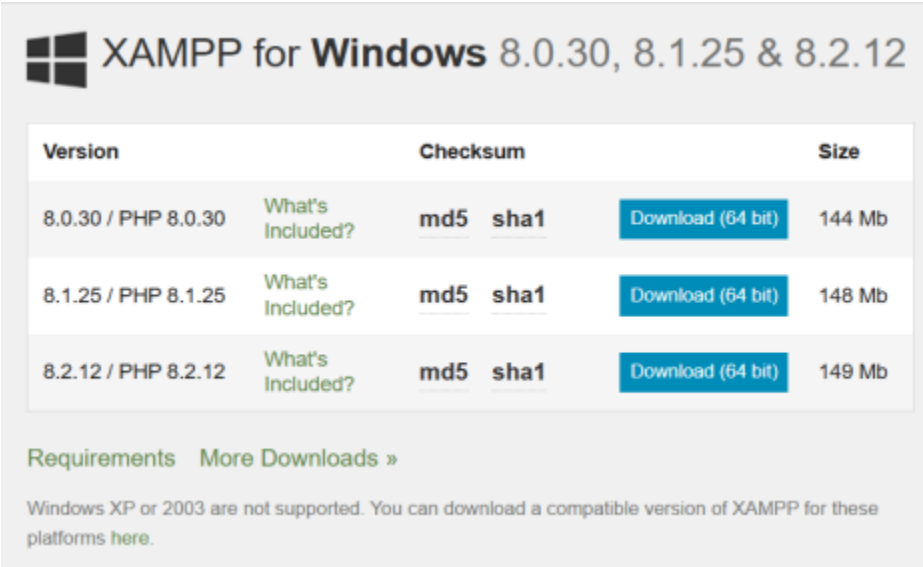
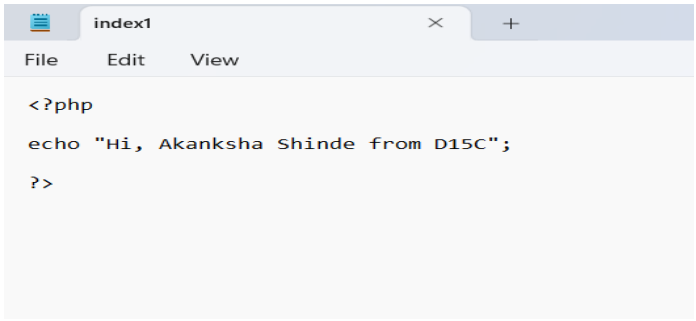
Experiment 1

Part 1 : a) To develop a website and host it on your local machine on a VM

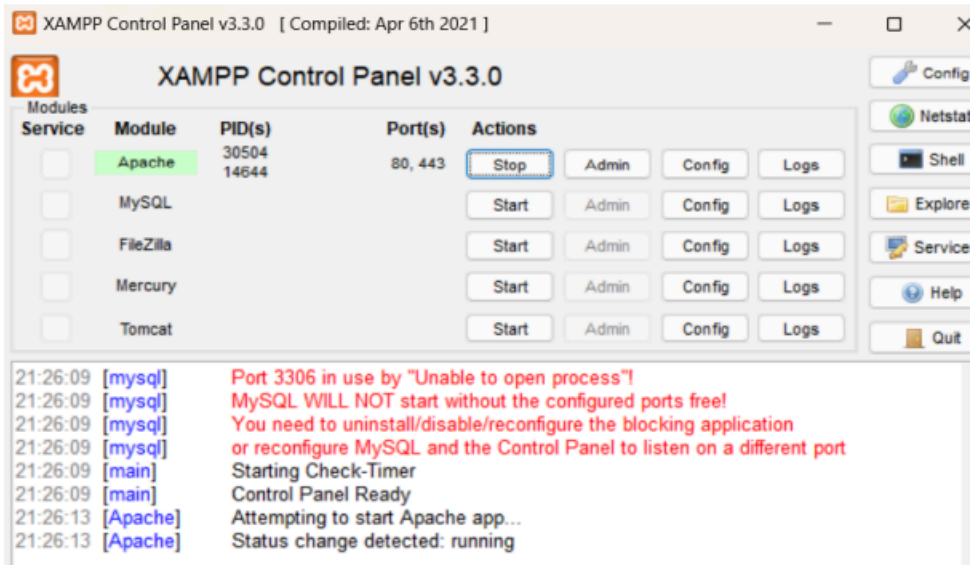
Download xampp from this website https://www.apachefriends.org/download.html



After downloading just select Apache and MySQL you can skip other options as it is not required.
Now go in the xampp folder and inside the htdocs file create a php file just like this

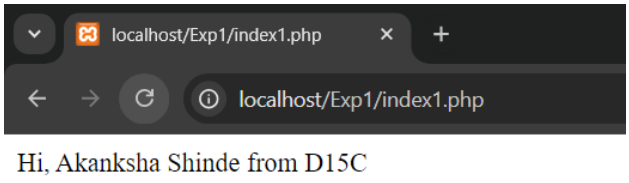


Save it and go to the xampp control panel and start the Apache and MySQL



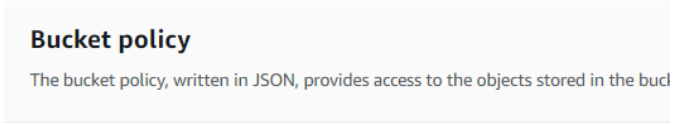
Go in the browser and type localhost and hit enter you will get to see the file you created like in this case index1.php

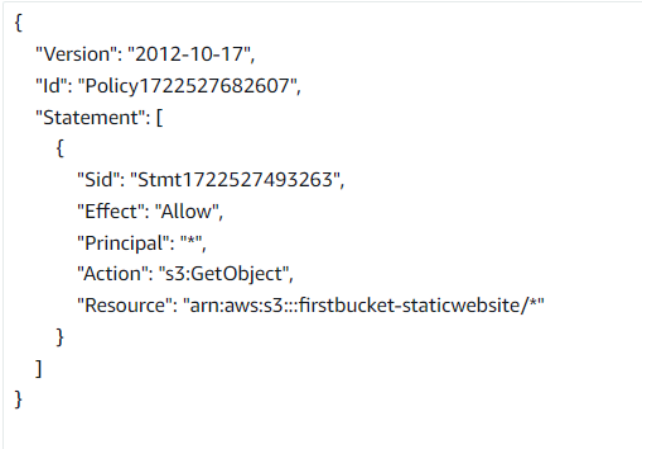
Click on the index1.php and the website is hosted locally.



Hosting a static website on Amazon S3

After creating the bucket on AWS add the file in it that is to be hosted and configure the visibility as public and add /* at the end of the resource key.

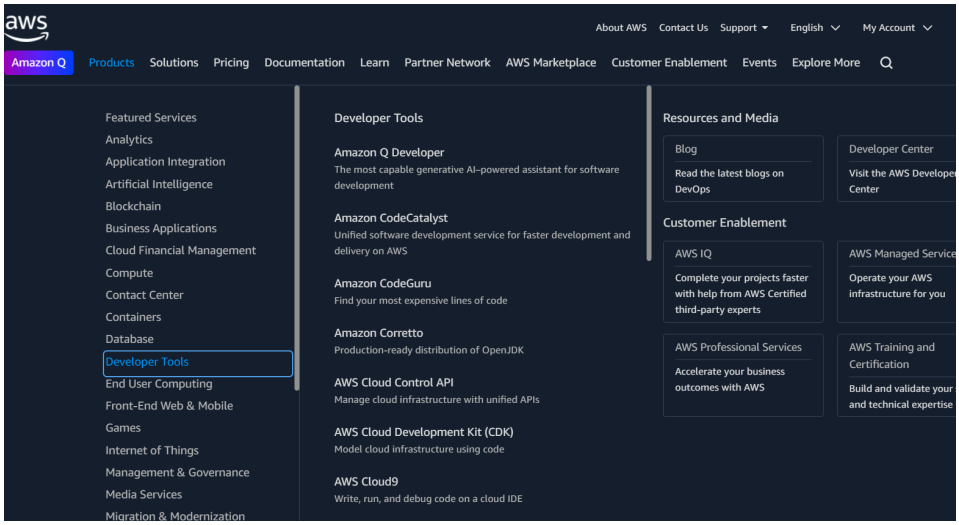




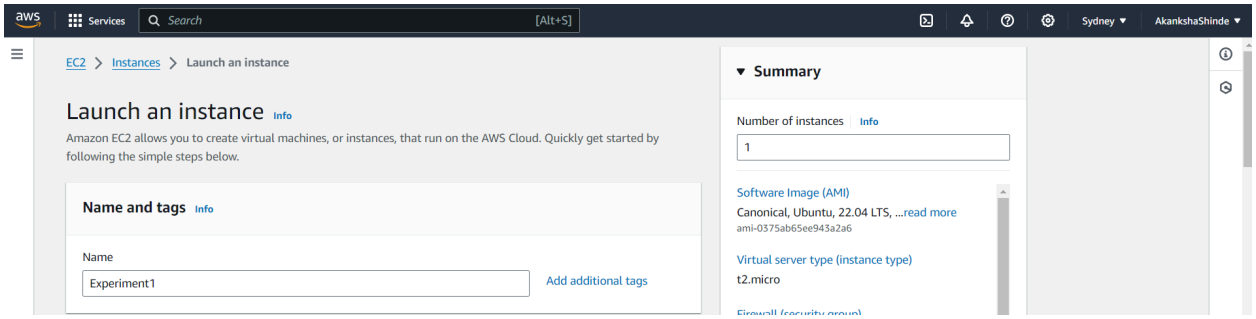
Copy the url from the Bucket ARN and paste it in the browser and the static website is hosted.

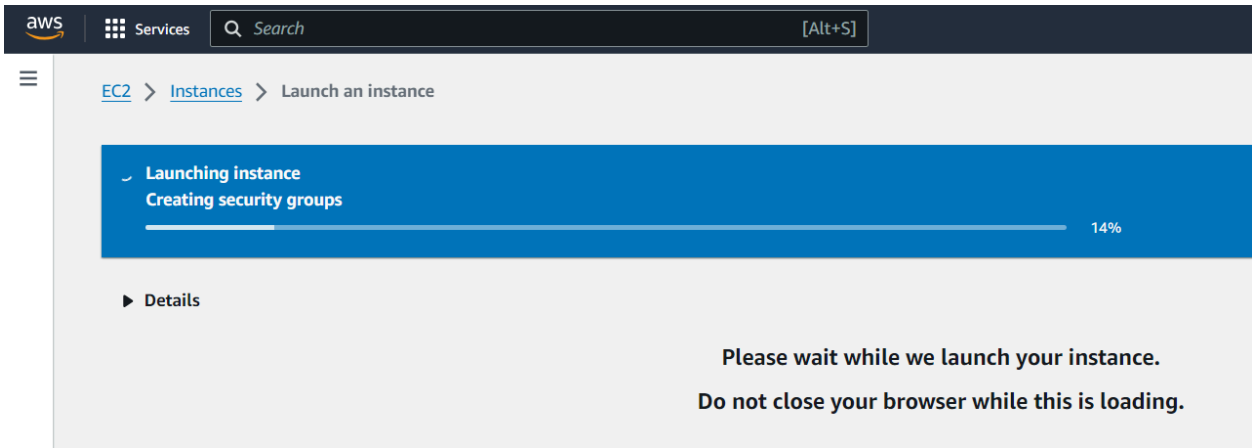
To Create EC2 instance

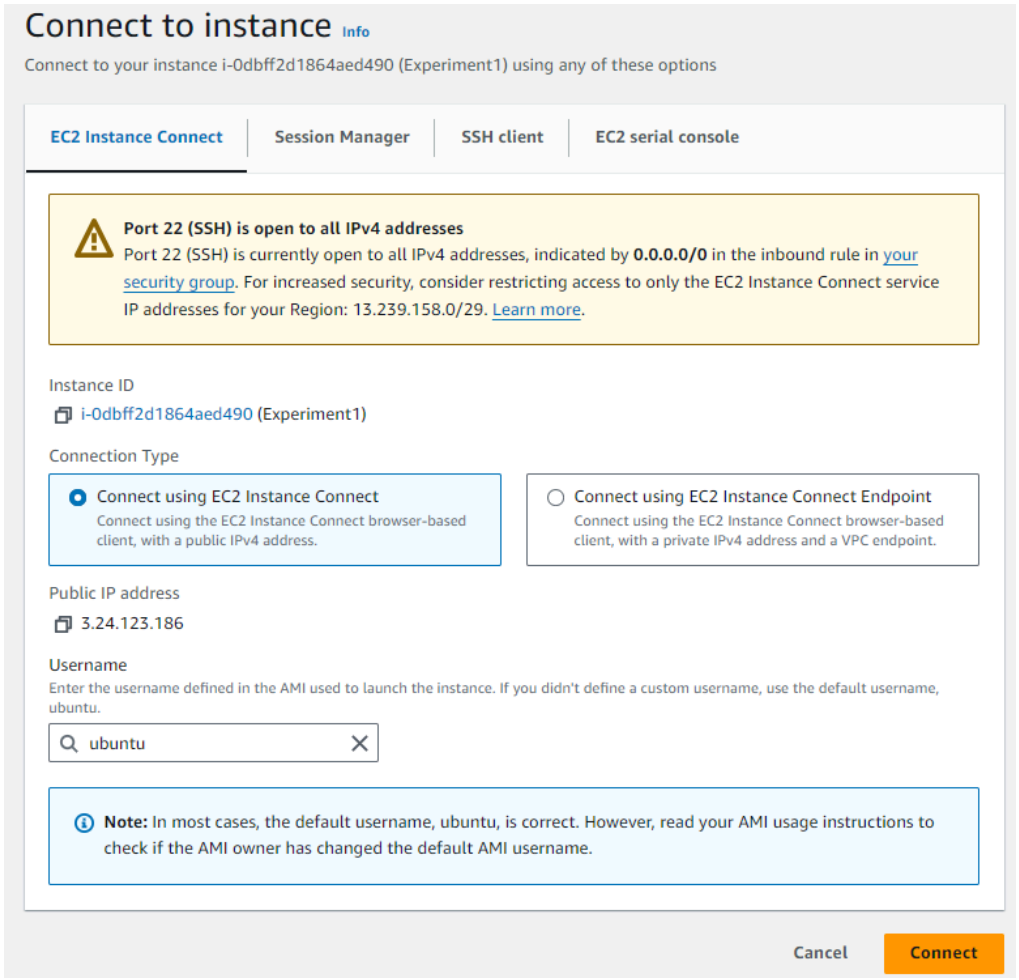
Go to AWS services and search for developer tools



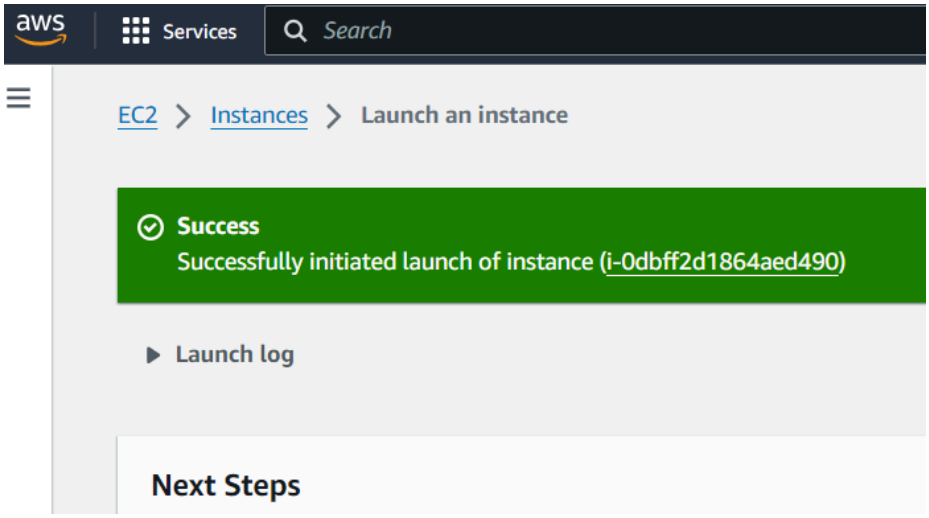
Search for EC2 instance and launch an instance



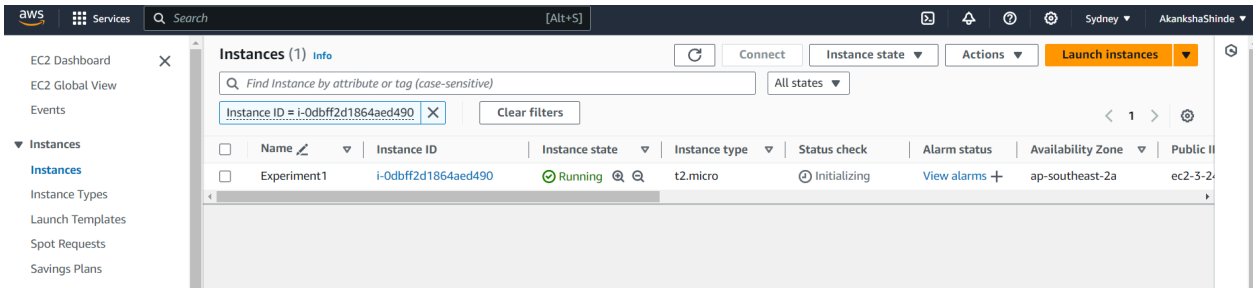




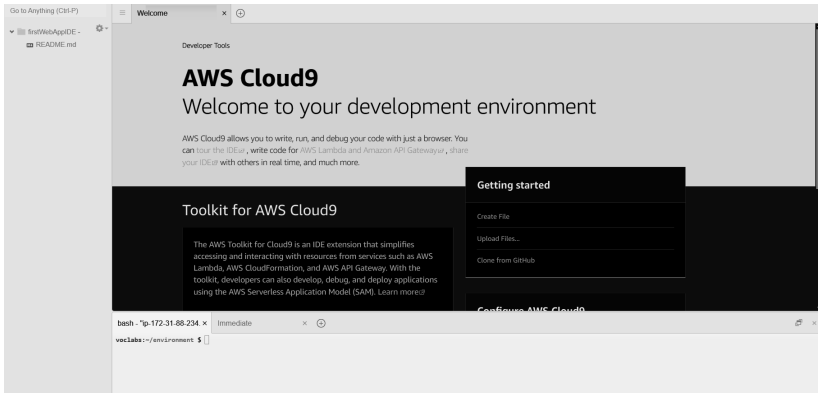

```
  System load:  0.03              Processes:             107
  Usage of /:   6.5% of 24.05GB   Users logged in:       0
  Memory usage: 21%               IPv4 address for eth0: 172.31.15.82
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-15-82:~$
```

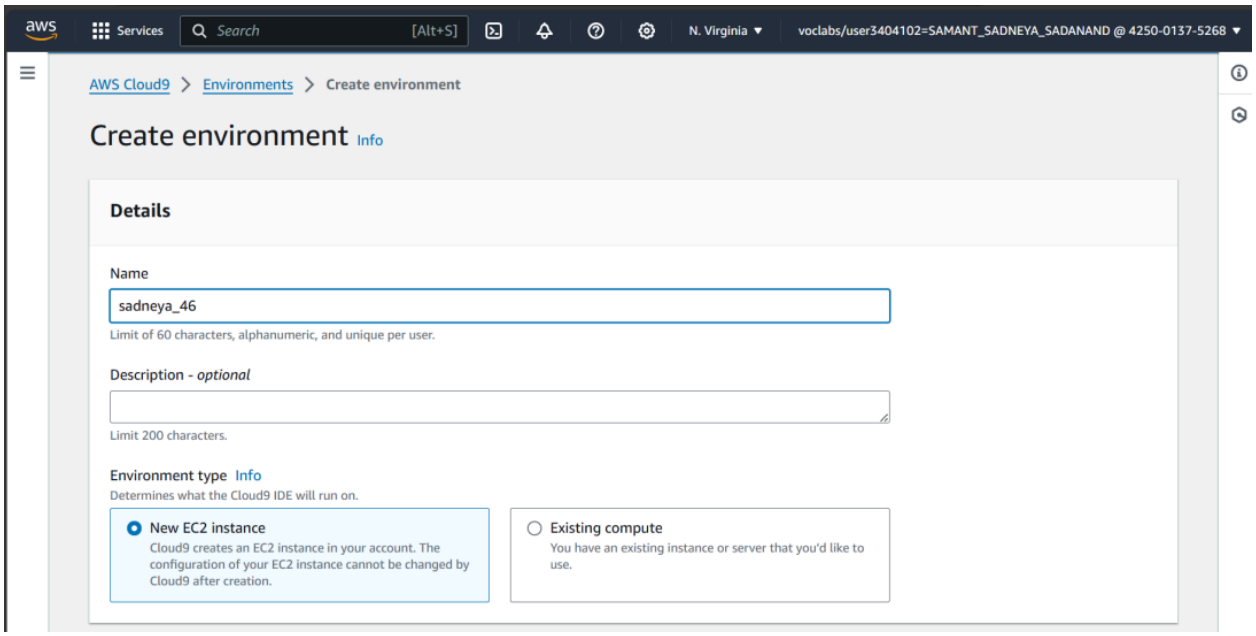


An instance would be recorded with the name you created.



Experiment 1 B





New EC2 instance

Instance type Info

The memory and CPU of the EC2 instance that will be created for Cloud9 to run on.

- (●) **t2.micro (1 GiB RAM + 1 vCPU)**
  Free-tier eligible. Ideal for educational users and exploration.
- ( ) **t3.small (2 GiB RAM + 2 vCPU)**
  Recommended for small web projects.
- ( ) **m5.large (8 GiB RAM + 2 vCPU)**
  Recommended for production and most general-purpose development.
- ( ) **Additional instance types**
  Explore additional instances to fit your need.

Platform Info

This will be installed on your EC2 instance. We recommend Amazon Linux 2023.

Amazon Linux 2023 ▼

Timeout

How long Cloud9 can be inactive (no user input) before auto-hibernating. This helps prevent unnecessary charges.

30 minutes ▼

Network settings Info

Connection

How your environment is accessed.

- (●) **AWS Systems Manager (SSM)**
  Accesses environment via SSM without opening inbound ports (no ingress).
- ( ) **Secure Shell (SSH)**
  Accesses environment directly via SSH, opens inbound ports.

▶ VPC settings Info

▼ Tags - *optional* Info

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Network settings Info

Connection

How your environment is accessed.

- ○ AWS Systems Manager (SSM)
  Accesses environment via SSM without opening inbound ports (no ingress).
- ◉ Secure Shell (SSH)
  Accesses environment directly via SSH, opens inbound ports.

▼ VPC settings Info

Amazon Virtual Private Cloud (VPC)

The VPC that your environment will access. To allow the AWS Cloud9 environment to connect to its EC2 instance, attach an internet gateway (IGW) to your VPC. Create new VPC

vpc-051bba342b3626898
Name -

Subnet

Used to setup your VPC configuration. To use a private subnet, select AWS Systems Manager (SSM) as the connection type. Create new subnet

No preference
Uses default subnet in any Availability Zone

AWS Cloud9

My environments

Shared with me

All account environments

Documentation

Creating sadneya_46. This can take several minutes. While you wait, see Best practices for using AWS Cloud9

For capabilities similar to AWS Cloud9, explore AWS Toolkits in your own IDE and AWS CloudShell in the AWS Management Console.

Learn more

AWS Cloud9 > Environments

Environments (1) Delete | View details | Open in Cloud9 | Create environment

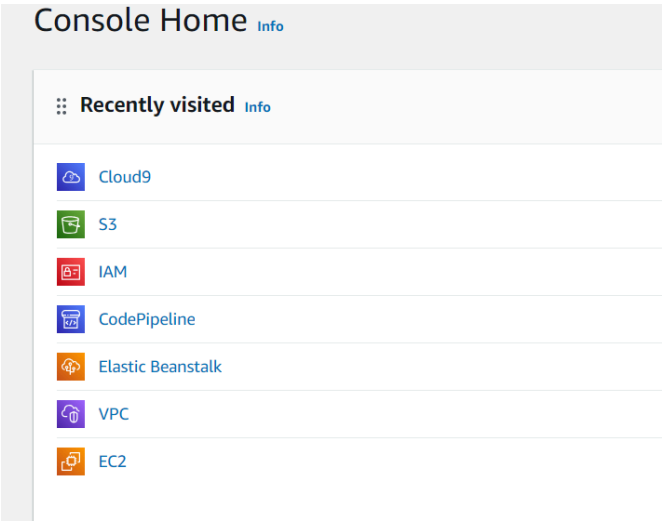
My environments

1

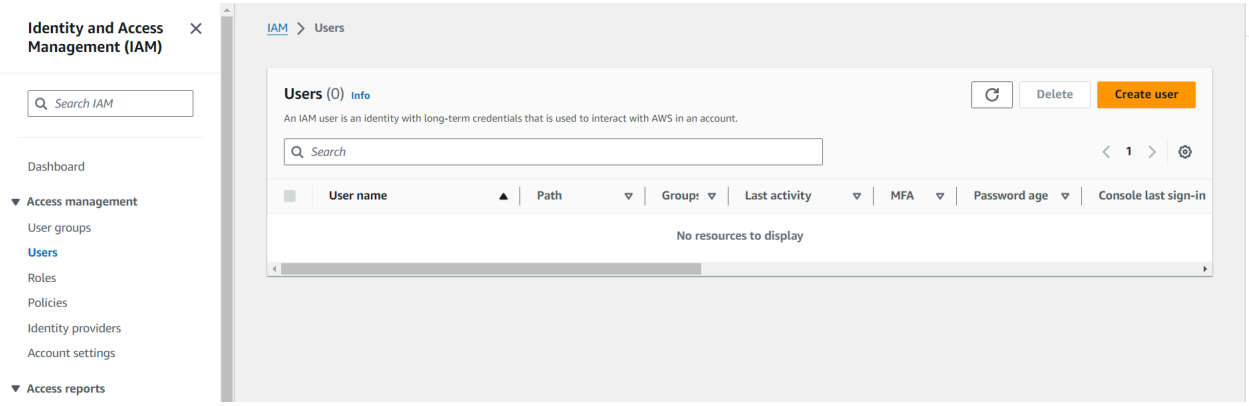
	Name	Cloud9 IDE	Environment type	Connection	Permission	Owner ARN
○	sadneya_46	Open	EC2 instance	Secure Shell (SSH)	Owner	arn:aws:sts::42500137526 role/voclabs/user3404102=S/

IAM

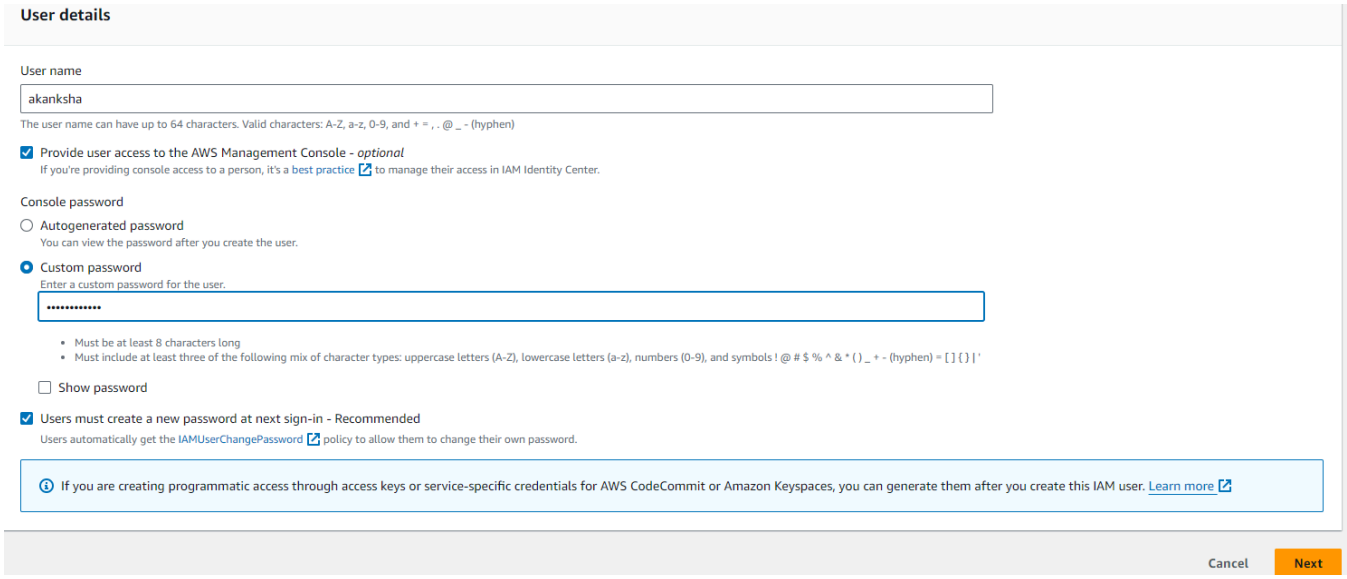
Go to IAM services



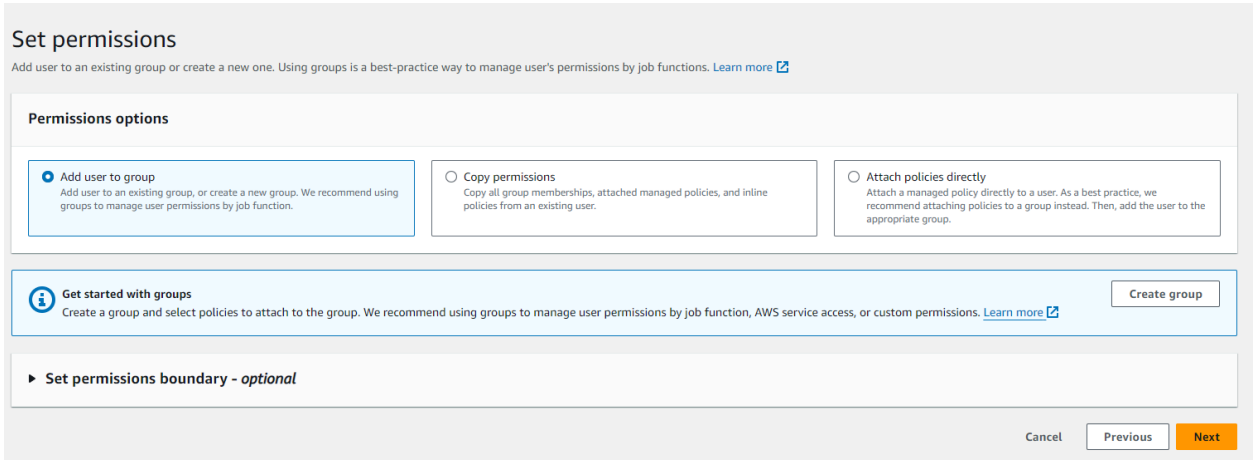
Click on create user



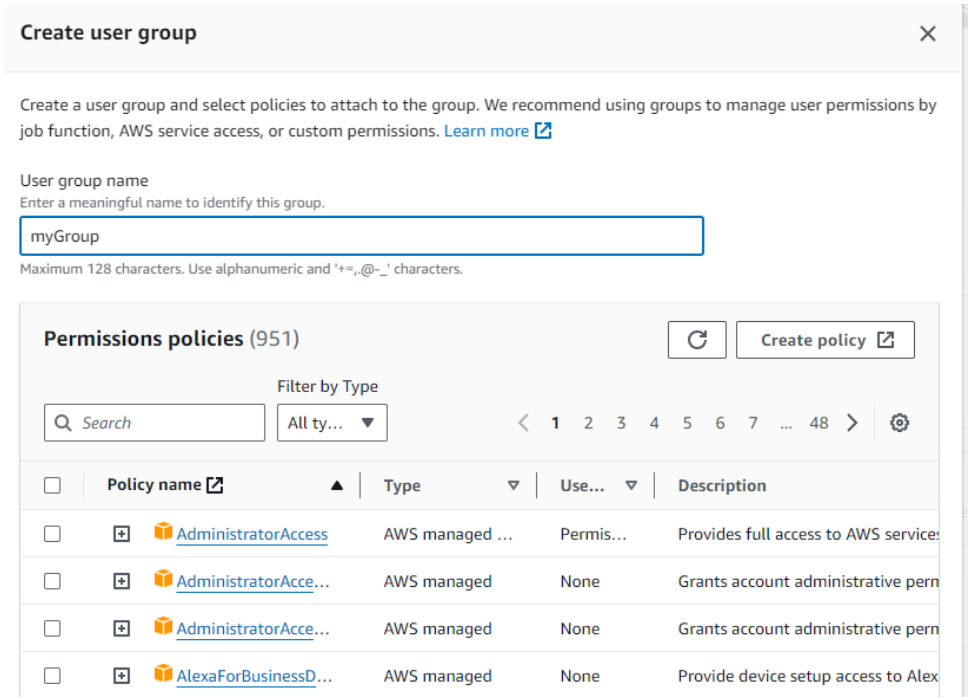
Give username and set password



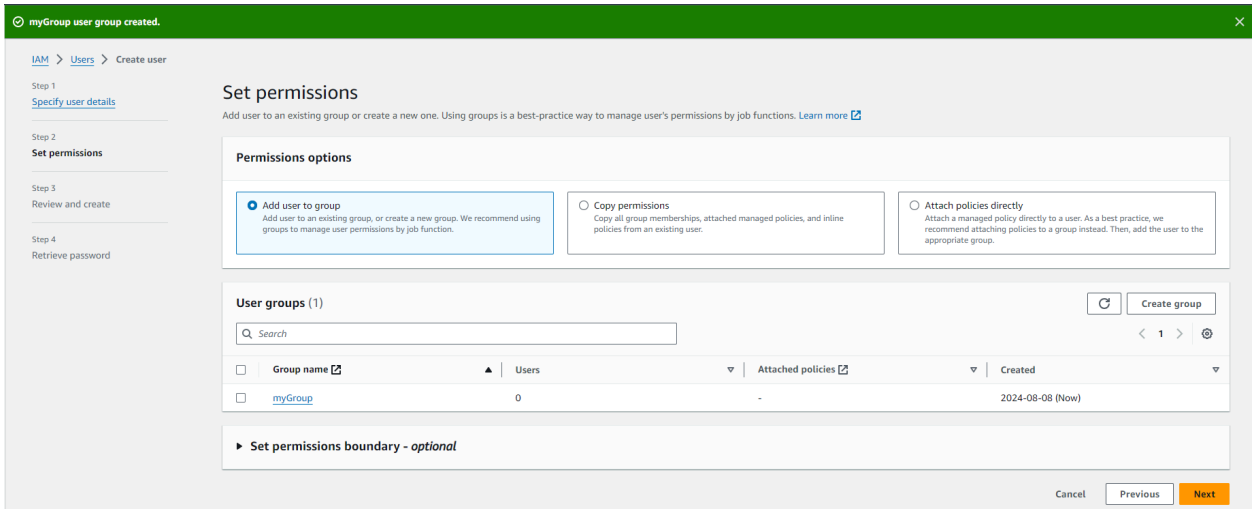
Set permissions as below



Create user group



You must see the below screen



Experiment No.: 6

A. Creating docker image using terraform

Prerequisite:
1) Download and Install Docker Desktop from https://www.docker.com/

Step 1: Check the docker functionality

```
C:\Users\Student.VESIT505-04>docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
  run         Create and run a new container from an image
  exec        Execute a command in a running container
  ps          List containers
  build       Build an image from a Dockerfile
  pull        Download an image from a registry
  push        Upload an image to a registry
  images      List images
  login       Log in to a registry
  logout      Log out from a registry
  search      Search Docker Hub for images
  version     Show the Docker version information
  info        Display system-wide information
```

```
PS C:\Users\Student.VESIT505-04> docker --version
Docker version 24.0.6, build ed223bc
PS C:\Users\Student.VESIT505-04>
```

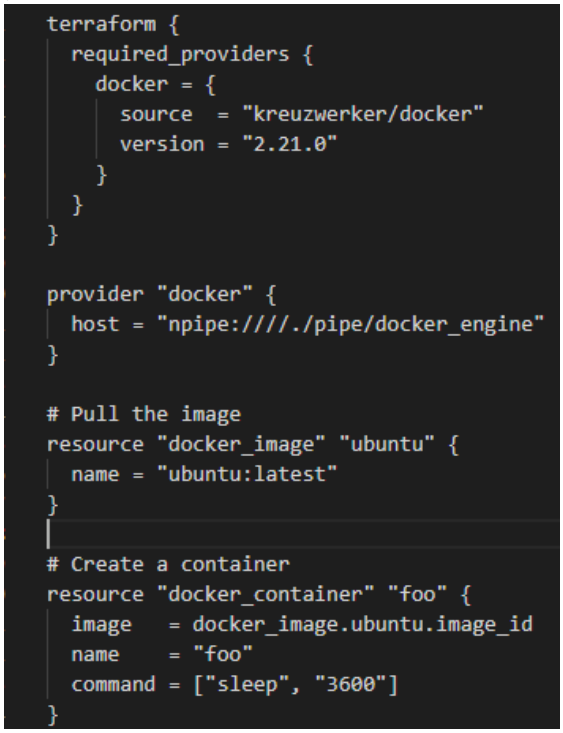
Now, create a folder named 'Terraform Scripts' in which we save our different types of scripts which will be further used in this experiment.

Step 2: Firstly create a new folder named 'Docker' in the 'TerraformScripts' folder. Then create a new docker.tf file using Atom editor and write the followingcontents into it to create a Ubuntu Linux container.

