

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)
на тему:

«LMS система для закладів загальної шкільної освіти»

Виконав студент групи ПЗ-19-3
ОВЕРЧУК Валентина Сергіївна

Керівник роботи:
ПАНАРІНА Ірина Володимирівна

Консультант:
КУЗЬМЕНКО Олександр Вікторович

Рецензент:
ЧИЖМОТРИЯ Олексій Володимирович

Житомир – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ЗАТВЕРДЖУЮ»

В.о. зав. кафедри інженерії
програмного забезпечення
Андрій МОРОЗОВ
«14» лютого 2023 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ОВЕРЧУК Валентина Сергіївна**

Керівник роботи: **ПАНАРІНА Ірина Володимирівна**

Тема роботи: «LMS система для закладів загальної шкільної освіти»,
затверджена Наказом закладу вищої освіти від «14» лютого 2023 р., №70с

Вихідні дані для роботи: **Предмет дослідження: методи та засоби проєктування LMS систем. Об'єкт дослідження: реалізація програмного продукту «LMS система для закладів загальної шкільної освіти».**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Кузьменко О.В.	14.01.2023р.	14.01.2023р.
2	Кузьменко О.В.	22.03.2023р.	22.03.2023р.
3	Кузьменко О.В.	10.05.2023р.	10.05.2023р.

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі	14 січня 2023 - 28 січня 2023	виконано
2	Пошук, огляд та аналіз аналогічних розробок	7 лютого 2023 - 16 лютого 2023	виконано
3	Формулювання технічного завдання	2 березня 2023 – 18 березня 2023	виконано
4	Опрацювання літературних джерел	19 березня 2023- 29 березня 2023	виконано
5	Проектування структури	7 квітня 2023- 15 квітня 2023	виконано
6	Написання програмного коду, проектування бази даних додатку	16 квітня 2023- 25 травня 2023	виконано
7	Тестування	27 травня 2023- 1 червня 2023	виконано
8	Написання пояснювальної записки	2 червня 2023- 10 червня 2023	виконано
9	Попередній захист	5 травня 2023	виконано

РЕФЕРАТ

Випускна робота бакалавра представляє розроблену веб-орієнтовану LMS систему для закладів загальної шкільної освіти та пояснювальної записки.

Пояснювальна записка до кваліфікованої роботи складається з переліку умовних скорочень, вступу, трьох розділів, висновків, списку використаної літератури та додатку.

Текстова частина викладена на 55 сторінках друкованого тексту. Пояснювальна записка має 12 сторінок додатків. Список використаних джерел містить 18 найменувань і займає 2 сторінки. В роботі наведено 25 рисунка та 15 таблиць. Загальний обсяг роботи - 69 сторінок.

В роботі визначені головні завдання для розробки LMS системи, проаналізовано аналоги продукту та методи розв'язання поставленої задачі, обґрунтовано вибір архітектури системи. Представлено алгоритм роботи програмного комплексу, структуру та взаємодію його компонентів та описана програмна реалізація системи.

Ключові слова: БАЗА ДАНИХ, MVC, MYSQL, PHP, Yii2.

					ІПЗ.КР.Б – 121 – 23 – ПЗ			
Зм.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Оверчук В.С.			«LMS система для закладів загальної шкільної освіти» Пояснювальна записка	Літ.	Арк.	Аркушів
Керівник		Панаріна І.В.					2	59
						Житомирська політехніка, група ІПЗ-19-3		
Н. контр.		Тичина О.П.						
Зав. каф.		Морозов А.В.						

ABSTRACT

Bachelor's graduation work is a developed web-oriented LMS system for general school education institutions and an explanatory note.

The explanatory note to the diploma project consists of a list of conditional abbreviations, introduction, three sections, conclusions, a list of used literature and an appendix.

The text part is laid out on 55 pages of printed text. The explanatory note has 12 pages of appendices. The list of used sources contains 18 names and occupies 2 pages. The paper contains 25 pictures and 15 tables. The total amount of work is 69 pages.

The main tasks for the development of the system are defined in the work, analogues of the product and methods for solving the task are analyzed, the choice of the system architecture is justified. The algorithm of the software complex, the structure and interaction of its components are presented and the software implementation of the system is described.

Keywords: DB, MVC, MYSQL, PHP, Yii2.

					ИП3.КР.Б – 121 – 23 – И3	
						3

ЗМІСТ

РЕФЕРАТ.....	6
ABSTRACT	3
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ LMS СИСТЕМИ ДЛЯ ЗАКЛАДІВ ЗАГАЛЬНОЇ ШКІЛЬНОЇ ОСВІТИ.....	9
1.1. ПОСТАНОВКА ЗАДАЧІ	9
1.2. АНАЛІЗ АНАЛОГІВ ПРОГРАМНОГО ПРОДУКТУ.....	10
1.3. ВИБІР АРХІТЕКТУРИ ВЕБ-ОРІЄНТОВАНОЇ СИСТЕМИ	15
1.4. ОБҐРУНТУВАННЯ ВИБОРУ ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ ТА ВИМОГИ ДО АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ.....	17
Висновки до розділу 1.....	18
РОЗДІЛ 2. ПРОЄКТУВАННЯ LMS СИСТЕМИ ДЛЯ ЗАКЛАДІВ ЗАГАЛЬНОЇ ШКІЛЬНОЇ ОСВІТИ.....	19
2.1. ВИЗНАЧЕННЯ ВАРІАНТІВ ВИКОРИСТАННЯ ТА ОБ’ЄКТНО-ОРІЄНТОВАНОЇ СТРУКТУРИ СИСТЕМИ	19
2.2. РОЗРОБКА БАЗИ ДАНИХ СИСТЕМИ.....	23
2.3. ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ АЛГОРИТМІВ РОБОТИ СИСТЕМИ.....	29
2.4. РЕАЛІЗАЦІЯ ДОДАТКА «LMS СИСТЕМА ДЛЯ ЗАКЛАДІВ ЗАГАЛЬНОЇ ШКІЛЬНОЇ ОСВІТИ»	33
Висновки до розділу 2.....	39
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ LMS СИСТЕМИ ДЛЯ ЗАКЛАДІВ ЗАГАЛЬНОЇ ШКІЛЬНОЇ ОСВІТИ.....	40
3.1. ПОРЯДОК ВСТАНОВЛЕННЯ ТА НАЛАШТУВАННЯ ПАРАМЕТРІВ СИСТЕМИ. .	40
3.2. СТРУКТУРА ІНТЕРФЕЙСУ LMS СИСТЕМИ ЗАКЛАДІВ ЗАГАЛЬНОЇ ШКІЛЬНОЇ ОСВІТИ	41

3.3. ТЕСТУВАННЯ РОБОТИ ПРОГРАМНОГО КОМПЛЕКСУ	49
Висновки до розділу 3.....	51
ВИСНОВКИ	52
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	54
ДОДАТКИ.....	56

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних.

LMS – Learning Management System (з англ. система керування навчання).

CRUD – Create, Read, Update, Delete: операції обробки даних в базі даних.

					ІПЗ.КР.Б – 121 – 23 – ПЗ	
						6

ВСТУП

Актуальність теми. Створення системи управління навчанням (LMS) для шкіл стає все більш актуальним у сучасному освітньому просторі. LMS - це програмне забезпечення, яке дозволяє вчителям керувати та відстежувати різні аспекти навчального процесу. Він слугує централізованою платформою, яка покращує комунікацію, співпрацю та загальну ефективність в освітній екосистемі.

LMS пропонує численні переваги для шкіл, учнів та вчителів. Перш за все, це гнучке та доступне навчальне середовище. Учні можуть отримати доступ до уроків, завдань і ресурсів з будь-якого місця і в будь-який час, якщо у них є підключення до Інтернету. Така гнучкість сприяє персоналізованому навчанню в індивідуальному темпі, дозволяючи учням навчатися у зручний для них час і повертатися до контенту за потреби. Школи можуть збирати та аналізувати дані про прогрес, залученість та успішність учнів, надаючи цінну інформацію для вчителів та адміністраторів. Ці дані можуть бути використані для розробки навчальних стратегій, визначення сфер, які потребують покращення, та цілеспрямованого втручання для підтримки учнів, які відчують труднощі в навчанні. Використовуючи функції аналітики та звітності, школи можуть підвищити ефективність викладання, оптимізувати розподіл ресурсів і, зрештою, покращити результати навчання учнів [12].

Метою випускної роботи є проєктування та розробка LMS системи для закладів загальної шкільної освіти. Дослідити стандартні функціональні особливості сучасних LMS-систем.

Для досягнення цієї мети в даній роботі будуть розглянуті наступні розділи:

- Аналіз основних функціональних особливостей сучасних LMS-систем, де буде проаналізовано основні функції, пов'язані з програмним забезпеченням для керування навчанням, а також

оцінено наявні програмні рішення, доступні на ринку для задоволення цих потреб;

- Вимоги до LMS системи для закладів загальної шкільної освіти, де будуть визначені та обговорені специфічні вимоги та функціональні можливості, необхідні для створення системи;
- Проєктування архітектури LMS системи для закладів загальної шкільної освіти, де буде окреслено архітектурний дизайн запропонованої системи, включаючи опис архітектури, визначення ключових модулів, діаграму класів та модель користувацького інтерфейсу.

Об’єктом дослідження є реалізація програмного продукту «LMS система для закладів загальної шкільної освіти».

Предметом дослідження є методи та засоби проєктування LMS систем.

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ LMS СИСТЕМИ ДЛЯ ЗАКЛАДІВ ЗАГАЛЬНОЇ ШКІЛЬНОЇ ОСВІТИ

1.1. Постановка задачі

Основною метою розробки системи є покращити навчальний процес та результати навчання учнів. LMS слугує централізованою платформою, яка сприяє ефективній комунікації, співпраці та доступу до навчальних ресурсів. Вона має на меті забезпечити гнучке та доступне навчальне середовище, спростити адміністративні завдання. Адміністратори можуть реєструвати учнів та вчителів та контролювати розподіл предметів, а учні можуть переглядати та отримувати доступ до призначених їм уроків.

У випускній кваліфікаційній роботі необхідно розробити веб-систему, яка буде містити такий функціонал:

Функції клієнтської частини для учня:

- Перегляд власного профілю, редагування певних полів;
- Сортування, фільтрація та пошук предметів, уроків;
- Проходження навчальних матеріалів;
- Перегляд активних та пройдених матеріалів;
- Перегляд власних досягнень.

Функції клієнтської частини для вчителя:

- Перегляд власного профілю, редагування певних полів;
- Перегляду профілю учня;
- Створення, редагування та видалення уроків;
- Перегляд результатів проходження матеріалу, як окремого учня, так і цілого класу;
- Завантаження результатів.

Функції адміністративної частини:

- Реєстрація Учня та Вчителя;
- Створення, редагування та видалення класів;
- Додавання учнів до класів;
- Створення, редагування та видалення предметів;
- Архівування БД;
- Робота з каталогом уроків: CRUD для уроків, фільтрації та сортування даних;
- CRUD для домашніх завдань, фільтрації та сортування даних.

Результатом реалізації поставленого завдання є функціональна LMS система для закладів загальної шкільної освіти.

1.2. Аналіз аналогів програмного продукту

Розглянемо наявні аналоги, які найбільше подібні до теми кваліфікаційної роботи: Google Classroom, Moodle, Schoology, наведемо їх можливості та порівняємо їх. Основні фактори для порівняння будуть включати легкість використання як для вчителів, так і для учнів, можливості для комунікації, оцінювання успішності, можливості управління на рівні школи.

Google Classroom. Google Classroom - це широко використовувана система керування навчанням, яка дозволяє вчителям розподіляти завдання, спілкуватися з учнями та керувати своїми класами онлайн. Вона надає такі функції, як спільний доступ до документів, подання завдань та обговорення в класі. Вона легко інтегрується з іншими інструментами Google, що робить її популярною серед шкіл, які використовують G Suite for Education [13]. Доступ до платформи - <https://classroom.google.com/>

Переваги:

- Безплатність - Google Classroom є безплатним інструментом, який можна використовувати безплатно з будь-яким акаунтом Google;

- Легкість використання - інтерфейс Google Classroom досить інтуїтивно зрозумілий та простий у використанні, навіть для тих, хто не має досвіду роботи з цифровими інструментами;
- Інтеграція з Google Drive - Google Classroom інтегрований з Google Drive, що дозволяє легко додавати та ділитися документами, презентаціями та іншими матеріалами зі своїми студентами.

Недоліки:

- Має обмежені можливості статистики та звітності, що може бути недостатньо для більш складних проєктів;
- Проблеми з конфіденційністю - хоча Google заявляє, що забезпечує конфіденційність та захист даних. Наприклад, компанія може збирати та аналізувати дані про користувачів для покращення своїх продуктів і послуг, що може порушити приватність користувачів.

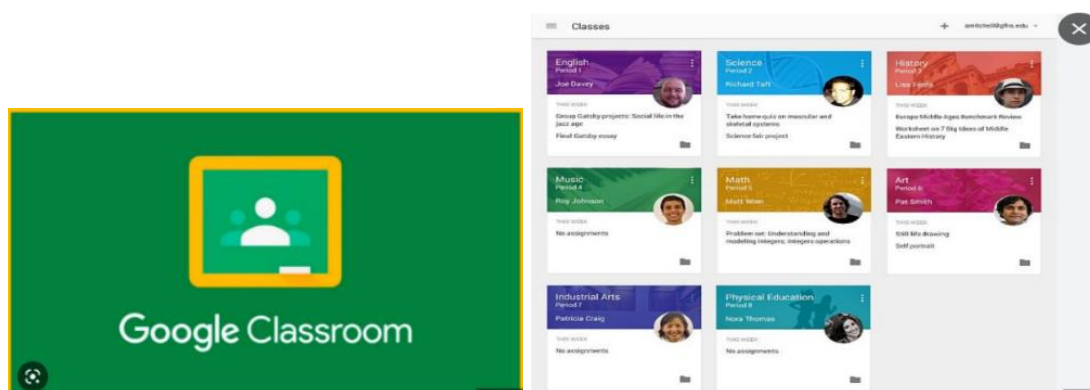


Рис. 1.1 - Логотип та сторінка класів у Google Classroom

Moodle. Moodle - це відкрите програмне забезпечення для управління навчанням та електронного навчання. Його створено з метою надання платформи для навчання в інтернет-середовищі, що підтримує інтерактивність, співпрацю та розширюваність. Moodle широко використовується у школах, університетах, компаніях та інших освітніх організаціях [14]. Доступ до платформи - <https://moodle.org/login/index.php>

Переваги:

- Moodle є відкритою системою з відкритим вихідним кодом, що означає, що ви можете отримати доступ до програмного коду,

змінювати його та адаптувати до своїх потреб. Це надає гнучкість і можливість налаштування системи під конкретні вимоги вашої навчальної організації;

- Moodle дозволяє легко організовувати курси навчання, включаючи структуру курсу, матеріали, завдання, терміни, оцінки та звіти про прогрес учнів. Ви можете встановлювати права доступу до курсів, керувати групами студентів та аналізувати прогрес кожного учня;
- Moodle пропонує інструменти для співпраці та соціального навчання, такі як форуми, блоги, чати, спільні завдання тощо. Вони сприяють активній взаємодії між учнями, обміну ідеями та спільному конструюванню знань.

