Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет _____Інфокомунікацій_____

(повна назва)

Кафедра _____Інформаційно-мережної інженерії_____

(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти _____перший (бакалаврський)_____

_____Розробка сайту для дистанційного навчання_____

(тема)

Виконав:

здобувач ____четвертого_____ року навчання,

групи _____ТРІМІ-21-2_____

_____Денис Калашник_____

(власне ім’я, прізвище)

Спеціальність _172 Телекомунікації_____

_____та радіотехніка_____

(код і повна назва спеціальності)

Тип програми ____освітньо-професійна_____

Освітня програма ____Інформаційно-мережна

___інженерія_____

(повна назва освітньої програми)

Керівник _ст. викл. Олександр Бибка_____

(посада, власне ім’я, прізвище)

Допускається до захисту

Завідувач кафедри __ІМІ__ _____ __Валерій Безрук____

(підпис) (власне ім’я, прізвище)

2025 р.

Не містить відомостей, заборонених до відкритого публікування

Студент _____ / __Калашник Д.С._____/

(підпис) (прізвище та ініціали)

Керівник _____ / ___Бибка О.І._____/

(підпис) (прізвище та ініціали)

Харківський національний університет радіоелектроніки

Факультет _____Інфокомунікацій_____

Кафедра _____Інформаційно-мережної інженерії_____

Рівень вищої освіти _____перший (бакалаврський)_____

Спеціальність_____172 Телекомунікації та радіотехніка_____

(код і повна назва)

Тип програми _____освітньо-професійна_____

Освітня програма _____Інформаційна-мережна інженерія_____

(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«____»_____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувачеві _____Калашнику Денису Сергійовичу_____

(прізвище, ім'я, по батькові)

1. Тема роботи _____Розробка сайту для дистанційного навчання_____

затверджена наказом університету від ___23___травня_____2025_р.____№_410 Ст.____

2. Термін подання здобувачем роботи до екзаменаційної комісії __17__червня__2025_р.___

3. Вихідні дані до роботи ___Технічне завдання. Розробити сайт для дистанційного_____ проведення тестів. Вимоги до сайту: реєстрація користувачів, питання і кілька варіантів____ відповідей, випадковий вибір питань, можливість завантаження нових тестів, вхід за_____ паролем, журнал для зберігання результатів, можливість показу результатів користувачу, обмежений час на кожне питання 1 хвилина.

4. Перелік питань, що потрібно опрацювати в роботі

Вступ

1. Аналіз систем онлайн-тестування

2. Проєктування та реалізація веб-сайту

3. Тестування роботи сайту

Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _Слайди у___ форматі PowerPoint (мета та задачі роботи, основні моменти реалізації веб-сайту, _____ відображення на рисунках, висновки)_____ _____

_____

_____

_____

_____

_____

_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основний розділ	ст. викладач кафедри ІМІ Бибка Олександр Іванович		20.06

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Аналіз предметної області, огляд джерел, збір інформації	23.05-25.05	виконано
2	Аналіз існуючих рішень	26.05-28.05	виконано
3	Вибір інструментів розробки сайту	29.05-30.05	виконано
4	Розробка сайту	01.06-12.06	виконано
4	Тестування системи, виправлення помилок	13.06-14.06	виконано
6	Підготовка графічних матеріалів	15.06-16.06	виконано

Дата видачі завдання __23___-__травня_____2025_ р.

Здобувач _____Денис Калашник

(підпис)

Керівник роботи _____ст. викл. Олександр Бибка

(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка 94 с., 25 рис., 1 табл., 2 додатки, 24 джерела

Об’єкт роботи – веб-сайт системи дистанційного навчання.

Мета роботи – розробити веб-сайт для проведення онлайн-тестування.

Проведено огляд популярних систем тестування та платформ дистанційного навчання. Визначено основні вимоги до функціоналу системи. Спроєктовано архітектуру веб-додатку, створено базу даних та реалізовано функції для зручного користування сайтом. Веб-сайт розгорнуто у відкритому доступі на платформі PythonAnywhere та проведено тестування основного функціоналу системи.

ДИСТАНЦІЙНЕ НАВЧАННЯ, Django, ОНЛАЙН-ТЕСТУВАННЯ, БАЗА ДАНИХ, ВЕБ-САЙТ, PythonAnywhere.

THE ABSTRACT

Explanatory slip 94 p., 25 fig., 1 tab., 2 app., 24 sources.

The object of work is a website of a distance learning system.

Purpose – to develop a website for online learning system.

A review of popular testing systems and distance learning platforms is carried out. The basic requirements for the system functionality are determined. The architecture of the web application is designed, a database is created and functions for convenient use of the site are implemented. The website was deployed in the public domain on the PythonAnywhere platform and the basic functionality of the system was tested.

DISTANCE LEARNING, Django, ONLINE TESTING, DATABASE, WEBSITE, PythonAnywhere.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

СУБД – система управління базами даних.

CSV (від англ. Comma-Separated Values) – це абревіатура, що в перекладі означає значення, розділені комами.

JSON (від англ. JavaScript Object Notation) – перекладається з англійської, як текстовий формат обміну даними.

HTML (від англ. HyperText Markup Language) – розшифровується як мова розмітки гіпертексту. Це стандартна мова розмітки для створення вебсторінок, яка використовується для опису структури та вмісту вебсторінок.

CSS (від англ. Cascading Style Sheets) – перекладається з англійської, як каскадні таблиці стилів. Це мова стилів, що використовується для опису зовнішнього виду вебсторінок, написаних мовами розмітки даних, таких як HTML.

JS (від англ. JavaScript) – це мова програмування для інтерактивності вебсайтів.

PHP (від англ. JavaScript) – препроцесор гіпертексту.

SQL (від англ. Structured Query Language) – це мова програмування, призначена для управління реляційними базами даних.

ORM (від англ. Object-Relational Mapping) – перекладається з англійської, як об’єктно-реляційне відображення.

HTTP (від англ. HyperText Transfer Protocol) – перекладається з англійської, як протокол передачі гіпертексту.

URL (від англ. Uniform Resource Locator) – перекладається з англійської, як уніфікований покажчик ресурсу.

API (від англ. Application Programming Interface) – перекладається з англійської, як інтерфейс прикладного програмування.

XSS (від англ. Cross-Site Scripting) – перекладається з англійської, як вразливість міжсайтового скриптингу.

CSRF (від англ. Cross-Site Request Forgery) – перекладається з англійської, як міжсайтове підроблення запиту.

MVT (від англ. Model-View-Template) – це архітектурний шаблон Django.

VPS (від англ. Virtual Private Server) – це абревіатура, що в перекладі означає віртуальний приватний сервер.

UI (від англ. User Interface) – це абревіатура, що в перекладі означає інтерфейс користувача.

UX (від англ. User Experience) – це абревіатура, що в перекладі означає досвід користувача.

ID (від англ. Identifier) – це унікальний номер або код.

OWASP (від англ. Open Web Application Security Project) – це міжнародна організація, що займається покращенням безпеки програмного забезпечення.

2FA (від англ. Two-Factor Authentication) – двохфакторна автентифікація.

ACL (від англ. Access Control List) – список керування доступом.

XML (від англ. eXtensible Markup Language) – розширювана мова розміток.

BSD (від англ. Berkeley Software Distribution) – це сімейство операційних систем, що походить від UNIX.

ВСТУП

У сфері освіти дедалі частіше використовуються онлайн-інструменти для перевірки знань. Перехід до дистанційного та змішаного навчання створив потребу у зручних та надійних рішеннях, які дозволяють ефективно контролювати рівень засвоєння матеріалу в режимі онлайн. Традиційні способи перевірки, такі як письмові завдання або усне опитування, виявилися недостатньо ефективними в умовах дистанційного навчання, особливо під час нинішніх реалій таких як війна.

У ході підготовки дипломної роботи було розглянуто існуючі інструменти для онлайн-тестування (Google Forms, Moodle, Quizizz, Kahoot), а також технології, які застосовуються при створенні подібних систем. Особливу увагу приділено технічним рішенням, мові програмування Python, фреймворкам Django та Flask, системам управління базами даних, механізмам автентифікації, захисту даних та інтеграції з форматами даних CSV та JSON. До того ж, було проведено порівняння можливих середовищ для розміщення сайту, проаналізовано їх переваги, обмеження та доцільність застосування у навчальних цілях.

Метою дипломної роботи є ознайомлення з сучасними веб технологіями, що використовуються у сфері онлайн-тестування, з подальшим обґрунтуванням вибору оптимальних інструментів для створення власного веб-сайту. Здобуті знання та результати аналізу були безпосередньо застосовані при реалізації індивідуального проекту і стали теоретичною базою для подальшого написання дипломної роботи. Сформовано цілісний огляд предметної області, який дозволив у достатньому обсязі підійти до розробки функціонального веб-сайту для онлайн-тестування.

1 АНАЛІЗ СИСТЕМ ОНЛАЙН ТЕСТУВАННЯ

1.1 Google Forms

Google Forms – це онлайн-сервіс для створення та аналізу анкет, опитувань, тестів і форм зворотного зв’язку. Він входить до складу пакету Google Workspace і працює через браузер, що робить його зручним та доступним для широкого кола користувачів без необхідності встановлення додаткового програмного забезпечення [1].

Форми можна швидко створювати завдяки зручному інтерфейсу, який дозволяє додавати різні типи запитань – від відкритих відповідей до варіантів із множинним вибором. Крім того, кілька користувачів можуть одночасно працювати над однією формою, змінюючи або редагуючи її в режимі реального часу (рис.1.1).

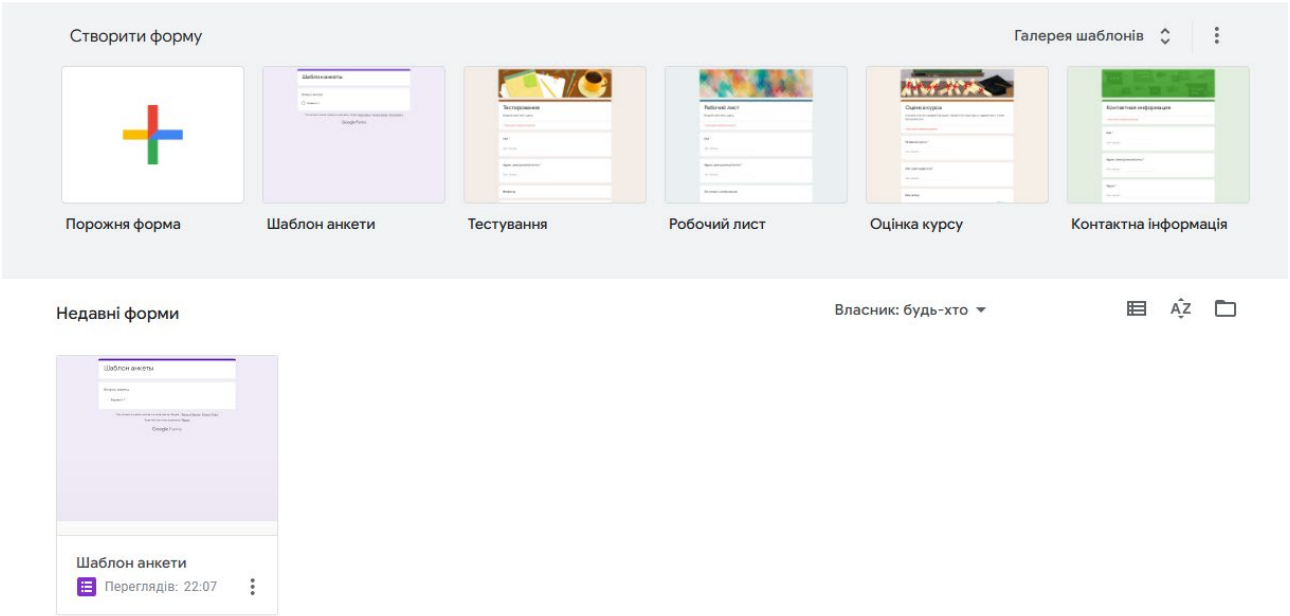

Рисунок 1.1 – Головне вікно Google Forms

Також, інструмент надає можливості для візуального налаштування: зміну кольорів, вставку зображень і логотипів. Це дозволяє адаптувати форму під стиль організації. Крім опитувань, Google Forms активно використовується в освітній сфері, наприклад для створення тестів, які автоматично оцінюють відповіді, допомагаючи вчителям швидко отримати результати тестування та оцінити рівень знань студентів [2].

