МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет водного господарства та природокористування
Навчально-науковий інститут кібернетики,
інформаційних технологій та інженерії
Кафедра комп'ютерних технологій та економічної кібернетики

Допущено до захисту:

Завідувач кафедри комп'ютерних технологій та економічної кібернетики
д. е. н., проф. П. М. Грицюк

_____
«____» _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬО-КВАЛІФІКАЦІЙНОГО РІВНЯ «МАГІСТР»

Вебплатформа координації стейкхолдерів онлайн-репетиторства та комплексне тестування її функціональності

Виконала:
здобувачка вищої освіти за ОПП «Інформаційні технології в бізнесі» спеціальності 126 «Інформаційні системи та технології», групи ІТБ-61м
Сельвесюк Ірина Ігорівна

Керівник:
к.е.н., доцент Волошин В.С.

Рецензент:
к.т.н., доцент Василів В.Б.

Рівне – 2025

РЕФЕРАТ

Кваліфікаційна робота магістра: 59 с., 41 рис., 6 табл., 13 літературних джерел.

Актуальність теми полягає у розвитку онлайн-репетиторства як сучасної форми організації навчального процесу, що поєднує гнучкість дистанційного навчання з індивідуальним підходом до здобувачів освіти. Зростання попиту на персоналізовані освітні послуги та розвиток цифрових технологій зумовлюють потребу у вебплатформах, які забезпечують ефективну координацію стейкхолдерів освітнього процесу.

Об'єкт дослідження – процес організації взаємодії учасників онлайн-репетиторства.

Предметною областю є вебплатформа координації стейкхолдерів онлайн-репетиторства, що забезпечує комунікацію, бронювання занять та обмін інформацією між студентами та викладачами.

Метою магістерської роботи є проєктування, розробка та тестування вебплатформи координації стейкхолдерів онлайн-репетиторства.

У магістерській роботі визначено сутність онлайн-репетиторства та особливості взаємодії його стейкхолдерів, проведено аналіз існуючих платформ для організації онлайн-навчання, спроєктовано архітектуру вебплатформи, реалізовано ключові модулі системи за допомогою сучасних технологій (Next.js, Nest.js, TypeORM, PostgreSQL), проведено комплексне функціональне тестування платформи та оцінку її ефективності.

КЛЮЧОВІ СЛОВА: ОНЛАЙН-РЕПЕТИТОРСТВО, ВЕБПЛАТФОРМА, ТЕСТУВАННЯ ФУНКЦІОНАЛЬНОСТІ, ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ, ДИСТАНЦІЙНЕ НАВЧАННЯ.

ЗМІСТ

ВСТУП

Онлайн-репетиторство стало невід'ємною частиною сучасної освітньої системи, оскільки воно поєднує гнучкість дистанційного навчання з індивідуальним підходом до кожного здобувача освіти. Стрімкий розвиток цифрових технологій та збільшення попиту на персоналізовані освітні послуги зумовили потребу в ефективних інструментах координації між усіма учасниками цього процесу. Тому створення вебплатформи, що забезпечує зручну взаємодію між стейкхолдерами, є актуальним завданням сучасної ІТ-сфери. Така платформа дозволяє організувати навчальний процес, обмін інформацією та забезпечує ефективну комунікацію між студентами й викладачами в інтегрованому середовищі.

Важливою складовою процесу розробки системи координації стейкхолдерів онлайн-репетиторства є тестування функціоналу вебплатформи, яке дозволяє забезпечити її стабільність та зручність використання студентами та викладачами.

Метою магістерської роботи є проєктування, розробка та тестування вебплатформи координації стейкхолдерів онлайн-репетиторства.

Об'єктом роботи є процес організації взаємодії учасників онлайн-репетиторства.

Предметом магістерської роботи є вебплатформа координації стейкхолдерів онлайн-репетиторства.

Для досягнення поставленої мети необхідно виконати такі завдання:

1. Визначити сутність онлайн-репетиторства та особливості взаємодії його стейкхолдерів.
2. Проаналізувати існуючі платформи для організації онлайн-навчання та індивідуальних занять.
3. Спроєктувати архітектуру вебплатформи координації стейкхолдерів.
4. Реалізувати вебплатформу засобами сучасних технологій розробки.
5. Провести тестування функціоналу вебплатформи та оцінити її ефективність.

Структура магістерської роботи визначена логікою проведеного дослідження та складається з вступу, трьох розділів, висновків і списку використаних джерел. У першому розділі розглянуто поняття та сутність онлайн-репетиторства, його особливості в умовах військового стану, а також проведено порівняльний аналіз існуючих онлайн-платформ для надання репетиторських послуг. Другий розділ зосереджено на сучасних тенденціях у тестуванні функціональності вебплатформ. Розглянуто основні методології та підходи до функціонального тестування, сучасні інструменти для перевірки програмного забезпечення, а також застосування штучного інтелекту та машинного навчання у тестуванні вебсистем. Третій розділ присвячено розробці вебплатформи, описано логічну модель даних, програмну реалізацію ключових модулів системи та проведено тестування функціональності платформи. У висновках узагальнено результати дослідження та визначено перспективи подальшого розвитку розробленої вебплатформи.

Під час виконання магістерської роботи використовувалися такі програмні продукти: Visual Studio Code, Next.js, Nest.js, TypeORM, PostgreSQL, Microsoft Word.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ З НАДАННЯ ПОСЛУГ ОНЛАЙН-РЕПЕТИТОРСТВА

1.1. Поняття та сутність онлайн-репетиторства як освітньої послуги

Репетиторство є однією з форм індивідуального навчання, основною метою якого є поглиблення знань учня та допомога у засвоєнні навчального матеріалу. Сучасні технології суттєво вплинули на методи навчання, спричинивши перехід від традиційних підходів до інноваційних, що використовують цифрові інструменти та інтерактивні платформи. З розвитком інформаційних технологій сформувалася нова форма цієї послуги – онлайн-репетиторство. Дистанційне навчання, зокрема електронне навчання (e-learning), є найбільш швидко зростаючим сектором освіти і навчання та одним із важливих інструментів її модернізації [2].

Онлайн-репетиторство передбачає проведення занять дистанційно за допомогою цифрових платформ і засобів комунікації, таких як Zoom, Google Meet, Skype, Microsoft Teams тощо. Онлайн-формат дозволяє здійснювати навчання без географічних обмежень, що значно розширює можливості як для учнів, так і для викладачів.

У дослідженні [1] становлення дистанційного навчання в Україні відзначено, що така форма навчання має як значні переваги, так і певні обмеження. Серед позитивних аспектів відзначають гнучкість у виборі часу та місця занять, широкий доступ до навчальних матеріалів та можливість організувати навчальний процес більш самостійно. Крім того, онлайн-платформи дозволяють учням обирати викладача за рейтингом, спеціалізацією та відгуками, а репетиторам працювати з більшою аудиторією.

Однак, попри зручність численних платформ і функцій, студенти стикаються з низкою проблем у процесі онлайн-навчання. Перш за все, ускладнюється комунікація та взаємодія між викладачами і студентами. Відсутність живого спілкування й безпосередньої підтримки викладача, а

також складнощі з самостійним навчанням залишаються значною перешкодою для ефективності дистанційного навчання [2].

За допомогою анкетування, педагогічного спостереження та опитування у роботі [2] було визначено, що більшість учнів та вчителів змогли адаптуватися до онлайн-формату, при цьому учні швидше опанували дистанційне навчання через більш активне користування цифровими технологіями.

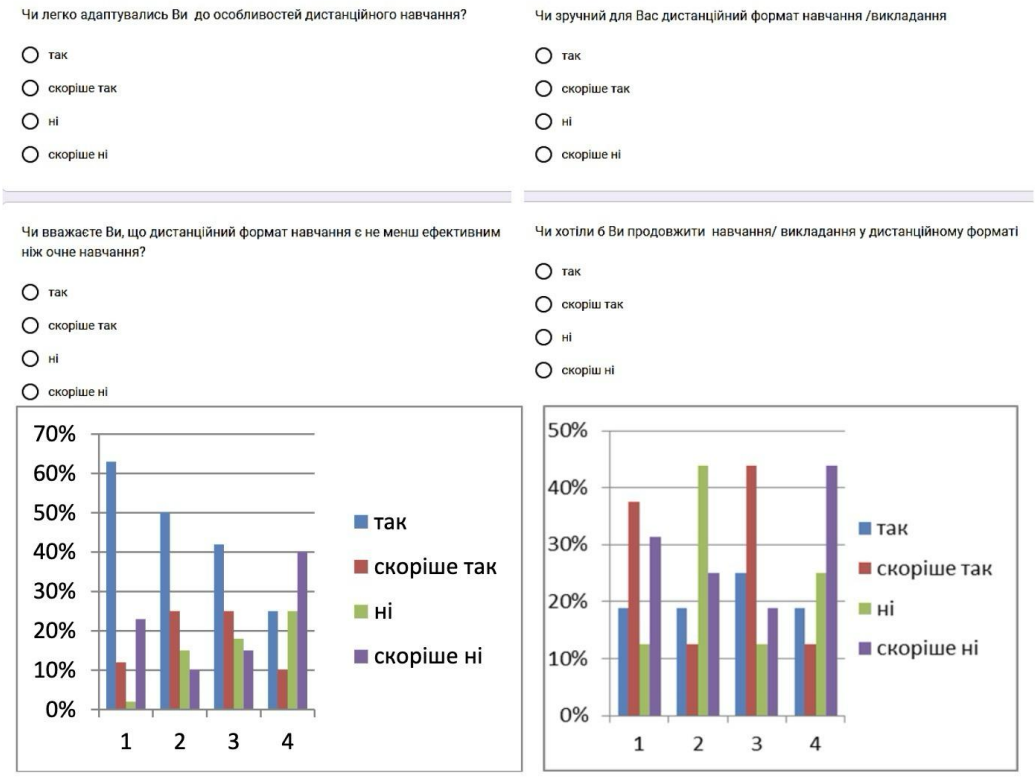

Рис. 1.1. Розподіл відповідей учнів (а) вчителів (б), де 1-4 порядок запитання в анкеті

Ці результати підкреслюють, що для ефективного функціонування онлайн-платформи важливо враховувати потреби та роль кожного учасника процесу, які безпосередньо впливають на якість та результативність надання послуг.

Отже, онлайн-репетиторство сьогодні є не просто альтернативою традиційному навчанню, а повноцінним сегментом освітнього ринку, що поєднує педагогічні принципи з технологічними можливостями. Його

розвиток стимулює створення нових інструментів координації між стейкхолдерами, серед яких особливе місце займають сучасні вебплатформи для організації та контролю навчального процесу.

1.2. Особливості онлайн-репетиторства в умовах військового стану

Військовий стан в Україні став серйозним викликом для всієї системи освіти. З початку повномасштабного вторгнення значна частина навчальних закладів була змушена перейти в онлайн-формат або частково зупинити роботу. Незважаючи на складні обставини, навчальний процес продовжується на високому рівні: студенти активно навчаються, а викладачі проводять заняття незалежно від свого місцезнаходження.

Онлайн-навчання перестало бути лише зручністю - воно стало необхідністю. Для багатьох учнів і студентів індивідуальні заняття з репетиторами стали єдиною можливістю продовжити освіту, особливо у випадках, коли школи чи університети не мали змоги повноцінно функціонувати. Як показує дослідження «Вплив локдауну шкіл на доступ до онлайн-навчання під час війни в Україні (2023)» [3], велика частина учнів з малозабезпечених сімей не мала змоги повноцінно брати участь у заняттях через відсутність технічних ресурсів, тоді як у більш заможних сім’ях доступ до онлайн-уроків був значно кращим.

Незважаючи на труднощі, попередній досвід дистанційного навчання під час пандемії COVID-19 дозволив викладачам швидко адаптуватися до нових умов і використовувати цифрові платформи для продовження навчального процесу. Онлайн-навчання стало інструментом забезпечення безперервності освіти, особливо для учнів, які були змушені переїжджати в інші регіони або перебували в зонах активних бойових дій.

Таким чином, дистанційне навчання виконувало важливу соціальну функцію — зменшувало освітні втрати в умовах війни та сприяло підтримці навчальної активності учнів і студентів [4]. Водночас дослідження відзначає, що ефективність онлайн-освіти значною мірою залежала від технічного забезпечення, цифрових навичок педагогів та здатності закладів освіти

адаптувати освітній процес до екстремальних умов [3]. В Таблиці 1 наведено ключові проблеми дистанційного навчання під час війни та можливі шляхи їх подолання, які допомагають зберегти ефективність освітнього процесу.

Таблиця 1.1

Проблеми та способи їх вирішення в онлайн-освіті під час війни в Україні

Проблема	**Можливі шляхи вирішення**
Технічні перебої: нестабільний інтернет, перебої з електроенергією, відсутність необхідного обладнання, що ускладнює онлайн-навчання	Забезпечення резервних каналів (записані лекції, офлайн-матеріали), надання студентам доступу до комп'ютерів у безпечних пунктах
Психологічний стрес як у студентів, так і у викладачів	Організація психологічної підтримки, групові онлайн та офлайн зустрічі, гнучке планування занять з врахуванням тривог, надання можливості асинхронного навчання
Втрата мотивації, зниження самоорганізації та проблеми з дисципліною	Використання змішаних форматів навчання (синхронне та асинхронне), встановлення чітких дедлайнів, підтримка спільноти однолітків
Відсутність прямого соціального контакту	Інтегрування групових онлайн-проєктів, дискусій та віртуальних зустрічей
Неможливість проведення практичних занять	Постійне оновлення змішаних моделей — поєднання онлайн з безпечними очними заняттями, використання симуляційних програм.

Попри виклики, онлайн навчання дозволяє продовжувати навчання навіть у ситуаціях, коли офлайн освіта неможлива.

Згідно з публікаціями в українських медіа (“Освіторія”, “Українська правда. Життя”, “BBC News Україна”), онлайн-репетиторство відіграє важливу соціальну роль — допомагає дітям зберігати відчуття стабільності та нормальності, а дорослим — підвищувати кваліфікацію або перекваліфіковуватися у сфері IT, цифрового маркетингу чи мовних курсів.

Отже, онлайн репетиторство в умовах воєнного стану — це не просто адаптація до нових реалій, а приклад стійкості та гнучкості освітньої системи України. Воно демонструє, що навіть у найскладніших обставинах можливе продовження навчання, розвиток і підтримка освітнього потенціалу нації.

1.3. Порівняльна характеристика існуючих онлайн платформ для репетиторства

На ринку з’явилась значна кількість платформ, які забезпечують зручну взаємодію між учнями та викладачами. До найпопулярніших з них належать BUKI, Preply, Superprof, GOAL та Mathema:

1. Платформа BUKI

BUKI — це українська онлайн-платформа, що виступає маркетплейсом для пошуку приватних репетиторів. Вона дає змогу користувачам обирати викладача за предметом, рівнем підготовки, місцем проживання чи форматом навчання (онлайн або офлайн).

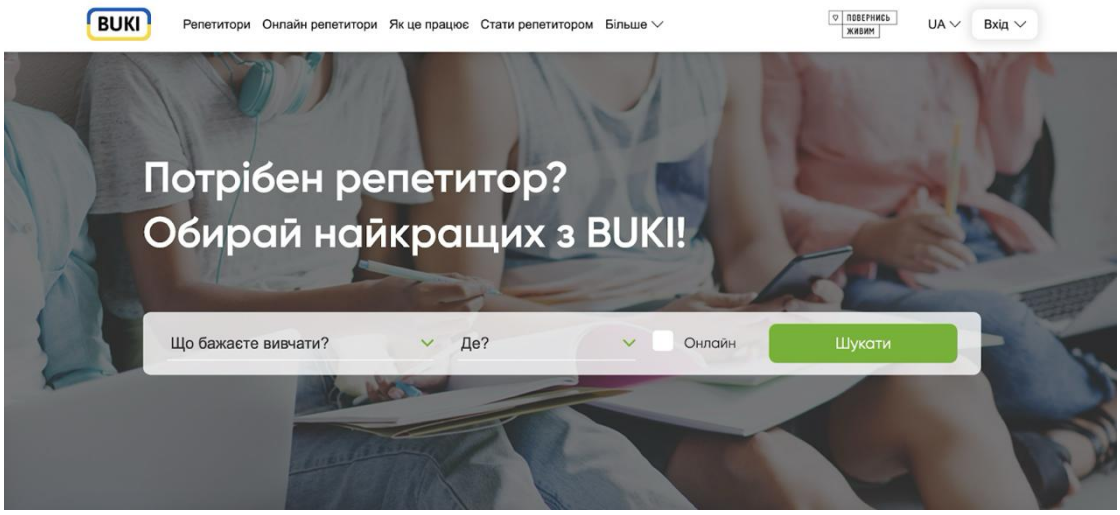

Рис. 1.2. Веб-платформа Buki

Сервіс працює як посередник між репетитором і студентом: платформа надає зручну систему пошуку, фільтри, рейтинги та відгуки. Репетитори

самостійно встановлюють вартість занять і можуть проходити верифікацію документів для підвищення довіри користувачів.

