

Ellie Parobek

Docker Lab

On your VM:

- 1) From home directory:
mkdir nodeapp
cd nodeapp
git clone <https://github.com/azat-co/mern.git>
mv mern/code/index.js .
mv mern/code/package.json .
rm -rf mern
- 2) Create a docker image with our app:
 - a. In a browser (either on host or VM) and go to: <https://hub.docker.com/explore/>
 - b. Search for “node”
 - c. Pick the “official” repository and look at the tags available.
 - d. Go to <http://nodejs.org> in another tab/window to see what the current LTS version is.
 - e. Read the information about choosing which image to use near the bottom of the page (same considerations apply to picking other images)
 - f. In the VM and in the nodeapp folder: **touch Dockerfile**
 - g. **code Dockerfile** and make it:

click on “tags” and pick a version that contains the latest stable version of Node.JS,
you can always change your mind if it doesn't work
FROM node:carbon

Create app directory
RUN mkdir -p /usr/src/app
WORKDIR /usr/src/app

Install app dependencies
COPY package.json /usr/src/app/
RUN npm install

Bundle app source
COPY . /usr/src/app

EXPOSE 3000
CMD ["node", "index.js"]

<save it>

h. Answer the following questions:

i. What does the FROM command do?

Sets the base image to use for the following instructions.

ii. What is the difference between RUN, CMD and ENTRYPOINT?

RUN executes and creates the image, CMD sets default commands and parameters, ENTRYPOINT creates a container and runs as an executable.

iii. What does the WORKDIR command do?

Sets the working directory for RUN, CMD, ENTRYPOINT, COPY, and ADD commands.

iv. What does the COPY command do?

Copies files or directories from src and adds them to the container at the dest path.

v. What does the EXPOSE command do?

Tells the container to listen to the specified ports at runtime.

i. Run: **sudo docker build -t {your-name}/{your-app-name}:{tag} .** Replace the {text} with your information. You can use "1.0" for the version and don't forget the ".".

You can ignore the warnings, etc.

j. Since we need Mongo as well:

code docker-compose.yml and make it:

```
version: '3'
```

```
services:
```

```
  mongo:
```

```
    image: mongo
```

```
    command: mongod --smallfiles
```

```
  networks:
```

```
    - all
```

web:

#use the values from Dockerfile above

image: **{your-name}/{your-app-name}:{tag}**

ports:

- "8080:3000"

depends_on:

- mongo

restart: always

networks:

- all

environment:

MONGODB_URI: "mongodb://mongo:27017/accounts"

networks:

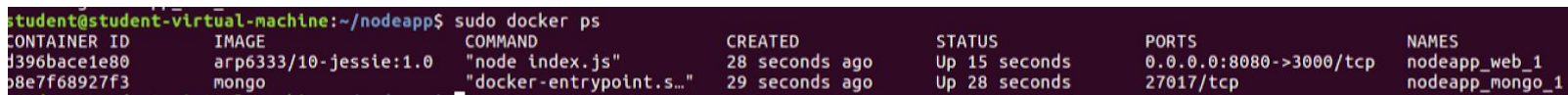
all:

<save it>

k. **sudo docker-compose up -d**

l. **sudo docker ps**

i. put a screen shot of the output here:



CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
d396bace1e80	arp6333/10-jessie:1.0	"node index.js"	28 seconds ago	Up 15 seconds	0.0.0.0:8080->3000/tcp	nodeapp_web_1
b8e7f68927f3	mongo	"docker-entrypoint.s..."	29 seconds ago	Up 28 seconds	27017/tcp	nodeapp_mongo_1

m. **ip addr show ens33** (copy the inet address)

n. On your Mac, open Postman:

Create a "GET" request to <http://<ip address>:8080/messages> – SEND

Paste your response here:

GET http://192.168.190.128:8080/messages Params Send Save

Authorization Headers Body Pre-request Script Tests Cookies Code

TYPE

Inherit auth from parent

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

This request is not inheriting any authorization helper at the moment. Save it in a collection to use the parent's authorization helper.

Body Cookies Headers (6) Test Results Status: 200 OK Time: 33 ms Size: 212 B

Pretty Raw Preview JSON

```
1 []
```

Create a "POST" request to the same URL, click on "Body", then "raw", then enter JSON similar to: {"message":"this is a test","name":"bryan french"} and click SEND.

Past your response here:

POST http://192.168.190.128:8080/messages Params Send Save

Authorization Headers Body Pre-request Script Tests Cookies Code

form-data x-www-form-urlencoded raw binary Text

```
1 {
2   "message": "test",
3   "name": "Ellie Parobek",
4 }
```

Body Cookies Headers (6) Test Results Status: 200 OK Time: 54 ms Size: 246 B

Pretty Raw Preview JSON

```
1 {
2   "_id": "5cba052ada71e9acf9522fd8"
3 }
```

Send the GET request again and paste your response here:

GET Params Send Save

Authorization Headers Body Pre-request Script Tests Cookies Code

TYPE
Inherit auth from parent

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

This request is not inheriting any authorization helper at the moment. Save it in a collection to use the parent's authorization helper.

Body Cookies Headers (6) Test Results Status: 200 OK Time: 6 ms Size: 338 B

Pretty Raw Preview JSON 🔍

```
1 [
2   {
3     "_id": "5cba0553da71e96a64522fd9",
4     "message": null,
5     "name": null
6   },
7   {
8     "_id": "5cba052ada71e9acf9522fd8",
9     "message": null,
10    "name": null
11  }
12 ]
```

(I accidentally called POST twice so that might be why there's two?)

Go to the VM: `sudo docker logs nodeapp_web_1` and paste the response here:

```
student@student-virtual-machine:~/nodeapp$ sudo docker logs nodeapp_web_1
{}
{}

```