| 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 NO.1

## Basic Git commands

1. Check git
   version git
   –version

```
ubuntu@ubuntu:~$ git --version
git version 2.25.1
ubuntu@ubuntu:~$
```

   –

2. Create folder and initiliaze.

```
ubuntu@ubuntu:~$ git --version
git version 2.25.1
ubuntu@ubuntu:~$ mkdir newuser
ubuntu@ubuntu:~$ cd newuser/
ubuntu@ubuntu:~/newuser$ git init
Initialized empty Git repository in /home/ubuntu/newuser/.git/
ubuntu@ubuntu:~/newuser$
```

3. Configure Git
   git config --global user.name "usernewncrd"
   git config --global user.email “symca669@gmail.com”

```
ubuntu@ubuntu:~/newuser$ git config --global user.name "usernewncrd"
ubuntu@ubuntu:~/newuser$ git config --global user.email "symca669@gmail.com"
ubuntu@ubuntu:~/newuser$
```

4. Create a new project folder
   mkdir git-demo
   cd git-demo

```
ubuntu@ubuntu:~/newuser$ mkdir git-demo
ubuntu@ubuntu:~/newuser$ cd git-demo/
ubuntu@ubuntu:~/newuser/git-demo$
```

5. git init

```
ubuntu@ubuntu:~/newuser/git-demo$ git init
Initialized empty Git repository in /home/ubuntu/newuser/git-demo/.git/
ubuntu@ubuntu:~/newuser/git-demo$
```

6. Create and track a file:
   echo "Hello User" >
   file.txt git add file.txt
   git commit -m "Initial commit"

```
ubuntu@ubuntu:~/newuser/git-demo$ echo "Hello User"> file.txt
ubuntu@ubuntu:~/newuser/git-demo$ git add file.txt
ubuntu@ubuntu:~/newuser/git-demo$ git commit -m "Initial Commit"
[master (root-commit) 5da5867] Initial Commit
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
ubuntu@ubuntu:~/newuser/git-demo$
```

7. Check status and
   log: git status
   git log

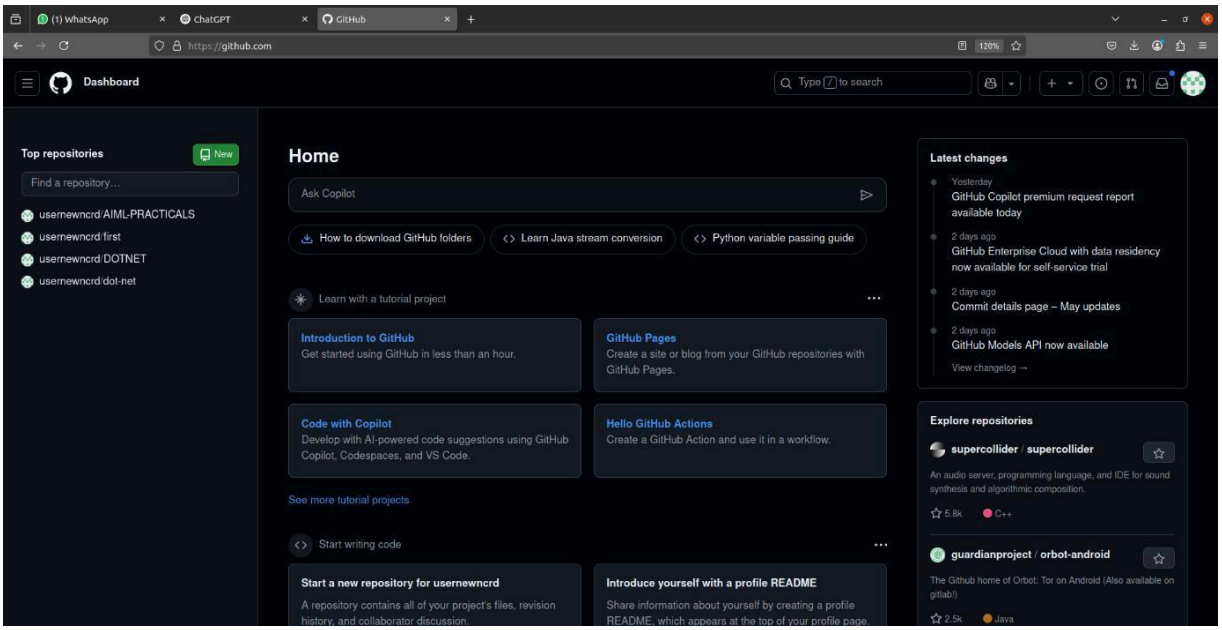
```
ubuntu@ubuntu:~/newuser/git-demo$ git status
On branch master
nothing to commit, working tree clean
ubuntu@ubuntu:~/newuser/git-demo$ git log
commit 5da586754b11433e7ab5ed5d1eafad9ad22d9289 (HEAD -> master)
Author: usernewncrd <symca669@gmail.com>
Date:   Sun May 18 13:52:53 2025 +0530

    Initial Commit
ubuntu@ubuntu:~/newuser/git-demo$
```

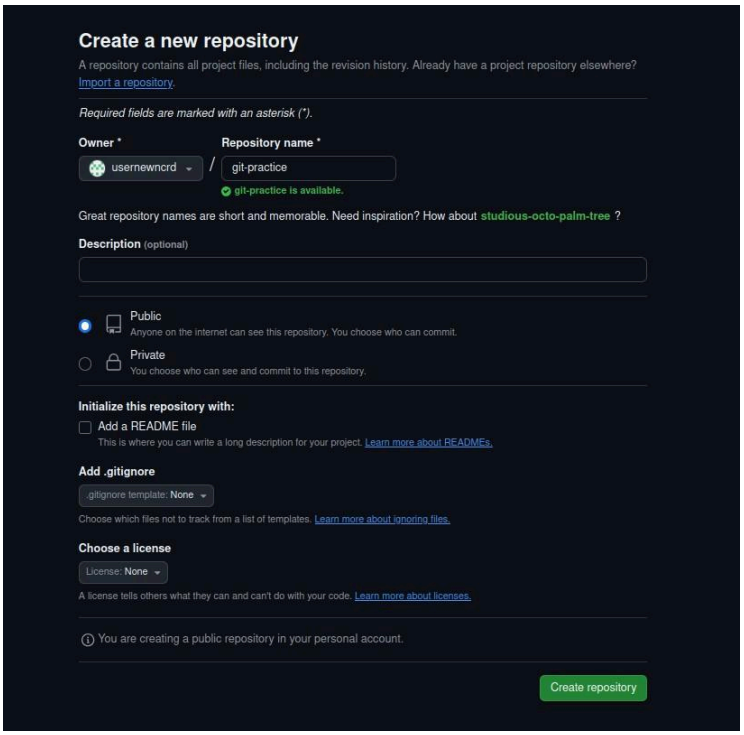
# PRACTICAL NO.2

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

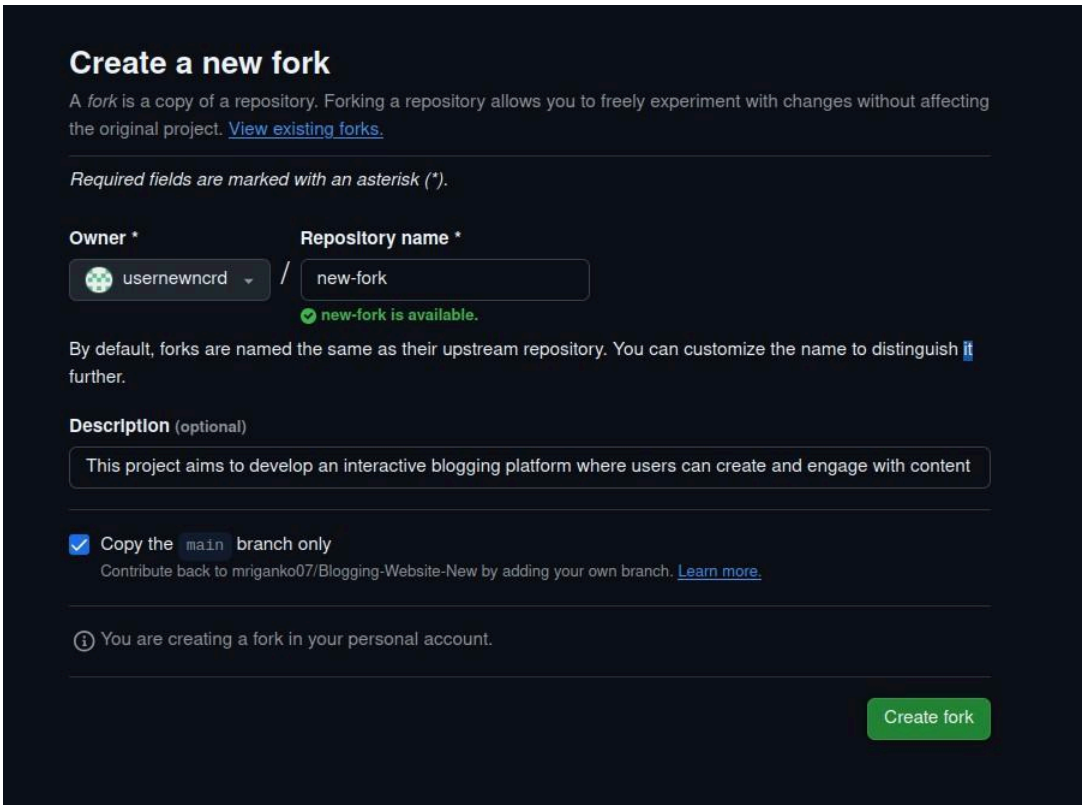
1. Create a GitHub account and log in.