2. Платформа Preply

Preply — міжнародна освітня платформа, яка об’єднує викладачів і студентів у понад 180 країнах світу. Основний фокус — вивчення іноземних мов, хоча також представлені інші предмети (математика, інформатика, бізнес-навички тощо).

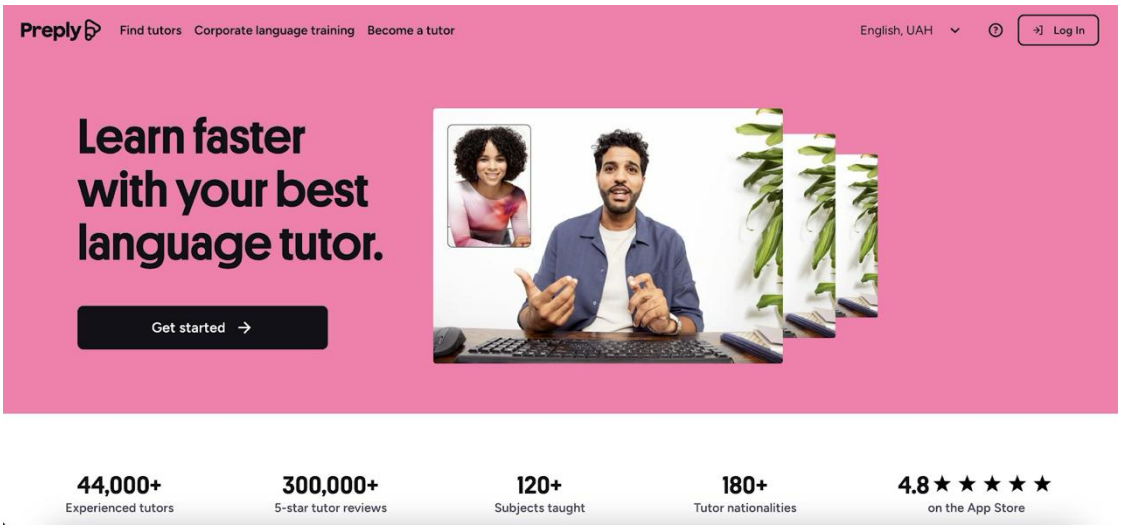

Рис. 1.3. Веб-платформа Preply

Платформа використовує алгоритми підбору викладачів, пропонує пробні уроки та надає користувачам зручний інтерфейс із відеозв’язком. Кожен репетитор проходить модерацію, заповнює профіль із відеопрезентацією, сертифікатами та відгуками.

3. Платформа Superprof

Superprof — міжнародна платформа-каталог, яка дозволяє знайти викладача практично з будь-якого предмета — від академічних дисциплін до спорту, музики чи творчих занять. Сервіс працює в більш ніж 40 країнах світу.

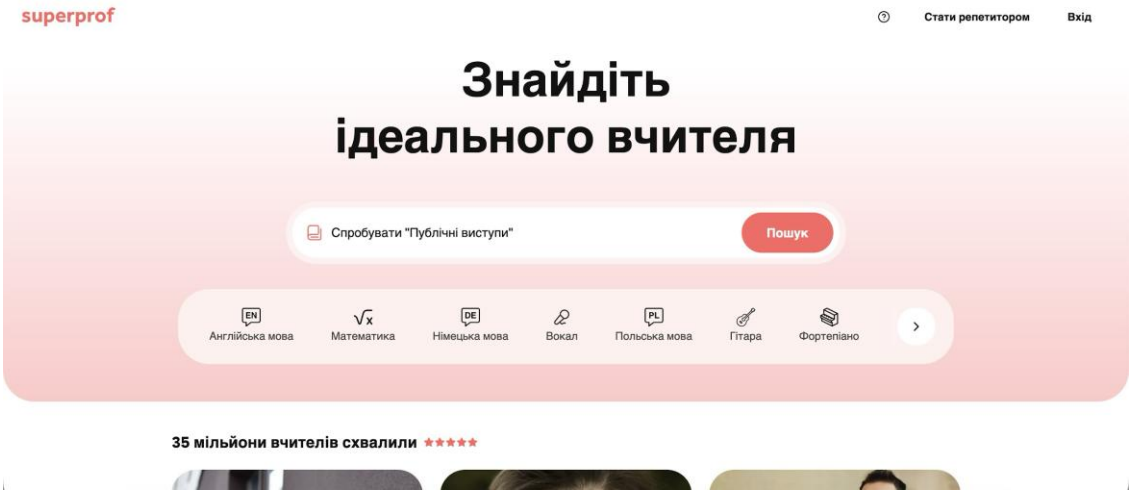

Рис. 1.4. Веб-платформа Superprof

Учні можуть переглядати профілі викладачів, читати відгуки та зв'язуватися безпосередньо через сайт. Оплата здійснюється поза платформою, а сервіс може стягувати абонентську плату з учнів за доступ до контактів викладачів.

4. Платформа GOAL

GOAL — українська платформа для підготовки учнів до ЗНО, НМТ та шкільних предметів. Вона поєднує функції маркетплейсу та освітньої системи, пропонуючи курси, групові та індивідуальні заняття.

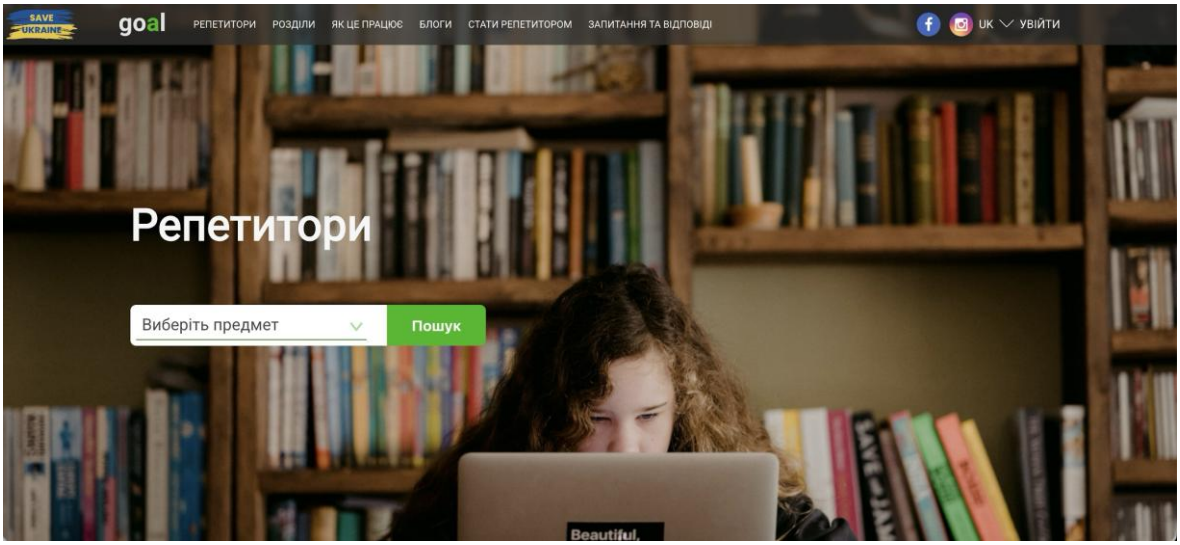

Рис. 1.5. Веб-платформа GOAL

Сервіс орієнтований на інтерактивне навчання: надає особисті кабінети для викладачів і студентів, можливість відстеження прогресу, автоматичну перевірку тестів і аналітику результатів. Платформа також забезпечує

гнучкість у виборі формату занять — групові чи індивідуальні уроки, а система рекомендацій дозволяє адаптувати навчальний процес під потреби кожного учня, підвищуючи ефективність підготовки.

5. Платформа Mathema

Mathema — онлайн-сервіс, орієнтований на індивідуальні заняття з математики, фізики та програмування. На відміну від маркетплейсів, ця платформа має власний штат викладачів, які проходять внутрішній відбір.

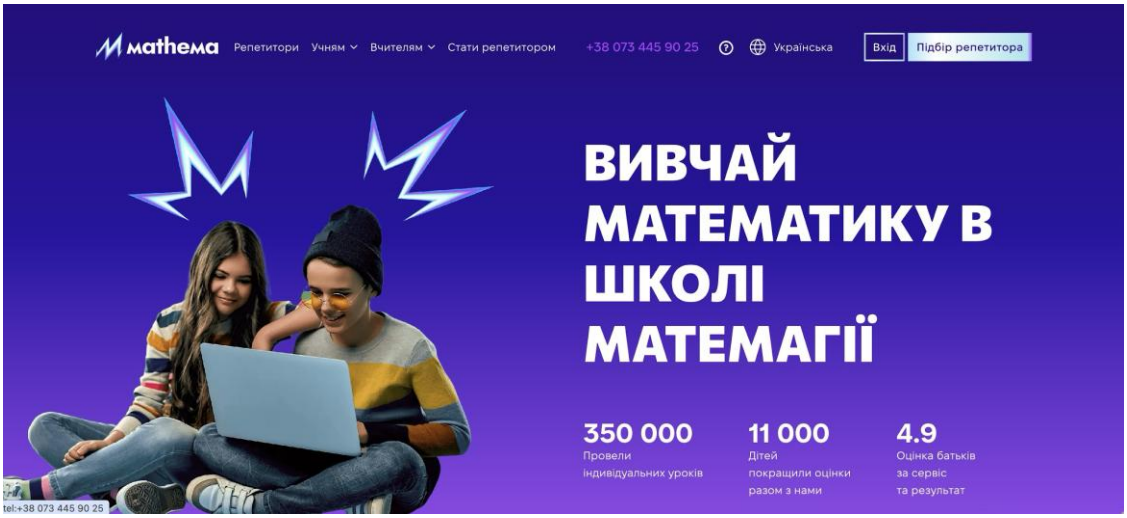

Рис. 1.6. Веб-платформа Mathema

Система Mathema дозволяє планувати індивідуальні заняття, вести відеоуроки безпосередньо на сайті, а також отримувати домашні завдання та персоналізовані рекомендації для покращення навчальних результатів. Платформа має власний штат викладачів, які проходять внутрішній відбір, що забезпечує високий рівень якості навчання

Кожна з описаних платформ має свої особливості та призначена для різних потреб користувачів. BUKI та GOAL орієнтовані на український ринок і поєднують функції маркетплейсу та освітньої системи, тоді як міжнародні платформи Preply та Superprof дають доступ до викладачів з усього світу. Платформи також відрізняються підходом до роботи з викладачами: деякі мають власний штат педагогів (Mathema), інші дозволяють студентам обирати та оцінювати репетиторів самостійно. В Таблиці 1.2 описані особливості платформ для їх детального порівняння.

Таблиця 1.2

Порівняльна таблиця платформ онлайн-репетиторства

Платформа	**Основний фокус**	**Відбір викладачів**	**Інструменти для навчання**	**Основні переваги**
BUKI	Шкільні предмети, підготовка до ЗНО	Часткова верифікація документів	Без вбудованих інтерактивних інструментів	Велика локальна база викладачів, можливість обрати за місцем і форматом навчання
Preply	Іноземні мови, загальні предмети	Модерація, відео-презентації, відгуки	Вбудовані інструменти для онлайн-занять, мобільний застосунок	Міжнародна аудиторія, пробні уроки, зручний інтерфейс
Superprof	Академічні та творчі дисципліни	Мінімальний відбір	Відсутні вбудовані інструменти	Великий вибір предметів, прямий контакт із викладачами
GOAL	Шкільна підготовка, ЗНО	Внутрішній відбір викладачів	Інтерактивні навчальні модулі, аналітика прогресу	Вбудований функціонал для контролю прогресу та інтерактивного навчання
Mathema	Математика, фізика, ІТ	Внутрішній відбір викладачів	Повна інтеграція навчального процесу, відеоуроки, домашні завдання	Висока якість навчання, контроль прогресу, персоналізовані рекомендації

У результаті дослідження були визначені популярні веб-платформи для пошуку репетиторів, а також складено короткий опис і порівняльну таблицю цих платформ. Вона містить ключові особливості, типи курсів, цільову аудиторію та особливості кожної платформи.

На основі отриманих даних визначено ключові особливості, які можна використовувати при розробці власної веб-платформи для пошуку репетитора, а саме: можливість зручного пошуку викладача за предметом і рівнем підготовки, інтеграція інструментів для онлайн-занять, система рейтингів та відгуків для оцінки якості викладання, а також персоналізовані рекомендації для учнів на основі їхніх потреб і результатів навчання.

РОЗДІЛ 2. СУЧАСНІ ТРЕНДИ У ТЕСТУВАННІ ФУНКЦІОНАЛЬНОСТІ ВЕБПЛАТФОРМ

2.1. Методології та підходи до функціонального тестування

Процес тестування програмного забезпечення відіграє ключову роль у забезпеченні його якості, надійності та відповідності вимогам користувачів. Основна мета тестування полягає у оцінюванні функціональних, нефункціональних, структурних та архітектурних характеристик програмного продукту, а також у перевірці впливу внесених змін на роботу системи.

Функціональне тестування спрямоване на перевірку того, що саме має виконувати система, тобто чи працюють її функції відповідно до вимог замовника. На відміну від функціонального, нефункціональне тестування оцінює якість роботи системи — її надійність, продуктивність, безпеку, сумісність і зручність використання.

Основними видами функціонального тестування є димове тестування (Smoke Testing), тестування критичного шляху (Critical Path Testing) та розширене тестування (Extended Testing).

Димове тестування (Smoke Testing) – це перевірка основної працездатності системи, чи працюють ключові функції перед початком детального тестування. Метою є впевнитися, що продукт взагалі можна тестувати далі: він запускається, виконує базові дії та не має критичних помилок, які блокують роботу. Термін «димове тестування» походить із технічної інженерії: перевірку нового обладнання вважали успішною, якщо з нього не йшов дим після запуску.

Тестування критичного шляху (Critical Path Testing) - тип тестування зосереджений на найважливіших функціях, якими користується більшість користувачів. Тестувальник перевіряє так званий “критичний шлях” — основну послідовність дій, яку виконує користувач для досягнення своєї мети. Якщо цей шлях не працює, вся система вважається нефункціональною, незалежно від справності другорядних функцій.

Розширене тестування (Extended Testing) - передбачає перевірку всіх функцій, описаних у вимогах**,** а також нестандартних сценаріїв використання системи.

Також, не менш важливим є те, на якому рівні тестування виконується перевірка. Рівні тестування визначають, що саме перевіряється — окремий модуль, взаємодія компонентів чи робота всієї системи [6]. Основними рівнями тестування є модульне, інтеграційне, системне та приймальне тестування, детальний опис кожного рівня представлено в Таблиці 2.1.

Таблиця 2.1

Опис рівнів тестування

Рівень тестування	**Мета тестування**	**Результат**
Модульне (Unit Testing)	Перевірка правильності роботи окремих функцій або модулів	Підтвердження коректності реалізації окремих компонентів
Інтеграційне (Integration Testing)	Перевірка взаємодії між об’єднаними модулями	Виявлення дефектів у зв’язках між компонентами
Системне (System Testing)	Перевірка повної системи на відповідність функціональним і нефункціональним вимогам	Підтвердження цілісності та стабільності системи
Приймальне (Acceptance Testing)	Оцінка відповідності програмного продукту бізнес-вимогам і готовності до впровадження	Рішення про прийняття системи в експлуатацію

Залежно від рівня доступу до коду та знань про внутрішню структуру системи виділяють три основні методи: тестування чорного ящика (Black Box Testing), тестування білого ящика (White Box Testing) та тестування сірого ящика (Grey Box Testing).

Згідно з ISTQB (International Software Testing Qualifications Board) [5], ці методи відносяться до технік тестування і призначені для визначення умов тестування, тестових випадків та даних для перевірки програмного забезпечення:

1. Тестування чорного ящика:

Цей метод зосереджений на поведінці системи та перевірці відповідності функціональним вимогам. Тестувальник не знає внутрішньої структури коду і працює на рівні користувацького інтерфейсу перевіряючи, чи програмний продукт виконує очікувані дії.

Наприклад, при тестуванні веб-додатку перевіряються такі дії користувача: вхід у систему, додавання товару у кошик, оформлення замовлення. Для цього виду тестування знання програмування не потрібні.

2. Тестування білого ящика:

Тестування білого ящика передбачає повний доступ до внутрішньої структури коду. Мета – перевірити логіку реалізації та правильність роботи окремих компонентів. Цей метод зазвичай застосовують розробники, які можуть аналізувати код, контролювати всі гілки виконання та перевіряти умови. Наприклад, модульне тестування функцій та перевірка алгоритмів.

3. Тестування сірого ящика:

Метод поєднує підхід чорного та білого ящика: тестувальник частково знає внутрішню структуру програми та одночасно працює на рівні користувача. Метод дозволяє перевіряти систему за користувацькими сценаріями з урахуванням особливостей реалізації, що підвищує ефективність виявлення дефектів, які складно знайти при повністю чорному або білому підході.

Водночас, для ефективного планування тестування, застосовуються техніки тестування – стратегії та правила для розробки тестової документації та вибору тестових умов, які використовуються під час тест-дизайну

Тест-дизайн (Test Design) – це процес створення тестів, це процес створення тестів на основі специфікацій, вимог або іншої документації для перевірки правильності, повноти та якості програмного забезпечення [5].

