|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MCAL23 DevOps Lab INDEX | | | | |
| **Sr.No** | **Title** | **CO** | **Date** | **Sign** |
| 1 | Demonstrate basic Git commands | CO2 |  |  |
| 2 | Create and fork repositories in Git Hub. Apply branch, merge and rebase concepts. | CO2 |  |  |
| 3 | Demonstrate Git for Collabration | CO2 |  |  |
| 4 | Demonstrate Collabrating and cloning using Git | CO2 |  |  |
| 5 | Using GitLab Web IDE | CO2 |  |  |
| 6 | Demonstrate CI/CD Workflow in GitLab using .py, .bash, .java file | CO2 |  |  |
| 7 | Demonstrate setting Jenkins CI/CD pipeline. | CO3 |  |  |
| 8 | Demonstrate Setting up of a CI/CD pipeline to build and deploy a web application to a  local HTTP server | CO3 |  |  |
| 9 | Demonstrate basic Docker commands | CO3 |  |  |
| 10 | Develop a simple containerized application using Docker | CO3 |  |  |
| 11 | Demonstrate add-on ansible commands | CO4 |  |  |
| 12 | Demonstrate Ansible Playbooks | CO4 |  |  |

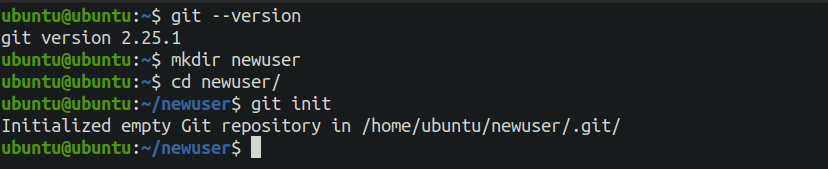
# PRACTICAL- 1

## Aim: Basic Git commands

#### Check git version git –version

\_

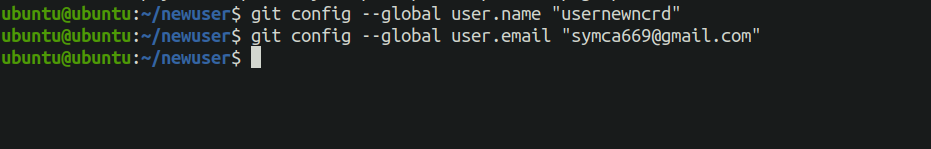
#### Create folder and initiliaze.



1. Configure Git

git config --global user.name "usernewncrd"

git config --global user.email “[symca669@gmail.com](mailto:symca669@gmail.com)”

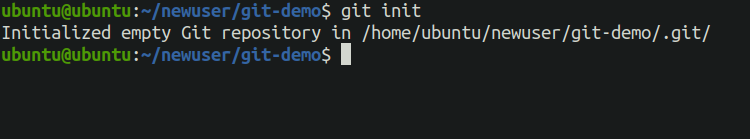


1. Create a new project folder mkdir git-demo

cd git-demo

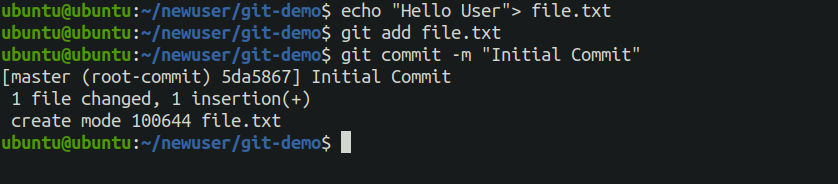


1. git init



1. Create and track a file: echo "Hello User" > file.txt git add file.txt

git commit -m "Initial commit"



1. Check status and log: git status

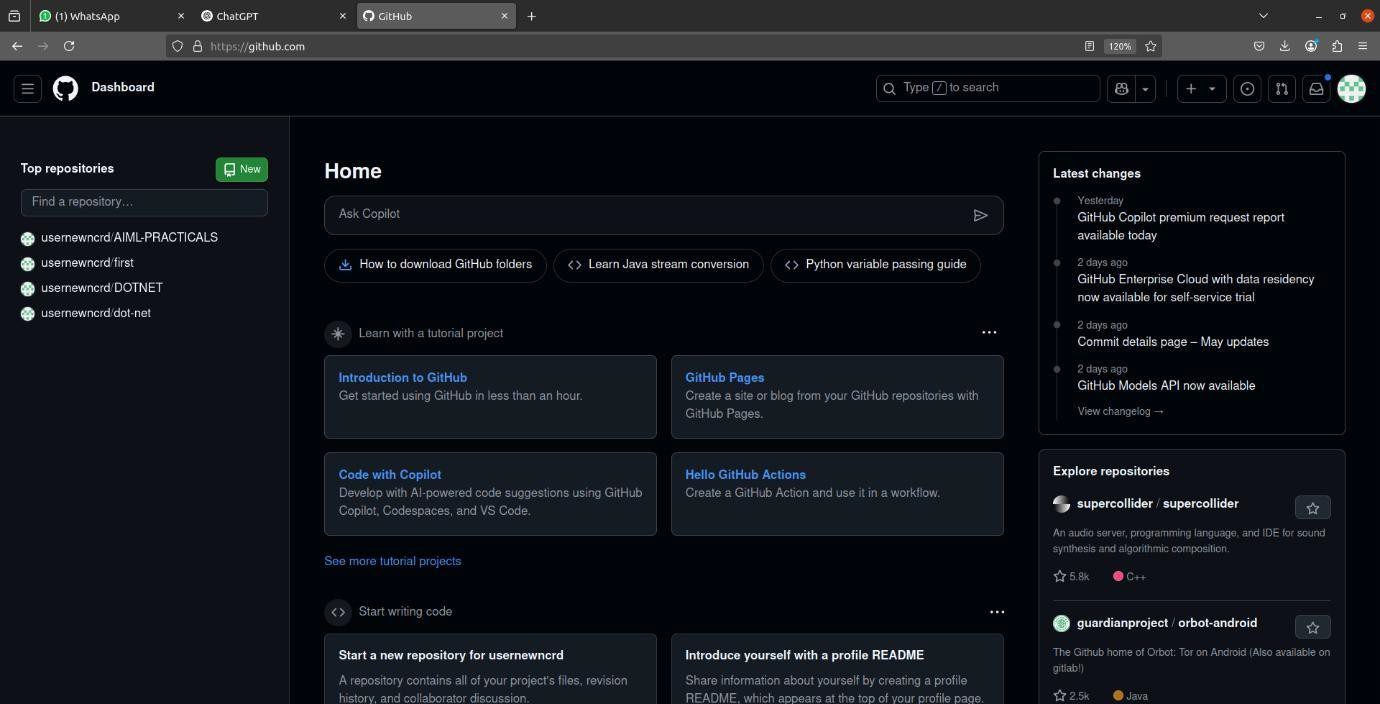
git log



# PRACTICAL- 2

## Aim: Create and fork repositories in GitHub. Apply branch, merge, rebase concepts.

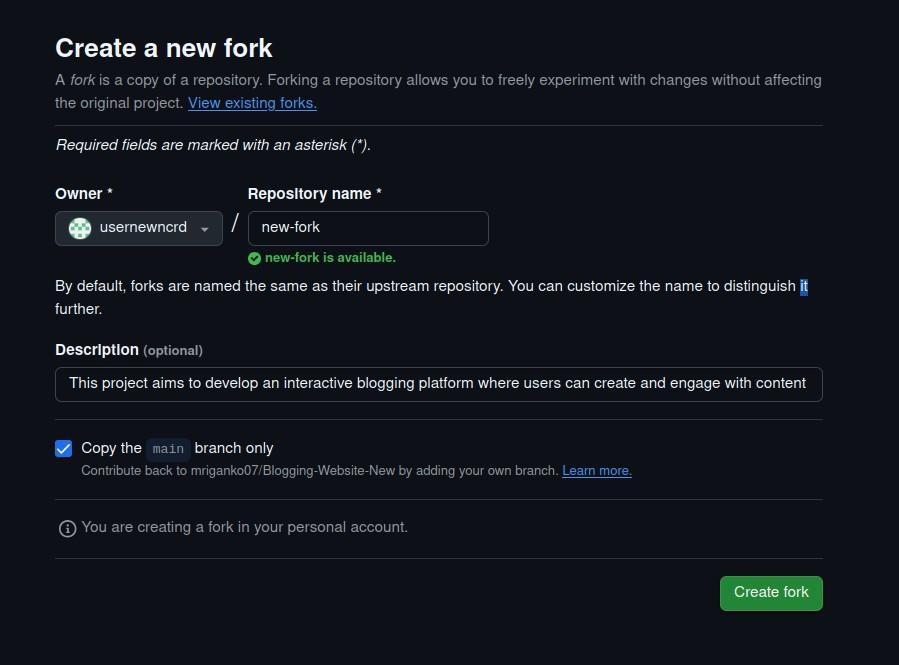
#### Create a GitHub account and log in.



