Міністерство освіти і науки України
Криворізький національний університет
Факультет інформаційних технологій
Кафедра автоматизації, комп'ютерних наук і технологій

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти − бакалавр
за освітньо-професійною програмою
«Комп'ютерні науки»
зі спеціальності
122 – Комп'ютерні науки

Тема роботи:
«**Розробка інформаційної системи управління навчанням для навчального закладу з вивчення іноземних мов**»

Виконав студент гр. КН-20 _____ Мізюкін П. В.

Керівник _____ Харламенко В. Ю.

Нормоконтроль _____ Маринич І. А.

Завідувач кафедри _____ Рубан С. А.

Кривий Ріг – 2024

КРИВОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет: *інформаційних технологій*

Кафедра: *автоматизації, комп'ютерних наук і технологій*

Ступінь вищої освіти: *Бакалавр*

Спеціальність: *122 – Комп'ютерні науки*

ЗАТВЕРДЖУЮ

Зав. кафедрою: *к.т.н. Рубан С.А.*

_____

« *27* » *березня* 2024 р.

ЗАВДАННЯ

на кваліфікаційну роботу бакалавра

студентові групи КН-20 *Мізюкіну Павлу Володимировичу*

1. Тема кваліфікаційної роботи: *«Розробка інформаційної системи управління навчанням для навчального закладу з вивчення іноземних мов»*

затверджено наказом по університету № *235с* від *27.03.2024 р.*

2. Термін здачі кваліфікаційної роботи: *05.06.2024 р.*

3. Склад кваліфікаційної роботи: *Пояснювальна записка обсягом 63с., додатки, презентація у Microsoft PowerPoint (17 слайдів) в електронному та друкованому вигляді*

4. Консультанти кваліфікаційної роботи:

Розділ 1-3 ст. викладач Харламенко В.Ю.

_____

Нормоконтроль доц. Маринич І. А.

5. Календарний план:

№	Етапи роботи	Термін виконання
1	*Вступ*	29.01.2024 – 07.02.2024
2	*Розділ 1*	07.02.2024 – 20.03.2024
3	*Розділ 2*	20.03.2024 – 10.05.2024
4	*Висновки*	10.05.2024 – 15.05.2024
5	*Оформлення кваліфікаційної роботи*	15.05.2024 – 31.05.2024
6	*Підготовка презентації та графічного матеріалу*	01.06.2024 – 05.06.2024
7	*Підготовка доповіді до захисту*	05.06.2024 – 10.06.2024

6. Дата видачі завдання: *29.01.2024р.*

Керівник _____ /Харламенко В. Ю./

7. Запевнення: *Я, Мізюкін Павло Володимирович, запевняю, що ця кваліфікаційна робота виконана самостійно, не містить академічного плагіату, фабрикації, фальсифікації. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.*

Із чинним Положенням про академічну доброчесність Криворізького національного університету ознайомлений.

Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі умисних порушень робота не допускається до захисту або оцінюється незадовільно.

Студент _____ / Мізюкін П.В./

АНОТАЦІЯ

Мізюкін П.В. Розробка системи управління навчальним процесом для навчального закладу.

Кваліфікаційна робота на здобуття ступеня вищої освіти – бакалавр, за спеціальністю 122 Комп'ютерні науки. Криворізький національний університет, Кривий Ріг, 2024.

Робота складається зі вступу, двох розділів, висновків, переліку використаної літератури з 18 позицій та 17 додатки. Загальний обсяг роботи становить 63 сторінок, з яких основний зміст роботи викладено на 48 сторінках, включає 12 таблиць і 25 рисунків.

В роботі було розроблено веб-додаток який представляє собою систему управління навчанням LMS (Learning Management System) – програмне забезпечення яке дозволяє керувати навчальним процесом у освітньому закладі. У даній роботі реалізовано функції ведення електронного журналу викладачем (вибір групи, відмічання заняття, додавання коментарів учням, редагування записів у журналі, виставлення оцінок та інше), завантаження навчальних матеріалів викладачем, перегляд розкладу, перегляд новин навчального закладу. Для моделювання функції системи було розроблено UML діаграми: Use Case та діаграму класів. Для ознайомлення з навігацією по сайту, розроблено мапу сайту. Для розробки були використані технології ASP .NET Core – для реалізації серверного функціоналу, Entity Framework – для взаємодії з базою даних Microsoft SQL Server, Razor Pages та Bootstrap – для реалізації клієнтської частини. Для зручності опанування додатку було розроблено інструкцію користувача з детальним описом користувацьких екранів. Розроблений додаток має практичну цінність і використовується у навчальному процесі одного із освітніх закладів.

LMS, ASP. NET CORE, ENTITY FRAMEWORK, БД, SQL, HTML, CSS

ЗМІСТ

ВСТУП

У сучасному світі зростає значення освіти як ключового елементу розвитку суспільства. Зміни в економіці, технологіях та ринку праці ставлять перед нами завдання постійного навчання та адаптації. Системи управління навчальним процесом (Learning Management System, далі LMS) стають невід'ємною частиною освітнього процесу, дозволяючи оптимізувати та поліпшувати навчання.

Традиційні методи навчання часто обмежені просторово та часово. Завдяки глобалізації, студенти та працівники можуть знаходитися в різних кутках світу, і традиційні підходи втрачають свою ефективність. LMS дозволяють навчатися та тренуватися в будь-який зручний час та місце, забезпечуючи доступність навчальних ресурсів для всіх.

Зростання технологічного розвитку ставить перед сучасними системами освіти виклик у впровадженні та підтримці новітніх методів та інструментів. LMS допомагають у впровадженні інтерактивних та інноваційних методів навчання, сприяючи підготовці кадрів, які готові до викликів та потреб ринку праці.

LMS дозволяють створювати індивідуальні програми для кожного учасника, враховуючи його потреби та рівень знань. Це сприяє кращому розумінню матеріалу та підвищує загальну ефективність навчання.

LMS дозволяють ефективно відстежувати та аналізувати прогрес кожного учасника. Це надає можливість вчителям та тренерам оперативно реагувати на індивідуальні труднощі та потреби студентів, а також оцінювати ефективність самого навчального процесу.

Головною метою роботи роботи є створення інноваційної та ефективної платформи для навчання, яка відповідає сучасним вимогам освіти та ринку праці. Ця система покликана забезпечити доступність, гнучкість та персоналізацію навчання, сприяючи розвитку навичок та знань, необхідних для успішної адаптації у сучасному світі.

РОЗДІЛ 1. АНАЛІЗ ПРОГРАМНОГО ОБ'ЄКТА ТА ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

1.1 Основне призначення

Система управління навчанням (LMS) — це програмне забезпечення або веб-платформа, яка використовується для організації, проведення та оцінювання навчального процесу. Вона застосовується для електронного навчання і зазвичай складається з двох компонентів: сервера, що виконує основні функції, та користувацького інтерфейсу (UI), який використовують викладачі, студенти та адміністратори.

LMS зазвичай надає викладачам можливість створювати та доставляти навчальні матеріали, відстежувати активність учнів і оцінювати їхні результати. Вона також може забезпечувати інтерактивні функції для учнів, такі як обговорення тем, відеоконференції та форуми для дискусій.

Такі системи широко використовуються в компаніях, державних установах, а також у традиційних і онлайн-освітніх закладах. Вони можуть підвищити ефективність традиційних методів навчання, а також зекономити час і ресурси організацій. Добре функціонуюча система дозволяє викладачам і адміністраторам ефективно керувати такими аспектами, як реєстрація користувачів, доступ до матеріалів, планування, комунікація, проведення тестів, сертифікація та сповіщення.

Системи управління навчальним процесом, LMS (Learning Management Systems) призначені для підтримки та оптимізації освітнього процесу за допомогою технологій.

Нижче, на рис. 1.1 зображено типову структуру LMS.

Рисунок 1.1 – Структура LMS

Основне призначення таких систем зазвичай, має такі ключові аспекти:

- Організація та управління курсами: LMS дозволяє створювати та структурувати курси, організовувати навчальні матеріали та ресурси, планувати навчальну діяльність і встановлювати дедлайни.
- Дистанційне навчання: Платформи LMS підтримують дистанційне навчання, надаючи студентам доступ до навчальних матеріалів і завдань в будь-який час і з будь-якого місця з доступом до Інтернету.
- Комунікація та співпраця: LMS забезпечують засоби для спілкування між студентами і викладачами через форуми, чати, обговорення та відеоконференції, що сприяє активній взаємодії та співпраці.
- Оцінювання і зворотний зв'язок: Системи управління навчальним процесом включають інструменти для створення тестів, вікторин, завдань і інших форм оцінювання. Викладачі можуть швидко і зручно оцінювати роботу студентів, надавати коментарі та зворотний зв'язок.
- Моніторинг і аналітика: LMS надають інструменти для відстеження прогресу студентів, аналізу їх успішності, а також створення звітів про активність студентів і ефективність навчання.

- Управління контентом: Системи LMS дозволяють завантажувати, зберігати та організовувати навчальні матеріали в різних форматах (текстові документи, відео, аудіо, зображення тощо), забезпечуючи їх зручний доступ для студентів.
- Інтеграція з іншими системами: Багато LMS підтримують інтеграцію з іншими освітніми і адміністративними системами, такими як бібліотеки, електронні журнали, фінансові системи, що забезпечує єдине освітнє середовище.
- Персоналізація навчання: LMS можуть пропонувати адаптивне навчання, де контент і завдання підлаштовуються під індивідуальні потреби та рівень знань кожного студента, сприяючи більш ефективному навчанню.
- Адміністративне управління: Платформи LMS допомагають з адміністративними завданнями, такими як реєстрація студентів, ведення журналів, управління календарем та інші організаційні аспекти навчального процесу.[1-2]

1.2 Сучасні рішення управління навчальним процесом

Moodle – це платформа для дистанційного навчання, яка використовується для створення онлайн-курсів і управління навчальним процесом.

Основні цілі та функції Moodle включають:

- Управління курсами: Moodle дозволяє створювати, редагувати та адмініструвати курси, завантажувати навчальні матеріали, організовувати навчальний процес і відстежувати прогрес студентів.
- Дистанційне навчання: Платформа забезпечує можливість проводити навчання онлайн, що дозволяє студентам і викладачам працювати незалежно від їхнього місцезнаходження.

- Комунікація та співпраця: Moodle надає інструменти для взаємодії між студентами та викладачами, такі як форуми, чати, відеоконференції та групові проекти.
- Оцінювання і зворотний зв'язок: Викладачі можуть створювати різні типи завдань і тестів, оцінювати роботи студентів, а також надавати коментарі та зворотний зв'язок.
- Модульність і розширюваність: Moodle підтримує різноманітні плагіни та модулі, що дозволяє розширювати функціональні можливості платформи відповідно до потреб освітнього закладу.
- Аналітика та звіти: Платформа надає інструменти для аналізу даних про успішність студентів, що дозволяє викладачам і адміністраторам краще розуміти та покращувати навчальний процес.

Завдяки цим функціям Moodle широко використовується у школах, університетах та інших освітніх установах для організації та підтримки навчального процесу.

На рис. 1.2 наведено зразок інтерфейсу користувача системи Moodle.

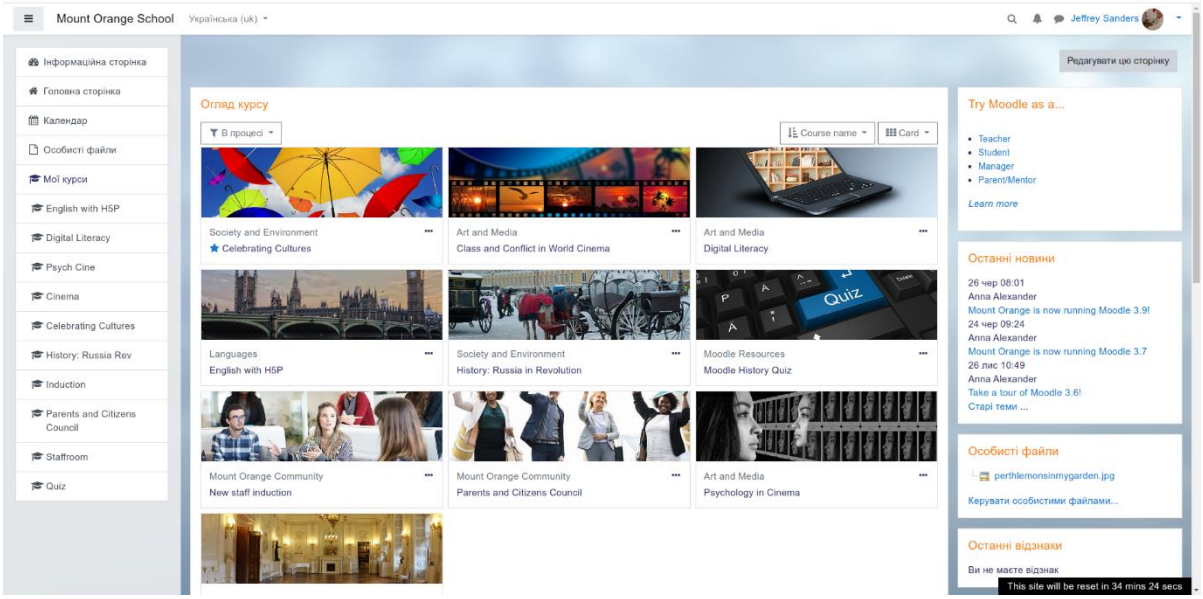

Рисунок 1.2 – Інтерфейс користувача системи Moodle

Проаналізувавши функціонал та режими роботи системи управління навчальним процесом Moodle можна визначити наступні переваги та недоліки

Переваги:

- Відкрите програмне забезпечення.
- Гнучкість та можливості налаштувань.
- Велика спільнота користувачів.
- Різноманітні інструменти для співпраці та оцінювання.

Недоліки:

- Складний інтерфейс для адміністраторів.
- Може вимагати додаткових зусиль для налаштування та оновлення.[3]

Google Classroom – це безкоштовна веб-платформа, розроблена компанією Google, яка використовується для спрощення створення, розподілу та оцінювання завдань у навчальному процесі.