Згідно з ISTQB, тест-дизайн включає:

- Розробку та пріоритизацію чек-листів, тест-кейсів і наборів тест-кейсів
- Визначення необхідних тестових даних

- Проєктування тестового середовища та підбір інструментів
- Забезпечення двосторонньої трасованості між вимогами, тестовими умовами та тестами

Правильне використання технік тестування дозволяє зменшити кількість тестів, водночас забезпечивши широке покриття вимог. Найпоширеніші техніки тестування включають:

1. Розбиття на класи еквівалентності (Equivalence Partitioning):

Підхід, за якого тестові дані поділяються на групи (класи), що містять елементи з однаковими характеристиками або очікуваною поведінкою системи. Тестувальник обирає лише кілька значень із кожної групи, оскільки передбачається, що решта даних у цьому класі дадуть той самий результат. Такий метод особливо ефективний, коли маємо велику кількість вхідних даних, метод дозволяє суттєво скоротити кількість тестів без ризику зниження якості перевірки.

2. Аналіз граничних значень (Boundary Value Analysis):

Підхід, який застосовується у випадках, коли дані впорядковані або числові. Межові значення – це «краї» кожного еквівалентного класу. Основна увага приділяється саме цим межам, оскільки більшість помилок виникає саме на границях дозволених значень, а не в їх середині.

3. Парне тестування (Pairwise Testing):

Техніка, спрямована на оптимізацію кількості тестів шляхом перевірки всіх можливих комбінацій пар вхідних параметрів. Це дозволяє охопити значну частину потенційних помилок, зменшуючи кількість тест-кейсів, необхідних для повного покриття всіх варіацій.

Отже, грамотне поєднання рівнів, методів і технік тестування відповідно до цілей проєкту дає змогу підвищити надійність, продуктивність і стабільність програмного забезпечення, а також своєчасно виявити критичні помилки ще до його впровадження.

2.2. Сучасні інструменти функціонального тестування

Серед найбільш популярних і ефективних інструментів тестування сьогодні виділяють TestRail, Postman, Selenium та Cypress. Кожен із них орієнтований на певний тип тестування, а їхнє комплексне використання дозволяє забезпечити повне охоплення функціональних перевірок на різних рівнях системи.

TestRail — це платформа для управління тестуванням на основі штучного інтелекту, яка допомагає оптимізувати процеси тестування програмного забезпечення: структурувати тест-кейси, створювати тест-плани, відстежувати прогрес виконання тестів і формувати звітність у зручному та централізованому вигляді. Завдяки цьому TestRail виступає важливою частиною інфраструктури функціонального тестування, забезпечуючи прозорість та контроль як на етапі ручних перевірок, так і в поєднанні з автоматизацією.

Однією з ключових переваг TestRail є можливість гнучко організовувати тестову документацію: тест-кейси можна групувати за функціональністю, модулями або релізами, створювати окремі тестові цикли та запускати їх для різних збірок продукту [7]. Така структура полегшує масштабування тестування навіть у великих проєктах.

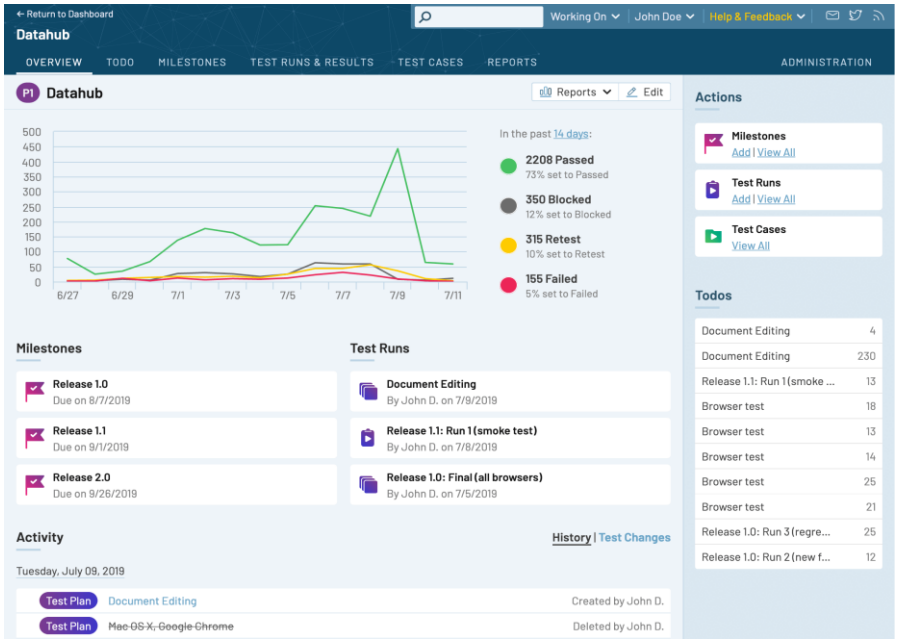

Рис. 2.1. Платформа для управління тестуванням TestRail

Postman — це потужний інструмент для тестування API, який широко використовується для перевірки взаємодії між клієнтською та серверною частинами програмного забезпечення.

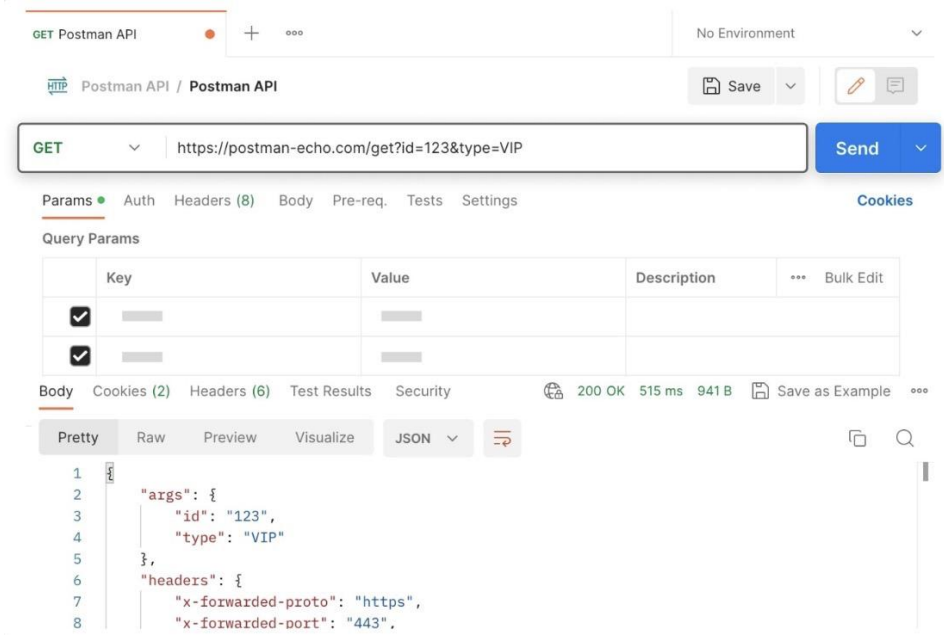

Рис. 2.2. Платформа для тестування та розробки API Postman

За його допомогою тестувальники можуть створювати, відправляти та автоматизувати HTTP-запити, перевіряти коректність відповідей, структуру даних, статус-коди та швидкодію серверів. Postman дозволяє організовувати тести у вигляді колекцій, що спрощує повторне використання сценаріїв та інтеграцію з CI/CD-процесами. Завдяки цьому забезпечується ефективна автоматизація тестування RESTful API, що є важливою складовою сучасних вебзастосунків [8].

Postman також має вбудоване середовище для написання тестових скриптів на JavaScript, що дає змогу проводити гнучку валідацію даних та виконувати складні перевірки.

Завдяки можливості параметризації, керування змінними середовищ, а також візуалізації відповідей, Postman є одним із найзручніших рішень для командної роботи та документування API. Інтеграція з Git, Jenkins, GitHub Actions та іншими платформами робить його важливим інструментом у комплексній автоматизації тестування в межах DevOps-процесів [9].

Selenium — це фреймворк для автоматизованого функціонального тестування вебзастосунків. Він імітує дії реального користувача в браузері, перевіряючи працездатність інтерфейсу, коректність роботи елементів сторінки, форм, кнопок і навігації. Selenium підтримує різні мови програмування (Java, Python, C#, JavaScript тощо) і може працювати з більшістю сучасних браузерів [10].

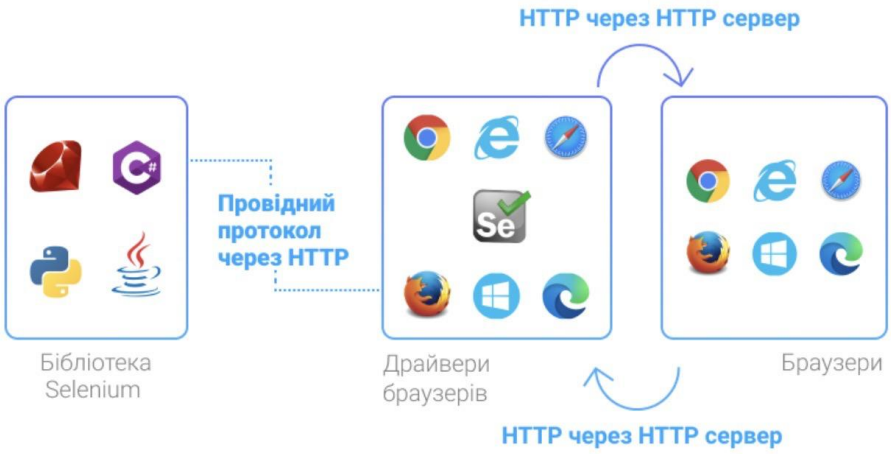

Рис. 2.3. Архітектура Selenium

Цей інструмент активно використовується в проєктах, побудованих за принципами DevOps і Agile, де важливо забезпечити безперервну інтеграцію та доставку програмного забезпечення. Selenium підтримує кілька мов програмування — Java, Python, C#, JavaScript, Ruby. Завдяки таким компонентам, як Selenium WebDriver, Selenium Grid і Selenium IDE, цей інструментарій підходить як для складних систем із великою кількістю модулів, так і для швидкого прототипування тестових сценаріїв.

Cypress — інструмент для автоматизованого тестування веб-інтерфейсів, який швидко здобув популярність завдяки своїй простоті, швидкодії та інтеграції з екосистемою JavaScript. На відміну від Selenium, Cypress працює безпосередньо всередині браузера, що дозволяє виконувати тести швидше та отримувати точнішу інформацію про помилки.

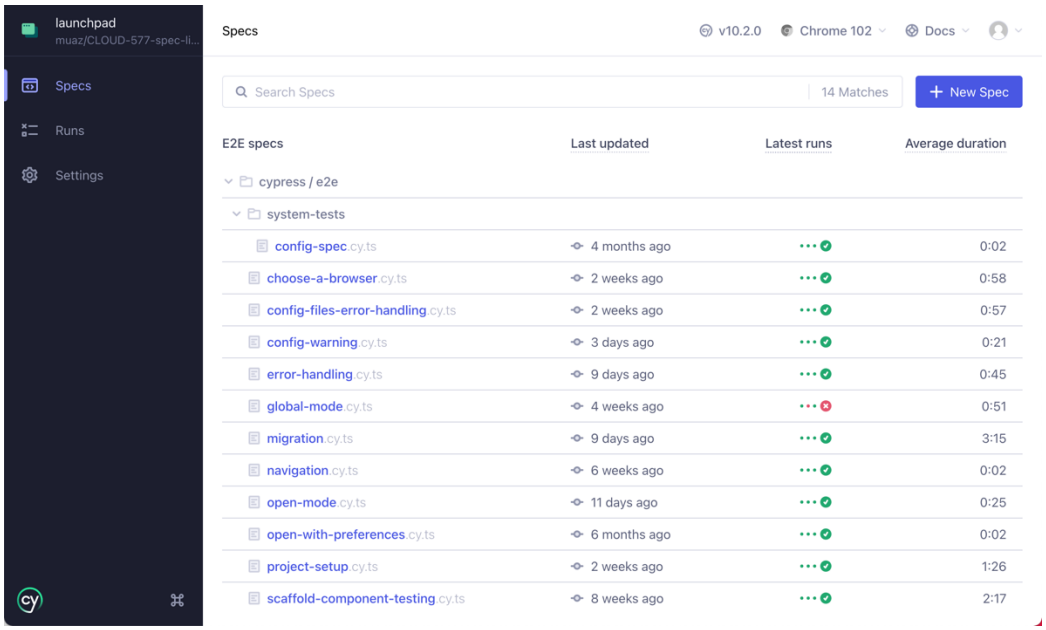

Рис. 2.4. Фреймворк для тестування веб-додатків Cypress

Cypress чудово підходить для end-to-end тестування — тобто перевірки наскрізних сценаріїв взаємодії користувача із системою. Інструмент забезпечує візуалізацію процесу виконання тестів у реальному часі, а також має зручний інтерфейс для налагодження. Завдяки цьому тестувальники можуть швидко аналізувати результати та виправляти дефекти.

Таким чином, використання комплексного набору інструментів для тестування дозволяє забезпечити всебічну перевірку програмного забезпечення на різних рівнях. Кожен з цих інструментів має свої переваги та специфіку застосування, а їхнє поєднання дозволяє підвищити ефективність тестування та забезпечити повне тестове покриття вимог.

2.3. Тренди у застосуванні штучного інтелекту та машинного навчання у тестуванні

Сучасні напрями розвитку забезпечення якості та тестування програмного забезпечення тісно пов’язані з активним впровадженням інструментів на основі штучного інтелекту та машинного навчання.

Застосування машинного навчання у тестуванні дозволяє не лише автоматизувати рутинні завдання, а й підвищити ефективність планування та виконання тестів [11]. Зокрема, моделі можуть:

- аналізувати попередні результати тестів;

- прогнозувати найбільш критичні зони програмного забезпечення;
- пропонувати найбільш ефективні сценарії перевірки;
- генерувати шаблони тест-кейсів, баг-репортів, чеклістів;
- проводити попередній аналіз специфікацій;
- структурувати вимоги;
- створювати базові матриці покриття.

Серед сучасних AI-інструментів, які використовуються для автоматизації тестування, можна виділити:

1. Testim — платформа для автоматизованого створення тестів на основі машинного навчання:

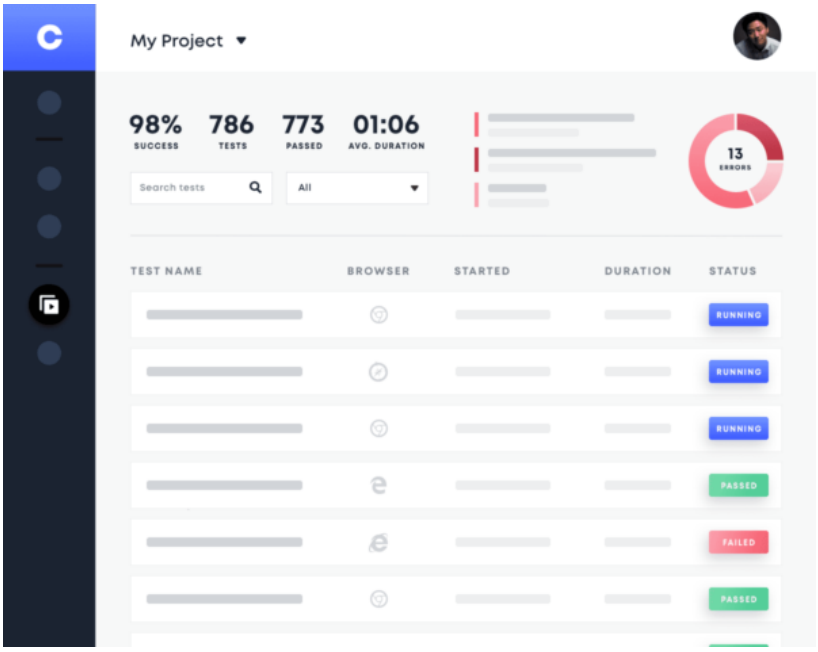

Рис. 2.5. Платформа Testim

Платформа самостійно генерує сценарії для веб-застосунків і адаптує їх під зміни інтерфейсу. Testim дозволяє автоматично виявляти нестабільні тести та оптимізувати тестові набори для зменшення часу виконання, що особливо корисно у великих проектах з постійними змінами коду.

