МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ІНФОРМАЦІЙНО-ОСВІТНЬОГО ПОРТАЛУ СТУДЕНТА»**

Виконала: студентка 4 курсу, групи 6.1220
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)
освітньої програми Комп'ютерні науки
(назва освітньої програми)
М.Д. Нагорна
(ініціали та прізвище)
Керівник завідувачка кафедри комп'ютерних наук, доцент, д.т.н. Шило Г.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент завідувач кафедри програмної інженерії, к.ф.-м.н., доцент, Лісняк А.О.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерних наук

Рівень вищої освіти бакалавр

Спеціальність 122 Комп'ютерні науки
(шифр і назва)

Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук,
д.т.н., доцент

_____ Шило Г.М.
(підпис)

" ____ " _____ 2024 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТЦІ

Нагорній Марії Дмитрівні
(прізвище, ім'я та по-батькові)

1. Тема роботи Розробка інформаційно-освітнього порталу студента

керівник роботи Шило Галина Миколаївна, д.т.н., доцент
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 21 » грудня 2023 року № 2180-с

2. Строк подання студентом роботи 05.06.2024

3. Вихідні дані до роботи 1. Постановка задачі.
2.Перелік літератури.
3. Схема інформаційних систем ЗНУ.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Розділи Вступ, Аналіз технічного завдання, Проєктування системи, Розробка модуля розкладу

4. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 22.12.2023

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	27.12.2023	
2.	Збір вихідних даних.	10.01.2024	
3.	Обробка методичних та теоретичних	10.02.2024	
	джерел.		
4.	Розробка першого та другого розділу.	20.03.2024	
5.	Розробка третього розділу.	29.04.2024	
6.	Оформлення та нормоконтроль	15.05.2024	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	23.06.2024	

Студент _____ М.Д. Нагорна
(підпис) (ініціали та прізвище)

Керівник роботи _____ Г.М. Шило
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г.Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка інформаційно-освітнього порталу студента»: 37 с., 22 рис., 1 табл., 10 джерел.

ВИКЛАДАЧ, ЕЛЕКТРОННИЙ КАБІНЕТ, ЗАКЛАД ВИЩОЇ ОСВІТИ, ІНФОРМАЦІЙНО-ОСВІТНІЙ ПОРТАЛ, ОСВІТНІЙ КОМПОНЕНТ, ПРОЄКТУВАННЯ, РОЗКЛАД, СТУДЕНТ, ФРЕЙМВОРК.

Об’єкт дослідження – розробка та впровадження інформаційно-освітнього порталу студента.

Мета роботи: розробка модуля розкладу для інформаційно-освітнього порталу студента.

Методи дослідження – порівняння, аналіз існуючих рішень, експеримент.

Розроблена інформаційно-освітня платформа містить програмне забезпечення для керування розкладом занять, зберігання та передачі даних з можливістю візуалізації у зручному користувацькому інтерфейсі.

Платформа реалізована за допомогою клієнтської частини на базі пакету Vuetify, а також серверної частини на фреймворку Laravel. Передача даних між клієнтською та серверною частинами здійснюється за допомогою HTTP-запитів через бібліотеку Axios.

Розроблена платформа дозволяє створити продуктивні умови для навчання і полегшити управління розкладом занять, забезпечуючи зручність і комфорт користувачів.

SUMMARY

Bachelor’s Qualifying Theses «Development of the student information and educational portal»: 37 pages, 22 figures, 1 tables, 10 references.

TEACHER, ELECTRONIC CABINET, INSTITUTION OF HIGHER EDUCATION, INFORMATION AND EDUCATIONAL PORTAL, EDUCATIONAL COMPONENT, DESIGN, SCHEDULE, STUDENT, FRAMEWORK.

Object of the study – development and implementation of the student's informational and educational portal.

Aim of the study: development of the schedule module for the student's information and educational portal.

Method of research – comparison, analys of existing solutions, experiment.

The developed information and educational platform includes software for managing the lesson schedule, storing and transmitting data with the possibility of visualization in a convenient user interface.

The platform is implemented using the client part based on the Vuetify package, as well as the server part based on the Laravel framework. Data transfer between the client and server parts is carried out using HTTP requests through the Axios library.

The developed platform allows you to create productive conditions for learning and facilitate the management of the lesson schedule, ensuring the convenience and comfort of users.

ЗМІСТ

ВСТУП

В сучасних умовах інформаційно-освітні портали відіграють вирішальну роль у системі вищої освіти. Ці онлайн-платформи стали необхідною складовою освітнього процесу, надаючи користувачам широкий спектр можливостей через Інтернет.

Зростання потреб у доступній та ефективній освіті стимулює розвиток інформаційно-освітніх порталів. Суспільство потребує гнучких та динамічних освітніх систем, які б відповідали викликам часу. Інформаційно-освітні портали роблять освіту доступною для широкого кола користувачів, незалежно від їхнього місцезнаходження та часу навчання.

Стрімкий розвиток інформаційних технологій створив сприятливі умови для проєктування та впровадження онлайн-платформ для навчання. Завдяки цьому інформаційно-освітні портали стають більш функціональними та зручними у використанні.

Сучасні педагоги дедалі частіше використовують інтерактивні та практико-орієнтовані методи навчання, які потребують використання онлайн-ресурсів. Інформаційно-освітні портали слугують ефективним інструментом для реалізації таких методів.

Однією з актуальних задач, яка потребує вдосконалення можливостей інформаційно-освітніх порталів, є розробка модуля розкладу занять. Традиційні методи надання розкладу, такі як публікація на веб-сайтах або розповсюдження друкованих копій, мають низку недоліків: неактуальність інформації, складність пошуку, незручність використання. Інтеграція модуля розкладу до інформаційно-освітнього порталу надає можливість вирішити ці проблеми та покращити загальну ефективність порталу.

Впровадження модуля розкладу занять дозволить зробити інформаційно-освітні портали більш функціональними та корисними для всіх учасників освітнього процесу. Це сприятиме покращенню якості освіти та зробить її більш доступною для широкого кола користувачів. Отже, метою кваліфікаційної роботи є розробка модуля розкладу для інформаційно-освітнього порталу студента

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Аналіз інформаційно-освітніх систем закладів вищих освіти в Україні та закордоном

В Україні, як і в європейських країнах, інтернет-індустрія для освітнього процесу досить розвинена. Університети найчастіше використовують власні інформаційні та освітні портали, які забезпечують ефективну організацію навчального процесу та сприяють підвищенню якості освіти.

Інформаційні та освітні портали є невід'ємною частиною сучасної системи освіти, оскільки вони надають студентам та викладачам легкий доступ до різних освітніх ресурсів. Такі портали найчастіше містять функції: доступ до навчальних матеріалів та розкладу занять, подання завдань, участь у форумах та дискусіях, а також отримання оцінок та відгуків від викладачів. Інтеграція з електронними бібліотеками та іншими академічними ресурсами також є важливим фактором, який значно підвищує рівень освіти студентів.

Для порівняння розглянемо освітні платформи двох університетів: Університет Палацького в Чехії та Сумський державний університет. Результати порівняння наведені в таблиці 1.1

Таблиця 1.1 – Порівняння освітніх платформ університетів

Критерій	Сумський державний університет	Університет Палацького в Чехії
Назва платформи	eLearning	STAG
Основні функції	Доступ до навчальних матеріалів	Доступ до навчальних матеріалів
	Розклад занять	Розклад занять

Продовження таблиці 1.1

Критерій	Сумський державний університет	Університет Палацького в Чехії
Основні функції	Завантаження домашніх завдань	Можливість запису на курси
	Проведення тестів	Проведення тестів
	Оцінювання	Перевірка балів за курс
	Комунікація через форуми та обговорення	Комунікація через форуми та обговорення
	Особистий кабінет	Особистий кабінет
Додаткові функції	Інтеграція з електронними бібліотеками	Інтеграція з електронними бібліотеками
	Можливість проведення онлайн-лекцій та відеоконференцій	Можливість проведення онлайн-лекцій та відеоконференцій
	Календар іспитів, заліків та інших подій	Календар іспитів, заліків та інших подій
	Сповіщення та новини	Сповіщення та новини
Тип платформи	Власна розробка на базі Moodle	Власна розробка

Обидві освітні платформи демонструють високий рівень розвиненості та ефективності у підтримці навчального процесу. Платформа Сумського державного університету, базована на Moodle, надає широкий спектр готових рішень та модулів, що може бути перевагою для швидкої імплементації та гнучкості. Платформа Університету Палацького в Чехії, будучи власною розробкою, забезпечує високу адаптивність та специфічні рішення для потреб університету.