Google Forms є зручною платформою з багатьма можливостями: змога надсилати форми електронною поштою, ділитись посиланням або вбудовувати на веб сайти за допомогою HTML-коду. Крім цього система передбачає додавання різних доповнень для розширення функціональності форм, наприклад, таймери для тестів або генератори питань тощо.

Google Forms пропонує багато різних шаблонів на будь-яку тематику для візуального покращення аби уникнути простоти та додати сторінці тесту приємного виду, що є гарними функціями для цієї системи.

1.2 Moodle

Moodle – це освітня платформа, яка працює як система керування навчанням або просто як універсальний інструмент для онлайн-освіти. Вона надає широкий набір функцій для викладачів, студентів та адміністраторів і активно використовується як у закладах вищої освіти так і для самостійного навчання.

Ця система є відкритим програмним забезпеченням з відкритим кодом (Open Source). Moodle абсолютно безкоштовна для використання, і для її роботи не потрібне жодне програмне забезпечення. Це дозволяє будь-якому навчальному закладу впровадити повноцінну, легальну освітню платформу без фінансових витрат, а за необхідності можна адаптувати її під власні потреби, змінюючи програмний код [4].

На сьогодні Moodle – одна з найпопулярніших навчальних систем у світі та в Україні. Кількість користувачів перевищує 400 мільйонів, і це число продовжує рости стрімкими темпами, ніж будь-який з її конкурентів. Ще у 2018 році ця платформа стала лідером глобального ринку систем дистанційного навчання.

У Європі приблизно 60% вищих навчальних закладів використовують саме Moodle. В Україні, де використання платних сервісів обмежене, Moodle користується неабияким попитом, і її обрання логічне. Також, як і в інших додатках є постійна підтримка у разі якихось труднощів або питань які виникають у користувача. Окрім цього є окремий форум спільноти Moodle – активні користувачі залюбки надають допомогу новачкам [3]. На рисунку 1.2 зображено сайт платформи Moodle.

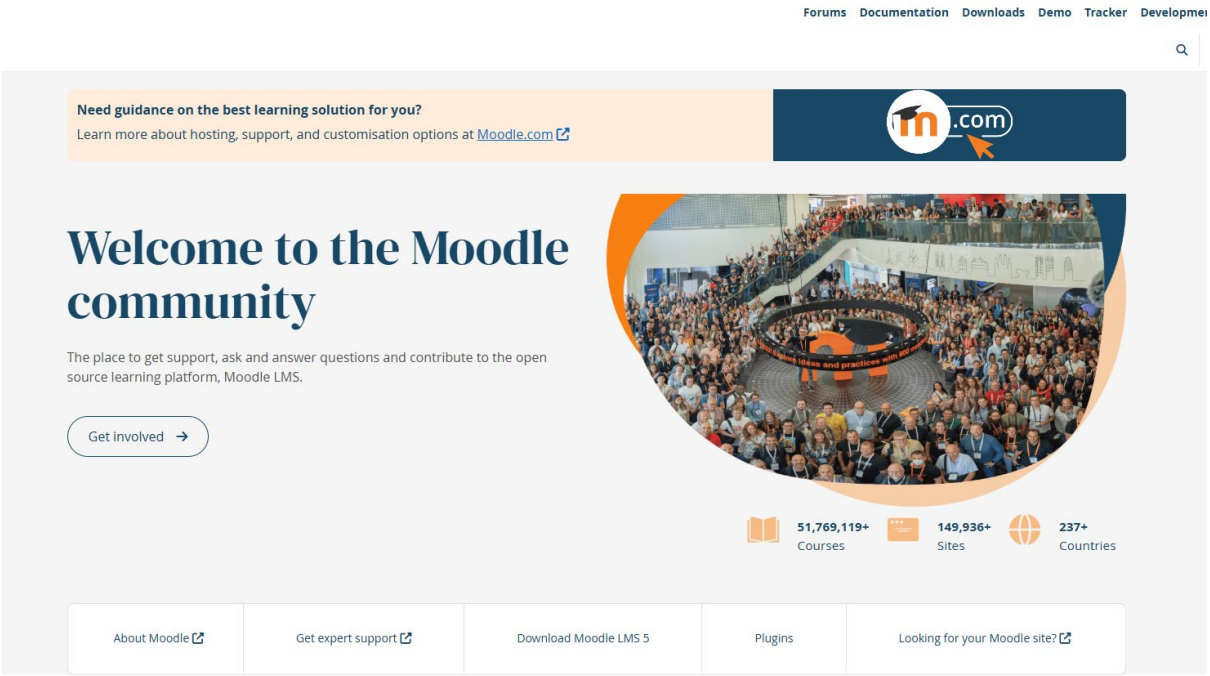

Рисунок 1.2 – Головна сторінка сайту Moodle

Ця платформа гнучка та надає користувачам інтерфейс більше ніж на 120 мовах, що дозволяє використовувати платформу у глобальному масштабі. Якщо буде необхідність встановлення власного веб-серверу з підтримкою баз даних, наприклад MySQL, то буде достатньо і мінімальних ресурсів таких як 2-ядерний процесор, 1 ГБ Оперативної пам’яті та 5 ГБ дискового простору. Проте ці

параметри можуть зростати залежно від обсягів курсу та кількості користувачів [4].

Також перевагою є доступність значної кількості тем для оформлення з можливістю зміни кольорів, шрифтів та елементів інтерфейсу. Кожному користувачу може бути призначена певні роль: учень, викладач, адміністратор або гість. Адміністратор має повний контроль над курсами, правами доступу і зовнішнім виглядом платформи. Moodle має офіційний мобільний додаток, що підтримує повноцінну взаємодію з системою як для звичайних користувачів, так і для адміністраторів. Програма працює стабільно через будь-який браузер.

1.3 Quizizz

Quizizz – це освітня онлайн-платформа типу гри-орієнтованого навчального процесу та залучення учнів через інтерактивні тести та завдання. Проєкт заснували у 2015 році в Бангалорі (Індія) двоє випускників BITS Pilani, Анкіт Гупта та Діпак Джой Чінапахті, під час викладання математики в школі: вони шукали інструмент, який би завдання цікавішими й мотивував студентів до навчання. Веб-сервіс Quizizz зображено на рисунку 1.3.

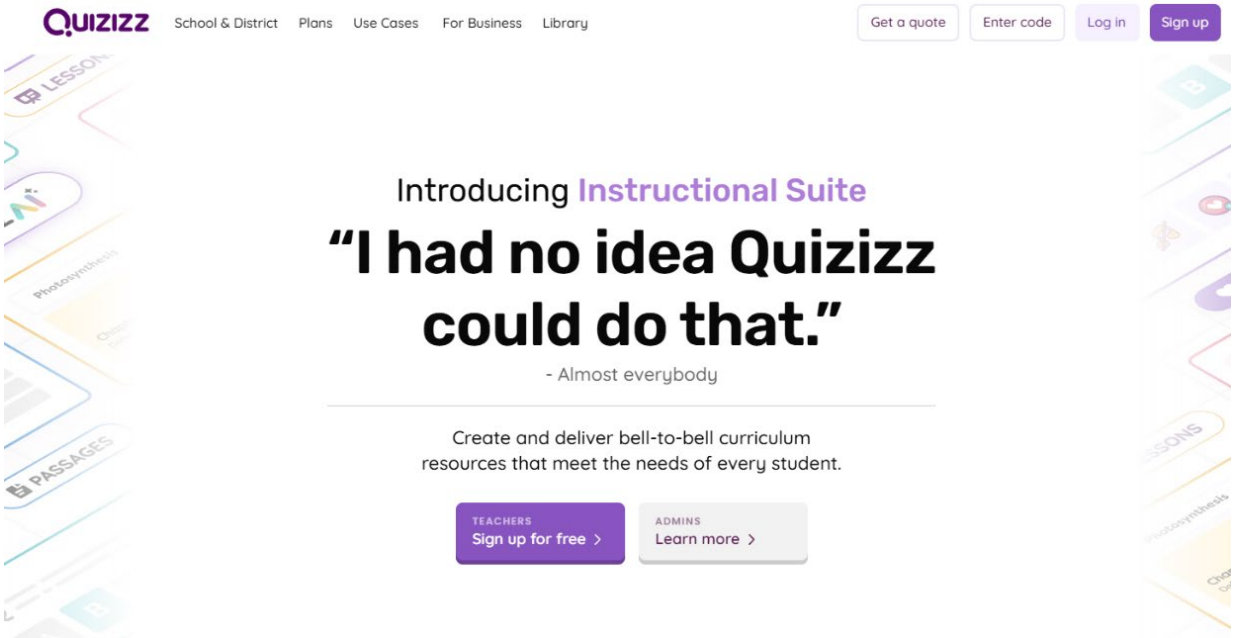

Рисунок 1.3 – Головна сторінка сайту Quizizz

Стартап швидко набрав обертів та перетворився на повноцінну міжнародну компанію з офісами в Санта-Моніці (Каліфорнія) та Бангалорі (Індія). Сьогодні Quizizz використовується в понад 150 країнах світу й підтримує мільйони студентів та викладачів. Платформа має багато пропозицій для цікавого та корисного навчання, наприклад, елементи змагання, таймери та таблиці лідерів, що робить навчання захопливим. Також платформа передбачає широкий вибір форматів питань таких як: множинний вибір, відкриті відповіді, шкали оцінювання тощо [6].

Завдяки інтеграції з Google Classroom, Microsoft Teams та іншими платформами, викладачі можуть використовувати Quizizz як аналог іншим освітнім середовищам. Сервіс підтримує не тільки шкільну та університетську програму але й корпоративне навчання, підготовку до сертифікацій та внутрішні опитування в компаніях.

1.4 ProProfs

Згідно інформації яку вдалося знайти на офіціальному сайті компанії, це багатофункціональна платформа для онлайн-навчання, яка надає інструменти для створення тестів, опитувань, курсів та навіть навчання за флеш-картками. Вона орієнтована як на освіту так і на бізнес-середовище, тобто, від шкіл і університетів до корпоративних тренінгів (рис. 1.4).

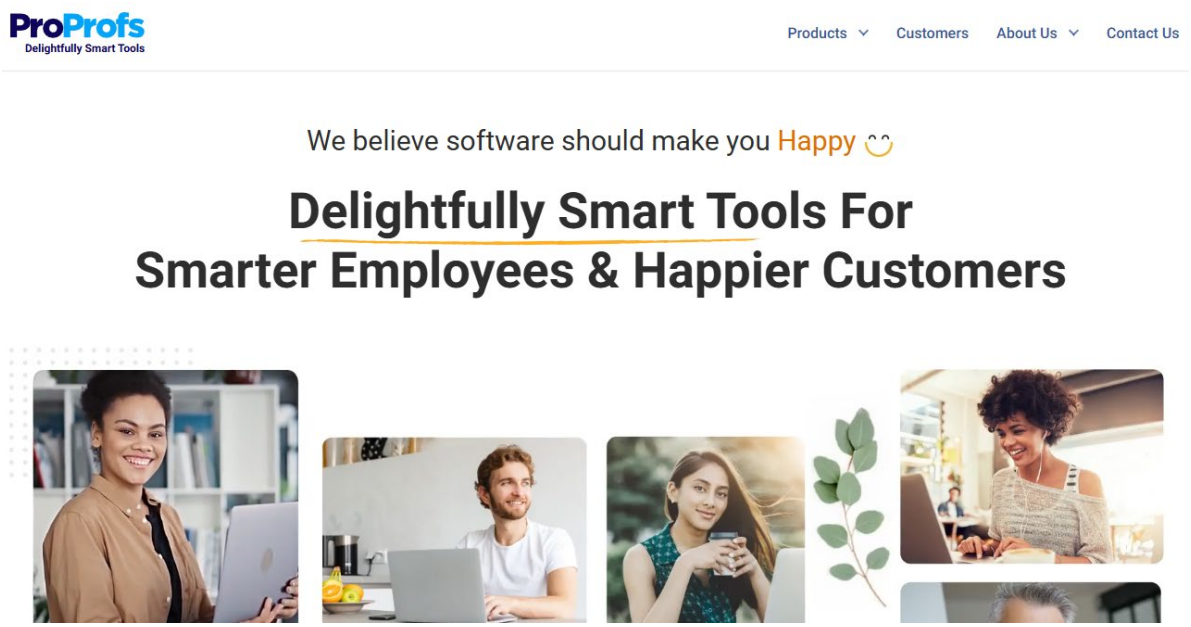

Рисунок 1.4 – Головна сторінка сайту ProProfs

За допомогою конструктора тестів (Quiz Maker) можна створювати тести з понад десятьма типами запитань, також можна додавати зображення, відео, таймери та навіть пояснення питань після тестування. Система управління навчанням, тобто LMS, дозволяє створювати повноцінні онлайн курси з відео, презентаціями, тестами та відстеженням прогресу. Ця система ідеально підходить для дистанційного навчання оскільки охоплює усі аспекти навчання.

