Step 3: Database Persistence with PostgreSQL & Docker

Software Engineering Project (Sprint 3)

Tutor: Haythem Ghazouani

1 Introduction

In the early stages of development, we used **H2**, an in-memory database. While fast, H2 loses all data when the application restarts. For a professional prototype, we need a persistent database like **PostgreSQL** running in a **Docker** container.

2 Configuring Docker Compose

We add a `db` service to our `docker-compose.yml`. This service uses the official PostgreSQL image.

Sprint 3 Task: Database Persistence

```
db:
  image: postgres:15-alpine
  environment:
    - POSTGRES_DB=se_db
  volumes:
    - ./data/postgres:/var/lib/postgresql/data
```

The **volume** mapping ensures that your data is saved on your hard drive, even if the container is deleted.

3 Spring Boot Profiles

We use Spring Profiles to separate development (H2) and production (PostgreSQL) settings.

1. **Default:** Uses H2 for local IDE execution.
2. **Prod:** Uses environment variables to connect to the Docker PostgreSQL container.

4 How to Start the full Stack?

From the root directory, run:

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

This will build your Java backend, your React frontend, and start the PostgreSQL database.

5 Exercise

1. Open the database inside the container using the command: `docker exec -it <db_container_id> psql -U user -d se_db`.
2. Identify the table created by Hibernate and run a `SELECT * FROM project;` query.