# 📘 APPLICATION DEPLOYMENT & CI/CD AUTOMATION REPORT

---

**1. Project Title**

**Deployment of React E-commerce Application using Docker, Jenkins, and AWS**

---

**2. Candidate Details**

- **Name:** Junaid Ahamed
- **Project Type:** DevOps – Application Deployment
- **Organization:** GUVI Geeks Network Pvt Ltd

---

**3. Project Objective**

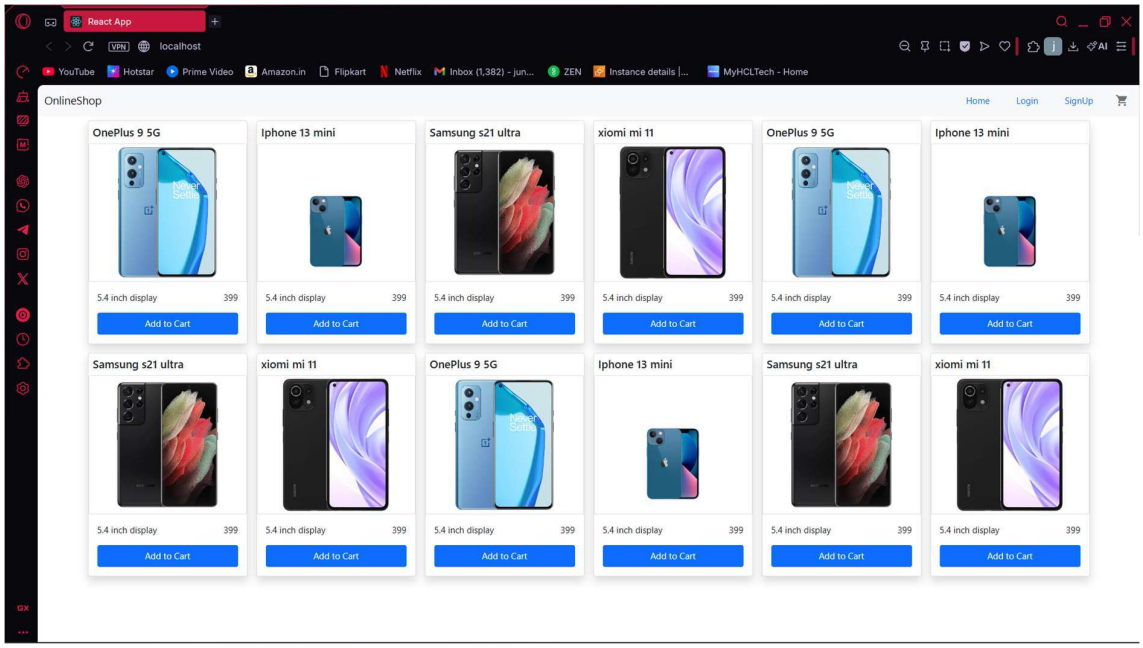
The objective of this project is to deploy a production-ready React application using DevOps best practices. The project includes containerization using Docker, CI/CD automation using Jenkins, cloud deployment on AWS EC2, Docker image management using Docker Hub, version control using GitHub CLI, and application health monitoring.

---

**4. Application Details**

- **Application Type:** ReactJS (Static Build – E-commerce Application)
- **Source Repository:**
- https://github.com/sriram-R-krishnan/devops-build
- **Deployment Port:**
- HTTP – Port 80

The repository contains pre-built React static files which are directly served using Nginx for production readiness.




---

## 5. Docker Implementation

### 5.1 Docker Installation & Environment Setup

Docker and Docker Compose were installed inside Ubuntu (WSL and EC2). The environment was verified using Docker version commands.

📸 **Screenshot:** Docker & Docker-Compose Installation Verification
*(Insert terminal screenshot showing docker --version and docker-compose --version)*

---

### 5.2 Dockerfile Creation

A Dockerfile was created using the nginx:alpine base image to serve the React build files.

**Key Points:**

- Lightweight Nginx image
- Static files copied to /usr/share/nginx/html
- Port 80 exposed

```
junaid@LAPTOP-GU5B805P:~$ git clone https://github.com/sriram-R-krishnan/devops-build.git
Cloning into 'devops-build'...
remote: Enumerating objects: 21, done.
remote: Total 21 (delta 0), reused 0 (delta 0), pack-reused 21 (from 1)
Receiving objects: 100% (21/21), 720.09 KiB | 7.50 MiB/s, done.
junaid@LAPTOP-GU5B805P:~$ cd devops-build
junaid@LAPTOP-GU5B805P:~/devops-build$ pwd
/home/junaid/devops-build
junaid@LAPTOP-GU5B805P:~/devops-build$ ls
build
junaid@LAPTOP-GU5B805P:~/devops-build$ nano Dockerfile
junaid@LAPTOP-GU5B805P:~/devops-build$ rm -f Dockerfile
junaid@LAPTOP-GU5B805P:~/devops-build$ nano Dockerfile
junaid@LAPTOP-GU5B805P:~/devops-build$ rm -f Dockerfile
junaid@LAPTOP-GU5B805P:~/devops-build$ cat <<EOF > Dockerfile
FROM nginx:alpine

RUN rm -rf /usr/share/nginx/html/*

COPY build /usr/share/nginx/html

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
EOF
junaid@LAPTOP-GU5B805P:~/devops-build$ cat Dockerfile
FROM nginx:alpine

RUN rm -rf /usr/share/nginx/html/*

COPY build /usr/share/nginx/html

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
junaid@LAPTOP-GU5B805P:~/devops-build$ ^[[200~docker build -t react-app .
^[[201~docker: command not found
junaid@LAPTOP-GU5B805P:~/devops-build$ docker build -t react-app .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.419MB
Step 1/5 : FROM nginx:alpine
alpine: Pulling from library/nginx
1074353eec0d: Pulling fs layer
25f453064fd3: Pulling fs layer
567f84da6fbd: Pulling fs layer
da7c973d8b92: Pulling fs layer
33f95a0f3229: Pulling fs layer
```

