**Docker Compose**

* Docker compose lets you define and run multi-container docker applications
* Using Docker compose, to run multi-container app on single shot
* You will have applications that interact with a database and depend on other services. So essentially, you’ll have multiple containers that interact with each other
* Docker compose lets you deploy and manage apps involving multiple containers
* Docker compose runs all the containers on a single host by default

If you want to deploy multiple applications like one from .net and another from java.

here both application from different platform IIS(.net) and JRE(java).

**Docker file**

A Docker file is a text configuration file. It describes step-by-step instructions of all the commands you need to run to assemble a Docker Image.

Docker file is a text configuration file that Docker reads in from top to bottom. It contains a bunch of instructions which informs Docker HOW the Docker image should get built

The act of running a Docker image creates a Docker container

* A Docker file is a recipe for creating Docker images
* A Docker image gets built by running a Docker command (which uses that Docker file)
* A Docker container is a running instance of a Docker image

**Docker Registry, Repository, Hub**

Docker registry is service that used to store docker images. This will allow docker user to push and pull the docker images with authenticated permission

Docker Hub is one of the docker registry

Under docker registry, we can create multiple repository to store docker images

**Steps to be follow**

1. Create Ubuntu server and login using putty
2. Install docker in Ubuntu
3. Prepare spring boot application
4. build spring boot application as war and deploy as docker image using docker
5. build spring boot application as docker image using docker compose

**Issues faced**

1. Access denied

**Sudo su** **– to give admin permission**

1. Working directory

**WORKDIR – we should give the path of war file**

1. Rename or remove container name

**We need to change the application name in build command**

1. Port issue

**We need to change the port**

**Docker commands**

# **Run the docker file**

docker run --name firstspringapplication -p 8080:8080 -d "firstspringapplication:latest"

* "firstspringapplication:latest" – image name
* “firstspringapplication” – container name

#**Build the docker file**

docker build --tag="firstspringapplication" .

#**to view list of containers**

docker container ls -a

#**remove container**

docker container rm container ID

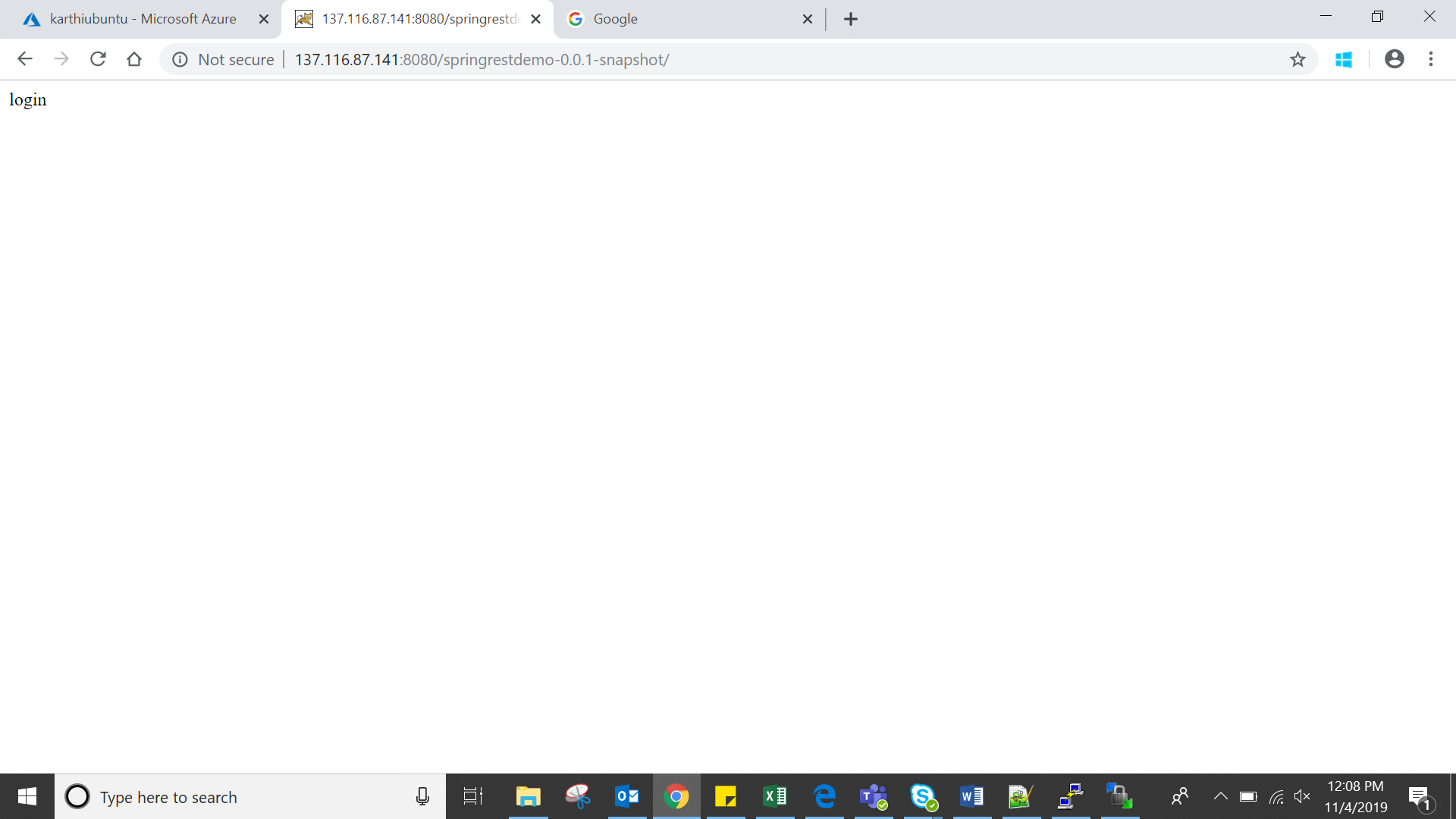
#**view the application which is deployed in docker**

