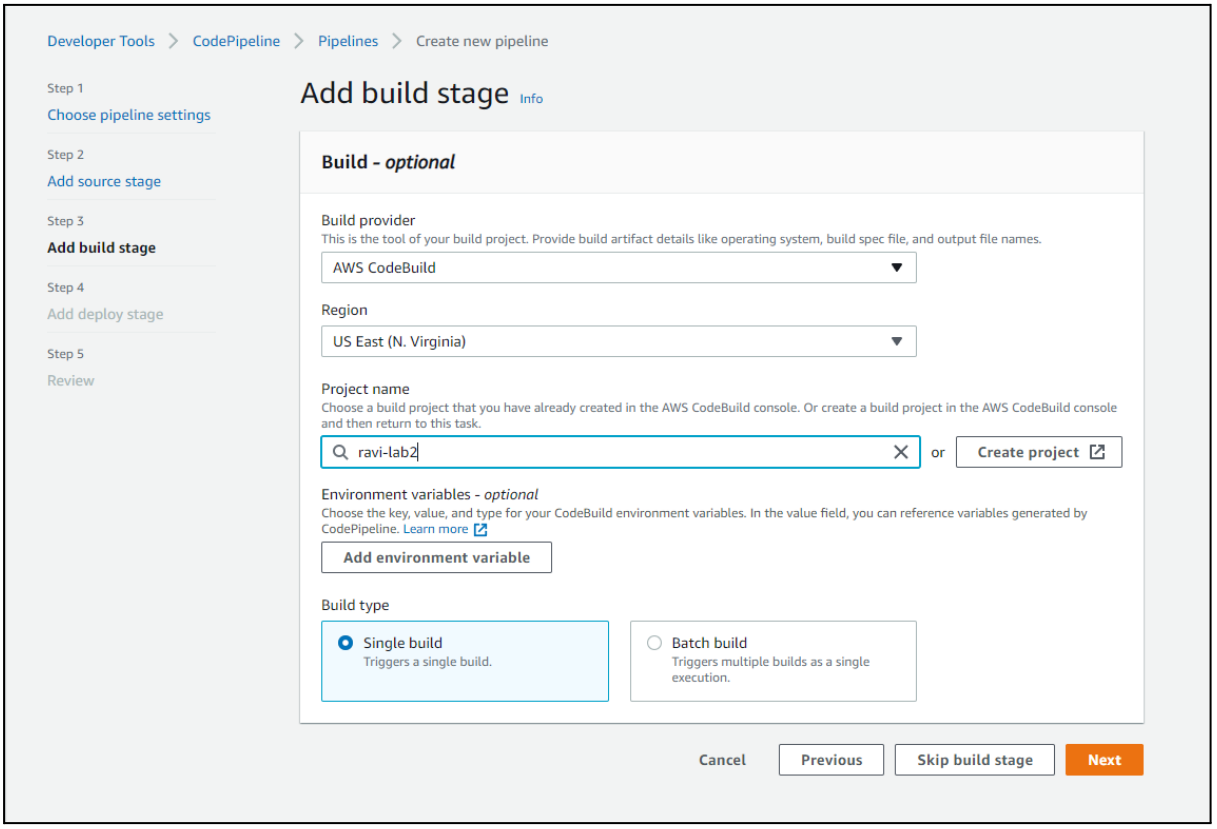
STEP 6:

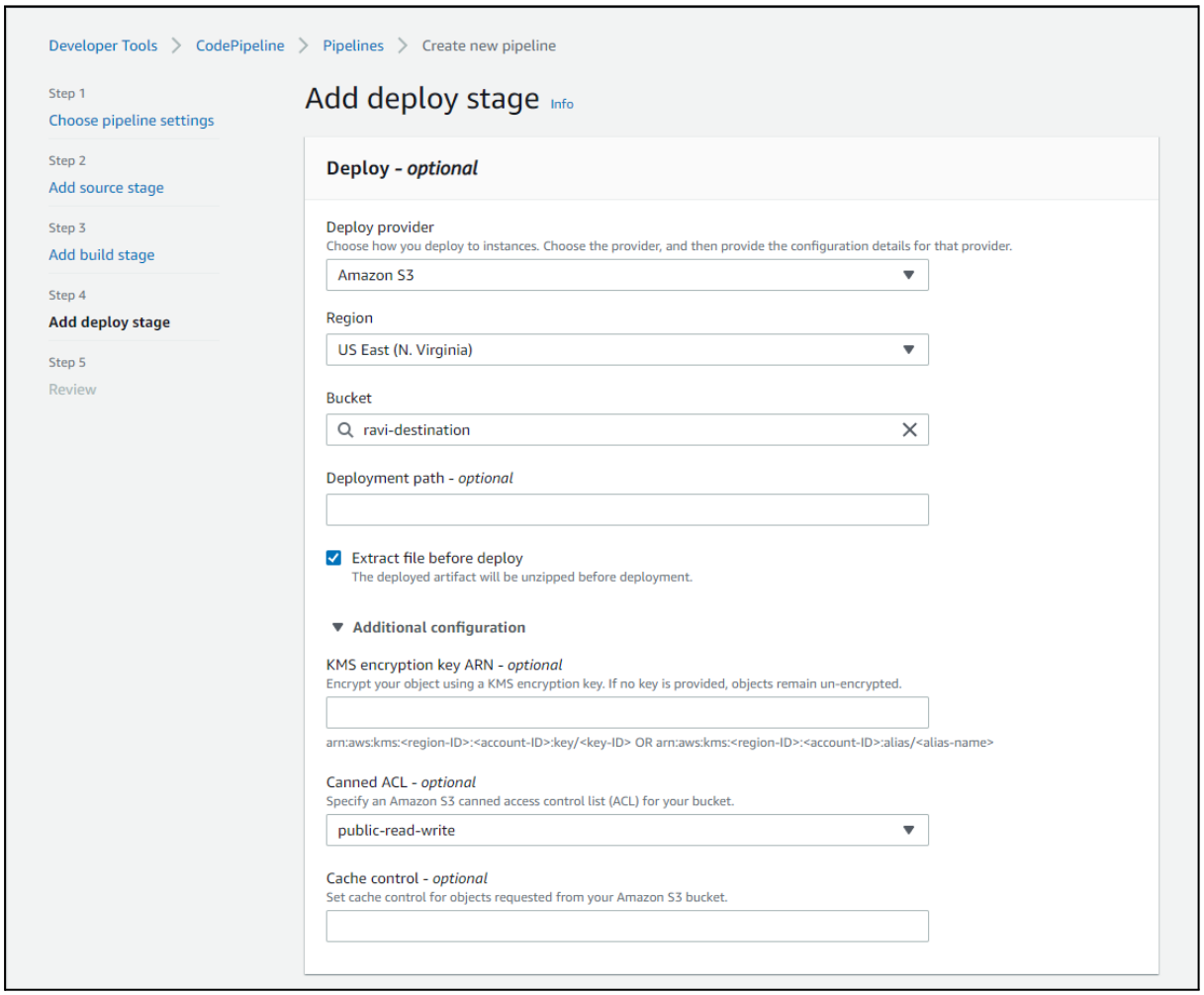

Environment type
Linux
Privileged
Enable this flag if you want to build Docker images or want your builds to get elevated privileges
Service role
New service role
Create a service role in your account
Existing service role
Choose an existing service role from your account
Role name
codebuild-ravi-lab2-service-role
Type your service role name
Additional configuration
Timeout, certificate, VPC, compute type, environment variables, file systems
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).


Project created
You have successfully created the following project: ravi-lab2
Create a notification rule for this project
Developer Tools > CodeBuild > Build projects > ravi-lab2
ravi-lab2
Notify
Share
Edit
Delete build project
Start build with overrides
Start build
Configuration
Source provider
GitHub
Primary repository
Raviipandey/It-s_Me.io
Artifacts upload location
-
Build badge
Disabled
Public builds
Disabled
Build history
Batch history
Build details
Build triggers
Metrics
Build history
Stop build
View artifacts
View logs
Delete builds
Retry build
Build run
Status
Build number
Source version
Submitter
Duration
Completed
No results
There are no results to display.

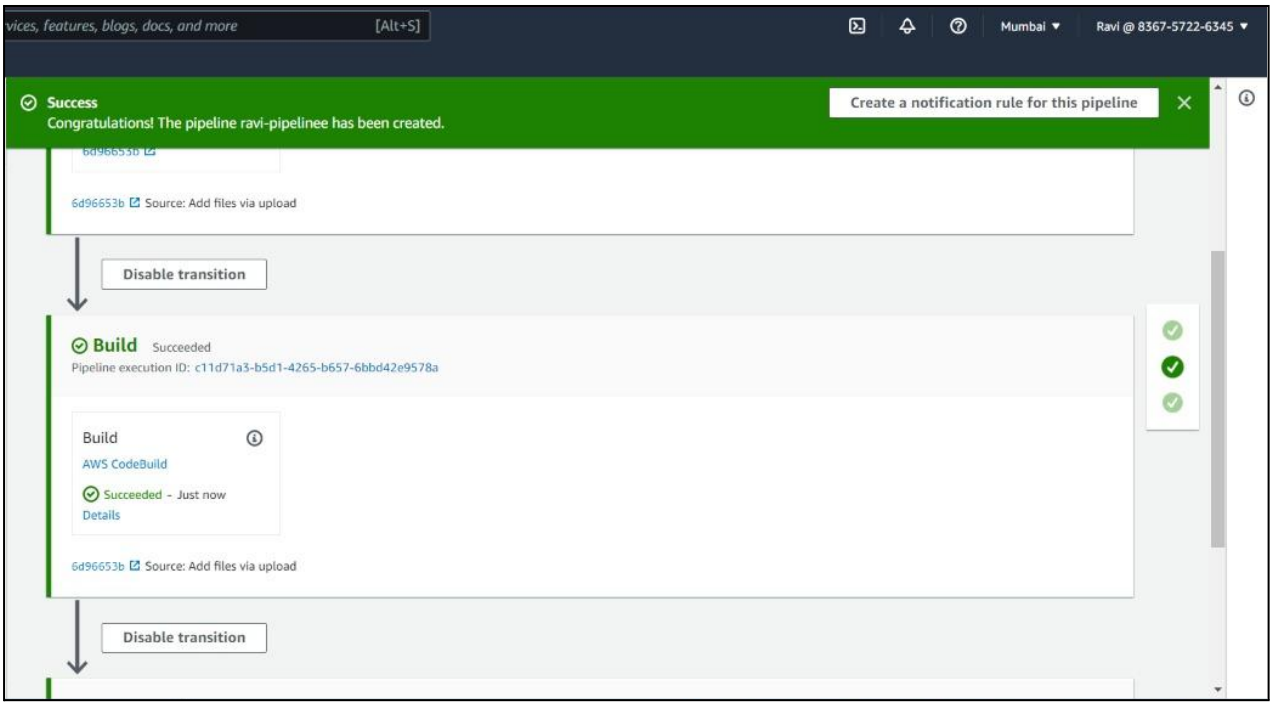

Developer Tools > CodePipeline > Pipelines > Create new pipeline
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Add source stage Info
Source
Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.
GitHub (Version 1)
Grant AWS CodePipeline access to your GitHub repository. This allows AWS CodePipeline to upload commits from GitHub to your pipeline.
Connected
You have successfully configured the action with the provider.
The GitHub (Version 1) action is not recommended
The selected action uses OAuth apps to access your GitHub repository. This is no longer the recommended method. Instead, choose the GitHub (Version 2) action to access your repository by creating a connection. Connections use GitHub Apps to manage authentication and can be shared with other resources. Learn more
Repository
Raviipandey/It-s_Me.io
Branch
main
Change detection options
Choose a detection mode to automatically start your pipeline when a change occurs in the source code.
GitHub webhooks (recommended)
Use webhooks in GitHub to automatically start my pipeline when a change occurs
AWS CodePipeline
Use AWS CodePipeline to check periodically for changes

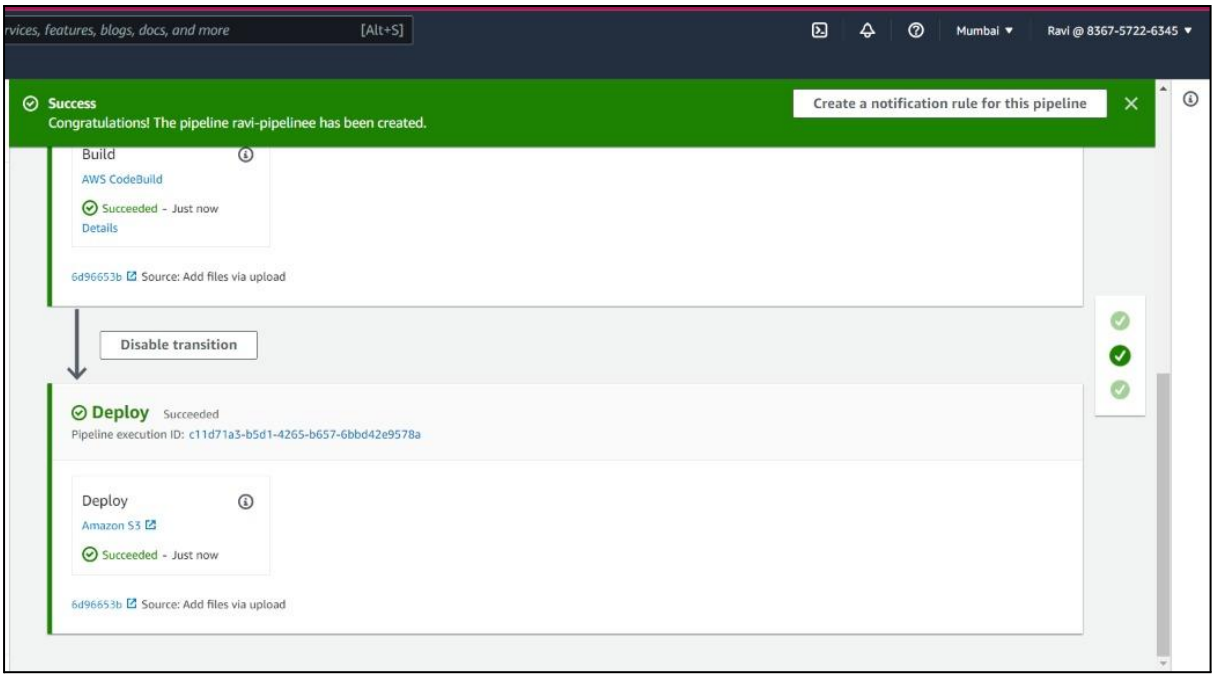
STEP 7:


Developer Tools > CodePipeline > Pipelines > Create new pipeline
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Add build stage Info
Build - optional
Build provider
This is the tool of your build project. Provide build artifact details like operating system, build spec file, and output file names.
AWS CodeBuild
Region
US East (N. Virginia)
Project name
Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.
ravi-lab2
or
Create project
Environment variables - optional
Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. Learn more
Add environment variable
Build type
Single build
Triggers a single build.
Batch build
Triggers multiple builds as a single execution.
Cancel
Previous
Skip build stage
Next


Developer Tools > CodePipeline > Pipelines > Create new pipeline
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Add deploy stage Info
Deploy - optional
Deploy provider
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.
Amazon S3
Region
US East (N. Virginia)
Bucket
ravi-destination
Deployment path - optional
Extract file before deploy
The deployed artifact will be unzipped before deployment.
Additional configuration
KMS encryption key ARN - optional
Encrypt your object using a KMS encryption key. If no key is provided, objects remain un-encrypted.
arn:aws:kms:<region-ID>:<account-ID>:key/<key-ID> OR arn:aws:kms:<region-ID>:<account-ID>:alias/<alias-name>
Canned ACL - optional
Specify an Amazon S3 canned access control list (ACL) for your bucket.
public-read-write
Cache control - optional
Set cache control for objects requested from your Amazon S3 bucket.

FINAL OUTPUT:





ADVANCE DEVOPS LAB

Name : Ravi Pandey

Roll No: 43

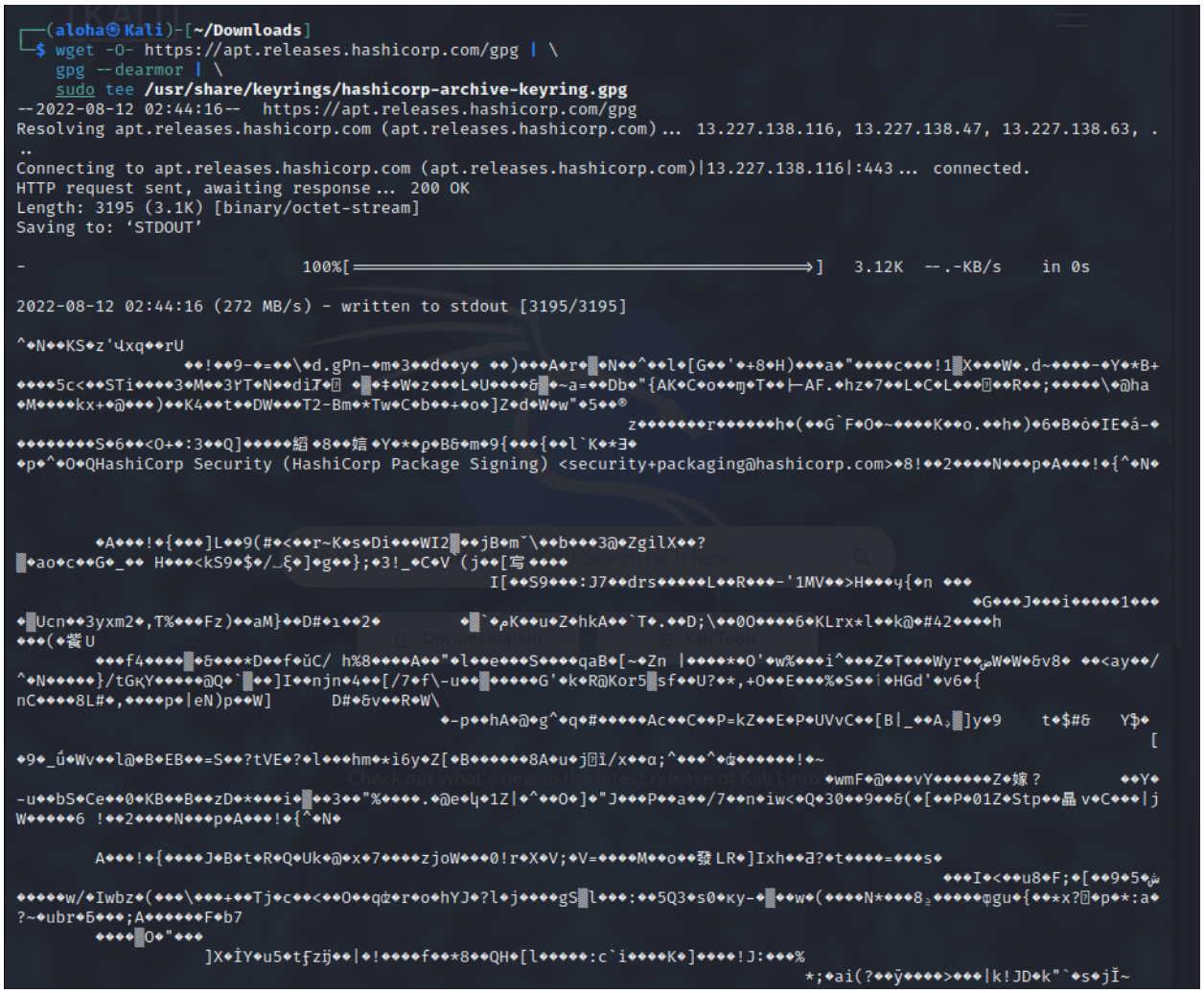
Exp No - 05

Aim:

To understand terraform lifecycle, core concepts/terminologies and install it on a Linux Machine

Output:

STEP 1 - Install the HashiCorp GPG key



STEP 2 - Unzip terraform

```
┌──(aloha㉿Kali)-[~/Downloads]
└─$ unzip terraform_1.2.7_linux_amd64.zip
Archive:  terraform_1.2.7_linux_amd64.zip
  inflating: terraform
```

STEP 3 - Verify the version

```
┌──(aloha㉿Kali)-[~/Downloads]
└─$ terraform -version
Terraform v1.2.7
on linux_amd64

┌──(aloha㉿Kali)-[~/Downloads]
└─$ terraform -h
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init          Prepare your working directory for other commands
  validate      Check whether the configuration is valid
  plan          Show changes required by the current configuration
  apply         Create or update infrastructure
  destroy       Destroy previously-created infrastructure

All other commands:
  console       Try Terraform expressions at an interactive command prompt
  fmt           Reformat your configuration in the standard style
  force-unlock  Release a stuck lock on the current workspace
  get           Install or upgrade remote Terraform modules
  graph         Generate a Graphviz graph of the steps in an operation
  import        Associate existing infrastructure with a Terraform resource
  login         Obtain and save credentials for a remote host
  logout        Remove locally-stored credentials for a remote host
  output        Show output values from your root module
  providers     Show the providers required for this configuration
  refresh       Update the state to match remote systems
  show          Show the current state or a saved plan
  state         Advanced state management
  taint         Mark a resource instance as not fully functional
  test          Experimental support for module integration testing
  untaint       Remove the 'tainted' state from a resource instance
  version       Show the current Terraform version
  workspace     Workspace management

Global options (use these before the subcommand, if any):
  -chdir=DIR    Switch to a different working directory before executing the
                given subcommand.
  -help         Show this help output, or the help for a specified subcommand.
  -version      An alias for the "version" subcommand.
```

