

CI/CD Mini Project

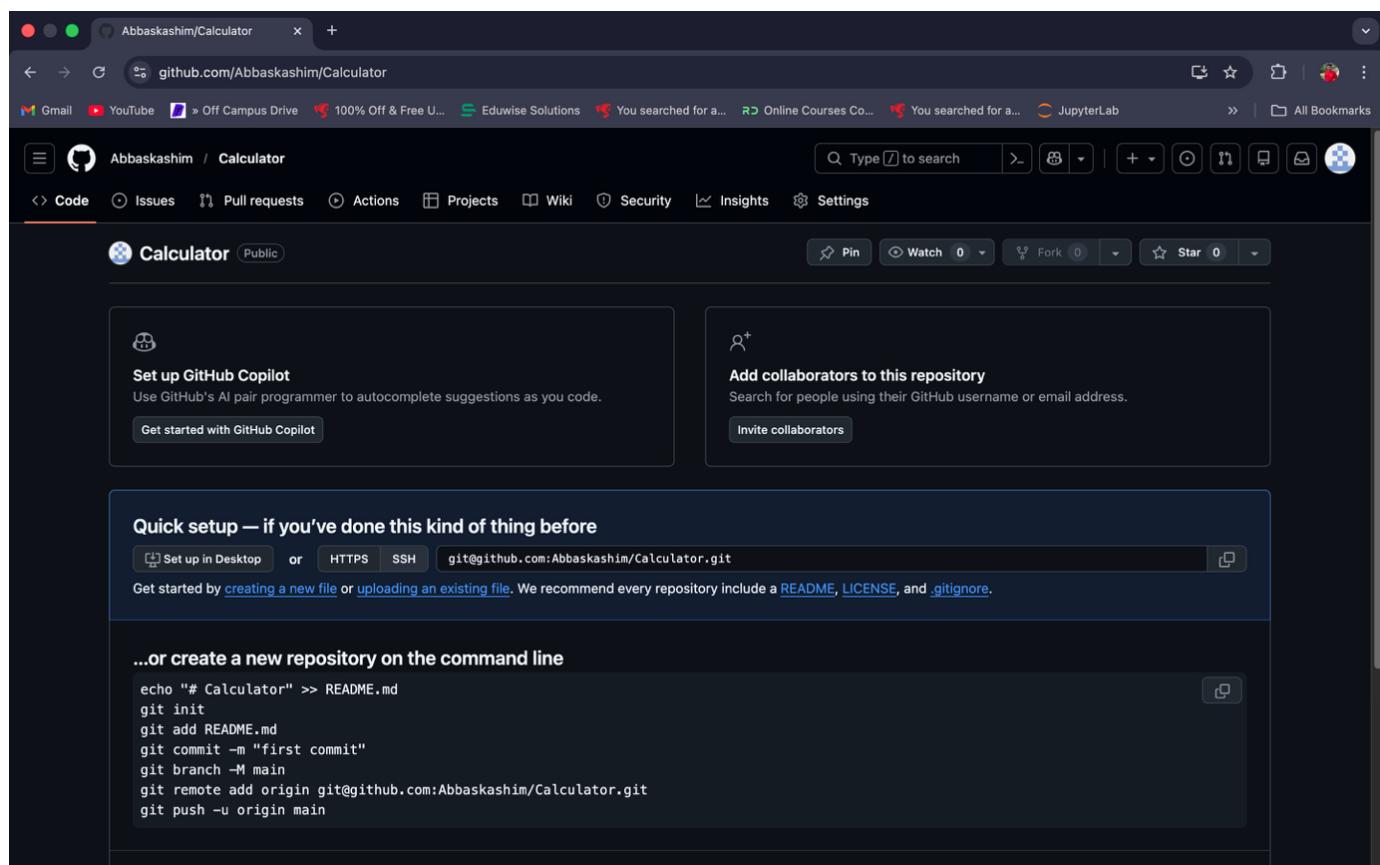
PART A – Jenkins Freestyle Job

Requirements:

✓ GIT REPOSITORIES : <https://github.com/Abbaskashim/Calculator.git>

STEP A1: CREATE GITHUB REPOSITORY

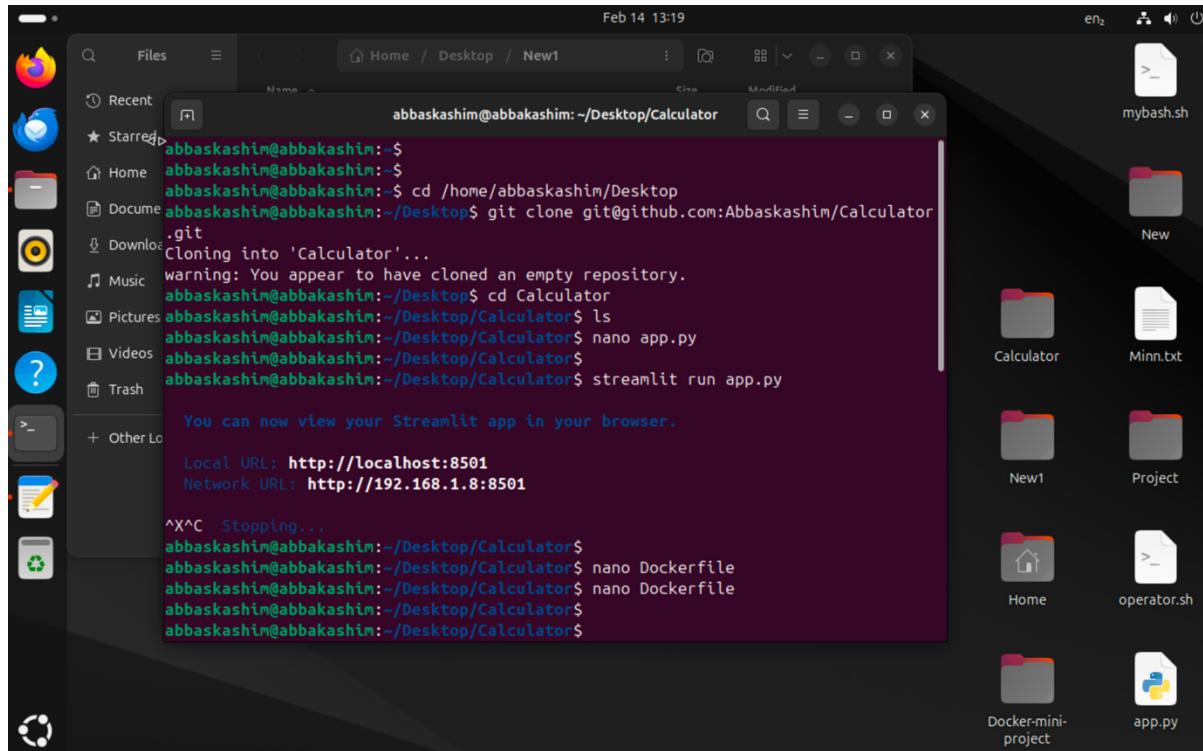
1. Open **github.com**
2. Click **New Repository**
3. Repository name: **Calculator**
4. Select **Public**
5. Click **Create Repository**



STEP A2: CREATE PROJECT FILES (LOCAL SYSTEM)

