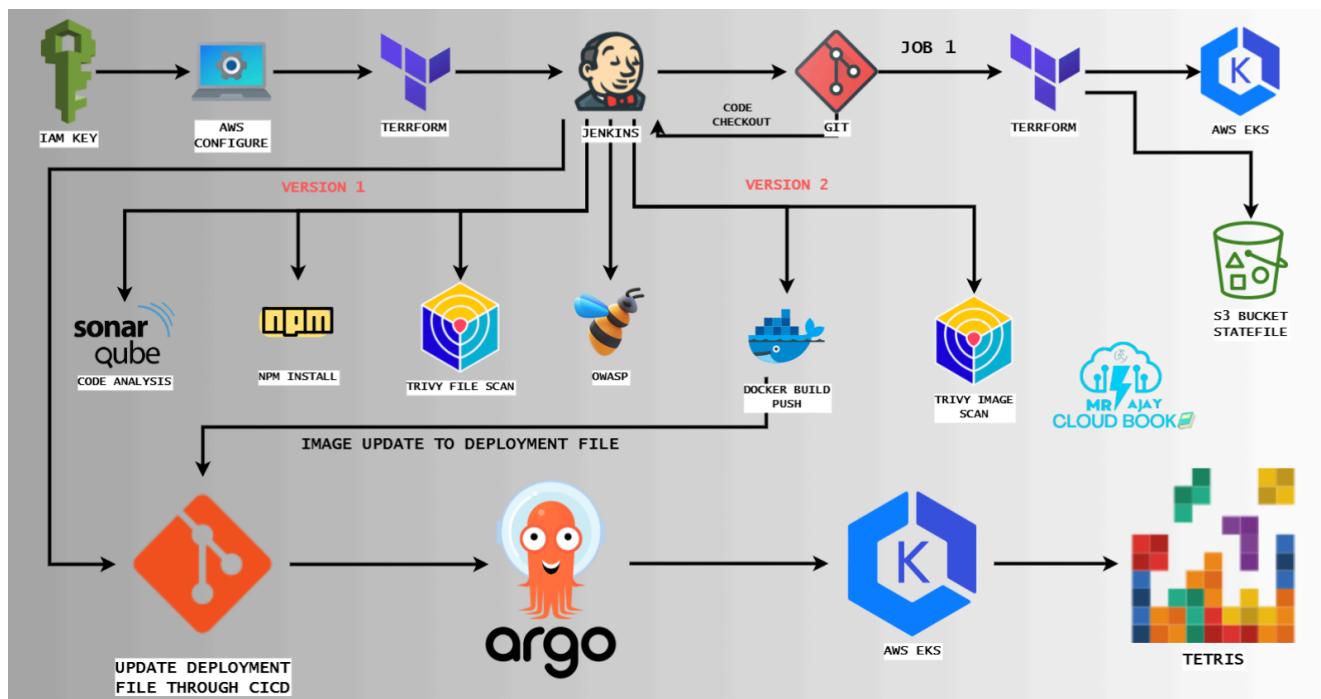


DevOps

Automating Tetris Deployments: DevSecOps with ArgoCD, Terraform, and Jenkins for Two Game Versions

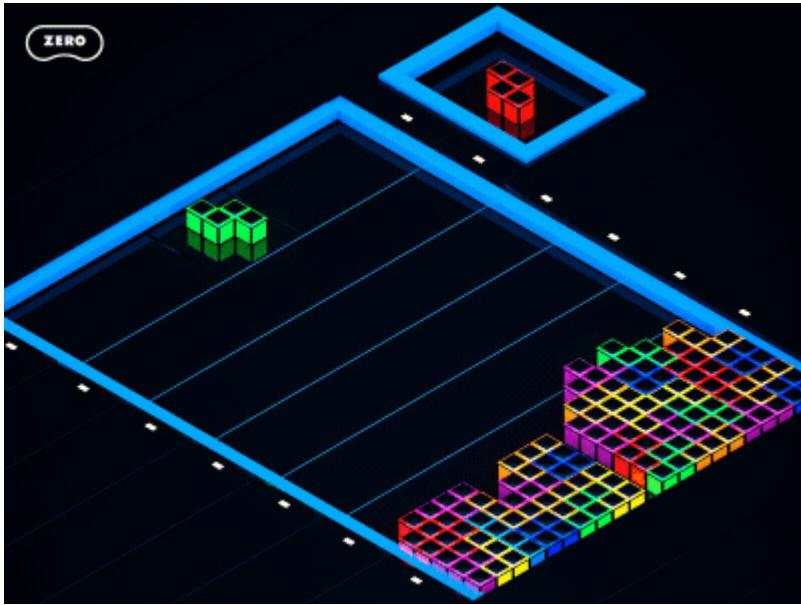


mrcloudbook.com · 8 January 2024



Welcome to the world of DevSecOps automation, where cutting-edge technologies merge to streamline the deployment of one of the most beloved classic games, Tetris. In this blog, we'll embark on a fascinating journey that unveils the power of ArgoCD, Terraform, and Jenkins in orchestrating a seamless and secure deployment pipeline for not one, but two distinct versions of the Tetris game.

Imagine a scenario where your Tetris application effortlessly transitions through various stages of development, from infrastructure provisioning to continuous integration and delivery, all while maintaining the highest standards of security. Join us as we delve into the details of how these sophisticated tools work together to create an automated, efficient, and foolproof DevOps workflow, capable of handling multiple game versions, designed to impress even the most discerning tech enthusiasts.



GitHub REPOSITORIES

TETRIS-VERSION1

<https://github.com/Aj7Ay/Tetris-V1.git>

TETRIS-VERSION2

<https://github.com/Aj7Ay/Tetris-V2.git>

TETRIS_MANIFEST

<https://github.com/Aj7Ay/Tetris-manifest.git>

Prerequisites:

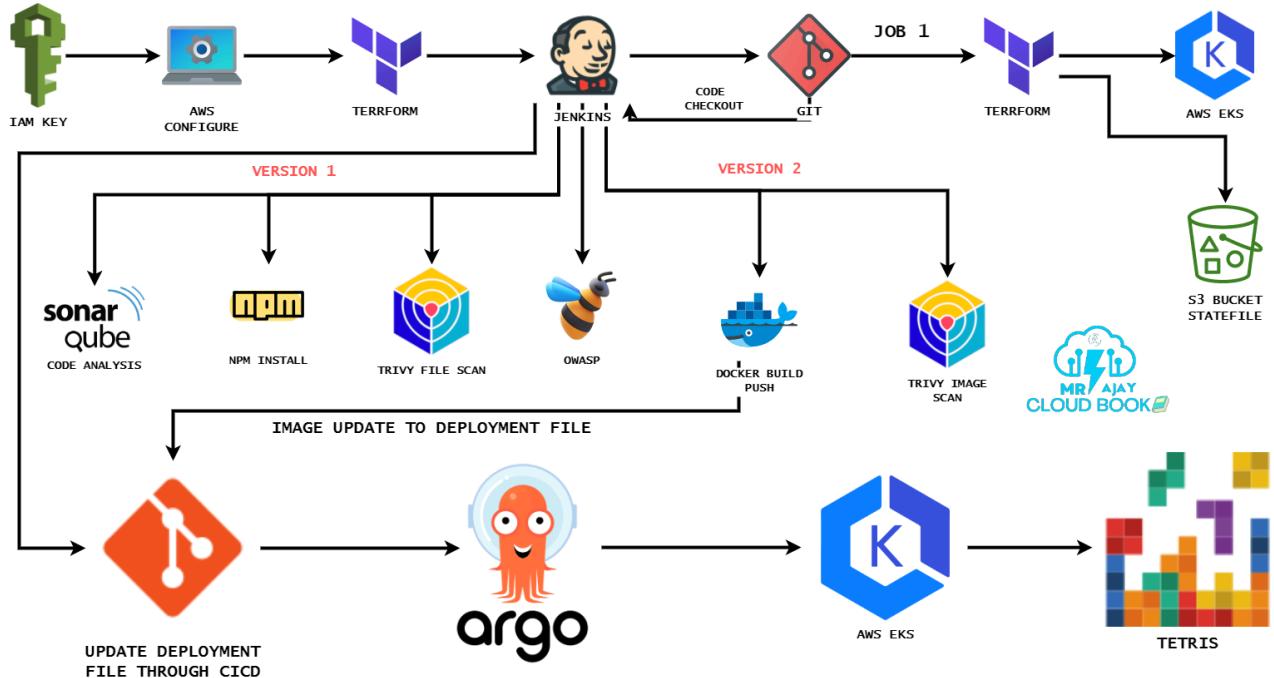
- 1. AWS Account:** To get started, you'll need an active AWS account. Ensure that you have access and permission to create and manage AWS resources.

2. AWS CLI: Install the AWS Command Line Interface (CLI) on your local machine and configure it with your AWS credentials. This is essential for managing your AWS resources.

3. IAM User and Key Pair: Create an IAM (Identity and Access Management) user with the necessary permissions to provision resources on AWS. Additionally, generate an IAM Access Key and Secret Access Key for programmatic access. Ensure that you securely manage these credentials.

4. S3 Bucket: Set up an S3 bucket to store your Terraform state files. This bucket is crucial for maintaining the state of your infrastructure and enabling collaboration.

5. Terraform: Install Terraform on your local machine. Terraform is used for provisioning infrastructure as code and managing AWS resources. Make sure to configure Terraform to work with your AWS credentials and your S3 bucket for state storage.



Contents [hide]

[Step1: How to install and setup Terraform on Windows](#)

[Step2: Download the AWS CLI Installer:](#)

[Step3: create an IAM user](#)

[Step4: Aws Configure](#)

[Step5: Terraform files and Provision](#)

[Configure in Global Tool Configuration](#)

[Configure Sonar Server in Manage Jenkins](#)

[Version 1.0](#)

[ARGO CD SETUP](#)

[Login](#)

[Version 2.0](#)

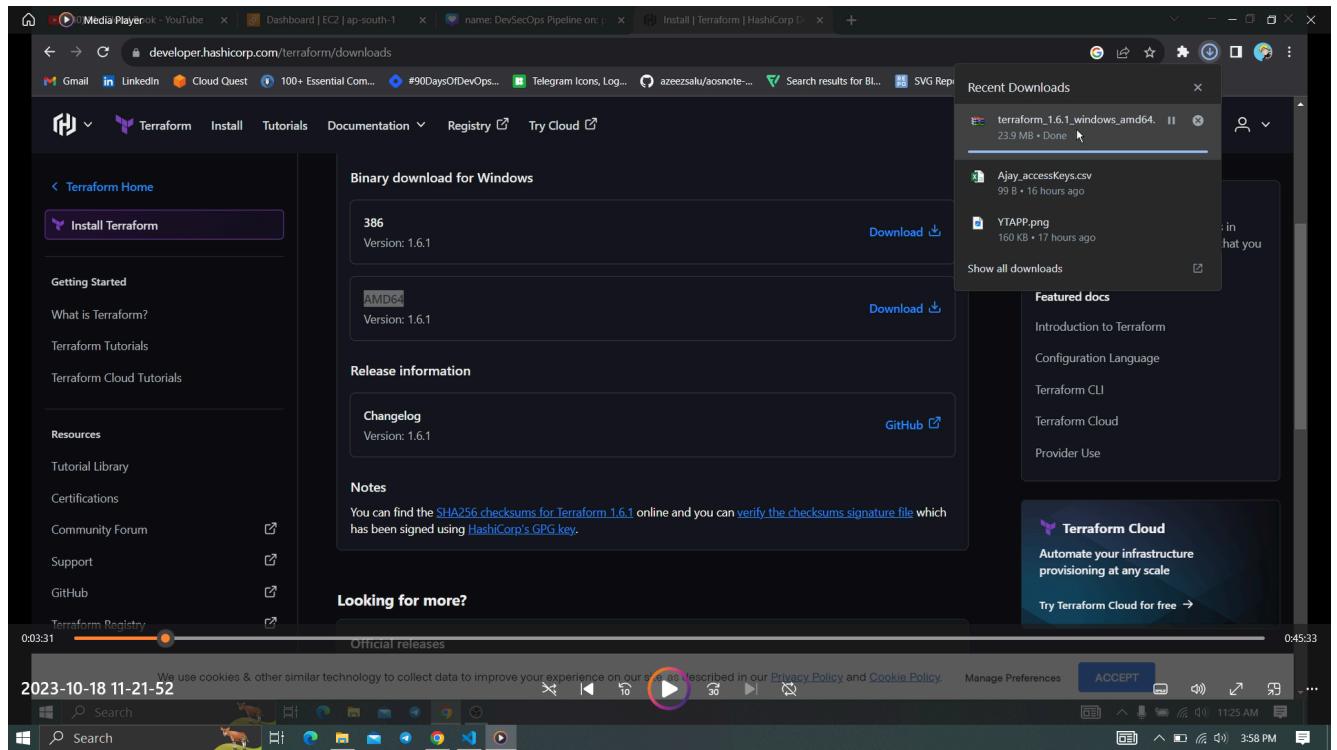
[YOUTUBE VERSION OF SECOND TETRIS USED TRIGGER JOB](#)

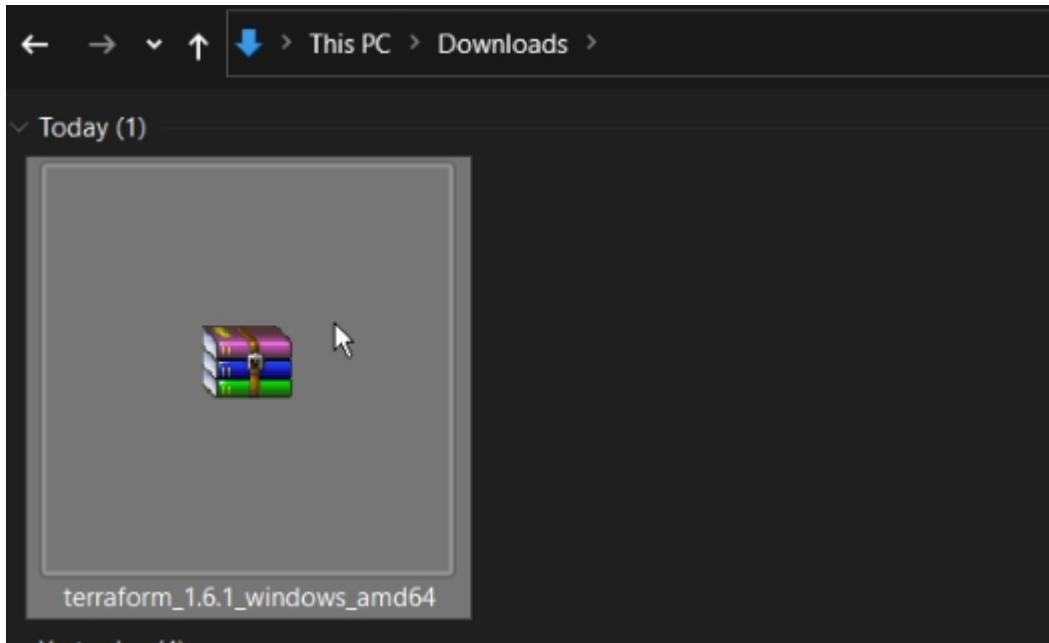
[DESTRUCTION OF RESOURCES](#)

Step1: How to install and setup Terraform on Windows

Download Terraform:

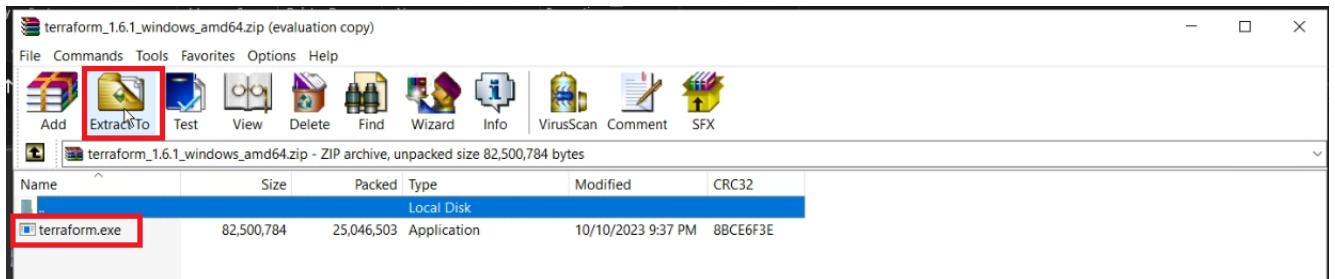
Visit the official Terraform website: terraform.io/downloads.html



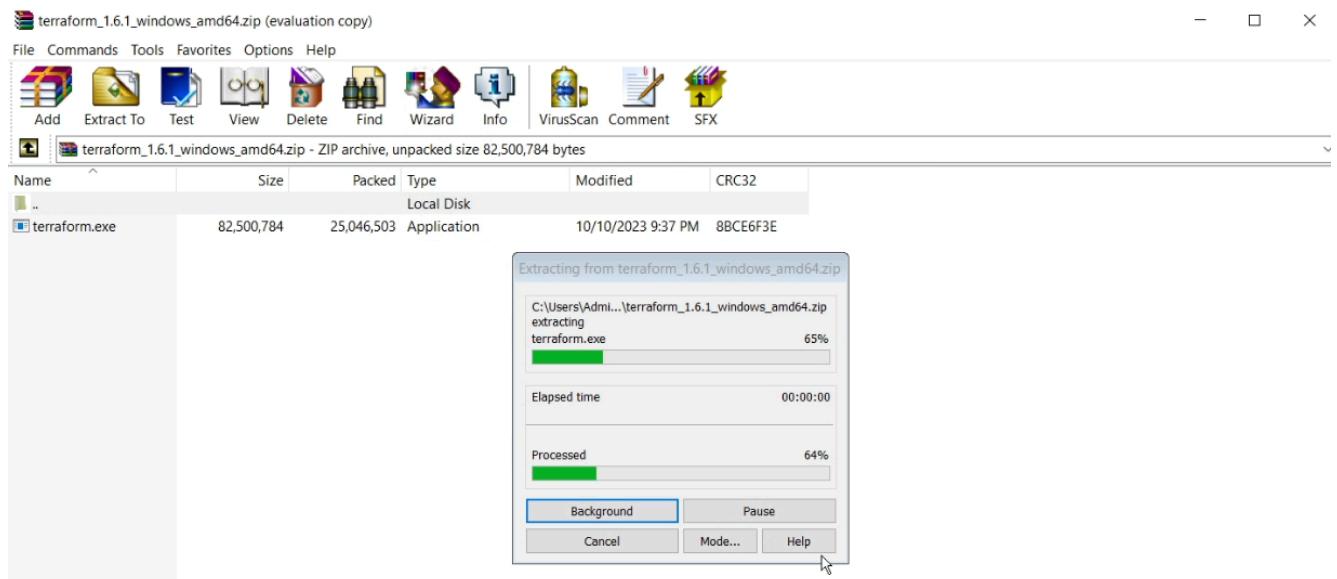
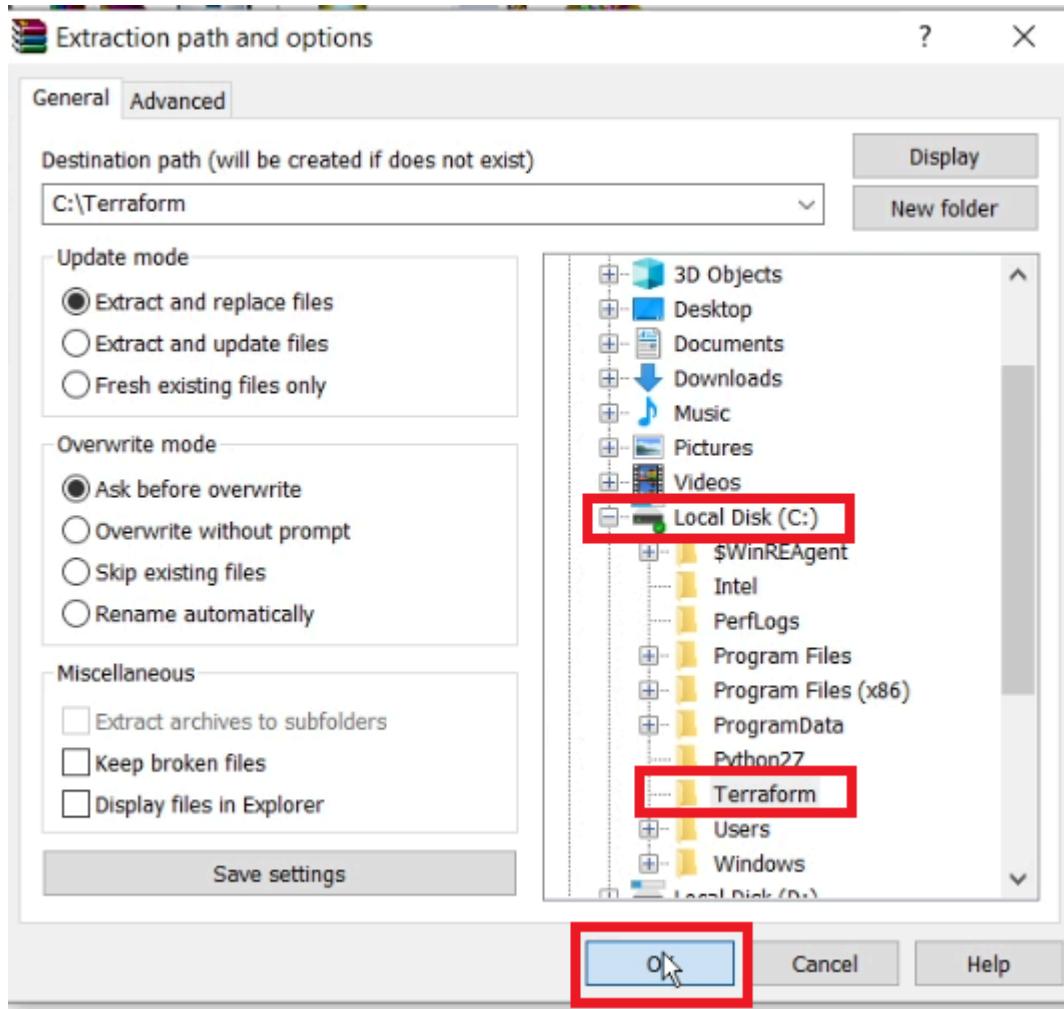


Extract the ZIP Archive:

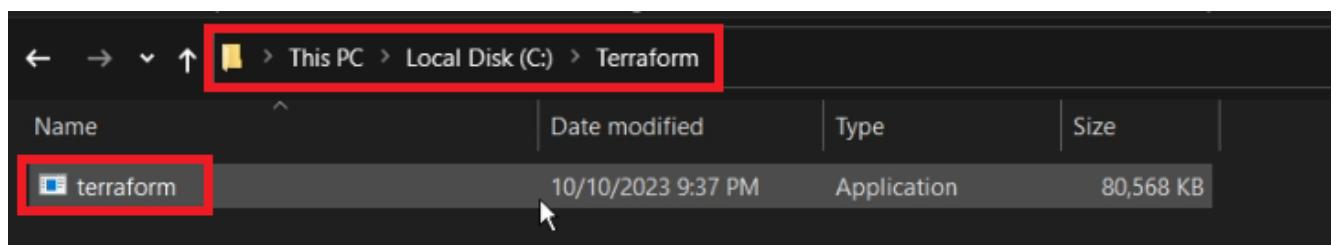
Once the download is complete, extract the contents of the ZIP archive to a directory on your computer. You can use a tool like 7-Zip or the built-in Windows extraction tool. Ensure that you extract it to a directory that's part of your system's PATH.



