Прикарпатський національний університет імені Василя Стефаника
Факультет математики та інформатики
Кафедра комп'ютерних наук та інформаційних
технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

першого (бакалаврського) рівня освіти
на тему:

"Розробка платформи тестування з генеруванням тестів при використанні штучного інтелекту"

Виконала: студентка IV курсу, ІСТ-41
спеціальності 126 "Інформаційні системи
та технології"
Гречин І.В

Науковий керівник: наук. ступінь,
доц. к.т.н. Семаньків М. В.

Рецензент: наук. ступінь,
доц. к.т.н. Іляш Ю. Ю.

Івано-Франківськ – 2025 р.

АНОТАЦІЯ

У бакалаврській роботі представлено розробку веб-платформи EduTest для автоматизованого створення тестових завдань із використанням методів штучного інтелекту. Робота спрямована на вирішення актуальних проблем організації дистанційного контролю знань, підвищення ефективності оцінювання та зменшення трудомісткості створення тестів.

Обсяг роботи — 81 сторінок.
Дипломна робота містить 3 розділи, 1 таблицю, 36 рисунки, 4 додатки.
Ключові слова: тестування знань, штучний інтелект, веб-платформа, автоматизація, EduTest.

ABSTRACT

The thesis presents the development of a web platform called EduTest designed for automated creation, execution, and assessment of test tasks using artificial intelligence methods. The work aims to solve current challenges of organizing distance knowledge assessment, improving the efficiency of evaluation, and reducing the complexity of test preparation.

The volume of the work is 81 pages.
The thesis contains 3 chapters, 1 table, 36 figures, 4 appendices.
Keywords: knowledge testing, artificial intelligence, web platform, automation, EduTest.

ЗМІСТ

ВСТУП

Сучасний розвиток інформаційних технологій сприяє автоматизації освітніх процесів, зокрема у сфері тестування знань. Традиційні методи оцінювання, такі як письмові контрольні роботи або усне опитування, мають низку недоліків, серед яких витрати часу на перевірку, людський фактор при оцінюванні та складність масштабування. Впровадження онлайн-систем тестування дозволяє значно підвищити ефективність оцінювання знань студентів і зробити процес більш об'єктивним та зручним як для викладачів, так і для учнів.

Актуальність теми. Розробка платформ для онлайн-тестування стає все більш затребуваною, особливо у зв’язку з поширенням дистанційного навчання. Багато існуючих рішень, таких як Google Forms або Moodle, не завжди відповідають специфічним потребам викладачів і студентів, а процес створення тестів залишається трудомістким. Інтеграція штучного інтелекту для автоматичної генерації тестових завдань на основі лекційного матеріалу дозволяє значно спростити процес підготовки тестів та покращити їх якість. Це дослідження спрямоване на розробку платформи EduTest, яка дозволяє автоматично створювати тестові завдання на основі навчальних матеріалів, що робить процес навчання більш ефективним.

Об'єктом дослідження є процес автоматизованого тестування знань у навчальному процесі. *Предметом дослідження* є розробка та впровадження веб-платформи для створення та проведення тестувань із застосуванням штучного інтелекту для автоматичної генерації тестових завдань.

Метою дослідження є розробка та впровадження веб-платформи EduTest для автоматичного створення тестових завдань на основі навчальних матеріалів із використанням штучного інтелекту, що дозволяє підвищити ефективність тестування знань.

Для досягнення цієї мети необхідно вирішити такі завдання:

- розробити архітектуру системи, визначити стек технологій та структуру бази даних;
- створити веб-інтерфейс платформи для зручного керування тестами;
- реалізувати механізм генерації тестів за допомогою штучного інтелекту на основі тексту лекцій;
- протестувати ефективність автоматично згенерованих тестів та порівняти їх з традиційними методами створення тестів;
- оцінити можливості інтеграції та подальшого розвитку системи.

Методи дослідження. Для виконання роботи використано такі методи дослідження:

- теоретичний аналіз наукових джерел та існуючих систем тестування;
- порівняльний аналіз функціоналу онлайн-платформ;
- проєктування та моделювання архітектури платформи;
- експериментальне тестування роботи системи та аналіз отриманих результатів.

Структура роботи. Бакалаврська робота складається з вступу, трьох розділів основної частини, висновків, списку використаних джерел та додатків. У першому розділі аналізуються існуючі системи тестування та обґрунтовується необхідність створення власної платформи. У другому розділі описується процес розробки платформи EduTest, включно з її архітектурою, структурою бази даних, інтерфейсом та алгоритмом генерації тестів за допомогою штучного інтелекту. У третьому розділі розглядаються результати тестування платформи, аналізуються переваги та недоліки автоматично згенерованих тестів, а також визначаються можливі напрямки подальшого розвитку.

Розділ 1. ДОСЛІДЖЕННЯ ОБЛАСТІ ТА АНАЛІЗ СУЧАСНИХ ПІДХОДІВ

1.1 Еволюція та ключові особливості систем тестування знань

Ідея оцінювання знань має глибоке історичне коріння, що сягає ще античних часів. Одними з перших відомих прикладів організованого тестування були державні іспити в Стародавньому Китаї, де вже у VI столітті н.е. існувала система оцінювання кандидатів на посади в адміністрації. Ці іспити вимагали від претендентів знань класичних текстів, уміння писати есе та вирішувати логічні задачі. Таким чином, уже тоді оцінювання стало засобом відбору та перевірки кваліфікації [9].

У другій половині XX століття, з поширенням комп'ютерів, починається новий етап у розвитку систем тестування. У 1980-х роках з'являються перші комп'ютеризовані системи оцінювання, які дозволяли автоматизувати перевірку знань. Хоча на початку їх функціонал був доволі обмежений, поступово такі системи ставали все складнішими, забезпечуючи обробку результатів у реальному часі та адаптацію рівня складності завдань до користувача. Наприклад, ранні програми могли перевіряти лише тестові завдання з вибором відповіді, але з часом з'явилися можливості для аналізу відкритих відповідей та оцінки складніших завдань [9].

У 1990–2000-х роках відбувається активне впровадження вебтехнологій в освіту. З'являються перші масові платформи дистанційного навчання, такі як Moodle, Blackboard та інші. Саме в цей період тестування стає інтегрованою частиною освітніх середовищ, з'являються банки тестових завдань, механізми зворотного зв'язку, статистики та аналітики. Крім того, значна увага починає приділятись інклюзивності тестів та їхній адаптації до потреб різних категорій користувачів. Наприклад, платформи почали пропонувати функції для студентів із особливими потребами, такі як збільшення шрифту, аудіосупровід чи додатковий час на виконання завдань [9].

З 2010-х років починається активне впровадження елементів штучного інтелекту у процес тестування. Адаптивні системи починають використовувати алгоритми машинного навчання для аналізу поведінки користувача, передбачення правильності відповідей і персоналізації навчального шляху. Сучасні освітні платформи на кшталт Duolingo, Coursera, Khan Academy вже зараз демонструють можливості використання аналітики даних для створення індивідуалізованих програм навчання, де тести відіграють ключову роль як інструмент моніторингу та корекції знань. Наприклад, Duolingo адаптує уроки залежно від того, як швидко користувач засвоює нові слова, а Coursera може рекомендувати додаткові матеріали, якщо студент провалив певний модуль [2].

Останніми роками, особливо після 2020 року, значення цифрового тестування ще більше зросло у зв’язку з глобальним переходом на дистанційне навчання. Платформи з використанням ШІ, відеоаналізу, голосового розпізнавання та біометричної ідентифікації забезпечують як зручність, так і безпеку тестування. Водночас постають нові виклики, пов’язані з етичними аспектами, конфіденційністю даних і рівнем довіри до автоматизованих оцінювальних систем. Наприклад, у 2021 році деякі платформи, такі як ProctorU, зіткнулися з критикою через надмірне використання даних студентів під час онлайн-іспитів, що викликало дискусії про баланс між безпекою та приватністю. Крім того, у багатьох країнах, зокрема в Україні, де доступ до технологій може бути нерівномірним, виникають проблеми з забезпеченням однакових умов для всіх студентів під час дистанційного тестування [2].

Ще одним важливим аспектом сучасного розвитку систем тестування є їхня інтеграція з іншими технологіями, такими як віртуальна та доповнена реальність. Наприклад, у медичних вишах уже тестуються симуляції, де студенти можуть практикувати операції чи діагностику в віртуальному середовищі, а система автоматично оцінює їхні дії. Такі інновації дозволяють перевіряти не лише теоретичні знання, а й практичні навички, що раніше було складно зробити в рамках стандартних тестів. Проте ці технології поки що

доступні далеко не всюди через їхню високу вартість і потребу в спеціальному обладнанні [2].

Таким чином, розвиток систем тестування знань відбувався паралельно з розвитком технологій, педагогіки та соціальних запитів. Від іспитів при імператорських дворах до штучного інтелекту, що персоналізує навчальний досвід, — шлях тестування у світі освіти є прикладом гнучкої адаптації до змін та постійного вдосконалення. І хоча сучасні технології відкривають нові можливості, вони також ставлять перед освітянами завдання враховувати культурні, етичні та технічні особливості їхнього застосування.

Сьогодні системи тестування знань є невід'ємною частиною сучасного освітнього процесу, забезпечуючи ефективний механізм перевірки рівня підготовки студентів, слухачів курсів та працівників у корпоративному секторі. Завдяки автоматизації процесу оцінювання вони дозволяють значно зменшити навантаження на викладачів, підвищити об'єктивність виставлення оцінок та надати користувачам швидкий доступ до результатів [23].

Зараз системи тестування активно застосовуються не лише у традиційних навчальних закладах (школах, університетах), а й у дистанційному навчанні, сертифікаційних програмах, підготовці до іспитів та внутрішньо корпоративному навчанні. Вони дозволяють стандартизувати оцінювання та забезпечують єдиний підхід до перевірки знань на всіх рівнях навчального процесу. Наприклад, у великих компаніях такі системи часто використовують для оцінки нових співробітників або перевірки знань після тренінгів, що допомагає швидко визначити, чи готова людина до виконання своїх обов'язків [23].

Основними особливостями систем тестування є:

- Автоматизоване оцінювання – система самостійно перевіряє відповіді та формує результати тесту, що виключає суб'єктивність і економить час.
- Гнучкість у налаштуванні тестів – можливість створювати питання різних типів (з єдиною правильною відповіддю, множинним вибором,

відкритими відповідями тощо), а також налаштовувати час виконання чи порядок завдань.

- Використання аналітики – збереження та аналіз результатів для оцінки прогресу учнів. Наприклад, викладач може порівняти, як студент справлявся з подібними завданнями раніше, і зрозуміти, чи є покращення.
- Доступність та інтеграція – підтримка веб-доступу, можливість інтеграції з LMS (Learning Management System) та іншими платформами, такими як Google Classroom чи Zoom, що полегшує використання в різних умовах.
- Безпека та захист від списування – механізми запобігання шахрайству, такі як таймери, обмежений доступ до ресурсів, випадкове перемішування питань чи навіть блокування копіювання тексту [23].

Зі швидким розвитком штучного інтелекту (ШІ) системи тестування отримують нові можливості, зокрема автоматичну генерацію питань, персоналізоване навчання та вдосконалені алгоритми аналізу відповідей. Наприклад, ШІ може створювати унікальні завдання на основі попередніх відповідей студента, щоб перевірити саме ті теми, які викликають труднощі. Крім того, сучасні системи дозволяють інтегрувати інтерактивні елементи, такі як симуляції чи практичні завдання, що особливо корисно для технічних спеціальностей, де важливо перевірити не лише теорію, а й уміння застосовувати знання на практиці [23].

Водночас системи тестування стають дедалі більш адаптивними до потреб користувачів. Наприклад, деякі платформи пропонують функції для людей з особливими потребами: аудіоверсії питань для тих, хто має проблеми із зором, або спрощений інтерфейс для молодших школярів. Також у багатьох системах є можливість налаштувати мову інтерфейсу, що робить їх зручними для студентів із різних країн [23].

Ще однією важливою особливістю є можливість командної роботи. Деякі платформи дозволяють створювати групові тести, де студенти можуть співпрацювати, вирішуючи завдання разом, що сприяє розвитку навичок комунікації та спільного пошуку рішень. Наприклад, у корпоративному

навчанні такі тести часто використовують для оцінки командної роботи під час вирішення бізнес-кейсів [23].

Таким чином, сучасні системи тестування знань є ефективним інструментом у сфері освіти та професійної підготовки, що дозволяє оптимізувати навчальний процес та підвищити його якість. Вони постійно розвиваються, враховуючи нові технологічні можливості та потреби користувачів, і стають дедалі універсальнішими, щоб відповідати викликам сучасного світу.

1.2 Переваги та недоліки сучасних технологій та аналіз наявних платформ тестування

Сучасні технології тестування знань значно покращують освітній процес, забезпечуючи автоматизовану, об’єктивну та швидку перевірку знань. Завдяки використанню штучного інтелекту й машинного навчання з’являється можливість створювати адаптивні тести, які враховують рівень підготовки кожного користувача. Такі платформи дозволяють зберігати й аналізувати великі обсяги даних, що сприяє точнішій оцінці динаміки навчання. Проте, поряд з перевагами, існують і певні недоліки: складність оцінювання відкритих чи творчих відповідей, потреба в технічній підтримці, навчанні персоналу та ризик технічних збоїв. Також надмірне використання автоматизації може зменшити роль викладача, що впливає на мотивацію студентів. Отже, ефективне впровадження таких технологій потребує ретельного планування та поєднання автоматизованих рішень із педагогічною участю [3].

Сучасні платформи для тестування знань пропонують широкий спектр функцій, які дозволяють автоматизувати процес оцінювання, забезпечити зручний доступ для користувачів і вдосконалити освітній процес. Розглянемо деякі популярні платформи та їх ключові можливості. Популярні платформи тестування:

1. Google Forms (рисунок 1.1) — це безкоштовний інструмент від Google, призначений для створення простих онлайн-тестів, анкет і опитувань. Він дозволяє додавати запитання з одним або кількома варіантами відповіді, відкриті запитання, шкали оцінювання тощо. Google Forms підтримує автоматичне оцінювання, якщо попередньо вказати правильні відповіді. Всі зібрані відповіді зручно зберігаються в Google Sheets, що дає змогу виконувати їх подальший аналіз. Сервіс також надає можливості для обмеження доступу до форми за часом або за обліковими записами користувачів, що особливо корисно у навчальному середовищі. Завдяки інтеграції з іншими сервісами Google (Docs, Classroom) Google Forms є зручним рішенням для базового тестування [8].

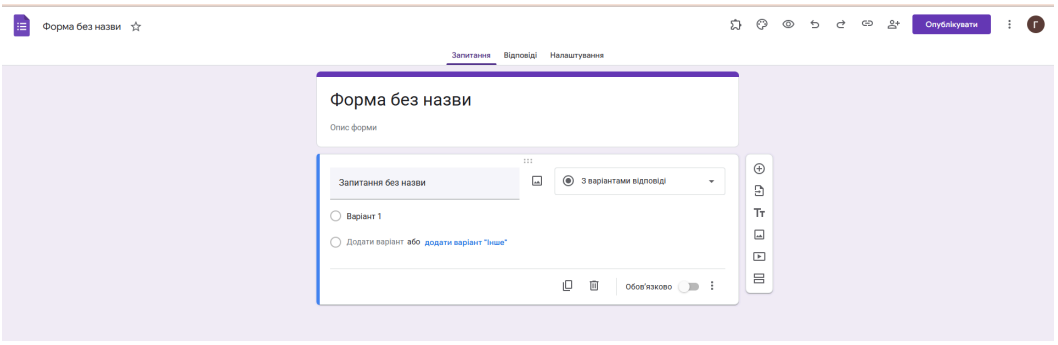

Рисунок 1.1 – Google Forms

2. Moodle (рисунок 1.2, знаходиться на наступній сторінці) — це потужна система управління навчанням (LMS), яка дозволяє створювати повноцінні навчальні курси, включаючи лекції, завдання та тести. Платформа підтримує широкий спектр типів запитань: множинний вибір, встановлення правильної послідовності, числові відповіді, есе та інші. Система дозволяє налаштовувати час проходження тесту, кількість спроб, адаптивне оцінювання, а також автоматично аналізує результати й надає детальну статистику. Moodle підтримує ролі користувачів (адміністратор, викладач, студент) та інтеграцію з різноманітними плагінами, такими як Zoom, SCORM, H5P тощо. Такий функціонал робить Moodle ідеальним вибором для середніх і вищих навчальних закладів, які прагнуть організувати повноцінний освітній процес онлайн [5].

