**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"**



**ІКНІ**

Кафедра **ПЗ**

ЗВІТ

До лабораторної роботи № 4

**На тему:** *“Основні типи та функції доступу до БД**MySQL”*

**З дисципліни:** *“Програмування в Інтернет”*

**Лектор:**

асистент каф. ПЗ

Степанов Д. С.

**Виконав:**

ст. гр. ПЗ-22

Гамела Б. А.

**Перевірила:**

старша викладачка каф. ПЗ

Грицай О. Д.

«\_\_\_»  \_\_\_\_\_\_\_\_\_  2025 р.

∑ = \_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_

Львів – 2025

**Тема роботи:** Основні типи та функції доступу до БДMySQL.

**Мета роботи:** Засвоїти елементи створення, модифікації, читання та занесення даних з таблиць БД засобами РНР.

# Теоретичні відомості

PHP підтримує роботу з базами даних MySQL через три основні інтерфейси: об'єктно-орієнтований (MySQLi OOI), процедурний (MySQLi) та PDO (PHP Data Objects). Вони дозволяють створювати, модифікувати, читати та видаляти дані в таблицях MySQL.

**Основні характеристики:**

* **Компіляція PHP**: Для роботи з MySQL потрібна підтримка (--with-mysql=/path/to/mysql). Без вказівки шляху використовуються вбудовані бібліотеки MySQL.
* **Інтерфейси доступу**:
  + **MySQLi (OOI)**: Об'єктно-орієнтований підхід, використовує клас mysqli для підключення та виконання запитів (напр., $conn = new mysqli(...)).
  + **MySQLi (процедурний)**: Використовує функції, такі як mysqli\_connect(), для роботи з БД.
  + **PDO**: Універсальний інтерфейс для роботи з різними БД (включаючи MySQL), підтримує обробку помилок через винятки (try-catch) і є більш гнучким для зміни джерел даних.
* **Підключення до БД**:
  + MySQLi: перевірка підключення через connect\_error (OOI) або mysqli\_connect\_error() (процедурний).
  + PDO: використовує new PDO() з обробкою помилок через PDOException.
* **Створення БД**:
  + MySQLi: запит CREATE DATABASE виконується через $conn->query() (OOI) або mysqli\_query() (процедурний).
  + PDO: запит виконується через $conn->exec() з обробкою помилок.
* **Формат відповідей**: Сервер повертає дані у JSON, наприклад, помилки ({status: false, error: {...}}) або успішні операції ({status: true, id: 1}).
* **Застосування**: Інтерфейси використовуються для створення форм, обробки даних студентів (додавання, редагування, видалення) та повернення результатів у JSON. PDO рекомендується для проєктів, де можлива зміна БД, завдяки його універсальності.

# Формулювання завдання

1. Розробити web-сторінку згідно [макета](https://cacoo.com/diagrams/ZvVhYS3UpG5PdbBy/EDE3A) (wireframe).

2. Сторінка повинна відповідати наступним [вимогам](https://docs.google.com/document/d/15B3YB6pn-m4WZZtV4-iVfSlNV4h-sJA7Tcr-TV-rNzI/edit?usp=sharing).

# Результати виконання (спільні 3-4 лабораторна)

Репозиторій проекту на github.com: [PI-Labs](https://github.com/thething1001/PI-Labs)