2. Functionize – інструмент, що використовує штучний інтелект для генерації та оптимізації тест-кейсів:

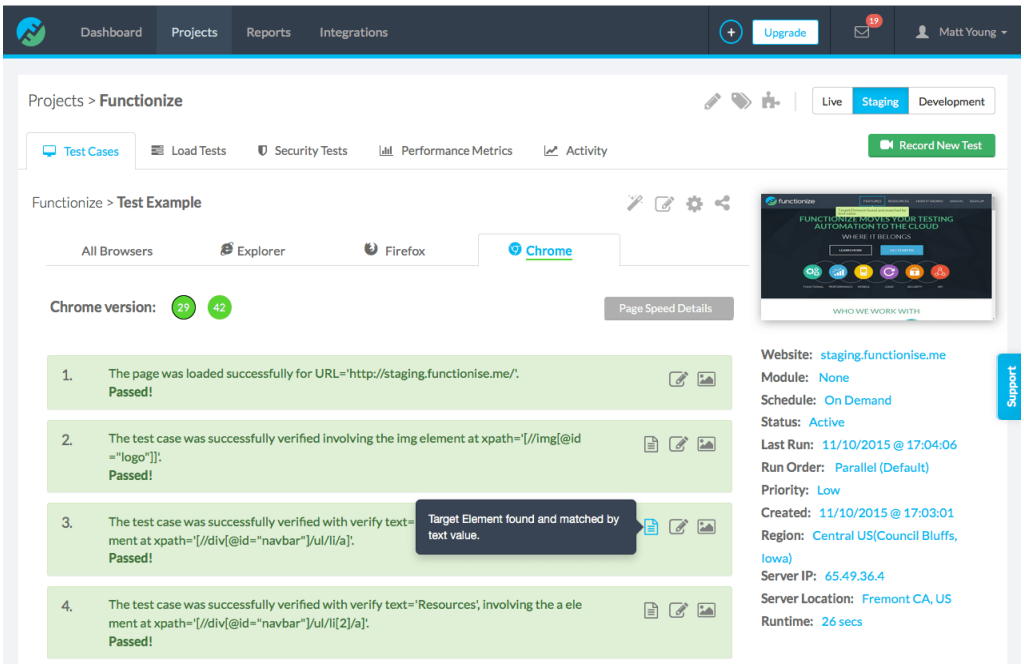

Рис. 2.6. Платформа Functionize

Functionize аналізує код і визначає ділянки, де ймовірність помилок найвища, автоматично пропонує тестові сценарії та легко інтегрується з процесами CI/CD. Це дозволяє швидше тестувати програму та ефективніше використовувати ресурси для перевірки основних функцій.

1. Applitools – сервіс для візуального тестування, що застосовує машинне навчання для порівняння інтерфейсів користувача:

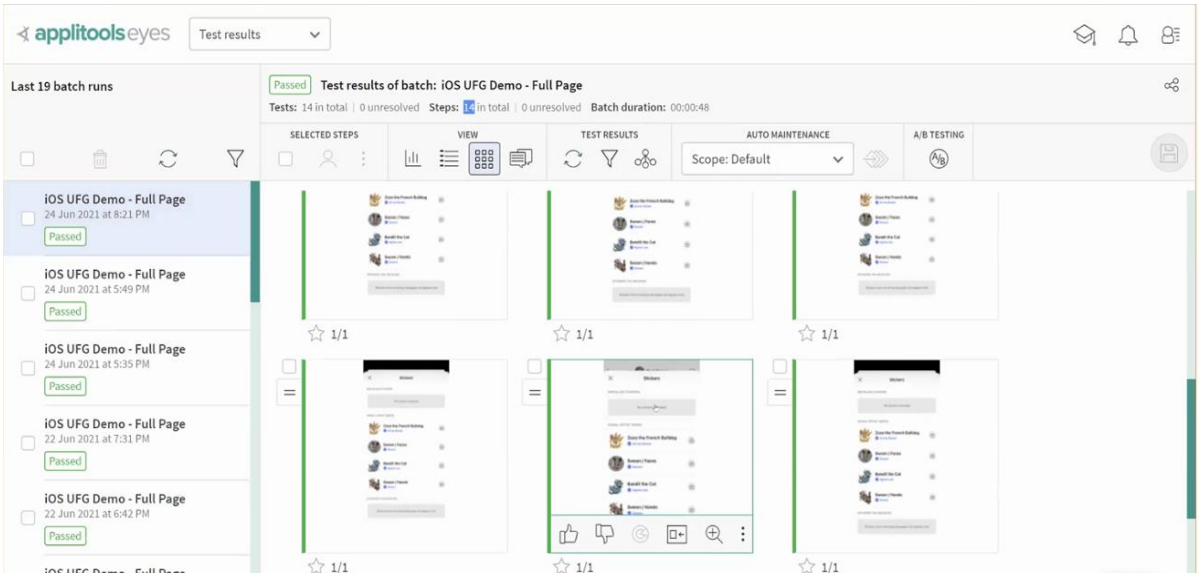

Рис. 2.7. Платформа Applitools

Applitools автоматично виявляє UI-помилки, зміни дизайну та невідповідності макетам, дозволяє проводити кросбраузерне тестування, що значно скорочує час перевірки візуальної частини веб-застосунків.

2. Mabl – сервіс, який комбінує автоматизацію тестування з аналізом поведінки користувачів та прогнозуванням областей підвищеного ризику у коді.

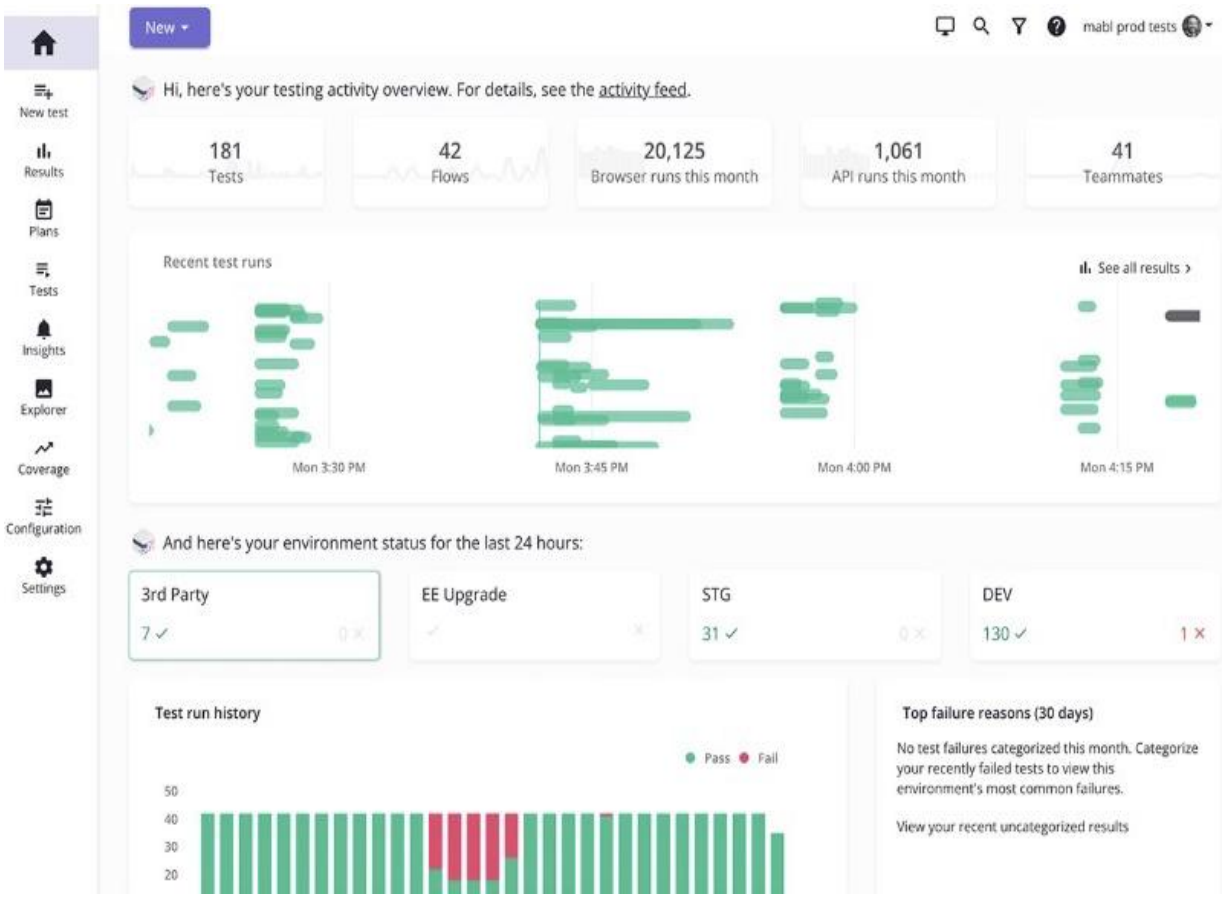

Рис. 2.8. Платформа Mabl

Mabl генерує тестові сценарії на основі реальних даних, виявляє потенційні дефекти та інтегрується з CI/CD-процесами. Інструмент також дозволяє автоматично оновлювати тестові набори при змінах у додатку, що підвищує покриття тестами та зменшує необхідність ручного втручання.

Проте, інструменти штучного інтелекту не можуть повністю замінити ручне тестування. Навіть з використанням автоматизації та ШІ залишаються завдання, що потребують критичного мислення. Автоматизовані тести часто потребують регулярного обслуговування і можуть давати хибні результати без ручної перевірки. Таким чином, ручне тестування залишається незамінним для забезпечення повного покриття функціональних і нефункціональних вимог, а також для оцінки якості продукту з точки зору кінцевого користувача.

РОЗДІЛ 3. ВЕБПЛАТФОРМА КООРДИНАЦІЇ СТЕЙКХОЛДЕРІВ ОНЛАЙН-РЕПЕТИТОРСТВА

3.1. Логічна модель даних

Логічне проектування полягає в створенні логічної моделі на основі вибраної моделі даних [12].

Логічна модель бази даних є абстракцією структури даних, яка описує, як дані будуть організовані та зв'язані між собою. Логічна модель бази даних зазвичай включає сутності (наприклад, таблиці), атрибути цих сутностей, а також відносини між ними. Вона відображає концептуальні моделі бізнесу та дозволяє створити чітке уявлення про дані, які необхідні для підтримки функціональних вимог системи.

Основною метою логічної моделі є забезпечення незалежності від конкретних технологічних рішень, таких як система управління базами даних (СУБД), що дозволяє проектувати базу даних, яка буде адаптуватися до різних технічних вимог на етапі реалізації.

Логічна модель слугує основою для подальшого створення фізичної моделі, яка безпосередньо визначає структуру таблиць, індекси, обмеження та інші елементи, які необхідні для ефективної роботи бази даних. Побудова логічної моделі бази даних дозволяє графічно зобразити зв'язки бази даних та спрощує сприйняття даних.

Між реляційними таблицями можливо встановити наступні типи зв'язків:

- 1:1 (один-до-одного) - кожен запис в одній таблиці відповідає лише одному запису в іншій таблиці;
- 1:N (один-до-багатьох) один запис в одній таблиці може бути пов'язаний з кількома записами в іншій таблиці;
- N:N (багато-до-багатьох) - кілька записів в одній таблиці можуть бути пов'язані з кількома записами в іншій таблиці.

У збалансованій структурі реляційної бази даних кожен запис в будь-якій таблиці повинна унікально ідентифікуватися, тобто значення деяких полів в таблиці не повинні повторюватися у всій безлічі записів, цей унікальний ідентифікатор називається первинним ключем [13].

Веб-сайт з пошуку репетиторів дозволяє реєструватися учням, переглядати інформацію про репетиторів з потрібного предмета та залишати заявки на навчання. Права адміністратора дозволяють редагувати список репетиторів, видаляти та додавати нових вчителів, а також переглядати заявки, залишені учнями. Створені таблиці та їх поля наведено в Таблиці 3.1.

Таблиця 3.1

Таблиці бази даних

Назва таблиці	**Поля таблиці**	**Опис таблиці**
Користувачі (users)	Ідентифікатор, Електронна пошта, Пароль, Ім'я, Прізвище, Роль, Підтверджений викладач, Дата створення, Дата оновлення	Основна інформація про користувачів платформи
Відгуки (review)	Ідентифікатор, Оцінка, Коментар, Дата створення, Ідентифікатор викладача, Ідентифікатор студента	Відгуки студентів про викладачів
Освіта (education)	Ідентифікатор, Ступінь, Університет, Роки навчання, Ідентифікатор викладача, Дата створення, Дата оновлення	Інформація про освітній рівень викладачів
Викладачі (teacher)	Ідентифікатор, Біографія, Заголовок/опис, Роки досвіду, Почасова ставка, Місцезнаходження, Статус, Дата створення, Дата оновлення, Ідентифікатор користувача	Дані про викладачів, їх профіль та досвід
Студенти (student)	Ідентифікатор, Фото профілю, Дата створення, Дата оновлення, Мови, Ідентифікатор користувача	Інформація про студентів та їх акаунти

Предмети викладача (teacher_subjects)	Ідентифікатор, Ідентифікатор предмета, Ідентифікатор викладача	Зв'язок викладачів з предметами, які вони викладають
Предмети (subject)	Ідентифікатор, Назва, Опис, Категорія, Дата створення, Дата оновлення	Список предметів з описом та категоріями
Доступність викладача (availability)	Ідентифікатор, День тижня, Час початку, Час завершення, Ідентифікатор викладача	Графік доступності викладачів для бронювання
Досвід роботи (experience)	Ідентифікатор, Посада, Місцезнаходження, Роки, Опис, Дата створення, Дата оновлення, Ідентифікатор викладача	Додатковий професійний досвід викладачів
Бронювання (booking)	Ідентифікатор, Дата, Нотатка, Статус, Дата створення, Дата оновлення, Ідентифікатор студента, Ідентифікатор викладача	Інформація про заплановані уроки та їх статус

Між таблицями бази даних встановлено зв’язки типу один-до-багатьох, а також опція On Delete, яка виконує видалення зв’язаних даних у разі видалення атрибуту з головної таблиці. Каскадне оновлення даних полів пов’язаних таблиць між якими встановлено зв’язок багато-до-багатьох встановлено за допомогою опції CASCADE.

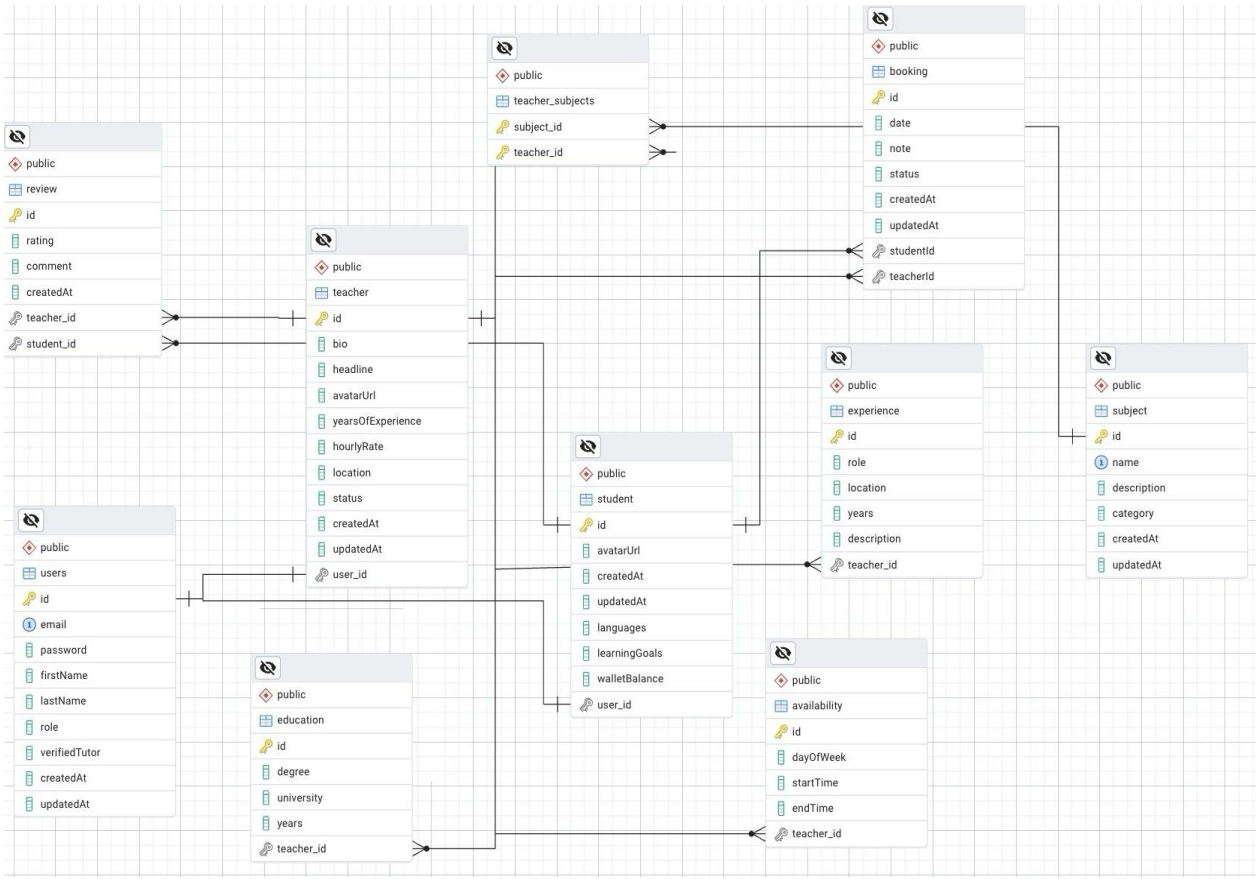

Рис. 3.1. Логічна модель бази даних

Для створення бази даних інформаційної системи було використано PostgreSQL.

