Docker

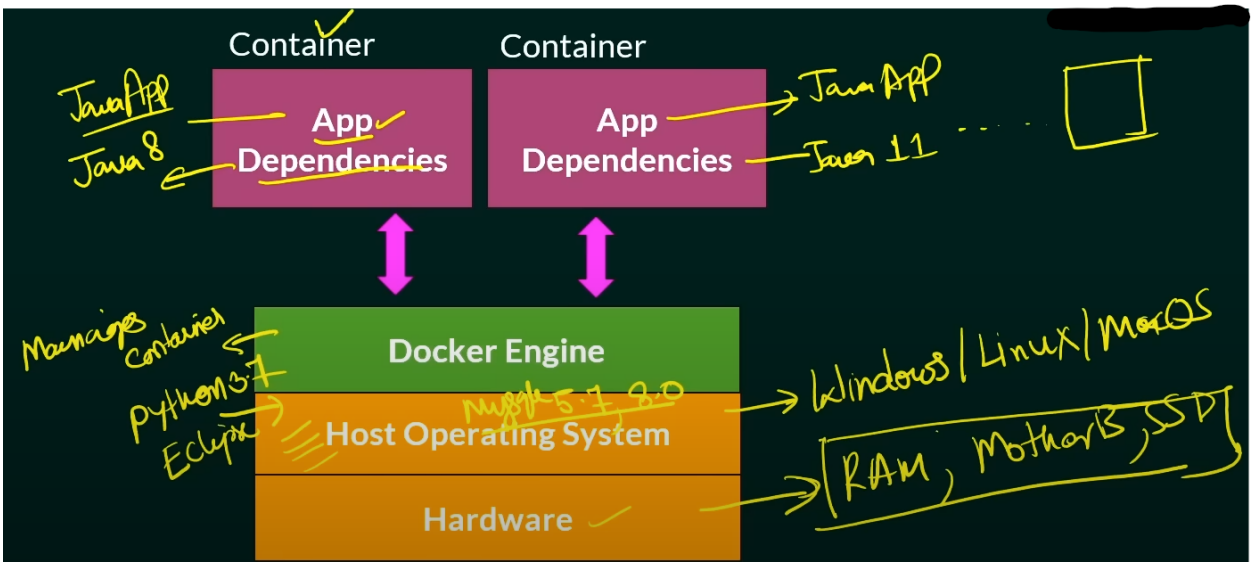
- Docker is an open-source platform to develop, shipping, and running the application.
- Docker is a platform that packages our application and it's all the dependencies into a container.

Why Docker Comes into the Picture?

1. Containerization: It will pack our application and its dependencies into one container.
2. Portability: We can easily move our application from one system to another system.
3. Rapid Development and Scalability: It will help to run the same application with the same dependencies into another system. Like the Developer will make all the applications with the dependencies and it will make it a container and docker will help to move the same application to the testing team and deployment team so It will make rapid development and scalability.
4. Isolation and Security: In docker, there are lots of containers but they will not collapse with each other so they are isolated and secure too.

Example: If I develop one application with code in Java 7 and in another system, I have Java 8 version and if I run that application in this system then some dependencies do not run so docker will help to make all the containers of Java 7 and it's all dependencies and we can run our application to in Java 8 version system environment.

Docker Architecture



Docker File:

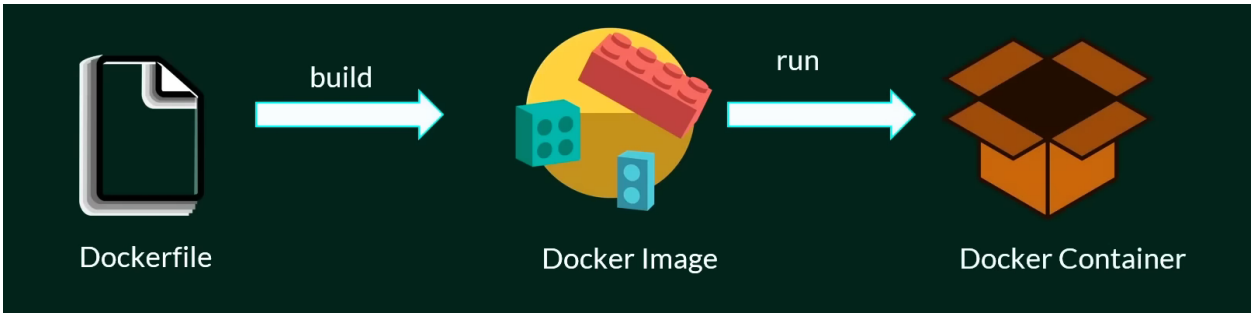
- **Text document which contains all the command lines which can the user call into other command lines to run the docker image.**

Docker Image:

- **It is a blueprint or template for a docker container.**

Docker Container:

- **Running a docker image instance is called Docker Container.**
- The Docker container holds all the data to run our whole application.



