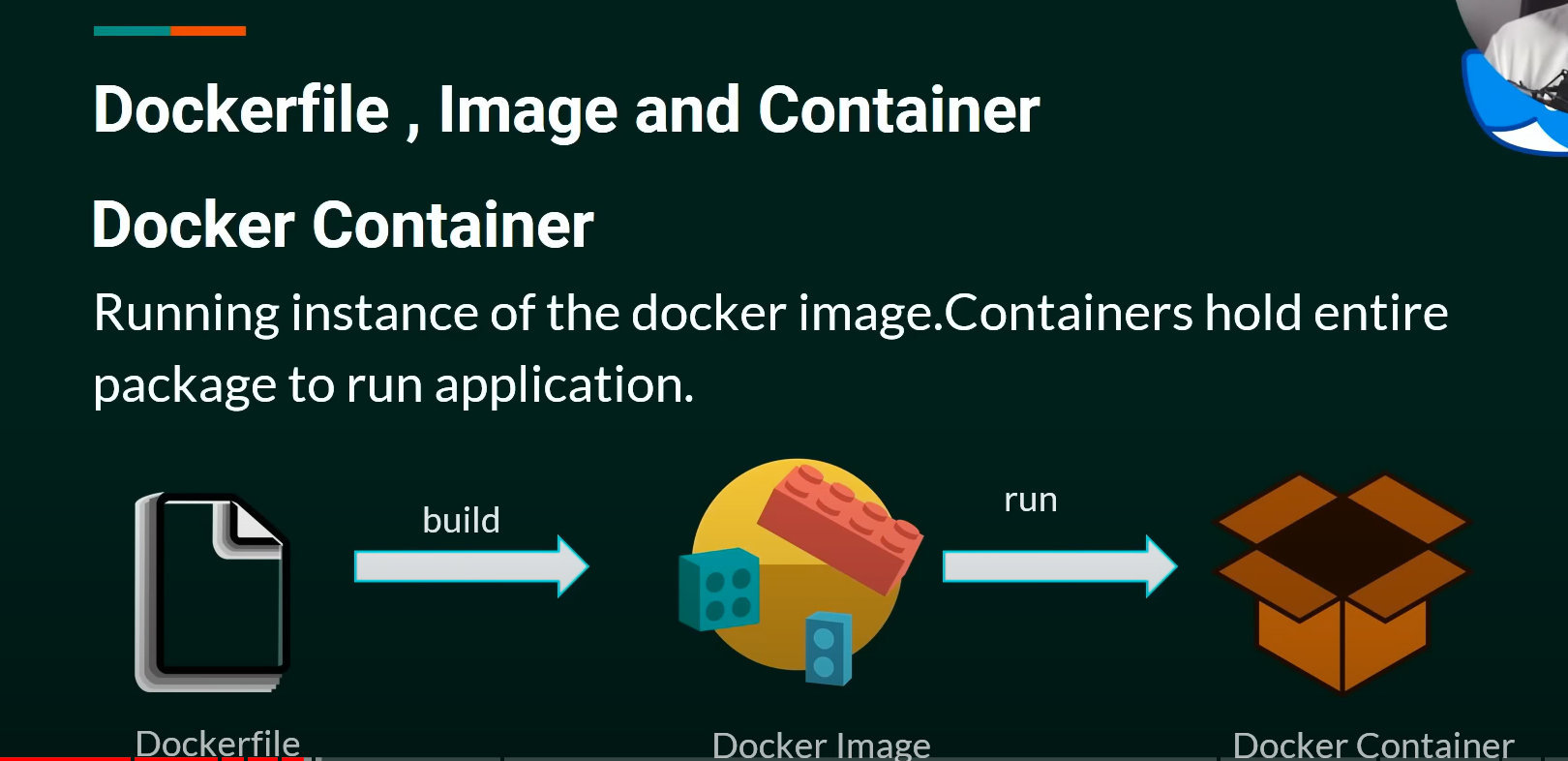
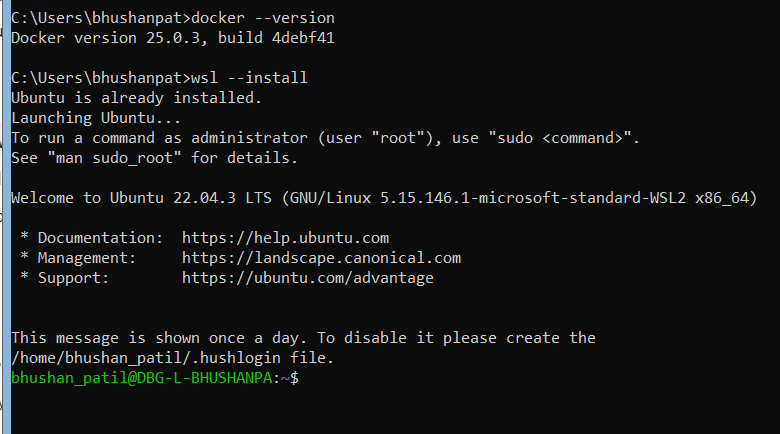
Docker concept



Windows download

Docker Desktop Installer.exe

Link : <https://docs.docker.com/desktop/install/windows-install/>



2)

Link https://learn.microsoft.com/en-us/windows/wsl/install

wsl –install

username Bhushan\_patil

pass root

C:\Users\bhushanpat>docker --version

Docker version 25.0.3, build 4debf41

C:\Users\bhushanpat>docker run hello-world

REPOSITORY TAG IMAGE ID CREATED SIZE

hello-world latest d2c94e258dcb 12 months ago 13.3kB

run spring boot application

Step 1: Create a Dockerfile

Create a Dockerfile in the root directory of your Spring Boot project. This file contains the instructions to build a Docker image for your application. Here’s a basic example of a Dockerfile:

# Use an official OpenJDK 21 runtime as a parent image

FROM openjdk:21

-jdk-slim

# Set the working directory

WORKDIR /app

# Copy the executable JAR file into the container

COPY target/demo-21.jar demo-21.jar

# Expose the port your application runs on

EXPOSE 9292

# Run the JAR file

ENTRYPOINT ["java", "-jar", "demo-21.jar.jar"]

**Step 2: Build the Spring Boot Application**

Package your Spring Boot application into a JAR file. This can be done using Maven or Gradle.

For Maven:

bash

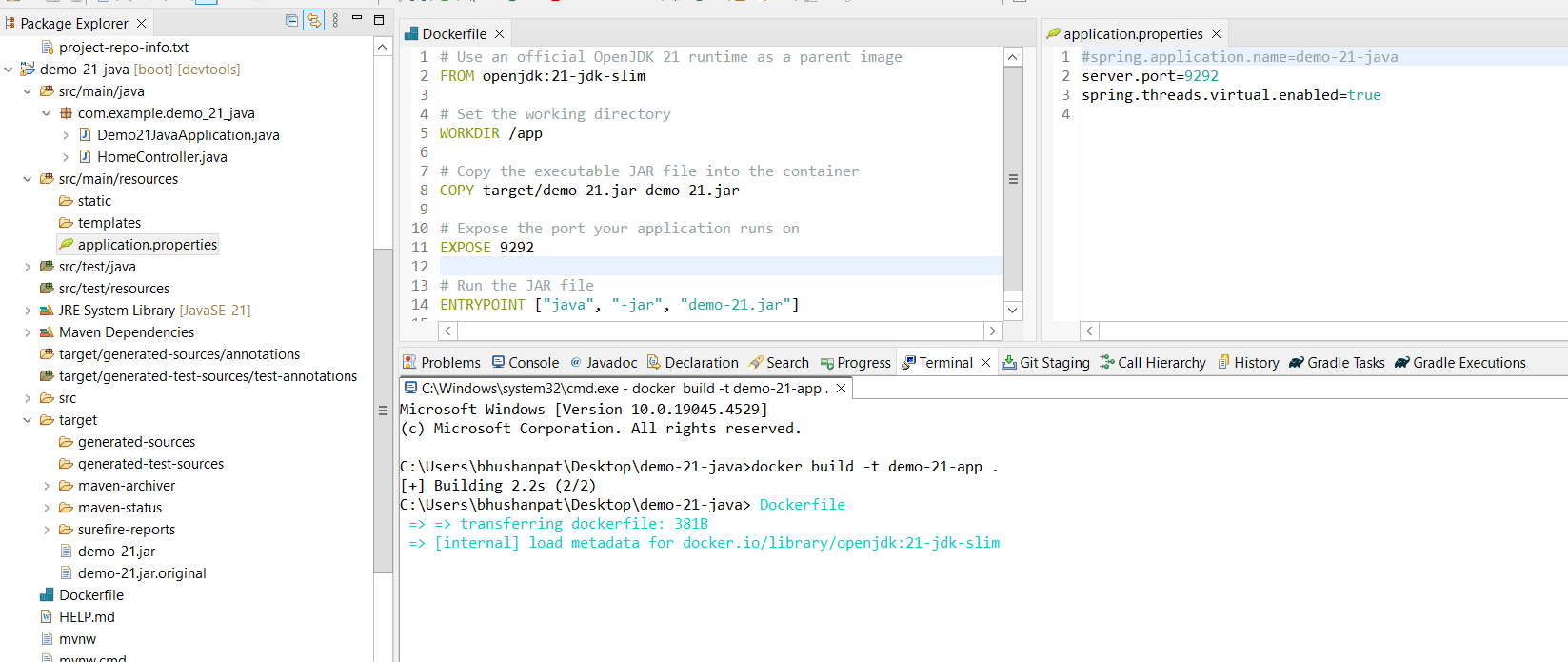
Copy code

mvn clean package

**Step 3:** Build the Docker Image

docker build -t demo-21-app .

(anything we can give demo-21-app ---- images name)