PostgreSQL — популярний альтернативний засіб віддаленого управління базами даних. PostgreSQL спрощує створення, редагування, структурування та управління базою даних, використовуючи зручний інтерфейс. Використовуючи PostgreSQL, було створено наступні таблиці:

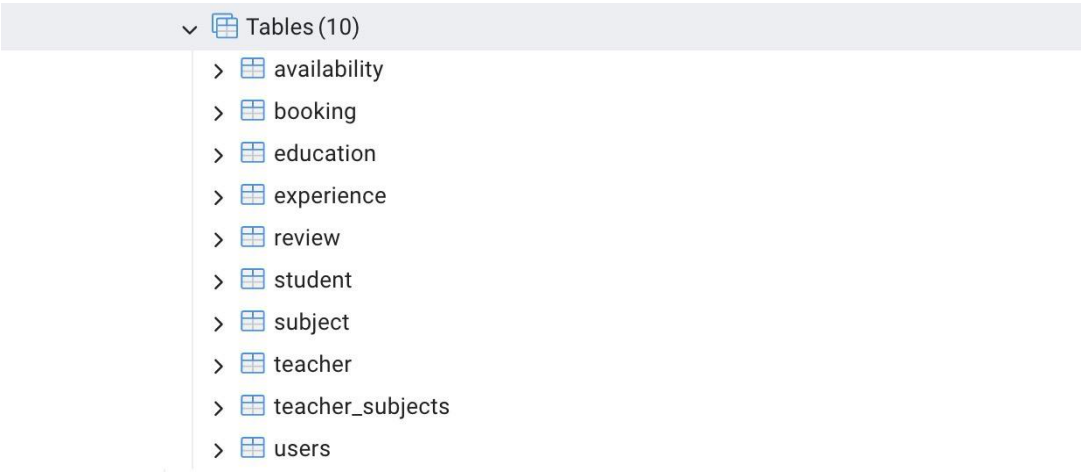

Рис.3.2. Таблиці бази даних в PostgreSQL

Розглянемо кожну таблицю окремо:

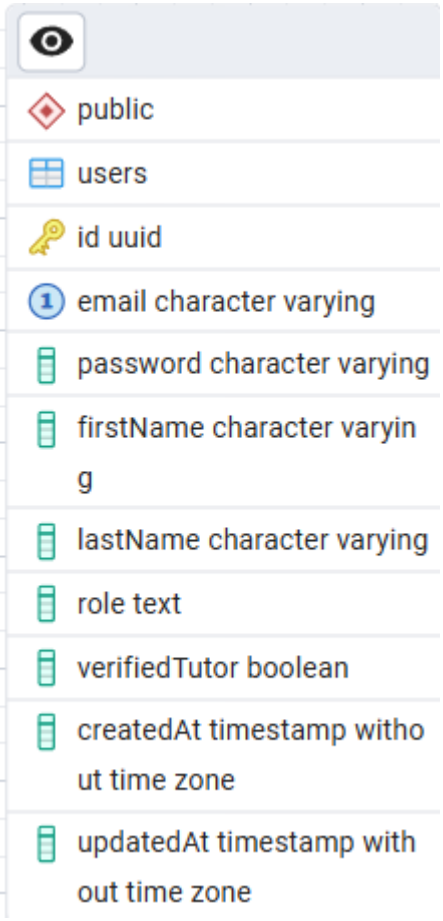

Рис. 3.3. Таблиця «Користувачі»

Таблиця «Користувачі» використовується для зберігання даних про зареєстрованих користувачів, їх імені, прізвища, електронної пошти, паролю, та прав доступу на сайті.

SQL запит для створення таблиці «Користувачі»:

```
CREATE TABLE public.users (
    id uuid DEFAULT uuid_generate_v4() NOT NULL,
    email varchar NOT NULL,
    "password" varchar NOT NULL,
    "firstName" varchar NOT NULL,
    "lastName" varchar NOT NULL,
    "role" text DEFAULT 'STUDENT'::text NOT NULL,
    "verifiedTutor" bool DEFAULT false NOT NULL,
    "createdAt" timestamp DEFAULT now() NOT NULL,
```

"updatedAt" timestamp DEFAULT now() NOT NULL,
CONSTRAINT "PK_a3ffb1c0c8416b9fc6f907b7433" PRIMARY KEY (id),
CONSTRAINT "UQ_97672ac88f789774dd47f7c8be3" UNIQUE (email)
);

Таблиця «Студенти» призначена для збереження інформації про студентів та їх облікові записи, включно з особистими даними, електронною поштою, роллю в системі та статусом підтвердження.»

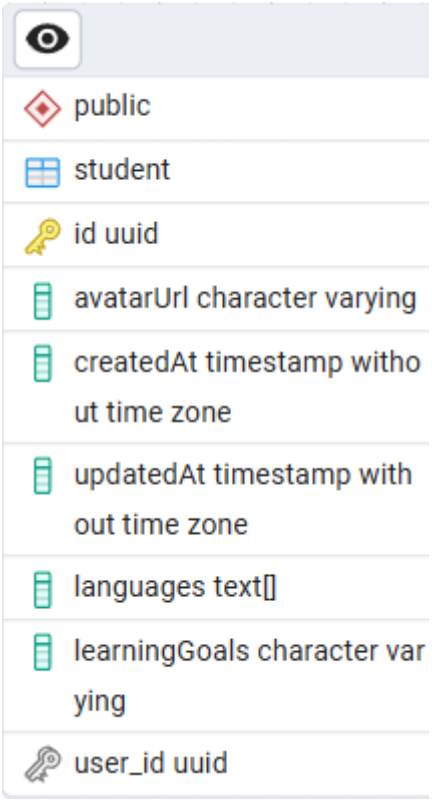

Рис. 3.4. Таблиця “Студенти”

Запит на створення таблиці «Студенти»:

CREATE TABLE public.student (
id uuid DEFAULT uuid_generate_v4() NOT NULL,
"avatarUrl" varchar NULL,
"createdAt" timestamp DEFAULT now() NOT NULL,
"updatedAt" timestamp DEFAULT now() NOT NULL,
languages _text NOT NULL,
"learningGoals" varchar NOT NULL,
user_id uuid NULL,
CONSTRAINT "PK_3d8016e1cb58429474a3c041904" PRIMARY KEY (id),
CONSTRAINT "REL_0cc43638ebcf41dfab27e62dc0" UNIQUE (user_id),

CONSTRAINT "FK_0cc43638ebcf41dfab27e62dc09" FOREIGN KEY (user_id) REFERENCES public.users(id) ON DELETE CASCADE
);

Наступна таблиця «Вчителі» відображає дані про репетиторів, їх ім’я, прізвище, предмет, який викладає репетитор, оплату за годину та опис.

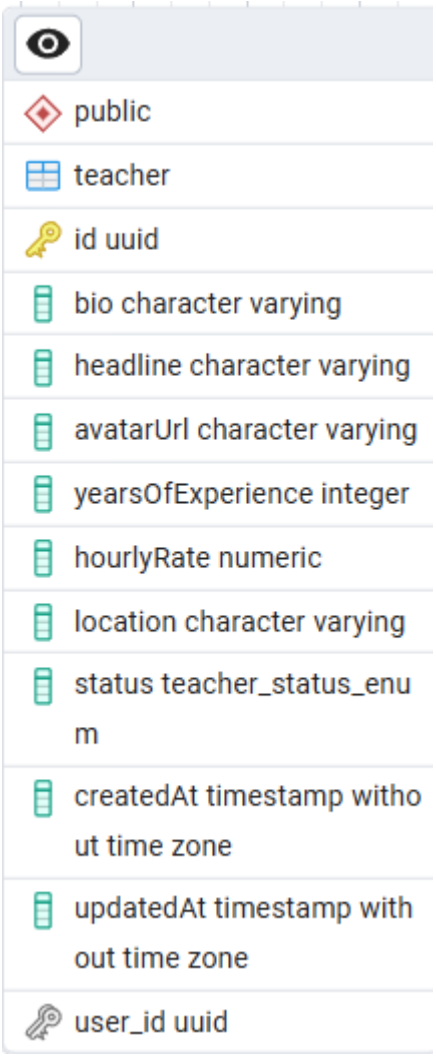

Рис 3.5. Таблиця «Вчителі»

Таблицю «Вчителі» було створено за допомогою наступного SQL запиту:

CREATE TABLE public.teacher (
id uuid DEFAULT uuid_generate_v4() NOT NULL,
bio varchar NULL,
headline varchar NULL,
"avatarUrl" varchar NULL,
"yearsOfExperience" int4 NULL,

"hourlyRate" numeric NULL,
"location" varchar NULL,
status public."teacher_status_enum" DEFAULT 'PENDING'::teacher_status_enum NOT NULL,
"createdAt" timestamp DEFAULT now() NOT NULL,
"updatedAt" timestamp DEFAULT now() NOT NULL,
user_id uuid NULL,
CONSTRAINT "PK_2f807294148612a9751dacf1026" PRIMARY KEY (id),
CONSTRAINT "REL_93f6fa64874b010c5f3a87c3b8" UNIQUE (user_id),
CONSTRAINT "FK_93f6fa64874b010c5f3a87c3b8b" FOREIGN KEY (user_id) REFERENCES public.users(id) ON DELETE CASCADE
);

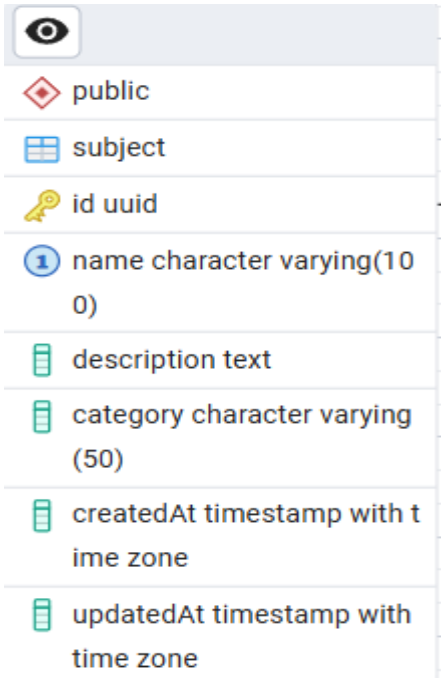

Рис. 3.6. Таблиця «Предмети»

Таблиця «Предмети» відображає інформацію про предмети, за якими можна знайти репетитора на веб-платформі.

SQL запит для створення таблиці «Предмети»:

CREATE TABLE public.subject (
id uuid DEFAULT uuid_generate_v4() NOT NULL,
name varchar(100) NOT NULL,
description text NULL,
category varchar(50) NULL,
"createdAt" timestamp DEFAULT now() NOT NULL,
"updatedAt" timestamp DEFAULT now() NOT NULL,

CONSTRAINT "PK_12eee115462e38d62e5455fc054" PRIMARY KEY (id),
CONSTRAINT "UQ_d011c391e37d9a5e63e8b04c977" UNIQUE (name)
);

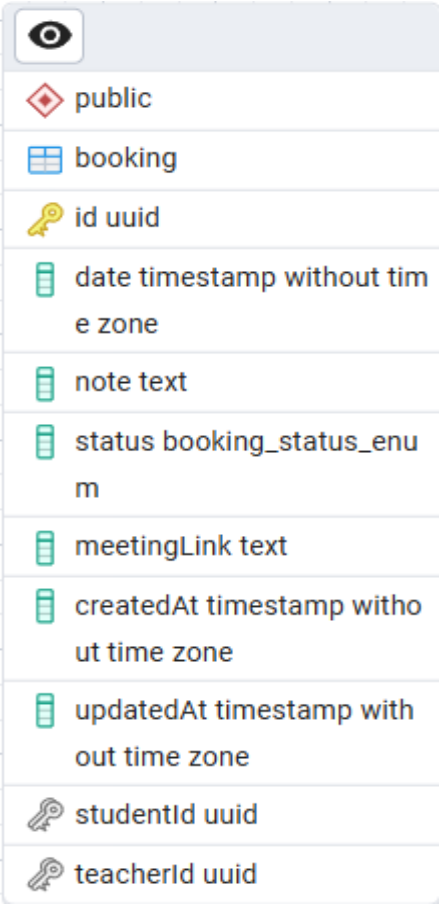

Рис. 3.7. Таблиця «Бронювання»

Таблиця «Заявки» зберігає інформацію про заявки на навчання, залишені користувачами сайту. Таблиця передає інформацію про репетитора, до якого було подано заявку, учня, що подав заявку, вік учня, рівень знань учня, мету навчання, номер телефону учня та бажаний час заняття.

Таблицю «Бронювання» було створено за допомогою наступного SQL запиту:

CREATE TABLE public.booking (
id uuid DEFAULT uuid_generate_v4() NOT NULL,
"date" timestamp NOT NULL,
note text NULL,
status public."booking_status_enum" DEFAULT 'pending'::booking_status_enum NOT NULL,
"createdAt" timestamp DEFAULT now() NOT NULL,

"updatedAt" timestamp DEFAULT now() NOT NULL,
"studentId" uuid NULL,
"teacherId" uuid NULL,
"meetingLink" text NULL,
CONSTRAINT "PK_49171efc69702ed84c812f33540" PRIMARY KEY (id),
CONSTRAINT "FK_16524af86c0e3b3dd66517eec6a" FOREIGN KEY ("studentId") REFERENCES public.student(id) ON DELETE CASCADE,
CONSTRAINT "FK_9f46fa28cd1be174e2ad12bc2a3" FOREIGN KEY ("teacherId") REFERENCES public.teacher(id) ON DELETE CASCADE
);

Також, було створено допоміжні зв'язуючі таблиці:

1. Таблиця public.teacher_subjects (Зв’язок Репетитор–Предмет):

Таблиця типу «багато-до-багатьох» (Many-to-Many), яка дозволяє одному репетитору викладати декілька предметів, а одному предмету — мати кількох викладачів.

2. Таблиця public.availability (Графік Репетитора):

Зберігає часові слоти, коли репетитор доступний для проведення уроків. Ця інформація є основою для функціоналу календаря та бронювання занять.

3. Таблиця public.education (Освіта Репетитора):

Містить дані про формальну освіту репетитора. Інформація допомагає підвищити довіру студентів до профілю.

4. Таблиця public.experience (Досвід Репетитора) :

Зберігає записи про професійний досвід репетитора, що не пов’язаний з формальною освітою, наприклад, роботу за фахом.

5. Таблиця public.review (Відгуки та Рейтинги) :

Фіксує зворотний зв’язок щодо якості наданих послуг і є ключовою для системи рейтингу репетиторів.

Усі допоміжні таблиці пов’язані з основним профілем репетитора (teacher_id) або студента (student_id). Це дозволяє швидко агрегувати дані про кваліфікацію, розклад та рейтинг для відображення у профілі користувача.

Отже, в результаті роботи було створено схему даних та логічну модель інформаційної системи для пошуку репетиторів. Логічна модель включає

чотири таблиці, зв’язані між собою зв’язком один-до-багатьох. Розробка логічної моделі виконувалась за допомогою інструменту MySQL Front, який є зручний в використанні, має інтуїтивно зрозумілий інтерфейс, дозволяє легко створювати первинні та зовнішні ключі.

3.2. Програмна реалізація

Після входу на платформу користувач потрапляє на головну сторінку, на якій відображається список предметів, що дозволяє користувачу одразу ознайомитися з доступними напрямками навчання. Користувач може переглянути доступних репетиторів за конкретним предметом або зареєструватися для подальшого користування платформою.

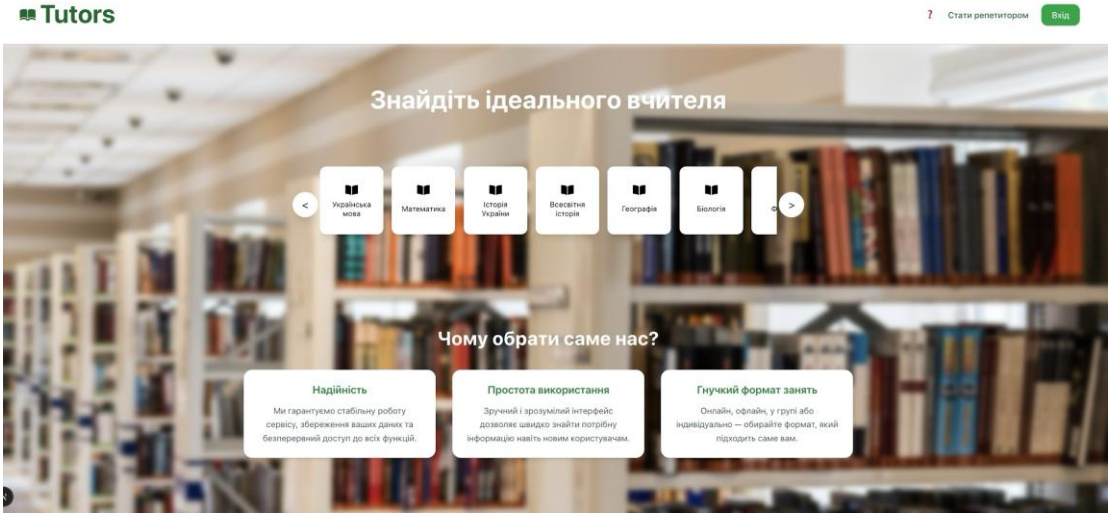

Рис. 3.8. Головна сторінка вебплатформи

Для забезпечення безпеки доступу до сайту було розроблено функціонал авторизації та реєстрації користувачів, що дозволяє користувачам створювати обліковий запис та отримувати доступ до функцій сайту.

