

## Mr Cloud Book

Home Blog DevSecOps Contact About Me Testimonials

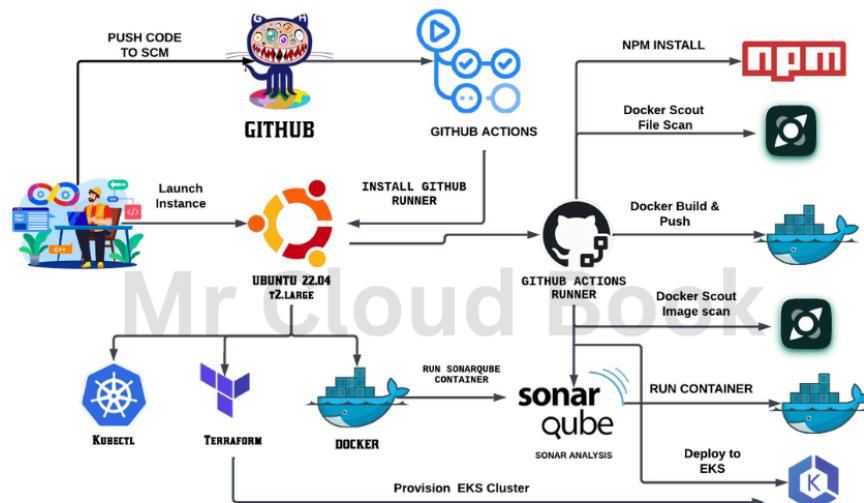
Search Blogs

DevOps

# GitHub Actions Scouting Myntra App | DevSecOps



mrcloobook.com · 21 January 2024

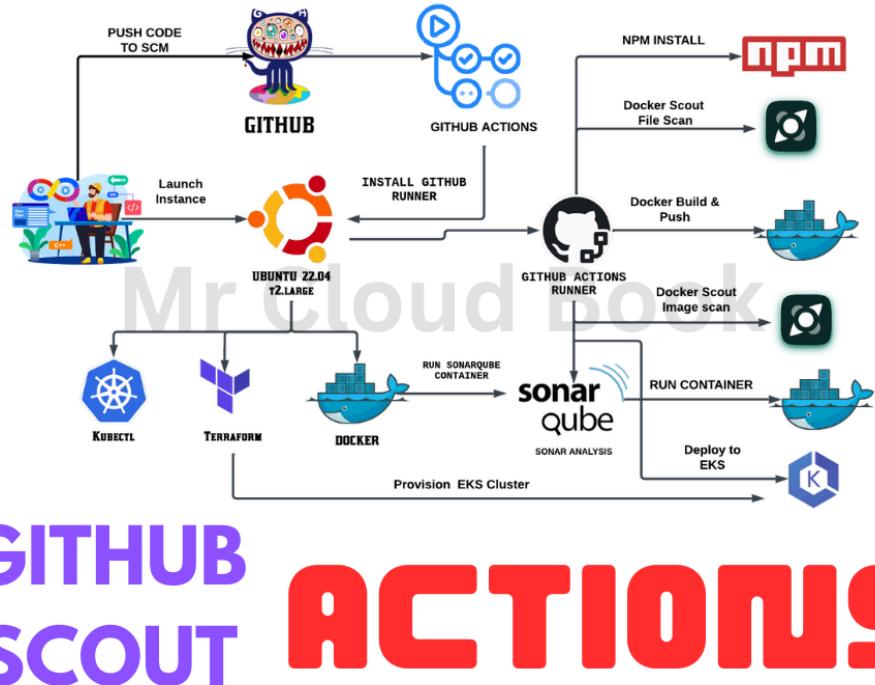


**GITHUB SCOUT ACTIONS**

In today's fast-paced world of software development, automation is the name of the game. GitHub Actions is the ace up the sleeve of modern developers, enabling them to streamline their daily workflows in practical and impactful ways. In this article, we'll explore how GitHub Actions is making a real difference in real-life scenarios.

From Continuous Integration (CI) and Continuous Deployment (CD) to code quality assurance and security scanning, GitHub Actions brings automation to every

aspect of the development process. With custom workflows, enhanced collaboration, and release management, this tool empowers developers to be more efficient, reliable, and productive. Discover how GitHub Actions is not just a concept but a transformative solution in the daily lives of developers.



GitHub: <https://github.com/Aj7Ay/Myntra-Clone.git>

## Contents [ hide ]

[STEP 1A: Setting up AWS EC2 Instance and IAM Role](#)

[STEP 1B: IAM ROLE](#)

## STEP 1A: Setting up AWS EC2 Instance and IAM Role

- 1. Sign in to the AWS Management Console:** Access the AWS Management Console using your credentials
- 2. Navigate to the EC2 Dashboard:** Click on the “Services” menu at the top of the page and select “EC2” under the “Compute” section. This will take you to the EC2 Dashboard.
- 3. Launch Instance:** Click on the “Instances” link on the left sidebar and then click the “Launch Instance” button.

**4. Choose an Amazon Machine Image (AMI):** In the “Step 1: Choose an Amazon Machine Image (AMI)” section:

- Select “AWS Marketplace” from the left-hand sidebar.
- Search for “Ubuntu” in the search bar and choose the desired Ubuntu AMI (e.g., Ubuntu Server 22.04 LTS).
- Click on “Select” to proceed.

**5. Choose an Instance Type:** In the “Step 2: Choose an Instance Type” section:

- Scroll through the instance types and select “t2.large” from the list.
- Click on “Next: Configure Instance Details” at the bottom.

**6. Configure Instance Details:** In the “Step 3: Configure Instance Details” section, you can leave most settings as default for now. However, you can configure settings like the network, subnet, IAM role, etc., according to your requirements.

- Once done, click on “Next: Add Storage.”

**7. Add Storage:** In the “Step 4: Add Storage” section:

- You can set the size of the root volume (usually /dev/sda1) to 30 GB by specifying the desired size in the “Size (GiB)” field.
- Customize other storage settings if needed.
- Click on “Next: Add Tags” when finished.

**8. Add Tags (Optional):** In the “Step 5: Add Tags” section, you can add tags to your instance for better identification and management. This step is optional but recommended for organizational purposes.

- Click on “Next: Configure Security Group” when done.

**9. Configure Security Group:** In the “Step 6: Configure Security Group” section:

- Create a new security group or select an existing one.
- Ensure that at least SSH (port 22) is open for inbound traffic to allow remote access.
- You might also want to open other ports as needed for your application’s requirements.

- Click on “Review and Launch” when finished.

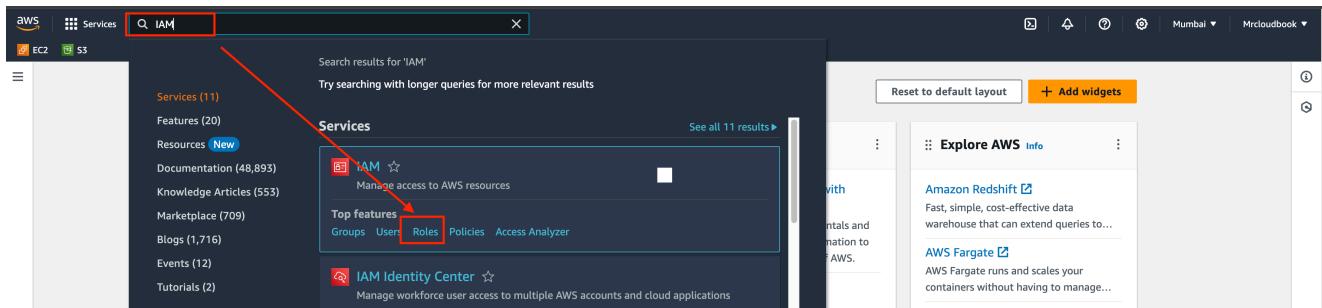
**10. Review and Launch:** Review the configuration details of your instance. If everything looks good:

- Click on “Launch” to proceed.
- A pop-up will prompt you to select or create a key pair. Choose an existing key pair or create a new one.
- Finally, click on “Launch Instances.”

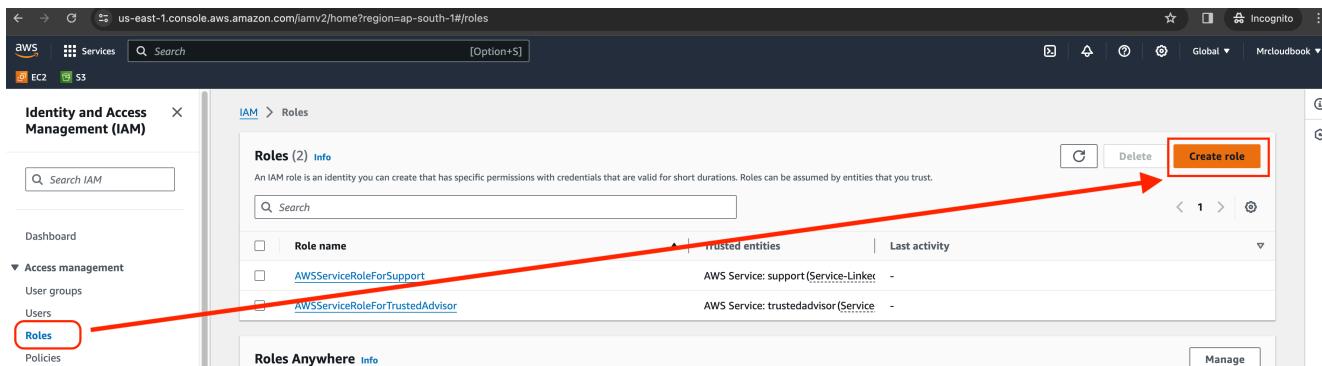
**11. Accessing the Instance:** Once the instance is launched, you can connect to it using SSH. Use the private key associated with the selected key pair to connect to the instance’s public IP or DNS address.

## STEP 1B: IAM ROLE

Search for IAM in the search bar of AWS and click on roles.



