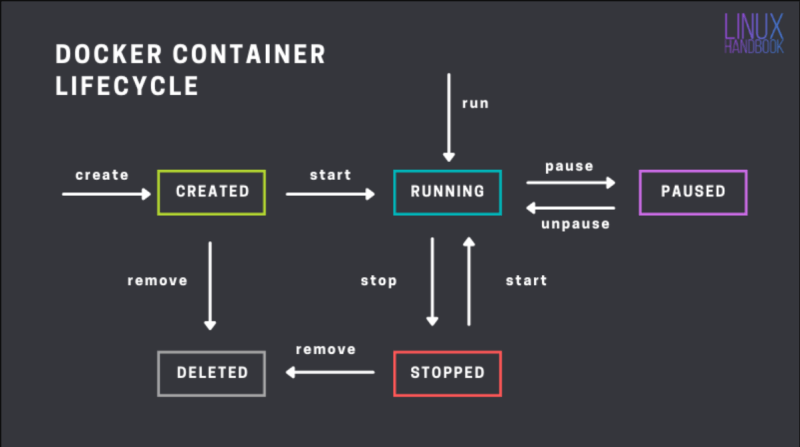
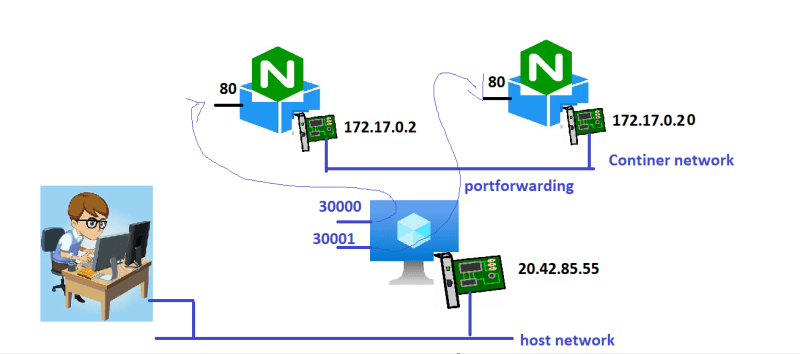
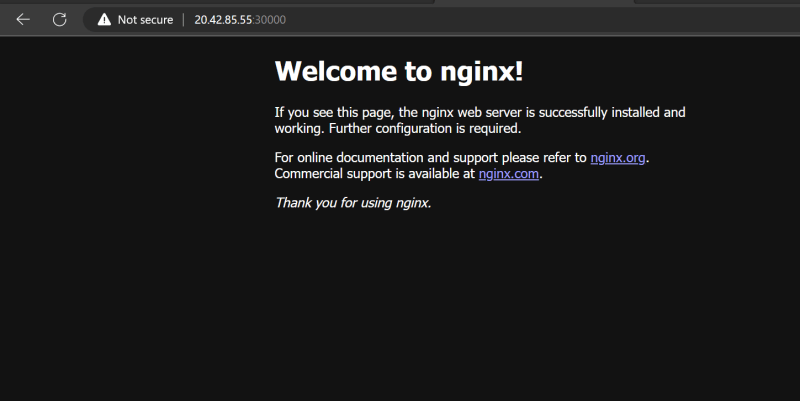
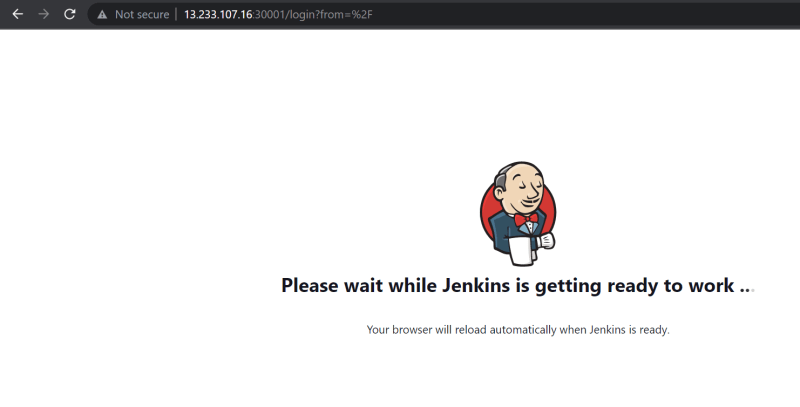
**Docker container lifecycle**