При вході на сайт користувачеві доступні функції авторизації або реєстрації. Авторизація відбувається за допомогою спеціальної форми.

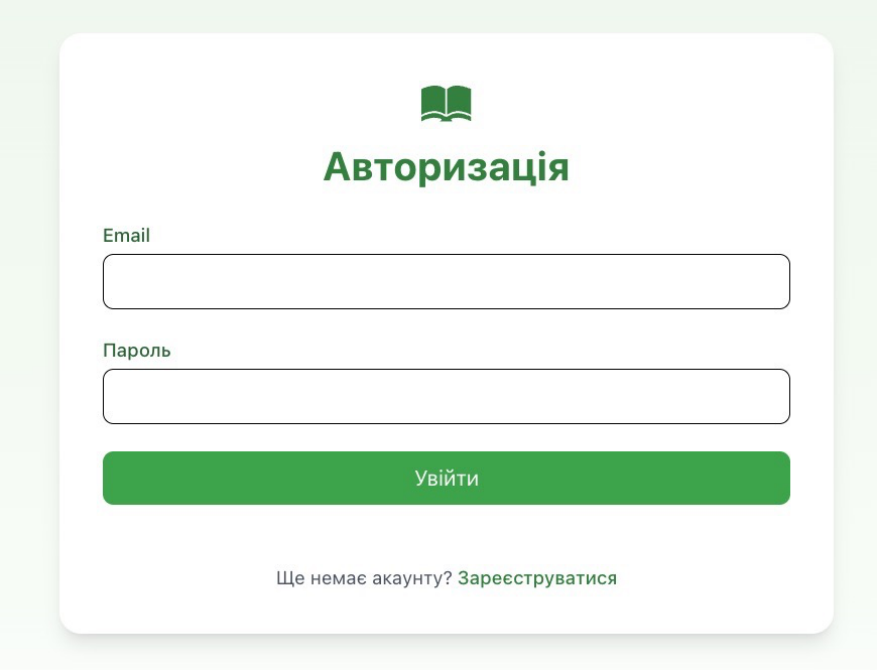

Рис. 3.9. Форма авторизації користувача

Взаємодія форми для авторизації користувачів сайту з базою даних реалізовано за допомогою наступного auth модуля. Даний модуль включає логіку авторизації, видання токенів та перевірки вхідних даних.

```
@HttpCode(HttpStatus.OK)
 @Post('login')
 async login(@Body() loginUserDto: LoginUserDto, @Res() res: Response)
{
  const { accessToken, refreshToken, user } =
   await this.authService.login(loginUserDto);
  res.cookie(TokenName.ACCESS, accessToken, {
   httpOnly: true,
   secure: process.env.NODE_ENV === Environment.PRODUCTION,
   sameSite: 'strict',
   maxAge: ACCESS_TOKEN_EXPIRATION,
  });

  res.cookie(TokenName.REFRESH, refreshToken, {
   httpOnly: true,
   secure: process.env.NODE_ENV === Environment.PRODUCTION,
   sameSite: 'strict',
   maxAge: REFRESH_TOKEN_EXPIRATION,
  });
  const userDto = plainToInstance(UserResponseDto, user, {
   excludeExtraneousValues: true,
  });
```

```
  return res.send({ message: 'Login successful', userDto });
 }
```

На платформі передбачено два окремі процеси реєстрації — для студентів та для викладачів. Реєстрація студента здійснюється через відповідну форму, де користувач вводить необхідні дані, такі як ім’я, прізвище, електронну пошту, пароль, цілі навчання. Для викладачів доступна окрема форма для створення облікового запису репетитора. Під час реєстрації викладач зазначає персональні дані, педагогічний досвід, розмір погодинної оплати.

При реєстрації користувача на сайті встановлюється модифікатор доступу “STUDENT” або “TEACHER”, в залежності від форми реєстрації. Розмежування полягає в різних функція у системі, у відмінності доступних сторінок для відображення, і валідації ендпоінтів на сервері. Валідація ролей здійснюється на клієнті, так як технологія NextJS включає серверний рендеринг сторінок:

```
const user = useAuthStore((state) => state.user);

 const isTeacher =
  Array.isArray(user?.role) && user.role.includes(Role.Teacher);
 useEffect(() => {
  if (!isAuthLoading && (!user || !isTeacher)) {
   router.replace(ROUTES.LOGIN);
  }
 }, [user, isTeacher, isAuthLoading, router]);

if (isTeacher && user) {
  return (
   <div className="flex min-h-screen">
    <TeacherSidebar />
    <main className="flex-1 p-8">{children}</main>
   </div>
  );
 }
```

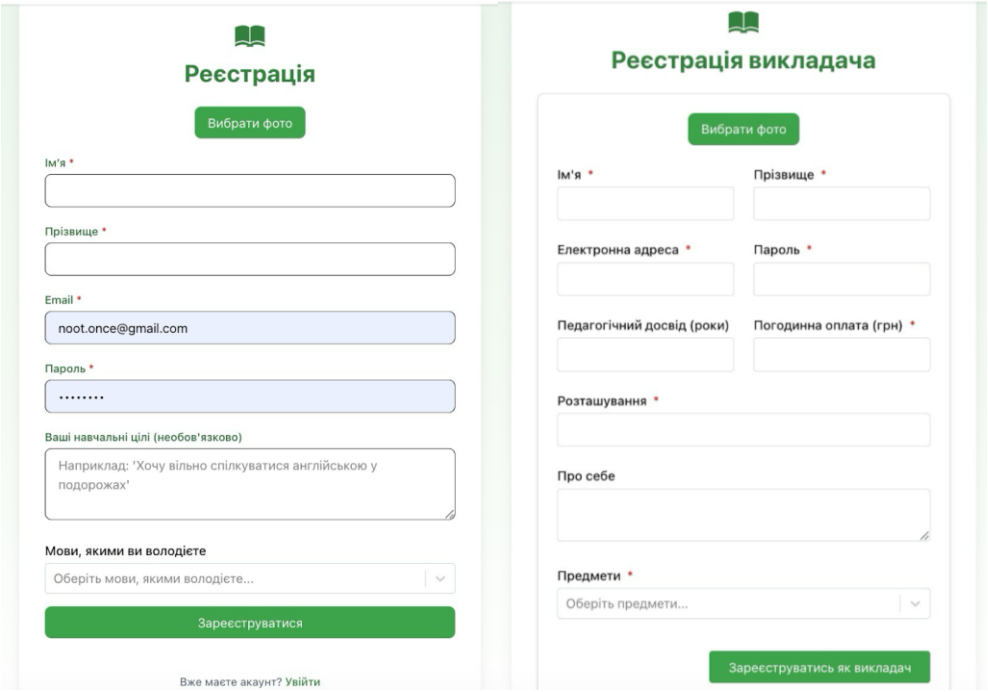

Рис. 3.10. Форма реєстрації студента та викладача

Функція реєстрації нового користувача в модулі авторизації створює обліковий запис у базі даних, а також додає відповідний профіль — студента або викладача, залежно від вибраної ролі. Далі наведено код, що реалізує цей процес:

```
public async register(
    data: CreateStudentDto | CreateTeacherDto,
  ): Promise<AuthResponse> {
    let teacher: Teacher | undefined;
    let student: Student | undefined;
    const user = await this.userService.createUser(data);

    if (data.role === Role.Teacher) {
      teacher = await this.teacherService.createTeacher(data, user);
    }

    if (data.role === Role.Student) {
      student = await this.studentService.createStudent(data, user);
    }

    const payload = this.generatePayload(user);
    const tokens = await this.tokenService.generateTokens(payload);

    return { user, teacher, student, ...tokens };
  }
```

Навігація сайту здійснюється за допомогою меню, яке розташоване в верхній частині сторінки. Меню містить посилання на основні розділи сайту, а саме:

- Для студента: профіль користувача, головна сторінка, сповіщення, допомога, вихід з акаунту.

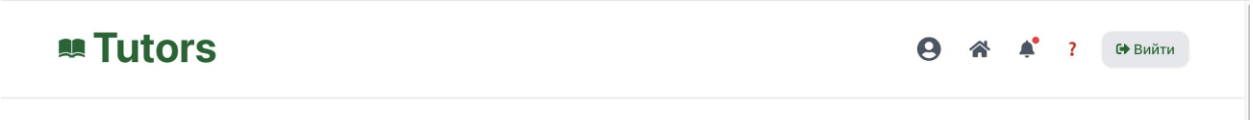

Рис. 3.11. Навігаційне меню для студента

- Для викладача: профіль викладача, календар, редагування профілю, мої студенти, допомога, віхід з акаунту.

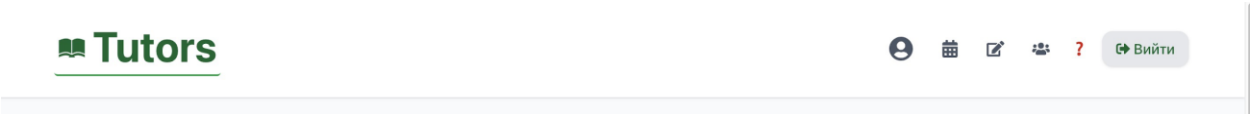

Рис. 3.12. Навігаційне меню для викладача

Для перегляду репетиторів користувач переходить на сторінку пошуку репетиторів, де представлений список доступних викладачів. На цій сторінці можна здійснювати пошук за конкретним предметом, а також відсортовувати результати за такими критеріями, як ціна, дата створення профілю, досвід викладача, рейтинг.

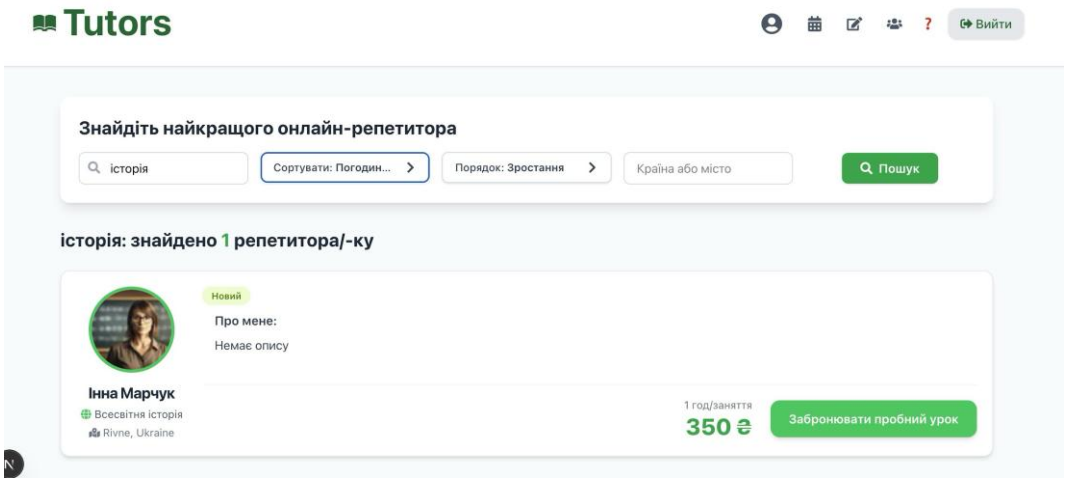

Рис. 3.13. Сторінка пошуку репетиторів

Для обробки запитів користувачів і отримання списку викладачів з фільтрацією, сортуванням та пагінацією у платформі використовується метод

контролера, який викликає метод сервісу. Сервіс, в свою чергу, приймає параметри ліміту, відступу, проводить LEFT JOIN у таблиці, фільтрує дані і повертає коректний масив об’єктів на клієнт.

```
@Get()
async getTeachers(@Query() query: GetTeachersQueryDto) {
   return this.teacherService.findAllSorted(query);
}
```

За допомогою кнопки «Забронювати пробний урок» користувач може залишити заявку на заняття в репетитора через спеціальну форму.

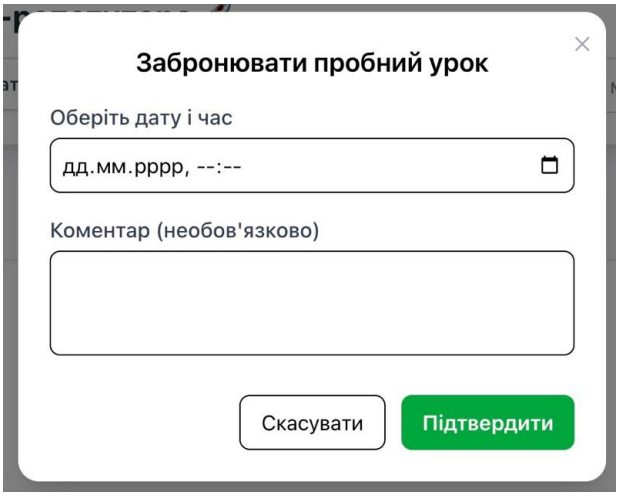

Рис. 3.14. Форма подачі заявки на заняття з репетитором

Бронювання створюється та зберігається у базі даних за допомогою наступної функції:

```
const teacher = await this.teacherService.findProfileByUserId(
    dto.teacherId,
  );
  if (!teacher) {
    console.error(`Teacher not found for id: ${dto.teacherId}`);
    throw new NotFoundException(
      `Teacher profile not found for id ${dto.teacherId}`,
    );
  }
  const booking = this.bookingRepository.create({
    student,
    teacher,
    date: new Date(dto.date),
    note: dto.note,
```

```
  });

  return this.bookingRepository.save(booking);
}
```

Після відправки заявки на бронювання уроку, в профілі викладача відображається відповідна заявка на сторінці «Мої студенти».

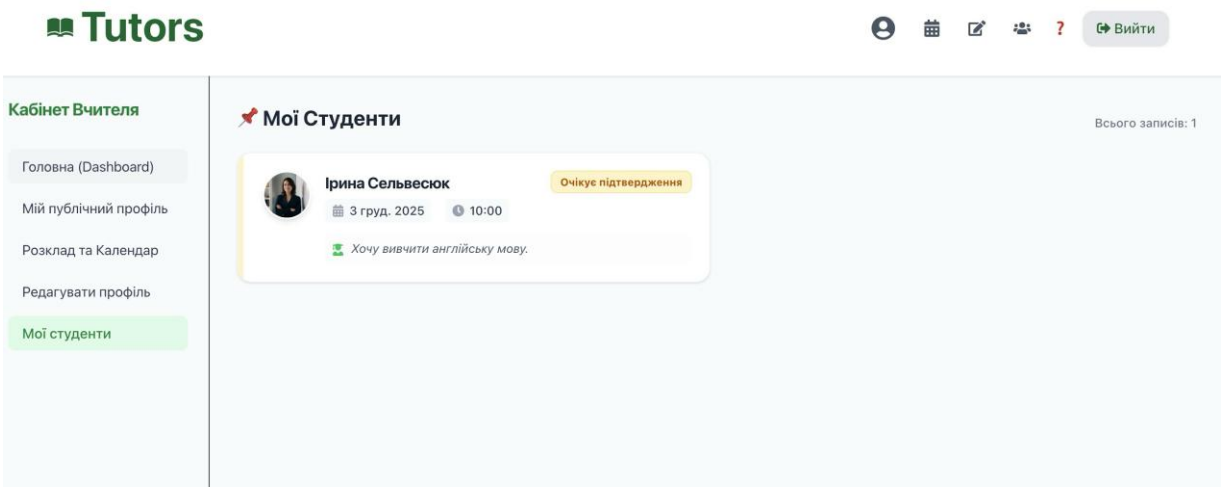

Рис. 3.15. Форма подачі заявки на заняття з репетитором

Викладач може підтвердити бронювання за допомогою спеціального вікна.

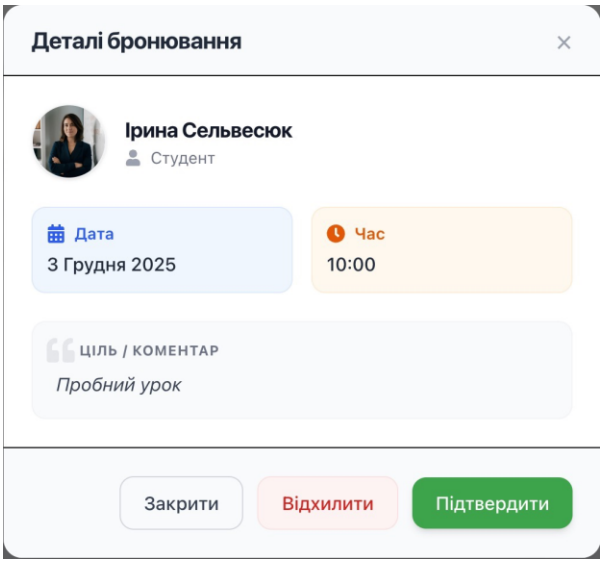

Рис. 3.16. Форма подачі заявки на заняття з репетитором

Після підтвердження, студент отримує на пошту відповідний лист з посиланням на Google Meet.

