

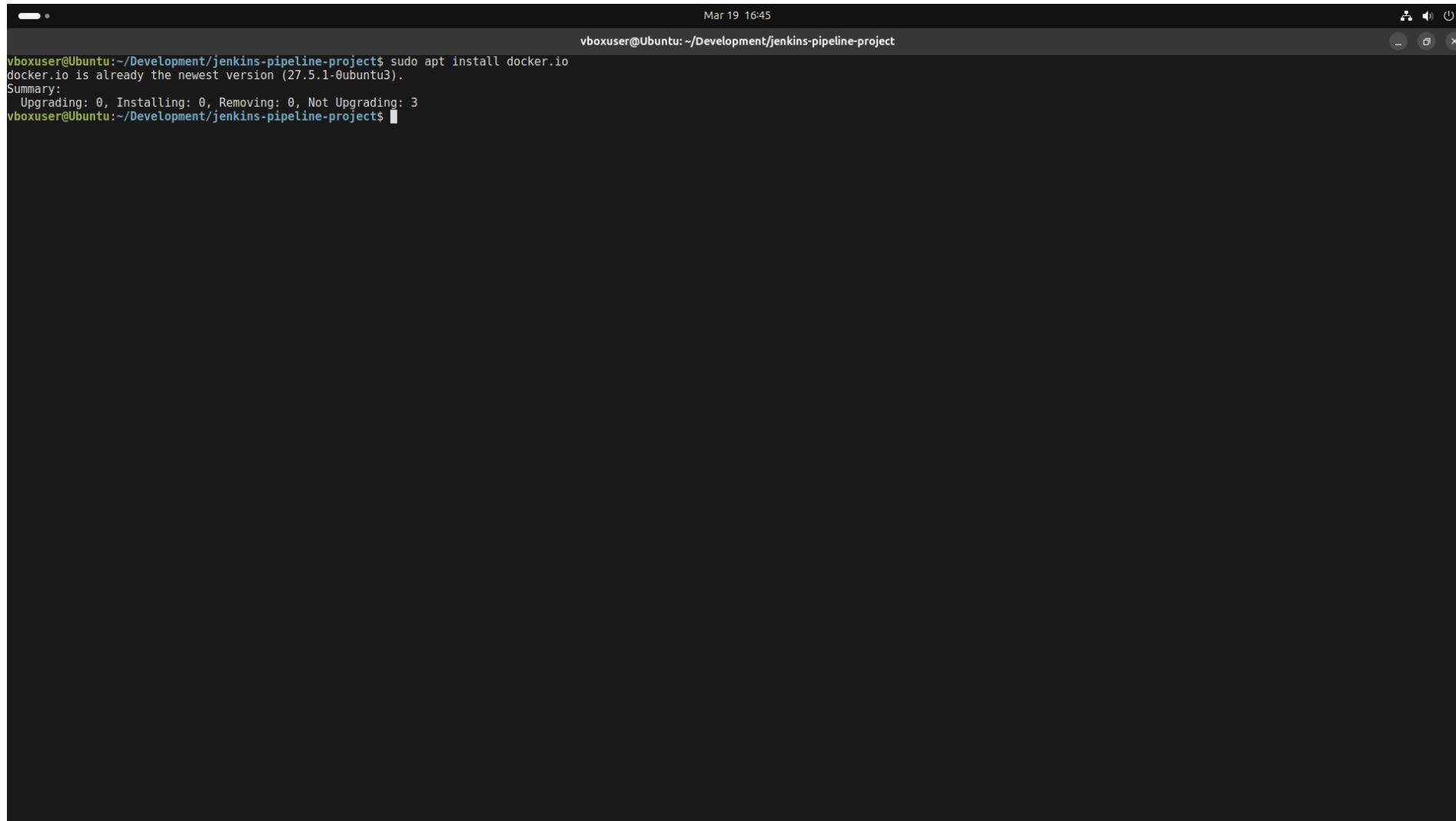
Setting up a pipeline project in Jenkins

Day - 2

24MCR029

Harikrishnan N

Installing docker.io



Mar 19 16:45
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project\$ sudo apt install docker.io
docker.io is already the newest version (27.5.1-0ubuntu3).
Summary:
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 3
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project\$

Installing docker-compose

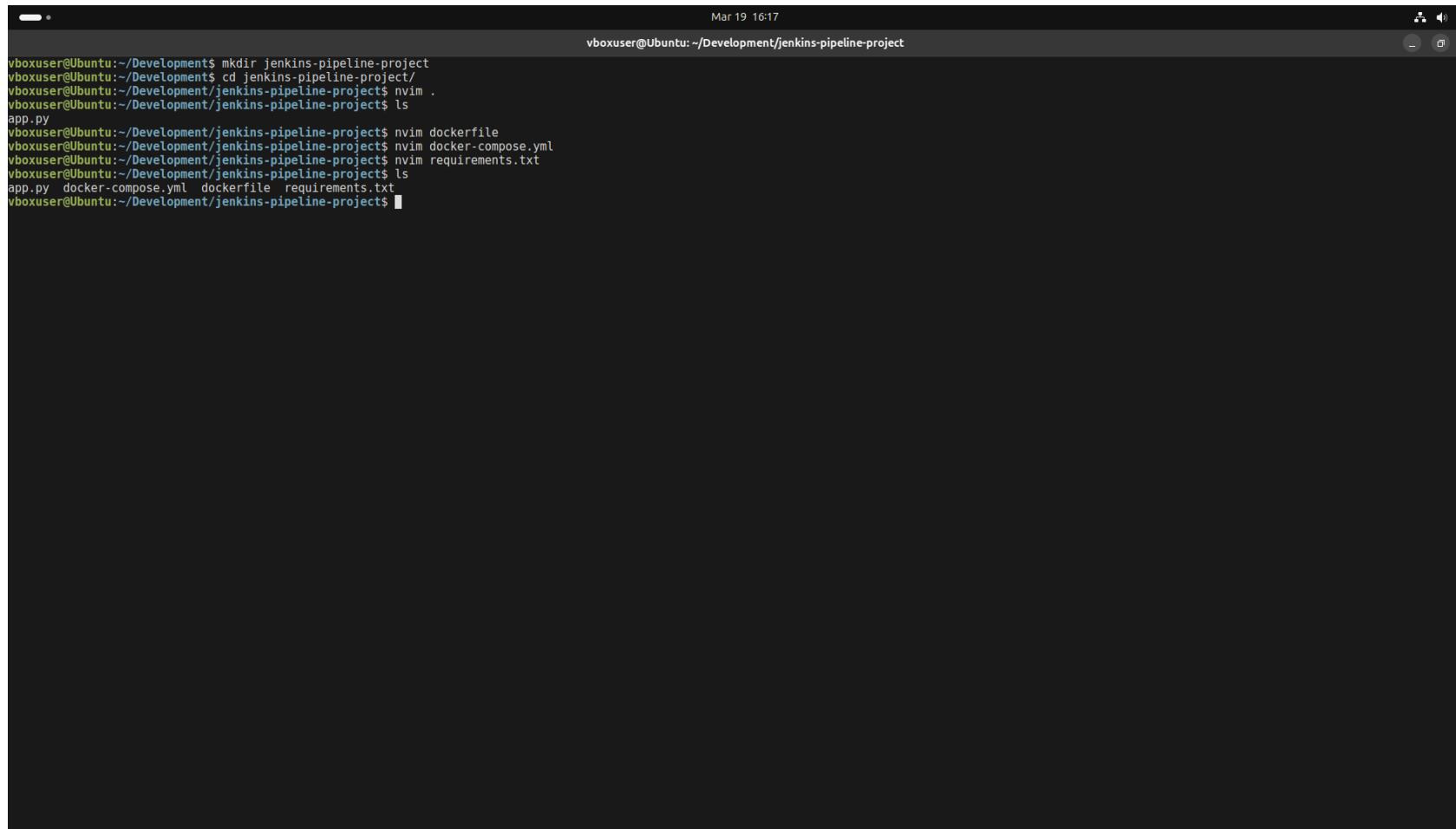
```
Mar 19 16:39
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project

Selecting previously unselected package runc.
Preparing to unpack .../runc_1.2.5-0ubuntu1_arm64.deb ...
Unpacking runc (1.2.5-0ubuntu1) ...
Selecting previously unselected package containerd.
Preparing to unpack .../3-containerd_2.0.2-0ubuntu1_arm64.deb ...
Unpacking containerd (2.0.2-0ubuntu1) ...
Selecting previously unselected package docker.io.
Preparing to unpack .../4-docker.io_27.5.1-0ubuntu3_arm64.deb ...
Unpacking docker.io (27.5.1-0ubuntu3) ...
Selecting previously unselected package ubuntu-fan.
Preparing to unpack .../5-ubuntu-fan_0.12.16_all.deb ...
Unpacking ubuntu-fan (0.12.16) ...
Setting up runc (1.2.5-0ubuntu1) ...
Setting up bridge-utils (1.7.1-4ubuntu1) ...
Setting up pigz (2.8-1) ...
Setting up containerd (2.0.2-0ubuntu1) ...
Created symlink '/etc/systemd/system/multi-user.target.wants/containerd.service' → '/usr/lib/systemd/system/containerd.service'.
Setting up ubuntu-fan (0.12.16) ...
Created symlink '/etc/systemd/system/multi-user.target.wants/ubuntu-fan.service' → '/usr/lib/systemd/system/ubuntu-fan.service'.
Setting up docker.io (27.5.1-0ubuntu3) ...
info: Selecting GID from range 100 to 999 ...
info: Adding group 'docker' (GID 125) ...
Created symlink '/etc/systemd/system/multi-user.target.wants/docker.service' → '/usr/lib/systemd/system/docker.service'.
Created symlink '/etc/systemd/system/sockets.target.wants/docker.socket' → '/usr/lib/systemd/system/docker.socket'.
Processing triggers for man-db (2.13.0-1) ...
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo: curl: command not found
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ sudo apt install curl -y
Installing:
curl

Summary:
Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 3
Download size: 252 kB
Space needed: 515 kB / 20.4 GB available

Get:1 http://ports.ubuntu.com/ubuntu-ports plucky/main arm64 curl arm64 8.12.1-3ubuntu1 [252 kB]
Fetched 252 kB in 6s (42.7 kB/s)
Selecting previously unselected package curl.
(Reading database ... 154814 files and directories currently installed.)
Preparing to unpack .../curl_8.12.1-3ubuntu1_arm64.deb ...
Unpacking curl (8.12.1-3ubuntu1) ...
Setting up curl (8.12.1-3ubuntu1) ...
Processing triggers for man-db (2.13.0-1) ...
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
  % Total    % Received % Xferd  Average Speed   Time   Time     Time  Current
                                 Dload  Upload Total   Spent    Left Speed
0     0    0     0    0     0      0 --:--:-- --:--:--      0
0     0    0     0    0     0      0 0:00:01 --:--:--      0
100 69.7M 100 69.7M  0     0 188k      0 0:06:19 0:06:19 --:--:-- 239k
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ ls
app.py docker-compose.yml dockerfile requirements.txt
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ ls
app.py docker-compose.yml dockerfile requirements.txt
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ sudo apt sudo apt sudo apt sudo apt sudo apt sudo apt ^C
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$
```

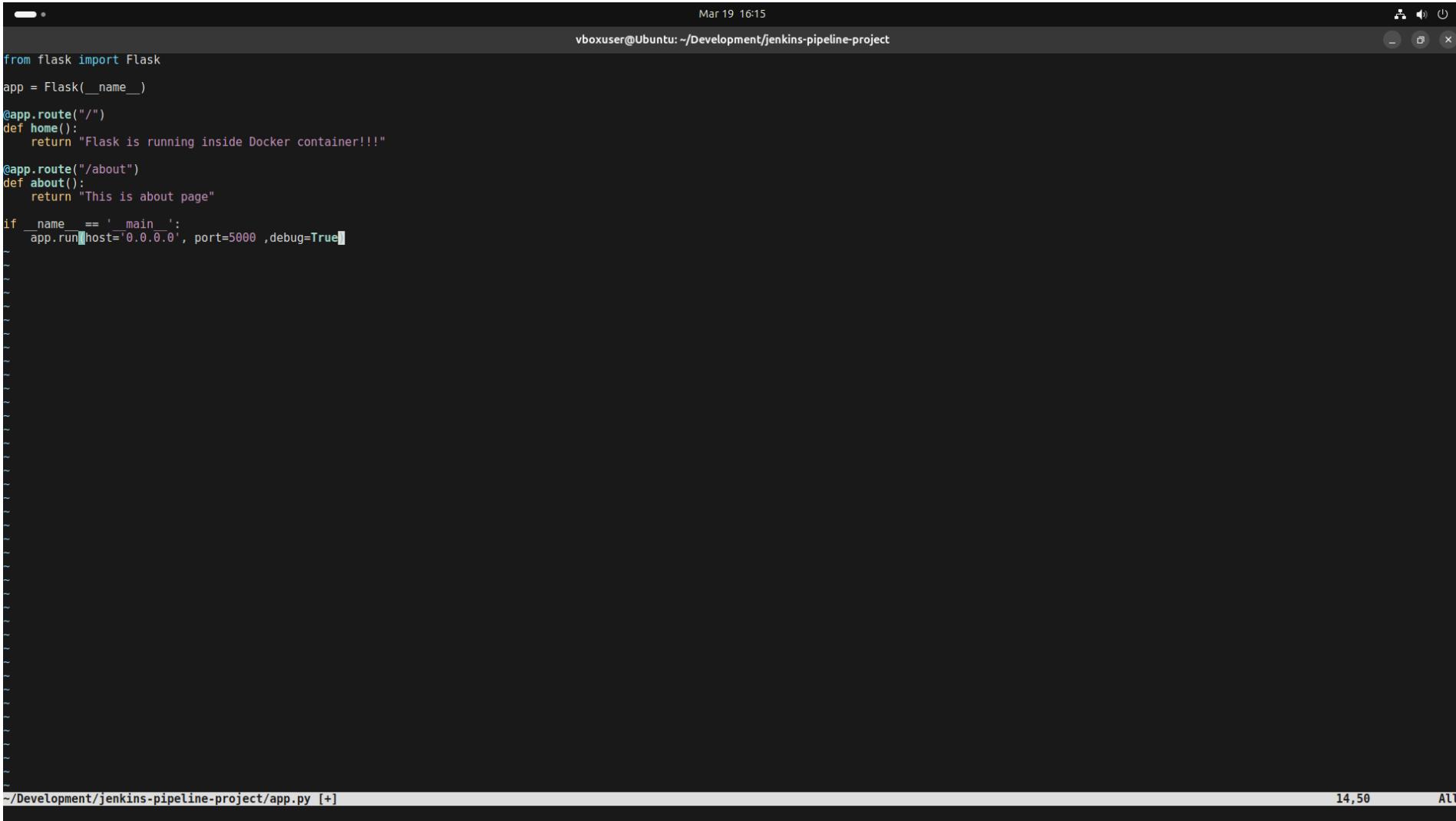
Creating a simple python flask application and running it using docker



The screenshot shows a terminal window on a Linux system (Ubuntu) with a dark theme. The terminal title is "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project". The timestamp at the top left is "Mar 19 16:17". The terminal content shows the following command history:

```
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ mkdir jenkins-pipeline-project
vboxuser@Ubuntu:~/Development$ cd jenkins-pipeline-project/
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ nvim .
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ ls
app.py
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ nvim dockerfile
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ nvim docker-compose.yml
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ nvim requirements.txt
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ ls
app.py  docker-compose.yml  dockerfile  requirements.txt
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$
```