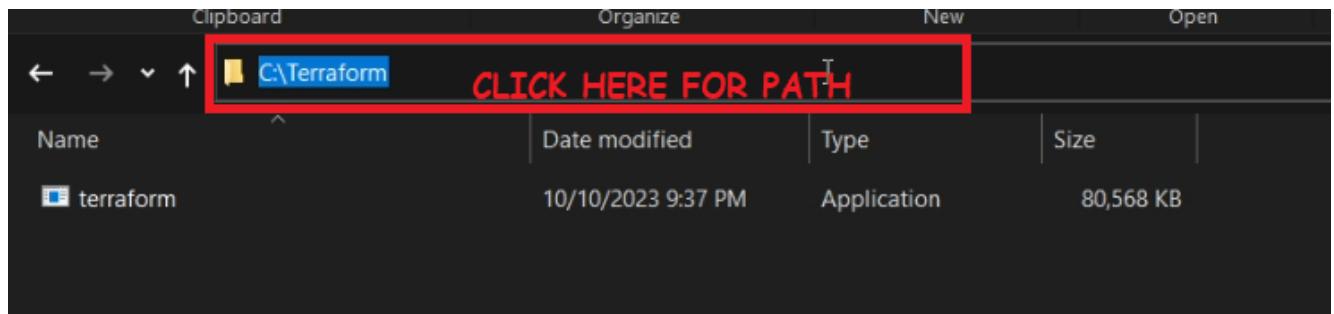
Remember that I created a Terraform Directory in C drive



Extracted to C drive



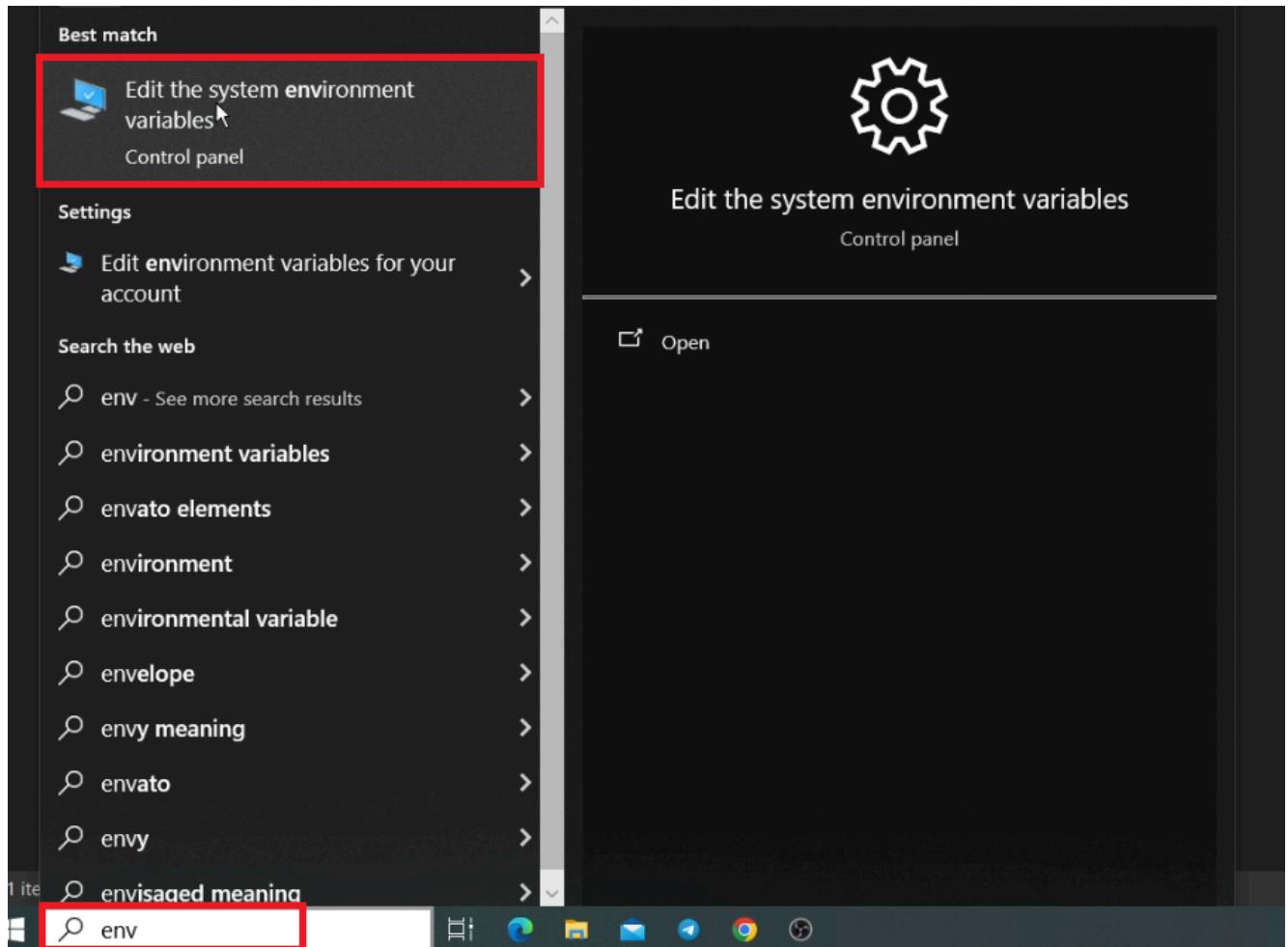
Copy the path



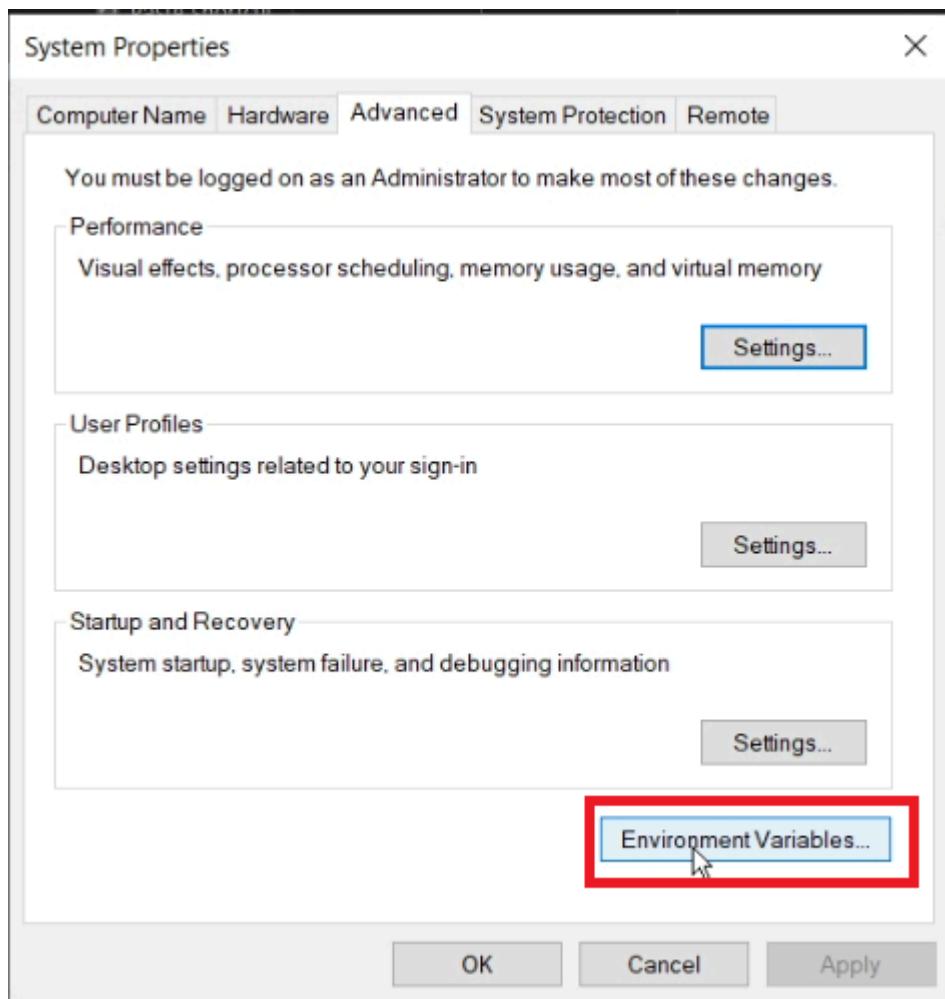
Add Terraform to Your System's PATH:

To make Terraform easily accessible from the command prompt, add the directory where Terraform is extracted to your system's PATH environment variable. Follow these steps:

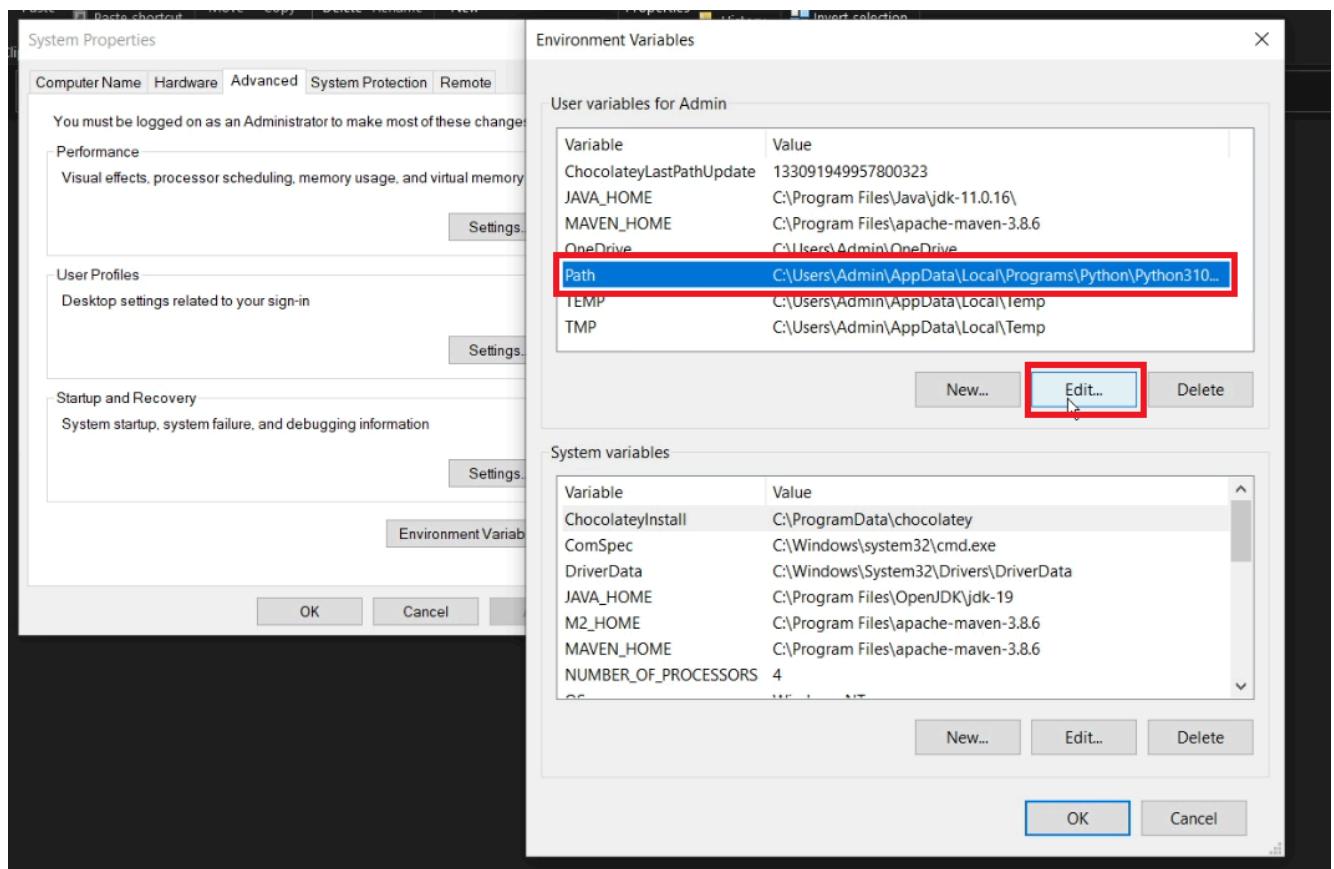
Search for “Environment Variables” in your Windows search bar and click “Edit the system environment variables.”



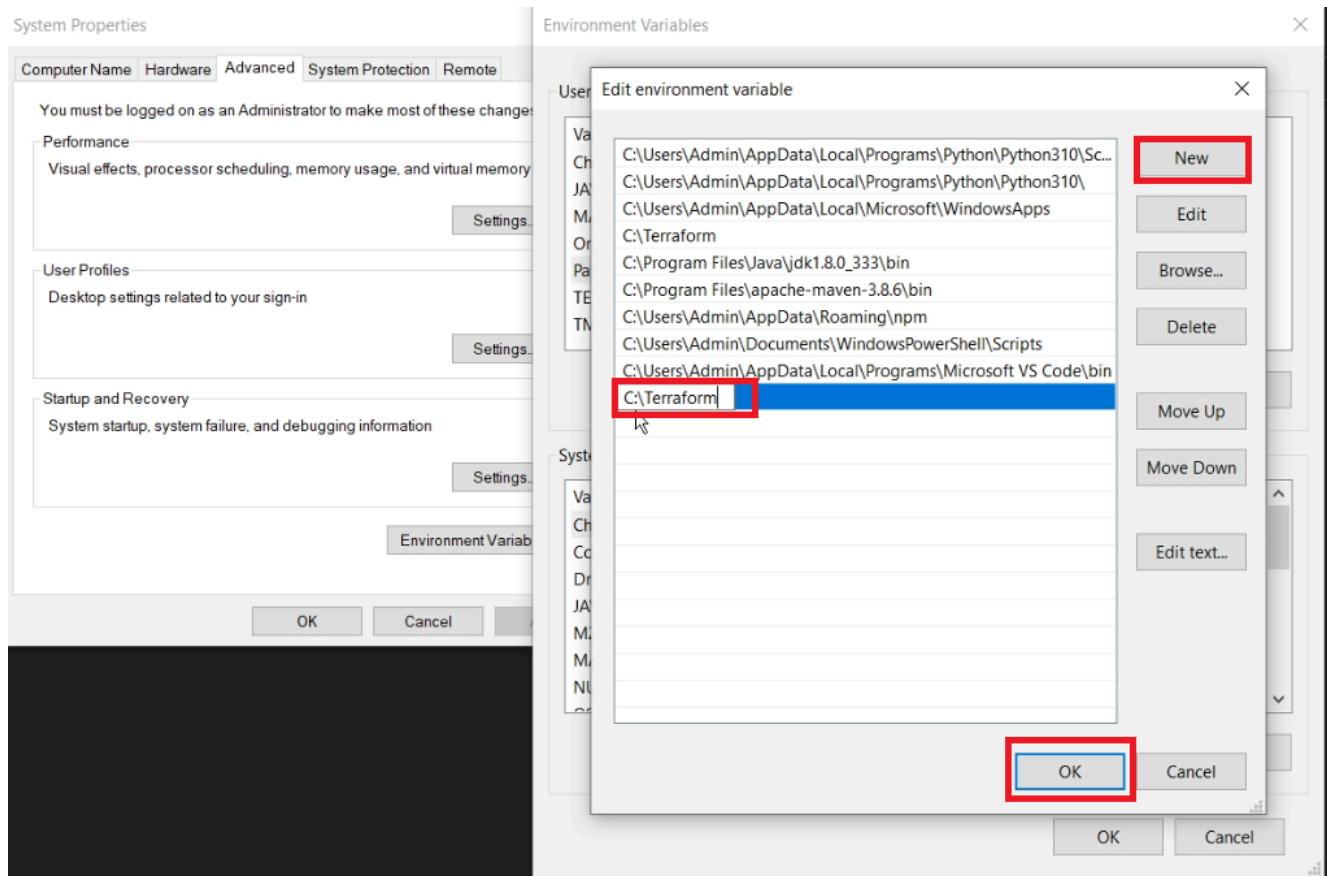
In the “System Properties” window, click the “Environment Variables” button.



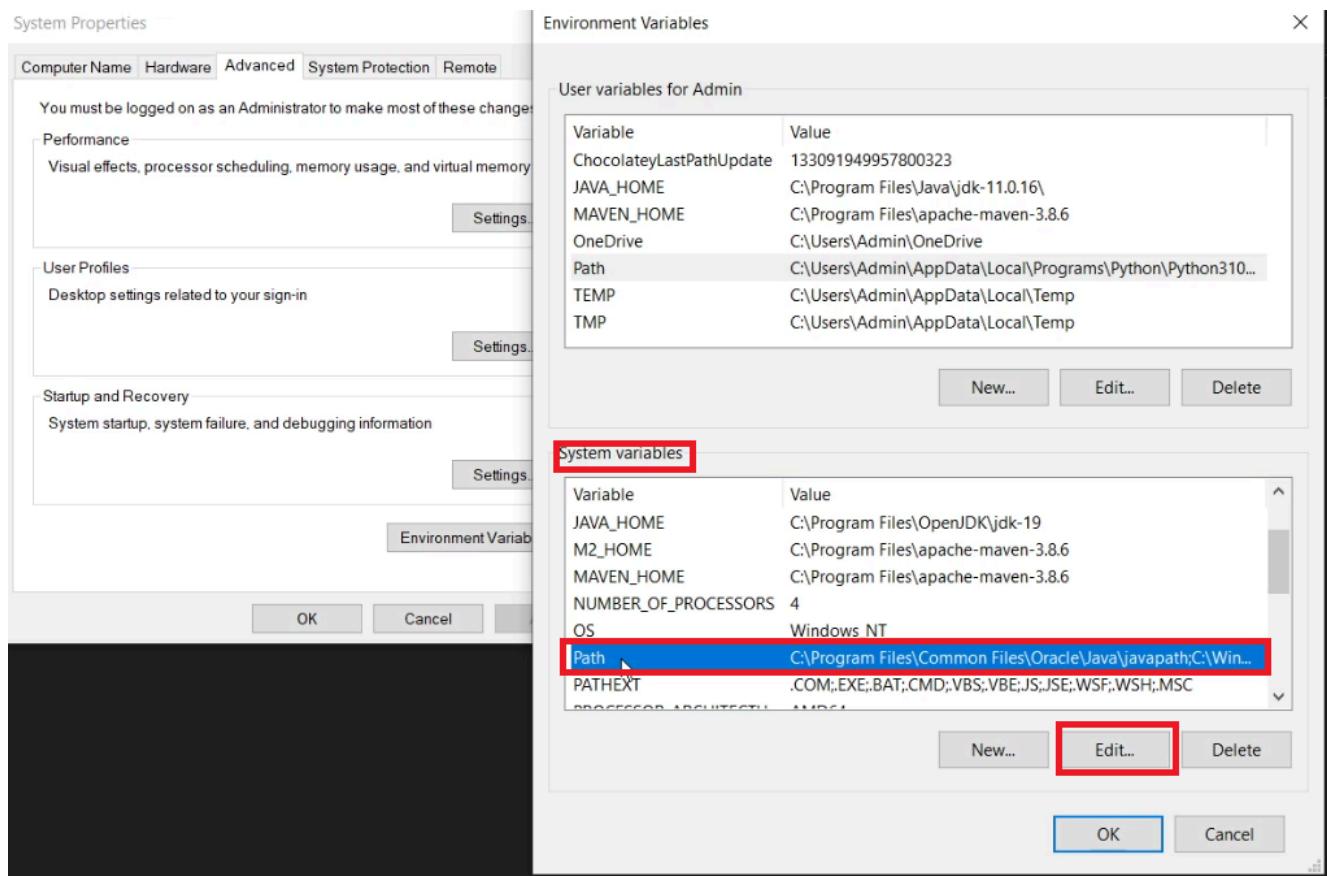
Under “User variables for Admin,” find the “Path” variable and click “Edit.”



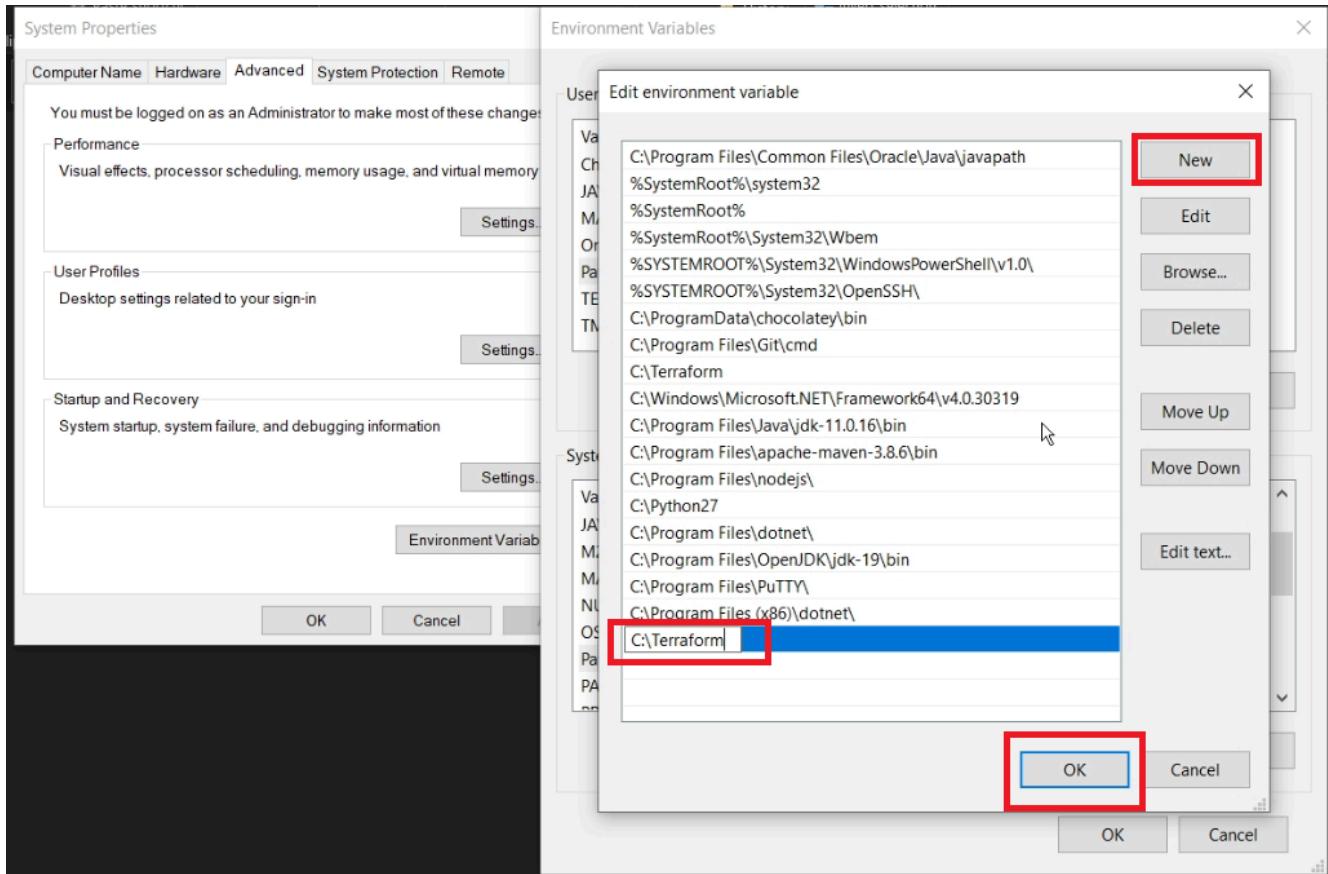
Click on New paste the copied path and click on OK



Under "System variables," find the "Path" variable and click "Edit."



Click “New” and add the path to the directory where you extracted Terraform (e.g., C:\path\to\terraform).



Click “OK” to close the Environment Variables windows.

Click “OK” again to close the System Properties window.

Verify the Installation:

Open a new Command Prompt or PowerShell window.

Type `terraform --version` and press Enter. This command should display the Terraform version, confirming that Terraform is installed and in your PATH.

```
Select Command Prompt
Microsoft Windows [Version 10.0.19045.3570]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>terraform --version
Terraform v1.6.1
on windows_amd64

C:\Users\Admin>
```

Your Terraform installation is now complete, and you can start using Terraform to manage your infrastructure as code.

