Containers

COMP260: Distributed Systems

7: Containers

Register Attendance

Figure 1: Attendance monitoring is in place. It is your responsability to ensure that you have signed yourself in.

FALMOUTH
UNIVERSITY

What are containers?

One Definition

"A software container provides a standard packaging and distribution format that is generic and widespread, enabling greatly increased carrying capacity, lower costs, economies of scale and ease of handling."

(Arundel & Domingus - 2019)

"The container format contains everything the application needs to run, baked into an image file that can be executed by a container runtime (Docker in our case)."

(Arundel & Domingus - 2019)

Simply put. . .

Package Software into Standardized Units for Development, Shipment and Deployment

Docker vs. Virtual Machines

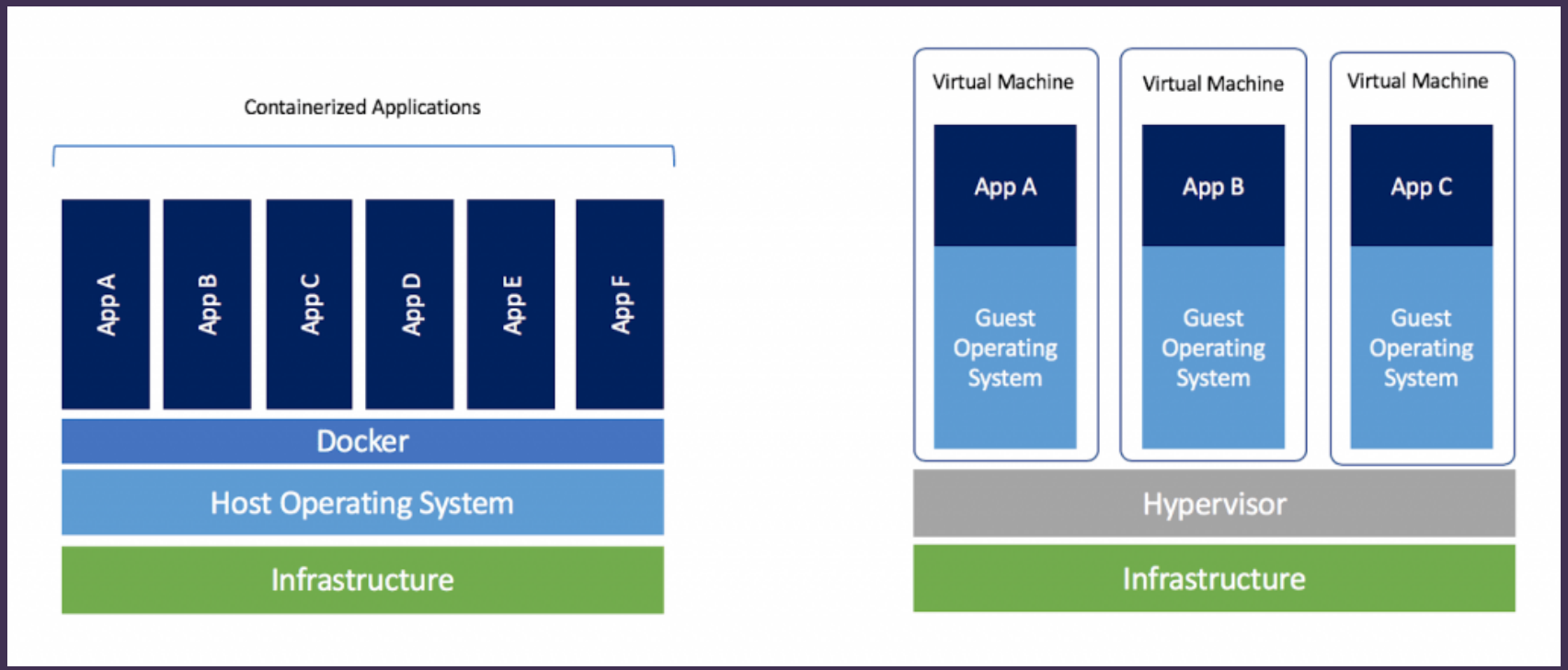


Figure 2: A hypervisor is a computer software, firmware or hardware that creates and runs virtual machines.