Contents of app.py

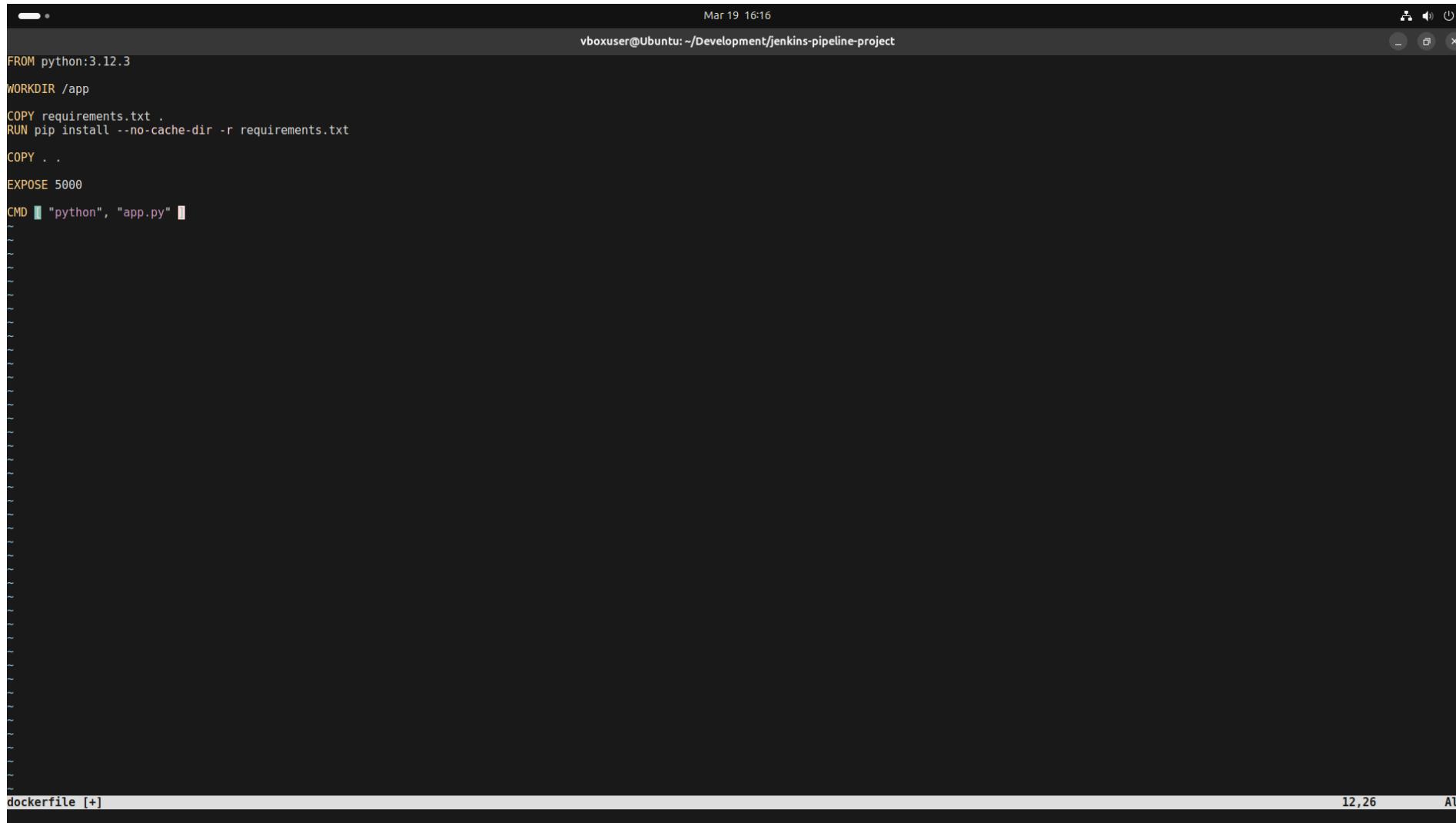


The screenshot shows a terminal window with the following details:

- Terminal title: Mar 19 16:15
- Terminal path: vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project
- Code content:

```
from flask import Flask
app = Flask(__name__)
@app.route("/")
def home():
    return "Flask is running inside Docker container!!!"
@app.route("/about")
def about():
    return "This is about page"
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```
- Bottom status bar: ~/Development/jenkins-pipeline-project/app.py [+] 14,50 All

Contents of dockerfile

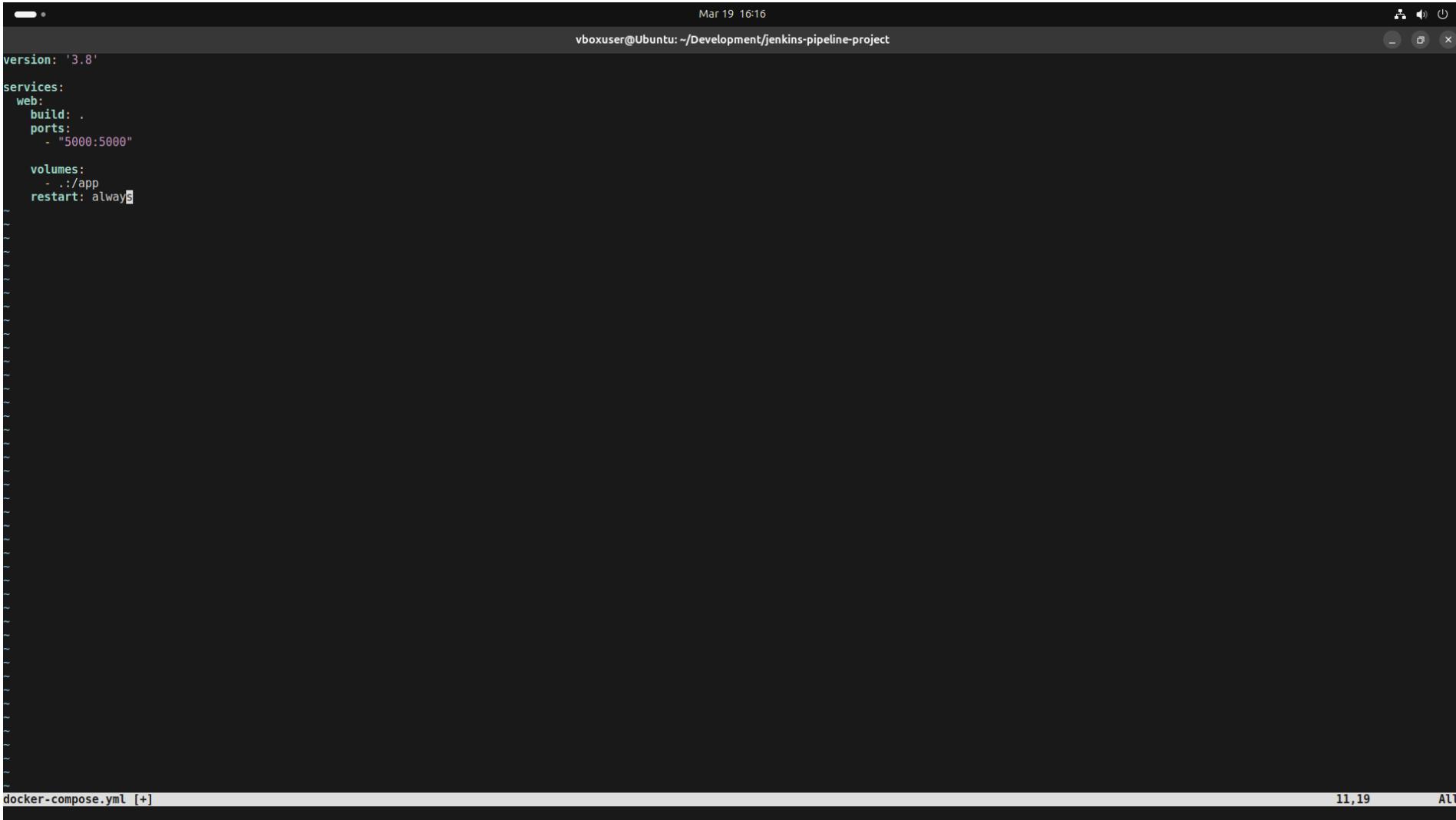


A screenshot of a terminal window titled "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project". The window shows the following Dockerfile content:

```
FROM python:3.12.3
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY . .
EXPOSE 5000
CMD ["python", "app.py"]
```

The terminal window has a dark background and light-colored text. The title bar includes the date and time: "Mar 19 16:16". The bottom status bar shows "dockerfile [+] 12,26 All".

Contents of docker-compose.yml

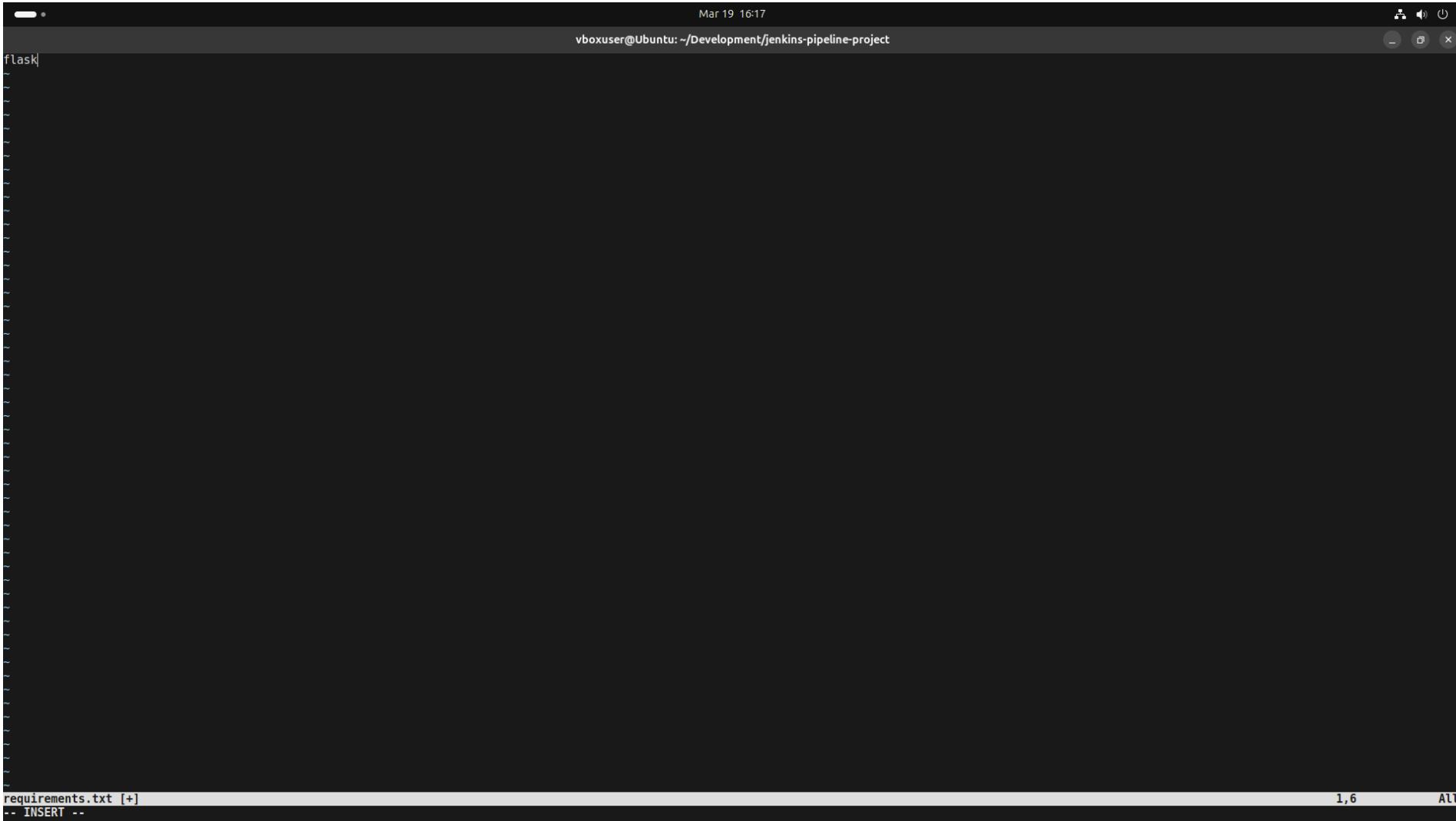


A screenshot of a terminal window titled "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project". The window shows the configuration file "docker-compose.yml" with the following content:

```
version: '3.8'
services:
  web:
    build: .
    ports:
      - "5000:5000"
    volumes:
      - ..:/app
    restart: always
```

The terminal window has a dark background and light-colored text. The status bar at the bottom shows "docker-compose.yml [+] 11,19 All".

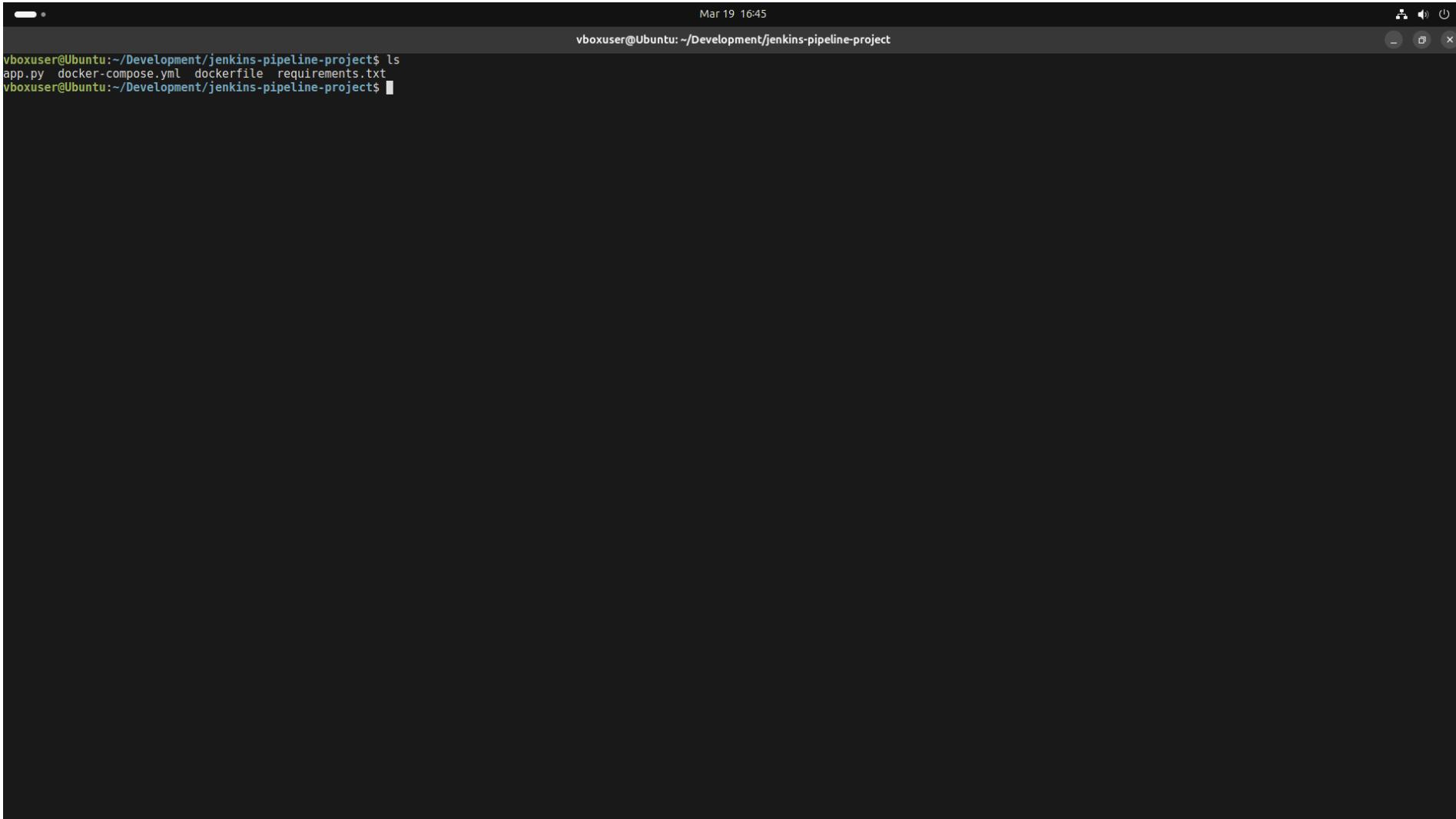
Contents of requirements.txt



A screenshot of a terminal window titled "Mar 19 16:17" and "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project". The window shows a single line of text: "flask". At the bottom of the terminal, the status bar displays "requirements.txt [+] 1,6 All" and "-- INSERT --".

```
flask
```

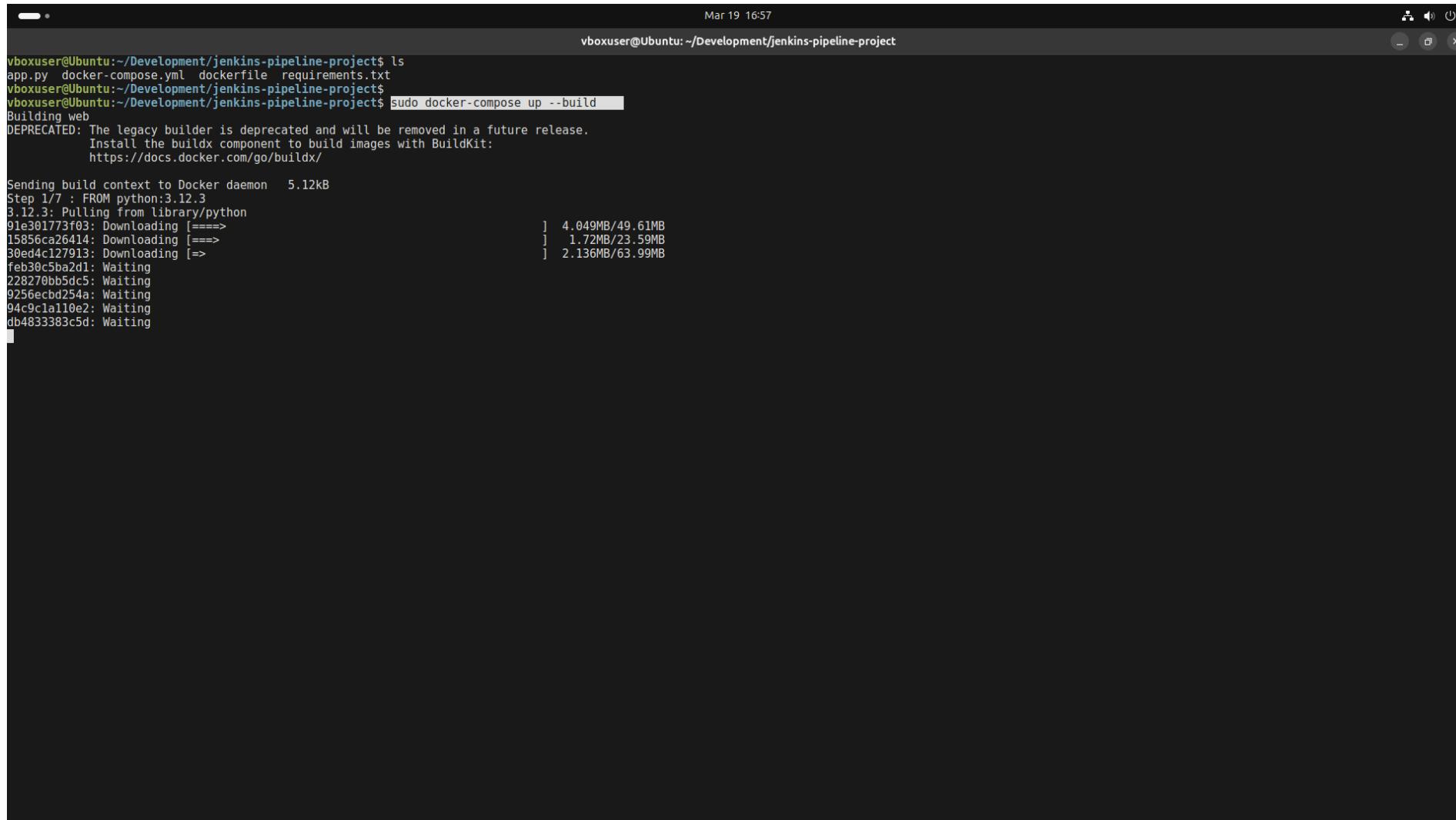
Overall project files



A screenshot of a terminal window titled "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project". The window shows the command "ls" being run, listing four files: "app.py", "docker-compose.yml", "dockerfile", and "requirements.txt". The terminal has a dark background and light-colored text. The title bar includes the date and time "Mar 19 16:45".