Основні цілі та функції Google Classroom включають:

- Організація навчального процесу: Classroom допомагає викладачам створювати та структурувати курси, завантажувати навчальні матеріали, розміщувати завдання, а також відстежувати прогрес учнів.
- Комунікація та співпраця: Платформа надає інструменти для взаємодії між студентами та викладачами через коментарі, сповіщення та обговорення, що сприяє кращому розумінню матеріалу і співпраці.
- Ефективне розподілення завдань: Викладачі можуть створювати та роздавати завдання, тести та інші навчальні матеріали онлайн, а студенти – здавати їх, використовуючи Google Docs, Sheets, Slides та інші інструменти Google.
- Оцінювання та зворотний зв'язок: Викладачі можуть швидко та зручно оцінювати роботи студентів, надавати зворотний зв'язок і коментарі, що сприяє поліпшенню навчального процесу.
- Інтеграція з іншими сервісами: Google: Classroom інтегрується з іншими інструментами Google, такими як Google Drive, Calendar і Gmail, що

робить роботу з матеріалами та завданнями більш зручною та ефективною.

- Доступність та мобільність: Оскільки Classroom є веб-платформою, вона доступна з будь-якого пристрою з підключенням до Інтернету, що дозволяє студентам і викладачам працювати в будь-який час і в будь-якому місці.
- Безпека та конфіденційність: Google Classroom забезпечує високу рівень безпеки та захисту даних користувачів, що важливо для освітніх закладів.

На рис. 1.3 наведено зразок користувацького інтерфейсу Google Classroom.

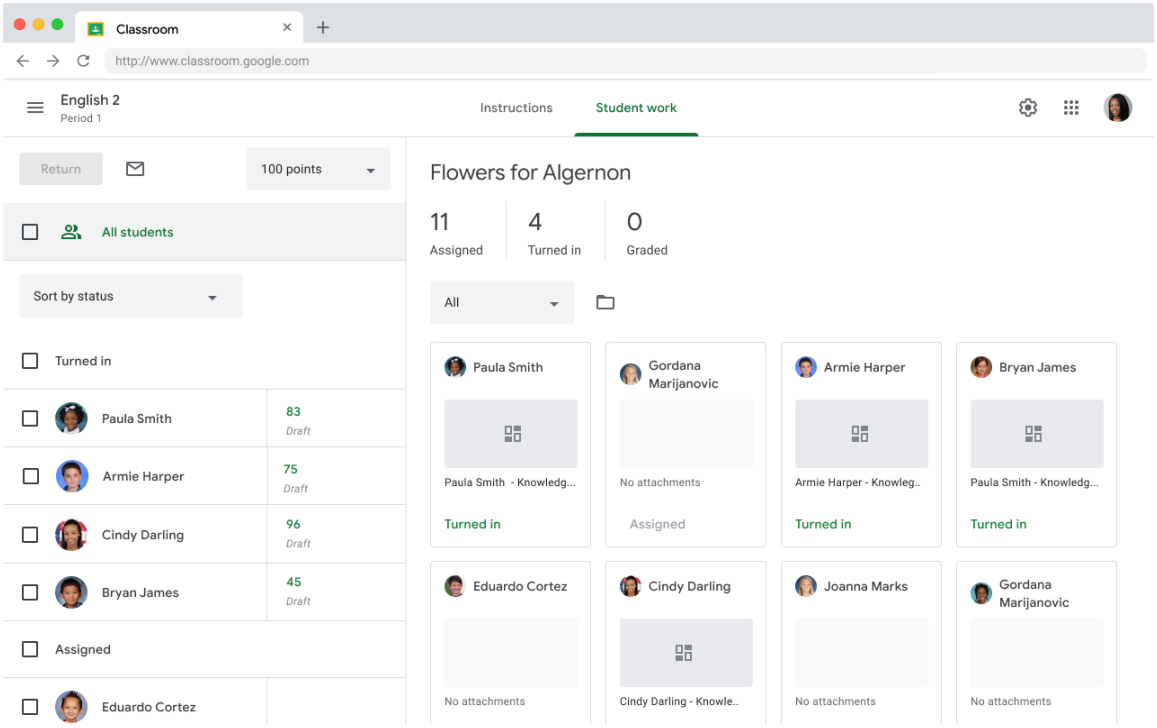

Рисунок 1.3 – Інтерфейс користувача системи Google Classroom

Переваги:

- Простота використання: Інтуїтивно зрозумілий інтерфейс, що не потребує спеціальних навичок для роботи.

- Економія часу*:* Автоматизація багатьох процесів, таких як роздача завдань та збір відповідей, що дозволяє викладачам зосередитися на навчанні.
- Централізоване управління: Можливість управління всіма аспектами курсу в одному місці.

Недоліки:

- Обмеження в організації навчального матеріалу*:* Можливості організації і структуризації навчальних матеріалів можуть бути обмеженими, що може ускладнювати навігацію по курсу для студентів.
- Відсутність повної підтримки для різних педагогічних підходів: Google Classroom може бути не повністю адаптованим для підтримки певних педагогічних методів або навчальних стратегій, таких як проектне навчання або проблемно-орієнтоване навчання.

Виконано аналіз існуючих рішень у галузі систем управління навчальним процесом, що дало змогу визначити основні функціональні та нефункціональні вимоги до запропонованого у дипломі додатку.[4]

1) МКР – представляє програмний комплекс "Автоматизована система управління навчальним закладом" (АСУ НЗ). Ця система забезпечує інтегроване управління вищим навчальним закладом, включаючи модулі для навчального процесу, деканату, абітурієнтів, методичного відділу, відділу кадрів тощо. Вона також включає веб-портал для відображення розкладів, успішності, навчальних планів, оплати за гуртожиток, тестування студентів та інших функцій.

На рис. 1.4 наведено зразок користувацького інтерфейсу МКР.

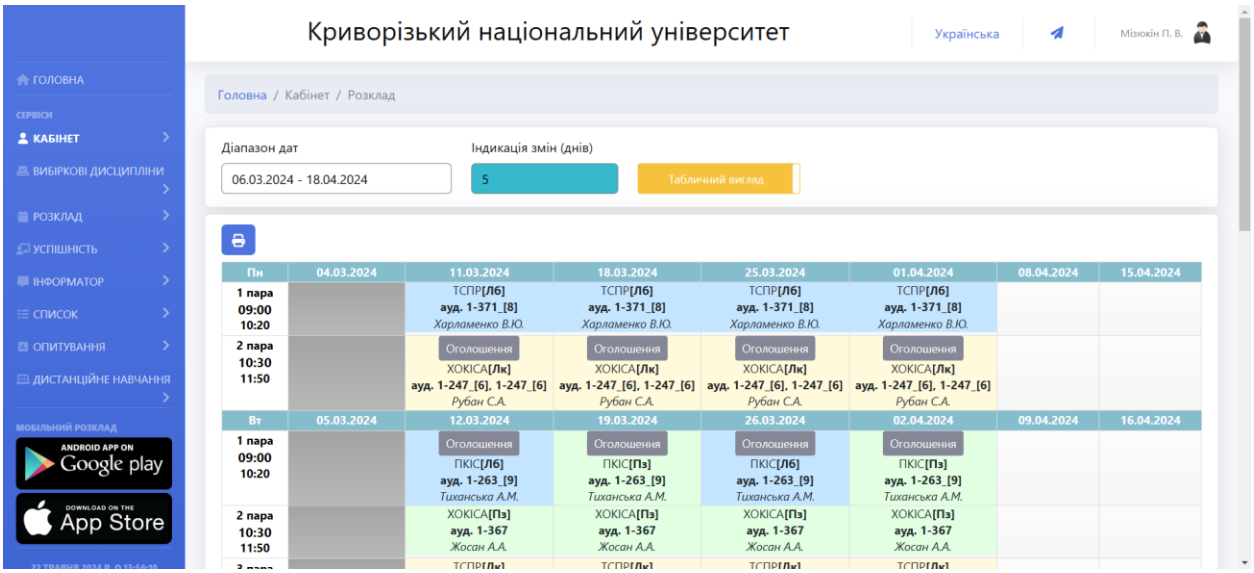

Рисунок 1.4 – Зразок інтерфейсу користувача платформи МКР

Основні функції:

- Моніторинг успішності студентів.
- Управління розкладом занять.
- Тестування студентів.
- Формування друкованих і статистичних форм.

Переваги:

- Інтеграція різних функцій управління в єдину систему.
- Зручність доступу до інформації через веб-портал.
- Можливість створення індивідуальних навчальних траєкторій.
- Незалежність від розробників завдяки інструментам для створення форм.

Недоліки:

- Можливі проблеми з налаштуванням та адаптацією до специфічних потреб окремих навчальних закладів.
- Потреба у навчанні персоналу для ефективного використання всіх функцій системи.
- Залежність від надійності інтернет-з'єднання для доступу до веб-порталу.[5]

Blackboard LMS (Learning Management System) — це платформа для управління навчанням, яка широко використовується в освітніх закладах і для корпоративного навчання. Вона надає комплексний набір інструментів і функцій для підтримки навчальних процесів в онлайн і змішаних форматах. На рис. 1.5 зображено скрін інтерфейсу коритувача системи Blackboard LMS.

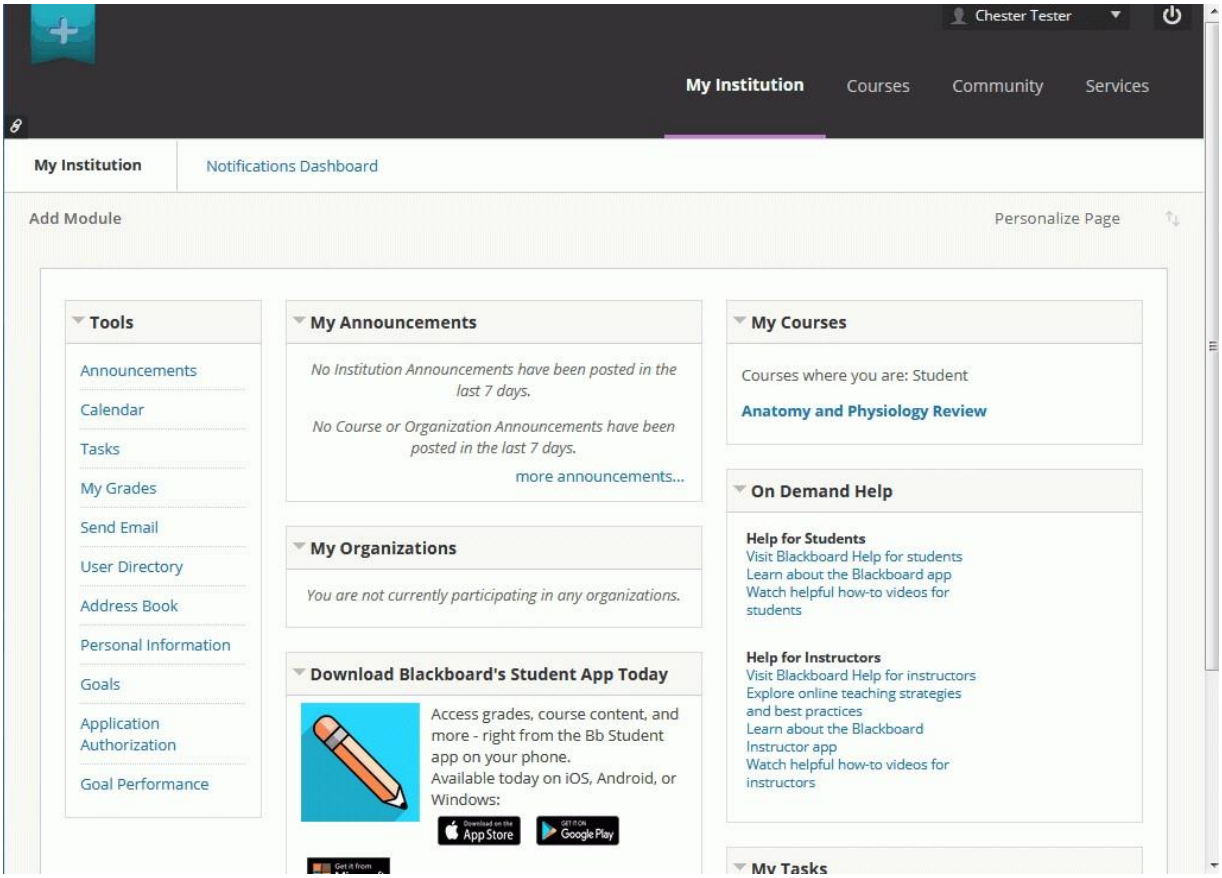

Рисунок 1.5 – Зразок інтерфейсу користувача

Основні призначення Blackboard LMS:

- Управління курсами: Дозволяє викладачам створювати та керувати навчальними матеріалами, такими як лекції, завдання, тести та іспити. Організовує навчальні ресурси у структурованому вигляді.
- Комунікація та співпраця: Надає різні інструменти для комунікації, такі як дискусійні форуми, повідомлення та оголошення для покращення взаємодії між студентами та викладачами. Підтримує групову роботу та співпрацю через групові завдання, вікі та блоги.

- Оцінювання та зворотний зв'язок: Дозволяє створювати та адмініструвати різні види оцінювання, включаючи тести, опитування та анкети. Надає інструменти для оцінювання та надання зворотного зв'язку, включаючи рубрики та функції вбудованого оцінювання.
- Відстеження та аналітика: Включає інструменти для моніторингу прогресу та успішності студентів через детальні аналітичні та звітні інструменти. Допомагає викладачам визначати студентів, які потребують додаткової підтримки.
- Інтеграція та налаштування: Підтримує інтеграцію з іншими освітніми технологіями та інструментами, такими як програми для перевірки плагіату, сторонні постачальники контенту та зовнішні бази даних. Пропонує можливості для налаштування середовища навчання відповідно до специфічних потреб закладу або курсу.

Переваги:

- Покращений навчальний досвід: Забезпечує багатий набір інструментів для створення залучаючих та інтерактивних навчальних досвідів.
- Гнучкість і зручність: Підтримує різні форми навчання, включаючи повністю онлайн, змішані та гібридні курси.
- Ефективність адміністрування: Спрощує управління курсами та адміністративні завдання, економлячи час викладачів.
- Покращення успішності студентів: Надає інструменти для персоналізованого навчання та раннього втручання, що сприяє кращій академічній успішності.

Недоліки:

- Складність: Великий набір функцій може бути складним для нових користувачів, вимагаючи навчання та підтримки для ефективного використання.
- Вартість: Впровадження та ліцензійні платежі можуть бути високими, що може стати бар'єром для менших навчальних закладів.

- Інтерфейс користувача: Деякі користувачі знаходять інтерфейс менш інтуїтивним порівняно з новішими платформами LMS.[6]

1.3 Задача розробки та основні вимоги

Задача даного проєкта є розробка LMS, метою якої є автоматизація навчальних процесів приватного освітнього закладу.

