**Graded Project on Building CI-CD Pipeline Tool**

Create a complete CI-CD pipeline using bash, python and crontabs. The list of tasks is specified below:

* Task 1: Set Up a Simple HTML Project
  + - Create a simple HTML project and push it to a GitHub repository.
* Task 2: Set Up an AWS EC2/Local Linux Instance with Nginx
* Task 3: Write a Python Script to Check for New Commits
  + - Create a Python script to check for new commits using the GitHub API.
* Task 4: Write a Bash Script to Deploy the Code
  + - Create a bash script to clone the latest code and restart Nginx.
* Task 5: Set Up a Cron Job to Run the Python Script
  + - Create a cron job to run the Python script at regular intervals.
* Task 6: Test the Setup
  + - Make a new commit to the GitHub repository and check that the changes are automatically deployed.

**Task 1: Set Up a Simple HTML Project**

Create a simple HTML project and push it to a GitHub repository.

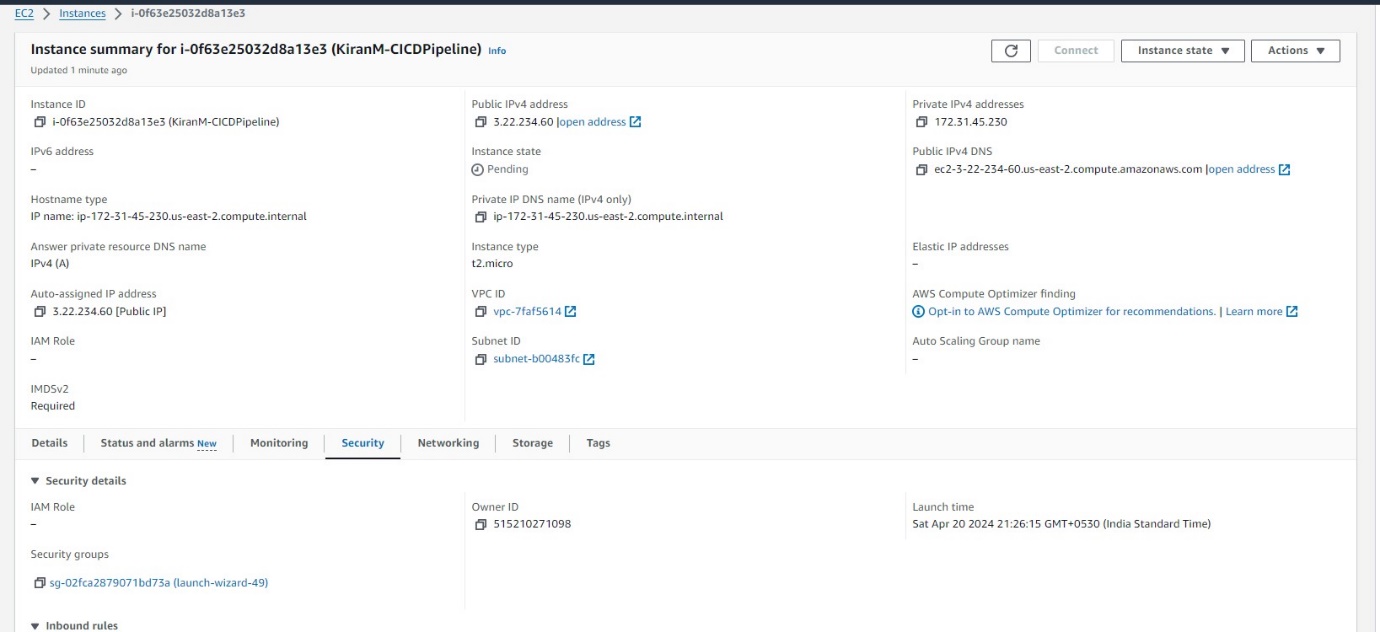
Please find the html file in github here : <https://github.com/mahalekiran/CICD-Pipeline/blob/dev/Hello-World.html>

Followed the steps as below:

1. Created a new file Hello-World.html and added content.
2. Execute the command git status
3. Execute the command git add .
4. Execute the command git commit -m “Made modifications in html”
5. Execute the command git push