---

### 5.3 Docker Image Build

The Docker image was built successfully using the Dockerfile.

```
junaid@LAPTOP-GU5B805P:~/devops-build$ cat <<EOF > build.sh
#!/bin/bash

echo "Building Docker image..."
docker build -t react-app .
EOF
junaid@LAPTOP-GU5B805P:~/devops-build$ chmod +x build.sh
junaid@LAPTOP-GU5B805P:~/devops-build$ ./build.sh
Building Docker image...
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.421MB
Step 1/5 : FROM nginx:alpine
 ---> 04da2b0513cd
Step 2/5 : RUN rm -rf /usr/share/nginx/html/*
 ---> Using cache
 ---> 29926da7f24e
Step 3/5 : COPY build /usr/share/nginx/html
 ---> Using cache
 ---> 2a38f7bf7f85
Step 4/5 : EXPOSE 80
 ---> Using cache
 ---> 8b2cc724ad0f
Step 5/5 : CMD ["nginx", "-g", "daemon off;"]
 ---> Using cache
 ---> 7a7ef912cf1f
Successfully built 7a7ef912cf1f
Successfully tagged react-app:latest
junaid@LAPTOP-GU5B805P:~/devops-build$ cat <<EOF > deploy.sh
#!/bin/bash

echo "Stopping old container..."
docker stop react-app || true
docker rm react-app || true

echo "Starting new container..."
docker run -d -p 80:80 --name react-app react-app
EOF
junaid@LAPTOP-GU5B805P:~/devops-build$ chmod +x deploy.sh
junaid@LAPTOP-GU5B805P:~/devops-build$ ./deploy.sh
Stopping old container...
Error response from daemon: No such container: react-app
Error response from daemon: No such container: react-app
Starting new container...
dab4421a61733ca2c6a012ff7ff78880f425414f46ae89aca16f838a18c9db06
docker: Error response from daemon: failed to set up container networking: driver failed programming external connectivity on endpoint
 react-app (018e6778814fc3eaa1244b862354ab6aab53f94988630ceaab55371e29720b9a): Bind for 0.0.0.0:80 failed: port is already allocated
```

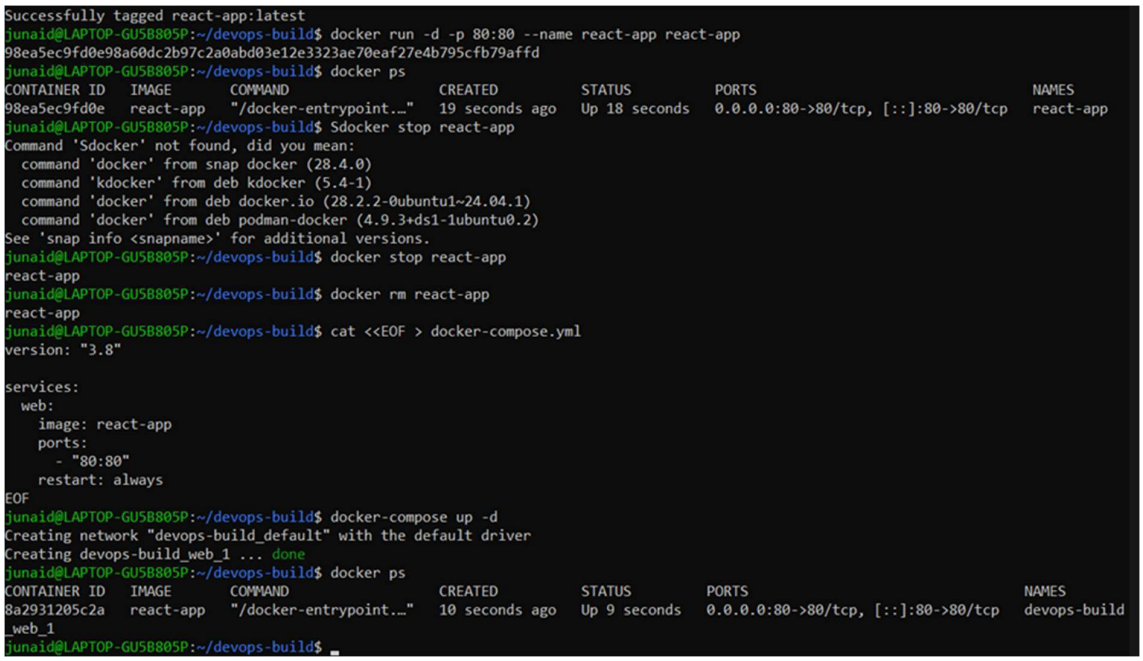
---

### 6. Docker Compose Implementation

A docker-compose.yml file was created to manage container lifecycle.

**Features:**

- Application runs on port 80
- Restart policy enabled
- Simplified deployment




---

### 7. Bash Scripting

Two Bash scripts were created for automation.

