

# DOCKER WORKSHOP

En introduksjon til bruk av Docker.

Bouvet backend gruppa, høsten 2023.



bouvet



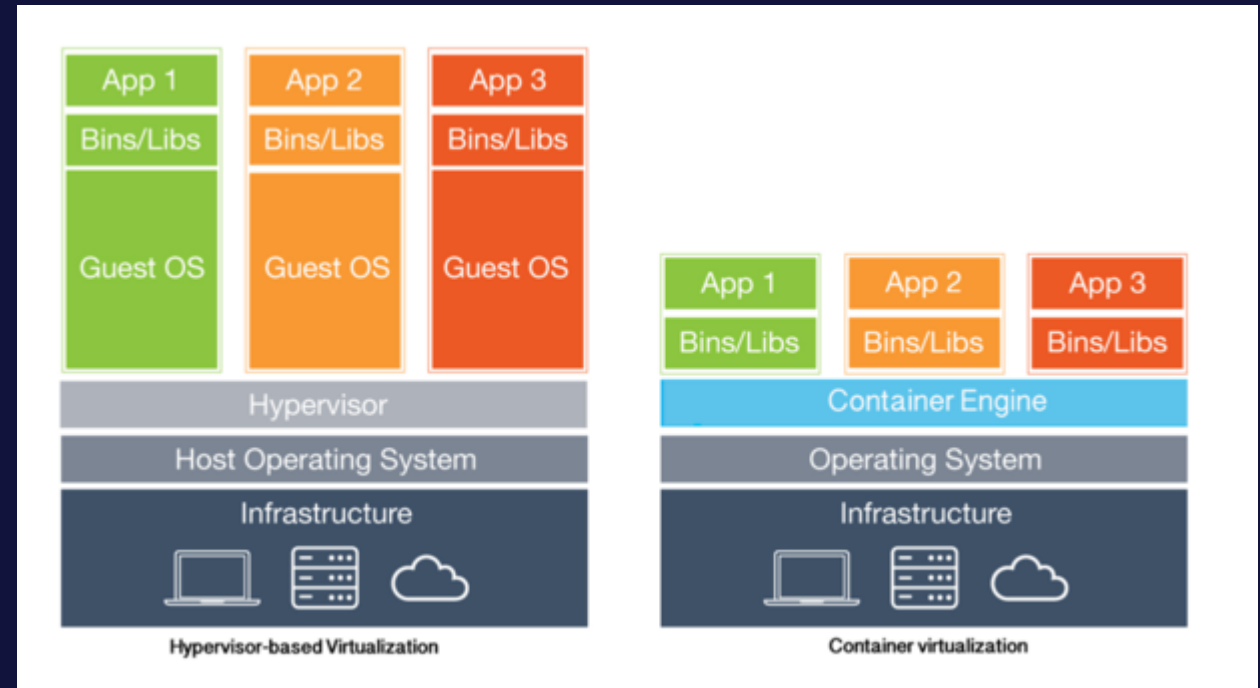
# Hva er Docker?

## Docker er en plattform for applikasjoner.

Docker er en teknologi som gir oss muligheten til å pakke inn programmer i små containere som deler på maskinvare og operativ systemer.

### Fordeler:

- Isolering av applikasjon
- Betydelig bedre sikkerhet
- Dele på maskinvare / nettverk etc.
- Benytte ferdige imager å bygge videre på
- Flyttbart mellom datasystemer.
- Fleksibel for legacy systemer etc.
- Automatisere utrulling av applikasjoner.
- Lagre imager i sentrale lagre.
- Lett å bygge og slette kontainere.
- Kort oppstartstid for containere.
- Mulighet for bruk i kubernetes etc.



# Hvorfor bruke Docker?

Å paketere programmer, tjenester, databaser etc i små Kontainere, gjør at man kan utnytte maskinvaren mye bedre enn ved å ha separate maskiner eller Virtuelle maskiner for hver applikasjon.

Slike kontainere kan dele på samme maskin eller kjøres i et Kluster av maskiner som f.eks under kubernetes.

