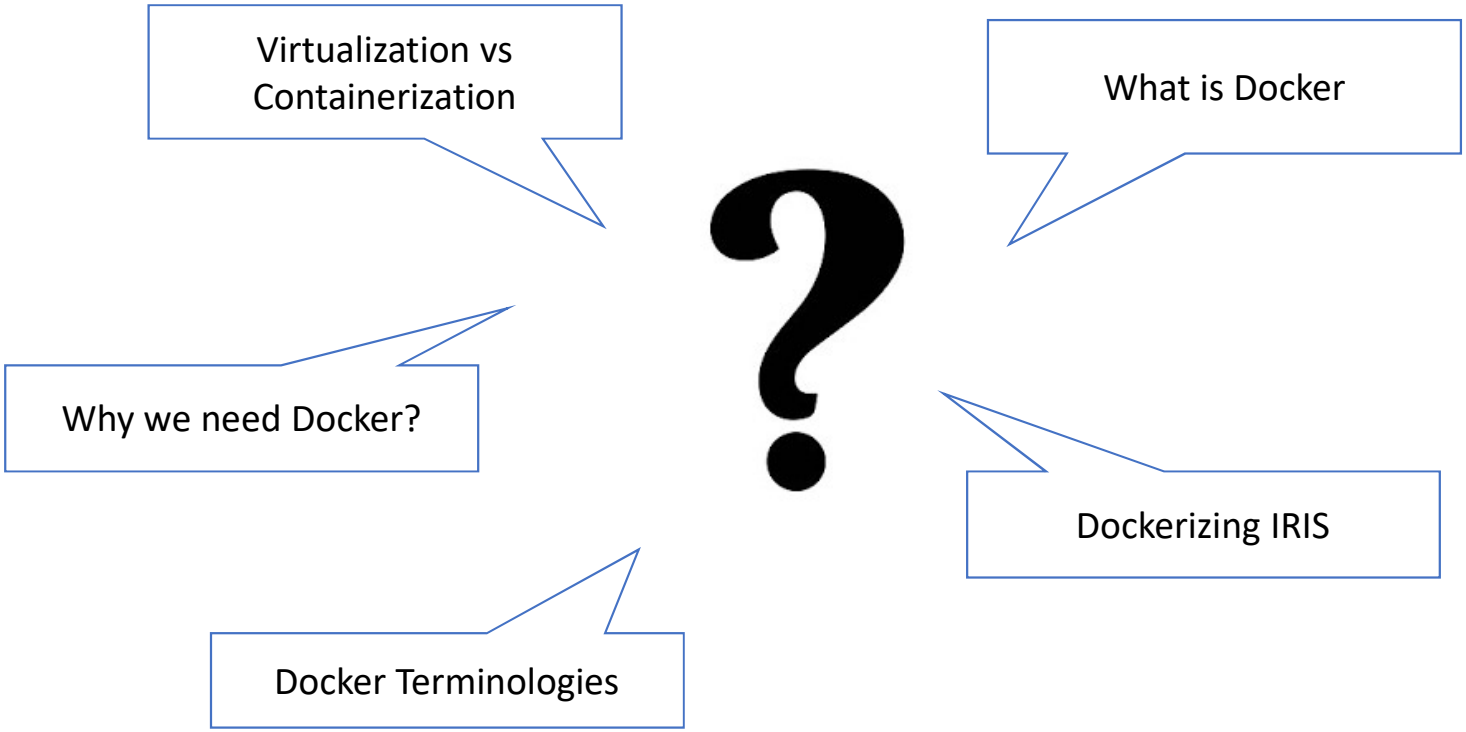
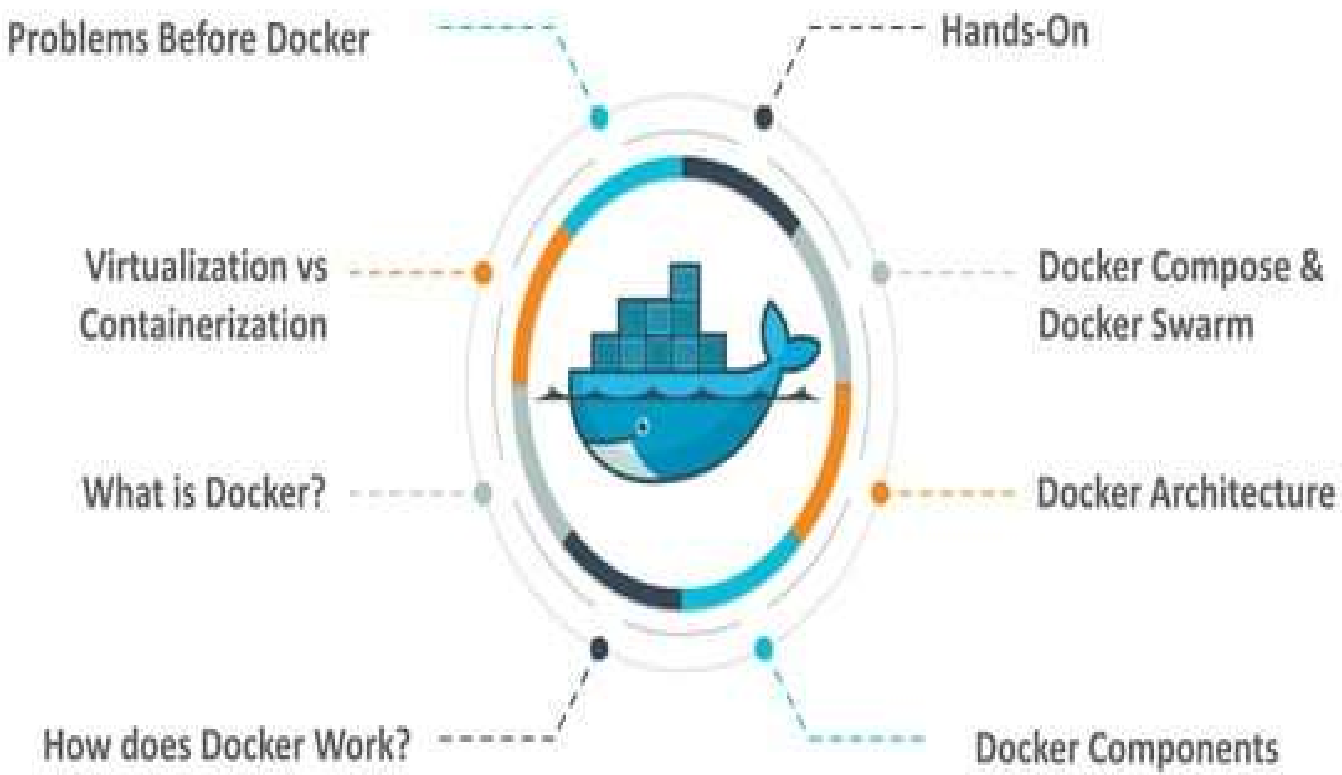