#### 7.1 build.sh

- Automates Docker image build

#### 7.2 deploy.sh

- Stops old containers
- Deploys application using Docker Compose

```
junaid@LAPTOP-GU5B805P:~/devops-build$ cat <<EOF > build.sh
#!/bin/bash

echo "Building Docker image..."
docker build -t react-app .
EOF
junaid@LAPTOP-GU5B805P:~/devops-build$ chmod +x build.sh
junaid@LAPTOP-GU5B805P:~/devops-build$ ./build.sh
Building Docker image...
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.421MB
Step 1/5 : FROM nginx:alpine
 ---> 04da2b0513cd
Step 2/5 : RUN rm -rf /usr/share/nginx/html/*
 ---> Using cache
 ---> 29926da7f24e
Step 3/5 : COPY build /usr/share/nginx/html
 ---> Using cache
 ---> 2a38f7bf7f85
Step 4/5 : EXPOSE 80
 ---> Using cache
 ---> 8b2cc724ad0f
Step 5/5 : CMD ["nginx", "-g", "daemon off;"]
 ---> Using cache
 ---> 7a7ef912cf1f
Successfully built 7a7ef912cf1f
Successfully tagged react-app:latest
junaid@LAPTOP-GU5B805P:~/devops-build$ cat <<EOF > deploy.sh
#!/bin/bash

echo "Stopping old container..."
docker stop react-app || true
docker rm react-app || true

echo "Starting new container..."
docker run -d -p 80:80 --name react-app react-app
EOF
junaid@LAPTOP-GU5B805P:~/devops-build$ chmod +x deploy.sh
junaid@LAPTOP-GU5B805P:~/devops-build$ ./deploy.sh
Stopping old container...
Error response from daemon: No such container: react-app
Error response from daemon: No such container: react-app
Starting new container...
dab4421a61733ca2c6a012ff7ff78880f425414f46ae89aca16f838a18c9db06
docker: Error response from daemon: failed to set up container networking: driver failed programming external connectivity on endpoint
 react-app (018e6778814fc3eaa1244b862354ab6aab53f94988630ceaab55371e29720b9a): Bind for 0.0.0.0:80 failed: port is already allocated
```

---

### 8. Version Control (GitHub – CLI Only)

Git operations were performed strictly using CLI.

**Steps Performed:**

- .gitignore and .dockerignore created
- dev branch created
- Code committed and pushed using CLI
- SSH authentication configured

```
junaid@LAPTOP-GU5B805P:~/devops-build$ git status
On branch dev
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   .gitignore
        modified:   docker-compose.yml

junaid@LAPTOP-GU5B805P:~/devops-build$ git commit -m "Dockerized app with compose, scripts and Jenkins p
ipeline"
[dev 04973cb] Dockerized app with compose, scripts and Jenkins pipeline
 2 files changed, 4 insertions(+), 3 deletions(-)
junaid@LAPTOP-GU5B805P:~/devops-build$ git push -u origin dev
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 445 bytes | 445.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:Junaid-dot-max/devops-build.git
   0507b38..04973cb  dev -> dev
branch 'dev' set up to track 'origin/dev'.
junaid@LAPTOP-GU5B805P:~/devops-build$ git status
On branch dev
Your branch is up to date with 'origin/dev'.

nothing to commit, working tree clean
```

**GitHub Repository URL:**

https://github.com/Junaid-dot-max/devops-build

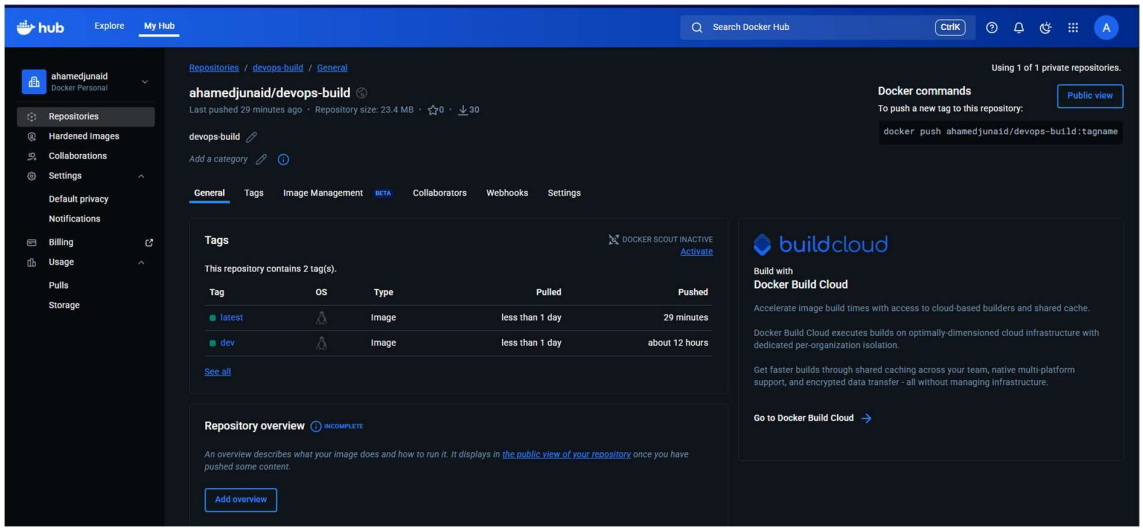
---

### 9. Docker Hub Integration

Two Docker Hub repositories were created as per requirements:

| Environment | Repository | Visibility |
|---|---|---|
| DEV | ahamedjunaid/devops-build | Public |
| PROD | ahamedjunaid/devops-build-prod | Private |