Script:

```
terraform
  { required_providers
  {docker = {
    source = "kreuzwerker/docker"
```

```
    version = "2.21.0"
   }
  }
}

provider "docker" {
 host = "npipe:////.//pipe//docker_engine"
}

# Pulls the image
resource "docker_image" "ubuntu"
  {name = "ubuntu:latest"
}

# Create a container
resource "docker_container" "foo"
  { image =
 docker_image.ubuntu.image_idname =
 "foo"
}
```

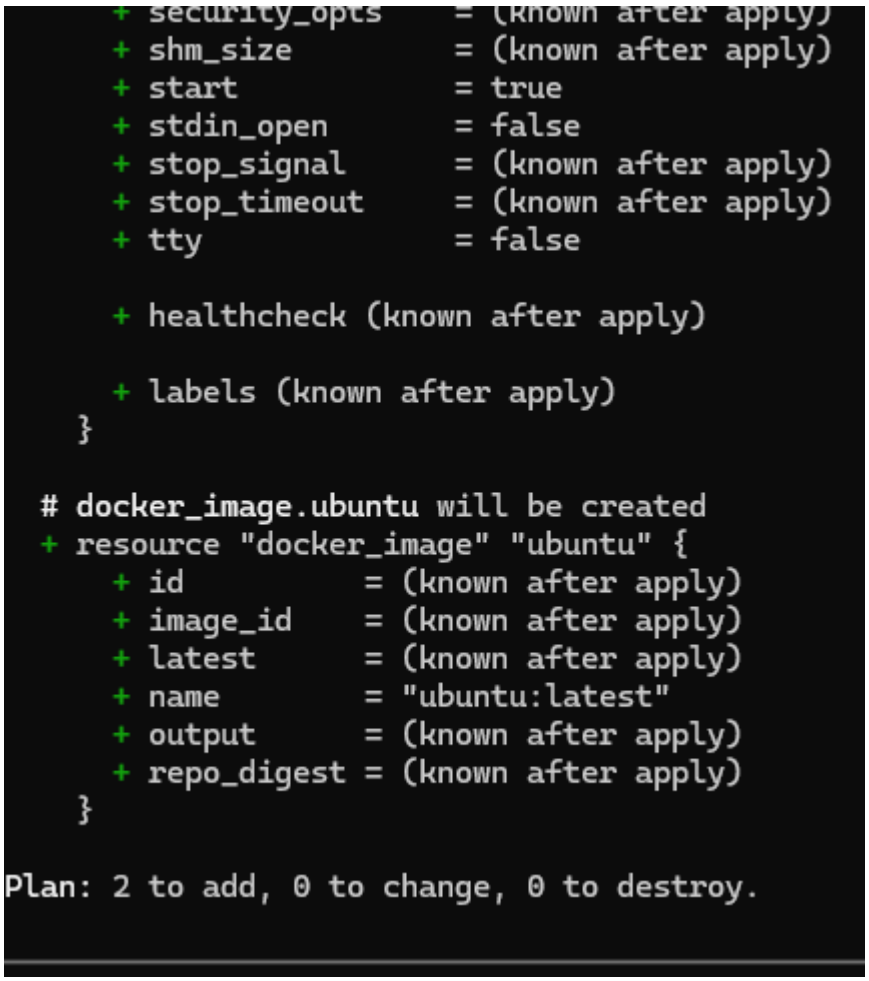


Step 3: Execute Terraform Init command to initialize the resources

```
PS C:\Users\Student.VESIT505-04\documents\terraformScripts\Docker> terraform in
Initializing the backend...
Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "2.21.0"...
- Installing kreuzwerker/docker v2.21.0...
- Installed kreuzwerker/docker v2.21.0 (self-signed, key ID BD080C4571C6104C)
Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html
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.
PS C:\Users\Student.VESIT505-04\documents\terraformScripts\Docker>
```

Step 4: Execute Terraform plan to see the available resources

```
PS C:\Users\INFT505-20\documents\terraformScripts\Docker> terraform plan

Terraform used the selected providers to generate the following execution
with the following symbols:
  + create

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = (known after apply)
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = (known after apply)
      + init             = (known after apply)
      + ip_address       = (known after apply)
```



Step 5: Execute Terraform apply to apply the configuration, which will automatically create and run the Ubuntu Linux container based on our configuration. Using command : "**terraform apply"**

```
PS C:\Users\INFT505-20\documents\terraformScripts\Docker> terraform apply

Terraform used the selected providers to generate the following execution
with the following symbols:
  + create

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = (known after apply)
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = (known after apply)
```

Run Docker images, Before Executing Apply step:

```
PS C:\Users\INFT505-20\documents\terraformScripts\Docker> docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
PS C:\Users\INFT505-20\documents\terraformScripts\Docker>
```

Docker images, After Executing Apply step:

```
PS C:\Users\INFT505-20\Documents\terraformScripts\Docker> docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    edbfe74c41f8   3 weeks ago   78.1MB
PS C:\Users\INFT505-20\Documents\terraformScripts\Docker>
```

Step 6: Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container.

```
Plan: 0 to add, 0 to change, 2 to destroy.

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

  Enter a value: yes

docker_container.foo: Destroying... [id=ea58639e1df08080f14701c6fc53
docker_container.foo: Destruction complete after 0s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
PS C:\Users\INFT505-20\Documents\terraformScripts\Docker>
```

Docker images After Executing Destroy step

```
PS C:\Users\INFT505-20\Documents\terraformScripts\Docker> docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
PS C:\Users\INFT505-20\Documents\terraformScripts\Docker>
```

Name: Akanksha Shinde Class: D15C Roll No: 53

Experiment No 7

Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Theory: Static application security testing (SAST), or static analysis, is a testing methodology that
analyzes source code to find security vulnerabilities that make your organization's applications susceptible to attack. SAST scans an application before the code is compiled. It's also known as white box testing.

What problems does SAST solve?

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being
considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth
guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise.

It's important to note that SAST tools must be run on the application on a regular basis, such as during daily/monthly builds, every time code is checked in, or during a code release.

Why is SAST important?

Developers dramatically outnumber security staff. It can be challenging for an organization to find the resources to perform code reviews on even a fraction of its applications. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally, they are much faster than manual secure code reviews performed by humans. These tools can scan millions of lines of code in a matter of minutes. SAST tools automatically identify critical vulnerabilities—such as buffer overflows, SQL injection, cross-site scripting, and others—with high confidence. Thus, integrating static analysis into the SDLC can yield

dramatic results in the overall quality of the
code developed.

What are the key steps to run SAST effectively?

There are six simple steps needed to perform SAST efficiently in organizations that have a very large number of applications built with different languages, frameworks, and platforms.

1. **Finalize the tool.** Select a static analysis tool that can perform code reviews of applications written in the programming languages you use. The tool should also be able to comprehend the underlying framework used by your software.

2. **Create the scanning infrastructure, and deploy the tool.** This step involves handling the licensing requirements, setting up access control and authorization, and procuring the resources required (e.g., servers and databases) to deploy the tool.

3. **Customize the tool.** Fine-tune the tool to suit the needs of the organization. For example, you might configure it to reduce false positives or find additional security vulnerabilities by writing new rules or updating existing ones. Integrate the tool into the build
environment, create dashboards for tracking scan results, and build custom reports.

4. **Prioritize and onboard applications.** Once the tool is ready, onboard your applications. If you have a large number of applications, prioritize the high-risk applications to scan first. Eventually, all your applications should be onboarded and scanned regularly, with
application scans synced with release cycles, daily or monthly builds, or code check-ins.

5. **Analyze scan results.** This step involves triaging the results of the scan to remove false positives. Once the set of issues is finalized, they should be tracked and provided to the deployment teams for proper and timely remediation.

6. **Provide governance and training.** Proper governance ensures that your development teams are employing the scanning tools properly. The software security touchpoints
should be present within the SDLC. SAST should be incorporated as part of your application development and deployment process.

Integrating Jenkins with SonarQube:

Name: Akanksha Shinde Class: D15C Roll No: 53

Windows installation
Step 1 Install JDK 1.8
Step 2 download and install jenkins
https://www.blazemeter.com/blog/how-to-install-jenkins-on-windows

Ubuntu installation

https://www.digitalocean.com/community/tutorials/how-to-install-java-with-apt on-ubuntu-20-04#installing-the-default-jre-jdk

Step 1 Install JDK 1.8
sudo apt-get install openjdk-8-jre

sudo apt install default-jre

https://www.digitalocean.com/community/tutorials/how-to-install-jenkins-on-ubuntu 20-04

Open SSH

Prerequisites:

- Jenkins installed
- Docker Installed (for SonarQube)

(sudo apt-get install docker-ce=5:20.10.15~3-0~ubuntu-jammy docker-ce-cli=5:20.10.15~3-0~ubuntu-jammy containerd.io docker-compose-plugin)

- SonarQube Docker Image

Steps to integrate Jenkins with SonarQube

1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.

2. Run SonarQube in a Docker container using this command -

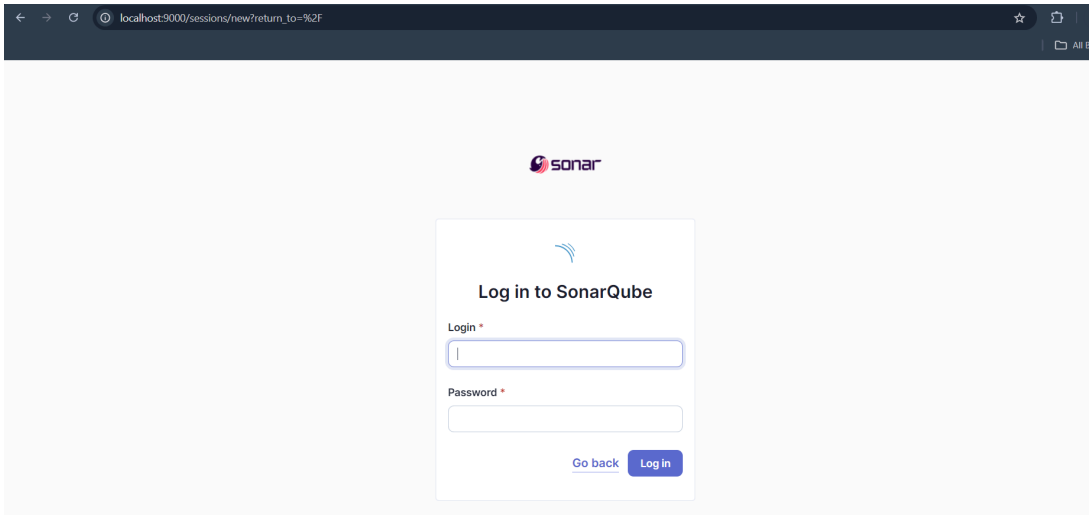
Warning: run below command only once

```
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
```

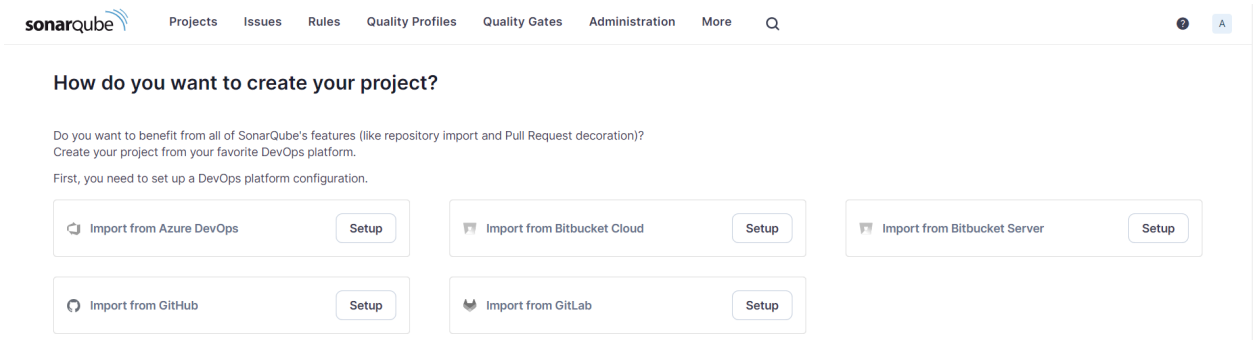
Name: Akanksha Shinde Class: D15C Roll No: 53

```
PS C:\Users\akank> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:lates
t
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
781f61a017c237dcaf6a61148bd90ff1e62e3704f1f80d9ed77c290ac3b2bd32
PS C:\Users\akank>
```

3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.

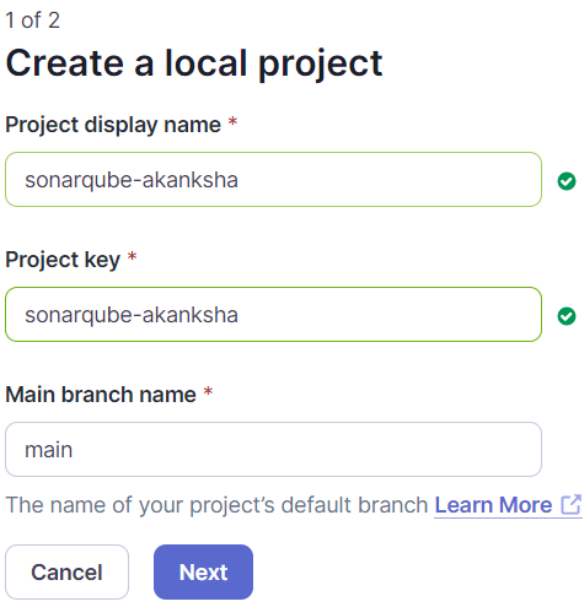


4. Login to SonarQube using username *admin* and password *admin*.

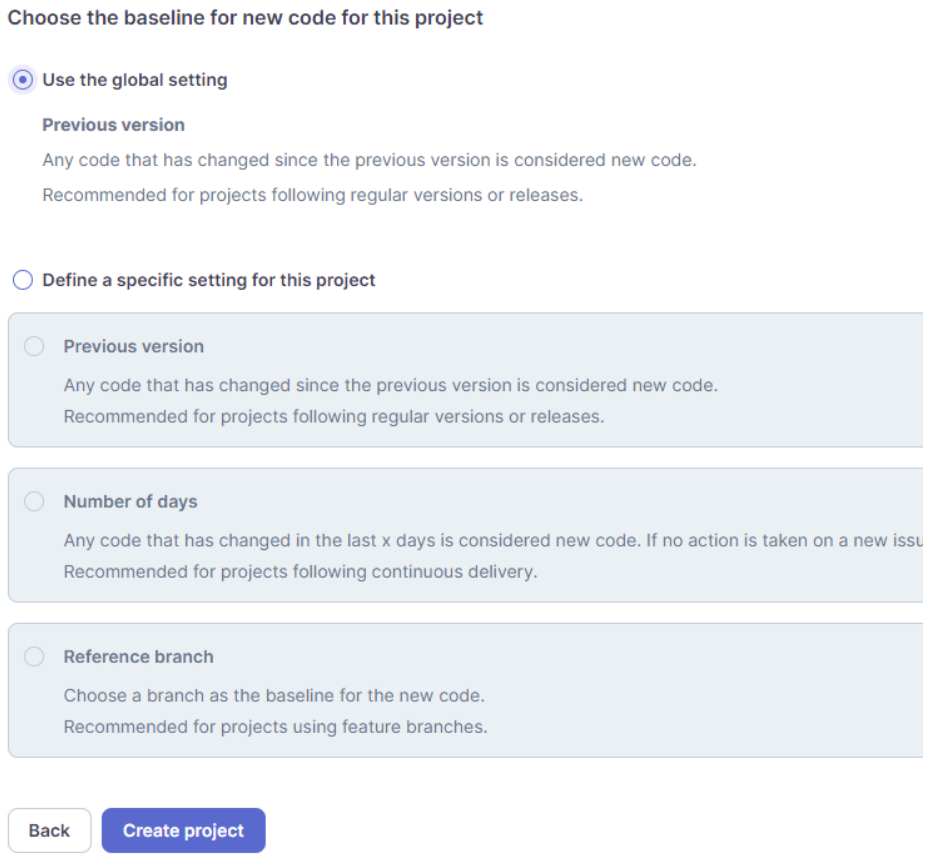


5. Click on create local project which will be somewhere in the left of the screen. Create a manual project in SonarQube with the name **sonarqube**

Name: Akanksha Shinde Class: D15C Roll No: 53



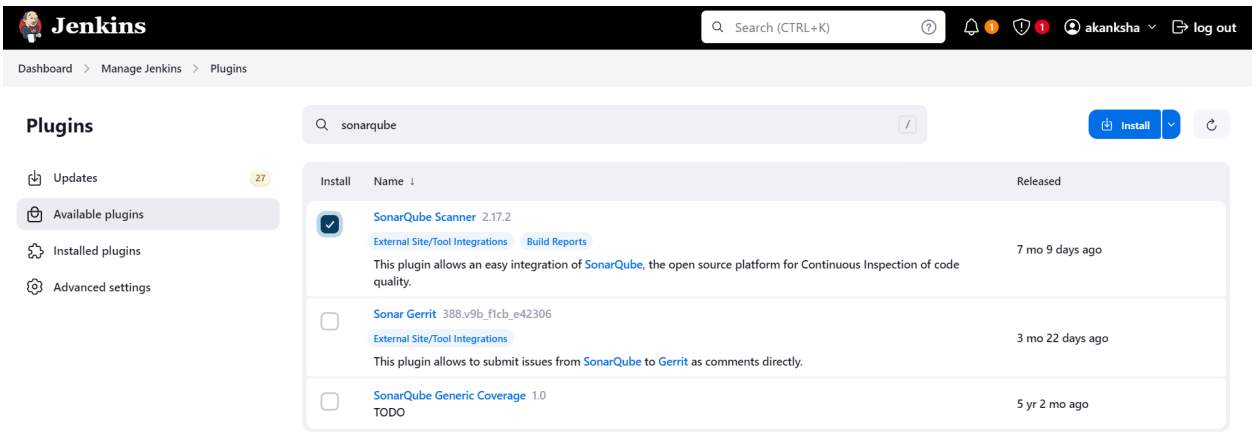
After clicking on next select use the global setting and click create project.



Setup the project and come back to Jenkins Dashboard.

Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.

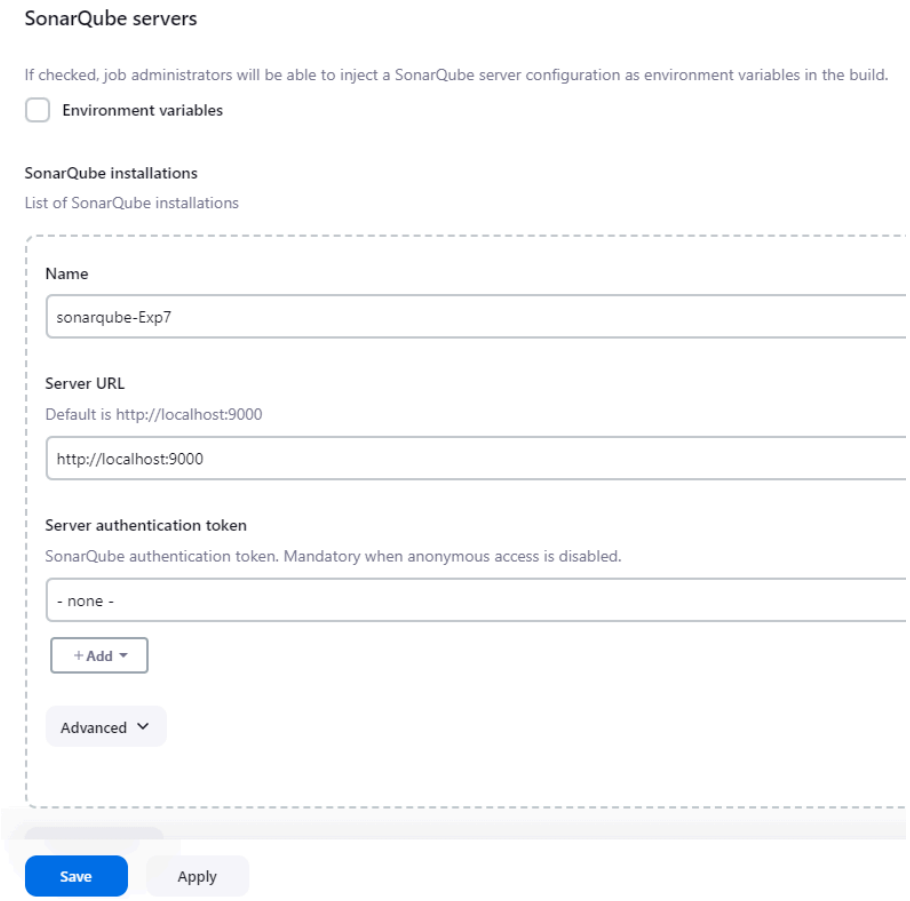
Name: Akanksha Shinde Class: D15C Roll No: 53



6. Go to manage jenkins and click on ‘System’, look for SonarQube Servers and enter details.

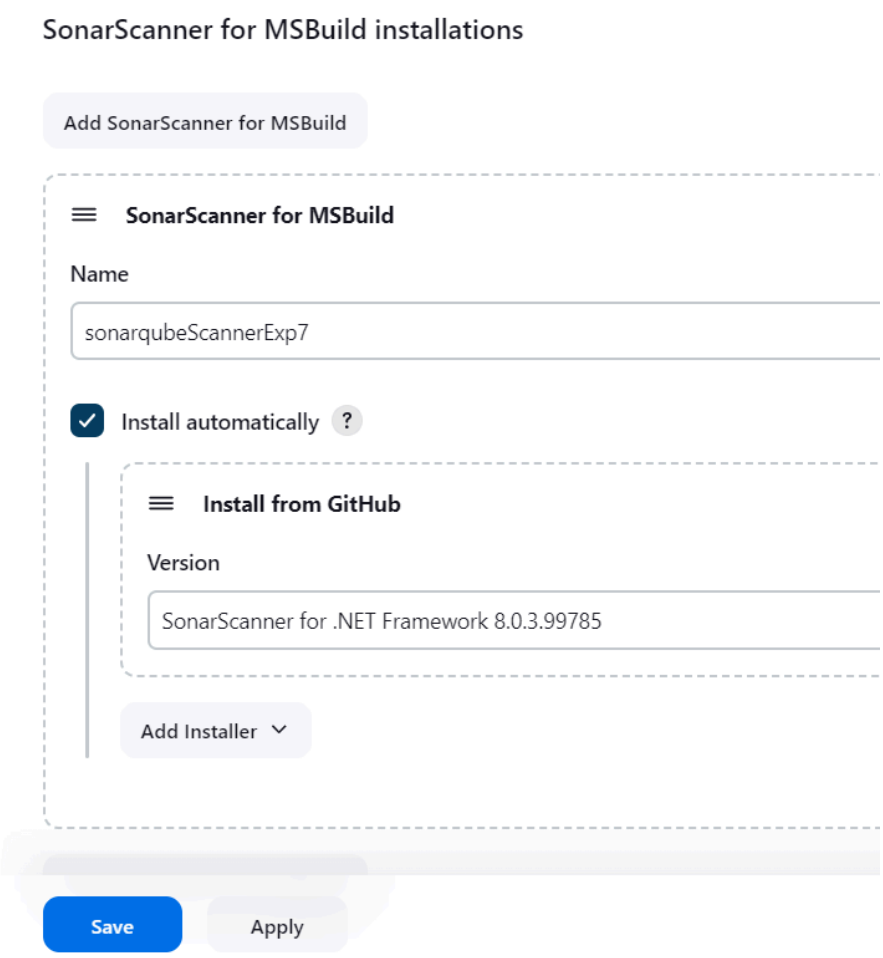
   Enter the Server Authentication token if needed. (not necessary I skipped it).

   Do the following task and click save.

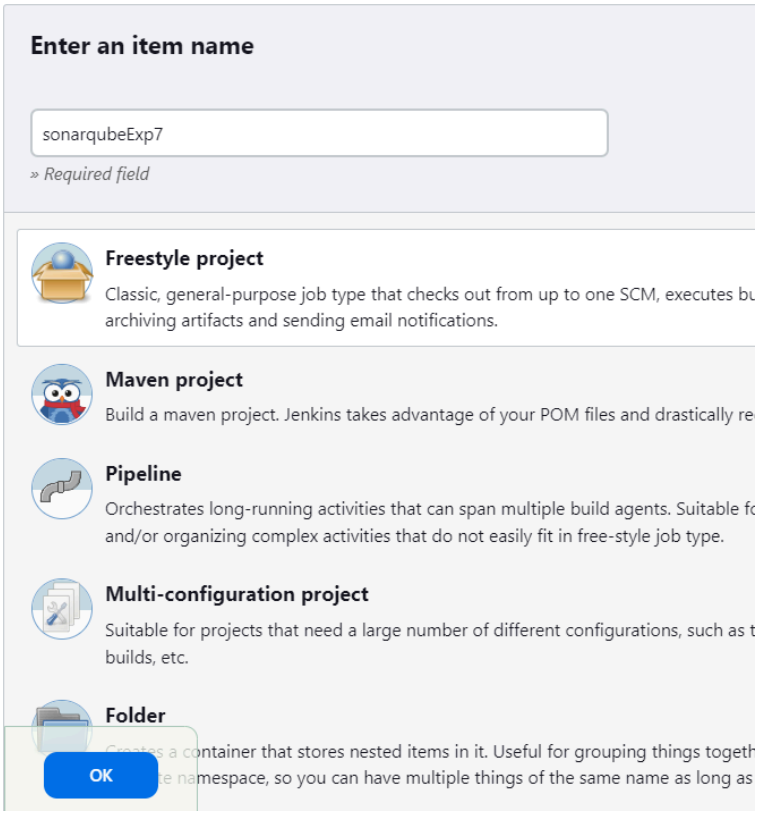


7. Go to manage jenkins and click tools then search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically and click save.

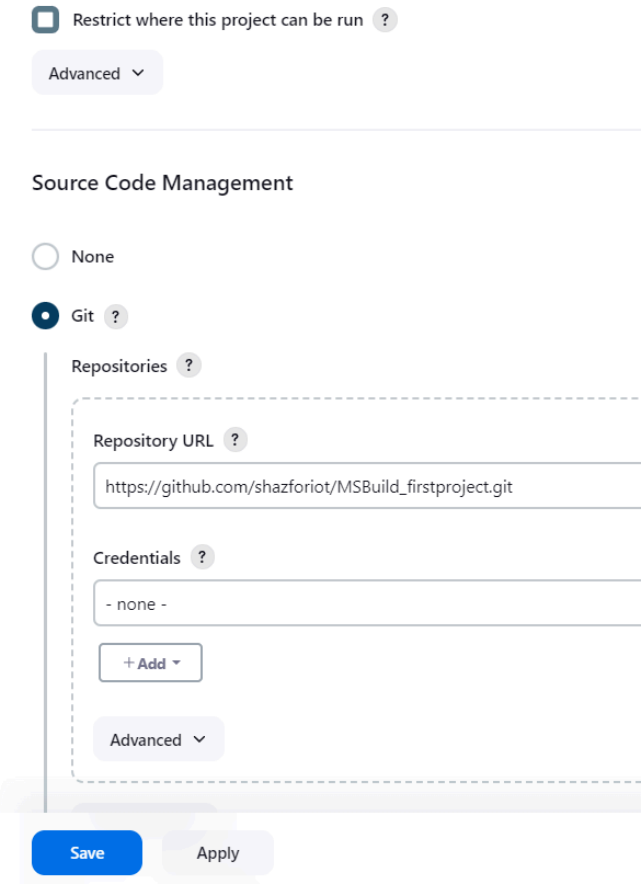
Name: Akanksha Shinde Class: D15C Roll No: 53



8. After the configuration, create a New Item in Jenkins, choose a freestyle project.



9. Choose this GitHub repository in Source Code Management.
https://github.com/shazforiot/MSBuild_firstproject.git
It is a sample hello-world project with no vulnerabilities and issues just to test.



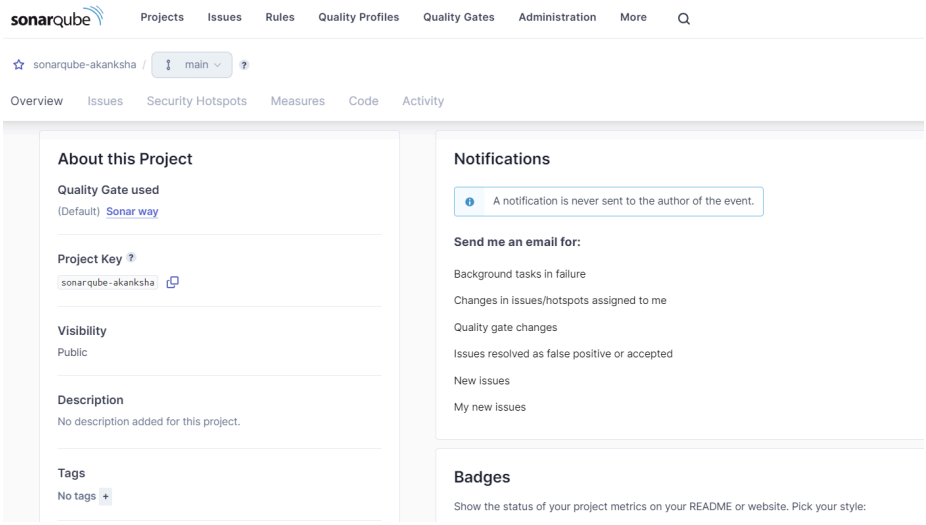
the integration.

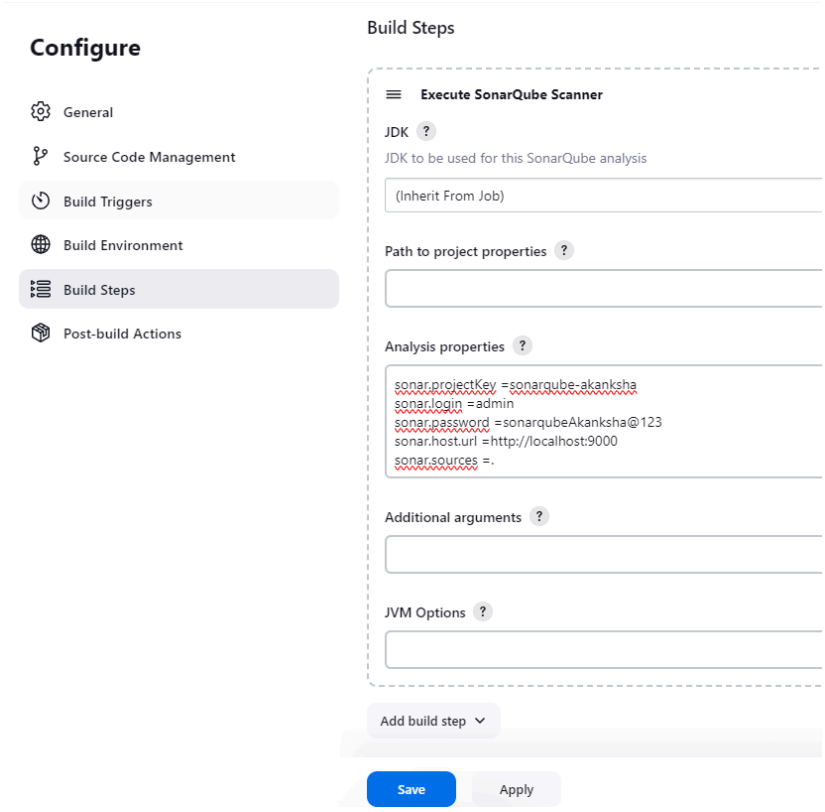
10. Under myProject -> configuration -> Build-> Execute SonarQube Scanner -> enter these Analysis properties(to get this go back to sonarqube and click on your project name after that click project information at the right of the screen).

Mention the SonarQube Project Key, Login, Password, Source path and Host URL. 11. Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user.

Put the following code in analysis properties and put your credentials in it and then apply and save.

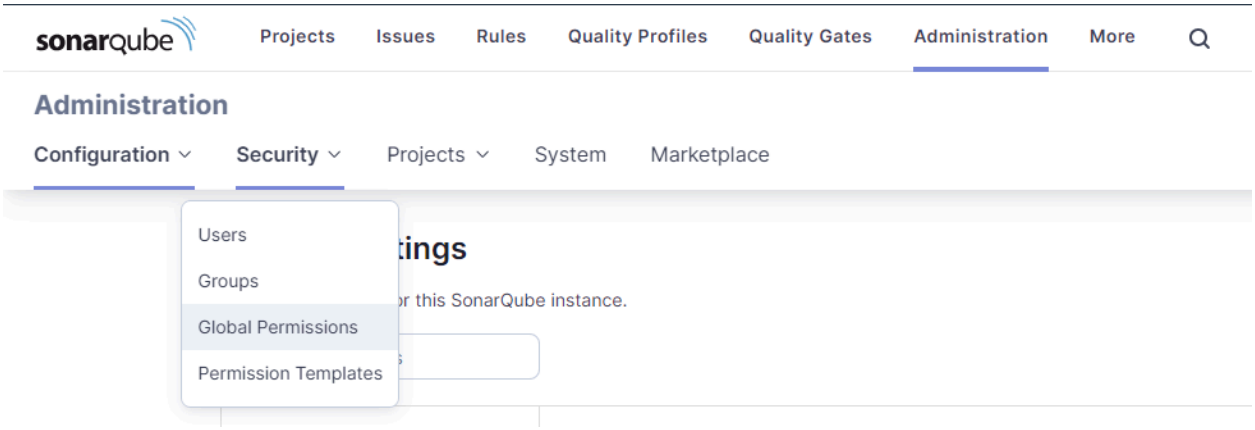
Name: Akanksha Shinde Class: D15C Roll No: 53



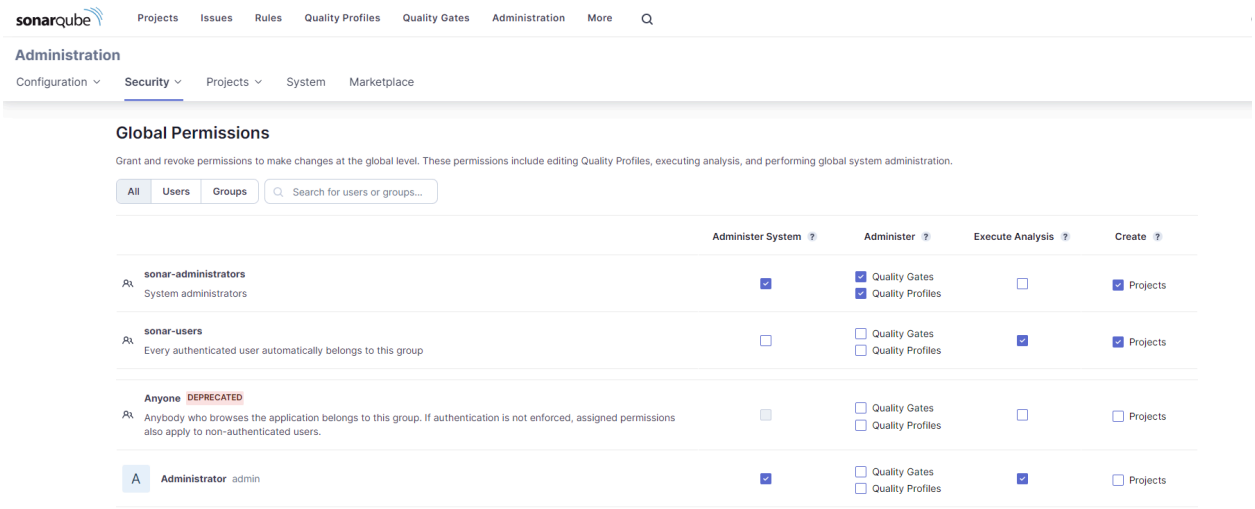


Go to sonarqube and click on administration and click on security and select global permissions from drop down menu

Name: Akanksha Shinde Class: D15C Roll No: 53

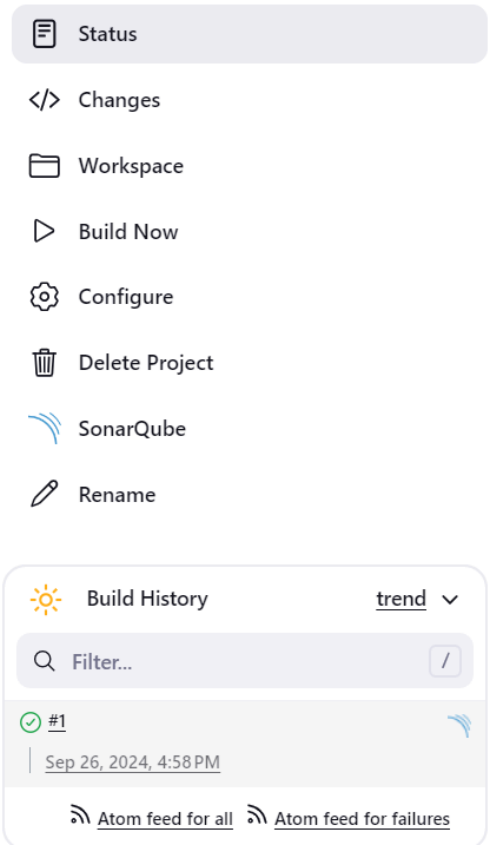


Make sure to check all the checkboxes that I have selected.



12. Go to jenkins and click Build Now.

Name: Akanksha Shinde Class: D15C Roll No: 53



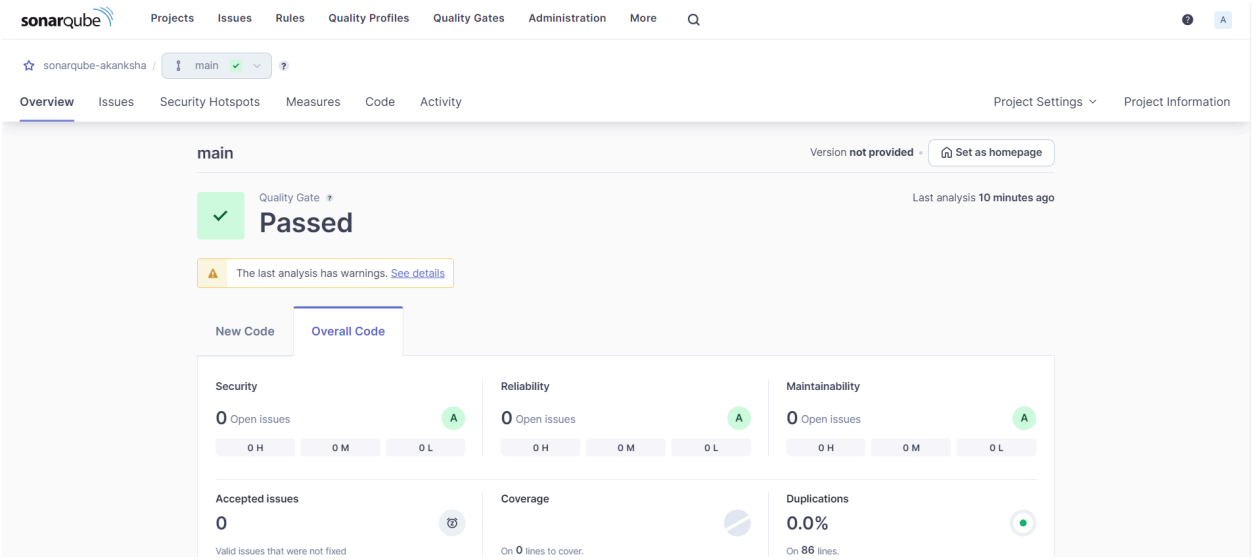
Check the console output.

