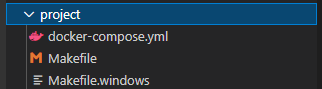
Things need to take care while coding in go

1. Use function name in first letter in capital to make it global function
2. SSL and TLS both are same. TLS is the updated version of SSL



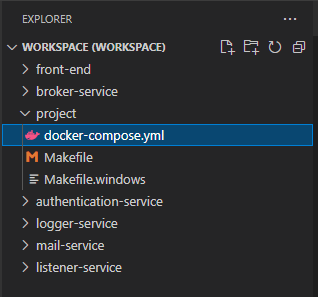
Basic things need to take care while using docker

Install docker from

<https://docs.docker.com/engine/install/>

Do the necessary registration and add credentials in the PC app

Create the docker-compose.yml file inside the project directory as below



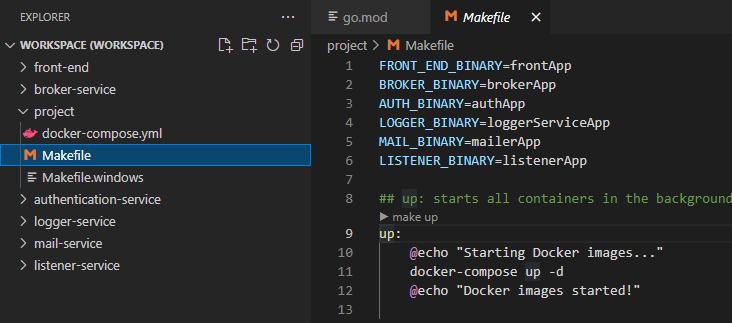
What is docker-compose.yml file will do?

1. It will list down all the services, database used, port used, protocol used in creating the application
2. It will act as configuration file for docker application

Create the make file

Make file will reduce the efforts of the putting commands manually needed while compiling

It will merge long command into short so that will save time



How to use docker?

Used docker swarm services from docker

Docker swarm is the container orchestration tool

Docker swarm and docker classic swarm are 2 different parameters, specially docker classic swarm is the end of life but docker swarm is not.

Docker swarm is used when you have a instance and you need to ntegrate that to large nodes, for example you services into the larger server for scalability docker swarm wil make this very easy for you.

More detail of swarm is given here

<https://docs.docker.com/engine/swarm/>

Before deploying the docker services you need to push your site to somewhere. and after that you will pull that on server easily

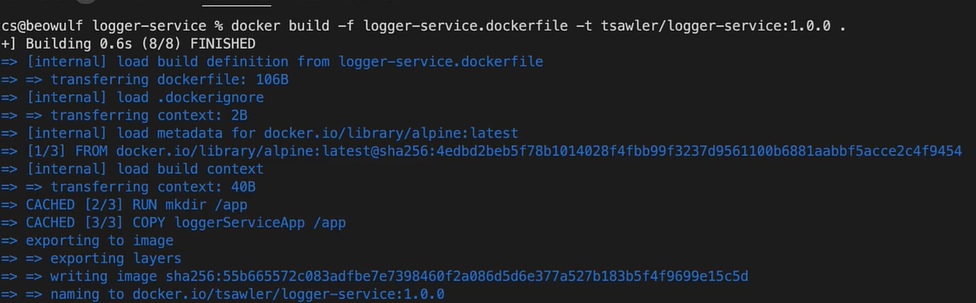
Go to hub.docker.com

and register your self

Command to build the docker file is

docker build -f logger-service.dockerfile -t name/service:version .

docker build -f logger-service.dockerfile -t yagnik/logger-service:1.0.0 .



Now it is time to push the files

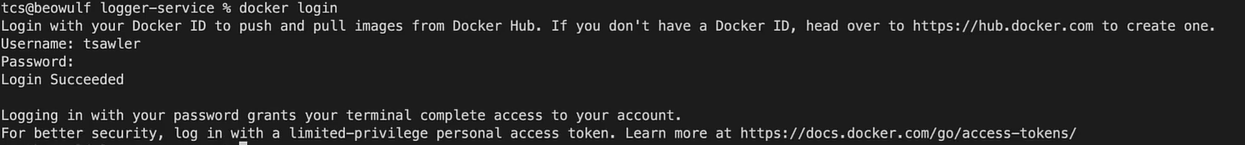
To do that you needed the ID and password

of hub.docker.com

Fire command

docker login

Give your ID and password which was you given on the hub plateform



Command

docker push name/service:version

docker push yagnik/logger-service:1.0.0

Now it is time to create docker yml file

version: '3'

# NOTE: for all of the images for the 5 microservices, you'll

# have to change the image: <name> part to whatever your image names are

services:

broker-service:

image: tsawler/broker-service:1.0.0

ports:

- "8080:80"

deploy:

mode: replicated

replicas: 1

listener-service:

image: tsawler/listener-service:1.0.0

deploy:

mode: replicated

replicas: 1

authentication-service:

image: tsawler/authentication-service:1.0.0

deploy:

mode: replicated

replicas: 1

environment:

DSN: "host=postgres port=5432 user=postgres password=password dbname=users sslmode=disable timezone=UTC connect\_timeout=5"

logger-service:

image: tsawler/logger-service:1.0.0

deploy:

mode: replicated

replicas: 1

mailer-service:

image: tsawler/mail-service:1.0.0

deploy:

mode: replicated

replicas: 1

environment:

MAIL\_DOMAIN: localhost

MAIL\_HOST: mailhog

MAIL\_PORT: 1025

MAIL\_ENCRYPTION: none

MAIL\_USERNAME: ""

MAIL\_PASSWORD: ""

FROM\_NAME: "John Smith"

FROM\_ADDRESS: john.smith@example.com

rabbitmq:

image: 'rabbitmq:3.9-alpine'

deploy:

mode: global

mailhog:

image: 'mailhog/mailhog:latest'

ports:

- "8025:8025"

deploy:

mode: global

mongo:

image: 'mongo:4.2.17-bionic'

ports:

- "27017:27017"

deploy:

mode: global

environment:

MONGO\_INITDB\_DATABASE: logs

MONGO\_INITDB\_ROOT\_USERNAME: admin

MONGO\_INITDB\_ROOT\_PASSWORD: password

volumes:

- ./db-data/mongo/:/data/db

postgres:

image: 'postgres:14.0'

ports:

- "5432:5432"

restart: always

deploy:

mode: replicated

replicas: 1

environment:

POSTGRES\_USER: postgres

POSTGRES\_PASSWORD: password

POSTGRES\_DB: users

volumes:

- ./db-data/postgres/:/var/lib/postgresql/data/