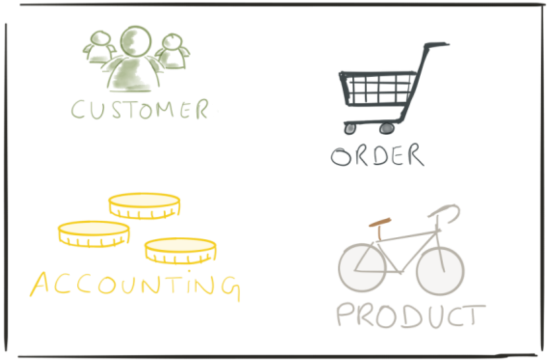
Microservices & Docker

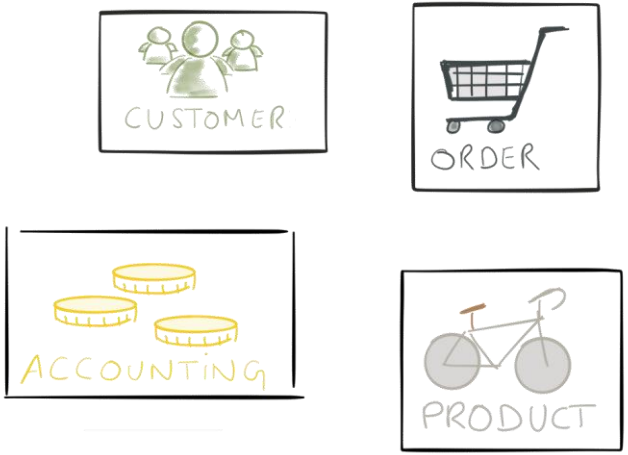
Microservices

- A software development technique
- A variant of the service-oriented architecture (SOA) architectural style
- Structure an application as a collection of loosely coupled services.
- Services are fine-grained and the protocols are lightweight.