Console Output

```
Started by user akanksha
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\Exp7
The recommended git tool is: NONE
No credentials specified
 > C:\Program Files\Git\bin\git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\Exp7\.git # timeout=10
Fetching changes from the remote Git repository
 > C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
 > C:\Program Files\Git\bin\git.exe --version # timeout=10
 > git --version # 'git version 2.43.0.windows.1'
 > C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* #
timeout=10
 > C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
 > C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
 > C:\Program Files\Git\bin\git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
First time build. Skipping changelog.
[Exp7] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube-scannerExp7\bin\sonar-scanner.bat -
Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=sonarqube-akanksha -Dsonar.login=admin -Dsonar.host.url=http://localhost:9000 -Dsonar.sources=. -
Dsonar.password=admin123 -Dsonar.projectBaseDir=C:\ProgramData\Jenkins\.jenkins\workspace\Exp7
16:59:00.866 WARN  Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'
16:59:00.874 INFO  Scanner configuration file: C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube-scannerExp7\bin\..\conf\son
```

```
5.x or higher, see https://redirect.sonarsource.com/doc/install-configure-scanner-msbuild.html
16:59:30.166 INFO  Sensor C# [csharp] (done) | time=1ms
16:59:30.166 INFO  Sensor Analysis Warnings import [csharp]
16:59:30.168 INFO  Sensor Analysis Warnings import [csharp] (done) | time=1ms
16:59:30.168 INFO  Sensor C# File Caching Sensor [csharp]
16:59:30.169 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
16:59:30.170 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=1ms
16:59:30.170 INFO  Sensor Zero Coverage Sensor
16:59:30.183 INFO  Sensor Zero Coverage Sensor (done) | time=15ms
16:59:30.188 INFO  SCM Publisher SCM provider for this project is: git
16:59:30.191 INFO  SCM Publisher 4 source files to be analyzed
16:59:30.693 INFO  SCM Publisher 4/4 source files have been analyzed (done) | time=502ms
16:59:30.697 INFO  CPD Executor Calculating CPD for 0 files
16:59:30.697 INFO  CPD Executor CPD calculation finished (done) | time=0ms
16:59:30.705 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
16:59:31.003 INFO  Analysis report generated in 118ms, dir size=201.0 kB
16:59:31.071 INFO  Analysis report compressed in 53ms, zip size=22.4 kB
16:59:31.272 INFO  Analysis report uploaded in 197ms
16:59:31.274 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube-akanksha
16:59:31.275 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
16:59:31.275 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=903576df-83d0-4126-b8a2-6abcd1f0b97e
16:59:31.291 INFO  Analysis total time: 20.478 s
16:59:31.292 INFO  SonarScanner Engine completed successfully
16:59:31.330 INFO  EXECUTION SUCCESS
16:59:31.332 INFO  Total time: 30.459s
Finished: SUCCESS
```

13. Once the build is complete, check the project in SonarQube.



In this way, we have integrated Jenkins with SonarQube for SAST.

Conclusion

To conclude, I have understood the importance of SAST and have successfully integrated Jenkins with SonarQube for Static Analysis and Code Testing. In the first testing didn't go right there was the error of JAVA_HOME exists but not being able to point it. So I solved it by setting the path of java bin in manage jenkins and then global settings again I created a freestyle and project and was successfully able to perform the experiment.

Name: Akanksha Shinde Class: D15C Roll No: 53

Name : Akanksha Shinde Class: D15C Roll No: 53

Experiment No. 8

Aim: 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.

Theory:

What is SAST?

Static application security testing (SAST), or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization's applications susceptible to attack. SAST scans an application before the code is compiled. It's also known as white box testing.

What problems does SAST solve?

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise.

It's important to note that SAST tools must be run on the application on a regular basis, such as during daily/monthly builds, every time code is checked in, or during a code release.

Why is SAST important?

Developers dramatically outnumber security staff. It can be challenging for an organization to find the resources to perform code reviews on even a fraction of its applications. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally, they are much faster than manual secure code reviews performed by humans. These tools can scan millions of lines of code in a matter of minutes. SAST tools automatically identify critical vulnerabilities—such as buffer overflows, SQL injection, cross-site scripting, and others—with high confidence.

Name : Akanksha Shinde Class: D15C Roll No: 53

What is a CI/CD Pipeline?

CI/CD pipeline refers to the Continuous Integration/Continuous Delivery pipeline. Before we dive deep into this segment, let's first understand what is meant by the term ‘pipeline’?

A pipeline is a concept that introduces a series of events or tasks that are connected in a sequence to make quick software releases. For example, there is a task, that task has got five different stages, and each stage has got some steps. All the steps in phase one have to be completed, to mark the latter stage to be complete.

Now, consider the CI/CD pipeline as the backbone of the DevOps approach. This Pipeline is responsible for building codes, running tests, and deploying new software versions. The Pipeline executes the job in a defined manner by first coding it and then structuring it inside several blocks that may include several steps or tasks.

What is SonarQube?

Name : Akanksha Shinde Class: D15C Roll No: 53

SonarQube is an open-source platform developed by SonarSource for continuous inspection of code quality. Sonar does static code analysis, which provides a detailed report of bugs, code smells, vulnerabilities, code duplications.
It supports 25+ major programming languages through built-in rulesets and can also be extended with various plugins.

Benefits of SonarQube

- **Sustainability -** Reduces complexity, possible vulnerabilities, and code duplications, optimising the life of applications.
- **Increase productivity -** Reduces the scale, cost of maintenance, and risk of the application; as such, it removes the need to spend more time changing the code
- **Quality code -** Code quality control is an inseparable part of the process of software development.
- **Detect Errors -** Detects errors in the code and alerts developers to fix them automatically before submitting them for output.
- **Increase consistency -** Determines where the code criteria are breached and enhances the quality
- **Business scaling -** No restriction on the number of projects to be evaluated
- **Enhance developer skills -** Regular feedback on quality problems helps developers to improve their coding skills

Integrating Jenkins with SonarQube:

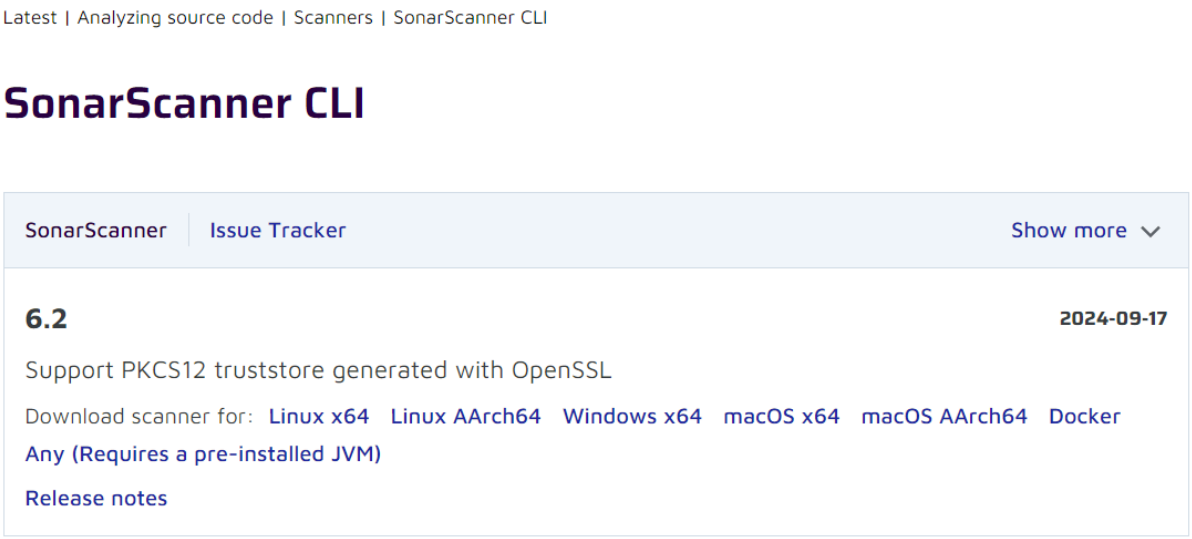
Prerequisites:

- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

Name : Akanksha Shinde Class: D15C Roll No: 53

Steps to create a Jenkins CI/CD Pipeline and use SonarQube to perform SAST

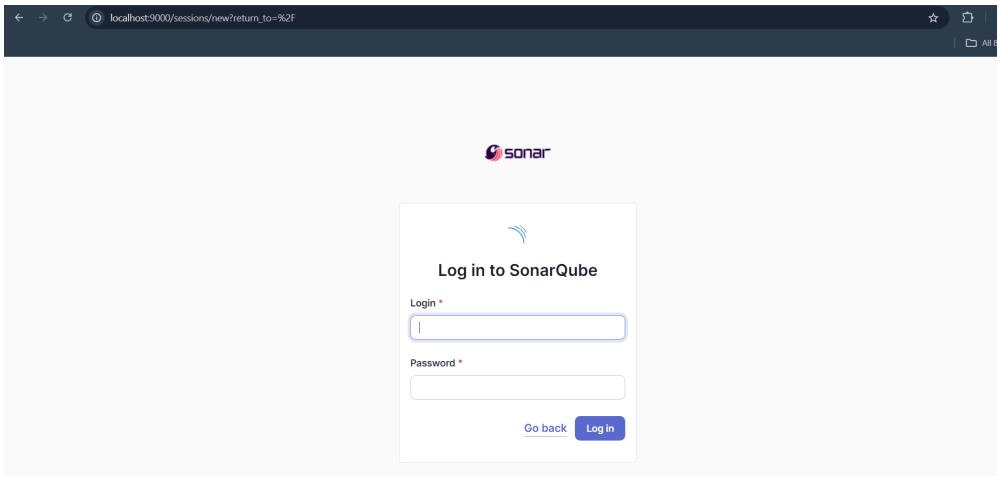
1. **Download sonar scanner and extract the file.**
   https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/



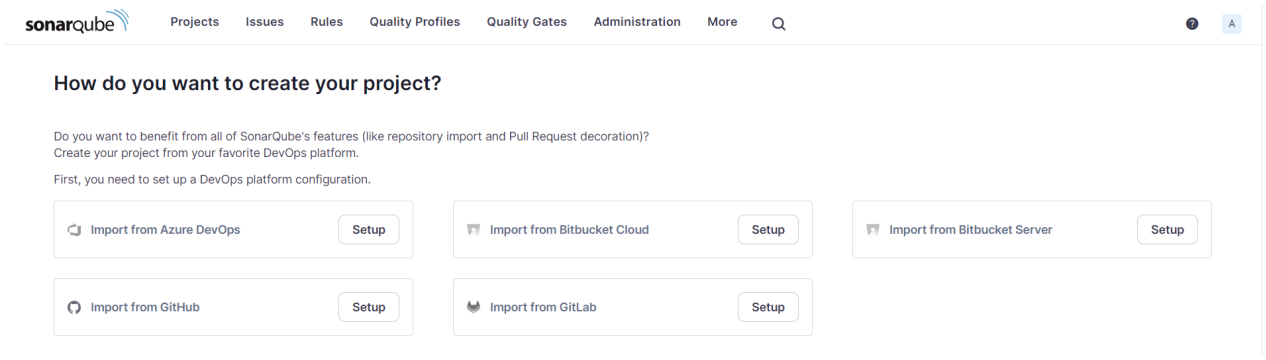
2. Install sonarqube image
Command:
docker pull sonarqube (no need to do it again as we already did it in Exp 7)
Then run the image by running this command
docker run -d -p 9000:9000 sonarqube

3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.

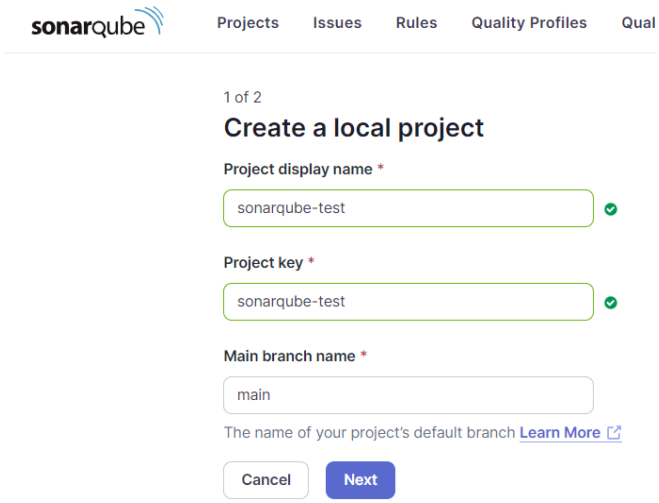
Name : Akanksha Shinde Class: D15C Roll No: 53



4. Login to SonarQube using username and password.

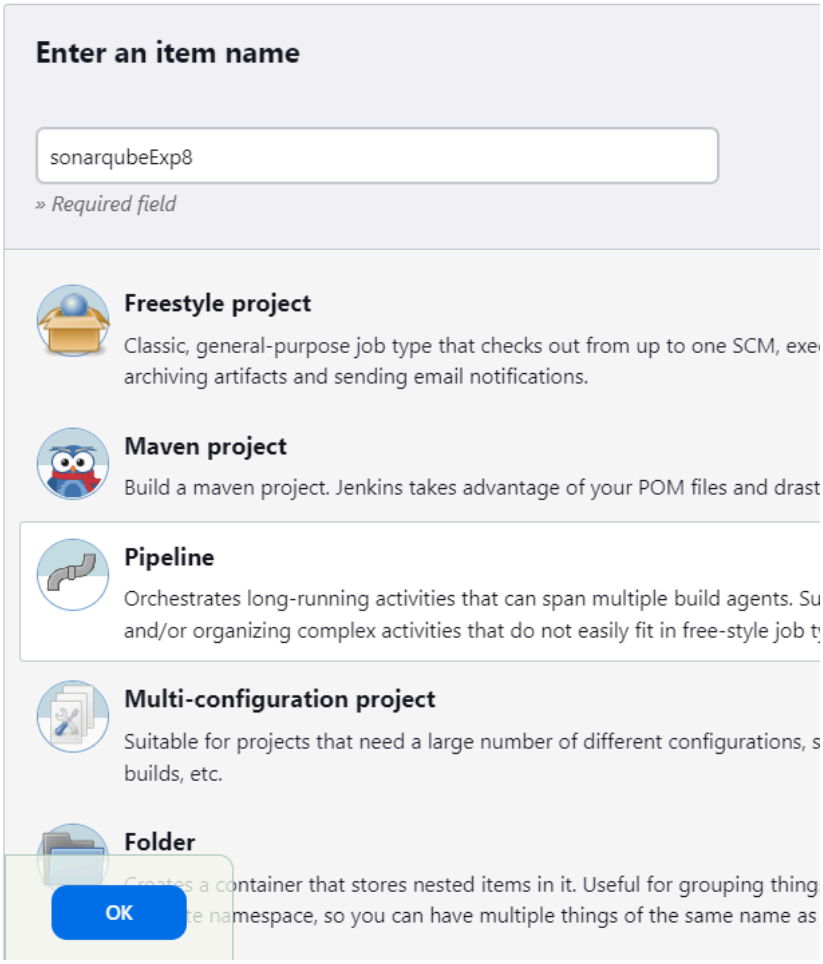


5. Create a manual project in SonarQube with the name **sonarqube-test**



Setup the project and come back to Jenkins Dashboard.

6. Create a New Item in Jenkins, choose **Pipeline**.



7. Under Pipeline Script, enter the following -

```
node {
   stage('Cloning the GitHub Repo') {
      git 'https://github.com/shazforiot/GOL.git'
   }
   stage('SonarQube analysis') {
     withSonarQubeEnv('sonarqube') {
        sh "<PATH_TO_SONARQUBE_FOLDER>//bin//sonar-scanner \
         -D sonar.login=<SonarQube_USERNAME> \
         -D sonar.password=<SonarQube_PASSWORD> \
         -D sonar.projectKey=<Project_KEY> \
-D sonar.exclusions=vendor/**,resources/**,**/*.java \
         -D sonar.host.url=http://127.0.0.1:9000/"
     }
```

```
    }
}
```

Pipeline

Definition

Pipeline script

Script

```
node {
    stage('Cloning the GitHub Repo') {
        git 'https://github.com/shazforiot/GOL.git'
    }

    stage('SonarQube Analysis') {
        withSonarQubeEnv('sonarqubeExp7') {
            bat """
                "C:\\Users\\akank\\Downloads\\sonar-scanner-6.2.0.4584-windows-x64\\bin\\sonar-scanner.bat" ^
                -Dsonar.login=admin ^
                -Dsonar.password=admin123 ^
                -Dsonar.projectKey=sonarqube-test ^
                -Dsonar.exclusions=vendor/**,resources/**,.java ^
                -Dsonar.host.url=http://localhost:9000
            """
        }
    }
}
```

Use Groovy Sandbox

Pipeline Syntax

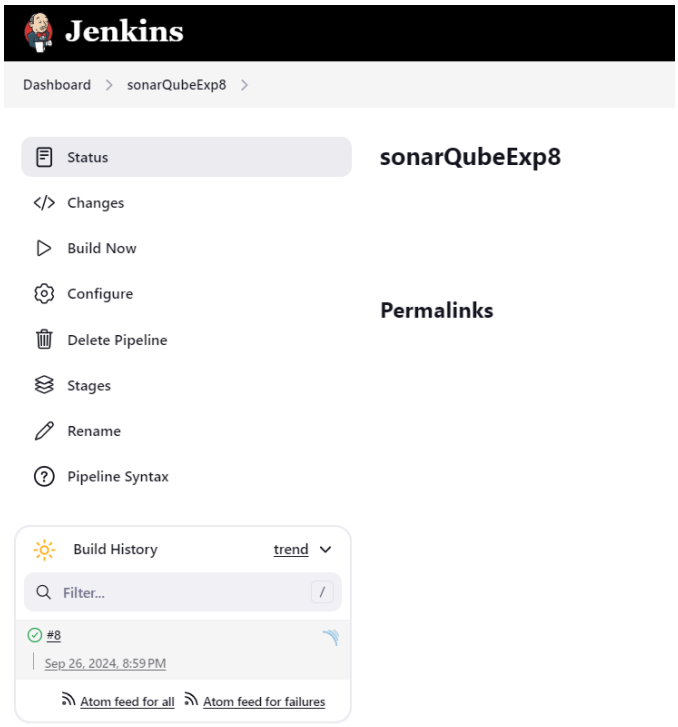
Save Apply

Above put your login, password, projectKey just like did it in exp7.

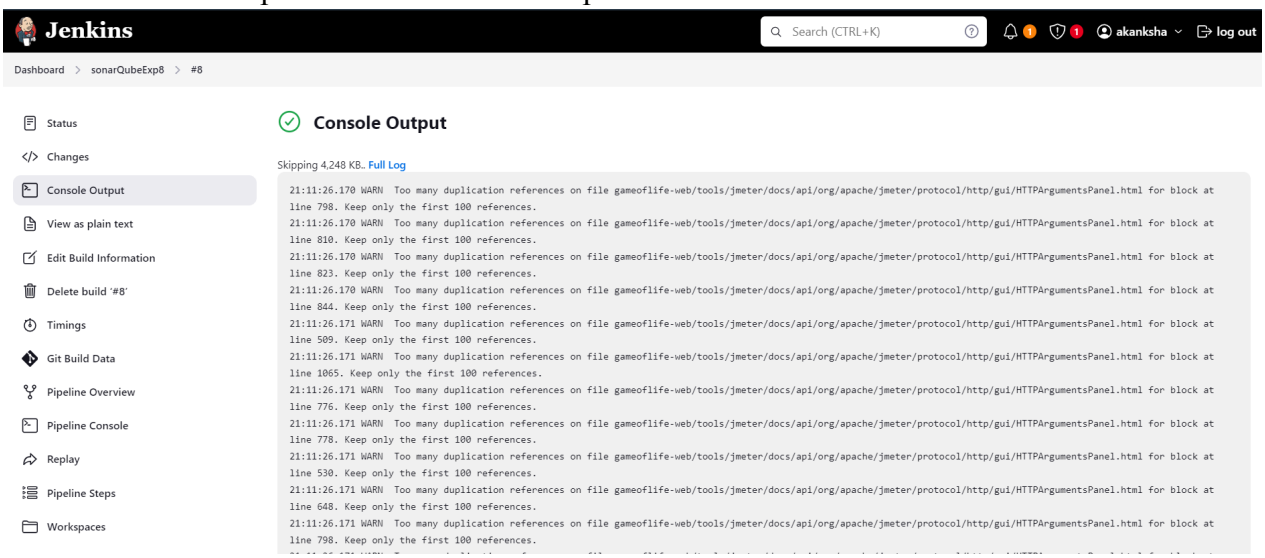
It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

8. Run The Build.

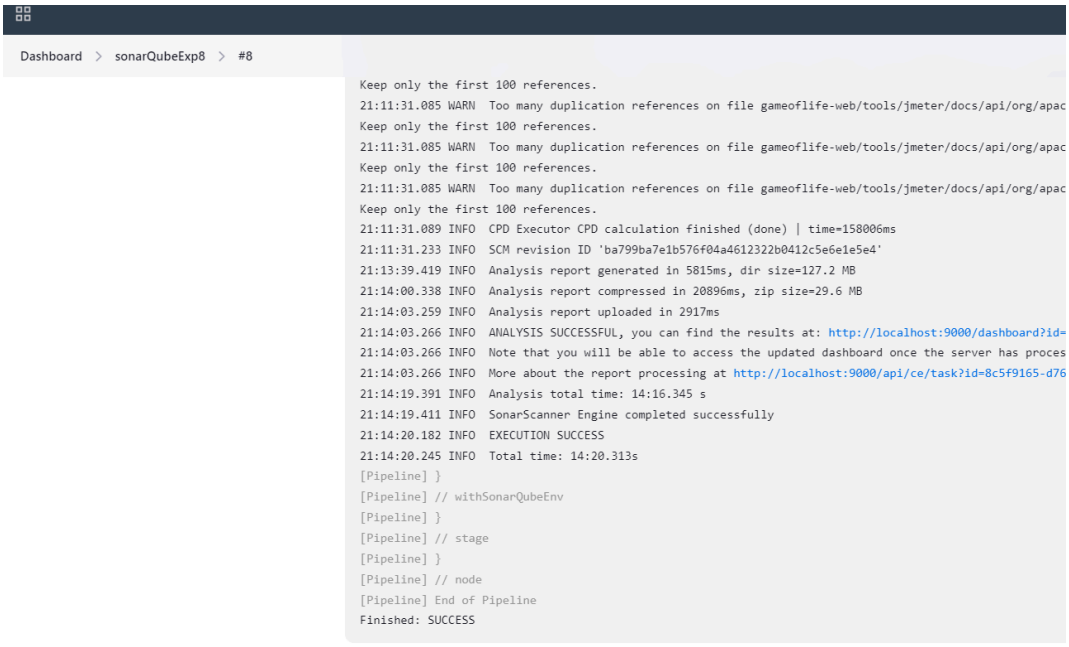
Name : Akanksha Shinde Class: D15C Roll No: 53



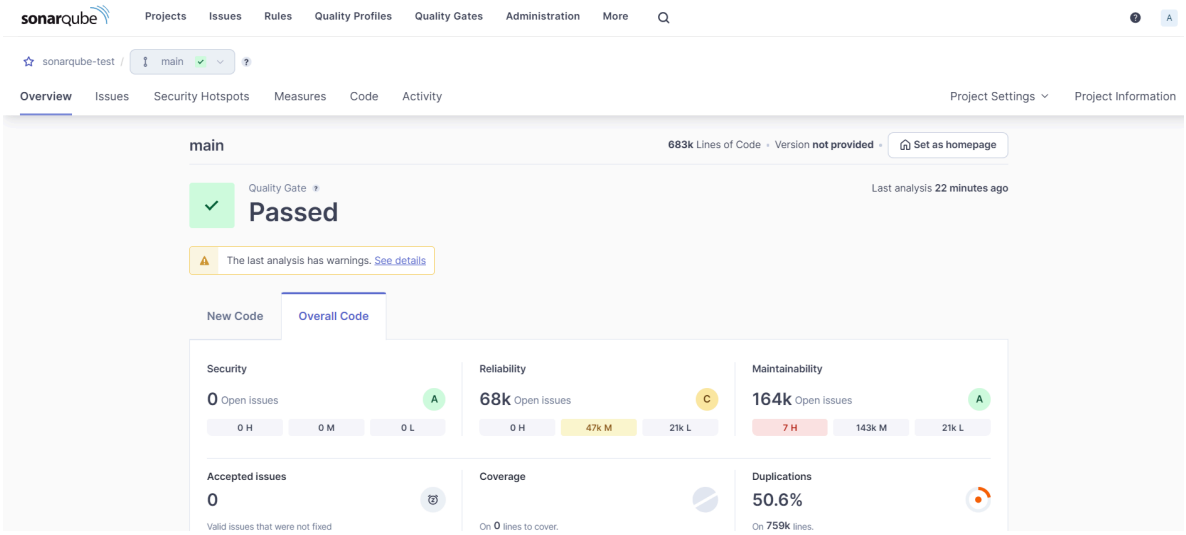
9. Check the console output once the build is complete.



Name : Akanksha Shinde Class: D15C Roll No: 53



10. After that, check the project in SonarQube.

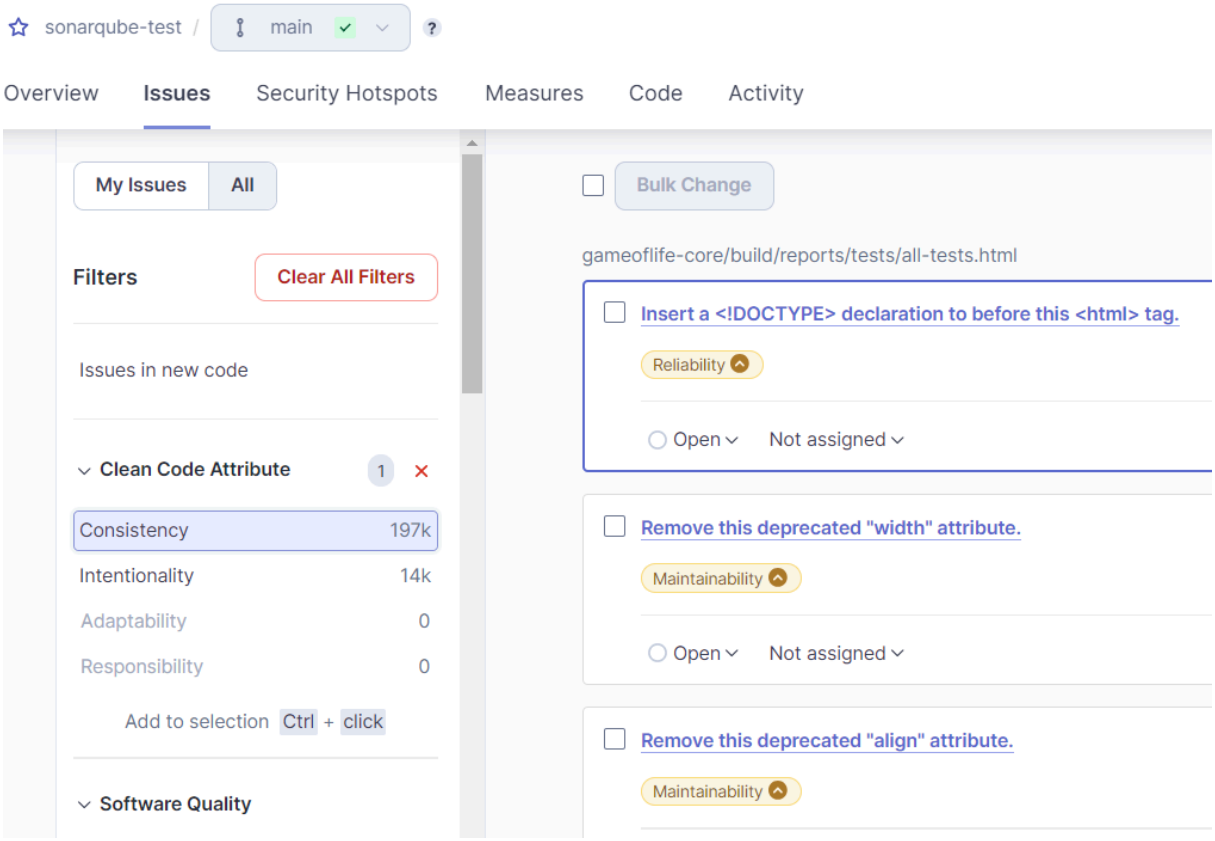


Under different tabs, check all different issues with the code.

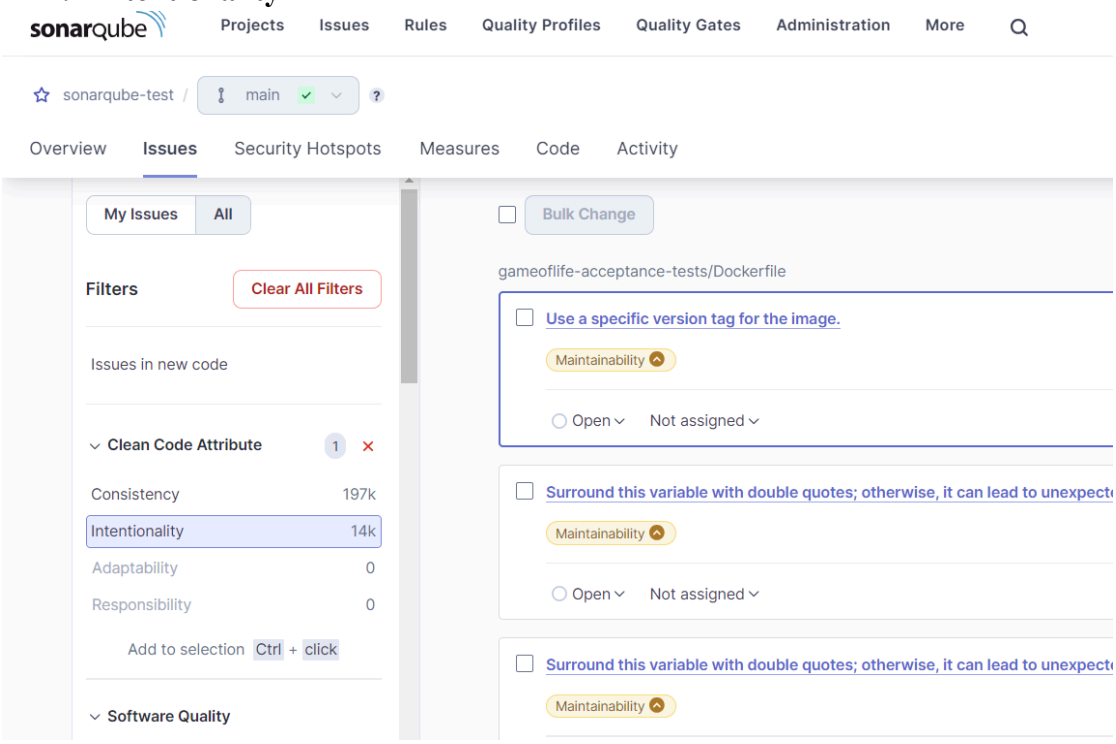
11. To check the **Code Problems** click on **issues-**

1. **Consistency**

Name : Akanksha Shinde Class: D15C Roll No: 53

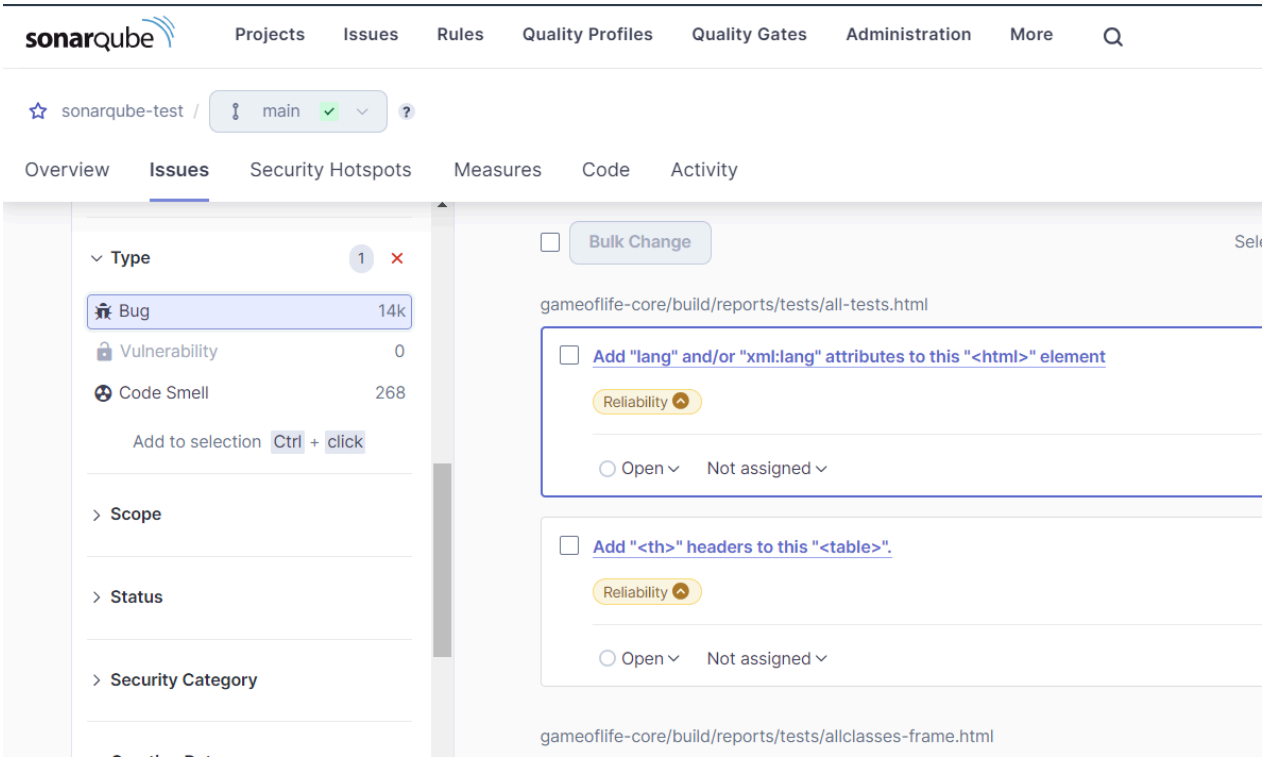


2. Intentionality

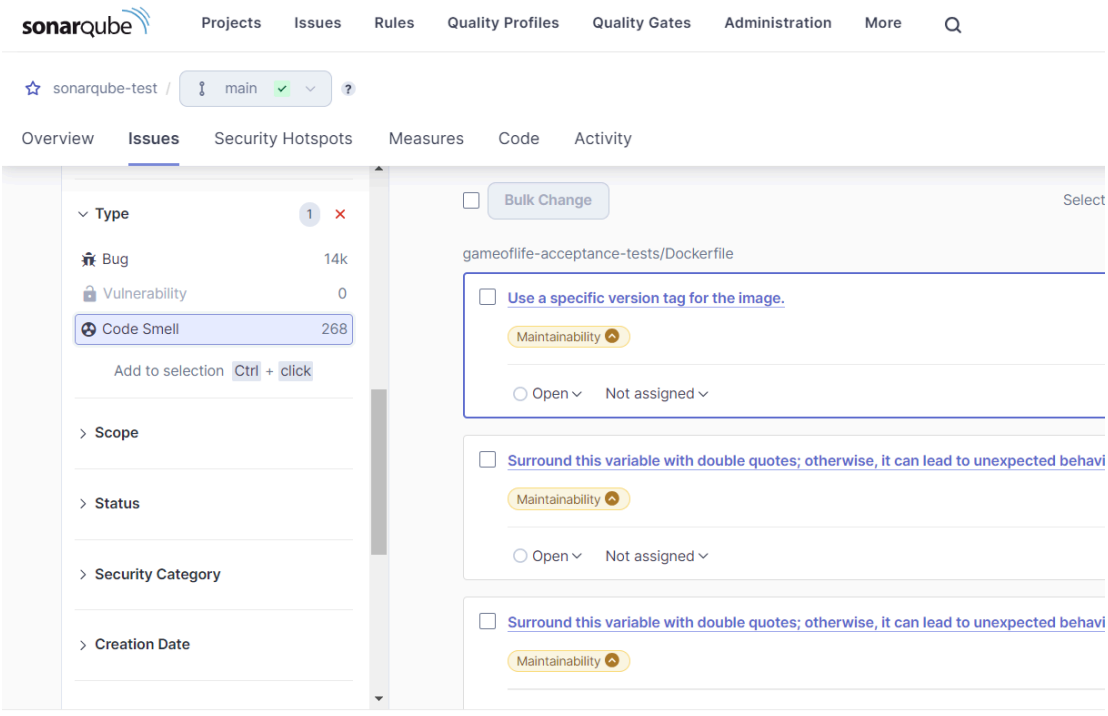


Name : Akanksha Shinde Class: D15C Roll No: 53

Bugs

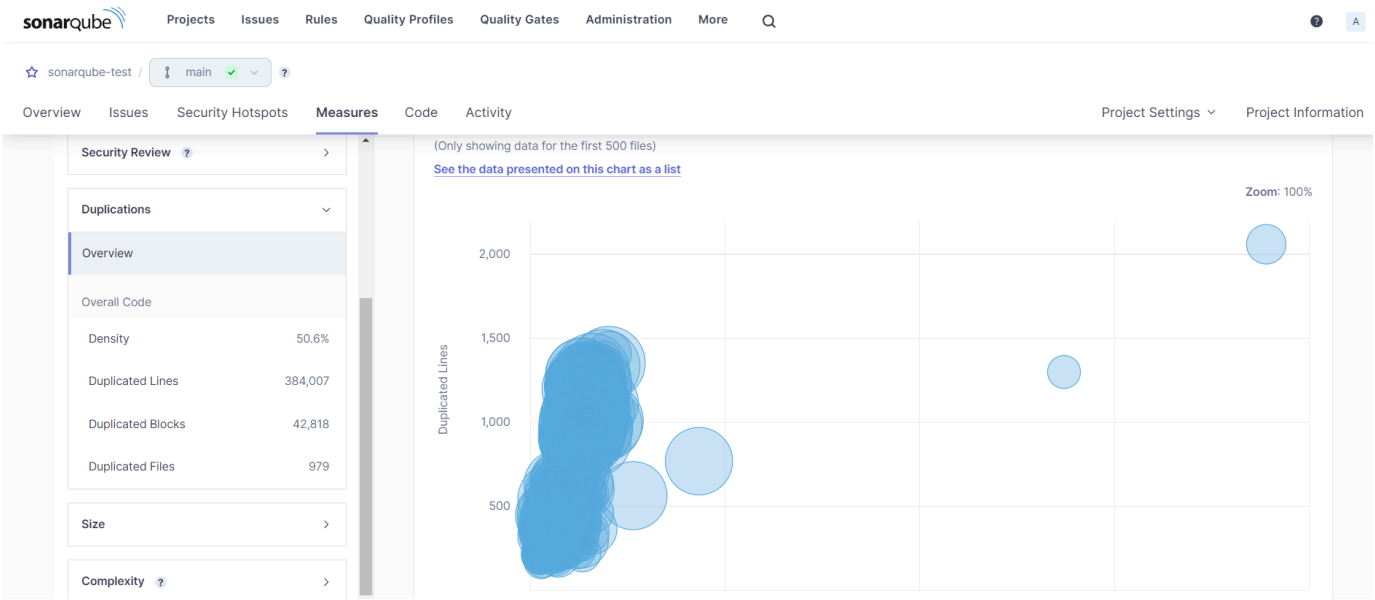


Code Smells

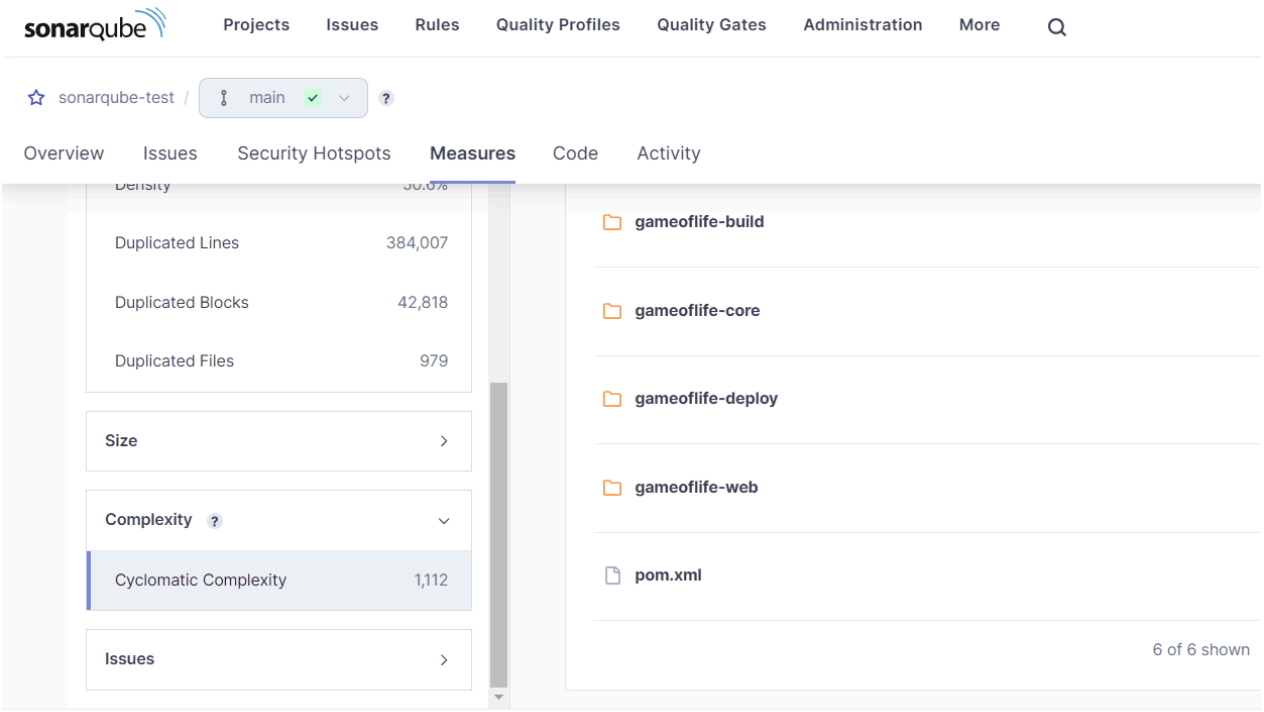


Name : Akanksha Shinde Class: D15C Roll No: 53

Duplicates



Cyclomatic Complexities



In this way, we have created a CI/CD Pipeline with Jenkins and integrated it with SonarQube to find issues in the code like bugs, code smells, duplicates, cyclomatic complexities, etc.

