WARWICK

Introduction to Docker

James Tripp, Senior Research Software Engineer

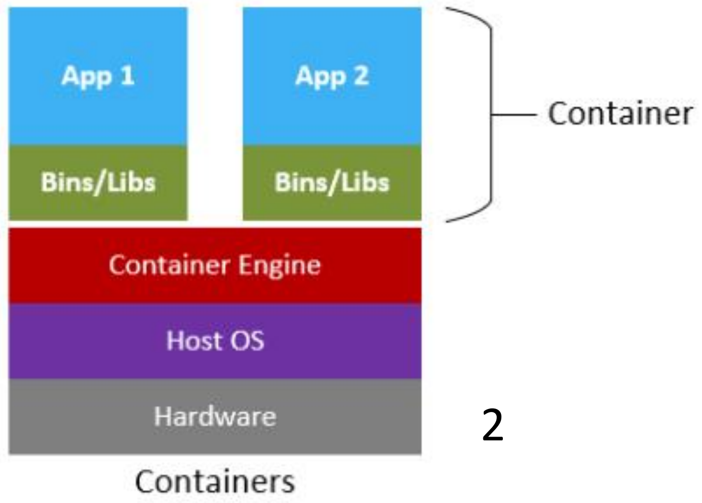
18/11/22

Plan

- Introduction to Containers
- What is Docker?
- Using Docker
- Going Further

Introduction to containers

“Containers are packages of software that contain all of the necessary elements to run in any environment. In this way, containers virtualize the operating system and run anywhere” [1]



2

An application and dependencies in an easy to transport bundle.

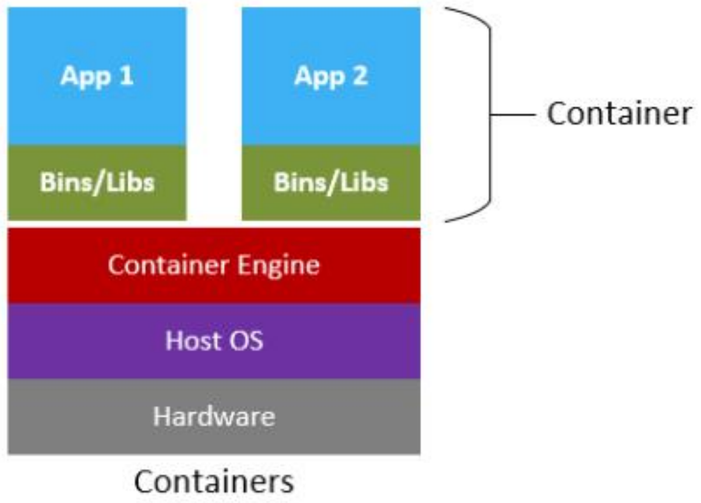
[1]What are containers? | Google Cloud

[2]Demystifying containers, Docker, and Kubernetes - Microsoft Open Source Blog

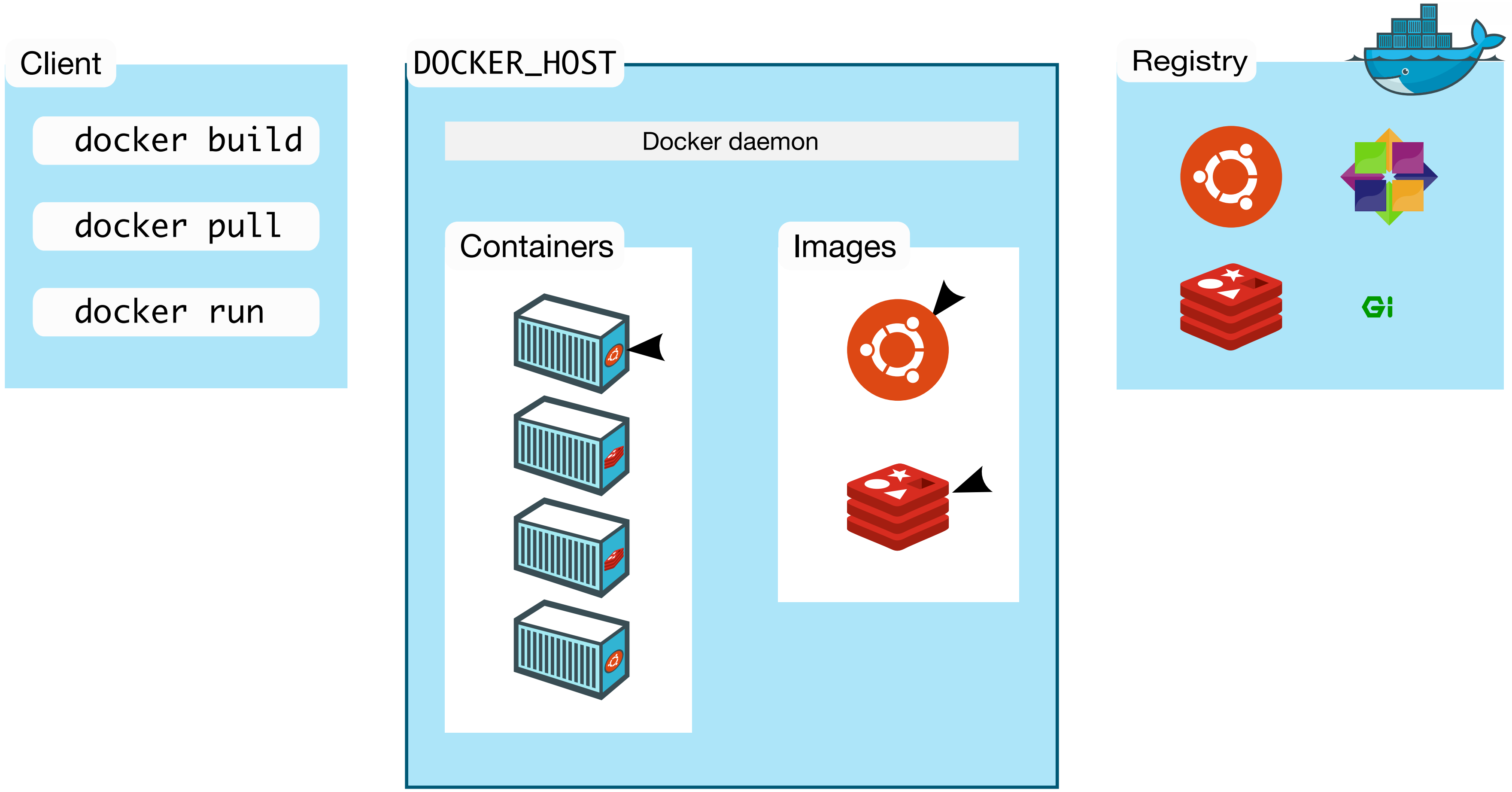
Introduction to containers

Different container engines:

- Singularity/apptainer (often used in HPC)
- Podman (developed by redhat)
- Docker (commonly used)



What is Docker?



1

[1]Docker overview | Docker Documentation

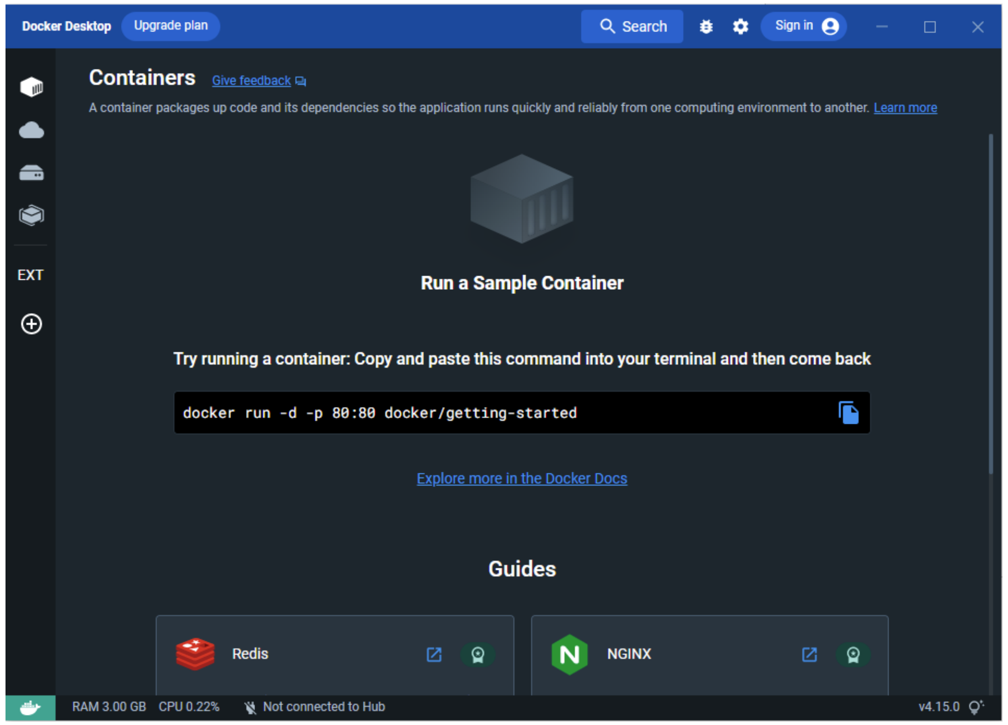
What is Docker?