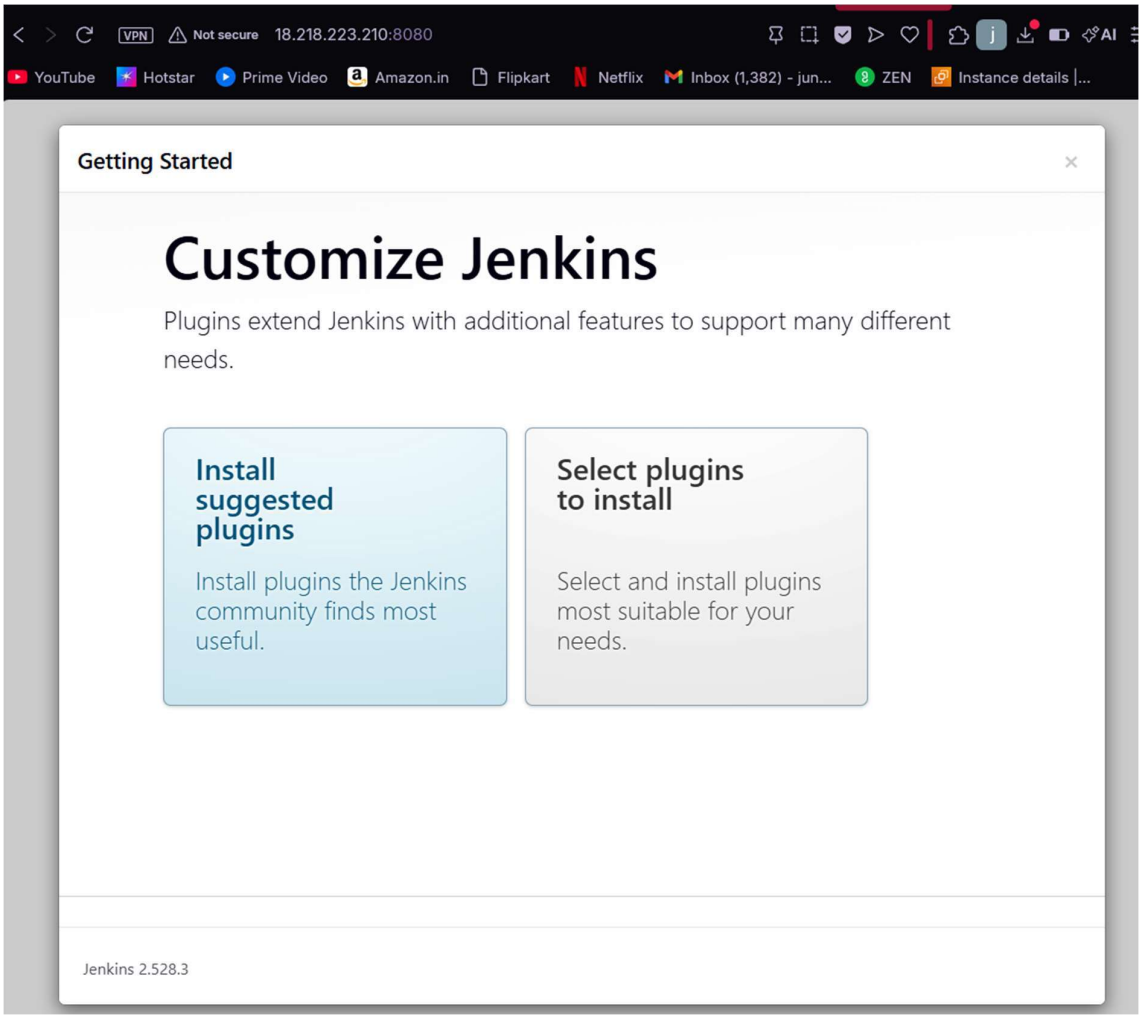
Docker images were tagged and pushed successfully.



## 10. Jenkins CI/CD Pipeline

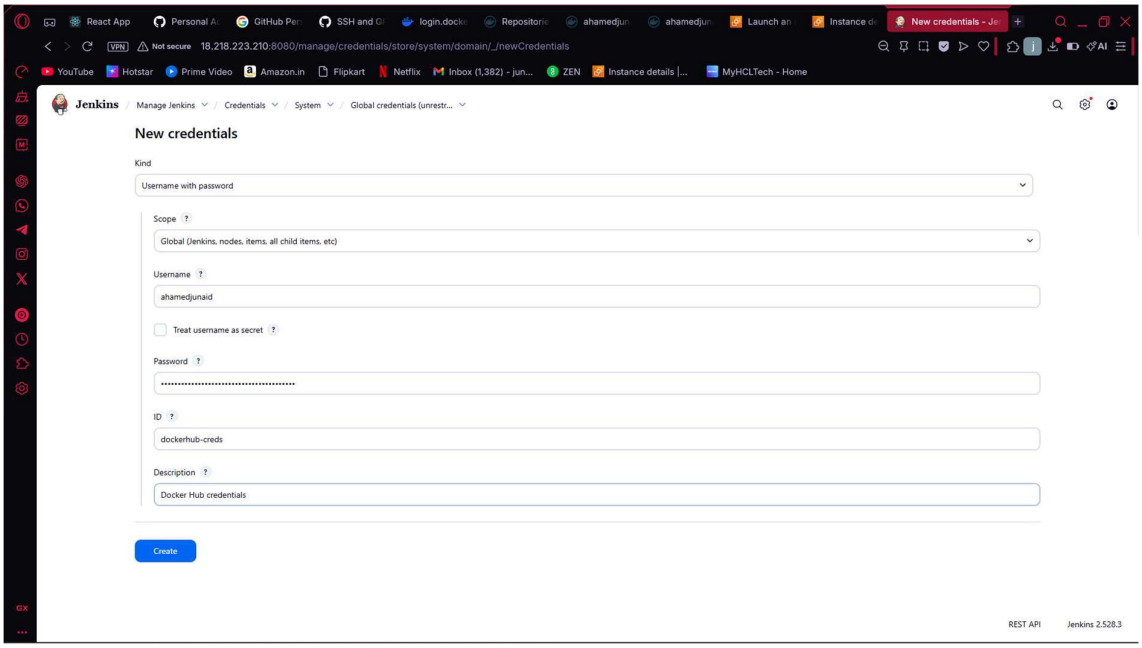
### 10.1 Jenkins Installation

Jenkins was installed on AWS EC2 and accessed via port 8080.