Some more info

Virual Machines	Containers
2GB+	10-150MB
Full operating System	Shares host kernel
Contains irrelevant files	Only required files
Emulated CPU	Run on host CPU
upto 30% slower	Runs like binary executable

Docker

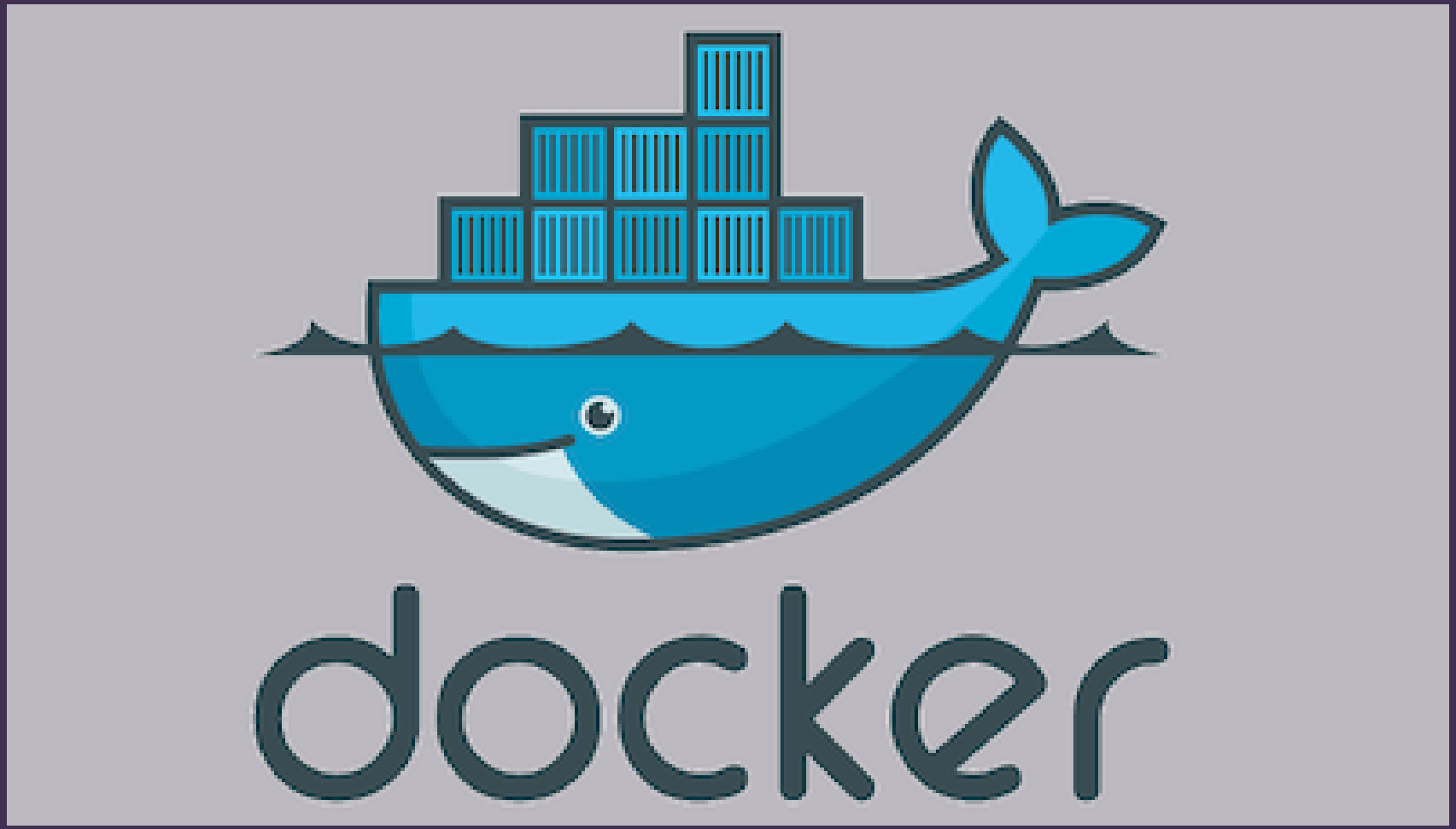
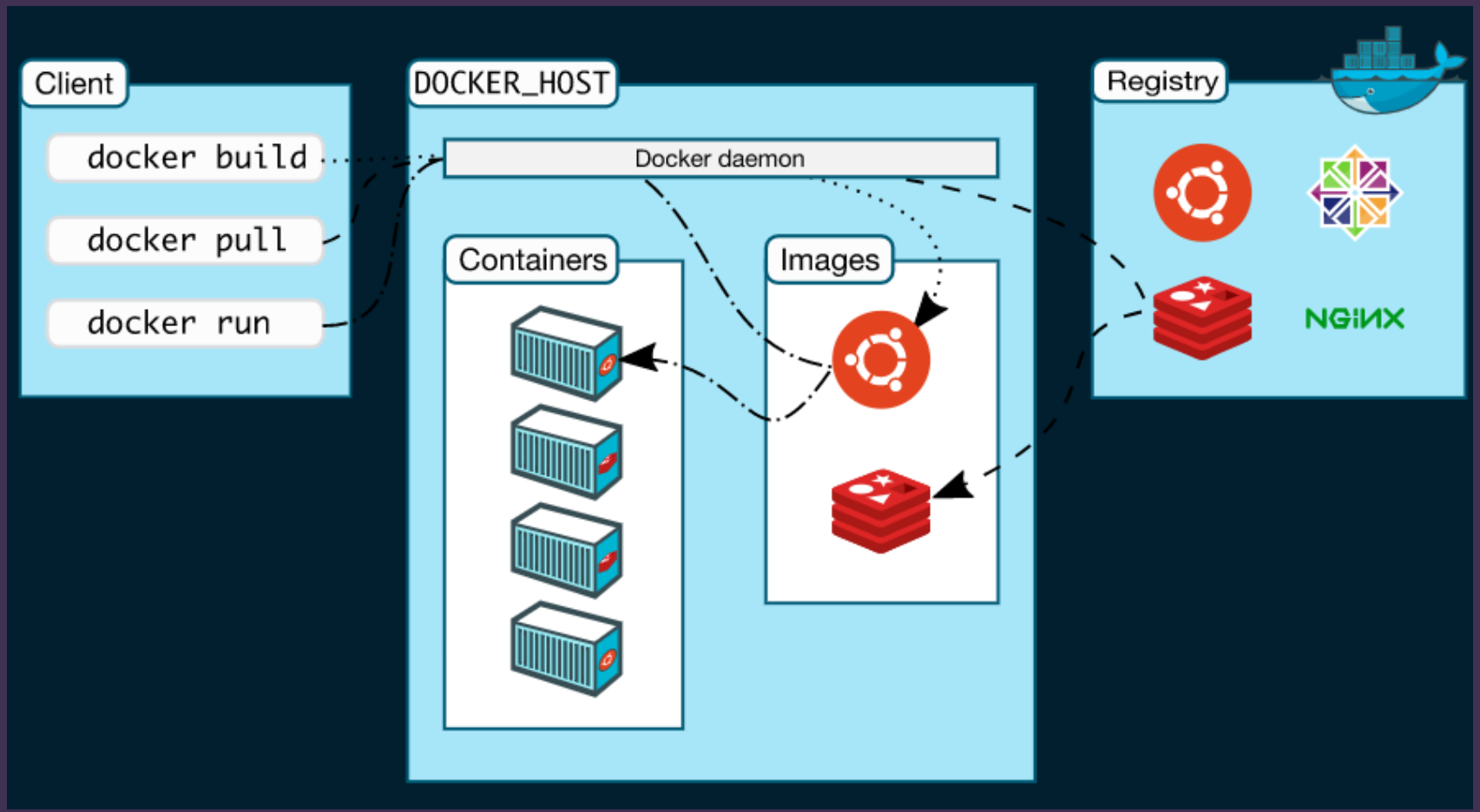


