

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

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «магістр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Навчальна платформа для людей
з обмеженими можливостями»**

Виконала студентка групи ЗІПЗм-22-1
КАРАПЕТ Людмила Василівна

Керівник роботи:
САВІЦЬКИЙ Роман Святославович

Рецензент:
ЄФРЕМОВ Юрій Миколайович

Житомир – 2023

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

«ЗАТВЕРДЖУЮ»

зав. кафедри інженерії
програмного забезпечення

Тетяна ВАКАЛЮК

«18» вересня 2023 р.

ЗАВДАННЯ

на кваліфікаційну роботу

Здобувач вищої освіти: **КАРАПЕТ Людмила Василівна**

Керівник роботи: **САВІЦЬКИЙ Роман Святославович**

Тема роботи: **«Навчальна платформа для людей з обмеженими
можливостями»**,

затверджена Наказом закладу вищої освіти від **«18» вересня 2023 р.**
№ 481/с

Вихідні дані для роботи: розроблена платформа для навчання, яка відповідає високим стандартам вебдоступності, що забезпечує ефективне та комфортне навчання для всіх користувачів, надаючи можливості навчання та організації навчального процесу.

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Савіцький Р.С	22.09.2023	22.09.2023
2	Савіцький Р.С	22.09.2023	22.09.2023
3	Савіцький Р.С	22.09.2023	22.09.2023

РЕФЕРАТ

Магістерська робота, що подана для розгляду, присвячена дослідженню теми «Навчальної платформи для людей з обмеженими можливостями». Робота охоплює вступ, три основні розділи, висновки, перелік використаних джерел із 33 назвою, 26 графічних ілюстрацій та 7 лістингів. Загальний обсяг дисертації складає 93 сторінок, з яких 86 відводиться для основного тексту.

Мета розробки платформи для навчання з обмеженими можливостями полягає в створенні інклюзивного освітнього середовища, яке враховує і задовольняє потреби учнів та вчителів із різними видами обмежених можливостей. Акцентуючи увагу на вчителів, ця платформа спрямована на створення легкозасвоюваних інструментів, ресурсів та методик для ефективної взаємодії вчителів, не обтяжуючих їх особливостями.

Розроблена платформа для навчання осіб з обмеженими можливостями має інтуїтивно зрозумілий інтерфейс, який відповідає основним принципам вебдоступності. У платформі успішно інтегровано щоденник з оцінками учнів, який дозволяє вчителям та батькам миттєво відстежувати академічний прогрес. Крім того, наявний календар, де вчителі можуть легко створювати та планувати уроки, а також зручне поле з уроками за певний місяць сприяє ефективній організації навчального процесу.

Для забезпечення високого рівня доступності платформи був проведений ретельний аналіз різноманітних алгоритмів та методологій, що використовуються у сучасній практиці вебдоступності. Для кращого розуміння того, як поліпшити доступність платформи, детально розглядались проблеми вебдоступності в навчальних платформах. Здійснено аналіз архітектур, функціональності та алгоритмів навчальних платформ з акцентом на покращення взаємодії з користувачами різних потреб. Досліджуються ефективні практики, які можуть служити для оптимізації власної освітньої платформи. Використовуючи стандарти WCAG, які визначають критерії доступності

вебконтенту, створено компоненти платформи, спрямовані на забезпечення високого рівня доступності.

Цю платформу розроблено з використанням React, що гарантує високий рівень інтерактивності та сприяє утворенню ефективного та зручного користувацького інтерфейсу. Використання Remix дозволяє створювати високопродуктивні односторінкові додатки з складними маршрутами. TypeScript забезпечує типізацію коду, що підвищує надійність та зручність розробки. MongoDB використовується для ефективного зберігання та управління даними з використанням гнучкої NoSQL бази даних. Впровадження Prisma спрощує взаємодію та оптимізує запити до бази даних.

Ключові слова: платформа для навчання, вебдоступність, WCAG, accessibility, Tailwind, Remix, Prisma, Mongodb, React, TypeScript.

ABSTRACT

The master's thesis submitted for consideration is devoted to the study of the topic «Educational platform for people with disabilities.» The work includes an introduction, three main chapters, conclusions, a list of used sources with 33 titles, 26 graphic illustrations and 7 listings. The total volume of the dissertation is 93 pages, of which 86 are reserved for the main text.

The purpose of developing a platform for learning with disabilities is to create an inclusive educational environment that takes into account and meets the needs of students and teachers with different types of disabilities. Focusing on teachers, this platform aims to create easy-to-use tools, resources and techniques for effective teacher interaction without overwhelming them.

The developed platform for learning people with disabilities has an intuitive interface that meets the basic principles of web accessibility. The platform has successfully integrated a student grade diary that allows teachers and parents to instantly track academic progress. In addition, the available calendar, where teachers can easily create and plan lessons, as well as a convenient field with lessons for a certain month, contributes to the effective organization of the educational process.

To ensure a high level of accessibility of the platform, a thorough analysis of various algorithms and methodologies used in modern web accessibility practice was carried out. For a better understanding of how to improve the accessibility of the platform, the problems of web accessibility in educational platforms were considered in detail. An analysis of architectures, functionality and algorithms of educational platforms was carried out with an emphasis on improving interaction with users of various needs. Effective practices that can serve to optimize one's own educational platform are being studied. Using WCAG standards, which define accessibility criteria for web content, platform components aimed at ensuring a high level of accessibility have been created.

This platform is developed using React, which guarantees a high level of interactivity and contributes to the creation of an effective and convenient user

interface. Using Remix allows you to create high-performance single-page applications with complex routing. TypeScript provides code typing, which increases the reliability and convenience of development. MongoDB is used for efficient data storage and management using a flexible NoSQL database. Prisma implementation simplifies interaction and optimizes database queries.

Keywords: learning platform, web accessibility, WCAG, accessibility, Tailwind, Remix, Prisma, MongoDB, React, TypeScript

ЗМІСТ

РЕФЕРАТ.....	2
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	8
ВСТУП.....	9
РОЗДІЛ 1. РОЗГЛЯД ТЕОРЕТИЧНИХ ОСНОВ ТА АНАЛІЗУ ДЛЯ СТВОРЕННЯ НАВЧАЛЬНОЇ ПЛАТФОРМИ ДЛЯ ЛЮДЕЙ З ОБМЕЖЕНИМИ МОЖЛИВОСТЯМИ.....	13
1.1 Аналіз алгоритмів та методології в області вебдоступності.....	13
1.2 Розгляд проблем доступності в навчальних платформах та пошук ефективних рішень.....	15
1.3 Аналоги з високою вебдоступністю в освітній сфері.....	18
Висновки до розділу 1.....	24
РОЗДІЛ 2. АНАЛІЗ, ПРОЕКТУВАННЯ ТА ТЕХНОЛОГІЇ ДЛЯ ДОСТУПНОЇ ОСВІТНЬОЇ ПЛАТФОРМИ.....	25
2.1. Аналіз існуючих підходів до платформ для навчання людей з обмеженою доступністю.....	25
2.2. Визначення вимог та потреб цільової аудиторії.....	26
2.3. Огляд сучасних технологій для розробки доступних освітніх рішень.....	28
2.4. Проектування та архітектура рішення.....	30
Висновки до розділу 2.....	40
РОЗДІЛ 3. РОЗРОБКА МАРШРУТІВ, АПРОБАЦІЯ ТА ПІДТВЕРДЖЕННЯ НАУКОВОЇ НОВИЗНИ МАРШРУТІВ.....	41
3.1 Реалізація маршрутів та компонентів.....	41
3.1.1 Розробка маршруту входу, реєстрації та виходу для режимів вчителя та учня.....	41
3.1.2 Реалізація маршруту home.....	51

3.1.3 Розробка та впровадження Щоденника.....	55
3.1.4 Реалізація та оптимізація Календаря.....	67
3.1.5 Розділ уроки: Розробка та Впровадження.....	76
3.2 Апробація та підтвердження наукової новизни навчальної платформи для людей з обмеженими можливостями.....	81
3.2.1 Оцінка за Lighthouse показнику доступності.....	81
3.2.2 Збільшення конверсії з урахуванням вебдоступності.....	83
Висновки до розділу 3.....	85
ВИСНОВКИ.....	87
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	89

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

W3C – World Wide Web Consortium

WCAG – Web Content Accessibility Guidelines

CSS – Cascading Style Sheets

DB – Database

ORM – Object-Relational Mapping

UI – User Interface

HTML – HyperText Markup Language

ВСТУП

В сучасному інформаційному суспільстві, навчання та доступ до знань є ключовими факторами для особистісного та професійного розвитку. Однак, не всі користувачі мають рівні можливості для отримання якісної освіти, зокрема люди з обмеженими можливостями зазнають значних труднощів при користуванні традиційними освітніми платформами. Це обумовлено не лише фізичними обмеженнями, але і відсутністю адаптованого навчального контенту та інструментів для взаємодії.

Актуальність теми розробки платформи для навчання для людей з обмеженими можливостями обумовлена нагальною потребою в суспільстві створити рівні можливості для всіх громадян, незалежно від їхніх фізичних або когнітивних можливостей. Люди з обмеженими можливостями стикаються з викликами та бар'єрами при отриманні якісної освіти через відсутність адаптованих та доступних засобів навчання.

Створення платформи для навчання, орієнтованої на інклюзію, сприятиме створенню безбар'єрного освітнього середовища та допоможе розвивати потенціал кожної особи, незалежно від її фізичних або когнітивних характеристик. Такий підхід відповідає сучасним вимогам суспільства до рівних можливостей та інклюзивного розвитку.

Метою даної магістерської роботи є розробка та впровадження платформи для навчання, спрямованої на покращення методів та засобів взаємодії з користувачами з обмеженими можливостями. Проект має на меті забезпечення високого рівня зручності та доступності для користувачів з обмеженими можливостями, а також надання всіх необхідних функціональних можливостей для ефективного навчання та розвитку навичок.

Розробка платформи для навчання з обмеженими можливостями передбачає вирішення конкретних завдань:

1. Аналіз існуючих платформ для навчання: провести докладний аналіз наявних освітніх платформ та сервісів, оцінити їхню відповідність потребам осіб з обмеженими можливостями. Визначити переваги та недоліки існуючих рішень.

2. Визначення потреб користувачів: здійснити ретельне вивчення потреб та вимог користувачів з обмеженими можливостями у сфері навчання. Визначити основні аспекти, які слід враховувати при розробці інклюзивної освітньої платформи.

3. Проектування інклюзивного інтерфейсу: розробити дизайн інтерфейсу, що враховує особливості користувачів з різними видами обмежень. Забезпечити можливості адаптації та персоналізації інтерфейсу.

4. Розробка функціоналу: створити освітній контент та реалізувати можливості ведення щоденника та розкладу для користувачів. Розробити інтерактивні уроки та завдання, орієнтовані на різні типи обмежень.

5. Технічна імплементація: вибрати та налаштувати технології для реалізації платформи. Розробити серверну та клієнтську частини, враховуючи вимоги до доступності.

6. Проаналізувати покращення показників вебресурсу після впровадження методів вебдоступності.

Предметом дослідження виступають методи та платформи для навчання людей з обмеженими можливостями.

Об'єкт дослідження: освітнє середовище для людей з обмеженими можливостями та розроблена платформа для їхнього навчання.

Наукова новизна дипломної роботи полягає в розробці та імплементації інклюзивної освітньої платформи, яка спеціально адаптована для користувачів з обмеженими можливостями. Основним внеском у науку є створення зручного та доступного інтерфейсу, а також реалізація важливого функціоналу для ефективного навчання цільової аудиторії.

В ході дослідження було проведено глибокий аналіз впливу впровадження методів вебдоступності на покращення показників вебресурсу. Основним об'єктом дослідження стало вивчення впливу інтеграції інклюзивних підходів та технічних змін на функціональність та доступність вебсайту для різних категорій користувачів.

Розглядалися різні аспекти, спрямовані на оптимізацію взаємодії з вебсайтом для різних категорій користувачів. Вдосконалення структури інформації для полегшення орієнтації користувачів, використовуючи чіткі заголовки, списки та логічну організацію контенту. Застосування семантичних тегів покращило розуміння контенту для допоміжних технологій та забезпечило більшу доступність. Також звернула увагу на кольорову доступність, забезпечивши достатню контрастність та враховуючи візуальні обмеження користувачів. Реалізація клавіатурної навігації та адаптивного дизайну полегшила використання сайту для різних користувачів. Також враховано роль, ім'я та значення компонентів для ефективного використання допоміжними технологіями. Врахування візуального фокусу та лейбли покращили доступність для осіб із різними обмеженнями. Можливість автозаповнення форм спростила взаємодію з платформою. Ці зміни спрямовані на створення більш ефективного та доступного вебсайту для всіх користувачів, незалежно від їхніх можливостей.

Особливий акцент зроблено на створенні інтуїтивно зрозумілого та ергономічного інтерфейсу, який забезпечує не лише доступність, а й комфортність для користувачів з обмеженими можливостями. Важливим досягненням є впровадження ряду освітніх інструментів та функцій, спрямованих на підвищення ефективності процесу навчання цільової аудиторії. Такий підхід відкриває нові можливості для інклюзивної освіти та сприяє розвитку сучасного освітнього середовища, де кожен учень може отримати якісні знання, незалежно від їхніх фізичних чи психічних можливостей.

Практична цінність реалізованої платформи виявляється у забезпеченні ефективного відстеження академічного прогресу, спрощенні процесу взаємодії між учнями та вчителями, а також створенні зручного середовища для здобуття освіти особами з обмеженими можливостями. Інтегрований функціонал, такий як щоденник та розклад, виправдовує практичне використання платформи, забезпечуючи студентам та викладачам ефективний інструмент для навчання та викладання. Реалізація доступних форм входу та реєстрації, а також можливість легкої зміни режимів користувачів, враховує потреби різних груп користувачів, забезпечуючи широкий спектр можливостей для навчання та взаємодії усіх учасників освітнього процесу. Такий практичний підхід допомагає створювати відкрите та інклюзивне освітнє середовище, відповідаючи сучасним стандартам рівних можливостей та інноваційному підходу до навчання.

Публікації. За темою атестаційної магістерської роботи було опубліковано тези [1].

РОЗДІЛ 1. РОЗГЛЯД ТЕОРЕТИЧНИХ ОСНОВ ТА АНАЛІЗУ ДЛЯ СТВОРЕННЯ НАВЧАЛЬНОЇ ПЛАТФОРМИ ДЛЯ ЛЮДЕЙ З ОБМЕЖЕНИМИ МОЖЛИВОСТЯМИ

1.1 Аналіз алгоритмів та методології в області вебдоступності.

Вебдоступність є важливою складовою сучасного інтернет-простору, оскільки вона спрямована на забезпечення рівних можливостей та доступу до інформації для всіх користувачів, незалежно від їхніх фізичних або когнітивних можливостей. Цей розділ висвітлить ключові алгоритми та методи у сфері вебдоступності, що визначають напрямки досліджень та інновацій у цьому важливому напрямку інформаційних технологій [1].

Науковий підхід до вебдоступності передбачає дослідження різних аспектів, що включають технічні, психологічні та соціальні виміри. Дослідження взаємодії користувача з вебресурсами враховує особливості різних груп користувачів, зокрема тих, у кого є візуальні, слухові, моторні обмеження. Технологічні інновації, такі як розробка адаптивних інтерфейсів та використання штучного інтелекту, нейронних мереж, також стають об'єктом наукових досліджень [2].

Ключові стандарти та рекомендації. Однією з основних наукових основ вебдоступності є розробка та удосконалення стандартів. Наприклад, вироблені консорціумом W3C, Web Content Accessibility Guidelines (WCAG) визначають принципи та критерії, які ресурс повинен виконувати для того, щоб вважатися доступним. Наукове співтовариство активно досліджує ефективність цих стандартів і пропонує внески для їхнього постійного вдосконалення [3].

Науковці вивчають можливості розробки технологій, які можуть автоматично адаптуватися до різних потреб користувачів. Це включає в себе розробку адаптивних інтерфейсів, технічних рішень для покращення навігації та сприяння взаємодії з вебресурсами [4].

Наукові дослідження охоплюють і педагогічні аспекти, включаючи розробку ефективних освітніх програм для підготовки фахівців у галузі вебдоступності. Дослідження педагогічних стратегій дозволяють підвищити усвідомленість серед веброзробників та дизайнерів стосовно важливості вебдоступності [3].

Наукове співтовариство активно працює над розробкою і вдосконаленням методів тестування та оцінки рівня вебдоступності. Це включає в себе автоматизовані засоби та об'єктивні критерії для визначення ефективності ресурсів у відповідності до стандартів [2].

В області вебдоступності існує ряд ключових алгоритмів та підходів, спрямованих на створення вебсередовища, яке б було доступним для широкого кола користувачів.

Accessible Rich Internet Applications (ARIA): цей алгоритм надає розширені атрибути для HTML-елементів, що полегшує взаємодію з вебдодатками для осіб із різними обмеженнями. ARIA дозволяє розробникам надавати додаткову інформацію та контекст для користувачів, які використовують програми читання екрану [5].

Web Content Accessibility Guidelines (WCAG): WCAG визначає критерії доступності для вебконтенту і включає в себе рекомендації для створення бар'єропроглядних інтерфейсів. Заснований на принципах сприйняття, розуміння, навігації та взаємодії, WCAG встановлює стандарти, які сприяють створенню вебсайтів та додатків, що враховують різні потреби користувачів.

Screen Readers: використання програмних засобів для читання екрану (screen readers) є важливим компонентом вебдоступності. Ці інструменти конвертують текстовий контент вимовленою або тактильною інформацією, забезпечуючи доступ осіб із візуальними обмеженнями [6].

Навігація за допомогою клавіатури: забезпечення можливості навігації та взаємодії з вебресурсами за допомогою клавіатури є важливим для тих, хто не може використовувати мишу. Врахування адаптивного та responsive дизайну

дозволяє оптимально відображати та взаємодіяти з контентом на різних пристроях [7].

Ці алгоритми та підходи взаємодіють та співпрацюють для створення доступних та вдосконалених вебсередовищ, враховуючи різноманітні потреби користувачів. Постійне дослідження та впровадження нових технологій і методів тестування допомагають розвивати і вдосконалювати стандарти вебдоступності.

1.2 Розгляд проблем доступності в навчальних платформах та пошук ефективних рішень

Навчальні платформи, у сучасному освітньому ландшафті, виконують не тільки функцію інструменту для отримання знань, але й визначають новий стандарт освіти. Зростання цифрового впливу у навчанні відкриває нові можливості для глобального доступу до знань, але одночасно породжує значні виклики у сфері доступності.