1. Create a repository on GitHub (e.g., git-practice).

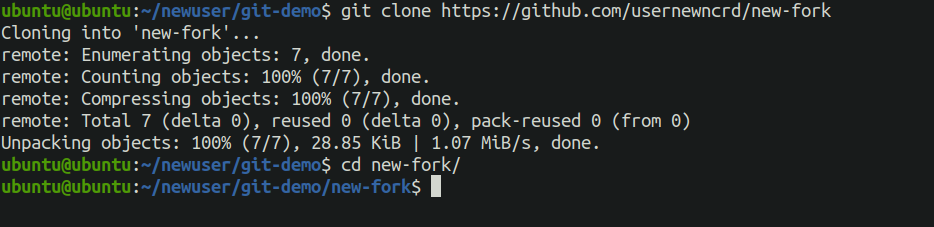


1. Fork any public repository or your own from another account



1. Clone the forked repo:

git clone https://github.com/usernewncrd/git-practice.git cd git-practice



1. Create a branch:

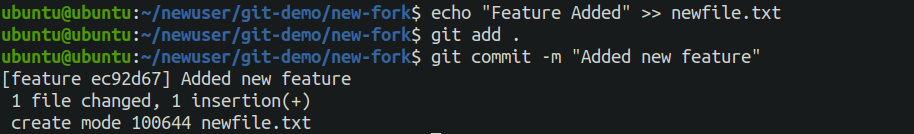
git checkout -b feature



1. Make changes, then commit:

echo "Feature added" >> newfile.txt git add .

git commit -m "Added new feature"



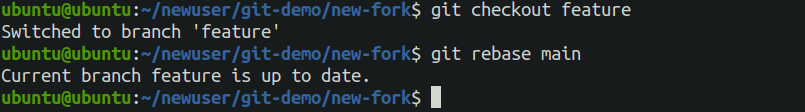
1. Merge branch into main: git checkout master

git merge feature



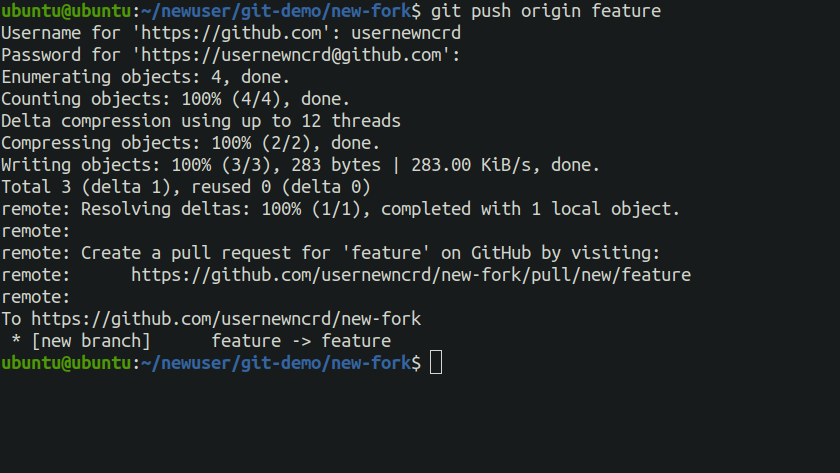
1. Rebase branch (alternative to merge): git checkout feature

git rebase master



1. Push to GitHub:

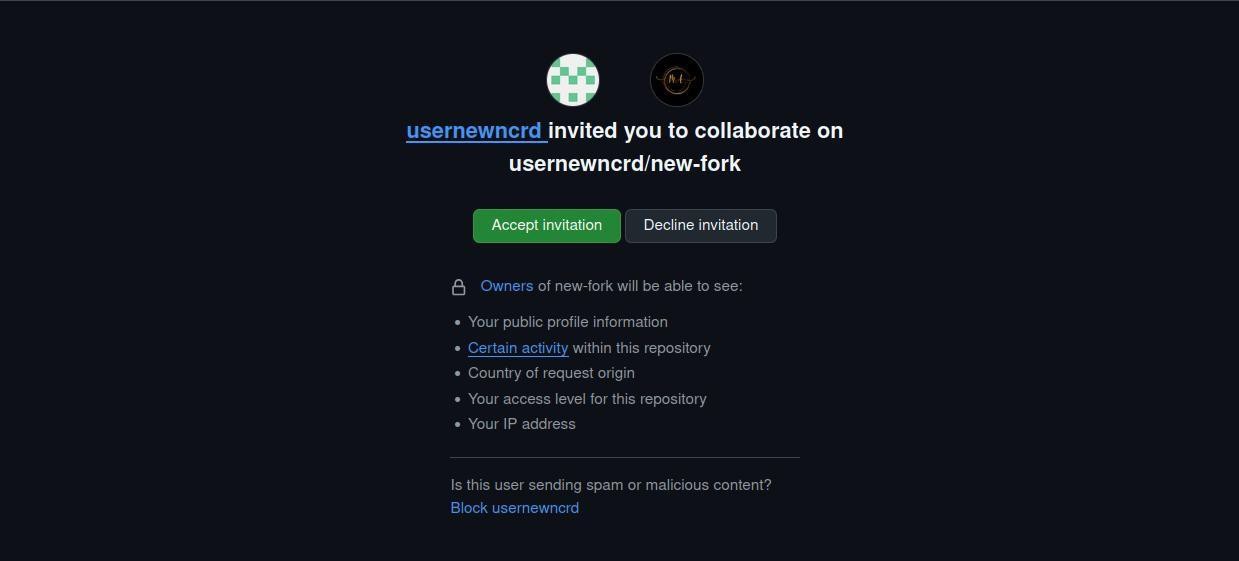
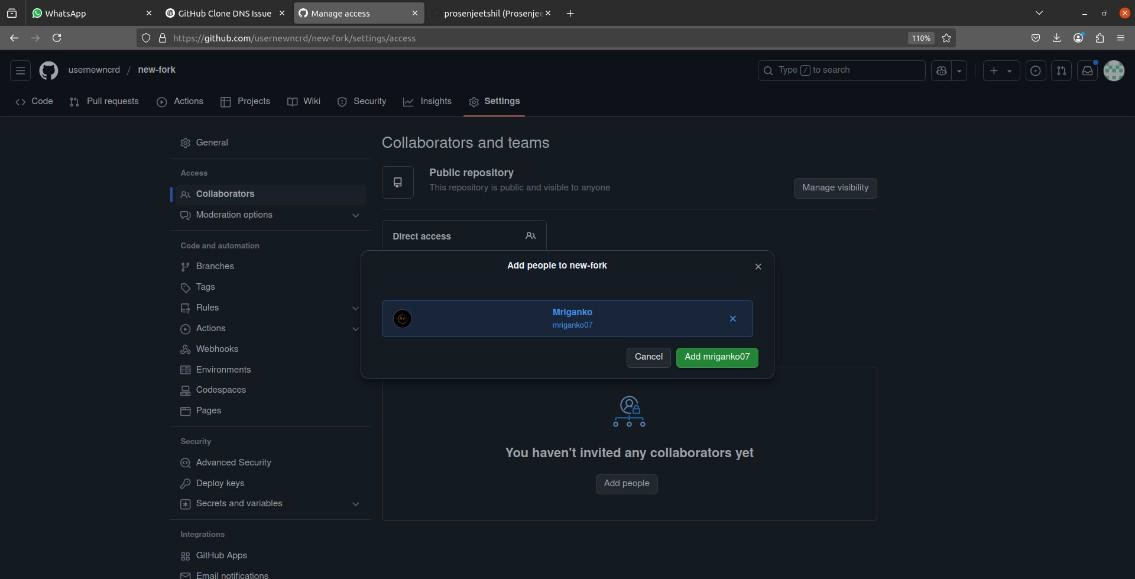
git push origin feature



# PRACTICAL-3

## Aim: Using Git for Collaboration

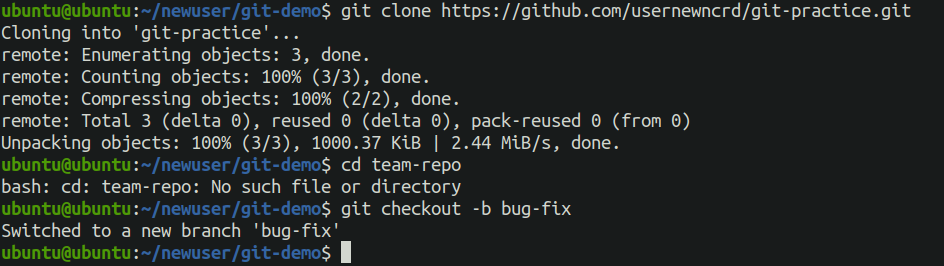
1. Using Git for Collaboration



1. Friend clones the repo:

git clone <https://github.com/usernewncrd/git-practice.git> cd team-repo

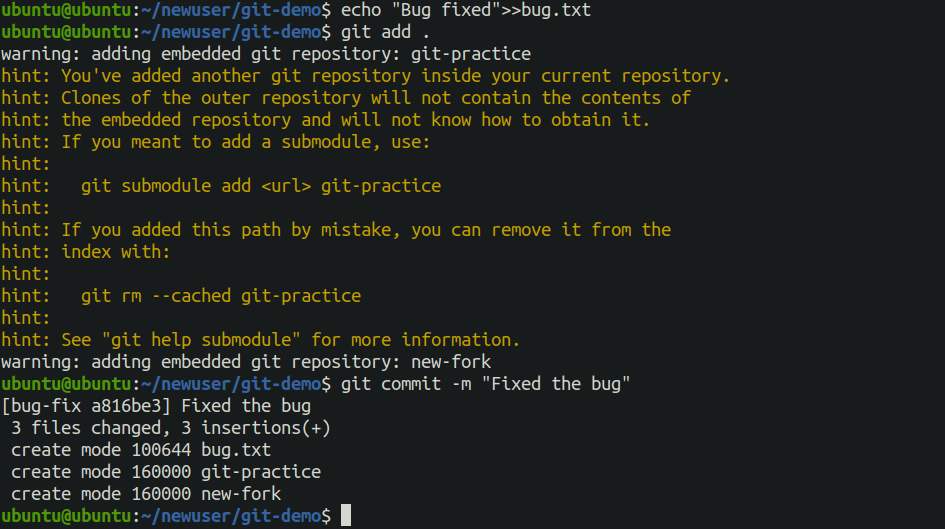
git checkout -b bug-fix



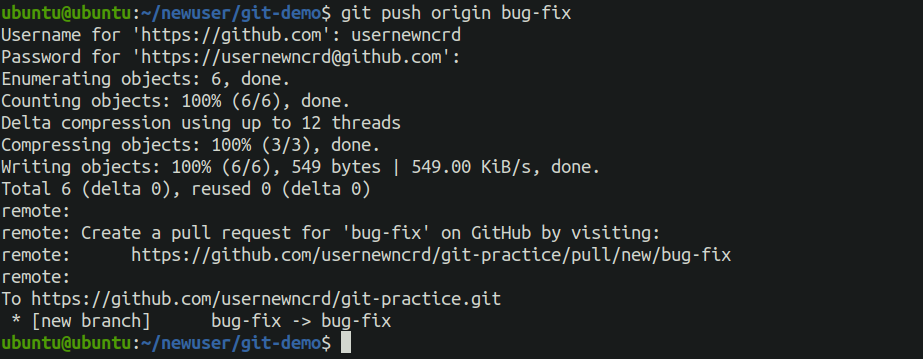
1. Friend makes changes and pushes: echo "Bug fixed" >> bug.txt

git add .

