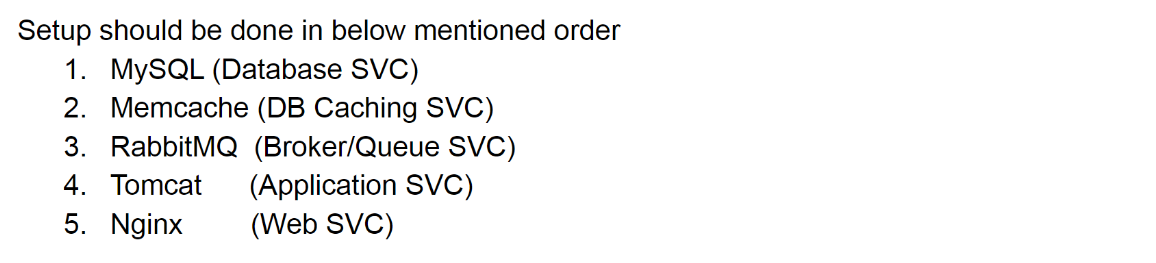


STEPS:

1. Using the resource in the project (C:\gitrepos\cloudcodesandsecurity\DevOps\Projects\Vprofile\_Project\_Setup\Manual\_provisioning)



1. Create a Docker Engine EC2 instance follow steps in the doc: Setup of Docker
2. open GitBash and don’t SSH yet
3. cd c:/
4. mkdir docker-engine
5. cd docker-engine/
6. vagrant init ubuntu/bionic64
7. vim Vagrantfile

Uncomment: config.vm.network “private\_network”, ip: “set and note your IP” (e.g. 192.168.33.12)

:wq

1. vagrant up
2. vagrant ssh
3. don’t do sudo –i yet
4. We need to install Docker Engine in the Vagrant

* Follow the Documentation on (<https://docs.docker.com/engine/install/ubuntu/> ),
* Updating the apt package index and installing packages to allow apt to use a repository:
* sudo apt-get update
* sudo apt-get install \

apt-transport-https \

ca-certificates \

curl \

gnupg-agent \

software-properties-common -y

* Adding Docker’s official GPG key:

sudo install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

sudo chmod a+r /etc/apt/keyrings/docker.gpg

* Setup the Repository :

echo \

"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \

"$(. /etc/os-release && echo "$VERSION\_CODENAME")" stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

* Install Docker Engine

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y

* Verify that the Docker Engine installation is successful by running the hello-world image:

systemctl status docker

docker - -version

1. whoami (To get the user )
2. Add the User to the Docker group to have access
3. sudo usermod –a –G Docker (username .e.g. vagrant)
4. exit and login again
5. vagrant ssh
6. id (To check if Vagrant is part of Docker group)
7. docker images (To check the list of images)
8. sudo systemctl status docker
9. We need to build Dockerfile to host the services, in step 1 above
10. Use Intellij IDEA on the system and save it into same folder where the vagrantfile is:
11. Open the Intellij, click on file ….. New ….. Project from Version Control

URL = https://github.com/devopshydclub/vprofile-project.git (The URL of the github project to clone)

Directory = Select the Directory where the Vagrantfile is (Docker-engine) and create a folder named vprofile-project

click on clone

1. Click on Master at the extreme right bottom, to change the branch to origin/docker by clicking on checkout
2. On the left pane, in Docker-files we would write our Dockerfile and save it
3. Go to hub.docker.com and login
4. click on organization tab above and create a free 1 -3 team member
5. Create a Repository inside the Organization
6. click on the first drop down arrow and select the organization name you created
7. Name = vprofileapp (These Repo will be for Tomcat)
8. check public and click on create

* Create another Repository inside the Organization
* click on the first drop down arrow and select the organization name you created
* Name = vprofileweb (These Repo will be for Nginx)
* check public and click on create
* Create another Repository inside the Organization
* click on the first drop down arrow and select the organization name you created
* Name = vprofiledb (These Repo will be for MySQL)
* check public and click on create

1. Return back to the Intellij, right click on the folder (Docker-files)
2. click on New, click on Directory
3. Give the Directory a name (app)
4. Right click on the App directory, click on New and click on file
5. Give it the name = Dockerfile
6. click on Add , to accept the Git request if any
7. Go to hub.docker.com and search for Tomcat official with Tag = 8-jre11

FROM tomcat:8-jre11  
RUN rm -rf /usr/local/tomcat/webapps/\*  
COPY target/vprofile-v2.war /usr/local/tomcat/webapps/ROOT.war  
  
EXPOSE 8080  
CMD ["catalina.sh", "run"]  
WORKDIR /usr/local/tomcat/  
VOLUME /usr/local/tomcat/webapps/

1. Return back to the Intellij, right click on the folder (Docker-files)
2. click on New, click on Directory
3. Give the Directory a name (db)
4. Right click on the db directory, click on New and click on file
5. Give it the name = Dockerfile
6. click on Add , to accept the Git request if any
7. Go to hub.docker.com and search for MySQL official with Tag = 5.7.25

FROM mysql:5.7.25  
LABEL "Project"="Vprofile"  
LABEL "Author"="Ola-Gabriel"  
  
ENV *MYSQL\_ROOT\_PASSWORD*="vprodbpass"  
ENV *MYSQL\_DATABASE*="accounts"  
  
  
ADD db\_backup.sql docker-entrypoint-initdb.d/db\_backup.sql

1. Return back to the Intellij, right click on the folder (Docker-files)
2. click on New, click on Directory
3. Give the Directory a name (web)
4. Right click on the web directory, click on New and click on file
5. Give it the name = Dockerfile
6. click on Add , to accept the Git request if any
7. Go to hub.docker.com and search for Nginx official with Tag = Default is ok, don’t specify

FROM nginx  
LABEL "Project"="Vprofile"  
LABEL "Author"="Ola-Gabriel"  
  
RUN rm -rf /etc/nginx/conf.d/default.conf  
COPY nginvproapp.conf /etc/nginx/conf.d/vproapp.conf

1. Go to the GitBash, and create the Target/vprofile:V2.war (Artifact) for the Tomcat service
2. docker - -version
3. cd /vagrant/
4. ls
5. cd vprofile-project/
6. ls
7. cd Docker-files/
8. ls
9. ls –R
10. cd ../..
11. cd vagrant/
12. cd vprofile-project/
13. ls
14. The pom.xml is the artifact, we need jdk and Maven to build
15. sudo apt install openjdk-8-jdk –y && sudo apt install maven –y
16. We need to fillin the details of the Backend services to connect to the Db
17. check the application.properties file in these (C:\gitrepos\cloudcodesandsecurity\DevOps\ Docker\Containerizing Project\vprofile-project\src\main\resources) for configuration details
18. pwd
19. ls
20. mvn install
21. ls
22. ls target/
23. ls Docker-files/app/
24. cp –r target Docker-files/app/
25. ls
26. cd Docker-files/app/
27. ls
28. docker build –t vprofile/vprofileapp:V1 **.** (These command builds the Docker image and pushes to the repository)
29. docker images
30. cd ../
31. ls
32. cd db/
33. docker build –t vprofile/vprofiledb:V1 **.** (These command builds the Docker image and pushes to the repository)
34. docker images
35. cd ../
36. cd web/
37. docker build –t vprofile/vprofileweb:V1 **.** (These command builds the Docker image and pushes to the repository)
38. docker images
39. We need 2 other Images (Rabbitmq & Memcache), we don’t need to customize them. Just pull them
40. cd web/
41. docker pull memcached
42. docker pull rabbitmq
43. We need to install Docker compose to run all these containers and connect all (the 5 services) together
44. Use the documentation: (docs.docker.com/compose/install/ ) (video timestamp is 51.32mins)
45. Install Docker Desktop and it comes with compose. Use the documentation as a strong guide
46. the path should be: /usr/local/bin/docker-compose
47. sudo chmod +x /usr/local/bin/docker-compose
48. docker-compose - -version
49. Use intellij to write the Docker compose YAML file, save as: docker-compose.yml
50. version: '3'  
    services:  
     vprodb:  
     image: vprofile/vprofiledb:V1  
     ports:  
     - "3306:3306"  
     volumes:  
     - vprodbdata:/var/lib/mysql  
     environment:  
     - MYSQL\_ROOT\_PASSWORD=vprodbpass  
      
     vprocache01:  
     image: memcached  
     ports:  
     - "11211:11211"  
      
     vpromq01:  
     image: rabbitmq  
     ports:  
     - "15672:15672"  
     environment:  
     - RABBITMQ\_DEFAULT\_USER=guest  
     - RABBITMQ\_DEFAULT\_PASS=guest  
      
     vproapp:  
     image: vprofile/vprofileapp:V1  
     ports:  
     - "8080:8080"  
     volumes:   
     - vproappdata:/usr/local/tomcat/webapps  
      
     vproweb:  
     image: vprofile/vprofileweb:V1  
     ports:  
     - "80:80"  
    volumes:  
     vprodbdata: {}  
     vproappdata: {}
51. Go to Docker engine in GitBash
52. cd /vagrant/vprofile-project/
53. mkdir compose
54. mv docker-compose.yml compose/
55. cd compose/
56. ls
57. docker-compose - -help (To view a list of commands)
58. docker-compose up –d
59. docker logs (container name) to see logs of the container
60. Go to your browser: (Use the public IP for your Vagrantfile you specified at the beginning):80
61. If everything is fine in the browser, it is time to push the images to Docker Hub
62. docker login
63. username = account name
64. password = set pass
65. docker push vprofile/vprofileapp:V1 (repository name/project name)
66. Go to hub.docker.com and search for mysql official image