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



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




4. Clone the forked repo:
   git clone https://github.com/usernewncrd/git-practice.git cd
   git-practice

```
ubuntu@ubuntu:~/newuser/git-demo$ git clone https://github.com/usernewncrd/new-fork
Cloning into 'new-fork'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 7 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (7/7), 28.85 KiB | 1.07 MiB/s, done.
ubuntu@ubuntu:~/newuser/git-demo$ cd new-fork/
ubuntu@ubuntu:~/newuser/git-demo/new-fork$
```

5. Create a branch:
   git checkout -b feature

```
ubuntu@ubuntu:~/newuser/git-demo/new-fork$ git checkout -b feature
Switched to a new branch 'feature'
```

6. Make changes, then commit:
   echo "Feature added" >> newfile.txt
   git add .
   git commit -m "Added new feature"

```
ubuntu@ubuntu:~/newuser/git-demo/new-fork$ echo "Feature Added" >> newfile.txt
ubuntu@ubuntu:~/newuser/git-demo/new-fork$ git add .
ubuntu@ubuntu:~/newuser/git-demo/new-fork$ git commit -m "Added new feature"
[feature ec92d67] Added new feature
 1 file changed, 1 insertion(+)
 create mode 100644 newfile.txt
```

7. Merge branch into main:
   git checkout master
   git merge feature

```
ubuntu@ubuntu:~/newuser/git-demo/new-fork$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
ubuntu@ubuntu:~/newuser/git-demo/new-fork$ git merge feature
Updating d0bf9b1..ec92d67
Fast-forward
 newfile.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 newfile.txt
ubuntu@ubuntu:~/newuser/git-demo/new-fork$
```

8. Rebase branch (alternative to merge):
   git checkout feature
   git rebase master

```
ubuntu@ubuntu:~/newuser/git-demo/new-fork$ git checkout feature
Switched to branch 'feature'
ubuntu@ubuntu:~/newuser/git-demo/new-fork$ git rebase main
Current branch feature is up to date.
ubuntu@ubuntu:~/newuser/git-demo/new-fork$
```

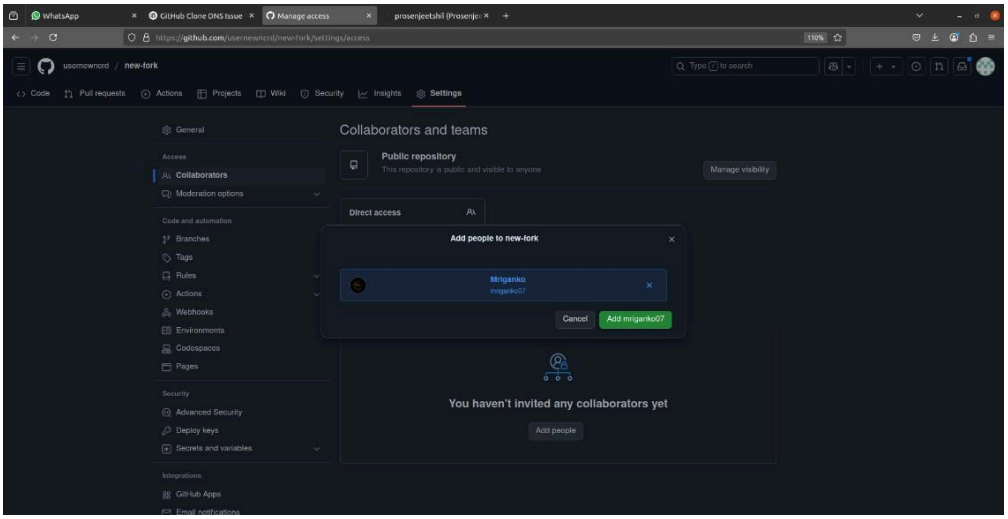
9. Push to GitHub:
   git push origin feature

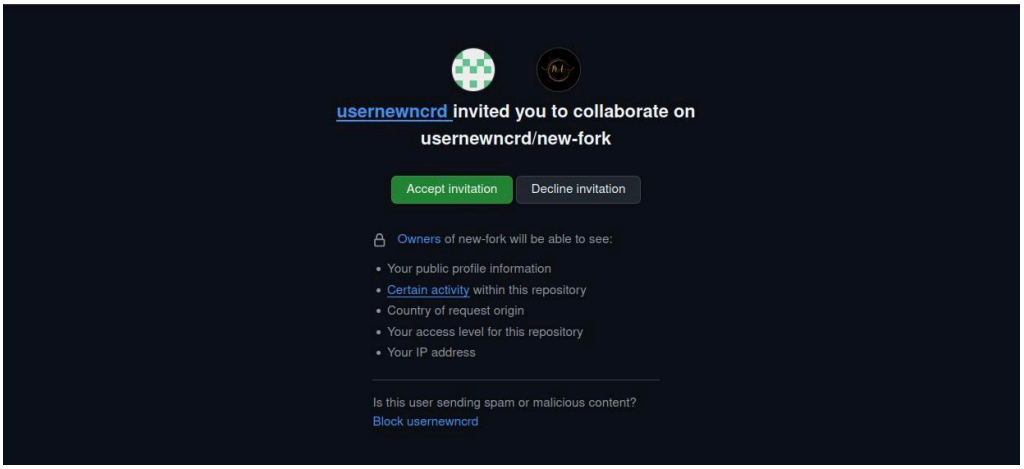
```
ubuntu@ubuntu:~/newuser/git-demo/new-fork$ git push origin feature
Username for 'https://github.com': usernewncrd
Password for 'https://usernewncrd@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 283 bytes | 283.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:      https://github.com/usernewncrd/new-fork/pull/new/feature
remote:
To https://github.com/usernewncrd/new-fork
 * [new branch]      feature -> feature
ubuntu@ubuntu:~/newuser/git-demo/new-fork$
```

# PRACTICAL NO.3

## Using Git for Collaboration

1. Using Git for Collaboration




2. Friend clones the repo:
   git clone https://github.com/usernewncrd/git-practice.git
   cd team-repo
   git checkout -b bug-fix

```
ubuntu@ubuntu:~/newuser/git-demo$ git clone https://github.com/usernewncrd/git-practice.git
Cloning into 'git-practice'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 1000.37 KiB | 2.44 MiB/s, done.
ubuntu@ubuntu:~/newuser/git-demo$ cd team-repo
bash: cd: team-repo: No such file or directory
ubuntu@ubuntu:~/newuser/git-demo$ git checkout -b bug-fix
Switched to a new branch 'bug-fix'
ubuntu@ubuntu:~/newuser/git-demo$
```

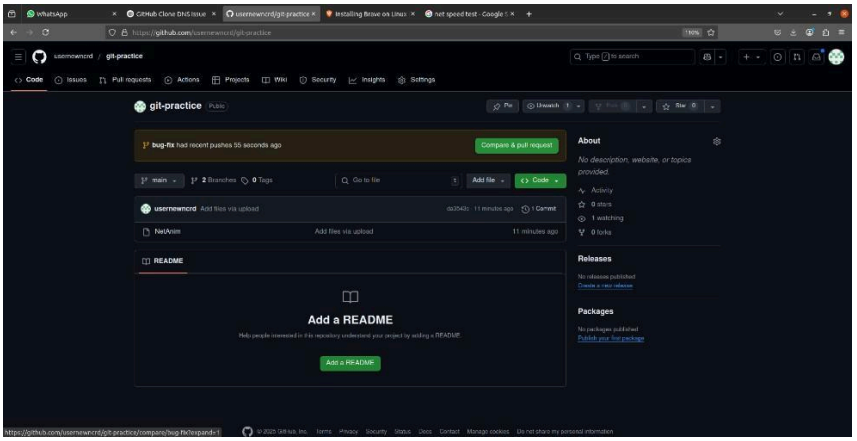
3. Friend makes changes and pushes:
   echo "Bug fixed" >> bug.txt
   git add .
   git commit -m "Fixed a bug"

```
ubuntu@ubuntu:~/newuser/git-demo$ echo "Bug fixed">>bug.txt
ubuntu@ubuntu:~/newuser/git-demo$ git add .
warning: adding embedded git repository: git-practice
hint: You've added another git repository inside your current repository.
hint: Clones of the outer repository will not contain the contents of
hint: the embedded repository and will not know how to obtain it.
hint: If you meant to add a submodule, use:
hint:
hint:   git submodule add <url> git-practice
hint:
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:
hint:   git rm --cached git-practice
hint:
hint: See "git help submodule" for more information.
warning: adding embedded git repository: new-fork
ubuntu@ubuntu:~/newuser/git-demo$ git commit -m "Fixed the bug"
[bug-fix a816be3] Fixed the bug
 3 files changed, 3 insertions(+)
 create mode 100644 bug.txt
 create mode 160000 git-practice
 create mode 160000 new-fork
ubuntu@ubuntu:~/newuser/git-demo$
```

4. git push origin bug-fix

```
ubuntu@ubuntu:~/newuser/git-demo$ git push origin bug-fix
Username for 'https://github.com': usernewncrd
Password for 'https://usernewncrd@github.com':
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (6/6), 549 bytes | 549.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'bug-fix' on GitHub by visiting:
remote:      https://github.com/usernewncrd/git-practice/pull/new/bug-fix
remote:
To https://github.com/usernewncrd/git-practice.git
 * [new branch]      bug-fix -> bug-fix
ubuntu@ubuntu:~/newuser/git-demo$
```

5. Pull Request



# PRACTICAL NO.4

## Collaborating and Cloning using GitHub

1. Clone a public repository:
   git clone https://github.com/usernewncrd /git-practice.git

```
ubuntu@ubuntu:~/newuser/git-demo$ git clone https://github.com/usernewncrd/git-practice.git
Cloning into 'git-practice'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 1000.37 KiB | 2.44 MiB/s, done.
```

2. Create a branch:
   git checkout -b update-readme

```
ubuntu@ubuntu:~/newuser/git-demo$ git checkout -b update-readme
Switched to a new branch 'update-readme'
ubuntu@ubuntu:~/newuser/git-demo$
```

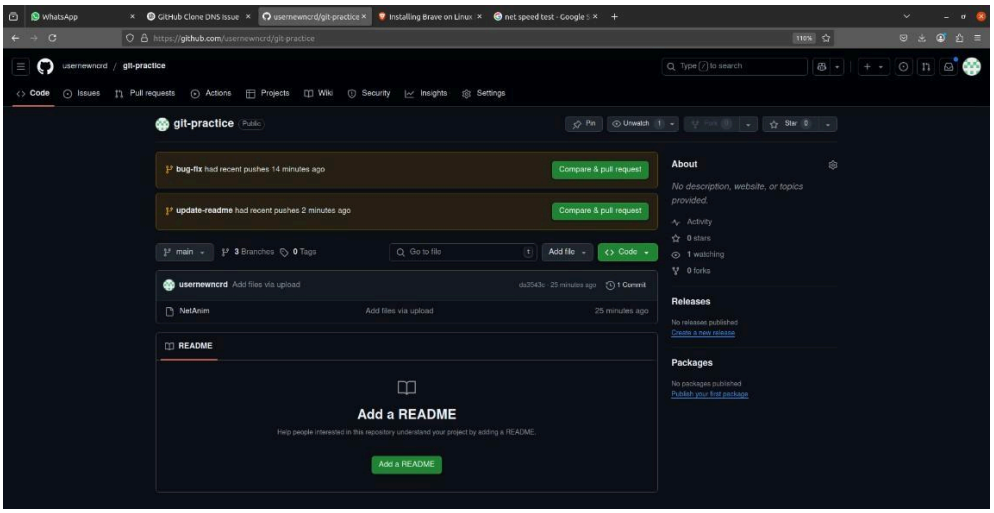
3. Edit and commit changes:
   echo "Added a line" >>
   README.md git add README.md
   git commit -m "Updated README"

```
ubuntu@ubuntu:~/newuser/git-demo$ echo "Added a line">>README.md
ubuntu@ubuntu:~/newuser/git-demo$ git add README.md
ubuntu@ubuntu:~/newuser/git-demo$ git commit -m "Updated README"
[update-readme 11aa668] Updated README
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
ubuntu@ubuntu:~/newuser/git-demo$
```

4. Push and open pull request:

```
ubuntu@ubuntu:~/newuser/git-demo$ git push origin update-readme
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 290 bytes | 290.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'update-readme' on GitHub by visiting:
remote:      https://github.com/usernewncrd/git-practice/pull/new/update-readme
remote:
To https://github.com/usernewncrd/git-practice.git
 * [new branch]      update-readme -> update-readme
ubuntu@ubuntu:~/newuser/git-demo$
```