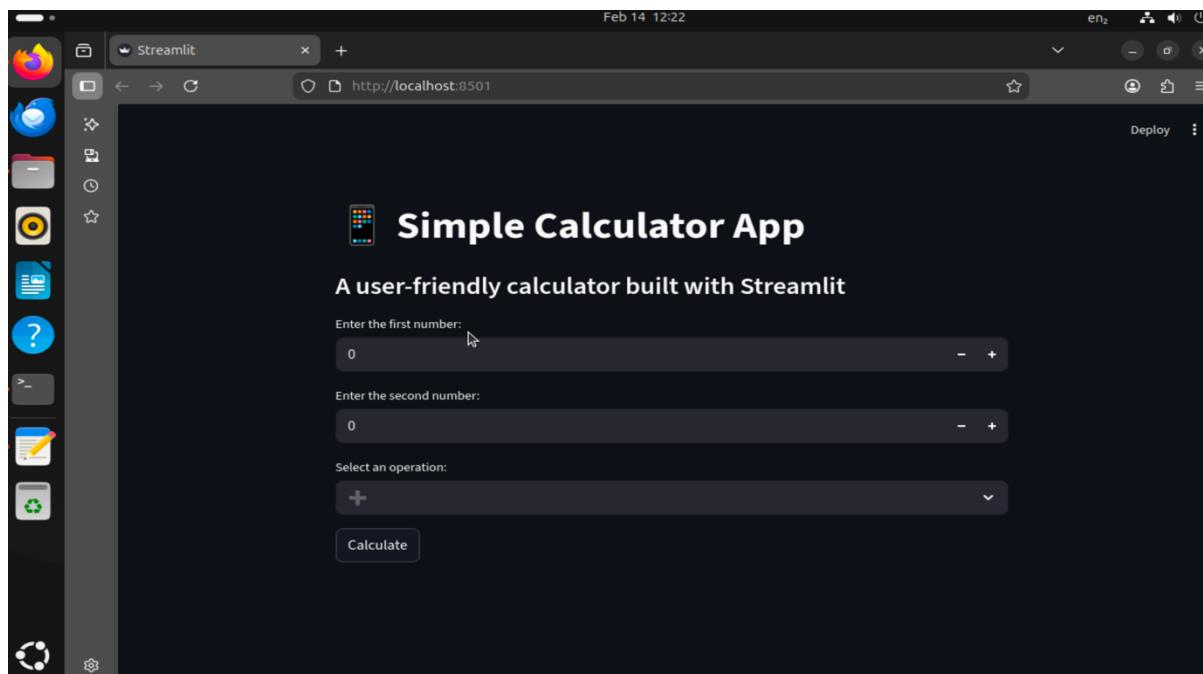
Open Terminal : Clone the Repository
Create app.py & Dockerfile

```
nano app.py  
nano Dockerfile
```



The terminal window shows the following command sequence:

```
abaskashim@abbaakashim:~/Desktop/Calculator$ git clone git@github.com:Abbaskashim/Calculator.git  
Cloning into 'Calculator'...  
warning: You appear to have cloned an empty repository.  
abaskashim@abbaakashim:~/Desktop$ cd Calculator  
abaskashim@abbaakashim:~/Desktop/Calculator$ ls  
abaskashim@abbaakashim:~/Desktop/Calculator$ nano app.py  
abaskashim@abbaakashim:~/Desktop/Calculator$ streamlit run app.py  
  
You can now view your Streamlit app in your browser.  
Local URL: http://localhost:8501  
Network URL: http://192.168.1.8:8501  
  
^X^C Stopping...  
abaskashim@abbaakashim:~/Desktop/Calculator$ nano Dockerfile  
abaskashim@abbaakashim:~/Desktop/Calculator$ nano Dockerfile  
abaskashim@abbaakashim:~/Desktop/Calculator$  
abaskashim@abbaakashim:~/Desktop/Calculator$
```



STEP A3: PUSH CODE TO GITHUB

```
git add .
git status
git commit -m "jenkins-docker-ci-project"
git remote add origin git@github.com:Abbaskashim/Calculator.git
git push
```

The terminal window shows the following command sequence:

```
root@abbakashim:/home/abbakashim/Desktop/Calculator$ nano Dockerfile
root@abbakashim:/home/abbakashim/Desktop/Calculator$ nano Dockerfile
root@abbakashim:/home/abbakashim/Desktop/Calculator$ git add .
root@abbakashim:/home/abbakashim/Desktop/Calculator$ git status
On branch main
No commits yet
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:  Dockerfile
    new file:  app.py

root@abbakashim:/home/abbakashim/Desktop/Calculator$ git commit -m "jenkins-docker-ci-project"
[main (root-commit) 2056f26] jenkins-docker-ci-project
 2 files changed, 39 insertions(+)
 create mode 100644 Dockerfile
 create mode 100644 app.py
root@abbakashim:/home/abbakashim/Desktop/Calculator$ git push
```

The terminal window is titled "root@abbakashim:/home/abbakashim/Desktop/Calculator". The background shows a dark-themed desktop environment with various icons and a file browser window.

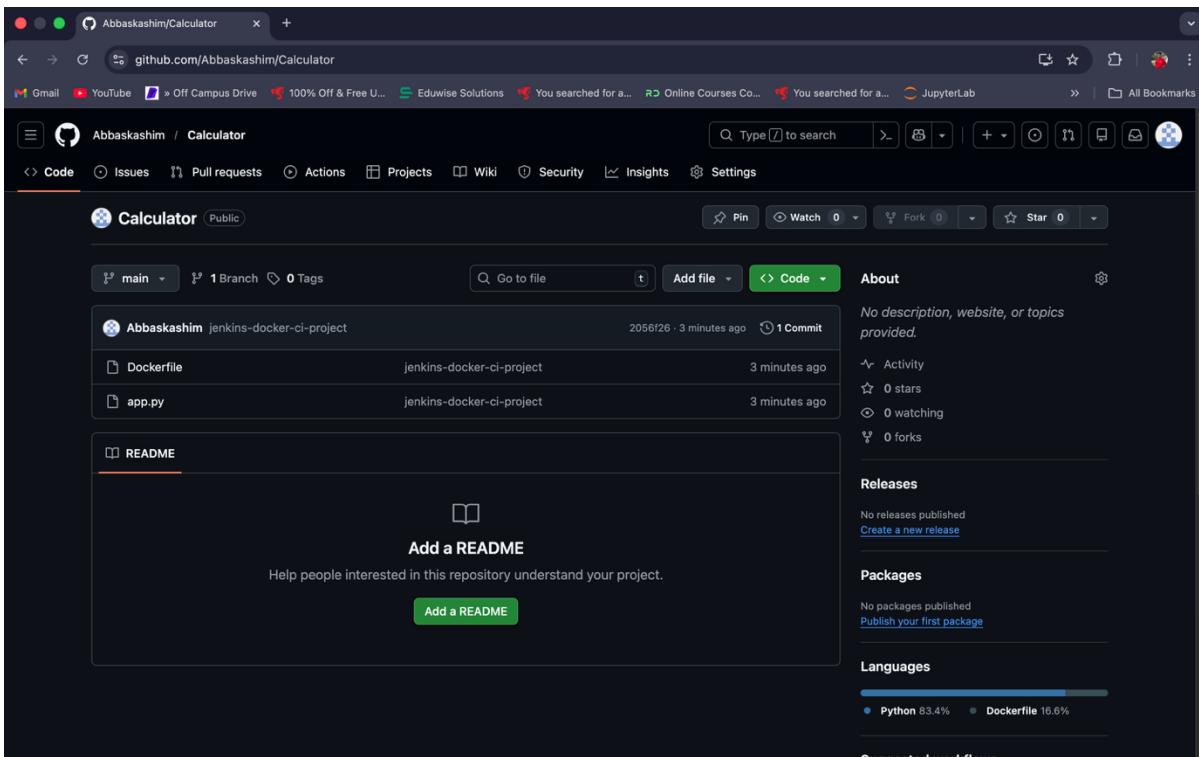
The terminal window shows the continuation of the command sequence from the previous screenshot:

```
No commits yet
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:  Dockerfile
    new file:  app.py

abbaskashim@abbakashim:/home/abbakashim/Desktop/Calculator$ git commit -m "jenkins-docker-ci-project"
[main (root-commit) 2056f26] jenkins-docker-ci-project
 2 files changed, 39 insertions(+)
 create mode 100644 Dockerfile
 create mode 100644 app.py
abbaskashim@abbakashim:/home/abbakashim/Desktop/Calculator$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 782 bytes | 391.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:Abbaskashim/Calculator.git
 * [new branch]      main -> main
abbaskashim@abbakashim:/home/abbakashim/Desktop/Calculator$
```

The terminal window is titled "root@abbakashim:/home/abbakashim/Desktop/Calculator". The background shows a dark-themed desktop environment with various icons and a file browser window.

