Self Paced Lab 4 - Deploy containerized app in AWS EC2 (Docker)

Why do we need to learn this topic? –

- Containers are a very popular app packaging and deployment stack for micro-services
- Developers / Cloud engineers are often tasked to containerize apps for deployment
- Containerized app are the gaining acceptance as a deployment strategy in CI / CD lifecycle

What will we learn: -

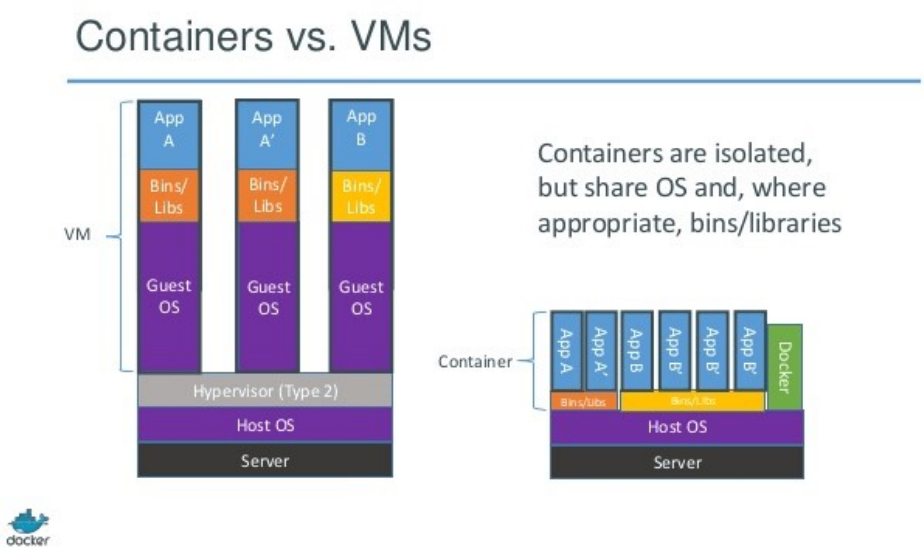
1. How to setup Docker in EC2 VM in AWS - Ubuntu
2. How to dockerize the demo "dotnet react" app and access it via a public URL
3. Launch a third party dockerized app in a local Ubuntu VM instance

Pre-requisites-

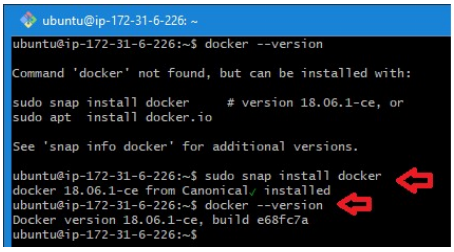
- **Disconnect from project / client VPN** – Verify that IP country is India @ https://www.iplocation.net/

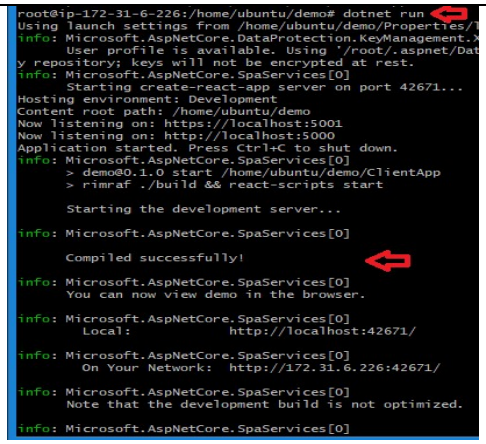
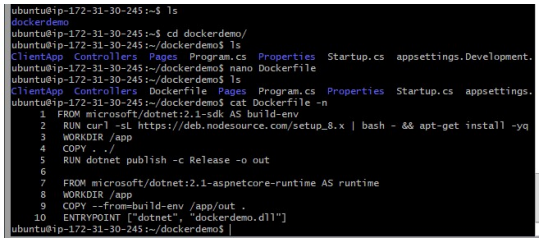
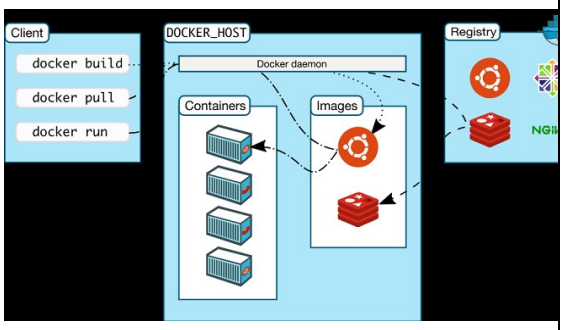
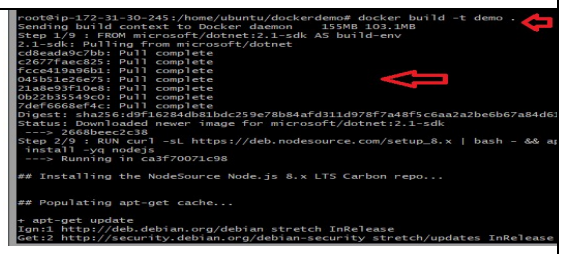
Concepts: (must read)