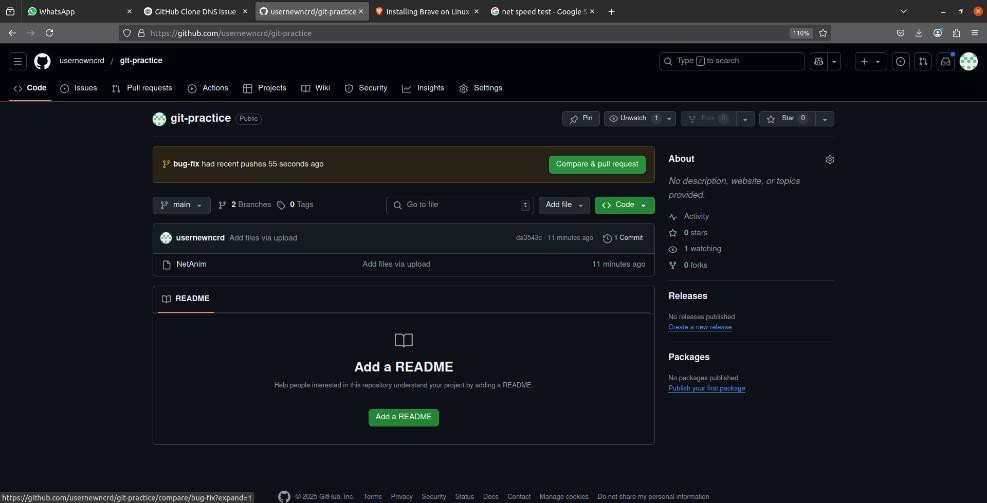
git commit -m "Fixed a bug"



1. git push origin bug-fix



1. Pull Request

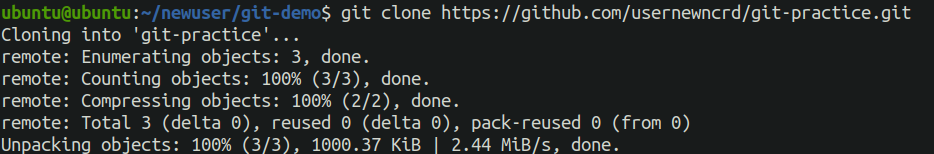


# PRACTICAL-4

## Aim: Collaborating and Cloning using GitHub

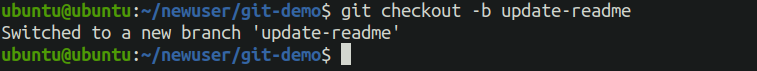
1. Clone a public repository:

git clone https://github.com/usernewncrd /git-practice.git



1. Create a branch:

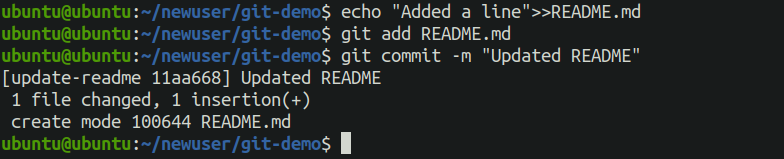
git checkout -b update-readme



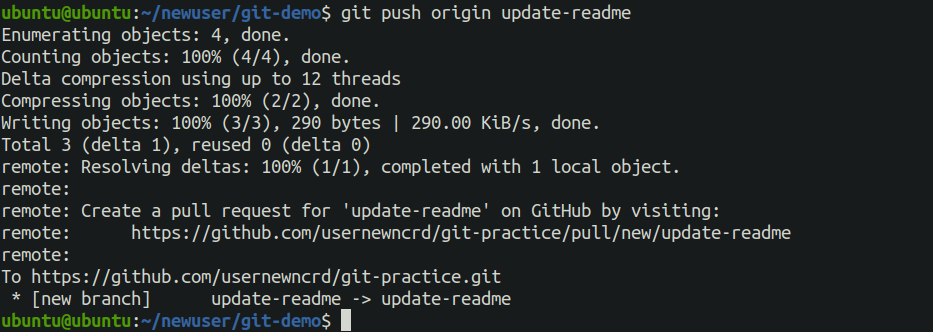
1. Edit and commit changes:

echo "Added a line" >> README.md git add README.md

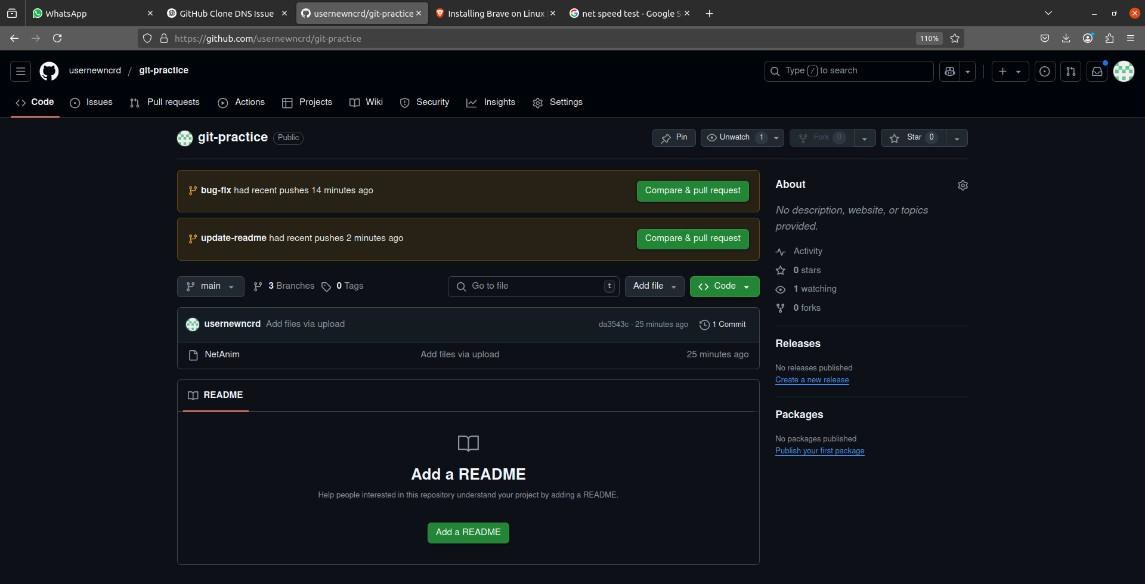
git commit -m "Updated README"



1. Push and open pull request:



1. git push origin update-readme



# PRACTICAL-5

## Aim: Using GitLab Web IDE

Using GitLab Web IDE is a convenient way to edit, commit, and manage your code directly in your

browser without needing a local IDE. Here’s a quick overview of how to use it effectively:

Opening GitLab Web IDE

1. Go to your project in GitLab.
2. Click the Web IDE button (usually near the top right of the repository page).
   * You can also access it via https://gitlab.com/<namespace>/<project>/-

/ide/project/<branch-name>/edit.

Basic Features

|  |  |
| --- | --- |
| Feature | Description |
| File Explorer | View and manage your project files on the left panel. |
| Editor Window | Edit files with syntax highlighting and autocompletion. |
| Terminal (optional) | For Git commands, builds, or other scripts (if enabled). |
| Commit Panel | Stage, commit, and push your changes. |
| Branches | Switch between branches and create new ones. |

Workflow Example

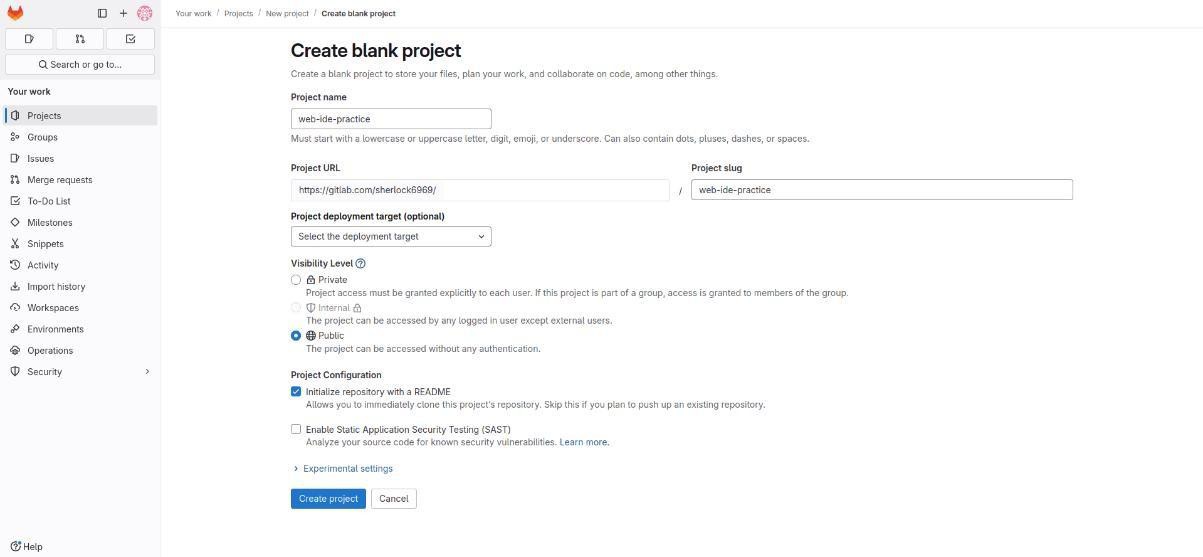
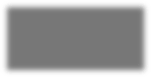
1. Edit files: Click on a file in the left panel and make your changes.
2. Stage changes: Go to the "Changes" tab → check the files to stage.
3. Write a commit message: In the "Commit" section.
4. Commit & Push:
   * Choose "Commit to current branch" if you're working on a feature branch.
   * Or "Create merge request" if you're done and ready to merge.

Tips

* Auto Save: It autosaves your file edits, but you still need to commit changes.
* Live Preview (for web apps): Available for GitLab Pages-based previews if set up.
* Extensions: Web IDE supports VS Code-like extensions for enhanced functionality.

