**Лабораторна робота 16.**

**Реалізація back-end REST-API для відображення вмісту однієї відомості.**

**Мета роботи:** навчитись підключатись до БД із зовнішнього коду та виконувати команди вибірки та модифікації даних.

**Задачі:**

* Налаштувати Open server panel для можливості підключення до інстанції SQL Server
* Реалізувати підключення до серверу та виведення тестових даних
* Реалізувати класи для роботи із БД студентських іспитів
* Розробити сервіс для роботи із таблицею результатів студентських іспитів
* Виконання індивідуального завдання згідно варіанту

1. У рамках поточної та наступних робіт клієнтський застосунок буде розроблятись у вигляді веб-сторінки із використанням мови PHP для реалізації серверної (back-end) частини та HTML+CSS+JS для реалізації клієнтської (font-end) частини. Взаємодія між ними буде організована через REST-API у вигляді обміну даними у JSON форматі за протоколом http. В рамках даної та наступних робіт мається на увазі відповідний рівень володіння мовами PHP, JS і основам ООП та не приділяється детальної уваги розгляду синтаксичних конструкцій.

Також у якості WAMP (windows + apache + mysql + php) буде використано open server panel, завантажити який та отримати детальну інформацію про налаштування можна на офіційному сайті «https://ospanel.io/».

1. Починаємо нашу роботу із налаштування open server panel для можливості підключення до SQL Server. Річ у тім, що більшість веб-серверів після установки вже налаштовані для підключення та роботи із СКБД MySQL, але нажаль, не з SQL Server. Тому доведеться скопіювати та підключити деякі додаткові бібліотеки, щоб зробити це можливим. На даний момент на офіційному сайті Microsoft за посиланням «https://docs.microsoft.com/ru-ru/sql/connect/php/microsoft-php-driver-for-sql-server?view=sql-server-2017» доступні для завантаження драйвери для найбільш популярних операційних систем (Windows, MacOS та найпопулярніших версій Linux). Також є детальна інструкція щодо їх установки та приклади коду для підключення до БД.

Тут наведемо, що потрібно зробити у комп’ютерному класі нашої локальної мережі (якщо це ще не зроблено). У папці із лабораторними роботами знайдіть директорію «php-drivers». У папці, з якої запускається OS Panel зайдіть до «modules\php\PHP-7.1-x64\ext» (замість PHP-7.1-x64 може бути інша папка, якщо ви використовуєте іншу версію – подивитись на вкладці «модулі» вікна налаштувань OS Panel. Скопіювати із «php-drivers» до «modules\php\PHP-7.1-x64\ext» файли «php\_pdo\_sqlsrv\_71\_ts\_x64.dll» та «php\_sqlsrv\_71\_ts\_x64.dll» (або «php\_pdo\_sqlsrv\_71\_ts\_x86.dll» та «php\_sqlsrv\_71\_ts\_x86.dll» у випадку 32-х розрядної ОС). Назва файлу також має бути відповідною у випадку використання версії PHP 7.0 або 7.2.

Зайдемо до файлу налаштувань модуля PHP (рис.1)

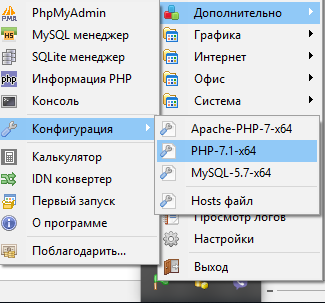


Рис.1. Редагування php.ini в open server panel

У розділі [PHP] знаходимо рядки, що починаються із «extension=»та додаємо 2 рядки із скопійованими попередньо бібліотеками (рис2)

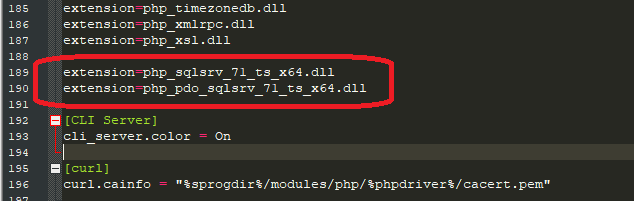


Рис.2. Підключення бібліотек драйверів SQL Server у php.ini

У папці OS Panel директорії «domains» створюємо папку «sql-server» а у ній файл index.php наступного вмісту (рис.3). Замість «localhost» має бути вказане ім’я нашого SQL-серверу.