```
Mar 19 16:45
vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project
app.py docker-compose.yml dockerfile requirements.txt
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$
```

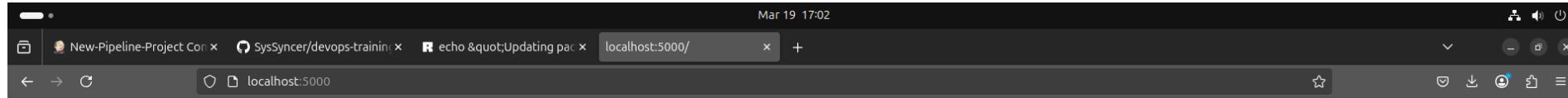
Building an image using docker-compose



The screenshot shows a terminal window on a Linux system (Ubuntu) with the title "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project". The terminal displays the command "sudo docker-compose up --build" being run, followed by output indicating the build process for a service named "web". The output includes a warning about the legacy builder being deprecated, instructions to use BuildKit, and a progress bar for pulling the Python image.

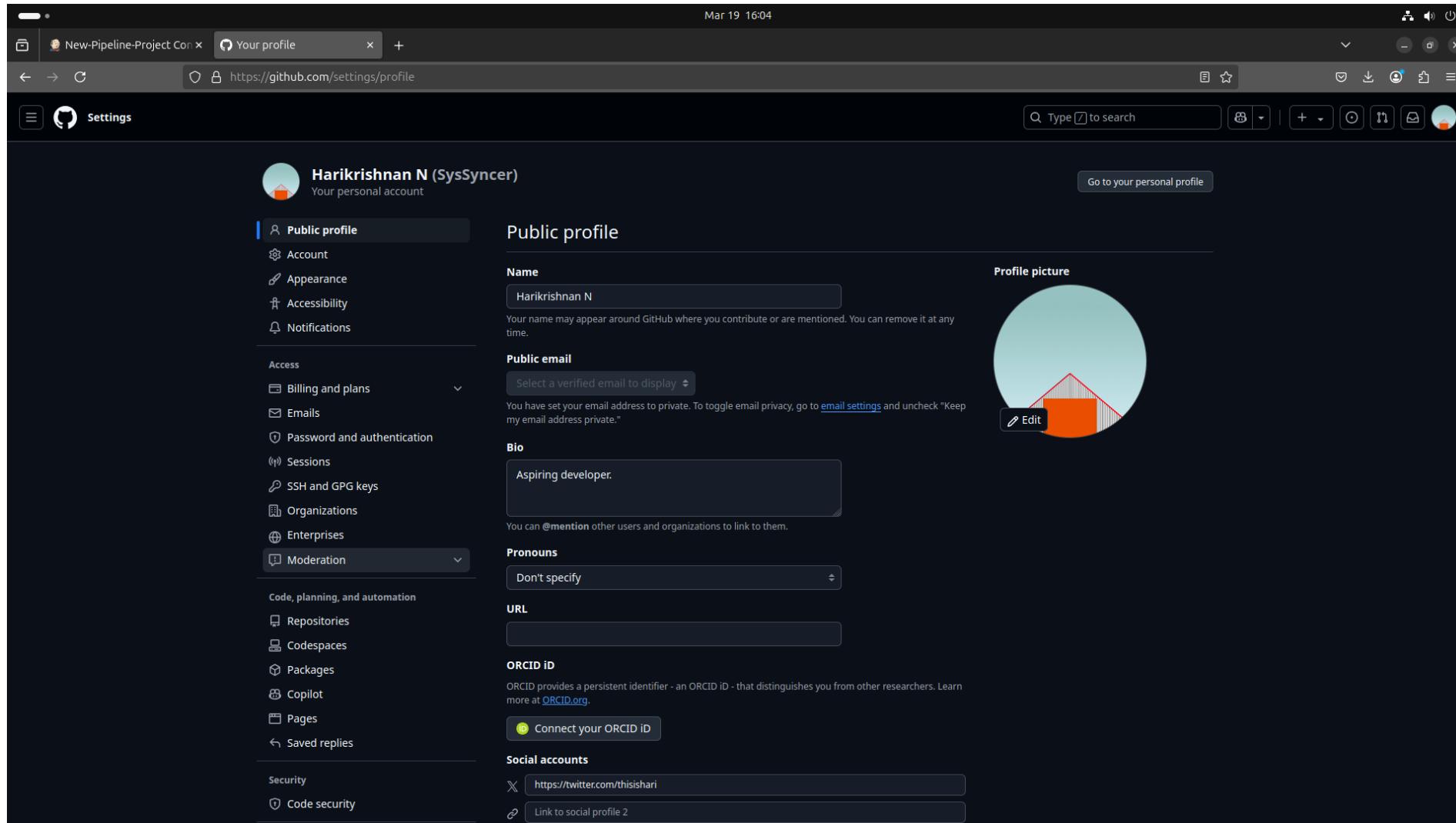
```
Mar 19 16:57
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ ls
app.py docker-compose.yml dockerfile requirements.txt
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ sudo docker-compose up --build
Building web
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
  Install the buildx component to build images with BuildKit:
  https://docs.docker.com/go/buildx/
Sending build context to Docker daemon 5.12kB
Step 1/7 : FROM python:3.12.3
3.12.3: Pulling from library/python
91e301773f03: Downloading [====>] 4.049MB/49.61MB
15856ca26414: Downloading [====>] 1.72MB/23.59MB
30ed4c127913: Downloading [=>] 2.136MB/63.99MB
feb30c5ba2d1: Waiting
228270bb5dc5: Waiting
9256ecbd254a: Waiting
94c9c1a110e2: Waiting
db4833383c5d: Waiting
```

Flask application now running inside docker container



Flask is running inside Docker container!!!

GitHub Personal Access Token Generation 1



The screenshot shows the GitHub Settings interface for a user named Harikrishnan N (SysSyncer). The left sidebar is titled "Settings" and contains several sections: "Public profile" (selected), "Account", "Appearance", "Accessibility", "Notifications", "Access" (with "Billing and plans" expanded), "Emails", "Password and authentication", "Sessions", "SSH and GPG keys", "Organizations", "Enterprises", "Moderation" (selected), "Code, planning, and automation" (with "Repositories", "Codespaces", "Packages", "Copilot", "Pages", and "Saved replies" listed), and "Security" and "Code security". The main content area is titled "Public profile" and includes fields for "Name" (Harikrishnan N), "Profile picture" (a placeholder image of a mountain), "Public email" (a dropdown menu showing "Select a verified email to display"), "Bio" (Aspiring developer.), "Pronouns" (Don't specify), "URL" (an empty input field), and "ORCID ID" (a link to connect an ORCID ID). Below the URL field, there is a "Social accounts" section with two entries: "https://twitter.com/thisishari" and "Link to social profile 2". The top of the browser window shows the URL https://github.com/settings/profile and the date Mar 19 16:04.

GitHub Personal Access Token Generation 2

The screenshot shows the GitHub profile settings page. The left sidebar is collapsed, and the main area displays various profile customization options. The 'Developer settings' section is currently selected, indicated by a dark gray background.

Pronouns: Don't specify

URL: [Empty input field]

ORCID iD: [Empty input field] Connect your ORCID iD

Social accounts: https://twitter.com/thisishari, Link to social profile 2, Link to social profile 3, Link to social profile 4

Company: [Empty input field]

Location: [Empty input field]

Display current local time: [unchecked checkbox] Other users will see the time difference from their local time.

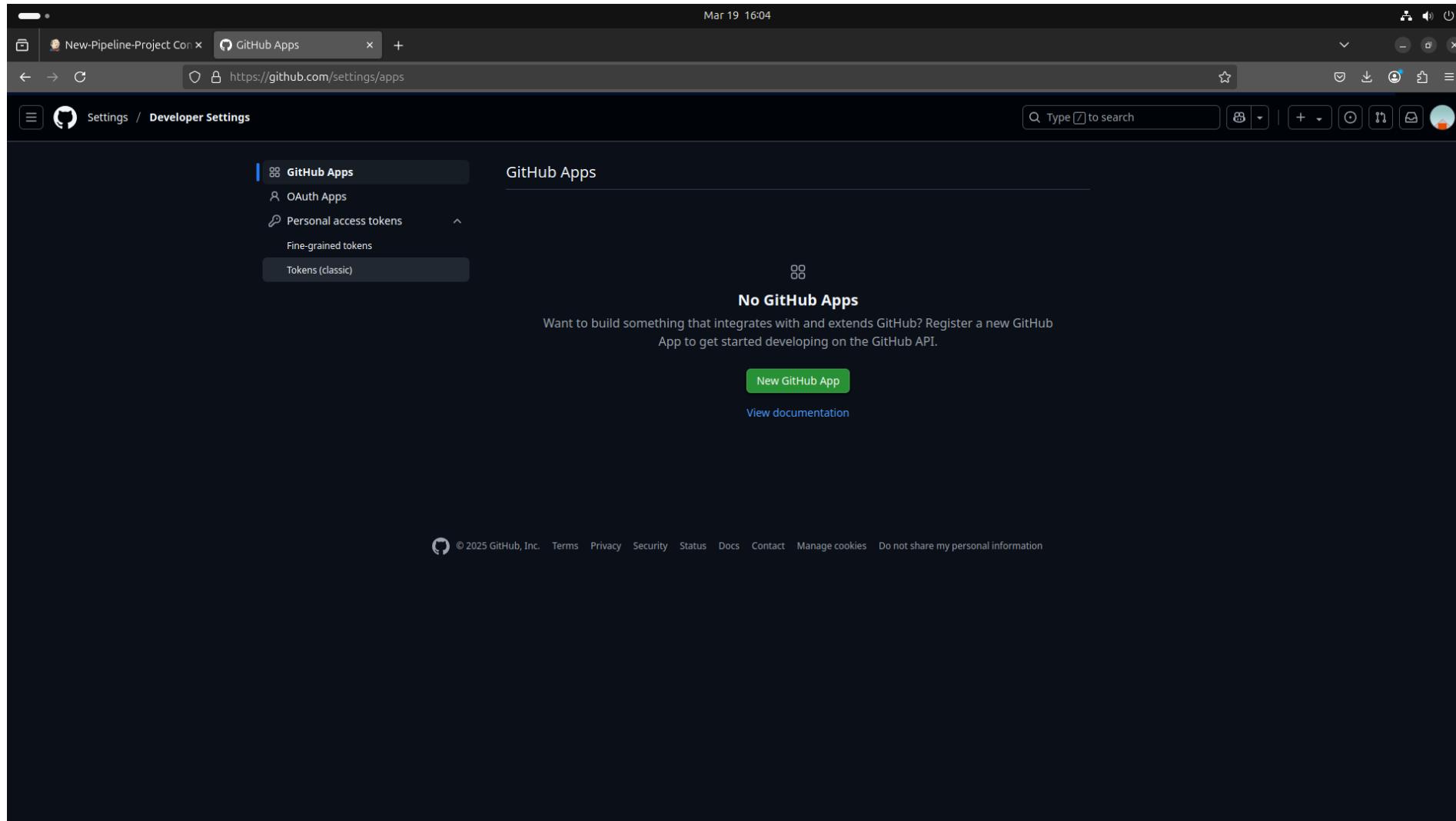
All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user profile appears. Please see our [privacy statement](#) to learn more about how we use this information.

Contributions & activity

Make profile private and hide activity

<https://github.com/settings/apps>

GitHub Personal Access Token Generation 3



GitHub Personal Access Token Generation 4

The screenshot shows the GitHub Personal Access Tokens page. The URL is <https://github.com/settings/tokens>. The page displays three existing tokens:

- vbox-kongu** — admin:repo_hook, repo, workflow
Expires on Fri, Apr 18 2025.
- devops-training** — admin:enterprise, admin:gpg_key, admin:org, admin:org_hook, admin:public_key, admin:repo_hook, admin:ssh_signing_key, audit_log, codespace, copilot, delete:packages, delete_repo, gist, notifications, project, repo, user, workflow, write:discussion, write:network_configurations, write:packages
Last used within the last week
Expires on Fri, May 16 2025.
- my token** — admin:enterprise, admin:gpg_key, admin:org, admin:org_hook, admin:public_key, admin:repo_hook, admin:ssh_signing_key, audit_log, codespace, copilot, delete:packages, delete_repo, gist, notifications, project, repo, user, workflow, write:discussion, write:packages
Last used within the last 7 months
Expires on Tue, Aug 26 2025.

A modal window is open, titled "Personal access tokens (classic)", with the sub-section "Generate new token". It contains two options:

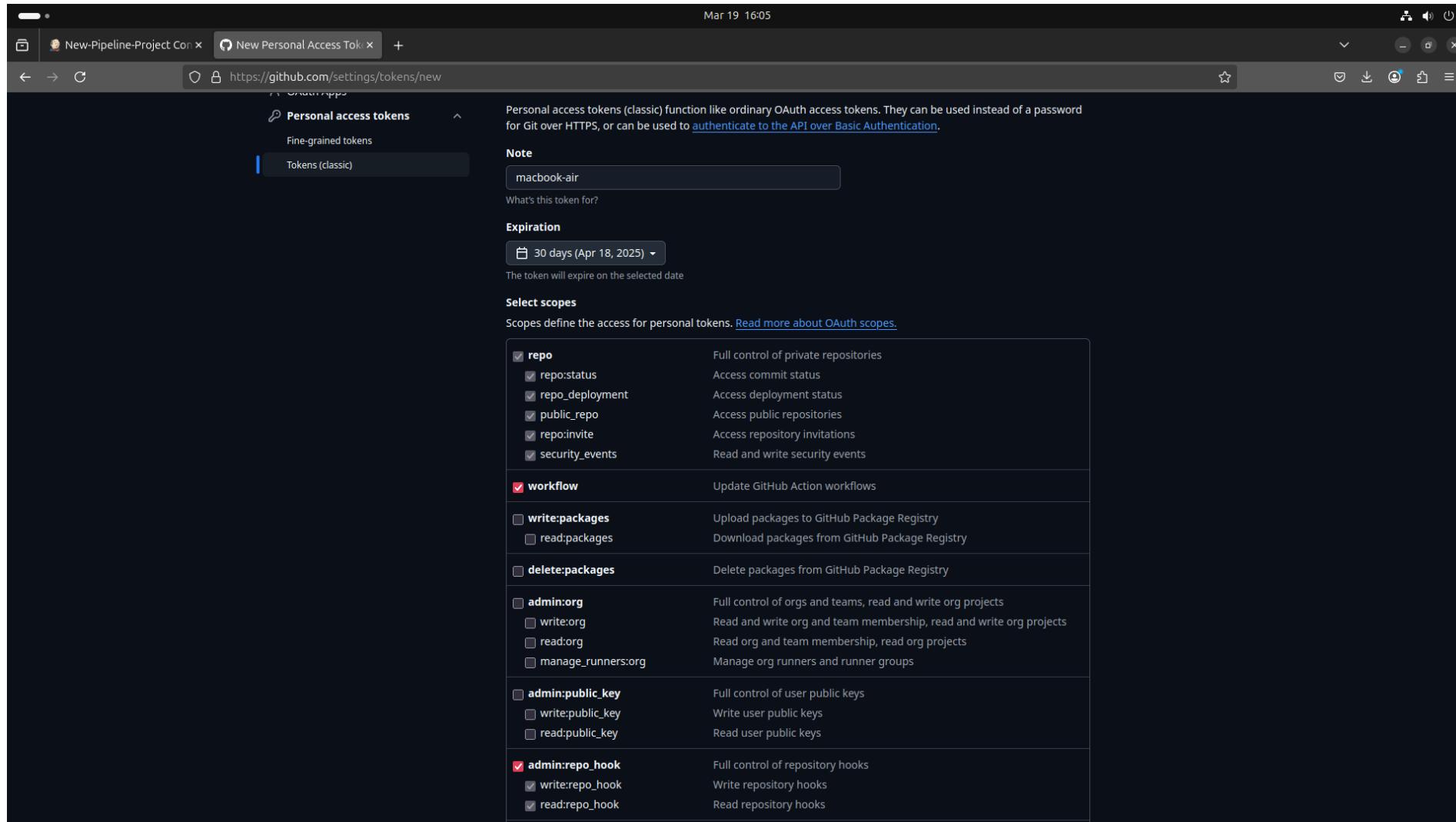
- Generate new token** Fine-grained, repo-scoped
- Generate new token (classic)** For general use (selected)

At the bottom of the modal, it says: "Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#)".

At the bottom of the page, there is a footer with links: © 2025 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#).

The URL in the address bar is <https://github.com/settings/tokens/new>.

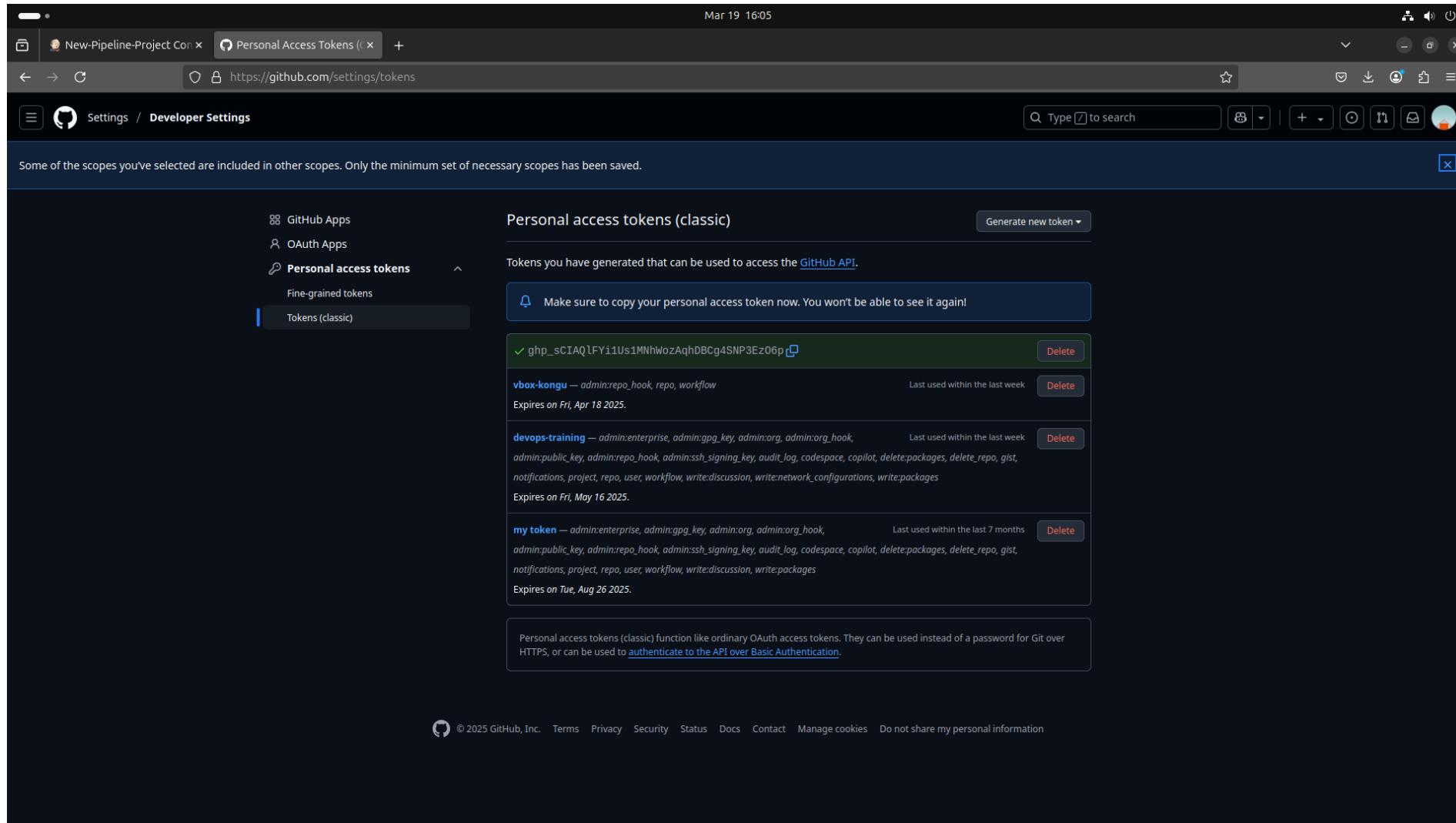
GitHub Personal Access Token Generation 5



The screenshot shows the GitHub settings page for generating a new personal access token. The URL is <https://github.com/settings/tokens/new>. The token name is "macbook-air". The expiration date is set to "30 days (April 18, 2025)". The "Tokens (classic)" scope is selected. The "Select scopes" section lists various OAuth scopes:

Scope	Description
<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input type="checkbox"/> read:org	Read org and team membership, read org projects
<input type="checkbox"/> manage_runners:org	Manage org runners and runner groups
<input type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys
<input checked="" type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input checked="" type="checkbox"/> write:repo_hook	Write repository hooks
<input checked="" type="checkbox"/> read:repo_hook	Read repository hooks

GitHub Personal Access Token Generation 6



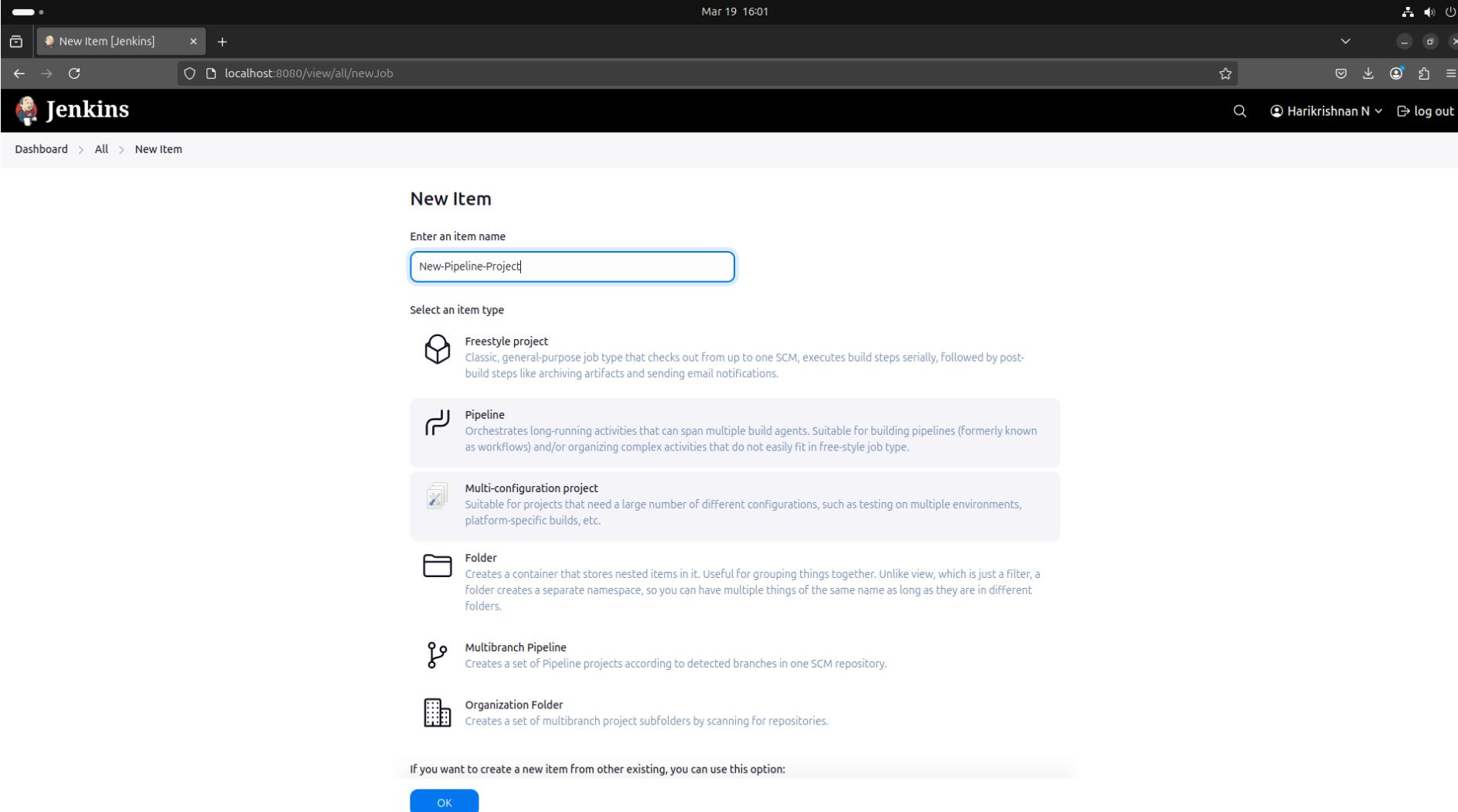
The screenshot shows the GitHub 'Personal Access Tokens' page. The URL is <https://github.com/settings/tokens>. The page title is 'Personal Access Tokens (classic)'. A sidebar on the left lists 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens', with 'Tokens (classic)' selected. A message at the top says, 'Some of the scopes you've selected are included in other scopes. Only the minimum set of necessary scopes has been saved.' A prominent message in a blue box says, 'Make sure to copy your personal access token now. You won't be able to see it again!' Below this, three tokens are listed:

- ghp_sCIAQlFYi1Us1MNhWozAqhDBCg4SNP3E206p** (green checkmark) - Last used within the last week, Expires on Fri, Apr 18 2025.
- vbox-kongu** - admin:repo_hook, repo, workflow - Last used within the last week, Expires on Fri, May 16 2025.
- my token** - admin:enterprise, admin:gpg_key, admin:org, admin:org_hook, admin:public_key, admin:repo_hook, admin:ssh_signing_key, audit_log, codespace, copilot, delete:packages, delete_repo, gist, notifications, project, repo, user, workflow, write:discussion, write:network_configurations, write:packages - Last used within the last 7 months, Expires on Tue, Aug 26 2025.

At the bottom, a note states: 'Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.'

Page footer: © 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

Creating a pipeline project with Jenkins and dockerhub



The screenshot shows the Jenkins 'New Item' creation interface. At the top, the browser title bar reads 'New Item [Jenkins]' and the address bar shows 'localhost:8080/view/all/newJob'. The Jenkins logo is in the top-left corner, and the user 'Harikrishnan N' is logged in.

The main section is titled 'New Item' and has a sub-section 'Enter an item name' with the input field containing 'New-Pipeline-Project'. Below this, there's a 'Select an item type' section with the following options:

- Freestyle project**: Described as a classic, general-purpose job type.
- Pipeline**: Described as orchestrating long-running activities for building pipelines.
- Multi-configuration project**: Described as suitable for projects with many configurations.
- Folder**: Described as creating a container for grouping items.
- Multibranch Pipeline**: Described as creating a set of Pipeline projects based on detected branches.
- Organization Folder**: Described as creating subfolders for multibranch projects by scanning repositories.

At the bottom, a note says 'If you want to create a new item from other existing, you can use this option:' followed by an 'OK' button.

Configuring pipeline script 1

The screenshot shows the Jenkins configuration interface for a new pipeline project. The top navigation bar indicates the URL is `localhost:8080/job/New-Pipeline-Project/configure`. The left sidebar has tabs for General, Triggers, Pipeline (which is selected and highlighted in blue), and Advanced. The main content area is titled "Pipeline" and contains the sub-section "Definition". A dropdown menu is open, showing "Pipeline script" (which is selected and highlighted in blue) and "Pipeline script from SCM". Below the dropdown is a large text input field for the pipeline script. At the bottom of this section is a checkbox labeled "Use Groovy Sandbox" with a checked status. There is also a link to "Pipeline Syntax". The "Advanced" section at the bottom includes a dropdown menu set to "Advanced". At the very bottom are two buttons: "Save" and "Apply".

Configuring pipeline script 2

The screenshot shows the Jenkins Pipeline configuration page for a project named "New-Pipeline-Project". The page is titled "Configure" and displays the "Pipeline" section. The "Definition" dropdown is set to "Pipeline script from SCM". The "SCM" dropdown is set to "Git". Under "Repositories", there is one repository defined with the URL "https://github.com/SysSyncer/jenkins-pipeline-project.git" and no credentials selected. Under "Branches to build", the branch specifier is set to "*/*master". At the bottom of the page are "Save" and "Apply" buttons.

Configure

Pipeline

Define your Pipeline using Groovy directly or pull it from source control.

Definition

Pipeline script from SCM

SCM

Git

Repositories

Repository URL

https://github.com/SysSyncer/jenkins-pipeline-project.git

Credentials

- none -

Jenkins Credentials Provider

Jenkins

Add Repository

Branches to build

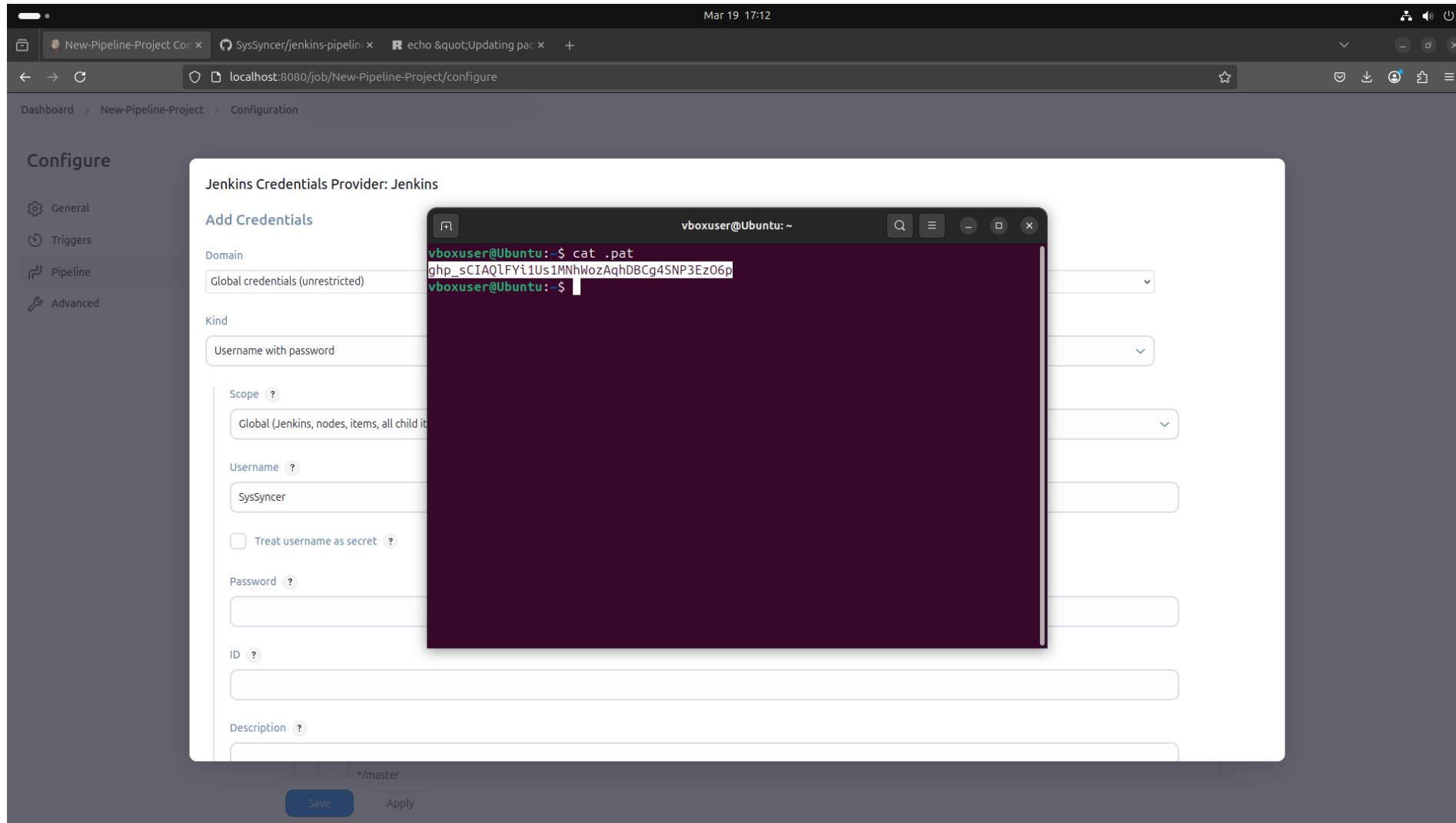
Branch Specifier (blank for 'any')

*/master

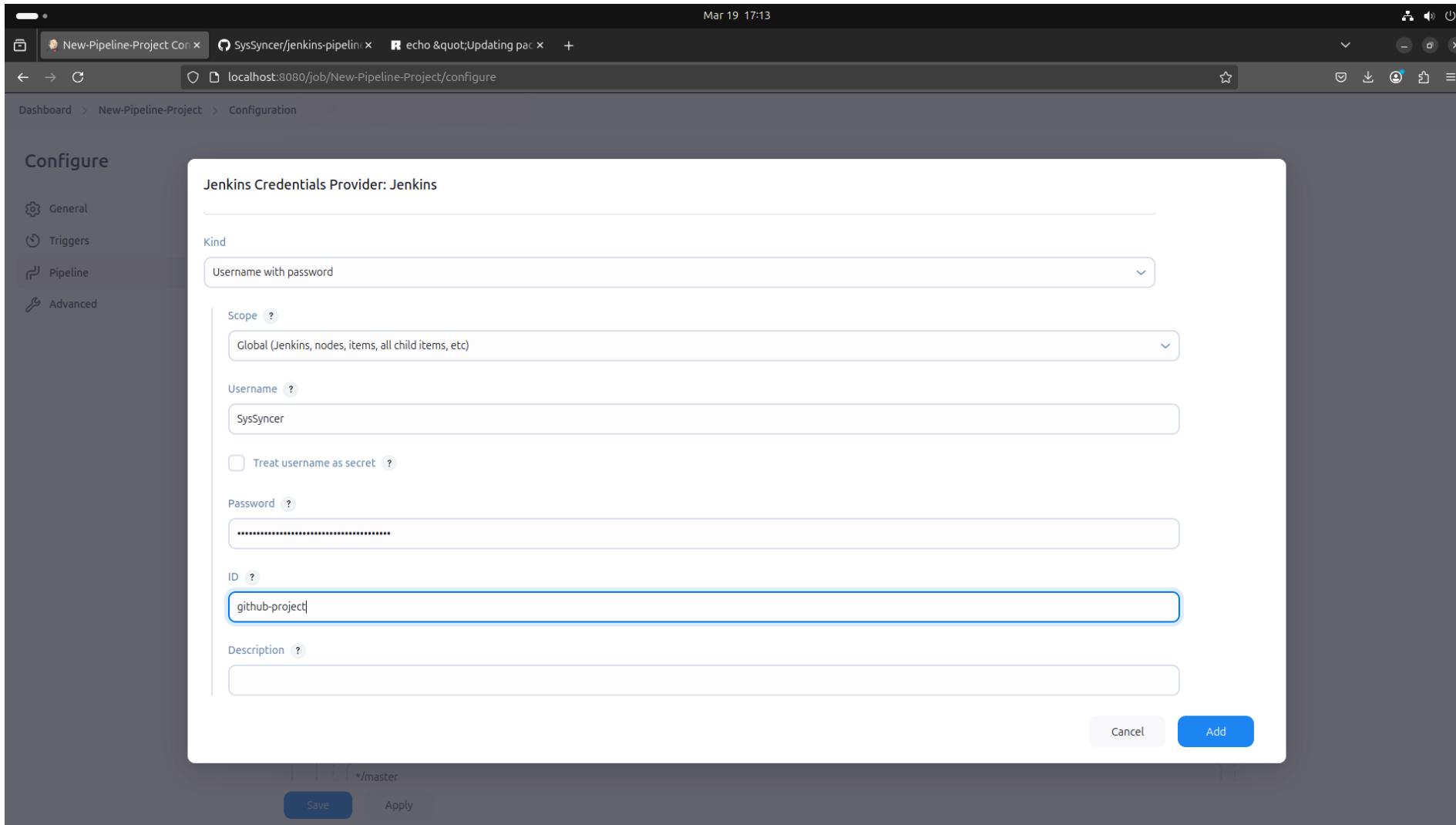
Save

Apply

Configuring pipeline script 3



Configuring pipeline script 4



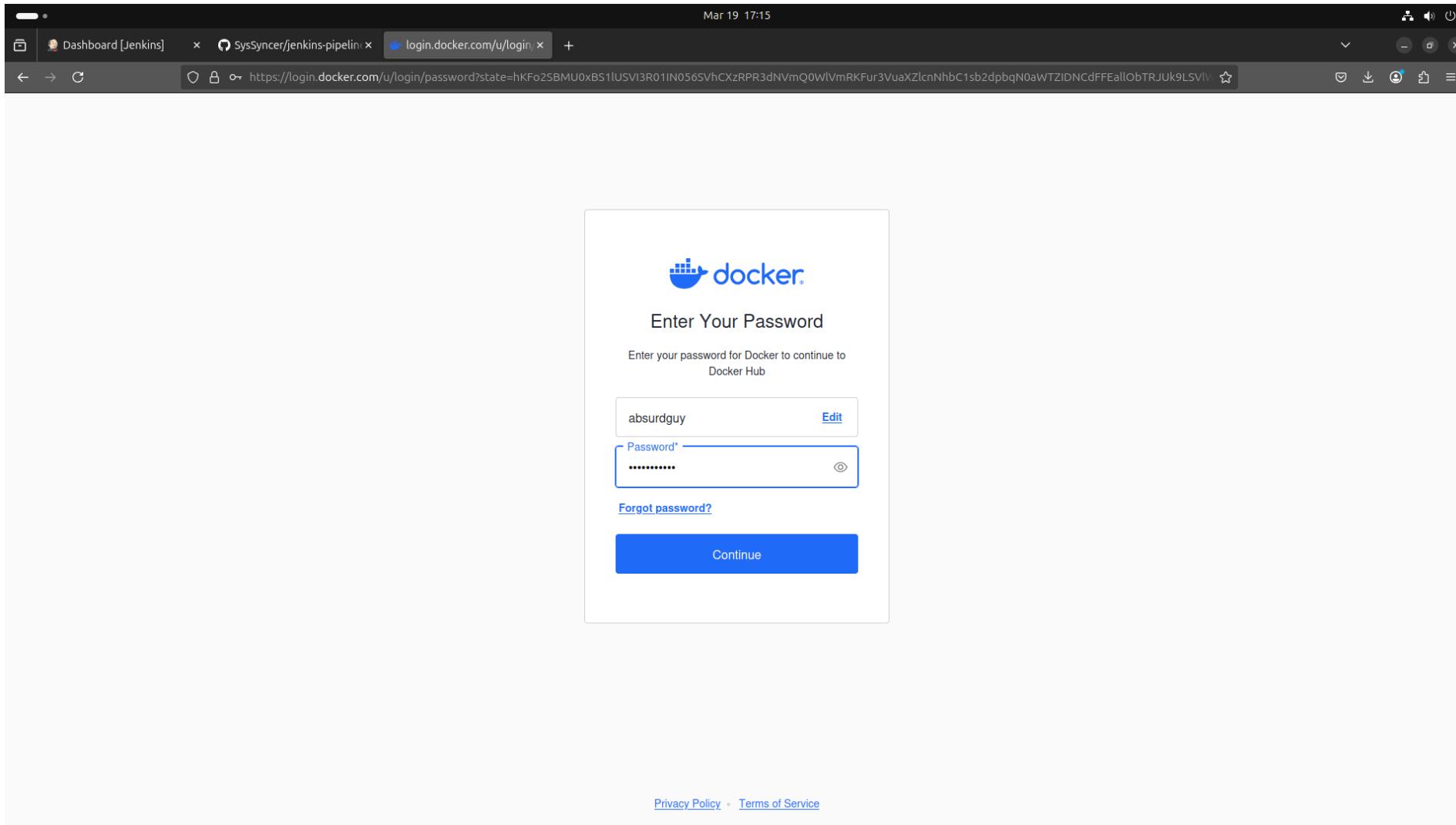
Configuring pipeline script 5

The screenshot shows the Jenkins Pipeline configuration page for the 'New-Pipeline-Project' job. The URL in the browser is `localhost:8080/job/New-Pipeline-Project/configure`. The left sidebar has tabs for 'General', 'Triggers', 'Pipeline' (which is selected), and 'Advanced'. The main configuration area includes:

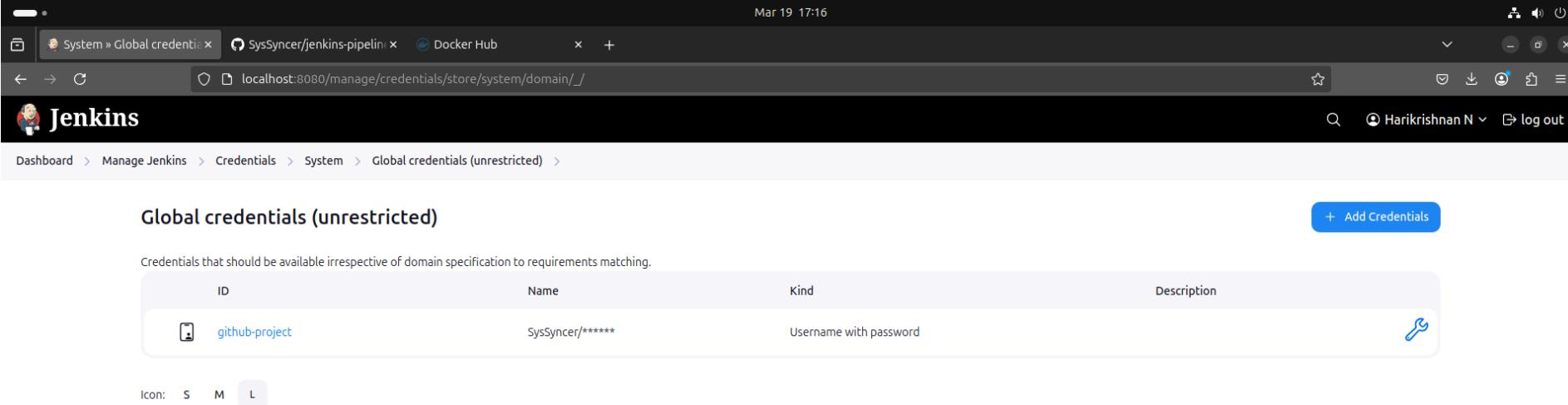
- Credentials:** A dropdown menu showing '-none-'.
- Add Repository:** A button to add a new repository.
- Branches to build:** A section with a text input field containing 'main'.
- Additional Behaviours:** A dropdown menu showing '(Auto)'.
- Script Path:** A text input field containing 'Jenkinsfile'.

At the bottom are 'Save' and 'Apply' buttons.

SignIn with docker hub



Add docker credentials in global credentials



The screenshot shows the Jenkins Global credentials (unrestricted) page. The URL in the browser is `localhost:8080/manage/credentials/store/system/domain/_/newCredentials`. The page title is "Global credentials (unrestricted)". A blue button at the top right says "+ Add Credentials". Below the title, a sub-instruction reads "Credentials that should be available irrespective of domain specification to requirements matching." A table lists one credential:

ID	Name	Kind	Description
 github-project	SysSyncer/*****	Username with password	

At the bottom left, there are icons for "Icon: S M L". At the bottom right, links for "REST API" and "Jenkins 2.492.2" are visible.

Adding docker credentials

The screenshot shows the Jenkins 'New credentials' configuration page. The URL in the browser is `localhost:8080/manage/credentials/store/system/domain/_/newCredentials`. The page title is 'New credentials'. The 'Kind' dropdown is set to 'Username with password'. The 'Scope' dropdown is set to 'Global (Jenkins, nodes, items, all child items, etc.)'. The 'Username' field contains 'absurdguy'. The 'Password' field contains a redacted password. The 'ID' field contains 'docker-hub-project'. The 'Description' field is empty. A blue 'Create' button is at the bottom left.

New credentials

Kind

Username with password

Scope

Global (Jenkins, nodes, items, all child items, etc.)

Username

absurdguy

Treat username as secret

Password

.....

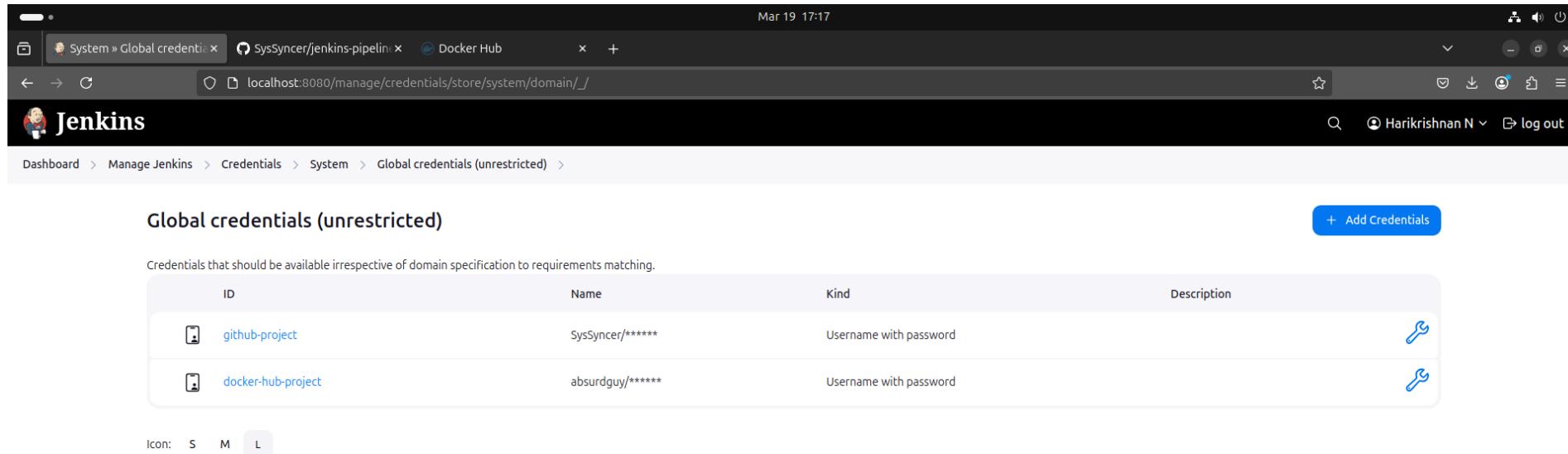
ID

docker-hub-project

Description

Create

Global credentials

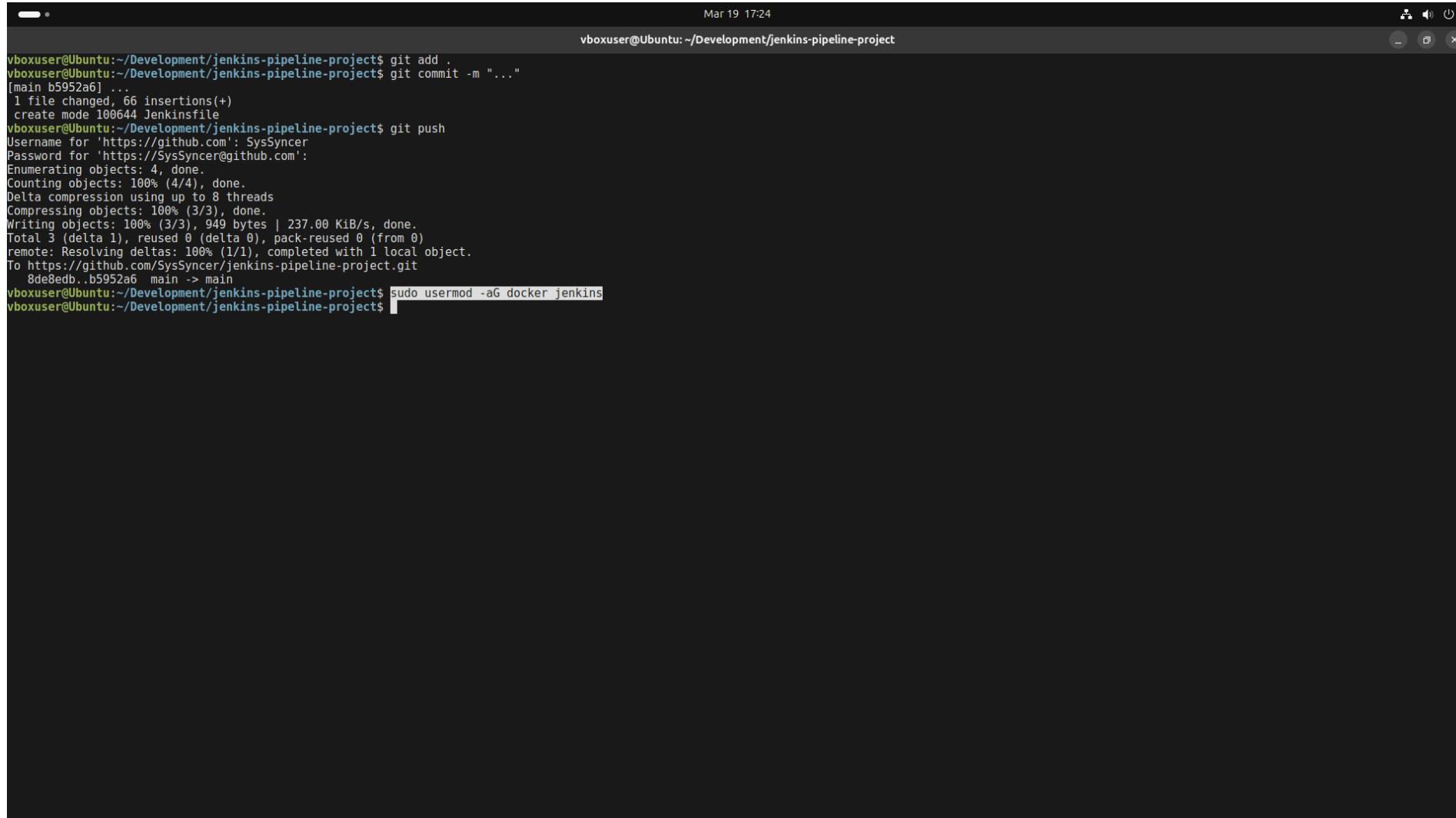


The screenshot shows the Jenkins Global credentials (unrestricted) page. The browser title bar displays "System » Global credentials" and "localhost:8080/manage/credentials/store/system/domain/_/". The Jenkins header includes the logo, user name "Harikrishnan N", and "log out". The page title is "Global credentials (unrestricted)". A blue button at the top right says "+ Add Credentials". Below the title, a message states "Credentials that should be available irrespective of domain specification to requirements matching." A table lists two credentials:

ID	Name	Kind	Description
github-project	SysSyncer/*****	Username with password	
docker-hub-project	absurdguy/*****	Username with password	

At the bottom left, there is a "Icon:" label followed by size options "S" (selected), "M", and "L".

Appending Jenkins and docker in groups



A screenshot of a terminal window titled "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project". The window shows a command-line session:

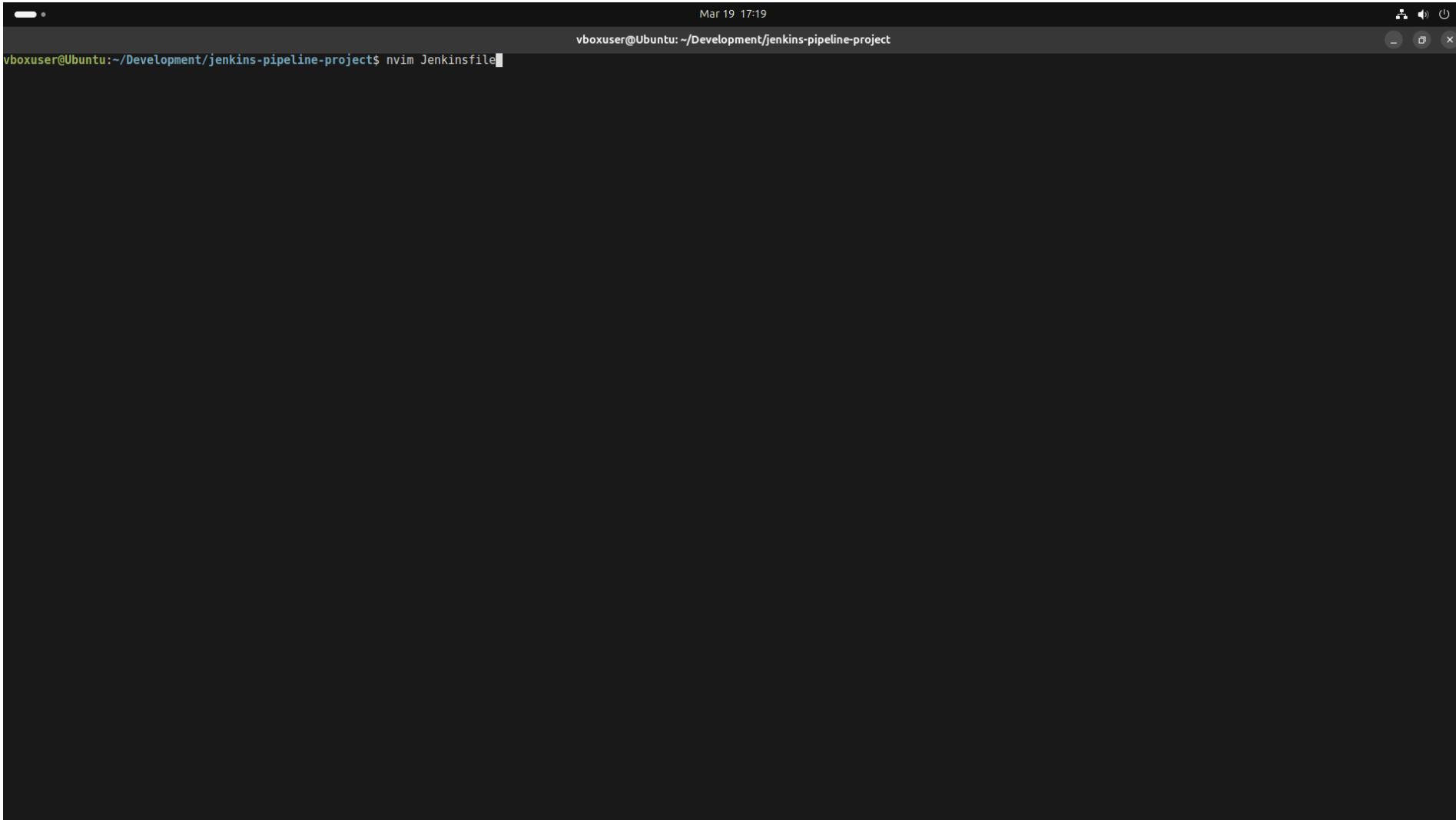
```
Mar 19 17:24
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ git add .
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ git commit -m "..."
[main b5952a6] ...
1 file changed, 66 insertions(+)
create mode 100644 Jenkinsfile
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ git push
Username for 'https://github.com': SysSyncer
Password for 'https://SysSyncer@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 949 bytes | 237.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/SysSyncer/jenkins-pipeline-project.git
 8de8edb..b5952a6 main -> main
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ sudo usermod -aG docker jenkins
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$
```

Restarting Jenkins services

The screenshot shows a terminal window on a Linux system (Ubuntu) with the following session history:

```
Mar 19 17:25
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ git add .
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ git commit -m "..."
[main b5952a6] ...
1 file changed, 66 insertions(+)
create mode 100644 Jenkinsfile
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ git push
Username for 'https://github.com': SysSyncer
Password for 'https://SysSyncer@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 949 bytes | 237.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/SysSyncer/jenkins-pipeline-project.git
 8de8edb..b5952a6 main -> main
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ sudo usermod -aG docker jenkins
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ sudo systemctl restart jenkins
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$
```

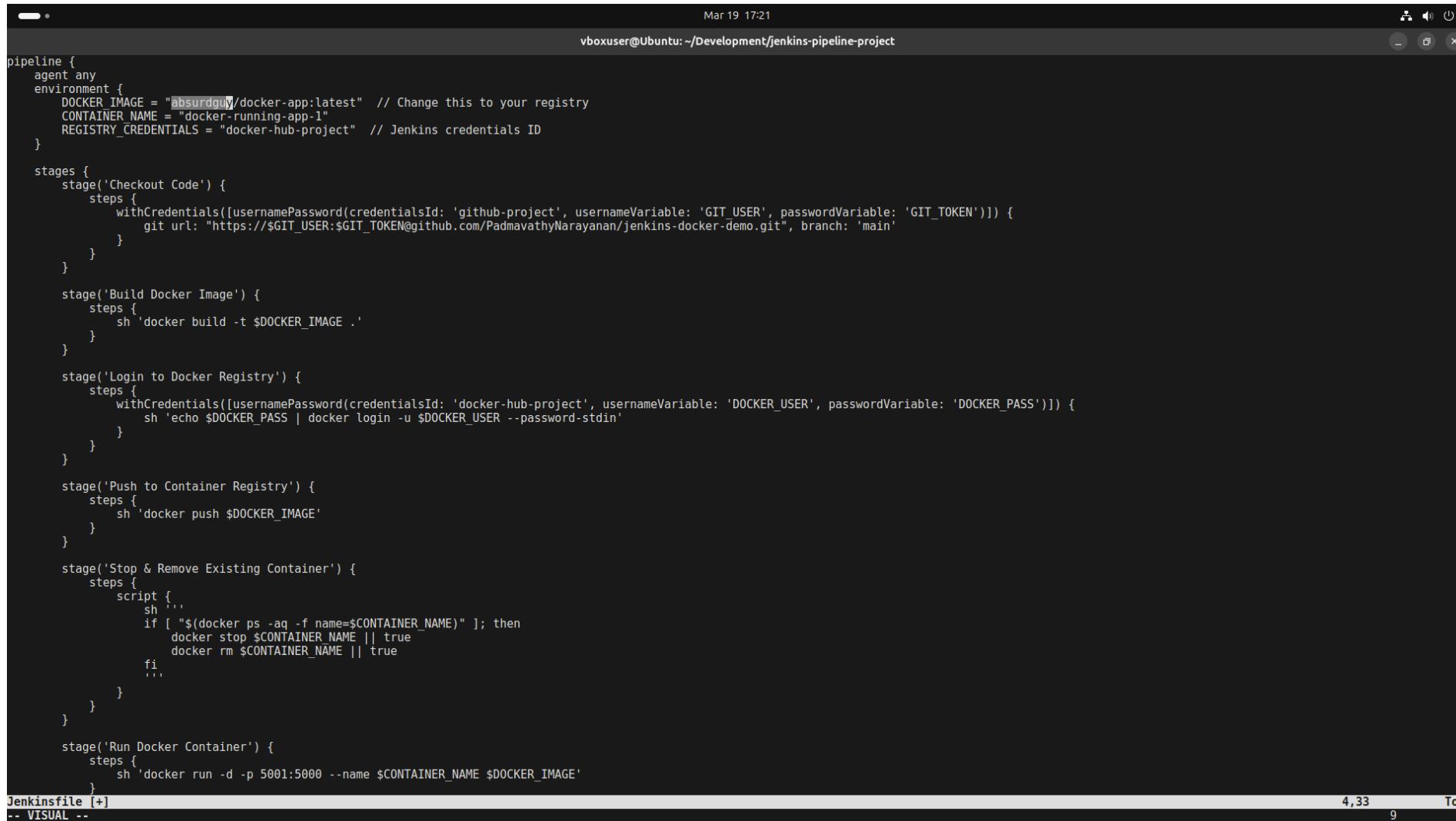
Creating Jenkinsfile to our local repo



A screenshot of a terminal window titled "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project". The window shows the command "nvim Jenkinsfile" being run at the prompt. The terminal has a dark background and light-colored text. The title bar includes the date and time "Mar 19 17:19".

