

Hands-On Lab

Deploy & Manage an Amazon-like Shopping Website using DevOps Tools (Terraform, Jenkins (CI-CD), SonarQube, Docker & Trivy on AWS Cloud)

Scenario:

Imagine a startup called ShopifyLite is developing an Amazon clone app. They use Terraform for infrastructure management and Jenkins for CI/CD. Developers write code, Jenkins tests it, and if successful, Terraform deploys it to AWS. This automated process ensures fast and reliable updates, enabling ShopifyLite to offer a seamless shopping experience to users while efficiently managing their infrastructure.

Description:

This project involves creating an Amazon clone app named ShopifyLite. Using Terraform for infrastructure management and Jenkins for CI/CD, the team automates testing and deployment processes. Developers push code changes, Jenkins tests them, and Terraform deploys updates to the Shopping Website hosted on AWS Cloud. This ensures a fast and reliable app launch, allowing users to browse and shop seamlessly.

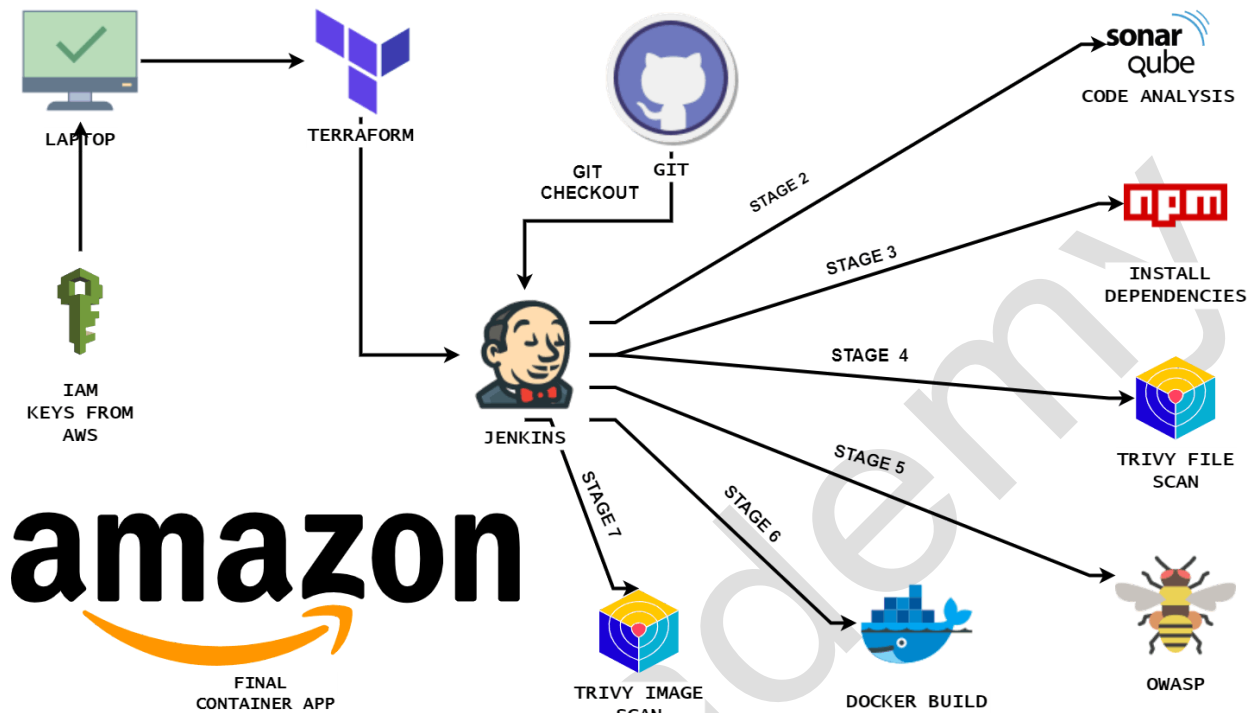
Contents

1 Introduction & Key Concepts.....	4
1.1 What is Terraform?.....	4
1.2 What is Docker?.....	5
1.3 What is OWASP?.....	5
1.4 What is SonarQube?.....	5
2 Documentation.....	6
3 Pre-Requisite.....	7
4 Cost to Perform this Lab.....	8
5 Launch & Configure EC2 Instance.....	9
6 Set up Terraform & AWS CLI.....	15
6.1 Setup Terraform.....	15
6.2 Set up AWS CLI.....	16
7 Create IAM User.....	19
8 Creating Workspace & Clone Terraform Repo from GitHub.....	24
9 Building a Simple Infrastructure using Terraform.....	25
10 Initialize Terraform.....	26
11 Validating the plan.....	27
12 Execute the plan.....	29
13 Setup Sonarqube and Jenkins.....	32
13.1 Setup Sonarqube.....	32
13.2 Setup Jenkins.....	33
14 Set up CI-CD Pipeline.....	38
14.1 Install Plugins.....	38
14.2 Add Credentials for SonarQube and Docker.....	40
15 Set up the Tools for Jenkins.....	46
15.1 Add JDK & NodeJS.....	46
15.2 Add Docker & SonarQube.....	47
15.3 Add OWASP dependency check.....	48
16 Configure Global Settings for Sonarube.....	49
16.1 Run the Pipeline.....	49
17 Share your learnings on LinkedIn & Community.....	54
17.1 On LinkedIn.....	55
17.2 Share wins in the Progress Diary.....	57
18 Cleanup Resources.....	58
18.1 Destroying the Infrastructure.....	58
19 Troubleshooting.....	59
19.1 Build In-Progress/ Build Failure.....	60
19.1.1 Issue:.....	60
19.1.2 Reason:.....	60
19.1.3 Resolution:.....	60
19.2 Could not find 'java' executable in JAVA_HOME or PATH.....	61
19.2.1 Issue.....	61

19.2.2 Reason:.....	62
19.2.3 Fix:.....	62
20 Summary.....	64

K21Academy

1 INTRODUCTION & KEY CONCEPTS



1.1 What is Terraform?

Terraform is an open-source Infrastructure as Code (IaC) tool developed by HashiCorp. It allows users to define and provision infrastructure in a declarative configuration language. With Terraform, you can manage and automate the deployment of infrastructure resources across various cloud providers, on-premises environments, and even third-party services.

Key features of Terraform include:

Declarative Configuration: Infrastructure is defined in human-readable configuration files, specifying the desired state of the infrastructure.

Multi-Cloud Support: Terraform is cloud-agnostic and supports multiple cloud providers, including Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), and others.

Resource Graph: Terraform builds a dependency graph of resources, enabling efficient provisioning and parallel execution of infrastructure changes.

State Management: Terraform keeps track of the current state of infrastructure, allowing it to plan and apply only the necessary changes to achieve the desired state.

1.2 What is Docker?

Docker is a platform and set of tools designed to make it easier to create, deploy, and run applications by using containers. Containers allow developers to package up an application with all the parts it needs, such as libraries and dependencies, and ship it as one package. This ensures that the application will run on any environment.

1.3 What is OWASP?

OWASP stands for the Open Web Application Security Project. It is a non-profit organization focused on improving the security of software. OWASP provides resources, tools, and best practices to help organizations develop, deploy, and maintain secure web applications. They also publish a list of the top web application security risks, known as the OWASP Top 10, which serves as a guide for developers and security professionals to prioritize their efforts in addressing common vulnerabilities.

1.4 What is SonarQube?

SonarQube is an open-source platform used for continuous inspection of code quality to perform automatic reviews with static analysis of code to detect bugs, code smells, and security vulnerabilities on over 20 programming languages. It integrates with various CI/CD tools and provides detailed reports on code quality metrics, helping development teams to improve code maintainability, reliability, and security throughout the software development lifecycle.

2 DOCUMENTATION

1. **Terraform:**

<https://www.terraform.io/docs/>

2. **Terraform Provider:**

<https://registry.terraform.io/providers/hashicorp/google/latest>

3. **Docker:**

<https://www.docker.com/>

4. **AWS S3:**

<https://aws.amazon.com/s3/>

5. **AWS CLI:**

<https://aws.amazon.com/cli/>

6. **Jenkins:**

<https://www.jenkins.io/doc/book/installing/linux/>

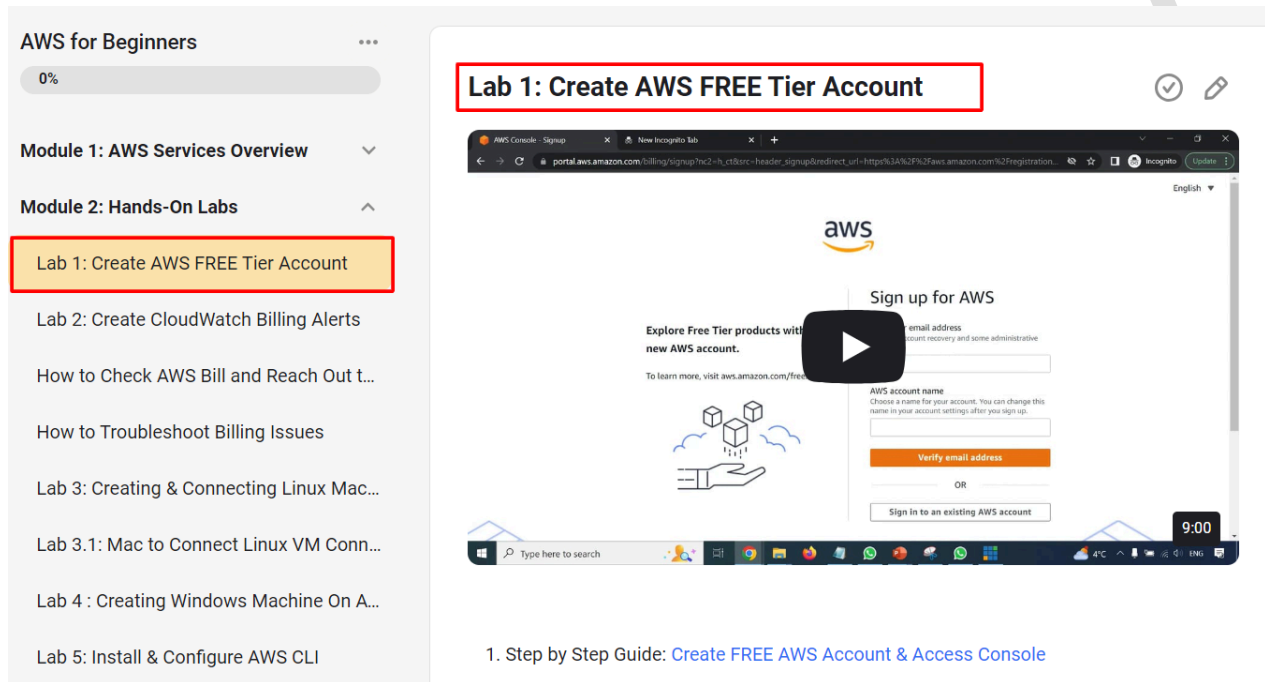
7. **SonarQube:**

<https://www.sonarsource.com/products/sonarqube/downloads/>

3 PRE-REQUISITE

To perform this lab, you'll need an AWS Cloud account. Please refer to the following Activity Guide to create one.

[Register for an AWS Free Tier Account, Amazon Web Services & log in to the AWS Console](#)



The screenshot displays the 'AWS for Beginners' lab interface. On the left, a sidebar lists various labs, with 'Lab 1: Create AWS FREE Tier Account' selected and highlighted. The main area features a video player for the same lab, showing the AWS sign-up process. Below the video, a link provides a step-by-step guide for creating a free AWS account and accessing the console.

Ensure we have a DockerHub account..

Note: If you don't have a Docker Hub account, please check this official link.

<https://hub.docker.com/>

4 COST TO PERFORM THIS LAB

Cost Estimate: The estimated cost for this lab is **\$0.0928 per hour.**

Since we're using a **t2.large** instance, it will incur charges whether you're on the **Free Tier** or a **Pay-As-You-Go** account.

- **EC2 Instance (t2.large):** ~\$0.0928 per hour per instance.

For more detailed pricing information, refer to the link below:

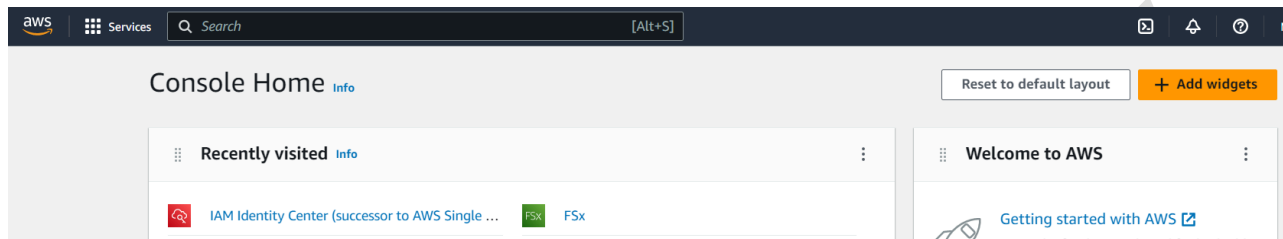
👉 [EC2 Pricing](#)

K21Academy

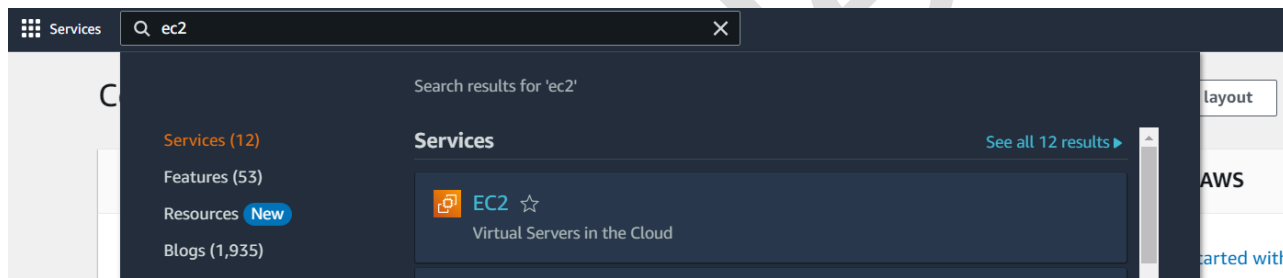
5 LAUNCH & CONFIGURE EC2 INSTANCE

In this section, we are creating an EC2 instance and configuring it so we can install Terraform and Docker in our next step.

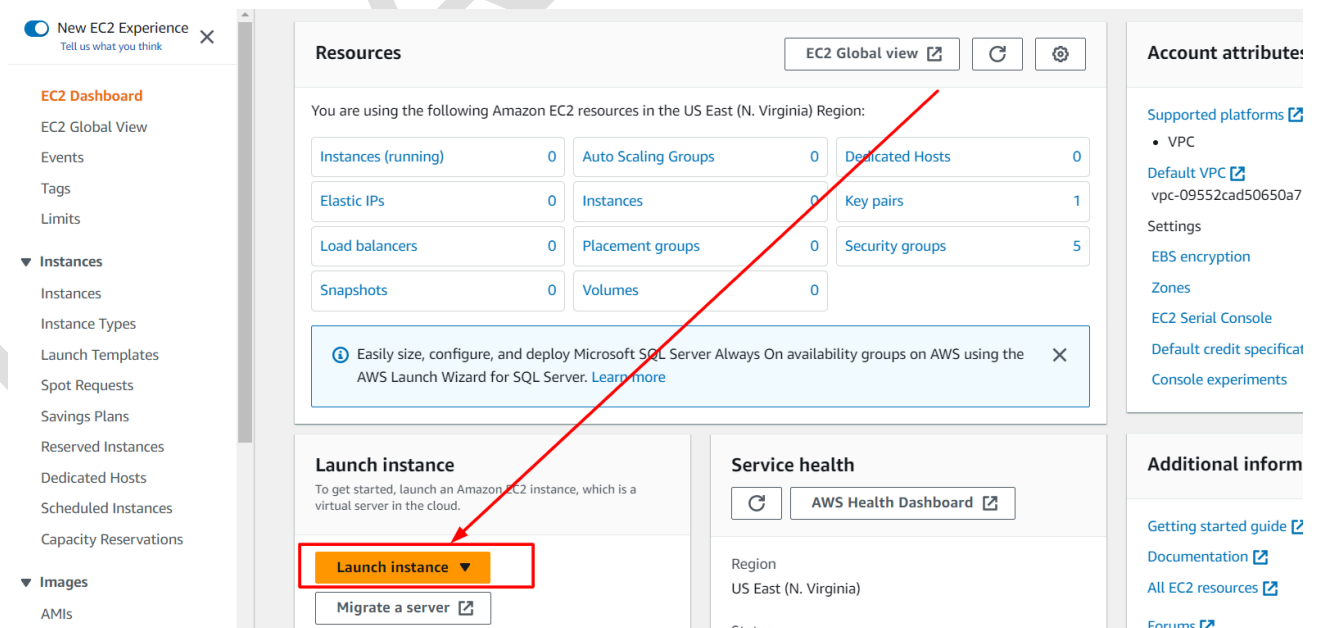
1. Open your **console**.