Для розробки запропонованої системи були розроблені наступні вимоги:

Вимоги до функціональних характеристик:

Бізнес вимоги:

- Аналітика та звітність: LMS повинна забезпечувати інструменти для аналізу навчальних даних та статистики успішності студентів. Це допоможе вчителям та адміністрації школи відстежувати прогрес студентів, виявляти слабкі місця та розробляти ефективні стратегії навчання.
- Безпека та конфіденційність*:* Оскільки LMS буде містити конфіденційну інформацію про студентів та їхні досягнення, система повинна забезпечувати надійний захист даних та персональну інформацію від несанкціонованого доступу
- Онлайн керування та управління процесом навчання: Система має вирішувати задачі налагодження процесів керування та управління навчальним процесом, які можуть полегшити та вдосконалити роботу та ефективність адміністрації та вчителів.

Вимоги користувачів:

- Зручний та інтуїтивно зрозумілий інтерфейс*:* Користувачі очікують, що інтерфейс системи буде легким у використанні та інтуїтивно зрозумілим. Навігація повинна бути зрозумілою, а основні функції – доступними на перший погляд.

- Мобільна сумісність: У зв’язку зі зростанням використання мобільних пристроїв, користувачі очікують, що система буде доступною та зручною для використання на різних мобільних платформах (смартфони, планшети). Це включає в розробку адаптивної веб-версії для мобільних пристроїв.

Склад основних функцій:

- Функція авторизації та аутентифікації: Система повинна забезпечувати можливість входу в систему для учнів, вчителів та адміністраторів за допомогою ідентифікаційних даних, таких як ім’я користувача та пароль.
- Електронний журнал: Система повинна мати функцію ведення електронного журналу, функіонал такий як «Відмітити присутніх на занятті»
- Створення та додавання вчителем матеріалів курсу: Система повинна дозволяти вчителям створювати, редагувати та керувати курсами. Кожен курс повинен мати можливість включати різноманітні матеріали, завдання та тести.
- Система оцінювання: LMS повинна мати інтегровану систему оцінювання завдання тести та інші форми оцінки знань студентів.
- Розклад (календар): Система повинна забезпечувати можливість відстеження розкладу уроків та важливих подій у календарі.

Вимоги до нефункціональних характеристик:

- Сумісність з різними платформами: Система повинна бути сумісною з різними операційними системами та пристроями, такими як ПК, планшети та мобільні телефони. Це забезпечить учням та вчителям доступ до курсів з будь-якого пристрою.
- Гнучкість налаштувань: LMS повинна мати гнучкі налаштування, що дозволяють адміністраторам школи налаштовувати та персоналізувати систему під конкретні потреби заклади освіти.

- Підтримка інтерактивних матеріалів: Система повинна підтримувати різноманітні інтерактивні матеріали для навчання, такі як відео-уроки, аудіозаписи, інтерактивні тести тощо.
- Забезпечення зручної навігації: Система повинна мати зручну навігацію, яка дозволяє користувачам швидко знаходити потрібну інформацію та матеріали курсу. Це може включати в себе структуровану систему меню та швидкий пошук.

Висновки до розділу

У даному розділі виконано аналіз існуючих рішень у сфері автоматизації навчального процесу у приватних навчальних закладах. Було проаналізовано Moodle, Google Classroom, автоматизовану систему управління навчальним закладом (АСУ НЗ), Blackboard LMS.

Застосування подібних систем значно підвищує ефективність ведення навчального процесу, що призводить до підвищення якості навчання, покращення успішності учнів, а також мають інтуїтивно зрозумілий інтерфейс, що не потребує спеціальних навичок для роботи. Але, не дивлячись на очевидні переваги, дані системи мають один значний недолік – це висока вартість та складність функціоналу, навантаженість надмірним функціоналом.

За результатами проведеного аналізу були сформовані задачі розробки та вимоги (функціональні та нефункціональні) до системи, що розроблюється.

РОЗДІЛ 2. РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ НАВЧАЛЬНИМ ПРОЦЕСОМ ТА ЇЇ ПРОГРАМНО-ТЕХНІЧНА РЕАЛІЗАЦІЯ

2.1 Діаграма варіантів використання (Use Case)

Діаграма варіантів використання (Use Case) використовується в процесі розробки програмного забезпечення для опису функціональних вимог до системи з погляду користувача. Вона допомагає визначити, як різні користувачі (актори) взаємодіятимуть із системою. На рис 2.1 продемонстровано, як різні користувачі взаємодіють з LMS системою освітнього закладу.

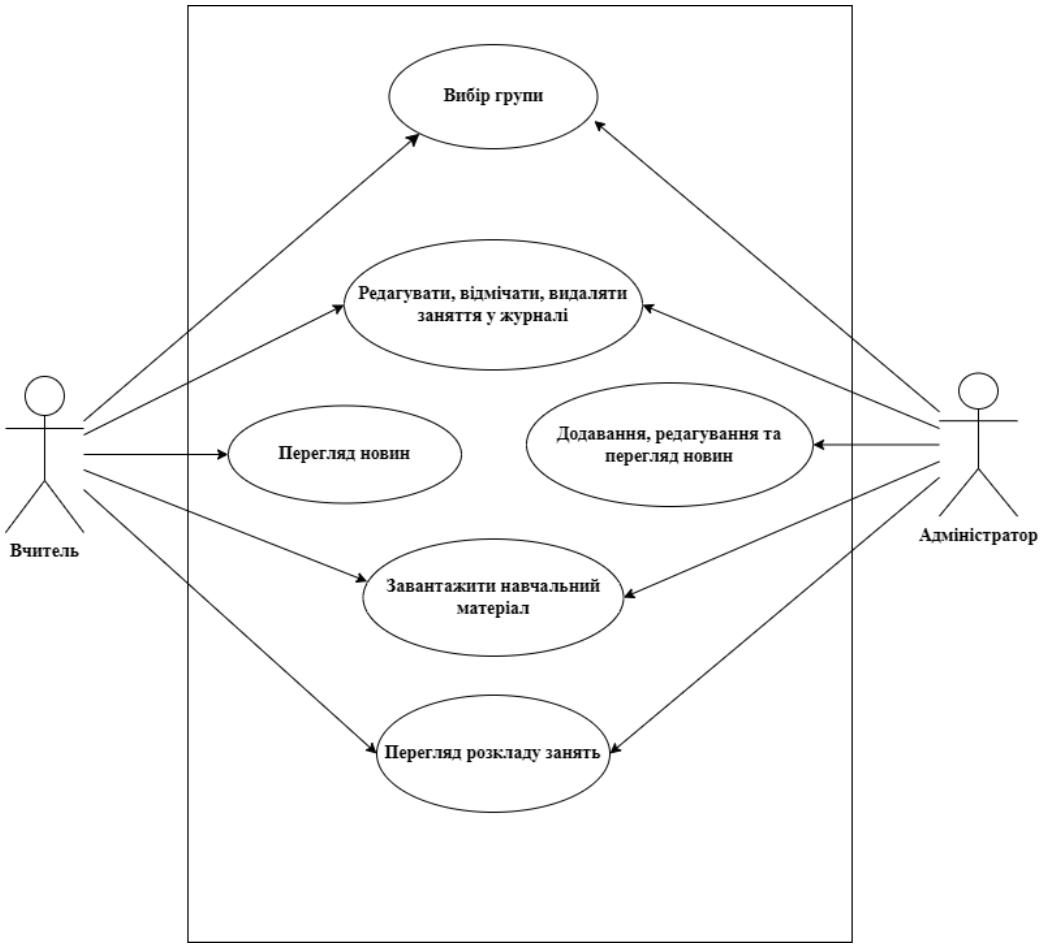

Рисунок 2.1 – Діаграма варіантів використання системи LMS

Діаграма складається із прецедентів акторів і зв’язків. В запропонованій системі, акторами є викладач і адміністратор. Прецедентами описані функції кожного актора (майбутні функції додатку). Система LMS має наступні функції:

Функції адміністратора:

- Завантажити матеріали
- Відмітити заняття
- Редагувати дані про минуле заняття
- Видалити минуле заняття
- Перегляд розкладу

Функції вчителя:

- Реєстрація на платформі
- Відмітити заняття
- Редагувати дані про минуле заняття
- Видалити минуле заняття

2.2 Діаграма класів

Діаграма класів є однією з основних діаграм в мові моделювання UML (Unified Modeling Language) і використовується для візуального представлення структури системи з точки зору класів, їх атрибутів, методів та відносин між ними.

Основне призначення діаграми класів включає наступні аспекти:

- Моделювання структури системи: Діаграма класів допомагає розробникам моделювати та візуалізувати основні компоненти системи та їх взаємозв'язки, що є основою для подальшої реалізації.
- Дизайн об'єктно-орієнтованих систем*:* Вона використовується для розробки об'єктно-орієнтованого дизайну, описуючи класи, їх атрибути, методи та відносини між ними (наслідування, асоціації, агрегація, композиція).

- Комунікація між учасниками проекту: Діаграма класів служить засобом комунікації між різними учасниками проекту, включаючи розробників, аналітиків, архітекторів і замовників, забезпечуючи єдине розуміння структури системи.
- Документування архітектури системи: Вона використовується для документування архітектури та дизайну системи, що полегшує підтримку і розвиток програмного забезпечення в майбутньому.
- Планування реалізації*:* Діаграма класів допомагає в плануванні процесу реалізації, розбиваючи систему на окремі класи і визначаючи їх відповідальності та взаємодії.
- Аналіз вимог і проектування: Вона допомагає в аналізі вимог до системи, дозволяючи проектувальникам і аналітикам визначити необхідні класи і взаємозв'язки між ними на основі функціональних вимог.
- Підтримка рефакторингу: Діаграма класів корисна під час рефакторингу коду, оскільки вона дозволяє зрозуміти поточну структуру системи та виявити можливі покращення або оптимізації.

Діаграма класів є критично важливим інструментом у процесі розробки програмного забезпечення, забезпечуючи чітке і структуроване уявлення про компоненти системи та їх взаємозв'язки.

На рис 2.2 зображено діаграму класів до запропонованої роботи.

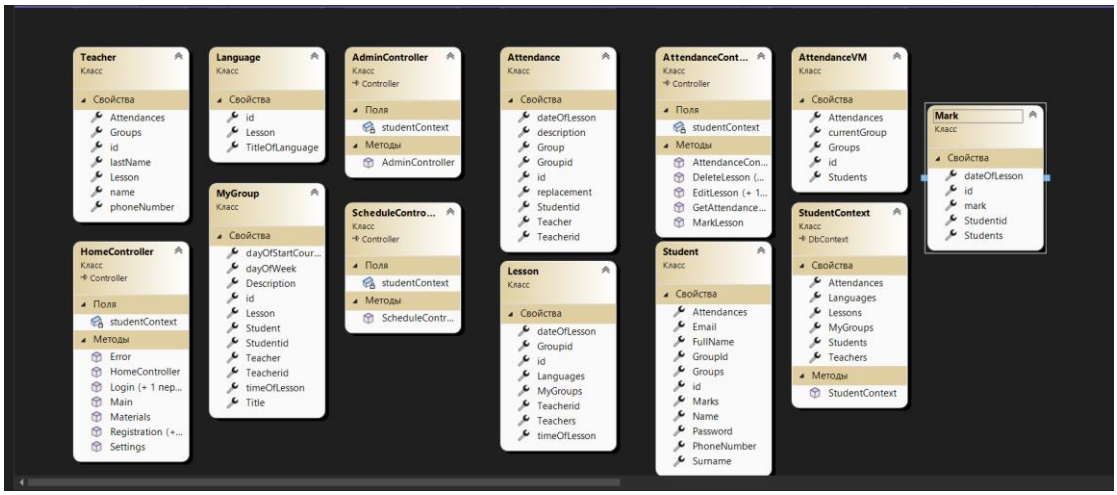

Рисунок 2.2 – Діаграма класів LMS

Ця діаграма класів показує структуру LMS-системи, де кожен клас відповідає певній таблиці бази даних або функціональності системи. Взаємозв’язки між класами відображають зовнішні ключі в таблицях бази даних, що забезпечує цілісність даних і полегшує навігацію між зв’язаними даними.

2.3. Структури даних

Клас HomeController представляє контролер, що відповідає за взаємодію користувачів з LMS-системою через веб-інтерфейс, структуру даних зображено у табл. 2.1

Таблиця. 2.1 – Структура класу HomeController

Назва поля або методу	Тип даних	Призначення
1	2	3
studentContext	StudentContext	Взаємодія з базою даних
appEnviroment	IWebHostEnviroment	Завантаження навчальних матеріалів
AboutUs()	IActionResult	Перенаправлення на сторінку «Про нас»
Materials()	IActionResult	Перенаправлення на сторінку «Матеріали»
AddFile()	IActionResult	Завантаження та зберігання файлів – навчальних матеріалів
Login()	IActionResult	Перевіряє введену інформацію про користувача та чи є у БД користувач з такими даними.

Продовження табл. 1.1

1	2	3
News()	IActionResult	Перенаправлення користувача на сторінку новин
Registration	IActionResult	Зберігає введену інформацію про користувача який реєструється на платформі, та перенаправляє його на сторінку «Новини»

Клас studentContext представляє контекст бази даних для фреймворку EF Core, який відповідає за управління з'єднанням за базою даних і відстеження змін у моделях.

Структуру класу studentContext наведено у табл. 2.2.

Таблиця 2.2 – Структура класу studentContext

Назва поля або методу	Тип даних	Призначення
1	2	3
Students	DbSet<Student>	Створення списку студентів, з таблиці БД
Teachers	DbSet<Teacher>	Створення списку учителів, з таблиці БД
Languages	DbSet< Language >	Створення списку курсів, з таблиці БД
MyGroups	DbSet< MyGroup >	Створення списку груп, з таблиці БД
Lessons	DbSet< Lesson >	Створення списку уроків, з таблиці БД

Продовження табл. 2.2

1	2	3
Attendances	DbSet<Attendance>	Створення списку занять, з таблиці БД
Marks	DbSet<Mark>	Створення списку оцінок, з таблиці БД
Database.EnsureCreated()	IActionResult	Метод для забезпечення створення бази даних, якщо вона ще не існує

Клас AttendanceController представляє контролер, що відповідає за реалізацію функцій електронного журналу, а саме: відмічання, редагування та видалення занять. Щоб описати модель контролера було розроблено модель на основі класу AttendanceController.

Структуру класу AttendanceController наведено у табл. 2.3

Таблиця 2.3 – Структура класу AttendanceController

Назва поля або методу	Тип даних	Призначення
studentContext	StudentContext	Взаємодія з базою даних
GetAttendance	IactionResult	Вилучення з таблиці БД даних про заняття
MarkLesson	IactionResult	Реалізовує функцію відмічання заняття
EditLesson	IactionResult	Реалізовує функцію редагування заняття
DeleteLesson	IActionResult	Реалізовує функцію видалення заняття

Щоб описати модель даних студента було розроблено модель на основі класу Student.

