**Форма № Н-6.01у**

# Кременчуцький національний університет імені Михайла Остроградського

(повне найменування вищого навчального закладу)

# Кафедра автоматизації та інформаційних систем

(повна назва кафедри, циклової комісії)

**КУРСОВИЙ ПРОЕКТ (РОБОТА)**

# з дисципліни «Організація баз даних та знань»

(назва дисципліни)

# на тему: «База даних онлайн тестування»

Студента 3 курсу КН-17-1 групи  
ступінь вищої освіти бакалавр

(бакалавр, магістр)

спеціальність 122 – «Комп’ютерні науки»   
освітньо-професійна програма

«Комп’ютерні науки»

Маменка О. В.

(прізвище та ініціали)

Керівник старш. викл. кафедри АІС

Васильєв Д. О.\_\_\_\_\_\_\_\_\_

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала

Кількість балів: . Оцінка: ЄКТС \_\_\_

Члени комісії Д. О. Васильєв

(підпис) (ініціали та прізвище)

М. Г. Когдась

(підпис) (ініціали та прізвище)

Т. В. Горлова

(підпис) (ініціали та прізвище)

# Кременчук 2023 рік

**Форма № Н-9.01у**

КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

Кафедра автоматизації та інформаційних систем

Дисципліна «Організація баз даних та знань»

Освітній ступінь «Бакалавр»

Спеціальність 122 – «Комп’ютерні науки»

Освітньо-професійна програма «Комп’ютерні науки»

Курс 3 групи КН-17-1 семестр 6\_\_

**ЗАВДАННЯ**

**НА КУРСОВИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТУ**

Маменку Олександру Володимировичу

(прізвище, ім’я, по батькові)

1. Тема проекту (роботи) «База даних онлайн тестування»

2. Строк подання студентом проекту (роботи) 20 травня 2020 р.

3. Вихідні дані до проекту (роботи) MySQL, мова програмування PHP, середовище розробки Visual Studio Code

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці):\_\_\_\_\_

Опис об’єктів та процесів, які відбуваються в предметній області

Моделювання бази даних

Концептуальне проектування

Логічне проектування

Фізичне проектування

Розробка клієнтської частини додатку

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

6. Дата видачі завдання 08 березня 2023 р.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  пор. | Назва етапів курсового проекту | Терміни виконання етапів проекту | Вказівки та  зауваження викладача (з зазначенням дати консультації) | Оцінювання етапів проекту | | |
| за націо- нальною шкалою | за шкалою ЄКТС | кількість балів |
| 1 | Отримання завдання |  |  |  | | |
| 2 | Аналіз предметної області |  |  |  | | |
| 3 | Вхідні і вихідні одиниці інформаційної системи |  |  |  | | |
| 4 | Проектування бази даних (концептуальне та логічне) |  |  |  | | |
| 5 | Розробка таблиць бази даних |  |  |  | | |
| 6 | Розробка клієнтської частини додатку |  |  |  | | |
| 7 | Опис інтерфейсу користувача |  |  |  | | |
|  | Оформлення пояснювальної записки |  |  |  | | |
|  |  |  |  |  | | |
|  | Подання проекту на захист |  |  |  | | |
|  | Разом | 13 тижнів |  |  | | |

Студент

(підпис)

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_Д. О. Васильєв\_\_\_\_

(підпис) (ініціали та прізвище)

«08» березня 2023 р.

РЕФЕРАТ

Курсовий проект містить: 57 сторінок, 3 розділи, 16 рис., 7 таблиць, 7 використаних джерел.

Об’єкт дослідження – онлайн тестування знань.

При виконанні курсового проекту був створений веб-сайт для проходження онлайн тестування. Код програми виконаний на мові програмування PHP з використанням веб-фреймворку Laravel. Розроблено в середовищі Visual Studio Code.

Метою роботи є створення веб-сайту для проходження простого онлайн тестування. У ході виконання курсового проекту розроблено базу даних сайту онлайн тестування та сам сайт.

Для досягнення мети в роботі вирішуються наступні задачі: реєстрація та авторизація користувачів на сайті, збереження, редагування та видалення інформації про курси, тести, запитання та відповіді.

Результатом виконання всіх етапів є повноцінний веб-додаток, який виконує всі поставлені задачі.

PHP, LARAVEL, MYSQL, MVC, МОДЕЛЮВАННЯ, БАЗА ДАНИХ, ОНЛАЙН ТЕСТУВАННЯ

**ЗМІСТ**

*Змн*.

Арк.

№ докум.

*Підпис*

Дата

Арк.

1

*122 – КП.2023.01.00 ПЗ*

*Розроб.*

*Маменко О. В.*

*Перевір.*

Васильєв Д. О.

*Н. контр.*

Горлова Т. В.

*База даних онлайн тестування*

*Пояснювальна записка*

Літ.

Аркушів

57

КрНУ Кафедра АІС

[ВСТУП 2](#_Toc231883455)

[1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ](#_Toc231883456) 7

[1.1 Опис об’єктів та процесів, які відбуваються в предметній області](#_Toc231883457) 7

[1.2 Вхідні дані інформаційної системи](#_Toc231883458) 8

[1.3 Вихідні дані інформаційної системи](#_Toc231883458) 8

[Висновки до розділу](#_Toc231883458) 8

[2 ПРОЕКТУВАННЯ БАЗИ ДАНИХ](#_Toc231883461) 9

[2.1 Моделювання бази даних](#_Toc231883466) 9

[2.2 Концептуальне проектування](#_Toc231883467) 12

[2.3 Логічне проектування](#_Toc231883467) 13

[2.4 Фізичне проектування](#_Toc231883467) 14

[Висновки до розділу](#_Toc231883467) 17

[3 ПРОГРАМНА ЧАСТИНА](#_Toc231883456) 18

[3.1 Середовище розробки додатку](#_Toc231883457) 18

[3.2 Розробка клієнтської частини додатку](#_Toc231883458) 20

[Висновки до розділу](#_Toc231883458) 26

[ВИСНОВКИ](#_Toc231883469) 27

СПИСОК ЛІТЕРАТУРИ28

Додаток А Код розробленого додатку29

**ВСТУП**

Під час інформаційної діяльності людина збирає і накопичує відомості про все, що її оточує. До появи обчислювальної техніки вся інформація зберігалася у письмовому або друкованому вигляді. Однак зі збільшенням обсягів інформації загострювалося питання її зберігання та обробки. Щоб користувач легко міг знаходити потрібну інформацію, вона має бути організована певним чином. Це стосується не лише інформації у комп’ютері, а й будь-якої інформації про об’єкти реального світу. Постійна зміна параметрів суспільних явищ, вдосконалення технологій зумовлює значне збільшення обсягів використовуваних даних. Для впорядкування їх об’єднують в певні групи за класифікаційними ознаками, зокрема, тематикою, сферою застосування тощо. Це можуть бути архіви, описи майна і матеріалів, бухгалтерські документи, особисті справи відділів кадрів, кримінальні справи на злочинців тощо. І всюди для користування інформацією необхідні засоби для її систематизації і швидкого пошуку.

База даних – сукупність даних, організованих відповідно до концепції, яка описує характеристику цих даних і взаємозв’язки між їх елементами. В загальному випадку база даних містить схеми, таблиці, подання, збережені процедури та інші об’єкти. Дані у базі організовують відповідно до моделі організації даних. Таким чином, сучасна база даних, крім саме даних, містить їх опис та може містити засоби для їх обробки.

У сучасних інформаційних системах для забезпечення роботи з базами даних використовують системи керування базами даних (СКБД). Система керування базами даних – це система, заснована на програмних та технічних засобах, яка забезпечує визначення, створення, маніпулювання, контроль, керування та використання баз даних. Застосунки для роботи з базою даних можуть бути частиною СКБД або автономними. СКБД дозволяють ефективно працювати з базами даних, обсяг яких робить неможливим їх ручне опрацювання.

Через тісний зв’язок баз даних з СКБД під терміном «база даних» інколи необґрунтовано та неточно мають на увазі систему керування базами даних. Але варто розрізняти базу даних – сховище даних, та СКБД – засоби для роботи з базою даних. СКБД з інформаційної системи може бути видалена, але база даних продовжить існувати. І навпаки: СКБД може функціонувати без жодної бази даних.

В загальному випадку базу даних неможливо просто перемістити з однієї СКБД до іншої. Але СКБД використовують стандарти, які уніфікують ряд операцій по роботі з даними і дозволяють різним застосункам працювати з базами даних різних СКБД. СКБД часто класифікують за моделлю організації даних. Найуживаніші СКБД використовують реляційну модель, у якій дані подають у виді таблиць. Для кінцевого користувача (та прикладних програм) робота з базою даних напряму неможлива. Всі маніпуляції над даними здійснюють через спеціальні запити, які надсилають до СКБД. СКБД опрацьовує їх і повертає результат. Безпосередньо з базою даних працює виключно СКБД.

Сучасні СКБД забезпечують функції щодо керування даними, які можна поділити на такі групи:

* Оголошення даних – створення, зміна та видалення визначень, які описують організацію даних;
* Модифікація даних – додавання даних, їх редагування та видалення;
* Отримання даних – надання даних за запитом застосунку у формі, яка дозволяє їх безпосереднє використання. Дані можуть надаватись або у формі, в якій вони зберігаються у базі даних, або в іншій формі (наприклад, через поєднання різних даних);
* Адміністрування даних – реєстрування та відслідковування дій користувачів, дотримання безпеки роботи з даними, забезпечення надійності та цілісності даних, моніторинг продуктивності, резервне копіювання та відновлення даних тощо.

Бази даних класифікують за різними критеріями. За моделлю організації даних розрізняють такі бази даних:

* ієрархічні;
* мережеві;
* реляційні;
* об’єктні;
* об’єктно-орієнтовані.

Ієрархічні бази даних можуть бути представлені як дерево, що складається з об’єктів різних рівнів. Верхній рівень займає один об’єкт, другий – об’єкти другого рівня і т. д.

Між об’єктами існують зв’язки, кожен об’єкт може включати в себе декілька об’єктів більш низького рівня. Такі об’єкти перебувають у відношенні предка (об’єкт більш близький до кореня) до нащадка (об’єкт більш низького рівня), при цьому можлива ситуація, коли об’єкт-предок не має нащадків або має їх декілька, тоді як у об’єкта-нащадка обов’язково тільки один предок. Об’єкти, що мають загального предка, називаються близнюками.

Мережеві бази даних подібні до ієрархічних, за винятком того, що в них є покажчики в обох напрямках, які з’єднують споріднену інформацію.

До основних понять мережевої моделі бази даних відносяться: рівень, елемент (вузол), зв’язок.

Вузол – це сукупність атрибутів даних, що описують деякий об’єкт. На схемі ієрархічного дерева вузли представляються вершинами графа. У мережній структурі кожен елемент може бути пов’язаний з будь-яким іншим елементом.

Незважаючи на те, що ця модель вирішує деякі проблеми, пов’язані з ієрархічною моделлю, виконання простих запитів залишається досить складним процесом.

Також, оскільки логіка процедури вибірки даних залежить від фізичної організації цих даних, то ця модель не є повністю незалежною від програми. Іншими словами, якщо необхідно змінити структуру даних, то потрібно змінити і додаток.

Реляційна модель орієнтована на організацію даних у вигляді двовимірних таблиць. Кожна реляційна таблиця являє собою двовимірний масив і має наступні властивості:

* кожен елемент таблиці – один елемент даних;
* всі осередки в стовбці таблиці однорідні, тобто всі елементи в стовбці мають однаковий тип (числовий, символьний тощо);
* кожен стовбець має унікальне ім’я;
* однакові рядки в таблиці відсутні;
* порядок проходження рядків і стовбців може бути довільним.

Об’єктна СУБД ідеально підходить для інтерпретації складних даних, на відміну від реляційних СУБД, де додавання нового типу даних досягається ціною втрати продуктивності або за рахунок різкого збільшення термінів і вартості розробки додатків. Об’єктна база, на відміну від реляційної, не вимагає модифікації ядра при додаванні нового типу даних. Новий клас і його екземпляри просто надходять у зовнішні структури бази даних. Система управління ними залишається без змін.

Об’єктно-орієнтована база даних (ООБД) – база даних, в якій дані оформлені у вигляді моделей об’єктів, що включають прикладні програми, які управляються зовнішніми подіями. Результатом поєднання можливостей (особливостей) баз даних і можливостей об’єктно-орієнтованих мов програмування є об’єктно-орієнтовані системи управління базами даних (ООСУБД). ООСУБД дозволяють працювати з об’єктами баз даних також, як з об’єктами у програмуванні в об’єктно-орієнтованих мовах програмування. ООСУБД розширює мови програмування, прозоро вводячи довготривалі дані, управління паралелізмом, відновлення даних, асоційовані запити й інші можливості.

Об’єктно-орієнтовані бази даних зазвичай рекомендовані для тих випадків, коли потрібна високопродуктивна обробка даних, які мають складну структуру.

За технологією обробки даних бази даних поділяються на централізовані й розподілені.

Централізована база даних зберігається у пам’яті однієї обчислювальної системи. Якщо ця обчислювальна система є компонентом мережі ЕОМ, можливий розподілений доступ до такої бази. Такий спосіб використання баз даних часто застосовують у локальних мережах ПК.

Розподілена база даних складається з декількох, можливо пересічних або навіть дублюючих одна одну частин, які зберігаються в різних ЕОМ обчислювальної мережі. Робота з такою базою здійснюється за допомогою системи управління розподіленою базою даних (СУРБД).

За способом доступу до даних бази даних поділяються на бази даних з локальним доступом і бази даних з віддаленим (мережевим) доступом.

Системи централізованих баз даних з мережевим доступом припускають різні архітектури подібних систем, такі як файл-сервер і клієнт-сервер.

Архітектура файл-сервер систем БД з мережевим доступом передбачає виділення однієї з машин мережі в якості центральної (сервер). На такій машині зберігається спільно використовувана централізована БД. Усі інші машини мережі виконують функції робочих станцій, за допомогою яких підтримується доступ користувальницької системи до централізованої бази даних. Файли бази даних відповідно до призначених для користувача запитів передаються на робочі станції, де в основному і проводиться обробка. При великій інтенсивності доступу до одних і тих же даних продуктивність інформаційної системи падає. Користувачі можуть створювати також на робочих станціях локальні БД, які використовуються ними монопольно.

У концепції клієнт-сервер мається на увазі, що крім зберігання централізованої бази даних центральна машина (сервер бази даних) повинна забезпечувати виконання основного обсягу обробки даних. Запит на дані, який видається клієнтом (робочою станцією), породжує пошук і вилучення даних на сервері. Витягнуті дані (але не файли) транспортуються по мережі від сервера до клієнта. Специфікою архітектури клієнт-сервер є використання мови запитів SOL.

При роботі з базами даних використовують мови спеціального призначення.

Мова визначення даних (Data Definition Language, DDL) – це мова, яка описує дані та структури даних, а також визначає взаємозв’язки між ними.