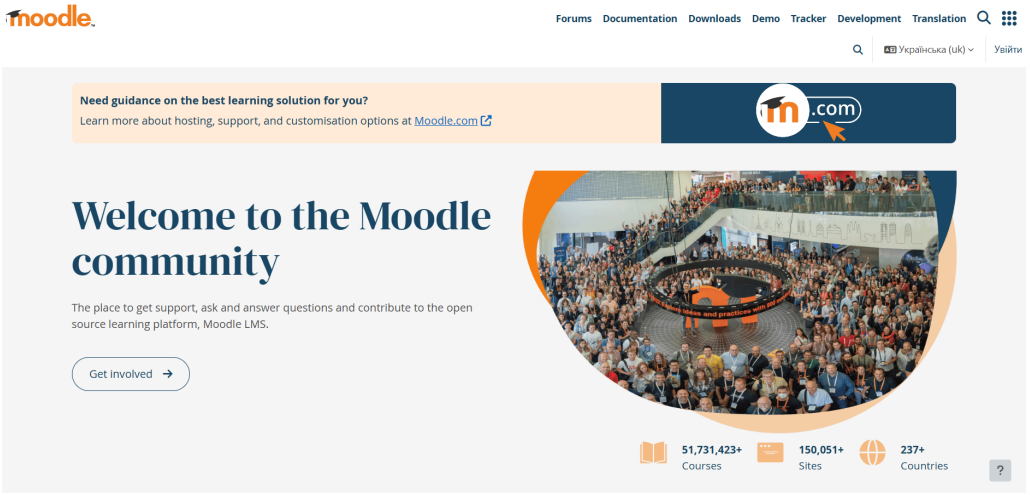

Рисунок 1.2 – Moodle

3. Kahoot! (рисунок 1.3) орієнтований на створення інтерактивних ігрових вікторин, які спрямовані на залучення учнів та підвищення їх мотивації. Платформа дозволяє створювати яскраві запитання з мультимедійним супроводом, які учасники проходять у режимі реального часу. Всі гравці відповідають одночасно, а система миттєво показує результати та формує рейтинг. Такий формат особливо популярний у школах, адже перетворює навчання на захопливу гру. Крім того, Kahoot! підтримує самостійний режим проходження (challenge), що дозволяє використовувати платформу і для домашнього навчання [1].

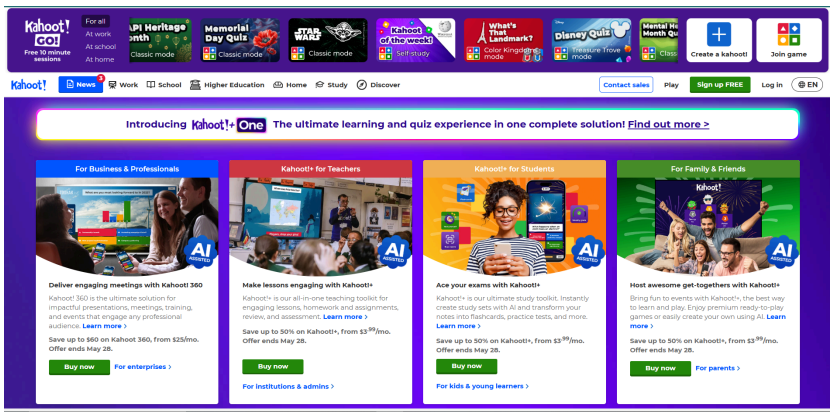

Рисунок 1.3 – Kahoot!

4. Quizizz (рисунок 1.4) подібна до Kahoot!, але надає ширші можливості для проведення тестів як у режимі реального часу, так і асинхронно. Сервіс має яскравий інтерфейс із гейміфікацією, візуальними ефектами та аватарами. Після кожного запитання учень отримує зворотний зв'язок, що допомагає краще засвоїти матеріал. Quizizz також дозволяє переглядати статистику, зокрема правильні й неправильні відповіді, середній час виконання тесту та бал. Користувачі можуть створювати власні запитання або використовувати вже готові шаблони з великого банку. Платформа дає можливість створювати комплексні уроки, поєднуючи навчальний матеріал і тести в одному потоці [7].

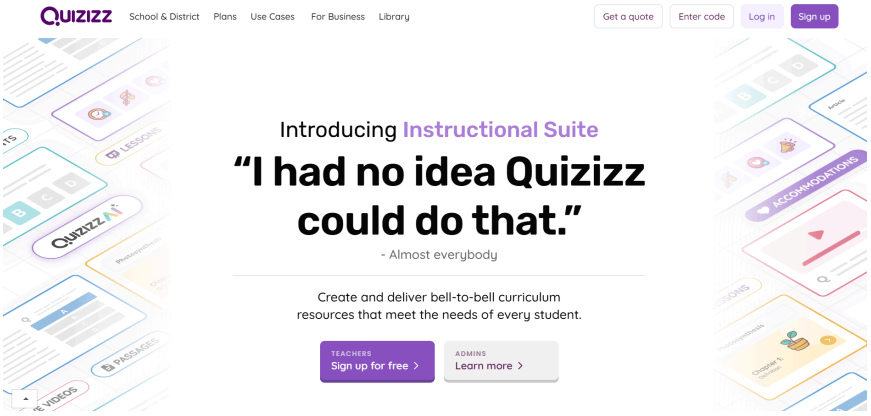

Рисунок 1.4 – Kahoot!

5. Testmoz (рисунок 1.5) — це ще один популярний сервіс, який вирізняється простотою у використанні. Його головна перевага — можливість швидко створити тест без реєстрації. Платформа підтримує запитання з одиничною або множинною відповіддю, а також текстові відповіді. Адміністратор може обмежити кількість спроб, тривалість проходження тесту або зробити тест анонімним. Результати можна експортувати у форматі CSV для подальшого аналізу. Крім того, Testmoz дозволяє випадковим чином ротаціювати порядок запитань та відповідей, що допомагає запобігти списуванню [6].

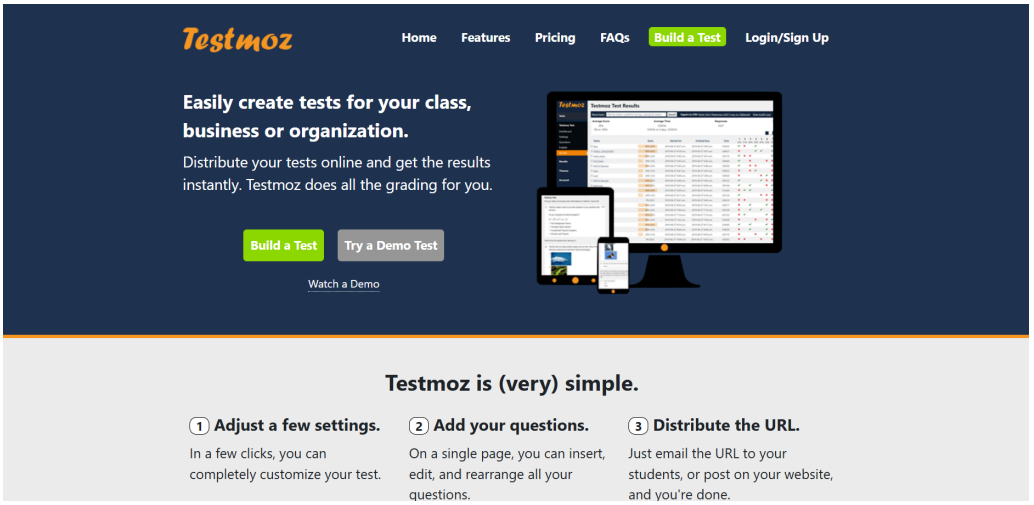

Рисунок 1.5 – Testmoz

Отже, вибір платформи визначається специфічними потребами користувачів: якщо важлива інтерактивність – підійдуть Kahoot! або Quizizz, для комплексного управління навчальними курсами – Moodle, для простоти використання – Google Forms чи Testmoz, а для глибокого аналізу результатів – спеціалізовані платформи з розширеною аналітикою. Використання штучного інтелекту у тестуванні відкриває нові перспективи автоматизації процесу оцінювання та персоналізації навчання.

1.3 Методи автоматизованого тестування знань

Автоматизоване тестування знань є важливим інструментом у сучасній освіті, що дозволяє ефективно оцінювати рівень підготовки студентів, зменшувати навантаження на викладачів та забезпечувати об’єктивність оцінювання. Існує кілька основних методів автоматизованого тестування, які використовуються в навчальних закладах, корпоративному навчанні та сертифікаційних програмах, а їхнє різноманіття дає змогу адаптувати процес до різних потреб і дисциплін [33].

Одним із найпоширеніших методів є тестування з вибором відповіді. Такі тести передбачають питання з кількома варіантами, де потрібно обрати правильний. Цей метод простий у реалізації та дозволяє швидко отримувати результати, що робить його зручним для масових перевірок, таких як вступні іспити чи шкільні контрольні [33].

Інший популярний підхід — тести з відкритими відповідями, де студент самостійно формулює відповідь. Цей метод краще відображає розуміння матеріалу, вимагаючи логічного викладу думок, але потребує додаткових зусиль для аналізу або ручної перевірки. Застосування штучного інтелекту може полегшити цей процес, аналізуючи текст на відповідність критеріям [33].

Завдання на відповідність є ще одним ефективним методом автоматизованого тестування знань. У таких завданнях користувачам потрібно зіставити елементи двох списків, наприклад, терміни з їх визначеннями чи події з датами. Цей метод допомагає оцінити не лише пам’ять, а й логічні зв’язки, що робить його популярним у гуманітарних, природничих та технічних дисциплінах. Наприклад, студенти можуть поєднувати назви рослин із їхніми характеристиками або економічні поняття з прикладами [33].

Комп’ютеризоване імітаційне тестування застосовується для перевірки практичних навичок у спеціалізованих галузях, таких як медицина, програмування чи авіація. Учасникам пропонуються змодельовані ситуації, де вони приймають рішення, а система оцінює їхні дії. Наприклад, у медичній симуляції студент може діагностувати віртуального пацієнта, обираючи аналізи чи лікування, а в програмуванні — виправляти помилки в коді [33].

Ігрові методи тестування, що використовують гейміфікацію, також набирають популярності для підвищення залученості. Це можуть бути інтерактивні вікторини, симуляції чи навчальні ігри. Наприклад, платформи типу Kahoot пропонують командні змагання з таймером, де студенти заробляють очки за правильні відповіді, а в іграх типу CodeCombat учні проходять рівні, вирішуючи кодові задачі [33].

Метод тестування з впорядкуванням (сортуванням) стає усе більш затребуваним, особливо в технічних дисциплінах. Студентам пропонують перелік елементів — наприклад, етапи алгоритму чи історичні події — і просять розташувати їх у правильному порядку. У хімії це може бути послідовність реакцій, а в програмуванні — кроки реалізації циклу, що розвиває логічне мислення [33].

Метод тестування на основі кейсів популярний у бізнес-освіті та юриспруденції. Студентам пропонують реальні ситуації — наприклад, розв'язання конфлікту чи аналіз судового прецеденту — і просять обрати рішення чи обґрунтувати його. Система може оцінювати відповіді за ключовими словами, хоча для складних кейсів потрібна комбінація ШІ та людської перевірки [33].

Застосування різних методів автоматизованого тестування знань дозволяє зробити процес оцінювання більш ефективним, гнучким та адаптивним, що сприяє покращенню якості освіти та підготовки фахівців у різних сферах. У наступних трьох підрозділах буде детальніша інформація про такі методи тестування, як з одним варіантом відповіді, з декількома варіантами відповіді та з відкритою відповіддю.

1.4 Забезпечення академічної доброчесності під час тестування

Системи тестування знань є невід'ємною частиною сучасного освітнього процесу, забезпечуючи ефективний механізм перевірки рівня підготовки студентів, слухачів курсів та працівників у корпоративному секторі [23]. Завдяки автоматизації процесу оцінювання вони дозволяють значно зменшити навантаження на викладачів, підвищити об'єктивність виставлення оцінок та надати користувачам швидкий доступ до результатів [23]. Сьогодні такі системи активно застосовуються не лише у традиційних навчальних закладах (школах, університетах), а й у дистанційному навчанні, сертифікаційних програмах,

підготовці до іспитів та внутрішньо корпоративному навчанні. Вони дозволяють стандартизувати оцінювання та забезпечують єдиний підхід до перевірки знань на всіх рівнях навчального процесу. Наприклад, у великих компаніях такі системи часто використовують для оцінки нових співробітників або перевірки знань після тренінгів, що допомагає швидко визначити, чи готова людина до виконання своїх обов'язків [23].

Основними особливостями систем тестування є:

1. Автоматизоване оцінювання: Система самостійно перевіряє відповіді та формує результати тесту, що виключає суб'єктивність і економить час [23].
2. Гнучкість у налаштуванні тестів: Можливість створювати питання різних типів (з єдиною правильною відповіддю, множинним вибором, відкритими відповідями тощо), а також налаштовувати час виконання чи порядок завдань [23].
3. Використання аналітики: Збереження та аналіз результатів для оцінки прогресу учнів. Наприклад, викладач може порівняти, як студент справлявся з подібними завданнями раніше, і зрозуміти, чи є покращення [23].
4. Доступність та інтеграція: Підтримка веб-доступу, можливість інтеграції з LMS (Learning Management System) та іншими платформами, такими як Google Classroom чи Zoom, що полегшує використання в різних умовах [23].
5. Безпека та захист від списування: Механізми запобігання шахрайству, такі як таймери, обмежений доступ до ресурсів, випадкове перемішування питань чи навіть блокування копіювання тексту [23].

Зі швидким розвитком штучного інтелекту (ШІ) системи тестування отримують нові можливості, зокрема автоматичну генерацію питань, персоналізоване навчання та вдосконалені алгоритми аналізу відповідей. Наприклад, ШІ може створювати унікальні завдання на основі попередніх відповідей студента, щоб перевірити саме ті теми, які викликають труднощі.

Крім того, сучасні системи дозволяють інтегрувати інтерактивні елементи, такі як симуляції чи практичні завдання, що особливо корисно для технічних спеціальностей, де важливо перевірити не лише теорію, а й уміння застосовувати знання на практиці [23].

Водночас системи тестування стають дедалі більш адаптивними до потреб користувачів. Наприклад, деякі платформи пропонують функції для людей з особливими потребами: аудіоверсії питань для тих, хто має проблеми із зором, або спрощений інтерфейс для молодших школярів. Також у багатьох системах є можливість налаштувати мову інтерфейсу, що робить їх зручними для студентів із різних країн [23]. Ще однією важливою особливістю є можливість командної роботи. Деякі платформи дозволяють створювати групові тести, де студенти можуть співпрацювати, вирішуючи завдання разом, що сприяє розвитку навичок комунікації та спільного пошуку рішень. Наприклад, у корпоративному навчанні такі тести часто використовують для оцінки командної роботи під час вирішення бізнес-кейсів [23].

У процесі онлайн-тестування забезпечення академічної доброчесності є ключовим завданням, адже від цього залежить справедливість оцінки знань і довіра до результатів [31]. У сучасному цифровому середовищі, де доступ до інформації став майже необмеженим, учасники можуть шукати способи отримати перевагу через обман, списування, передачу відповідей іншим або використання заборонених ресурсів [31]. Платформа EduTest приділяє значну увагу цьому аспекту, впроваджуючи комплекс технічних і організаційних рішень, спрямованих на підтримку академічної чесності та створення прозорого середовища для тестування.

Платформа включає кілька інноваційних механізмів, які допомагають запобігти нечесним діям:

1. Блокування копіювання вмісту тесту: Щоб уникнути поширення питань чи відповідей поза платформою, EduTest реалізує обмеження на копіювання тексту. Це досягається через відключення подій copy, cut, contextmenu у браузері, а також обмеження функції select за допомогою

Angular та JavaScript. Наприклад, якщо студент спробує виділити текст для копіювання, система видасть попередження, а дія буде заблокована, що ускладнює створення "зливів" тестів у соціальних мережах чи інших платформах [31].

2. Виявлення спроб зробити знімок екрана: Для захисту вмісту тестів платформа відстежує натискання клавіш, таких як PrintScreen, або комбінацій, як-от Alt + PrintScreen, Windows + Shift + S чи використання вбудованих інструментів знімків у сучасних ОС. При виявленні таких дій система може виводити сповіщення з нагадуванням про правила тестування або, у серйозних випадках, блокувати подальше виконання тесту, фіксуючи інцидент у журналі подій [31].
3. Обмеження кількості спроб: Щоб уникнути багаторазового проходження тесту з метою "навчання на помилках", платформа встановлює обмежену кількість спроб для кожного користувача. Ця інформація зберігається в базі даних, а система автоматично блокує доступ після вичерпання лімітів, що мотивує студентів готуватися ретельніше з першого разу [31].
4. Облік та контроль часу: Для кожного тесту задається максимальний час на виконання, після закінчення якого відповіді більше не приймаються. Це не лише дисциплінує студентів, а й унеможливлює використання зовнішніх ресурсів під час тесту. Наприклад, якщо тест розрахований на 30 хвилин, система автоматично завершує сесію після цього терміну, зберігаючи лише надані відповіді [31].