Недоліки:

- Налаштування Moodle може вимагати певних технічних навичок та розуміння системи. Це може бути складно для некваліфікованих користувачів або адміністраторів;
- Moodle вимагає відповідного хостинг-середовища для ефективної роботи, особливо при великому обсязі користувачів та курсів. Це може потребувати додаткових витрат на хостингові послуги.

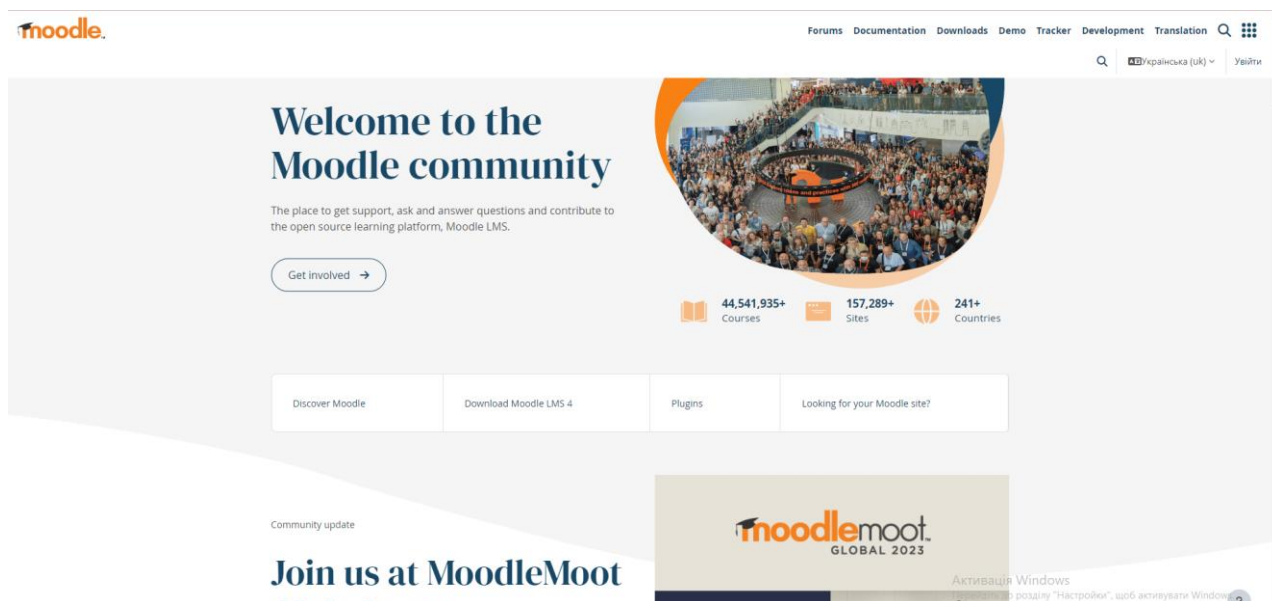


Рис. 1.2 - Початкова сторінка застосунку Moodle

Schoology. Schoology є потужною системою управління навчанням, яка ставить на перше місце співпрацю та взаємодію між учнями та вчителями. Вона пропонує широкий спектр інструментів та можливостей, щоб забезпечити ефективне навчання та залучення досягнень учнів.

Одним з ключових переваг Schoology є можливість створювати курси, які відповідають потребам навчальної програми. Вчителі можуть легко створювати зміст, завдання та матеріали для кожного курсу, що допомагає створити структуроване та цікаве навчальне середовище [15]. Доступ до платформи - <http://www.schoology.com/>

Переваги:

- Простий у використанні інтерфейс, що полегшує навігацію та використання для вчителів, учнів та батьків;
- Можливість вчителям створювати та оцінювати завдання, вести журнал успішності учнів, надавати звіти та аналітику щодо успішності предметів;
- Платформа може легко інтегруватися з різноманітними зовнішніми системами, такими як Google Apps, Office 365, LTI-сумісні додатки та інші, що дозволяє розширити функціональні можливості;
- Schoology має мобільні додатки для iOS та Android, що дозволяють учням, вчителям та батькам отримувати доступ до предметів та матеріалів зі своїх мобільних пристроїв.

Недоліки:

- Обмежені можливості безплатної версії: Деякі додаткові функції та інтеграції доступні лише у платній версії Schoology, що може бути обмежувальним для деяких користувачів;
- Налаштування деяких аспектів Schoology може вимагати певної технічної експертизи, особливо для адміністраторів системи;

- Використання Schoology може бути пов'язане з платою за ліцензію або підписку, особливо для шкіл або установ, які потребують розширеного функціонала.

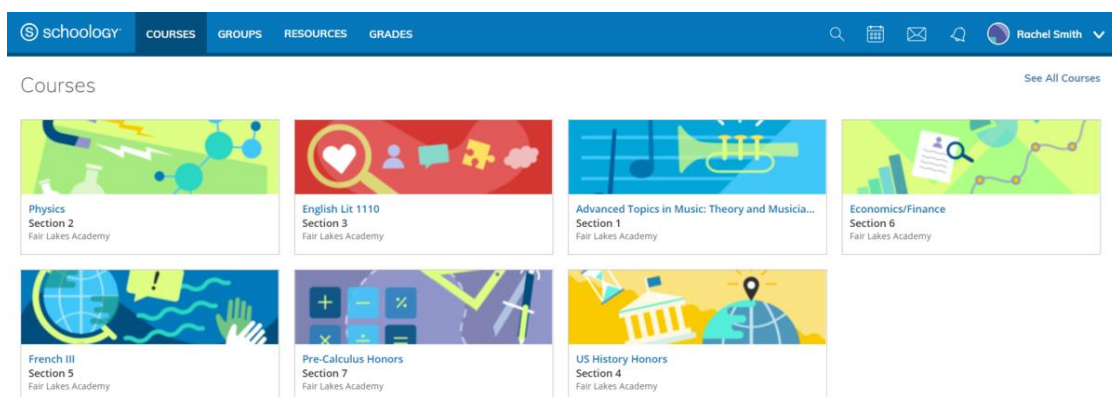


Рис. 1.3 - Інтерфейс платформи Schoology

Таблиця 1.1

Порівняння функцій LMS-систем

Функції/Платформи	Google Classroom	Moodle	Schoology
Створення уроків	+	+	+
Додавання вчителів/учнів	+	+	+
Домашнє завдання	+	+	+
Можливість редагування документів	+	+	+
Система відстеження успішності студентів	-	+	+
Інтеграція зі сторонніми сервісами	+	+	+
Зручність	+	-	+
Легкий старт	+	-	+
Безплатність	+	+	-

Після ретельного огляду та аналізу трьох аналогів освітніх платформ, виявлено, що кожен з них має свої переваги та недоліки. Google Classroom, попри свою безплатність, обмежує можливість відстеження успішності учнів. Moodle

також пропонує безплатне використання, але його складна архітектура може створювати деякі труднощі.

Враховуючи цю розбіжність і потреби вдосконалення та спрощення процесу навчання в школах, ми маємо намір розробити нашу власну систему управління навчанням (LMS). Ми прагнемо створити продукт, який поєднає зручний інтерфейс, безплатне використання та необхідний функціонал, щоб задовольнити потреби користувачів.

Основні характеристики нашої LMS системи будуть включати:

1. Зручний інтерфейс: Ми розробимо інтуїтивно зрозумілий та легкий у використанні інтерфейс, щоб навчання стало приємним та зрозумілим для всіх користувачів.
2. Безплатність: Ми віримо у доступність освіти для всіх, тому наша LMS система буде повністю безплатною для використання.
3. Основний функціонал: Ми забезпечимо можливість реєстрації учнів та вчителів, створення класів, додавання, перегляду, редагування та видалення предметів та уроків, а також можливість перевірки домашніх завдань.
4. Додатковий функціонал: Ми також розглянемо можливість впровадження статистики та рейтингу навчання учнів, що допоможе вчителям та учням в оцінці та вдосконаленні свого прогресу.

Наша мета - створити продукт, який задовольнятиме потреби користувачів та сприятиме ефективному навчанню у школах. Ми прагнемо покращити процес навчання та зробити його доступним для всіх учасників освітнього процесу.

1.3. Вибір архітектури веб-орієнтованої системи

Для розробки LMS системи ми використали MVC [1] (Model-View-Controller) архітектуру з метою ефективного розділення логіки програми від її інтерфейсу та забезпечення повторного використання компонентів.

MVC є потужним патерном проєктування веб-додатків, який складається з трьох ключових компонентів. При застосуванні MVC, програма розбивається на наступні частини:

1. Модель (Model): Цей компонент відповідає за управління даними та бізнес-логікою програми. Він містить в себе всю інформацію, необхідну для обробки та збереження даних.
2. Представлення (View): Представлення відповідає за візуальне відображення інтерфейсу користувача. Воно забезпечує відображення даних з моделі та інтерактивність для користувача.
3. Контролер (Controller): Контролер виконує роль посередника між моделлю та представленням. Він обробляє вхідні запити від користувача, оновлює модель та оновлює відображення, щоб відобразити зміни.

Застосування MVC дозволяє нам здійснювати зміни в одному компоненті з мінімальним впливом на інші компоненти або навіть без будь-якого впливу. Це сприяє підтримці коду, полегшує розширення та розвиток системи, а також сприяє повторному використанню компонентів.

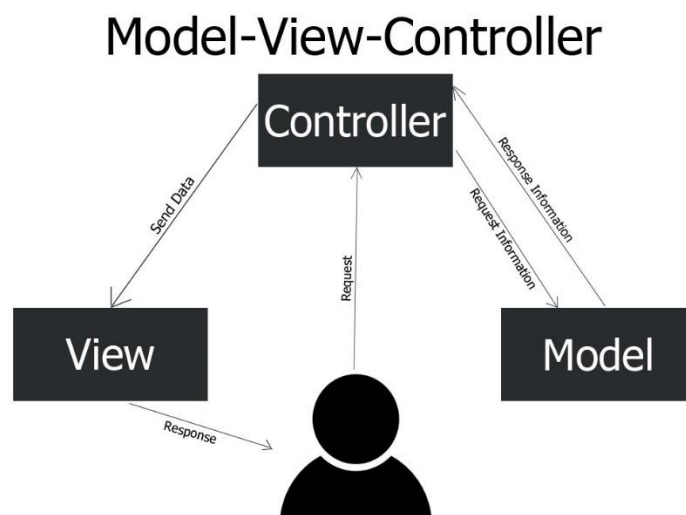


Рис. 1.4 - Загальний алгоритм роботи MVC

Зосередимося на циклі обробки запиту, оскільки клієнтсько-серверна архітектура [2] передбачає такий процес. Після того, як сервер отримує запит від користувача, формується функція для виконання запитів до бази даних та

обробки даних сесії. Далі перевіряється, чи має користувач доступ до запитованої сторінки. Якщо доступ відсутній, користувач буде перенаправлений на сторінку помилки. Якщо ж доступ є, запит оброблюється. У разі необхідності, сервер звертається до бази даних та обробляє отримані дані. Після обробки запиту, сервер формує відповідне представлення і надсилає згенерований HTML-код користувачеві. У розробці передбачено, що користувач буде взаємодіяти з сервером лише через графічний інтерфейс, що відповідає патерну MVC.

1.4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

У цьому проєкті було обрано Open Server як локальне середовище розробки та PHP Yii2 як фреймворк для створення додатка та система керування базами даних MySQL.

Open Server [3] надає комплексну та зручну платформу для локальної розробки. Він пропонує повний набір програмного забезпечення для веб-серверів, включаючи Apache, MySQL і PHP, об'єднаних разом у простий в установці пакет. За допомогою Open Server розробники можуть швидко налаштувати локальне середовище розробки без необхідності індивідуального встановлення та налаштування окремих компонентів. Це значно скорочує час і складність налаштування, дозволяючи розробникам зосередитися на розробці додатків.

Вибір PHP-фреймворку Yii2 [4] обґрунтований наступними факторами. Перш за все, це використання архітектурного шаблону проєктування MVC, який забезпечує чітке розмежування між моделлю даних, представленням та логікою взаємодії користувача з системою. Це сприяє покращенню підтримки коду і забезпечує зручність розробки. Yii2 надає широкі можливості для розробки різних типів застосунків, включаючи веб-сайти, інформаційні системи та RESTful API [5]. Фреймворк має багатофункціональність, таку як робота з користувачами, використання ORM ActiveRecord, можливість автоматичної

генерації коду, підтримку кешування та механізм доступу на основі RBAC [6]. Це спрощує розробку і забезпечує ефективну роботу з даними. Фреймворк має зручний візуальний генератор моделей, контролерів і CRUD, що спрощує створення необхідного коду. Загалом, Yii2 володіє простотою розробки, низьким стартом розробки та широким набором функцій, що робить його ідеальним вибором для проєкту.

Для розробки веб-інтерфейсу було використано мову розмітки сторінок HTML5 та стильову мову CSS3. Для забезпечення гнучкості та швидкості розробки використано CSS-фреймворк Bootstrap4 [16]. Для backend-частини – розширення AdminLTE.

Клієнтська частина повинна забезпечувати максимальну сумісність з різними веб-браузерами, такими як Mozilla, FireFox та Google Chrome, і працювати на різних платформах, забезпечуючи зручний та надійний доступ до основних функцій.

Висновки до розділу 1

Під час написання першого розділу було проведено детальний аналіз предметної області, що дозволило визначити основні аспекти майбутньої LMS системи для закладів загальної шкільної освіти.

В результаті огляду наявних аналогів було вирішено, що веб-орієнтованій системі необхідний зручний та доступний для користувачів функціонал, а також унікальний інтерфейс.

Для розробки системи було обрано та обґрунтовано архітектурний шаблон проєктування MVC, використання PHP-фреймворка Yii2, а також був складений перелік засобів розробки для сучасних веб-орієнтованих систем. Для зберігання даних було обрано реляційну СУБД - MySQL.

На стороні клієнта для створення інтерфейсних сторінок і стилізації використовуються стандартні веб-технології, такі як HTML5 та CSS (бібліотека Bootstrap 4). Було підкреслено важливість сумісності з різними веб-браузерами та платформами, щоб забезпечити зручний та надійний доступ.

РОЗДІЛ 2. ПРОЄКТУВАННЯ LMS СИСТЕМИ ДЛЯ ЗАКЛАДІВ ЗАГАЛЬНОЇ ШКІЛЬНОЇ ОСВІТИ

2.1. Визначення варіантів використання та об'єктно-орієнтованої структури системи

Цей розділ надає детальний огляд варіантів використання, вимог та специфікацій, які програмний продукт повинен виконувати для забезпечення ефективного виконання своїх функцій. Це включає в себе відповідність всім відповідним стандартам, специфікаціям та іншим формальним документам, які визначають його роботу.

Система складається з двох основних модулів: клієнтської та адміністративної частин. Клієнтська частина забезпечує інтерфейс для взаємодії з користувачами, в той час, як адміністративна частина дозволяє управляти системою.

Система передбачає наявність різних користувачів, кожен з яких виконує свою унікальну роль. Ці ролі включають учнів, вчителів та адміністраторів. Учні та вчителі в основному використовують клієнтську частину системи, яка надає їм доступ до необхідного функціоналу. З іншого боку, адміністратори мають доступ до адміністративної частини, де вони можуть управляти системою.

Окрім того, в цьому розділі описані основні можливості системи, її обмеження та показники якості. Це включає в себе розгляд функціональності, яку система може надати, обмежень, які вона може мати, та якість роботи, яку вона може забезпечити.

Бізнес-вимоги:

1. Основні цілі: проєкт створюється з метою спрощення навчального процесу та взаємодії вчителя та учня;
2. Представлення проєкту: проєкт буде реалізовано у вигляді сайту, що містить структуровану інформацію по даній тематичній області.

Вимоги зовнішніх користувачів (Учень)

1. Можливість переглядати власний профіль, редагувати певні поля;

2. Можливість сортування, фільтрації та пошуку матеріалів;
3. Можливість проходження навчальних матеріалів;
4. Можливість проходження домашнього завдання;
5. Можливість перегляду виконаних та невиконаних завдань;
6. Можливість перегляду власних досягнень.

Вимоги зовнішніх користувачів (Вчитель)

1. Можливість переглядати власний профіль, редагувати певні поля;
2. Можливість перегляду профілю учня;
3. Можливість створення, редагування та видалення уроків;
4. Можливість перегляду результатів надісланих робіт на перевірку, як окремого учня, так і цілого класу;
5. Завантаження результатів;
6. Можливість перегляду статистики.

Вимоги внутрішніх користувачів (Адміністратор):

1. Можливість зміни будь-якої інформації у БД;
2. Можливість архівування БД;
3. Можливість реєструвати учнів та вчителів;
4. Можливість створення класів;
5. Можливість додавання учнів до класів;
6. Можливість створення предметів.

Характеристика об'єкта комп'ютеризації:

Користувач додатка має достатньо можливостей та зручно розроблений інтерфейс для взаємодії з учнями чи вчителями та адміністрацією.

Функціональні вимоги:

1. Авторизація в системі: В системі повинна бути представлена можливість створення облікового запису користувача та присвоєння йому відповідної ролі (учень, вчитель, адміністратор).
2. Можливість збереження інформації: Система повинна зберігати інформацію і надавати можливість керувати нею.

Нефункціональні вимоги:

1. Сприйняття

- Час, потрібний для навчання роботи з веб-додаток для звичайних користувачів — 30 хвилин, для досвідчених — 5 хвилин;
- Інтерфейс LMS системи повинен бути інтуїтивно зрозумілим та зручним для користувача, не вимагати від нього додаткової підготовки.

2. Надійність

- Доступність – час, потрібний для обслуговування системи не повинен перевищувати 10% від загального часу роботи;
- Середній час безперервної роботи – 20 робочих днів;
- Максимальна норма помилок та дефектів в роботі системи – 1 помилка на 1000 запитів користувача.

3. Продуктивність

- Система для навчання повинна підтримувати, як найменше 100 одночасно працюючих користувачів.

4. Можливість експлуатації

- Масштабування – система повинна мати можливість збільшувати продуктивність, зі збільшенням користувачів таким чином, щоб це ніяк негативно не відобразилося на роботі.

Системні вимоги:

1. Вимоги до середовища виконання:

Система повинна задовольняти зазначеним вимогам на комп'ютері в наступній мінімальній комплектації:

- підтримка браузера *Google Chrome* версії 45+ або
- підтримка браузера *Mozilla Firefox* версії 38+ або
- підтримка браузера *Microsoft Edge* версії 12+ або
- підтримка браузера *Safari* версії 9+

- операційна система Windows 7 або вище, Linux 8.10 та вище, MacOS 10.6 Snow Leopard та вище;
- 32-розрядний (x86) або 64-розрядний (x64) процесор із тактовою частотою 1 ГГц або вище;
- 1 ГБ (для 32-розрядних процесорів) або 2 ГБ (для 64-розрядних процесорів) ОЗУ;
- 16 ГБ (для 32-розрядної системи) або 20 ГБ (для 64-розрядної системи) вільного місця на жорсткому диску.

2. Вимоги до СУБД та доступу до даних:

- У ядрі системи повинна бути представлена СУБД реляційного доступу.

Аналіз вимог користувачів дав підстави для визначення варіантів використання LMS системи для закладів загальної шкільної освіти. На рис. 2.1 зображено побудовану діаграму варіантів використання.

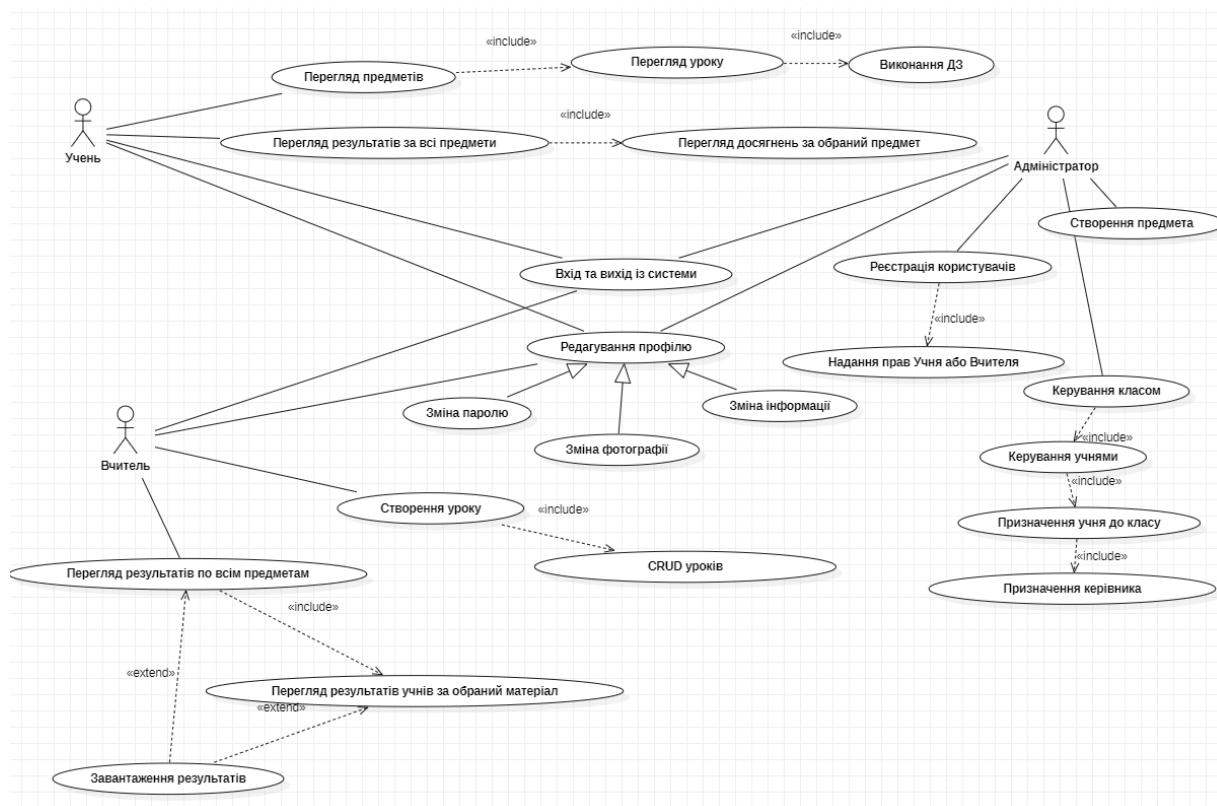


Рис. 2.1 - Діаграма варіантів використання LMS системи для закладів загальної шкільної освіти

2.2. Розробка бази даних системи

В створюваному веб-додатку буде використана база даних MySQL. Бази даних MySQL широко використовуються в сучасних електронних пристроях для зберігання даних різних додатків [7].

Реалізована база даних складається з 13 таблиць, які містять у собі всі дані для роботи програми.

- 8 таблиць даних моделей системи: user, school_classes, class_student, subject, lesson, subject_class, homework, homework_submission.
- 4 таблиці даних авторизації: auth_item, auth_item_child, auth_assignment, auth_rule
- 1 службова таблиця: migration

Найменування та призначення яких представлено у таблиці 2.1.

Таблиця 2.1

Інформація про таблиці бази даних

Назва таблиці	Призначення
user	Таблиця містить зареєстрованих користувачів
school_classes	Таблиця містить класи
class_student	Таблиця містить відношення учнів до класу
subject	Таблиця містить предмети
lesson	Таблиця містить уроки
subject_class	Таблиця містить відношення предмета до класу
homework	Таблиця містить домашні завдання
homework_submission	Таблиця містить відправленні рішення
auth_item	Таблиця містить дані про ролі та дозволи

auth_item_child	Таблиця містить підпорядкування ролей та дозволів
auth_assignment	Таблиця містить відношення користувачів до ролей
auth_rule	Таблиця містить правила
migration	Таблиця містить інформацію про міграцію даних

На рисунку 2.2 представлена діаграма бази даних.

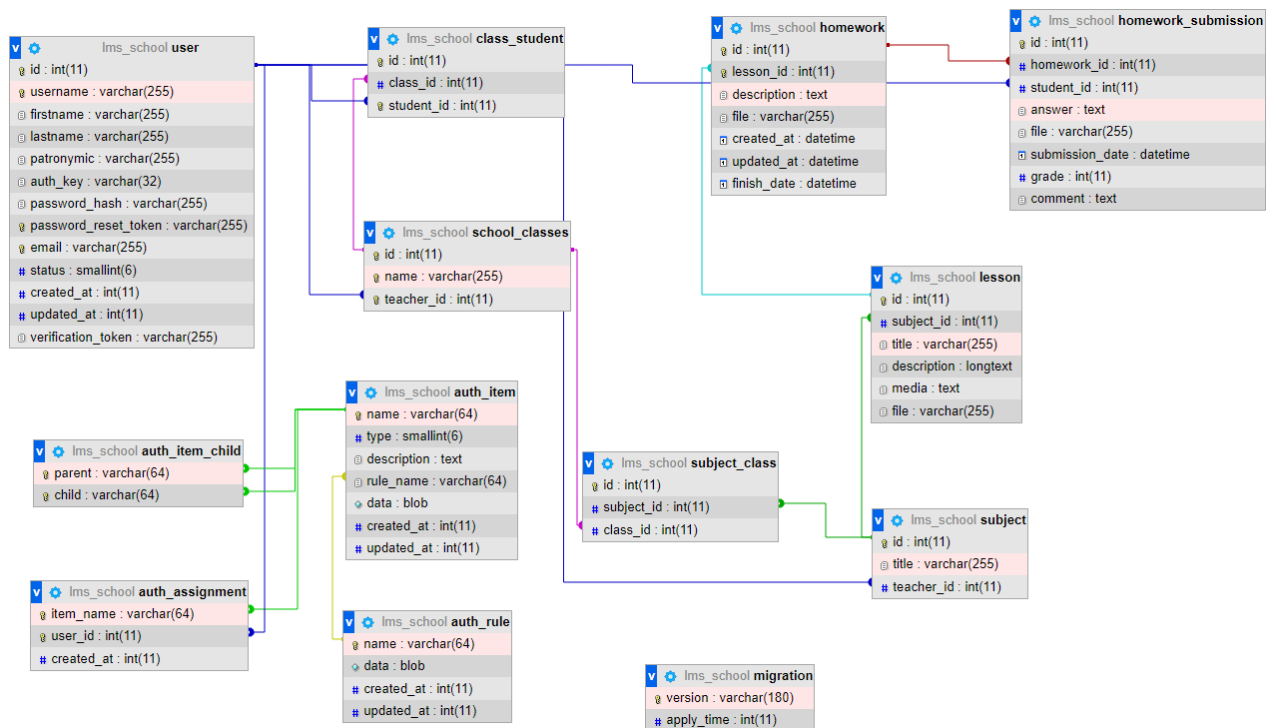


Рис. 2.2 - Діаграма бази даних

Далі наведемо опис таблиць бази даних (таблиці 2.2-2.14).

Таблиця 2.2

Опис полів таблиці «auth_assignment»

Назва	Тип даних	PK	FK	Опис поля
item_name	varchar(64)	+	-	Назва ролі
user_id	varchar(64)	-	+	Ідентифікатор користувача

Продовження таблиці 2.2

created_at	int(11)	-	-	Дата привласнення ролі користувачу
------------	---------	---	---	------------------------------------

Таблиця 2.3

Опис полів таблиці «auth_item»

Назва	Тип даних	PK	FK	Опис поля
name	varchar(64)	+	-	Ім'я ролі
type	smallint(6)	-	-	Тип. Якщо 1 – це роль, якщо 2 – дозвіл
description	text	-	-	Опис ролі чи дозволу
rule_name	varchar(64)	-	+	Ім'я правила, яке застосовується до даного дозволу чи ролі
data	blob	-	-	Додаткові дані в серіалізованому вигляді
created_at	int(11)	-	-	Дата створення
updated_at	int(11)	-	-	Дата оновлення

Таблиця 2.4

Опис полів таблиці «auth_item_child»

Назва	Тип даних	PK	FK	Опис поля
Parent	varchar(64)	-	+	Ім'я батьківського дозволу чи ролі
Child	varchar(64)	-	+	Ім'я дочірнього дозволу чи ролі

Таблиця 2.5

Опис полів таблиці «auth_rule»

Назва	Тип даних	PK	FK	Опис поля
name	varchar(64)	+	-	Ім'я правила

Продовження таблиці 2.5

data	blob	-	-	Серіалізований об'єкт yii\rbac\Rule
created_at	int(11)	-	-	Дата створення
updated_at	int(11)	-	-	Дата оновлення

Таблиця 2.6

Опис полів таблиці «migration»

Назва	Тип даних	PK	FK	Опис поля
version	varchar(180)	-	-	Версія міграції
apply_time	int(11)	-	-	Дата створення

Таблиця 2.7

Опис полів таблиці «user»

Назва	Тип даних	PK	FK	Опис поля
id	int(11)	+	-	Ідентифікатор
username	varchar(255)	-	-	Нікнейм
firstname	varchar(255)	-	-	Ім'я
lastname	varchar(255)	-	-	Прізвище
patronymic	varchar(255)	-	-	Наявність вищої освіти
auth_key	varchar(32)	-	-	Автентифікаційний код
password_hash	varchar(255)	-	+	Захешований пароль
password_reset_token	varchar(255)	-	-	Токен для відновлення пароля
email	varchar(255)	-	-	Електронна пошта
status	Smallint(6)	-	-	Статус
created_at	Int(11)	-	-	Дата створення
updated_at	Int(11)	-	-	Дата оновлення

Продовження таблиці 2.7

verification_token	varchar(255)	-	-	Токен для верифікації
Image	varchar(255)	-	-	Ім'я картинки з профілю
phone	varchar(255)	-	-	Номер телефону

Таблиця 2.8

Опис полів таблиці «school_classes»

Назва	Тип даних	PK	FK	Опис поля
id	int(11)	+	-	Ідентифікатор
name	varchar(255)	-	-	Назва класу
teacher_id	int(11)	-	+	Ідентифікатор вчителя

Таблиця 2.9

Опис полів таблиці «class_student»

Назва	Тип даних	PK	FK	Опис поля
id	int(11)	+	-	Ідентифікатор
class_id	int(11)	-	+	Ідентифікатор класу
student_id	int(11)	-	+	Ідентифікатор учня

Таблиця 2.10

Опис полів таблиці «subject»

Назва	Тип даних	PK	FK	Опис поля
id	int(11)	+	-	Ідентифікатор
title	varchar(255)	-	-	Назва предмета
teacher_id	int(11)	-	+	Ідентифікатор вчителя

Таблиця 2.11

Опис полів таблиці «subject_class»

Назва	Тип даних	PK	FK	Опис поля
-------	-----------	----	----	-----------

Продовження таблиці 2.11

id	nt(11)	+	-	Ідентифікатор
subject_id	int(11)	-	+	Ідентифікатор предмета
class_id	int(11)	-	+	Ідентифікатор класу

Таблиця 2.12

Опис полів таблиці «lesson»

Назва	Тип даних	PK	FK	Опис поля
id	int(11)	+	-	Ідентифікатор
subject_id	int(11)	-	+	Ідентифікатор предмета
title	varchar(255)	-	-	Назва уроку
description	longtext	-	-	Опис
file	varchar(255)	-	-	Ім'я файлу

Таблиця 2.13

Опис полів таблиці «homework»

Назва	Тип даних	PK	FK	Опис поля
id	int(11)	+	-	Ідентифікатор
lesson_id	int(11)	-	+	Ідентифікатор уроку
description	text	-	-	Опис
file	varchar(255)	-	-	Ім'я файлу
created_at	datetime	-	-	Дата створення
updated_at	datetime	-	-	Дата оновлення
finish_date	datetime	-	-	Остання дата відправки рішення

Таблиця 2.14

Опис полів таблиці «homework_submission»

Назва	Тип даних	PK	FK	Опис поля
id	int(11)	+	-	Ідентифікатор

homework_id	int(11)	-	+	Ідентифікатор домашнього завдання
student_id	int(11)	-	+	Ідентифікатор учня
answer	text	-	-	Відповідь
file	varchar(255)	-	-	Ім'я файлу
submission_date	datetime	-	-	Дата відправки
grade	int(11)	-	-	Оцінка
comment	text	-	-	Коментар

2.3. Проектування та реалізація алгоритмів роботи системи

В даному розділі буде побудована діаграма діяльності, яка є важливим інструментом на різних етапах розробки програмного забезпечення. Діаграми діяльності, схожі на блок-схеми, є популярними в інформаційних технологіях.

UML-діаграми діяльності дозволяють детально відобразити конкретні сценарії використання, представляючи потік дій у системі. Вони також можуть використовуватись для моделювання бізнес-процесів, де вони допомагають визначити послідовність подій та вимог [8].

Основні переваги та особливості використання діаграм діяльності такі:

- Діаграми діяльності можуть доповнювати будь-який елемент моделі з динамічною поведінкою;
- Діаграми діяльності можуть відображати дії, які відбуваються одночасно;
- Плавальні доріжки можуть використовуватись для розподілу діяльності відповідно до ролей або об'єктів, які виконують їх;
- Траєкторії об'єктів дозволяють показати зміни станів об'єктів під час виконання діяльності;
- Складні діяльності можуть бути деталізовані, розбиваючи їх на декілька дій та використовуючи вкладені діаграми;

- Діаграми діяльностей можуть бути використані для проєктування процесів або операцій, виступаючи як візуальна мова програмування.

На рис. 2.3 зображена діаграма діяльності, на якій зображено послідовність створення профілю для учня. Адміністратор реєструє учня, вказуючи основну інформацію: прізвище, ім'я, по батькові, пошту та клас, до якого учень буде доданий. Після успішного створення профілю, на пошту учня приходить повідомлення про реєстрацію в системі. У повідомленні міститься згенерований логін та посилання для встановлення пароля. Перейшовши по посиланню, учень встановлює пароль і має можливість увійти в систему, ввівши логін та пароль. Після успішної авторизації є можливість переглядати уроки та домашні завдання.

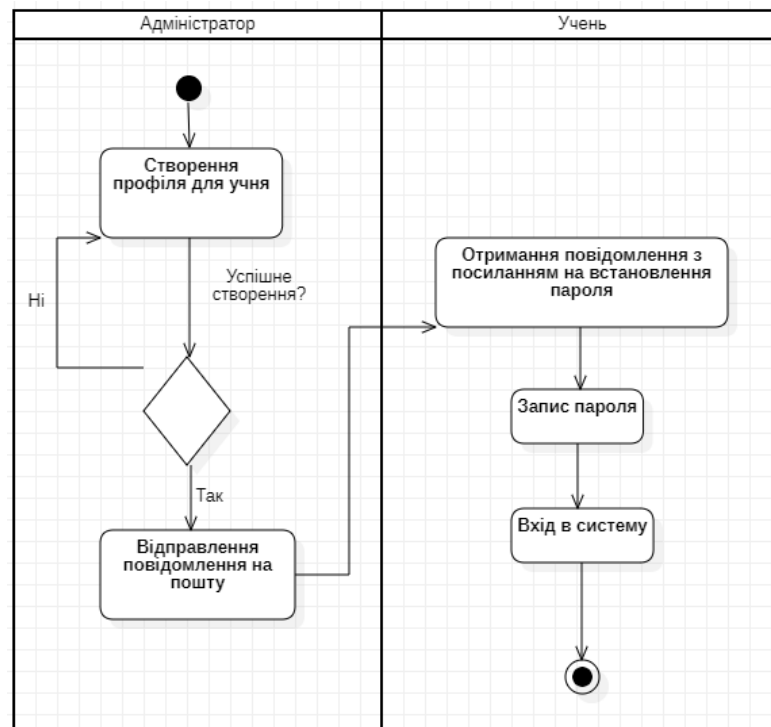


Рис. 2.3 - Діаграма діяльності створення профілю для учня

Наступним кроком є ініціатива вчителя - створення уроку. Він має доступ лише до тих предметів, які він викладає. При переході до обраного предмета, вчителю надається можливість створити урок, де він може вказати основну інформацію та, при необхідності, додати прикріплений файл. Окрім цього, вчителю надається можливість створити домашнє завдання для створеного уроку. Він може описати завдання, додати файл та встановити термін здачі.

Після створення уроку, учень отримує можливість опрацювати його. Він може вивчити матеріал, виконати поставлені завдання та підготуватися до домашнього завдання. Однак, якщо термін здачі вже минув, учень втрачає можливість відправити своє рішення.

Вчитель має можливість переглядати розв'язки, які надіслані учнями, та оцінювати. Крім того, йому надається можливість залишати коментарі, як зворотний зв'язок щодо виконання завдань. Це дозволяє вчителю надати конструктивні поради та підтримку учням, а також сприяє зростанню їхньої навчальної успішності.

Таким чином, процес створення уроку та виконання домашнього завдання в системі забезпечує вчителю можливість ефективно організувати навчальний процес, а учням - систематично опрацьовувати навчальний матеріал та вчасно виконувати завдання.

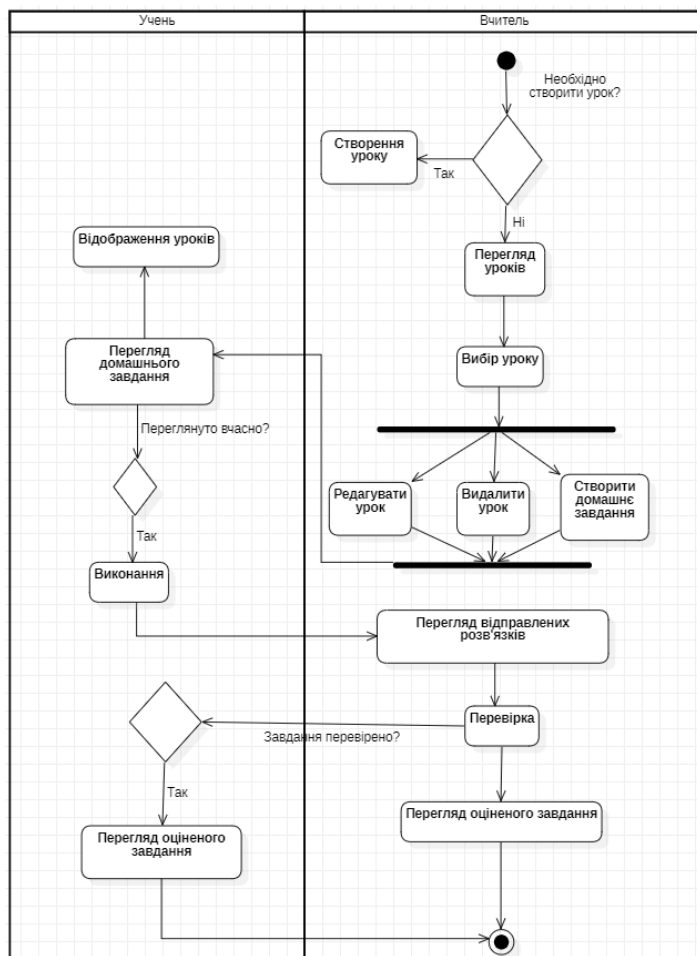


Рис. 2.4 - Діаграма діяльності створення уроку та оцінювання домашнього завдання

Останній, але надзвичайно важливий етап у реалізації проєктованої системи - розробка діаграми послідовності. Діаграма послідовності є ключовим інструментом, який відображає процеси, що відбуваються в системі без прямої участі користувача, та демонструє взаємодію між різними компонентами системи [9]. Коли клієнт відправляє запит на певний URL, веб-сервер перехоплює цей запит та передає його до маршрутизатора. Маршрутизатор визначає, який контролер та дія повинні бути викликані для обробки цього запиту. Потім контролер звертається до моделі, яка отримує дані з бази даних. Отримані дані проходять обробку та повертаються до представлення, де вони відображаються користувачу.

Для даного додатку була розроблена загальна діаграма послідовності, яка представлена на рис. 2.5.

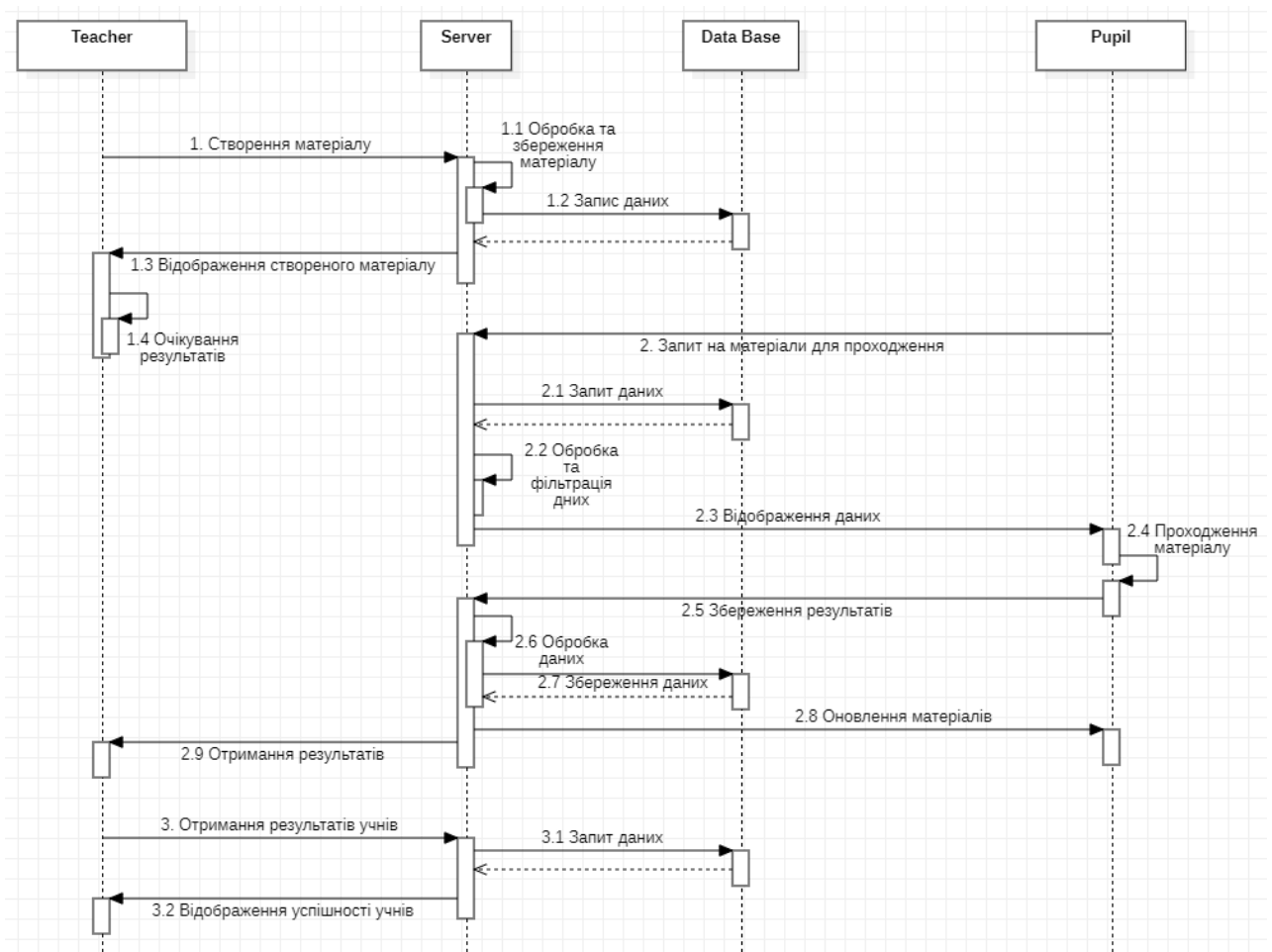


Рис. 2.5 – Діаграма послідовності системи

2.4. Реалізація додатка «LMS система для закладів загальної шкільної освіти»

У цьому розділі ми розглянемо діаграми компонентів, які демонструють відносини між різними елементами програмного забезпечення, включаючи вихідний код, бінарні та виконувані компоненти. Кожен модуль програмного забезпечення буде представлений як окремий компонент.

Деякі з цих компонентів присутні під час компіляції, деякі під час збірки, а деякі - під час виконання програми. Варто відзначити, що діаграма компонентів показує тільки структурні аспекти, а для відображення конкретних екземплярів компонентів потрібно використовувати діаграму розгортання.

Компоненти - це фізичні частини системи, які відповідають за втілення класів, відносин та функціональної поведінки програмної системи. У даному контексті (рис. 2.6), компоненти включають:

- MySQL - базу даних
- Site Context - компонент, який відповідає за відображення контексту на веб-сторінці. Контекст залежить від ролей, які будуть наведені нижче.
- Student – тип користувача у системі (учень);
- Teacher – тип користувача у системі (вчитель);
- Admin – тип користувача у системі (адміністратор);
- Models – компоненти, які є модулями в програмному коді
 - Subject – модель для CRUD предметів
 - Subject Search – модель для пошуку предметів
 - Lesson - модель для CRUD уроків
 - Lesson Search – модель для пошуку предметів
 - Class – модель для CRUD класів
 - Homework - модель для CRUD домашнього завдання
 - Homework Search – модель для пошуку домашніх завдань
 - HomeworkSubmission – модель для CRUD відправлених завдань

- Progress – модель, що відображає успішність учнів;
- Student – модель для створення, редагування та додавання учня до системи
- Teacher – модель для створення, редагування та додавання вчителя до системи

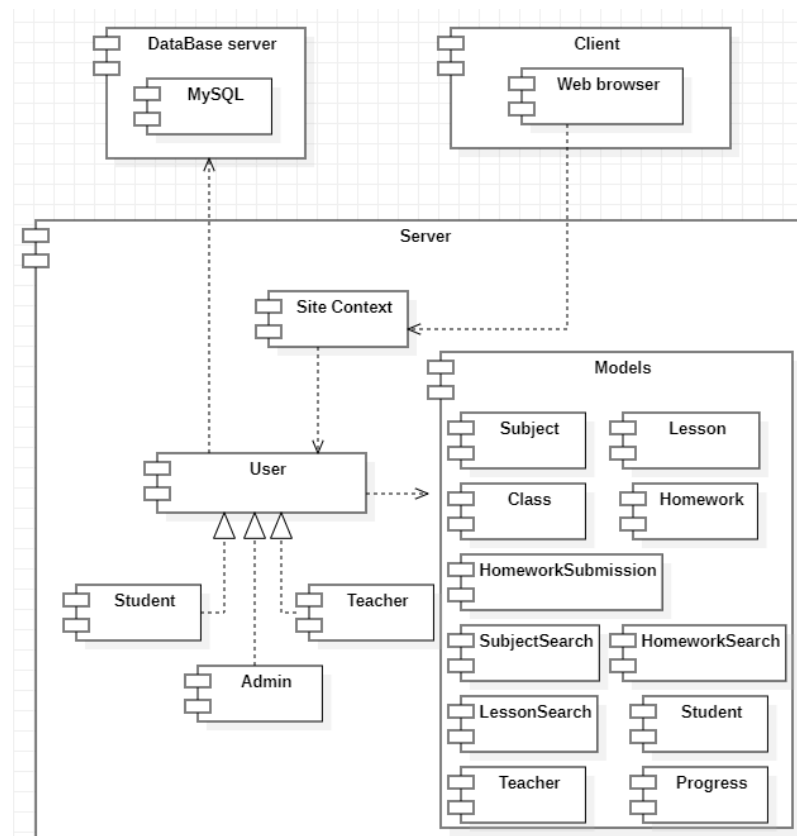


Рис. 2.6 – Діаграма компонентів

Для того, щоб додаток функціонував, потрібно здійснити ряд підготовчих кроків. Одним з найважливіших кроків є підключення програмного додатка до бази даних (БД), що забезпечує збереження та доступ до інформації. У фреймворку Yii2 процес підключення до БД здійснюється в файлі конфігурації config/main-local.php [10].

Цей файл має ключове значення для встановлення з'єднання з базою даних і визначення параметрів доступу, таких як хост, ім'я користувача, пароль, назва бази даних та інші. Шляхом налагодження правильних налаштувань в файлі main-local.php, програмний додаток зможе взаємодіяти з базою даних і

виконувати необхідні операції, такі як збереження, отримання та модифікація даних.

Це важливий крок, оскільки без належного підключення до бази даних додаток не зможе працювати зі збереженими даними та виконувати запити до БД. Отже, налаштування файлу `main-local.php` [18] є необхідною передумовою для успішної роботи додатка з базою даних у фреймворку Yii2.

```
return [
    'components' => [
        'db' => [
            'class' => 'yii\db\Connection',
            'dsn' => 'mysql:host=localhost;dbname=lms_school',
            'username' => 'root',
            'password' => 'root',
            'charset' => 'utf8',
        ],
        'mailer' => [
            'class' => 'yii\swiftmailer\Mailer',
            'viewPath' => '@common/mail',
            'useFileTransport' => true,
        ],
    ],
];
```

Лістинг 2.1 – Підключення до БД

Функція `actionCreate` (лістинг 2.2) відповідає за створення акаунта для учня в системі. При виклику цієї функції, спочатку створюється новий об'єкт моделі `User`, який представляє користувача системи.

У разі отримання POST-запиту (тобто коли форма була відправлена), функція перевіряє, чи були дані успішно завантажені з POST-запиту та чи пройшли валідацію. Якщо так, генерується випадковий пароль для користувача і встановлюється логін, що формується з прізвища, імені та по батькові користувача. Далі, генерується токен для скидання пароля.

Після цього, користувач зберігається в базі даних. Якщо збереження пройшло успішно, функція призначає роль `"Student"` (учень) для даного користувача, прив'язує його до певного класу та відправляє листа з реєстраційними даними на електронну пошту користувача. Крім того, встановлюється сповіщення про успішне створення акаунта, і користувач перенаправляється на сторінку перегляду деталей цього акаунта.

Якщо POST-запит не був отриманий, модель користувача заповнюється значеннями за замовчуванням. На останок, функція відображає сторінку створення акаунта, передаючи модель користувача та список класів для відображення на сторінці.

Ця функція є частиною програмного коду, яка додає можливість створення акаунтів учнів в системі, надаючи їм необхідні ролі, класи та надсилаючи листи з реєстраційними даними.

```
public function actionCreate()
{
    $model = new User();
    if ($this->request->isPost) {
        if ($model->load(Yii::$app->request->post()) && $model->validate()) {
            // Згенеруємо випадковий пароль для користувача
            $password = Yii::$app->security->generateRandomString(8);
            $this->setPassword($model, $password);
            // Встановлюємо логін
            $model->username = $this->generateUsername($model->lastname,
            $model->firstname, $model->patronymic);
            $model->generatePasswordResetToken();
            // Зберігаємо користувача
            if ($model->save()) {
                // Призначаємо роль користувачеві
                $auth = Yii::$app->authManager;
                $student = $auth->getRole('Student');
                $auth->assign($student, $model->id);
                // Додаємо до класу
                $this->assignStudentToClass($model->id, $model->class_id);
                // Відправляємо листа
                $this->sendRegistrationEmail($model, $password);
                Yii::$app->session->setFlash('success', 'Учень успішно
зареєстрований.');
```

```
                return $this->redirect(['view', 'id' => $model->id]);
            }
        }
    }
    $model->loadDefaultValues();
    return $this->render('create', [
        'model' => $model,
        'classList' => $this->getClassList(),
    ]);
}
```

}

Лістинг 2.2 – Метод створення облікового запису для учня

Функція `actionUpdate` (лістинг 2.3) відповідає за редагування уроку в системі. При виклику цієї функції, спочатку отримується модель уроку з бази даних за його ідентифікатором.

При отриманні POST-запиту (тобто коли форма була відправлена), функція завантажує дані з POST-запиту в модель. Також визначається новий файл, який може бути прикріплений до уроку.

Далі, перевіряється, чи був вибраний новий файл. Якщо так, проводиться валідація моделі. Якщо валідація успішна, старий файл видаляється, а новий файл завантажується і зберігається. У разі виникнення помилок при валідації нового файлу, попередній файл залишається без змін. Якщо новий файл не був вибраний, попередній файл залишається без змін. Після цього, модель уроку зберігається в базі даних. Якщо збереження успішне, користувач перенаправляється на сторінку перегляду деталей цього уроку.

Ця функція є частиною програмного коду, який надає можливість редагування уроків в системі, включаючи можливість змінювати опис уроку та прикріплювати нові файли.

```
public function actionUpdate($id)
{
    $model = $this->findModel($id);
    $oldFile = $model->file; // Зберегти старий файл
    if ($model->load(Yii::$app->request->post())) {
        $model->file = UploadedFile::getInstance($model, 'file');
        // Перевіряємо, чи вибрано новий файл
        if ($model->file) {
            if ($model->validate()) {
                // Видаляємо старий файл
                if ($oldFile) {
                    unlink($model->getFolder() . $oldFile);
                }
                // Зберігаємо новий файл
                $model->file = $model->upload();
            } else {
                // Якщо виникли помилки при валідації нового файлу
                // Залишаємо попередній файл без змін
                $model->file = $oldFile;
            }
        } else {
            // Якщо не обрано новий файл, зберігаємо попередній файл
            $model->file = $oldFile;
        }
        if ($model->save()) {
```

```

        return $this->redirect(['view', 'id' => $model->id]);
    }
}

return $this->render('update', [
    'model' => $model
]);
}

```

Лістинг 2.3 – Метод редагування уроку

Дана логіка схожа для створення та редагування домашнього завдання та його виконання.

Відображення домашніх завдань для учня здійснюється таким чином, щоб він мав можливість бачити лише завдання тих предметів, які були поширені серед його класу. Дану функцію (лістинг 2.4) використано в моделі HomeworkSearch для отримання списку доступних предметів для поточного учня. Основна мета функції - створити асоціативний масив, де ключами будуть ідентифікатори предметів, а значеннями - їх назви.

```

public function getAvailableSubjects()
{
    $class_id = ClassStudent::findOne(['student_id' => Yii::$app->user->getId()])->class_id;

    $subjectIds = SubjectClass::find()
        ->select('subject_id')
        ->where(['class_id' => $class_id])
        ->column();

    $subjects = Subject::find()
        ->where(['id' => $subjectIds])
        ->all();

    return \yii\helpers\ArrayHelper::map($subjects, 'id', 'title');
}

```

Лістинг 2.4 – Метод для повернення масиву доступних завдань для учня

Для забезпечення безпеки та зручності збереження даних, було реалізовано функціонал імпорту та експорту бази даних у файл формату .sql [17]. Ця можливість доступна тільки адміністратору системи. Він може створити резервну копію бази даних в будь-який момент, використовуючи функцію експорту. Адміністратор також має можливість імпортувати резервну копію

безпосередньо через інтерфейс, щоб відновити базу даних. Такий підхід забезпечує надійність та зручність управління даними в системі.

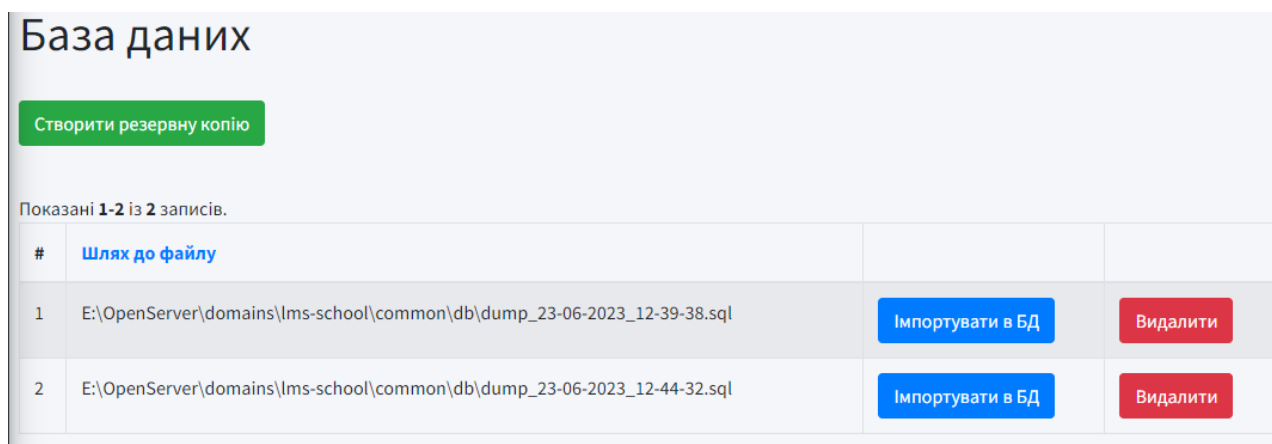


Рис. 2.7 – Сторінка для створення резервної копії БД

Висновки до розділу 2

В процесі написання другого розділу було розроблено діаграми варіантів використання, активності та компонентів. Досліджено варіанти використання та основний функціонал, який має бути наявний у додатку для користування системою в навчальних закладах.

