Project 1

Automate docker built and push using Jenkinsfile

1) **Setup a Simple Flask App**

Project Structure

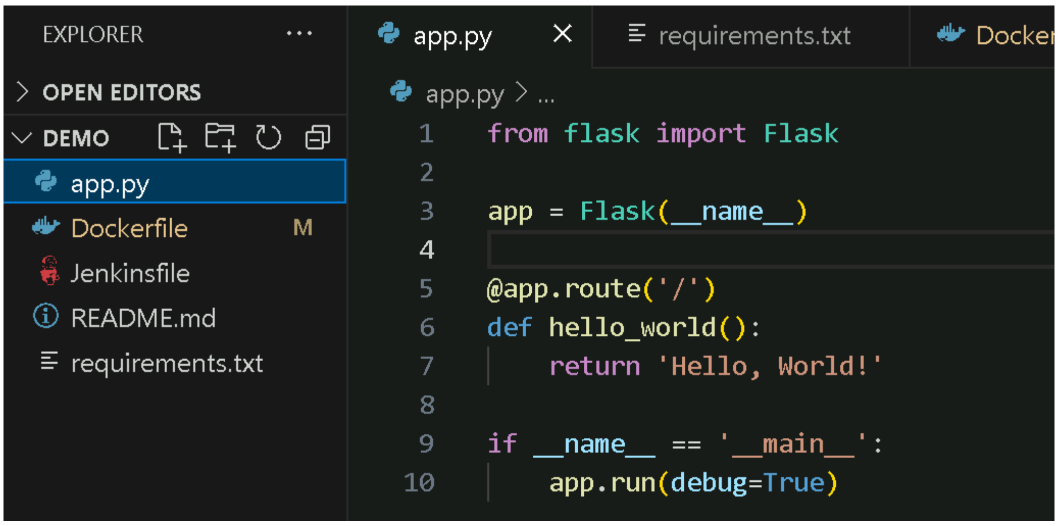
```
my-flask-app
│— app.py
│— requirements.txt
│— Dockerfile
│— Jenkinsfile
```

app.py: The main Flask application file.

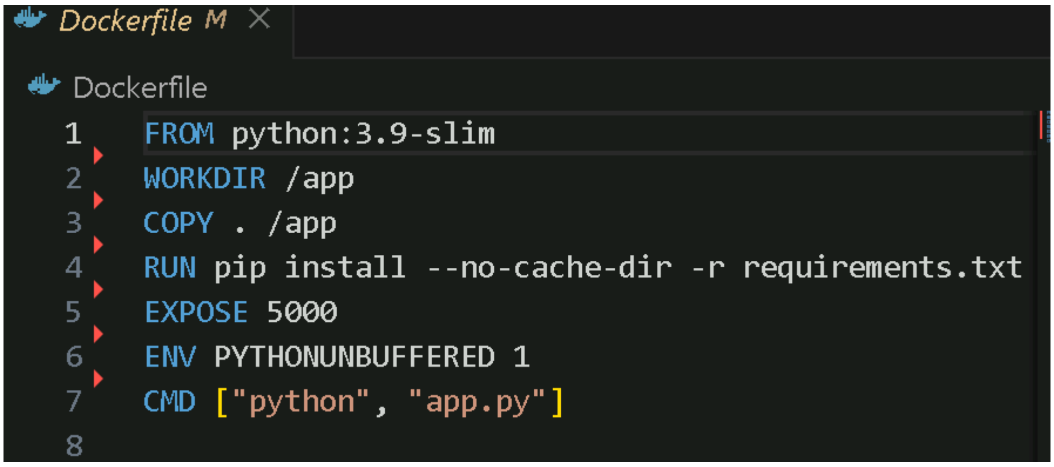
requirements.txt: List of dependencies (Flask and others).

Dockerfile: Defines the Docker image for the Flask app.

