Docker Compose for Machine Learning & Data Science

WEEK 1 — Introduction & Setup

Goal: Understand Docker Compose, set it up, and run your first JupyterLab container for ML.

Concepts You'll Learn:

- Docker Containers – Isolated environments packaging code and dependencies.
- Images vs Containers – Blueprints vs running instances.
- Docker Compose – YAML-based orchestration for multi-container apps.
- YAML Structure – Services, volumes, networks.
- Containerized ML Environment – Reproducible setups for consistent results.

Step 1: Install Prerequisites

Install Docker Desktop (Windows/Mac) or Docker Engine (Linux). Verify installation:

```
docker --version docker compose version
```

Step 2: Create Project Folder

```
mkdir docker-ml-lab cd docker-ml-lab
```

Step 3: Dockerfile

```
FROM python:3.11-slim WORKDIR /app RUN pip install --no-cache-dir numpy pandas
scikit-learn matplotlib seaborn jupyterlab EXPOSE 8888 CMD ["jupyter", "lab",
"--ip=0.0.0.0", "--allow-root", "--no-browser"]
```

Step 4: docker-compose.yml

```
version: "3.9" services: jupyterlab: build: . container_name: ml_lab ports: -
"8888:8888" volumes: - ./notebooks:/app/notebooks environment: -
JUPYTER_ENABLE_LAB=yes restart: unless-stopped
```

Step 5: Launch Environment

```
docker compose up --build
```

Open the provided token URL (e.g., http://127.0.0.1:8888/lab?token=...) to access JupyterLab.

Step 6: Test

```
import numpy as np, pandas as pd, sklearn print("Libraries loaded successfully!")
```

Mini Project: Build a reproducible ML-ready Jupyter environment using Docker Compose.

Skill	You Should Be Able To...	Verified
Install Docker & Compose	Run docker compose version	■
Write Dockerfile for Python	Understand image layers	■
Create docker-compose.yml	Run containerized app	■
Run JupyterLab	Access via localhost:8888	■
Persist notebooks	Files remain after restart	■