**Task 2: Set Up an AWS EC2/Local Linux Instance with Nginx**

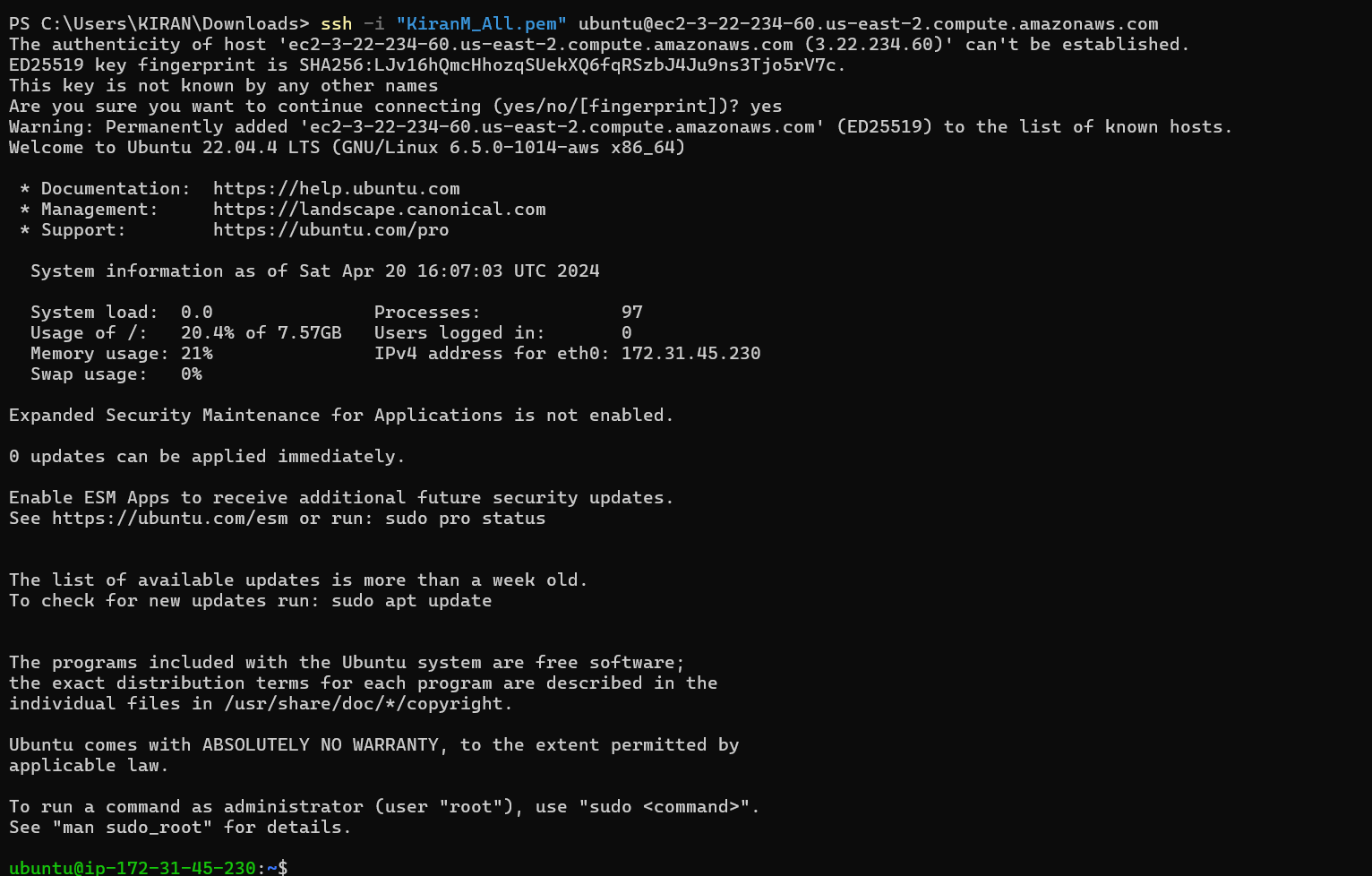
1. Login to AWS
2. Go to EC2 services and create a new instance