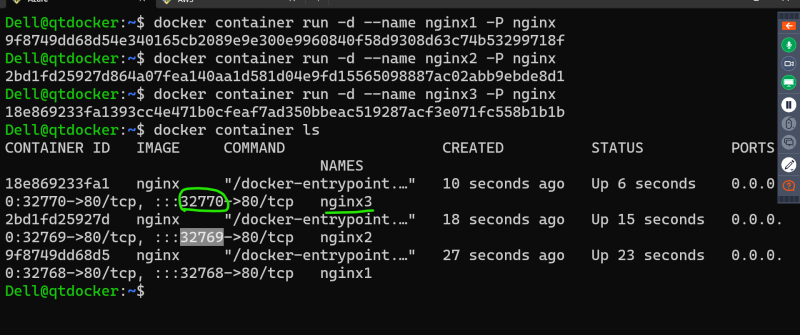
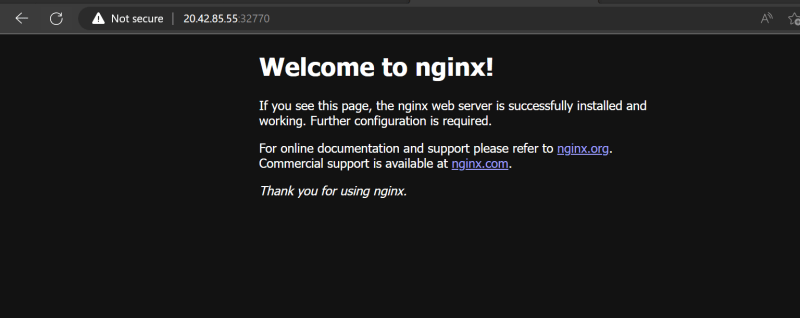
* Docker lifecycle states
  + Created
  + Running
  + Paused
  + Stopped
  + Deleted



Accessing the applications inside docker containers

* From now the machine where we have installed docker will referred as host and the docker container will be referred as container
* We have access to host network & as of now containers are created in private container network, so to access applications inside containers we use port-forwarding  
  
* command: docker container run -d -p <host-port>:<container-port> <image>
* Create a nginx container and expose on port 30000 docker container run -d -p 30000:80 --name nginx1 nginx  
  
* Create a jenkins container & expose 8080 port on 30001 port of host docker container run -d -p 30001:8080 --name jenkins1 jenkins/jenkins



* To assing any random free port on host to container port docker container run -d -P image
* Lets create 3 nginx containers  
    
  

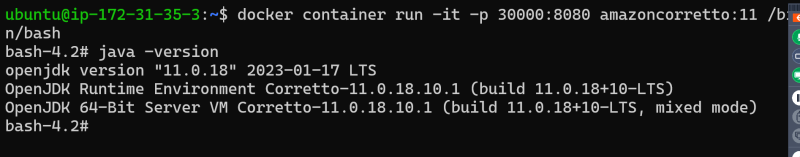
**Exercise**

1. Install docker on a linux vm
2. Run 1 httpd containers (apache container) which runs on 80 port
3. try accessing any application
4. stop the containers
5. try accessing
6. start the continers and access this should work
7. pause the containers, access the application
8. unpause the containers, access the application
9. delete the container