- Asp.net with Docker intro :
  - https://medium.com/@alcalawilfre/asp-net-core-with-docker-a-beginers-guide-4f490f644a89
  - https://docs.docker.com/engine/examples/dotnetcore/
- Container intro : https://www.docker.com/resources/what-container



	AWS EC2 Docker Setup – Ubuntu only Cloud 9 has docker pre-installed	

1	Docker is not part of the core Ubuntu AMI setup we have been using in past labs. One step install – I
	
	$ sudo snap install docker
	
	$ sudo nano /etc/environment
	
	Suffix ":/snap/bin/" to the PATH
	
	$ docker –version
	`Docker version 18.06.1-ce,XXX`
		
		
		if docker version is not displayed …
		Reboot VM (in sudo mode) - # reboot

	Dockerization of asp.net core app	
1	If this is a fresh VM - Create and run “dotnet new react” as per Lab 1 verify the dotnet app is still functional Once verified – shut down the app in preparation of its dockerization	
2	In project root folder create a new file as below # nano Dockerfile Add the following lines in the file – and replace <name of proj>.dll in last line as relevant in your case FROM microsoft/dotnet:2.1-sdk AS build-env RUN curl -sL https://deb.nodesource.com/setup_8.x | bash - && apt-get install -yq nodejs WORKDIR /app COPY . ./ RUN dotnet publish -c Release -o out FROM microsoft/dotnet:2.1-aspnetcore-runtime AS runtime WORKDIR /app COPY --from=build-env /app/out . ENTRYPOINT ["dotnet", "dotnetdemoapp.dll"]	 Dockerfile sample is attached to this classroom post for convenience. Adjust DLL name in docker file according to your DLL
3	Key Concepts: • Dockerfile contains instruction on how to build the Docker image. Syntax • “docker build” command – converts these instructions into a Docker image • Docker image is a file containing app runtime + codebase (but not full OS – assumes linux) • “docker run” command - creates an in-memory instance of the image (a.k.a the container) that can be accessed over http.	
4	Elevate to Sudo su mode # `docker build -t demo .` # docker images # docker run -p 5000:80 -d demo # docker ps # curl http://127.0.0.1:5000/api/sampledata/weatherforecasts -v Expose the app via public url on port 80 using nginx as usual	 Docker build will trigger the 9 step commands as listed in Dockerfile – download the dotnet SDK + nodejs runtime and then execute dotnet publish

After build – the image does not need the SDK and Nodejs runtime as they result in > 2 GB image size

Docker will download a lightweight dotnet runtime as the final layer (replacing all previous heavyweight layers) and setup the image to execute dotnet <name>.dll when launching the image as a container. That's the purpose of second FROM stmt.

Docker run launches the container and triggers dotnet command. The web app can be exposed outside EC2 via nginx as usual.

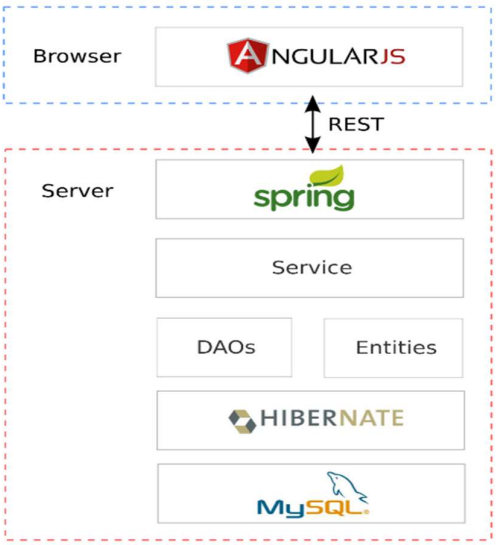
Please read up on Docker concepts in the concept links provided in top of guide – especially the flags used