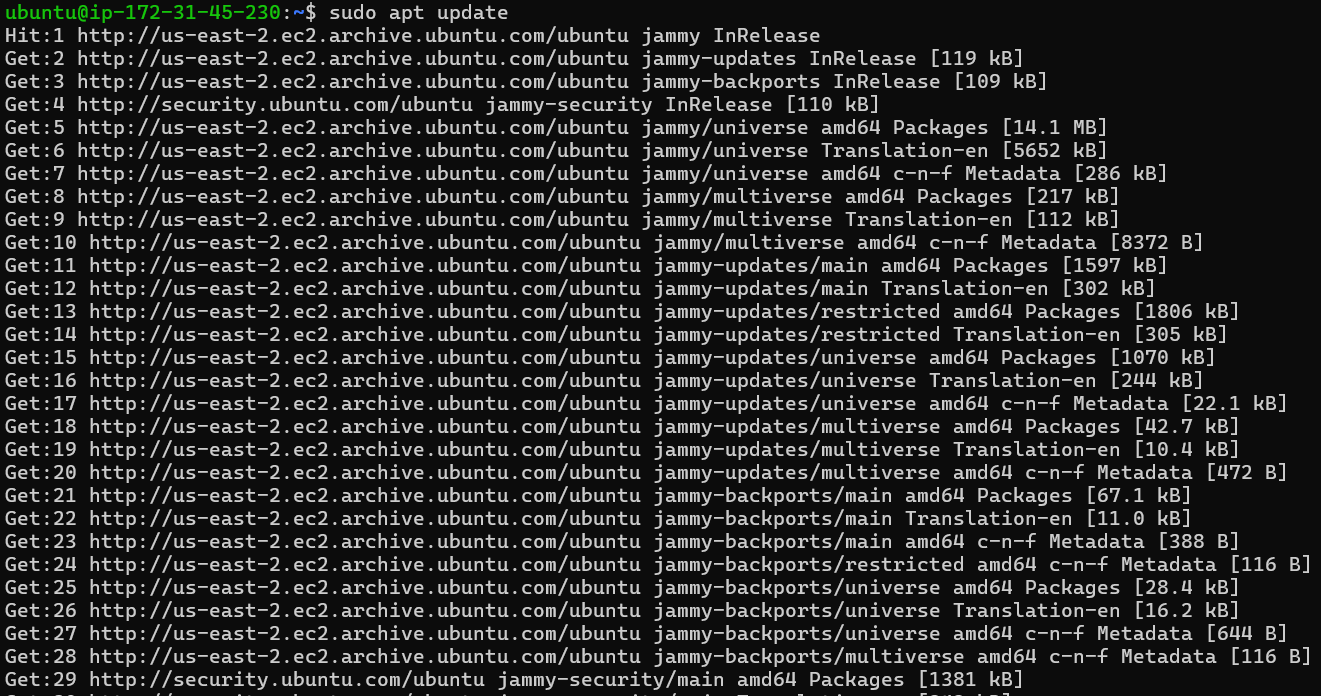
1. SSH into your EC2 instance

In EC2 created instance, click on connect. Go to SSH client and execute the command from the pem file on local machine.

ssh -i "KiranM\_All.pem" [ubuntu@ec2-3-22-234-60.us-east-2.compute.amazonaws.com](mailto:ubuntu@ec2-3-22-234-60.us-east-2.compute.amazonaws.com)

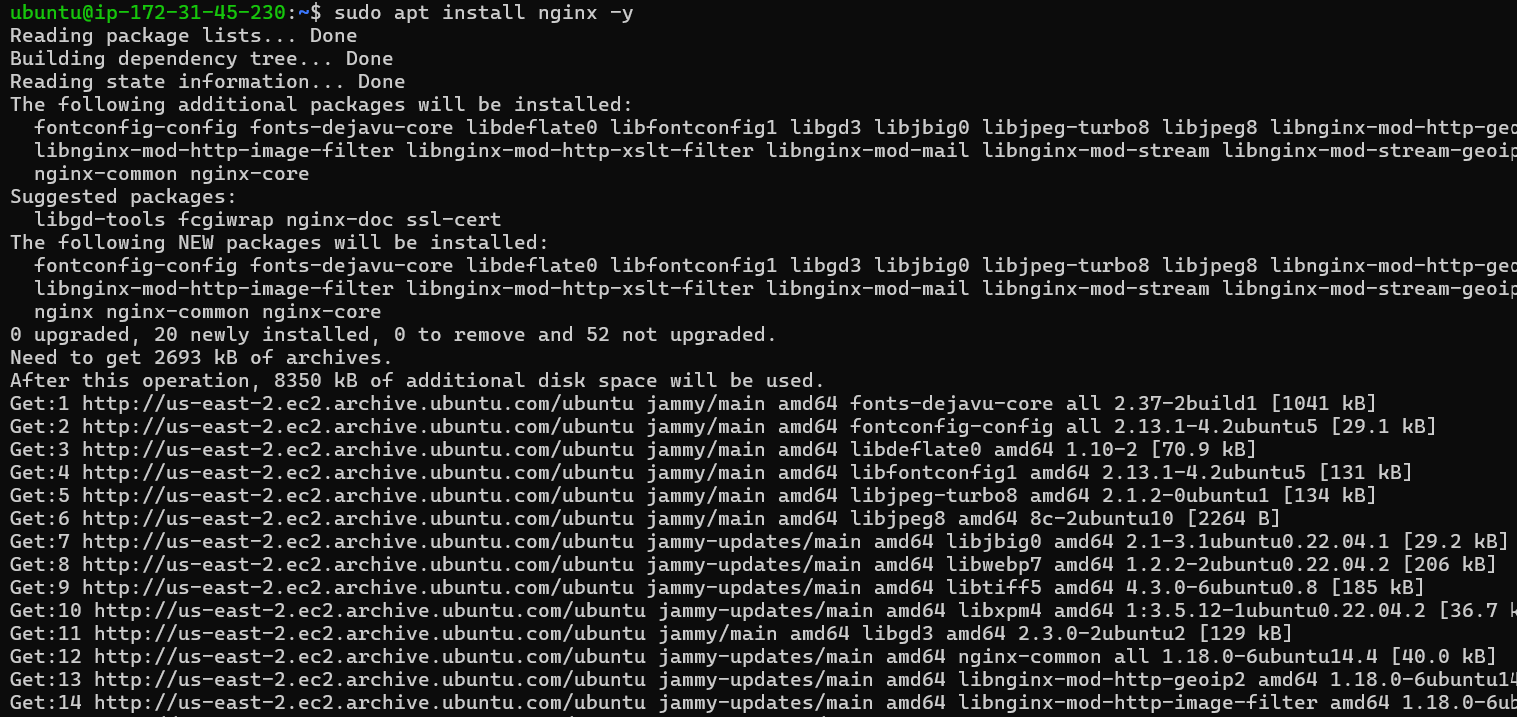


sudo apt update



Install nginx:

sudo apt install nginx -y

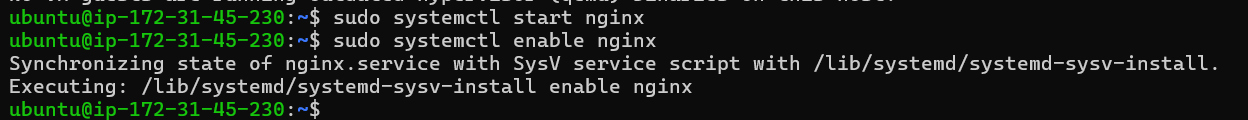


Start Nginx:

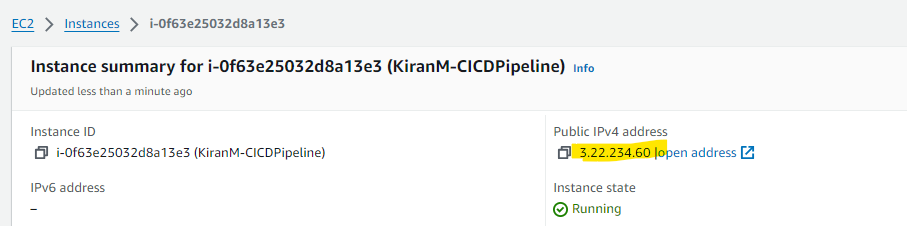
sudo systemctl start nginx

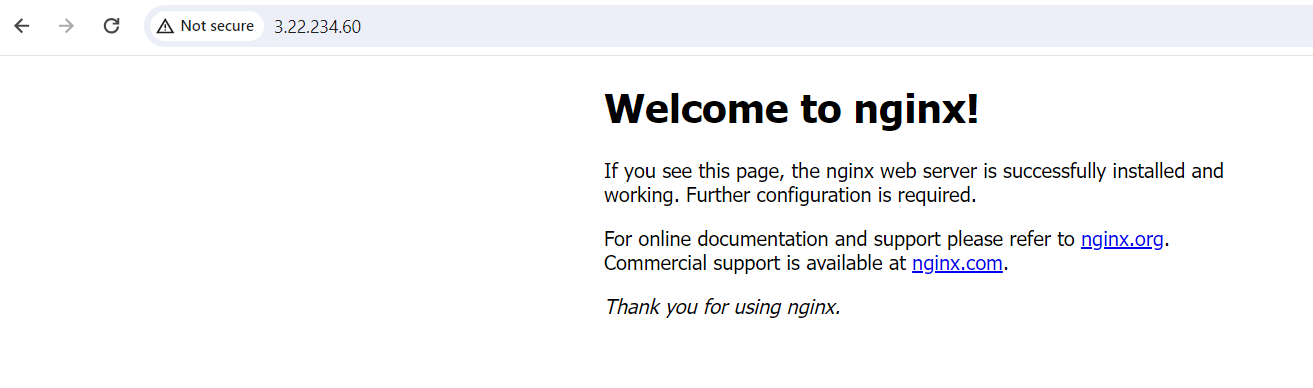
Enable nginx:

sudo systemctl enable nginx



Open the public IP in web browser:





**Task 3: Write a Python Script to Check for New Commits**

Create a Python script to check for new commits using the GitHub API.

**Code:**

import getpass

import subprocess

import requests

def get\_github\_token():

# Get the personal access token from the user

github\_token = getpass.getpass(prompt='Enter your GitHub personal access token: ')

return github\_token

def get\_current\_commit\_sha():

try:

# Run the git log command

result = subprocess.run(['git', 'log', '-1', '--pretty=format:%H'], stdout=subprocess.PIPE)

# Decode the output from bytes to string

current\_commit\_sha = result.stdout.decode().strip()

return current\_commit\_sha

except Exception as e:

print(f"Error getting current commit SHA: {e}")

return None

def get\_latest\_commit(username, repository, token):

url = f'https://api.github.com/repos/{username}/{repository}/commits'

headers = {'Authorization': token}

response = requests.get(url, headers=headers)

if response.status\_code == 200:

commits = response.json()

if commits:

latest\_commit\_sha = commits[0]['sha']

return latest\_commit\_sha

else:

return None

else:

print(f"Error: Unable to fetch commits. Status Code: {response.status\_code}")

return None

def check\_for\_new\_commits(username, repository, current\_commit\_sha, token):

latest\_commit\_sha = get\_latest\_commit(username, repository, token)

if latest\_commit\_sha and latest\_commit\_sha != current\_commit\_sha:

print("New commits found!")

print(f"Latest Commit SHA: {latest\_commit\_sha}")

else:

print("No new commits.")

if \_\_name\_\_ == "\_\_main\_\_":

try:

github\_username = 'mahalekiran'

repository\_name = 'CICD-Pipeline'

# Get latest SHA commit

current\_commit\_sha = get\_current\_commit\_sha()

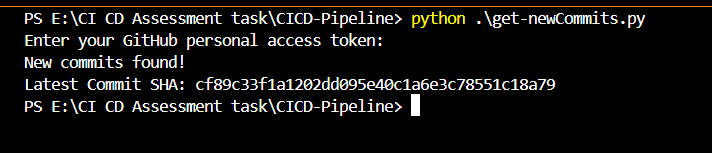
#GitHub personal access token

token = get\_github\_token()

check\_for\_new\_commits(github\_username, repository\_name, current\_commit\_sha, token)

except Exception as error:

print("Some error occurred: "+ str(error))



**Task 4: Write a Bash Script to Deploy the Code**

Create a bash script to clone the latest code and restart Nginx.