**Containerizing spring petclinic**

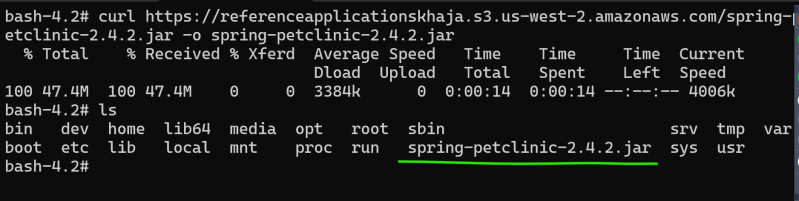
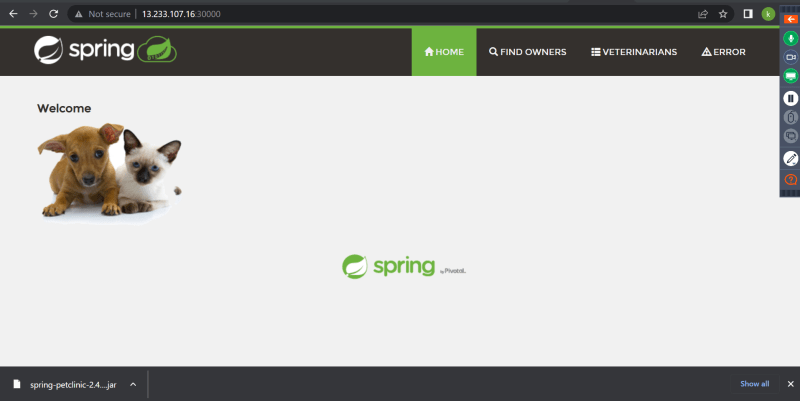
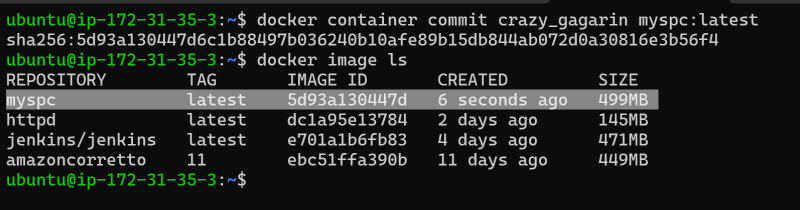
* I have spring petclinic version 2.4.2 which requires java 11 and runs on port 8080
* to start application java -jar spring-petclinic-2.4.2.jar
* What is required:
  + jdk 11
  + jar file
* How to access application
  + http over port 8080
* Lets start the amazoncorretoo based container with port 8080 exposed <https://hub.docker.com/_/amazoncorretto>

docker container run -it -p 30000:8080 amazoncorretto:11 /bin/bash

  
\* now lets download the spring petclinic <https://referenceapplicationskhaja.s3.us-west-2.amazonaws.com/spring-petclinic-2.4.2.jar>

curl https://referenceapplicationskhaja.s3.us-west-2.amazonaws.com/spring-petclinic-2.4.2.jar -o spring-petclinic-2.4.2.jar

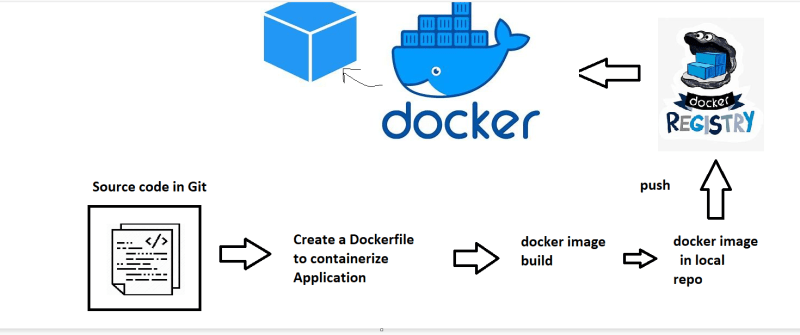
ls

  
\* Run the application java -jar spring-petclinic-2.4.2.jar  
  
\* Now to create a image from a running container, lets login into linux vm, so lets use docker container commit  
  