ADVANCE DEVOPS LAB

Name : Ravi Pandey
Roll No: 43
Exp No - 06

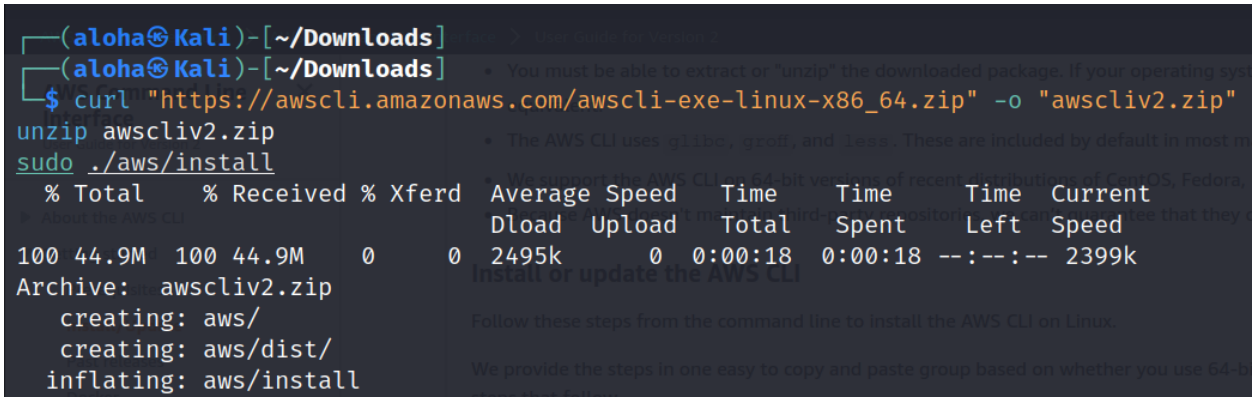
Aim:

To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform

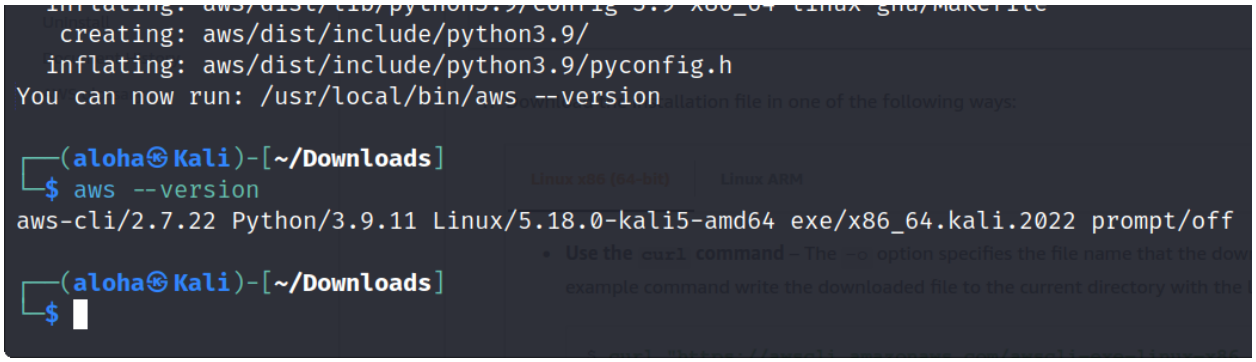
Output:

Building Infrastructure

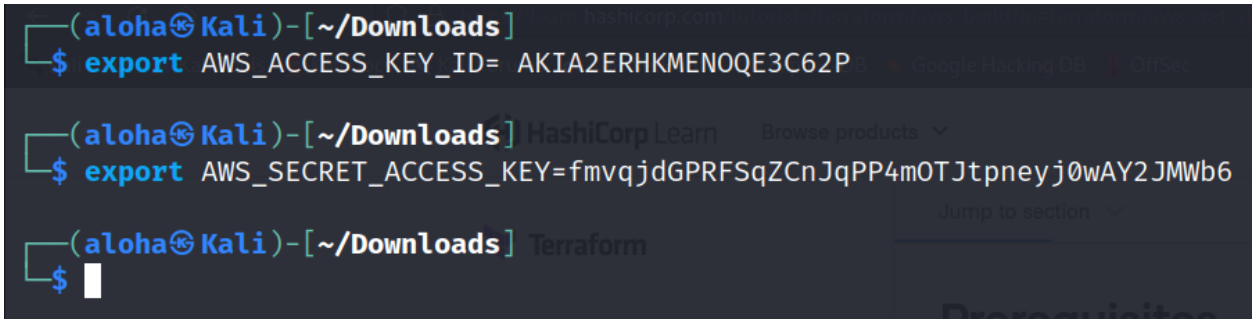
STEP 1 - Installing AWS CLI



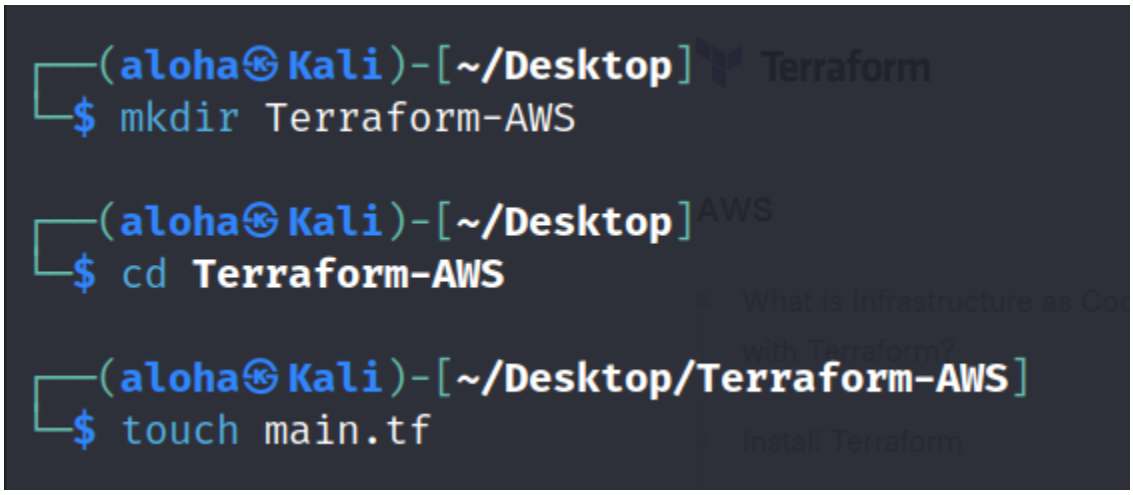
STEP 2 - Verification of AWS-CLI



STEP 3 - AWS Configure(Access and Secret key)



STEP 4 - Making Folders



STEP 5 - Configuration

```
┌──(aloha㉿Kali)-[~/Desktop/Terraform-AWS]
└─$ nano main.tf

┌──(aloha㉿Kali)-[~/Desktop/Terraform-AWS]
└─$ cat main.tf
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "⇝ 4.16"
    }
  }

  required_version = "⩾ 1.2.0"
}

provider "aws" {
  region  = "us-west-2"
}

resource "aws_instance" "app_server" {
  ami           = "ami-830c94e3"
  instance_type = "t2.micro"

  tags = {
    Name = "ExampleAppServerInstance"
  }
}
```

STEP 6 - Terraform init

```
┌──(aloha㉿Kali)-[~/Desktop/Terraform-AWS]
└─$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 4.16"...
- Installing hashicorp/aws v4.25.0...
- Installed hashicorp/aws v4.25.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

STEP 7- Format and Validation

```
┌──(aloha㉿Kali)-[~/Desktop/Terraform-AWS]
└─$ terraform fmt
main.tf

┌──(aloha㉿Kali)-[~/Desktop/Terraform-AWS]
└─$ terraform validate
Success! The configuration is valid.
```

STEP 8 - Terraform Apply

```
Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.app_server: Creating...
aws_instance.app_server: Still creating... [10s elapsed]
aws_instance.app_server: Still creating... [20s elapsed]
aws_instance.app_server: Still creating... [30s elapsed]
aws_instance.app_server: Still creating... [40s elapsed]
aws_instance.app_server: Creation complete after 46s [id=i-0c85527a94c57b64a]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

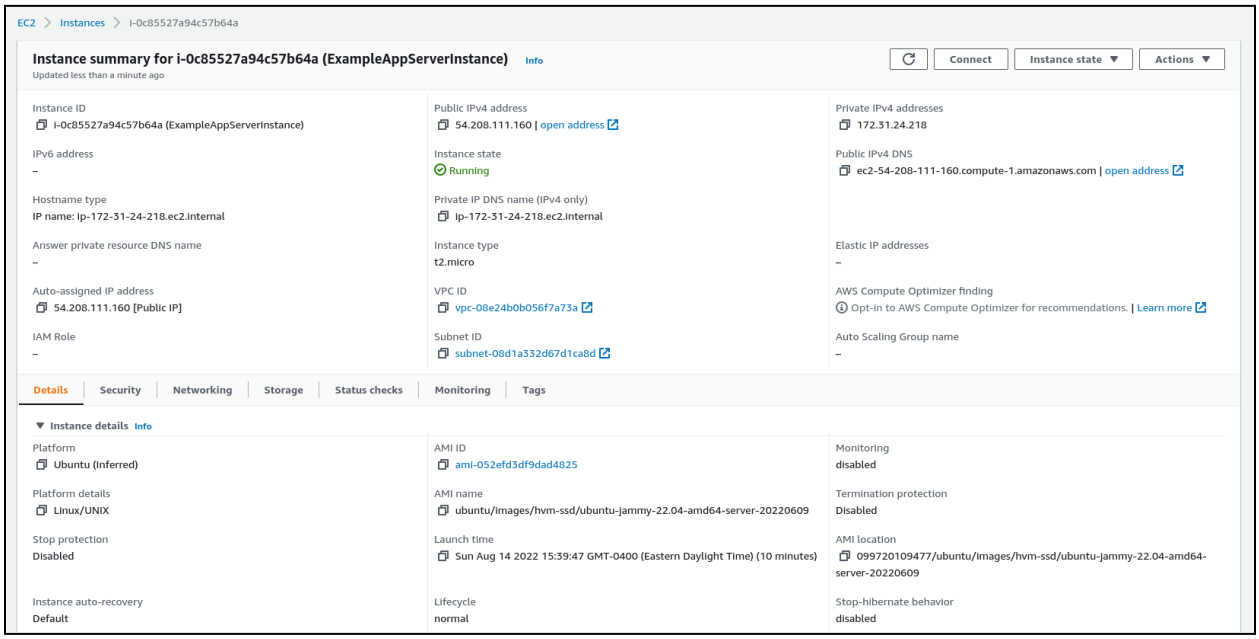
STEP 9 - Terraform Show

```
┌──(aloha㉿Kali)-[~/Desktop/Terraform-AWS]
└─$ terraform show
# aws_instance.app_server:
resource "aws_instance" "app_server" {
    ami                                  = "ami-052efd3df9dad4825"
    arn                                  = "arn:aws:ec2:us-east-1:696940912922:instance/i-0c85527a94c57b64a"
    associate_public_ip_address          = true
    availability_zone                    = "us-east-1a"
    cpu_core_count                       = 1
    cpu_threads_per_core                 = 1
    disable_api_stop                     = false
    disable_api_termination              = false
    ebs_optimized                        = false
    get_password_data                    = false
    hibernation                          = false
    id                                   = "i-0c85527a94c57b64a"
    instance_initiated_shutdown_behavior = "stop"
    instance_state                       = "running"
    instance_type                        = "t2.micro"
    ipv6_address_count                   = 0
    ipv6_addresses                       = []
    monitoring                           = false
    primary_network_interface_id         = "eni-0fd411e5af6a5f488"
    private_dns                          = "ip-172-31-24-218.ec2.internal"
    private_ip                           = "172.31.24.218"
    public_dns                           = "ec2-54-208-111-160.compute-1.amazonaws.com"
    public_ip                            = "54.208.111.160"
    secondary_private_ips                = []
    security_groups                      = [
        "default",
    ]
    source_dest_check                    = true
    subnet_id                            = "subnet-08d1a332d67d1ca8d"
    tags                                 = {
        "Name" = "ExampleAppServerInstance"
    }
    tags_all                             = {
        "Name" = "ExampleAppServerInstance"
    }
    tenancy                              = "default"
    user_data_replace_on_change          = false
    vpc_security_group_ids               = [
        "sg-046fc22136f55a9c4",
    ]

    capacity_reservation_specification {
        capacity_reservation_preference = "open"
    }

    credit_specification {
        cpu_credits = "standard"
    }
```

STEP 10 - Terraform State

```
┌──(aloha㉿Kali)-[~/Desktop/Terraform-AWS]
└─$ terraform state list
aws_instance.app_server
```

STEP 11 - Checking Instance in AWS



Changing Infrastructure

STEP 1 - Change AMI number according to Selected region in main.tf

```
resource "aws_instance" "app_server" {
  ami           = "ami-830c94e3"
  instance_type = "t2.micro"
```

AMI ID

ami-052efd3df9dad4825

STEP 2 - Terraform Apply

```
  # aws_instance.app_server must be replaced
-/+ resource "aws_instance" "app_server" {
      ~ ami                                  = "ami-830c94e3" -> "ami-08d70e59c07c61a3a" # forces replacement
      ~ arn                                  = "arn:aws:ec2:us-west-2:696940912922:instance/i-0a3ceb8ea6695c613" ->
(known after apply)
```

```
Plan: 1 to add, 0 to change, 1 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.app_server: Destroying... [id=i-0a3ceb8ea6695c613]
aws_instance.app_server: Still destroying... [id=i-0a3ceb8ea6695c613, 10s elapsed]
aws_instance.app_server: Still destroying... [id=i-0a3ceb8ea6695c613, 20s elapsed]
aws_instance.app_server: Still destroying... [id=i-0a3ceb8ea6695c613, 30s elapsed]
aws_instance.app_server: Destruction complete after 34s
aws_instance.app_server: Creating...
aws_instance.app_server: Still creating... [10s elapsed]
aws_instance.app_server: Still creating... [20s elapsed]
aws_instance.app_server: Still creating... [30s elapsed]
aws_instance.app_server: Still creating... [40s elapsed]
aws_instance.app_server: Still creating... [50s elapsed]
aws_instance.app_server: Still creating... [1m0s elapsed]
aws_instance.app_server: Still creating... [1m10s elapsed]
aws_instance.app_server: Still creating... [1m20s elapsed]
aws_instance.app_server: Creation complete after 1m25s [id=i-04d3a3aad27e610f6]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

Destroying Infrastructure

```
┌──(aloha㉿Kali)-[~/Desktop/Terraform-AWS]
└──(aloha㉿Kali)-[~/Desktop/Terraform-AWS]
└─$ terraform destroy
aws_instance.app_server: Refreshing state... [id=i-04d3a3aad27e610f6]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with
the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_instance.app_server will be destroyed
  - resource "aws_instance" "app_server" {
      - ami                                  = "ami-08d70e59c07c61a3a" → null
      - arn                                  = "arn:aws:ec2:us-west-2:696940912922:instance/i-04d3a3aad27e610f6" →
null
```

```
Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_instance.app_server: Destroying... [id=i-04d3a3aad27e610f6]
aws_instance.app_server: Still destroying... [id=i-04d3a3aad27e610f6, 10s elapsed]
aws_instance.app_server: Still destroying... [id=i-04d3a3aad27e610f6, 20s elapsed]
aws_instance.app_server: Still destroying... [id=i-04d3a3aad27e610f6, 30s elapsed]
aws_instance.app_server: Destruction complete after 33s

Destroy complete! Resources: 1 destroyed.
```

ADVANCED DEVOPS LAB

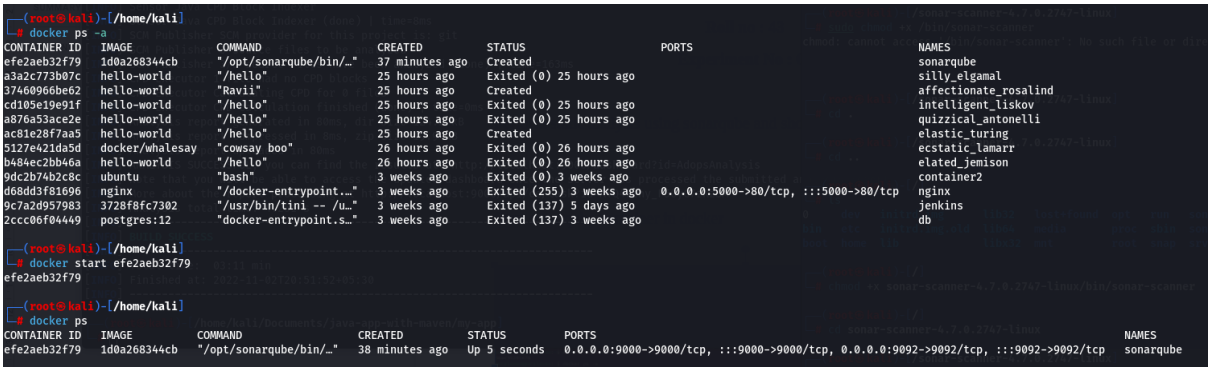
Name : Ravi Pandey

Roll no : 43

Experiment No : 08

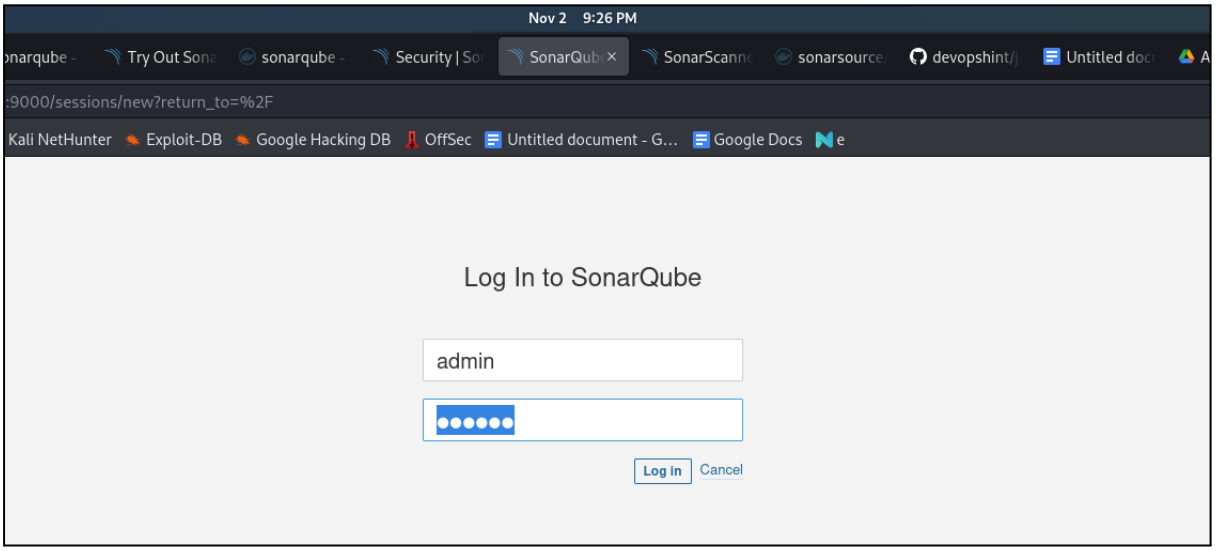
Aim: Perform static analysis using sonarqube and show the analysis

Output: Start sonarqube container in docker



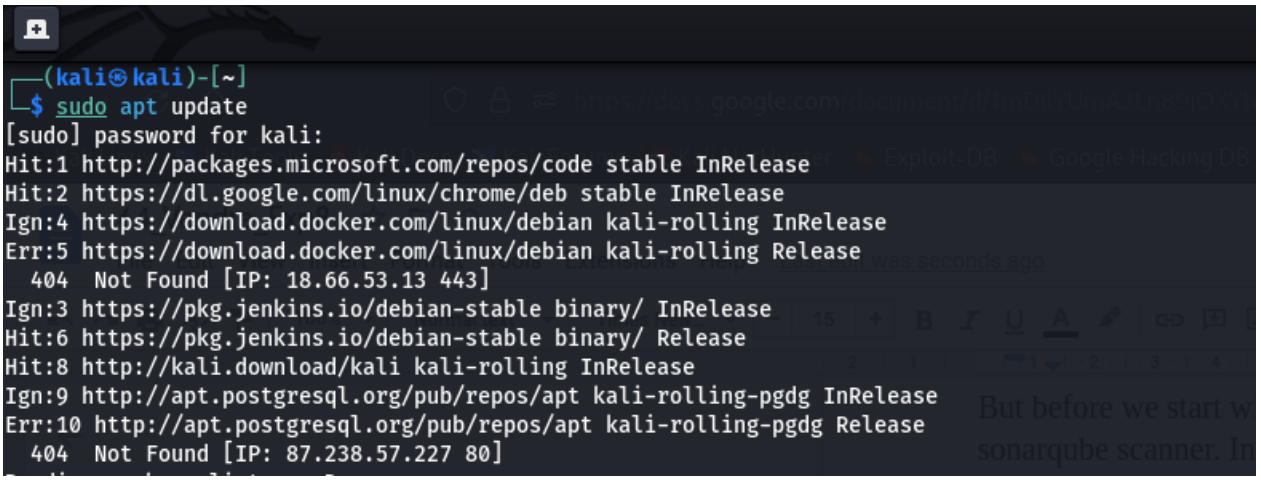
```
┌──(root㉿kali)-[/home/kali]
└─# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS                      PORTS                                       NAMES
efe2aeb32f79   1d0a268344cb   "/opt/sonarqube/bin/…"   37 minutes ago   Created                                                         sonarqube
a3a2c773b07c   hello-world    "/hello"                 25 hours ago     Exited (0) 25 hours ago                                         silly_elgamal
37460966be62   hello-world    "Ravii"                  25 hours ago     Created                                                         affectionate_rosalind
cd105e19e91f   hello-world    "/hello"                 25 hours ago     Exited (0) 25 hours ago                                         intelligent_liskov
a876a53ace2e   hello-world    "/hello"                 25 hours ago     Exited (0) 25 hours ago                                         quizzical_antonelli
ac81e28f7aa5   hello-world    "/hello"                 25 hours ago     Created                                                         elastic_turing
5127e421da5d   docker/whalesay   "cowsay boo"          26 hours ago     Exited (0) 26 hours ago                                         ecstatic_lamarr
b484ec2bb46a   hello-world    "/hello"                 26 hours ago     Exited (0) 26 hours ago                                         elated_jemison
9dc2b74b2c8c   ubuntu         "bash"                   3 weeks ago      Exited (0) 3 weeks ago                                          container2
d68dd3f81696   nginx          "/docker-entrypoint.…"   3 weeks ago      Exited (255) 3 weeks ago    0.0.0.0:5000->80/tcp, :::5000->80/tcp   nginx
9c7a2d957983   3728f8fc7302   "/usr/bin/tini -- /u…"   3 weeks ago      Exited (137) 5 days ago                                         jenkins
2ccc06f04449   postgres:12    "docker-entrypoint.s…"   3 weeks ago      Exited (137) 3 weeks ago                                        db