C:\Users\bhushanpat\Desktop\demo-21-java>docker images

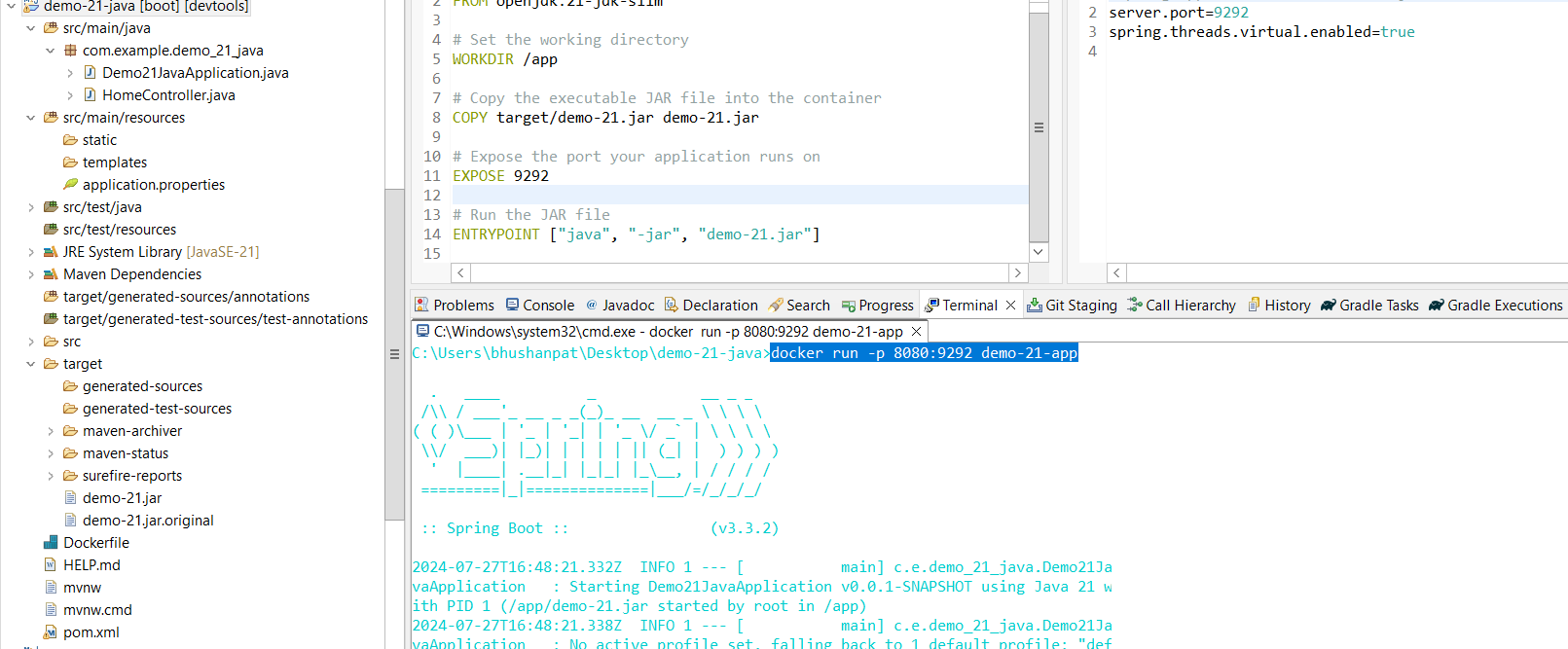
REPOSITORY TAG IMAGE ID CREATED SIZE

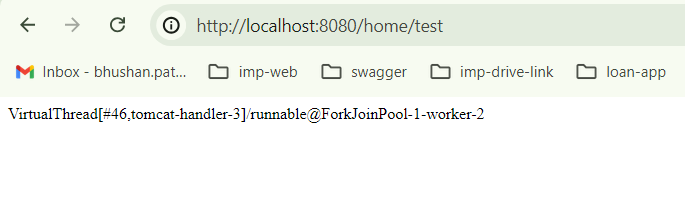
demo-21.jar latest e01ce14f7b44 5 minutes ago 460MB

hello-world latest d2c94e258dcb 15 months ago 13.3kB

Step 4: Run the Docker Container

docker run -p 8080:9292 demo-21-app





How to stop CONTAINER

C:\Users\bhushanpat>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

e8b770ecbe50 demo-21-app "java -jar demo-21.j…" 4 minutes ago Up 4 minutes 0.0.0.0:8080->9292/tcp peaceful\_pare