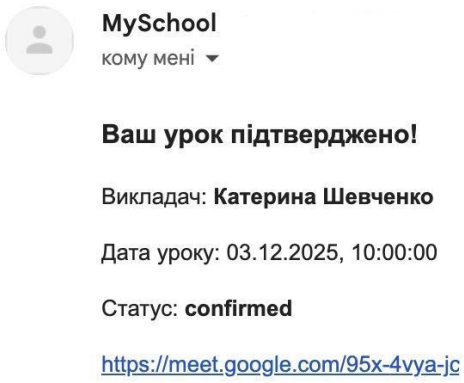

Рис. 3.17. Лист підтвердження проведення заняття

Даний функціонал реалізований за допомогою SMTP протоколу в поєднанні з наступним кодом:

```
const info: SMTPTransport.SentMessageInfo =
      await this.transporter.sendMail({
       from: `"MySchool" <${process.env.MAIL_USER!}>`,
       to,
       subject,
       html,
      });

    this.logger.log(`Email sent: ${info.messageId}`);
```

Заняття буде доступне викладачу на сторінці «Розклад та Календар».

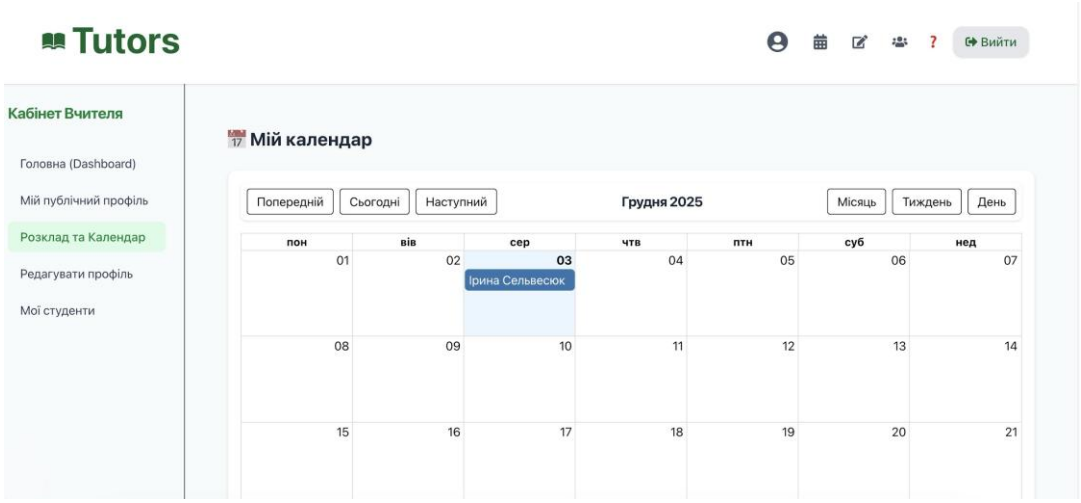

Рис. 3.18. Форма подачі заявки на заняття з репетитором

Функціонал для отримання всіх бронювань викладача з бази даних реалізовано за допомогою методу:

```
async getBookingsForTeacher(userId: string) {
   const teacher = await this.teacherService.findProfileByUserId(userId);
```

```
  return this.bookingRepository.find({
    where: { teacher: { id: teacher.id } },
    relations: { student: { user: true } },
    order: { date: 'ASC' },
  });
}
```

Посилання на гугл-зустріч реалізовано за допомогою коду:

```
const event: calendar_v3.Schema$Event = {
      summary: title,
      description: `${description}\n\ Google Meet link: ${meetLink}`,
      start: {
        dateTime: startTime.toISOString(),
        timeZone: 'UTC',
      },
      end: {
        dateTime: endTime.toISOString(),
        timeZone: 'UTC',
      },
    };

    await this.calendar.events.insert({
      calendarId: this.CALENDAR_ID,
      requestBody: event,
    });
```

На платформі користувач може переглянути свій профіль на спеціальній сторінці, де відображається інформація про кількість студентів, найближчі уроки, проведені уроки,підтверджені уроки та минулі уроки.

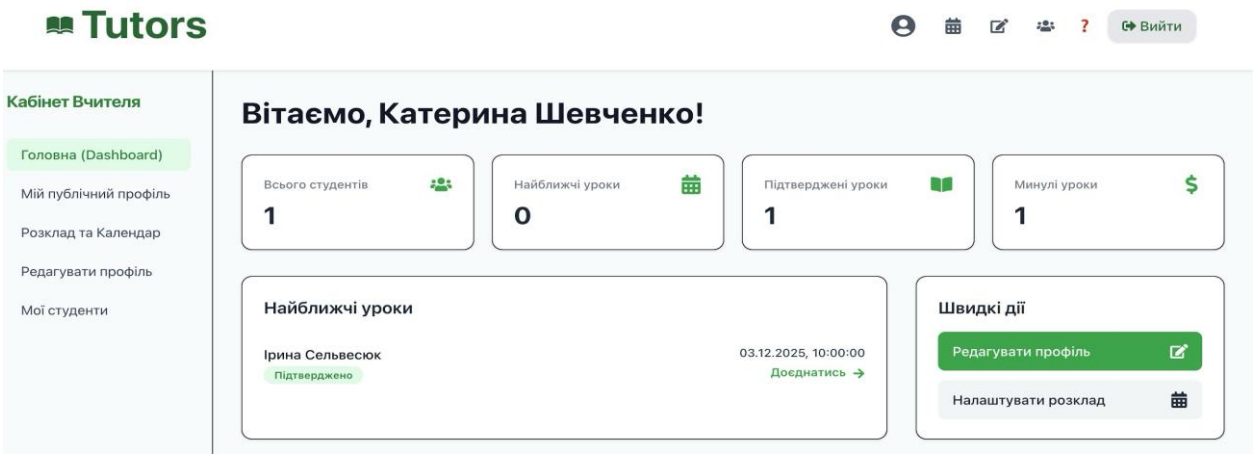

Рис. 3.19. Головна сторінка репетитора, з загальною інформацією

Також, розроблено функціонал редагування профілю за допомогою відповідного пункту навігаційного меню, або кнопки на сторінці профілю.

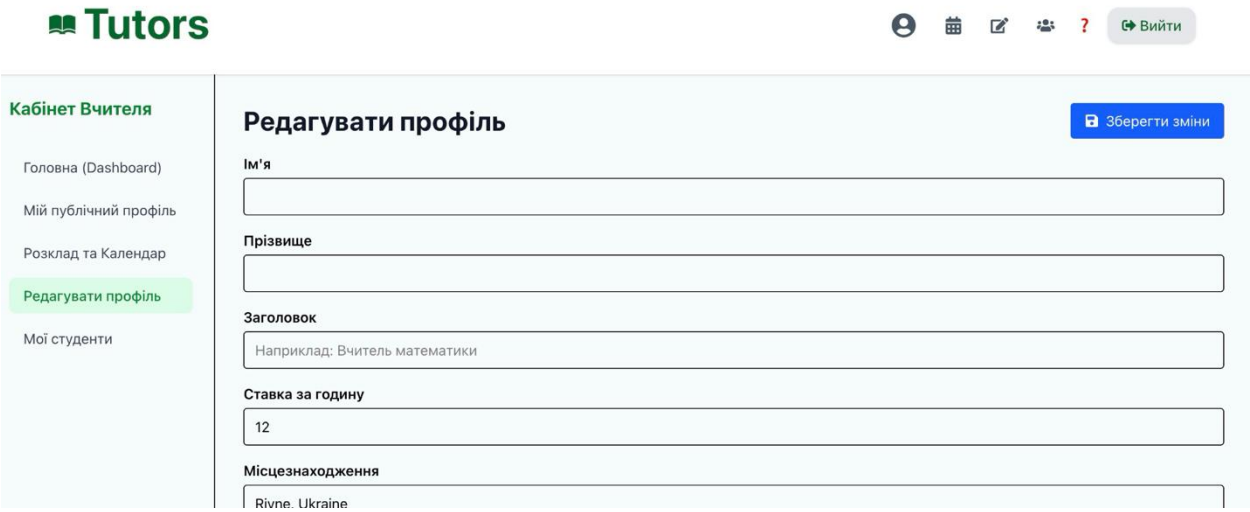

Рис. 3.20. Форма редагування даних профілю репетитора

Можна переглянути свій профіль, який буде відображатись учням за допомогою сторінки «Мій публічний профіль».

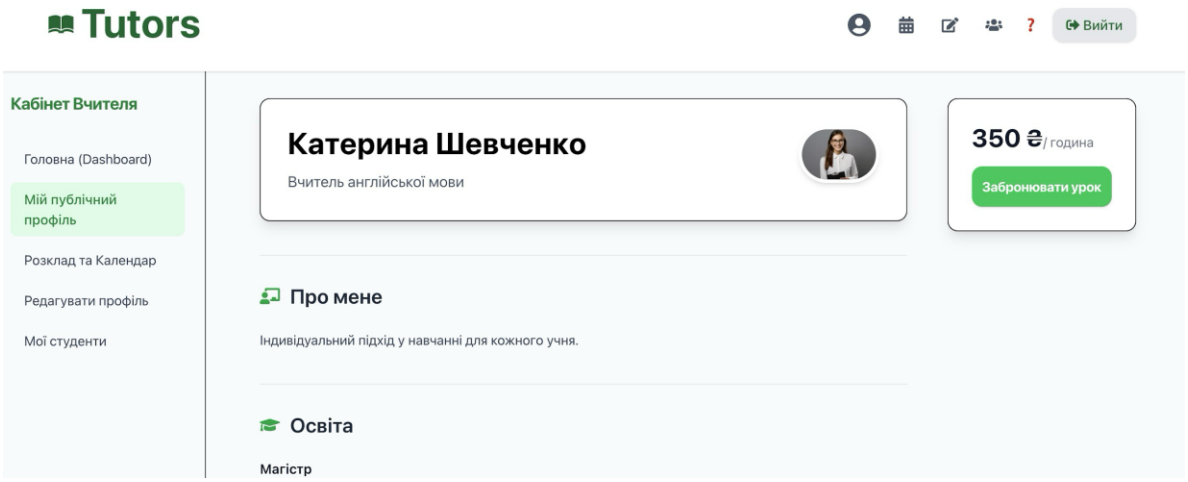

Рис 3.21. Сторінка загальнодоступного профілю репетитора

Для студента реалізовано можливість залиши відгук за допомогою спеціальної форми:

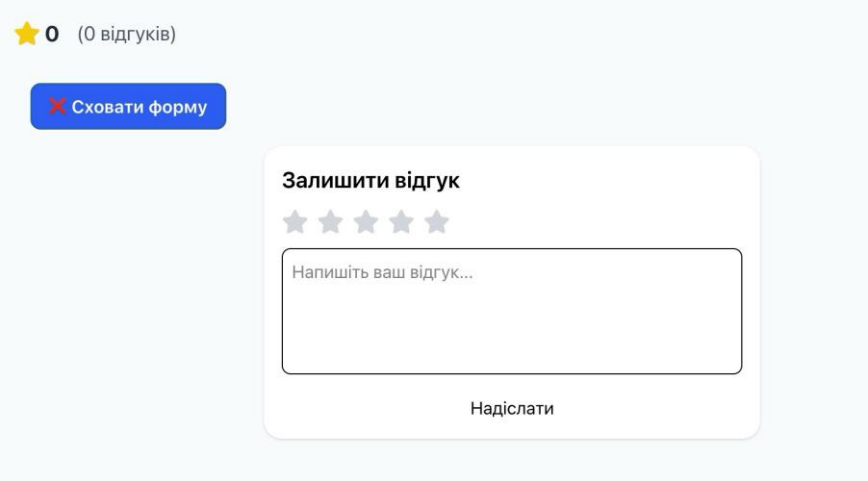

Рис 3.22. Форма відгуку

Відгук відобразиться в профілі викладача з відповідним рейтингом.

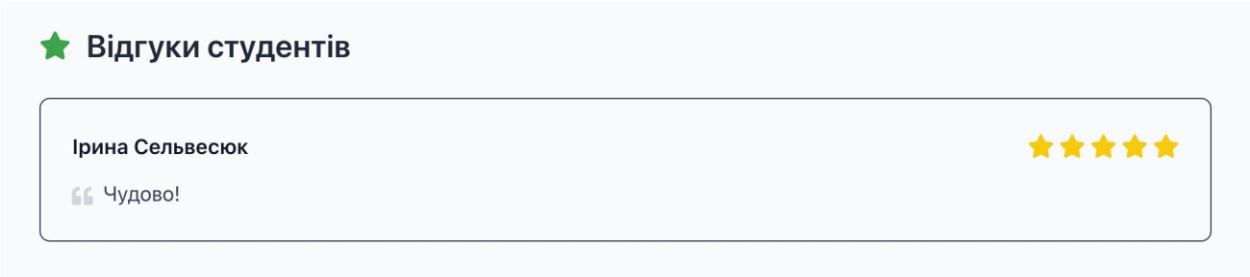

Рис 3.23. Відображення відгуку на сторінці репетитора

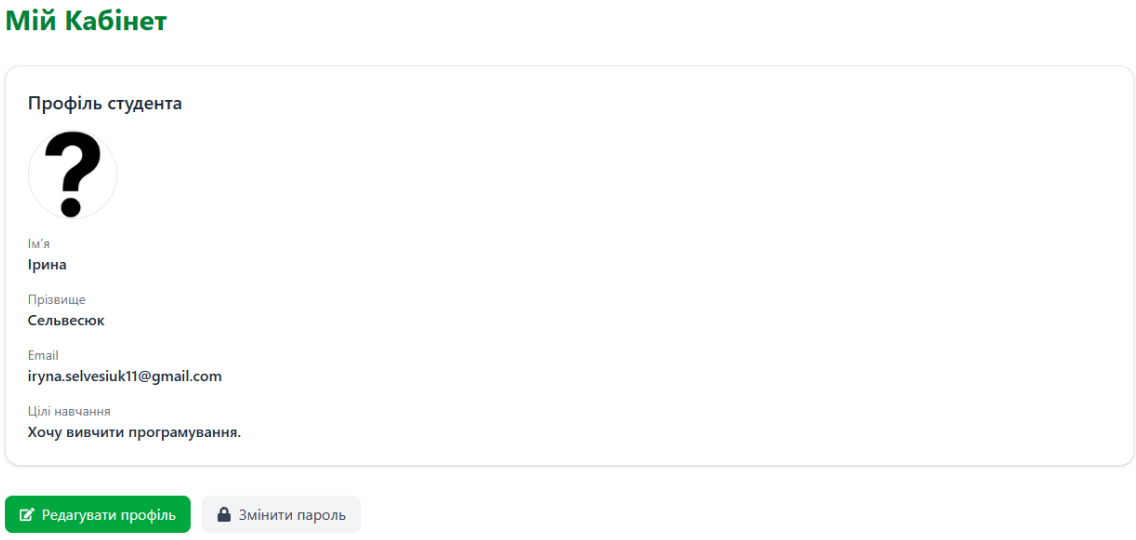

Рис 3.24. Особистий профіль студента

Сторінка особистого профілю студента включає в себе відображення інформації, яку користувач надав при реєстрації, фото, та двох кнопок - редагувати сторінку, та змінити пароль.