┌──(root㉿kali)-[/home/kali]
└─# docker start efe2aeb32f79
efe2aeb32f79

┌──(root㉿kali)-[/home/kali]
└─# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS         PORTS                                                                                                 NAMES
efe2aeb32f79   1d0a268344cb   "/opt/sonarqube/bin/…"   38 minutes ago   Up 5 seconds   0.0.0.0:9000->9000/tcp, :::9000->9000/tcp, 0.0.0.0:9092->9092/tcp, :::9092->9092/tcp   sonarqube
```

After the service is up, write in the browser `localhost:9000` and you will see the following

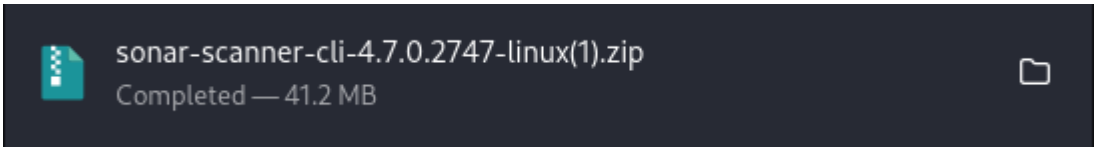


But before we start with the static code analysis, we need to download the sonarqube scanner. In order to install sonarqube we need to follow the steps given below:

1. sudo apt–get update

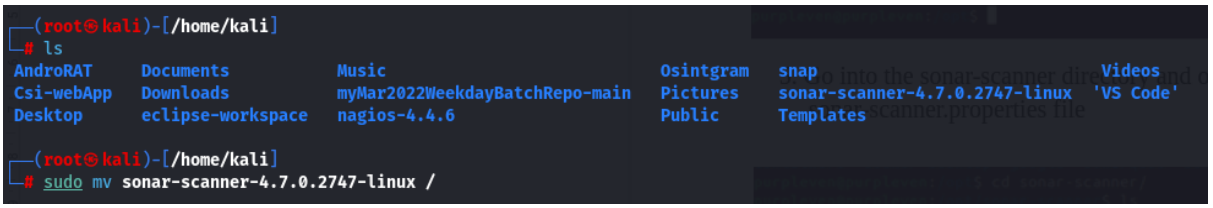


2. Install sonar scanner from the official link https://docs.sonarqube.org/latest/analysis/scan/sonarscanner/

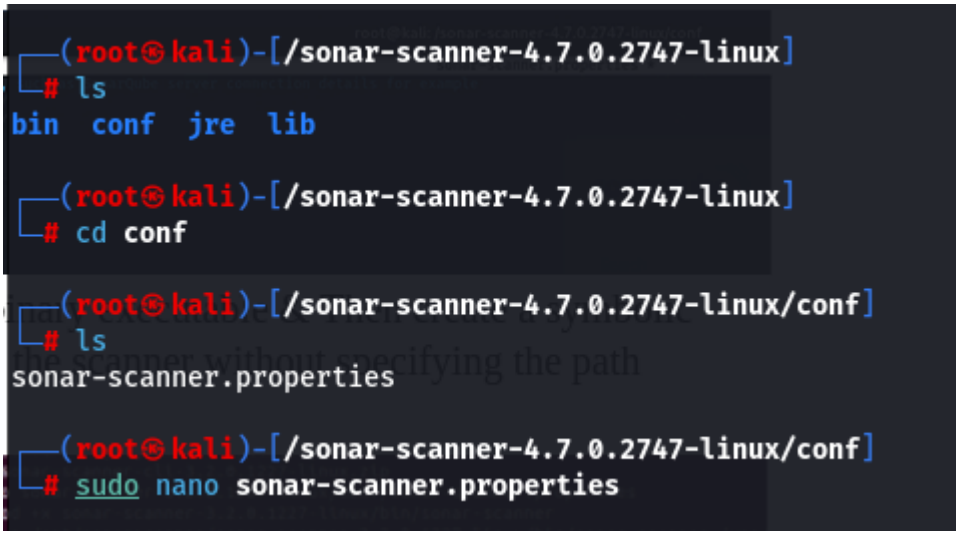


3. Unzip the installed zip file

4. Move the installed file into sonar-scanner in opt directory where sonarqube is present



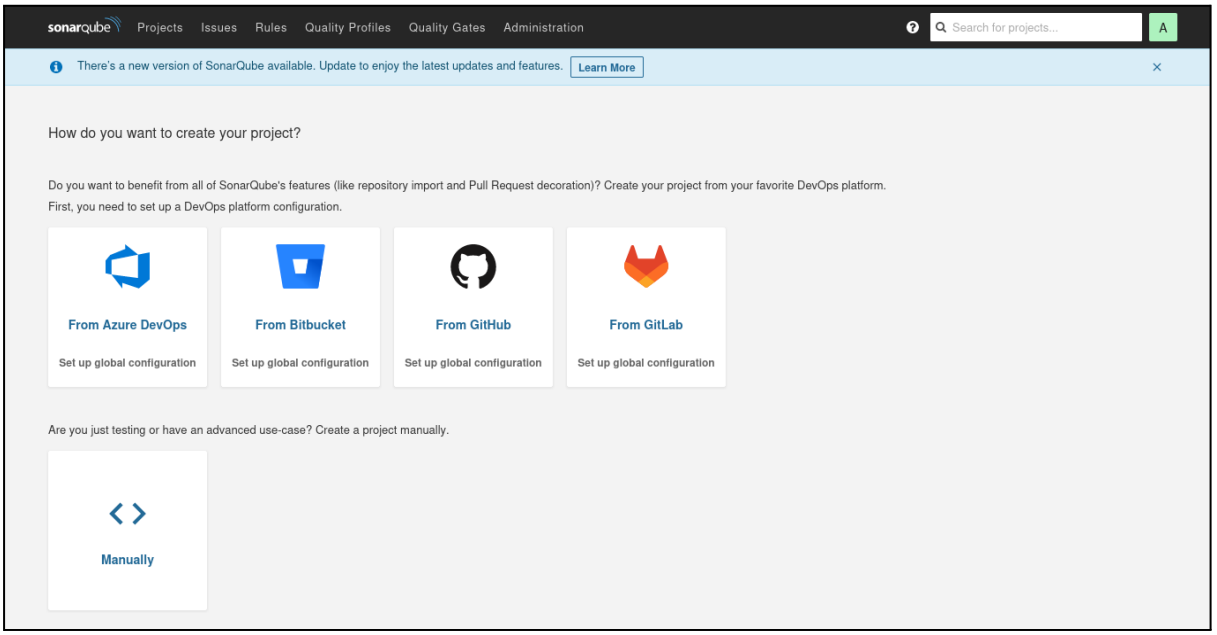
5. Go into the sonar-scanner directory and open the sonar-scanner.properties file



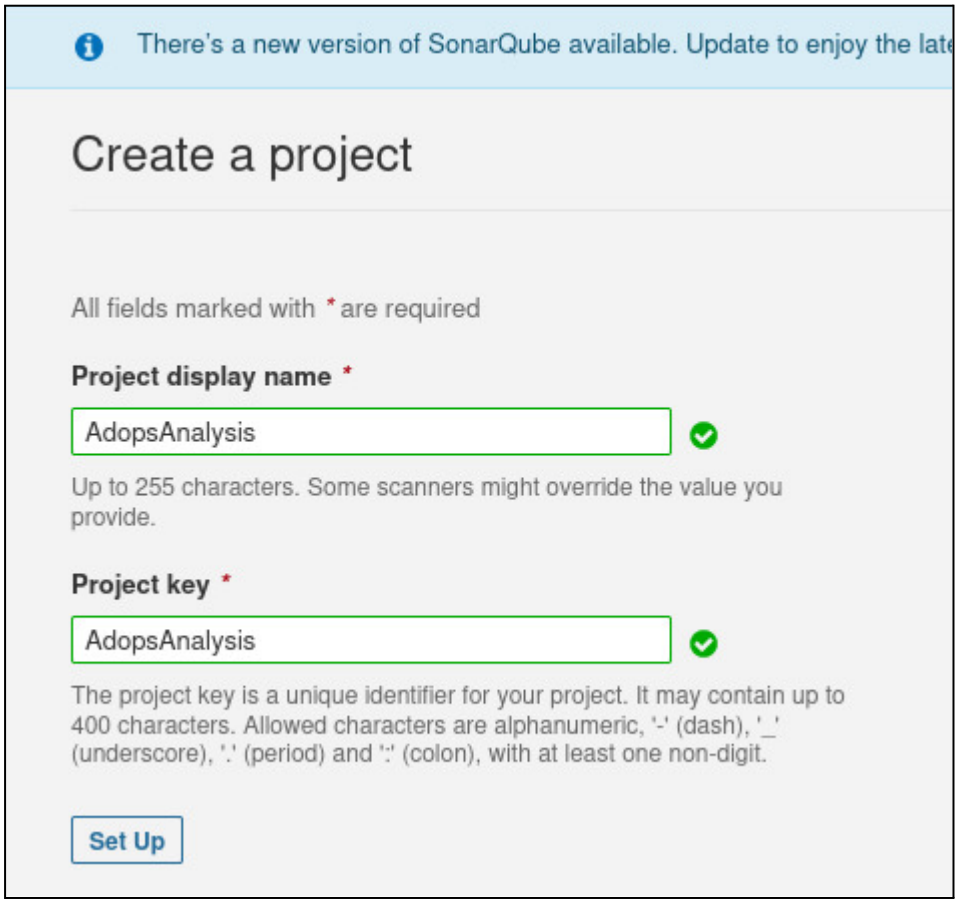
6. Add the localhost and encoding

7. Now make the scanner binary executable & Then create a symbolic link so that you can call the scanner without specifying the path

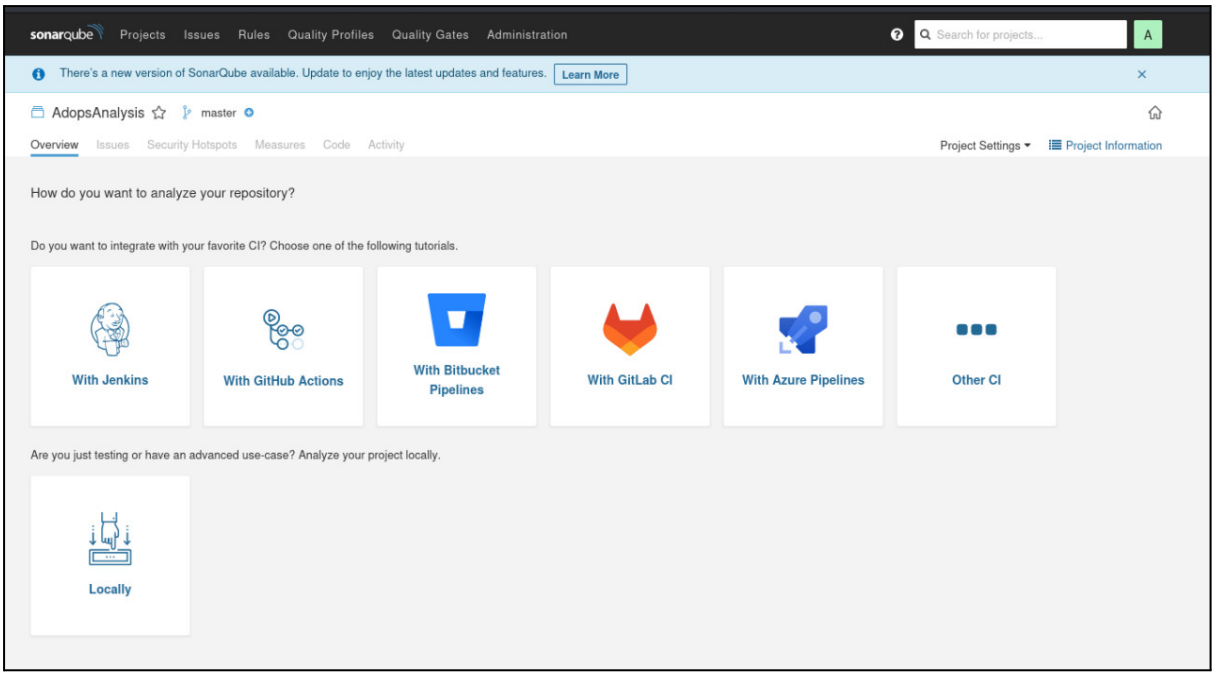
Now that our sonar-scanner is installed, we will create a project in our sonarqube that is logged in through http://localhost:9000/projects/create



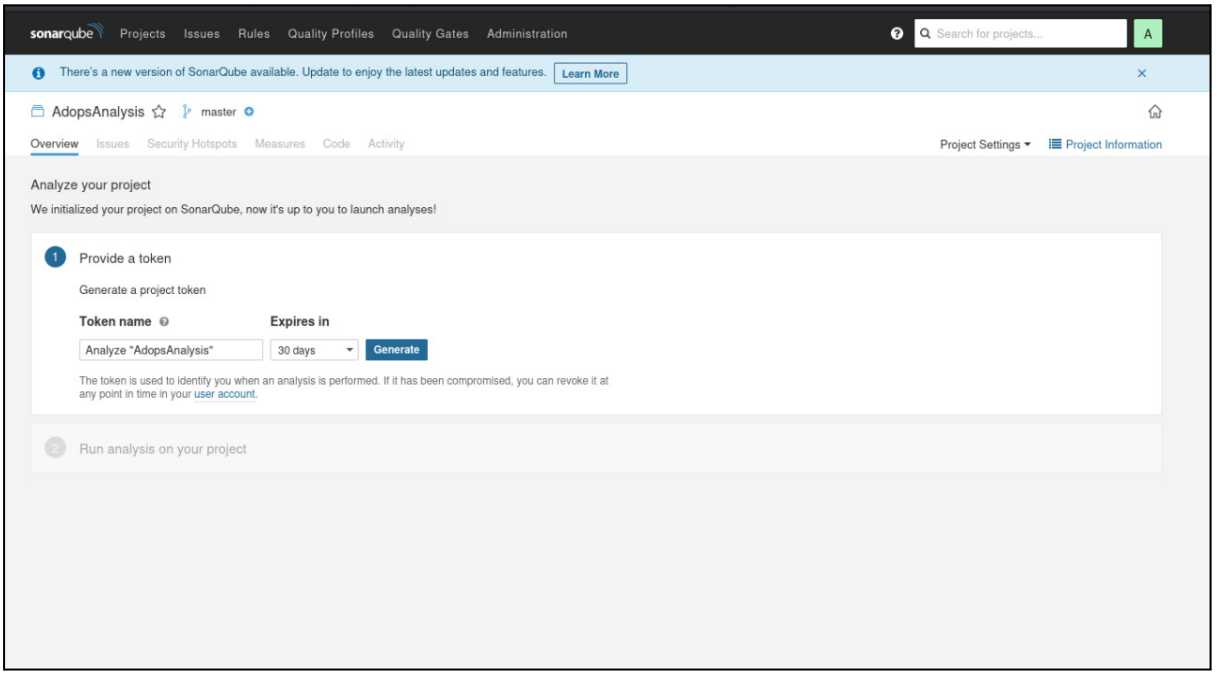
Click on manually and create the project by giving in the values for the asked fields



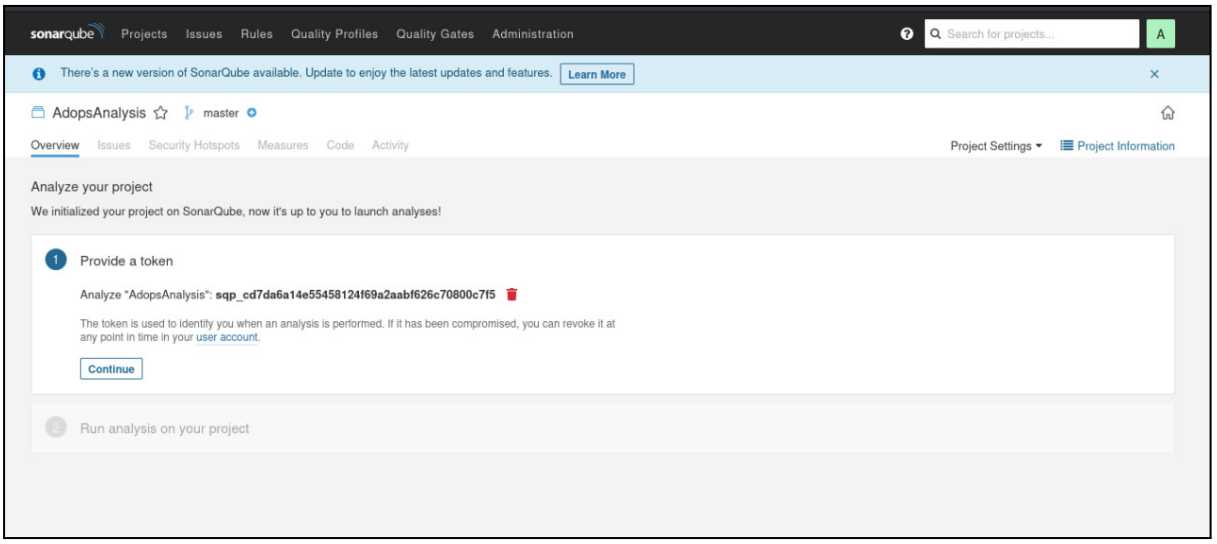
It will look something like this after clicking set up, now in order to analyze the code, click on locally



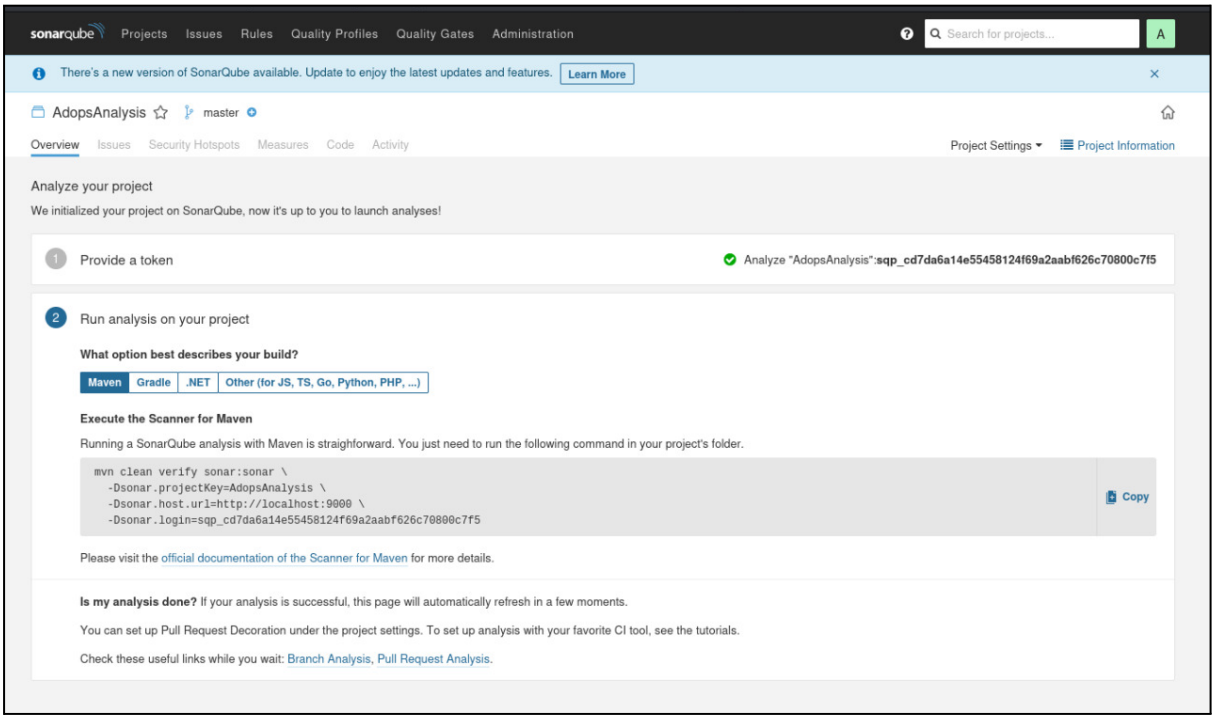
Click on generate a token



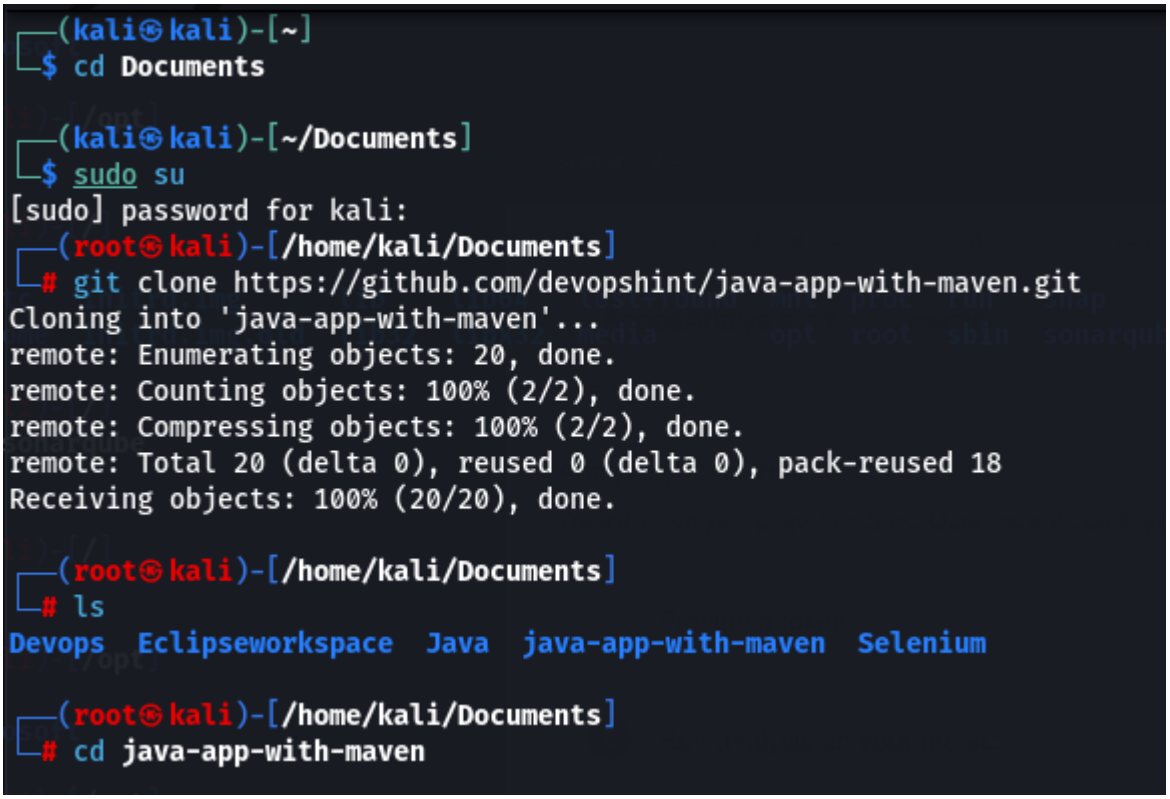
The following output will be received

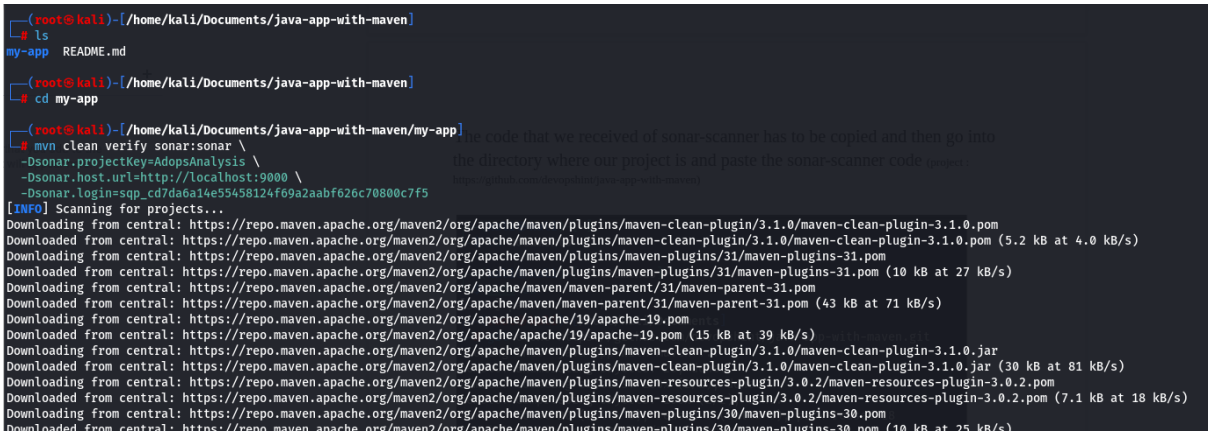


After the token is copied and clicked on continue, we need to enter the following



The code that we received of sonar-scanner has to be copied and then go into the directory where our project is and paste the sonar-scanner code (project : https://github.com/devopshint/java-app-with-maven)

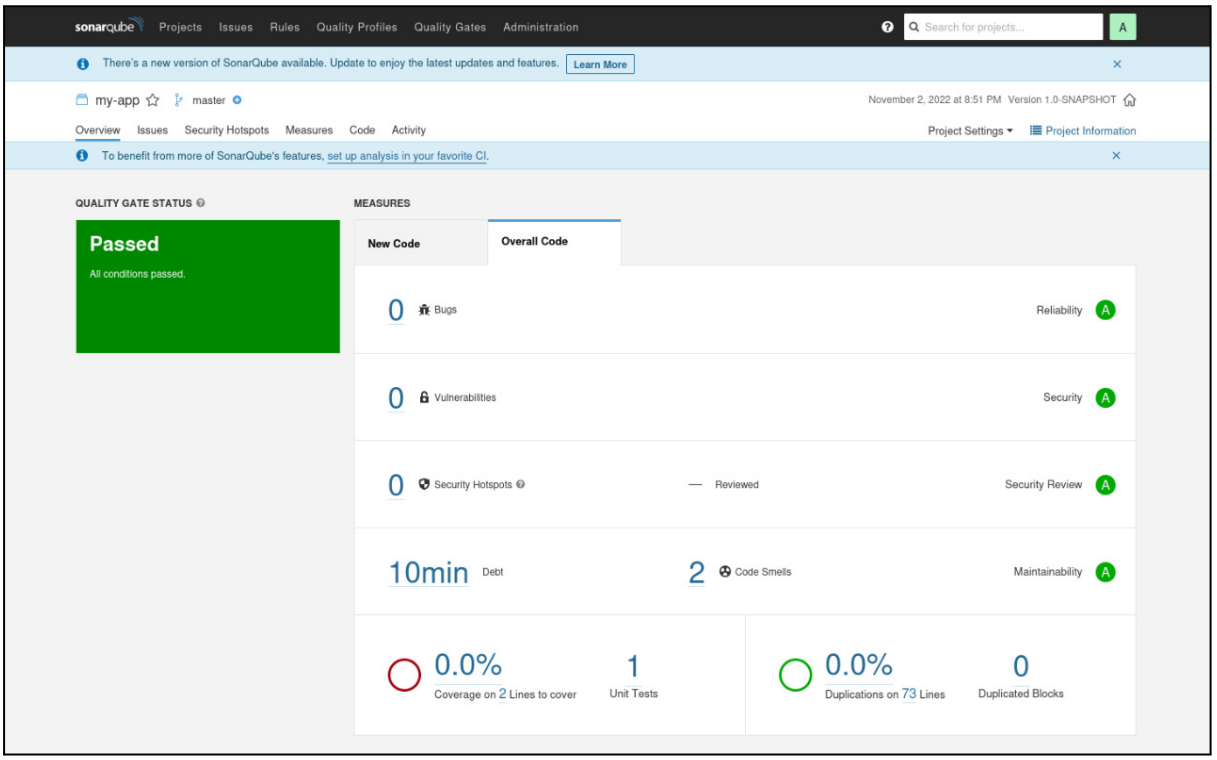


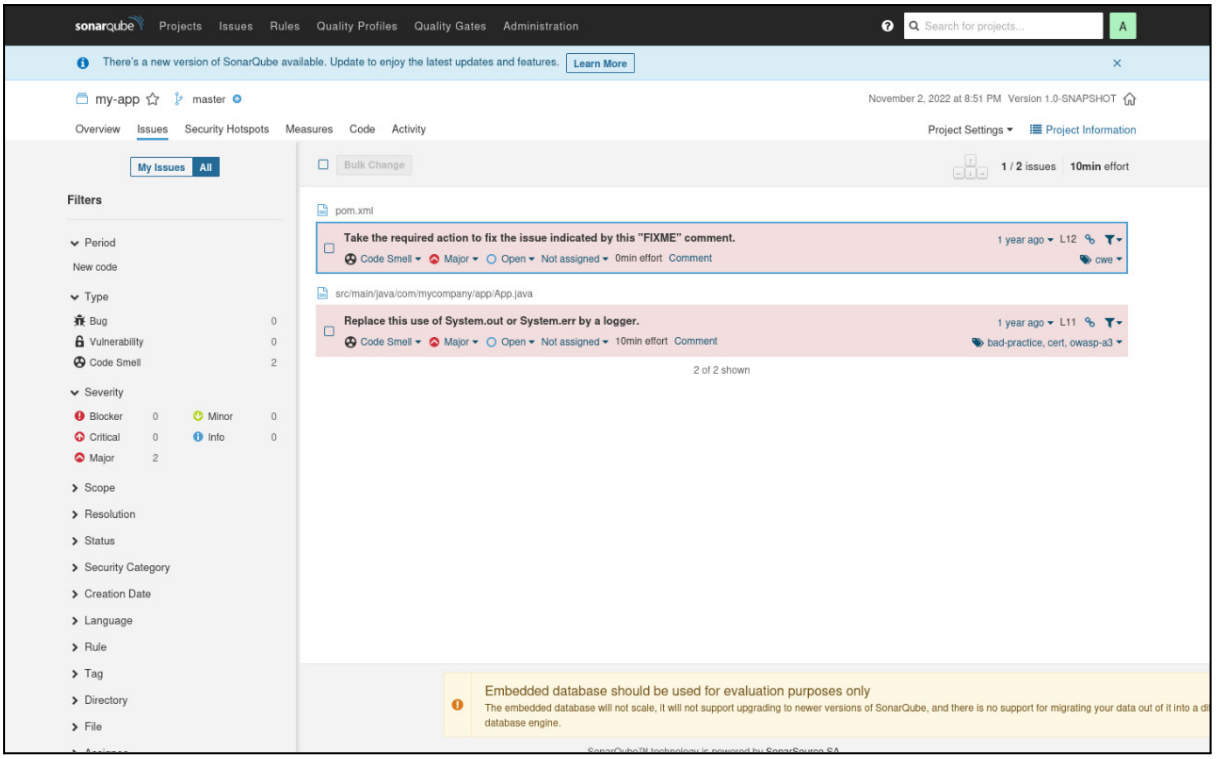


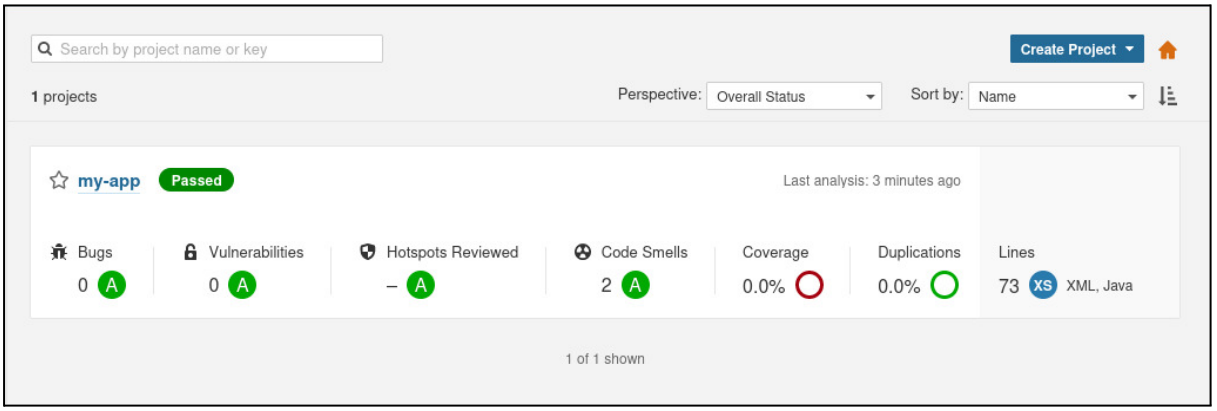
After it is run, the output is successful