5. git push origin update-readme

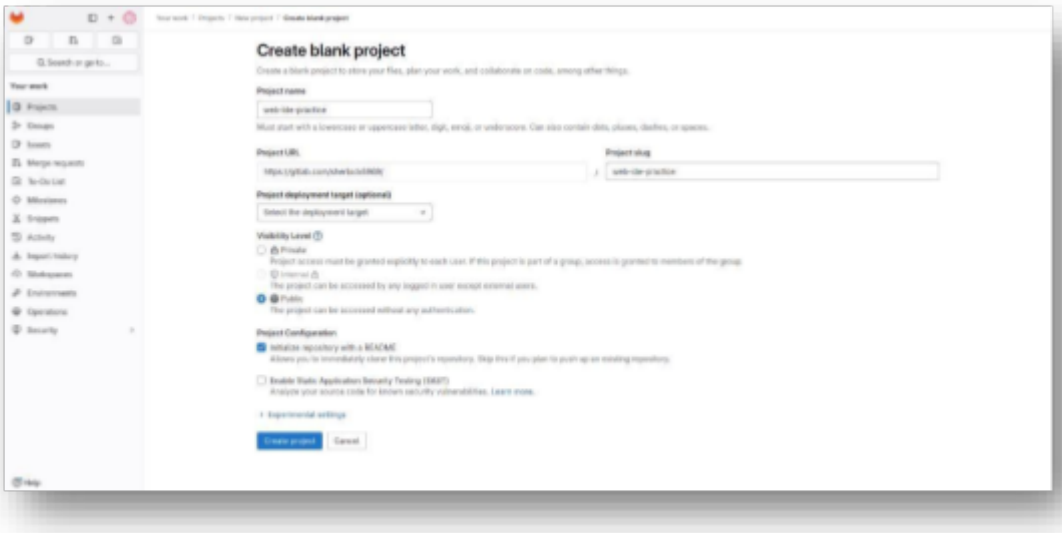


# PRACTICAL NO.5

## Using GitLab Web IDE

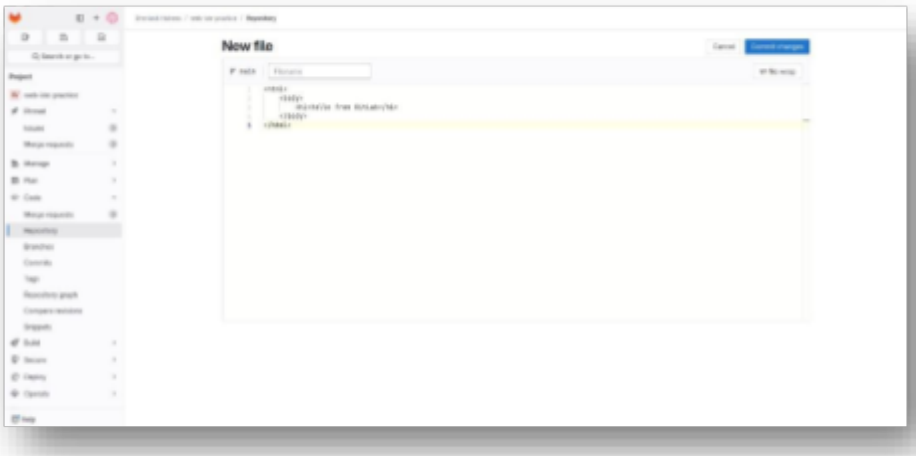
### Steps:

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

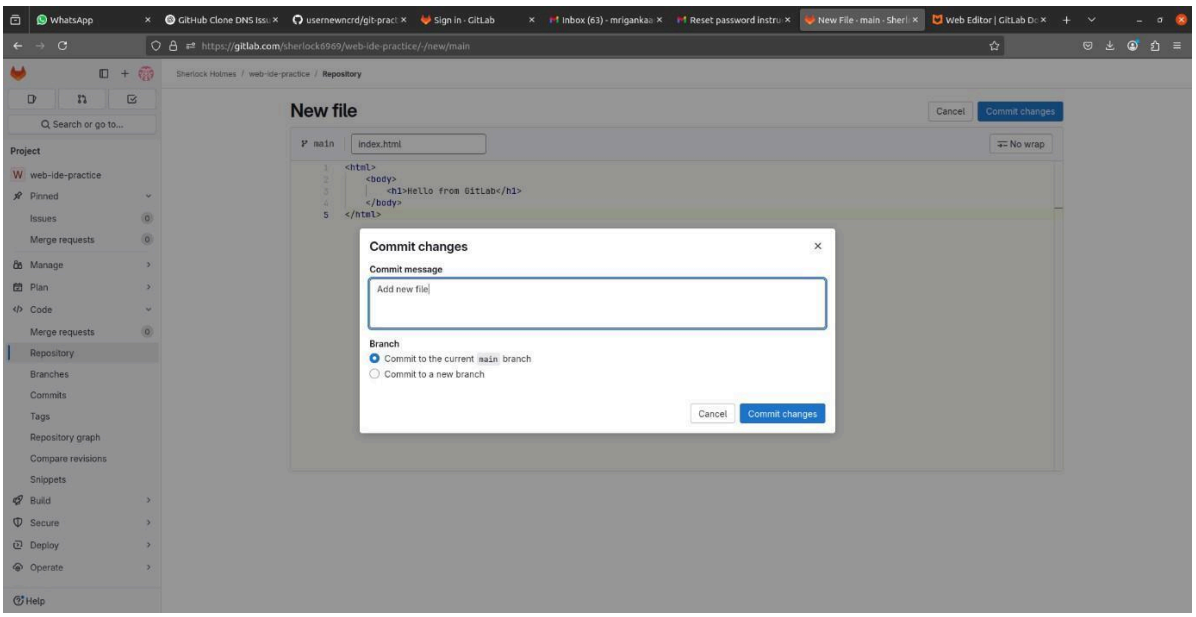


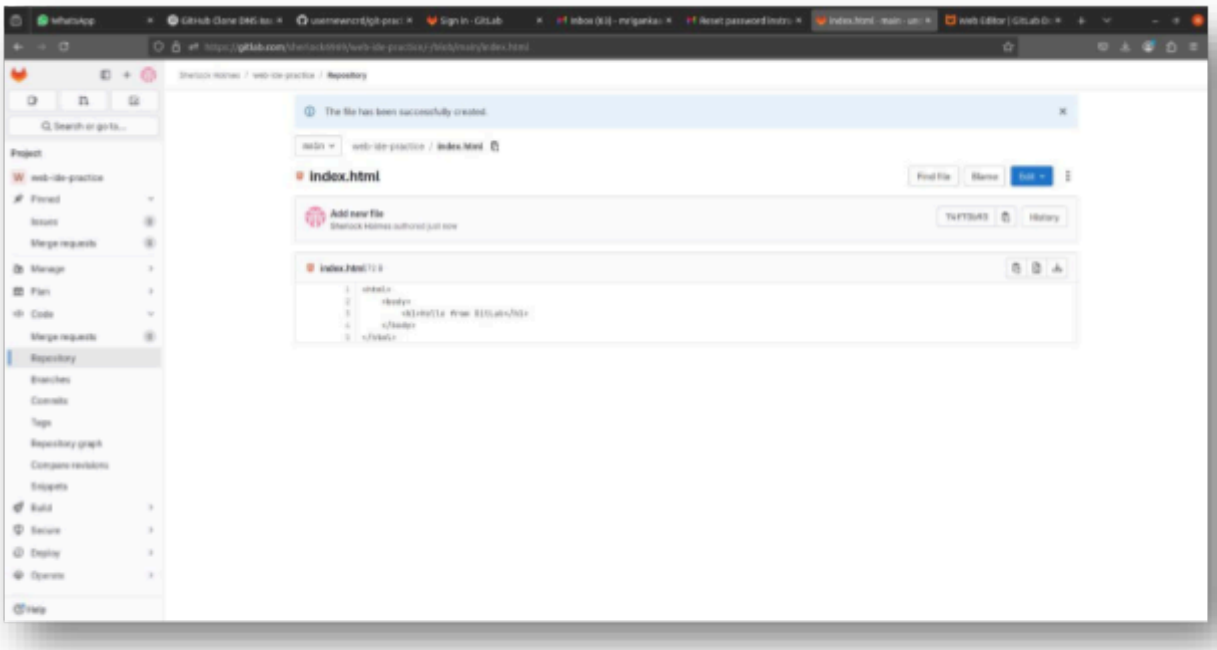

4. Create a file (index.html):

```
<html>
 <body>
    <h1>Hello from GitLab</h1>
  </body>
</html>
```



4. Click Commit and push changes.

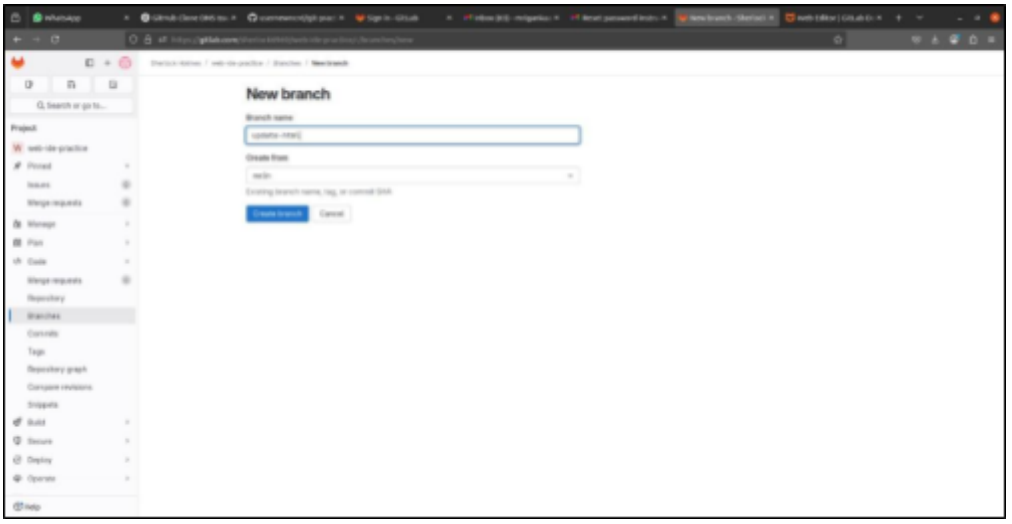


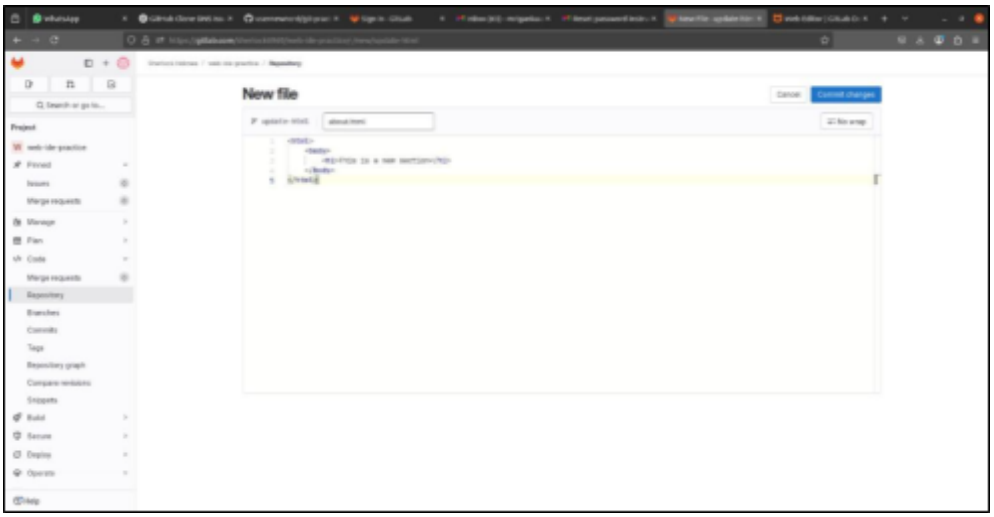
# PRACTICAL NO.6

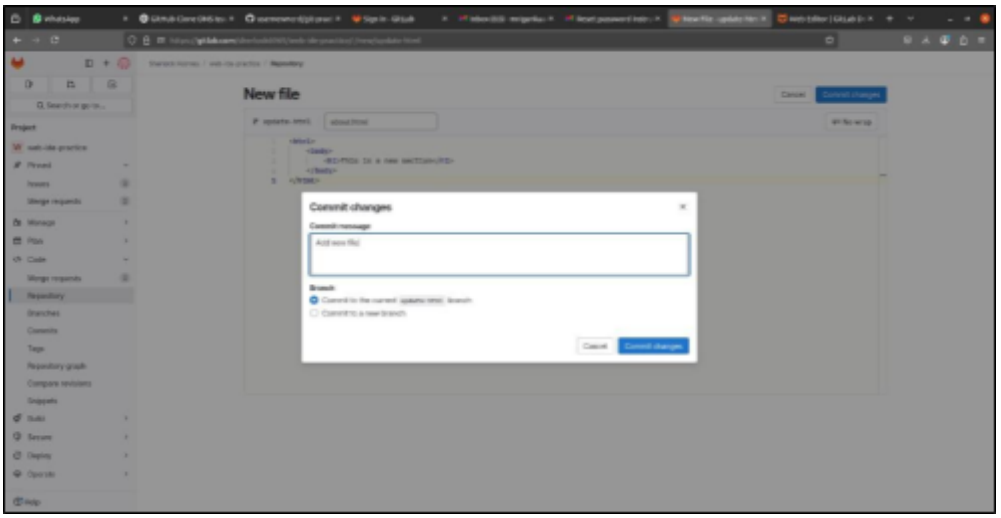
## Performing merge requests using GitLab

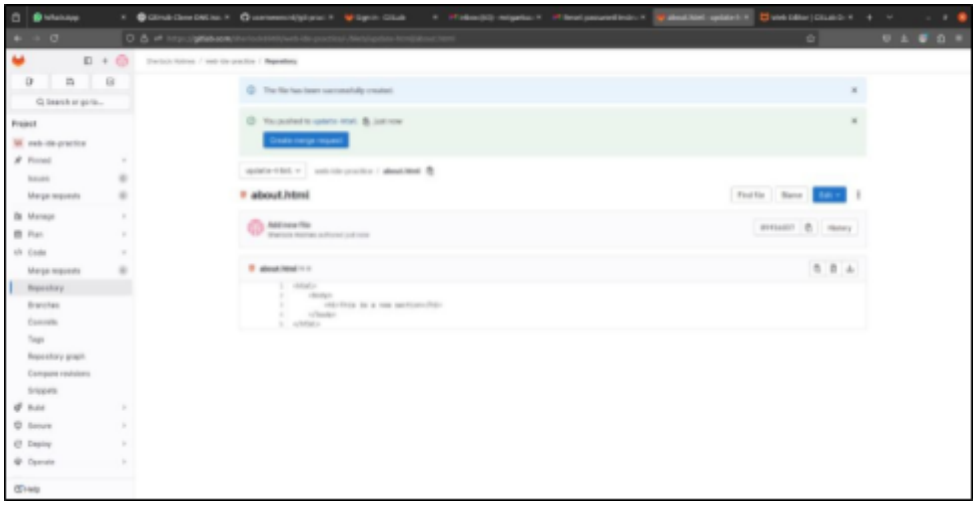
1. Create a new branch in Web IDE.



