

GDMS – WHO WE ARE



CYBER









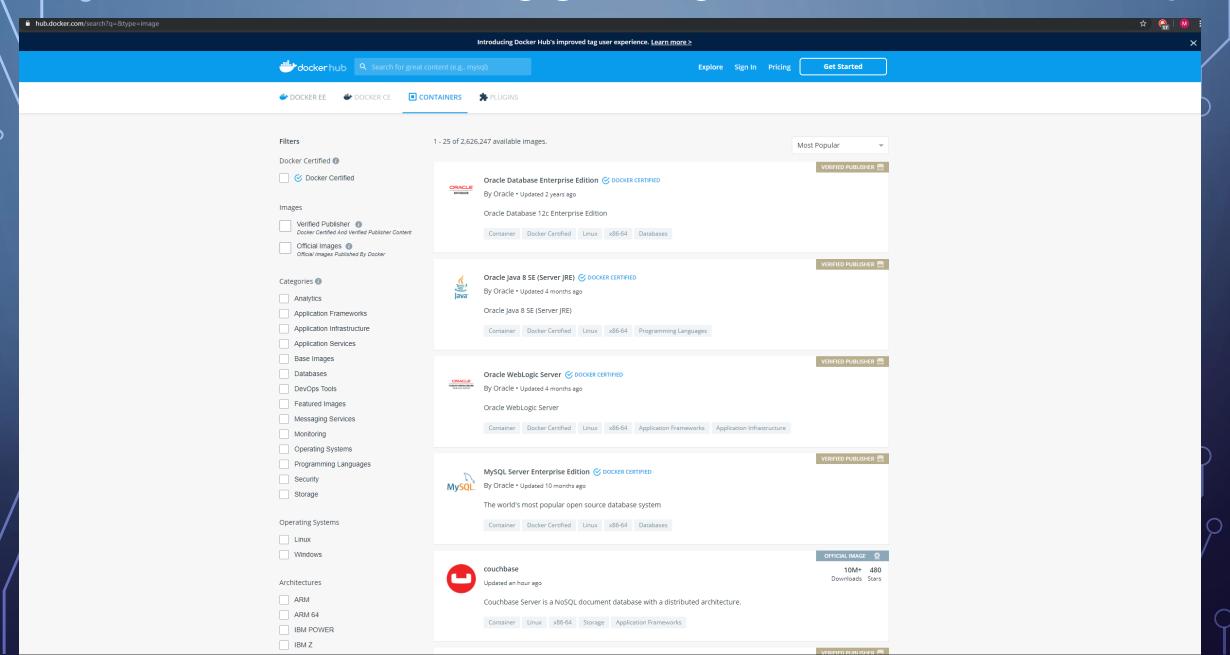


- Allows us to package code into a runnable image
- Portable over various OS Linux/Mac/Windows
- Easily start/stop parts of our application
- Reuse functionality by reusing images

CONTAINERS VS. VIRTUAL MACHINES

VMs	Containers
Heavyweight	Lightweight
Limited performance	Native performance
Each VM runs in its own OS	All containers share the host OS
Hardware-level virtualization	OS virtualization
Startup time in minutes	Startup time in milliseconds
Allocates required memory	Requires less memory space
Fully isolated and hence more secure	Process-level isolation, possibly less secure

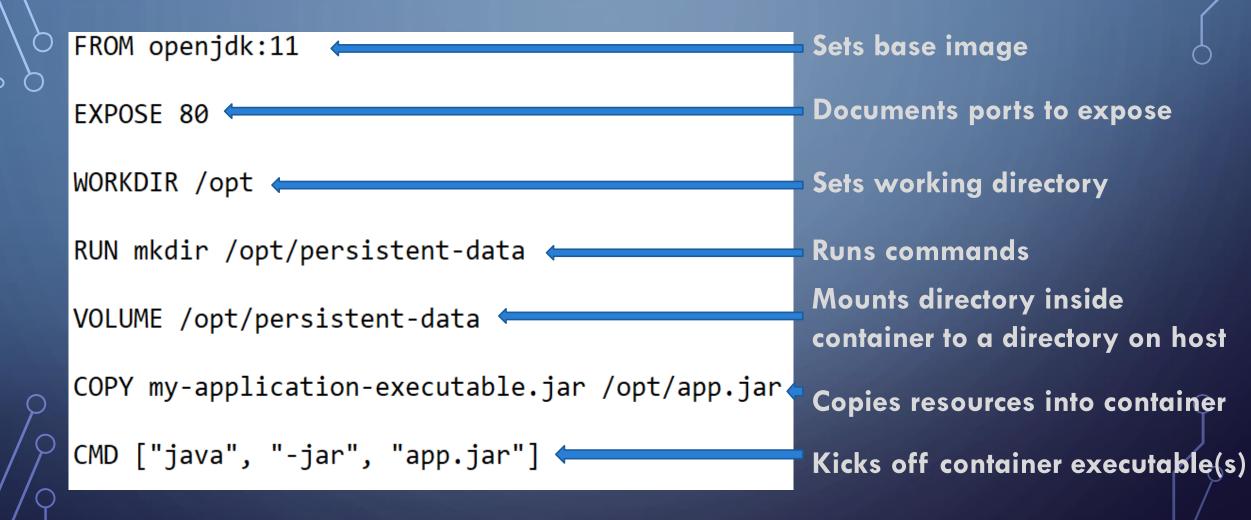
DOCKER HUB



CREATING OUR OWN IMAGES

- 1. Pick an appropriate base image (e.g. OpenJDK/Python/etc.)
- 2. Create a Dockerfile for our application
- 3. Build the Docker image and tag with a name

EXAMPLE DOCKERFILE



Note: Best practice is to name this file "Dockerfile" (no extension)

BUILDING DOCKER IMAGE

- To build, navigate to directory with Dockerfile and run:
 - \$ docker build -t my-application-name:1.0.0 .
- This creates an image named "my-application-name" version 1.0.0
- You can see all docker images by running "docker images":

<pre>\$ docker images</pre>				
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
openjdk	8	e8d00769c8a8	4 days ago	488MB
my-application-name	1.0.0	e8d00769c8a8	4 days ago	488MB

RUNNING DOCKER IMAGE

bo To run basic image, you can use:

\$ docker run -it my-application-name:1.0.0

•Some basic flags for run command:

-p 80:8080 Publish host port to container port (respectively)

-d Runs the container detached (in background)

-it Start in interactive mode with TTY

-e VAR=foo Declares an environment variable inside the container

GENERAL DYNAMICS Mission Systems

Questions after the workshop?

Matt McQuin
Matthew.mcquin@gd-ms.com
Matt.mcquin@gmail.com