

WTF November Mini Project – Group G

This document presents all required deliverables for the WTF November Mini Project, demonstrating CI/CD deployment using Jenkins, Docker, and AWS EC2.

1. Jenkinsfile

```
pipeline {
    agent any

    environment {
        DOCKERHUB_REPO = 'susan22283/wtf_nov_project'
        CONTAINER_NAME = 'wtf_nov_mini_project'
        EC2_HOST        = 'ubuntu@ec2-52-91-123-0.compute-1.amazonaws.com'
        IMAGE_TAG        = "build-${env.BUILD_NUMBER}"
    }

    stages {
        stage('Checkout') {
            steps {
                checkout scm
            }
        }

        stage('Run tests') {
            steps {
                sh '''
                echo "Running tests... (currently no real tests)"
                '''
            }
        }

        stage('Build Docker image') {
            steps {
                sh '''
                echo "Building Docker image..."
                docker build -t ${DOCKERHUB_REPO}:${IMAGE_TAG} .
                '''
            }
        }

        stage('Push image to Docker Hub') {
            steps {
                withCredentials([
                    usernamePassword(
                        credentialsId: 'dockerhub-creds',
                        usernameVariable: 'DOCKER_USER',
                        passwordVariable: 'DOCKER_PASS'
                    )
                ]) {
                    sh '''
                    echo "$DOCKER_PASS" | docker login -u "$DOCKER_USER" --password-stdin
                    docker push ${DOCKERHUB_REPO}:${IMAGE_TAG}
                    docker logout
                    '''
                }
            }
        }

        stage('Deploy to EC2') {
            steps {
                sshagent(credentials: ['ec2-ssh-key']) {
                    sh '''
                    echo "Deploying to EC2..."

                    ssh -o StrictHostKeyChecking=no ${EC2_HOST} "
                    docker pull ${DOCKERHUB_REPO}:${IMAGE_TAG} &&
                    docker rm -f ${CONTAINER_NAME} || true &&
                    docker run -d --name ${CONTAINER_NAME} -p 8085:8000 ${DOCKERHUB_REPO}:${IMAGE_TAG}
                    "
                    '''
                }
            }
        }
    }

    post {
        always {
            sh 'docker image prune -f || true'
        }
    }
}
```

2. Jenkins Job Console Output

```
Started by user Amechi Susan Ogechi
Obtained Jenkinsfile from git https://github.com/women-techsters-fellowship/november_mini_project
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/wtf_nov_project
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential github-ssh-key
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/wtf_nov_project/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/women-techsters-fellowship/november_mini_project #
timeout=10
Fetching upstream changes from https://github.com/women-techsters-fellowship/november_mini_project
```

2. Jenkins Job Console Output (cont.)

```
timeout=10
Fetching upstream changes from https://github.com/women-techsters-fellowship/november_mini_project
> git --version # timeout=10
> git --version # 'git version 2.43.0'
using GIT_SSH to set credentials
Verifying host key using known hosts file
> git fetch --tags --force --progress -- https://github.com/women-techsters-fellowship/
november_mini_project +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse origin/Group_G^{commit} # timeout=10
Checking out Revision f8eaa62910cb813f40e79803e0e2d7c6ba8f1650 (origin/Group_G)
> git config core.sparsecheckout # timeout=10
> git checkout -f f8eaa62910cb813f40e79803e0e2d7c6ba8f1650 # timeout=10
Commit message: "Added EC2 host to ALLOWED_HOSTS"
> git rev-list --no-walk 10d161327a4d9795652c25372655c7e3a1ffc5 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withEnv
```

2. Jenkins Job Console Output (cont.)

```
[Pipeline] {  
[Pipeline] withEnv  
[Pipeline] {  
[Pipeline] stage  
[Pipeline] { (Checkout)  
[Pipeline] checkout  
Selected Git installation does not exist. Using Default  
The recommended git tool is: NONE  
using credential github-ssh-key  
  > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/wtf_nov_project/.git # timeout=10  
Fetching changes from the remote Git repository  
  > git config remote.origin.url https://github.com/women-techsters-fellowship/november_mini_project #  
timeout=10  
Fetching upstream changes from https://github.com/women-techsters-fellowship/november_mini_project  
  > git --version # timeout=10  
  > git --version # 'git version 2.43.0'  
using GIT_SSH to set credentials  
Verifying host key using known hosts file  
  > git fetch --tags --force --progress -- https://github.com/women-techsters-fellowship/  
november_mini_project +refs/heads/*:refs/remotes/origin/* # timeout=10
```