Step2: Download the AWS CLI Installer:

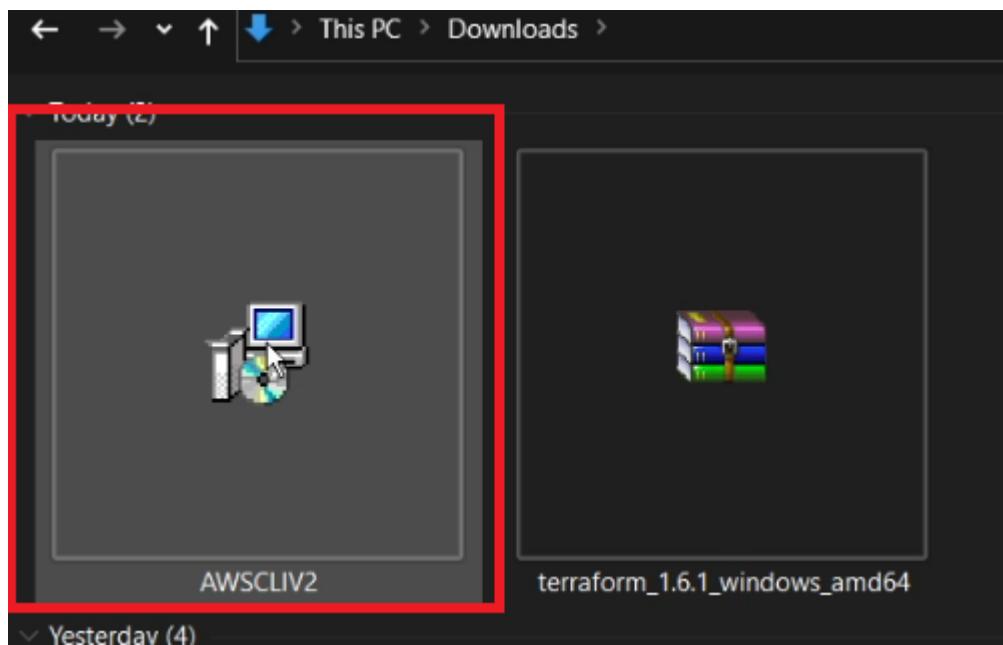
Visit the AWS CLI Downloads page: aws.amazon.com/cli/

Under “Install the AWS CLI,” click on the “64-bit” link to download the AWS CLI installer for Windows.

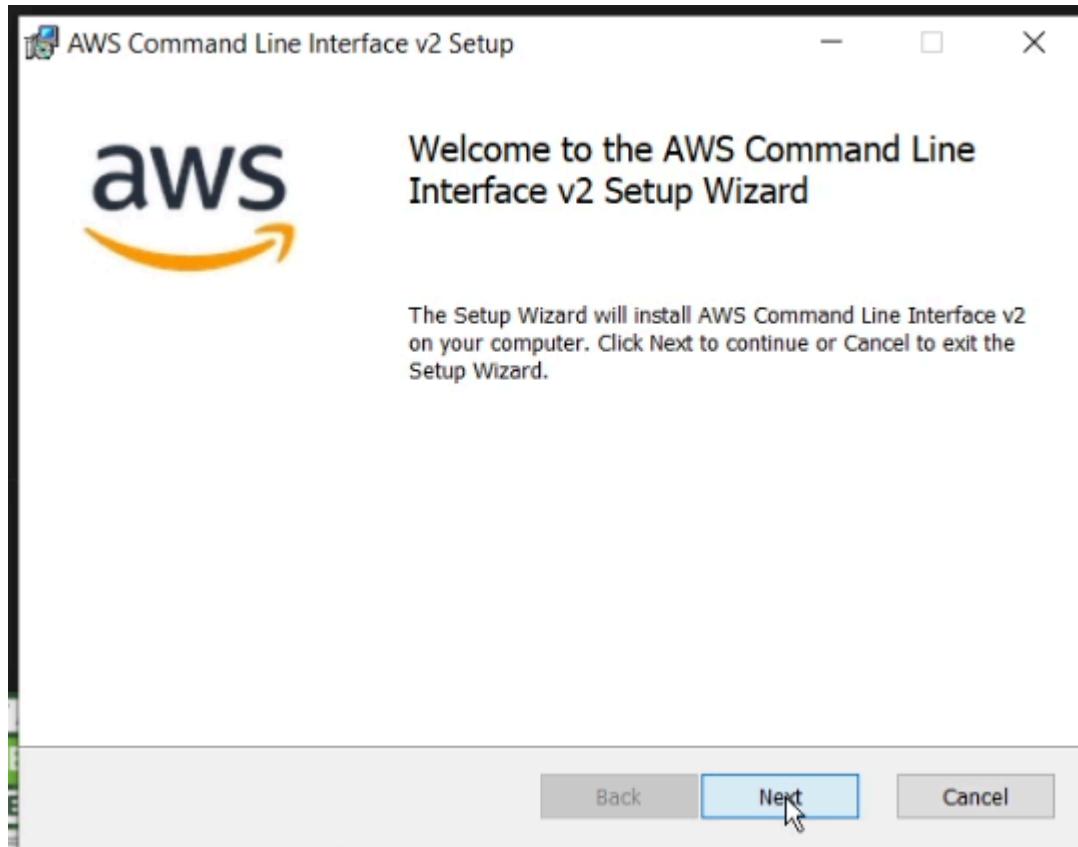
The screenshot shows a web browser displaying the AWS CLI Downloads page at aws.amazon.com/cli/. The URL bar has a red box around it. The main content area is titled "AWS Command Line Interface". On the right, there are sections for "Windows", "MacOS", and "Linux". The "Windows" section contains the text: "Download and run the 64-bit Windows installer." A red box highlights this text. Below the sections, there is a note: "The AWS CLI v2 offers several new features including improved installers, new configuration options such as AWS IAM Identity Center (successor to AWS SSO), and various interactive features." To the left, a sidebar lists "AWS Command Line Interface" under "RESOURCES" and "Documentation", "Tools", and "Release Notes" under "RELATED LINKS".

Run the Installer:

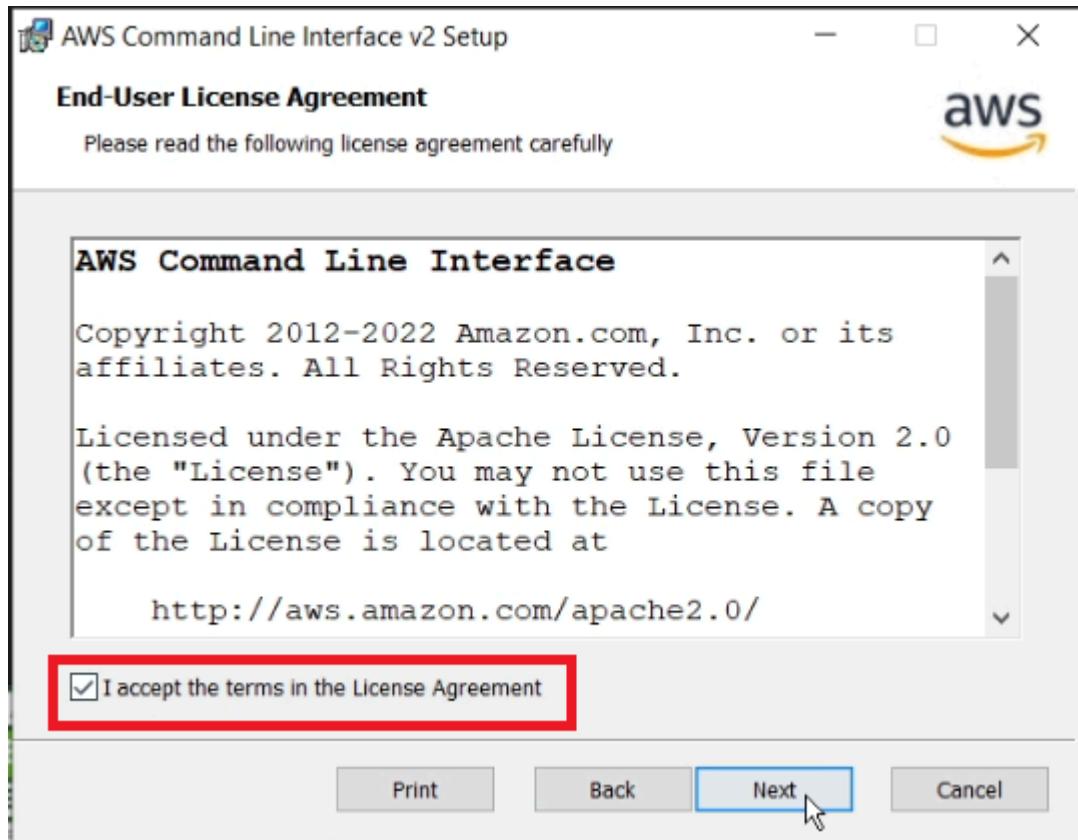
Locate the downloaded installer executable (e.g., AWSCLIV2.exe) and double-click it to run the installer.



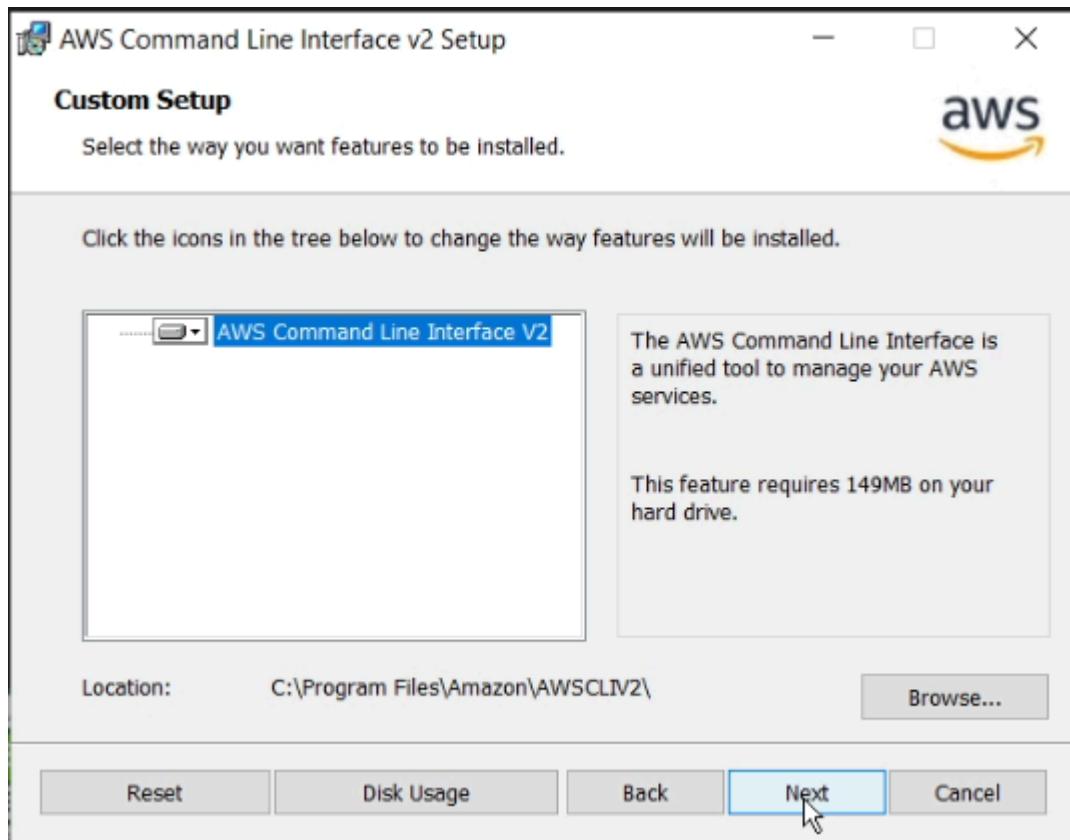
Click on Next



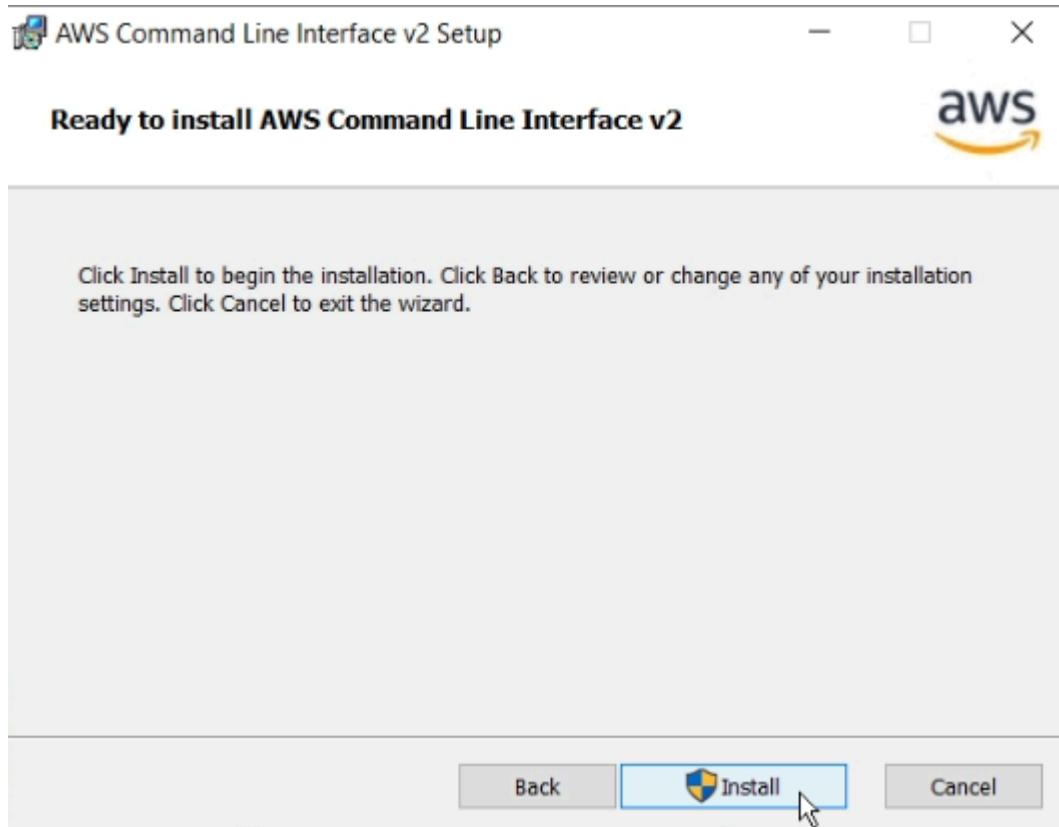
Agree to the terms and click on Next



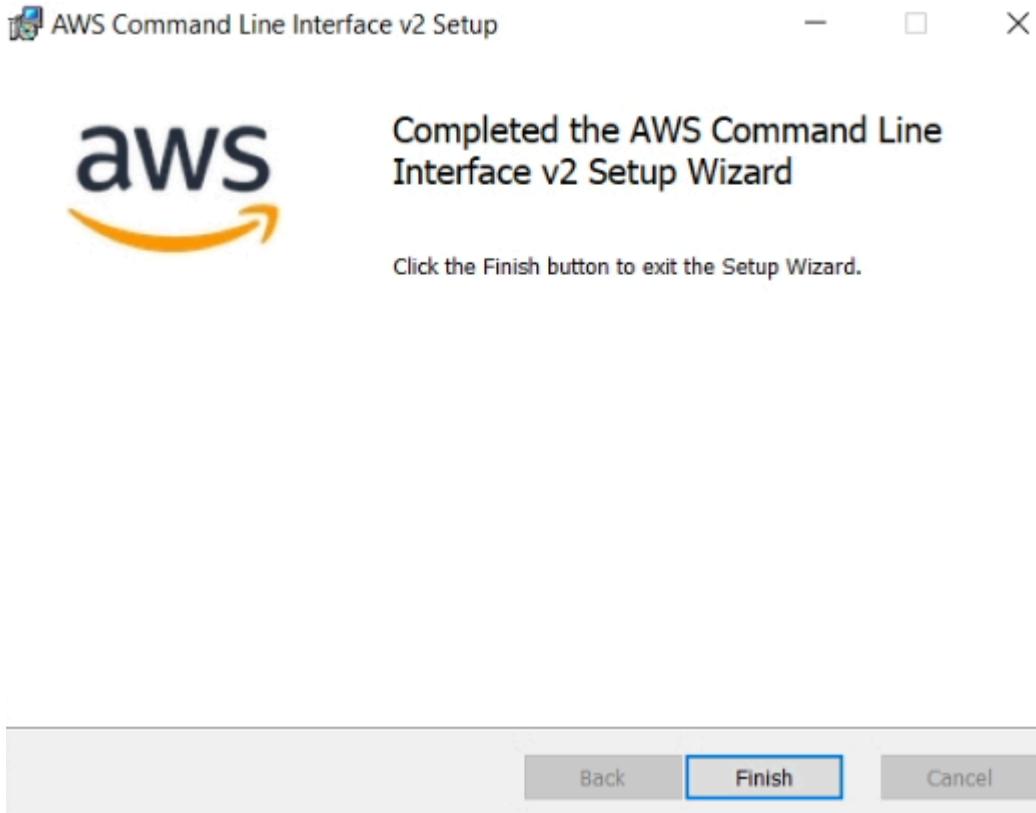
Click Next



Click on install



Click Finish Aws cli is installed



Verify the Installation:

Open a Command Prompt or PowerShell window.

Type **aws --version** and press Enter. This command should display the AWS CLI version, confirming that the installation was successful.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.3570]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Downloads>aws --version
aws-cli/2.13.26 Python/3.11.6 Windows/10 exe/AMD64 prompt/off

C:\Users\Admin\Downloads>
```

Step3: create an IAM user

Navigate to the **AWS console**

Click the “Search” field.

The screenshot shows the AWS Console Home page. At the top, there are links for Services, Search, and a region selector (Hyderabad). Below the search bar, there are three main cards:

- Recently visited** (Info): Lists services: EC2, S3, Elastic Container Service, Elastic Container Registry, VPC, IAM, and RDS. A "View all services" link is at the bottom.
- Welcome to AWS**: Includes sections for "Getting started with AWS" (with a rocket icon), "Training and certification" (with a graduation cap icon), and "What's new with AWS?" (with a lightbulb icon).
- AWS Health** (Info): Shows 0 open issues (Past 7 days), 0 scheduled changes (Upcoming and past 7 days), and 0 other notifications (Past 7 days). A "Go to AWS Health" link is at the bottom.

At the bottom left, there are "Feedback" and "Language" buttons. The footer includes links for Feedback, Language, Privacy, Terms, and a copyright notice: "© 2023, Amazon Web Services India Private Limited or its affiliates.".

Search for IAM

The screenshot shows the AWS Services search results for 'iam'. The search bar at the top contains 'Q iam'. The sidebar on the left lists categories: Services (8), Features (19), Resources (New), Blogs (1,506), Documentation (118,748), Knowledge Articles (30), Tutorials (2), Events (13), and Marketplace (412). The main area displays the search results for 'iam'.

Services

- IAM** (Manage access to AWS resources)
- IAM Identity Center (successor to AWS Single Sign-On)** (Manage workforce user access to multiple AWS accounts and cloud app)
- Resource Access Manager** (Share AWS resources with other accounts or AWS Organizations)

Click “Users”

Management (IAM)

Search IAM

[Dashboard](#)

▼ Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

▼ Access reports

Access analyzer

Archive rules

IAM dashboard

Security recommendations 1

✓ Root user has MFA

Having multi-factor authentication (MFA) for the root user improves security for this account.

✓ Root user has no active access keys

Using access keys attached to an IAM user instead of the root user improves security.

⚠ Update your access permissions for AWS Billing, Cost Management, and Account consoles

We are replacing the following IAM actions for Billing, Cost Management, and Account console granular IAM actions: aws-portal:ViewBilling, aws-portal:ModifyBilling, aws-portal:ViewAccount portal:ModifyAccount, aws-portal:ViewPaymentMethods, aws-portal:ModifyPaymentMethods, :portal:ViewUsage, purchase-orders:ViewPurchaseOrders, and purchase-orders:ModifyPurchase To ensure you don't lose access to AWS Billing, Cost Management, and Account console base update your existing IAM policies to include the new IAM actions before July 2023. Examples impacted include AWS Cost Explorer, AWS Budgets, Billing console, and more. For more information please visit [blog](#).

IAM resources

Click “Add users”

The screenshot shows the AWS IAM Users page. At the top, there is a search bar with the placeholder text "[+S]" and several navigation icons. Below the header, a message states: "What is used to interact with AWS in an account." On the right side of the header, there are buttons for "Global" and "Aj yegireddi". The main area displays a table of users. The first user listed is "e" (Email: e@example.com). The table includes columns for "Last activity", "MFA", "Password age", and "Active key age". The "Add users" button is located at the top right of the user list, and it is also highlighted with an orange circle. The table has a header row with dropdown arrows for sorting.

Click the “User name” field.

[Create user](#)

Specify user details

User details

User name

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

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.

 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](#)

Type “Terraform” or as you wish about the name

Click Next

acters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

nt Console - *optional*

best practice to manage their access in IAM Identity Center.

through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate [Learn more](#)

Cancel

Next

Click “Attach policies directly”



. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

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.

Each to the group. We recommend using groups to manage user permissions by job permissions. [Learn more](#)

Create group

Click this checkbox with Administrator access

Permissions policies (1046)
Choose one or more policies to attach to your new user.

Filter distributions by text, property or value

<input type="checkbox"/>	Policy name	Type
<input type="checkbox"/>	AccessAnalyzerServiceRolePolicy	AWS managed
<input checked="" type="checkbox"/>	AdministratorAccess	AWS managed
<input type="checkbox"/>	AdministratorAccess-Amplify	AWS managed
<input type="checkbox"/>	AdministratorAccess-AWSElasticBe...	AWS managed
<input type="checkbox"/>	AlexaForBusinessDeviceSetup	AWS managed
<input type="checkbox"/>	AlexaForBusinessFullAccess	AWS managed
<input type="checkbox"/>	AlexaForBusinessGatewayExecution	AWS managed

Click "Next"

loud...	AWS managed	0
	AWS managed	0
:cess	AWS managed	0
s	AWS managed	0
is	AWS managed	0
rAccess	AWS managed	0
:cess	AWS managed	0

Assign permissions for this user. Use this advanced feature used to delegate permission management to others. [Learn more](#)

[Cancel](#)

[Previous](#)

[Next](#)

© 2023, Amazon Web Services India Private Limited or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Click “Create user”

Type	Used as
AWS managed - job function	Permissions policy

to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

[Cancel](#)

[Previous](#)

[Create user](#)

© 2023, Amazon Web Services India Private Limited or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Click newly created user in my case Ajay

IAM > Users

Users (2) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS.

User name	Groups	Last active
Ajay	None	Never
mr-cloud-book	None	22 hours

Access management

- User groups
- Users**
- Roles
- Policies
- Identity providers
- Account settings

Access reports

- Access analyzer
- Archive rules
- Analyzers
- Settings

Click “Security credentials”

Summary

ARN arn:aws:iam::677356083070:user/Ajay	Console access Disabled	Access key 1 Not enabled
Created February 28, 2023, 14:41 (UTC+05:30)	Last console sign-in -	Access key 2 Not enabled

Permissions **Groups** **Tags** **Security credentials** **Access Advisor**

Permissions policies (1)

Permissions are defined by policies attached to the user directly or through groups.

Policy name	Type	Attach
AdministratorAccess	AWS managed - job function	Directly

Click “Create access key”

[Dashboard](#)[Access management](#)[User groups](#)[Users](#)[Roles](#)[Policies](#)[Identity providers](#)[Account settings](#)[Access reports](#)[Access analyzer](#)[Archive rules](#)[Analyzers](#)[Settings](#)[Credential report](#)

No MFA devices. Assign an MFA device to impr

[Assign MFA](#)

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AW inactive) at a time. [Learn more](#)

[Create access key](#)**No access keys**

As a best practice, avoid using long-term credentials like access keys. Ins

[Create access key](#)

Click this radio button with the CLI

The screenshot shows the AWS IAM 'Create access key' wizard. The top navigation bar includes the AWS logo, Services, a search bar, and a keyboard shortcut [Alt+S]. The breadcrumb trail shows: IAM > Users > Ajay > Create access key.

Step 1: Access key best practices & alternatives

A section titled "Access key best practices & alternatives" with the sub-instruction "Avoid using long-term credentials like access keys to improve your security. Cor".

Step 2 - optional: Set description tag

Step 3: Retrieve access keys

Access key best practices & alternatives

A section titled "Access key best practices & alternatives" with the sub-instruction "Avoid using long-term credentials like access keys to improve your security. Cor".

Command Line Interface (CLI)

Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

Local code

Local code
You plan to use this access key to enable application code in a local development access your AWS account.

Application running on an AWS compute service

Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS con Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

Agree to terms

Application running outside AWS
You plan to use this access key to enable an application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.

Other
Your use case is not listed here.

 **Alternatives recommended**

- Use [AWS CloudShell](#), a browser-based CLI, to run commands. [Learn more](#)
- Use the [AWS CLI V2](#) and enable authentication through a user in IAM Identity Center. [Learn more](#)

I understand the above recommendation and want to proceed to create an access key.



Feedback

Language

Click Next

Application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.

Use AWS CloudShell, a browser-based CLI, to run commands. [Learn more](#)

Enable authentication through a user in IAM Identity Center. [Learn more](#)

I understand the above recommendation and want to proceed to create an access key.

[Cancel](#) [Next](#)

© 2023, Amazon Web Services India Private Limited or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Click “Create access key”

onal

ed to this user as a tag and shown alongside the access key.

A good description will help you rotate this access key confidently later.

ers, numbers, spaces representable in UTF-8, and: _ . : / = + - @

Cancel**Previous****Create access key**

Download .csv file

Retrieve access keys

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key**Secret access key**
 AKIAZ3NMRS57HAYY5GSX

 **** Show

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [Best practices for managing AWS access keys](#).

Download .csv file**Done**

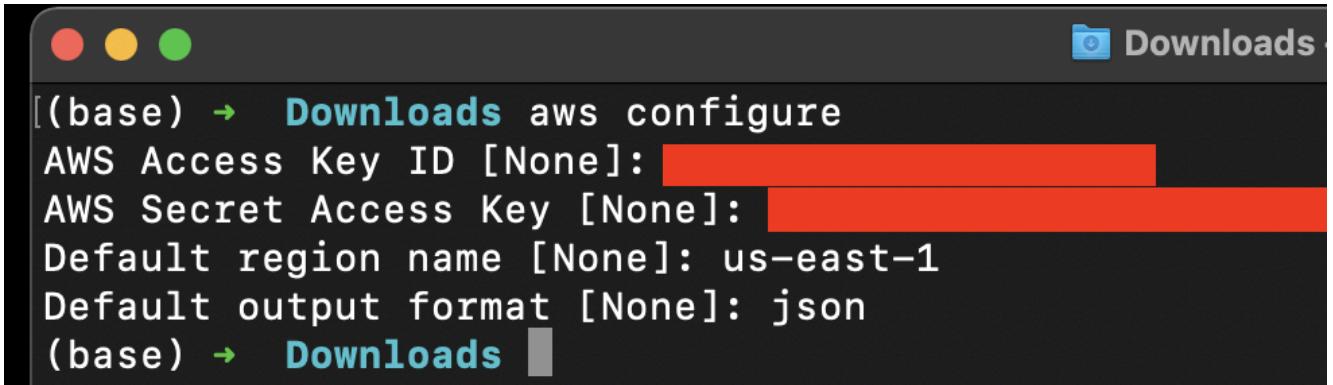
Step4: Aws Configure

Go to vs code or Cmd your wish