Conclusion:

We set up Jenkins and SonarQube to help check our code for problems automatically. First, we used Docker to get SonarQube running. Then, we made a project for it and connected Jenkins using a special plugin. We also added the details for the SonarQube server.

Next, we created a Jenkins pipeline that can pull our code from GitHub and check it for mistakes. Here I had faced errors while mentioning the path of the sonar-scanner.bat file and also my login, password credentials were not working so I had to create a token for that.Thus, ultimately the experiment was performed successfully. And every time when we work on our code it can find bugs, bad code, and security issues. This makes our code better and helps work faster.

Name: Akanksha Shinde Class: D15C Roll No: 53

Experiment No 9

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

Theory:

What is Nagios?

Nagios is an open-source software for continuous monitoring of systems, networks, and infrastructures. It runs plugins stored on a server that is connected with a host or another server on your network or the Internet. In case of any failure, Nagios alerts about the issues so that the technical team can perform the recovery process immediately.

Nagios is used for continuous monitoring of systems, applications, service and business processes in a DevOps culture.

Why We Need Nagios tool?
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 monitor 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

Features of Nagios

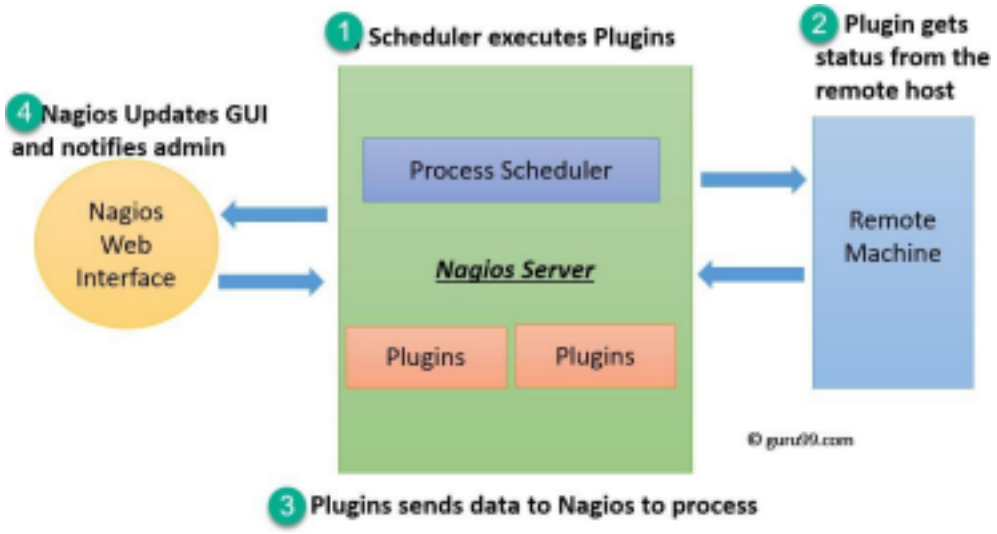
Following are the important features of Nagios monitoring tool:

- Relatively scalable, Manageable, and Secure
- Good log and database system
- Informative and attractive web interfaces
- Automatically send alerts if condition changes
- If the services are running fine, then there is no need to do check that host is an alive
- Helps you to detect network errors or server crashes
- You can troubleshoot the performance issues of the server.
- The issues, if any, can be fixed automatically as they are identified during the monitoring process
- You can monitor the entire business process and IT infrastructure with a single pass ● The product's architecture is easy to write new plugins in the language of your choice ● Nagios allows you to read its configuration from an entire directory which helps you to decide how to define individual files
- Utilizes topology to determine dependencies

- Monitor network services like HTTP, SMTP, HTTP, SNMP, FTP, SSH, POP, etc.
- Helps you to define network host hierarchy using parent hosts
- Ability to define event handlers that runs during service or host events for proactive problem resolution
- Support for implementing redundant monitoring hosts

Nagios Architecture

Nagios is a client-server architecture. Usually, on a network, a Nagios server is running on a host, and plugins are running on all the remote hosts which should be monitored.



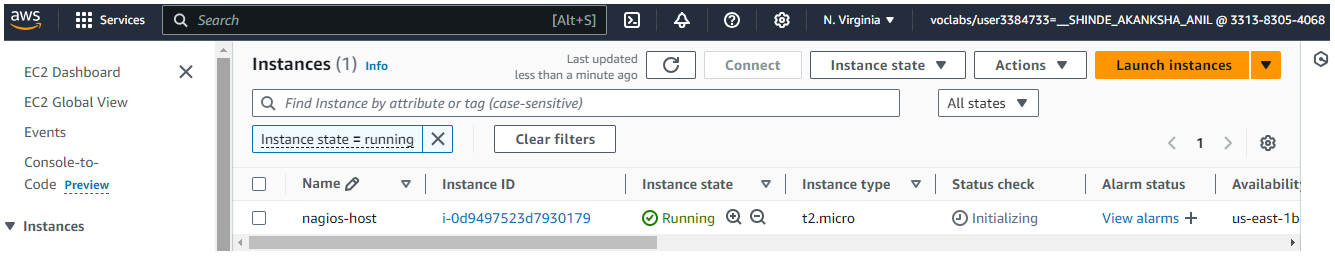
1. The scheduler is a component of the server part of Nagios. It sends a signal to execute the plugins at the remote host.
2. The plugin gets the status from the remote host
3. The plugin sends the data to the process scheduler
4. The process scheduler updates the GUI and notifications are sent to admins.

Installation of Nagios

Prerequisites: AWS Free Tier

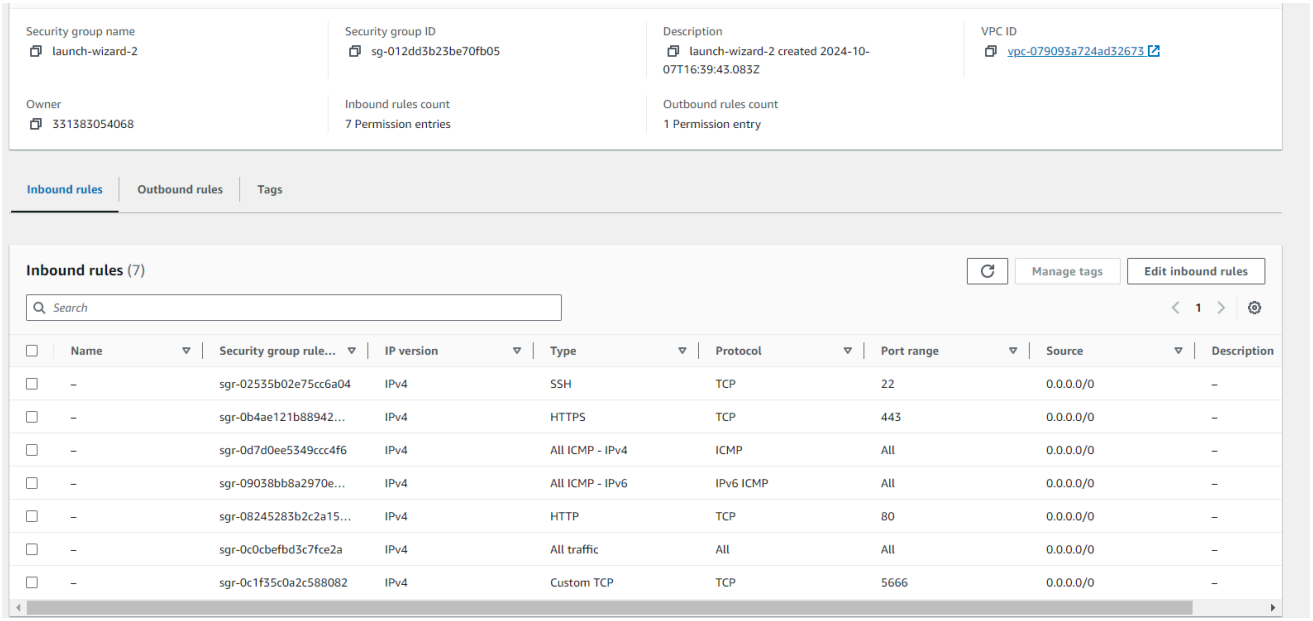
Steps:

1. Create an Amazon Linux EC2 Instance in AWS and name it - nagios-host



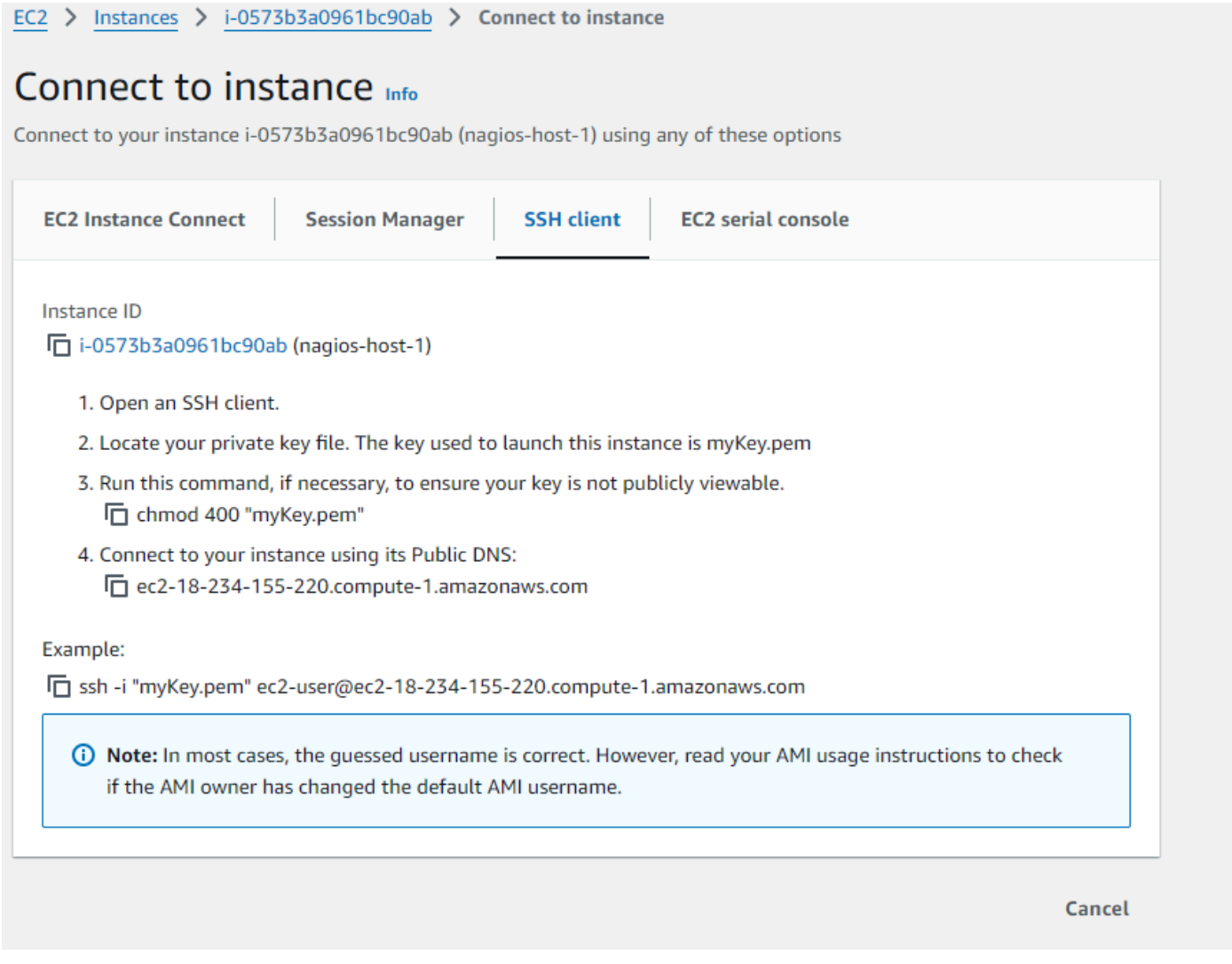
2. Under Security Group, make sure HTTP, HTTPS, SSH, ICMP are open from everywhere.

Name: Akanksha Shinde Class: D15C Roll No: 53



You have to edit the inbound rules of the specified Security Group for this like above.

Go in ssh client copy the command



Name: Akanksha Shinde Class: D15C Roll No: 53

3. SSH into Your EC2 instance or simply use EC2 Instance Connect from the browser.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\akank> ssh -i "C:\Users\akank\Downloads\nagiosExp.pem" ec2-user@ec2-54-91-248-79.compute-1.amazonaws.com
The authenticity of host 'ec2-54-91-248-79.compute-1.amazonaws.com (54.91.248.79)' can't be established.
ED25519 key fingerprint is SHA256:4dvXOJ92bJuuoDfhNtTc2oFlpN/S/Cu9H9YbZjfYsSg.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-91-248-79.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Mon Oct  7 17:04:25 2024 from 18.206.107.28
[ec2-user@ip-172-31-36-137 ~]$
```

4. Update the package indices and install the following packages using yum

sudo yum update
sudo yum install httpd php
sudo yum install gcc glibc glibc-common
sudo yum install gd gd-devel

```
Installed:
  apr-1.7.2-2.amzn2023.0.2.x86_64                    apr-util-1.6.3-1.amzn2023.0.1.x86_64          apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_
  generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch  httpd-2.4.62-1.amzn2023.x86_64                httpd-core-2.4.62-1.amzn2023.x86_64
  httpd-filesystem-2.4.62-1.amzn2023.noarch          httpd-tools-2.4.62-1.amzn2023.x86_64          libbrotli-1.0.9-4.amzn2023.0.2.x86_64
  libsodium-1.0.19-4.amzn2023.x86_64                 libxslt-1.1.34-5.amzn2023.0.2.x86_64          mailcap-2.1.49-3.amzn2023.0.3.noarch
  mod_http2-2.0.27-1.amzn2023.0.3.x86_64             mod_lua-2.4.62-1.amzn2023.x86_64              nginx-filesystem-1:1.24.0-1.amzn2023.0.4.n
  php8.3-8.3.10-1.amzn2023.0.1.x86_64                php8.3-cli-8.3.10-1.amzn2023.0.1.x86_64       php8.3-common-8.3.10-1.amzn2023.0.1.x86_64
  php8.3-fpm-8.3.10-1.amzn2023.0.1.x86_64            php8.3-mbstring-8.3.10-1.amzn2023.0.1.x86_64  php8.3-opcache-8.3.10-1.amzn2023.0.1.x86_6
  php8.3-pdo-8.3.10-1.amzn2023.0.1.x86_64            php8.3-process-8.3.10-1.amzn2023.0.1.x86_64   php8.3-sodium-8.3.10-1.amzn2023.0.1.x86_64
  php8.3-xml-8.3.10-1.amzn2023.0.1.x86_64

Complete!
[ec2-user@ip-172-31-46-30 ~]$
```

5. Create a new Nagios User with its password. You'll have to enter the password twice for confirmation.

sudo adduser -m nagios
sudo passwd nagios

```
[ec2-user@ip-172-31-46-30 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-46-30 ~]$
```

6. Create a new user group

sudo groupadd nagcmd

7. Use these commands so that you don't have to use sudo for Apache and Nagios

sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache

```
[ec2-user@ip-172-31-46-30 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-46-30 ~]$ sudo usermod -a -G nagcmd nagios
[ec2-user@ip-172-31-46-30 ~]$ sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-46-30 ~]$
```

8. Create a new directory for Nagios downloads

mkdir ~/downloads
cd ~/downloads

```
[ec2-user@ip-172-31-46-30 ~]$ mkdir ~/downloads
[ec2-user@ip-172-31-46-30 ~]$ cd ~/downloads
[ec2-user@ip-172-31-46-30 downloads]$
```

9. Use wget to download the source zip files.

wget
http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-4.0.8.tar. gz

wget http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz

```
nagios-4.0.8.tar.gz                 100%[=================================================================>]   1.72M  --.-KB/s    in 0

2024-10-04 03:59:15 (24.4 MB/s) - 'nagios-4.0.8.tar.gz' saved [1805059/1805059]
```

```
[ec2-user@ip-172-31-46-30 downloads]$ wget http://nagios-plugins.org/download/nagios-plugins-2.0.3
--2024-10-04 04:00:01--  http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:80... connected.
HTTP request sent, awaiting response... 200 OK
```

10. Use tar to unzip and change to that directory.
tar zxvf 6kqcx

Go to update nagios 4.5.5 run following command.

```
[ec2-user@ip-172-31-46-30 downloads]$ cd nagios-4.5.5
[ec2-user@ip-172-31-46-30 nagios-4.5.5]$
```

11. configure file by this command

./configure --with-command-group=nagcmd

```
checking for snprintf... yes
checking for asprintf... yes
checking for vasprintf... yes
checking for sigaction... yes
checking for C99 vsnprintf... yes
checking for library containing getservbyname... none required
checking for library containing connect... none required
checking for initgroups... yes
checking for setenv... yes
checking for strdup... yes
checking for strstr... yes
checking for strtoul... yes
```

Will get an error saying **ssl not found** so we need to install it by running the following command sudo yum install openssl-devel

```
Installed:
  openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64

Complete!
```

Now again run the command ./configure --with-command-group=nagcmd

```
[ec2-user@ip-172-31-46-30 nagios-4.5.5]$  ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
```

12. Compile the source code.

make all

At the end you see this message

```
For more information on obtaining support for Nagios, visit:

       https://support.nagios.com

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

Enjoy.
```

13. Install binaries, init script and sample config files. Lastly, set permissions on the external command directory.

sudo make install

```
  make install-commandmode
     - This installs and configures permissions on the
       directory for holding the external command file

  make install-config
     - This installs sample config files in /usr/local/nagios/etc

make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5'
[ec2-user@ip-172-31-46-30 nagios-4.5.5]$
```

sudo make install-init

```
[ec2-user@ip-172-31-46-30 nagios-4.5.5]$ sudo make install-init
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.service
[ec2-user@ip-172-31-46-30 nagios-4.5.5]$
```

sudo make install-config

```
*** 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.

[ec2-user@ip-172-31-46-30 nagios-4.5.5]$
```

sudo make install-commandmode

```
[ec2-user@ip-172-31-46-30 nagios-4.5.5]$ sudo make install-commandmode
/usr/bin/install -c -m 775 -o nagios -g nagcmd -d /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***
```

Run the command sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin for setting the password.

```
[ec2-user@ip-172-31-46-30 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
```

Restart the services with sudo service httpd restart

```
[ec2-user@ip-172-31-46-30 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-46-30 nagios-4.5.5]$
```

14: Extract the files from the downloaded Nagios plugin 2.4.11 run the following command first change the directory.
cd ~/downloads
tar zxvf nagios-plugins-2.4.11.tar.gz

Name: Akanksha Shinde Class: D15C Roll No: 53

15: Change the directory to nagios-plugins-2.4.11 and run the config command to configure. cd nagios-plugins-2.4.11

```
[ec2-user@ip-172-31-46-30 downloads]$ cd nagios-plugins-2.4.11
[ec2-user@ip-172-31-46-30 nagios-plugins-2.4.11]$
```

./configure --with-nagios-user=nagios --with-nagios-group=nagios

```
config.status: creating pkg/solaris/pkginfo
config.status: creating po/Makefile.in
config.status: creating config.h
config.status: config.h is unchanged
config.status: executing depfiles commands
config.status: executing libtool commands
config.status: executing po-directories commands
config.status: creating po/POTFILES
config.status: creating po/Makefile
[ec2-user@ip-172-31-46-30 nagios-plugins-2.4.11]$
```

```
       Checked 0 service escalations.
necking for circular paths...
       Checked 1 hosts
       Checked 0 service dependencies
       Checked 0 host dependencies
       Checked 5 timeperiods
necking global event handlers...
necking obsessive compulsive processor commands...
necking misc settings...

otal Warnings: 0
otal Errors:   0
```

14. Edit the config file and change the email address.

sudo nano /usr/local/nagios/etc/objects/contacts.cfg

```
define contact {

    contact_name            nagiosadmin             ; Short name of user
    use                     generic-contact         ; Inherit default values from generic-contact template (defined above)
    alias                   Nagios Admin            ; Full name of user
    email                   2022.akanksha.shinde@ves.ac.in ; <****** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}

|

###############################################################################
#
# CONTACT GROUPS
#
###############################################################################

# We only have one contact in this simple configuration file, so there is
                                [ Cancelled ]
^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo         M-A Set Mark     M-] To Bracket
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line    M-E Redo         M-6 Copy         ^Q Where Was
```

15. Configure the web interface.

sudo make install-webconf

```
[ec2-user@ip-172-31-87-75 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ 0 -eq 1 ]; then \
        ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf;
fi
```

16. Create a nagiosadmin account for nagios login along with password. You'll have to specify the password twice.

sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

```
[ec2-user@ip-172-31-87-75 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
```

17. Restart Apache

sudo service httpd restart

```
[ec2-user@ip-172-31-87-75 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-87-75 nagios-4.5.5]$
```

18. Go back to the downloads folder and unzip the plugins zip file.

cd ~/downloads

tar zxvf nagios-plugins-2.0.3.tar.gz

```
[ec2-user@ip-172-31-87-75 nagios-4.5.5]$ cd ~/downloads
tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
```

19. Compile and install plugins

cd nagios-plugins-2.0.3

./configure --with-nagios-user=nagios --with-nagios-group=nagios make

sudo make install

```
[ec2-user@ip-172-31-87-75 downloads]$ cd nagios-plugins-2.4.11
./configure --with-nagios-user=nagios --with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
```

20. Start Nagios

Add Nagios to the list of system services

sudo chkconfig --add nagios

sudo chkconfig nagios on

Verify the sample configuration files

sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg If there are no errors, you can go ahead and start Nagios.

sudo service nagios start

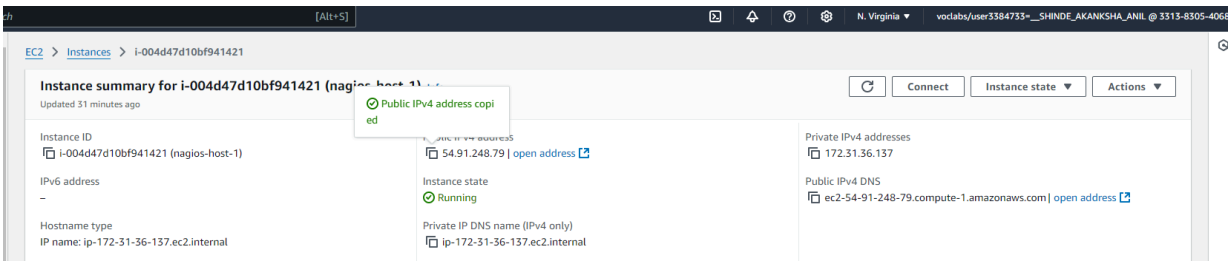
```
2-user@ip-172-31-46-218 ~]$ sudo service nagios start
rting nagios (via systemctl):                              [  OK  ]
2-user@ip-172-31-46-218 ~]$
```

21. Check the status of Nagios

sudo systemctl status nagios

22. Go back to EC2 Console and copy the Public IP address of this instance

```
ec2-user@ip-172-31-36-137:~,
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-36-137 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-36-137 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Mon 2024-10-07 17:42:18 UTC; 21s ago
       Docs: https://www.nagios.org/documentation
    Process: 65198 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 65204 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 65207 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.6M
        CPU: 70ms
     CGroup: /system.slice/nagios.service
             ├─65207 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─65208 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─65209 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─65210 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─65211 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─65254 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
```



23. Open up your browser and look for http://<your_public_ip_address>/nagios Enter username as nagiosadmin and password which you set in Step 16.

24. After entering the correct credentials, you will see this page.

This means that Nagios was correctly installed and configured with its plugins so far.

Conclusion:

While performing the experiment initially I faced error in the end that service is dead this was due to I had not properly given the access to necessary networks in the security groups so after giving all the access and additionally my password was not matching eventhough it was the same that I entered previously so we still can change the password not need to perform the whole experiment again just go back and copy the ssh link and go where you have downloaded the .pem file again and use the command "sudo cat/usr/local/nagios/etc/htpasswd.users" then "sudo htpasswd /usr/local/nagios/etc/htpasswd.users nagiosadmin" and restart the nagios service again by running "sudo systemctl restart nagios". Hence the experiment was performed successfully.

Experiment 10

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

Steps:

Prerequisites: AWS Free Tier, Nagios Server running on Amazon Linux Machine.

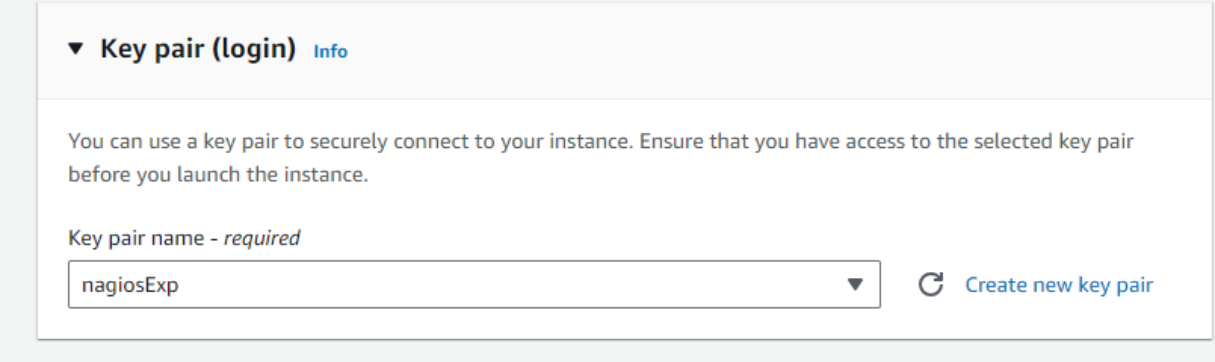
1. **To Confirm that Nagios is running on the server side, run this *sudo systemctl status nagios* on the "NAGIOS HOST" else if it is stopped run sudo service nagios start.**

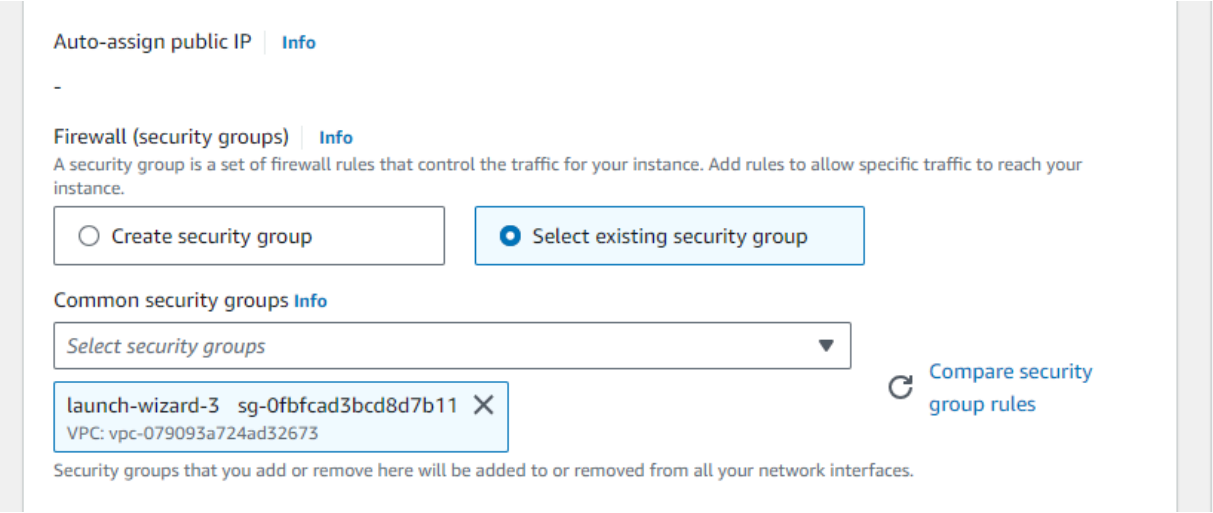
```
[ec2-user@ip-172-31-36-137 ~]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-36-137 ~]$ sudo service httpd status
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; prese>
    Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
     Active: active (running) since Wed 2024-10-09 05:45:21 UTC; 1min 12s a>
       Docs: man:httpd.service(8)
   Main PID: 3255 (httpd)
     Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; B>
      Tasks: 177 (limit: 1112)
     Memory: 16.8M
```

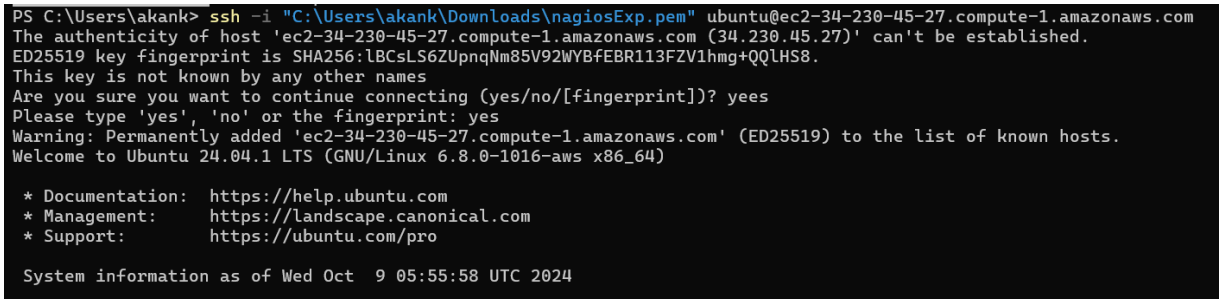
   you can proceed if you get this message.

2. **Before we begin, to monitor a Linux machine, create an Ubuntu 20.04 server EC2 Instance in AWS**.

   Provide it with the same security group as the Nagios Host and name it 'linux-client' alongside the host.







For now, leave this machine as is, and go back to your nagios HOST machine.

3. On the server, run this command

ps -ef | grep nagios

```
[ec2-user@ip-172-31-36-137 ~]$ ps -ef | grep nagios
nagios      2938       1  0 05:37 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios      2939    2938  0 05:37 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      2940    2938  0 05:37 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      2941    2938  0 05:37 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      2942    2938  0 05:37 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      2943    2938  0 05:37 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
ec2-user    4301    4212  0 05:59 pts/2    00:00:00 grep --color=auto nagio
[ec2-user@ip-172-31-36-137 ~]$
```

4. Become a root user and create 2 folders

sudo su

mkdir /usr/local/nagios/etc/objects/monitorhosts

Mkdir

```
[ec2-user@ip-172-31-36-137 ~]$ sudo su
[root@ip-172-31-36-137 ec2-user]# mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-36-137 ec2-user]#
```

run this command now

/usr/local/nagios/etc/objects/monitorhosts/linuxhosts

```
[root@ip-172-31-36-137 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
[root@ip-172-31-36-137 ec2-user]#
```

5. Copy the sample localhost.cfg file to linuxhost folder

cp /usr/local/nagios/etc/objects/localhost.cfg
/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

```
define host {

    use                     linux-server            ; Name of host template to use
                                                    ; This host definition will inherit all variables that are defined
                                                    ; in (or inherited by) the linux-server host template definition.
    host_name               localhost
    alias                   localhost
    address                 127.0.0.1
}



###############################################################################
#
# HOST GROUP DEFINITION
#
###############################################################################
                                                [ Read 159 lines ]
^G Help         ^O Write Out    ^W Where Is     ^K Cut          ^T Execute      ^C Location     M-U Undo        M-A Set Mark    M-] To Bracket
^X Exit         ^R Read File    ^\ Replace      ^U Paste        ^J Justify      ^/ Go To Line   M-E Redo        M-6 Copy        ^Q Where Was
```

6. Open linuxserver.cfg using nano and make the following changes

nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

Change the hostname to linuxserver (EVERYWHERE ON THE FILE) Change address to the public IP address of your **LINUX CLIENT**.

Change hostgroup_name under hostgroup to linux-servers1

```
define host {

    use                     linux-server1            ; Name of host template to use
                                                     ; This host definition will inherit all variables that are defined
                                                     ; in (or inherited by) the linux-server host template definition.
    host_name               linuxserver
    alias                   Linuxserver
    address                 34.230.45.27
}
```

```
# HOST GROUP DEFINITION
#
###############################################################################

# Define an optional hostgroup for Linux machines

define hostgroup {

    hostgroup_name          linux-servers1           ; The name of the hostgroup
    alias                   Linux Servers            ; Long name of the group
    members                 linuxserver          ; Comma separated list of hosts that belong to this group
}
```

Everywhere else on the file, change the hostname to linuxserver instead of localhost.

7. Open the Nagios Config file and add the following line

nano /usr/local/nagios/etc/nagios.cfg

```
# LOG FILE
# This is the main log file where service and host events are logged
# for historical purposes.  This should be the first option specified
# in the config file!!!

log_file=/usr/local/nagios/var/nagios.log
```

##Add this line

cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/

```
# You can also tell Nagios to process all config files (with a .cfg
# extension) in a particular directory by using the cfg_dir
# directive as shown below:

#cfg_dir=/usr/local/nagios/etc/servers
#cfg_dir=/usr/local/nagios/etc/printers
#cfg_dir=/usr/local/nagios/etc/switches
#cfg_dir=/usr/local/nagios/etc/routers
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
```

8. Verify the configuration files

```
[root@ip-172-31-36-137 ec2-user]# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
```

You are good to go if there are no errors.

9. Restart the nagios service

service nagios restart

```
nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Mon 2024-10-07 17:42:18 UTC; 21s ago
       Docs: https://www.nagios.org/documentation
    Process: 65198 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 65204 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 65207 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.6M
        CPU: 70ms
     CGroup: /system.slice/nagios.service
             ├─65207 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─65208 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─65209 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─65210 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
```

Now it is time to switch to the client machine.

10. SSH into the machine or simply use the EC2 InstanceConnectfeature.