```
[INFO] Sensor VB.NET Properties [vbnet] (done) | time=0ms
[INFO] ------------ Run sensors on project
[INFO] Sensor Analysis Warnings import [csharp]
[INFO] Sensor Analysis Warnings import [csharp] (done) | time=1ms
[INFO] Sensor Zero Coverage Sensor
[INFO] Sensor Zero Coverage Sensor (done) | time=6ms
[INFO] Sensor Java CPD Block Indexer
[INFO] Sensor Java CPD Block Indexer (done) | time=8ms
[INFO] SCM Publisher SCM provider for this project is: git
[INFO] SCM Publisher 3 source files to be analyzed
[INFO] SCM Publisher 3/3 source files have been analyzed (done) | time=163ms
[INFO] CPD Executor 1 file had no CPD blocks
[INFO] CPD Executor Calculating CPD for 0 files
[INFO] CPD Executor CPD calculation finished (done) | time=0ms
[INFO] Analysis report generated in 80ms, dir size=124.2 kB
[INFO] Analysis report compressed in 8ms, zip size=19.9 kB
[INFO] Analysis report uploaded in 80ms
[INFO] ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=AdopsAnalysis
[INFO] Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
[INFO] More about the report processing at http://localhost:9000/api/ce/task?id=AYQ47zny_F50yC7zKzSM
[INFO] Analysis total time: 4.251 s
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  03:11 min
[INFO] Finished at: 2022-11-02T20:51:52+05:30
[INFO] ------------------------------------------------------------------------
```

Now if localhost:9000 is opened, we will see the following







Conclusion:

The analysis is performed and displayed in sonarqube successfully

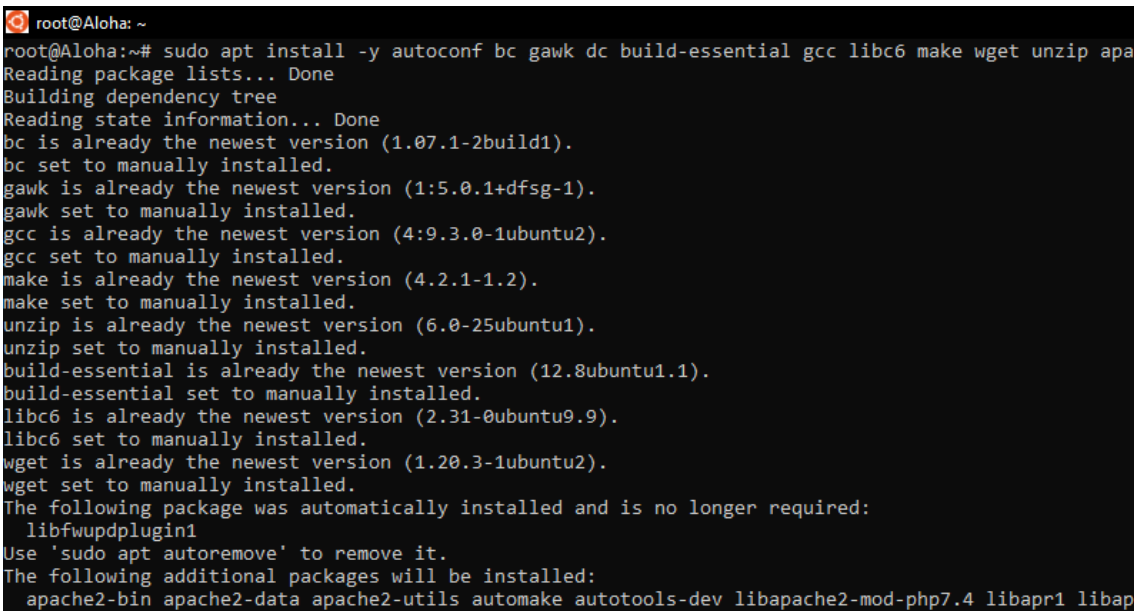
ADVANCE DEVOPS LAB
Name : Ravi Pandey
Roll no: 43
Exp - 09

Aim : To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

Theory : Continuous monitoring is a process of constant detecting, reporting, and responding to risks and events within an IT system.
This process is a vital DevOps security practice and has multiple goals:
● Provide real-time insight into system performance.
● Offer feedback on the overall health and security of IT infrastructure.
● Enhance visibility across IT operations and the DevOps pipeline.
● Identify the cause of incidents and apply mitigation before the problem results in downtime or a data breach

Output :

STEP 1 : Install Prerequisite Packages



STEP 2 : Install Nagios core

```
root@Aloha:~/nagios# ls
nagios-4.4.6.tar.gz
root@Aloha:~/nagios# sudo tar -xvf nagios-4.4.6.tar.gz
nagios-4.4.6/
nagios-4.4.6/.gitignore
nagios-4.4.6/.travis.yml
nagios-4.4.6/CONTRIBUTING.md
nagios-4.4.6/Changelog
nagios-4.4.6/INSTALLING
nagios-4.4.6/LEGAL
nagios-4.4.6/LICENSE
nagios-4.4.6/Makefile.in
nagios-4.4.6/README.md
nagios-4.4.6/THANKS
nagios-4.4.6/UPGRADING
nagios-4.4.6/aclocal.m4
nagios-4.4.6/autoconf-macros/
```

```
root@Aloha:~/nagios/nagios-4.4.6# sudo ./configure --with-httpd-conf=/etc/apache2/sites-enabled
```

```
*** Configuration summary for nagios 4.4.6 2020-04-28 ***:

 General Options:
 -------------------------
        Nagios executable:  nagios
        Nagios user/group:  nagios,nagios
       Command user/group:  nagios,nagios
             Event Broker:  yes
        Install ${prefix}:  /usr/local/nagios
root@Aloha:~/nagios/nagios-4.4.6# sudo make all
   Check result directory:  /usr/local/nagios/var/spool/checkresults
          Init directory:  /etc/init.d
  Apache conf.d directory:  /etc/apache2/sites-enabled
             Mail program:  /bin/mail
                  Host OS:  linux-gnu
          IOBroker Method:  epoll

 Web Interface Options:
 ------------------------
                HTML URL:  http://localhost/nagios/
                 CGI URL:  http://localhost/nagios/cgi-bin/
 Traceroute (used by WAP):  /usr/sbin/traceroute


Review the options above for accuracy.  If they look okay,
type 'make all' to compile the main program and CGIs.
```

STEP 3: Run the make all command to compile the program alongside the CGIs:

```
*** Support Notes *******************************************

If you have questions about configuring or running Nagios,
please make sure that you:

     - Look at the sample config files
     - Read the documentation on the Nagios Library at:
          https://library.nagios.com

before you post a question to one of the mailing lists.
Also make sure to include pertinent information that could
help others help you.  This might include:

     - What version of Nagios you are using
     - What version of the plugins you are using
     - Relevant snippets from your config files
     - Relevant error messages from the Nagios log file

For more information on obtaining support for Nagios, visit:

       https://support.nagios.com

************************************************************

Enjoy.
```

STEP 4: Create group users and install sample config files

```
root@Aloha:~/nagios/nagios-4.4.6# sudo make install-groups-users
groupadd -r nagios
useradd -g nagios nagios
root@Aloha:~/nagios/nagios-4.4.6#
```

```
root@Aloha:~/nagios/nagios-4.4.6# sudo usermod -a -G nagios www-data
root@Aloha:~/nagios/nagios-4.4.6#
```

```
root@Aloha:~/nagios/nagios-4.4.6# sudo make install
cd ./base && make install
root@Aloha:~/nagios/nagios-4.4.6# sudo make install-init
/usr/bin/install -c -m 755 -d -o root -g root /etc/init.d
/usr/bin/install -c -m 755 -o root -g root startup/default-init /etc/init.d/nagios
root@Aloha:~/nagios/nagios-4.4.6#




make[2]: Entering directory '/root/nagios/nagios-4.4.6/cgi'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/sbin
for file in *.cgi; do \
       /usr/bin/install -c -s -m 775 -o nagios -g nagios $file /usr/local/nagios/sbin; \
done
make[2]: Leaving directory '/root/nagios/nagios-4.4.6/cgi'
make[1]: Leaving directory '/root/nagios/nagios-4.4.6/cgi'
cd ./html && make install
```