Code:

#!/bin/bash

GIT\_REPO="https://github.com/mahalekiran/CICD-Pipeline.git"

NGINX\_SERVICE="nginx"

# Cloning the latest code

if [ -d "/tmp/latest\_code" ] && [ "$(ls -A /tmp/latest\_code)" ]; then

# Pull the latest changes

echo "Pulling the latest changes from the repository..."

cd /tmp/latest\_code || exit

git pull

else

# Clone the repository

echo "Cloning the repository into /tmp/latest\_code..."

git clone "$GIT\_REPO" /tmp/latest\_code || exit

fi

# Check if the clone was successful

if [ $? -ne 0 ]; then

echo "Error: Failed to clone the repository."

exit 1

fi

# Replace old code with the latest

echo "Replacing old code with the latest..."

sudo rm -rf /var/www/html/\*

sudo cp -r /tmp/latest\_code/\* /var/www/html/

# Restart Nginx

echo "Restarting Nginx..."

sudo systemctl restart $NGINX\_SERVICE

# Check if Nginx restart was successful

if [ $? -ne 0 ]; then

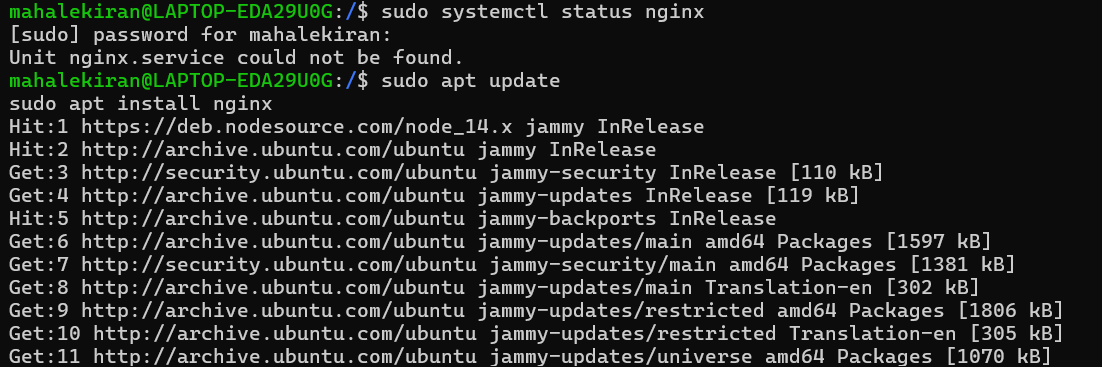
echo "Error: Failed to restart Nginx."

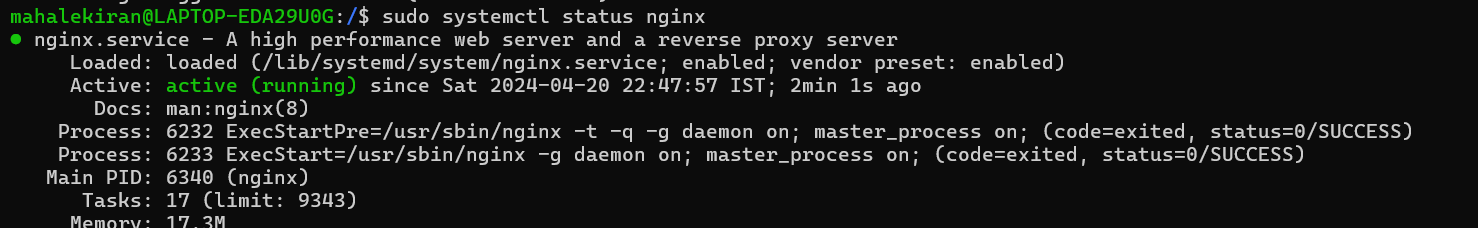
exit 1

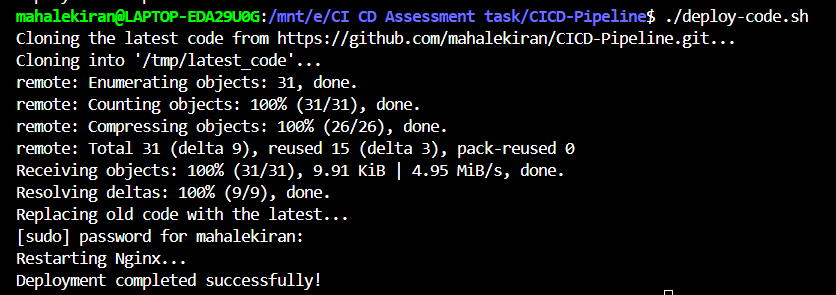
fi

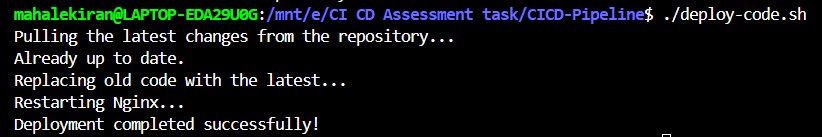
echo "Deployment completed successfully!"

