**Docker Lab – Edwin Chen**

**ISTE.341.01**

1. From some directory:

**mkdir nodeapp**

**cd nodeapp**

**git clone ssh://bdfvks@solace.ist.rit.edu/var/courses/repos/341/dockerlab** (replace bdfvks with your RIT username.

**mv dockerlab/index.js .**

**mv dockerlab/package.json .**

**rm -rf dockerlab (on windows, you can just remove that folder and any subfolders)**

**code index.js and change line 36 to be insertOne instead of insert**

1. Create a docker image with our app:
   1. In a browser and go to: <https://hub.docker.com/explore/>
   2. Search for “node”
   3. Pick the “official” repository and look at the tags available.
   4. Go to <http://nodejs.org> in another tab/window to see what the current LTS version is.
   5. Read the information about choosing which image to use near the bottom of the page (same considerations apply to picking other images)
   6. In the nodeapp folder:

**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

A screenshot of a computer program

Description automatically generated

**<save it>**

* 1. Answer the following questions:  
     1. What does the FROM command do?  
        The FROM command sets the base image for Docker. In this case it uses the official Node.js image as it is tagged with carbon.
     2. What is the difference between RUN, CMD and ENTRYPOINT?  
        - RUN executes a command when building the image.

- CMD executes the command when the container starts and provides default arguments.

- ENTRYPOINT is used to specify what command to run and executes when the container starts

* + 1. What does the WORKDIR command do?  
       WORKDIR is used to set the working directory inside the image for instructions.
    2. What does the COPY command do?  
       COPY is used to copy files or directories from the host machine into the image.
    3. What does the EXPOSE command do?  
       EXPOSE is used to tell Docker that the container will listen on a certain port. In this case, that is 3000. (The port still needs to be mapped when running)
  1. Run: **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.

* 1. Since we need Mongo as well:

**code docker-compose.yml** and make it (leave off “version: ‘3’):  
  
A screenshot of a computer program

Description automatically generated

**<save it>**

* 1. **docker-compose up -d**
  2. **docker ps** 
     1. put a screen shot of the output here:

A screenshot of a computer

AI-generated content may be incorrect.

* 1. Using curl (or Postman), create a “GET” request to local:8080/messages : **curl http://localhost:8080/messages**

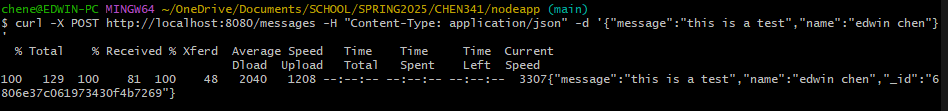
Paste your reponse here:

A screen shot of a black background

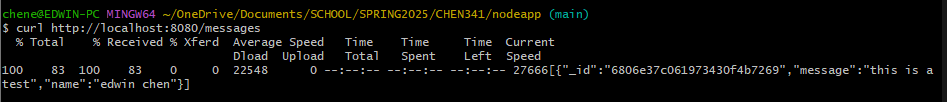
AI-generated content may be incorrect.

Create a “POST” request to the same URL (all on one line): **curl -X POST http://localhost:8080/messages -H "Content-Type: application/json"   
-d '{"message":"this is a test","name":"bryan french"}'**

Past your response here:



Send the GET request again and paste your response here:



Go to the terminal/command prompt (on Windows, you may have to replace the \_’s with –‘s – you can check with docker ps):

**docker logs nodeapp\_web\_1** and paste the response here:

A black screen with a black background

AI-generated content may be incorrect.