```
aws configure
```

Provide your Aws Access key and Secret Access key



```
(base) → Downloads aws configure
AWS Access Key ID [None]: [REDACTED]
AWS Secret Access Key [None]: [REDACTED]
Default region name [None]: us-east-1
Default output format [None]: json
(base) → Downloads
```

Step5: Terraform files and Provision

CHANGE YOUR S3 BUCKET NAME IN THE BACKEND.TF

Clone the repository to your local file explorer

```
https://github.com/Aj7Ay/Tetris-V1.git
cd Tetris-V1
cd Jenkins-terraform
ls #to see files
```

```
PS C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform> ls

Directory: C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform

Mode                LastWriteTime       Length Name
----              ->----          ----- 
-a----  10/26/2023 10:34 AM           206 backend.tf
-a----  10/26/2023 9:09 AM          2382 install_jenkins.sh
-a----  10/26/2023 10:32 AM         1849 Main.tf
-a----  10/25/2023 6:18 PM           202 provider.tf

PS C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform>
```

Now you can see the terraform files to provision the AWS Ec2 with Jenkins installed,sonarqube container and Trivy, Aws cli,Kubectl and terraform

Shell script



```
#!/bin/bash
sudo apt update -y
wget -O - https://packages.adoptium.net/artifactory/api/gpg/key/public
echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/artifactory/api/debian-stable/jenkins.io-2023.key" | sudo tee /etc/apt/sources.list.d/jenkins.list &> /dev/null
sudo apt update -y
sudo apt install temurin-17-jdk -y
/usr/bin/java --version
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo gpg --dearmor -o /usr/share/keyrings/jenkins-keyring.gpg
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.gpg] https://pkg.jenkins.io/debian-stable/ jenkins.io-2023 main > /etc/apt/sources.list.d/jenkins.list &> /dev/null
sudo apt-get update -y
sudo apt-get install jenkins -y
sudo systemctl start jenkins
sudo systemctl status jenkins

#install docker
sudo apt-get update
sudo apt-get install docker.io -y
sudo usermod -aG docker ubuntu
newgrp docker
sudo chmod 777 /var/run/docker.sock
docker run -d --name sonar -p 9000:9000 sonarqube:lts-community

# install trivy
sudo apt-get install wget apt-transport-https gnupg lsb-release -y
```

```
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor > /usr/share/keyrings/trivy.gpg
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb/ stable main" | sudo tee /etc/apt/sources.list.d/trivy.list
sudo apt-get update
sudo apt-get install trivy -y

#install terraform
sudo apt install wget -y
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
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
sudo apt update && sudo apt install terraform

#install Kubectl on Jenkins
sudo apt update
sudo apt install curl -y
curl -LO https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable) | sudo tee /usr/local/bin/kubectl >> /dev/null
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
kubectl version --client

#install Aws cli
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
sudo apt-get install unzip -y
unzip awscliv2.zip
sudo ./aws/install
```

Now inside the Jenkins-terraform directory

Open the terminal and provide the below commands

```
terraform init
```



```
PS C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform> terraform init
```

Initializing the backend...

Successfully configured the backend "s3"! Terraform will automatically use this backend unless the backend configuration changes.

Initializing provider plugins...

- Finding hashicorp/aws versions matching "~> 5.0"...
- Installing hashicorp/aws v5.22.0...

Now validate the code to see any issues in Terraform files



```
terraform validate
```



```
PS C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform> terraform validate  
Success! The configuration is valid.
```

```
PS C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform>
```

Now see the plan for resources



```
terraform plan
```



```
PS C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform> terraform plan
```

After the plan Apply to provision resources



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

```
PS C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform> terraform apply --auto-approve
```

```
Plan: 5 to add, 0 to change, 0 to destroy.

aws_iam_role.example_role: Creating...
aws_security_group.Jenkins-sg: Creating...
aws_iam_role.example_role: Creation complete after 1s [id=Jenkins-terraform]
aws_iam_role_policy_attachment.example_attachment: Creating...
aws_iam_instance_profile.example_profile: Creating...
aws_security_group.Jenkins-sg: Creation complete after 2s [id=sg-086b7a68255d23e19]
aws_iam_role_policy_attachment.example_attachment: Creation complete after 1s [id=Jenkins-terraform-20231026050902014700000001]
aws_iam_instance_profile.example_profile: Creation complete after 2s [id=Jenkins-terraform]
aws_instance.web: Creating...
aws_instance.web: Still creating... [10s elapsed]
aws_instance.web: Still creating... [20s elapsed]
aws_instance.web: Still creating... [30s elapsed]
aws_instance.web: Creation complete after 32s [id=i-0323f37f837248e53]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.
```

Apply completed you can see Ec2 is created in the Aws console

Instances (1) Info		C	Connect	Instance state ▾	Actions ▾	Launch instances ▾
Find Instance by attribute or tag (case-sensitive)						
Instance state = running X		Clear filters				
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Jenkins-ARGO	i-0323f37f837248e53	Running Q Q	t2.large	Initializing	No alarms +	ap-south-1b ec2

Now copy the public IP address of ec2 and paste it into the browser

```
<Ec2-ip:8080> #you will Jenkins login page
```

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

Continue

Connect your Instance to Putty or Mobaxtreme and provide the below command for the Administrator password



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



```
ubuntu@ip-172-31-33-57:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
0ed1cb07ea7447c5a47d723022e74968
ubuntu@ip-172-31-33-57:~$ █
```

Now, install the suggested plugins.

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins will now get installed and install all the libraries.

Create an admin user

Getting Started

Create First Admin User

Username

Password

Confirm password

Full name

E-mail address

Jenkins 2.414.1

Skip and continue as admin

Save and Continue

Click on save and continue.

Jenkins Dashboard

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Build Queue ▾

No builds in the queue.

Build Executor Status ▾

1 Idle
2 Idle

Create a job →

Set up a distributed build

→ Set up an agent
→ Configure a cloud
→ Learn more about distributed builds

Now Copy the public IP again and paste it into a new tab in the browser with 9000



<ec2-ip:9000> #runs sonar container



Instances | EC2 | ap-south-1 | petstore Config [Jenkins] | SonarQube

52.66.140.95:9000/sessions/new?return_to=%2F

Login

Password

Log in Cancel

Enter username and password, click on login and change password



```
username admin
password admin
```

Instances | EC2 | ap-south-1 | petstore Config [Jenkins] | SonarQube | +

6.140.95.9000/account/reset_password

Web Servic... Coaching on DevO... LinkedIn Docker — A Beginn... Cloud Quest 100+ Essential Com... Advanced End-to-E...

Update your password

This account should not use the default password.

Enter a new password

All fields marked with * are required

Old Password *

New Password *

Confirm Password *

Update

Update New password, This is Sonar Dashboard.

Not secure | 52.66.140.95:9000/projects/create

Gmail YouTube Amazon Web Service... Coaching on DevO... LinkedIn Docker — A Beginn... Cloud Quest 100+ Essential Com... Advanced End-to-E... LINUX - YouTube T... How to Install Jenk...

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

Search for projects... A

How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)? Create your project from your favorite DevOps platform. First, you need to set up a DevOps platform configuration.

From Azure DevOps **From Bitbucket Server** **From Bitbucket Cloud** **From GitHub** **From GitLab**

Set up global configuration Set up global configuration Set up global configuration Set up global configuration Set up global configuration

Now go to Putty and see whether it's installed Trivy, Terraform, Aws cli, Kubectl or not.



```
trivy --version  
aws --version  
terraform --version  
kubectl version
```



```
ubuntu@ip-172-31-11-71:~$  
ubuntu@ip-172-31-11-71:~$ trivy --version  
Version: 0.46.0  
ubuntu@ip-172-31-11-71:~$  
ubuntu@ip-172-31-11-71:~$ aws --version  
aws-cli/2.13.29 Python/3.11.6 Linux/5.19.0-1025-aws exe/x86_64.ubuntu.22 prompt/off  
ubuntu@ip-172-31-11-71:~$  
ubuntu@ip-172-31-11-71:~$ terraform --version  
Terraform v1.6.2  
on linux_amd64  
ubuntu@ip-172-31-11-71:~$  
ubuntu@ip-172-31-11-71:~$ kubectl --version  
error: unknown flag: --version  
See 'kubectl --help' for usage.  
ubuntu@ip-172-31-11-71:~$ kubectl version  
Client Version: v1.28.3  
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3  
Error from server (Forbidden): <html><head><meta http-equiv='refresh' content='1;url=/login?from=%2Fversion%3D32s'></script></head><body style='background-color:white; color:white;'>  
  
Authentication required  
<!--  
-->  
  
</body></html>  
ubuntu@ip-172-31-11-71:~$ █
```

That is done now go to Jenkins and add a terraform plugin to provision the AWS EKS using the Pipeline Job.

Go to Jenkins dashboard -> Manage Jenkins -> Plugins

Available Plugins, Search for Terraform and install it.

The screenshot shows the Jenkins Marketplace interface. A search bar at the top contains the text "Terraform". Below it, a table lists a single plugin: "Terraform 1.0.10". The "Install" button next to it is greyed out, indicating it's already installed. The "Name" column has a dropdown arrow. The "Released" column shows "3 yr 8 mo ago". A note below the table states: "This plugin provides a build wrapper for Terraform to launch and destroy infrastructure."

Go to Putty and use the below command

let's find the path to our Terraform (we will use it in the tools section of Terraform)



```
which terraform
```



A screenshot of a Putty terminal window. The title bar says "Quick connect...". The session name is "1. ubuntu@ip-172-31-42-229: ~". The command "which terraform" was run, and the output is "/usr/bin/terraform". The prompt "ubuntu@ip-172-31-42-229:~\$" appears again.

Now come back to Manage Jenkins -> Tools

Add the terraform in Tools

Terraform installations

Add Terraform

Terraform

Name

Install directory

Install automatically ?

Add Terraform

Save
Apply

Apply and save.

CHANGE YOUR S3 BUCKET NAME IN THE BACKEND.TF

Now create a new job for the Eks provision

Enter an item name

» Required field

Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline

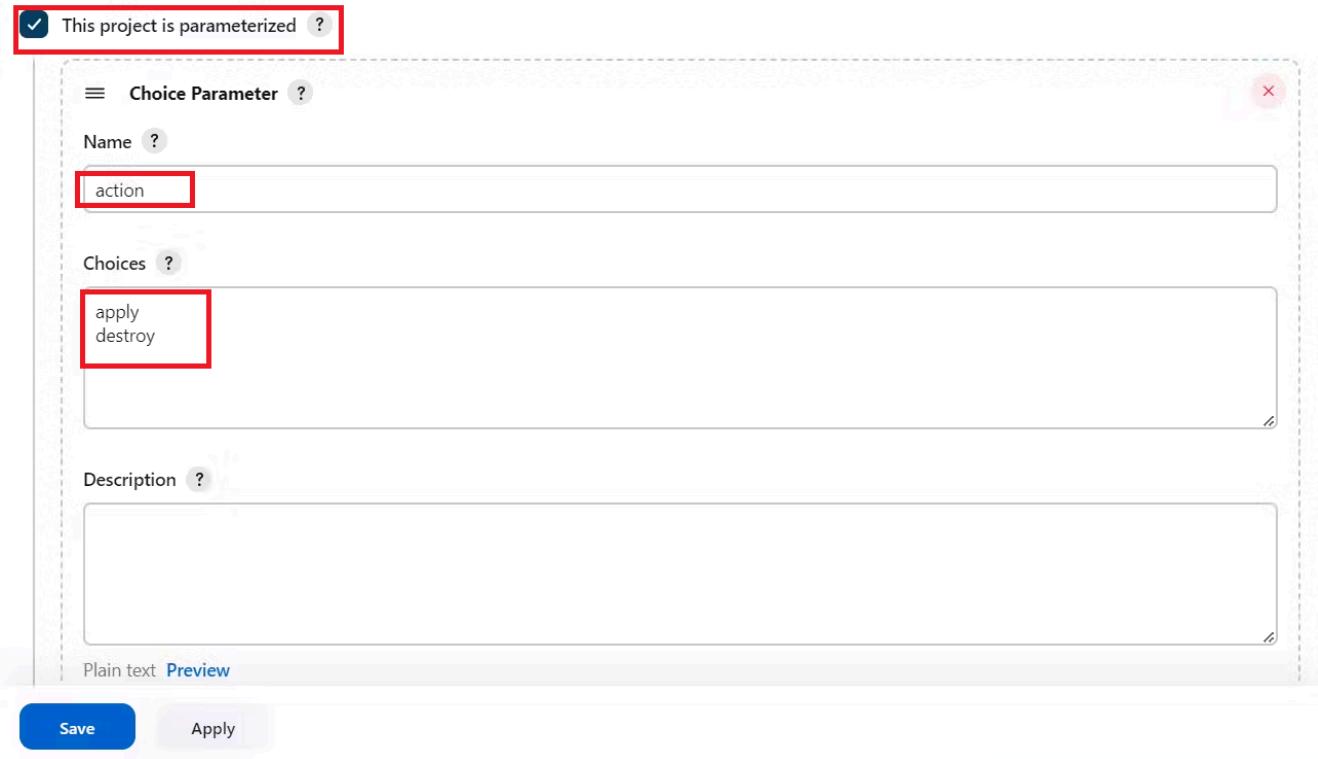
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds etc.

I want to do this with build parameters to apply and destroy while building only.

you have to add this inside job like the below image



Let's add a pipeline



```
pipeline{
    agent any
    stages {
        stage('Checkout from Git'){
            steps{
                git branch: 'main', url: 'https://github.com/Aj7Ay/Tetris'
            }
        }
        stage('Terraform version'){
            steps{
                sh 'terraform --version'
            }
        }
        stage('Terraform init'){
            steps{
                dir('Eks-terraform') {
                    sh 'terraform init'
                }
            }
        }
    }
}
```

```
stage('Terraform validate'){
    steps{
        dir('Eks-terraform') {
            sh 'terraform validate'
        }
    }
}

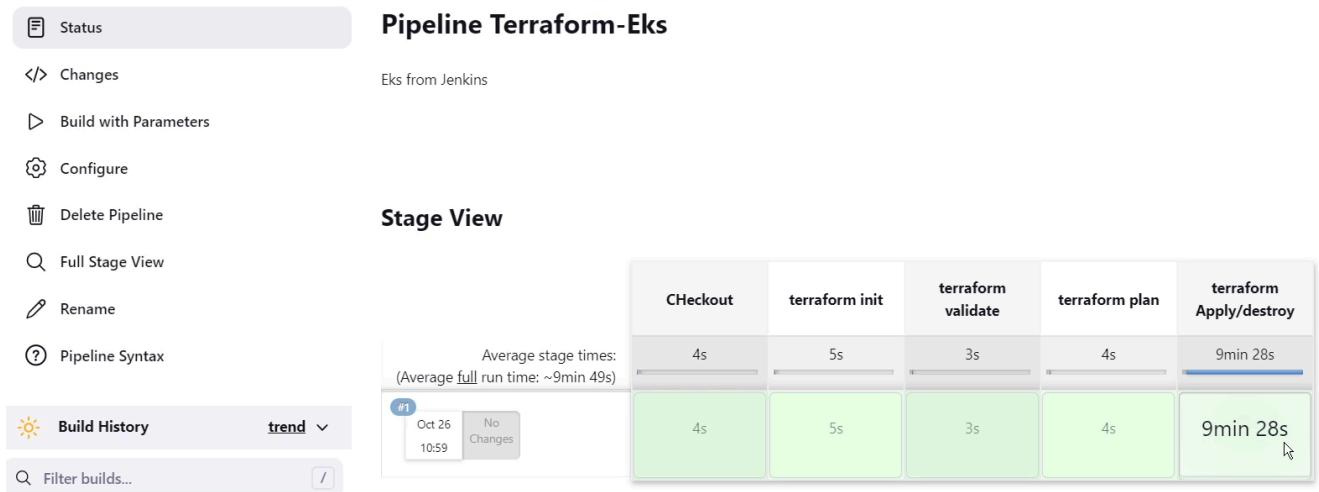
stage('Terraform plan'){
    steps{
        dir('Eks-terraform') {
            sh 'terraform plan'
        }
    }
}

stage('Terraform apply/destroy'){
    steps{
        dir('Eks-terraform') {
            sh 'terraform ${action} --auto-approve'
        }
    }
}
}
```

Let's apply and save and Build with parameters and select action as apply

The screenshot shows the AWS CloudFormation console for a pipeline named 'Terraform-Eks'. The main title is 'Pipeline Terraform-Eks'. On the left, there's a sidebar with actions: Status, Changes, Build with Parameters (which is highlighted with a red box), Configure, Delete Pipeline, Full Stage View, and Rename. Below the sidebar is a message: 'This build requires parameters:'. Underneath, there's a table with one row. The first column is 'action' and the second column is a button labeled 'apply' (also highlighted with a red box). At the bottom right of the page are two buttons: a green 'Build' button with a red border and a grey 'Cancel' button.

Stage view it will take max 10mins to provision



Check in Your Aws console whether it created EKS or not.

EKS > Clusters

Clusters (1) Info

Filter clusters

Cluster name	Status	Kubernetes version	Provider
EKS_CLOUD	Active	1.28	EKS

Ec2 instance is created for the Node group

Instances (1/2) Info

Find Instance by attribute or tag (case-sensitive)

Instance state = running

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Put
Jenkins-ARGO	i-0323f37f837248e53	Running	t2.large	2/2 checks passed	No alarms	ap-south-1b	ec2
	i-049634a401c64808b	Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b	ec2

Now let's build Tetris version 1

We need some plugins to complete this process

Go to Jenkins dashboard

Manage Jenkins -> Plugins -> Available Plugins

Search for the Below Plugins

Eclipse Temurin installer

Sonarqube Scanner

NodeJs

Owasp Dependency-Check

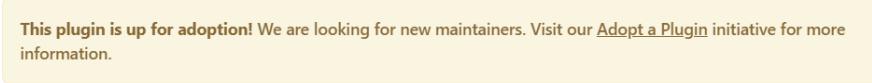
Docker

Docker Commons

Docker Pipeline

Docker API

Docker-build-step

 Eclipse Temurin installer 1.5	Provides an installer for the JDK tool that downloads the JDK from https://adoptium.net	 <p>This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.</p>	1 yr 0 mo ago
 SonarQube Scanner 2.16.1	External Site/Tool Integrations Build Reports	This plugin allows an easy integration of SonarQube , the open source platform for Continuous Inspection of code quality.	15 days ago
 NodeJS 1.6.1	npm	NodeJS Plugin executes NodeJS script as a build step.	2 mo 10 days ago
 OWASP Dependency-Check 5.4.3	Security DevOps Build Tools Build Reports	This plug-in can independently execute a Dependency-Check analysis and visualize results. Dependency-Check is a utility that identifies project dependencies and checks if there are any known, publicly disclosed, vulnerabilities.	1 mo 16 days ago
 Docker 1.5	Cloud Providers Cluster Management docker	This plugin integrates Jenkins with Docker	1 mo 21 days ago

<input checked="" type="checkbox"/>	Docker Commons 439.va_3cb_0a_6a_fb_29	3 mo 17 days ago
Provides the common shared functionality for various Docker-related plugins.		
<input checked="" type="checkbox"/>	Docker Pipeline 572.v950f58993843	2 mo 15 days ago
Build and use Docker containers from pipelines.		
<input checked="" type="checkbox"/>	Docker API 3.3.1-79.v20b_53427e041	4 mo 22 days ago
Library plugins (for use by other plugins) docker		
This plugin provides <code>docker-java</code> API for other plugins.		
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.		
<input checked="" type="checkbox"/>	docker-build-step 2.10	

Configure in Global Tool Configuration

Goto Manage Jenkins → Tools → Install JDK(17) and NodeJs(16) → Click on Apply and Save

Dashboard > Manage Jenkins > Tools
JDK Installations

Add JDK

JDK

Name: jdk17

Install automatically:

Install from adoptium.net:

- Version: jdk-17.0.8.1+1
- Add Installer

Dashboard > Manage Jenkins > Tools

NodeJS

Name: node16

Install automatically:

Install from nodejs.org:

Version: NodeJS 16.2.0

For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail

Force 32bit architecture:

Global npm packages to install

Specify list of packages to install globally -- see `npm install -g`. Note that you can fix the packages version by using the syntax `'packageName@version'`

For Sonarqube use the latest version

[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](#)**For Owasp use the 6.5.1 version**[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 ▾](#)**Use the latest version of Docker**

Add Docker

Docker

Name: docker

Install automatically ?

Download from docker.com

Docker version: latest

Add Installer ▾

Click apply and save.

Configure Sonar Server in Manage Jenkins

Grab the Public IP Address of your EC2 Instance, Sonarqube works on Port 9000, so <Public IP>:9000. Goto your Sonarqube Server. Click on Administration → Security → Users → Click on Tokens and Update Token → Give it a name → and click on Generate Token

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

Administration

Configuration ▾ Security ▾ Projects ▾ System Marketplace

General Edit global Find i

Users Groups Global Permissions Permission Templates

click on update Token

SCM Accounts	Last connection	Groups	Tokens
Administrator admin	< 1 hour ago	sonar-administrators sonar-users	0

Update Tokens

Create a token with a name and generate

Tokens of Administrator

Generate Tokens

Name	Expires in
Enter Token Name	30 days
<input type="button" value="Generate"/>	

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

Copy :squ_21d162904c1c72cf8b39665f96480185c99dc2f9

Name	Type	Project	Last use	Created	Expiration
Jenkins	User		Never	September 8, 2023	October 8, 2023
					<input type="button" value="Revoke"/>

copy Token

Goto Jenkins Dashboard → Manage Jenkins → Credentials → Add Secret Text. It should look like this

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

New credentials

Kind: Secret text

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

Secret: POST THE TOKEN HERE

ID: Sonar-token

Description: Sonar-token

You will see 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

Now, go to Dashboard → Manage Jenkins → System and Add like the below image.

SonarQube servers

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

Environment variables Enable injection of SonarQube server configuration as build environment variables

SonarQube installations

List of SonarQube installations

Name	<input type="text" value="sonar-server"/>	X
Server URL	Default is http://localhost:9000	
	<input type="text" value="http://13.232.17.191:9000"/>	
Server authentication token	SonarQube authentication token. Mandatory when anonymous access is disabled.	
	<input type="text" value="Sonar-token"/>	
	<button>Add</button>	
<input type="button" value="Save"/> <input type="button" value="Apply"/>		

Click on Apply and Save

In the Sonarqube Dashboard add a quality gate also

Administration-> Configuration->Webhooks

The screenshot shows the SonarQube Administration interface. The top navigation bar has a red box around the 'Administration' tab. Below it, the main menu includes 'Configuration', 'Security', 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', and 'Administration'. Under 'Administration', there are tabs for 'General Settings', 'Encryption', and 'Webhooks', with a red box highlighting 'Webhooks'. A search bar is present. The main content area displays user information, with a table showing columns for 'SCM Accounts', 'Last connection', 'Groups', and 'Tokens'. One user, 'Administrator admin', is listed with a green profile icon, last connected ' $< 1 \text{ hour ago}$ ', and group 'sonar-administrators'. A 'Create User' button is located in the top right corner of the main content area.

Click on Create

No webhook defined.

Add details