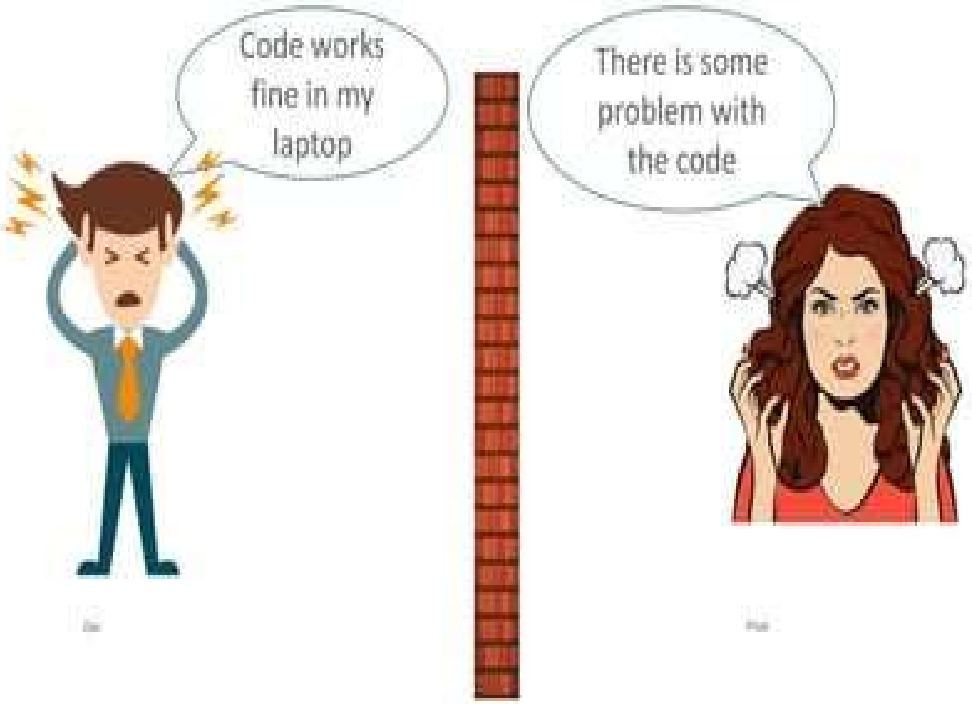