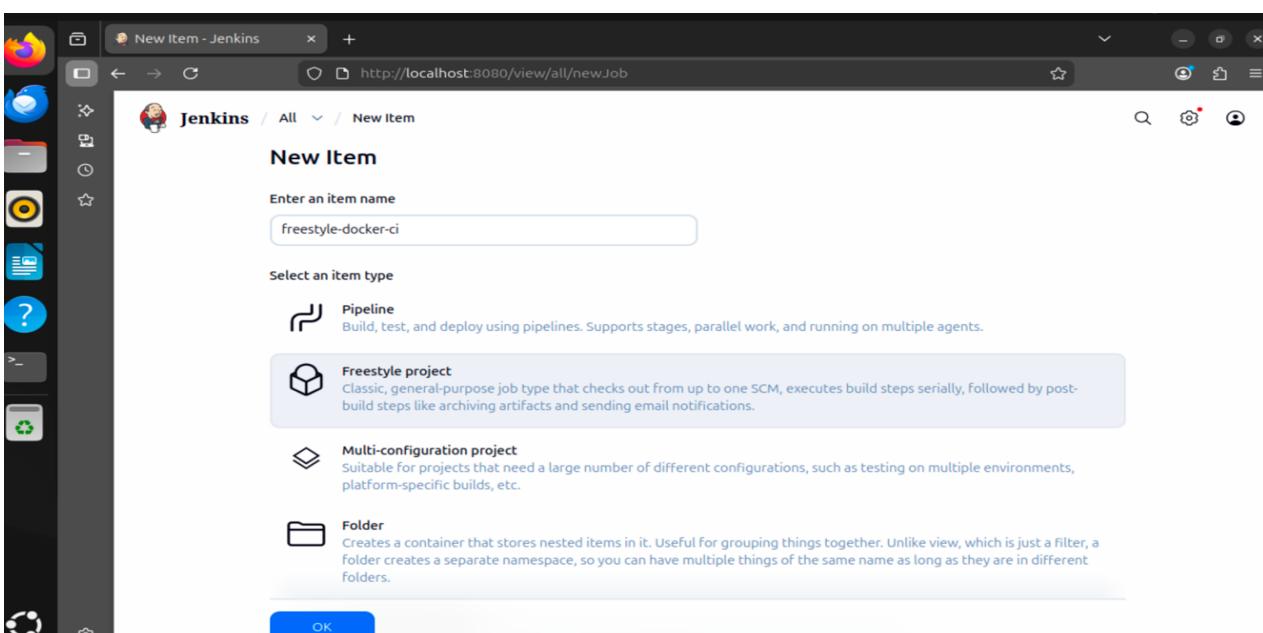
Output:



STEP A4: CREATE FREESTYLE JOB IN JENKINS

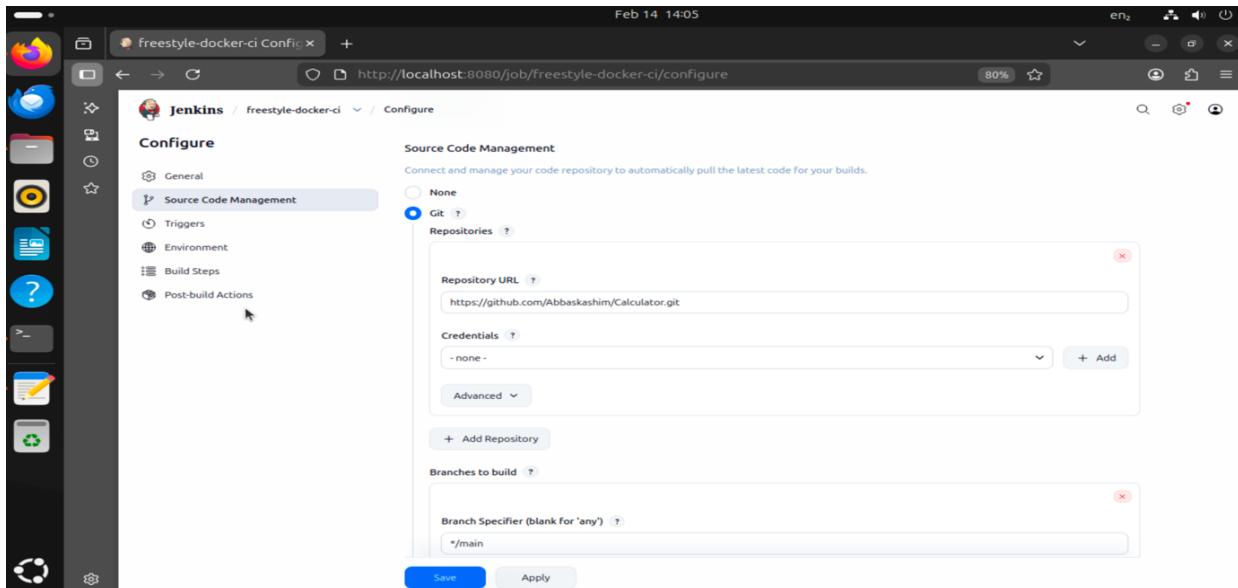
What to do

1. Open Jenkins dashboard
2. Click New Item
3. Enter name: freestyle-docker-ci
4. Select Freestyle project
5. Click OK



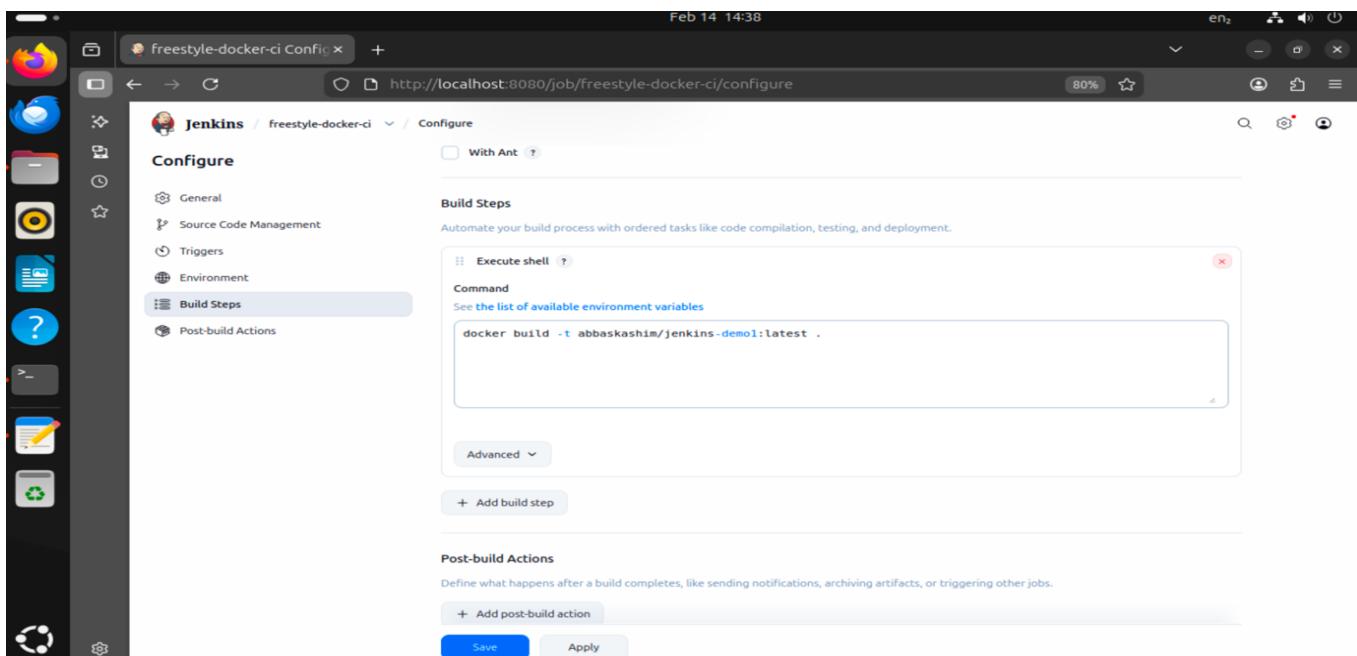
STEP A5: CONNECT JENKINS TO GITHUB

1. Click **Configure**
2. Scroll to **Source Code Management**
3. Select **Git**
4. Repository URL: <https://github.com/Abbaskashim/Calculator.git>
5. Branch: `main`
6. Click **Save**