Command line:

1. **docker -v / docker --version:** To check the version of docker.

```
// to view docker version
docker -v
```

▼ Output:

```
Docker version 24.0.2, build cb74dfc
```

2. **docker pull <name of image>:** To pull the image from the docker hub.

```
// to pull the OpenJDK
docker pull openjdk
```

▼ Output:

```
Using default tag: latest
latest: Pulling from library/openjdk
197c1adcd755: Pull complete
57b698b7af4b: Pull complete
95a27dbe0150: Pull complete
```

```
Digest: sha256:9b448de897d211c9e0ec635a485650aed6e28d4eca1efbc34940560a480b3f1f
Status: Downloaded newer image for openjdk:latest
docker.io/library/openjdk:latest
```

3. **docker pull <name of image>:<version>:** to pull the specific version of image.

```
// to pull the OpenJDK version 18
docker pull openjdk:18
```

▼ Output:

```
18: Pulling from library/openjdk
Digest: sha256:9b448de897d211c9e0ec635a485650aed6e28d4eca1efbc34940560a480b3f1f
Status: Downloaded newer image for openjdk:18
docker.io/library/openjdk:18
```

4. **docker images:** To see the images available in our docker.

```
// to view all images in our docker
docker images
```

▼ Output:

```
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    9c7a54a9a43c   7 weeks ago    13.3kB
openjdk       18        71260f256d19   4 months ago   470MB
openjdk       latest    71260f256d19   4 months ago   470MB
```

5. **docker search <name of image>:** To search any image from CMD.

```
// to search mysql image
docker search mysql
```

▼ Output:

```
NAME                            DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
mysql                           MySQL is a widely used, open-source relation…   14263     [OK]
mariadb                         MariaDB Server is a high performing open sou…   5453      [OK]
percona                         Percona Server is a fork of the MySQL relati…   616       [OK]
phpmyadmin                      phpMyAdmin - A web interface for MySQL and M…   829       [OK]
bitnami/mysql                   Bitnami MySQL Docker Image                      90                  [OK]
circleci/mysql                  MySQL is a widely used, open-source relation…   29
bitnami/mysqld-exporter                                                         5
ubuntu/mysql                    MySQL open source fast, stable, multi-thread…   51
cimg/mysql                                                                      0
rapidfort/mysql                 RapidFort optimized, hardened image for MySQL   23
rapidfort/mysql8-ib             RapidFort optimized, hardened image for MySQ…   9
google/mysql                    MySQL server for Google Compute Engine          23                  [OK]
hashicorp/mysql-portworx-demo                                                   0
```

```
rapidfort/mysql-official        RapidFort optimized, hardened image for MySQ…   7
newrelic/mysql-plugin           New Relic Plugin for monitoring MySQL databa…   1                    [OK]
databack/mysql-backup           Back up mysql databases to... anywhere!         86
bitnamicharts/mysql                                                             0
linuxserver/mysql               A Mysql container, brought to you by LinuxSe…   38
mirantis/mysql                                                                  0
docksal/mysql                   MySQL service images for Docksal - https://d…   0
linuxserver/mysql-workbench                                                     50
vitess/mysqlctld                vitess/mysqlctld                                1                    [OK]
eclipse/mysql                   Mysql 5.7, curl, rsync                          0                    [OK]
drupalci/mysql-5.5              https://www.drupal.org/project/drupalci         3                    [OK]
drupalci/mysql-5.7              https://www.drupal.org/project/drupalci         0
```

6. **docker run <image name>:** It will run image and start the container.

```
// to run image to start container
docker run openjdk
```

▼ Output:

```
Jun 29, 2023 6:45:32 AM java.util.prefs.FileSystemPreferences$1 run
INFO: Created user preferences directory.
|  Welcome to JShell -- Version 18.0.2.1
|  For an introduction type: /help intro

jshell>
```

7. **docker ps:** It will show the running container.

```
// to view the running container
docker ps
```

▼ Output:

```
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

8. **docker ps -a:** It will show all the containers.

```
// It will show all the containers details
docker ps -a
```

▼ Output:

```
CONTAINER ID   IMAGE         COMMAND    CREATED              STATUS                          PORTS     NAMES
600fe35c0254   openjdk       "jshell"   About a minute ago   Exited (0) About a minute ago             exciting_robinson
4f9895ed2cd6   hello-world   "/hello"   35 minutes ago       Exited (0) 35 minutes ago                 hopeful_hellman
8bc102c322bf   hello-world   "/hello"   50 minutes ago       Exited (0) 50 minutes ago                 determined_mendel
```

9. docker run --name <new name of container> -d <old name or container id>: To change the name of container.

```
// It will show all the containers details
docker run --name openjdkcontainer -d openjdk
```

▼ Output:

```
// before change
CONTAINER ID   IMAGE         COMMAND    CREATED              STATUS                          PORTS     NAMES
600fe35c0254   openjdk       "jshell"   About a minute ago   Exited (0) About a minute ago             exciting_robinson


// after chnage
CONTAINER ID   IMAGE         COMMAND    CREATED             STATUS                         PORTS     NAMES
50dde38e74d8   openjdk       "jshell"   11 seconds ago      Exited (0) 9 seconds ago                  openjdkcontainer
```

10. docker run --name <new name of container> -it -d <name of container>: it will continues run our container.

```
// to run our container continues
docker run --name openjdkrun -it -d openjdk
```

▼ Output:

```
// output
04e6768b4bd471cfdccec1f3aa58999e9a102782684b454698bec253bbd35da4

// to check if our container is running
PS C:\Users\nitin> docker ps
CONTAINER ID   IMAGE     COMMAND    CREATED         STATUS         PORTS     NAMES
04e6768b4bd4   openjdk   "jshell"   6 seconds ago   Up 5 seconds             openjdkrun
```

11.

9. **docker exec -it <container id> <command>:** It will help to enter into running container.

```
// to enter into the running container
docker exec -it 04e6768b4bd4 jshell
```

▼ Output:

```
|  Welcome to JShell -- Version 18.0.2.1
|  For an introduction type: /help intro
```

```
jshell>
```

10. **docker inspect <container id>:** it will provide the details of the container.

```
// to inspect the details of the container
docker inspect 04e6768b4bd4
```

▼ Output:


```
[
    {
        "Id": "04e6768b4bd471cfdccec1f3aa58999e9a102782684b454698bec253bbd35da4",
        "Created": "2023-06-29T07:10:35.713219718Z",
        "Path": "jshell",
        "Args": [],
        "State": {
            "Status": "running",
            "Running": true,
            "Paused": false,
            "Restarting": false,
            "OOMKilled": false,
            "Dead": false,
            "Pid": 8655,
            "ExitCode": 0,
            "Error": "",
            "StartedAt": "2023-06-29T07:10:36.065175014Z",
            "FinishedAt": "0001-01-01T00:00:00Z"
        },
        "Image": "sha256:71260f256d19f4ae5c762601e5301418d2516ca591103b1376f063be0b7ba056",
        "ResolvConfPath": "/var/lib/docker/containers/04e6768b4bd471cfdccec1f3aa58999e9a102782684b454698bec253bbd35da4/resolv.conf",
        "HostnamePath": "/var/lib/docker/containers/04e6768b4bd471cfdccec1f3aa58999e9a102782684b454698bec253bbd35da4/hostname",
        "HostsPath": "/var/lib/docker/containers/04e6768b4bd471cfdccec1f3aa58999e9a102782684b454698bec253bbd35da4/hosts",
        "LogPath": "/var/lib/docker/containers/04e6768b4bd471cfdccec1f3aa58999e9a102782684b454698bec253bbd35da4/04e6768b4bd471cfdccec1f
        "Name": "/openjdkrun",
        "RestartCount": 0,
        "Driver": "overlay2",
        "Platform": "linux",
        "MountLabel": "",
        "ProcessLabel": "",
        "AppArmorProfile": "",
        "ExecIDs": null,
        "HostConfig": {
            "Binds": null,
            "ContainerIDFile": "",
            "LogConfig": {
                "Type": "json-file",
                "Config": {}
            },
            "NetworkMode": "default",
            "PortBindings": {},
            "RestartPolicy": {
                "Name": "no",
                "MaximumRetryCount": 0
            },
            "AutoRemove": false,
            "VolumeDriver": "",
            "VolumesFrom": null,
            "ConsoleSize": [
                40,
                156
            ],
            "CapAdd": null,
            "CapDrop": null,
            "CgroupnsMode": "host",
            "Dns": [],
            "DnsOptions": [],
            "DnsSearch": [],
            "ExtraHosts": null,
            "GroupAdd": null,
            "IpcMode": "private",
```