```
Mar 19 17:19
vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project$ nvim Jenkinsfile
```

Adding Jenkinsfile configuration 1



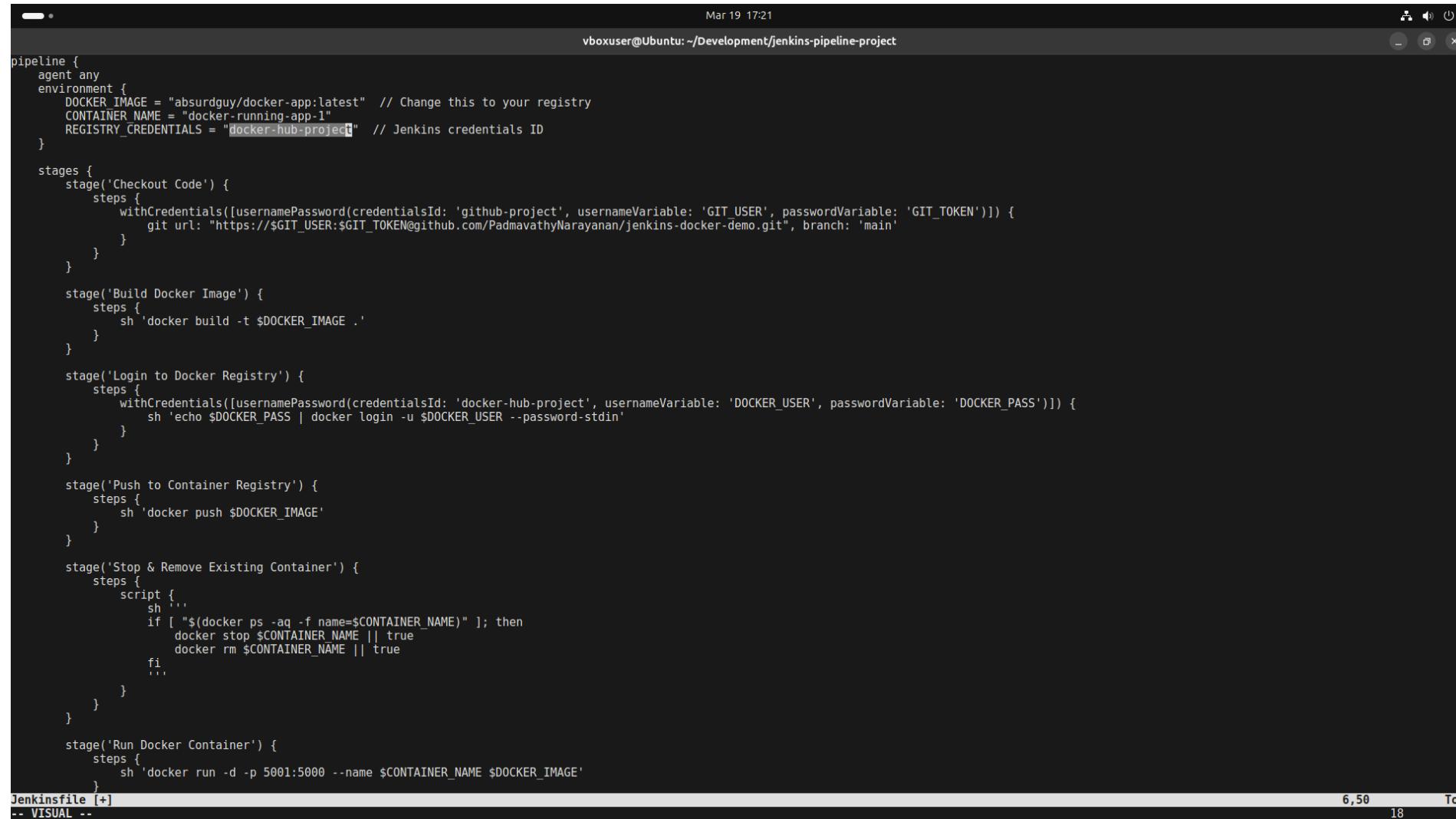
The screenshot shows a terminal window with the following details:

- Timestamp: Mar 19 17:21
- User: vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project
- Content: A Jenkins pipeline script (Jenkinsfile) for a Docker pipeline.

