

---

# Jenkins CI/CD Pipeline for Dockerized Python App on AWS

Project by  
**R. Bhavani**

# Table of Contents

| S.No | Content                               | Pg. No |
|------|---------------------------------------|--------|
| 1.   | Project Overview .....                | 3      |
| 2.   | Architecture Diagram .....            | 4      |
| 3.   | WSL Ubuntu Terminal Integration ..... | 5      |
| 4.   | GitHub Repository .....               | 6      |
| 5.   | EC2 Instance .....                    | 7      |
| 6.   | Jenkins Setup .....                   | 8      |
| 7.   | Pipeline Run .....                    | 9-10   |
| 8.   | Docker .....                          | 11     |
| 9.   | Git .....                             | 12     |
| 10.  | Output results .....                  | 13     |

# 1. Project Overview

---

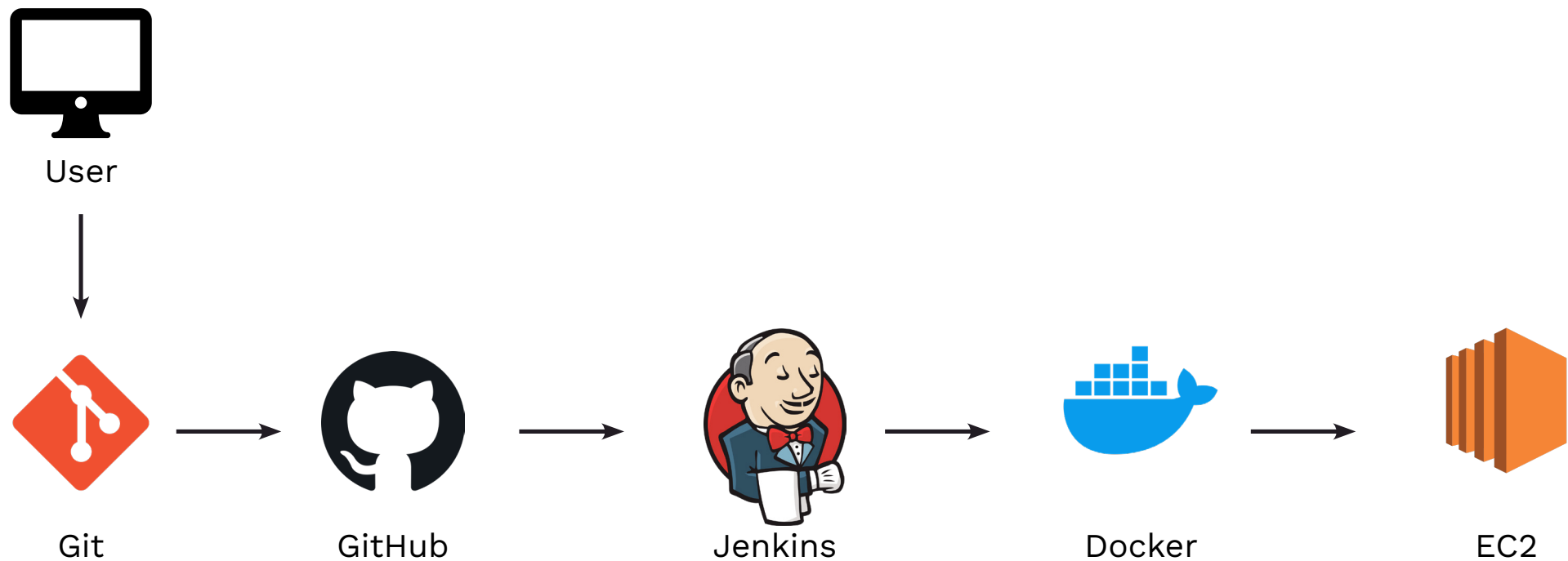
This project contains a complete walkthrough of the CI/CD pipeline project, along with placeholders for screenshots to demonstrate each step.

- **WSL Ubuntu Terminal Integration**
- **GitHub Repository**
- **EC2 Instance**
- **Jenkins Setup**
- **Pipeline Run**
- **Docker**
- **Python Flask Web Application**

Below are the steps, configuration details, and screenshots demonstrating the working application.