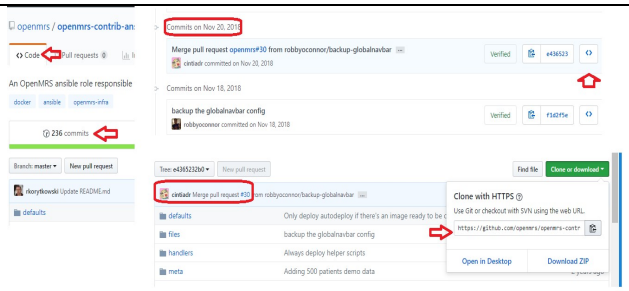
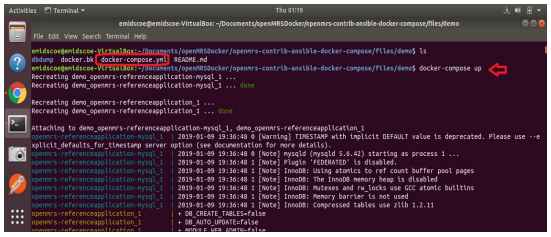
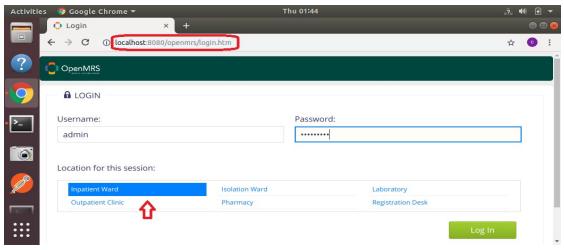
Each docker build process creates images that you can view using $ docker images

When done with Lab - cleanup VM disk space by deleting older images like dotnetSDK using docker rmi command.


```
root@ip-172-31-30-245:/home/ubuntu/dockerdemo# docker images
REPOSITORY          TAG
IMAGE ID            CREATED
SIZE
demo                latest
56d0303961ad        16 seconds ago
257MB
```

In case of error during Docker build - delete broken docker image before building again:
$ docker images
Note down broken image as having <none> in repository
$ docker rmi <imageid>

	Launch third party Dockerized app (capabilities demo)	
1	In this section we will deploy a fully functional Healthcare-oriented Java+Spring web application using Docker with a single command (sudo mode) # docker-compose up The goal is to have you experience first-hand the value addition that Docker bring to the table for rapid deployment and dev ops The Java web application is called OpenMRS - it's a FOSS product that is used to implement EMR in developing countries. https://wiki.openmrs.org/display/docs/Introduction+to+OpenMRS https://atlas.openmrs.org/ Tech stack: https://wiki.openmrs.org/display/docs/Tech+Stack+and+Technologies+Used	In traditional deployments - we would have to install individual components: **JRE + tomcat webserver + MySQL + codeBase** for the above stack on a VM – it would be a multi-step process.  Docker combines all steps into a single config driven step
2	OpenMRS has created 2 Docker images, for App + Db and uploaded them to online repositories. Docker command (docker-compose up) will parse the instruction in docker-compose.yml (same folder) Yml file instructions are to download online Docker images and launch them as containers locally – with relevant environment variables and sequence (orchestration).	

3	In local Ubuntu VM, with Docker installed. Create folder for OpenMRS Visit github page https://github.com/openmrs/openmrs-contrib-ansible-docker-compose Why can't we try it in EC2? This app needs min 2GB RAM for quick docker launch (especially Db). Some participants managed to launch it in Cloud9	
4	navigate in Terminal to git clone folder in m/c openMRS >> openmrs-contrib-ansible-docker-compose >> files >> demo# verify presence of **docker-compose.yml** Execute following command in super user mode $ sudo su # `docker-compose up` This will download all the dockerized pieces of the OpenMRS tech stack – each pre-configured. The OpenMRS app will now be available in the browser with 5000 demo patient encounter records pre-populated.	.  It may take upto 5 minutes  Login info : **Admin / Admin123**
5	Post login – start with patient search Select a record to view the patient demographic info Select a visit hyperlink – to see visit details