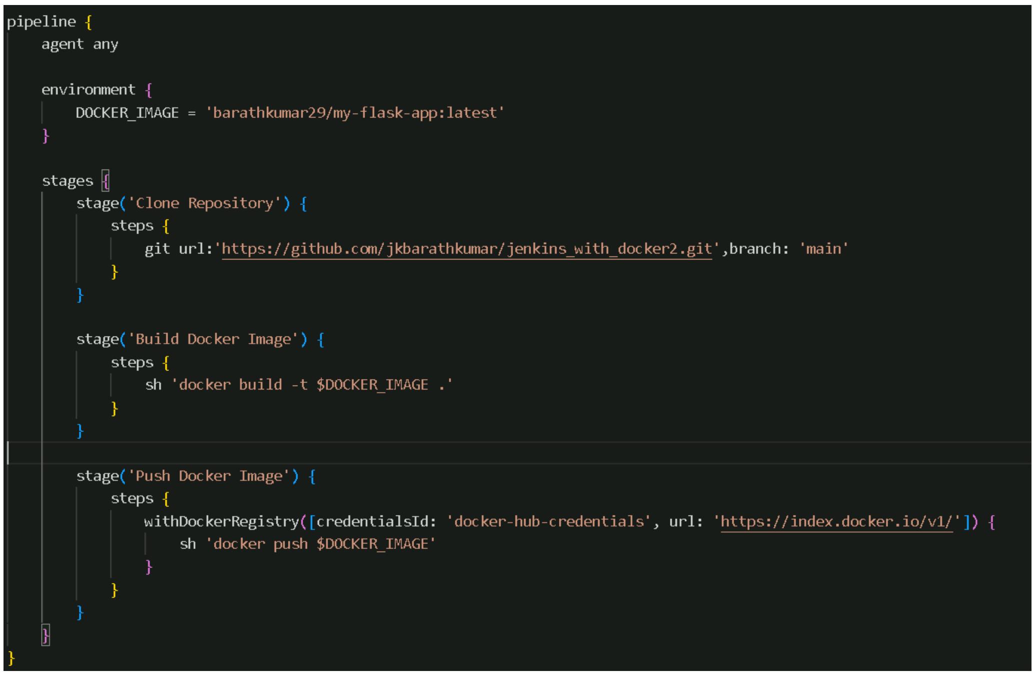
Jenkinsfile: Contains the Jenkins pipeline configuration.



```python
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, World!'

if __name__ == '__main__':
    app.run(debug=True)
```



```dockerfile
FROM python:3.9-slim
WORKDIR /app
COPY . /app
RUN pip install --no-cache-dir -r requirements.txt
EXPOSE 5000
ENV PYTHONUNBUFFERED 1
CMD ["python", "app.py"]
```

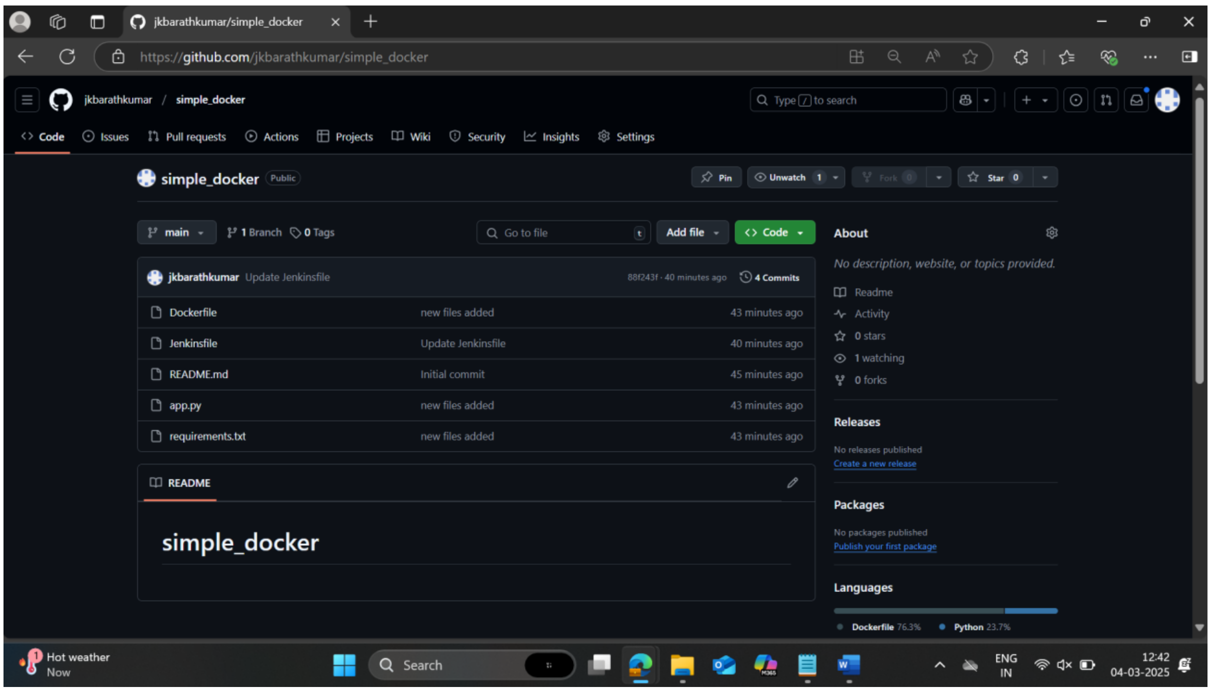


```
pipeline {
    agent any

    environment {
        DOCKER_IMAGE = 'barathkumar29/my-flask-app:latest'
    }

    stages {
        stage('Clone Repository') {
            steps {
                git url:'https://github.com/jkbarathkumar/jenkins_with_docker2.git',branch: 'main'
            }
        }

        stage('Build Docker Image') {
            steps {
                sh 'docker build -t $DOCKER_IMAGE .'
            }
        }

        stage('Push Docker Image') {
            steps {
                withDockerRegistry([credentialsId: 'docker-hub-credentials', url: 'https://index.docker.io/v1/']) {
                    sh 'docker push $DOCKER_IMAGE'
                }
            }
        }
    }
}
```