```
            "Cgroup": "",
            "Links": null,
            "OomScoreAdj": 0,
            "PidMode": "",
            "Privileged": false,
            "PublishAllPorts": false,
            "ReadonlyRootfs": false,
            "SecurityOpt": null,
            "UTSMode": "",
            "UsernsMode": "",
            "ShmSize": 67108864,
            "Runtime": "runc",
            "Isolation": "",
            "CpuShares": 0,
            "Memory": 0,
            "NanoCpus": 0,
            "CgroupParent": "",
            "BlkioWeight": 0,
            "BlkioWeightDevice": [],
            "BlkioDeviceReadBps": [],
            "BlkioDeviceWriteBps": [],
            "BlkioDeviceReadIOps": [],
            "BlkioDeviceWriteIOps": [],
            "CpuPeriod": 0,
            "CpuQuota": 0,
            "CpuRealtimePeriod": 0,
            "CpuRealtimeRuntime": 0,
            "CpusetCpus": "",
            "CpusetMems": "",
            "Devices": [],
            "DeviceCgroupRules": null,
            "DeviceRequests": null,
            "MemoryReservation": 0,
            "MemorySwap": 0,
            "MemorySwappiness": null,
            "OomKillDisable": false,
            "PidsLimit": null,
            "Ulimits": null,
            "CpuCount": 0,
            "CpuPercent": 0,
            "IOMaximumIOps": 0,
            "IOMaximumBandwidth": 0,
            "MaskedPaths": [
                "/proc/asound",
                "/proc/acpi",
                "/proc/kcore",
                "/proc/keys",
                "/proc/latency_stats",
                "/proc/timer_list",
                "/proc/timer_stats",
                "/proc/sched_debug",
                "/proc/scsi",
                "/sys/firmware"
            ],
            "ReadonlyPaths": [
                "/proc/bus",
                "/proc/fs",
                "/proc/irq",
                "/proc/sys",
                "/proc/sysrq-trigger"
            ]
        },
        "GraphDriver": {
            "Data": {
                "LowerDir": "/var/lib/docker/overlay2/591a288b6106b6b3de2d00bb9be63cd22e4a48b1bd3b31e4623bff2e0ba8a00c-init/diff:/var/l
                "MergedDir": "/var/lib/docker/overlay2/591a288b6106b6b3de2d00bb9be63cd22e4a48b1bd3b31e4623bff2e0ba8a00c/merged",
                "UpperDir": "/var/lib/docker/overlay2/591a288b6106b6b3de2d00bb9be63cd22e4a48b1bd3b31e4623bff2e0ba8a00c/diff",
                "WorkDir": "/var/lib/docker/overlay2/591a288b6106b6b3de2d00bb9be63cd22e4a48b1bd3b31e4623bff2e0ba8a00c/work"
            },
            "Name": "overlay2"
        },
        "Mounts": [],
        "Config": {
            "Hostname": "04e6768b4bd4",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": true,
            "OpenStdin": true,
```

```
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/java/openjdk-18/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "JAVA_HOME=/usr/java/openjdk-18",
                "LANG=C.UTF-8",
                "JAVA_VERSION=18.0.2.1"
            ],
            "Cmd": [
                "jshell"
            ],
            "Image": "openjdk",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": {}
        },
        "NetworkSettings": {
            "Bridge": "",
            "SandboxID": "f04abe47d4b3cc56d6e9739d8a3b8760dfbe6342e19204c8b25886627425b312",
            "HairpinMode": false,
            "LinkLocalIPv6Address": "",
            "LinkLocalIPv6PrefixLen": 0,
            "Ports": {},
            "SandboxKey": "/var/run/docker/netns/f04abe47d4b3",
            "SecondaryIPAddresses": null,
            "SecondaryIPv6Addresses": null,
            "EndpointID": "1afe40ae43b2b356477f717b50ba16a63d00b2a9925667ab535122acd67ea728",
            "Gateway": "172.17.0.1",
            "GlobalIPv6Address": "",
            "GlobalIPv6PrefixLen": 0,
            "IPAddress": "172.17.0.2",
            "IPPrefixLen": 16,
            "IPv6Gateway": "",
            "MacAddress": "02:42:ac:11:00:02",
            "Networks": {
                "bridge": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": null,
                    "NetworkID": "a027636e0d66b29b262304dd94c105c42b07642d17924ed77d5d3cf64173df1d",
                    "EndpointID": "1afe40ae43b2b356477f717b50ba16a63d00b2a9925667ab535122acd67ea728",
                    "Gateway": "172.17.0.1",
                    "IPAddress": "172.17.0.2",
                    "IPPrefixLen": 16,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
                    "MacAddress": "02:42:ac:11:00:02",
                    "DriverOpts": null
                }
            }
        }
    }
]
```


11. **docker run --name <new name of container> <name changer container name>:** To change the name of container.