\* remove all the containers and run the myspc image based container

docker container run -d -p 30001:8080 --name spc1 myspc:latest java -jar spring-petclinic-2.4.2.jar

* This is not a useful approach as we are creating images manually
* DOcker has a better way i.e. Dockerfile

**Dockerfile based Image building**

* Workflow  
  
* Dockerfile is a text file with instructions <https://docs.docker.com/engine/reference/builder/>
* The basic syntax INSTRUCTION arguments
* In Docker we have concept of base image i.e. to run your application using some existing image
* We can use a base image called as scratch which has nothing in it
* In majority of the cases we take what is required to run our application as base image.

**Basic Instructions**

* FROM: <https://docs.docker.com/engine/reference/builder/#from> for official docs. use tag all the time (donot use latest)
* RUN: The commands to be executed while building the image to install/configure your appliation <https://docs.docker.com/engine/reference/builder/#run>
* CMD: This command will be executed while starting the container. <https://docs.docker.com/engine/reference/builder/#cmd> for official docs
* EXPOSE: This adds ports to be exposed while starting the container <https://docs.docker.com/engine/reference/builder/#expose> for official docs

**Springpetclinic Dockerfile**

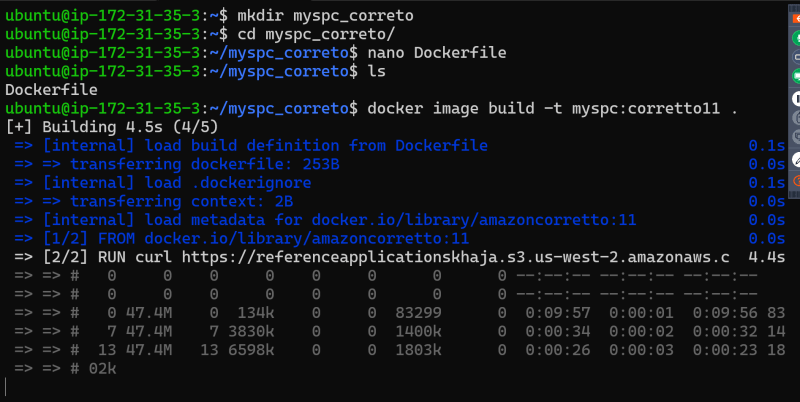
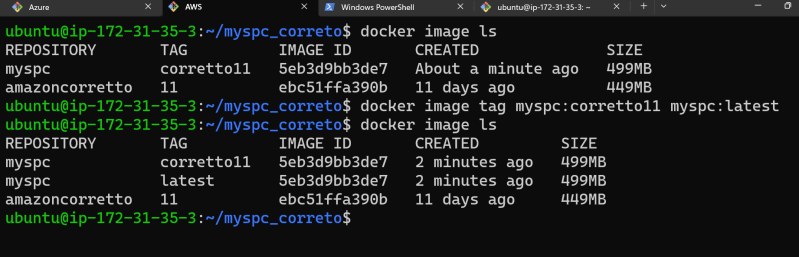
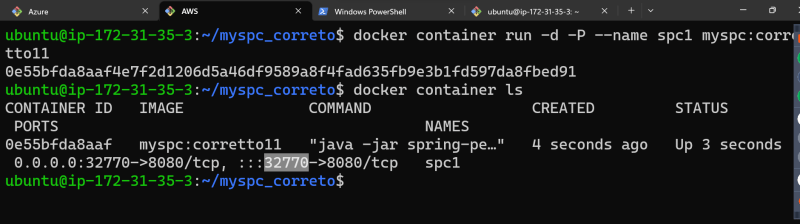
* Lets do two ways
  + use any image with java11 already as base image amazoncorretto:11
  + use any image with slim os as base image alpine:3
* Dockerfile- based on amazoncorreto:11

FROM amazoncorretto:11

RUN curl https://referenceapplicationskhaja.s3.us-west-2.amazonaws.com/spring-petclinic-2.4.2.jar -o spring-petclinic-2.4.2.jar