Проблеми доступності виникають для різних груп користувачів, включаючи осіб із візуальними, слуховими, моторними та когнітивними обмеженнями. Наприклад, користувачі з візуальними обмеженнями можуть зазнавати труднощів у взаємодії з інтерфейсами та контентом, тоді як особи із слуховими обмеженнями можуть відчувати нестачу доступу до аудіо- та відео-матеріалів.

У цьому контексті виникає необхідність в ретельному вивченні основних проблем доступності та впровадженні ефективних рішень для забезпечення усім користувачам рівних можливостей у доступі до освіти через вебплатформи. Важливо розглядати не лише технічні аспекти, але і враховувати різноманітні потреби різних груп користувачів для створення дійсно інклюзивного освітнього середовища [8].

Розглянемо проблеми доступності в навчальних платформах та їхні розв'язання:

Візуальна доступність

Проблема: користувачі з візуальними обмеженнями можуть стикатися із труднощами взаємодії з вмістом та інтерфейсами. Недостатня контрастність, відсутність текстових описів та неадаптовані графічні елементи ускладнюють сприйняття [9].

Рішення:

- розробка гнучких інтерфейсів, які можна легко адаптувати до різних розмірів та типів екранів, з урахуванням вимог високої контрастності;
- впровадження технологій, які дозволяють користувачам самостійно налаштовувати кольорову гаму та розмір шрифту.

Аудіальна доступність

Проблема: слухові обмеження можуть ускладнювати сприйняття аудіо-матеріалів. Відсутність текстових описів для аудіо- та відео-контенту обмежує доступ до інформації [9].

Рішення:

- забезпечення текстових транскрипцій та субтитрів для аудіо- та відео-матеріалів для полегшення сприйняття інформації;
- впровадження автоматичних систем генерації субтитрів для відео з можливістю ручної корекції.

Моторна доступність

Проблема: користувачі з обмеженою моторною функцією можуть відчувати труднощі у взаємодії з ресурсами через неадекватно адаптовані елементи керування та навігації [10].

Рішення:

- розробка інтерфейсів із врахуванням різних типів обмежень моторної функції та можливість вибору способу введення;

- використання технологій голосового керування для полегшення навігації та виконання функцій.

Когнітивна доступність

Проблема: деякі користувачі можуть зазнавати труднощі у засвоєнні інформації через складність мови, відсутність структуризації чи неправильне використання мультимедійних засобів [10].

Рішення:

- використання простої та зрозумілої мови для подачі інформації та створення структурованих, легко усвідомлюваних модулів навчання;
- розробка інтерактивних завдань та персоналізованих підходів для різних ступенів когнітивних можливостей користувачів.

Отже можемо зробити певний висновок: навчальні платформи в сучасному освітньому ландшафті стають не лише засобом отримання знань, але й керують новими стандартами освіти. Зростання цифрового впливу у навчанні відкриває широкі можливості для глобального доступу до знань, але одночасно постає великий виклик у сфері доступності.

Проблеми доступності у навчальних платформах торкаються різних груп користувачів, включаючи осіб із візуальними, слуховими, моторними та когнітивними обмеженнями. Це вимагає уважного вивчення основних проблем і впровадження ефективних рішень для забезпечення рівних можливостей у доступі до освіти через вебплатформи.

Зазначені різновиди проблем та їхні розв'язання виокремлюють технічні та концептуальні аспекти, які потребують уваги при розробці та вдосконаленні навчальних платформ. Важливо підходити до цього завдання комплексно, з урахуванням різноманітних потреб користувачів, щоб створити інклюзивне освітнє середовище для всіх.

1.3 Аналоги з високою вебдоступністю в освітній сфері

Coursera – це освітня платформа, яка спеціалізується на наданні доступу до онлайн-курсів від провідних університетів та організацій по всьому світу. Основна мета Coursera полягає в забезпеченні високоякісної освіти в будь-який час та в будь-якому місці через Інтернет.

Візуальна доступність:

Інтерфейс Coursera розроблений з урахуванням принципів візуальної доступності. Простий та інтуїтивно зрозумілий дизайн сприяє легкості навігації для користувачів з різними рівнями візуальних можливостей. Крім того, налаштування параметрів відображення дозволяє користувачам адаптувати інтерфейс під свої потреби [11].

Аудіальна доступність:

Кожен відеорок на Coursera супроводжується текстовим описом та субтитрами. Це робить зміст доступним для користувачів з будь-яким рівнем слухових можливостей, надаючи можливість отримати інформацію за допомогою читання тексту [11].

Моторна доступність:

Інтерфейс платформи Coursera адаптується для різних типів пристроїв та методів введення. Це дозволяє користувачам з обмеженою моторною активністю легко переміщатись та взаємодіяти з ресурсами, використовуючи доступні методи керування [11].

Когнітивна доступність:

Курси на Coursera створюються, враховуючи принципи чіткої структури та простої мови. Це сприяє полегшенню засвоєння інформації користувачами з різними рівнями когнітивних можливостей. Використання доступної мови та чіткої логіки робить матеріали зрозумілими та доступними для всіх [6]. На рисунку нижче зображено ресурс Coursera (рис. 1.1).

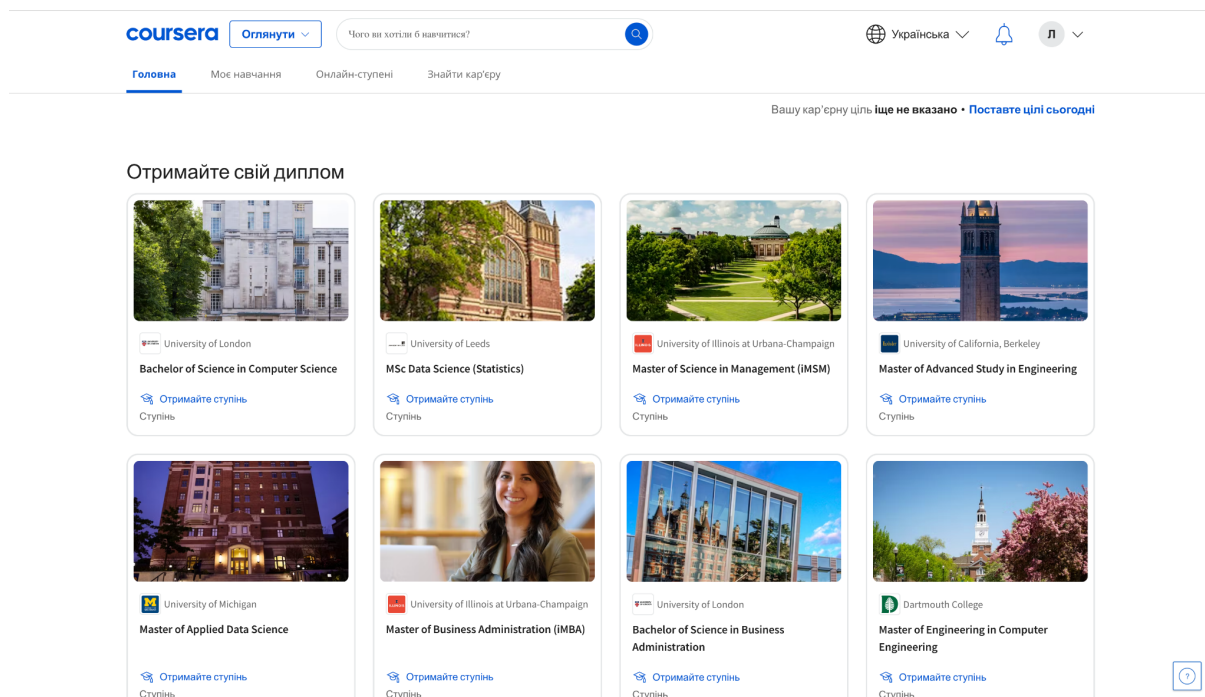


Рис. 1.1 Навчальна платформа Coursera

edX – це освітня платформа, яка об'єднує широкий спектр курсів від провідних університетів та організацій із різних країн світу. Розроблена з огляду на принципи доступності, *edX* використовує різноманітні стратегії для полегшення навчання для користувачів із різними обмеженнями [12].

Візуальна доступність: інтерфейс *edX* розроблений з урахуванням візуальної доступності. Користувачі можуть змінювати параметри відображення, такі як розмір шрифту, контрастність та інші, для того щоб краще пристосувати платформу до своїх потреб [12].

Аудіальна доступність: всі відеоуроки на *edX* супроводжуються субтитрами та текстовими описами. Це робить навчальний контент доступним для користувачів із слуховими відмінностями, дозволяючи отримати інформацію через читання тексту [12].

Моторна доступність: інтерфейси *edX* підтримують різні методи введення, включаючи жестове управління та голосове керування. Це полегшує взаємодію для користувачів із обмеженою моторною активністю [12].

Когнітивна доступність: матеріали на платформі структуровані таким чином, щоб полегшити засвоєння складної інформації. Використання простої мови та чіткої логіки сприяє розумінню матеріалів для користувачів із різними рівнями когнітивних можливостей [12]. Приклад логічної структуризації матеріалів зображено на рисунку нижче (рис. 1.2).

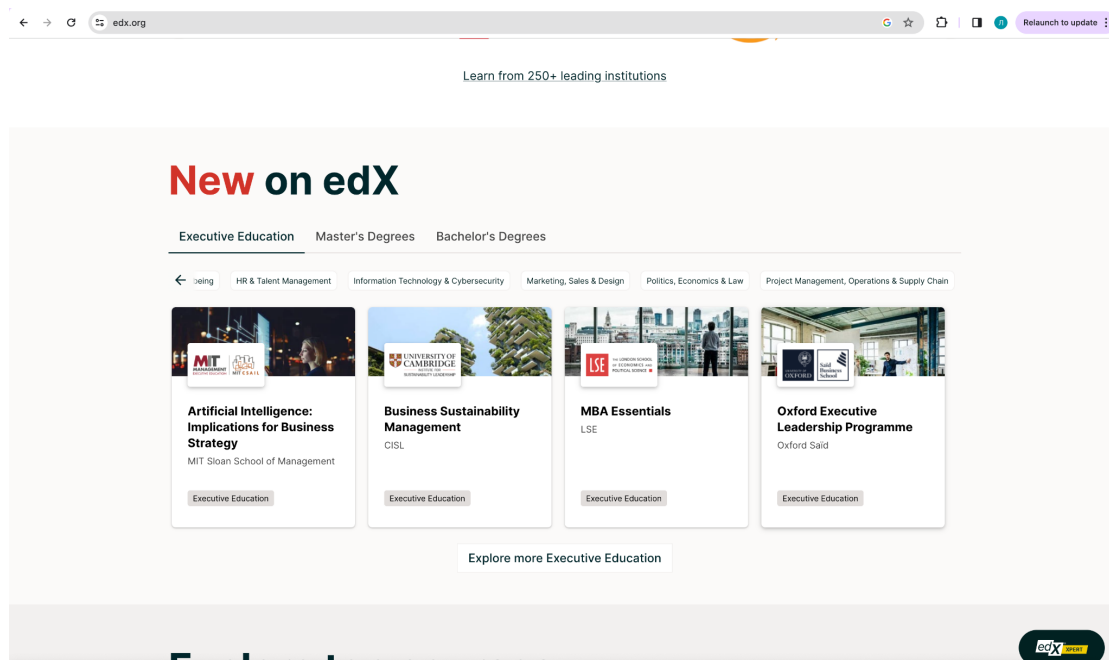


Рис. 1.2 Навчальна платформа edX

Google Classroom – це освітня платформа, яка спрямована на полегшення взаємодії між вчителями та учнями, забезпечуючи віртуальний простір для навчання та обміну ресурсами. Розроблена компанією Google, вона поєднує в собі інструменти для організації занять, створення завдань та зручного обміну інформацією [13].

Візуальна доступність:

Google Classroom приділяє увагу візуальній доступності через використання простого та ергономічного дизайну. Користувачі можуть налаштовувати параметри відображення, такі як темні та світлі теми, що сприяє комфортній роботі з платформою [13].

Аудіальна доступність:

Google Classroom дбає про аудіальну доступність, забезпечуючи можливість додавання аудіозаписів та голосових коментарів до завдань. Це полегшує взаємодію з контентом для користувачів із слуховими особливостями[13].

Моторна доступність:

Інтерфейс Google Classroom пристосований для різних методів введення, включаючи тачскрін-жести та голосове управління. Це дозволяє користувачам з різними рівнями моторних можливостей ефективно взаємодіяти з платформою[13].

Когнітивна доступність:

Google Classroom (рис. 1.3) пропонує інструменти для індивідуалізації та персоналізації навчання, що сприяє адаптації матеріалів до потреб різних груп користувачів з різними рівнями когнітивних можливостей [13].

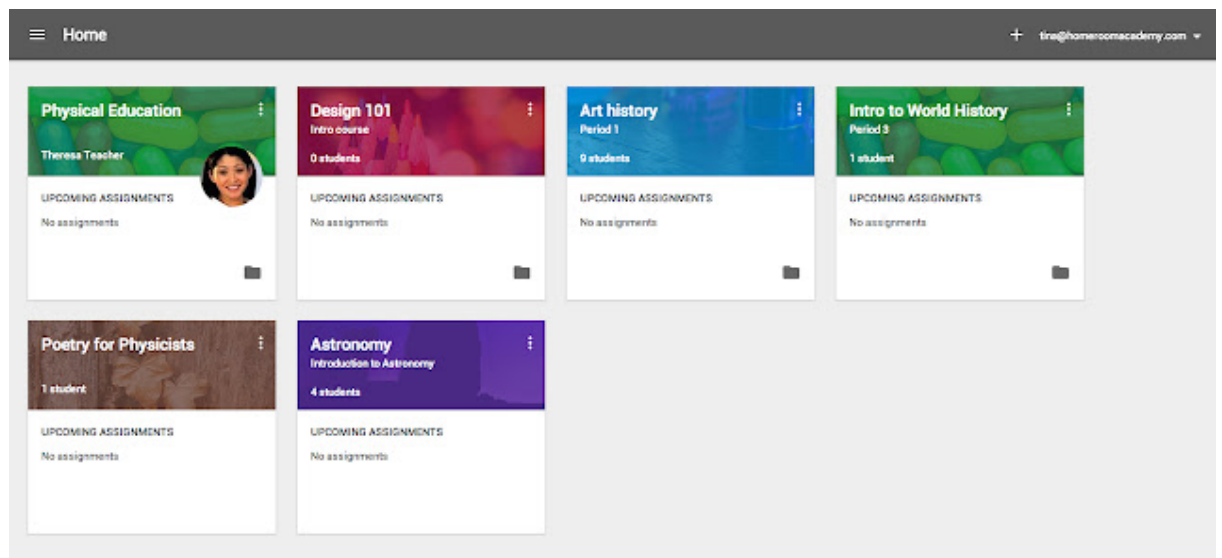


Рис. 1.3 Навчальна платформа Google Classroom

Освітня платформа EduForAll створена з урахуванням різноманітних потреб користувачів та спрямована на оптимізацію навчального процесу. Забезпечено інтеграцію ключових компонентів, таких як щоденник, календар,

розділ із уроками, форми входу та реєстрації, що утворює єдиний та лаконічний інструментарій.

У розділі «Щоденник» використовується зручний інструмент для ведення обліку успішності учнів. Можливість отримання списку учнів та додавання/редагування, також видалення оцінок робить його зручним для вчителів.

Календар, розроблений за допомогою бібліотеки FullCalendar, надає потужний інструмент для відображення та управління розкладом уроків. Він дозволяє додавати та видаляти уроки та забезпечує можливість перегляду розкладу за різними періодами такими як місяць, тиждень та день.

Розділ із уроками надає зручну можливість перегляду розкладу за визначеними параметрами та за визначений період часу. Застосована система сортування, інтерактивність та інтеграція з іншими компонентами роблять його корисним для вчителів та учнів.

Форми для входу та виходу обладнані кнопками, що дозволяють зручно перемикаєти режими між учителем та учнем. Вони також мають кнопки для швидкої зміни ролі користувача, забезпечуючи легку навігацію та зручний доступ до функціоналу.

Платформа EduForAll відзначається високим рівнем доступності та зручності користування завдяки ряду інноваційних рішень у дизайні та функціоналі.

Користувачам доступно перемикаєти між різними секціями за допомогою табів. Це дозволяє ефективно використовувати різні функції та сервіси, не втрачаючи часу на надмірні пошуки.

Однією з ключових особливостей є контрастний дизайн, який використовує чорно-білі тони. Це не лише надає платформі стильний вигляд, але й полегшує виділення та розрізнення елементів для користувачів із порушеннями зору.

У веброзробці використано правильні компоненти, кожен з яких має відповідну роль, ім'я та лейбл. Це сприяє як правильному функціонуванню, так і зручності для користувачів з використанням асистивних технологій.

Використання семантичних тегів підвищує доступність та розуміння структури сторінки для асистивних технологій та користувачів із обмеженими можливостями.

Форми на платформі обладнані автозаповненням, що зберігає час користувача та робить введення інформації більш зручним. Кнопки активно виділяються при фокусуванні, що полегшує їх розпізнавання та натискання.

Особлива увага приділена реакції платформи на зміни розміру екрану. При збільшенні масштабу не виникає горизонтального скролінгу, а всі компоненти гнучко підлаштовуються під ширину екрану, забезпечуючи зручну та приємну взаємодію.

EduForAll відзначається інноваційним дизайном, високою доступністю та рядом унікальних функцій, роблячи її відмінним вибором для ефективного та комфортного навчання, в порівнянні дизайну та функціоналу популярні платформи, такі як Coursera, edX та Google Classroom.

Висновки до розділу 1

У висновку можна зазначити, що обговорені аспекти вебдоступності, освітні платформи та проблеми доступності на інтернет-платформах допоможуть створити сучасне, відкрите та інклюзивне освітнє середовище.

Вебдоступність, визнана ключовою складовою інтернет-простору, є фундаментальним елементом, що спрямований на забезпечення рівних можливостей та доступу до інформації для всіх користувачів, незалежно від їхніх фізичних чи когнітивних можливостей.

Навчальні платформи, такі як Coursera, edX і Google Classroom, демонструють високий ступінь візуальної, аудіальної, моторної та когнітивної доступності. Вони використовують технологічні та дизайнерські інновації, такі як текстові описи, субтитри, адаптивні інтерфейси та інші, для створення середовища, де навчання є доступним для всіх.

