Docker

A Guide to Containerization

What is Docker?

- Docker is an open-source platform that enables developers to build, package, and deploy applications as lightweight containers.
- It provides an environment to run applications with all dependencies included.
- Containers ensure consistency across different environments.

Benefits of Docker

- Portability: Works on any system that supports Docker.
- Efficiency: Uses fewer resources than virtual machines.
- Scalability: Easily scales applications across multiple servers.
- Faster Deployment: Containers start in seconds.
- Consistency: Eliminates environment-specific issues.

Docker vs Virtual Machines

- Virtual Machines run full OS with allocated resources.
- Docker uses shared OS kernel, making it lightweight.
- VMs take more time to boot, while containers start quickly.
- Containers use less system memory compared to VMs.

Key Docker Components

- Docker Engine: Core of Docker for building and running containers.
- Docker Image: A template with the necessary environment and application.
- Docker Container: A running instance of a Docker Image.
- Docker Hub: A registry for sharing and storing container images.
- Docker Compose: A tool for managing multi-container applications.

Docker Workflow

- Write a Dockerfile to define the application environment.
- Build an image using `docker build`.
- Run a container from the image using `docker run`.
- Manage running containers with `docker ps`, `docker stop`, etc.
- Push images to Docker Hub or private registries.

Common Docker Commands

- docker --version` → Check Docker version
- docker images` → List available images
- docker ps` → Show running containers
- docker run -d -p 8080:80 nginx` → Run a container in detached mode
- docker stop <container_id>` → Stop a running container
- docker rm <container_id>` → Remove a container
- docker rmi <image_id>` → Remove an image

Dockerfile Example

- A simple Dockerfile for a Python application

```
FROM python:3.8
WORKDIR /app
COPY . /app
RUN pip install -r requirements.txt
CMD ["python", "app.py"]
```

Running a Container

1. Build the image:

   docker build -t myapp .

2. Run the container:

   docker run -d -p 5000:5000 myapp

3. View running containers:

   docker ps

Docker Compose

- Used to define and manage multi-container applications.
- Uses a `docker-compose.yml` file to specify services.
- Example:

```
version: '3'
services:
  web:
    image: nginx
    ports:
      - "8080:80"
```

Conclusion

- Docker simplifies application deployment and scaling.
- Containers provide consistency across different environments.
- Learn to use Docker with Dockerfiles, Compose, and common CLI commands.
- Practice by containerizing your applications.