Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

Бакалавр
(назва освітнього ступеня)

на тему: «Розробка веб-застосунку для створення і проведення онлайн-опитувань та аналізу результатів з використанням HTML, CSS, Java Script, PHP та MySQL»

Виконав(ла): студент(ка) IV курсу, групи СП – 42
спеціальності 121 – Інженерія програмного забезпечення
(шифр і назва спеціальності)

(підпис) Лунак О.Г. (прізвище та ініціали)

Керівник (підпис) Цуприк Г.Б. (прізвище та ініціали)

Нормоконтроль (підпис) Стоянов Ю.М. (прізвище та ініціали)

Завідувач кафедри (підпис) Петрик М.Р. (прізвище та ініціали)

Рецензент (підпис) (прізвище та ініціали)

Тернопіль – 2025

АНОТАЦІЯ

Кваліфікаційна робота бакалавра. Тернопільський національний технічний університ імені Івана Пулюя, кафедра програмної інженерії, спеціальність 121 «Інженерія програмного забезпечення», ТНТУ, 2025, сторінок – 98, таблиць – 5, рисунків – 49, презентація.

Тема: Розробка веб-застосунку для створення і проведення онлайн-опитувань та аналізу результатів з використанням HTML, CSS, JavaScript, PHP та MySQL.

Кваліфікаційна робота бакалавра присвячена процесу аналізу, проєктуванню та розробки системи для проведення онлайн-опитувань. Систему реалізовано у якості веб-застосунку(веб-сайту), що зумовлено її доступністю для користувачів будь-якої операційної системи, відносною простотою в користуванні та можливістю подальшого масштабування. Здійснено порівняльний аналіз конкурентних сервісів, які надають схожі послуги, враховано їх переваги та недоліки.

У ході роботи було сплановано архітектуру Frontend частини веб-застосунку. Використовуючи мови HTML, CSS з фреймворком Bootstrap та JavaScript з використання AJAX для обміну даними між браузером і сервером – було створено клієнтську частину сайту. Для Backend частини було використано мову PHP для обробки запитів з фронтенду та базу даних MySQL, яка зберігає всю інформацію системи.

Крім того, у роботі детально описано процес тестування онлайн-опитувань та враховано можливості модифікації системи.

Ключові слова: веб-застосунок, клієнтська частина, модифікація, онлайн-опитування, тестування, AJAX, Backend частина, Frontend частина.

ABSTRACT

Bachelor’s qualification work. Ternopil National Technical University named after Ivan Puluj, Department of Software Engineering, specialty 121 «Software Engineering», TNTU, 2025, pages – 98, tables – 5, images – 49, presentation.

Topic: Development of a web application for creating, conducting and analyzing online surveys using HTML, CSS, JavaScript, PHP and MySQL.

The bachelor’s qualification work is devoted to the analysis, design and development of a system for conducting online surveys. The solution is implemented as a web application (website), which ensures cross-platform accessibility, ease of use and the possibility of future scaling. A comparative review of competing services offering similar functionality was carried out, with their advantages and drawbacks taken into account.

The architecture of the Frontend was planned and implemented using HTML and CSS with the Bootstrap framework, while JavaScript and AJAX were employed for data exchange between the browser and the server. The Backend is written in PHP, handling requests from the frontend, and a MySQL database stores all system information.

The work also provides a detailed description of the testing process of the online-survey system and discusses potential avenues for system modification.

Keywords: AJAX, Backend, client side, Frontend, modification, online survey, testing, web application.

ЗМІСТ

ВСТУП

У сучасному інформаційному суспільстві збір, обробка та аналіз даних відіграють ключову роль у прийнятті рішень у бізнесі, освіті, маркетингу, державному управлінні та соціологічних дослідженнях. Онлайн-опитування стали одним із найпоширеніших інструментів, що дає змогу швидко і зручно отримувати зворотній зв’язок від широкого кола респондентів. Проте зі зростанням обсягу опитувальних компаній, різноманітністю типів запитань та потребою у аналітиці результатів виникає потреба у створенні спеціалізованого веб-застосунку для роботи з опитуваннями, який передбачає не просто реалізацію можливості створювати питання й збирати відповіді, а створення єдиного середовища, де весь цикл відбувається максимально зручно. Користувачі повинні мати змогу швидко формувати власні шаблони запитань опитування під свій бренд і надавати доступ до них респондентам. Водночас важливо забезпечити високу продуктивність платформи за збільшеного навантаження з можливістю одночасно обслуговувати сотні чи тисячі користувачів без затримок та збоїв.