Аналіз проблем доступності на освітніх платформах підкреслює важливість комплексного підходу до їх розв'язання. Технічні та педагогічні аспекти, врахування різних потреб користувачів, а також використання інноваційних рішень є ключовими елементами створення інклюзивного освітнього середовища.

У розділу підкреслюється, що розвиток сучасної освіти тісно пов'язаний з розробкою та впровадженням вебдоступних технологій та платформ. Спільні зусилля в науці, технологіях та освіті допомагають не лише зробити освіту більш доступною, а й визначити нові стандарти в інклюзивному навчанні, сприяючи створенню глобального освітнього простору для всіх.

РОЗДІЛ 2. АНАЛІЗ, ПРОЕКТУВАННЯ ТА ТЕХНОЛОГІЇ ДЛЯ ДОСТУПНОЇ ОСВІТНЬОЇ ПЛАТФОРМИ

2.1. Аналіз існуючих підходів до платформ для навчання людей з обмеженою доступністю

Платформи для навчання, спрямовані на користувачів з обмеженими можливостями, визначаються своєю здатністю забезпечувати доступність, інклюзивність та ефективність для різних груп користувачів. Розглянемо аналіз існуючих підходів до таких платформ:

1. Універсальний дизайн і доступність: найефективніші платформи для навчання для людей з обмеженою доступністю розробляються з урахуванням принципів універсального дизайну. Це включає створення інтерфейсів та функціоналу, які легко адаптуються до різних потреб користувачів і забезпечують ефективний доступ до контенту навіть людям із сенсорними, зоровими чи слуховими обмеженнями [14].

2. Використання адаптивних технологій: найновітніші платформи використовують адаптивні технології, такі як штучний інтелект та машинне навчання, для персоналізації навчання. Це дозволяє адаптувати зміст до індивідуальних потреб користувачів і створює більш ефективне навчальне середовище [14].

3. Мультимедійний контент з аудіо-описом: платформи для навчання активно використовують мультимедійний контент, проте в інтересах доступності для осіб з вадами зору вони включають аудіо-опис до візуального матеріалу. Це дозволяє користувачам з обмеженими можливостями отримувати повноцінний досвід вивчення, навіть якщо вони не мають можливості переглядати чи читати текст [15].

4. Функціонал для взаємодії з екранними читачами: забезпечення сумісності з екранними читачами та надання адаптованих функціональних

можливостей для навігації — це ключовий аспект доступності. Це дозволяє користувачам із вадами зору ефективно взаємодіяти з платформою [16].

5. Оцінка та звітність щодо доступності: ефективні платформи включають інструменти для внутрішньої та зовнішньої оцінки доступності. Вони проводять тестування на відповідність стандартам доступності та надають звіти для забезпечення відкритості щодо своєї практики [17].

Цей аналіз спрямований на визначення найбільш передових підходів до створення навчальних платформ, які дійсно відкриті та ефективні для всіх користувачів, незалежно від їхніх можливостей чи обмежень.

2.2. Визначення вимог та потреб цільової аудиторії

З метою забезпечення максимальної доступності для різних категорій користувачів, платформа для навчання EduForAll враховує ряд особливостей доступного дизайну, а саме:

1. Висококонтрастний дизайн для дальтоніків: дизайн інтерфейсу платформи включає в себе висококонтрастний режим, зокрема чорний текст на білому фоні. Це не лише полегшує читання для всіх користувачів, але й спеціально адаптовано для людей з дальтонізмом, забезпечуючи оптимальний контраст тексту та фону [18].

2. Навігація табами для користувачів з поганою моторикою: платформа надає можливість навігації табами для користувачів з поганою моторикою. Це забезпечує ефективну та зручну взаємодію з інтерфейсом за допомогою клавіатури, дозволяючи швидко перемикатися між різними елементами без необхідності використання миші [19].

3. Підтримка Voice Over для зручності користування: однією з ключових особливостей платформи є її повна сумісність із Voice Over. Запрограмовано так, щоб всі тексти та важливі інформаційні елементи коректно читалися системним голосом, надаючи користувачам максимальний комфорт та доступність [20].

4. Динамічні відгуки та адаптація: платформа активно використовує динамічні відгуки та адаптивні елементи інтерфейсу, що дозволяє забезпечити зручність користування для різних груп користувачів. Це включає в себе зміни у відображенні при зміні розмірів екрану або виборі альтернативних режимів доступу [21].

5. Автозаповнення форм: форми на платформі обладнані автозаповненням, що зберігає час користувача та робить введення інформації більш зручним.

6. Безперервний текст при збільшенні масштабу: при збільшенні масштабу текст на платформі залишається чітким і не має пікселізації, забезпечуючи зручну читабельність.

7. Використання семантичних тегів: у веброзробці використовуються семантичні теги, що сприяють поліпшенню доступності та розуміння структури сторінки для асистивних технологій та користувачів із обмеженими можливостями [21].

8. Інтуїтивно зрозумілі компоненти платформи EduForAll розроблені так, щоб користувачі з легкістю могли орієнтуватись та використовувати їх без додаткових пояснень. Наприклад, при додаванні, редагуванні чи видаленні оцінок чи уроків, використовується модальне вікно. Це спеціальне вікно, яке виходить на передній план, блокуючи основний контент та виводячи лише необхідні елементи для взаємодії.

Платформа EduForAll виправдовує свою мету забезпечення максимальної доступності та зручності для користувачів. З висококонтрастним дизайном, навігацією табами, підтримкою Voice Over та іншими інноваціями, вона створює оптимальні умови для навчання. Автозаповнення форм, чіткий текст при збільшенні масштабу та інтуїтивно зрозумілі компоненти [21].

2.3. Огляд сучасних технологій для розробки доступних освітніх рішень

У світлі наукових та технологічних досягнень, розробка інноваційних освітніх рішень для людей з обмеженими можливостями стає актуальною ніж будь-коли. Це вимагає не лише підходу, що враховує індивідуальні потреби, але й використання передових технологій для створення інклюзивного та доступного навчального середовища.

В наш час обрані технології стають не лише інструментами розробки, але й ключовими кроками у напрямку створення платформ для навчання, які враховують індивідуальні особливості користувачів. Цей огляд присвячений розгляду та аналізу таких технологій як MongoDB, Prisma, Remix, React, TypeScript і Tailwind, які обрані для створення інклюзивної освітньої платформи [22].

MongoDB відзначається гнучкістю та простотою управління даними. Ця документ-орієнтована база даних ідеально підходить для сховища великої кількості освітніх матеріалів та інформації про користувачів. Її схема визначається динамічно, що полегшує процес розробки та зміни.

Prisma є сучасним ORM-інструментом, який спрощує роботу з базами даних та взаємодію з ними. В поєднанні з React та TypeScript, Prisma стає потужним інструментом для роботи з даними, забезпечуючи типізацію та безпеку взаємодії з базою даних [22].

Remix є інноваційним фреймворком для розробки вебдодатків, який пропонує високий рівень продуктивності та модульності. Він спрощує процес розробки та взаємодії між компонентами, роблячи його ідеальним вибором для створення платформи для освіти.

React є одним з найпопулярніших фреймворків для розробки інтерфейсів. Він надає декларативний підхід до створення UI, що полегшує розробку складних вебдодатків. З React легко створювати компоненти та ефективно оновлювати їх стан [22].

TypeScript, як строго типізована обгортка поверх JavaScript, забезпечує безпеку та підтримує читабельність коду. В поєднанні з React, TypeScript дозволяє розробникам масштабувати свої проекти, зменшуючи кількість помилок та полегшуючи рефакторинг [22].

Tailwind CSS — це модульний та високопродуктивний CSS-фреймворк. Його особливість у тому, що він дозволяє швидко та ефективно створювати інтерфейси, використовуючи зрозумілі класи. Зручний синтаксис Tailwind сприяє швидкій інтеграції та розробці вигляду платформи [13].

Чому саме ці технології:

- гнучкість та легкість управління даними: MongoDB дозволяє зберігати дані у документ-орієнтованому форматі, що підходить для зберігання різноманітної освітньої інформації;
- типізація та безпека даних: використання TypeScript та Prisma забезпечує статичну типізацію та безпеку даних, що допомагає уникнути багатьох типових помилок;
- швидка розробка та модульність: Remix дозволяє швидко розробляти та підтримувати проекти завдяки своїй модульності та зручному інтерфейсу;
- реактивний UI та компонентний підхід: використання React дозволяє легко створювати взаємодіючі та динамічні інтерфейси для освітніх рішень;
- інтеграція та оптимізація вигляду: Tailwind CSS сприяє швидкій інтеграції та розробці зручного та естетичного вигляду платформи.

Всі ці технології взаємодіють між собою для створення потужної, швидкої та доступної платформи для освіти, а їхні особливості і переваги взаємодоповнюють одна одну для оптимального результату.

2.4. Проектування та архітектура рішення

У межах проектування платформи для навчання людей з обмеженими можливостями, основний акцент зроблено на забезпеченні максимальної зручності та ефективності для вчителів та учнів. Давайте детальніше розглянемо ключові аспекти розділу «Проектування та Архітектура Рішення»:

Форма входу та реєстрації

Платформа розроблена з урахуванням зручності користування та адаптивності до потреб різних категорій користувачів. Наявність двох окремих форм – для учнів та вчителів – визначається за допомогою кнопки «Перемикач Режимів». Це інтуїтивно зрозуміле рішення дозволяє не лише легко розпізнати статус користувача, але й адаптує інтерфейс та функціонал до конкретних потреб кожної групи.

Зокрема, кнопки перемикання не обмежуються лише режимами, але також забезпечують зручний перехід між формами входу та реєстрації. Такий підхід підсилює зручність використання платформи для кожного користувача.

Режим вчителя можна побачити на рисунку нижче (рис. 2.1)

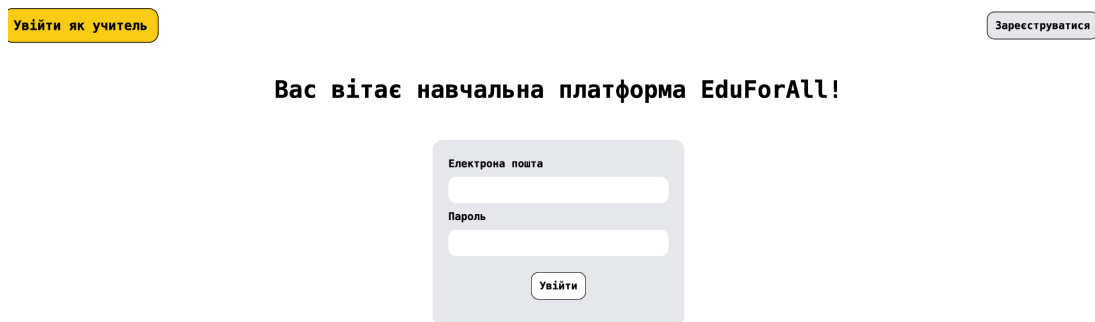
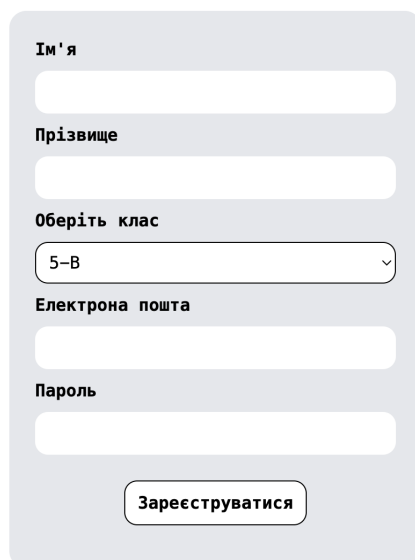


Рис. 2.1 Вигляд режиму учителя на формі входу в систему

Надто важливо відзначити, що реєстраційна форма для учнів додатково включає поле для вибору класу, реєстраційну форму зображено на рисунку нижче (рис. 2.2). Ця деталь дозволяє точно ідентифікувати клас учня, сприяючи зручній навігації вчителів та оптимізації керування класами. Такий

інтегрований підхід до реєстрації підкреслює нашу відданість створенню адаптивної та користувацько-орієнтованої платформи.

Вас вітає навчальна платформа EduForAll!



Ім'я

Прізвище

Оберіть клас

5-8

Електронна пошта

Пароль

Зареєструватися

Рис. 2.2 Реєстраційна форма для режиму учень

Платформа EduForAll дозволяє вам легко та ефективно перемикатися між різними розділами, використовуючи зручні таби для навігації. Кожна форма на платформі оснащена автозаповненням, забезпечуючи швидке та зручне введення інформації. Кожен тег `input` має свій власний лейбл, спеціально розроблений для оптимальної взаємодії з програмами читання екрану. Забезпечено прив'язку кожного `input` до тегу `label`, що підвищує доступність та розуміння структури сторінки. Крім того, форми оснащені обробкою помилок, що робить користування платформою безпечним та безперебійним.

Меню та Вихід з Системи

Ліве меню, доступне після входу, є центром функціональності для вчителів та учнів, це показано на рисунку (рис. 2.3). Воно створене так, щоб забезпечити зручний та ефективний доступ до всіх можливостей платформи, спрощуючи навігацію та покращуючи взаємодію з системою.

Зверху екрану розташовано привітання, яке відображає ім'я та прізвище користувача, відображено на рисунку (рис. 2.3). Це не лише створює

персоналізований та дружелюбний інтерфейс, але й надає користувачеві важливу інформацію щодо його облікового запису. Окрім того, вказано режим, у якому користувач увійшов – чи це режим учня, чи вчителя, що полегшує йому сприйняття його ролі та можливостей.

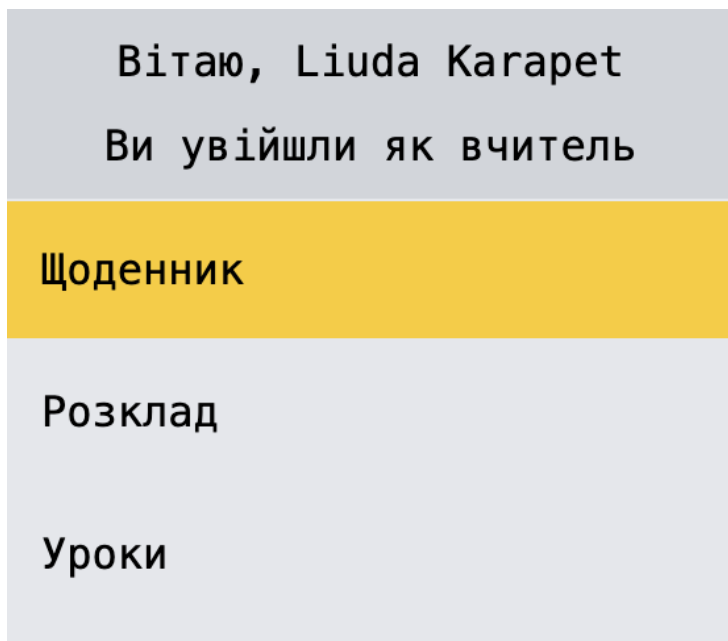


Рис. 2.3 Меню навчальної платформи EduForAll

Опція «Вихід з Системи» відзначається зручністю та чіткістю, дозволяючи користувачеві швидко та безпечно завершити сесію. Кнопка не лише забезпечує безпечний вихід, але також забезпечує ефективний перехід, спрямовуючи користувача на форму входу. Це сприяє не лише швидкому вийденню з системи, але й робить цей процес максимально зручним та легким для користувача. На рисунку нижче (рис. 2.4) можна побачити активну кнопку «Вихід».

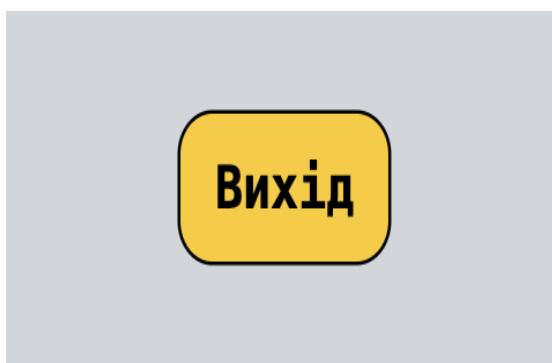


Рис. 2.4 Зображено активну кнопку «Вихід».

Щоденник: Оцінки та Редагування

В верхній частині розділу «Щоденник» знаходиться зручне поле параметрів, де вчителі можуть визначити обраний клас, предмет та місяць. Це інтуїтивно зрозуміле інтерфейсне рішення робить ведення обліку оцінок швидким та ефективним процесом. На наступному рисунку (рис. 2.5) можна побачити вигляд тулбару в розділі «Щоденник».

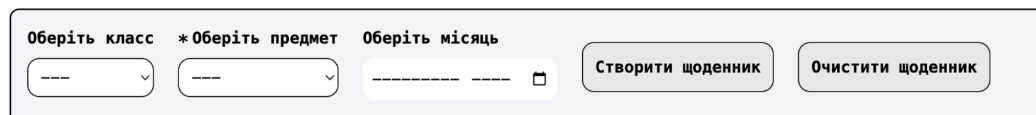


Рис. 2.5 Поле параметрів в розділі «Щоденник»

Після вибору параметрів, система автоматично створює таблицю, в якій перелічені усі учні обраного класу. Ця таблиця, яка зображена на рисунку (рис. 2.6), відображає наявність оцінок для кожного учня у вигляді комірка, які підсвічуються жовтим кольором. Додатковий символ у вигляді плюса на ячейці вказує, що в цей конкретний день заплановано проведення уроку, і вчитель може виставити оцінку.

Щоденник

Оберіть клас *Оберіть предмет Оберіть місяць																																		
5-8 Українська November 2023																																		
Учні	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30				
Vasja Pupkin	+						+	4	+	+			4	4	+	5	+											+						
Петро Щур	+						+	+	+	+			10	+	+	+	4											+						
Таня Вакуленко	+						+	+	+	+			+	+	+	+	+											+						

Рис. 2.6 Таблиця з переліком учнів та їх оцінок

В розділі «Щоденник» впроваджено модальні вікна для редагування, додавання та видалення оцінок. Натискання на комірку, що має оцінку, відкриває модальне вікно, де вчителі можуть виконувати редагування та видалення. Якщо комірка містить плюсики, відбувається аналогічний процес, проте із можливістю додавання нових оцінок. Ця інноваційна функціональність підсилює зручність та ефективність ведення обліку успішності, надаючи вчителям повний контроль над процесом оцінювання в реальному часі.

Календар: планування та перегляд

Розділ «Календар» виявляється важливою складовою для планування та координації розкладу класу на визначений період. Платформа пропонує вчителям зручні інструменти для організації уроків та ефективного ведення навчального процесу. Вигляд цього розділу можна побачити на рисунку (рис. 2.7).

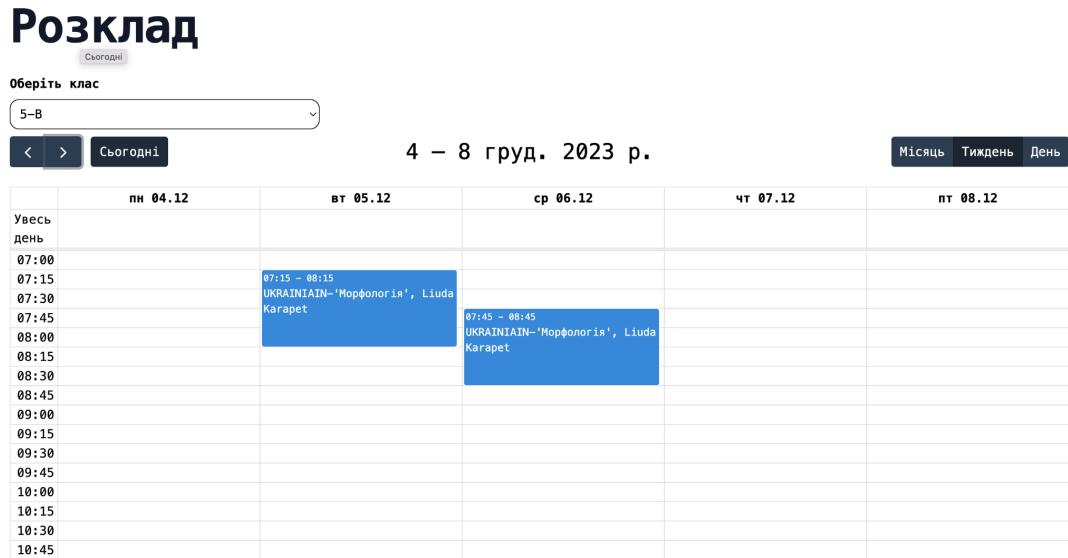


Рис. 2.7 Календар у режимі тиждень

У верхній частині розділу розташоване дропдаун-меню, яке дозволяє вибрати конкретний клас для перегляду розкладу. Використання стандартного календаря надає можливість вчителям переглядати розклад у форматі місяця, тижня або дня, забезпечуючи гнучкість планування на різних часових періодах.

Особливістю є можливість додавати нові уроки, вказуючи необхідні деталі, такі як опис, кабінет та тема. Використання стандартних додаткових функцій календаря, таких як перегляд повного місяця, тижня або дня, дозволяє зручно оцінювати графік та взаємодіяти з ним.

Забезпечуючи вчителю лише доступ до його предметів та обмежуючи можливості редагування інших предметів, платформа забезпечує безпеку та структурованість управління розкладом. Функції видалення та редагування уроків додають гнучкості вчителям у керуванні навчальним процесом.

Розділ «Уроки» надає зручний інструмент для перегляду історії проведених уроків з можливістю детальної фільтрації за періодом. Уроки вже автоматично відфільтровані за періодом від початку місяця до його завершення, а також за конкретним часовим проміжком.

Деталізована інформація про кожен урок включає тему, час початку та закінчення, кабінет та опис. Це надає вчителям можливість отримати повний та структурований звіт про проведені уроки, а також ефективно контролювати відповідність освітньої програми.

Розділ «Уроки» : планування та перегляд

Розділ «Уроки», що показано на рисунку (рис. 2.8) виконує важливу роль у веденні обліку та контролі за викладанням різних тем згідно з навчальним планом. Забезпечуючи можливість перевірки даних уроків та порядку їх викладення, цей розділ сприяє систематизації та оцінці навчального процесу.

Предмети

Оберіть клас	* Оберіть предмет	Оберіть місяць	Отримати уроки
5-B	Українська	November 2023	

Урок 1

Тема: Морфологія
Опис: Морфологія
Початок уроку: 21/11/2023 9:10:
Кінець уроку 22/11/2023 9:45:
Кабінет:

Урок 2

Тема: Морфологія
Опис: Морфологія
Початок уроку: 27/11/2023 05:45
Кінець уроку 27/11/2023 06:15
Кабінет:

Рис. 2.8 Розділ «Уроки» на освітній платформі EduForAll

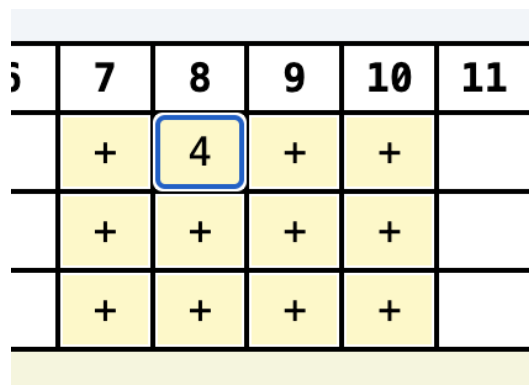
Головне завдання нашої платформи – висока вебдоступність. У сучасному цифровому середовищі важливо забезпечити, щоб освітня платформа була доступною для різних категорій користувачів, незалежно від їхніх можливостей та потреб. З цього приводу, приділяємо особливу увагу певним аспектам, що роблять наш вебсайт доступним та зручним для використання всіма.

Табова навігація:

Опис: вебсайт забезпечує можливість перемикання між інтерактивними елементами за допомогою клавіші Tab.

У розробці вебсайту використані спеціальні HTML-теги, такі як `button`, `input`, а тощо, які вже мають вбудовану підтримку табової навігації, автоматично встановлюючи порядок фокусу між інтерактивними елементами. Приклад фокусування на комірці у розділі «Щоденник» зображено на рисунку (рис. 2.9), використано тег `button`, для того щоб будь можливість додавати оцінки. Крім того, для кращої доступності використовуються атрибути, які надають додаткові вказівки асистивним технологіям, поліпшуючи взаємодію та розуміння контенту користувачами із різними потребами [14];

Важливість: для користувачів із зниженою рухливістю або використанням асистивних технологій, така навігація дозволяє ефективно взаємодіяти з вмістом без використання миші.



	7	8	9	10	11
	+	4	+	+	
	+	+	+	+	
	+	+	+	+	

Рис. 2.9 Активна комірка у розділі «Щоденник»

Масштабованість контенту:

Опис: при збільшенні масштабу на 200%, вміст автоматично адаптується до ширини екрану користувача [14].

За допомогою CSS, а саме властивостей `flex` створено привабливий дизайн вебсайту, який при збільшенні масштабу залишається структурованим. Також не з'являється горизонтального скролу та утримується чіткість тексту навіть при збільшенні масштабу на 200%. Крім того, вміст автоматично адаптується до ширини екрану, щоб забезпечити комфортне користування.

Вигляд розділу «Календар» при збільшенні масштабу до 200% можна побачити на рисунку (рис. 2.10).

Важливість: забезпечує комфортне читання та сприяє доступності для осіб із зоровими обмеженнями, дозволяючи їм зручно сприймати текст та інші елементи.

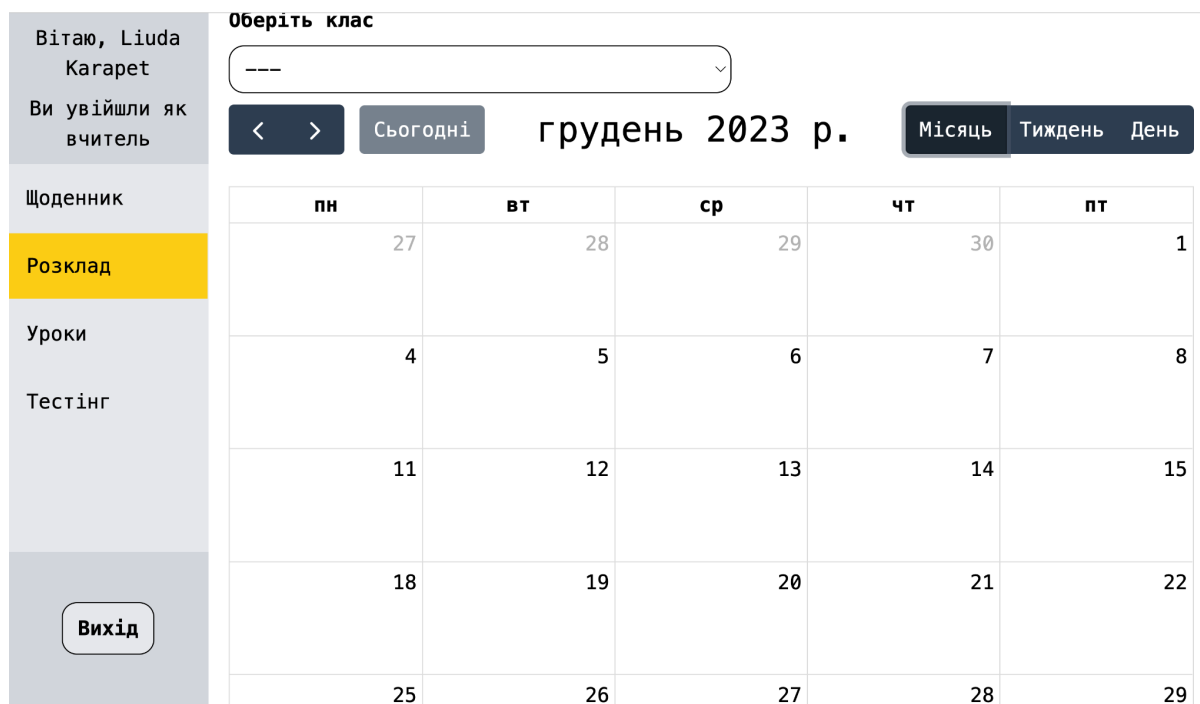


Рис. 2.10 Розділ «Календар» при збільшенні масштабу до 200%

Семантичні теги та ролі:

Опис: всі елементи мають визначений тип та роль за допомогою семантичних тегів, що надає додаткову інформацію асистивним технологіям [15]. Приклад використання семантичного тегу `nav` та ролі `menubar` показано на рисунку нижче (рис. 2.11).

Важливість: підвищує доступність для користувачів з визначеними потребами, дозволяючи їм точно розуміти та взаємодіяти із вмістом сайту.

```

▼<nav class="w-auto flex-1 overflow-y-auto flex flex-col" aria-label="EduForAll"> flex
  ▼<ul role="menubar" aria-label="EduForAll" aria-orientation="vertical" class="flex flex
    -col"> flex
      <a role="menuitem" class="nav-link p-3.5 m-[1px]" href="/home/school-diary">Щоденник
      </a>
      <a role="menuitem" class="nav-link p-3.5 m-[1px]" active="" href="/home/schedule" aria-
        current="page">Розклад</a>
      <a role="menuitem" class="nav-link p-3.5 m-[1px]" href="/home/lessons">Уроки</a>
      <a role="menuitem" class="nav-link p-3.5 m-[1px]" href="/home/testing">Тестінг</a>
    </ul>
  </nav>

```

Рис. 2.11. Використання семантичних тегів та ролей у меню платформи
Контрастний дизайн:

Опис: тема має високий контраст, зокрема чорний текст на білому фоні, для полегшення читання та розрізнення елементів [16].

Забезпечення високої контрастності на сайті через використання контрастного чорного тексту на білому фоні. Це конкретне значення контрастності забезпечує максимальну візуальну різницю між текстовими елементами та фоном, що допомагає полегшити читання та покращити видимість. Застосування максимальної контрастності є важливим аспектом дизайну, особливо для користувачів із зниженим зором чи іншими візуальними особливостями. Для прикладу контрастності можна переглянути наступний рисунок (рис. 2.12).

Важливість: забезпечує зручність користування особам із зниженим зором та дальтонізмом, покращуючи видимість та розпізнавання кольорів.

Рис. 2.12. Модальне вікно додавання уроку

Збільшувальні кнопки з контрастним кольором [16]:

Опис: кнопки на вебсайті збільшуються при наведенні та мають жовте відзначення для покращення їх видимості.

Виділення кнопок жовтим кольором при наведенні сприяє поліпшенню їх видимості, особливо для тих, хто має обмеження в зорі. Збільшення розміру кнопок при наведенні полегшує їх використання користувачами з обмеженою моторикою, забезпечуючи більше місця для натискання. Адаптивний дизайн забезпечує правильне відображення кнопок на різних пристроях, що дозволяє користувачам легко взаємодіяти з ними навіть на мобільних пристроях. Використання семантичного HTML-коду, такого як `button`, сприяє покращенню доступності для асистивних технологій та полегшує розуміння контексту кнопок. Вигляд кнопок показано на рисунку нижче (рис. 2.13).

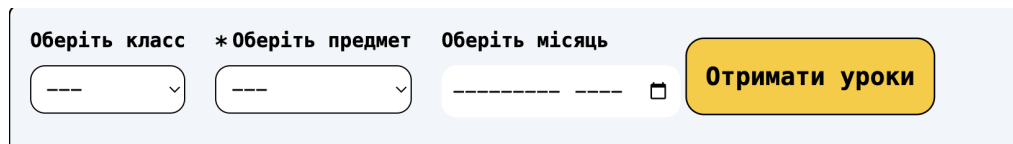


Рис. 2.13. Поле параметрів розділу «Уроки»

Важливість: полегшує використання сайту для осіб із зниженим зором або обмеженою моторикою, покращуючи точність та доступність інтерактивних елементів.

Цей розділ ретельно враховує потреби вчителів та учнів, створюючи функціонал, який сприяє ефективній роботі та полегшує введення навчального процесу для всіх учасників. Інклюзивний дизайн та новаторські підходи забезпечують універсальність EduForAll, де кожен користувач може легко взаємодіяти з системою та отримувати високоякісний досвід навчання.

Висновки до розділу 2

Загальний висновок розділу можемо зробити такий, що розроблена освітня платформа має високий потенціал для створення доступного та інклюзивного навчального середовища. Застосування передових технологій, таких як MongoDB, Prisma, Remix, React, TypeScript і Tailwind CSS, сприяє гнучкості, ефективності та зручності використання.

Важливим аспектом є фокус на індивідуальних потребах користувачів з обмеженими можливостями, що виявляється у вигляді зручного інтерфейсу та різноманітних функцій для вчителів та учнів. Високий рівень персоналізації та можливість легкої адаптації до різних груп користувачів роблює платформу відмінним інструментом для організації навчання.

Проектування платформи враховує важливі аспекти, такі як форми входу та реєстрації, меню та системи оцінювання, які сприяють зручності та ефективності користування для обох типів користувачів. Особлива увага приділяється безпеці та структурованості управління розкладом та історією уроків.

Забезпечення високої вебдоступності та врахування різних потреб користувачів через табову навігацію, масштабованість контенту, семантичні теги, контрастний дизайн і збільшувальні кнопки забезпечує широкий доступ до платформи для всіх.

РОЗДІЛ 3. РОЗРОБКА МАРШРУТІВ, АПРОБАЦІЯ ТА ПІДТВЕРДЖЕННЯ НАУКОВОЇ НОВИЗНИ МАРШРУТІВ

3.1 Реалізація маршрутів та компонентів

3.1.1 Розробка маршруту входу, реєстрації та виходу для режимів вчителя та учня

Цей фрагмент коду визначає loader та action функції, які використовуються в Remix Framework для обробки запитів і виконання операцій на сервері. Також визначено тип `ActionData` та імпортовані різні модулі та компоненти. Основні елементи коду [22]:

1. `ActionData` тип – визначає тип даних, які можуть бути повернуті з action функції. Включає об'єкт із помилками (`errors`), загальною помилкою (`error`), та полями форми (`fields`) [22].
2. loader функція:
 - функція, яка викликається при завантаженні сторінки;
 - отримує дані про класи за допомогою `getAllClasses`;
 - перевіряє, чи користувач вже авторизований за допомогою `getUser(request)`;
 - в залежності від результату, викликає `redirect(«/»)` або повертає дані класів.
3. action функція:
 - функція, яка викликається при виконанні дій на сторінці (наприклад, при натисканні кнопок);
 - отримує дані з форми, що була відправлена клієнтом;
 - перевіряє коректність отриманих даних та виконує валідацію;

- в залежності від дії («login» або «register») викликає відповідні функції (login або register) [23];
- повертає результат відповідно до виконаної дії, або повертає помилку, якщо дані некоректні.

В цьому коді також дотримані певні стандарти, такі як використання типізації, використання хуків з React та Remix Framework, та реакція на статусні коди при поверненні відповіді. Помилки та статус коди 400 використовуються для повідомлення клієнту про помилки введених даних або невірний тип запиту.

Лістинг 3.1

Реалізація маршруту login

```
import { Fragment, useState } from "react";
import { useActionData, useLoaderData } from "@remix-run/react";
import { useRef, useEffect } from "react";
import { Layout } from "~/components/layout";
import { FormField } from "~/components/form-fields";
import {
  ActionFunction,
  json,
  LoaderFunction,
  redirect,
} from "@remix-run/node";
import { login, register, getUser } from "~/utils/auth.server";
import {
  validateEmail,
  validateName,
  validatePassword,
} from "~/utils/validators.server";
import { getAllClasses } from "~/utils/classroom.server";
import { DropdownMenu } from "~/components/dropDownMenu";
import { formOptionsFromArray } from "~/helpers/formHelpers";
import { Role } from "@prisma/client";

type ActionData = {
  errors: Record<string, string> | null;
  error: string | null;
  fields: {
    email: string;
    password: string;
    firstName: string;
    lastName: string;
  };
};
```

```

} | null;
};

export const loader: LoaderFunction = async ({ request }) => {
  const classes = await getAllClasses();
  return (await getUser(request)) ? redirect("/") : json({ classes });
};

type IClassId = string | null;
export const action: ActionFunction = async ({ request }) => {
  const form = await request.formData();
  const role = form.get("role");
  const action = form.get("_action");
  let firstName = form.get("firstName");
  let lastName = form.get("lastName");
  const email = form.get("email");
  const password = form.get("password");
  const classId = form.get("class_id");

  if (
    typeof action !== "string" ||
    typeof email !== "string" ||
    typeof password !== "string"
  ) {
    return json({ error: `Invalid Form Data`, form: action }, { status:
400 });
  }

  if (
    action === "register" &&
    (typeof firstName !== "string" || typeof lastName !== "string")
  ) {
    return json({ error: `Invalid Form Data`, form: action }, { status:
400 });
  }

  const errors = {
    email: validateEmail(email),
    password: validatePassword(password),
    ...(action === "register"
      ? {
          firstName: validateName((firstName as string) || ""),
          lastName: validateName((lastName as string) || ""),
        }
      : {}),
  };
};

```

```

if (Object.values(errors).some(Boolean))
  return json(
    {
      errors,
      fields: { email, password, firstName, lastName },
      form: action,
    },
    { status: 400 }
  );
switch (action) {
  case "login": {
    return await login({ email, password }, role as Role);
  }
  case "register": {
    firstName = firstName as string;
    lastName = lastName as string;
    return await register(
      { email, password, firstName, lastName },
      role as Role,
      classId as IClassId
    );
  }
  default:
    return json({ error: `Invalid Form Data` }, { status: 400 });
}
};

```

Фрагмент коду який розміщений містить в собі використання різних хуків `useEffect` та `useState` з бібліотеки `React`, а також деякі виклики функцій з бібліотеки `Remix Framework`.

- `useEffect(() => { document.title = 'EduForAll - Вхід'; });` – встановлює заголовок сторінки на «EduForAll - Вхід» при завантаженні компонента;
- `const { classes } = useLoaderData();` – використовує `useLoaderData` для отримання даних, які були завантажені на сервері під час фази завантаження сторінки. У данному випадку, отримується об'єкт класів [24].

Стани та хуки:

- `const [action, setAction] = useState(«login»);` встановлює початковий стан для типу дії (login чи register) [24];

- `const [role, setRole] = useState(«student»);`: встановлює початковий стан для ролі користувача (student чи teacher);
- `const actionData: ActionData | undefined = useActionData();`: використовує `useActionData` для отримання даних, що повертаються з серверу під час виконання дій. Отримує об'єкт `actionData`, який містить інформацію про помилки, дані форми та загальну помилку [25];
- `const firstLoad = useRef(true);`: використовує `useRef` для створення посилання на змінну, яка буде зберігатися між рендерами компонента.

Робота з Помилками та Даними Форми:

- `const [errors, setErrors] = useState(actionData?.errors || {});`: -- встановлює стан для помилок валідації форми [26];
- `const [formError, setFormError] = useState(actionData?.error || "");`: – встановлює стан для загальної помилки форми [26];
- `const [formData, setFormData] = useState({/*...*/});`: встановлює стан для даних форми, включаючи email, password, firstName та lastName [26];
- `const handleInputChange = (event: React.ChangeEvent<HTMLInputElement>, fields: string) => {/*...*/};`: функція, яка обробляє зміни у введенні форми та оновлює стан форми в залежності від змін.

Додаткові `useEffect`:

- `useEffect(() => {/*...*/}, [action]);`: – викликається при зміні типу дії (action). Якщо це не перший рендер, очищає стан помилок та дані форми.
- `useEffect(() => {/*...*/}, [formData]);`: – викликається при зміні даних форми. Якщо це не перший рендер, очищає загальну помилку форми [26].
- `useEffect(() => { firstLoad.current = false; }, []);`: – викликається тільки під час першого рендеру компонента. Встановлює `firstLoad.current` в `false`, вказуючи, що перший рендер вже відбувся.

Реалізація маршруту login

```

export default function Login() {
  useEffect(() => {
    document.title = 'EduForAll - Вхід';
  });

  const { classes } = useLoaderData();
  const [action, setAction] = useState("login");
  const [role, setRole] = useState("student");

  const actionData: ActionData | undefined = useActionData();
  const firstLoad = useRef(true);
  const [errors, setErrors] = useState(actionData?.errors || {});
  const [formError, setFormError] = useState(actionData?.error || "");

  const [formData, setFormData] = useState({
    email: actionData?.fields?.email || "",
    password: actionData?.fields?.password || "",
    firstName: actionData?.fields?.lastName || "",
    lastName: actionData?.fields?.firstName || "",
  });

  const handleInputChange = (
    event: React.ChangeEvent,
    fields: string
  ) => {
    setFormData((form) => ({ ...form, [fields]: event.target.value }));
  };

  const STUDENT_ROLE = "student";
  const isStudent = (role: string): boolean => role === STUDENT_ROLE;

  useEffect(() => {
    if (!firstLoad.current) {
      const newState = {
        email: "",
        password: "",
        firstName: "",
        lastName: "",
      };
      setErrors(newState);
      setFormError("");
      setFormData(newState);
    }
  }, [action]);
}

```

```

useEffect(() => {

  if (!firstLoad.current) {
    setFormError("");
  }
}, [formData]);

useEffect(() => {
  firstLoad.current = false;
}, []);

```

Цей код в основному відповідає за ініціалізацію та обробку станів компонента, а також за відображення та оновлення даних форми та повідомлень про помилки.

Компонент під назвою Login є частиною інтерфейсу користувача і включає в себе різноманітні елементи, такі як кнопки, форму, і заголовки, з метою надання можливості користувачеві авторизуватися або зареєструватися на навчальній платформі «EduForAll». Давайте розглянемо основні аспекти цього компонента, зосереджуючись на аспектах доступності:

- ізоляція фрагментів з Fragment – використання `<Fragment>` допомагає уникнути додавання зайвого DOM-елементу, що поліпшує читабельність коду;
- заголовок сторінки (`document.title`) – заголовок сторінки встановлюється на «EduForAll - Вхід», що допомагає визначити зміст сторінки для користувача, використовуючи екранні читачі [27];
- використання `role`, `aria-label` та `aria-required` – у кнопках використовуються атрибути `role`, `aria-label` та `aria-required` для забезпечення кращої доступності. Наприклад `role=«link»` вказує, що кнопка функціонує як посилання; `aria-label` надає текстовий опис, який необхідний для екранних читачів або інших адаптивних технологій; `aria-required` позначає, які поля є обов'язковими для заповнення [27];

- зміна ролі та дії з використанням кнопок – кнопки «Увійти як учитель/учень» та «Перейти до реєстрації/входу» дозволяють користувачеві легко змінити свою роль та тип дії;
- валідація форми та виведення помилок - при введенні неправильних даних у формі виводяться повідомлення про помилки, що полегшує їх виправлення. Для кожного поле вводу використовуються атрибути `ariaLabel`, `ariaRequired`, та `htmlFor` для покращення доступності [27].
- використання `DropDownMenu` – коли реєстрація виконується як студент, виводиться випадаючий список (`DropDownMenu`) для вибору класу, що полегшує вибір класу студентам;
- зміна мови та стилів – компонент використовує текст на українській мові, що поліпшує зручність для цільової аудиторії;
- анімаційні та стилізовані компоненти – використання анімацій, стилізованих кнопок та заголовків додає інтерактивності та естетичного вигляду, не знижуючи при цьому доступність.

Лістинг 3.3

Реалізація маршруту login

```
<Fragment>
  <Layout>
    <header className="flex align-center justify-between">
      <section>
        <button
          type="button"
          role="link"
          aria-label={`Увійти як ${isStudent(role) ? "вчитель" :
"студент"}}`
          onClick={() => setRole(isStudent(role) ? "teacher" :
"student")}
          className="button"
        >
          Увійти як {isStudent(role) ? "учитель" : "учень"}
        </button>
      </section>
    <section>
      <button
        type="button"
```

```

role="link"
  aria-label={`Перейти до ${action == "login" ? "реєстрації" :
"входу"}}`}
  onClick={() => setAction(action == "login" ? "register" :
    "login")}
    className=«button»
  >
    {action === "login" ? "Зареєструватися" : "Увійти"}
  </button>
</section>
</header>
<main className="h-full items-center flex flex-col gap-y-4">
  <section>
    <h1 className="m-10 text-4xl font-extrabold text-center">
      Вас вітає навчальна платформа EduForAll!
    </h1>
  </section>

  <section>
    <form method="post" className="rounded-2xl bg-gray-200 p-6
w-96">
      <div role="alert" className="text-xs font-semibold
text-center tracking-wide text-red-500 w-full">
        {formError}
      </div>
      {action === "register" && (
        <>
          <FormField
            htmlFor="firstName"
            label="Ім'я"
            ariaLabel="Введіть своє ім'я"
            ariaRequired={true}
            onChange={(e) => handleInputChange(e, "firstName")}
            value={formData.firstName}
            error={errors?.firstName}
          />
          <FormField
            htmlFor="lastName"
            label="Прізвище"
            ariaLabel="Введіть своє прізвище"
            ariaRequired={true}
            onChange={(e) => handleInputChange(e, "lastName")}
            value={formData.lastName}
            error={errors?.lastName}
          />
        </>
      )}
    </form>
  </section>

```

```

        {role === "student" && (
          <DropDownMenu
label="Оберіть клас"
          ariaLabel="Оберіть клас"
          ariaRequired={true}
          options={formOptionsFromArray(classes, "id",
            "name")}
          name="class_id"
        />
        )}
      </>
    )}

    <FormField
      htmlFor="email"
      label="Електрона пошта"
      ariaLabel="Введіть свою електронну пошту"
      ariaRequired={true}
      value={formData.email}
      onChange={(e) => handleInputChange(e, "email")}
      error={errors?.email}
    />

    <FormField
      htmlFor="password"
      label="Пароль«"
      ariaLabel="Введіть свій пароль"
      ariaRequired={true}
      value={formData.password}
      onChange={(e) => handleInputChange(e, "password")}
      error={errors?.password}
    />
    <input type="hidden" name="role" value={role} />
    <section className="w-full text-center">
      <button
        type="submit"
        name="m_action"
        value={action}
        aria-label={
          action === "login"
            ? "Увійти в аккаунт"
            : "Зареєструвати свій аккаунт"
        }
      />
      className="button !bg-white hover:!bg-yellow-400 focus:!bg-yellow-400"
    >

```

```

        {action === "login" ? "Увійти" : "Зареєструватися"}
      </button>

</section>
  </form>
</section>
</main>
</Layout>
</Fragment>

```

Цей компонент взаємодіє із користувачем, надаючи йому зручний інтерфейс для авторизації та реєстрації, забезпечуючи при цьому високий рівень доступності для різних користувачів, включаючи тих, які використовують екранні читачі та інші технології адаптивного введення.

Вигляд форми реєстрації подивимось нижче (рис 3.1).

Увійти як учитель

Зареєструватися

Вас вітає навчальна платформа EduForAll!

Електронна пошта

Пароль

Увійти

Рис. 3.1. Форма входу на освітній платформі EduForAll

3.1.2 Реалізація маршруту home

Для створення домашньої сторінки з елементами навігації. Розглянемо ключові складові та обговоримо елементи, які спрямовані на поліпшення доступності.

Реалізація маршруту home

```

import type { LoaderFunction } from "@remix-run/node";
import { NavLink, Outlet, useLoaderData, useNavigate } from
"@remix-run/react";
import { requireUserIdAndRole } from "~/utils/auth.server";
import { Layout } from "~/components/layout";
import { getUserByIdAndRole } from "~/utils/user.server";
import { LogoutComponent } from "~/components/logoutComponent";
import { UserWelcome } from "~/components/user-welcome";
import { getAllClassrooms } from "~/utils/classroom.server";
import { useEffect } from "react";

export const loader: LoaderFunction = async ({ request, params }) => {
  const { userId, userRole } = await requireUserIdAndRole(request);
  const user = await getUserByIdAndRole(userId, userRole);
  const allClasses = await getAllClassrooms();
  return { user, allClasses };
};

export default function Home() {
  useEffect(() => {
    document.title = 'Домашня сторінка';
  });

  const { user, allClasses } = useLoaderData();
  const navigate = useNavigate();

  // KEYBOARD CONTROLS

  const links = [
    { label: "Щоденник", ariaLabel: "Перейти до щоденника", to:
"school-diary" },
    { label: "Розклад", ariaLabel: "Перейти до розкладу", to: "schedule"
},
    { label: "Уроки", ariaLabel: "Перейти до уроків", to: "lessons" }
  ];

  const generateLinkClassName = (isActive: boolean, isPending: boolean):
string => {
    let className = "nav-link p-3.5 m-[1px] ";

    if (isPending) {
      className += "pending ";
    }
  }

```

```

    if (isActive) {
      className += "active ";
    }

    return className;
  };

  return (
    <Layout>
      <div className="h-full flex">
        <aside className="w-1/6 bg-gray-200 flex flex-col">
          <section className="flex items-center justify-center p-2 pb-0 text-center bg-gray-300">
            {user && <UserWelcome key={user.id} profile={user.profile} />}
          </section>
          <section className="flex items-center justify-center p-2 text-center bg-gray-300">
            Ви увійшли як {user?.type === 'teacher' ? 'вчитель' : 'учень'}
          </section>

          <nav className="w-auto flex-1 overflow-y-scroll flex flex-col" aria-label="EduForAll">
            <ul role="menubar" aria-label="EduForAll" aria-orientation="vertical" className="flex flex-col">
              {links.map(link => (
                <NavLink
                  key={link.to}
                  className={({ isActive, isPending }) =>
                    generateLinkClassName(isActive, isPending)}
                  to={link.to}
                  role="menuitem"
                >
                  {link.label}
                </NavLink>
              ))}
            </ul>
          </nav>
          <section>
            <LogoutComponent />
          </section>
        </aside>
        <main className="w-5/6 p-4">
          <Outlet />
        </main>
      </div>
    </Layout>
  );
}

```

```

    </div>
  </Layout>
);
}

```

Цей компонент використовується для створення домашньої сторінки додатка і містить ряд елементів для навігації користувача. Давайте розглянемо його ключові складові, звертаючи особливу увагу на аспекти, що сприяють поліпшенню доступності [27].

Заголовок сторінки:

- використання `useEffect` для динамічного встановлення заголовку сторінки («Домашня сторінка»). Це може бути корисно для користувачів із зокрема екранними читачами, які отримують аудіо інформацію про зміни заголовку [27].

Навігаційна панель:

- бічна навігаційна панель (`aside`) містить розділи для вітання користувача та відображення його ролі (вчитель чи учень) [27];
- навігаційна панель також включає перелік посилань на різні розділи додатка: «Щоденник», «Розклад», «Уроки»;
- використання класів та стилів для активних та очікуючих посилань, що полегшує сприйняття користувачами.

Контентна область:

- головна частина містить зміст кожного розділу, який відображається в залежності від вибраного посилання.

Додаткові елементи:

- використання `UserWelcome` для привітання користувача та відображення інформації про його профіль;
- компонент `LogoutComponent` для можливості вийти з облікового запису.

Доступність:

- додано атрибут `aria-label` до навігаційної панелі для покращення доступності [29];
- використано роль `menuitem` для кожного посилання у навігаційному списку.

Цей компонент відображається як зручний та добре структурований інтерфейс для користувачів, а елементи доступності спрямовані на те, щоб забезпечити зручність використання для всіх користувачів, включаючи тих, що використовують технології адаптивного введення чи екранні читачі. Зображення компоненту наведено на рисунку (рис. 3.2).

Вітаю, Liuda Karapet
Ви увійшли як вчитель

Щоденник

Розклад
Уроки

Вихід

Оберіть клас * Оберіть предмет Оберіть місяць

Створити щоденник Очистити щоденник

Учні	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Оксана Коваленко																														
Віталій Шевченко																														
Надія Коваленко																														

Рис. 3.2 Навігаційна панель у навчальній платформі

3.1.3 Розробка та впровадження Щоденника

Цей компонент представляє собою сторінку «Щоденника» додатка і має ряд функціональностей, що стосуються завантаження даних, взаємодії з користувачем та відображення контенту. Давайте розглянемо ключові елементи компонента з особливою увагою на аспекти доступності:

- заголовок сторінки: використано `useEffect` для динамічного встановлення заголовку сторінки («Щоденник»). Це дозволяє користувачам, використовуючи асистивні технології, отримувати інформацію про поточну сторінку [30];

- завантаження даних: використано `loader` для завантаження даних перед рендерингом компонента. Запит на сервер включає отримання даних про уроки за певний період та клас;

- взаємодія з користувачем: використано `SchoolDiaryToolbar` для відображення інтерфейсу вибору класу та періоду. Також визначено `action` для обробки взаємодії з формами, зокрема, додавання, оновлення та видалення оцінок;

- таблиця уроків: використано компонент `DynamicTable` для відображення динамічної таблиці з уроками та оцінками. Також додається статичне поле «Учні», а заголовки колонок представлені числами від 1 до 30 (залежно від кількості днів у місяці) [31];

- доступність: задано відповідний заголовок для сторінки та деяких елементів інтерфейсу. Використано `aria-label` та атрибут `role` для поліпшення доступності елементів інтерфейсу [32];

- стандартне відображення: якщо дані ще не завантажені, відображається стандартний вигляд таблиці із заголовком та списком учнів;

- застосування стилів: використано стилізацію для поліпшення візуального сприйняття та забезпечення коректного відображення на різних пристроях.

Цей компонент добре структурований та має високий рівень доступності, забезпечуючи зручність використання для різних категорій користувачів.