Мова маніпулювання даними (Data Manipulation Language, DML) – це мова, яку підтримує СКБД і яка забезпечує виконання операцій отримання, додавання, зміни та видалення даних.

Мова запитів (Query language) – це мова для користувачів, яка забезпечує отримання та оброблення даних у базі даних.

**1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ**

**1.1 Опис об’єктів та процесів, які відбуваються в предметній області**

Предметна область – сукупність пов’язаних між собою функцій, завдань управління, за допомогою яких досягається виконання поставлених цілей, це частина реального світу, що представляє інтерес для конкретного дослідження.

Інформаційним відображенням всієї предметної області економічного об’єкта служить інформаційна база. Інформаційна база складається з однієї або декількох баз даних.

Одним із напрямів удосконалення процесу навчання є розробка оперативної системи контролю знань, умінь і навичок, що дозволяє об’єктивно оцінювати знання студентів. В даний час існує велика кількість різноманітних способів проведення контролю і оцінки знань. Найбільш широко застосовується тестування як один з методів контролю засвоєння учнями знань з дисципліни, який володіє рядом визначених переваг перед традиційними методами контролю знань. Інструментом для вимірювання за шкалою досягнень учня є правильно сконструйований тест, який відповідає не тільки предмету навчання, але і його завданням і служить розвитку системного підходу до вивчення навчальної дисципліни.

База даних онлайн тестування повинна полегшувати проведення тестування. Користувачі з легкістю зможуть додавати нові курси, тести, запитання та підписки на існуючі курси, також в базі даних будуть зберігатися результати останнього пройденого користувачем тестування для кожного з тестів.

Метою є – дослідження структури бази даних та принципів її функціонування. Завдання – формування бази даних для сайту онлайн тестування, редагування, пошук, додавання нових записів.

У даній базі даних предмет дослідження – функціонування сервісу проведення онлайн тестування.

База даних створюється для сайту на якому може зареєструватися кожен бажаючий. Основне завдання бази даних збереження та захист інформації про користувачів та курси.

**1.2 Вхідні дані інформаційної системи**

Вхідними даними інформаційної системи є предметна область «Онлайн тестування», яка міститиме:

* дані про користувача (ім’я, логін, пароль, адреса електронної пошти);
* створені користувачем курси, тести, питання та відповіді на них;
* підписка користувача на курси;
* відповіді дані користувачем на запитання під час проходження тестування.

**1.3 Вихідні дані інформаційної системи**

В якості вихідних даних інформаційної системи буде веб-сайт який дозволить проводити просте онлайн тестування. Сформована база даних даного веб-сайту міститиме список користувачів, курсів, тестів, запитань, відповідей на запитання та відповіді на питання надані користувачем під час проходження ним тестування.

**Висновки до розділу**

Під час виконання аналізу предметної області було виконано дослідження предметної області, сформовано загальну структуру бази даних. Були розглянуті вхідні і вихідні дані інформаційної системи.

Для виконання цих задач, було використано спеціальну літературу, Інтернет ресурси, а також інформацію отриману підчас вивчення курсу «Організація баз даних та знань».

**2 ПРОЕКТУВАННЯ БАЗИ ДАНИХ**

**2.1 Моделювання бази даних**

Процес проектування бази даних являє собою послідовність переходів від неформального мовного опису інформаційної структури предметної області до формалізованого опису об’єктів предметної області в термінах деякої моделі.

Проектування бази даних складається з таких етапів:

* системний аналіз предметної області;
* концептуальне проектування;
* логічне проектування;
* фізичне проектування.

Системний аналіз передбачає мовний опис реальних об’єктів предметної області, визначення зв’язків між об’єктами, дослідження характеристик об’єктів і зв’язків. Результати дослідження використовуються при концептуальному проектуванні бази даних.

Для визначення складу і структури предметної області застосовуються або функціональний, або предметний підходи. Функціональний підхід застосовує рух «від задач» і використовується у тих випадках, коли заздалегідь відомі функції майбутніх користувачів бази даних, а також відомі всі задачі, для яких інформаційних потреб створюється база даних. В цьому випадку на основі виробничих документів, опитувань замовників можна чітко визначити мінімальний набір об’єктів предметної області та їх взаємозв’язок.

Предметний підхід застосовується у тому випадку, коли інформаційні потреби майбутніх користувачів чітко не визначені. В цьому випадку не можна чітко визначити мінімальний набір об’єктів предметної області. В опис предметної області включаються об’єкти та зв’язки, які є найбільш характерними та найбільш суттєвими для неї.

У практичній діяльності використовується комплексний підхід, який з одного боку дозволяє розв’язувати конкретні інформаційні та функціональні задачі, а з іншого боку – враховує можливість додавання нових застосувань.

У загальному випадку існує два підходи до проектування баз даних: низхідне проектування і висхідне проектування.

Низхідне проектування починається з визначення наборів даних, потім визначаються елементи даних для кожного з таких наборів. Цей процес включає в себе ідентифікацію різних типів сутностей і визначення атрибутів кожної сутності. Низхідне проектування включає операції декомпозиції, що передбачає заміну вихідної множини відношень, що входять в схему бази даних, іншою множиною відношень, які є проекціями вихідних відношень.

Цей підхід рекомендується застосовувати у тих випадках, коли кількість, різноманітність та складність сутностей, зв’язків і транзакцій значна за розмірами. Найбільш поширеними моделями для цього проектування є моделі «сутність-зв’язок» або ER-моделі (Entity-Relationship model). З їх допомогою визначаються важливі для предметної області об’єкти (сутності), їх властивості (атрибути) і відношення один з одним (зв’язки).

Сутність (Entity) – певний об'єкт предметної області, що буде зберігатися в базі даних. Для кожної сутності має бути визначений унікальний ідентифікатор, а кожний екземпляр сутності має однозначно ідентифікуватися і відрізнятися від всіх інших екземплярів сутності його типу. Для кожної сутності мають бути визначені певні властивості: ім’я кожної сутності має бути унікальним, і до одного і того ж імені повинна завжди застосовуватися одна й та ж інтерпретація; сутність володіє набором атрибутів, що до неї належать, або успадковуються через зв'язок; сутність має володіти атрибутом або набором атрибутів, що однозначно ідентифікуватимуть екземпляр сутності; кількість зв'язків для кожної сутності є необмеженою.

Зв'язок (Relationship) – являє собою з'єднання між двома або більше сутностями. Кожен зв'язок реалізується через значення атрибутів сутності. Зв'язок позначається дієсловом та має бути унікальним. Зв'язки характеризуються ступенями та класом належності сутності до зв'язку. Ступінь зв'язку – відношення числа сутностей, що беруть участь в утворенні зв'язку. Існує три типи зв'язків: один-до-одного, один-до-багатьох і багато-до-багатьох.

Атрибут – певна характеристика сутності, що має місце в розглянутій предметній області. Примірник атрибута – це певна характеристика окремого екземпляру сутності. Атрибут визначається типом характеристики та набором значень, що він може приймати. Атрибути є двох видів: обов'язкові, необов'язкові. Для першого випадку атрибут не може приймати невизначених значень, а для іншого навпаки. Атрибут може описувати деяку властивість, а також входити до складу унікального ідентифікатора.

Висхідне проектування починається з виявлення елементів даних, які потім групуються в набори даних.

Спочатку визначаються атрибути, які потім об’єднуються в сутності. Висхідне проектування включає операції синтезу, що передбачає виконання компоновки із заданої множини функціональних залежностей між об’єктами предметної області вихідних відношень схеми бази даних.

Цей підхід рекомендується застосовувати у тому випадку, якщо розробляється невелика база даних з незначною кількістю об’єктів, атрибутів і транзакцій.

Концептуальне проектування полягає в створенні концептуальної моделі, яку відображає концептуальна схема бази даних. На цьому етапі визначаються об’єкти, зв’язки між об’єктами, атрибути, ключові атрибути.

Логічне проектування полягає в створенні логічної моделі на основі вибраної моделі даних. На цьому етапі необхідно вже знати яка СУБД буде застосовуватися в системі (ієрархічна, мережна, реляційна, об’єктно-орієнтована). Для перевірки вірності логічної моделі застосовується нормалізація. Крім того логічна модель перевіряється на умову забезпечення всіх транзакцій користувачів. Фізичне проектування полягає в описі засобів фізичної реалізації логічного проекту БД. Фізичні моделі визначають засоби розміщення даних в середовищі зберігання і засоби доступу до цих даних, які підтримуються на фізичному рівні.

**2.2 Концептуальне проектування**

Концептуальна модель бази даних – це високорівнева об’єктно-орієнтована модель предметної області, що представляє собою об’єктну область у вигляді набору об’єктів, що володіють певними властивостями та знаходяться у певних відносинах.

Основна мета розробки високорівневої моделі даних полягає в створенні моделі користувальницького сприйняття даних та узгодженні більшої кількості технічних аспектів, зв’язаних з проектуванням бази даних. Концептуальна модель даних не прив’язана до конкретної фізичної реалізації баз даних та не залежить від конкретної СУБД.

Конкретний вид і зміст концептуальної моделі бази даних визначається обраним для цього формальним апаратом. Зазвичай використовуються графічні нотації, подібні ER-діаграмам.

Концептуальна модель створюється на основі представлень про предметну область кожного типу користувачів, що представляють собою набір даних, що необхідні користувачу для рішення своїх задач. Основні концепції моделі включають такі поняття як сутність (об’єкт), відношення (зв'язок), типи сутностей, типи зв’язків та атрибути.

Із предметної області винесемо основні сутності:

* користувач;
* курс;
* тест;
* питання;
* відповідь.

Концептуальна модель бази даних онлайн тестування зображена на рис. 2.1.

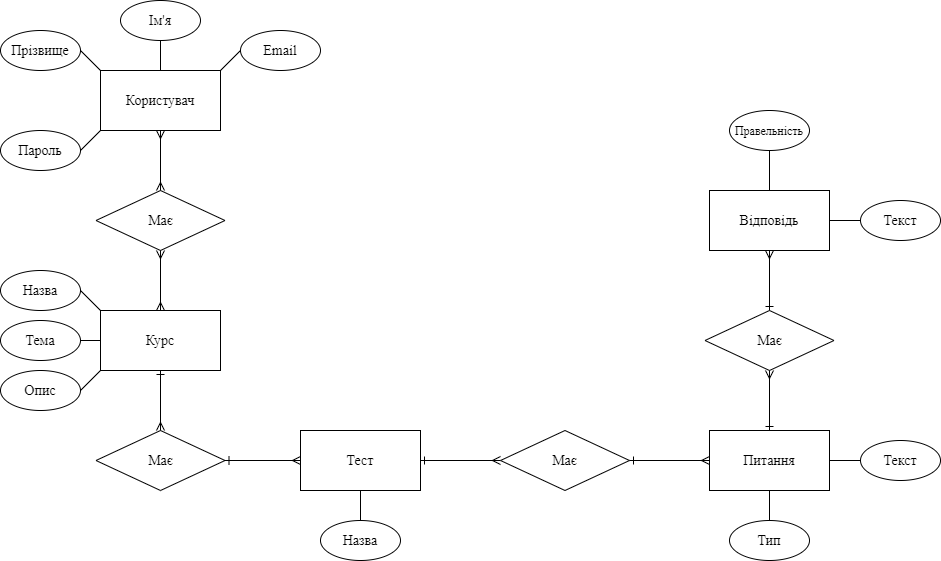


Рисунок 2.1 – Концептуальна модель бази даних

**2.3 Логічне проектування**

Логічне проектування полягає в створенні логічної моделі на основі вибраної моделі даних. На цьому етапі необхідно визначити яка СУБД буде застосовуватися в системі. Для перевірки вірності логічної моделі застосовується нормалізація. Перетворення локальної концептуальної моделі даних в локальну логічну модель полягає у видаленні із концептуальної моделі небажаних елементів та перетворення отриманих моделей в локальні логічні моделі. До небажаних елементів відносяться:

* зв’язки типу «багато-до-багатьох»;
* рекурсивні зв’язки;
* зв’язки з атрибутами.

Логічна модель бази даних онлайн тестування зображена на рис. 2.2.

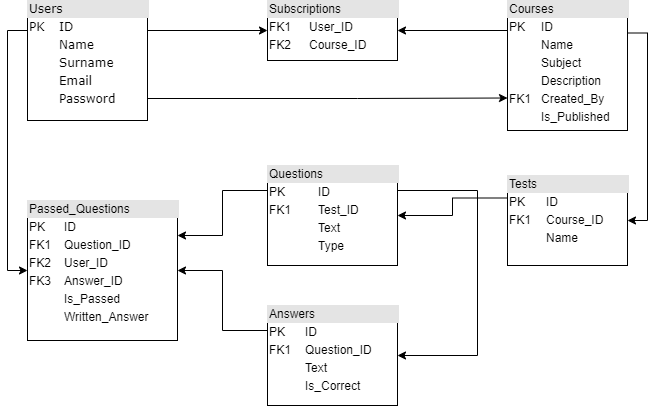


Рисунок 2.2 – Логічна модель бази даних

**2.3 Фізичне проектування**

Фізичне проектування полягає в описі засобів фізичної реалізації логічного проекту бази даних. Фізичні моделі визначають засоби розміщення даних в середовищі зберігання і засоби доступу до цих даних, які підтримуються на фізичному рівні. У процесі проектування визначається структура реляційної бази даних (склад таблиць, їх структура і логічні зв’язки). Структура таблиці визначається складом стовпців, типом даних, ключами таблиці.

Фізична модель бази даних онлайн тестування зображена на рис. 2.3.

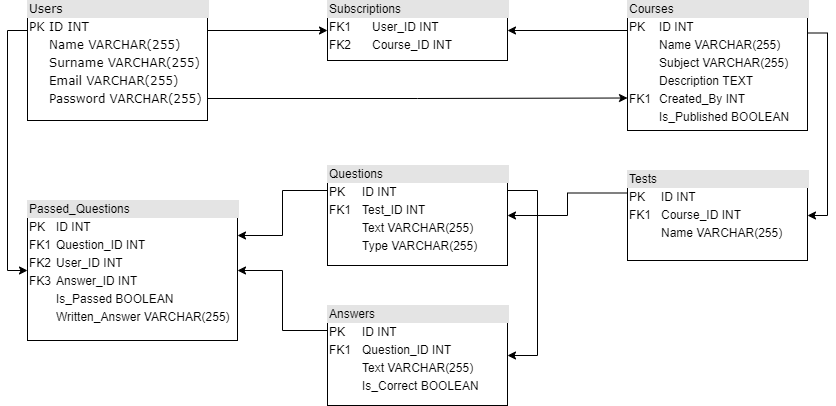


Рисунок 2.3 – Фізична модель бази даних

Таблиця «Users» містить інформацію про зареєстрованих користувачів.

Таблиця 2.1 – «Users»

|  |  |  |
| --- | --- | --- |
| Поле таблиці | Тип даних | Опис поля |
| ID | Integer | Унікальний ідентифікатор користувача. |
| Name | Varchar(255) | Ім’я користувача. |
| Surname | Varchar(255) | Прізвище користувача. |
| Email | Varchar(255) | Адреса електронної пошти користувача. |
| Password | Varchar(255) | Пароль користувача. |

Таблиця «Courses» містить інформацію про курси.