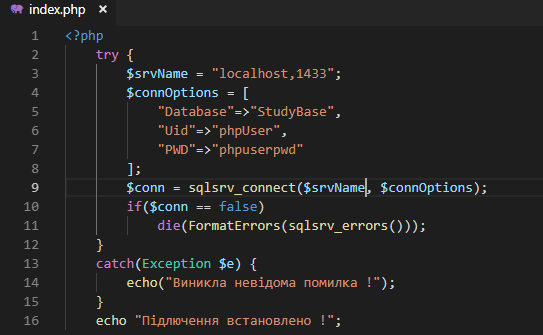


Рис.3. Перевірка успішності підключення до SQL Server.

Перезапустимо OS Panel та у адресному рядку браузера наберемо «http://sql-server/». Якщо все виконано вірно, маємо отримати повідомлення про встановлення підключення (рис.4)

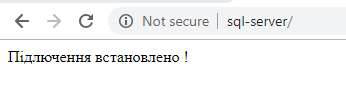
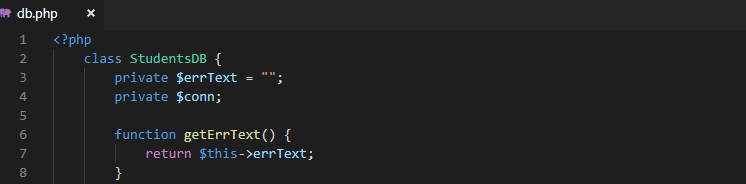
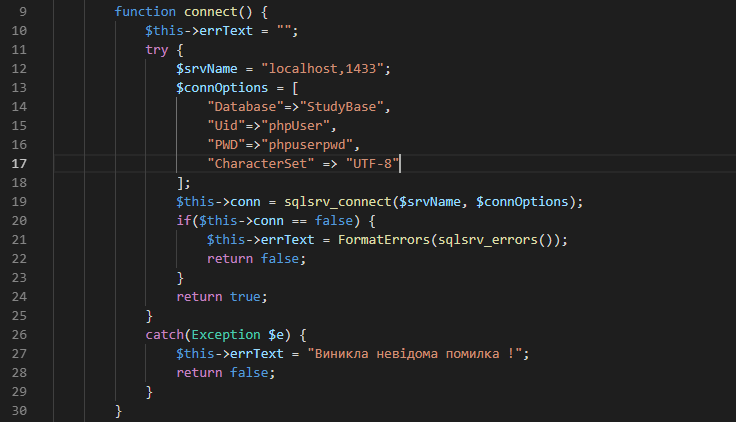


Рис.4. Повідомлення про встановлення підключення.

1. Виконаємо читання даних щодо однієї з студентських відомостей та їх вивід на екран. Відразу зазначимо, що підхід, що наведений нижче не є найпростішою реалізацією даної задачі, але від використовує методологію ООП для представлення та роботи із даними, тому є більш правильним з точки зору перспектив подальшого розширення системи.

Для початку реалізуємо клас для роботи із нашою БД в цілому. Від буде вміти підключатися (та відключатися) до БД, виконувати запит та повертати результат у вигляді масиву. Реалізуємо його, створивши директорію «data», а у ній «db.php» (рис.5)







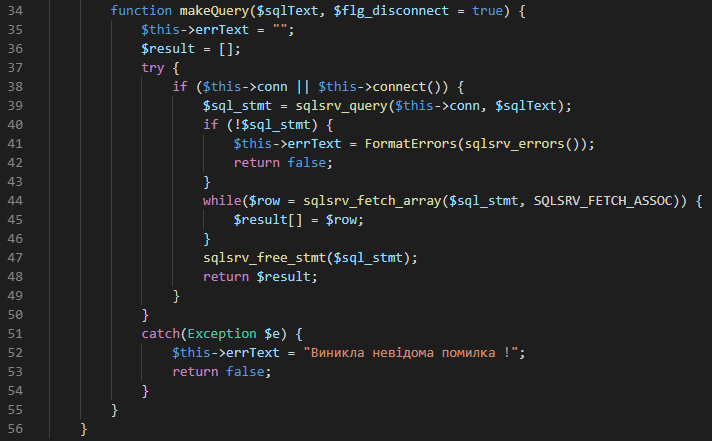


Рис.5. Клас для роботи із БД студентів

Далі у тій же директорії реалізуємо клас для кожної із сутностей, з якими матимемо справу (студент, група, дисципліна, викладач) із відповідними полями, конструктором та набором гетерів (при реалізації визначаємо лише ті поля, що будуть використовуватись, так, наприклад, у викладача поки не буде поля «кафедра») (рис.6-9)