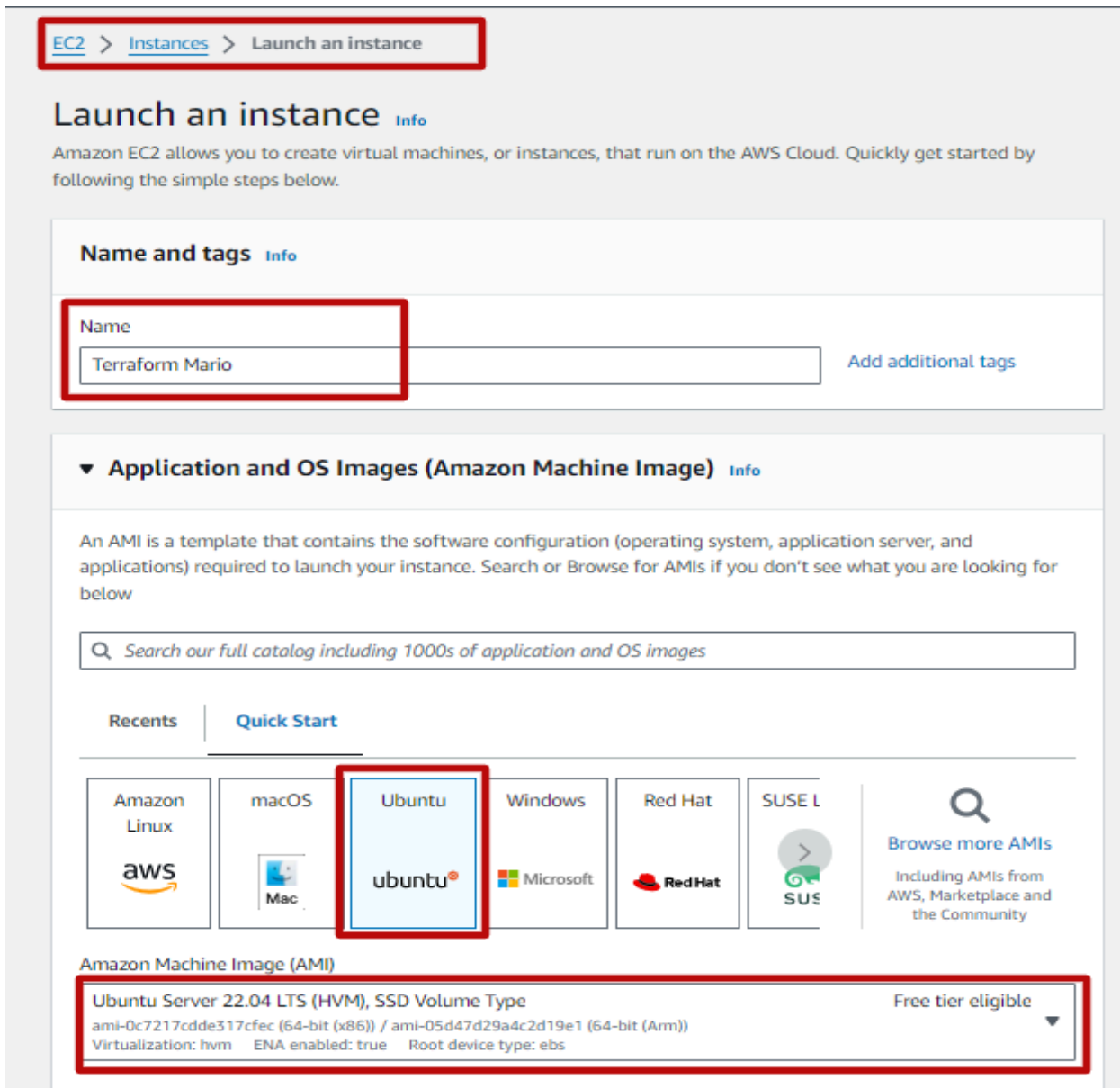
2. Search for **EC2** in the search bar and click on it.



3. Click on **Launch Instances**.



- Choose name as **Terraform** & choose **Ubuntu Server**, select 22.04 LTS architecture and Click on **Select** & Choose an Instance type **t2.micro** as it is in free tier.



EC2 > Instances > Launch an instance

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name

Terraform Mario [Add additional tags](#)

▼ Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents Quick Start

Amazon Linux

macOS

Ubuntu

Windows

Red Hat

SUSE L

[Browse more AMIs](#)
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type Free tier eligible

ami-0c7217cdde317cfec (64-bit (x86)) / ami-05d47d29a4c2d19e1 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs


- Select an existing key pair or create a new one, we will create a new one, enter the name of the Key-pair as mario Keypair, select **.ppk** and Create the Key Pair.
- Now in networking, Click on **Create a new Security Group** and allow all the https,http.

▼ Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required

mariokey ▼

 Create new key pair

▼ Network settings Info Edit

Network Info

vpc-046021852ccb29093 | Default_VPC

Subnet Info

No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group

Select existing security group

We'll create a new security group called 'launch-wizard-18' with the following rules:

☒ Allow SSH traffic from

Helps you connect to your instance


Anywhere
0.0.0.0/0 ▼

☒ Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

☒ Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server


 Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. X

▼ Configure storage Info Advanced


1x 8 GiB gp2 ▼ Root volume (Not encrypted)

Add new volume

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance

 Click refresh to view backup information

The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.



0 x File systems Edit

Copyright© K21Academy | All Rights Reserved

Sharing, Reselling, or duplication of this content is strictly prohibited without K21Academy's written permission.

7. Click on **Launch Instance** & Click on **View All Instances**.

EC2 > Instances > Launch an instance

Success
Successfully initiated launch of instance (i-027ba060342207db8)

► Launch log

Next Steps

Get notified of estimated charges
Create [billing alerts](#) to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier)

How to connect to your instance
Your instance is launching and it might be a few minutes until it is in the running state, when it will be ready for you to use
Click [View Instances](#) to monitor your instance's status. Once your instance is in the 'running' state, you can connect to it from the Instances screen. Find out [how to connect to your instance](#)

[View more resources to get you started](#)

View all instances

8. Refresh and you shall see your instances are up and running, and the **Status check** has changed to **2/2 checks**.

Instances (2) Info

Find Instance by attribute or tag (case-sensitive) All states

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input type="checkbox"/>	Terraform Proj...	i-00e7fabbc0e5d7cfb	Stopped	t2.micro	-	View alarms +	us-east-1d	-
<input type="checkbox"/>	EKS-instance	i-0a20bccb811f4143d	Stopped	t2.micro	-	View alarms +	us-east-1d	-

9. Click on **connect** and you are connected to your machine.

Connect to instance [Info](#)

Connect to your instance i-039e6686d3af7b69c (Terraform Mario) using any of these options

EC2 Instance Connect

Session Manager

SSH client

EC2 serial console

Instance ID

 i-039e6686d3af7b69c (Terraform Mario)

Connection Type

☒ Connect using EC2 Instance Connect

Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address.

☐ Connect using EC2 Instance Connect Endpoint



Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.


Public IP address

 54.160.203.116

Username

Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ubuntu.

 ubuntu 

 **Note:** In most cases, the default username, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Cancel

Connect

```
aws Services Search [Alt+S]
System load: 0.080078125 Processes: 96
Usage of /: 20.6% of 7.57GB Users logged in: 0
Memory usage: 21% IPv4 address for eth0: 172.31.29.0
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-172-31-29-0:~$

i-039e6686d3af7b69c (Terraform Mario)
PublicIPs: 54.160.203.116 PrivateIPs: 172.31.29.0
```

Now you have successfully connected to your **Ubuntu** instance

Copyright© K21Academy | All Rights Reserved

Sharing, Reselling, or duplication of this content is strictly prohibited without K21Academy's written permission.

10. Run the following commands for root user permission.

```
sudo su
```

&

```
apt update -y
```

```
ubuntu@ip-172-31-29-0:~$ sudo su
root@ip-172-31-29-0:/home/ubuntu# apt update -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1377 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [273 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [1431 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [235 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1049 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [237 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [22.1 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [42.1 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [10.1 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [472 B]
```

6 SET UP TERRAFORM & AWS CLI

6.1 Setup Terraform

Setting up Terraform involves installing the Terraform software on your system. This enables you to create, modify, and manage infrastructure as code using Terraform configuration files.

1. Install the wget package using the command:

```
sudo apt install wget -y
```

```
root@ip-172-31-29-0:/home/ubuntu# sudo apt install wget -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
wget is already the newest version (1.21.2-2ubuntu1).
wget set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 74 not upgraded.
root@ip-172-31-29-0:/home/ubuntu#
```

2. Download and add HashiCorp GPG key to the keyring:

```
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg
--dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
```

```
root@ip-172-31-29-0:/home/ubuntu# wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
--2024-02-20 06:45:47-- https://apt.releases.hashicorp.com/gpg
Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.com)... 99.84.108.3, 99.84.108.36, 99.84.108.40, ...
Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)|99.84.108.3|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3980 (3.9K) [binary/octet-stream]
Saving to: 'STDOUT'

0%[
100%[=====] 3.89K --.-KB/s in 0s

2024-02-20 06:45:47 (1.20 GB/s) - written to stdout [3980/3980]
root@ip-172-31-29-0:/home/ubuntu#
```

```
echo "deb
[signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
https://apt.releases.hashicorp.com $(lsb_release -cs) main" |
sudo tee /etc/apt/sources.list.d/hashicorp.list
```

```
root@ip-172-31-29-0:/home/ubuntu# echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt
/sources.list.d/hashicorp.list
deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com jammy main
root@ip-172-31-29-0:/home/ubuntu#
```

3. Update the package index and install Terraform:

```
sudo apt update && sudo apt install terraform -y
```

```
root@ip-172-31-29-0:/home/ubuntu# sudo apt update && sudo apt install terraform -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 https://apt.releases.hashicorp.com jammy InRelease [12.9 kB]
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:6 https://apt.releases.hashicorp.com jammy/main amd64 Packages [119 kB]
Fetched 132 kB in 1s (146 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
74 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  terraform
0 upgraded, 1 newly installed, 0 to remove and 74 not upgraded.
Need to get 26.9 MB of archives.
After this operation, 84.5 MB of additional disk space will be used.
Get:1 https://apt.releases.hashicorp.com jammy/main amd64 terraform amd64 1.7.3-1 [26.9 MB]
Fetched 26.9 MB in 0s (81.2 MB/s)
Selecting previously unselected package terraform.
(Reading database ... 65163 files and directories currently installed.)
Preparing to unpack .../terraform_1.7.3-1_amd64.deb ...
Unpacking terraform (1.7.3-1) ...
Setting up terraform (1.7.3-1) ...
```

```
root@ip-172-31-29-0:/home/ubuntu# which terraform
/usr/bin/terraform
root@ip-172-31-29-0:/home/ubuntu#
```

Now you have successfully installed Terraform.

6.2 Set up AWS CLI

Aws using your computer's command lines instead of clicking around on a website. It helps you do things like telling AWS to create or manage stuff, all by typing commands in a special language your computer understands. It's like giving orders to AWS with your keyboard instead of a mouse.

1. Install the unzip package using the command:

```
sudo apt -y install unzip curl
```



```
root@ip-172-31-29-0:/home/ubuntu# sudo apt -y install unzip curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
curl is already the newest version (7.81.0-1ubuntu1.15).
curl set to manually installed.
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 74 not upgraded.
Need to get 175 kB of archives.
After this operation, 386 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 unzip amd64 6.0-26ubuntu3.2 [175 kB]
Fetched 175 kB in 0s (8315 kB/s)
```

2. Download the AWS CLI installer package using curl:

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip"
-o "awscliv2.zip"
```

```
root@ip-172-31-29-0:/home/ubuntu# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left  Speed
100 57.3M  100 57.3M    0     0  109M      0  --:--:-- --:--:-- --:--:--  109M
root@ip-172-31-29-0:/home/ubuntu#
```

3. Unzip the AWS CLI installer package:

```
unzip awscliv2.zip
```

```
root@ip-172-31-29-0:/home/ubuntu# unzip awscliv2.zip
```

4. Run the installer script with sudo privileges to install AWS CLI:

```
sudo ./aws/install
```

```
root@ip-172-31-29-0:/home/ubuntu# sudo ./aws/install
You can now run: /usr/local/bin/aws --version
root@ip-172-31-29-0:/home/ubuntu#
```

5. Run the below command to check the AWS CLI Version.