STEP A6: BUILD DOCKER IMAGE

1. Click Configure
2. Scroll to Build
3. Click Add Build Step
4. Select Execute shell
5. Paste: `docker build -t abbaskashim/jenkins-demo1:latest .`



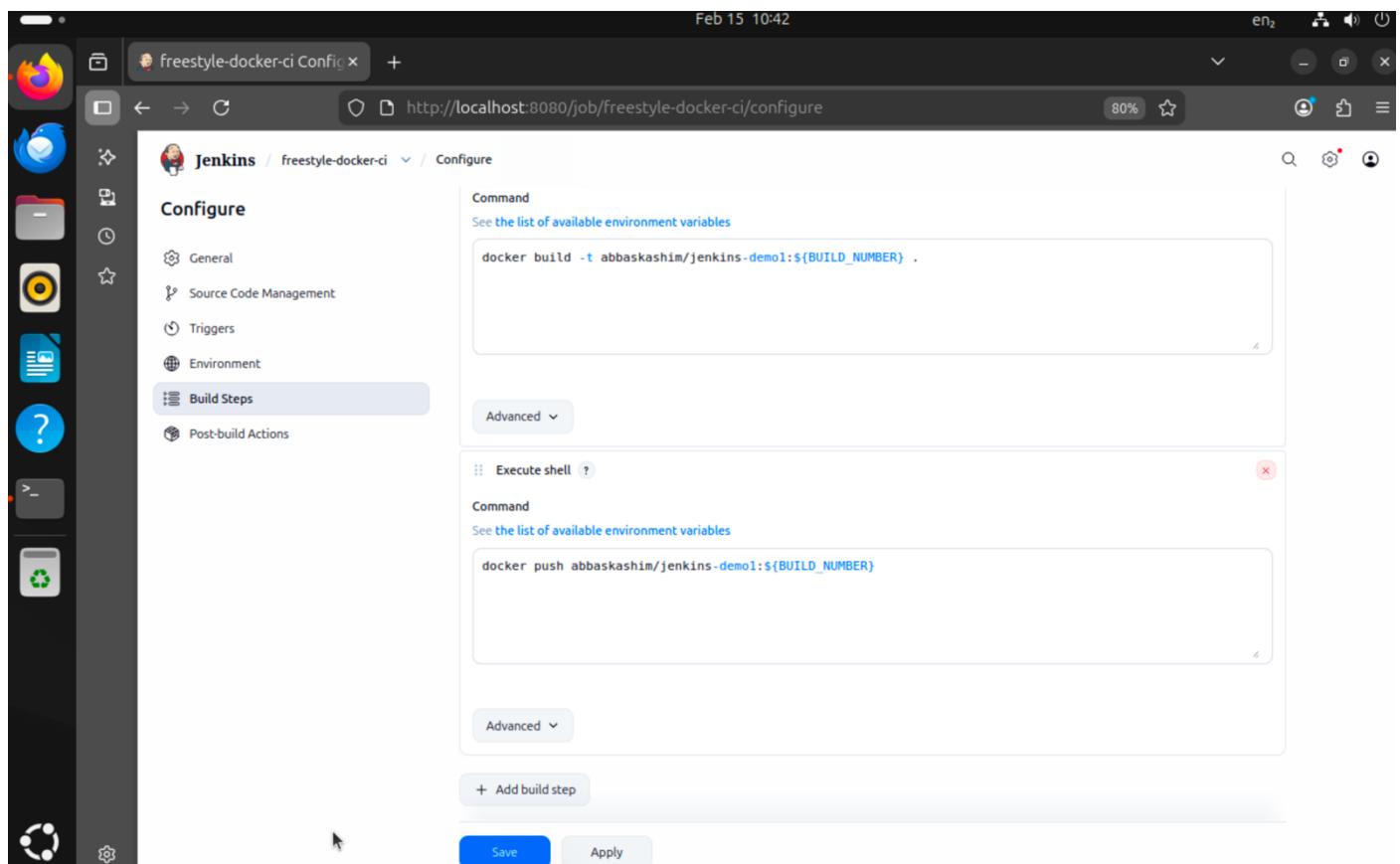
The screenshot shows a Firefox browser window with the address bar at `http://localhost:8080/job/freestyle-docker-ci/2/console`. The main content area displays the Jenkins build log for job `freestyle-docker-ci`, build #2. The log output is as follows:

```
#8 56.19 Collecting referencing>=0.28.4
#8 56.21 Downloading referencing-0.36.2-py3-none-any.whl (26 kB)
#8 56.26 Collecting six>=1.5
#8 56.29 Downloading six-1.17.0-py2.py3-none-any.whl (11 kB)
#8 56.52 Installing collected packages: pytz, watchdog, urllib3, tzdata, typing-extensions, tornado, toml, tenacity, smmap, six, rpdbs-py, pyarrow, protobuf, pillow, packaging, numpy, narwhal, MarkupSafe, idna, click, charset_normalizer, certifi, cachetools, blinker, attrs, requests, referencing, python-dateutil, jinja2, gitdb, pydeck, pandas, jsonschema specifications, gipython, jsonschema, altair, streamlit
#8 67.00 Successfully installed MarkupSafe-3.0.3 altair-5.5.0 attrs-25.4.0 blinker-1.9.0 cachetools-6.2.6 certifi-2026.1.4 charset_normalizer-3.4.4 click-8.1.8 gitdb-4.0.12 gipython-3.1.46 idna-3.11 jinja2-3.1.6 jsonschema-4.25.1 jsonschema-specifications-2025.1.9 narwhal-2.16.0 numpy-2.0.2 packaging-25.0 pandas-2.3.3 pillow-11.3.0 protobuf-6.33.5 pyarrow-21.0.0 pydeck-0.9.1 python-dateutil-2.9.0.post0 pytz-2025.2 referencing-0.36.2 requests-2.32.5 rpdbs-py-0.27.1 six-1.17.0 smmap-5.0.2 streamlit-1.50.0 tenacity-9.1.2 toml-0.10.2 tornado-6.5.4 typing-extensions-4.15.0 tzdata-2025.3 urllib3-2.6.3 watchdog-6.0.0
#8 67.00 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
#8 67.23
#8 67.23 [notice] A new release of pip is available: 23.0.1 -> 26.0.1
#8 67.23 [notice] To update, run: pip install --upgrade pip
#8 DONE 67.9s

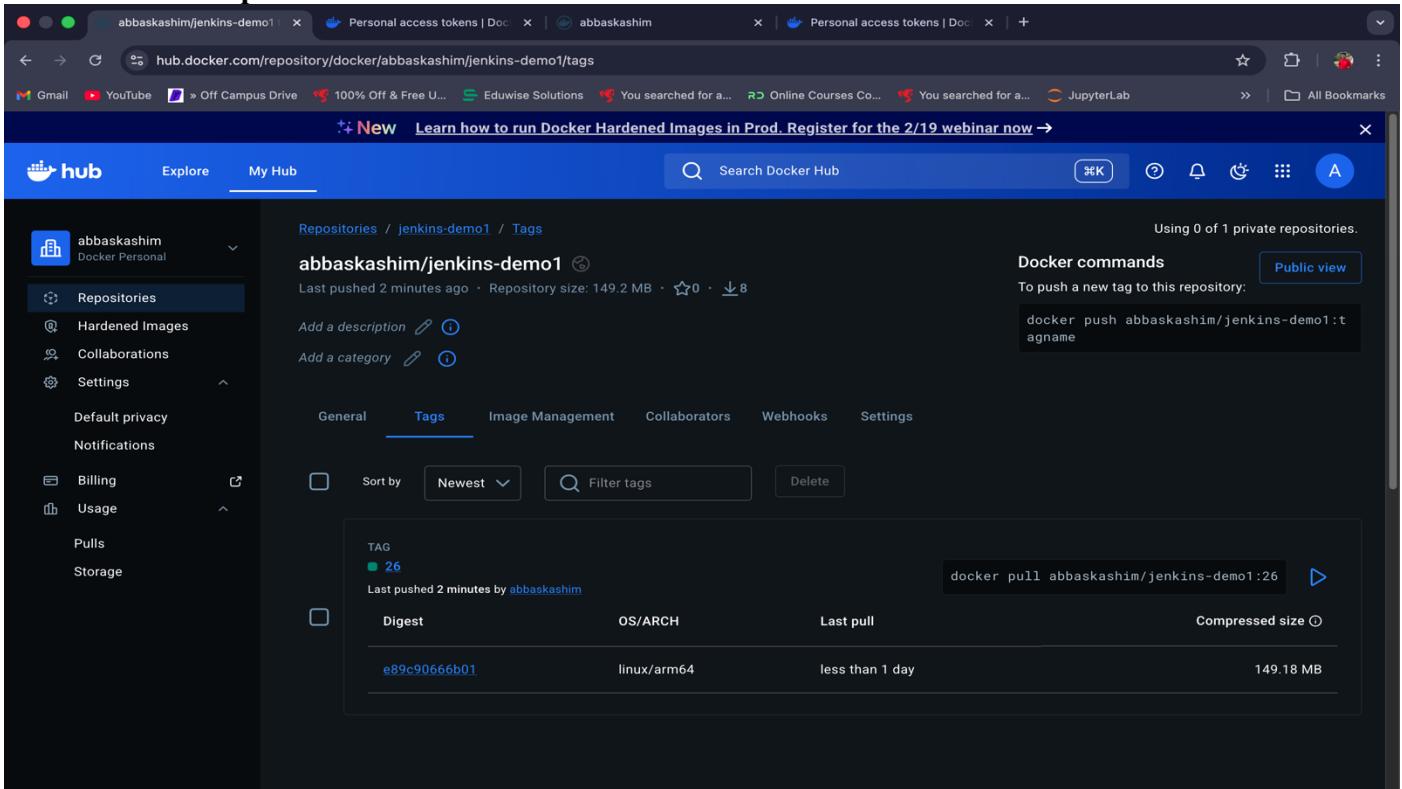
#9 exporting to image
#9 exporting layers
#9 exporting layers 1.6s done
#9 writing image sha256:d1406f2fbedb4d0f5a93efa7cf0d1161fc58eb9e099ad79b2a3f260397f88f9d done
#9 naming to docker.io/abbaskashim/jenkins-demo:latest done
#9 DONE 1.6s
Finished: SUCCESS
```

STEP A7: PUSH IMAGE TO DOCKER HUB

1. Click **Add Build Step** → **Execute shell**
 2. Paste: `docker build -t abbaskashim/jenkins-demo1:latest .`

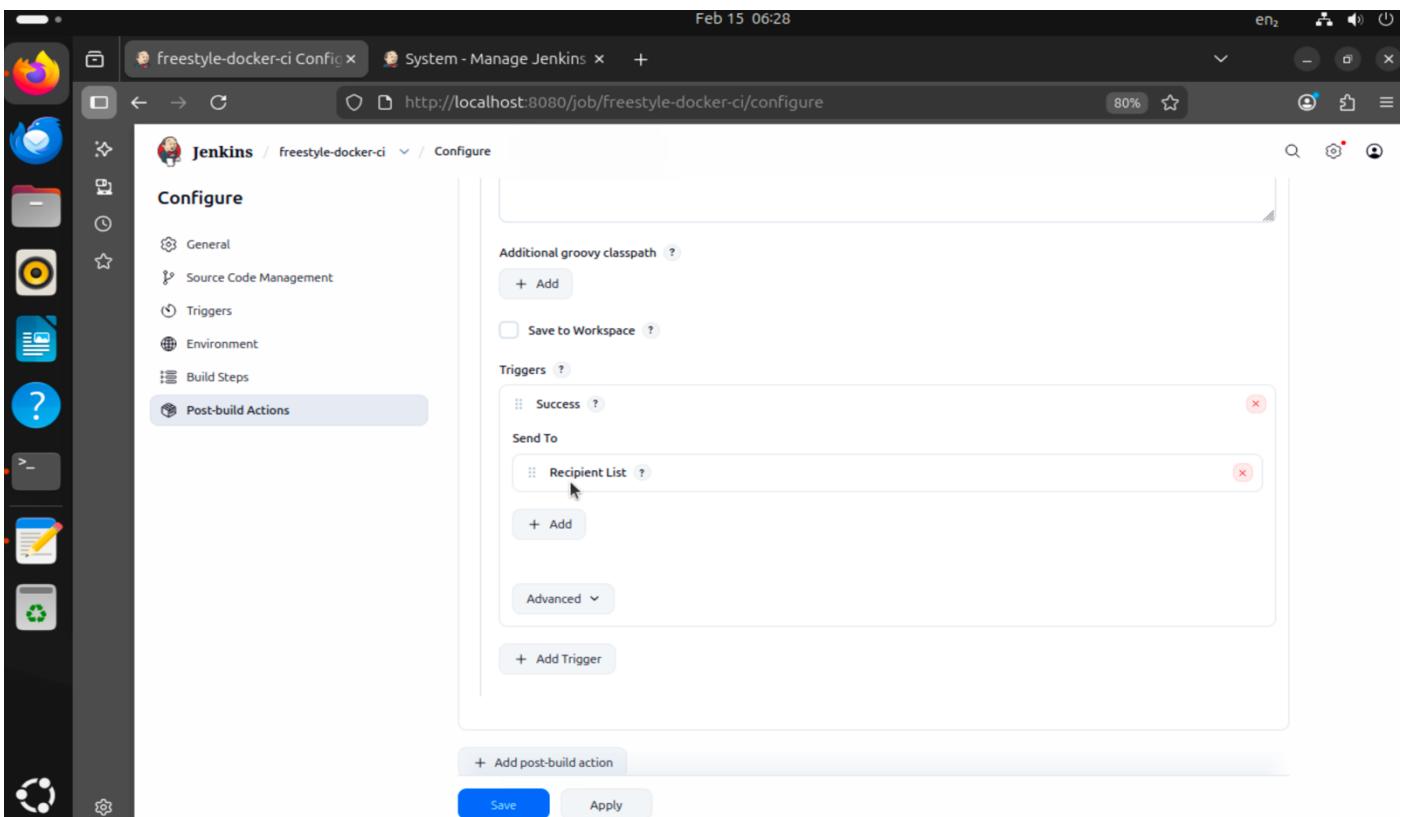


Docker Push Output:



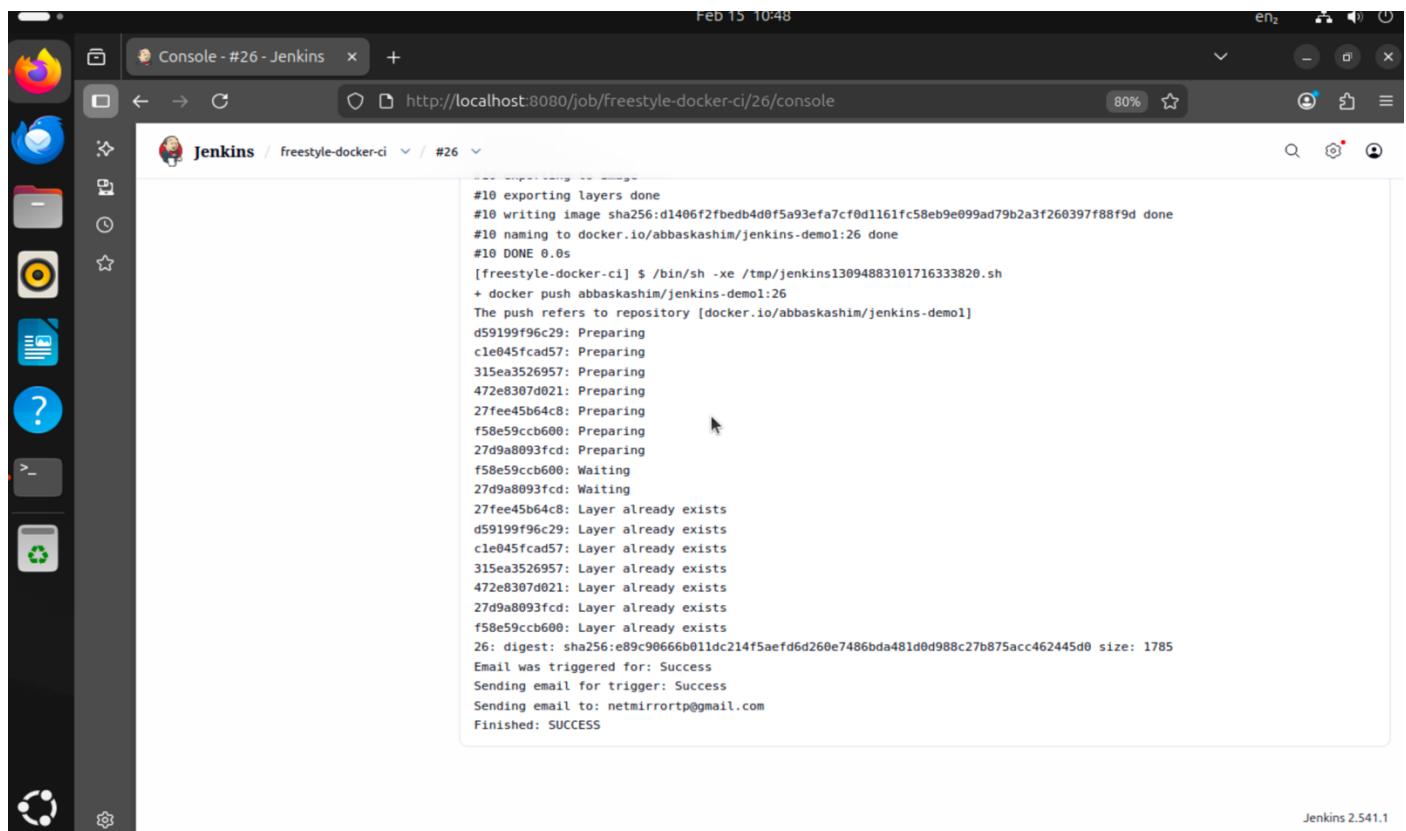
STEP A8: EMAIL NOTIFICATION

1. Scroll to Post-build Actions
2. Select Email Notification
3. Enable Send on success
4. Add your email



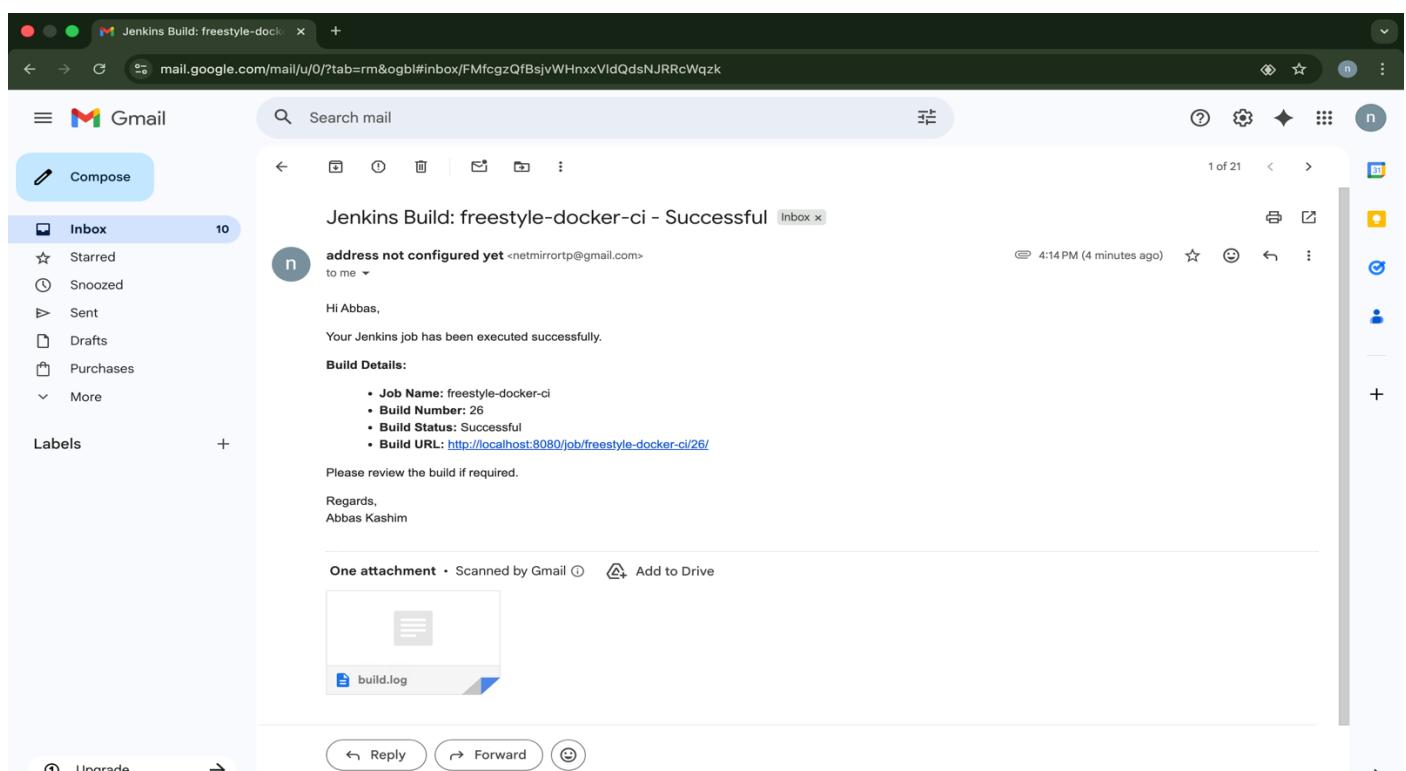
STEP A9: RUN FREESTYLE JOB

1. Click **Build Now**
2. Open **Console Output**



Feb 15 10:48
Console - #26 - Jenkins
<http://localhost:8080/job/freestyle-docker-ci/26/console> 80%
Jenkins / freestyle-docker-ci / #26
Feb 15 10:48 [jenkins] \$ docker build -t abbakashim/jenkins-demo1 .
#10 exporting layers done
#10 writing image sha256:d1406f2fbedb4d0ff5a93efa7cf0d1161fc58eb9e099ad79b2a3f260397f88f9d done
#10 naming to docker.io/abbakashim/jenkins-demo1:26 done
#10 DONE 0.0s
[freestyle-docker-ci] \$ /bin/sh -xe /tmp/jenkins13094883101716333820.sh
+ docker push abbakashim/jenkins-demo1:26
The push refers to repository [docker.io/abbakashim/jenkins-demo1]
d59199f96c29: Preparing
cle045fcad57: Preparing
315ea3526957: Preparing
472e8307d021: Preparing
27fee45b64c8: Preparing
f58e59ccb600: Preparing
27d9a8093fcfd: Preparing
f58e59ccb600: Waiting
27d9a8093fcfd: Waiting
27fee45b64c8: Layer already exists
d59199f96c29: Layer already exists
cle045fcad57: Layer already exists
315ea3526957: Layer already exists
472e8307d021: Layer already exists
27d9a8093fcfd: Layer already exists
f58e59ccb600: Layer already exists
26: digest: sha256:e89c90666b01ldc214f5aefdf6d260e7486bda481d0d988c27b875acc462445d0 size: 1785
Email was triggered for: Success
Sending email for trigger: Success
Sending email to: netmirrortp@gmail.com
Finished: SUCCESS