Github link for the code: jkbarathkumar/jenkins_with_docker2

2. Push the Code to GitHub

- **Make sure you have a GitHub repository created for the project.**
- **Push all the files (app.py, requirements.txt, Dockerfile, Jenkinsfile) to the GitHub repository**



https://github.com/jkbarathkumar/simple_docker

jkbarathkumar / simple_docker

simple_docker Public

main · 1 Branch · 0 Tags

jkbarathkumar Update Jenkinsfile · 88f243f · 40 minutes ago · 4 Commits

Dockerfile	new files added	43 minutes ago
Jenkinsfile	Update Jenkinsfile	40 minutes ago
README.md	Initial commit	45 minutes ago
app.py	new files added	43 minutes ago
requirements.txt	new files added	43 minutes ago

README

simple_docker

About

No description, website, or topics provided.

Readme · Activity · 0 stars · 1 watching · 0 forks

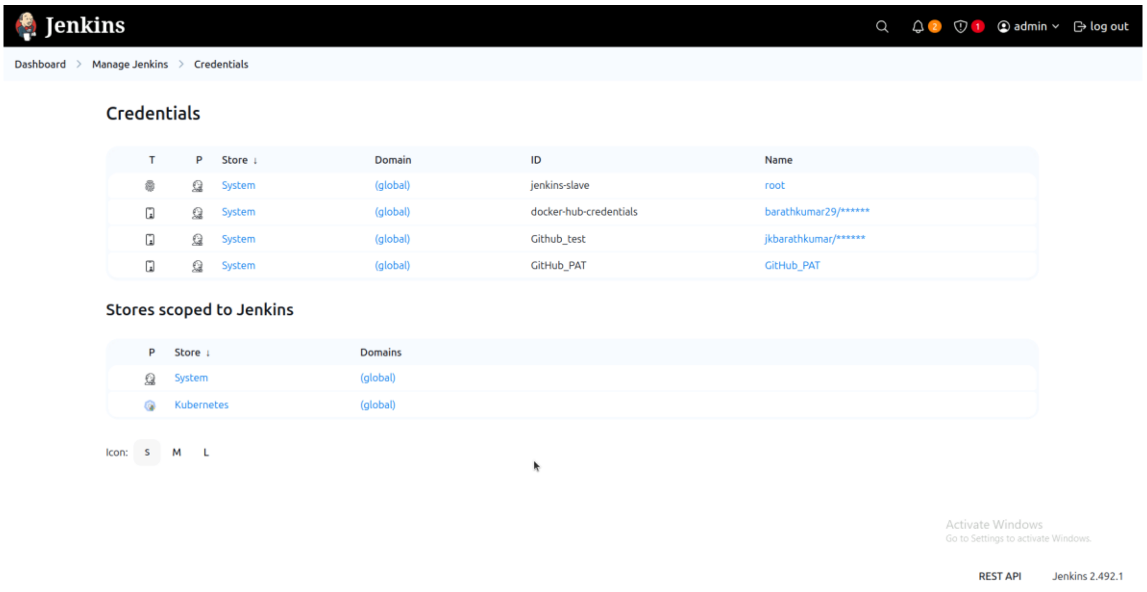
Releases: No releases published. Create a new release

Packages: No packages published. Publish your first package

Languages: Dockerfile 76.3% · Python 23.7%

3. Configure Docker Hub Credentials in Jenkins

- **Go to Jenkins > Manage Jenkins > Manage Credentials.**
- **Add new credentials:**
  - **Username: Your Docker Hub username.**
  - **Password: Your Docker Hub password (or token).**
  - **ID: Name it something like dockerhub-creds (the same name used in the Jenkinsfile).**



Dashboard > Manage Jenkins > Credentials

Credentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	jenkins-slave	root
		System	(global)	docker-hub-credentials	barathkumar29/******
		System	(global)	Github_test	jkbarathkumar/******
		System	(global)	GitHub_PAT	GitHub_PAT

Stores scoped to Jenkins

P	Store ↓	Domains
	System	(global)
	Kubernetes	(global)