WARWICK

- Windows or MacOS users can install Docker Desktop from
  - Docker Engine
  - And a pretty ui
  - https://www.docker.com/products/docker-desktop/

- Linux users can install the docker engine
  - Same core functionality
  - https://docs.docker.com/engine/install/

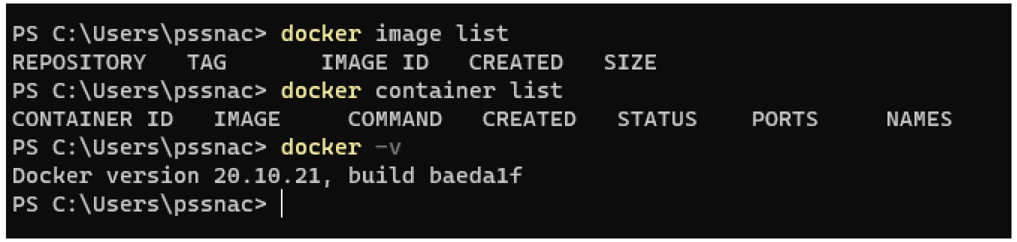
- The docker engine is accessed via the terminal



A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another.

Try running a container: Copy and paste this command into your terminal and then come back

```
docker run -d -p 80:80 docker/getting-started
```

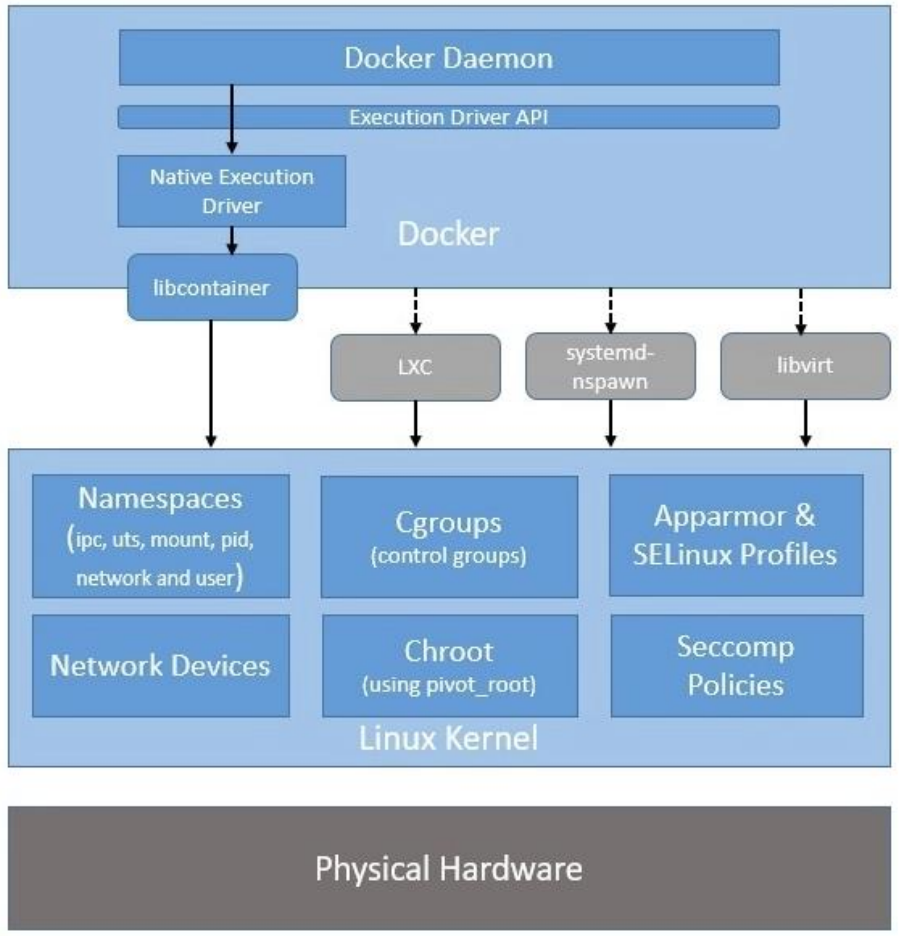


```
PS C:\Users\pssnac> docker image list
REPOSITORY    TAG        IMAGE ID    CREATED    SIZE
PS C:\Users\pssnac> docker container list
CONTAINER ID   IMAGE     COMMAND    CREATED    STATUS     PORTS      NAMES
PS C:\Users\pssnac> docker -v
Docker version 20.10.21, build baeda1f
PS C:\Users\pssnac> |
```

What is Docker?

A small aside

- Technologies such as chroot, namespaces and cgroups keep the processes and filesystems separate.
- In other words, “Docker is simulating various Linux distributions, environments or installs instead of running them.”[2]