## 2. Architecture Diagram

---



# 3. WSL Ubuntu Terminal Integration

---

- Opened and configured Ubuntu WSL Terminal.
- Fetched all necessary files:

Ubuntu WSL → Git → Github

```
dmin@DESKTOP-6BDGARK:~$ mkdir temp
dmin@DESKTOP-6BDGARK:~$ cd temp
dmin@DESKTOP-6BDGARK:~/temp$ nano app.py
dmin@DESKTOP-6BDGARK:~/temp$ nano requirements.txt
dmin@DESKTOP-6BDGARK:~/temp$ nano Dockerfile
dmin@DESKTOP-6BDGARK:~/temp$ nano .dockerignore
dmin@DESKTOP-6BDGARK:~/temp$ nano Jenkinsfile
dmin@DESKTOP-6BDGARK:~/temp$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/dmin/temp/.git/
dmin@DESKTOP-6BDGARK:~/temp$ git add .
dmin@DESKTOP-6BDGARK:~/temp$ git commit -m "Initial Commitment"
[master (root-commit) e4681cb] Initial Commitment
 5 files changed, 60 insertions(+)
 create mode 100644 .dockerignore
 create mode 100644 Dockerfile
 create mode 100644 Jenkinsfile
 create mode 100644 app.py
 create mode 100644 requirements.txt
dmin@DESKTOP-6BDGARK:~/temp$ git remote add origin https://github.com/bhavani2909/jenkinspythonapp.git
dmin@DESKTOP-6BDGARK:~/temp$ git branch -M main
dmin@DESKTOP-6BDGARK:~/temp$ git push -u origin main
```

*Image 1 : Pushing all necessary files using the Ubuntu WSL → Git → Github*

# 4. GitHub Repository

---

Accessed GitHub via PAT Tokens generated to fetch all the files from the remote repository.

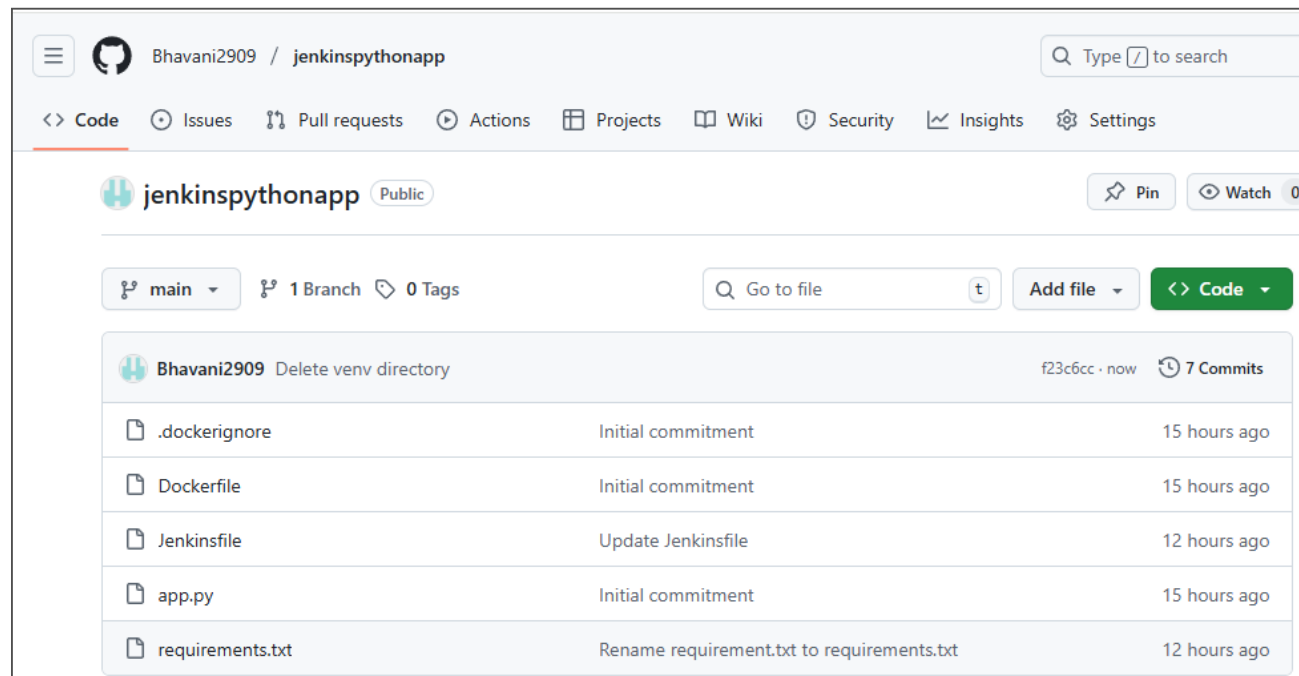


Image 2 : Repository page showing Jenkinsfile, Dockerfile, app.py, requirements.txt.