EXPOSE 8080

CMD ["java", "-jar", "spring-petclinic-2.4.2.jar"]

* Lets build the image based on amazoncorreto  
    
  
* Now lets create a container docker container run -d -P --name spc1 myspc:corretto11  
  
* Approach 2: Start from some os

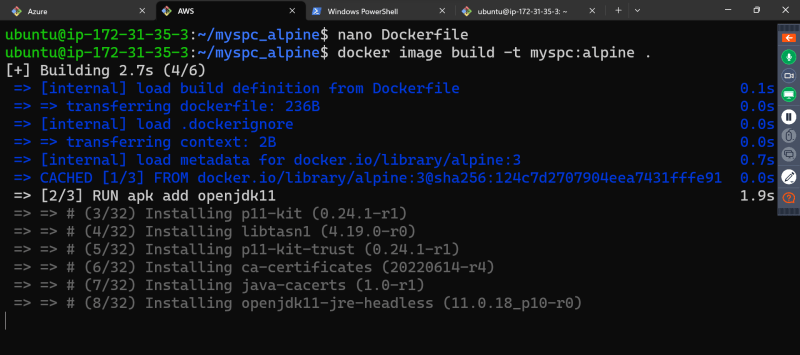
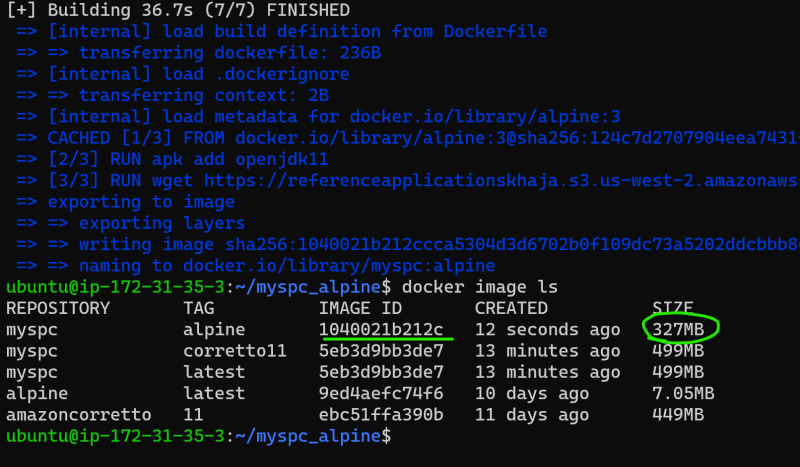
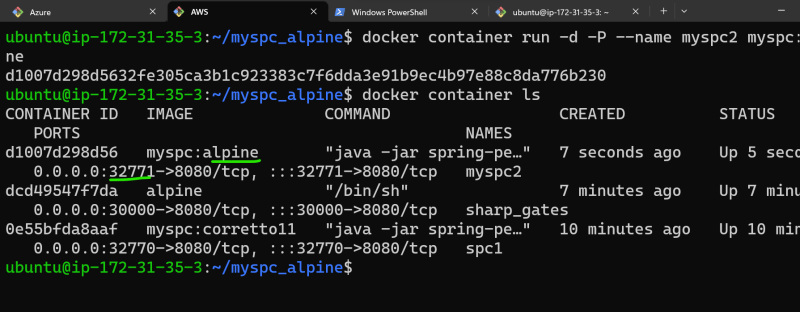
FROM alpine:3

RUN apk add openjdk11

RUN wget https://referenceapplicationskhaja.s3.us-west-2.amazonaws.com/spring-petclinic-2.4.2.jar

EXPOSE 8080

CMD ["java", "-jar", "spring-petclinic-2.4.2.jar"]

* Build the image  
    
  
* Lets run the container docker container run -d -P --name myspc2 myspc:alpine  
  

**Immutable Infrastructure**

* Any infra changes will not be done on infra directly rather we create some infra as code option and change the configuration.