Також присутня підтримка популярних платформ: Google Workspace, Zoom, Microsoft Teams та інші сучасні платформи. Всі продукти ProProfs працюють на мобільних пристроях, що вказують на гарну масштабованість системи, доречі, тут є адаптивний дизайн та інтерфейси для мобільних інтерфейсів [7].

Аналітика тут на високому рівні, звіти формуються за результатами тестування, також присутні такі функції: відвідуваність курсів, середній бал, час проходження тесту тощо.

1.5. Порівняння функціоналу систем

Для наочності проведено порівняльний аналіз платформ, розглянутих вище, за функціональними характеристиками (табл. 1.1).

Таблиця 1.1 – Порівняльна таблиця платформ навчання

Критерій	Google Forms	Moodle	Quizizz	ProProfs
Тип платформи	Онлайн-форма для збору даних	Система управління навчанням (LMS)	Гра-орієнтована платформа для тестування	(LMS) та конструктор тестів та опитувань
Призначення	Опитування, реєстрації, зворотній зв’язок	Повноцінне дистанційне та змішане навчання	Тести, оцінювання, залучення учнів	Курси, тести, опитування, сертифікації
Типи запитань	Вибір, текст, шкали	Багато форматів, відкриті завдання	Множинний вибір, перетягування відповідей	10+ форматів, включно з есе
Автоматичне оцінювання	Присутнє або обмежено	Так	Так	Так
Мобільність	Так	Так	Так	Так
Інтерфейс українською мовою	Так	Так	Частково	Частково
Спільна робота	Так (Кілька редакторів)	Так	Так	Так
Тарифи	Повністю безкоштовна	Безкоштовно, але хостинг окремо	Безкоштовно, платна підписка	Безкоштовна, функції обмежені

Продовження таблиці 1.1.

Плагіни та інтеграції	Мінімальні	Велика кількість плагінів	Google Classroom, Microsoft Teams, Zoom	Salesforce, Mailchimp, Zoom, SCORM
Підтримка мультимедіа	Зображення, відео	Аудіо, відео, документи	Відео, зображення в запитаннях	Аудіо, відео, документи, презентації
Аналітика і звітність	Основна статистика відповідей	Потужна система звітів	Звіти з деталізацією	Розширена аналітика з експортом в CSV, PDF
Сертифікація	Ні	Так	Немає офіційних сертифікатів	Так
Застосування	Швидкі опитування, збирання даних	Університети, школи, дистанційні курси	Школи, неформальне навчання, інтерактивні тести	Бізнес-навчання, курси з сертифікацією, корпоративний сектор

1.6 Огляд технологій веб-розробки

Веб-розробка включає використання різноманітних мов програмування, бібліотек та фреймворків, які забезпечують створення функціональних та зручних за інтерфейсом веб-сайтів. Залежно від архітектури проєкту застосовуються технології, як для серверних частин, так і для клієнтських, кожна з яких виконує свої завдання у побудові веб-додатку.

1.6.1 Python

Python – мова високого рівня програмування, яка вважається однією з найпопулярніших мов у світі. Її активно використовують у провідних компаніях: Google, Facebook, Instagram, IBM та інші. Це не дивно, бо Python цінують за простоту синтаксису, універсальність та ефективність – вона підходить для веб-

розробки, автоматизації, аналізу даних, штучного інтелекту та багатьох інших напрямках.

Python вважається інтерпретованою скриптовою об’єктно-орієнтованою мовою програмування з динамічною типізацією. Це означає, що програму можна запускати одразу після написання, без компіляції, а типи змінних визначаються автоматично. Завдяки цьому Python дозволяє швидко знаходити помилки та ефективно працювати над програмним кодом [8].

Синтаксис мови є лаконічним і зрозумілим, наприклад, для виведення простого повідомлення достатньо буде одного рядка коду, саме тому ця мова підходить для початківців, але водночас надає достатньо інструментів для складних рішень у проєктах високого рівня. Отже, можна сказати що ця мова не важка для розуміння але вона також є затребуваною, не має значення яка операційна система стоїть на комп’ютері код буде працювати як на Windows, Linux або навіть на macOS [9].

Недоліків не так багато, але вони присутні, сюди можна віднести:

- великий обсяг споживання пам’яті;

- залежність від зовнішніх бібліотек та низьку швидкість, бо послідовне виконання коду часто призводить до повільного виконання процесу.

Хоч і є певна низка недоліків, але все одно Python ще довго буде залишатися актуальною та затребуваною мовою програмування через зручне користування та написанням коду для веб-сайтів та роботи зі штучним інтелектом.

1.6.2 JavaScript

JavaScript – це мова програмування, що широко використовується для створення інтерактивних веб-сайтів. Наприклад, коли користувач наводить курсор миші на зображення або на кнопку і воно змінюється або трапляється якась інша дія то це саме JavaScript реалізовує цю поведінку. Логіка програми виглядає так, користувач взаємодіє з певним елементом сторінки, потім

активується певна подія, після цього вже запускається скрипт JS, і в кінцевому результаті відбуваються зміни в інтерфейсі залежно від заданого завдання [10].

Мова JS підтримується всіма наявними браузерами та працює на будь-якій операційній системі як і Python, що робить її універсальним інструментом для веброзробки. Щодня створюються тисячі сайтів і більшість з них мають інтерактивні елементи які реалізовані за допомогою JavaScript, саме це робить її актуальною довгі роки.

JavaScript вважається мовою новачків бо вона не важка для вивчення і розуміння процесів. Її синтаксис простий і зрозумілий, що дозволяє максимально сконцентруватися на розумінні логіки програмування, а не на тому щоб вникати в технічні деталі програми. На мові JavaScript можна організовувати (реалізовувати) багато корисних функцій серед них це: взаємодія з користувачем, обробка даних, розробка мобільних та десктопних застосунків а також створювати анімації для веб-сторінки [11].

Як і будь яка мова має свої недоліки, першою проблемою є те, що у мові JavaScript виникають труднощі з роботою типів даних, і саме помилки можна виявити тільки під час виконання програми. Другою проблемою можна впевнено назвати обмежену підтримку класичного ООП. Що мається на увазі? Тут можна повернутися у минуле, де до 2015 року JavaScript не мав повноцінної підтримки класів. Він використовував функції-конструктори замість класів, мав прототипне наслідування, а не класичне, мав динамічну типізацію, що ускладнювало пошук (виявлення) помилок і так далі [11].

У стандарті ES6 JavaScript все ж таки отримав синтаксис слів, але усередині це так і залишається прототипним наслідуванням – клас це лише надбудова для зручності.

Отже, можна зробити висновок, що ця мова програмування буде залишатися провідною багато років і її актуальність росте з кожним днем, навіть для компаній гігантів або звичайних компаній які працюють над реалізацією та вдосконалення сторінок сайту та додавання сайту візуальної краси.

1.6.3 PHP

PHP називають скриптовою мовою програмування загального призначення з відкритим вихідним кодом. Ця мова була спеціально розроблена для створення вебзастосунків та надає нам змогу вбудовувати програмний код прямо у HTML-документи. PHP широко використовується багатьма програмістами та новачками завдяки її гнучкості та простоті. На відміну від JavaScript, який виконується у браузері, PHP-код обробляється на сервері і користувач бачить тільки результат який виконаний програмою, а не сам вихідний код. Це дозволяє покращити безпеку і контролювати динамічну поведінку сайту [12].

Основною перевагою PHP є те, що його легко засвоїти навіть без досвіду програмування, водночас, мова пропонує безліч інструментів для користувача, для того щоб він навчився створювати базові шаблони (коди), а також складні і продуктивні системи та масштабовані веб-застосунки. Можна по-різному використовувати PHP, але більш поширене застосування – це створення серверних скриптів. Для цього необхідно мати PHP-інтерпретатор, вебсервер Apache або інший на розсуд користувача, а також браузер, аби встановити такий комплект не потрібно мати сучасні комплектуючі бо навіть це все легко ставиться навіть на звичайний комп’ютер.

PHP також дозволяє запускати скрипти з командного рядка, без веб-сервера, це зручно для автоматизації завдань через планувальники (cron на Linux або Task Scheduler на Windows).

Мова PHP підтримується практично всіма популярними веб-серверами. Її можна запускати як окремий процес або як модуль веб-сервера, що дає розробникам гнучкість у виборі архітектури. PHP підтримується більшістю сучасних систем включаючи BSD та Solaris. PHP підтримує процедурний, об’єктно-орієнтований стиль програмування, а також їх комбінацію, що дозволяє

створювати код будь-якої складності, від невеликих фрагментів коду до модульних та масштабованих систем.

Можливості використання цієї мови великі, бо вона має значний функціонал від попередніх мов програмування. Обробка форм, робота з cookie, сесіями, генерація не тільки HTML, але й можливість виводити PDF, зображення, JSON, XML та інші формати. Також можна додати підтримку великої кількості баз даних таких як: MySQL, PostgreSQL, SQLite, MongoDB. PHP може працювати з файлами та кешем на сервері, він бере на себе значну кількість функцій допомагаючи користувачі у плавному та логічному навчанні. Ну і остання особливість – це робота з іншими мовами наприклад, взаємодія з Java та .NET.

Отже, це універсальна мова програмування, що відіграє ключову роль у веброзробці. Вона дозволяє створювати не тільки динамічні сайти та вебзастосунки, але й взаємодіяти з базами даних, генерувати будь-які типи файлів та гнучко інтегруватися з різноманітними сервісами та технологіями. З урахуванням її широкої популярності і значного кола підтримки, і спільноти, PHP залишається актуальним інструментом у сучасній IT-галузі [12].

1.6.4 Фреймворки Django, Flask та Laravel

Django є відомим фреймворком побудованим на мові програмуванні Python. Його головна мета це спростити та пришвидшити процес створення веб-застосунків, надаючи розробнику набір готових інструментів для ефективної роботи. Завдяки Django можна створювати багаторівневі проєкти без значних зусиль. Філософією фреймворку є принцип DRY (Don`t Repeat Yourself), який зменшує дублювання коду та уникає повторення, організовуючи логіку застосунку в зручну або у структуровану схему [13].

Основними можливостями якими володіє Django це:

- підтримка створення інтерактивних та сумісних з різними браузерами інтерфейсів;

- робота з базами даних, сюди входять зберігання, обробка та керування

користувацькими даними;

- автоматична обробка запитів користувачів;
- вбудована адміністративна панель, яка допомагає редагувати вміст без знання коду;
- великий набір бібліотек і компонентів для розробки як простих сайтів так і масштабованих систем.

Django застосунок влаштований таким чином, що кожен додаток має власні шаблони, моделі, представлення та маршрути, які разом формують повноцінний веб-застосунок. У даному фреймворку є хороші переваги щоб його використовувала більшість спеціалістів:

- швидкий старт (можливий завдяки генератору шаблонів і адміністративній панелі);
- підтримка модульної структури (дає змогу розширювати застосунок поки є потреби);
- вбудовані засоби захисту від SQL -ін’єкцій, CSRF, XSS.

Щодо Flask та Laravel, ці два фреймворка відрізняються один від одного тим, що Flask – це мінімалістичний фреймворк на Python, а Laravel – потужний фреймворк на базі мови PHP. Laravel пропонує великий набір функцій для розробки додатків на мові PHP, тим часом Flask орієнтований на гнучкість завдяки модульній структурі та легкій архітектурі для розробки на Python.

Як згадувалось раніше Laravel побудований на основі PHP, і він зрозумілий для тих хто вже мав досвід роботи з цією мовою. Завдяки добре структурованому матеріалу Laravel є доступним і зрозумілим для початківців. Flask відрізняється простотою синтаксису мови Python, його структура лаконічна, без зайвих речей, що полегшує вивчення фреймворку [14].

Flask показує гарну продуктивність у невеликих проєктах або додатках. Для масштабних рішень знадобиться додаткова конфігурація для розширення, оскільки Flask не орієнтований на масштабованість. А ось Laravel через свій значний функціонал може демонструвати менш швидку продуктивність в

проєктах, но його значна перевага в тому, що він оснащений інструментами кешування та оптимізації , які дозволяють обробляти високі навантаження [14].

Laravel має свої вбудовані механізми захисту від загроз з веб-сайтів включаючи SQL-ін’єкції, XSS, CSRF. А щодо Flask то тут можна сказати, що розробнику доведеться це налаштовувати вручну або за допомогою сторонніх рішень, бо Flask забезпечує тільки базовий рівень захисту, якого не достатньо для відбиття атак.