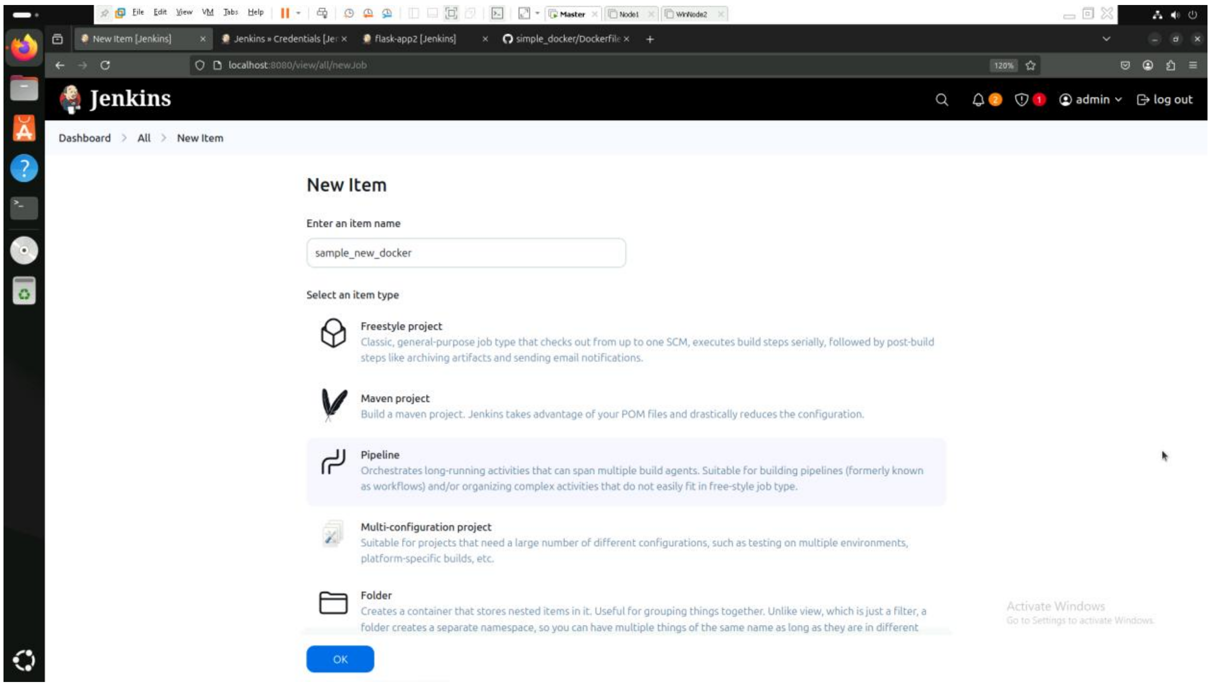
Icon: S M L

Activate Windows
Go to Settings to activate Windows.

REST API Jenkins 2.492.1

4. Create a New Pipeline in Jenkins

- **In Jenkins, click New Item > Pipeline.**
- **Enter a name for the pipeline.**
- **Under Pipeline Definition, select Pipeline script from SCM.**
  - **Select Git as the SCM.**
  - **Enter the GitHub repository URL (https://github.com/your-username/my-flask-app.git).**
  - **Set the branch (typically master or main).**
- **Click Save.**


Jenkins

Dashboard > All > New Item

New Item

Enter an item name

sample_new_docker

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

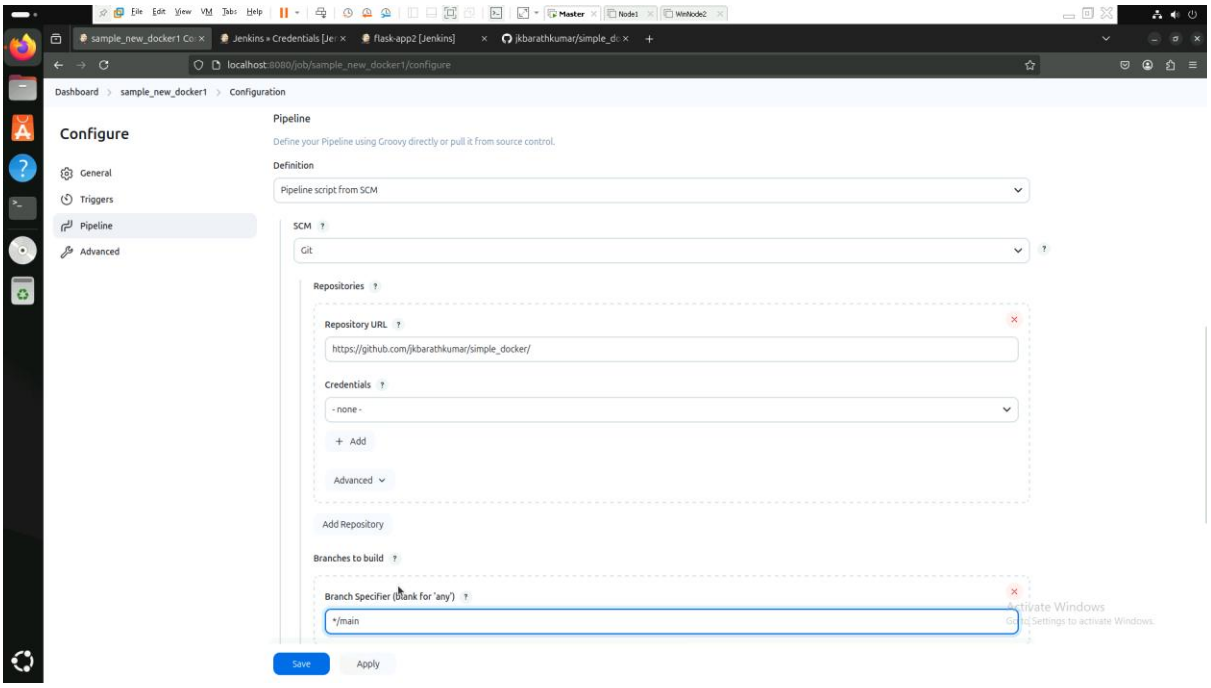
Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different

