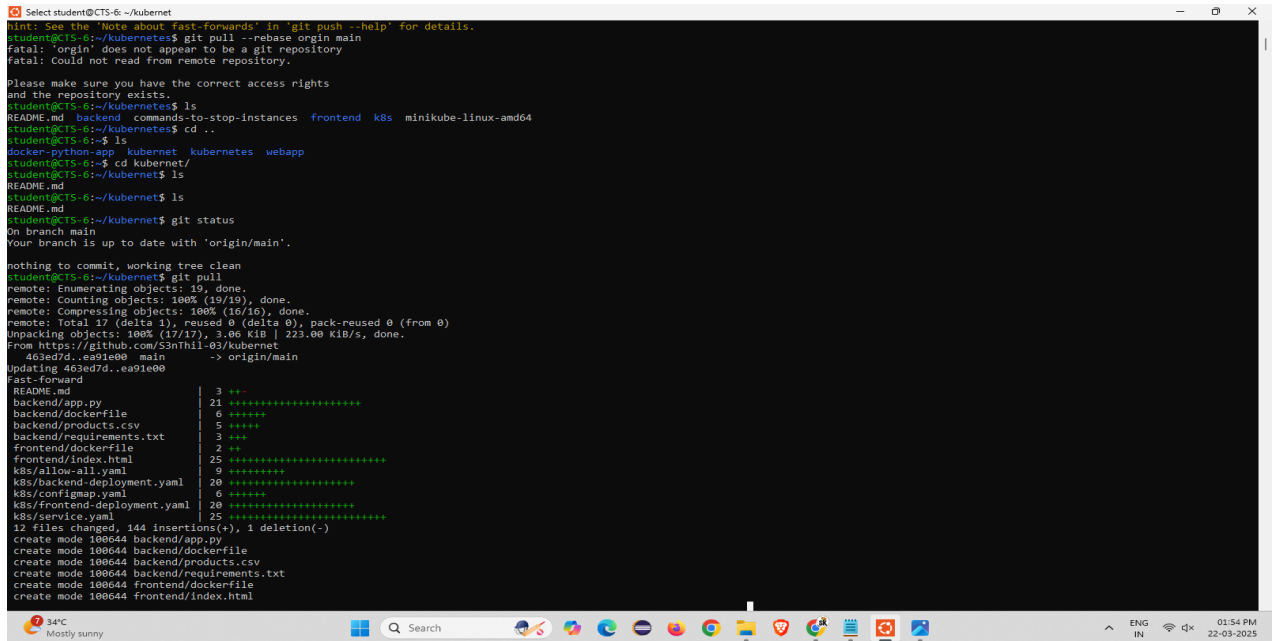


DEVOPS TRAINING

DAY 5 CONFIGURING PIPELINE

Step 1: Create github repository kubernetes and push the cd kubernetes files as frontend ,backend and k8s