Laravel підтримує декілька СУБД та має зручний ORM Eloquent, що полегшує роботу з базами даних і дозволяє використовувати запити в зручному режимі. Flask в цьому питанні спокійний, він підтримує роботу з будь-якими базами даних, але не має вбудованого ORM, бо розробник може використовувати SQLAlchemy або інші рішення які йому захочеться використовувати. Flask розгортається на серверах із підтримкою Python, за цього можуть виникати необхідності у налаштуванні додаткових середовищ, але забезпечує гнучкість системи. У той час Laravel розміщуються на серверах із підтримкою мови PHP. Такі провайдери як Laravel Forge та Envoyer, значно спрощують процес розгортання. Laravel популярніший ніж Flask, але попит на ці два фреймворка високі і зростають постійно [14].

1.6.5 Фронтенд-технології HTML, CSS

Оформлення сторінок сайту починається з мови HTML. Саме вона визначає логічну структуру вебсторінки та її вміст. Саме за допомогою HTML можна описувати які елементи буде містить сторінка – заголовки, текстові блоки, зображення, відео, списки, форми, таблиці тощо. Кожен елемент формується тегами, які передають браузеру, як слід візуалізувати певний фрагмент сайту. HTML виконує семантичну функцію, роблячи вміст сторінки зрозумілим для браузера [15].

Якщо HTML - це «скелет» сторінки, то CSS – це зовнішній вигляд сайту. З його допомогою можна визначити стилі елементів: кольори, шрифти, розміри,

розташування тексту та багато інших функцій. CSS забезпечує розділення структури вмісту та його візуального оформлення, що сприяє кращій організації коду та полегшує його написання. Окрім цього, CSS відповідає за адаптивний дизайн, завдяки чому вебсторінки коректно відображаються пристроях і розмірах екранів [15].

1.7 Бази даних

Бази даних відіграють важливу роль у збереженні та у обробці даних. Особливо важливим є використання систем управління базами даних (СУБД) у проєктах, пов’язаних з тестуванням, де необхідно зберігати дані користувачів, структуру тестів, відповіді та результати тестування. Для таких завдань необхідно обрати надійну, продуктивну та зручну у використанні СУБД, яка легко інтегрується з фреймворком і забезпечує масштабованість застосунку. Буде розглянуто кілька популярних СУБД , а також який є оптимальний вибір для реалізації функціоналу власної системи онлайн-тестування.

1.7.1 SQLite

SQLite є вбудованою реляційною системою управління баз даних з відкритим вихідним кодом. На відміну від інших СУБД, SQLite не потребує окремого серверу або якого складного рішення налаштування. Вона легковаговою бібліотекою, тобто яку можна інтегрувати у застосунок. SQLite зберігає всю базу даних в одному файлі на диску, завдяки її зручності в користуванні, вона добре підходить для використання у вбудованих пристроях або мобільних додатках [16].

Переваги SQLite :

- бібліотека займає мінімум дискового простору;

- простота і гнучкість;
- зручно переносити між платформами;
- чудово підходить для пристроїв які потребують локальної бази даних.

1.7.2 MySQL

MySQL – є реляційною системою управління базами даних з відкритим вихідним кодом. Вона зберігає дані у вигляді таблиць, ці таблиці можуть бути пов’язані між собою для зручного доступу та обробки інформації за допомогою мови SQL. Система MySQL має цікаву архітектуру та певні свої особливості, що робить її для різноманітних способів використання, а також для вбудованих рішень, де важливі швидкодія, надійність та масштабованість [17].

Архітектура та особливості:

- клієнт-серверна модель;
- трирівнева структура;
- ефективна робота з великими наборами даних.

Розробники часто обирають саме MySQL тому, що система має підтримку багатьох СУБД на одному сервері, це робить її особливою та зручною, особливо коли вона проста в адмініструванні. У цієї системи низьке навантаження на ресурси, та вона підтримується хостинг-провайдерами.

1.7.3 PostgreSQL

PostgreSQL – це реляційна система управління базами даних з відкритим кодом, яка працює за моделлю клієнт-сервер. Вона також використовує мову SQL для обробки та керування даними, підтримує розширені функції, включаючи транзакції, тригери, збережені процедури і повну відповідність до ACID (надійність транзакцій) [18].

PostgreSQL використовує протокол обміну повідомленнями через TCP/IP або Unix-сокети, де сервер і клієнт обмінюються повідомленнями для виконання операцій з базою даних.

Основні порти:

- TCP 5432 – стандартний порт для клієнтських підключень до PostgreSQL;

- TCP 5433 – використовуються для захищених з’єднань через SSL;

- TCP 5434, 5435 – додаткові порти для запуску декількох екземплярів PostgreSQL на одному сервері [18].

Psql – це утиліта командного рядка, яка входить до складу PostgreSQL. Дозволяє виконувати SQL-запити, створювати таблиці, керувати базами даних, а також користувачами прямо з терміналу.

1.8 Формати даних

1.8.1 CSV

CSV – це звичайний текстовий формат для зберігання табличних даних.

Особливості CSV:

- Кожен рядок (таблиці) – це окремий запис;

- Значення розділяються комою, але можуть використовуватись крапка з комою або табуляція.

- Якщо значення містить зарезервовані символи (кома, лапки) то воно береться в подвійні лапки;

- Для нецілих чисел використовують крапку (наприклад, 1356.78), але Excel може замінити її на кому залежно від налаштувань в певному регіоні.

Переваги формату CSV:

- Простий, легкий у створенні й читанні;

- Підтримується великою кількістю програм, зокрема Excel;

Недоліки формату CSV:

- формат не підтримує ієрархічну структуру;

- відсутні зв’язки між записами;

- формат призначений лише для табличних даних [19].

1.8.2 JSON

JSON – це текстовий формат для обміну даними, створений на основі синтаксису JavaScript. Ідеально підходить для зберігання структурованих ієрархічних даних [19].

Основні характеристики:

- має компактну і логічно впорядковану структуру;
- підтримує вкладені об’єкти та масиви;
- незалежний від відступів;
- він є стандартом (RFC 7159, ECMA 404);
- можливість зручно сприймати текст за допомогою розширення JSONView.

Переваги формату JSON:

- підтримує складну структуру даних;
- легко обробляється програмами та API;
- підходить для передачі даних між клієнтом і сервером.

1.9 Безпека та автентифікація у веб-додатках

Найкращими практиками безпеки є своєчасне виявлення проблем та швидка ліквідація наслідків. Усі дані які надходять від користувачів, мають бути перевірені та очищені від потенційно шкідливих елементів. Надійна автентифікація та керування сесіями може дозволити надійно захистити свій сервер або веб-сайт, якщо використовувати складні паролі та двофакторну автентифікацію (2FA). Якщо наша сесія не активно то рекомендовано її вимкнути, так буде безпечно.

Завдяки своєчасним оновленням можна уникнути прогалин в безпеці, бо застарілі бібліотеки можуть містити вже відомі вразливості, а регулярні оновлення не допустять щоб відбувались атаки на старі функції системи. Для

уникнення інших загроз користувачі повинні мати лише ті права які необхідні, це зменшить ризик а навіть шкоду у разі впровадження небезпечного програмного забезпечення. Журналювання дій та аналіз активностей які викликають підозру дозволяють виявити небажаного гостя та реагувати на спроби злому або на несанкціоновані запити.

Існує компанія, яка спеціалізується як раз таки на підвищені рівня безпеки, вона називається OWASP (Open Worldwide Application Security Project) – це незалежна організація яка зосереджена на рівні безпеки веб-застосунків. Ще у 2001 році вона об’єднала навколо себе розробників, аналітиків, фахівців з безпеки аби працювати над покращенням нашого кіберпростору та захисту у сфері веб-технологій. OWASP просуває найкращі практики з захисту та заохочує різні компанії приєднуватися до їх місії. Організація активно просуває методички з безпеки, власні інструменти та стандарти, аби компанії або самі розробники могли уникати вразливостей і почувати себе захищено від небажаних проблем. Поширені вразливості детально розписані у наступному підрозділі [20].

Автентифікація – це процес перевірки користувача на його достовірність під час спроби входу до системи або сервісу. Основна мета – це впевнитися, що особа, яка намагається отримати доступ, справді є тим, за кого себе видає. Автентифікація є базовим механізмом інформаційної безпеки, який запобігає несанкціонованому доступу до персональних чи службових даних. У повсякденному цифровому середовищі цей процес застосовується всюди – від користування смартфоном до доступу до банківських або освітніх платформ.

Важливо не плутати різницю між пов’язаними поняттями:

- Ідентифікація – це встановлення користувача за унікальним ідентифікатором, наприклад логіном або ID;

- Автентифікація – перевірка достовірності наданих облікових даних, пароля або коду підтвердження;

- Авторизація – надання або обмеження доступу до функціоналу системи після успішної автентифікації.

Ці етапи тісно пов’язані між собою але виконують різні функції в системі захисту та не є взаємозамінними.

1.9.1 Захист від SQL-ін’єкцій

SQL-ін’єкція – це тип хакерської атаки, при якій зловмисники використовують уразливості у веб-додатку для впровадження шкідливих SQL-команд. Це стає можливим, коли введення користувача не проходить належної перевірки або фільтрації. В результаті чого зловмисник може змінити або додати свій SQL-код у запити до бази даних і отримати несанкціонований доступ до інформації. Цей тип атак відомий ще з 1990-х років, коли веб-додатки почали активно використовуватися. Багато систем які були створені на базі PHP, ASP та інших технологій, мали серйозні прогалини в обробці вводу, цим і користувалися хакери [22].

Наслідки цих атак різноманітні, починаючи від крадіжки паролів, логінів, карток закінчуючи видаленням або спотворенням даних, тобто, злочинець може виконувати такі команди Delete та Drop.

Існують такі типи SQL-ін’єкцій:

1) Класичні;
2) Сліпі (Blind);
3) На основі часу (Time-based);
4) Union-ін’єкції, комбіновані методи;
5) Ін’єкції з помилками.

Класичними називають звичайні SQL-ін’єкції, вони впроваджують код в запити через поля форм. Сліпі, вони дозволяють отримувати дані, навіть якщо вони не виводяться у відповідях. На основі часу (Time-based) вони використовують затримки Sleep для аналізу результатів. Union-ін’єкції базуються на використанні SQL-оператора Union, який дозволяє об’єднувати результати кількох запитів. Комбіновані методи поєднують кілька технік одночасно-наприклад, сліпу ін’єкцію з time-based перевіркою або помилкову ін’єкцію з

Union-запитом, такі атаки небезпечні і можуть легко обійти захист, який блокує інші типи SQL-ін’єкцій [22].

Захиститися від таких атак можливо якщо дотримуватися цих пунктів:

- Очищення та валідація введення;
- Використовувати запити за параметрами;
- Використання фреймворків по типу Django;
- Обмеження доступу до БД;
- Інструменти для тестування безпеки: OWASP ZAP, Burp Suite, SQLMap.

1.9.2 Захист від XSS

XSS вважається найпоширенішою веб-уразливістю, яка дозволяє зловмисникам впровадити шкідливий код JavaScript у вебсторінку, що потім виконується у користувача. Такий код може використовуватися для викрадення особистих даних, перенаправлення на фішингові ресурси або автоматичного завантаження шкідливих файлів.

Види XSS-атак:

1) Reflected XSS (відбитий);
2) Stored XXS (збережений);
3) DOM-based XSS.

Reflected XSS працює наступним чином, шкідливий код передається через URL або форму, і браузер миттєво його виконує, щойно сторінка завантажується. Найчастіше застосовується у фішингових посиланнях. Stored XXS являє собою код який впроваджується у вебресурс, зберігається на сервері, як варіант у базі дани, а потім автоматично виконується, при кожному перегляді сторінки. DOM-based XSS, тут вразливість виникає через неправильну обробку JavaScript змінних із URL. Скрипт виконується лише в браузері без взаємодії з сервером.

Наслідки XSS-атак можуть бути серйозними як для користувачів так і для власників сайтів. В основному це крадіжки особистих даних, зокрема паролів, логінів або банківської інформації, но є і можливість натрапити на шкідливе ПЗ,

наприклад шляхом переадресації на підозрілі сайти. Також загрозою є фішинг– це створення фальшивих форм входу чи оплати з метою збору інформацію про користувача. Захиститися можна, якщо дотримуватися певних правил та принципів, наприклад перевіряти вихідні дані на правильний формат. Не треба виводити дані напряму у HTML-код , завжди треба кодувати їх аби шахраям було важко вицупити інформацію. Аби не потрапити у неприємну ситуацію також треба працювати з перевіреними API [23].

