

PHASE 3- SOLUTION DEVELOPMENT

PROJECT TITLE :- Setting up a CI/CD pipeline for automated deployment

COLLEGE NAME:-Dr. SMCE

GROUP MEMBERS:-

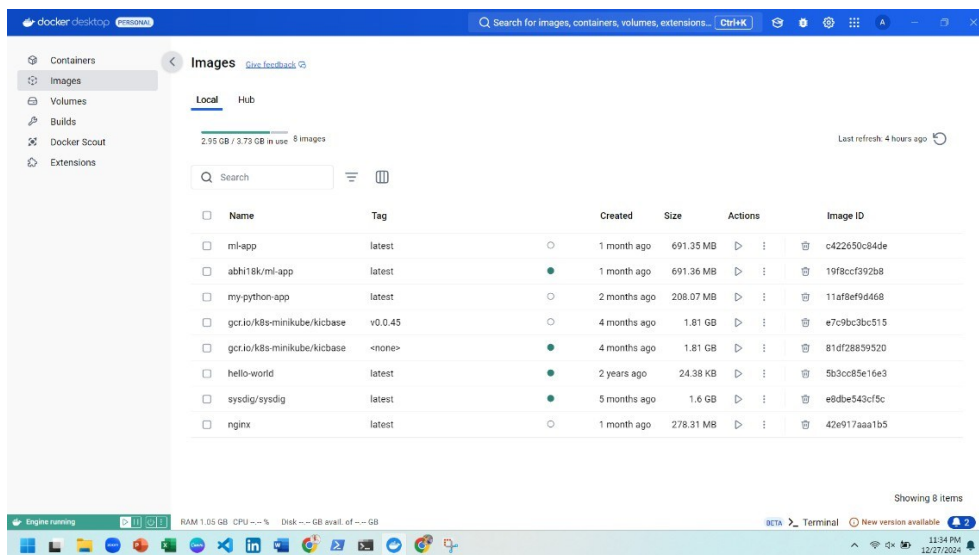
1. Amith C Y [USN:- 1CC21CS004]
2. Charan K M [USN:- 1CC21CS017]
3. Nitin Teja N[USN:-1CC21CS040]
4. Sunil C M [USN:-1CC21CS057]

SOLUTION DEVELOPMENT:

Implementing Containerization and Running Locally

Step 1: Set Up the Development Environment

1. Install Docker for containerization.



2. Install Kubernetes (Minikube) for local container orchestration.

```
C:\Users\abhi\Desktop\Devops\ml-app>minikube start
* minikube v1.34.0 on Microsoft Windows 11 Home Single Language 10.0.26100.2314 Build 26100.2314
* Using the docker driver based on existing profile
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.45 ...
* Updating the running docker "minikube" container ...
! Failing to connect to https://registry.k8s.io/ from both inside the minikube container and host machine
* To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/n
* Preparing Kubernetes v1.31.0 on Docker 27.2.0 ...
* Verifying Kubernetes components...
  - Using image docker.io/kubernetes/metrics-scraper:v1.0.8
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
  - Using image docker.io/kubernetes/dashboard:v2.7.0
* Some dashboard features require the metrics-server addon. To enable all features please run:

    minikube addons enable metrics-server

* Enabled addons: default-storageclass, storage-provisioner, dashboard
* Done! kubectrl is now configured to use "minikube" cluster and "default" namespace by default
```

```
C:\Users\abhi\Desktop\Devops\ml-app>minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

3. Install Jenkins for CI/CD pipeline automation.

The screenshot shows the Jenkins Dashboard interface. At the top, there's a search bar and user information. The main content area displays a table of builds for the 'DevSecOps Pipeline'.

S	W	Name	Last Success	Last Failure	Last Duration
⊗	🌩	DevSecOps Pipeline	N/A	2 mo 0 days #8	1.6 sec

Below the table, there are filters for 'Icon: S M L' and a 'Build Queue' section showing 'No builds in the queue.' and 'Build Executor Status' showing '0/2'.

REST API Jenkins 2.479.2

4. Install Git for version control and repository management.