```
pipeline {
    agent any
    environment {
        DOCKER_IMAGE = "absurdguy/docker-app:latest" // Change this to your registry
        CONTAINER_NAME = "docker-running-app-1"
        REGISTRY_CREDENTIALS = "docker-hub-project" // Jenkins credentials ID
    }
    stages {
        stage('Checkout Code') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'github-project', usernameVariable: 'GIT_USER', passwordVariable: 'GIT_TOKEN')]) {
                    git url: "https://$GIT_USER:$GIT_TOKEN@github.com/PadmavathyNarayanan/jenkins-docker-demo.git", branch: 'main'
                }
            }
        }
        stage('Build Docker Image') {
            steps {
                sh 'docker build -t $DOCKER_IMAGE .'
            }
        }
        stage('Login to Docker Registry') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'docker-hub-project', usernameVariable: 'DOCKER_USER', passwordVariable: 'DOCKER_PASS')]) {
                    sh 'echo $DOCKER_PASS | docker login -u $DOCKER_USER --password-stdin'
                }
            }
        }
        stage('Push to Container Registry') {
            steps {
                sh 'docker push $DOCKER_IMAGE'
            }
        }
        stage('Stop & Remove Existing Container') {
            steps {
                script {
                    sh '''
                        if [ "$(docker ps -aq -f name=$CONTAINER_NAME)" ]; then
                            docker stop $CONTAINER_NAME || true
                            docker rm $CONTAINER_NAME || true
                        fi
                    '''
                }
            }
        }
        stage('Run Docker Container') {
            steps {
                sh 'docker run -d -p 5001:5000 --name $CONTAINER_NAME $DOCKER_IMAGE'
            }
        }
    }
}
```

Jenkinsfile [+] 4,33 Top
-- VISUAL -- 9

Adding Jenkinsfile configuration 2



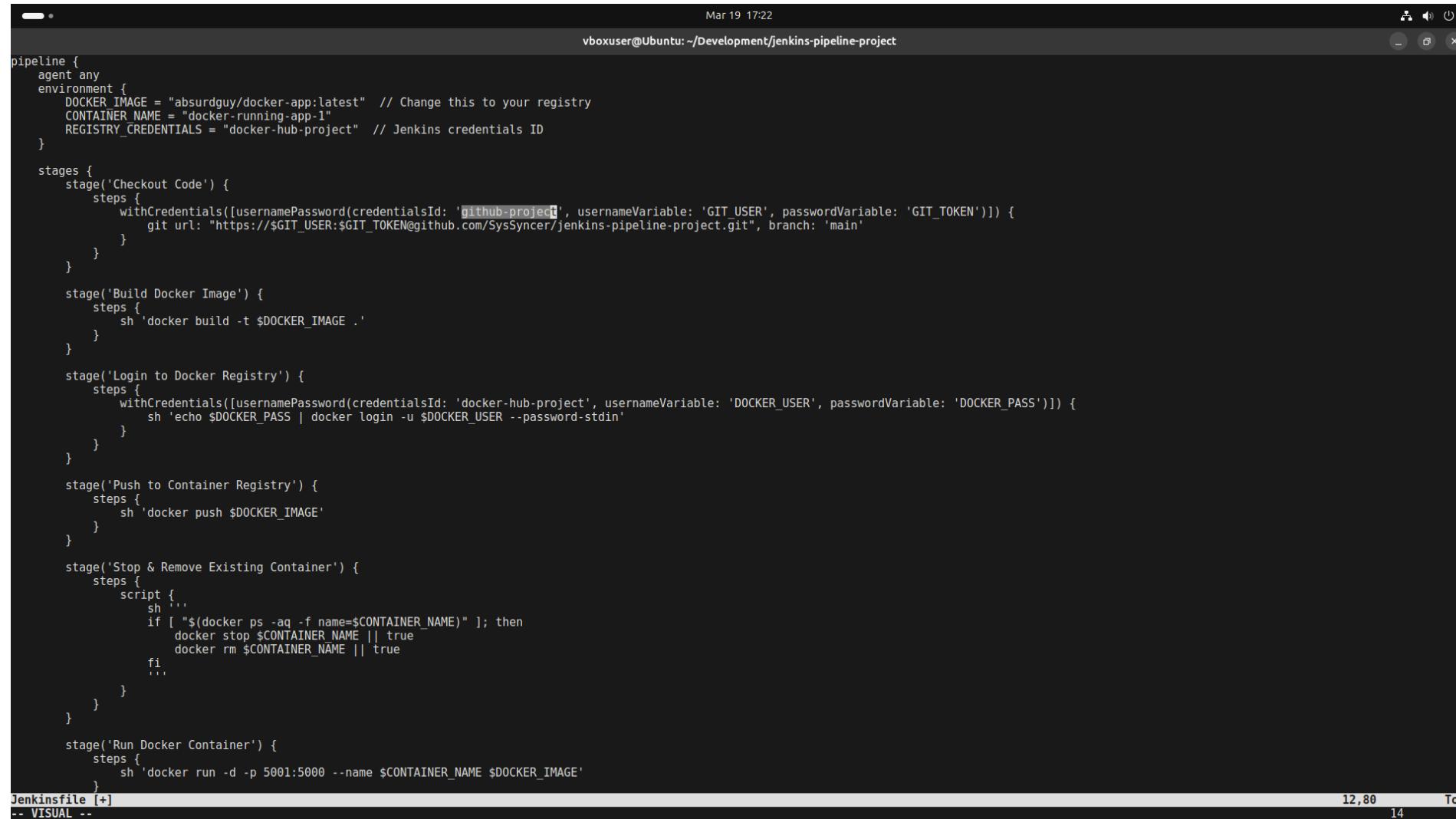
The screenshot shows a terminal window titled "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project" with a timestamp of "Mar 19 17:21". The terminal displays a Jenkins pipeline script. The script defines an environment with variables for Docker image, container name, and registry credentials. It includes stages for checkout, building a Docker image, logging into the Docker registry, pushing the image, stopping and removing existing containers, and running a new Docker container.