1.9.3 Захист від CSRF

CSRF – це тип кібератаки, при якому зловмисник змушує користувача, виконати небажану дію на сайті, де користувач вже авторизований, користувач цього само собою не знає. Ця атака використовує довіру вебсервера до браузера користувача.

Для боротьби з CSRF застосовується анти-CSRF-токен – це спеціальний унікальний рядок, який генерується для кожної сесії або форми. При відправці форми сервер перевіряє: чи існує токен? та чи збігається він із тим, що збережено у сесії користувача? Якщо токен неправильний або відсутній то запит відхиляється. Токен індивідуальний для кожного користувача , він зберігається на сервері в самій сесії або у cookie з обмеженим доступом.

CSRF-захист ускладнює кешування сторінок із формами бо токен має бути свіжим і унікальним. Зазвичай щоб обійти ці обмеження використовують підходи такі як: розбиття сторінки на частини та динамічне завантаження через AJAX. Ці способи активно використовуються задля безпеки в системі та уникнення зламу. [24].

1.10 Інструменти розгортання

Під час створення будь-якого веб-призначення, ключове значення мають інструменти, що забезпечують перевірку його функціональності(тестування) та успішне розгортання на сервері або у хмарному середовищі. Ці процедури дозволяють визначити якість роботи системи, виявити слабкі місця або збої та підготувати застосунок для використання реальними користувачами.

На моменті розробки найчастіше використовується локальне тестування, тобто запуск проходить на власному комп'ютері розробника. Такий підхід дає змогу швидко та оперативно виявляти, та ліквідовувати помилки завчасно уникаючи проблем, які можуть трапитись під час реалізації (завантаження) у хмарному середовищі.

Після завершення локального етапу тестування необхідно веб-застосунок розгорнути на хостингу або сервері для того щоб завантажити у відкритий доступ. Є декілька популярних платформ для розгортання сайту які ми розглянемо.

1.10.1 Heroku

Heroku – платформа яка дозволяє розміщувати застосунки без потреби в управлінні процесом та інфраструктурою. Вона підтримує різні мови програмування, що є важливим для веб-розробників, та людей які починають входити у сферу IT.

Переваги платформи Heroku:

- зручність та простота розгортання через GitHub;
- підтримка популярних баз даних PostgreSQL та Redis;
- гнучка конфігурація середовища.

Недоліки платформи:

- відсутній безкоштовний план;
- присутні обмеження за ресурсами на дешевих планах.

1.10.2 PythonAnywhere

PythonAnywhere – хмарна платформа для запуску застосунків на мові Python, зокрема Django. Вона дозволяє швидко налаштувати середовище, завантажити файли або мати синхронізацію з GitHub.

Перевагою PythonAnywhere є:

- проста інтеграція з Django та безкоштовні тарифи для освітніх проєктів;
- платформа має зручний інтерфейс для налаштування, без командного рядка;

Недоліком є те що платформа менш масштабована порівняно з Heroku , але зручні тарифи все компенсують.

2 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБ-САЙТУ

2.1 Архітектура застосунку

Основною мовою програмування обрано Python завдяки зручному синтаксису та популярності у веб-розробці, а також сумісності з фреймворком Django, а для розробки використовувалось середовище VS Code. Розробка веб-сайту реалізована з урахуванням принципів багаторівневої архітектури клієнт-серверної моделі. Сайт побудований із використанням фреймворку Django, який підтримує архітектурний шаблон Model-View-Template (MVT). Цей шаблон забезпечує чіткий поділ функціональних компонентів, де:

Модель (Model) – містить дані додатку, логіку роботи з базою даних та бізнес правила , а також забезпечує взаємодію з базою даних;

Представлення (View) – обробляє запити користувача і обирає відповідну модель для роботи з даними;

Шаблон (Template) – відповідає за відображення даних користувачу та формує HTML-код який повертається клієнту у браузер.

Клієнтська частина створена за допомогою технологій HTML, CSS та JavaScript. Це забезпечує відображення змісту у браузері користувача, а також можливість взаємодії з елементами інтерфейсу, наприклад, таймер тестування. Для підвищення зручності користування додатком застосовано фреймворк Bootstrap, що гарантує адаптивний дизайн сторінок на різних пристроях.

Архітектура веб-застосунку зображена на рисунку 2.1.

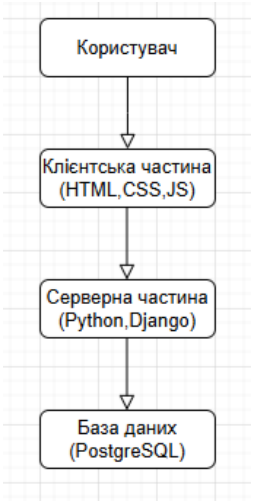

Рисунок 2.1 – Архітектура веб-застосунку для онлайн-тестування

На серверній частині Django виконує роль посередника між користувачем і базою даних. Обробка запитів здійснюється через систему маршрутизації «urls.py», яка перенаправляє запити до відповідних функцій-представлень (views). Кожне представлення звертається до відповідних моделей або повертає відповідний шаблон із динамічним вмістом.

Фреймворк Django надає потужний адміністративний інтерфейс, який дає змогу адміністратору керувати базою даних, додавати, видаляти або змінювати записи без необхідності створення окремих сторінок , це суттєво спрощує етапи розгортання та підтримки сайту.

Усі дії користувача із сайтом передбачаються як HTTP-запити, що реалізовані методом POST та GET. Django обробляє їх, використовуючи сесії, що дозволяє зберігати стан користувача, відстежувати активність. При реалізації сайту застосовано стандартні засоби безпеки, такі як CSRF-токени, для захисту від атак типу CSRF. Додатково сайт підтримує функціональність, пов’язану з реєстрацією, так само для вже зареєстрованих користувачів (логування), проходження тестів і переглядом результатів.

2.2 Моделі даних та структура БД

Для зберігання даних використовується система управління базами даних MySQL. Її обрано через надійність, простоту використання та підтримку складних типів даних, що важливо при реалізації додатку. Django використовує об’єктно-реляційне відображення (ORM), що дозволяє описувати структуру таблиць у вигляді Python-класів.

У проєкті створено такі моделі:

- User – містить дані про зареєстрованих користувачів (логін, електронна пошта, пароль, тип користувача). Django забезпечує аутентифікацію та керування сесіями за допомогою вбудованої моделі «AbstractUser» з можливістю її розширення.

- Test – тест, створений викладачем, має атрибути: назва, опис, тривалість у хвилинах, кількість спроб та автор.

- Question – запитання, яке належить до певного тесту. Поля включають в себе, текст питання, тип (одна або кілька правильних відповідей), зв’язок із тестом через ForeignKey.

- Answer – варіанти відповідей до кожного питання. Поля включають в себе, текст відповіді, правильність, зв’язок із питанням.

- Result – таблиця результатів проходження тесту. Містить інформацію про користувача, тест, дату проходження, кількість балів, правильних відповідей та загальний результат у відсотках.

Завдяки використанню ForeignKey реалізовано зв’язки між моделями. Наприклад, один тест має багато питань, одне питання має багато відповідей. Така структура забезпечує гнучкість і масштабованість даних. Усі зв’язки

реалізовані на рівні бази даних та відображаються у вигляді форм в адмін-панелі Django.

Окрім цього, у структурі є підтримка імпорту тестів за допомогою файлів CSV, що дозволяє викладачу додавати велику кількість питань та відповідей у базу. Проєкт містить окремий Python-скрипт «import_from_csv.py», який обробляє дані та додає їх до відповідних моделей. Для оптимізації запитів застосовується методи «select_related()» і «prefetch_related()», що зменшують кількість SQL-запитів до бази даних під час побудови статистики або перегляду результатів тестування, це зменшує навантаження на базу даних і підвищує загальну продуктивність веб-застосунку.

Модель Result дозволяє зберігати історію проходження тестів та виводити статистику для кожного студента або тесту. Це важливо для навчання, оскільки дозволяє аналізувати динаміку засвоєння матеріалу та виявляти типові помилки.

Таким чином, використання MySQL є технічно обґрунтованим рішенням, оскільки вона забезпечує надійність, можливість масштабування та ефективну роботу з великими обсягами даних. Завдяки цьому система може обробляти значні обсяги запитів без суттєвого навантаження на сервер.

Взаємозв’язки між ключовими елементами структури бази даних відображено на рисунку 2.2. Дана схема відображає логіку побудови та організації зберігання даних у MySQL.

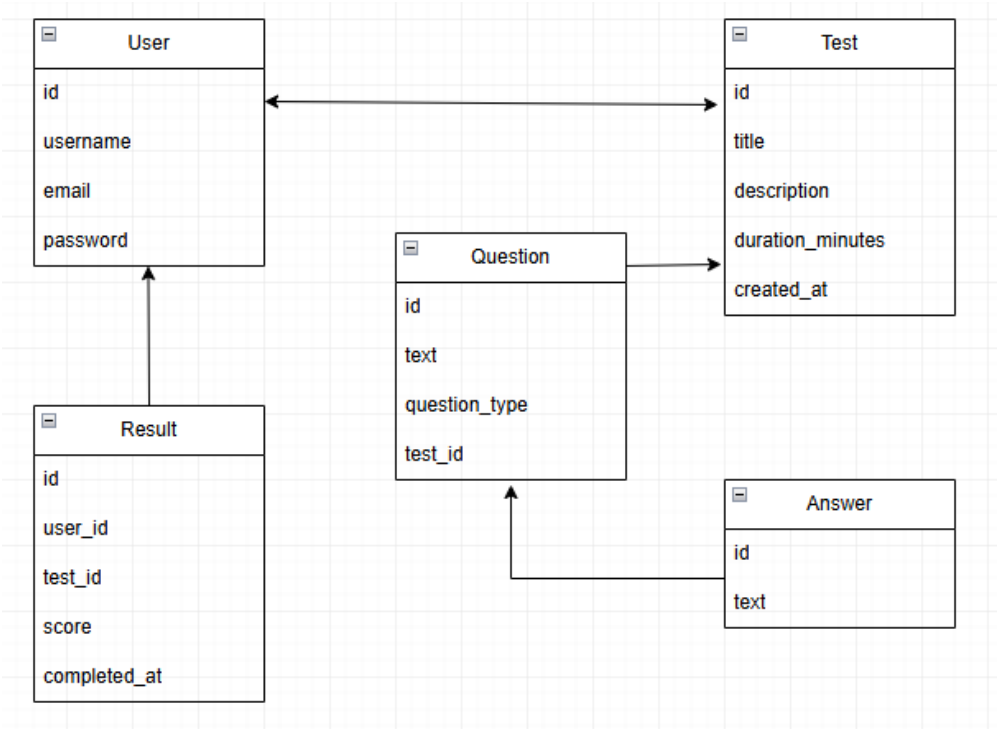

Рисунок 2.2 – Структура бази даних системи онлайн-тестування

2.3 Реєстрація та автентифікація користувачів

Один із ключових функціональних блоків системи є – модуль реєстрації та автентифікації користувачів. Його реалізація дозволяє розмежовувати права доступу між адміністраторами, викладачами та студентами, а також гарантує безпечну взаємодію з системою.

У проєкті використано базову модель AbstractUser з фреймворку Django, яка розширена додатковими атрибутами. У базі даних MySQL створена таблиця користувачів з основними полями:

- ім’я користувача;

- електронна пошта;

- пароль;

- дата створення;

- активність облікового запису;

- поле типу користувача – його ціль визначити роль користувача, наприклад, адміністратор, викладач або студент.

Дана модель зображена на рисунку 2.3.

```
class User(AbstractUser):
    ROLE_CHOICES = (
        ('student', 'Student'),
        ('teacher', 'Teacher'),
        ('admin', 'Admin'),
    )
    role = models.CharField(max_length=10, choices=ROLE_CHOICES, default='student')
```

Рисунок 2.3 – Фрагмент моделі користувача з реалізацією ролей

2.3.1 Реєстрація користувача

Сторінка реєстрації реалізована у вигляді HTML-форми, яку створено на основі Django Forms. При заповненні форми користувач вводить основні дані, які проходять перевірку дійсності, наприклад, унікальність пошти, відповідність паролю мінімальною кількості символів. У випадку успішної реєстрації, обліковий запис додається до бази і користувач автоматично входить у систему.

2.3.2 Авторизація

Для процесу авторизації застосовано стандартний механізм автентифікації «Django Authentification». Він працює через обробку POST-запиту до форми входу: у випадку успішного входу створюється сесія, яка пов’язана з користувачем, що дозволяє системі розпізнавати поточного відвідувача.