Virtualization vs Containerization
What is Docker
Why we need Docker?
Dockerizing IRIS
Docker Terminologies

Topics For Today's Session

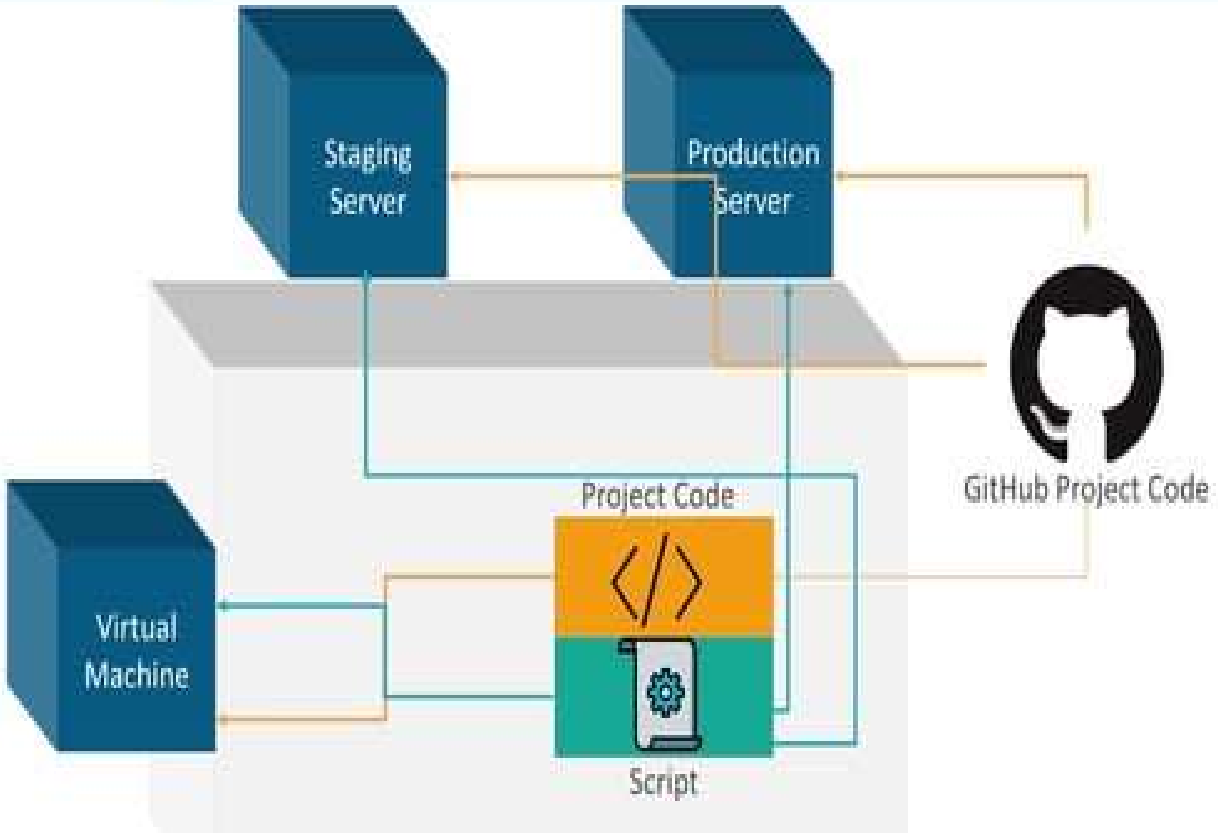


Problems Before Docker

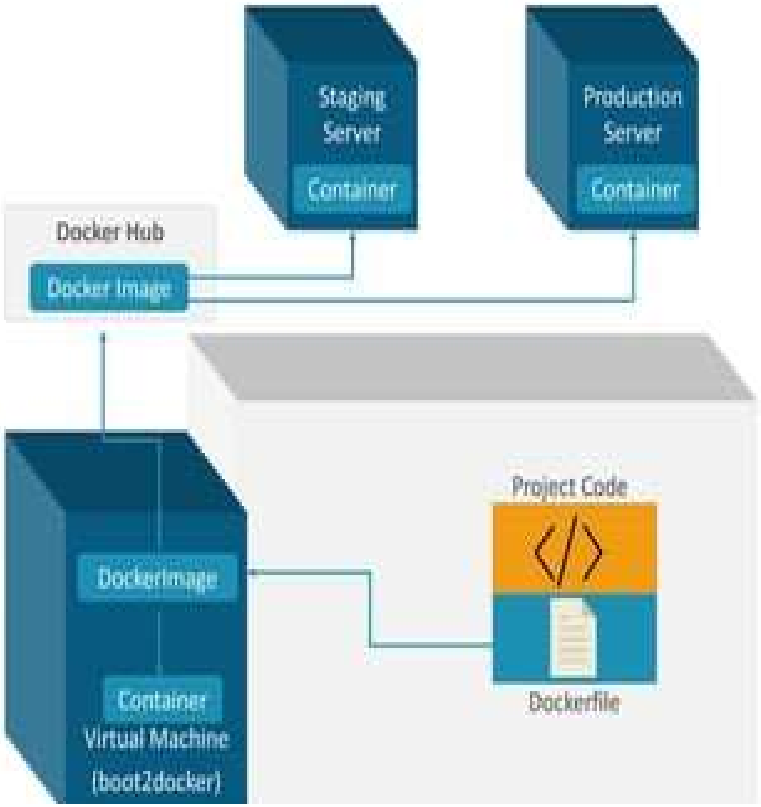



THE DEPENDENCIES WILL KILL YOU

Scenario Using Virtual Machines



Scenario After Using Docker



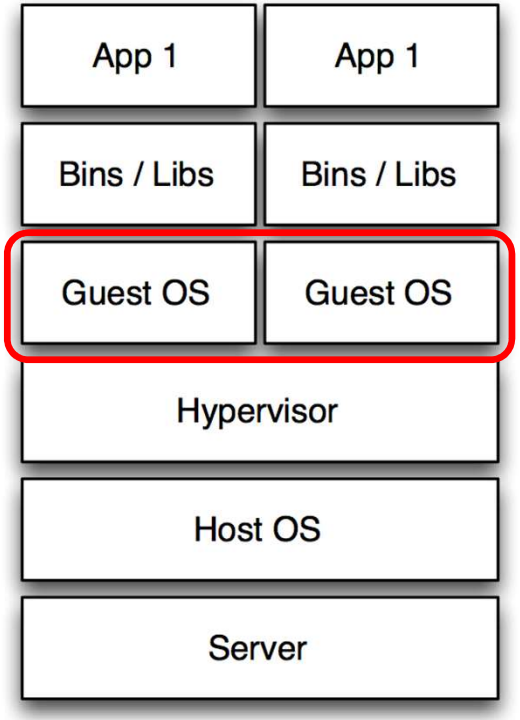

docker

Docker: Containerization for Software

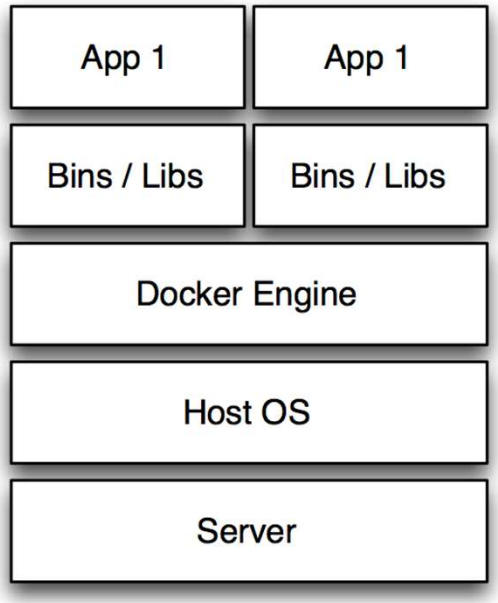

docker

VIRTUALIZATION
VS
CONTAINERIZATION

VM vs. Docker (Containers)