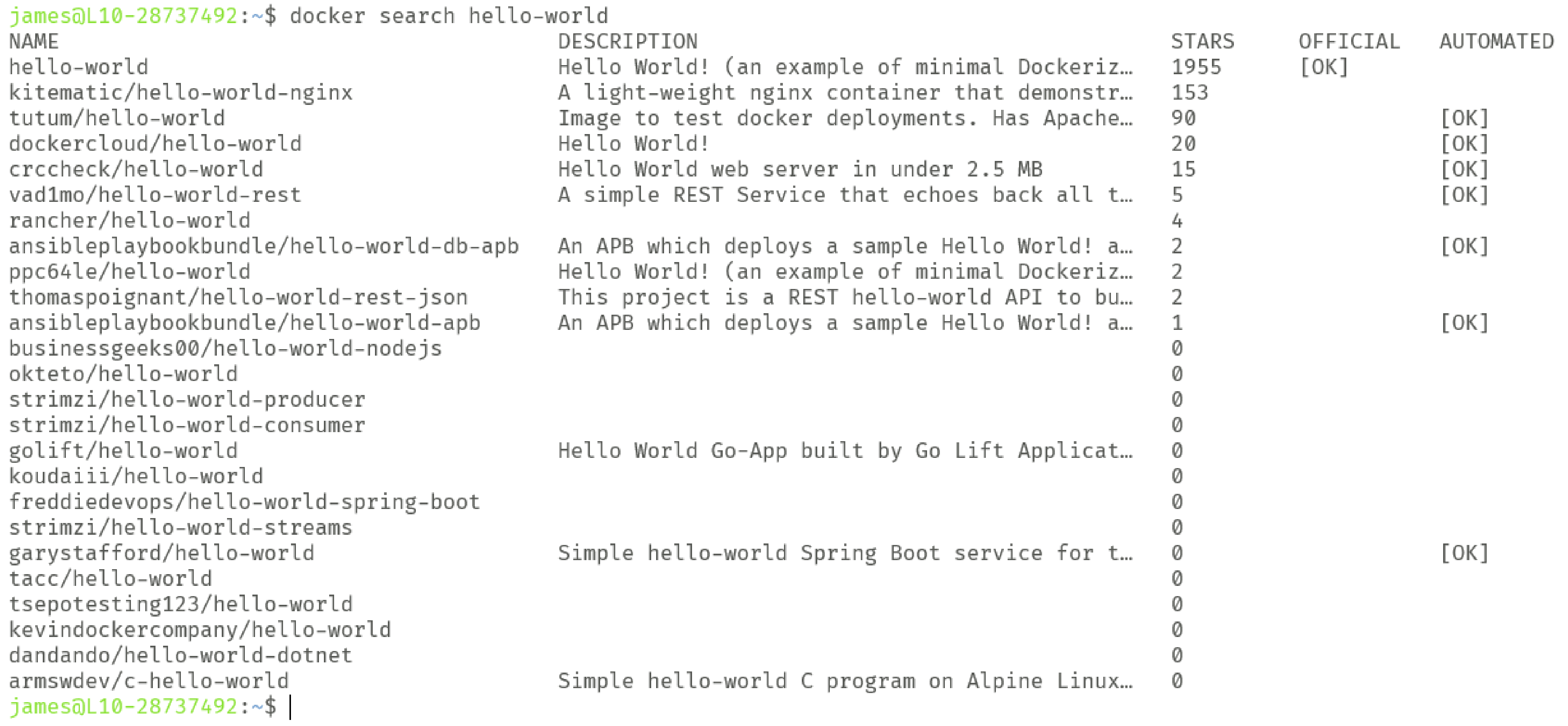


1

[1]https://www.baeldung.com/linux/docker-containers-evolution

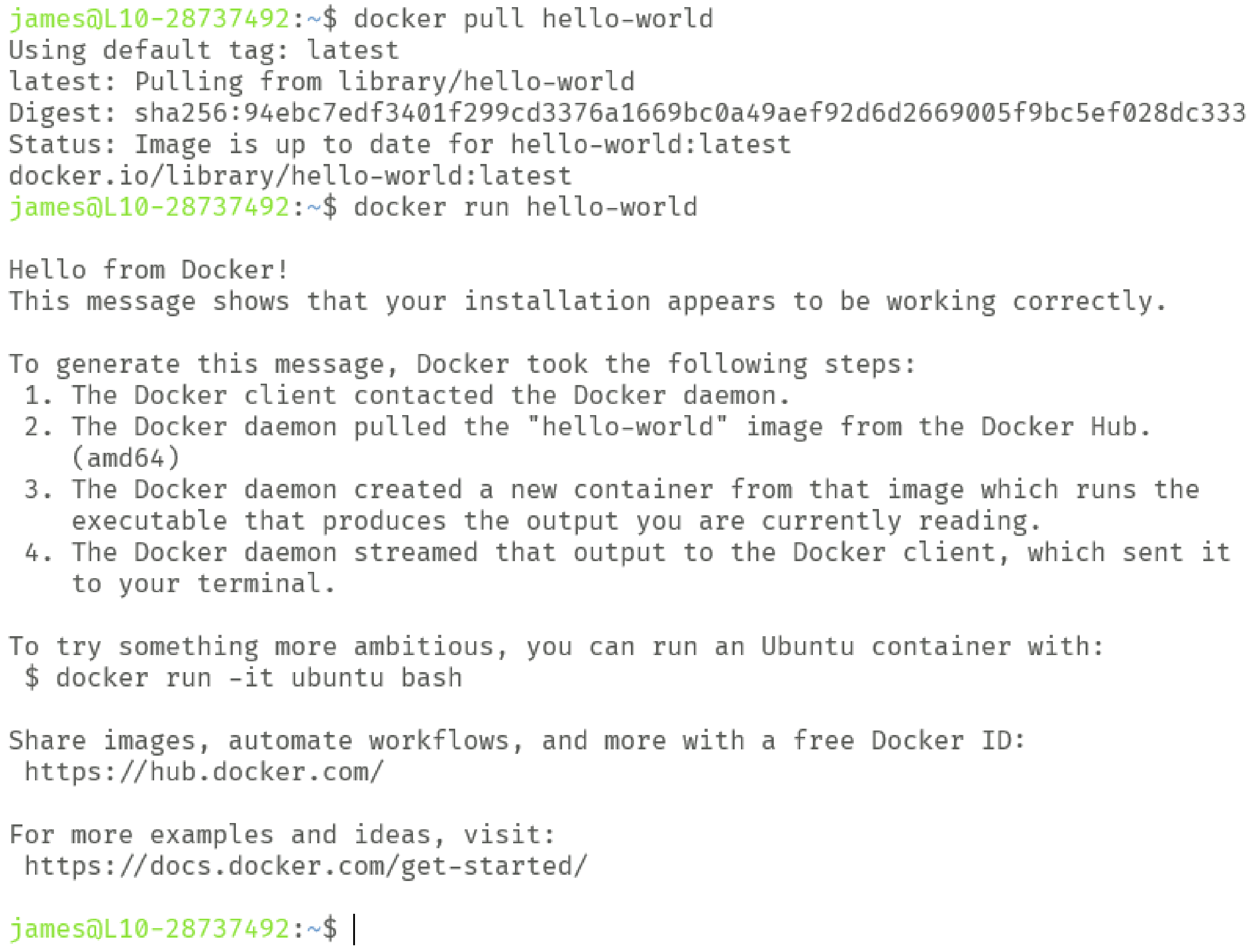
[2]https://www.codementor.io/blog/docker-technology-5x1kilcbow

Using Docker – hello world



```
james@L10-28737492:~$ docker search hello-world
NAME                                       DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
hello-world                                Hello World! (an example of minimal Dockeriz…   1955      [OK]
kitematic/hello-world-nginx                A light-weight nginx container that demonstr…   153
tutum/hello-world                          Image to test docker deployments. Has Apache…   90                   [OK]
dockercloud/hello-world                    Hello World!                                    20                   [OK]
crccheck/hello-world                       Hello World web server in under 2.5 MB          15                   [OK]
vad1mo/hello-world-rest                    A simple REST Service that echoes back all t…   5                    [OK]
rancher/hello-world                                                                        4
ansibleplaybookbundle/hello-world-db-apb   An APB which deploys a sample Hello World! a…   2                    [OK]
ppc64le/hello-world                        Hello World! (an example of minimal Dockeriz…   2
thomaspoignant/hello-world-rest-json       This project is a REST hello-world API to bu…   2
ansibleplaybookbundle/hello-world-apb      An APB which deploys a sample Hello World! a…   1                    [OK]
businessgeeks00/hello-world-nodejs                                                         0
okteto/hello-world                                                                         0
strimzi/hello-world-producer                                                               0
strimzi/hello-world-consumer                                                               0
golift/hello-world                         Hello World Go-App built by Go Lift Applicat…   0
koudaiii/hello-world                                                                       0
freddiedevops/hello-world-spring-boot                                                      0
strimzi/hello-world-streams                                                                0
garystafford/hello-world                   Simple hello-world Spring Boot service for t…   0                    [OK]
tacc/hello-world                                                                           0
tsepotesting123/hello-world                                                                0
kevindockercompany/hello-world                                                             0
dandando/hello-world-dotnet                                                                0
armswdev/c-hello-world                     Simple hello-world C program on Alpine Linux…   0
james@L10-28737492:~$
```

Using Docker – hello world



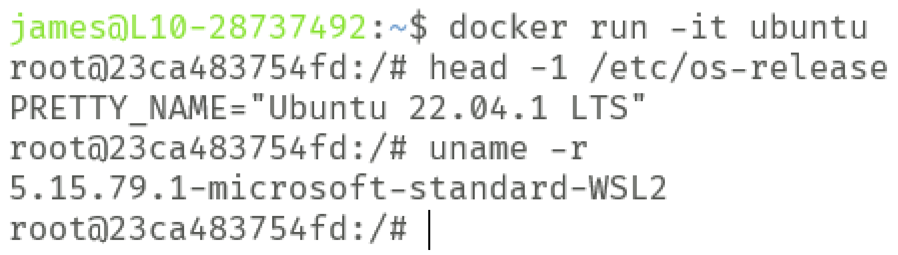
```
james@L10-28737492:~$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
Digest: sha256:94ebc7edf3401f299cd3376a1669bc0a49aef92d6d2669005f9bc5ef028dc333
Status: Image is up to date for hello-world:latest
docker.io/library/hello-world:latest
james@L10-28737492:~$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

james@L10-28737492:~$ |
```

Download the hello world image

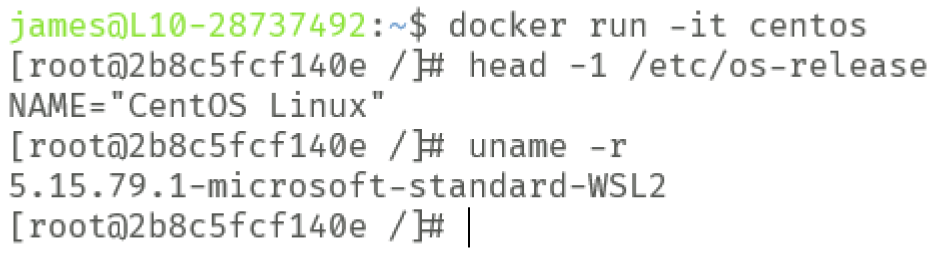
Create and run a container using that image

Using Docker - Distributions

WARWICK



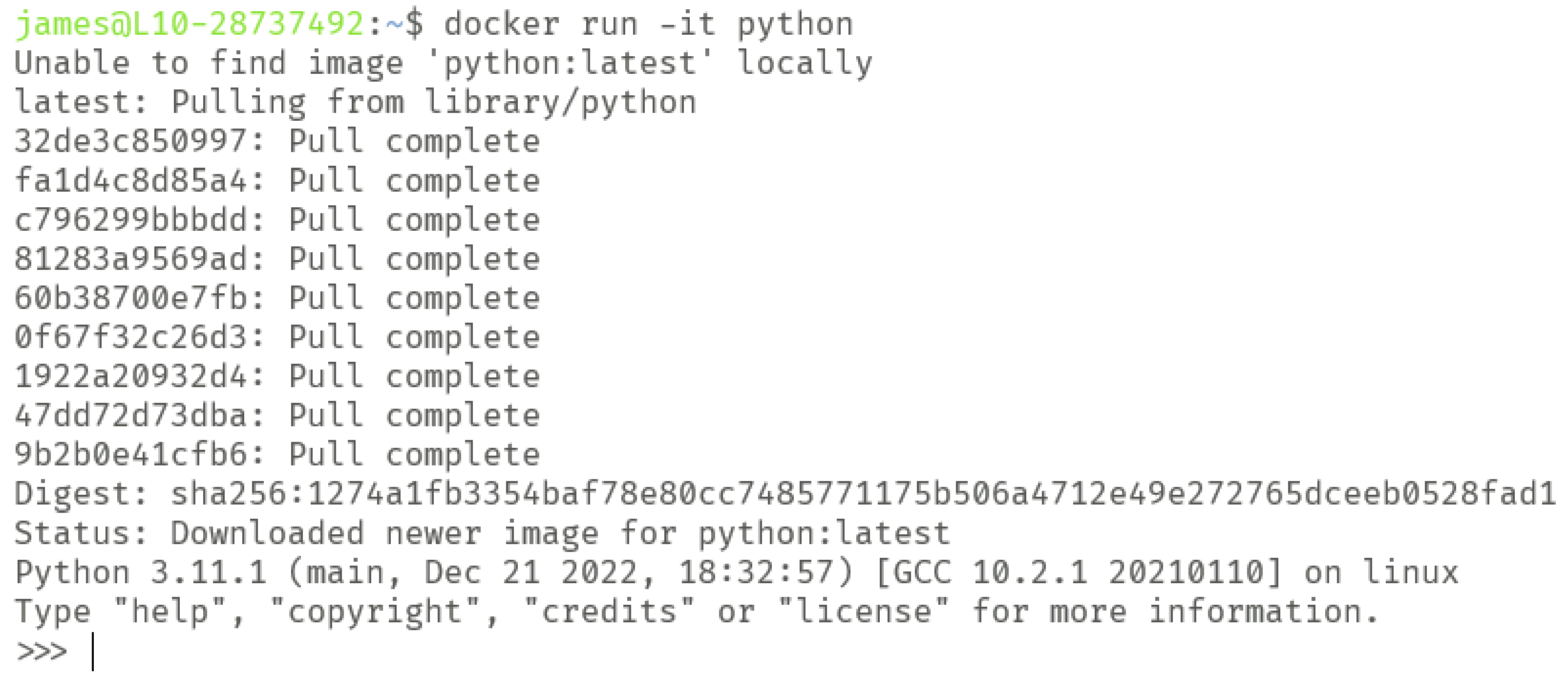
```
james@L10-28737492:~$ docker run -it ubuntu
root@23ca483754fd:/# head -1 /etc/os-release
PRETTY_NAME="Ubuntu 22.04.1 LTS"
root@23ca483754fd:/# uname -r
5.15.79.1-microsoft-standard-WSL2
root@23ca483754fd:/#
```



```
james@L10-28737492:~$ docker run -it centos
[root@2b8c5fcf140e /]# head -1 /etc/os-release
NAME="CentOS Linux"
[root@2b8c5fcf140e /]# uname -r
5.15.79.1-microsoft-standard-WSL2
[root@2b8c5fcf140e /]#
```