2. Jenkins Job Console Output (cont.)

```
[Pipeline] sh
+ echo Building Docker image...
Building Docker image...
+ docker build -t susan22283/wtf_nov_project:build-5 .
#0 building with "default" instance using docker driver

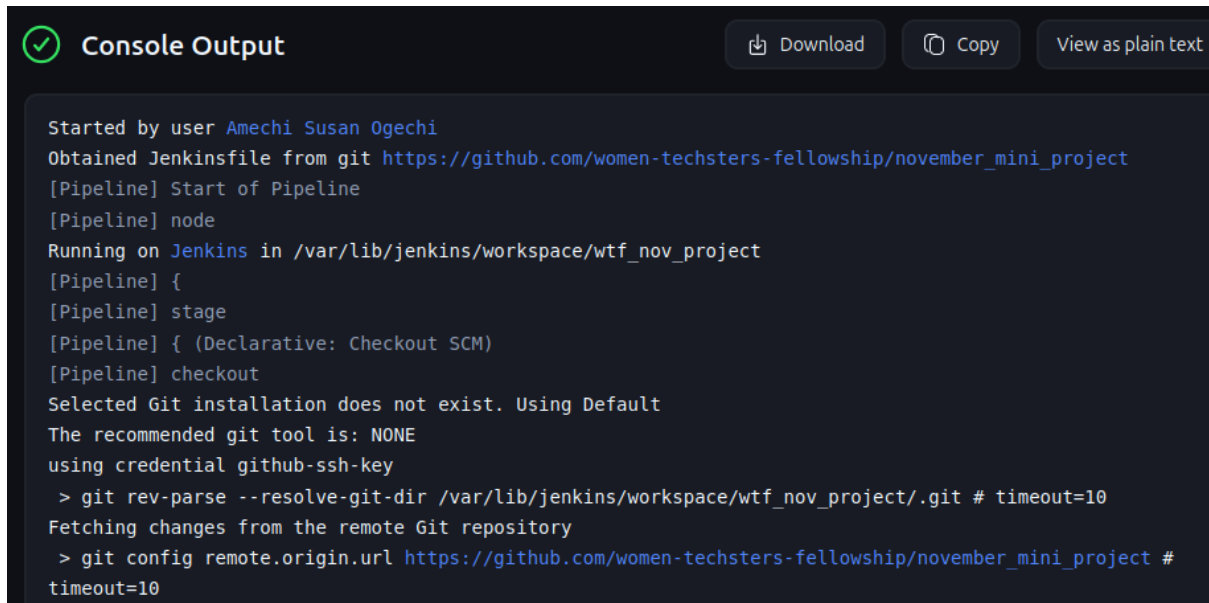
#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 1.12kB 0.0s done
#1 DONE 0.1s

#2 [internal] load metadata for docker.io/library/python:3.11-slim
#2 DONE 2.7s

#3 [internal] load .dockerignore
#3 transferring context:
#3 transferring context: 2B 0.0s done
#3 DONE 0.2s

#4 [1/6] FROM docker.io/library/python:3.11-
slim@sha256:7cd0079a9bd8800c81632d65251048fc2848bf9afda542224b1b10e0cae45575
```

2. Jenkins Job Console Output (cont.)

A screenshot of the Jenkins console output interface. At the top, there is a green checkmark icon followed by the text "Console Output". To the right of this header are three buttons: "Download", "Copy", and "View as plain text". The main area of the screenshot displays the console output text in a dark-themed font. The output shows the start of a Jenkins pipeline run by user Amechi Susan Ogechi, including steps for checking out a repository from GitHub and configuring the remote origin URL.

```
Started by user Amechi Susan Ogechi
Obtained Jenkinsfile from git https://github.com/women-techsters-fellowship/november_mini_project
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/wtf_nov_project
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential github-ssh-key
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/wtf_nov_project/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/women-techsters-fellowship/november_mini_project #
timeout=10
```

2. Jenkins Job Console Output (cont.)

```
+ echo Running tests... (currently no real tests)
Running tests... (currently no real tests)
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build Docker image)
[Pipeline] sh
+ echo Building Docker image...
Building Docker image...
+ docker build -t susan22283/wtf_nov_project:build-5 .
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 1.12kB 0.0s done
#1 DONE 0.1s

#2 [internal] load metadata for docker.io/library/python:3.11-slim
#2 DONE 2.7s

#3 [internal] load .dockerignore
```