```
aws --version
```

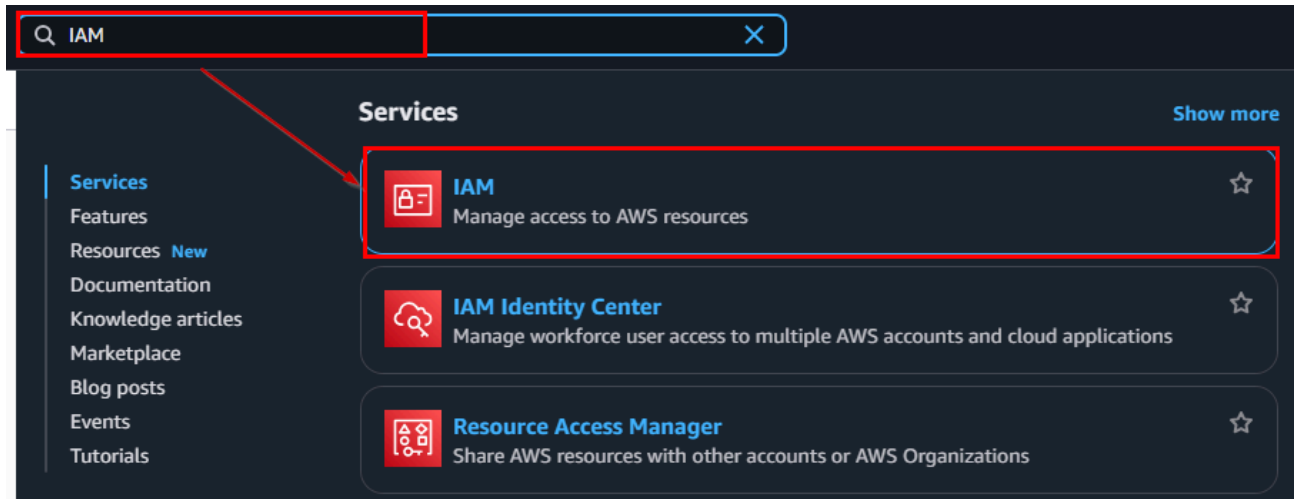
```
root@ip-172-31-29-0:/home/ubuntu# aws --version  
aws-cli/2.15.21 Python/3.11.6 Linux/6.2.0-1017-aws exe/x86_64.ubuntu.22 prompt/off  
root@ip-172-31-29-0:/home/ubuntu#
```

Now we have successfully installed AWS CLI.

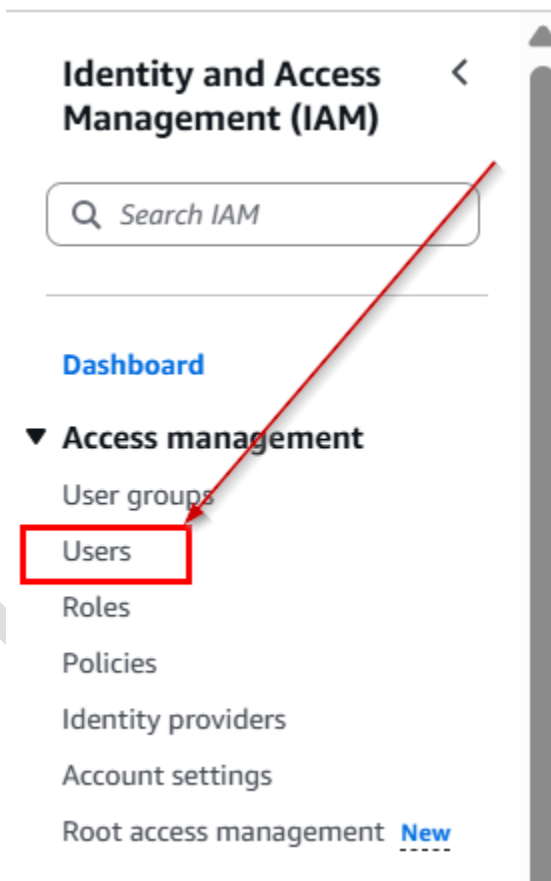
K21Academy

7 CREATE IAM USER

1. Go to the Dashboard & in the search bar type **IAM**



2. Click on **User** on the left side.



- Give a name to your user, **Amazon** and **tick** on Provide user access to management console and then click on **I want an IAM user option**

Specify user details

User details

User name
Amazon

☒ Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.

Are you providing console access to a person?

User type

☐ Specify a user in Identity Center - Recommended
We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.

☒ I want to create an IAM user
We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keyspaces, or a backup credential for emergency account access.

Console password

☐ Autogenerated password
You can view the password after you create the user.

☒ Custom password
Enter a custom password for the user.

Must be at least 10 characters long
Must include at least one uppercase letter (A-Z)
Must include at least one lowercase letter (a-z)
Must include at least one number (0-9)
Must include at least one non-alphanumeric character (@ # \$ % ^ & * () _ + = { } | ' " , . ~)

☐ Show password

☒ Users must create a new password at next sign-in - Recommended

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. [Learn more](#)

Cancel **Next**

- Choose a password for your user → **click next**.
- Attach the policies directly to your IAM user → **click next**

Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

Permissions options

☐ Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

☐ Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.

☒ Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (1/1220)

Choose one or more policies to attach to your new user.

Search Filter by Type

	Policy name	Type	Attached entities
<input type="checkbox"/>	AccessAnalyzerServiceRolePolicy	AWS managed	0
<input checked="" type="checkbox"/>	AdministratorAccess	AWS managed - job function	5
<input type="checkbox"/>	AdministratorAccess-Amplify	AWS managed	0
<input type="checkbox"/>	AdministratorAccess-AWSElasticBeanstalk	AWS managed	0

Note → I will provide administrator access for now but we are careful while attaching the policies on your workplace.

6. Click on **create user**.

Review and create

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details

User name Amazon	Console password type Custom password	Require password reset Yes
---------------------	--	-------------------------------

Permissions summary

Name	Type	Used as
AdministratorAccess	AWS managed - job function	Permissions policy
IAMUserChangePassword	AWS managed	Permissions policy

Tags - optional

Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tags.

[Cancel](#) [Previous](#) [Create user](#)

7. Download your password file if it is auto-generated otherwise, it is your choice

User created successfully

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

[View user](#)

Retrieve password

You can view and download the user's password below or email users instructions for signing in to the AWS Management Console. This is the only time you can view and download this password.

Console sign-in details

Console sign-in URL

[https://442042521384.signin.aws.amazon.com/console](#)

User name

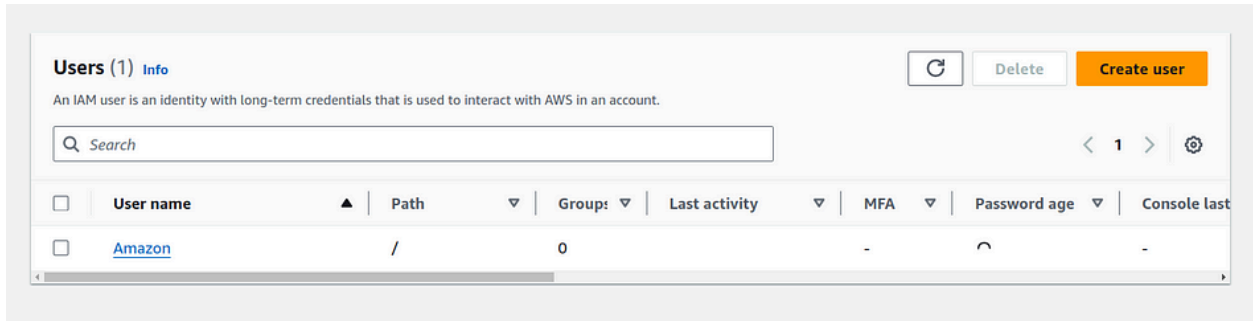
[Amazon](#)

Console password

***** [Show](#)

[Email sign-in instructions](#)

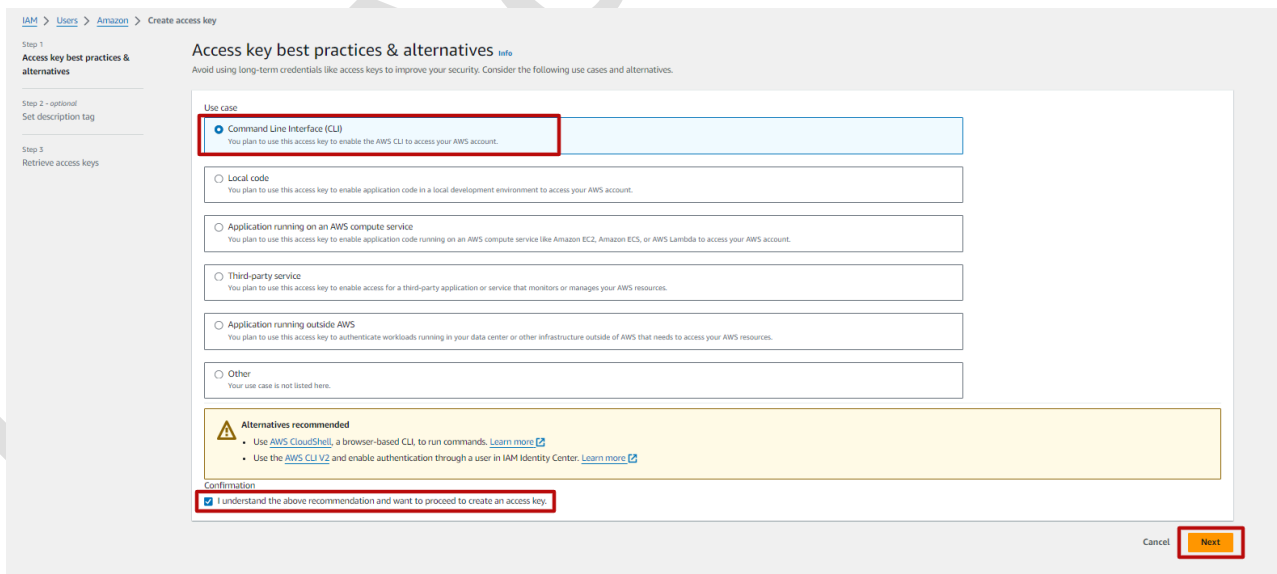
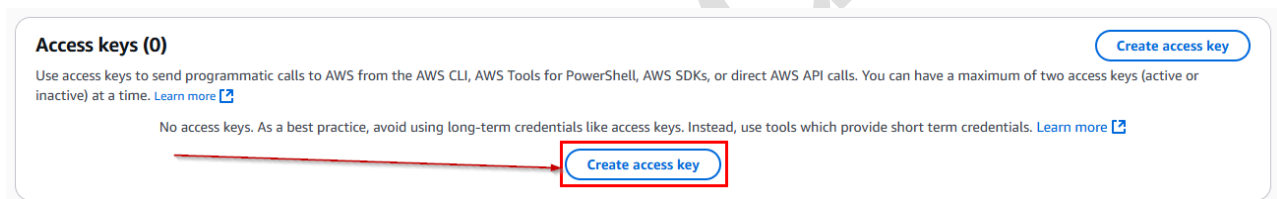
[Cancel](#) [Download .csv file](#) [Return to users list](#)

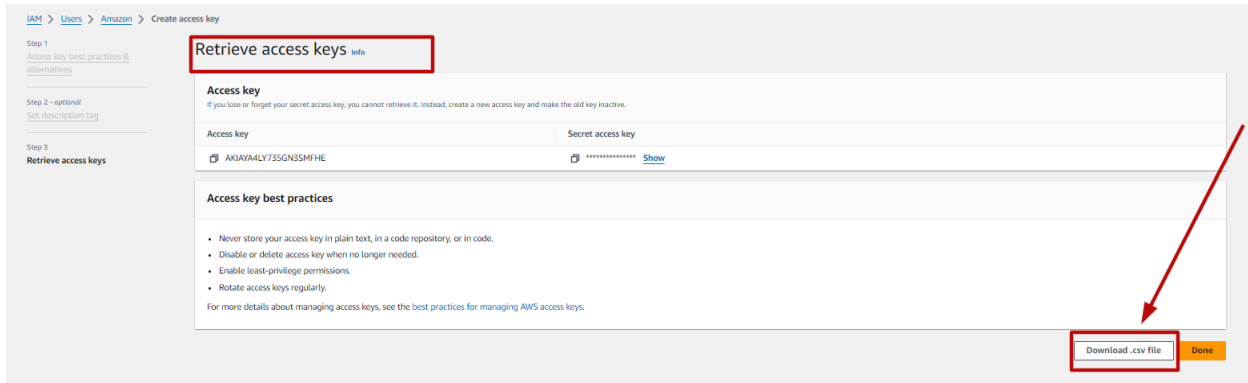


8. Now click on your IAM user → security credentials



9. Scroll down to **access keys** and create access keys.





Now you have successfully created your IAM user.

10. Go to your terminal and type → **aws configure**

11. Now it is asked for your **access key** and **secret key**. For this open your CSV file and paste the access and secret key and leave everything default.

```
root@ip-172-31-85-222:~# aws configure
AWS Access Key ID [None]: AKIAYA4LY735PCEX2C6B
AWS Secret Access Key [None]: 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
Default region name [None]:
Default output format [None]:
root@ip-172-31-85-222:~#
```

Now you are ready to configure AWS from your terminal.

8 CREATING WORKSPACE & CLONE TERRAFORM REPO FROM GITHUB

1. Create a directory named **"Amazon"** using the command:

```
mkdir Amazon
```

2. Navigate into the **"Amazon"** directory:

```
cd Amazon
```

```
C:\Users\Deep Sharma\Desktop>mkdir amazon
```

```
C:\Users\Deep Sharma\Desktop>cd amazon
```

3. Clone the GitHub repository using the following command:

```
git clone  
https://github.com/k21academyuk/Amazon-app-Deployment-using-terra  
form-and-jenkins
```

```
C:\Users\Deep Sharma\Desktop\amazon>git clone https://github.com/Aakibgithuber/Amazon-app-Deployment-using-terraform-and-jenkins.git  
Cloning into 'Amazon-app-Deployment-using-terraform-and-jenkins'...  
remote: Enumerating objects: 165, done.  
remote: Counting objects: 100% (165/165), done.  
remote: Compressing objects: 100% (153/153), done.  
remote: Total 165 (delta 36), reused 116 (delta 11), pack-reused 0 receiving objects: 67% (111/165), 1.29 MiB | 1.16 MiB/s  
Receiving objects: 100% (165/165), 3.17 MiB | 2.13 MiB/s, done.  
Resolving deltas: 100% (36/36), done.
```

4. Move into the cloned repository directory:

```
cd Amazon-app-Deployment-using-terraform-and-jenkins
```


9 BUILDING A SIMPLE INFRASTRUCTURE USING TERRAFORM

1. Go to folder → **cd JENKINS-TF**

```
root@ip-172-31-85-222:/home/ubuntu/amazon/Amazon-app-Deployment-using-terraform-and-jenkins# cd JENKINS-TF/
root@ip-172-31-85-222:/home/ubuntu/amazon/Amazon-app-Deployment-using-terraform-and-jenkins/JENKINS-TF# ls
Main.tf  install_jenkins.sh  provider.tf
root@ip-172-31-85-222:/home/ubuntu/amazon/Amazon-app-Deployment-using-terraform-and-jenkins/JENKINS-TF#
```

2. There are three files present **main.tf**, **install_jenkins.sh** , **provider.tf**
3. Open the file → **vim Main.tf**

```
root@ip-172-31-85-222:/home/ubuntu/amazon/Amazon-app-Deployment-using-terraform-and-jenkins/JENKINS-TF# vim Main.tf

resource "aws_instance" "amazon" {
  ami           = "ami-0c7217cdde317cfec"
  instance_type = "t2.large"
  key_name      = "my key"
  vpc_security_group_ids = [aws_security_group.Jenkins-sg.id]
  user_data      = templatefile("./install_jenkins.sh", {})

  tags = {
    Name = "Amazon clone"
  }
  root_block_device {
    volume_size = 30
  }
}
```

**Note: Change this section → ami = ami-0e001c9271cf7f3b9 # your ami id ,
key_name= #your key pair if any.**

Click on “I” in your keyboard to enter into insert mode, then use the navigation keys to delete and update the ami and Key name. Then Click “esc” Key and then type “:wq” to save the file. And then Press Enter.

main.tf includes userdata, which links **install_jenkins.sh** file on which execution installs Jenkins, Docker, trivy, and starts the SonarQube container on port 9000.

10 INITIALIZE TERRAFORM

Initializing Terraform is the process of preparing a Terraform project or configuration for use. When you initialize Terraform in a directory where you have your Terraform configuration files,

Terraform must initialize any configured backend before use.

1. Run **terraform init** to initialize the directory and download the provider plugins.

```
terraform init
```

```
root@ip-172-31-85-222:/home/ubuntu/amazon/Amazon-app-Deployment-using-terraform-and-jenkins/JENKINS-TF# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching ">= 5.0"...
- Installing hashicorp/aws v5.42.0...
- Installed hashicorp/aws v5.42.0 (signed by HashiCorp)

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

Terraform has been successfully initialized!

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

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

Note: Make sure you run terraform init command in the directory where your configuration files exist.

This section completes initializing Terraform, which is necessary to download all the backend configurations to work with the AWS provider.

11 VALIDATING THE PLAN

Once the terraform directory is initialized, we will create an execution plan and will validate if it is correct or there are any errors in this.