Завдяки цим заходам платформа EduTest створює більш справедливе та прозоре середовище проходження тестів, що стимулює студентів до самостійного виконання завдань і підвищує довіру до результатів тестування. Викладачі можуть бути впевнені в об’єктивності оцінок, а студенти отримують мотивацію краще готуватися, знаючи, що обман виявиться марним. У довгостроковій перспективі це сприяє формуванню відповідального ставлення до навчання, що є важливим для їхнього професійного зростання [31].

Таким чином, сучасні системи тестування знань, такі як EduTest, є ефективним інструментом у сфері освіти та професійної підготовки, що дозволяє оптимізувати навчальний процес та підвищити його якість. Вони постійно розвиваються, враховуючи нові технологічні можливості та потреби користувачів, і стають дедалі універсальнішими, щоб відповідати викликам сучасного світу [23, 31].

1.5 Тенденції розвитку систем тестування в умовах цифрової трансформації

У сучасному світі цифрові технології дедалі більше змінюють освіту, і штучний інтелект (ШІ) відіграє тут ключову роль, особливо в онлайн-навчанні та тестуванні [10]. Одна з головних переваг ШІ — це можливість автоматично створювати тести, аналізуючи навчальні матеріали. Системи, які розпізнають природну мову, здатні виокремити головні теми, логічні зв’язки між ними та генерувати запитання — від простих до складних, які вимагають аналізу, порівняння або навіть критичного підходу. Це значно полегшує життя викладачам, знімаючи з них частину рутинної роботи та даючи більше часу на індивідуальну взаємодію з учнями [10].

До того ж, ШІ здатен аналізувати сам процес проходження тесту: скільки часу студент витрачає на відповіді, чи повертається до певного запитання, чи довго вагається. Це допомагає краще зрозуміти не лише рівень знань, а й стиль мислення — наприклад, чи учень упевнений у своїх відповідях, чи просто вгадує, яка може бути правильною [10]. Штучний інтелект також корисний для боротьби з академічною недоброчесністю. Деякі платформи, включаючи EduTest, мають вбудовані функції спостереження за поведінкою студентів під час тестування: використовують камеру, фіксують підозрілу активність,

виявляють однакові або схожі відповіді. Це особливо актуально в умовах дистанційного навчання, коли контроль зі сторони викладача обмежений [20].

Крім того, системи на базі ШІ можуть допомагати студентам у навчанні навіть поза межами тестів. Наприклад, на основі помилок вони можуть підказувати, які теми варто повторити, або запропонувати додаткові матеріали — відео, тести, приклади. Це вже не просто інструмент перевірки знань, а справжній навчальний помічник [20]. Таким чином, інтеграція ШІ в платформу EduTest дозволяє створювати персоналізоване навчальне середовище, яке адаптується до потреб кожного студента, підвищуючи ефективність навчання та підтримуючи викладачів у їхній роботі.

Цифрова трансформація суттєво вплинула на розвиток систем тестування знань, відкриваючи нові горизонти для освітнього процесу та ставлячи перед розробниками нові виклики [28]. У останні роки ця сфера переживає справжній бум, і тенденції, які формуються, відображають як технологічні інновації, так і зміни в потребах студентів, викладачів та роботодавців. Однією з найпомітніших змін є перехід від традиційних методів оцінювання до гнучких, технологічно насичених рішень, які адаптуються до сучасного ритму життя та глобалізації освіти [28].

Перш за все, гейміфікація стала одним із ключових напрямків у розвитку тестування. Сьогодні багато платформ, таких як EduTest, додають ігрові елементи, такі як бали, рівні, бейджі чи змагання між учасниками, щоб зробити процес перевірки знань більш мотивуючим. Наприклад, у популярних додатках для вивчення мов, таких як Memrise, студенти можуть "прокачувати" свій рівень, проходячи тести, що нагадують квести. Цей підхід особливо ефективний для молодшого покоління, яке звикло до інтерактивних розваг, і допомагає утримувати увагу навіть під час складних тем. Викладачі також відзначають, що гейміфікація зменшує стрес перед тестами, адже замість страху провалу студенти відчувають азарт. Проте цей метод вимагає ретельного балансування, щоб не перетворити навчання на суцільну гру, де знання відходять на другий план [28].

Ще однією важливою тенденцією є інтеграція віртуальної (VR) та доповненої реальності (AR) у системи тестування. У технічних спеціальностях, наприклад, медицині чи інженерії, такі технології дозволяють створювати симуляційні тести, де студенти можуть відпрацьовувати практичні навички. Уявімо, як студент-медик у VR-симуляції проводить віртуальну операцію, а система автоматично оцінює його дії: правильність рухів, швидкість реагування чи точність діагностики. Подібні тести вже тестуються в деяких західних університетах, і хоча обладнання досі дороге, експерти прогнозують, що з часом воно стане доступнішим. У доповненій реальності тести можуть включати інтерактивні елементи, наприклад, накладання 3D-моделей на реальні об’єкти під час екзаменів із архітектури чи дизайну, що дозволяє перевіряти як теоретичні, так і прикладні знання [28]. У платформі EduTest ці технології можуть розширити можливості для створення інтерактивних і практичних тестів.

Ріст популярності мобільних платформ також суттєво впливає на розвиток тестування. Усе більше студентів використовують смартфони для навчання, і розробники адаптують системи під цей тренд. Наприклад, додатки типу Quizizz дозволяють проходити тести будь-де — у транспорті чи вдома, — а результати одразу синхронізуються з навчальною платформою. Це особливо зручно для тих, хто поєднує навчання з роботою чи має обмежений доступ до комп’ютерів. Водночас мобільний формат вимагає створення тестів із короткими питаннями та інтуїтивним інтерфейсом, що змушує дизайнерів шукати нові підходи до подачі матеріалу. Окрім того, мобільні платформи часто інтегрують push-сповіщення, нагадуючи студентам про наближення дедлайнів чи пропонуючи додаткові завдання для практики [25]. Платформа EduTest використовує ці можливості, щоб забезпечити гнучкість і доступність тестування для студентів.

Ще одним цікавим напрямком є використання блокчейну для забезпечення прозорості та безпеки результатів тестування. Ця технологія, відома завдяки криптовалютам, дозволяє створювати незмінні записи про

проходження тестів, що виключає можливість фальсифікації оцінок чи несанкціонованого доступу. У країнах, де освіта пов’язана з високими стейками — наприклад, при отриманні сертифікатів чи дипломів, — блокчейн може стати гарантією довіри. У 2023 році кілька європейських університетів почали експериментувати з такими системами, записуючи результати іспитів у децентралізовані реєстри, доступні як студентам, так і роботодавцям. Проте впровадження блокчейну потребує значних інвестицій у інфраструктуру, що поки що стримує його масове поширення [25]. Для платформи EduTest блокчейн може стати перспективним рішенням для забезпечення надійності сертифікатів і результатів тестів.

Не можна оминути й тренд на інклюзивність, який стає дедалі важливішим. Системи тестування адаптуються до потреб людей із різними можливостями: від текстових версій із підтримкою сурдоперекладу для слабочуючих до аудіоформатів для незрячих. У деяких країнах уже розробляють тести з підтримкою рідкісних мов чи діалектів, що дозволяє охопити ширшу аудиторію. Наприклад, у 2022 році в Індії запустили платформу, яка адаптує тести під 12 місцевих мов, включаючи тамільську та телугу, що допомогло залучити до навчання віддалені громади. Такий підхід не лише сприяє рівному доступу до освіти, а й збагачує культурний контент платформ [35].

Отже, інтеграція ШІ та сучасних технологічних тенденцій у платформу EduTest — це не мода, а логічний крок уперед. ШІ автоматизує створення тестів, аналізує поведінку студентів і забезпечує академічну доброчесність, тоді як гейміфікація, VR/AR, мобільні платформи, блокчейн та інклюзивність роблять тестування більш інтерактивним, безпечним і доступним. Ці технології разом створюють персоналізоване, ефективне і справді сучасне навчальне середовище, де кожен студент отримує підтримку відповідно до власних потреб. У контексті України, де цифрова трансформація освіти набирає обертів, платформа EduTest має потенціал стати інноваційним рішенням, адаптованим до національних особливостей і потреб.

1.6 Формулювання задачі та обґрунтування вибору методів

Розвиток цифрових технологій в освіті зумовлює необхідність удосконалення методів тестування знань, що забезпечують об'єктивність, адаптивність та ефективність оцінювання. Основною задачею даного дослідження є розробка платформи тестування знань, яка використовує сучасні технології, зокрема штучний інтелект, для автоматизації процесу створення тестових завдань, персоналізації навчального процесу та підвищення точності оцінювання.

Для досягнення цієї мети потрібно вирішити такі ключові завдання:

- Проаналізувати сучасні підходи до тестування знань та визначити їхні переваги й недоліки.
- Дослідити можливості застосування штучного інтелекту для автоматизації генерації тестових запитань.
- Розробити архітектуру програмного забезпечення, що дозволить інтегрувати ШІ у процес створення та перевірки тестів.

Вибір методів розв’язання поставленої задачі базується на сучасних тенденціях у сфері освітніх технологій та програмування. Використання алгоритмів обробки природної мови (NLP) дозволяє автоматично генерувати тестові запитання, аналізуючи навчальні матеріали. Методи машинного навчання сприяють адаптації рівня складності тестів відповідно до індивідуальних особливостей користувача.

Таким чином, правильний вибір методів та підходів до реалізації системи дозволить створити ефективний інструмент для тестування знань, що відповідатиме сучасним вимогам освіти та сприятиме підвищенню якості навчального процесу.

Висновки

Сучасні системи тестування знань відіграють ключову роль у сфері освіти та професійної підготовки, забезпечуючи ефективну, об’єктивну та автоматизовану перевірку знань. Вони дозволяють значно зменшити навантаження на викладачів, оптимізувати процес оцінювання та надати студентам персоналізований підхід до навчання.

Розвиток штучного інтелекту відкриває нові можливості для вдосконалення тестових систем. Алгоритми машинного навчання допомагають не лише автоматично генерувати питання на основі навчального матеріалу, а й аналізувати відповіді студентів, визначати їхній рівень знань та адаптувати складність тестів у реальному часі.

Інтеграція тестових платформ з освітніми системами (LMS) забезпечує зручність доступу та можливість детального аналізу результатів навчання. Додаткові механізми безпеки сприяють зниженню рівня списування та гарантують об’єктивність оцінювання.

Таким чином, використання сучасних систем тестування є важливим кроком у розвитку освіти, адже вони сприяють не лише перевірці знань, а й створенню адаптивного та ефективного навчального середовища. У майбутньому вдосконалення таких систем за допомогою штучного інтелекту та аналітики даних дозволить ще більше підвищити якість освіти та рівень підготовки фахівців у різних галузях.

Розділ 2. ПІДХІД ДО РОЗВ’ЯЗАННЯ ТА ІНСТРУМЕНТАЛЬНА БАЗА

2.1 Огляд засобів розробки та платформ для реалізації

Розробка системи автоматизованого тестування знань потребує використання різноманітних технологій, інструментів та платформ, які забезпечують зручність розробки, масштабованість та ефективність роботи. У цьому підрозділі розглянуто ключові засоби розробки, які можуть бути використані для реалізації платформи тестування.

2.1.1 Технології фронтенду платформи EduTest

Фронтенд платформи EduTest розроблено з використанням сучасних технологій, які забезпечують створення динамічного, зручного та адаптивного інтерфейсу користувача. Використання компонентно-орієнтованої архітектури, реактивного програмування та стандартів веб-розробки дозволяє досягти

високої продуктивності, масштабованості та зручності для всіх категорій користувачів.

Angular — сучасний фреймворк із відкритим кодом, створений і підтримуваний компанією Google, призначений для розробки динамічних односторінкових веб-застосунків (SPA – Single Page Applications) [16]. Його ключовими особливостями є:

- Компонентно-орієнтована архітектура: Дозволяє створювати модульні та повторно використовувані компоненти, що полегшує розробку та підтримку складних інтерфейсів.
- Двостороннє зв’язування даних: Забезпечує автоматичну синхронізацію між моделлю даних і відображенням у користувацькому інтерфейсі.
- Вбудований механізм маршрутизації: Дозволяє створювати складні навігаційні структури без перезавантаження сторінки.
- Реактивне програмування: Інтеграція з бібліотекою RxJS для обробки асинхронних подій і потоків даних.
- Модульність і тестування: Angular сприяє створенню структурованого коду, що легко тестується за допомогою таких інструментів, як Jasmine і Karma.

Angular є основою фронтенду платформи EduTest, оскільки забезпечує швидке створення складних і функціональних інтерфейсів, адаптованих до потреб сучасного вебу [16].

TypeScript — мова програмування, розроблена компанією Microsoft як надмножина JavaScript, яка додає підтримку статичної типізації [30]. Завдяки можливості визначення типів даних на етапі розробки, TypeScript дозволяє виявляти помилки ще до виконання коду, що значно підвищує надійність і передбачуваність роботи застосунку. Основні переваги TypeScript:

- Статична типізація: Дозволяє визначати типи змінних, параметрів і функцій, що підвищує надійність коду.

- Об’єктно-орієнтоване програмування: Підтримує класи, інтерфейси, модулі та інші конструкції, що сприяють структуризації великих проєктів.
- Інтеграція з Angular: Оскільки Angular повністю побудований на TypeScript, це забезпечує безшовну інтеграцію та полегшує розробку.

Використання TypeScript у поєднанні з Angular є галузевим стандартом, що забезпечує повну інтеграцію та ефективну розробку для платформи EduTest [30].

HTML5 і CSS3 — це новітні версії основних технологій веб-розробки, що відповідають за структурування контенту (HTML5) та його візуальне оформлення (CSS3) [29, 15]. Основні можливості:

- HTML5: Надає потужні засоби для семантичної розмітки сторінки, мультимедійної інтеграції (відео, аудіо) та побудови складних форм без використання сторонніх скриптів.
- CSS3: Забезпечує розширені можливості для стилізації елементів, включаючи анімацію, переходи, градієнти та адаптивне компонування (через медіазапити).

У поєднанні ці технології дозволяють створювати сучасні, адаптивні, кросбраузерні інтерфейси користувача, що забезпечують високий рівень доступності та зручності взаємодії для платформи EduTest [29, 15].

RxJS (Reactive Extensions for JavaScript) — це бібліотека для реалізації реактивного програмування в середовищі JavaScript [12]. Вона базується на концепції спостережуваних потоків (Observables) і дозволяє ефективно працювати з асинхронними подіями, такими як натискання кнопок, відповіді HTTP-запитів, таймери тощо. Основні переваги RxJS:

- Оператори: Дозволяють трансформувати, фільтрувати та комбінувати потоки даних (наприклад, map, filter, merge).
- Читаємість коду: Спрощує обробку складних асинхронних сценаріїв.
- Інтеграція з Angular: RxJS є фундаментальним інструментом для реалізації сервісів, реактивних форм і роботи з HTTP-запитами.

У рамках платформи EduTest RxJS використовується для управління асинхронними операціями, такими як оновлення результатів тестів у реальному часі або обробка взаємодії користувача з формами [12].

2.1.2 Технології бекенду та баз даних платформи EduTest

Бекенд платформи EduTest розроблено з використанням сучасних серверних технологій та систем керування базами даних, які забезпечують високу продуктивність, масштабованість і безпеку. Ці технології дозволяють обробляти велику кількість запитів, інтегруватися з фронтендом і надавати інтелектуальні можливості для освітнього процесу.

Node.js — це середовище виконання JavaScript, яке працює на сервері й дозволяє створювати високопродуктивні бекенд-застосунки [22]. Воно базується на рушії V8 від Google Chrome, що забезпечує швидке виконання JavaScript-коду. Основні особливості Node.js:

- Неблокуюча модель вводу/виводу: Дозволяє ефективно обробляти велику кількість одночасних запитів, що ідеально підходить для застосунків реального часу, таких як чати, системи онлайн-тестування чи сповіщення в платформі EduTest.
- Екосистема модулів: Завдяки менеджеру пакетів npm, Node.js має доступ до тисяч модулів, які спрощують створення RESTful API, підключення до баз даних і інтеграцію з фронтендом.
- Асинхронне програмування: Використання асинхронних функцій і промісів забезпечує швидку обробку запитів без блокування основного потоку виконання.

У платформі EduTest Node.js використовується для створення масштабованого бекенду, який забезпечує швидку обробку запитів, таких як авторизація користувачів, збереження результатів тестів і генерація звітів [22].

