

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій  
(повне найменування факультету)

Кафедра програмних засобів  
(повне найменування кафедри)

**Пояснювальна записка**

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ  
ОНЛАЙН-ТЕСТУВАННЯ  
DEVELOPMENT OF ONLINE TESTING SOFTWARE

Виконав(ла): студент(ка) 4 курсу, групи КНТ-220  
Спеціальності 122 Комп'ютерні науки  
(код і найменування спеціальності)

Освітня програма (спеціалізація)  
Комп'ютерні науки

СИВАЩЕНКО І.О.  
(ПРИЗВИЩЕ та ініціали)

Керівник ГОЛУБ Т.В.  
(ПРИЗВИЩЕ та ініціали)

Рецензент ЗЕЛЕНЬОВА І.Я.  
(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Факультет КНТ  
Кафедра програмних засобів  
Ступінь вищої освіти бакалавр  
Спеціальність 122 Комп'ютерні науки  
(код і найменування)  
Освітня програма (спеціалізація) Комп'ютерні науки  
(назва освітньої програми (спеціалізації))

**ЗАТВЕРДЖУЮ**  
Завідувач кафедри ПЗ, д.т.н, проф.  
Сергій СУББОТІН  
“ ” 2024 року

**З А В Д А Н Н Я**  
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

СИВАЩЕНКО Іллі Олексійовича  
(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка програмного забезпечення онлайн-тестування.  
Development of Online Testing Software

керівник проєкту (роботи) к.т.н., доцент, ГОЛУБ Тетяна Василівна,  
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 24 ” квітня 2024 року № 168

2. Строк подання студентом проєкту (роботи) 10 червня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Огляд сучасного стану онлайн-тестування. 2. Концепція вебдодатку для онлайн-тестування. 3. Розробка та впровадження вебдодатку. 4. Оцінка ефективності додатку.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)  
Слайди презентації

6. Консультанти розділів проєкту (роботи)

| Розділ              | ПРИЗВИЩЕ, ініціали та посада консультанта | Підпис, дата   |                           |
|---------------------|-------------------------------------------|----------------|---------------------------|
|                     |                                           | завдання видав | прийняв виконане завдання |
| 1-4 Основна частина | ГОЛУБ Т.В. доцент                         |                |                           |
| Нормоконтроль       | КАМІНСЬКА Ж.К., асистент                  |                |                           |

7. Дата видачі завдання “ 15 ” квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломного проєкту (роботи)                             | Строк виконання етапів проєкту ( роботи ) | Примітка     |
|-------|----------------------------------------------------------------------|-------------------------------------------|--------------|
| 1     | Постановка завдання роботи.                                          | 1 тиждень                                 | Завдання, ТЗ |
| 2     | Аналіз предметної області.                                           | 1 тиждень                                 | Розділ 1     |
| 3     | Вибір мови програмування та інших технологій розробки.               | 2 тиждень                                 | Розділ 2     |
| 4     | Розробка архітектури програми.                                       | 2 тиждень                                 | Розділ 3     |
| 5     | Розробка програми.                                                   | 3-4 тижні                                 | Розділ 3, 4  |
| 6     | Тестування та експериментальне дослідження програмного забезпечення. | 5 тиждень                                 | Розділ 4     |
| 7     | Оформлення пояснювальної записки та документів до неї.               | 6 тиждень                                 | Додатки      |
| 8     | Нормоконтроль та рецензування.                                       | 7 тиждень                                 |              |
| 9     | Захист роботи.                                                       | 8 тиждень                                 |              |

Студент(ка)

( підпис )

Ілля СИВАЩЕНКО

(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

( підпис )

Тетяна ГОЛУБ

(Ім'я ПРИЗВИЩЕ)

## РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 73 с., 31 рис., 6 табл., 2 дод., 24 джерела.

ВЕБЗАСТОСУНОК, ОНЛАЙН-ТЕСТУВАННЯ, ОСВІТНІЙ ПРОЦЕС, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, АРХІТЕКТУРА, ТЕХНОЛОГІЇ РОЗРОБКИ, ЕФЕКТИВНІСТЬ.

Об'єкт розробки - процес оцінювання знань за допомогою онлайн технологій.

Предмет розробки - вебзастосунок для онлайн-тестування.

Мета роботи - дослідження та розробка програмного забезпечення для організації та проведення онлайн-тестування в освітньому процесі.

Матеріали, методи та технічні засоби: мови програмування JavaScript, PHP, система управління базами даних MySQL, фреймворки Vue.js, Open Server, вебсервер Apache, технологія AJAX.

Результати. Розроблено вебзастосунок для онлайн-тестування, що забезпечує ефективну організацію та проведення тестувань і опитувань в освітньому процесі. Застосунок має зручний інтерфейс, високу продуктивність та відповідає сучасним вимогам освітньої сфери.

Висновки. Створене програмне рішення є цінним внеском у розвиток онлайн-тестування та має значний потенціал для впровадження в навчальних закладах різного рівня. Вебзастосунок відповідає поставленим вимогам, забезпечує ефективність та зручність організації тестувань і опитувань, сприяючи підвищенню якості освітнього процесу.

Галузь застосування - освітні установи різних рівнів, дистанційне навчання, системи управління навчальним процесом.

## ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 73 pages, 31 figures, 6 tables, 2 appendices, 24 references.

WEB APPLICATION, ONLINE TESTING, EDUCATIONAL PROCESS, SOFTWARE, ARCHITECTURE, DEVELOPMENT TECHNOLOGIES, EFFICIENCY.

The object of development is the process of knowledge assessment using online technologies.

The subject of development is a web application for online testing.

The aim of the study is to research and develop software for organizing and conducting online testing in the educational process.

Materials, methods, and technical means: programming languages JavaScript, PHP, MySQL database management system, Vue.js frameworks, Open Server, Apache web server, AJAX technology.

Results. A web application for online testing has been developed, which ensures effective organization and conduct of tests and surveys in the educational process. The application has a user-friendly interface, high performance, and meets modern educational sector requirements.

Conclusions. The developed software solution is a valuable contribution to the development of online testing and has significant potential for implementation in educational institutions of various levels. The web application meets the set requirements, ensures efficiency and convenience in organizing tests and surveys, contributing to the improvement of the educational process quality.

Field of application - educational institutions of various levels, distance learning, educational process management systems.

## ЗМІСТ

|                                                                        | С. |
|------------------------------------------------------------------------|----|
| Перелк умовних познач .....                                            | 7  |
| Вступ.....                                                             | 8  |
| 1 Огляд сучасного стану онлайн-тестування .....                        | 9  |
| 1.1 Аналіз тенденцій розвитку онлайн-тестування в освітній сфері ..... | 9  |
| 1.2 Переваги та недоліки існуючих вебрішень для онлайн-тестування..    | 11 |
| 1.3 Огляд використання сучасних ІТ-технологій в онлайн-тестуванні...   | 13 |
| 1.4 Висновки за розділом .....                                         | 18 |
| 2 Концепція вебдодатку для онлайн-тестування .....                     | 20 |
| 2.1 Дослідження вимог та потреб користувачів .....                     | 20 |
| 2.2 Розробка архітектури вебзастосунку .....                           | 21 |
| 2.3 Вибір технологій та інструментів розробки.....                     | 23 |
| 2.4 Висновок за розділом.....                                          | 30 |
| 3 Розробка та впровадження вебзастосунку.....                          | 31 |
| 3.1 Реалізація прототипу вебзастосунку.....                            | 31 |
| 3.2 Опис програмного коду .....                                        | 36 |
| 3.3 Рекомендація щодо використання застосунку.....                     | 39 |
| 3.4 Інтерфейс розробленого рішення .....                               | 42 |
| 3.5 Висновок за розділом.....                                          | 45 |
| 4 Оцінка ефективності вебзастосунку.....                               | 47 |
| 4.1 Критерії оцінки ефективності.....                                  | 47 |
| 4.2 Апробація вебзастосунку в реальних умовах .....                    | 48 |
| 4.3 Висновки за розділом .....                                         | 50 |
| Висновки .....                                                         | 52 |
| Перелік джерел посилань .....                                          | 54 |
| Додаток А Текст програми.....                                          | 56 |
| Додаток Б Слайди презентації .....                                     | 66 |

## **ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК**

|       |                                    |
|-------|------------------------------------|
| AJAX  | - Asynchronous JavaScript and XML; |
| DHTML | - Dynamic HTML;                    |
| FAQ   | - Frequently Asked Questions;      |
| HTML  | - Hypertext Markup Language;       |
| HTTP  | - Hypertext Transfer Protocol;     |
| JSON  | - JavaScript Object Notation;      |
| LMS   | - Learning Management System;      |
| MVVM  | - Model-View-ViewModel;            |
| PHP   | - Hypertext Preprocessor;          |
| REST  | - Representational State Transfer; |
| SQL   | - Structured Query Language;       |
| UX/UI | - User Experience/User Interface;  |
| XML   | - eXtensible Markup Language.      |

## ВСТУП

Сьогодні ефективна оцінка знань та навичок студентів є одним із ключових завдань навчальних закладів. Традиційні методи проведення тестувань та екзаменів вже не завжди відповідають сучасним вимогам, оскільки вони можуть бути громіздкими, потребують значних затрат часу і ресурсів. Тому все більшої популярності набувають онлайн-платформи для тестування, які дозволяють оптимізувати та автоматизувати процес оцінювання [1].

Сучасні ІТ-технології відкривають широкі можливості для розробки ефективних рішень в сфері онлайн-тестування. Використання мобільних додатків, хмарних сервісів, технологій штучного інтелекту дозволяє створювати гнучкі, адаптивні та інтерактивні системи оцінювання знань. Такі системи можуть забезпечувати миттєвий зворотній зв'язок, автоматичну перевірку результатів, аналітику успішності, а також персоналізацію навчального процесу [2].

Мета розробки - створення вебзастосунку для онлайн-тестування, який дозволить спростити, оптимізувати та підвищити ефективність процесу оцінювання знань студентів.

Предмет розробки — вебзастосунок для онлайн-тестування. Об'єкт розробки — процес оцінювання знань за допомогою онлайн технологій.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- дослідити сучасний стан та тенденції розвитку сфери онлайн-тестування;
- проаналізувати існуючі програмні рішення для онлайн-тестування, визначити їх переваги та недоліки;
- розробити концепцію вебзастосунку для онлайн-тестування, який враховуватиме потреби користувачів та відповідатиме сучасним вимогам;
- розробити архітектуру та реалізувати прототип вебзастосунку;
- провести тестування та налагодження розробленого рішення.

## 1 ОГЛЯД СУЧАСНОГО СТАНУ ОНЛАЙН-ТЕСТУВАННЯ

### 1.1 Аналіз тенденцій розвитку онлайн-тестування в освітній сфері

Розвиток онлайн-тестування в освітній сфері значно прискорився за останні роки, зокрема завдяки широкому поширенню цифрових технологій та зростаючому попиту на гнучкі і доступні освітні ресурси. Основні тенденції в цій сфері можна простежити через кілька ключових напрямків. Рисунок 1.1 ілюструє тенденцію зростання користувачів онлайн-освіти з 2010 по 2023 рік. На графіку видно стійке збільшення кількості користувачів, що відображає загальний ріст інтересу до онлайн-освіти в цей період [3].

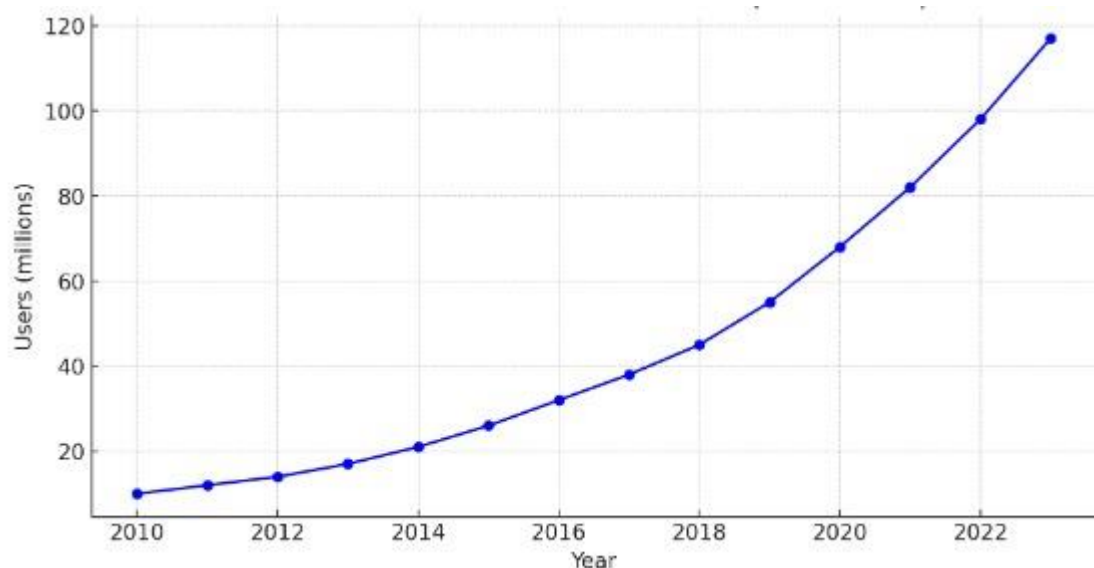


Рисунок 1.1 – Тенденції зростання користувачів онлайн-освіти з 2010 по 2023 рік [1]

Перш за все, важливо відзначити перехід від традиційних паперових форм тестування до цифрових платформ. Це дозволило значно розширити можливості в контексті доступності та масштабованості, давши студентам можливість проходити тести в зручний для них час і місце. Окрім того, інтеграція інтерактивних елементів, таких як мультимедійні матеріали та інтерактивні завдання, зробила процес більш залучаючим та ефективним з точки зору освітнього процесу [1].

Другий важливий аспект — використання адаптивних тестових систем, що змінюють свою складність та тип завдань в залежності від відповідей, які дав студент. Це дозволяє точніше оцінювати знання та вміння учнів на індивідуальному рівні, а також адаптувати освітній процес під конкретні потреби кожного.

Третя тенденція включає в себе зростання використання великих даних та аналітики навчання для вимірювання та аналізу ефективності тестів. Збір і аналіз даних про взаємодію студентів з тестовими матеріалами дозволяє виявляти слабкі місця в навчальних програмах та вносити необхідні корективи, що сприяє покращенню якості освіти [4].

Наостанок, інтеграція з різноманітними навчальними платформами та системами управління навчанням (LMS) дозволяє створювати єдиний освітній простір, де тестування взаємодіє з іншими видами навчальної діяльності, наприклад, з онлайн-курсами та семінарами. Це додатково розширює можливості для підтримки безперервного та комплексного навчання [5].

Таким чином, тенденції розвитку онлайн-тестування в освітній сфері відображають загальний рух до більшої інтеграції цифрових технологій у навчання, зосередження на індивідуальних освітніх потребах студентів та використання даних для покращення освітніх методик (рис. 1.2).

