1.Introduction

Docker is a platform that enables developers and system administrators to build, ship, and run applications in containers. In DevOps, Docker streamlines development, testing, and deployment by providing consistent environments across stages.

2.Why Docker in DevOps?

- Consistency: Eliminates the “works on my machine” issue.
- CI/CD Friendly: Integrates well with Jenkins, GitHub Actions, GitLab CI, etc.
- Lightweight & Fast: Containers use fewer resources than virtual machines.
- Isolation & Security: Each container runs in its own isolated environment.

3. Docker Installation:-

- Windows (Using Git Bash)
- Download Docker Desktop: https://www.docker.com/products/docker-desktop/
- Install and launch Docker Desktop.
- Run this in Git Bash to verify installation:

```
bash

CopyEdit

docker --version
```

- Linux

```
bash

CopyEdit

sudo apt update

sudo apt install docker.io -y

sudo systemctl start docker

sudo systemctl enable docker

docker --version
```

4. Basic Docker Commands

Command	Description
`docker --version`	Check Docker version
`docker pull <image>`	Download image from Docker Hub
`docker images`	List downloaded images

Command	Description
`docker run <image>`	Run a container
`docker ps`	List running containers
`docker stop <container_id>`	Stop a container
`docker rm <container_id>`	Remove a container
`docker rmi <image_id>`	Remove an image

5. Dockerfile: Create our Own Image

- Dockerfile
- CopyEdit
- Dockerfile Example
- FROM python:3.9-slim
- WORKDIR /app
- COPY . /app
- RUN pip install -r requirements.txt
- CMD ["python", "app.py"]

7.Build and run:

- bash
- CopyEdit
- docker build -t my-python-app .
- docker run -p 5000:5000 my-python-app

6.Docker Compose (Multi-Container App)

- docker-compose.yml

```
yaml

CopyEdit

version: '3'

services:

  web:

    build: .

    ports:

      - "5000:5000"
```

```
  redis:
    image: "redis:alpine"
```

- Run using:

```
bash
CopyEdit
docker-compose up
```

Real-Time Use Case Example:-

- Project: Web App Deployment
- Build: Flask app with Dockerfile
- CI/CD: Jenkins Pipeline triggers on GitHub push
- Containerization: Docker Image created & pushed
- Deployment: Deployed using Docker on EC2