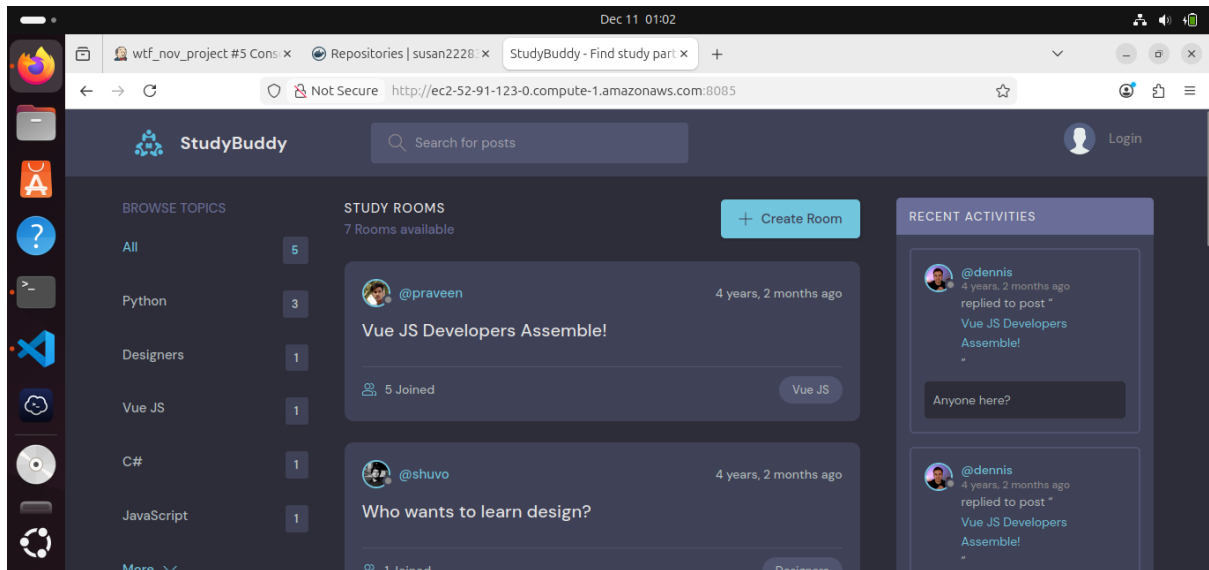

2. Jenkins Job Console Output (cont.)

```
27571c9cc27e: Layer already exists
600af8de593b: Layer already exists
77a2b55f8e8b: Layer already exists
fa384bf02ac1: Layer already exists
424dc4972605: Layer already exists
ba6bb5e095d2: Pushed
build-5: digest: sha256:61929e984ad23cf35fd03de714b291eaa5536d4b23d8b426cbe1869cf2aba266 size: 2209
+ docker logout
Removing login credentials for https://index.docker.io/v1/
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy to EC2)
[Pipeline] sshagent
[ssh-agent] Using credentials ubuntu
$ ssh-agent
SSH_AUTH_SOCK=/tmp/ssh-M7buuT1K73uu/agent.9995
SSH_AGENT_PID=9999
```

2. Jenkins Job Console Output (cont.)

```
[Pipeline] {  
[Pipeline] sh  
+ echo Deploying to EC2...  
Deploying to EC2...  
+ ssh -o StrictHostKeyChecking=no ubuntu@ec2-52-91-123-0.compute-1.amazonaws.com  
    docker pull susan22283/wtf_nov_project:build-5 &&  
    docker rm -f wtf_nov_mini_project || true &&  
    docker run -d --name wtf_nov_mini_project -p 8085:8000 susan22283/  
wtf_nov_project:build-5  
  
build-5: Pulling from susan22283/wtf_nov_project  
8846a30ba467: Pulling fs layer  
8846a30ba467: Download complete  
8846a30ba467: Pull complete  
Digest: sha256:61929e984ad23cf35fd03de714b291eaa5536d4b23d8b426cbe1869cf2aba266  
Status: Downloaded newer image for susan22283/wtf_nov_project:build-5  
docker.io/susan22283/wtf_nov_project:build-5  
wtf_nov_mini_project  
5df8cbf93cfbdb470b36dd39fe86f470058c12b99f94ce4d4bc7653b54735796  
[Pipeline] }
```

3. Application Running Live



URL: <http://ec2-52-91-123-0.compute-1.amazonaws.com:8085/>

4. EC2 and Docker Details

EC2 Public DNS: ec2-52-91-123-0.compute-1.amazonaws.com

EC2 Public IP: 52.91.123.0

AMI: Ubuntu 22.04 LTS

Instance type: t2.micro

Container internal port: 8000

Public port exposed (Docker mapping): 8085

Nginx reverse proxy port: 80

Docker Hub repository: susan22283/wtf_nov_project

Image tag used: susan22283/wtf_nov_project:build-5

Container run command (from Jenkins pipeline):

```
docker run -d --name wtf_nov_mini_project -p 8085:8000 susan22283/wtf_nov_project:build-5
```