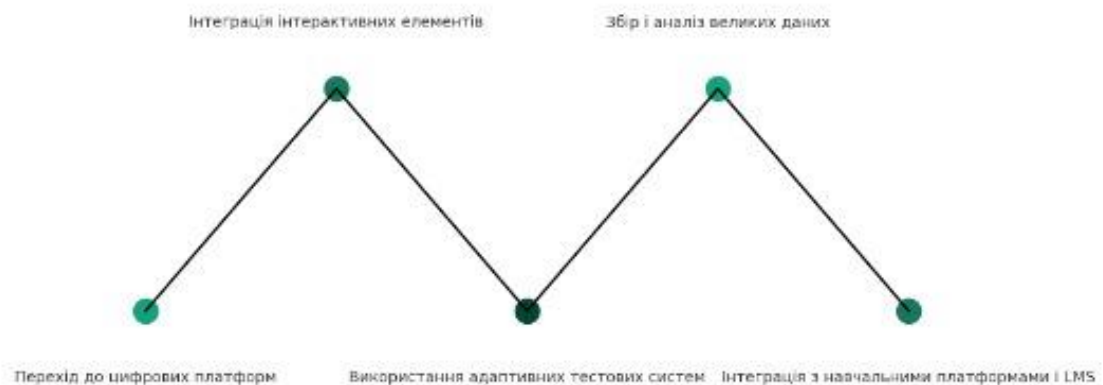


Рисунок 1.2 - Тенденції розвитку онлайн-тестування в освітній сфері

## **1.2 Переваги та недоліки існуючих веб рішень для онлайн-тестування**

Розробка вебсистем онлайн тестування відіграє ключову роль у сучасній освітній індустрії. Вона дозволяє організаціям і освітнім установам значно поліпшити ефективність та доступність навчальних процесів. Завдяки цим системам можливо швидко оцінювати знання великої кількості учнів незалежно від їхнього місцезнаходження. Це особливо актуально в умовах глобалізації та дистанційної освіти, яка стає все більш популярною. Онлайн тестування може здійснюватися у реальному часі, що допомагає викладачам оперативно відстежувати прогрес студентів та виявляти їхні освітні потреби [6].

Однак, існує низка проблем, які потрібно враховувати при розробці таких систем. Наприклад, питання безпеки даних є вкрай важливим, адже системи збирають та зберігають великі обсяги особистої інформації. Також необхідно забезпечити високу надійність та доступність сервісу, щоб уникнути перебоїв під час тестувань. Ще однією проблемою є необхідність інтеграції з іншими освітніми платформами та системами, що може бути технічно складно здійснити [7].

Переваги вебсистем онлайн тестування є значними. По-перше, ці системи забезпечують значну економію часу та ресурсів, оскільки процес оцінювання автоматизовано. По-друге, вони дозволяють проводити об'єктивне та стандартизоване оцінювання, знижуючи людський фактор та можливість помилок. Крім того, такі системи можуть надавати детальну зворотню інформацію учням та викладачам, допомагаючи поліпшити навчальний процес [8].

Актуальність розробки нових вебсистем онлайн тестування обумовлена зростаючим попитом на ефективніші та гнучкіші навчальні інструменти. Сучасний освітній процес потребує інновацій, що відповідають високим технологічним стандартам та вимогам користувачів. Нові системи

повинні бути сумісними з різними пристроями та підтримувати інтеграцію з іншими цифровими ресурсами, що підвищить ефективність навчання та зробить його більш доступним [9].

Причини для розробки нових систем також включають необхідність адаптації до змінюваних умов навчання, як-от збільшення кількості дистанційних курсів та програм. Розробка гнучких та адаптивних систем дозволить забезпечити більш персоналізований підхід до навчання, здатний враховувати індивідуальні потреби студентів. Також це дасть змогу освітнім установам відповідати на виклики сучасності, такі як глобалізація освітнього простору та потреба у постійному оновленні навчальних матеріалів [10].

З огляду на вищезазначене, розробка нових та вдосконалення існуючих вебсистем онлайн тестування є не тільки важливою, але й необхідною для сучасної освіти. Це дозволить не тільки покращити якість та доступність навчання, але й забезпечити його актуальність у динамічно змінюваних технологічних та соціальних умовах. Таблиця 1.1 відображає переваги та недоліки вебсистем онлайн-тестування.

Таблиця 1.1 – Переваги та недоліки онлайн-тестувань

| № | Переваги                                                                                  | Недоліки                                                             |
|---|-------------------------------------------------------------------------------------------|----------------------------------------------------------------------|
| 1 | Економія часу та ресурсів завдяки автоматизації оцінювання.                               | Проблеми з безпекою та приватністю даних.                            |
| 2 | Об'єктивне та стандартизоване оцінювання, зменшення людського фактора.                    | Потреба в надійній інтернет-з'єднанні для стабільної роботи системи. |
| 3 | Можливість надання детальної зворотньої інформації для удосконалення навчального процесу. | Складнощі з інтеграцією різноманітних освітніх платформ та систем.   |

Продовження таблиці 1.1

| № | Переваги                                                          | Недоліки                                                       |
|---|-------------------------------------------------------------------|----------------------------------------------------------------|
| 4 | Гнучкість та доступність навчання, незалежно від місця і часу.    | Можливі технічні збої, що впливають на доступність тестування. |
| 5 | Сумісність з різними цифровими пристроями та освітніми ресурсами. | Висока вартість розробки та підтримки передових технологій.    |

### 1.3 Огляд використання сучасних ІТ-технологій в онлайн-тестуванні

З року в рік зростає популярність і поширення вебресурсів, на базі яких створено багато програмного забезпечення для вирішення широкого кола практичних завдань. Серед таких завдань важливе місце займає організація тестів та опитувань у освітньому процесі [11].

«Наурок» безкоштовний освітній сервіс призначений для створення та проведення тестів онлайн із навчальною метою, ідеально відповідає для оцінювання знань на тематичному та контрольному рівні. Користувальницький інтерфейс доступний українською мовою (рис. 1.3).

Процес реєстрації в системі дуже простий: ви вводите своє ім'я, прізвище, електронну адресу, пароль, підтверджуєте реєстрацію і вказуєте свою роль (викладач або студент), після чого вже можете приступити до створення свого першого тесту.

Для створення тесту клікніть на кнопку «Створити перший тест зараз» або в розділі «Мої тести» виберіть «Створити новий тест». У новому вікні вгорі введіть текст питання і при необхідності додайте додаткові роз'яснення. Далі оберіть тип питання, як-от: одна правильна відповідь, кілька правильних відповідей, слово, номер або співвідношення.

Можна додати більше рядків для варіантів відповідей, натиснувши «Додати відповідь». «Вага питання» вказує на кількість балів, які можна

отримати за правильну відповідь. Біля варіантів відповідей за бажанням можна поставити позначку, яка вказує на правильність відповіді. На завершення натискайте «Готово» і переходьте до наступного питання.

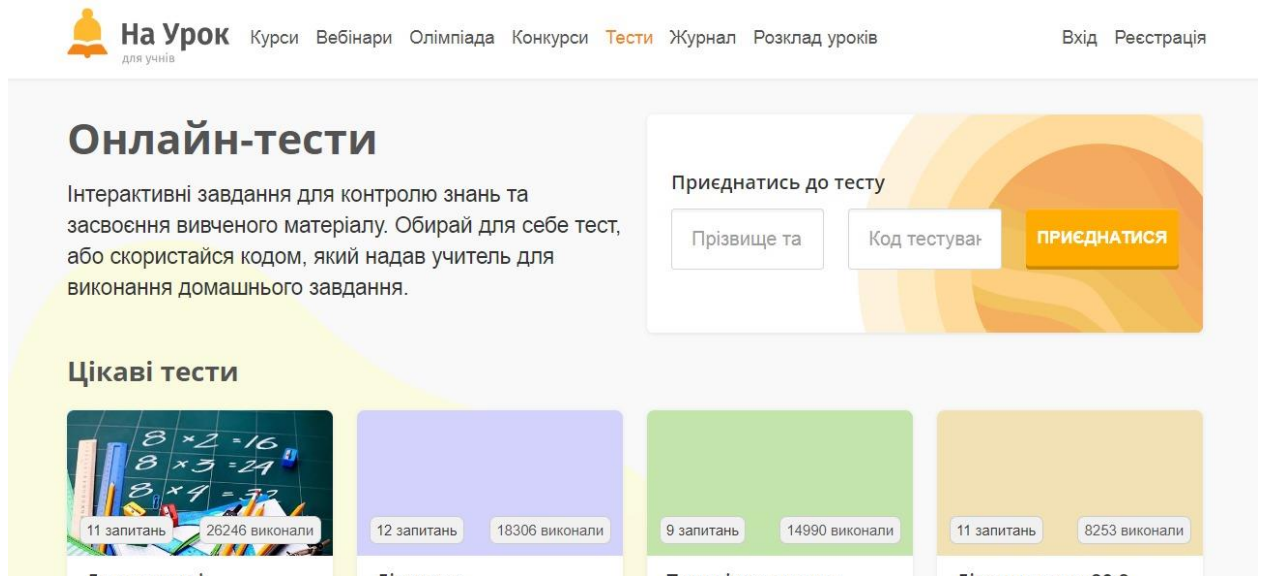


Рисунок 1.3 – «На урок» [12]

LeaningApps.org – безкоштовний освітній сервіс, що допомагає підтримувати навчальний процес через створення та зберігання інтерактивних ігрових вправ. Він розрахований на широку аудиторію, від школярів до дорослих, і використовується для узагальнення та систематизації знань з різних дисциплін (рис. 1.4).

Ці вправи не замінюють повний урок, але вони можуть слугувати елементом під час освітньої діяльності. Сайт особливо привабливий для учнів молодших та середніх класів завдяки ігровому формату завдань, хоча є можливість налаштувати складність завдань для різних освітніх рівнів, від дошкільної до післядипломної освіти [13].

На платформі доступні 18 типів завдань, таких як знаходження пари, кросворди, класифікація та інші, що включають аудіо та відео вміст, а також текстові і графічні задачі.

Предмети охоплюють широкий спектр дисциплін, включаючи мови, природничі науки, соціальні науки та багато інших, з можливістю вибору мови інтерфейсу, включно з українською.

Для початку роботи на сайті вчителі та учні повинні зареєструватись, вказавши свої електронні адреси. Учителі можуть додавати учнів до своїх класів і посилати завдання у вигляді QR-кодів або вставляти посилання на вправи на своїх вебсайтах.

Вчителі також можуть вибирати з різних категорій предметів і типів завдань в розділі "Перегляд вправ". Крім того, платформа надає можливість створювати власні завдання, що дозволяє використовувати різні формати, як звукові, так і візуальні, для кожного типу завдання.

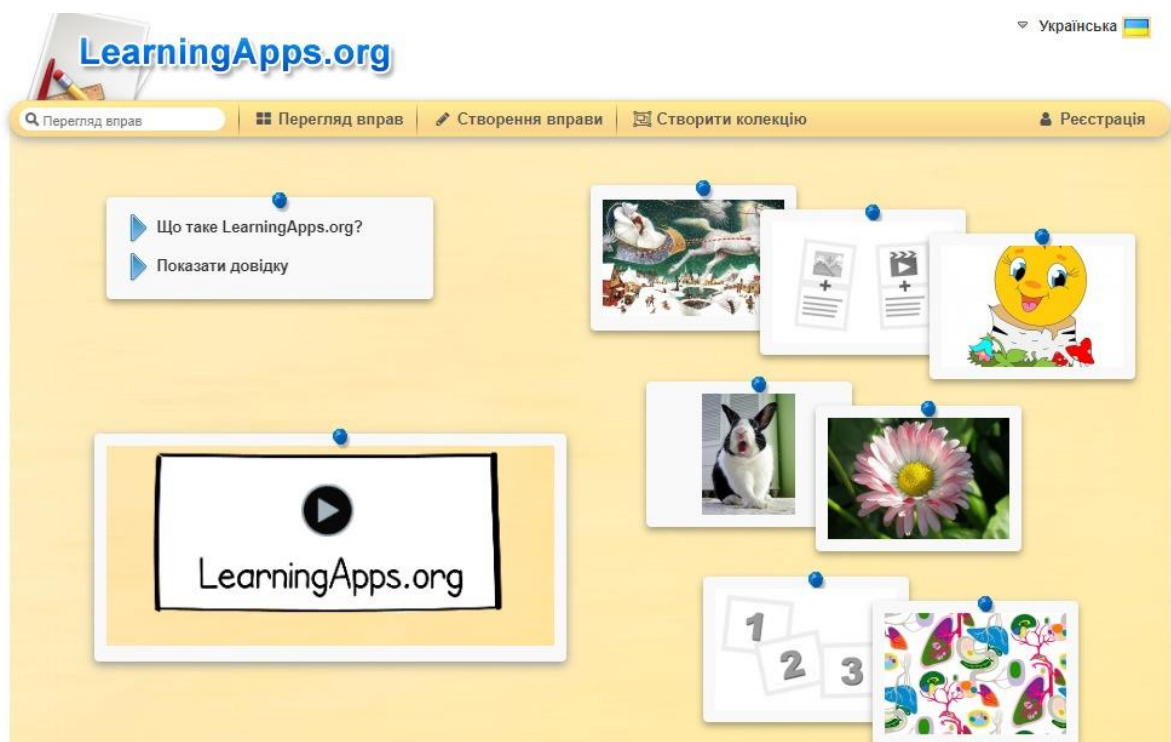


Рисунок 1.4 – LearningApps.org [14]

OnlineTextPad безкоштовний універсальний сервіс дозволяє проводити навчання та тестування онлайн. Вебсайт допомагає у створенні різноманітних навчальних матеріалів і типів завдань, які можна систематизувати за папками. На сайті також доступна велика кількість

завдань з різних шкільних дисциплін, а інтерфейс представлений декількома мовами, включаючи українську (рис. 1.5).

Сервіс включає розширений конструктор тестів, який пропонує численні налаштування для типів питань, результатів, статистичних звітів та оформлення завдань. Типи тестових питань охоплюють 17 варіантів, включаючи вибір однієї або кількох правильних відповідей, вільну форму відповіді, встановлення послідовностей, заповнення прогалін, інтерактивні завдання та багато інших.

Сервіс "Конструктор кросвордів" дозволяє створювати п'ять типів завдань, включаючи класичні та японські кросворди, сканворди, кольорові японські кросворди та філворди.

Платформа також надає інструкції зі створення онлайн опитувань через "Конструктор опитувань".

У секції "Діалогові тренажери" можна створювати розмовні вправи з віртуальними персонажами, які варіюються за складністю і тематикою. Вибір початкових реплік впливає на розвиток діалогу, що є корисним для вивчення мов.

"Комплексні завдання" об'єднують необмежену кількість тестів, кросвордів і логічних ігор, підходячи для домашніх, самостійних та контрольних завдань.

Будь-яке завдання можна опублікувати на сайті для загального доступу, а вчителям надається спеціальний html-код для розміщення завдань на особистих вебресурсах чи блогах.

