

Mr Cloud Book

Home Blog DevSecOps Contact About Me Testimonials

Search Blogs

DevOps

GitHub Actions: Netflix Deployment Powered by DevSecOps

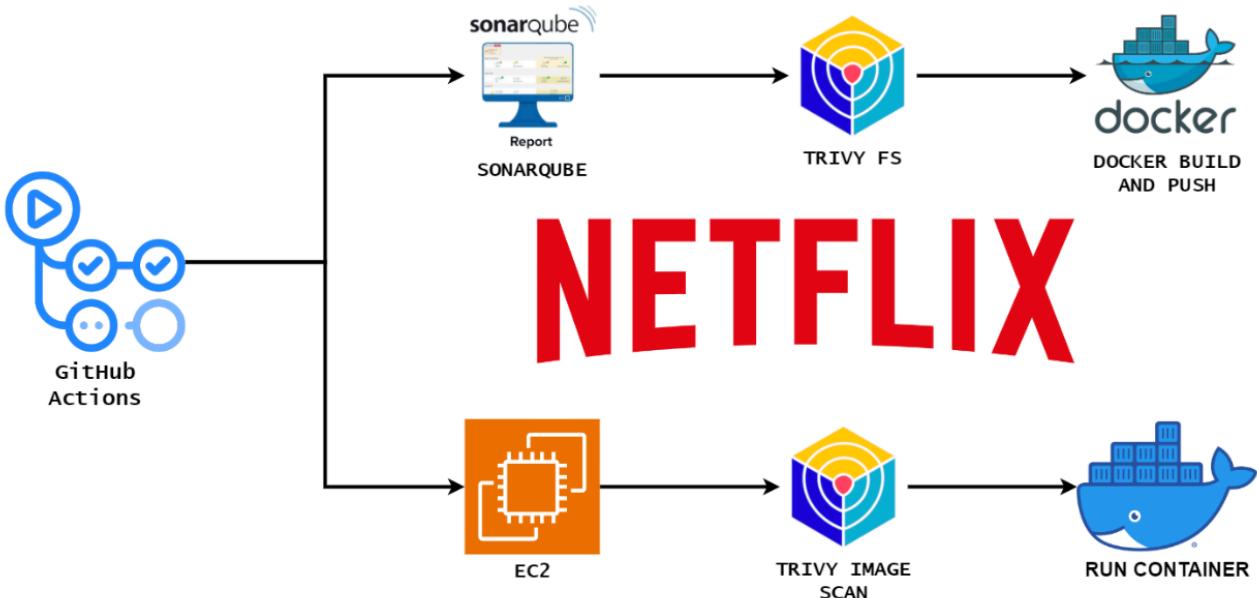


mrclocloudbook.com · 8 January 2024



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.



Contents [hide]

- Step1: Launch an Ec2 Instance
- Step2A: Install Docker and Run Sonarqube Container
- Step2B: Integrating SonarQube with GitHub Actions
- Step3: Let's scan files using Trivy
- Step4A: Docker build and push to Dockerhub
- Step4B: Create a TMDB API Key
- Step5A: Add a self-hosted runner to Ec2
- Step5B: Final workflow to run the container

Step1: Launch an Ec2 Instance

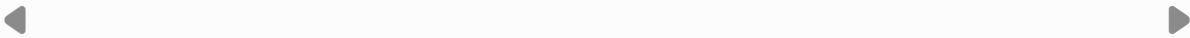
To launch an AWS EC2 instance with Ubuntu 22.04 using the AWS Management Console, sign in to your AWS account, access the EC2 dashboard, and click “Launch Instances.” In “Step 1,” select “Ubuntu 22.04” as the AMI, and in “Step 2,” choose “t2.medium” as the instance type. Configure the instance details, storage,

tags, and security group settings according to your requirements. Review the settings, create or select a key pair for secure access, and launch the instance. Once launched, you can connect to it via SSH using the associated key pair.

Step2A: Install Docker and Run Sonarqube Container

Connect to your Ec2 instance using Putty, Mobaxtreme or Git bash and install docker on it.

```
sudo apt-get update
sudo apt install docker.io -y
sudo usermod -aG docker ubuntu
newgrp docker
sudo chmod 777 /var/run/docker.sock
```



Pull the SonarQube Docker image and run it.

After the docker installation, we will create a Sonarqube container (Remember to add 9000 ports in the security group).

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



```
ubuntu@ip-172-31-42-253:~$ sudo chmod 777 /var/run/docker.sock
ubuntu@ip-172-31-42-253:~$ 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
44ba2882f8eb: Pull complete
2cabecc57fa36: Pull complete
c20481384b6a: Pull complete
b77b17ee74fb: Pull complete
38617faac714: Pull complete
706720f5875e: Pull complete
65a29568c257: Pull complete
Digest: sha256:1a118f8ab960d6c3d4ea8b4455a5a6560054511c88a6816f1603f764d5dcc77c
Status: Downloaded newer image for sonarqube:lts-community
4b66c96bf9ad3d62289436af7f752fdb0499309720ca3065e2f2e32301b50139
ubuntu@ip-172-31-42-253:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
4b66c96bf9ad sonarqube:lts-community "/opt/sonarqube/dock..." 9 seconds ago Up 5 seconds 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp sonar
ubuntu@ip-172-31-42-253:~$
```

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
```



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

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 your Sonarqube password & This is the Sonarqube dashboard

← → C 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 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
--	--	---	--	--

Step2B: 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

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

Are you just testing or have an advanced use-case? Create a project manually.

Manually

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

All fields marked with * are required

Project display name *
Netflix

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

Project key *
Netflix

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

Main branch name *
main

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

Set Up

On the next page click on With GitHub actions

How do you want to analyze your repository?

Do you want to integrate with your favorite CI? Choose one of the following tutorials.

- With Jenkins
- With GitHub Actions**
- With Bitbucket Pipelines
- With GitLab CI
- With Azure Pipelines
- Other CI

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**.

2 Create SonarQube Host URL 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

Let's Open your GitHub and select your Repository

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

Aj7Ay / Netflix-clone

Code Pull requests Actions Projects Security Insights Settings

Netflix-clone Public forked from [gsbarure/netflix-clone-react-typescript](#)

main 1 branch 0 tags

This branch is 16 commits ahead of gsbarure/main.

Aj7Ay Delete sonar-project.properties 54c7c49 2 minutes ago 111 commits

- Kubernetes Create service.yml last month
- public fix README 5 months ago
- src add data loader to GenreExplore page 5 months ago
- .dockerignore added Dockerfile last year
- .env Update .env last week

About

Netflix Clone using React, Typescript, Material UI

[netflix-clone-react-typescript.vercel.app](#)

Readme Activity 7 stars 0 watching 120 forks

Releases

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

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

Now go back to Your Sonarqube Dashboard

Copy SONAR_TOKEN and click on Generate Token

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` **COPY THIS ONE**
- In the **Value** field, enter an existing token, or a newly generated one. **Generate a token** **CLICK HERE**
- Click on **Add secret**

1. Click on **New repository secret**
 2. In the **Name** field, enter `SONAR_HOST_URL`
 3. In the **Value** field, enter `http://13.126.202.56:9000`
 4. Click on **Add secret**

Continue

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](#).

Token name	Expires in
Analyze "Netflix"	30 days
<input style="background-color: #0072bc; color: white; border: none; padding: 5px; width: 100%;" type="button" value="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_0fcfd59dfc0eef5378e95d8aebae06134b34bbe55 +

! New token "sqp_0fcfd59dfc0eef5378e95d8aebae06134b34bbe55" 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

Actions secrets / New secret

Name * SONAR_TOKEN

Secret * sqp_0fcfd59dfc0eef5378e95d8aebae06134b34bbe55

Add secret

General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Webhooks
Environments
Codespaces
Pages

Now go back to the Sonarqube Dashboard

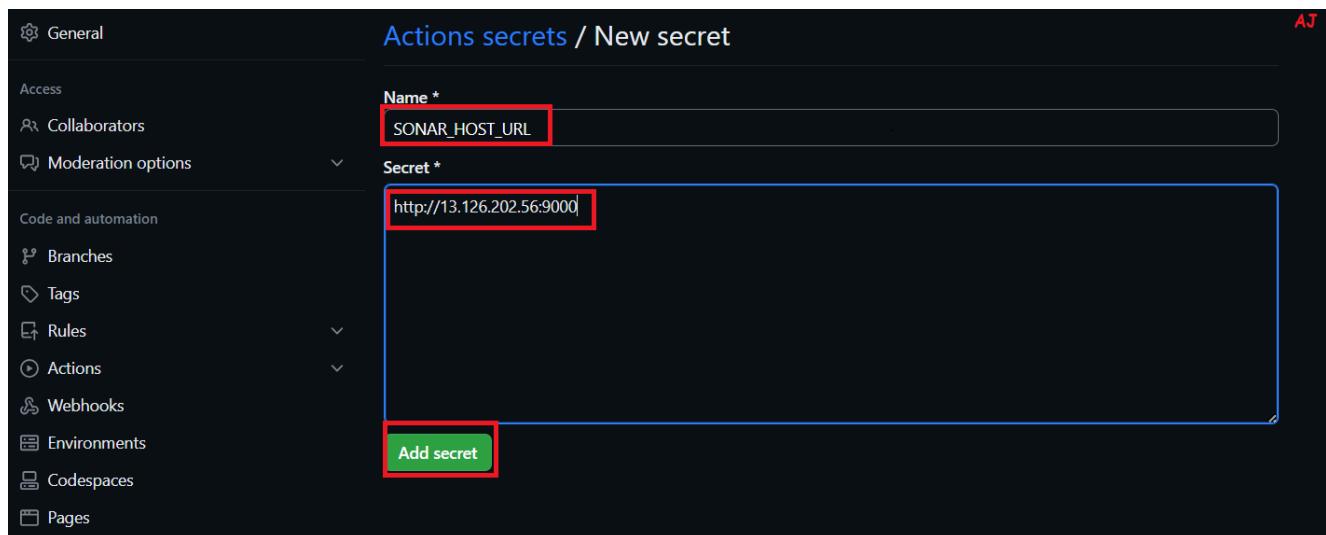
Copy the Name and Value

1 Create GitHub Secrets

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

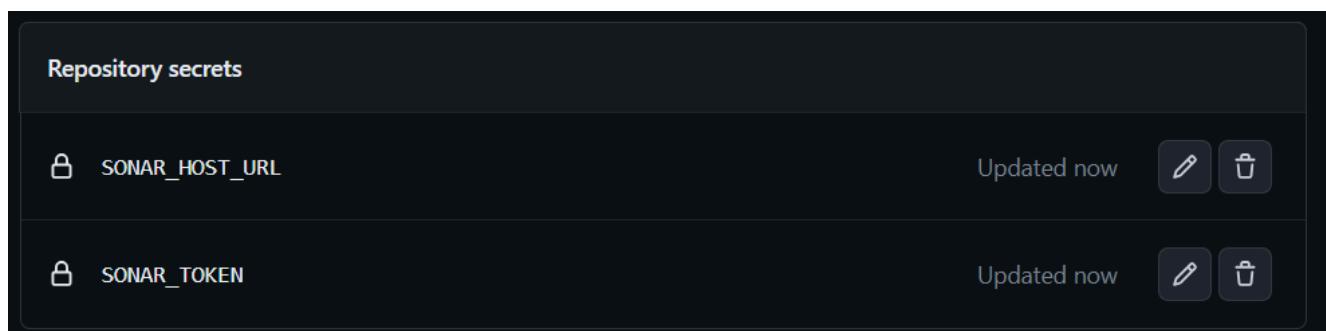
1. Click on **New repository secret**.
 2. In the **Name** field, enter `SONAR_TOKEN` 
 3. In the **Value** field, enter an existing token, or a newly generated one: 
 4. Click on **Add secret**.
-
1. Click on **New repository secret**.
 2. In the **Name** field, enter `SONAR_HOST_URL` 
 3. In the **Value** field, enter `http://13.126.202.56:9000` 
 4. 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' form. On the left, there's a sidebar with 'General' selected under 'Access', and 'Collaborators', 'Moderation options', and other sections like 'Code and automation', 'Branches', 'Tags', 'Rules', 'Actions', 'Webhooks', 'Environments', 'Codespaces', and 'Pages'. The main area has 'Name *' set to 'SONAR_HOST_URL' and 'Secret *' set to 'http://13.126.202.56:9000'. A large blue-bordered box surrounds the secret value input field. At the bottom right is a green 'Add secret' button.

Our Sonarqube secrets are added and you can see



The screenshot shows the 'Repository secrets' section. It lists two secrets: 'SONAR_HOST_URL' and 'SONAR_TOKEN'. Both were updated 'now'. Each secret has a pencil icon for editing and a trash bin icon for deleting.

Secret	Last Updated	Action
SONAR_HOST_URL	Updated now	 
SONAR_TOKEN	Updated now	 

Go to Sonarqube Dashboard and click on continue

The screenshot shows the Sonarqube dashboard for a project named 'Netflix'. The top navigation bar includes 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', and 'Administration'. A search bar at the top right contains the placeholder 'Search for projects...'. A user icon labeled 'AJAY' is visible. Below the navigation, there are tabs for 'Overview', 'Issues', 'Security Hotspots', 'Measures', 'Code', and 'Activity'. On the right, there are 'Project Settings' and 'Project Information' buttons. The main content area displays a step-by-step guide for creating GitHub secrets:

- 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: **Generate a token**.
- 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**.

A red box surrounds the 'Continue' button at the bottom left of the steps.

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

The screenshot shows the Sonarqube dashboard for the same 'Netflix' project. The top navigation and tabs are identical. The main content area displays a step-by-step guide for creating a workflow:

- Create GitHub Secrets**
- Create Workflow YAML File**

1. What option best describes your build?

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

IN MY CASE IAM USING REACT JS

3. You're all set!

A red box surrounds the 'Other (for JS, TS, Go, Python, PHP, ...)' button in the build type section.

Now it Generates and workflow for my Project

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