Розробка веб-застосунку для створення та проведення онлайн-опитувань і аналізу результатів з використанням технологій HTML, CSS, JavaScript, PHP та MySQL, що дасть змогу побудувати універсальну платформу, що працюватиме у будь-якому сучасному браузері без потреби встановлення додаткового програмного забезпечення. HTML і CSS забезпечують структуровану та приваблиу візуальну частину, JavaScript надасть динамічну взаємодію на клієнтському боці, PHP – серверну логіку, а MySQL надійне зберігання даних.

Процес розробки охоплюватиме кілька етапів, кожен з яких закладає основу для наступного та гарантує послідовну реалізацію вимог. Спершу буде проведено ретельний аналіз існуючих аналогів та їх функціоналу, що допоможе зрозуміти, які підходи для побудови опитувань та їх аналізу є найбільш ефективними, а також виявити недоліки щодо зручності в користуванні такими видами сервісів.

Наступним буде проєктування архітектури системи із створенням набору діаграм, які допоможуть в реалізації прототипу. Діаграма варіантів використання, яка продемонструє типові сценарії взаємодіїї між користувачами та адміністрацією системи. Діаграма класів, що міститиме в собі об’єкти предметної області і зв’язки між ними. Діаграма бази даних, яка схематично відображає основну структуру програми та взаємодію між полями. Останньою буде модель всієї структури веб-застосунку, де буде продемонстровано його роботу.

Розробка серверної логіки включатиме реалізацію інтерфейсів на мові програмува PHP із захистом від поширених векторів атак. Кожен клас відповідатиме за окремий метод – створення нового опитування, редагування наявного або збір відповідей. MySQL служитиме надійним сховищем всіх структурованих даних платформи, збереженням інформації про користувачів, налаштувань опитувань, відповіді респондентів і записаних результатів. Реєстрація й авторизація користувачів, організація прав доступу і зберігання дій передбачатиме використання перевірених безпечних бібліотек.

Паралельно з бекендом реалізується клієнтська частина із застосуванням адаптивного дизайну з використанням HTML-структури й CSS-сітки, що забезпечить коректне відображення на будь-якому екрані. Застосування JavaScript модулів, що завантажуються асинхронно через AJAX-запити, надасть миттєву реакцію інтерфейсу: отримані з сервера JSON-об’єкти негайно перетворюються на списки запитань та форми відповідей, без перезавантаження сторінки.

Заключним кроком стане комплексне тестування готового продукту, а саме демонстрація роботи сайту з наявним функціоналом для виявлення та усунення можливих помилок. Після успішного перевірки платформа буде готова до розгортання на середовищі, а її модульна архітектура забезпечить швидке додавання нових можливостей у майбутньому.

1 ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Вибір напрямку дослідження

У межах початкового етапу розробки проєкту потрібно чітко описати, який саме буде кінцевий продукт, визначити логіку, яка буде об’єднувати ідею застосунку, наявні технічні обмеження та охоплення потенційних користувачів. Від цих відповідей буде залежати усе: від вибору архітектури до інструментів реалізації.

Саме тому етапом дослідження стало визначення, яку роль у житті майбутніх користувачів має виконувати система онлайн-опитувань. Потенційна аудиторія складатиметься з викладачів, учнів та студентів, дослідників громадської думки, невеликих бізнес-команд, яким потрібно проста, але водночас гнучка платформа для швидкого збору даних та їх подальшого опрацювання. Користувачі очікують доступності сервісу, простого інтерфейсу і надійності збереження результатів. Ці фактори наштвохнули до вибору веб-застосунку, тому що браузер на відміну від окремого декстопного чи мобільного застосунку, не потребує додаткового встановлення, миттєво оновлюється і гарантує одинакове використання на різних операційних системах.

Веб-застосунок (або веб-додаток) – це різновид програмного забезпечення, який функціонує через браузер, де клієнтською стороною виступає сам браузер користувача, а серверною – веб-сервер. Основна логіка роботи веб-застосунку зазвичай реалізована на сервері, тоді як завдання браузера полягає у відображені контенту, що завантажується мережею з серверу, та передачі запитів користувача назад на сервер [1].