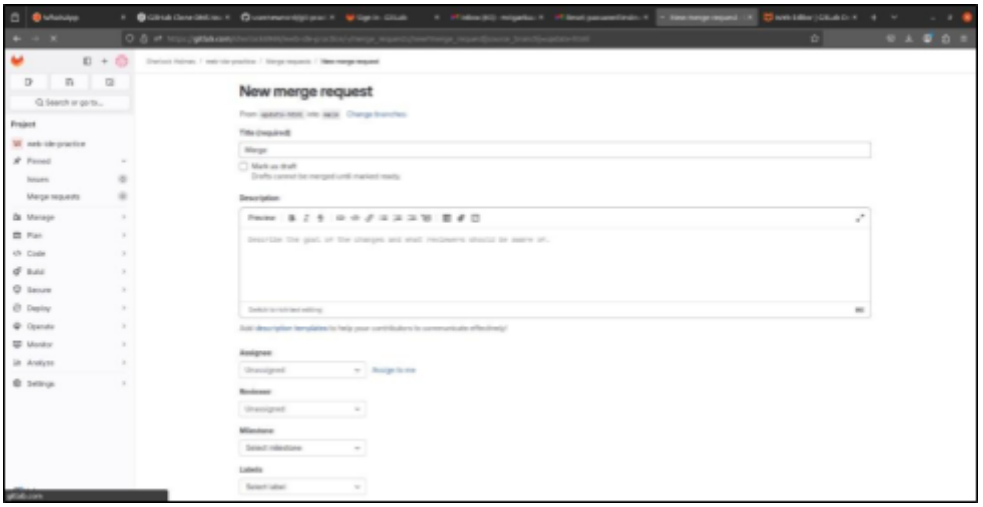

2. Add/edit a file and commit.



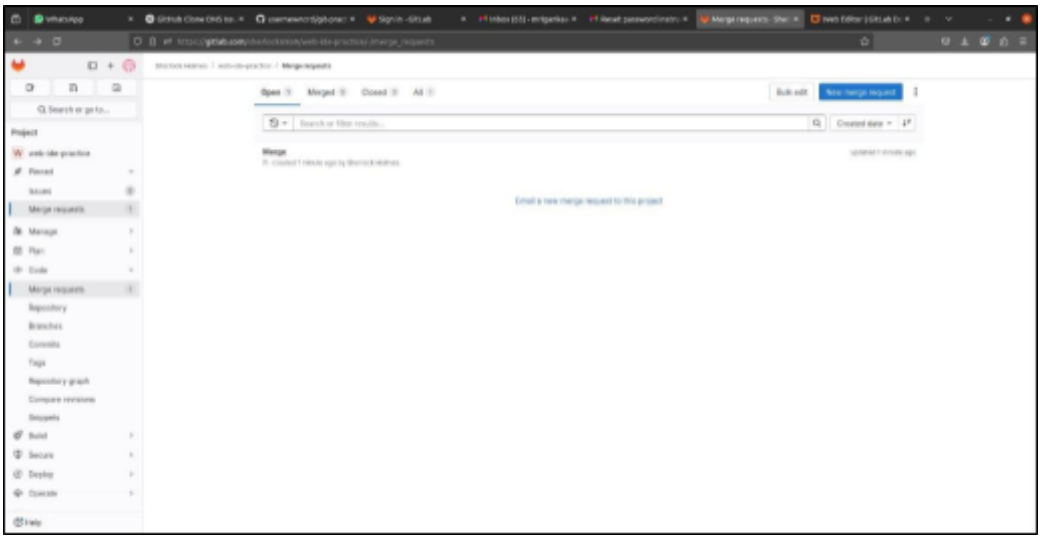


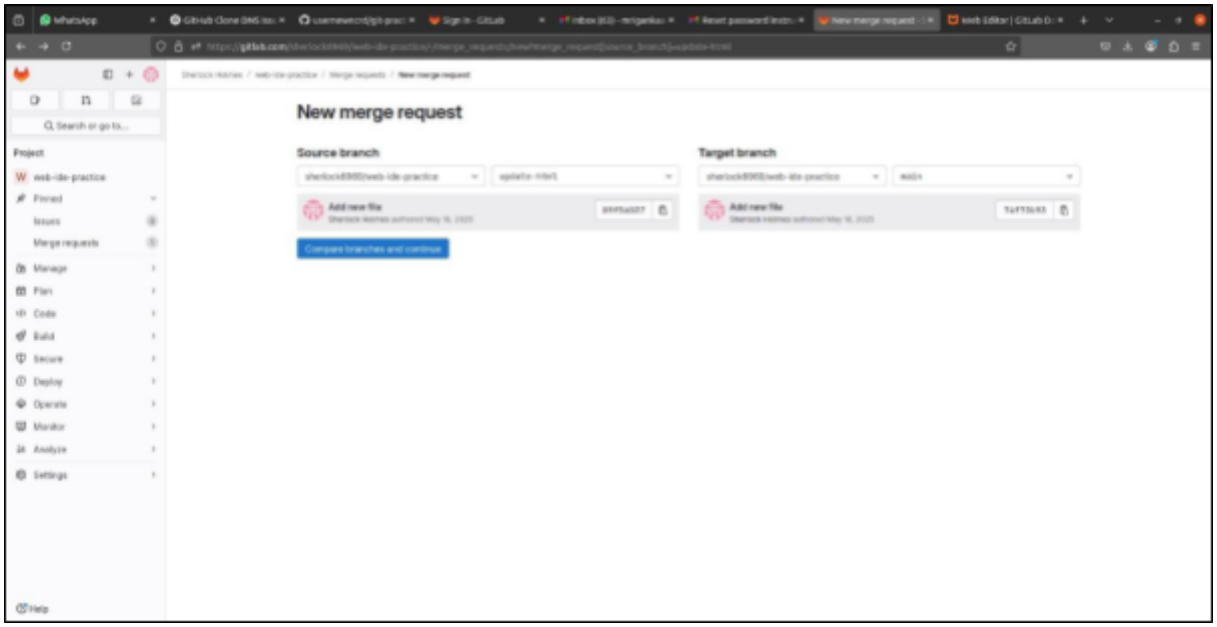

3. Click on Merge Requests > New Merge Request.



4. Select source and target branches.



5. Submit and merge after review.

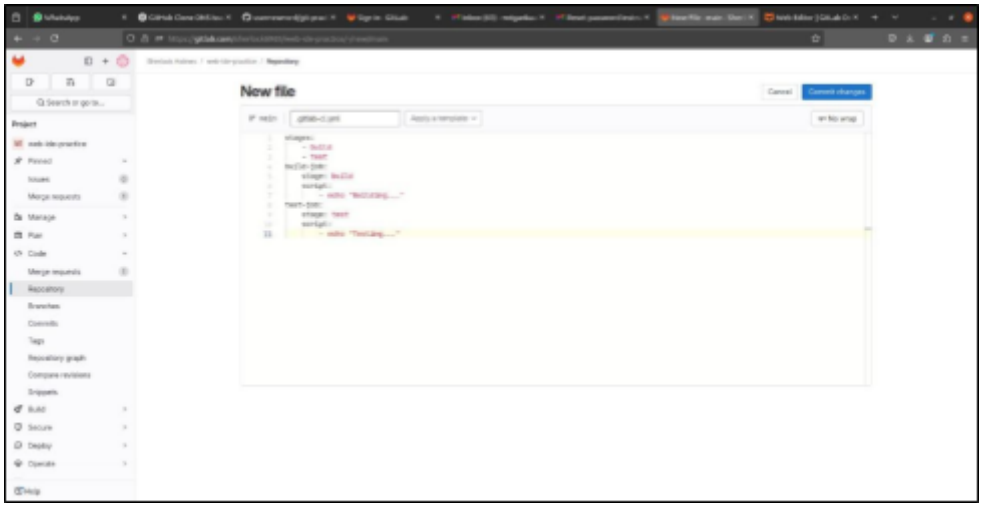


# PRACTICAL NO.7

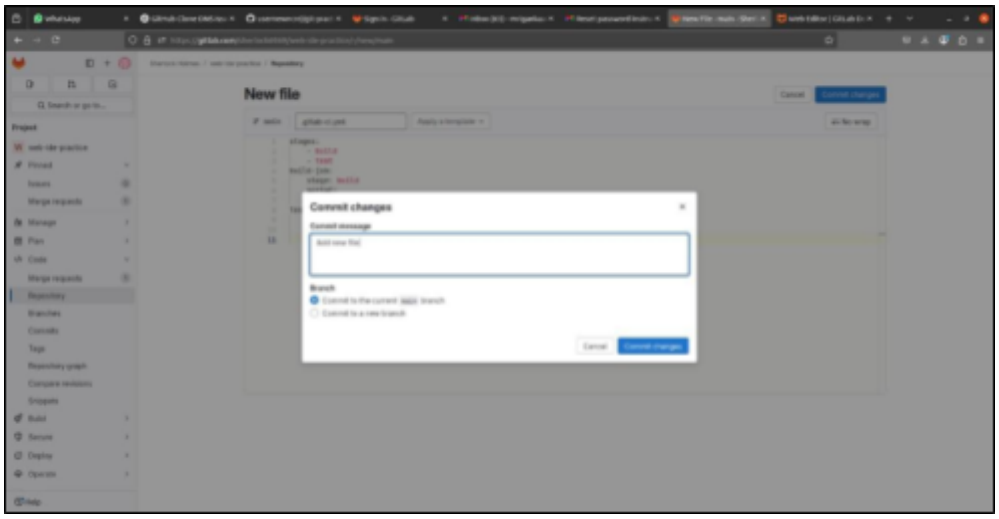
## Workflow management in GitLab

## Steps:

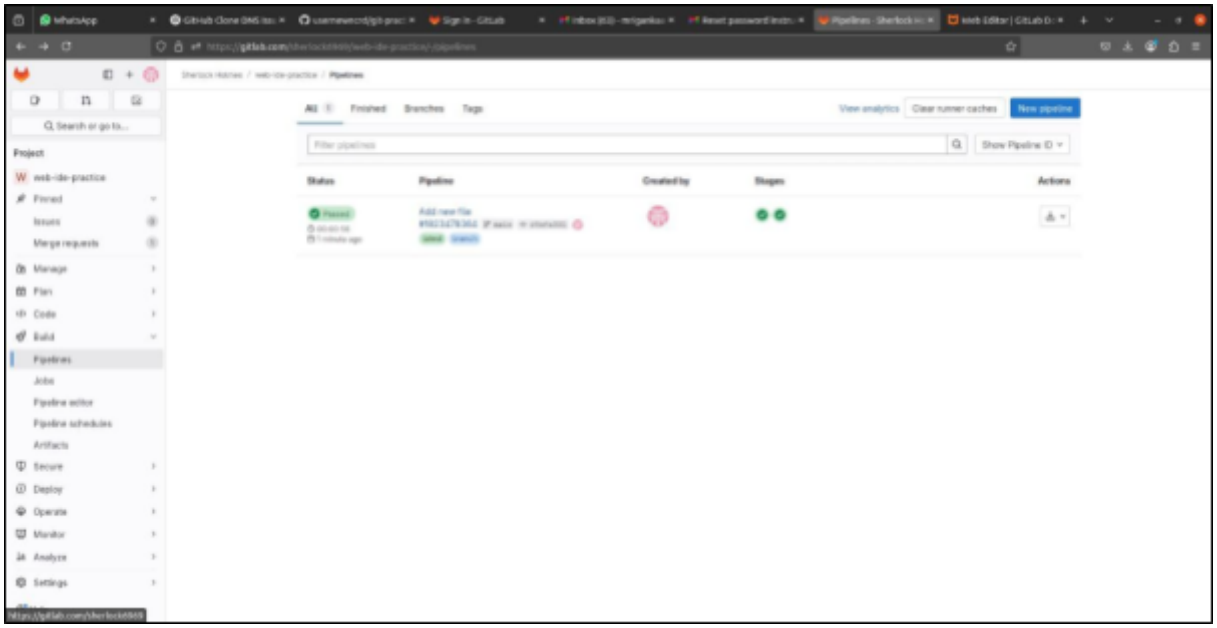
1. In your repo, create
   .gitlab-ci.yml: stages:
   - build
   - test
   build-job:
   stage: build
   script:
   - echo
   "Building..."
   test-job:
   stage: test
   script:
   - echo "Testing..."




