Backend Setup Guide

LEO-Based Assessment Tool

---

1. Purpose

This document describes how to set up and run the **backend service** of the *LEO-Based Assessment Tool*.

The backend is a **Spring Boot (Java 17)** application that provides RESTful APIs for learning outcome–based assessment, user management, and progress tracking. It connects to a **cloud-based PostgreSQL database (Neon)** and is consumed by the Electron frontend application.

End users do **not** interact with the backend directly.

---

2. Prerequisites

Required Software

For running the backend, the following tools are required:

- Git
- Java **17**
- Docker
- Docker Compose

Optional (for local development)

- Maven (or Maven Wrapper)
- PostgreSQL database access (Neon recommended)

---

3. Repository Setup

Clone the backend repository:

```
git clone https://github.com/piy678/SENGPRJ_Group6
cd SENGPRJ_Group6
```

---

4. Environment Configuration

Database credentials and sensitive configuration values are **not stored** in the repository. They must be provided via **environment variables** or a `.env` file.

Create a `.env` file in the project root directory:

```
SPRING_DATASOURCE_URL=jdbc:postgresql://<host>:<port>/<database>?
sslmode=require
SPRING_DATASOURCE_USERNAME=<username>
SPRING_DATASOURCE_PASSWORD=<password>

SERVER_PORT=8080
SPRING_JPA_HIBERNATE_DDL_AUTO=update
SPRING_JPA_SHOW_SQL=false

CORS_ALLOWED_ORIGINS=http://localhost:5173,http://13.53.169.202:5174
```

Notes

- Database access is provided by **Neon (PostgreSQL cloud)**
- `SPRING_JPA_HIBERNATE_DDL_AUTO=update` is suitable for development
- For production, `validate` or `none` is recommended
- `CORS_ALLOWED_ORIGINS` must include the frontend application URL

---

5. Running the Backend with Docker (Recommended)

The backend is designed to be run using **Docker and Docker Compose**.

From the project root directory:

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

To view logs:

```
docker compose logs -f
```

To stop the backend:

```
docker compose down
```

After startup, the backend is available at:

```
http://localhost:8080
```

---

6. Running the Backend Locally (Maven)

For local development without Docker, ensure all environment variables are set.

Run the application using Maven:

```
mvn clean spring-boot:run
```

Or using the Maven Wrapper:

```
./mvnw clean spring-boot:run
```

The backend will start on the configured port (default: `8080`).

---

7. Common Issues & Troubleshooting

Database Connection Errors

- Verify database URL, username, and password
- Ensure the Neon database is reachable
- Check that SSL mode is enabled if required by Neon

CORS Errors

- Add the frontend URL to `CORS_ALLOWED_ORIGINS`
- Restart the backend after changing environment variables

Port Already in Use

Change the backend port in `.env`:

```
SERVER_PORT=8081
```

---

8. API Access (Overview)

The backend exposes REST APIs used by the frontend, including:

- `/api/leos` – manage learning outcomes and dependencies
- `/api/students` – manage students and enrollments
- `/api/assessments` – record and update assessment results
- `/api/recommendations` – suggest next possible LEOs

---

9. Production Deployment (AWS EC2)

On an AWS EC2 Linux instance:

1. Install Docker and Docker Compose
2. Clone the backend repository
3. Create and configure the `.env` file
4. Start the backend using Docker Compose

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

The backend runs as a containerized service and connects to the Neon PostgreSQL database.

---

10. Related Project

Frontend (Electron UI):

https://github.com/piy678/SENGPRJ_Group6_FrontendPart

Group 6 — SENGPRJ
Supervisor: *Thomas Mandl*