1. Validate the code.

```
terraform validate
```

```
root@ip-172-31-85-222:/home/ubuntu/amazon/Amazon-app-Deployment-using-terraform-and-jenkins/JENKINS-TF# terraform validate
Success! The configuration is valid.

root@ip-172-31-85-222:/home/ubuntu/amazon/Amazon-app-Deployment-using-terraform-and-jenkins/JENKINS-TF#
```

2. Run **terraform plan** to validate the code and check if it's valid or not. It will show all the changes that are going to take place when we will execute this plan

```
terraform plan
```

```
root@ip-172-31-85-222:/home/ubuntu/amazon/Amazon-app-Deployment-using-terraform-and-jenkins/JENKINS-TF# terraform plan
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_instance.web will be created
+ resource "aws_instance" "web" {
  + ami               = "ami-080e1f13689e07408"
  + arn               = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone  = (known after apply)
  + cpu_core_count     = (known after apply)
  + cpu_threads_per_core = (known after apply)
  + disable_api_stop    = (known after apply)
  + disable_api_termination = (known after apply)
  + ebs_optimized      = (known after apply)
  + get_password_data   = false
  + host_id            = (known after apply)
  + host_resource_group_arn = (known after apply)
  + iam_instance_profile = (known after apply)
  + id                 = (known after apply)
  + instance_initiated_shutdown_behavior = (known after apply)
  + instance_lifecycle = (known after apply)
  + instance_state     = (known after apply)
  + instance_type       = "t2.large"
  + ipv6_address_count  = (known after apply)
  + ipv6_addresses     = (known after apply)
```

```
    + description      = "TLS from VPC"
    + from_port        = 9100
    + ipv6_cidr_blocks = []
    + prefix_list_ids  = []
    + protocol         = "tcp"
    + security_groups  = []
    + self             = false
    + to_port          = 9100
  },
]
+ name                = "Jenkins-Security Group"
+ name_prefix         = (known after apply)
+ owner_id            = (known after apply)
+ revoke_rules_on_delete = false
+ tags                = {
  + "Name" = "Jenkins-sg"
}
+ tags_all            = {
  + "Name" = "Jenkins-sg"
}
+ vpc_id              = (known after apply)
}
```

Plan: 3 to add, 0 to change, 0 to destroy.

In this section, we have completed validating our code and reviewed which resources will be modified once the plan executes.

12 EXECUTE THE PLAN

Once the execution plan is validated, we can now apply or execute the plan to create a resource group.

1. Run **terraform apply** to execute the code. Type **yes** when prompted or you can use **terraform apply --auto-approve** and it will approve automatically.

```
terraform apply
or
terraform apply --auto-approve
```

```
root@ip-172-31-85-222:/home/ubuntu/amazon/Amazon-app-Deployment-using-terraform-and-jenkins/JENKINS-TF# terraform apply --auto-approve
aws_security_group.Jenkins-sg: Refreshing state... [id=sg-085e16a621e722263]
```

```
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create
```

```
Terraform will perform the following actions:
```

```
# aws_instance.web will be created
+ resource "aws_instance" "web" {
  + ami                  = "ami-080e1f13689e07408"
  + arn                  = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone     = (known after apply)
  + cpu_core_count       = (known after apply)
  + cpu_threads_per_core  = (known after apply)
  + disable_api_stop     = (known after apply)
  + disable_api_termination = (known after apply)
  + ebs_optimized        = (known after apply)
  + get_password_data     = false
  + host_id              = (known after apply)
  + host_resource_group_arn = (known after apply)
  + iam_instance_profile  = (known after apply)
  + id                   = (known after apply)
  + instance_initiated_shutdown_behavior = (known after apply)
  + instance_lifecycle   = (known after apply)
  + instance_state       = (known after apply)
  + instance_type        = "t2.large"
```

```
+ root_block_device {
  + delete_on_termination = true
  + device_name            = (known after apply)
  + encrypted              = (known after apply)
  + iops                   = (known after apply)
  + kms_key_id             = (known after apply)
  + tags_all               = (known after apply)
  + throughput             = (known after apply)
  + volume_id              = (known after apply)
  + volume_size            = 30
  + volume_type            = (known after apply)
}
}
```

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

aws_instance.web: Creating...

aws_instance.web2: Creating...

aws_instance.web: Still creating... [10s elapsed]

aws_instance.web2: Still creating... [10s elapsed]

aws_instance.web: Still creating... [20s elapsed]

aws_instance.web2: Still creating... [20s elapsed]

aws_instance.web: Creation complete after 22s [id=i-0db2182abaedd9a98]

aws_instance.web2: Creation complete after 22s [id=i-0fd206d99d481c893]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Note: It will take 2-3 minutes for completion.

2. Go to your **AWS console** and check out the EC2 instances.

Instances (2) Info						
<input type="text" value="Find Instance by attribute or tag (case-sensitive)"/>				All states ▾		
<input type="checkbox"/>	Name ↗	Instance ID	Instance state		Instance type	Status check
<input type="checkbox"/>	amazon clone	i-054058dbb4f23521c	Running		t2.large	2/2 checks passed
<input type="checkbox"/>	Ubuntu	i-0b0da94ef33f3784d	Running		t2.micro	2/2 checks passed

The screenshot displays the AWS Management Console for the 'Jenkins-Security Group'. The left sidebar shows the navigation menu with categories like EC2 Dashboard, Instances, Images, and Elastic Block Store. The main content area shows the security group details:

- Security group name:** Jenkins-Security Group
- Security group ID:** sg-07040fe41b462285c
- Description:** Open 22,443,80,8080,9000
- VPC ID:** vpc-0edf8a60c9563ac93
- Owner:** 767397866747
- Inbound rules count:** 6 Permission entries
- Outbound rules count:** 1 Permission entry

Below the details, the 'Inbound rules' tab is selected, showing a table of 6 inbound rules:

Name	Security group rule...	IP version	Type	Protocol	Port range
-	sg-r-0317530b623fcb4ed	IPv4	Custom TCP	TCP	8080
-	sg-r-0b044655a0c7aee89	IPv4	Custom TCP	TCP	9000
-	sg-r-0076ca9a18b5589...	IPv4	SSH	TCP	22
-	sg-r-0b393118fd969acbc	IPv4	HTTPS	TCP	443
-	sg-r-0b7aa275bca7169cb	IPv4	HTTP	TCP	80
-	sg-r-07c0d63c3a860639f	IPv4	Custom TCP	TCP	3000

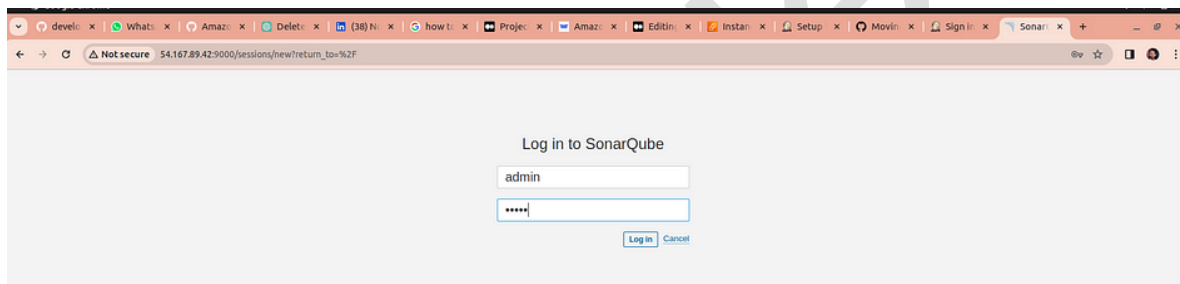
Here, we see the Amazon app instance is created by Terraform with the given configuration.

13 SETUP SONARQUBE AND JENKINS

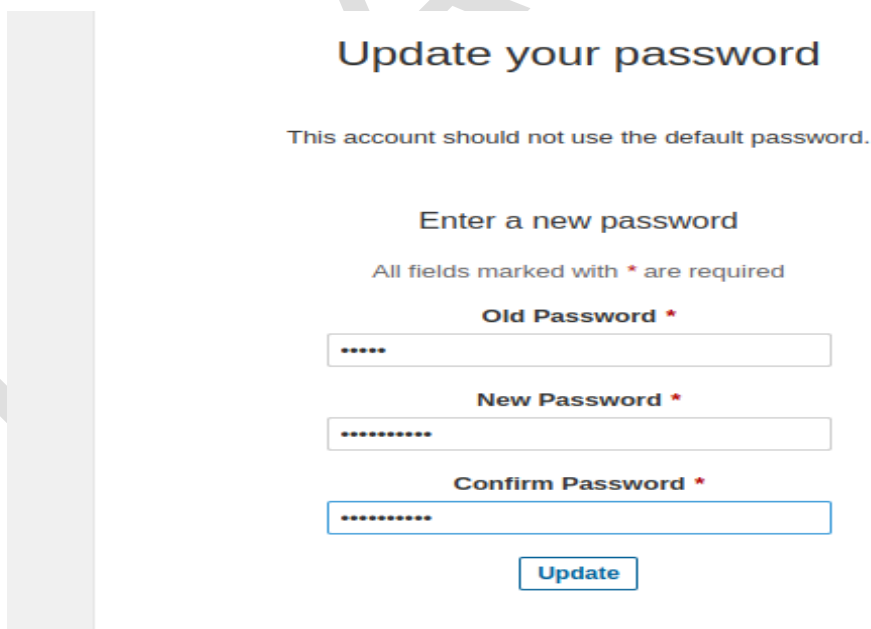
13.1 Setup Sonarqube

SonarQube is a platform used for continuous inspection of code quality to perform automatic reviews with static analysis of code to detect bugs, code smells, and security vulnerabilities. This step involves accessing SonarQube through its URL and configuring it by changing the default login credentials, setting up necessary configurations, and checking the version of Trivy, a vulnerability scanner.

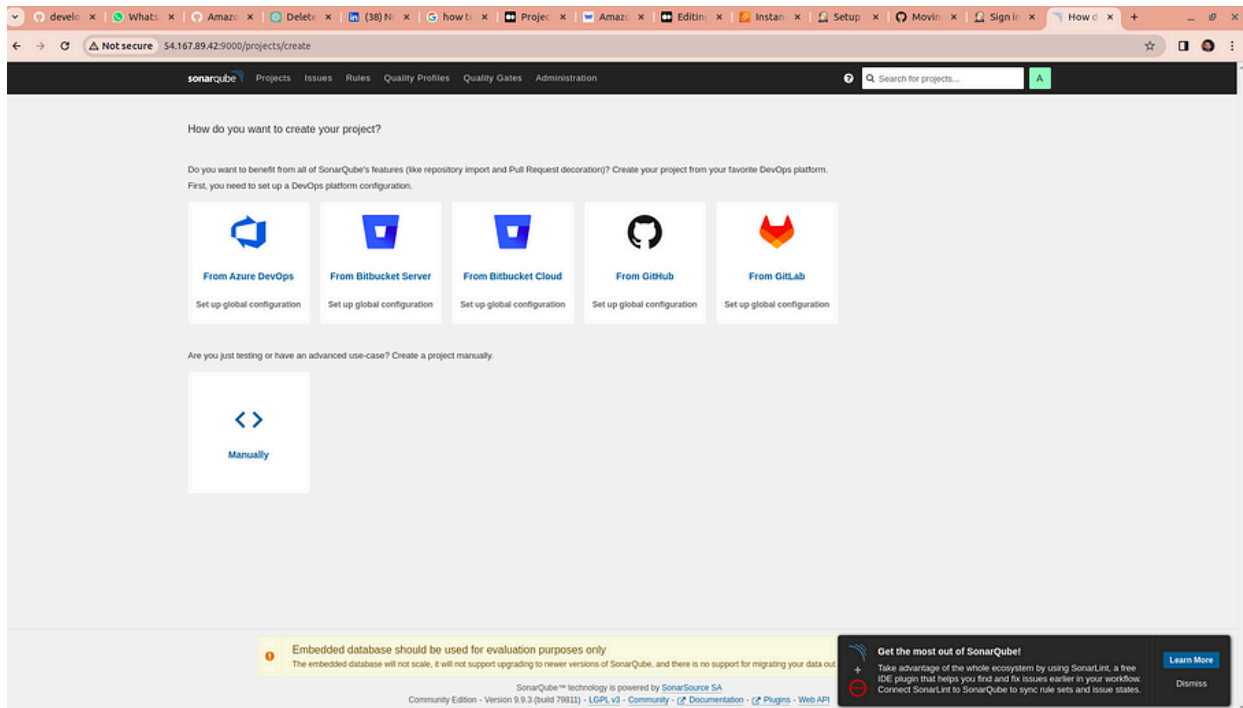
1. Copy your **public ip** of your **Amazon clone instance**.
2. Go to your browser and type →<publicip>:9000.
3. Initially, the **username** and **password** are admin.



4. Update your password.



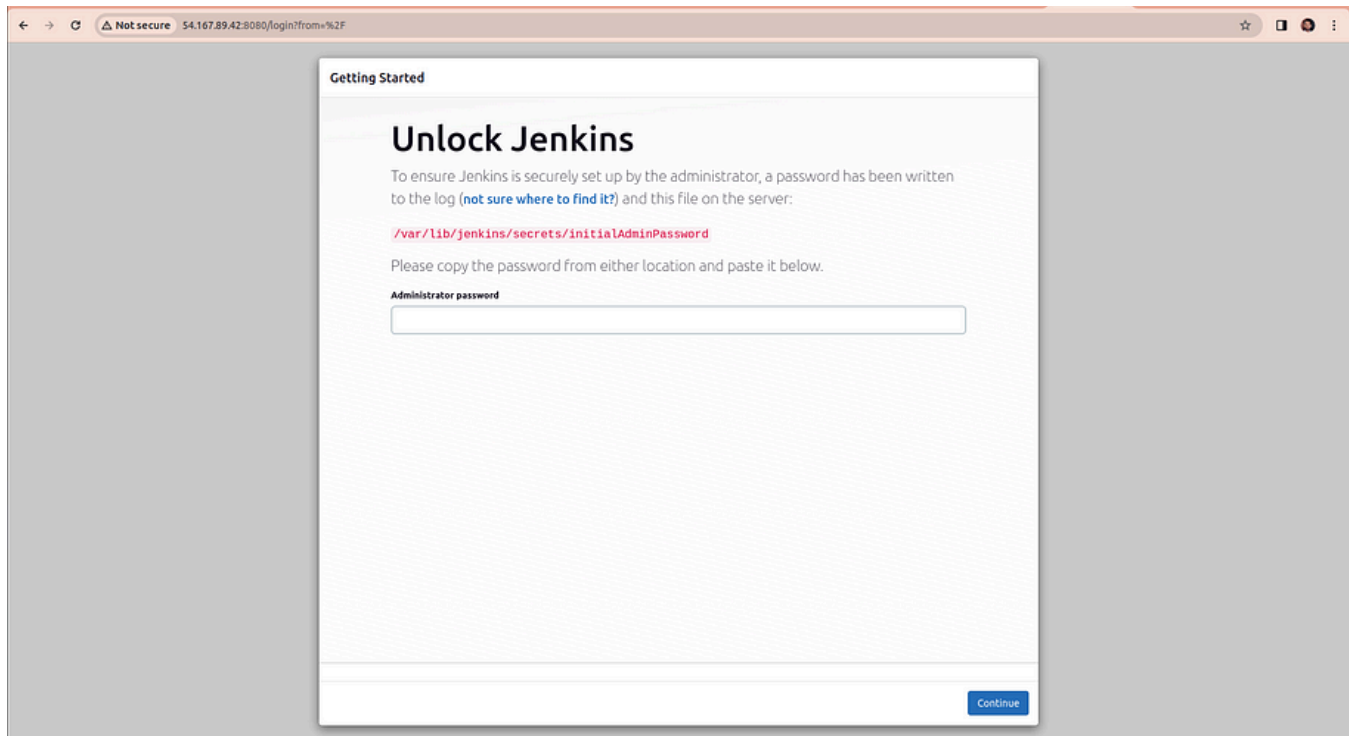
5. Welcome window of SonarQube.