Click on Create Role



Select entity type as AWS service

Use case as EC2 and click on Next.

**Trusted entity type**

- AWS service**  
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- AWS account**  
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- SAML 2.0 federation**  
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- Custom trust policy**  
Create a custom trust policy to enable others to perform actions in this account.

**Use case**  
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

**Service or use case**  
EC2

Choose a use case for the specified service.  
Use case

- EC2**  
Allows EC2 instances to call AWS services on your behalf.
- EC2 Role for AWS Systems Manager**  
Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.

For permission policy select Administrator Access (Just for learning purpose), click Next.

**Add permissions**

**Permissions policies (1/897) Info**

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

Policy name	Type	Description
<input checked="" type="checkbox"/> <b>AdministratorAccess</b>	AWS managed - job function	Provides full access to AWS services an...
<input type="checkbox"/> <b>AdministratorAccess-Amplify</b>	AWS managed	Grants account administrative permis...

Provide a Name for Role and click on Create role.

**Role details**

**Role name**  
Enter a meaningful name to identify this role.  
**MARIO**

**Description**  
Add a short explanation for this role.  
Allows EC2 instances to call AWS services on your behalf.

**Step 1: Select trusted entities**

**Trust policy**

```

1  {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": [
7         "sts:AssumeRole"
8       ],
9       "Principal": {
10         "Service": [
11           "ec2.amazonaws.com"
12         ]
13       }
14     }
15   ]
16 }
```

Role is created.

The screenshot shows the AWS IAM Roles page. A green banner at the top says "Role MARIO created." The table lists four roles: "AWSServiceRoleForSupport", "AWSServiceRoleForTrustedAdvisor", and "MARIO", which is highlighted with a red box. The "Create role" button is visible at the top right.

Now Attach this role to Ec2 instance that we created earlier, so we can provision cluster from that instance.

Go to EC2 Dashboard and select the instance.

Click on Actions -> Security -> Modify IAM role.

The screenshot shows the AWS EC2 Instances page. An instance named "Test" is selected. In the Actions menu, the "Security" option is expanded, and the "Modify IAM role" option is highlighted with a red box. A red arrow points from the "Modify IAM role" option to the "Actions" menu.

Select the Role that created earlier and click on Update IAM role.

The screenshot shows the "Modify IAM role" dialog for the "Test" instance. The "MARIO" role is selected in the dropdown menu. The "Update IAM role" button is highlighted with a blue box.

Connect the instance to Mobaxtreme or Putty.

## STEP 2: INSTALL REQUIRED PACKAGES ON INSTANCE

Create Docker & Docker scout Installation script

```
vi docker-setup.sh
```



## Script



```
# Add Docker's official GPG key:  
sudo apt-get update  
sudo apt-get install ca-certificates curl gnupg  
sudo install -m 0755 -d /etc/apt/keyrings  
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --de  
sudo chmod a+r /etc/apt/keyrings/docker.gpg  
  
# Add the repository to Apt sources:  
echo \  
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/d  
  $./etc/os-release && echo "$VERSION_CODENAME") stable" | \  
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null  
sudo apt-get update  
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-build  
curl -sSfL https://raw.githubusercontent.com/docker/scout-cli/main/insta  
sudo usermod -aG docker ubuntu  
newgrp docker  
sudo chmod 777 /var/run/docker.sock
```



## Provide executable versions



```
sudo chmod 777 docker-setup.sh  
sh docker-setup.sh
```



## Create Script for other packages



```
vi script-packages.sh
```



## Script



```
#!/bin/bash

sudo apt update -y
sudo touch /etc/apt/keyrings/adoptium.asc
sudo wget -O /etc/apt/keyrings/adoptium.asc https://packages.adoptium.net/maven3/gpg.key
echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/maven3/ adoptium-temurin-jdk main"
sudo apt update -y
sudo apt install temurin-17-jdk -y
/usr/bin/java --version

# 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) hashicorp main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update && sudo apt install terraform

# Install kubectl
sudo apt update
sudo apt install curl -y
curl -LO https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)
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

# Install Node.js 16 and npm
curl -fsSL https://deb.nodesource.com/gpgkey/nodesource.gpg.key | sudo gpg --dearmor -o /usr/share/keyrings/nodesource-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/nodesource-archive-keyring.gpg] https://deb.nodesource.com/node/16.x $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/nodesource.list
sudo apt update
```

```
sudo apt update
sudo apt install -y nodejs
```

```
#!/bin/bash
sudo apt update -
sudo touch /etc/apt/keyrings/adoptium.asc
sudo wget -O /etc/apt/keyrings/adoptium.asc https://packages.adoptium.net/artifactory/api/gpg/key/public
echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/artifactory/deb $(awk -F='^VERSION_CODENAME/{print$2}' /etc/os-release) main" | sudo tee /etc/apt/sources.list.d/adoptium.list
sudo apt update -
sudo apt install temurin-17-jdk -y
/usr/bin/java --version
# Install Trivy
sudo apt-get install wget apt-transport-https gnupg lsb-release -
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee /usr/share/keyrings/trivy.gpg > /dev/null
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb $(lsb_release -sc) main" | sudo tee -a /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 $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update && sudo apt install terraform
# Install kubectl
sudo apt update
sudo apt install curl -y
curl -L0 https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl
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
# Install Node.js 16 and npm
curl -fsSL https://deb.nodesource.com/gpgkey/nodesource.gpg.key | sudo gpg --dearmor -o /usr/share/keyrings/nodesource-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/nodesource-archive-keyring.gpg] https://deb.nodesource.com/node_16.x focal main" | sudo tee /etc/apt/sources.list.d/nodesource.list
sudo apt update
sudo apt install -y nodejs
~
```

Give executable permissions for script and run it



```
sudo chmod 777 script-packages.sh
sh script-packages.sh
```



```
ubuntu@ip-172-31-12-220:~$
ubuntu@ip-172-31-12-220:~$ vi script.sh
ubuntu@ip-172-31-12-220:~$
ubuntu@ip-172-31-12-220:~$ sudo chmod 777 script.sh
ubuntu@ip-172-31-12-220:~$
```

Hit.I http://ip-172-31-12-220:~\$



Check package versions



```
docker --version
terraform --version
```

```
aws --version  
kubectl version  
node -v  
java --version
```

```
ubuntu@ip-172-31-12-220:~$  
ubuntu@ip-172-31-12-220:~$ terraform --version  
Terraform v1.6.6  
on linux_amd64  
ubuntu@ip-172-31-12-220:~$  
ubuntu@ip-172-31-12-220:~$ aws --version  
aws-cli/2.15.8 Python/3.11.6 Linux/6.2.0-1017-aws exe/x86_64.ubuntu.22 prompt/off  
ubuntu@ip-172-31-12-220:~$  
ubuntu@ip-172-31-12-220:~$ kubectl version --client  
Client Version: v1.29.0  
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3  
ubuntu@ip-172-31-12-220:~$  
ubuntu@ip-172-31-12-220:~$ java --version  
openjdk 17.0.9 2023-10-17  
OpenJDK Runtime Environment Temurin-17.0.9+9 (build 17.0.9+9)  
OpenJDK 64-Bit Server VM Temurin-17.0.9+9 (build 17.0.9+9, mixed mode, sharing)  
ubuntu@ip-172-31-12-220:~$
```

Run the Sonarqube container

```
docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
```

```
ubuntu@ip-172-31-12-220:~$  
ubuntu@ip-172-31-12-220:~$ docker run -d --name sonar -p 9000:9000 sonarqube:lts-community  
Unable to find image 'sonarqube:lts-community' locally  
lts-community: Pulling from library/sonarqube  
3dd181f9be59: Pull complete  
0f838805bddf: Pull complete  
e7eee5bc80e6: Pull complete  
51526e7965d8: Pull complete  
ffcdc7c6c160: Pull complete  
9d141c530e5b: Pull complete  
bb9af13b2efe: Pull complete  
Digest: sha256:49ac473fc9da07052cdd205e4581a5b369adeaf65832830d62be86e419ea2e1f  
Status: Downloaded newer image for sonarqube:lts-community  
55667a4356161a07bd70dc432181869c7d3a7bd3b0391fb220ee2a0058751f2  
ubuntu@ip-172-31-12-220:~$
```

Now copy the IP address of the ec2 instance



ec2-public-ip:9000



The screenshot shows a browser window with three tabs at the top: 'Instances | EC2 | ap-south-1', 'petstore Config [Jenkins]', and 'SonarQube'. The main content area displays a SonarQube login form with the title 'Log in to SonarQube'. It contains two input fields: 'Login' and 'Password', and two buttons: 'Log in' and 'Cancel'. The URL in the address bar is 52.66.140.95:9000/sessions/new?return\_to=%2F.

Provide Login and password



```
login admin  
password admin
```



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 your Sonarqube password & This is the Sonarqube dashboard

Not secure | 52.66.140.95:9000/projects/create

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

Search for projects...

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 Set up global configuration	From Bitbucket Server Set up global configuration	From Bitbucket Cloud Set up global configuration	From GitHub Set up global configuration	From GitLab Set up global configuration
--	--	---	--	--

## Step 3: Integrating SonarQube with GitHub Actions

Integrating SonarQube with GitHub Actions allows you to automatically analyze your code for quality and security as part of your continuous integration pipeline.

We already have Sonarqube up and running

## On Sonarqube Dashboard click on Manually

The screenshot shows the Sonarqube 'Projects' creation interface. At the top, there's a header bar with links like 'Gmail', 'LinkedIn', 'Cloud Quest', '100+ Essential Com...', '#90DaysOfDevOps...', 'Telegram Icons, Log...', 'azeezsalu/aosnote...', 'Search results for Bl...', 'SVG Repo - Free SV...', and a search bar. Below the header, the main content area has a heading 'How do you want to create your project?'. It explains that users can benefit from SonarQube's features by creating a project from their favorite DevOps platform. It lists five automated import options: 'From Azure DevOps', 'From Bitbucket Server', 'From Bitbucket Cloud', 'From GitHub', and 'From GitLab', each with a 'Set up global configuration' link. Below these, a section titled 'Are you just testing or have an advanced use-case? Create a project manually.' contains a button labeled 'Manually' with a double-angle bracket icon. This 'Manually' button is highlighted with a red rectangular box.