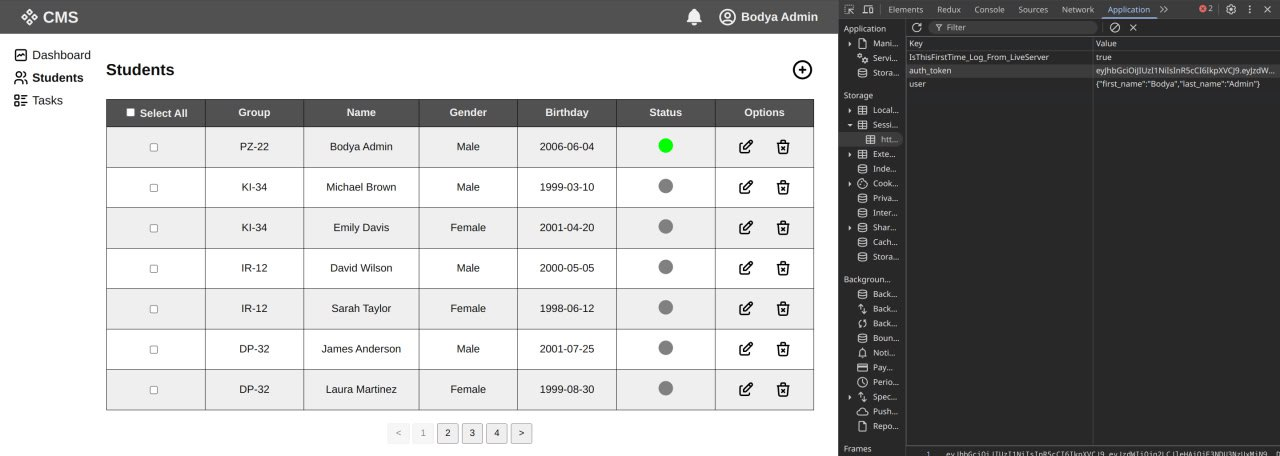
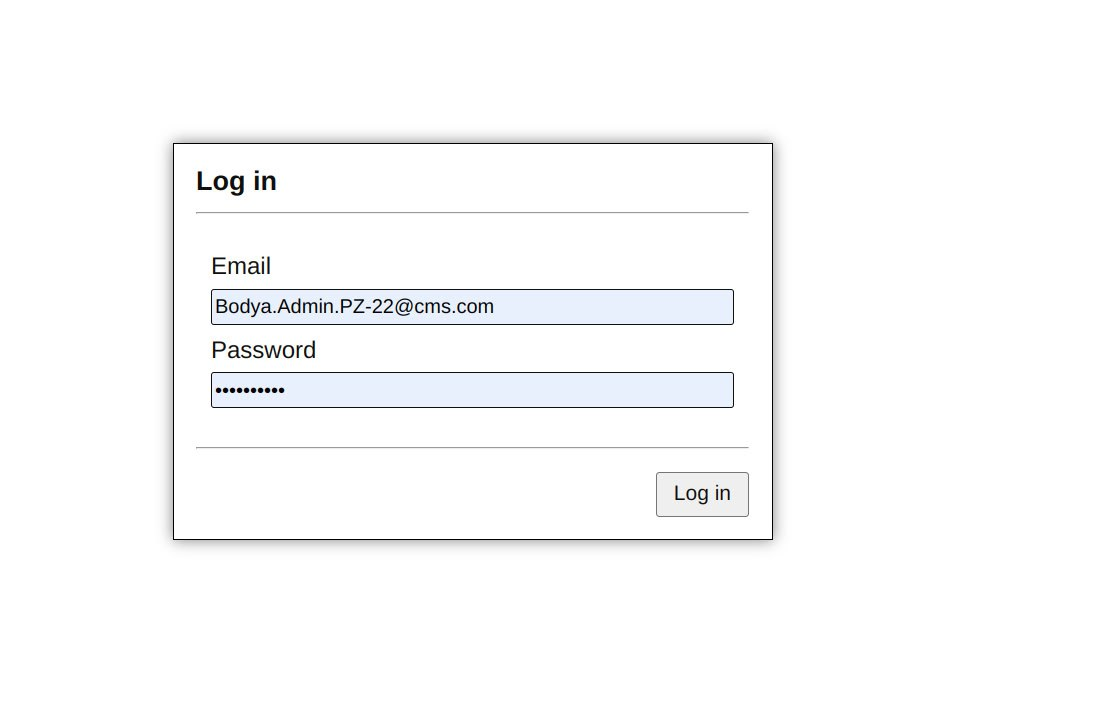


Рис. 1. Авторизація (JWT)