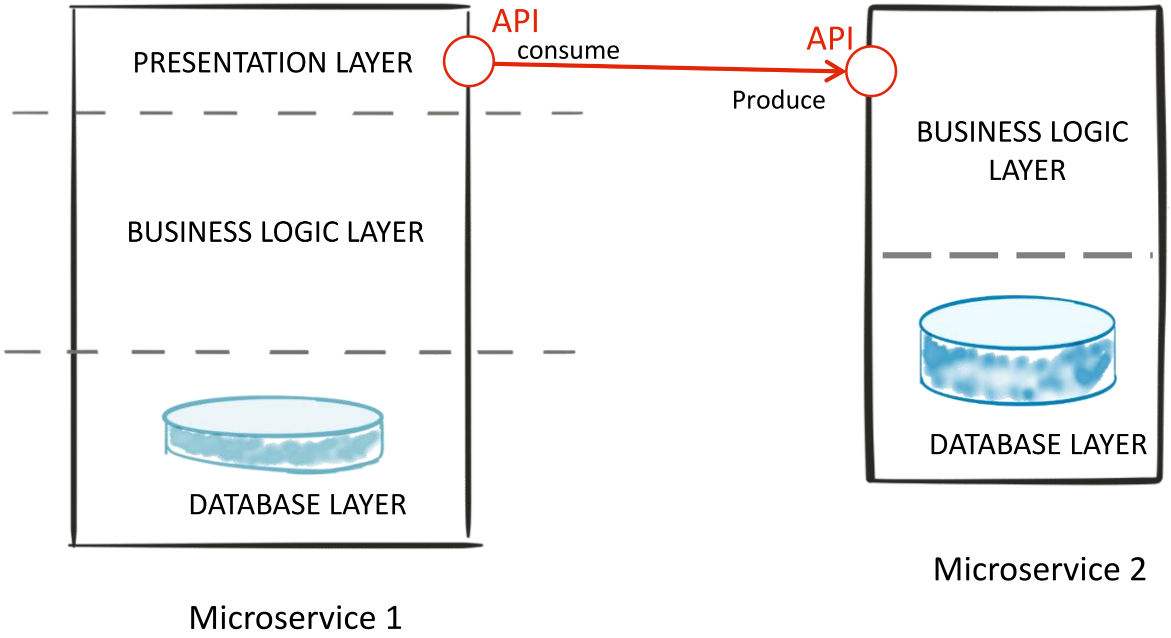
VS



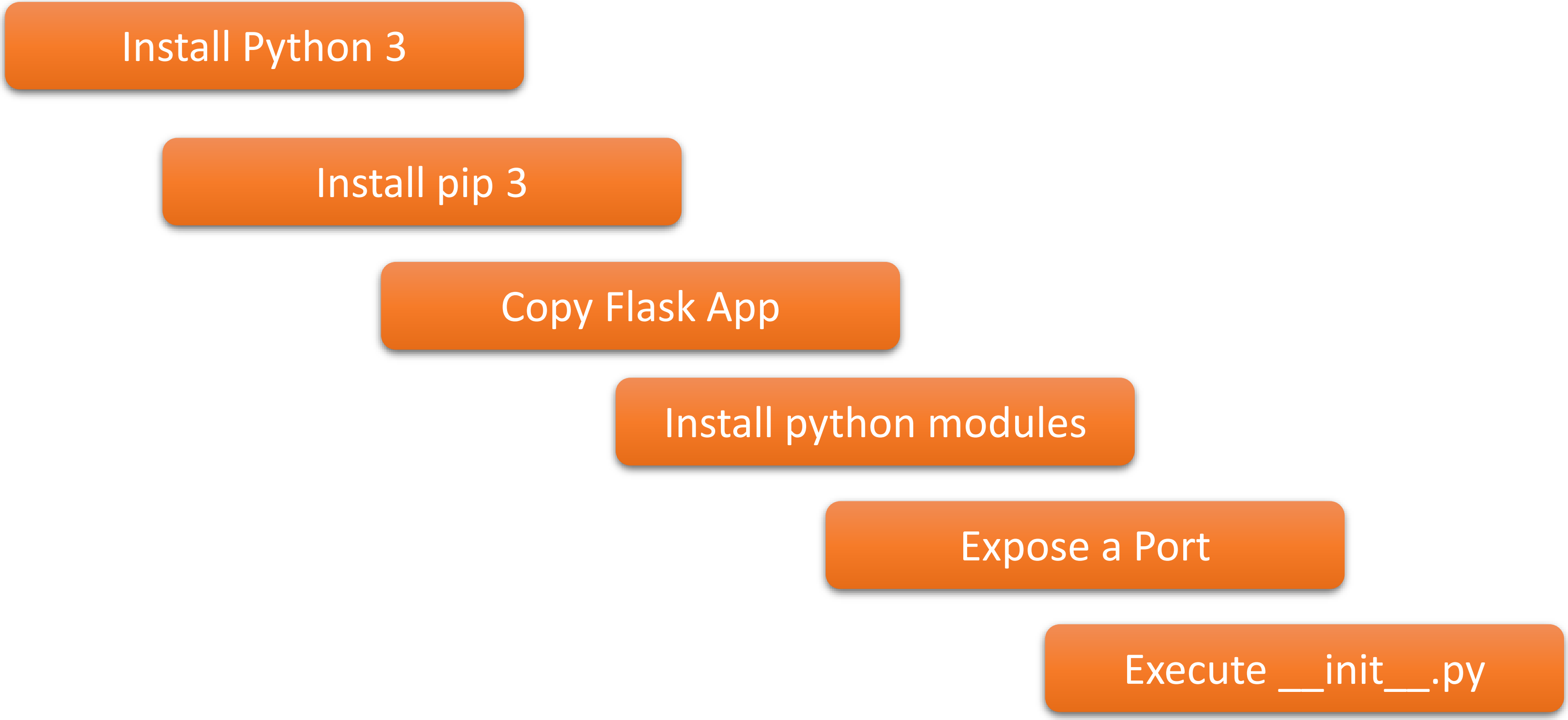
Monolithic



Microservices


API
consume
API
Produce
PRESENTATION LAYER
BUSINESS LOGIC LAYER
DATABASE LAYER
Microservice 1
BUSINESS LOGIC LAYER
DATABASE LAYER
Microservice 2

How to deploy the service?