```
sonar.projectKey=Netflix
```

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
```

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

This branch is 16 commits ahead of gsbarure:main.

Aj7Ay Delete sonar-project.properties 54c7c49 4 minutes ago 111 commits

Kubernetes	Create service.yml	last month
public	fix README	5 months 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=Netflix
```

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=Netflix



Add in GitHub like this

```
sonar.projectKey=Netflix
```

Let's add our workflow

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

This branch is 16 commits ahead of gsbarure:main.

Author	Commit Message	Date	Commits
Aj7Ay	Delete sonar-project.properties	54c7c49 4 minutes ago	111 commits
	Kubernetes	Create service.yml	last month
	public	fix README	5 months ago

Here is the file name



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



SonarQube Projects Issues Rules Quality Profiles Quality Gates Administration ? Search for projects... A

Netflix main Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

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
    #   timeout-minutes: 5
    #   env:
    #     SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
```

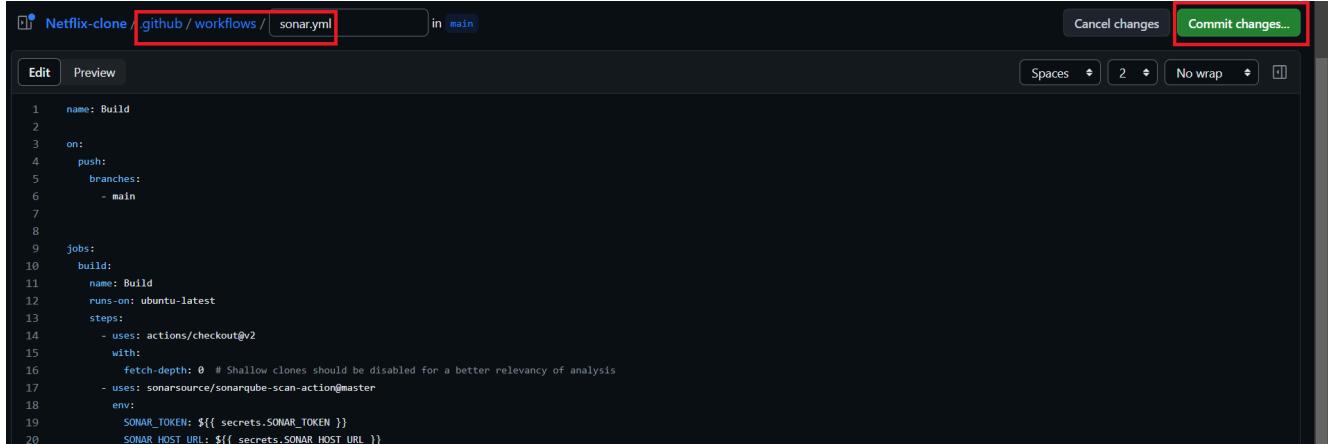
Copy

Copy content and add it to the file



```
name: Build,Analyze,scan
on:
  push:
    branches:
      - main
jobs:
  build-analyze-scan:
    name: Build
    runs-on: ubuntu-latest
    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 }}
```

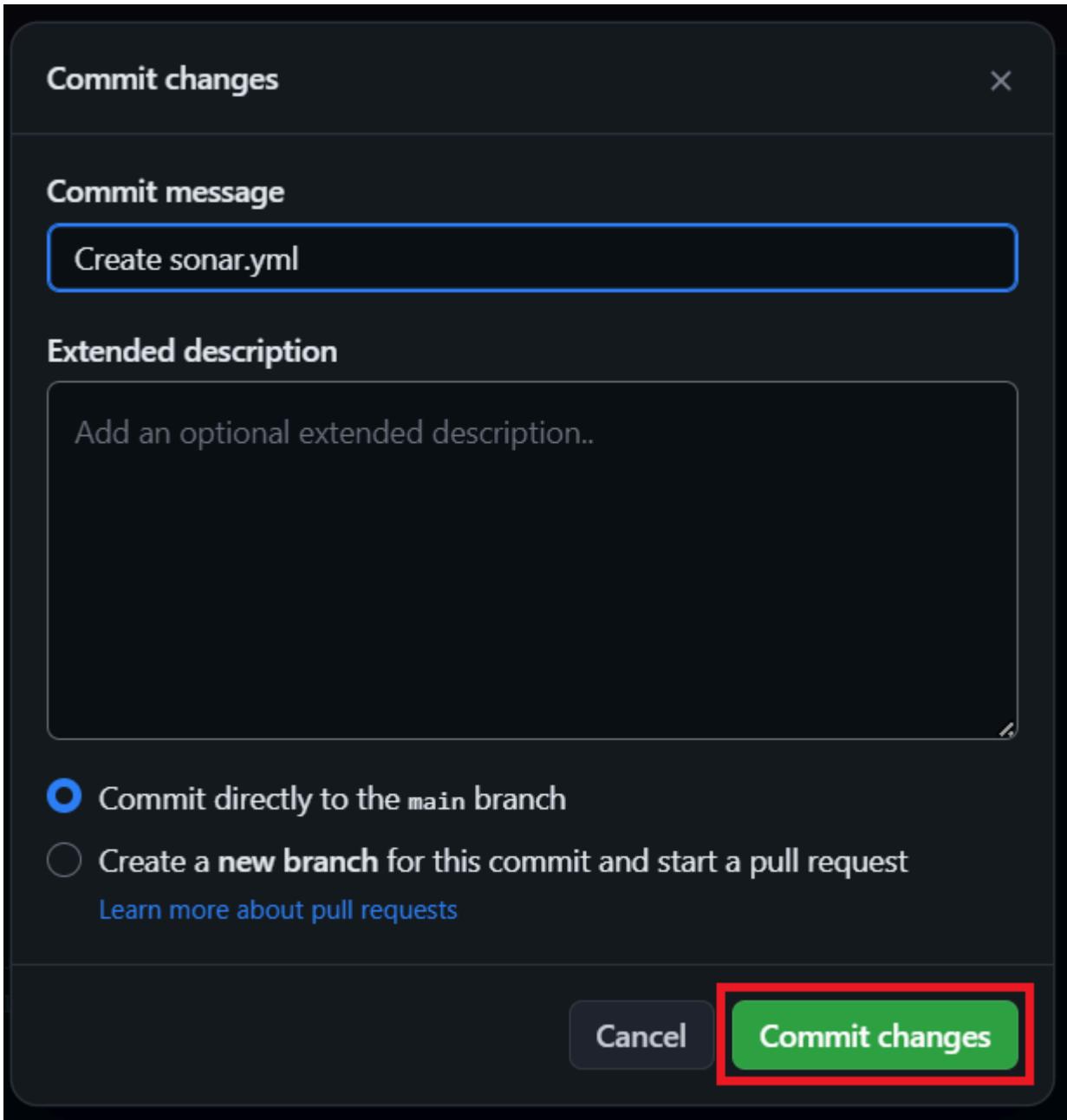


The screenshot shows the GitHub Actions workflow editor for a repository named 'Netflix-clone'. The workflow file is named 'sonaryml' and is located in the 'github / workflows' directory. The code in the file is:

```
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 }}
```

The 'Commit changes...' button at the top right of the editor is highlighted with a red box.