Ключовим елементом обох систем є функція розкладу занять, яка забезпечує організованість навчального процесу. Завдяки цьому, студенти та викладачі можуть легко стежити за змінами в розкладі та ефективно розподіляти свій навчальний та робочий час.

У Сумському державному університеті модуль розкладу надає користувачам можливість формування запитів за різними критеріями, такими як дата, група, викладач або аудиторія (рис. 1). Після введення параметрів запиту користувач може переглянути розклад занять, які відповідають цим критеріям, та експортувати його у форматах ICAL або PDF. Крім того, викладачі мають можливість додавати короткі коментарі до своїх занять, надаючи користувачам додаткову інформацію про них.

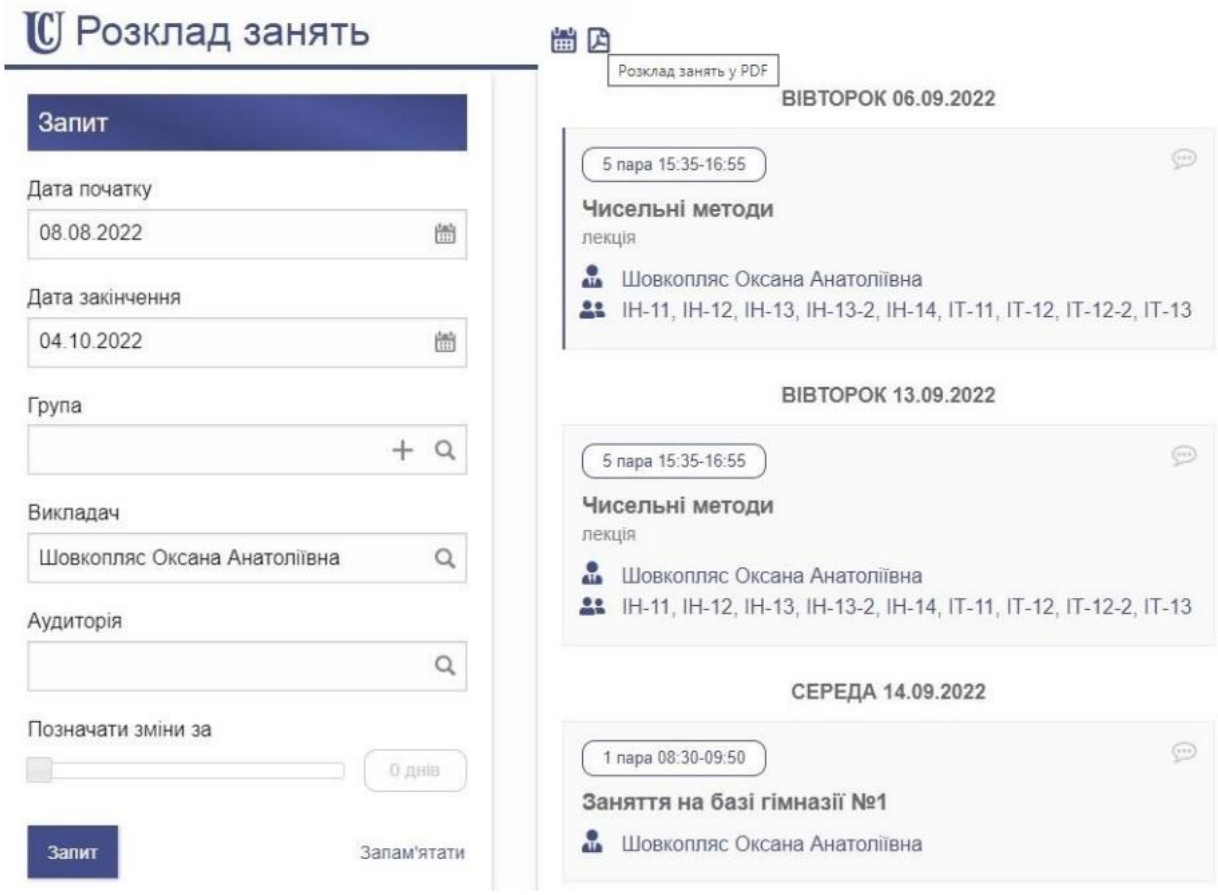


Рисунок 1 – Модуль розкладу занять Сумського державного університету

Університет Палацького реалізував зручну систему для своїх зареєстрованих користувачів, яка надає можливість перегляду особистого

розкладу. Кожен користувач може обрати потрібний семестр, рік та вид представлення розкладу за власним вибором. В результаті, він отримує повну інформацію про заняття на весь тиждень, включаючи час їх проведення (рис. 2).

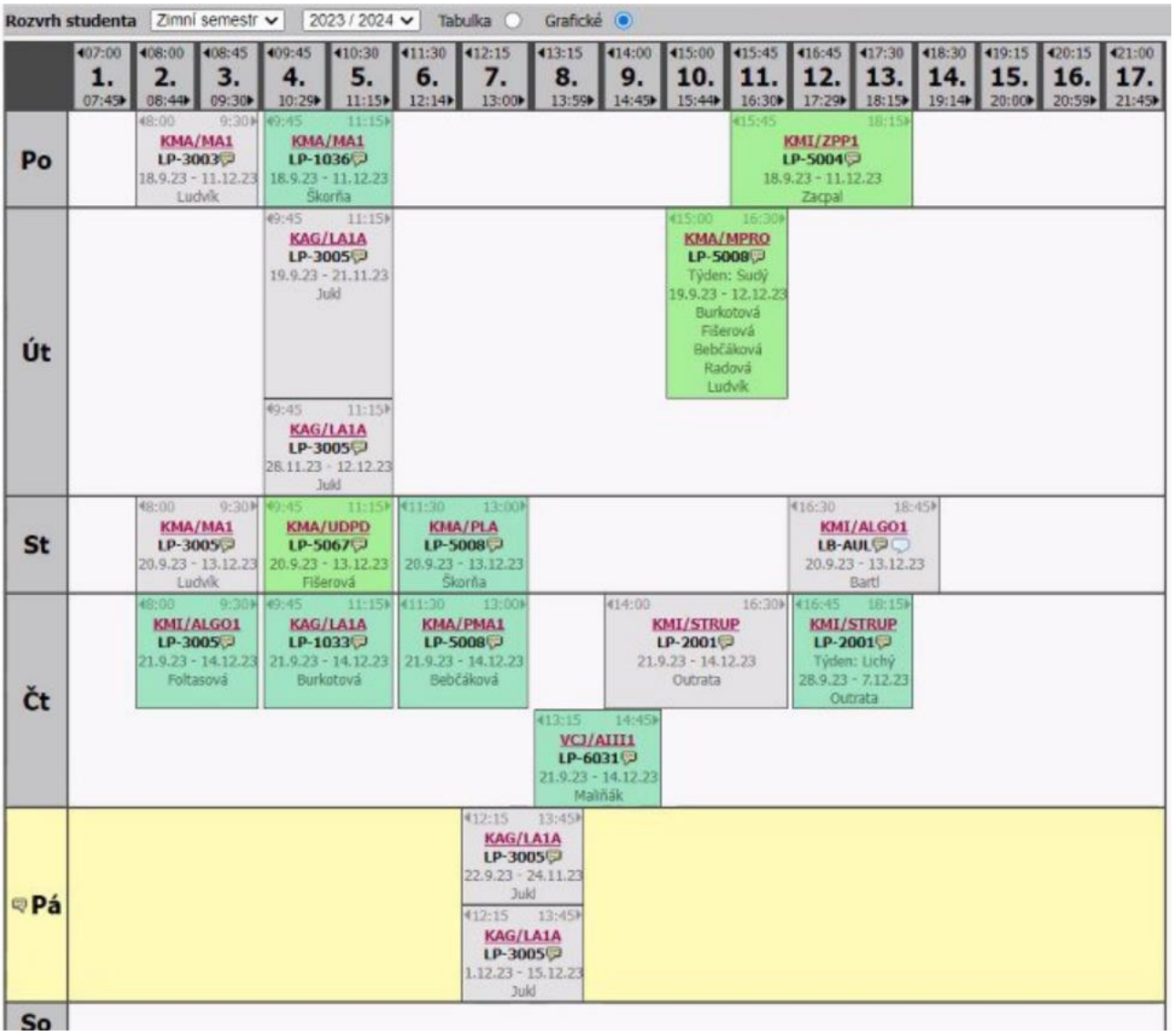


Рисунок 2 – Модуль розкладу занять Університету Палацького

Загалом, обидві платформи відіграють ключову роль у сучасному освітньому процесі, забезпечуючи доступність, інтерактивність та ефективність навчання для студентів та викладачів.

Отже, результати аналізу аналогів використано для визначення основних функціональних можливостей модуля розкладу для Запорізького національного університету.

1.2 Функціональні вимоги до модуля розкладу занять

Функціональні вимоги до розкладу занять визначають основні можливості системи, які гарантують ефективне організоване планування навчального процесу. Модуль розкладу занять є необхідною складовою як для студентів, так і для викладачів, тож функціональні вимоги можуть відрізнятися для цих двох груп користувачів:

а) перегляд розкладу:

- система повинна дозволяти користувачам усіх рівнів переглядати розклад занять тільки для своєї групи (для студента) чи прізвища (для викладача);
- інформація про розклад повинна бути чітко структурованою та легко читабельною;
- користувачі повинні мати можливість бачити всю інформацію про пару, таку як назва предмета, номер групи, прізвище викладача, день тижня, час заняття тощо;

б) редагування розкладу:

- викладачі та працівники університету повинні мати можливість вносити зміни до розкладу занять, наприклад, додавати нові заняття, видаляти заняття, змінювати час або місце проведення занять, викладача, групу;
- система повинна автоматично оновлювати інформацію;
- система повинна мати систему узгодження змін до розкладу, щоб уникнути конфліктів.

Впровадження таких функцій вимагає глибокого розуміння структури та потреб університету. Система повинна бути інтегрована з існуючими адміністративними та навчальними процесами, щоб забезпечити максимальну ефективність та зручність для користувачів.

1.3 Інтеграція з інформаційною системою університету

В Запорізькому національному університеті інформаційна система включає в себе ряд модулів, таких як:

- облік мешканців гуртожитку;
- система обліку балів за позанавчальну діяльність;
- відділ кадрів працівників;
- відділ кадрів студентів;
- сервіс авторизації;
- документообіг;
- електронний журнал;
- бібліотека;
- система збору та аналізу ключових даних для обліку, аудиту та контролю якості інтернаціоналізації;
- телефонний довідник.

Інтеграція модуля розкладу занять з існуючими модулями відкриває нові горизонти для оптимізації навчального процесу, покращуючи комунікацію та синхронізацію інформації між різними підрозділами університету.

Ось декілька прикладів того, як модуль розкладу занять може бути інтегрований з існуючими модулями:

- взаємодія з модулем реєстрації та авторизації студентів та викладачів дозволить автоматично оновлювати дані про групи та розклади;
- співпраця з бібліотечним модулем дозволить відображати ресурси, необхідні для підготовки до занять, і резервувати навчальні матеріали;
- синхронізація з модулем інтернаціоналізації сприятиме більш точній адаптації розкладу для іноземних студентів

Ці інтеграції забезпечать повноту та зручність користування системою для всіх учасників навчального процесу. Взаємодія з різними модулями

інформаційної системи університету дозволить автоматизувати та спростити багато аспектів планування та організації навчання.

1.4 Постановка задачі та основні функції

Постановка задачі:

- дослідження інформаційних порталів університетів;
- аналіз функціональних можливостей інформаційної системи ЗНУ;
- розробка структури модуля розкладу;
- вибір засобів реалізації;
- реалізація компонентів системи.

Основними функціями модуля розкладу інформаційно-освітнього порталу є:

а) для студента:

- перегляд актуального розкладу для своєї групи;
- інформація про кожне заняття: назва предмету, номер групи, викладач, день тижня, час, місце проведення;
- зручний інтерфейс для навігації по розкладу
- можливість перегляду типу тижня: чисельник та знаменник
- перегляд розкладу за будь-який день, місяць чи рік

б) для викладача:

- додавання, видалення та редагування занять;
- перегляд актуального розкладу для себе;
- інформація про кожне заняття: назва предмету, номер групи, викладач, день тижня, час, місце проведення;
- зручний інтерфейс для навігації по розкладу
- можливість перегляду типу тижня: чисельник та знаменник
- перегляд розкладу за будь-який день, місяць чи рік
- автоматичне оновлення розкладу для всіх користувачів.

2 ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Проєктування архітектури

Проєктування архітектури застосунку включає визначення основних компонентів системи, їх взаємодії та розподілу відповідальностей. Архітектура модуля розкладу занять базується на принципах MVC (Model-View-Controller), де:

- model (модель) База даних та основна логіка програми для роботи з нею, обробки запитів, проведення обчислень тощо [3];
- view (відображення): інтерфейс користувача, усі візуальні елементи, з якими буде взаємодіяти користувач [3];
- controller (контролер): обробку введених даних від користувача, оновлення моделі та вибір відповідного відображення для користувача.

Основними компонентами системи є користувачі, розклад занять, деталі заняття та редагування занять.

Для точного моделювання взаємодії між цими компонентами використовується UML-діаграма. Діаграма UML (уніфікована мова моделювання) – це тип схеми, яка представляє структуру та поведінку програмних систем. Зазвичай вона включає заняття, об'єкти, взаємозв'язки та взаємодії [2].

Спочатку представимо діаграму прецедентів, яка допоможе візуалізувати взаємодію між різними компонентами системи та їхніми користувачами (рис. 3).

Після розгляду функціональних вимог через надану діаграму, наступним кроком є побудова UML діаграми класів, що відображає статичну структуру системи, класи та відносини між ними (рис. 4). Діаграма класів є одним із найважливіших типів діаграм у UML, що показує, як саме функціональні вимоги будуть реалізовані всередині системи. Вона допоможе візуалізувати взаємодію між різними компонентами системи та їхніми користувачами.

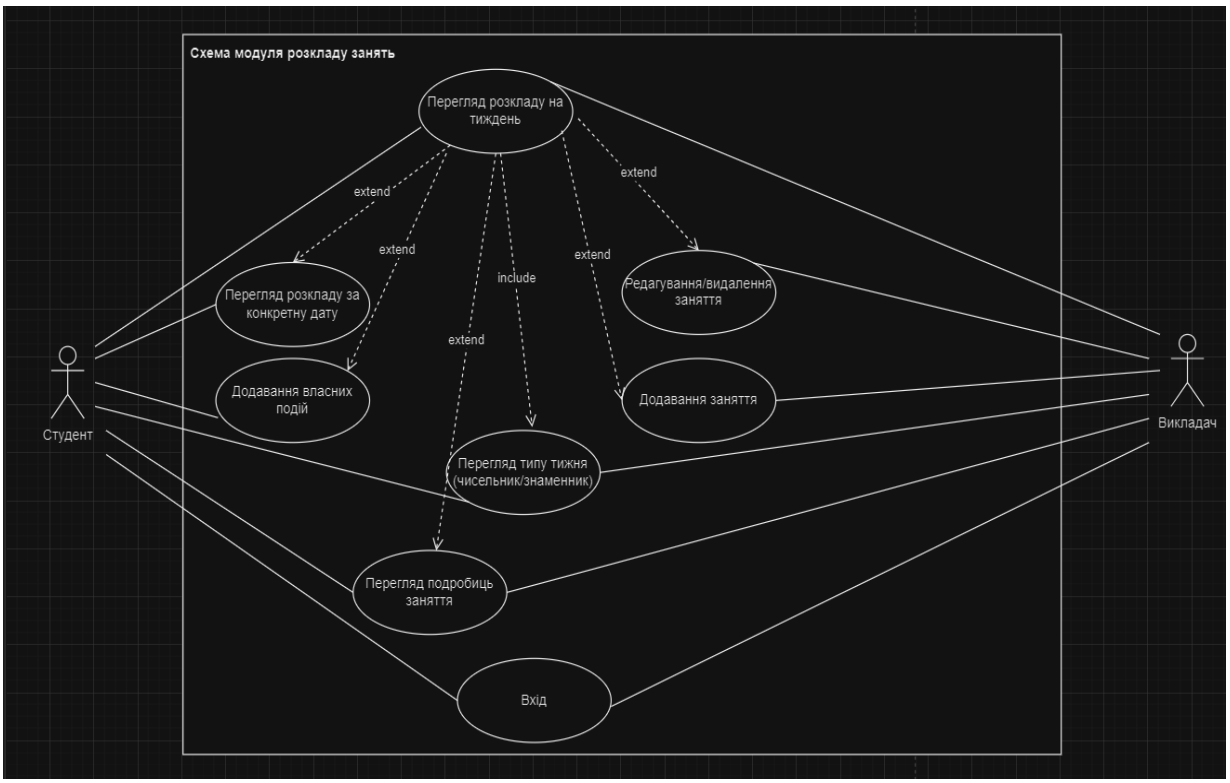


Рисунок 3 – UML діаграма прецедентів

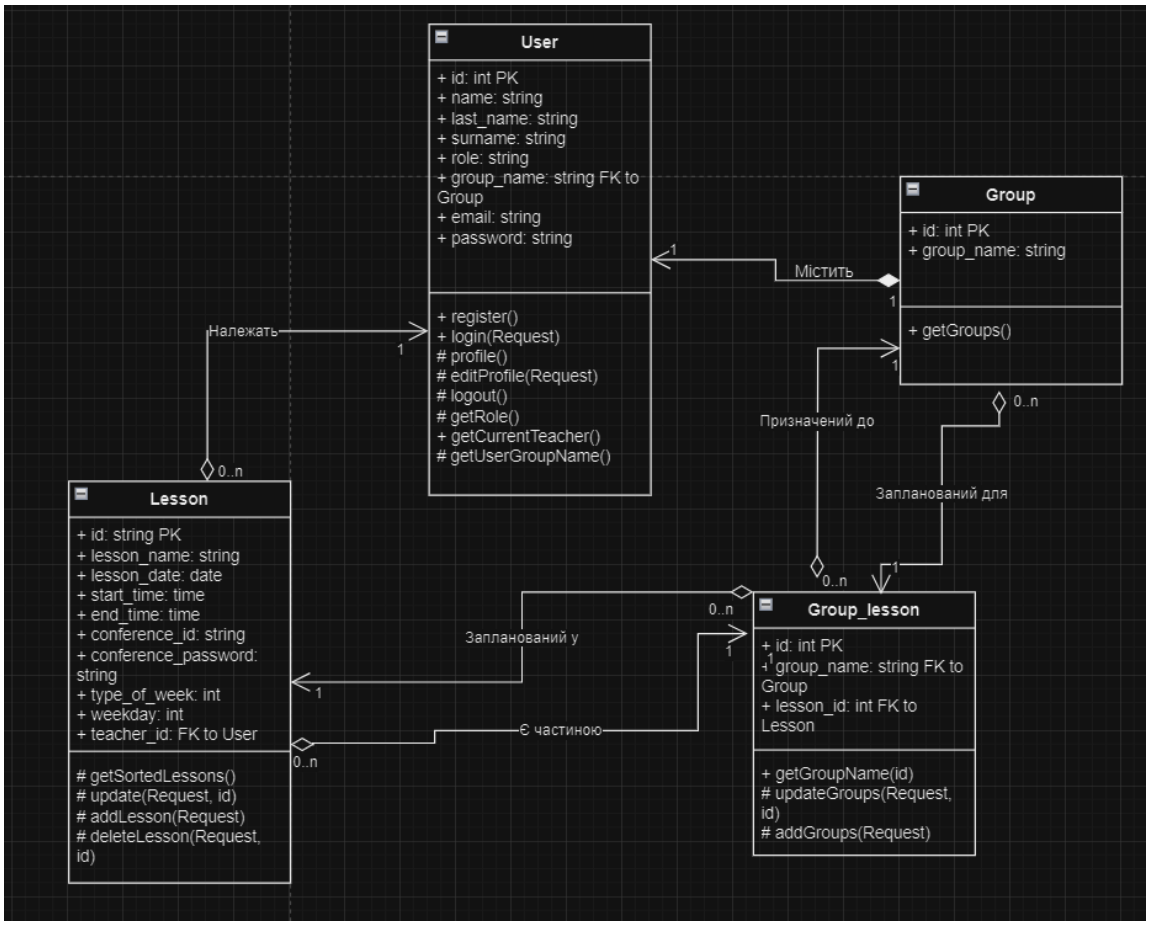


Рисунок 4 – UML діаграма класів

У наведеній діаграмі (рис. 4) наявні наступні зв’язки, бо:

- кожен користувач може мати багато предметів;
- кожен студент належить до певної групи, але одна група може містити багато студентів;
- кожен предмет викладається певним викладачем, але один викладач може викладати кілька предметів;
- багато предметів можуть належати до багатьох груп.

2.2 Концептуальне проєктування бази даних

Концептуальне проєктування бази даних є ключовим етапом у створенні ефективної та функціональної системи управління розкладом занять для студентів. Воно включає визначення основних сутностей, їх атрибутів і зв'язків між ними, що дозволяє побудувати логічну структуру даних, необхідну для задоволення потреб користувачів.

2.2.1 Вибір моделі бази даних

Для розробки модуля розкладу занять було обрано реляційну модель даних.

Ця модель використовує таблиці для відображення даних, які мають взаємозв’язок. Вона представляє дані у вигляді рядків, де кожен рядок відображає певний набір даних, а стовпці представляють окремі атрибути цих даних. Зв’язки між таблицями створюються за допомогою ключів, які дозволяють пов’язувати дані.

Реляційна модель даних широко використовується у програмному забезпеченні через свою простоту, гнучкість та масштабованість. Вона дозволяє ефективно зберігати та отримувати дані, підтримує складні запити та транзакції. Реляційна модель також ідеально підходить для обробки великих обсягів даних.

У контексті модуля розкладу занять, реляційна модель даних є найкращим вибором, оскільки вона дозволяє легко організовувати дані у вигляді таблиць та встановлювати зв'язки між різними аспектами розкладу. Це особливо важливо для зберігання і управління інформацією про розклад занять, оскільки ці дані можуть бути розподілені в різних таблицях, але потребують пов'язаності між собою для ефективного відображення та управління розкладом студента.

2.2.2 Вибір СУБД

Для розробки модуля розкладу занять було обрано MySQL – одну з найпопулярніших реляційних систем управління базами даних. MySQL відома своєю надійністю, швидкодією та широким спектром функцій, які забезпечують ефективну роботу з даними. Ця СУБД також підтримує складні запити та транзакції, що є важливим для даного проєкту.

2.2.3 ER-діаграма бази даних

ER-діаграма (діаграма «сутність-зв’язок») є важливим інструментом при візуалізації структури бази даних. Вона дозволяє визначити сутності, їхні атрибути та зв'язки між ними, що полегшує проєктування та розробку бази даних.

База даних міститиме такі таблиці: «Заняття», «Користувачі», «Групи», «Групи та заняття». Кожна таблиця буде пов’язана з іншими таблицями за допомогою зовнішніх ключів.

Таблиця «Користувачі» пов’язана з таблицею «Заняття» через поле «id», а з таблицею «Групи» через поле «назва групи».

Таблиця «Групи та заняття» виступає зв’язуючою для таблиць «Заняття» та «Групи» за допомогою полів «id» з таблиці «Заняття» та «назва групи» з таблиці «Групи». Це обумовлено тим, що кожне заняття може проводитися у декількох групах, так само як одна група може мати декілька занять.

Відповідно, ER-діаграма для даного випадку виглядатиме наступним чином (див. рис. 5).

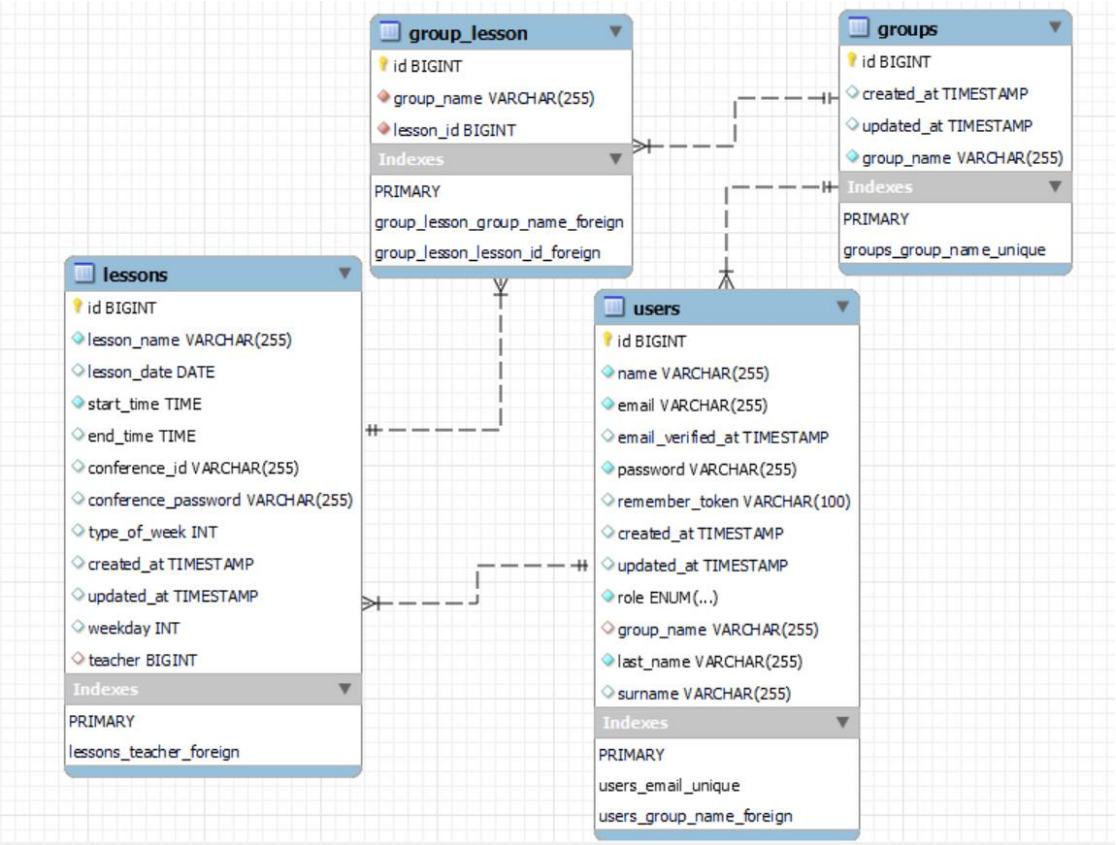


Рисунок 5 – ER-діаграма бази даних.

2.3 Вибір фреймворків та мови програмування

Вибір фреймворку та мови програмування є чи не найважливішою частиною розробки програмного забезпечення. Саме тому до цього питання слід підійти якомога відповідальніше.

Існує величезний вибір різних мов програмування. Проте для розробки модуля розкладу занять було обрано PHP для серверної частини та JavaScript для клієнтської.

Вибір мови PHP обумовлений її популярністю, широким розповсюдженням у розробці застосунків, підтримкою великої кількості

фреймворків та бібліотек, а також високою продуктивністю при роботі з базами даних

Вибір JavaScript для клієнтської частини обумовлений його універсальністю, можливістю створення інтерактивного користувацького інтерфейсу, широкою підтримкою фреймворків та бібліотек, а також високою продуктивністю та кросплатформеністю. Саме на цій мові програмування написана клієнтська частина таких сайтів, як Google, Facebook, Microsoft, Apple, YouTube, Instagram, Linkedin та багато іншого. Вибір мови JavaScript такими відомими компаніями свідчить про її прогресивність та надійність.

Для даних мов програмування було обрано такі фреймворки: Laravel та Vue.js.

Laravel забезпечує стабільну та гнучку серверну частину, що дозволяє легко інтегруватися з іншими системами, а також пропонує такі вбудовані функції, як логіка автентифікації, драйвери поштових бібліотек, безпека та міжсайтові сценарії, обробка помилок та винятків, веб-маршрутизація та черга повідомлень, вбудовані в систему [4].

Тоді як Vue.js забезпечує динамічний та інтерактивний інтерфейс користувача, відзначається простотою у використанні, реактивністю, гнучкістю, високою продуктивністю, модульністю, великою екосистемою, підтримкою TypeScript, двостороннім зв'язуванням даних та зручними інструментами для розробників.

3 РОЗРОБКА МОДУЛЯ РОЗКЛАДУ

3.1 Налаштування середовища розробки

Розробка застосунку на основі фреймворків Laravel для серверної частини та Vue.js для клієнтської частини потребує правильного налаштування оточення, конфігурації проєкту та інтеграції між цими двома компонентами.

У процесі розробки програмного застосунку використовується комп’ютер, на якому встановлена операційна система Windows. Однак, для оптимізації та спрощення процесу налаштування розробницького середовища, більш доцільним є використання Linux. Враховуючи це, рекомендується встановити Windows Subsystem for Linux (WSL).

Для створення проєкту Laravel необхідно прочитати офіційну документацію, де радять встановити Composer, PHP, Node.js та NPM і Docker.

Composer – це менеджер залежностей для PHP, який спрощує процес управління бібліотеками та пакетами у ваших проектах. По суті, це інструмент командного рядка, який допомагає легко встановлювати, оновлювати та автоматично завантажувати пакети PHP та їх залежності [5].

Docker – це інструмент для запуску програм і служб у невеликих, легких «контейнерах», які не заважають встановленому програмному забезпеченню або конфігурації вашого локального комп’ютера. Це означає, що не доведеться турбуватися про конфігурацію або налаштування складних інструментів розробки, таких як веб-сервери та бази даних на локальній машині [6].

Після встановлення основних інструментів та розгортання проєкту Laravel інсталюємо Vue.js та підключаємо базу даних.

3.2 Встановлення залежностей

У процесі розробки застосунків одним з ключових аспектів є аутентифікація користувача. Це важливий елемент, який забезпечує безпеку даних та персоналізацію досвіду користувача. В цьому контексті, пакет Laravel Sanctum виявляється надзвичайно корисним. Laravel Sanctum надає простий та елегантний спосіб реалізації аутентифікації користувача. Він використовується для створення нового користувача в системі, що дозволяє користувачам легко реєструватися та отримувати доступ до сервісів.

Також велику увагу було приділено візуальному представленню інтерфейсу користувача. Для цього було використано Vuetify – бібліотеку компонентів Vue.js, яка дозволяє ефективно та швидко створювати високоякісні, естетично привабливі інтерфейси користувача, що відповідають стандартам Material Design.

3.3 Розробка клієнтської частини

Розробка користувацького інтерфейсу є критично важливою частиною створення будь-якого програмного забезпечення, оскільки саме інтерфейс забезпечує взаємодію користувача з системою. У цьому підрозділі будуть розглянуті основні аспекти створення інтерфейсу, включаючи дизайн елементів, розробку алгоритмів для обробки дій користувача, а також побудову зручного та інтуїтивно зрозумілого інтерфейсу. Грамотно спроектований інтерфейс підвищує зручність використання системи, покращує користувацький досвід і забезпечує ефективну реалізацію функціональних можливостей додатку.

3.3.1 Форми в проєкті

Форми є важливим компонентом будь-якого застосунку, оскільки вони дозволяють користувачам взаємодіяти з системою, вводячи дані та отримуючи відповідь.

У даному проєкті всі форми були розроблені за допомогою тегу v-form з пакету Vuetify. На рисунку 6 представлено структуру форми Ця структура включає різні елементи, такі як поля для введення тексту, кнопки та інші компоненти, які використовуються для збору вхідних даних від користувача.

```
<v-card
  class="wrapper mx-auto pa-12 pb-8"
  elevation="8"
  max-width="448"
  rounded="lg"
>
  <v-form
    v-model="form"
    @submit.prevent="login"
    >
    <v-text-field v-model="email" :counter="30" prepend-inner-icon="mdi-email-outline" :rules="emailRules" label="Email" type="email" variant="outlined" required></v-text-field>
    <v-text-field
      v-model="password"
      :counter="50"
      :append-inner-icon="visible ? 'mdi-eye' : 'mdi-eye-off'"
      :rules="passwordRules"
      :type="visible ? 'text' : 'password'"
      prepend-inner-icon="mdi-lock-outline"
      variant="outlined"
      @click:append-inner="visible = !visible"
      label="Пароль"
      required
    ></v-text-field>
      <v-btn
        :disabled="!form"
        :loading="loading"
        class="mb-8"
        color="blue"
        size="large"
        variant="tonal"
        block
        type="submit"
        >
        Увійти
      </v-btn>
  </v-form>
      <v-card-text class="text-center">
        <router-link to="/register"
          class="text-blue text-decoration-none"
          rel="noopener noreferrer"
        >
          Немає акаунту? Зареєструватися <v-icon icon="mdi-chevron-right"></v-icon>
        </router-link>
      </v-card-text>
</v-card>
```

Рисунок 6 – Приклад форми

Ця форма містить два поля для введення тексту (v-text-field): одне для введення електронної пошти користувача, а інше – для введення пароля. Кожне поле має свої власні правила валідації (:rules), які перевіряють введені дані на відповідність таким критеріям: заповненість поля та кількість символів (див.

рис. 7). Кнопка входу (v-btn) активується, коли всі поля форми правильно заповнені.

Поле введення пароля дозволяє користувачам вмикати та вимикати відображення введених символів, натискаючи на іконку «око».

```
passwordRules: [
  v => !!v || 'Заповніть поле.',
  v => v.length >= 6 || 'Мінімум 6 символів',
],

emailRules: [
  v => !!v || 'Заповніть поле',
  v => (v && v.length >= 10) || 'Мінімум 10 символів',
],
```

Рисунок 7 – Приклад правил заповнення полей

Отже, в результаті використання вищезазначеної структури отримуємо форму, яку можна побачити на рис. 8.

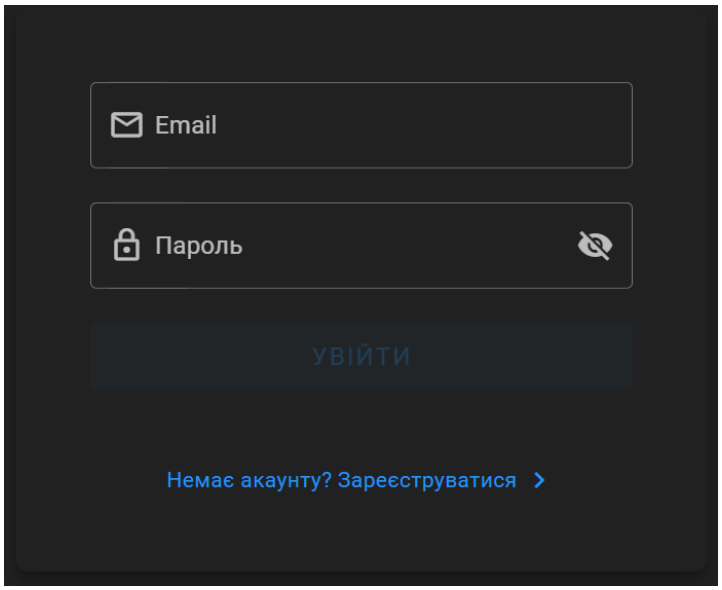


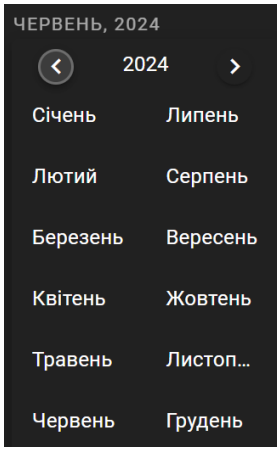
Рисунок 8 – Зовнішній вигляд форми

3.3.2 Створення календаря

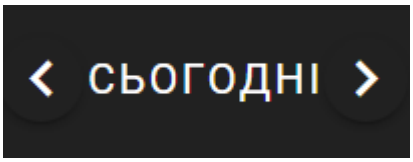
Компонент календаря був розроблений з метою надання користувачам зручного та інтуїтивно зрозумілого інтерфейсу для перегляду своїх занять.

Спочатку, при відкритті календаря, користувачу автоматично показується поточний тиждень з відповідними заняттями. Це дозволяє користувачам швидко ознайомитися зі своїм розкладом на найближчий час.

Крім того, даний компонент надає можливість перегляду занять за різні періоди часу: за конкретний тиждень, місяць або рік (див. рис. 9). Це забезпечує гнучкість та адаптивність інтерфейсу, враховуючи індивідуальні потреби користувача.



а) вибір місяця та року



б) вибір тижня та кнопка повернення на поточний тиждень

Рисунок 9 – Варіанти перегляду занять за різні періоди часу

Незалежно від обраного періоду часу, користувач завжди може швидко повернутися до перегляду поточного тижня, натиснувши на кнопку «сьогодні». Ця функція забезпечує зручність навігації та дозволяє легко орієнтуватися в своєму розкладі.

Також, при переході до будь-якої дати, видно тип тижня: чисельник чи знаменник.

При завантаженні сторінки спрацьовує функція getWeek(), яка обчислює дати кожного дня поточного тижня. Вона визначає перший день тижня і створює об'єкти для кожного дня тижня, які включають дату, день тижня та номер тижня. Ці об'єкти зберігаються у масиві week, який потім повертається. (рис. 10).

```
getWeek() {
    let week = [];
    let first = this.current.getDate() - this.current.getDay() + (this.current.getDay() === 0 ? -6 : 1);
    for (let i = 0; i < 6; i++) {
        let day = new Date(this.current.getFullYear(), this.current.getMonth(), first + i);
        let dayObject = {
            date: day,
            weekday: day.getDay() === 0 ? 7 : day.getDay(),
            weekNumber: this.getWeekNumber(day)[0]
        };
        week.push(dayObject);
    }
    return week;
},
```

Рисунок 10 – Функція getWeek()

Функція changeMonth() змінює місяць поточної дати на заданий місяць. Якщо перший день нового місяця припадає на вихідний (неділя або субота), дата коригується на найближчий робочий день (рис. 11). Після зміни місяця викликається функція updateCalendar() для оновлення календаря.

```
changeMonth(monthIndex) {
    this.current = new Date(this.current.getFullYear(), monthIndex, 1);
    if (this.current.getDay() === 0 || this.current.getDay() === 6) {
        this.current.setDate(this.current.getDate() + 7);
    }
    this.updateCalendar();
},
```

Рисунок 11 – Функція changeMonth()

Функція changeWeek() змінює поточний тиждень, додаючи або віднімаючи задану кількість тижнів від поточної дати. Після зміни дати оновлюються параметри тижня, тип тижня, назва місяця та рік (рис. 12).

```
changeWeek(weeks) {
    let newDate = new Date(this.current.setDate(this.current.getDate() + weeks * 7));
    this.current = newDate;
    this.week = this.getWeek();
    this.typeOfWeek = this.getTypeOfWeek();
    this.month = this.getMonthName();
    this.year = this.getYear();

},
```

Рисунок 12 – Функція changeWeek()

Таким чином, розглянуто основні функції для побудови календаря, які забезпечують динамічне відображення та управління поточним тижнем, місяцем та роком. Цей календар створює візуально привабливий та інтуїтивно зрозумілий інтерфейс (див. рис. 13).

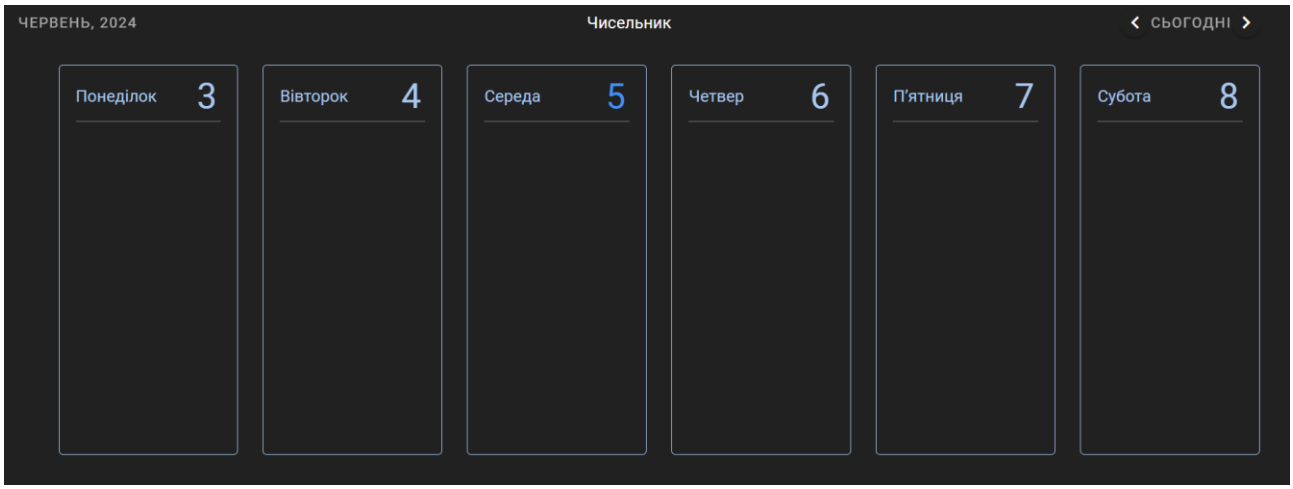


Рисунок 13 – Інтерфейс календаря

3.3.2 Розташування занять у календарі згідно з розкладом

Розташування занять у календарі включає відображення кожного часового інтервалу для занять протягом дня, дозволяючи користувачам переглядати та редагувати інформацію про заняття.

Для початку, компонент приймає відфільтрований за датою та типом тижня, а потім відсортований за часом, масив занять. Основний компонент розподіляє заняття по днях тижня. Кожен день тижня відображається у відповідній колонці, і всередині кожного дня відображаються заняття у визначені часові слоти (рис. 14).

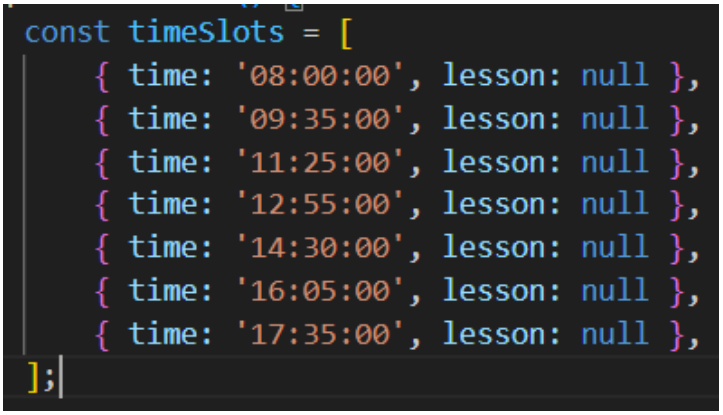


Рисунок 14 – Часові слоти

Якщо заняття знайдені у певному слоті, вони відображаються у вигляді кнопок з інформацією про заняття, включаючи назву уроку та час його проведення. Якщо ж ні – слот залишається пустим, займаючи місце у картці дня (див. рис. 15).

Студенти можуть бачити лише інформацію про заняття, натиснувши на нього. Адміністратори ж мають можливість ще і додавати нові заняття, використовуючи модальне вікно, що відкривається при натисканні на іконку додавання, та редагувати заняття, використовуючи модальне вікно, що відкривається при натисканні на іконку .

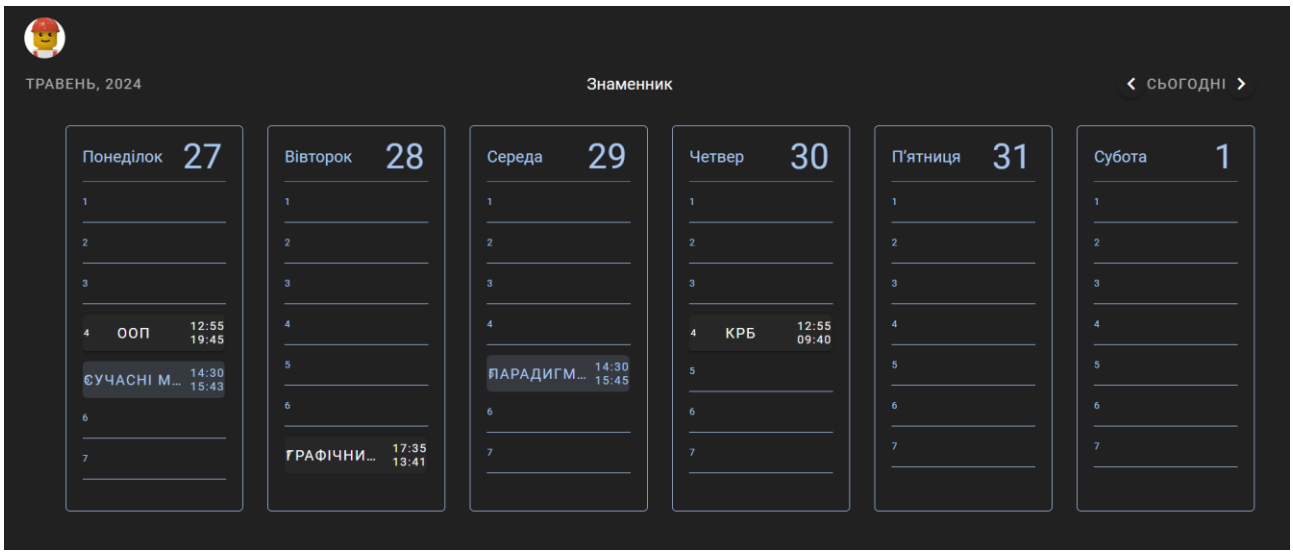


Рисунок 15 – Розташування занять у розкладі

3.3.3 Модальні вікна

Модальні вікна розроблені за допомогою компоненту Vuetify «v-dialog». Вони дозволяють переглядати, редагувати та видаляти заняття, не переходячи на окрему сторінку. Модальні вікна розроблені таким чином, що фон позаду них стає темнішим, забезпечуючи кращу фокусованість користувача на вмісті вікна. Інформація модальним вікнам передається через пропси, що дозволяє передавати дані про конкретне заняття до модального вікна для його відображення та редагування.

Відкриття та закриття модальних вікон реалізовано через керування станом компоненту «v-dialog». При натисканні на кнопку, що викликає модальне вікно, відповідний метод встановлює змінну «dialog» у значення «true», що відкриває модальне вікно. Закриття вікна відбувається або при натисканні на кнопку закриття, або при натисканні клавіші «Escape».

Таким чином, розробка інтерфейсу включає розподіл занять по днях тижня та часових слотах, а також використання модальних вікон для перегляду (рис. 16) та редагування інформації про заняття (рис. 17). Це забезпечує користувачам зручний та інтуїтивно зрозумілий спосіб взаємодії з календарем занять

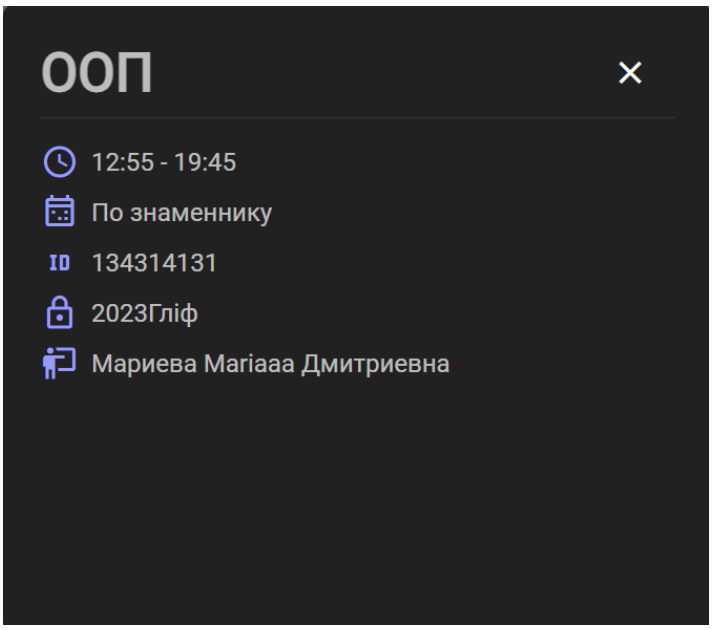
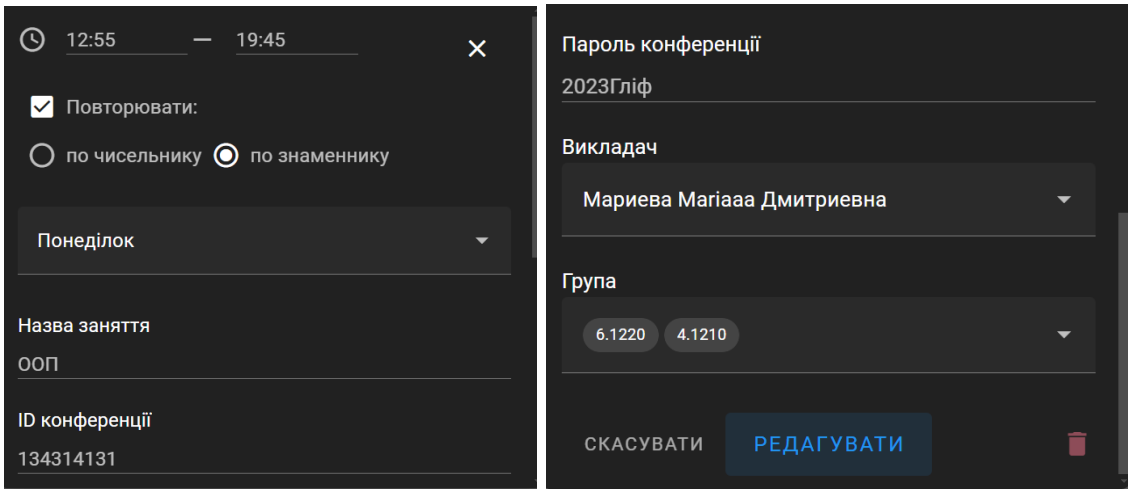


Рисунок 16 – Модальне вікно перегляду інформації



а) початок модального вікна

б) кінець модального вікна

Рисунок 17 – Модальне вікно редагування інформації

3.4 Розробка серверної частини

Розробка серверної частини є ключовою складовою будь-якого додатку, оскільки вона забезпечує обробку запитів від клієнтської частини, взаємодію з базою даних та виконання бізнес-логіки додатку.

3.4.1 Моделі, контролери та міграції

Для взаємодії з базою даних використовуються моделі, контролери та міграції.

Моделі Laravel забезпечують зручний доступ до таблиць бази даних та роботу з ними. Кожна таблиця має свою модель. Крім отримання записів з таблиці бази даних, моделі Eloquent також дозволяють вставляти, оновлювати і видаляти записи з таблиці [7].

Контролери спрощують потік даних між моделями та представленням, а саме обробляють вхідні HTTP-запити, визначають логіку обробки запитів, взаємодіють з моделями для отримання та збереження даних та відправляють відповіді клієнту.

Міграції використовуються для створення та оновлення структури бази даних.

3.4.2 Алгоритм взаємодії клієнтської частини із серверною

Взаємодія між клієнтською та серверною частиною додатку здійснюється за допомогою REST API (Representational State Transfer Application Programming Interface). REST API використовується для передачі даних та виконання операцій з базою даних між клієнтом і сервером за допомогою HTTP-запитів. В основі REST лежить концепція ресурсів, які можуть бути отримані, модифіковані або видалені через стандартні HTTP-методи, такі як get, post, put і delete. На рисунку 18 наведено принцип взаємодії клієнта з сервером за допомогою API.

У випадку побудови модуля розкладу занять, REST API використовується для отримання, додавання чи зміни даних в базі даних.

Рисунок 18 – Принцип роботи API запитів

З клієнтської сторони відправляється HTTP-запит до сервера за допомогою JavaScript бібліотеки Axios. В залежності від типу запиту сервер або повертає, або видаляє, або модифікує дані. На рис. 19 наведено приклад запиту отримання даних.

```
getLessonsByDate() {
    const token = localStorage.getItem('token');
    axios.get('/api/sorted-lessons', {
        headers: {
         'Authorization': `Bearer ${token}`
        }
    })
        .then(response => {
            this.lessonsByDate = response.data;
            console.log(this.lessonsByDate);
        })
        .catch(error => {
            console.error(error);
        });

},
```

Рисунок 19 – REST API запит

Цей запит отримує необхідні дані з сервера. Оскільки цей шлях захищений, для отримання даних потрібна додаткова перевірка авторизованості користувача. Запит використовує токен, збережений в локальному сховищі, для авторизації.

Після відправки запиту, він надходить до контролера, де виконується відповідна функція. В даному випадку, виконується функція повернення відсортованого масиву занять з бази даних (рис. 20).

```php
public function getSortedLessons() {
    $id = Auth::user()->id;
    $user = User::find($id);

    $lessons = Lesson::where(function ($query) use ($user) {
        if ($user->role == 'admin') {
            $query->where('teacher', $user->id);
        } elseif ($user->role == 'student') {
        $query->whereHas('groupLesson', function ($query) use ($user) {
            $query->where('group_name', $user->group_name);
        });
    }
    })->get();

    $lessonsByDate = $lessons->groupBy(function ($lesson) {
        if ($lesson->type_of_week && $lesson->weekday) {
            return $lesson->type_of_week . '-' . $lesson->weekday;
        } else {
            // Якщо немає type_of_week або weekday, використовувати lesson_date
            return $lesson->lesson_date;
        }
    })->map(function ($lessonsOnSameDay) {
        return $lessonsOnSameDay->sortBy('start_time')->values();
    });

    return response()->json($lessonsByDate);
}
```

Рисунок 20 – Функція отримання масиву занять

Ця функція спочатку отримує ідентифікатор поточного користувача та знаходить відповідний об’єкт користувача. Потім вона перевіряє його роль: адміністратор чи студент. Якщо користувач адміністратор, то функція знаходить всі заняття, в яких id викладача збігається з id поточного користувача. Якщо ж студент – функція перевіряє групу, до якої належить студент, та знаходить всі заняття для цієї групи. Заняття групуються за датою або типом тижня та

сортуються за часом початку. Нарешті, вона повертає відсортований масив занять у форматі JSON.

У разі успішного виконання функції, дані у форматі JSON повертаються назад у компонент Vue.js та записуються в необхідну змінну.

Таким чином побудовані всі запити від клієнтської частини до серверної.

3.4.3 Використання моделей і контролерів для взаємодії з базою даних

Використання моделей та контролерів Laravel є важливим етапом у розробці додатків, оскільки вони забезпечують обробку вхідних HTTP-запитів, взаємодію з моделями для отримання та збереження даних, а також відправку відповідей клієнту. Зокрема, розглянемо, як використовується модель GroupLesson та контролер GroupLessonController для отримання даних про розподіл занять по групах з бази даних.

Таблиці «Group» «Lesson» мають зв’язок «багато-до-багатьох», що означає, що одне заняття може проводитись у багатьох груп, так само як і одна група може мати багато занять. Тому, для зв’язку між цими таблицями, було створено ще одну таблицю та, відповідно, модель (рис. 21).

Метод lesson() визначає відношення «багато до одного» до моделі Lesson. Це означає, що кожен запис в GroupLesson може бути пов’язаний з одним записом в Lesson через lesson_id.

Аналогічно, метод group() визначає відношення «багато до одного» до моделі Group. Кожен запис в GroupLesson може бути пов’язаний з одним записом в Group через group_name.

У контролері GroupLessonController (див. рис. 22) метод getGroupName($id) використовує модель GroupLesson для отримання імені групи для даного lesson_id. Він використовує метод where для фільтрації записів GroupLesson за lesson_id, а потім використовує метод pluck для отримання значення group_name для цих записів. Результат повертається як JSON-відповідь.

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class GroupLesson extends Model
{
    use HasFactory;
    protected $table = 'group_lesson';

    public $timestamps = false;

    protected $fillable = [
        'lesson_id',
        'group_name',
    ];

    public function lesson() {
        return $this->belongsTo(Lesson::class, 'lesson_id');
    }

    public function group() {
        return $this->belongsTo(Group::class, 'group_name', 'group_name');
    }
}
```

Рисунок 21 – Модель GroupLesson

```php
class GroupLessonController extends Controller
{
    public function getGroupName($id) {
        $group = GroupLesson::where('lesson_id', $id)->pluck('group_name');
        return response()->json($group);
    }
```

Рисунок 22– Контролер GroupLessonController

Таким чином, модель GroupLesson та контролер GroupLessonController разом дозволяють взаємодіяти з таблицею group_lesson в базі даних, отримувати та зберігати інформацію про заняття та групи. Завдяки цьому, можна легко отримати інформацію про групу для конкретного заняття та відобразити цю інформацію в календарі.

ВИСНОВКИ

У дослідженні було розглянуто та розроблено інформаційно-освітню платформу студента, а саме модуль розкладу занять, що базується на клієнтській та серверній частинах. За допомогою аналізу функціональних вимог користувачів та використання сучасних технологій, було реалізовано зручний та ефективний інтерфейс, який надає можливість студентам та викладачам керувати розкладом занять та отримувати необхідну інформацію.

Завдяки розробленій платформі, користувачі можуть ефективно організовувати свій навчальний та робочий час, отримуючи доступ до розкладу занять у зручному форматі.

Отже, розроблена інформаційно-освітня платформа стане корисним інструментом для студентів та викладачів навчальних закладів, сприяючи покращенню організації навчального процесу та підвищенню ефективності управління розкладом занять.

ПЕРЕЛІК ПОСИЛАНЬ

1. Цифрові технології в освіті: сучасний досвід, проблеми та перспективи : монографія / Т. А. Васильєва та ін. ; за заг. ред. д-рки екон. наук, проф. Т. А. Васильєвої, д-ра екон. наук, проф. Ю. М. Петрушенка. Суми : Сумський державний університет, 2022. 150 с.

2. UML в Інтернеті. Aspose Products. URL : https://products.aspose.app/diagram/uk/uml (дата звернення: 10.01.2024).

3. MVC. Розробка вебсайтів Technologies. Brander. URL : https://brander.ua/technologies/mvc (дата звернення: 11.01.2024).

4. 12 головних переваг вибору Laravel Framework для вашого сайту. Cases media. URL : https://cases.media/en/article/12-golovnikh-perevag-viboru-laravel-framework-dlya-vashogo-saitu (дата звернення: 11.01.2024).

5. Що таке Composer і чому він використовується в Laravel? CloudDevs. URL : https://clouddevs.com/laravel/composer/ (дата звернення: 11.01.2024).

6. Installation. Laravel. URL : https://laravel.com/docs/11.x (дата звернення: 11.01.2024).

7. Eloquent. Eloquent: Getting Started. Laravel. URL : https://laravel.com/docs/11.x/eloquent (дата звернення: 15.01.2024).

8. Getting Started with MySQL. MySQL. URL : https://dev.mysql.com/doc/mysql-getting-started/en/ (дата звернення: 05.02.2024).

9. John Au-Yeung. Vue.js 3 By Example. Birmingham : Packt Publishing Ltd, 2021. 320 с.

10. Vue.js – навіщо він нам? Medium. URL : https://medium.com/@jstify.community/vue-js-%D0%BD%D0%B0%D0%B2%D1%96%D1%89%D0%BE-%D0%B2%D1%96%D0%BD-%D0%BD%D0%B0%D0%BC-981b5e17af39