

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра комп'ютерних систем та мереж

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістра

(ступінь вищої освіти)

на тему ІНТЕЛЕКТУАЛЬНА СИСТЕМА ТЕСТУВАННЯ ВОДІЇВ

(назва теми)

Виконав(ла): студент(ка) 2 курсу, групи КНТ-522м

Спеціальності 123 Комп'ютерна інженерія

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні системи та мережі

ШМИГЛЯ К.В.

(ПРИЗВИЩЕ та ініціали)

Керівник КУДЕРМЕТОВ Р.К.

(ПРИЗВИЩЕ та ініціали)

Рецензент КАПЛІСНКО Т.І.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет Комп'ютерних наук та технологій
 Кафедра Комп'ютерні системи та мережі
 Ступінь вищої освіти магістерський
 Спеціальність 123 Комп'ютерна інженерія
(код і найменування)
 Освітня програма (спеціалізація) Комп'ютерні системи та мережі
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри Кудерметов Р.К.

“24” жовтня 2023 року

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

ШМИГЛЯ Катерина Вікторівна

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Інтелектуальна система тестування водіїв
 керівник проєкту (роботи) к.т.н., доцент, КУДЕРМЕТОВ Равіль Камілович
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “24” жовтня 2023 року № 400

2. Строк подання студентом проєкту (роботи) 13 грудня 2023 року

3. Вихідні дані до проєкту (роботи) наявні методи покращення рівня знань
 та алгоритми визначення прогресу в засвоєні теоретичного матеріалу

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналіз предметної області;

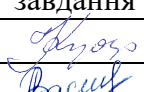
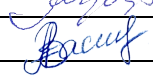
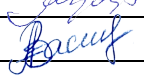
2. Проєктування та моделювання системи;

3. Реалізація системи;

4. Результати роботи.

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

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	КУДЕРМЕТОВ Р. К., доцент		
нормоконтроль	ПОЛЬСЬКА О.В., ст. викл.		

7. Дата видачі завдання 01.10.2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз наявних аналогів, визначення завдань у системі, визначення програмного забезпечення	10.10.2023	
2	Створення алгоритму функціонування проєкту, ER-моделі, семантичної мережі	14.10.2023	
3	Створення діаграми послідовностей та обрання принципів побудови інтерфейсу	17.10.2023	
4	Реалізація системи тестування	11.11.2023	
5	Створення інтерфейсу користувача	18.11.2023	
6	Тестування та експериментальне дослідження сайту	25.11.2023	
7	Оформлення результатів у ПЗ	01.12.2023	

Студент(ка)


 (підпис)

Катерина ШИМИГЛЯ

(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)


 (підпис)

Равіль КУДЕРМЕТОВ

(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

ПЗ: 73 с., 55 рис., 6 табл., 20 джерел.

ЗАПИТ, ІНТЕРФЕЙС, ПДР, DOM, CSS, HTML, SQL, PHP, UML

Об'єкт дослідження – засоби підготовки до складання теоретичного екзамену правил дорожнього руху.

Предмет дослідження – методи створення інтелектуальної системи на основі семантичної мережі.

Мета роботи – створення інтелектуальної системи тестування водіїв, щоб допомогти підвищити рівень знань кандидатів у водії, завдяки врахуванню помилок за темами у попередніх тестах і пропонуванню більшої кількості запитань за цими темами.

У першому розділі розглянуто аналоги вебсайтів для тестування водіїв, визначено основні вимоги для розробки проєкту, визначено основні типи завдань в системі та обрано програмне забезпечення.

У другому розділі виділено основні сценарії роботи авторизованого і неавторизованого користувача, описано основні послідовності, спроектовано ER-модель та семантичну мережу, виконано вибір засобів для розробки сайту.

У третьому розділі наведено реалізацію системи з використанням СУБД MySQL та phpMyAdmin для її адміністрування, а для взаємодії з даними – PHP, HTML, CSS та jQuery.

У четвертому розділі наведено результати роботи у вигляді прикладів основних сторінок сайту.

В результаті виконання роботи розроблено сайт з ПДР тестуваннями, що дозволяє проходити тестування, переглядати результати пройденого тестування та переглядати теорію. Також для авторизованого користувача відбувається аналіз проблемних тем, за якими в наступних тестах буде надано більше питань.

ABSTRACT

Explanatory note to the master's work: 73 p., 55 figures, 6 tables, 20 sources.

QUERY, INTERFACE, SDA, DOM, CSS, HTML, SQL, PHP, UML

Object research – means of preparation for passing the theoretical examination of traffic rules.

The subject of research is methods for creating an intelligent system based on a semantic web.

The purpose of the work is creating an intelligent driver testing system to help improve the knowledge of driver candidates by taking into account mistakes on topics in previous tests and offering more questions on these topics.

The first section reviews analogs of websites for testing drivers, identifies the main requirements for project development, defines the main types of tasks in the system, and selects software.

The second section identifies the main scenarios of an authorized and unauthorized user, describes the main sequences, designs an ER model and a semantic network, and selects tools for website development.

The third section describes the realization of the system using the MySQL database and phpMyAdmin for its administration, and PHP, HTML, CSS, and jQuery for data interaction.

The fourth section presents the results of the work in the form of examples of the main pages of the site.

As a result of the work, a website with SDA tests was developed, which allows to take tests, view the results of the test and review the theory. The authorized user can also analyze problematic topics for which more questions will be provided in the next tests.

ЗМІСТ

Скорочення та умовні позначки	7
Вступ.....	8
1 Аналіз предметної області	10
1.1 Аналіз вебсайтів для тестування водіїв	10
1.2 Результати аналізу.....	12
1.3 Типи завдань у системі.....	14
1.4 Програмне забезпечення	15
2 Проєктування та моделювання системи.....	16
2.1 Визначення функціональних вимог до проєкту.....	16
2.2 Алгоритм функціонування проєкту	17
2.3 ER-модель бази даних	24
2.4 Семантична мережа	28
2.5 Діаграми послідовності	30
2.6 Розгорнуті описи прецедентів	33
2.7 Обрані принципи побудови інтерфейсу користувача	37
2.8 Вибір засобів проєктування і розробки	40
3 Реалізація системи.....	41
3.1 Реляційна модель даних	41
3.2 Взаємодія з базою даних	42
3.3 Авторизація користувача.....	45
3.4 Відслідковування прогресу.....	46
3.5 Відбір питань	48
3.6 Використання jQuery	52
3.7 Реалізація інтерфейсу	55
4 Результати роботи	61
Висновки	70
Перелік джерел посилання.....	72

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ПДР – правила дорожнього руху;

DOM – Document Object Model;

CSS – Cascading Style Sheets;

HTML – Hyper Text Markup Language;

SQL – Structured Query Language;

PHP – Hypertext Preprocessor;

UML – Unified Modeling Language.

ВСТУП

З кожним роком кількість водіїв на дорогах стає все помітнішою. За статистикою, з 1 січня по 31 грудня 2022 року видано 1220774 водійських посвідчень. З них 429131 водій отримав документи вперше і це майже на 20 тисяч більше, ніж торік [1].

Кожен, хто хоче отримати водійські права, повинен скласти теоретичний і практичний іспит, і є досить приголомшлива статистика про успішність майбутніх водіїв. Так, сервісні центри поділилися першими результатами пілотного проекту прозорих іспитів, які проходив у 4 областях: Львів, Київ, Одеса, Вінниця. Протягом тижня з 1 по 7 червня 2023 року теоретичні іспити з усіх типів транспортних засобів складали 2524 кандидати у водії, а практичні – 2652. Успішність першої спроби написання теорії становить 23%, а успішність складання практики – 36% [2].

Зважаючи на такі результати стає зрозумілим, що окрім практичних навичок, майбутнім водіям не вистачає і теоретичних знань. Аби набути вищого рівня знань з теорії потрібно використовувати не лише метод його повторення і завчання. Для їх зміцнення доцільно використовувати тестування, як навчальний інструмент. Тестування є зручним методом навчанням, адже підготують до найголовнішого іспиту і надають можливість виявити прогалини у знаннях. Крім того, дозволяють більш реалістично оцінити власні сили. Кандидати у водії звикнуть до такого формату роботи і менше нервуватимуть під час іспиту у МРЕО.

Звичайні тестування можливо використовувати разом із сучасними технологіями, що дозволяє перевести їх у формат онлайн-тестувань. Вони мають такі переваги як:

- доступність – можливість проходження у будь-який час та у будь-якому місці, де є доступ до інтернету, та з будь якого пристрою: телефону, планшета, ноутбука, персонального комп'ютеру;

— автоматизація – перевіркою тесту займається машина, а не фахівець, що дозволяє отримати результати миттєво.

Враховуючи, що онлайн-тестування це повністю автоматизований процес, до нього можливо включити аналіз проблемних тем. Такий підхід має на меті додавати більше питань з тем де було допущено більше помилок.

Таким чином, обрана тема є актуальною.

Мета роботи полягає у створенні інтелектуальної системи тестування водіїв, щоб допомогти підвищити рівень знань кандидатів у водії, завдяки врахуванню помилок за темами у попередніх тестах і пропонуванню більшої кількості запитань за цими темами.

Для досягнення поставленої мети необхідно проаналізувати існуючі аналоги, спроектувати базу даних та семантичну мережу, реалізувати проект з використанням обраних засобів розробки.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз вебсайтів для тестування водіїв

Для того аби визначити список вимог до розроблюваної системи було оглянуто вже існуючі аналоги і проаналізовано, які саме можливості вони надають користувачу.

“Про Дорожній Рух” – сайт з онлайн-тестами до екзамену з ПДР, а також теоретичними матеріалами для підготовки (рис. 1.1).

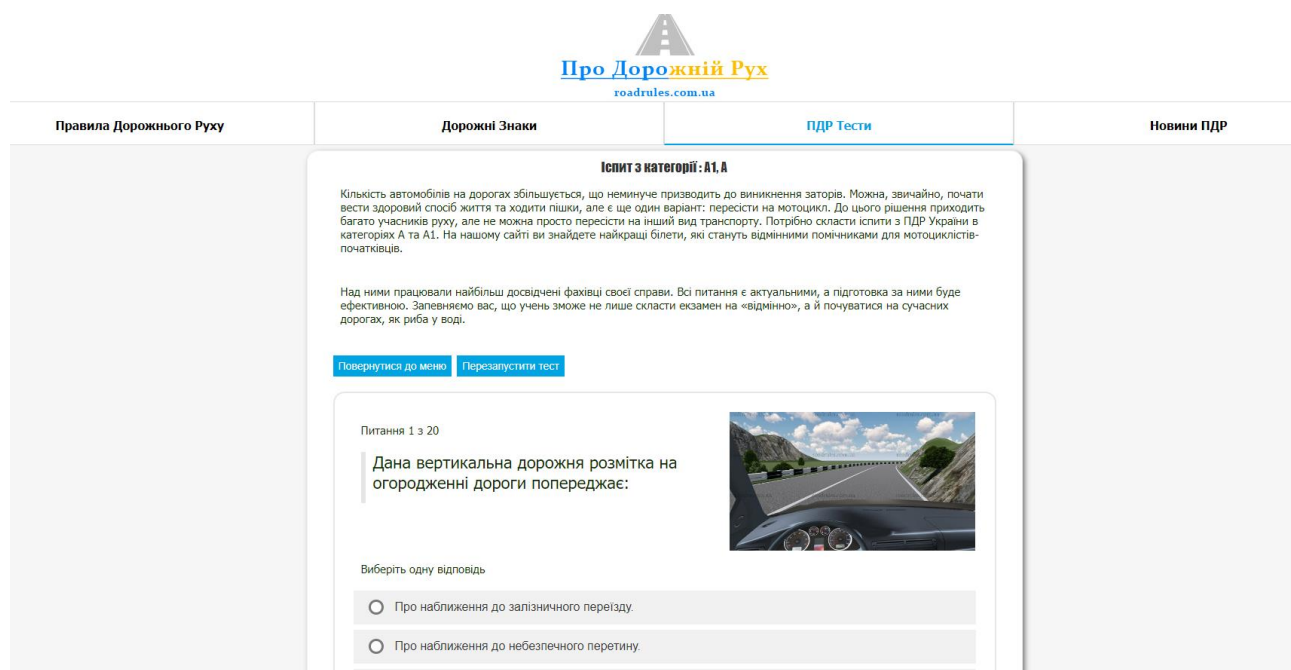


Рисунок 1.1 – Сторінка іспиту на сайті “Про Дорожній Рух” [3]

На сайті ви знайдете:

- ПДР Тести, які підготовлені у відповідності до чинних правил;
- максимально точно сформульовані запитання, відповіді на які гарантовано допоможуть скласти іспит ГСЦ;
- тести на здобуття навичок первинної медичної допомоги постраждалим в ДТП;
- питання по темам;

– іспити по категоріям.

Green Way Online – сервіс спрямований на якісну підготовку майбутніх українських автомобілістів (рис. 1.2). Над запитаннями тестів сайту не один рік працювали фахівці та викладачі провідних автошкіл України. Ведеться унікальна статистика успішності не тільки стосовно кожного учня, а й по Україні в цілому. Користувач на будь-якому етапі навчання може стежити за своєю успішністю, а також змагатися в знаннях з іншими учнями. За значні успіхи у навчанні йому нараховуватимуться різноманітні бонуси і бали.

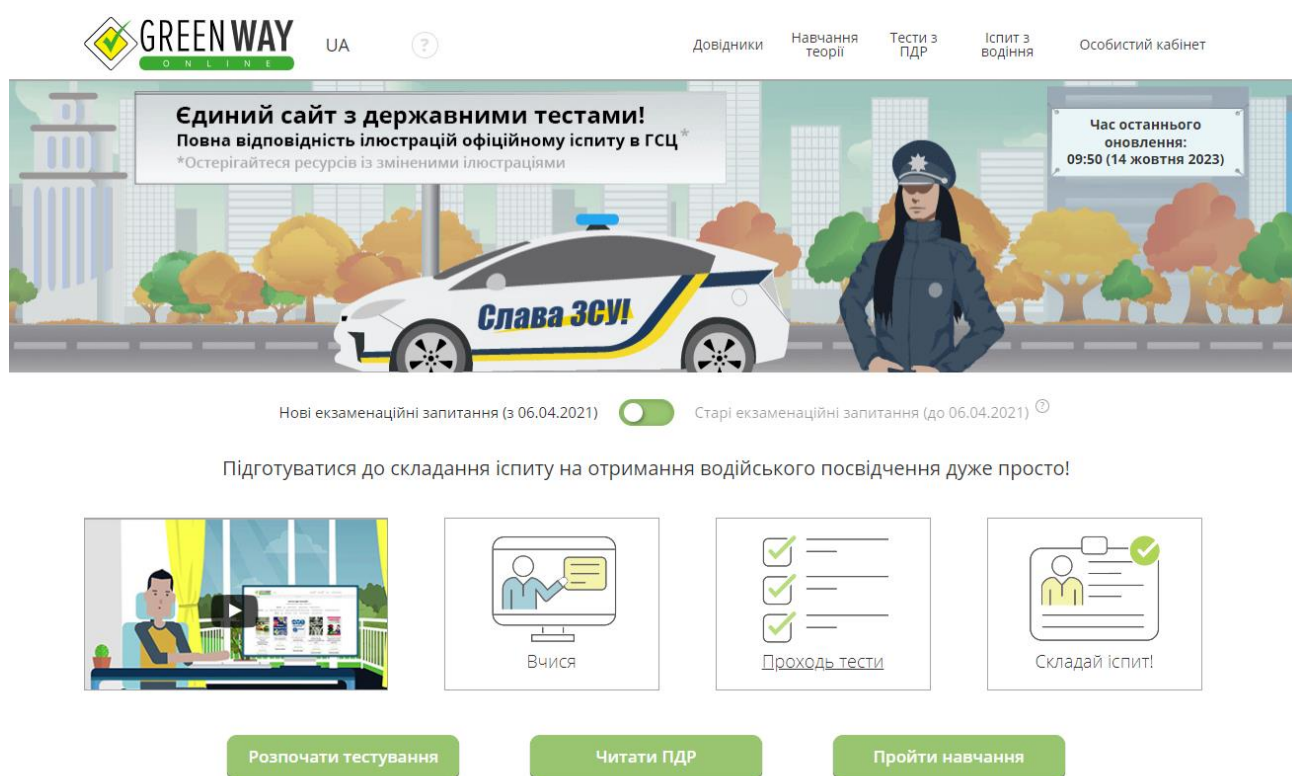


Рисунок 1.2 – Головна сторінка сайту Green Way Online [4]

Vodiy.ua – це сервіс для вивчення ПДР та підготовки до теоретичного іспиту у сервісних центрах МВС (рис. 1.3). Тут ви можете пройти тестування, переглянути правила дорожнього руху з функцією пошуку, картинками дорожніх знаків, дорожньої розмітки, системою перехресних посилань та підказок.

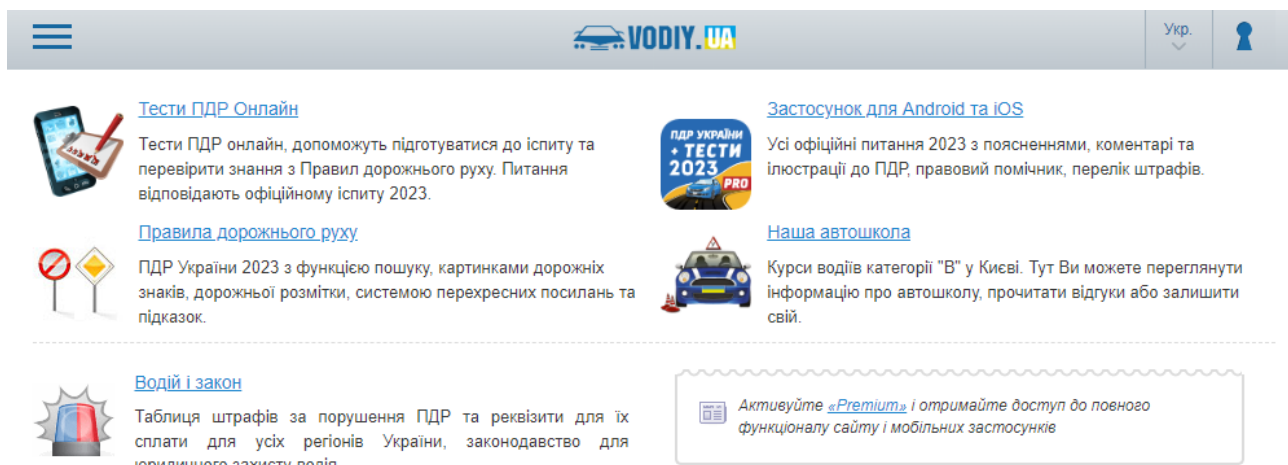


Рисунок 1.3 – Головна сторінка сайту Vodiya.ua [5]

1.2 Результати аналізу

За результатами аналізу аналогів було сформовано порівняльну таблицю сайтів за функціональними можливостями, які є пріоритетними для розробки проєкту (табл. 1.1).

Таблиця 1.1 – Порівняння розглянутих вебсайтів

Назва критерію	Назва сайту		
	Про Дорожній Рух	Green Way Online	Vodiya.ua
	1	2	3
Перегляд результатів тесту	+	+	+
Варіативність запитань у тестах	-	+	+

Продовження табл. 1.1

Назва критерію	1	2	3
Збереження результатів попередніх тестувань	-	+	+
Робота над помилками	-	+	+
Урахування попередніх результатів у наступних тестах	-	-	-
Відображення прогресу за темами	-	+	-

Зважаючи на результати порівняльного аналізу у таблиці, визначимо основні вимоги до розроблюваного проєкту:

- навігаційні;
- реєстрації та входу;
- введення даних;
- створення автоматичного тесту.

Таким чином, користувачі мають легко пересуватися між сторінками та легко знаходити потрібну їм інформацію; мати можливість створювати обліковий запис, вхід в систему; мати можливість додавати, редагувати чи видаляти таку інформацію як текстові дані; отримувати автоматично створені форми тестування з питаннями до неї на основі аналізу помилок користувача, та їх перевірка.