```
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\abhi>git --version
git version 2.47.1.windows.1

C:\Users\abhi>|
```

5. Install SonarQube, Trivy, OWASP ZAP for security scanning and vulnerability

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\abhi> cd C:\Tools\Trivy
PS C:\Tools\Trivy> .\trivy.exe --version
Version: 0.58.1
PS C:\Tools\Trivy> trivy image myimage:latest
2024-12-27T21:10:12+05:30      INFO    [vulndb] Need to update DB
2024-12-27T21:10:12+05:30      INFO    [vulndb] Downloading vulnerability DB...
2024-12-27T21:10:12+05:30      INFO    [vulndb] Downloading artifact...      repo="mirror.gcr.io/aquasec/trivy-db:2"
37.96 MiB / 37.96 MiB [-----] 100.00% 516.71 KiB p/s 1m55s
2024-12-27T21:12:09+05:30      INFO    [vulndb] Artifact successfully downloaded      repo="mirror.gcr.io/aquasec/trivy-db:2"
2024-12-27T21:12:09+05:30      INFO    [vuln] Vulnerability scanning is enabled
2024-12-27T21:12:09+05:30      INFO    [secret] Secret scanning is enabled
2024-12-27T21:12:09+05:30      INFO    [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2024-12-27T21:12:09+05:30      INFO    [secret] Please see also https://aquasecurity.github.io/trivy/v0.58/docs/scanner/secret#recommendation for faster se
cret detection
2024-12-27T21:12:14+05:30      FATAL   Fatal error      image scan error: scan error: unable to initialize a scanner: unable to initialize an image scanner:
unable to find the specified image "myimage:latest" in ["docker" "containerd" "podman" "remote"]: 4 errors occurred:
* docker error: unable to inspect the image (myimage:latest): Error response from daemon: No such image: myimage:latest
* containerd error: containerd socket not found: /run/containerd/containerd.sock
* podman error: unable to initialize Podman client: no podman socket found: CreateFile podman/podman.sock: The system cannot find the path specified
* remote error: GET https://index.docker.io/v2/library/myimage/manifests/latest: UNAUTHORIZED: authentication required; [map[Action:pull Class: Name
:library/myimage Type:repository]]

PS C:\Tools\Trivy> docker pull nginx:latest
latest: Pulling from library/nginx
1e109dd2a8d7: Download complete
c44f27309ea1: Download complete
2b9b96c5d9e5: Download complete
566e42bce1c: Download complete
d6d74085ff8f: Download complete
da8cc133ff82: Download complete
bd98674871f5: Download complete
Digest: sha256:42e917aa1b5b40dd0f6f7f4f857490ac7747d7ef7b391c774a41a8b994f15
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
```

```
Command Prompt - vault ser
-12-16T14:00:53Z
2024-12-27T21:38:37.759+0530 [INFO] core: post-unseal setup complete
2024-12-27T21:38:37.759+0530 [INFO] core: root token generated