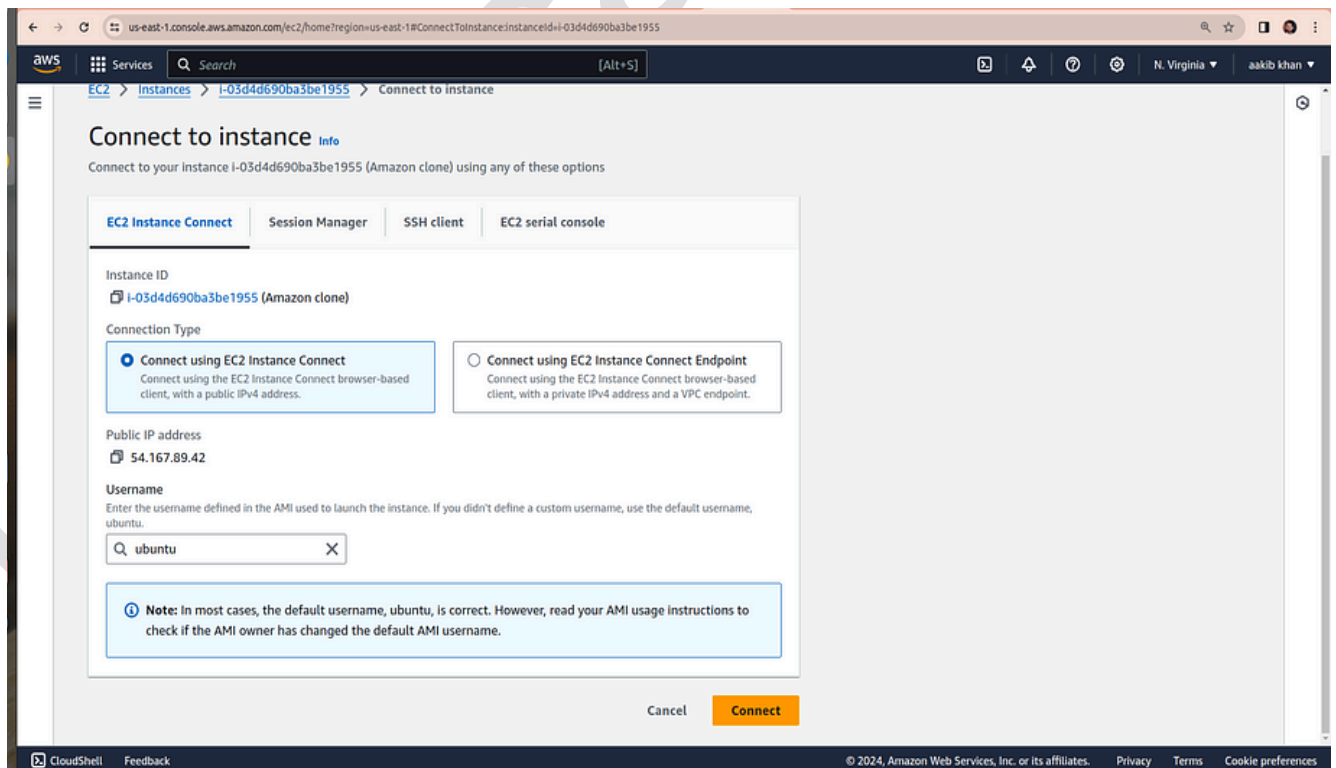
13.2 Setup Jenkins

Jenkins provides a variety of plugins to extend its functionality. In this step, you'll navigate to the plugin management section in Jenkins and install specific plugins required for the CI/CD pipeline without the need to restart Jenkins. These plugins include tools for managing Java, Node.js, and integrating with SonarQube for code analysis.

1. On browser type → **<public_ip>:8080**



2. Now for the **Password**, go to your Amazon EC2 and connect it.



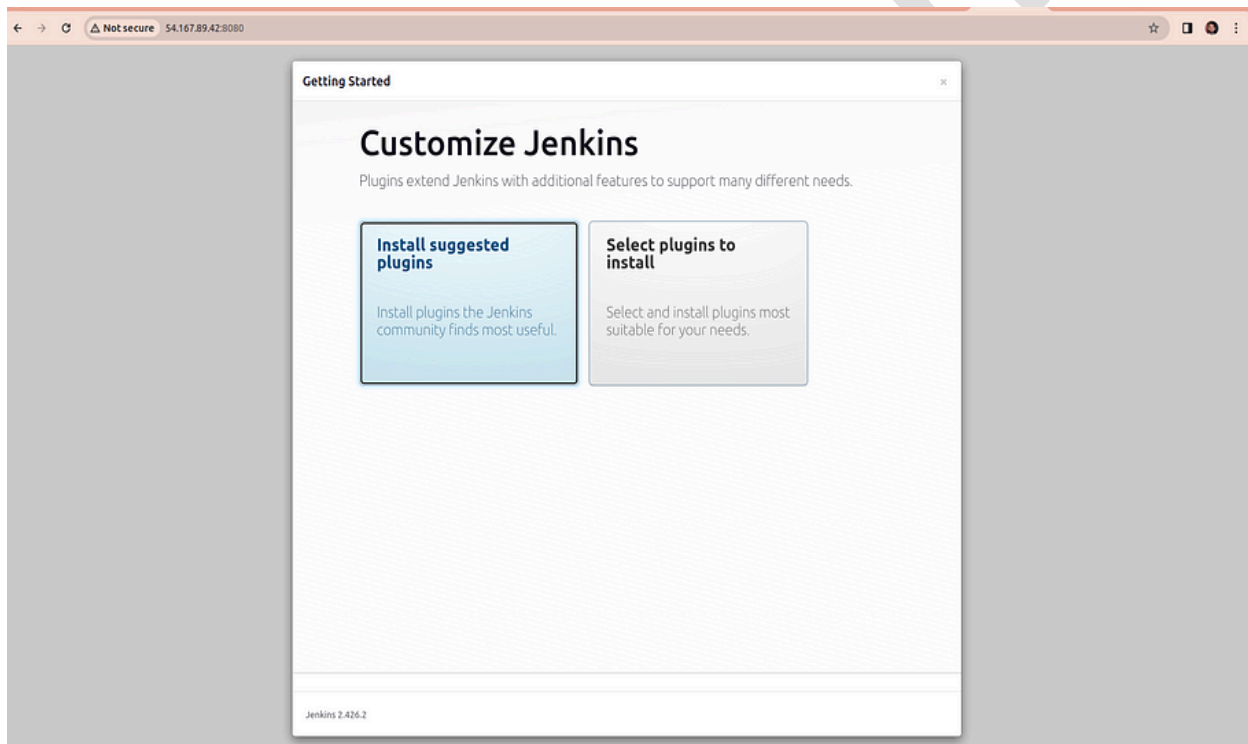
3. Run the below commands

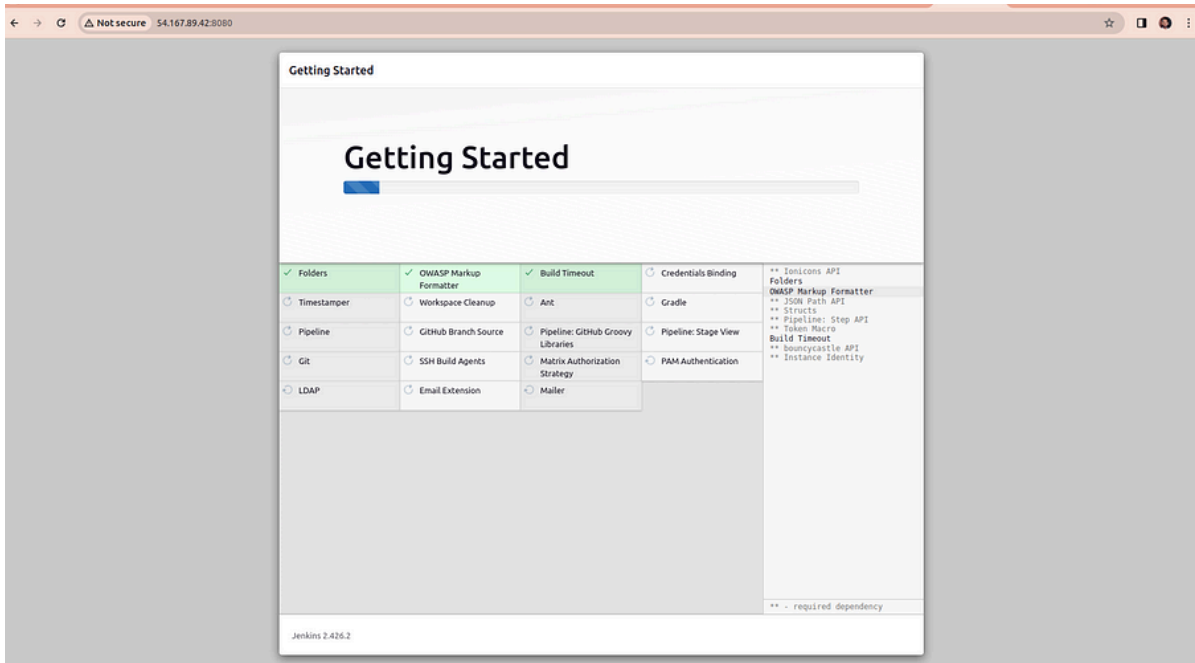
```
sudo su  
&  
cat /var/lib/jenkins/secrets/initialAdminPassword
```

```
ubuntu@ip-172-31-60-240:~$ sudo su  
root@ip-172-31-60-240:/home/ubuntu# cat /var/lib/jenkins/secrets/initialAdminPassword  
8d66b764789e4fcaa1b07314c4c19e73
```

Output is your password and paste it to your Jenkins.

4. Install the suggested plugins





5. Set up your Jenkins user

Getting Started

Username

deepsharma

Password

.....

Confirm password

.....



Full name

DeepakSharma

Jenkins 2.452.2

[Skip and continue as admin](#) [Save and Continue](#)

Manage Jenkins

 Search settings 

Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#).

Set up agent

Set up cloud

Dismiss

Java 17 end of life in Jenkins

More Info

Ignore

You are running Jenkins on Java 17, support for which will end on or after Mar 31, 2026. Refer to [the documentation](#) for more details.

System Configuration



System

Configure global settings and paths.



Tools

Configure tools, their locations and automatic installers.

Now you have successfully set up Sonareqube and Jenkins.

14 SET UP CI-CD PIPELINE

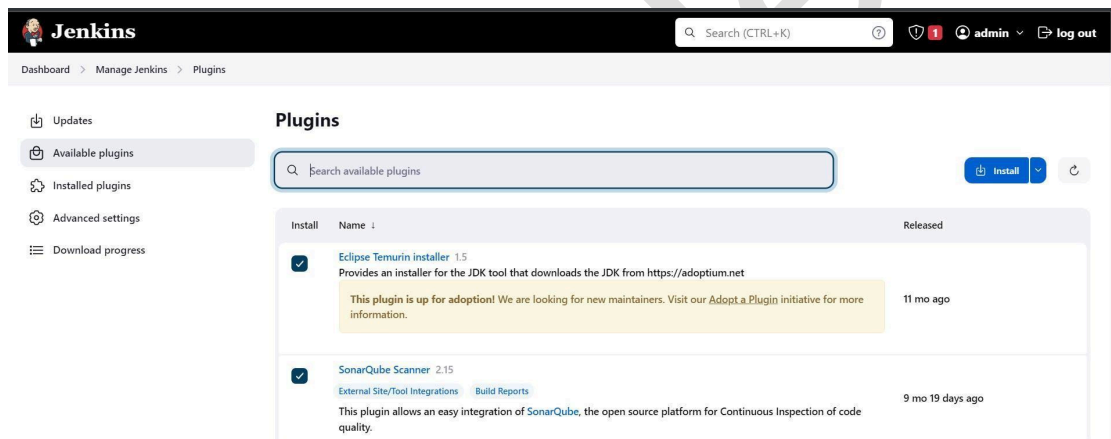
A CI/CD pipeline (Continuous Integration and Continuous Deployment/Delivery) is a set of automated processes and tools used to build, test, and deploy software. The primary goals are to ensure that code changes are reliably tested and deployed quickly and with minimal manual intervention, thus enhancing software development efficiency and quality.

14.1 Install Plugins

1. Goto Manage Jenkins → Plugins → Available Plugins → Install below plugins

1 → **Eclipse Temurin Installer** (Install without restart)

2 → **SonarQube Scanner** (Install without restart)



3 → **NodeJS Plugin** (Install Without restart)



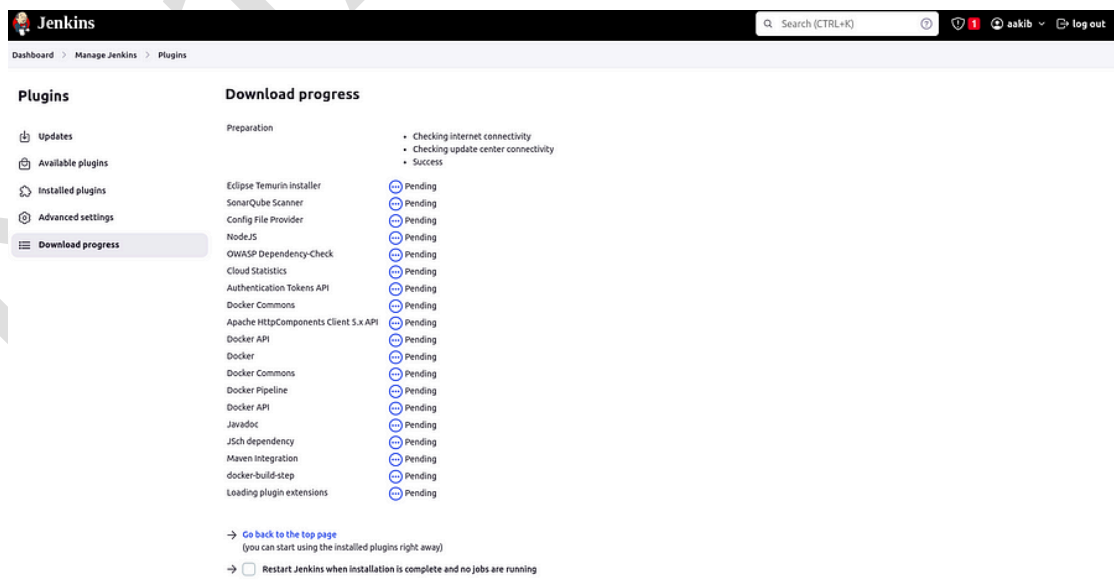
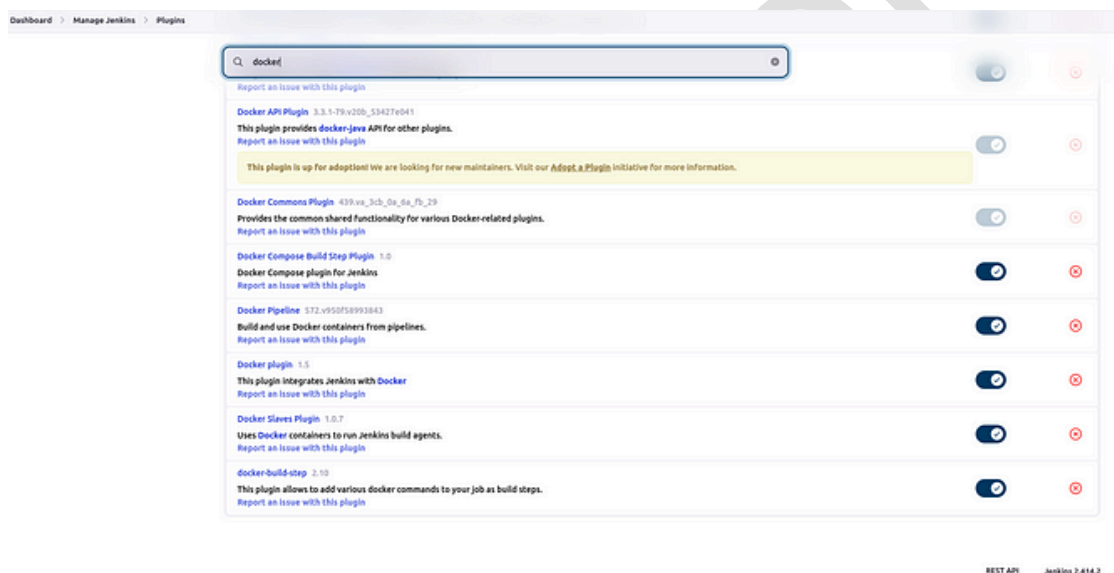
4 → **owasp** → The OWASP Plugin in Jenkins is like a “security assistant” that helps you find and fix security issues in your software. It uses the knowledge and guidelines from the Open Web Application Security Project (OWASP) to scan your web applications and provide suggestions on how to make them more secure. It’s a

tool to ensure that your web applications are protected against common security threats and vulnerabilities.