Click on Commit changes



Now workflow is created.

Click on Actions now

The screenshot shows the GitHub repository page for 'Netflix-clone'. The 'Actions' tab is selected, indicated by a red box. The page displays the status of the workflow: 'This branch is 18 commits ahead of gsbarure:main.' It lists a single workflow run by 'Aj7Ay' named 'Create sonar.yml'. The run was triggered 'now' and completed '56492ae now' with 113 commits. The workflow steps include '.github/workflows/Create sonar.yml' and 'Kubernetes/Create service.yml', both completed 'now' or 'last month'. The repository has 120 forks, 7 stars, and 0 watching. The 'About' section describes it as a 'Netflix Clone using React, Typescript, Material UI'.

Now it's automatically started the workflow

All workflows

2 workflow runs

Create sonar.yml #2

Triggered via push now Status: In progress

Aj7Ay pushed -> 56492ae main

Event: now Status: In progress Actor: Aj7Ay

Summary

Jobs

Build

Run details

Usage

Workflow file

Build 7s

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

Summary

Jobs

Build

Run details

Usage

Workflow file

Build Started 14s ago

Set up job 1s

Build sonarsource/sonarqube-scan-action@master 10s

Run actions/checkout@v2 3s

Run sonarsource/sonarqube-scan-action@master

Post Run actions/checkout@v2

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

Summary

Jobs

Build

Run details

Usage

Workflow file

Build succeeded now in 1m 7s

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_1711j1GN
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.

The screenshot shows a GitHub Actions build log for a 'Build' job. The log details the execution of a SonarQube scan action. It starts with 'Set up job' (1s), followed by 'Build sonarsource/sonarqube-scan-action@master' (10s). This step includes sub-tasks: 'Run actions/checkout@v2' (0s), 'Run sonarsource/sonarqube-scan-action@master' (53s), 'Post Run sonarsource/sonarqube-scan-action@master' (0s), 'Post Run actions/checkout@v2' (0s), and 'Complete job' (0s). A search bar at the top right is labeled 'Search logs'.

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

The screenshot shows the SonarQube dashboard with the 'Projects' tab selected. A single project, 'Netflix', is listed. The project status is 'Passed'. Key metrics shown include: Bugs (0 A), Vulnerabilities (0 A), Hotspots Reviewed (0.0% E), Code Smells (18 A), Coverage (0.0% O), Duplications (0.0% G), and Lines (3.2k S TypeScript...). A search bar at the top right is labeled 'Search for projects...'.

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

Step3: Let's scan files using Trivy

Add this code to your sonar.yml (I mean workflow)

```
- name: install trivy
  run: |
    #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
    echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasec...
    sudo apt-get update
    sudo apt-get install trivy -y
    #command to scan files
    trivy fs .
```

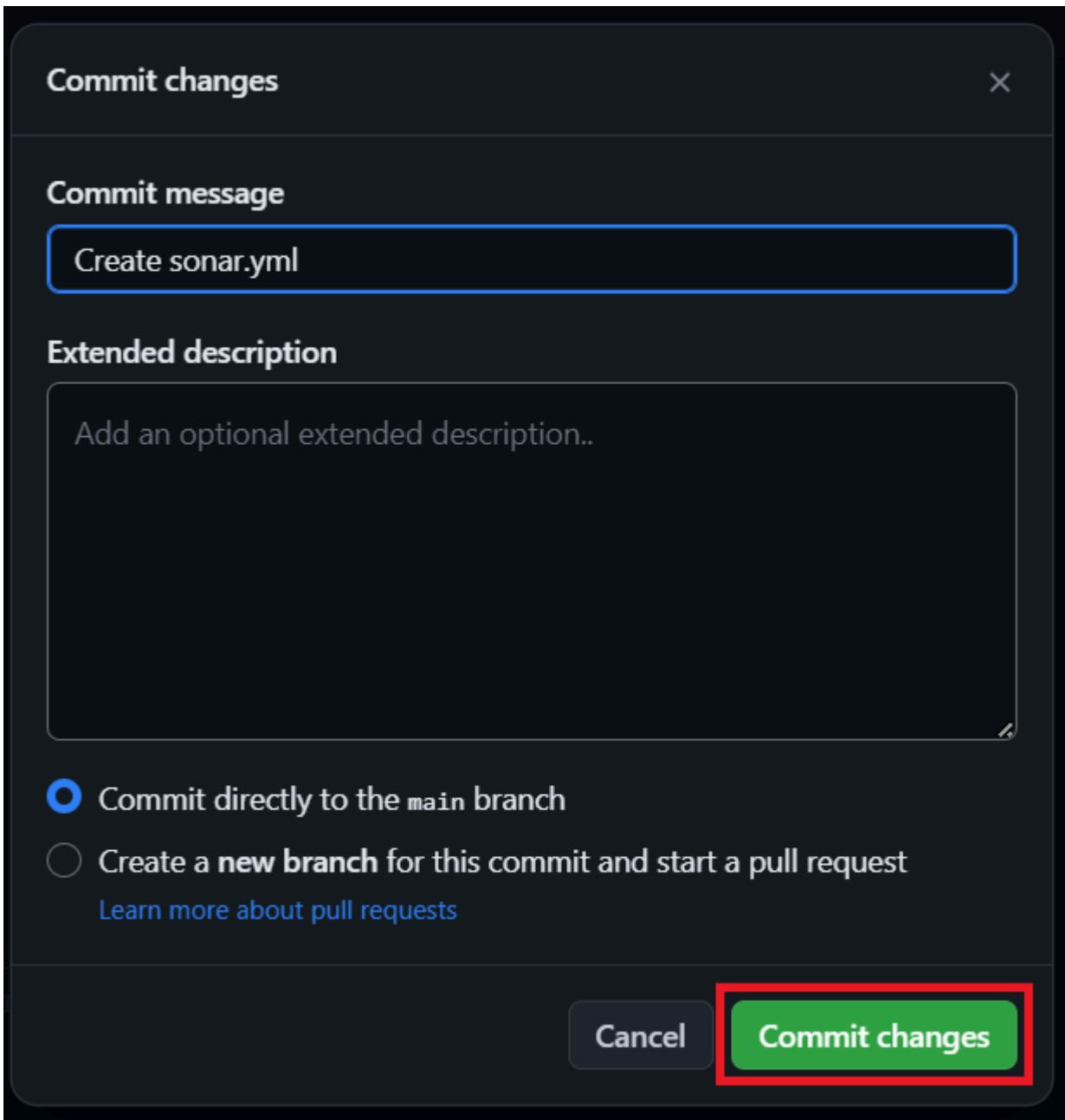
GitHub Actions workflow step that installs Trivy, a popular open-source vulnerability scanner for containers, and then uses it to scan the files.

I added a step in the workflow

The screenshot shows a GitHub Actions workflow configuration file named `sonar.yml`. The file is displayed in a code editor with tabs for "Edit" and "Preview". The code defines a workflow named `Build,Analyze,scan` that triggers on pushes to the `main` branch. It consists of two jobs: `build-analyze-scan` and `Build and analyze with SonarQube`. The first job uses the `actions/checkout@v2` action with shallow cloning disabled. The second job uses the `sonarsource/sonarqube-scan-action@master` action, setting environment variables `SONAR_TOKEN` and `SONAR_HOST_URL` from secrets. Both jobs run a series of commands to install Trivy and scan files.

```
name: Build,Analyze,scan
on:
  push:
    branches:
      - main
jobs:
  build-analyze-scan:
    name: Build
    runs-on: ubuntu-latest
    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 }}
      - name: Install trivy
        run:
          #Install trivy
          sudo apt-get update https gpgv 1sh release -y
          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
          #Scanning files
          trivy fs -
```

Commit changes



Click on actions again

The screenshot shows the GitHub repository page for 'Aj7Ay / Netflix-clone'. The 'Actions' tab is selected and highlighted with a red border. The page displays the status of a workflow named 'Aj7Ay Create sonar.yml'. It shows the workflow has 113 commits and was last updated 56492ae now. The repository details include 18 commits ahead of the 'main' branch, 1 branch, 0 tags, and 7 stars. The repository URL is listed as netflix-clone-react-typescript.vercel.app.

It started the workflow build

The screenshot shows the GitHub Actions interface for a workflow named "Update sonar.yml #5". The workflow has one job: "Build, Analyze, and Scan". The status is "Queued". The run was triggered via push now by user "Aj7Ay" on branch "main". The workflow file is "sonar.yml" and it runs on "push". A button labeled "Build, Analyze, and Scan" is visible.