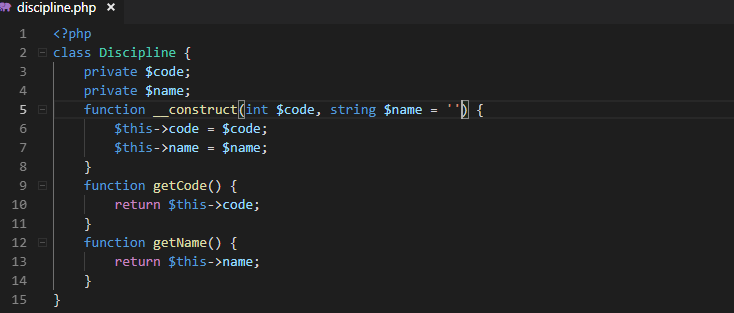


Рис.6. Реалізація класу дисципліна

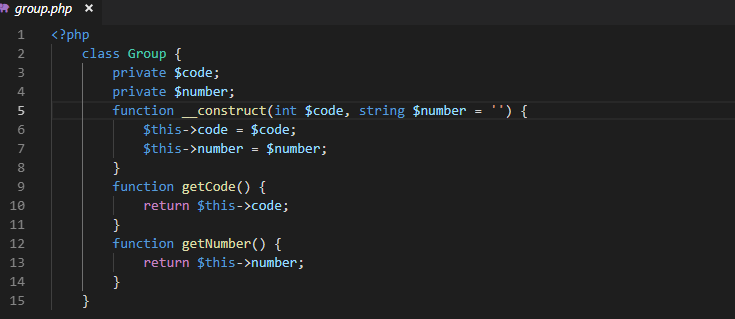


Рис.7. Реалізація класу група

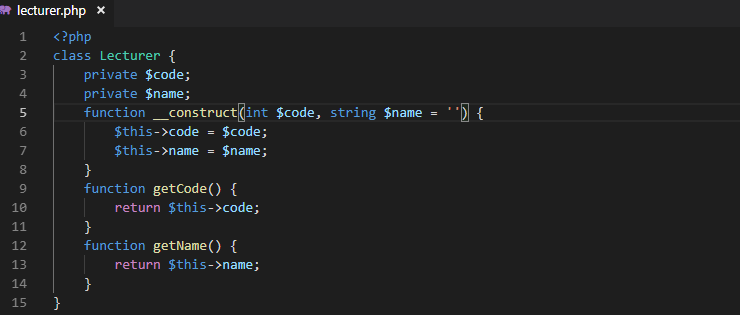


Рис.8. Реалізація класу викладач

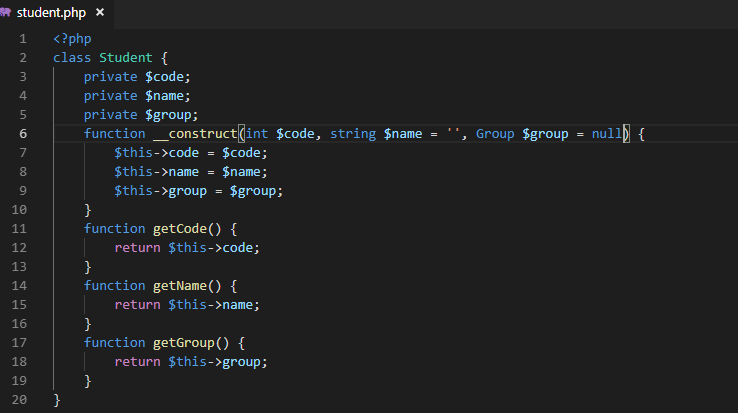
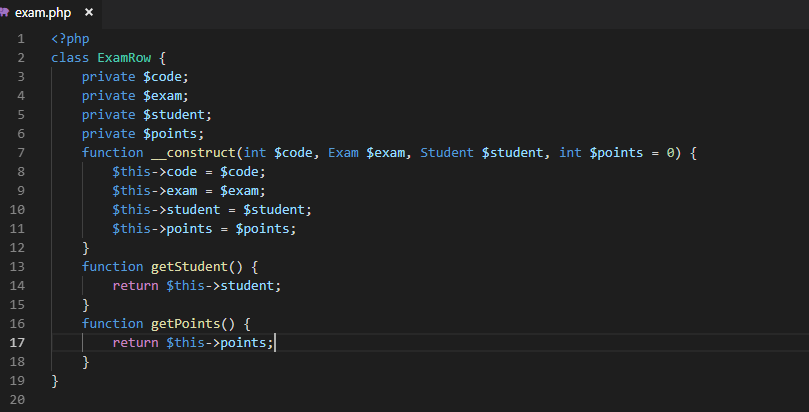
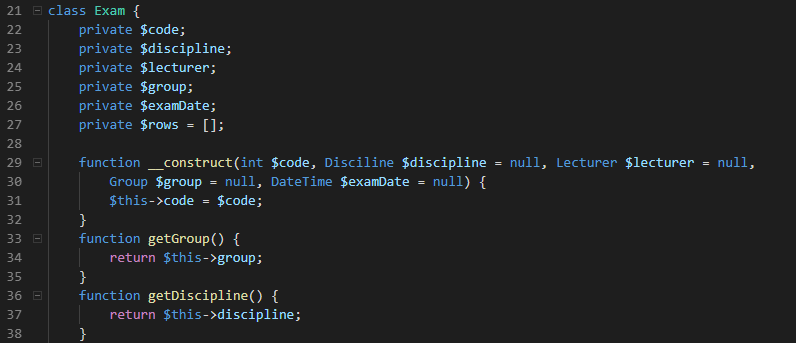
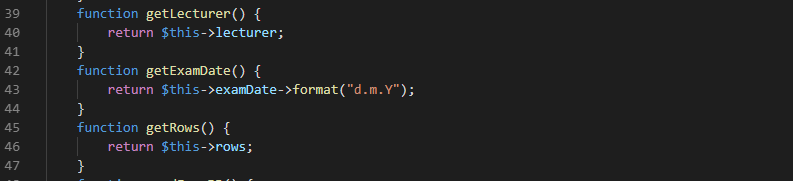


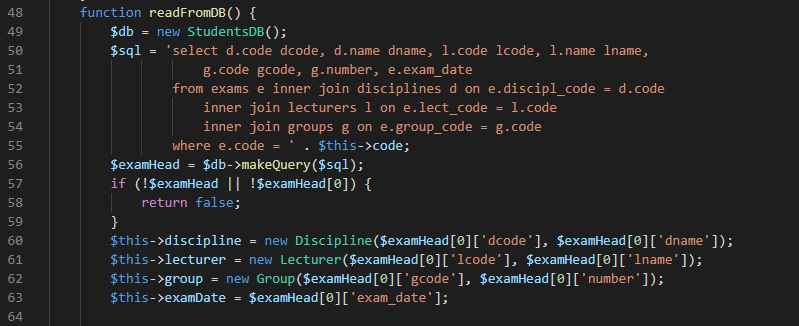
Рис.9. Реалізація класу студент

І, нарешті, реалізуємо класи для окремого рядку та всієї студентської відомості (рис.10). На відміну від всіх попередніх, у класі відомості окрім конструктора та гетерів буде також реалізований метод, що зчитує відомість із БД, використовуючи метод makeQuery класу StudentsDB, наведений на рис.5