Структура класу Student наведено у табл. 2.4

Таблиця 2.4 – Структура класу Student

Назва поля або методу	Тип даних	Призначення
id	int	Унікальний ідентифікатор студента
Name	String	Зберігає ім’я
Surname	String	Зберігає прізвище
Password	String	Зберігає пароль
PhoneNumber	String	Зберігає номер телефону
FullName	String	Поєднує ім’я та прізвище у один текстовий літерал
GroupId	Int	Ідентифікатор групи студента
Attendances	List<Attendance>	Створення списку занять, з таблиці БД
Groups	List<MyGroup>	Створення списку груп, з таблиці БД

Щоб описати модель даних групи було розроблено модель на основі класу MyGroup.

Структура класу MyGroup наведено у табл. 2.5

Таблиця 2.5 – Структура класу MyGroup

Назва поля або методу	Тип даних	Призначення
1	2	3
id	int	Унікальний ідентифікатор групи
Title	String	Назва групи

Продовження табл. 2.5

1	2	3
timOfLesson	DateTime	Час проведення заняття
dayOfWeek	int	День неділі
dayOfStartCourse	DateTime	Дата початку курса
Studentid	int	Ідентифікатор студента
Teacherid	int	Ідентифікатор вчителя
Languageid	int	Ідентифікатор уроку
Student	Student	Об’єкт моделі студента
Teacher	Teacher	Об’єкт моделі вчителя
Lessons	List<Lesson>	Створення списку уроків, з таблиці БД
Languages	List<Language>	Створення списку курсів, з таблиці БД

Щоб описати модель даних вчителя було розроблено модель на основі класу Teacher.

Структура класу Teacher наведено у табл. 2.6

Таблиця 2.6 – Структура класу Teacher

Назва поля або методу	Тип даних	Призначення
1	2	3
id	int	Унікальний ідентифікатор вчителя
name	String	Зберігає ім’я вчителя
lastName	String	Зберігає прізвище вчителя
FullName	String	Створює єдиний строковий літерал ім’я+прізвище
phoneNumber	String	Зберігає номер телефону вчителя

Продовження табл. 2.6

1	2	3
Attendances	List<Attendance>	Створення списку занять, з таблиці БД
Lessons	List<Lesson>	Створення списку уроків, з таблиці БД
MyGroups	List<MyGroup>	Створення списку груп, з таблиці БД

Щоб описати модель даних оцінки було розроблено модель на основі класу Mark.

Структура класу Mark наведено у табл. 2.7

Таблиця 2.7 – Структура класу Mark

Назва поля або методу	Тип даних	Призначення
id	int	Зберігає ім’я студента
Studentid	int	Зберігає прізвище студента
mark	int	Оцінка учня
Attendance_id	int	Унікальний ідентифікатор занять
Student	Student	Об’єкт моделі студента
Attendance	Attendance	Об’єкт моделі заняття

Щоб описати модель даних уроку було розроблено модель на основі класу Lesson.

Структура класу Lesson наведено у табл. 2.8

Таблиця 2.8 – Структура класу Lesson

Назва поля або методу	Тип даних	Призначення
id	int	Унікальний ідентифікатор
Groupid	int	Унікальний ідентифікатор групи
Teacherid	int	Унікальний ідентифікатор вчителя
timeOfLesson	String	Час проведення уроку
dateOflesson	DateTime	Дата проведення уроку
MyGroup	MyGroups	Об’єкт моделі групи

Щоб описати модель даних курсу було розроблено модель на основі класу Language.

Структура класу Language наведено у табл. 2.9

Таблиця 2.9 – Структура класу Language

Назва поля або методу	Тип даних	Призначення
id	id	Унікальний ідентифікатор
Title	String	Назва курсу
MyGroup	MyGroup	Об’єкт моделі групи

Щоб описати модель даних матеріалів було розроблено модель на основі класу FileModel.

Структура класу FileModel наведено у табл. 2.10

Таблиця 2.10 – Структура класу FileModel

Назва поля	Тип даних	Призначення
id	int	Унікальний ідентифікатор
Name	string	Ім’я файлу
Path	String	Шлях де зберігається файл

Щоб описати модель даних заняття було розроблено модель на основі класу Attendance.

Структура класу Attendance наведено у табл. 2.11

Таблиця 2.11 – Структура класу Attendance

Назва поля	Тип даних	Призначення
id	int	Унікальний ідентифікатор
Studentid	int	Унікальний ідентифікатор студента
Teacherid	int	Унікальний ідентифікатор вчителя
MyGroupid	int	Унікальний ідентифікатор групи
dateOfLesson	DateTime	Дата проведення заняття
description	string	Коментар до заняття
replacement	int	Заміна вчителя для заняття
MyGroup	MyGroup	Об’єкт моделі групи
Teacher	Teacher	Об’єкт моделі вчителя
Student	Student	Об’єкт моделі студента
Marks	List<Marks>	Створення списку оцінок, з таблиці БД

Щоб описати віртуальної моделі даних заняття було розроблено модель на основі класу AttendanceVM.

Структура класу AttendanceVM наведено у табл. 2.12

Таблиця 2.12 – Структура класу AttendanceVM

Назва поля	Тип даних	Призначення
1	2	3
id	int	Унікальний ідентифікатор

Продовження табл. 2.12

1	2	3
MyGroup	currentGroup	Позначає поточну групу
MyGroups	List<MyGroup>	Створення списку груп, з таблиці БД
Students	List<Student>	Створення списку студентів, з таблиці БД
Attendances	List<Attandance>	Створення списку занять, з таблиці БД

2.4. Структура бази даних

Було спроектовано базу даних на основі Microsoft SQL Server, на рис. 2.4.1 Наведено структуру бази даних запропонованого додатку. Розробка бази велась у середовищі Microsoft SQL Server Management Studio, там же було розроблено структурну схему запропонованої бази даних, на рис.

На рисунку 2.3 зображено склад таблиць та зв’язки між ними:

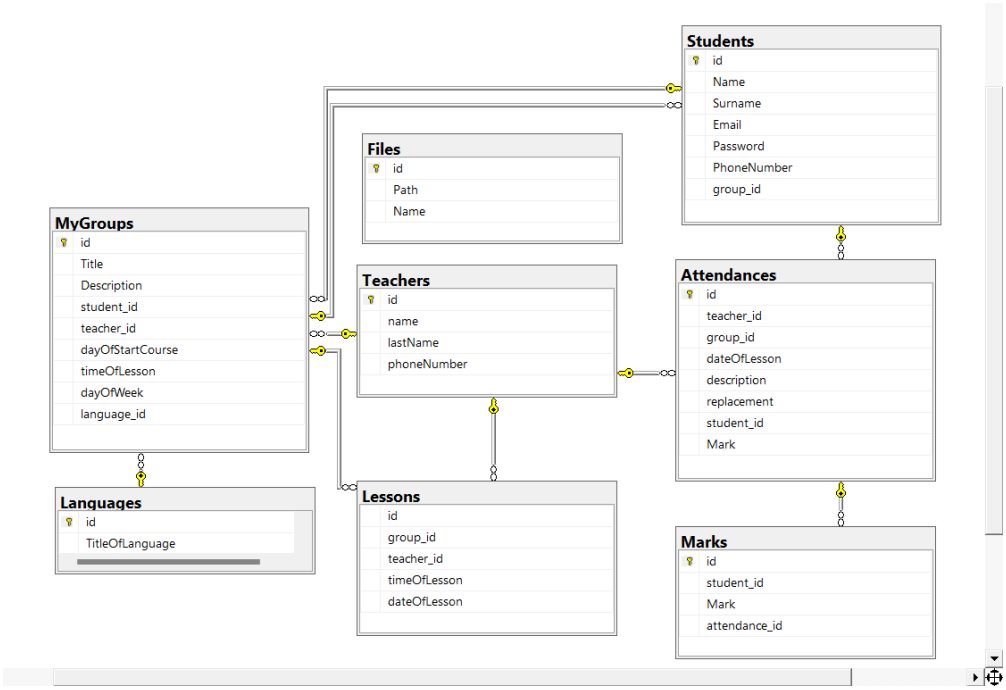

Рисунок 2.3 – структура бази даних Microsoft SQL Server Management

База даних складається з таких таблиць:

Lessons –зберігає інформацію про урок

Таблиця має такі поля:

- id: (primary key) – унікальний ідентифікатор уроку
- group_id – зовнішній ключ для зв’язку з таблицею MyGroups
- teacher_id – зовнішній ключ для зв’язку з таблицею Teachers
- timeOfLesson – час проводення заняття
- dateOfLesson – дата проведення заняття

MyGroups – зберагіння інформації про групу

Таблиця має такі поля:

- id (primary key) – унікальний ідентифікатор групи
- Title – назва групи
- Student_id – зовнішній ключ для зв’язку з таблицею Students
- Teacher_id – зовнішній ключ для зв’язку з таблицею Teachers
- dayOfStartCourse – дата початку курсу
- TimeOfLesson – час початку курсу
- dayOfWeek – день неділі
- language_id – зовнішній ключ для зв’язку з таблицею Languages

Students – зберігання інформації про студента

Таблиця має такі поля:

- id (primary key) – унікальний ідентифікатор студента
- Name – ім’я
- Surname – прізвище
- Email – електронна адреса
- Password – пароль
- PhoneNumber – номер телефона
- group_id – зовнішній ключ для зв’язку з таблицею MyGroups

Attendances – зберігання інформації про відмічені заняття

Таблиця має такі поля:

- id (primary key) – унікальний ідентифікатор відміченого заняття
- teacher_id – зовнішній ключ для зв’язку з таблицею Teachers
- group_id – зовнішній ключ для зв’язку з таблицею MyGroups
- dateOfLesson – дата проведення заняття
- description – коментар до заняття
- replacement – заміна вчителя
- student_id – зовнішній ключ для зв’язку з таблицею Students

Teachers – зберігання інформації про вчителя

Таблиця має такі поля

- id (primary key) – унікальний ідентифікатор відміченого заняття
- Name – ім’я
- LastName – прізвище
- phoneNumber – номер телефону

Languages – зберігання інформації про курс

Таблиця має такі поля

- id (primary key) – унікальний ідентифікатор курсу
- Title – назва курсу

Files – зберігання інформації про завантажені матеріали

Таблиця має такі поля

- id (primary key) – унікальний ідентифікатор
- Path – шлях до каталогу де зберігається файл
- Name – назва файлу

Здебільшого у базі даних між таблицями реалізовані зв’язки один до багатьох.

2.5 Характеристика технологій застосованих при розробці

Для розробки даного проекту було застосовано такі технології:

ASP.NET MVC (Model-View-Controller) — це програмний фреймворк для створення веб-додатків на платформі Microsoft .NET. Він забезпечує чітке розділення програми на три основні компоненти: модель, представлення і контролер. Це дозволяє розробникам створювати більш організований і керований код.

На рис. 2.4 зображено структуру фреймворку MVC.

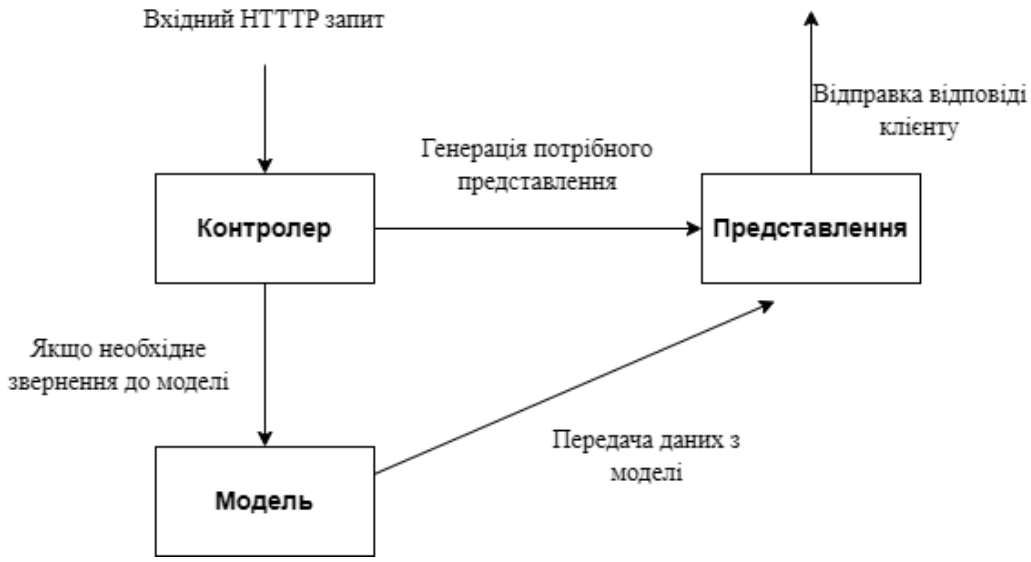

Рисунок 2.4 – Структура MVC.

Розглянемо кожен компонент детальніше:

Модель (Model):

- Відповідає за дані програми і бізнес-логіку.
- Включає класи, що представляють дані додатка, правила для їх обробки, методи для роботи з базою даних тощо.

Представлення (View):

- Відповідає за відображення даних користувачам.
- Включає HTML-сторінки, CSS, JavaScript і Razor-розмітку (синтаксис для вбудовування C# коду в HTML).

Контролер (Controller):

- Відповідає за обробку запитів користувачів.

- Приймає запити, взаємодіє з моделлю для отримання необхідних даних, і визначає, яке представлення потрібно повернути користувачеві.

Основні переваги ASP.NET MVC:

- Розділення обов'язків: Завдяки чіткому розділенню на модель, представлення і контролер, код стає більш організованим і легко підтримуваним.
- Модульність: Можливість легко змінювати або тестувати окремі компоненти додатка без впливу на інші.
- Тестованість: Зручність написання юніт-тестів для контролерів і моделей.
- Потужність та гнучкість: Можливість використовувати повний спектр можливостей платформи .NET та інтеграція з різними бібліотеками і інструментами.

ASP.NET MVC є частиною більш широкої платформи ASP.NET, яка включає також Web API для створення HTTP-сервісів та ASP.NET Core — більш нову, крос-платформенну версію фреймворка для створення сучасних веб-додатків.[7-9]

Entity Framework Core (EF Core) — це об'єктно-реляційний маппер (ORM) для .NET, який дозволяє розробникам працювати з базами даних за допомогою .NET об'єктів. Це полегшує доступ до даних шляхом автоматичного перетворення між об'єктами у вашій програмі та записами в базі даних.

На рис. 2.5 зображено структуру і принцип взаємодії EF Core з базою даних та ASP .NET Core.

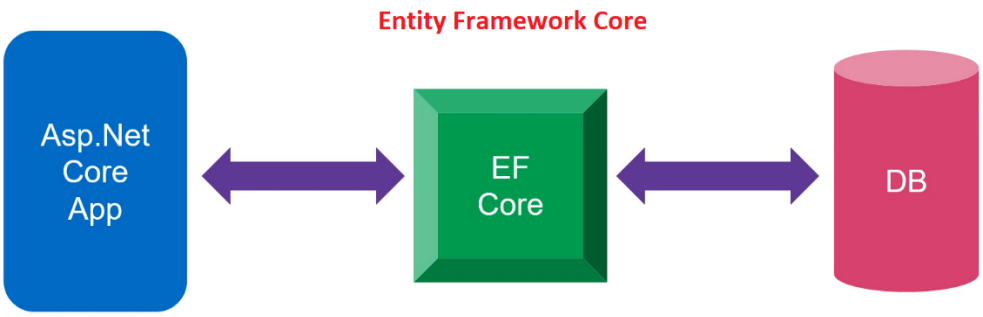

Рисунок 2.5 – Структура, принцип роботи EF Core

Основні риси EF Core:

- Підтримка різних баз даних: EF Core підтримує роботу з багатьма різними базами даних, такими як SQL Server, SQLite, PostgreSQL, MySQL та інші.
- Моделювання: Створення моделі даних, використовуючи класи C#. Модель визначає структуру даних, їх властивості та зв'язки між ними.
- Міграції: Міграції дозволяють змінювати структуру бази даних поступово, без втрати даних. Завдяки цьому дуже зручно виконувати управління змінами у структурі бази даних під час розробки додатка.
- LINQ (Language Integrated Query): LINQ дозволяє писати запити до бази даних у вигляді синтаксису C#, що значно спрощує роботу з даними і робить код більш зрозумілим.
- Відстеження змін: EF Core автоматично відстежує зміни, внесені в об'єкти, і дозволяє зберігати ці зміни в базі даних.
- Засів даних: Можливість автоматично заповнювати базу даних початковими даними під час створення або оновлення схеми.
- Конфігурація та адаптивність: Налаштування поведінки EF Core за допомогою Fluent API або атрибутів (Data Annotations) для більш точного контролю над мапінгом об'єктів і схемою бази даних.
- Основні переваги EF Core: Зниження кількості коду: Замість написання SQL-запитів вручну, ви працюєте з об'єктами, що значно спрощує і прискорює розробку.
- Тестованість: Легше написати юніт-тести для коду, що працює з даними. Підтримка різних платформ: EF Core підтримує крос-платформенність і може використовуватися в додатках, що працюють на Windows, Linux, macOS та інших платформах.

EF Core є частиною .NET Core і є еволюцією класичного Entity Framework, але з новою архітектурою, яка забезпечує кращу продуктивність і більшу гнучкість.[10-11]

Асинхронне програмування — це парадигма програмування, яка дозволяє виконувати операції у фоновому режимі, не блокуючи основний потік виконання програми. Це особливо корисно для виконання довготривалих або ресурсомістких операцій, таких як робота з мережею, доступ до файлів або взаємодія з базами даних, оскільки дозволяє програмі залишатися чуйною і швидко реагувати на дії користувача.

На рис. 2.6 зображено принцип роботи потоку програми під час виконання асинхронних операцій.

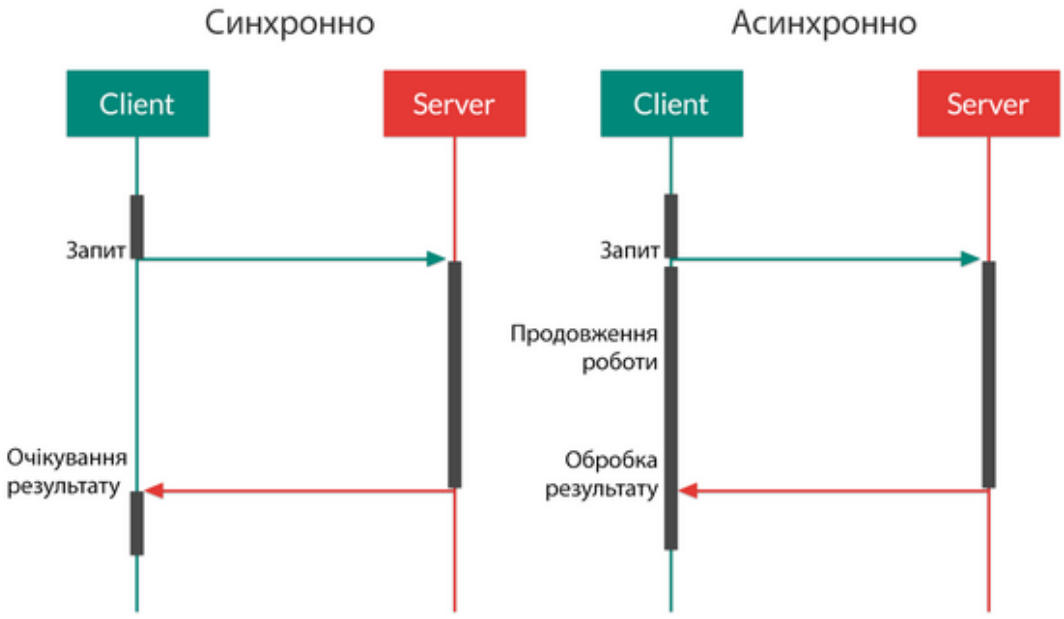

Рисунок 2.6 – Принцип роботи програми при використанні технології асинхронного програмування

Основні концепції асинхронного програмування:

- Асинхронні методи: Асинхронні методи виконуються у фоновому режимі і не блокують основний потік виконання. Вони позначаються ключовими словами async і await в мовах програмування, таких як C#.
- Task (Завдання): В C# асинхронні методи повертають об'єкти типу Task або Task<T>, які представляють операції, що виконуються асинхронно. Task може бути завершеним, невиконаним або виконуваним.

- Await (Очікування): Ключове слово await використовується для паузи виконання методу до завершення асинхронної операції. Це дозволяє повернути управління потоком виконання основного методу, дозволяючи іншим операціям продовжуватися.
- Асинхронні делегати та події: Асинхронне програмування також включає використання делегатів і подій, які можуть бути викликані асинхронно, що дозволяє реактивні дії на події.
- Основні переваги асинхронного програмування: Підвищена чуйність: Програма залишається чуйною і може виконувати інші завдання, поки довготривала операція завершується.
- Підвищена продуктивність: Дозволяє ефективніше використовувати ресурси, особливо в багатозадачних і серверних середовищах.
- Спрощення багатопоточності: Асинхронне програмування спрощує створення багатопоточних програм без необхідності явно керувати потоками.

HTML (HyperText Markup Language) і CSS (Cascading Style Sheets) — це основні технології для створення веб-сторінок.

HTML — це мова розмітки, яка використовується для створення структури веб-сторінок. Вона складається з елементів, які браузер використовує для відображення контенту.

Основні поняття HTML:

- Теги: HTML складається з тегів, які визначають різні елементи на сторінці.
- Атрибути: Теги можуть мати атрибути, які надають додаткову інформацію про елемент. Наприклад, атрибут href для посилань або src для зображень.
- Структура документа: HTML-документ має стандартну структуру, яка включає <html>, <head>, і <body>.

CSS — це мова стилізації, яка використовується для визначення зовнішнього вигляду HTML-елементів на веб-сторінці. CSS дозволяє відокремлювати візуальне представлення від структури HTML.

Основні поняття CSS:

- Селектори: CSS використовує селектори для вибору HTML-елементів, до яких буде застосовано стилі. Наприклад, селектор p вибирає всі <p> теги.
- Властивості та значення: Кожен стиль складається з властивості та значення. Властивості визначають, що ми хочемо змінити, а значення вказують, як це повинно виглядати.
- Каскадність: CSS працює за принципом каскадності, що означає, що стилі можуть бути перевизначені більш конкретними селекторами.
- Медіа-запити: CSS підтримує медіа-запити, що дозволяють застосовувати стилі залежно від розміру екрану або інших характеристик пристрою, що робить веб-сторінки адаптивними.[12]

Razor Pages — це функція ASP.NET Core, яка спрощує розробку веб-додатків за допомогою шаблонів. Вона пропонує модель, засновану на сторінках, що дозволяє розробникам працювати з веб-сторінками як з автономними одиницями. Razor Pages спрощує розробку, організацію і підтримку коду порівняно з традиційними підходами, такими як MVC (Model-View-Controller).

Основні особливості Razor Pages:

- Модель сторінки: Кожна сторінка в Razor Pages представлена класом PageModel, який містить логіку для сторінки. Це дозволяє чітко розділяти логіку і представлення.
- Синтаксис Razor: Razor Pages використовує Razor-синтаксис для вбудовування C# коду в HTML. Це спрощує створення динамічних веб-сторінок.
- Спрощена навігація: Завдяки моделі сторінок і використанню чітких URL-шляхів, навігація між сторінками в Razor Pages є інтуїтивно зрозумілою і простою.

- Інтеграція з іншими технологіями ASP.NET Core: Razor Pages добре інтегрується з іншими частинами ASP.NET Core, такими як залежне впорскування (Dependency Injection), налаштування маршрутів і Middleware.

Переваги Razor Pages:

- Простота і зручність: Менше файлів і менше коду для створення повнофункціональних веб-сторінок.
- Розділення обов'язків: Логіка і представлення розділені, що полегшує розробку і підтримку коду.
- Гнучкість і масштабованість: Легко адаптується для створення як малих, так і великих веб-додатків.
- Підвищена продуктивність: Можливість швидше створювати і оновлювати веб-сторінки завдяки простій моделі програмування.

Razor Pages — це потужний інструмент для створення сучасних веб-додатків на платформі ASP.NET Core, який пропонує розробникам інтуїтивно зрозумілий і ефективний підхід до роботи з веб-сторінками.[7]

SQL Server — це система управління базами даних, розроблена компанією Microsoft. Вона дозволяє зберігати, змінювати та керувати інформацією у структурованому форматі. SQL Server забезпечує високу продуктивність і надійність роботи з великими обсягами даних та підтримує різні типи даних і операцій.

Ця система використовується для зберігання та обробки даних в різних застосунках, включаючи корпоративні системи, веб-додатки та аналітичні платформи. SQL Server надає інструменти для створення та управління базами даних, забезпечує безпеку даних та підтримує резервне копіювання і відновлення інформації.

Завдяки своїй функціональності SQL Server широко використовується в бізнесі та організаціях різного масштабу для ефективного управління даними та підтримки прийняття рішень на основі аналізу інформації.

SQL Server Management Studio (SSMS) — це інтегроване середовище, розроблене компанією Microsoft для управління SQL Server. SSMS надає користувачам інструменти для налаштування, адміністрування та розробки баз даних у SQL Server.

Це середовище дозволяє виконувати різні завдання, такі як створення, редагування та видалення баз даних, написання та виконання SQL-запитів, керування безпекою і доступом до даних, а також моніторинг продуктивності системи. SSMS об'єднує графічний інтерфейс з потужними функціями, що дозволяє адміністраторам баз даних і розробникам ефективно працювати з SQL Server.

Цей інструмент є важливою складовою частиною екосистеми SQL Server і широко використовується для підтримки і оптимізації роботи баз даних, забезпечуючи зручний та ефективний спосіб керування даними.[13]

На рис. 2.7 зображено користувацький інтерфейс середовища SSMS.

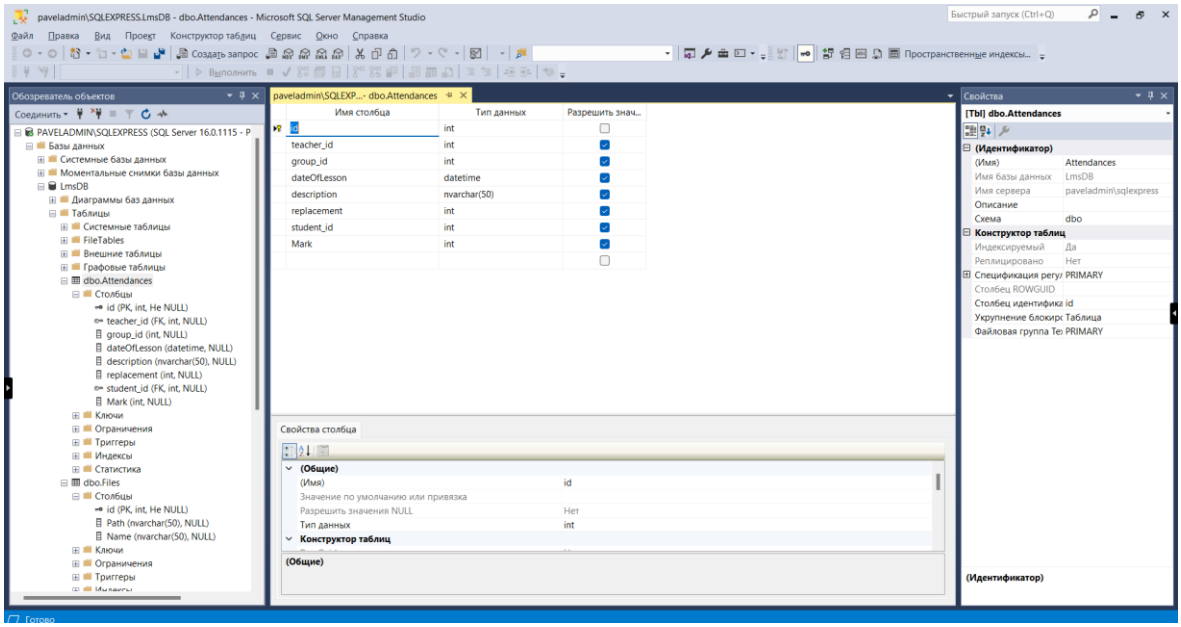

Рисунок 2.7 – Середовище управління БД SSMS

У цьому підрозділі було описано та обгрунтовано вибір технологій які були застосовані при розробці запропонованої роботи.

2.3. Інструкція використання

Цей розділ описує, як виглядають сторінки LMS системи та як ними користуватися. Цей пункт інструкції допоможе користувачам зрозуміти, як виглядають різні сторінки системи та який функціонал вони реалізують. Для кожної сторінки представлено її інтерфейс, функціональні елементи та перелік дій, які необхідно виконати для роботи із системою

На рис. 2.8 зображена мапа сайту. Вона надає структурований огляд основних сторінок і їх функціональних можливостей.

Рисунок 2.8 – Мапа сайту

2.3.1 Сторінка входу в систему

Робота з додатком починається з входу в систему. Вигляд сторінки входу в систему зображено на рисунку 2.9.

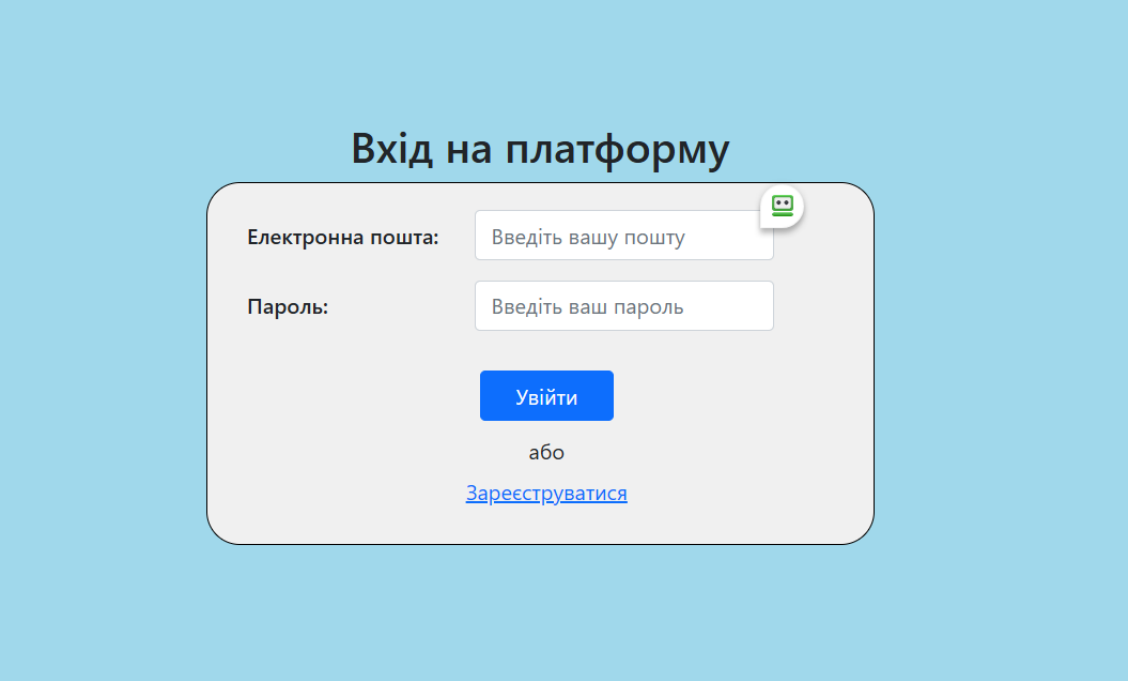

Рисунок 2.9 – Сторінка входу в систему

Для входу необхідно ввести логін і пароль. Додаток передбачає два рівні доступу – адміністратор та викладач. Введені логін і пароль співставляються з тими, з тими, що зберігаються у базі даних. Якщо у базі даних не знаходиться відповідний пароль, то відбувається перенаправлення на сторінку реєстрації, зображено на рис. 2.10.

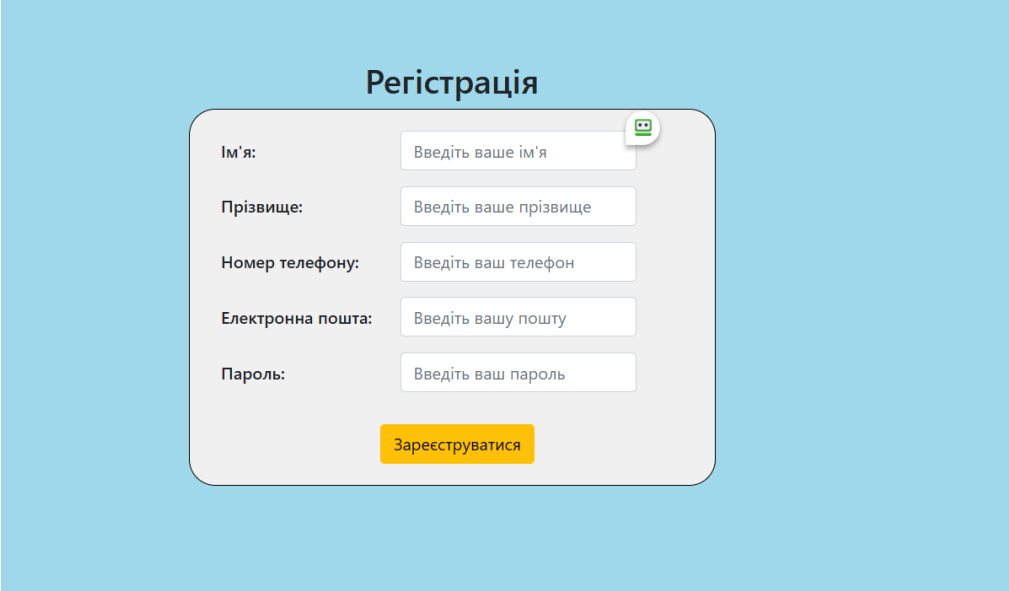

Рисунок 2.10 – Сторінка реєстрації у системі

Для реєстрації необхідно ввести відповідні дані (ім’я, прізвище, номер телефону, електронна адреса, пароль) та натиснуту кнопку «Зареєструватися». Після натискання на кнопку, дані введені у формі реєстрації відправляються до серверу бази даних та зберігаються у таблиці Student, після цього йде перенаправлення на сторінку «Новини».

2.3.2 Сторінка «Новини»

На сторінці новин користувачі платформи зможуть побачити останні, актуальні новини та заходи які проводяться освітнім закладом. На рис. 2.11 зображено вигляд сторінки «Новини».

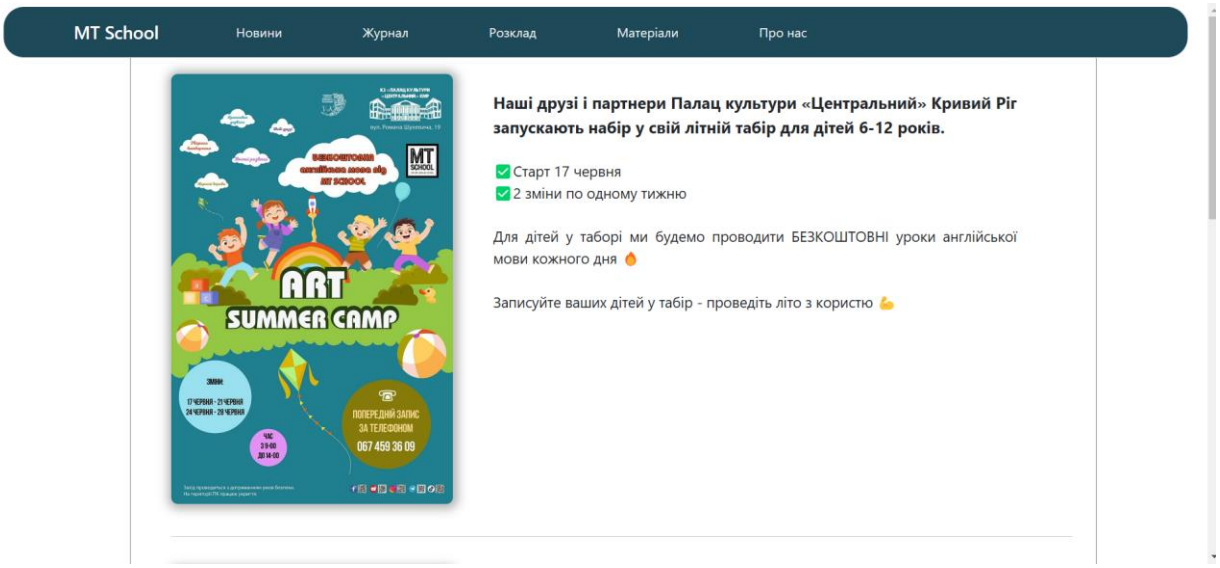

Рисунок 2.11 – Сторінка новин

За допомогою меню навігації вчитель та адміністратор можуть переходити на інші функціональні сторінки.

2.3.3 Сторінка «Журнал»

Після переходу на сторінку за допомогою навігаційному меню, для того щоб перейти на журнал занять певного учня викладач повинен вибрати в списку відповідну групу. Вигляд журналу та вибору групи зображено на рис. 2.12.

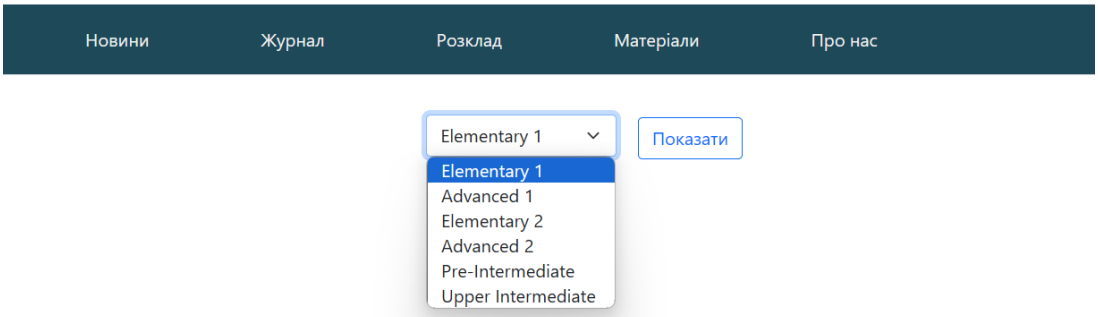

Рисунок 2.12 – Вигляд списку вибору групи

Після вибору групи та натискання на кнопку «Показати» на сторінці журналу виводиться прізвище учня, кількість проведених всього уроків, а також записи про проведені (відмічені) уроки. Під таблицю журналу є кнопка для додавання (відмічання) нових записів про проведені уроки. На рис. 2.13 зображено вигляд таблиці занять для певної групи.

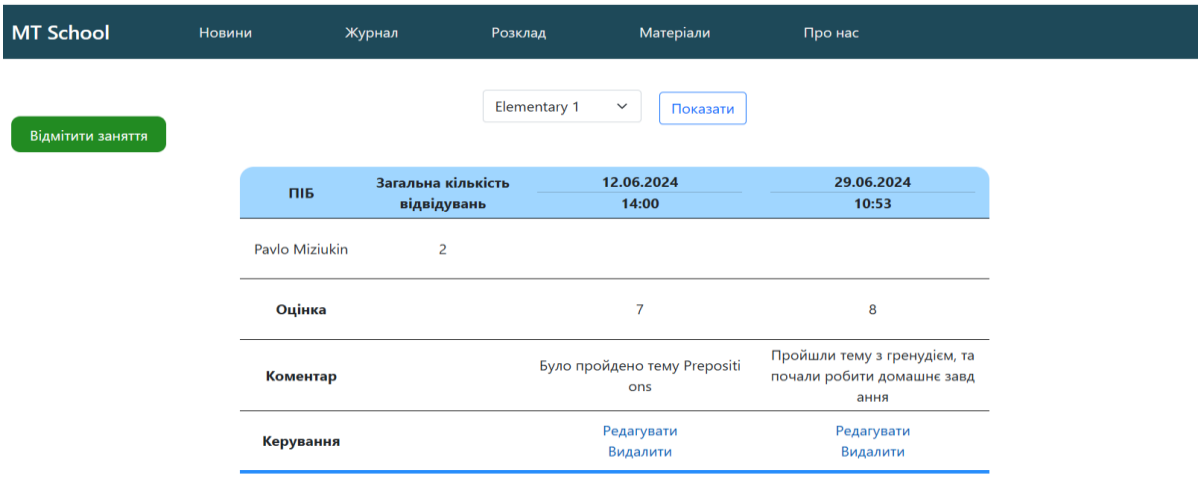

Рисунок 2.13 – Вигляд таблиці занять

На передостанньому рядку журналу відображається коментар до заняття. Коментар додається при внесенні інформації про пройдене заняття, відмити поточне заняття можна, за допомогою кнопки «Відмітити заняття». Після натискання на кнопку користувач потрапляє на сторінку відмічання, де він має внести дані про заняття для вибраної групи. На рис. 2.14 зображено вигляд відмічання заняття для вибраної групи

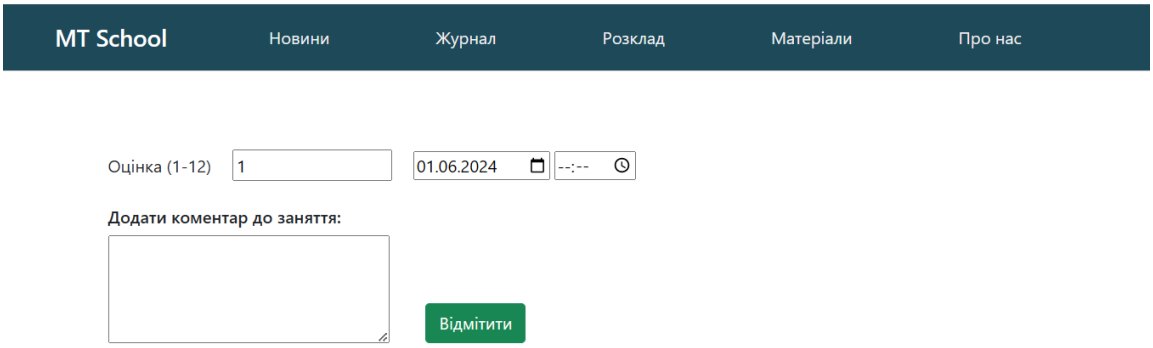

Рисунок 2.14 – Вигляд відмічання заняття

В колонці, де відмічено заняття є можливість відредагувати інформацію про проведене заняття (змінити оцінку, змінити коментар по заняттю) або видалити відмічене заняття. На рис. 2.15 зображено вигляд відмічання заняття для вибраної групи

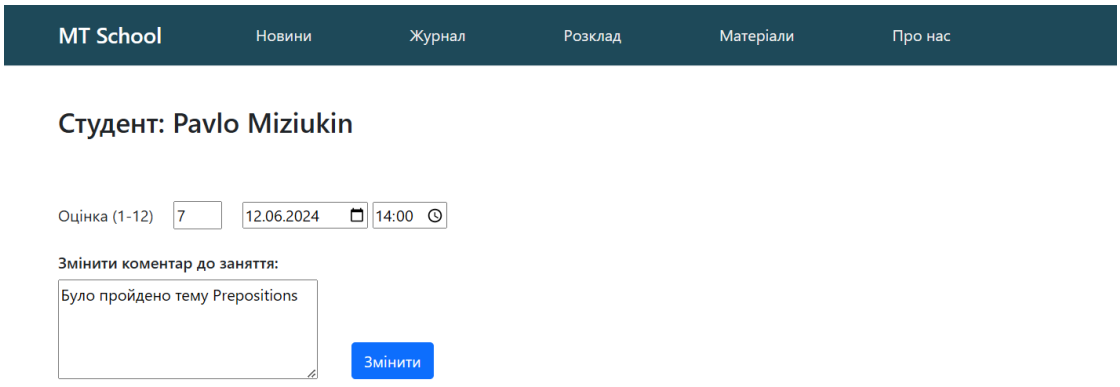

Рисунок 2.15 – Вигляд редагування заняття

Запис про заняття неможливо видалити одразу зі сторінки журналу. Необхідно підтвердити намір про видалення на окремій сторінці. Це зроблено для того щоб унеможливити випадкове видалення записів про заняття. На рис. 2.16 вигляд сторінки видалення запису заняття з таблиці бази даних.

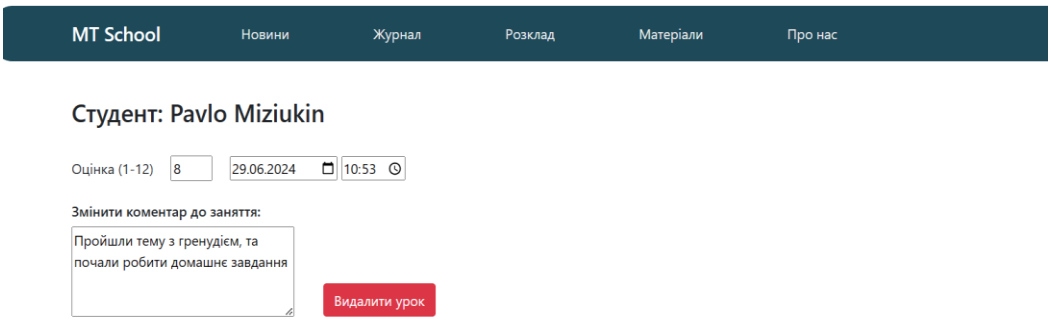

Рисунок 2.16 – Сторінка видалення заняття

Після натискання на кнопку «Видалення заняття » у базу даних надсилається запит на видалення запису про заняття для таблиці журналу та відбувається перехід до таблиці журналу.

2.3.3 Сторінка «Розклад»

Для того, щоб користувач мав змогу спостерігати за графіком уроків групи, було розроблено сторінку розкладу. Вчитель, який переходить на сторінку розкладу, буде бачити лише ті групи, які належать йому. На рис. 2.17 зображено вигляд сторінки розкладу.

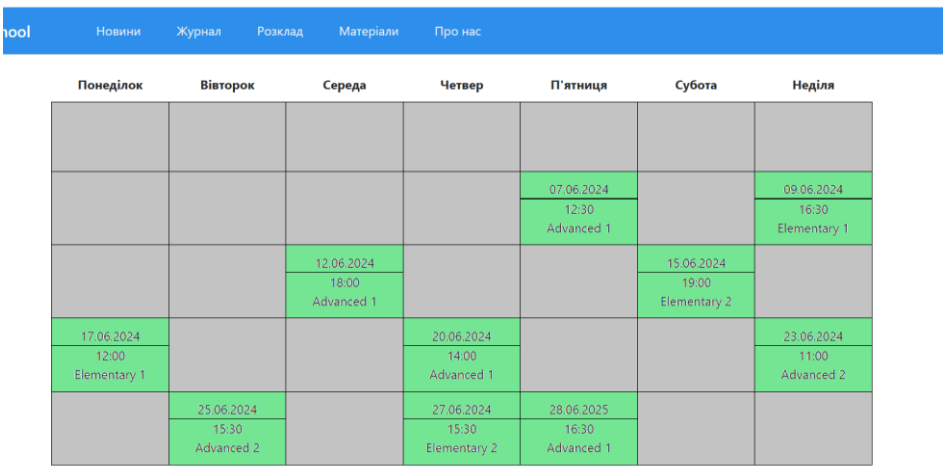

Рисунок 2.17 – Сторінка розкладу уроків

2.3.4 Сторінка «Матеріали»

Для того, щоб викладач міг завантажити та вивантажити навчальні матеріали було розроблено функціональну сторінку «Матеріали». Для того щоб завантажити навчальні матеріли на свій комп’ютер користувач має натиснути на назву файла, яка є посиланням на завантаження.Вигляд сторінки для Перегляду матеріалів зображено на рис. 2.18.

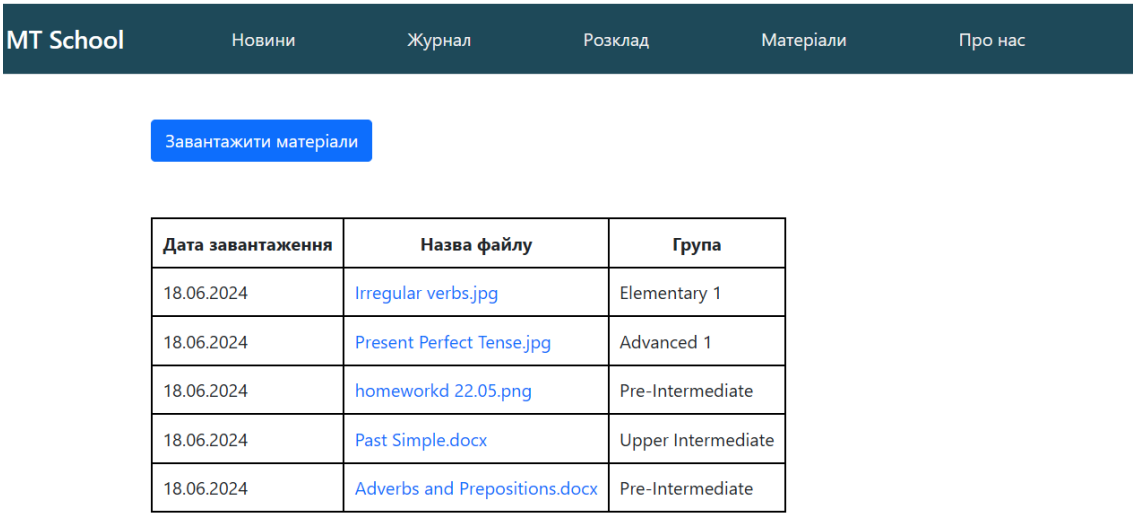

Рисунок 2.18 – Вигляд сторінки «Матеріали»

Для того, додати нові матеріали до таблиці файлів, користувач має натиснути на кнопку «Завантажити матеріали», після чого відбувається перехід на сторінку завантаження файлів до таблиці. Вигляд сторінки завантаження файлів на сервер зображено на рис. 2.19

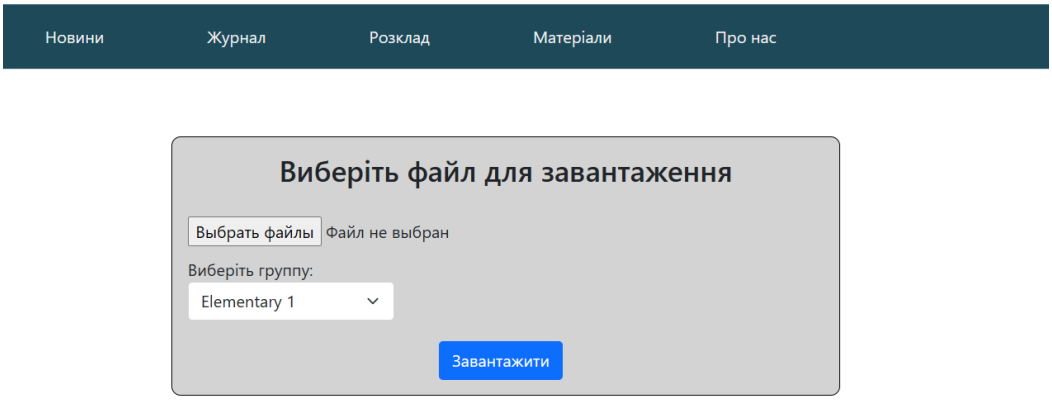

Рисунок 2.19 – Вигляд сторінки завантаження файлів

Виконано опис усіх функціональних сторінок системи, що дозволяє користувачеві ознайомитися з нею та побачити основний функціонал.

Висновки до розділу

В даному розділі було розроблено структуру запропонованої системи та наведено схеми і діаграми, які показають структуру запропонованої системи, а також інструкцію користувача.

Для моделювання функціоналу майбутньої системи виконано розробку діаграм варіантів використання та діаграму класів. Розроблено структуру бази даних на основі Microsoft SQL Server. Обгрунтовано вибір технологій необхідних для розробки запропонованої системи, а саме: ASP .NET Core MVC, асинхронне програмування, Entity Framework Core, Razor Pages, HTML/CSS.Для розробки програмного забезпечення були використанні середовища Microsoft Visual Studio 2022 – для написання програмного коду, клієнтської і серверної частини, а також Microsoft SQL Server Management Studio для розробки бази даних. Було розроблено структуру бази даних, визначено склад і структуру таблиць, що описують основні сутності додатку. Наведено інструкцію користувача, яка складається із опису користувацьких екранів.

ВИСНОВОКИ

Розробка LMS-системи для освітнього закладу є важливим етапом у вдосконаленні управління взаємодією зі студентами та викладачами. На основі аналізу потреб та вимог навчального закладу було визначено, що така система повинна включати в себе функції електронного журналу, розклад, можливість завантаження навчальних матеріалів ведення курсів та програм, аналітику та звітність, а також можливості автоматизації деяких рутинних процесів.

Розробка та впровадження LMS-системи для навчального закладу є важливим кроком у напрямку модернізації освітніх послуг. У майбутньому планується розширення функціоналу системи за рахунок інтеграції з іншими сервісами, використання штучного інтелекту для персоналізації навчального процесу та впровадження нових інструментів для аналізу даних.

У результаті розробки LMS-системи для навчального закладу очікується покращення ефективності управління студентською базою, підвищення задоволеності користувачів та оптимізація процесів взаємодії зі студентами. Така система допоможе навчальному закладу забезпечити більш якісне та персоналізоване обслуговування своїх клієнтів, а також відповідати вимогам сучасного освітнього середовища.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Study and Review of Learning Management System Software / Gaurav Srivastav, Mahima Sharma – Innovations in Computer Science and Engineering, 2003.– pp. 373-383.

2. Learning Manangment System / Paul Kirvan, Kate Brush. URL: https://www.techtarget.com/searchcio/definition/learning-management-system (дата звернення 12.05.2024)

3. Moodle LMS. Веб-сайт. URL: https://moodle.com/ (дата звернення 13.05.2024)

4. Google Classroom LMS. Веб-сайт. URL: https://classroom.google.com/ (дата звернення 13.05.2024)

5. Blackboard LMS. Веб-сайт. URL: https://www.blackboard.com/ (дата звернення 13.05.2024)

6. АСУ НЗ. Веб-сайт. URL: https://mkr.org.ua/ (дата звернення 14.05.2024)

7. ASP.NET Core in Action, Second Edition. E. Lock. 2021. - 832 pp.

8. MVC Design Pattern in ASP.NET Core: веб-сайт. URL: https://www.tektutorialshub.com/asp-net-core/asp-net-core-mvc-design-pattern/ (дата звернення 20.05.2024).

9. ASP.NET Core 2.0 MVC & Razor Pages for Beginners: How to Build a Website. D. Fugerburg. 2017. 1088 pp.

10. Entity Framework documentation. URL: https://docs.microsoft.com/en-us/ef/ (дата звернення 20.05.2024).

11. A. Freman. Pro Entity Framework Core 2 for ASP.NET Core MVC. 2018, pp. 650.

12. Andy, Harris HTML, XHTML and CSS All–In–One For Dummies® / Andy Harris. 2014. - 173 pp.

13. Короткевич. Д. SQL Server. Налаштування та оптимізація для фахівців. Київ: Лідер-Букс, 2017. 24 - 30 с.

14. Bootstrap documentation URL: https://getbootstrap.com/docs/5.3/getting-started/introduction/ (дата звернення 14.05.2024)

15. Razor Pages in ASP .NET URL: https://learn.microsoft.com/ru-ru/aspnet/core/razor-pages/?view=aspnetcore-8.0&tabs=visual-studio (дата звернення 13.03.2024)

16. Моркун Н. В., Маринич І. А. Методичні вказівки до виконання кваліфікаційної роботи бакалавру для студентів спеціальності 151 “Автоматизація та комп’ютерно-інтегровані технології”. Кривий Ріг : Видавничий центр КНУ, 2019. 50 с.

17. ДСТУ 3008:2015. Звіти у сфері науки і техніки. Структура і правила оформлення. Київ, ДП «УкрННЦ», 2015. 26с. (Інформація та документація).

18. ДСТУ 8302:2015. Бібліографічне посилання. Загальні вимоги та правила складання Київ, ДП «УкрННЦ», 2016. 16 с. (Інформація та документація).

19. ДСТУ 3582:2013. Бібліографічний опис. Скорочення слів і словосполучень в українській мові. Загальні вимоги та правила. Київ, ДП «УкрННЦ», 2013. 23 с. (Інформація та документація)

20. ДСТУ 3651.0-97 Метрологія. Одиниці фізичних величин. Основні одиниці фізичних величин Міжнародної системи одиниць. Основні положення, назви та позначення Київ, Держстандарт України, 1998. 27 с. (Інформація та документація).

