TASK2:

DOCKER PIPELINE

Installing Docker

- **Why?**
    - Docker is required to **containerize the application**, making it **portable and consistent** across different environments.

Installing Maven

- **Why?**
    - Maven is a build automation tool used for **Java applications**.
    - It helps in **compiling, packaging, and managing dependencies**.

Setting Up DockerHub Credentials in Jenkins

- **Why?**
    - Jenkins needs access to **DockerHub** to push and pull images.
    - A **Personal Access Token (PAT)** is used instead of a password for security.
- **Steps Taken:**

1. **Generated a DockerHub Token**
    - Navigated to **DockerHub → Account Settings → Personal Access Token**.
2. **Stored Credentials in Jenkins**
    - **Jenkins Dashboard → Manage Jenkins → Credentials → Add Credentials**.
    - Used **DockerHub username** and **generated token** as a password.
    - Created an **ID: docker-token** for use in the pipeline.

Creating a Jenkins Pipeline

- **Why?**
    - The pipeline automates **building, tagging, and pushing a Docker image** to DockerHub.

Pipeline Code:

```
pipeline {
agent any

environment {
DOCKERHUB_CREDENTIALS = credentials('token1')
APP_NAME = "malar751/docker-pipeline"
}

stages {
stage('SCM Checkout') {
steps {
git branch: 'main', url: 'https://github.com/lily4499/lil-node-app.git'
}
}
-
stage('Build Docker Image') {
steps {
sh 'docker build -t $APP_NAME:$BUILD_NUMBER .'
}
}

stage('Login to DockerHub') {
steps {
sh 'echo $DOCKERHUB_CREDENTIALS_PSW | docker login -u $DOCKERHUB_CREDENTIALS_USR
--password-stdin'
}
}

stage('Push Docker Image') {
steps {
sh 'docker push $APP_NAME:$BUILD_NUMBER'
}
}
}
}
```

Your repository should contain:

- Application code (index.js for Node.js, for example).
- A valid Dockerfile.
- A package.json (if it's a Node.js project).

If these files exist, the pipeline will run **successfully** and deploy your application.

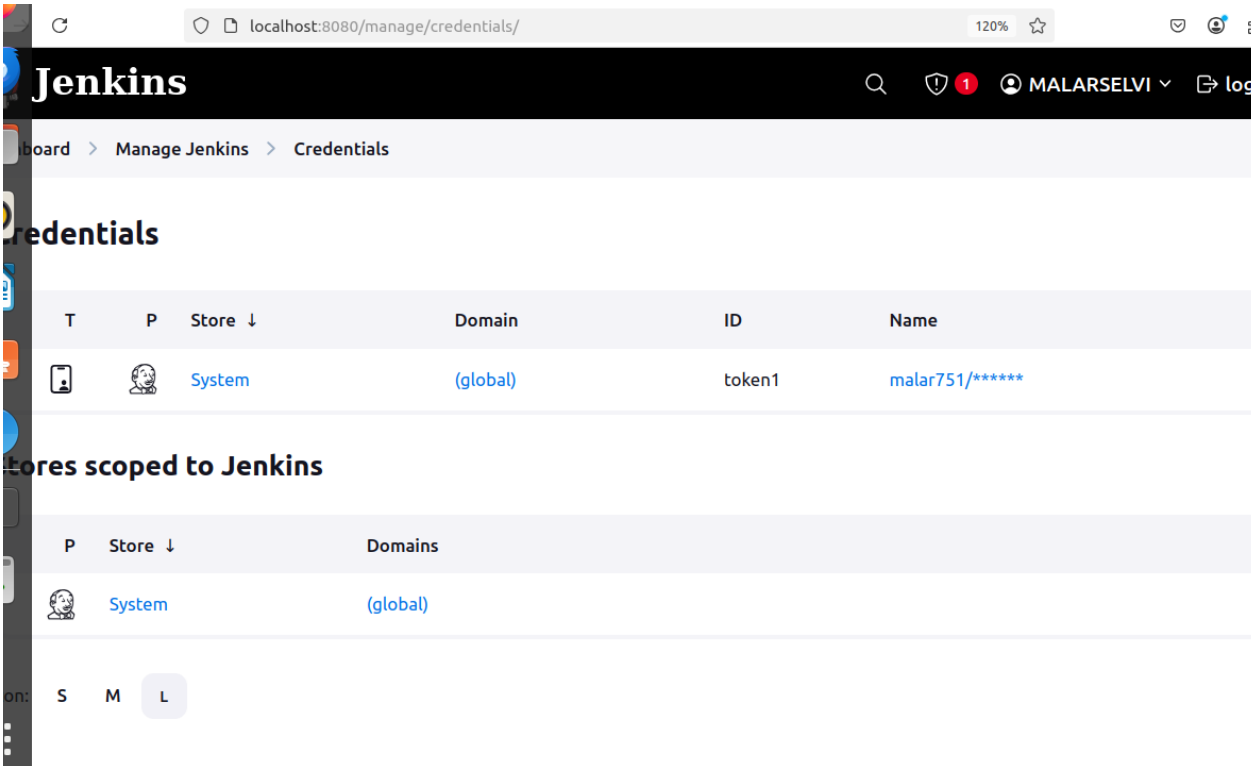
EXAMPLE: https://github.com/lily4499/lil-node-app.git