Next, provide a name for your project and provide a Branch name and click on setup

# Create a project

All fields marked with \* are required

## Project display name \*



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

## Project key \*



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

## Main branch name \*

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

On the next page click on With GitHub actions

The screenshot shows the SonarQube interface for a 'Netflix' project. At the top, there are navigation links: Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and a search bar. Below the header, the repository name 'Netflix' is shown with a star icon and a 'main' branch indicator. The main content area has tabs for Overview, Issues, Security Hotspots, Measures, Code, and Activity. A 'Project Settings' dropdown and a 'Project Information' link are also present. The 'Overview' tab is selected. A section titled 'How do you want to analyze your repository?' asks if the user wants to integrate with CI. It provides options for Jenkins, GitHub Actions, Bitbucket Pipelines, GitLab CI, Azure Pipelines, and Other CI. The 'With GitHub Actions' option is highlighted with a red box. At the bottom, a note says 'Are you just testing or have an advanced use-case? Analyze your project locally.'

This will Generate an overview of the Project and provide some instructions to integrate

**1 Create GitHub Secrets**

In your GitHub repository, go to **Settings > Secrets** and create two new secrets:

- Click on **New repository secret**.
- In the **Name** field, enter `SONAR_TOKEN`
- In the **Value** field, enter an existing token, or a newly generated one:
- Click on **Add secret**.

- Click on **New repository secret**.
- In the **Name** field, enter `SONAR_HOST_URL`
- In the **Value** field, enter `http://13.233.58.37:9000`
- Click on **Add secret**.

**Continue**

Let's Open your GitHub and select your Repository

In my case it is Myntra-clone and Click on Settings

The screenshot shows the GitHub repository 'Myntra-Clone' (Public). The 'Settings' tab is highlighted with a red circle. The repository has 1 Branch and 0 Tags. The code section shows three commits from 'Ajay Kumar Yegireddi' and 'Ajay Kumar Yegireddi' (feat: Dockerfile) made 19 minutes ago, and two commits from 'Ajay Kumar Yegireddi' (feat: Added myn files) made 21 minutes ago.

Commit	Message	Time
Ajay Kumar Yegireddi and Ajay Kumar Yegireddi (feat: Dockerfile)	feat: Dockerfile	d10a3f2 · 19 minutes ago
Ajay Kumar Yegireddi (feat: Added myn files)	feat: Added myn files	21 minutes ago
Ajay Kumar Yegireddi (feat: Added myn files)	feat: Added myn files	21 minutes ago

Search for Secrets and variables and click on and again click on actions

The screenshot shows the GitHub repository settings page for 'Mynta-Clone'. The left sidebar has sections like Collaborators, Moderation options, Code and automation, Branches, Tags, Rules, Actions, Webhooks, Environments, Codespaces, and Pages. The 'Actions' section is highlighted with a red box. The main area shows settings for the 'Default branch' (set to 'main') and a 'Social preview' section where users can upload an image. The 'Secrets and variables' section is also highlighted with a red box.

It will open a page like this click on New Repository secret

The screenshot shows the 'Repository secrets' page for the same repository. The left sidebar shows 'Actions' selected. The main area displays a message: 'This repository has no secrets.' with a prominent green 'New repository secret' button highlighted with a red box.

Now go back to Your Sonarqube Dashboard

Copy SONAR\_TOKEN and click on Generate Token

The screenshot shows the Sonarqube dashboard for the 'Netflix' project. The top navigation bar includes 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', and a search bar. The main content area is titled 'Create GitHub Secrets' and provides instructions for generating tokens:

1. Click on **New repository secret**.
2. In the **Name** field, enter `SONAR_TOKEN` and click **COPY THIS ONE**.
3. In the **Value** field, enter an existing token or a newly generated one, and click **Generate a token**.
4. Click on **Add secret**.

Below these steps, there are additional instructions for generating tokens for the 'SONAR\_HOST\_URL' secret.

Click on Generate

**Generate a project token**

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

<b>Token name</b>	<b>Expires in</b>
Analyze "myntra"	30 days

**Generate**  

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

**Continue**

Let's copy the Token and add it to GitHub secrets

**Generate a project token**

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

Analyze "Netflix": **sqp\_0fc59dfc0eef5378e95d8aebae06134b34bbe55** Copy Delete

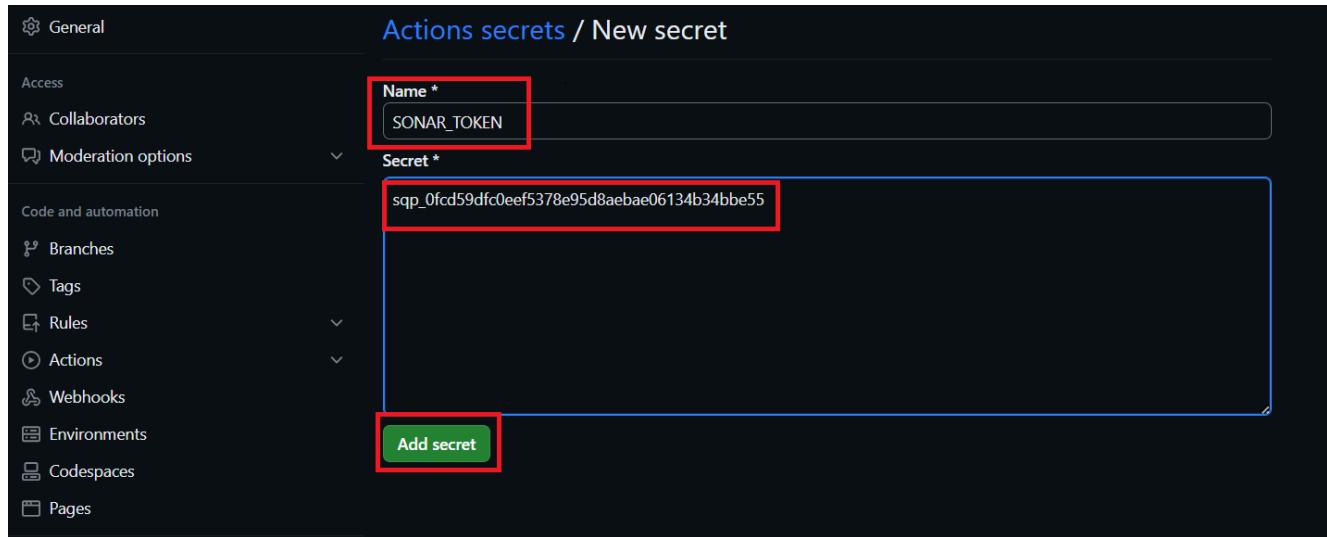
**!** New token "sqp\_0fc59dfc0eef5378e95d8aebae06134b34bbe55" has been created. Make sure you copy it now, you won't be able to see it again!

**Continue**

Now go back to GitHub and Paste the copied name for the secret and token

Name: SONAR\_TOKEN

Secret: Paste Your Token and click on Add secret



Now go back to the Sonarqube Dashboard

Copy the Name and Value

**1 Create GitHub Secrets**

In your GitHub repository, go to **Settings > Secrets** and create two new secrets:

- Click on **New repository secret**.
- In the **Name** field, enter **SONAR\_TOKEN**
- In the **Value** field, enter an existing token, or a newly generated one:
- Click on **Add secret**.

---

- Click on **New repository secret**.
- In the **Name** field, enter **SONAR\_HOST\_URL**
- In the **Value** field, enter **http://13.126.202.56:9000**
- Click on **Add secret**.

**Continue**

Go to GitHub now and paste-like this and click on add secret

The screenshot shows the 'Actions secrets / New secret' page in GitHub. On the left, a sidebar lists 'General', 'Access', 'Collaborators', 'Moderation options', 'Code and automation', 'Branches', 'Tags', 'Rules', 'Actions', 'Webhooks', 'Environments', 'Codespaces', and 'Pages'. The main area has fields for 'Name \*' (set to 'SONAR\_HOST\_URL') and 'Secret \*' (set to 'http://13.126.202.56:9000'). A green 'Add secret' button at the bottom is highlighted with a red box.

Our Sonarqube secrets are added and you can see

The screenshot shows the 'Repository secrets' page in GitHub. It lists two secrets: 'SONAR\_HOST\_URL' and 'SONAR\_TOKEN', both updated 'now'. Each secret has edit and delete icons next to it.

Go to Sonarqube Dashboard and click on continue

The screenshot shows the 'Create GitHub Secrets' step 1 in Sonarqube. It provides instructions to create secrets in GitHub: 1. Click on 'New repository secret', 2. Enter 'SONAR\_TOKEN' in the Name field, 3. Enter an existing token or generate one in the Value field, and 4. Click on 'Add secret'. Below this, it shows steps for the second secret: 1. Click on 'New repository secret', 2. Enter 'SONAR\_HOST\_URL' in the Name field, 3. Enter 'http://13.126.202.56:9000' in the Value field, and 4. Click on 'Add secret'. A red box highlights the 'Continue' button at the bottom.

Now create your Workflow for your Project. In my case, the Netflix project is built using React Js. That's why I am selecting Other

1 Create GitHub Secrets

2 Create Workflow YAML File

1. What option best describes your build?

Maven Gradle .NET Other (for JS, TS, Go, Python, PHP, ...)