5 → **Prometheus metrics** → to monitor Jenkins on the Grafana dashboard

6 → Download all the **Docker**-related plugins



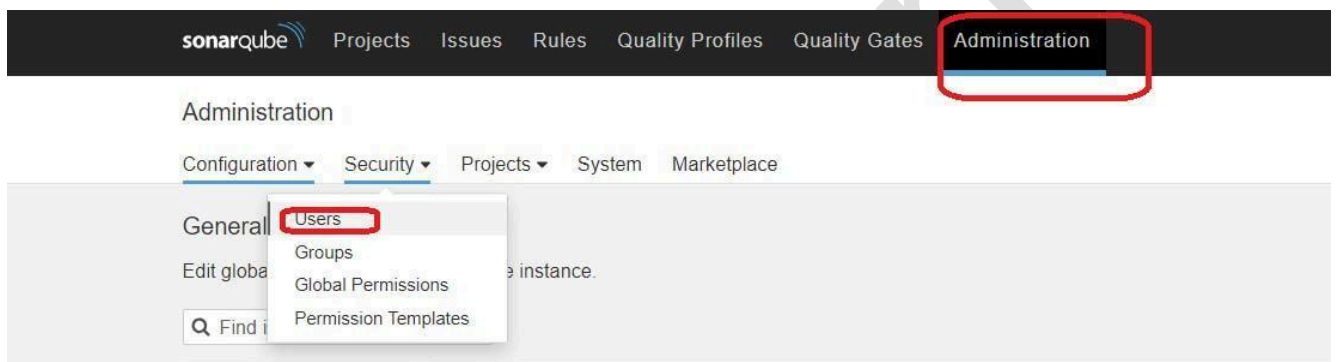
It will download all the required plugins.

14.2 Add Credentials for SonarQube and Docker

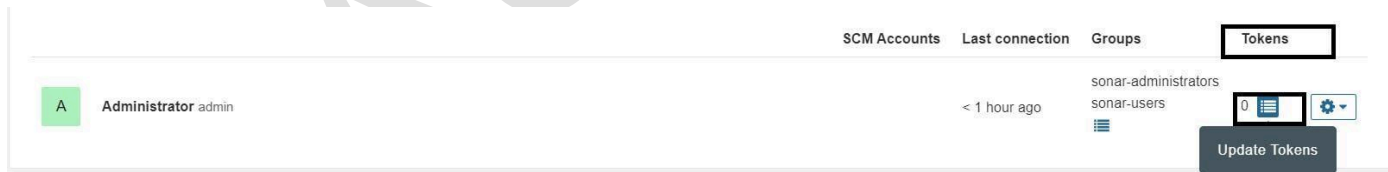
Set up Sonarqube Credentials

First, we generate a token for SonarQube to use in Jenkins credentials as secret text.

1. Go to <http://publicip:9000>
2. Now enter your **username** and **password**



3. Click on security → **users** → **token** → **generate token**



4. **token_name==jenkins**

Tokens of Administrator

Generate Tokens

Name Expires in

New token "jenkins" has been created. Make sure you copy it now, you won't be able to see it again!

Copy `squ_f8b6827a0b9a8e502eccb6b63a7dd326c5f11905`

Name	Type	Project	Last use	Created	Expiration	
jenkins	User		Never	January 20, 2024	February 19, 2024	<input type="button" value="Revoke"/>

[Done](#)

- Copy the token and go to your **Jenkins** → **manage Jenkins** → **credentials** → **global** → **add credentials**
- Select **secret text** from dropdown
- Secret text ==your token , id =jenkins** → click on **create**

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

New credentials

Kind

Scope ?

Secret

ID ?

Description ?

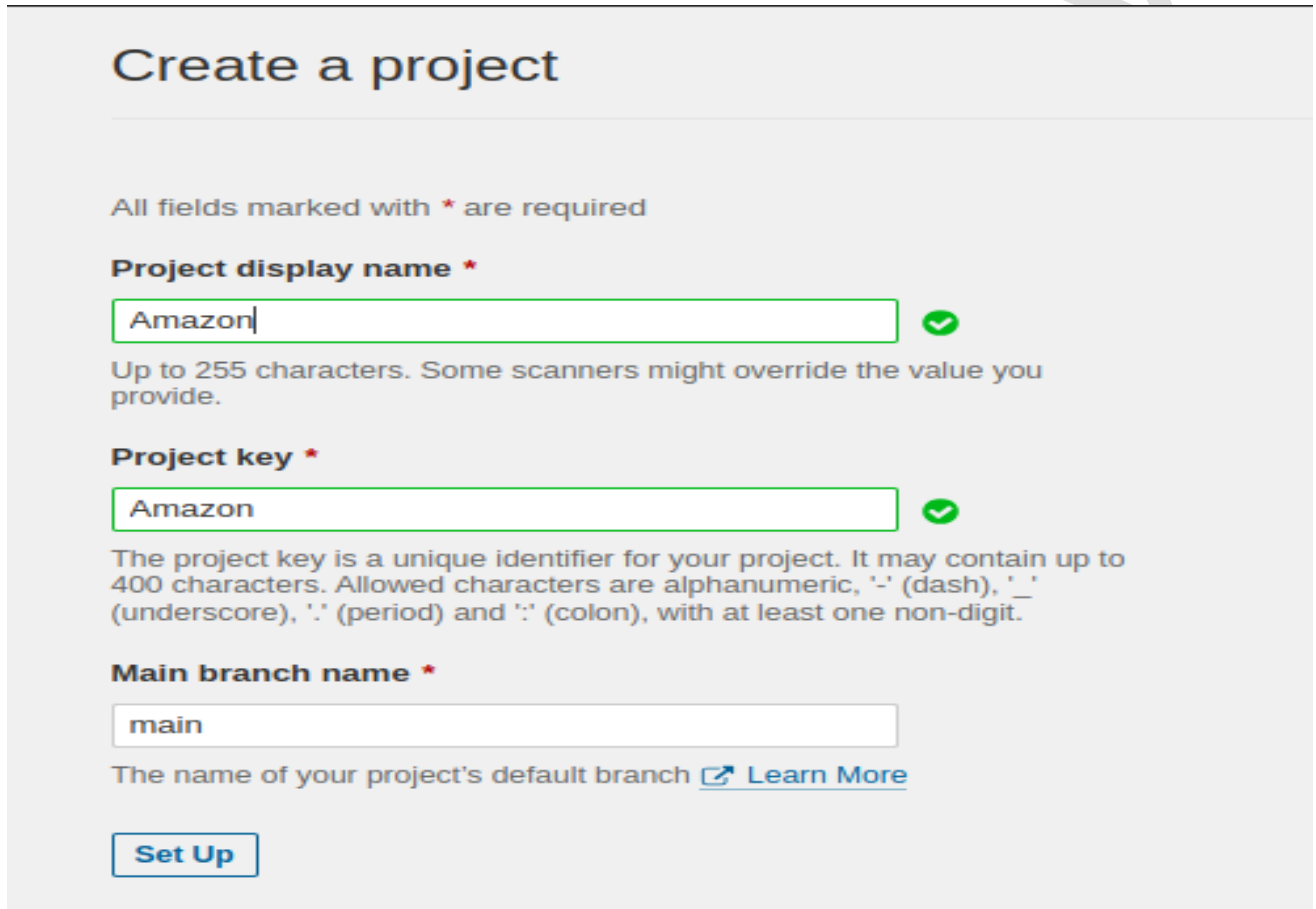
- You will find this page once you click on create.

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description	
Sonar-token	sonar	Secret text	sonar	

Set up Projects in SonarQube for Jenkins:

1. Go to your **SonarQube** server
2. Click on **projects**
3. In the name field, type **Amazon**, then click on **set up**



Create a project

All fields marked with * are required

Project display name *

Amazon ✓

Up to 255 characters. Some scanners might override the value you provide.

Project key *

Amazon ✓

The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.

Main branch name *

main

The name of your project's default branch [Learn More](#)

Set Up

4. Now Click on **Locally**.




5. Now add the Token name and click on **generate**.

Analyze your project


We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token

☒ Generate a project token

Token name  **Expires in**

Analyze "Amazon" 30 days **Generate**

 Please note that this token will only allow you to analyze the current project. If you want to use the same token to analyze multiple projects, you need to generate a global token in your [user account](#). See the [documentation](#) for more information.

☐ Use existing token

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point in time in your [user account](#).

2 Run analysis on your project

6. Click on Continue.


Amazon ☆ main

Overview Issues Security Hotspots Measures Code Activity Project Settings ▾ Project Information

Analyze your project

We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token

Analyze "Amazon": `sqp_2b3d416b7aa9393aff3c43d1618f611bed95bcbf3` 

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point in time in your [user account](#).

[Continue](#)

2 Run analysis on your project

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

Search for projects... A

Amazon ☆ main

Overview Issues Security Hotspots Measures Code Activity Project Settings ▾ Project Information

Analyze your project

We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token ✓ Analyze "Amazon"-sqp_2b3d416b7aa9393aff3c43d1618f611bed95bcbf3

2 Run analysis on your project

What option best describes your build?

Maven Gradle **MET** Other (for J5, TS, Go, Python, PHP...)

What is your OS?

Linux Windows macOS

Download and unzip the Scanner for Linux

Visit the [official documentation of the Scanner](#) to download the latest version, and add the `bin` directory to the `PATH` environment variable

Execute the Scanner

Running a SonarQube analysis is straightforward. You just need to execute the following commands in your project's folder:

```
sonar-scanner \
  -Dsonar.projectKey=Amazon \
  -Dsonar.sources=. \
  -Dsonar.host.url=http://52.23.179.38:9000 \
  -Dsonar.login=sqp_2b3d416b7aa9393aff3c43d1618f611bed95bcbf3
```

[Copy](#)

Please visit the [official documentation of the Scanner](#) for more details.

Is my analysis done? If your analysis is successful, this page will automatically refresh in a few moments.

You can set up Pull Request Decoration under the project settings. To set up analysis with your favorite CI tool, see the tutorials.

Check these useful links while you wait: [Branch Analysis](#), [Pull Request Analysis](#).

Embedded database should be used for evaluation purposes only

The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out

Get the most out of SonarQube!

Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow.

Connect SonarLint to SonarQube to sync rule sets and issue states.

[Learn More](#)

Dismiss

The SonarQube project for Jenkins is set up now.

Set up Docker Credentials

1. Go to your **Jenkins** → **manage jenkins** → **credentials** → **global** → **add credentials**
2. Provide your **username** and **password** of your Docker Hub
3. **id==docker**

New credentials

Kind
Username with password

Scope
Global (Jenkins, nodes, items, all child items, etc)

Username
aakibkhan1212

☐ Treat username as secret

Password

ID
docker

Description

Create

REST API Jenkins 2.426.2

Global credentials (unrestricted)

+ Add Credentials

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
jenkins	jenkins	Secret text	jenkins
docker	aakibkhan1212/*****	Username with password	

Icons: S M L

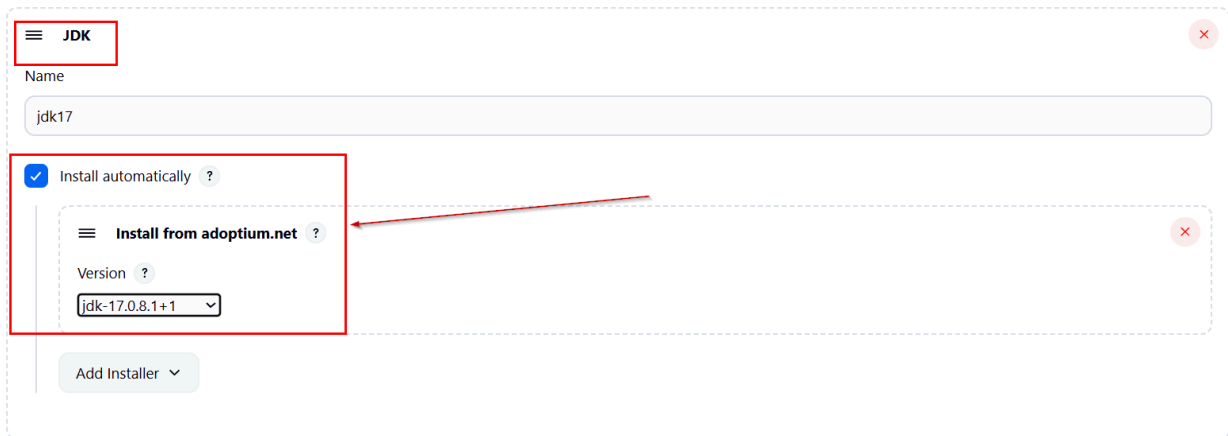
Now credentials for both are set up.

15 SET UP THE TOOLS FOR JENKINS

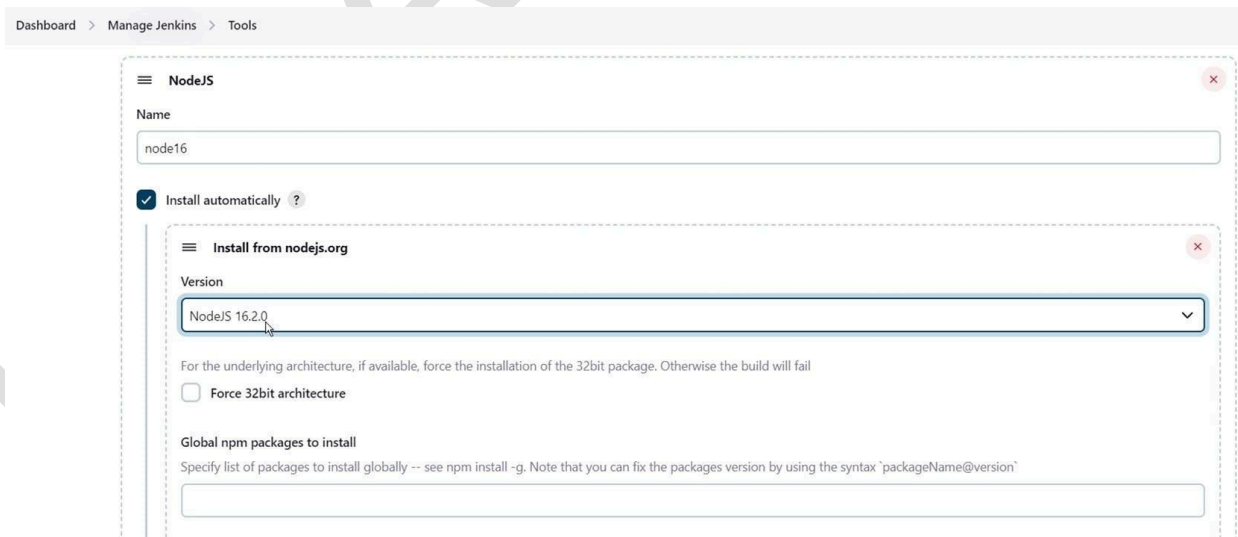
Go to manage Jenkins → tools

15.1 Add JDK & NodeJS

1. Click on add jdk and select **installer adoptium.net**
2. Choose **jdk 17.0.8.1+1 version** and in the name section enter **jdk17**



3. Click on add **nodejs**
4. Enter **node16** in name section
5. Choose version **nodejs 16.2.0**



15.2 Add Docker & SonarQube

1. Click on **add docker**
2. name==**docker**
3. Add installer ==**download from docker.com**

Dashboard > Manage Jenkins > Tools

Docker installations

Add Docker

×

≡ Docker

Name

docker

☒ Install automatically ?

≡ Download from docker.com

Docker version ?

latest

Add Installer ▾

4. Add a **sonar scanner**
5. name ==sonar-scanner

Dashboard > Manage Jenkins > Tools

SonarQube Scanner installations

Add SonarQube Scanner

×

≡ SonarQube Scanner

Name

sonar-scanner

☒ Install automatically ?