Figure 3: An open platform for developing, shipping, and running applications.

Architecture



Images

"An image is a read-only template with instructions for creating a Docker container." - Docker Docs

- Create an images using YAML inside a `Dockerfile'
- The YAML provides instructions for how to `build' the image
- Instructions create layers
- Only layers that change need to be rebuilt

Containers

"A container is a runnable instance of an image" - Docker Docs

- Verbs: create, start, stop, move, or delete
- Managed using the Docker CLI or API
- Isolated from other containers by default
- You control how network, storage and subsystem are

Services

"Services allow you to scale containers across multiple Docker daemons, which all work together as a swarm with multiple managers and workers." - Docker Docs

Dockerfile

Dockerfile > ...

```
1   FROM tiangolo/uwsgi-nginx-flask:python3.6-alpine3.7
2
3   ENV LISTEN_PORT=5000
4   EXPOSE 5000
5
6   # Indicate where uwsgi.ini lives
7   ENV UWSGI_INI uwsgi.ini
8
9   # Tell nginx where static files live.
10  ENV STATIC_URL /app/static
11
12  # Set the folder where uwsgi looks for the app
13  WORKDIR /app
14
15  # Copy the app contents to the image
16  COPY . /app
17
```

Commands - build

```
docker image build -t automatedchaos/20200308azuredocker:1.0 .
```

`-t` tag the image with a name and version

DON'T MISS THE DOT AT THE END

Commands - run

```
docker run --detach --publish 80:5000 --name webserver automatedchaos/20200308azuredocker
```

`--publish` forward incoming traffic on the host's port 80, to the container's port 5000

`--detach` run this container in the background

`--name` the name with which you can refer to your container in subsequent command

Commands - rm (remove)

```
docker container rm --force webserver
```

Docker Registry

"The Registry is a stateless, highly scalable server side application that stores and lets you distribute Docker images." Docker Docs

- Open source
- Free Docker Hub
- push and pull

Push in three steps. . .

STEP 1: Log in with CLI

```
docker login --username=yourhubusername
--email=youremail@company.com
```

STEP 2: Tag your image

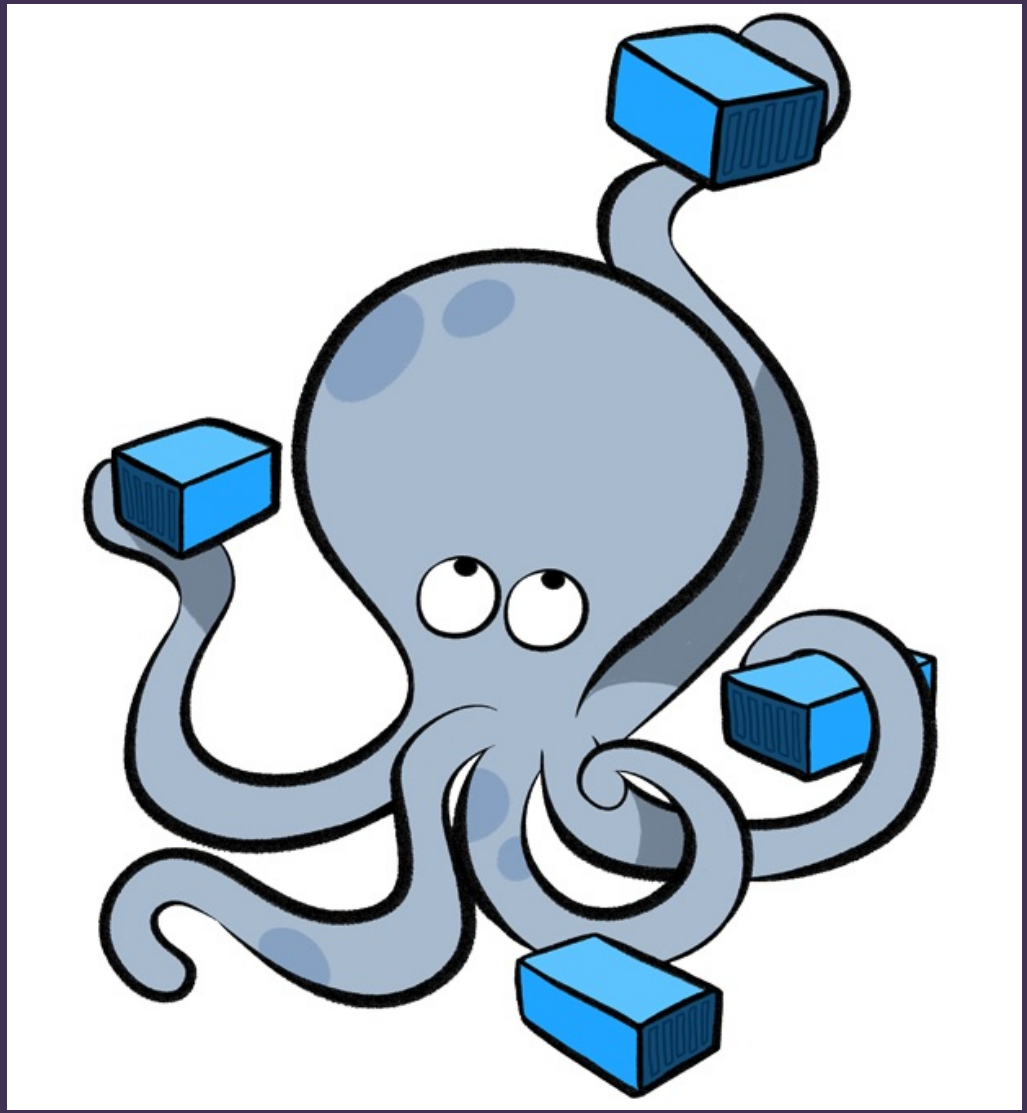
```
docker tag [IMAGE ID] yourhubusername/webserver:1.0
```