OK


Dashboard > sample_new_docker1 > Configuration

Configure

General
Triggers
Pipeline
Advanced

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/jkbarathkumar/simple_docker/

Credentials: - none -

+ Add

Advanced

Add Repository

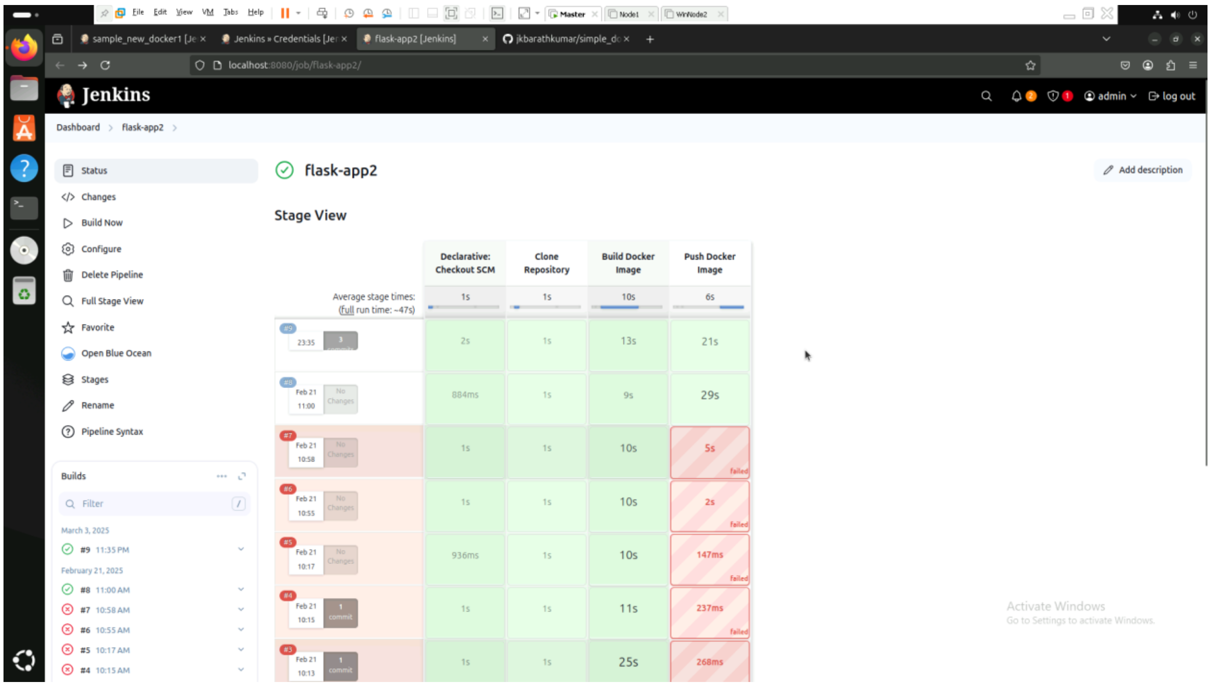
Branches to build

Branch Specifier (blank for 'any'): */main

Save Apply

5. Click Build Now

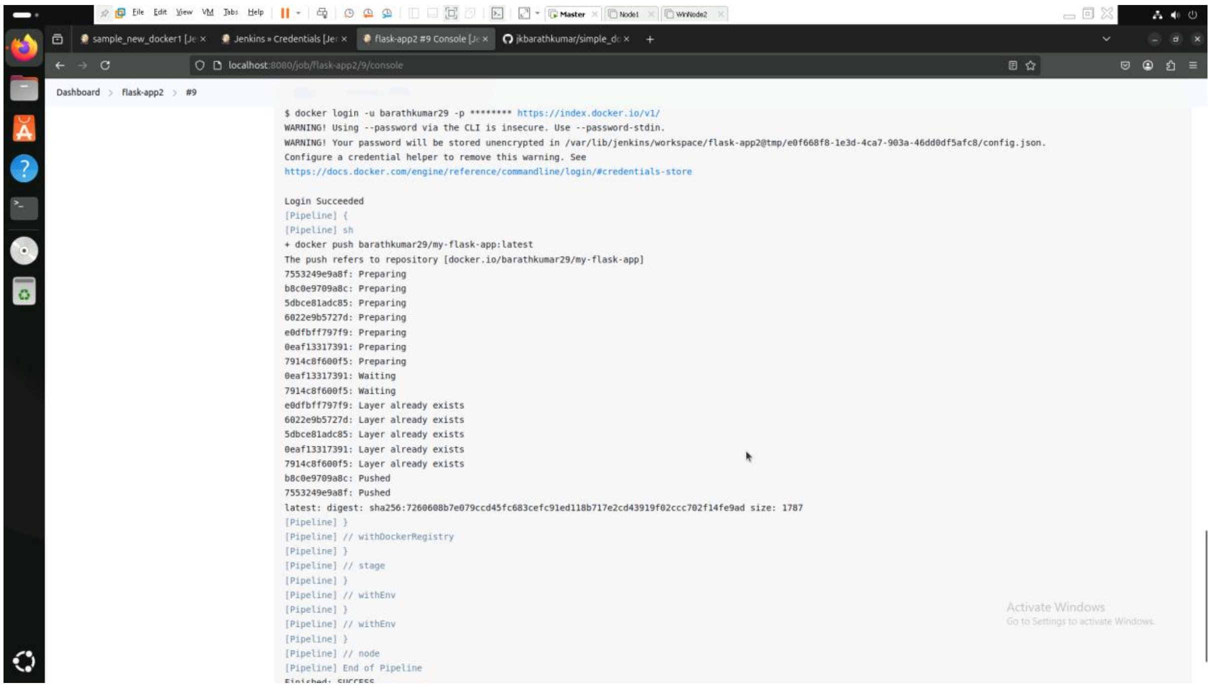
- **Click Build Now in Jenkins to trigger the build.**
- **Jenkins will:**
  - **Checkout the code from GitHub.**
  - **Build the Docker image.**
