**Docker compose**

**1..create a multifolder docker-compose.yml for your Maven web project (XYZ) with Tomcat.**

Since you have just **one service (Tomcat + your WAR app)**, the docker-compose.yml will look simple. Later you can extend it (e.g., add MySQL).

## 1.create the docker-compose.yml

version: '3.9'

services:

  tomcat-app:

    image: tomcat:9.0

    container\_name: tomcat\_app

    ports:

      - "8086:8080"   # Host 8086 maps to container 8080 (Tomcat portal)

      - "7007:8080"   # Host 7007 maps to container 8080 (your WAR app)

    volumes:

      - ./your-app.war:/usr/local/tomcat/webapps/your-app.war

    restart: unless-stopped

Steps to Use

Create a file docker-compose.yml in your project folder.

Run:

docker-compose up -d

This will pull kumbhambhargavi/XYZimage:latest from Docker Hub (if not already local), create the container, and expose port **7007**.

Access in browser:

http://localhost:7007/XYZ

(or / if WAR is renamed to ROOT.war).

## 🔹 Useful Commands

Start services:

docker-compose up -d

With this setup:

Access Tomcat manager at http://localhost:8086

Access your app at http://localhost:7007/your-app

If getting 404 in tomcat set the root:

Update docker-compose.yml volume

volumes:

- ./ROOT.war:/usr/local/tomcat/webapps/ROOT.war

**Recreate container:**

1.Stop services: **docker-compose down**

2.docker-compose up -d

3.docker ps

Now **http://localhost:8086/** will load your app directly.

Or

http://localhost:8086/appname/

**View logs:**

docker-compose logs -f

**2..Create the multifolder docker-compose.yml for Tomcat + WAR app to run together with Redis.**

**Find the docker-compose.yml file**

services:

  tomcat-app:

    image: tomcat:9.0

    container\_name: tomcat\_app

    ports:

      - "8086:8080"   # Host 8086 maps to Tomcat container 8080

    volumes:

      - ./app1.war:/usr/local/tomcat/webapps/ROOT.war

    environment:

      - REDIS\_HOST=redis

      - REDIS\_PORT=6379

    depends\_on:

      - redis

    restart: unless-stopped

  redis:

    image: redis:7.0

    container\_name: redis

    ports:

      - "6379:6379"

    restart: unless-stopped

∙ tomcat app → http://localhost:8086/

∙ Redis → connect via localhost:6379 from your app using

docker exec -it redis redis-cli

EXAMPLE :

PS C:\Users\BHARGAVI KUMBHAM\eclipse-workspace\XYZ> docker exec -it redis redis-cli

127.0.0.1:6379> PING

PONG

127.0.0.1:6379> SET mykey "Hello Redis"

OK

127.0.0.1:6379> SET mykey "Hello Redis" GET mykey

(error) ERR syntax error

127.0.0.1:6379> exit

PS C:\Users\BHARGAVI KUMBHAM\eclipse-workspace\XYZ>

**3.Create the create a multi-container docker-compose.yml for your app + MongoDB**

services:

  foodsystem-app:

    image: kumbhambhargavi/APPNAME:latest

    container\_name: APPNAME-container

    ports:

      - "7007:8080"  # Host 7007 → Container 8080

    depends\_on:

      - mongodb

    environment:

      - MONGO\_URI=mongodb://mongouser:mongopass@mongodb:27017/foodsystem

    restart: unless-stopped

  mongodb:

    image: mongo:6

    container\_name: mongodb

    environment:

      MONGO\_INITDB\_ROOT\_USERNAME: mongouser

      MONGO\_INITDB\_ROOT\_PASSWORD: mongopass

    ports:

      - "27017:27017"  # Host 27017 → Container 27017

    volumes:

      - mongodb-data:/data/db

    restart: unless-stopped

volumes:

  mongodb-data:

1. **docker-compose up -d**
2. **docker ps**

Access your app:**http://localhost:7007/APPNAME/**

**docker exec -it mongodb mongosh -u mongouser -p mongopass --authenticationDatabase admin**

1. docker exec → run command inside the container
2. -it → interactive terminal
3. mongodb → container name
4. mongo → starts Mongo shell
5. mongosh → the new MongoDB shell
6. -u / -p → username/password
7. --authenticationDatabase admin → authenticate against admin DB

**Switch to your database: use APPNAME**

**test> use APPNAME**

**switched to db appname**

**APPNAME> db.users.insertOne({ name: "Bhargavi", role: "admin", age: 50 })**

**{**

**acknowledged: true,**

**insertedId: ObjectId('68bbe63da2407e9461fa3350')**

**}**

**APPNAME> db.users.find()**

**[**

**{**

**\_id: ObjectId('68bbe63da2407e9461fa3350'),**

**name: 'Bhargavi',**

**role: 'admin',**

**age: 50**

**}**

**]**

**foodsystem> db.users.updateOne(**

**... { name: "Bhargavi" },**

**... { $set: { role: "superadmin" } }**

**... )**

**...**

**{**

**acknowledged: true,**

**insertedId: null,**

**matchedCount: 1,**

**modifiedCount: 1,**

**upsertedCount: 0**

**}**

**foodsystem> db.users.deleteOne({ name: "Bhargavi" })**

**{ acknowledged: true, deletedCount: 1 }**

**foodsystem> exit**

**4.Create the multifolder docker-compose.yml for your app + PostgreSQL**

A popular, open-source relational DB.

version: "3.9"

services:

foodsystem-app:

image: kumbhambhargavi/fimage:latest

container\_name: -container

ports:

- "7007:8080"

depends\_on:

- postgresdb

environment:

- DB\_HOST=postgresdb

- DB\_PORT=5432

- DB\_USER=fuser

- DB\_PASSWORD=fpass

- DB\_NAME=fappname

postgresdb:

image: postgres:15

container\_name: postgresdb

environment:

POSTGRES\_USER: fuser

POSTGRES\_PASSWORD: fpass

POSTGRES\_DB: fsystem

ports:

- "5432:5432"

volumes:

- pgdata:/var/lib/postgresql/data

volumes:

pgdata:

1.docker ps

2. \l (list of databases)

3. CREATE TABLE orders (

id SERIAL PRIMARY KEY,

item\_name VARCHAR(100),

quantity INT

);

4.

## OptionINSERT INTO orders (item\_name, quantity) VALUES ('Pizza', 2);

## 5.SELECT \* FROM orders;

## 6. \q (to quit)

## 7.http://localhost:7007/appname

## 

## 

## 

## 

## 2: **MongoDB**

If you prefer NoSQL (document database).

version: "3.9"

services:

foodsystem-app:

image: kumbhambhargavi/foodsystemimage:latest

container\_name: foodsystem-container

ports:

- "7007:8080"

depends\_on:

- mongodb

environment:

- MONGO\_URI=mongodb://mongouser:mongopass@mongodb:27017/foodsystem

mongodb:

image: mongo:6

container\_name: mongodb

environment:

MONGO\_INITDB\_ROOT\_USERNAME: mongouser

MONGO\_INITDB\_ROOT\_PASSWORD: mongopass

ports:

- "27017:27017"

volumes:

- mongodata:/data/db

volumes:

mongodata: