Create a three tier application

git clone <https://github.com/amitopenwriteup/dockermultitier.git>

cd dockermultitier

**Step 1: Create a Docker Network**

docker network create backnet

**Step 2: Create a Docker Volume for Persistent Data**

docker volume create db-data

**Step 3: Create a Password File (Without Docker Secrets)**

Instead of docker secret, manually create a file to store the database password:

echo "Welcome1" > db-password.txt

**Step 4: Run the MariaDB Container**

docker run -d \

--name db \

--restart always \

--network backnet \

--mount type=volume,source=db-data,target=/var/lib/mysql \

--mount type=bind,source=$(pwd)/db-password.txt,target=/run/secrets/db-password,readonly \

-p 3306:3306 \

-p 33060:33060 \

-e MYSQL\_DATABASE=example \

-e MYSQL\_ROOT\_PASSWORD=$(cat db-password.txt) \

mariadb:10-focal

**Verify the Running Container**

Check if the container is running:

docker ps

Check logs for errors:

docker logs db

**Connect to the MariaDB Container**

docker exec -it db mariadb -u root -p

(Enter the password from db-password.txt when prompted)

Create backend

**Step 1: Create a Volume for Backend Data**

docker volume create backend-data

**Step 2: Build the Backend Image**

Navigate to the **backend** directory (where your Dockerfile is located) and build the image:

cd backend

docker build -t backend-image .

cd ..

This creates a Docker image named **backend-image**.

To create the frontnet network, run the following command:

docker network create frontnet

**Step 4: Run the Backend Container**

docker run -d \

--name backend \

--restart always \

--network backnet \

--network frontnet \

--mount type=volume,source=backend-data,target=/app/data \

--mount type=bind,source=$(pwd)/db-password.txt,target=/run/secrets/db-password,readonly \

-p 8000:8000 \

-e MYSQL\_ROOT\_PASSWORD=$(cat db-password.txt) \

backend-image

**Step 5: Verify the Running Containers**

docker ps

docker logs backend

Test the Backend API

curl -X GET http://localhost:8000

**Build the Nginx Proxy Image**

docker build -t custom-nginx:latest ./proxy

**Step 2: Run the Proxy Container**

docker run -d \

--name proxy \

--network frontnet \

-p 81:80 \

-v ./proxy/nginx.conf:/etc/nginx/nginx.conf:ro \

--restart always \

custom-nginx:latest

**Step 3: Verify the Running Container**

docker ps

docker logs proxy

docker network inspect frontnet

docker network inspect backnet

**Next Steps**

* If you need to stop and remove the database:

docker stop db && docker rm db

docker volume rm db-data

docker stop backend && docker rm backend

docker stop proxy && docker rm proxy

docker rmi proxy

docker rmi backend

docker network rm backnet

docker network rm frontnet