  - **Push the image to Docker Hub.**



flask-app2

Stage View

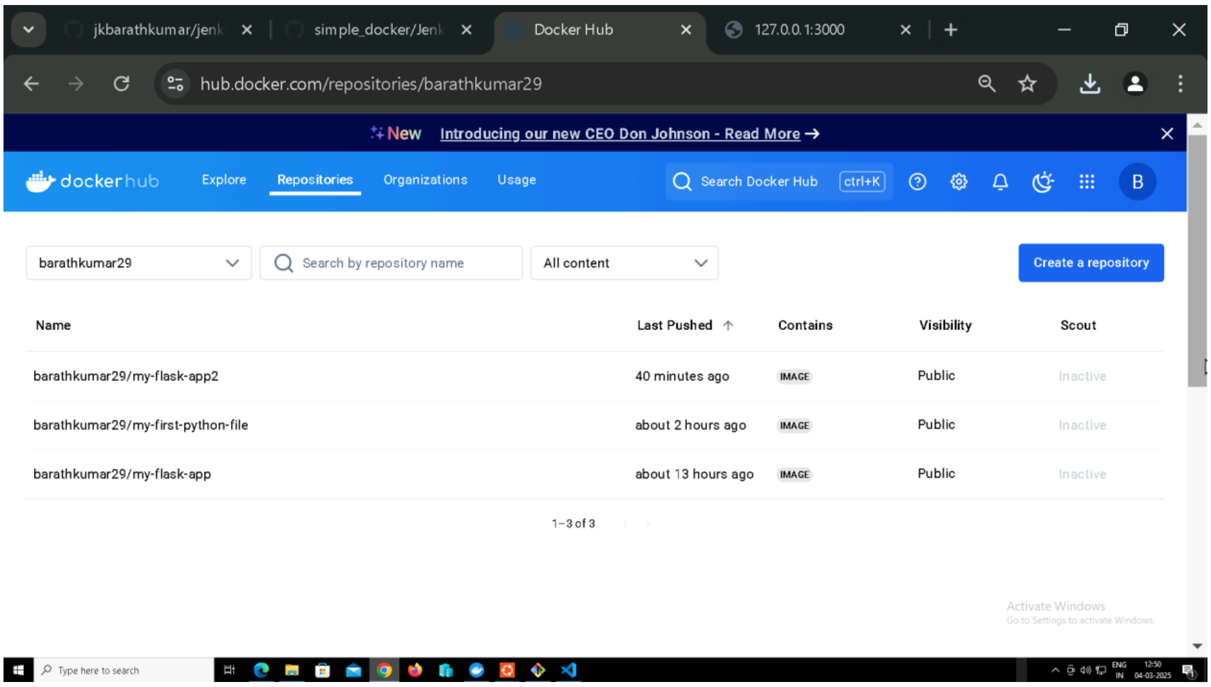
	Declarative: Checkout SCM	Clone Repository	Build Docker Image	Push Docker Image
Average stage times: (full run time: ~47s)	1s	1s	10s	6s
#9 23:35 3 commits	2s	1s	13s	21s
#8 Feb 21 11:00 No Changes	884ms	1s	9s	29s
#7 Feb 21 10:58 No Changes	1s	1s	10s	5s failed
#6 Feb 21 10:55 No Changes	1s	1s	10s	2s failed
#5 Feb 21 10:17 No Changes	936ms	1s	10s	147ms failed
#4 Feb 21 10:15 1 commit	1s	1s	11s	237ms failed
#3 Feb 21 10:13 1 commit	1s	1s	25s	268ms