Реалізація маршруту school-diary

```

import { SchoolDiaryToolbar } from "~/components/schoolDiaryToolbar";
import { json, type ActionFunction, type LoaderFunction } from
"@remix-run/node";
import { useLoaderData, useRouteLoaderData } from "@remix-run/react";
import { getClassWithStudents } from "~/utils/classroom.server";
import { getUserIdAndRole } from "~/utils/auth.server";
import { getLessonsByPeriodAndClass } from "~/utils/lessons.server";
import { getDaysInMonth, getFullMonthStartEndDays } from
~/helpers/timeConverter";
import DynamicTable from "~/components/DynamicTable";
import type { IClassroomWithStudents, ILessonWithMarks } from
~/types/project.types";
import { deleteMarkById, tCreateMark1, tUpdateMark } from
~/utils/marks.server";
import { useEffect, Fragment } from "react";
import { getUserByIdAndRole } from "~/utils/user.server";
import { Student } from "@prisma/client";

export const loader: LoaderFunction = async ({ request }) => {
  const { userId, userRole } = await getUserIdAndRole(request);
  if (!userId)
    throw new Response("Unauthorized", {
      status: 401,
    });

  let classroom;
  const url = new URL(request.url);

  if (userRole === "student") {
    const user = await getUserByIdAndRole(userId, userRole);
    const { classroomId } = user as Student;
    classroom = classroomId;
  } else {
    classroom = url.searchParams.get("class");
  }

  const period = url.searchParams.get("period");
  const lessonType = url.searchParams.get("lesson_type");

  if (classroom && period && lessonType) {
    const { firstDay, lastDay } = getFullMonthStartEndDays(period);
    const lessons = await getLessonsByPeriodAndClass(classroom, {
      firstDay,
      lastDay,
    });
  }
}

```

```

    }, lessonType);

    const classWithStudents = await getClassWithStudents(classroom);
    return json({ period, lessons, classWithStudents });
  }
  return null;
};

export const action: ActionFunction = async ({ request }) => {
  const form = await request.formData();
  const intent = form.get("intent");

  if (intent === "createMark") {
    const mark_valueS = form.get("mark_value");
    const mark_value = Number(mark_valueS);
    const mark_teacher_id = form.get("m_teacher_id");
    const mark_student_id = form.get("m_student_id");
    const mark_lesson_id = form.get("m_lesson_id");

    if (
      !mark_teacher_id ||
      typeof mark_teacher_id !== "string" ||
      !mark_student_id ||
      typeof mark_student_id !== "string" ||
      !mark_lesson_id ||
      typeof mark_lesson_id !== "string"
    ) {
      return json({ error: "Invalid formData" }, { status: 400 });
    }

    return await tCreateMark1(
      mark_value,
      mark_teacher_id,
      mark_student_id,
      mark_lesson_id
    );
  } else if (intent === "updateMark") {
    const mark_valueS = form.get("mark_value");
    const mark_value = Number(mark_valueS);
    const mark_id = form.get("m_mark_id");

    if (!mark_id || typeof mark_id !== "string") {
      return json({ error: "Invalid formData" }, { status: 400 });
    }

    return await tUpdateMark(mark_id, mark_value);
  }
};

```

```

} else if (intent === "deleteMark") {
  const mark_id = form.get("m_mark_id");

  if (!mark_id || typeof mark_id !== "string") {
    return json({ error: "Invalid formData" }, { status: 400 });
  }
  return await deleteMarkById(mark_id);
}
};

export default function SchoolDiary() {
  useEffect(() => {
    document.title = 'Щоденник';
  });

  const loaderData = useLoaderData();
  const { user, allClasses } = useRouteLoaderData("routes/home");
  const period: string = loaderData?.period || null;
  const lessons: ILessonWithMarks[] = loaderData?.lessons || null;
  const classWithStudents: IClassroomWithStudents =
    loaderData?.classWithStudents || null;

  const daysInMonth = getDaysInMonth(period);
  const columnsDefault = Array.from({ length: 30 }, (_, index) => index
+ 1);

  return (
    <Fragment>
      <div>
        <h1 className="mb-8 text-4xl font-extrabold leading-none
tracking-tight text-gray-900 md:text-5xl lg:text-6xl">
          Щоденник
        </h1>
        <SchoolDiaryToolbar classes={allClasses} userType={user.type} />

        {loaderData ? (
          <DynamicTable
            columnCount={daysInMonth || 0}
            classWithStudents={classWithStudents}
            lessons={lessons}
            userType={user.type}
          />
        ) : (
          <div className="w-full overflow-x-scroll" id="pseudo_table">
            <div className="rounded-b-lg">

```

```

<table className="w-full border-collapse border bg-white">
  <thead>
    <tr className="border">
      <th className="border">Учні</th>
      {columnsDefault.map((header, index) => (
        <th key={index} className="border min-w-[25px]">
          {header}
        </th>
      ))}
    </tr>
  </thead>
  <tbody>
    <tr className="border">
      <th>Оксана Коваленко</th>
    </tr>
    <tr className="border">
      <th>Віталій Шевченко</th>
    </tr>
    <tr className="border">
      <th>Надія Коваленко</th>
    </tr>
  </tbody>
</table>
</div>
</div>
)}
</div>
</Fragment>
);
}

```

Компонент DynamicTable використовується для створення динамічної таблиці оцінок учнів з урахуванням ключових аспектів.

Організація компонента:

- використані бібліотеки React, Fragment, useEffect та useState для створення компонента [30];
- імпортовано Modal та MarkForm для використання в модальному вікні.

Визначення пропсів:

- інтерфейс columnCountProps визначає типи пропсів, які отримує компонент;

- за допомогою пропсів передається кількість стовпців, дані про учнів та оцінки уроків.

Ініціалізація та стан:

- створені змінні та стани для управління даними та станом модального вікна.

Ефекти та завантаження даних:

- використано `useEffect` для завантаження даних про уроки після рендерингу компонента.

Взаємодія з користувачем:

- задана функція `handleClick` для обробки кліків та відкриття/закриття модального вікна;
- обмежено функціонал для користувачів типу «student».

Побудова таблиці:

- створено масив `columns` для визначення стовпців таблиці;
- створено динамічну таблицю, яка відображає дані про учнів та оцінки уроків.

Доступність:

- використано `aria-label`, атрибути `role` та інші засоби для поліпшення доступності елементів інтерфейсу [31].

Модальне вікно:

- використано `Modal` та `MarkForm` для створення модального вікна для введення оцінок.

Стилізація та відображення:

- використано CSS класи для стилізації та визначення вигляду таблиці та елементів.

Додатковий функціонал:

- визначено можливість виділення рядка та комірки в таблиці;
- передбачено можливість вибору уроку для виставлення оцінок через модальне вікно.

Доступність клавiш «Tab»:

- ця таблиця містить кнопку у кожній комірці, яка дозволяє користувачам переходити до комірки за допомогою клавiші «Tab» для поліпшення навігації.

Лістинг 3.6

Реалізація компоненту DynamicTable

```
import React, { Fragment, useEffect, useState } from "react";
import { Modal } from "../modal";
import { MarkForm } from "../MarkForm";
import {
  findMarkOfLessonDateAndStudentId,
  formColumnsByLessons,
  getAttributtes,
  isLessonInThisDate,
} from "~/helpers/school-dairy-helpers";
import type {
  DaysWithLessons,
  IClassroomWithStudents,
  ILessonWithMarks,
} from "~/types/project.types";
import { Role } from "@prisma/client";

interface columnCountProps {
  columnCount: number;
  classWithStudents: IClassroomWithStudents;
  lessons: ILessonWithMarks[];
  userType: Role;
}

export default function DymanicTable({
  columnCount,
  classWithStudents,
  lessons,
  userType,
}: columnCountProps) {
  let columns = Array.from(
    { length: columnCount },
    (_, index: number) => index + 1
  );

  // todo - only for 1 Lesson type !!!!!!!! (+ connected to teacher)
  const [daysWithLessons, setDaysWithLessons] =
    useState<DaysWithLessons | []>(
      []
    )
}
```

```

);
const [isOpenModal, setIsOpenModal] = useState<boolean>(false);
const [lessonId, setLessonId] = useState<string>("");
const [studentId, setStudentId] = useState<string>("");
const [teacherId, setTeacherId] = useState<string>("");
const [markId, setMarkId] = useState<string>("");

const [selectedCell, setSelectedCell] = useState<{
  row: number;
  column: number;
}>({ row: 0, column: 0 });

useEffect(() => {
  setDaysWithLessons(() => formColumnsByLessons(lessons));
}, [lessons]);

const handleClick = (e: React.SyntheticEvent<EventTarget>) => {
  if (userType === "student") return;
  setIsOpenModal(!isOpenModal);
  if (!e?.target || !(e.target instanceof HTMLButtonElement))
return;
  if (
    e.target.dataset.lessonId &&
    e.target.dataset.studentId &&
    e.target.dataset.teacherId
  ) {
    setLessonId(e.target.dataset.lessonId);
    setStudentId(e.target.dataset.studentId);
    setTeacherId(e.target.dataset.teacherId);
  } else {
    setLessonId("");
    setStudentId("");
    setTeacherId("");
  }

  if (e.target.dataset.markId) {
    setMarkId(e.target.dataset.markId);
  } else {
    setMarkId("");
  }
};

const handleCloseModal = () => {
  setIsOpenModal(false);
};

```

```

return (
  <Fragment>
    <div className="rounded-b-lg w-full overflow-x-scroll"
id="table-wrapper">
      <React.Fragment>
        <table className="w-full border-collapse border-2
border-black bg-white">
          <thead>
            <tr className="border-2 border-black">
              <th className="border-2 border-black">Учні</th>
              {columns.map((column) => (
                <th
                  className="border-2 border-black min-w-[25px]"
                  key={column}
                >
                  {column}
                </th>
              ))}
            </tr>
          </thead>
          <tbody>
            {classWithStudents &&
              classWithStudents?.students?.length > 0 &&
              classWithStudents.students.map((e, rowIndex) => (
                <tr
                  key={e.id}
                  className={`border-2 border-black ${
                    rowIndex === selectedCell.row ? "selected-row" :
""
                  }`}
                >
                  <th>
                    {e.profile.firstName} {e.profile.lastName}
                  </th>
                  {columns.map((column, columnIndex) => (
                    <td
                      className={`border-2 border-black h-[2rem]
hover:outline-2 hover:outline-blue-500 ${
                        rowIndex === selectedCell.row &&
                        columnIndex === selectedCell.column
                          ? "selected-cell"
                          : ""
                      }`}
                      key={columnIndex}
                    >
                      {daysWithLessons &&

```

```

isLessonInThisDate(column, daysWithLessons)
&& (
    <button
      type="button"
      onClick={ (e) => {
        handleClick(e);
      }}
      className="block w-full h-full
bg-yellow-100"
      data-teacher-id={
        getAttributtes(column,
daysWithLessons)
          ?.teacherId
      }
      data-student-id={e.id}
      data-lesson-id={
        getAttributtes(column,
daysWithLessons)
          ?.lessonId
      }
      data-mark-id={
        findMarkOfLessonDateAndStudentId(
          column,
          e.id,
          daysWithLessons
        )?.id
      }>
      {findMarkOfLessonDateAndStudentId(
        column,
        e.id,
        daysWithLessons
      )?.value
        ? findMarkOfLessonDateAndStudentId(
          column,
          e.id,
          daysWithLessons
        )?.value
        : "+"}
    </button>
  )}
</td>
  )}
</tr>
)}
</tbody>
<Modal
  isOpenModal={isOpenModal} handleClick={handleClick}

```

```

        className="w-2/3 p-10"
      >
        <MarkForm
          lessonId={lessonId}
          studentId={studentId}
          teacherId={teacherId}
          markId={markId}
          onCloseModal={handleCloseModal}
        />
      </Modal>
    </table>
  </React.Fragment>
</div>
</Fragment>
) ;
}

```

На рисунку (рис. 3.3) зображено вигляд щоденника у платформі для навчання з обмеженими можливостями.

Вітаю, Liuda Karapet
Ви увійшли як вчитель

Щоденник

Оберіть клас
 5-B

* Оберіть предмет
 Українська

Оберіть місяць
 November 2023

Створити щоденник

Очистити щоденник

Учні	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Vasia Pupkin	+						+	4	+	+			4	4	+	5	+										+			
Петро Щур	+						+	+	+	+			10	+	+	+	4										+			
Таня Вакуленко	+						+	+	+	+			+	+	+	+	+										+			

Рис. 3.3 Відображення щоденника на навчальній платформі

3.1.4 Реалізація та оптимізація Календаря

Цей код представляє компонент `Schedule`, який відповідає за відображення та управління розкладом уроків. Давайте розглянемо його ключові аспекти:

`LoaderFunction`:

- компонент використовує `LoaderFunction` для завантаження даних перед відображенням сторінки [32];
- здійснює запит до сервера за допомогою `getUserIdAndRole`, отримуючи ID користувача та його роль [32];
- якщо користувач не авторизований (ID відсутній), викидає виняток зі статусом 401;
- повертає дані про ID та роль користувача.

`ActionFunction`:

- відповідає за обробку дій користувача, таких як створення нового уроку;
- отримує дані форми за допомогою `request.formData()` [33];
- перевіряє обов'язкові поля форми (`lesson_type`, `lesson_topic`, `start_time`, `end_time`, `teacher_id`, `class_id`);
- перевіряє валідність типів даних переданої інформації;
- викликає функцію `createLesson`, яка створює новий урок за наданими даними.

React-компонент `Schedule`:

- використовує `useEffect` для встановлення заголовку сторінки («Розклад»);
- використовує `useLoaderData` для отримання даних, завантажених за допомогою `LoaderFunction` [32];
- отримує ID та роль користувача, а також дані про класи та користувача з маршруту;

- використовує стан (selectedClass), який зберігає вибраний клас користувачем;
- виводить заголовок «Розклад» та, якщо користувач - вчитель, виводить випадаючий список DropdownMenu для вибору класу;
- виводить календар Calendar, передаючи йому необхідні дані, такі як роль користувача, вибраний клас, ID вчителя, дані форми та профіль користувача.

Цей компонент створений для надання інтерфейсу та функціоналу для перегляду та управління розкладом уроків, а також для створення нових уроків користувачами з відповідними правами.

Лістинг 3.7

Реалізація маршруту schedule

```
import { ActionFunction, LoaderFunction, json } from "@remix-run/node";
import {
  useActionData,
  useLoaderData,
  useRouteLoaderData,
} from "@remix-run/react";
import { useEffect, useState } from "react";
import { Calendar } from "~/components/Calendar";
import { DropdownMenu } from "~/components/dropDownMenu";
import { formOptionsFromArray } from "~/helpers/formHelpers";
import { getUserIdAndRole } from "~/utils/auth.server";
import { createLesson } from "~/utils/lessons.server";

export const loader: LoaderFunction = async ({ request }) => {
  const { userId, userRole } = await getUserIdAndRole(request);
  if (!userId)
    throw new Response("Unauthorized", {
      status: 401,
    });
  return { userId, userRole };
};

export const action: ActionFunction = async ({ request }) => {
  const form = await request.formData();
  const obj = Object.fromEntries(form.entries());
  if (!obj) return;
  const {
    lesson_type,
```

```

    lesson_topic,
    lesson_description,
    lesson_location,
    start_time,
    end_time,
    teacher_id,
    class_id,
  } = obj;
  if (
    !lesson_type ||
    !lesson_topic ||
    !start_time ||
    !end_time ||
    !teacher_id ||
    !class_id
  )
    return json({ error: "Required fields are missing" }, { status: 400
});

// todo error handling

  if (
    typeof lesson_type !== "string" ||
    typeof lesson_topic !== "string" ||
    typeof lesson_description !== "string" ||
    typeof lesson_location !== "string" ||
    typeof start_time !== "string" ||
    typeof end_time !== "string" ||
    typeof teacher_id !== "string" ||
    typeof class_id !== "string"
  )
    return json({ error: "Invalid Form Data" }, { status: 400 });
  // todo - ts error

  return await createLesson({
    type: lesson_type,
    topic: lesson_topic,
    description: lesson_description,
    startTime: start_time,
    endTime: end_time,
    teacherId: teacher_id,
    classroomId: class_id,
  });
};

```

```

export default function Schedule() {
  useEffect(() => {
    document.title = 'Розклад';
  });
  const { userId, userRole } = useLoaderData();
  const { allClasses, user } = useRouteLoaderData("routes/home");
  const [selectedClass, setSelectedClass] = useState();
  const actionData = useActionData();

  useEffect(() => {
    if (userRole === "teacher") return;
    setSelectedClass(user?.classroomId);
  }, [])
  return (
    <div>
      <h1 className="mb-8 text-4xl font-extrabold leading-none tracking-tight text-gray-900 md:text-5xl lg:text-6xl">
        Розклад
      </h1>

      {userRole === "teacher" && (
        <DropdownMenu
          hasEmptyOption={true}
          emptyOptionTitle="---"
          options={formOptionsFromArray(allClasses, "id", "name")}
          label="Оберіть клас"
          ariaLabel="Оберіть клас"
          ariaRequired={true}
          name="class"
          inputValue={selectedClass}
          onChange={e => {
            setSelectedClass(e.target.value);
          }}
          controlled={true}
        />
      )}
      <div className="bg-white">
        <Calendar
          userRole={userRole}
          selectedClass={selectedClass}
          teacherId={userId}
          actionData={actionData}
          profile={user?.profile}
        />
      </div>
    </div>
  )
}

```

```

</div>
);
}

```

Компонент Calendar, який використовує бібліотеку FullCalendar для відображення та управління календарем уроків. Розглянемо його ключові аспекти:

Імпорт бібліотек та компонентів:

- імпортується FullCalendar та деякі з його плагінів для різних видів відображення календаря [32];
- використовуються інші бібліотеки та компоненти, такі як useFetcher, useEffect, useRef, useState, Modal, і CreateLessonModal.

Ініціалізація станів та змінних:

- створюються стани та змінні для відстеження статусу завантаження, відкриття/закриття модального вікна, часу початку та завершення уроку;
- використовується useRef для отримання посилання на екземпляр календаря.

Використання useEffect:

- здійснюється виклик ефектів при зміні actionData (даних, які отримуються через LoaderFunction);
- моніторить стан fetcher (використовується для відправки запитів на сервер), реагує на зміни та виводить повідомлення у консоль.

Обробники подій календаря:

- eventChangeHandler та eventRemoveHandler реагують на зміни та видалення уроків у календарі. Вони викликають функцію submit з fetcher для взаємодії із сервером;
- handleEventClick обробляє кліки на подіях календаря та, якщо вчитель підтверджує видалення, видаляє подію.

Інтерфейс користувача:

- виводиться модальне вікно CreateLessonModal, яке містить форму для створення нового уроку, коли користувач клікає на певний день у календарі;
- виводить індикатор завантаження, який з'являється під час завантаження даних.

FullCalendar :

- використовує FullCalendar для відображення та інтерактивного управління календарем [32];
- взаємодіє з сервером для отримання та оновлення даних уроків;
- має різні налаштування, такі як види періодів, формат часу, та розташування.

Модальне вікно:

- використовується модальне вікно для відображення форми створення уроку (CreateLessonModal).