OpenAI API — це потужний інструмент, який надає розробникам доступ до сучасних мовних моделей штучного інтелекту, зокрема GPT (Generative Pre-trained Transformer) [27]. Цей API дозволяє створювати інтелектуальні системи з широким спектром можливостей. Основні характеристики:

- Обробка природної мови: OpenAI API може генерувати тексти, відповідати на запитання, класифікувати інформацію, аналізувати тексти та виконувати інші завдання, що є корисним для створення автоматичних систем оцінки відкритих питань або генерації навчального контенту.
- Гнучке налаштування: API підтримує параметри, такі як температура (для контролю креативності відповідей), довжина відповіді та підказки, що дозволяють адаптувати модель до конкретних потреб.
- Безпека: Використання API-ключів забезпечує захищену автентифікацію та контроль доступу.

У контексті EduTest OpenAI API може використовуватися для автоматизації створення тестових питань, аналізу відповідей студентів або надання інтелектуальних підказок під час проходження тестів [27].

PostgreSQL — це потужна об'єктно-реляційна система керування базами даних (СКБД) з відкритим кодом, яка відома своєю надійністю, гнучкістю та підтримкою складних операцій [26]. Основні особливості PostgreSQL:

- Складні запити та транзакції: Підтримує розширені SQL-запити, транзакції з ACID-сумісністю, що забезпечує цілісність даних навіть у складних сценаріях.
- Гнучкість: Дозволяє створювати власні типи даних, функції, оператори та індекси, що робить її ідеальною для кастомізації під потреби платформи.
- Масштабованість: Підтримує розподілені бази даних і реплікацію, що забезпечує стабільну роботу при високих навантаженнях.

У платформі EduTest PostgreSQL використовується для зберігання структурованих даних, таких як профілі користувачів, питання тестів,

результати та статистика, забезпечуючи надійність і швидкий доступ до даних [26].

Supabase — це платформа з відкритим кодом, яка базується на PostgreSQL і надає зручний інтерфейс для роботи з базами даних [18]. Вона спрощує розробку та інтеграцію бекенду з клієнтськими додатками. Основні характеристики Supabase:

- Інтерфейс до PostgreSQL: Надає зручний доступ до всіх можливостей PostgreSQL через графічний інтерфейс і API, що полегшує управління даними.
- RESTful API: Автоматично генерує API для швидкої інтеграції з фронтендом, що дозволяє легко підключати платформу до систем тестування, таких як EduTest.
- Зберігання мультимедіа: Підтримує зберігання навчальних матеріалів, зображень, відео та інших файлів, що є важливим для створення інтерактивних тестів.
- Масштабованість: Завдяки хмарній архітектурі Supabase забезпечує стабільну роботу навіть при великій кількості користувачів і обсягах даних.

У платформі EduTest Supabase використовується для спрощення управління базою даних, швидкої інтеграції з фронтендом і забезпечення зберігання мультимедійного контенту, такого як зображення чи відео для тестових питань [18].

2.1.3 Інструменти розробки та тестування платформи EduTest

Для розробки, тестування та управління кодом платформи EduTest використовуються сучасні інструменти, які забезпечують ефективність, зручність і високу якість розробки.

WebStorm — це професійне інтегроване середовище розробки (IDE) від компанії JetBrains, яке спеціально адаптоване для роботи з JavaScript, TypeScript, Angular, Node.js та іншими сучасними технологіями [24]. Основні особливості WebStorm:

- Підсвітка синтаксису та автодоповнення: Забезпечує інтелектуальну підтримку коду, що значно прискорює написання та зменшує кількість помилок.
- Інструменти для відлагодження: Вбудовані дебагери дозволяють відстежувати помилки в коді як на стороні клієнта (фронтенд), так і на стороні сервера (бекенд).
- Підтримка версійного контролю: Інтеграція з Git, GitHub та іншими системами контролю версій спрощує управління змінами в проєктах, включаючи створення гілок, злиття та вирішення конфліктів.
- Інтеграція з фреймворками: WebStorm пропонує спеціалізовані інструменти для Angular, Node.js і TypeScript, а також підтримку плагінів для розширення функціоналу [24].

Postman — це провідний інструмент для тестування та розробки REST API, який значно спрощує роботу з HTTP-запитами [17]. Він дозволяє розробникам перевіряти, налагоджувати та документувати API, що є критично важливим для створення надійного бекенду платформи EduTest. Основні можливості Postman:

- Тестування HTTP-запитів: Дозволяє надсилати запити (GET, POST, PUT, DELETE тощо), переглядати відповіді та перевіряти їх коректність.
- Колекції запитів: Дозволяє створювати, зберігати та повторно використовувати набори запитів для швидкого тестування API.
- Підтримка автентифікації: Підтримує різні методи автентифікації, такі як API-ключі та OAuth.
- Інтеграція з CI/CD: Postman легко інтегрується з процесами безперервної інтеграції та доставки, що дозволяє автоматизувати тестування API.

- Документування та співпраця: Надає зручний інтерфейс для створення документації API та обміну колекціями запитів між членами команди.

У платформі EduTest Postman використовується для тестування RESTful API, що забезпечує зв’язок між фронтендом і бекендом, дозволяючи перевіряти коректність роботи API для авторизації, створення тестів і обробки результатів [17].

Git — це розподілена система контролю версій, яка дозволяє відстежувати зміни в коді та координувати роботу кількох розробників. GitHub — це хмарна платформа для розміщення Git-репозиторіїв і спільної розробки [21].

Основні особливості:

- Контроль версій: Git дозволяє створювати гілки, фіксувати зміни (коміти), об’єднувати зміни та вирішувати конфлікти, забезпечуючи гнучке управління кодом.
- Спільна розробка: GitHub надає онлайн-доступ до репозиторіїв, інструменти для код-рев’ю, pull request-ів, управління проєктами та інтеграцію з CI/CD-пайплайнами.
- Прозорість і ефективність: GitHub підтримує автоматизацію розгортання, створення документації та командну роботу, що спрощує координацію між розробниками.

У контексті EduTest Git і GitHub використовуються для управління вихідним кодом платформи, забезпечення версійного контролю та координації роботи команди розробників. Це дозволяє швидко вносити зміни, тестувати їх і розгортати оновлення, зберігаючи цілісність проєкту [21].

2.1.4 Порівняння Angular та React

Чому саме Angular як основний фреймворк для реалізації платформи EduTest? Angular — це сучасний фреймворк від Google, який забезпечує розширені можливості для розробки односторінкових застосунків (SPA). Основні причини вибору Angular для реалізації платформи EduTest [32]:

1. Angular забезпечує поділ проєкту на логічні модулі, що покращує масштабованість і підтримку коду.
2. TypeScript додає строгість типізації, що зменшує кількість помилок і покращує передбачуваність коду.
3. Вбудовані можливості:
    - Двосторонній зв'язок даних (Two-Way Data Binding) для спрощеного управління станом;
    - Інструменти для роботи з формами, включаючи реактивні форми та вбудовану валідацію.
    - Оптимізоване управління HTTP-запитами через HttpClient.
    - Підтримка асинхронних потоків даних через RxJS

Для порівняння з іншими технологіями розглянемо відмінності між Angular та React у таблиці 2.1 [13, 32, 34].

Критерій	**Angular**	**React**
Тип	Повноцінний фреймворк із вбудованими інструментами.	Бібліотека UI, що потребує додаткових інструментів.
Мова	TypeScript (з вбудованою підтримкою).	JavaScript (TypeScript підтримується, але не є обов'язковим).

Двосторонній зв'язок	Вбудована підтримка (Two-Way Data Binding).	Односторонній потік даних (One-Way Data Binding).
Швидкість розробки	Висока завдяки CLI та вбудованим рішенням.	Потребує додаткових налаштувань та бібліотек.
Масштабованість	Підходить для великих корпоративних рішень.	Більш гнучкий для малих і середніх проєктів.
Підтримка	Google, регулярні оновлення.	Meta, акцент на зворотну сумісність.
Обробка асинхронних даних	Використовує RxJS та реактивне програмування.	Потребує сторонніх бібліотек, таких як Redux або Zustand.

Таблиця 2.1 – Порівняння Angular та React для веб-розробки

React є чудовим вибором для простих або кастомізованих UI, але для комплексної системи з інтеграцією багатьох модулів Angular є кращим вибором завдяки своїй повноцінності та підтримці TypeScript. Таким чином, вибір Angular є стратегічним рішенням, яке відповідає потребам платформи EduTest [34].

Сучасні засоби розробки надають широкий вибір технологій для реалізації автоматизованої системи тестування знань. Використання

веб-фреймворків (Angular), баз даних (PostgreSQL) та штучного інтелекту дозволяє створити ефективну, масштабовану платформу для освітніх потреб.

2.2 Вимоги до програмного забезпечення EduTest

Платформа EduTest розроблена з урахуванням сучасних технологічних стандартів, що забезпечують високу ефективність, надійність та зручність використання для всіх учасників освітнього процесу. Вимоги до програмного забезпечення EduTest включають як технічні аспекти, так і функціональні характеристики, необхідні для оптимальної роботи системи.

1. Операційна система та середовище виконання

Для коректної роботи платформи EduTest необхідне програмне середовище, яке підтримує роботу з сучасними веб-технологіями. Основними операційними системами для роботи з EduTest є:

- Windows (версії 10 і вище)
- macOS (версії 10.14 і вище)
- Linux (наприклад, Ubuntu 20.04 LTS або новіші)

2. Програмні компоненти та середовище розробки

Для розробки та підтримки EduTest використовуються сучасні веб-технології:

- Frontend: Angular (версія 12 і вище), HTML5, CSS3, TypeScript
- Backend: Node.js, Express.js або інші сучасні серверні технології
- База даних: Реляційні бази даних (наприклад, PostgreSQL або MySQL)
- API: RESTful API для взаємодії між фронтендом і бекендом

3. Функціональні вимоги

Платформа EduTest повинна забезпечувати наступні функціональні можливості:

- Реєстрація та авторизація користувачів: можливість реєстрації студентів, викладачів із різними правами доступу.
- Створення і проведення тестів: інтерфейс для викладачів, що дозволяє створювати різноманітні типи тестів (множинний вибір, запитання з відкритою відповіддю тощо).
- Аналіз результатів тестів: автоматична оцінка тестів з можливістю перегляду результатів.
- Інтерфейс для користувачів: інтуїтивно зрозумілий інтерфейс для проходження тестів, зручний для студентів.

4. Вимоги до апаратного забезпечення

Для забезпечення стабільної роботи платформи, сервери повинні відповідати таким вимогам:

- Процесор: Мульти-ядерний процесор (мінімум 4 ядра).
- Оперативна пам'ять: Мінімум 8 ГБ оперативної пам'яті для серверного середовища.
- Місце на диску: Залежно від обсягу даних, рекомендовано мати не менше 100 ГБ вільного місця на диску для бази даних.

Висновки

У розділі 2 були розглянуті основні методи та інструменти, що використовуються для автоматизованого тестування знань, а також вимоги до програмного забезпечення, необхідного для реалізації платформи EduTest.

1. Методи тестування: було детально проаналізовано методи тестування з одним варіантом відповіді та з кількома варіантами, що є основними інструментами для оцінки знань студентів. Кожен з цих методів має свої переваги, що дозволяють досягти високої точності в перевірці знань на різних етапах навчального процесу.
2. Засоби розробки та платформи для реалізації: огляд сучасних засобів розробки, таких як фреймворки, платформи та інструменти для створення тестових платформ, показав важливість вибору відповідних технологій для успішної реалізації проекту, який має бути гнучким і масштабованим.
3. Штучний інтелект: Використання технологій штучного інтелекту в процесі автоматичної генерації тестових завдань є важливим кроком до підвищення ефективності тестування. Завдяки можливостям аналізу навчальних матеріалів та створення адаптивних тестів, що враховують індивідуальні особливості учнів, система стає більш персоналізованою та ефективною.
4. Вимоги до програмного забезпечення: Визначено важливі технічні та функціональні вимоги до програмного забезпечення для ефективної роботи платформи EduTest, зокрема підтримка різних операційних систем і браузерів, а також забезпечення високої безпеки даних та зручного інтерфейсу для користувачів.

Загалом, реалізація платформи EduTest базується на сучасних технологіях і методах, що дозволяють створити високоякісну та адаптивну систему тестування. Її використання дозволяє значно підвищити ефективність навчання та оцінки знань студентів, забезпечуючи при цьому високий рівень безпеки та зручності для користувачів.

Розділ 3. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПЛАТФОРМИ EDUTEST

3.1 Загальна структура та архітектура системи

Архітектура EduTest є багаторівневою, модульною та орієнтованою на розподіл відповідальностей між компонентами. Система поділена на два головних рівні — бекенд і фронтенд, що забезпечує масштабованість, зручність супроводу та подальшого розвитку платформи. Структуру файлової системи можна переглянути у Додатку А.

Бекенд розташований у папці backend, реалізований із чітким дотриманням принципів модульності. Складається з таких основних підсистем:

- config – зберігає конфігураційні файли для підключення до бази даних, налаштування портів, середовищ розгортання тощо.
- controllers – обробники HTTP-запитів, що виступають проміжною ланкою між клієнтом і бізнес-логікою застосунку.

- models – визначення структури бази даних, реалізовані через реляційні моделі без використання ORM.
- repo (repository) – шар доступу до даних, що дозволяє чітко розділити бізнес-логіку та логіку зберігання даних.
- routes – налаштування маршрутизації, яка направляє запити до відповідних контролерів.

Фронтенд розташований у папці pages, написаний з використанням Angular. Основні компоненти логічно згруповані за призначенням:

- entrance, registration, main-page – реалізація сторінок входу, реєстрації користувача та головної сторінки системи.
- student, teacher – окремі модулі для ролей користувачів: студента та викладача.
- services, models – забезпечують обробку та зберігання даних, а також взаємодію з API.
- Глобальні стилі, спільні для різних компонентів, зберігаються в папці styles.

Для кращого розуміння роботи системи в Додатку Б представлені діаграми діяльності (див. Рисунок Б.1 і Рисунок Б.2), що демонструють процеси взаємодії викладача та студента з системою.

Проєкт структуровано на кілька ключових компонентів (рисунок Б.1), кожен із яких виконує специфічні функції для забезпечення стабільної роботи системи. Каталог backend містить серверну логіку, environments включає конфігурації для різних середовищ, models зберігає визначення моделей даних, pages об'єднує файли сторінок додатку, services реалізує бізнес-логіку сервісів, а styles містить стилізаційні файли. Окрім цього, у корені знаходяться статичні ресурси в каталозі assets. Така архітектура забезпечує модульність, масштабованість та легкість підтримки проєкту, дозволяючи ефективно розподіляти відповідальність між компонентами та сервісами. У наступних підрозділах буде ретельний опис кожного з компонентів.

3.2 Опис структури серверної частини (backend)

Серверна частина проєкту, розташована в каталозі backend, включає кілька ключових підкаталогів, які забезпечують функціональність серверної логіки. Каталог config містить конфігураційні файли, необхідні для налаштування бази даних. controllers реалізує обробку HTTP-запитів та логіку контролерів. На рисунку 3.1, що знаходиться на наступній сторінці, показана структура контролера.

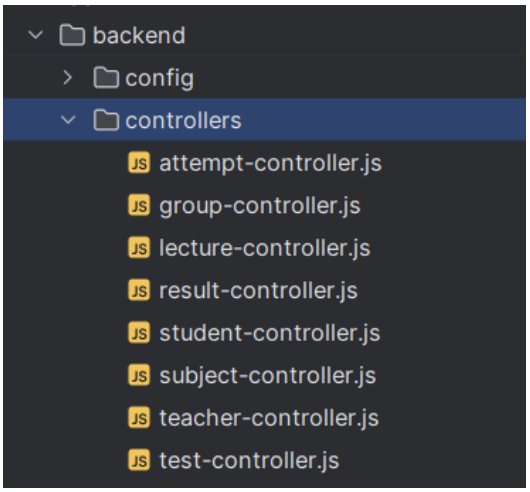

Рисунок 3.1 – Контролери серверної частини

```
1 usage  IraHrechyn
const getGroupById = async (req, res) : Promise<...>  => {
  try {
    const group : Group  = await groupRepository.getGroupById(req.params.id);
    if (!group) {
      return res.status(404).json({ message: 'Group not found.' });
    }
    res.json(group);
  } catch (error) {
    res.status(500).json({ message: 'Failed to fetch group.' });
  }
};
```

Рисунок 3.2 – Приклад реалізації методу getGroupById у контролері

Код на рисунку 3.2 демонструє реалізацію асинхронного методу getGroupById, який використовується в контролері для отримання групи за її ідентифікатором. Метод викликає репозиторій groupRepository.getGroupById і повертає результат у відповіді. Якщо група не знайдена, повертається статус 404 із повідомленням "Group not found". У разі виникнення помилки обробка здійснюється в блоці catch, де повертається статус 500 із повідомленням "Failed to fetch group". Це приклад типової обробки запитів у серверній логіці.

models зберігає визначення даних, що використовуються в системі. Структуру файлів цього каталогу можна побачити нижче на рисунку 3.3

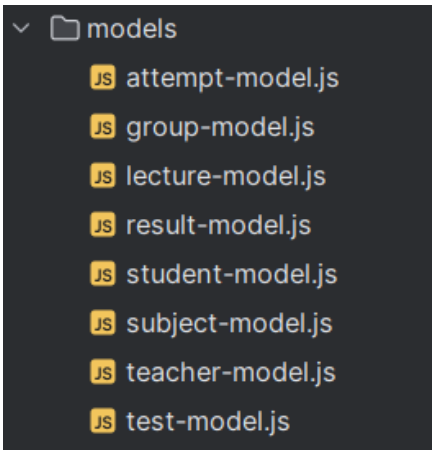

Рисунок 3.3 – Структура файлів моделей даних у каталозі models

На рисунку 3.4 зображено код визначення моделі Teacher, яка успадковується від класу Model. Модель містить атрибути id (ціле число, унікальний первинний ключ), name (рядок, обов’язкове поле), email (рядок, унікальне поле, обов’язкове) та password (рядок, обов’язкове поле). Також вказані опції, такі як назва моделі (Teacher), назва таблиці (teachers) та ввімкнені мітки часу (timestamps: true).

```
Teacher.init(
  attributes: {
    id: {
      type: DataTypes.INTEGER,
      unique: true,
      primaryKey: true
    },
    name: {
      type: DataTypes.STRING,
      allowNull: false,
```

Рисунок 3.4 – Приклад моделі Teacher у файлі teacher-model.js

Каталог repo відповідає за взаємодію з репозиторіями даних (рисунок 3.5), тоді як routes визначає маршрути для обробки запитів, забезпечуючи структуру API (рисунок 3.7).