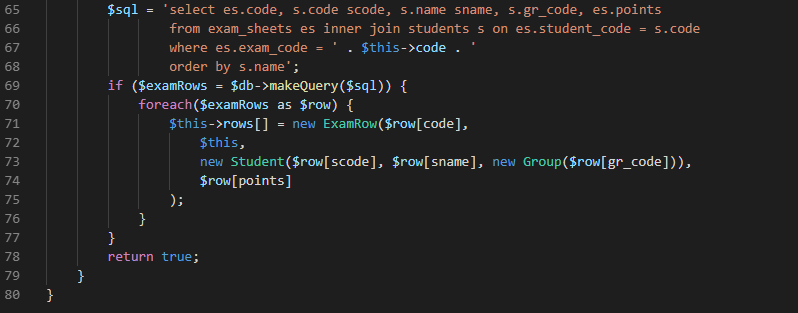


Рис.10. Класи для відомості та її окремого рядку

Для більш зручного підключення всіх реалізованих класів в основному коді створимо у папці «data» файл «autorun.php» (рис.11)

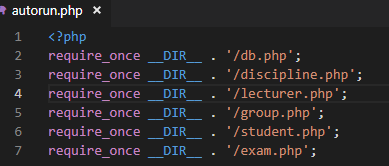
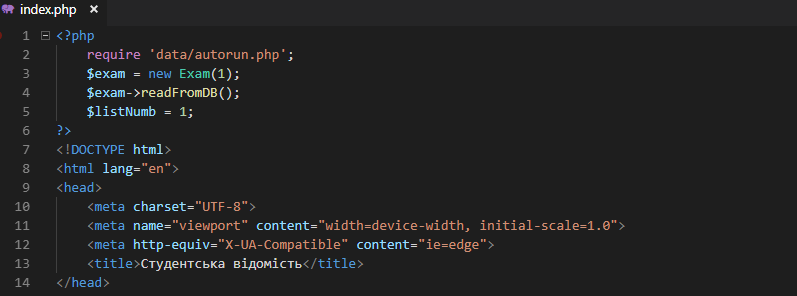
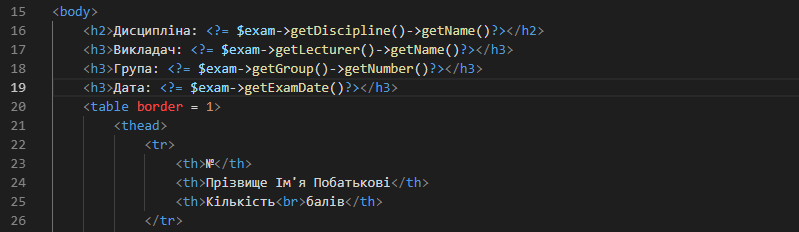


Рис.11. Підключення всіх реалізованих класів в autorun.php

І нарешті створимо сторінку, на якій нарешті відобразимо студентську відомість. Для цього змінимо index.php, де створимо екземпляр класу Exam, виконаємо читання даних із БД за допомогою його методу readFromDB() та відобразимо дані об’єкту на сторінці (рис.12). Поки що не приділяємо уваги стилізації.





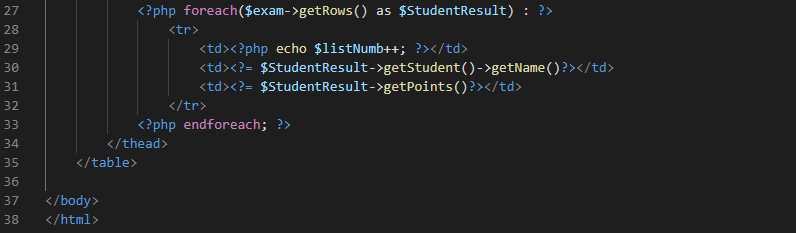


Рис.12. Сторінка, що відображає студентську відомість

Переглянемо результат, оновивши сторінку «http://sql-server/» у нашому браузері (рис.13)

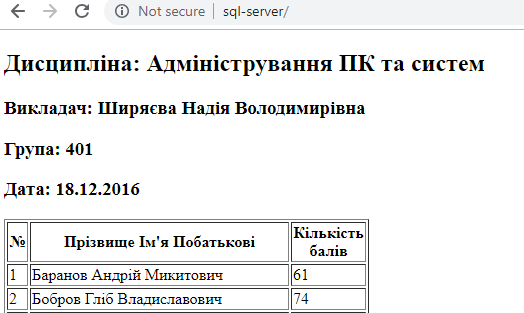


Рис.13. Зовнішній вигляд головної сторінки

1. Наступний крок – реалізація повернення даних студентської відомості у вигляді REST-API у форматі JSON. Для цього створимо у папці проекту директорію «api», а в ній – index.php (рис.14)