**Steps:**

1. Sign up at [https://gitlab.com](https://gitlab.com/)
2. Create a project.
3. Click on Web IDE in your repository.



1. Create a file (index.html):

<html>

<body>

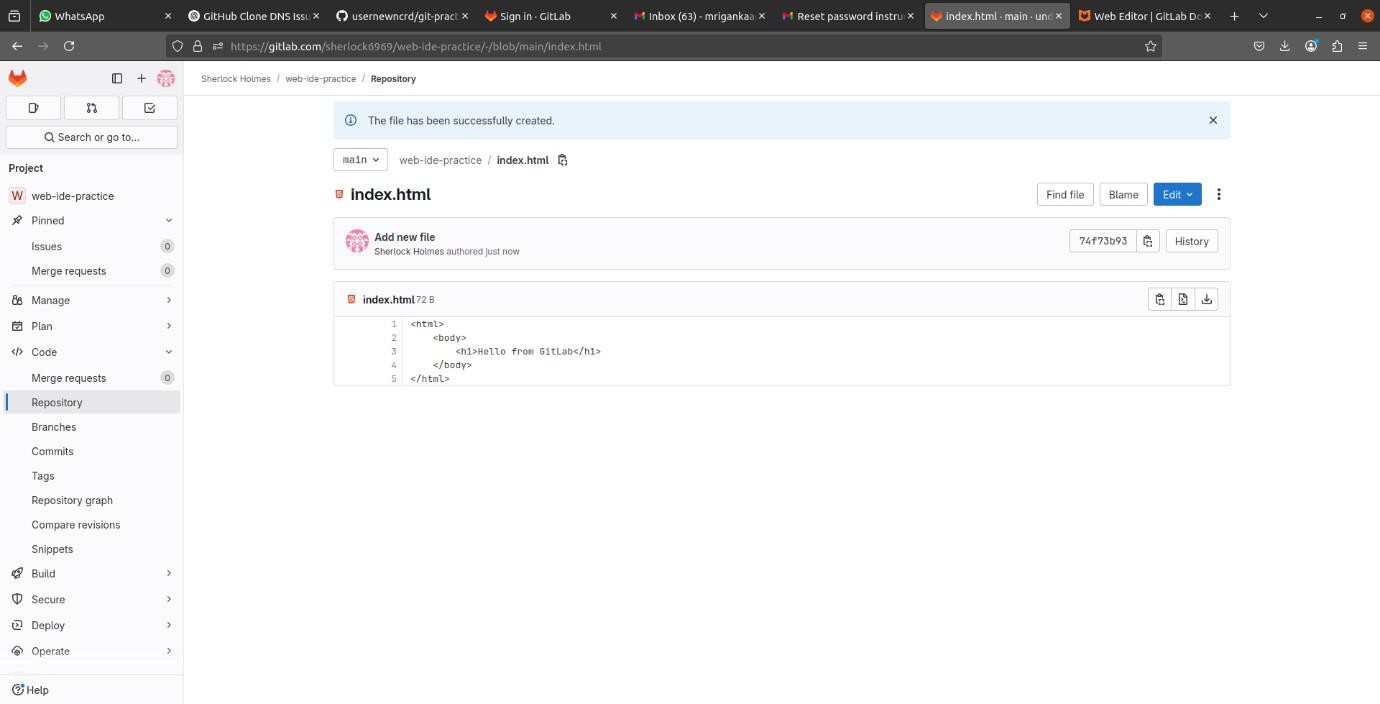
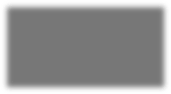
<h1>Hello from GitLab</h1>

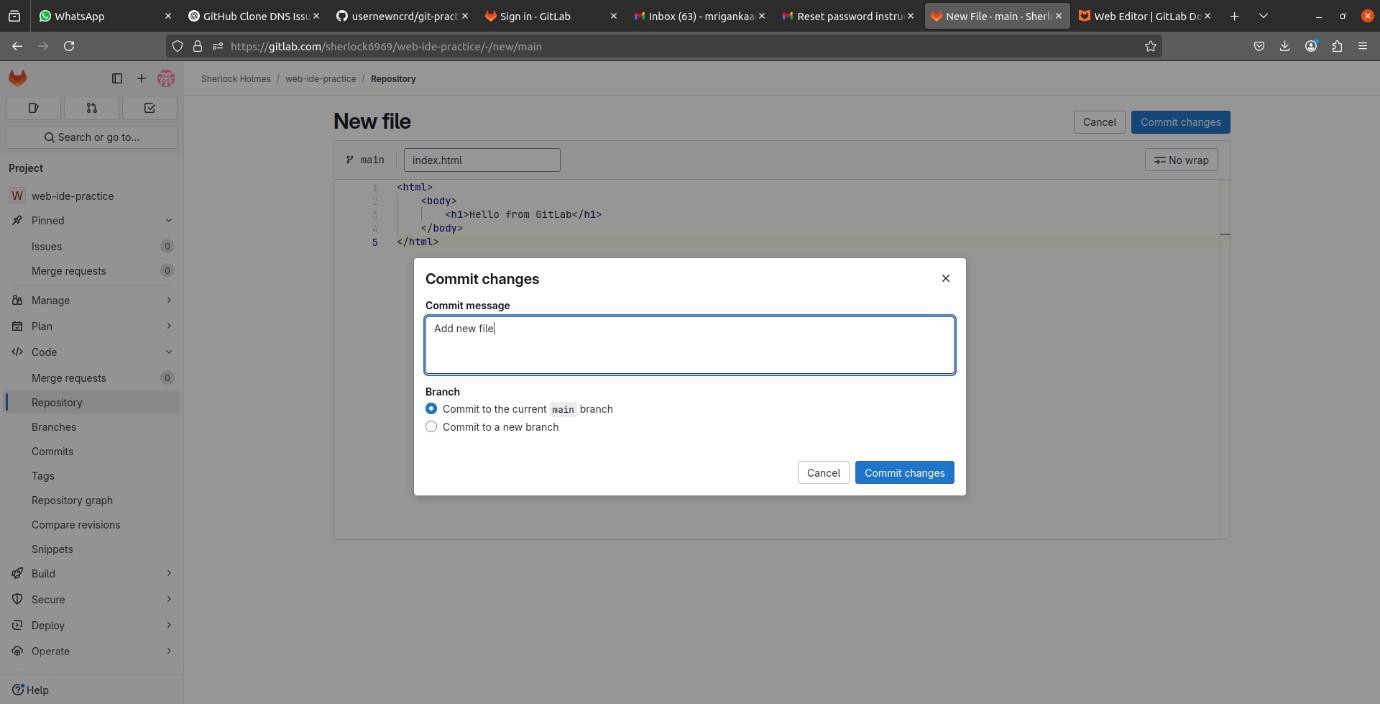
</body>

</html>



1. Click Commit and push changes.





**Practical No.:6**

**Demonstrate CI/CD Workflow in GitLab using .py, .bash, .java file**

**Bash**

**A screenshot of a computer

AI-generated content may be incorrect.**

echo "This is from my bash script"

touch myFile.txt

echo "Helloo 69" > myFile.txt

echo "Testt"

**A screenshot of a chat

AI-generated content may be incorrect.**

stages:

- build

bash\_execute:

stage: build

script:

- bash ./basic.sh

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**PYTHON**

**A screenshot of a computer

AI-generated content may be incorrect.**

**script.py**

print("HEllO FROM NAGOBA")

**A screenshot of a computer

AI-generated content may be incorrect.**

**.gitlab-ci.yml**

stages:

- test

python\_script:

stage: test

image: python:3.10

script:

- python script.py

**A screenshot of a chat

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**JAVA**

**A screenshot of a computer

AI-generated content may be incorrect.**

**JAvaaa.java**

class JAvaaa{

public static void main(String a[]){

System.out.println("Hello World!");

System.out.println("Agent 47 arrived in Lahore!");

}

}

**A screenshot of a computer

AI-generated content may be incorrect.**

**.gitlab-ci.yml**

stages:

- build

- test

before\_script:

- apt-get update && apt-get install -y openjdk-17-jdk

build:

stage: build

script:

- javac JAvaaa.java

- ls -ls

artifacts:

paths:

- JAvaaa.class

only:

- main

test:

stage: test

when: manual

script:

- ls -l

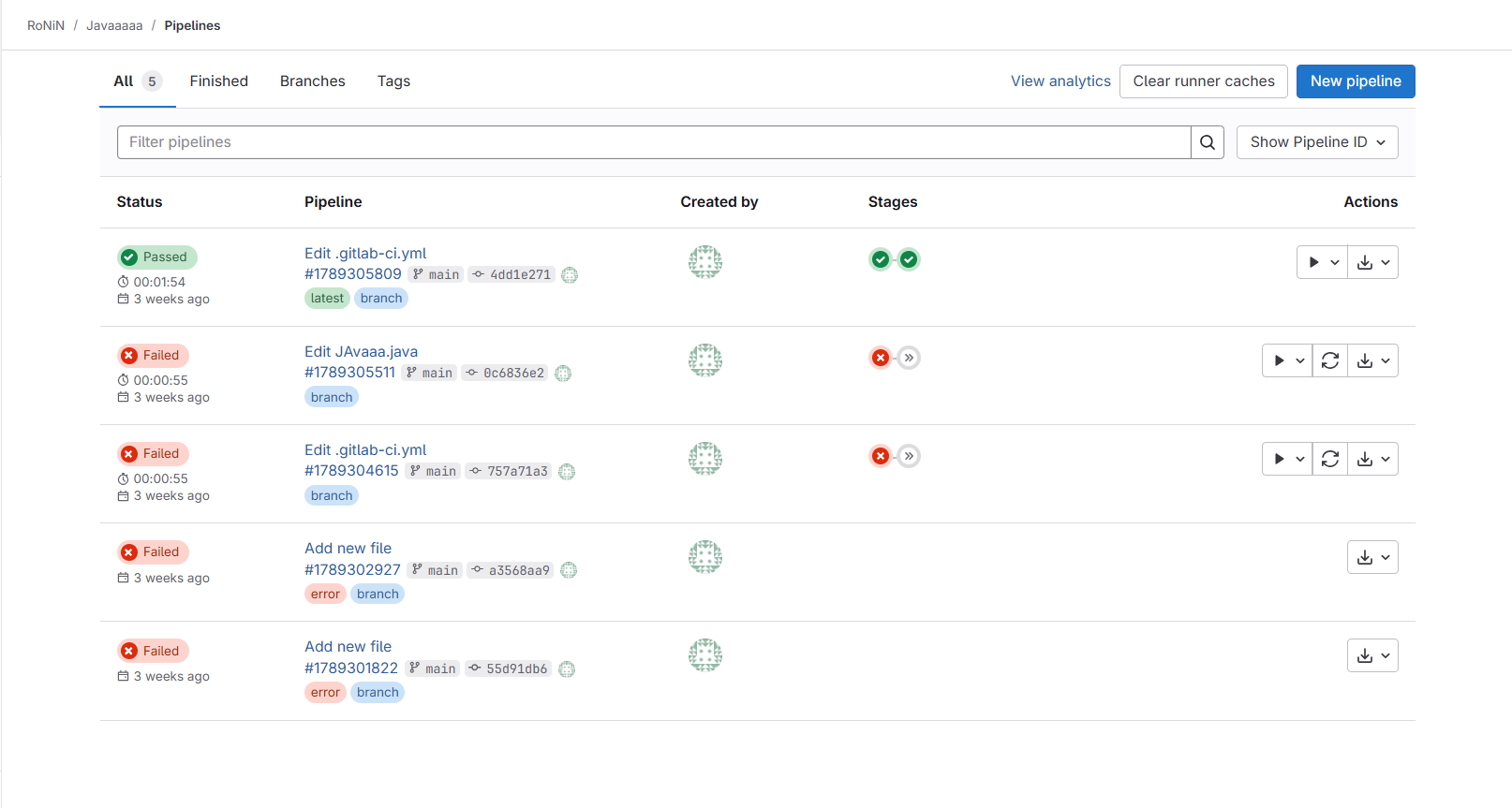
- java JAvaaa

only:

- main

**A screenshot of a computer

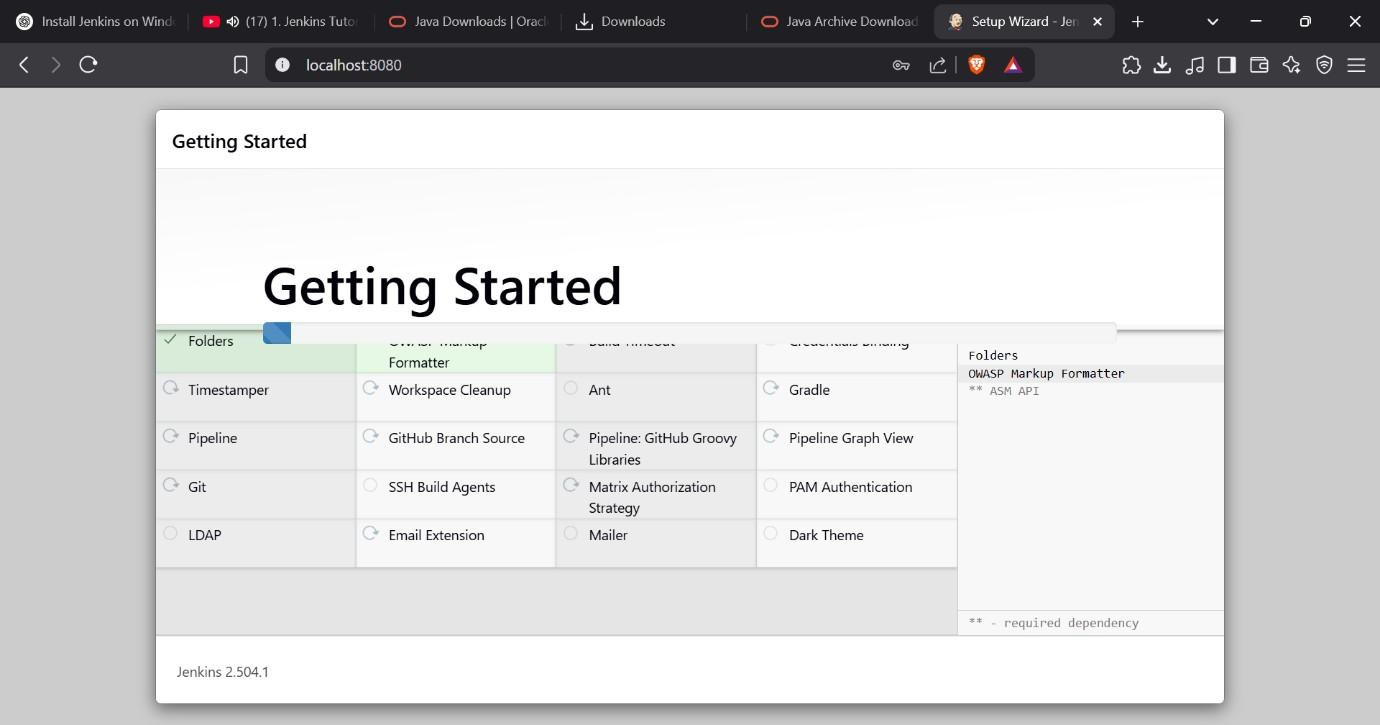
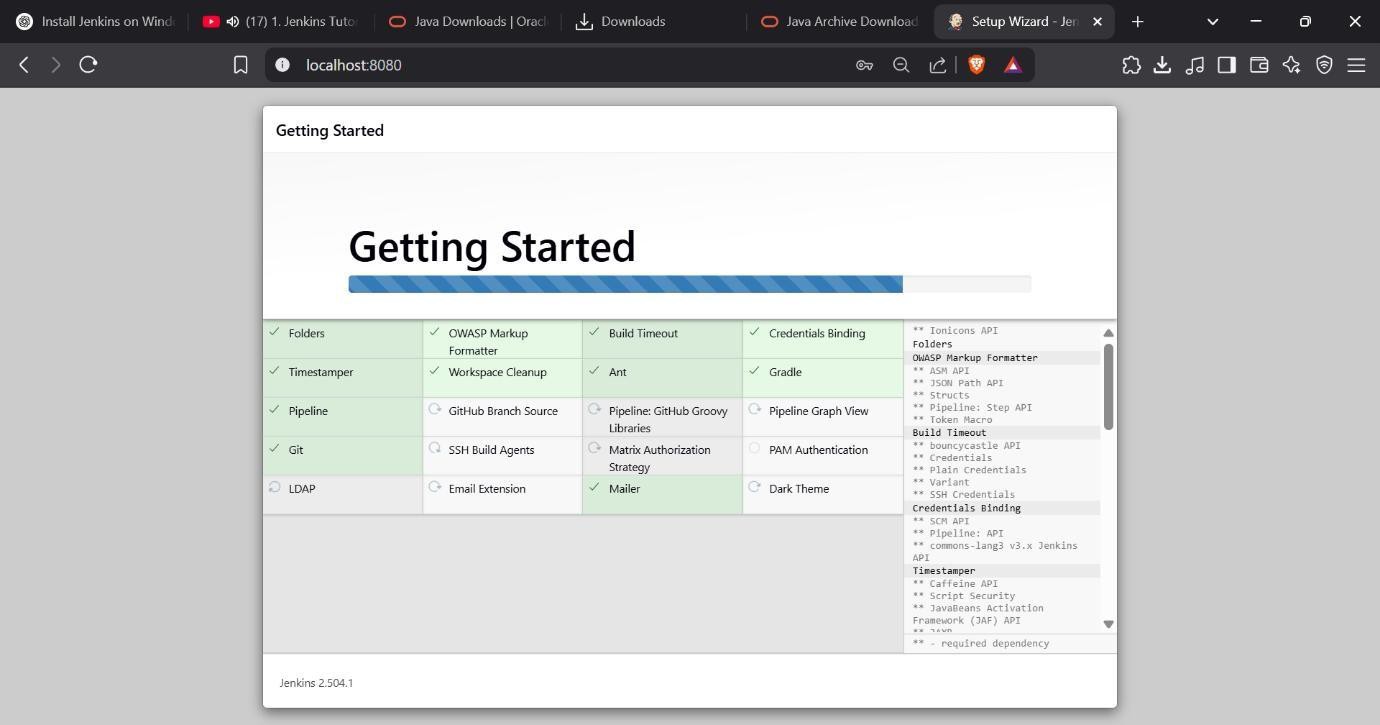
AI-generated content may be incorrect.**

****

**Practical No.:7**

**Demonstrate setting Jenkins CI/CD pipeline.**

1. **Install Jenkins (visit** [**https://www.jenkins.io**](https://www.jenkins.io/)**)**
2. **Run Jenkins: http://localhost:8080**



1. **Create a new Pipeline project: CI-CD-Demo**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

1. **Add Pipeline Script > Script:**

pipeline {

agent any // Defines where the pipeline runs

stages {

stage('Build') { // Defines a step in the pipeline

steps {

echo 'Building the project...' // Print message to console

}

}

stage('Test') {

steps {

echo 'Running tests...'

}

}

stage('Deploy') {

steps {

echo 'Deploying the application...'

}

}

}

post {

success {

echo 'Pipeline completed successfully!' // Runs if the pipeline is successful

}

failure {

echo 'Pipeline failed!' // Runs if any stage fails

}

}

}

A screenshot of a computer

AI-generated content may be incorrect.

1. **Save and click Build Now.**

**A white screen with black border

AI-generated content may be incorrect.**

1. **Check output in Console Output.**

A screenshot of a computer

AI-generated content may be incorrect.

**Practical No.:8**

**Demonstrate Setting up of a CI/CD pipeline to build and deploy a web application to a local HTTP server**

**Create a new Dynamic web project (Eclipse IDE for enterprise java and web developers)**

**Index.jsp :**

<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<html>

<head>

<title>CookiesDemo</title>

</head>

<body>

<h2>CookiesDemo - </h2>

<form action="CookiesDemo.jsp" method="get">

Name - <input type="text" name="usernm">

<input type="submit" value="Submit Query">

</form>

</body>

</html>

**CookiesDemo.jsp:**

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<%@ page import="jakarta.servlet.http.Cookie" %>

s<%@ page import="java.io.\*" %>

<html>

<head>

<title>Session Management using Cookies</title>

</head>

<body>

<h2>Session Management using Cookies</h2>

<%

String username = request.getParameter("usernm");

Cookie[] cookies = request.getCookies();

int visitCount = 0;

boolean userExists = false;

if (cookies != null) {

for (Cookie cookie : cookies) {

if (cookie.getName().equals("visitCount")) {

visitCount = Integer.parseInt(cookie.getValue());

}

if (cookie.getName().equals("username")) {

userExists = true;

}

}

}

visitCount++;

Cookie visitCookie = new Cookie("visitCount", String.valueOf(visitCount));

visitCookie.setMaxAge(60 \* 60 \* 24);

response.addCookie(visitCookie);

if (!userExists && username != null) {

Cookie userCookie = new Cookie("username", username);

userCookie.setMaxAge(60 \* 60 \* 24);

response.addCookie(userCookie);

}

%>

<p>Hello <%= username != null ? username : "Guest" %> You have hit the page <%= visitCount %> time(s)</p>

<a href="CookiesDemo.jsp?usernm=<%= username %>">Hit Again</a>

</body>

</html>

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Add Pipeline Script > Script:**

pipeline {

agent any

stages {

stage('Checkout Code') {

steps {

script {

git branch: 'master', url: 'https://github.com/YadneshTeli/DevopsJenkins'

}

}

}

stage('Verify Files') {

steps {

bat 'dir /S /B'

}

}

stage('Deploy') {

steps {

script {

def srcPath = "CookiesDemo/src/main/webapp"

def destPath = "C:\\Program Files\\Apache Software Foundation\\Tomcat 11.0\\webapps\\Index"

if (fileExists(srcPath)) {

bat "xcopy /E /I \"${srcPath}\" \"${destPath}\""

} else {

error "Source directory ${srcPath} does not exist!"