```
ubuntu@ip-172-31-40-193:~$
```

11. Make a package index update and install gcc, nagios-nrpe-server and the plugins.

sudo apt update -y

```
ubuntu@ip-172-31-40-130:~$ sudo apt update -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [380 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
```

sudo apt install gcc -y

```
ubuntu@ip-172-31-40-130:~$ sudo apt install gcc -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu cpp cpp-13 cpp-13-x86-64-linux-gnu cpp-x86-64-linux-gnu fontconfig-config fonts-dejavu-core
  fonts-dejavu-mono gcc-13 gcc-13-base gcc-13-x86-64-linux-gnu gcc-x86-64-linux-gnu libaom3 libasan8 libatomic1 libbinutils libc-dev-bin libc-devtools
  libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libde265-0 libdeflate0 libfontconfig1 libgcc-13-dev libgd3 libgomp1 libgprofng0
  libheif-plugin-aomdec libheif-plugin-aomenc libheif-plugin-libde265 libheif1 libhwasan0 libisl23 libitm1 libjbig0 libjpeg-turbo8 libjpeg8 liblerc4
  liblsan0 libmpc3 libquadmath0 libsframe1 libsharpyuv0 libtiff6 libtsan2 libubsan1 libwebp7 libxpm4 linux-libc-dev manpages-dev rpcsvc-proto
Suggested packages:
```

sudo apt install -y nagios-nrpe-server nagios-plugins

```
ubuntu@ip-172-31-40-130:~$ sudo apt install -y nagios-nrpe-server nagios-plugins
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'monitoring-plugins' instead of 'nagios-plugins'
The following additional packages will be installed:
  libavahi-client3 libavahi-common-data libavahi-common3 libcups2t64 libdbi1t64 libldb2 libmysqlclient21 libnet-snmp-perl libpq5 libradcli4 libsmbclient0
  libsnmp-base libsnmp40t64 libtalloc2 libtdb1 libtevent0t64 liburiparser1 libwbclient0 monitoring-plugins-basic monitoring-plugins-common
  monitoring-plugins-standard mysql-common python3-gpg python3-ldb python3-markdown python3-samba python3-talloc python3-tdb rpcbind samba-common
  samba-common-bin samba-dsdb-modules samba-libs smbclient snmp
```

12. Open nrpe.cfg file to make changes.

sudo nano /etc/nagios/nrpe.cfg

Under allowed_hosts, add your nagios host IP address like so

```
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd

allowed_hosts=127.0.0.1,54.163.184.143


# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE daemon will allow clients
# to specify arguments to commands that are executed.  This option only works
# if the daemon was configured with the --enable-command-args configure script
```

13. Restart the NRPE server

sudo systemctl restart nagios-nrpe-server

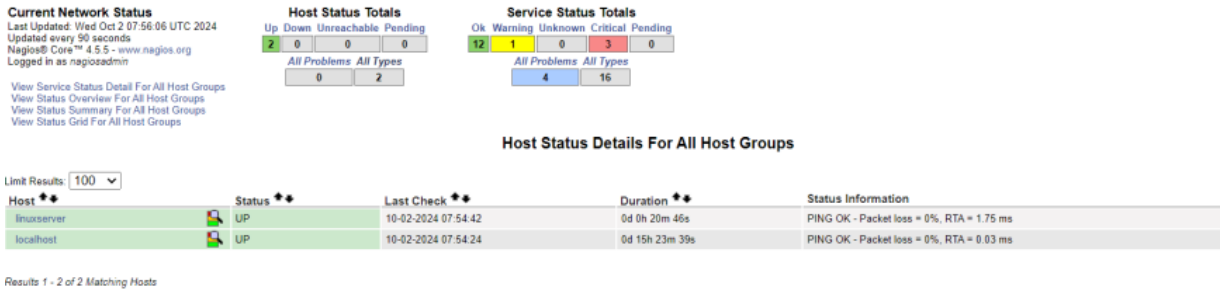
```
ubuntu@ip-172-31-40-130:~$ sudo systemctl restart nagios-nrpe-server
ubuntu@ip-172-31-40-130:~$
```

14. Now, check your nagios dashboard and you'll see a new host being added.

Go to your nagios page and click on host in left.

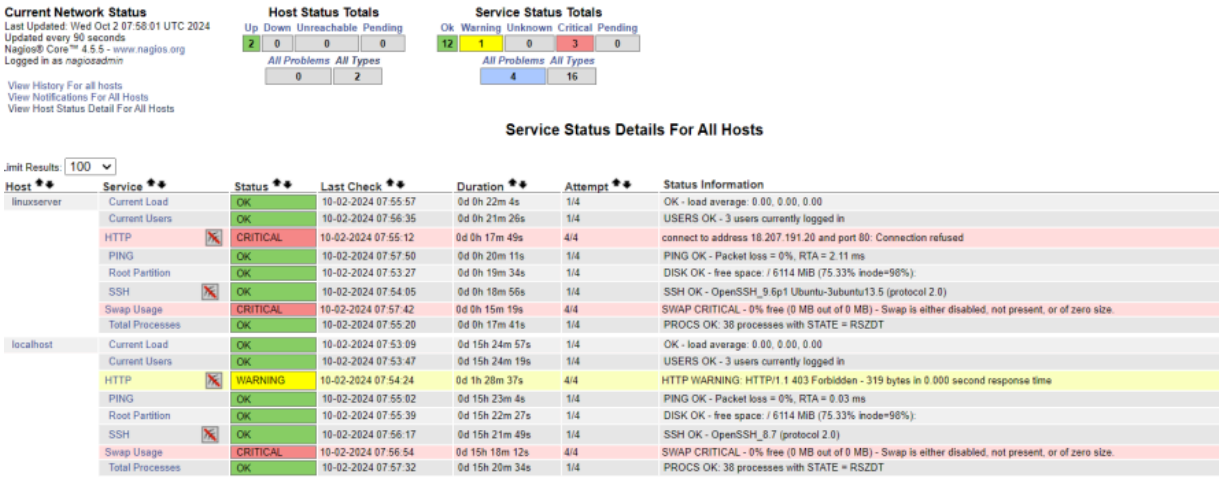
Name: AKANKSHA SHINDE CLASS: D15C ROLL NO: 53

Click on linuxserver to see the host details



You can click Services to see all services and ports being monitored.

Name: AKANKSHA SHINDE CLASS: D15C ROLL NO: 53



Current Network Status
Last Updated: Wed Oct 2 07:58:01 UTC 2024
Updated every 90 seconds
Nagios® Core™ 4.5.5 - www.nagios.org
Logged in as nagiosadmin

View History For all hosts
View Notifications For All Hosts
View Host Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
2	0	0	0

All Problems	All Types
0	2

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
12	1	0	3	0

All Problems	All Types
4	16

Service Status Details For All Hosts

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
linuxserver	Current Load	OK	10-02-2024 07:55:57	0d 0h 22m 4s	1/4	OK - load average: 0.00, 0.00, 0.00
	Current Users	OK	10-02-2024 07:56:35	0d 0h 21m 26s	1/4	USERS OK - 3 users currently logged in
	HTTP	CRITICAL	10-02-2024 07:55:12	0d 0h 17m 49s	4/4	connect to address 18.207.191.20 and port 80: Connection refused
	PING	OK	10-02-2024 07:57:50	0d 0h 20m 11s	1/4	PING OK - Packet loss = 0%, RTA = 2.11 ms
	Root Partition	OK	10-02-2024 07:53:27	0d 0h 19m 34s	1/4	DISK OK - free space: / 6114 MiB (75.33% inode=98%):
	SSH	OK	10-02-2024 07:54:05	0d 0h 18m 56s	1/4	SSH OK - OpenSSH_9.6p1 Ubuntu-3ubuntu13.5 (protocol 2.0)
	Swap Usage	CRITICAL	10-02-2024 07:57:42	0d 0h 15m 19s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.
	Total Processes	OK	10-02-2024 07:55:20	0d 0h 17m 41s	1/4	PROCS OK: 38 processes with STATE = RSZDT
localhost	Current Load	OK	10-02-2024 07:53:09	0d 15h 24m 57s	1/4	OK - load average: 0.00, 0.00, 0.00
	Current Users	OK	10-02-2024 07:53:47	0d 15h 24m 19s	1/4	USERS OK - 3 users currently logged in
	HTTP	WARNING	10-02-2024 07:54:24	0d 1h 28m 37s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.000 second response time
	PING	OK	10-02-2024 07:55:02	0d 15h 23m 4s	1/4	PING OK - Packet loss = 0%, RTA = 0.03 ms
	Root Partition	OK	10-02-2024 07:55:39	0d 15h 22m 27s	1/4	DISK OK - free space: / 6114 MiB (75.33% inode=98%):
	SSH	OK	10-02-2024 07:56:17	0d 15h 21m 49s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)
	Swap Usage	CRITICAL	10-02-2024 07:56:54	0d 15h 18m 12s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.
	Total Processes	OK	10-02-2024 07:57:32	0d 15h 20m 34s	1/4	PROCS OK: 38 processes with STATE = RSZDT

As you can see, we have our linuxserver up and running. It is showing critical status on HTTP due to permission errors and swap because there is no partition created.

Recommended

- Terminate both of your EC-2 instances to avoid charges.
- Delete the security group if you created a new one (it won't affect your bill, you may avoid it)

Conclusion:

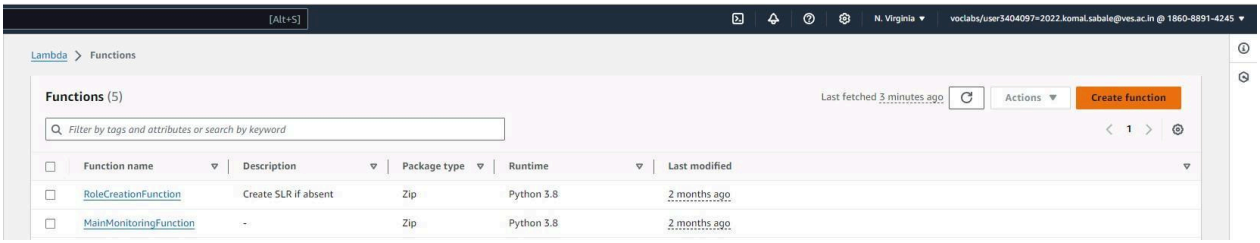
Make sure to start the nagios instance created in the previous experiment as without it there would be a failure. While performing the experiment I did not replaced the localhost name to linuxserver I did it only at one place so do make sure to replace the localhost name to linuxserver everywhere in the editor. After running all the necessary commands the experiment was performed successfully.

Name: Akanksha Shinde Class: D15C Roll No: 53

Experiment 11

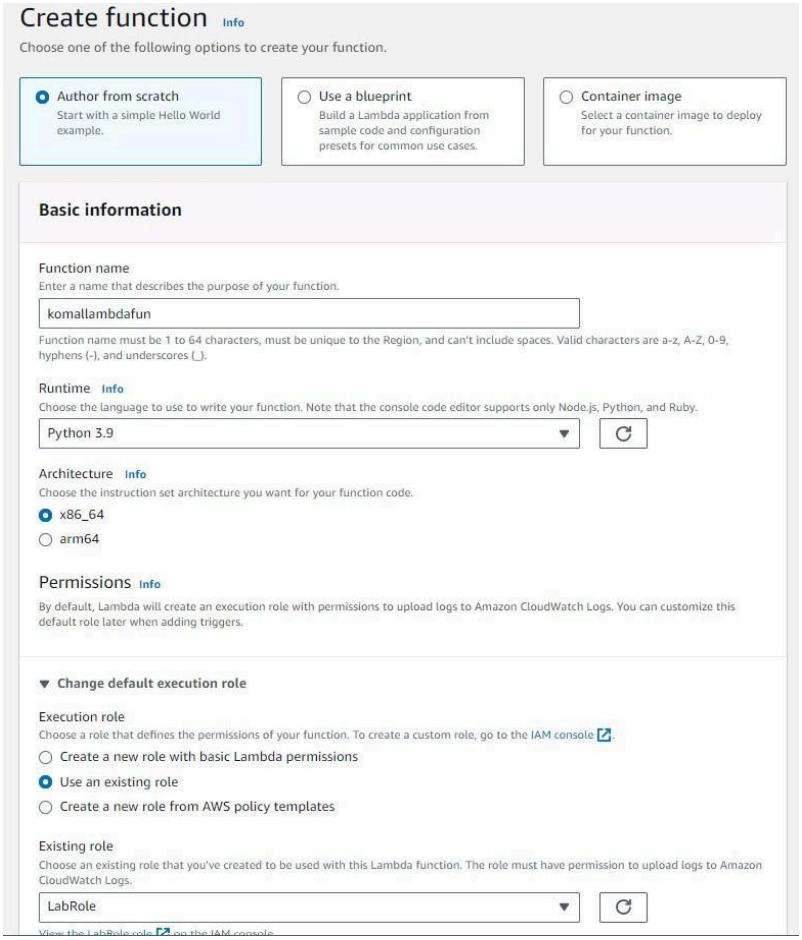
Steps to create an AWS Lambda function

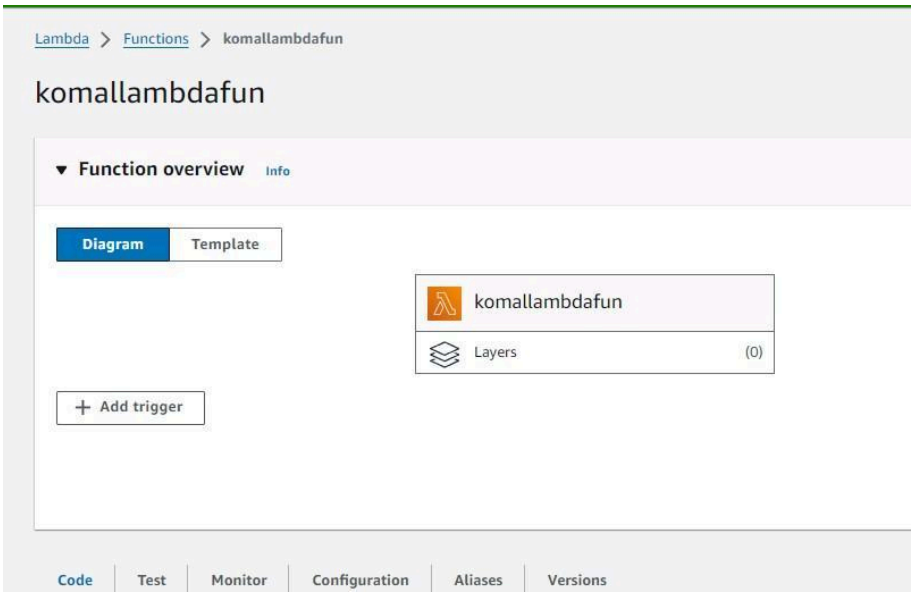
1. Open up the Lambda Console and click on the Create button. Be mindful of where you create your functions since Lambda is region-dependent.



2. Choose to create a function from scratch or use a blueprint, i.e templates defined by AWS for you with all configuration presets required for the most common use cases.

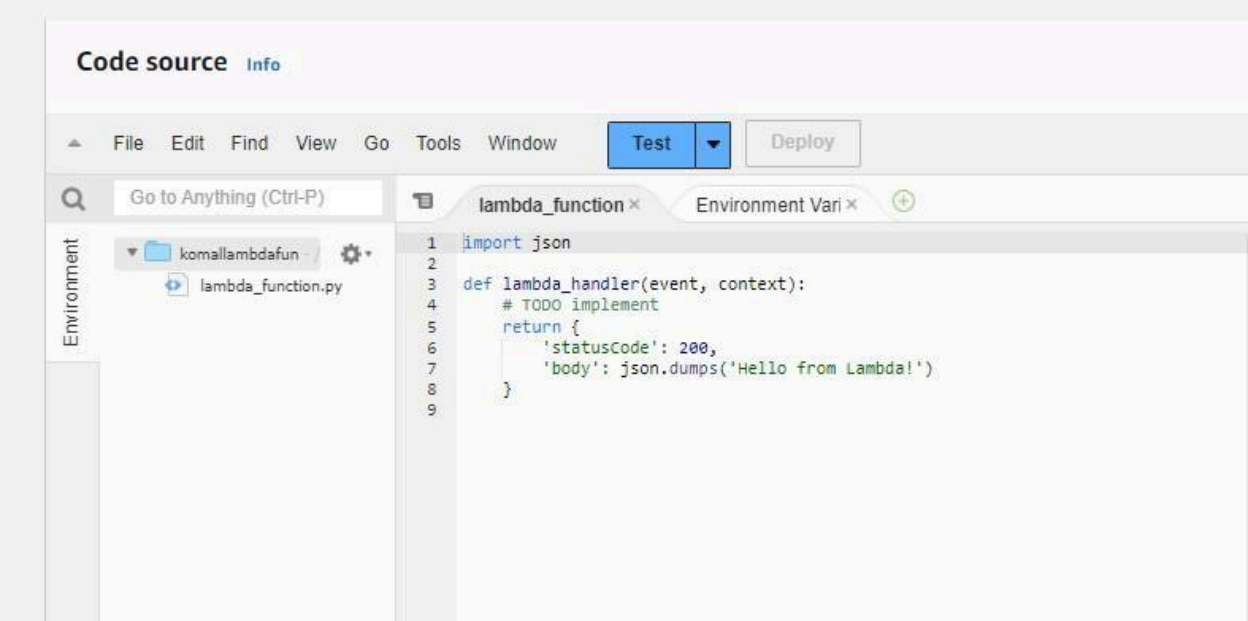
Name: Akanksha Shinde Class: D15C Roll No: 53



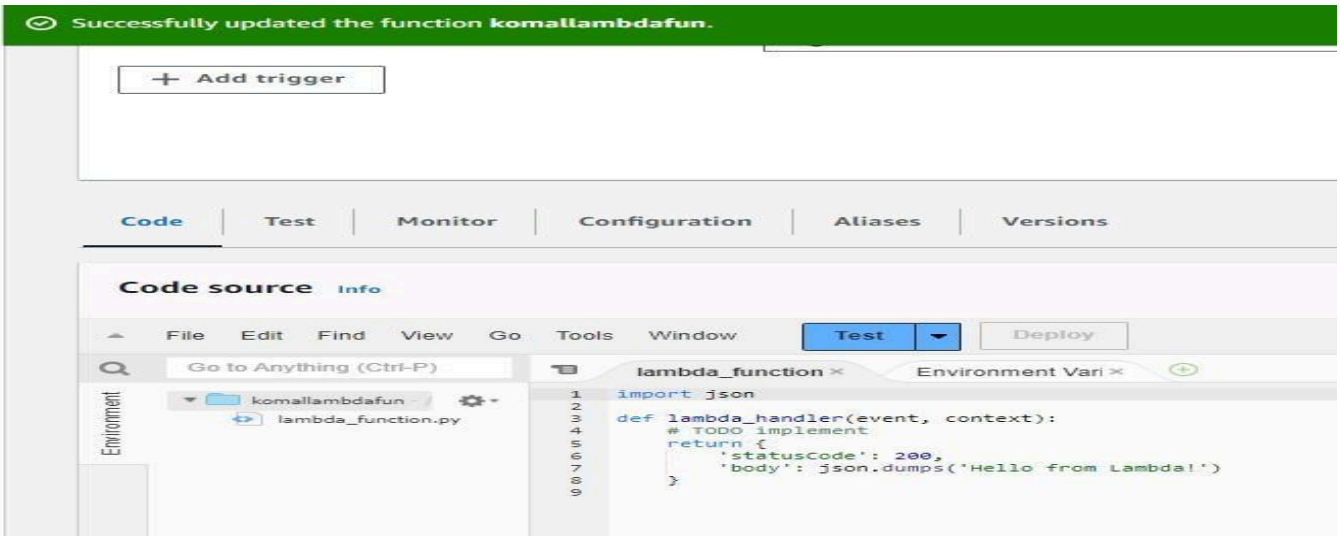


Name: Akanksha Shinde Class: D15C Roll No: 53

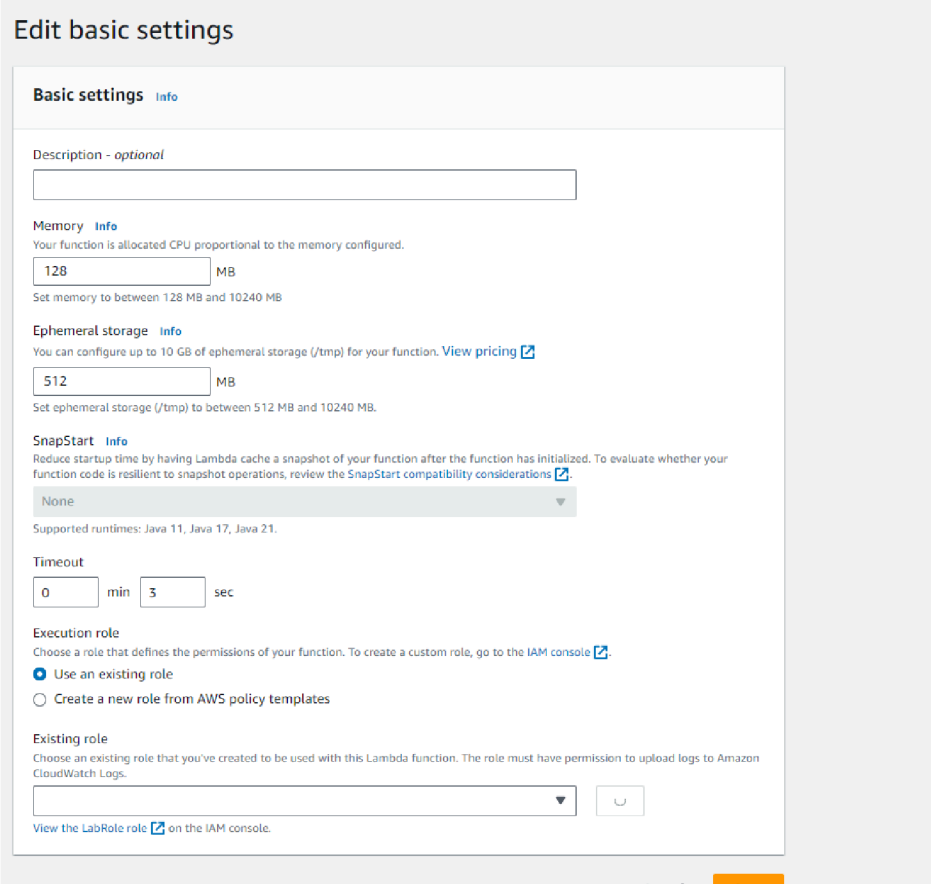
3. This process will take a while to finish and after that, you'll get a message that your function was successfully created.



4. To change the configuration, open up the Configuration tab and under General Configuration, choose Edit.

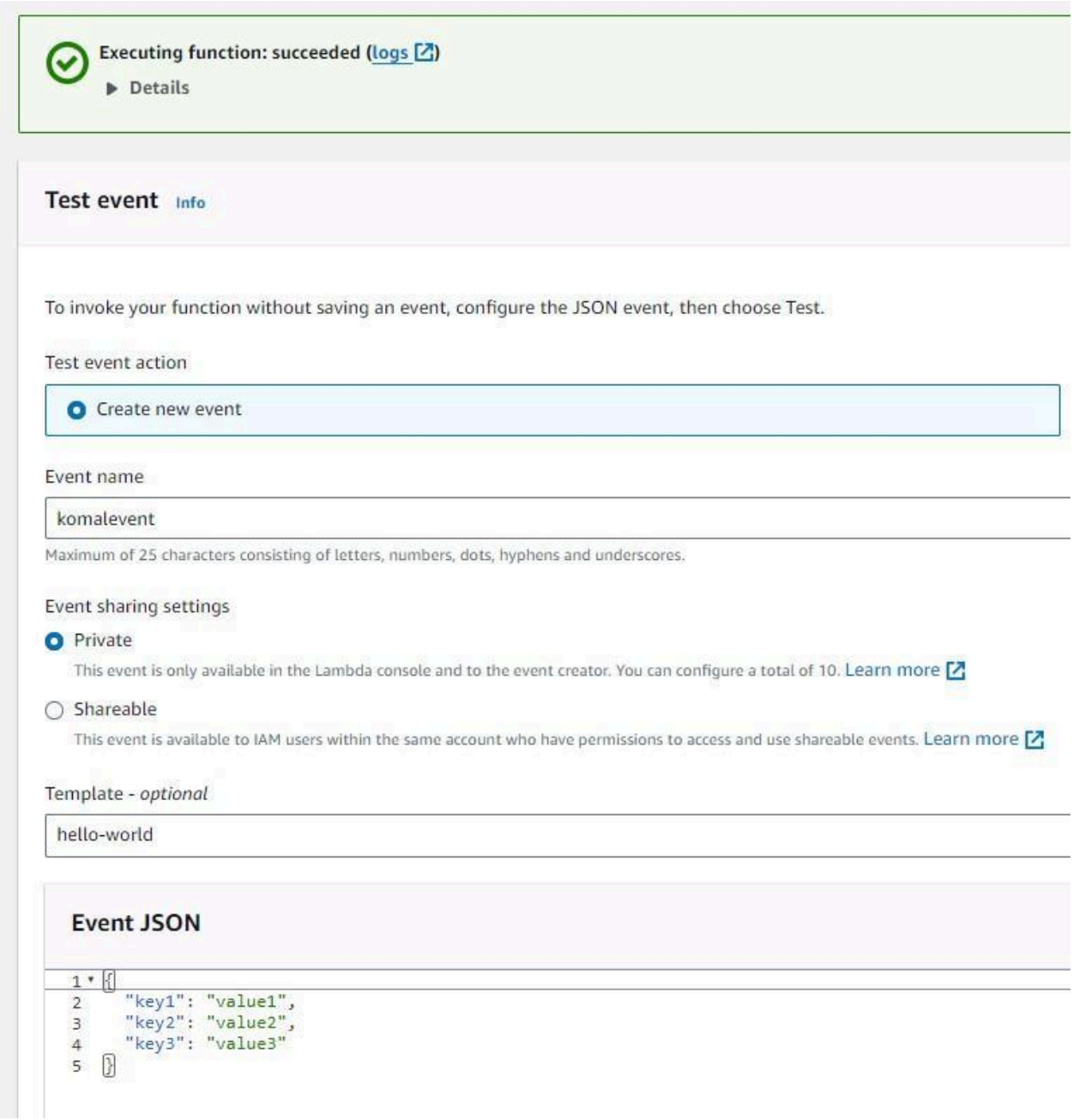


Name: Akanksha Shinde Class: D15C Roll No: 53



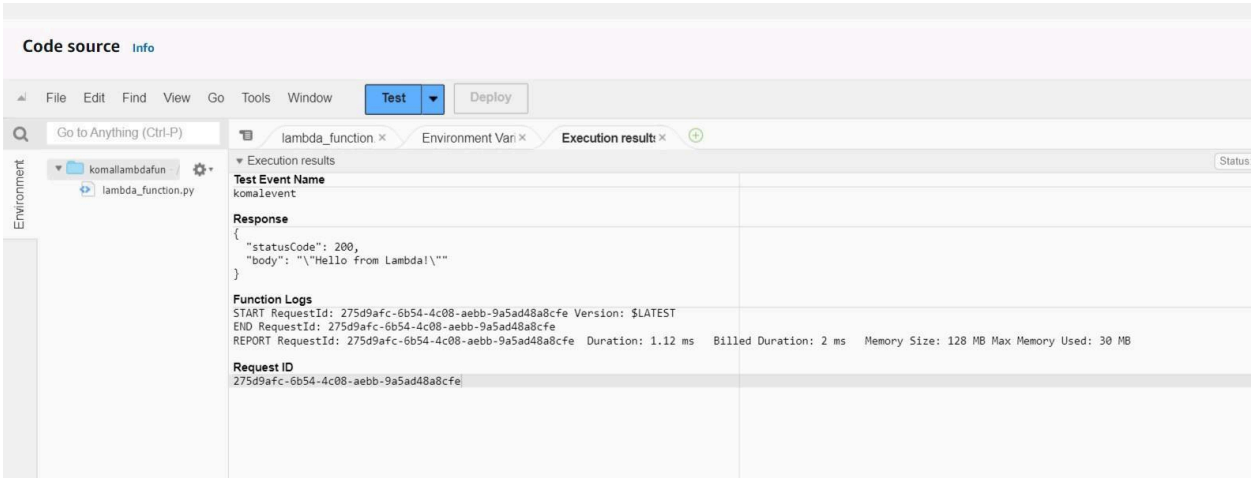
5. You can make changes to your function inside the code editor. You can also upload a zip file of your function or upload one from an S3 bucket if needed. Press Ctrl + S to save the file and click Deploy to deploy the changes
6. Click on Test and you can change the configuration, like so. If you do not have anything in the request body, it is important to specify two curly braces as valid JSON, so make sure they are there

Name: Akanksha Shinde Class: D15C Roll No: 53



7. Now click on Test and you should be able to see the results.

Name: Akanksha Shinde Class: D15C Roll No: 53



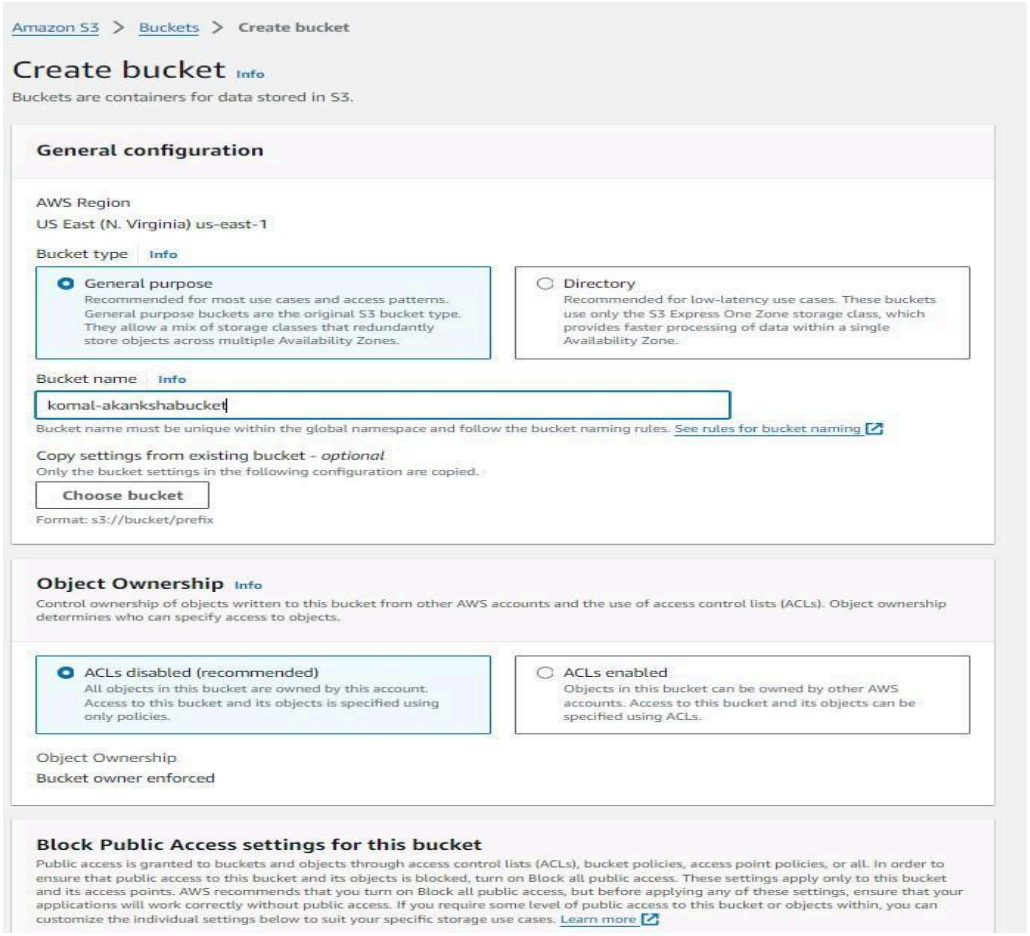
Conclusion:

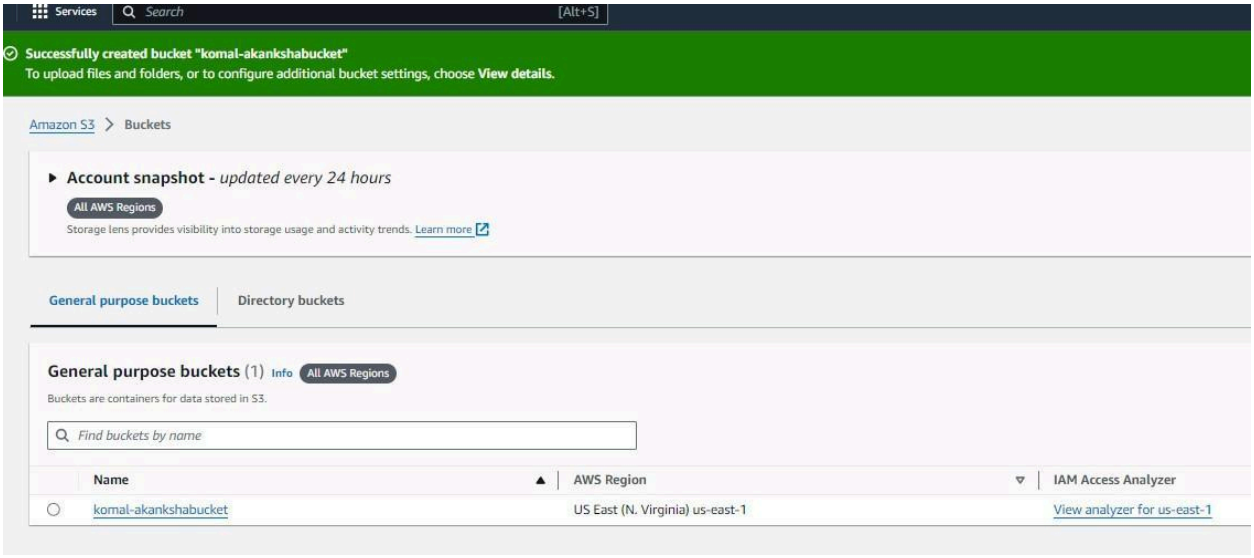
AWS Lambda automatically manages the compute resources, executes your code in response to specific events such as API calls, file uploads, or database updates, and scales based on the demand. The workflow of AWS Lambda involves defining a function with specific logic, configuring triggers that will invoke the function, and setting permissions to control access. Lambda supports multiple programming languages, including Python, Java, and Node.js, enabling developers to choose the best fit for their applications. Creating your first Lambda function is straightforward: you write the code, define triggers, and deploy, allowing you to quickly build and run applications without the overhead of managing infrastructure. This simplicity and flexibility make AWS Lambda an excellent tool for building modern, event-driven applications.

Name: Akanksha Shinde Class: D15C Roll No: 53

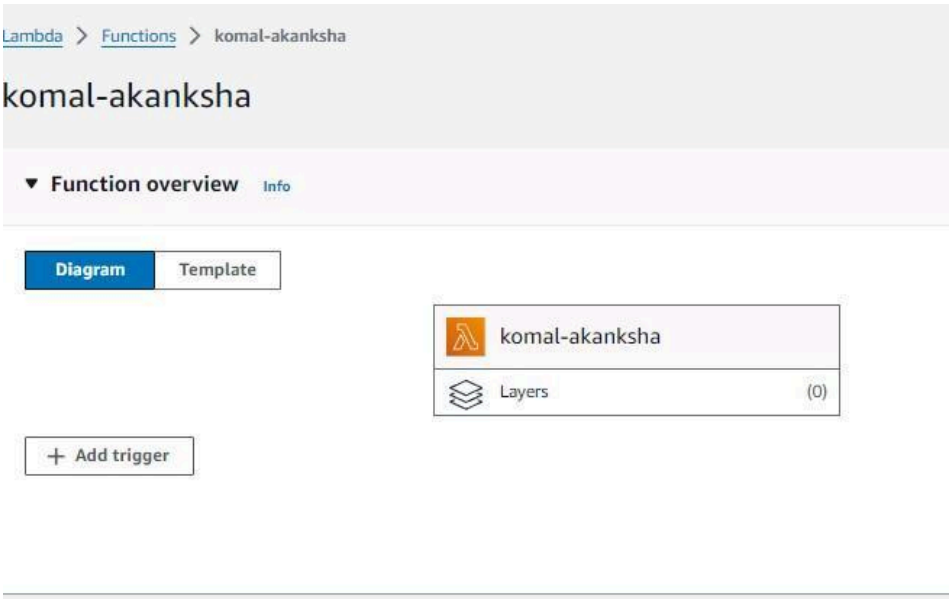
Experiment 12

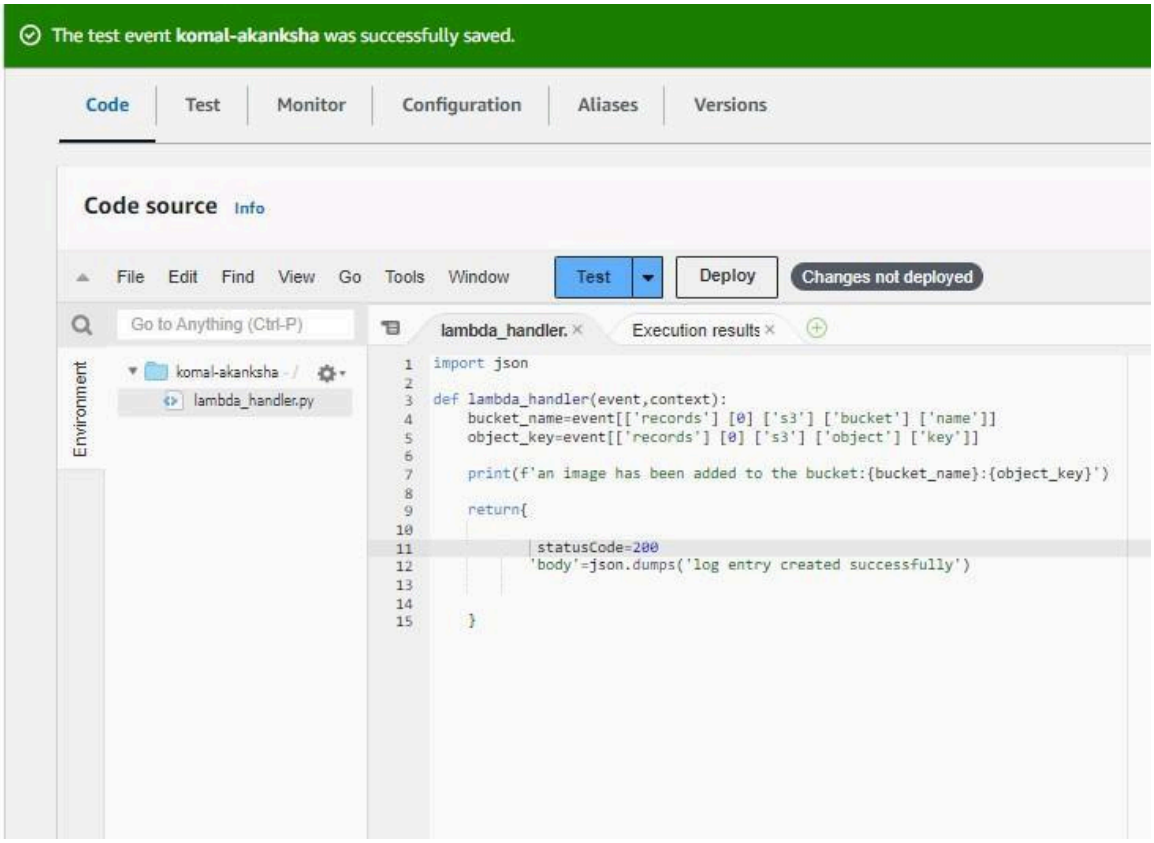
1)Create an S3 bucket .



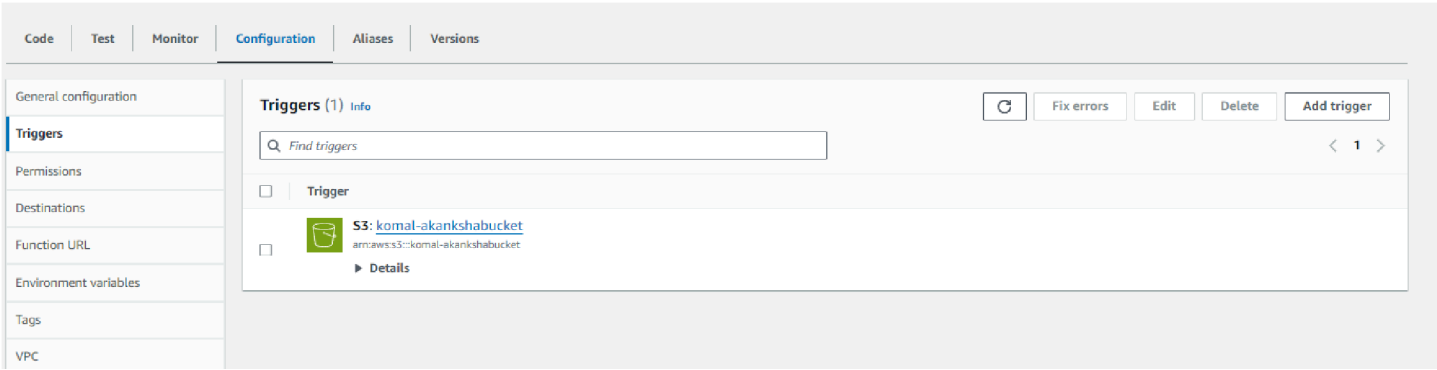


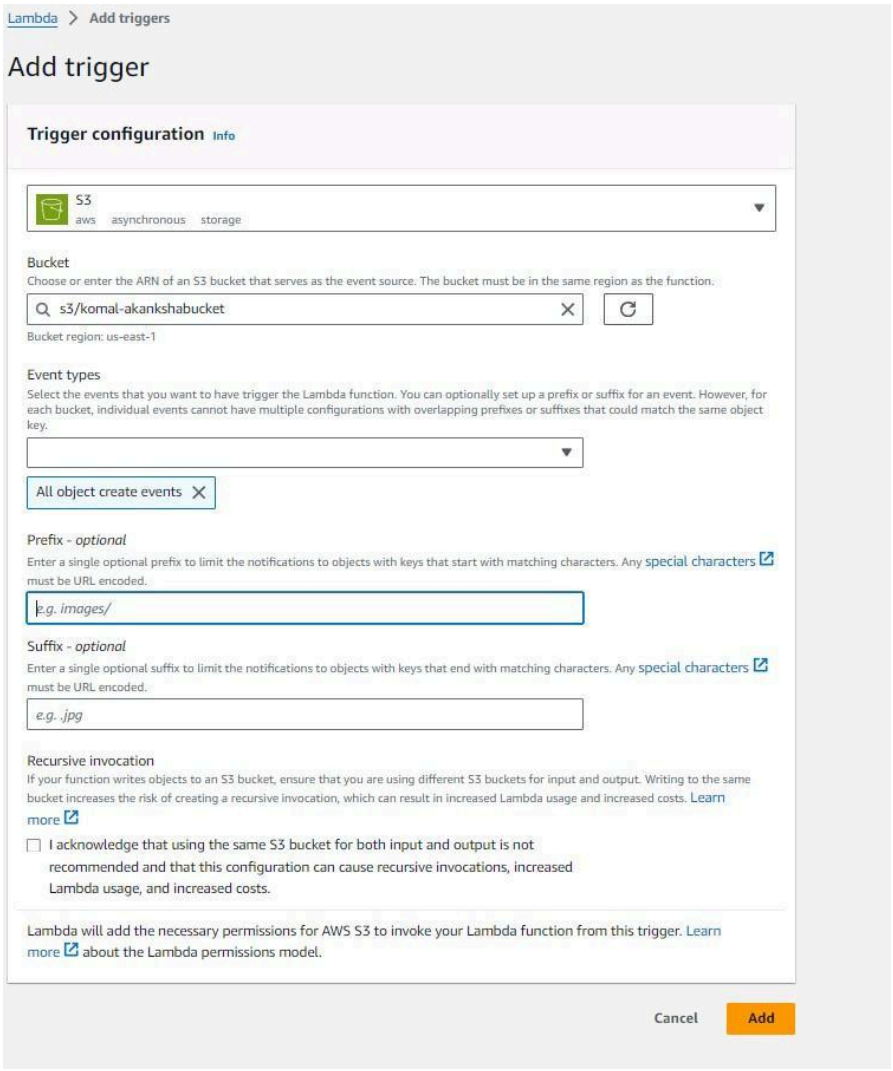
2)Create a Lambda Function.



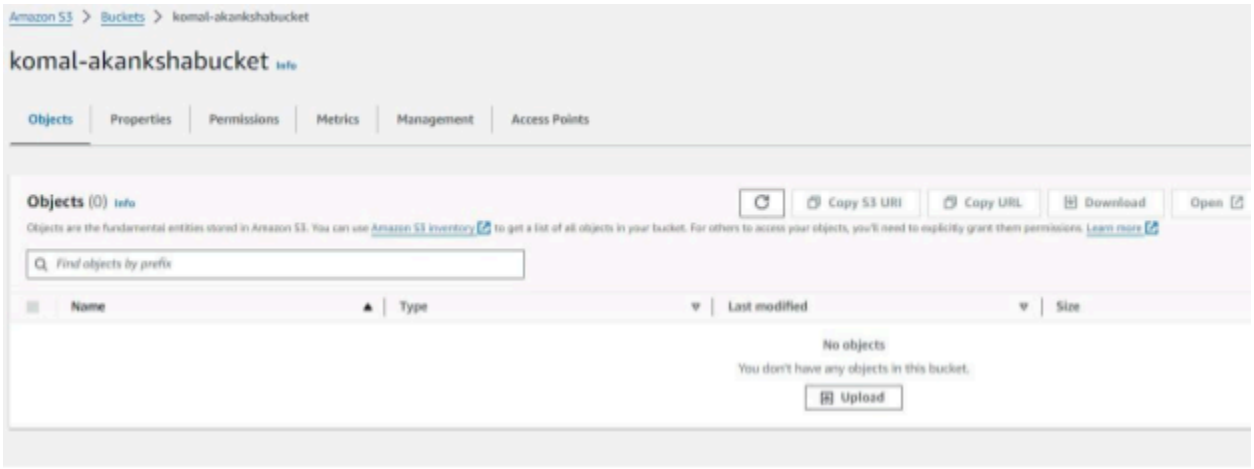


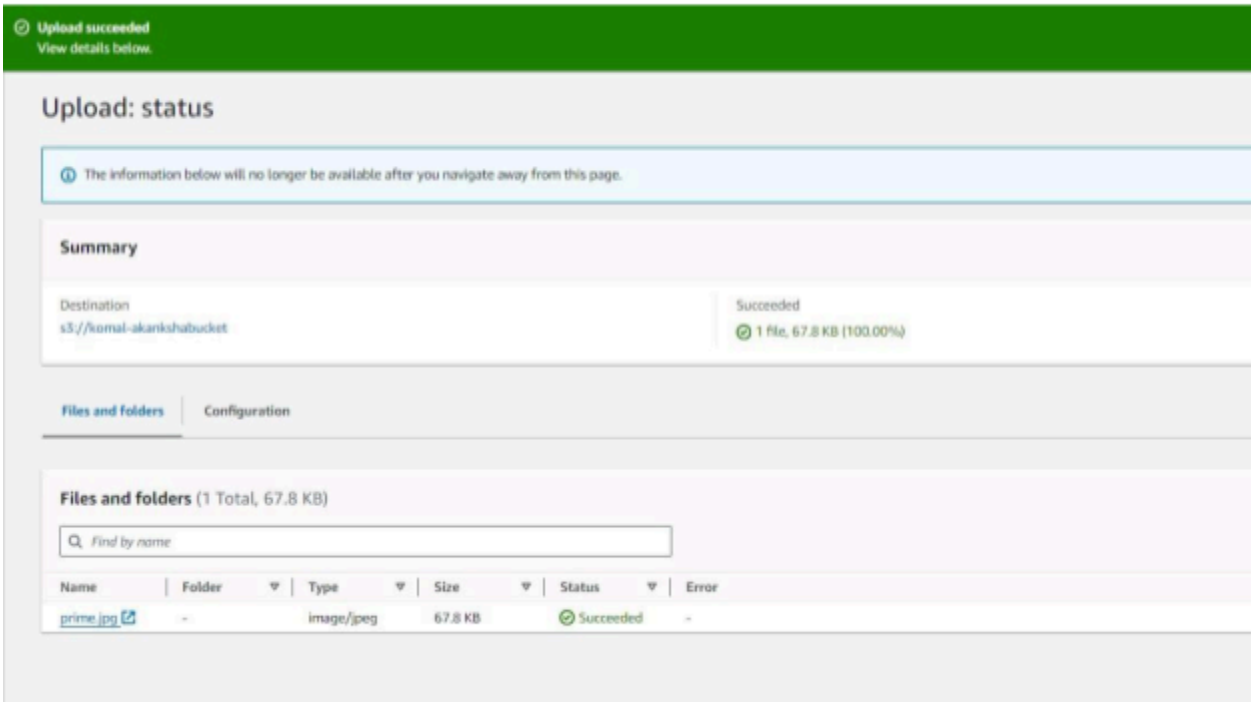
3) Add triggers.



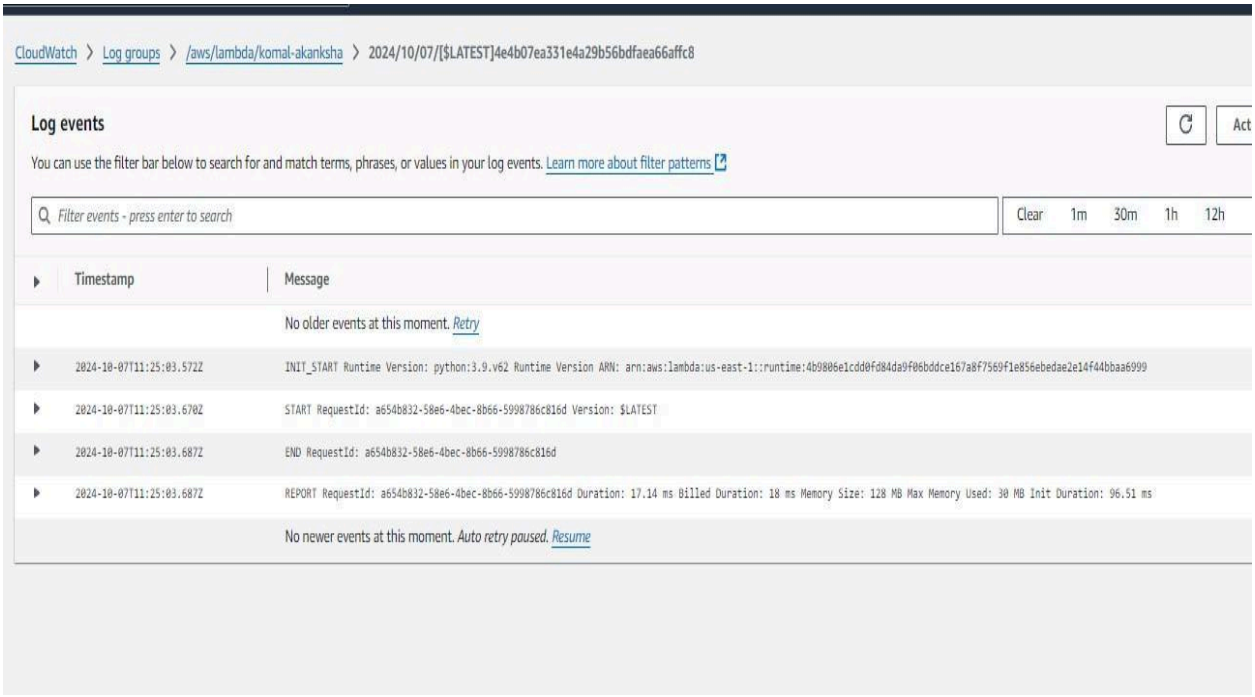


4)Go to S3 and upload an image.





5)Check the log for Lambda Function.



Conclusion:

Integrating AWS Lambda with S3 enables automated, real-time processing of events like file uploads. In this scenario, a Lambda function is set up to log a message whenever an image is uploaded to a designated S3 bucket. This integration highlights the advantages of serverless computing by automating processes without the need for manual involvement or server management. By using AWS Lambda, developers can create event-driven workflows that scale efficiently, reducing operational complexity and enabling rapid deployment of solutions that react to specific events within cloud-based environments.

Name: Akanksha Shinde Class: D15C Roll No: 53

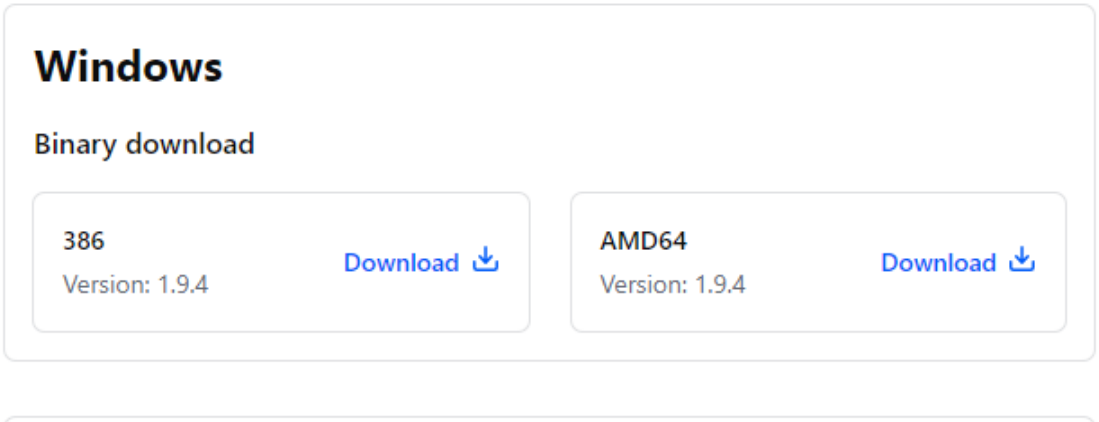
A) Installation and Configuration of Terraform in Windows

Step 1: Download terraform

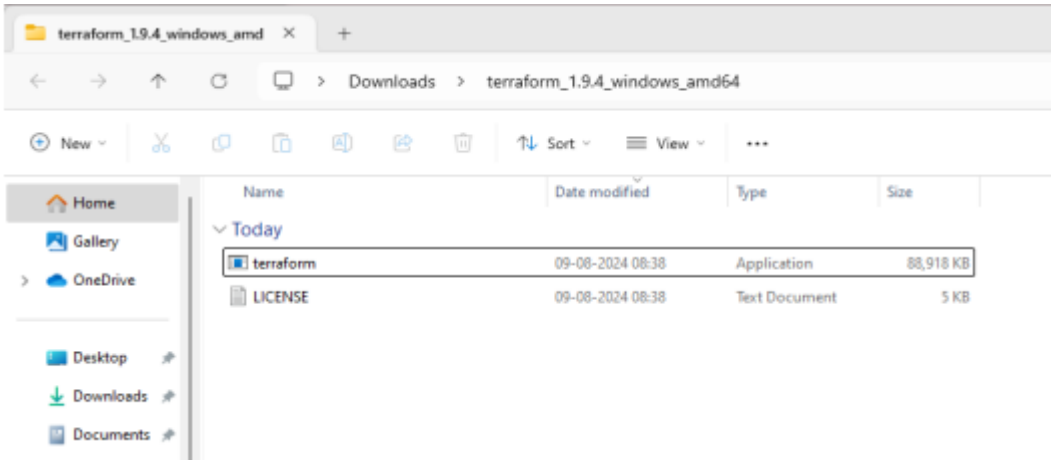
To install Terraform, First Download the Terraform Cli Utility for windows from terraforms official website

website:https://www.terraform.io/downloads.html

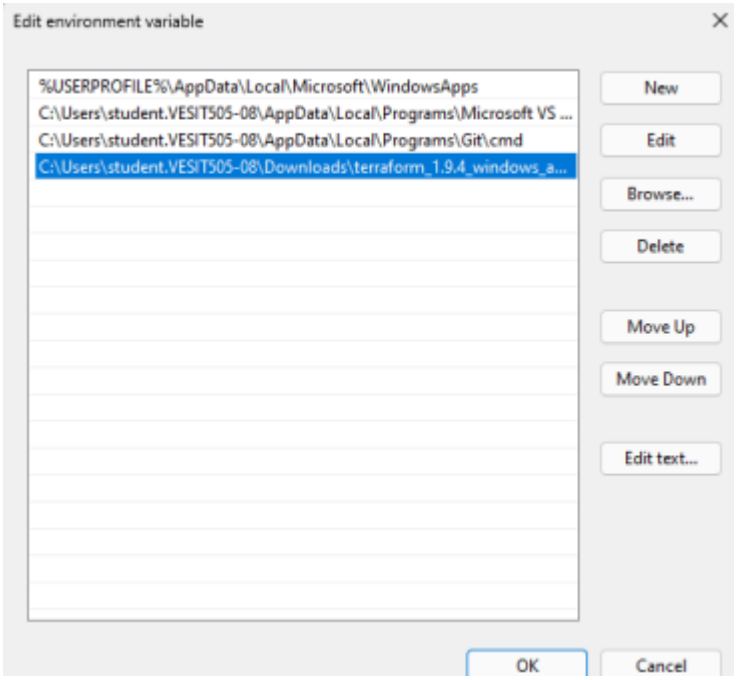
Select the Operating System Windows followed by either 32bit or 64 bit based on your OS type. Select AMD64



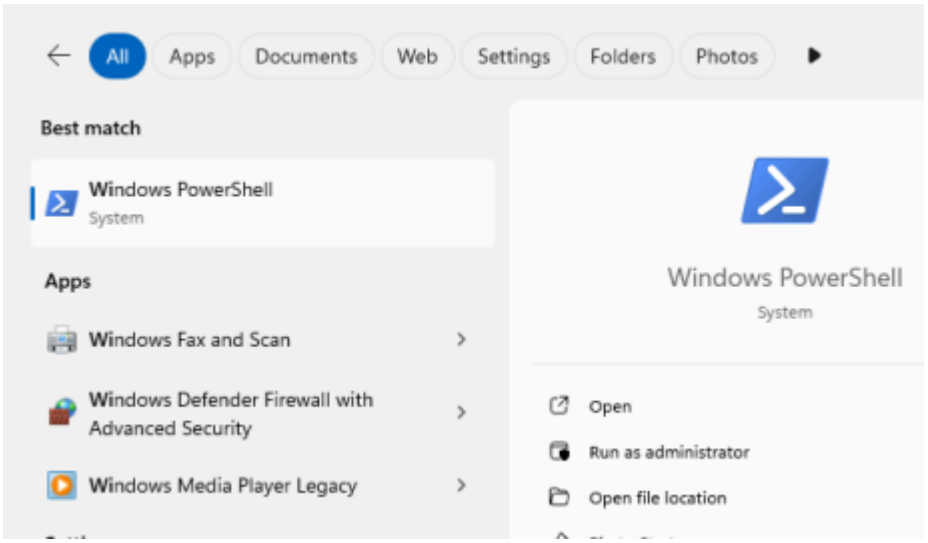
Step 2: Extract the downloaded setup file Terraform.exe in C:\Terraform directory



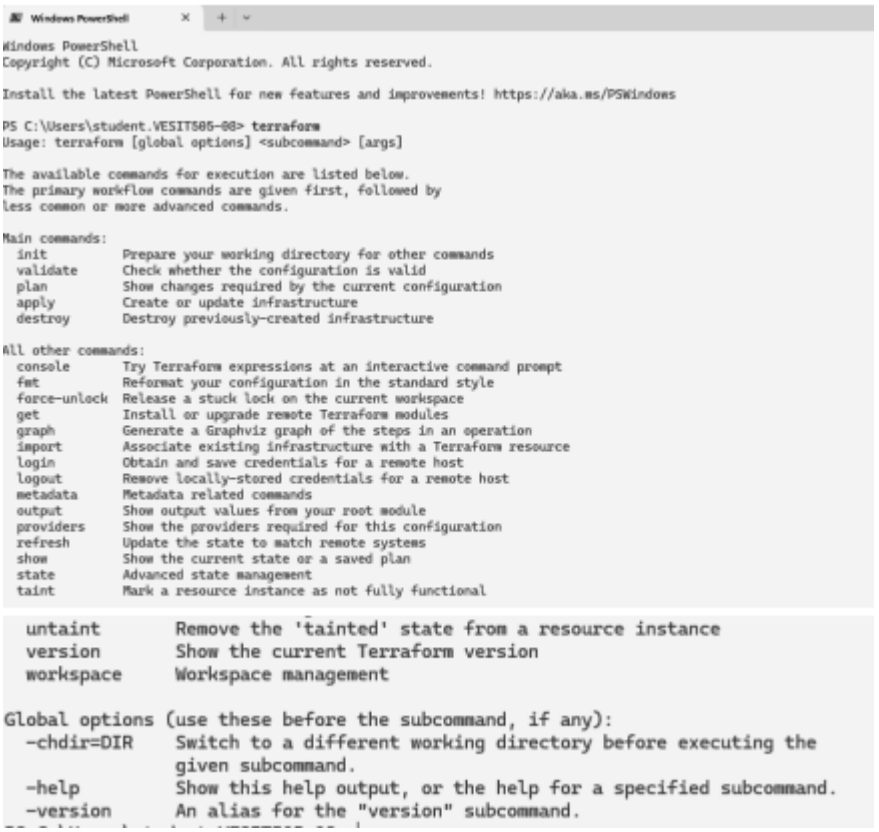
Step 3: Set the System path for Terraform in Environment Variables



Step 4: Open PowerShell with Admin Access



Step 5 : Open Terraform in command line and check its functionality



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\student.VESIT505-08> terraform
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
  metadata      Metadata related commands
  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
  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.
```