- There are images for many Linux distributions
- The file systems are different, but the kernel is the same
- Worth noting if your code relies on a different/recompiled kernel

Using Docker - Applications



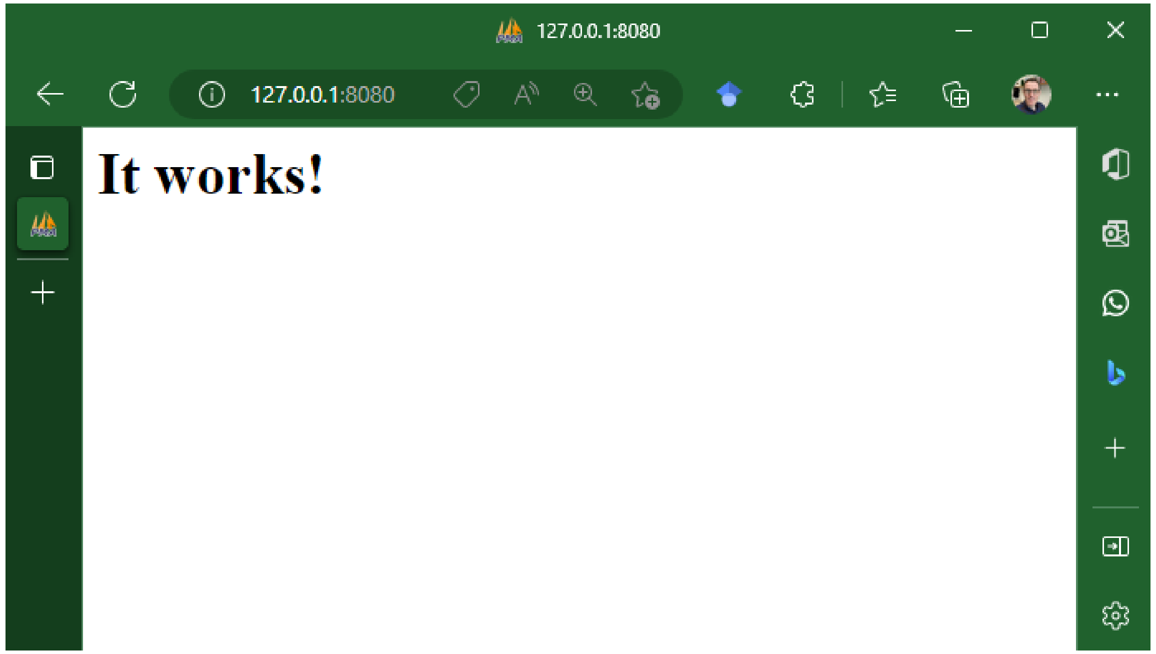
```
james@L10-28737492:~$ docker run -it python
Unable to find image 'python:latest' locally
latest: Pulling from library/python
32de3c850997: Pull complete
fa1d4c8d85a4: Pull complete
c796299bbbdd: Pull complete
81283a9569ad: Pull complete
60b38700e7fb: Pull complete
0f67f32c26d3: Pull complete
1922a20932d4: Pull complete
47dd72d73dba: Pull complete
9b2b0e41cfb6: Pull complete
Digest: sha256:1274a1fb3354baf78e80cc7485771175b506a4712e49e272765dceeb0528fad1
Status: Downloaded newer image for python:latest
Python 3.11.1 (main, Dec 21 2022, 18:32:57) [GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> |
```

- Made up of several ‘layers’
- Image Layer Details - python:latest | Docker Hub

Using Docker – Apache

```
james@L10-28737492:~$ docker run -dit -p 8080:80 httpd:2.4
ecc5fb5abcacf7b1cc6c4b6751b681a2dd90d5cd2a97cb23d877c7c9b3b52f79
james@L10-28737492:~$ |
```

- Docker has network capabilities
- Httpd container in detached mode with an interactive terminal (tty)
- The : specifies the tag. 2.4 corresponds to httpd v2.4



It works!

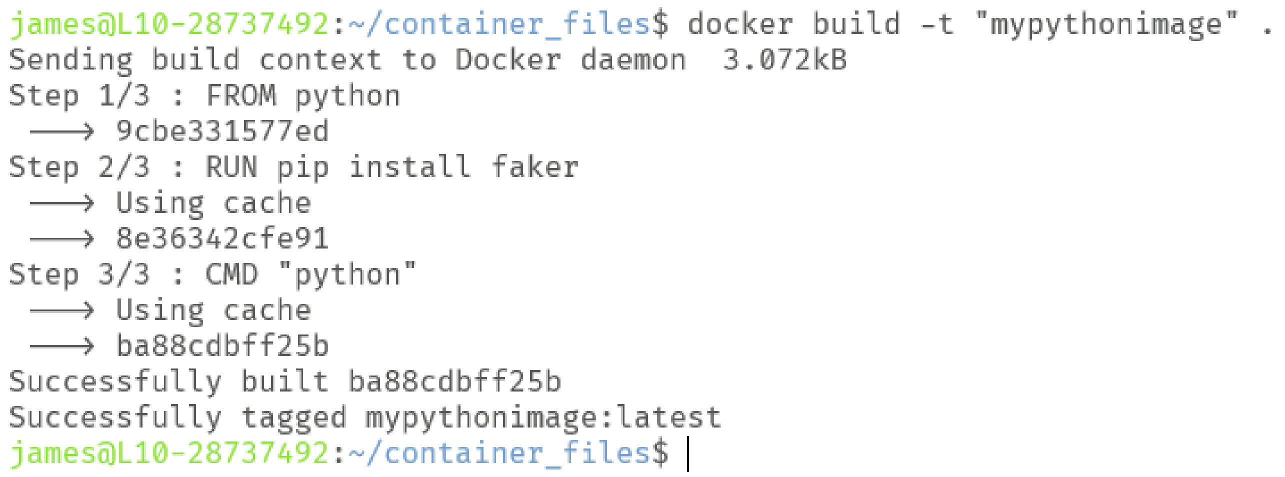
Using Docker – Making images

WARWICK

- Dockerfiles tell docker how to build an image.

