Docker and Docker Swarm

- starts at 9:06 pm

AGENDA

More Instructions in Dockerfile

Multi Stage Builds

Docker Inspect

Building efficient image

Optimising Image

- Flattening a Docker Image
- Docker slim

Registry

→ More Instructions in Dockerfile

HEALTHCHECK

HEALTHCHECK <options> CMD command.

- --interval 30s
- --timeout 30s
- --start-period 0s

-- retries 3

FROM nginx:alpine

HEALTHCHECK --interval=30s --timeout=5s --retries=3 --start-period=5s CMD curl -f http://localhost/ || exit 1

docker build -t healthcheck-demo .

docker run -it -d healthcheck-demo → docker ps (states)

HEALTHCHECK NONE

→ ARG

→ only accessible during build time.

FROM alpine:latest

ARG GREETING="Hello"

RUN echo $GREETING

CMD ["sh", "-c", "echo Hi $GREETING"]

-- build-arg → used to pass value to ARG

→ ENV

```
FROM alpine:latest

ARG GREETING="Hello"

ENV GREETING=$GREETING

RUN echo $GREETING

CMD ["sh", "-c", "echo Hi $GREETING"]
```

```
docker build --build-arg GREETING="Hello, Docker!" -t env-example .
```

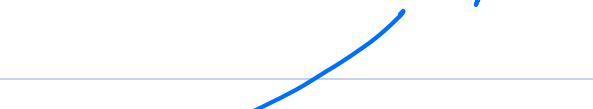
```
docker run -e GREETING="Vedant!" env-example
```

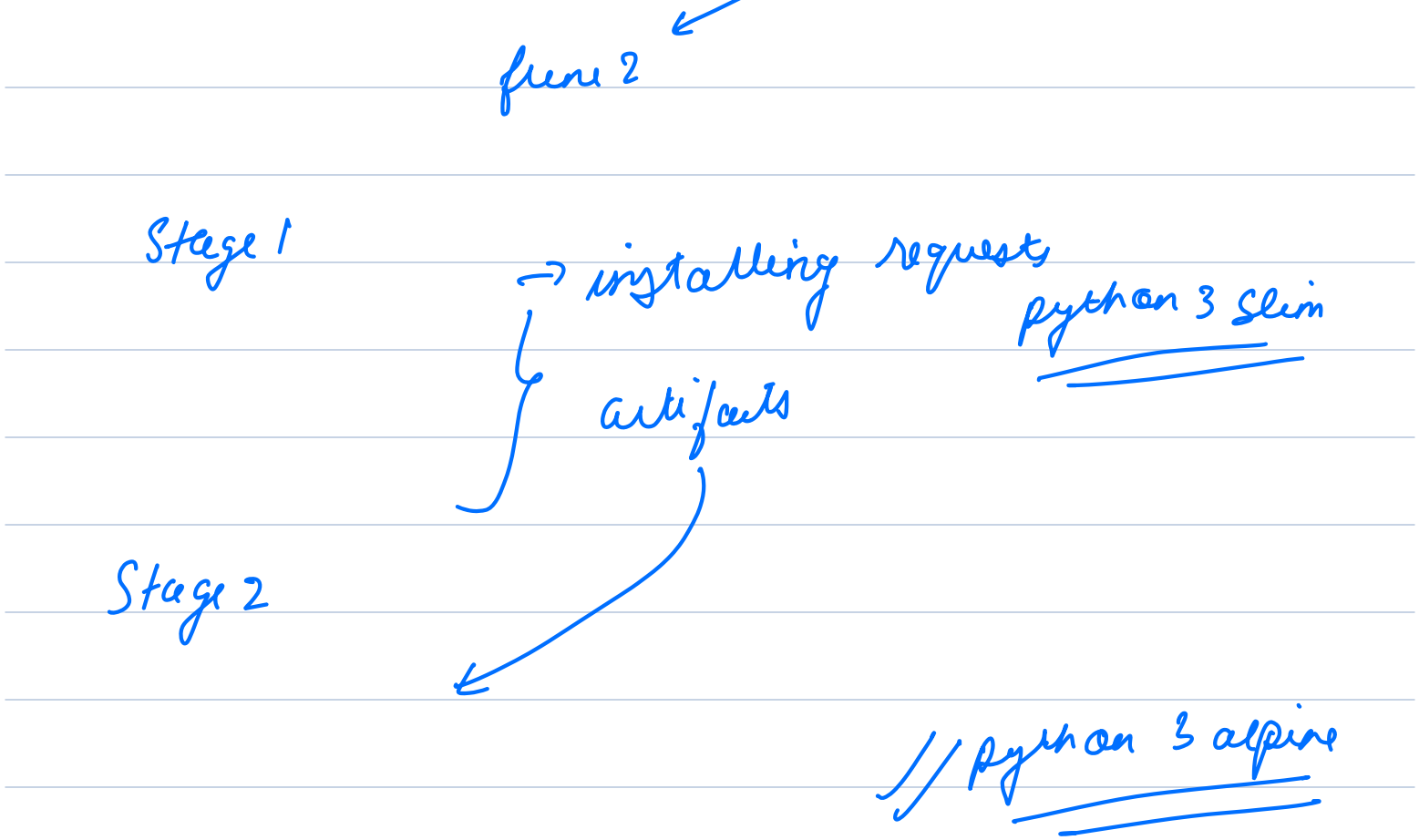
→ USER

```
FROM ubuntu:latest

RUN useradd -m myuser

USER myuser

CMD ["whoami"]
```