Click on Build, Analyze and scan

The screenshot shows the GitHub Actions workflow log for the "Build, Analyze, and Scan" job. It highlights the step "Install Trivy" and the sub-step "Setting up trivy (0.46.0) ...". The log output shows Trivy unpacking and preparing to install version 0.46.0.

```

    75 (Reading database ... 50%
    76 (Reading database ... 55%
    77 (Reading database ... 60%
    78 (Reading database ... 65%
    79 (Reading database ... 70%
    80 (Reading database ... 75%
    81 (Reading database ... 80%
    82 (Reading database ... 85%
    83 (Reading database ... 90%
    84 (Reading database ... 95%
    85 (Reading database ... 100%
    86 (Reading database ... 225949 files and directories currently installed.)
    87 Preparing to unpack .../trivy_0.46.0_amd64.deb ...
    88 Unpacking trivy (0.46.0) ...
89 Setting up trivy (0.46.0) ...
90 NEEDRESTART-VER: 3.5
91 NEEDRESTART-KCUR: 6.2.0-1014-azure
92 NEEDRESTART-KEXP: 6.2.0-1014-azure
93 NEEDRESTART-KSTA: 1

```

It installed Trivy version 0.46.0 and scanned files also. See report

The screenshot shows the GitHub Actions workflow log for the "Build, Analyze, and Scan" job. It highlights the step "Scan files with Trivy" and the sub-step "Total: 1 (UNKNOWN: 0, LOW: 0, MEDIUM: 0, HIGH: 1, CRITICAL: 0)". The log output shows Trivy running a secret scanning command and detecting a vulnerability in a Yarn lock file.

```

1 ► Run # Replace the following line with the command to scan your specific files or directories
7 2023-10-19T03:02:28.821Z      INFO  Need to update DB
8 2023-10-19T03:02:28.821Z      INFO  DB Repository: ghcr.io/aquasecurity/trivy-db
9 2023-10-19T03:02:28.821Z      INFO  Downloading DB...
10 28.95 MiB / 40.53 MiB [-----> .....] 71.43% ? p/s ?40.53 MiB / 40.53 MiB [-----> .....]
-----> ] 100.00% ? p/s ?40.53 MiB / 40.53 MiB [-----> ] 100.00% ? p/s ?40.53 MiB /
40.53 MiB [-----> ] 100.00% 19.30 MiB p/s ETA 0s40.53 MiB / 40.53 MiB [-----> ] 100.00% 19.30 MiB p/s ETA 0s40.53 MiB / 40.53 MiB [-----> ]
> ] 100.00% 19.30 MiB p/s ETA 0s40.53 MiB / 40.53 MiB [-----> ] 100.00% 18.05 MiB p/s ETA 0s40.53 MiB / 40.53 MiB [-----> ] 100.00% 18.05 MiB p/s ETA 0s40.53 MiB / 40.53 MiB [-----> ]
18.05 MiB p/s ETA 0s40.53 MiB / 40.53 MiB [-----> ] 100.00% 24.83 MiB p/s 2023-10-19T03:02:31.167Z INFO  Vulnerability scanning is enabled
11 2023-10-19T03:02:31.167Z      INFO  Secret scanning is enabled
12 2023-10-19T03:02:31.167Z      INFO  If your scanning is slow, please try '--scanners vuln' to disable secret scanning
13 2023-10-19T03:02:31.167Z      INFO  Please see also https://aquasecurity.github.io/trivy/v0.46/docs/scanner/secret/#recommendation for faster secret detection
14 2023-10-19T03:02:31.202Z      INFO  Number of language-specific files: 1
15 2023-10-19T03:02:31.202Z      INFO  Detecting yarn vulnerabilities...
16
17 yarn.lock (yarn)
18
19 Total: 1 (UNKNOWN: 0, LOW: 0, MEDIUM: 0, HIGH: 1, CRITICAL: 0)
20
21
22 | Library | Vulnerability | Severity | Status | Installed Version | Fixed Version | Title |
23 |-----|-----|-----|-----|-----|-----|-----|
24 | CVE-2022-46175 | HIGH | fixed | 2.2.1 | 2.2.2, 1.0.2 | Prototype Pollution in JSON5 via Parse Method |

```

The file scan is completed, this is another security check

Step4A: Docker build and push to Dockerhub

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 account settings interface. The user is logged in as 'sevenajay'. The 'Security' tab is selected. A red box labeled '3' highlights the 'Security' tab in the sidebar. A red box labeled '2' highlights the 'Account Settings' option in the dropdown menu. A red box labeled '4' highlights the 'New Access Token' button.

Description	Scope	Last Used	Created	Active
An Ansible	Read, Write, Delete	Sep 16, 2023 08:07:09	Sep 15, 2023 18:08:25	Yes

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

The screenshot shows the 'New Access Token' dialog box. The 'Access Token Description' field contains 'Netflix' (highlighted with a red box). The 'Access permissions' dropdown is set to 'Read, Write, Delete' (highlighted with a red box). Below the permissions, a note states: 'Read, Write, Delete tokens allow you to manage your repositories.' At the bottom right are 'Cancel' and 'Generate' buttons, with 'Generate' highlighted with a red box.

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

The screenshot shows a GitHub repository page for 'Aj7Ay / Netflix-clone'. The 'Settings' tab is highlighted. The repository details show it's a public fork from 'gsbarure/netflix-clone-react-typescript'. The code tab shows 1 branch and 0 tags. The commit history shows 16 commits ahead of the main branch. The repository has 7 stars, 120 forks, and 0 watching. The 'About' section describes it as a 'Netflix Clone using React, Typescript, Material UI' and provides a link to 'netflix-clone-react-typescript.vercel.app'. The 'Releases' section is also visible.

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 shows 'Actions secrets and variables' with a note about encrypted secrets and non-sensitive variables. It has tabs for 'Secrets' (selected) and 'Variables'. Under 'Repository secrets', it says 'There are no secrets for this repository.' 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 options like General, Access, Collaborators, Moderation options, Code and automation, Branches, Tags, Rules, Actions, Webhooks, Environments, Codespaces, and Pages. The 'Name *' field is filled with 'DOCKERHUB_USERNAME' and the 'Secret *' field contains 'sevenajay'. A red box highlights the 'Add secret' button at the bottom.

Click on Add Secret.

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

Name



DOCKERHUB_TOKEN



This screenshot shows the same 'Actions secrets / New secret' page. The 'Name *' field now contains 'DOCKERHUB_TOKEN' and the 'Secret *' field contains 'dckr_pat_Qy-YtjVN3MNTfnGjHdnawLisjLU'. A red box highlights the 'Add secret' button at the bottom.

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

Step4B: Create a TMDB API Key

Next, we will create a TMDB API key

Open a new tab in the Browser and search for TMDB

The Movie Database (TMDB) is a popular, user editable database for movies and TV shows.

API Reference
Welcome to version 3 of The Movie Database (TMDB) API. This is ...

API key
TMDB Talk - API Terms of Use - API Reference - OAS - Basics

Login to your account
The Movie Database (TMDB) is a popular, user editable database ...

Popular Movies
Now Playing - Top Rated Movies - Upcoming Movies - ...

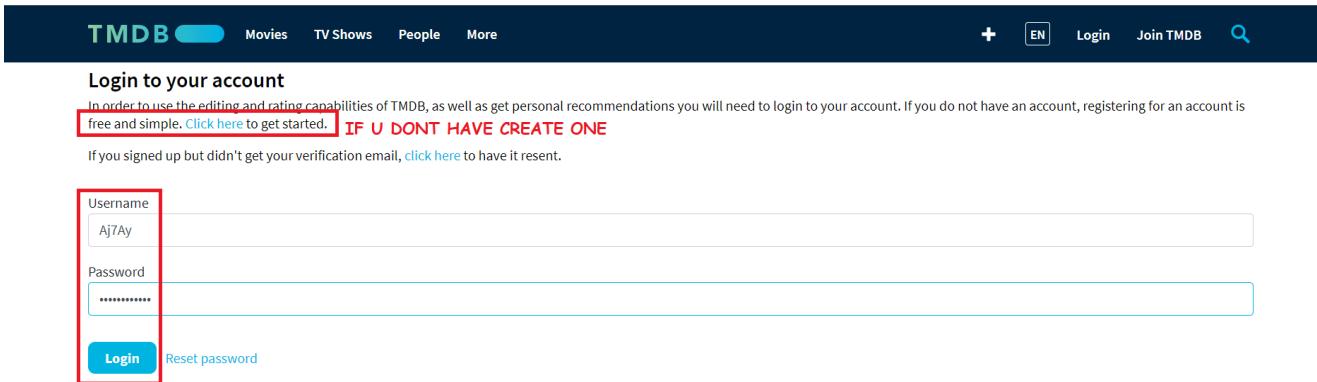
The Movie Database

- Website: themoviedb.org
- Category: film database
- Date launched: 2008
- Owner: Fan TV

Click on the first result, you will see this page

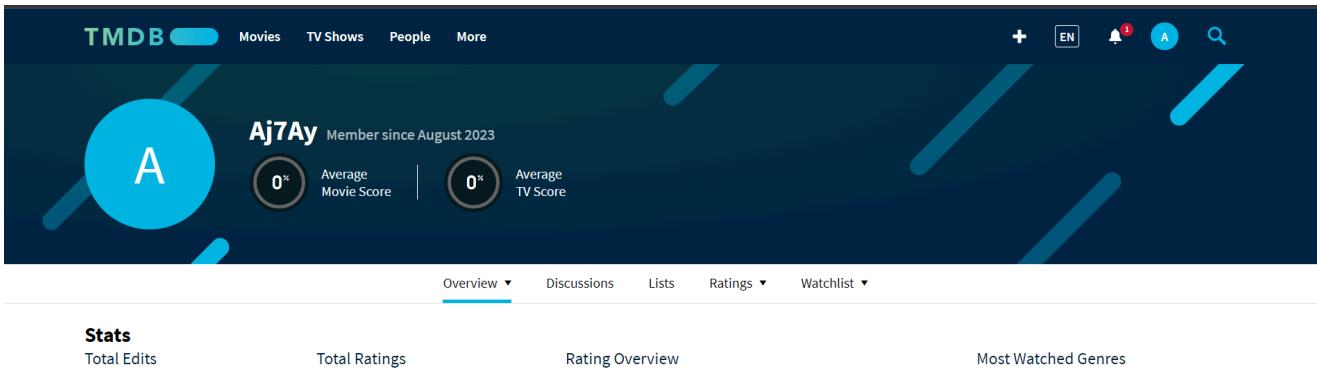
Click on the Login on the top right. You will get this page.

You need to create an account here. click on click here. I have an account that's why I added my details there.