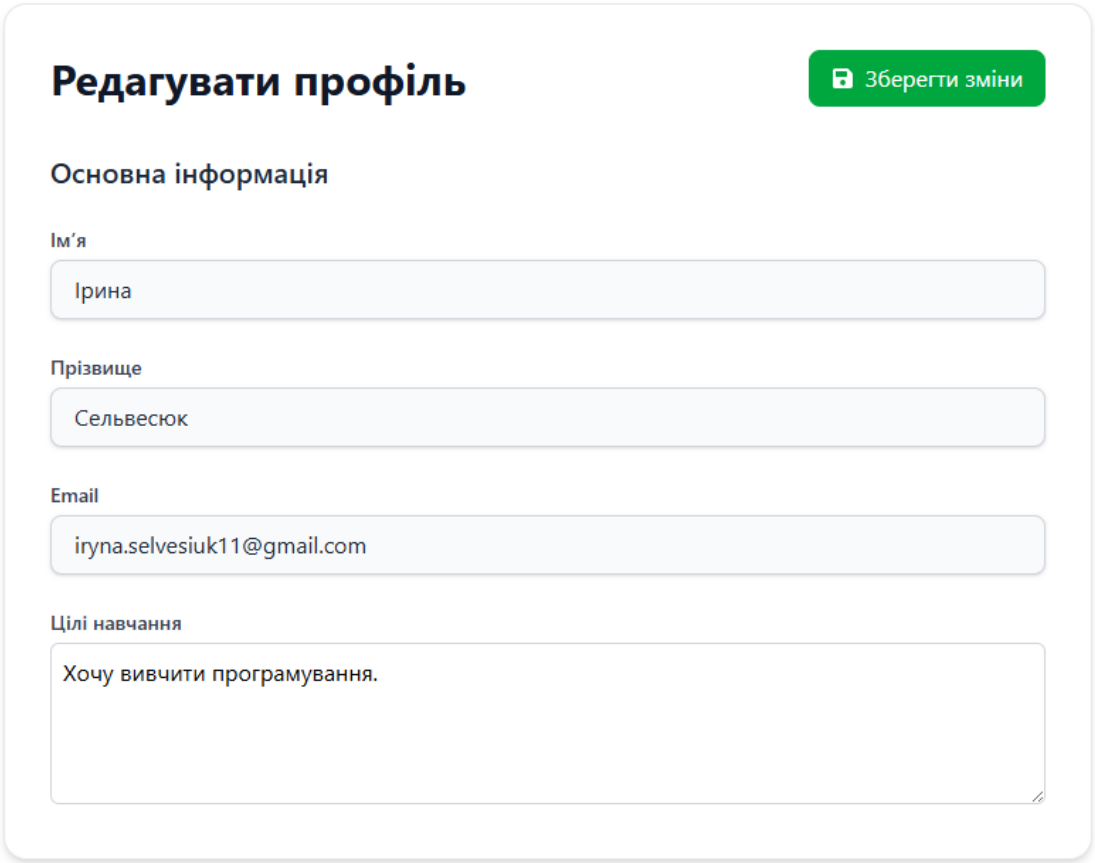

Рис 3.25. Форма редагування профілю студента

Форма включає в себе валідацію даних у полях, та відправлення даних для оновлення через PATCH запит на сервер. Використовується двостороння валідація на клієнті та на севері.

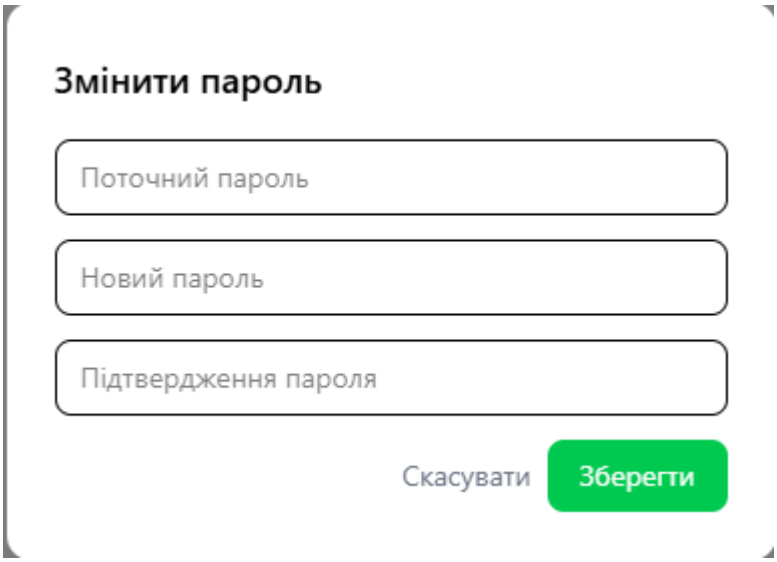

Рис 3.26. Модальне вікно зміни паролю

Модальне вікно зміни паролю включає в себе 3 поля, так як будь-які маніпуляції з чутливими даними користувача потребуються більш строгої валідації. Реалізовано за допомогою наступного коду контроллеру:

```
@UseGuards(JwtAuthGuard)
@Patch('change-password')
async changePassword(
  @Req() req: RequestWithUser,
  @Body() body: { currentPassword: string; newPassword: string },
) {
  const userId = req.user.id;

  return this.userService.updatePassword(
    userId,
    body.currentPassword,
    body.newPassword,
  );
}
```

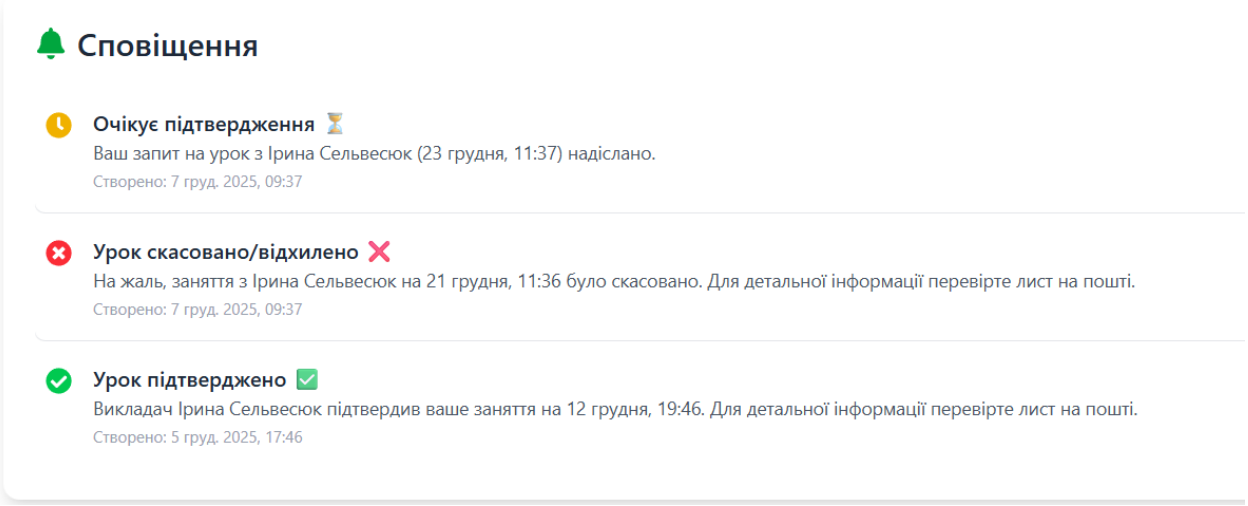

Рис 3.27. Сторінка сповіщень користувача

Сторінка сповіщень дозволяє студенту переглядати статуси своїх бронювань. Існує 3 статуси бронювань, кожен відображається з відповідним стилем.

```
switch (status) {
  case 'confirmed':
    return {
      type: 'success',
      icon: <FaCheckCircle className="text-green-500" />,
      title: 'Урок підтверджено',
      message: `Викладач ${teacherName} підтвердив ваше заняття на ${formattedDate}. Для детальної інформації перевірте лист на пошті.`,
    };
  case 'rejected':
    return {
      type: 'error',
      icon: <FaTimesCircle className="text-red-500" />,
      title: 'Урок скасовано/відхилено',
      message: `На жаль, заняття з ${teacherName} на ${formattedDate} було скасовано. Для детальної інформації перевірте лист на пошті.`,
    };
  case 'pending':
  default:
    return {
      type: 'info',
      icon: <FaClock className="text-yellow-500" />,
      title: 'Очікує підтвердження',
      message: `Ваш запит на урок з ${teacherName} (${formattedDate}) надіслано.`,
    };
```

}

Отже, розроблені функціональні можливості системи забезпечують ефективну взаємодію користувачів з платформою. Користувачі мають можливість переглядати доступні предмети, обирати репетиторів, залишат заявки на навчання.

Завдяки механізму розмежування прав, система забезпечує чітке відокремлення функцій для різних категорій користувачів: адміністраторів та звичайних користувачів. Адміністратор має доступ до окремих даних, таких як заявки учнів, і може редагувати інформацію про репетиторів, що забезпечує ефективне управління контентом сайту.

3.3. Комплексне тестування функціональності вебплатформи

У межах комплексного тестування було проведено декілька видів перевірок, що охоплюють як функціональні, так і нефункціональні аспекти роботи вебплатформи. Функціональне тестування включило в себе перевірку всіх основних сценаріїв використання системи. Було протестовано:

1. Реєстрацію студентів та викладачів: коректність валідації даних, перевірку форматів електронної пошти, унікальність користувача, обробку помилок.
2. Авторизацію: правильність перевірки облікових даних, обробку неправильних логінів/паролів, відновлення доступу.
3. Пошук викладачів: роботу фільтрів, сортування, коректну передачу параметрів із клієнтської частини на сервер.
4. Функціональність профілю викладача: можливість редагування інформації, додавання освіти, досвіду роботи, ставки за годину.
5. Систему відгуків: оцінювання викладача, відображення рейтингу викладача.

Тестування проводилось на різних типах облікових записів (студент, викладач), що дало змогу оцінити коректність роботи розмежування прав доступу.

Розроблено чекліст для тестування додатку з розподілом на функції та визначенням відповідних типів, методів і рівнів тестування.

Таблиця 3.2

Чекліст тестування вебплатформи

Tutors App		**Статус**
1. Реєстрація		
1.a	Перевірка валідації	Passed
1.b	Усі поля порожні	Passed
1.c	Невірний формат email	Passed
1.d	Email вже зареєстровано	Passed
1.e	Пароль менше 8 символів	Passed
2. Авторизація		
2.a	Успішна авторизація з валідними даними	Passed
2.b	Авторизація з некоректним паролем	Passed
2.c	Авторизація з неіснуючим email	Passed
2.d	Авторизація без введення даних	Passed
2.e	Вихід із системи	Passed
3. Особистий кабінет студента		
3.a	Редагування профілю та успішне збереження	Passed
3.b	Перегляд списку заброньованих занять	Passed
3.c	Зміна пароля	Passed
4. Пошук викладачів		
4.a	Пошук викладача за іменем	Passed
4.b	Пошук викладача за предметом	Passed
4.c	Пошук без результатів – відображення повідомлення	Passed
4.d	Пошук з порожнім запитом – повідомлення/початковий список	Passed
4.e	Тестування сортування викладачів за рейтингом / ціною	Passed
5. Відображення викладачів		
5.a	Коректне фото, ім’я, предмет, рейтинг	Passed
5.b	Перехід на сторінку профілю викладача	Passed
5.c	Відображення ціни за годину	Passed
6. Бронювання заняття		
6.a	Успішне бронювання доступного слоту	Passed

6.b	Бронювання слоту, який уже зайнятий іншим студентом	Passed
6.c	Спроба бронювання без авторизації	Passed
6.e	Неможливість бронювати заняття у минулому часі	Passed

Для забезпечення повного охоплення функціональних можливостей вебплатформи було розроблено набір тест-кейсів.

Таблиця 3.3

Приклад створених тест-кейсів для тестування вебплатформи

ID	Назва	Передумови	Кроки тесту	Очікуваний результат
01	Реєстрація студента з валідними даними	Користувач знаходиться на сторінці реєстрації	1. Заповнити всі поля валідними даними. 2. Натиснути «Зареєструватися ».	Створюється новий акаунт, користувача перенаправляє на сторінку профілю/авторизації.
02	Реєстрація з уже існуючою електронною поштою	У базі існує акаунт з цією поштою	1. Ввести email, що вже використовується .2. Заповнити інші поля.3. Натиснути «Зареєструватися ».	Відображається повідомлення «Користувач з такою поштою вже існує».
03	Авторизація з валідними даними	Користувач зареєстрований	1. Ввести email і пароль.2. Натиснути «Увійти».	Користувач успішно входить у систему.
04	Авторизація з неправильним паролем	Користувач зареєстрований	1. Ввести правильний email і неправильний пароль.2. Натиснути «Увійти».	Відображається повідомлення «Невірний пароль».
05	Пошук викладача за предметом	Користувач авторизований	1. Ввести предмет у поле пошуку (наприклад, «Математика»).2.	Відображається список викладачів за обраним предметом.

			Натиснути «Пошук».	
06	Використання фільтра за ціною	Є список викладачів	1. Вибрати фільтр «Ціна» та встановити діапазон.2. Застосувати фільтр.	Список викладачів оновлюється згідно з фільтром.
07	Перегляд публічного профілю викладача	Користувач авторизований	1. Відкрити сторінку викладача зі списку.2. Переглянути фото, опис, досвід, відгуки.	Профіль викладача успішно відображається.
08	Бронювання уроку	Користувач авторизований	1. Відкрити профіль викладача.2. Натиснути «Записатися на урок».3. Вибрати дату та час.4. Підтвердити.	Урок успішно заброньовано, студент отримує підтвердження.
09	Спроба бронювання без авторизації	Користувач не авторизований	1. Натиснути «Записатися на урок».	Система перенаправляє на сторінку авторизації.
10	Додавання відгуку після заняття	Студент має завершений урок	1. Відкрити профіль викладача.2. Натиснути «Залишити відгук».3. Ввести оцінку та текст.4. Підтвердити.	Відгук зберігається та відображається у профілі викладача.

Під час тестування було виявлено кілька дефектів вебплатформи:

- некоректно розміщені UI-елементи;
- пароль, що складався лише з пробілів, проходив початкову перевірку на мінімальну довжину.
- дрібні текстові неточності;

- при повторній спробі авторизації після помилки попереднє повідомлення не зникало без перезавантаження сторінки;
- система дозволяла вибрати дату заняття у минулому;
- при виборі декількох фільтрів одночасно (предмет + ціна + досвід) інколи застосовувався лише один із них.

Після їх виправлення, повторне тестування підтвердило відсутність помилок. Тестування забезпечило охоплення як функціональних, так і нефункціональних аспектів системи.

ВИСНОВКИ

В процесі виконання магістерської роботи було розроблено веб-платформу координації стейкхолдерів онлайн-репетиторства, яка забезпечує ефективну взаємодію між студентами та викладачами. У ході дослідження було проаналізовано сутність онлайн-репетиторства, розглянуто особливості організації дистанційного навчання, а також проведено огляд сучасних платформ та інструментів, що використовуються у сфері онлайн-освіти.

Зокрема, було реалізовано функціонал авторизації та реєстрації користувачів. Користувачі можуть переглядати доступні предмети та вибирати репетиторів відповідно до своїх потреб, а також залишати заявки на навчання. Для реалізації програмної частини були застосовані сучасні технології, такі як Next.js, Nest.js, TypeORM та PostgreSQL, що забезпечило високу продуктивність, безпеку та масштабованість системи.

Особливу увагу приділено тестуванню функціональності вебплатформи. Було розроблено та виконано тест-кейси для перевірки коректності роботи ключових модулів системи. Проведене тестування дало змогу виявити та усунути дефекти, підвищивши зручність використання системи.

Отримані результати підтверджують, що реалізована вебплатформа успішно виконує поставлені завдання та може бути використана як інструмент для організації та координації онлайн-репетиторства. Вона відповідає сучасним вимогам до вебсистем та має потенціал подальшого розвитку. Перспективними напрямами удосконалення є інтеграція системи онлайн-оплат для автоматизації фінансових розрахунків між стейкхолдерами, та впровадження системи аналітики для моніторингу ефективності навчального процесу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Петрощук Н., Матвійчук О., Карчина Л. Проблеми становлення дистанційного навчання в Україні (за матеріалами соціологічного дослідження). Науково-педагогічні студії, 2021, № 5, С. 53–65. DOI: 10.32405/2663-5739-2021-5-53-65.
2. МесарошЛ. Дослідження поглядів учнів та вчителів стосовно дистанційного навчання. Освіта. Інноватика. Практика, 2025. Том13, No3. С. 52-56.
3. Ovcharuk O., Ivaniuk I., Leshchenko M. Impact of school lockdown on access to online instruction during the war in Ukraine. European Journal of Education, 2023. DOI: 10.1111/ejed.12589.
4. Shuliak I., Ostapchuk I., García Laborda J. Online education in Ukraine in extreme conditions: constraints and challenges. Computer Assisted Language Learning Electronic Journal (CALL-EJ), 2024, Т. 25, № 1, С. 208–227.
5. International Software Testing Qualifications Board (ISTQB). Certified Tester Foundation Level Syllabus v4.0.1 : навчальний посібник. 2023. 50 с.
6. Software quality — different types of software testing: веб-сайт. URL: https://www.tuleap.org/software-quality-different-types-software-testing (дата звернення: 29.11.2025).
7. Популярні інструменти для створення тест-кейсів: веб-сайт. URL: https://career.qatestlab.eu/ua/blog/popular-tools-for-creating-test-cases/ (дата звернення: 29.11.2025).
8. Postman CLI Newman — автоматизація тестів API: веб-сайт. URL: https://itpraktik.com/postman-cli-newman/ (дата звернення: 10.11.2025).
9. Postman: огляд інструмента для тестування API: веб-сайт. URL: https://foxminded.ua/postman/ (дата звернення: 10.11.2025).
10. Selenium WebDriver як інструмент для автоматизованого тестування: веб-сайт. URL: https://training.qatestlab.com/blog/technical-articles/selenium-webdriver/ (дата звернення: 10.11.2025).)

11. Stocco A., Shehory O., Jahangirova G., Riccio V., Barash G., Farchi E., Saha D. Software testing in the machine learning era. Empirical Software Engineering, 2023, Т. 28, Стаття 74. DOI: 10.1007/s10664-023-10326-7.
12. Node.js v23.3.0 documentation: веб-сайт. URL: https://nodejs.org/docs/latest/api/ (дата звернення: 15.11.2025).
13. Зв'язки між таблицями: веб-сайт. URL: http://moodle.nati.org.ua/mod/book/tool/print/index.php?id=12087 (дата звернення: 15.11.2025).