Step by Steps to build the Application on Centos OS 7:

1. **Approach Notes/Initial Analysis:**  **04/07/2019: Started at 8:00 AM**
2. Create as Centos 7 as Host OS.
3. Install the required packages, create a script for that.
4. Now create the dockerfile and necessary other folders for the PHP container with nginx as webserver
5. Create the cronjob for the 1 AM daily to trigger it and send an email.
6. Check the network setting to make sure both the containers are in bridge mode, that is both the containers should be in the same network.
7. Create the separate docker file for the MySQL container.
8. Using SCP, to create a connection between both the containers.
9. Call the connection status API from PHP file in PHP docker to connect to the MySQL container.

**2) Started the Environment build-up/Prerequisites: Time: 8:10 AM**

**Option 1**: Load an Oracle Virtual Machine in your laptop, and start a Centos 7 VM in it, but make sure **it** CPU is X86.

Note: My Laptop had i648 as processor, do a road blocker:

**Option 2:** Create a VPC and an EC2 instance with Centos 7 in AWS. Assign one Public IP to it along with eth0 interface. **Time: 8:25 AM**

**3) Now the Host OS – Centos 7 for the Docker is built up:**

1. This will be a base Centos 7 without updated information.
2. Check for Updated: yum check-updates
3. Create a Shell Script to perform the rest of the task: docker\_start.sh

#!/bin/bash

curl -fsSL https://get.docker.com/ | sh

sleep 50

**Logs of Installation**:

[root@ip-10-0-0-249 centos]# curl -fsSL https://get.docker.com/ | sh

# Executing docker install script, commit: 2f4ae48

From : https://download.docker.com/linux/centos/gpg

**#Start the docker service**

systemctl start docker.service

sleep 20

**#Check the Status, if the docker service has started or not**

systemctl status docker.service

**Logs of Status:**

[root@ip-10-0-0-249 centos]# sudo systemctl status docker

● docker.service - Docker Application Container Engine

Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; vendor preset: disabled)

Active: active (running) since Tue 2019-07-02 03:16:38 UTC; 1min 1s ago

Docs: https://docs.docker.com

Main PID: 14829 (dockerd)

Tasks: 8

sleep 20

# Create a symlink to make sure docker comes back, even after the reboot of Host OS

systemctl enable docker

sleep 20

#Packages for compose and other related upgrades

yum install epel-release

sleep 10

yum install -y python-pip

sleep 10

pip install docker-compose

sleep 10

pip install –upgrade

sleep 10

yum upgrade python\*

sleep 10 **Time: 9:13AM**

**# Package to install cron job, which was not installed by default in my centos os**

yum install cronie

4) **Create a docker-compose file:**

The docker-compose.yml file is our configuration and interface for using Docker Compose. Docker Compose, we are specifying how our application is structured and each item linked together. We can start and run the servers using docker compose.

1. Define your app’s environment with a Dockerfile so it can be reproduced anywhere.
2. Define the services that make up your app in docker-compose.yml so they can be run together in an isolated environment.

Note: change the port information in the Docker compose file to specify the port being used by your service which building as well. For me 80 was not allowed by AWS, so used a different port.

docker-compose build

sleep 10

echo "Docker image has been build"

docker-compose up -d

sleep 10

echo "Docker compose is running" **Time: 9:35AM**

**5) Let us Create a DockerFile:**

# Install the nginx server

RUN yum install -y epel-release && yum install -y nginx

# Start the nginx server as default webserver

RUN systemctl start nginx

#Enable the nginx server at Docker start time

RUN systemctl enable nginx

#Copy the config of nginx

COPY nginx.conf /etc/nginx/nginx.conf

#Install PHP7 and then update it

RUN yum install -y http://rpms.remirepo.net/enterprise/remi-release-7.rpm && \

yum install -y yum-utils && \

yum-config-manager --enable remi-php72 && \

yum -y update **Time: 9:50AM**

**# Install PHP FMP**

RUN yum install -y php-fpm php-cli php-bcmath php-dba php-gd php-intl php-mbstring php-mysql

COPY conf/php.ini /etc/php/7.1/fpm/conf.d/40-custom.ini

# **Install wkhtmltopdf 0.12.0 is now obsolete now as per official information, so I went with standard one that is 12.4. There might be a way to build the RPM again for the 12.0 build, but didn’t had that much time to R&D on it.**

yum -y install zlib fontconfig freetype libX11 libXext libXrender **Time : 10:10**

**# To Initiate Cron Job: You need to add the crontab file:**

COPY conf/crontab /etc/crontab

SHELL=/bin/bash

PATH=/sbin:/bin:/usr/sbin:/usr/bin

MAILTO=perminder.mehta@gmail.com

1. 1 \* \* \* root /root/ELMO\_project/docker\_startup.sh >> /var/log/index.log

6) Different Folders and what it contains:

a) www : contains the information about the index.php file.

b) config : contains information about the config files and if anyone needs to be changed.

c) sites: information about the each site on the container

6) DRY run it: Run the docker images and make sure image is build properly and load the image

[root@ip-10-0-0-249 ELMO\_project]# docker-compose build

……..

Successfully built 998f31a8ba71

Successfully tagged elmo\_project\_php:latest

Run the Docker-Compose:

[root@ip-10-0-0-249 conf]# docker-compose up

Starting elmo\_project\_code\_1 ...

Starting elmo\_project\_code\_1 ... done

elmo\_project\_php\_1 is up-to-date

Recreating elmo\_project\_web\_1 ... done

Attaching to elmo\_project\_mysql\_1, elmo\_project\_code\_1, elmo\_project\_php\_1, elmo\_project\_web\_1

php\_1 | [02-Jul-2019 13:03:20] NOTICE: fpm is running, pid 1

php\_1 | [02-Jul-2019 13:03:20] NOTICE: ready to handle connections

^CGracefully stopping... (press Ctrl+C again to force)

Stopping elmo\_project\_web\_1 ... done

Stopping elmo\_project\_php\_1 ... done

Stopping elmo\_project\_mysql\_1 ... done

**8) Now need to assign IP address to PHP docker container**: **Times: 10:20 AM**

sudo docker network ls

Check the network status if it is running in bridge mode or not. It should run in bridge mode

sudo ip link add eth0 type bridge

sudo ip addr add 192.168.1.0/24 dev eth0 sudo ip link set eth0 up

After creating the docker bridge, you will need to add the following line in /etc/default/docker file:

sudo nano /etc/default/docker

Add the following line at the end of file:

DOCKER\_OPTS=”-b=eth0”

Save and close the file and start docker service:

sudo service docker start

You should see the new bridge interface eth0 by running the following command on host machine:

sudo docker inspect "container ID"

1. **Pulled the default container of mysql: Time: 10:40 AM**

Docker pull mysql

Create a Docker file for that and then run it :

docker run --name elmo\_mysql -e MYSQL\_ROOT\_PASSWORD=mysecretpw -v mysql:/var/lib/mysql -d elmo\_mysql:latest

docker inspect elmo-mysql | grep IPAddress

to get the IP address, use the same IP in the index.php to connect to the database

Need to Update the code in git and then write the steps as well in doc, this is a complete project and spent 3 hours to do all activities in total.