Install Python 3

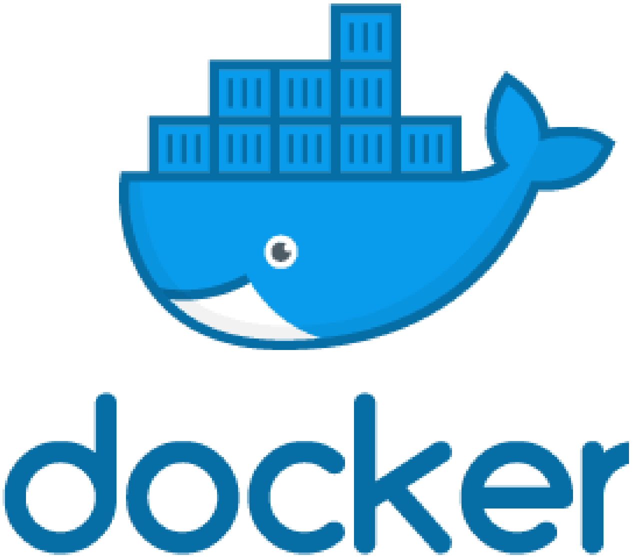
Install pip 3

Copy Flask App

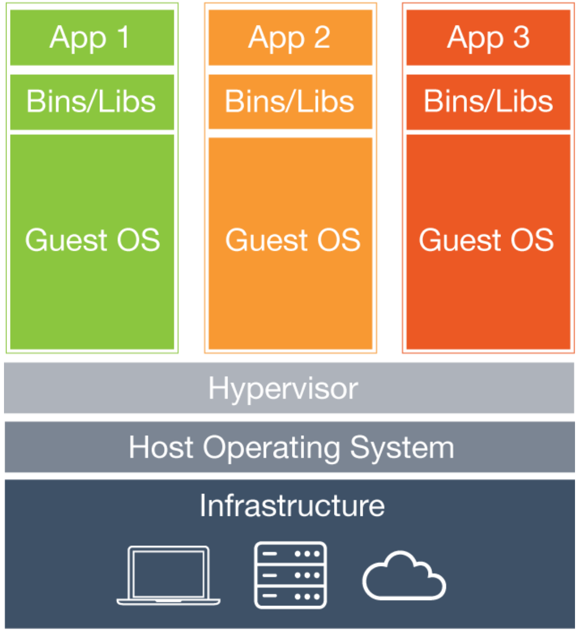
Install python modules

Expose a Port

Execute __init__.py



Docker is a platform for developers and system admins to **develop, deploy, and run** applications with containers