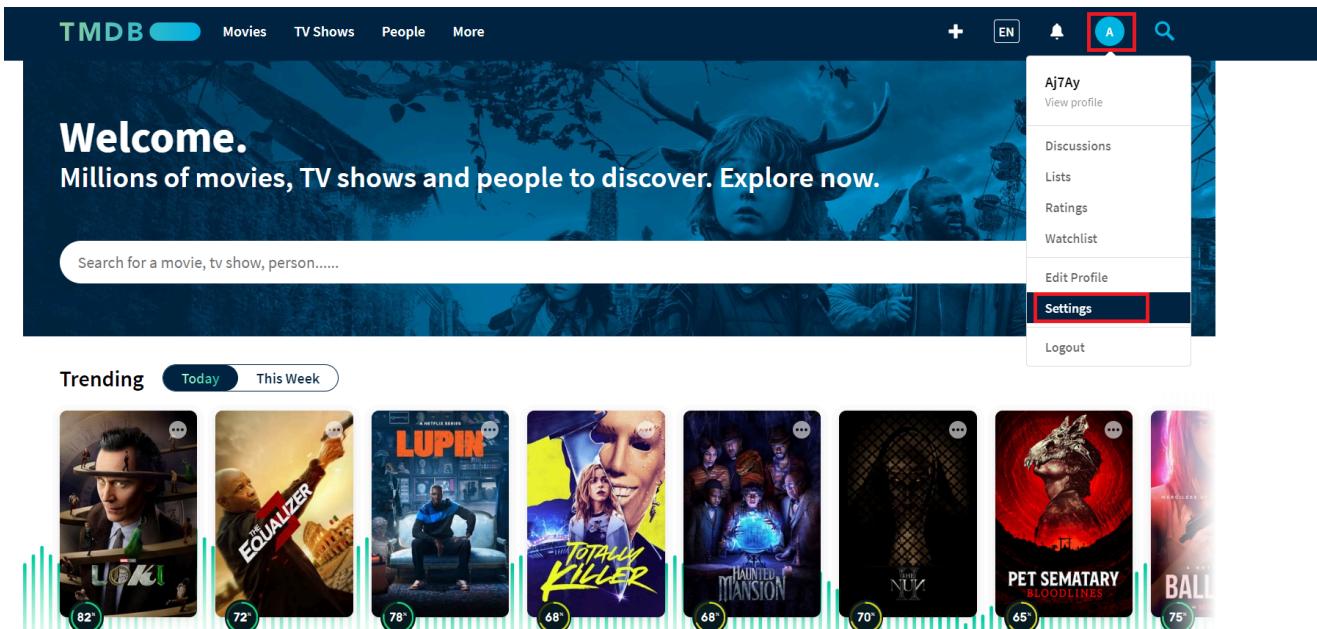
The screenshot shows the TMDB login page. At the top, there's a navigation bar with 'TMDB' and links for 'Movies', 'TV Shows', 'People', and 'More'. On the right are buttons for '+', 'EN', 'Login', 'Join TMDB', and a search icon. Below the navigation is a message: 'In order to use the editing and rating capabilities of TMDB, as well as get personal recommendations you will need to login to your account. If you do not have an account, registering for an account is free and simple. Click here to get started.' A red box highlights the text 'IF U DONT HAVE CREATE ONE'. Below this, it says 'If you signed up but didn't get your verification email, click here to have it resent.' A form for logging in is shown with fields for 'Username' (Aj7Ay) and 'Password' (redacted), and buttons for 'Login' (highlighted with a red box) and 'Reset password'.

once you create an account you will see this page.



The screenshot shows the TMDB profile page for 'Aj7Ay'. The header includes the TMDB logo, language switch ('EN'), a notification bell with '1', a user icon, and a search icon. Below the header, the profile section shows a large blue circle with a white letter 'A', the name 'Aj7Ay', and the text 'Member since August 2023'. It also displays two circular progress bars: 'Average Movie Score' at 0% and 'Average TV Score' at 0%. Below the profile section are tabs for 'Overview' (highlighted with a blue underline), 'Discussions', 'Lists', 'Ratings', and 'Watchlist'. Under the 'Overview' tab, there are sections for 'Stats' (Total Edits, Rating Overview, Most Watched Genres), 'Recent Activity' (Last edits, Last discussions, Last lists, Last ratings, Last watchlist), and 'Trending' (Today, This Week). A sidebar on the left lists 'Aj7Ay' (View profile), 'Discussions', 'Lists', 'Ratings', 'Watchlist', 'Edit Profile', 'Settings' (highlighted with a red box), and 'Logout'.

Let's create an API key, By clicking on your profile and clicking settings.



The screenshot shows the TMDB home page with a 'Welcome' banner and a search bar. Below the banner is a 'Trending' section with movie and TV show cards. On the right, a user profile menu is open for 'Aj7Ay', showing options like 'View profile', 'Discussions', 'Lists', 'Ratings', 'Watchlist', 'Edit Profile', 'Settings' (highlighted with a red box), and 'Logout'. The 'Settings' option is highlighted with a red box.

Now click on API from the left side panel.

The screenshot shows the TMDB Account Settings page. On the left, a sidebar menu includes 'Edit Profile', 'Account Settings' (which is currently selected), 'Streaming Services', 'Notification Settings', 'Blocked Users', 'Import List', 'Sharing Settings', 'Sessions', and 'API'. The 'API' option is highlighted with a red box. The main content area is titled 'Account Settings' and contains fields for 'Default Language' (set to English (en-US)), 'Fallback Language' (set to None (Don't Fallback)), 'Country' (set to India), 'Timezone - Auto detect?' (checked, set to Asia - Kabul), 'Include Adult Items in Search?' (set to No), and 'Filter Profanity?' (set to Yes).

Now click on create

The screenshot shows the TMDB API Overview page. The left sidebar includes 'Edit Profile', 'Account Settings', 'Streaming Services', 'Notification Settings', 'Blocked Users', 'Import List', 'Sharing Settings', 'Sessions', and 'API'. The 'API' option is highlighted with a purple box. The main content area has tabs for 'API', 'Overview', and 'Create', with 'Create' being the active tab. It contains text about TMDB's API service and a link to attribution logos. Below this are sections for 'Documentation' (link to developer.themoviedb.org) and 'Support' (link to support forums). A 'Request an API Key' section includes a link to generate a new key.

Click on Developer

The screenshot shows the TMDB API Create page. The left sidebar includes 'Edit Profile', 'Account Settings', 'Streaming Services', 'Notification Settings', 'Blocked Users', and 'Import List'. The 'API' option is highlighted with a purple box. The main content asks 'What type of API key do you wish to register?'. It shows two options: 'Developer' (selected, highlighted with a red box) and 'Professional'. The 'Developer' option includes a list: 'You are an individual', 'Your project is still in development', 'Your project is non profit', and 'Your project is ad supported'. The 'Professional' option includes a list: 'You represent a company', 'Your project is for profit (not ad supported)', and 'You are an OEM or hardware vendor'.

Now you have to accept the terms and conditions.

12. General Terms

- 1. Relationship of the Parties.** Notwithstanding any provision hereof, for all purposes of the Terms of Use, you and TMDB shall be and act independently and not as partner, joint venturer, agent, employee or employer of the other. You shall not have any authority to assume or create any obligation for or on behalf of TMDB, express or implied, and you shall not attempt to bind TMDB to any contract.
- 2. Invalidity of Specific Terms.** If any provision of the Terms of Use is found by a court of competent jurisdiction to be invalid, the parties nevertheless agree that the other provisions of this agreement will remain in full force and effect and the court should endeavor to give effect to the parties' intentions as reflected in the invalid provision.
- 3. Location of Lawsuit and Choice of Law.** THE TERMS OF USE AND THE RELATIONSHIP BETWEEN YOU AND TMDB SHALL BE GOVERNED BY THE LAWS OF THE STATE OF CALIFORNIA WITHOUT REGARD TO ITS CONFLICT OF LAW PROVISIONS. YOU AND TMDB AGREE TO SUBMIT TO THE PERSONAL JURISDICTION OF THE COURTS LOCATED WITHIN THE COUNTY OF SAN MATEO, CALIFORNIA.
- 4. No Waiver of Rights by TMDB.** TMDB's failure to exercise or enforce any right or provision of the Terms of Use shall not constitute a waiver of such right or provision.
- 5. No Transfer.** The rights and obligations of these Terms of Use is personal to you and may not be transferred by you, either voluntarily or by operation of law.
- 6. Notice.** Any notice to be sent to you under these Terms of Use may be sent via email, post, or any other reasonable means, at the contact information provided by you to TMDB from time to time. It is your obligation to insure that this information is current.

Miscellaneous. The section headings and subheadings contained in this agreement are included for convenience only, and shall not limit or otherwise affect the terms of the Terms of Use. Any construction or interpretation to be made of the Terms of Use shall not be construed against the drafter. The Terms of Use constitute the entire agreement between TMDB and you with respect to the subject matter hereof.

This Agreement was last updated on: July 28, 2014.

Cancel

Accept

CW

Provide basic details

Settings

- Edit Profile
- Account Settings
- Streaming Services
- Notification Settings
- Blocked Users
- Import List
- Sharing Settings
- Sessions

API Overview Create

Type of Use

Desktop Application

Application Name

dEM

Application URL

NOT AVAILABLE

Application Summary

jdf

The screenshot shows the GitHub API settings page. On the left, there's a sidebar with options like 'Sharing Settings', 'Sessions', 'API' (which is selected), and 'Delete Account'. The main area has fields for 'First Name' (Code), 'Last Name' (Word), 'Email Address' (redacted), 'Address 1' (ddddd), 'City' (fffffssss), 'Zip Code' (225588), and a dropdown for 'Phone Number (no parenthesis or dashes)' with 'India' selected. A red box highlights the 'Submit' button.

Click on submit and you will get your API key.

The screenshot shows the GitHub API settings page. On the left, there's a sidebar with options like 'Notification Settings', 'Blocked Users', 'Import List', 'Sharing Settings', 'Sessions', 'API' (selected), and 'Delete Account'. The main area has sections for 'Support' (with a link to forums) and 'API Details' (with a link to edit details). Below these is a section for 'API Key' where the value '8f757ea1c6e3dc2deef92fb4682d7e78' is displayed in a red-bordered box. At the bottom, there's a section for 'API Read Access Token' with a long token value.