Таблиця 2.2 – «Courses»

|  |  |  |
| --- | --- | --- |
| Поле таблиці | Тип даних | Опис поля |
| ID | Integer | Унікальний ідентифікатор курсу. |
| Name | Varchar(255) | Назва курсу. |
| Subject | Varchar(255) | Тема курсу. |
| Description | Text | Опис курсу. |
| Created\_By | Integer | Ідентифікатор користувача, який створив даний курс. |
| Is\_Published | Boolean | Стан публікації курсу. |

Таблиця «Subscriptions» містить інформацію про підписку користувача на певний курс.

Таблиця 2.3 – «Subscriptions»

|  |  |  |
| --- | --- | --- |
| Поле таблиці | Тип даних | Опис поля |
| User\_ID | Integer | Ідентифікатор користувача, який підписався на курс. |
| Course\_ID | Integer | Ідентифікатор курсу на який підписався користувач. |

Таблиця «Tests» містить інформацію про тест.

Таблиця 2.4 – «Tests»

|  |  |  |
| --- | --- | --- |
| Поле таблиці | Тип даних | Опис поля |
| ID | Integer | Унікальний ідентифікатор тесту. |
| Course\_ID | Integer | Ідентифікатор курсу до якого належить даний тест. |
| Name | Varchar(255) | Назва тесту. |

Таблиця «Questions» містить інформацію про питання.

Таблиця 2.5 – «Questions»

| Поле таблиці | Тип даних | Опис поля |
| --- | --- | --- |
| ID | Integer | Унікальний ідентифікатор запитання. |
| Test\_ID | Integer | Ідентифікатор тесту до якого належить дане питання. |
| Text | Varchar(255) | Текс запитання. |
| Type | Varchar(15) | Тип запитання. |

Таблиця «Answers» містить відповіді на питання.

Таблиця 2.6 – «Answers»

|  |  |  |
| --- | --- | --- |
| Поле таблиці | Тип даних | Опис поля |
| ID | Integer | Унікальний ідентифікатор відповіді. |
| Question\_ID | Integer | Ідентифікатор запитання. |
| Text | Varchar(35) | Текст відповіді. |
| Is\_Correct | Boolean | Правильність відповіді. |

Таблиця «Passed\_Questions» містить інформацію про пройдені запитання.

Таблиця 2.7 – «Passed\_Questions»

|  |  |  |
| --- | --- | --- |
| Поле таблиці | Тип даних | Опис поля |
| ID | Integer | Унікальний ідентифікатор пройденого запитання. |
| Question\_ID | Integer | Ідентифікатор запитання. |
| User\_ID | Integer | Ідентифікатор користувача. |
| Answer\_ID | Integer | Ідентифікатор відповіді користувача. |
| Written\_Answer | Varchar(255) | Текст відповіді користувача. |
| Is\_Passed | Boolean | Правильність відповіді на запитання. |

**Висновки до розділу**

Під час проектування бази даних були розкриті особливості її моделювання та структури. Були розглянуті три етапи проектування бази даних, а саме: концептуальний, логічний і фізичний. Була описана внутрішня структура даних, таблиці та їх поля.

**3 ПРОГРАМНА ЧАСТИНА**

**3.1 Середовище розробки**

Для розробки бази даних було обрано одну з найпоширеніших СКБД – MySQL.

Система управління базами даних MySQL – розробка шведської компанії MySQL AB. СУБД MySQL є програмним забезпеченням з відкритим вихідним кодом, поширюваним за ліцензією GNU (GPL) і комерційною ліцензією для ситуацій, які не підпадають під дію ліцензії GPL. MySQL підтримує реляційну модель даних, тобто є реляційною СУБД.

Починаючи з версії 5.0, СУБД MySQL практично повністю задовольняє стандарту структурованої мови запитів SQL.

Розглянемо основні переваги СУБД MySQL:

* + Висока якість. MySQL характеризується стійкою роботою.
  + Поряд з Oracle, MySQL вважається однією з найшвидших СУБД в світі.
  + Відкритий код доступний для перегляду і модернізації, що дозволяє постійно покращувати програмний продукт.
  + MySQL, розроблена з використанням мов C / C ++, протестована на багатьох платформах, серед яких Windows, Linux, FreeBSD, Mac OS X, OS / 2, Solaris та інші.
  + MySQL підтримує API для С, C ++, Eiffel, Java, Perl, PHP, Python, Ruby і Tcl. MySQL можна успішно застосовувати як для побудови веб-сторінок з використанням Perl, PHP і Java, так і для роботи прикладної програми, створеної з використанням Delphi, Builder C ++ або платформи .NET.
  + MySQL надає широкий вибір типів таблиць, в тому числі і сторонніх розробників, що дозволяє реалізувати оптимальну для розв'язуваної задачі продуктивність і функціональність.

Ще в версії MySQL 4.1 з'явилися такі важливі нововведення, як повна підтримка вкладених запитів і підтримка транзакцій. А у версії MySQL 5.0 стали доступними такі важливі механізми:

* + збережені процедури і функції, які об'єднують в собі цілі послідовності запитів;
  + тригери, тобто збережені процедури, прив'язані до події зміни таблиці;
  + уявлення – вибірки даних, які можна уявити як повноцінні реально існуючі таблиці бази даних;
  + курсори, що дозволяють в циклі переглянути кожен рядок результуючої таблиці запитів;
  + інформаційна схема, тобто стерпний набір уявлень системної таблиці, в якій зберігається різноманітна внутрішня інформація;
  + обробники помилок;
  + безліч нових функцій.

При розробці додатку був використаний веб-фреймворк Laravel.

Laravel – це безкоштовний PHP веб-фреймворк з відкритим кодом, призначений для розробки веб-додатків за архітектурною схемою model-view-controller (MVC).

Модель-представлення-контролер (англ. Model-View-Controller, MVC) – архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення.

Мета шаблону – гнучкий дизайн програмного забезпечення, який повинен полегшувати подальші зміни чи розширення програм, а також надавати можливість повторного використання окремих компонентів програми. Крім того використання цього шаблону у великих системах сприяє впорядкованості їхньої структури і робить їх більш зрозумілими за рахунок зменшення складності.

У рамках архітектурного шаблону MVC програма поділяється на три окремі, але взаємопов'язані частини з розподілом функцій між компонентами. Модель (Model) відповідає за зберігання даних та їх структуру. Вигляд (View) відповідальний за представлення цих даних користувачеві, тобто інтерфейс програми. Контролер (Controller) керує компонентами, отримує сигнали у вигляді реакції на дії користувача (зміна положення курсора миші, натискання кнопки, ввід даних в текстове поле) і передає дані у модель.

**3.2 Розробка клієнтської частини додатку**

Зайшовши на сайт користувач потрапить на головну сторінку (рис. 3.1), де відображаються усі опубліковані курси.

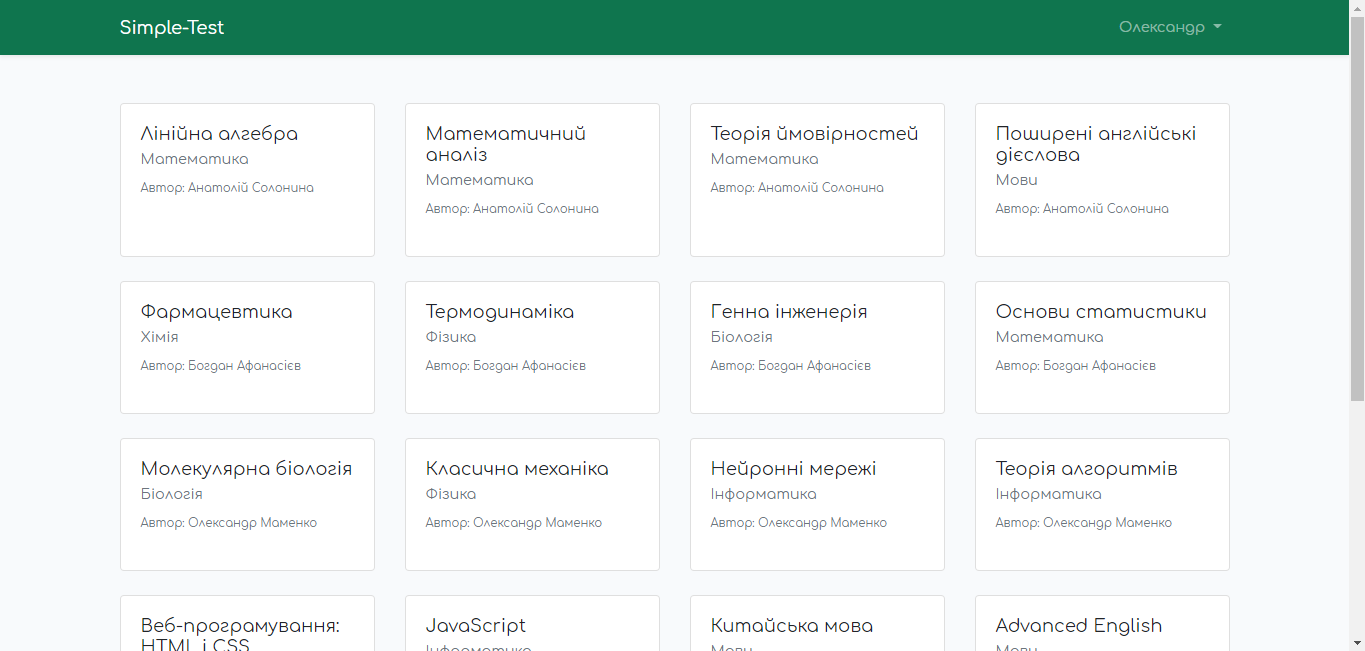


Рисунок 3.1 – Головна сторінка

Для подальшої взаємодії з сайтом користувачеві потрібно зареєструватися (рис. 3.2) або увійти (рис. 3.3), якщо він уже зареєстрований.

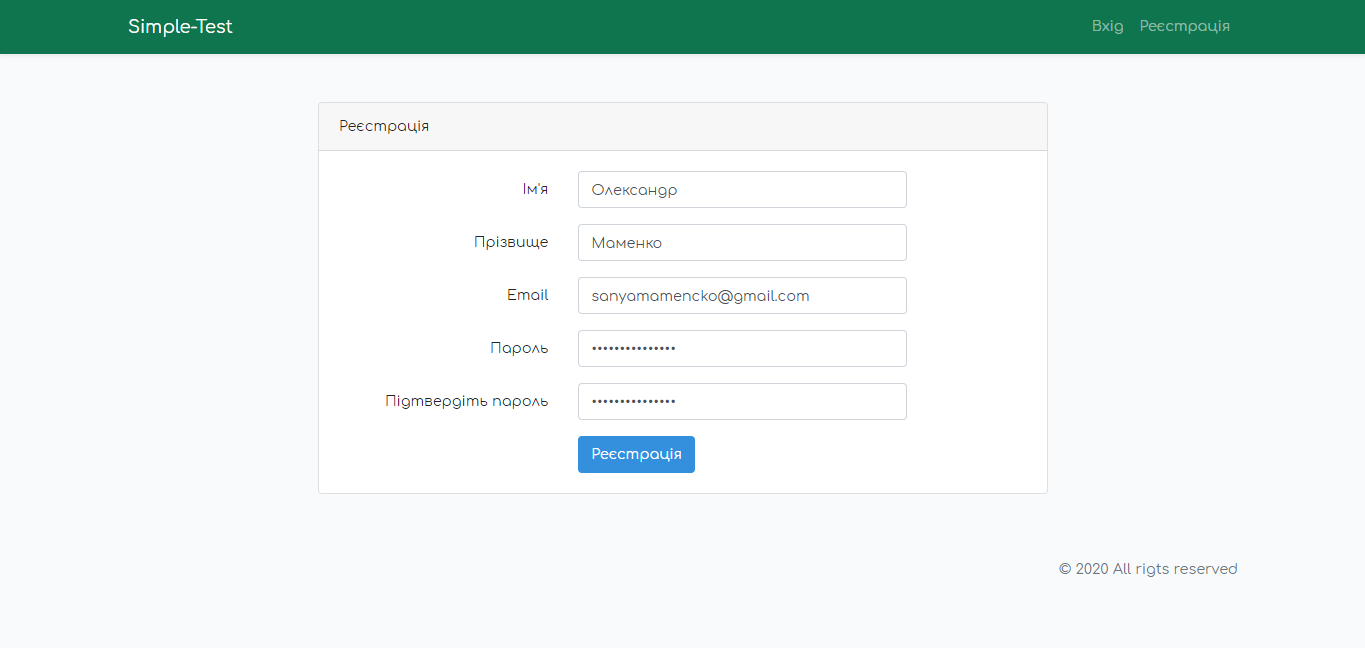


Рисунок 3.2 – Форма реєстрації

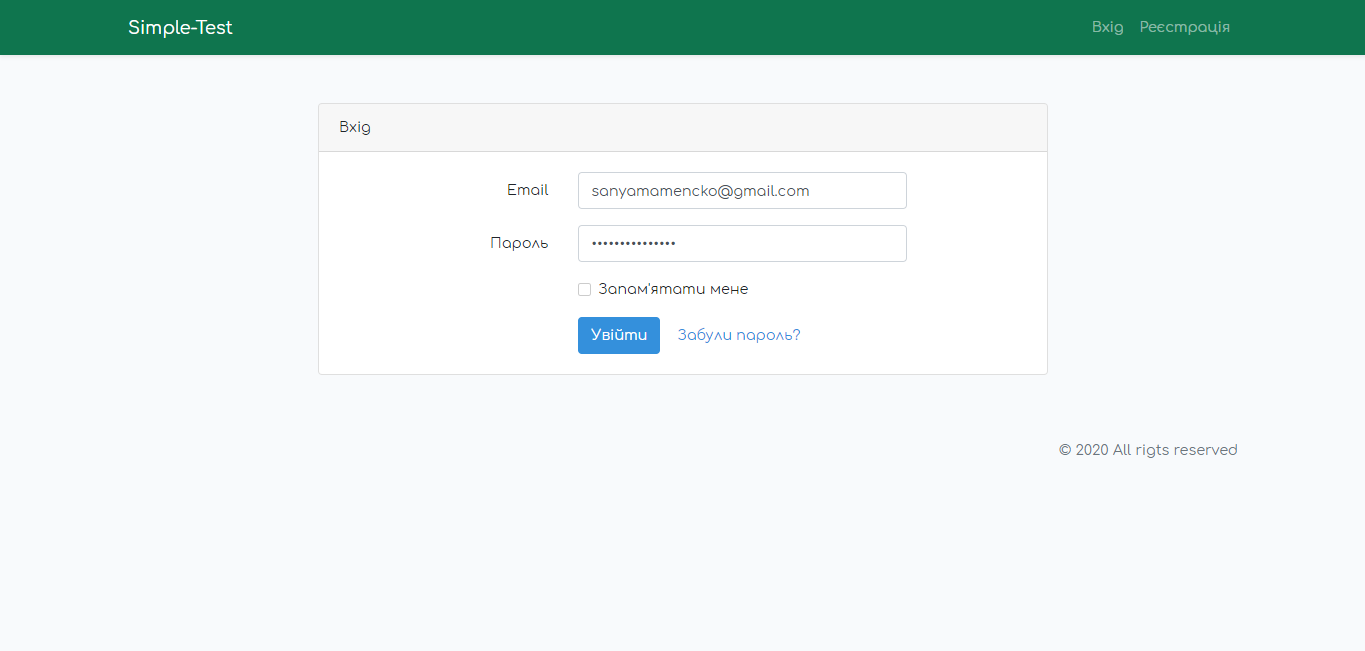


Рисунок 3.3 – Форма авторизації

Після реєстрації або входу, користувача буде пере направлено на сторінку його профілю (рис. 3.4).