App 1
Bins/Libs
Guest OS

App 2
Bins/Libs
Guest OS

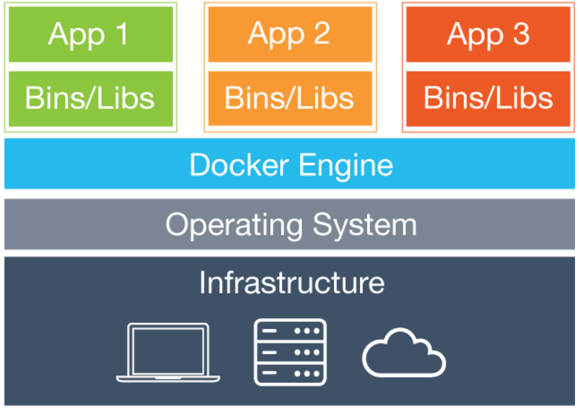
App 3
Bins/Libs
Guest OS

Hypervisor

Host Operating System

Infrastructure

Virtual machines



App 1
Bins/Libs

App 2
Bins/Libs

App 3
Bins/Libs

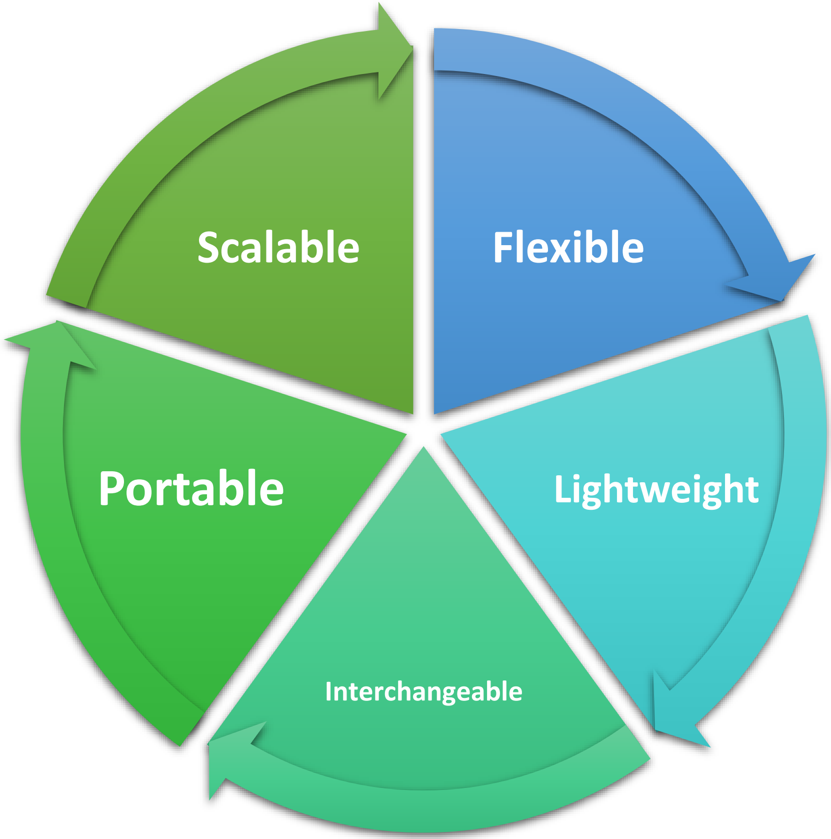
Docker Engine

Operating System

Infrastructure

Containers

Why Docker


Scalable
Flexible
Portable
Lightweight
Interchangeable

Docker Concepts

- An **image** is an executable package that includes everything needed to run an application--the code, a runtime, libraries, environment variables, and configuration files.

- A **Dockerfile** is a text document that contains all the commands to assemble an image.

- A **container** is a runtime instance of an image--what the image becomes in memory when executed.

- **Docker Hub** is the world's largest library and community for container images.
  https://hub.docker.com/

Activity 1: Install Docker

- Linux: https://docs.docker.com/install/linux/docker-ce/ubuntu/

- Windows 10 (Not Home edition): https://docs.docker.com/docker-for-windows/install/

- Mac: https://docs.docker.com/docker-for-mac/install/

Cheat sheet

```
## List Docker CLI commands
docker
docker container --help

## Display Docker version and info
docker --version
docker version
docker info

## Execute Docker image
docker run hello-world

## List Docker images
docker image ls


 ## List Docker containers (running, all, all in quiet mode)
docker container  ls
docker container --all
docker container  -aq
```

Activity 2: Create a Dockerfile

```
FROM alpine:latest
RUN apk add --no-cache python3-dev
RUN pip3 install --upgrade pip
WORKDIR /service
COPY .  /service
RUN pip3 install –r requirements.txt
EXPOSE 5000
CMD python3 __init__.py
___
ENTRYPOINT [“python3”]
CMD [“__init__.py”]
```

Activity 3: Build & Run

```
docker build -t my_service .
```

```
docker run -p 5000:5000 -t my_service
```

```
docker stop CONTAINER_ID
```

Cheat Sheet

- Stopping all containers

```
$ docker stop $(docker ps -aq)
```

- Removing all containers

```
$ docker rm $(docker ps -aq)
```

- List Docker containers (running, all)

```
$ docker container ls -a
```

Activity 4: Run another Instance

```
docker run -p 5000:5000 -t my_service
```

Open *Restlet Client* or *Postman* to test the Containers

Next Week’s Lab Activity

Create a Docker Image for the service you wrote in the first Lab

Note: Depending on your code, you might need to install various python packages or software. These should be added in the Dockerfile