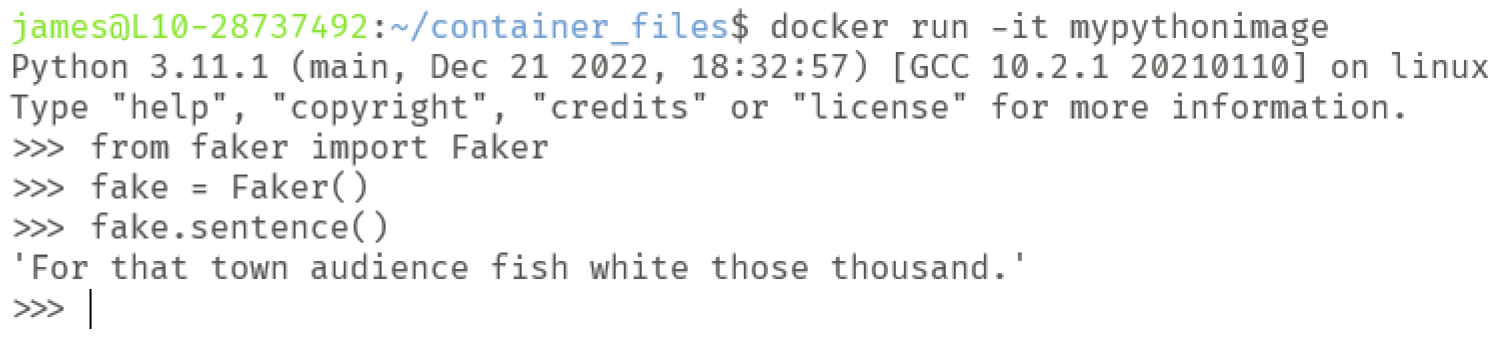
```
FROM python               # builds on the official python image
RUN pip install faker     # installs the faker python module
CMD "python“              # run the python command
```

- Name the file Dockerfile and run in the folder



```
james@L10-28737492:~/container_files$ docker build -t "mypythonimage" .
Sending build context to Docker daemon  3.072kB
Step 1/3 : FROM python
 ---> 9cbe331577ed
Step 2/3 : RUN pip install faker
 ---> Using cache
 ---> 8e36342cfe91
Step 3/3 : CMD "python"
 ---> Using cache
 ---> ba88cdbff25b
Successfully built ba88cdbff25b
Successfully tagged mypythonimage:latest
james@L10-28737492:~/container_files$ |
```

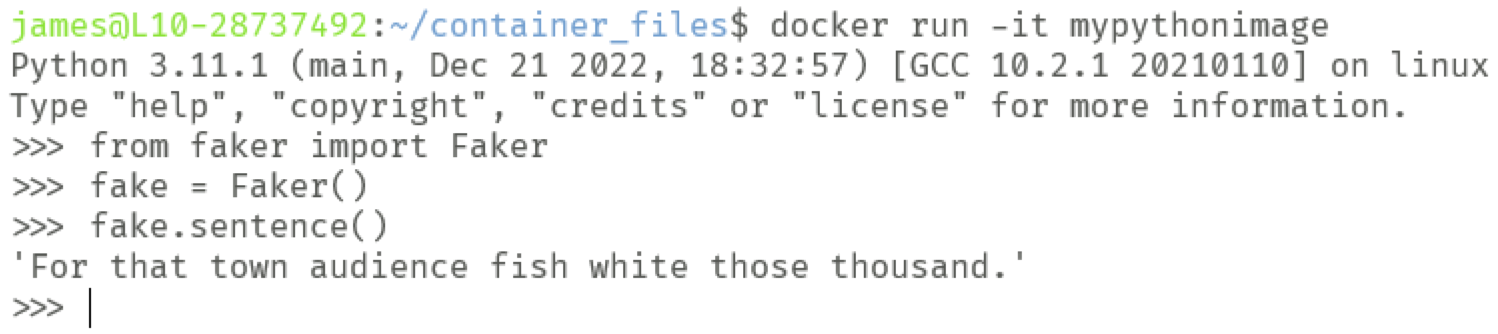
Using Docker – Making images



```
james@L10-28737492:~/container_files$ docker run -it mypythonimage
Python 3.11.1 (main, Dec 21 2022, 18:32:57) [GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from faker import Faker
>>> fake = Faker()
>>> fake.sentence()
'For that town audience fish white those thousand.'
>>> 
```

- Note: containers are destroyed once the command in cmd ends, all files in the container are destroyed!

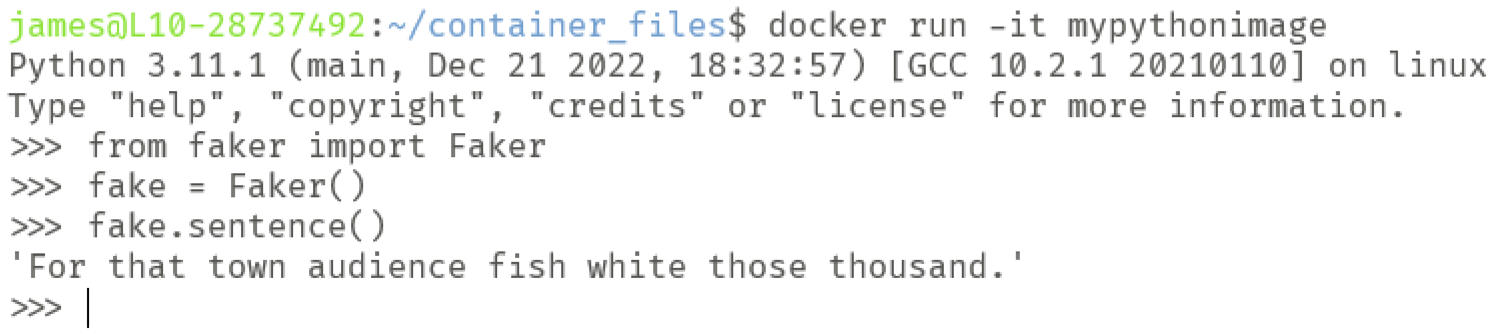
Using Docker – Making images



```
james@L10-28737492:~/container_files$ docker run -it mypythonimage
Python 3.11.1 (main, Dec 21 2022, 18:32:57) [GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from faker import Faker
>>> fake = Faker()
>>> fake.sentence()
'For that town audience fish white those thousand.'
>>> |
```

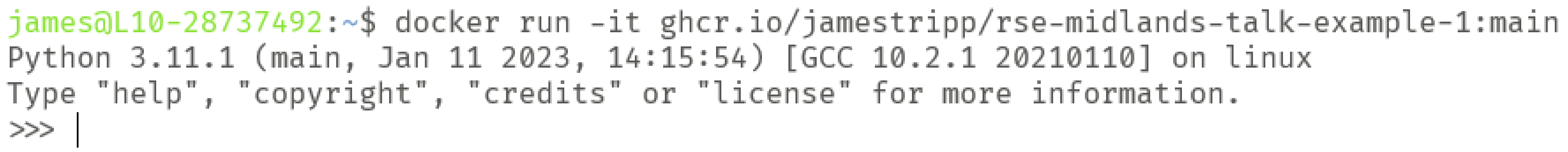
- Note: containers are destroyed once the command in cmd ends, all files in the container are destroyed!

Using Docker – Making images



```
james@L10-28737492:~/container_files$ docker run -it mypythonimage
Python 3.11.1 (main, Dec 21 2022, 18:32:57) [GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from faker import Faker
>>> fake = Faker()
>>> fake.sentence()
'For that town audience fish white those thousand.'
>>> |
```

- Note: containers are destroyed once the command in cmd ends, all files in the container are destroyed!
- Also available via github via github actions. Code at [jamestripp/rse-midlands-talk-example-2]()



```
james@L10-28737492:~$ docker run -it ghcr.io/jamestripp/rse-midlands-talk-example-1:main
Python 3.11.1 (main, Jan 11 2023, 14:15:54) [GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> |
```

Using Docker – Accessing files

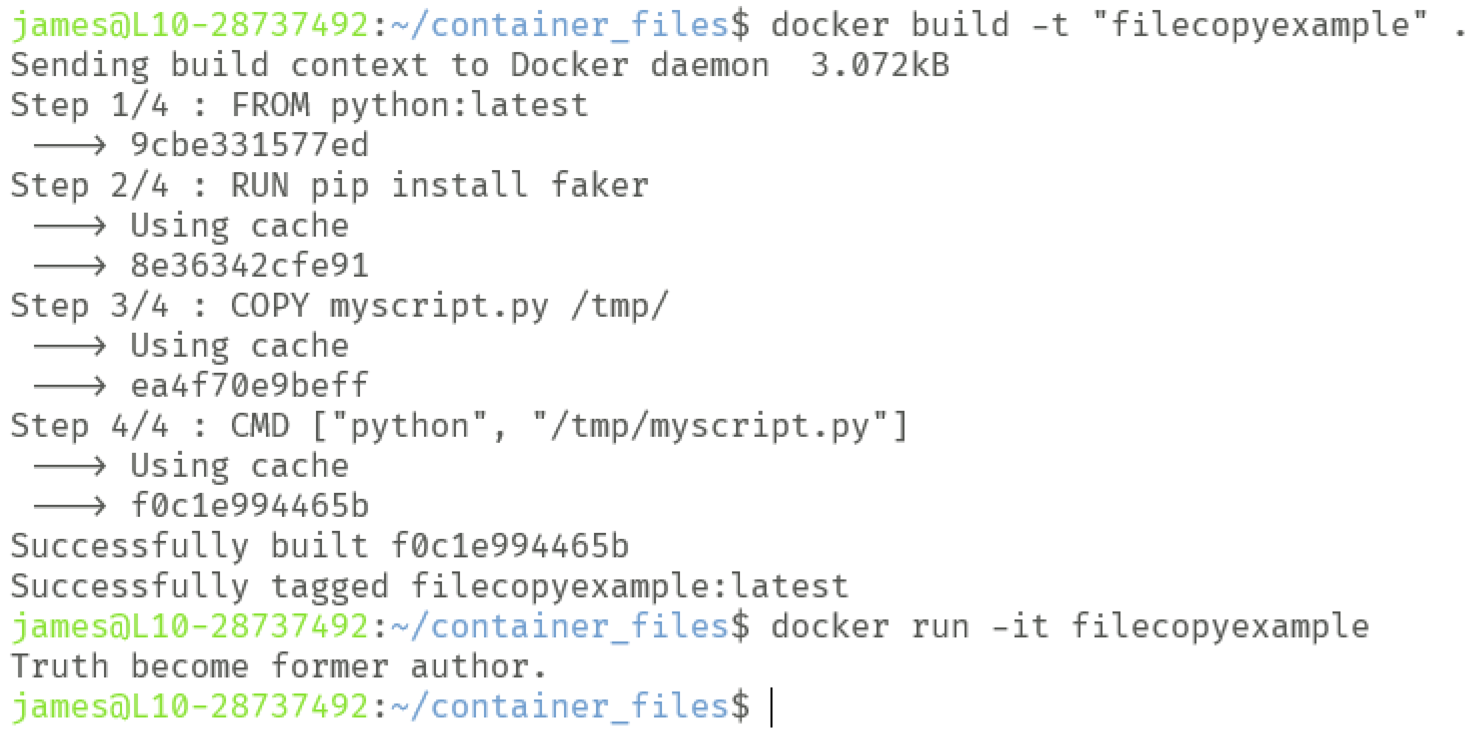
Include it in the image

myscript.py

```python
#!/bin/python
from faker import Faker

fake = Faker()

print(fake.sentence())
```