### 10.2 Jenkins Credentials Configuration

- Docker Hub credentials added
- GitHub SSH key configured
- Secure authentication enabled

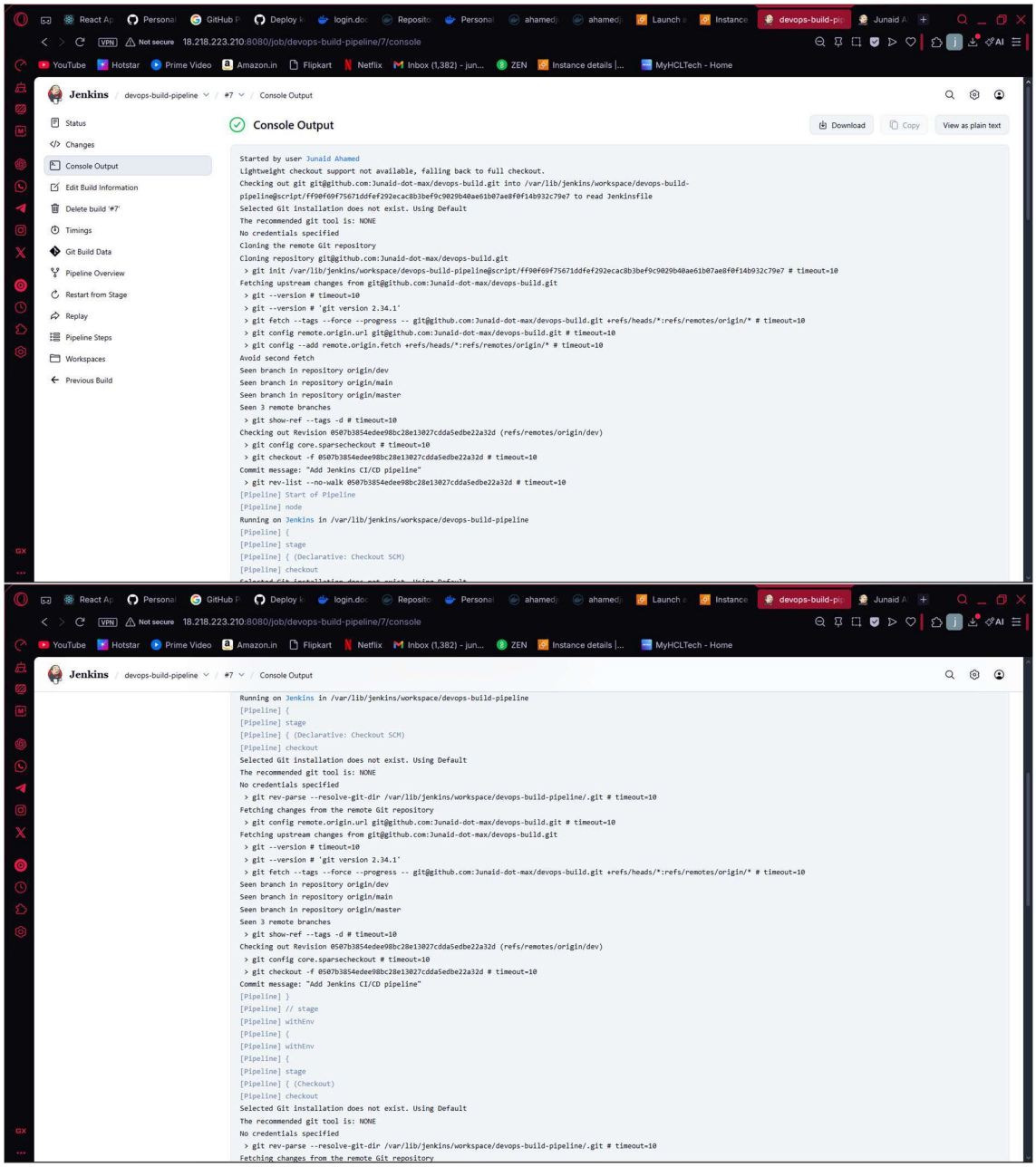



---

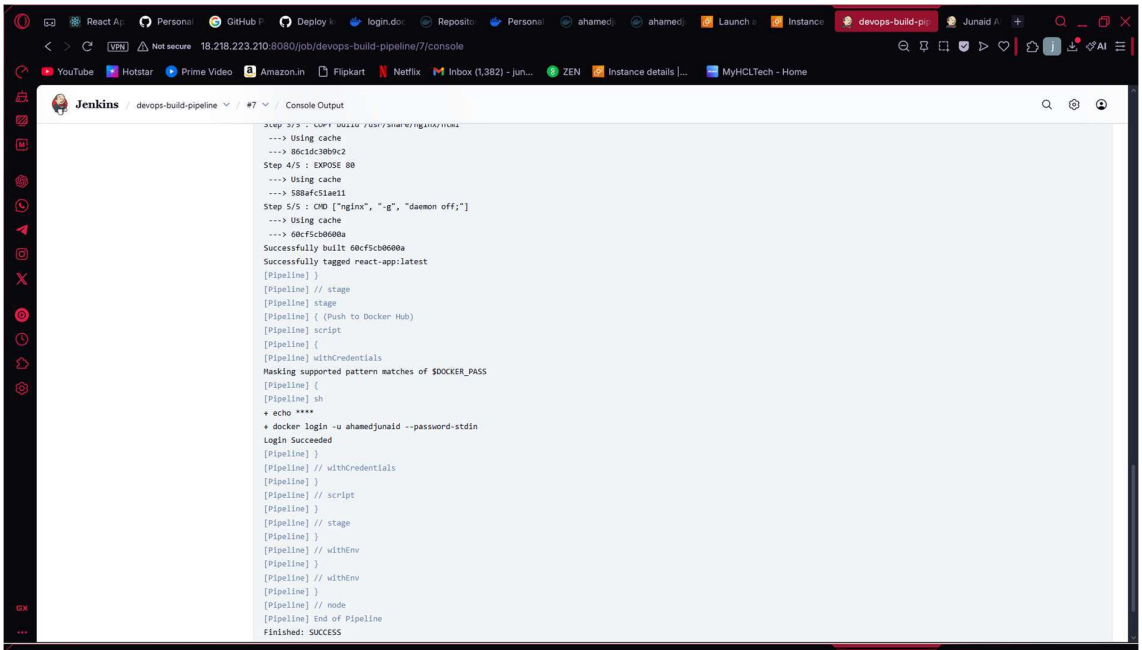
### 10.3 Jenkins Pipeline Execution

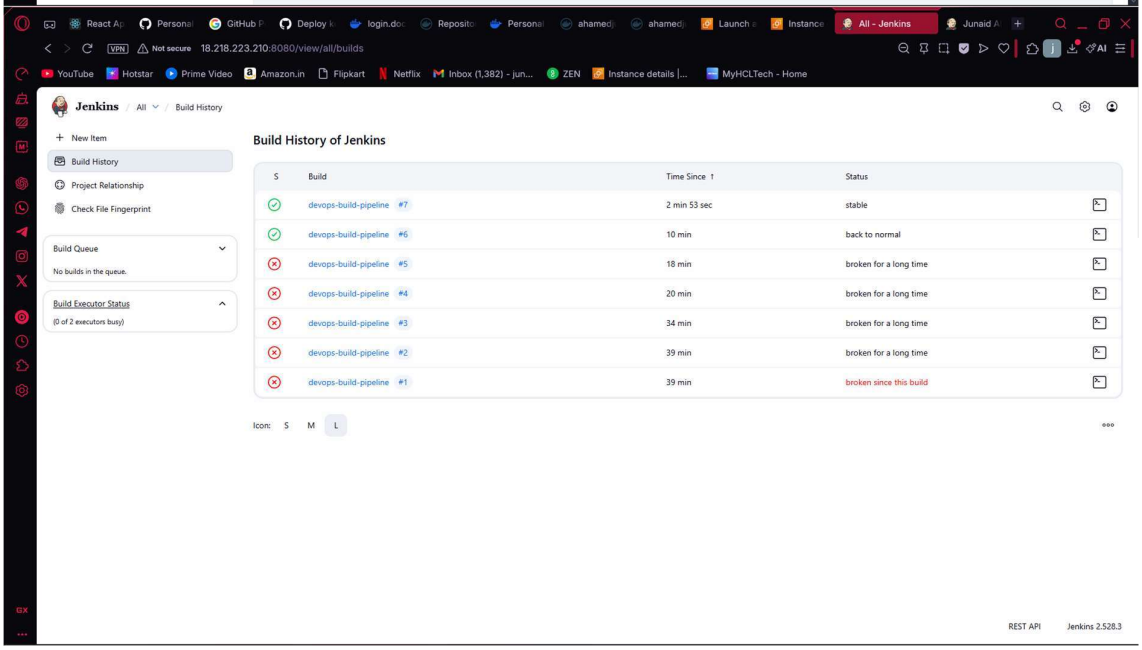
**Pipeline Logic:**

- Push to dev → Build & push DEV image
- Merge to main → Build & push PROD image


Jenkins / devops-build-pipeline / #7 / Console Output
Status
Changes
Console Output
Edit Build Information
Delete build '#7'
Timings
Git Build Data
Pipeline Overview
Restart from Stage
Replay
Pipeline Steps
Workspaces
Previous Build
Console Output
Download
Copy
View as plain text
Started by user Junaid Ahamed
Lightweight checkout support not available, falling back to full checkout.
Checking out git git@github.com:Junaid-dot-max/devops-build.git into /var/lib/jenkins/workspace/devops-build-pipeline@script/ff90f69f75671ddfef292ecac8b3bef9c9029b40ae61b07ae8f0f14b932c79e7 to read Jenkinsfile
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository git@github.com:Junaid-dot-max/devops-build.git
> git init /var/lib/jenkins/workspace/devops-build-pipeline@script/ff90f69f75671ddfef292ecac8b3bef9c9029b40ae61b07ae8f0f14b932c79e7 # timeout=10
Fetching upstream changes from git@github.com:Junaid-dot-max/devops-build.git
> git --version # timeout=10
> git --version # 'git version 2.34.1'
> git fetch --tags --force --progress -- git@github.com:Junaid-dot-max/devops-build.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url git@github.com:Junaid-dot-max/devops-build.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
Seen branch in repository origin/dev
Seen branch in repository origin/main
Seen branch in repository origin/master