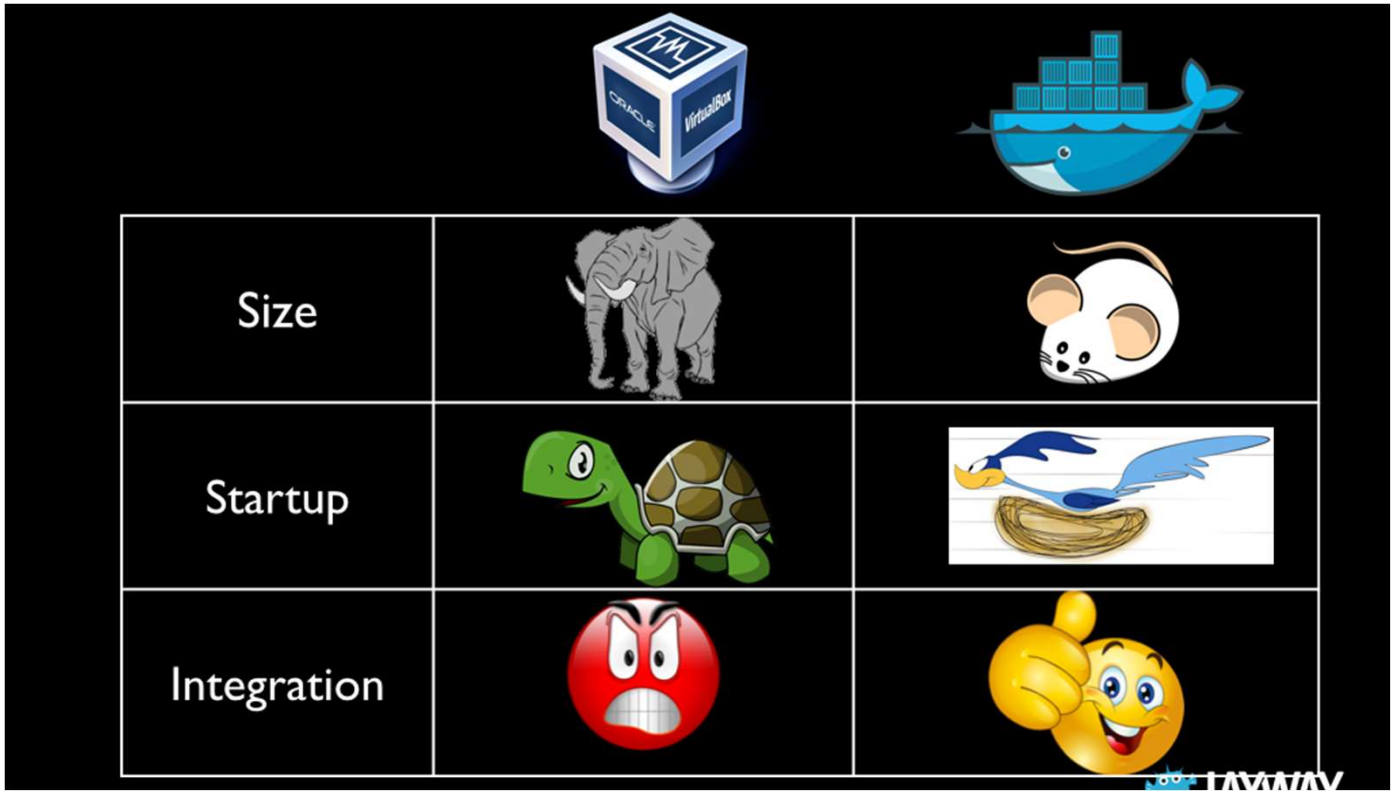
Virtual Machines



Docker

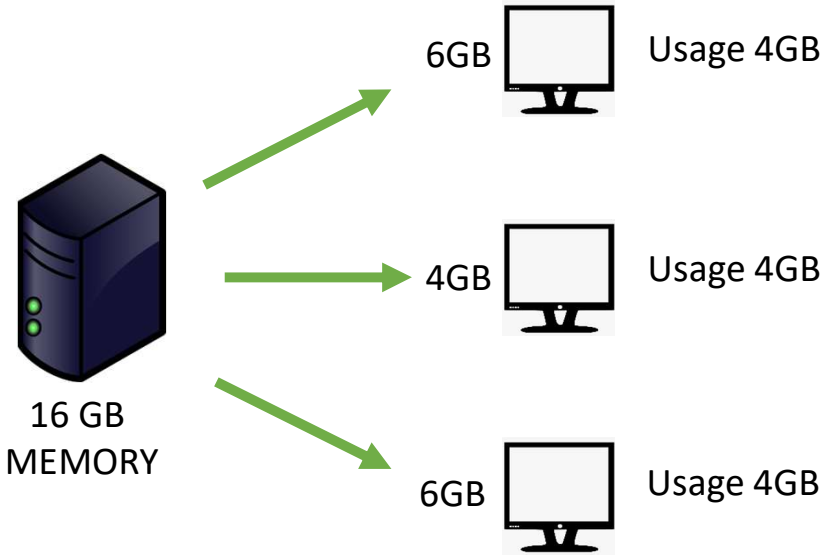
Docker Engine
Docker engine is the layer on which Docker runs.
It's a lightweight runtime and tooling that manages containers, builds, and more.