IN MY CASE I AM USING REACT JS

3 You're all set!

Now it Generates and workflow for my Project

(Use your files for this block please)

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

Netflix main

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

2. Create a `.sonar-project.properties` file in your repository and paste the following code:

```
sonar.projectKey=Netflix
```

**Copy**

3. Create or update your `.github/workflows/build.yml` YAML file with the following content:

```
name: Build

on:
  push:
    branches:
      - main

jobs:
  build:
    name: Build
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
        with:
          fetch-depth: 0 # Shallow clones should be disabled for a better relevancy of analysis
      - uses: sonarsource/sonarqube-scan-action@master
        env:
          SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
          SONAR_HOST_URL: ${{ secrets.SONAR_HOST_URL }}
      # If you wish to fail your job when the Quality Gate is red, uncomment the
      # following lines. This would typically be used to fail a deployment.
      # - uses: sonarsource/sonarqube-quality-gate-action@master
      #   with:
      #     qualitygate: sonarqube-quality-gate
```

**Copy**

Go back to GitHub. click on Add file and then create a new file

Mynta-Clone Public

main 1 Branch 0 Tags

Go to file Add file Code

Ajay Kumar Yegireddi and Ajay Kumar Yegireddi feat: Dockerfile

+ Create new file

Upload files

public feat: Added myn files 24 minutes ago

Go back to the Sonarqube dashboard and copy the file name and content

2. Create a **sonar-project.properties** file in your repository and paste the following code:

```
sonar.projectKey=myntra
```

 Copy

Here file name (in my case only )



sonar-project.properties



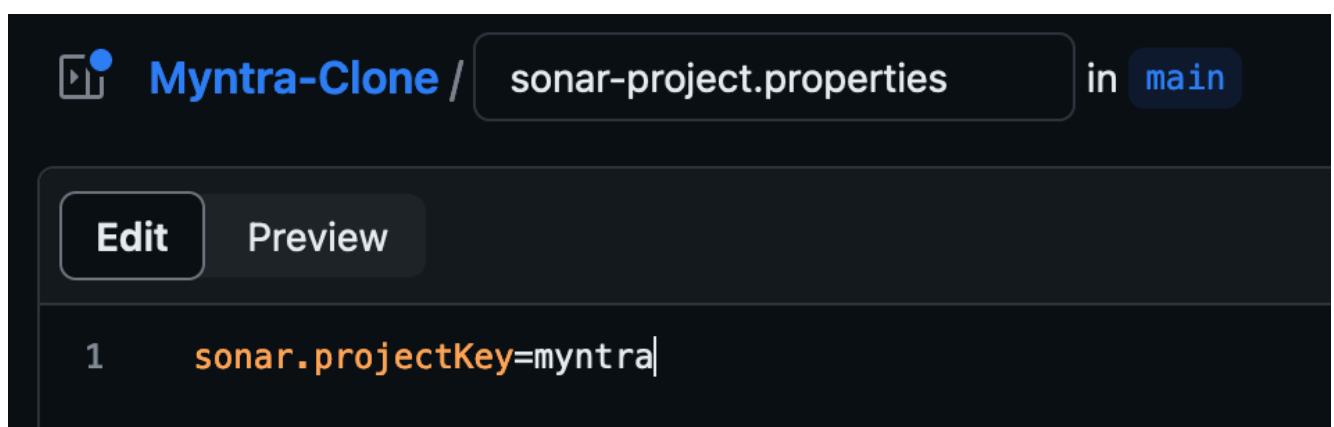
The content to add to the file is (copied from the above image)



sonar.projectKey=myntra



Add in GitHub like this (sample images)



The screenshot shows a GitHub repository named "Myntra-Clone". Inside the repository, there is a file named "sonar-project.properties" located in the "main" branch. The file content is displayed as follows:

```
1 sonar.projectKey=myntra
```

The "Edit" button is visible on the left side of the file view.

Commit changes

Let's add our workflow

To do that click on Add file and then click on Create a new file

A screenshot of a GitHub repository page for 'Myntra-Clone'. The repository is public and has 1 branch and 0 tags. The main branch is selected. A search bar at the top right says 'Go to file'. Below it, there's a 'Add file' button with a dropdown menu showing 'Create new file' and 'Upload files', both highlighted with red boxes. A commit message 'feat: Added myn files' is visible, along with a timestamp '24 minutes ago'.

Here is the file name

.github/workflows/build.yml #you can use any name iam using sonar.yml

A screenshot of the SonarQube interface for the 'Netflix' project. The 'Overview' tab is selected. A step 3 instruction says 'Create or update your .github/workflows/build.yml YAML file with the following content:'. The code shown is:

```

name: Build

on:
  push:
    branches:
      - main

jobs:
  build:
    name: Build
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - with:
          fetch-depth: 0 # Shallow clones should be disabled for a better relevancy of analysis
      - uses: sonarsource/sonarqube-scan-action@master
      - env:
          SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
          SONAR_HOST_URL: ${{ secrets.SONAR_HOST_URL }}
      # If you wish to fail your job when the Quality Gate is red, uncomment the
      # following lines. This would typically be used to fail a deployment.
      # - uses: sonarsource/sonarqube-quality-gate-action@master
      #   timeout-minutes: 5
      #   env:
      #     SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}

```

A 'Copy' button is highlighted with a red box on the right side of the code editor.

Copy content and add it to the file

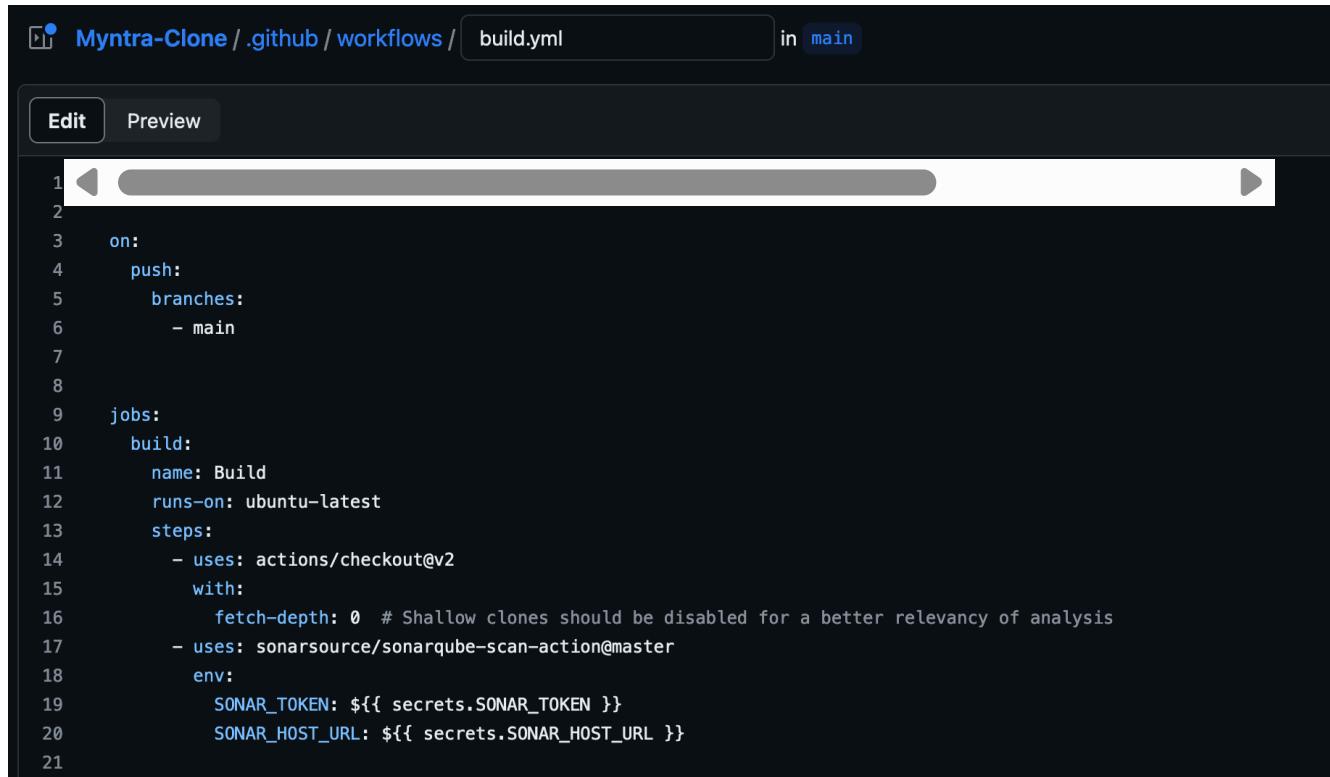
name: Build,Analyze,scan

```

on:
  push:
    branches:
      - main

```

```
jobs:
  build-analyze-scan:
    name: Build
    runs-on: [self-hosted]
    steps:
      - name: Checkout code
        uses: actions/checkout@v2
        with:
          fetch-depth: 0 # Shallow clones should be disabled for a better relevancy of analysis
      - name: Build and analyze with SonarQube
        uses: sonarsource/sonarqube-scan-action@master
        env:
          SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
          SONAR_HOST_URL: ${{ secrets.SONAR_HOST_URL }}
```



The screenshot shows the GitHub Actions workflow configuration for the 'Mynta-Clone' repository. The workflow is named 'build.yml' and is set to run in the 'main' branch. It contains one job named 'build'. The job has an 'on' trigger for pushes to the 'main' branch. The job itself consists of two steps: a checkout step using the 'actions/checkout@v2' action with 'fetch-depth: 0' to disable shallow clones, and a SonarQube scan step using the 'sonarsource/sonarqube-scan-action@master' action. Both steps require environment variables 'SONAR\_TOKEN' and 'SONAR\_HOST\_URL' to be defined.

```
1  Mynta-Clone / .github / workflows / build.yml
2
3  on:
4    push:
5      branches:
6        - main
7
8
9  jobs:
10   build:
11     name: Build
12     runs-on: ubuntu-latest
13     steps:
14       - uses: actions/checkout@v2
15         with:
16           fetch-depth: 0 # Shallow clones should be disabled for a better relevancy of analysis
17       - uses: sonarsource/sonarqube-scan-action@master
18         env:
19           SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
20           SONAR_HOST_URL: ${{ secrets.SONAR_HOST_URL }}
```

Commit changes

Now click on Actions

The screenshot shows the GitHub repository 'Myntra-Clone' with the 'Actions' tab highlighted by a red circle. Other tabs visible include 'Pull requests', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The repository is described as 'Public'.