Seen 3 remote branches
> git show-ref --tags -d # timeout=10
Checking out Revision 0507b3854edee98bc28e13027cdda5edbe22a32d (refs/remotes/origin/dev)
> git config core.sparsecheckout # timeout=10
> git checkout -f 0507b3854edee98bc28e13027cdda5edbe22a32d # timeout=10
Commit message: "Add Jenkins CI/CD pipeline"
> git rev-list --no-walk 0507b3854edee98bc28e13027cdda5edbe22a32d # timeout=10
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/devops-build-pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout

Jenkins / devops-build-pipeline / #7 / Console Output
Running on Jenkins in /var/lib/jenkins/workspace/devops-build-pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/devops-build-pipeline/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url git@github.com:Junaid-dot-max/devops-build.git # timeout=10
Fetching upstream changes from git@github.com:Junaid-dot-max/devops-build.git
> git --version # timeout=10
> git --version # 'git version 2.34.1'
> git fetch --tags --force --progress -- git@github.com:Junaid-dot-max/devops-build.git +refs/heads/*:refs/remotes/origin/* # timeout=10
Seen branch in repository origin/dev
Seen branch in repository origin/main
Seen branch in repository origin/master
Seen 3 remote branches
> git show-ref --tags -d # timeout=10
Checking out Revision 0507b3854edee98bc28e13027cdda5edbe22a32d (refs/remotes/origin/dev)
> git config core.sparsecheckout # timeout=10
> git checkout -f 0507b3854edee98bc28e13027cdda5edbe22a32d # timeout=10
Commit message: "Add Jenkins CI/CD pipeline"
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Checkout)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/devops-build-pipeline/.git # timeout=10
Fetching changes from the remote Git repository