}

}

}

}

}

}

* **Open Manager app from the Tomcat panel by entering username and password –**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

* **Click on the link present in the Jenkin’s Console Output –**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

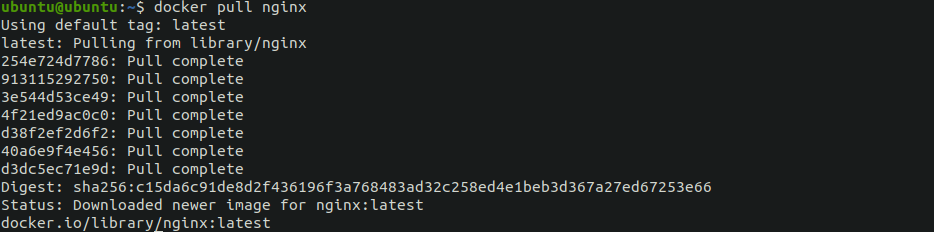
# PRACTICAL-9

## Aim: Explore docker commands for content management

1. Check Docker version docker –version



1. Pull a Docker image from Docker Hub docker pull nginx



1. List all Docker images docker images



1. Run a container from an image

docker run -d -p 8080:80 --name mynginx nginx

This will run the Nginx container and map port 80 (inside the container) to port 8080 (on your host).



1. List all running containers docker ps



1. Copy content from host to container

docker cp index.html mynginx:/usr/share/nginx/html/

Replace index.html with your actual file. This copies a file into the running container.



1. Copy content from container to host

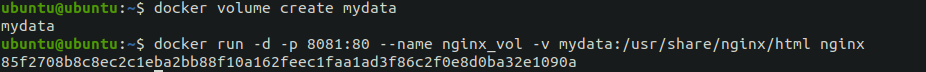
docker cp mynginx:/usr/share/nginx/html/index.html .



1. Create and use Docker volume for persistent content docker volume create mydata

docker run -d -p 8081:80 --name nginx\_vol -v mydata:/usr/share/nginx/html nginx

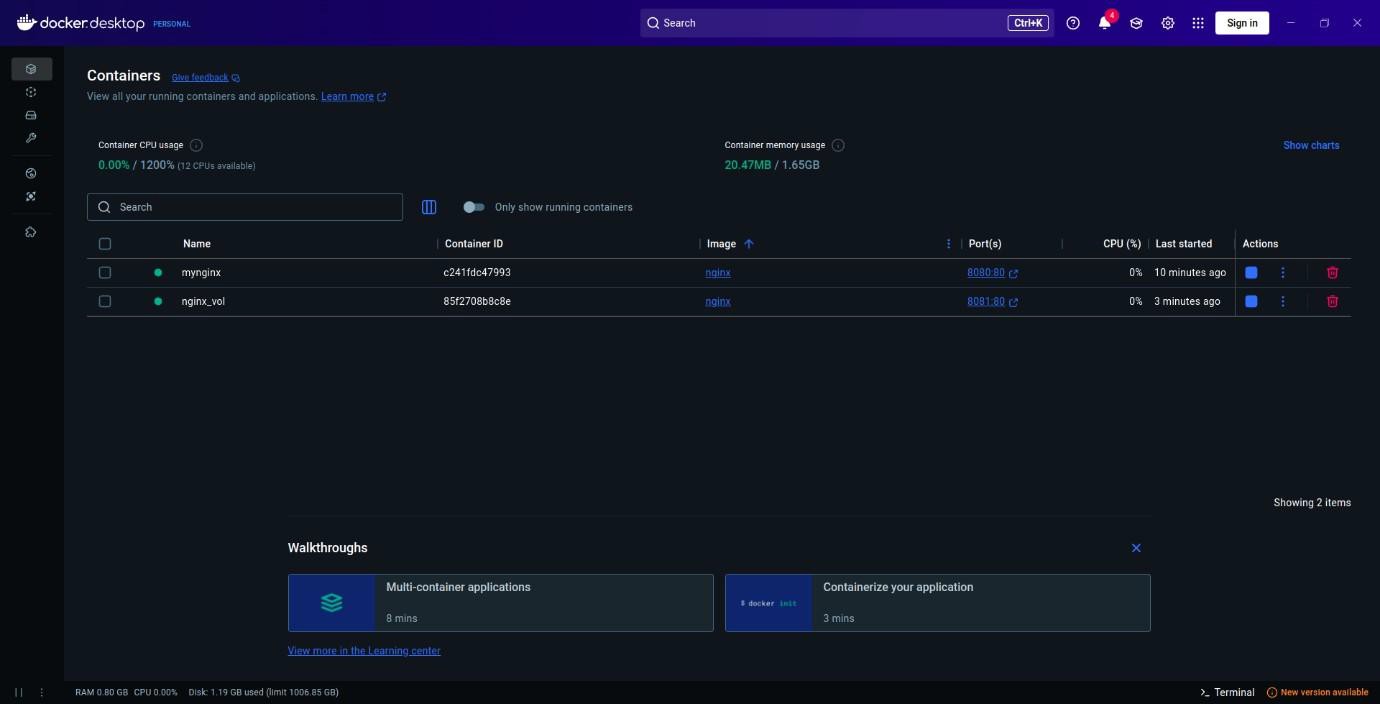
Now any data added to the /usr/share/nginx/html inside the container will persist even if the container is removed.



1. List Docker volumes docker volume ls



1. Remove a container docker rm -f mynginx Remove an image docker rmi nginx



# PRACTICAL-10

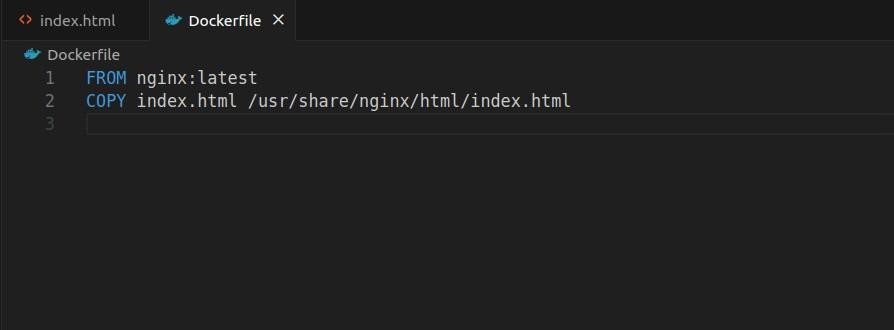
## Aim: Develop a simple containerized application using Docker

### Develop a Simple Containerized Application using Docker

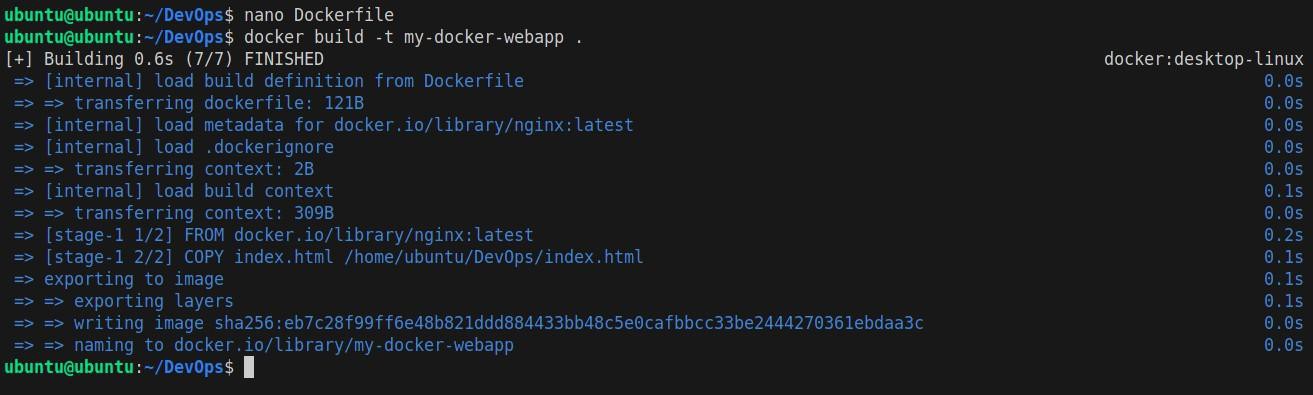
1. Index.html



1. DockerfIle :-



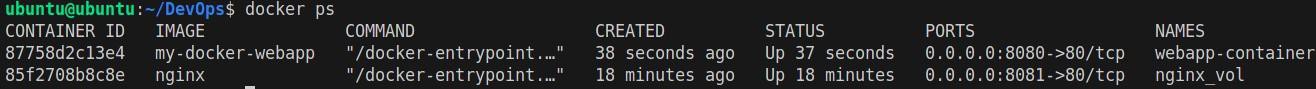
1. docker build -t my-docker-webapp .