Dockerfile

```dockerfile
FROM python:latest

RUN pip install faker

COPY myscript.py /tmp/

CMD ["python", "/tmp/myscript.py"]
```



```
james@L10-28737492:~/container_files$ docker build -t "filecopyexample" .
Sending build context to Docker daemon  3.072kB
Step 1/4 : FROM python:latest
 ⟶ 9cbe331577ed
Step 2/4 : RUN pip install faker
 ⟶ Using cache
 ⟶ 8e36342cfe91
Step 3/4 : COPY myscript.py /tmp/
 ⟶ Using cache
 ⟶ ea4f70e9beff
Step 4/4 : CMD ["python", "/tmp/myscript.py"]
 ⟶ Using cache
 ⟶ f0c1e994465b
Successfully built f0c1e994465b
Successfully tagged filecopyexample:latest
james@L10-28737492:~/container_files$ docker run -it filecopyexample
Truth become former author.
james@L10-28737492:~/container_files$ |
```

Using Docker – Accessing files

WARWICK

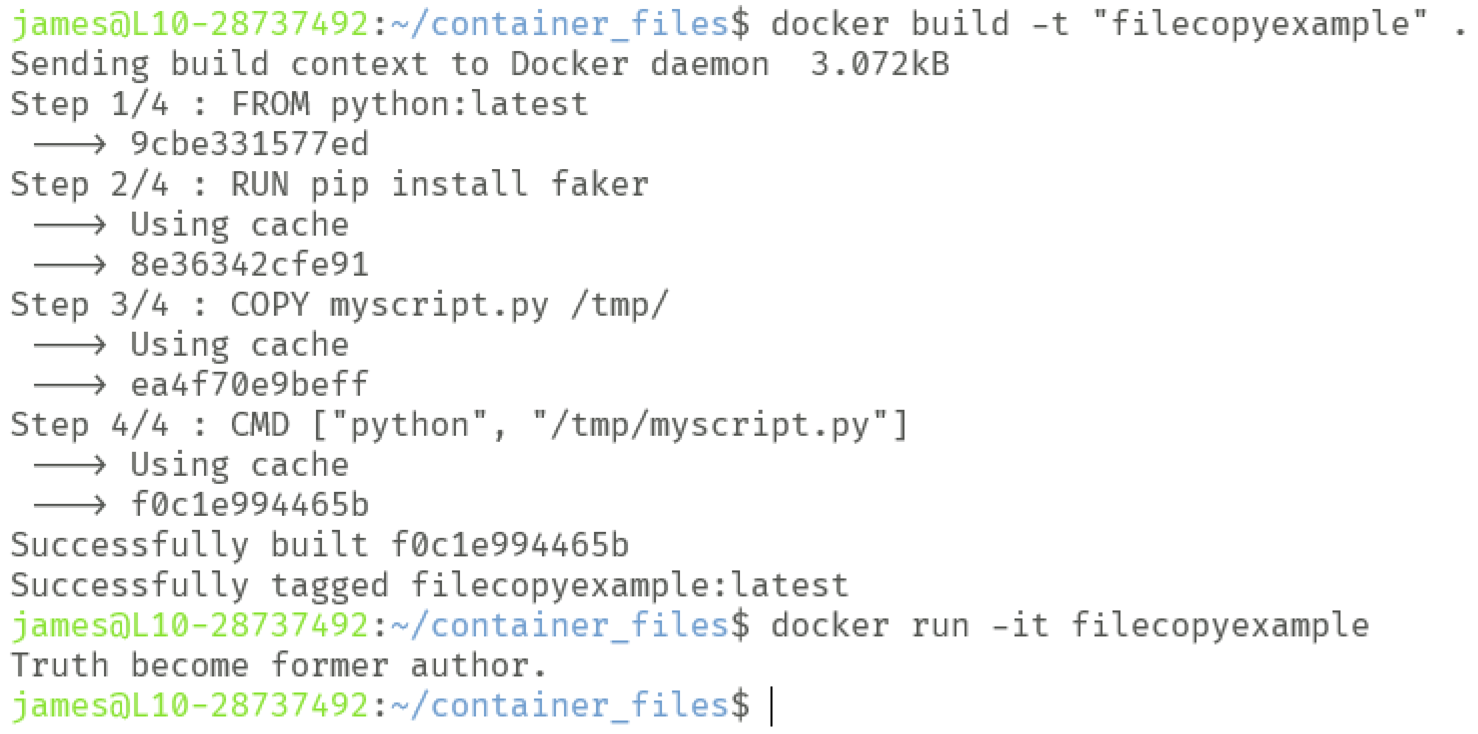
Include it in the image

myscript.py

```python
#!/bin/python
from faker import Faker

fake = Faker()

print(fake.sentence())
```

Dockerfile

```dockerfile
FROM python:latest

RUN pip install faker

COPY myscript.py /tmp/

CMD ["python", "/tmp/myscript.py"]
```



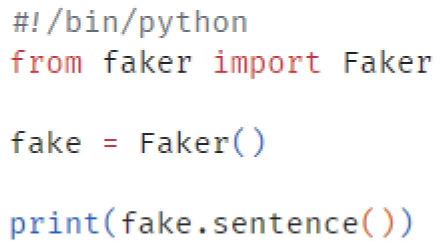
```
james@L10-28737492:~/container_files$ docker build -t "filecopyexample" .
Sending build context to Docker daemon  3.072kB
Step 1/4 : FROM python:latest
 ⟶ 9cbe331577ed
Step 2/4 : RUN pip install faker
 ⟶ Using cache
 ⟶ 8e36342cfe91
Step 3/4 : COPY myscript.py /tmp/
 ⟶ Using cache
 ⟶ ea4f70e9beff
Step 4/4 : CMD ["python", "/tmp/myscript.py"]
 ⟶ Using cache
 ⟶ f0c1e994465b
Successfully built f0c1e994465b
Successfully tagged filecopyexample:latest
james@L10-28737492:~/container_files$ docker run -it filecopyexample
Truth become former author.
james@L10-28737492:~/container_files$ 
```