Також було спроектовано базу даних для зберігання даних, що використовуватимете додаток. Продемонстровано діаграму спроектованої бази даних, проведено опис кожної таблиці.

В останньому пункті було успішно втілено всі задумані та запроектовані функції системи, створивши повноцінний та функціональний програмний продукт. Представлено широкий спектр методів, що дозволяють реалізувати основні та найважливіші функціональні можливості системи.

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ LMS СИСТЕМИ ДЛЯ ЗАКЛАДІВ ЗАГАЛЬНОЇ ШКІЛЬНОЇ ОСВІТИ

3.1. Порядок встановлення та налаштування параметрів системи.

Система користувача повинна задовольняти вимогам на комп'ютері, для повної взаємодії з LMS системою, що знаходиться в наступній мінімальній комплектації:

- операційна система Windows 7 або вище, Linux 8.10 та вище, MacOS 10.6 Snow Leopard та вище;
- 32-розрядний (x86) або 64-розрядний (x64) процесор із тактовою частотою 1 ГГц або вище;
- 1 ГБ (для 32-розрядних процесорів) або 2 ГБ (для 64-розрядних процесорів) ОЗУ;
- 16 ГБ (для 32-розрядної системи) або 20 ГБ (для 64-розрядної системи) вільного місця на жорсткому диску;

Розгортання діаграми є важливим інструментом для візуалізації архітектури програмного продукту. Вона демонструє обчислювальні вузли, компоненти та об'єкти, які функціонують під час роботи програми. На цій діаграмі можуть бути відображені лише ті компоненти, які активно використовуються в процесі роботи програми [11].

На рисунку 3.1. представлена діаграма розгортання системи. Діаграма має наступну структуру:

- User PC – веб-інтерфейс в браузері;
- PHP – проєктований програмний продукт;
- DataBase server – інтерфейс для взаємодії з базою даних;
- Web Server – веб-сервер, на якому працює програмне забезпечення.

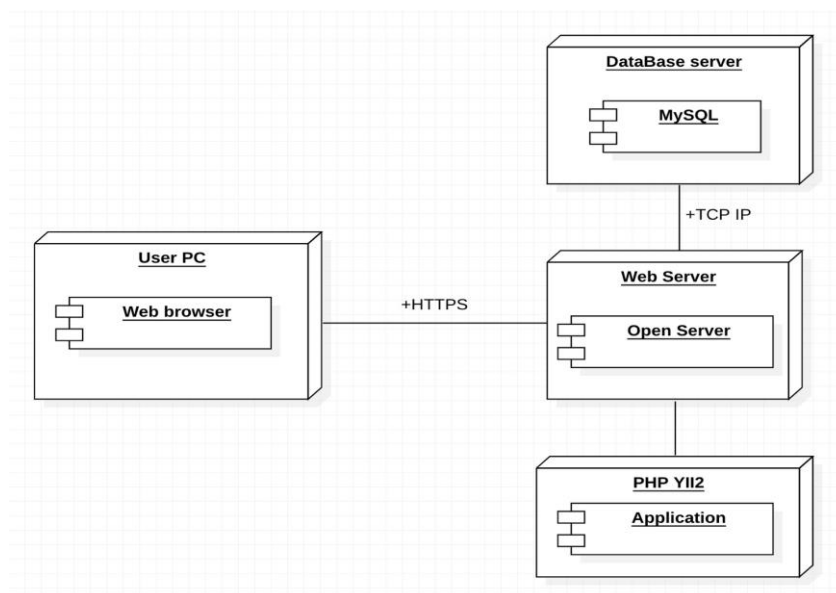


Рис. 3.1- Діаграма розгортання

3.2. Структура інтерфейсу LMS системи закладів загальної шкільної освіти

В даній системі користувач немає змоги зареєструватися. Реєстрація учня та вчителя відбувається за допомогою адміністратора. Адміністративна частина доступна за запитом <http://lms-school/admin>, де потрібно ввести свій логін та пароль. Після переходу за посиланням адміністратор опиняється на головній сторінці (рис. 3.2). На цій сторінці відображається посилання на створення резервної копії БД та деякі статистичні дані, а саме загальну кількість учнів, вчителів, класів та предметів.

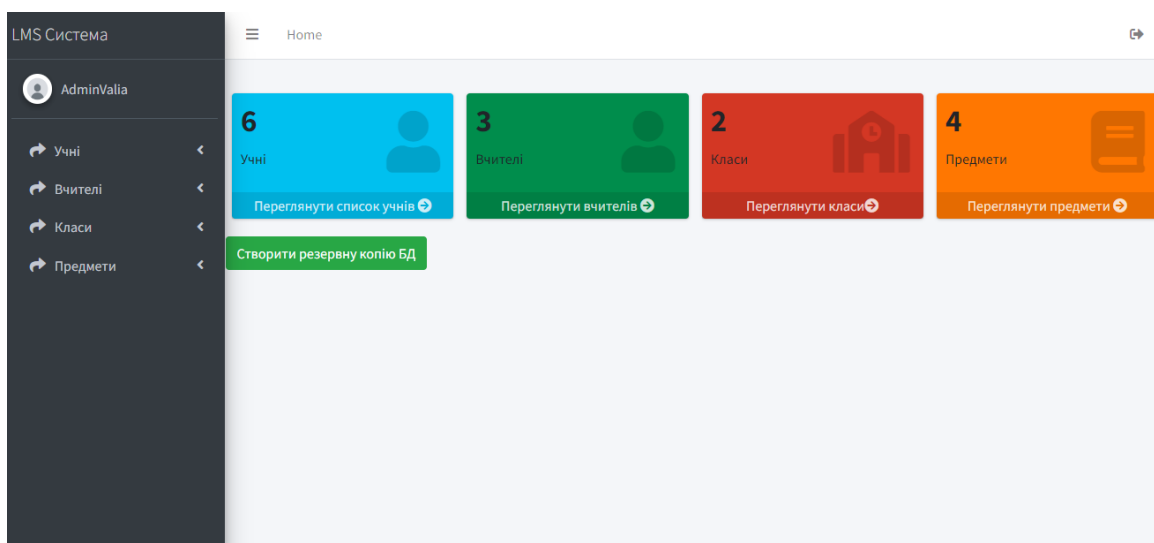


Рис. 3.2 - Головна сторінка адміністративної частини

До функцій адміністративної частини входять операції щодо створення облікових записів для учнів та вчителів, класів та предметів. Доступ до адміністративної частини відкритий лише користувачам з роллю Admin.

Розпочнемо зі створення облікового запису для учня. Адміністратор має можливість переглядати список учнів та додавати/редагувати учнів (рис. 3.3). Доступний пошук за логіном, прізвищем, іменем, по батькові та за поштою. Також є можливість фільтрувати, обираючи певний клас з випадного списку. Оскільки учні є зареєстрованими користувачами, цей блок включає функції реєстрації та обробки даних користувачів через адміністративну частину системи. Після успішної реєстрації учня, йому приходить повідомлення на пошту із вказаним логіном та посиланням для встановлення пароля (рис. 3.4).

Створити обліковий запис учня

Показані 1-5 із 5 записів.

#	Клас	Логін	Ім'я	Прізвище	По-батькові	Email	
	Виберіть клас ▾						
1	1-A	ромашербина	Роман	Щербина	Олександрович	romasherbyna@gmail.com	 
2	3-B	твколесник	Тетяна	Колесник	Володимирівна	kolesniktania08@gmail.com	 
3	1-A	івбондаренко	Інна	Бондаренко	Вайлівна	inna@gmail.com	 
4	1-A	іспилипенко	Іван	Пилипенко	Сергійович	bod@gmail.com	 
5	3-B	мімаляр	Марина	Маляр	Ігнатівна	maryna@gmail.com	 

Рис. 3.3 - Інтерфейс сторінки «Учні»

Реєстрацію успішно завершено в LMS системі Вхідні x


valia.overchuk@gmail.com
 кому мені ▾

Шановний(а) **Мельник Олександр Миколайович**,

Вітаю! Ви успішно зареєстровані в нашій системі. Ми раді вітати вас як учня нашої школи 3-Б класу. Нижче наведені деталі вашого облікового запису:

Логін: оммельник

Для завершення процесу реєстрації та встановлення пароля, будь ласка, перейдіть за наступним посиланням:

[Встановити пароль](#)

Будь ласка, встановіть надійний пароль, який ви зможете запам'ятати. Збережіть ці дані у надійному місці та не передавайте їх іншим особам.

Якщо у вас виникнуть питання або потрібна допомога, звертайтеся до нас. Ми з радістю допоможемо вам.

З повагою,
Адміністрація школи

Рис. 3.4 – Відправлений лист учню

Адміністратор може переглядати список вчителів. Застосований фільтр дозволяє здійснювати пошук за всіма полями. Крім того, адміністратор може фільтрувати список вчителів за класом, де він є класним керівником. Колонка "Керівник в класі" відображає назву класу, в якому вчитель є класним керівником. Також доступні дії для редагування кожного вчителя окремо. Це допомагає адміністратору з легкістю керувати обліковими записами вчителів.

Створити обліковий запис вчителя

Показані 1-3 із 3 записів.

#	Логін	Ім'я	Прізвище	По-батькові	Email	Керівник в класі	
						Виберіть клас	
1	TeacherValia	Валентина	Оверчук	Сергіївна	ipz193_ovs@student.ztu.edu.ua	1-А	 
2	асірач	Анна	Сірач	Сергіївна	ipz193_sas@student.ztu.edu.ua	3-Б	 
3	азмельник	Анна	Мельник	Анатоліївна	annM@gmail.com	(не задано)	 

Рис. 3.5 - Інтерфейс сторінки «Вчителі»

Адміністратор може змінювати такі дані про учня як прізвище, ім'я, по батькові, електронна пошта та клас. Випадаючий список "Клас" містить список класів, з якого можна обрати потрібний клас для користувача. Після внесення необхідних змін, адміністратор може натиснути кнопку "Зберегти", щоб зберегти оновлені дані облікового запису.

Прізвище

Щербина

Ім'я

Роман

По-батькові

Олександрович

Email

romasherbyna@gmail.com

Клас

1-А

Зберегти

Рис. 3.6 - Редагування даних для учня

У списку класів адміністратор може переглядати інформацію про класи, а також здійснювати пошук і фільтрацію за допомогою різних полів. На сторінці списку класів відображаються наступні дані про кожен клас: назва класу та інформація про класного керівника. Також, у кожному рядку таблиці є стовпець, який відображає кількість учнів, що належать до даного класу. Ця інформація оновлюється автоматично і відображає кількість учнів, які зареєстровані в цьому класі. Адміністратор може створювати нові класи, натискаючи кнопку "Створити клас". Також, для кожного класу доступні дії редагування та видалення, які можна виконати, вибравши відповідні піктограми у кожному рядку таблиці. Для переходу до сторінки редагування класу адміністратор може натиснути на посилання, яке знаходиться у відповідному рядку таблиці.

Створити клас				
Показані 1-2 із 2 записів.				
#	Назва класу	Класний керівник	Кількість учнів	
1	1-А	Оверчук Валентина Сергіївна	3	  
2	3-Б	Сірач Анна Сергіївна	2	  

Рис. 3.7 - Інтерфейс сторінки «Класи»

Лише адміністратор має можливість створювати нові предмети та вибирати вчителів, які будуть їх викладати. Після цього, адміністратор розподіляє ці предмети серед класів. Кожен учень, що належить до класу, якому призначений цей предмет, отримує доступ до уроків і може виконувати домашні завдання.

Після успішного входу на наш сайт, ви зустрінете головну сторінку нашої системи. Ця сторінка є вхідним пунктом до всіх функцій і можливостей, які ми пропонуємо. На головній сторінці є швидкий доступ до основних функцій системи. Ви зможете легко знайти та відкрити свої предмети, уроки та домашні завдання.

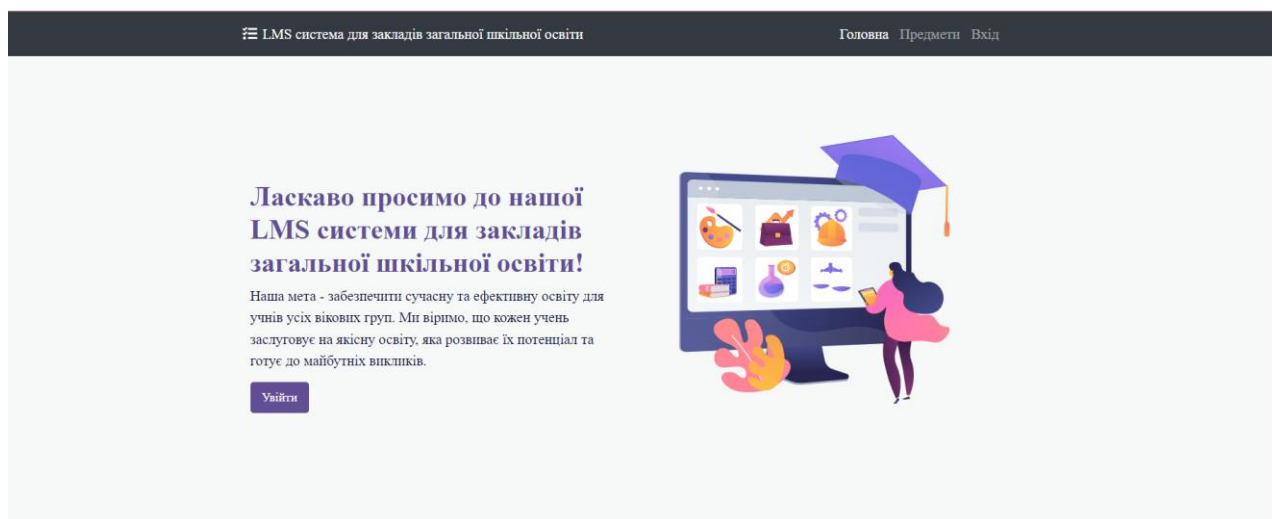


Рис. 3.8 - Головна сторінка системи

Для авторизації в системі потрібно ввести логін та пароль. Усім зареєстрованим користувачам, незалежно від їхньої ролі, доступний перегляд власного профілю (рис. 3.9) та можливість редагування певних полів в ньому. Ця функція надає зручність та контроль користувачам, дозволяючи їм налаштовувати та оновлювати особисту інформацію згідно з їхніми потребами та вимог. Якщо користувач – Учень, він може переглядати уроки та виконувати домашні завдання. На сторінці «Домашні завдання» учень може переглянути список всіх домашніх завдань (рис. 3.10). Кожне завдання належить до певного предмета і має назву уроку, до якого воно відноситься. Також є можливість побачити оцінку та коментар, якщо вже виконано це завдання. Крім того, учень може переглянути поточний статус домашнього завдання, який показує, чи воно було виконане. Якщо він ще не виконав завдання, то має змогу натиснути кнопку «Виконати», щоб перейти до завдання і виконати його. Якщо завдання вже виконано, учень може натиснути кнопку «Переглянути», щоб переглянути свою відправлену роботу. Є можливість використовувати фільтри, щоб швидко знайти домашнє завдання за предметом або статусом. Також існує сортування за оцінкою. Лістинг контролеру та моделей, що відповідає за наведені вище функції представлені у додатку А.

конкретного домашнього завдання. Учень бачить опис завдання, дату створення, термін здачі та можливість завантажити файл (рис. 3.10).

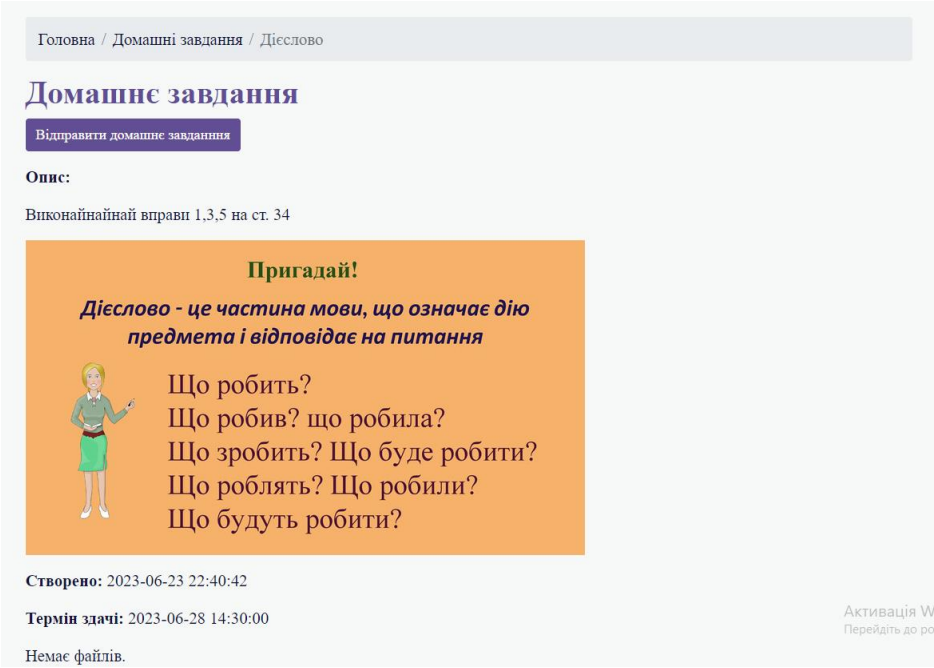


Рис. 3.10 - Відображення домашнього завдання для учня

Після виконання завдання, учень може відстежувати статус перевірки (рис. 3.11). Учень має можливість відправити свій розв'язок домашнього завдання, якщо він ще не був відправлений. Якщо завдання вже відправлено, він може переглянути його статус, оцінку, коментар від вчителя та переглянути свій розв'язок.

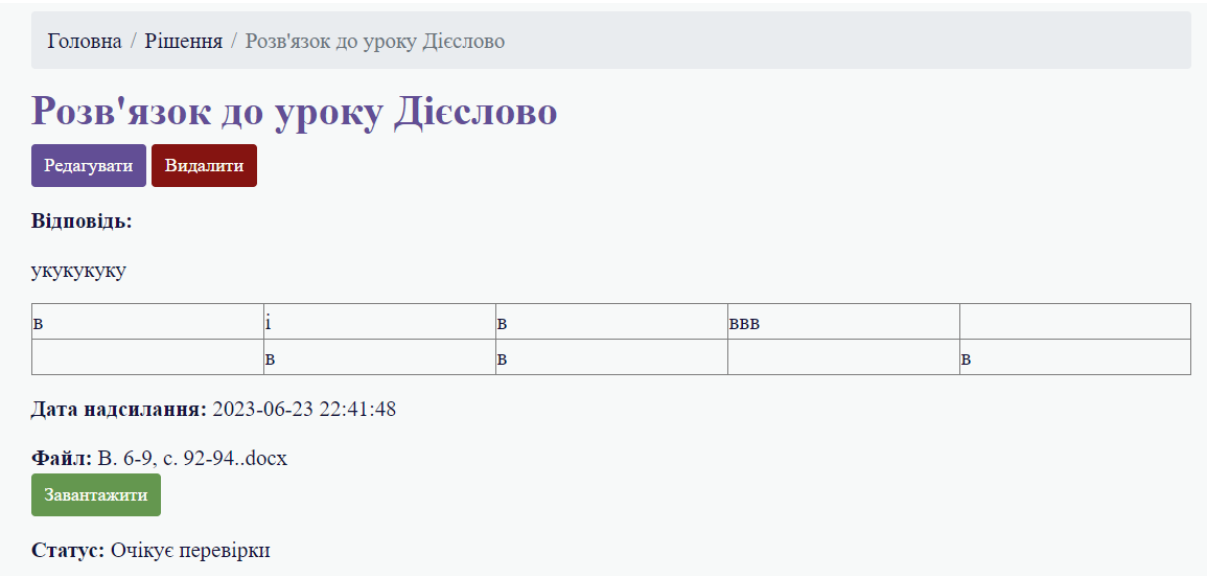


Рис. 3.11 - Відображення домашнього завдання для учня

Якщо вчитель відкриває домашнє завдання, він отримує розширені можливості для керування ним. Він має змогу редагувати та видаляти завдання, а також переглядати розв'язки, надані учнями. Для кожного учня, який здав домашнє завдання, вчитель може переглянути інформацію про його клас, ім'я, статус розв'язку, оцінку та коментар. Вчитель також може переглянути деталі розв'язку та виконати його перевірку (рис. 3.12).

Проте, важливо зазначити, що якщо хоча б один учень вже виконав домашнє завдання і надав свій розв'язок, вчитель втрачає можливість видалити або редагувати завдання. Це зроблено з метою забезпечити цілісність процесу навчання та врахувати зусилля учнів, які вже працювали над завданням (рис. 3.13).

Таким чином, вчитель має повний контроль над домашніми завданнями, спілкуючись з учнями та оцінюючи їхні досягнення, забезпечуючи ефективну взаємодію та розвиток навчального процесу.

Розв'язок

Відповідь:

укукукуку

В	і	В	ВВВ	
	В	В		В

Дата надсилання: 2023-06-23 22:41:48

Файл: В. 6-9, с. 92-94..docx

[Завантажити](#)

Grade

Необхідно заповнити "Grade".

Comment

File Edit View Insert Format Tools

Table

↶ ↷ ...

p 0 words tiny

[Оцінити](#)

Рис. 3.12 Відображення форми для перевірки надісланого домашнього завдання

Головна / Домашні завдання / Дієслово

Домашнє завдання

Опис:

Виконай/най вправи 1,3,5 на ст. 34

Пригадай!

Дієслово - це частина мови, що означає дію предмета і відповідає на питання



Що робить?
Що робив? що робила?
Що зробить? Що буде робити?
Що роблять? Що робили?
Що будуть робити?

Створено: 2023-06-23 22:40:42

Термін здачі: 2023-06-28 14:30:00

Немає файлів.

Клас	Учень	Статус розв'язку	Оцінка	Коментар	Розв'язок
1-А	Щербина Роман Олександрович	Перевірено	11	Молодець	Переглянути

Рис. 3.13 Відображення домашнього завдання та відправлених рішень

3.3. Тестування роботи програмного комплексу

Тестування розробленої системи на працездатність – є невід’ємним процесом під час розробки програмного коду.

Виконаємо перевірки для знаходження помилок та своєчасного виправлення їх.

Створення класу та призначення класного керівника

Передумови: перебування Адміністратора на сторінці «Створити клас»;

Кроки: вводимо назву класу та натискаємо на випадаючий список з відображенням вчителів;

Очікуваний результат: з’являється випадаючий список лише з тими вчителями, які ще не є класним керівником в існуючих класах;

Фактичний результат: з’являється випадаючий список лише з тими вчителями, які ще не є класним керівником в існуючих класах.

Створення облікового запису учня

Передумови: перебування користувача на сторінці «Додати учня»;

Кроки: відправити заповнену форму відповідними даними:

- Прізвище: Іваненко

- Ім'я: Іван
- По батькові: Іванович
- Пошта: testvaliadypлом@gmail.com
- Клас: 1-А

Очікуваний результат: учень відобразиться в загальному списку учнів, а на його пошту прийде лист з даними для входу;

Фактичний результат: учень відобразиться в загальному списку учнів, а на його пошту прийде лист з даними для входу.

Заборона здачі домашнього завдання, якщо термін здачі вже минув

Передумови: перебування Учня на сторінці «Предмети»;

Кроки: обрати предмет Українська мова. Перейти на урок Вигуки. Натиснути на кнопку «Відправити домашнє завдання»;

Очікуваний результат: учень бачить повідомлення, що термін здачі вже закінчився і відображаються наступні уроки в даному предметі;

Фактичний результат: учень бачить повідомлення, що термін здачі вже закінчився і відображаються наступні уроки в даному предметі.

Зміна зображення профілю

Передумови: перебування користувача на сторінці «Мій профіль»;

Кроки: користувач натискає на кнопку «Редагувати зображення», далі завантажує нове і натискає на кнопку «Зберегти»;

Очікуваний результат: старе фото профілю видалиться і заміниться на нове;

Фактичний результат: старе фото профілю видалиться і заміниться на нове.

Експорт резервної копії бази даних

Передумови: перебування адміністратора на головній сторінці адміністративної частини;

Кроки: адміністратор натискає на кнопку «Створити резервну копію БД»;

Очікуваний результат: створений файл з резервною копією завантажено в папку для архіву БД та відображено на сторінці новий запис з дампом;

Фактичний результат: створений файл з резервною копією завантажено в папку для архіву БД та відображено на сторінці новий запис з дампом.

Було проведено повноцінне тестування всього функціоналу платформи з метою перевірки його працездатності. Під час цього тестування не було виявлено жодних помилок або недоліків у роботі системи. Додаток повністю готовий для зручного використання користувачами. Всі необхідні функції та можливості доступні та працюють належним чином.

Висновки до розділу 3

У даному розділі були встановлені вимоги до комп'ютерної системи для повноцінної взаємодії з додатком «LMS система для закладів загальної шкільної освіти». Також була розроблена діаграма розгортання системи, на якій зображено компоненти та об'єкти, що використовуються під час роботи програми.

Далі було представлено структуру інтерфейсу системи для закладів загальної шкільної освіти. Для кожної з ролей був показаний доступний вигляд інтерфейсу та описано функціональні можливості системи. Особлива увага була приділена тестуванню програмного комплексу, результати якого були позитивними, свідчаючи про готовність програмного продукту до повного використання.

В результаті виконаних дій було отримано готовий продукт під назвою «LMS система для закладів загальної шкільної освіти», який повністю задовольняє всі вимоги та функціональні можливості системи, визначені раніше.

ВИСНОВКИ

Під час виконання випускної кваліфікаційної роботи бакалавра було спроектовано та реалізовано LMS систему для закладів загальної шкільної освіти. В ході виконання даної роботи було ретельно проаналізовано обрану тематику, а також проведено порівняльний огляд існуючих аналогів, що відображають подібну ідею чи принцип роботи. Цей аналіз дав змогу сформулювати основні вимоги та визначити завдання для майбутньої системи.

В результаті проведеного дослідження було визначено, що найбільш оптимальною архітектурою для розробки нашої платформи є веб-орієнтована архітектура Model-View-Controller (MVC). Цей вибір забезпечує ефективне розмежування моделі даних, інтерфейсу користувача та логіки взаємодії, що приводить до покращеної гнучкості, модульності та можливості швидкого виправлення проблем у системі. Застосування MVC дозволяє нам зручно керувати різними компонентами платформи та забезпечує високий рівень масштабованості та підтримки системи.

Також були обґрунтовані вибір інструментів та апаратних вимог для розробки програмного комплексу. Використано Open Server, MySQL, PHP та Yii2 фреймворк на сервері, HTML та CSS на клієнтській стороні.

Було розроблено діаграми варіантів використання, активності та компонентів. Досліджено варіанти використання та основний функціонал, який має бути наявний у додатку для користування системою в навчальних закладах.

В результаті виконання кваліфікованої роботи було успішно втілено всі задумані та запроектовані функції системи, створивши повноцінний та функціональний програмний продукт. Представлено широкий спектр методів, що дозволяють реалізувати основні та найважливіші функціональні можливості системи.

Результати випускної роботи можуть стати основою для подальшого розвитку та вдосконалення платформи управління навчанням для закладів загальної шкільної освіти. Даний програмний продукт може удосконалюватися шляхом введення нових функцій: створення розкладу і сповіщення про урок,

					ІПЗ.КР.Б – 121 – 23 – ПЗ	
						52

створення конструктора для побудови уроку з можливістю перевіркою знань, шляхом тестування.