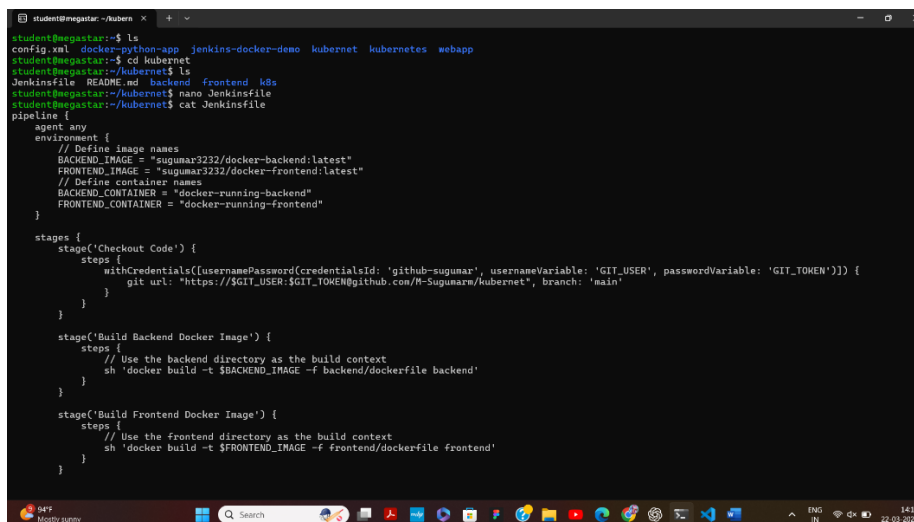


```
Select student@CTS-6: ~/kubernetes
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
student@CTS-6:~/kubernetes$ git pull --rebase origin main
fatal: 'origin' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
student@CTS-6:~/kubernetes$ ls
README.md  backend  commands-to-stop-instances  frontend  k8s  minikube-linux-amd64
student@CTS-6:~/kubernetes$ cd ..
student@CTS-6:~$ ls
docker-python-app  kubernetes  webapp
student@CTS-6:~/kubernetes$ ls
README.md
student@CTS-6:~/kubernetes$ ls
README.md
student@CTS-6:~/kubernetes$ git status
On branch main
Your branch is up to date with 'origin/main'.

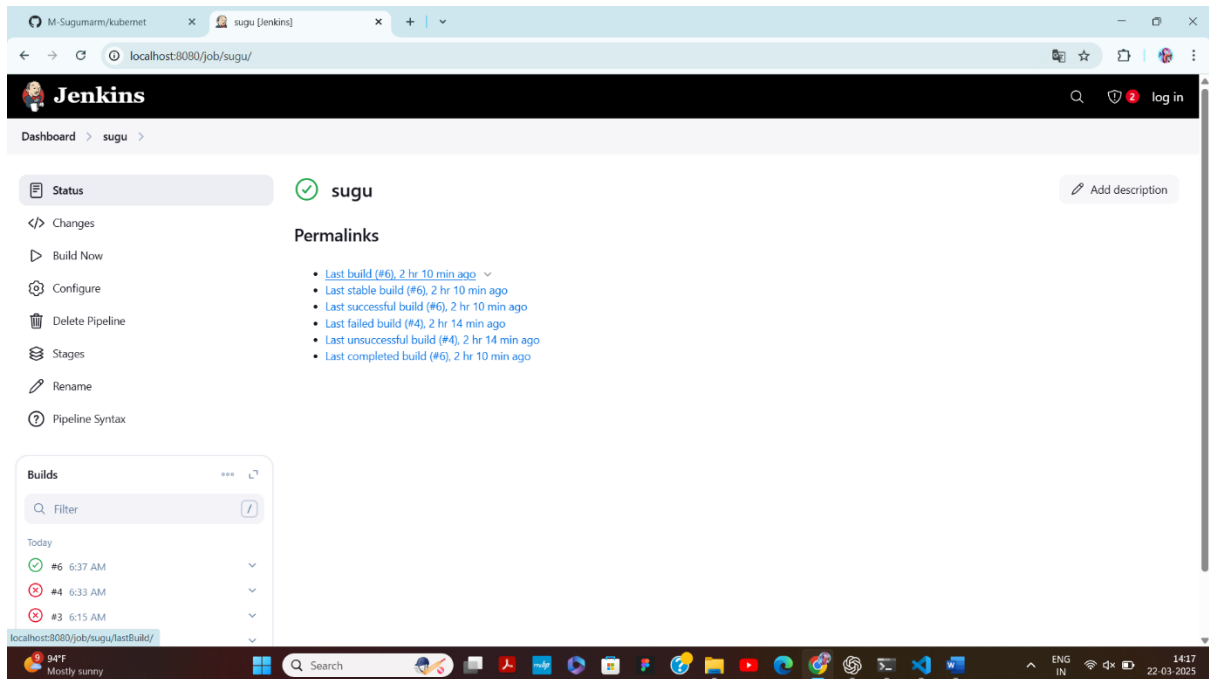
nothing to commit, working tree clean
student@CTS-6:~/kubernetes$ git pull
remote: Enumerating objects: 19, done.
remote: Counting objects: 100% (19/19), done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 17 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (17/17), 3.06 KiB | 223.00 KiB/s, done.
From https://github.com/S3nTh1l-03/kubernetes
  403ed7d..ea91e00  main    -> origin/main
Updating 403ed7d..ea91e00
Fast-forward
 README.md          | 3 +++
 backend/app.py     | 21 ++++++
 backend/dockerfile | 6 +++++
 backend/products.csv | 5 +++++
 backend/requirements.txt | 3 +++
 frontend/dockerfile | 2 ++
 frontend/index.html | 25 ++++++
 k8s/allow-all.yaml | 9 ++++++
 k8s/backend-deployment.yaml | 20 ++++++
 k8s/configmap.yaml | 6 +++++
 k8s/frontend-deployment.yaml | 20 ++++++
 k8s/service.yaml | 25 ++++++
 12 files changed, 144 insertions(+), 1 deletion(-)
 create mode 100644 backend/app.py
 create mode 100644 backend/dockerfile
 create mode 100644 backend/products.csv
 create mode 100644 backend/requirements.txt
 create mode 100644 frontend/dockerfile
 create mode 100644 frontend/index.html
```

Step 2: create Jenkinsfile and add the following code and push into the github repo kubernetes