```
// to change the name of the container
docker run --name golangcontainer golang
```

▼ Output:

```
//Before changing the name of the container

CONTAINER ID   IMAGE         COMMAND                  CREATED         STATUS                     PORTS     NAMES
3bb135e44f48   golang        "bash"                   7 seconds ago   Exited (0) 6 seconds ago             jolly_ptolemy
```

```
// After changing name of the container

CONTAINER ID   IMAGE         COMMAND                  CREATED         STATUS                     PORTS     NAMES
4b48c77f0bcc   golang        "bash"                   9 seconds ago   Exited (0) 8 seconds ago             golangcontainer
```

12. **docker run —name <name of running container> -it -d <name of the container>:** It will make our container detach (Running continues in the background).

```
//run the image in detached
docker run --name database -d golang

// to run the Golang container in the background
docker run --name golangcontainer1 -it -d golang
```

▼ Output:

```
CONTAINER ID   IMAGE     COMMAND   CREATED          STATUS          PORTS     NAMES
b0ff1c1416f3   golang    "bash"    11 seconds ago   Up 11 seconds             golangcontainer1
```

13. **docker start <name of previous running container>:** This common will start the container.

```
// start container
docker start golangcontainer1
```

▼ Output:

```
CONTAINER ID   IMAGE     COMMAND   CREATED          STATUS          PORTS     NAMES
b0ff1c1416f3   golang    "bash"    19 minutes ago   Up 36 seconds             golangcontainer1
```

14. **docker stop <name of running container>:** This common will stop the container.

```
//Stop container
docker stop golangcontainer1
```

▼ Output:

```
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

15. **docker rm <container id>**: To remove the container from our docker.

```
// to remove the container
docker rm b0ff1c1416f3
```

▼ Output:

```
//Before removing container

CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS                       PORTS     NAMES
b0ff1c1416f3   golang        "bash"                   22 minutes ago   Exited (137) 48 seconds ago            golangcontainer1
4b48c77f0bcc   golang        "bash"                   31 minutes ago   Exited (0) 31 minutes ago              golangcontainer
3bb135e44f48   golang        "bash"                   37 minutes ago   Exited (0) 37 minutes ago              jolly_ptolemy
67290d4333f2   postgres      "docker-entrypoint.s…"   3 hours ago      Exited (1) 3 hours ago                 postgresContainer
50dde38e74d8   openjdk       "jshell"                 5 hours ago      Exited (0) 27 minutes ago              openjdkcontainer
4f9895ed2cd6   hello-world   "/hello"                 6 hours ago      Exited (0) 6 hours ago                 hopeful_hellman


//After removing container

CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS                      PORTS     NAMES
4b48c77f0bcc   golang        "bash"                   32 minutes ago   Exited (0) 32 minutes ago             golangcontainer
3bb135e44f48   golang        "bash"                   37 minutes ago   Exited (0) 37 minutes ago             jolly_ptolemy
67290d4333f2   postgres      "docker-entrypoint.s…"   3 hours ago      Exited (1) 3 hours ago                postgresContainer
50dde38e74d8   openjdk       "jshell"                 5 hours ago      Exited (0) 28 minutes ago             openjdkcontainer
4f9895ed2cd6   hello-world   "/hello"                 6 hours ago      Exited (0) 6 hours ago                hopeful_hellman
```

16. **docker rm -f <container id>:** To remove background running container.

```
// to remove the running container
docker rm -f b0ff1c1416f3
```

▼ Output:

```
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

17. **docker rmi <image id>:** To remove the image from our docker.