На рисунку 3.6 показаний код репозиторію groupRepository, який містить асинхронні методи для роботи з групами. Метод getAllGroups повертає всі групи за допомогою Group.findAll(). getGroupById отримує групу за ідентифікатором через Group.findByPk(). addGroup створює нову групу з переданими даними за допомогою Group.create(). updateGroup оновлює групу за ідентифікатором з новими даними за допомогою Group.update(), a deleteGroup видаляє групу за ідентифікатором через Group.destroy().

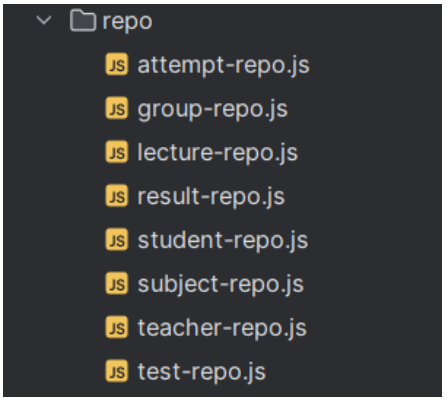

Рисунок 3.5 – Структура файлів репозиторіїв у каталозі repo

```
5+ usages  IraHrechyn
export const groupRepository :{…}  = {
  getAllGroups: async () : Promise<…>  => {
    return await Group.findAll();
  },

  getGroupById: async (id) : Promise<Group>  => {
    return await Group.findByPk(id);
  },

  addGroup: async (groupData) : Promise<Group>  => {
    return await Group.create(groupData);
  },

  updateGroup: async (id, groupData) : Promise<…>  => {
    const [updated : number ] = await Group.update(groupData,  options: { where: { id } });
    return updated ? await Group.findByPk(id) : null;
  },

  deleteGroup: async (id) : Promise<number>  => {
    return await Group.destroy( options: { where: { id } });
  },
```

Рисунок 3.6 – Приклад репозиторію groupRepository у файлі group-repo.js

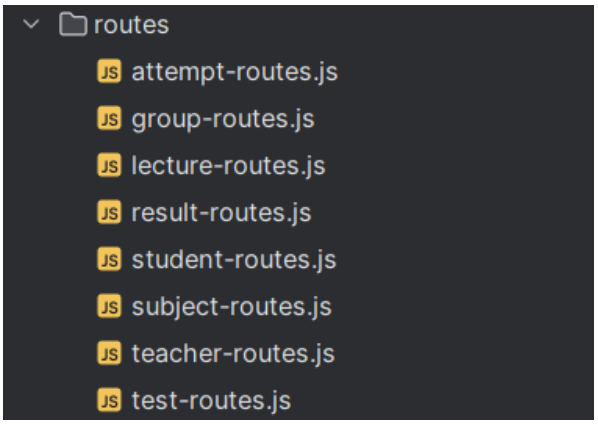

Рисунок 3.7 – Структура файлів маршрутів у каталозі routes

```
import express from 'express';
import teacherController from '../controllers/teacher-controller.js';
const router : Router  = express.Router();

router.post( path: '/', teacherController.addTeacher);

router.get( path: '/', teacherController.getAllTeachers);

router.get( path: '/:id', teacherController.getTeacherById);

router.put( path: '/:id', teacherController.updateTeacher);

router.delete( path: '/:id', teacherController.deleteTeacher);

2 usages  IraHrechyn
export default router;
```

Рисунок 3.8 – Приклад маршрутів у файлі teacher-routes.js

На рисунку 3.8 показаний код маршрутизації для роботи з викладачами в файлі з каталогу routes. Використовується express.Router() для створення маршрутів. POST на / викликає teacherController.addTeacher для додавання викладача. GET на / викликає teacherController.getAllTeachers для отримання всіх викладачів, а GET на /:id — teacherController.getTeacherById для отримання викладача за ідентифікатором. PUT на /:id викликає teacherController.updateTeacher для оновлення даних викладача, а DELETE на /:id — teacherController.deleteTeacher для видалення викладача.

3.3 Опис структури сторінок користувацького інтерфейсу (pages)

Каталог pages містить підкаталоги для окремих сторінок додатку: entrance відповідає за вхідну сторінку, main-page — за головну сторінку, registration — за сторінку реєстрації, student — за функціонал студентського інтерфейсу, а teacher — за інтерфейс викладача.

А зараз детальніше саме про student і teacher. Тут знаходиться, можна сказати, основний функціонал цієї платформи, адже в цих каталогах якраз прописана логіка для створення груп, предметів, генерування тестів і їх проходження.

3.3.1 Аналіз архітектури та функціоналу інтерфейсу викладача

На рисунку 3.9 видна файлова структура, яка міститься у каталозі teacher.

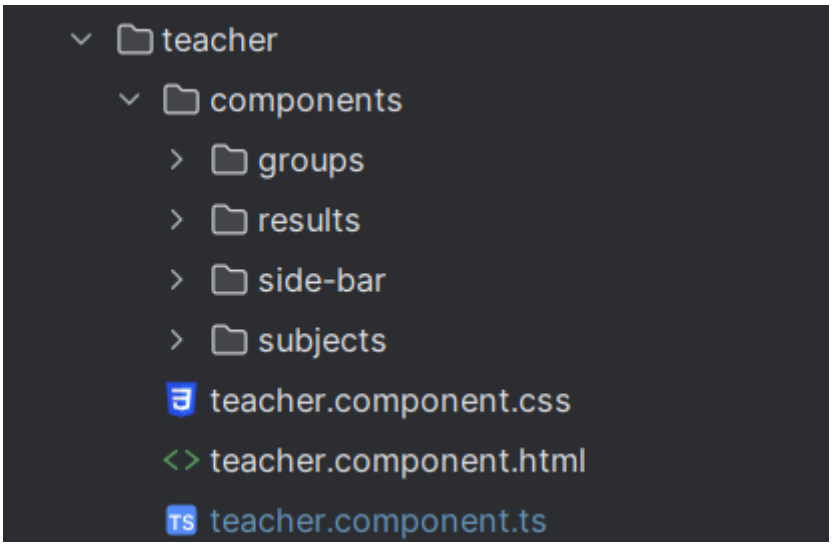

Рисунок 3.9 – Структура файлів сторінок у каталозі teacher

Вміст каталогу groups у структурі проєкту: confirm-dialog, creating-form та group-details. Компонент confirm-dialog призначений для відображення діалогового вікна, яке дозволяє користувачу підтвердити або скасувати видалення групи. Він включає повідомлення "Ви дійсно хочете видалити групу?" та дві кнопки: "Скасувати" (з подією cancel) для відмови від дії та "Так" (з подією confirm) для підтвердження видалення. Компонент creating-form

призначений для створення нової групи, дозволяючи користувачу вводити назву групи через текстове поле groupName. Він також включає функцію пошуку студентів за ім’ям або email через поле studentSearchQuery, з автозаповненням (filteredStudents) та кнопкою додавання (+). Вибрані студенти відображаються в секції selectedStudents із можливістю видалення (-). Компонент group-details призначений для відображення та редагування деталей групи. Він показує назву групи та список студентів, дозволяючи редагувати назву, додавати або видаляти студентів через пошук, а також видаляти групу з підтвердженням через confirm-dialog. Кнопки "Редагувати групу", "Видалити групу" та "Готово" забезпечують управління групою.

У results відображаються результатів тестів студентів. Він показує список предметів, тести для кожного предмета та деталізовані результати за групами, включаючи бали та відкриті відповіді для кожної спроби. Користувач може розгорнути або згорнути тести та результати, а також переглядати інформацію, якщо вона доступна, або повідомлення про відсутність даних.

Компонент side-bar призначений для навігації в інтерфейсі викладача. Він відображає ім’я користувача та іконку профілю, а також містить вкладки для переходу до розділів "Предмети", "Групи" та "Результати", з відповідними іконками та підсвіткою активної вкладки.

Каталог subjects містить підкаталог components, який об'єднує компоненти, пов'язані з функціоналом предметів. У ньому присутні:

- creating-form — призначений для створення нового предмета в системі. Він дозволяє ввести назву предмета (subjectName), підтвердити її, а потім за бажанням додати групи до предмета через вибір зі списку (groups). Вибрані групи (selectedGroups) можна видаляти, а після завершення вибору натиснути "Зберегти" для збереження предмета з обраними групами.
- edit-subject-form — призначений для редагування назви предмета (subjectName) та управління групами, пов’язаними з ним. Він дозволяє

додавати або видаляти групи (availableGroups) за допомогою кнопок "✔" і "✖" із перемиканням стану через toggleGroup. Користувач може скасувати зміни (cancelEdit) або зберегти їх (saveChanges).

- subject-page — призначений для відображення сторінки конкретного предмета. Він включає заголовок із кнопкою виходу на головну сторінку, кнопки для створення нового тесту або генерації тесту, а також список наявних тестів із можливістю редагування або видалення.
- test-creating — містить компоненти для створення тестів. У ньому є підкаталоги first-step (перший крок створення тесту) та second-step (другий крок створення тесту).
- test-generation — містить компоненти для генерації тестів. У ньому є підкаталог components із вкладеними папками second-step, third-step та upload-files, які відповідають за другий і третій кроки генерації тесту, а також завантаження файлів.

3.3.2 Аналіз архітектури та функціоналу інтерфейсу студента

На рисунку 3.10 видна файлова структура, яка міститься у каталозі student.

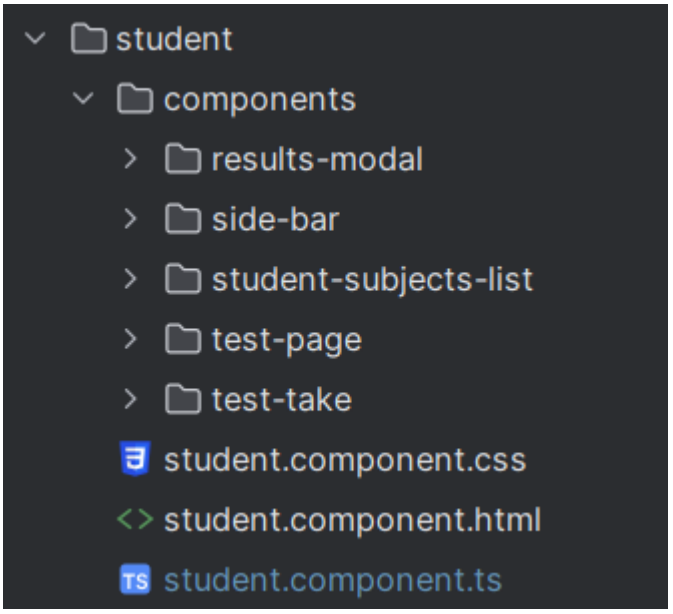

Рисунок 3.10 – Структура файлів сторінок у каталозі student

Компонент results-modal призначений для відображення результатів тесту у модальному вікні. Він показує кількість правильних відповідей із загальної кількості запитань та містить кнопку "Закрити" для завершення перегляду. Компонент side-bar призначений для відображення бічної панелі в інтерфейсі студента. Він показує ім’я студента з іконкою користувача та список груп, до яких належить студент, із назвою кожної групи та ім’ям викладача. З допомогою student-subjects-list відображаються списки предметів студента. Кожен предмет представлений у вигляді картки з унікальним кольором фону, а клік на назву викликає подію для подальшої взаємодії. Компонент test-page призначений для відображення сторінки тестів студента. Він показує назву предмета та список доступних тестів з їхніми назвами і статусами, а також кількість залишених спроб. Клік на тест активує подальшу взаємодію, а декоративні елементи додають візуальний стиль. У test-take студент може проходити обраний тест. Він відображає назву тесту, таймер, навігацію між питаннями з позначками відповідей, варіанти відповідей (один, кілька або відкритий тип), а також кнопку "Завершити тест" для підрахунку результатів, які відображаються у results-modal.

3.3.3 Опис сервісів та моделей даних

Сервіси в каталозі services (рисунок 3.11) реалізують бізнес-логіку та забезпечують взаємодію між компонентами системи. ai-test-generator.service.ts генерує тести автоматично, attempt.service.ts обробляє спроби проходження тестів, email.service.ts відповідає за відправку електронних листів, group.service.ts управляє групами, lecture.service.ts працює з лекціями, result.service.ts обробляє результати тестів, student.service.ts управляє даними студентів, subject.service.ts — предметами, teacher.service.ts — викладачами, а test.service.ts — тестами.

Моделі в каталозі models (рисунок 3.12) визначають структури даних для різних ентіті системи. attempt-data.ts описує дані спроб тестів, constants.ts містить константи, group-data.model.ts — структуру груп, lecture-data.ts —

лекцій, result-data.model.ts — результатів тестів, student-data.model.ts — студентів, subject-data.model.ts — предметів, teacher-data.model.ts — викладачів, а test-data.model.ts — тестів. Ці файли забезпечують уніфіковане представлення даних для роботи з базою даних та логікою додатку.

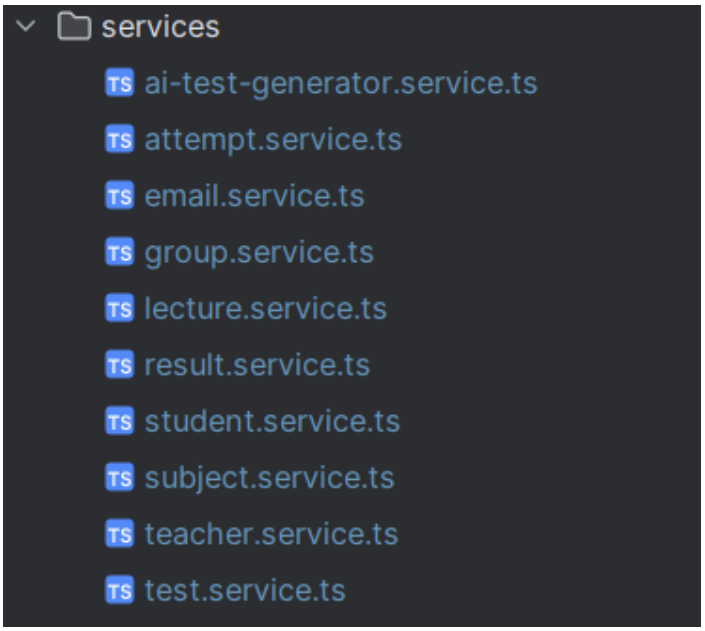

Рисунок 3.11 – Файли сервісів, які містяться у каталозі services

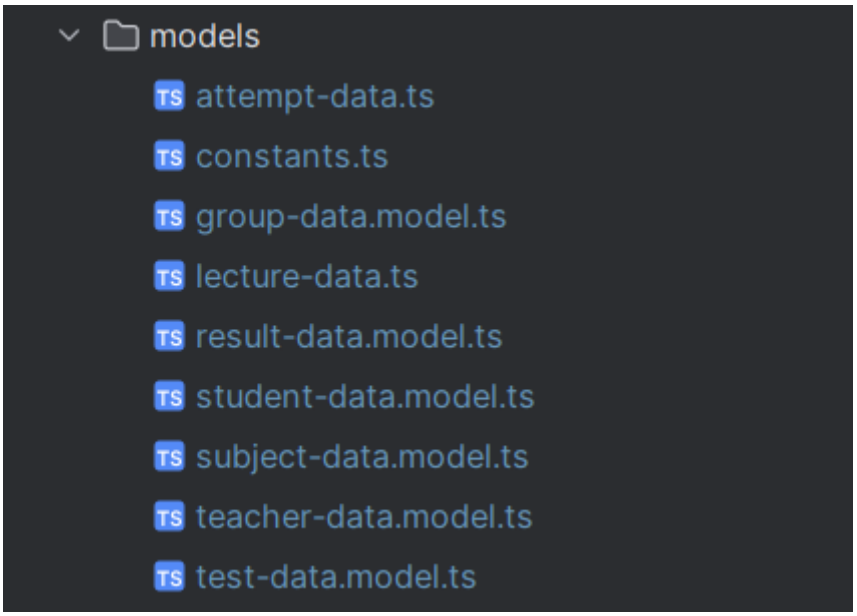

Рисунок 3.12 – Структура файлів сторінок у каталозі models

3.3.4 Сервіс генерації тестових завдань на основі лекцій

Для автоматичного створення тестів у системі EduTest реалізовано спеціальний сервіс AiTestGeneratorService, який інтегрує можливості OpenAI через API для генерації завдань на основі тексту лекцій. Цей сервіс забезпечує створення трьох типів запитань: з однією правильною відповіддю (single), з кількома правильними відповідями (multiple) та з відкритою відповіддю (open).