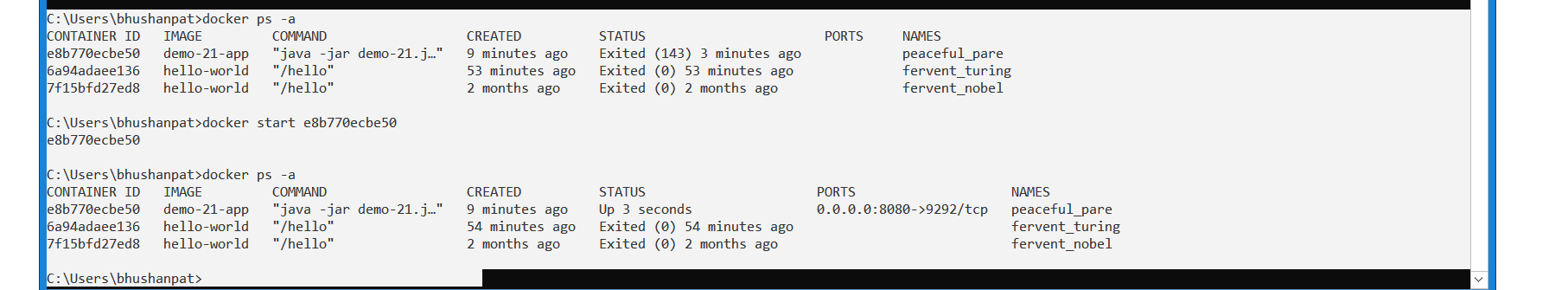
C:\Users\bhushanpat>docker stop e8b770ecbe50

e8b770ecbe50

C:\Users\bhushanpat>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES





C:\Users\bhushanpat>docker ps -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

e8b770ecbe50 demo-21-app "java -jar demo-21.j…" 9 minutes ago Exited (143) 3 minutes ago peaceful\_pare

6a94adaee136 hello-world "/hello" 53 minutes ago Exited (0) 53 minutes ago fervent\_turing

7f15bfd27ed8 hello-world "/hello" 2 months ago Exited (0) 2 months ago fervent\_nobel