```
// to remove the image
docker rmi 9c7a54a9a43c
```

▼ Output:

```
//Before deleting the image from the docker

REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
golang        latest    77246b1c2182   7 days ago     845MB
postgres      latest    1921dda0e2c5   2 weeks ago    412MB
```

```
hello-world   latest    9c7a54a9a43c   7 weeks ago    13.3kB
openjdk       latest    71260f256d19   4 months ago   470MB


//After deleting the image from the docker

golang       latest    77246b1c2182   7 days ago     845MB
postgres     latest    1921dda0e2c5   2 weeks ago    412MB
openjdk      latest    71260f256d19   4 months ago   470MB
```

18. **docker kill <container id1>….:** To stop the containers with more than one container id.

```
// to stop more than one container
docker kill ad1db8522359 0c340a0affef a266d703ce88
```

▼ Output:

```
//Before killing the running container.
CONTAINER ID   IMAGE     COMMAND    CREATED          STATUS          PORTS     NAMES
ad1db8522359   golang    "bash"     7 seconds ago    Up 6 seconds              database1
0c340a0affef   openjdk   "jshell"   43 minutes ago   Up 43 minutes             jdk
a266d703ce88   golang    "bash"     43 minutes ago   Up 43 minutes             golang


//After running the kill command
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

19. **docker run —name <name of image> -e POSTGRES_PASSWORD=<passwoard> -d postgres: To run the Postgres to continues in the background.**

```
// to run Postgres continues
docker run –name batabase1 -e POSTGRES_PASSWORD=admin -d postgres

docker run -d -p 5432:5432 -e POSTGRES_PASSWORD=admin -e POSTGRES_USER=postgres -e POSTGRES_DB=studentdb -v C:\postgres_container_data:/var

// to enter into bash
PS C:\Users\nitin> docker exec -it 33b6307f3e89 /bin/bash


// to execute the Postgres
psql -h localhost -p 5432 -U postgres
```

Dockerfile:

- Docker can build images automatically by reading the instructions from a `Dockerfile`.
- A `Dockerfile` is a text document that contains all the commands a user could call on the command line to assemble an image.

`FROM <image>`	Defines a base for your image.
`RUN <command>`	Executes any commands in a new layer on top of the current image and commits the result. `RUN` also has a shell form for running commands.
`WORKDIR <directory>`	Sets the working directory for any `RUN`, `CMD`, `ENTRYPOINT`, `COPY`, and `ADD` instructions that follow it in the Dockerfile.

`COPY <src> <dest>`	Copies new files or directories from `<src>` and adds them to the filesystem of the container at the path `<dest>`.
`CMD <command>`	Lets you define the default program that is run once you start the container based on this image. Each Dockerfile only has one CMD, and only the last CMD instance is respected when multiple exist.

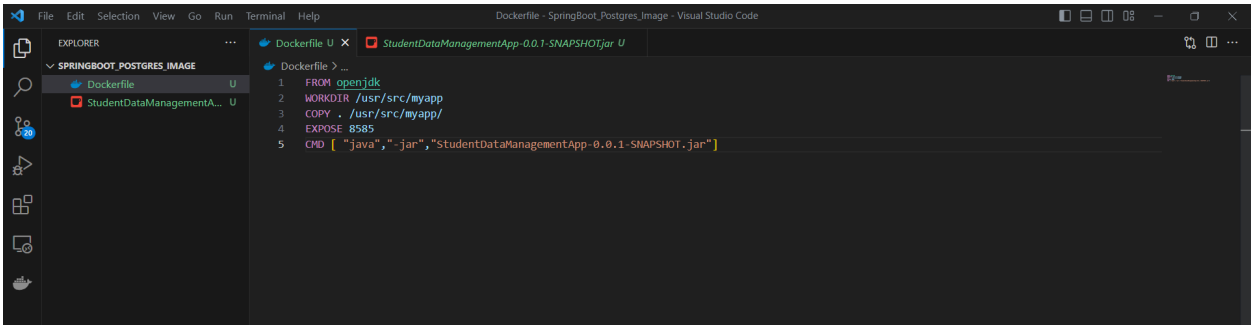
Docker Image Creation:

Step 1: Make our application into executable jar file

Example: StudentDataManagementApp-0.0.1-SNAPSHOT.jar

Step 2: Make folder and paste this jar into this and make a Dockerfile as per docker docs.

Example:



Step 3: Run the below docker command

```
docker build -t <name of the image> .
```