VM vs Docker



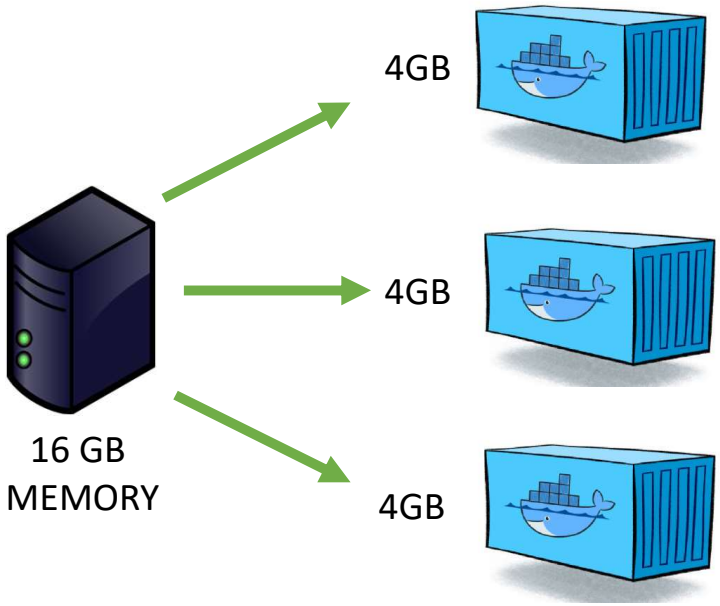
VM vs Docker

In case of VM



4 GB MEMORY remain unused and cannot be allocated to another VM

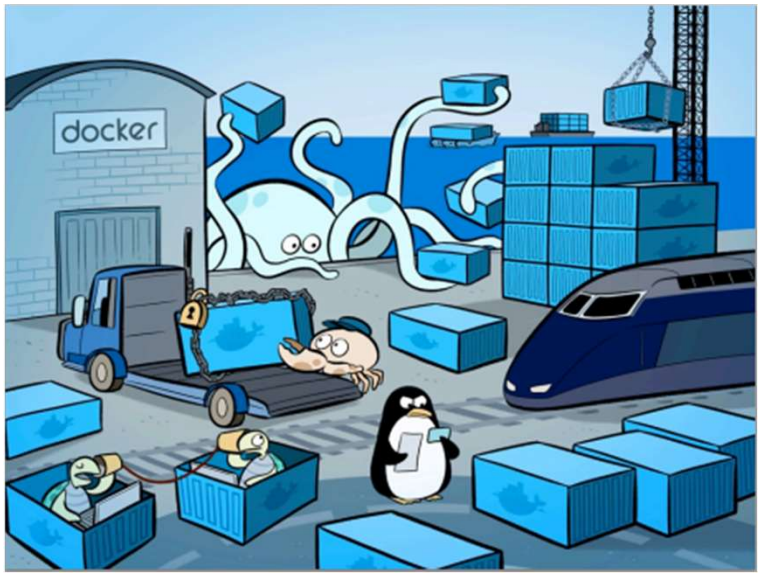
In case of Docker



4 GB MEMORY remain unused and can be allocated to another container as containers share resources

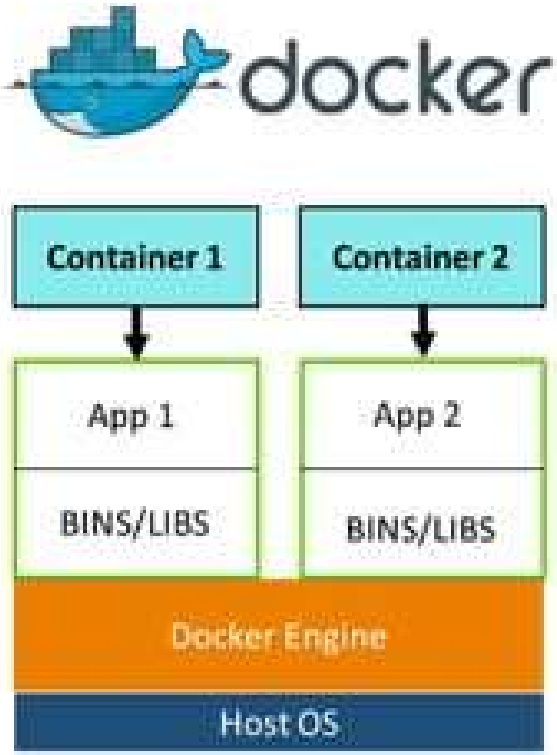
WHAT IS DOCKER?

Docker



“Docker is an open platform for developers and sysadmins to build, ship, and run distributed applications”

What is Docker?



- Docker is a tool designed to make it easier to create, deploy, and run applications by using containers.
- Docker containers are lightweight alternatives to Virtual Machines, and it uses the host OS.
- You don't have to pre-allocate any RAM in containers.

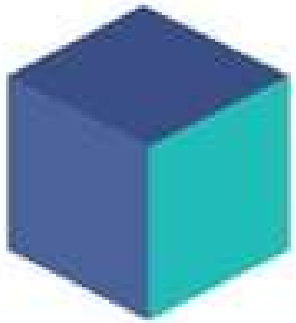
How Does Docker Work?

Docker Engine uses a Client Server Architecture.