2. Commit and push.



3. Go to CI/CD > Pipelines and view the build/test stages.

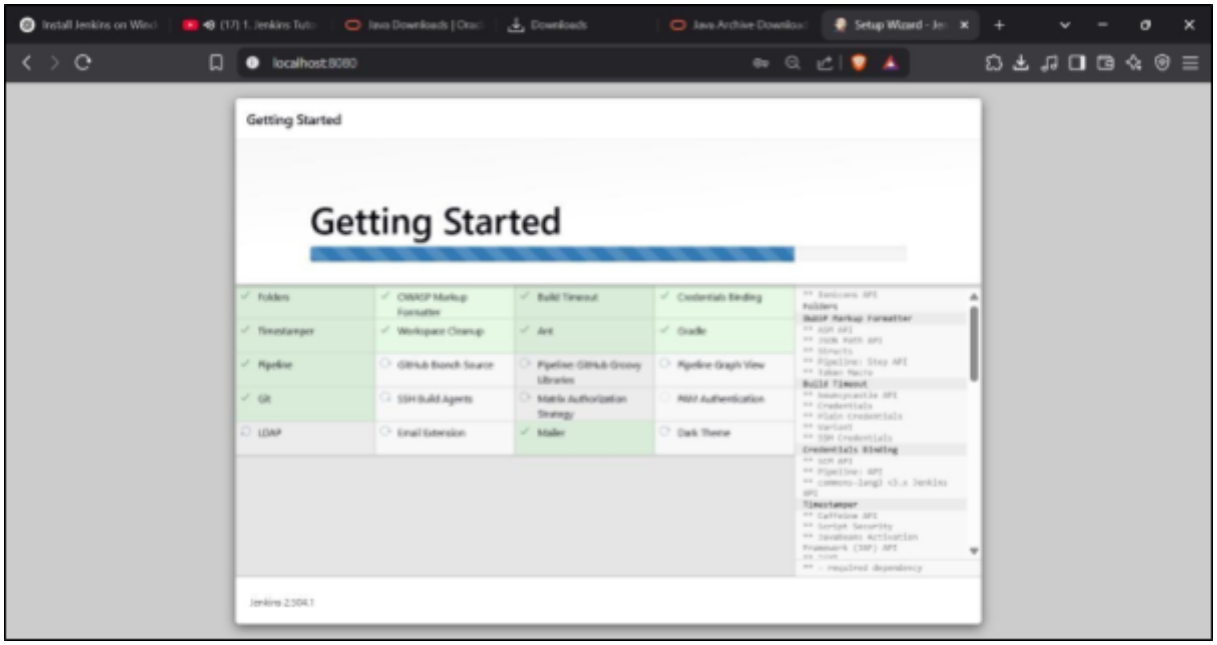



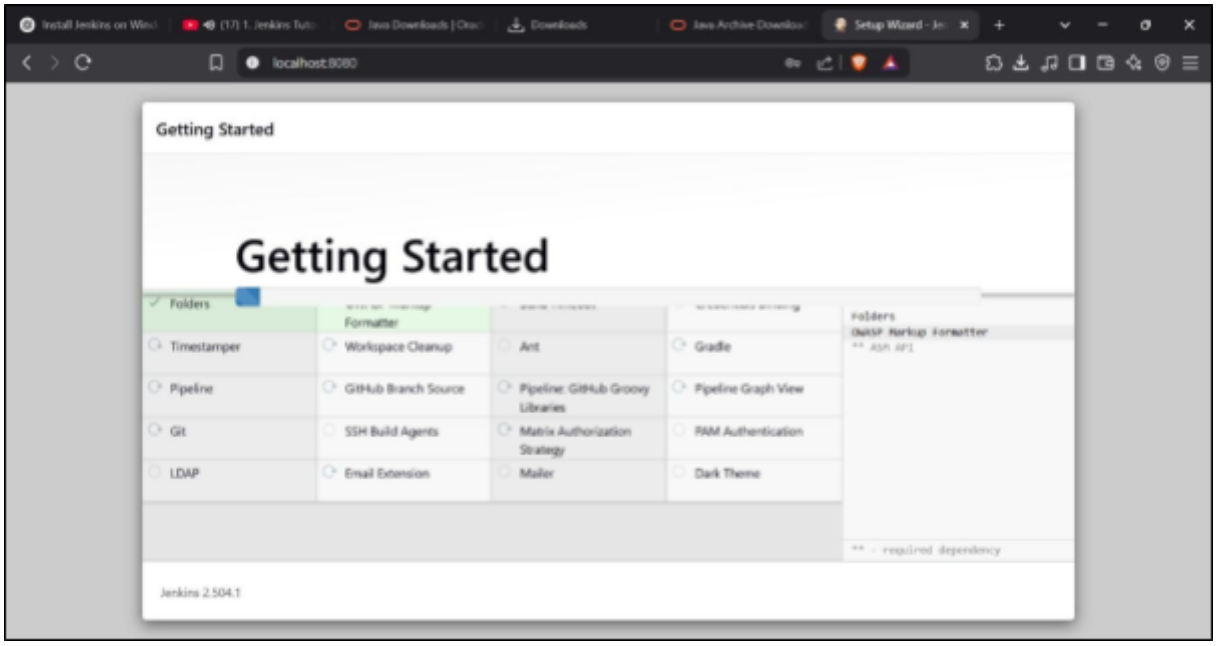
# PRACTICAL NO.-8

## Demonstrate Continuous Integration and development using Jenkins

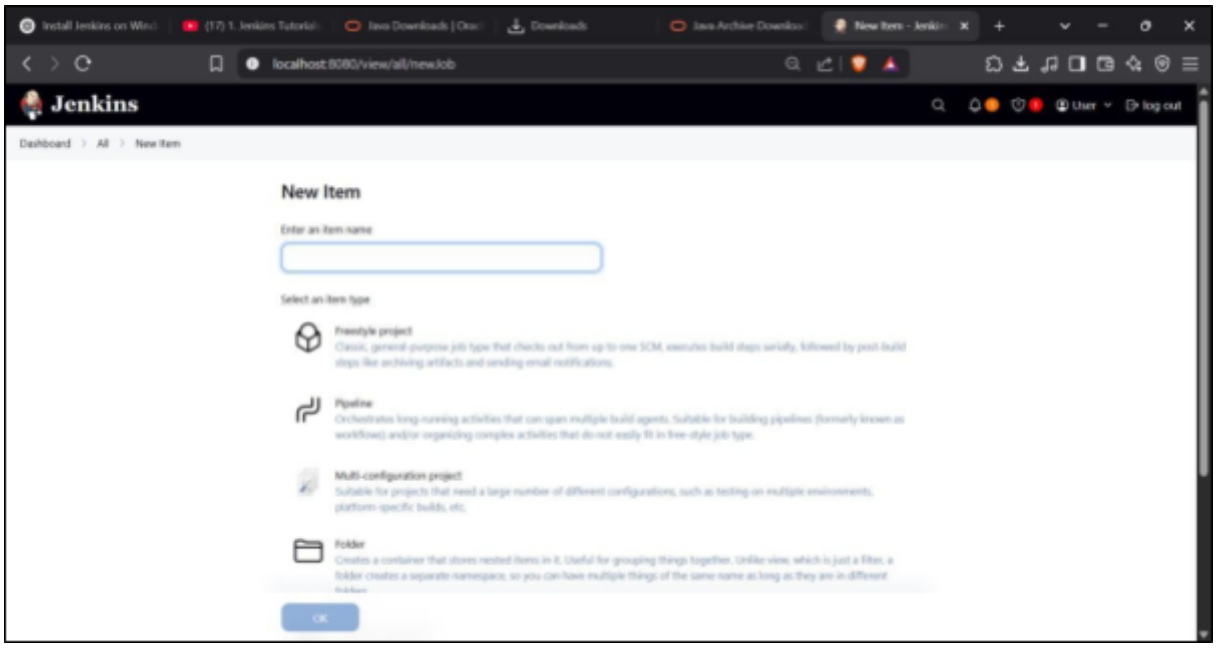
## Steps:

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

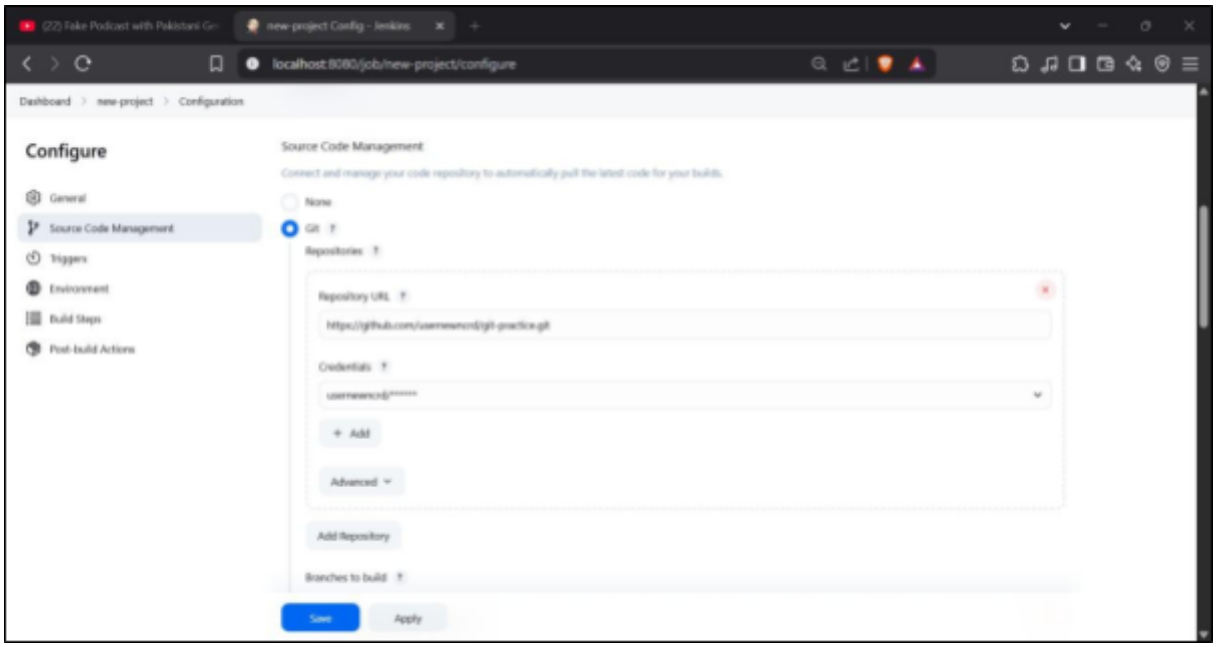


3. Create new Freestyle Project: CI-Demo




4. Under Source Code Management, choose Git and enter your repo URL.



5. Add Build Step > Execute Shell: echo "Building Project..."
   echo "Run tests..."


6. Save and click Build Now.


7. Check output in Console Output.

```
+ echo 'Building Project...'
Building Project...
+ echo 'Run tests...'
Run tests...
Finished: SUCCESS
```

# PRACTICAL NO.9

## Explore docker commands for content management

1. Check Docker
   version docker
   –version

```
ubuntu@ubuntu:~$ docker --version
Docker version 28.1.1, build 4eba377
```

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

```
ubuntu@ubuntu:~$ docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
254e724d7786: Pull complete
913115292750: Pull complete
3e544d53ce49: Pull complete
4f21ed9ac0c0: Pull complete
d38f2ef2d6f2: Pull complete
40a6e9f4e456: Pull complete
d3dc5ec71e9d: Pull complete
Digest: sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
```

3. List all Docker
   images docker
   images

```
ubuntu@ubuntu:~$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
nginx        latest    a830707172e8   4 weeks ago   192MB
```

4. 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).

```
ubuntu@ubuntu:~$ docker run -d -p 8080:80 --name mynginx nginx
c241fdc47993e83fe932231e1ba068b8953126eb87a89916c50ebabdc088254c
```

5. List all running
   containers docker ps

```
ubuntu@ubuntu:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                   NAMES
c241fdc47993   nginx     "/docker-entrypoint.…"   27 seconds ago   Up 26 seconds   0.0.0.0:8080->80/tcp   mynginx
```

6. 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.

```
ubuntu@ubuntu:~$ docker cp index.html mynginx:/usr/share/nginx/html/
lstat /home/ubuntu/index.html: no such file or directory
```

7. Copy content from container to host
   docker cp mynginx:/usr/share/nginx/html/index.html .

```
ubuntu@ubuntu:~$ docker cp index.html mynginx:/usr/share/nginx/html/
lstat /home/ubuntu/index.html: no such file or directory
```

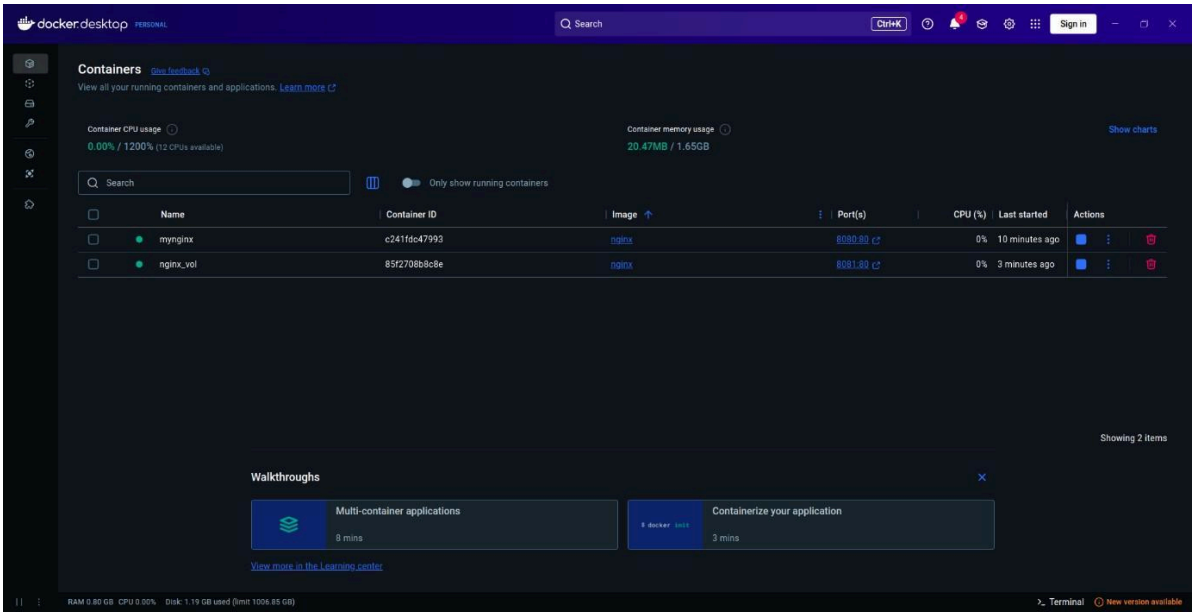
8. Create and use Docker volume for persistent content
   docker volume create mydata
   docker run -d -p 8081:80 --name nginx_vol -v mydata:/usr/share/nginx/html nginx
   Now any data added to the /usr/share/nginx/html inside the container will persist even if the container is removed.

```
ubuntu@ubuntu:~$ docker volume create mydata
mydata
ubuntu@ubuntu:~$ docker run -d -p 8081:80 --name nginx_vol -v mydata:/usr/share/nginx/html nginx
85f2708b8c8ec2c1eba2bb88f10a162feec1faa1ad3f86c2f0e8d0ba32e1090a
```

9. List Docker
   volumes docker
   volume ls

```
ubuntu@ubuntu:~$ docker volume ls
DRIVER    VOLUME NAME
local     mydata
```