2024-12-27T21:38:37.759+0530 [INFO] core: pre-seal teardown starting
2024-12-27T21:38:37.759+0530 [INFO] core: pre-seal teardown complete
2024-12-27T21:38:37.759+0530 [INFO] core: cluster-listener.tcp: starting listener: listener_address=127.0.0.1:8201
2024-12-27T21:38:37.760+0530 [INFO] core: cluster-listener: serving cluster requests: cluster_listen_address=127.0.0.1:8201
2024-12-27T21:38:37.760+0530 [INFO] core: post-unseal setup starting
2024-12-27T21:38:37.760+0530 [INFO] core: loaded wrapping token key
2024-12-27T21:38:37.760+0530 [INFO] core: successfully setup plugin runtime catalog
2024-12-27T21:38:37.760+0530 [INFO] core: successfully setup plugin catalog: plugin-directory=""
2024-12-27T21:38:37.761+0530 [INFO] core: successfully mounted: type=system version="v1.18.3+builtin.vault" path=sys/ namespace="ID: root. Path: "
2024-12-27T21:38:37.761+0530 [INFO] core: successfully mounted: type=identity version="v1.18.3+builtin.vault" path=identity/ namespace="ID: root. Path: "
2024-12-27T21:38:37.761+0530 [INFO] core: successfully mounted: type=cubbyhole version="v1.18.3+builtin.vault" path=cubbyhole/ namespace="ID: root. Path: "
2024-12-27T21:38:37.762+0530 [INFO] core: successfully mounted: type=token version="v1.18.3+builtin.vault" path=token/ namespace="ID: root. Path: "
2024-12-27T21:38:37.762+0530 [INFO] core: rollback: Starting the rollback manager with 256 workers
2024-12-27T21:38:37.762+0530 [INFO] core: rollback: starting rollback manager
2024-12-27T21:38:37.762+0530 [INFO] core: restoring leases
2024-12-27T21:38:37.763+0530 [INFO] expiration: lease restore complete
2024-12-27T21:38:37.763+0530 [INFO] identity: entities restored
2024-12-27T21:38:37.763+0530 [INFO] identity: groups restored
2024-12-27T21:38:37.763+0530 [INFO] core: post-unseal setup complete
2024-12-27T21:38:37.763+0530 [INFO] core: vault is unsealed
2024-12-27T21:38:37.769+0530 [INFO] core: successful mount: namespace="" path=secret/ type=kv version="v0.20.0+builtin"

WARNING! dev mode is enabled! In this mode, Vault runs entirely in-memory
and starts unsealed with a single unseal key. The root token is already
authenticated to the CLI, so you can immediately begin using Vault.

You may need to set the following environment variables:

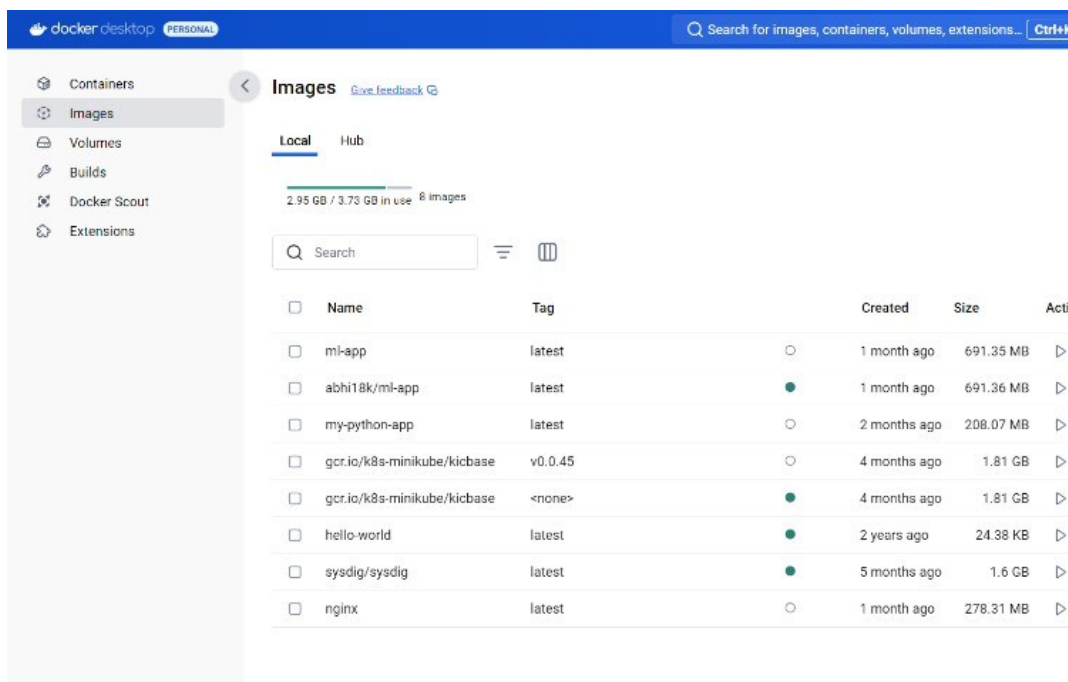
PowerShell:
$env:VAULT_ADDR="http://127.0.0.1:8200"
cmd.exe:
set VAULT_ADDR=http://127.0.0.1:8200

The unseal key and root token are displayed below in case you want to
seal/unseal the Vault or re-authenticate.

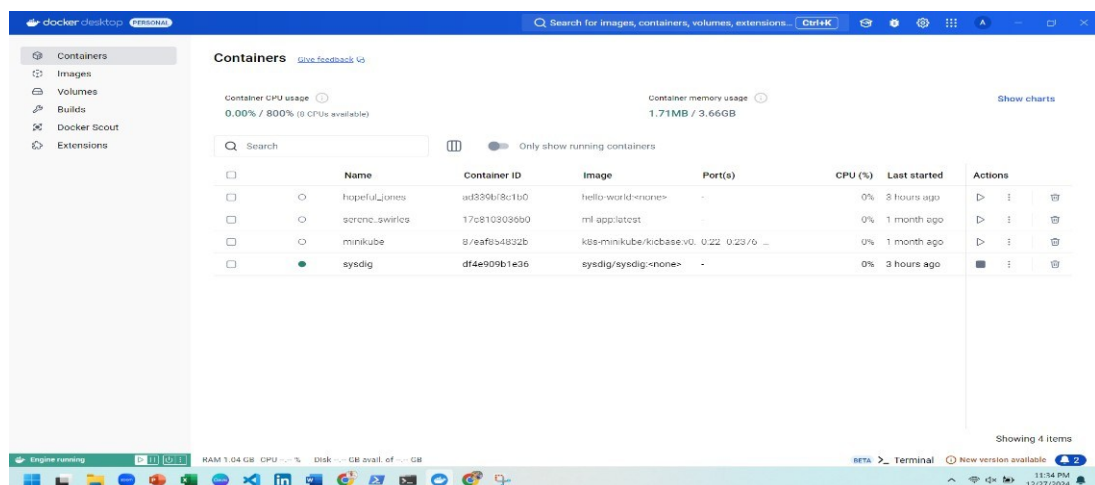
Unseal Key: 2yuEAYhi2X+uvxeWIAgrMMdyZIpNJTii2e9JncH7ji=
Root Token: hvs.ng55LLpXmMabP4Hx3Wsv9cjf
```