STEP 3: Push

```
docker push yourhubusername/webserver:1.0
```

DEMO TIME

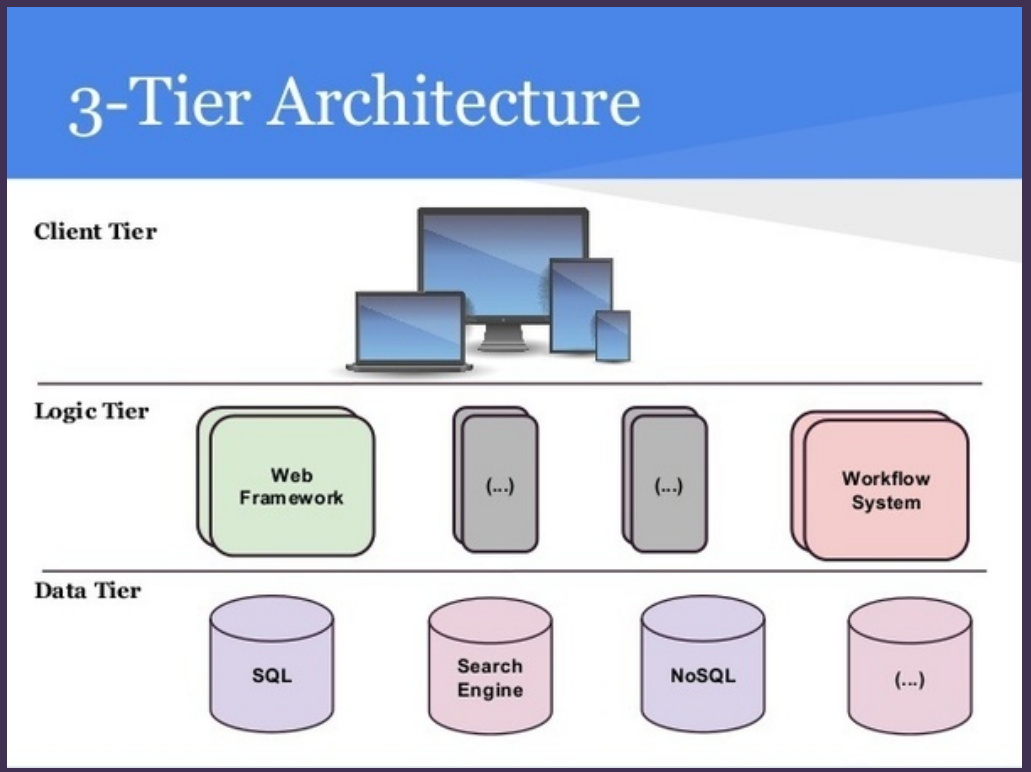
Docker Compose



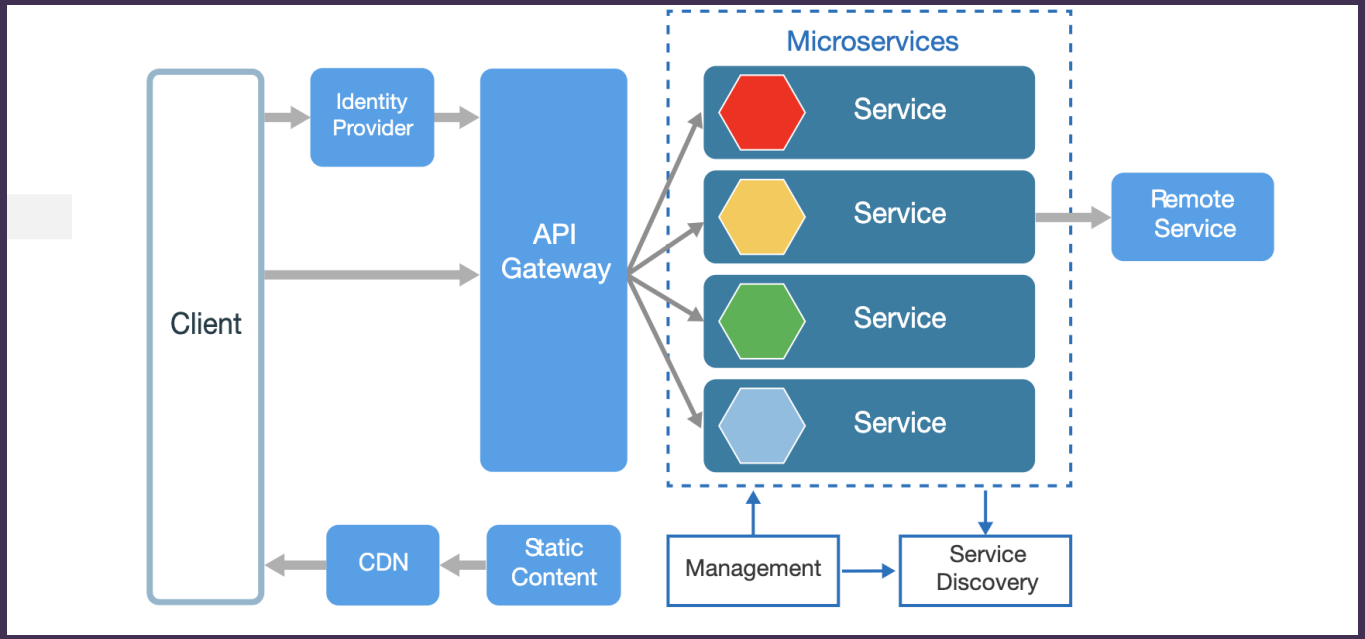
What is Docker Compose

- Compose is a tool for defining and running multi-container Docker applications.
- Great for smaller deployments that require a little extra automation
- Perfect for three tier architecture
- Single host deployment
- Works well with automated testing environments
- baby steps towards using larger scale orchestration tools

Three Tier Architecture



Microservices



Features

The features of Compose that make it effective are:

- Multiple isolated environments on a single host
- Preserve volume data when containers are created
- Only recreate containers that have changed = Variables and moving a composition between environments

Three Steps to Docker Compose Greatness

- Define each image in a standard Docker file
- define the relationships between each image in the docker-compose.yml file
- spin up your app using `docker-compose up`

You still manage the images and containers using docker CLI

Volumes

Volumes are a mechanism for persisting data generated by and used by Docker containers.

- shared with other containers
- Easy to back-up/migrate
- Volume drivers allow remote hosts and encryption
- Can be prepopulated