Система підтримує функції:

- вхід у систему;

- вихід із системи;

- обробка помилок при авторизації;

- захист від CSRF-атак через вбудовані механізми Django;

- перевірка прав доступу (наприклад, тільки адміністратор може додавати нові тести).

Процес авторизації через POST-запит показано на рисунку 2.4.

```python
def login_view(request):
    if request.method == 'POST':
        username = request.POST.get('username')
        password = request.POST.get('password')
        user = authenticate(request, username=username, password=password)
        if user is not None:
            login(request, user)
            return redirect('home')
        else:
            messages.error(request, 'Неправильний логін або пароль')
    return render(request, 'testing_app/login.html')
```

Рисунок 2.4 – Функція login_view для обробки авторизації користувача на веб-сайті

Нижче на рисунку 2.5 представлено фрагмент шаблону HTML-сторінки авторизації, реалізованої за допомогою мови шаблонів Django. Наведений код відповідає за формування інтерфейсу входу до системи: обробку помилок, виведення полів логіну та паролю тощо (рис. 2.5).

```
<div class="card shadow-sm p-4">
    <h4 class="mb-4">{% trans "🔐 Вхід до облікового запису" %}</h4>

    <form method="post" novalidate>
        {% csrf_token %}
        {% if form.non_field_errors %}
            <div class="alert alert-danger">
                {{ form.non_field_errors }}
            </div>
        {% endif %}

        {% for field in form %}
            <div class="mb-3">
                <label class="form-label" for="{{ field.id_for_label }}">{{ field.label }}</label>
                {{ field }}
                {% if field.errors %}
                    <div class="text-danger small">{{ field.errors|join:", " }}</div>
                {% endif %}
            </div>
        {% endfor %}

        <button type="submit" class="btn btn-primary w-100">{% trans "Увійти" %}</button>
    </form>

    <div class="text-center mt-3">
        <a href="{% url 'register' %}" class="btn btn-link">{% trans "Немає облікового запису?" %}</a>
    </div>
</div>
```

Рисунок 2.5 - HTML-шаблон входу до системи

При кожному запиті до сайту, який вимагає аутентифікації система перевіряє статус сесії. Якщо користувач неавторизований – його перенаправляють на сторінку входу, це гарантує, що доступ до захищених частин сайту таких як: статистика тесту , панель користувача, має тільки зареєстрований користувач.

2.3.3 Ролі користувачів та права доступу

В проєкті присутнє розмежування ролей для забезпечення доступу до функціоналу системи. Зроблено це для того, щоб користувач мав доступ лише до тих розділів сайту, які відповідають його повноваженням. Система підтримує наступні ролі користувачів, кожна з яких виконує визначені завдання:

- Студент – може проходити тести, переглядати свої результати;

- Викладач – створює тести, імпортує питання, переглядає результати студентів;

- Адміністратор – має повний доступ до всієї системи, включаючи управління користувачами (рис. 2.6)

```
class User(AbstractUser):
    ROLE_CHOICES = (
        ('student', 'Student'),
        ('teacher', 'Teacher'),
        ('admin', 'Admin'),
    )
    role = models.CharField(max_length=10, choices=ROLE_CHOICES, default='student')
```

Рисунок. 2.6 – Ролі користувачів

Усі паролі зберігаються в захищеному вигляді, використовуючи хеш-функції на рівні ORM Django. Такий спосіб відповідає сучасним вимогам безпеки до збереження облікових даних.

2.4 Формування тестів і питань

Формування тестів у системі реалізується через веб-інтерфейс, доступний для авторизованих викладачів або адміністраторів. Кожен тест складається з набору питань і відповідей, що зберігаються у базі даних.

У проєкті реалізовано моделі Test, Question, Answer, які пов’язані між собою. Кожне питання належить до одного тесту, а кожна відповідь – до певного питання , тож взаємозв’язки організовано через поля ForeignKey.

Додавання запитань до тесту реалізується за допомогою форми «Question Form», яка динамічно будується в залежності від запитання та дозволяє обирати одну або декілька правильних відповідей, залежно від типу запитання.

На рисунку 2.7 відображено функцію "question_view», яка відповідає за покрокове виведення запитань та обробку відповідей користувача під час проходження тесту. Після кожної відповіді дані зберігаються в сесії, а після завершення всіх питань користувача перенаправляють до результатів.

```python
@login_required
def question_view(request, test_id):
    question_ids = request.session.get('question_ids')
    current_index = request.session.get('current_question', 0)

    if current_index >= len(question_ids):
        return redirect('test_result', test_id=test_id)

    question_id = question_ids[current_index]
    question = get_object_or_404(Question, pk=question_id)
    form = QuestionForm(question=question)

    if request.method == 'POST':
        form = QuestionForm(request.POST, question=question)
        if form.is_valid():
            selected = list(form.cleaned_data['answers'].values_list('id', flat=True))
            answers = request.session.get('answers', [])
            answers.append({'question_id': question.id, 'selected': selected})
            request.session['answers'] = answers
            request.session['current_question'] = current_index + 1
            return redirect('question_view', test_id=test_id)

    return render(request, 'testing_app/question.html', {
        'question': question,
        'form': form,
        'question_number': current_index + 1,
        'total': len(question_ids),
        'time_limit': 60,
    })
```

Рисунок 2.7 – Функція «question_view» для обробки тестових питань

2.5 Реалізація таймера

Таймер є важливою частиною системи тестування, оскільки дозволяє обмежити час на проходження кожного тесту. У даному проєкті таймер реалізовано за допомогою JavaScript. Таймер стартує з моменту початку тестування. Серверна частина зберігає час початку у сесії за допомогою змінної start_time (рис. 2.7).

Клієнтська частина відповідає за зворотній відлік. Таймер виводиться у HTML-шаблоні question.html і автоматично завершує тест після закінчення часу (рис. 2.8).

```
<script>
    let seconds = {{ time_limit }};
    const timerDisplay = document.getElementById("time-left");

    const countdown = setInterval(() => {
        seconds--;
        if (seconds <= 0) {
            clearInterval(countdown);
            document.querySelector("form").submit();
        } else {
            timerDisplay.textContent = seconds;
        }
    }, 1000);
</script>
```

Рисунок 2.8 – Фрагмент коду таймера для обмеження часу тестування

Коли час завершився, форма автоматично відправляється на сервер, і користувач більше не має можливості дати відповідь. Така реалізація зумовлена для забезпечення доброчесності при проходженні тестувань.

2.6 Завантаження тестів

Система передбачає можливість завантаження тестових даних у форматі CSV-файлу, що значно полегшує процес наповнення платформи необхідними навчальними матеріалами. Доступ до певних функцій обмежено лише для авторизованих користувачів з відповідними правами, що забезпечує контроль за результатами студентів.

2.6.1 Завантаження тестів із CSV-файл

Завантаження тестів реалізоване у функції upload_csv(request), яка приймає файл із формою і зчитує його за допомогою модуля csv.DictReader. Кожен рядок CSV містить дані про тест, запитання, варіанти відповіді та ознаку правильності відповіді (рис.2.9).

```python
@login_required
@csrf_exempt
def upload_csv(request):
    message = None

    if request.method == 'POST':
        file = request.FILES.get('csv_file')
        if not file or not file.name.endswith('.csv'):
            message = "❌ Файл має бути у форматі CSV."
        else:
            decoded_file = file.read().decode('utf-8').splitlines()
            reader = csv.DictReader(decoded_file)

            current_test = None
            for row in reader:
                test_title = row.get('test_title')
                question_text = row.get('question')
                answer_text = row.get('answer')
                is_correct = row.get('is_correct', '').strip().lower() in ('1', 'true', 'так')

                if test_title and (not current_test or current_test.title != test_title):
                    current_test, _ = Test.objects.get_or_create(title=test_title)

                if current_test and question_text and answer_text:
                    question, _ = Question.objects.get_or_create(test=current_test, text=question_text)
                    Answer.objects.get_or_create(question=question, text=answer_text, is_correct=is_correct)

            message = "✅ Імпорт завершено."

    return render(request, 'testing_app/upload_csv.html', {'message': message})
```

Рисунок 2.9 – Фрагмент коду для завантаження тестів формату CSV-файл

2.6.2 Експорт результатів

Експорт результатів реалізований як окрема функція, доступна лише адміністраторам. Захист здійснюється за допомогою @staff_member_required, який гарантує, що лише користувачі з відповідними правами мають можливість користуватися функціональністю вивантаження результатів з сайту, що надає право на виконання лише тим користувачам, які мають ознаку «staff» у системі.

Функція export_results(request) створює CSV-файл зі зведеною статистикою для кожного користувача (рис. 2.10).

```
@staff_member_required
def export_results(request):
    response = HttpResponse(content_type='text/csv')
    response['Content-Disposition'] = 'attachment; filename="user_results.csv"'

    writer = csv.writer(response)
    writer.writerow(['Користувач', 'Тест', 'Бали', 'Загальна кількість', 'Час завершення', 'Тривалість'])

    for result in UserResult.objects.select_related('user', 'test'):
        writer.writerow([
            result.user.username,
            result.test.title,
            result.score,
            result.total,
            result.completed_at.strftime('%Y-%m-%d %H:%M:%S'),
            str(result.duration) if result.duration else ''
        ])

    return response
```

Рисунок 2.10 - Функція export_results(request) у файл user_results.csv

Файл експорту можна відкрити у будь-якому табличному редакторі, наприклад у Excel для подальшого аналізу або архівування. Такий підхід дозволяє швидко отримати повну статистику по всім користувачам та тестам у зручному форматі.

2.7 Збереження результатів і журналювання

Після проходження тесту система автоматично формує та зберігає результати у базі даних. Для цього використовується модель UserResult, яка містить повну інформацію про користувача, тест, отримані бали, загальну кількість питань, час завершення тесту та тривалість виконання.

Збереження здійснюється у функції test_result(request, test_id), яка обробляє відповіді користувача після проходження тесту. Ця функція порівнює вибрані відповіді з правильними, обчислює підсумковий бал, визначає тривалість проходження тесту та створює відповідний запис у таблиці UserResult. Фрагмент коду наведений нижче на рисунку 2.11.

```
UserResult.objects.create(
    user=request.user,
    test=test,
    score=score,
    total=total,
    completed_at=now(),
    duration=duration
)
```

Рисунок 2.11 – Створення запису результату проходження тесту

Даний допис дозволяє зберегти повну інформацію про спробу проходження тесту. Поле completed_at=now() фіксує поточний час завершення, а duration обчислюється на основі часу старту, збереженого в сесії та поточного моменту. Завдяки цьому адміністратор або викладач може оцінити не тільки якість виконання але й швидкість відповіді.

Результати зберігаються на постійній основі і доступні для перегляду в особистому кабінеті користувача, а також можуть бути експортовані адміністратором у CSV-файл.

2.8 Опис шаблонів інтерфейсу користувача

У веб-застосунку було реалізовано систему шаблонів на основі механізму Django Templates. Усі сторінки інтерфейсу користувача створені з використанням мови HTML та стилізовані за допомогою CSS-фреймворку Bootstrap. Цей фреймворк дозволяє досягти сучасного та адаптивного дизайну, який коректно відображається як на комп’ютерах, так і на мобільних пристроях.

Кожна сторінка сайту наслідує базовий шаблон base.html, який містить загальні компоненти: шапку сайту, навігаційне меню, базові стилі. У шаблонах використано стандартні блоки для динамічного підключення контенту.

Наявні сторінки інтерфейсу користувача:

- login.html – форма входу до облікового запису;

- register.html – форма реєстрації нового користувача;

- home.html – головна сторінка с доступними тестами;

- question.html – сторінка проходження тесту;

- result.html – сторінка результатів після завершення тестів;

- profile.html – особистий кабінет користувача з відображенням статистики.

На рисунку 2.12 зображено HTML-шаблон сторінки входу до системи. Даний шаблон реалізує повноцінну форму з динамічним відображенням полів введення, перевірки відповідності та виведенням повідомлень про помилки.

```
<div class="card shadow-sm p-4">
    <h4 class="mb-4">{% trans "🔐 Вхід до облікового запису" %}</h4>

    <form method="post" novalidate>
        {% csrf_token %}
        {% if form.non_field_errors %}
            <div class="alert alert-danger">
                {{ form.non_field_errors }}
            </div>
        {% endif %}

        {% for field in form %}
            <div class="mb-3">
                <label class="form-label" for="{{ field.id_for_label }}">{{ field.label }}</label>
                {{ field }}
                {% if field.errors %}
                    <div class="text-danger small">{{ field.errors|join:", " }}</div>
                {% endif %}
            </div>
        {% endfor %}

        <button type="submit" class="btn btn-primary w-100">{% trans "Увійти" %}</button>
    </form>

    <div class="text-center mt-3">
        <a href="{% url 'register' %}" class="btn btn-link">{% trans "Немає облікового запису?" %}</a>
    </div>
</div>
```

Рисунок 2.12 – Фрагмент шаблону login.html

2.9 URL-маршрутизація та обробка запитів

У Django веб-застосунок побудовано на основі чіткого механізму маршрутизації, що дозволяє зв’язати URL-адреси з відповідними функціями або класами обробки запитів. Для цього використовується файл urls.py, в якому визначаються маршрути та прив’язуються до представлень «views», що реалізують бізнес-логіку сайту.