Якщо детально описати усі вимоги до системи, то вона повинна надавати такі можливості:

- переглядати результати тесту;
- зміна запитань у тестах;
- збереження результатів попередніх тестів;
- урахування попередніх результатів у наступних тестах;
- відображення прогресу за темами.

Для підтримки визначеного функціоналу необхідно аби користувач був зареєстрований, адже лише в такому випадку буде можливим відслідковувати прогрес і визначати проблемні теми за якими необхідно надавати більше запитань. У випадку коли користувач не зареєстрований запитання у тесті кожного разу будуть випадковими.

Таким чином, проведений аналіз існуючих систем продемонстрував, що частина сайтів для проходження тестувань для підготовки до складання іспиту в МРЕО не враховує, з якими саме темами у користувача виникають труднощі. Урахування цих труднощів дозволить не лише поліпшити процес навчання, а і краще засвоїти погано опрацьований матеріал. Тому в розроблюваній системі важливо фіксувати помилки у певній темі, аби пропонувати майбутньому водію більше питань з неї наступного разу.

1.3 Типи завдань у системі

При створенні подібних систем дуже важливим є не лише питання, які формуються для тестування майбутніх водіїв в межах ПДР, а і те, у якому вигляді ці завдання можуть бути подані.

У даній роботі розглядається 3 види завдань:

- завдання із питанням за зображенням;
- завдання із одним чітким варіантом відповіді;

— завдання із одним варіантом відповіді, що складається з переліку правильних відповідей.

1.4 Програмне забезпечення

Розробка програмного забезпечення проводиться на основі операційної системи Windows 10 з використанням ХАМРР. ХАМРР є безкоштовним набором програмного забезпечення, що використовується для створення локального серверу на комп'ютері. Він призначений для розробки та тестування вебсайтів перед їх розгортанням на живому сервері.

Серед переваг ХАМРР можна виділити дуже просту інсталяцію, сервери для використання у різних системах, доступність в головному меню всіх функцій, що дозволяє переглянути статус сервера і встановлені бібліотеки. Проте при використанні ХАМРР слід звернути увагу на кількість пам'яті яку вона займає, що може бути більше 100МБ і залежить від обраної комплектації.

Збірка ХАМРР включає в себе Apache, MySQL та PHP, що використовуються у проєкті. Версії продуктів що використовуються:

- Apache версії 2.4.54;
- MySQL версії 10.4.27;
- PHP версії 8.1.12.

Зважаючи на те, що сайт призначений для тестування, для його подальшого розгортання на живому сервері, необхідно аби хостинг включав дисковий простір від 1 ГБ до 5 ГБ, через наявність питань та зображень. Оперативної пам'яті має бути від 2 ГБ до 4 ГБ, якщо кількість відвідувачів буде помірною. Якщо кількість відвідувачів буде значною, кількість оперативної пам'яті має бути від 4 ГБ до 8 ГБ або більше. Також вебсервер має бути з підтримкою PHP і MySQL, версія PHP від 7 і вище, версія MySQL від 5.6 і вище, HTTPS протокол.

2 ПРОЄКТУВАННЯ ТА МОДЕЛЮВАННЯ СИСТЕМИ

2.1 Визначення функціональних вимог до проєкту

Проєкт повинен надавати можливість роботи з таким набором функцій:

- перегляд теоретичних матеріалів;
- проходження тестування;
- перегляд результатів пройденого тестування;
- перегляд результатів попередніх тестувань;
- аналіз помилок і враховування їх у наступних тестах;
- статистика результатів за темами питань;
- варіативність питань у тестах.

Відповідно до вимог, перерахованих вище, зрозуміло, що є функції, які можуть виконуватися, якщо користувач авторизований. Якщо користувач авторизований, він може взаємодіяти з усім функціоналом (рис. 2.1).

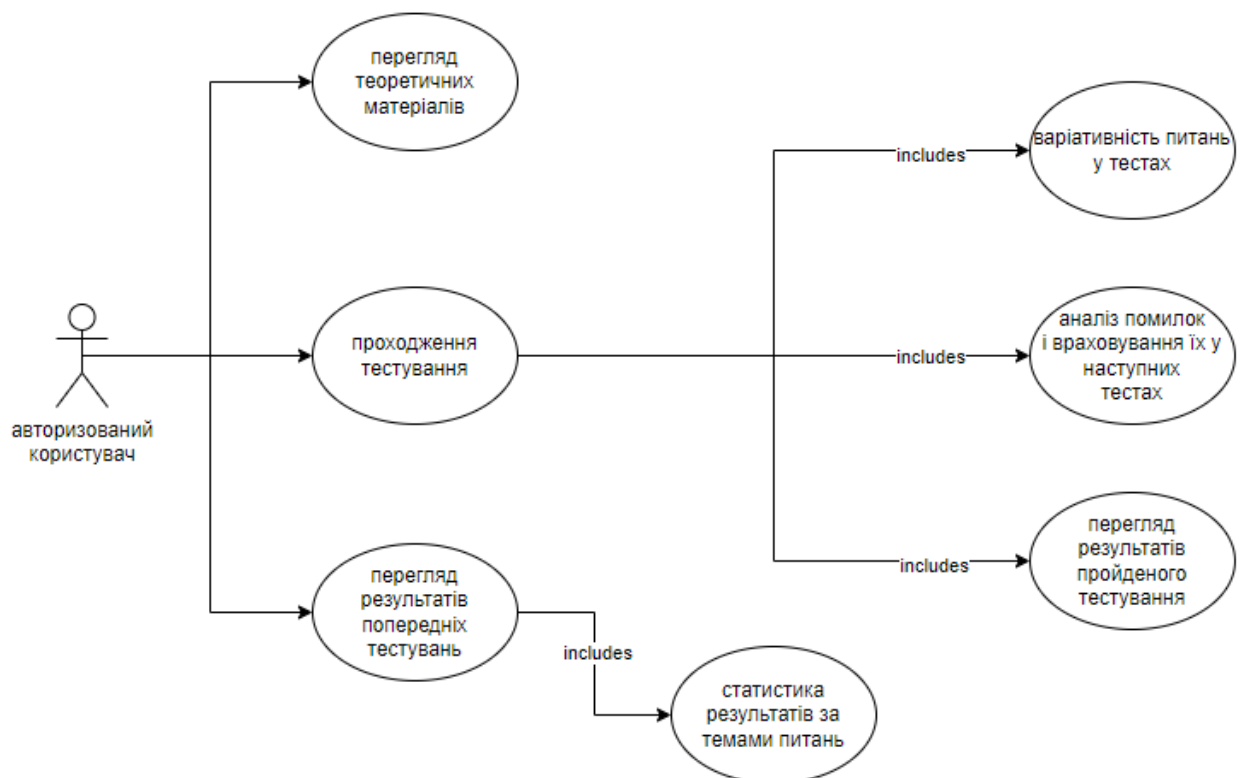


Рисунок 2.1 – Діаграма прецедентів авторизованого користувача

Якщо користувач не авторизований, він має доступ до таких можливостей як (рис. 2.2):

- перегляд теоретичних матеріалів;
- проходження тестування;
- перегляд результатів пройденого тестування;
- варіативність питань у тестах.

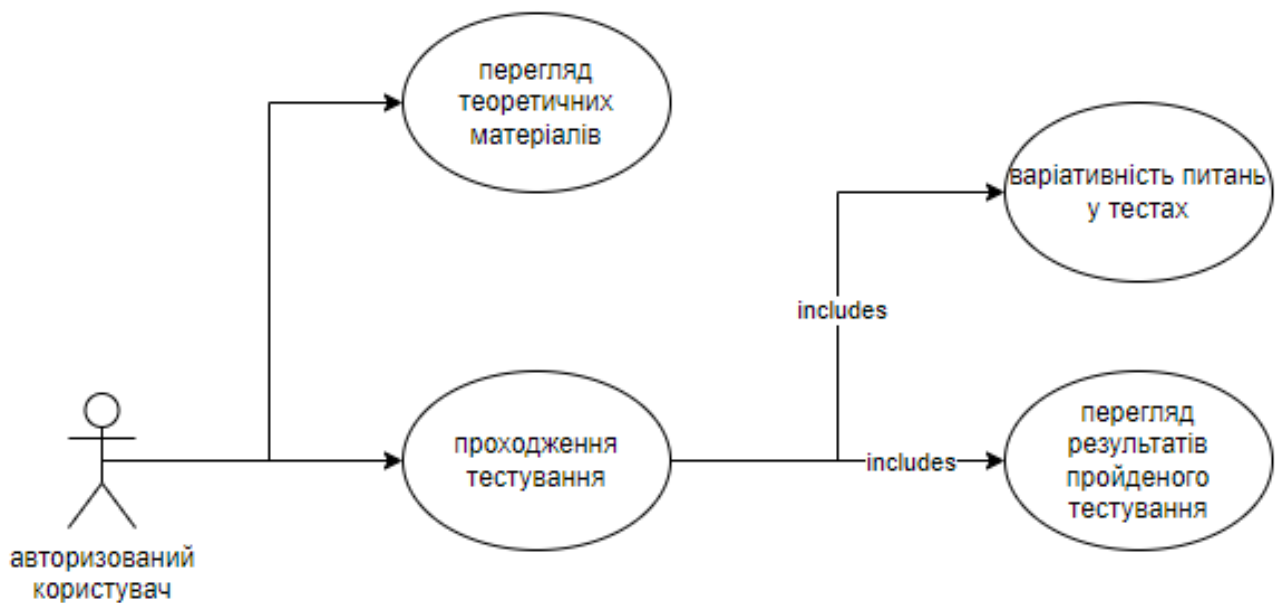


Рисунок 2.2 – Діаграма прецедентів не авторизованого користувача

2.2 Алгоритм функціонування проєкту

Роботу сайту розділено на чотири основні частини з урахуванням того чи авторизований користувач.

Коли користувач бажає зареєструватися або авторизуватися (рис. 2.3) на сайті завантажуватиметься форма з відповідними полями, які необхідно заповнити. Якщо було обрано і заповнено форму для авторизації, відбуватиметься перевірка чи є збіг між введеними даними та інформацією у базі

даних. Якщо значення вірні авторизація пройшла успішна, у протилежному випадку потрібно повторити заповнення форми. У разі реєстрації користувач має заповнити відповідну форму, після чого дані будуть додані до бази даних та перезавантажено сайт з функціоналом авторизованого користувача.

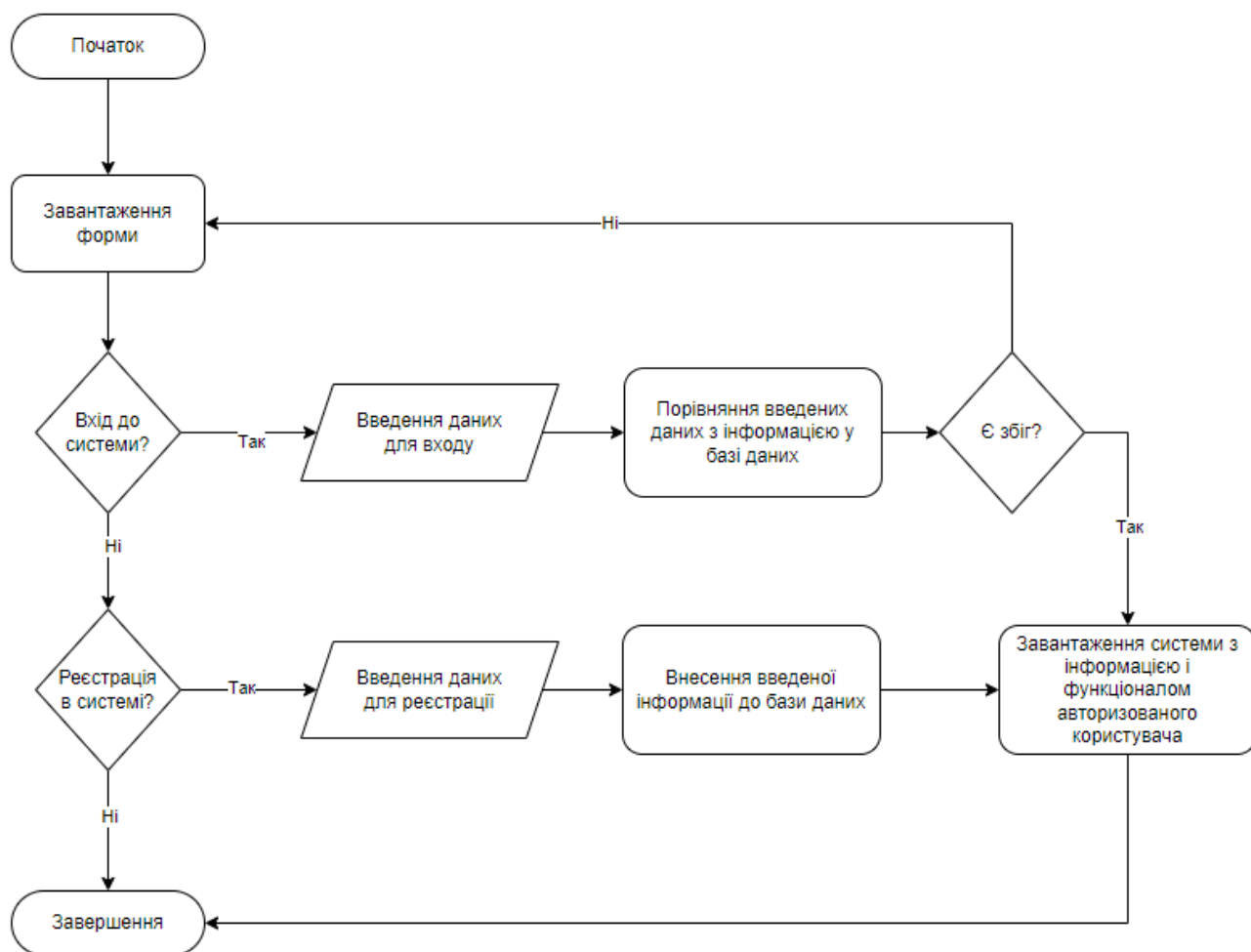


Рисунок 2.3 – Схема входу/реєстрації користувача

Коли користувач обирає пройти тестування (рис. 2.4) спершу здійснюється перевірка чи авторизований користувач, адже від цього залежить, як обиратимуться питання для тесту. У випадку коли користувач авторизований, значення має кількість попередніх спроб. Якщо умови з авторизації і кількості складених тестів не виконані, питання до тесту обиратимуться випадково. В протилежному випадку вибір питань відбуватиметься з урахуванням попередніх спроб.

У будь-якому випадку після визначення питань до тесту вони виводитимуться на екран користувача для надання відповідей, після чого користувач зможе переглянути свої результати. Вони будуть збережені, якщо до початку проходження тесту користувач був авторизований, а також буде оновлено прогрес за темами.

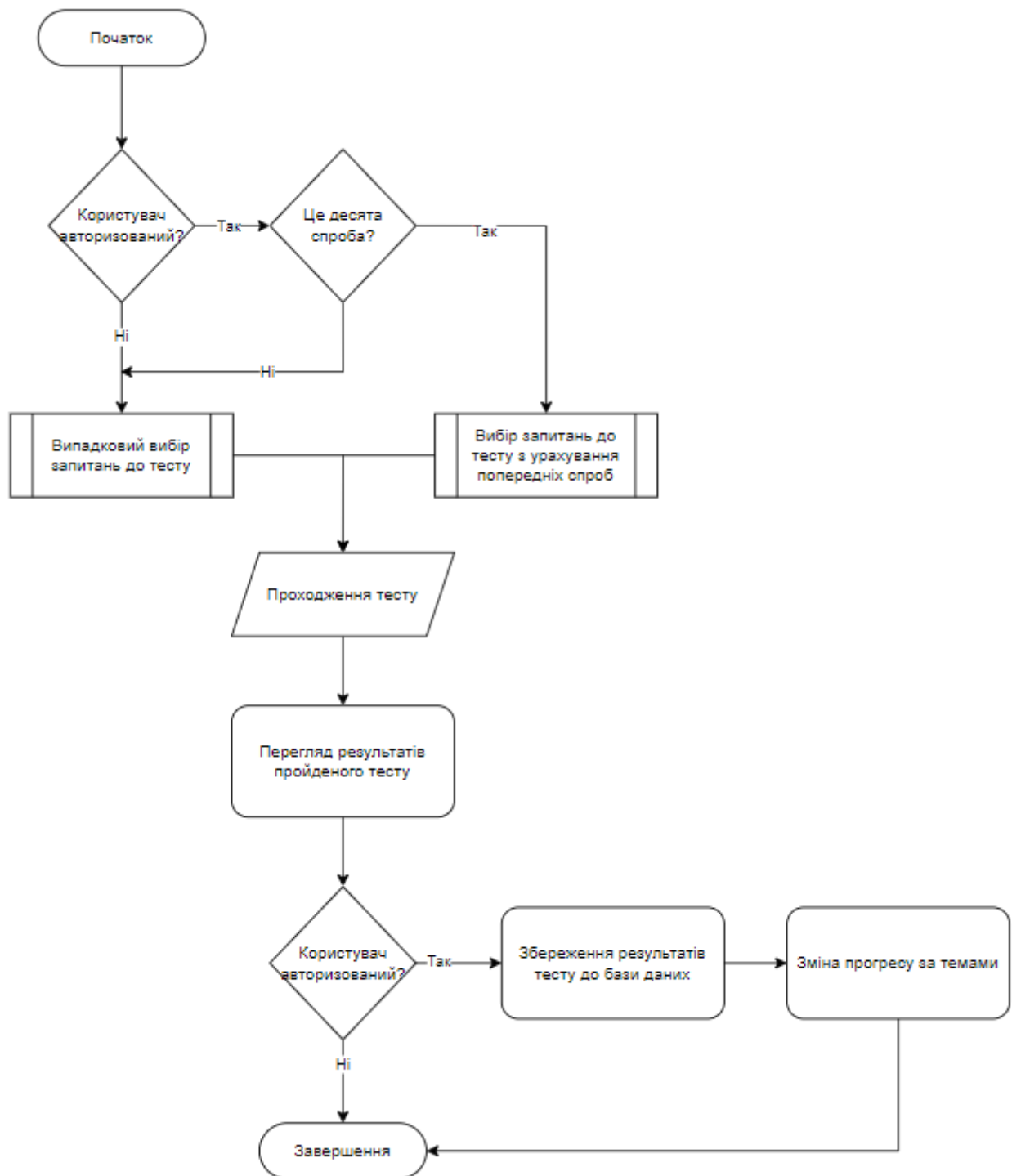


Рисунок 2.4 – Схема проходження тестування

Якщо користувач не авторизований питання тесту обиратимуться випадковим чином (рис. 2.5).