If there are issues of nginx coming while executing it, then install the nginx:









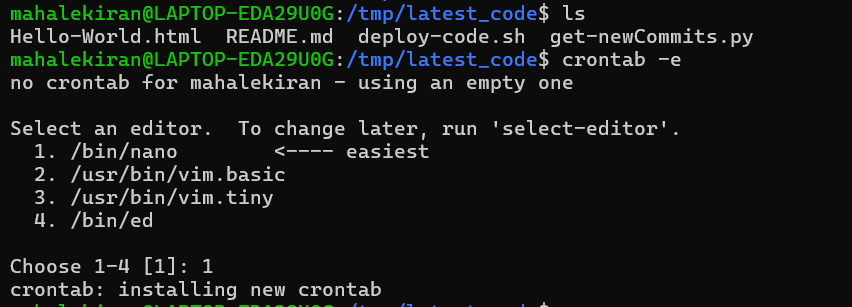
**Task 5: Set Up a Cron Job to Run the Python Script**

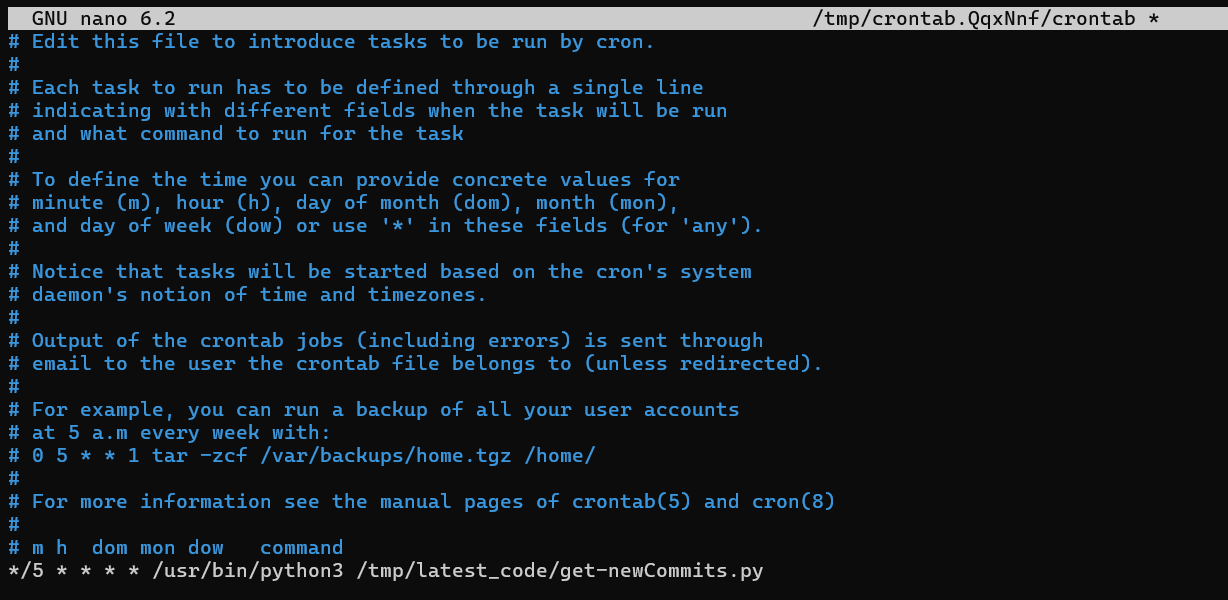
Create a cron job to run the Python script at regular intervals.

Execute command in terminal:

crontab -e

Select editor from 1 to 4





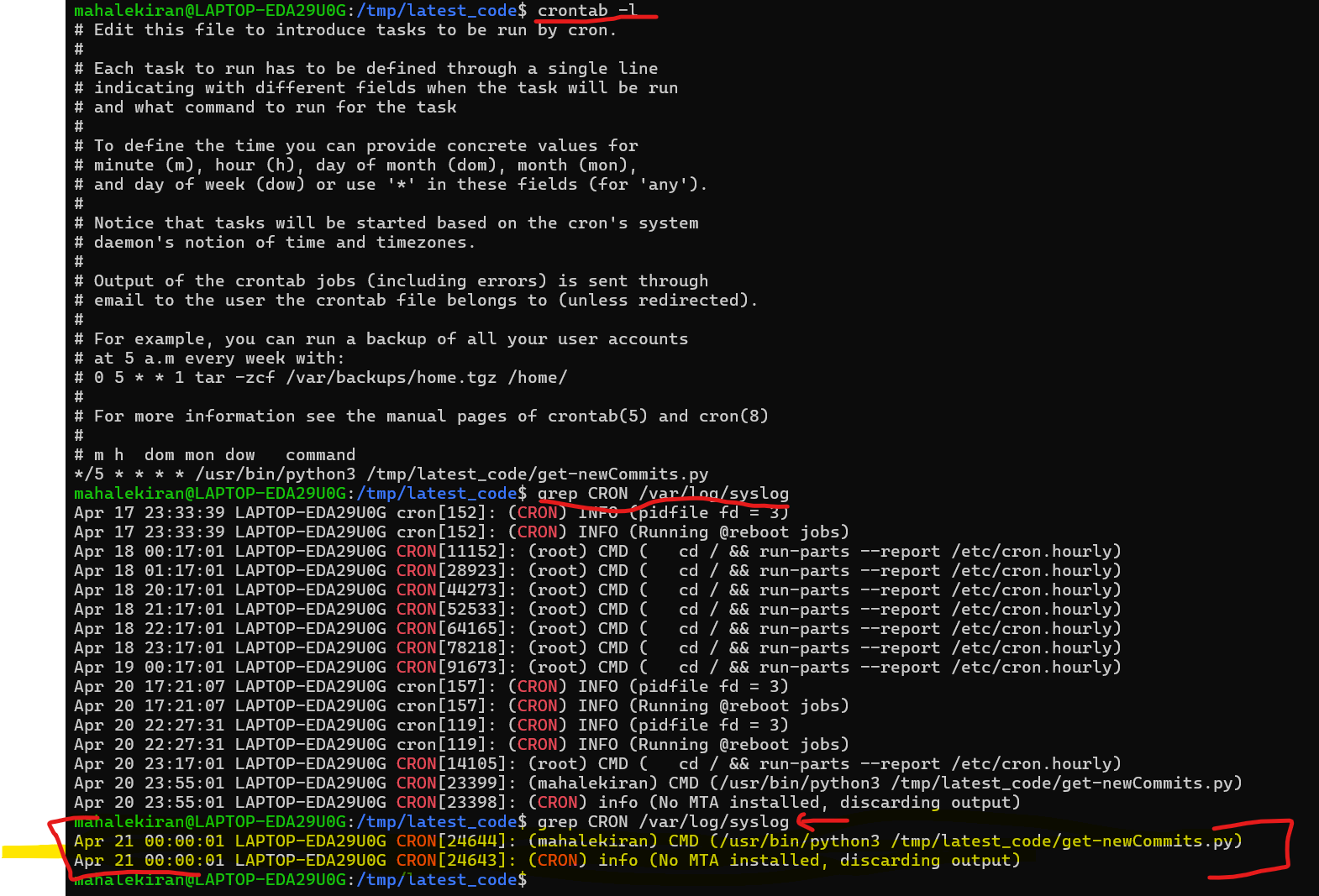
To execute python code every 5 minutes, add the following code:

\*/5 \* \* \* \* /usr/bin/python3 /tmp/latest\_code/get-newCommits.py

Save and Exit from Editor

To check if the job is ecxecuting, we will run 2 commands:

1. crontab -l
2. grep CRON /var/log/syslog



**Task 6: Test the Setup**

Make a new commit to the GitHub repository and check that the changes are automatically deployed.

To automate the deployment process of deploy-code.sh, we have to create Github action i.e. Workflow.

Please create a yml file i.e. deploy.yml in .github/workflows folder

name: Deploy

on:

push:

branches:

- main

jobs:

deploy:

runs-on: ubuntu-latest

steps:

- name: Checkout code

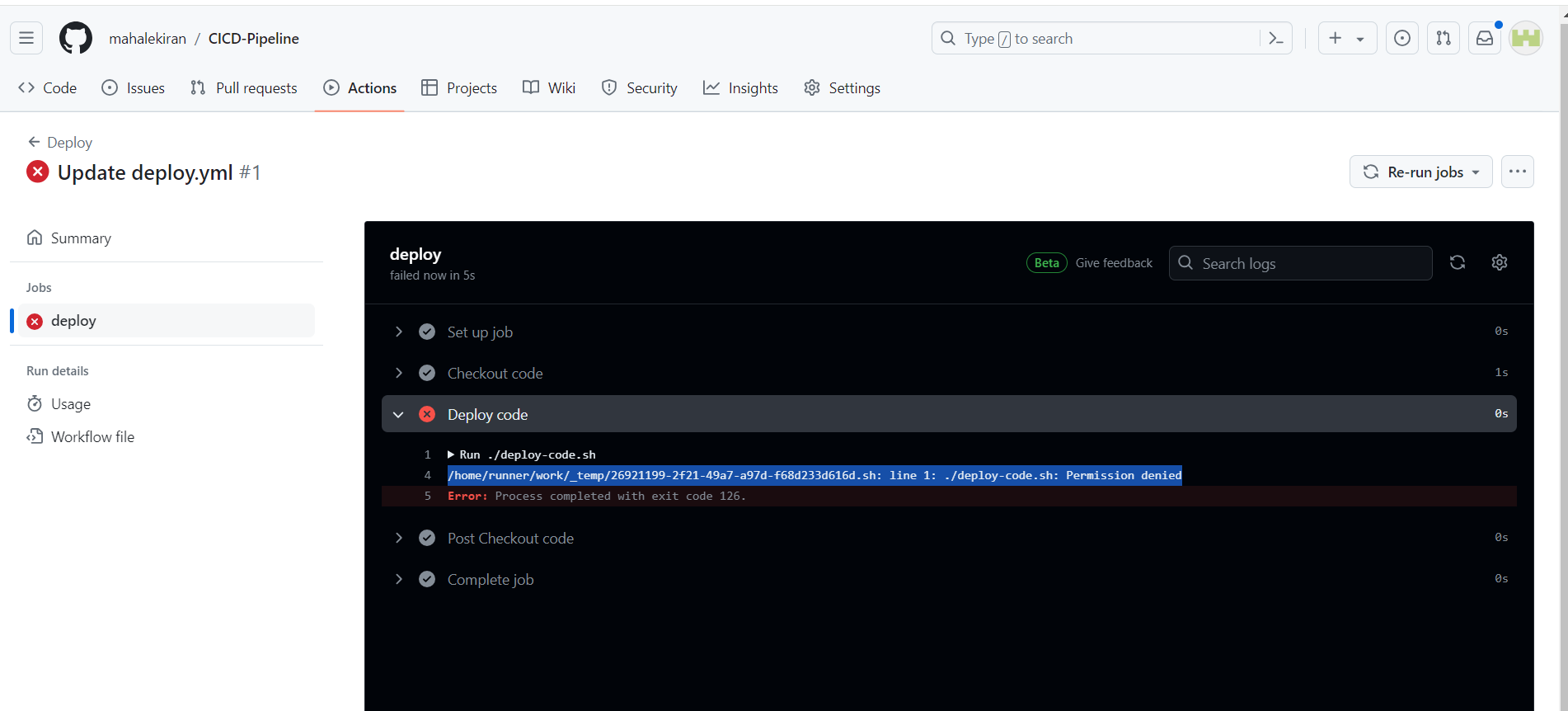
uses: actions/checkout@v2

- name: Deploy code

run: |