Click on Workflow

The screenshot shows the 'All workflows' page with one workflow run listed. The workflow is named 'Create build.yml'. It was triggered by a push from user 'Aj7Ay' (commit 170286b) and is currently in progress. The status bar indicates it started 'now' and is 'In progress'.

Click on build

The screenshot shows the detailed view of the 'Create build.yml' workflow. It triggered via a push from 'Aj7Ay' (commit 170286b) on the 'main' branch. The build status is 'In progress'. A specific step, 'Build', is highlighted with a red box and took 9 seconds to complete.

Let's click on Build and see what are the steps involved

The screenshot shows the logs for the 'Build' step. It includes a summary of the job setup and execution. One action, 'Run sonarsource/sonarqube-scan-action@master', is highlighted with a red box.

Click on Run Sonarsource and you can do this after the build completion

```

Summary
Build
succeeded now in 1m 7s
Build
Run sonarsource/sonarqube-scan-action@master
135 INFO: SCM Publisher 69 source files to be analyzed
136 INFO: SCM Publisher 69/69 source files have been analyzed (done) | time=994ms
137 INFO: CPD Executor 6 files had no CPD blocks
138 INFO: CPD Executor Calculating CPD for 60 files
125 INFO: CPD Executor CPD calculation finished (done) | time=43ms
126 INFO: Analysis report generated in 130ms, dir size=432.8 kB
127 INFO: Analysis report compressed in 215ms, zip size=270.2 kB
128 INFO: Analysis report uploaded in 107ms
129 INFO: ANALYSIS SUCCESSFUL, you can find the results at: ***/dashboard?id=Netflix
130 INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
131 INFO: More about the report processing at ***/api/ce/task?id=AyTftis7lwc_i711jGN
132 INFO: Analysis total time: 32.013 s
133 INFO: -----
134 INFO: EXECUTION SUCCESS
135 INFO: -----
136 INFO: Total time: 52.778s
137 INFO: Final Memory: 17M/64M
138 INFO: -----

```

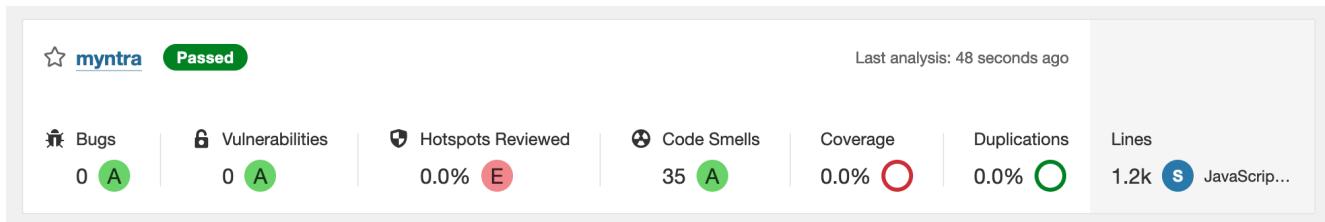
Build complete.

```

Summary
Build
succeeded now in 1m 7s
Build
Set up job
Build sonarsource/sonarqube-scan-action@master
1 Build container for action use: '/home/runn...
2 Run actions/checkout@v2
3 Run sonarsource/sonarqube-scan-action@master
4 Post Run sonarsource/sonarqube-scan-action@master
5 Post Run actions/checkout@v2
6 Complete job

```

Go to the Sonarqube dashboard and click on projects and you can see the analysis



If you want to see the full report, click on issues.

## Step 4: Add GitHub Runner

Go to GitHub and click on Settings -> Actions -> Runners

Repository name: Netflix-clone

**Template repository**

**Require contributors to sign off on web-based commits**

**Default branch**

The default branch is considered the "base" branch in your repository, against which all pull requests and code commits are automatically made, unless you specify a different branch.

main

Click on New self-hosted runner

New self-hosted runner

Host your own runners and customize the environment used to run jobs in your GitHub Actions workflows. [Learn more about self-hosted runners](#).

**There are no runners configured**

[Learn more about using runners](#) to run actions on your own servers.

Now select Linux and Architecture X64

Runners / Add new self-hosted runner · Aj7Ay/Netflix-clone

Adding a self-hosted runner requires that you download, configure, and execute the GitHub Actions Runner. By downloading and configuring the GitHub Actions Runner, you agree to the [GitHub Terms of Service](#) or [GitHub Corporate Terms of Service](#), as applicable.

**Runner image**

macOS

Linux

Windows

**Architecture**

x64

Download

Use the below commands to add a self-hosted runner

## Download

```
# Create a folder
$ mkdir actions-runner && cd actions-runner

# Download the latest runner package
$ curl -o actions-runner-linux-x64-2.310.2.tar.gz -L
https://github.com/actions/runner/releases/download/v2.310.2/actions-runner-linux-x64-2.310.2.tar.gz

# Optional: Validate the hash
$ echo "fb28a1c3715e0a6c5051af0e6eff9c255009e2eec6fb08bc2708277fbb49f93" actions-runner-linux-x64-2.310.2.tar.gz" | shasum -a 256 -c

# Extract the installer
$ tar xzf ./actions-runner-linux-x64-2.310.2.tar.gz
```

## Configure

```
# Create the runner and start the configuration experience
$ ./config.sh --url https://github.com/Aj7Ay/Netflix-clone --token A2MXW45MNGB3QV6SJ6D5LWTFGCRPW

# Last step, run it!
$ ./run.sh
```

## Using your self-hosted runner

```
# Use this YAML in your workflow file for each job
runs-on: self-hosted
```

Go to Putty or Mobaxtreme and connect to your ec2 instance

And paste the commands

NOTE: USE YOUR RUNNER COMMANDS (EXAMPLE CASE IAM USING MINE)

```
mkdir actions-runner && cd actions-runner
```



```
ubuntu@ip-172-31-32-28:~$ 
ubuntu@ip-172-31-32-28:~$ 
ubuntu@ip-172-31-32-28:~$ mkdir actions-runner && cd actions-runner
ubuntu@ip-172-31-32-28:~/actions-runner$ █
```

The command “mkdir actions-runner && cd actions-runner” is used to create a new directory called “actions-runner” in the current working directory and then immediately change the current working directory to the newly created “actions-

runner” directory. This allows you to organize your files and perform subsequent actions within the newly created directory without having to navigate to it separately.



```
curl -o actions-runner-linux-x64-2.310.2.tar.gz -L https://github.com/ac
```



This command downloads a file called “actions-runner-linux-x64-2.310.2.tar.gz” from a specific web address on GitHub and saves it in your current directory.

```
ubuntu@ip-172-31-32-28:~/actions-runner$ curl -o actions-runner-linux-x64-2.310.2.tar.gz -L https://github.com/actions/runners/download/v2.310.2/actions-runner-linux-x64-2.310.2.tar.gz
ubuntu@ip-172-31-32-28:~/actions-runner$ ls -l
total 183028
-rw-rw-r-- 1 ubuntu ubuntu 187416718 Oct 19 02:33 actions-runner-linux-x64-2.310.2.tar.gz
ubuntu@ip-172-31-32-28:~/actions-runner$
```

Let's validate the hash installation



```
echo "fb28a1c3715e0a6c5051af0e6eff9c255009e2eec6fb08bc2708277fbb49f93"
```



```
ubuntu@ip-172-31-32-28:~/actions-runner$ echo "fb28a1c3715e0a6c5051af0e6eff9c255009e2eec6fb08bc2708277fbb49f93" | shasum -a 256 -c
actions-runner-linux-x64-2.310.2.tar.gz: OK
ubuntu@ip-172-31-32-28:~/actions-runner$
```

Now Extract the installer



```
tar xzf ./actions-runner-linux-x64-2.310.2.tar.gz
```



```
ubuntu@ip-172-31-32-28:~/actions-runner$  
ubuntu@ip-172-31-32-28:~/actions-runner$  
ubuntu@ip-172-31-32-28:~/actions-runner$ tar xzf ./actions-runner-linux-x64-2.310.2.tar.gz  
ubuntu@ip-172-31-32-28:~/actions-runner$
```

Let's configure the runner



```
./config.sh --url https://github.com/Aj7Ay/Netflix-clone --token A2MXW45MNGB3QV6SJ6D5LWTGCRPW
```



```
ubuntu@ip-172-31-32-28:~/actions-runner$  
ubuntu@ip-172-31-32-28:~/actions-runner$  
ubuntu@ip-172-31-32-28:~/actions-runner$ ./config.sh --url https://github.com/Aj7Ay/Netflix-clone --token A2MXW45MNGB3QV6SJ6D5LWTGCRPW  
-----  
          AJAY  
-----  
Self-hosted runner registration  
  
# Authentication  
  
✓ Connected to GitHub  
  
# Runner Registration  
  
Enter the name of the runner group to add this runner to: [press Enter for Default] CLICK ENTER  
Enter the name of runner: [press Enter for ip-172-31-32-28] aws-netflix PROVIDE A RUNNER NAME HERE  
This runner will have the following labels: 'self-hosted', 'Linux', 'X64'  
Enter any additional labels (ex. label-1,label-2): [press Enter to skip] aws-netflix LABEL NAME  
✓ Runner successfully added  
✓ Runner connection is good  
  
# Runner settings  
  
Enter name of work folder: [press Enter for _work] Enter  
✓ Settings Saved.  
ubuntu@ip-172-31-32-28:~/actions-runner$
```

If you provide multiple labels use commas for each label

Let's start runner



```
./run.sh
```



```
ubuntu@ip-172-31-32-28:~/actions-runner$ ./run.sh
```