Da kan man flytte kontainere mellom noder ette behov, eller kjøre flere kontainere på samme maskin, eller spredt utover flere noder for last balansering eller feil håndtering.

En stor fordel er at man kan benytte seg av forhånds lagde Kontainer imager med os, databaser etc, som man selv kan Bygge videre på. Da slipper man å vedlikeholde os og annen Infrastruktur avhengigheter selv.



# Hva er forskjellen mello VM og kontainer?

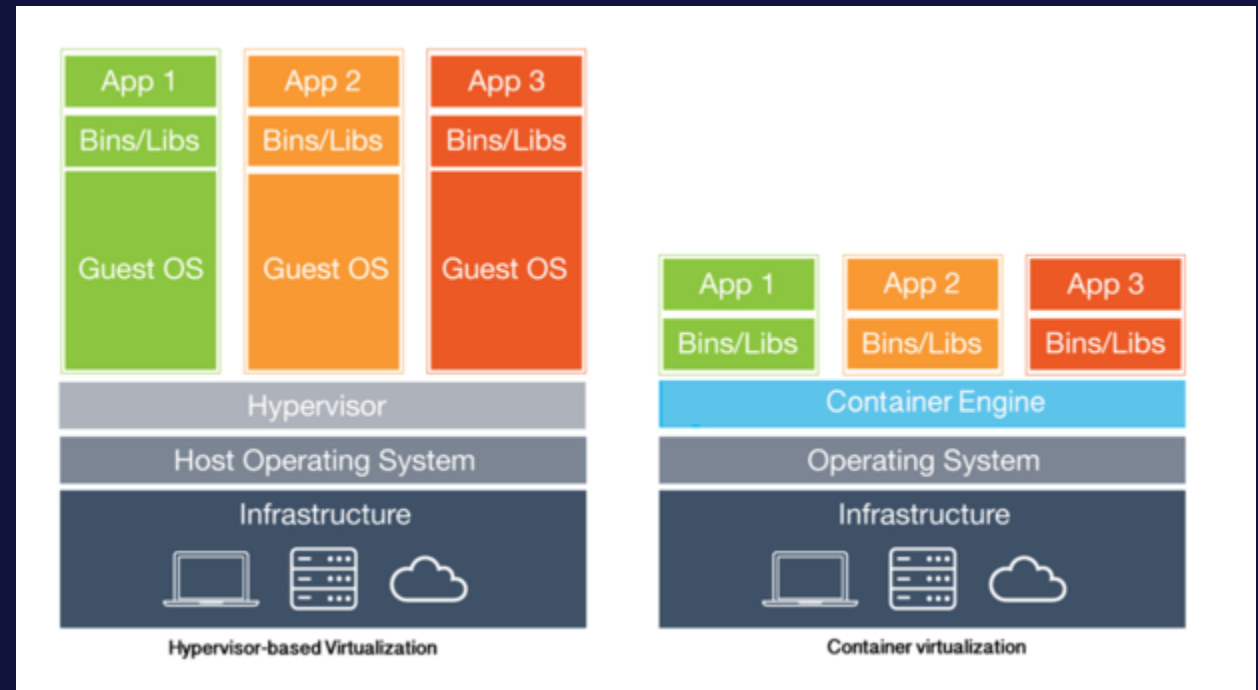
Kontainer(e) deler på maskinvare, Operativsystem og selve kontainer systemet.

Virtuelle maskiner deler på maskinvaren, men Bringer selv operativ systemer, nødvendige Komponenter og applikasjonen.