<http://137.116.87.141:8080/springrestdemo-0.0.1-snapshot/>

137.116.87.141 – Ubuntu server IP address

8080 – Expose port

springrestdemo-0.0.1-snapshot – war name



**docker file**

# **Pull base image – to setup the infra for application like which environments**

From tomcat:8-jre8

# **Maintainer**

MAINTAINER sampleapp

#**working directory – to avoid the default path which taken by docker. Workdir is used to set path for war file**

WORKDIR /home/adminuser/springbootapp/

# **Copy to images tomcat path – to copy from the war file, which is in Ubuntu server (already mentioned in workdir) to tomcat application path**

COPY springrestdemo-0.0.1-snapshot.war /usr/local/tomcat/webapps/springrestdemo-0.0.1-snapshot.war

#**Expose the port**

EXPOSE 8080

**Docker compose**

# **to install docker compose**

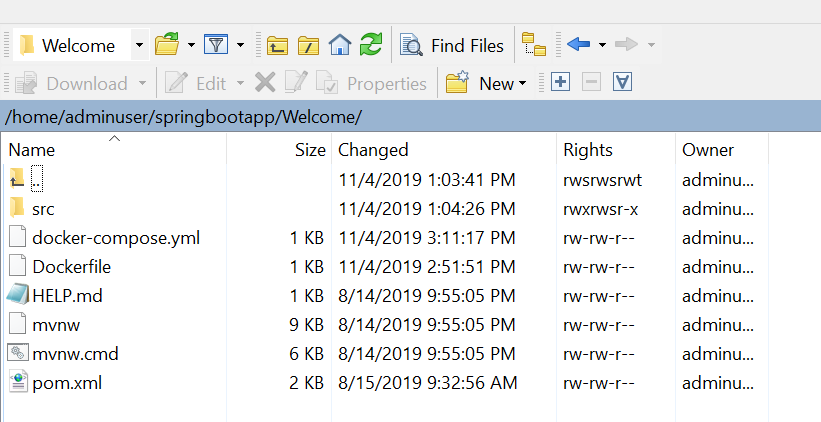
sudo curl -L "https://github.com/docker/compose/releases/download/1.24.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

**To check the docker compose installed**

docker-compose --version



**Docker file and YAML file**



**Docker file commands**

FROM maven:3.5.2-jdk-8-alpine AS MAVEN\_TOOL\_CHAIN

COPY pom.xml /tmp/

COPY src /tmp/src/

WORKDIR /tmp/

RUN mvn package

# Pull base image

FROM tomcat:9.0-jre8-alpine

# Maintainer

MAINTAINER sampleapp

# Copy to images tomcat path

COPY --from=MAVEN\_TOOL\_CHAIN /tmp/target/Welcome-0.0.1-SNAPSHOT.war $CATALINA\_HOME/webapps/Welcome-0.0.1-SNAPSHOT.war