✓ Connected to GitHub

Current runner version: '2.310.2'  
2023-10-19 02:35:20Z: Listening for Jobs

## EKS provision

Clone the repo onto your instance



```
git clone https://github.com/Aj7Ay/Myntra-Clone.git  
cd Myntra-Clone  
cd EKS-TF
```



This changes the directory to EKS terraform files

Change your S3 bucket in the backend file

Initialize the terraform



```
terraform init
```



```
ubuntu@ip-172-31-36-122:~/TIC-TAC-TOE/Eks-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 v5.23.1...
- Installed hashicorp/aws v5.23.1 (signed by HashiCorp)

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

Terraform has been successfully initialized!

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

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

Validate the configuration and syntax of files



terraform validate



```
ubuntu@ip-172-31-36-122:~/TIC-TAC-TOE/Eks-terraform$  
ubuntu@ip-172-31-36-122:~/TIC-TAC-TOE/Eks-terraform$ terraform validate  
Success! The configuration is valid.
```

Plan and apply



terraform plan

terraform apply --auto-approve



```
ubuntu@ip-172-31-36-122:~/TIC-TAC-TOE/Eks-terraform$ 
ubuntu@ip-172-31-36-122:~/TIC-TAC-TOE/Eks-terraform$ terraform apply
data.aws_iam_policy_document.assume_role: Reading...
data.aws_vpc.default: Reading...
data.aws_iam_policy_document.assume_role: Read complete after 0s [id=3552664922]
data.aws_vpc.default: Read complete after 1s [id=vpc-0f6bdd74ced5c07c0]
data.aws_subnets.public: Reading...
data.aws_subnets.public: Read complete after 0s [id=ap-south-1]

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

Terraform will perform the following actions:

# aws_eks_cluster.example will be created
+ resource "aws_eks_cluster" "example" {
    + arn          = (known after apply)
    + certificate_authority = (known after apply)
    + cluster_id   = (known after apply)
    + created_at   = (known after apply)
    + endpoint     = (known after apply)
    + id           = (known after apply)
    + identity     = (known after apply)
    + name         = "EKS_CLOUD"
    + platform_version = (known after apply)
    + role_arn     = (known after apply)
    + status        = (known after apply)
    + tags_all     = (known after apply)
    + version       = (known after apply)
}
```

It will take 10 minutes to create the cluster

Cluster name	Status	Kubernetes version	Provider
EKS_CLOUD	Active	1.28	EKS

Node group ec2 instance

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
Jenkins-ARGO	i-049634a401c64808b	Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b	ec2

Now add the remaining steps

Next, install npm dependencies

- name: NPM Install
 

```
run: npm install # Add your specific npm install command
```

This step runs `npm install` to install Node.js dependencies. You can replace this with your specific npm install command.

Create a Personal Access token for your Dockerhub account

Go to docker hub and click on your profile -> Account settings -> security -> New access token

The screenshot shows the Docker Hub user interface. At the top, there's a navigation bar with links for Explore, Repositories, Organizations, Help, and an Upgrade button. On the right, a user profile is shown with the name "sevenajay" and a dropdown menu. This dropdown menu is highlighted with a red box and contains options: "What's New", "My Profile", "Account Settings" (which is also highlighted with a red box and has a red number "2" next to it), "Billing", and "Sign out". Below the dropdown, the main content area shows the user's profile picture and name "sevenajay", followed by "User Joined August 10, 2022". A sidebar on the left lists account management options: General, Security (which is highlighted with a red box and has a red number "3" next to it), Default Privacy, Notifications, Convert Account, and Deactivate Account. The main content area is titled "Access Tokens" and displays a table of existing tokens. The table columns are Description, Scope, Last Used, Created, and Active. One token is listed: "Ansible" with Scope "Read, Write, Delete", Last Used "Sep 16, 2023 08:07:09", Created "Sep 15, 2023 18:08:25", and Active status "Yes". At the bottom of the table, there's a "New Access Token" button, which is highlighted with a red box and has a red number "4" next to it.

It asks for a name Provide a name and click on generate token

## New Access Token

A personal access token is similar to a password except you can have many tokens and revoke access to each one at any time. [Learn more](#)

Access Token Description \*

Access permissions

Read, Write, Delete

Read, Write, Delete tokens allow you to manage your repositories.

Cancel Generate

Two-factor authentication adds an extra layer of security to your account by

Copy the token save it in a safe place, and close

## Copy Access Token

When logging in from your Docker CLI client, use this token as a password. [Learn more](#)

### ACCESS TOKEN DESCRIPTION

Netflix

### ACCESS PERMISSIONS

Read, Write, Delete

To use the access token from your Docker CLI client:

1. Run `docker login -u sevenajay`

2. At the password prompt, enter the personal access token.

`dckr_pat_4lnKFbaykudW-ueAmMIDcRMyFmA`



WARNING: This access token will only be displayed once. It will not be stored and cannot be retrieved. Please be sure to save it now.

**Copy and Close**

Now Go to GitHub again and click on settings

Search for Secrets and variables and click on and again click on actions

The screenshot shows the GitHub repository settings page for 'Netflix-clone'. The left sidebar has sections like Collaborators, Moderation options, Code and automation (Branches, Tags, Rules, Actions, Webhooks, Environments, Codespaces, Pages), Security (Code security and analysis, Deploy keys, Secrets and variables), and Actions. The 'Secrets and variables' section is highlighted with a red box. The main area shows the 'Default branch' (main) and 'Social Preview' (Upload an image to customize your repository's social media preview). A green button 'New repository secret' is visible.

It will open a page like this click on New Repository secret

The screenshot shows the 'Actions secrets and variables' page. The left sidebar includes 'General', 'Access', 'Collaborators', 'Moderation options', 'Code and automation' (Branches, Tags, Rules, Actions, Webhooks, Environments, Codespaces, Pages), 'Security' (Code security and analysis, Deploy keys, Secrets and variables), and 'Actions'. The 'Secrets and variables' section is highlighted with a red box. The main area has tabs 'Secrets' (selected) and 'Variables'. It shows 'Environment secrets' (Manage environments) and 'Repository secrets' (Manage environments). A green button 'New repository secret' is visible.

Add your Dockerhub username with the secret name as

```
DOCKERHUB_USERNAME #use your dockerhub username
```

The screenshot shows the 'Actions secrets / New secret' page. On the left, there's a sidebar with various repository settings like General, Access, Collaborators, and Code and automation. The main area has two fields: 'Name \*' containing 'DOCKERHUB\_USERNAME' and 'Secret \*' containing 'sevenajay'. A green 'Add secret' button at the bottom is highlighted with a red box.

Click on Add Secret.

Let's add our token also and click on the new repository secret again

Name



DOCKERHUB\_TOKEN



This screenshot is identical to the one above, showing the 'Actions secrets / New secret' page. The 'Name \*' field now contains 'DOCKERHUB\_TOKEN' and the 'Secret \*' field contains 'dckr\_pat\_Qy-YtjVN3MNTfnGjHdnawLisjLU'. The 'Add secret' button is again highlighted with a red box.

Paste the token that you generated and click on Add secret.



```
- name: Docker Login
  run: docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_TOKEN }}

- name: Docker Scout Scan
  run: |
    docker-scout quickview fs://.
    docker-scout cves fs://.

- name: Docker build and push
  run: |
    # Run commands to build and push Docker images
    docker build -t myntra .
    docker tag myntra sevenajay/myntra:latest
    docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_TOKEN }}
    docker push sevenajay/myntra:latest
  env:
    DOCKER_CLI_ACI: 1

- name: Docker Scout Image Scan
  run: |
    docker-scout quickview sevenajay/myntra:latest
    docker-scout cves sevenajay/myntra:latest
```



## 1. Docker Login:

- This step is logging into Docker Hub using the provided Docker Hub username and token.
- The Docker Hub username and token are stored as secrets (`DOCKERHUB_USERNAME` and `DOCKERHUB_TOKEN`), which is a best practice for keeping sensitive information secure.

## 2. Docker Scout Scan:

- This step is using a tool called `docker-scout` to perform scans on the Docker image.
- The `quickview` command is used to perform a quick review of the Docker image file system.

- The `cves` command is used to check for Common Vulnerabilities and Exposures (CVEs) in the Docker image.

### 3. Docker build and push:

- This step involves building a Docker image using the `docker build` command.
- It then tags the built image with the name “sevenajay/myntra:latest” using the `docker tag` command.
- Subsequently, it logs into Docker Hub again and pushes the built image to Docker Hub using the `docker push` command.
- The `env` section sets an environment variable `DOCKER_CLI_ACI` to 1.

### 4. Docker Scout Image Scan:

- Similar to the second step, this step performs scans on the Docker image that was just built and pushed to Docker Hub.
- It uses `docker-scout` with the `quickview` and `cves` commands to inspect the image’s file system and check for vulnerabilities.

In summary, this workflow automates the process of building, tagging, and pushing a Docker image to Docker Hub. Additionally, it performs security scans on both the local Docker image and the pushed image on Docker Hub using the `docker-scout` tool.

Commit changes after adding the code.

[Mynta-Clone / .github / workflows / build.yml](#) in [main](#)

[Edit](#) [Preview](#) [Spaces](#) [2](#) [No wrap](#) [Copy](#)