Let's add the below step to the workflow

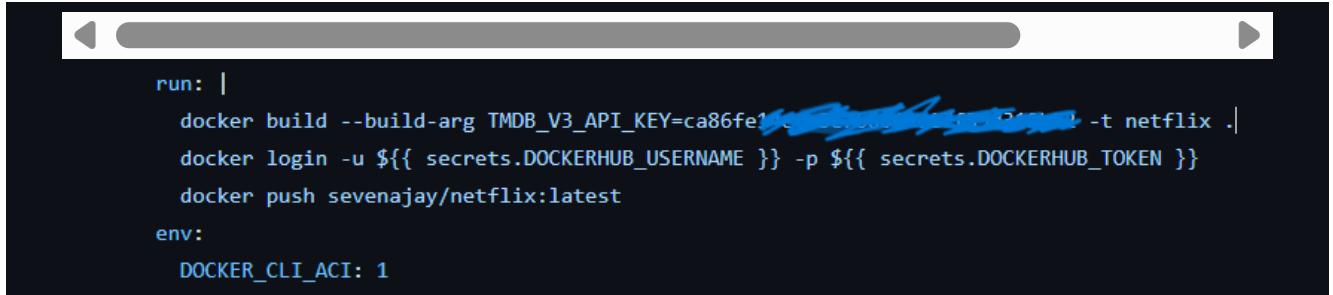
You have to add API at the Build command

Change your username also

```
- name: Docker build and push
  run: |
    #run commands to build and push docker images
    docker build --build-arg TMDB_V3_API_KEY=<USE YOUR API KEY> -t netfli
    docker tag netflix sevenajay/netflix:latest
    docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_PASSWORD }}
```

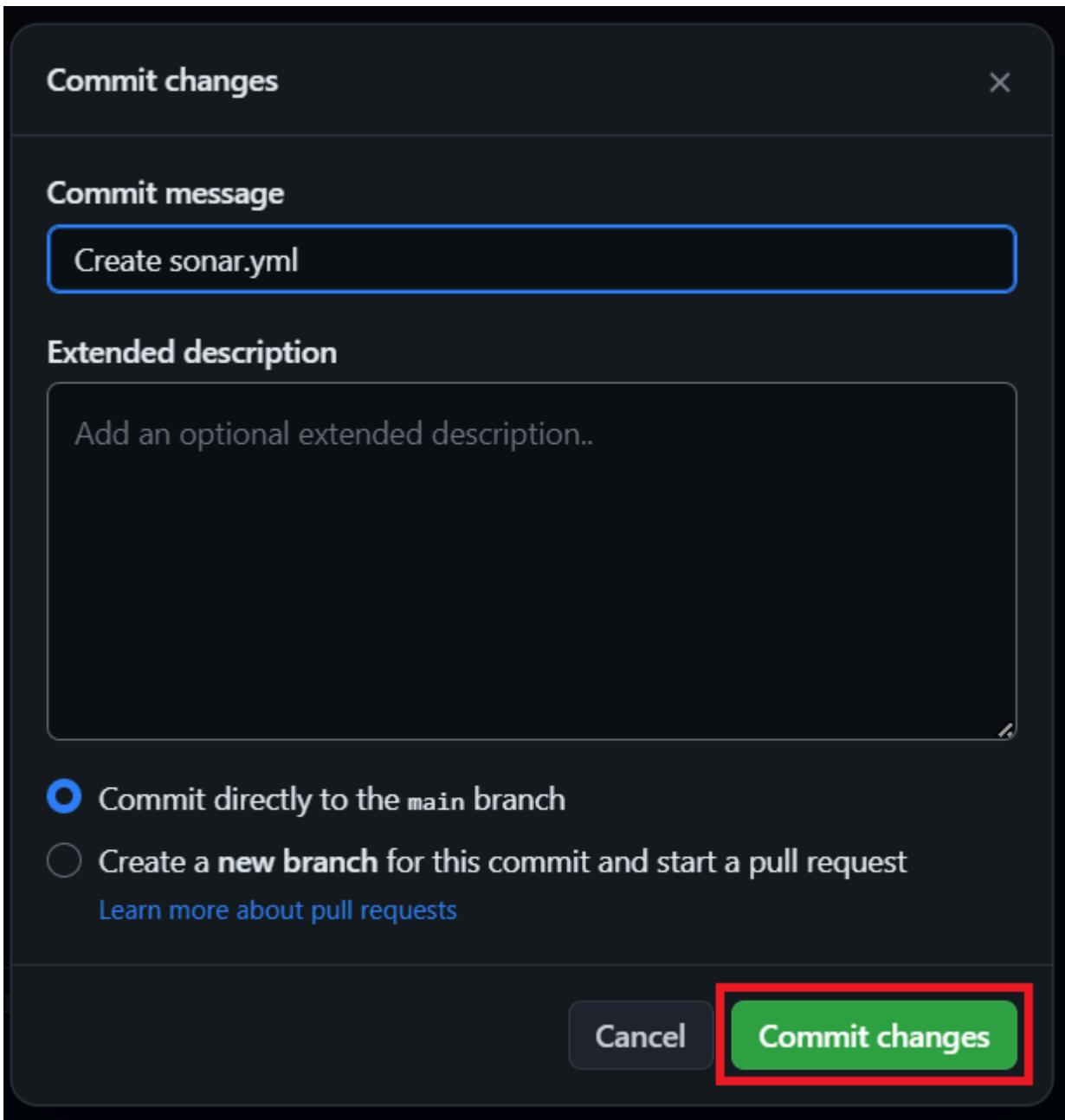
```
docker push sevenajay/netflix:latest  
env:  
DOCKER_CLI_ACI: 1
```

You can see the image, I already added



```
run: |  
  docker build --build-arg TMDB_V3_API_KEY=ca86fe1... -t netflix .  
  docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_TOKEN }}  
  docker push sevenajay/netflix:latest  
env:  
DOCKER_CLI_ACI: 1
```

Let's commit changes



Click on actions again

The screenshot shows the GitHub repository page for 'Netflix-clone'. The 'Actions' tab is selected, indicated by a red border. The page displays the status of a workflow run. It shows '18 commits ahead of gsbarure:main'. A specific commit by 'Aj7Ay' titled 'Create sonar.yml' is highlighted, showing it was created 'now' and has '113 commits'. The commit details show files like '.github/workflows' and 'Kubernetes' with their respective creation times. The repository summary on the right indicates it's a 'Netflix Clone using React, Typescript, Material UI' with 7 stars and 0 watching.

It started the workflow build

The screenshot shows the GitHub Actions interface for a workflow named "Update sonar.yml #5". The "Summary" tab is selected. A single job, "Build, Analyze, and Scan", is listed under the "sonar.yml" section. The status is "Queued". The job was triggered via a push to the main branch by user Aj7Ay. The total duration and artifacts are both listed as "-". There is a "Build, Analyze, and Scan" button at the bottom of the job card.

Click on Build, Analyze and Scan. You will see this Docker image is building now

The screenshot shows the GitHub Actions logs for the "Build and Push Docker Image" job. The logs start with "Started 1m 51s ago" and show the command-line output of the build process. The logs include commands like `COPY ./package.json .`, `RUN yarn install`, and `COPY ./yarn.lock .`. The build completed successfully after 49 seconds.

```

Build and Push Docker Image
Started 1m 51s ago
49s
Build Docker image
86 #12 [builder 2/7] WORKDIR /app
87 #12 DONE 0.0s
88 #13 [builder 3/7] COPY ./package.json .
89 #13 DONE 0.0s
90 #14 [Builder 4/7] COPY ./yarn.lock .
91 #14 DONE 0.0s
92 #15 [builder 5/7] RUN yarn install
97 #15 0.381 yarn install v1.22.19
98 #15 0.455 [1/4] Resolving packages...
99 #15 0.624 [2/4] Fetching packages...
100 #15 23.96 [3/4] Linking dependencies...
101 #15 23.96 warning "@emotion/react > @emotion/babel-plugin@11.10.5" has unmet peer dependency "@babel/core@^7.0.0".
102 #15 23.96 warning "@emotion/react > @emotion/babel-plugin > @babel/plugin-syntax-jsx@7.18.6" has unmet peer dependency "@babel/core@^7.0.0-0".
103 #15 23.97 warning " > slick-carousel@1.8.1" has unmet peer dependency "jquery@>1.8.0".
104 #15 33.87 [4/4] Building fresh packages...
105 #15 34.06 success Saved lockfile.
106 #15 34.07 Done in 33.70s,
107 #15 DONE 34.8s
108 #16 [Builder 6/7] COPY .
109 #16 DONE 0.0s
110 #17 [builder 7/7] RUN yarn build
113 #17 0.491 yarn run v1.22.19
114 #17 0.531 $ tsc && vite build

```

Build Succeeded

The screenshot shows the GitHub Actions summary for the "Build and Push Docker Image" job. The status is "succeeded now in 2m 17s". The job history shows the following steps: Set up job, Build sonarsource/sonarqube-scan-action@master, Checkout code, Build and analyze with SonarQube, Build Docker image, Post Build and analyze with SonarQube, Post Checkout code, and Complete job. All steps are marked as successful.

If you go to your Docker hub, you will find that the image is pushed to Dockerhub

The screenshot shows the Dockerhub repository page for "sevenajay/netflix". The repository has 1 tag. The Docker commands section shows the command "docker push sevenajay/netflix:tagname". A "Public View" button is available. The repository details include a description placeholder, a last push timestamp, and a "Edit" icon.

Step5A: Add a self-hosted runner to Ec2

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

The screenshot shows the GitHub repository settings for 'Netflix-clone'. The 'Actions' section is selected, and the 'Runners' sub-section is highlighted with a red box. The 'General' tab is selected under 'Actions'. The 'Default branch' section shows 'main' as the default branch.

Click on New self-hosted runner

The screenshot shows the 'Runners' page under the 'Actions' section. A green button labeled 'New self-hosted runner' is highlighted with a red box. The page displays a message: 'Host your own runners and customize the environment used to run jobs in your GitHub Actions workflows. [Learn more about self-hosted runners](#)'. Below this, it says 'There are no runners configured' and provides a link to 'Learn more about using runners'.

Now select Linux and Architecture X64

The screenshot shows the 'Runners / Add new self-hosted runner' page. On the left, there's a sidebar with options like General, Access, Collaborators, Moderation options, Code and automation, Branches, Tags, Rules, Actions, Webhooks, and Environments. The main area has sections for 'Runner image' (with radio buttons for macOS, Linux, and Windows, where Linux is selected), 'Architecture' (with a dropdown menu set to x64), and a 'Download' button.

Use the below commands to add a self-hosted runner

The screenshot shows the 'Download' section of the GitHub Actions Runners page. It contains three code snippets:

- Create**: Shell commands to download and extract the runner package.
- Configure**: Shell commands to run the configuration script.
- Using your self-hosted runner**: YAML code for a workflow file to run jobs on the self-hosted runner.