					ІІЗ.КР.Б – 121 – 23 – ІІЗ	
						53

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Фрімен А. Head First Design Patterns / А. Фрімен, Е. Фрімен. – Sebastopol: O'Reilly Media, 2016. – 694 с.
2. Джонсон М. Design Patterns: Elements of Reusable Object-Oriented Software / М. Джонсон. – Boston: Addison-Wesley Professional, 2019. – 395 с.
3. Сміт Д. Open Server: Complete Guide to Web Development / Дж Сміт. – New York: XYZ Publishers, 2022. – 320 с.
4. Томпсон Р. PHP Framework Mastery: Yii2 in Practice / Р. Томпсон. – San Francisco: ABC Publications, 2019. – 400 с.
5. Веб-сервіси RESTful: Швидкий старт [Електронний ресурс] – Режим доступу до ресурсу: <https://yiiframework.com.ua/uk/doc/guide/2/rest-quick-start/>
6. Authorization [Електронний ресурс] – Режим доступу до ресурсу: <https://www.yiiframework.com/doc/guide/2.0/en/security-authorization>
7. Nixon R. Learning MySQL: Get a Handle on Your Data / Robin Nixon. – Sebastopol: O'Reilly Media, 2007. – 584 с.
8. UML Activity Diagram Tutorial. [Електронний ресурс] – Режим доступу до ресурсу: <https://www.lucidchart.com/pages/uml-activity-diagram>
9. Sequence Diagram Tutorial [Електронний ресурс] – Режим доступу до ресурсу: <https://online.visual-paradigm.com/diagrams/tutorials/sequence-diagram-tutorial/>
10. Caldarelli F. Yii2 by Example / Fabrizio Caldarelli. – Birmingham: Packt Publishing, 2015. – 406 с.
11. The Easy Guide to UML Deployment Diagrams [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: <https://creatly.com/guides/deployment-diagram-tutorial/>
12. Kirvan P. learning management system (LMS) [Електронний ресурс] / Р. Kirvan, К. Brush – Режим доступу до ресурсу: <https://www.techtarget.com/searchcio/definition/learning-management-system>

					ІІЗ.КР.Б – 121 – 23 – ІІЗ	
						54

13. Google Classroom [Електронний ресурс] – Режим доступу до ресурсу:
<https://classroom.google.com/>
14. Moodle [Електронний ресурс] – Режим доступу до ресурсу:
<https://moodle.org/>
15. Schoology Learning [Електронний ресурс] – Режим доступу до ресурсу:
<https://www.schoology.com/>
16. Bootstrap Documentation: Layout [Електронний ресурс] – Режим доступу до ресурсу: <https://getbootstrap.com/docs/4.0/layout/overview/>
17. В'єскас Д. Л. SQL Queries for Mere Mortals: A Hands On Guide to Data Manipulation in SQL / Д. Л. В'єскас, М. Д. Гернандес. – Indianapolis: Addison-Wesley Professional, 2014. – 792 с.
18. Калдареллі Ф. Yii2 By Example / Ф. Калдареллі. – Birmingham: Packt Publishing, 2016. – 332 с.

ДОДАТКИ

					ПЗ.КР.Б – 121 – 23 – ПЗ	
						56

Лістинг контролера HomeworkController:

```
<?php
namespace frontend\controllers;

use frontend\models\Homework;
use frontend\models\HomeworkSearch;
use frontend\models\HomeworkSubmissionSearch;
use frontend\models\Lesson;
use Yii;
use yii\web\Controller;
use yii\web\NotFoundHttpException;
use yii\filters\VerbFilter;
use yii\web\UploadedFile;

/**
 * HomeworkController implements the CRUD actions for Homework model.
 */
class HomeworkController extends Controller
{
    /**
     * @inheritdoc
     */
    public function behaviors()
    {
        return array_merge(
            parent::behaviors(),
            [
                'verbs' => [
                    'class' => VerbFilter::className(),
                    'actions' => [
                        'delete' => ['POST'],
                    ],
                ],
            ]
        );
    }

    /**
     * Lists all Homework models.
     *
     * @return string
     */
}
```

```

        */
public function actionIndex()
{
    $searchModel = new HomeworkSearch();
    $dataProvider = $searchModel->search($this->request->queryParams);

    return $this->render('index', [
        'searchModel' => $searchModel,
        'dataProvider' => $dataProvider,
    ]);
}

/**
 * Displays a single Homework model.
 * @param int $id ID
 * @return string
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionView($id)
{
    $searchModel = new HomeworkSubmissionSearch();

    $params = $this->request->queryParams;
    $params['HomeworkSubmissionSearch']['homework_id'] = $id;
    $dataProvider = $searchModel->search($params);

    return $this->render('view', [
        'model' => $this->findModel($id),
        'dataProvider' => $dataProvider,
    ]);
}

/**
 * Creates a new Lesson model.
 * @var UploadedFile[]
 * If creation is successful, the browser will be redirected to the 'view'
page.
 * @return string|\yii\web\Response
 */
public function actionCreate($lesson_id)
{
    $model = new Homework();
    $lesson = Lesson::findOne(['id' => $lesson_id]);

```

```

        if ($this->request->isPost) {
            if ($model->load(Yii::$app->request->post())) {
                $model->lesson_id = $lesson_id;
                $model->file = UploadedFile::getInstance($model, 'file');
            }
            if ($homework = $model->saveHomework()) {
                return $this->redirect(['view', 'id' => $homework->id]);
            }
        } else {
            $model->loadDefaultValues();
        }

        return $this->render('create', [
            'model' => $model,
            'lesson' => $lesson
        ]);
    }

    public function actionDownload($id)
    {
        $model = $this->findModel($id);
        $filePath = $model->getFolder() . $model->file;

        if (file_exists($filePath)) {
            return Yii::$app->response->sendFile($filePath, $model->file);
        } else {
            throw new NotFoundHttpException('The requested file does not
exist.');
```

}
 }
 /**
 * Updates an existing Homework model.
 * If update is successful, the browser will be redirected to the 'view'
page.
 * @param int \$id ID
 * @return string|\yii\web\Response
 * @throws NotFoundHttpException if the model cannot be found
 */
 public function actionUpdate(\$id)
 {
 \$model = \$this->findModel(\$id);
 \$oldFile = \$model->file; // Зберегти старий файл

```

        if ($model->load(Yii::$app->request->post())) {
            $model->file = UploadedFile::getInstance($model, 'file');

            // Перевіряємо, чи вибрано новий файл
            if ($model->file) {
                if ($model->validate()) {
                    // Видаляємо старий файл
                    if ($oldFile) {
                        unlink($model->getFolder() . $oldFile);
                    }
                    // Зберігаємо новий файл
                    $model->file = $model->upload();
                } else {
                    // Якщо виникли помилки при валідації нового файлу
                    // Залишаємо попередній файл без змін
                    $model->file = $oldFile;
                }
            } else {
                // Якщо не обрано новий файл, зберігаємо попередній файл
                $model->file = $oldFile;
            }

            if ($model->save()) {
                return $this->redirect(['view', 'id' => $model->id]);
            }
        }
        return $this->render('update', [
            'model' => $model
        ]);
    }

    /**
     * Deletes an existing Homework model.
     * If deletion is successful, the browser will be redirected to the 'index'
    page.
     * @param int $id ID
     * @return \yii\web\Response
     * @throws NotFoundHttpException if the model cannot be found
     */
    public function actionDelete($id)
    {
        $homework = $this->findModel($id);
        $lesson_id = $homework->lesson->id;

```

```

        $this->findModel($id)->delete();
        return $this->redirect(['lesson/view', 'id' => $lesson_id]);
    }

    /**
     * Finds the Homework model based on its primary key value.
     * If the model is not found, a 404 HTTP exception will be thrown.
     * @param int $id ID
     * @return Homework the loaded model
     * @throws NotFoundHttpException if the model cannot be found
     */
    protected function findModel($id)
    {
        if (($model = Homework::findOne(['id' => $id])) !== null) {
            return $model;
        }
        throw new NotFoundHttpException('The requested page does not exist.');
    }
}

```

Лістинг моделі Homework:

```
<?php

namespace frontend\models;

use Yii;
use yii\behaviors\TimestampBehavior;
use yii\db\ActiveRecord;
use yii\web\UploadedFile;
/**
 * This is the model class for table "homework".
 *
 * @property int $id
 * @property int $lesson_id
 * @property string $description
 * @property string|null $file
 * @property string $created_at
 * @property string $updated_at
 * @property string|null $finish_date
 *
 * @property HomeworkSubmission[] $homeworkSubmissions
 * @property Lesson $lesson
 */
class Homework extends \yii\db\ActiveRecord
{
    /**
     * {@inheritdoc}
     */
    public static function tableName()
    {
        return 'homework';
    }
    /**
     * {@inheritdoc}
     */
    public function behaviors()
    {
        return [
            [
                'class' => TimestampBehavior::class,
                'attributes' => [
                    ActiveRecord::EVENT_BEFORE_INSERT => ['created_at',
```

```

        'updated_at'],

        ActiveRecord::EVENT_BEFORE_UPDATE => ['updated_at'],

    ],

    'value' => function () {

        return date('Y-m-d H:i:s');

    },

],

];

}

public function rules()

{

    return [

        [['lesson_id', 'description'], 'required'],

        [['lesson_id'], 'integer'],

        [['description'], 'string'],

        [['finish_date'], 'safe'],

        ['finish_date', 'datetime', 'format'=>'php:Y-m-d H:i'],

        [['lesson_id'], 'unique'],

        [['lesson_id'], 'exist', 'skipOnError' => true, 'targetClass' =>

Lesson::class, 'targetAttribute' => ['lesson_id' => 'id']],

        [['file'], 'file', 'skipOnEmpty' => true, 'extensions' => 'pdf, doc,

docx, ppt, pptx, xls,xlsx']

    ];

}

public function attributeLabels()

{

    return [

        'lesson_id' => 'Урок',

        'description' => 'Опис',

        'file' => 'Файл',

        'created_at' => 'Створено',

        'updated_at' => 'Оновлено',

        'finish_date' => 'Термін здачі',

    ];

}

/**

 * Gets query for [[HomeworkSubmissions]].

 * @return \yii\db\ActiveQuery

 */

public function getHomeworkSubmissions()

{

```

```

        return $this->hasMany(HomeworkSubmission::class, ['homework_id' =>
'id']);
    }
    /**
     * Gets query for [[Lesson]].
     * @return \yii\db\ActiveQuery
     */
    public function getLesson()
    {
        return $this->hasOne(Lesson::class, ['id' => 'lesson_id']);
    }
    public function getFolder()
    {
        return Yii::getAlias('@frontend') . '/web/documents/homework/';
    }
    public function upload()
    {
        if ($this->file instanceof UploadedFile) {
            $fileName = $this->file->baseName . '.' . $this->file->extension;
            $filePath = $this->getFolder() . $fileName;
            $this->file->saveAs($filePath);
            return $fileName;
        }
        return false;
    }
    public function saveHomework()
    {
        if ($this->validate()) {
            $homework = new \frontend\models\Homework();
            $homework->lesson_id = $this->lesson_id;
            $homework->description = $this->description;
            $homework->file = $this->upload();
            $homework->finish_date = $this->finish_date;
            if ($homework->save()) {
                return $homework;
            } else {
                Yii::error('Помилка при збереженні домашнього завдання.',
$homework->errors);
            }
        }
        return null;
    }
}

```

					ІІЗ.КР.Б – 121 – 23 – ІІЗ	
						64

Лістинг моделі HomeworkSearch:

```
<?php

namespace frontend\models;

use Yii;
use yii\base\Model;
use yii\data\ActiveDataProvider;
use frontend\models\Homework;

/**
 * HomeworkSearch represents the model behind the search form of
 * `frontend\models\Homework`.
 */
class HomeworkSearch extends Homework
{
    public $lessonTitle;
    public $subject_id;
    public $status;

    public $statusOptions = [
        'checked' => 'Перевірено',
        'awaiting' => 'Очікує перевірки',
        'not_submitted' => 'Не виконане завдання',
    ];

    /**
     * {@inheritdoc}
     */
    public function rules()
    {
        return [
            [['lessonTitle'], 'string'],
            [['id', 'lesson_id', 'subject_id'], 'integer'],
            [['description', 'file', 'created_at', 'updated_at', 'finish_date'],
            'safe'],
            [['status'], 'in', 'range' => array_keys($this->statusOptions)],
        ];
    }

    /**
     * {@inheritdoc}
     */
}
```

					ІІЗ.КР.Б – 121 – 23 – ІІЗ	
						65

```

    */
public function scenarios()
{
    // bypass scenarios() implementation in the parent class
    return Model::scenarios();
}

/**
 * Creates data provider instance with search query applied
 *
 * @param array $params
 *
 * @return ActiveDataProvider
 */
public function search($params)
{
    $query = Homework::find();

    $dataProvider = new ActiveDataProvider([
        'query' => $query,
    ]);

    $this->load($params);

    if (!$this->validate()) {
        return $dataProvider;
    }

    // grid filtering conditions
    $query->andWhere([
        'id' => $this->id,
        'created_at' => $this->created_at,
        'updated_at' => $this->updated_at,
        'finish_date' => $this->finish_date,
    ]);

    $query->joinWith(['lesson', 'lesson.subject']);
    $query->andWhere(['like', 'lesson.title', $this->lessonTitle]);
    $query->andWhere(['lesson.subject_id' => $this->subject_id]);

    $query->andWhere(['like', 'description', $this->description])
        ->andWhere(['like', 'file', $this->file]);

```

```

$query->leftJoin('homework_submission', 'homework_submission.homework_id
= homework.id');

if ($this->status === 'checked') {
    $query->andWhere(['>', 'homework_submission.grade', 0]);
} elseif ($this->status === 'awaiting') {
    $query->andWhere(['AND',
        ['IS', 'homework_submission.grade', null],
        ['IS NOT', 'homework_submission.id', null]]);
} elseif ($this->status === 'not_submitted') {
    $query->andWhere(['IS', 'homework_submission.id', null]);
}

if ($params['HomeworkSearch']['student_id']) {
    $class_id = ClassStudent::findOne(['student_id' =>
$params['HomeworkSearch']['student_id']])->class_id;
    $subject_id = SubjectClass::find()->select('subject_id')
        ->where(['class_id' => $class_id])->column();
    $query->andFilterWhere([
        'lesson_id' => Lesson::find()
            ->select('id')
            ->where(['subject_id' => $subject_id])
    ]);
}

$dataProvider->sort->attributes['grade'] = [
    'asc' => ['homework_submission.grade' => SORT_ASC],
    'desc' => ['homework_submission.grade' => SORT_DESC],
];

return $dataProvider;
}

public function getAvailableSubjects()
{
    $class_id = ClassStudent::findOne(['student_id' => Yii::$app->user-
>getId()])->class_id;

    $subjectIds = SubjectClass::find()
        ->select('subject_id')
        ->where(['class_id' => $class_id])
        ->column();

```

```

        $subjects = Subject::find()
            ->where(['id' => $subjectIds])
            ->all();

        return \yii\helpers\ArrayHelper::map($subjects, 'id', 'title');
    }

    public function getStatusOptions()
    {
        return $this->statusOptions;
    }
}

```