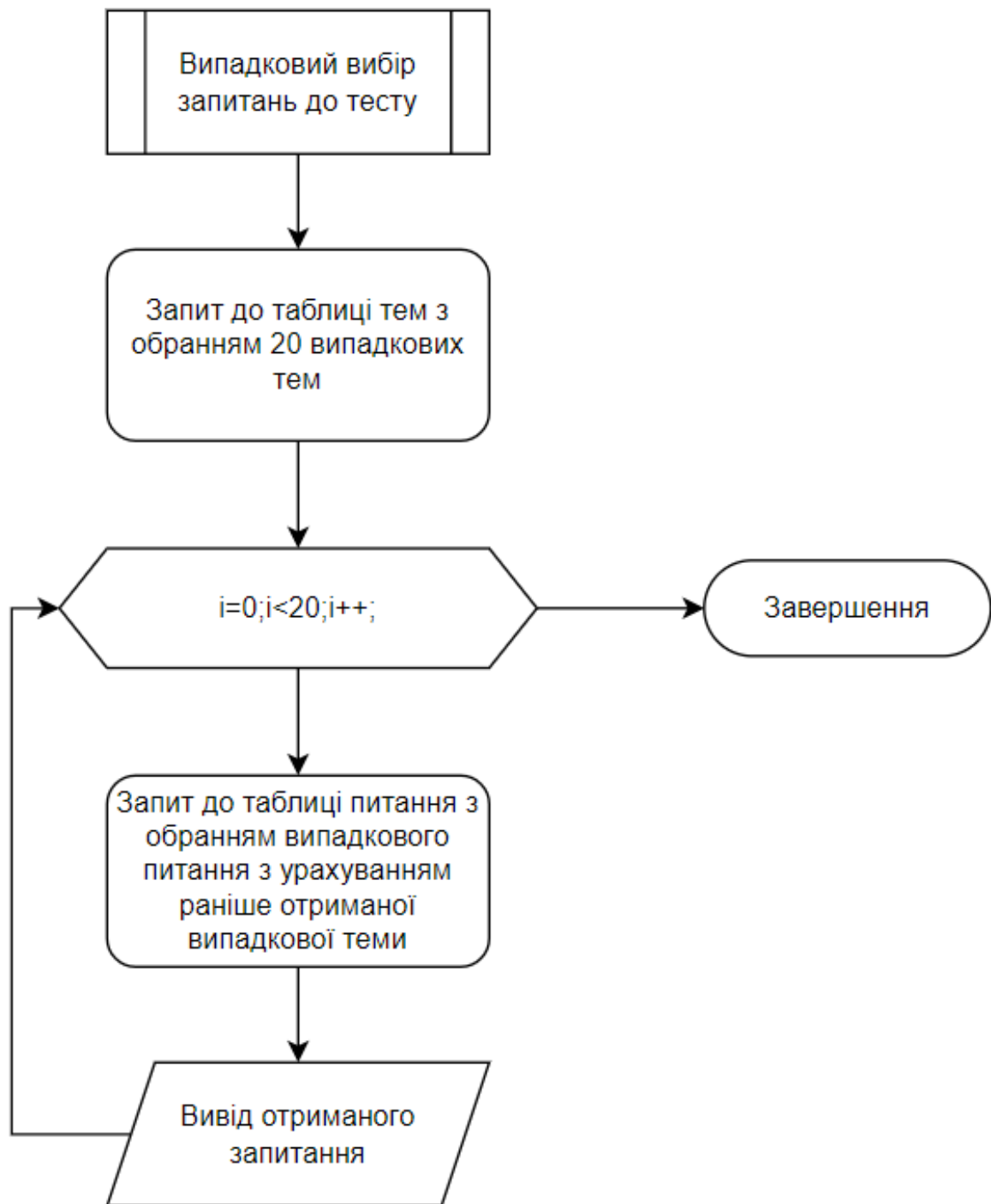


Рисунок 2.5 – Схема випадкового вибору запитань до тесту

Також вони будуть випадковими, якщо це перші десять тестів для авторизованого користувача, що необхідна для визначення проблемних тем.

Кількість у десять питань обрана довільно і ця кількість може змінитися під час експлуатації системи, спираючись на результати успішності проходження тестів і виявлені важких тем, адже на практиці разом зі збором статистики у виявлені складніших тем, коли більший відсоток користувачів матиме нижчий бал в певній тематиці, буде можливо зробити висновки про необхідну кількість «тестових» спроб. У випадку коли ці десять спроб виконані можливо обрати питання з урахуванням попередніх спроб (рис. 2.6).

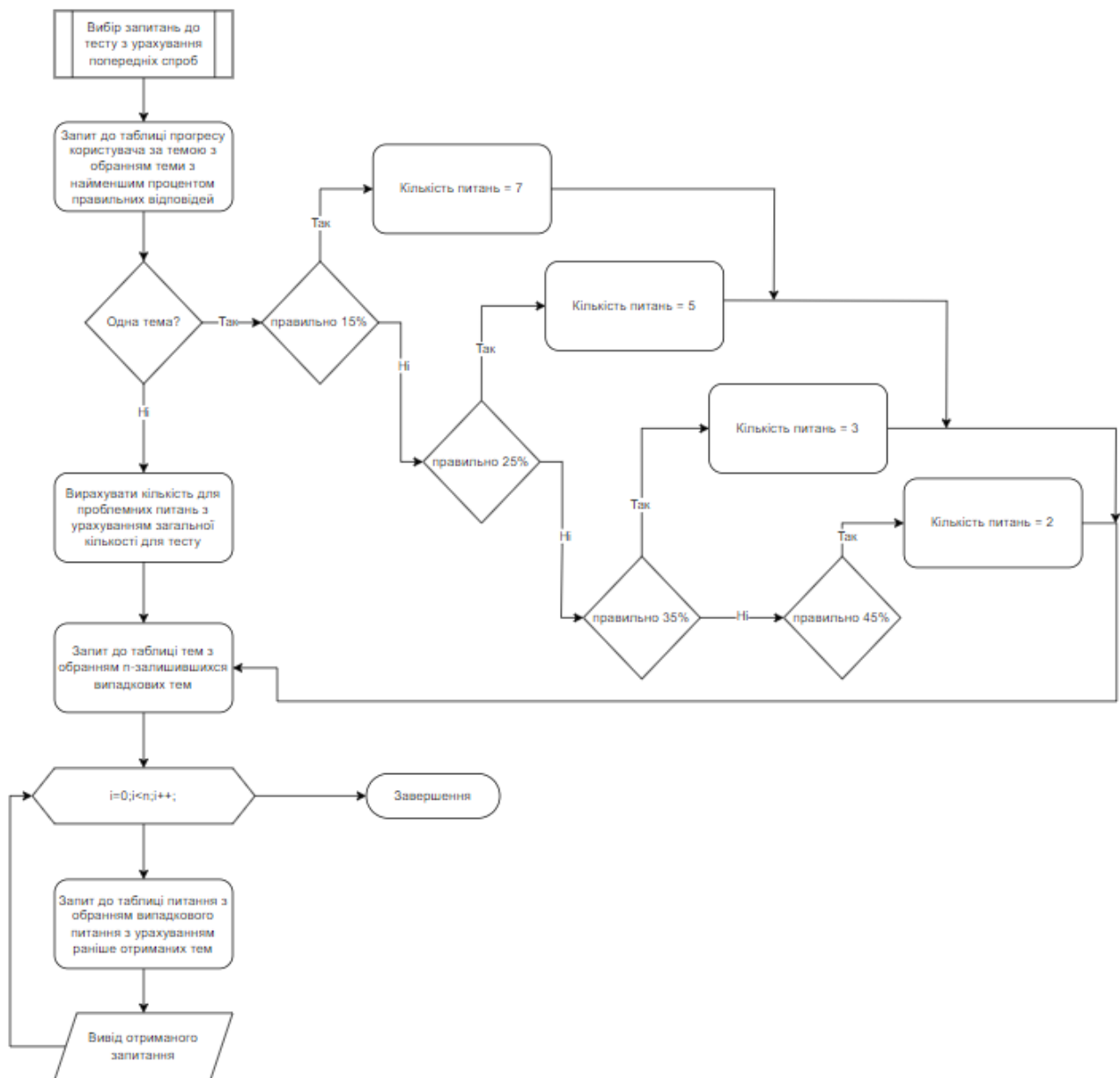


Рисунок 2.6 – Схема вибору запитань до тесту з урахуванням попередніх спроб

У алгоритмі відсотки та кількість питань також є умовними, адже потребують статистичного підтвердження у ефективності під час експлуатації. У будь-якому разі під час обрання питань вони виводитимуться користувачу, після чого він може відповісти на питання і переглянути свої результати. Якщо користувач авторизований, результати зберігатимуться до бази даних, аби він знову міг їх переглянути, а також прогрес за темами буде оновлено.

Коли користувач авторизований і він може переглянути результати вже пройдених тестувань (рис. 2.7).

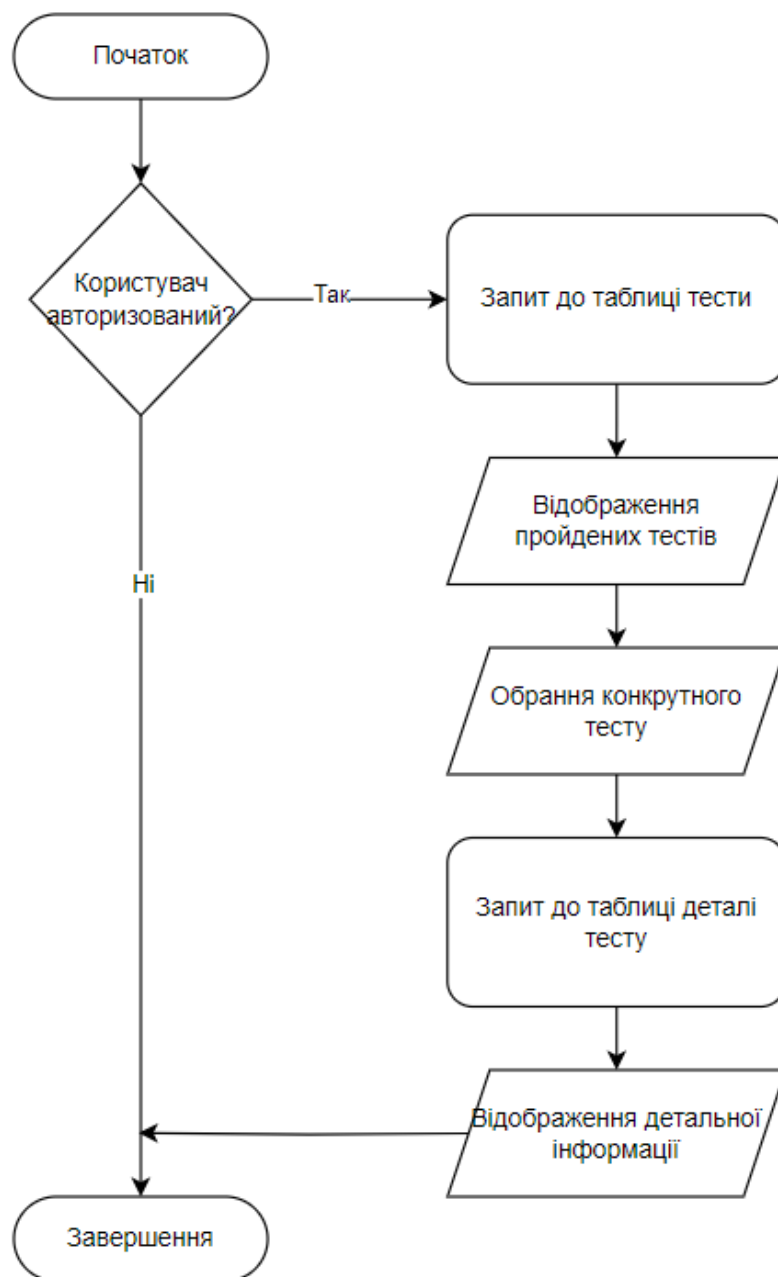


Рисунок 2.7 – Схема перегляду результатів пройдених тестів

Спершу буде виведено список усіх складених тестів з короткою інформацією, а після обрання конкретного варіанту для перегляду результати буде завантажено більш детально, для цього також виконуватиметься запит з отримання необхідних даних.

Також авторизований користувач може дізнатися свій прогрес за темами запитань у тесті (рис. 2.8).

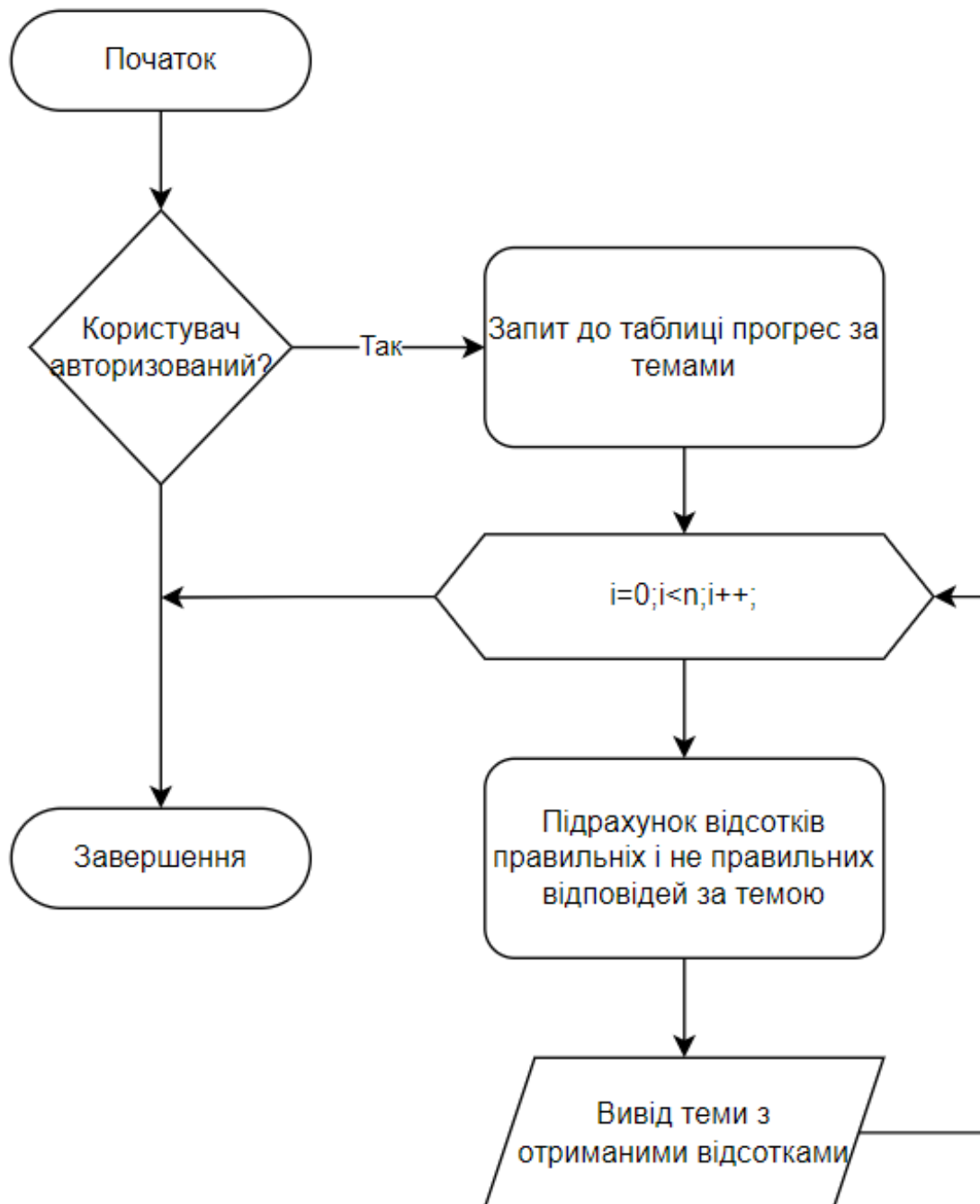


Рисунок 2.8 – Схема статистика результатів за темами питань

Для цього буде зроблено запит до відповідної таблиці аби отримати загальну кількість питань, кількість питань з правильною відповіддю, не правильно та без відповіді. Так як отриманий результат з бази даних є переліком тем необхідно виконати проходження по кожній з них для виконання підрахунків. Після чого буде вираховано відсоток прогресу та виведено користувачу.

2.3 ER-модель бази даних

Під час проектування бази даних було використано ER-модель – це модель даних, яка призначена для спрощення процесу проектування бази даних та представлення бази даних у вигляді наочних графічних діаграм [6].

Згідно з виділеними сутностями було розроблено наступну ER-модель, що показує усі зв'язки, що мають бути між таблицями. Різним кольором виділено поля, що повторюватимуться в таблицях відповідно розробленій моделі. Кожен перший атрибут у кожної сутності є первинним ключем, а виділені кольором поля, що не є першими для всіх сутностей ER-моделі - це зовнішні ключи. Також відмічено які саме смислові зв'язки з'єднують вказані сутності.

Сутність - це збірне поняття, деяка абстракція реального об'єкта, процесу, явища чи деякого уявлення про об'єкт [7].

Для проєкту виділено такі сутності:

- користувач – особа, яка користується сайтом;
- прогрес за темою – інформація, що відображає успішність за певною темою, спираючись на кількість правильних і неправильних відповідей до питань теми;
- складені тести – коротка інформація про тести, що були вже складені користувачем;

- деталі тесту – більш розширена інформація про складений тест, що включає питання і відповіді на нього;
- питання – форма тексту, яка потребує відповіді;
- теми – розділена інформація за спільним змістом.

Через розміри розробленої ER-моделі її було розділено на частини для більш зручного перегляду.

На рисунку 2.9 наведено першу частину ER-моделі, де використано такі сутності як користувач, складені тести та прогрес за темою.

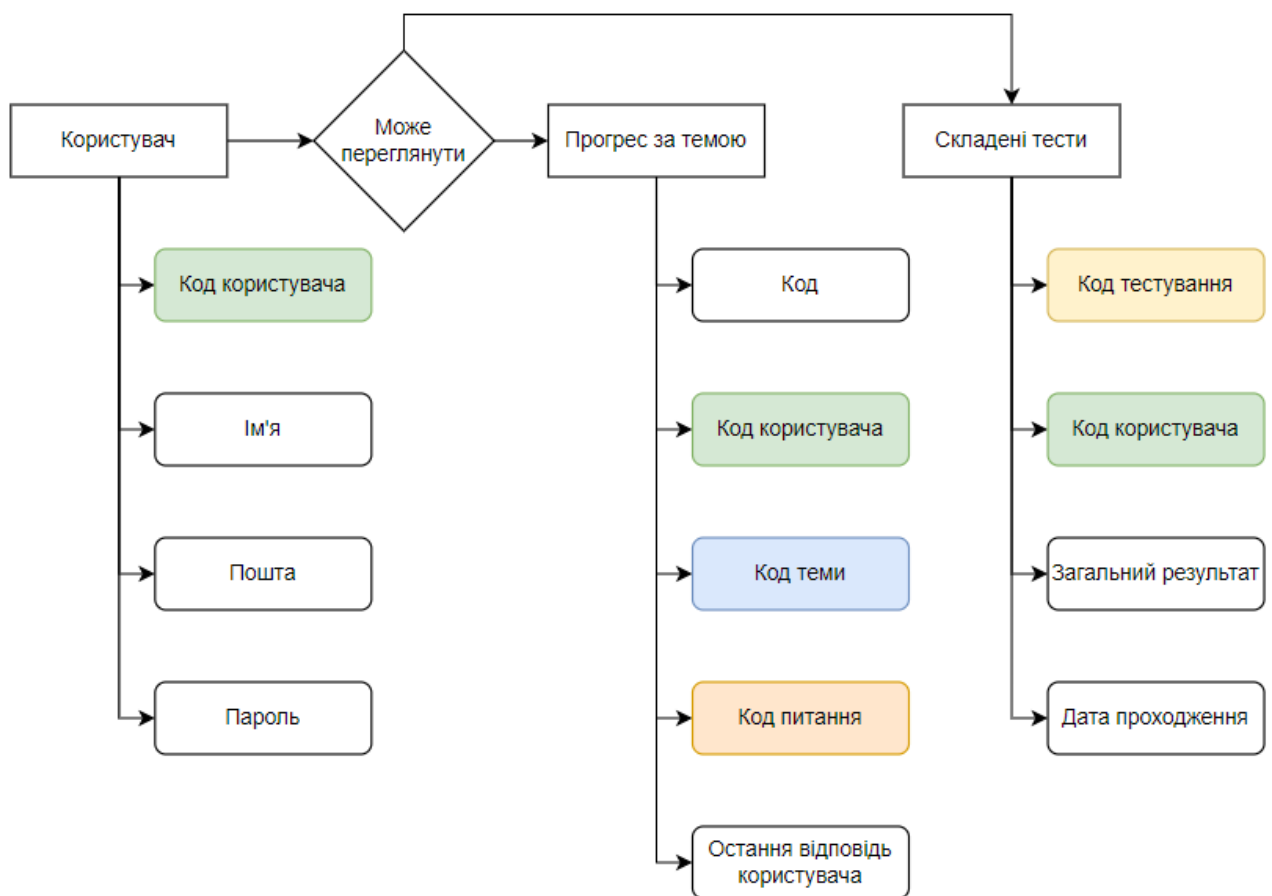


Рисунок 2.9 – Перша частина ER-моделі

Сутність користувач має такі атрибути як код користувача, ім'я, пошта та пароль. Сутність прогрес за темами має такі атрибути як код, код користувача, код теми, код питання та остання відповідь користувача. Сутність складені тести має такі атрибути як код тестування, код користувача, загальний результат та

дата народження. Сутність користувач пов'язана з сутностями складені тести та прогрес за темами, адже користувач може переглянути кожну зв'язану з ним сутність.