# 5. EC2 Instance

- Launched Amazon Linux EC2 and upgraded installation.
- Installed Java 17 for Jenkins setup.

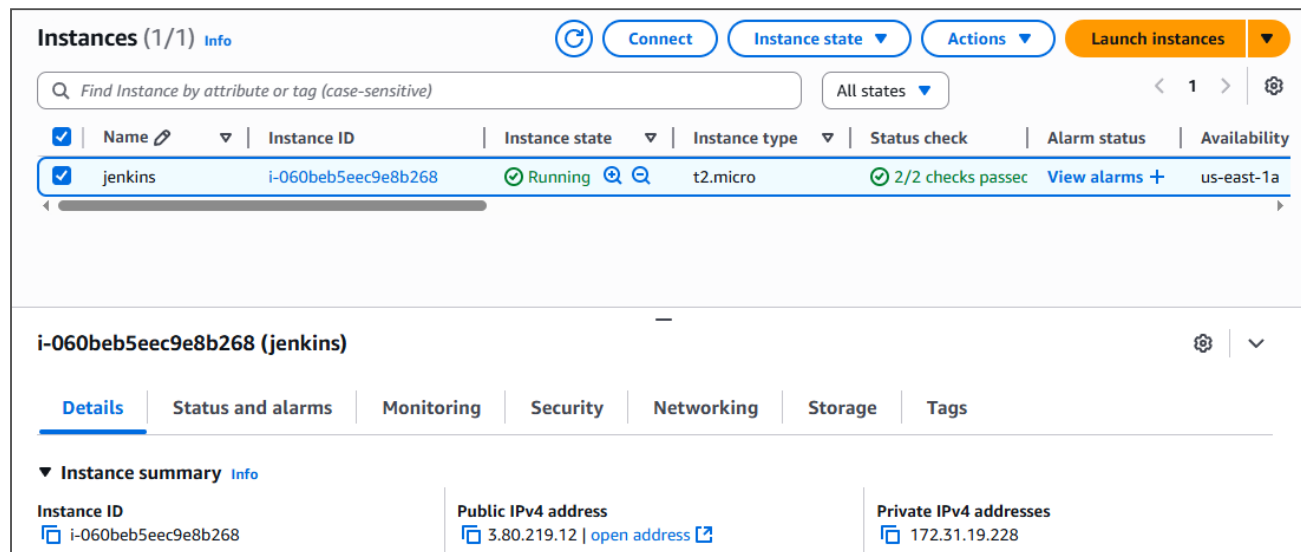


Image 3 : AWS EC2 Dashboard showing running instance.

# 6. Jenkins Setup

- Jenkins is installed after the configuration of Java.