Принцип роботи полягає в наступному:

- Сервіс отримує лекційний матеріал за допомогою LectureService;
- Формується спеціальний prompt, який надсилається до моделі GPT-4o;
- Модель генерує JSON-масив тестових запитань, що відповідає заданим критеріям;
- Відповідь обробляється, і результат повертається у вигляді масиву об'єктів Question.

Метод, який відповідає за генерацію - generateTest(). Це головний метод сервісу, який:

- Отримує лекційний матеріал через lectureService.getLectureById();
- Формує prompt (інструкцію для моделі), в якому чітко задається структура запитань, формат JSON та кількість кожного типу завдань (рисунок В.1);
- Надсилає цей prompt до моделі GPT-4o;
- Отримує відповідь і намагається її перетворити у масив об'єктів типу Question;

3.4 Принцип роботи програми

На рисунку 3.13 показана головна сторінка платформи "EduTest". На ній розташовано дві кнопки: "Увійти" для входу в систему та "Зареєструватись" для створення нового облікового запису.

Рисунок 3.13 – Головна сторінка

Після натискання кнопок "Увійти" або "Зареєструватись" з’являється сторінка (рисунок 3.14), де потрібно вказати Вашу роль - "Викладач" або "Студент". Поки ви не виберете роль, кнопка “Далі” не стане клікабельною.

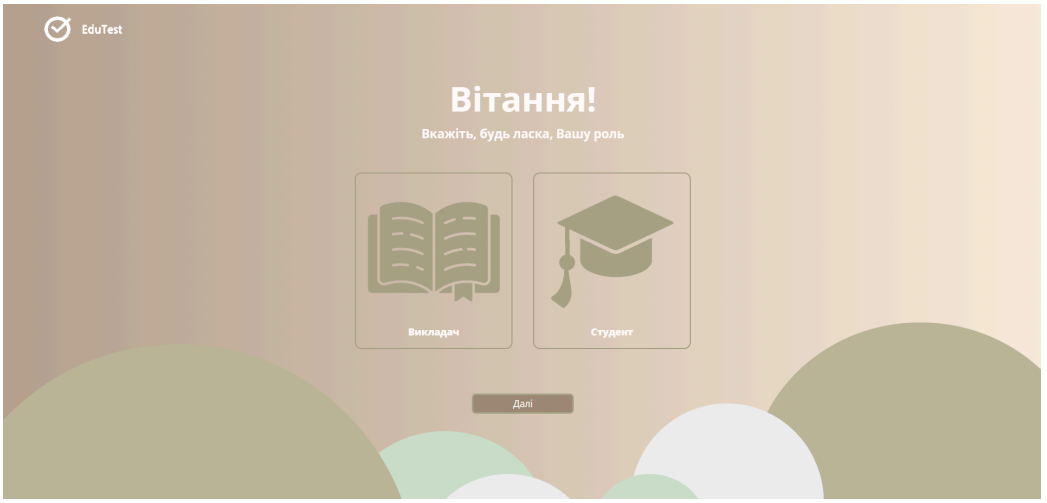

Рисунок 3.14 – Сторінка вибору ролі користувача в "EduTest"

Рисунок 3.15 – Форми для реєстрації та входу на платформу

Після вибору ролі потрібно ввести свої дані: якщо користувач реєструється, то треба вказати ім’я та прізвище, електронну пошту і створений пароль; якщо входить, то тільки електронну пошту і пароль. Форми для реєстрації та входу показані на рисунку 3.15, що знаходиться вище.

Якщо користувач зареєструвався/увійшов як викладач, то його перекидає на сторінку викладача (рисунок 3.16). Сторінка викладача включає ліву панель із профільною інформацією та меню, а також основну робочу область. У лівій панелі відображається ім’я викладача з іконкою профілю, а також пункти меню: "Предмети", "Групи" та "Результати". У верхній частині сторінки є поле пошуку ("Пошук...") та кнопка "Вийти" праворуч. У центрі розміщено картку з написом "Натисніть, щоб додати предмет", що пропонує викладачу створити новий предмет.

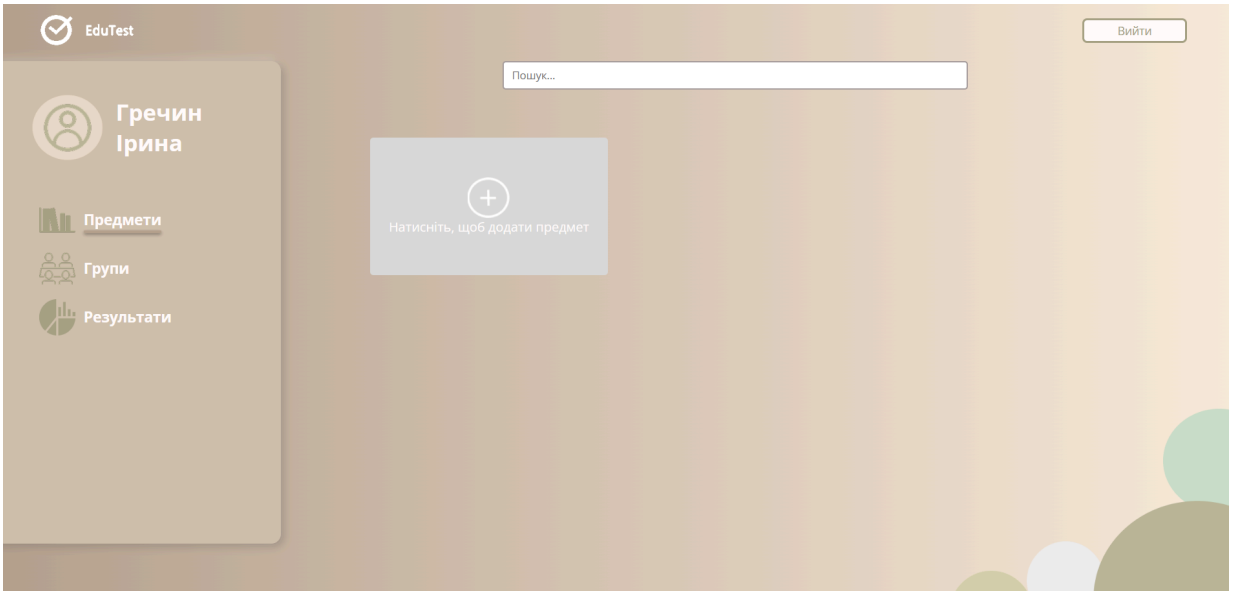

Рисунок 3.16 – Сторінка викладача

Перш ніж створити предмет, варто створити групу (хоча можна і створити спочатку предмет, а потім додати групи). Для того, щоб створити групи потрібно в меню вибрати пункт "Групи". Після цього з’явиться картка з написом “Натисніть, щоб додати групу”, на неї потрібно натиснути.

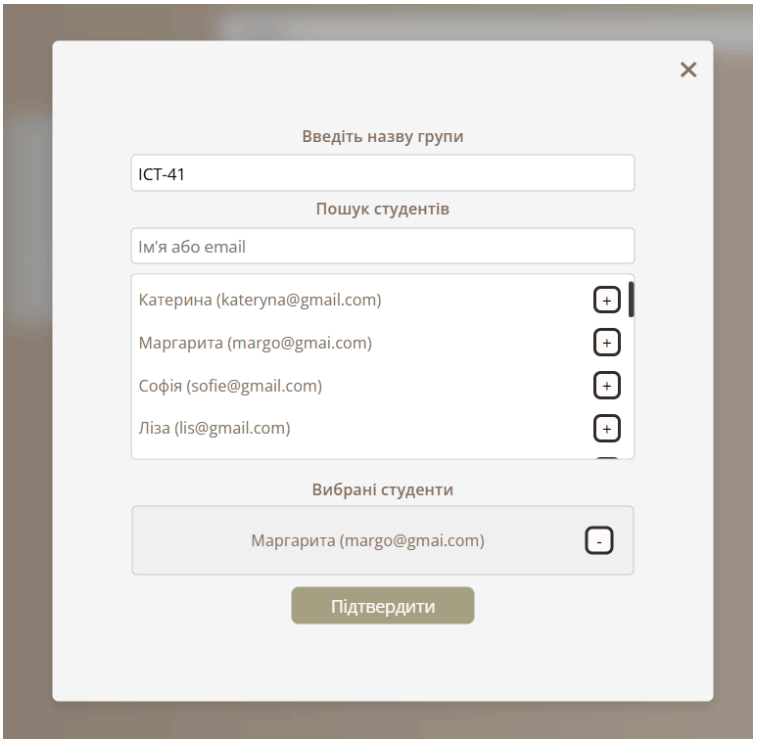

Рисунок 3.17 – Модальне вікно для створення груп

З’являється модальне вікно (рисунок 3.17), де потрібно вказати назву групи і вибрати студентів. У разі помилкового вибору, студента можна забрати зі списку, натиснувши на “-” біля його імені та прізвища. Після підтвердження на сторінці з’являється картка з цією групою (рисунок 3.18).

Рисунок 3.18 – Картка з групою ICT-41

При натисненні на картку можна побачити всіх студентів, доданих до групи (рисунок 3.19). Також групу можна видалити та редагувати (змінити назву, додати/видалити студентів). При видаленні з’являється модальне вікно, яке перепитує викладача, чи точно він бажає видалити цю групу (рисунок В.1).

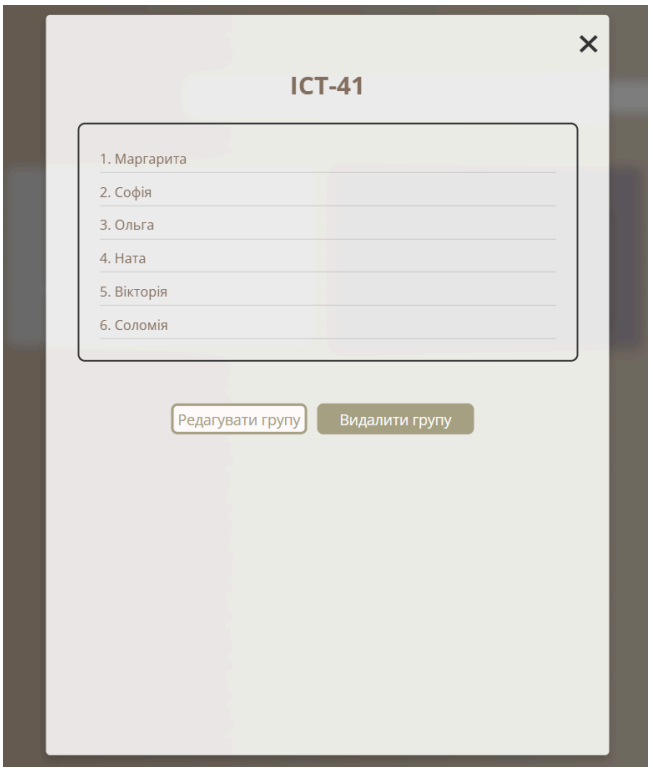

Рисунок 3.19 – Інформація про групу

Після того, як група була створена, можна переходити до створення предмету. Потрібна натиснути на картку з написом "Натисніть, щоб додати предмет". Далі з’являється модальне вікно, де можна ввести назву предмета додати (за бажанням) групу (показано на рисунку 3.20).

Рисунок 3.20 – Додавання групи до предмету

Далі картка зі створеним предметом відобразиться на сторінці (рисунок 3.21).

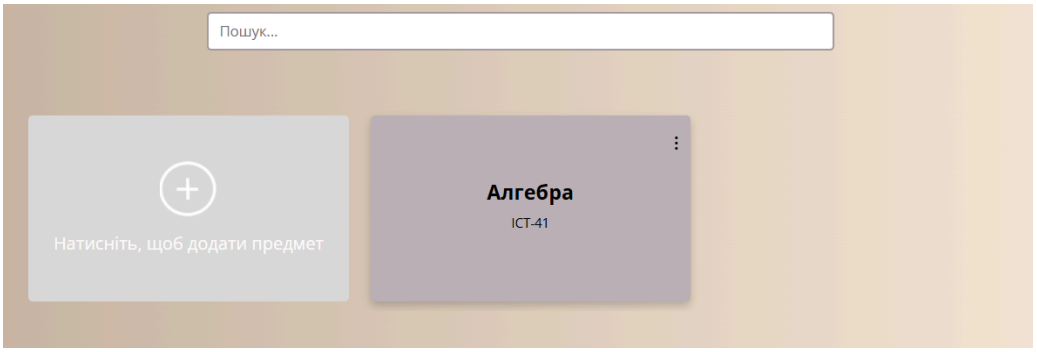

Рисунок 3.21 – Картка з предметом “Алгебра”

Предмет можна редагувати — змінювати його назву, додавати або видаляти пов’язані групи — або повністю видалити (рисунок В.2). При натисканні на предмет відкривається його сторінка, на якій відображаються всі тести, пов’язані з цим предметом. Також на цій сторінці розміщено дві кнопки: «Додати новий тест» та «Згенерувати тест». Процес створення тесту вручну та

автоматичної генерації складається з кількох кроків, при цьому перший крок є спільним для обох варіантів (рисунок 3.22).

Рисунок 3.22 – Крок 1. Налаштування тесту

На цьому кроці потрібно виконати наступне:

- вказати назву тесту у полі "Вкажіть назву тесту";
- встановити тривалість тесту;
- вказати кількість спроб;
- вказати дату і час відкриття та закриття тесту;
- натиснути кнопку "Далі" для переходу до наступного етапу або "Назад" для повернення на сторінку предмету.

Якщо тест створюється вручну, то другий крок, який є завершальним, показано на рисунку 3.23. На цьому етапі користувач може створювати завдання, які будуть включені до тесту. Підтримуються три типи завдань: з однією правильною відповіддю, з кількома правильними відповідями та з відкритою відповіддю. Кількість завдань не обмежена. Після додавання усіх необхідних питань тест можна зберегти або повернутися до попереднього кроку для внесення змін.

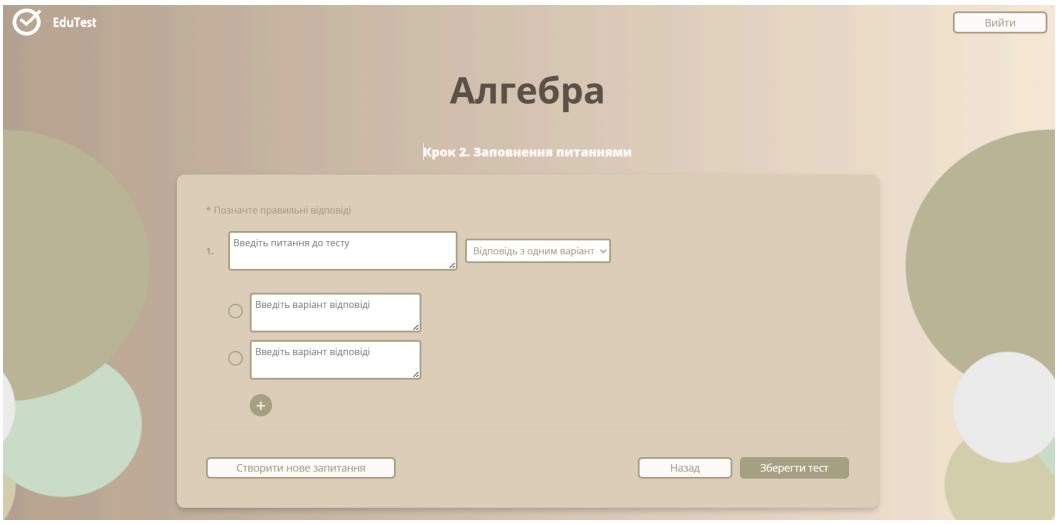

Рисунок 3.23 – Крок 2. Заповнення питаннями

Другий етап створення тесту шляхом автоматичного генерування показано на рисунку 3.24. На цьому кроці користувач має змогу виконати такі дії:

- вказати кількість запитань з однією правильною відповіддю у відповідному полі;
- вказати кількість запитань з кількома правильними відповідями;
- вказати кількість запитань з відкритою відповіддю;
- натиснути кнопку «Далі» для переходу до наступного етапу або кнопку «Назад» для повернення до попереднього кроку.