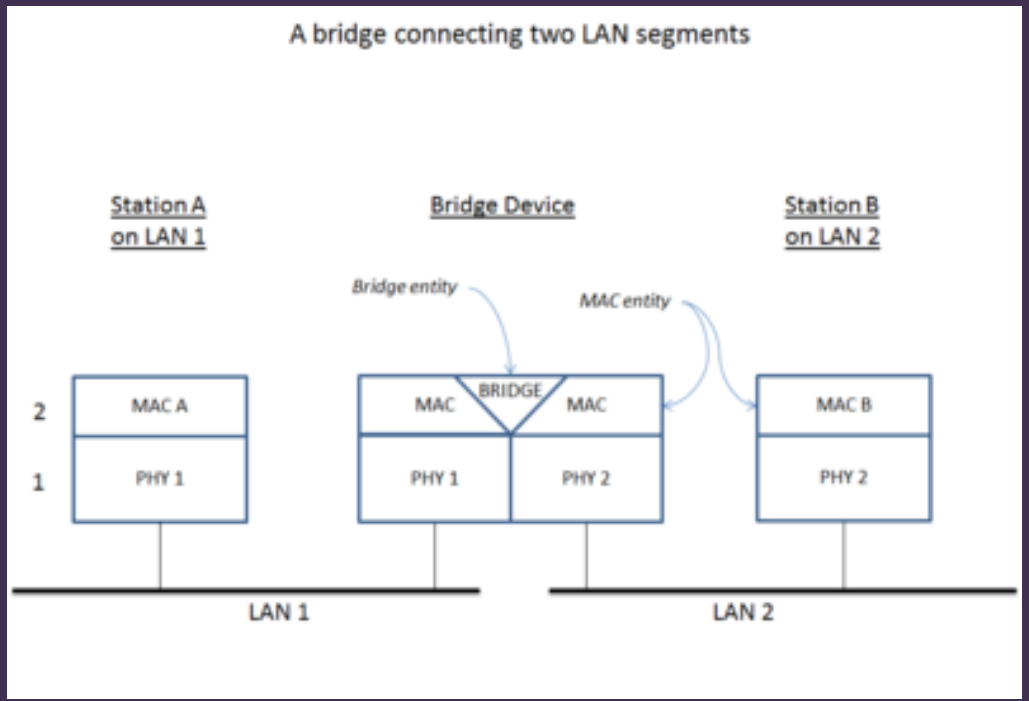
Networks

Networking allows individual containers to communicate with each other

- Networks are a separate entity in the docker world:
  `docker network ls`
- Creating custom networks is simple and advised
- Can be connected to external networks

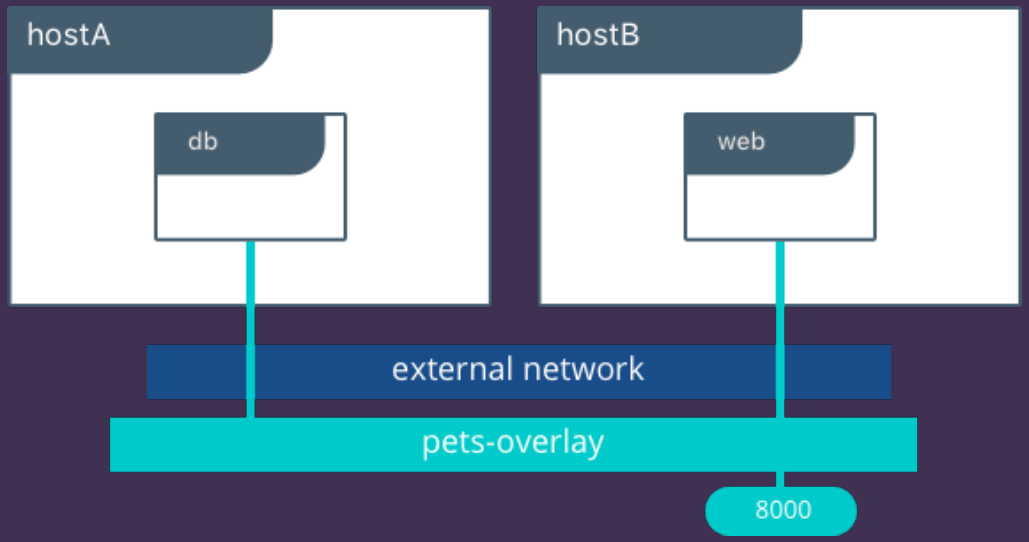
Basic Types of Network - **Bridge**

"A bridge network uses a software bridge which allows containers connected to the same bridge network to communicate, while providing isolation from containers which are not connected to that bridge network." - Docker Docs