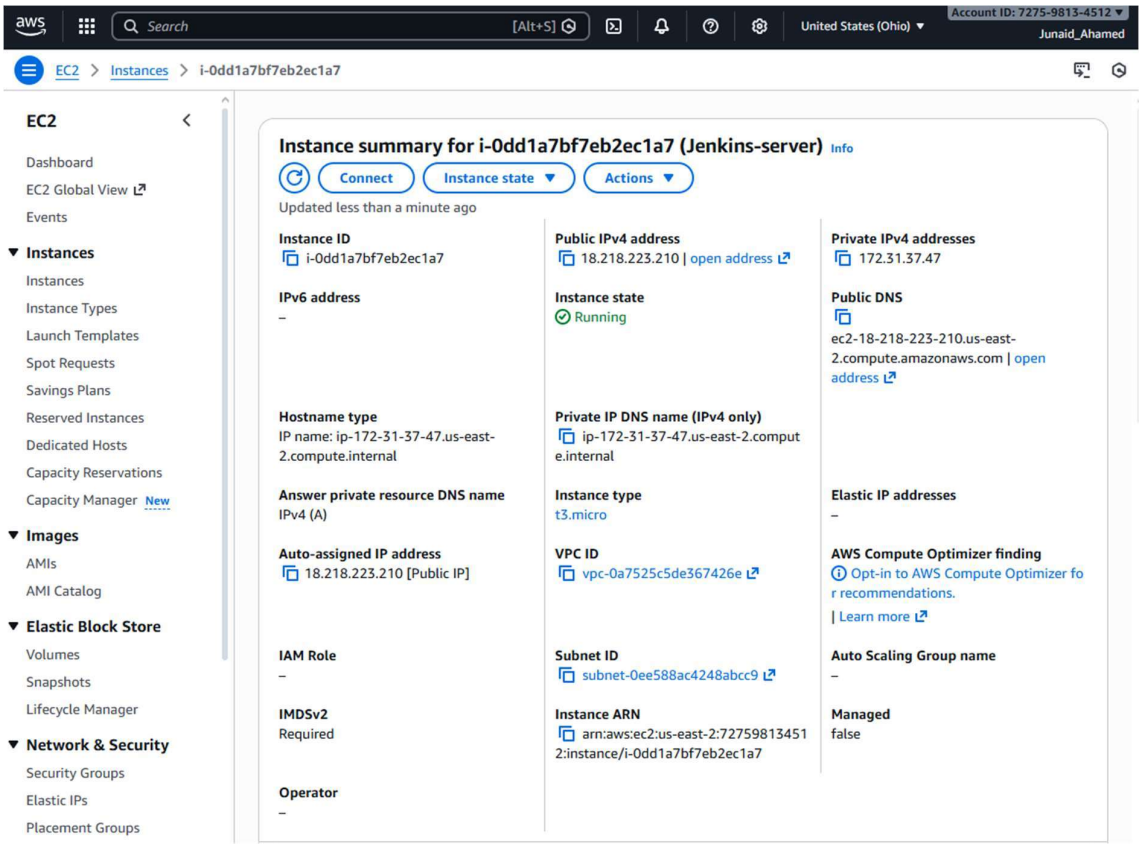




---

## 11. AWS EC2 Deployment

### 11.1 EC2 Instance Details

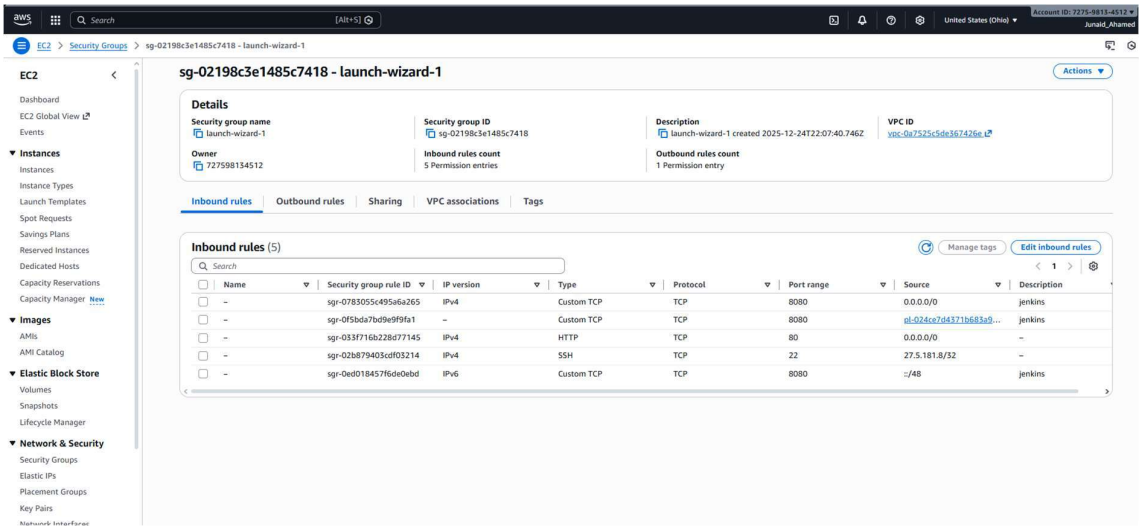
- Instance Type: t3.micro
- OS: Ubuntu 22.04
- Docker & Jenkins installed




### 11.2 Security Group Configuration

Security Groups were configured as per requirement:

| Port | Purpose | Access |
|---|---|---|
| 80 | Application | 0.0.0.0/0 |
| 22 | SSH | My IP |
| 8080 | Jenkins | My IP |