Components of Docker

Dockerfile

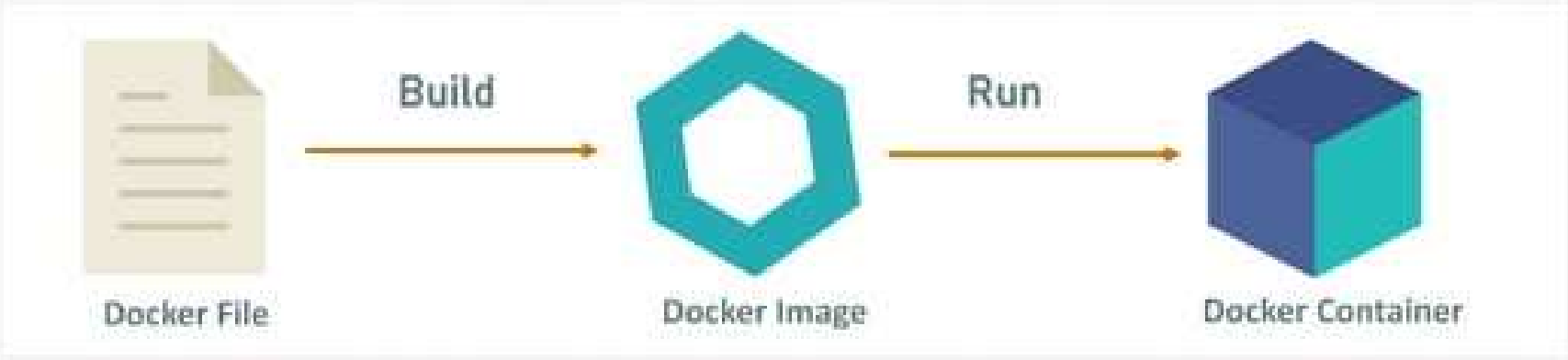
DockerImage



Docker Container

Docker Registry

Docker File, Images & Container



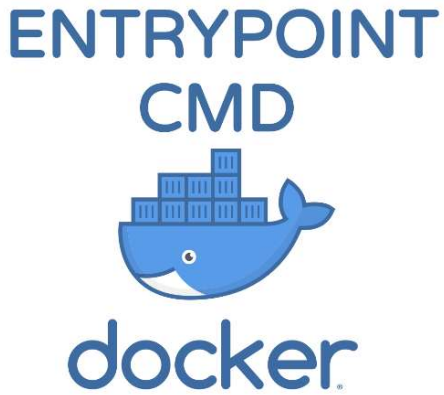
A text document which contains all the commands that a user can call on the command line to assemble an image.

Read Only Template Used To Create Containers
Built By Docker Users
Stored In Docker Hub Or Your Local Registry

Isolated Application Platform
Contains Everything Needed To Run The Application
Built From One Or More Images

Dockerfile

- A text document that contains all the commands a user could call on the command line to assemble an image.
- Executes several command-line instructions in succession.
- To build an image from docker file execute
  docker build . –t imageName:tag




```
FROM centos
USER iris
WORKDIR /home/iris
RUN mkdir -p Core
ENV USER iris
ADD ./CoreSource Core
COPY entrypoint.sh .
RUN chown -R iris:iris /home/iris/Core
ENTRYPOINT ["../entrypoint.sh"]
~
~
~
~
~
```

Docker Registry

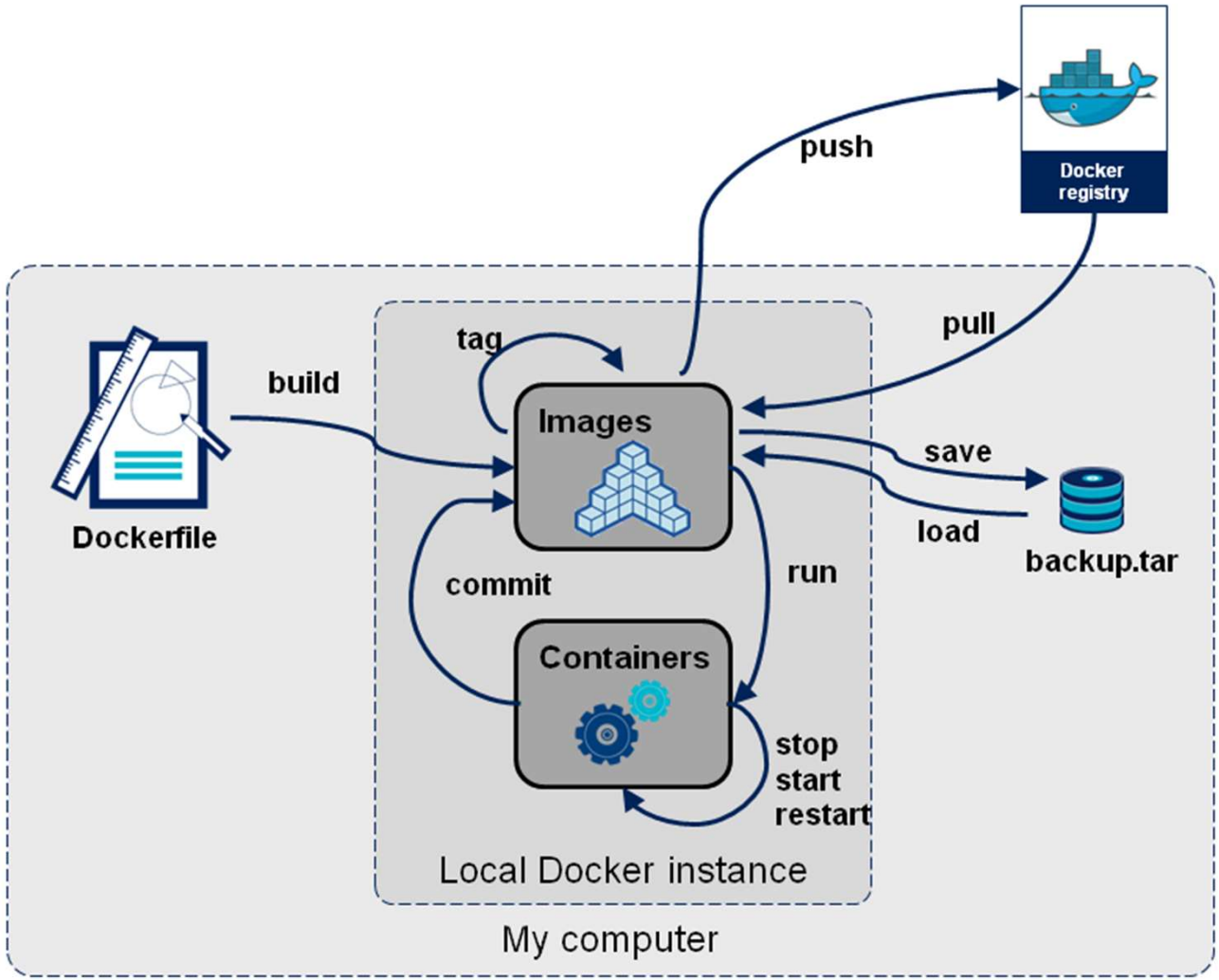
- Docker Registry is a storage component for Docker Images
- We can store the Images in either Public / Private repositories
- Docker Hub is Docker's very own cloud repository

Why Use Docker Registries?

