□ Che cos'è Docker

Docker è una piattaforma per creare, eseguire e distribuire applicazioni dentro **container**, ambienti leggeri e isolati che contengono tutto il necessario per far funzionare un'applicazione (codice, librerie, dipendenze, ecc).

□□ Come si usa Docker

Comandi base:

- Visualizzare container attivi: `docker ps`
- Visualizzare tutte le immagini: `docker images`
- Accedere dentro un container: `docker exec -it nome-container bash`
- Fermare un container: `docker stop nome-container`
- Eliminare un container: `docker rm nome-container`
- Eliminare un'immagine: `docker rmi nome-immagine`

□ Come si crea un'immagine Docker

1. Crea un file chiamato `Dockerfile`, ad esempio:

```
FROM node:18
WORKDIR /app
COPY . .
RUN npm install
CMD ["node", "index.js"]
```

2. Costruisci l'immagine:

docker build -t nome-immagine .

□ Come si esegue un'immagine Docker (run)

docker run -d --name nome-container nome-immagine

- `-d`: esegue in background
- `--name`: assegna un nome al container

□ Come si esporta un'immagine Docker

docker save nome-immagine > immagine.tar

Per importarla su un altro sistema:

bash

```
docker load < immagine.tar
```

Come si crea un volume Docker

```
docker volume create nome-volume
```

Come si monta un volume in un container

```
docker run -v nome-volume:/percorso/container nome-immagine
```

```
Esempio:
docker run -v dati:/app/data myapp
```

Come si monta un'immagine (interattiva)

```
docker run -it nome-immagine /bin/bash
```

Che cos'è un file YAML

YAML è un formato leggibile usato per descrivere configurazioni in maniera gerarchica (indentata). È

usatissimo in Kubernetes e su AWS per definire pod, servizi, deployment, ecc.

Esempio:

```
apiVersion: v1
kind: Pod
metadata:
  name: mio-pod
spec:
  containers:
    - name: mio-container
      image: nginx
```

Come si usa YAML su AWS con Kubernetes

Su AWS si usa EKS (Elastic Kubernetes Service) per creare cluster Kubernetes.

Esempio di `deployment.yaml`:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mia-app
spec:
  replicas: 2
  selector:
    matchLabels:
      app: mia-app
  template:
    metadata:
      labels:
        app: mia-app
    spec:
      containers:
        - name: mio-container
          image: nome-immagine
          ports:
            - containerPort: 80
```

Comando per applicarlo:
kubectl apply -f deployment.yaml

□ Comandi Kubernetes utili

kubectl get pods
kubectl get deployments
kubectl describe pod nome-pod
kubectl delete pod nome-pod