Using Docker – Accessing files

WARWICK

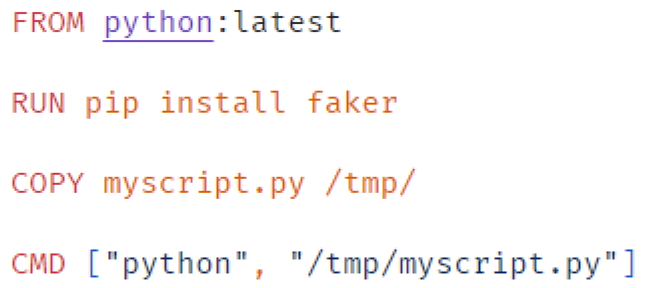
Include it in the image

myscript.py

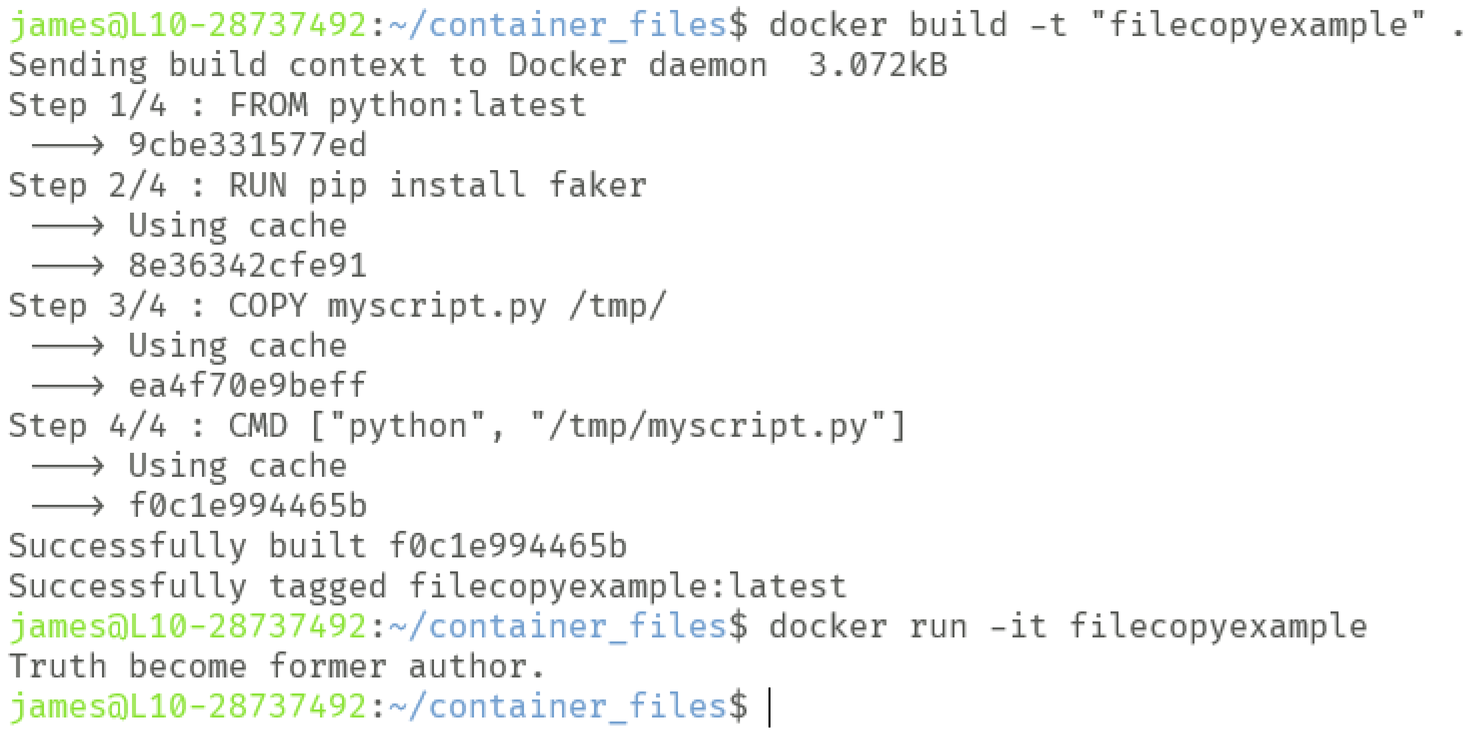


```python
#!/bin/python
from faker import Faker

fake = Faker()

print(fake.sentence())
```

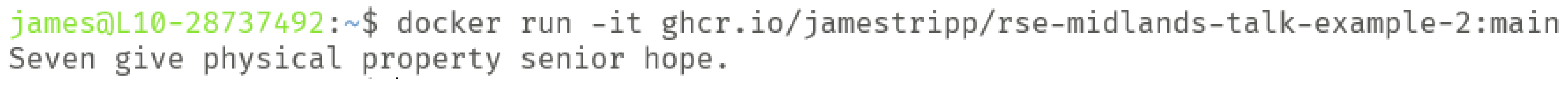
Dockerfile



```dockerfile
FROM python:latest

RUN pip install faker

COPY myscript.py /tmp/

CMD ["python", "/tmp/myscript.py"]
```



```
james@L10-28737492:~/container_files$ docker build -t "filecopyexample" .
Sending build context to Docker daemon  3.072kB
Step 1/4 : FROM python:latest
 ⟶ 9cbe331577ed
Step 2/4 : RUN pip install faker
 ⟶ Using cache
 ⟶ 8e36342cfe91
Step 3/4 : COPY myscript.py /tmp/
 ⟶ Using cache
 ⟶ ea4f70e9beff
Step 4/4 : CMD ["python", "/tmp/myscript.py"]
 ⟶ Using cache
 ⟶ f0c1e994465b
Successfully built f0c1e994465b
Successfully tagged filecopyexample:latest
james@L10-28737492:~/container_files$ docker run -it filecopyexample
Truth become former author.
james@L10-28737492:~/container_files$ |
```



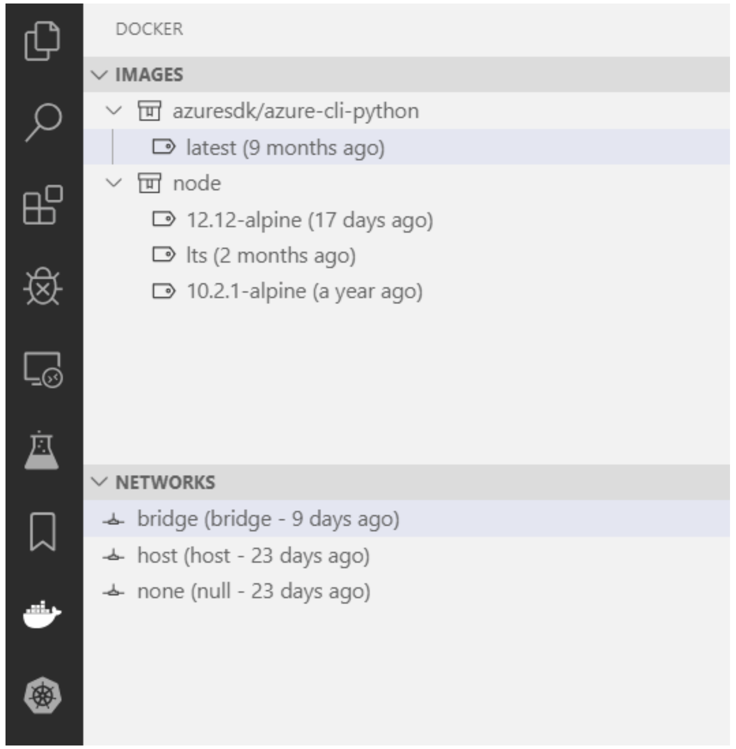
```
james@L10-28737492:~$ docker run -it ghcr.io/jamestripp/rse-midlands-talk-example-2:main
Seven give physical property senior hope.
```

Using Docker – Accessing files

- Also
  - mount a local folder as a volume (Volumes | Docker Documentation)
  - copy the file over from the command line (docker cp | Docker Documentation)
- There is lots of documentation out there

Next steps: VSCode

- VSCode has excellent Docker support (Docker extension for Visual Studio Code)
- Autocomplete is excellent
- GUI support for building images, etc.


DOCKER

IMAGES

- azuresdk/azure-cli-python
  - latest (9 months ago)
- node
  - 12.12-alpine (17 days ago)
  - lts (2 months ago)
  - 10.2.1-alpine (a year ago)

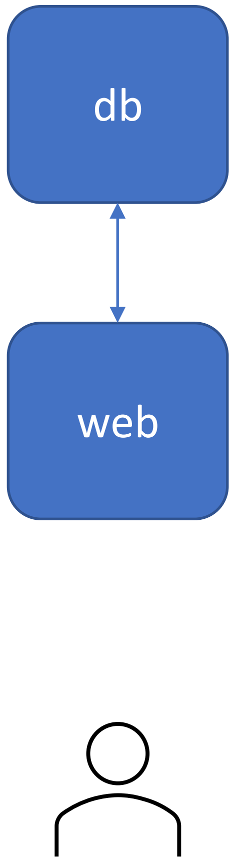
NETWORKS

- bridge (bridge - 9 days ago)
- host (host - 23 days ago)
- none (null - 23 days ago)

Next Steps: Docker Compose

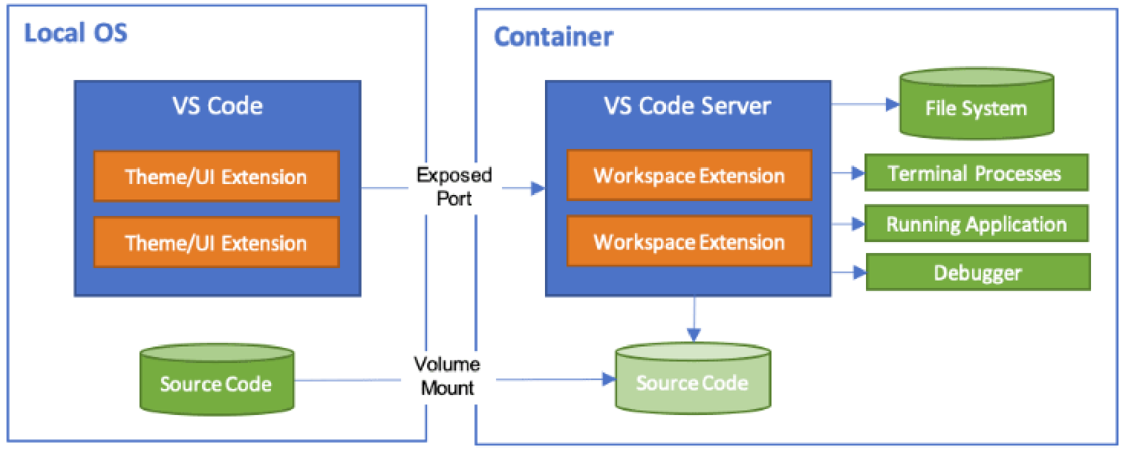
- Linking a group of containers together within a network (Django example)

```
services:
  db:
    image: postgres
    volumes:
      - ./data/db:/var/lib/postgresql/data
    environment:
      - POSTGRES_DB=postgres
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=postgres
  web:
    build: .
    command: python manage.py runserver 0.0.0.0:8000
    volumes:
      - .:/code
    ports:
      - "8000:8000"
    environment:
      - POSTGRES_NAME=postgres
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=postgres
    depends_on:
      - db
```