≡ Install from Maven Central

Version

SonarQube Scanner 5.0.1.3006 ▾

Add Installer ▾

Add SonarQube Scanner

Save Apply

15.3 Add OWASP dependency check

Adding the Dependency-Check plugin in the “Tools” section of Jenkins allows you to perform automated security checks on the dependencies used by your application

1. Add **dependency check**
2. name == **DP-Check**
3. From the add installer, select **install from github.com**

Dashboard > Manage Jenkins > Tools

Dependency-Check installations

Add Dependency-Check

≡ **Dependency-Check**

Name

DP-Check

☒ Install automatically ?

≡ **Install from github.com**

Version

dependency-check 6.5.1

Add Installer ▾

Click one save. Now we have configured all the required tools.

16 CONFIGURE GLOBAL SETTINGS FOR SONARUBE

1. Go to **manage Jenkins** → **System** → **Configure global setting** → **add SonarQube servers**
2. name == **sonar-server**
3. Server_url == **http://public_ip:9000**
4. Server authentication token == **jenkins** → **it is created in SonarQube security configurations**

SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

☐ Environment variables

SonarQube installations

List of SonarQube installations

Name
sonar-server

Server URL
Default is http://localhost:9000
http://52.23.179.38:9000

Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
jenkins

+ Add

Advanced

16.1 Run the Pipeline

1. Go to new item → **select pipeline** → in the name section type **amazon-pipeline**
2. Scroll down to the **pipeline script** and copy and paste the following code from the **link below**
<https://github.com/k21academyuk/Amazon-app-Deployment-using-terraform-and-jenkins/blob/main/Pipeline>
3. Before saving, change the **image repository** to your **Docker Hub username** as shown in the screenshot below.

```

48     steps {
49         sh "trivy fs . > trivyfs.txt"
50     }
51 }
52 stage("Docker Build & Push"){
53     steps{
54         script{
55             withDockerRegistry(credentialsId: 'docker', toolName: 'docker'){
56                 sh "docker build -t amazon-clone ."
57                 sh "docker tag amazon-clone deepsharma/amazon-clone:latest "
58                 sh "docker push deepsharma/amazon-clone:latest "
59             }
60         }
61     }
62 }
63 stage("TRIVY"){
64     steps{
65         sh "trivy image deepsharma/amazon-clone:latest > trivyimage.txt"
66     }
67 }
68 stage('Deploy to container'){
69     steps{
70         sh 'docker run -d --name amazon-clone -p 3000:3000 deepsharma/amazon-clone:latest'
71     }
72 }
73 }
74 }

```

Insert your dockerhub
username

4. Click **apply** and **save**.

Dashboard > netflix > Configuration

Configure

General

Advanced Project Options

Pipeline

Pipeline

Definition

Pipeline script

Script ?

```

3 agent any
4 tools{
5     jdk "jdk17"
6     nodejs "node16"
7 }
8 environment {
9     SCANNER_HOME=tool "sonar-scanner"
10 }
11 stages {
12     stage('clean workspace'){
13         steps{
14             cleanWs()
15         }
16     }
17     stage('Checkout from Git'){
18         steps{
19             git branch: 'main', url: 'https://github.com/K21Academyuk/Deploy-Netflix-Clone-on-Kubernetes.git'

```

☒ Use Groovy Sandbox ?

Pipeline Syntax

Save Apply

5. Click on build now → it will take about 10–15 min

- Status
- Changes
- Build Now**
- Configure
- Delete Pipeline
- Full Stage View
- SonarQube
- Rename
- Pipeline Syntax

6. You could check out the console output.

```
Dashboard > Amazon-pipeline > #2

[Pipeline] /
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { [DMSFP FS SCAN]
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] dependencyCheck
Unpacking https://github.com/jeremylong/DependencyCheck/releases/download/v9.8.9/dependency-check-9.8.9-release.zip to
/var/lib/jenkins/tools/org.jenkinsci.plugins.DependencyCheck.tools.DependencyCheckInstallation/DP-Check on Jenkins
[INFO] Checking for updates
[WARN] An NVD API Key was not provided - it is highly recommended to use an NVD API key as the update can take a VERY long time without an API Key
[INFO] NVD API has 236,458 records in this update
[INFO] Downloaded 10,000/236,458 (4%)
[INFO] Downloaded 20,000/236,458 (8%)
[INFO] Downloaded 30,000/236,458 (13%)
[INFO] Downloaded 40,000/236,458 (17%)
[INFO] Downloaded 50,000/236,458 (21%)
[INFO] Downloaded 60,000/236,458 (25%)
[INFO] Downloaded 70,000/236,458 (30%)
[INFO] Downloaded 80,000/236,458 (34%)
[INFO] Downloaded 90,000/236,458 (38%)
[INFO] Downloaded 100,000/236,458 (42%)
[INFO] Downloaded 110,000/236,458 (47%)
[INFO] Downloaded 120,000/236,458 (51%)
[INFO] Downloaded 130,000/236,458 (55%)
[INFO] Downloaded 140,000/236,458 (59%)
[INFO] Downloaded 150,000/236,458 (63%)
[INFO] Downloaded 160,000/236,458 (68%)
}
```

Note: While running the pipeline script if you are facing any issue of **Build in-progress** or **Build Failure**, Follow the [troubleshooting section 19.1](#).

Note: While running the pipeline script if you are facing any issue of **Could not find 'java' executable in JAVA_HOME or PATH**, follow the [troubleshooting section 19.2](#).

7. owasp dependency check result

Dependency-Check Results

SEVERITY DISTRIBUTION

File Name	Vulnerability	Severity	Weakness
+ axios:1.3.4	OSVINDEX CVE-2023-45857	Medium	CWE-352
+ css-what:3.4.2	OSVINDEX CVE-2022-21222	High	CWE-1333
+ ejs:3.1.9	NVD CVE-2023-29827	Critical	CWE-74
+ follow-redirects:1.15.2	NVD CVE-2023-26159	Medium	CWE-601
+ nth-check:1.0.2	NVD CVE-2021-3803	High	CWE-1333
+ postcss:7.0.39	NVD CVE-2023-44270	Medium	CWE-74
+ postcss:8.4.21	NVD CVE-2023-44270	Medium	CWE-74
+ semver:6.3.0	OSVINDEX CVE-2022-25883	High	CWE-1333
+ semver:7.3.8	OSVINDEX CVE-2022-25883	High	CWE-1333
+ tough-cookie:4.1.2	NVD CVE-2023-26136	Critical	CWE-1321

Jenkins 2.426.2

8. After a lot of errors, we did it.

Amazon-pipeline

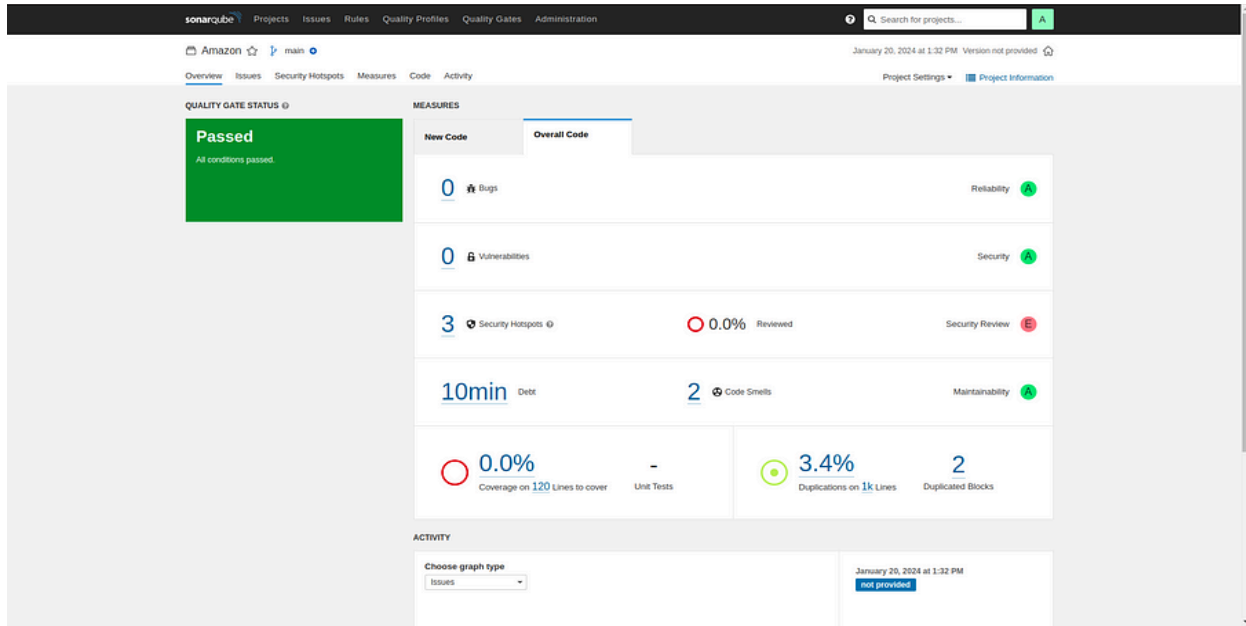
Dependency-Check Trend

Legend: Unassigned (grey), Low (green), Medium (yellow), High (orange), Critical (red)

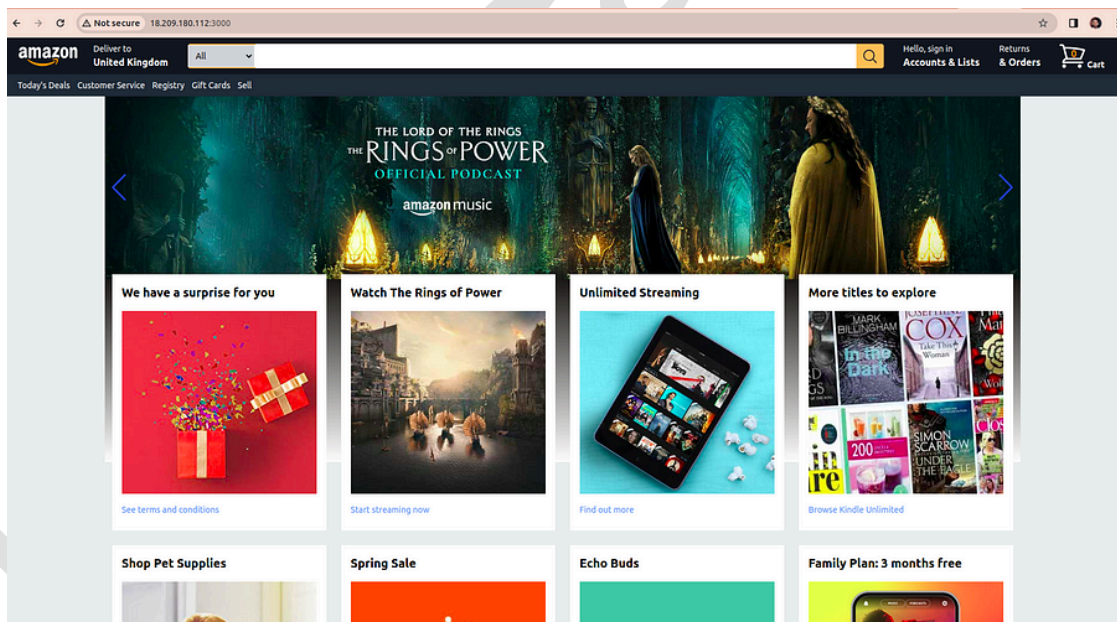
Stage View

	Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY	Deploy to container
Average stage times: (Average full run time: ~9min 8s)	145ms	265ms	865ms	25s	344ms	16s	3min 20s	23s	1min 20s	1min 35s	845ms
#17 20 Jan 2024, 14:54	147ms	274ms	856ms	23s	382ms	16s	3min 14s	23s	1min 35s	3min 9s	1s
#16 20 Jan 2024, 14:46	144ms	257ms	875ms	27s	306ms	16s	3min 26s	23s	1min 5s	49ms	50ms

SonarQube Quality Gate



9. Now our Docker image is built, pushed, and deployed into a container, and our app is live and running on **port no. 3000** to check → **http://<your-public-ip>:3000**



Hurrah! Here's our Amazon app

17 SHARE YOUR LEARNINGS ON LINKEDIN & COMMUNITY

In this section, you are going to share whatever you learned during this lab on LinkedIn and in the community.

Please watch the below 📺 video below to understand why it's important to post & how to post:

<https://www.skool.com/k21academy/classroom/0c24b6af?md=f7975c8e525b4d5d8191e131bdef5740>

LinkedIn for Visibility & Authority ...

40%

LinkedIn: Update Profile, Post Content: Wh...

FAQ: If I Publish, What Current Employer T...

Why it's Imp. to Prepare & Share Not... ✓

Example Post: What & How to Post ... ✓

Useful ChatGPT Prompts ✓

How to create LinkedIn Banner from Canv... ✓

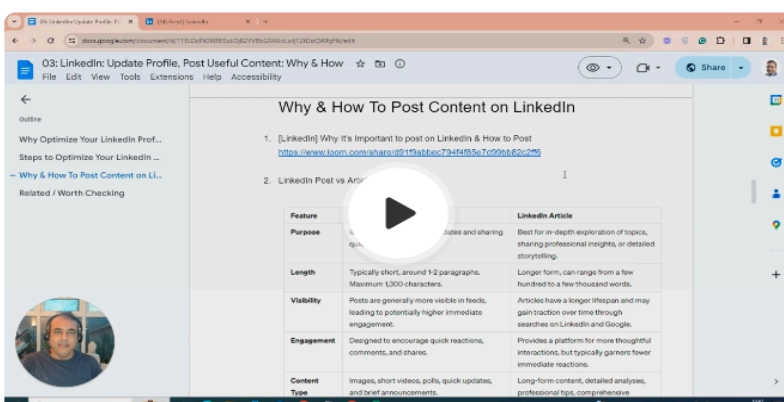
LinkedIn Profile Review ^

Neha: LinkedIn Feedback to Updat... ✓

Deb: Australia Azure + Oracle Cloud ✓

Gagan: LinkedIn Profile Feedback ✓

LinkedIn: Update Profile, Post Content: Why & How



Feature	LinkedIn Post vs Article	LinkedIn Article
Purpose	Quick updates and sharing	Best for in-depth exploration of topics, sharing professional insights, or detailed storytelling.
Length	Typically short, around 1-2 paragraphs. Maximum 1,300 characters.	Longer form, can range from a few hundred to a few thousand words.
Visibility	Posts are generally more visible in feeds, leading to potentially higher immediate engagement.	Articles have a longer lifespan and may gain traction over time through searches on LinkedIn and Google.
Engagement	Designed to encourage quick reactions, comments, and shares.	Provides a platform for more thoughtful interactions, but typically garners fewer immediate reactions.
Content Type	Images, short videos, polls, quick updates, and brief announcements.	Long-form content, detailed analyses, professional tips, comprehensive

Resources

[Related Document](#)

- As you know, building a strong professional profile is essential in today's job market. One way to showcase your skills and knowledge is by sharing your labs and projects on LinkedIn.
- Your LinkedIn profile is a powerful tool in your job search. Sharing your labs and projects is just one way to demonstrate your expertise and stand out to potential employers. If you don't have LinkedIn, we strongly recommend creating one for yourself.
- By doing so, you can demonstrate to potential employers or connections that you have hands-on experience in your field and are actively engaged in learning and growing your skills.

17.1 On LinkedIn

Take a screenshot and share it on your LinkedIn. This will attract recruiters and employers to your profile and increase your reach. **Do remember to tag**

K21Academy (<https://www.linkedin.com/company/k21academy>) &

Atul Kumar (<https://www.linkedin.com/in/atulk21academy/>) as we will circulate in our network too to increase your reach.

Example Video & Post

<https://www.skool.com/k21academy/classroom/0c24b6af>

LinkedIn for Visibility & Authority

40%

LinkedIn: Update Profile, Post Content: Wh...

FAQ: If I Publish, What Current Employer T...

Why it's Imp. to Prepare & Share Not...

Example Post: What & How to Post ...

Useful ChatGPT Prompts

How to create LinkedIn Banner from Canv...

LinkedIn Profile Review

Neha: LinkedIn Feedback to Updat...