sample_new_docker1 [Je
Jenkins » Credentials [Je
flask-app2 #9 Console [Je
jkbarathkumar/simple_dc
localhost:8080/job/flask-app2/9/console
Dashboard > flask-app2 > #9
Activate Windows
Go to Settings to activate Windows.

```
$ docker login -u barathkumar29 -p ******** https://index.docker.io/v1/
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /var/lib/jenkins/workspace/flask-app2@tmp/e0f668f8-1e3d-4ca7-903a-46dd0df5afc8/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[Pipeline] {
[Pipeline] sh
+ docker push barathkumar29/my-flask-app:latest
The push refers to repository [docker.io/barathkumar29/my-flask-app]
7553249e9a8f: Preparing
b8c0e9709a8c: Preparing
5dbce81adc85: Preparing
6022e9b5727d: Preparing
e0dfbff797f9: Preparing
0eaf13317391: Preparing
7914c8f600f5: Preparing
0eaf13317391: Waiting
7914c8f600f5: Waiting
e0dfbff797f9: Layer already exists
6022e9b5727d: Layer already exists
5dbce81adc85: Layer already exists
0eaf13317391: Layer already exists
7914c8f600f5: Layer already exists
b8c0e9709a8c: Pushed
7553249e9a8f: Pushed
latest: digest: sha256:7260608b7e079ccd45fc683cefc91ed118b717e2cd43919f02ccc702f14fe9ad size: 1787
[Pipeline] }