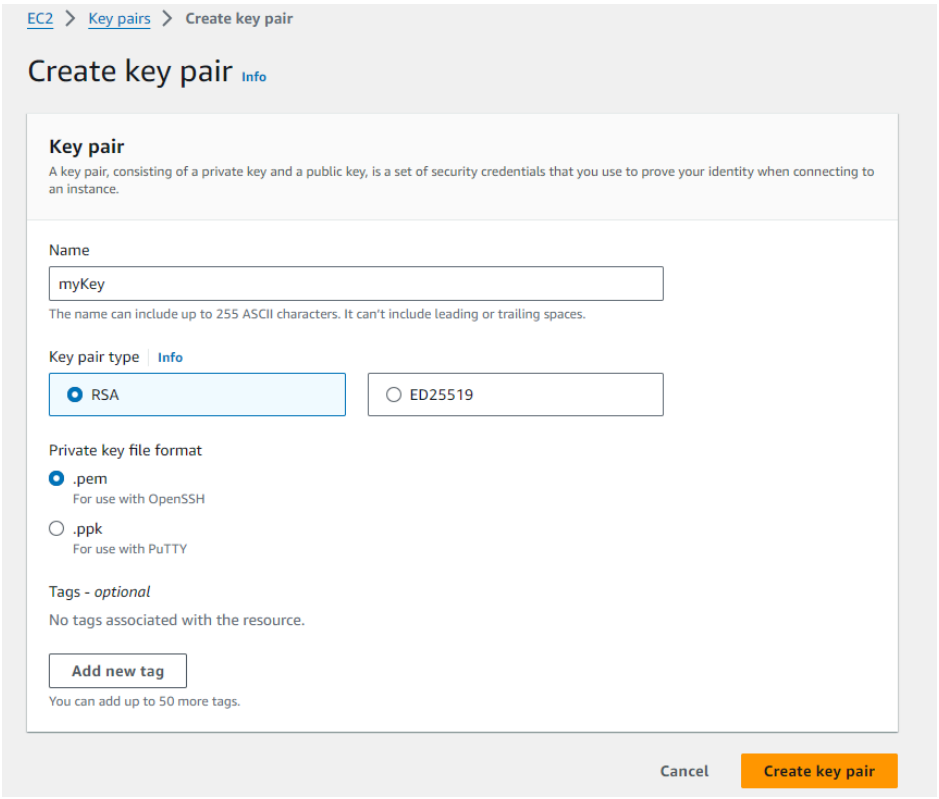
Name: Akanksha Shinde Class: D15C Roll No: 53

Experiment 4

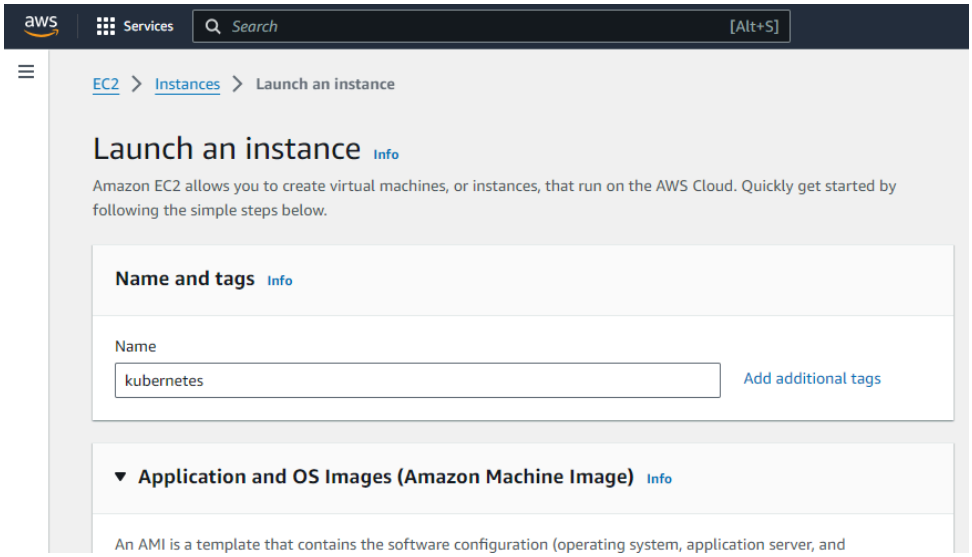
Aim: To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

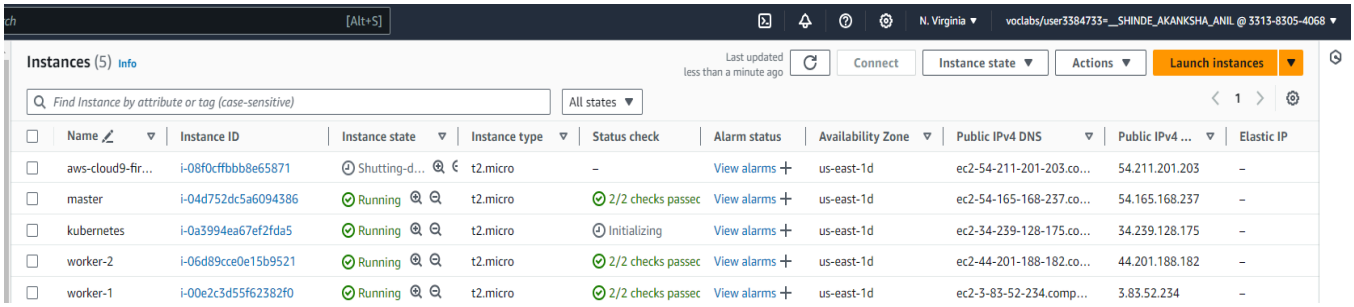
Steps:

1. Create a key pair and from Private key format select .pem. A .pem file will be downloaded in your machine.

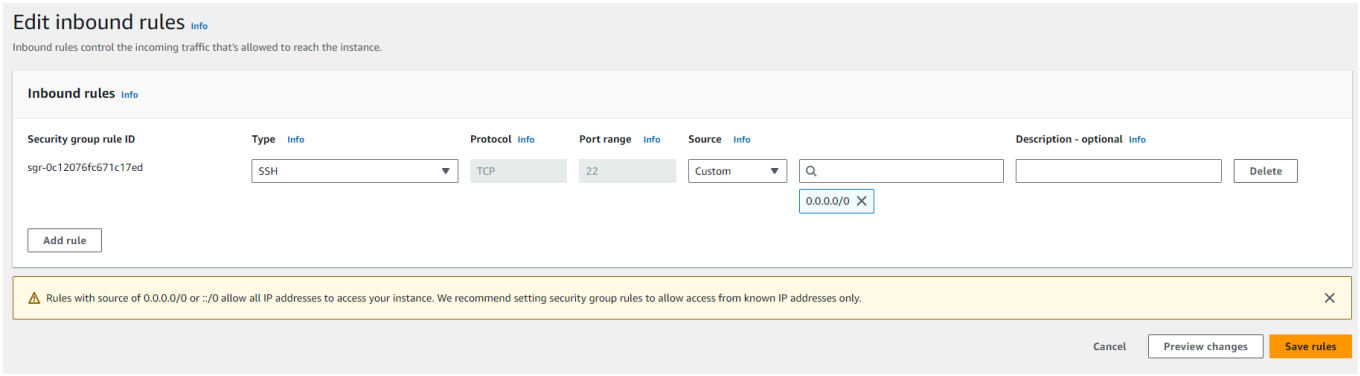


Now, Create AWS EC2 instance

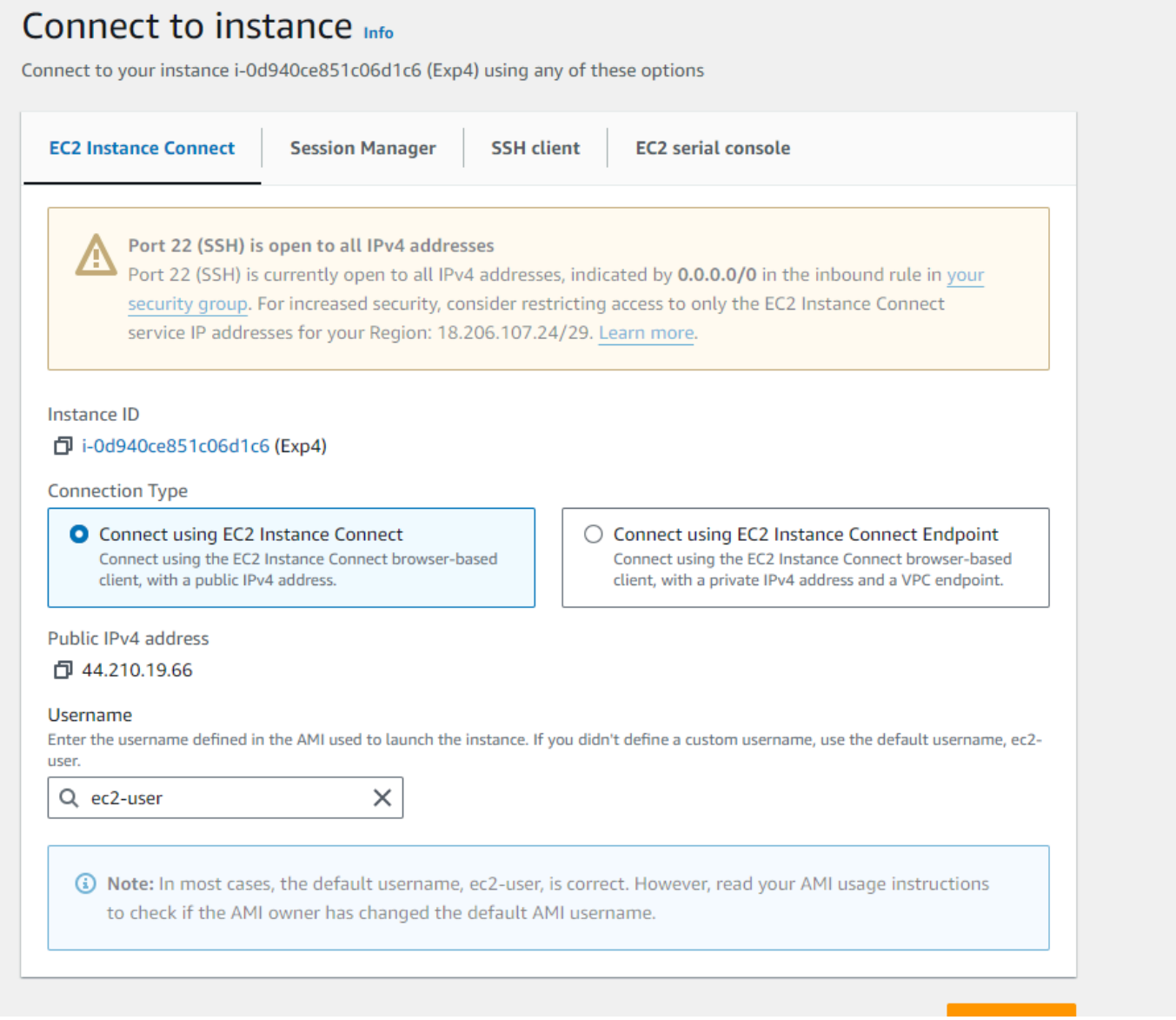




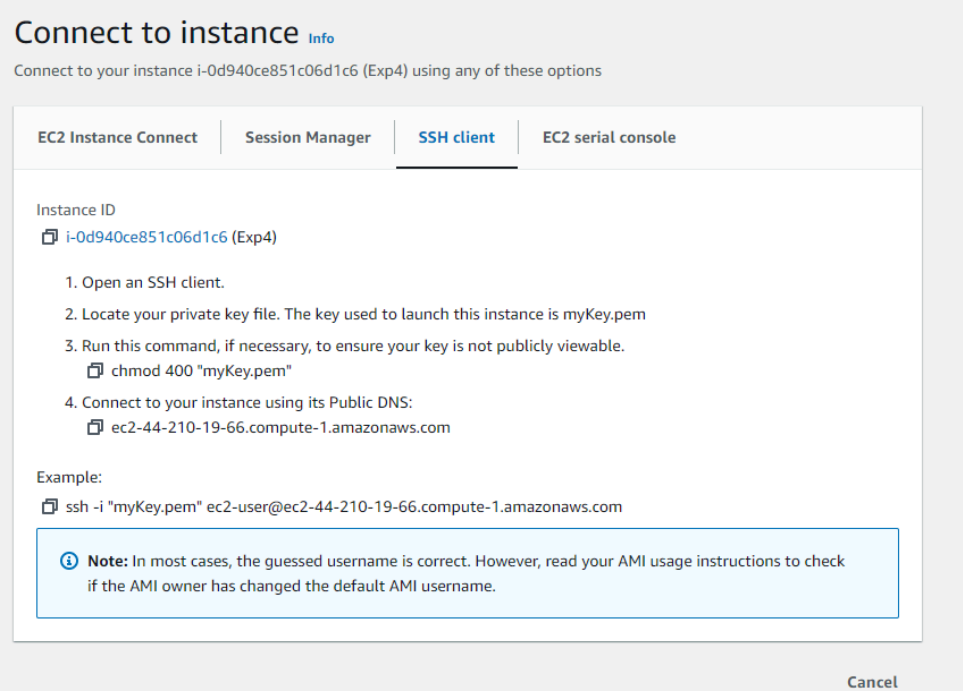
2. Edit the Security Group Inbound Rules to allow SSH



3.click on the id of the ec2 instance created and click connect you will get to see this page



Now click on the SSH client and copy the third command and the command below the example in notepad.