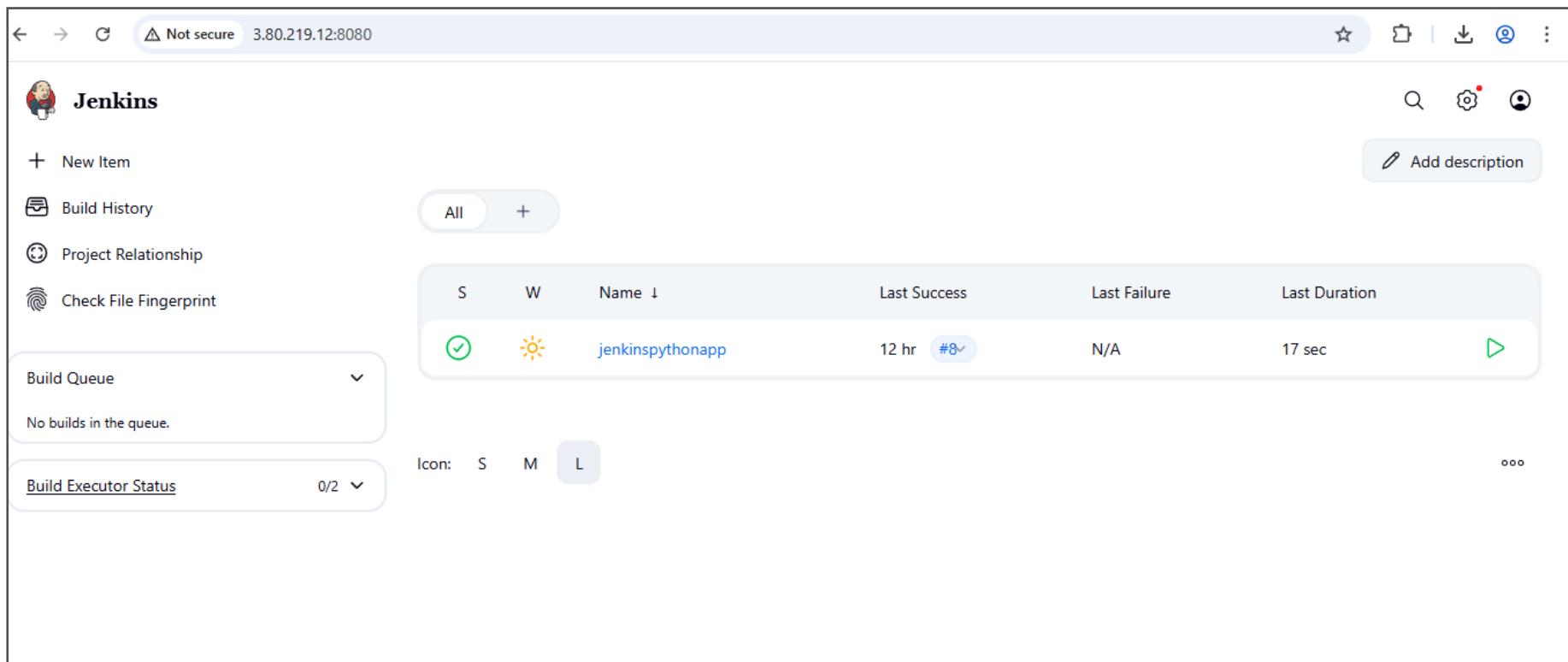


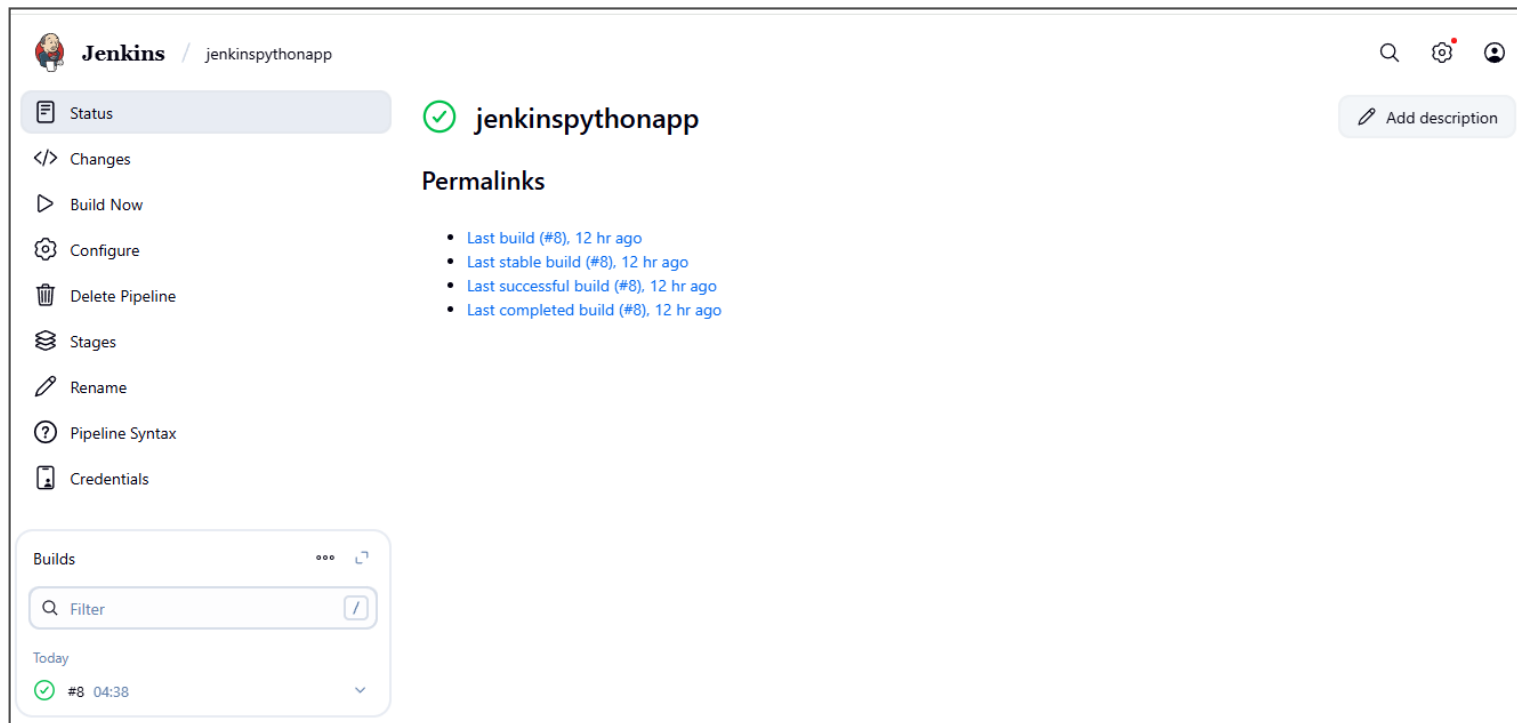
Image 4 : Jenkins job configuration with repository URL.