На рисунку 2.10 наведено другу частину ER-моделі, у якій використано такі сутності як складені тести, деталі тесту, питання та вказано, що сутність складені тести має зв'язок з сутністю деталі тесту, яку наведено в попередній частині ER-моделі на рисунку 2.9.

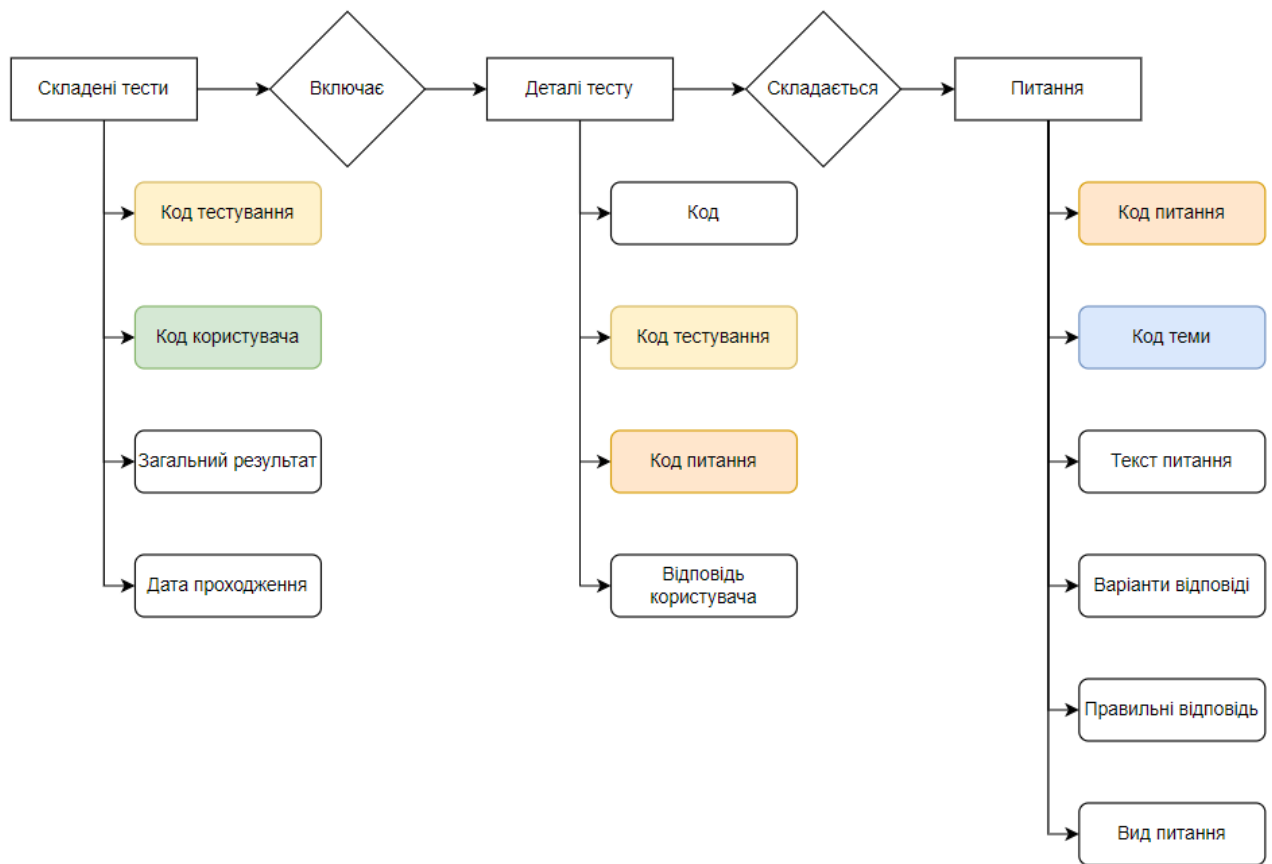


Рисунок 2.10 – Друга частина ER-моделі

Сутність деталі тесту має такі атрибути як код, код тестування, код питання та відповідь користувача. Сутність питання має такі атрибути як код питання, код теми, текст питання, варіанти відповіді, правильні відповіді, вид питання. Сутність складені тести пов'язана з сутністю деталі тесту, бо матимуть більш детальну інформацію, що зберігатиметься в окремій таблиці відносно кожного питання, що були у тесті. Сутність деталі тесту пов'язана з сутністю питання, бо

вони складатимуться з питань і варіантів відповідей, що матимуть показуватися щоб користувач міг побачити яку відповідь він дав і, якщо вона є хибною, правильну.

На рисунку 2.11 наведено третю частину ER-моделі. Тут використано такі сутності як питання і теми та вказано, що сутність питання має зв'язок з сутністю теми, яку наведено в попередній частині ER-моделі на рисунку 2.10.

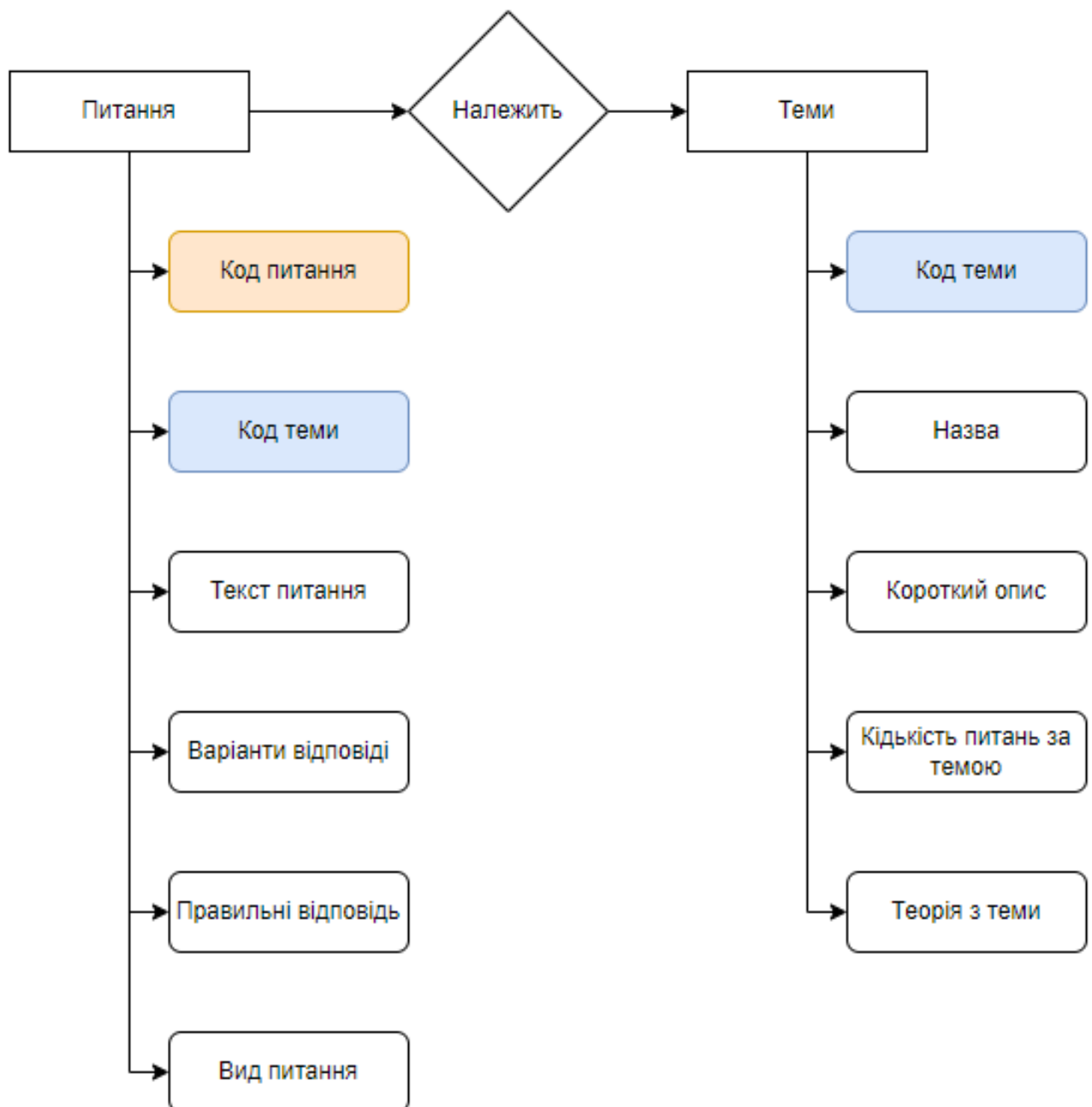


Рисунок 2.11 – Третя частина ER-моделі

Сутність теми має такі атрибути як код теми, назва, короткий опис, кількість питань за темою та теорія з теми. Сутність питання пов'язана з сутністю теми, бо кожне питання належить до певної теми за теорією якої воно було складене.

2.4 Семантична мережа

Для виконання завдань, які потребують людського розуміння, зазвичай використовуються інтелектуальні системи, що застосовуються у різних галузях, зокрема у освітній. Такі системи здатні до збору, аналізу та інтерпретації інформації аби вирішувати конкретні завдання та виконувати конкретні функції.

Створення інтелектуальної системи вимагає використання різних технологій і спираючись на тему проєкту, можна визначити, що доцільно буде використовувати технологію персоналізованих рекомендацій, яка полягає у використанні алгоритмів, що аналізують дані про користувачів та їх взаємодії з контентом, щоб рекомендувати послуги або інші елементи, які можуть бути цікавими користувачу [8, 9].

Інтелектуальні системи пов'язані з семантичними мережами, так як вони забезпечують простий спосіб для подання знань у графічному виді. Особлива увага тут приділяється зв'язкам між різними одиницями інформації та між різними масивами знань [10].

Таким чином, використовуючи технологію персоналізованих рекомендацій та семантичну мережу можливо створити інтелектуальну систему, що аналізуватиме структуру знань користувачів та персоналізовано визначатиме релевантні теми для наступних питання у тестах.

Створення семантичної мережі полегшує побудову логіки для створення запитів до бази даних і визначення необхідної інформації. Наведена семантична

мережа за темою допомагає узагальнити абстракції, що робить роботу з різними типами сутностей та їх відносинами.

Семантичну мережу можна представити у вигляді графу, який, відповідно до теми проєкту, можна описати таким чином:

- теми питань – вершини семантичної мережі;
- питання – вузли мережі;
- зв'язки між темами та питаннями – гілки у мережі, що показують які питання відносяться до кожної теми;
- атрибути питання – види питань, що розрізняються за тими у яких є зображення та без нього.

На рисунку 2.12 схематично наведено семантичну мережу за темою.

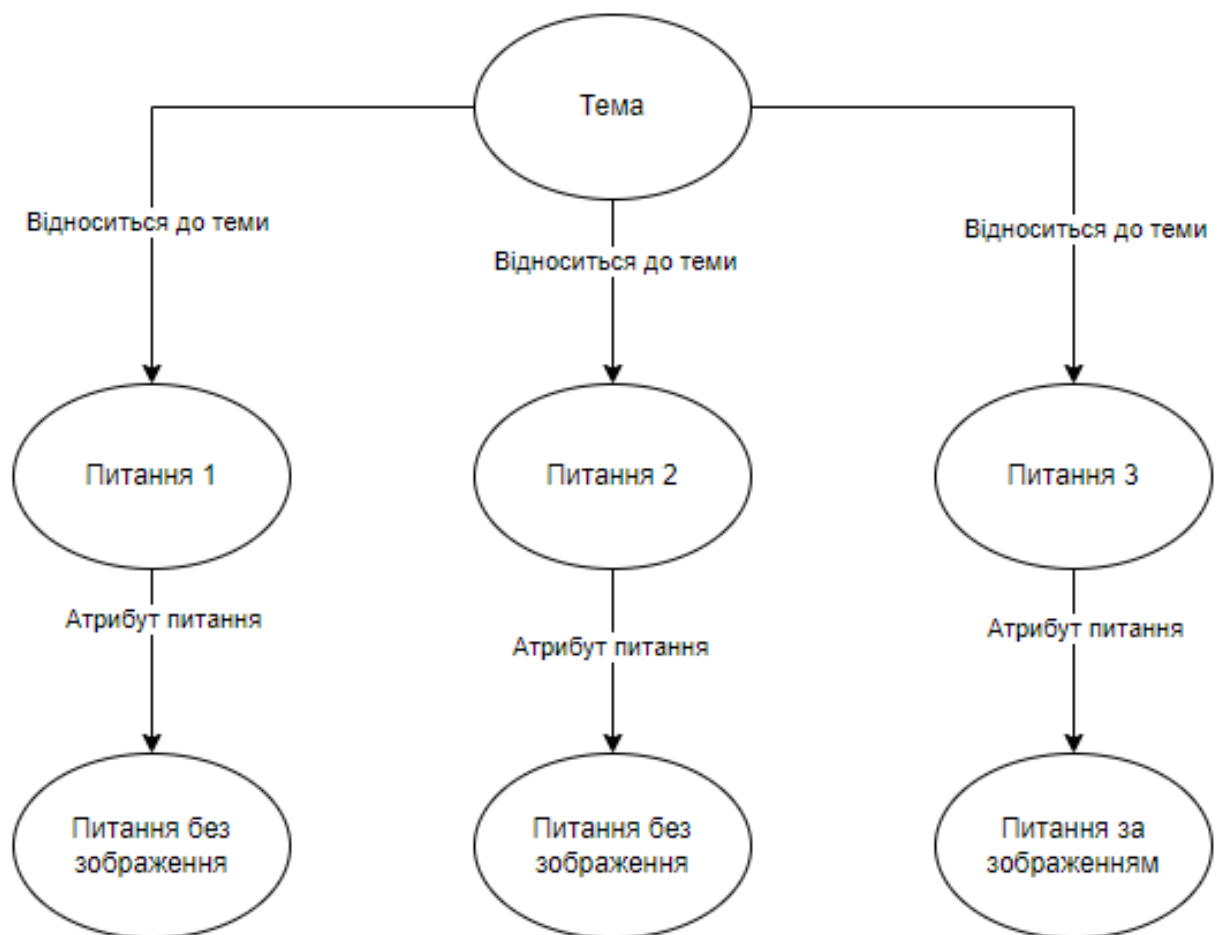


Рисунок 2.12 – Семантична мережа за темою

2.5 Діаграми послідовності

Діаграма послідовностей призначена для відображення об'єктів впорядкованих за часом. Вона використовується для уточнення діаграми прецедентів. Діаграми послідовностей зазвичай містять об'єкти, що взаємодіють у рамках сценарію та повідомлення, якими вони обмінюються, і які повертають результати, що пов'язані з повідомленнями.

На рисунку 2.13 зображення діаграми для процесу авторизації/реєстрації користувача на сайті.

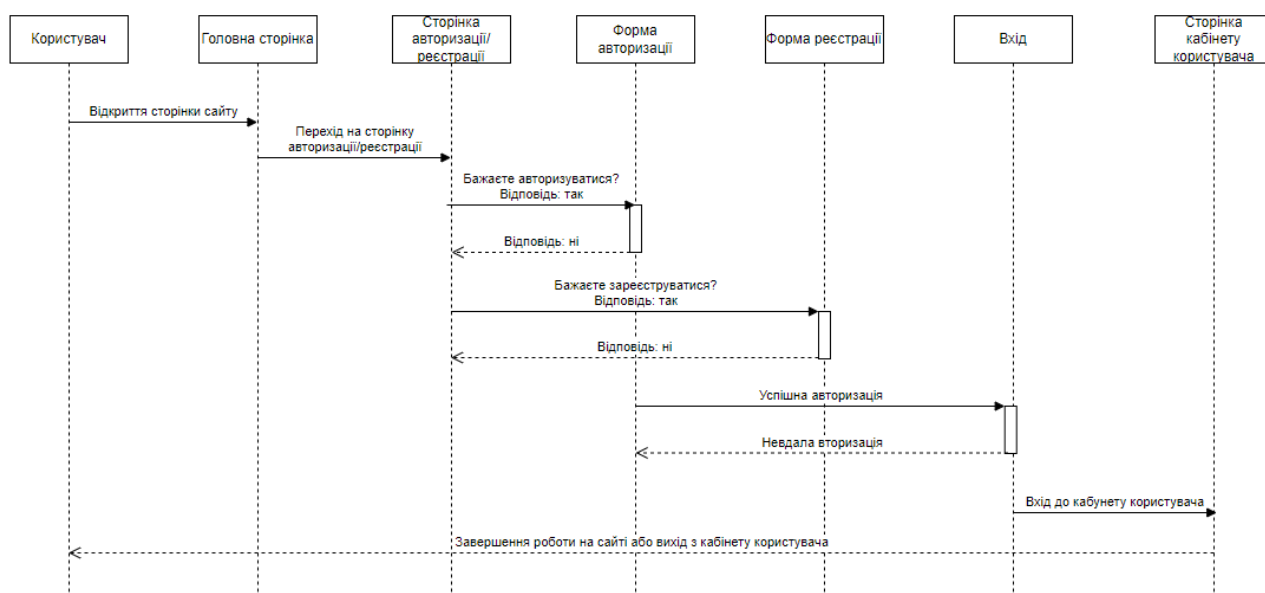


Рисунок 2.13 – UML-діаграма послідовності процесу авторизації/реєстрації на сайті

Користувач має можливість проходити тестування, навіть якщо він не авторизувався. Але у випадку якщо він не авторизований тоді його результати тесту не буде збережено, прогрес за темами не буде сформовано і питання кожного разу будуть лише випадковими. Процес старту і проходження тесту наведено на рисунку 2.14.

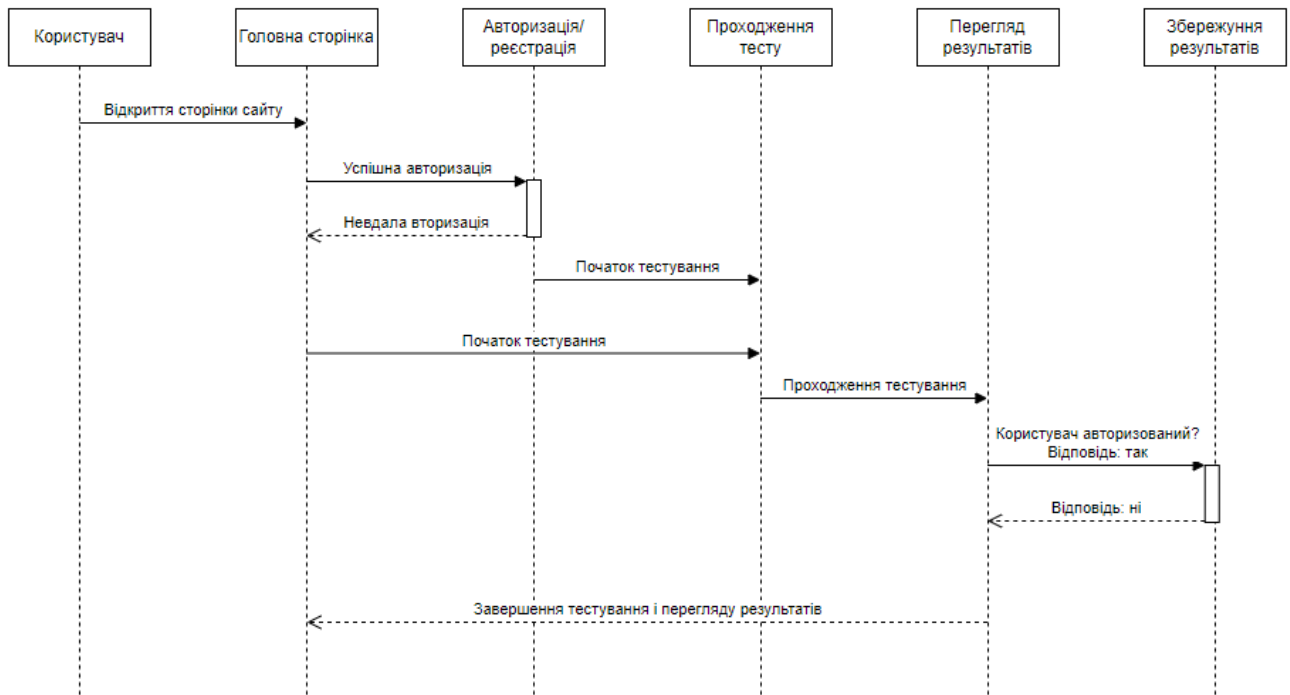


Рисунок 2.14 – UML-діаграма послідовності процесу проходження тестування

Якщо користувач був авторизований, після проходження тестування результат зберігається і його можливо переглянути повторно. Процес перегляду збережених результатів тестувань наведено на рисунку 2.15.

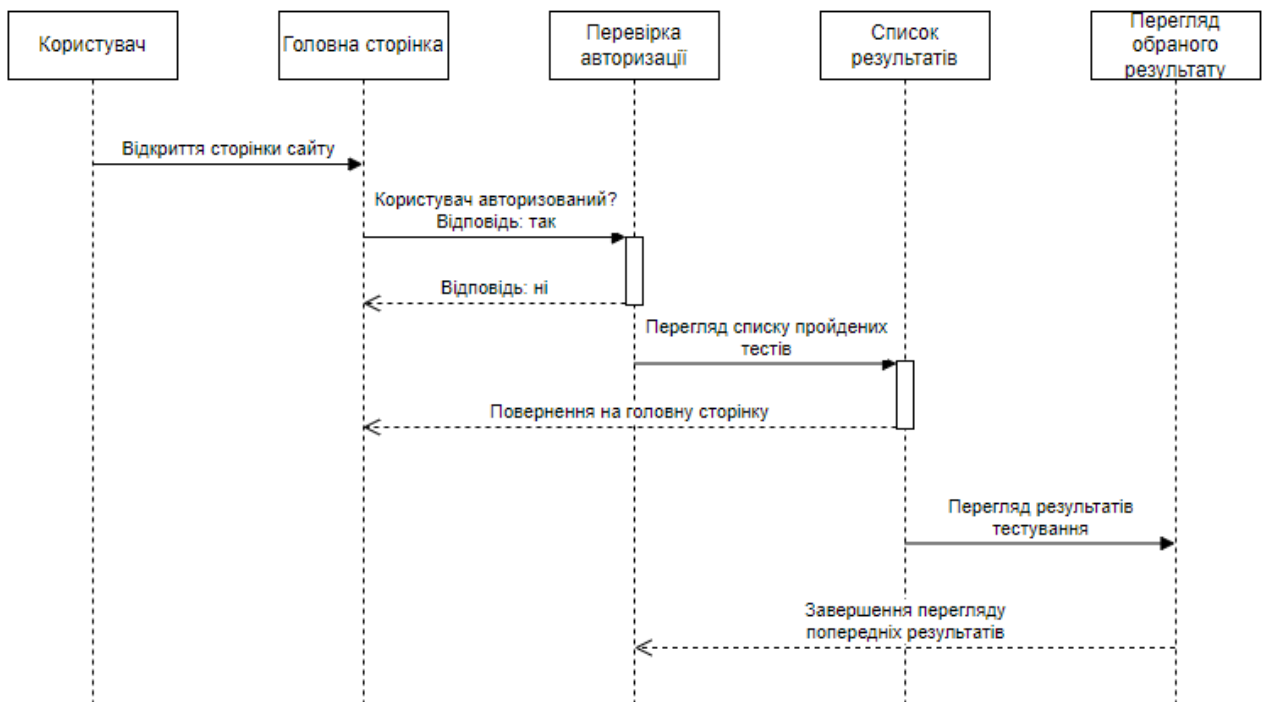


Рисунок 2.15 – UML-діаграма перегляду збережених результатів

Також якщо користувач авторизований він може переглянути свій прогрес за темами. Процес перегляду прогресу наведено на рисунку 2.16.

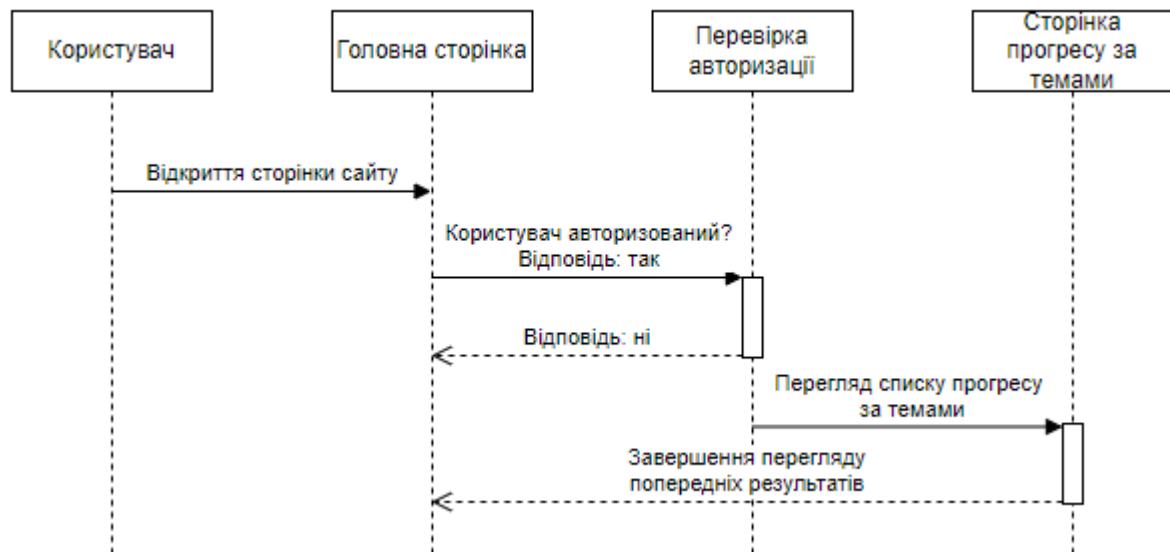


Рисунок 2.16 – UML-діаграма перегляду прогресу за темами

Для додаткового повторення або отримання нових знань користувач має можливість переглянути правила ПДР, що містять розписані правила руху. Процес перегляду правил наведено на рисунку 2.17.



Рисунок 2.17 – UML-діаграма перегляду правил ПДР

2.6 Розгорнуті описи прецедентів

У таблиці 2.1 наведено більш розгорнутий опис до процесу авторизації/реєстрації користувача на сайті, що наведено у розділі 2.7 у вигляді UML-діаграми і який визначає набір додаткових дій які користувач може виконувати знаходячись на сайті.

Таблиця 2.1 – Опис прецеденту авторизації/реєстрації

	Опис
Основний виконавець	Користувач сайту.
Зацікавлені особи та їх потреби	Користувач сайту: виконує реєстрацію або авторизацію на сайті.
	Сайт: пропонує користувачу ввести свої дані для входу до системи або виконати вхід без реєстрації.
Передумови	Потрібно перейти на головну сторінку сайту та натиснути кнопку особистого кабінету.
Результати (Постумови)	Користувач успішно потрапить до системи, якщо не виникне не передбачуваної помилки.
Основний успішний сценарій (основний процес)	1. Вхід на головну сторінку сайту. 2. Обрання варіанту входу. 3. Успішний вхід. 4. Перехід до кабінету користувача.

У таблиці 2.2 наведено більш розгорнутий опис до процесу проходження тестування користувачем на сайті, у випадку коли він не авторизований, адже

процес вибору питань загалом працює за схожим алгоритмом для всіх користувачів, що наведено у розділі 2.7 у вигляді UML-діаграми.

Таблиця 2.2 – Опис прецеденту проходження тестування

	Опис
Основний виконавець	Користувач сайту.
Зацікавлені особи та їх потреби	Користувач сайту: переходить на сторінку проходження тестування.
	Сайт: запускає тестування, виводить результати пройденого тестування, зберігає результати і оновлює прогрес за темами, якщо користувач був авторизований.
Передумови	Вхід на головну сторінку, виконання авторизації за бажанням користувача, перехід на сторінку тестування.
Результати (Постумови)	Проходження тесту і отримання результатів.
Основний успішний сценарій (основний процес)	1. Вхід на головну сторінку сайту. 2. Перехід на сторінку тестування. 3. Проходження тестування. 4. Отримання результатів тестування.

У таблиці 2.3 наведено більш розгорнутий опис до процесу перегляду збережених результатів тестування користувачем на сайті, що наведено у розділі 2.7 у вигляді UML-діаграми, у випадку коли він авторизований, інакше цей варіант буде недоступний для нього, і він не побачить необхідну сторінку серед переліку пунктів меню сайту, адже результати зберігаються до бази даних лише

коли користувач зайшов до системи і пункт меню також відображатиметься лише за умови авторизації.

Таблиця 2.3 – Опис прецеденту перегляду збережених результатів тестування

	Опис
Основний виконавець	Авторизований користувач сайту.
Зацікавлені особи та їх потреби	Авторизований користувач сайту: переглядає результати попередніх тестувань.
Зацікавлені особи та їх потреби	Сайт: завантажує список пройдених тестувань і деталі обраного пройденого тестування.
Передумови	Вхід на головну сторінку, виконання авторизації, перехід на сторінку результатів тестувань.
Результати (Постумови)	Перегляд детальної інформації обраного результату тестування.
Основний успішний сценарій (основний процес)	1. Вхід на головну сторінку сайту. 2. Авторизація на сайті. 3. Перехід на сторінку зі списком збережених результатів тестів. 4. Обрання певного результату зі списку. 5. Перехід до детальної інформації результатів тестування. 6. Перегляд цієї інформації.

У таблиці 2.4 наведено більш розгорнутий опис до процесу перегляду прогресу за темами на відповідній сторінці сайту наведеному у розділі 2.7 у вигляді UML-діаграми, який буде доступний також лише у випадку коли

користувач є авторизованим у системі, так само як і для перегляду попередніх результатів тестувань, адже необхідний пункт меню не буде відображено і не буде можливо виконати потрібні розрахунки для визначення прогресу.

Таблиця 2.4 – Опис прецеденту перегляду прогресу за темами

	Опис
Основний виконавець	Авторизований користувач сайту.
Зацікавлені особи та їх потреби	Авторизований користувач сайту: переглядає прогрес за темами.
	Сайт: завантажує сторінку зі список прогресу за темами.
Передумови	Вхід на головну сторінку сайту, виконання авторизації, перехід на сторінку прогресу за темами.
Результати (Постумови)	Перегляд списку прогресу за темами.
Основний успішний сценарій (основний процес)	1. Вхід на головну сторінку сайту. 2. Авторизація на сайті. 3. Перехід на сторінку зі списком прогресу за темами. 4. Перегляд цієї інформації.

У таблиці 2.5 наведено більш розгорнутий опис до процесу перегляду теоретичної інформації користувачем на сторінці сайту з правилами ПДР за темами, що наведено у розділі 2.7 у вигляді UML-діаграми, який є доступним для перегляду усіма користувачами сайту, і не потребує авторизації чи реєстрації у системі.

Таблиця 2.5 – Опис прецеденту перегляду теорії з правил ПДР

	Опис
Основний виконавець	Користувач сайту.
Зацікавлені особи та їх потреби	Користувач сайту: бажає переглянути теоретичну інформацію з правила ПДР.
	Сайт: завантажує сторінку правила ПДР з переліком тем та детальною інформацією до них.
Передумови	Вхід на головну сторінку сайту, обрання в головному меню сторінки пункту про правила ПДР для переходу на цю сторінку.
Результати (Постумови)	Перегляд правил ПДР з детальною інформацією до кожної теми.
Основний успішний сценарій (основний процес)	1. Вхід на головну сторінку сайту. 2. Обрання в головному меню сторінки правил ПДР. 3. Перехід на обрану сторінку. 4. Перегляд списку тем. 5. Обрання теми та перегляд детальної інформації до неї.

2.7 Обрані принципи побудови інтерфейсу користувача

Для створення хорошого інтерфейсу було проаналізовано способи, що використовуються для побудови вдалого юзабіліті.

Інтуїтивно зрозумілий інтерфейс користувача означає, що людина зайшовши на сайт сприймала його легко і їй не потрібно було витрачати зайвий час аби дізнатися яка кнопка що робить і натискати їх навмання, та розуміти як користуватися сайтом [11].

Сайт тестування є здебільшого має текстовий контент, тому його логічний інтерфейс має бути таким:

- назва сайту має розміщуватися у самому верху сторінки;
- нижче або поруч з назвою сайт має розміщуватися меню з розділами сайту;
- кнопки мають мати зрозуміле значення.

Також інтерфейс має бути мінімалістичним. Не потрібно розміщувати якнайбільше категорій, меню, кнопок. Якщо можливо описати щось однією фразою, воно не повинне бути записане трьома [11].

Необхідно пам'ятати про спілкування з користувачем, тобто користувач повинен розуміти, як його дія зараз обробляється системою. Щоб користувач міг спостерігати за станом системи, необхідно розміщувати прогрес-бари. На рисунку 2.18 наведено приклад прогрес-бару для сайту.



Рисунок 2.18 - Приклад прогрес-бару для сайту [11]

Для різних типів дій стилі кнопок мають бути різними. У всіх інтерфейсах кожна кнопка може виконувати різні дії: перейти кудись, розкрити меню, відкрити в новому вікні, скачати тощо. Щоб не заплутати користувача, важливо аби елементи на які можна натиснути і ті що є не клікабельними повинні бути різними. Їх можна виділяти кольором та розміром. Користувач має бачити на що він може натиснути та перейти кудись [11].

Так як на сайті передбачена авторизація, форми мають бути короткими. Щоб користувач не відчував неприязнь до процесу реєстрації, цей процес необхідно зробити швидким та простим. Форма має обмежуватися легше найнеобхіднішою інформацією. На рисунку 2.19 наведено приклад простої форми.

Register for Evernote

Full name

Email address*

Username*

Password*

Confirm password*

☐ You accept our [Terms of Service](#) and confirm that you are at least 13 years old.

Рисунок 2.19 - Проста форма [11]

Також назви полів краще розміщувати над рядками введення. Для людини інформація легше сприймається згори донизу.

Також необхідно аби логотип сайт або його назва була клікабельною та вела на головну сторінку. Композиція сайт була однорідною - однакові шрифти,

форми, розташування елементів, а також однакова кольорова палітра. Лівостороннє вирівнювання тексту для легкого сприйняття та читання тексту є не менш важливим.

2.8 Вибір засобів проєктування і розробки

Для зберігання усієї необхідної інформації було обрано СУБД MySQL через її масштабованість, зручність експлуатації та зв'язність. MySQL може підтримувати роботу з базами даних великого розміру, це є доцільним, адже кількість користувачів може збільшуватися, як і кількість питань до тестів та інформації, яку необхідно зберігати для користувачів. Зручність у експлуатації полягає в її легкому адмініструванні, а зв'язність у можливості доступу до бази даних кількох користувачів одночасно з будь-якої точки Internet [12].

Для адміністрування СУБД MySQL було обрано вебдодаток phpMyAdmin через:

- інтуїтивно зрозумілий інтерфейс;
- зручні запити SQL;
- кросплатформений доступ – можливо отримати доступ з будь-якого пристрою з веб-браузером [13].

Для взаємодії з базою даних обрано мову програмування PHP, так як PHP разом з phpMyAdmin дозволяє мати більший контроль над сайтом та базою даних, що допомагає створити власні функції для роботи з даними. А також обрана мова програмування дозволяє включити обробку форм, автентифікацію користувачів та генерацію звітів, що є необхідними складовими проєкту [14].

Для створення інтерфейсу користувача буде використовуватися HTML – мова розмітки гіпертекстових документів та CSS – мова для стилізації веб-сторінок та jQuery, що є бібліотекою JavaScript.

Таким чином, у цьому розділі виділено основні сценарії роботи авторизованого і не авторизованого користувача. Описано основні послідовності, що надаються для реалізації роботи функцій, які має сайт. Спроектовано ER-модель для бази даних та семантичну мережу. Виконано вибір засобів проектування і розробки сайту.

3 РЕАЛІЗАЦІЯ СИСТЕМИ

Однією із основних компонентів розроблюваної системи є база даних. У даному дипломному проєкті при створенні системи будемо використовувати реляційну базу даних.

3.1 Реляційна модель даних

Реляційна модель бази даних – це модель даних, призначена для їх організації у вигляді таблиць та відношень між ними. Відношення між таблицями визначаються за полями в них, що містять подібну інформацію [15].

Розроблену реляційну модель даних наведено на рисунку 3.1, де наведено всі створені таблиці разом з їх полями, що використовуються в проєкті для збереження та оновлення інформації, та зв'язки створені між полями цих таблиць.

Усі об'єкти бази даних ПДР тестів зв'язані між собою за схожими полями, що містяться в таблицях. Ці поля автоматично заповнюватимуться під час збереження нових даних та оновлюватимуться у зв'язаних таблицях, якщо відповідні комірки будуть оновлені у відповідній таблиці.

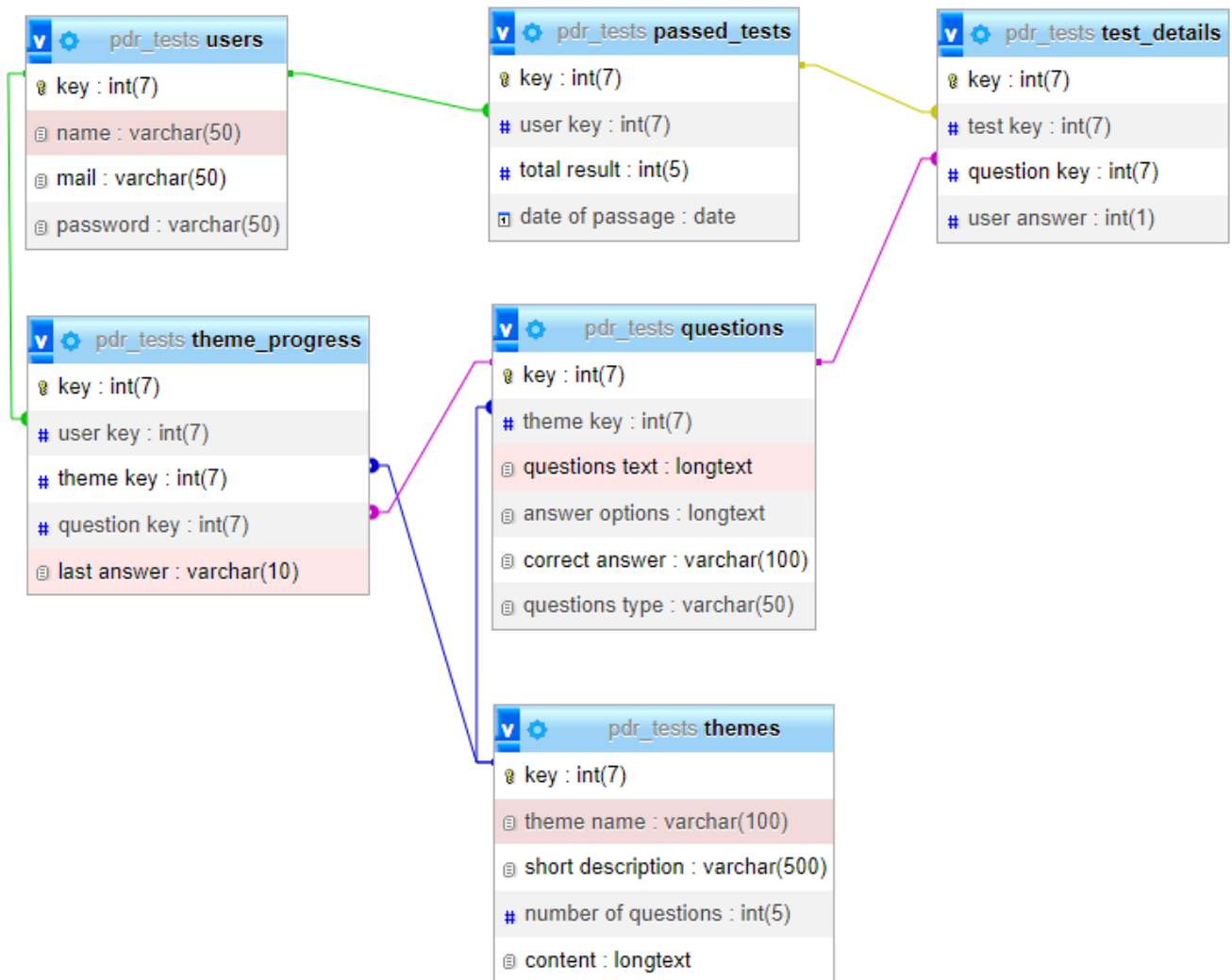


Рисунок 3.1 – Реляційна модель бази даних ПДР тестів

3.2 Взаємодія з базою даних

Для початку роботи з базою даних необхідно встановити з нею з'єднання. Для цього необхідно створити об'єкт класу `mysql` – він призначений для взаємодії з MySQL, виконанням SQL-запитів та отриманням результатів цих запитів. При створенні об'єкту класу `mysql` потрібно вказати назву хосту з базою даних, ім'я користувача бази даних, що має повний доступ до неї, пароль цього користувача та назву бази даних.