Кожен маршрут описується через функцію path(), де задається шлях, функція-обробник та ім’я маршруту (рис.2.13).

```python
from django.urls import path
from . import views
from .views import LogoutWithMessageView, login_view

urlpatterns = [
    path('', views.home, name='home'),
    path('register/', views.register, name='register'),
    path('login/', login_view, name='login'),
    path('logout/', LogoutWithMessageView.as_view(), name='logout'),
    path('profile/', views.profile, name='profile'),
    path('my-results/', views.my_results, name='my_results'),

    path('test/<int:test_id>/start/', views.start_test, name='start_test'),
    path('test/<int:test_id>/question/', views.question_view, name='question_view'),
    path('test/<int:test_id>/result/', views.test_result, name='test_result'),

    path('upload-csv/', views.upload_csv, name='upload_csv'),
    path('export-results/', views.export_results, name='export_results'),
    path('contact/', views.contact, name='contact'),
]
```

Рисунок 2.13 – Файл urls.py із маршрутизацією основних функцій

Ці маршрути дозволяють користувачу переміщуватись між сторінками сайту, надсилати дані форм, відкривати окремі тести тощо. Запити типу GET використовуються для відображення сторінок, а POST – для надсилання даних, таких як форма входу або відповіді на тест (рис.2.13)

Для обмеження доступу використовується декоратор @login_required, який не дозволяє відкривати певні маршрути неавторизованим користувачам. Приклад обробки POST-запиту наведений на рисунку 2.14.

```python
def login_view(request):
    if request.method == 'POST':
        username = request.POST.get('username')
        password = request.POST.get('password')
        user = authenticate(request, username=username, password=password)
        if user is not None:
            login(request, user)
            return redirect('home')
        else:
            messages.error(request, 'Неправильний логін або пароль')
    return render(request, 'testing_app/login.html')
```

Рисунок 2.14 - Обробка POST-запиту для автентифікації користувач

3 ТЕСТУВАННЯ РОБОТИ САЙТУ

3.1 Планування процесу тестування

Після завершення етапу проєктування треба перевірити працездатність, функціональність та стабільність сайту. Основною метою є виявлення помилок, оцінка відповідності реалізованого функціоналу системи, а також забезпечення надійної роботи веб-додатку в умовах реального використання.

Оскільки система є веб-застосунком, що взаємодіє з базою даних та іншими системними компонентами, тестування системи охоплює кілька рівнів:

- модульне тестування функціональних компонентів – має мету перевірити, чи працюють коректно окремі частини коду (моделі, класи, функції);

- інтеграційне тестування взаємодії частин системи – тобто, перевірка взаємодії між модулями, наприклад, чи правильно передаються дані з форми авторизації до сервера і чи повертається відповідь;

- функціональне тестування сценаріїв користувача – основна ціль впевнитися, що сайт виконує ті функції, які від нього очікуються, відповідно до вимог. Наприклад, користувач реєструється, проходить тест, а потім система відображає оцінку;

- тестування безпеки (авторизація права доступу) – мета цього рівня полягає в тому, щоб надати гарантії, що незареєстровані або неавторизовані користувачі не матимуть доступу до захищених функцій;

- перевірка на сумісність з браузерами та пристроями – мета, перевірити, чи інтерфейс сайту працює однаково у браузерах і на різних пристроях;

- ручне тестування інтерфейсів – тобто, це самостійна перевірка елементів інтерфейсу, за яким адміністратор може перевірити функції кнопок або форм.

3.2 Сценарії перевірки функціональності

У системі реалізовано кілька ключових функцій, які були перевірені за допомогою ручного тестування. Для кожної з функцій було складено сценарії, що відтворюють типову поведінку користувачів

3.2.1 Реєстрація та автентифікація

Тестувались наступні випадки:

- успішна реєстрація користувача;

- повторна реєстрація з тією самою поштою (очікувана помилка);

- авторизація з правильними даними;

- спроба входу з неправильним паролем;

- обмеження доступу до захищених сторінок;

- перенаправлення користувача після входу на головну сторінку.

Приклад реєстрації користувача наведений на рисунку 3.1.

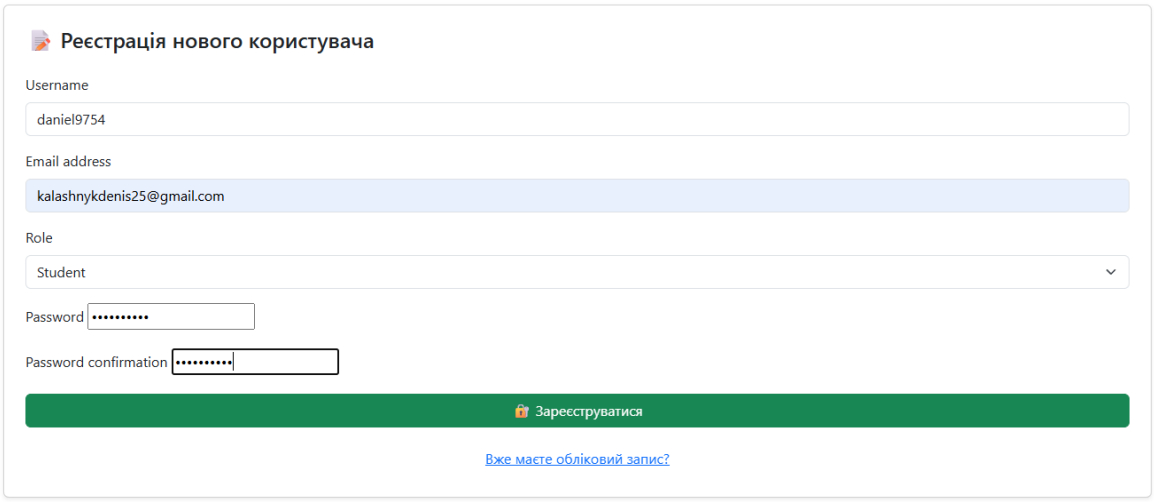

Рисунок 3.1 – Процес реєстрації нового користувача

3.2.2 Тестування функціоналу веб-сайту

Було перевірено наступний функціонал системи:

- імпорт запитань із CSV-файлу;

- відображення тесту на головній сторінці користувача;

- проходження тесту з фіксацією відповідей;

- завершення тесту та запис результатів;

- виведення статистики у профілі користувача.

На рисунках 3.2-3.5 відображено процес тестування користувача.

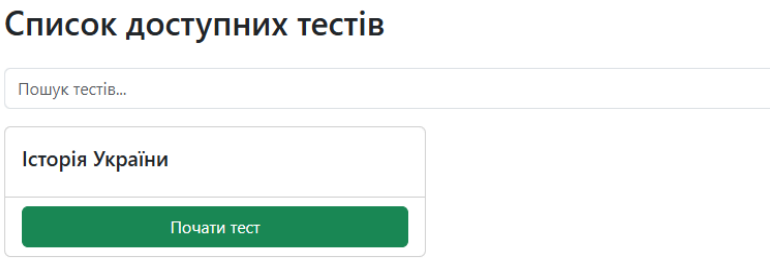

Рисунок 3.2 – Сторінка сайту «Список доступних тестів»

Після того як користувач обрав певний тест його перекидає на сторінку тесту і починається відлік таймеру для проходження тесту де у нього є 60 секунд на відповідь. Якщо користувач не встигне обрати відповідь за наданий час то система автоматично перекине його на наступне питання. Процес тестування (рис.3.3).

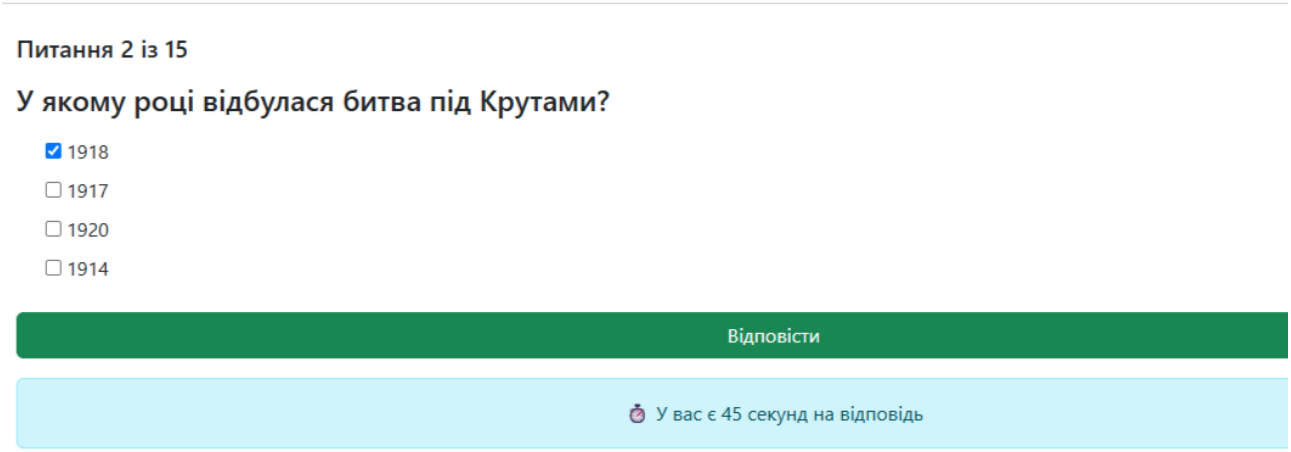

Рисунок 3.3 – Процес тестування студента

Коли студент (користувач) закінчить тестування, йому буде надіслано автоматичне повідомлення про успішне завершення тестування, і він зможе подивитися свої бали за пройдений тест (рис. 3.4).

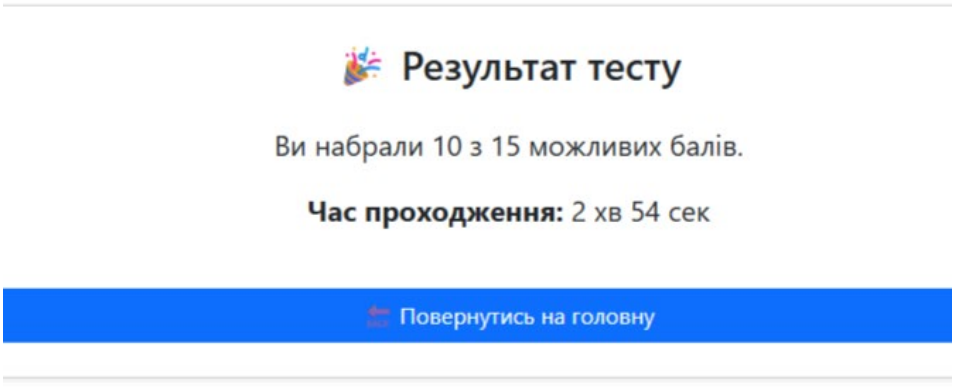

Рисунок 3.4 – Результат тестування студента

Також користувач може подивитися свої результати у власному профілі, який створено для зручності тестування, а також для візуального покращення сторінки сайту. Даний рисунок відображає сторінку результатів користувача з датою проходження тесту та відповідно отриманими балами (рис.3.5).

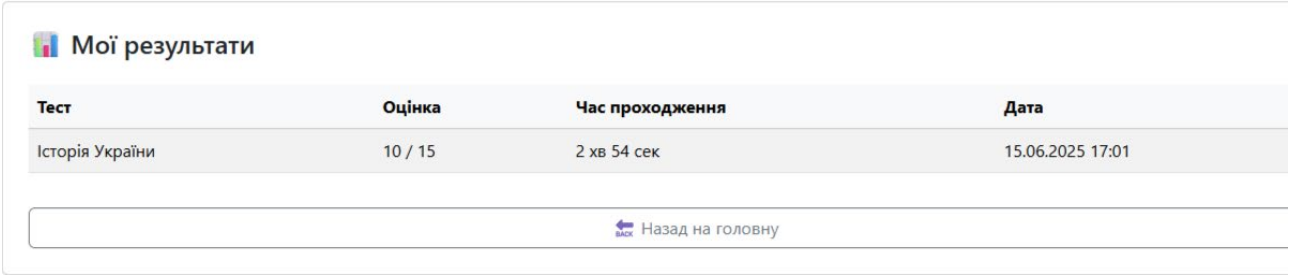

Рисунок 3.5 – Сторінка результатів студента

Усі перевірки дали гарні результати. Було зафіксовано правильну роботу таймера, збереження відповідей після кожного питання та виведення підсумкової оцінки. Результати виводяться у профіль користувача.