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

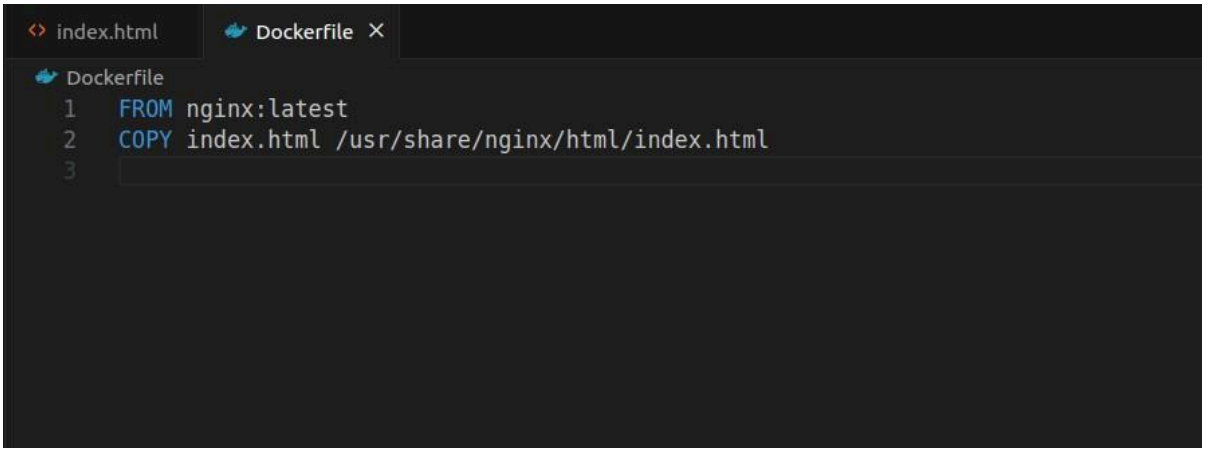


# PRACTICAL NO.10

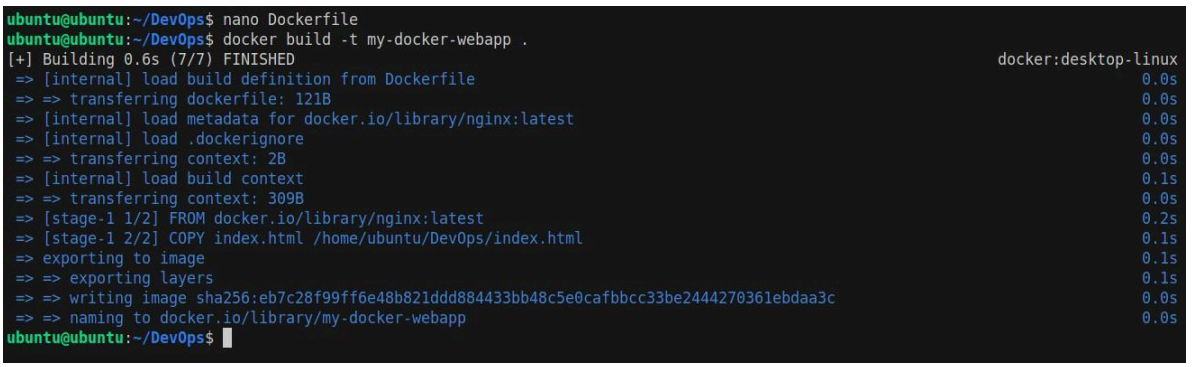
## Develop a simple containerized application using Docker

1. Index.html


2. DockerfIle :-




3. docker build -t my-docker-webapp .



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

```
ubuntu@ubuntu:~/DevOps$ docker run -d -p 8080:80 --name webapp-container my-docker-webapp
87758d2c13e4eb227c0bb149148952a661a46b92867ef336a4dd2ad74a993e3f
ubuntu@ubuntu:~/DevOps$
```

5. docker ps

```
ubuntu@ubuntu:~/DevOps$ docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS          PORTS                   NAMES
87758d2c13e4   my-docker-webapp   "/docker-entrypoint.…"   38 seconds ago   Up 37 seconds   0.0.0.0:8080->80/tcp    webapp-container
85f2708b8c8e   nginx              "/docker-entrypoint.…"   18 minutes ago   Up 18 minutes   0.0.0.0:8081->80/tcp    nginx_vol
```

6. docker stop webapp-container

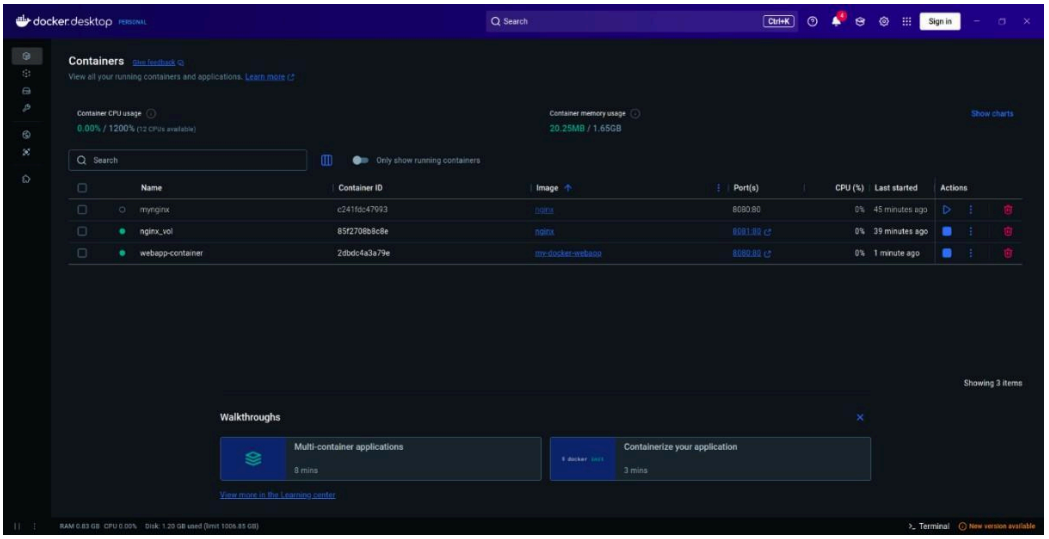
```
ubuntu@ubuntu:~/DevOps$ docker stop webapp-container
webapp-container
```

7. docker rm webapp-container

```
ubuntu@ubuntu:~/DevOps$ docker rm webapp-container
webapp-container
```

8. docker rmi my-docker-webapp

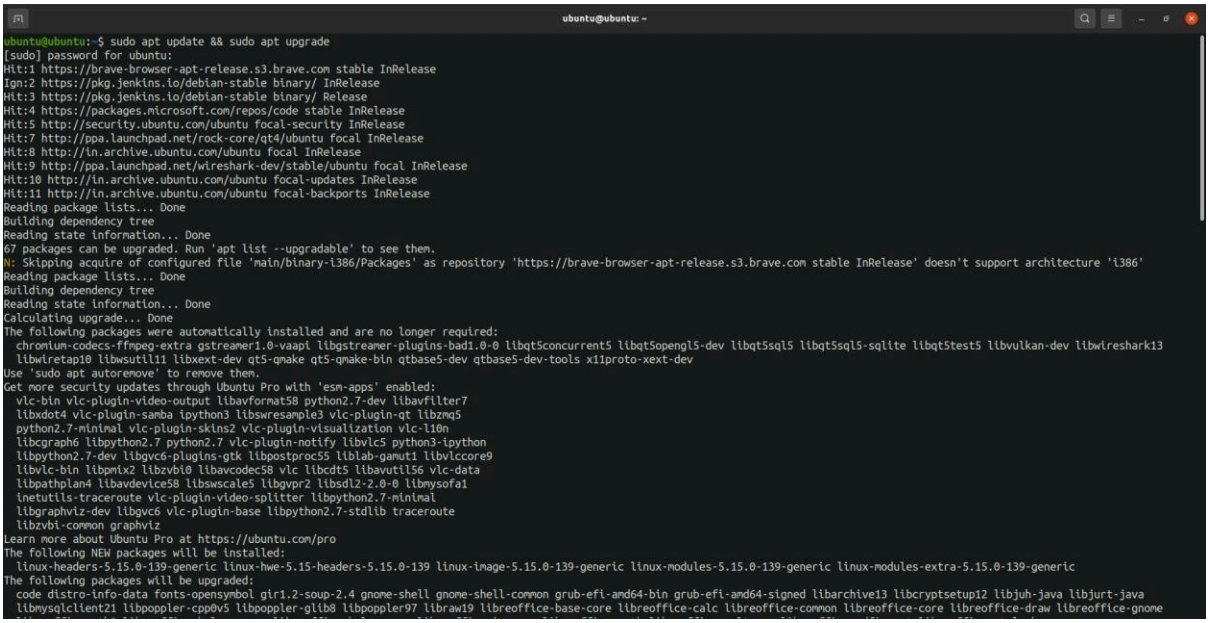
```
ubuntu@ubuntu:~/DevOps$ docker rmi my-docker-webapp
Untagged: my-docker-webapp:latest
Deleted: sha256:eb7c28f99ff6e48b821ddd884433bb48c5e0cafbbcc33be2444270361ebdaa3c
```

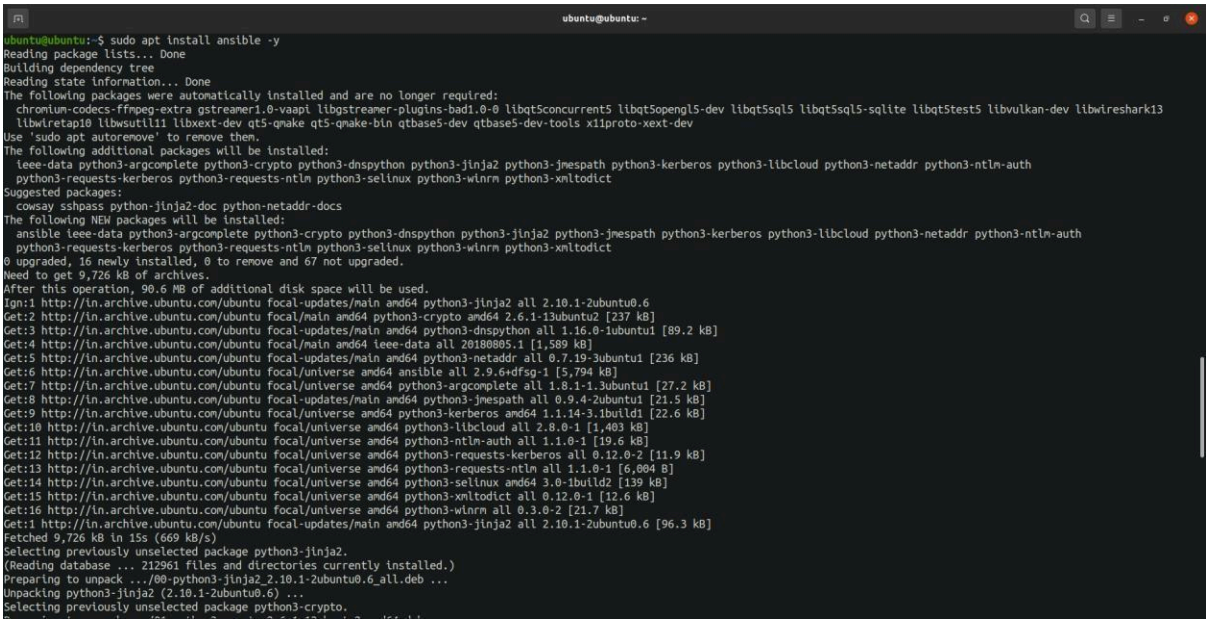
# PRACTICAL NO.11

## Ad-hoc Ansible Commands

Step 1: Update your VM




Step 2: Install Ansible



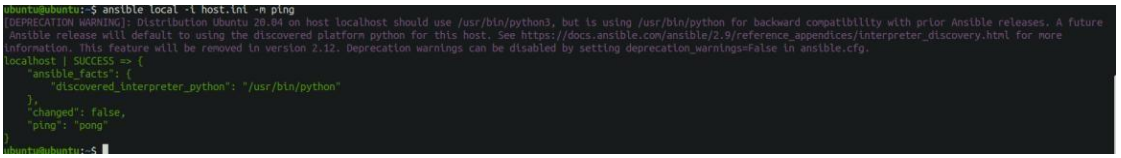

Step 3: Check version:

```
ubuntu@ubuntu:~$ ansible --version
ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.10 (default, Mar 18 2025, 20:04:55) [GCC 9.4.0]
ubuntu@ubuntu:~$
```

```
ubuntu@ubuntu:~$ nano host.ini
ubuntu@ubuntu:~$
```