Email Notification:



Jenkins Build: freestyle-docker-ci - Successful

address not configured yet <netmirrortp@gmail.com>
to me ▾
Hi Abbas,
Your Jenkins job has been executed successfully.
Build Details:

- Job Name: freestyle-docker-ci
- Build Number: 26
- Build Status: Successful
- Build URL: <http://localhost:8080/job/freestyle-docker-ci/26/>

Please review the build if required.
Regards,
Abbas Kashim

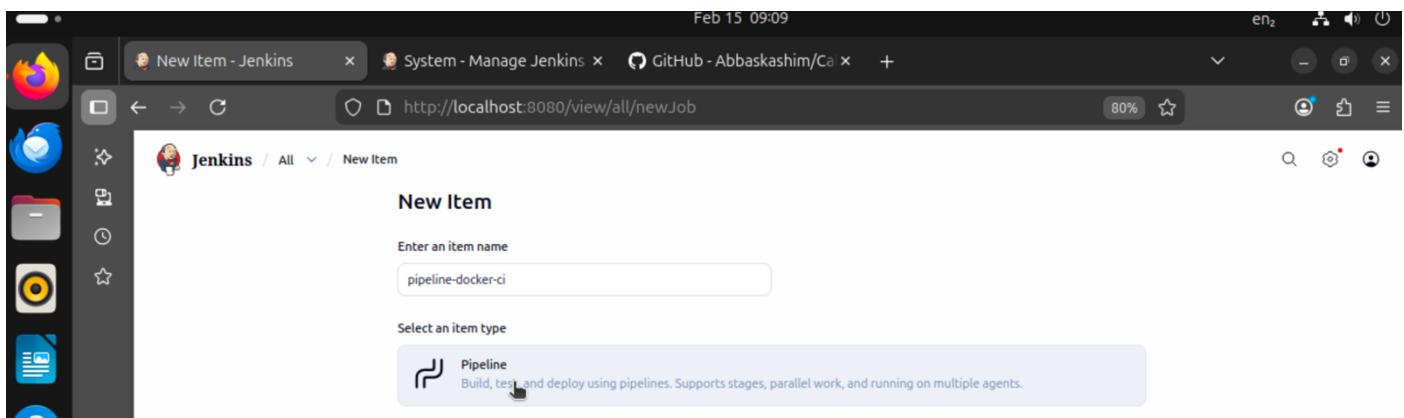
One attachment • Scanned by Gmail ⓘ Add to Drive

build.log

PART B – Jenkins Pipeline Job

STEP B1: CREATE PIPELINE JOB

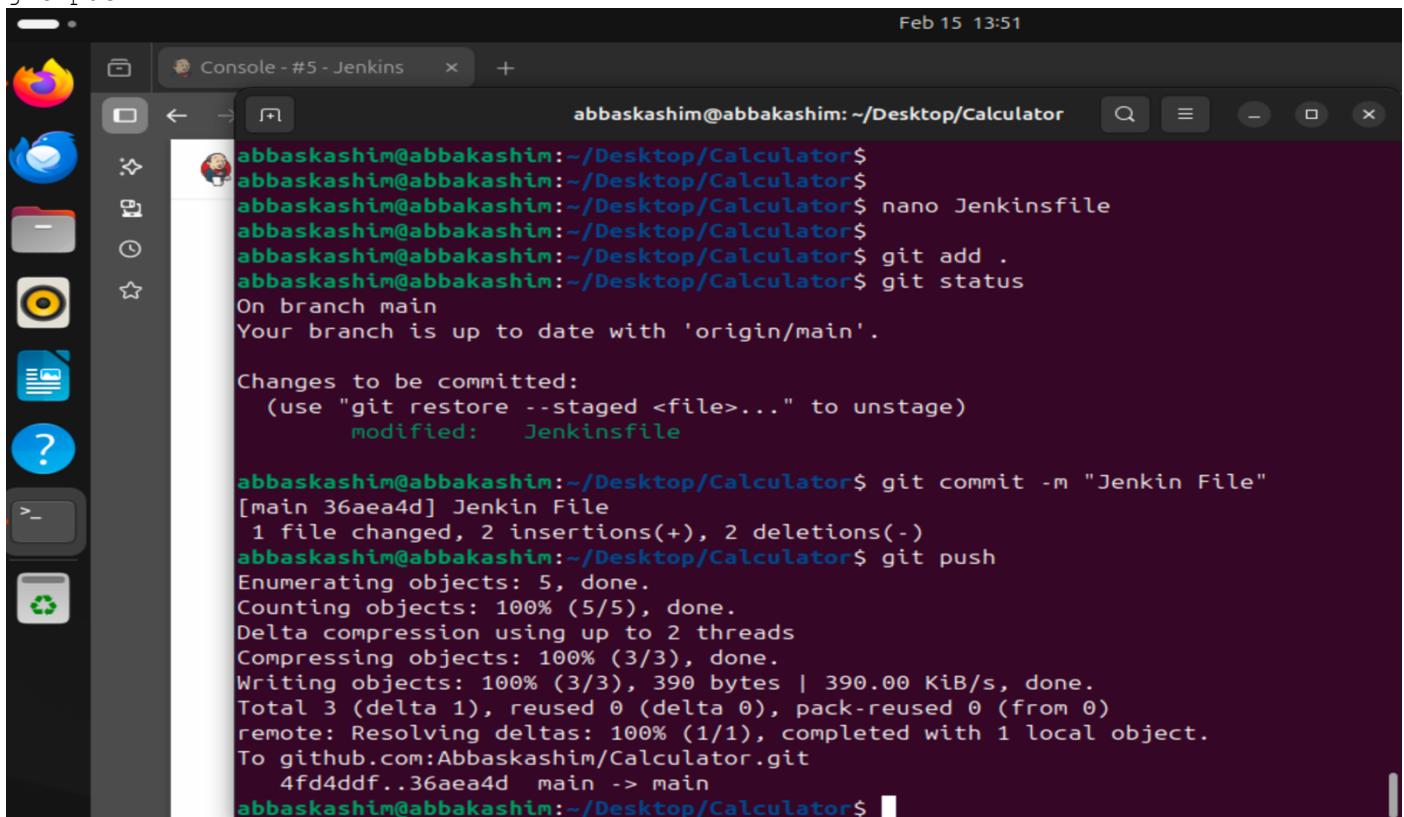
1. Jenkins → New Item
2. Job name: pipeline-docker-ci
3. Select Pipeline
4. Click OK



