Dockerfile Instructions

FROM	To use current Official Repositories as the basis for your image Dockerfile must have FROM as its first instruction tag or digest value is optional	FROM <image> FROM<image>:<tag> FROM <image>@<digest>	FROM Ubuntu FROM Ubuntu:14.04
RUN	RUN instruction will execute any commands in a new layer on top of the current image and commit the results. The resulting committed image will be used for the next step in the Dockerfile.	RUN <command> (shell form, the command is run in a shell, which by default is /bin/sh -c on Linux or cmd /S /C on Windows) RUN ["executable", "param1", "param2"] (exec form)	RUN apt-get update && apt-get install openjdk-8-jre-headless -y RUN apt-get update && apt-get install -y \ aufs-tools \ automake \ build-essential \ curl \ dpkg-sig \ && rm -rf /var/lib/apt/lists/*
LABEL	Labels are a mechanism for applying metadata to Docker objects, including: • Images • Containers • Local daemons • Volumes • Networks • Swarm nodes • Swarm services A label is a key-value pair, stored as a string. You can specify multiple labels for an object, but each key-value pair must be unique within an object.	LABEL com.example.version="0.0.1-beta" LABEL vendor="QShore" LABEL com.example.release-date="2015-02-12"	
EXPOSE	The EXPOSE instruction indicates the ports on which a container will listen for connections. Consequently, you should use the common, traditional port for your application. For example, an image containing the Apache web server would use EXPOSE 80, while an image containing MongoDB would use EXPOSE 27017 and so on	EXPOSE <port> [<port>...]	#Expose Tomcat EXPOSE 8080
COPY	copies new files or directories from <src> and adds them to the filesystem of the container at the path <dest>.	COPY <src>... <dest> COPY ["<src>",... "<dest>"] (this form is required for paths containing whitespace)	COPY test relativeDir/ # adds "test" to `WORKDIR`/relativeDir/ COPY test /absoluteDir/ # adds "test" to /absoluteDir/
ENTRYPOINT	to set the image's main command, allowing that image to be run as though it was that command (and then use CMD as the default flags).	ENTRYPOINT ["s3cmd"] CMD ["--help"]	docker run s3cmd

Dockerfile & Docker registry exercise

Step 1	Create a folder "Mydocker"	mkdir mydocker
	Change to the folder	cd mydocker
Step 2	Create "tomcat-users.xml" file and copy the below content into it	nano tomcat-users.xml
	tomcat-users.xml <?xml version="1.0" encoding="utf-8"?> <tomcat-users> <role rolename="manager-gui"/> <role rolename="manager-status"/> <role rolename="manager-script"/> <role rolename="manager-jmx"/> <role rolename="admin-gui"/> <role rolename="admin-script"/> <user username="admin" password="admin" roles="manager-gui, manager-script, admin-gui, admin-script"/> </tomcat-users>	
Step 3	Create dockerfile with nano or vim editor in the same folder	nano dockerfile
	Copy the below set of instructions into the dockrefile	
	FROM ubuntu:16.04 RUN apt-get update && apt-get -y upgrade RUN apt-get -y install software-properties-common RUN add-apt-repository ppa:webupd8team/java -y RUN apt-get -y update # Accept the license RUN echo "oracle-java8-installer shared/accepted-oracle-license-v1-1 boolean true" | debconf-set-selections RUN apt-get install oracle-java8-installer -y RUN apt-get install oracle-java8-set-default -y # Nano Installation RUN apt-get install nano -y RUN apt-get update -y # Here comes the tomcat installation RUN apt-get install tomcat8 -y RUN apt-get install tomcat8-docs tomcat8-examples tomcat8-admin -y RUN echo "JAVA_HOME=/usr/lib/jvm/java-8-oracle" >> /etc/default/tomcat8 #Copy tomcat-users.xml to the new image COPY tomcat-users.xml /var/lib/tomcat8/conf/ # Expose the default tomcat port	

	EXPOSE 8080 # Start the tomcat (and leave it hanging) CMD service tomcat8 start && tail -f /var/lib/tomcat8/logs/catalina.out	
Step 4	Build the image Make sure that you are in mydocker folder and below files are present dockerfile tomcat-users.xml	docker build -t qshore/tomcat:1.0 .
	Create container	docker run -d --rm --name="cat1" -p 8080:8080 qshore/tomcat:1.0
Step 5	Create docker registry…	docker pull registry
Step 6	Run docker registry	docker run -d --rm --name="My Dock Store" -p 5000:5000 registry
Step 7	Create a tag to the image created by you in step 4	docker tag qshore/tomcat:1.0 localhost:5000/tomcat
Step 8	Push your image to your docker registry	docker push localhost:5000/tomcat
Step 9	Remove your local image. Make sure it's container is stopped and killed.	docker rmi qshore/tomcat:1.0
Step 10	Pull your image from your docker registry	docker pull localhost:5000/tomcat