```
pipeline {
    agent any
    environment {
        DOCKER_IMAGE = "absurdguy/docker-app:latest" // Change this to your registry
        CONTAINER_NAME = "docker-running-app-1"
        REGISTRY_CREDENTIALS = "docker-hub-project" // Jenkins credentials ID
    }
    stages {
        stage('Checkout Code') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'github-project', usernameVariable: 'GIT_USER', passwordVariable: 'GIT_TOKEN')]) {
                    git url: "https://$GIT_USER:$GIT_TOKEN@github.com/PadmavathyNarayanan/jenkins-demo.git", branch: 'main'
                }
            }
        }
        stage('Build Docker Image') {
            steps {
                sh 'docker build -t $DOCKER_IMAGE .'
            }
        }
        stage('Login to Docker Registry') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'docker-hub-project', usernameVariable: 'DOCKER_USER', passwordVariable: 'DOCKER_PASS')]) {
                    sh 'echo $DOCKER_PASS | docker login -u $DOCKER_USER --password-stdin'
                }
            }
        }
        stage('Push to Container Registry') {
            steps {
                sh 'docker push $DOCKER_IMAGE'
            }
        }
        stage('Stop & Remove Existing Container') {
            steps {
                script {
                    sh '''
                        if [ "$(docker ps -aq -f name=$CONTAINER_NAME)" ]; then
                            docker stop $CONTAINER_NAME || true
                            docker rm $CONTAINER_NAME || true
                        fi
                    '''
                }
            }
        }
        stage('Run Docker Container') {
            steps {
                sh 'docker run -d -p 5001:5000 --name $CONTAINER_NAME $DOCKER_IMAGE'
            }
        }
    }
}
```

Jenkinsfile [+] -- VISUAL --

6,50 Top 18

Adding Jenkinsfile configuration 3

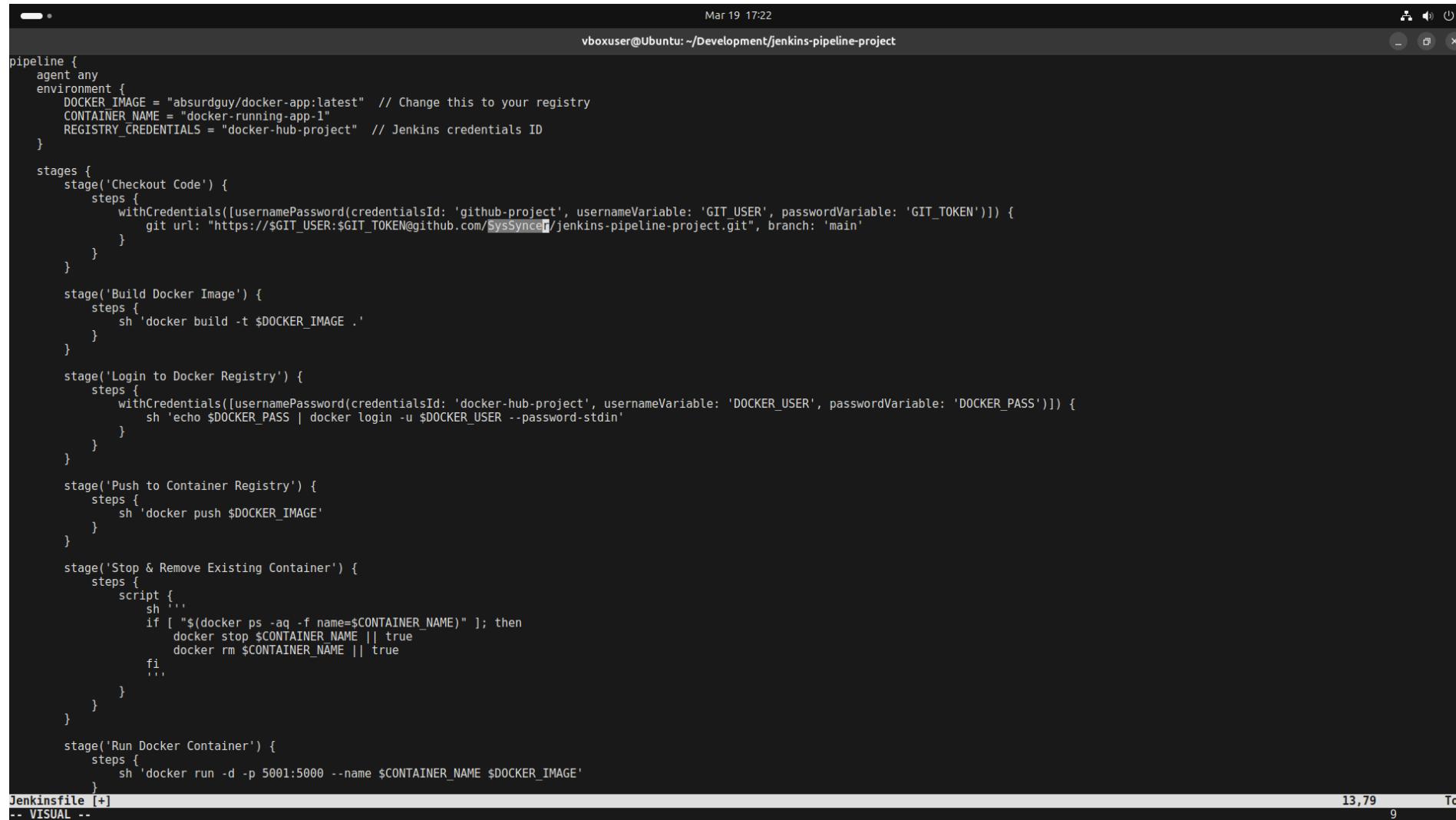


The screenshot shows a terminal window with the following details:

- Terminal title: `vboxuser@Ubuntu:~/Development/jenkins-pipeline-project`
- Date and time: `Mar 19 17:22`
- File path: `Jenkinsfile [+]`
- Page numbers: `12,80 Top 14`
- Content: A Jenkins pipeline script (Jenkinsfile) with several stages: Checkout Code, Build Docker Image, Login to Docker Registry, Push to Container Registry, Stop & Remove Existing Container, and Run Docker Container. It uses Docker images, git credentials, and docker commands like build, push, stop, and rm.

```
pipeline {
    agent any
    environment {
        DOCKER_IMAGE = "absurdguy/docker-app:latest" // Change this to your registry
        CONTAINER_NAME = "docker-running-app-1"
        REGISTRY_CREDENTIALS = "docker-hub-project" // Jenkins credentials ID
    }
    stages {
        stage('Checkout Code') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'github-project', usernameVariable: 'GIT_USER', passwordVariable: 'GIT_TOKEN')]) {
                    git url: "https://$GIT_USER:$GIT_TOKEN@github.com/SysSyncer/jenkins-pipeline-project.git", branch: 'main'
                }
            }
        }
        stage('Build Docker Image') {
            steps {
                sh 'docker build -t $DOCKER_IMAGE .'
            }
        }
        stage('Login to Docker Registry') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'docker-hub-project', usernameVariable: 'DOCKER_USER', passwordVariable: 'DOCKER_PASS')]) {
                    sh 'echo $DOCKER_PASS | docker login -u $DOCKER_USER --password-stdin'
                }
            }
        }
        stage('Push to Container Registry') {
            steps {
                sh 'docker push $DOCKER_IMAGE'
            }
        }
        stage('Stop & Remove Existing Container') {
            steps {
                script {
                    sh '''
                        if [ "$(docker ps -aq -f name=$CONTAINER_NAME)" ]; then
                            docker stop $CONTAINER_NAME || true
                            docker rm $CONTAINER_NAME || true
                        fi
                    '''
                }
            }
        }
        stage('Run Docker Container') {
            steps {
                sh 'docker run -d -p 5001:5000 --name $CONTAINER_NAME $DOCKER_IMAGE'
            }
        }
    }
}
```

Adding Jenkinsfile configuration 4



The screenshot shows a terminal window titled "vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project". The terminal displays a Jenkins pipeline script named "Jenkinsfile". The script defines a pipeline with several stages: Checkout Code, Build Docker Image, Login to Docker Registry, Push to Container Registry, Stop & Remove Existing Container, and Run Docker Container. It uses Docker credentials and environment variables like DOCKER_IMAGE, CONTAINER_NAME, and REGISTRY_CREDENTIALS. The terminal interface includes standard window controls (minimize, maximize, close) and a status bar at the bottom.

```
Mar 19 17:22
vboxuser@Ubuntu: ~/Development/jenkins-pipeline-project

pipeline {
    agent any
    environment {
        DOCKER_IMAGE = "absurdguy/docker-app:latest" // Change this to your registry
        CONTAINER_NAME = "docker-running-app-1"
        REGISTRY_CREDENTIALS = "docker-hub-project" // Jenkins credentials ID
    }
    stages {
        stage('Checkout Code') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'github-project', usernameVariable: 'GIT_USER', passwordVariable: 'GIT_TOKEN')]) {
                    git url: "https://$GIT_USER:$GIT_TOKEN@github.com/sysSync1/jenkins-pipeline-project.git", branch: 'main'
                }
            }
        }
        stage('Build Docker Image') {
            steps {
                sh 'docker build -t $DOCKER_IMAGE .'
            }
        }
        stage('Login to Docker Registry') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'docker-hub-project', usernameVariable: 'DOCKER_USER', passwordVariable: 'DOCKER_PASS')]) {
                    sh 'echo $DOCKER_PASS | docker login -u $DOCKER_USER --password-stdin'
                }
            }
        }
        stage('Push to Container Registry') {
            steps {
                sh 'docker push $DOCKER_IMAGE'
            }
        }
        stage('Stop & Remove Existing Container') {
            steps {
                script {
                    sh '''
                        if [ "$(docker ps -aq -f name=$CONTAINER_NAME)" ]; then
                            docker stop $CONTAINER_NAME || true
                            docker rm $CONTAINER_NAME || true
                        fi
                    '''
                }
            }
        }
        stage('Run Docker Container') {
            steps {
                sh 'docker run -d -p 5001:5000 --name $CONTAINER_NAME $DOCKER_IMAGE'
            }
        }
    }
}

Jenkinsfile [+]
-- VISUAL --
13,79      Top
9
```

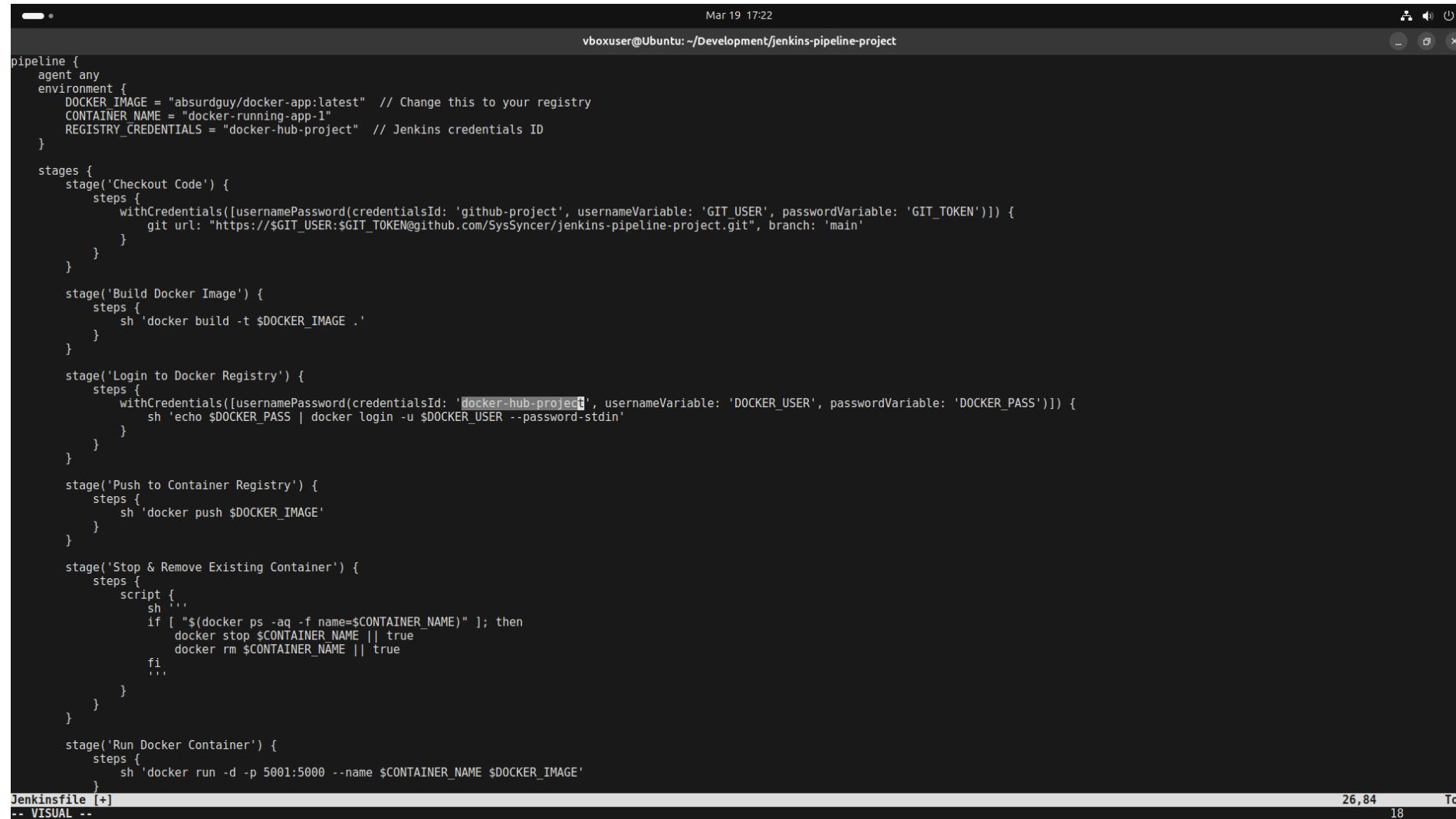
Adding Jenkinsfile configuration 5

```
Mar 19 17:22
vboxuser@Ubuntu:~/Development/jenkins-pipeline-project

pipeline {
    agent any
    environment {
        DOCKER_IMAGE = "absurdguy/docker-app:latest" // Change this to your registry
        CONTAINER_NAME = "docker-running-app-1"
        REGISTRY_CREDENTIALS = "docker-hub-project" // Jenkins credentials ID
    }
    stages {
        stage('Checkout Code') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'github-project', usernameVariable: 'GIT_USER', passwordVariable: 'GIT_TOKEN')]) {
                    git url: "https://$GIT_USER:$GIT_TOKEN@github.com/SysSyncer/Jenkins-pipeline-project.git", branch: 'main'
                }
            }
        }
        stage('Build Docker Image') {
            steps {
                sh 'docker build -t $DOCKER_IMAGE .'
            }
        }
        stage('Login to Docker Registry') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'docker-hub-project', usernameVariable: 'DOCKER_USER', passwordVariable: 'DOCKER_PASS')]) {
                    sh 'echo $DOCKER_PASS | docker login -u $DOCKER_USER --password-stdin'
                }
            }
        }
        stage('Push to Container Registry') {
            steps {
                sh 'docker push $DOCKER_IMAGE'
            }
        }
        stage('Stop & Remove Existing Container') {
            steps {
                script {
                    sh '''
                        if [ "$(docker ps -aq -f name=$CONTAINER_NAME)" ]; then
                            docker stop $CONTAINER_NAME || true
                            docker rm $CONTAINER_NAME || true
                        fi
                    '''
                }
            }
        }
        stage('Run Docker Container') {
            steps {
                sh 'docker run -d -p 5001:5000 --name $CONTAINER_NAME $DOCKER_IMAGE'
            }
        }
    }
}

Jenkinsfile [+]
-- VISUAL --
```

Adding Jenkinsfile configuration 6



The screenshot shows a terminal window with the following details:

- Terminal title: Mar 19 17:22
- Terminal prompt: vboxuser@Ubuntu:~/Development/jenkins-pipeline-project
- Content: A Jenkins pipeline script (Jenkinsfile) with syntax highlighting.