Jenkins handles:

- Pulling the latest code from GitHub.
- Building a Docker image.
- Pushing it to DockerHub.
- Deploying it automatically.

Final Outcome

- Automated the Docker image build process inside Jenkins.
- Stored images in DockerHub for easy deployment.
- Ensured repeatability of deployment across environments.

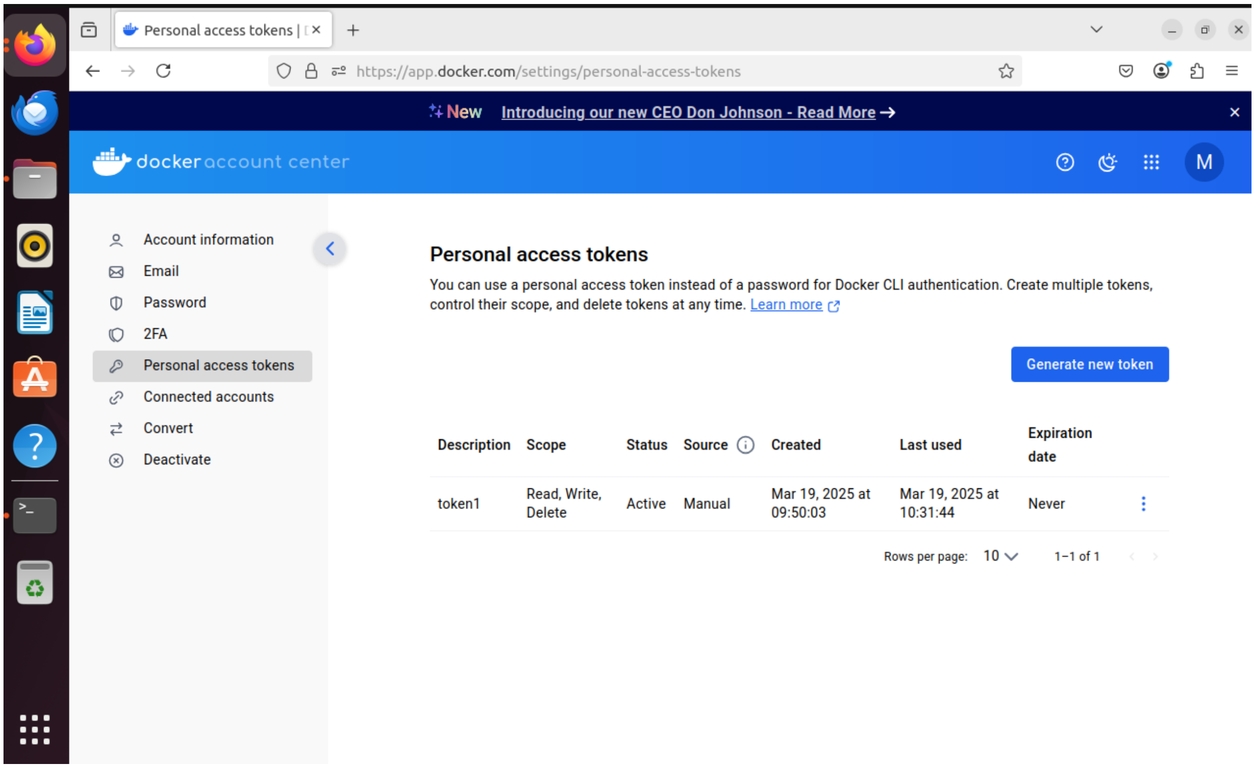

localhost:8080/manage/credentials/
120%
Jenkins
1
MALARSELVI
log
board > Manage Jenkins > Credentials

redentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	token1	malar751/******

tores scoped to Jenkins

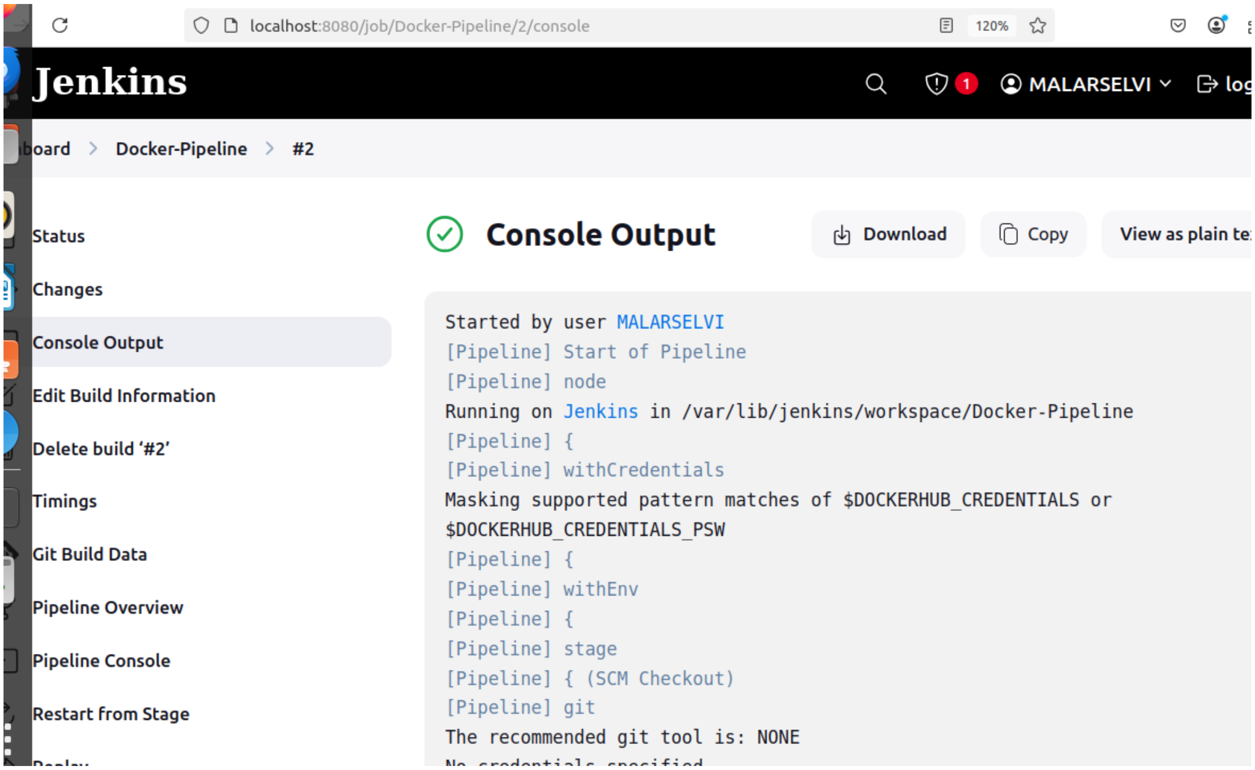
P	Store ↓	Domains
	System	(global)


Personal access tokens | [
https://app.docker.com/settings/personal-access-tokens
New Introducing our new CEO Don Johnson - Read More →
docker account center
M
Account information
Email
Password
2FA
Personal access tokens
Connected accounts
Convert
Deactivate

Personal access tokens

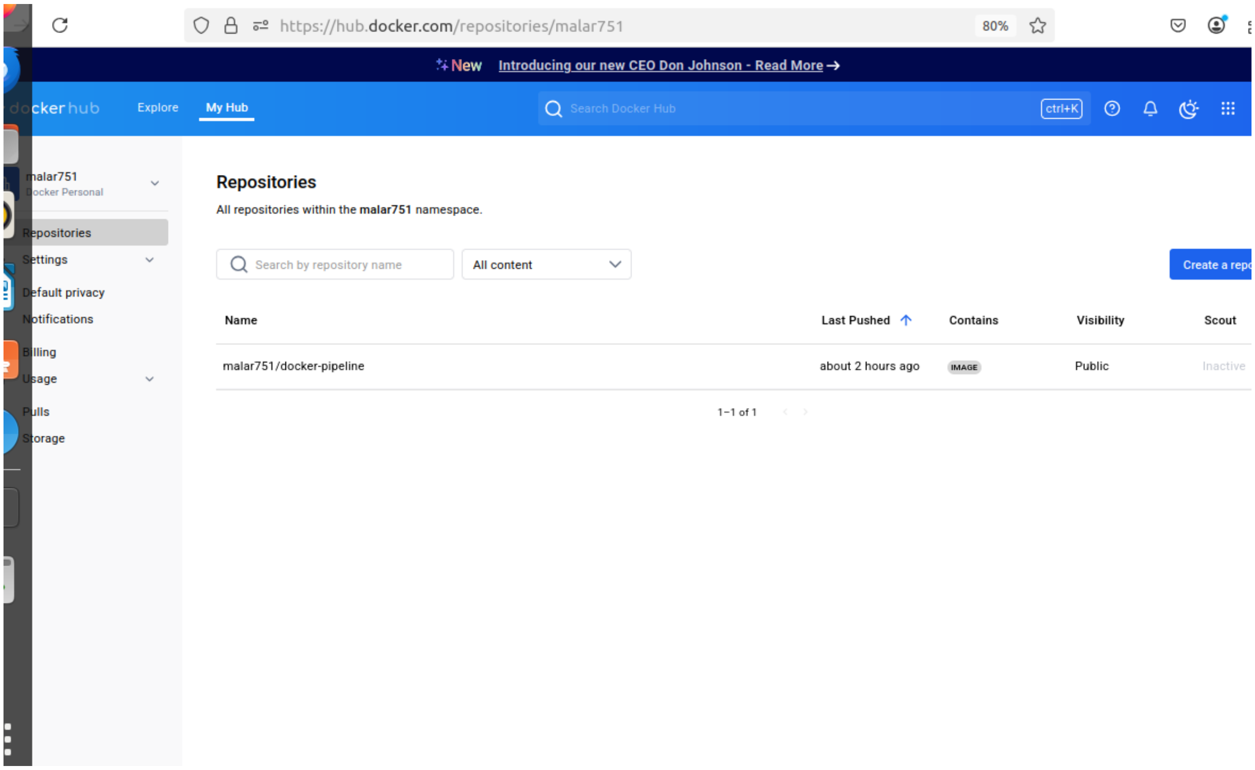
You can use a personal access token instead of a password for Docker CLI authentication. Create multiple tokens, control their scope, and delete tokens at any time. Learn more

Description	Scope	Status	Source	Created	Last used	Expiration date
token1	Read, Write, Delete	Active	Manual	Mar 19, 2025 at 09:50:03	Mar 19, 2025 at 10:31:44	Never