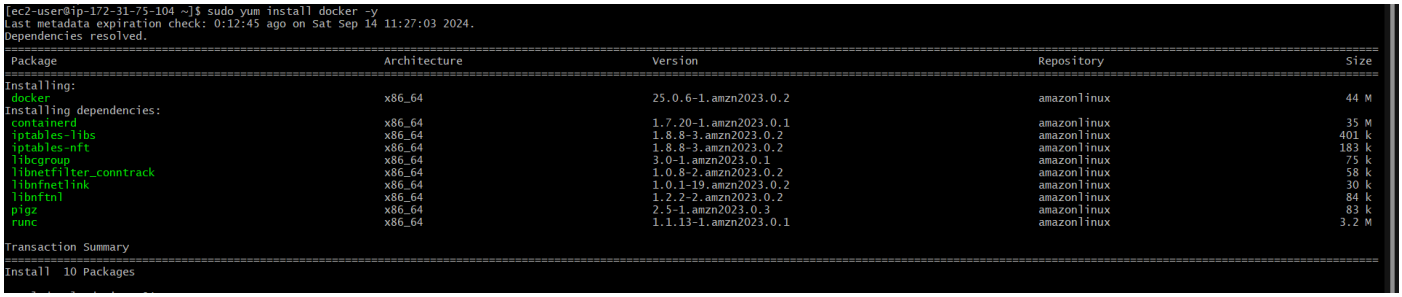


Go to git bash terminal and go to downloads where your .pem file has been downloaded and paste the copied commands.

```
Akanksha Shinde@AkankshaShinde MINGW64 ~/downloads
$ chmod 400 "myKey.pem"

Akanksha Shinde@AkankshaShinde MINGW64 ~/downloads
$ ssh -i "myKey.pem" ec2-user@ec2-44-210-19-66.compute-1.amazonaws.com
The authenticity of host 'ec2-44-210-19-66.compute-1.amazonaws.com (44.210.19.66
)' can't be established.
ED25519 key fingerprint is SHA256:i+yKgHfjRavbh4Is9J7Spd+rR/laVV+otA8HG1EvaTM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes

Warning: Permanently added 'ec2-44-210-19-66.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-75-104 ~]$
[ec2-user@ip-172-31-75-104 ~]$
```

4. Install Docker

To install the docker run this command “sudo yum install docker -y”



```
[ec2-user@ip-172-31-75-104 ~]$ sudo yum install docker -y
Last metadata expiration check: 0:12:45 ago on Sat Sep 14 11:27:03 2024.
Dependencies resolved.
================================================================================
 Package              Architecture        Version                  Repository  Size
================================================================================
Installing:
 docker               x86_64              25.0.6-1.amzn2023.0.2    amazonlinux  44 M
Installing dependencies:
 containerd           x86_64              1.7.20-1.amzn2023.0.1    amazonlinux  35 M
 iptables-libs        x86_64              1.8.8-3.amzn2023.0.2     amazonlinux 401 k
 iptables-nft         x86_64              1.8.8-3.amzn2023.0.2     amazonlinux 183 k
 libcgroup            x86_64              3.0-1.amzn2023.0.1       amazonlinux  75 k
 libnetfilter_conntrack x86_64            1.0.8-2.amzn2023.0.2     amazonlinux  58 k
 libnfnetlink         x86_64              1.0.1-19.amzn2023.0.2    amazonlinux  30 k
 libnftnl             x86_64              1.2.2-2.amzn2023.0.2     amazonlinux  84 k
 pigz                 x86_64              2.5-1.amzn2023.0.3       amazonlinux  83 k
 runc                 x86_64              1.1.13-1.amzn2023.0.1    amazonlinux 3.2 M

Transaction Summary
================================================================================
Install  10 Packages
```

Now, configure cgroup in a daemon.json file by using following commands
First perform this : cd /etc/docker
Then run this : cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
"max-size": "100m"
},
"storage-driver": "overlay2"
}
EOF

After EOF automatically the code will get completed as you hit enter after typing till EOF.

```
[ec2-user@ip-172-31-75-104 ~]$ cd /etc/docker
[ec2-user@ip-172-31-75-104 docker]$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
[ec2-user@ip-172-31-75-104 docker]$
```

After this now, s run the following command to enable and start docker and also to load the daemon.json file.
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker

```
[ec2-user@ip-172-31-75-104 docker]$ sudo systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/sys
tem/docker.service.
[ec2-user@ip-172-31-75-104 docker]$ sudo systemctl daemon-reload
[ec2-user@ip-172-31-75-104 docker]$ sudo systemctl restart docker
[ec2-user@ip-172-31-75-104 docker]$
```

Now check the version of the docker which will tell that whether you performed all tasks correctly or not. If yes then it will show the version else the otherwise.

```
[ec2-user@ip-172-31-75-104 docker]$ docker -v
Docker version 25.0.5, build 5dc9bcc
[ec2-user@ip-172-31-75-104 docker]$
```

5. Install Kubernetes

To install kubernetes disable the SELinux for doing so run the command

1. sudo setenforce 0
2. sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config

```
[ec2-user@ip-172-31-75-104 ~]$ sudo setenforce 0
[ec2-user@ip-172-31-75-104 ~]$ sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selin
ux/config
[ec2-user@ip-172-31-75-104 ~]$
```

Add kubernetes using the below commands -
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF

```
[ec2-user@ip-172-31-75-104 ~]$ cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
[ec2-user@ip-172-31-75-104 ~]$ |
```

Now, To install kubelet ,kubeadm, kubectl run the following command

1. sudo yum update
2. sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes

```
[ec2-user@ip-172-31-75-104 ~]$ sudo yum update
Kubernetes                                                                 96 kB/s |  17 kB     00:00
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-75-104 ~]$  sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Last metadata expiration check: 0:00:21 ago on Sat Sep 14 12:05:54 2024.
Dependencies resolved.
==========================================================================================================
 Package                    Architecture      Version                      Repository             Size
==========================================================================================================
Installing:
 kubeadm                    x86_64            1.30.5-150500.1.1            kubernetes             10 M
 kubectl                    x86_64            1.30.5-150500.1.1            kubernetes             10 M
 kubelet                    x86_64            1.30.5-150500.1.1            kubernetes             17 M
Installing dependencies:
 conntrack-tools            x86_64            1.4.6-2.amzn2023.0.2         amazonlinux           208 k
 cri-tools                  x86_64            1.30.1-150500.1.1            kubernetes            8.6 M
 kubernetes-cni             x86_64            1.4.0-150500.1.1             kubernetes            6.7 M
 libnetfilter_cthelper      x86_64            1.0.0-21.amzn2023.0.2        amazonlinux            24 k
 libnetfilter_cttimeout     x86_64            1.0.0-19.amzn2023.0.2        amazonlinux            24 k
 libnetfilter_queue         x86_64            1.0.5-2.amzn2023.0.2         amazonlinux            30 k

Transaction Summary
==========================================================================================================
Install  9 Packages
```

```
Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64     cri-tools-1.30.1-150500.1.1.x86_64        kubeadm-1.30.5-150500.1.1.x86_64                    kubectl-1.30.5-150500.1.1.x86_64
  kubelet-1.30.5-150500.1.1.x86_64                kubernetes-cni-1.4.0-150500.1.1.x86_64    libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64
  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64

Complete!
[ec2-user@ip-172-31-75-104 ~]$
```

Do some configurations to allow bridging.

1. sudo swapoff -a
2. echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
3. sudo sysctl -p

```
[ec2-user@ip-172-31-75-104 ~]$ sudo swapoff -a
[ec2-user@ip-172-31-75-104 ~]$ echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sys
ctl.conf
net.bridge.bridge-nf-call-iptables=1
[ec2-user@ip-172-31-75-104 ~]$ sudo sysctl -p
net.bridge.bridge-nf-call-iptables = 1
[ec2-user@ip-172-31-75-104 ~]$
```

6. Initialize the Kubecluster
Run the following command

sudo kubeadm init --pod-network-cidr=10.244.0.0/16

If you get any warning regarding the CPU or ram space just then only run the following command -

sudo kubeadm init --pod-network-cidr=10.244.0.0/16 **--ignore-preflight-errors=NumCPU,Mem**

```
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.75.104:6443 --token 5l2kvw.26ysk1yk3vx3cdm9 \
        --discovery-token-ca-cert-hash sha256:7ad09cacafa9be4798dff17aef5271313eb82ca7ea0b85a8c47
d0f671768fd56
[ec2-user@ip-172-31-75-104 ~]$
```

Copy the mkdir and chown commands from the top and execute them

```
To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Then, add a common networking plugin called flannel as mentioned in the code.
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.y
ml

```
[ec2-user@ip-172-31-75-104 ~]$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
[ec2-user@ip-172-31-75-104 ~]$
```

7. Now that the cluster is up and running, we can deploy our nginx server on this cluster.
Apply this deployment file using this command to create a deployment

kubectl apply -f https://k8s.io/examples/application/deployment.yaml

```
[ec2-user@ip-172-31-75-104 ~]$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
[ec2-user@ip-172-31-75-104 ~]$
```

Run ‘kubectl get pods’ to verify if the deployment was properly created and the pod is working correctly.

```
[ec2-user@ip-172-31-75-104 ~]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-8jdlf    0/1     Pending   0          18s
```

8. Lastly, port forward the deployment to your localhost so that you can view it.
   `kubectl port-forward $POD_NAME 8080:80`

```
[ec2-user@ip-172-31-75-104 ~]$ kubectl port-forward nginx 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
```

9. Verify your deployment
Open up a new terminal and ssh to your EC2 instance.
Then, use this curl command to check if the Nginx server is running.
curl --head http://127.0.0.1:8080
If the response is 200 OK and you can see the Nginx server name, your deployment was successful. We have successfully deployed our Nginx server on our EC2 instance.

Conclusion:

An EC2 instance was launched and I enabled SSH access by modifying the inbound rules. Docker and Kubernetes were then installed, and internet bridging was configured. After setting up the cluster, we integrated the Flannel networking plugin. With the cluster running, we deployed an Nginx server and confirmed its successful deployment. I verified the Nginx deployment using Kubernetes commands and ensured it was accessible through the configured port. The entire setup demonstrated successful

integration of Docker, Kubernetes, and networking components within the EC2 environment.

Name: Akanksha Shinde Class: D15C Roll No: 53

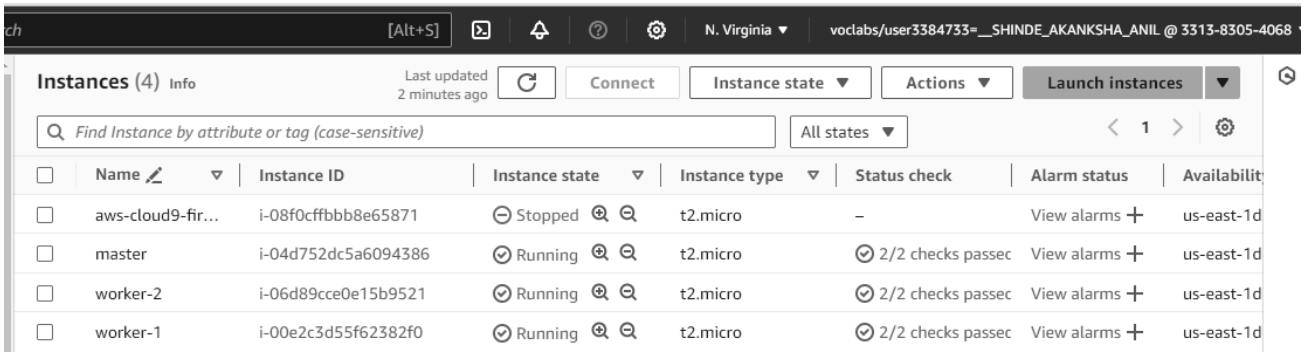
Experiment: 3

Aim: To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

Steps:

1. Create 3 EC2 Ubuntu Instances on AWS.

(Name 1 as Master, the other 2 as worker-1 and worker-2)



2. Edit the Security Group Inbound Rules to allow SSH and do it for all the three machines.

```
Akanksha Shinde@AkankshaShinde MINGW64 ~/downloads (master)
$ ssh -i "server.pem" ec2-user@ec2-54-174-206-93.compute-1.amazonaws.com
The authenticity of host 'ec2-54-174-206-93.compute-1.amazonaws.com (54.174.206.
93)' can't be established.
ED25519 key fingerprint is SHA256:T+tsGyI15gAvUvjeAZ7GjDIWXHOaI4EPF5g5oICrkoQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-174-206-93.compute-1.amazonaws.com' (ED25519)
 to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
```

3. From now on, until mentioned, perform these steps on all 3 machines. Install Docker for all the 3 machines

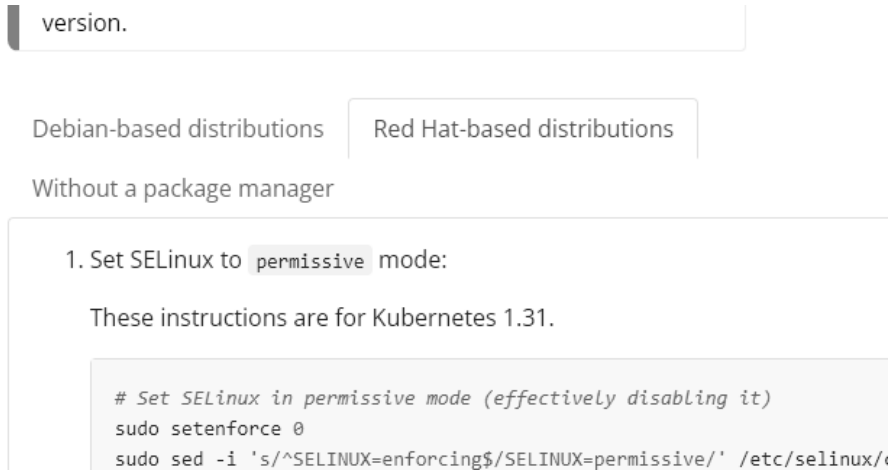
```
[root@ip-172-31-90-172 ec2-user]# yum install docker -y
Last metadata expiration check: 0:21:16 ago on Fri Aug 30 04:01:12 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture            Version
================================================================================
Installing:
 docker                  x86_64                  25.0.6-1.amzn2023.0.1
Installing dependencies:
 containerd              x86_64                  1.7.20-1.amzn2023.0.1
 iptables-libs           x86_64                  1.8.8-3.amzn2023.0.2
 iptables-nft            x86_64                  1.8.8-3.amzn2023.0.2
 libcgroup               x86_64                  3.0-1.amzn2023.0.1
 libnetfilter_conntrack  x86_64                  1.0.8-2.amzn2023.0.2
 libnfnetlink            x86_64                  1.0.1-19.amzn2023.0.2
 libnftnl                x86_64                  1.2.2-2.amzn2023.0.2
 pigz                    x86_64                  2.5-1.amzn2023.0.3
```

Start the docker by running the command systemctl start docker in the terminal of all the ec2 instance.

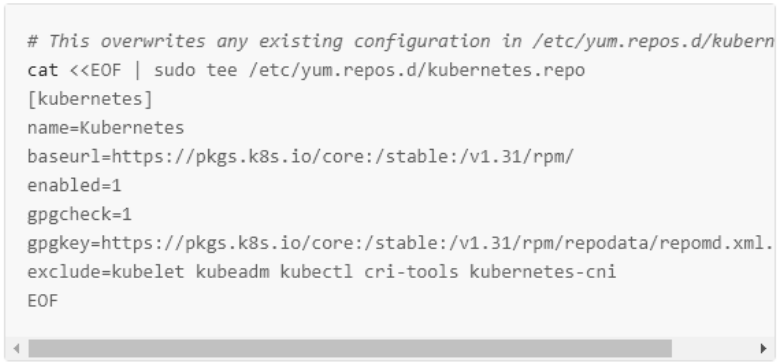
```
Complete!
[root@ip-172-31-82-133 ec2-user]# systemctl start docker
[root@ip-172-31-82-133 ec2-user]#
```

4. Install the kubernetes on all 3 machines by searching for kubeadm and click on install kubernetes.

Select the red hat based distribution. This process will automatically disable SELinux before configuring kubelet so no need to run it separately in terminal**.**



Copy the below script, to install kubernetes we need a kubernetes repo so this script helps us in getting that and paste it in the terminal.



```
Installed:
  containerd-1.7.20-1.amzn2023.0.1.x86_64       docker-25.0.6-1.amzn2023.0.1.x86_64      iptables-libs-1
  iptables-nft-1.8.8-3.amzn2023.0.2.x86_64      libcgroup-3.0-1.amzn2023.0.1.x86_64      libnetfilter_cc
  libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64     libnftnl-1.2.2-2.amzn2023.0.2.x86_64     pigz-2.5-1.amzr
  runc-1.1.11-1.amzn2023.0.1.x86_64

Complete!
[root@ip-172-31-90-172 ec2-user]# systemctl start docker
[root@ip-172-31-90-172 ec2-user]# sudo su
[root@ip-172-31-90-172 ec2-user]# yum repolist
repo id                                                    repo name
amazonlinux                                                Amazon Linux 2023 repository
```

Run the command yum repolist to check whether the kubernetes repo has installed or not if successful installed then you can see a repo named as kubernetes

```
[root@ip-172-31-90-172 ec2-user]# yum repolist
repo id                                                    repo name
amazonlinux                                                Amazon Linux 2023 repository
kernel-livepatch                                           Amazon Linux 2023 Kernel Livepatch repository
kubernetes                                                 Kubernetes
[root@ip-172-31-90-172 ec2-user]#
```

Do the above steps for all the instances i.e for worker-1 and worker-2.

5. Perform this ONLY on the Master machine. Initialize the Kubecluster

sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all

```
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.12.28:6443 --token 4bqwb8.lua2ud01lr02uu55 \
    --discovery-token-ca-cert-hash sha256:b4edc7948be9bca50767f623b58e0612feedc144a7364f95be8dbd8c4614a169
```

Copy the join command and keep it in a notepad, we'll need it later.

Copy the mkdir and chown commands from the top and execute them

```
[ec2-user@ip-172.31.12.28 docker]$ mkdir -p $HOME/.kube
sudo cp -i  /etc/kubernetes/admin.conf  $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Then, add a common networking plugin called flammel file as mentioned in the code.

kubectl apply -f

https://raw.githubusercontent.com/coreos/flannel/master/Documentation/ k ube-flannel.yml

```
[ec2-user@ip-172.31.12.28 docker]$ kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
```

Check the created pod using this command

Now, keep a watch on all nodes using the following command - watch kubectl get nodes

6. Perform this ONLY on the worker machines
   Run the following command
   sudo yum install iproute-tc -y
   sudo systemctl enable kubelet
   sudo systemctl restart kubelet

Check the status of the pods using the following command
This command will show the status of all the pods.
kubectl get pods -n kube-system
Following command will show the status of the pod named daemonset.
kubectl get daemonset -n kube-system

```
[ec2-user@ip-172.31.12.28 docker]$ kubectl get pods -n kube-system
NAME                                                READY   STATUS             RESTARTS   AGE
coredns-55cb5b8774-fx12f                            1/1     Running            0          100s
coredns-55cb5b8774-xn14v                            1/1     Running            0          100s
etcd-ip-172.31.12.28.ec2.internal                   1/1     Running            0          75s
kube-apiserver-ip-172.31.12.28.ec2.internal         1/1     Running            1          2m
kube-controller-manager-ip-172.31.12.28.ec2.internal 0/1 CrashLoopBackOff   1              70s
kube-proxy-4dv8m                                    1/1     Running            2          100s
kube-scheduler-ip-172.31.12.28.ec2.internal         1/1     Running            1          76s

[ec2-user@ip-172.31.12.28 docker]$ kubectl get daemonset -n kube-system
NAME         DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
kube-proxy   1         1         1       1            1           kubernetes.io/os=linux   3m
```

That's it, we now have a Kubernetes cluster running across 3 AWS EC2 Instances. This cluster can be used to further deploy applications and their loads being distributed across these machines.

Conclusion:

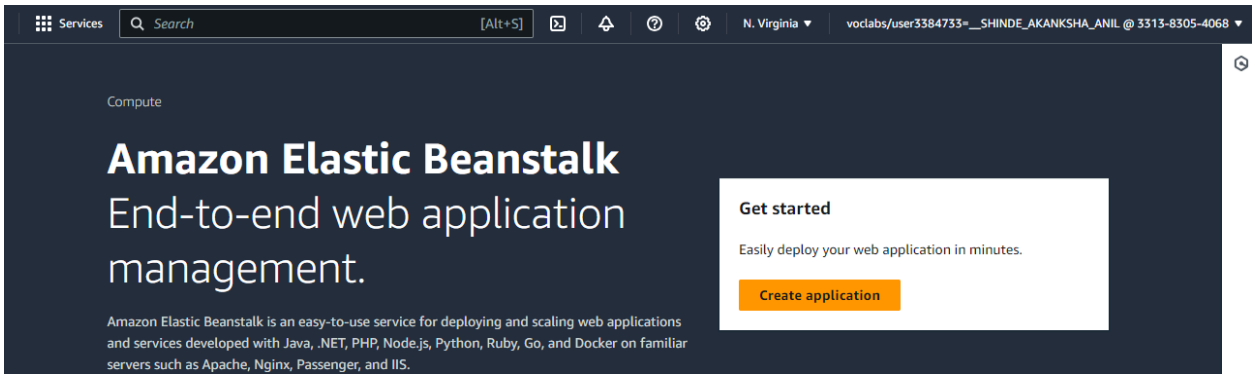
Kubernetes cluster was successfully established using three AWS EC2 instances, which includes one Master and two Worker nodes. The process began with the creation of instances and configuration of settings to begin the communication. Docker was installed on all machines followed by the installation of Kubernetes components and the necessary repositories. The Master node was initialized with the `kubeadm init` command, and a plugin called Flannel was deployed to enable pod communication. Performing correct commands on the Worker nodes ensured they joined the cluster effectively. Also necessary commands confirmed the status of pods which indicated the proper working. Overall, the experiment provided information about working of the deployment and management of containerized applications in a distributed environment.

Name: Akanksha Shinde Class: D15C Roll No: 53

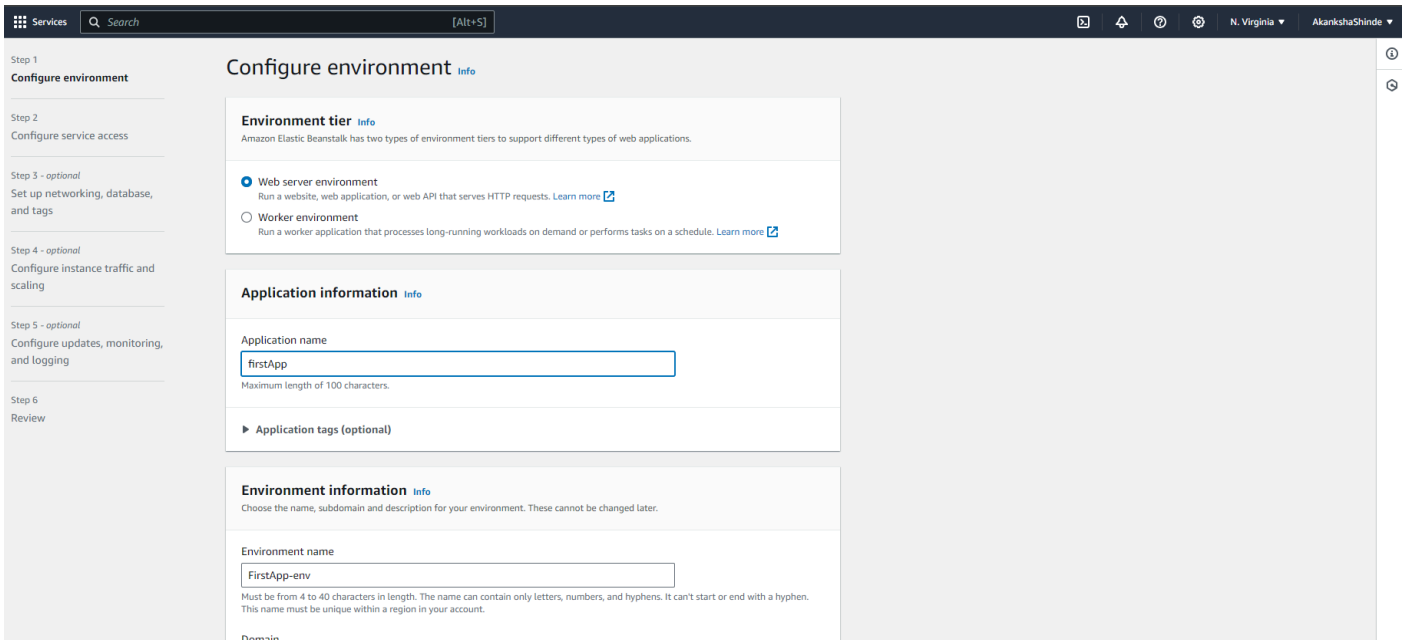
Experiment No: 2

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

1. Login to your AWS account and search for Elastic Beanstalk in the search box.

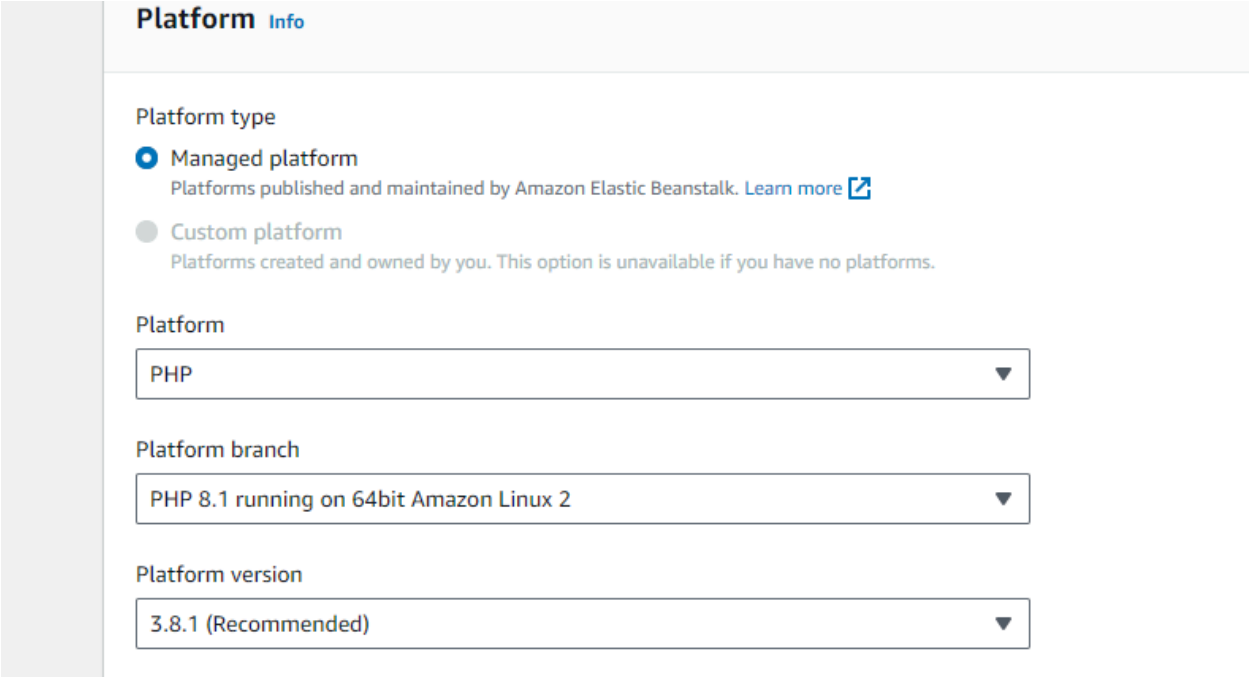


2. Open up Elastic Beanstalk and name your web app.



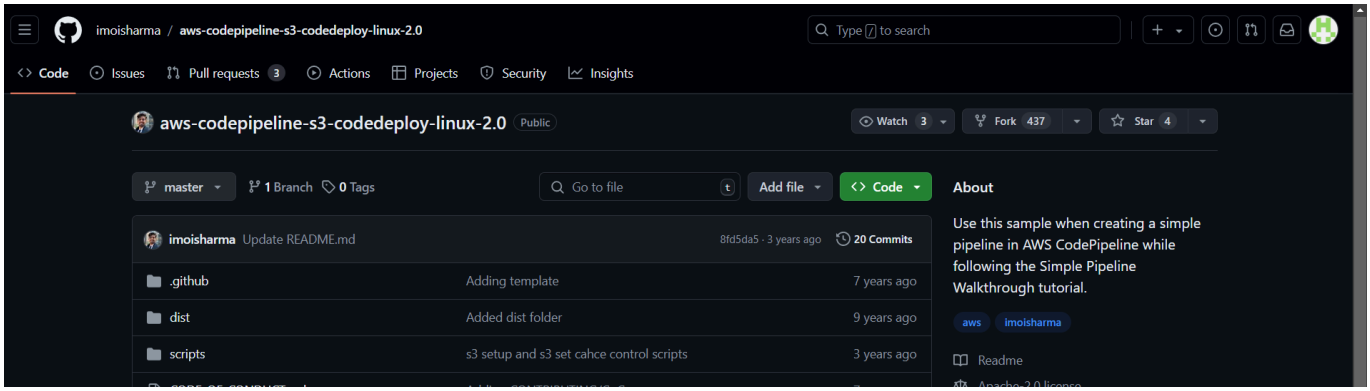
Name: Akanksha Shinde Class: D15C Roll No: 53

3. Choose PHP from the drop-down menu and then click Create Application.



4. Beanstalk creates a sample environment for you to deploy your application. By default, it creates an EC2 instance, a security group, an Auto Scaling group, an Amazon S3 Bucket, Amazon CloudWatch alarms and a domain name for your Application.

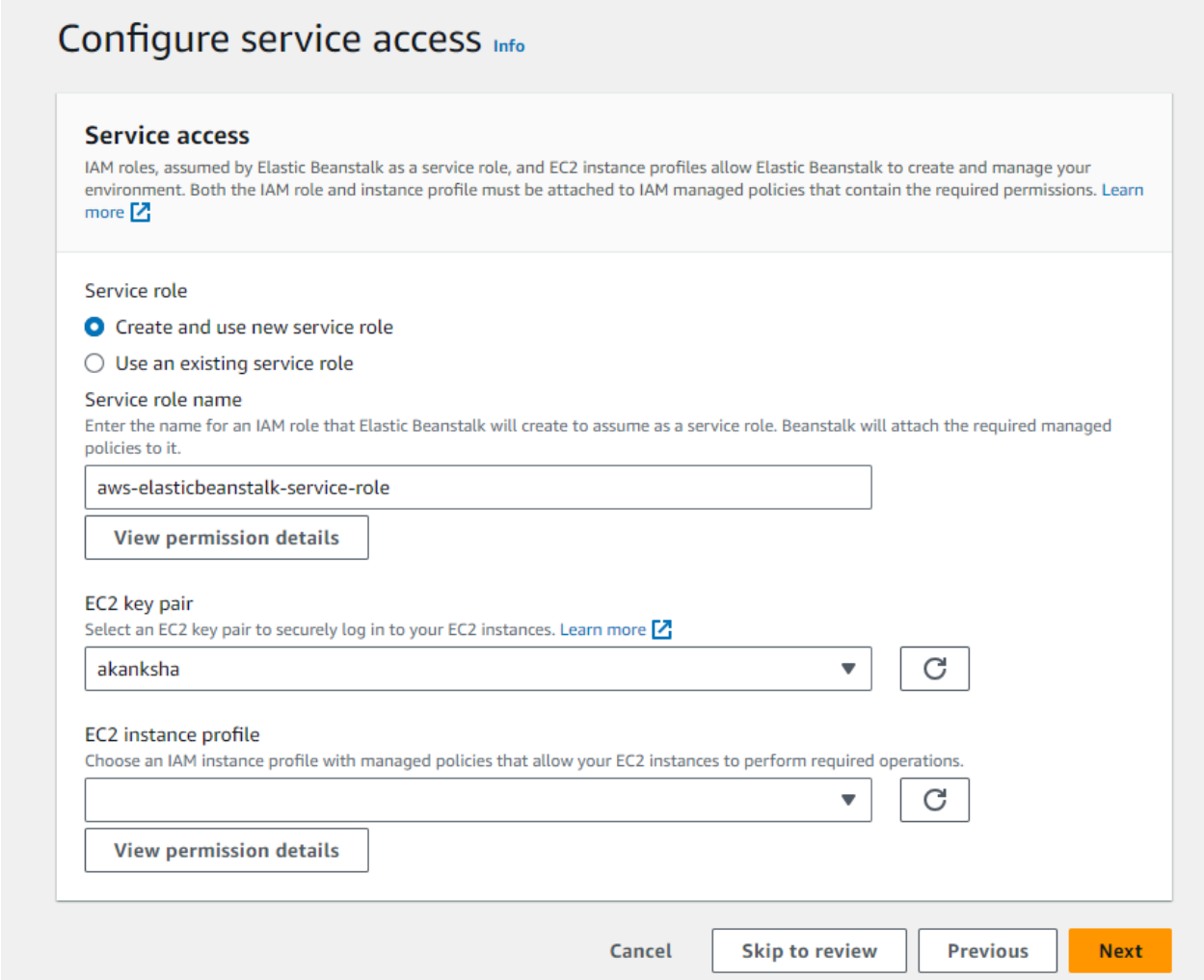
Step 2: Get a copy of your sample code



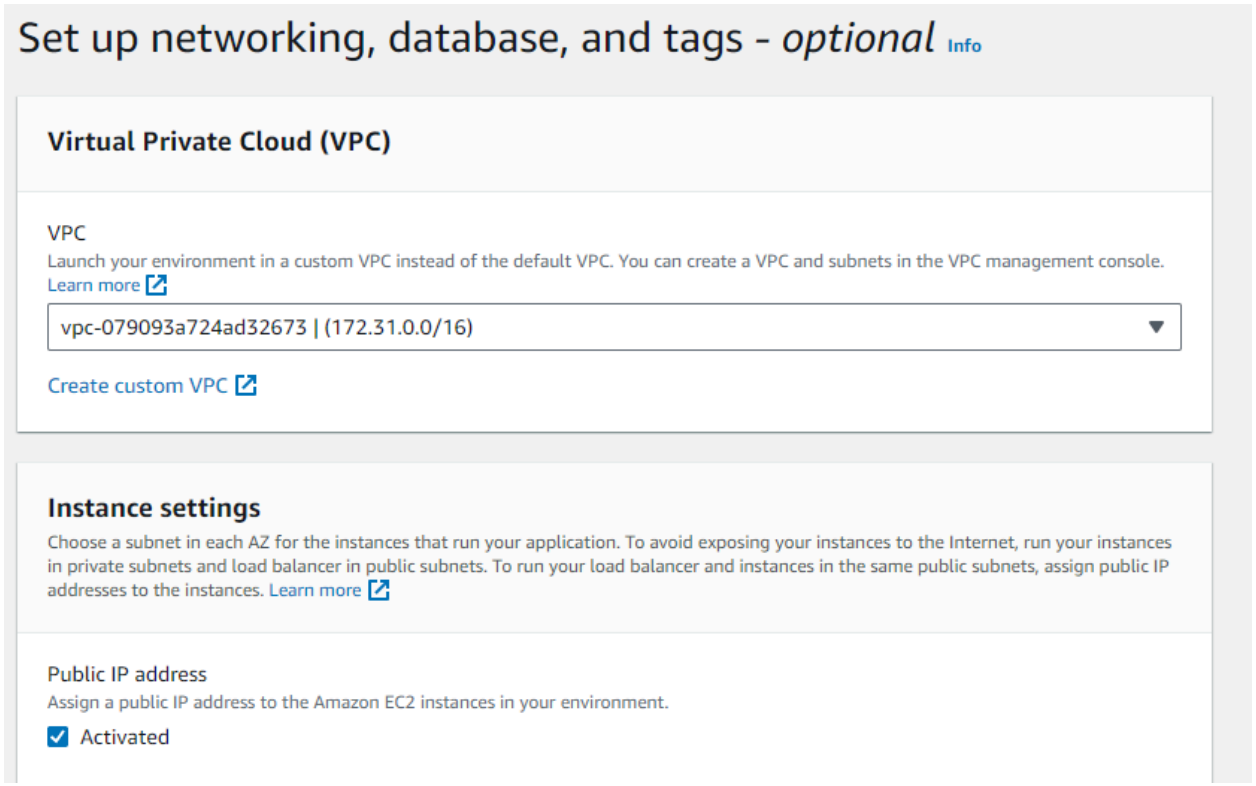
In this step, we will get the sample code from this GitHub Repository to later host it. The pipeline takes code from the source and then performs actions on it.
For this experiment, as a source, we will use this forked GitHub repository. We can alternatively also use Amazon S3 and AWS CodeCommit.

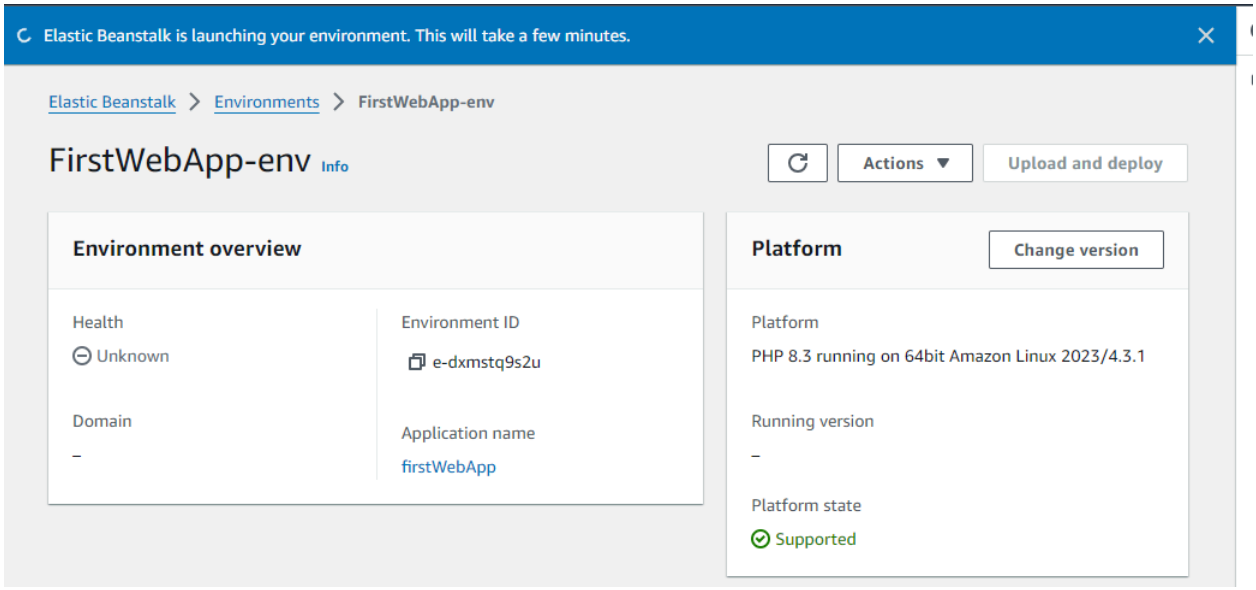
Name: Akanksha Shinde Class: D15C Roll No: 53

Go to the repository shared above and simply fork it.

