Docker Assignment – Step■by■Step Guide (WSL2 + Windows)

This guide shows exactly how to build and run a two■service app (Express frontend + Flask backend) with Docker and Docker Compose on Windows using WSL2. It includes all commands, file layouts, and troubleshooting.

0) Prerequisites (once)

• Windows 10/11 with WSL2 and Ubuntu 24.04 installed. • Docker Desktop running (Settings → Resources → WSL Integration → enable Ubuntu). • Confirm in WSL:

```
$ docker version $ docker compose version
```

1) Project Structure

Create this structure (you already have it):

```
docker-assignment/ ■■ backend/ ■ ■■ app.py ■ ■■ requirements.txt ■ ■■
Dockerfile ■■ frontend/ ■ ■■ server.js ■ ■■ package.json ■ ■■
package-lock.json ■ ■■ Dockerfile ■■ docker-compose.yml
```

2) Final App Code

backend/app.py

```
from flask import Flask, request app = Flask(__name__) @app.route("/submit",
methods=["POST"]) def submit(): # Accept form (primary) and JSON (fallback) name
= request.form.get("name") if not name: data = request.get_json(silent=True) or
{} name = data.get("name") return f"Hello {name}, Flask received your data!"
@app.get("/healthz") def healthz(): return "ok", 200 if __name__ == "__main__":
app.run(host="0.0.0.0", port=5000)
```

backend/requirements.txt

```
flask
```

frontend/server.js

```
const express = require("express"); const bodyParser = require("body-parser");
const axios = require("axios"); const app = express(); const PORT = 3000;
app.use(bodyParser.urlencoded({ extended: true })); app.get("/", (req, res) => {
res.send(` Submit `); }); app.post("/submit", async (req, res) => { try { const
params = new URLSearchParams(); params.append("name", req.body.name || ""); const
response = await axios.post( "http://backend:5000/submit", params.toString(), {
headers: { "Content-Type": "application/x-www-form-urlencoded" } } );
res.send(`Response from Flask: ${response.data}`); } catch (err) {
res.status(500).send("Error: " + err.message); } }); app.listen(PORT, "0.0.0.0",
() => console.log(`Frontend running on ${PORT}`));
```

3) Dockerfiles & Compose

frontend/Dockerfile

```
FROM node:18-alpine WORKDIR /app COPY package*.json ./ RUN npm ci --omit=dev COPY
. . EXPOSE 3000 CMD ["node", "server.js"]
```

backend/Dockerfile

```
FROM python:3.11-slim WORKDIR /app COPY requirements.txt . RUN pip install
--no-cache-dir -r requirements.txt COPY . . EXPOSE 5000 CMD ["python", "app.py"]
```

docker-compose.yml

```
services: backend: build: ./backend ports: - "5000:5000" healthcheck: test: ["CMD", "curl", "-f", "http://localhost:5000/healthz"] interval: 5s timeout: 3s retries: 15 frontend: build: ./frontend ports: - "3000:3000" depends_on: backend: condition: service_healthy
```

4) .dockerignore (recommended)

frontend/.dockerignore

```
node_modules npm-debug.log .vscode .git
```

backend/.dockerignore

```
venv __pycache__ *.pyc .vscode .git
```

5) Build & Run (WSL Terminal)

```
cd "/mnt/c/WEZO/All Learning/Devops/docker-assignment" docker compose build docker compose up # run in background: # docker compose up -d # stop: # docker compose down
```

Open in the browser:

• Frontend → http://localhost:3000 • Backend health → http://localhost:5000/healthz

6) Quick Tests

```
# Test backend directly curl -X POST -d "name=Wezo" http://localhost:5000/submit # Tail logs docker compose logs -f backend docker compose logs -f frontend
```

7) Push Images to Docker Hub

```
docker login docker tag docker-assignment-frontend YOUR_DOCKERHUB_USERNAME/frontend:latest docker tag docker-assignment-backend YOUR_DOCKERHUB_USERNAME/backend:latest docker push YOUR_DOCKERHUB_USERNAME/frontend:latest docker push YOUR_DOCKERHUB_USERNAME/backend:latest
```

8) Push Code to GitHub (submission)

```
git init git add . git commit -m "Dockerized Express + Flask with Compose" git branch -M main git remote add origin https://github.com/YOUR_GITHUB_USERNAME/docker-assignment.git git push -u origin main
```

9) Troubleshooting (Fast)

• Compose not found → install Compose v2 plugin after adding Docker's repo for Ubuntu Noble. • Pipe errors → ensure Docker Desktop is running and WSL integration is enabled. • Name shows 'None' → frontend must send application/x-www-form-urlencoded. • Frontend can't reach backend → use http://backend:5000/submit (service name).

Submit your GitHub repository link per the assignment instructions.