Console Output

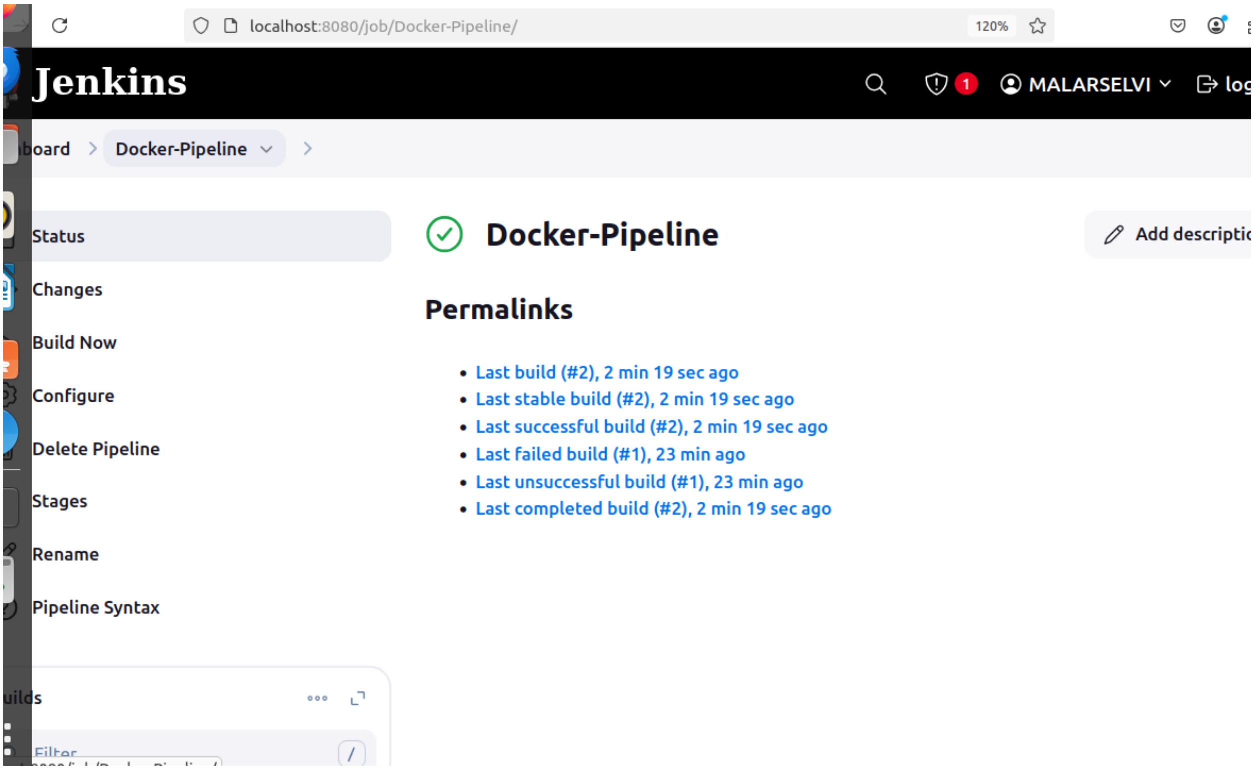
```
Started by user MALARSELVI
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/Docker-Pipeline
[Pipeline] {
[Pipeline] withCredentials
Masking supported pattern matches of $DOCKERHUB_CREDENTIALS or
$DOCKERHUB_CREDENTIALS_PSW
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (SCM Checkout)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
```



Repositories

All repositories within the **malar751** namespace.

Name	Last Pushed	Contains	Visibility	Scout
malar751/docker-pipeline	about 2 hours ago	IMAGE	Public	Inactive



localhost:8080/job/Docker-Pipeline/

Jenkins

MALARSELVI

board > Docker-Pipeline >

Docker-Pipeline

Permalinks

- Last build (#2), 2 min 19 sec ago
- Last stable build (#2), 2 min 19 sec ago
- Last successful build (#2), 2 min 19 sec ago
- Last failed build (#1), 23 min ago
- Last unsuccessful build (#1), 23 min ago
- Last completed build (#2), 2 min 19 sec ago