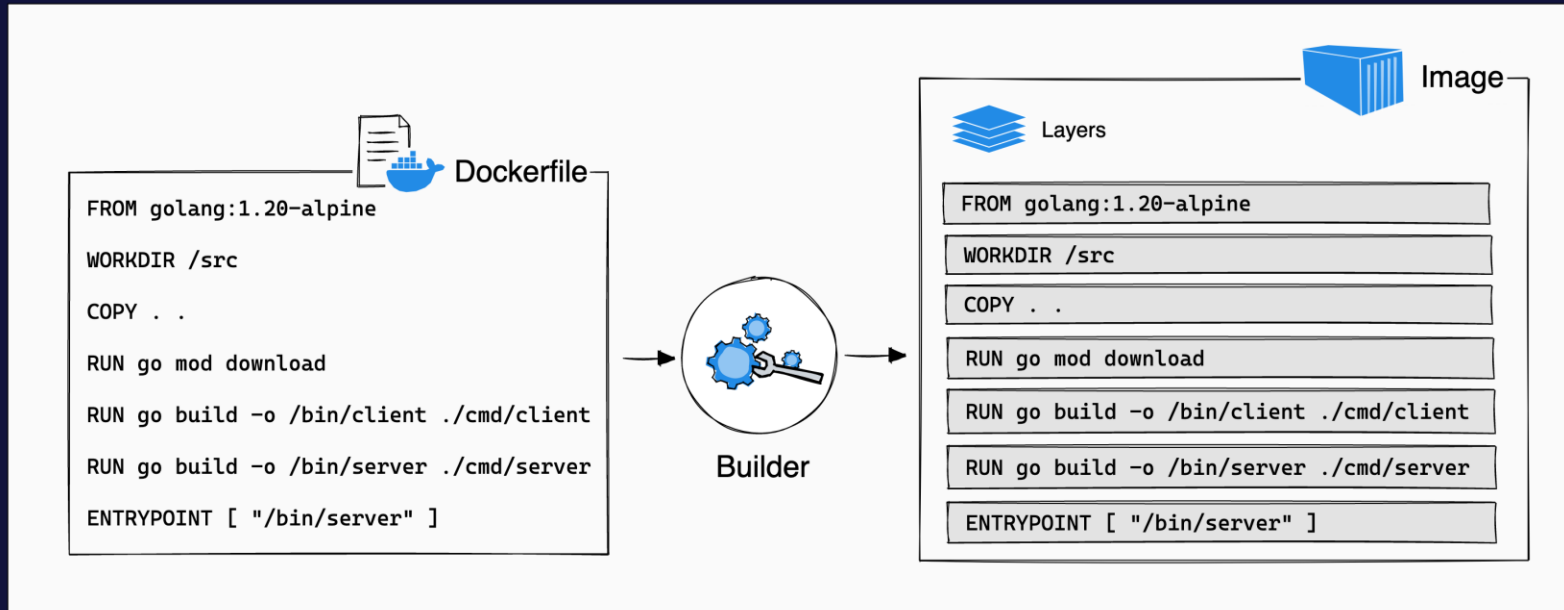
Virtuelle maskiner trenger vesentlig mere Maskinvare som minne og lagring. Krever mye mere administrasjon og vedlikehold og starter tregere.

Kontainere er ofte svært små og bygger på andre imager for nødvendige støtte systemer og deler på operativsystemet og maskinvaren i bunn.

Kontainere er lette og raske å starte og å fjerne. Uten bruk av Volumes lagrer de ikke data etter at de stoppes / slettes.



# Anatomien til en Docker kontainer



Kontainere er lagvis laget. Man starter med en grunn image som f.eks alpine linux, eller en som også inneholder en spesialisering for dot.net e.l.

På denne bygger man lagvis på sine egne avhengigheter og til slutt selve applikasjonen eller tjenesten e.l.

Trenger man lagring utover levetiden til kontaineren, så må man sette opp et volume som lagrer data også etter at kontaineren er stoppet.