Deb: Australia Azure + Oracle Cloud

Gagan: LinkedIn Profile Feedback

LinkedIn Profile Makeover Session

Feedback on Content Creation / Blo...

Example Post: What & How to Post & Commenting

Example Link >> https://www.linkedin.com/posts/mehta7_azureai-promptengineering-clouddevelopment-activity-724493685596605313-Yrti

Chat Prompt for commenting >>

Suggest useful, engaging comment for this linkedIn post that grabs recruiter and potetnial employers attention

https://www.linkedin.com/*****

Check this LinkedIn post example

https://www.linkedin.com/posts/izhar-alam-_azure-dp300-database-activity-7061214360462852097-FYQ5/

Here is a sample that includes steps inside a post followed by a screenshot:

Copyright© K21Academy | All Rights Reserved

Sharing, Reselling, or duplication of this content is strictly prohibited without K21Academy's written permission.



Adelakun Joshua • 3rd+

+ Follow ...

1d • Edited •

AWS CLOUDWATCH BILLING ALARM

Problem:

Today, one of my clients reached out to me. They have AWS billing challenges. They just got an email alert from AWS stating that, they have to pay certain amount of bills. When they evaluated their end of the month service bills, they discovered that they are billed for a lot of services that they are actually not using on a day-to-day basis. But unfortunately, they didnt properly monitor and manage their aws resources, which incur charges.

My recommendation:

I recommended AWS Cloudwatch alarm to my client. And i configure the cloudwatch alarm for them.

AWS Cloudwatch:

Amazon CloudWatch is a monitoring and management service that provides data and actionable insights for AWS, on-premises, hybrid, and other cloud applications and infrastructure resources.

How I Set the Cloudwatch Billing Alarm Up:

1. I Opened the CloudWatch console at <https://lnkd.in/gugv5tEE> ✓.
 2. In the navigation pane, I choose Alarms, and then choose All alarms. I Chosed Create alarm.
 3. Then, I Selected 'metric'. In Browse, I choose 'Billing', and then 'Total Estimated Charge'.
 4. I Selected the box for the 'EstimatedCharges metric', and then I choose 'Select metric'.
 5. For 'Statistic', I choose Maximum.
 6. For 'Period', I choose 6 hours.
 7. For 'Threshold type', I choose Static.
 8. For 'Whenever EstimatedCharges is' . . ., I selected Greater.
- For 'than' . . ., I defined the value that i want to cause the alarm to trigger. For example \$10 USD.

SNS topic included the email address by which my client will receive email when the billing amount crosses the billing threshold specified.

Note:

You can select an existing Amazon SNS topic, create a 'new Amazon SNS topic', or use a topic ARN to notify other account. If you want your alarm to send multiple notifications for the same alarm state or for different alarm states, choose 'Add notification'.

11. I chose Next.

12. Under Name and description, I entered a name for the alarm. (Optional) Enter a description of the alarm.

13. Then I Chose 'Next'.

14. Under 'Preview and create', after confirming that the configuration is correct, and then I chose 'Create alarm'.

Conclusion: Organization that properly set up a cloudwatch billing alarm, will be able to properly monitor their aws resources usage. The metrics provided by the cloudwatch alarm through the dashboard will also help the organization to effectively control their spending.

#aws

#awscloud

#JTechconsult

#K21Academy: Learn Cloud From Experts

#AtulKumar

#SumtiMehta

Cloudwatch billing alarm architectural diagram:



Other examples:

<http://go.k21academy.com/4ISBFcz>

<http://go.k21academy.com/4IS1GbD>

17.2 Share wins in the Progress Diary

Share screenshots in Cloud School Community under your **'Progress Diary'** (If you have not created a Progress Diary yet, then check at

<https://www.skool.com/k21academy/classroom/8f3838d7?md=ad7a2acc7df34280901462cfc65e9f16>).

This will boost your confidence in the progressive path you are following, and encourage/inspire others to perform these hands-on labs.

Start Here And Helpful Resources ...

0%

Start Here

How To Cancel Your Cloud School Membe...

Create a Progress Diary Post

FAQs

- How to Check Schedule & Live Session ...
- 1. Is there a Mobile App for K21 Acade...
- 2. Progress Tracking for Courses
- 3a. Adjust Video Speed 1.5x or 2.x
- 3b. Screen Quality Adjustment
- 4. How to Create Ubuntu Machine on A...
- 5. How long will we have access to Sko...

Create a Progress Diary Post

This post is designed to help you set up your Progress Diary, which will serve as a log for tracking your learning journey and achievements within the **K21 Academy** platform.

Course Name: DevOps Job Oriented

✅ **Lab name:** Install Jenkins on Amazon Linux 2023 AMI

Learning Summary: In this lab, I learned how to launch a Linux-based EC2 instance on AWS, install the Jenkins server on it, and perform initial configuration. I gained hands-on experience with deploying Jenkins on the cloud, setting it up for automation tasks, and preparing it for building and managing CI/CD pipelines.

18 CLEANUP RESOURCES

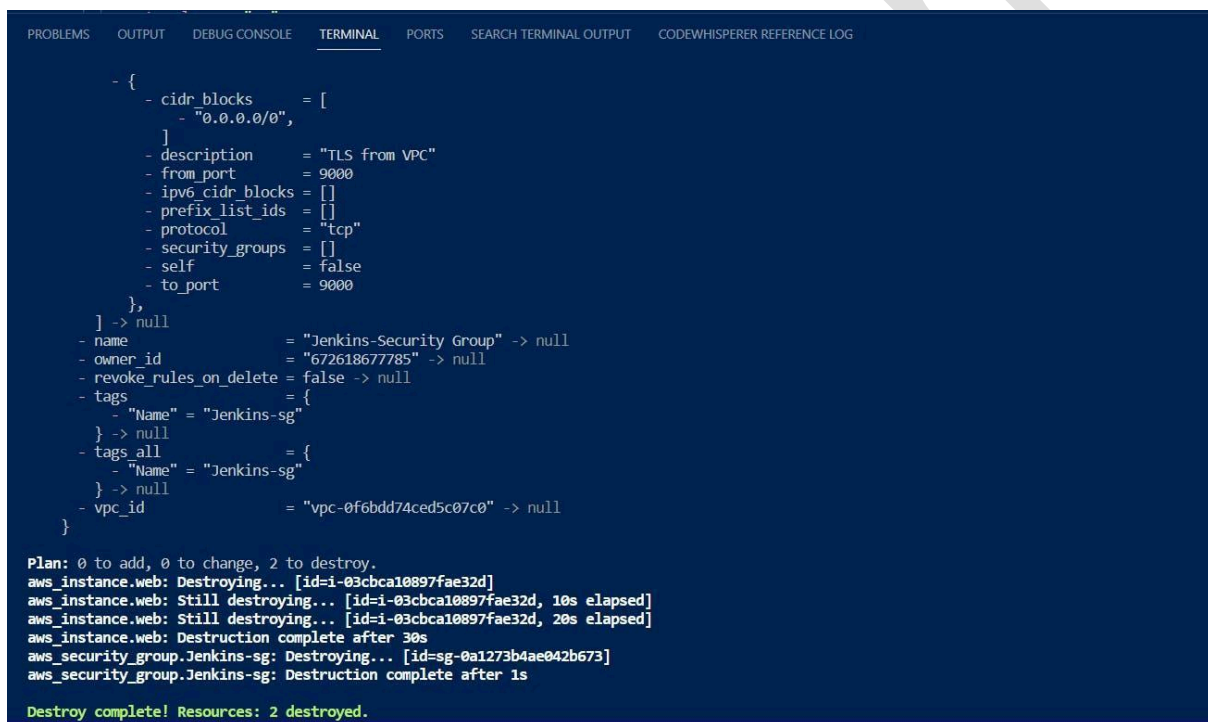
Note: Pls delete all the resources of this lab as they will not be used in the next lab.

18.1 Destroying the Infrastructure

1. Go to VS Code and provide the below command, or go to the path where you provisioned the EC2 instance.

```
terraform destroy --auto-approve
```

After 3 -5 minutes all things are destroyed



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH TERMINAL OUTPUT CODEWHISPERER REFERENCE LOG

- {
  - cidr_blocks = [
    - "0.0.0.0/0",
  ]
  - description = "TLS from VPC"
  - from_port   = 9000
  - ipv6_cidr_blocks = []
  - prefix_list_ids = []
  - protocol     = "tcp"
  - security_groups = []
  - self         = false
  - to_port      = 9000
},
] -> null
- name           = "Jenkins-Security Group" -> null
- owner_id       = "672618677785" -> null
- revoke_rules_on_delete = false -> null
- tags           = {
  - "Name" = "Jenkins-sg"
} -> null
- tags_all       = {
  - "Name" = "Jenkins-sg"
} -> null
- vpc_id         = "vpc-0f6bdd74ced5c07c0" -> null
}

Plan: 0 to add, 0 to change, 2 to destroy.
aws_instance.web: Destroying... [id=i-03cbca10897fae32d]
aws_instance.web: Still destroying... [id=i-03cbca10897fae32d, 10s elapsed]
aws_instance.web: Still destroying... [id=i-03cbca10897fae32d, 20s elapsed]
aws_instance.web: Destruction complete after 30s
aws_security_group.Jenkins-sg: Destroying... [id=sg-0a1273b4ae042b673]
aws_security_group.Jenkins-sg: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.
```

Now go to your EC2 and terminate your Instance.

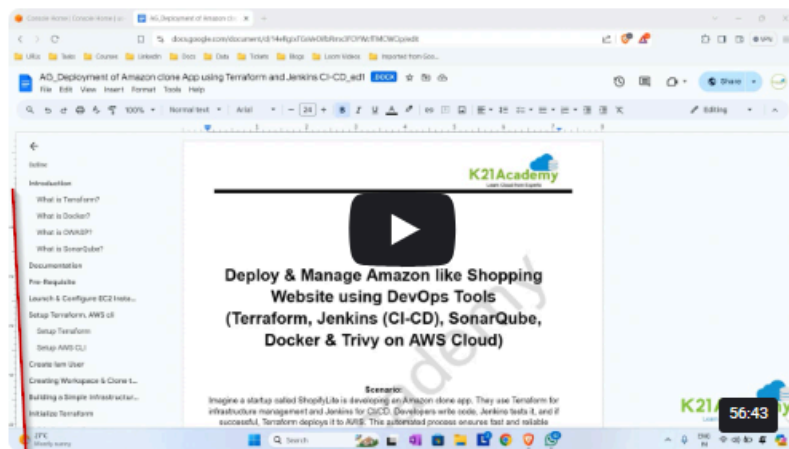
19 TROUBLESHOOTING

In this section, we outline some common issues you may encounter during the process, along with explanations of why these problems may occur. This will help you quickly identify and resolve any challenges you face.

Note: If you hit any issue in performing this lab, then add your issue in the thread below this lab (linked to this thread in the community)

<https://www.skool.com/k21academy/qa-or-issues-or-progress-for-deploy-manage-amazon-website-using-devops-tools>

Deploy & Manage Amazon Website using DevOps Tools



Prerequisites:

1. To get started with AWS, we recommend that you [create a free tier account](#) on their website

Resources

[Project Activity Guide](#)

Deepak Sharma
 just now • Troubleshooting

Q/A or Issues or Progress for Deploy & Manage Amazon Website us...

If you encounter any issues or have questions about Deploy & Manage Amazon Website using DevOps Tools, please post them in this thread. Once you have completed the lab,

0
 0

19.1 Build In-Progress/ Build Failure

19.1.1 ISSUE:

```
Dashboard > amazon-pipeline > #4  
  
INFO: Total time: 26.598s  
INFO: Final Memory: 18M/67M  
INFO: -----  
[Pipeline] }  
[Pipeline] // withSonarQubeEnv  
[Pipeline] }  
[Pipeline] // withEnv  
[Pipeline] }  
[Pipeline] // stage  
[Pipeline] stage  
[Pipeline] { (quality gate)  
[Pipeline] tool  
[Pipeline] envVarsForTool  
[Pipeline] tool  
[Pipeline] envVarsForTool  
[Pipeline] withEnv  
[Pipeline] {  
[Pipeline] script  
[Pipeline] {  
[Pipeline] waitForQualityGate  
Checking status of SonarQube task 'AZByIwMwIQq4o52mbofg' on server 'sonar-server'  
SonarQube task 'AZByIwMwIQq4o52mbofg' status is 'IN_PROGRESS'
```

19.1.2 REASON:

This build is in progress or failed because we haven't update the **dockerhub username** in the pipeline script.

19.1.3 RESOLUTION:

Change the **image repository** to your **Docker Hub username** as shown in the screenshot below in the [section 15.1, step 3](#).

```

48     steps {
49         sh "trivy fs . > trivyfs.txt"
50     }
51 }
52 stage("Docker Build & Push"){
53     steps{
54         script{
55             withDockerRegistry(credentialsId: 'docker', toolName: 'docker'){
56                 sh "docker build -t amazon-clone ."
57                 sh "docker tag amazon-clone deepsharma/amazon-clone:latest "
58                 sh "docker push deepsharma/amazon-clone:latest "
59             }
60         }
61     }
62 }
63 stage("TRIVY"){
64     steps{
65         sh "trivy image deepsharma/amazon-clone:latest > trivyimage.txt"
66     }
67 }
68 stage('Deploy to container'){
69     steps{
70         sh 'docker run -d --name amazon-clone -p 3000:3000 deepsharma/amazon-clone:latest'
71     }
72 }
73 }
74 }

```

Insert your dockerhub username

19.2 Could not find 'java' executable in JAVA_HOME or PATH

19.2.1 ISSUE

Jenkins / amazon-pipeline / #2 / Pipeline Overview

Start Tool Install clean workspace Checkout from Git Sonarqube Analysis quality gate Install Dependencies OWASP FS SCAN TRIVY FS SCAN Docker Build & Push

Search

❌ Sonarqube Analysis 0.96s Started 8m 53s ago Jenkins

- ✅ Tool Install 7.9s
- ✅ clean workspace 0.46s
- ✅ Checkout from Git 1.6s
- ❌ Sonarqube Analysis 0.96s
- » quality gate 0.11s
- » Install Dependencies 48ms
- » OWASP FS SCAN 57ms
- » TRIVY FS SCAN 55ms
- » Docker Build & Push 74ms

❌ Use a tool from a predefined Tool Installation `jdk 17` 41ms

✅ Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. 33ms

✅ Use a tool from a predefined Tool Installation `node16` 49ms

✅ Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. 91ms

❌ `$SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=Amazon -Dsonar.projectKey=Amazon` 0.31s

```

0 + /var/lib/jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/sonar-scanner/bin/sonar-scanner -Dsonar.projectName=Amazon
-Dsonar.projectKey=Amazon
1 Could not find 'java' executable in JAVA_HOME or PATH.
2 script returned exit code 1

```

19.2.2 REASON:

- SonarQube Scanner requires Java to run.
- Even though you installed **jdk17** in your pipeline, the **sonar-scanner** step is not picking it up because:
 - **JAVA_HOME** is not set correctly, or
 - **PATH** does not include the **java** binary from JDK 17 during the scanner execution.

19.2.3 Fix:

In **Step 14.1**,

- Name your JDK tool as **jdk17** (exact spelling).
- Select **Adoptium JDK 17.0.8.1+1** version.

14.1 Add JDK & NodeJS

1. Click on **add jdk** and select **installer adoptium.net**
2. Choose **jdk 17.0.8.1+1 version** and in the name section enter **jdk 17**



In **Step 14.2**,

- Add **Sonar Scanner**.
- Name it exactly **sonar-scanner**.
- Select **latest version** from Maven Central.

20 SUMMARY

- Project focuses on deploying an **Amazon clone application**.
- **Terraform** manages infrastructure as code for consistency and automation.
- **Jenkins** handles CI/CD by automating build, test, and deployment.
- Developers push code changes → Jenkins tests → Terraform deploys to cloud.
- Ensures **rapid, reliable, and seamless deployment** for smooth user experience.