Next steps: DevContainers

- Install VSCode, Docker and the Dev Containers extension
- Select create dev container and the programming language you want to develop in
- Container is created and VScode has access to it
- (demo if time)



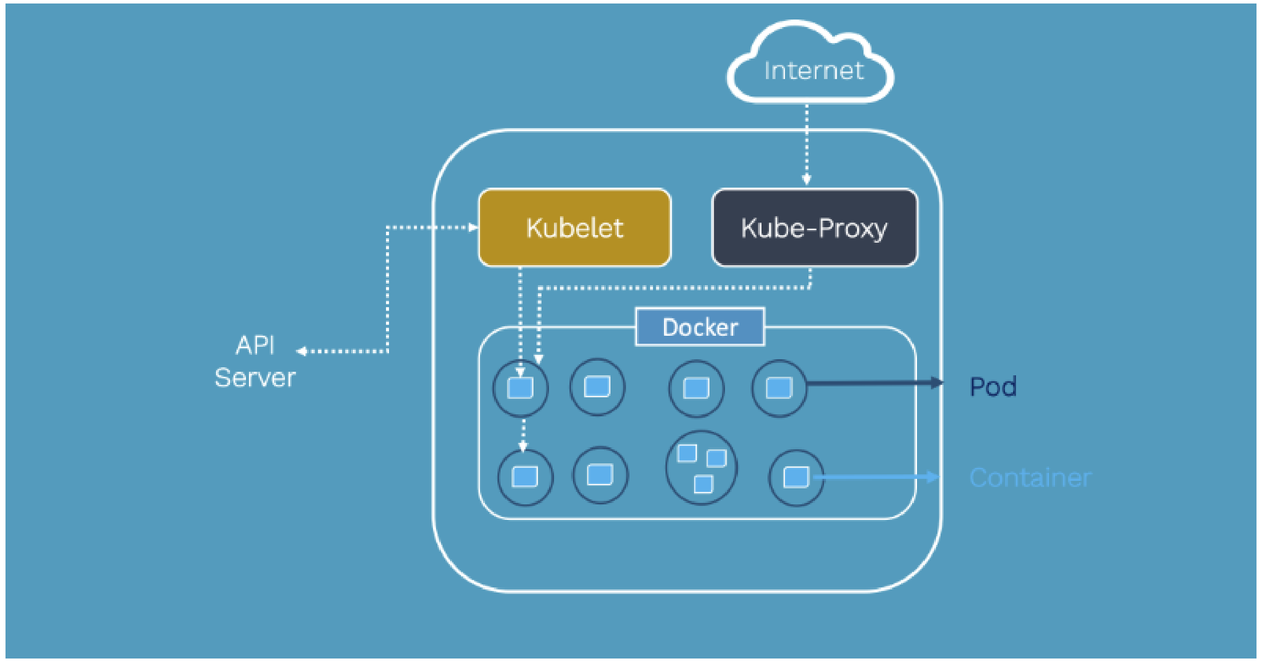
[1]

[1] Developing inside a Container using Visual Studio Code Remote Development

Next Steps: Kubernetes

- Can we deploy groups of one or more containers? In Kubernetes these are called pods

- Kubernetes vs Docker

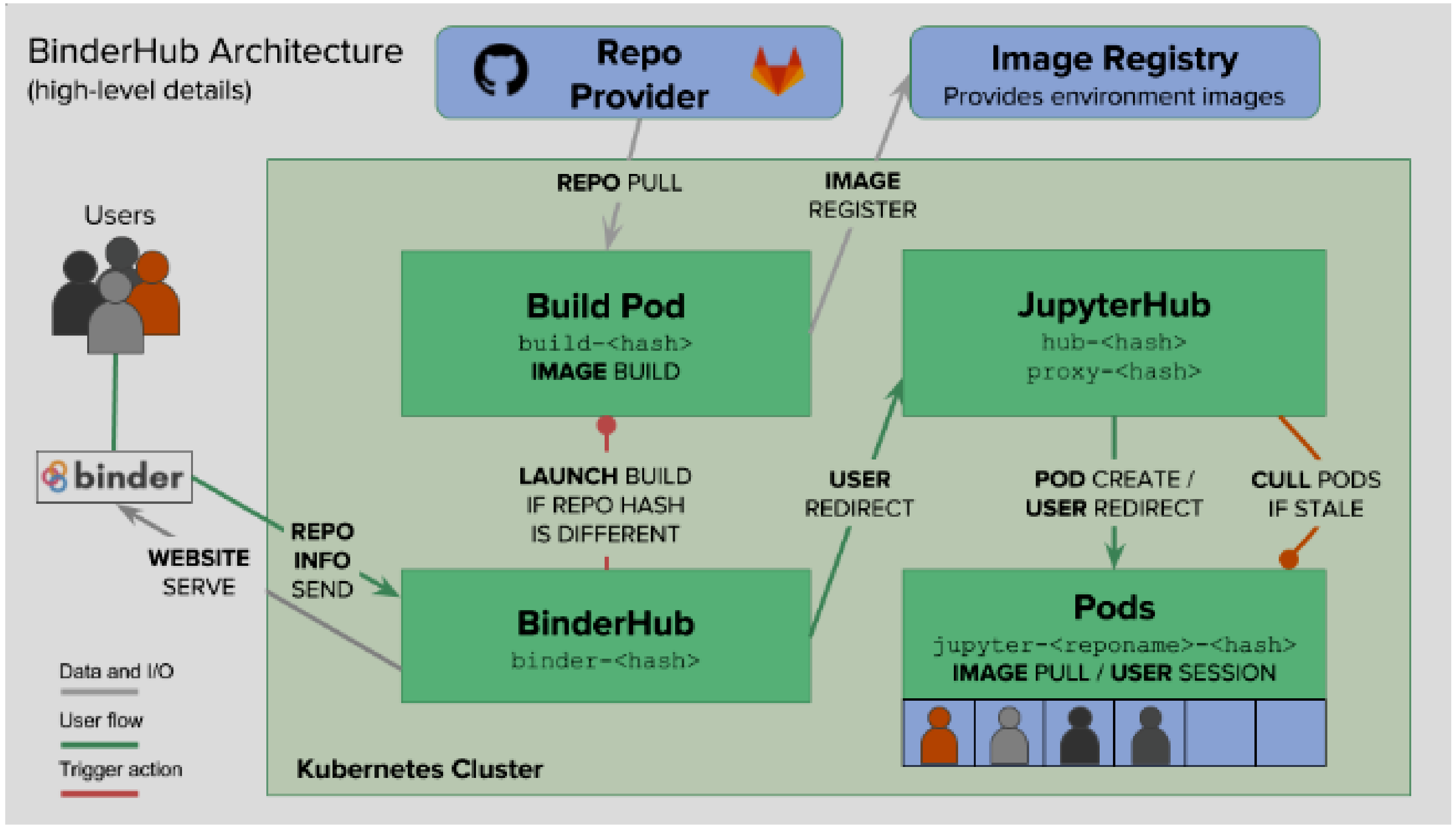


1

[1]Kubernetes Architecture | K8S DevOps – @tkssharma | Tarun Sharma | My Profile

Next Steps: Binderhub

- Click GitHub link and get a notebook (e.g., conda)!!
- Jupyter notebooks run in pods
- Provides replicated environments



1

[1] The BinderHub Architecture — BinderHub documentation

Next Steps: Academia

WARWICK

- Containers are used throughout academia. Replication ≠ reproduction?[1]
- Delved into some of the literature around Docker and R in [my previous presentation]
- Lots of fun papers.

[1]Juristo, N., & Vegas, S. (2010). Replication, reproduction and re-analysis: Three ways for verifying experimental findings. In *Proceedings 1st Int. Workshop on Replication in Empirical Software Eng. Research (RESER 2010)*.

Next Steps: Further reading

Boettiger, C. (2015). An introduction to Docker for reproducible research. *ACM SIGOPS Operating Systems Review*, *49*(1), 71-79.

Kurtzer, G. M., Sochat, V., & Bauer, M. W. (2017). Singularity: Scientific containers for mobility of compute. *PloS one*, *12*(5), e0177459.

Sultan, S., Ahmad, I., & Dimitriou, T. (2019). Container security: Issues, challenges, and the road ahead. *IEEE Access*, *7*, 52976-52996.

Wratten, L., Wilm, A., & Göke, J. (2021). Reproducible, scalable, and shareable analysis pipelines with bioinformatics workflow managers. *Nature methods*, *18*(10), 1161-1168.

Watada, J., Roy, A., Kadikar, R., Pham, H., & Xu, B. (2019). Emerging trends, techniques and open issues of containerization: a review. *IEEE Access*, *7*, 152443-152472.

Thanks for listening

Questions?