[Pipeline] // withDockerRegistry
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

6. Verify Docker Image on Docker Hub

- After the build finishes, log into your Docker Hub account.
- You should see the my-flask-app image under Repositories with the latest tag.

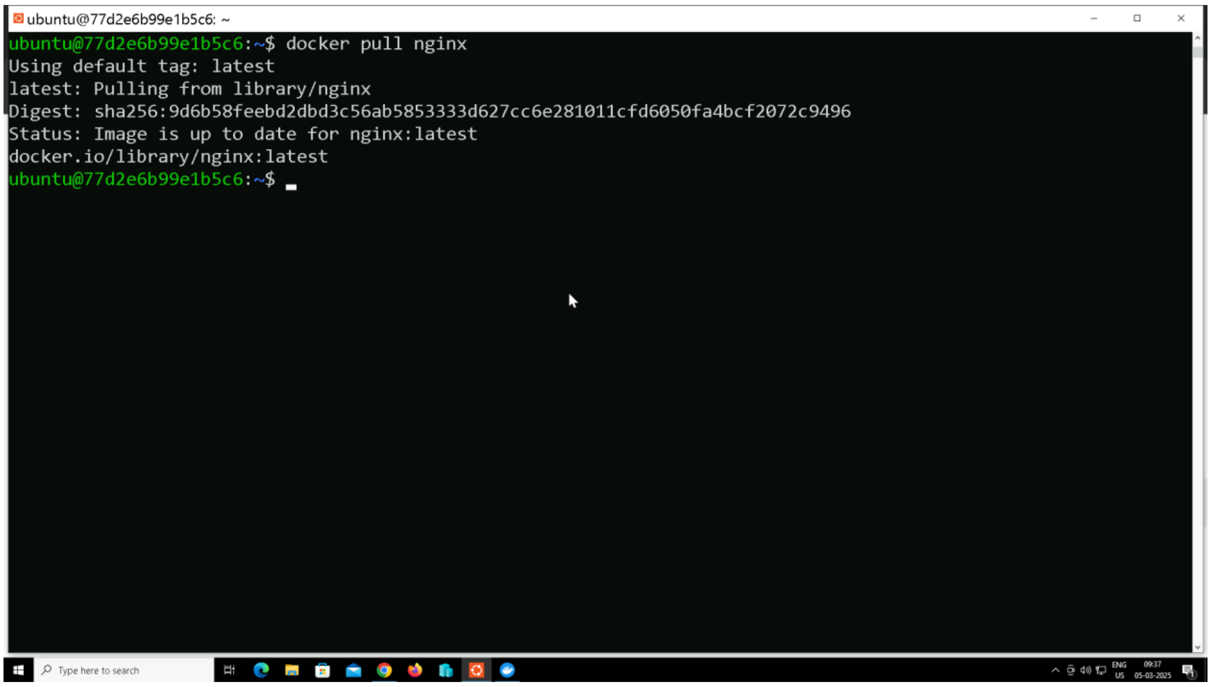


Name	Last Pushed	Contains	Visibility	Scout
barathkumar29/my-flask-app2	40 minutes ago	IMAGE	Public	Inactive
barathkumar29/my-first-python-file	about 2 hours ago	IMAGE	Public	Inactive
barathkumar29/my-flask-app	about 13 hours ago	IMAGE	Public	Inactive

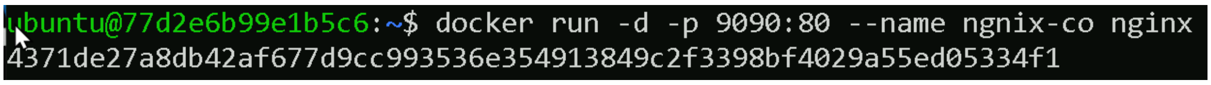
1–3 of 3

Project 2

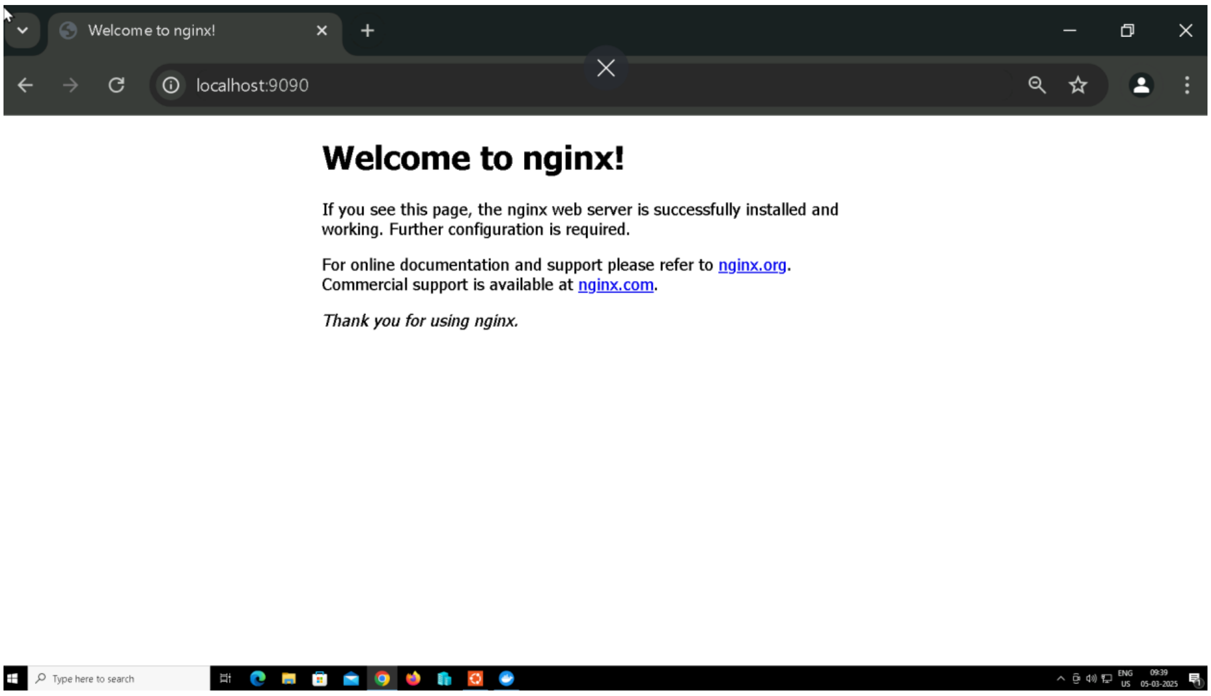
Deploying Ngnix server with Docker



```
ubuntu@77d2e6b99e1b5c6:~$ docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
Digest: sha256:9d6b58feebd2dbd3c56ab5853333d627cc6e281011cfd6050fa4bcf2072c9496
Status: Image is up to date for nginx:latest
docker.io/library/nginx:latest
ubuntu@77d2e6b99e1b5c6:~$
```



```
ubuntu@77d2e6b99e1b5c6:~$ docker run -d -p 9090:80 --name ngnix-co nginx
4371de27a8db42af677d9cc993536e354913849c2f3398bf4029a55ed05334f1
```



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.