Лістинг 3.8

Реалізація компоненту Calendar

```
import FullCalendar from "@fullcalendar/react";
import dayGridPlugin from "@fullcalendar/daygrid";
import interactionPlugin from "@fullcalendar/interaction";
import timeGridPlugin from "@fullcalendar/timegrid";
import { useFetcher } from "@remix-run/react";
import { useEffect, useRef, useState } from "react";
import { Modal } from "../modal";
import CreateLessonModal from "../CreateLessonModal";
import { localWithTZtoIsoString } from "~/helpers/timeConvertor";
import {
  convertToCalendarFormatSingle
} from "~/helpers/calendarHelpers";
import ukLocale from "@fullcalendar/core/locales/uk"
export const Calendar = ({
  selectedClass,
  teacherId,
  actionData,
  profile,
  userRole,
}) => {
  const fetcher = useFetcher();
  const calendarRef = useRef(null);
  const [loading, setLoading] = useState(false);
  const [isOpenModal, setIsOpenModal] = useState<boolean>(false);
```

```

const [startTime, setStartTime] = useState<string>("");
const [endTime, setEndTime] = useState<string>("");

useEffect(() => {
  if (!actionData) return;
  const calendarNewEvent = convertToCalendarFormatSingle(
    actionData,
    teacherId,
    profile
  );
  calendarRef.current.calendar.addEvent(calendarNewEvent);
}, [actionData]);

useEffect(() => {
  const isLoading =
    fetcher.state === "loading" || fetcher.state === "submitting";
  const isDone = fetcher.state === "idle" && fetcher.data !== null;

  if (isLoading) {
    console.log("loading...");
    setLoading(true);
  } else {
    setLoading(false);
  }

  if (isDone) {
    console.log("change successfull", fetcher.data);
  }
}, [fetcher]);

const eventChangeHandler = async (e) => {
  const { oldEvent, event, revert } = e;
  const newStart = localWithTZtoIsoString(event.startStr);
  const newEnd = localWithTZtoIsoString(event.endStr);
  const formData = new FormData();
  formData.append("intent", "update");
  formData.append("id", oldEvent.id);
  formData.append("start", newStart);
  formData.append("end", newEnd);
  console.log(formData);
  fetcher.submit(formData, {
    method: "PATCH",
    action: "/service/lessons-actions",
  });
};

```

```

const eventRemoveHandler = async (e) => {
  const { event, revert } = e;
  const formData = new FormData();
  formData.append("intent", "delete");
  formData.append("id", event.id);
  fetcher.submit(formData, {
    method: "DELETE",
    action: "/service/lessons-actions",
  });
};

const handleEventClick = (e) => {
  const eventOwnerId = e.event.extendedProps?.serviceData?.ownerTeacherId;
  if (!eventOwnerId || teacherId !== eventOwnerId) return;

  if (
    confirm(`Ви дійсно хочете видалити '${e.event.title}'`)
  ) {
    e.event.remove();
  }
};

const handleDateSelect = (selectArgs: any) => {
  console.log("DataSelect", selectArgs);

  const startInISO = selectArgs.start.toISOString();
  const endInISO = selectArgs.end.toISOString();
  setStartTime(startInISO);
  setEndTime(endInISO);

  setIsOpenModal(!isOpenModal);

  let calendarApi = selectArgs.view.calendar;
};

const handleLoading = () => {
  console.log("loading calendar");
};

const handleCloseModal = () => {
  setIsOpenModal(false);
};

return (
  <div className="index-route">
    <Modal
      isOpenModal={isOpenModal}
      handleClick={() => {

```

```

        setIsOpenModal(!isOpenModal);
    }}
    className="w-2/3 p-10">
    <CreateLessonModal
        teacherId={teacherId}
        classId={selectedClass}
        startTime={startTime}
        endTime={endTime}
        onCloseModal={handleCloseModal}
    />
</Modal>
{loading && <div className="absolute">loading...</div>}
<FullCalendar
    ref={calendarRef}
    slotDuration="00:15:00"
    slotLabelInterval="00:15"
    slotEventOverlap={false}
    weekends={false}
    locales={[ukLocale]}
    locale="uk"
    plugins={[dayGridPlugin, timeGridPlugin, interactionPlugin]}
    events={selectedClass ? `/service/lessons/${selectedClass}` :
        ""}
    selectable={userRole === "teacher" ? true : false}
    editable={false}
    eventOverlap={false}
    select={handleDateSelect}
    eventClick={handleEventClick}
    lazyFetching={true}
    loading={handleLoading}
    initialView="dayGridMonth"
    headerToolbar={{
        left: "prev,next today",
        center: "title",
        right: "dayGridMonth,timeGridWeek,timeGridDay",
    }}
    nowIndicator={true}
    firstDay={1}
    navLinks={true}
    eventTimeFormat={{
        hour: "2-digit",
        minute: "2-digit",
        hour12: false,
    }}
    slotLabelFormat={{
        hour: "2-digit",
        minute: "2-digit",
    }}

```

```

        hour12: false,
      })
      slotMinTime="07:00:00"
      slotMaxTime="18:00:00"
      eventChange={eventChangeHandler}
      eventRemove={eventRemoveHandler}
    />
  </div>
);

```

Цей компонент дозволяє вчителям та іншим користувачам відслідковувати та змінювати розклад уроків, а також створювати нові уроки за допомогою взаємодії з календарем.

Ось такий має вигляд календар з вибраним параметром «Місяць» (рис 3.4).



Рис. 3.4 Календар з вибраним парметром «Місяць»

3.1.5 Розділ уроки: Розробка та Впровадження

Цей компонент представляє собою сторінку для відображення розкладу занять на підставі вибраних параметрів, таких як клас, період та тип уроку. Давайте розглянемо ключові аспекти цього компонента:

LoaderFunction:

- компонент використовує LoaderFunction для завантаження даних перед відображенням сторінки [33];
- здійснює запит до сервера для отримання інформації про користувача (ID та роль);
- перевіряє, чи користувач авторизований. Якщо ні, повертає null;
- отримує параметри з URL (class, period, lesson_type) та викликає функцію для отримання уроків за вказаними параметрами [33];
- повертає дані у форматі JSON, які використовуються в компоненті під час рендерингу.

React-компонент:

- використовує useEffect для встановлення заголовку сторінки («Предмети»);
- використовує useLoaderData для отримання даних, завантажених за допомогою LoaderFunction [33];
- отримує дані про період, кількість днів у місяці та масив уроків.

Sorting Lessons:

- сортує уроки за часом початку за зростанням;
- використовує об'єкт Lesson для представлення кожного уроку з необхідною інформацією.

Відображення інтерфейсу:

- виводить заголовок «Предмети» та панель інструментів для вибору параметрів [33];
- за допомогою SchoolLessonsToolbar надає можливість вибрати клас;
- по суті, виводить уроки у вигляді карточок ClassroomSession;
- виводить повідомлення «Оберіть параметри», якщо дані не завантажено.

Реалізація маршруту lessons

```

import {
  json,
  type LoaderFunction,
} from "@remix-run/node";
import { Form, useLoaderData, useRouteLoaderData } from
"@remix-run/react";
import { useEffect, Fragment } from "react";
import ClassroomSession from "~/components/ClassroomSession";
import { SchoolLessonsToolbar } from
~/components/schoolLessonsToolBar";
import { getDaysInMonth, getFullMonthStartEndDays } from
~/helpers/timeConverter";
import { Lessons } from "~/types/project.types";
import { getUserId } from "~/utils/auth.server";
import { getLessonsByParameters } from "~/utils/lessons.server";

export const loader: LoaderFunction = async ({ request }) => {
  const userId = await getUserId(request);
  if (!userId) {
    return null;
  }
  const url = new URL(request.url);
  const classroom = url.searchParams.get("class");
  const period = url.searchParams.get("period");
  const lessonType = url.searchParams.get("lesson_type");

  if (classroom && period && lessonType) {
    const { firstDay, lastDay } = getFullMonthStartEndDays(period);
    const lessons = await getLessonsByParameters(
      classroom,
      {
        firstDay,
        lastDay,
      },
      userId,
      lessonType
    );

    return json({ period, lessons });
  }
  return null;
};

export default function Lessons() {
  useEffect(() => {

```

```

    document.title = 'Предмети';
  });

  const loaderData = useLoaderData();

  const period: string = loaderData?.period || null;
  const daysInMonth = getDaysInMonth(period);

  const lessons: Lessons[] = loaderData?.lessons || null;
  const { allClasses } = useRouteLoaderData("routes/home");

  let sortedLessons;

  if( lessons && lessons.length > 1){
    sortedLessons = lessons.slice().sort((a, b) => {
      const startTimeA = new Date(a.startTime);
      const startTimeB = new Date(b.startTime);

      if (startTimeA < startTimeB) {
        return -1;
      } else if (startTimeA > startTimeB) {
        return 1;
      }

      return 0;
    });
  } else {
    sortedLessons = lessons;
  }
  return (
    <Fragment>
      <h1 className="mb-8 text-4xl font-extrabold leading-none tracking-tight text-gray-900 md:text-5xl lg:text-6xl">
        Предмети
      </h1>

      <SchoolLessonsToolbar classes={allClasses} />
      {loaderData ? (
        <div>
          {sortedLessons.map((el, index) => (
            <ClassroomSession
              key={el.id}
              index={index + 1}
              type={el.type}

```

```

        startDate={el.startTime.substring(0, 10)}
        endDate={el.endTime.substring(0, 10)}
        startTime={el.startTime.substring(11, 16)}
        endTime={el.endTime.substring(11, 16)}
        location={el.location}
        topic={el.topic}
        description={el.description}
      />
    ))}
  </div>
) : (
  <div>
    <h1>Оберіть параметри</h1>
  </div>
) }
</Fragment>
);
}

```

Цей компонент створений для надання користувачам можливості перегляду розкладу уроків за обраними параметрами, забезпечуючи зручність навігації та відображення інформативного інтерфейсу. Вигляд компонента зображено на рисунку нижче (рис 3.5).

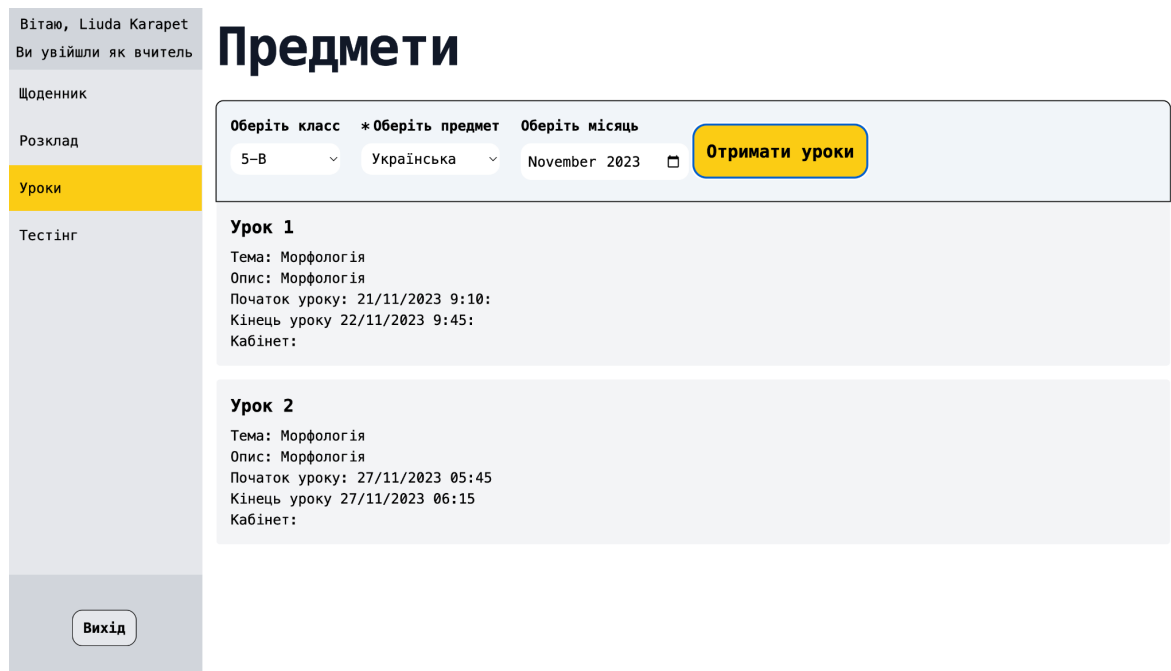


Рис. 3.5 Вигляд розділу предмети на навчальній платформі

3.2 Апробація та підтвердження наукової новизни навчальної платформи для людей з обмеженими можливостями

3.2.1 Оцінка за Lighthouse показнику доступності

Вдосконалення доступності та продуктивності нашої освітньої платформи відзначено високими балами в Lighthouse, який акцентує особливу увагу на аспектах доступності.

Розділ «Календар»: Найвищий рівень доступності, здобутий в розділі «Календар» з повним балом 100 за індексом Lighthouse, показано на рисунку (рис. 3.6) свідчить про виняткову оптимізацію. Це означає, що користувачі можуть легко та швидко взаємодіяти з розкладом та плануванням, отримуючи максимальний комфорт.

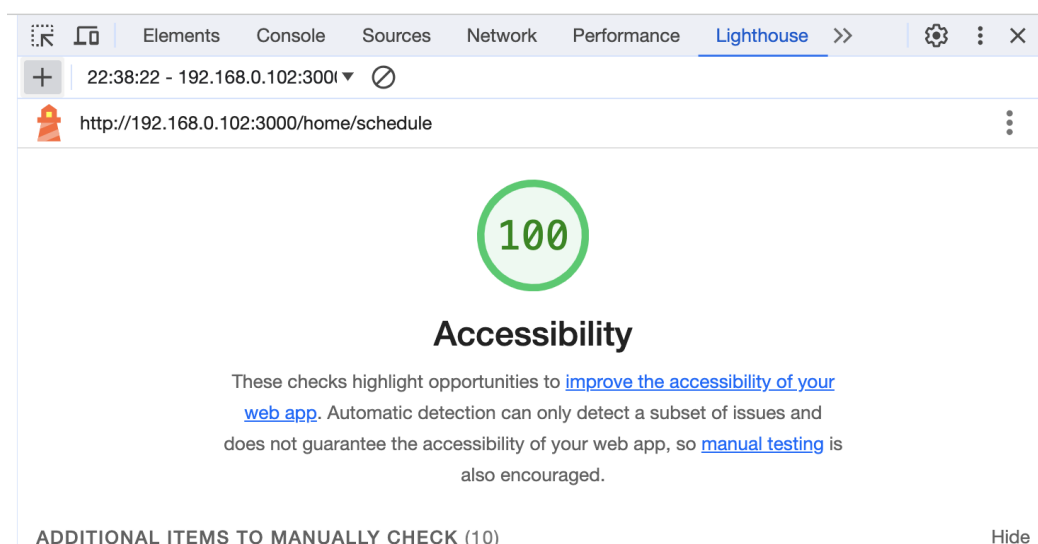


Рис. 3.6 Показник accessibility в розділі «Календар» в Lighthouse

Розділ «Щоденник»: Імпресивний рейтинг у 91 бал Lighthouse в розділі «Щоденник» свідчить про високий рівень вебдоступності інтерфейсу, оптимізованого для всіх користувачів. Рейтинг можна переглянути на рисунку (рис. 3.7).

Показник Lighthouse є не лише числовим відображенням, а й підтвердженням нашої мети забезпечити ефективну взаємодію вчителів та учнів із щоденником, забезпечуючи при цьому максимально широкий та

різноманітний доступ до цього розділу. Платформа не лише технічно досконала, але також враховує важливість людської взаємодії та різноманітності користувачів. Високий рейтинг Lighthouse є важливим підтвердженням того, що наш щоденник не обмежується лише функціональністю, але дійсно виділяється своєю винятковою доступністю та уважністю до потреб кожного, хто користується нашою платформою (рис. 3.7).

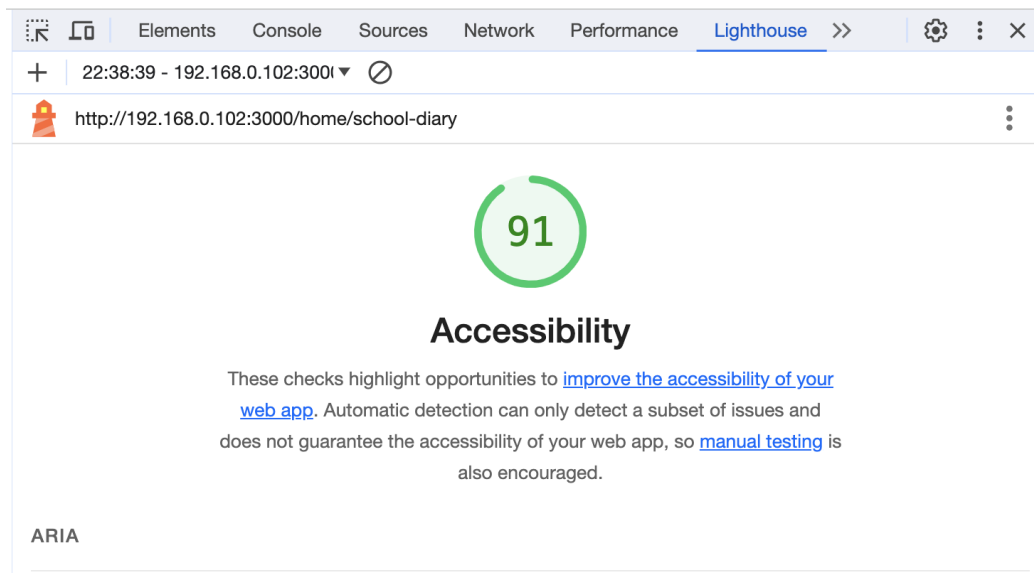


Рис. 3.7 Показник accessibility в розділі «Щоденник» в Lighthouse

Розділ «Уроки»: Незважаючи на трошки більш невисокий рейтинг у 90 балів за індексом Lighthouse в розділі «Уроки», цей показник підтверджує, що розділ «Уроки» відповідає всім стандартам і готовий до ефективного використання. Показник доступності у розділі показана на рисунку нижче (рис. 3.8).

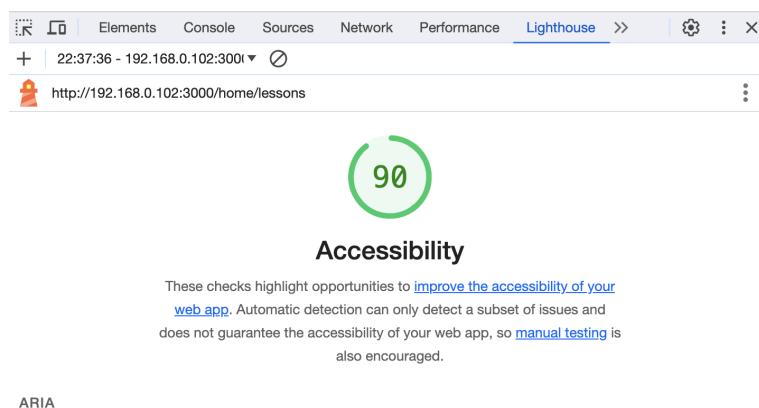


Рис. 3.8 Показник accessibility в розділі «Уроки» в Lighthouse

Підібрані підходи та методи аналізу доступності надають впевненість у правильності їх вибору. Платформа, маючи високий рівень доступності, демонструє, що обрані стратегії та методології виявилися ефективними.