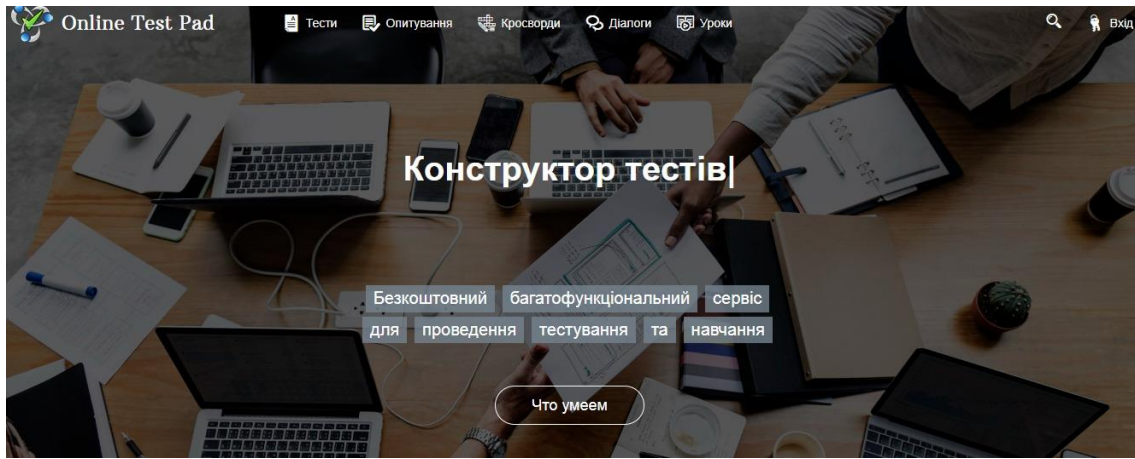


Рисунок 1.5 – OnlineTextPad [15]

Таблиця 1.2 порівняння трьох освітніх сервісів з точки зору розробки та функціоналу ресурсу.

Таблиця 1.2 – Порівняння сучасних сервісів онлайн-тестування

| Критерії                        | Наурок                                   | LeaningApps.org                           | OnlineTextPad                     |
|---------------------------------|------------------------------------------|-------------------------------------------|-----------------------------------|
| Орієнтація на аудиторію         | Школярі молодших та середніх класів      | Широка аудиторія від школярів до дорослих | Для дорослих                      |
| Мовна підтримка                 | Українська, англійська, російська        | Багатомовна підтримка                     | Українська                        |
| Спеціалізація завдань           | Ігрові вправи                            | Різноманітні навчальні матеріали          | Навчальні і тестувальні матеріали |
| Рівні складності                | Від дошкільного до післядипломної освіти | Всі рівні                                 | Стандартні рівні                  |
| Інтеграція з іншими платформами | Обмежена                                 | Так                                       | Обмежена                          |
| Підтримка різних типів завдань  | Ігри, тести, інтерактивні завдання       | Розширені типи тестів, кросворди          | 17 типів тестових питань          |

Продовження таблиці 1.2

| Критерії                                     | Наурок             | LeaningApps.org                             | OnlineTextPad             |
|----------------------------------------------|--------------------|---------------------------------------------|---------------------------|
| Вбудовані інструменти для створення контенту | Є                  | Розширений конструктор тестів та кросвордів | Конструктор тестів        |
| Можливість публікації завдань                | Так                | Так                                         | Так                       |
| Наявність інтерфейсних мов                   | Мультиязычний      | Мультиязычний                               | Українська                |
| Спеціальні функції для навчання              | Розмовні тренажери | Конструктор кросвордів, діалогові тренажери | Немає спеціальних функцій |

Ця таблиця допомагає оцінити особливості кожного сервісу і визначити, який з них найкраще відповідає певним потребам користувачів або освітніх установ.

#### 1.4 Висновки за розділом

Розвиток онлайн-тестування характеризується стійкою тенденцією зростання популярності та застосування. Цьому сприяють такі ключові чинники, як підвищення гнучкості та доступності процесу оцінювання знань, зростання ефективності за рахунок автоматизації, персоналізація навчання на основі аналітики, а також поширення сучасних технологій, зокрема адаптивного тестування та інтеграції з іншими освітніми платформами.

Існуючі вебрішення для онлайн-тестування демонструють значні переваги, зокрема економію часу та ресурсів, об'єктивність оцінювання, можливість надання детального зворотного зв'язку. Водночас, вони стикаються з певними проблемами, такими як забезпечення безпеки даних, технічна надійність, складність інтеграції з іншими системами.

Огляд сучасних ІТ-технологій, що застосовуються в онлайн-тестуванні, свідчить про наявність широкого спектру спеціалізованих сервісів, які пропонують різноманітні функціональні можливості. Ці рішення різняться за спеціалізацією, рівнем складності, підтримкою мов, наявністю вбудованих інструментів для створення контенту, тощо.

Таким чином, аналіз тенденцій розвитку, переваг і недоліків існуючих рішень, а також використання сучасних ІТ-технологій в онлайн-тестуванні, є важливою основою для розробки нових ефективних програмних систем, які б відповідали актуальним потребам освітньої сфери.

## **2 КОНЦЕПЦІЯ ВЕБЗАСТОСУНКУ ДЛЯ ОНЛАЙН-ТЕСТУВАННЯ**

### **2.1 Дослідження вимог та потреб користувачів**

У наш час, завдяки прогресу в технологіях та все більшій доступності Інтернету, освіта переживає революційні зміни. Інтернет став не тільки джерелом необмеженої освітньої інформації, а й активним місцем для інтелектуальної діяльності студентів, які мають можливість постійно здобувати нові знання та удосконалювати свої вміння. Викладачам, у свою чергу, надається змога оптимізувати контроль знань, зокрема, через переведення тестування в онлайн-формат. Цікавість до розробки та застосування освітніх вебресурсів викликана великими очікуваннями щодо підвищення ефективності та якості навчання [16].

Проблеми, які потрібно вирішити, включають:

- забезпечення персональним кабінетом для створення тестів та опитувань;
- швидке здійснення контролю знань;
- автоматизація процесів тестування для дистанційного навчання;
- розробка зручного графічного інтерфейсу для використання на комп'ютерах і мобільних пристроях.

Вхідні дані мають включати інформацію про тести, збережену в базі даних, а результатом роботи мають бути дані про пройдені тести та опитування. Система повинна пропонувати розроблений пошук та зручне представлення інформації, з можливістю перегляду детальних результатів.

Програмна архітектура має складатись з трьох компонентів: клієнт, сервер, та база даних. Основні користувачі системи — адміністратори, викладачі, та студенти. Система повинна надавати повну інформацію про тести та опитування, створені викладачами, і бути швидкою у відображенні в популярних браузерях [17].

## 2.2 Розробка архітектури вебзастосунку

У процесі розгляду завдання та шляхів його реалізації, було прийнято рішення створити програмний комплекс з використанням вебтехнологій. Центальною перевагою такого підходу є можливість використання програми на різноманітних пристроях, завдяки браузерам, які виступають у ролі універсальної платформи [18].

Обрана тришарова архітектура програмного комплексу включає в себе клієнтську частину, сервер і базу даних. Основна обробка даних та бізнес-логіка зосереджені на сервері, який також контролює доступ до програми та виконує ідентифікацію користувачів для забезпечення безпеки даних. Вся взаємодія з додатком для користувачів відбувається через вебінтерфейс, який дозволяє налаштовувати програму та оглядати результати роботи, а також проводити первинну обробку даних перед їх надсиланням на сервер (рис. 2.1).

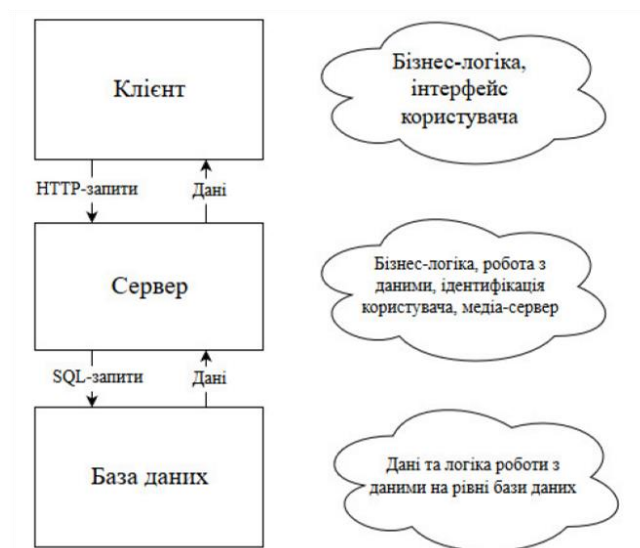


Рисунок 2.1 - Архітектури вебзастосунку

На рівні бази даних головною задачею є зберігання і забезпечення цілісності інформації, яку сервер використовує у подальшій роботі. Деякі

елементи бізнес-логіки також можливо втілити безпосередньо в базі даних, якщо вони не потребують зовнішніх даних.

Сервер був створений з використанням API, що є набором правил і протоколів для будівництва та взаємодії програмного забезпечення. API дозволяє розробникам ефективно створювати програми, використовуючи готові блоки та методи взаємодії. В контексті веброзробки API часто стосується HTTP-запитів та форматів відповідей, які можуть бути структуровані в XML чи JSON. Останнім часом спостерігається тенденція до переходу від традиційних вебсервісів до більш гнучких та ефективних стилів вебресурсів, що працюють за принципом REST або орієнтованої архітектури ресурсів. Також API веб часто дозволяє створювати нові додатки шляхом комбінування функцій декількох інших API, що є основою для так званих гібридних додатків [19].

При створенні клієнтського застосунку використовувався фреймворк, що базується на шаблоні MVVM.

Цей шаблон проектування спрощує процес відділення інтерфейсу користувача від розробки логіки застосунку, званої моделлю. Він передбачає створення моделі представлення, яка трансформує дані для їхнього використання та підтримки. Модель представлення діє подібно до моделі, керуючи процесом відображення даних і може виконувати роль медіатора, контролюючи доступ до логіки застосунку згідно з правилами, що застосовуються на інтерфейсі.

Шаблон MVVM найкраще підходить для платформ з можливістю прив'язки даних, що дозволяє змінам у даних автоматично синхронізуватись з їх відображенням.

Шаблон складається з трьох основних компонентів (рис. 2.2):

- модель (Model) зберігає фундаментальні дані, які потрібні для функціонування застосунку;
- представлення (View) становить графічний інтерфейс, наприклад, вікна та кнопки;

- модель представлення (ViewModel) служить абстракцією представлення, в той же час забезпечуючи зв'язок даних, які повинні бути представлені, з моделлю.

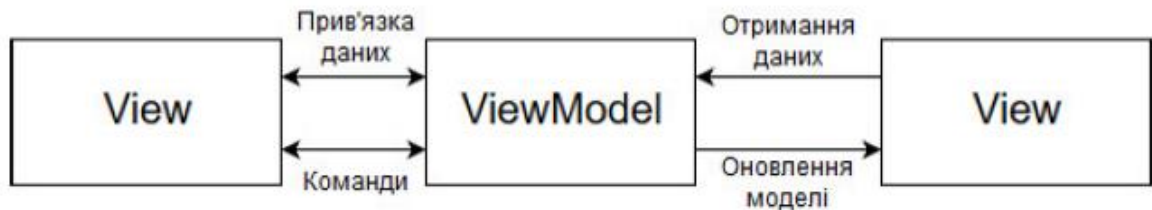


Рисунок 2.2 - Шаблон MVVM

### 2.3 Вибір технологій та інструментів розробки

Програмна система створена з використанням трьохрівневої архітектури, яка включає в себе різні технології на кожному рівні, щоб забезпечити сумісність із різними платформами за допомогою відкритих технологій.

На клієнтському рівні були використані такі технології: програмування на JavaScript, вебфреймворк Vue.js для розробки додатків та Vuetify для створення інтерфейсів [20].

Серверний рівень побудований за допомогою PHP, використання серверної платформи Open Server та вебсервера Apache.

Для рівня бази даних вибрано MySQL. У процесі створення вебсистеми також було застосовано AJAX для асинхронної взаємодії з сервером.

JavaScript - це динамічна, об'єктно-орієнтована мова з прототипною структурою, яка переважно використовується для написання сценаріїв вебсторінок, забезпечення взаємодії з користувачем, управління браузером і динамічної зміни контенту.

JavaScript визначається як мова з динамічною типізацією та підтримує кілька парадигм програмування. Вона включає автоматичне керування пам'яттю, наслідування засноване на прототипах, і функції як елементи першого класу [21].

Застосування JavaScript є різноманітним: від створення інтерактивних вебсторінок та односторінкових застосунків до розробки серверних та мобільних програм, а також інтеграції у програмне забезпечення та PDF-документи.

Незважаючи на зовнішню схожість у назві, JavaScript та Java є різними мовами з різною семантикою, хоча мають схожі стандартні бібліотеки та конвенції іменування. Обидві мови походять від C, але JavaScript також вплинули мови Self та Scheme.

НР - це широко використовувана серверна скриптова мова, яка спеціалізується на створенні динамічного вебконтенту. Вона має велике розповсюдження серед веброзробників та підтримується багатьма хостинг-платформами. РНР є відкритим програмним забезпеченням, доступним для використання [22].

РНР перетворює скрипти на HTML-код на стороні сервера перед їх відправленням клієнту, приховуючи вихідний код від користувача, що сприяє безпеці, однак може обмежити інтерактивність сторінки.

Ця мова включає вбудовані бібліотеки для різноманітних систем управління базами даних та можливість взаємодії з широким спектром БД через стандартні драйвери.

РНР відома завдяки багатому набору вбудованих функцій для веброзробки, включаючи:

- автоматичне отримання POST та GET-параметрів та змінних середовища сервера;
- підключення до численних систем БД, наприклад MySQL, SQLite та інші;
- автоматична обробка HTTP-заголовків;

- менеджмент HTTP-авторизації, кукісів та сесій;
- взаємодія з файлами, сокетом;
- обробка завантажених файлів;
- робота з XForms.

PHP інтерпретує код між відкриваючим (<?) та закриваючим (?>) тегами, що дає змогу легко вбудовувати PHP-код у HTML.

Програмісти знайдуть PHP знайомою, адже її синтаксис позичений з мов C та Perl, знижуючи поріг входу для вивчення PHP. Незважаючи на молодість, PHP широко використовується для створення скриптів в Інтернеті, ставши популярною мовою серед веброзробників.

PHP володіє багатою функціональністю, яка дозволяє ефективно уникати написання складних функцій з нуля, сприяючи швидкій розробці. Як інтерпретована мова, PHP обробляє скрипти достатньо швидко, що робить її підходящою для створення розгорнутих вебзастосунків.

PHP також може використовуватися для створення парсерів, що автоматично аналізують і обробляють вміст вебсторінок, згідно з вказаними завданнями.

Завдяки своїй швидкості та інтерпретованій природі, PHP стала важливим інструментом для створення вебзастосунків, що обслуговують велику кількість користувачів, забезпечуючи високу продуктивність та універсальність для розробників.

Open Server є переносною серверною платформою та програмним середовищем, створеним за рекомендаціями веброзробників. Ця платформа включає комплекс серверного програмного забезпечення та має гнучкий, багатофункціональний інтерфейс, що підтримує адміністрування та налаштування своїх компонентів. Open Server широко використовується для створення, дебагу та тестування вебпроектів, а також надання вебсервісів у локальних мережах [23].

Платформа містить Apache, nginx, PHP, MySQL, phpMyAdmin, FileZilla FTP-сервер. Apache - це вебсервер відкритого коду для різних

операційних систем, який підтримує широкий спектр можливостей через модулі, що розширюють його основні функції. Це включає підтримку різноманітних мов програмування та можливість господарювати декількома сайтами з одного сервера [24].

Apache використовується для передачі статичного та динамічного контенту через HTTP. Як кешувальний проксі-сервер, Apache може покращити продуктивність вебзвернень користувачів у локальній мережі. Його конфігурація може бути налаштована для кешування певних типів файлів або виключення їх з кешу, з визначенням максимального розміру кешу і автоматичним видаленням застарілих об'єктів.

Основні функції Apache, такі як обробка конфігураційних файлів, протокол HTTP і модульна система, розробляються Apache Software Foundation. Хоча Apache може працювати без зовнішніх модулів, їх використання значно розширює можливості сервера. Конфігурація Apache базується на текстових файлах і може включати загальну конфігурацію сервера, віртуального хоста та директив рівня директорії.