STEP B2: CREATE & PUSH JENKINSFILE TO GITHUB

In your project folder: nano Jenkinsfile

```
git add .  
git commit -m "Jenkin file"  
git push
```



```
abbaakashim@abbakashim:~/Desktop/Calculator$ nano Jenkinsfile  
abbaakashim@abbakashim:~/Desktop/Calculator$ git add .  
abbaakashim@abbakashim:~/Desktop/Calculator$ git status  
On branch main  
Your branch is up to date with 'origin/main'.  
  
Changes to be committed:  
  (use "git restore --staged <file>" to unstage)  
    modified:   Jenkinsfile  
  
abbaakashim@abbakashim:~/Desktop/Calculator$ git commit -m "Jenkin File"  
[main 36aea4d] Jenkin File  
 1 file changed, 2 insertions(+), 2 deletions(-)  
abbaakashim@abbakashim:~/Desktop/Calculator$ git push  
Enumerating objects: 5, done.  
Counting objects: 100% (5/5), done.  
Delta compression using up to 2 threads  
Compressing objects: 100% (3/3), done.  
Writing objects: 100% (3/3), 390 bytes | 390.00 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 github.com:AbbasKashim/Calculator.git  
  4fd4ddf..36aea4d  main -> main  
abbaakashim@abbakashim:~/Desktop/Calculator$
```

STEP B4: CONFIGURE PIPELINE JOB

1. Job → **Configure**
2. Definition: Pipeline script from SCM
3. SCM: Git
4. Repo URL
5. Branch: main
6. Script Path: Jenkinsfile
7. Save

The screenshot shows the Jenkins Pipeline configuration page for a job named "pipeline-docker-ci". The left sidebar has "Pipeline" selected under the "Configure" section. The main panel is titled "Pipeline" and says "Define your Pipeline using Groovy directly or pull it from source control." Under "Definition", "Pipeline script from SCM" is selected. In the "SCM" section, "Git" is chosen, and the "Repository URL" is set to "https://github.com/AbbasKashim/Calculator.git". There are no credentials defined.

The screenshot shows the continuation of the Jenkins Pipeline configuration page. The "Branches to build" section has "Branch Specifier (blank for 'any')" set to "*main". The "Script Path" is set to "Jenkinsfile". The "Lightweight checkout" checkbox is checked.

STEP B5: RUN PIPELINE

1. Click **Build Now**
2. Watch stages turn green

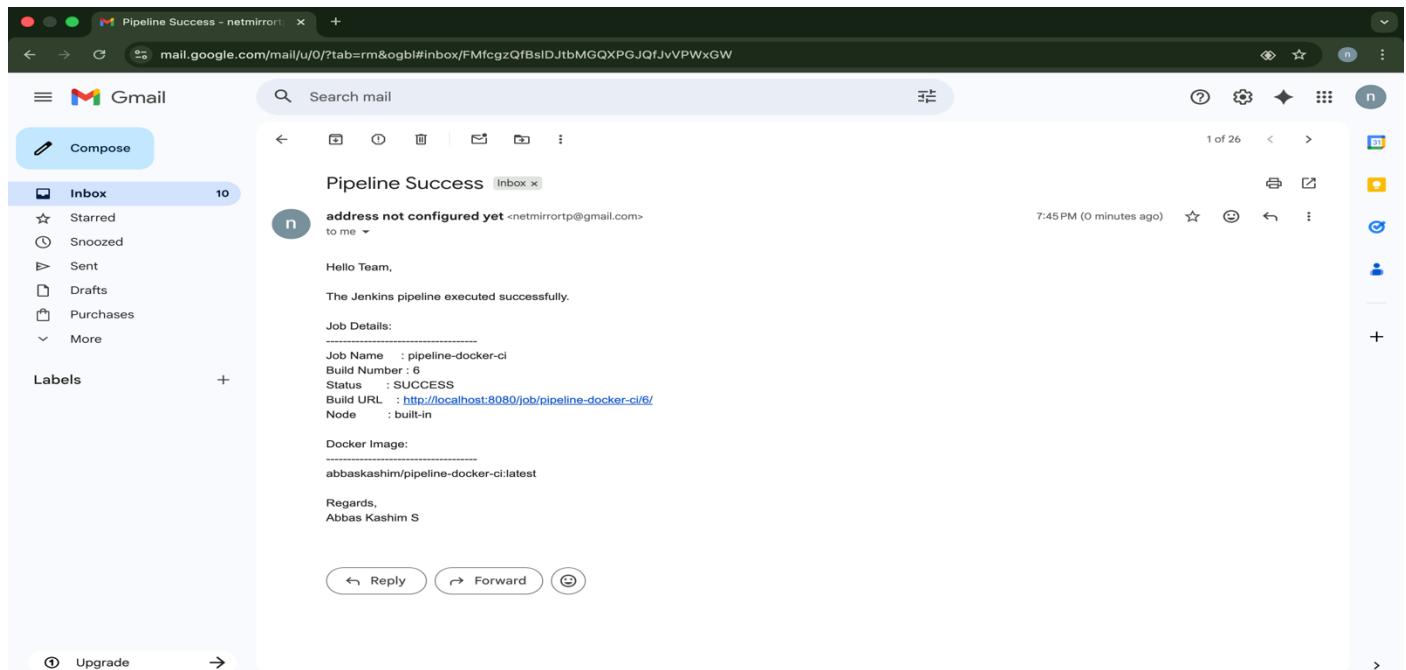
The screenshot shows the Jenkins pipeline status page for the job 'pipeline-docker-ci'. The top navigation bar includes links for 'Status', 'Changes', 'Build Now', 'Configure', 'Delete Pipeline', 'Stages', 'Rename', and 'Pipeline Syntax'. Below this, a 'Builds' section lists three builds: #6 (14:14), #5 (13:46), and #3 (13:22). Each build entry has a dropdown arrow indicating more details. To the right, a list of recent builds is shown, all marked with a green checkmark and labeled 'Last build (#6), 1 min 27 sec ago'.

The screenshot shows the Jenkins console output for build #6. The output log is displayed in a large text area, showing the execution of a Docker pipeline. The log includes commands like 'Preparing' for various layers, layer existence checks ('Layer already exists'), and finally the successful digest calculation and size information. The log concludes with '[Pipeline] End of Pipeline' and 'Finished: SUCCESS'.

```
315ea3526957: Preparing
472e8307d021: Preparing
27fee45b64c8: Preparing
f58e59ccb600: Preparing
27d9a8093fcfd: Preparing
f58e59ccb600: Waiting
27d9a8093fcfd: Waiting
315ea3526957: Layer already exists
472e8307d021: Layer already exists
d59199f96c29: Layer already exists
27fee45b64c8: Layer already exists
cle045fcad57: Layer already exists
27d9a8093fcfd: Layer already exists
f58e59ccb600: Layer already exists
latest: digest: sha256:e89c90666b011dc214f5aefdf6d260e748abda481d0d988c27b875acc462445d0 size: 1785
[Pipeline]
[Pipeline] // withCredentials
[Pipeline]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] mail
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // withEnv
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Step B6: Email Notification from Pipeline

1. Email sent after successful pipeline execution
2. Must include job and build detail



Part C: Comparison & Analysis

Step 11: Freestyle vs Pipeline Comparison

| Criteria | Freestyle Job | Pipeline Job |
|----------------------|--|---|
| Configuration Method | Configured using the Jenkins web UI (manual setup) | Configured using code written in a Jenkinsfile |
| Version Control | ✗ Not version controlled (stored in Jenkins UI) | ✓ Version controlled along with source code |
| Reusability | Low – difficult to reuse across jobs | High – pipeline code can be reused and shared |
| Complexity Handling | Suitable for simple, linear tasks | Designed for complex workflows (CI/CD pipelines) |
| Maintainability | Harder to maintain as changes are manual | Easy to maintain due to code-based configuration |
| Scalability | Limited scalability | Highly scalable for large projects |
| Error Handling | Basic error handling | Advanced error handling using stages and conditions |
| Collaboration | Limited team collaboration | Better collaboration through shared Jenkinsfile |
| Automation Level | Partial automation | Full CI/CD automation |
| Best Use Case | Small projects or beginner tasks | Enterprise-level CI/CD pipelines |