3.3 Тестування ролей та прав доступу

В системі реалізовано три основні ролі, які обмежують права користувача в залежності від його ролі:

- Адміністратор – має доступ до абсолютно всіх функцій системи;

- Викладач – створює та керує тестами;

- Студент – проходить тестування та має змогу переглянути власні результати.

Тестування прав доступу мала мету перевірити, що студент не зможе редагувати тести, а також, що викладач не буде мати доступ до панелі адміністратора.

3.4 Перевірка функціональності інтерфейсу

Інтерфейс було протестовано вручну в браузерах Google Chrome та Microsoft Edge. У обох випадках інтерфейс відображався так як було треба, елементи не виходили за межі екрану, а адаптивність за допомогою Bootstrap дозволяла зручно працювати з мобільних пристроїв. Тестування проводилось на комп’ютері з роздільною здатністю екрану 1920×1080 та на мобільному пристрої з роздільною здатність 2340×1080.

3.5 Тестування імпорту CSV

Важливим аспектом функціональності системи є можливість імпортувати запитання та відповіді з CSV-файлом. У ході перевірки функціональності було здійснено:

- завантаження файлу з правильною структурою;

- перевірку виведення повідомлення про успішний імпорт;

- спробу імпорту з неправильним форматом (очікувана помилка);

- дублювання запитань та їх ігнорування.

Імпортувати файли CSV мають змогу лише адміністратори або викладачі. Для студентів дана функція обмежена оскільки користувачі з роллю Student не мають відповідних прав доступу. На рисунку 3.6 показано головну сторінку з елементом «Завантажити CSV».

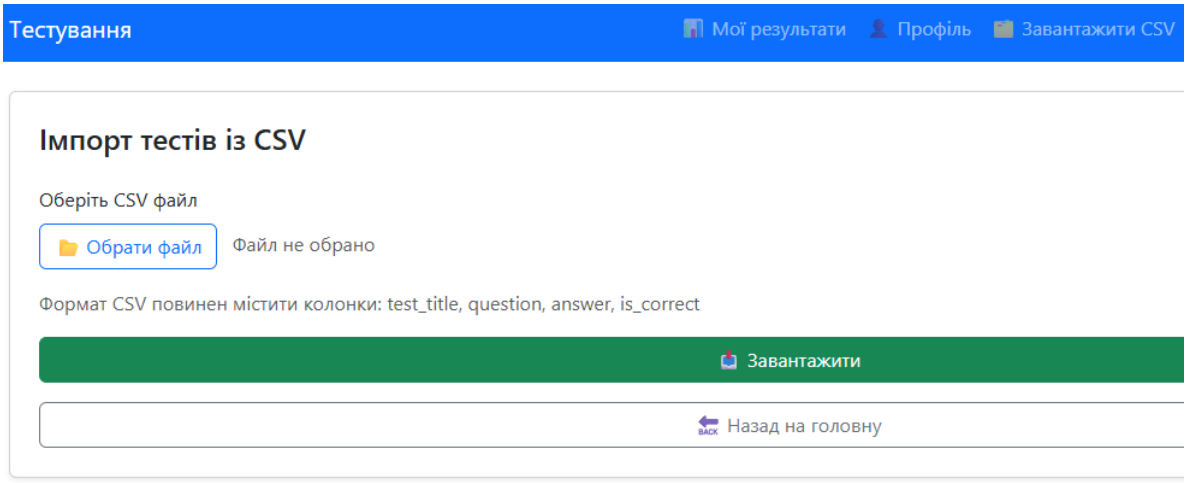

Рисунок 3.6 – Функціонал імпорту тестів CSV-файлів

3.6 Журнал активностей

Система була розроблена за допомогою стандартного модуля logging. Основні події, які фіксуються:

- вхід/вихід користувача;

- створення або редагування тестів;

- імпорт файлів;

- проходження тесту;

- помилки при автентифікації.

Перелічені журнали дозволяють адміністратору виявляти як несанкціоновані спроби авторизації так і діагностувати проблеми, які виникають у користуванні платформою. Адміністратор має доступ до збережених результатів користувачів, що дозволяє контролювати успішність проходження тестів та переглядати деталі виконання. Результати автоматично фіксуються у базі даних після завершення тесту (рис.3.7).

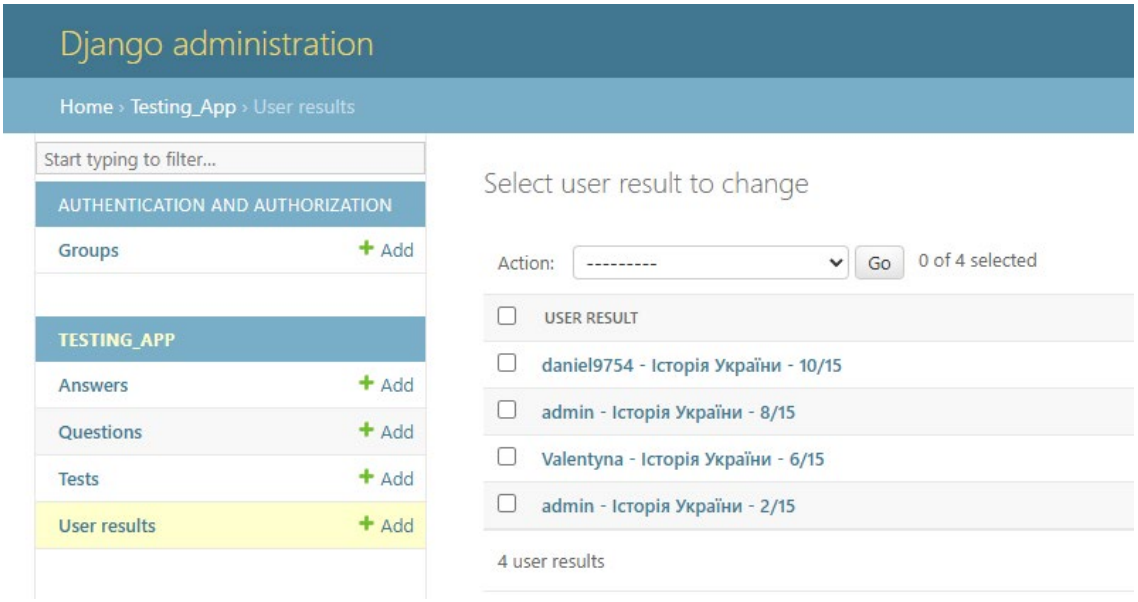

Рисунок 3.7 – Перегляд результатів тестування через панель адміністратора

3.7 Рекомендації для подальшого покращення

У подальшому удосконаленні веб-сайту варто зосередити увагу на наступних напрямах:

- розширити статистику тестувань для викладачів та адміністраторів;

- додати покращення інтерфейсу, наприклад, темну тему, адаптивну верстку для мобільних пристроїв;

- реалізувати вдосконалену форму відгуків, що дозволить користувачам надсилати пропозиції щодо покращень сайту, так і для повідомлень про помилки;

- реалізувати систему сповіщень через електронну пошту;

- реалізувати резервне копіювання даних для зменшення ризиків втрати інформації;

- зробити можливість перейти на більш масштабовану систему управління базами даних.

ВИСНОВКИ

Метою дипломного проєкту було створення власного веб-сайту для онлайн-тестування.

За результатами аналізу сучасних систем тестування було обрано оптимальний варіант для розробки веб-сайту. В якості середовища для розробки застосунку було обрано VS Code. Сайт створений мовою Python, оскільки вона має широку популярність, підтримує велику кількість бібліотек та фреймворків для розробки веб-додатків. Для спрощення розробки обрано фреймворк Django, який забезпечує високий рівень безпеки, гнучкість та можливість швидкої розробки за допомогою вбудованих компонентів.

Для надійного рішення зберігання даних використано базу даних MySQL. Під час проєктування клієнтської частини було обрано HTML, CSS та JavaScript для формування інтерфейсу користувача, забезпечення адаптивності та зручного відображення сторінок. Архітектура та структура бази даних обрані з урахуванням вимог і особливостей проєктування.

Розроблений веб-сайт протестований на функціональність, зручність та простоту використання. Сайт дозволяє розгортати проєкт на більшій системі управління базами даних, оскільки архітектура проєкту та рішення стосовно обраних компонентів розробки надають таку можливість.

ПЕРЕЛІК ПОСИЛАНЬ

1. Google Forms [Електронний ресурс] – Режим доступу: https://Https://wedex.com.ua/blog/google-forms-shho-cze-ta-yak-stvoryty/

2. Google форми - від простого до найпростішого [Електронний ресурс] – Режим доступу: https://elit-web.ua/ua/blog/google-forms

3. Що таке Moodle? [Електронний ресурс] – Режим доступу: https://moodle.org/mod/page/view.php?id=8174

4. Що таке платформа Moodle [Електронний ресурс] – Режим доступу: https://hostpro.ua/blog/ua/what-is-the-moodle-platform/

5. Quizizz [Електронний ресурс] – Режим доступу: https://vchymo.com/app/application/Quizizz

6. Сім платформ для створення тестів [Електронний ресурс] – Режим доступу: https://osvita.ua/school/method/technol/45747/

7. ProProfs [Електронний ресурс] – Режим доступу: https://www.proprofs.com/

8. Що таке мова програмування Python? [Електронний ресурс] – Режим доступу: https://freehost.com.ua/ukr/faq/wiki/chto-takoe-jazik-programmirovanija-python/

9. Що таке Python і де він використовується [Електронний ресурс] – Режим доступу: https://dan-it.com.ua/uk/blog/python-chto-jeto-za-jazyk-programmirovanija-i-gde-ego-ispolzujut/#__Python-3

10. JavaScript - краща мова для програмування [Електронний ресурс] – Режим доступу: https://apeps.kpi.ua/javascript-krashcha-mova-dlia-programuvania

11. Мова програмування JavaScript: Ідеальний вибір для початківців [Електронний ресурс] – Режим доступу: https://point.te.ua/programming/programming-language-javascript-the-perfect-choice-for-beginners/

12. Що таке PHP та що з цим можна робити? [Електронний ресурс] – Режим доступу: https://www.php.net/manual/uk/introduction.php

13. Що таке Django і як з ним працювати в Python [Електронний ресурс] – Режим доступу: https://highload.tech/uk/shho-take-django-i-yak-z-nym-pratsyuvaty-v-python/.

14. Laravel vs Flask: Choosing the Right Framework for Web Development [Електронний ресурс] – Режим доступу: https://www.sinelogix.com/laravel-vs-flask/

15. Що таке фронтенд розробка: складові, етапи та технології [Електронний ресурс] – Режим доступу: https://wezom.com.ua/ua/blog/chto-takoe-front-end-razrabotka

16. SQLite [Електронний ресурс] – Режим доступу: https://www.guru99.com/uk/sqlite-tutorial.html

17. База даних MySQL [Електронний ресурс] – Режим доступу: https://promoter.net.ua/articles/baza-danix-mysql.html

18. Мова структурованих запитів PostGres (PostgreSQL) [Електронний ресурс] – Режим доступу: https://cqr.company/ua/wiki/protocols/postgres-structured-query-language/

19. Формат даних [Електронний ресурс] – Режим доступу: https://www.data.in.ua/pohovorymo-pro-formaty-danykh/

20. OWASP - забезпечення безпеки веб-додатків [Електронний ресурс] – Режим доступу: https://foxminded.ua/owasp-tse/

21. Автентифікація (аутентифікація): що це, методи та особливості [Електронний ресурс] – Режим доступу: https://www.zen.com/ua/blog/personal-finance-uk/authentication-definition-methods-features/

22. SQL ін’єкції та захист від них [Електронний ресурс] – Режим доступу: https://foxminded.ua/sql-iniektsii/

23. Що таке XSS атака і які є методи захисту? [Електронний ресурс] – Режим доступу: https://foxminded.ua/xss-ataka/

24. Як реалізувати CSRF-захист [Електронний ресурс] – Режим доступу: https://symfony.com.ua/doc/current/security/csrf.html