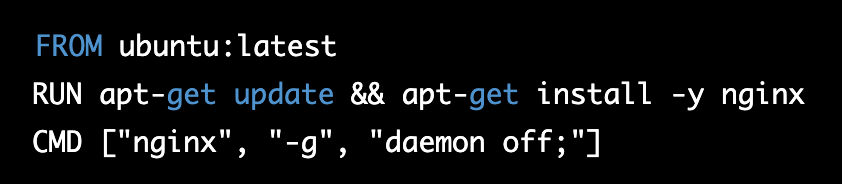
PART 1

* install Docker on your machine
* create a new directory with mkdir for the exercise and navigate to it in the command line
* create a new file called "Dockerfile" in the directory, without any file extension.
* enter the following content to the file



This will create a new Docker image based on the latest version of Ubuntu, and then install the nginx web server

by yourself:

* build the Docker image by running the following command in the command line (hint docker build <something>)
* start the container with port forwarding 8080:80 (8080 for the host machine and the 80 on the container) and with detach mode
* navigate to "[http://localhost:8080](http://localhost:8080/)"
* add a new HTML file in the container and see the change(without stoping the container).

hints:

- go to nginx docs to find where you need to add the html file

- vi or nano exist?

* now create a app dir and add the html file to app dir on the host machine
* now copy the app dir from the Dockerfile to the relevant path to serve the web app (note that when the container is running it depend on the images that he created from )
* run again the container with with volume to the nginx path that serve html files in the container

and create a second.html file in the host volume

PART 2 (optional)

* now add a new different service to the container
* backup the Dockerfile from part1 and

add this content to the Dockerfile:

RUN apt-get install -y nodejs

* create a new directory in the app directory called "nodeApp" and in this directory, create a new file called "server.js" and write a simple NodeJS server that serves the HTML page you created in the previous exercise and copy it to the container.
* expose port 8080 in the Dockerfile
* run the container but now with port 8080 and with server.js service and override the default command from the Dockerfile

break : read about multi stage Dockerfile

BONUS

* add a new service to the container, a database service - MongoDB.
* add db.js file that connect to db and create data in a collection
* modify the server.js to include the db.js file
* create a user in mongo web and sign in
* create db cluster in mongoDB ui
* add the uri to the db.js
* now leave the db cluster and create local mongo server that run in other container ("mongodb://localhost:27017") in your dockerfile
* add a ARG var for db uri in the Dockerfile that you can override in the docker and ENV that use that value run command
* in the db.js use the mongoDB env var to connect
* change the db.js to connect to the local mongo server
* create a new directory in the nodeApp directory called "models" and in this directory, have them create a new file called "db.js" and write a simple MongoDB connection to the server.js file.
* modify the server.js file to include the db.js file and use the MongoDB connection to store and retrieve data
* build the image again and run the container