Basic Types of Network - **Overlay**

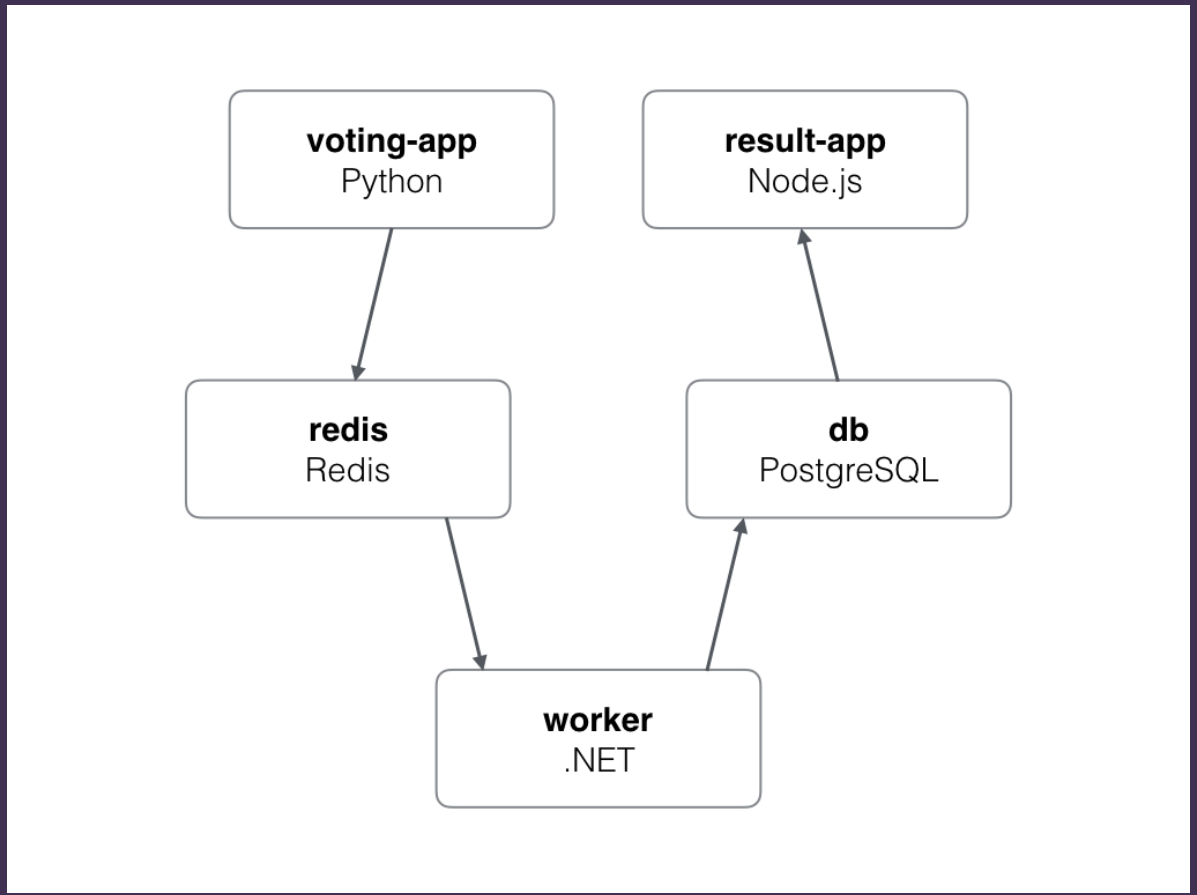
"The overlay driver (. . .) decouples the container network from the underlying physical network (the underlay). This has the advantage of providing maximum portability across various cloud and on-premises networks. Network policy, visibility, and security is controlled centrally through the Docker Universal Control Plane (UCP)." - Church, M, 2016



Suggestions

- Use version 3.0
- Keep your compose file the same in development and production
- Always use the Alpine version where possible
- Use the internal DNS rather than assigning specific IPs
- Share environment variables accross all containers

The Demo App


voting-app
Python
result-app
Node.js
redis
Redis
db
PostgreSQL
worker
.NET

Components

- A front-end web app in Python or ASP.NET Core which lets you vote between two options
- A Redis or NATS queue which collects new votes
- A .NET Core, Java or .NET Core 2.1 worker which consumes votes and stores them in. . .
- A Postgres or TiDB database backed by a Docker volume
- A Node.js or ASP.NET Core SignalR webapp which shows the results of the voting in real time

docker-compose.yml

Let's take a look The Official Example App