Лабораторна робота № 4

Тема: Основні типи та функції доступу до БД **MySQL**

Мета: Засвоїти елементи створення, модифікації, читання та занесення даних з таблиць БД засобами PHP

Типи інтерфейсів доступу до БД

Для використання MySQL-сервера баз даних необхідно компілювати php програми з відповідною підтримкою (опція --with-mysql). Якщо шлях до mysql не вказується, php використовуватиме вбудовані в mysql бібліотеки клієнта. Специфікація шляху до mysql: --with-mysql=/path/to/mysql. Тоді php використовує відповідні бібліотеки.

Для використання MySQL-сервера баз даних необхідно компілювати php програми з відповідною підтримкою (опція --with-mysql). Якщо шлях до mysql не вказується, php використовуватиме вбудовані в mysql бібліотеки клієнта. Специфікація шляху до mysql: --with-mysql=/path/to/mysql. Тоді php використовує відповідні бібліотеки.

Сучасна версія PHP 8 підтримує доступ до бази даних MySQL командами, які в своїй назві використовую термін MySQLi для позначення об'єктно-орієнтованого та процедурного варіантів використання. Таке позначення використовується якщо реалізується доступ тільки до MySQL і не передбачається переналаштування на іншу базу даних. Якщо ж в програмі можливі зміни джерела даних, раціонально використати об'єктно-орієнтований інтерфейс доступу PDO (PHP Data Objects), який передбачає підключення до 12 різних баз даних.

Продемонструємо на невеличких прикладах команди різних інтерфейсів :

Об'єктно-орієнтований підхід – під'єднання до бази даних :

```
<?php
…….
$conn = new mysqli($servername, $username, $password);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

Процедурний підхід – під'єднання до бази даних

```
<?php
…….
$conn = mysqli_connect($servername, $username, $password);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

Під'єднання до бази даних у PDO:

```
<?php
. . . . . .
try {
    $conn = new PDO("mysql:host=$servername;dbname=myDB",
$username, $password);
    $conn->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully";
    }
catch(PDOException $e)
    {
    echo "Connection failed: " . $e->getMessage();
    }
?>
```

Типи інтерфейсів доступу до БД

В такому ж порядку подамо приклади створення бази даних :

ООП :

```
 <?php
. . . . . .
$sql = "CREATE DATABASE myDB";
if ($conn->query($sql) === TRUE) {
    echo "Database created successfully";
} else {
    echo "Error creating database: " . $conn->error;
}
$conn->close();
?>
```

ПП.:

```
<?php
. . . . . .
$sql = "CREATE DATABASE myDB";
if (mysqli_query($conn, $sql)) {
    echo "Database created successfully";
} else {
    echo "Error creating database: " . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

PDO :

```
 <?php
. . . . . .
try {
    . . . . . .
    $sql = "CREATE DATABASE myDBPDO";
    $conn->exec($sql);
    echo "Database created successfully<br>";
    }
```

```
catch(PDOException $e)
    {
    echo $sql . "<br>" . $e->getMessage();
    }
$conn = null;
?>
```

Приклади відповідей серверної частини у json форматі:

Приклад повернення помилки

```
{status: false, error:{code: 100, message: "Not found student"}}
```

Приклад успішного додавання студента

```
{status: true, error:null, id: 1}
```

Приклад отримання інформації про студента

```
{status: true, error:null, user: {
id: 1,
name_first: "Test1",
name_last: "Test2",
status: true
 }}
```

Завдання до лабораторної роботи № 4:

Спроектувати базу даних з таблицею студентів яка має містити дані про студента відповідно до наданої таблиці у макеті **Students** (**https://cacoo.com/diagrams/ZvVhYS3UpG5PdbBy/EDE3A**).
Використовуємо Gitlab (https://gitlab.com)

Розробити 2 складові частини проекту :

- Сайт з формою, що надсилає на сервер дані студента з форми у вигляді json або пост форми
- Серверна частина перевіряє чи наявні та коректні всі поля
- Серверна частина має мати методи для додавання/редагування та видалення студента. Методи мають працювати з базою даних. Додавати/змінювати або видаляти дані.
- Серверні методи після опрацювання з БД мають повертати інформацію у json форматі. Повертати помилку у json якщо не існує юзера, якщо немає підключення до бази і т.д. Коли все успішно виконано повертати результат також у json
- Сайт реагує на відповіді. Якщо прийшла помилка відображає, якщо немає помилки додає/оновлює або видаляє студента у таблиці.