# 7. Pipeline Run

---

- Added a “New item” and scheduled it with the necessary node configurations.
- Ensured the built-in node is online.
- Verified the success of the console output stages : checkout → build → deploy.



*Image 5 : Jenkins console output showing successful stages (checkout → build → deploy)*

 **Jenkins** / jenkinspythonapp / #8

Status

Changes

Console Output

Edit Build Information

Delete build '#8'

Timings

Git Build Data

Pipeline Overview

Restart from Stage

Replay

Pipeline Steps

Workspaces

✓ Console Output

Download

Copy

View as plain text

```
Started by user Bhavani
Obtained Jenkinsfile from git https://github.com/Bhavani2909/jenkinspythonapp
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/jenkinspythonapp
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
The recommended git tool is: git
using credential J
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/jenkinspythonapp/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/Bhavani2909/jenkinspythonapp # timeout=10
Fetching upstream changes from https://github.com/Bhavani2909/jenkinspythonapp
> git --version # timeout=10
> git --version # 'git version 2.50.1'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/Bhavani2909/jenkinspythonapp
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
```

Image 6 :Jenkins job success page with green checkmark.

## 8. Docker

- Installed and verified docker images via AWS EC2.
- The containerized and dockerized image ID is seen.

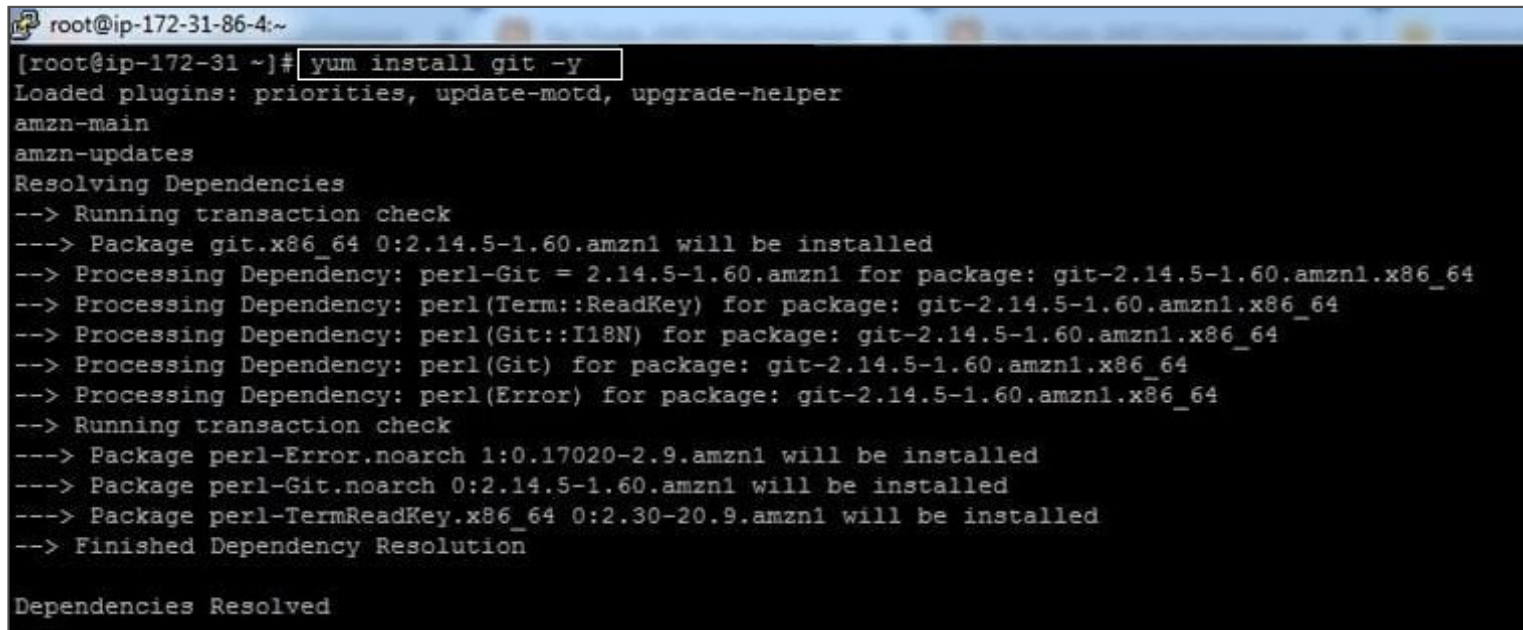
[illegible]

