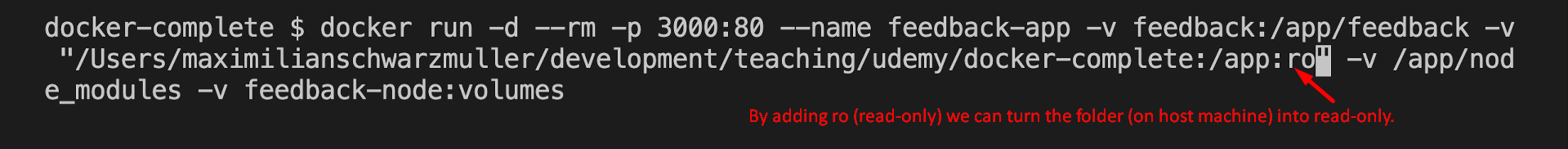
1. docker run <image-name> : Creates a new container and run it in attached mode by default.
   1. **-p <host-port>:<container-port>** <image-name>
      1. To map host and container port to expose container port to host machine.
   2. -d: To run in detached.
   3. -i : interactive mode. Like in case of we want to give input to some process. (Pre-Condition: it should be in attached mode).
   4. -t : TTY To allocate a terminal. If a container is created with this -t, then it will be remembered when running docker start <>
   5. -rm: To remove this created container automatically when stopped anyway.
   6. --name: To name a container. Default: Any random name.
   7. -v:
      1. <vol-name>:<container-path> 🡺 To defined named volume.
      2. <host-path>:<container-path> 🡺 To define Bind Mount.  
         “C:\jatin\Practice\Docker\data-volumes-03-adj-node-code\data-volumes-03-adj-node-code:/app”

“**C:\jatin\Practice\Docker\data-volumes-03-adj-node-code\data-volumes-03-adj-node-code**:/app:**ro**” To make host mapped path read-only for container.  


1. docker container ls -a
   1. docker ps -a
      1. ps: Process. As each container is a process. It lists all the running containers.
         1. -a, --all 🡺 To list stopped containers too.
2. docker build . 🡺 Dot represents the current working dir and it must have Dockerfile.
   1. -t, --tag: To tag an image. syntax 🡺 name:tag where name is repo and tag is index
3. docker start<container id or name>. To run already existing stopped container.
   1. -a: to start in attach mode.
   2. -i: interactive mode.
   3. -t: If container is already created with -t like docker run -t <image> then no need now as it is remembered.
4. docker attach <container id or name> : To attach in,out,err streams to a container.
5. docker logs: To fetch log of a container.
   1. -f: To follow future log.
6. docker prune: To prune all stopped containers.
7. docker rm <container name>: To remove container.
   1. -f: To remove running container.
8. docker rmi <imageid> To remove an image.
9. docker image
   1. prune: To remove all dangling images means without tag.
      1. -a: To remove all unused images dangling and non-dangling.
10. docker stop <container name>
11. docker image inspect <container>
12. docker cp
    1. To copy from host into a running container🡺
       1. docker cp <source> <container\_id:<path>> 🡺 docker cp dummy/. boring\_mobile:/test
    2. To copy from a running container into host
       1. docker cp <container\_id:<path>> <destination>🡺 docker cp boring\_mobile:/test dummy/.
13. Image Registry.
    1. **docker push jatinbansaldocker/ms-payees-openapi:1.0**

**jatinbansaldocker 🡸 Username ms-payees-openapi🡸 Repo 1.0 version.**docker push Host:Name

* 1. docker login and docker logout
  2. docker pull imagename:optionalTag.

1. run npm install: Will be executed during docker build.  
   cmd npm install: Will be executed immediately as soon container is created..
2. **Volume**:  
   A screenshot of a computer

   Description automatically generated
   1. docker volume
      1. rm vol\_name
      2. prune
      3. ls: Will list only anonymous and named volumes not bind-mount.
      4. create <volume-name>: To create a vol and then we can use it as named volume.
      5. Diagram

         Description automatically generatedf

Dockerfile

1. To build our own custom image and its name is Case-sensitive & without extension & create in the root directory of your app.
2. **FROM**: To specify base image.
3. **COPY**:
   1. **COPY . .**
      1. The first dot represents the path on the host machine. It says to include everything from CWD and this path must have Dockerfile.
      2. The 2nd Dot represents the root directory inside the image.
   2. **COPY . /app**
      1. /app 🡺 Instructing to create app folder in the root Dir of image as default is root directory.
4. **RUN: To run a command in image’s file system**
   1. **RUN install node**
      1. **Run:** As image Root Dir is default directory. So command will be run there.
5. **WORKDIR /path/to/workingdir**
   1. WORKDIR /app
      1. From now on, all the RUN commands will be run relative to /app where /app is relative to image root directory.
6. **CMD: To specify startup instruction that will be executed as soon container from this image is created.**
   1. CMD [“node”, “server.js”]
      1. Will run 🡺 node server.js
   2. **Default**: Command from the base image will be executed.  
      If no base image and no CMD you specify, then error.
7. **EXPOSE:** To expose a port inside the Docker to host machine. This is just for documentation purpose and will not define the port.
   1. EXPOSE 8080
8. **Docker build** **.**
   1. To build an image out of Dockerfile.
9. VOLUME 

01: First Dockerfile Example

1. Dockerfile examples:
2. A screenshot of a computer

   Description automatically generated
3. Now run the build command to create Docker Image from the Dockerfile.  
   **docker build .**Run the above command from the location where Dockerfile exists as the dot represents the current location and it must have Dockerfile.  
   **Output**:   
   A screenshot of a computer

   Description automatically generated
4. **Let us run the above created image.**   
   A screenshot of a computer

   Description automatically generated
5. Let’s map the port 80 from container to host machine.  
   