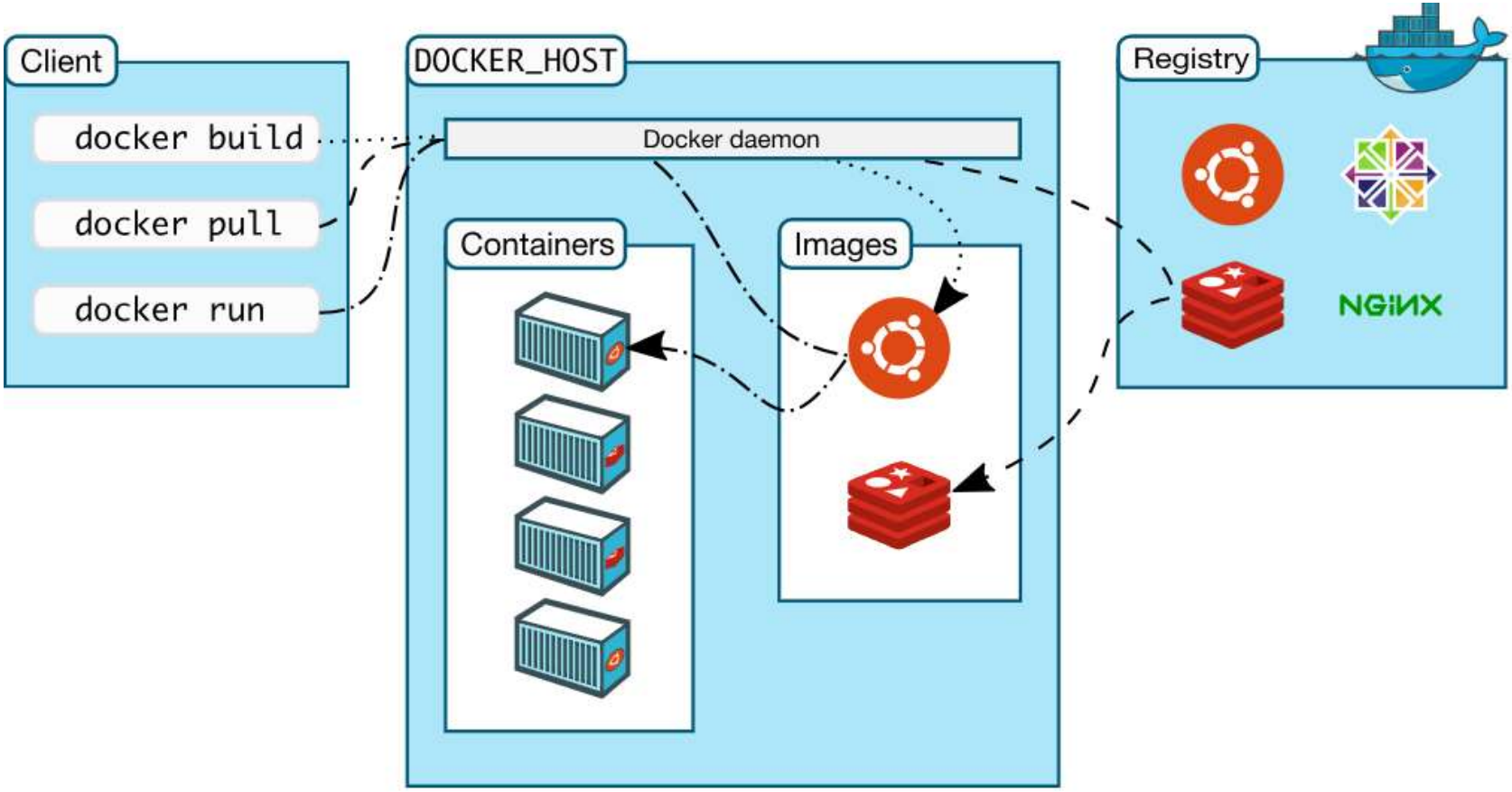
- Control where your Images are being stored
- Integrate image storage with your in-house development workflow

Docker Compose

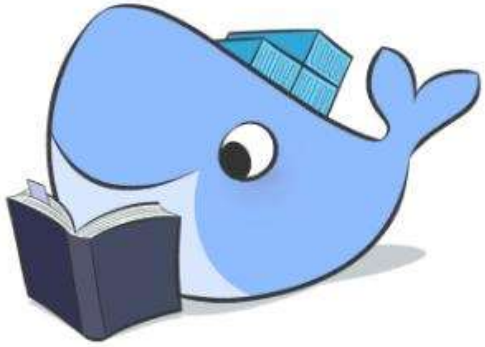
- Compose is a tool for defining and running multi-container Docker applications
- Uses a yaml file to configure application's services
  docker-compose.yml
- A single command creates and starts all the services
  docker-compose up


Docker registry
push
pull
tag
build
Images
save
load
backup.tar
Dockerfile
commit
run
Containers
stop
start
restart
Local Docker instance
My computer

Docker's Working Architecture



Docker Commands



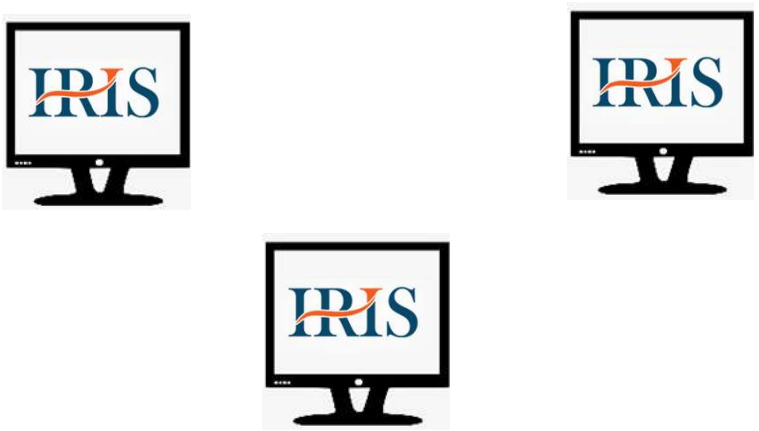
- Check version
  docker --version
- Create an image
  docker build . –t tagname
- Show images
  docker images
  docker image ls
- Remove image
  docker rmi imageID

- Run container
  docker run imageName
- Start container
  docker start containerId
- Stop container
  docker stop containerId
- List container
  docker ps -a
- Remove container
  docker rm containerId

WHY WE NEED DOCKER?