Високі показники доступності за індексом Lighthouse служать важливим підтвердженням правильності підходу до забезпечення високого стандарту доступності освітньої платформи.

3.2.2 Збільшення конверсії з урахуванням вебдоступності

У ході застосування методів та практик вебдоступності, а також покращення механізмів та алгоритмів взаємодії з користувачами відбулось збільшення конверсії для користувачів із різними обмеженнями. Було використано різноманітні підходи та стратегії, спрямовані на створення відкритого та доступного вебсередовища для всіх користувачів.

Аудіо-спрямовані адаптації для слухових обмежень: важливим етапом стала імплементація аудіо-спрямованих адаптацій. Це включало в себе створення текстових підписів та детальних описів аудіо-контенту. Такий підхід сприяв покращенню взаємодії користувачів із слуховими обмеженнями, забезпечуючи їм повноцінний доступ до аудіо-інформації.

Оптимізація інтерфейсу для зорових обмежень: для користувачів із зоровими обмеженнями проводилася ретельна оптимізація інтерфейсу. Використання адаптивних шрифтів, контрастних кольорів та інших візуальних елементів сприяло створенню користувацького інтерфейсу, що легко сприймається та зручного для використання.

Пристосування для когнітивних обмежень: для поліпшення взаємодії з платформою для користувачів із когнітивними обмеженнями, вносилися зміни в структуру та організацію контенту. Особливий акцент робився на спрощенні інформаційного середовища для полегшення розуміння та сприяння зручності використання.

Оптимізація для моторних обмежень: для користувачів із моторними обмеженнями реалізовувалися зручні елементи управління та оптимізовані аспекти інтерфейсу. Це включало в себе поліпшення навігації та розміщення елементів для полегшення використання платформи.

Отримані графіки вказує на значуще підвищення конверсії для різних груп осіб з обмеженими можливостями, такими як вади слуху, зору, когнітивні та моторні вади. Це свідчить про успішність впроваджених змін у навчальній платформі. Результати апробації наголошують на тому, що впроваджені методи вебдоступності виявились ефективними у поліпшенні якості взаємодії та реакції платформи на потреби користувачів із різними обмеженнями (рис 3.9).

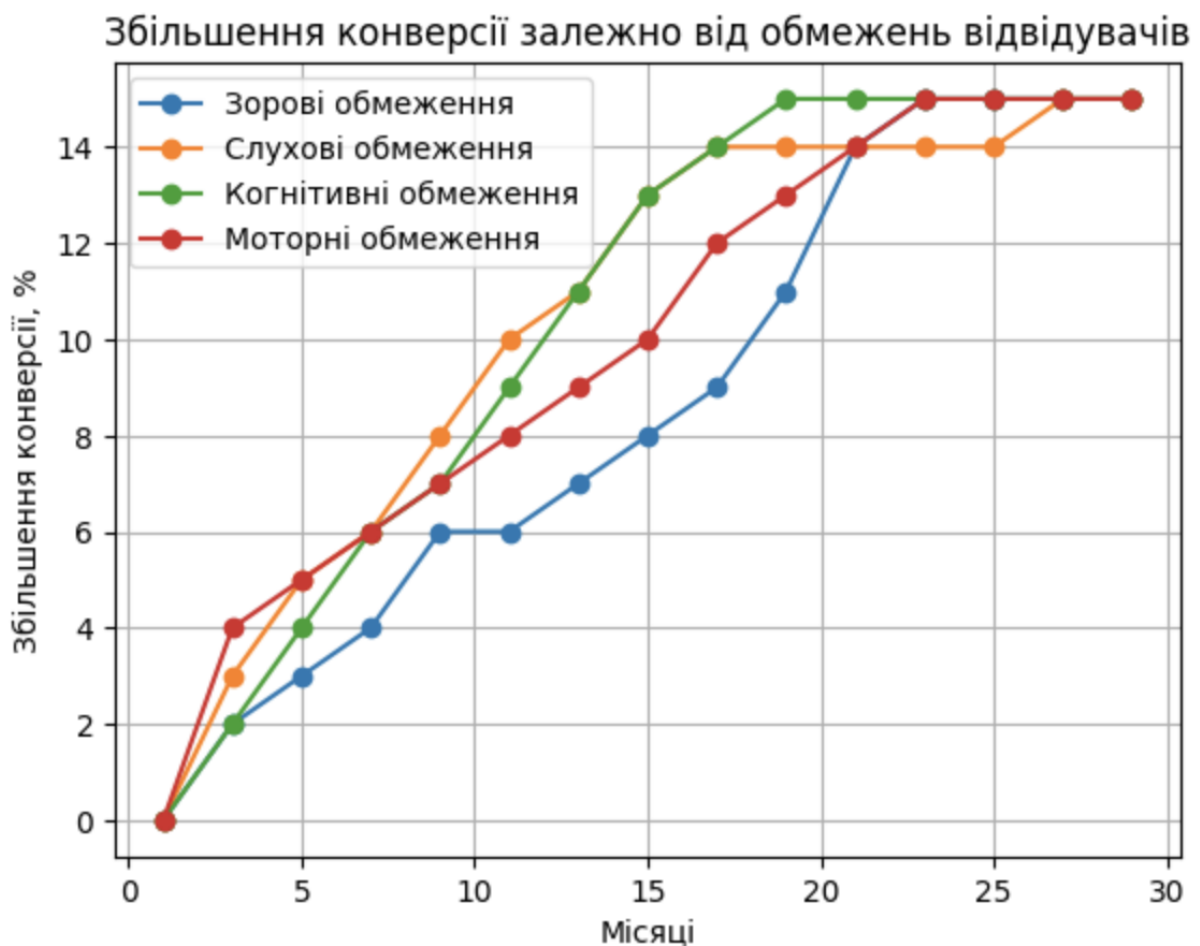


Рис 3.9 Збільшення конверсії залежно від обмежень відвідувачів

Висновки до розділу 3

У цьому розділі було впроваджено ряд ключових функціональностей, включаючи щоденник, календар, розділ із уроками, а також форми входу та реєстрації. Ці компоненти взаємодіють, утворюючи єдиний та компактний інструментарій, спрямований на покращення навчального процесу та оптимізацію взаємодії між учасниками платформи.

У щоденнику, вибравши необхідні параметри, можна отримати список учнів з бази даних, включаючи їхні оцінки. Крім того, є можливість додавати, редагувати та видаляти оцінки для відповідного дня та уроку, що надає зручний інструмент для ведення обліку успішності учнів.

Використання бібліотеки FullCalendar для створення календаря дозволило реалізувати потужний інструмент для відображення та управління розкладом уроків. У календарі реалізовано можливість додавання та видалення уроків. Календарний інтерфейс дозволяє переглядати уроки за різними періодами: на місяць, тиждень та день. Крім того, календарний перегляд уроків виключно обмежено переміщенням та редагуванням тільки власних уроків, забезпечуючи вчителям зручність в управлінні своїм графіком, а іншим користувачам - можливість лише перегляду розкладу.

Розроблений розділ уроки, дозволяє користувачам зручно переглядати розклад уроків за визначеними параметрами. Система сортування, інтерактивність та інтеграція з іншими компонентами роблять його цілком функціональним та корисним інструментом для вчителів та учнів.

Форми для входу та виходу обладнані кнопками, що дозволяють зручно перемикати режими між учителем та учнем. У цих формах реалізовані відповідні кнопки для швидкої зміни ролі користувача, забезпечуючи легку навігацію та зручний доступ до функціоналу для обох категорій користувачів.

Крім того, введені кнопки для перемикання між формами входу та реєстрації дозволяють користувачам з легкістю переходити від одного етапу до

іншого, що сприяє зручності в роботі з системою та ефективному використанню її можливостей.

Особлива увага приділялася питанням доступності, забезпечуючи, що навчальна платформа є доступною для всіх користувачів, незалежно від їхніх особливостей та потреб. На сайті впроваджено зручну навігацію за допомогою використання табів, що дозволяє користувачам легко переміщатись між різними секціями та функціональністю. При цьому враховано можливість збільшення масштабу, що гарантує адаптивність контенту до різних розмірів екрану, забезпечуючи комфортний перегляд для користувачів з різними вимогами щодо відображення.

В розробці сайту було використано семантичні теги та надані відповідні ролі для елементів, що допомагає покращити доступність та розуміння структури сторінки для асистивних технологій та користувачів з обмеженими можливостями.

Окрема увага приділялася створенню висококонтрастного дизайну, який сприяє кращій видимості та відокремленню елементів для користувачів з порушеннями зору. Активні елементи підсвічуються, щоб надати візуальний звіт про їхню активність.

Підходи та стратегії, спрямовані на поліпшення вебдоступності, виявились успішними, що підтверджують високі бали Lighthouse та позитивні результати апробації. Реалізація аудіо-спрямованих адаптацій, оптимізація інтерфейсу для зорових та моторних обмежень, а також адаптація для когнітивних обмежень сприяли створенню відкритого та доступного вебсередовища для всіх користувачів. Підвищення конверсії для різних груп осіб з обмеженими можливостями свідчить про успішність здійснених змін, які виявились ефективними у покращенні якості взаємодії та реакції платформи на потреби користувачів із різними обмеженнями.

ВИСНОВКИ

У ході дослідження та розробки платформи для навчання людей з обмеженими можливостями, було проведено комплексний розгляд наукових основ і методів у сфері вебдоступності. Детально розглянуто проблеми доступності в навчальних платформах та виявлено ефективні рішення, спрямовані на поліпшення доступу до освітніх ресурсів для всіх користувачів.

Аналіз існуючих аналогів з високою вебдоступністю в освітній сфері дозволив визначити передовий досвід та вдосконалення, які вдало використовуються для забезпечення рівних можливостей у навчанні. Враховуючи ці висновки, були сформульовані вимоги та потреби цільової аудиторії, спрямовані на максимально широкий охоплення користувачів з різними обмеженнями.

Проектування та архітектура рішення були спрямовані на створення інтуїтивного та доступного інтерфейсу. Реалізація маршрутів входу, реєстрації та виходу для режимів вчителя та учня, розробка та впровадження щоденника, оптимізація календаря, а також розділ уроків, враховують принципи вебдоступності.

Велика увага приділялась врахуванню доступності при збільшенні масштабу до 200%. Компоненти платформи динамічно змінюють своє розташування для полегшення сприйняття при збільшенні масштабу. Дизайн виконаний в контрастних кольорах, використовуються атрибути `aria label` та інші засоби для поліпшення взаємодії з платформою. При збільшенні екрану шрифт підлаштовується для зручності користувачів, а навігація табами забезпечує зручний перехід по всьому сайту.

У розробці цієї платформи використано передові технології, такі як Remix, React, TypeScript, Tailwind CSS, Prisma та MongoDB. Це забезпечує високу інтерактивність, надійність та зручний дизайн, а також швидке та ефективне взаємодію з базою даних.

Всі ці елементи разом формують інноваційний та інклюзивний інструмент для вчителів, спрямований на поліпшення навчального процесу та забезпечення рівних можливостей для всіх учнів. Цей проект не лише спрощує роботу вчителів, але й сприяє створенню освітнього середовища, яке враховує та відповідає потребам сучасного учня.

Всі ці елементи разом формують інноваційний та інклюзивний інструмент для вчителів, спрямований на поліпшення навчального процесу та забезпечення рівних можливостей для всіх учнів. Високі показники доступності за індексом Lighthouse та підвищення конверсії підтверджують успішність здійснених змін, які виявилися ефективними у покращенні якості взаємодії та реакції платформи на потреби користувачів із різними обмеженнями.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Карапет Л. В. Навчальна платформа для людей з обмеженими можливостями/ Р. С. Савіцький. // VI Всеукраїнська науково-технічна конференція «Комп'ютерні технології: інновації, проблеми, рішення» 29-30 листопада 2023 р. м. Житомир.
2. Романишин, Ю. Л. (2023). *Теоретичні і методичні засади проектування веббазованого освітнього середовища університету* (Doctoral dissertation, Інституті цифровізації освіти Національної академії педагогічних наук України).
3. Давиденко, Г. (2023). Цифрова інклюзія та доступність: соціальна діджиталізація.
4. Юзьвяк, А. М. (2022). Багатокритеріальний підхід до оцінювання юзабіліті вебсайту. *Збірник тез доповідей підготовлено за матеріалами Міжнародної наукової інтернет-конференції (випуск 71) 18-19 жовтня 2022 р. на сайті www.konferenciaonline.org.ua*, 112.
5. Кремень, В. Г., Биков, В. Ю., Ляшенко, О. І., Литвинова, С. Г., Луговий, В. І., Мальований, Ю. І., ... & Топузов, О. М. (2022). Науково-методичне забезпечення цифровізації освіти України: стан, проблеми, перспективи. *Вісник Національної академії педагогічних наук України*, 4(2), 1-49.
6. Атакулова, К. Ю. (2023). Дослідження методів проектування інтерфейсу вебсайтів для забезпечення доступності користувачам з обмеженими можливостями.
7. Прохоренко, Л., & Прохоренко, Д. (2023). Дистанційне навчання дітей з особливими потребами: створення моделі. *Освіта осіб з особливими потребами: шляхи розбудови.*, (22), 87-102.

8. Чеботарьова, Г., & Манічева, Н. (2023). Огляд методик дистанційного навчання у вищих учбових закладах в сучасних умовах. *Вісник науки та освіти*, (7 (13)).
9. Жмакін, А., Коваль, В., Любчак, В., & Шпіцглуз, С. (2022). Програмно-технічні рішення створення бюджетного варіанту комп'ютерних систем навчального та офісного призначення. *Інформаційні технології та суспільство*, (1 (3)), 23-30.
10. Пашкевич, А. С. (2023). *Програмний застосунок для вибору вибіркових дисциплін* (Bachelor's thesis, КПП ім. Ігоря Сікорського).
11. Wei, X., & Taecharungroj, V. (2022). How to improve learning experience in MOOCs an analysis of online reviews of business courses on Coursera. *The International Journal of Management Education*, 20(3), 100675.
12. Agarwal, A. (2022). Learning platforms: edX. *Executive education after the pandemic: A vision for the future*, 277-286.
13. Sharpe, S., & Young, G. (2023). Using Google Classroom as Assistive Technology in Universally Designed Classrooms. *Canadian Journal of Learning and Technology*, 49(1), 1-17
14. Шкуро, В. (2023, October). Врахування стандартів вебдоступності при розробленні навчальних онлайн-платформ [Considering Web Accessibility Standards When Developing Educational Online Platforms]. In Semigina, TV, Shkuro, VP, Novak, AS (2023). *Considering web accessibility standards into account when developing online educational platforms. Today's intellectual resource: scientific problems, development and questions: materials of the 1st International Scientific Conference* (pp. 69-73).
15. Georgakas, D. (2023). How Do People Perceive Content?. In *AIYY Unraveled: Become a Web Accessibility Ninja* (pp. 29-35). Berkeley, CA: Apress.
16. Матвієнко, В. О. (2022). *Імплементация стандартів вебдоступності при розробці інтерфейсів для користувачів з вадами зору* (Master's thesis, КПП ім. Ігоря Сікорського).

17. van Wee, B. (2022). Accessibility and equity: A conceptual framework and research agenda. *Journal of Transport Geography*, 104, 103421.
18. Ismail, A., & Kuppusamy, K. S. (2022). Web accessibility investigation and identification of major issues of higher education websites with statistical measures: A case study of college websites. *Journal of King Saud University-Computer and Information Sciences*, 34(3), 901-911.
19. Germano, R. S., & Silveira, I. F. (2022, June). WCAG-Easy Tool: A tool based in the WCAG to learn web accessibility. In *2022 17th Iberian Conference on Information Systems and Technologies (CISTI)* (pp. 1-6). IEEE.
20. Zuch, R. *Applying Navigability & Accessibility to a Self-Coded Digital Interactive Portfolio* (Master's thesis, SUNY Polytechnic Institute).
21. Савіцький, Р. С. (2023). Безбар'єрність вебпорталів освітніх навчальних закладів України. *Технічна інженерія*, (1(91), 172–177. [https://doi.org/10.26642/ten-2023-1\(91\)-172-177](https://doi.org/10.26642/ten-2023-1(91)-172-177)
22. Building remix applications [Електронний ресурс] – Режим доступу до ресурсу: <https://www.mongodb.com/developer/products/mongodb/building-remix-applications/>.
23. Fullstack Authentication with Remix using Prisma, MongoDB and Typescript [Електронний ресурс] – Режим доступу до ресурсу: https://dev.to/isnan_h/fullstack-authentication-with-remix-using-prisma-mongodb-and-typescript-c1e.
24. Accessibility Errors [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3schools.com/accessibility/accessibility_errors.php.
25. Accessibility Forms Introduction [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3schools.com/accessibility/accessibility_forms_intro.php.

26. Remix Client entry [Электронный ресурс] – Режим доступа до ресурсу: <https://remix.run/docs/en/main/file-conventions/entry.client>.
27. Accessibility Headings Introduction [Электронный ресурс] – Режим доступа до ресурсу: https://www.w3schools.com/accessibility/accessibility_headings_intro.php.
28. Remix React Router [Электронный ресурс] – Режим доступа до ресурсу: <https://remix.run/docs/en/main/discussion/react-router>.
29. Accessibility Semantic Elements [Электронный ресурс] – Режим доступа до ресурсу: https://www.w3schools.com/accessibility/accessibility_semantic_elements.php.
30. Accessibility Visual Focus [Электронный ресурс] – Режим доступа до ресурсу: https://w3schools.com/accessibility/accessibility_visual_focus.php.
31. Build A Fullstack App with Remix, Prisma & MongoDB: Project Setup [Электронный ресурс] – Режим доступа до ресурсу: <https://www.prisma.io/blog/fullstack-remix-prisma-mongodb-1-7D0BfTXBmB6r>
32. Big Calendar [Электронный ресурс] – Режим доступа до ресурсу: <https://jqense.github.io/react-big-calendar/examples/index.html?path=/story/about-big-calendar--page>.
33. Remix Blog Tutorial - Remix, React, Prisma, MongoDB, Vercel (Part 1) [Электронный ресурс] – Режим доступа до ресурсу: <https://dev.to/chrisbenjamin/remix-blog-tutorial-remix-react-prisma-mongodb-vercel-1hhb>.