```
#in url section of quality gate
<http://jenkins-public-ip:8080>/sonarqube-webhook/
```



Create Webhook

All fields marked with * are required

Name *

 ✓

URL *

 ✓

Secret

If provided, secret will be used as the key to generate the HMAC hex (lowercase) digest value in the 'X-Sonar-Webhook-HMAC-SHA256' header.

Create Cancel

Now add Docker credentials to the Jenkins to log in and push the image

Manage Jenkins -> Credentials -> global -> add credential

Add DockerHub Username and Password under Global Credentials

Kind

Username with password

Scope ?

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

Username ?

sevenajay

 Treat username as secret ?

Password ?

.....

ID ?

docker

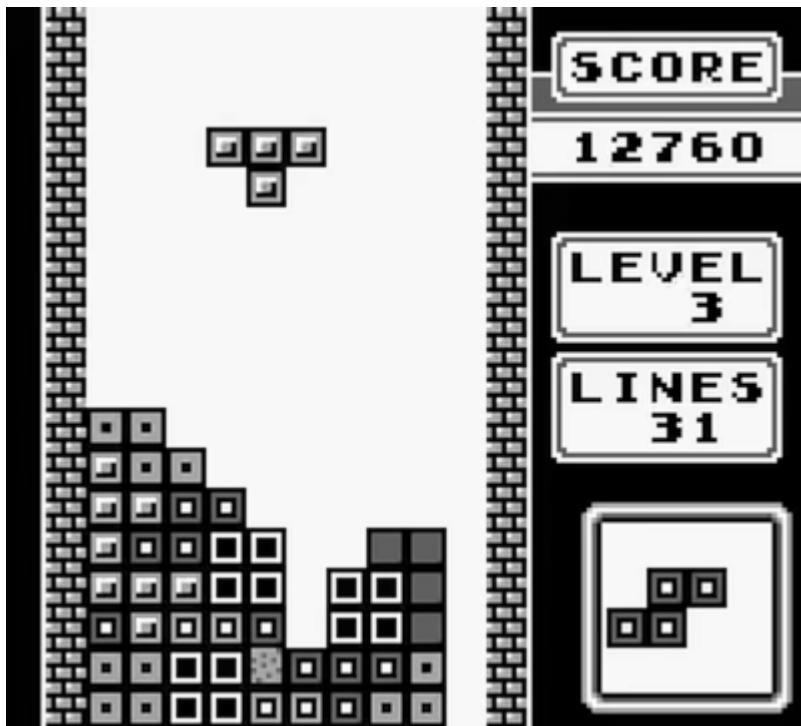
Description ?

docker

[Create](#)

Create.

Version 1.0



Now let's create a new job for our pipeline

Enter an item name

» Required field

Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Add this to Pipeline



```
pipeline{
    agent any
    tools{
        jdk 'jdk17'
        nodejs 'node16'
    }
    environment {
        SCANNER_HOME=tool 'sonar-scanner'
    }
    stages {
        stage('clean workspace'){
            steps{
                cleanWs()
            }
        }
        stage('Checkout from Git'){
            steps{
                git branch: 'main', url: 'https://github.com/Aj7Ay/TetrisVersion1.0'
            }
        }
        stage("Sonarqube Analysis"){
            steps{
                withSonarQubeEnv('sonar-server') {
                    sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=TetrisVersion1.0 -Dsonar.projectKey=TetrisVersion1.0 '''
                }
            }
        }
    }
}
```

```
        }
    }
    stage("quality gate"){
        steps {
            script {
                waitForQualityGate abortPipeline: false, credentialId: 'jenkins-slave'
            }
        }
    }
    stage('Install Dependencies') {
        steps {
            sh "npm install"
        }
    }
    stage('OWASP FS SCAN') {
        steps {
            dependencyCheck additionalArguments: '--scan ./ --disable-dependency-checks'
            dependencyCheckPublisher pattern: '**/dependency-check-report.html'
        }
    }
    stage('TRIVY FS SCAN') {
        steps {
            sh "trivy fs . > trivyfs.txt"
        }
    }
    stage("Docker Build & Push"){
        steps{
            script{
                withDockerRegistry(credentialsId: 'docker', toolName: 'Docker')
                sh "docker build -t tetrисv1 ."
                sh "docker tag tetrисv1 sevenajay/tetrисv1:latest"
                sh "docker push sevenajay/tetrисv1:latest"
            }
        }
    }
    stage("TRIVY"){
        steps{
            sh "trivy image sevenajay/tetrисv1:latest > trivymage.txt"
        }
    }
}
```

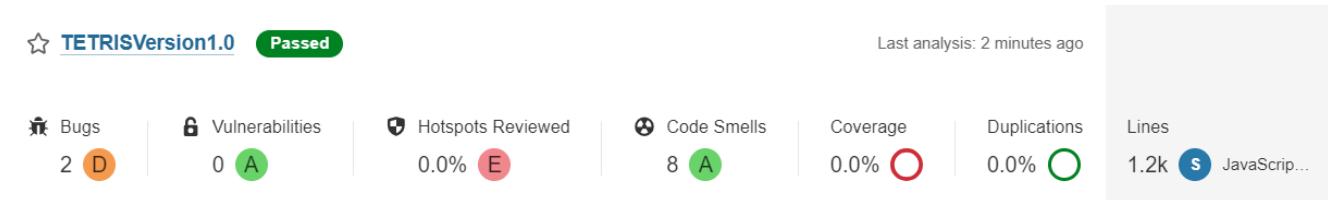
Click on Apply and save.

Build now

Stage view

Declarative: Tool Install	CO	Sonar analysis	QG	NPM	TRIVY FS	OWASP FS SCAN	Docker build and push	TRIVY IMAGE
10s	1s	29s	327ms	20s	3s	12min 29s	1min 9s	57s
199ms	1s	35s	346ms	12s	4s	6min 47s	3min 27s	2min 52s

To see the report, you can go to Sonarqube Server and go to Projects.



You can see the report has been generated and the status shows as passed. You can see that there are 1.2k lines it scanned. To see a detailed report, you can go to issues.

OWASP, You will see that in status, a graph will also be generated and Vulnerabilities.

Dependency-Check Results

Severity Distribution			
15	47	25	
File Name	Vulnerability	Severity	Weakness
⊕ @babel/preset-env:7.12.1	NVD CVE-2023-45133	High	CWE-697
⊕ ansi-html:0.0.7	NVD CVE-2021-23424	High	NVD-CWE-noinfo
⊕ ansi-regex:4.1.0	NVD CVE-2021-3807	High	CWE-1333
⊕ ansi-regex:5.0.0	NVD CVE-2021-3807	High	CWE-1333
⊕ async:2.6.3	NVD CVE-2021-43138	High	CWE-1321
⊕ browserslist:4.14.2	NVD CVE-2021-23364	Medium	CWE-1333
⊕ browserslist:4.16.3	NVD CVE-2021-23364	Medium	CWE-1333
⊕ css-what:3.4.2	OSSINDEX CVE-2022-21222	High	CWE-1333
⊕ decode-uri-component:0.2.0	NVD CVE-2022-38778	Medium	CWE-20
⊕ decode-uri-component:0.2.0	NVD CVE-2022-38900	High	CWE-20

When you log in to Dockerhub, you will see a new image is created

sevenajay / tetrisv1

Description

This repository does not have a description

Last pushed: a minute ago

Docker commands

To push a new tag to this repository:

```
docker push sevenajay/tetrisv1:tagname
```

Let's create a GitHub Token. Go to GitHub

Click on Your Profile on the top right

Aj7Ay

Overview Repositories 42 Projects Packages Stars 1

You unlocked new Achievements with private contributions! Show them off by including private contributions in your settings.

Aj7Ay / README.md

MR CLOUD BOOK
AWS DEVOPS - TUTORIAL - TIPS - LINUX - PYTHON
AJAY KUMAR YEGIREDDI

A passionate DevOps Engineer from India

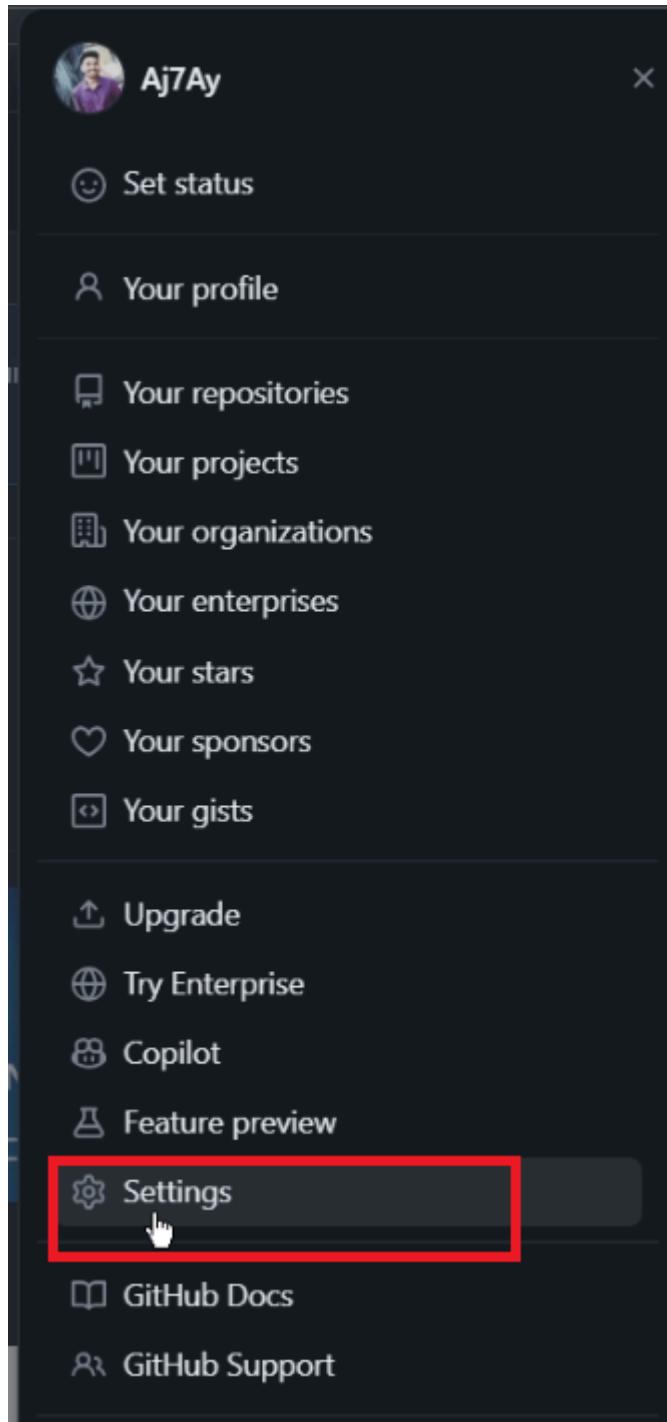
Profile views 3,601

- I'm currently learning DevSecOps & Kubernetes and GitHub actions

Achievements

Sign out

Now click on Settings



Now go down and search for Developer settings and click on it

The screenshot shows the GitHub Profile settings page. On the left sidebar, there are several sections: Pages, Saved replies, Security (with 'Code security and analysis'), Integrations, Applications, Scheduled reminders, Archives, Security log, Sponsorship log, and a 'Developer settings' link which is highlighted with a red box. On the right side, there are sections for Company (with two 'Link to social profile' buttons), Location (with a text input field), and a 'Display current local time' checkbox. Below these is a note about optional fields and privacy. At the bottom right is a green 'Update profile' button.

Now click on Personal Access tokens

The screenshot shows the 'Developer settings' page under 'Personal access tokens'. The 'Personal access tokens' section is highlighted with a red box. Other options like 'GitHub Apps' and 'OAuth Apps' are also visible.

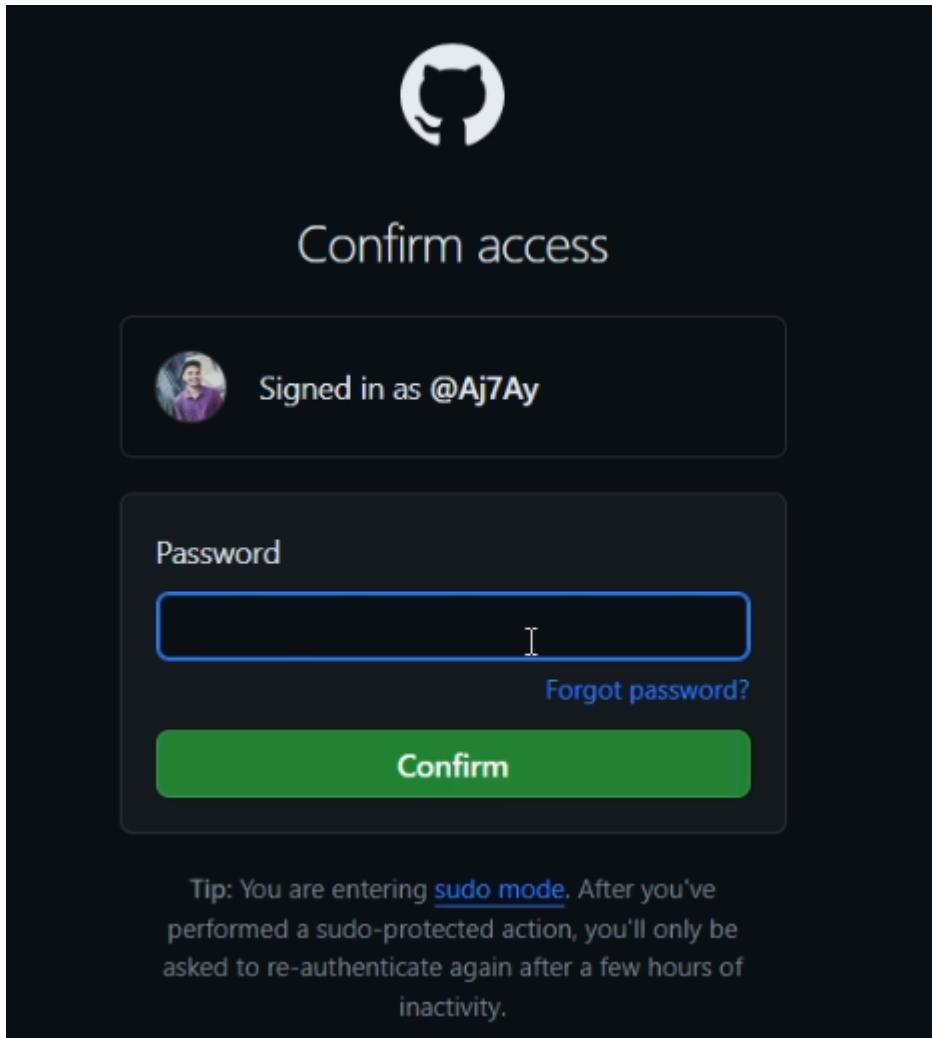
Click on Tokens (Classic)

The screenshot shows the 'Personal access tokens' page under 'Tokens (classic)'. The 'Tokens (classic)' section is highlighted with a red box. Other options like 'GitHub Apps', 'OAuth Apps', and 'Fine-grained tokens' are visible. A 'Beta' badge is present next to 'Fine-grained tokens'.

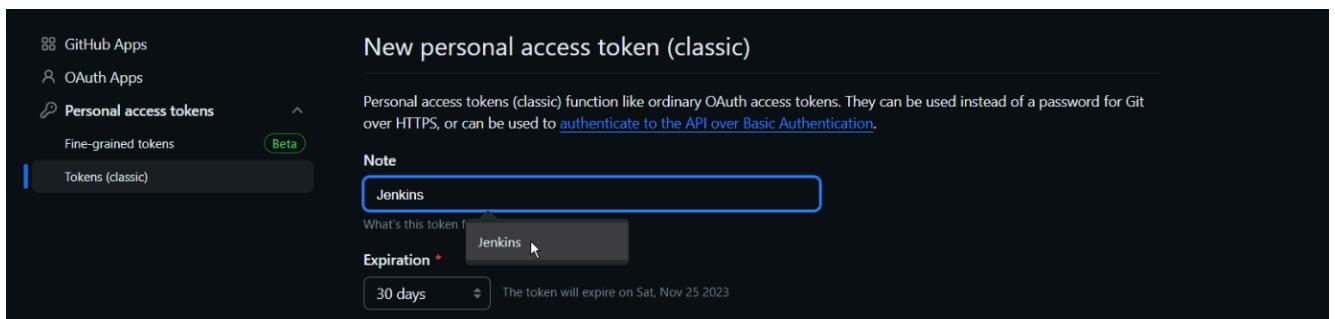
Click on Generate New token -> Generate new token (classic)

The screenshot shows the 'Personal access tokens (classic)' page. It features a 'Generate new token' button at the top right and another 'Generate new token (classic)' button within a 'Fine-grained, repo-scoped' section. Both buttons are highlighted with red boxes.

Now it asks for access provide your GitHub password



Now provide a name for the token



Click on all check box

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input checked="" type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input checked="" type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input checked="" type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input checked="" type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input checked="" type="checkbox"/> read:org	Read org and team membership, read org projects
<input checked="" type="checkbox"/> manage_runners:org	Manage org runners and runner groups

Now click on Generate token

<input checked="" type="checkbox"/> admin:enterprise	Full control of enterprises
<input checked="" type="checkbox"/> manage_runners:enterprise	Manage enterprise runners and runner groups
<input checked="" type="checkbox"/> manage_billing:enterprise	Read and write enterprise billing data
<input checked="" type="checkbox"/> read:enterprise	Read enterprise profile data
<input checked="" type="checkbox"/> audit_log	Full control of audit log
<input checked="" type="checkbox"/> read:audit_log	Read access of audit log
<input checked="" type="checkbox"/> codespace	Full control of codespaces
<input checked="" type="checkbox"/> codespace:secrets	Ability to create, read, update, and delete codespace secrets
<input checked="" type="checkbox"/> copilot	Full control of GitHub Copilot settings and seat assignments
<input checked="" type="checkbox"/> manage_billing:copilot	View and edit Copilot for Business seat assignments
<input checked="" type="checkbox"/> project	Full control of projects
<input checked="" type="checkbox"/> read:project	Read access of projects
<input checked="" type="checkbox"/> admin:gpg_key	Full control of public user GPG keys
<input checked="" type="checkbox"/> write:gpg_key	Write public user GPG keys
<input checked="" type="checkbox"/> read:gpg_key	Read public user GPG keys
<input checked="" type="checkbox"/> admin:ssh_signing_key	Full control of public user SSH signing keys
<input checked="" type="checkbox"/> write:ssh_signing_key	Write public user SSH signing keys
<input checked="" type="checkbox"/> read:ssh_signing_key	Read public user SSH signing keys

Generate token
Cancel

Now copy the token

The screenshot shows the GitHub 'Personal access tokens (classic)' page. On the left, there's a sidebar with options like 'GitHub Apps', 'OAuth Apps', 'Personal access tokens' (which is selected and has a 'Beta' badge), 'Fine-grained tokens', and 'Tokens (classic)'. At the top right are buttons for 'Generate new token' and 'Revoke all'. Below the sidebar, a message says 'Tokens you have generated that can be used to access the [GitHub API](#)'. A prominent message at the top of the main area says 'Make sure to copy your personal access token now. You won't be able to see it again!'. Below this is a card containing a generated token: 'ghp_E1rYRlwf8v6DvtQy6efEgirKduVJmf0OwyRU' with a green checkmark icon and a 'Delete' button. A note at the bottom explains that these tokens function like OAuth access tokens.

Now go to the Jenkins dashboard

Manage Jenkins -> credentials -> Global

Add credential

The screenshot shows the Jenkins 'New credentials' configuration page under 'Global credentials (unrestricted)'. The 'Kind' dropdown is set to 'Secret text'. The 'Scope' dropdown is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Secret' field contains '.....' followed by 'GITHUB TOKEN' (with the first part redacted). The 'ID' field is 'github' (with the entire field redacted). The 'Description' field is 'github'. At the bottom is a 'Create' button, which is highlighted with a red box.

Let's add the Image Updater stage to the Pipeline



```
#add inside environment
environment {
    GIT_REPO_NAME = "Tetris-manifest"
    GIT_USER_NAME = "Aj7Ay"      # change your Github Username here
}
# add these stages after trivy image scan
stage('Checkout Code') {
    steps {
        git branch: 'main', url: 'https://github.com/Aj7Ay/Tetr'
    }
}
```

```

        }
    }
    stage('Update Deployment File') {
        steps {
            script {
                withCredentials([string(credentialsId: 'github', var
                    NEW_IMAGE_NAME = "sevenajay/tetrisv1:latest"    #!
                    sh "sed -i 's|image: .*|image: $NEW_IMAGE_NAME|'
                    sh 'git add deployment.yml'
                    sh "git commit -m 'Update deployment image to $NE
                    sh "git push https:// ${GITHUB_TOKEN}@github.com/`$REPO_N
                }
            }
        }
    }
}

```

Complete Pipeline



```

pipeline{
    agent any
    tools{
        jdk 'jdk17'
        nodejs 'node16'
    }
    environment {
        SCANNER_HOME=tool 'sonar-scanner'
        GIT_REPO_NAME = "Tetris-manifest"
        GIT_USER_NAME = "Aj7Ay"      # change your Github Username here
    }
    stages {
        stage('clean workspace'){
            steps{
                cleanWs()
            }
        }
        stage('Checkout from Git'){
            steps{
                git branch: 'main', url: 'https://github.com/Aj7Ay/Tetr
            }
        }
    }
}

```

```

    }
    stage("Sonarqube Analysis ){
        steps{
            withSonarQubeEnv('sonar-server') {
                sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectKey=TetrisVersion1.0 '''
            }
        }
    }
    stage("quality gate"){
        steps {
            script {
                waitForQualityGate abortPipeline: false, credentials:
            }
        }
    }
    stage('Install Dependencies') {
        steps {
            sh "npm install"
        }
    }
    stage('OWASP FS SCAN') {
        steps {
            dependencyCheck additionalArguments: '--scan ./ --disableCheckForVulnerableDependencies'
            dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
        }
    }
    stage('TRIVY FS SCAN') {
        steps {
            sh "trivy fs . > trivyfs.txt"
        }
    }
    stage("Docker Build & Push"){
        steps{
            script{
                withDockerRegistry(credentialsId: 'docker', toolName: 'Docker')
                sh "docker build -t tetrисv1 ."
                sh "docker tag tetrисv1 sevenajay/tetrисv1:latest"
                sh "docker push sevenajay/tetrисv1:latest "
            }
        }
    }
    stage("TRIVY"){
        steps{

```