```
root@Aloha:~/nagios/nagios-4.4.6# sudo make install-commandmode
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***

root@Aloha:~/nagios/nagios-4.4.6#
```

```
root@Aloha:~/nagios/nagios-4.4.6# sudo make install-config
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc/
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/nagios.c
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/cgi.cfg
/usr/bin/install -c -b -m 660 -o nagios -g nagios sample-config/resource
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template
bjects/templates.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template
jects/commands.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template
jects/contacts.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template
/objects/timeperiods.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template
bjects/localhost.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template
ects/windows.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template
ects/printer.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template
cts/switch.cfg

*** Config files installed ***

Remember, these are *SAMPLE* config files.  You'll need to read
the documentation for more information on how to actually define
services, hosts, etc. to fit your particular needs.
```

STEP 5: Set up Apache and Nagios UI

```
root@Aloha:~/nagios/nagios-4.4.6# sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/apache2/sites-enabled/nagios.conf
if [ 0 -eq 1 ]; then \
        ln -s /etc/apache2/sites-enabled/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

*** Nagios/Apache conf file installed ***

root@Aloha:~/nagios/nagios-4.4.6#
```

```
root@Aloha:~/nagios/nagios-4.4.6# sudo a2enmod rewrite cgi
Enabling module rewrite.
Enabling module cgi.
To activate the new configuration, you need to run:
  service apache2 restart
root@Aloha:~/nagios/nagios-4.4.6#
```

STEP 6: Create nagios user and set password

```
root@Aloha:~/nagios/nagios-4.4.6# sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
root@Aloha:~/nagios/nagios-4.4.6#
```

Step 7: Install nagios plugin

```
root@Aloha:~/nagios# sudo apt install monitoring-plugins nagios-nrpe-plugin -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfwupdplugin1
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  libdbi1 libradcli4 libtirpc-common libtirpc3 monitoring-plugins-basic monitoring-plugins-common
  monitoring-plugins-standard python3-crypto python3-gpg python3-samba python3-tdb rpcbind samba-common
  samba-common-bin samba-dsdb-modules smbclient
Suggested packages:
  icinga | icinga2 nagios-plugins-contrib fping postfix | sendmail-bin | exim4-daemon-heavy | exim4-daemon-light
  qstat heimdal-clients python3-markdown python3-dnspython cifs-utils
The following NEW packages will be installed:
  libdbi1 libradcli4 libtirpc-common libtirpc3 monitoring-plugins monitoring-plugins-basic
  monitoring-plugins-common monitoring-plugins-standard nagios-nrpe-plugin python3-crypto python3-gpg python3-samba
  python3-tdb rpcbind samba-common samba-common-bin samba-dsdb-modules smbclient
0 upgraded, 18 newly installed, 0 to remove and 23 not upgraded.
Need to get 4992 kB of archives.
After this operation, 33.4 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 libtirpc-common all 1.2.5-1ubuntu0.1 [7712 B]
Get:2 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 libtirpc3 amd64 1.2.5-1ubuntu0.1 [77.9 kB]
Get:3 http://archive.ubuntu.com/ubuntu focal/main amd64 rpcbind amd64 1.2.5-8 [42.8 kB]
Get:4 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 samba-common all 2:4.13.17~dfsg-0ubuntu1.20.04.1 [69.
1 kB]
```

```
root@Aloha:/usr/local/nagios/etc# ll
total 84
drwxrwxr-x 3 nagios nagios  4096 Sep 17 23:21 ./
drwxr-xr-x 8 root   root    4096 Sep 17 23:17 ../
-rw-rw-r-- 1 nagios nagios 13710 Sep 17 23:17 cgi.cfg
-rw-r--r-- 1 root   root      50 Sep 17 23:22 htpasswd.users
-rw-rw-r-- 1 nagios nagios 45843 Sep 17 23:17 nagios.cfg
drwxrwxr-x 2 nagios nagios  4096 Sep 17 23:17 objects/
-rw-rw---- 1 nagios nagios  1312 Sep 17 23:17 resource.cfg
root@Aloha:/usr/local/nagios/etc#
```

```
50 # directive as shown below:
51
52 cfg_dir=/usr/local/nagios/etc/servers
53 cfg_dir=/usr/local/nagios/etc/printers
54 cfg_dir=/usr/local/nagios/etc/switches
55 cfg_dir=/usr/local/nagios/etc/routers
56
57
58
```

```
# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {

    contact_name            nagiosadmin             ; Short name of user
    use                     generic-contact         ; Inherit default values from generic-c
bove)
    alias                   Nagios Admin            ; Full name of user
    email                   rshanker084@gmail.com   ; <<***** CHANGE THIS TO YOUR EMAIL ADDR
}



###############################################################################
#
# CONTACT GROUPS
#
###############################################################################
```

root@Aloha: /usr/local/nagios/etc

GNU nano 4.8 resource.cfg

```
##############################################################################
#
# RESOURCE.CFG - Sample Resource File for Nagios 4.4.6
#
#
# You can define $USERx$ macros in this file, which can in turn be use
# in command definitions in your host config file(s).  $USERx$ macros
# useful for storing sensitive information such as usernames, password
# etc.  They are also handy for specifying the path to plugins and
# event handlers - if you decide to move the plugins or event handlers
# a different directory in the future, you can just update one or two
# $USERx$ macros, instead of modifying a lot of command definitions.
#
# The CGIs will not attempt to read the contents of resource files, so
# you can set restrictive permissions (600 or 660) on them.
#
# Nagios supports up to 256 $USERx$ macros ($USER1$ through $USER256$)
#
# Resource files may also be used to store configuration directives for
# external data sources like MySQL...
#
##############################################################################

# Sets $USER1$ to be the path to the plugins
#$USER1$=/usr/local/nagios/libexec
$USER1$=/usr/lib/nagios/plugins
# Sets $USER2$ to be the path to event handlers
#$USER2$=/usr/local/nagios/libexec/eventhandlers

# Store some usernames and passwords (hidden from the CGIs)
#$USER3$=someuser
#$USER4$=somepassword
```

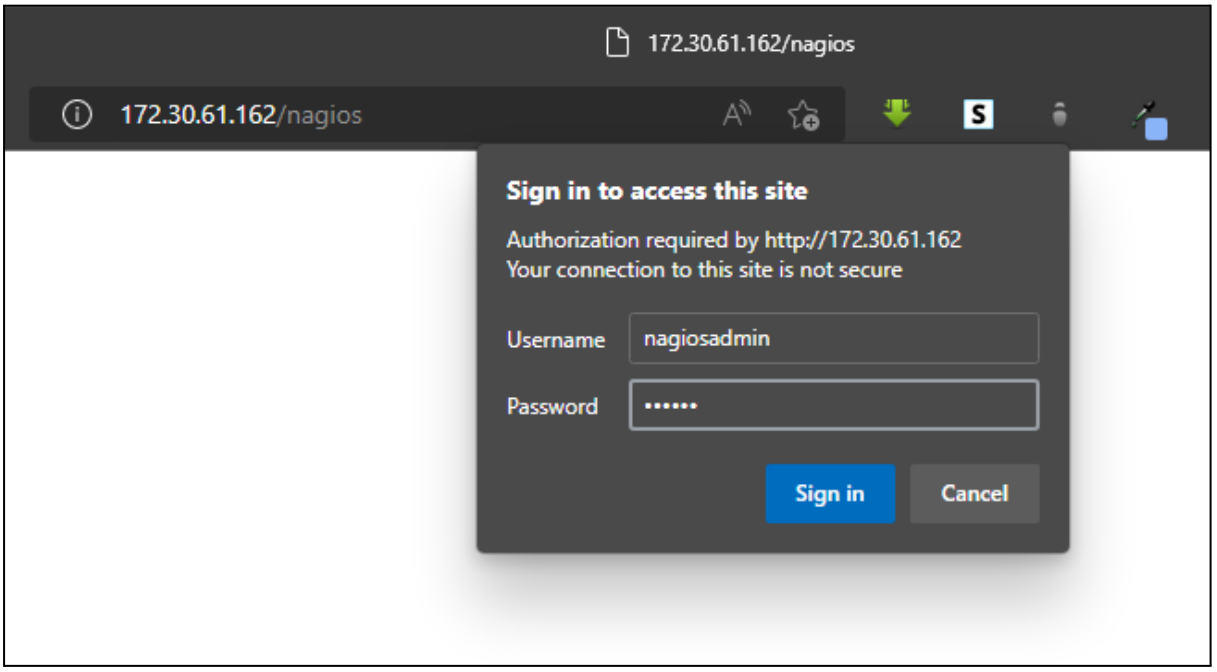
```
##############################################################################
#
# SAMPLE PERFORMANCE DATA COMMANDS
#
# These are sample performance data commands that can be used to send performance
# data output to two text files (one for hosts, another for services).  If you
# plan on simply writing performance data out to a file, consider using the
# host_perfdata_file and service_perfdata_file options in the main config file.
#
##############################################################################

define command {

    command_name    process-host-perfdata
    command_line    /usr/bin/printf "%b" "$LASTHOSTCHECK$\t$HOSTNAME$\t$HOSTSTATE$\t$HOSTA
}



define command {

    command_name    process-service-perfdata
    command_line    /usr/bin/printf "%b" "$LASTSERVICECHECK$\t$HOSTNAME$\t$SERVICEDESC$\t$
}
define command{
        command_name check_nrpe
        command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$
}
```

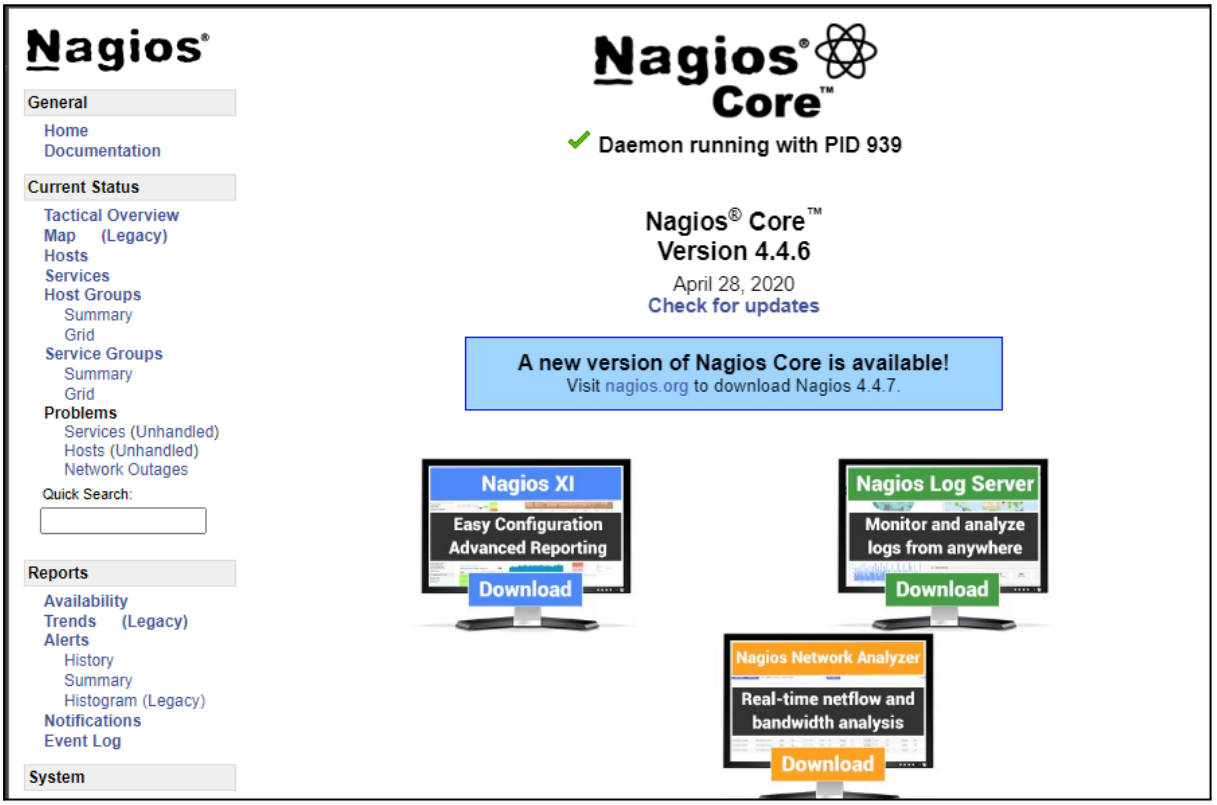
Step 8 : Start the Nagios Daemon and check status

```
root@Aloha:~# sudo systemctl restart apache2
root@Aloha:~# sudo systemctl restart nagios
root@Aloha:~# sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apache2
root@Aloha:~# sudo systemctl enable nagios
nagios.service is not a native service, redirecting to systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable nagios
```

```
root@Aloha:~# sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2022-09-17 23:40:55 IST; 2min 30s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 709 (apache2)
      Tasks: 6 (limit: 4314)
     Memory: 11.4M
     CGroup: /system.slice/apache2.service
             ├─709 /usr/sbin/apache2 -k start
             ├─710 /usr/sbin/apache2 -k start
             ├─711 /usr/sbin/apache2 -k start
             ├─712 /usr/sbin/apache2 -k start
             ├─713 /usr/sbin/apache2 -k start
             └─714 /usr/sbin/apache2 -k start

Sep 17 23:40:55 Aloha systemd[1]: Starting The Apache HTTP Server...
Sep 17 23:40:55 Aloha systemd[1]: Started The Apache HTTP Server.
root@Aloha:~# sudo systemctl status nagios
● nagios.service - LSB: Starts and stops the Nagios monitoring server
     Loaded: loaded (/etc/init.d/nagios; generated)
     Active: active (running) since Sat 2022-09-17 23:41:05 IST; 2min 40s ago
       Docs: man:systemd-sysv-generator(8)
      Tasks: 20 (limit: 4314)
     Memory: 6.1M
     CGroup: /system.slice/nagios.service
             ├─939 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─940 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─941 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
```

Access the tool by opening the browser and navigating to the http://server-IP/nagios URL.





ADVANCED DEVOPS LAB

Name : Ravi Pandey

Roll no : 43

Experiment No : 10

Aim:

To perform Port, Service monitoring, Windows/Linux server monitoring using Nagios.

Theory:

Nagios is an open source monitoring system for computer systems. It was designed to run on the Linux operating system and can monitor devices running Linux, Windows and Unix operating systems (OSes).

Nagios runs both agent-based and agentless configurations. Independent agents are installed on any hardware or software system to collect data that is then reported back to the management server. Agentless monitoring uses existing protocols to emulate an agent. Both approaches can monitor file system usage, OS metrics, service and process states and more.

Here are the important reasons to use Nagios monitoring tool:

- Detects all types of network or server issues
- Helps you to find the root cause of the problem which allows you to get the permanent solution to the problem
- Active monitoring of your entire infrastructure and business processes
- Allows you to monitors and troubleshoot server performance issues
- Helps you to plan for infrastructure upgrades before outdated systems create failures
- You can maintain the security and availability of the service
- Automatically fix problems in a panic situation

Output:

Install all plugins

```
┌──(root㉿kali)-[/home/kali]
└─# sudo apt install monitoring-plugins nagios-nrpe-plugin -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
monitoring-plugins is already the newest version (2.3.1-1).
The following NEW packages will be installed:
  nagios-nrpe-plugin
0 upgraded, 1 newly installed, 0 to remove and 662 not upgraded.
Need to get 31.0 kB of archives.
After this operation, 85.0 kB of additional disk space will be used.
Get:1 http://http.kali.org/kali kali-rolling/main amd64 nagios-nrpe-plugin amd64 4.1.0-1+b1 [31.0 kB]
Fetched 31.0 kB in 1s (22.5 kB/s)
Selecting previously unselected package nagios-nrpe-plugin.
(Reading database ... 344263 files and directories currently installed.)
Preparing to unpack .../nagios-nrpe-plugin_4.1.0-1+b1_amd64.deb ...
Unpacking nagios-nrpe-plugin (4.1.0-1+b1) ...
Setting up nagios-nrpe-plugin (4.1.0-1+b1) ...
Scanning processes...
Scanning candidates...
Scanning processor microcode...
Scanning linux images...
```

Go and check the files that are present in the following directory

```
┌──(root㉿kali)-[/home/kali/nagios]
└─# cd /usr/local/nagios/etc

┌──(root㉿kali)-[/usr/local/nagios/etc]
└─# ls
cgi.cfg  htpasswd.users  nagios.cfg  objects  resource.cfg
```

Uncomment these lines

```
# You can specify individual object config files as shown below:
cfg_file=/usr/local/nagios/etc/objects/commands.cfg
cfg_file=/usr/local/nagios/etc/objects/contacts.cfg
cfg_file=/usr/local/nagios/etc/objects/timeperiods.cfg
cfg_file=/usr/local/nagios/etc/objects/templates.cfg
```

Then create the following directories

```
┌──(root㉿kali)-[/usr/local/nagios/etc]
└─# mkdir servers printers switches routers
```

Then add this path in resource.cfg file for $USER1$

```
# Sets $USER1$ to be the path to the plugins
#$USER1$=/usr/local/nagios/libexec
$USER1$=/usr/lib/nagios/plugins
```

Now go to objects directory

```
┌──(root㉿kali)-[/usr/local/nagios/etc]
└─# cd objects
┌──(root㉿kali)-[/usr/local/nagios/etc/objects]
└─# ls
commands.cfg  contacts.cfg  localhost.cfg  printer.cfg  switch.cfg  templates.cfg  timeperiods.cfg  windows.cfg
```

Open file contacts.cfg in any editor and change with your email id

```
define contact {

    contact_name            nagiosadmin             ; Short name of user
    use                     generic-contact         ; Inherit default values from generic-contact template (defined a>
    alias                   Nagios Admin            ; Full name of user
    email                   rshanker084@gmail.com   ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}
```

Now go to commands.cfg and enter the following define at the end

```
define command{
        command_name check_nrpe
        command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$
}
```

Check and update the firewall in the following manner

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw status
Status: inactive
```

Restart and enable both apache and nagios

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl restart apache2

┌──(kali㉿kali)-[~]
└─$ sudo systemctl restart nagios

┌──(kali㉿kali)-[~]
└─$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apache2
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.

┌──(kali㉿kali)-[~]
└─$ sudo systemctl enable nagios
```

Check their status

Apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: disabled)
     Active: active (running) since Fri 2022-11-04 00:11:18 IST; 1min 47s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 14558 (apache2)
      Tasks: 7 (limit: 8666)
     Memory: 13.3M
        CPU: 35ms
     CGroup: /system.slice/apache2.service
             ├─14558 /usr/sbin/apache2 -k start
             ├─14560 /usr/sbin/apache2 -k start
             ├─14561 /usr/sbin/apache2 -k start
             ├─14562 /usr/sbin/apache2 -k start
             ├─14563 /usr/sbin/apache2 -k start
             ├─14564 /usr/sbin/apache2 -k start
             └─14565 /usr/sbin/apache2 -k start

Nov 04 00:11:18 kali systemd[1]: Starting The Apache HTTP Server...
Nov 04 00:11:18 kali systemd[1]: Started The Apache HTTP Server.
```

nagios

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl status nagios
● nagios.service - Nagios
     Loaded: loaded (/etc/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Fri 2022-11-04 00:11:40 IST; 1min 29s ago
   Main PID: 14595 (nagios)
      Tasks: 20 (limit: 8666)
     Memory: 5.9M
        CPU: 30ms
     CGroup: /system.slice/nagios.service
             ├─14595 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
             ├─14597 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14598 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14599 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14600 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14601 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14602 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14603 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14604 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14605 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14606 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14607 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14608 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14609 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14610 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14611 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14612 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14613 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─14614 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─14621 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
```

Now in order to do port monitoring do the following

Go into the following directories

```
┌──(kali㉿kali)-[~]
└─$ sudo -i
┌─(Message from Kali developers)
│
│ This is a minimal installation of Kali Linux, you likely
│ want to install supplementary tools. Learn how:
│ ⇒ https://www.kali.org/docs/troubleshooting/common-minimum-setup/
│
└─(Run: "touch ~/.hushlogin" to hide this message)
┌──(root㉿kali)-[~]
└─# cd /usr/local

┌──(root㉿kali)-[/usr/local]
└─# cd nagios

┌──(root㉿kali)-[/usr/local/nagios]
└─# cd etc

┌──(root㉿kali)-[/usr/local/nagios/etc]
└─# ls
cgi.cfg  htpasswd.users  nagios.cfg  objects  printers  resource.cfg  routers  servers  switches
```

In servers directory, create a file called as hosts.cfg