1. docker run -d -p 8080:80 --name webapp-container my-docker-webapp



1. docker ps



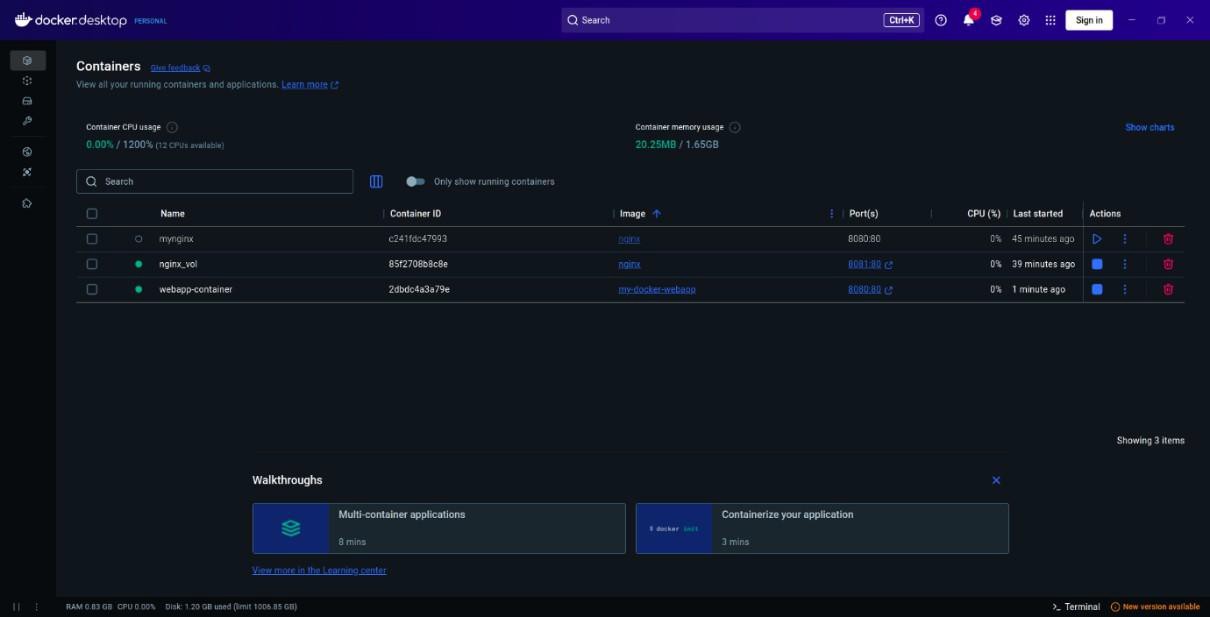
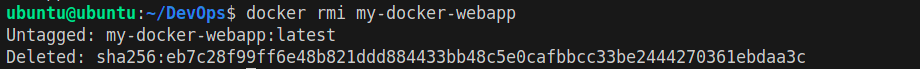
1. docker stop webapp-container

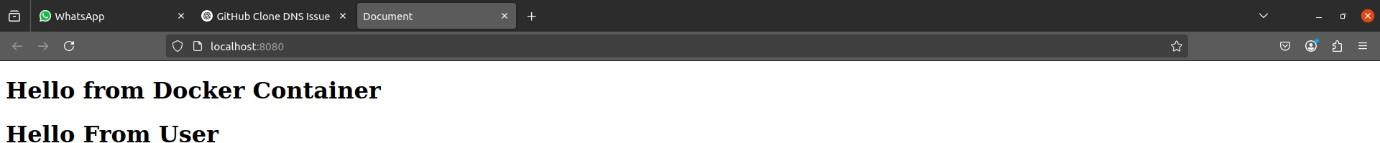


1. docker rm webapp-container



1. docker rmi my-docker-webapp

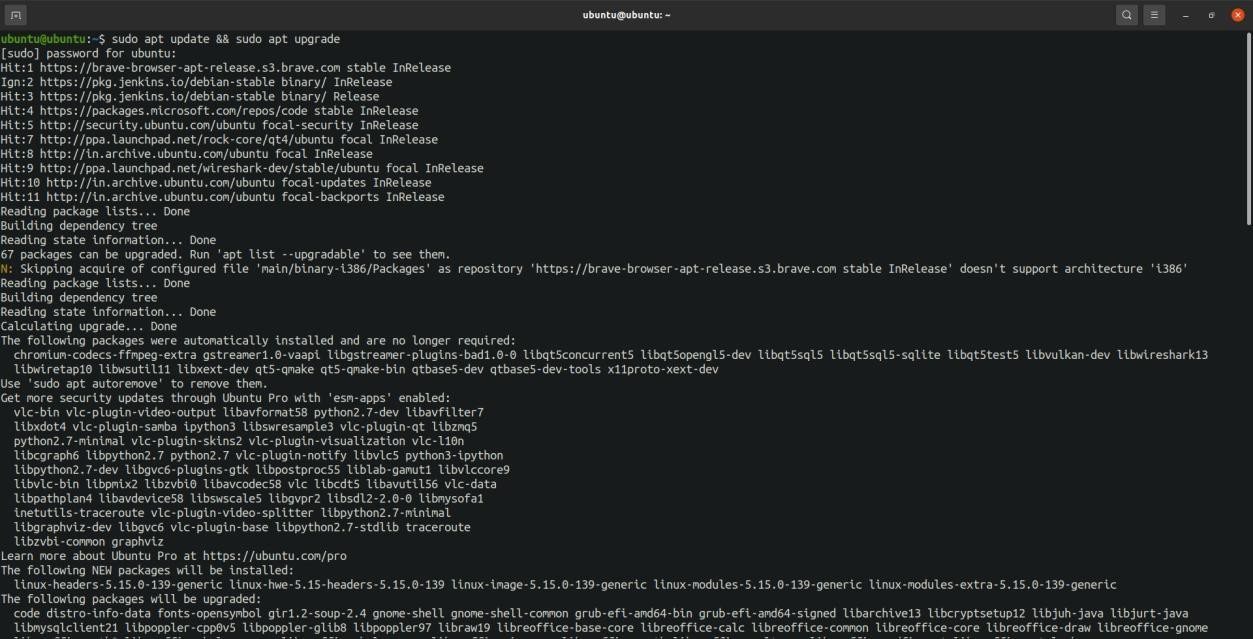




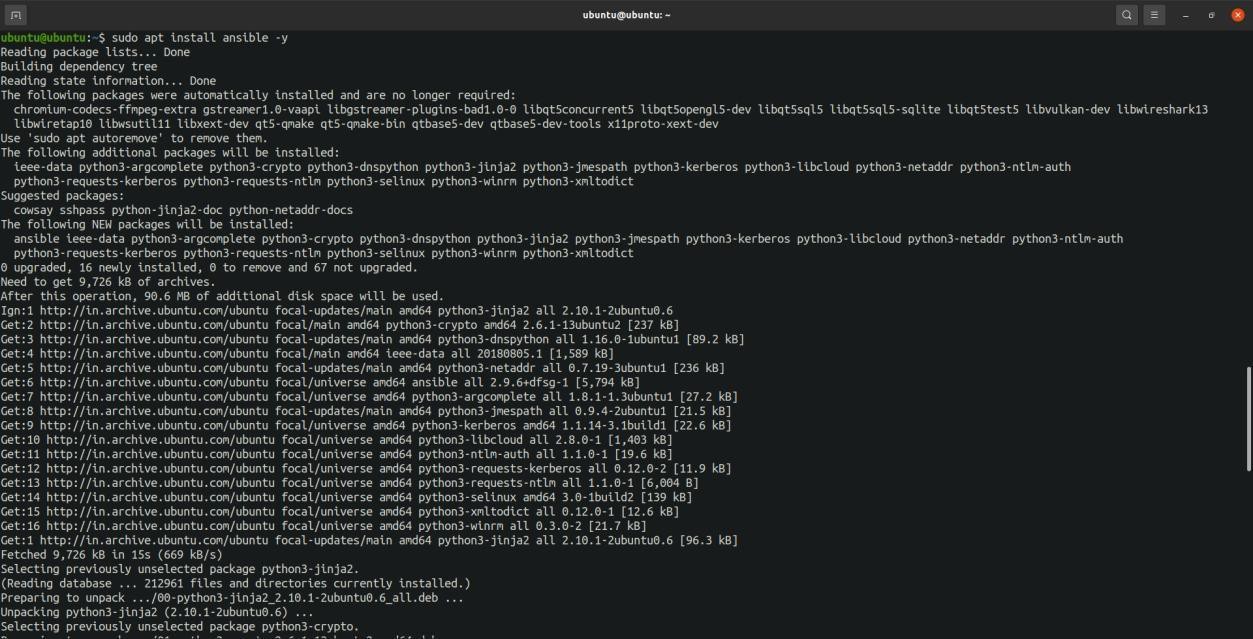
# PRACTICAL-11

## Aim: Ad-hoc Ansible Commands

Step 1: Update your VM



Step 2: Install Ansible



Step 3: Check version:

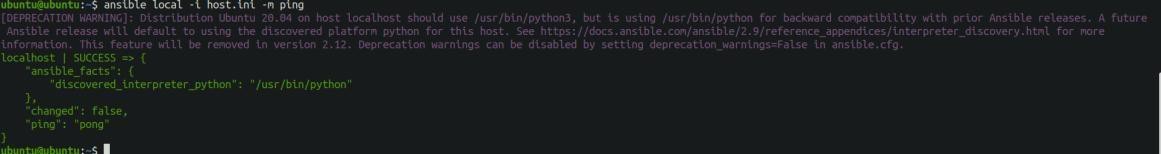






1. Ping the remote host

ansible local -i host.ini -m ping



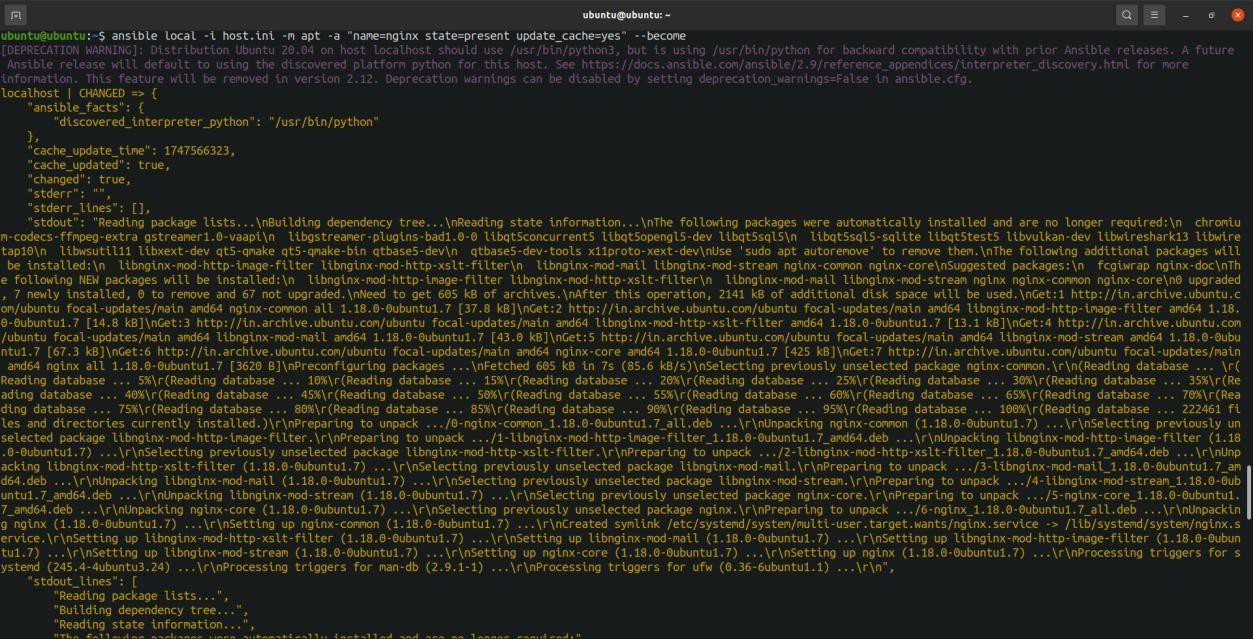
1. Check uptime

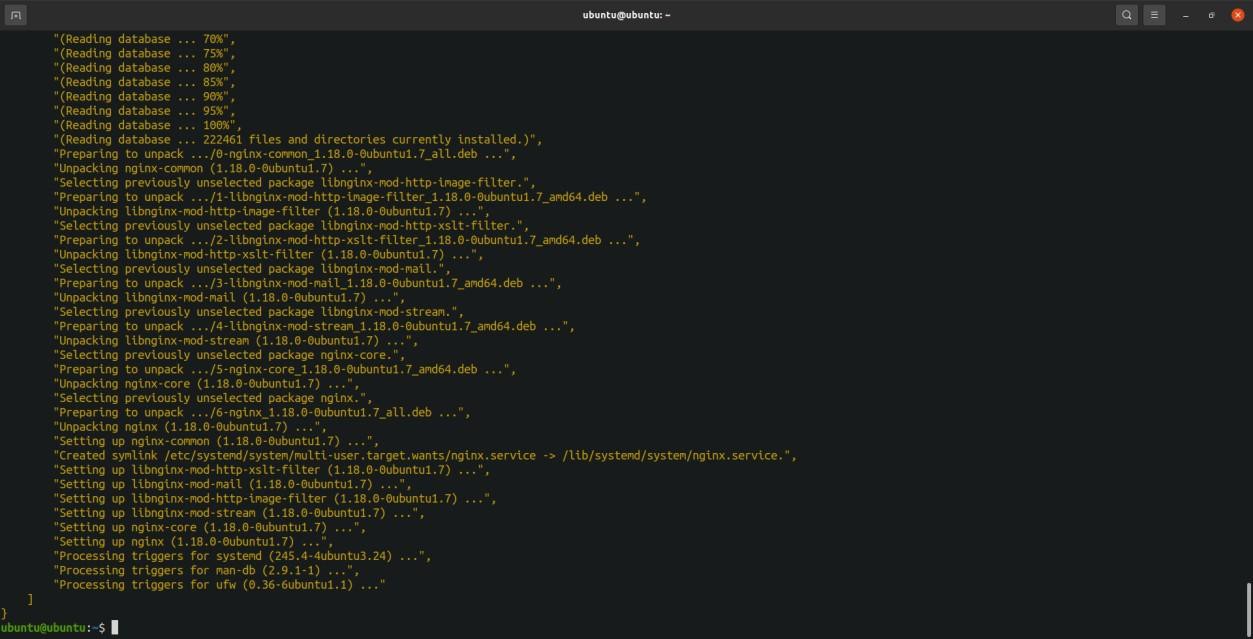
ansible local -i host.ini -a "uptime"



1. Install a package

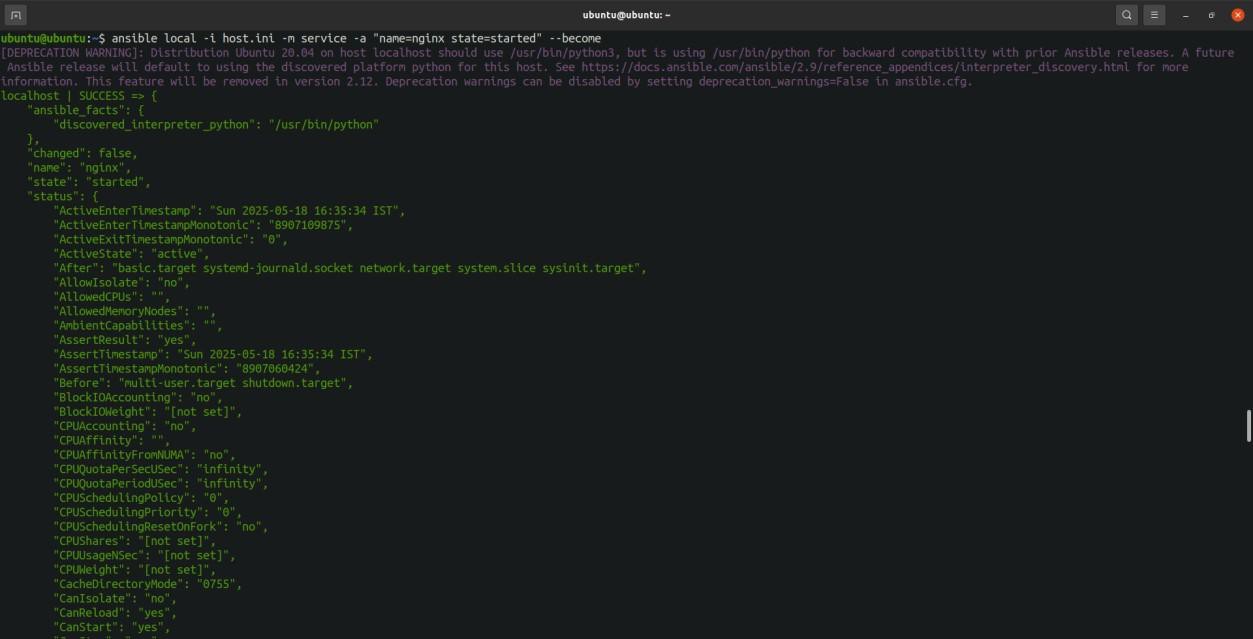
ansible local -i host.ini -m apt -a "name=nginx state=present update\_cache=yes" –become

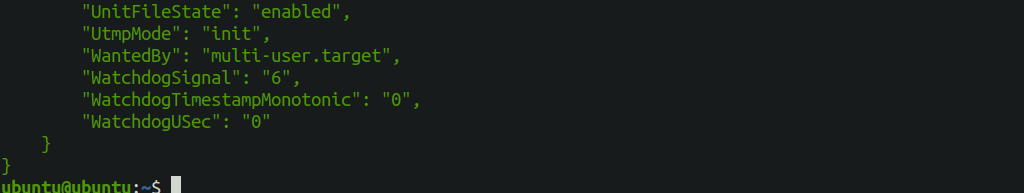




1. Start a service

ansible local -i host.ini -m service -a "name=nginx state=started" –become





# PRACTICAL-12

## Aim: Using Ansible Playbooks

### Install and Start Nginx

install\_nginx.yml:

- name: Install and start Nginx on web servers hosts: webservers

become: true tasks:

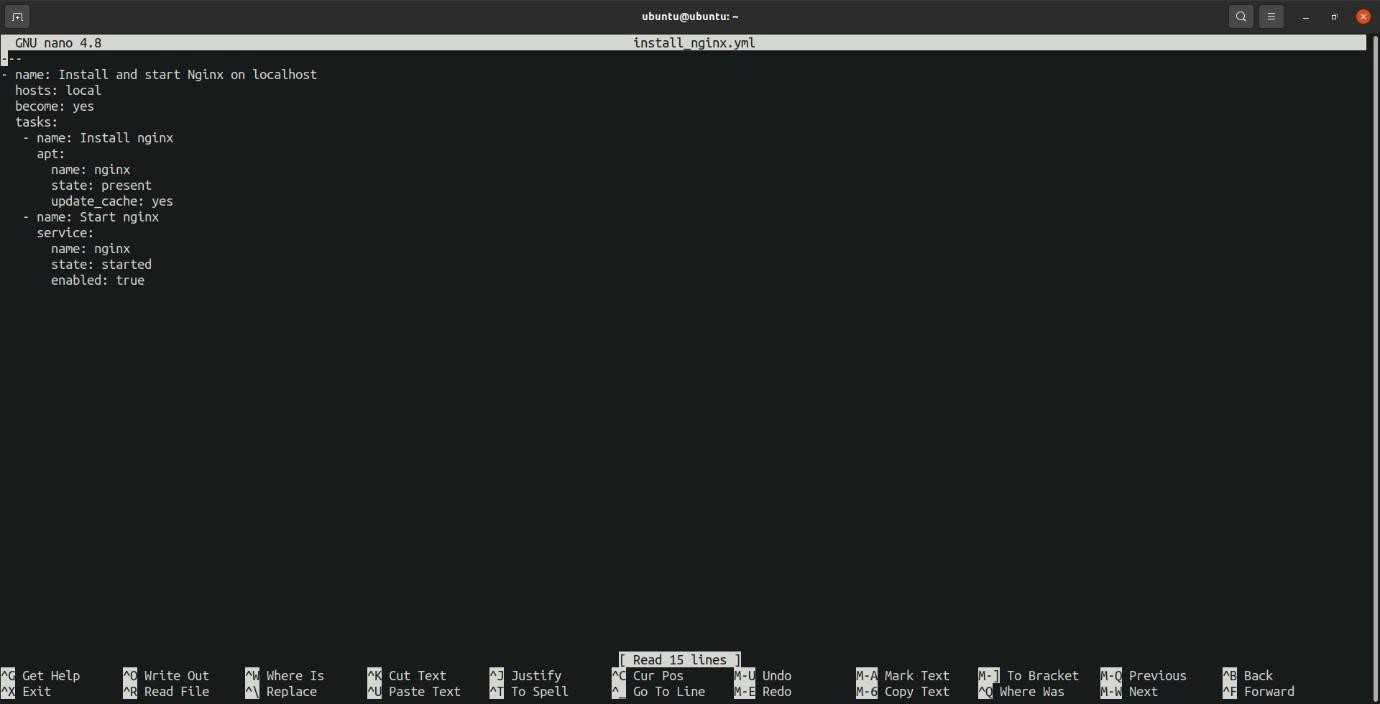
* name: Install Nginx apt:

name: nginx state: present update\_cache: yes

* name: Start Nginx service:

name: nginx state: started enabled: true





Run the Playbook:

ansible-playbook -i hosts.ini install\_nginx.yml

