Step 5: Deployment Containerization with Docker

University: Tek-Up
Group: ING-4-SDIA

Tutor: Haythem Ghazouani

1 Introduction

In professional environments, "it works on my machine" is not an acceptable excuse. **Docker** allows us to package our application, its environment, and its dependencies into a single "container" that runs identically on any server.

Why Containerize?

Containerization ensures that the exact versions of Python, Node.js, and ML libraries are used, preventing version conflicts between team members.

2 Phase 1: The Dockerfile

A `Dockerfile` is a text document that contains all the commands a user could call on the command line to assemble an image.

2.1 Backend Container

We use a lightweight Python image and install our `requirements.txt`.

```
FROM python:3.9-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install -r requirements.txt
COPY . .
CMD ["python", "code/app.py"]
```

2.2 Frontend Container (Multi-stage)

We use a **multi-stage build** to keep the final image small. We build the React app in stage 1 and serve only the results with Nginx in stage 2.

3 Phase 2: Orchestration with Docker Compose

Since our project has two parts (API and Frontend), we use **Docker Compose** to run them together.

Create a `docker-compose.yml` file:

```yaml
version: '3.8'
services:
  backend:
    build:
      context: .
      dockerfile: Dockerfile.backend
    ports:
      - "8000:8000"
  frontend:
    build:
      context: .
      dockerfile: Dockerfile.frontend
    ports:
      - "5173:80"
```

4 Launching the Stack

To build and start both services:

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

Stop everything with:

```
docker-compose down
```

5 Exercise

1. **Persistence:** Modify the Compose file to mount a volume so that logged MLflow experiments are saved even if the container is deleted.
2. **Networking:** Change the React app code to use the service name `http://backend:8000` instead of `localhost` (Hint: Docker internal networking).
3. **Optimization:** Research **Docker Slim** or Alpine images to reduce the backend image size.