```
┌──(root㉿kali)-[/usr/local/nagios/etc]
└─# cd servers

┌──(root㉿kali)-[/usr/local/nagios/etc/servers]
└─# nano hosts.cfg
```

Add the following ip and name according to your machine and ip address (in our case, we have used a mobile and used its wifi IP)

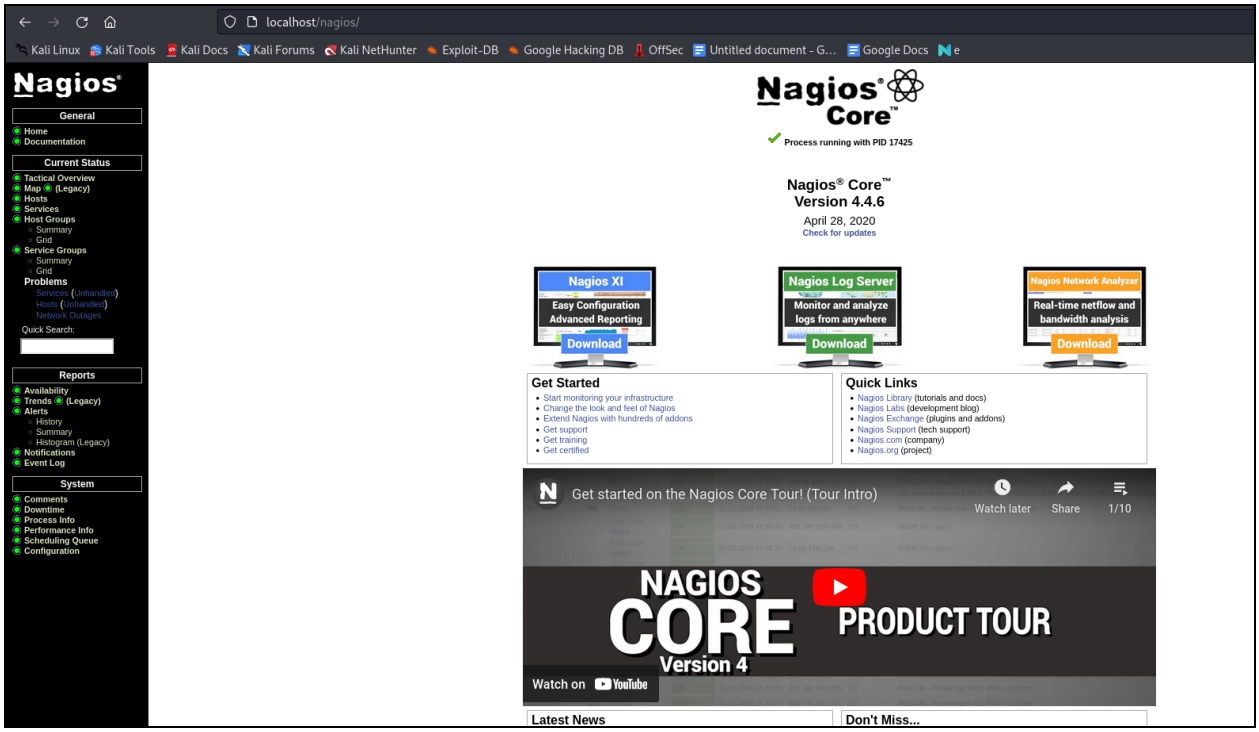
```
  GNU nano 6.4
define host {
        use                     linux-server
        host_name               Xiaomi
        alias                   phone
        address                 192.168.0.103
        max_check_attempts      5
        check_period            24x7
        notification_interval   30
        notification_period     24x7
}
```

Now restart the nagios service

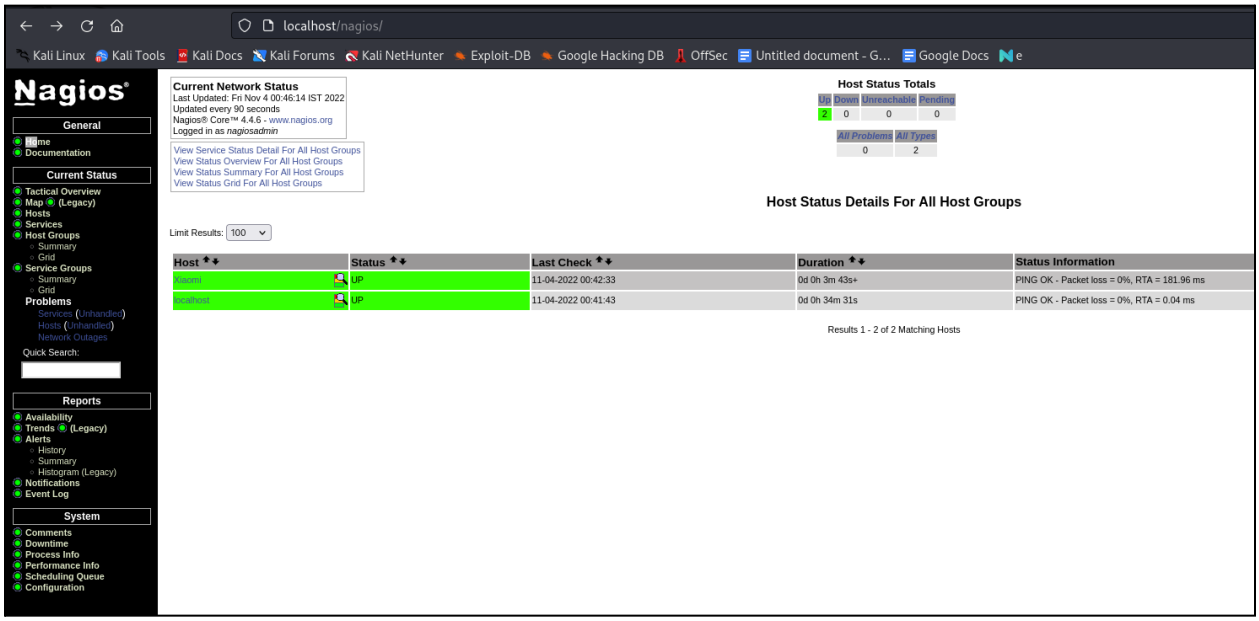
```
┌──(root㉿kali)-[/usr/local/nagios/etc/servers]
└─# service nagios start

┌──(root㉿kali)-[/usr/local/nagios/etc/servers]
└─# service nagios restart
```

Open (ip_addr/nagios)

Put in your credentials, you will see the following screen



Now, go into hosts, we will be able to monitor our client



Conclusion:

Thus port monitoring is completed successfully.

Reference:

https://www.youtube.com/watch?v=UMHgRnPXoEw

https://www.youtube.com/watch?v=6T_RCywnLB8&list=PLvoGk4CSiH0sfH21iVgaPKx3DlI8dJYte&index=4

ADVANCED DEVOPS LAB

Name : Ravi Pandey

Roll no : 43

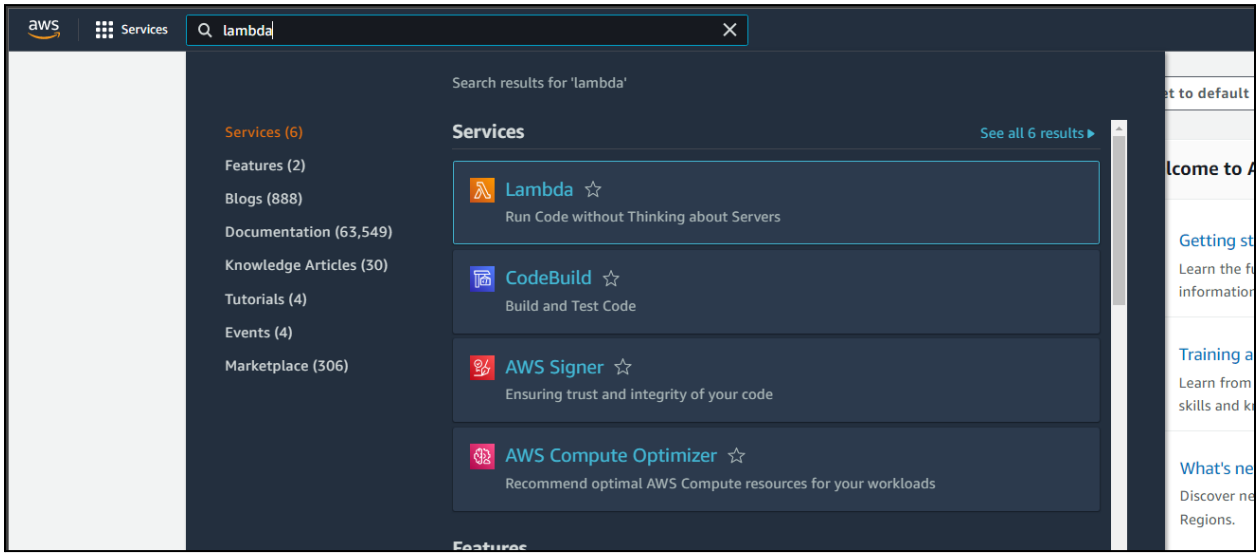
Experiment No : 11

Aim:

To understand AWS Lambda, its workflow, and various functions and create your first Lambda functions using Python / Java / Nodejs

Output:

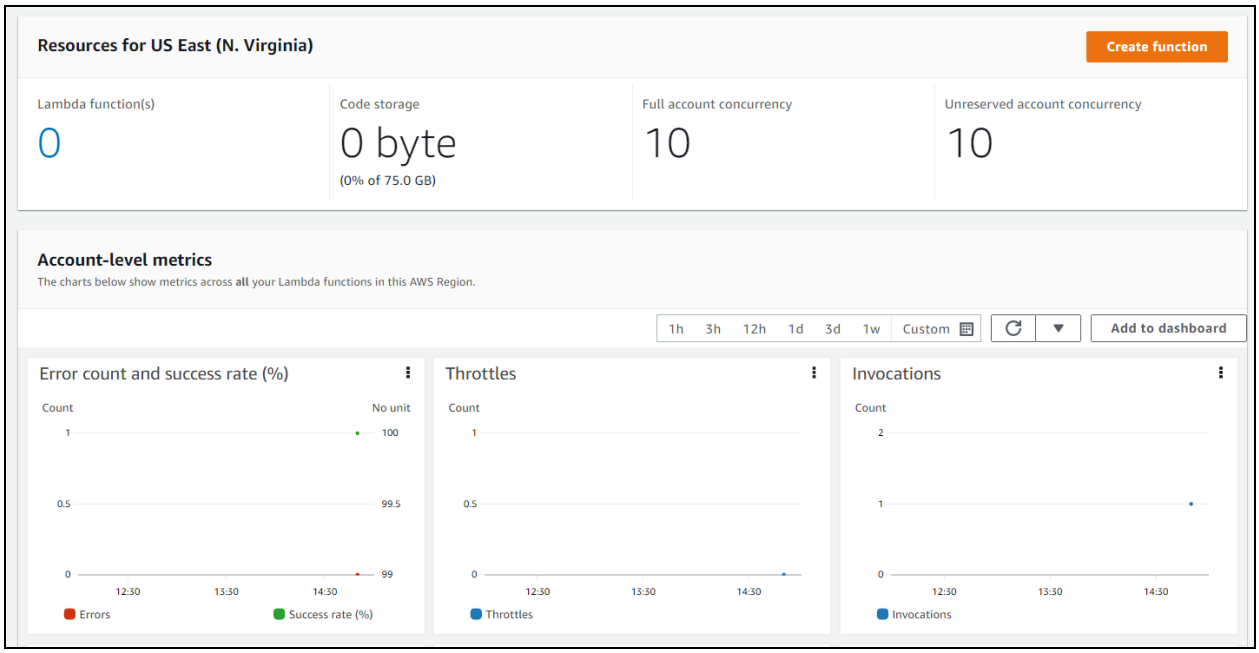
Step 1 : Enter The lambda console



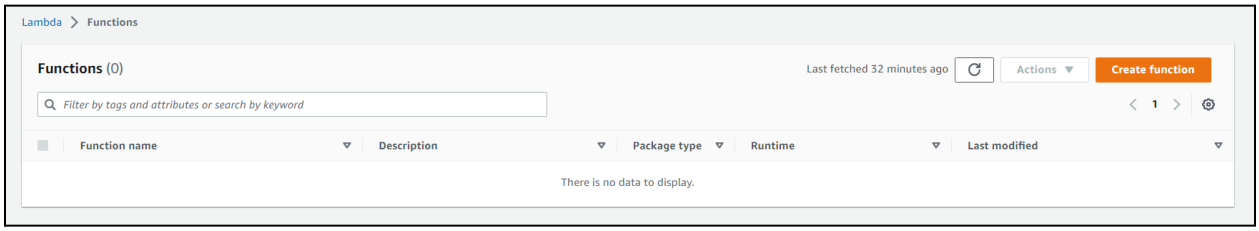
Step 2 : Select a lambda blueprint

Blueprints provide example code to do some minimal processing. Most blueprints process events from specific event sources, such as Amazon S3, Amazon DynamoDB, or a custom application.

a. In the AWS Lambda console, choose Create function.



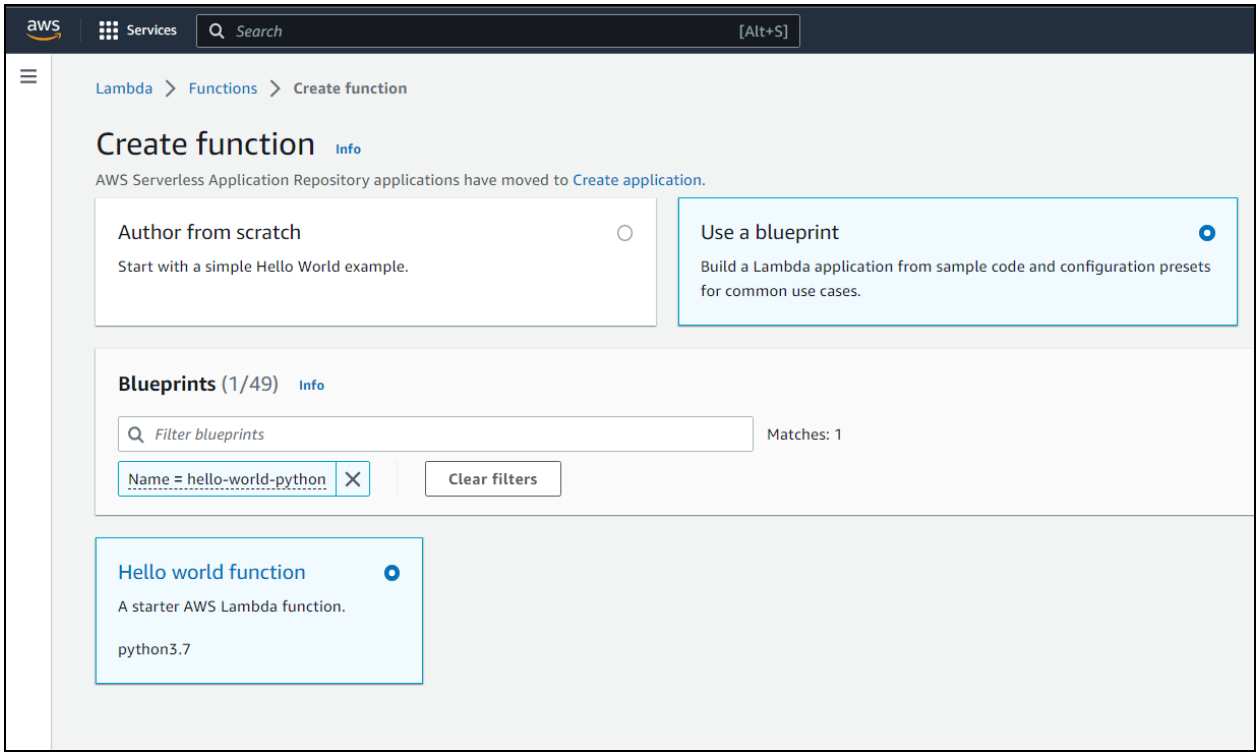
b. In functions, the create function is present



c. Select use a blueprint.

d. In the Filter box, enter hello-world-python and select the hello-world-python blueprint.

e. Then choose Configure.



Step 3 : Configure and create your lambda function

A Lambda function consists of code you provide, associated dependencies, and configuration. The configuration information you provide includes the compute resources you want to allocate (for example, memory), execution timeout, and an IAM role that AWS Lambda can assume to execute your Lambda function on your behalf.

a. You will now enter Basic information about your Lambda function.

Basic information:

Name: You can name your Lambda function here. For this tutorial, enter hello-world-python.

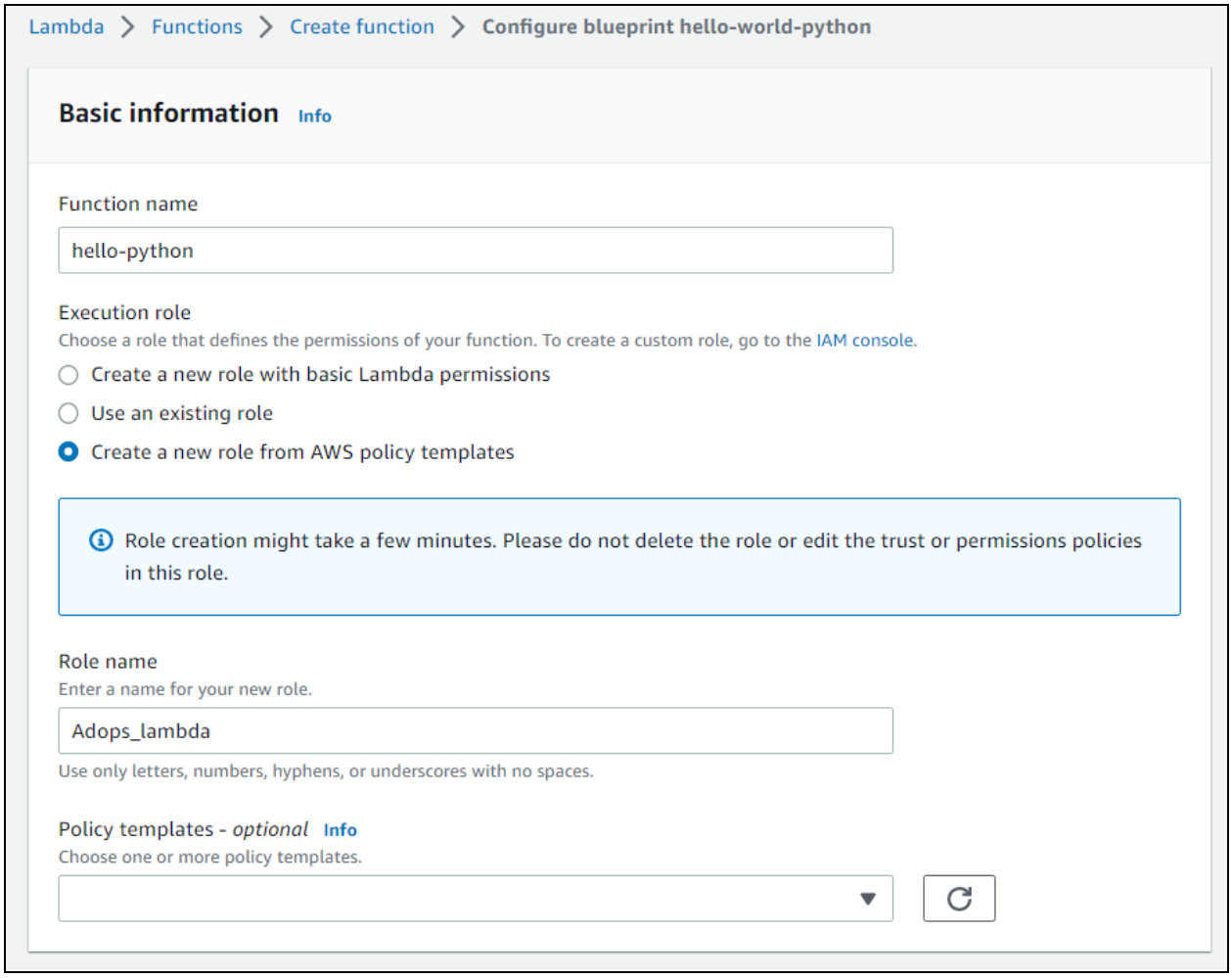
Role: You will create an IAM role (referred to as the execution role) with the necessary permissions that AWS Lambda can assume to invoke your Lambda

function on your behalf. Select Create a new role from AWS policy templates.

Role name: type lambda_basic_execution.

Lambda function code:

b. Go to the bottom of the page and choose the Create function.



Lambda function code

Code is preconfigured by the chosen blueprint. You can configure it after you create the function. Learn more about deploying Lambda functions.

This function contains external libraries.

Runtime	Architecture
Python 3.7	x86_64

