**DOCKER IMAGES**:

CREATE AN IMAGE :

1)SPECIFY A BASE IMAGE

2)RUN SOME ADDITIONAL COMMANDS TO INSTALL OTHER PROGRAMS

3)SPECIFY A COMMAND TO IMAGE TO WHAT TO DO ONCE A CONTAINER IS STARTED

Docker build . -> to build an image

Docker build -t image\_name:latest .

#use existing base image

FROM alpine

#run commands to install additional programs

RUN apk add --update redis

RUN apk add --update gcc

#write a command to tell the image what to do once the container is started

CMD ["redis-server"]

CREATING AN IMAGE FOR NODE JS APPLICATION

FROM alpine

WORKDIR /usr/app

COPY ./ ./

RUN apk add --update nodejs

RUN apk add --update npm

RUN npm install

CMD ["npm","start"]

CONTAINER PORT MAPPING:-

Docker run -p INCOMING REQUEST TO PORT IN LOCAL : PORT ON WHICH APPLICATION IS RUNNING ON DOCKER IMAGE\_NAME

Docker run -it image\_name sh

Will open a shell to perform all the operations

To AVOID UNNECESSARY REBUILDS

FROM alpine

WORKDIR /usr/app

RUN add –update nodejs

RUN add –update npm

COPY ./package.json ./

RUN npm install

COPY ./ ./

CMD [“npm”,”start”]

FOR NODE\_REDIS

FROM **node**:alpine

WORKDIR /usr/app/node\_redis

COPY ./package.json ./

RUN npm install

COPY ./ ./

CMD ["npm","start"]

INTRODUCTION TO DOCKER COMPOSE

Docker-compose up ->docker run my\_image

Docker-compose up –build -> docker build .

Docker run my\_image

Docker-compose up -d : launch in background

Docker-compose down

We can exit from the process by specifying process.exit(0) and it automatically stops our container

RESTART POLICY IN DOCKER:

“no”:Never attempt to restart this container if it stops or crashes

“always”:If this container stops “for any reason” always attempt to restart this

“on-failure”:Only restart if container stops with an error code

“unless-stopped”:Always restart unless we forcibly stop it

Docker-compose ps :status of containers