```

6   jobs:
7     build-analyze-scan:
8       name: Build, Analyze, and Scan
9       runs-on: [self-hosted]
10      steps:
11        - name: Checkout code
12          uses: actions/checkout@v2
13          with:
14            fetch-depth: 0 # Shallow clones should be disabled for a better relevancy of analysis
15
16        - name: Build and analyze with SonarQube
17          uses: sonarsource/sonarqube-scan-action@master
18          env:
19            SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
20            SONAR_HOST_URL: ${{ secrets.SONAR_HOST_URL }}
21
22        - name: NPM Install
23          run: npm install # Add your specific npm install command
24
25        - name: Docker Login
26          run: docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_TOKEN }}
27
28        - name: Docker Scout Scan
29          run: |
30            docker-scout quickview fs://.
31            docker-scout cves fs://.
32
33        - name: Docker build and push
34          run: |
35            # Run commands to build and push Docker images
36            docker build -t myntra .
37            docker tag myntra sevenajay/myntra:latest
38            docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_TOKEN }}
39            docker push sevenajay/myntra:latest
40          env:
41            DOCKER_CLI_ACI: 1
42

```

Use [Control + Shift + m](#) to toggle the [tab](#) key moving focus. Alternatively, use [esc](#) then [tab](#) to move to the next interactive element on the page.

Use [Control + Space](#) or [Option + Space](#) to trigger autocomplete in most situations.

[NPM Install](#) 23s

12 npm audit fix  
13 To address all issues (including breaking changes), run:  
14 npm audit fix --force  
15 Run 'npm audit' for details.

[Docker Login](#) 2s

6 WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.  
7 Configure a credential helper to remove this warning. See  
8 <https://docs.docker.com/engine/reference/commandline/login/#credentials-store>  
9 Login Succeeded

[Docker Scout Scan](#) 18s

1 ► Run docker-scout quickview fs://.  
4 ...Reading file system  
5 ✓ File system read  
6 ...Indexing  
7 ✓ Indexed 1202 packages  
9 Target | fs://. | 6C 16H 10M 0L  
10 What's Next?  
11 View vulnerabilities → docker scout cves fs://.

4m 32s

```

✓ Docker build and push
161 #12 naming to docker.io/library/myntra 0.0s done
162 #12 DONE 51.4s
163 WARNING! Using --password via the CLI is insecure. Use --password-stdin.
164 WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
165 Configure a credential helper to remove this warning. See
166 https://docs.docker.com/engine/reference/commandline/login/#credentials-store
167
168 Login Succeeded
169 The push refers to repository [docker.io/**/myntra]
170 d0331403fe37: Preparing
171 2b226e33680c: Preparing
172 50c4d058160d: Preparing
173 39b250d3b84b: Preparing
174 f5e6da03a6bf: Preparing
175 173e955be7d1: Preparing
176 c694e1c012a5: Preparing
177 77a628713662: Preparing
178 5af4f8f59b76: Preparing
179 c694e1c012a5: Waiting
180 77a628713662: Waiting
181 5af4f8f59b76: Waiting
182 173e955be7d1: Waiting
183 1f5e6da03a6bf: Pushed
184 39b250d3b84b: Pushed
185 d0331403fe37: Pushed
186 173e955be7d1: Mounted from library/node
187 c694e1c012a5: Mounted from library/node
188 77a628713662: Mounted from library/node
189 5af4f8f59b76: Mounted from library/node
190 2b226e33680c: Pushed
191 50c4d058160d: Pushed
192 latest: digest: sha256:ecd95b410dc331c59b9d92dd30203f398f39e9b4a2fecdd656be0fecae5142 size: 2210

```

## Docker Image scan

```

✓ Docker Scout Image Scan
2m 6s

9 ▶ Run docker-scout quickview ***/myntra:latest
14   ...Storing image for indexing
15   ✓ Image stored for indexing
16   ...Indexing
17   ✓ Indexed 1395 packages
18
21 Target      | ***/myntra:latest | 6C   16H   10M   0L   1?
22 digest       | 93c35e035841    |
23 Base image   | node:21-alpine  | 0C   0H    0M   0L   1?
24 Updated base image | node:20-alpine | 0C   0H    0M   0L   1?
25
26
27 What's Next?
28 View vulnerabilities → docker scout cves ***/myntra:latest
29 View base image update recommendations → docker scout recommendations ***/myntra:latest
30 Include policy results in your quickview by supplying an organization → docker scout quickview ***/myntra:latest --org <organization>
31
32   ✓ SBOM of image already cached, 1395 packages indexed
33   ✘ Detected 25 vulnerable packages with a total of 23 vulnerabilities
34
35
36 ## Overview
37
38           | Analyzed Image
39
40 Target      | ***/myntra:latest
41 digest       | 93c35e035841
42 platform    | linux/amd64
43 vulnerabilities | 6C   16H   10M   0L   1?
44 size         | 253 MB
45 packages     | 1395
46
47
48 ## Packages and Vulnerabilities
49
50 1C   2H   0M   0L   loader-utils 2.0.2

```

## Image is pushed to Dockerhub

 sevenajay / myntra

### Description

This repository does not have a description 

 Last pushed: about 19 hours ago

### Docker commands

To push a new tag to this repository:

[Public View](#)

```
docker push sevenajay/myntra:tagname
```

## DEPLOY



## deploy:

```
needs: build-analyze-scan  
runs-on: self-hosted # Use your self-hosted runner label here
```



This section defines another job named “deploy.” It specifies that this job depends on the successful completion of the “build-analyze-scan” job. It also runs on a self-hosted runner. You should replace `self-hosted` with the label of your self-hosted runner.



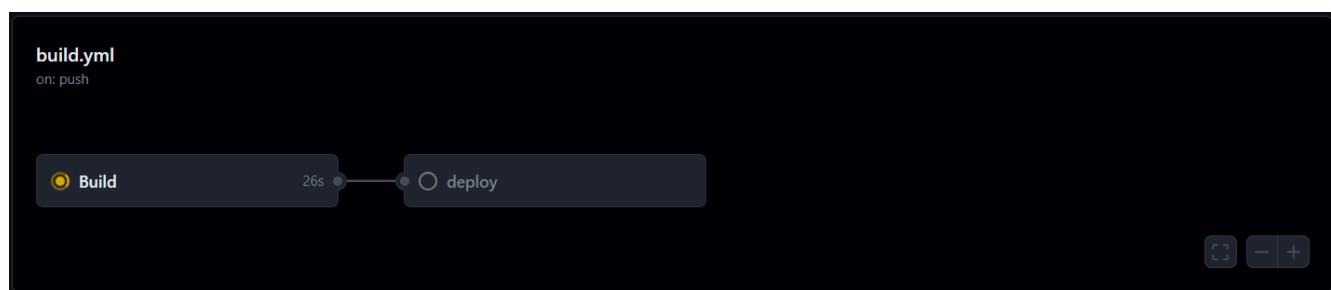
```
- name: Run the container  
run: docker run -d --name myntra -p 3000:3000 sevenajay/myntra:latest
```



This step runs a Docker container named “ticgame” in detached mode ( `-d` ). It maps port 3000 on the host to port 3000 in the container. It uses the Docker image tagged as `sevenajay/myntra:latest` .

If you run this workflow.

## Output



```

✓ Docker Build and push
98 af1982f6d133: Layer already exists
99 c5b325bb721: Layer already exists
100 be322b479aee: Layer already exists
101 d41bcd3a037b: Layer already exists
102 fe0dd845e767b: Layer already exists
103 f25ec1d93a58: Layer already exists
104 3220beed9b06: Layer already exists
105 794ce8b1b516: Layer already exists
106 684f82921421: Layer already exists
107 9af5f53e8f62: Layer already exists
108 42642eab0757: Pushed
109 9fc715bef52e: Pushed
110 latest: digest: sha256:0295f011d6645ab0e1c5fc00f37d90b6f2d624c0be4df5a593675bfb32dd6c3 size: 3055

✓ Image scan
1 ► Run trivy image ***/tic-tac-toe:latest > trivyimage.txt
4
5 2023-10-29T07:10:08.686Z      INFO    Vulnerability scanning is enabled
6 2023-10-29T07:10:08.686Z      INFO    Secret scanning is enabled
7 2023-10-29T07:10:08.686Z      INFO    If your scanning is slow, please try '--scanners vuln' to disable secret scanning
7 2023-10-29T07:10:08.686Z      INFO    Please see also https://aquasecurity.github.io/trivy/v0.46/docs/scanner/secret/#recommendation for faster secret detection

```

## Image scan report

```

✓ Docker Scout Image Scan
2m 6s
9 ► Run docker-scout quickview ***/myntra:latest
14 ...Storing image for indexing
15 ✓ Image stored for indexing
16 ...Indexing
17 ✓ Indexed 1395 packages
18
21 Target      | ***/myntra:latest | 6C   16H   10M   0L   1?
22 digest       | 93c35e035841          |
23 Base image   | node:21-alpine        | 0C   0H   0M   0L   1?
24 Updated base image | node:20-alpine        | 0C   0H   0M   0L   1?
25
26
27 What's Next?
28 View vulnerabilities → docker scout cves ***/myntra:latest
29 View base image update recommendations → docker scout recommendations ***/myntra:latest
30 Include policy results in your quickview by supplying an organization → docker scout quickview ***/myntra:latest --org <organization>
31
32 ✓ SBOM of image already cached, 1395 packages indexed
33 ✘ Detected 25 vulnerable packages with a total of 23 vulnerabilities
34
35
36 ## Overview
37
38 | Analyzed Image
39
40 Target      | ***/myntra:latest
41 digest       | 93c35e035841
42 platform    | linux/amd64
43 vulnerabilities | 6C   16H   10M   0L   1?
44 size         | 253 MB
45 packages     | 1395
46
47
48 ## Packages and Vulnerabilities
49
50 1C   2H   0M   0L   loader-utils 2.0.2

```

Deployed to the container.

deploy	
succeeded 2 minutes ago in 1m 51s	
<input type="checkbox"/>	Search logs
<input checked="" type="checkbox"/>	Set up job 0s
<input checked="" type="checkbox"/>	docker pull image 3s
1	▶ Run docker pull sevenajay/tic-tac-toe:latest
4	latest: Pulling from sevenajay/tic-tac-toe
5	Digest: sha256:6a293a885184298ae304db17a4ba827f2ff0a12f8267c14f6cdef400ed349059
6	Status: Image is up to date for sevenajay/tic-tac-toe:latest
7	docker.io/sevenajay/tic-tac-toe:latest
<input type="checkbox"/>	Image scan 1m 42s
<input checked="" type="checkbox"/>	Deploy to container 1s
1	▶ Run docker run -d --name game -p 3000:3000 sevenajay/tic-tac-toe:latest
4	529004a51c3519f99d6017a007df1b9938b77b1d3e26e22a51f69aaea3a2a0b
<input type="checkbox"/>	Complete job 0s

## output



ec2-ip:3000



The screenshot shows the Myntra website on a desktop browser. The header features the Myntra logo and navigation links for MEN, WOMEN, KIDS, and HOME & LIVING. A search bar is on the right, along with Profile, Wishlist, and Bag icons. A banner at the top indicates a sale ends in 0 h: 46 m: 32 s. A red promotional box on the left offers flat Rs.150 off on a minimum purchase of Rs.999. Below the offer is the Myntra logo. To the right, a couple is shown against a yellow background. The bottom of the screen shows the Windows taskbar with the Start button, a search bar, and various pinned icons like File Explorer, Edge, and Mail. The system tray shows the date as 29-Apr-23 and the time as 11:13 PM.

# Deploy to EKS



```
- name: Update kubeconfig
  run: aws eks --region cluster-region update-kubeconfig --name cluster-
```



This step updates the kubeconfig to configure `kubectl` to work with an Amazon EKS cluster in the region with the name of your cluster.