Step 2: Containerizing the Application

1. Create a Docker file for the application:
 - Define the base image (e.g., node:16-alpine).
 - Set up the working directory and dependencies.
 - Expose required ports.
 - Define the startup command (CMD).

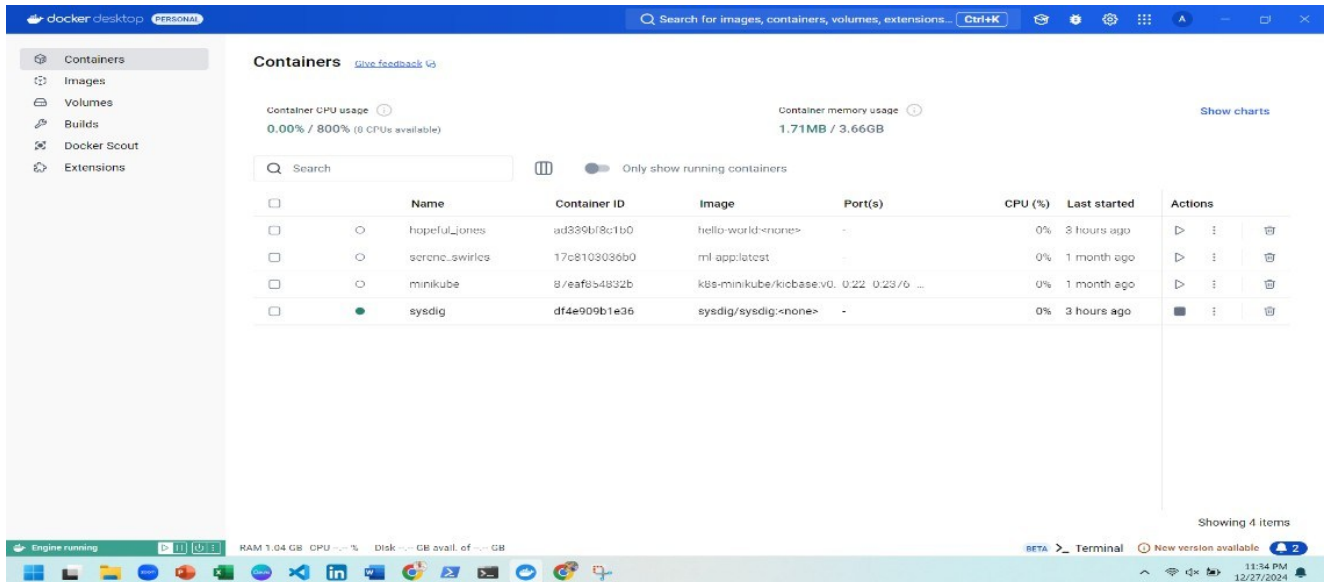


2. Build and run the container locally:



3. `docker build -t ml-app .`

`docker run -d -p 8080:8080 ml-app`



Step 3: Implement CI/CD Security

1. Set up Jenkins for automating CI/CD pipeline.
2. Integrate SonarQube for Static Application Security Testing (SAST).
3. Automate security scanning in the CI/CD pipeline using OWASP ZAP.
4. Deploy to a Kubernetes cluster using Minikube.