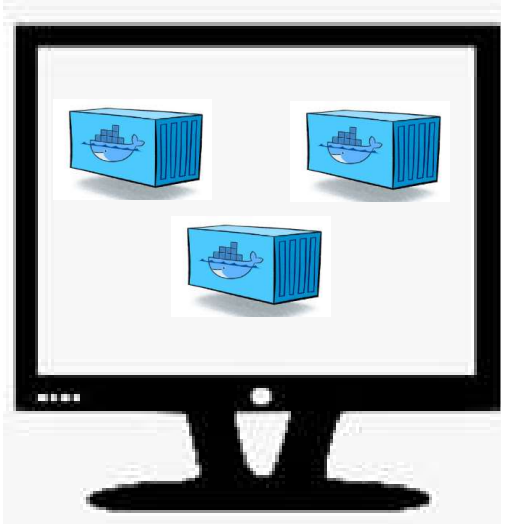
Why we need Docker?

Deploying IRIS through installer requires multiple VMs



Costly due to Infrastructure Requirements

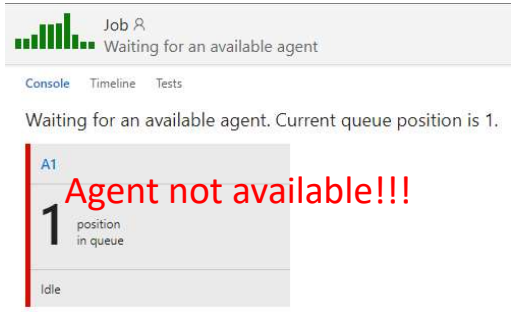
Integration in Docker is cheaper, faster and scalable



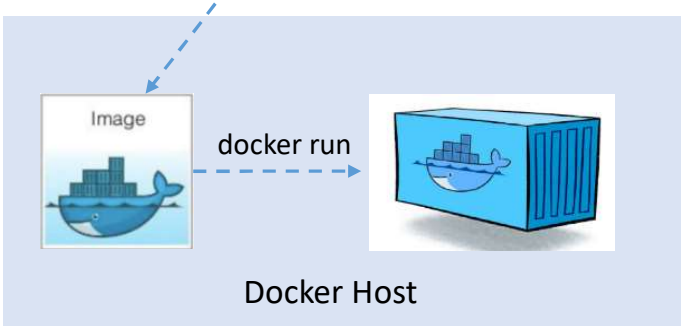
Why we need Docker? (Contd...)

TFS is slow

Build Server is slow

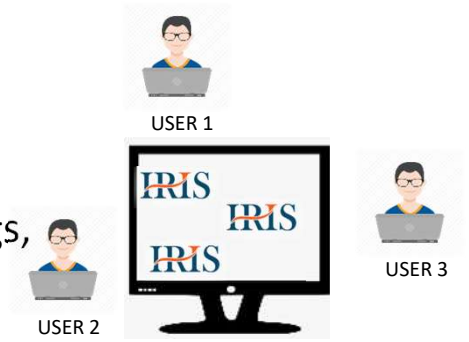


Agent not available!!!

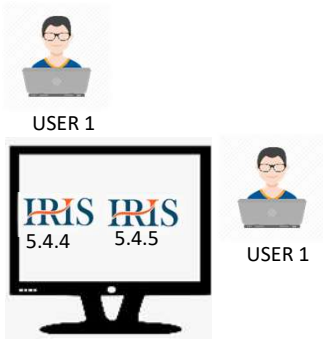


Why we need Docker? (Contd...)

- Allows anyone to work on the same project with the same settings, irrespective of the local host environment.



- Multiple development/qa environments can be run from the same host each one having different configurations



- Projects can be tested on different OS.

Why we need Docker? (Contd...)

- Ease of use

  The mantra is: “build once, run anywhere.”

- Speed

  Docker containers are very lightweight and fast

- Docker Hub

  “app store for Docker images.” Docker Hub has tens of thousands of public images created by the community that are readily available for use.

- Modularity and Scalability

  It’s become easier to link individual containers together to create your application, making it easy to scale or update components independently in the future.

Docker Benefits

1. Local development environments can be set up that are exact replicas of a live environment/server.
2. It simplifies collaboration by allowing anyone to work on the same project with the same settings, irrespective of the local host environment.
3. Multiple development environments can be run from the same host each one having different configurations, operating systems, and software.
4. Projects can be tested on different servers.
5. It gives you instant application portability. Build, ship, and run any application as a portable container that can run almost anywhere.

Why Docker?

- **Ease of use.** It allows anyone to package an application on their laptop, which in turn can run unmodified anywhere
  - The mantra is: “build once, run anywhere.”
- **Speed.** Docker containers are very lightweight and fast. Since containers are just sandboxed environments running on the kernel, they take up fewer resources. You can create and run a Docker container in seconds, compared to VMs which might take longer because they have to boot up a full virtual operating system every time.
- **Docker Hub.** Docker users also benefit from the increasingly rich ecosystem of Docker Hub, which you can think of as an “app store for Docker images.” Docker Hub has tens of thousands of public images created by the community that are readily available for use.
- **Modularity and Scalability.** Docker makes it easy to break out your application’s functionality into individual containers. With Docker, it’s become easier to link containers together to create your application, making it easy to scale or update components independently in the future.

THANK YOU!!