Q1

git config --global user.name "Your Name"

git config --global user.email "you@example.com"

git clone https://github.com/your-username/your-repo.git

cd your-repo

echo "New changes" >> file.txt

git add .

git commit -m "Updated file"

git push origin main

git pull origin main

Q2

**Docker Installation Check**

docker --version

docker info

**Pull & Run Ubuntu**

docker pull ubuntu

docker run -it ubuntu

**Manage Containers**

docker ps

docker ps -a

docker stop <container\_id>

docker rm <container\_id>

**Manage Images**

docker images

docker rmi <image\_name>

**Flask App Code**

📄 app.py

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route("/")

def home():

return "Hello from Docker!"

if \_\_name\_\_ == "\_\_main\_\_":

app.run(host="0.0.0.0", port=5000)

📄 requirements.txt

flask

📄 Dockerfile

FROM python:3.8-slim

WORKDIR /app

COPY . .

RUN pip install -r requirements.txt

CMD ["python", "app.py"]

**Build and Run Docker Container**

docker build -t flask-app .

docker run -d -p 5000:5000 flask-app

**Push to Docker Hub**

docker login

docker tag flask-app your\_dockerhub\_username/flask-app

docker push your\_dockerhub\_username/flask-app

Q3

**Step 1: Create Project Structure**

mkdir microservice-app

cd microservice-app

mkdir app db

**Step 2: Create Flask App Code**

📄 app/app.py

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route('/')

def home():

return "Hello from Microservice App!"

if \_\_name\_\_ == '\_\_main\_\_':

app.run(host='0.0.0.0', port=5000)

**Step 3: Add Python Dependencies**

📄 app/requirements.txt

flask

**Step 4: Create Dockerfile**

📄 app/Dockerfile

FROM python:3.9

WORKDIR /app

COPY requirements.txt .

RUN pip install -r requirements.txt

COPY . .

CMD ["python", "app.py"]

**Step 5: Create docker-compose.yml**

📄 docker-compose.yml

version: '3'

services:

web:

build: ./app

ports:

- "5000:5000"

depends\_on:

- db

db:

image: mysql:5.7

restart: always

environment:

MYSQL\_DATABASE: sampledb

MYSQL\_USER: user

MYSQL\_PASSWORD: password

MYSQL\_ROOT\_PASSWORD: rootpass

ports:

- "3306:3306"

**Step 6: Build and Run Containers**

docker-compose up --build

**Step 7: Access App**

* Open browser:  
  👉 <http://localhost:5000>  
  ✅ Output: Hello from Microservice App!

**Step 8: Stop and Clean Up**

docker-compose down

q4

q5

**✅ Prerequisites**

* **Install Java JDK 21**
* Confirm Java is installed:

java -version

**🔽 Step 1: Download & Extract Maven**

1. Download ZIP from:  
   👉 <https://maven.apache.org/download.cgi>  
   🔹 Select: apache-maven-3.9.6-bin.zip  
   ❌ Do **NOT** select src.zip or .tar.gz
2. Extract to:

C:\Program Files\Apache\Maven\apache-maven-3.9.6

**⚙️ Step 2: Set Environment Variables**

Open **Command Prompt (as Admin)** and run:

setx MAVEN\_HOME "C:\Program Files\Apache\Maven\apache-maven-3.9.6" /M

setx PATH "%PATH%;C:\Program Files\Apache\Maven\apache-maven-3.9.6\bin" /M

🔁 **Restart Command Prompt**, then check Maven:

mvn -v

**📦 Step 3: Create a Maven Project**

mvn archetype:generate -DgroupId=com.example -DartifactId=demo-project -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

📁 After execution, the folder demo-project will be created.

**📄 Step 4: Add a Dependency**

Open demo-project/pom.xml, add inside <dependencies> block:

<dependency>

<groupId>com.google.code.gson</groupId>

<artifactId>gson</artifactId>

<version>2.10.1</version>

</dependency>

**🛠️ Step 5: Build the Project**

Navigate to your project directory:

cd demo-project

Run Maven commands:

mvn compile

mvn test

mvn package

This generates:

target/demo-project-1.0-SNAPSHOT.jar

**🚀 Step 6: Run the Application**

If App.java contains a main() method:

java -cp target/demo-project-1.0-SNAPSHOT.jar com.example.App

Q6

**🧹 Step 1: Delete Maven Files**

Go to the demo-project folder and delete:

pom.xml

Do **NOT** delete:

src/

**⚙️ Step 2: Install Gradle on Windows**

1. Download binary-only ZIP from:  
   👉 <https://gradle.org/releases/>
2. Extract it to:

C:\Gradle\gradle-8.x.x

1. Set environment variables from **Admin Command Prompt**:

setx GRADLE\_HOME "C:\Gradle\gradle-8.x.x" /M

setx PATH "%PATH%;%GRADLE\_HOME%\bin" /M

1. Restart Command Prompt and verify:

gradle -v

**📄 Step 3: Create Gradle Build Files**

**🔧 File 1: build.gradle**

Create this inside demo-project/:

plugins {

id 'java'

}

group = 'com.example'

version = '1.0-SNAPSHOT'

repositories {

mavenCentral()

}

dependencies {

testImplementation 'junit:junit:4.13.2'

}

// Compatibility (optional)

java {

sourceCompatibility = JavaVersion.VERSION\_17

targetCompatibility = JavaVersion.VERSION\_17

}

**🔧 File 2: settings.gradle**

Create this inside demo-project/:

groovy

CopyEdit

rootProject.name = 'demo-project'

**📁 Final Folder Structure**

demo-project/

├── build.gradle

├── settings.gradle

└── src/

├── main/java/com/example/App.java

└── test/java/com/example/AppTest.java

**🛠️ Step 4: Build with Gradle**

From the demo-project/ directory:

gradle build

gradle test

Output JAR will be in:

build/libs/demo-project-1.0-SNAPSHOT.jar

**🚀 Step 5: Run the App**

If App.java has a main() method:

java -cp build/libs/demo-project-1.0-SNAPSHOT.jar com.example.App