На рисунку 3.2 наведено код, двох функцій для з'єднання з базою даних та завершення з'єднання.

```
function connectDB() {  
    $servername = "localhost";  
    $username = "root";  
    $password = "";  
    $dbname = "pdr_tests";  
  
    $db_connect = new mysqli($servername, $username, $password, $dbname);  
  
    if ($db_connect->connect_error) {  
        die("Connect error: " . $db_connect->connect_error);  
    }  
  
    return $db_connect;  
}  
  
function closeDB($db_connect) {  
    $db_connect->close();  
}
```

Рисунок 3.2 - Функції з'єднання з базою даних та завершення з'єднання

Необхідно перевіряти чи вдало пройшло з'єднання з базою даних. Для цього використовується така властивість як `connect_error`, що міститиме рядок з описом помилки, якщо вона виникла. Важливо робити таку перевірку, оскільки це дозволяє обробляти ситуацію, коли з'єднання не було успішним, а також виводити повідомлення про помилку. Ще одним важливим моментом є не забувати закривати встановлене з'єднання після завершення роботи з базою даних. Рекомендується закривати з'єднання, адже це дозволяє серверу бази даних звільняти ресурси, що були виділені для обслуговування.

Властивість `close()` необхідна саме для цього. Для забезпечення з'єднання з базою даних, а також його закриття, було створено дві функції, в окремий файл, що підключається на сторінках де необхідно обробляти запити до бази даних. Це дозволяє уникнути дублювання коду.

Для отримання інформації з бази даних використовується запит з командою SELECT, що повертає вказані стовпці. Це можуть бути одразу всі стовпці з таблиці так і стовпці обрані за певними умовами. Він необхідний для створення тестів, виводу теоретичного матеріалу та особистої інформації авторизованого користувача. Окрім цього запит SELECT використовується для перевірки даних вже при авторизації порівнюючи введену інформацію з тою, що є у таблиці користувачів.

На рисунку 3.3 наведено декілька запитів, що використовують команду SELECT.

```
$sql = "SELECT `key`, `questions text`, `answer options`, `correct answer` FROM `questions` ORDER BY RAND() LIMIT 20";
$sql = "SELECT `key`, `theme name`, `short description`, `content` FROM `themes`";
$sql = "SELECT `key`, `user key`, `total result`, `date of passage` FROM `passed_tests`";
$sql = "SELECT `questions text`, `answer options`, `correct answer`, `theme key` FROM `questions` WHERE `key` IN ($question_key)";
$sql = "SELECT `questions text`, `answer options`, `correct answer`, `theme key` FROM `questions` WHERE `key` IN ($for_query) ORDER BY FIELD(`key`, $for_query)";
```

Рисунок 3.3 - Запити з командою SELECT

Для оновлення даних користувача, а також інформацію, що до прогресу за темами у відповідних таблицях, використовується запит з командою UPDATE, що оновлює значення введені користувачем або отримані після роботи розроблених алгоритмів в заданих стовпцях таблиці.

На рисунку 3.4 наведено приклад запитів з командою UPDATE.

```
$sql = "UPDATE `users` SET `name` = '$new_username', `mail` = '$new_email', `password` = '$new_password' WHERE `key` = $user_id";
```

Рисунок 3.4 - Запити з командою UPDATE

Для додавання нових користувачів, результатів тестів, а також даних, що необхідні для відслідковування прогресу за темами у відповідні таблиці використовується запит з командою INSERT. Він додає рядки до вказаної таблиці, заповнюючи їх відповідними значеннями, що були введені користувачем чи отримані під час виконання алгоритмів проєкту.

На рисунку 3.5 наведено приклад запиту з командою INSERT.

```

$stmt = $conn->prepare("INSERT INTO `users` (`key`, `name`, `mail`, `password`) VALUES (?, ?, ?, ?)");
$stmt->bind_param("iss", $new_key, $new_username, $new_email, $new_password);

$sql = $db_connect->prepare("INSERT INTO `passed_tests` (`key`, `user key`, `total result`, `date of passage`) VALUES (?, ?, ?, ?)");
$sql->bind_param("iiis", $passed_tests_key, $user_id, $total_result, $currentDateTime);

$sql = $db_connect->prepare("INSERT INTO `test_details` (`key`, `test key`, `question key`, `user answer`) VALUES (?, ?, ?, ?)");
$sql->bind_param("iiis", $test_details_key, $passed_tests_key, $questions_keys[$i], $u_answer);

```

Рисунок 3.5 - Запит з командою INSERT

Запит з командою DELETE використовується для видалення записів з таблиці. Такий запит виконується у випадку коли користувач хоче видалити свій обліковий запис. Тоді за його особистим ідентифікатором з бази даних буде видалено усю інформацію, що включає не тільки дані для авторизації, а результати тестувань і прогрес за темами.

На рисунку 3.6 наведено приклад запиту з командою DELETE.

```

$sql = "DELETE FROM `users` WHERE `key` = '$user_id'";

```

Рисунок 3.6 - Запит з командою DELETE

3.3 Авторизація користувача

Через наявність у проєкті ролі авторизованого користувача, що має функціонал, який тільки він може виконувати, постала необхідність у відслідковуванні стану авторизації. Для цього у PHP є механізм сесій, що призначений для зберігання даних на сервері, які використовуються на протязі всіх запитів. Він дозволяє створювати змінні, що будуть зберігатися на сервері та є доступними на всій тривалості сесії, поки користувач не залишить сайт.

Сесії часто використовуються саме для зберігання стану входу, а також передачі даних між різними сторінками сайту. Крім того, для кожного

користувача створюється окрема сесія, що є унікальною і дані у ній доступні лише для цього користувача.

Таким чином у проєкті під час авторизації або реєстрації користувача розпочнеться нова сесія. Це виконується за допомогою функції PHP `session_start()`. Після того як сесію розпочато виконується запит до бази даних для перевірки інформації при авторизації, чи запит на додавання інформації нового користувача.

Коли один із запитів, залежно від обраної дії, було успішно виконано до змінних сесії додається ідентифікатор користувача та його ім'я. Для цього використовується `$_SESSION`, що є асоціативним масивом для зберігання змінних сесії. Далі завдяки цьому масиву на сторінках сайту відбувається проста перевірка умови якої визначають чи присутні додані змінні у поточній сесії, що дозволяє визначити стан користувача як авторизований чи не авторизований і згідно з результатами відображати інформацію доступну певній колі, а також аналізувати успішність за темами для створення тестувань.

Вище було вже згадано, що сесія завершується коли користувач залишає сайт, пороте він також може це зробити натиснувши кнопку виходу в особистому кабінеті. Ця кнопка виконує PHP функцію `session_destroy()`, що видаляє всі дані сесії, що були додані, тим самим завершуючи її.

3.4 Відслідковування прогресу

Для відслідковування прогресу розроблено функцію згідно алгоритму наведеного у розділі 2.2. На початку якого здійснюється перевірка на авторизованість користувача. Це виконується за допомогою механізму сесій, що описаний у попередньому розділі.

Коли авторизацію підтверджено виконується запити з командою `SELECT` до декількох таблиць, а саме `theme_progress` та `themes`, на отримання значення

ключового поля теми, загальної кількості питань за темою з таблиці `themes` та останньої відповіді користувача на питання, ключового поля теми і ключового поля питання, що відповідає темі за якою сформоване запитання, з таблиці `theme_progress`. Запит до таблиці `themes` виконується на отримання усіх стовпців. Запит до таблиці `theme_progress` виконується на отримання стовпців, що відповідають ідентифікатору користувача.

На рисунку 3.7 наведено приклади запитів до цих таблиць.

```
$sql_1 = "SELECT `key`, `number of questions` FROM `themes`";  
$sql_2 = "SELECT `theme key`, `question key`, `last answer` FROM `theme_progress` WHERE `user key` = '$user_id'";
```

Рисунок 3.7 – Запити до таблиць `themes` та `theme_progress`

Стосовно значення останньої відповіді користувача – це значення оновлюється кожного разу при проходженні тесту і лише для тих питань, що були наявні у пройденому тестуванні, звісно коли користувач авторизований. Це поле містить текстове значення чи вірна відповідь (`correct`), чи ні (`wrong`). Завдяки чому легше запам'ятати і вирахувати прогрес, адже не потрібно проходити по всіх результатах попередніх тестів конкретного користувача для знаходження значення останньої відповіді для кожного питання і кожної теми, що збільшило б кількість коду і кількість процесів, що має для цього бути виконана, а також збільшило час виконання підрахунків.

Після отримання необхідних значень виконується підрахунок кількості вірних і невірних відповідей. Також одночасно з цим запам'ятовуються ідентифікатори питань в яких було зроблено помилки, бо надалі це знадобиться аби визначити який тип запитань є проблемним. Далі підраховується кількість питань, на які не було надано жодної відповіді, тобто воно ще не зустрічалося у тесті. І у завершення виконується підрахунок самого прогресу за темою, що визначається у відсотках, відносно кількості правильних відповідей та кількості усіх питань, не враховуючи кількість питань без відповіді.

На рисунку 3.8 наведено код, що виконує всі вище перераховані етапи.

```

$i = 0;
$n = 0;
$correct_answer = array();
$wrong_answer = array();
$wrong_question_key = array();
foreach($theme_key as $theme){
    foreach($last_answer as $answer){
        if ($theme == $themes_key[$i]){
            if ($answer == "correct"){
                $correct_answer[$i] += 1;
            } else if ($answer == "wrong"){
                $wrong_answer[$i] += 1;
                $wrong_question_key[$i][$n] = $question_key[$n];
            }
        }
        $n++;
    }
    $i++;
}

$i = 0;
$without_answer = array();
for($n = 0; $n < count($theme_key); $n++){
    $without_answer[$i] = $number_or_questions[$i] - ($wrong_answer[$i] + $correct_answer[$i]);
    $i++;
}

$i = 0;
$theme_progress = array();
for($n = 0; $n < count($theme_key); $n++){
    $theme_progress[$i] = ($correct_answer[$i] * 100) / ($number_or_questions[$i] - $without_answer[$i]);
    $i++;
}

```

Рисунок 3.8 – Підрахунок прогресу за темою

3.5 Відбір питань

Для вибору питань для тесту розроблено функцію згідно алгоритму наведеного у розділі 2.2. У першу чергу перевіряється авторизація та запит до бази даних на отримання складених тестів. Не обов'язково отримувати усі стовпці таблиці, адже для визначення кількості вистачить одного, у нашому випадку беремо лише ключове поле (кількість тестів, що вже були складені користувачем).

На рисунку 3.9 наведено код, що реалізує етапи перераховані вище.

```
if (isset($_SESSION['user_id'])){
    $sql_0 = "SELECT `key` FROM `passed_tests` WHERE `user key` = '$_SESSION['user_id']'";
    $result = $db_connect->query($sql_0);
```

Рисунок 3.9 – Перевірка авторизації та кількості складених тестів

Далі визначається тема з найменшою кількістю правильних відповідей. Для цього проходимо по всім значенням і порівнюємо їх. Якщо значення наступної теми менше ніж попередньої її значення запам'ятовується. Також запам'ятовується порядковий номер цієї теми.

На рисунку 3.10 наведено частину коду, що реалізує описану вище частину.

```
if ($result->num_rows >= 10){
    $problem_theme = 100;
    $problem_theme_key = array();
    $i = 0;
    for($n = 0; $n < count($theme_key); $n++){
        if ($theme_progress[$i] < $problem_theme){
            $problem_theme = $theme_progress[$i];
            $problem_theme_key[0] = $i;
        }
        $i++;
    }
}
```

Рисунок 3.10 – Визначення теми з найменшою кількістю правильних відповідей

Також виконується додаткова перевірка, адже може бути декілька тем з однаковою найменшою кількістю правильних відповідей. Для цього виконується повторне проходження по масиву зі значеннями кількості неправильних відповідей за темами де порівнюється, визначене у попередній частині коду, найменше значення. Якщо знайдено ще теми з таким самим значенням ми додаємо його до масиву проблемних тем.

На рисунку 3.11 наведено частину коду, що реалізує описану вище частину.

```

    $i = 0;
    $j = 1;
    for($n = 0; $n < count($theme_key); $n++){
        if ($theme_progress[$i] == $problem_theme){
            $j++;
            $problem_theme_key[$j] = $i;
        }
        $i++;
    }

```

Рисунок 3.11 – Пошук схожих проблемних тем

На рисунку 3.12 наведено частину коду, що реалізує визначення типу питання в якому за темою було зроблено більше помилок.

```

    $i = 0;
    $questions_type = array();
    $text = array();
    $image = array();
    foreach($theme_key as $theme){
        $for_query = '';
        for($n = 0; $n < count($wrong_question_key[$i]); $n++){
            if ($wrong_question_key == end($questions_keys)) {
                $for_query .= $wrong_question_key[$i][$n];
            } else {
                $for_query .= $wrong_question_key[$i][$n].',';
            }
        }
        $sql = "SELECT `questions type` FROM `questions` WHERE `key` IN ($for_query) ORDER BY FIELD(`key`, $for_query)";
        $result = $db_connect->query($sql);
        while ($row = $result->fetch_row()) {
            if ($row["questions type"] == "text"){
                $text[$i] += 1;
            } else if ($row["questions type"] == "image"){
                $image[$i] += 1;
            }
        }
        $i++;
    }

    for($n = 0; $n < count($theme_key); $n++){
        if ($text[$n] > $image[$n]){
            $questions_type[$n] = "text";
        } else if ($text[$n] < $image[$n]){
            $questions_type[$n] = "image";
        }
    }
}

```

Рисунок 3.12 – Визначення проблемного типу питання

Для цього виконується запит до таблиці з питаннями з якої отримується значення типу питання за ключовим полем. Значення ключів відповідають лише тим питанням в яких були помилки. Отримані значення записуються у масив, після чого відбувається проходження по масиву, де порівнюються значення для підрахунку кількості типів тих питань, які найчастіше зустрічаються.

Наступний етап (рис. 3.13) після визначення проблемного типу питання, це визначення кількості питань для тесту, орієнтовно прогресу за проблемною темою і також отримання з бази даних цієї кількості питань.

```

$questions_for_text = array();
$general_questions_count = 0;
if ($problem_theme[0] < 46){
    for($n = 0; $n < count($problem_theme); $n++){
        $questions_count = 0;
        if ($problem_theme[$n] < 15){
            if (($general_questions_count + 7) > 20){
                $questions_count = 20 - $general_questions_count;
            } else {
                $questions_count = 7;
            }
        } else if ($problem_theme[$n] < 25){
            if (($general_questions_count + 5) > 20){
                $questions_count = 20 - $general_questions_count;
            } else {
                $questions_count = 5;
            }
        } else if ($problem_theme[$n] < 35){
            if (($general_questions_count + 3) > 20){
                $questions_count = 20 - $general_questions_count;
            } else {
                $questions_count = 3;
            }
        } else if ($problem_theme[$n] < 46){
            if (($general_questions_count + 2) > 20){
                $questions_count = 20 - $general_questions_count;
            } else {
                $questions_count = 2;
            }
        }
        $general_questions_count += $questions_count;
    }

    $sql_3 = "SELECT `key`, `questions text`, `answer options`, `correct answer` FROM `questions`
    WHERE `theme key` = '$sql_2[$problem_theme_key[$n]]', `questions type` = '$questions_type[$n]' ORDER BY RAND() LIMIT $questions_count";
    $result = $db_connect->query($sql_3);
    if ($result) {
        while ($row = $result->fetch_row()) {
            array_push($questions_for_text, $row);
        }
    }
}
}

```

Рисунок 3.13 – Додава питань до тесту

Якщо прогрес за темою менше 15% буде обрано 7 питань за темою, якщо менше 25% - 5 питань, якщо менше 35% - 3 питання і якщо менше 46% - 2 питання. Також під час визначення цієї кількості відбувається перевірка на кількість вже доданих питань до тесту, адже це не має перевищувати 20 питань,

у разі коли, спираючись на успішність теми, цю кількість буде перевищено, надаватиметься менша кількість. За визначеною кількістю до бази даних виконуються запит на отримання питань і їх віріантів відповідей.

На рисунку 3.14 наведено код, що доповнює залишок питань для повного тесту. Якщо проблемних тем було декілька, випадкових питань може залишатися ще певна кількість тих, що потрібно дібрати ти повного тесту. Для цього від загальної кількості, що складає 20 питань, віднімається кількість вже доданих після аналізу тем питань і з бази даних випадковим чином отримується ця кількість. Якщо користувач має високу успішність усі 20 питань будуть випадковими. Також якщо за проблемними темами уже було дібрано 20 питань додаткова вибірка випадкових не потрібна.

```
if ($general_questions_count > 0 && $general_questions_count != 20 && $general_questions_count < 20){
    $number = 20 - $general_questions_count;
    $sql_4 = "SELECT `key`, `questions text`, `answer options`, `correct answer` FROM `questions` ORDER BY RAND() LIMIT $number";
    $result = $db_connect->query($sql_4);
    if ($result) {
        while ($row = $result->fetch_row()) {
            array_push($questions_for_text, $row);
        }
    }
} else if ($general_questions_count == 0){
    $sql_4 = "SELECT `key`, `questions text`, `answer options`, `correct answer` FROM `questions` ORDER BY RAND() LIMIT 20";
    $result_questions_for_text = $db_connect->query($sql_4);
}
```

Рисунок 3.14 – Доповнення тесту питаннями

3.6 Використання jQuery

У проєкті jQuery використовується маніпулювання DOM елементами та обробки подій замість великої кількості коду для ручного вибору та зміни елементі використання методів jQuery.

jQuery для перемикування між формами авторизації та реєстрації використовується обробка події натискання на кнопку, що розміщуються в кінці форми. Для визначення на який елемент необхідно натиснути аби перемкнутися

між формами використовується селектор `$(".body .login_div .to_register")`, що знаходить елемент з класом `to_register`, який знаходиться в елементах з класом `login_div` та `body`. Далі використовується метод `.on("click", function(){...})`, який виконується при кліку на знайдений елемент. Після чого відбувається зміна стилів для елементів:

- елемент з класом `register_div` стає видимим, для нього встановлюється властивість `display` зі значенням `block`;
- елемент з класом `login_div` стає невидимим, для нього встановлюється властивість `display` зі значенням `none`.

На рисунку 3.15 наведено код, що здійснює зміну відображення форм авторизації та реєстрації.

```
$(".body .login_div .to_register").on("click", function(){
    $('.body .register_div').css({"display": "block"});
    $('.body .login_div').css({"display": "none"});
});

$(".body .login_div .to_login").on("click", function(){
    $('.body .register_div').css({"display": "none"});
    $('.body .login_div').css({"display": "block"});
});
```

Рисунок 3.15 – Код для зміни відображення форм авторизації та реєстрації

Таким самим шляхом відбувається розкриття більш детальної інформації правил ПДР за темами та детальної інформації пройденого тестування.

При проходженні тесту відображається індикатор прогресу і аби він оновлювався актуальною інформацією використовується jQuery. При натисканні на кнопки для перемикання між питаннями збільшується або зменшується ширина блоку, що показує прогрес на 5% – це значення визначено відносно цілої ширини блоку та усієї кількості питань, а також збільшує або зменшує текстове

значення прогресу аби також відобразити його у відсотках. Текстове значення так само змінюється на 5%.