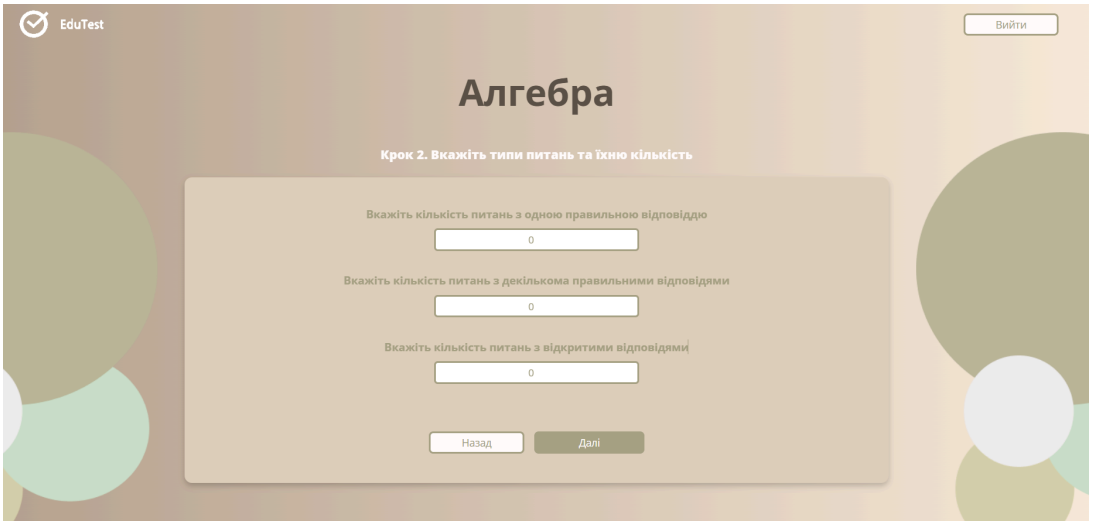

Рисунок 3.24 – Крок 2. Визначення типів і кількості питань

Після вибору типів і кількості запитань викладач має завантажити лекційний матеріал, на основі якого буде згенеровано тест (рисунок 3.25). Це можна зробити двома способами: перетягнути файл у відповідну зону або обрати його з файлового каталогу. Після завантаження відображається повідомлення про результат: «Файл успішно завантажений» або «Помилка завантаження файлу».

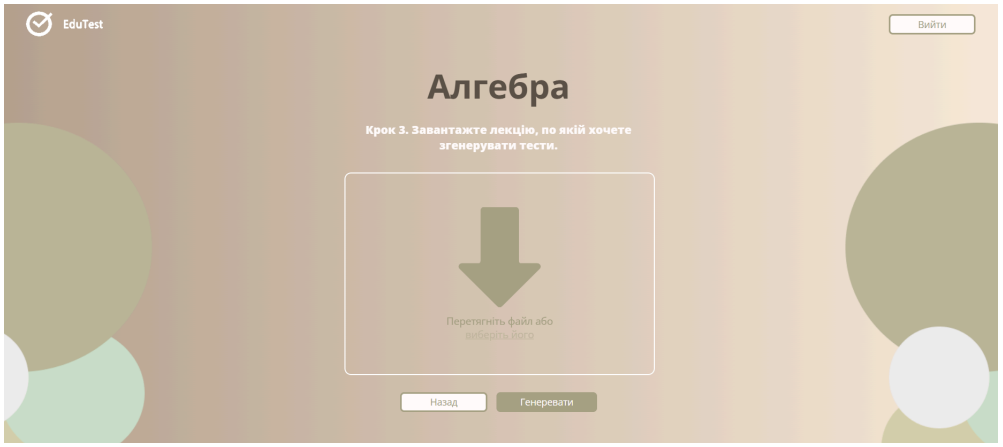

Рисунок 3.25 – Крок 3. Завантаження лекції

Після успішного завантаження тесту потрібно натиснути кнопку “Генерувати”. Далі на сторінці відобразяться готові згенеровані тести.

На рисунку 3.26 можна побачити, як виглядає сторінка студента. У лівій частині екрана є інформація про студента: ім'я "Назар", група "ІСТ-41" . У центрі сторінки є поле пошуку з написом "Пошук...", під ним відображається список предметів.

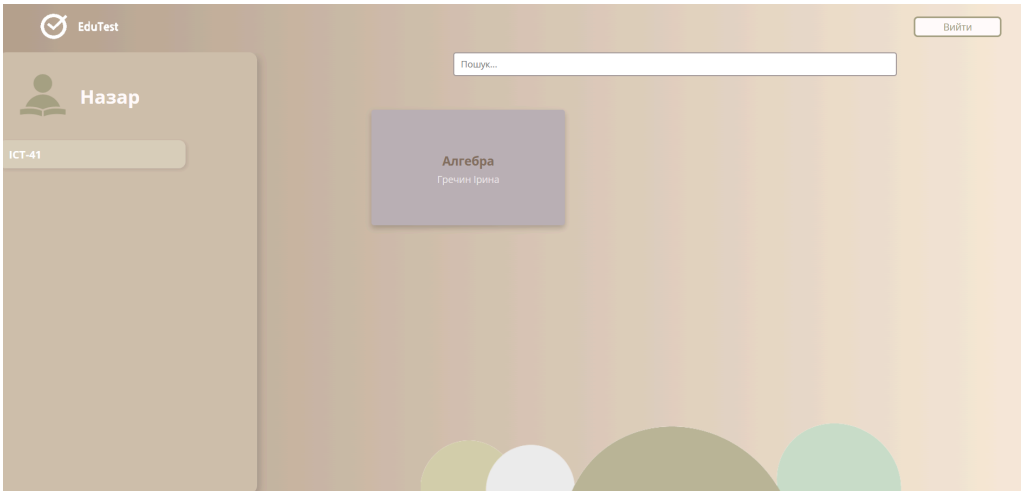

Рисунок 3.26 – Сторінка студента

При натисканні на назву предмета відкривається список доступних тестів з цього предмета (рисунок 3.27). Кожен тест можна пройти лише у тому випадку, якщо він відкритий для проходження і у студента залишилися доступні спроби. У протилежному разі тест позначатиметься як недоступний.

Рисунок 3.27 – Відображення тестів для студента

Під час проходження тесту у студента відкривається сторінка, як на рисунку 3.28.

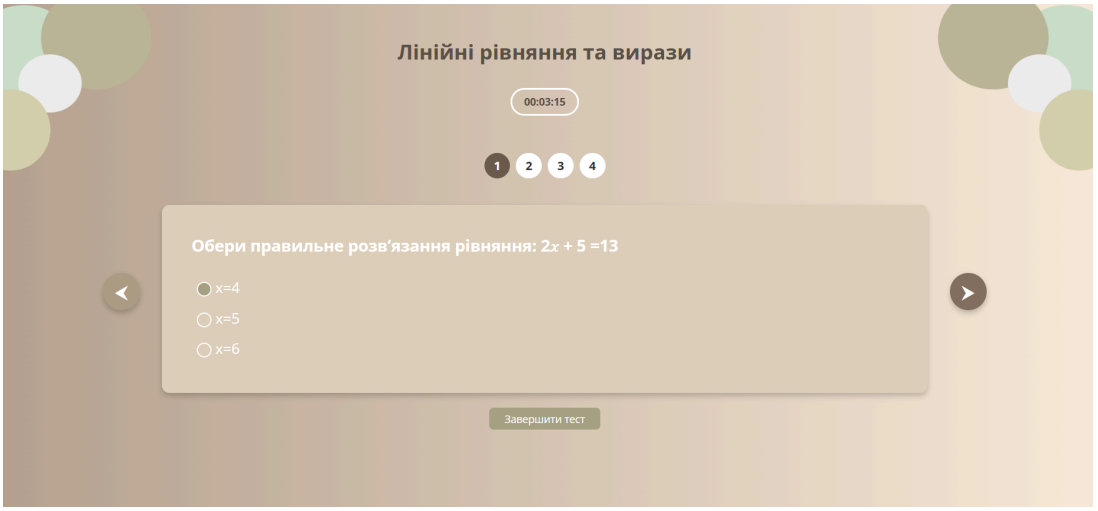

Рисунок 3.28 – Інтерфейс проходження тесту

На цій сторінці розміщено такі елементи:

- у верхній частині — заголовок із назвою теми тесту;
- таймер, який відображає залишок часу для виконання тесту;
- панель навігації із номерами завдань;
- область із поточним завданням;

- кнопки навігації (вліво та вправо) для переходу між завданнями;
- кнопка «Завершити» внизу сторінки — стає активною лише після виконання всіх завдань.

Для забезпечення академічної доброчесності під час виконання тесту студенту обмежено доступ до функцій копіювання, створення скріншотів та інструментів розробника (DevTools). Після завершення тестування студенту автоматично відображається модальне вікно з результатами, яке містить інформацію про кількість правильних відповідей (рисунок 3.29).

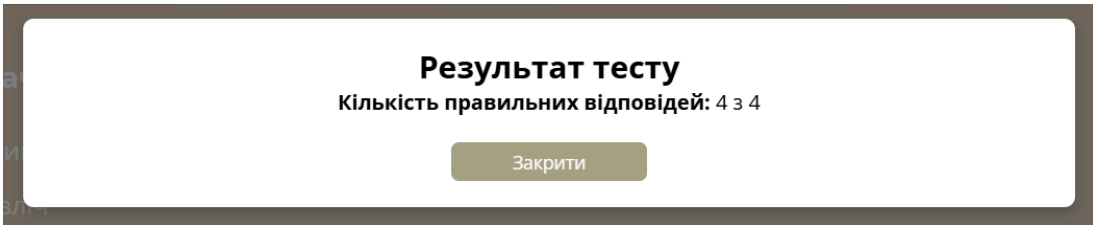

Рисунок 3.29 – Результати тесту

Після проходження тесту викладач може бачити результати студентів (рисунок 3.30).

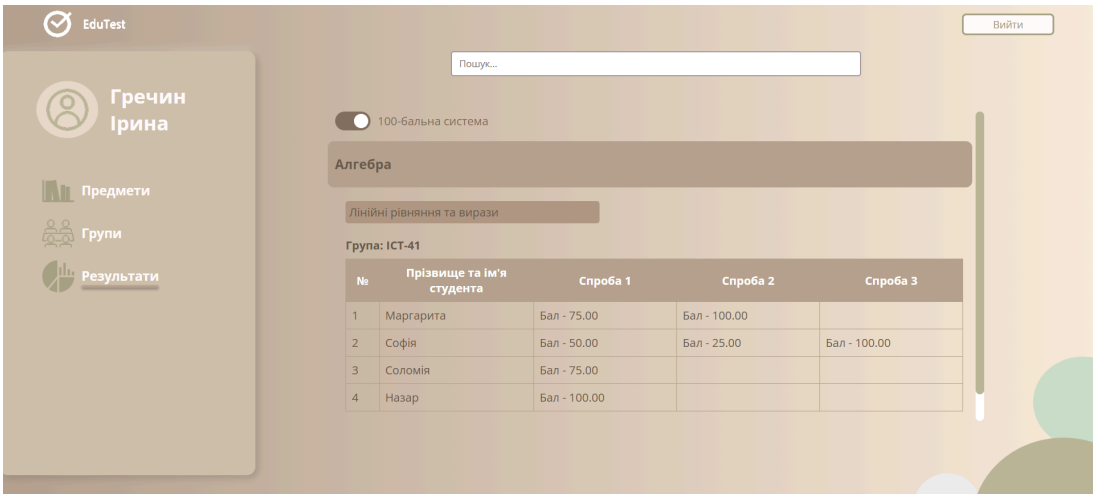

Рисунок 3.30 – Результати тесту

На цій сторінці відображаються всі предмети, створені відповідним викладачем. Після натискання на назву предмета з’являється список тестів, що належать до цього курсу. Обравши конкретний тест, викладач отримує таблицю з результатами проходження тесту для кожної академічної групи,

що була прикріплена до предмета. Це дозволяє швидко переглянути успішність студентів у межах конкретної теми чи завдання.

Висновки

У даному розділі було розглянуто процес проєктування та реалізації платформи EduTest — вебзастосунку для організації та проведення тестування у навчальному процесі. Описано загальну структуру системи та її архітектуру, яка базується на клієнт-серверній моделі з чітким розмежуванням відповідальностей між фронтендом і бекендом.

Були детально проаналізовані ключові компоненти платформи: серверна частина, побудована на основі REST API, клієнтський інтерфейс, орієнтований на дві основні ролі користувачів — викладача та студента, а також внутрішні сервіси та моделі, що забезпечують обробку і зберігання даних.

Окрему увагу приділено функціоналу викладача, який включає створення предметів, тестів, груп, перегляд результатів, та інтерфейсу студента, який дозволяє проходити тести та переглядати власні бали. Крім основного функціоналу, у платформі реалізовано авторизацію користувачів, можливість проходження тестів кілька разів, а також зручне відображення результатів за групами. Завдяки зручному інтерфейсу та продуманій архітектурі платформа забезпечує ефективну взаємодію між усіма учасниками навчального процесу.

ВИСНОВКИ

У процесі виконання бакалаврської роботи було досягнуто головної мети — розроблено веб-платформу EduTest для автоматизованого створення та проведення тестувань із використанням штучного інтелекту. В умовах активного розвитку дистанційного навчання та зростаючої потреби в ефективному інструменті для оцінювання знань студентів, реалізація такого програмного продукту є надзвичайно актуальною.

В результаті дослідження та практичної реалізації були успішно вирішені всі поставлені у вступі завдання:

1. Проведено ґрунтовний аналіз сучасних онлайн-платформ для тестування знань (Google Forms, Moodle та інші), визначено їх функціональні особливості, переваги й недоліки. Встановлено, що наявні інструменти часто не забезпечують достатньої гнучкості у створенні тестових завдань, що підкреслило необхідність розробки платформи з інтелектуальною генерацією тестів.
2. Розроблено архітектуру системи на основі клієнт-серверної моделі з чітким розмежуванням відповідальностей між фронтендом і бекендом. Використані технології (Angular, Node.js, PostgreSQL) забезпечують масштабованість, стабільність і зручність подальшого розвитку платформи EduTest.
3. Створено зручний інтерфейс користувача для двох основних ролей — викладача та студента. Інтерфейс викладача дозволяє створювати предмети, додавати групи студентів, генерувати тести за допомогою ШІ або вручну, а також переглядати детальні результати. Інтерфейс студента забезпечує зручне проходження тестів, доступ до балів та історії тестувань.
4. Реалізовано ключовий механізм платформи — генерацію тестових завдань на основі лекційного матеріалу з використанням методів

машинного навчання. Алгоритми обробки тексту лекцій дозволяють створювати запитання різних типів (множинний вибір, відкриті відповіді тощо), які точно відповідають змісту навчального матеріалу, економлячи час викладачів і підвищуючи якість тестів.

5. Проведено експериментальне тестування платформи EduTest. Порівняння ефективності автоматично згенерованих тестів із тестами, створеними вручну, показало, що ШІ здатен створювати завдання високої якості, які охоплюють ключові аспекти теми та дозволяють об’єктивно оцінити знання студентів.
6. Запропоновано шляхи вдосконалення платформи: оптимізація алгоритму генерації, розширення можливостей аналітики для викладача, інтеграція з електронними журналами та іншими освітніми сервісами.

До основних досягнень проєкту слід віднести:

- Розробка повноцінної платформи EduTest, яка об’єднує засоби управління навчальним процесом із інноваційною функцією інтелектуальної генерації тестів за допомогою ШІ;
- створення гнучкої архітектури, яка дозволяє легко розширювати функціональність;
- забезпечення високого рівня юзабіліті інтерфейсу як для викладача, так і для студента.

Проте в ході розробки було виявлено деякі проблемні питання, що потребують подальшого опрацювання. Насамперед це стосується:

- відсутності модуля перевірки якості сформованих тестів у автоматичному режимі;
- недостатньої адаптації платформи до різних мов навчання або змішаних форматів матеріалів (наприклад, текст із формулами, графіками тощо).