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$  
ubuntu@ip-172-31-32-28:~/actions-runner$  
ubuntu@ip-172-31-32-28:~/actions-runner$ 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  
% Total    % Received % Xferd  Average Speed   Time   Time  Current  
          Dload Upload   Total Spent  Left Speed  
0     0      0       0      0      0      0      0      0  
100 178M 100 178M 0      0 82.7M 0 0:00:02 0:00:02 119M  
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$ 
ubuntu@ip-172-31-32-28:~/actions-runner$ 
ubuntu@ip-172-31-32-28:~/actions-runner$ echo "fb28a1c3715e0a6c5051af0e6eff9c255009e2eec6fb08bc2708277fb49f3" actions-runner-linux-x64-2.310.2.tar.gz | 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$ 
```

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
```

Step5B: Final workflow to run the container

Let's add a deployment workflow



```
deploy:
  needs: build-analyze-scan
  runs-on: [aws-netflix]
  steps:
    - name: Pull the docker image
      run: docker pull sevenajay/netflix:latest
    - name: Trivy image scan
      run: trivy image sevenajay/netflix:latest
    - name: Run the container netflix
      run: docker run -d --name netflix -p 8081:80 sevenajay/netflix::
```



```

deploy:
  needs: build-and-push
  runs-on: [aws-netflix] LABEL NAME I PROVIDED FOR RUNNER
  steps:
    - name: Pull image from docker hub
      run: docker pull sevenajay/youtube:latest PULLS DOCKER IMAGE
    - name: scan image
      run: trivy image sevenajay/youtube:latest SCANS OUR IMAGE
    - name: Run docker container
      run: docker run -d --name netflix -p 8081:80 sevenajay/netflix:latest STARTS CONTAINER

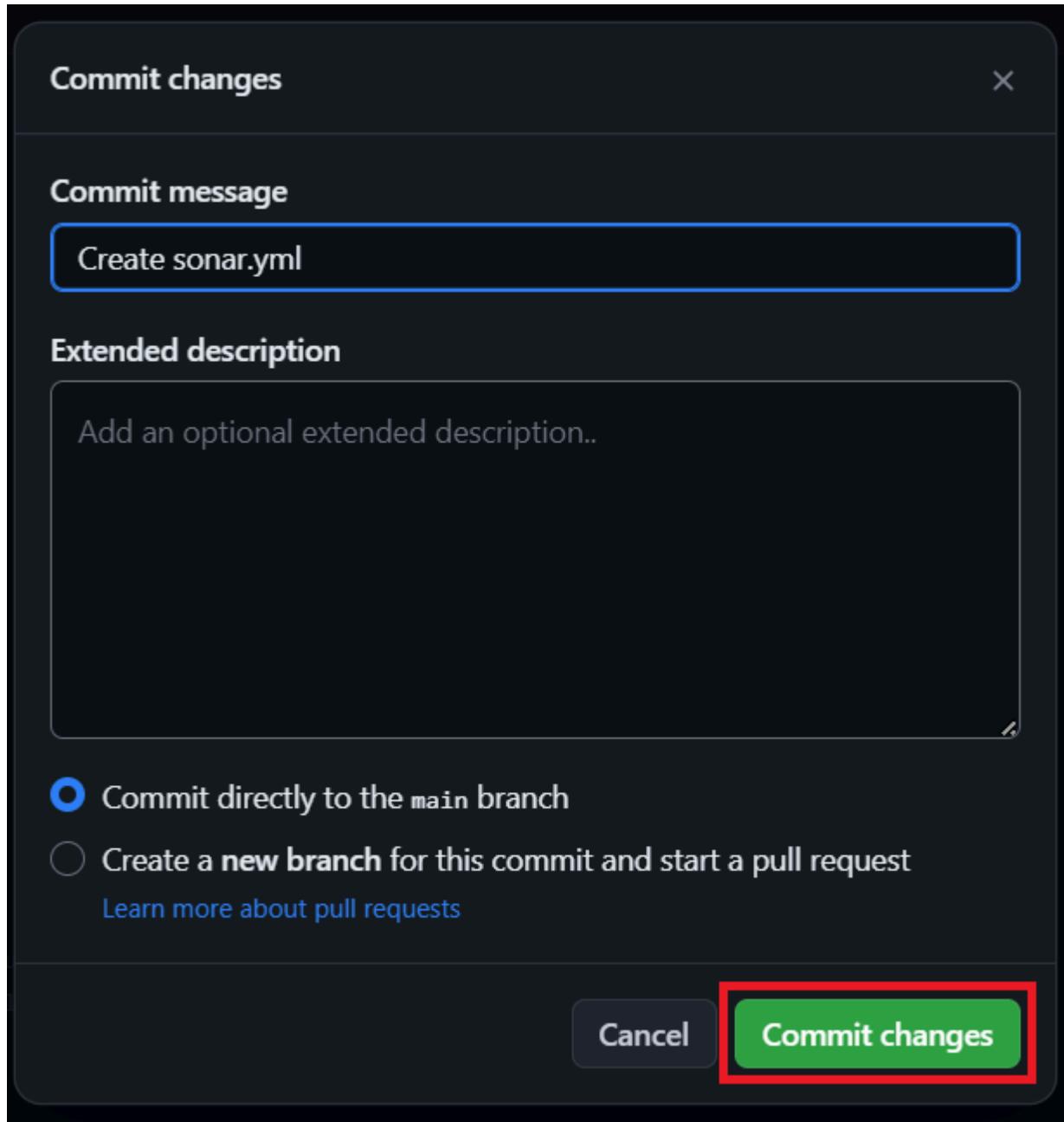
```

1. **deploy:** : This is the name of a workflow or job, likely associated with a CI/CD pipeline. It specifies what should happen when this deployment job is triggered.
2. **needs: build-analyze-scan** : This line indicates that this deployment job depends on the successful completion of a previous job named “build-analyze-scan.” In other words, it waits for “build-analyze-scan” to finish before starting.
3. **runs-on: [aws-netflix]** : This job is set to run on a specific type of runner or environment, labeled as “aws-netflix.” Runners are the environments where jobs are executed, and “aws-netflix” suggests that this deployment might be intended for an AWS-based infrastructure.
4. **steps :** : This section lists the individual steps or tasks to be executed as part of the deployment job.
 - **name: Pull the docker image** : This step has a descriptive name. It uses the docker pull command to fetch a Docker image labeled “sevenajay/netflix:latest.” This is a common step in container-based deployments, where it ensures that the latest version of the Docker image is available locally.
 - **name: Trivy image scan** : This step performs a security scan on the Docker image “sevenajay/netflix:latest” using a tool called Trivy. Trivy is used for vulnerability scanning of container images.
 - **name: Run the container netflix** : This step starts a Docker container named “netflix” using the image “sevenajay/netflix:latest.” It runs the container in detached mode (“-d”) and maps port 8081 on the host to port 80 in the container, making the service accessible via port 8081 on the host.

This workflow is designed to automate the deployment of a Docker container, with checks for the latest image, a security scan, and launching the container. The success of this job depends on the success of the preceding “build-analyze-scan”

job, and it's meant to be executed on the specified runner, possibly in an AWS environment.

Commit changes

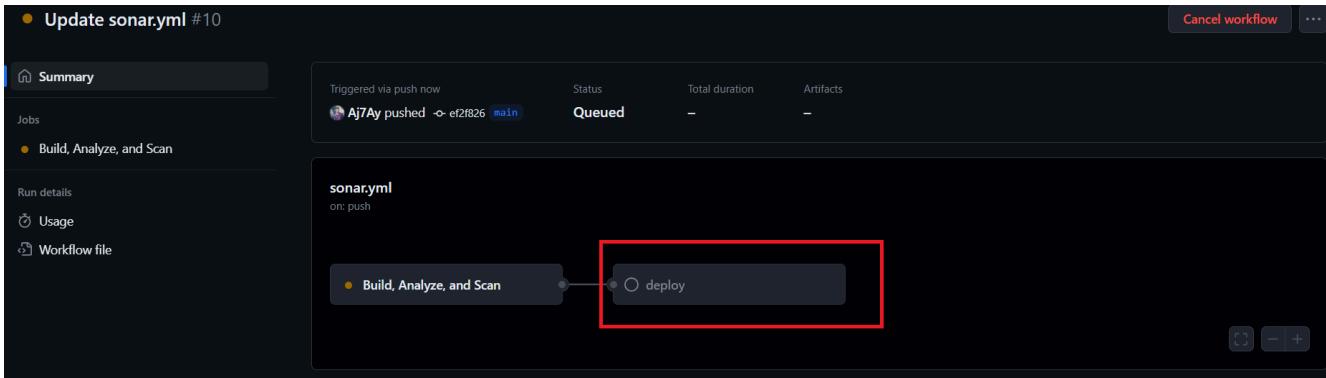


Click on actions again

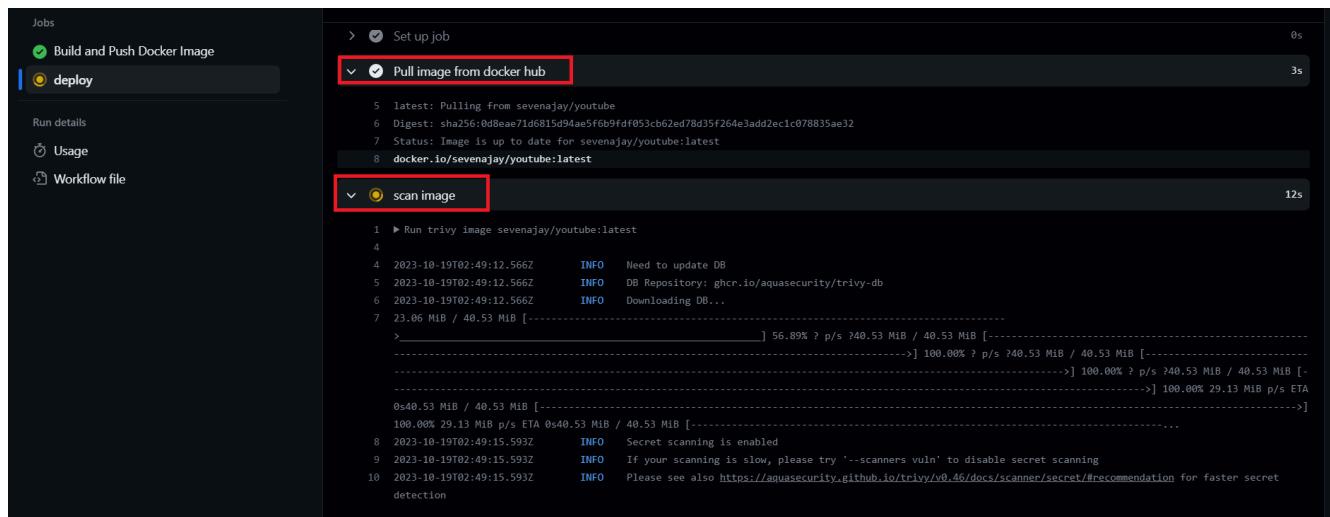
You will see two different Jobs now

Click on Build and Push docker image Build (using the above image), you will see this once the first job completes