Image 7 : Terminal showing `docker images` & `docker ps` with container running.

# 9. Git

---

- Ensured that the Git is installed and configured in EC2 Terminal.



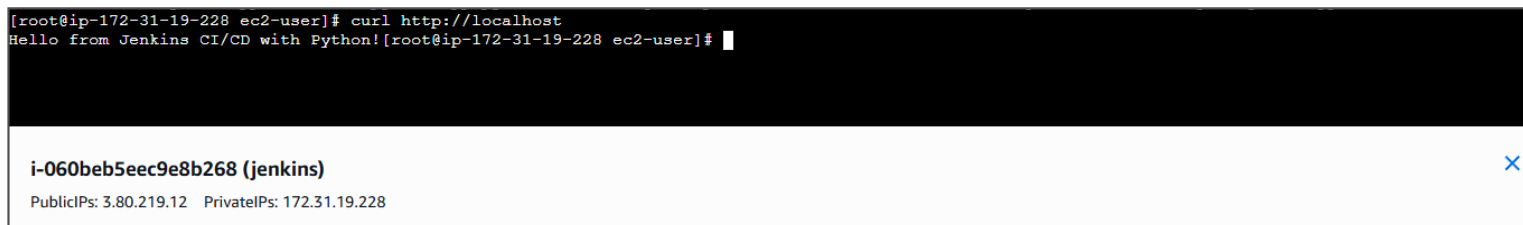
```
root@ip-172-31-86-4:~  
[root@ip-172-31 ~]# yum install git -y  
Loaded plugins: priorities, update-motd, upgrade-helper  
amzn-main  
amzn-updates  
Resolving Dependencies  
--> Running transaction check  
---> Package git.x86_64 0:2.14.5-1.60.amzn1 will be installed  
--> Processing Dependency: perl-Git = 2.14.5-1.60.amzn1 for package: git-2.14.5-1.60.amzn1.x86_64  
--> Processing Dependency: perl(Term::ReadKey) for package: git-2.14.5-1.60.amzn1.x86_64  
--> Processing Dependency: perl(Git::I18N) for package: git-2.14.5-1.60.amzn1.x86_64  
--> Processing Dependency: perl(Git) for package: git-2.14.5-1.60.amzn1.x86_64  
--> Processing Dependency: perl(Error) for package: git-2.14.5-1.60.amzn1.x86_64  
--> Running transaction check  
---> Package perl-Error.noarch 1:0.17020-2.9.amzn1 will be installed  
---> Package perl-Git.noarch 0:2.14.5-1.60.amzn1 will be installed  
---> Package perl-TermReadKey.x86_64 0:2.30-20.9.amzn1 will be installed  
--> Finished Dependency Resolution  
  
Dependencies Resolved
```

*Image 8 : Git Installation.*

# 10. Output Results

---

- Verified the execution with “Curl” via EC2 Terminal.
- The Python application is seen to run on the browser showing Public IP.



*Image 9 : Terminal with `curl http://localhost` showing app response.*



*Image 10 : Browser showing EC2 Public IP with app running (Hello message).*

**Thank you!**