У перспективі розвиток платформи EduTest може йти в кількох напрямках:

- інтеграція з навчальними планами, що дозволить автоматично формувати тести відповідно до програми дисципліни;

- Використання NLP (Natural Language Processing) для глибшого аналізу лекційного матеріалу та кращої генерації змістовних запитань;
- Можливість колаборації між викладачами, створення спільних баз тестових завдань;
- Впровадження статистичного модуля з аналізом успішності, виявленням типових помилок студентів та формуванням рекомендацій для студента.

Отже, виконана бакалаврська робота продемонструвала практичну значущість використання штучного інтелекту в освітніх IT-системах, зокрема в автоматизованій генерації тестів, яка стала ключовим досягненням платформи EduTest. Цей функціонал дозволяє значно скоротити час на підготовку тестів, підвищити їхню якість і адаптувати до потреб студентів, що робить платформу інноваційним інструментом для вищих навчальних закладів, коледжів і шкільної освіти. Реалізована система не лише вирішує поточні завдання, а й відкриває широкі перспективи для подальших наукових досліджень і технічних удосконалень у сфері інтелектуальних освітніх технологій.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Биков В. Ю., Шишкіна М. П. Інноваційні технології навчання у цифровій школі. – Київ: Педагогічна думка, 2021. – 248 с. (№7)
2. Воронін В. М. Системи тестування в освіті: історія та сучасність. – Львів: ЛНУ, 2022. – 88 с. (№2)
3. Гриневич Л. М. Цифрова трансформація освіти: виклики, рішення, перспективи. – Київ: Освіта України, 2021. – 192 с. (№11)
4. Гриценко О. О. Методи тестування знань: традиційні та комп’ютерні технології. – Київ: Наукова думка, 2018. – 182 с. (№3)
5. Дьяків В. В. Системи управління навчанням: Moodle у навчальному процесі. – Київ: Ліра-К, 2021. – 196 с. (№6)
6. Романенко І. Г. Інструменти для онлайн-тестування: порівняльний аналіз. – Харків: ХНУ, 2022. – 88 с. (№9)
7. Савченко О. Я. Гейміфікація навчального процесу. – Львів: ЛНУ ім. І. Франка, 2021. – 132 с. (№8)
8. Халявка М. Д. Інформаційні технології в освіті. – Київ: Видавничий дім «Слово», 2020. – 204 с. (№5)
9. Шевчук, О. П. Психологія і педагогіка: навчальний посібник. – Х.: Основа, 2021. – 312 с. (№1)
10. AI-Assisted Assessment in Education / ed. by M. Yilmaz, B. Chang. – Cham: Springer, 2024. – 314 p. (№10)
11. Assessing by Multiple Choice Questions // UNSW Teaching Gateway : веб-сайт. – URL: https://www.teaching.unsw.edu.au/assessing-multiple-choice-questions (дата звернення: 20.05.2025). (№16)
12. Bacon B. RxJS in Action. – Shelter Island, NY: Manning Publications, 2017. – 360 p. (№23)

13. Bampakos A. Learning Angular – Fourth Edition. – Birmingham: Packt Publishing, 2023. – 450 p. (№33)
14. Bridge P., Appleyard R., Wilson R. Automated multiple-choice testing for summative assessment: What do students think? – Washington, DC: ERIC, 2007. – 19 p. (№17)
15. Coyier C., Keith J. CSS3: The Missing Manual. – Sebastopol, CA: O'Reilly Media, 2018. – 400 p. (№22)
16. Freeman A., Sanderson A. Pro Angular. – New York: Apress, 2022. – 750 p. (№19)
17. García M., Lee J. API Testing with Postman: From Beginner to Pro. – Boston: Addison-Wesley, 2021. – 320 p. (№29)
18. Gospodarek G. The Supabase Book: Build Modern Web Apps with PostgreSQL, Auth, Storage and Realtime Data. – Warsaw: LearnPub, 2022. – 184 p. (№27)
19. Haladyna T. M. Developing and Validating Multiple-Choice Test Items. – New York: Routledge, 2004. – 360 p. (№18)
20. Harper L., Nguyen T. AI-Driven Testing: Transforming Education Through Technology. – San Francisco, CA: Jossey-Bass, 2023. – 290 p. (№35)
21. Hattori Y. DevOps Unleashed with Git and GitHub. – Birmingham: Packt Publishing, 2024. – 320 p. (№30)
22. Herron A. Node.js: Modern Web Development Server-Side — 2nd ed. — Birmingham: Packt Publishing, 2019. — 350 p. (№24)
23. Holmes W., Bialik M., Fadel C. AI in Education: Promises and Implications for Teaching and Learning. – Boston: Center for Curriculum Redesign, 2019. – 105 c. (№4)
24. Kellner P. Learning WebStorm. – New York: Packt Publishing, 2019. – 250 p. (№28)
25. Kim L., Fernandez R. (eds.) Educational Technology and Mobile Learning: Contemporary Approaches and Applications. – London: Springer, 2024. – 342 p. (№13)

26. Korry Douglas, Susan Douglas. PostgreSQL: The Comprehensive Guide to Building, Programming, and Administering PostgreSQL Databases. – New York: McGraw-Hill, 2003. – 608 p. (№26)
27. Kublik S. Programming GPT-3: Building Smart Applications with the OpenAI API. – New York: Apress, 2022. – 280 p. (№25)
28. Petrova A., Smith J. (eds.) Digital Transformation in Education: Trends, Technologies, and Innovations. – New York: Routledge, 2023. – 298 p. (№12)
29. Pilgrim M. HTML5: Up and Running. – Sebastopol, CA: O'Reilly Media, 2017. – 250 p. (№21)
30. Rosenwasser D., Turnquist R. Programming TypeScript: Making Your JavaScript Applications Scale. – Sebastopol, CA: O'Reilly Media, 2020. – 250 p. (№20)
31. Smith J., Brown T. Online Assessment Security: Strategies and Technologies. – New York, NY: Springer, 2020. – 320 p. (№34)
32. Steyer M. Modern Angular. – Graz: Angular Architects, 2024. – 250 p. (№31)
33. Weiss D. J., Şahin A. Computerized Adaptive Testing: From Concept to Implementation. – New York: Guilford Press, 2024. – 360 p. (№15)
34. Wieruch R. The Road to React – 2024 Edition. – Self-published, 2024. – 350 p. (№32)
35. Woolf M., Baker B. (eds.) Artificial Intelligence in Education: Adaptive Learning Systems and Student Behavior Analysis. – Cham: Springer, 2023. – 410 p. (№14)

ДОДАТОК А

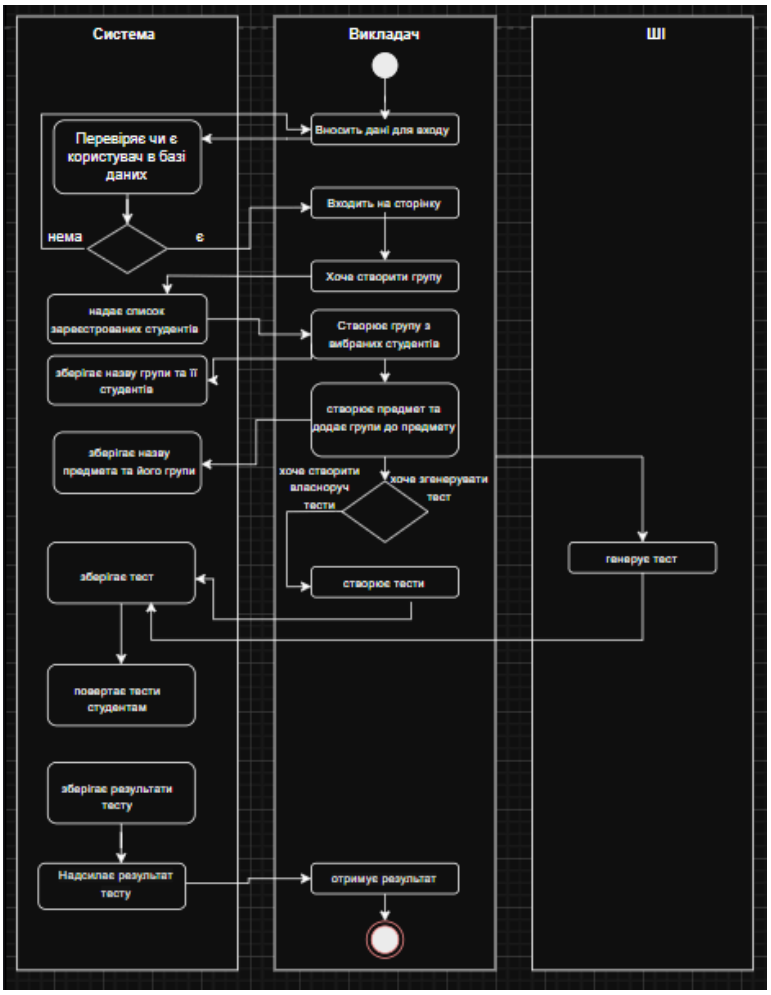

Рисунок А.1 - Діаграма взаємодії викладача із системою

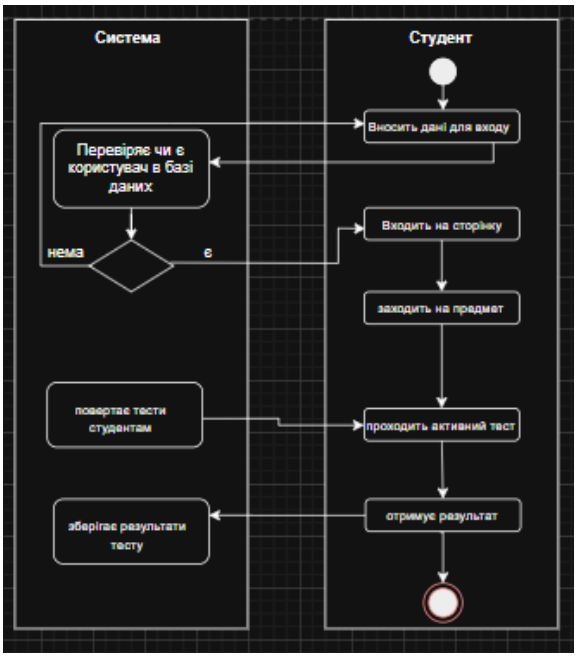

Рисунок А.2 - Діаграма взаємодії студента

ДОДАТОК Б

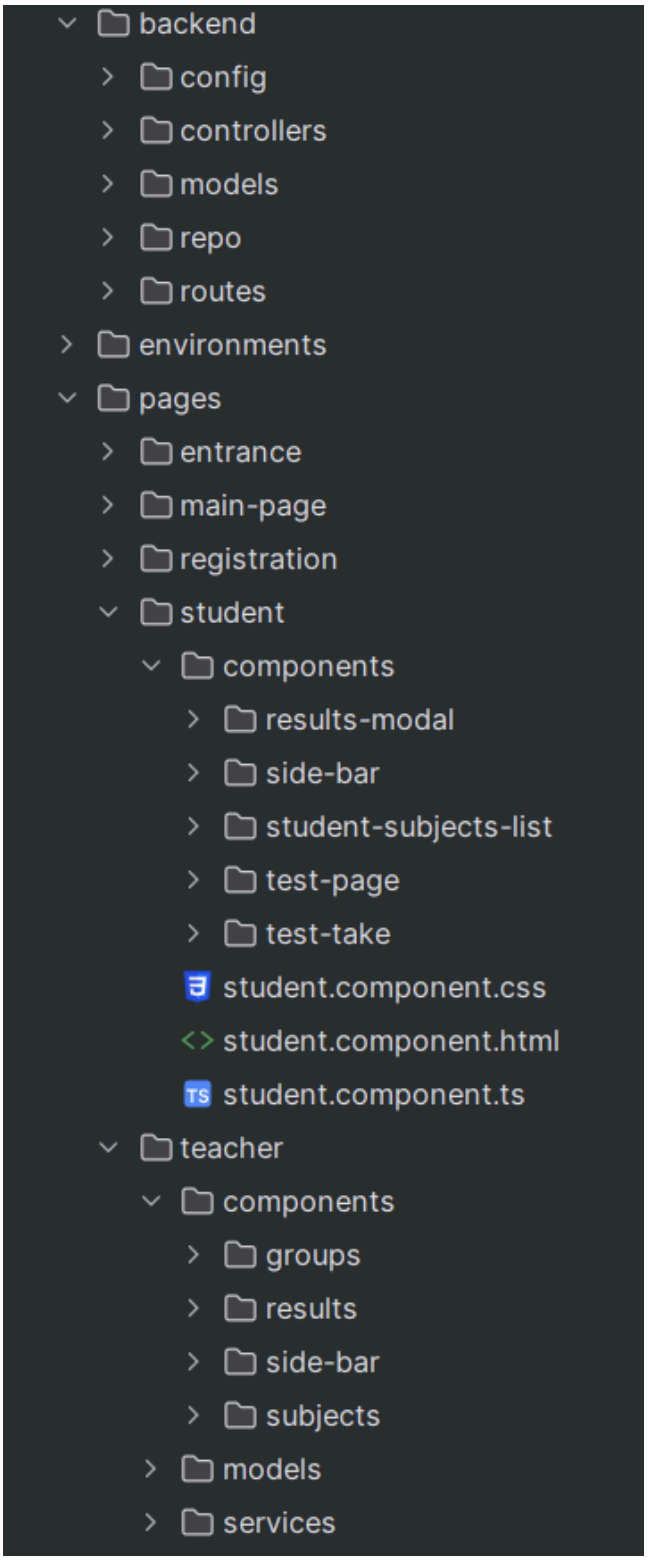

Рисунок Б.1 - Файлова структура веб-застосунку

ДОДАТОК В

```
    const prompt : string  = `
На основі  лекційного матеріалу згенеруй тестові запитання у форматі JSON. Формат кожного питання:
{
  "id": number,
  "question_text": string,
  "options": string[] | null,
  "correct_answer": string | null,
  "type": 'single' | 'multiple' | 'open'
}

Вимоги:
- ${singleCount} питань з однією правильною відповіддю (type: 'single')
- ${multipleCount} питань з декількома правильними відповідями (type: 'multiple')
- ${openCount} питань з відкритою відповіддю (type: 'open')

Ось лекція:
${lectureText}

Поверни **тільки JSON-масив питань**, без пояснень.`;
```

Рисунок В.1 - prompt, який надсилається до моделі GPT-4o

ДОДАТОК Г

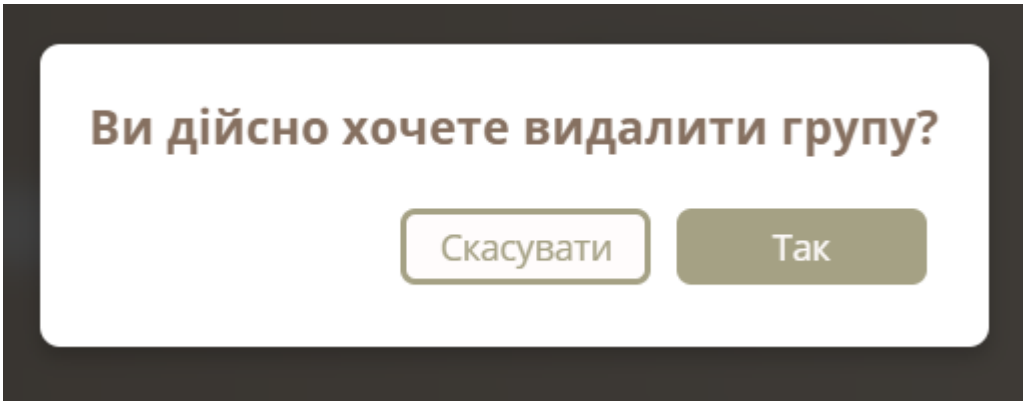

Рисунок В.1 - Модальне вікно підтвердження видалення групи

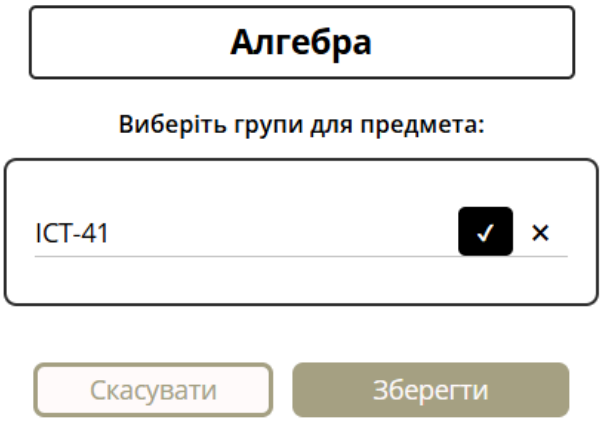

Рисунок В.2 - Модальне вікно для редагування предмету