Now come back by clicking on Summary and click on Deploy now



You can see how it's pulling image and scanning image



It starts running the job on your Ec2 instance

```
✓ Connected to GitHub

Current runner version: '2.310.2'
2023-10-19 02:43:51Z: Listening for Jobs
2023-10-19 02:49:02Z: Running job: deploy
```

Now it's completed running the container

```

deploy
succeeded 1 minute ago in 3m 39s

> ⚡ scan image
    ✓ Run docker container
    ✓ Run docker run -d --name netflix -p 8081:80 sevenajay/netflix:latest
      9 Unable to find image 'sevenajay/netflix:latest' locally
      10 latest: Pulling from sevenajay/netflix
      11 9398808236ff: Pulling fs layer
      12 708173787fc8: Pulling fs layer
      13 b5b131b0c886: Pulling fs layer
      14 ab69664ce136: Pulling fs layer
      15 d7f3c29ebbc5: Pulling fs layer
      16 80bb00910f42: Pulling fs layer
      17 ba13ff899438: Pulling fs layer
      18 4f4fb700ef54: Pulling fs layer
      19 4b7da8813c60: Pulling fs layer
      20 d73ddab721a5: Pulling fs layer
      21 ab69664ce136: Waiting
      22 d7f3c29ebbc5: Waiting
      23 80bb00910f42: Waiting
      24 ba13ff899438: Waiting
      25 4f4fb700ef54: Waiting
      26 4b7da8813c60: Waiting
      27 d73ddab721a5: Waiting
      28 b5b131b0c886: Verifying Checksum
      29 b5b131b0c886: Download complete
      30 708173787fc8: Verifying Checksum
      31 708173787fc8: Download complete
      32 9398808236ff: Verifying Checksum
      33 9398808236ff: Download complete
      34 9398808236ff: Pull complete

```

You will see this in the instance

```

Current runner version: '2.310.2'
2023-10-19 02:43:51Z: Listening for Jobs
2023-10-19 02:49:02Z: Running job: deploy
2023-10-19 02:52:46Z: Job deploy completed with result: Succeeded

```

On GitHub, you will see this. the build succeeded

Triggered via push 9 minutes ago	Status	Total duration	Artifacts
Aj7Ay pushed → 2d7cf13 main	Success	7m 1s	-

sonar.yml
on: push

```

Build and Push Docker I... 3m 1s → deploy 3m 39s

```

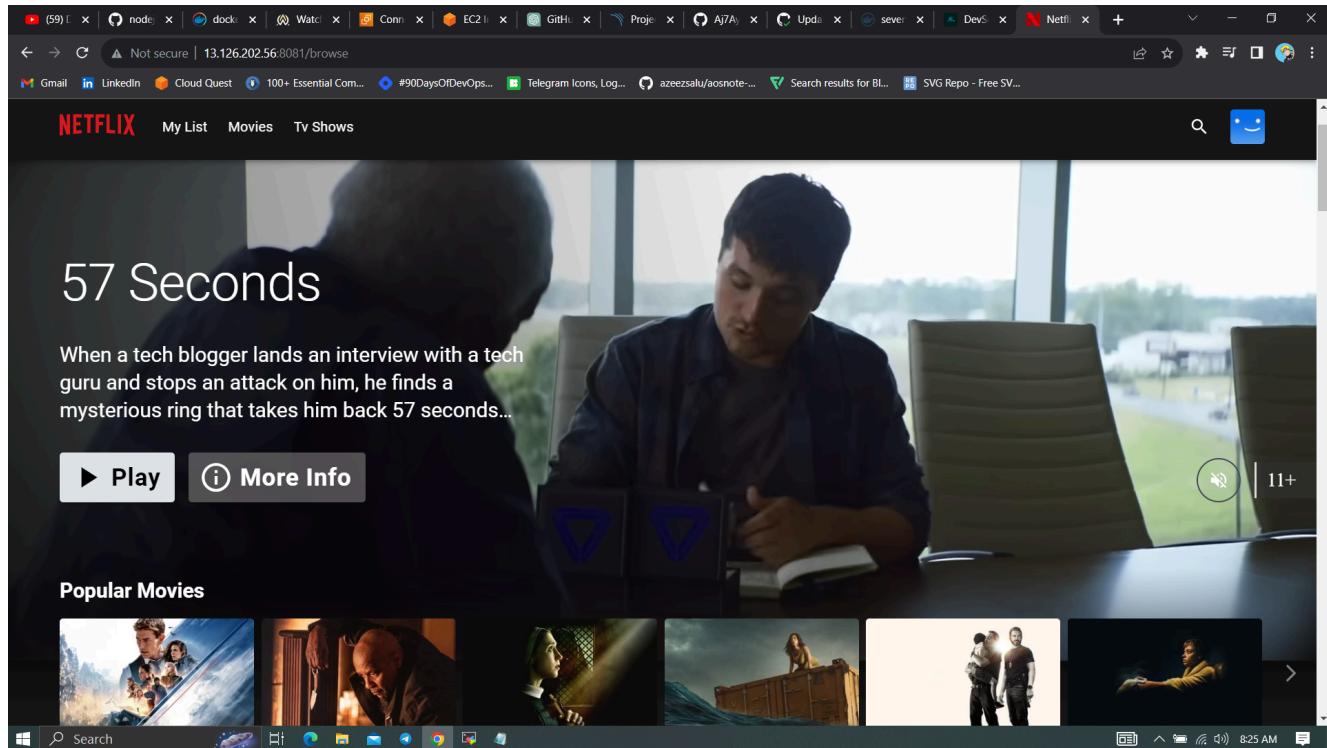
Now copy your ec2 instance ip and go to the browser



Ec2-instance-ip:8081



You will see Netflix app will run



Deployment is done.

FULL WORKFLOW



```

name: Build,Analyze,scan
on:
  push:
    branches:
      - main
jobs:
  build-analyze-scan:
    name: Build
    runs-on: ubuntu-latest
    steps:
      - name: Checkout code
        uses: actions/checkout@v2
        with:
          fetch-depth: 0 # Shallow clones should be disabled for a beta
      - 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: install trivy
  run: |
    #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 /etc/apt/trivy-trusty.gpg > /dev/null
    echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb/ /" | sudo tee /etc/apt/sources.list.d/trivy.list > /dev/null
    sudo apt-get update
    sudo apt-get install trivy -y
    #scanning files
    trivy fs .
```

```
- name: Docker build and push
  run: |
    #run commands to build and push docker images
    docker build --build-arg TMDB_V3_API_KEY=Aj7ay86fe14eca3e7686f...
    docker tag netflix sevenajay/netflix:latest
    docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_PASSWORD }}
    docker push sevenajay/netflix:latest
```

```
env:
  DOCKER_CLI_ACI: 1
```

```
deploy:
  needs: build-analyze-scan
  runs-on: [aws-netflix]
  steps:
    - name: Pull the docker image
      run: docker pull sevenajay/netflix:latest
    - name: Trivy image scan
      run: trivy image sevenajay/netflix:latest
    - name: Run the container netflix
      run: docker run -d --name netflix -p 8081:80 sevenajay/netflix:latest
```

```

1 name: Build,Analyze,scan
2
3 on:
4   push:
5     branches:
6       - main
7
8
9 jobs:
10   build-analyze-scan:
11     name: Build
12     runs-on: ubuntu-latest
13     steps:
14       - name: Checkout code
15         uses: actions/checkout@v2
16         with:
17           fetch-depth: 0 # Shallow clones should be disabled for a better relevancy of analysis
18
19       - name: Build and analyze with SonarQube
20         uses: sonarsource/sonarqube-scan-action@master
21         env:
22           SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
23           SONAR_HOST_URL: ${{ secrets.SONAR_HOST_URL }}
24
25       - name: install trivy
26         run: |
27           #Install trivy
28           sudo apt-get install wget apt-transport-https gnupg lsb-release -y
29           wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee /usr/share/keyrings/trivy.gpg > /dev/null
30           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
31           sudo apt-get update
32           sudo apt-get install trivy -y
33           #scanning files
34           trivy fs .
35
36     - name: Docker build and push
37       run: |
38         #run commands to build and push docker images
39         docker build --build-arg TMDB_V3_API_KEY=Aj7ay86fe14eca3e76869b92 -t netflix .
40         docker tag netflix sevenajay/netflix:latest
41         docker login -u ${{ secrets.DOCKERHUB_USERNAME }} -p ${{ secrets.DOCKERHUB_TOKEN }}
42         docker push sevenajay/netflix:latest
43         env:
44           DOCKER_CLI_ACI: 1
45
46   deploy:
47     needs: build-analyze-scan
48     runs-on: [aws-netflix]
49     steps:
50       - name: Pull the docker image
51         run: docker pull sevenajay/netflix:latest
52       - name: Trivy image scan
53         run: trivy image sevenajay/netflix:latest
54       - name: Run the container netflix
55         run: docker run -d --name netflix -p 8081:80 sevenajay/netflix:latest

```

Clear the instance.

I hope you found this blog insightful and that you've learned something new about how GitHub Actions can supercharge your Netflix deployments through the lens of DevSecOps. As technology evolves, staying informed and adaptable is key to thriving in the world of software development. If you have any questions or would like to share your thoughts, feel free to reach out. Your feedback and engagement are invaluable as we continue to explore and embrace the exciting innovations in the tech landscape. Thank you for joining me on this journey of discovery!



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

One response to “GitHub Actions: Netflix Deployment Powered by DevSecOps”



Artem

16 May 2024

Hello,

This is really cool project. I've learned some new features here. Keep up the good work.

Full workflow has one little issue in deploy job. The step named “Trivy image scan” is not running correctly. I think that is because of trivy missing on our ec2 (self-hosted runner).

Thanks for your time.

Regards,

Artem

[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](#)

© 2024 · Powered by [Mr Cloud Book](#)

[Follow Us on YouTube](#)