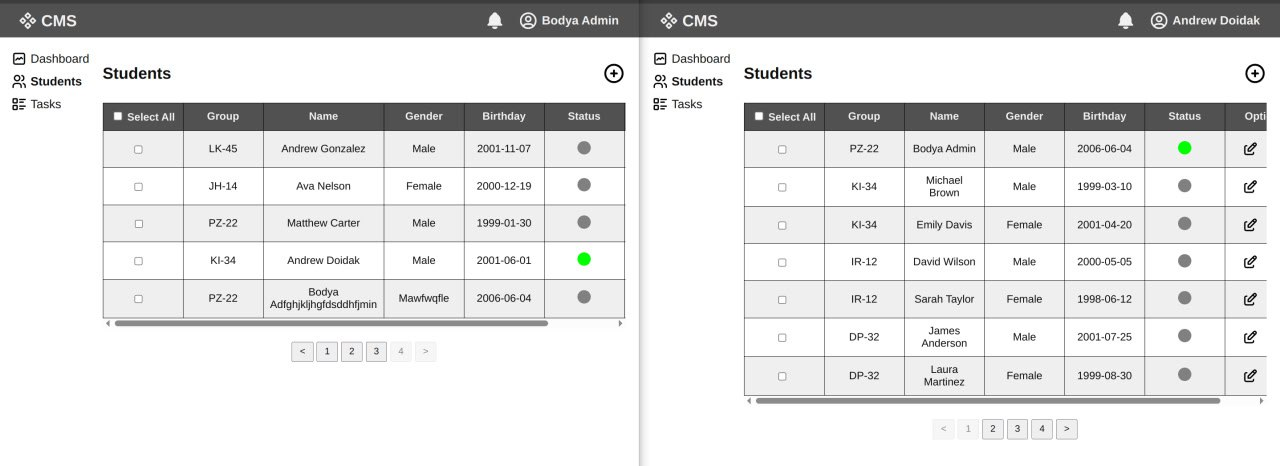


Рис. 2. Відображення студентів та статусу

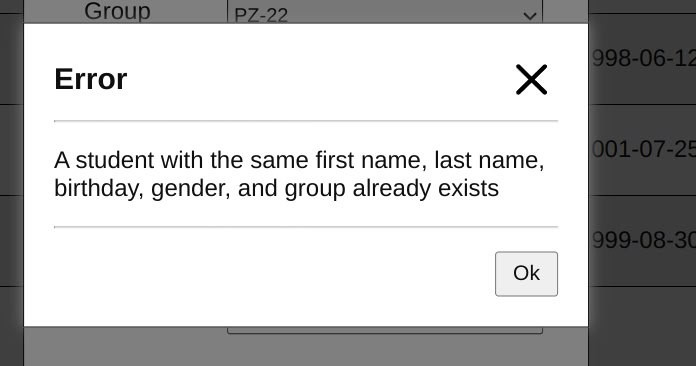


Рис. 3. Повідомлення про помилки з backend’у

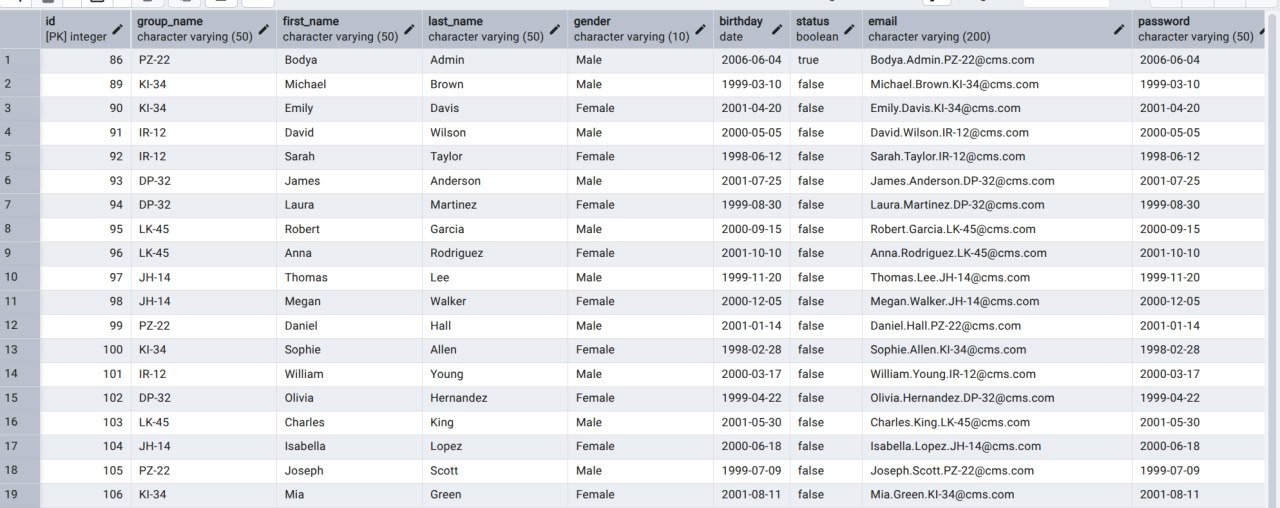


Рис. 4. База даних

# Висновки (спільні 3-4 лабораторна)

Під час виконання лабораторних робіт №3 та №4 я поглибив знання про серверну веб-розробку з використанням PHP та взаємодію з базами даних PostgreSQL. Було розроблено веб-сторінку з формою для додавання, редагування та видалення даних студентів, що взаємодіє з сервером через JSON-відповіді. Серверна частина, побудована на Apache, використовувала PDO для обробки запитів до PostgreSQL, валідації даних та повернення результатів у JSON-форматі (успішні операції або помилки). Для авторизації застосовано JWT. На клієнтській стороні використано JavaScript для обробки подій, відображення помилок та оновлення таблиці студентів.

Робота розвинула практичні навички створення динамічних веб-додатків, поглибила розуміння серверної логіки, роботи з PostgreSQL через PDO, авторизації за допомогою JWT та інтеграції клієнт-серверних технологій, що сприятиме подальшому розвитку компетенцій у веб-розробці.