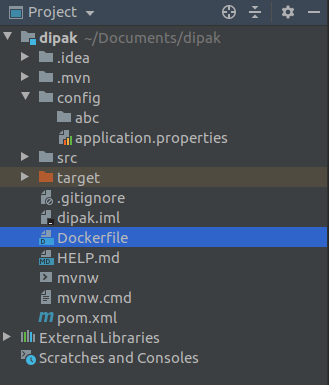
**Run Spring Boot Application in Docker**

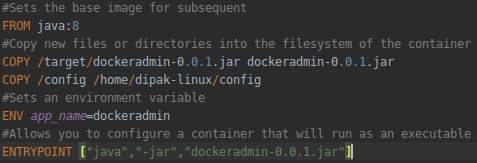
Install java in docker demon

docker pull java:8

create **Dockerfile** in project root



write script in **Dockerfile**



Build image from **Dockerfile**

docker build -t dockeradmin .

**Docker Compose**

**Docker Compose**is used to run multiple containers as a single service. For example, suppose you have two microservice **mic1,mic2 and mongodb**, you could create one file which would start both the containers as a service without the need to start each one separately.

**Docker compose for microservice**

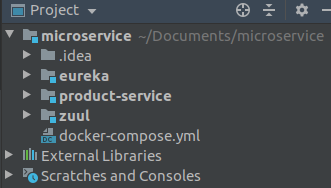
**1st :** Install docker compose from this link <https://docs.docker.com/compose/install/>

I have 2 microservice **mic1,mic2** and **mongodb**. Create **Dockerfile** in each microservice.

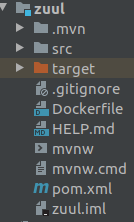
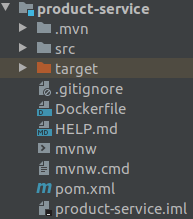
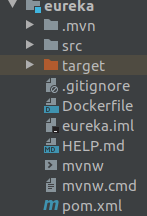
Then create **docker-compose.yml** file. Follow a sample docker-compose file which have in google drive folder

docker-compose run all container as a one singale service.

This is an microservice project and this have a **docker-compose.yml** file



each microservice have Dockerfile



this is docker.compose script

*#set version of docker compose*

version: '2.2'

*#create service*

services:

*#service name*

eureka:

*#container name*

container\_name: eureka

*#set docker file path and file name*

build:

*#path*

context: ./eureka

*#filename*

dockerfile: Dockerfile

*#image name*

image: eureka:latest

*# The container will also always start on daemon startup*

restart: always

*# use syslog driver to manage log of container*

logging:

driver: syslog

options:

*# this is a syslog server address.*

*# If we use elk stack then we use ip and port of logstash*

syslog-address: "tcp://172.17.0.1:5000"

*# This is the format of log. It is optional*

syslog-format: "rfc3164"

*# this is app-name which do identify of each container logs*

*# here we use tag name as container name*

tag: "node-red”

*#set environment variable*

*#Environment variable help you to set spring boot application.properties config at runtime*

*#like - spring.cloud.config.uri: http://configserver:8888*

environment:

SERVER\_NAME: demoserver

*#set port on which run container*

ports:

- "8761:8761"

*# to run a command in docker container*

command: use any command to run in docker container

*#Basically, links is no longer needed because its main purpose, making container reachable by another by adding environment variable, is included implicitly with network. When containers are placed in the same network, they are reachable by each other using their container name and other alias as host*

links:

- db

*#choose network*

networks:

- microservice

*#-------------------------------------#*

*# Don't use container\_name,ports of a #*

*# service which you can do scale up #*

*# and down #*

*#-------------------------------------#*

product-service:

build:

context: ./product-service

dockerfile: Dockerfile

image: product:latest

depends\_on:

- eureka

networks:

- microservice

- mongodb

zuul:

build:

context: ./zuul

dockerfile: Dockerfile

image: zuul:latest

ports:

- "8765:8765"

depends\_on:

- eureka

networks:

- microservice

*#create network*

networks:

*#this network is pre existing network outside of docker compose.*

*#connect external network this way. external network is the network which already exist*

*#external network use to create centralized database container*

mongodb:

*#this use to skip remove external network on 'docker-compose down' command which is outside of docker compose.*

external: true

*#this is the alias name of network*

name: mongodb

*#this network is internal network in docker compose.*

*#this remove on 'docker-compose down' command*

microservice:

name: microservice

**Keep in Mind :**

when run microservice project in docker you must see the link which use in microservice application.properties file like **eureka server url,config server url .** At devlopment time this url structure is like this

eureka.client.serviceUrl.defaultZone = http://localhost:8761/eureka

but when it deploye in docker you must change this url like this

eureka.client.serviceUrl.defaultZone = <http://eureka:8761/eureka>

the **eureka** is the name of docker container name where eureka server is hosted. In docker as container name is the **host name**.

**Create Centralized Container**

which access by any others container. Here mongodb container is centralized container

mongodb

------------------------------

mic1 mic2 mic3   
 ------------------------------



--- 1st check mongodb container is running or not. If not running then run it using ‘**docker run**’ command

--- then create network using this command

**docker network create network-name**

**Ex:** **docker network create mongodb**

--- connect mongodb container with newly created network using this command

**docker network connect network-name container-name**

**Ex:** **docker network connect mongodb my-mongo**

--- then use this network in docker compose. To see **networks** section of **docker-compose.yml** file for details

--- In spring boot application.properties file must use hostname of database url the name of mongodb container.

**Ex:** **spring.data.mongodb.host=my-mongo**