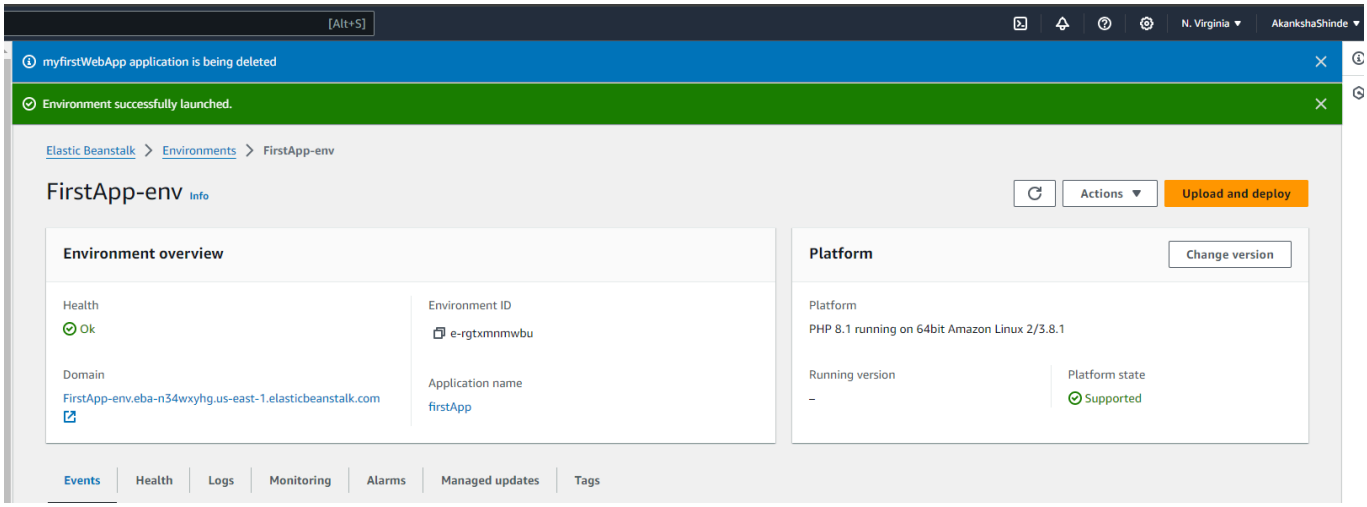


Name: Akanksha Shinde Class: D15C Roll No: 53





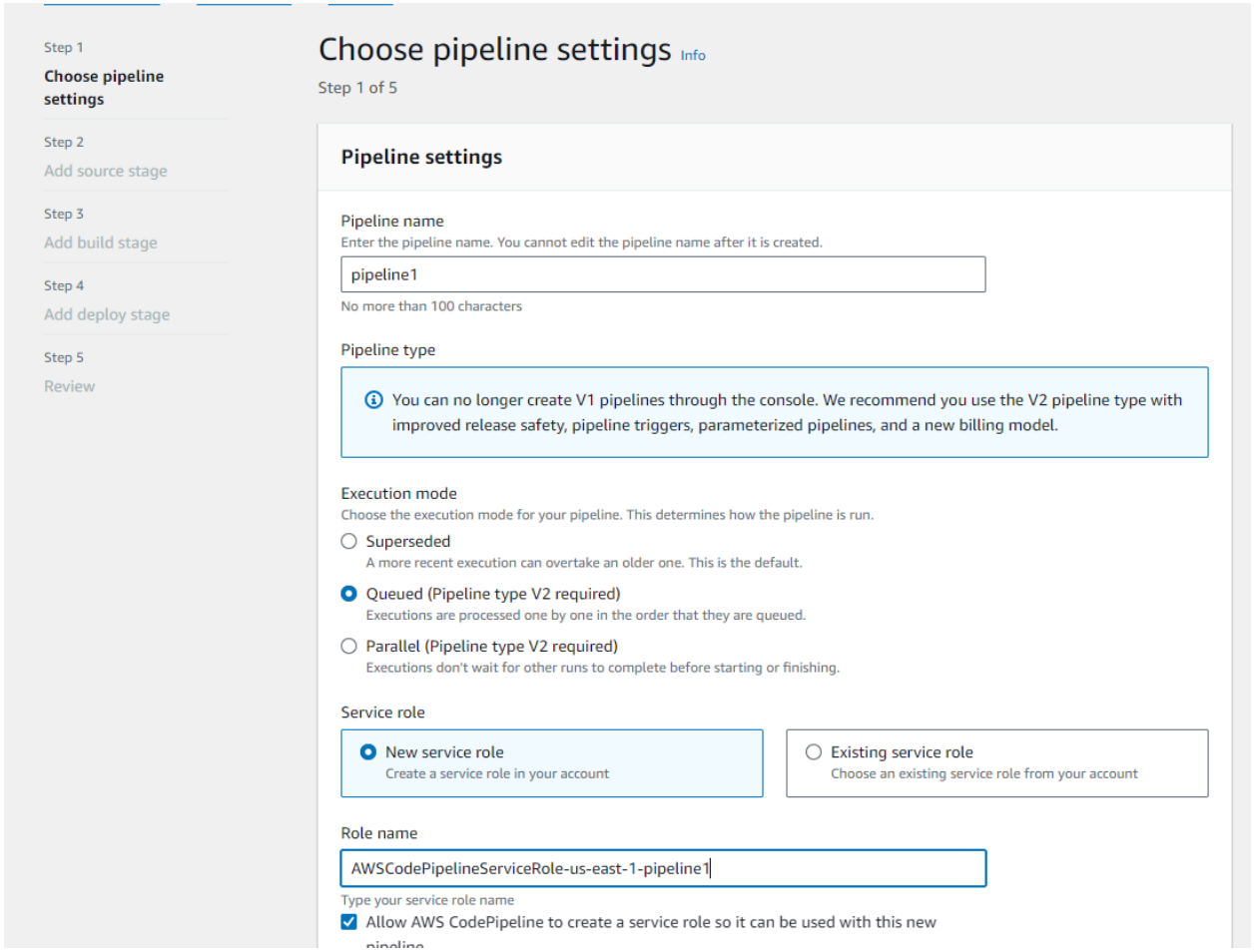
Name: Akanksha Shinde Class: D15C Roll No: 53



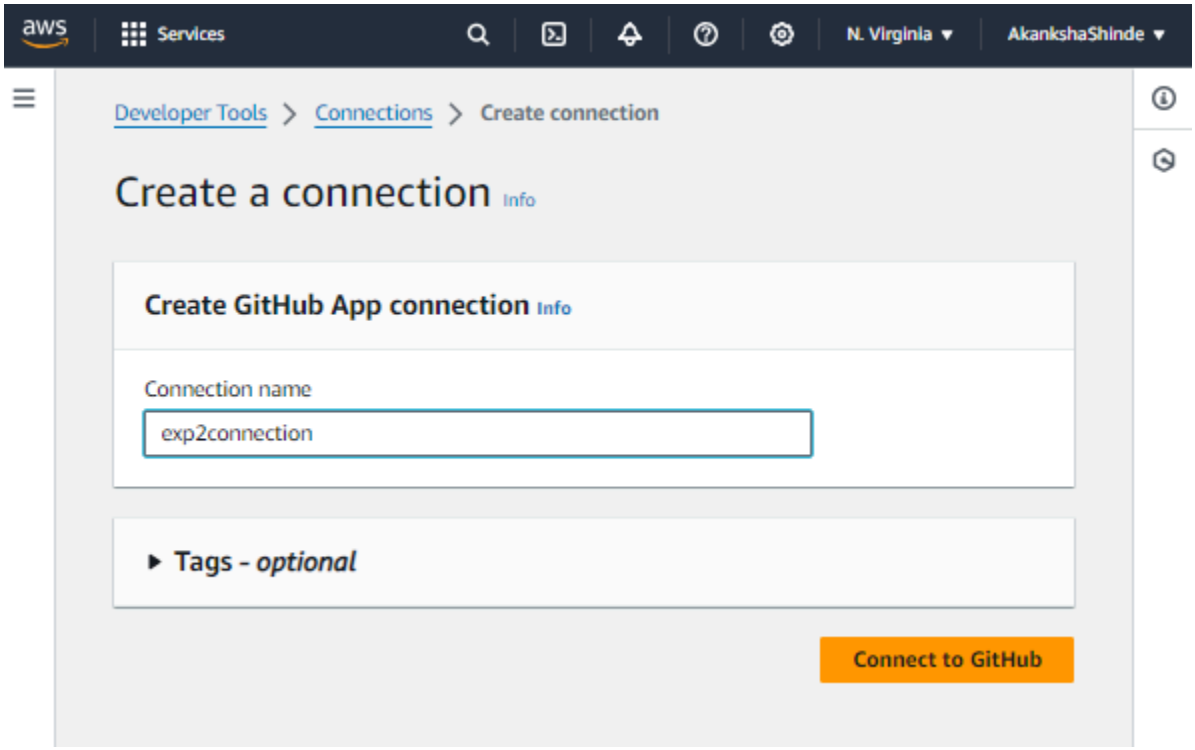
Now to create pipeline,

1. Go to AWS Developer Tools -> CodePipeline and create a new Pipeline.

Fill in the initial settings first.

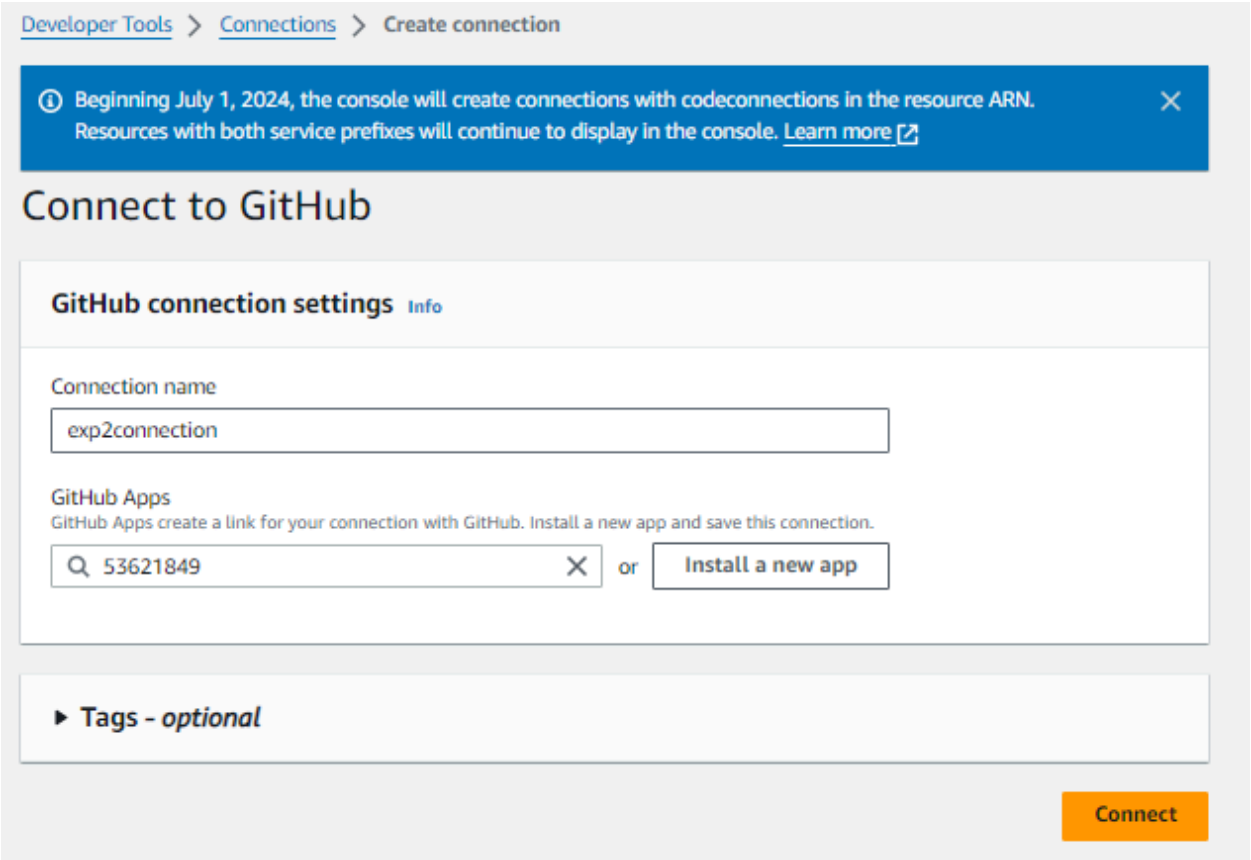


Name: Akanksha Shinde Class: D15C Roll No: 53

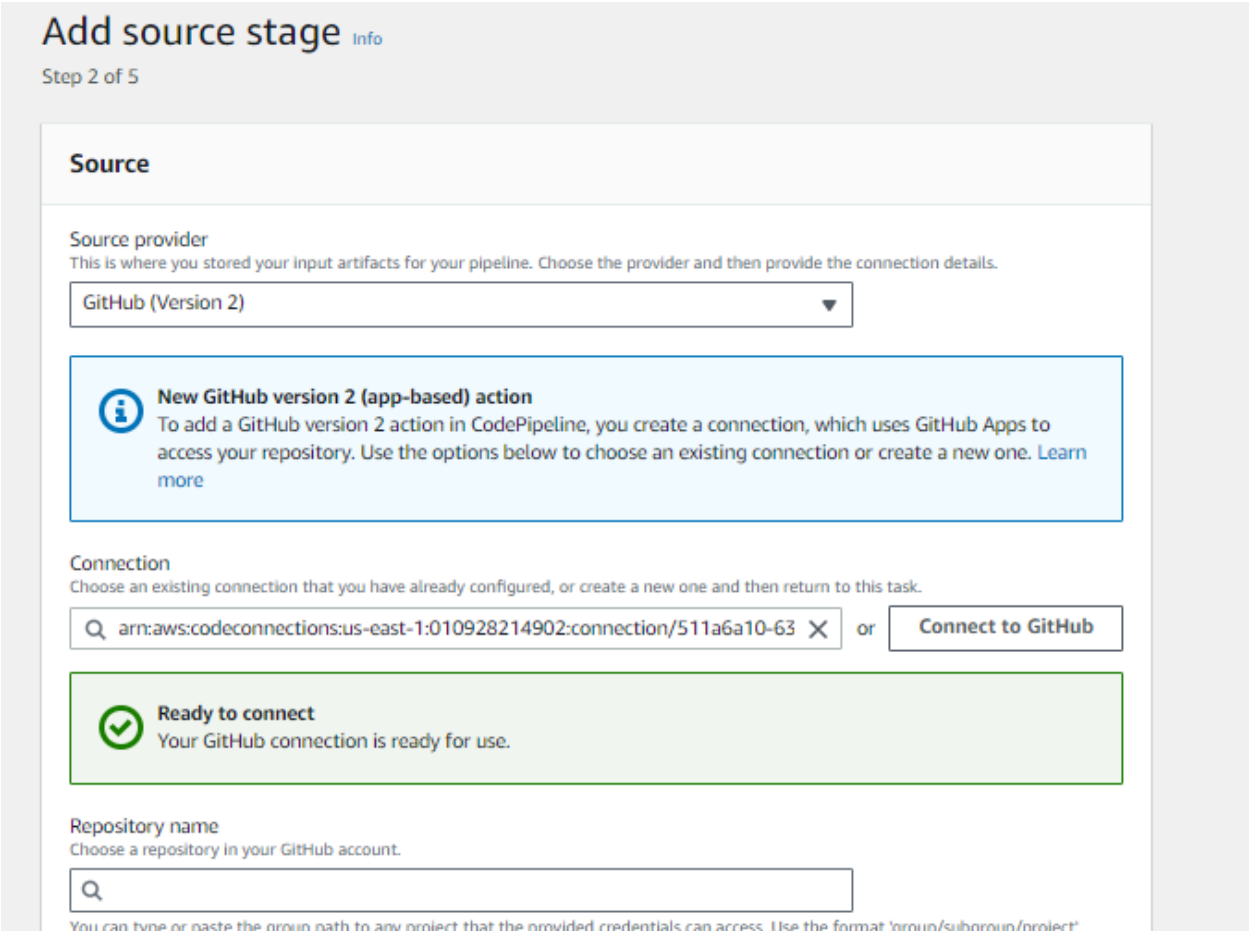


2. In the source stage, choose GitHub v2 as the provider, then connect your GitHub account to AWS by creating a connection. You'd need your GitHub credentials and then you'd need to authorize and install AWS on the forked GitHub Repository.

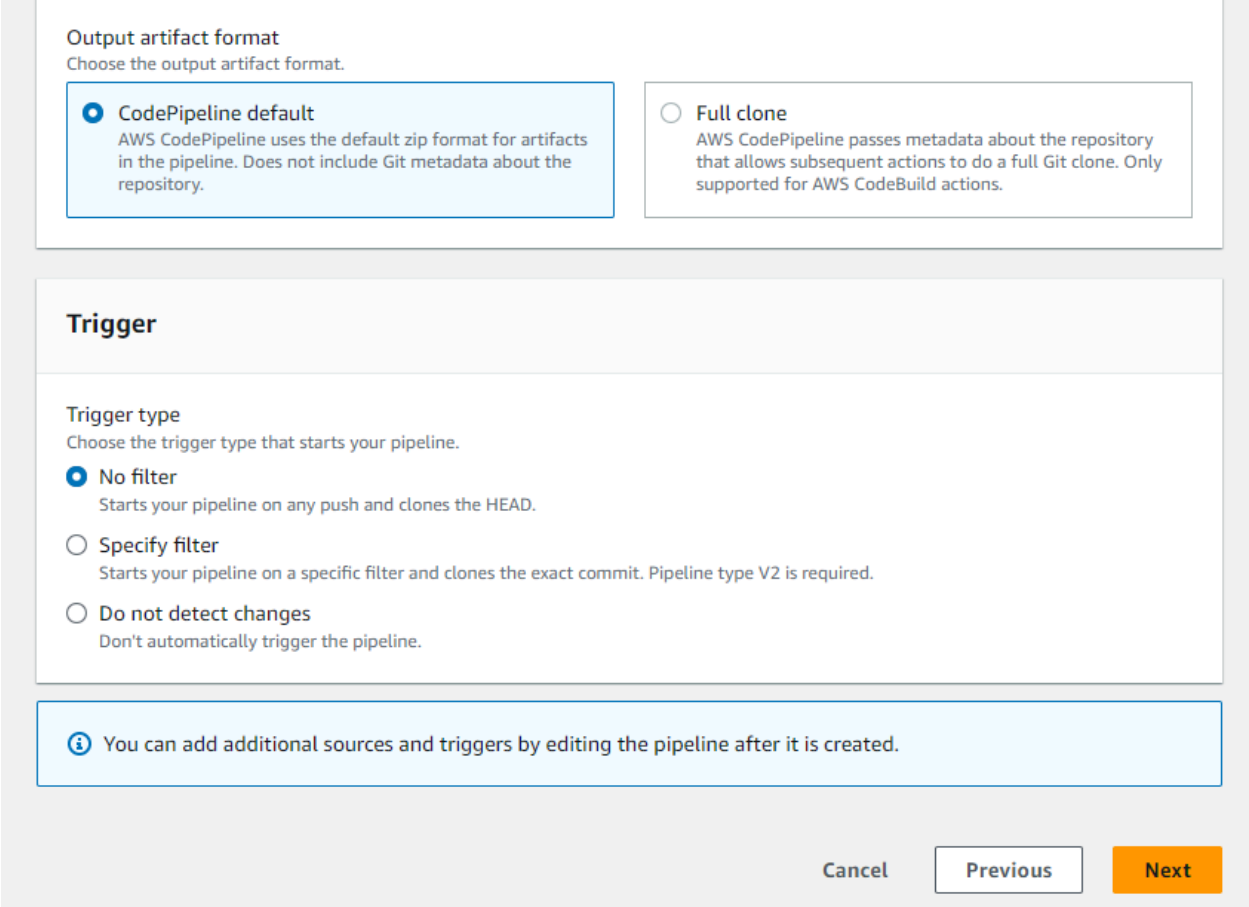
Name: Akanksha Shinde Class: D15C Roll No: 53



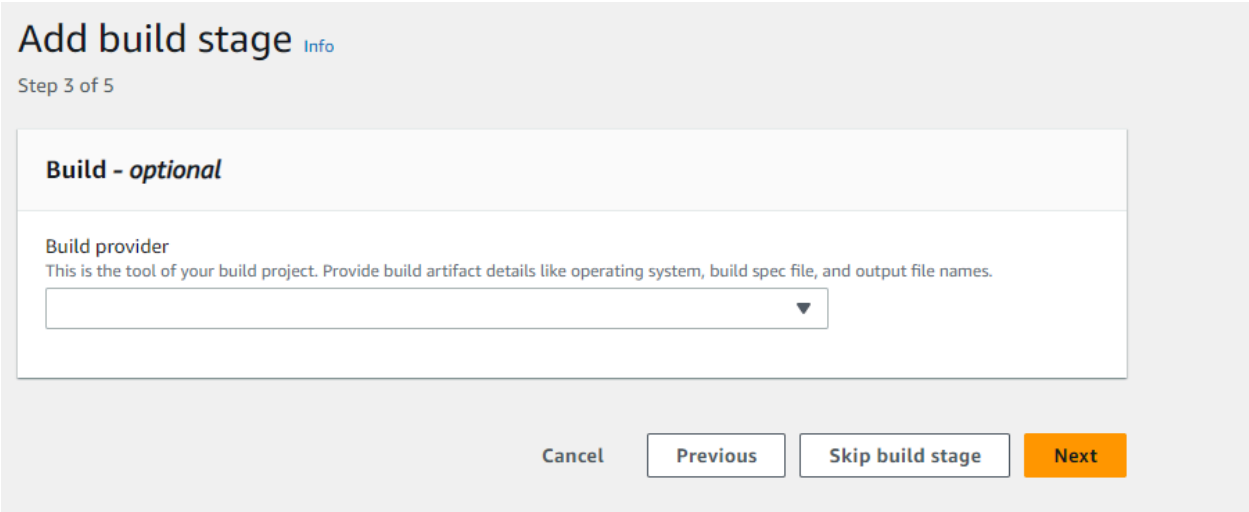
Name: Akanksha Shinde Class: D15C Roll No: 53



Name: Akanksha Shinde Class: D15C Roll No: 53



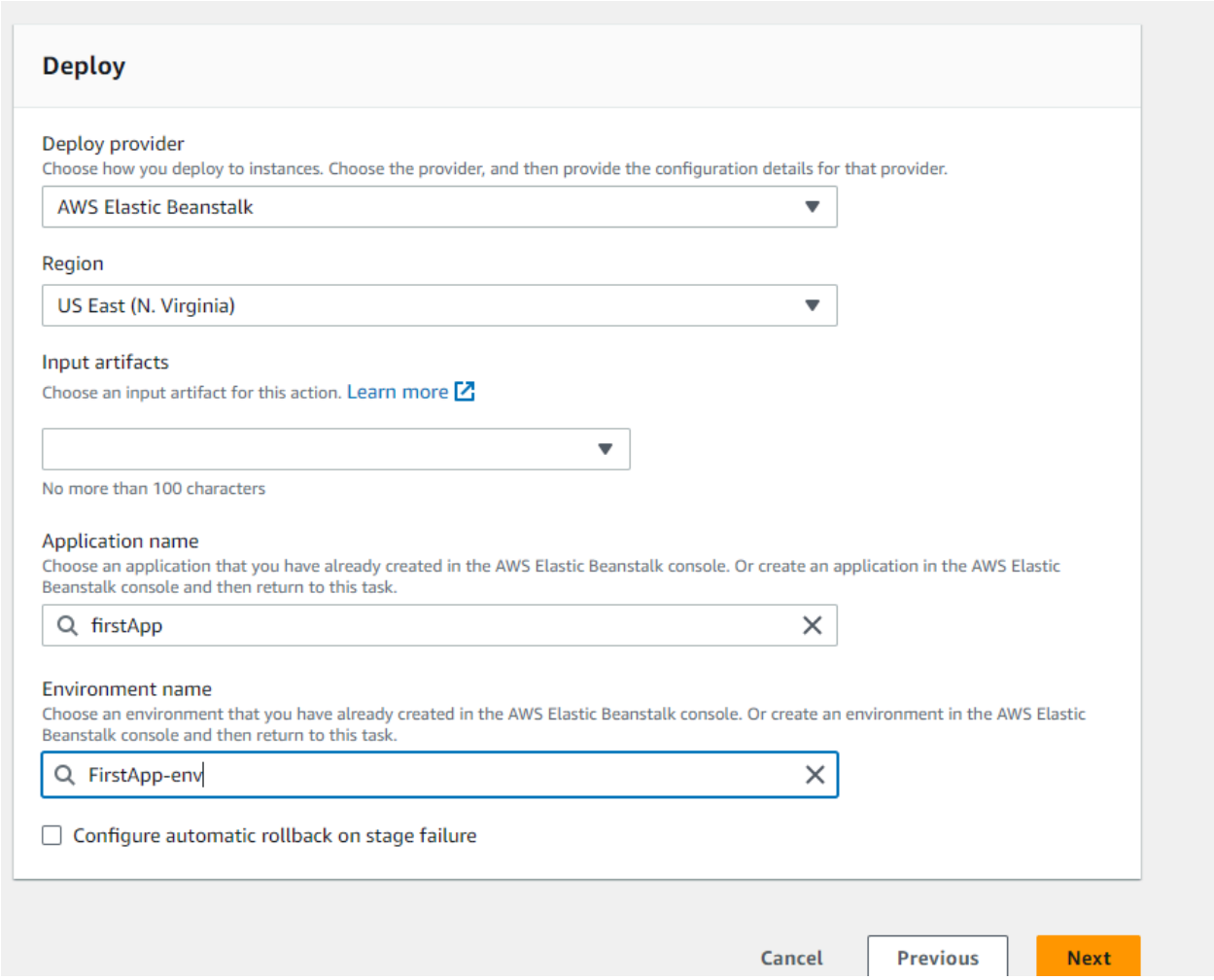
3. Then, simply choose this forked repository and the branch which you will be able to find in the search box. After that, click Continue and skip the build stage. Proceed to the Deployment stage.



Name: Akanksha Shinde Class: D15C Roll No: 53

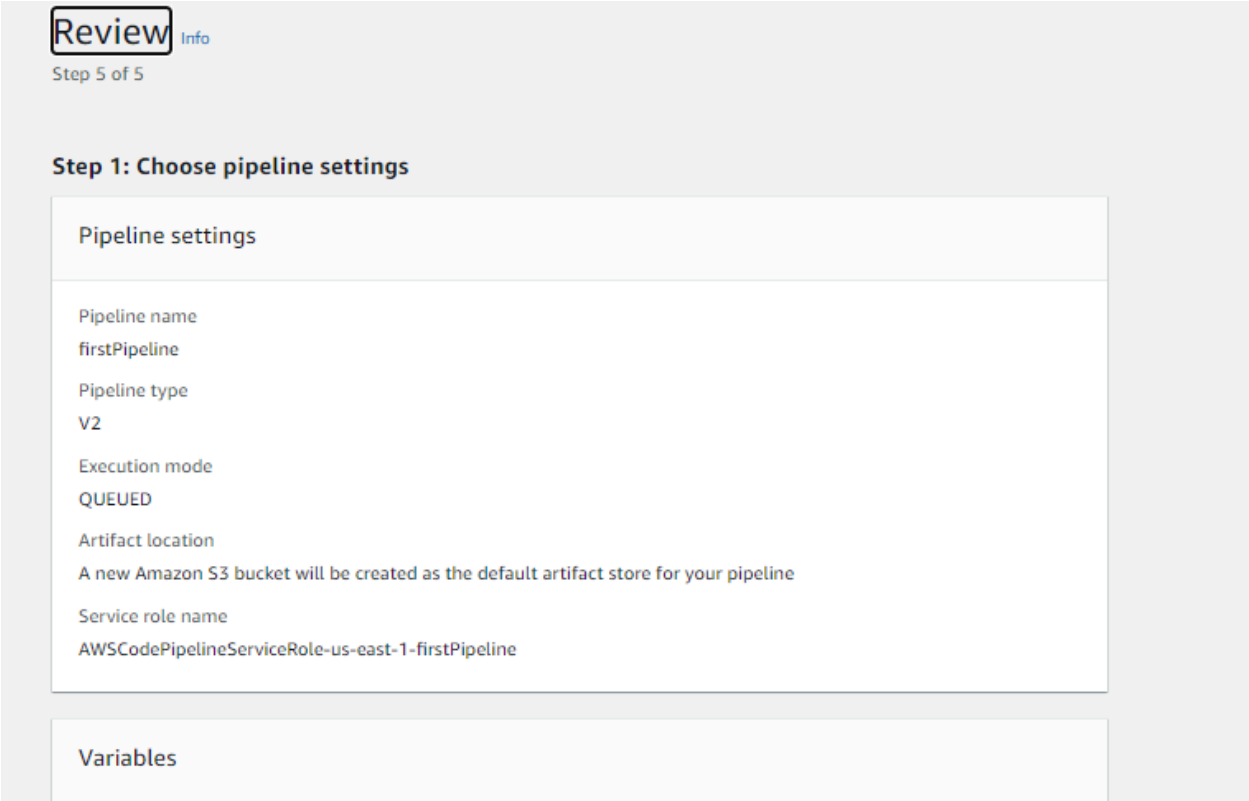
Step 4: Deployment

1. Choose Beanstalk as the Deploy Provider, same region as the Bucket and Beanstalk, name and environment name. Click Next, Review and create the pipeline.

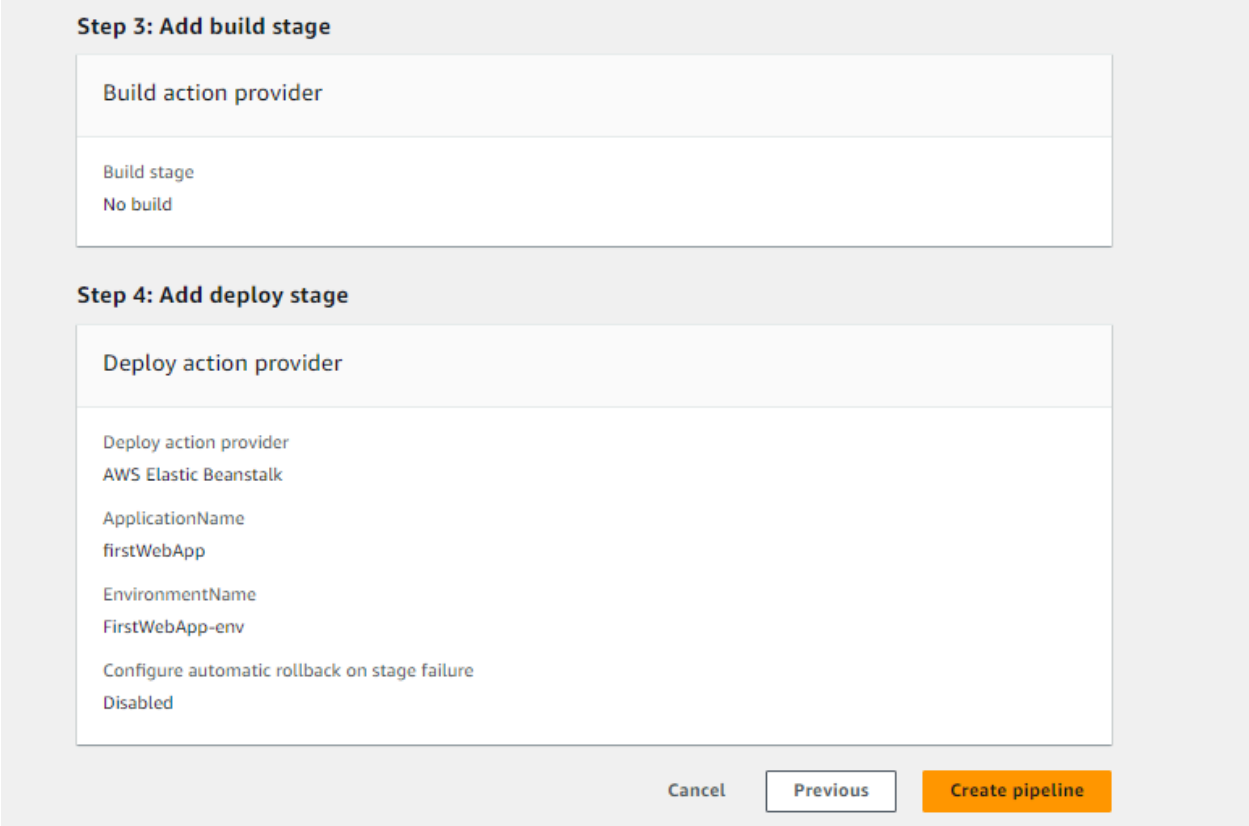


Name: Akanksha Shinde Class: D15C Roll No: 53

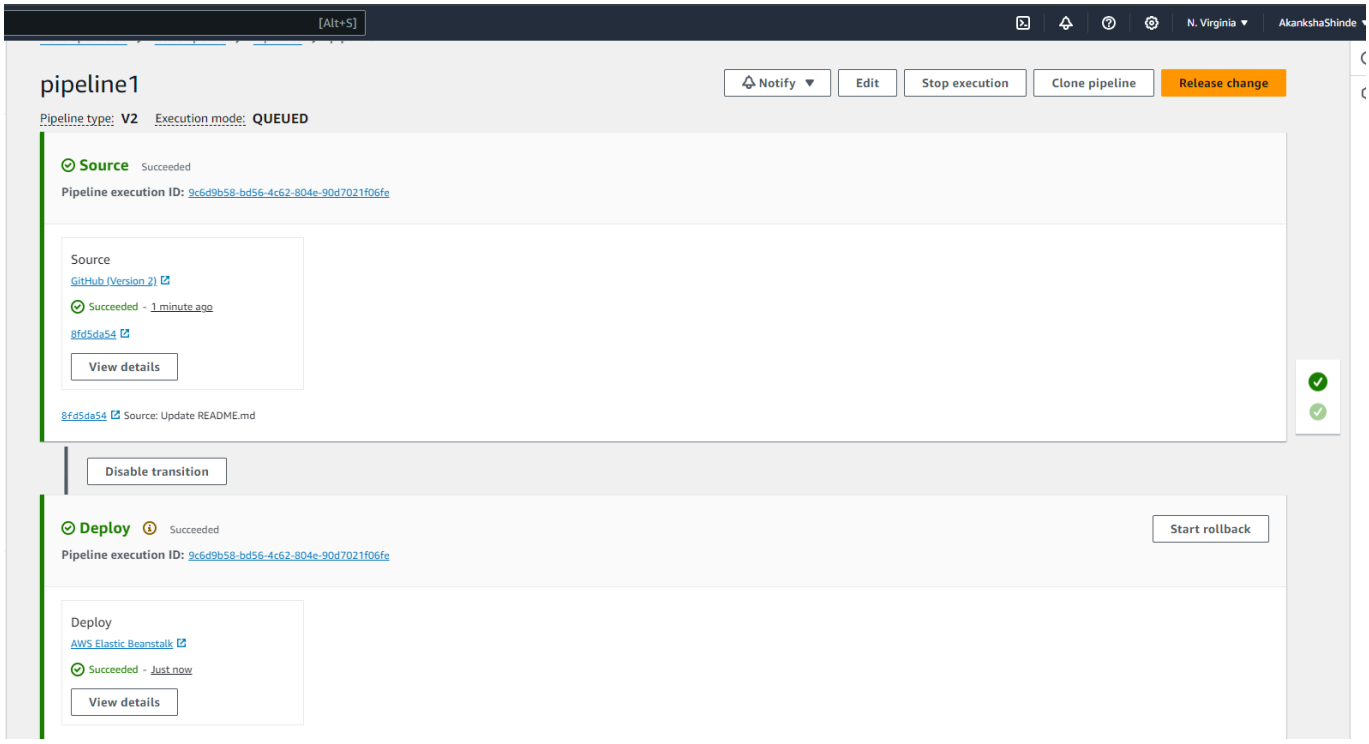
2. Review all the settings and click on create pipeline



Name: Akanksha Shinde Class: D15C Roll No: 53

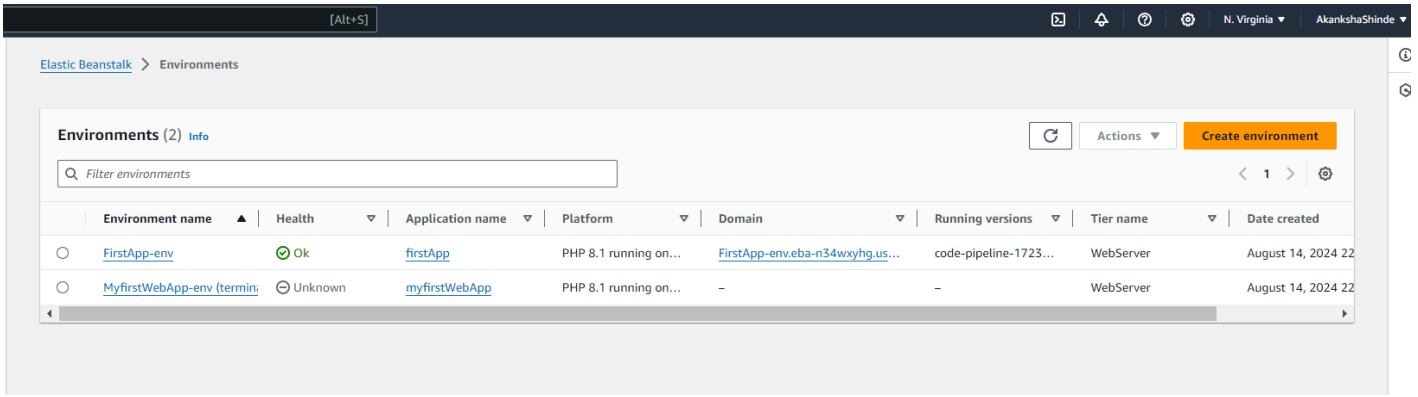


Finally you will be able to see this screen where you can infer the code has been deployed successfully.

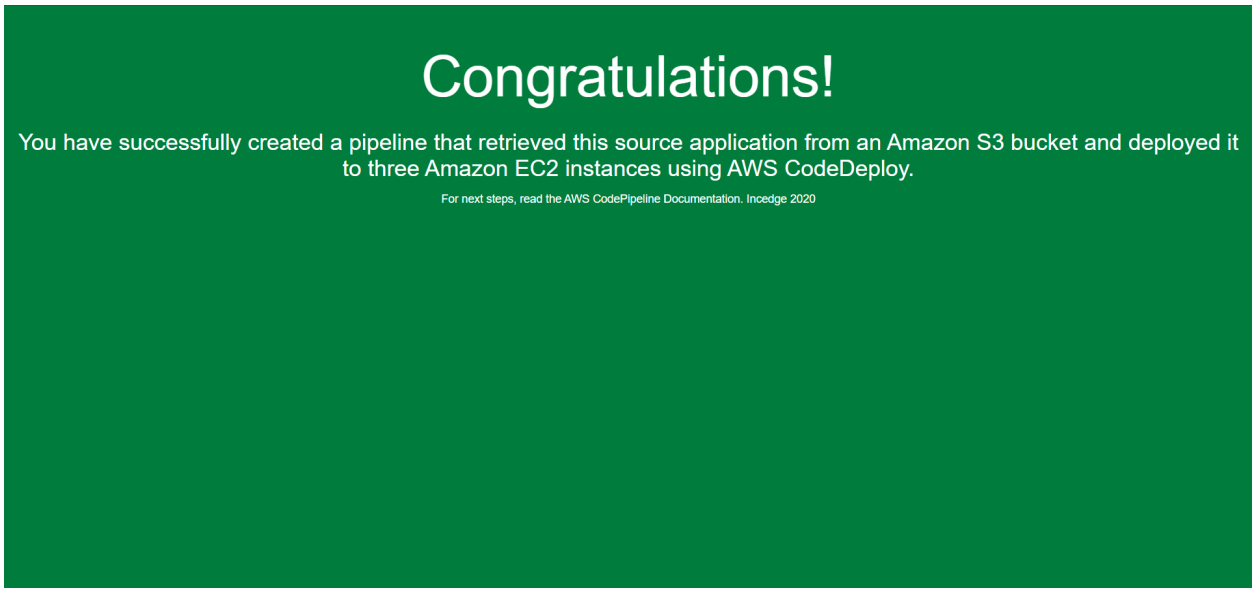


Name: Akanksha Shinde Class: D15C Roll No: 53

In a few minutes, we will have our pipeline created. Once we have the success message on the Deploy part, we can go ahead and check our URL provided in the EBS environment.



This is the sample website we just created.



If you can see this, that means that you successfully created an automated software using CodePipeline.