C:\Users\bhushanpat>docker start e8b770ecbe50

e8b770ecbe50

C:\Users\bhushanpat>docker ps -a

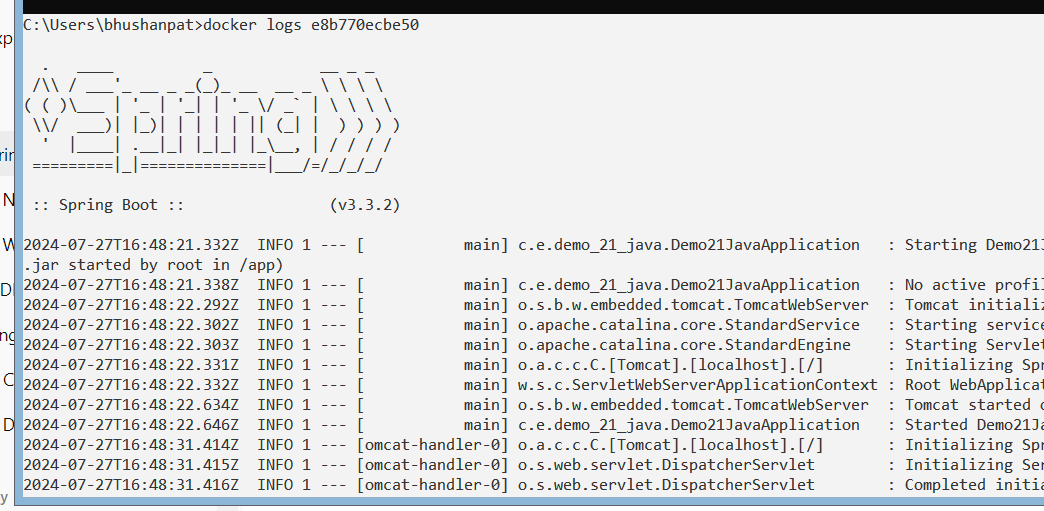
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

e8b770ecbe50 demo-21-app "java -jar demo-21.j…" 9 minutes ago Up 3 seconds 0.0.0.0:8080->9292/tcp peaceful\_pare

6a94adaee136 hello-world "/hello" 54 minutes ago Exited (0) 54 minutes ago fervent\_turing

7f15bfd27ed8 hello-world "/hello" 2 months ago Exited (0) 2 months ago fervent\_nobel

Docker container logs

To view logs for the container with ID e8b770ecbe50:  


3. Access a Running Container’s Shell

C:\Users\bhushanpat>docker exec -it e8b770ecbe50 /bin/bash

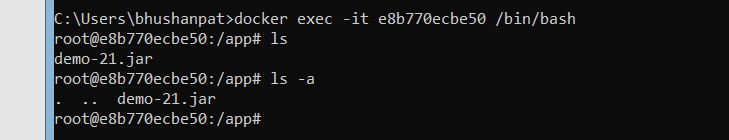
root@e8b770ecbe50:/app# ls

demo-21.jar

root@e8b770ecbe50:/app# ls -a

. .. demo-21.jar

root@e8b770ecbe50:/app#



**Typing exit**: This will close the shell session and return you to your local terminal.

Exit

Inspect Container Details

docker inspect e8b770ecbe50

pull command:

1. C:\Users\bhushanpat>docker pull openjdk

List:

1. C:\Users\bhushanpat>docker images
2. Docker pull python

Image run check command

C:\Users\bhushanpat>docker ps -a

Important Docker Command

Docker version

Docker search

Docker pull

Docker run

Docker ps

Docker stop

Docker restart

Docker kill

Docker exec

Docker login

Docker commit

Docker push

Docker network

Docker history

Docker rmi

Docker ps -a

Docker logs

Docker volume

Docker logout

1. Run springboot application

docker build -t your-image-name .

docker run -p 8080:9091 your-image-name

Pull MongoDB Docker Image and Run Container

docker pull mongo:latest

 Run the MongoDB container:

docker run -d -p 27017:27017 --name mongodb mongo:latest

docker run -d -p 27017:27017 --name mongodb mongo:latest –auth

C:\Users\bhushanpat>docker exec -it mongodb /bin/bash

root@6105fc1ef13f:/# mongosh

Current Mongosh Log ID: 66a679ae65b3330d11149f47

Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.2.10

Using MongoDB: 7.0.12

Using Mongosh: 2.2.10

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).

You can opt-out by running the disableTelemetry() command.

test> use admin

switched to db admin

admin> db.createUser({

... user: "admin",

... pwd: "adminPassword",

... roles: [ { role: "userAdminAnyDatabase", db: "admin" }, "readWriteAnyDatabase" ]

... })

{ ok: 1 }

school> db.createUser({

... user: "schoolUser",

... pwd: "schoolPassword",

... roles: [ { role: "readWrite", db: "school" } ]

... })

MongoServerError[Unauthorized]: Command createUser requires authentication

school>