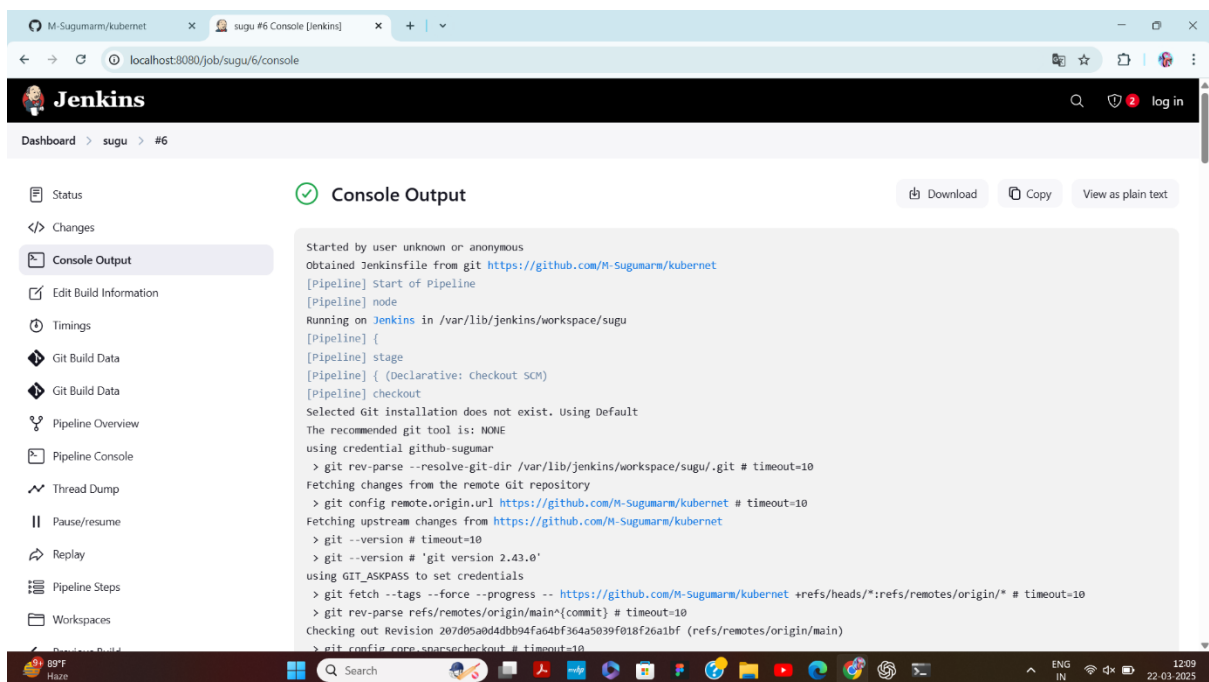


```
student@megastar: ~/kubern
config.yaml  docker-python-app  jenkins-docker-demo  kubernetes  kubernetes  webapp
student@megastar:~/kubern$ cd kubernetes
student@megastar:~/kubern$ ls
Jenkinsfile  README.md  backend  frontend  k8s
student@megastar:~/kubern$ nano Jenkinsfile
student@megastar:~/kubern$ cat Jenkinsfile
pipeline {
    agent any
    environment {
        // Define image names
        BACKEND_IMAGE = "sugumar3232/docker-backend:latest"
        FRONTEND_IMAGE = "sugumar3232/docker-frontend:latest"
        // Define container names
        BACKEND_CONTAINER = "docker-running-backend"
        FRONTEND_CONTAINER = "docker-running-frontend"
    }
    stages {
        stage('Checkout Code') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'github-sugumar', usernameVariable: 'GIT_USER', passwordVariable: 'GIT_TOKEN')]) {
                    git url: "https://$GIT_USER:$GIT_TOKEN@github.com/M-Sugumar/kubernetes", branch: 'main'
                }
            }
        }
        stage('Build Backend Docker Image') {
            steps {
                // Use the backend directory as the build context
                sh 'docker build -t $BACKEND_IMAGE -f backend/dockerfile backend'
            }
        }
        stage('Build Frontend Docker Image') {
            steps {
                // Use the frontend directory as the build context
                sh 'docker build -t $FRONTEND_IMAGE -f frontend/dockerfile frontend'
            }
        }
    }
}
```

Step 3: Open Jenkins create a item in pipeline and click ok and go to configure add the repo url and credentials and click build



Step 4 : open console output and check build is complete or not.



Step 5 : go to dashboard > manage Jenkins > plugins and install the Kubernetes once it all download success will shown.

The screenshot shows the Jenkins web interface. At the top is a black header with the Jenkins logo and name on the left, and search, shield, and 'log in' links on the right. Below the header is a breadcrumb trail: 'Dashboard > Manage Jenkins > Plugins'. The main content area is titled 'Download progress' and features a sidebar on the left with navigation links: 'Updates', 'Available plugins', 'Installed plugins', 'Advanced settings', and 'Download progress' (which is highlighted). The 'Download progress' section shows a 'Preparation' step with a red circle containing the number 3, followed by a list of steps: 'Checking internet connectivity', 'Checking update center connectivity', and 'Success'. Below this, a list of installed plugins is shown with green checkmarks and 'Success' status: 'Kubernetes Client API', 'Authentication Tokens API', 'Kubernetes Credentials', 'Kubernetes', and 'Loading plugin extensions'. At the bottom of the section, there are two links: 'Go back to the top page (you can start using the installed plugins right away)' and 'Restart Jenkins when installation is complete and no jobs are running' (with an unchecked checkbox).

REST API Jenkins 2.492.2