На рисунку 3.16 наведено частину коду, що зміню текстове та стильове відображення індикатору прогресу проходження тесту.

```

$(".test_form .test_navigation_buttons .prev").on("click", function(){
    if (!$(this).hasClass("disabled")) {
        prevSlideIndex = slideIndex - 1;
        n = -1;
        showSlides(slideIndex += n);
        $(".test_form .test_progress .test_bar").css({"width": 5*prevSlideIndex + "%"});
        $(".test_form .test_progress .test_bar").text(5*prevSlideIndex + "%");
    }
});

$(".test_form .test_navigation_buttons .next").on("click", function(){
    if (!$(this).hasClass("disabled")) {
        nextSlideIndex = slideIndex + 1;
        n = 1;
        showSlides(slideIndex += n);
        $(".test_form .test_progress .test_bar").css({"width": 5*nextSlideIndex + "%"});
        $(".test_form .test_progress .test_bar").text(5*nextSlideIndex + "%");
    }
});

```

Рисунок 3.16 – Код для зміни тексту та стилю для індикатору прогресу проходження тесту

Для перемикання між питаннями тесту використовується функція `showSlides(n)`. Функція отримує номер питання до якого виконується перехід та колекцію елементів, що є блоками питань. Вона перевіряє чи не має передане значення номеру питання неприпустиме значення, тобто більше або менше за загальну кількість питань. Далі функція проходить по всім знайденим питанням та встановлює властивість `display` зі значенням `none`, аби їх приховати. Після чого для питання, номер якого було передано встановлюється властивість `display` зі значенням `block`, аби його відобразити. В кінці відбувається перевірка аби замінити кнопку для переходу вперед на кнопку перегляду результатів, якщо це останнє питання.

На рисунку 3.17 наведено код функції для перемикавання між питаннями тесту.

```
function showSlides(n) {
    let i;
    let slides = document.getElementsByClassName("slide");
    if (n > slides.length) {
        slideIndex = 1;
    }
    if (n < 1) {
        slideIndex = slides.length;
    }
    for (i = 0; i < slides.length; i++) {
        slides[i].style.display = "none";
    }
    slides[slideIndex-1].style.display = "block";

    if (slideIndex == 1) {
        $(".test_form .test_navigation_buttons .prev").addClass("disabled");
    } else {
        $(".test_form .test_navigation_buttons .prev").removeClass("disabled");
    }

    if (slideIndex == slides.length) {
        $(".test_form .test_navigation_buttons .next").css({"display": "none"});
        $(".test_form .test_navigation_buttons .test_result").css({"display": "block"});
    } else {
        $(".test_form .test_navigation_buttons .next").css({"display": "block"});
        $(".test_form .test_navigation_buttons .test_result").css({"display": "none"});
    }
}
```

Рисунок 3.17 – Функція для перемикавання між питаннями тесту

3.7 Реалізація інтерфейсу

Для реалізації інтерфейсу використовувалися HTML та CSS. Для кожної сторінки створено окремі файли зі стилями та шаблонів сторінок для зменшення нагромадження коду, а також для уникнення повторення коду. До прикладу шаблони <header> та <footer> та набір стилів до них винесені в окремі файли, адже мають бути присутні на всіх сторінках.

Тег <header> представляє собою ступний контент та навігаційні засоби і може використовуватися як заголовок сайту. Саме тут розміщується логотип сайту та його меню.

Елемент використовується для створення маркованого списку, а для створення елементів списку. Ці елементи часто використовуються для впорядкування навігації сайту. Тут також присутні елементи <a>, що використовуються для визначення посилань для переходу на визначені сторінки. Заголовок сайту складається з основного списку, що містить текстовий заголовок сайту, та меню сайту, що містить другий список для елементів навігації.

На рисунку 3.18 наведено реалізацію заголовку сайту.

```
<header>
  <div class="nav-bar">
    <ul>
      <li class="nav-bar-left">
        <a class="text_logo_header" href="home.php">ПДР Тестування</a>
      </li>
      <li class="nav-bar-center">
        <ul class="menu-list">
          <li>
            <a href="pravula_pdr.php">Правила ПДР</a>
          </li>
          <li>
            <a href="test.php">Тестування</a>
          </li>
          <?php if (isset($_SESSION['user_id'])) { ?>
            <li>
              <a href="passed_tests.php">Пройдені тестування</a>
            </li>
          <?php } ?>
        </ul>
      </li>
      <li class="nav-bar-right">
        <a class="log_in_button" href="<?php echo (isset($_SESSION['user_id'])) ? 'user.php' : 'login_register.php'; ?>">
          Особистий кабінет
        </a>
      </li>
    </ul>
  </div>
</header>
```

Рисунок 3.18 – Реалізація заголовку сайту

Тег <body> є одним з ключових тегів при створенні сайту, що містить основний контент усіх сторінок. До прикладу цей тег на головній сторінці містить такі теги як:

- <div> що призначений для розділення контенту;
- <h2> – спеціальний тег для заголовків;
- <table> – для представлення даних у вигляді таблиці;

- <tr> – для створення одного рядка таблиці;
- <th> – для створення однієї комірки заголовка таблиці;
- – для відображення зображення на сторінці.

На рисунку 3.19 наведено приклад створення таблиці на головній сторінці.

```
<table>
  <tr>
    <th></th>
    <th>Без авторизації</th>
    <th>З авторизацією</th>
  </tr>
  <tr>
    <td>Перегляд теоретичних матеріалів</td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
  </tr>
  <tr>
    <td>Проходження тестування</td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
  </tr>
  <tr>
    <td>Перегляд результатів пройденого тестування</td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
  </tr>
  <tr>
    <td>Варіативність питань у тестах</td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
  </tr>
  <tr>
    <td>Аналіз помилок і враховування їх у наступних тестах</td>
    <td><div class="check_no"><?php echo $check_icon; ?></div></td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
  </tr>
  <tr>
    <td>Статистика результатів за темами питань</td>
    <td><div class="check_no"><?php echo $check_icon; ?></div></td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
  </tr>
  <tr>
    <td>Перегляд результатів попередніх тестувань</td>
    <td><div class="check_no"><?php echo $check_icon; ?></div></td>
    <td><div class="check_yes"><?php echo $check_icon; ?></div></td>
  </tr>
</table>
```

Рисунок 3.19 – Створення таблиці на головній сторінці

Також на сторінках у самому кінці відображається тег `<footer>`. Зазвичай в ньому розміщується інформація про автора, або власника сайту, інформація про авторські права або додаткова навігація разом з логотипом. Через те що заголовок сайту має фіксовану позицію, тобто залишається на тому ж місці, вгорі сторінки, при прокрутці вниз і вгору, у `<footer>` не розміщено додаткове навігаційне меню, а лише наявна назва сайту.

Крім того на сайті присутні кнопки та форми, що також створені з використання спеціальних елементів HTML. Так для створення кнопок зазвичай використовується `<button>`, що створює клікабельну кнопку.

Для створення форм використовується елемент `<form>`. Він представляє собою розділ документа, що містить набір інтерактивних елементів, які користувач може відправити на сервер. У своїх атрибутах елемент `<form>` має обов'язково мати `action` – адресу, за якою обробляється передана формою інформація та `method` – метод, який браузер використовує для відправки форми. Атрибут `method` може приймати такі значення як `post` та `get`. Відмінність між ними у тому, що дані передані методом `get` додаються до адреси атрибута `action`.

Для заповнення форми необхідною інформацією використовується інтерактивний елемент `<input>`, що необхідний для отримання даних від користувача. Він має атрибут `type`, який визначає як відображатиметься елемент і у розроблених формах він приймає такі значення як:

- `email` – для зазначення електронної пошти;
- `text` – для однорядкового тексту;
- `password` – для однорядкового тексту, чиє значення буде приховане за символом “зірочка”.

Поруч з елементом `<input>` зазвичай використовується елемент `<label>`. Він необхідний для підпису інтерактивного елемента. Його атрибут `for` має містити ID елемента для якого призначений цей підпис, тому при натисканні на елемент `<input>` курсор користувача автоматично потрапить до відповідного елемента `<label>` аби він міг його заповнити.

На рисунку 3.20 наведено код форма авторизації та реєстрації.

```

<div class="login_div">
  <h2 class="login_title">Автори́зація</h2>
  <form class="login" action="login_register_service.php" method="post">
    <label for="username">Іме́'я користу́вача</label>
    <input type="text" id="username" name="username" required>

    <label for="password">Паро́ль</label>
    <input type="password" id="password" name="password" required>

    <button type="submit" name="login">Автори́зуватися</button>
  </form>
  <button class="to_register">Зареє́струватися</button>
</div>

<div class="register_div">
  <h2 class="register_title">Реє́страція</h2>
  <form class="register" action="login_register_service.php" method="post">
    <label for="username">Іме́'я користу́вача</label>
    <input type="text" id="username" name="username" required>

    <label for="email">По́шта</label>
    <input type="email" id="email" name="email" required>

    <label for="password">Паро́ль</label>
    <input type="password" id="password" name="password" required>

    <button type="submit" name="register">Зареє́струватися</button>
  </form>
  <button class="to_login">Автори́зуватися</button>
</div>

```

Рисунок 3.20 – Код форм авторизації та реєстрації

Для забезпечення більшої гнучкості інтерфейсу у стилях проєкту використовується властивість CSS flex, що надає елементам здатність розтягуватися та стискатися заповнюючи собою доступне вільне місце. Ця властивість допомагає у більш простому створенні стовпців з елементів на сторінці, а також визначені їх розміру і розміщення на екранах різного розміру.

На рисунку 3.21 наведено приклад використання властивості flex у проєкті для впорядкування елементів, що відображають стислу інформацію пройдених тестувань і вибудовані у лінію, що містить по три елементи і переносить наступні три на новий рядок.

```

.body .passed_tests {
  display: flex;
  flex-direction: row;
  flex-wrap: wrap;
  padding: 15px;
}

.body .passed_test {
  width: calc(100% / 3.25);
  margin: 5px;
  color: ■ #ffffff;
  text-align: center;
  padding: 5px;
  border-radius: 22px;
  cursor: pointer;
}

```

Рисунок 3.21 – Приклад використання властивості flex у проєкті

Наявність властивості flex надає можливість встановити інші властивості, що також допомагають у визначенні гнучкості елементів. Вони також наведені на рисунку ще і до цих властивостей належить flex-direction та flex-wrap.

Властивість flex-direction – вказує на те як flex елементи будуть розміщуватися відносно головної осі та напрямку. Серед найвикористовуваніших значень row – елементи буде упорядковано вздовж головної осі, column – елементи буде упорядковано вздовж поперечної осі.

Властивість flex-wrap – задає правило виводу елементів flex – в один рядок чи кілька. Це дозволяє або забороняє переносити елементи на наступний рядок, якщо екран перегляду зменшується.

Також завдяки властивості flex можливо задати ширину елементів таку як `width: calc(100% / 3.25)`, що дозволяє вирахувати розмір кожного елементу відносно всієї ширини батьківського елементу і наприклад побудувати елементи всередині нього у три рівні за розміром колонки.

4 РЕЗУЛЬТАТИ РОБОТИ

Головна сторінка - перше, що бачить користувач, відвідуючи сайт вперше. На ній важливо розмістити коротку інформацію, що зверне увагу користувача до продукту, проєкту чи компанії. На рисунку 4.1 наведено розроблену головну сторінку проєкту.

ПДР Тестування

Правила ПДР Тестування

Особистий кабінет



Складай іспит на отримання водійського посвідчення легко!

Тестування є зручним методом навчання, адже підготуєш до найголовнішого іспиту і миттєво можеш перевірити результати проходження. Крім того, дозволяєш більш реалістично оцінити власні сили. Кандидати у водії звикнуть до такого формату роботи і менше нервуватимуть під час іспиту у МРЕО.

Наша система допоможе підвищити рівень знань кандидатів у водії, заохочувати виконання правил за темами у попередніх тестах і пропонує більше кількості запитань за цими темами.

ПЕРЕВАГИ АТОРИЗАЦІЇ

	Без авторизації	З авторизацією
Перегляд теоретичних матеріалів	✓	✓
Пройдення тестування	✓	✓
Перегляд результатів пройденого тестування	✓	✓
Варіативність питань у тестах	✓	✓
Аналіз помилок і врахування їх у наступних тестах	✗	✓
Статистика результатів за темами питань	✗	✓
Перегляд результатів попередніх тестувань	✗	✓

25%

55%

100%

Покращуй свої знання разом з нами та складай іспит з першої спроби!

ПДР Тестування

Рисунок 4.1 – Головна сторінка проєкту

На головні сторінці розміщено інформацію стосовно того, чому саме варто практикуватися за допомогою тестів, які переваги має авторизований користувач та просте гасло. Також у верхній частині головної сторінки розміщується навігаційне меню, завдяки якому можливо пересуватися між різними сторінками сайту. Так у лівій частині меню розміщено текстовий логотип, у правій – посилання на сторінку з правил ПДР, сторінку тестування та особистого кабінету.

На рисунку 4.2 наведено сторінку правил ПДР.

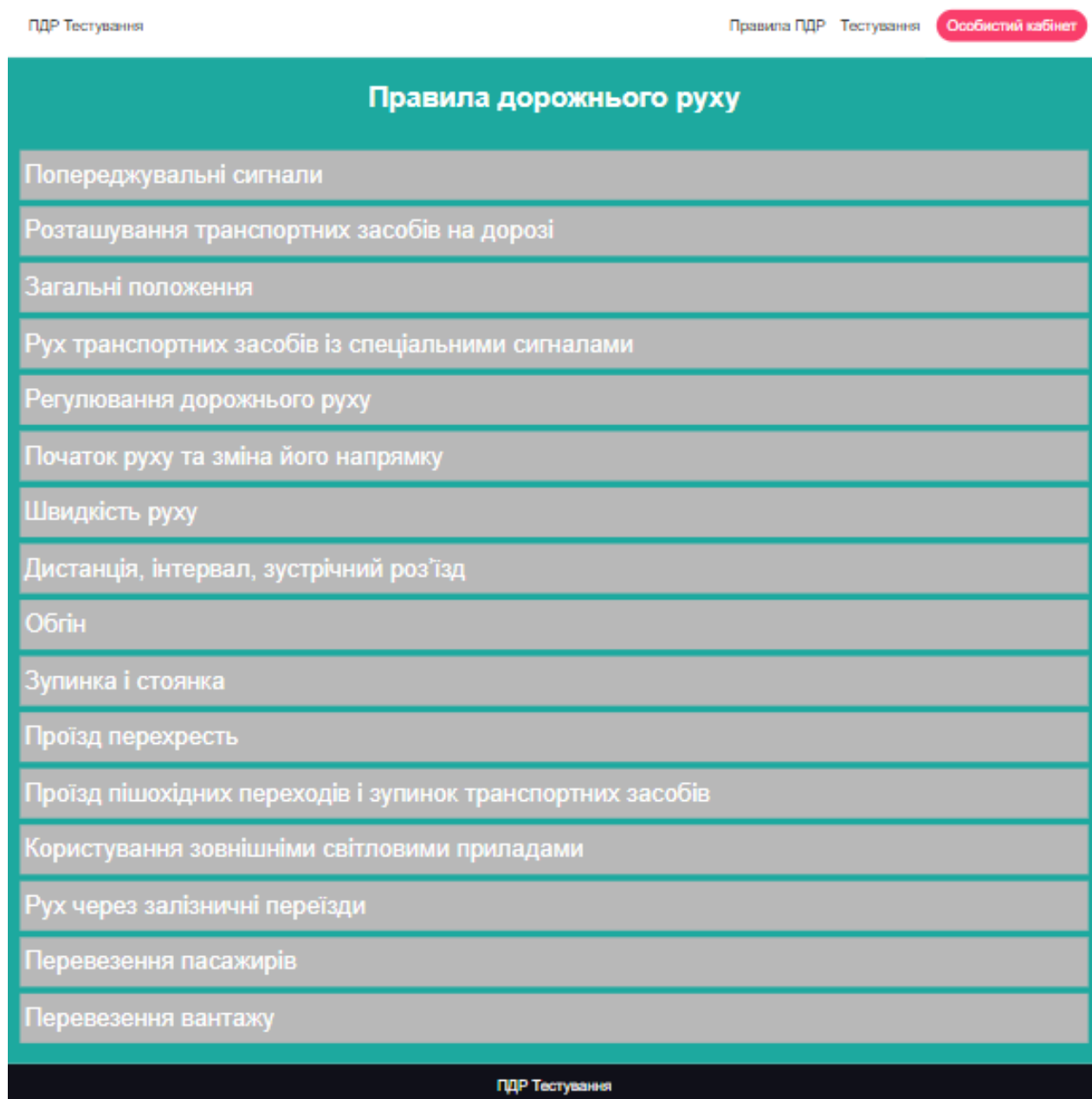


Рисунок 4.2 – Сторінка правил ПДР

Спочатку завантаження сторінки відображаються лише заголовки основних розділів, через їх велику кількість. При наведенні миші вони виділяються блакитним кольором, завдяки чому зрозуміло, що заголовок є клікабельним. Кожний розділ також має додаткові підпункти, що відображаються при натисканні на відповідний розділ. Вони також виділяються кольором при наведенні, адже містять розгорнуту інформацію стосовно теми.

На рисунку 4.3 наведено приклад розгорнутого підпункту теми та повної теорії до нього.

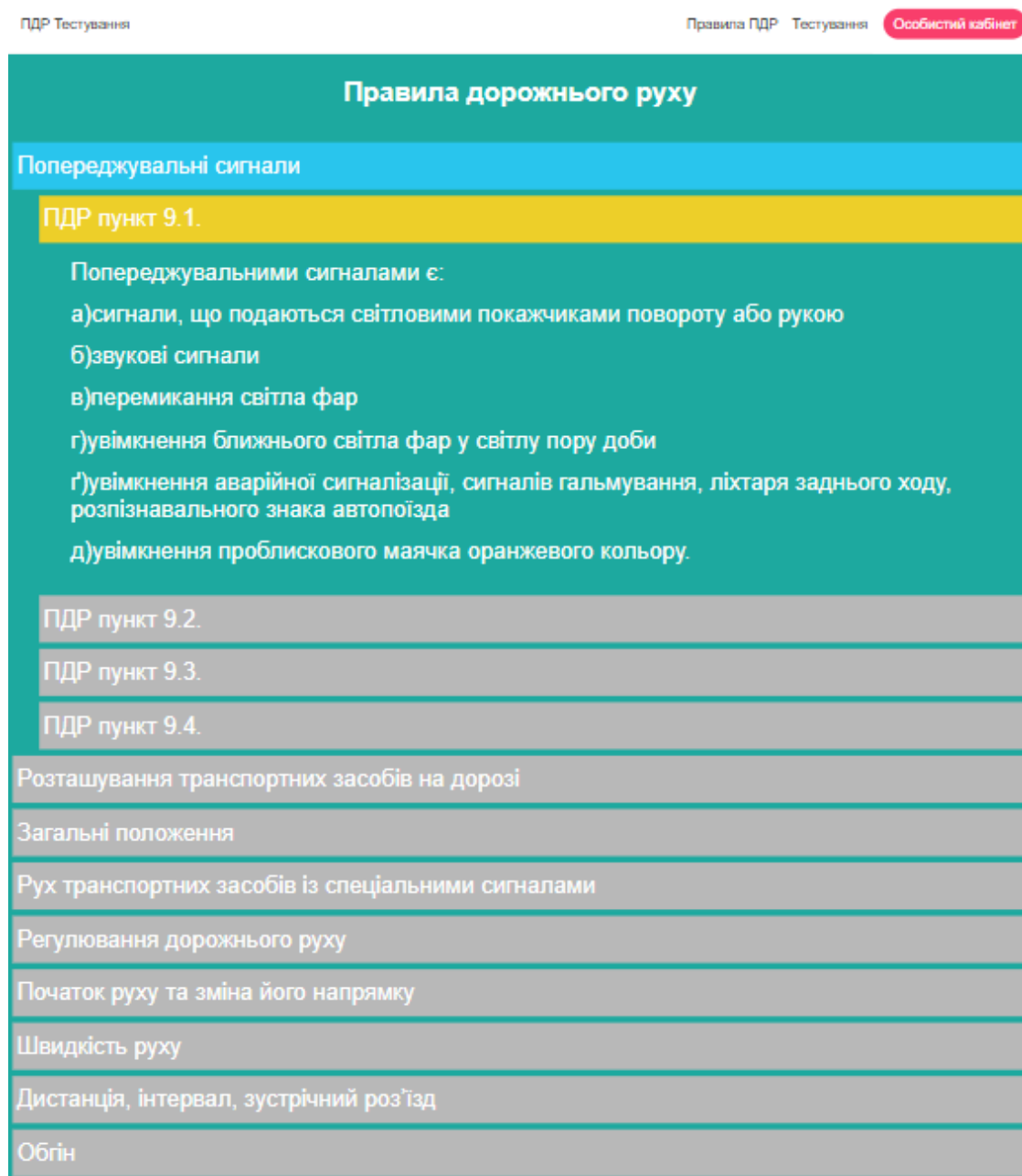


Рисунок 4.3 – Розгорнутий підпункт за темою

На рисунку 4.4 наведено сторінку тестування. Тест складається з 20 питань, кожне з яких з'являється на екрані одне за одним. У верхній частині тесту розміщується індикатор прогресу у відсотках. У нижній частині тесту розміщено кнопки для навігації між питаннями.