```
- name: Deploy to EKS
  run: kubectl apply -f deployment-service.yml
```



This step deploys Kubernetes resources defined in the `deployment-service.yml` file to the Amazon EKS cluster using `kubectl apply`.

## Complete Workflow



```
name: Build,Analyze,scan

on:
  push:
    branches:
      - main

jobs:
  build-analyze-scan:
    name: Build
    runs-on: [self-hosted]
    steps:
      - name: Checkout code
        uses: actions/checkout@v2
        with:
          fetch-depth: 0 # Shallow clones should be disabled for a beti
```

```
- name: Build and analyze with SonarQube
  uses: sonarsource/sonarqube-scan-action@master
  env:
    SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
    SONAR_HOST_URL: ${{ secrets.SONAR_HOST_URL }}

- name: npm install dependency
  run: npm install

- name: Docker Login
  run: docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_PASSWORD }}

- name: Docker Scout Scan
  run: |
    docker-scout quickview fs://.
    docker-scout cves fs://.

- name: Docker build and push
  run: |
    # Run commands to build and push Docker images
    docker build -t myntra .
    docker tag myntra sevenajay/myntra:latest
    docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_PASSWORD }}
    docker push sevenajay/myntra:latest
  env:
    DOCKER_CLI_ACI: 1

- name: Docker Scout Image Scan
  run: |
    docker-scout quickview sevenajay/myntra:latest
    docker-scout cves sevenajay/myntra:latest

deploy:
  needs: build-analyze-scan
  runs-on: [self-hosted]
  steps:
    - name: docker pull image
      run: docker pull sevenajay/myntra:latest

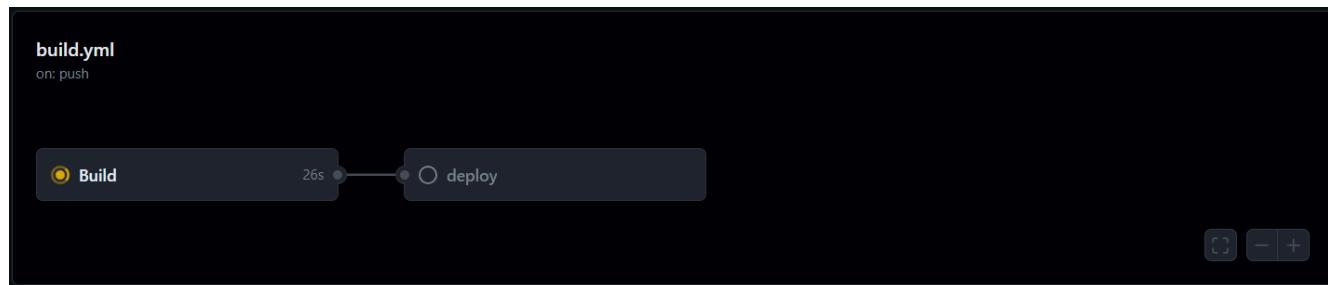
    - name: Deploy to container
      run: docker run -d --name game -p 3000:3000 sevenajay/myntra:latest

    - name: Update kubeconfig
      run: aws eks --region ap-south-1 update-kubeconfig --name EKS_CI
```

```
- name: Deploy to kubernetes
  run: kubectl apply -f deployment-service.yml
```

commit changes

Run this workflow now



Docker Build and push 1m 41s

```
98 af1982f6d133: Layer already exists
99 c5b325bdd721: Layer already exists
100 be322b479aee: Layer already exists
101 d41bcd3a037b: Layer already exists
102 fe0d845e767b: Layer already exists
103 f25ec1d93a58: Layer already exists
104 3220beed9b06: Layer already exists
105 794ce8b1b516: Layer already exists
106 684f82921421: Layer already exists
107 9af5f53e8f62: Layer already exists
108 42642eab0757: Pushed
109 9fc715bef52e: Pushed
110 latest: digest: sha256:0295f011d6645ab0e1c5fc00f37d90b6f2d624c0be4df5a593675bfb32dd6c3 size: 3055
```

Image scan 1m 59s

```
1 ► Run trivy image ***/tic-tac-toe:latest > trivymage.txt
4
5 2023-10-29T07:10:08.686Z      INFO    Vulnerability scanning is enabled
5 2023-10-29T07:10:08.686Z      INFO    Secret scanning is enabled
6 2023-10-29T07:10:08.686Z      INFO    If your scanning is slow, please try '--scanners vuln' to disable secret scanning
7 2023-10-29T07:10:08.686Z      INFO    Please see also https://aquasecurity.github.io/trivy/v0.46/docs/scanner/secret/#recommendation for faster secret detection
```

deploy succeeded 2 minutes ago in 1m 51s

Set up job 0s

docker pull image 3s

```
1 ► Run docker pull sevenajay/tic-tac-toe:latest
4 latest: Pulling from sevenajay/tic-tac-toe
5 Digest: sha256:6a293a885184298ae304db17a4ba827f2ff0a12f8267c14f6cdef400ed349059
6 Status: Image is up to date for sevenajay/tic-tac-toe:latest
7 docker.io/sevenajay/tic-tac-toe:latest
```

Image scan 1m 42s

Deploy to container 1s

```
1 ► Run docker run -d --name game -p 3000:3000 sevenajay/tic-tac-toe:latest
4 529004a51c3519f99d6017a007df1b9938b77b1d3e26e22a51f69aaeae3a2a0b
```

Complete job 0s

Deployed to the container.

Deployed to EKS

```

v ✓ Deploy to container
  1 ► Run docker run -d --name game -p 3000:3000 sevenajay/tic-tac-toe:latest
  4 5d3ad75d034b56ec3ed230bb801b934d7e5902711bffc8868d6614f3c165d187

> ✓ Update kubeconfig
  0s

v ✓ Deploy to kubernetes
  1 ► Run kubectl apply -f deployment-service.yml
  4
  4 deployment.apps/tic-tac-toe created
  5 service/tic-tac-toe-service created
  1s

v ✓ Send a Slack Notification
  1 ► Run act10ns/slack@v1
  9
  9 Reading config file .github/slack.yml...
  10 Sent deploy status of SUCCESS to Slack!
  1s

> ✓ Complete job
  0s

```

Triggered via push 12 minutes ago	Status	Total duration	Artifacts
 Aj7Ay pushed -o- c18d5c7 main	Success	7m 13s	-

**deployed.yml**

```
on: push
```



```

graph LR
    A[Build] --> B[deploy]
    A -- "4m 33s" --> B
    B -- "1m 59s" --> C

```

Job completed.

Let's go to the Ec2 ssh connection

Provide this command



```
kubectl get all
```



```
YfMUW9R00A6gn-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
service/kubernetes  ClusterIP   10.100.0.1   <none>
service/tetris-service  LoadBalancer  10.100.212.20  a0725d5c76f3e47a8af3c7391c061307-244742789.ap-south-1.elb.amazonaws.com  PORT(S)        AGE
                                                               443/TCP       123m
                                                               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 in the security group for the Node group instance.

After that copy the external IP and paste it into the browser

output

## Destruction workflow

```
name: Build,Analyze,scan
```

on:

```
push:  
  branches:  
    - main
```

jobs:

```
  build-analyze-scan:
```

```
    name: Build
```

```
    runs-on: [self-hosted]
```

```
    steps:
```

```
      - name: Checkout code
```

```
        uses: actions/checkout@v2
```

```
        with:
```

```
          fetch-depth: 0 # Shallow clones should be disabled for a beta
```

```
      - name: Deploy to container
```

```
        run: |
```

```
          docker stop myntra
```

```
          docker rm myntra
```

```
      - name: Update kubeconfig
```

```
        run: aws eks --region ap-south-1 update-kubeconfig --name EKS_CI
```

```
      - name: Deploy to kubernetes
```

```
        run: kubectl delete -f deployment-service.yml
```



It will delete the container and delete the Kubernetes deployment.

Stop the self-hosted runner.

Now go inside the Myntra-clone

To delete the Eks cluster



```
cd /home/ubuntu  
cd Myntra-clone
```

```
cd EKS-TF  
terraform destroy --auto-approve
```

It w



Meanwhile, delete the Dockerhub Token

Once cluster destroys

Delete The ec2 instance and IAM role.

Delete the secrets from GitHub also.



**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

2 responses to “GitHub Actions Scouting Myntra App | DevSecOps”



Naveen

13 March 2024

Is it a multi-microservice based project or just a front end?

[Reply](#)

**mrcloudbook.com**

17 March 2024

Just FE

[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

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

AI, DevOps

How to Install docker-ai  
on Ubuntu 24.04

15 June 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](#)