```
1  import json
2
3  print('Loading function')
4
5
6  def lambda_handler(event, context):
7      #print("Received event: " + json.dumps(event, indent=2))
8      print("value1 = " + event['key1'])
9      print("value2 = " + event['key2'])
10     print("value3 = " + event['key3'])
11     return event['key1']  # Echo back the first key value
12     #raise Exception('Something went wrong')
13
```

Cancel **Create function**

Successfully created the function **hello-python**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > hello-python

hello-python

Throttle | Copy ARN | Actions

Function overview Info

hello-python

Layers (0)

+ Add trigger

+ Add destination

Description
A starter AWS Lambda function.

Last modified
27 seconds ago

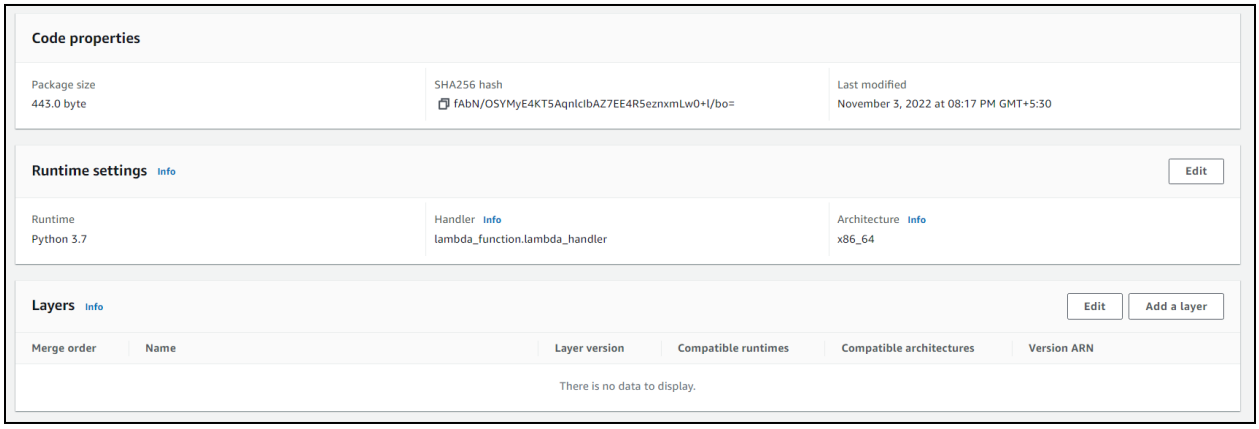
Function ARN
arn:aws:lambda:us-east-1:696940912922:function:hello-python

Function URL Info
-

c. Runtime: Currently, you can author your Lambda function code in Java, Node.js, C#, Go, or Python. For this tutorial, use Python 3.7 as the runtime.

d. Handler: You can specify a handler (a method/function in your code) where AWS Lambda can begin executing your code. AWS Lambda provides event data as input to this handler, which processes the event.

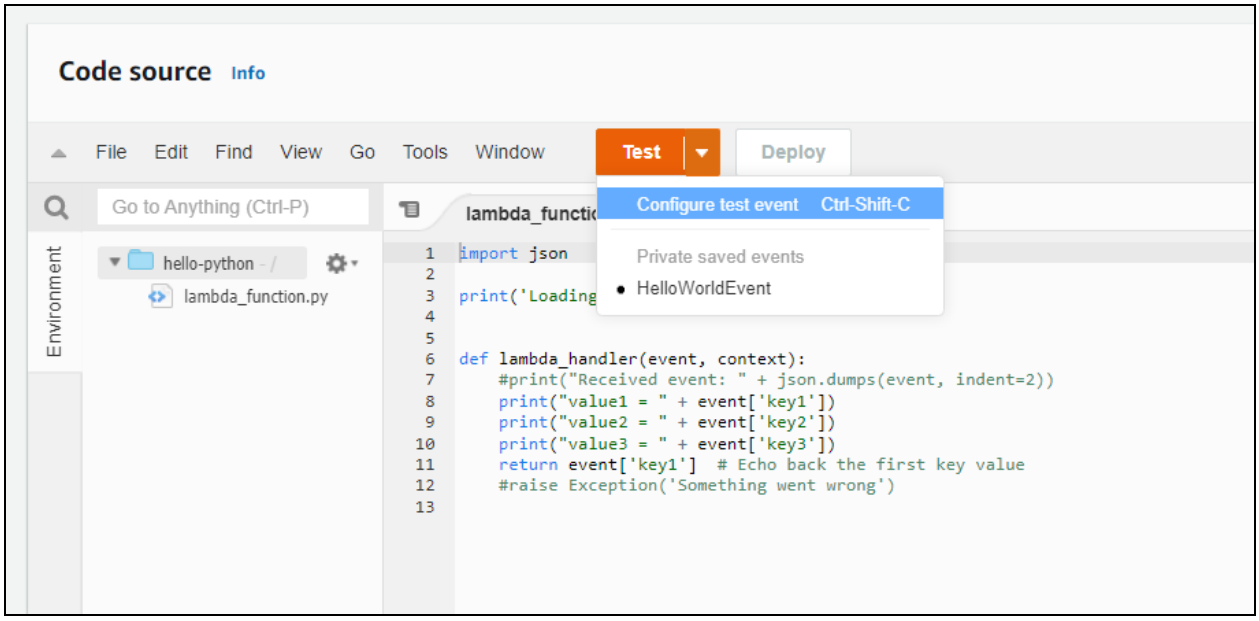
In this example, Lambda identifies this from the code sample and this should be pre-populated with lambda_function.lambda_handler.



Step 4 : Invoke lambda function and verify results

The console shows the hello-world-python Lambda function. You can now test the function, verify results, and review the logs.

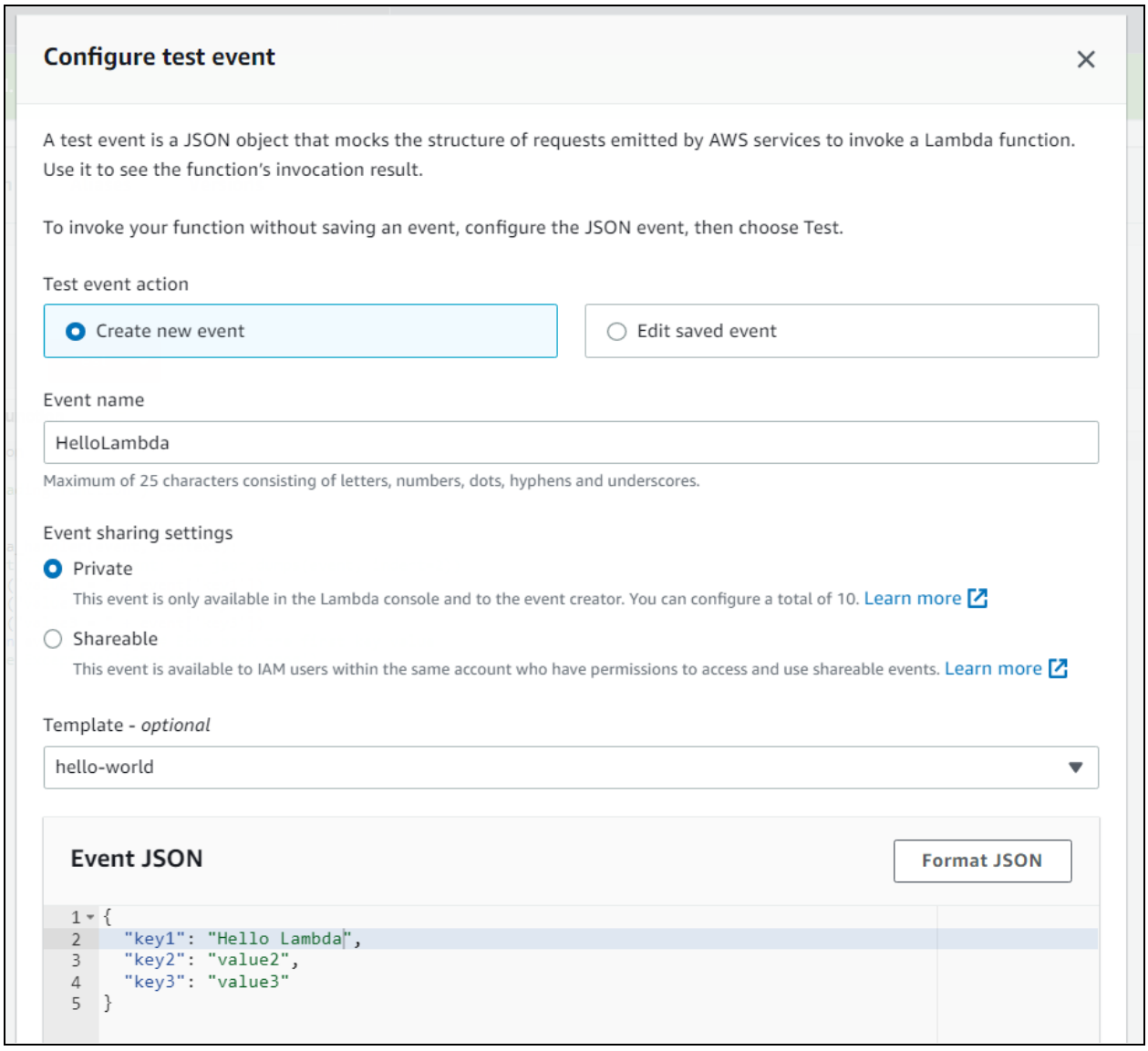
a. Select Configure Test Event from the drop-down menu called Test.



b. The editor pops up so you can enter an event to test your function.

- Select Create new event.
- Type in an event name like HelloWorldEvent.
- Retain default setting of Private for Event sharing settings.
- Choose hello-world from the template list.
- You can change the values in the sample JSON, but don't change the event structure.

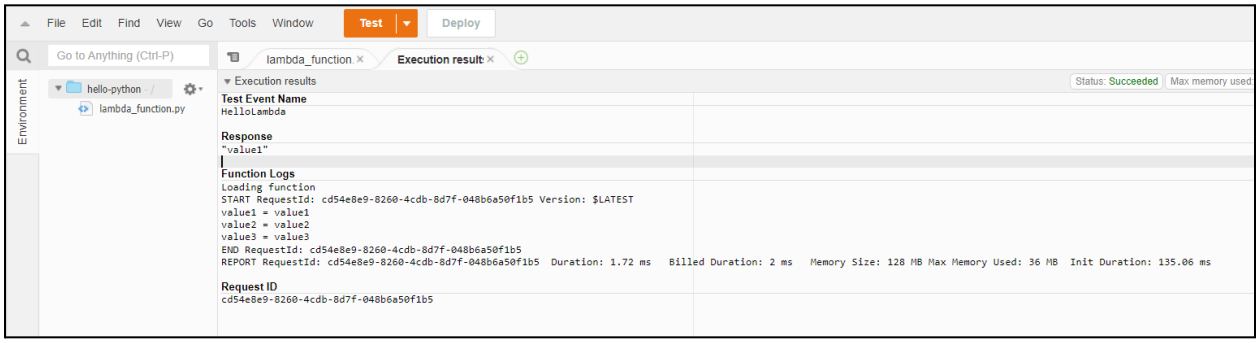
For now, replace value1 with hello world!. Select Create.



c. Choose Test.

d. Upon successful execution, view the results in the console:

- The Execution results tab verifies that the execution succeeded.
- The Function Logs section will show the logs generated by the Lambda function execution as well as key information reported in the Log output.



Step 5 : Monitor your metrics

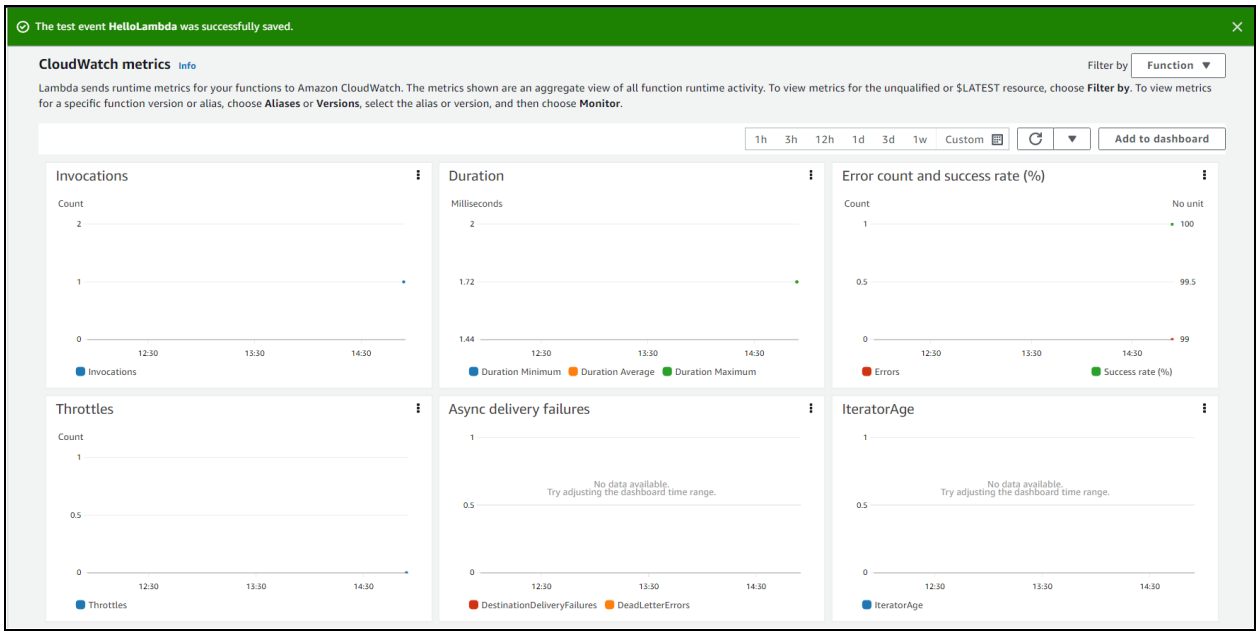
AWS Lambda automatically monitors Lambda functions and reports metrics through Amazon CloudWatch. To help you monitor your code as it executes, Lambda automatically tracks the number of requests, the latency per request, and the number of requests resulting in an error and publishes the associated metrics.

a. Invoke the Lambda function a few more times by repeatedly choosing the Test button. This will generate the metrics that can be viewed in the next step.

b. Select the Monitor tab to view the results.

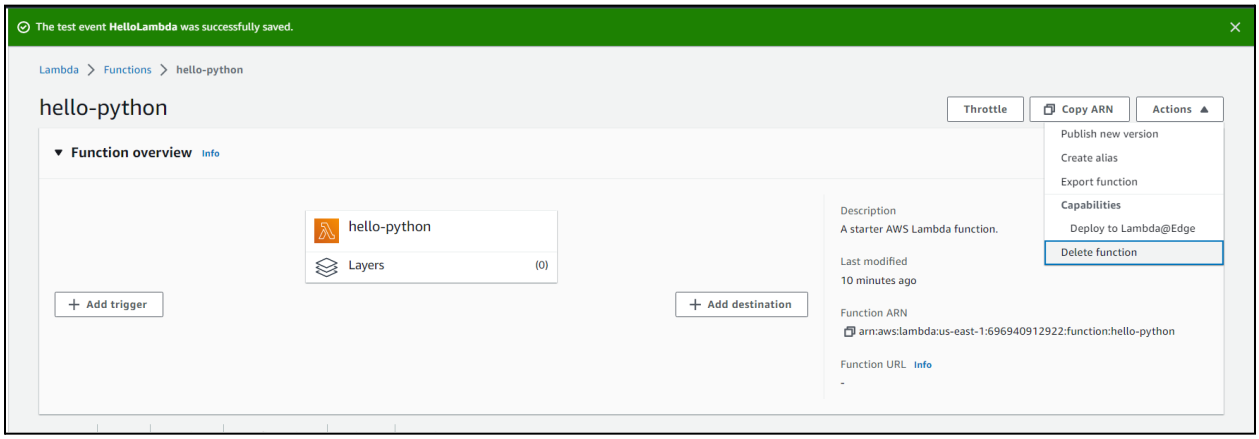
c. Scroll down to view the metrics for your Lambda function. Lambda metrics are reported through Amazon CloudWatch. You can leverage these metrics to set custom alarms. For more information about CloudWatch, see the Amazon CloudWatch Developer Guide.

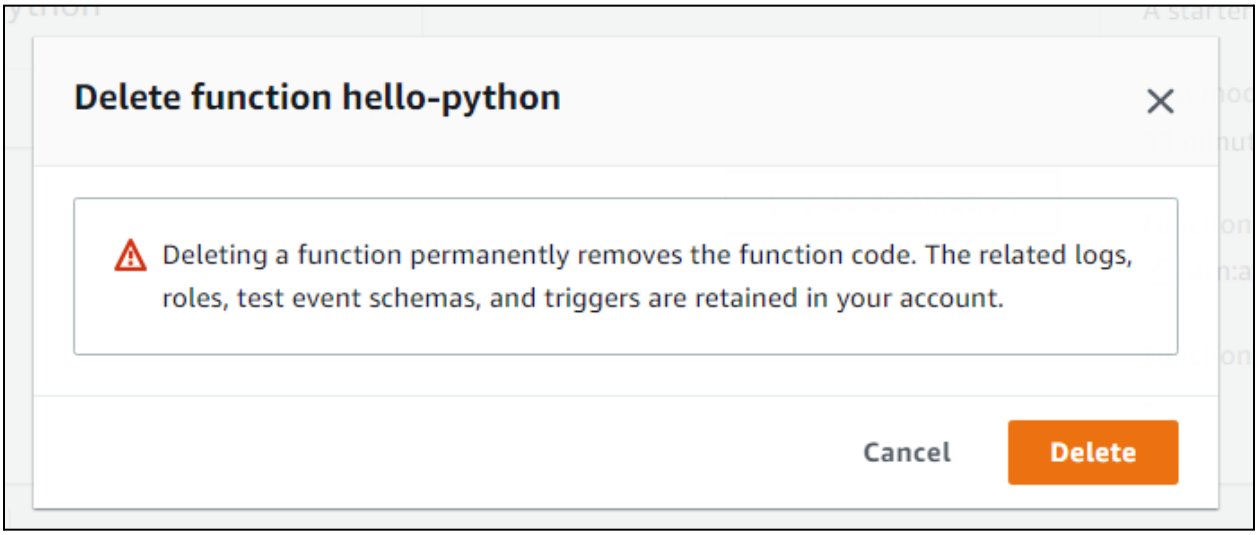
The Monitoring tab will show seven CloudWatch metrics: Invocations, Duration, Error count and success rate (%), Throttles, Async delivery failures, IteratorAge, and Concurrent executions.

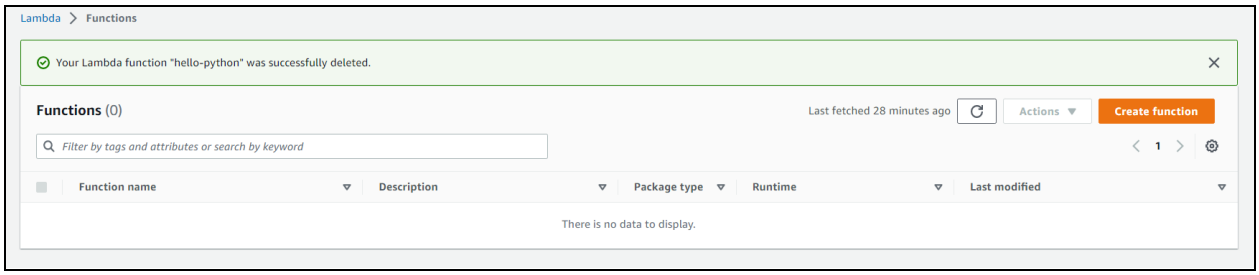


Step 6 : Delete the lambda function