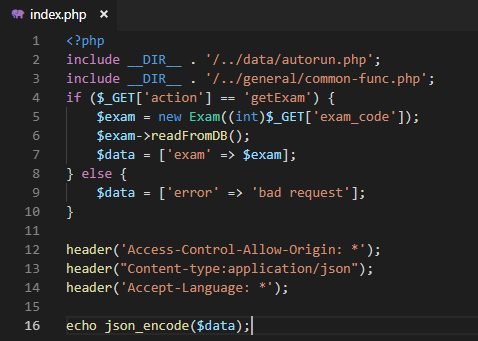


Рис.14. Повернення даних у JSON форматі

Перевіримо результат, набравши в адресному рядку «http://sql-server/api/?action=getExam&exam\_code=1» (рис.15).

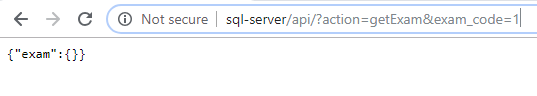


Рис.15. Дані іспиту у JSON форматі.

Бачимо, що об’єкт повертається порожнім. Це відбулося тому, що всі властивості у наших класах приватні, а json\_encode за замовченням повертає лише публічні властивості. Змінити це, можна імплементувавши у класі інтерфейс JsonSerializable та відповідно визначивши метод jsonSerialize(). Отже, імплементуємо інтерфейс (рис.16) та визначивши метод jsonSerialize() для всіх наших класів (рис.17-20)













Рис.16. Імплементація інтерфейсу JsonSerializable

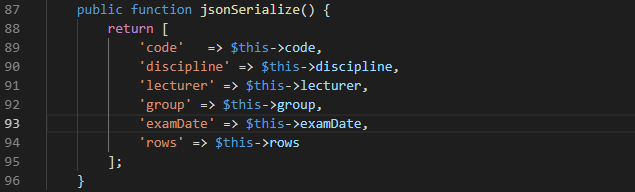


Рис.17. Метод jsonSerialize() для Exam

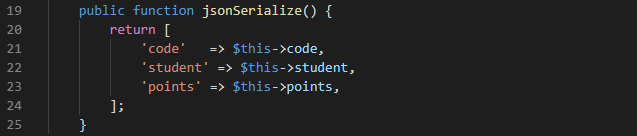


Рис.18. Метод jsonSerialize() для ExamRow

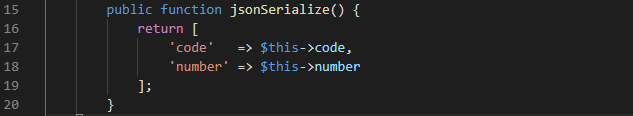


Рис.19. Метод jsonSerialize() для Group

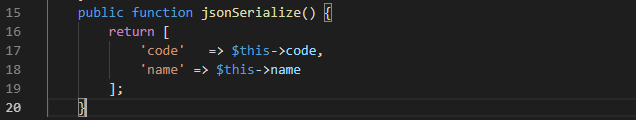


Рис.20. Метод jsonSerialize() для Discipline, Lecturer та Student

Перевіримо результат, повторивши запит «http://sql-server/api/?action=getExam&exam\_code=1» (рис.21).

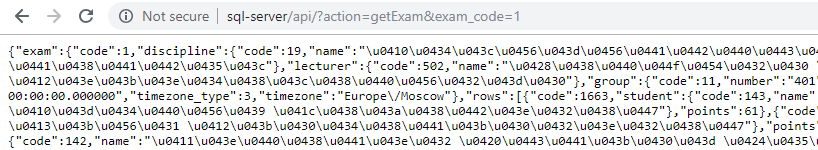


Рис.21. Дані іспиту у JSON форматі після виправлень

Тепер бачимо, що дані повертаються у повному обсязі.

1. Кінець роботи

**Завдання для індивідуального виконання.**

Для БД свого варіанту організувати виведення на екран даних із таблиці, що має у своєму складі мінімум один зовнішній ключ. Реалізувати back-end REST-API сервіс, що повертає ті ж самі дані у форматі JSON за GET запитом на визначений URL.