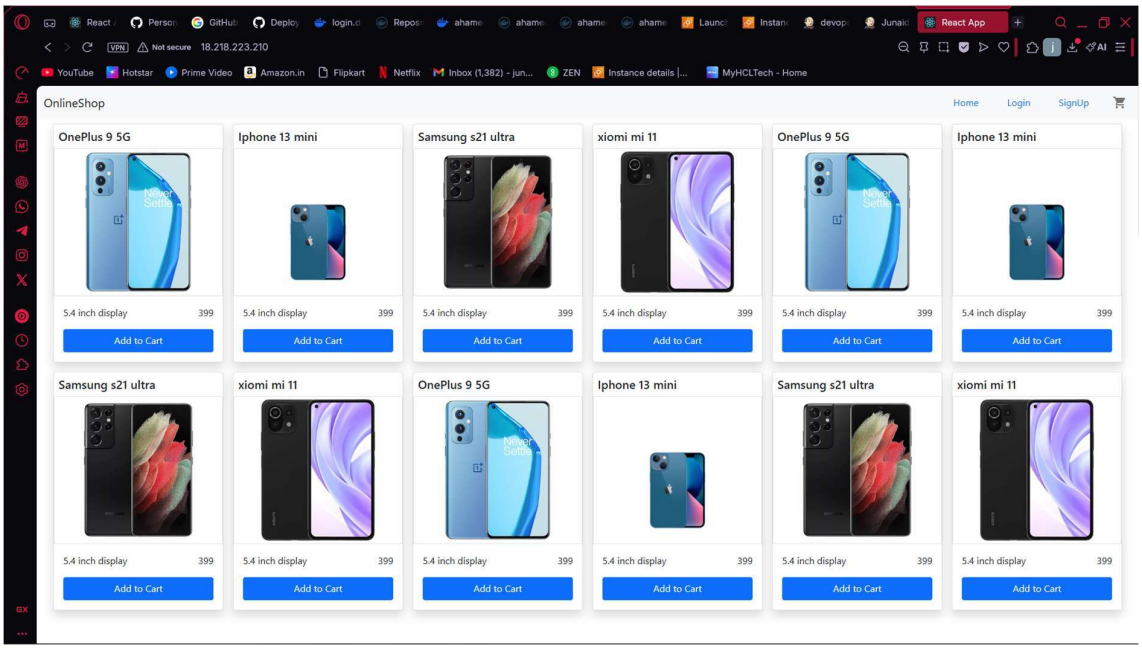



## 12. Application Deployment Verification

The application was successfully deployed and accessed via EC2 public IP.

**Application URL:**

http://18.218.223.210

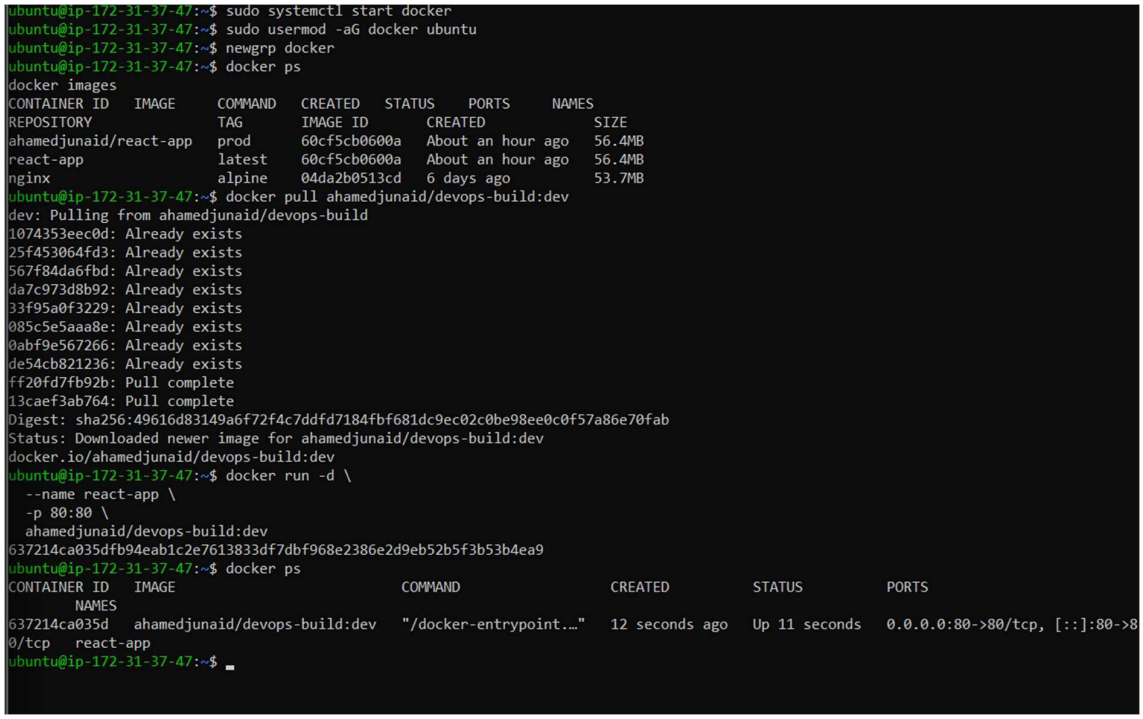



## 13. Monitoring & Health Check

Application health was monitored by:

- Docker container status
- HTTP availability check

The application remained continuously available, confirming deployment stability.




---

### 14. Conclusion

This project demonstrates a complete DevOps lifecycle implementation including containerization, CI/CD automation, cloud deployment, secure access control, and monitoring. All requirements specified by GUVI were successfully implemented and validated.

---

### 15. Tools & Technologies Used

- Docker
- Docker Compose
- Jenkins
- Git & GitHub
- Docker Hub
- AWS EC2
- Ubuntu Linux

---

✅ **END OF REPORT**