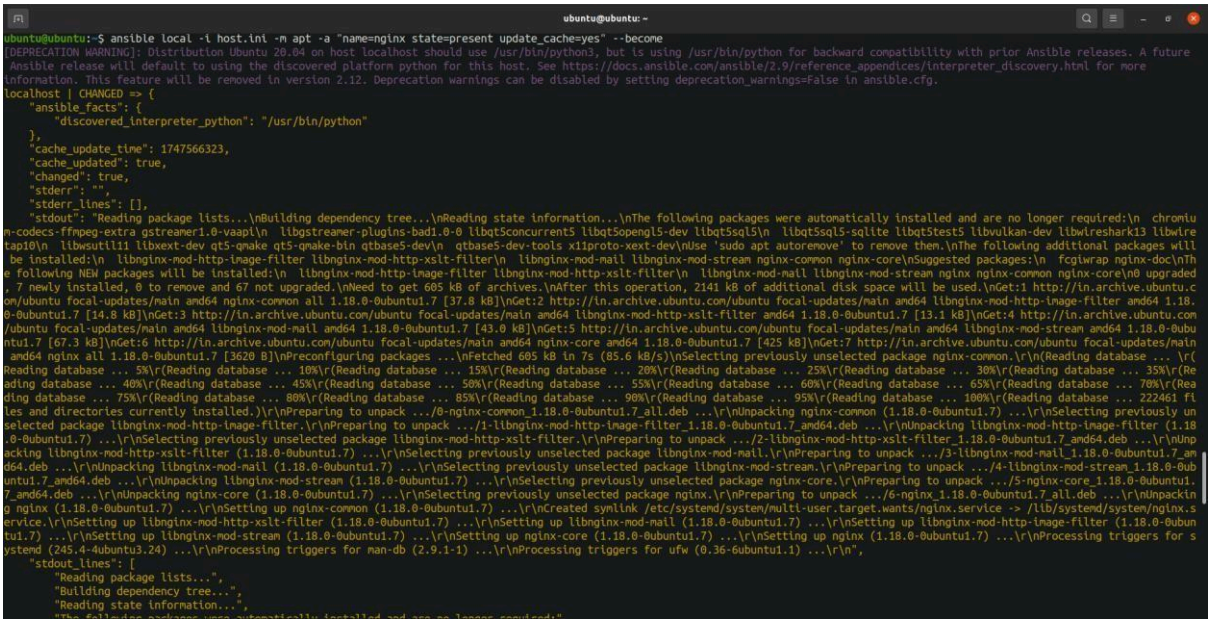
```
  GNU nano 4.8
localhost ansible_connection=local
```

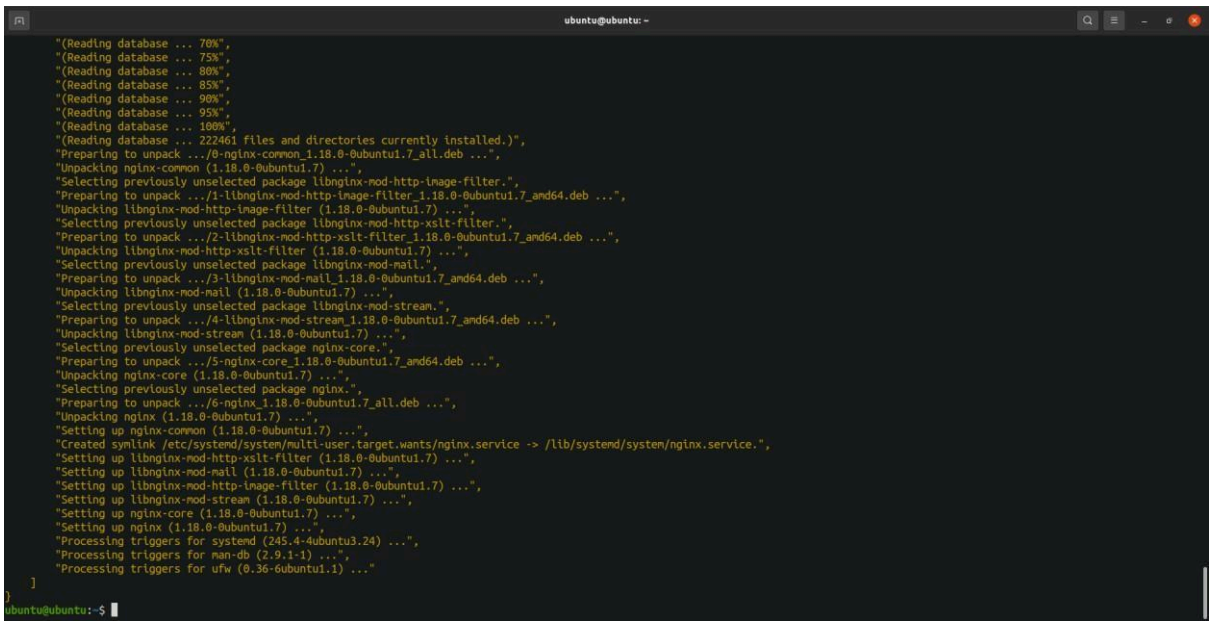
1. Ping the remote host
   ansible local -i host.ini -m ping



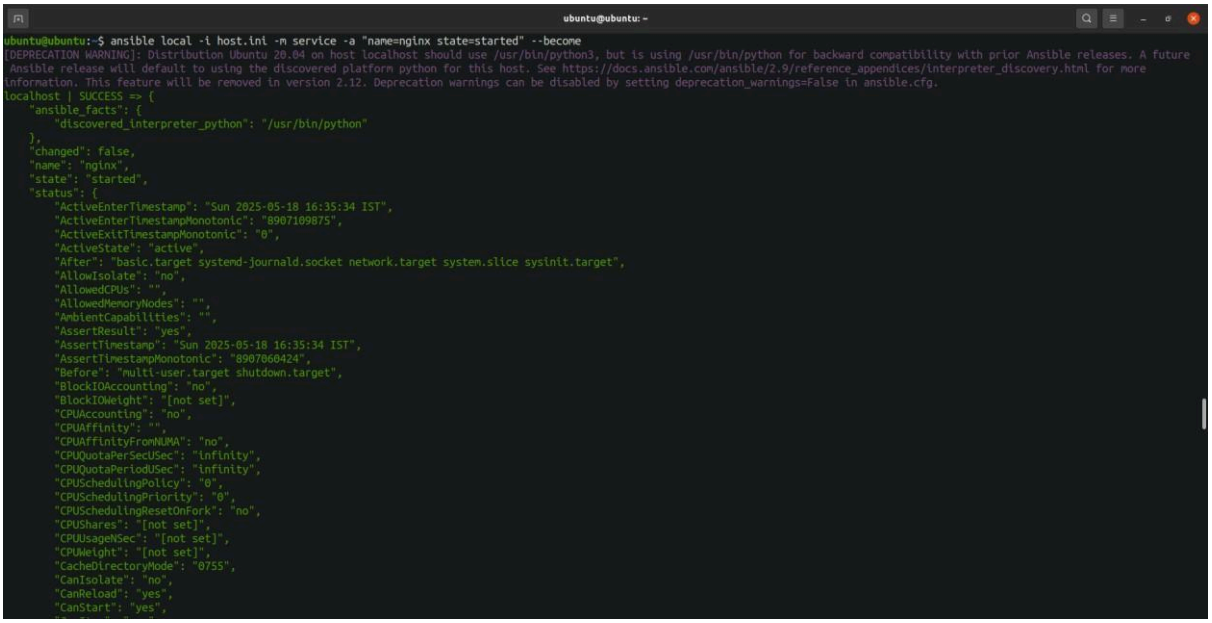

2. Check uptime
   ansible local -i host.ini -a "uptime"


3. Install a package
   ansible local -i host.ini -m apt -a "name=nginx state=present update_cache=yes" –become






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




# PRACTICAL NO.12

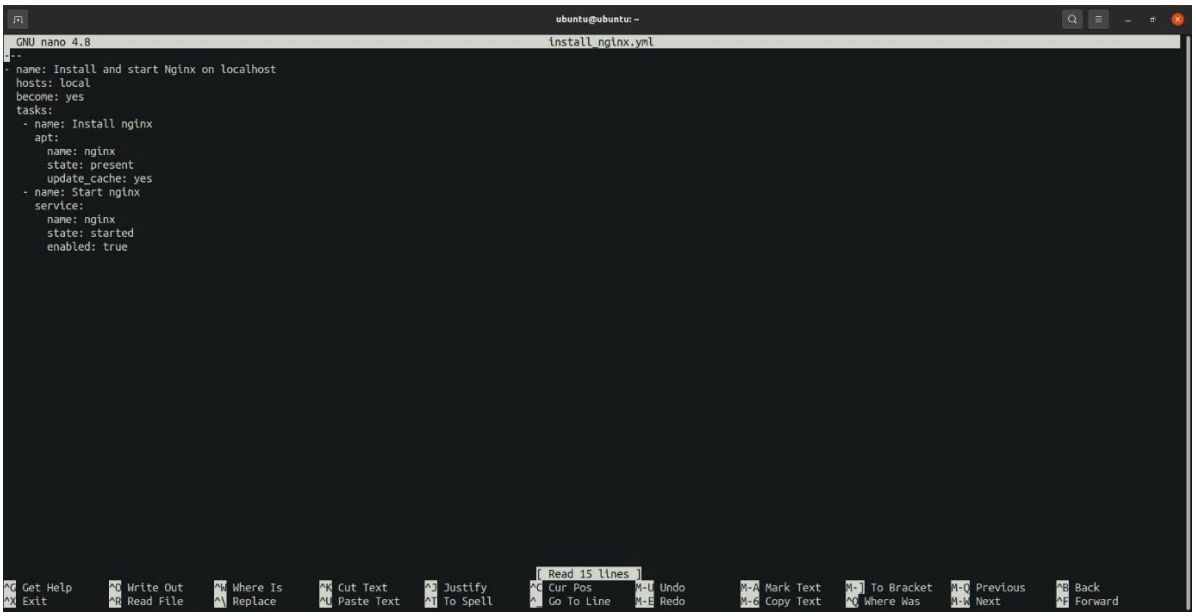
## Using Ansible Playbooks

### 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
```

```
ubuntu@ubuntu:~$ nano install_nginx.yml
```



Run the Playbook:
ansible-playbook -i hosts.ini install_nginx.yml

```
ubuntu@ubuntu:~$ ansible-playbook -i host.ini install_nginx.yml

PLAY [Install and start Nginx on localhost] ************************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
[DEPRECATION WARNING]: Distribution Ubuntu 20.04 on host localhost should use /usr/bin/python3, but is using /usr/bin/python for backward compatibility with prior Ansible releases. A future
 Ansible release will default to using the discovered platform python for this host. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more
information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
ok: [localhost]

TASK [Install nginx] ***********************************************************************************************************************************************************************************
ok: [localhost]

TASK [Start nginx] *************************************************************************************************************************************************************************************
ok: [localhost]

PLAY RECAP *********************************************************************************************************************************************************************************************
localhost                  : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```