→ multi-stage Build.

func1 → output





Simple python.

→ script.py

→ requirement.txt

→ Dockerfile

→ script.py

```
import requests

def main():
    print("Hello from a Python script!")
    response = requests.get("https://api.github.com")
```

```
    print("GitHub API Status Code:", response.status_code)


if __name__ == "__main__":
    main()
```

requirements.txt

```
requests
chardet
```

Dockerfile

```
FROM python:3.10-slim AS build-stage

WORKDIR /app

COPY requirements.txt .

RUN pip install --no-cache-dir -r requirements.txt

COPY script.py .


FROM python:3.10-alpine

COPY --from=build-stage /usr/local/lib/python3.10/site-packages /usr/local/lib/python3.10/site-packages

COPY --from=build-stage /app/script.py /app/script.py

WORKDIR /app

CMD ["python", "script.py"]
```

Stage 1

Stage 2

Break → 10:30 pm

Docker Inspect

docker inspect [options] object

--format = '{{ .Id }}'

→ Building Efficient Images
  → faster deployment
  → easier to maintain
  → lower storage costs
  → better performance
  → reduced resource consumption.
  → security

why?

How?

① Multi-Stage Builds.

② Use minimal base Image.

③ Minimise the number of layers

```
RUN apt-get update

RUN apt-get install -y curl
```

```
RUN apt-get update && apt-get install -y curl
```

④ Clean up after installation

```
RUN apt-get update && apt-get install -y \
  curl \
  && rm -rf /var/lib/apt/lists/*
```

⑤ use the correct copy and ADD commands.

⑥ use .dockerignore file

.git
.log

tmp

tmp/*

→ Faster image builds.

⑦ use caching effectively.

```
COPY requirements.txt .
RUN pip install -r requirements.txt
COPY . .
```

→ V1

→ V2

⑧ Use ARGs for build time variables.

→ Flattening a Docker Image.

→ reducing the number of layer in docker image.

→ image size maybe reduced.

```
docker run --name flat-demo-container ubuntu
```

Export the Container Filesystem:

docker export flat-demo-container > flattened.tar

Import the Flattened Filesystem into a New Image:

cat flattened.tar | docker import - flattened-image

docker history.

→ Docker Slim

used for reducing image size

Pulled dslim image

↓

created container

docker pull dslim/slim

docker run -it -d --name slim-container -v /var/run/docker.sock:/var/run/docker.sock dslim/slim

docker exec -it slim-container docker-slim build nginx

docker run -d --name nginx-slim -p 8080:80 nginx.slim

curl http://localhost:8080

Docker Registry

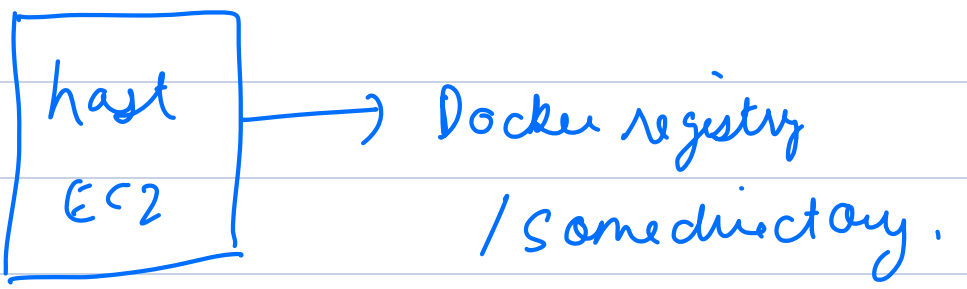
<username>/<repository>:<tag>

vedant120 / vedant120 : nginx-slim

docker tag nginx.slim vedant120/vedant120:latest

docker push vedant120/vedant120:latest

→ Setting up our Own Private Docker Registry.



docker pull registry:2

docker run -d -p 5000:5000 --name registry -v /opt/registry:/var/lib/registry registry:2

docker tag nginx.slim localhost:5000/vedant120/vedant120:latest

Push the Image to Your Private Registry

docker push localhost:5000/vedant120/vedant120:latest