```
        sh "trivy image sevenajay/tetrisv1:latest > trivyimage.txt"
    }
}

stage('Checkout Code') {
    steps {
        git branch: 'main', url: 'https://github.com/Aj7Ay/Tetris'
    }
}

stage('Update Deployment File') {
    steps {
        script {
            withCredentials([string(credentialsId: 'github', variable: 'GITHUB_TOKEN')])
                NEW_IMAGE_NAME = "sevenajay/tetrisv1:latest"      # New image name
                sh "sed -i 's|image: .*|image: $NEW_IMAGE_NAME|' deployment.yaml"
                sh 'git add deployment.yaml'
                sh "git commit -m 'Update deployment image to $NEW_IMAGE_NAME'"
                sh "git push https://${GITHUB_TOKEN}@github.com/sevenajay/Tetris.git"
        }
    }
}
}
```

Click on Apply and save

Now click on Build and you can see no image inside the deployment file

Tetris-manifest / deployment.yaml

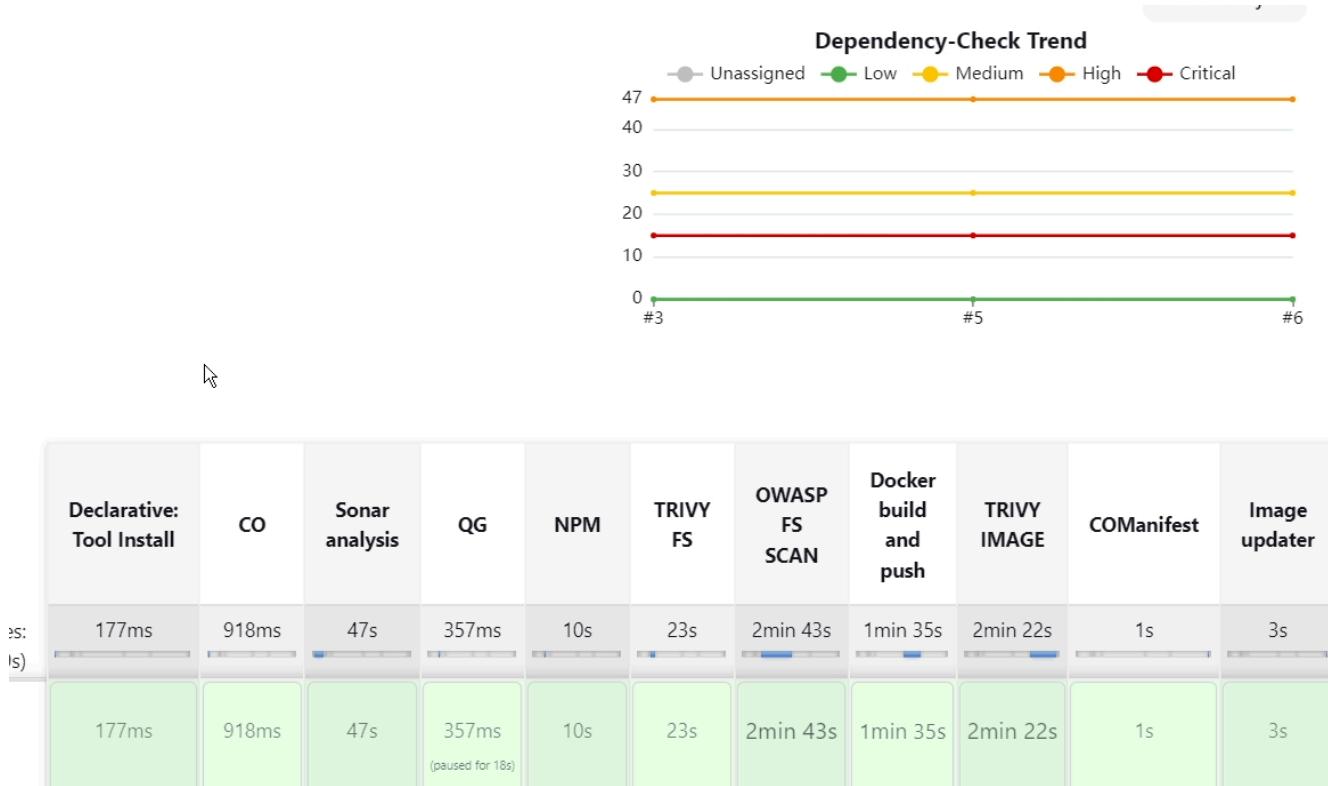
Aj7Ay Update deployment.yaml c8df42b · 13 hours ago History

Code Blame 35 lines (33 loc) · 550 Bytes

Raw ⌂ ⌄ ⌅ ⌆ ⌇

```
1 ---  
2 apiVersion: apps/v1  
3 kind: Deployment  
4 metadata:  
5   name: tetris  
6 spec:  
7   replicas: 3  
8   selector:  
9     matchLabels:  
10    app: tetris  
11 template:  
12   metadata:  
13     labels:  
14       app: tetris  
15   spec:  
16     containers:  
17     - name: tetris  
18       image: image update  
19     ports:  
20       - containerPort: 3000 # Use port 3000  
21
```

Stage view



Now if you see the deployment file in the Tetris manifest repo

The image is automatically updated after the build

Tetris-manifest / deployment.yml

```

Code Blame 35 lines (33 loc) · 563 Bytes

1   ---
2   apiVersion: apps/v1
3   kind: Deployment
4   metadata:
5     name: tetris
6   spec:
7     replicas: 3
8     selector:
9       matchLabels:
10      app: tetris
11     template:
12       metadata:
13         labels:
14           app: tetris
15       spec:
16         containers:
17           - name: tetris
18             image: sevenajay/tetrisv1:latest
19             ports:
20               - containerPort: 3000 # Use port 3000
21

```

Let's Update the kubeconfig

Go to Putty of your Jenkins instance SSH and enter the below command

```
aws eks update-kubeconfig --name <CLUSTER NAME> --region <CLUSTER REGION>
aws eks update-kubeconfig --name EKS_CLOUD --region ap-south-1
```

Let's see the nodes

```
kubectl get nodes
```

```
ubuntu@ip-172-31-11-71:~$ kubectl get nodes
NAME           STATUS   ROLES      AGE     VERSION
ip-172-31-13-85.ap-south-1.compute.internal   Ready    <none>    111m   v1.28.1-eks-43840fb
```

ARGO CD SETUP

Let's install ArgoCD

ARGOCD INSTALLATION LINK

You will redirected to this page

The screenshot shows the AWS EKS Workshop interface. On the left, there's a sidebar with navigation links for 'Introduction', 'Start the workshop...', 'Launch using eksctl', 'Beginner' (with 'Migrate to EKS', 'Resource Management', 'CI/CD with CodePipeline', 'Logging with Amazon OpenSearch', 'Fluent Bit, and OpenSearch', 'Dashboards', 'Monitoring using Prometheus and Grafana', and 'Monitoring using Pixie'), and 'Intermediate' (with 'Migrate to EKS', 'Resource Management', 'CI/CD with CodePipeline', 'Logging with Amazon OpenSearch', 'Fluent Bit, and OpenSearch', 'Dashboards', 'Monitoring using Prometheus and Grafana', and 'Monitoring using Pixie'). The main content area has a title 'INSTALL ARGO CD' and a 'Warning' message: 'This workshop has been deprecated and archived. The new Amazon EKS Workshop is now available at www.eksworkshop.com'. Below this is a section titled 'ArgoCD Architecture' with a diagram. The diagram illustrates the workflow: a user interacts with a UI or CLI, which connects to an API. The API interacts with a Repository Service (containing a 'search application' and a 'guestbook application') and a git repository. The git repository receives 'PR merge' and 'webhook event' notifications, which trigger updates to the applications.

All those components could be installed using a manifest provided by the Argo Project: use the below commands

```
kubectl create namespace argocd
kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/ai
```

The screenshot shows AWS CloudShell on the 'ap-south-1' region. The terminal window displays the command execution process. It starts with creating a namespace:

```
[cloudshell-user@ip-10-2-87-125 ~]$ kubectl create namespace argocd
```

Then it applies the manifest from the GitHub URL:

```
[cloudshell-user@ip-10-2-87-125 ~]$ kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/v2.4.7/manifests/install.yaml
```

The output shows numerous resources being created, including custom resource definitions, service accounts, role-based access control (RBAC) configurations, and various controllers and roles.

COMMANDS ARGOCD

By default, argocd-server is not publicly exposed. For this project, we will use a Load Balancer to make it usable:



```
kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "LoadBa' [REDACTED]
```



AWS CloudShell

ap-south-1

```
[cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "LoadBalancer"}}'
service/argocd-server patched
[cloudshell-user@ip-10-2-87-125 ~]$ [REDACTED]
```

One load balancer will created in the AWS



Searched node

Capacity Reservations

Images

- AMIs
- AMI Catalog

Elastic Block Store

- Volumes
- Snapshots
- Lifecycle Manager

Network & Security

- Security Groups

[EC2](#) > Load balancers

Load balancers (1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Name	DNS name	State	VPC ID	Availability Zones	Type	Date c
a646906ffc5264bf8807d8... d85414d5b69f	a646906ffc5264bf8807d8... d85414d5b69f	-	vpc-0f6bdd74ced5c07c0	3 Availability Zones	classic	October (UTC+0)

Wait about 2 minutes for the LoadBalancer creation



```
sudo apt install jq -y [REDACTED]
```



```
export ARGOCD_SERVER=`kubectl get svc argocd-server -n argocd -o json | [REDACTED]
```



when you run this command, it will export the hostname of the ArgoCD server's load balancer and store it in the `ARGOCD_SERVER` environment variable, which you can then use in other commands or scripts to interact with the ArgoCD server. This can be useful when you need to access the ArgoCD web UI or interact with the server programmatically.

```
ap-south-1
[cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ export ARGOCD_SERVER=`kubectl get svc argocd-server -n argocd -o json | jq --raw-output '.status.loadBalancer.ingress[0].hostname'` [cloudshell-user@ip-10-2-87-125 ~]$
```

If run this command you will get the load balancer external IP



```
echo $ARGOCD_SERVER
```



```
[cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ echo $ARGOCD_SERVER a646906ffc5264bf8807d85414d5b69f-937542690.ap-south-1.elb.amazonaws.com [cloudshell-user@ip-10-2-87-125 ~]$
```

Login

The command you provided is used to extract the password for the initial admin user of ArgoCD, decode it from base64 encoding, and store it in an environment variable named `ARGO_PWD`.



```
export ARGO_PWD=`kubectl -n argocd get secret argocd-initial-admin-secret
```



```
[cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ [cloudshell-user@ip-10-2-87-125 ~]$ export ARGO_PWD=`kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath=".data.password" | base64 -d` [cloudshell-user@ip-10-2-87-125 ~]$
```

If you want to see your password provide the below command

```
echo $ARGO_PWD
```



```
[cloudshell-user@ip-10-2-87-125 ~]$  
[cloudshell-user@ip-10-2-87-125 ~]$  
[cloudshell-user@ip-10-2-87-125 ~]$ echo $ARGO_PWD  
ky21EiuybUiMev8k  
[cloudshell-user@ip-10-2-87-125 ~]$
```

Now copy the load balancer IP and paste it into the browser

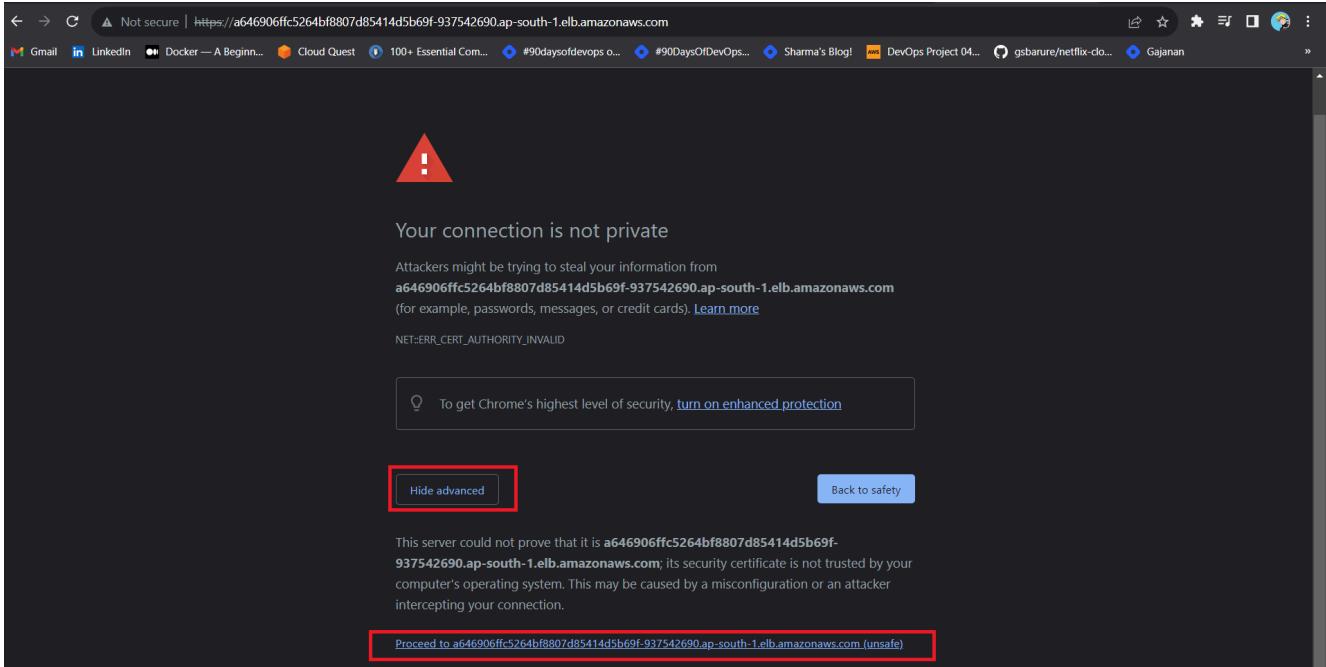


```
echo $ARGOCD_SERVER
```

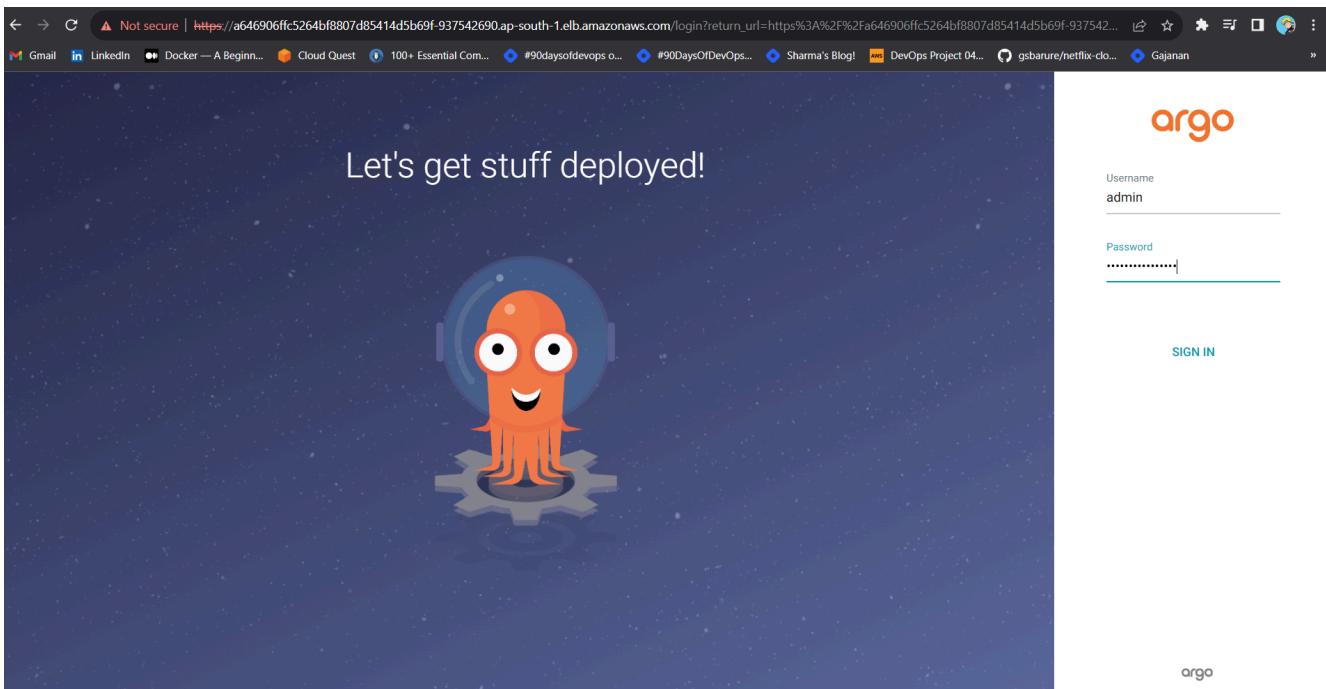


```
[cloudshell-user@ip-10-2-87-125 ~]$  
[cloudshell-user@ip-10-2-87-125 ~]$  
[cloudshell-user@ip-10-2-87-125 ~]$ echo $ARGOCD_SERVER  
a646906ffc5264bf8807d85414d5b69f-937542690.ap-south-1.elb.amazonaws.com  
[cloudshell-user@ip-10-2-87-125 ~]$  
[cloudshell-user@ip-10-2-87-125 ~]$
```

Now you will see this page. if you get an error click on advanced and click on proceed.



Now you will see this page and log in to ArgoCD



Username is admin

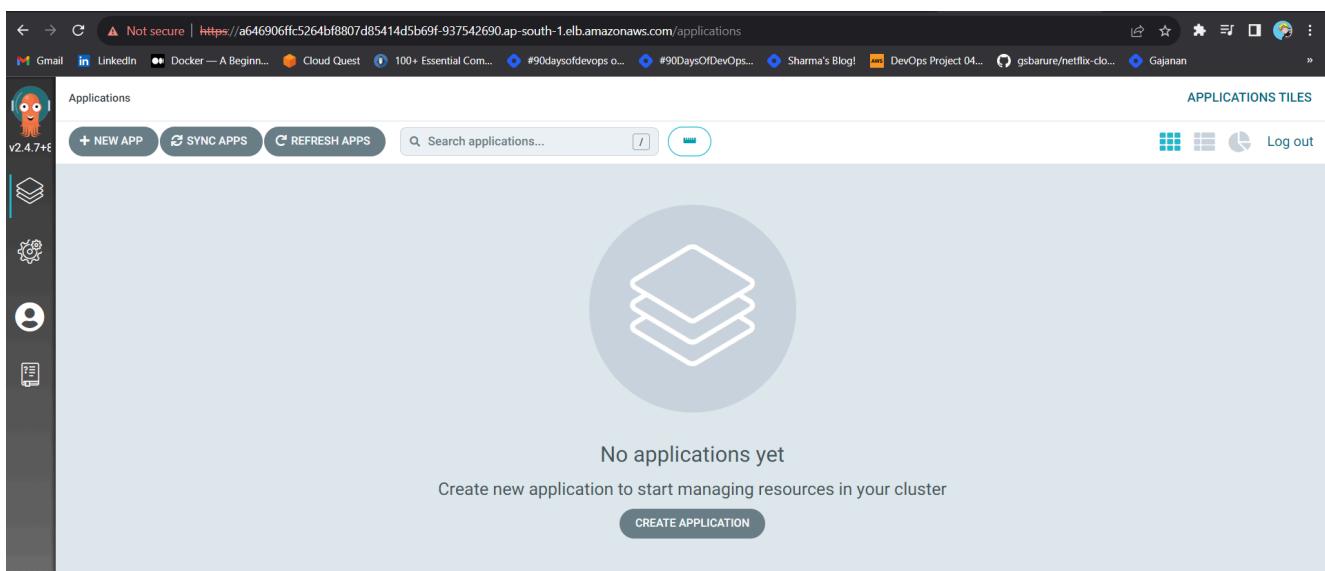
For the password, you have to provide the below command and copy it