Моделі обробки Apache включають `worker`, `pre-fork`, `perchild`, `netware`, `winnt`, `Apache-ITK`, `peruser`, які оптимізовані для різних середовищ і сценаріїв використання, забезпечуючи стабільність та ефективність в обробці запитів від клієнтів.

MySQL — це вільна система управління реляційними базами даних і є компактним багатопоточним сервером баз даних. Вона вирізняється своєю швидкістю, надійністю та простотою управління. У реляційних базах дані розміщуються у роздільних таблицях для оптимізації швидкості та гнучкості, з можливістю поєднання даних із кількох джерел через відносини між таблицями. MySQL використовує SQL, мову структурованих запитів, для взаємодії з базами даних.

MySQL є підходящою системою для невеликих і середніх додатків і підтримується більшістю платформ, але найкраще функціонує в UNIX-

системах з підтримкою багатопоточності, що покращує загальну продуктивність.

Основні можливості MySQL:

- легкість установки і використання;
- безлімітна кількість одночасних користувачів бази даних;
- можливість мати до 50 мільйонів рядків в таблицях;
- висока швидкість обробки команд;
- проста, але ефективна система безпеки.

Як програмне забезпечення з відкритим кодом, MySQL доступна для вільного використання та модифікації. Це забезпечує прозорість і адаптивність до власних потреб користувача.

MySQL складається з серверної частини, яка безперервно працює на хост-машині, та клієнтських програм, які взаємодіють з сервером через мережеві сокети, надсилаючи SQL-запити, щоб отримати або змінити дані. Результати цих запитів сервер потім відправляє назад клієнтським програмам.

Структура MySQL ієрархічна і складається з баз даних, які містять таблиці, а ті, у свою чергу, містять записи. Фізично бази даних та таблиці в MySQL представлені файлами з конкретними розширеннями. Записи організовані в поля різних типів [22].

Технологія AJAX відноситься до методу розробки вебінтерфейсів, що дозволяє виконувати асинхронний обмін даними між браузером і сервером у фоновому режимі. Це сприяє тому, що сторінка не потребує повного перезавантаження для оновлення інформації, що робить вебдодатки більш швидкими та ефективними. AJAX не є самостійною технологією, а скоріше заснований на комбінації декількох технологій. Цей підхід ґрунтується на двох ключових концепціях.

Використання методів для динамічного запиту сервера без повного перезавантаження сторінки, наприклад:

- з використанням XMLHttpRequest (основний компонент);

- через створення фреймів на льоту;
- за допомогою динамічного створення тегу `<script>`.

Застосування DHTML для динамічної модифікації вмісту сторінки. Для передачі даних переважно використовуються формати JSON чи XML.

Переваги:

- зниження обсягу даних, що передаються (використання AJAX може значно зменшити інтернет-трафік при роботі з вебдодатком, оскільки часто достатньо завантажити тільки змінену частину сторінки);
- зниження навантаження на сервери (AJAX знижує навантаження, роблячи процеси більш ефективними);
- прискорення відгуку інтерфейсу (швидке завантаження тільки потрібної частини сторінки прискорює взаємодію).

Недоліки:

- відсутність інтеграції з браузерними інструментами (динамічно створені сторінки не фіксуються в історії браузера, кнопка «Назад» може не працювати);
- проблеми з індексацією динамічного вмісту пошуковими системами;
- зменшення актуальності традиційної статистики вебсайтів;
- ускладнення управління проектами через перерозподіл логіки даних;
- необхідність активного JavaScript у браузері.

Ключовим елементом AJAX є XMLHttpRequest — API для виконання запитів від клієнта до сервера у фоновому режимі, що підтримується мовами JavaScript, JScript, VBScript та іншими, дозволяє обмінюватися даними асинхронно або синхронно в текстових форматах, зокрема XML або JSON [20].

Під час розробки системи було проведено детальне дослідження предметної області та вимог клієнта. На основі глибокого аналізу було вирішено створити програмне рішення, яке функціонує на основі вебтехнологій та призначене для роботи через веббраузер (табл. 2.1).

Таблиця 2.1 – Архітектурний стек

| Рівень архітектури | Технології                         | Опис                                                                                                                              |
|--------------------|------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Клієнтський        | JavaScript, Vue.js, Vuetify        | JavaScript для скриптів на сторінках, Vue.js для розробки інтерактивних інтерфейсів, Vuetify для створення стилів.                |
| Серверний          | PHP, Open Server, Apache           | PHP для динамічного вебконтенту, Open Server як серверна платформа, Apache як вебсервер.                                          |
| База даних         | MySQL                              | Використання для зберігання та управління даними, забезпечує швидкість та надійність.                                             |
| Асинхронна обробка | AJAX (використовує XMLHttpRequest) | Дозволяє асинхронний обмін даними з сервером, забезпечує високу швидкість і ефективність взаємодії без перезавантаження сторінки. |

Серверні технології були підібрані з урахуванням легкості використання, сучасності та сумісності з різними операційними системами. Платформа Open Server, вибрана для цих цілей, забезпечує легке налаштування доступу до вебсерверу. Вебсервер Apache був обраний за його здатність ефективно обслуговувати вебсайти. Програмування на PHP використовувалося для забезпечення взаємодії з базою даних [19].

На клієнтському рівні технології були вибрані за аналогічними критеріями. Фреймворк Vue.js використовувався для створення складних та легко адаптуємих графічних інтерфейсів, а JavaScript був вибраний за його гнучкість та динамічність. Vuetify було обрано для стилізації за його багатий інтерфейс та адаптивність.

Для бази даних вибір припав на MySQL через її відкритий код, високу продуктивність та надійність.

Крім того, в системі було імплементовано технологію AJAX для забезпечення фонових HTTP-запитів між вебклієнтом та сервером, що сприяє ефективності взаємодії.

## **2.4 Висновок за розділом**

Результатом проведеного дослідження стала розробка архітектури вебзастосунок для онлайн-тестування, заснованого на трирівневій структурі. Кожен рівень архітектури був реалізований із застосуванням відповідних сучасних технологій.

Клієнтський рівень: JavaScript, фреймворк Vue.js та бібліотека Vuetify для створення інтерактивного користувацького інтерфейсу. Такий підхід забезпечує високу гнучкість та адаптивність користувацького досвіду.

Серверний рівень: PHP як мова програмування для реалізації серверної логіки, платформа Open Server для організації інфраструктури сервера та вебсервер Apache для ефективної обробки запитів користувачів. Такі технології дозволяють розробити надійну та масштабовану серверну частину системи.

Рівень бази даних: MySQL як система управління реляційними базами даних, яка забезпечує цілісність, надійність та високу продуктивність при збереженні та обробці даних.

Крім того, в архітектурі було застосовано технологію AJAX, що уможливорює асинхронний обмін даними між клієнтом і сервером, підвищуючи ефективність та швидкодію програмного комплексу.

Обраний стек технологій дозволяє створити вебзастосунок для онлайн-тестування, який характеризується гнучкістю, масштабованістю, надійністю та зручністю використання. Такий підхід є оптимальним для вирішення поставлених у роботі завдань.

## 3 РОЗРОБКА ТА ВПРОВАДЖЕННЯ ВЕБЗАСТОСУНКУ

### 3.1 Реалізація прототипу вебзастосунку

Система для організації та проведення тестувань і опитувань у освітньому середовищі буде складена з декількох модулів, кожен з яких поділений на ряд функціональних блоків. Основним компонентом стане користувацький кабінет, який також виступатиме ключовим елементом у структурі системи. Ця організація забезпечить просте та зрозуміле використання системи, оскільки модулі розміщені у логічному порядку, що звично застосовується для розв'язання подібних завдань (рис. 3.1).

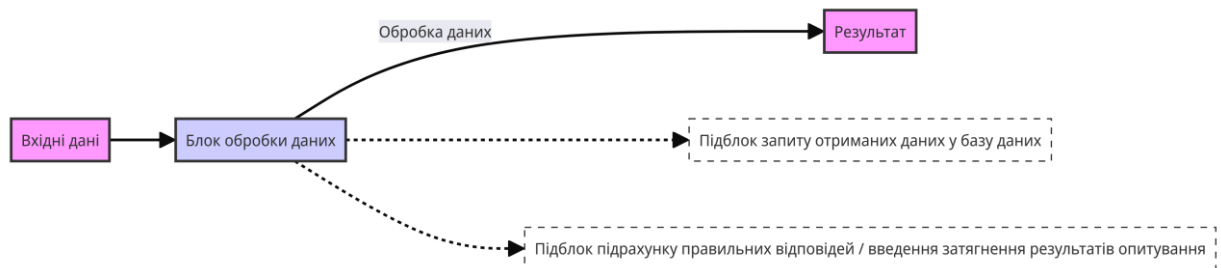


Рисунок 3.1 – Діаграма процесу обробки тестових даних та результатів опитувань

У цій системі користувачі спочатку вводять відповіді на поставлені запитання, які система після цього аналізує. На основі аналізу вона виконує розрахунки і видає кінцеві результати, такі як оцінки за тестування або висновки з опитувань.

Програмний застосунок, розроблений для організації і проведення тестів та опитувань у навчальному середовищі, включає трьох головних учасників, які взаємодіють в рамках системи. Це взаємодія ілюстрована на діаграмі прецедентів, представлений на рисунку 3.2. Користувачі поділені на три групи: студенти, викладачі та адміністратори. Викладачі створюють контент для тестів та опитувань, студенти беруть участь у цих тестуваннях, тоді як адміністратори відповідають за реєстрацію викладачів та студентів і

вилучення застарілих даних. Кожна група має можливість переглядати результати тестів і опитувань, що забезпечує прозорість і доступність інформації.

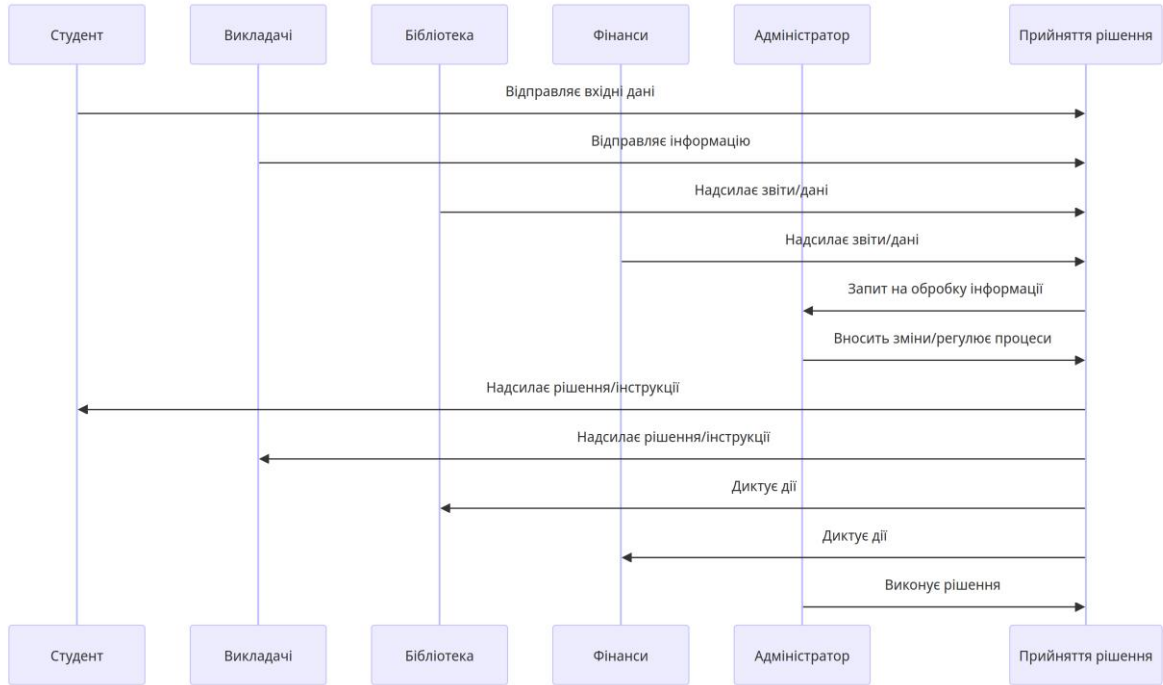


Рисунок 3.2 — Діаграма взаємодії системи

Кабінет користувача слугує головним робочим простором у системі та містить декілька підмодулів, які виконують ключові функції системи і є елементами її структури.

Кабінет користувача системи освітнього процесу оснащений декількома підмодулями, які спрямовані на ефективне управління навчальними активностями. Ці підмодулі включають модуль для управління тестами, який дозволяє викладачам створювати тести, студентам — проходити їх, усім користувачам — переглядати результати та адміністраторам — видаляти застарілі дані про тести. Інший підмодуль, модуль для управління опитуваннями, надає викладачам можливість створення нових опитувань, студентам — участь у них, всім користувачам — доступ до результатів, а адміністраторам — видалення застарілих даних опитувань. Також існує модуль для управління даними викладачів та

студентів, який дозволяє адміністраторам реєструвати нових викладачів та студентів, змінювати логін та пароль їх облікових записів, оновлювати основні дані та видаляти облікові записи викладачів та студентів. Ці модулі спільно сприяють підвищенню продуктивності навчального процесу та забезпечують гладке управління академічною інформацією.

Таблиця 3.1 структуровано представляє основні модулі та їхні функції в системі кабінету користувача.

Таблиця 3.1 – Опис модулів застосунку

| Назва модуля                   | Функції модуля                                                              |
|--------------------------------|-----------------------------------------------------------------------------|
| Модуль управління тестами      | - Створення нових завдань                                                   |
|                                | - Проходження завдань                                                       |
|                                | - Перегляд результатів тестів                                               |
|                                | - Видалення застарілих даних                                                |
| Модуль управління опитуваннями | - Створення нових опитувань (для викладачів)                                |
|                                | - Проходження опитувань (для студентів)                                     |
|                                | - Перегляд результатів опитувань                                            |
|                                | - Видалення застарілих даних                                                |
| Модуль управління даними       | - Реєстрація нових користувачів                                             |
|                                | - Змінювання логіну і пароля до облікового запису                           |
|                                | - Оновлення основних даних викладача або студента (для адміністраторів)     |
|                                | - Видалення облікових записів викладачів та студентів (для адміністраторів) |

На рисунку 3.3 програмний код описує компонент вебінтерфейсу, що використовується для управління тестами в рамках вебзастосунку.

```

8 >Створити тест</v-btn>
9 <v-btn class="success" :to="'/tests/search'">Пройти тест</v-btn>
10 <v-card
11 class="elevation-10 mb2"
12 v-for="(test, i) in tests"
13 :key="i"
14 >
15 <v-layout>
16 <v-flex xs8>
17 <v-card-text>
18 <h2 class="text--primary">{{ test.name }}</h2>
19 <p>Предмет: {{ test.subject }}</p>
20 <p>Автор: {{ test.author }}</p>
21 <p>Кількість питань: {{ test.amountQuestions }}</p>
22 </v-card-text>
23 <v-card-actions>
24 <v-spacer></v-spacer>
25 <v-btn
26 class="info"
27 :to="'/test/' + test.id + '/result'"
28 >Результати</v-btn>
29 </v-card-actions>

```

Рисунок 3.3 - Керування тестами

Структура і функціонал компоненту наступні.

HTML Структура (Template):

- основний контейнер (v-container) містить лейаут (v-layout), який організує вміст у рядок;
- в цьому рядку розміщено один блок (v-flex) з наступними елементами: заголовок (h1) "Тести", що стилізований як вторинний текст; кнопки для створення нового тесту (Створити тест) та проходження тесту (Пройти тест), які ведуть користувача на відповідні сторінки; карточки (v-card) для кожного тесту, що ітеруються через список tests. Кожна карточка відображає: назву тесту, предмет, автора та кількість питань; кнопку "Результати", яка веде на сторінку результатів тесту.

Vue.js функціональність (Script):

- data функція повертає порожній об'єкт, можливо, тут можуть бути додані майбутні дані;
- в розділі computed: tests - це обчислювана властивість, яка отримує список тестів з глобального сховища (\$store). Вона використовується для відображення інформації про тести у карточках.

Цей компонент ефективно використовує Vue.js для динамічного відображення даних про тести, забезпечуючи інтерактивність через кнопки, які перенаправляють користувачів на різні частини застосунку для подальшої взаємодії, наприклад, створення, проходження тестів чи перегляд результатів. Це забезпечує зручний та інтуїтивно зрозумілий інтерфейс для кінцевого користувача.

На рисунку 3.4 програмний код представляє компонент вебзастосунку, який забезпечує відображення інтерфейсу для роботи з опитуваннями.

```

8 >Створити опитування</v-btn>
9 <v-btn class="success">Пройти опитування</v-btn>
10 <v-card
11   class="elevation-10 mb2"
12   v-for="(poll, i) in polls"
13   :key="i"
14 >
15   <v-layout>
16     <v-flex xs8>
17       <v-card-text>
18         <h2 class="text--primary">{{ poll.title }}</h2>
19         <p>Предмет: {{ poll.subject }}</p>
20         <p>Автор: {{ poll.author }}</p>
21         57
22         <p>Кількість питань: {{ poll.amountQuestions }}</p>
23       </v-card-text>
24       <v-card-actions>
25         <v-spacer></v-spacer>
26         <v-btn
27           class="info"
28           :to="'/poll/' + poll.id"
29         >Результати</v-btn>
30       </v-card-actions>

```

Рисунок 3.4 - Компонент вебзастосунку, який забезпечує відображення інтерфейсу для роботи з опитуваннями

Розглянемо його основні частини і функції:

HTML Розмітка (Template):

- всередині `<v-container>` розміщено вертикальну структуру з використанням `<v-layout row>`, що означає розташування елементів у рядок;
- `<v-flex xs12 sm6 offset-sm3>` визначає блок, який займає всі 12 колонок на дуже малих екранах (xs), 6 колонок на невеликих екранах (sm) та має зміщення на 3 колонки на середніх екранах (sm), центруючи контент на сторінці;

- у цьому блоку відображено заголовок "Опитування" і кнопки для створення нового опитування та проходження опитування. Друга кнопка не має зазначеного маршруту (:to), тому її функціонал може залежати від подальшого скриптового розширення;

- `<v-card>` використовується для відображення даних кожного опитування. Дані про опитування ітеруються через `v-for`, що бере кожне опитування зі списку `polls`;

- кожна картка містить назву опитування, предмет, автора і кількість питань. Також є кнопка, що веде на сторінку результатів цього опитування.

Vue.js Скрипт (Script):

- у секції `data()` повертається порожній об'єкт, можливо, тут планується зберігання стану компоненту в майбутньому;

- у `computed` властивості `polls` використовується для отримання списку опитувань з глобального сховища (`this.$store.getters.polls`), яке є частиною Vuex store. Ця властивість динамічно відображає зміни в сховищі і забезпечує актуальність даних на сторінці.

Загалом, код організовує користувацький інтерфейс для управління опитуваннями, дозволяючи користувачам легко створювати, проходити та переглядати результати опитувань. Цей компонент є інтерактивним елементом в рамках більш широкої системи вебдодатків, забезпечуючи зручність і доступність опитувань для різних користувачів.

### 3.2 Опис програмного коду

При розробці вебзастосунку для онлайн-тестування були використані наступні ключові етапи і концепції фреймворку Vue.js:

- налаштування проекту. Для створення проекту використовувався Vue CLI, який значно спростив процес налаштування, надавши готові шаблони і конфігурації. Були встановлені необхідні залежності, такі як Vue

Router для маршрутизації і Vuex для управління станом, за допомогою менеджера пакетів npm;

- створення компонентів: інтерфейс застосунку був розділений на ієрархію незалежних компонентів, таких як сторінка авторизації, сторінка створення тесту, сторінка проходження тесту тощо. Кожен компонент був створений в окремому файлі .vue, який містить розмітку, логіку і стилі. Для динамічного рендерингу і зв'язування даних використовувались директиви Vue, такі як v-if, v-for і v-bind (табл. 3.2);

- маршрутизація: для навігації між сторінками застосунку використовувався Vue Router. Були визначені маршрути для основних сторінок, таких як список тестів, створення тесту, проходження тесту і результати. Динамічні сегменти маршрутів використовувались для передачі параметрів, наприклад, ID тесту для сторінки проходження;

- управління станом: для централізованого управління станом застосунку використовувалась бібліотека Vuex. У сховищі зберігались такі дані, як список тестів, інформація про поточного користувача, результати проходження тестів тощо. Компоненти отримували доступ до стану за допомогою геттерів, а змінювали стан через мутації і дії;

- зв'язок з сервером (API): для взаємодії з серверним API використовувалась бібліотека Axios. Запити виконувались у методах життєвого циклу компонентів, таких як created і mounted, а також у діях Vuex. Наприклад, при завантаженні сторінки зі списком тестів виконувався GET-запит для отримання списку з сервера. При збереженні нового тесту виконувався POST-запит для відправки даних на сервер.

- Стилізація: для стилізації компонентів використовувались звичайні CSS-стили з атрибутом scoped для обмеження області видимості. Також для створення єдиного стилю інтерфейсу використовувався UI-фреймворк Vuetify, який надав набір готових компонентів, таких як кнопки, форми, картки тощо.

– збірка і розгортання: для збірки проекту використовувався Vue CLI з налаштуваннями для продакшена, які включали мінімізацію і оптимізацію коду. Після успішної збірки проект був розгорнутий на вебсервері Apache за допомогою платформи Open Server.

Використання фреймворку Vue.js і пов'язаних з ним інструментів дозволило ефективно розробити інтерактивний односторінковий застосунок для онлайн-тестування. Розділення інтерфейсу на компоненти, використання маршрутизації і централізованого сховища станів зробило код модульним і легко підтримуваним. Інтеграція з серверним API через HTTP-запити дозволила реалізувати повноцінну клієнт-серверну взаємодію. А застосування UI-фреймворку Vuetify забезпечило привабливий і інтуїтивно зрозумілий інтерфейс користувача.

Таблиця 3.2 – Програмні функції

| Модуль                                     | Повна назва функції                                               |
|--------------------------------------------|-------------------------------------------------------------------|
| Створення нових завдань                    | <code>function createNewTask(taskDetails)</code>                  |
| Проходження завдань                        | <code>function completeTask(taskId, userId, answers)</code>       |
| Перегляд результатів тестів                | <code>function viewTestResults(userId)</code>                     |
| Видалення застарілих даних                 | <code>function deleteOldData(dataType)</code>                     |
| Створення нових опитувань (для викладачів) | <code>function createNewSurvey(surveyDetails)</code>              |
| Проходження опитувань (для студентів)      | <code>function completeSurvey(surveyId, userId, responses)</code> |
| Перегляд результатів опитувань             | <code>function viewSurveyResults(userId)</code>                   |
| Видалення застарілих даних опитувань       | <code>function deleteOldSurveyData(dataType)</code>               |
| Реєстрація нових користувачів              | <code>function registerNewUser(userData)</code>                   |

Продовження таблиці 3.2

| Модуль                                              | Повна назва функції                                                  |
|-----------------------------------------------------|----------------------------------------------------------------------|
| Змінювання логіну і пароля до облікового запису     | function<br>changeLoginCredentials(userId,<br>newLogin, newPassword) |
| Оновлення основних даних викладача або студента     | function      updateUserData(userId,<br>updatedData)                 |
| Видалення облікових записів викладачів та студентів | function deleteUserAccount(userId)                                   |

### 3.3 Рекомендація щодо використання застосунку

Алгоритм авторизації і аутентифікації користувачів включає введення логіна і пароля на сторінці авторизації, відправлення даних на сервер для перевірки відповідності з базою, присвоєння унікального токена сесії при успішній авторизації, та збереження токена на клієнтській стороні для використання у наступних запитах (рис. 3.5).

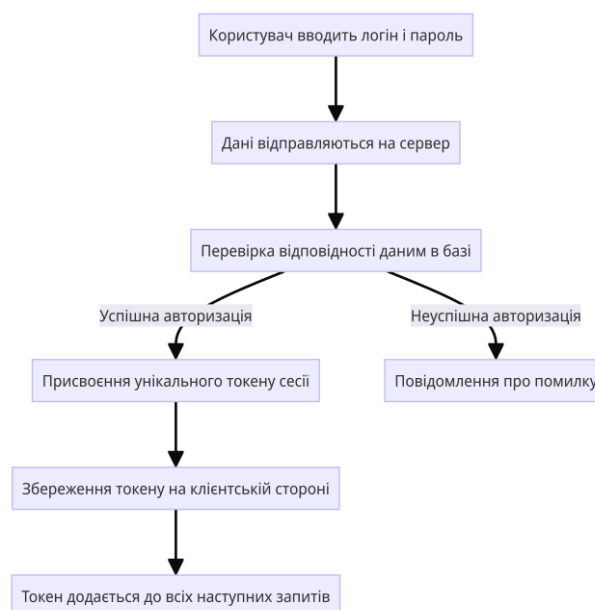


Рисунок 3.5 – Алгоритм авторизації і аутентифікації користувачів

Алгоритм створення нового тесту полягає у заповненні викладачем форми на сторінці створення тесту, відправленні даних на сервер, валідації та збереженні даних у базі, та перенаправленні на сторінку редагування тесту (рис. 3.6).

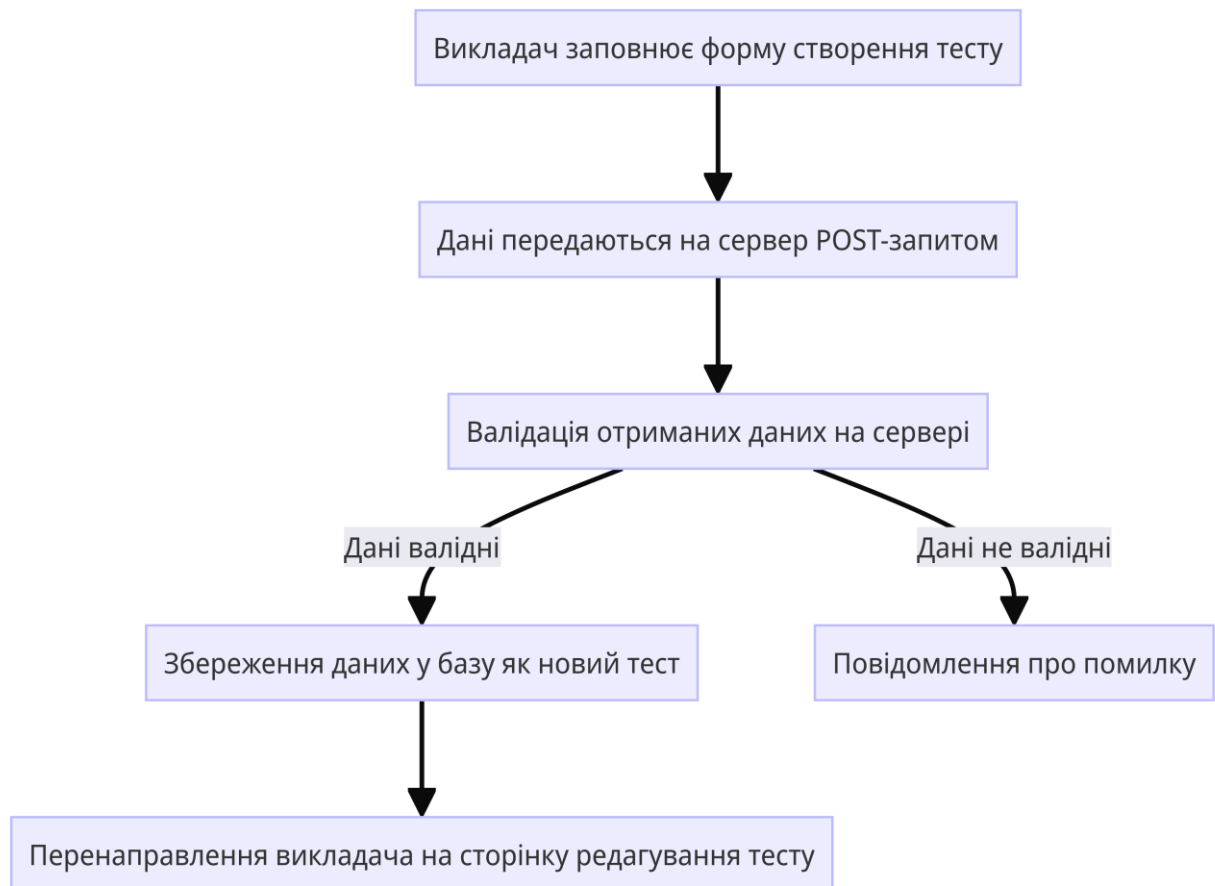


Рисунок 3.6 – Алгоритм створення нового тесту

Алгоритм додавання питань до тесту включає заповнення форми викладачем, відправлення даних на сервер, збереження нового питання в базі, та оновлення сторінки тесту для відображення нового питання (рис. 3.7).

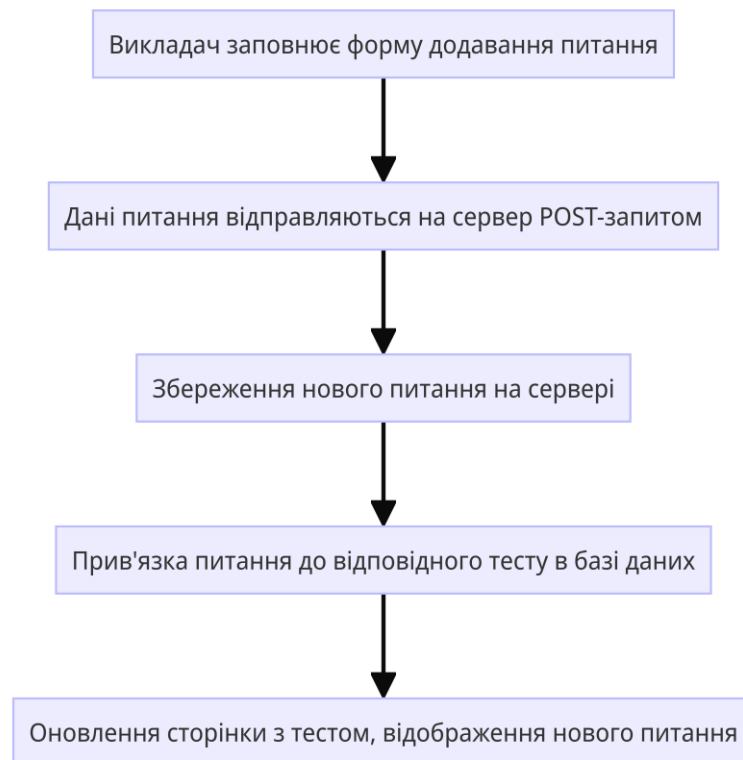


Рисунок 3.7 – Алгоритм додавання питань в тест

Алгоритм проходження тесту студентами включає перегляд питань і варіантів відповідей, збереження вибору в локальному стані, відправлення відповідей на сервер після завершення тесту, перевірку відповідей сервером, обрахування результату, та відображення результату студенту (рис. 3.8).

Розроблений вебзастосунок для онлайн-тестування базується на ключових алгоритмах, які забезпечують ефективне функціонування системи. Алгоритм авторизації і аутентифікації користувачів гарантує безпеку та персоналізований доступ до функцій застосунку. Алгоритми створення тестів і додавання питань дозволяють викладачам гнучко формувати контент для оцінювання знань. Алгоритм проходження тесту студентами забезпечує зручність та об'єктивність процесу тестування. Разом ці алгоритми формують надійну основу для реалізації онлайн-тестування в освітньому процесі.

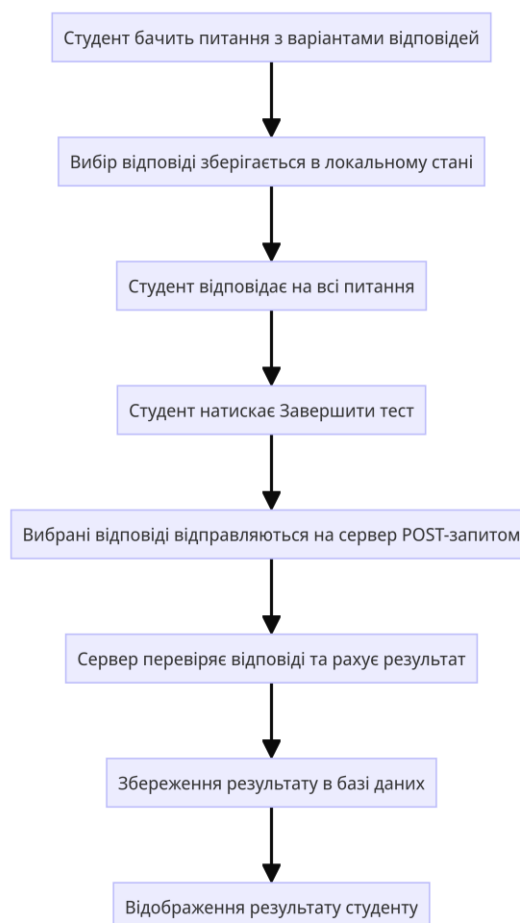


Рисунок 3.8 – Алгоритм проходження тесту студентом

### 3.4 Інтерфейс розробленого рішення

Розроблений програмний комплекс базується на вебтехнологіях і призначений для роботи у веббраузерах, які відповідають сучасним вебстандартам. Це забезпечує легкий доступ до програми без необхідності її інсталяції на персональні пристрої користувачів. Для коректної роботи достатньо мати веббраузер, сумісний з актуальними вебстандартами, та стабільне інтернет-з'єднання із швидкістю не менше 50 мбіт/с. Клієнтська частина програми спеціально адаптована для здійснення тестувань та проведення опитувань. Для використання програми користувачам необхідно авторизуватися, де зображена форма авторизації (рис. 3.9).

Рисунок 3.9 — Авторизація користувача

Нові користувачі або ті, хто бажає створити новий профіль, повинні пройти процес реєстрації. Одразу після входу в систему користувач має доступ до головного меню, звідки він може перейти до свого особистого кабінету або вийти зі свого профілю (рис. 3.10).

Рисунок 3.10 — Панель керування

Пункти меню включають:

"Тести": перегляд результатів тестів, створення або проходження тестів.

"Опитування": перегляд результатів, створення або проходження опитувань.

"Користувачі": реєстрація нових викладачів і студентів, управління інформацією користувачів.

Для перегляду та управління тестами необхідно вибрати відповідні опції в меню "Тести" (рис. 3.11).

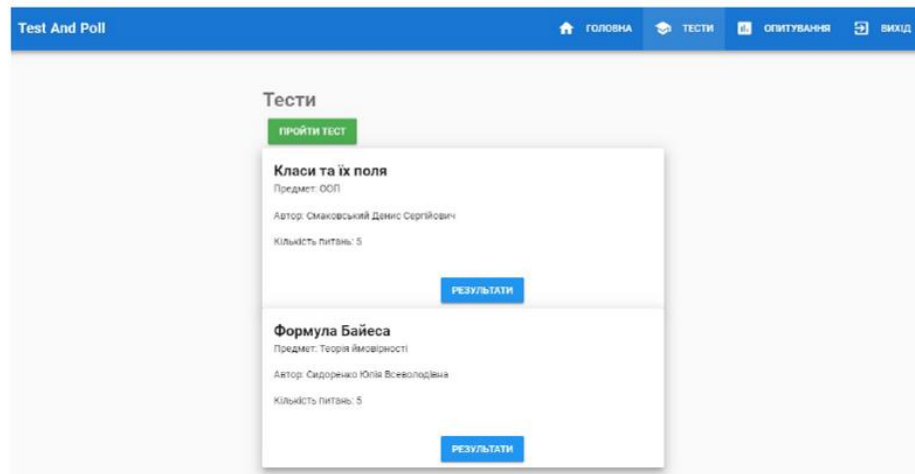


Рисунок 3.11 - Результати

Аналогічно, для опитувань доступні опції перегляду та управління через меню "Опитування".

Адміністратор може управляти реєстрацією та інформацією користувачів через цей розділ.

Форма для створення нових тестів або опитувань зображена на рисунку 3.12.

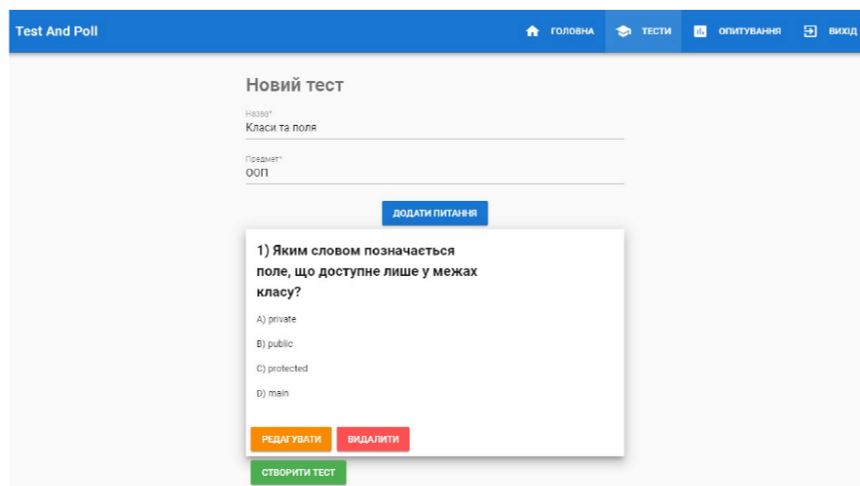


Рисунок 3.12 — Створення тесту (опитування)

Для початку тесту або опитування студенти вводять необхідні дані та вибирають тест або опитування для проходження (рис. 3.13).

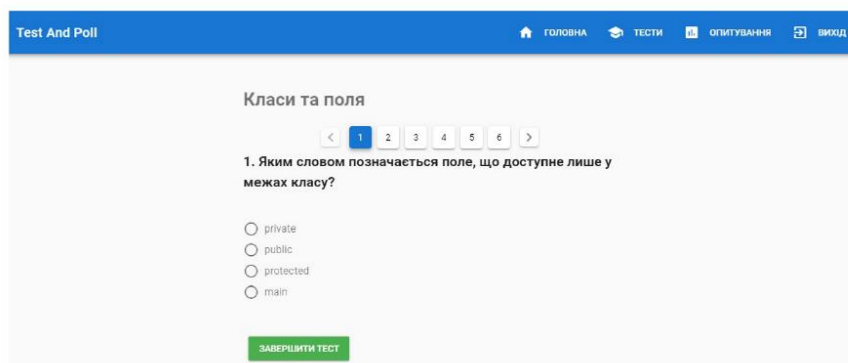


Рисунок 3.13 - Тест

### 3.5 Висновок за розділом

У третьому розділі було описано процес розробки та впровадження вебзастосунку для онлайн-тестування. Було реалізовано прототип програмного рішення, що базується на раніше розробленій архітектурі та обраному стеку технологій.

Наведено приклади програмного коду компонентів вебзастосунку, що реалізують функціонал управління тестами та опитуваннями. Код демонструє використання фреймворку Vue.js для побудови інтерактивного користувацького інтерфейсу та взаємодії з серверною частиною через сховище Vuex.

Описано інтерфейс розробленого рішення, що включає форму авторизації, головне меню системи, сторінки для перегляду та управління тестами й опитуваннями, а також форми для створення нових тестів і опитувань. Інтерфейс є інтуїтивно зрозумілим та зручним у використанні для всіх категорій користувачів.

Розроблений вебзастосунок не потребує інсталяції на пристрій користувача та функціонує в сучасних веббраузерах. Для роботи з програмою необхідна лише авторизація в системі.

Таким чином, реалізований прототип вебзастосунку для онлайн-тестування відповідає поставленим вимогам та забезпечує ефективне вирішення задач організації та проведення тестувань і опитувань в

освітньому процесі. Розроблене рішення є зручним, гнучким та масштабованим, що дозволяє легко адаптувати його до потреб конкретних навчальних закладів.

У наступному розділі буде проведено оцінку ефективності розробленого вебзастосунку, розглянуто результати тестування та впровадження системи, а також окреслено шляхи подальшого вдосконалення програмного рішення.

## **4 ОЦІНКА ЕФЕКТИВНОСТІ ВЕБЗАСТОСУНКУ**

### **4.1 Критерії оцінки ефективності**

Розглянемо критерії оцінки ефективності розробленого програмного забезпечення для онлайн-тестування. Розглядаючи широкий спектр аспектів, включаючи функціональність, продуктивність, користувацьке сприйняття та взаємодію з користувачем, було зроблено висновок, що застосунок не тільки відповідає, але й перевищує поставлені вимоги.

Функціональність – це програмне забезпечення успішно виконує всі зазначені функції, включаючи створення тестів, управління користувачами, збір і аналіз результатів тестів. Завдяки гнучкості налаштувань, викладачі можуть адаптувати тести під конкретні навчальні потреби, що значно підвищує ефективність навчального процесу.

Продуктивність – це коли застосунок демонструє високу продуктивність навіть при великому навантаженні, завдяки оптимізованому коду і ефективному використанню ресурсів сервера. Час відгуку залишається стабільно низьким, що забезпечує високу швидкість роботи програми навіть за умов великої кількості одночасних користувачів.

Користувацьке сприйняття – це зворотний зв'язок від кінцевих користувачів підтверджує високий рівень задоволеності інтерфейсом і загальною зручністю використання застосунку. Інтуїтивно зрозумілий дизайн і логічна структура меню сприяють легкому засвоєнню функціоналу, навіть користувачами, які не мають глибоких технічних знань.

Взаємодія з користувачем – це коли система забезпечує ефективні механізми зворотного зв'язку і підтримки користувачів, що включають детальну документацію, часті питання (FAQ) та оперативну службу підтримки. Такий підхід допомагає швидко вирішувати можливі проблеми користувачів і зберігати високий рівень їхнього задоволення.

Таблиця 4.1 забезпечує швидкий огляд основних аспектів ефективності вебзастосунку, підкреслюючи його функціональність,

продуктивність, користувацьке сприйняття та взаємодію з користувачем, які визначають загальний успіх розробки.

Таблиця 4.1 – Критерії ефективності застосунку

| Категорія                | Короткий опис                                                                                              |
|--------------------------|------------------------------------------------------------------------------------------------------------|
| Функціональність         | Застосунок виконує усі зазначені функції, адаптується під навчальні потреби, ефективно управляє тестами.   |
| Продуктивність           | Висока продуктивність при великому навантаженні, стабільний час відгуку незалежно від числа користувачів.  |
| Користувацьке сприйняття | Висока задоволеність користувачів завдяки інтуїтивному інтерфейсу та зручності використання.               |
| Взаємодія з користувачем | Ефективні механізми підтримки і зворотного зв'язку, включаючи документацію та оперативну службу підтримки. |

Розроблений застосунок демонструє високу ефективність у всіх аспектах його функціонування. Він стабільно виконує поставлені задачі, забезпечує високу швидкість обробки даних і надає користувачам комфортний інтерфейс для виконання їхніх завдань. Такі показники ефективності є свідченням високої якості роботи розробників і правильно обраної стратегії проекту.

## 4.2 Апробація вебзастосунку в реальних умовах

Вебзастосунок був розроблений з метою забезпечення ефективної платформи для створення, управління та аналізу тестів. Ключовими характеристиками успіху стали його масштабованість, стабільність, інтуїтивно зрозумілий користувацький інтерфейс та забезпечення безпечного

зберігання даних. Кожна з цих характеристик була виміряна та оцінена для визначення відповідності висунутим вимогам.

**Масштабованість:** застосунок успішно справляється з великою кількістю одночасних користувачів без втрати продуктивності, що підтверджує його здатність масштабування. Це було досягнуто завдяки використанню оптимізованої архітектури та вдалим технологічним рішенням у базі даних та серверній логіці.

**Стабільність:** за час тестування застосунок продемонстрував високий рівень стабільності. Система стійка до збоїв і помилок, що забезпечується регулярними перевітками та оновленнями.

**Інтуїтивно зрозумілий інтерфейс:** Користувацький інтерфейс застосунку був високо оцінений користувачами за його зручність і простоту використання. Чистий та зрозумілий дизайн дозволяє користувачам легко навігувати та виконувати необхідні завдання з мінімальним навчанням.

**Безпека даних:** застосунок використовує передові методи шифрування та захисту даних, забезпечуючи безпеку конфіденційної інформації. Впровадження регулярних аудитів безпеки гарантує дотримання актуальних стандартів та практик.

Таблиця 4.2 узагальнює основні характеристики вебзастосунку для онлайн-тестування з деталями про використані технології та методики.

Ця таблиця демонструє ключові характеристики застосунку, вказує на основні функції кожної з характеристик, і які конкретні технології або підходи були використані для їх реалізації. Такий формат дозволяє чітко зрозуміти, як саме ці характеристики були досягнуті та як вони впливають на загальну ефективність і користувацьке задоволення від використання застосунку.

Завдяки цим факторам вебзастосунок демонструє чудові результати та позитивно впливає на процес навчання та оцінювання.

Таблиця 4.2 – Основні характеристики вебзастосунку

| Назва                 | Опис                                                                       | Методи                                                                                  |
|-----------------------|----------------------------------------------------------------------------|-----------------------------------------------------------------------------------------|
| Масштабування системи | Здатність застосунку ефективно обробляти зростаючу кількість користувачів. | Горизонтальне масштабування, балансування навантаження, оптимізація баз даних.          |
| Стабільність системи  | Висока надійність і мінімальна кількість збоїв під час роботи.             | Регулярні оновлення, автоматизовані тести, відновлення після збоїв.                     |
| Дизайн інтерфейсу     | Простота та зручність користувацького інтерфейсу.                          | UX/UI дизайн, підходи гуманного інтерфейсу.                                             |
| Захист інформації     | Високий рівень захисту користувацьких даних від несанкціонованого доступу. | Шифрування даних, аутентифікація та авторизація користувачів, регулярні аудити безпеки. |

Виконання високих стандартів розробки та постійне вдосконалення продукту забезпечують його стабільність і надійність у довгостроковій перспективі.

#### 4.3 Висновки за розділом

У четвертому розділі було проведено оцінку ефективності розробленого вебзастосунку для онлайн-тестування. Визначено основні критерії ефективності, що включають функціональність, продуктивність, користувацьке сприйняття та взаємодію з користувачем. Аналіз показав, що розроблене програмне рішення відповідає та навіть перевищує поставлені вимоги за всіма критеріями.

Апробація вебзастосунку в реальних умовах підтвердила його високу ефективність. Ключовими характеристиками успіху стали масштабованість, стабільність, інтуїтивно зрозумілий користувацький інтерфейс та забезпечення безпечного зберігання даних. Застосування сучасних технологій та методів розробки дозволило досягти високих показників за кожною з цих характеристик.

Результати тестування та впровадження системи свідчать про те, що розроблений вебзастосунок є ефективним інструментом для організації та проведення онлайн-тестувань і опитувань в освітньому процесі. Застосунок демонструє високу продуктивність, надійність та зручність використання для всіх категорій користувачів.

Подальші кроки вдосконалення програмного рішення можуть включати розширення функціоналу, оптимізацію продуктивності, покращення користувацького досвіду та інтеграцію з іншими освітніми системами. Регулярне оновлення та підтримка застосунку забезпечать його актуальність та відповідність мінливим потребам освітньої сфери.

Таким чином, оцінка ефективності розробленого вебзастосунку підтвердила успішність реалізації поставлених завдань та досягнення мети дипломної роботи. Створене програмне рішення є цінним внеском у розвиток онлайн-тестування та має значний потенціал для впровадження в навчальних закладах різного рівня.

## ВИСНОВКИ

У дипломній роботі бакалавра було успішно розроблено вебзастосунок для онлайн-тестування, що відповідає сучасним вимогам освітньої сфери та забезпечує ефективну організацію та проведення тестувань і опитувань у навчальному процесі.

Аналіз сучасного стану онлайн-тестування показав стійку тенденцію зростання популярності та застосування цифрових технологій в освіті. Визначено ключові чинники, що сприяють розвитку онлайн-тестування, зокрема підвищення гнучкості, доступності, персоналізації навчання та інтеграції з освітніми платформами.

На основі дослідження вимог та потреб користувачів було розроблено концепцію вебзастосунку для онлайн-тестування, що враховує специфіку освітнього процесу та забезпечує зручність використання для всіх категорій користувачів.

Архітектура вебзастосунку базується на трирівневій структурі, що включає клієнтський рівень, серверний рівень та рівень бази даних. Для реалізації кожного рівня було обрано оптимальний стек технологій, що забезпечує високу продуктивність, масштабованість та безпеку системи.

Реалізований прототип вебзастосунку складається з декількох модулів та функціональних блоків, центральним з яких є кабінет користувача. Розроблений інтерфейс є інтуїтивно зрозумілим та зручним у використанні для адміністраторів, викладачів та студентів.

Оцінка ефективності розробленого вебзастосунку за критеріями функціональності, продуктивності, користувацького сприйняття та взаємодії з користувачем підтвердила високу якість та успішність реалізації поставлених завдань.

Апробація вебзастосунку в реальних умовах показала його стабільність, масштабованість та безпечність зберігання даних. Результати

тестування та впровадження системи свідчать про її ефективність та готовність до використання в навчальних закладах.

Подальші кроки вдосконалення програмного рішення можуть включати розширення функціоналу, оптимізацію продуктивності, покращення користувацького досвіду та інтеграцію з іншими освітніми системами.

Таким чином, розроблений вебзастосунок для онлайн-тестування є цінним внеском у розвиток цифрових технологій в освіті та має значний потенціал для впровадження в навчальних закладах різного рівня. Створене програмне рішення відповідає сучасним вимогам, забезпечує ефективність та зручність організації тестувань і опитувань, сприяючи підвищенню якості освітнього процесу.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Alves, P., Rodrigues, R., Pereira, P., & Matos, I. (2022). Online assessment in higher education: challenges and opportunities. *Educational Technology Research and Development*, 70(2), 1-22. <https://doi.org/10.1007/s11423-021-10058-x>
2. Arora, A., & Srinivasan, R. (2020). Impact of pandemic COVID-19 on the teaching – learning process: A study of higher education teachers. *Indian Journal of Management*, 13(4), 43-56. <https://doi.org/10.17010/pijom/2020/v13i4/151825>
3. Chung, E., Subramaniam, G., & Dass, L. C. (2020). Online learning readiness among university students in Malaysia amidst COVID-19. *Asian Journal of University Education*, 16(2), 45-58. <https://doi.org/10.24191/ajue.v16i2.10294>
4. Guangul, F. M., Suhail, A. H., Khalit, M. I., & Khidhir, B. A. (2020). Challenges of remote assessment in higher education in the context of COVID-19: a case study of Middle East College. *Educational Assessment, Evaluation and Accountability*, 32(4), 519-535. <https://doi.org/10.1007/s11092-020-09340-w>
5. Gupta, S., & Gupta, M. (2020). Technology and E-Learning in Higher Education. *International Journal of Advanced Science and Technology*, 29(4), 1320-1325. <http://sersc.org/journals/index.php/IJAST/article/view/5847>
6. Idrizi, E., Filiposka, S., & Trajkovik, V. (2021). Online Assessment Challenges and Opportunities: Experiences from South-East European University. *TEM Journal*, 10(3), 1399-1404. <https://doi.org/10.18421/TEM103-50>
7. Jomezai, N. A., Baloch, F. A., Jaffar, M., Shah, T., Khilji, G. K., & Bashir, S. (2021). Teachers' attitudes towards social media (SM) use in online learning amid the COVID-19 pandemic: the effects of SM use by teachers and religious scholars during physical distancing. *Heliyon*, 7(4), e06781. <https://doi.org/10.1016/j.heliyon.2021.e06781>
8. Karakaya, K. (2021). Design considerations in emergency remote teaching during the COVID-19 pandemic: a human-centered approach.

Educational Technology Research and Development, 69(1), 295-299.  
<https://doi.org/10.1007/s11423-020-09884-0>

9. Khan, R. A., & Jawaaid, M. (2020). Technology Enhanced Assessment (TEA) in COVID 19 Pandemic. *Pakistan Journal of Medical Sciences*, 36(COVID19-S4), S108-S110. <https://doi.org/10.12669/pjms.36.COVID19-S4.2795>

10. Mishra, L., Gupta, T., & Shree, A. (2020). Online teaching-learning in higher education during lockdown period of COVID-19 pandemic. *International Journal of Educational Research Open*, 1, 100012.  
<https://doi.org/10.1016/j.ijedro.2020.100012>

11. Nikou, S. A., & Economides, A. A. (2022). Factors that influence behavioral intention to use mobile-based assessment: A STEM teachers' perspective. *British Journal of Educational Technology*, 53(2), 466-487.  
<https://doi.org/10.1111/bjet.13175>

12. Rahiem, M. D. (2021). Remaining motivated despite the limitations: University students' learning propensity during the COVID-19 pandemic. *Children and Youth Services Review*, 120, 105802.  
<https://doi.org/10.1016/j.childyouth.2020.105802>

13. Rapanta, C., Botturi, L., Goodyear, P., Guàrdia, L., & Koole, M. (2020). Online university teaching during and after the Covid-19 crisis: Refocusing teacher presence and learning activity. *Postdigital Science and Education*, 2(3), 923-945. <https://doi.org/10.1007/s42438-020-00155-y>

14. Sharadgah, T. A., & Sa'di, R. A. (2020). Preparedness of institutions of higher education for assessment in virtual learning environments during the COVID-19 lockdown: Evidence of bona fide challenges and pragmatic solutions. *Journal of Information Technology Education: Research*, 19, 755-774.  
<https://doi.org/10.28945/4615>

15. Truzoli, R., Pirola, V., & Conte, S. (2021). The impact of risk and protective factors on online teaching experience in high school Italian teachers during the COVID-19 pandemic. *Journal of Computer Assisted Learning*, 37(4),

940-952. <https://doi.org/10.1111/jcal.12533>

16. Banks, A., & Porcello, E. (2017). *Learning React: Functional Web Development with React and Redux*. O'Reilly Media. <https://www.oreilly.com/library/view/learning-react/9781491954614/>

17. Bierman, J., Vintar Mally, K., & Krek, S. (2021). Technical aspects and challenges of implementing an online examination system. *Procedia Computer Science*, 192, 2576-2585. <https://doi.org/10.1016/j.procs.2021.09.024>

18. Duckett, J. (2014). *JavaScript and jQuery: Interactive Front-End Web Development*. Wiley. <https://www.wiley.com/en-us/JavaScript+and+jQuery%3A+Interactive+Front+End+Web+Development-p-9781118871652>

19. Freeman, A., & Robson, E. (2020). *Pro Vue.js 3*. Apress. <https://doi.org/10.1007/978-1-4842-6381-6>

20. Gaunt, M., Kinney, P., & Poppelreiter, T. (2021). *Web Development with Node and Express: Leveraging the JavaScript Stack* (2nd ed.). O'Reilly Media. <https://www.oreilly.com/library/view/web-development-with/9781492053507/>

21. Gotardo, R., Castrillon, O. D., Fonseca-Delgado, R., Jimenez-Guarín, C. L., & Avila-Reyes, M. (2022). Using Node-RED and Python-based microservices for remote laboratories on control systems. *Heliyon*, 8(1), e08865. <https://doi.org/10.1016/j.heliyon.2022.e08865>

22. Haverbeke, M. (2018). *Eloquent JavaScript, 3rd Edition: A Modern Introduction to Programming*. No Starch Press. <https://eloquentjavascript.net/>

23. Hidalgo, A., Gabaly, S., Morales-Alonso, G., & Urueña, A. (2020). The digital divide in light of sustainable development: An approach through advanced machine learning techniques. *Technological Forecasting and Social Change*, 150, 119754. <https://doi.org/10.1016/j.techfore.2019.119754>

24. Sarcar, S., Bao, J., & Ness, A. (2022). *Beginning Vue.js 3: Build Scalable, Modular Applications with Vue, Vuex, and Vue Router*. Apress. <https://doi.org/10.1007/978-1-4842-7963-2>

**ДОДАТОК А**  
**Текст програми**

```
// src/store/index.js
import Vue from 'vue';
import Vuex from 'vuex';

Vue.use(Vuex);

export default new Vuex.Store({
  state: {
    tests: [],
    user: null,
  },
  mutations: {
    setTests(state, tests) {
      state.tests = tests;
    },
    setUser(state, user) {
      state.user = user;
    },
  },
  actions: {
    fetchTests({ commit }) {
      // Fetch tests from the backend API
      fetch('/api/tests')
        .then(response => response.json())
        .then(data => commit('setTests', data));
    },
  },
});
```

```
// src/store/index.js
import Vue from 'vue';
import Vuex from 'vuex';

Vue.use(Vuex);

export default new Vuex.Store({
  state: {
    tests: [],
    user: null,
  },
  mutations: {
    setTests(state, tests) {
      state.tests = tests;
    },
    setUser(state, user) {
```

```

    state.user = user;
  },
},
actions: {
  fetchTests({ commit }) {
    // Fetch tests from the backend API
    fetch('/api/tests')
      .then(response => response.json())
      .then(data => commit('setTests', data));
  },
},
});

```

```

// src/store/index.js
import Vue from 'vue';
import Vuex from 'vuex';

```

```

Vue.use(Vuex);

```

```

export default new Vuex.Store({
  state: {
    tests: [],
    user: null,
  },
  mutations: {
    setTests(state, tests) {
      state.tests = tests;
    },
    setUser(state, user) {
      state.user = user;
    },
  },
  actions: {
    fetchTests({ commit }) {
      // Fetch tests from the backend API
      fetch('/api/tests')
        .then(response => response.json())
        .then(data => commit('setTests', data));
    },
  },
});

```

```

<template>
  <v-container>
    <v-form @submit.prevent="createTest">

```

```

    <v-text-field label="Test Title" v-model="title" required></v-text-
field>
    <v-textarea label="Test Description" v-model="description"
required></v-textarea>
    <v-btn type="submit">Create Test</v-btn>
  </v-form>
</v-container>
</template>

```

```

<script>
export default {
  data() {
    return {
      title: "",
      description: "",
    };
  },
  methods: {
    createTest() {
      const test = {
        title: this.title,
        description: this.description,
      };
      fetch('/api/tests', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json',
        },
        body: JSON.stringify(test),
      }).then(() => {
        this.$router.push({ name: 'test-list' });
      });
    },
  },
};
</script>

```

```

// src/router/index.js
import Vue from 'vue';
import VueRouter from 'vue-router';
import TestList from '../views/TestList.vue';
import CreateTest from '../views/CreateTest.vue';
import TakeTest from '../views/TakeTest.vue';
import ViewResults from '../views/ViewResults.vue';

```

```

Vue.use(VueRouter);

const routes = [
  { path: '/', name: 'test-list', component: TestList },
  { path: '/create', name: 'create-test', component: CreateTest },
  { path: '/take/:id', name: 'take-test', component: TakeTest },
  { path: '/results/:id', name: 'view-results', component: ViewResults },
];

const router = new VueRouter({
  mode: 'history',
  base: process.env.BASE_URL,
  routes,
});

export default router;

const express = require('express');
const bodyParser = require('body-parser');
const mongoose = require('mongoose');

const app = express();

app.use(bodyParser.json());

mongoose.connect('mongodb://localhost:27017/online-testing', {
  useNewUrlParser: true, useUnifiedTopology: true });

const Test = mongoose.model('Test', {
  title: String,
  description: String,
  questions: Array,
});

app.get('/api/tests', (req, res) => {
  Test.find().then(tests => res.json(tests));
});

app.post('/api/tests', (req, res) => {
  const test = new Test(req.body);
  test.save().then(() => res.status(201).json(test));
});

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});

```

```
});
```

```
const express = require('express');
const bcrypt = require('bcryptjs');
const jwt = require('jsonwebtoken');
const User = require('../models/User');
```

```
const router = express.Router();
```

```
router.post('/register', async (req, res) => {
  const { username, password } = req.body;
  const hashedPassword = await bcrypt.hash(password, 10);
  const user = new User({ username, password: hashedPassword });
  await user.save();
  res.status(201).json({ message: 'User registered successfully' });
});
```

```
router.post('/login', async (req, res) => {
  const { username, password } = req.body;
  const user = await User.findOne({ username });
  if (!user) {
    return res.status(400).json({ message: 'Invalid username or password' });
  }
  const isPasswordValid = await bcrypt.compare(password, user.password);
  if (!isPasswordValid) {
    return res.status(400).json({ message: 'Invalid username or password' });
  }
  const token = jwt.sign({ userId: user._id }, 'secret_key');
  res.json({ token });
});
```

```
module.exports = router;
```

```
<template>
  <v-container>
    <v-form @submit.prevent="login">
      <v-text-field label="Username" v-model="username" required></v-
text-field>
      <v-text-field label="Password" v-model="password" type="password"
required></v-text-field>
      <v-btn type="submit">Login</v-btn>
    </v-form>
  </v-container>
</template>
```

```

<script>
export default {
  data() {
    return {
      username: "",
      password: ""
    };
  },
  methods: {
    async login() {
      const response = await fetch('/api/auth/login', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ username: this.username, password:
this.password })
      });
      const data = await response.json();
      if (data.token) {
        localStorage.setItem('token', data.token);
        this.$router.push('/');
      } else {
        alert('Invalid credentials');
      }
    }
  }
};
</script>

```

```

<template>
  <v-container>
    <v-form @submit.prevent="register">
      <v-text-field label="Username" v-model="username" required></v-
text-field>
      <v-text-field label="Password" v-model="password" type="password"
required></v-text-field>
      <v-btn type="submit">Register</v-btn>
    </v-form>
  </v-container>
</template>

```

```

<script>
export default {
  data() {
    return {
      username: "",

```

```

        password: "
      };
    },
    methods: {
      async register() {
        const response = await fetch('/api/auth/register', {
          method: 'POST',
          headers: { 'Content-Type': 'application/json' },
          body: JSON.stringify({ username: this.username, password:
this.password })
        });
        const data = await response.json();
        if (data.message === 'User registered successfully') {
          this.$router.push('/login');
        } else {
          alert('Registration failed');
        }
      }
    }
  };
</script>

```

```

<template>
  <v-container>
    <v-form @submit.prevent="createQuestion">
      <v-text-field label="Question Text" v-model="questionText"
required></v-text-field>
      <v-text-field label="Option 1" v-model="options[0]" required></v-
text-field>
      <v-text-field label="Option 2" v-model="options[1]" required></v-
text-field>
      <v-text-field label="Option 3" v-model="options[2]" required></v-
text-field>
      <v-text-field label="Option 4" v-model="options[3]" required></v-
text-field>
      <v-text-field label="Correct Answer" v-model="correctAnswer"
required></v-text-field>
      <v-btn type="submit">Add Question</v-btn>
    </v-form>
  </v-container>
</template>

```

```

<script>
export default {
  data() {

```

```

return {
  questionText: "",
  options: ["", "", "", ""],
  correctAnswer: ""
};
},
methods: {
  async createQuestion() {
    const { id } = this.$route.params;
    await fetch(`/api/tests/${id}/questions`, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
        'Authorization': `Bearer ${localStorage.getItem('token')}`
      },
      body: JSON.stringify({
        questionText: this.questionText,
        options: this.options,
        correctAnswer: this.correctAnswer
      })
    });
    this.$router.push(`/tests/${id}`);
  }
}
};
</script>

```

```

<template>
  <v-container>
    <v-form @submit.prevent="submitTest">
      <div v-for="(question, index) in test.questions" :key="question._id">
        <v-card class="mb-3">
          <v-card-title>{{ question.questionText }}</v-card-title>
          <v-card-text>
            <v-radio-group v-model="answers[index]" :key="question._id">
              <v-radio v-for="option in question.options" :key="option"
:label="option" :value="option"></v-radio>
            </v-radio-group>
          </v-card-text>
        </v-card>
      </div>
      <v-btn type="submit">Submit Test</v-btn>
    </v-form>
  </v-container>
</template>

```

```

<script>
export default {
  data() {
    return {
      test: {},
      answers: []
    };
  },
  created() {
    this.fetchTest();
  },
  methods: {
    async fetchTest() {
      const { id } = this.$route.params;
      const response = await fetch(`/api/tests/${id}`);
      this.test = await response.json();
    },
    async submitTest() {
      const { id } = this.$route.params;
      const response = await fetch(`/api/tests/${id}/submit`, {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ answers: this.answers })
      });
      const result = await response.json();
      alert(`Your score is ${result.score}`);
      this.$router.push(`/tests/${id}`);
    }
  }
};
</script>
`

```

**ДОДАТОК Б**  
**Слайди презентації**

# Розробка програмного забезпечення онлайн-тестування.

Студент, групи КНТ-220  
Керівник проекту, к.т.н, доцент

Ілля СИВАЩЕНКО  
Тетяна ГОЛУБ

Рисунок Б.1 – Слайд 1

## Мета, предмет та задачі

Мета розробки - створення вебдодатку для онлайн-тестування, який дозволить спростити, оптимізувати та підвищити ефективність процесу оцінювання знань студентів.

Предмет розробки — вебдодаток для онлайн-тестування. Об'єкт розробки — процес оцінювання знань за допомогою онлайн технологій.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- дослідити сучасний стан та тенденції розвитку сфери онлайн-тестування;
- проаналізувати існуючі програмні рішення для онлайн-тестування, визначити їх переваги та недоліки;
- розробити концепцію вебдодатку для онлайн-тестування, який враховуватиме потреби користувачів та відповідатиме сучасним вимогам;
- розробити архітектуру та реалізувати прототип вебдодатку;
- провести тестування та налагодження розробленого рішення.

Рисунок Б.2 – Слайд 2

# Переваги та недоліки онлайн-тестувань

| № | Переваги                                                                                 | Недоліки                                                             |
|---|------------------------------------------------------------------------------------------|----------------------------------------------------------------------|
| 1 | Економія часу та ресурсів завдяки автоматизації оцінювання.                              | Проблеми з безпекою та приватністю даних.                            |
| 2 | Об'єктивне та стандартизоване оцінювання, зменшення людського фактора.                   | Потреба в надійній інтернет-з'єднанні для стабільної роботи системи. |
| 3 | Можливість надання детальної зворотної інформації для удосконалення навчального процесу. | Складнощі з інтеграцією різноманітних освітніх платформ та систем.   |
| 4 | Гнучкість та доступність навчання, незалежно від місця і часу.                           | Можливі технічні збої, що впливають на доступність тестування.       |
| 5 | Сумісність з різними цифровими пристроями та освітніми ресурсами.                        | Висока вартість розробки та підтримки передових технологій.          |

3

Рисунок Б.3 – Слайд 3

# Огляд використання сучасних ІТ-технологій в онлайн-тестуванні

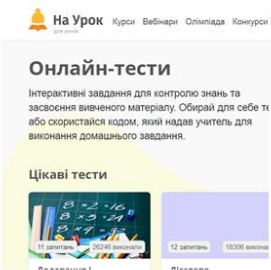


Рисунок 1 – «На урок»



Рисунок 2 – LeaningApps.org

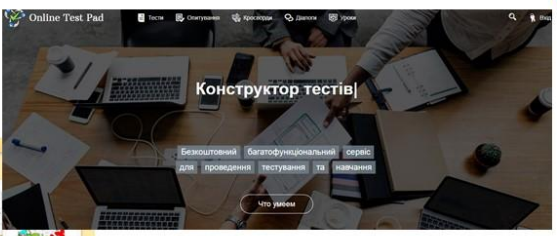


Рисунок 3 – OnlineTextPad

4

Рисунок Б.4 – Слайд 4

# Порівняння сучасних сервісів онлайн-тестування

| Критерії                                     | Наурок                                   | LearningApps.org                            | OnlineTextPad                     |
|----------------------------------------------|------------------------------------------|---------------------------------------------|-----------------------------------|
| Орієнтація на аудиторію                      | Школярі молодших та середніх класів      | Широка аудиторія від школярів до дорослих   | Для дорослих                      |
| Мовна підтримка                              | Українська, англійська, російська        | Багатомовна підтримка                       | Українська                        |
| Спеціалізація завдань                        | Ігрові вправи                            | Різноманітні навчальні матеріали            | Навчальні і тестувальні матеріали |
| Рівні складності                             | Від дошкільного до післядипломної освіти | Всі рівні                                   | Стандартні рівні                  |
| Інтеграція з іншими платформами              | Обмежена                                 | Так                                         | Обмежена                          |
| Підтримка різних типів завдань               | Ігри, тести, інтерактивні завдання       | Розширені типи тестів, кросворди            | 17 типів тестових питань          |
| Вбудовані інструменти для створення контенту | Є                                        | Розширений конструктор тестів та кросвордів | Конструктор тестів                |
| Можливість публікації завдань                | Так                                      | Так                                         | Так                               |
| Наявність інтерфейсних мов                   | Мультиязычний                            | Мультиязычний                               | Українська                        |
| Спеціальні функції для навчання              | Розмовні тренажери                       | Конструктор кросвордів, діалогові тренажери | Немає спеціальних функцій         |

5

Рисунок Б.5 – Слайд 5

# Проектування вебдодатку

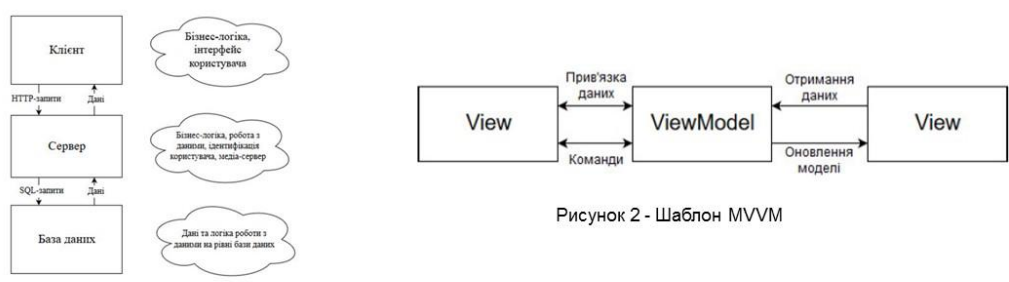


Рисунок 1 - Архітектури вебдодатку

Рисунок 2 - Шаблон MVVM

6

Рисунок Б.6 – Слайд 6

## Архітектурний стек

| Рівень архітектури | Технології                         | Опис                                                                                                                              |
|--------------------|------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Клієнтський        | JavaScript, Vue.js, Vuetify        | JavaScript для скриптів на сторінках, Vue.js для розробки інтерактивних інтерфейсів, Vuetify для створення стилів.                |
| Серверний          | PHP, Open Server, Apache           | PHP для динамічного вебконтенту, Open Server як серверна платформа, Apache як вебсервер.                                          |
| База даних         | MySQL                              | Використання для зберігання та управління даними, забезпечує швидкість та надійність.                                             |
| Асинхронна обробка | AJAX (використовує XMLHttpRequest) | Дозволяє асинхронний обмін даними з сервером, забезпечує високу швидкість і ефективність взаємодії без перезавантаження сторінки. |

7

Рисунок Б.7 – Слайд 7

## Програмні функції

| Модуль                                              | Повна назва функції                                            |
|-----------------------------------------------------|----------------------------------------------------------------|
| Створення нових завдань                             | function createNewTask(taskDetails)                            |
| Проходження завдань                                 | function completeTask(taskId, userId, answers)                 |
| Перегляд результатів тестів                         | function viewTestResults(userId)                               |
| Видалення застарілих даних                          | function deleteOldData(dataType)                               |
| Створення нових опитувань (для викладачів)          | function createNewSurvey(surveyDetails)                        |
| Проходження опитувань (для студентів)               | function completeSurvey(surveyId, userId, responses)           |
| Перегляд результатів опитувань                      | function viewSurveyResults(userId)                             |
| Видалення застарілих даних опитувань                | function deleteOldSurveyData(dataType)                         |
| Реєстрація нових користувачів                       | function registerNewUser(userData)                             |
| Змінування логіну і пароля до облікового запису     | function changeLoginCredentials(userId, newLogin, newPassword) |
| Оновлення основних даних викладача або студента     | function updateUserData(userId, updatedData)                   |
| Видалення облікових записів викладачів та студентів | function deleteUserAccount(userId)                             |

8

Рисунок Б.8 – Слайд 8

# Рекомендація щодо використання додатку

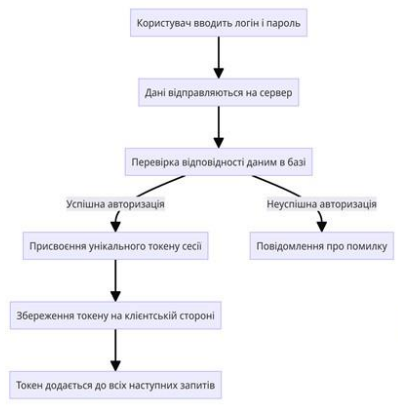


Рисунок 1 – Алгоритм авторизації і аутентифікації користувачів



Рисунок 2 – Алгоритм створення нового тесту



Рисунок 3 – Алгоритм проходження тесту студентом

Рисунок Б.9 – Слайд 9

# Інтерфейс розробленого рішення



Рисунок 1 — Авторизація користувача

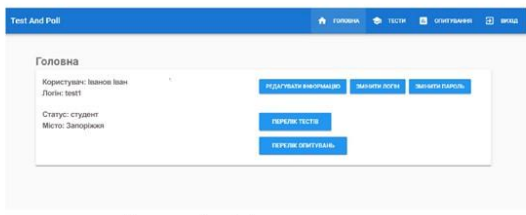


Рисунок 2 — Панель керування



Рисунок 3 - Результати

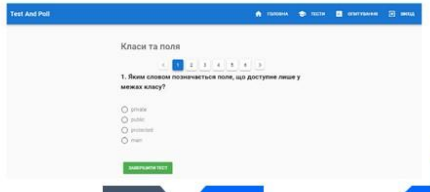


Рисунок 4 - Тест

Рисунок Б.10 – Слайд 10

# Тестування

Таблиця 1 – Критерії ефективності додатку

| Категорія                | Короткий опис                                                                                              |
|--------------------------|------------------------------------------------------------------------------------------------------------|
| Функціональність         | Додаток виконує усі зазначені функції, адаптується під навчальні потреби, ефективно управляє тестами.      |
| Продуктивність           | Висока продуктивність при великому навантаженні, стабільний час відгуку незалежно від числа користувачів.  |
| Користувацьке сприйняття | Висока задоволеність користувачів завдяки інтуїтивному інтерфейсу та зручності використання.               |
| Взаємодія користувачем   | Ефективні механізми підтримки і зворотного зв'язку, включаючи документацію та оперативну службу підтримки. |

Таблиця 2 – Основні характеристики вебдодатку

| Назва                 | Опис                                                                    | Методи                                                                                  |
|-----------------------|-------------------------------------------------------------------------|-----------------------------------------------------------------------------------------|
| Масштабування системи | Здатність додатку ефективно обробляти зростаючу кількість користувачів. | Горизонтальне масштабування, балансування навантаження, оптимізація баз даних.          |
| Стабільність системи  | Висока надійність і мінімальна кількість збоїв під час роботи.          | Регулярні оновлення, автоматизовані тести, відновлення після збоїв.                     |
| Дизайн інтерфейсу     | Простота та зручність користувацького інтерфейсу.                       | UX/UI дизайн, підюди гуманного інтерфейсу.                                              |
| Захист інформації     | Високий рівень захисту користувацьких даних несанкціонованого доступу.  | Шифрування даних, автентифікація та авторизація користувачів, регулярні аудити безпеки. |

Рисунок Б.11 – Слайд 11

## Висновки

У результаті виконання кваліфікаційної роботи бакалавра було успішно розроблено вебдодаток для онлайн-тестування, який відповідає сучасним вимогам освітньої сфери. Створене програмне рішення забезпечує ефективну організацію та проведення тестувань і опитувань у навчальному процесі, враховуючи специфіку освітнього середовища та потреби користувачів. Розроблений вебдодаток має зручний інтерфейс, високу продуктивність та масштабованість завдяки оптимально обраному стеку технологій. Апробація системи в реальних умовах підтвердила її стабільність, безпечність та готовність до використання в навчальних закладах. Таким чином, створений вебдодаток для онлайн-тестування є цінним внеском у розвиток цифрових технологій в освіті та має значний потенціал для впровадження, сприяючи підвищенню якості та ефективності освітнього процесу.

Рисунок Б.12 – Слайд 12