```
pipeline {
    agent any
    environment {
        DOCKER_IMAGE = "absurdguy/docker-app:latest" // Change this to your registry
        CONTAINER_NAME = "docker-running-app-1"
        REGISTRY_CREDENTIALS = "docker-hub-project" // Jenkins credentials ID
    }
    stages {
        stage('Checkout Code') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'github-project', usernameVariable: 'GIT_USER', passwordVariable: 'GIT_TOKEN')]) {
                    git url: "https://$GIT_USER:$GIT_TOKEN@github.com/SysSyncer/jenkins-pipeline-project.git", branch: 'main'
                }
            }
        }
        stage('Build Docker Image') {
            steps {
                sh 'docker build -t $DOCKER_IMAGE .'
            }
        }
        stage('Login to Docker Registry') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'docker-hub-project', usernameVariable: 'DOCKER_USER', passwordVariable: 'DOCKER_PASS')]) {
                    sh 'echo $DOCKER_PASS | docker login -u $DOCKER_USER --password-stdin'
                }
            }
        }
        stage('Push to Container Registry') {
            steps {
                sh 'docker push $DOCKER_IMAGE'
            }
        }
        stage('Stop & Remove Existing Container') {
            steps {
                script {
                    sh '''
                        if [ "$(docker ps -aq -f name=$CONTAINER_NAME)" ]; then
                            docker stop $CONTAINER_NAME || true
                            docker rm $CONTAINER_NAME || true
                        fi
                    '''
                }
            }
        }
        stage('Run Docker Container') {
            steps {
                sh 'docker run -d -p 5001:5000 --name $CONTAINER_NAME $DOCKER_IMAGE'
            }
        }
    }
}
```

Bottom status bar: Jenkinsfile [+] 26,84 Top
-- VISUAL -- 18

Jenkins Dashboard

The screenshot shows the Jenkins dashboard interface. At the top, there are three tabs: 'Dashboard [Jenkins]', 'SysSyncer/jenkins-pipeline', and 'Docker Hub'. The main header includes the date 'Mar 19 17:16' and user information 'Harikrishnan N log out'. On the left, a sidebar contains links for 'New Item', 'Build History', 'Manage Jenkins' (which is highlighted), and 'My Views'. Below the sidebar are sections for 'Build Queue' (empty) and 'Build Executor Status' (0/2). The main content area displays a table of projects:

S	W	Name	Last Success	Last Failure	Last Duration
Green checkmark	Cloud icon	New-Freestyle-Project	1 hr 17 min #2	N/A	43 sec
Ellipsis icon	Sun icon	New-Pipeline-Project	N/A	N/A	N/A

At the bottom right, there are navigation icons for 'S' (Small), 'M' (Medium), and 'L' (Large). The footer includes links for 'REST API' and 'Jenkins 2.492.2', and a URL 'localhost:8080/manage'.

build now - New-Pipeline-project

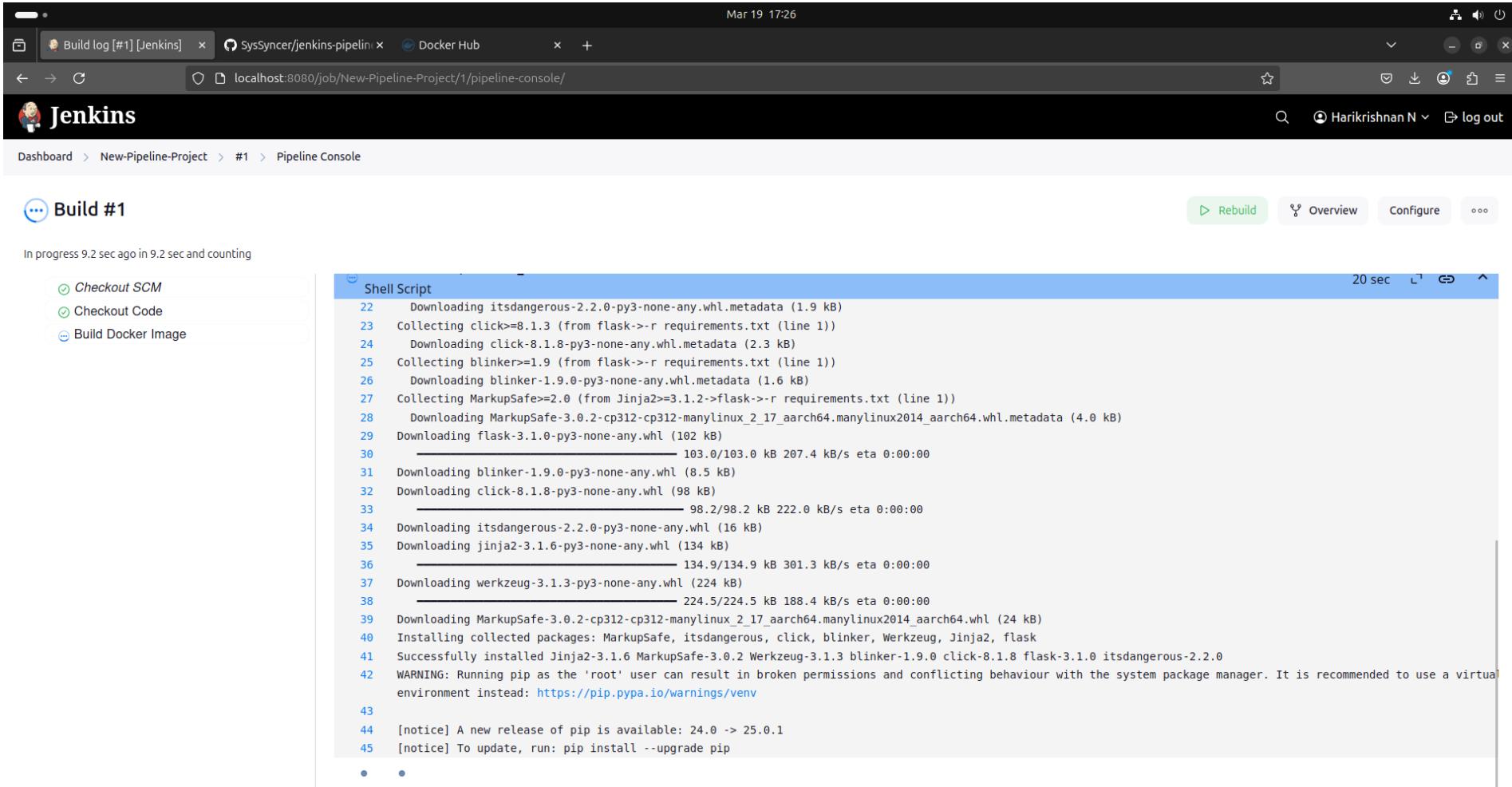
The screenshot shows a web browser window with the Jenkins interface. The title bar indicates the page is 'localhost:8080/job/New-Pipeline-Project/'. The top navigation bar includes tabs for 'New-Pipeline-Project [Jen]', 'SysSyncer/jenkins-pipel', and 'Docker Hub'. The user 'Harikrishnan N' is logged in.

The main content area displays the 'New-Pipeline-Project' job details. On the left, a sidebar lists actions: Status (selected), Changes, Build Now, Configure, Delete Pipeline, Stages, Rename, and Pipeline Syntax. The 'Build Now' button is highlighted with a red border.

The central part of the screen shows the 'Builds' section, which currently displays 'No builds'.

At the bottom, a green button labeled 'Build scheduled' with a checkmark icon is visible. The footer of the page shows links for 'REST API' and 'Jenkins 2.492.2'.

Building New-Pipeline-project script



The screenshot shows a Jenkins Pipeline Console for a job named "New-Pipeline-Project" with build number #1. The build is currently in progress, having started 9.2 seconds ago and continuing to count. On the left, there's a sidebar with three items: "Checkout SCM", "Checkout Code", and "Build Docker Image". The main area is titled "Shell Script" and displays the following log output:

```
22  Downloading itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
23  Collecting click==8.1.3 (from flask->r requirements.txt (line 1))
24  Downloading click-8.1.8-py3-none-any.whl.metadata (2.3 kB)
25  Collecting blinker==1.9 (from flask->r requirements.txt (line 1))
26  Downloading blinker-1.9.0-py3-none-any.whl.metadata (1.6 kB)
27  Collecting MarkupSafe==2.0 (from Jinja2>=3.1.2->flask->r requirements.txt (line 1))
28  Downloading MarkupSafe-3.0.2-cp312-cp312-manylinux_2_17_aarch64_manylinux2014_aarch64.whl.metadata (4.0 kB)
29  Downloading flask-3.1.0-py3-none-any.whl (102 kB)
30  ━━━━━━━━━━━━━━━━ 103.0/103.0 kB 207.4 kB/s eta 0:00:00
31  Downloading blinker-1.9.0-py3-none-any.whl (8.5 kB)
32  Downloading click-8.1.8-py3-none-any.whl (98 kB)
33  ━━━━━━━━━━━━━━ 98.2/98.2 kB 222.0 kB/s eta 0:00:00
34  Downloading itsdangerous-2.2.0-py3-none-any.whl (16 kB)
35  Downloading jinja2-3.1.6-py3-none-any.whl (134 kB)
36  ━━━━━━━━━━━━━━ 134.9/134.9 kB 301.3 kB/s eta 0:00:00
37  Downloading werkzeug-3.1.3-py3-none-any.whl (224 kB)
38  ━━━━━━━━━━━━━━ 224.5/224.5 kB 188.4 kB/s eta 0:00:00
39  Downloading MarkupSafe-3.0.2-cp312-cp312-manylinux_2_17_aarch64_manylinux2014_aarch64.whl (24 kB)
40  Installing collected packages: MarkupSafe, itsdangerous, click, blinker, Werkzeug, Jinja2, flask
41  Successfully installed Jinja2-3.1.6 MarkupSafe-3.0.2 Werkzeug-3.1.3 blinker-1.9.0 click-8.1.8 flask-3.1.0 itsdangerous-2.2.0
42  WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
43
44  [notice] A new release of pip is available: 24.0 -> 25.0.1
45  [notice] To update, run: pip install --upgrade pip
```

At the bottom of the log, there are two dots indicating more content.

Build Status

The screenshot shows a Jenkins Pipeline Console for a job named "New-Pipeline-Project". The build number is #1, and it is currently in progress, having been started 9.2 seconds ago. The pipeline stages listed on the left are:

- Checkout SCM
- Checkout Code
- Build Docker Image
- Login to Docker Registry
- Push to Container Registry
- Stop & Remove Existing Container
- Run Docker Container
- Post Actions** (highlighted with a yellow background)

The "Post Actions" stage details are as follows:

- Started 0.87 sec ago
- Queued 0 ms
- Took 0.17 sec
- Success
- Running on Jenkins
- [View as plain text](#)

A green message bar at the bottom of the stage summary states: "Build, push, and container execution successful!". Below this message, there is a "Print Message" link and a timestamp of 46 ms.

Build Completed

The screenshot shows a Jenkins build completed page for a project named "New-Pipeline-Project". The build number is #1, which was started on Mar 19, 2025, at 5:26:02 PM. The build duration was 1 min 54 sec. The revision is b5952a6bc3cefc512ca51d1a6dd3ab34de3d0408, from the repository https://github.com/SysSyncer/jenkins-pipeline-project.git. The build log indicates no changes were made. A warning message is present about insecure interpolation of sensitive variables.

Mar 19 17:28

New-Pipeline-Project #1 | SysSyncer/jenkins-pipeline-project | Docker Hub

localhost:8080/job/New-Pipeline-Project/1/

Jenkins

Dashboard > New-Pipeline-Project > #1

Status #1 (Mar 19, 2025, 5:26:02 PM) Add description Keep this build forever

Changes Started by user Harikrishnan N Started 2 min 5 sec ago Took 1 min 54 sec

Console Output Edit Build Information Delete build '#1' Timings

Git Build Data Revision: b5952a6bc3cefc512ca51d1a6dd3ab34de3d0408 Repository: https://github.com/SysSyncer/jenkins-pipeline-project.git

Git Build Data Revision: b5952a6bc3cefc512ca51d1a6dd3ab34de3d0408 Repository: https://SysSyncer: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@github.com/SysSyncer/jenkins-pipeline-project.git

Pipeline Overview Pipeline Console

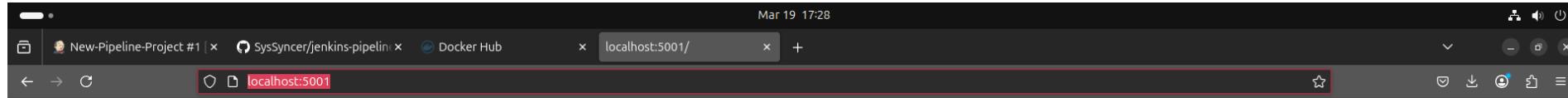
Restart from Stage Replay Pipeline Steps Workspaces

</> No changes.

The following steps that have been detected may have insecure interpolation of sensitive variables ([click here for an explanation](#)):

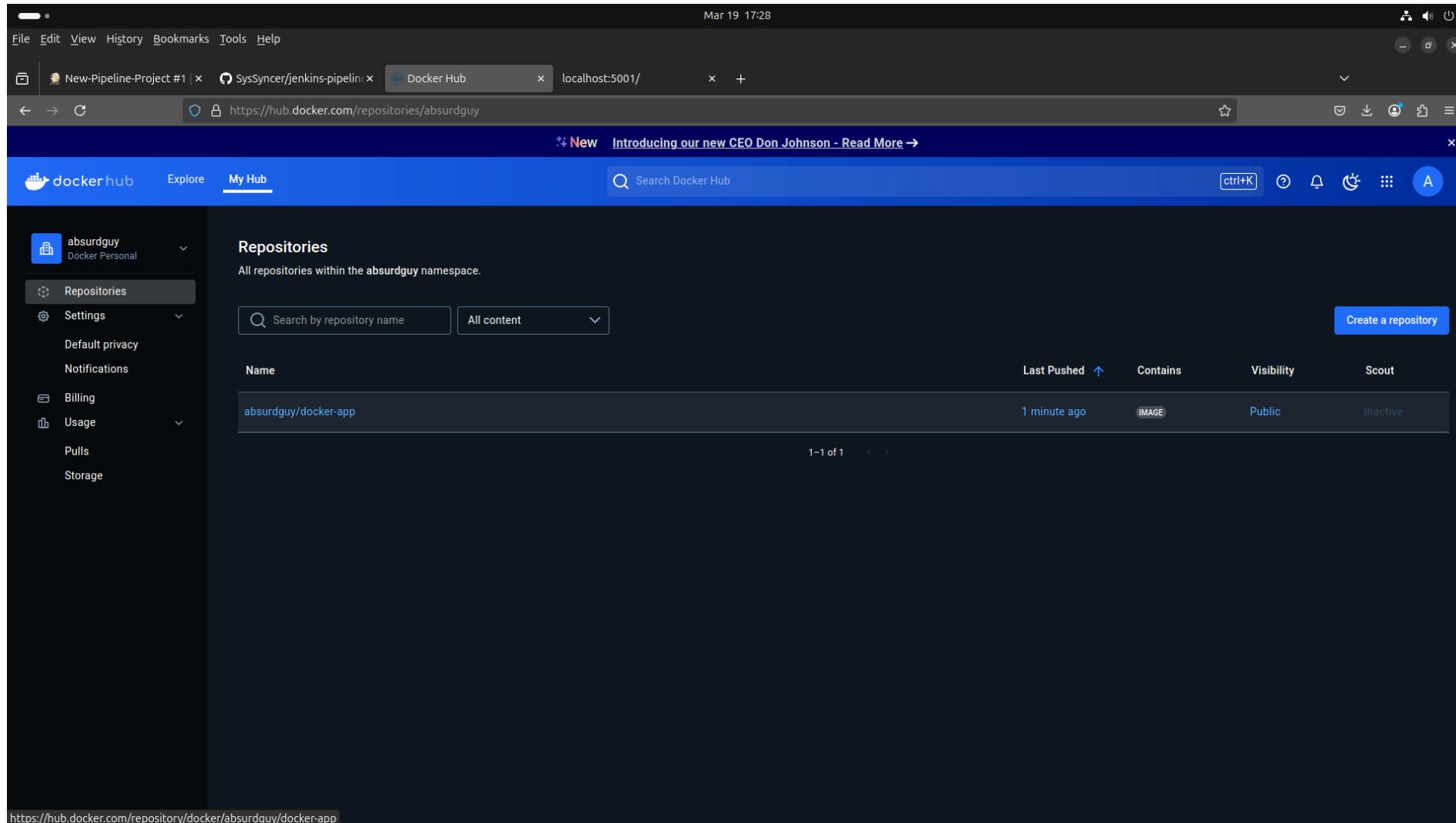
- git: [GIT_TOKEN]
- refs/remotes/origin/main

Flask application running on port 5001 now



Flask is running inside Docker container!!!

Flask running container can be seen in the docker hub dashboard



End of day 2 assignment