```
echo $ARGO_PWD
```

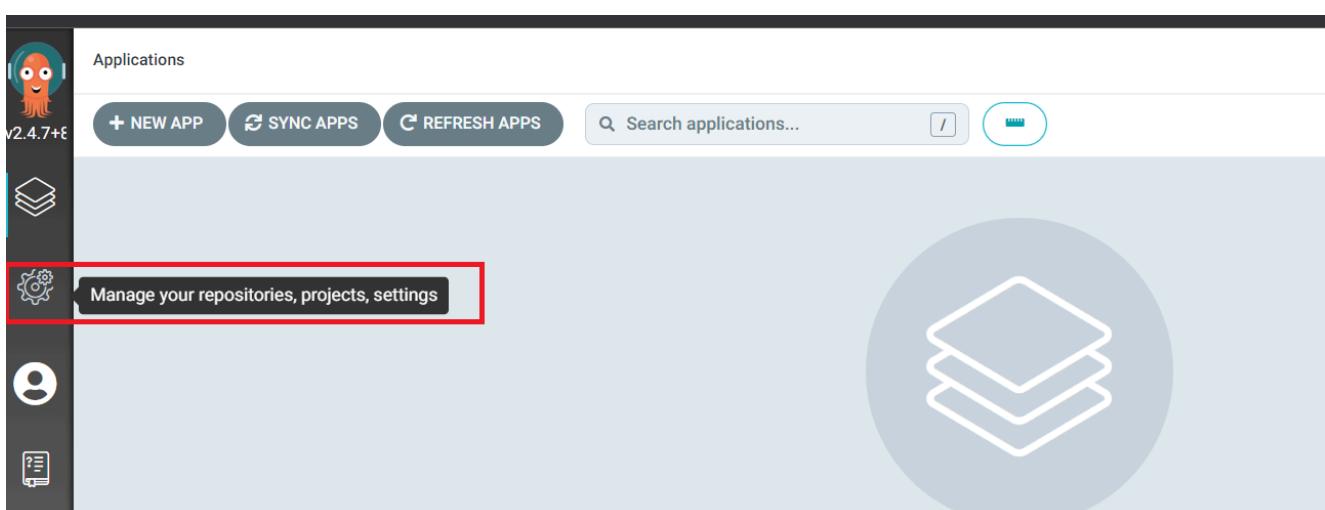
```
[cloudshell-user@ip-10-2-87-125 ~]$  
[cloudshell-user@ip-10-2-87-125 ~]$  
[cloudshell-user@ip-10-2-87-125 ~]$ echo $ARGO_PWD  
ky21EiuybUiMev8k  
[cloudshell-use
```

Emoji	Win+Period
Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Paste as plain text	Ctrl+Shift+V
Select all	Ctrl+A

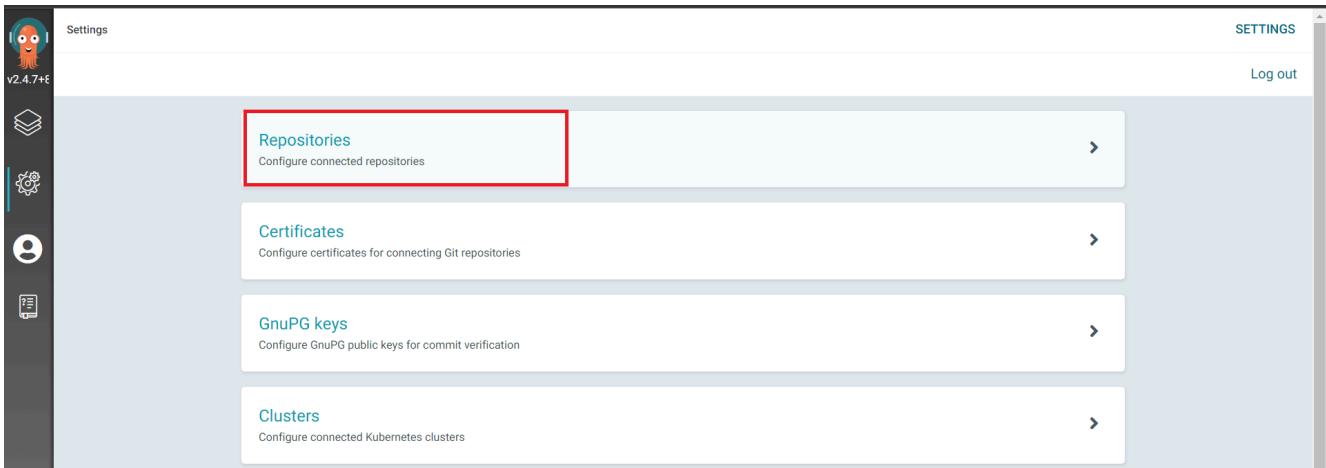
Click on Sign in and you will see this page.



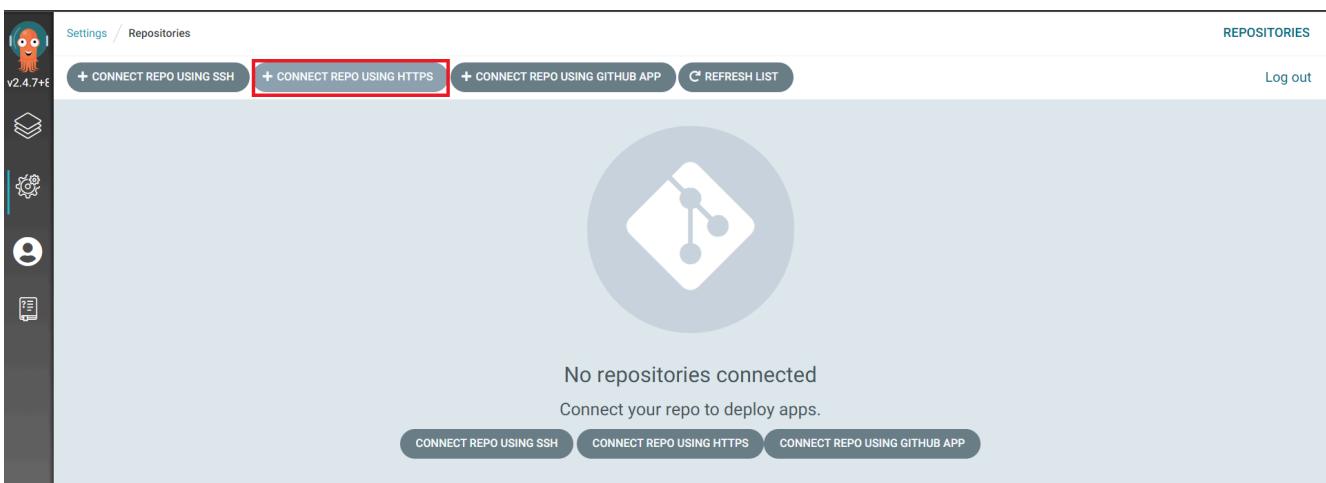
Now click on the Setting gear icon in the left side panel



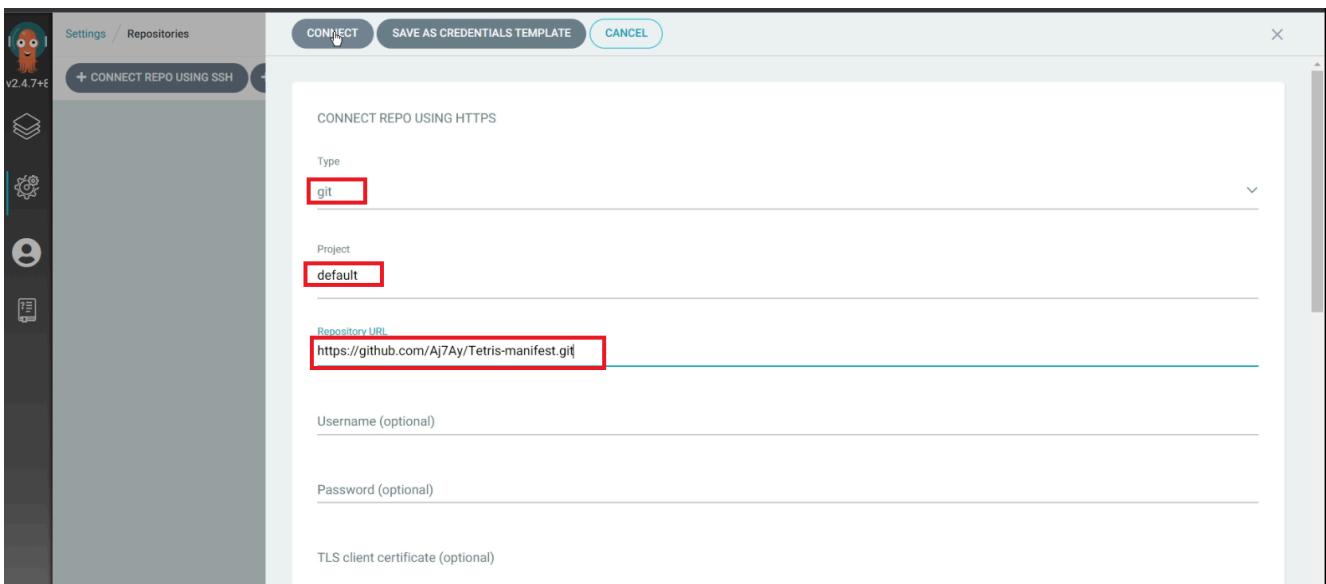
Click on Repositories



Now click on Connect Repo Using HTTPS



Add Github details, Type as git, Project as default and provide the GitHub URL of this manifest and click on connect



You will get Connection Status as Successful

The screenshot shows the ArgoCD web interface. On the left is a sidebar with icons for Helm, Kubernetes, and Settings. The main area has tabs for 'Settings' and 'Repositories'. At the top, there are buttons for connecting repos via SSH, HTTPS, or GitHub App, and a 'REFRESH LIST' button. Below these are three cards: one for 'git' with the URL <https://github.com/Aj7Ay/Tetris-manifest.git> and a 'Successful' connection status.

Click on Manage Your application

This screenshot shows the same ArgoCD interface as above, but the 'Manage your applications, and diagnose health problems.' button in the top navigation bar is highlighted with a red box.

You will see this page and click on New App

The screenshot shows the 'Applications' page. The left sidebar includes a 'New App' button, which is highlighted with a red box. Other buttons include 'SYNC APPS' and 'REFRESH APPS'. A search bar and a 'Log out' link are also visible.

Now provide the following details as in the image

The screenshot shows the 'CREATE' dialog for a new application. The 'Application Name' field contains 'tetris', the 'Project Name' field contains 'default', and the 'SYNC POLICY' dropdown is set to 'Automatic'. There are also checkboxes for 'PRUNE RESOURCES' and 'SELF HEAL'.

Field	Value
Application Name	tetris
Project Name	default
Sync Policy	Automatic
Prune Resources	<input type="checkbox"/>
Self Heal	<input type="checkbox"/>

SOURCE

Repository URL

GIT ✓

Revision

Branches ▾

Path

X

Revision

HEAD

Branches ▾

Path

DESTINATION

Cluster URL

URL ▾

Namespace

default

Click on Create.

You can see our app is created in Argo-cd

tetris

Project: default

Labels:

Status: Missing OutOfSync Syncing

Repository: https://github.com/Aj7Ay/Tetris-manifest...

Target Ref: HEAD

Path: ./

Destination: in-cluster

Namespace: default

SYNC **REFRESH** **DELETE**

Click on tetris and it will create another load balancer in AWS

Load balancers (2)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Create load balancer

Name	DNS name	State	VPC ID	Availability Zones	Type	Date
ab005ebc4af0a4b929...	ab005ebc4af0a4b929deb...	-	vpc-0f6bdd74ced5c07c0	3 Availability Zones	classic	Oct 26
a0725d5c76f3e47a8af...	a0725d5c76f3e47a8af3c7...	-	vpc-0f6bdd74ced5c07c0	3 Availability Zones	classic	Oct 26

Now click on three dots beside tetris-service and click on the details

APP DETAILS **SYNC** **SYNC STATUS** **HISTORY AND ROLLBACK** **DELETE** **REFRESH**

APP HEALTH **Progressing**

CURRENT SYNC STATUS **Synced** **LAST SYNC RESULT** **Sync OK**

Author: Jenkins <jenkins@ip-172-31-11-71.ap-south-1.compute.in...>

Comment: Update deployment image to sevenajay/tetrisv1:latest

DETAILS

FILTERS

NAME: tetris

KINDS: SVC

SYNC STATUS: Synced (2), OutOfSync (0)

Now copy the hostname address

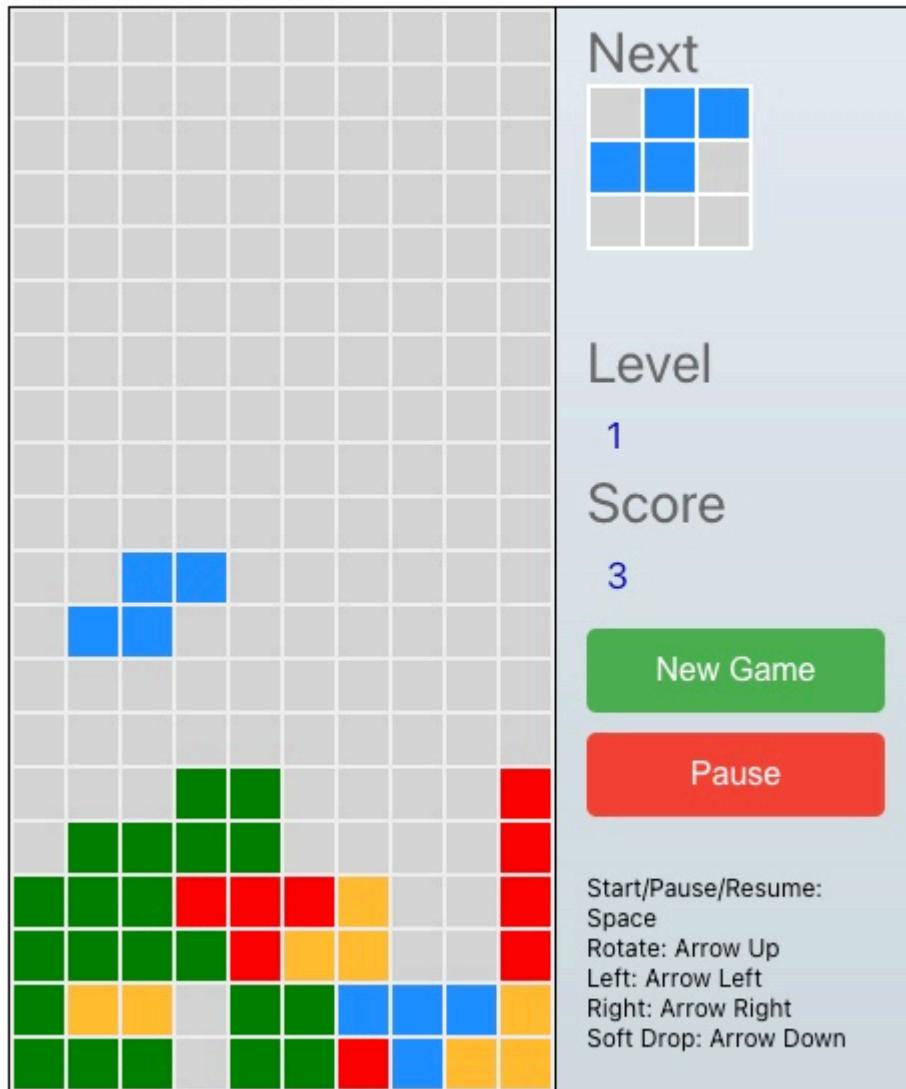
The screenshot shows the ArgoCD interface for the 'tetris' application. On the left, there's a sidebar with icons for Home, Applications, Workspaces, and Help. The main area displays the 'tetris-service' configuration under the 'svc' tab. The 'SUMMARY' tab is selected, showing the following details:

KIND	Service
NAME	tetris-service
NAMESPACE	default
CREATED_AT	10/08/2023 13:19:01
TYPE	LoadBalancer
HOSTNAMES	a200ae62c84984addb6895a965eaa459-986705060.ap-south-1.elb.amazonaws.com
STATUS	Synced
HEALTH	Healthy

The 'HOSTNAMES' row is highlighted with a red box.

Paste it in a browser you will see this page

React Tetris



If you don't get the output

Go to Jenkins machine SSH Putty and provide the below the command



```
kubectl get all
```



```
YfMUW0R00A6qn-5q
ubuntu@ip-172-31-11-71:~$ kubectl get all
NAME                                     READY   STATUS    RESTARTS   AGE
pod/tetris-698657b8dc-4nqmb   1/1     Running   0          77s
pod/tetris-698657b8dc-shd2x   1/1     Running   0          77s
pod/tetris-698657b8dc-tktwq   1/1     Running   0          77s

NAME              TYPE        CLUSTER-IP      EXTERNAL-IP           PORT(S)          AGE
service/kubernetes  ClusterIP   10.100.0.1   <none>            443/TCP         123m
service/tetris-service LoadBalancer  10.100.212.20  a0725d5c76f3e47a8af3c7391c061307-244742789.ap-south-1.elb.amazonaws.com  80:31849/TCP   77s

NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/tetris  3/3     3           3           77s

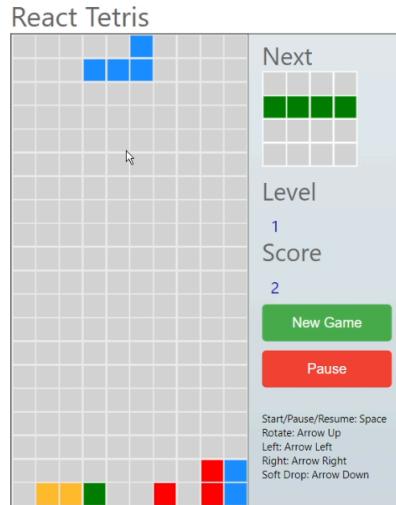
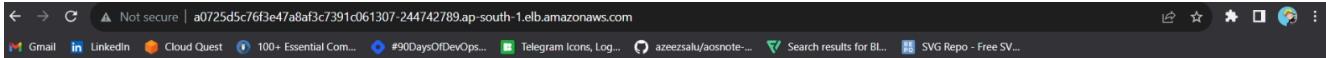
NAME          DESIRED  CURRENT   READY   AGE
replicaset.apps/tetris-698657b8dc  3       3       3       77s
ubuntu@ip-172-31-11-71:~$
```

Open the port that you get like on the above page for the Node group Ec2 instance.



You have to open whatever port you get on The Node group EC2 instance security group

Then you can play the version 1 game



Version 2.0



Let's Build version 2.0 Tetris game

Add this pipeline to the Job



```
pipeline{
    agent any
    tools{
        jdk 'jdk17'
        nodejs 'node16'
    }
}
```

```

environment {
    SCANNER_HOME=tool 'sonar-scanner'
    GIT_REPO_NAME = "Tetris-manifest"
    GIT_USER_NAME = "Aj7Ay"      # change your Github Username here
}
stages {
    stage('clean workspace'){
        steps{
            cleanWs()
        }
    }
    stage('Checkout from Git'){
        steps{
            git branch: 'main', url: 'https://github.com/Aj7Ay/Tetris-manifest'
        }
    }
    stage("Sonarqube Analysis"){
        steps{
            withSonarQubeEnv('sonar-server') {
                sh '''$SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=TetrisVersion2.0 -Dsonar.projectKey=TetrisVersion2.0'''
            }
        }
    }
    stage("quality gate"){
        steps {
            script {
                waitForQualityGate abortPipeline: false, credentialsId: 'jenkins-admin'
            }
        }
    }
    stage('Install Dependencies') {
        steps {
            sh "npm install"
        }
    }
    stage('OWASP FS SCAN') {
        steps {
            dependencyCheck additionalArguments: '--scan ./ --disableCheckForKnownVulnerabilities'
            dependencyCheckPublisher pattern: '**/dependency-check-report.html'
        }
    }
    stage('TRIVY FS SCAN') {
        steps {
            sh "trivy fs . > trivyfs.txt"
        }
    }
}

```

```

    }
}

stage("Docker Build & Push"){
    steps{
        script{
            withDockerRegistry(credentialsId: 'docker', toolName
                sh "docker build -t tetrisv2 ."
                sh "docker tag tetrisv2 sevenajay/tetrisv2:lates"
                sh "docker push sevenajay/tetrisv2:latest "
            }
        }
    }
}

stage("TRIVY"){
    steps{
        sh "trivy image sevenajay/tetrisv2:latest > trivyimage.t
    }
}

stage('Checkout Code') {
    steps {
        git branch: 'main', url: 'https://github.com/Aj7Ay/Tetr
    }
}

stage('Update Deployment File') {
    steps {
        script {
            withCredentials([string(credentialsId: 'github', var
                NEW_IMAGE_NAME = "sevenajay/tetrisv2:latest"      #
                sh "sed -i 's|image: .*|image: $NEW_IMAGE_NAME|'
                sh 'git add deployment.yml'
                sh "git commit -m 'Update deployment image to $NI
                sh "git push https://${GITHUB_TOKEN}@github.com/(
            }
        }
    }
}
}

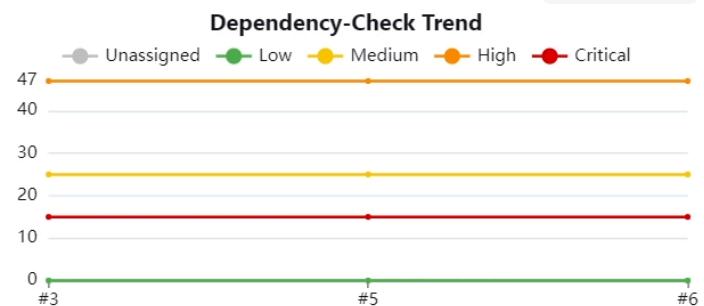
```

Mr Cloud Book

[Home](#) [Blog](#) [DevSecOps](#) [Contact](#) [About Me](#) [Testimonials](#)

Search Blogs

Stage view



Declarative: Tool Install	CO	Sonar analysis	QG	NPM	TRIVY FS	OWASP FS SCAN	Docker build and push	TRIVY IMAGE	COManifest	Image updater
177ms	918ms	47s	357ms	10s	23s	2min 43s	1min 35s	2min 22s	1s	3s
177ms	918ms	47s	357ms (paused for 18s)	10s	23s	2min 43s	1min 35s	2min 22s	1s	3s

To see the report, you can go to Sonarqube Server and go to Projects

TETRISVersion2.0 Passed
Last analysis: 2 minutes ago

Bugs 1 C

 Vulnerabilities 0 A

 Hotspots Reviewed 0.0% E

 Code Smells 12 A

 Coverage 0.0% O

 Duplications 0.0% O

 Lines 1k S JavaScript, ...

You can see the report has been generated and the status shows as passed. You can see that there are 1k lines it scanned. To see a detailed report, you can go to issues.

OWASP, You will see that in status, a graph will also be generated and Vulnerabilities.

Dependency-Check Results

Severity Distribution			
15	47	25	
Search			
File Name	Vulnerability	Severity	Weakness
⊕ @babel/preset-env:7.12.1	NVD CVE-2023-45133	High	CWE-697
⊕ ansi-html:0.0.7	NVD CVE-2021-23424	High	NVD-CWE-noinfo
⊕ ansi-regex:4.1.0	NVD CVE-2021-3807	High	CWE-1333
⊕ ansi-regex:5.0.0	NVD CVE-2021-3807	High	CWE-1333
⊕ async:2.6.3	NVD CVE-2021-43138	High	CWE-1321
⊕ browserslist:4.14.2	NVD CVE-2021-23364	Medium	CWE-1333
⊕ browserslist:4.16.3	NVD CVE-2021-23364	Medium	CWE-1333
⊕ css-what:3.4.2	OSSINDEX CVE-2022-21222	High	CWE-1333
⊕ decode-uri-component:0.2.0	NVD CVE-2022-38778	Medium	CWE-20
⊕ decode-uri-component:0.2.0	NVD CVE-2022-38900	High	CWE-20

When you log in to Dockerhub, you will see a new image is created

sevenajay / tetrisv2

Description

This repository does not have a description

Docker commands

To push a new tag to this repository:

```
docker push sevenajay/tetrisv2:tagname
```

[Public View](#)

Now if you go to GitHub Tetris manifest image is updated automatically

```

diff --git deployment.yaml deployment.yaml
--- deployment.yaml
+++ deployment.yaml
@@ -15,7 +15,7 @@ spec:
 15   15     spec:
 16   16       containers:
 17   17         - name: tetris
 18 - 18           image: sevenajay/tetrisv1:latest
 18 + 18           image: sevenajay/tetrisv2:latest
 19   19         ports:
 20   20           - containerPort: 3000 # Use port 3000
 21   21

```

0 comments on commit 5b160ed

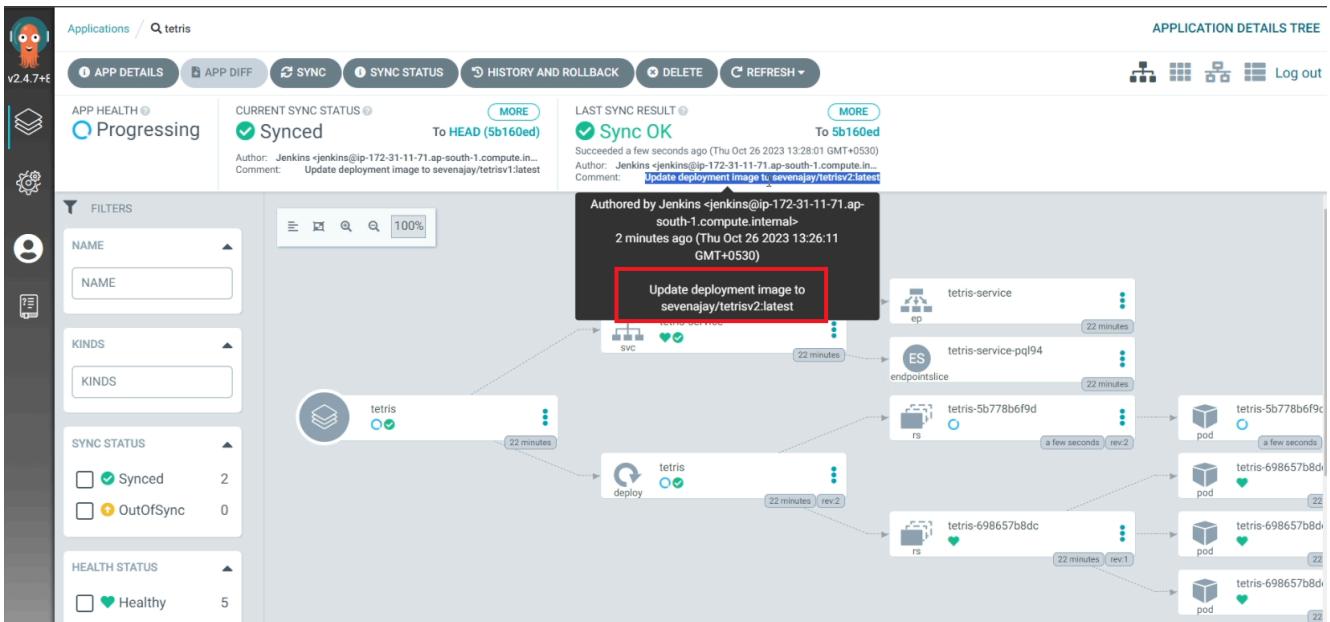
Code Blame 35 lines (33 loc) • 563 Bytes

```

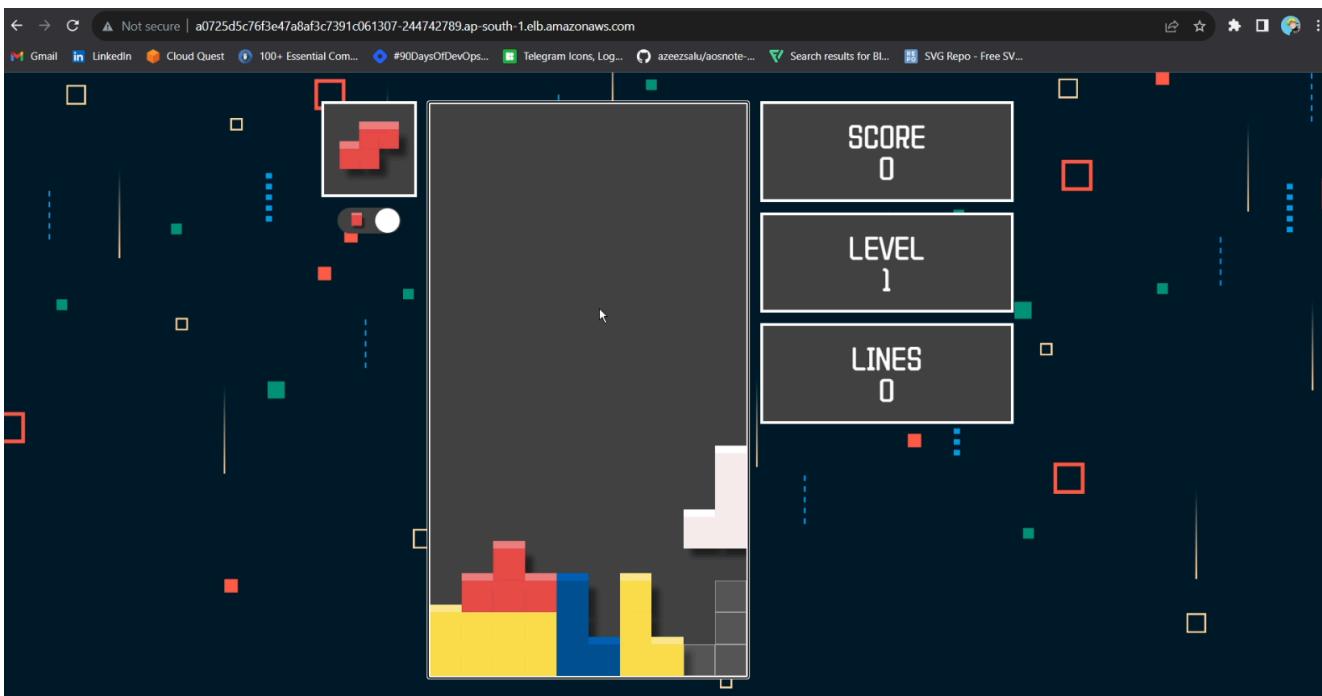
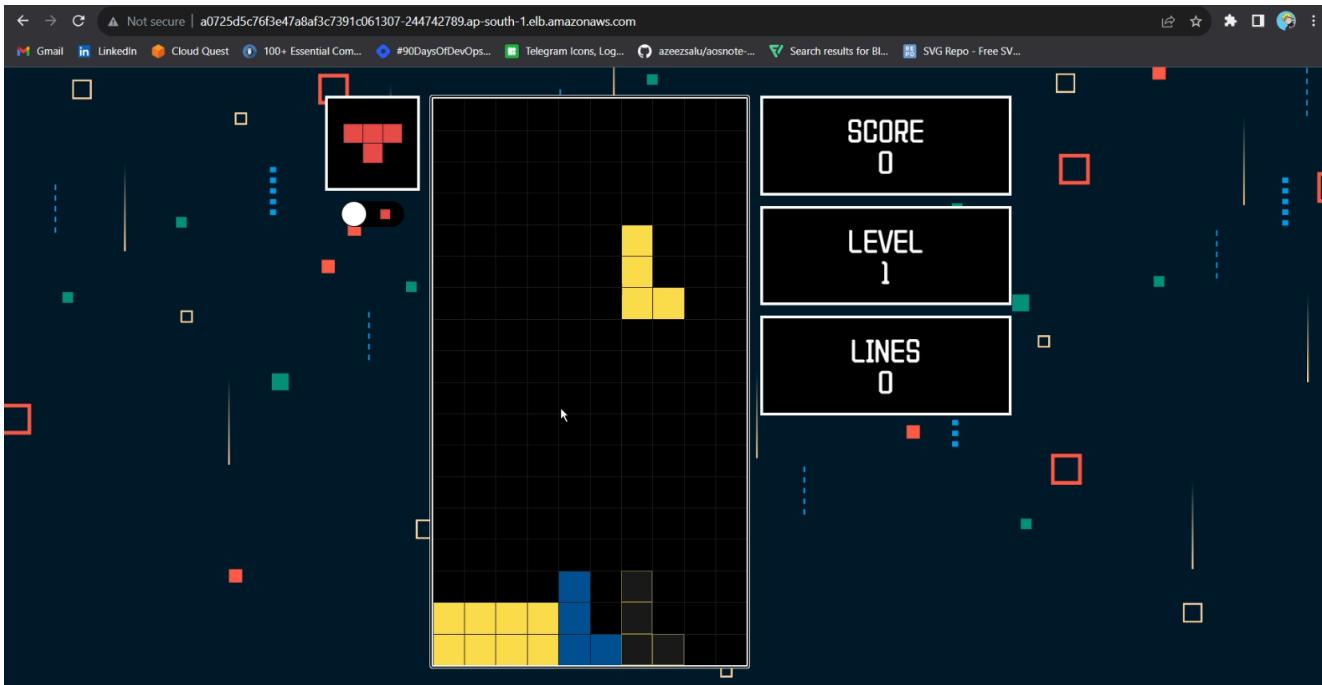
1  ---
2  apiVersion: apps/v1
3  kind: Deployment
4  metadata:
5    name: tetris
6  spec:
7    replicas: 3
8    selector:
9      matchLabels:
10     app: tetris
11    template:
12      metadata:
13        labels:
14          app: tetris
15      spec:
16        containers:
17        - name: tetris
18          image: sevenajay/tetrisv2:latest
19        ports:
20        - containerPort: 3000 # Use port 3000
21
22  ---

```

If you go to Argo CD Now it will automatically update the image and you can play the version 2 game



You can play version 2 in the browser now. Just refresh the old link



YOUTUBE VERSION OF SECOND TETRIS USED TRIGGER JOB

create a job for manifest (I mean image updater)

Enter an item name

manifest

» Required field

 **Freestyle project**
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

 **Maven project**
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **Multi-configuration project**

Add the below pipeline



```
pipeline{
    agent any
    environment {
        GIT_REPO_NAME = "Tetris-manifest"
        GIT_USER_NAME = "Aj7Ay"      # change your Github Username here
    }
    stages {
        stage('Checkout Code') {
            steps {
                git branch: 'main', url: 'https://github.com/Aj7Ay/Tetris-manifest'
            }
        }
        stage('Update Deployment File') {
            steps {
                script {
                    withCredentials([string(credentialsId: 'github', variable: 'GITHUB_TOKEN')])
                    NEW_IMAGE_NAME = "sevenajay/tetrisv2:latest"      # change your Docker image name here
                    sh "sed -i 's|image: .*|image: $NEW_IMAGE_NAME|' deployment.yaml"
                    sh 'git add deployment.yaml'
                    sh "git commit -m 'Update deployment image to $NEW_IMAGE_NAME'"
                    sh "git push https://${GITHUB_TOKEN}@github.com/${GIT_REPO_NAME}.git"
                }
            }
        }
    }
}
```

```

    }
}
```

Click on Apply and save

In the Tetris Job Use the below pipeline



```

pipeline{
    agent any
    tools{
        jdk 'jdk17'
        nodejs 'node16'
    }
    environment {
        SCANNER_HOME=tool 'sonar-scanner'
    }
    stages {
        stage('clean workspace'){
            steps{
                cleanWs()
            }
        }
        stage('Checkout from Git'){
            steps{
                git branch: 'main', url: 'https://github.com/Aj7Ay/TetrisVersion2.0'
            }
        }
        stage("Sonarqube Analysis"){
            steps{
                withSonarQubeEnv('sonar-server') {
                    sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=TetrisVersion2.0 -Dsonar.projectKey=TetrisVersion2.0 '''
                }
            }
        }
        stage("quality gate"){
            steps {
                script {
                    waitForQualityGate abortPipeline: false, credentials:
                }
            }
        }
    }
}
```


The last stage will trigger the Above manifest job and it will update the deployment file and update the latest image

Build History

S	W	Name ↓	Last Success	Last Failure	Last Duration
...	...	manifest	N/A	N/A	N/A
✓	...	Terraform-Eks	2 hr 24 min #1	N/A	9 min 49 sec
✓	...	TETRIS	34 min #6	42 min #5	8 min 29 sec

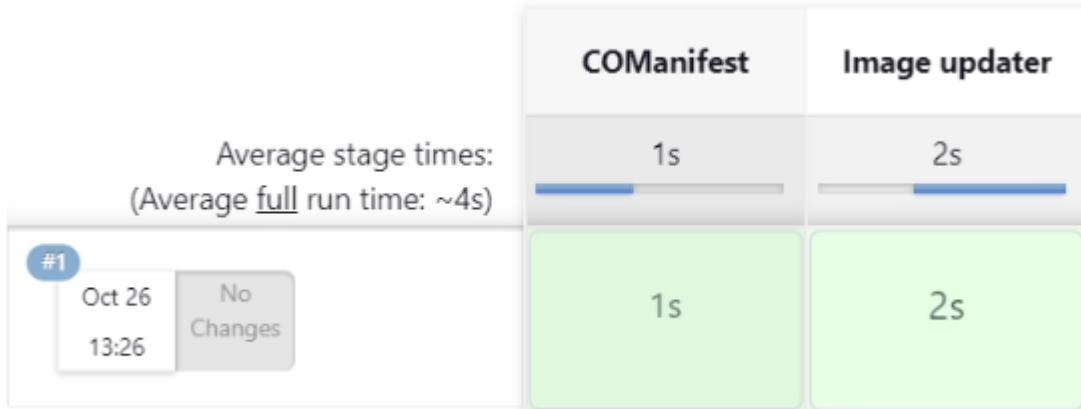
Build Queue: No builds in the queue.

Build Executor Status

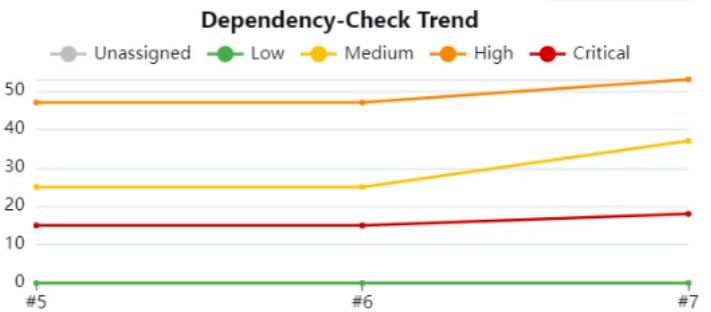
TETRIS TRIGGERS THE manifest job

REST API Jenkins 2.414.3

Manifest stage view



Tetris stage view



Declarative: Tool Install	CO	Sonar analysis	QG	NPM	TRIVY FS	OWASP FS SCAN	Docker build and push	TRIVY IMAGE	trigger manifest pipeline
349ms	2s	59s	604ms	39s	24s	2min 36s	2min 31s	2min 38s	11s
349ms	2s	59s	604ms (paused for 13s)	39s	24s	2min 36s	2min 31s	2min 38s	11s

Image updated

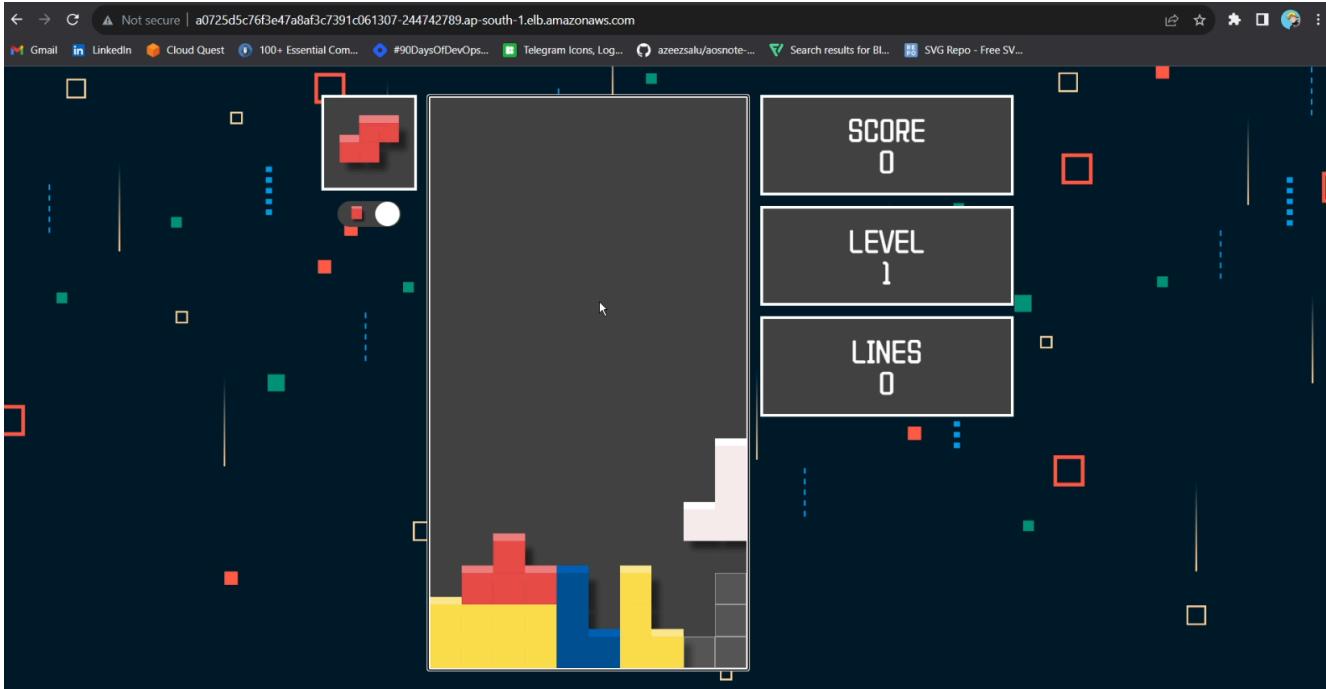
Code **Blame** 35 lines (33 loc) · 563 Bytes

```

1  ---
2  apiVersion: apps/v1
3  kind: Deployment
4  metadata:
5    name: tetris
6  spec:
7    replicas: 3
8    selector:
9      matchLabels:
10     app: tetris
11    template:
12      metadata:
13        labels:
14          app: tetris
15    spec:
16      containers:
17      - name: tetris
18        image: sevenajay/tetrisv2:latest
19        ports:
20        - containerPort: 3000  # Use port 3000
21
22  ---

```

Play Game in the browser



DESTRUCTION OF RESOURCES

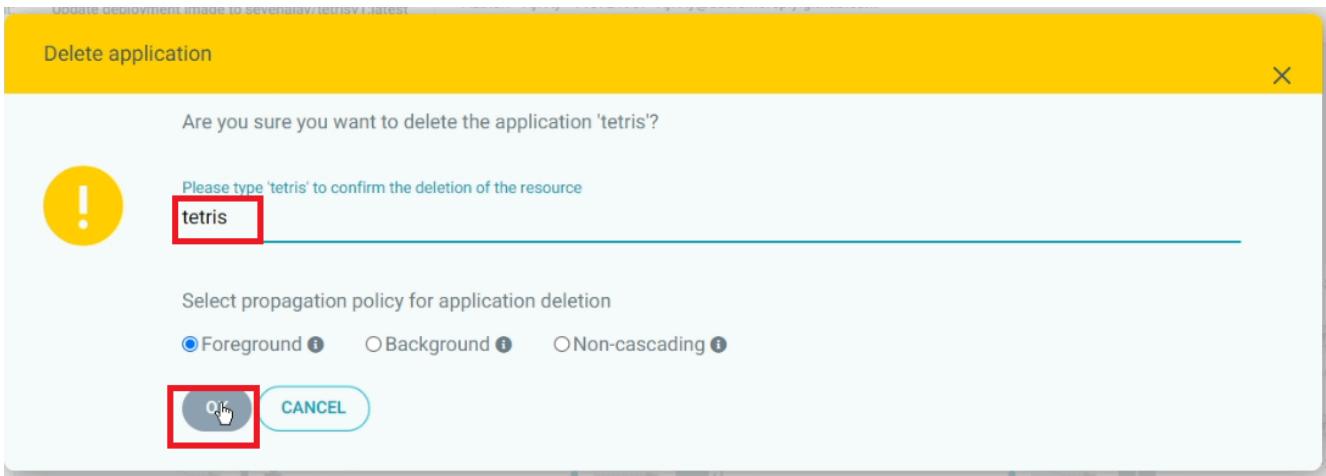
First, delete the app in ARGO CD

Go to Argo CD and click on Tetris App

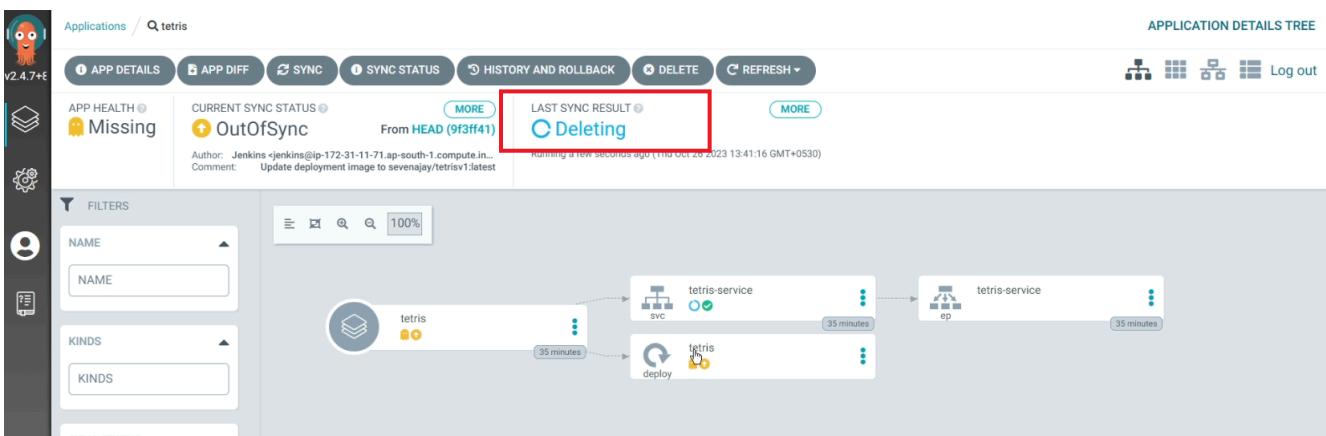
Click on Delete

APP HEALTH	CURRENT SYNC STATUS	LAST SYNC RESULT
Healthy	Synced To HEAD (9f3ff41)	Sync OK Succeeded 4 minutes ago (Thu Oct 26 2023 13:36:42 GMT+0530) Author: Aj7Ay <110721907+Aj7Ay@users.noreply.github.com> - Comment: Update deployment image to sevenajay/tetrisv1:latest

Now Provide the app name and click on OK



The app is deleted now



Now go to Putty and Remove Argo CD Service

```
kubectl delete svc argocd-server -n argocd
```

```
ubuntu@ip-172-31-11-71:~$ kubectl delete svc argocd-server -n argocd
service "argocd-server" deleted
```

Now you can see in the Aws console that load balancers will be deleted.

The screenshot shows the AWS EC2 Load Balancers page. At the top, there are navigation links: 'EC2 > Load balancers'. Below that is a search bar labeled 'Filter load balancers' and a button labeled 'Actions ▾'. A prominent orange button on the right says 'Create load balancer'. The main area has a header 'Load balancers' with a sub-instruction 'Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.' Below the header is a table with columns: Name, DNS name, State, VPC ID, Availability Zones, Type, and Date created. A search bar at the top of the table is empty. The message 'No resources to display' is centered below the table.

Now Go to Jenkins Dashboard and click on Terraform-Eks job

And build with parameters and destroy action

It will delete the EKS cluster that provisioned

The screenshot shows the Jenkins Pipeline Terraform-Eks job configuration. On the left, there's a sidebar with options: Status, Changes, Build with Parameters (which is selected and highlighted with a red box), Configure, Delete Pipeline, Full Stage View, and Rename. In the center, the title is 'Pipeline Terraform-Eks'. It says 'This build requires parameters:' and shows a dropdown menu for 'action' with 'destroy' selected (also highlighted with a red box). At the bottom, there are two buttons: 'Build' (highlighted with a red box) and 'Cancel'.

After 10 minutes cluster will delete and wait for it. Don't remove ec2 instance till that time.



Cluster deleted

The screenshot shows the ArgoCD interface with a header 'Clusters (0) Info'. Below it is a search bar with placeholder 'Filter clusters' and navigation arrows. A table has columns 'Cluster name', 'Status', 'Kubernetes version', and 'Provider'. A message 'No clusters' with the subtext 'You do not have any clusters.' is centered. A 'Create cluster' button is at the bottom.

Now go to Visual Studio code and delete the Ec2 instance



`terraform destroy --auto-approve`



```
PS C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform> terraform destroy --auto-approve
```

```
Plan: 0 to add, 0 to change, 5 to destroy.
aws_iam_role_policy_attachment.example_attachment: Destroying... [id=Jenkins-terraform-20]
aws_instance.web: Destroying... [id=i-0323f37f837248e53]
aws_iam_role_policy_attachment.example_attachment: Destruction complete after 1s
aws_instance.web: Still destroying... [id=i-0323f37f837248e53, 11s elapsed]
aws_instance.web: Still destroying... [id=i-0323f37f837248e53, 21s elapsed]
aws_instance.web: Still destroying... [id=i-0323f37f837248e53, 31s elapsed]
aws_instance.web: Destruction complete after 31s
aws_iam_instance_profile.example_profile: Destroying... [id=Jenkins-terraform]
aws_security_group.Jenkins-sg: Destroying... [id=sg-086b7a68255d23e19]
aws_iam_instance_profile.example_profile: Destruction complete after 1s
aws_iam_role.example_role: Destroying... [id=Jenkins-terraform]
aws_security_group.Jenkins-sg: Destruction complete after 1s
aws_iam_role.example_role: Destruction complete after 0s
```

Destroy complete! Resources: 5 destroyed.

```
PS C:\Users\Admin\Desktop\PROJECT\cf2tf\tetris-react-js\Jenkins-terraform>
```

Every resource is deleted.

In this journey through the world of DevSecOps and automated deployments, we've explored the remarkable capabilities of ArgoCD, Terraform, and Jenkins. From the initial infrastructure setup to deploying not one but two versions of the beloved Tetris game, we've witnessed how these tools can work harmoniously to create a streamlined and secure DevOps pipeline.

The power of automation is not just a convenience; it's a necessity in today's fast-paced development landscape. With the right tools and practices, you can save time, reduce errors, and ensure the highest standards of security in your deployments. This blog has been a testament to the incredible possibilities that arise when you combine the art of DevOps with the precision of modern cloud-native technologies.

As you embark on your own DevSecOps journey, we hope the insights shared here inspire you to explore, innovate, and continually strive for excellence in your deployment workflows. Remember that the world of technology is ever-evolving, and your willingness to adapt and embrace automation will be a key driver of your success.

Thank you for joining us on this adventure. If you found this blog valuable, please share it with your fellow tech enthusiasts, and stay tuned for more exciting content. Until next time, happy automating and deploying!



Ajay Kumar Yegireddi is a DevSecOps Engineer and System Administrator, with a passion for sharing real-world DevSecOps projects and tasks. **Mr. Cloud Book**, provides hands-on tutorials and practical insights to help others master DevSecOps tools and workflows. Content is designed to bridge the gap between development, security, and operations, making complex concepts easy to understand for both beginners and professionals.

Comments

6 responses to “Automating Tetris Deployments: DevSecOps with ArgoCD, Terraform, and Jenkins for Two Game Versions”



Jay

10 January 2024

I would like to know the entire AWS monthly cost roughly for this devsecops automation?

[Reply](#)**mrcloudbook.com**

10 January 2024

For all projects or for one project

[Reply](#)**Hamza**

18 March 2024

for this one project

[Reply](#)**Abhinandan Vishwakarma**

22 January 2024

keep doing sir it's Helping a more and more students which one is not affordable to pay DevOps Engineer Course Fees thank u so much for your this contribution.

[Reply](#)**mrcloudbook.com**

23 January 2024

Definitely

[Reply](#)**Saketh**

20 March 2024

HELLO MRcloudbook , I have created jenkins-server instance , and then using above steps YOU mentioned I created a terraform pipeline to create the EKS-cluster , when I ran the pipeline it destroyed my jenkins instance. so then in vs code I done the terraform plan of Eks cluster , it showed .. {Plan: 8 to add, 0 to change, 5 to destroy }.

[Reply](#)

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Website

Save my name, email, and website in this browser for the next time I comment.

I'm not a robot

reCAPTCHA
[Privacy](#) - [Terms](#)

[Post Comment](#)

Uncategorized

Day -1: Kick Off Cloud Security with AWS Registration

18 August 2025

Uncategorized

How to Automate Incident Response : How Q Developer Helped Me Automate a Daily Pain Point

22 July 2025

AI

How to Run Docker Model Runner on Ubuntu 24.04

11 July 2025

Upskill with Ajay: DevSecOps Mastery

Join Mr Cloud book to master DevSecOps through real-world projects. Learn CI/CD, security integration, automation, and more, gaining hands-on skills for industry-level challenges.



Important Links

[Privacy Policy](#)

[Terms & Conditions](#)

[Contact](#)

Resources

[Blog](#)

[YouTube Channel](#)