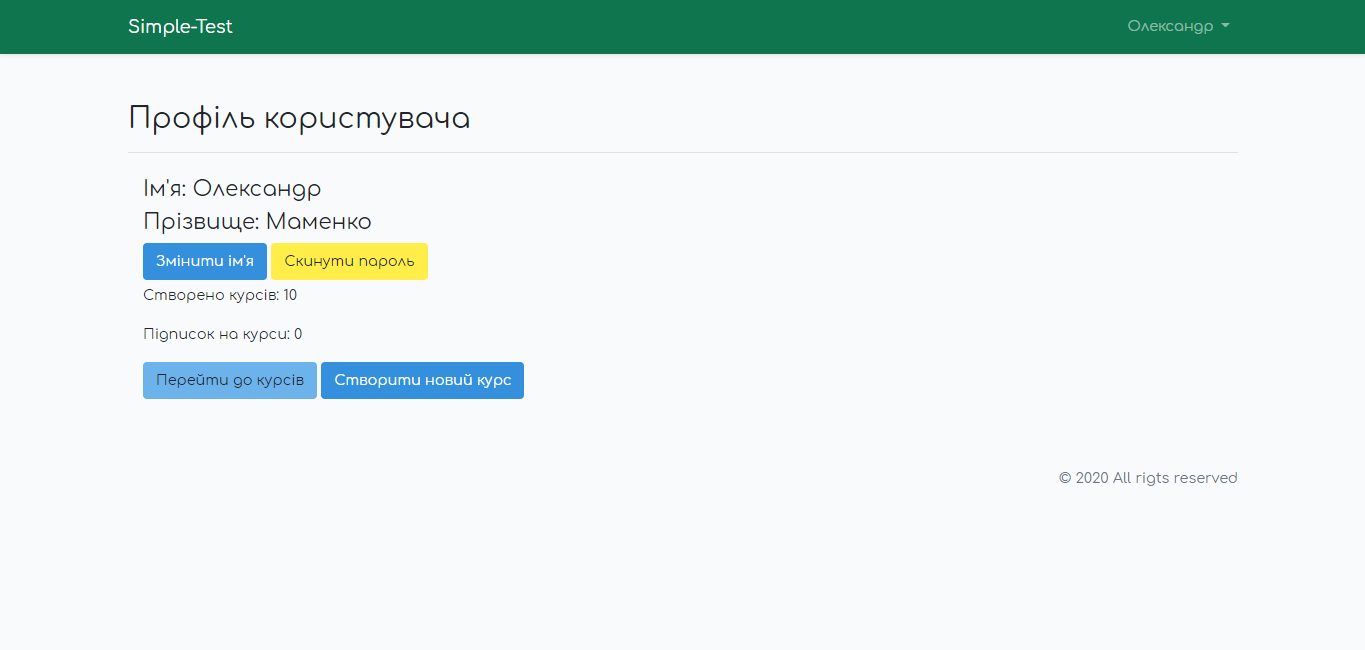


Рисунок 3.4 – Сторінка профілю користувача

Користувач може переглянути свої підписки на курси (рис. 3.5) та створені ним курси (рис. 3.6) або створити новий курс. Також, за бажанням, користувач може змінити своє ім’я (рис. 3.7).

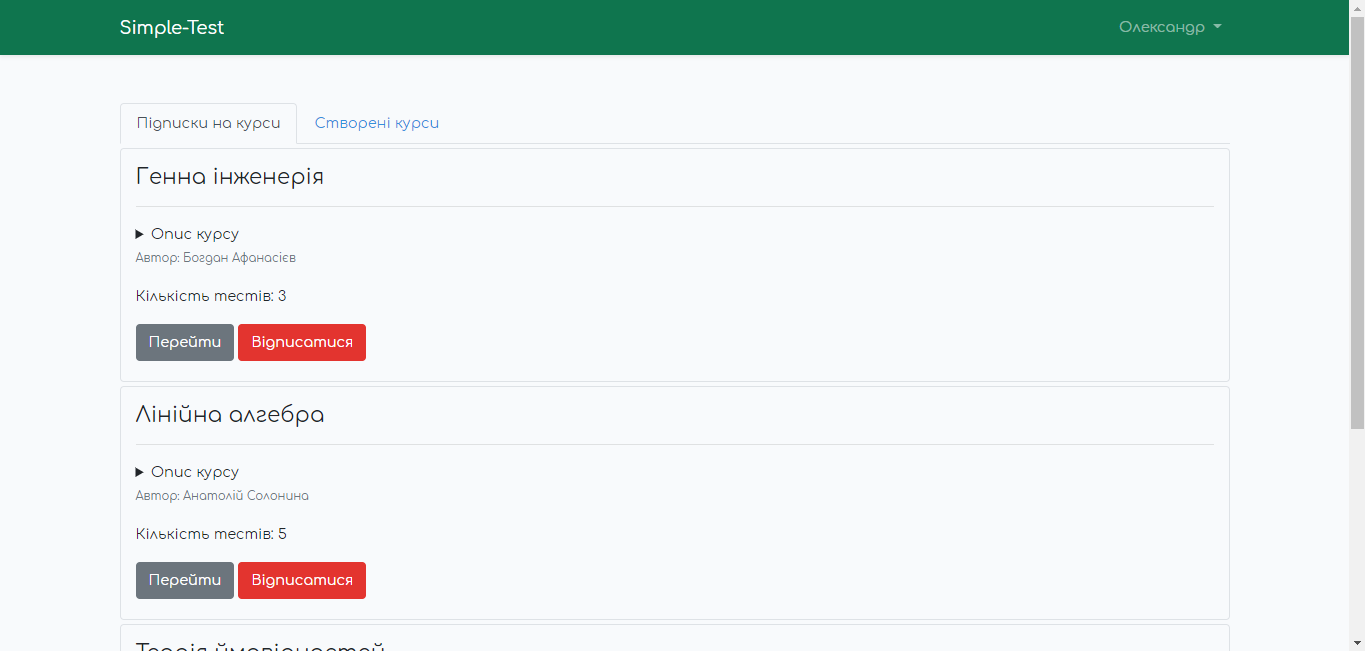


Рисунок 3.5 – Підписки користувача на курси

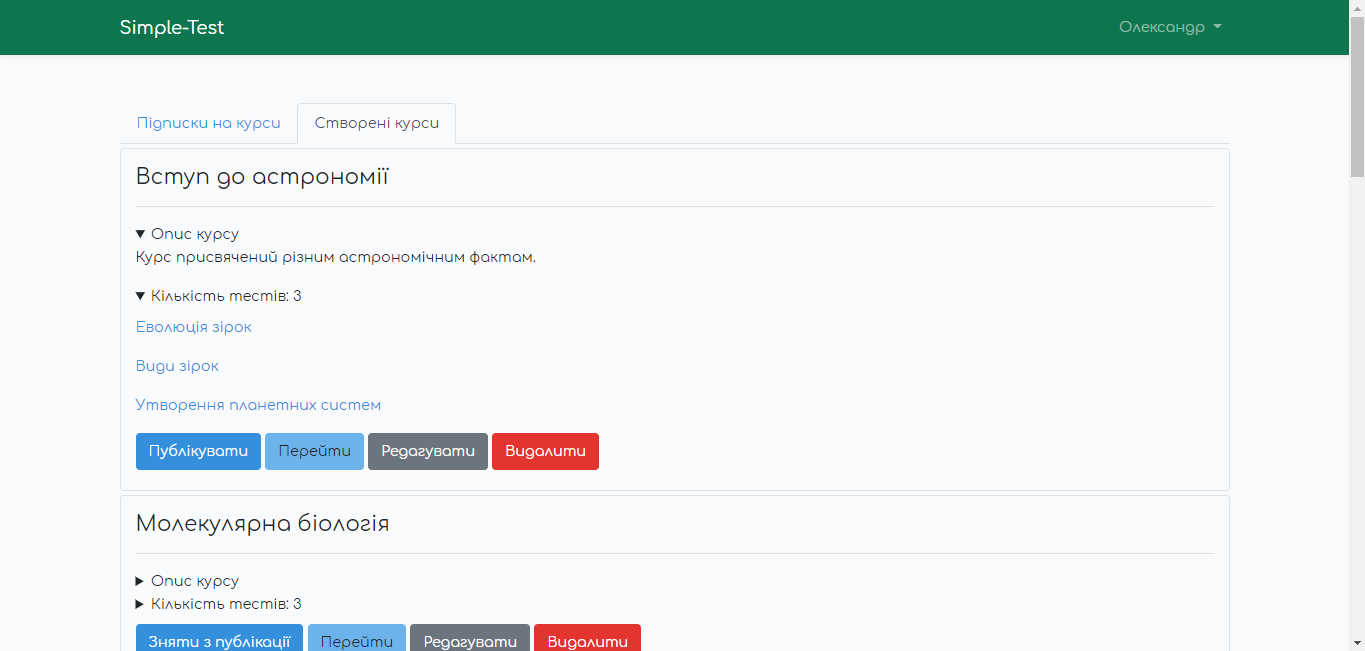


Рисунок 3.5 – Створені користувачем курси

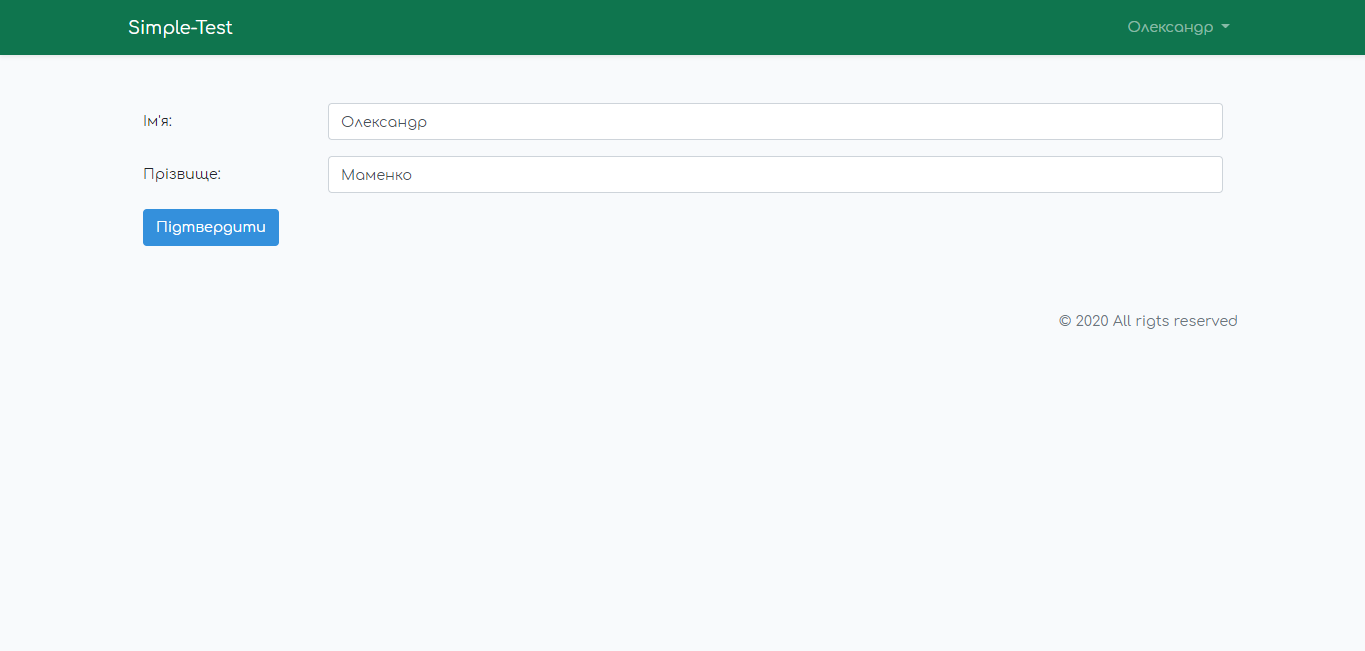


Рисунок 3.7 – Форма зміни імені користувача

Створюючи курс (рис. 3.8) користувач повинен вказати його назву, тему та необов’язково, опис курсу.

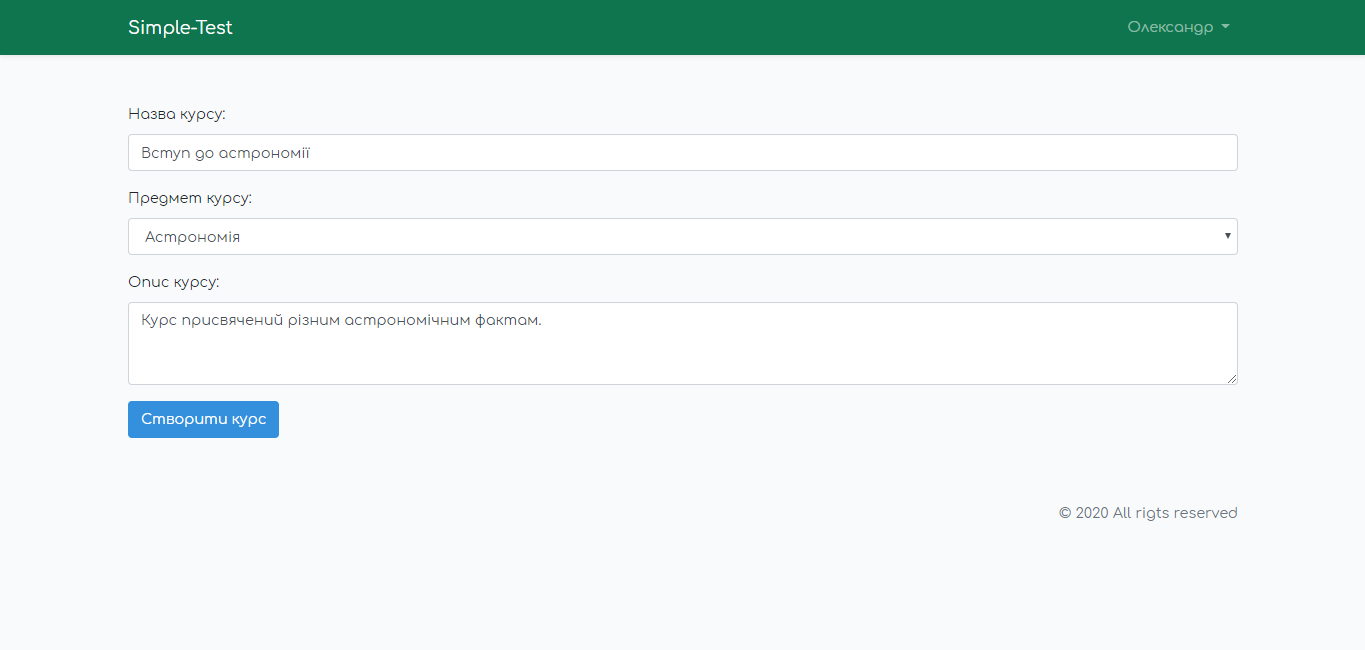


Рисунок 3.8 – Форма створення курсу

Створений курс можна відредагувати та додати до нього тести (рис. 3.9).

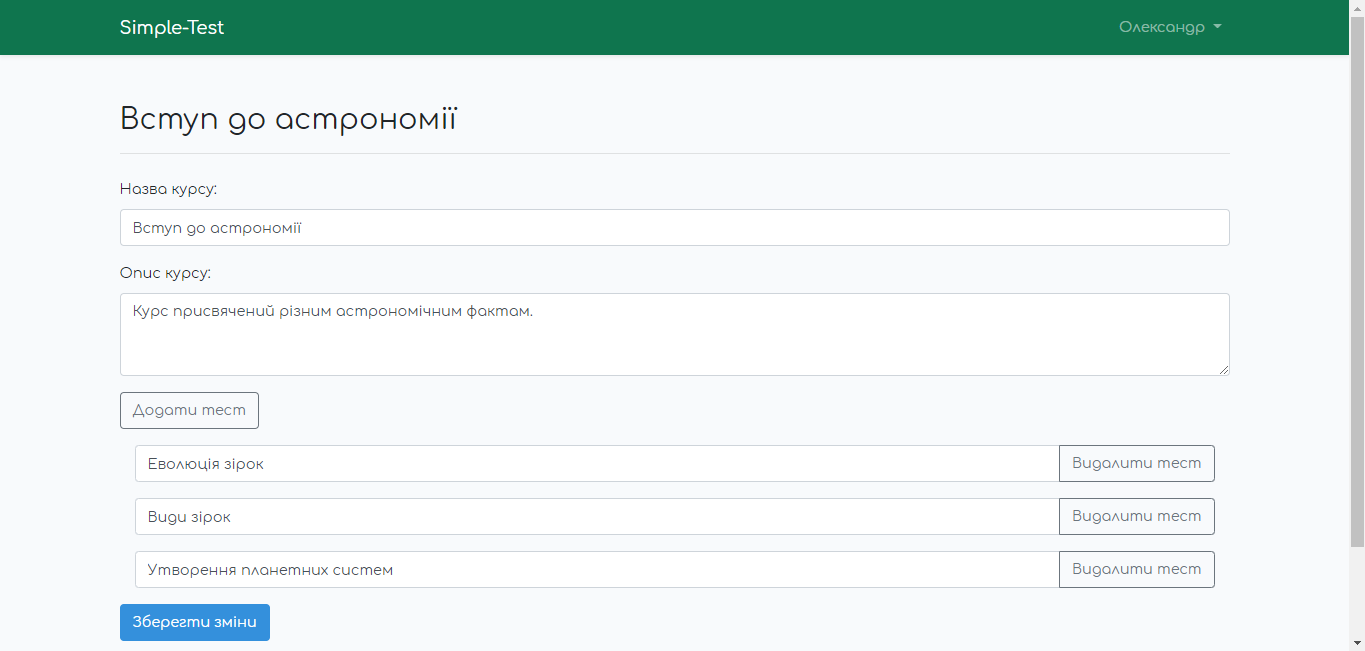


Рисунок 3.9 – Форма редагування курсу

Додавши тести користувач може додавати до них нові запитання та редагувати існуючі (рис. 3.10). Користувач має три типи запитань: з однією правильною відповіддю, з декількома правильними відповідями та з необхідністю записати відповідь. Для перших двох типів може бути від двох до шести варіантів відповідей.

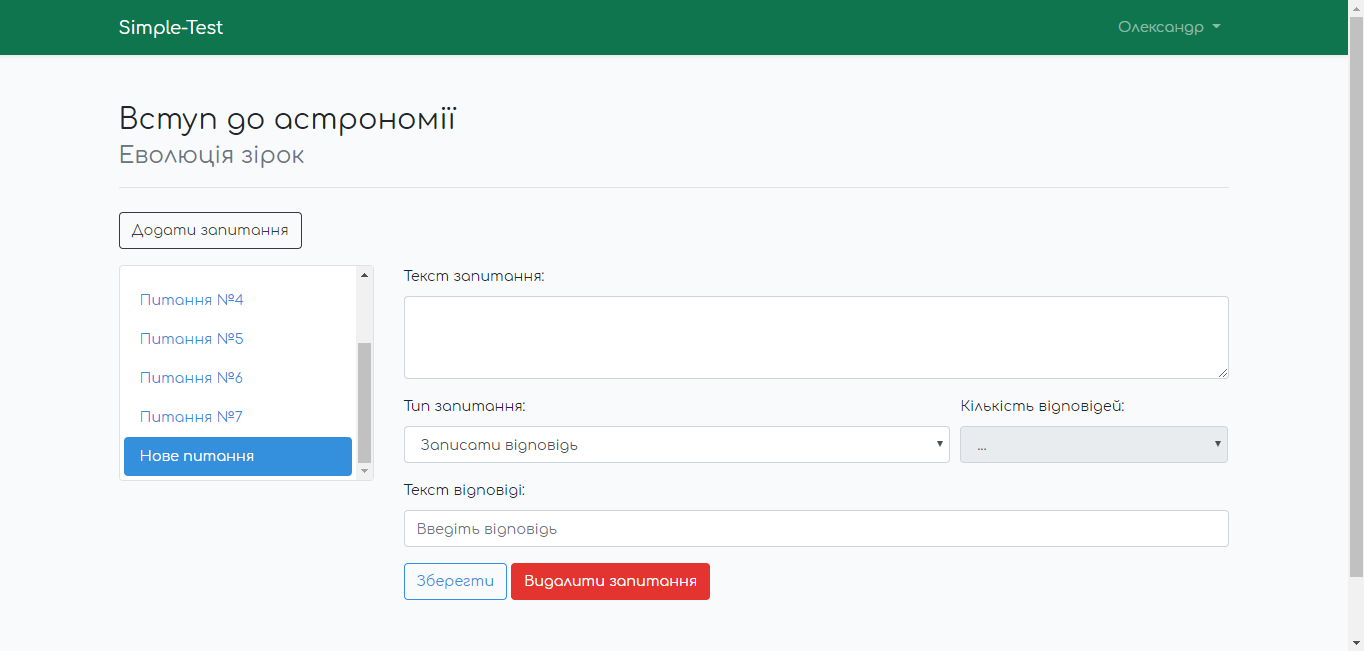


Рисунок 3.10 – Форма редагування запитань

Користувач може переглянути інформацію про будь-який курс (рис. 3.11), там він може підписатися на курс, почати проходити тести (рис. 3.12) та переглянути результати останнього тестування (рис. 3.13).

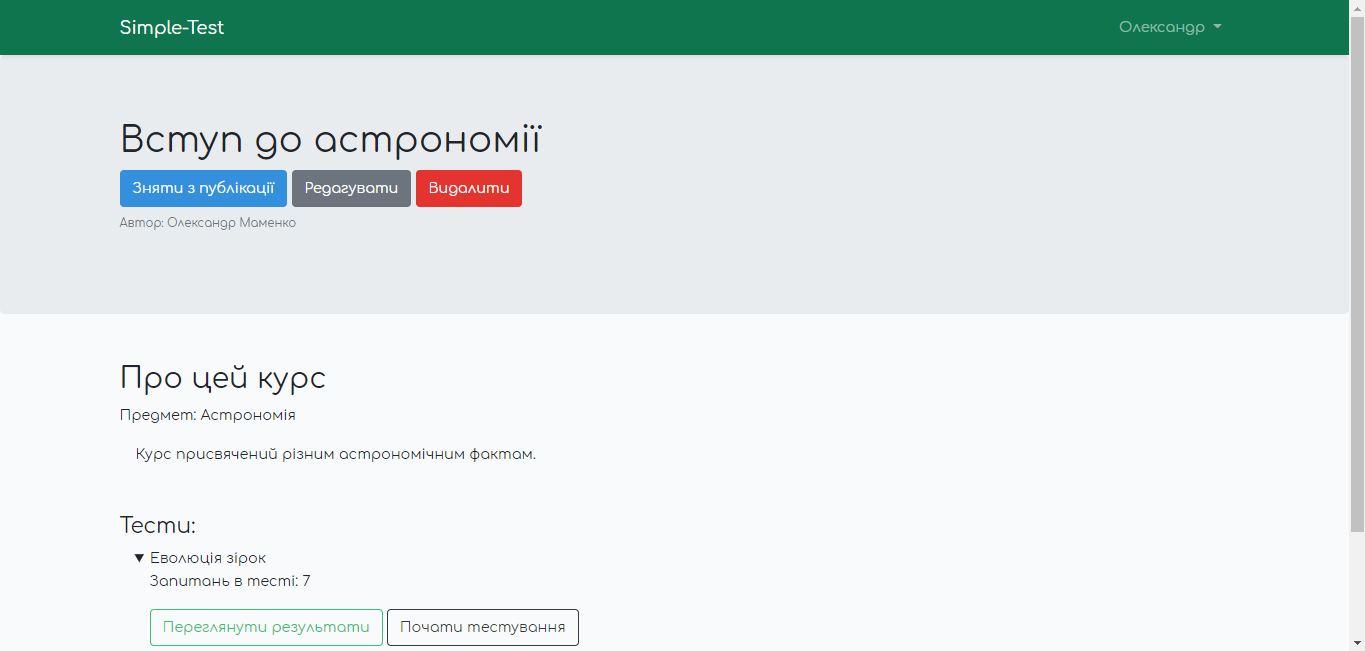


Рисунок 3.11 – Інформація про курс

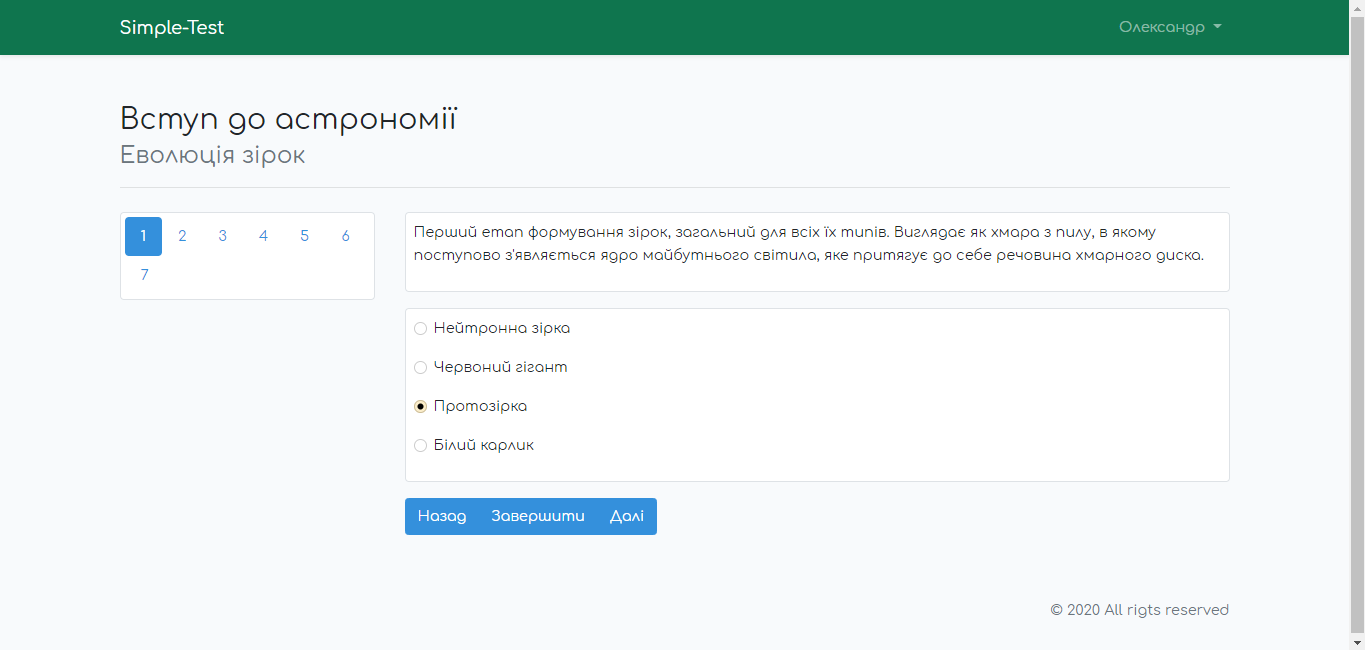


Рисунок 3.12 – Сторінка тестування

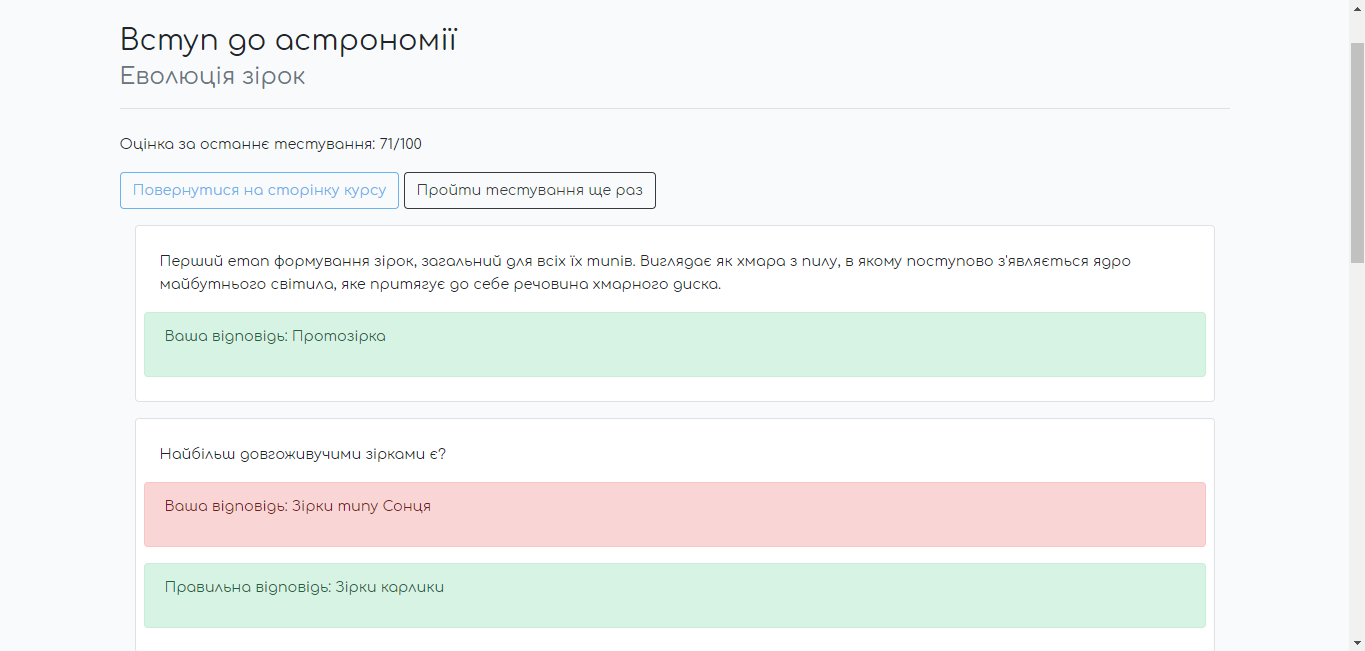


Рисунок 3.13 – Результати останнього тестування

**Висновки до розділу**

У даному розділі було розроблено програмне забезпечення додатку, описано його архітектуру та клієнтський інтерфейс. Для зручності, інтерфейс був описаний у вигляді скріншотів з програми і коментарів до кожного з них.

**ВИСНОВКИ**

Метою даного курсового проекту було створення веб-сайту для проходження онлайн тестування. Користувачі можуть проходити тести з існуючих курсів або створювати власні. Для даної мети був розроблений простий і зрозумілий клієнтський інтерфейс.

На початку роботи було виконано аналіз предметної області, проаналізовано опис вхідних і вихідних даних.

На наступному етапі, була опрацьована інформація про предметну область та використана для побудови концептуальної, логічної та фізичної моделей бази даних. Для реалізації проекту був використаний PHP веб-фреймворк Laravel та реляційна система керування базами даних MySQL.

В результаті виконання всіх етапів роботи, було отримано повноцінний додаток, який виконує поставлену задачу, а також відповідає усім вимогам.

**СПИСОК ЛІТЕРАТУРИ**

1. Історія розвитку баз даних. 2014. URL:

<https://studopedia.su/11_33898_Istoriya-rozvitku-baz-danih.html>.

2. Моделювання бази даних. URL: https://mykonspekts.ru/1-45886.html.

3. Bell C. Introducing the MySQL 8 Document Store. New York: Apress, 2018. 556 p.

4. Kofler M. MySQL. New York: Apress, 2001. 659 p.

5. Krogh J. W. MySQL 8 Query Performance Tuning. New York: Apress, 2020. 965 p.

6. MacIntyre P., Tatroe K. Programming PHP. Creating Dynamic Web Pages   
(4 ed). O’Reilly Media, 2020. 897 p.

7. Morrison M., Beighley L. Head First PHP & MySQL: A Brain-Friendly Guide. O'Reilly Media, 2009. 814 p.

**Додаток А**

Код розробленого додатку

app.blade.php

<!doctype html>

<html lang="{{ str\_replace('\_', '-', app()->getLocale()) }}">

<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <meta name="csrf-token" content="{{ csrf\_token() }}">

    <title>@yield('title', 'Simple-Test')</title>

    <link href="https://fonts.googleapis.com/css2?family=Comfortaa:wght@400;700&display=swap" rel="stylesheet">

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">

    <link href="{{ asset('css/app.css') }}" rel="stylesheet">

</head>

<body>

    <div id="app">

        <header class="navbar navbar-expand-md navbar-dark bg-nav shadow-sm">

            <div class="container">

                <a class="navbar-brand" href="{{ route('home') }}">

                    Simple-Test

                </a>

                <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false">

                    <span class="navbar-toggler-icon"></span>

                </button>

                <div class="collapse navbar-collapse" id="navbarSupportedContent">

                    <ul class="navbar-nav ml-auto">

                        @guest

                            <li class="nav-item">

                                <a class="nav-link" href="{{ route('login') }}">Вхід</a>

                            </li>

                            @if (Route::has('register'))

                                <li class="nav-item">

                                    <a class="nav-link" href="{{ route('register') }}">Реєстрація</a>

                                </li>

                            @endif

                        @else

                            <li class="nav-item dropdown">

                                <a id="navbarDropdown" class="nav-link dropdown-toggle" href="#" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false" v-pre>

                                    {{ Auth::user()->name }} <span class="caret"></span>

                                </a>

                                <div class="dropdown-menu dropdown-menu-right" aria-labelledby="navbarDropdown">

                                    <a class="dropdown-item" href="{{ route('profile') }}">

                                       Профіль

                                    </a>

                                    <a class="dropdown-item" href="{{ route('userCourses') }}">

                                       Мої курси

                                    </a>

                                    <a class="dropdown-item" href="{{ route('createCourse') }}">

                                        Новий курс

                                    </a>

                                    <div class="dropdown-divider"></div>

                                    <a class="dropdown-item" href="{{ route('logout') }}"

                                       onclick="event.preventDefault();

                                                     document.getElementById('logout-form').submit();">

                                        Вихід

                                    </a>

                                    <form id="logout-form" action="{{ route('logout') }}" method="POST" style="display: none;">

                                        @csrf

                                    </form>

                                </div>

                            </li>

                        @endguest

                    </ul>

                </div>

            </div>

        </header>

        <main>

            @yield('content')

        </main>

        <footer class="footer py-3">

            <div class="container d-flex justify-content-end">

                <p class="text-muted">&copy; 2020 All rigts reserved</p>

            </div>

        </footer>

    </div>

    <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js" integrity="sha384-J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJoZ+n" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo" crossorigin="anonymous"></script>

    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js" integrity="sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6" crossorigin="anonymous"></script>

    <script src="{{ asset('js/closeAlert.js') }}"></script>

    @stack('scripts')

</body>

</html>

register.blade.php

@extends('layouts.app')

@section('title')Реєстрація@endsection

@section('content')

<div class="container my-5">

    <div class="row justify-content-center">

        <div class="col-md-8">

            <div class="card">

                <div class="card-header">Реєстрація</div>

                <div class="card-body">

                    <form method="POST" action="{{ route('register') }}">

                        @csrf

                        <div class="form-group row">

                            <label for="name" class="col-md-4 col-form-label text-md-right">Ім'я</label>

                            <div class="col-md-6">

                                <input id="name" type="text" class="form-control @error('name') is-invalid @enderror" name="name" value="{{ old('name') }}" required autocomplete="name" autofocus>

                                @error('name')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="form-group row">

                            <label for="surname" class="col-md-4 col-form-label text-md-right">Прізвище</label>

                            <div class="col-md-6">

                                <input id="surname" type="text" class="form-control @error('surname') is-invalid @enderror" name="surname" value="{{ old('surname') }}" required autocomplete="surname" autofocus>

                                @error('surname')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="form-group row">

                            <label for="email" class="col-md-4 col-form-label text-md-right">Email</label>

                            <div class="col-md-6">

                                <input id="email" type="email" class="form-control @error('email') is-invalid @enderror" name="email" value="{{ old('email') }}" required autocomplete="email">

                                @error('email')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="form-group row">

                            <label for="password" class="col-md-4 col-form-label text-md-right">Пароль</label>

                            <div class="col-md-6">

                                <input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" required autocomplete="new-password">

                                @error('password')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="form-group row">

                            <label for="password-confirm" class="col-md-4 col-form-label text-md-right">Підтвердіть пароль</label>

                            <div class="col-md-6">

                                <input id="password-confirm" type="password" class="form-control" name="password\_confirmation" required autocomplete="new-password">

                            </div>

                        </div>

                        <div class="form-group row mb-0">

                            <div class="col-md-6 offset-md-4">

                                <button type="submit" class="btn btn-primary">

                                    Реєстрація

                                </button>

                            </div>

                        </div>

                    </form>

                </div>

            </div>

        </div>

    </div>

</div>

@endsection

login.blade.php

@extends('layouts.app')

@section('title')Вхід@endsection

@section('content')

<div class="container my-5">

    <div class="row justify-content-center">

        <div class="col-md-8">

            <div class="card">

                <div class="card-header">Вхід</div>

                <div class="card-body">

                    <form method="POST" action="{{ route('login') }}">

                        @csrf

                        <div class="form-group row">

                            <label for="email" class="col-md-4 col-form-label text-md-right">Email</label>

                            <div class="col-md-6">

                                <input id="email" type="email" class="form-control @error('email') is-invalid @enderror" name="email" value="{{ old('email') }}" required autocomplete="email" autofocus>

                                @error('email')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="form-group row">

                            <label for="password" class="col-md-4 col-form-label text-md-right">Пароль</label>

                            <div class="col-md-6">

                                <input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" required autocomplete="current-password">

                                @error('password')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="form-group row">

                            <div class="col-md-6 offset-md-4">

                                <div class="form-check">

                                    <input class="form-check-input" type="checkbox" name="remember" id="remember" {{ old('remember') ? 'checked' : '' }}>

                                    <label class="form-check-label" for="remember">

                                        Запам'ятати мене

                                    </label>

                                </div>

                            </div>

                        </div>

                        <div class="form-group row mb-0">

                            <div class="col-md-8 offset-md-4">

                                <button type="submit" class="btn btn-primary">

                                    Увійти

                                </button>

                                @if (Route::has('password.request'))

                                    <a class="btn btn-link" href="{{ route('password.request') }}">

                                        Забули пароль?

                                    </a>

                                @endif

                            </div>

                        </div>

                    </form>

                </div>

            </div>

        </div>

    </div>

</div>

@endsection

home.blade.php

@extends('layouts.app')

@section('content')

<div class="container my-5">

    <div class="row row-cols-1 row-cols-sm-2 row-cols-md-3 row-cols-lg-4">

        @foreach($courses as $course)

        @if($course->is\_published)

        <div class="col mb-4">

            <div class="card h-100">

                <div class="card-body">

                    <h5 class="card-title">{{ $course->name }}</h5>

                    <h6 class="card-subtitle mb-2 text-muted">{{ $course->subject }}</h6>

                    <p class="card-text"><small class="text-muted">Автор: {{ $course->author->name }} {{ $course->author->surname }}</small></p>

                    <a class="stretched-link" href="{{ route('courseInfo', $course->id) }}"></a>

                </div>

            </div>

        </div>

        @endif

        @endforeach

    </div>

    {{ $courses->links() }}

</div>

@endsection

profile.blade.php

@extends('layouts.app')

@section('title')Профіль@endsection

@section('content')

<div class="container my-5">

    <h2>Профіль користувача</h2>

    <hr class="mb-4">

    <div class="container">

        <h4 class="">Ім'я: {{ $user->name }}</h4>

        <h4 class="">Прізвище: {{ $user->surname }}</h4>

        <a class="btn btn-primary mb-1" href="{{ route('changeName') }}">Змінити ім'я</a>

        <a class="btn btn-warning mb-1" href="{{ route('password.request') }}">Скинути пароль</a>

        <p class="">Створено курсів: {{ $user->createdCourses->count() }}</p>

        <p class="">Підписок на курси: {{ $user->courses->count() }}</p>

        <a class="btn btn-info mb-1" href="{{ route('userCourses') }}">Перейти до курсів</a>

        <a class="btn btn-primary mb-1" href="{{ route('createCourse') }}">Створити новий курс</a>

    </div>

</div>

@endsection

chandeName.blade.php

@extends('layouts.app')

@section('title')Профіль@endsection

@section('content')

<div class="container my-5">

    <div class="container">

        <form action="#" method="post">

            @csrf

            <div class="form-group row">

                <label for="name" class="col-sm-2 col-form-label">Ім'я:</label>

                <div class="col-sm-10">

                    <input id="name" type="text" class="form-control @error('name') is-invalid @enderror" name="name" value="{{ $user->name }}">

                    @error('name')

                    <span class="invalid-feedback">

                        <strong>{{ $message }}</strong>

                    </span>

                    @enderror

                </div>

            </div>

            <div class="form-group row">

                <label for="surname" class="col-sm-2 col-form-label">Прізвище:</label>

                <div class="col-sm-10">

                    <input id="surname" type="text" class="form-control @error('surname') is-invalid @enderror" name="surname" value="{{ $user->surname }}">

                    @error('surname')

                    <span class="invalid-feedback">

                        <strong>{{ $message }}</strong>

                    </span>

                    @enderror

                </div>

            </div>

            <button type="submit" class="btn btn-primary">Підтвердити</button>

        </form>

    </div>

</div>

@endsection

userCourses.blade.php

@extends('layouts.app')

@section('title')Мої курси@endsection

@section('content')

<div class="container my-5">

    @error('error')

    <div class="alert alert-danger alert-dismissible fade show" role="alert">

        {{ $message }}

        <button type="button" class="close" data-dismiss="alert">

            <span aria-hidden="true">&times;</span>

         </button>

    </div>

    @endif

    <ul class="nav nav-tabs" role="tablist">

        <li class="nav-item">

            <a class="nav-link active" data-toggle="tab" href="#subscribes" role="tab">Підписки на курси</a>

        </li>

        <li class="nav-item">

            <a class="nav-link" data-toggle="tab" href="#createdCourses" role="tab">Створені курси</a>

        </li>

    </ul>

    <div class="tab-content">

        <div class="tab-pane fade show active" id="subscribes" role="tabpanel">

            @foreach($subscriptions as $subscription)

            <div class="container my-1 py-3 border rounded">

                <h4>{{ $subscription->name }}</h4><hr>

                <details>

                    <summary>Опис курсу</summary>

                    <p>{{ $subscription->description }}</p>

                </details>

                <p><small class="text-muted">Автор: {{ $subscription->author->name }} {{ $subscription->author->surname }}</small></p>

                <p>Кількість тестів: {{ $subscription->tests->count() }}</p>

                <a href="{{ route('courseInfo', $subscription->id) }}" class="btn btn-secondary mb-1">Перейти</a>

                <a href="{{ route('unsubscribe', $subscription->id) }}" class="btn btn-danger mb-1">Відписатися</a>

            </div>

            @endforeach

        </div>

        <div class="tab-pane fade" id="createdCourses" role="tabpanel">

            @foreach($createdCourses as $course)

            <div class="container my-1 py-3 border rounded">

                <h4>{{ $course->name }}</h4><hr>

                <details>

                    <summary>Опис курсу</summary>

                    <p>{{ $course->description }}</p>

                </details>

                <details class="mb-2">

                    <summary class="mb-2">Кількість тестів: {{ $course->tests->count() }}</summary>

                    @foreach($course->tests as $test)

                    <p><a href="{{ route('editTest', $test->id) }}">{{ $test->name }}</a></p>

                    @endforeach

                </details>

                @if($course->is\_published)

                <a href="{{ route('unpublishCourse', $course->id) }}" class="btn btn-primary mb-1">Зняти з публікації</a>

                @else

                <a href="{{ route('publishCourse', $course->id) }}" class="btn btn-primary mb-1">Публікувати</a>

                @endif

                <a href="{{ route('courseInfo', $course->id) }}" class="btn btn-info mb-1">Перейти</a>

                <a href="{{ route('updateCourse', $course->id) }}" class="btn btn-secondary mb-1">Редагувати</a>

                <a href="{{ route('deleteCourse', $course->id) }}" class="btn btn-danger mb-1">Видалити</a>

            </div>

            @endforeach

        </div>

    </div>

    <div class="tab-pane fade show active" id="subscribes" role="tabpanel" aria-labelledby="subscribes-tab"></div>

</div>

@endsection

createCourse.blade.php

@extends('layouts.app')

@section('title')Створити курс@endsection

@section('content')

<div class="container my-5">

    <form method="POST" action="{{ route('saveCourse') }}">

        @csrf

        <div class="form-group">

            <label for="name">Назва курсу:</label>

            <input type="text" class="form-control @error('name') is-invalid @enderror" id="name" name="name" placeholder="Введіть назву курсу">

            @error('name')

            <span class="invalid-feedback">{{ $message }}</span>

            @enderror

        </div>

        <div class="form-group">

            <label for="subject">Предмет курсу:</label>

            <select class="form-control" id="subject" name="subject">

            @foreach($subjects as $subj)

            <option>{{ $subj }}</option>

            @endforeach

            </select>

        </div>

        <div class="form-group">

            <label for="description">Опис курсу:</label>

            <textarea class="form-control" id="description" name="description" rows="3" placeholder="Введіть опис курсу"></textarea>

        </div>

        <button type="submit" class="btn btn-primary">Створити курс</button>

    </form>

</div>

@endsection

updateCourse.blade.php

@extends('layouts.app')

@section('title')Редагування курсу@endsection

@section('content')

<div class="container my-5">

    <h2>{{ $course->name }}</h2>

    <hr class="mb-4">

    @error('test')

    <div class="alert alert-danger alert-dismissible fade show" role="alert">

        {{ $message }}

        <button type="button" class="close" data-dismiss="alert">

            <span aria-hidden="true">&times;</span>

         </button>

    </div>

    @enderror

    <form method="POST" action="{{ route('saveUpdatedCourse', $course->id) }}">

        @csrf

        <div class="form-group">

            <label for="name">Назва курсу:</label>

            <input type="text" class="form-control @error('name') is-invalid @enderror" id="name" name="name" value="{{ $course->name }}">

            @error('name')

            <span class="invalid-feedback">

                <strong>{{ $message }}</strong>

            </span>

            @enderror

        </div>

        <div class="form-group">

            <label for="description">Опис курсу:</label>

            <textarea class="form-control" id="description" name="description" rows="3">{{ $course->description }}</textarea>

        </div>

        <div class="form-group" id="tests-form">

        @error('test.\*')

        <div class="alert alert-danger alert-dismissible fade show" role="alert">

            {{ $message }}

            <button type="button" class="close" data-dismiss="alert">

                <span aria-hidden="true">&times;</span>

            </button>

        </div>

        @enderror

            <button type="button" class="btn btn-outline-secondary" onclick="addTest()">Додати тест</button>

            @foreach($course->tests as $test)

            <div class="container my-3 input-group" id="{{ $test->id }}">

                <input type="text" class="form-control" id="test-{{ $test->id }}" name="test[{{ $test->id }}]" value="{{ $test->name }}">

                <div class="input-group-append">

                    <button type="button" class="btn btn-outline-secondary" data-test-id="{{$test->id}}" onclick="removeTest(this)">Видалити тест</button>

                </div>

            </div>

            @endforeach

        </div>

        <button type="submit" class="btn btn-primary">Зберегти зміни</button>

    </form>

</div>

@endsection

@push('scripts')

<script src="{{ asset('js/testInputs.js') }}"></script>

@endpush

editTest.blade.php

@extends('layouts.app')

@section('title')Редагування тесту {{ $test->name }}@endsection

@section('content')

<div class="container my-5">

    <h2>{{ $test->course->name }}<br>

        <small class="text-muted">{{ $test->name }}</small>

    </h2>

    <hr class="mb-4">

    @if($errors->any())

    <div class="alert alert-danger alert-dismissible fade show" role="alert">

        @foreach($errors->all() as $error)

        <p>{{ $error }}</p>

        @endforeach

        <button type="button" class="close" data-dismiss="alert">

            <span aria-hidden="true">&times;</span>

        </button>

    </div>

    @endif

    <div id="warning"></div>

    <button type="button" class="btn btn-outline-dark mb-3" id="btn" data-route="{{ route('saveQuestion', $test->id) }}" onclick="add(this)">Додати запитання</button>

    <div class="row">

        <div class="col-md-3">

            <div class="nav flex-column nav-pills border rounded p-1 mb-3 bg-white" id="v-pills-tab" role="tablist">

                @foreach($test->questions as $question)

                <a class="nav-link" id="question-tab-{{ $question->id }}" data-toggle="pill" href="#question-{{ $question->id }}" role="tab">Питання №{{ $loop->iteration }}</a>

                @endforeach

            </div>

        </div>

        <div class="col-md-9">

            <div class="tab-content" id="v-pills-tabContent">

                @foreach($test->questions as $question)

                <div class="tab-pane fade" id="question-{{ $question->id }}" role="tabpanel">

                    <form method="POST" action="{{ route('updateQuestion', [$test->id, $question->id]) }}">

                        @csrf

                        <div class="form-group">

                            <label for="text-{{ $question->id }}">Текст запитання:</label>

                            <textarea class="form-control" name="text" id="text-{{ $question->id }}" rows="3">{{ $question->text }}</textarea>

                        </div>

                        <div class="form-row">

                            <div class="form-group col-sm-6 col-md-7 col-lg-8">

                                <label for="type-{{ $question->id }}">Тип запитання:</label>

                                <select class="form-control" name="type" id="type-{{ $question->id }}" data-id="{{ $question->id }}" onchange="typeSelect(this)">

                                    @foreach($questionTypes as $key => $value)

                                    <option value="{{ $key }}" @if($question->type == $key) selected @endif>{{ $value }}</option>

                                    @endforeach

                                </select>

                            </div>

                            <div class="form-group col-sm-6 col-md-5 col-lg-4">

                                <label for="answersCount-{{ $question->id }}">Кількість відповідей:</label>

                                <select class="form-control" id="answersCount-{{ $question->id }}" data-id="{{ $question->id }}" onchange="answersSelect(this)" @if($question->type == 'writeAnswer') disabled @endif>

                                    @for($i = 2; $i < 7; $i++)

                                    <option value="{{ $i }}" @if($question->answers->count() == $i) selected @endif>{{ $i }}</option>

                                    @endfor

                                </select>

                            </div>

                        </div>

                        <div id="answers-block-{{ $question->id }}">

                            <div class="form-group">

                            @if($question->type == 'writeAnswer')

                                <label for="answer-1">Текст відповіді:</label>

                                <input type="text" class="form-control" id="answer-1" name="answers[{{ $question->answers[0]->id }}]" value="{{ $question->answers[0]->text }}">

                            @else

                                @foreach($question->answers as $answer)

                                    <label for="answer-{{ $answer->id }}">Текст відповіді №{{ $loop->iteration }}:</label>

                                    <div class="input-group">

                                        <input type="text" class="form-control" id="answer-{{ $answer->id }}" name="answers[{{ $answer->id }}]" value="{{ $answer->text }}">

                                        <div class="input-group-append">

                                            <div class="input-group-text">

                                                @if($question->type == 'oneAnswer')

                                                    <input type="radio" name="right-answers[]" value="{{ $answer->id }}" @if($answer->is\_correct) checked @endif>

                                                @elseif($question->type == 'fewAnswers')

                                                    <input type="checkbox" name="right-answers[]" value="{{ $answer->id }}" @if($answer->is\_correct) checked @endif>

                                                @endif

                                            </div>

                                        </div>

                                    </div>

                                @endforeach

                            @endif

                            </div>

                        </div>

                        <button type="submit" class="btn btn-outline-primary">Зберегти</button>

                        <a href="{{ route('deleteQuestion', [$test->id, $question->id]) }}" class="btn btn-danger">Видалити запитання</a>

                    </form>

                </div>

                @endforeach

            </div>

        </div>

    </div>

</div>

@endsection

@push('scripts')

<script src="{{ asset('js/questionInputs.js') }}"></script>

@endpush

courseInfo.blade.php

@extends('layouts.app')

@section('title'){{ $course->name }}@endsection

@section('content')

<div class="jumbotron">

    <div class="container">

        <h1 class="display-5">{{ $course->name }}</h1>

        @if($course->created\_by === Auth::user()->id)

            @if($course->is\_published)

            <a href="{{ route('unpublishCourse', $course->id) }}" class="btn btn-primary mb-1">Зняти з публікації</a>

            @else

            <a href="{{ route('publishCourse', $course->id) }}" class="btn btn-primary mb-1">Публікувати</a>

            @endif

        <a href="{{ route('updateCourse', $course->id) }}" class="btn btn-secondary mb-1">Редагувати</a>

        <a href="{{ route('deleteCourse', $course->id) }}" class="btn btn-danger mb-1">Видалити</a>

        @elseif($isSubscribed)

        <a href="#" class="btn btn-secondary">Перейти до курсу</a>

        <a href="{{ route('unsubscribe', $course->id) }}" class="btn btn-danger mb-1">Відписатися</a>

        @else

        <a href="{{ route('subscribe', $course->id) }}" class="btn btn-success mb-1">Підписатися на курс</a>

        @endif

        <p><small class="text-muted">Автор: {{ $course->author->name }} {{ $course->author->surname }}</small></p>

    </div>

</div>

<div class="container my-5">

    @error('error')

    <div class="alert alert-danger alert-dismissible fade show" role="alert">

        {{ $message }}

        <button type="button" class="close" data-dismiss="alert">

            <span aria-hidden="true">&times;</span>

         </button>

    </div>

    @enderror

    <h2>Про цей курс</h2>

    <p>Предмет: {{ $course->subject }}</p>

    <p class="mx-3">{{ $course->description }}</p>

</div>

<div class="container">

    <h4>Тести:</h4>

    <div class="container">

        @foreach($course->tests as $test)

        <details>

            <summary>{{ $test->name }}</summary>

            <div class="container mb-3">

                @if($course->is\_published || $course->created\_by === Auth::user()->id)

                <p>Запитань в тесті: {{ $test->questions->count() }}</p>

                <a href="{{ route('testingResults', $test->id) }}" class="btn btn-outline-success mb-1">Переглянути результати</a>

                <a href="{{ route('testing', $test->id) }}" class="btn btn-outline-dark mb-1">Почати тестування</a>

                @else

                <p>Курс знаходиться на редагуванні.</p>

                @endif

            </div>

        </details>

        @endforeach

    </div>

</div>

@endsection

testing.blade.php

@extends('layouts.app')

@section('title')Тест {{ $test->name }}@endsection

@section('content')

<div class="container my-5">

    <h2>{{ $test->course->name }}<br>

        <small class="text-muted">{{ $test->name }}</small>

    </h2>

    <hr class="mb-4">

    <form action="{{ route('testingEvaluation', $test->id) }}" method="post">

        @csrf

        <div class="row">

            <div class="col-md-3">

                <div class="nav d-flex flex-wrap align-content-start nav-pills border rounded p-1 mb-3 bg-white" id="pills-tab" role="tablist">

                    @foreach($test->questions as $question)

                    <a class=@if($loop->first)"nav-link active"@else"nav-link"@endif id="question-tab-{{ $question->id }}" data-toggle="pill" href="#question-{{ $question->id }}" role="tab">{{ $loop->iteration }}</a>

                    @endforeach

                </div>

            </div>

            <div class="col-md-9">

                <div class="tab-content" id="pills-tabContent">

                    @foreach($test->questions as $question)

                    <div class=@if($loop->first)"tab-pane fade active show"@else"tab-pane fade"@endif id="question-{{ $question->id }}" role="tabpanel">

                        <div class="border rounded bg-white p-2">

                            <p>{{ $question->text }}</p>

                        </div>

                        <div class="form-group my-3 bg-white p-2 rounded border">

                            @if($question->type === 'writeAnswer')

                            <label for="answer-{{ $question->answers[0]->id }}">Ваша відповідь:</label>

                            <input type="text" class="form-control bg-light" name="answers[{{ $question->id }}]" placeholder="Ведіть відповідь">

                            @else

                            @foreach($question->answers as $answer)

                            <div class="form-check mb-3">

                                @if($question->type === 'oneAnswer')

                                <input class="form-check-input" id="answer-{{ $question->id }}-{{ $answer->id }}" type="radio" name="answers[{{ $question->id }}]" value="{{ $answer->id }}">

                                @elseif($question->type === 'fewAnswers')

                                <input class="form-check-input" id="answer-{{ $question->id }}-{{ $answer->id }}" type="checkbox" name="answers[{{ $question->id }}][]" value="{{ $answer->id }}">

                                @endif

                                <label class="form-check-label" for="answer-{{ $question->id }}-{{ $answer->id }}">{{ $answer->text }}</label>

                            </div>

                            @endforeach

                            @endif

                        </div>

                    </div>

                    @endforeach

                </div>

                <div class="btn-group" role="group">

                    <button type="button" id="previous" class="btn btn-primary">Назад</button>

                    <button type="submit" id="submit" class="btn btn-primary">Завершити</button>

                    <button type="button" id="next" class="btn btn-primary">Далі</button>

                </div>

            </div>

        </div>

    </form>

</div>

@endsection

@push('scripts')

<script src="{{ asset('js/testing.js') }}"></script>

@endpush

testingResults.blade.php

@extends('layouts.app')

@section('title')Тест {{ $test->name }}@endsection

@section('content')

<div class="container my-5">

    <h2>{{ $test->course->name }}<br>

        <small class="text-muted">{{ $test->name }}</small>

    </h2>

    <hr class="mb-4">

    <p>Оцінка за останнє тестування: {{ $result \* 100 }}/100</p>

    <a href="{{ route('courseInfo', $test->course->id) }}" class="btn btn-outline-info">Повернутися на сторінку курсу</a>

    <a href="{{ route('testing', $test->id) }}" class="btn btn-outline-dark">Пройти тестування ще раз</a>

    <div class="container my-3">

        @foreach($test->questions as $question)

        <div class="border rounded bg-white p-2 mb-3">

            <p class="m-3">{{ $question->text }}</p>

            @if($question->type === 'writeAnswer')

            <div class="alert {{ $userAnswers->where('question\_id', $question->id)->pluck('is\_passed')->first() ? 'alert-success' : 'alert-danger' }}">

                <p>Ваша відповідь: {{ $userAnswers->where('question\_id', $question->id)->pluck('written\_answer')->first() ?? 'Ви не вказали відповідь' }}</p>

            </div>

            @if(!$userAnswers->where('question\_id', $question->id)->pluck('is\_passed')->first())

            <div class="alert alert-success">

                <p>Правильна відповідь: {{ $question->answers->where('is\_correct', true)->pluck('text')->first() }}</p>

            </div>

            @endif

            @elseif($question->type === 'oneAnswer')

            <div class="alert {{ $userAnswers->where('question\_id', $question->id)->pluck('is\_passed')->first() ? 'alert-success' : 'alert-danger' }}">

                <p>Ваша відповідь: {{ $userAnswers->where('question\_id', $question->id)->first()->answer->text ?? 'Ви не вказали відповідь' }}</p>

            </div>

            @if(!$userAnswers->where('question\_id', $question->id)->pluck('is\_passed')->first())

            <div class="alert alert-success">

                <p>Правильна відповідь: {{ $question->answers->where('is\_correct', true)->pluck('text')->first() }}</p>

            </div>

            @endif

            @elseif($question->type === 'fewAnswers')

                @foreach($userAnswers->where('question\_id', $question->id) as $val)

                <div class="alert {{ $val->is\_passed ? 'alert-success' : 'alert-danger' }}">

                    <p>Ваша відповідь: {{ $val->answer->text }}</p>

                </div>

                @endforeach

                @if($userAnswers->where('question\_id', $question->id)->pluck('is\_passed')->contains(false))

                <div class="alert alert-success">

                    <p>Правильна відповідь:

                    @foreach($question->answers->where('is\_correct', true) as $val)

                    {{ $val->text.'; ' }}

                    @endforeach

                    </p>

                </div>

                @endif

            @endif

        </div>

        @endforeach

    </div>

</div>

@endsection

@push('scripts')

@endpush

web.php

<?php

use Illuminate\Support\Facades\Route;

use Illuminate\Support\Facades\Auth;

Route::get('/', 'HomeController@index')->name('home');

Auth::routes();

Route::get('/profile', 'ProfileController@index')->name('profile');

Route::get('/profile/changename', 'ProfileController@changeName')->name('changeName');

Route::post('/profile/changename', 'ProfileController@saveName')->name('saveName');

Route::get('/courses', 'CoursesController@userCourses')->name('userCourses');

Route::get('/create-course', 'CoursesController@createCourse')->name('createCourse');

Route::post('/create-course', 'CoursesController@saveCourse')->name('saveCourse');

Route::get('/courses/{id}/update', 'CoursesController@updateCourse')->name('updateCourse');

Route::post('/courses/{id}/update', 'CoursesController@saveUpdatedCourse')->name('saveUpdatedCourse');

Route::get('/courses/{id}/delete', 'CoursesController@deleteCourse')->name('deleteCourse');

Route::get('/course/{id}', 'CoursesController@courseInfo')->name('courseInfo');

Route::get('/course/{id}/subscribe', 'CoursesController@subscribe')->name('subscribe');

Route::get('/courses/{id}/unsubscribe', 'CoursesController@unsubscribe')->name('unsubscribe');

Route::get('/courses/{id}/publish', 'CoursesController@publishCourse')->name('publishCourse');

Route::get('/courses/{id}/unpublish', 'CoursesController@unpublishCourse')->name('unpublishCourse');

Route::get('/courses/test/{id}/edit', 'TestsController@editTest')->name('editTest');

Route::post('/courses/test/{id}/edit', 'TestsController@saveQuestion')->name('saveQuestion');

Route::post('/courses/test/{test\_id}/question/{questions\_id}', 'TestsController@updateQuestion')->name('updateQuestion');

Route::get('/courses/test/{test\_id}/question/{questions\_id}/delete', 'TestsController@deleteQuestion')->name('deleteQuestion');

Route::get('/courses/test/{id}', 'TestingController@index')->name('testing');

Route::post('/courses/test/{id}', 'TestingController@testingEvaluation')->name('testingEvaluation');

Route::get('/courses/test/{id}/results', 'TestingController@testingResults')->name('testingResults');

HomeController.php

<?php

namespace App\Http\Controllers;

use App\Http\Models\Course;

use Illuminate\Http\Request;

class HomeController extends Controller

{

    public function index(){

        $courses = Course::orderByDesc('created\_at')->paginate(20);

        return view('home', ['courses' => $courses]);

    }

}

ProfileController.php

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

class ProfileController extends Controller

{

    /\*\*

     \* Create a new controller instance.

     \*

     \* @return void

     \*/

    public function \_\_construct()

    {

        $this->middleware('auth');

    }

    /\*\*

     \* Show the application dashboard.

     \*

     \* @return \Illuminate\Contracts\Support\Renderable

     \*/

    public function index(){

        $user = Auth::user();

        return view('profile', ['user' => $user]);

    }

    public function changeName(){

        $user = Auth::user();

        return view('changeName', ['user' => $user]);

    }

    public function saveName(Request $req){

        $req->validate(['name' => 'required|min:3|max:255',

                        'surname' => 'required|min:3|max:255'],

                       ['name.\*' => "Ім'я має містити принаймні 3 символи!",

                        'surname.\*' => 'Прізвище має містити принаймні 3 символи!']);

        $user = Auth::user();

        $user->name = $req->input('name');

        $user->surname = $req->input('surname');

        $user->save();

        return redirect()->route('profile');

    }

}

CoursesController.php

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Http\Models\User;

use App\Http\Models\Course;

use App\Http\Models\Test;

use Illuminate\Support\Facades\Auth;

class CoursesController extends Controller

{

    public function \_\_construct()

    {

        $this->middleware('auth');

    }

    public function userCourses(){

        $createdCourses = User::find(Auth::user()->id)->createdCourses->sortByDesc('created\_at');

        $subscriptions = User::find(Auth::user()->id)->courses->sortByDesc('created\_at');

        return view('userCourses', ['createdCourses' => $createdCourses, 'subscriptions' => $subscriptions]);

    }

    public function createCourse(){

        $subjects = ['Астрономія', 'Біологія', 'Інформатика', 'Математика', 'Мови', 'Фізика', 'Хімія' ];

        return view('createCourse', ['subjects' => $subjects]);

    }

    public function saveCourse(Request $req){

        $req->validate(['name' => 'required|min:3|max:255'],

                       ['name.\*' => 'Назва курсу має містити принаймні 3 символи!']);

        $user = User::find(Auth::user()->id);

        $course = new Course();

        $course->name = $req->input('name');

        $course->subject = $req->input('subject');

        $course->description = $req->input('description');

        $user->createdCourses()->save($course);

        return redirect()->route('userCourses');

    }

    public function updateCourse($id){

        $course = Course::find($id);

        return view('updateCourse', ['course' => $course]);

    }

    public function saveUpdatedCourse($id, Request $req){

        $req->validate(['name' => 'required|min:3|max:255',

                        'test' => 'required',

                        'test.\*' => 'required|min:3|max:255'],

                       ['name.\*' => 'Назва курсу має містити принаймні 3 символи!',

                        'test.required' => 'Курс повинен мати принаймні один тест!',

                        'test.\*.\*' => 'Назва тесту має містити принаймні 3 символи!']);

        $course = Course::find($id);

        $course->name = $req->input('name');

        $course->description = $req->input('description');

        $course->save();

        $newTests = [];

        $oldTests = [];

        foreach($req->input('test') as $key => $value){

            if($key < 0){

                $newTests[] = ['name' => $value];

            }

            else{

                $oldTests += [$key => $value];

            }

        }

        $toDelete = array\_diff($course->tests->modelKeys(), array\_keys($oldTests));

        Test::destroy(array\_values($toDelete));

        foreach (Test::find(array\_keys($oldTests)) as $test) {

            $test->name = $oldTests[$test->id];

            $test->save();

        }

        $course->tests()->createMany($newTests);

        return redirect()->route('userCourses');

    }

    public function deleteCourse($id){

        Course::find($id)->delete();

        return redirect()->route('userCourses');

    }

    public function courseInfo($id){

        $course = Course::find($id);

        $isSubscribed = $course->users()->where('user\_id', Auth::user()->id)->exists();

        return view('courseInfo', ['course' => $course, 'isSubscribed' => $isSubscribed]);

    }

    public function subscribe($id){

        $user = User::find(Auth::user()->id);

        $course = Course::find($id);

        $user->courses()->attach($course);

        return redirect()->route('courseInfo', $id);

    }

    public function unsubscribe($id){

        $user = User::find(Auth::user()->id);

        $course = Course::find($id);

        $user->courses()->detach($course);

        return redirect()->back();

    }

    public function publishCourse($id){

        $course = Course::find($id);

        if($course->tests->count() < 1){

            return redirect()->back()->withErrors(['error' => ['Курс повинен мати принаймні один тест!']]);

        }

        foreach ($course->tests as $test) {

            if($test->questions->count() < 5)

                return redirect()->back()->withErrors(['error' => ["Тест $test->name має менше п'яти запитань!"]]);

        }

        $course->is\_published = true;

        $course->save();

        return redirect()->back();

    }

    public function unpublishCourse($id){

        $course = Course::find($id);

        $course->is\_published = false;

        $course->save();

        return redirect()->back();

    }

}

TestsController.php

<?php

namespace App\Http\Controllers;

use App\Http\Models\Question;

use App\Http\Models\Test;

use Illuminate\Http\Request;

class TestsController extends Controller

{

    public function \_\_construct()

    {

        $this->middleware('auth');

    }

    public function editTest($id){

        $test = Test::find($id);

        $questionTypes = [  'oneAnswer' => 'Одна правельна відповідь',

                            'fewAnswers' => 'Декілька правельних відповідей',

                            'writeAnswer' => 'Записати відповідь' ];

        return view('editTest', ['test' => $test, 'questionTypes' => $questionTypes]);

    }

    public function saveQuestion($id, Request $req){

        $req->validate(['text' => 'required|min:10',

                        'answers' => 'required',

                        'answers.\*' => 'required'],

                       ['text.\*' => 'Текст питання має містити принаймні 10 символів!',

                        'answers.\*' => 'Питання має мати відповіді!']);

        if($req->input('type') === 'oneAnswer' || $req->input('type') === 'fewAnswers'){

            $req->validate(['right-answers' => 'required'], ['right-answers.\*' => 'Ви не вказали правильну відповідь!']);

        }

        $test = Test::find($id);

        $question = $test->questions()->create(['text' => $req->input('text'), 'type' => $req->input('type')]);

        foreach($req->input('answers') as $key => $value){

            if($req->input('type') == 'writeAnswer'){

                $question->answers()->create(['text' => $value, 'is\_correct' => true]);

            }

            else{

                if (in\_array($key, $req->input('right-answers'))) {

                    $question->answers()->create(['text' => $value, 'is\_correct' => true]);

                }

                else{

                    $question->answers()->create(['text' => $value, 'is\_correct' => false]);

                }

            }

        }

        return redirect()->route('editTest', $id);

    }

    public function updateQuestion($test\_id, $question\_id, Request $req){

        $req->validate(['text' => 'required|min:10',

                        'answers' => 'required',

                        'answers.\*' => 'required'],

                       ['text.\*' => 'Текст питання має містити принаймні 10 символів!',

                        'answers.\*' => 'Питання має мати відповіді!']);

        if($req->input('type') === 'oneAnswer' || $req->input('type') === 'fewAnswers'){

            $req->validate(['right-answers' => 'required'], ['right-answers.\*' => 'Ви не вказали правильну відповідь!']);

        }

        $question = Question::find($question\_id);

        $question->text = $req->input('text');

        $question->type = $req->input('type');

        $question->save();

        if($req->input('type') != $question->type || count($req->input('answers')) != $question->answers()->count()){

           $question->answers()->delete();

           foreach($req->input('answers') as $key => $value){

                if($req->input('type') == 'writeAnswer'){

                    $question->answers()->create(['text' => $value, 'is\_correct' => true]);

                }

                else{

                    if (in\_array($key, $req->input('right-answers'))) {

                        $question->answers()->create(['text' => $value, 'is\_correct' => true]);

                    }

                    else{

                        $question->answers()->create(['text' => $value, 'is\_correct' => false]);

                    }

                }

            }

        }

        else{

            return 'Dodelat';

        }

        return redirect()->route('editTest', $test\_id);

    }

    public function deleteQuestion($test\_id, $question\_id){

        Question::find($question\_id)->delete();

        return redirect()->route('editTest', $test\_id);

    }

}

TestingController.php

<?php

namespace App\Http\Controllers;

use App\Http\Models\Question;

use App\Http\Models\Test;

use Illuminate\Http\Request;

class TestsController extends Controller

{

    public function \_\_construct()

    {

        $this->middleware('auth');

    }

    public function editTest($id){

        $test = Test::find($id);

        $questionTypes = [  'oneAnswer' => 'Одна правельна відповідь',

                            'fewAnswers' => 'Декілька правельних відповідей',

                            'writeAnswer' => 'Записати відповідь' ];

        return view('editTest', ['test' => $test, 'questionTypes' => $questionTypes]);

    }

    public function saveQuestion($id, Request $req){

        $req->validate(['text' => 'required|min:10',

                        'answers' => 'required',

                        'answers.\*' => 'required'],

                       ['text.\*' => 'Текст питання має містити принаймні 10 символів!',

                        'answers.\*' => 'Питання має мати відповіді!']);

        if($req->input('type') === 'oneAnswer' || $req->input('type') === 'fewAnswers'){

            $req->validate(['right-answers' => 'required'], ['right-answers.\*' => 'Ви не вказали правильну відповідь!']);

        }

        $test = Test::find($id);

        $question = $test->questions()->create(['text' => $req->input('text'), 'type' => $req->input('type')]);

        foreach($req->input('answers') as $key => $value){

            if($req->input('type') == 'writeAnswer'){

                $question->answers()->create(['text' => $value, 'is\_correct' => true]);

            }

            else{

                if (in\_array($key, $req->input('right-answers'))) {

                    $question->answers()->create(['text' => $value, 'is\_correct' => true]);

                }

                else{

                    $question->answers()->create(['text' => $value, 'is\_correct' => false]);

                }

            }

        }

        return redirect()->route('editTest', $id);

    }

    public function updateQuestion($test\_id, $question\_id, Request $req){

        $req->validate(['text' => 'required|min:10',

                        'answers' => 'required',

                        'answers.\*' => 'required'],

                       ['text.\*' => 'Текст питання має містити принаймні 10 символів!',

                        'answers.\*' => 'Питання має мати відповіді!']);

        if($req->input('type') === 'oneAnswer' || $req->input('type') === 'fewAnswers'){

            $req->validate(['right-answers' => 'required'], ['right-answers.\*' => 'Ви не вказали правильну відповідь!']);

        }

        $question = Question::find($question\_id);

        $question->text = $req->input('text');

        $question->type = $req->input('type');

        $question->save();

        if($req->input('type') != $question->type || count($req->input('answers')) != $question->answers()->count()){

           $question->answers()->delete();

           foreach($req->input('answers') as $key => $value){

                if($req->input('type') == 'writeAnswer'){

                    $question->answers()->create(['text' => $value, 'is\_correct' => true]);

                }

                else{

                    if (in\_array($key, $req->input('right-answers'))) {

                        $question->answers()->create(['text' => $value, 'is\_correct' => true]);

                    }

                    else{

                        $question->answers()->create(['text' => $value, 'is\_correct' => false]);

                    }

                }

            }

        }

        else{

            return 'Dodelat';

        }

        return redirect()->route('editTest', $test\_id);

    }

    public function deleteQuestion($test\_id, $question\_id){

        Question::find($question\_id)->delete();

        return redirect()->route('editTest', $test\_id);

    }

}