# Inninstallere Docker

## På en Linux maskin eller under WSL på Windows

- `sudo apt-get update`
- `sudo apt-get install ca-certificates curl gnupg`
- `sudo install -m 0755 -d /etc/apt/keyrings`
- `curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg`
- `sudo chmod a+r /etc/apt/keyrings/docker.gpg`
- `echo \`
- `"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \`
- `"$(. /etc/os-release && echo "$VERSION_CODENAME)" stable" | \`
- `sudo tee /etc/apt/sources.list.d/docker.list > /dev/null`
- `sudo apt-get update`
- `sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin`
- `sudo docker run hello-world`

# Installere podman / podman desktop

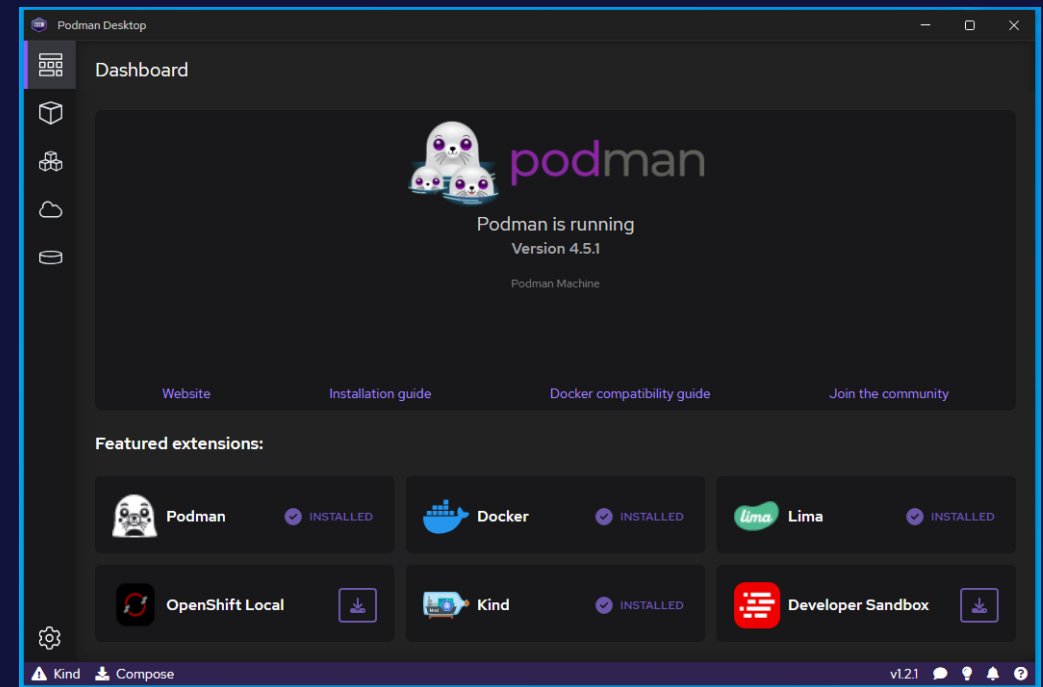
## En opensource alternativ til Docker

Last ned Podman Desktop for Windows fra:

<https://podman-desktop.io/downloads/windows>

For å innstallere under WSL eller på Linux maskiner, kan Man enkelt skrive i wsl / bash:

```
sudo apt -y install podman
```



# Enkel image script

## Et enkelt Python basert eksempel

```
from flask import Flask  
app = Flask(__name__)
```

```
@app.route("/")  
def hello():  
    return "Hello World!"
```

```
FROM python:3-alpine  
RUN python -m pip install --upgrade pip  
RUN pip install flask==2.1.*
```

```
# install app  
COPY hello.py /
```

```
# final configuration  
ENV FLASK_APP=hello  
EXPOSE 8000  
CMD flask run --host 0.0.0.0 --port 8000
```



# Kjøre kontainere basert på egne imager

- `podman build .`
- `podman run --rm -p 127.0.0.1:8000:8000 [hash value from build]`
- Videre kan man pushe imager til repositories lokalt eller f.eks dockerhub for distribusjon til andre servere e.l.
- Åpne nettleser og naviger til: <http://127.0.0.1:8000/>

# Oppgave: Lag en enkel container

- Start med å installere docker / podman enten lokalt eller på sky tjeneste.
- Forsikre deg om at det virker ved å kjøre 'hello-world' container.
- Lag det lille Python scriptet og benytt docker script til å bygge din egen container.
- Start opp kontaineren. Sjekk at den kjører, eks: 'docker container ls'
- Prøv å gå til <http://127.0.0.1:8000/>