EXPOSE 8082

**YAML file commands**

version: '3'

services:

web:

build: .

ports:

- "8082:8082"

redis:

image: "welcome\_web:latest"

**Docker compose commands**

**Docker-compose up** – it will run the all the application in container

**Docker-compose down** – incase of any error like already running in same ports, change the application name we need to change in YAML file and down the docker-compose using this command

**Docker-compose run hola env** - command allows you to run one-off commands for your services. Example : YAML web, redis, hola like

**Docker-compose ps** – see what is currently running

**Docker run –d –p 8080:8080 firstsampleapp:latest** – run the docker images

docker build --tag="firstsampleapp" . – build the docker images

**Docker images push & pull into Docker Hub**

1. First, we need to tag the image

**docker tag 12b62c490e69 21121990/warapp:war**

**12b62c490e69 –** docker image id

**21121990 –** docker hub username

**Warapp –** docker image name

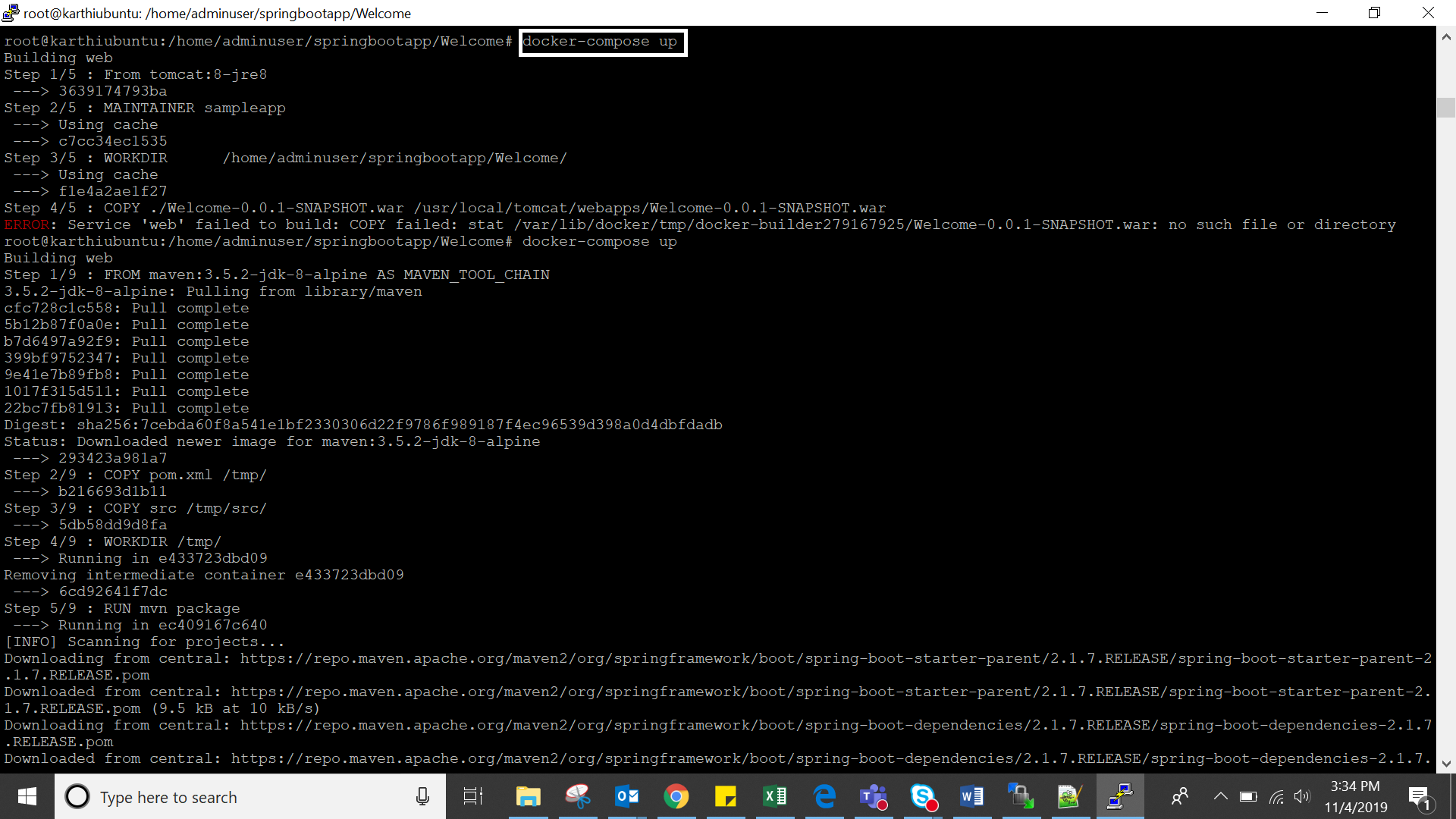
**War –** tag name for ref

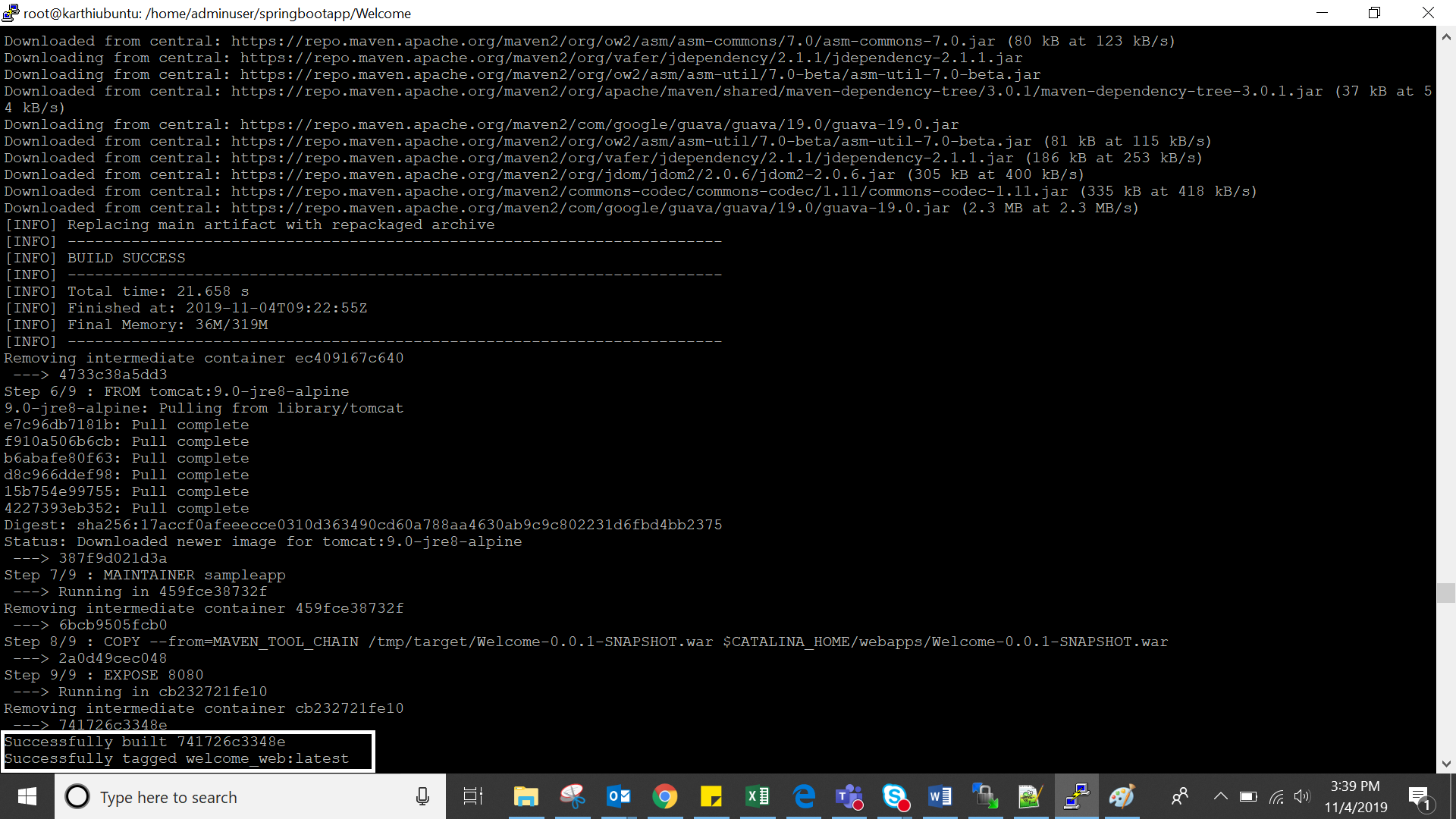
1. Push into docker hub

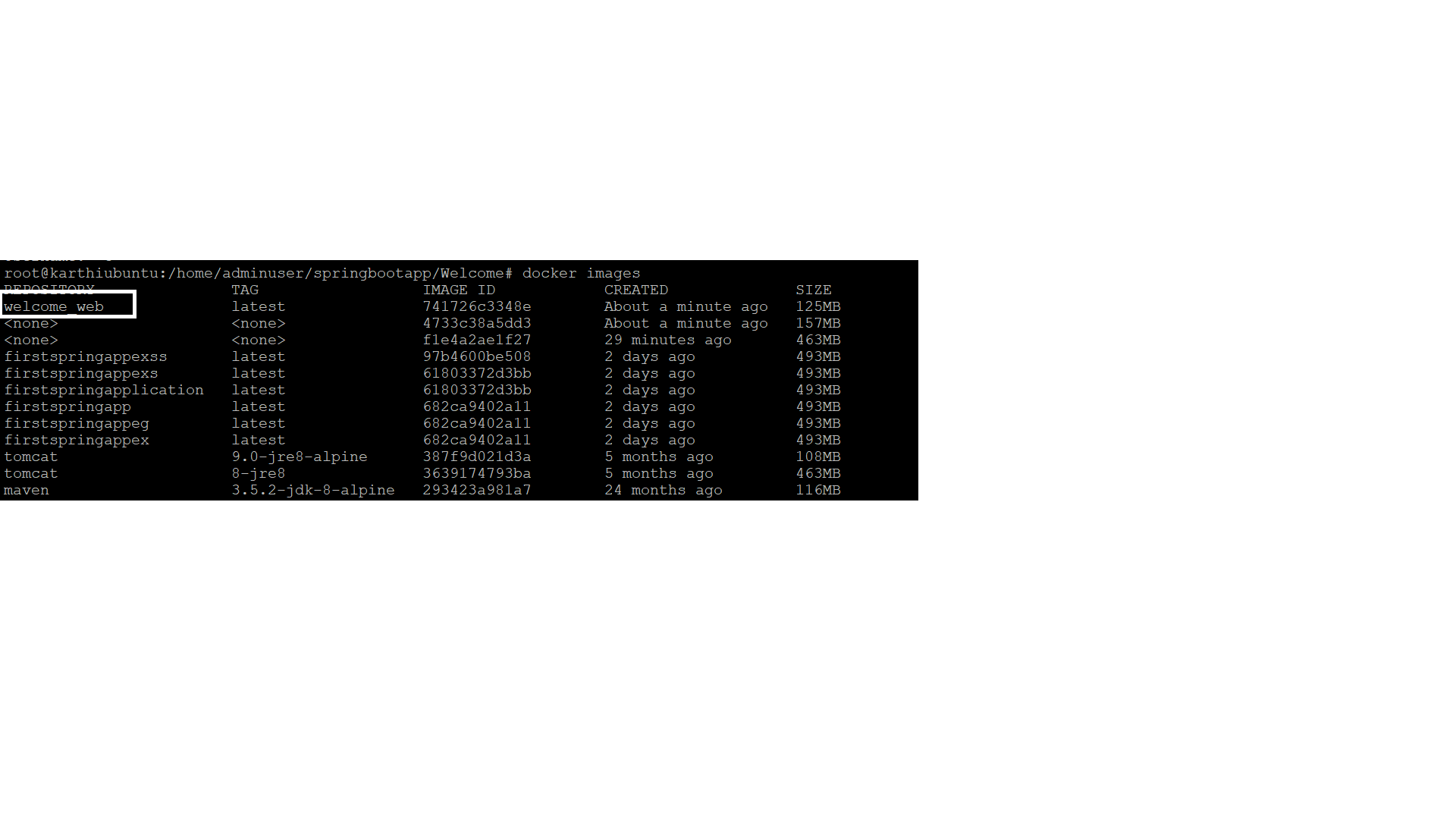
**docker push 21121990/warapp**

1. Pull images from docker hub

**docker pull 21121990/warapp:war**

****

****

****