./deploy.sh

Go to Actions and you will see pipeline executing.



As you can see, we got an error of Permission denied. So we need to add permission in pipeline.

We will modify the yml file as below:

name: Deploy

on:

push:

branches:

- main

jobs:

deploy:

runs-on: ubuntu-latest

steps:

- name: Checkout code

uses: actions/checkout@v2

- name: Set Permission

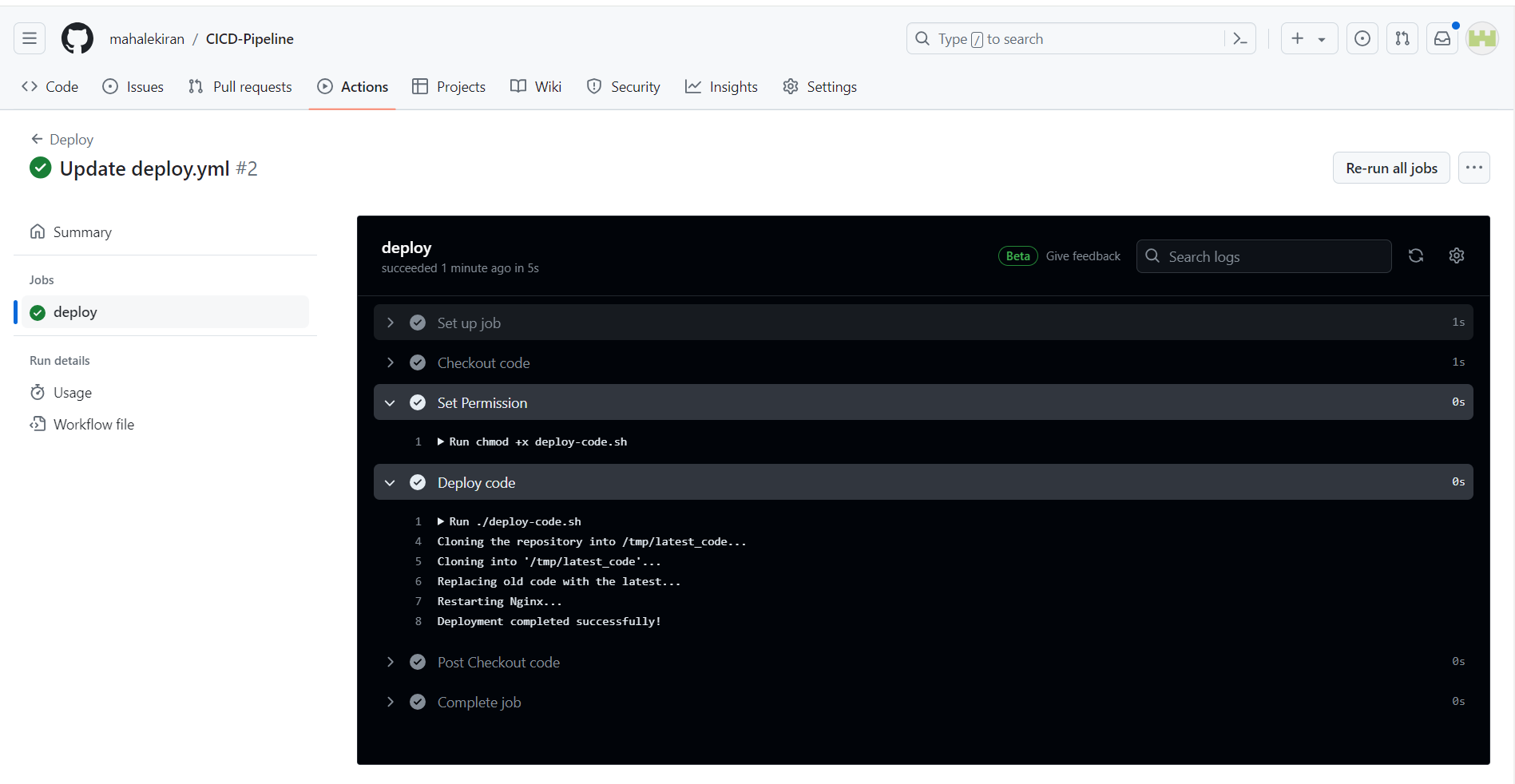
run: chmod +x deploy-code.sh

- name: Deploy code

run: |

./deploy-code.sh

Automatically pipeline executes:



Now from next time if we commit anything in main branch, the pipeline will be triggered automatically.

Here, I modified in Readme file and pushed the code.

Automatically the pipeline executed.