ПДР Тестування Правила ПДР Тестування **Особистий кабінет**

5%

Чи повинні Ви як водій подавати сигнали світловим показником перед перестроюванням?

Виберіть одну відповідь

- ☐ Повинен тільки за наявності інших учасників дорожнього руху.
- ☐ Повинен завжди.
- ☐ Зобов'язаний, якщо встигає їх ввімкнути.

Попереднє Наступне

Рисунок 4.4 – Сторінка тестування

На рисунку 4.5 наведено вигляд завершення тестування, коли всі питання вже пройдено.

ПДР Тестування Правила ПДР Тестування **Особистий кабінет**

100%

Водії оперативних транспортних засобів, виконуючи невідкладне службове завдання, увімкнувши проблiсковий маячок червоного або синього кольору, можуть відступати від:

Виберіть одну відповідь

- ☐ Правил маневрування й обгону.
- ☐ Правил проїзду залізничних переїздів.
- ☐ Вимог сигналів регулювальника.
- ☒ Відповіді 1, 2 і 3.

Попереднє Результат

Рисунок 4.5 – Завершення тесту

Тут кнопка для переходу на наступне питання змінюється на кнопку для перегляду результатів тестування. Також поки користувач її не натиснув у нього ще є можливість повернутися до попередніх питань, аби змінити відповідь.

На рисунку 4.6 наведено сторінку перегляду результатів тестування.

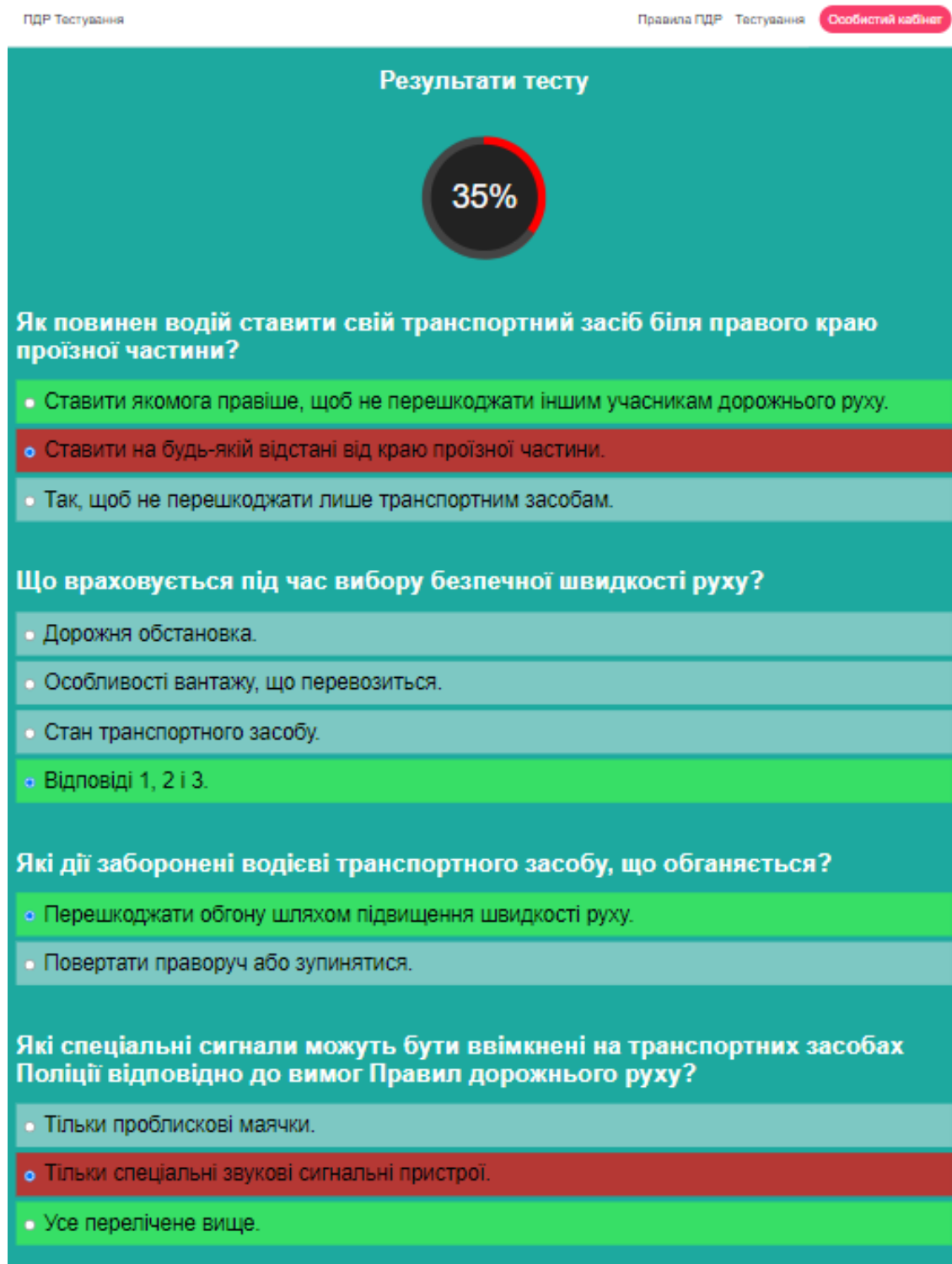


Рисунок 4.6 – Сторінка перегляду результатів тестування

У верхній частині відображається загальний результат у відсотках. Нижче виводяться всі питання тесту з варіантами відповідей серед яких варіант питання, що був обраний користувачем, відповідно відзначено відміченою радіокнопкою. Якщо користувач помилився, що відповідь виділяється червоним, а правильна – зеленим. Якщо ж він відповів вірно, його відповідь виділяється зеленим.

На рисунку 4.7 та 4.8 наведено форми авторизації та реєстрації. Вони розміщені на одній сторінці, тому користувач може перемикатися між ними використовуючи відповідну кнопку, що знаходиться під формою.

ПДР Тестування Правила ПДР Тестування Особистий кабінет

Авторизація

Ім'я користувача

Пароль

Авторизуватися

Зареєструватися

Рисунок 4.7 – Форма авторизації

ПДР Тестування Правила ПДР Тестування Особистий кабінет

Реєстрація

Ім'я користувача

Пошта

Пароль

Зареєструватися

Авторизуватися

Рисунок 4.8 – Форма реєстрації

На рисунку 4.9 наведено форму для редагування даних користувача. У полях виводяться поточні дані користувача, які можна змінити ввівши нові значення у відповідні поля і натиснувши кнопку “Зберегти зміни”. Також під формою є кнопка для виходу з облікового запису.

Рисунок 4.9 – Форма редагування даних користувача

На рисунку 4.10 наведено сторінку з переліком попередніх тестувань. У переліку міститься коротка інформація – дата проходження та загальний результат. Також кожний елемент підсвічується кольором, що визначається за загальним результатом. Якщо значення менше за 50% - це червоний колір, якщо значення від 50% до 90% - це оранжевий колір, більше 90% - зелений колір.

Дата проходження тестування	Загальний результат
2023-11-29	20
2023-11-30	40
2023-11-30	50
2023-11-30	75

Рисунок 4.10 – Сторінка списку попередніх тестувань

Для перегляду більш детальної інформації з пройденого тестування, необхідно натиснути на тест, що цікавить. Після цього перелік пройдених тестів буде сховано і відображено результати обраного тестування (рис. 4.11).

ПДР Тестування
Правила ПДР
Тестування
Пройдені тестування
Особистий кабінет

Повернутися до списку
Дата проходження тестування - 2023-11-30

75%

Чи повинні Ви як водій подавати сигнали світловим покажчиком перед перестроюванням?

- ☐ Повинен тільки за наявності інших учасників дорожнього руху.
- ☒ Повинен завжди.
- ☐ Зобов'язаний, якщо встигає їх ввімкнути.

Як повинні рухатися водії нерейкових транспортних засобів на дорогах із двома смугами й більше для руху в кожному напрямку?

- ☐ Повинні рухатися в будь-якій смузі незалежно від інтенсивності руху.
- ☐ Повинні вибирати смугу для руху залежно від інтенсивності руху.
- ☒ Повинні рухатися якнайближче до правого краю проїзної частини.
- ☐ Повинні рухатися якнайближче до правого краю проїзної частини, крім випадків, коли виконуються випередження, об'їзд або перестроювання перед поворотом ліворуч чи розворотом.

Чи дозволено водієві транспортного засобу з увімкненим проблісковим маячком синього або червоного кольору рухатися на заборонний сигнал регулювальника?

- ☐ Дозволено завжди.
- ☒ Заборонено.

З якої смуги дозволено виконати поворот при в'їзді на перехрестя, де організовано круговий рух?

- ☐ З крайньої правої.

Рисунок 4.11 – Сторінка перегляду деталей раніше пройденого тестування

Вони мають схожий вигляд до сторінки перегляду результатів тестування, що відображаються одразу після завершення тесту. Відмінність полягає лише у наявності кнопки повернення до списку пройдених тестів, а також підписом, що вказує на дату проходження тестування.

На рисунку 4.12 наведено сторінку, що містить список тем і прогрес за ними, що виводиться у відсотках.

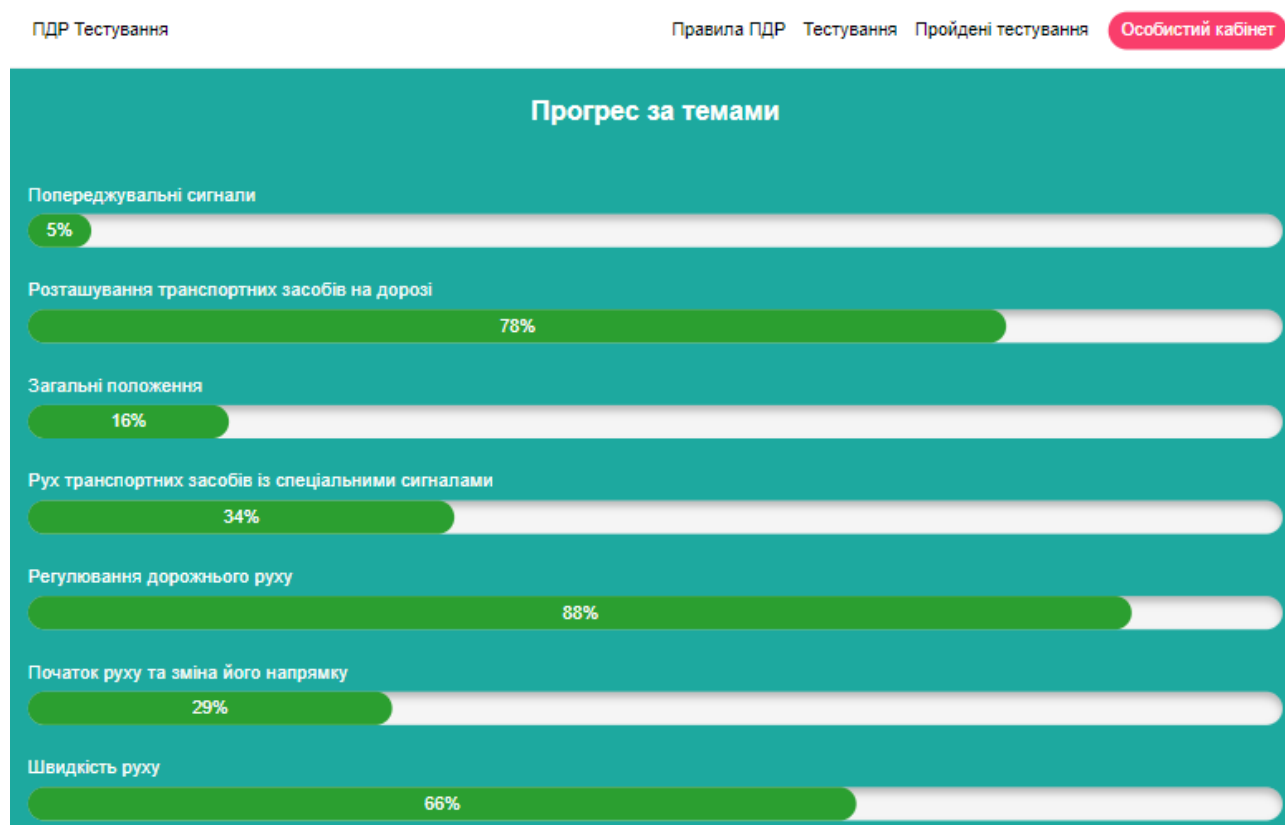


Рисунок 4.12 – Сторінка прогресу за темами

ВИСНОВКИ

Таким чином, у магістерській роботі було обрано та проаналізовано існуючі аналоги, на основі яких складено список вимог, які мають задовольняти систему. За складеним списком вимог виділено основні сценарії роботи, що описують можливості авторизованого і не авторизованого користувача. Базуючись на основних функціях, які надає сайт, описано основні послідовності. Визначено яка інформація зберігатиметься в базі даних, на основі чого спроектовано ER-модель та семантичну мережу. Також було виділено основні правила за якими будувався інтерфейс користувача.

Спираючись на спроектовані моделі було обрано СУБД MySQL та phpMyAdmin для її адміністрування, а для взаємодії з даними – PHP, HTML, CSS та jQuery використовуватимуться для інтерфейсу користувача.

У процесі проєктування було проведено тестування сайту у трьох напрямках. Перший напрямок тестування включав перевірку на коректну відправку запитів до бази даних і отримання очікуваної інформації у відповідь. Під час виконання цього етапу тестування було виявлено, що під час виконання запиту не отримувалися всі пости, що очікувалися. Запити здійснювалися до таблиці питань та результатів тестування. Цю проблему було вирішено додаванням необхідних назв стовпців до запиту, а також коригуванням переліку значень, що відповідають ключовому полю, за якими суворо визначалось які стовпці мають бути отримані.

Другий напрямок тестування включав перевірку на запис інформації до бази даних. Під час виконання цього етапу тестування було виявлено, що для нових записів створювалися однакові ключі, що викликало помилку і зберігало лише перше значення. Цю проблему було вирішено зміною функції, що генерує випадкове число.

Третій напрямок тестування включав перевірку роботи інтерфейсу користувача. Під час цього етапу тестування було перевірено кожну сторінку

сайту на коректність відображення і було виявлено проблеми з перемиканням між питаннями та відображенням інформації, що призначена лише авторизованому користувачу. Для вирішення цих проблем було додано перевірки на кількість питань та номер питання до скрипту jQuery, а також додано перевірки для визначення авторизації користувача при відображенні відповідних елементів.

У результаті проєкт було реалізовано у вигляді сайт з використанням ХАМРР, що забезпечувала локальний сервер для проведення розробки і її тестування. Створений сайт надає можливість перегляду теоретичної інформації, проходження тестувань та перегляд їх результатів, авторизацію та реєстрацію користувача, перегляд авторизованим користувачем результатів попередніх тестувань та прогресу за темами, а врахування помилок у пройдених тестах за темами і додавання більшої кількості питань з тих тем в наступних тестуваннях.

Розроблений сайт має підвищити рівень теоретичних знань майбутніх водіїв для складання екзамену ПДР.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. У 2022 році в Україні видали більше водійських посвідчень ніж торік. URL: <https://autoconsulting.ua/article.php?sid=52981> (дата звертання: 21.10.2023).
2. Які шанси здати іспит на права з першого разу: статистика. URL: https://auto.24tv.ua/yaki_shansy_zdaty_ispyt_na_prava_z_pershoho_razu_statystyka_n45788 (дата звертання: 21.10.2023).
3. Про Дорожній Рух. URL: <https://roadrules.com.ua/pdr-testi/ekzamen-online.html> (дата звертання: 22.10.2023).
4. Green Way Online. URL: <https://green-way.com.ua/uk> (дата звертання: 22.10.2023).
5. Vodiya.ua. URL: <https://vodiya.ua/> (дата звертання: 22.10.2023).
6. Поняття ER-моделі. Поняття сутності (entity). Атрибути. Види атрибутів. URL: <https://www.bestprog.net/uk/2019/01/24/the-concept-of-er-model-the-concept-of-essence-and-communication-attributes-attribute-types-ua/> (дата звертання: 23.10.2023).
7. Основні поняття моделі ENTITY-RELATIONSHIP (Сутність - Зв'язку). URL: https://stud.com.ua/97322/informatika/osnovni_ponyattya_modeli_entity_relationshi_p_sutnist_zvyazku (дата звертання: 07.12.2023).
8. Пасічник В. В., Юнчик В. Л., Федонюк А. А. Добір електронних навчальних ресурсів з використанням рекомендаційної системи: матеріали XII міжнародної науково-практичної конференції, м.Луцьк-Світязь, 2-4 червня 2023р. С.129 – 131.
9. Брусенцев В. О., Асєєв Г. Г., Ткачук М. А. Інтелектуальні системи: конспект лекцій. Харківська державна академія культури, 2022р., Л. 3-4. С.34 – 78.

10. Семантичні та фреймові мережі. URL: <https://studfile.net/preview/3270294/> (дата звертання: 23.10.2023).
11. Як зробити інтерфейс «user friendly»: 17 прийомів, що працюють. URL: <https://hostiq.ua/blog/ukr/user-friendly-interface/> (дата звертання: 05.12.2023).
12. Основні переваги субд MySQL. URL: <https://studfile.net/preview/5607354/page:3/> (дата звертання: 23.10.2023).
13. Можливості та переваги phpmyadmin. URL: <https://ua.waykun.com/articles/mozhливosti-ta-perevagi-phpmyadmin.php> (дата звертання: 23.10.2023).
14. PHP+MySQL. URL: <https://astwellsoft.com/uk/blog/tehnology/php-mysql.html> (дата звертання: 24.10.2023).
15. Реляційні бази даних. URL: <https://www.miyklas.com.ua/p/informatica/10-klas/sistemi-keruvannia-bazami-danikh-326161/skbd-reliatciini-bazi-danikh-326453/re-a33e13f3-aec7-46f1-90d9-6fabf6e20db8> (дата звертання: 24.10.2023).
16. Куликовська Н.А., Тіменко А.В., Ільяшенко М.Б., Шмигля К.В. Використання методів NLP в аналізі тональності текстів з вебресурсів: матеріали XI міжнародної науково-практичної конференції, м.Запоріжжя, 12-14 грудня 2022р. С.85 – 87.
17. Вдовиченко І.Н., Хоцкіна В.Б. Інтелектуальні системи: Навчальний посібник. Державний університет економіки і технологій, м.Кривий Ріг, 2023р. С.37 – 60.
18. Андрєєв, А. С. Особливості створення семантичних мереж. Харківський національний університет радіоелектроніки, м.Харків, 2021р. С.1 – 4. URL: <https://openarchive.nure.ua/items/ab807185-0c52-4448-ad60-8c5221a44d61>.
19. Глушко С.М. Інтелектуальна система кластеризації помилок як складова автоматизації тестування: маг. дис.: 05.09.2019. Київ, 2019р. 125 с.
20. Блощаневич К.О. Інформаційна технологія інтелектуального аналізу даних для автоматизації тестування: маг. дис.: 08.09.2019. Київ, 2019р. 84 с.