a. Select the Actions button and select Delete function.







Conclusion:

First AWS Lambda function is created successfully.

Reference:

Run a Serverless "Hello, World!" with AWS Lambda

ADVANCED DEVOPS LAB

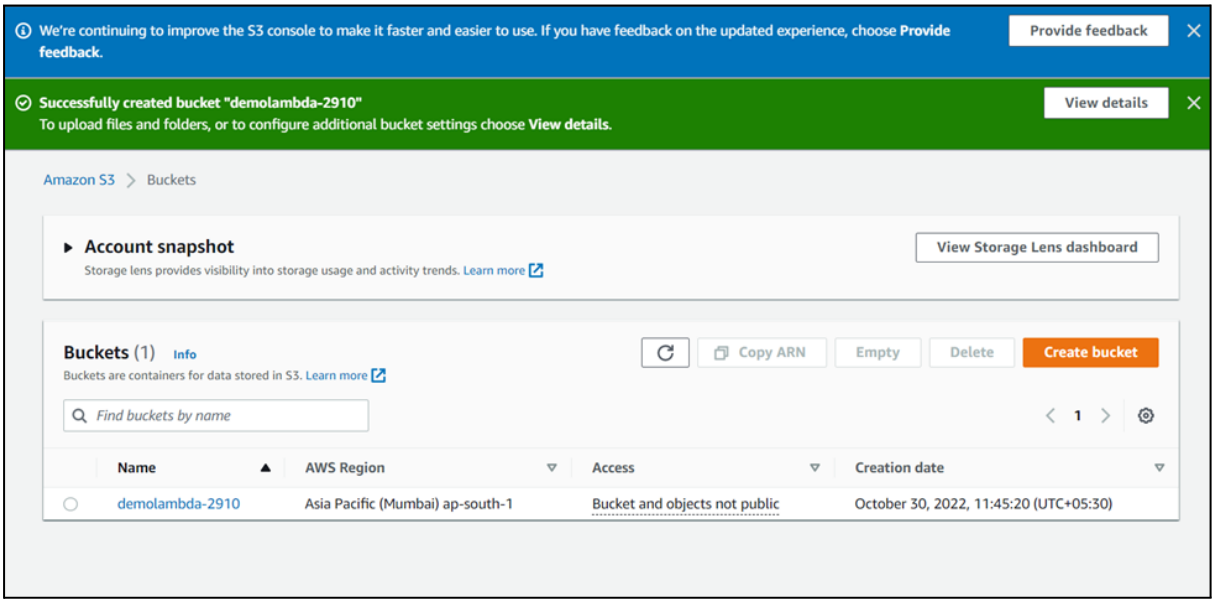
Name : Ravi Pandey
Roll no : 43
Experiment No : 12

To start using AWS Lambda with Amazon S3, we need the following −

- Create S3 Bucket
- Create role which has permission to work with s3 and lambda
- Create a lambda function and add s3 as the trigger.

Creating S3 Bucket

Step 1

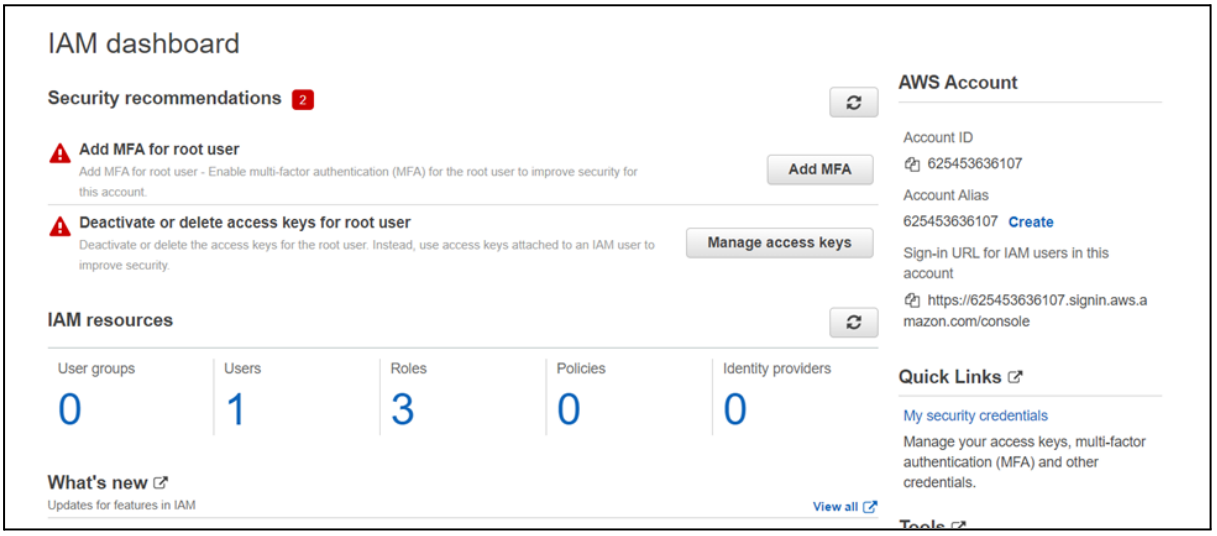


Create Role that Works with S3 and Lambda

To create role that works with S3 and Lambda, please follow the Steps given below −

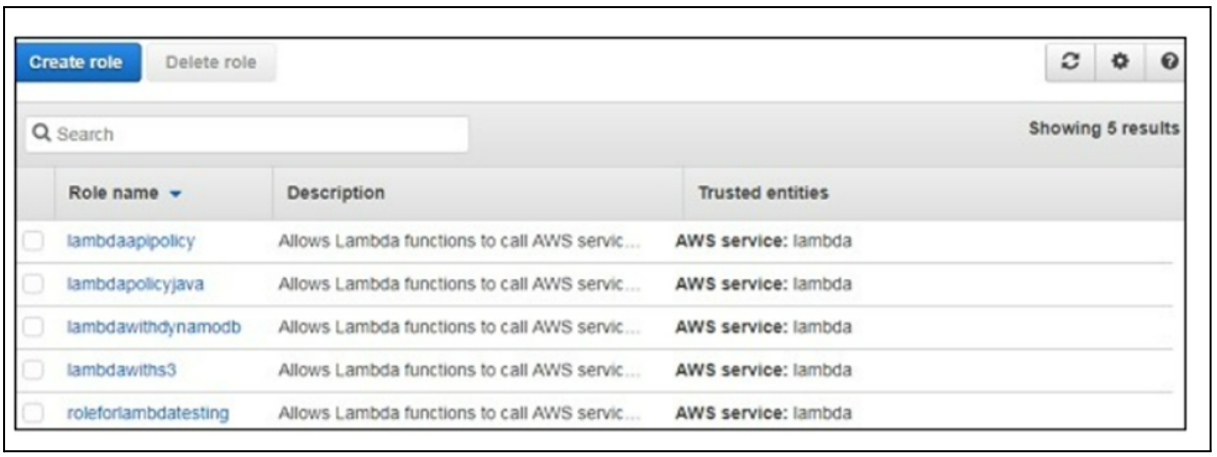
Step 1

Go to AWS services and select IAM as shown below −



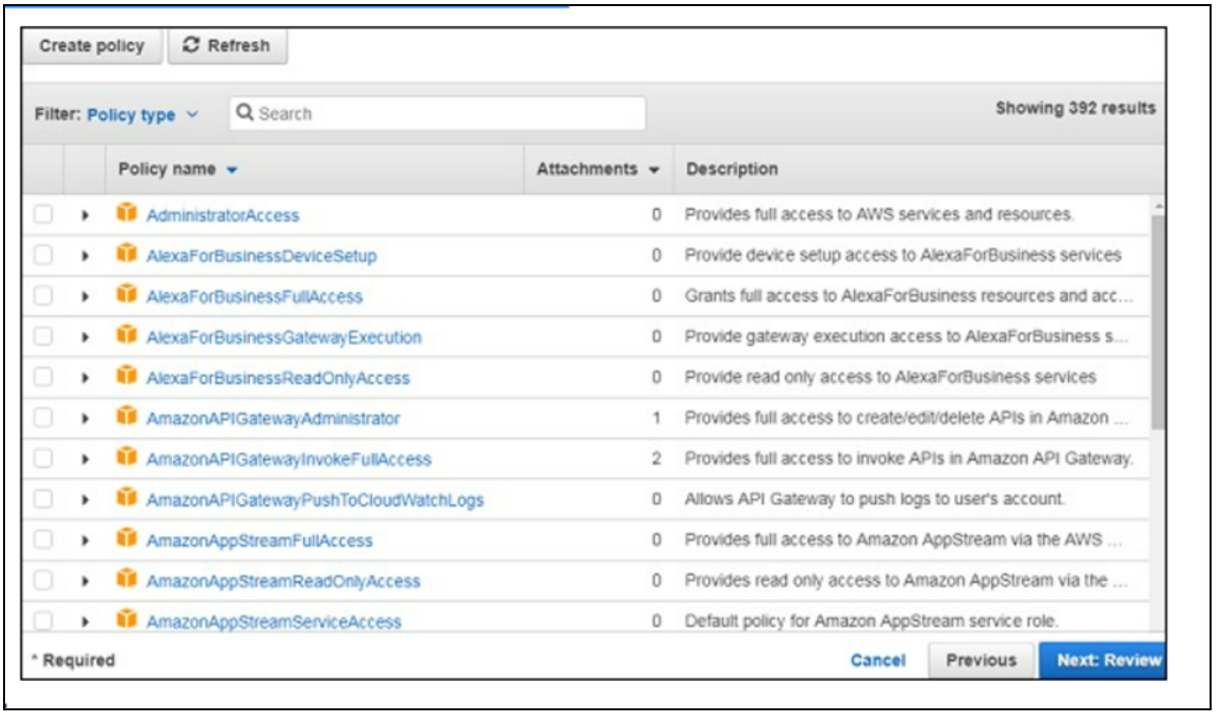
Step 2

Now, click IAM -> Roles as shown below −



Step 3

Add the permission from below and click Review.



Step 4

Observe that we have chosen the following permissions −

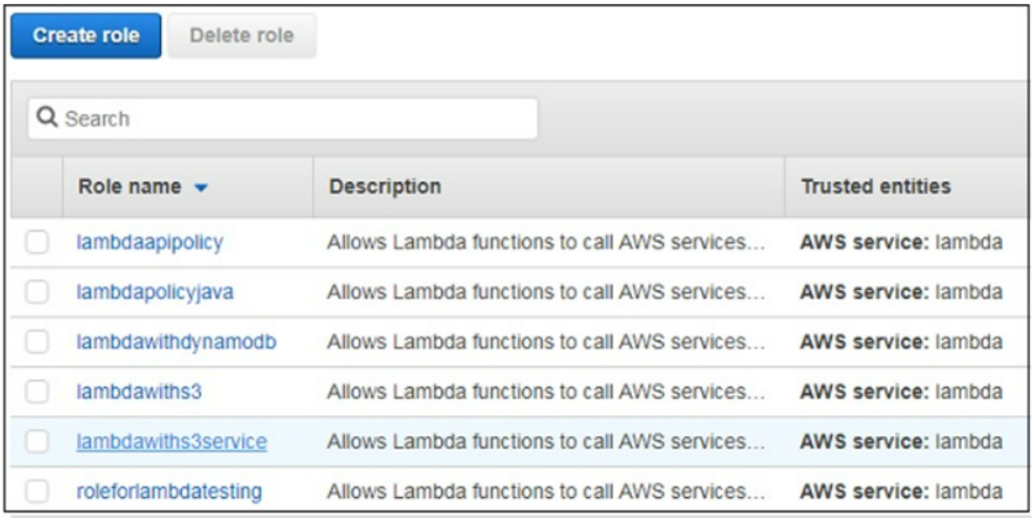
We have selected AmazonS3FullAccess, AWSLambdaFullAccess and CloudWatchFullAccess.

Step 5

Create Lambda function and Add S3 Trigger

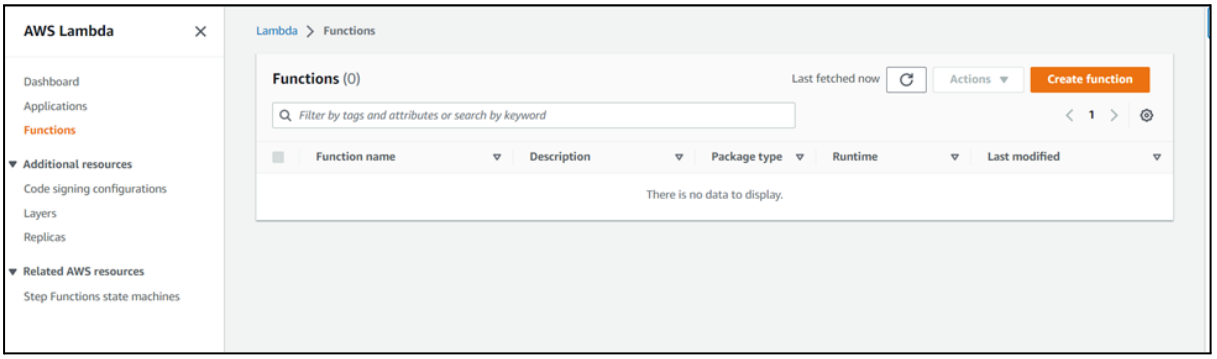
In this section, let us see how to create a Lambda function and add a S3 trigger to it. For this purpose, you will have to follow the Steps given below −

Now, enter the Role name, Role description and click ***Create Role*** button at the bottom.



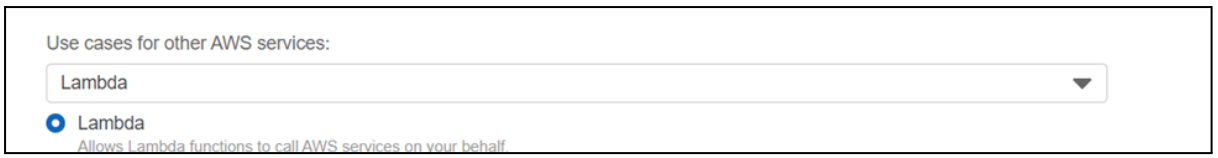
Step 1

Go to AWS Services and select Lambda as shown below −

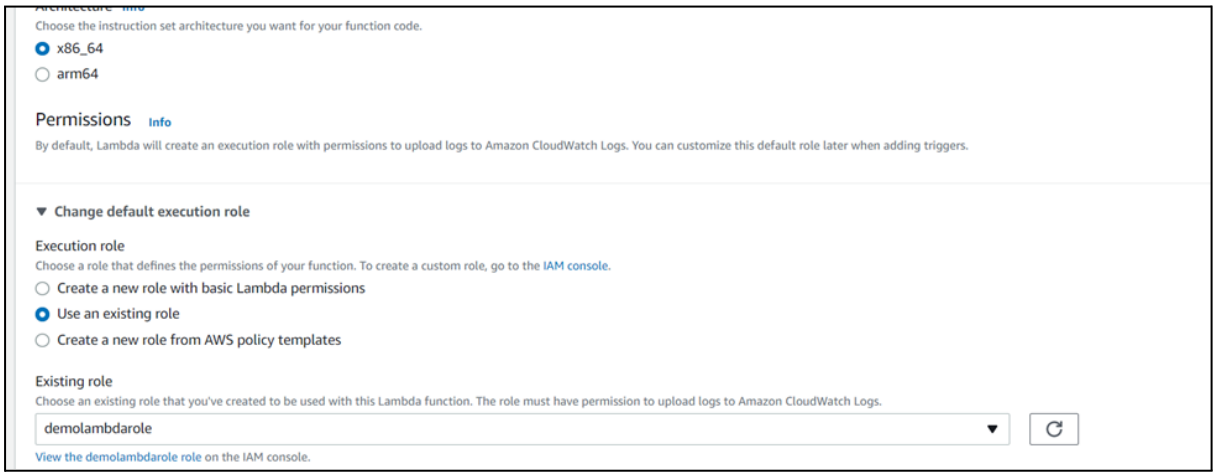


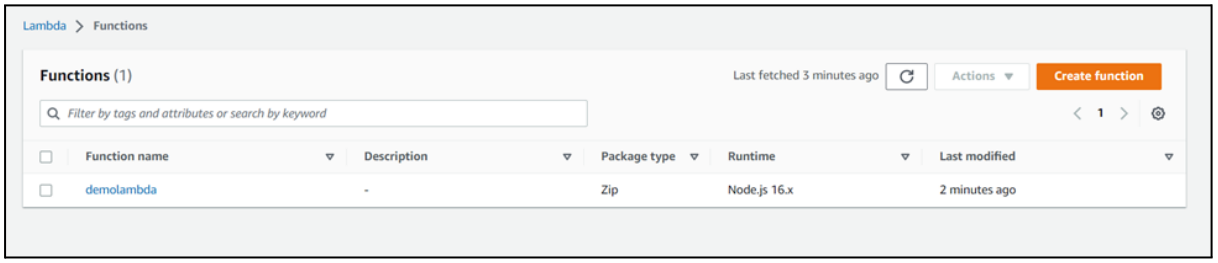
Step 2

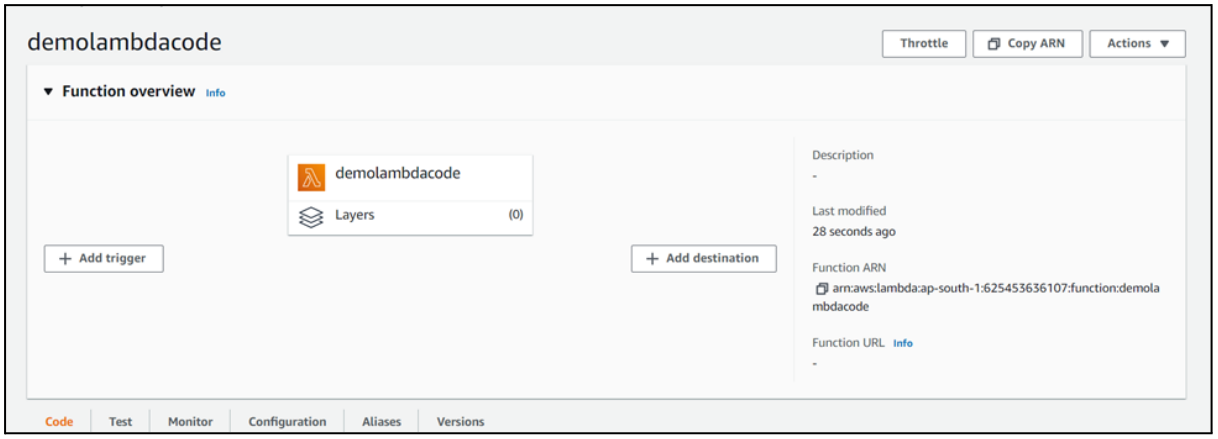
Click Lambda and follow the process for adding Name. Choose the Runtime, Role etc. and create the function. The Lambda function that we have created is shown in the screenshot below −


Use cases for other AWS services:
Lambda
Lambda
Allows Lambda functions to call AWS services on your behalf.


Basic information
Function name
Enter a name that describes the purpose of your function.
demolambdacode
Use only letters, numbers, hyphens, or underscores with no spaces.
Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Node.js 16.x
Architecture Info

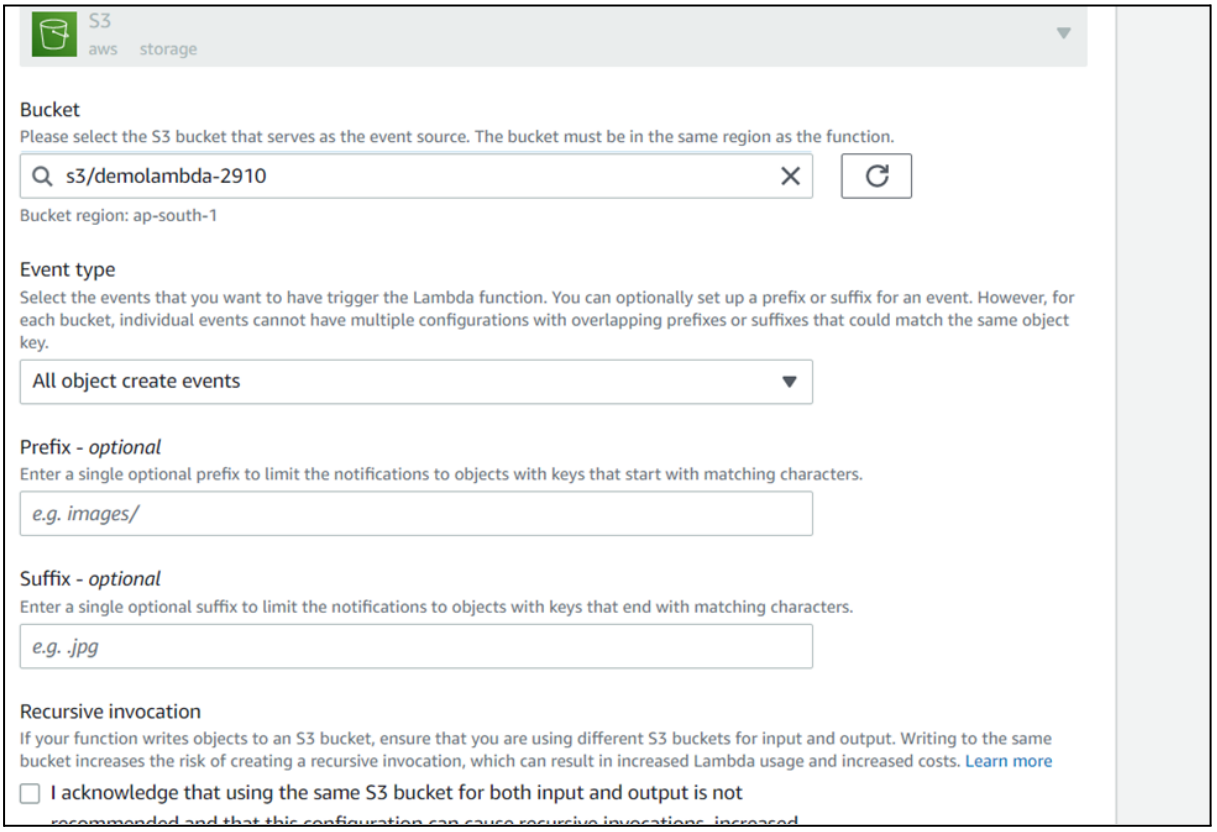

Choose the instruction set architecture you want for your function code.
x86_64
arm64
Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.
Change default execution role
Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
Use an existing role
Create a new role from AWS policy templates
Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
demolambdarole
View the demolambdarole role on the IAM console.


Lambda > Functions
Functions (1)
Last fetched 3 minutes ago
Actions
Create function
Filter by tags and attributes or search by keyword
Function name
Description
Package type
Runtime
Last modified
demolambda
-
Zip
Node.js 16.x
2 minutes ago


demolambdacode
Throttle
Copy ARN
Actions
Function overview Info
demolambdacode
Layers (0)
Add trigger
Add destination
Description
-
Last modified
28 seconds ago
Function ARN
arn:aws:lambda:ap-south-1:625453636107:function:demolambdacode
Function URL Info
-
Code
Test
Monitor
Configuration
Aliases
Versions

Hence a lambda function is created

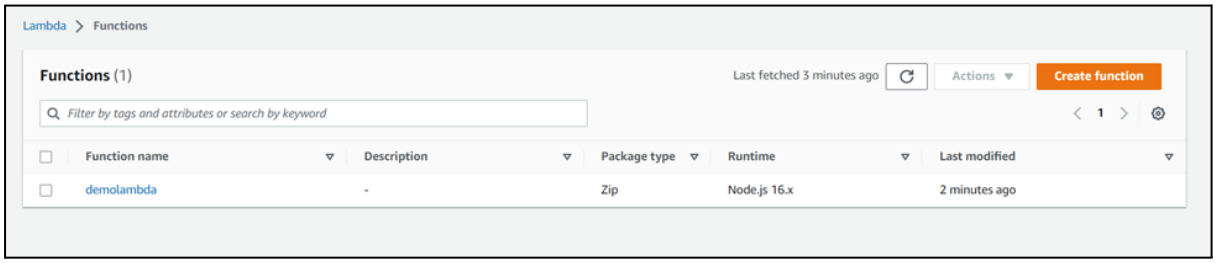
Now Let's Create a trigger



Below is the code written in node.js to pinned a log about an object that has been added to s3.

<u>Now to test our lambda function with s3 trigger.</u>

Step 1.add a image to s3 bucket



Step 2.

Now go to cloudwatch you will see the logs of newly happened activity in s3.