ДОДАТОК А

Лістинг коду класів контролерів

На рисунках нижче буде показано код класу HomeController, AttendanceCotroller, ScheduleController, які відповідають за обробку запитів, що надходять до домашньої сторінки або інших сторінок, пов’язаних з головною частиною веб-додатку. Він забезпечує зв’язок між запитами користувачів та представленням.

```
using LMSproject.Data;
using LMSproject.Models;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Rendering;
using Microsoft.EntityFrameworkCore;
using System.Diagnostics;
using System.Net;

namespace LMSproject.Controllers
{
    public class HomeController : Controller
    {

        StudentContext studentContext;
        IWebHostEnvironment appEnviroment;
        public HomeController(StudentContext studentContext, IWebHostEnvironment appEnviroment)
        {
            this.studentContext = studentContext;
            this.appEnviroment = appEnviroment;
        }
        public IActionResult AboutUs()
        {

            return View();
        }


```

Рисунок А.1 – Контекст БД та метод сторінки AboutUs

```
        public IActionResult Materials()
        {
            return View(studentContext.Files.ToList());
        }
        [HttpPost]
        public async Task<IActionResult> AddFile(IFormFile uploadedFile)
        {
            if (uploadedFile ≠ null)
            {
                //path to File folder
                string path = "/Files/" + uploadedFile.FileName;
                //save file in Files folder in catalod wwwroot

                using (var fileStream = new FileStream(appEnviroment.WebRootPath + path, FileMode.Create))
                    {
                        await uploadedFile.CopyToAsync(fileStream);
                    }
                    FileModel file = new FileModel { Name = uploadedFile.FileName, Path = path };
                    studentContext.Files.Add(file);
                    studentContext.SaveChanges();

            }
            return RedirectToAction("Materials");
        }
```

Рисунок А.2 – Метод сторінки Materials та метод Addfile, який реалізує завантаження файлів на сторінку

```
public IActionResult Login()
{
    return View();
}
[HttpPost]
public IActionResult Login(Student student)
{
    if (studentContext.Students.FirstOrDefault(x => x.Email == student.Email) != null
        && studentContext.Students.FirstOrDefault(x => x.Password == student.Password) != null)
        return View("News");
    else
        return View("Registration");

}

public IActionResult Registration()
{
    return View();
}
[HttpPost]
public IActionResult Registration(Student student)
{
    studentContext.Students.Add(student);
    studentContext.SaveChanges();
    return RedirectToAction("News");
}
```

Рисунок А.3 – Методи сторінки входу Login, Registration

```
    }
    public IActionResult News()
    {
        return View();
    }


    [ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]
    public IActionResult Error()
    {
        return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });
    }
}
```

Рисунок А.4 – Метод сторінки новин News

На рисунках нижче зображено програмний код класу AttendanceController

```
namespace LMSproject.Controllers
{
    public class AttendanceController : Controller
    {
        StudentContext studentContext;
        public AttendanceController(StudentContext studentContext)
        {
            this.studentContext = studentContext;
        }


        public IActionResult GetAttendance()
        {
            List<MyGroup> groups =studentContext.MyGroups.ToList();
            ViewData["groups"] = new SelectList(studentContext.MyGroups, "id", "Title");
            return View();
        }
        [HttpPost]
        public IActionResult GetAttendance(int id)
        {
            AttendanceVM vm = new AttendanceVM();
            vm.currentGroup = studentContext.MyGroups.FirstOrDefault(x => x.id == id);
            vm.Students=studentContext.Students.Include("Marks").Where(x => x.GroupId==id).ToList();
            vm.Attendances = studentContext.Attendances.Where(x => x.MyGroupid == id).ToList();
            ViewBag.NumOfLessons=studentContext.Attendances.Where(x=>x.MyGroupid==id).Count();
            ViewBag.Stud = vm.currentGroup.Student.FullName;

            ViewData["groups"] = new SelectList(studentContext.MyGroups, "id", "Title");
            return View("GetAttendance",vm);
        }
```

Рисунок А.5 – Контекст БД та метод сторінки журналу

```
public IActionResult MarkLesson(int id)
{

    List<MyGroup> groups = studentContext.MyGroups.ToList();
    ViewData["groups"] = new SelectList(studentContext.MyGroups, "id", "Title");
    ViewBag.teacher = studentContext.Teachers.FirstOrDefault(tch => tch.id == 1).name;
    Attendance attendance = new Attendance();
    attendance.MyGroupid = id;
    return View("MarkLesson", attendance);
}
[HttpPost]
public IActionResult MarkLesson(Attendance attendance)
{
    attendance.id = 0;
    attendance.Teacherid = 1;
    attendance.Studentid = 1;

    studentContext.Attendances.Add(attendance);
    studentContext.SaveChanges();
    ViewData["groups"] = new SelectList(studentContext.MyGroups, "id", "Title");
    return View("GetAttendance");
}
```

Рисунок А.6 – Метод відмічання заняття MarkLesson

```
public IActionResult EditLesson(int id)
{
    Attendance attendance = studentContext.Attendances.FirstOrDefault(x=>x.id==id);
    ViewBag.student = studentContext.Students.FirstOrDefault(st => st.id == attendance.Studentid).FullName;
    ViewBag.teacher = studentContext.Teachers.FirstOrDefault(tch => tch.id == 1).FullName;
    return View("EditLesson", attendance);
}
[HttpPost]
public IActionResult EditLesson(Attendance attendance)
{
    studentContext.Entry(attendance).State = EntityState.Modified;
    studentContext.SaveChanges();
    return RedirectToAction("GetAttendance");
}
public IActionResult DeleteLesson(int id)
{
    Attendance attendance = studentContext.Attendances.FirstOrDefault(x => x.id == id);
    ViewBag.student = studentContext.Students.FirstOrDefault(st => st.id == attendance.Studentid).FullName;
    ViewBag.teacher = studentContext.Teachers.FirstOrDefault(tch => tch.id == 1).FullName;
    return View("DeleteLesson", attendance);
}
[HttpPost]
public IActionResult DeleteLesson(Attendance attendance)
{
    studentContext.Entry(attendance).State = EntityState.Deleted;
    studentContext.SaveChanges();
    return RedirectToAction("GetAttendance");
}
```

Рисунок А.7 – Методи зміни заняття та видалення заняття

Нижче на рис. А.8 зображено програмний код класу ScheduleController

```
public class ScheduleController : Controller
{
    StudentContext studentContext;
    public ScheduleController(StudentContext studentContext)
    {
        this.studentContext = studentContext;

    }

    public IActionResult GetSchedule()
    {
        DateTime date = DateTime.Now;
        var firstDayOfMonth = new DateTime(date.Year, date.Month, 1);

        ViewBag.firstDayOfMonth = (int)(new DateTime(date.Year, date.Month, 1).DayOfWeek);
        ViewBag.daysInMonth = DateTime.DaysInMonth(date.Year, date.Month);
        var lastDayOfMonth = firstDayOfMonth.AddMonths(1).AddDays(-1);
        int id = 1;
       List<Lesson> lessons = studentContext.Lessons.Where(x => x.dateOfLesson.Month==DateTime.Now.Month).ToList();

        return View(lessons);
    }
}
```

Рисунок А.8 – Контекст БД та метод отримання розкладу

ДОДАТОК Б

Лістинг коду функціональних сторінок

Нижче буде показаний код представлення або сторінки моєї LMS системи. Як приклад я наведу такі сторінки: GetAttendance, DeleteLesson, MarkLesson, EditLesson, GetSchedule, Materials. Це базові сторінки, які необхідні для управління головним функціоналом системи

```
@model LMSproject.Models.AttendanceVM

<div id="selectBlock">

    <form name="getStudent" asp-action="GetAttendance" method="post">
        <select name="id" class="form-select"  asp-items="ViewBag.groups">
        </select>
        <input type="submit" class="btn btn-outline-primary" title="Показати" value="Показати" />
    </form>

</div>

<div>
    @if (Model != null)
    {
        <table id="attendance">
            <thead>
                <tr>
                    <th>
                        ПІБ
                    </th>
                    <th>
                        Загальна кількість відвідувань
                    </th>
                    @foreach (var item in Model.Attendances)
                    {
                        <th>@item.dateOfLesson</th>

                    }
                </tr>
            </thead>
            <tbody>
                <tr>
                    <td>
                    @ViewBag.Stud
```

Рисунок Б.1 – Лістинг сторінки GetAttendance

```
34                  <td>
35                  @ViewBag.Stud
36                  </td
37                  <td>@ViewBag.NumOfLessons</td>
38              </tr>
39              <tr>
40                  <th>
41                      Оцінка
42                  </th>
43                  <td></td>
44                  @foreach (var item in Model.Attendances)
45                  {
46                      if (item != null)
47                      {
48                          <td>@item.Mark</td>
49                      }
50                      else
51                      {
52                          <td>
53                              <input type="number" />
54                          </td>
55                      }
56                  }
57              </tr>
58              <tr>
59                  <th>
60                      Description of lesson
61                  </th>
62                  <td>
63                  </td>
64                  @if (Model != null)
65                  {
66
67                      foreach (var item in Model.Attendances)
68                      {
69                          <td>@item.description</td>
70                      }
71                  }
72              </tr>
73              <tr>
74                  <th>
```

Рисунок Б.2 – Продовження лістингу сторінки GetAttendance

```
                    <th>
                        Edit
                    </th>

                    <td>
                    </td>
                    @if (Model != null)
                    {

                        foreach (var item in Model.Attendances)
                        {
                            <td>
                                @Html.ActionLink("Редагувати запис", "EditLesson", "Attendance", new { id = item.id })
                                @Html.ActionLink("Видалити заняття", "DeleteLesson", "Attendance", new { id = item.id })
                            </td>
                        }
                    }
                </tr>
            </tbody>
        </table>

@Html.ActionLink("Відмітити заняття", "MarkLesson", "Attendance", new { id = Model.currentGroup.id })
    }
</div>
```

Рисунок Б.3 – Продовження лістингу сторінки GetAttandance

```
@model LMSproject.Models.Attendance

<form asp-action="MarkLesson" method="post">
    <input type="hidden" asp-for="MyGroupid" />
    <div>
        <h3>@ViewBag.student</h3>
    </div>
    <div>
        <h3>
            @ViewBag.teacher
        </h3>
        <select name="id" asp-items="ViewBag.groups">
        </select>
    </div>
    <div class="align-content-center">
        <input type="date" asp-for="dateOfLesson" />
        <label for="markInput">Оцінка (1-12)</label>
        <input type="number" asp-for="Mark" id="markInput" min="0" max="12" style="margin: 1.5rem 1rem; width: 150px" />
        <textarea asp-for="description" id="story" rows="4" cols="33">

        </textarea>
        <input type="submit" value="Додати" />
    </div>
</form>
```

Рисунок Б.4 – Лістинг сторінки MarkLesson

```
@model LMSproject.Models.Attendance
C:\Users\lenovo\Desktop\LMSproject\LMSProject fresh\Views\Attendance\GetAttendance.cshtml*
<form asp-action="EditLesson" method="post" asp-controller="Attendance">
    <input type="hidden" asp-for="id" />
    <input type="hidden" asp-for="Studentid" />
    <input type="hidden" asp-for="MyGroupid" />
    <div style="margin-top: 40px;">
        <h3>Студент: @ViewBag.student</h3>
    </div>
    <div style="margin-bottom: 40px">
        <h3>
            Вчитель: @ViewBag.teacher
        </h3>
        <input type="hidden" asp-for="Teacherid" />
    </div>
    <div class="align-content-center">
        <label for="markInput">Оцінка (1-12)</label>
         <input type="number" asp-for="Marks" id="markInput" min="0" max="12" style="margin: 1.5rem 1rem; width: 50px" />
        <input type="date" asp-for="dateOfLesson" />
        <div>
            <h6>
                Змінити коментар до заняття:
            </h6>
            <textarea asp-for="description" id="story" rows="4" cols="33">
        </textarea>
                <input style="margin-bottom: 30px; margin-left: 30px" type="submit" value="Додати" class="btn btn-primary" />
        </div>
    </div>
</form>
```

Рисунок Б.5 – Лістинг сторінки EditLesson

```
@model LMSproject.Models.Attendance

<form asp-action="DeleteLesson" method="post">
    <input type="hidden" asp-for="id" />
    <input type="hidden" asp-for="Studentid" />
    <input type="hidden" asp-for="MyGroupid" />
    <input type="hidden" asp-for="Teacherid" />
    <div style="margin-top: 40px;">
        <h3>Студент: @ViewBag.student</h3>
    </div>
    <div style="margin-bottom: 40px">
        <h3>
            Вчитель: @ViewBag.teacher
        </h3>
    </div>
    <div class="align-content-center">
        <label for="markInput">Оцінка (1-12)</label>
         <input type="number" asp-for="Marks" id="markInput" min="0" max="12" style="margin: 1.5rem 1rem; width: 50px" />

        <input type="date" asp-for="dateOfLesson" />
        <div>
        <h6>
            Змінити коментар до заняття:
        </h6>
        <textarea asp-for="description" id="story" rows="4" cols="33"></textarea>
            <input style="margin-bottom: 30px; margin-left: 30px"  type="submit" value="Видалити урок" class="btn btn-danger" />
        </div>
    </div>

</form>
```

Рисунок Б.6 – Лістинг сторінки DeleteLesson

```
@model List<Lesson>

<style>
    td {
        border: 1px solid black;
    }
</style>

<table id="schedule">
    <thead>
        <tr>
            <th>Monday</th>
            <th>Tuesday</th>
            <th>Wendesday</th>
            <th>Thirstday</th>
            <th>Friday</th>
            <th>Saturday</th>
            <th>Sunday</th>
        </tr>
    </thead>
    <tbody>
        @{
            int a = 1;
            @for (int i = 0; i < 5; i++)
            {
                <tr>

                    @for (int j = 0; j < 7; j++)
                    {

                        @if (i == 0)
                        {
                            if (j ≥ ViewBag.firstDayOfMonth)
                            {
                                <td style="width:50px; height:50px; background-color: green;">
                                    @{
                                        a++;
                                        for (int k = 0; k < Model.Count; k++)
                                        {
                                            if (a == Model[k].dateOfLesson.Day)
                                            {
```

Рисунок Б.7 – Лістинг сторінки GetSchedule

```
                                @Model[k].dateOfLesson
                                ;
                                @Model[k].timeOfLesson
                                ;
                                @Model[k].MyGroup.Title
                                ;
                            }
                        }
                    }

                </td>
            }
            else
            {
                <td style="width:50px; height:50px; background-color: gray;"></td>
            }
        }
        else
        {
            if (a > 1 && a < ViewBag.daysInMonth)
            {
                <td style="width:50px; height:50px; background-color: green;">
                    @{
                        a++;
                        for (int k = 0; k < Model.Count; k++)
                        {
                            if (a == Model[k].dateOfLesson.Day)
                            {
                                @Model[k].dateOfLesson
                                ;
                                @Model[k].timeOfLesson
                                ;
                            }
                        }
                    }
                </td>
            }
            else
            {
```

Рисунок Б.8 – Продовження лістингу сторінки GetSchedule

```
                {
                    <td style="width:50px; height:50px; background-color: gray;"></td>
                }
            }
        }
    </tr>
}
    }
</tbody>
</table>
```

Рисунок Б.9 – Продовження лістингу сторінки GetSchedule