Веб-застосунки виділяються рядом переваг, що робить їх привабливими для великої кількості розробників та користувачів. Основним є змога отримати доступ до веб-застосунків з будь-якого пристрою та операційної системи підключеного до інтернету. Для їх використання не потрібно встановалювати додаткове програмне

забезпечення, що спрощує процес доступу. Усі зміни та покращення застосунку здійснюються безпосередньо на сервері. Завдяки цьому користувачам не потрібно самостійно встановлювати оновлення, що забезпечує постійний доступ до найактуальнішої версії застосунку. Підтримка та вдосконалення веб-застосунку відбувається бех необхідності розробки окремих версій для різних платформ або пристроїв. Це значно спрощує процес тех. підтримки та дозволяє швидко реагувати на потреби користувачів. Веб-застосунки легко адаптуються до зростання кількості користувачів та обсягу даних без потреб у суттєвих змінах програмного коду чи архітектури системи. Це робить їх придатними як для малих, так і для масштабних проєктів. А завдяки централізованому зберіганню даних на сервері або у хмарному середовищі підвищується рівень безпеки інформації. Використання резервного копіювання дозволяє мінімізувати ризики втрати даних. Також ризик несанкціоноваго вторгнення зменшується, а введеня заходів безпеки стає більш простішим. Не варто забувати, що сучасні веб-застосунки мають можливість легко взаємодіють з іншими веб-сервісами та API, що дозволяє розширювати їхні функціональні можливості та інтегрувати нові сервіси без значних витрат часу та ресурсів [1].

Виділяють декілька типів веб-застосунків, кожен з яких має свої особливості: статичні веб-застосунки; динамічні веб-застосунки; односторінкові програми; веб-портали та системи управління контентом [1].

— Статичні веб-застосунки – це найпростіший тип застосунків, що містить мінімальний обсяг контенту. Такий додаток зазвичай створений за допомогою HTML та CSS і залишається незмінним, доки розробник не оновить код вручну. Серверна логіка у таких проєктах відсутня.

— Динамічні веб-застосунки – забезпечують змінний контент, який оновлюється автоматично при кожному зверненні користувача. Зазвичай даний тип застосунків підключається до баз даних, а керування вмістом здійснюється через панель адміністрування з якої адміністратори можуть виправляти або змінювати вміст програми, включаючи текст та зображення, без необхідності зміни коду.

— Односторінкові програми (SPA, Single Page App) – завантажують весь основний контент лише один раз на початку і оновлюють дані без перевантаження сторінки. Це забезпечує високу швидкість роботи й плавну взаємодію з користувачами.

— Веб-портали – це система з єдиною точкою доступу до різноманітних ресурсів і сервісів. Через головну сторінку користувачі можуть переходити до різних розділів або категорій, що робить навігацію ефективною та зрозумілою.

— Система управління контентом (CMS, Content Management Systems) – дає змогу створювати, редагувати та керувати веб-вмістом без необхідності глибоких знань у програмуванні.

Кожен веб-застосунок побудований на сучасному клієнт-серверному стеку, здатному задовольнити одночасно потреби у простоті створення опитувань і вимоги глибокого аналізу результатів. Клієнтська частина забезпечує інтерфейс, що довзволяє учасникам онлайн-опитувань надсилати запити серверу та відображати отримані результати, які в своє чергу повертає сам сервер. Створюється клієнтська частина за допомогою мов програмування HTML, CSS та JavaScript, а також із використанням бібліотек і фреймоворків, таких як React, Bootstrap, Angular або jQuery [2].

Серверна частина очікує надходження запитів від клієнтів веб-застосунку, а потім повертає правильні відповіді на них. Найкращим варіантом для серверу є простий інтерфейс, який буде зрозумілий клієнтам, щоб вони довго не розбирались і не заглиблювались в особливості самої системи, а саме в програмне забезпечення, яке надає їм послуги. Серверна частина буде реалізована на мові програмування PHP [2].

Комунікація між клієнтською частиною та серверною частиною буде відбуватись за допомогою AJAX. Цей набір технологій дозволить здійснювати запити до сервера без необхідності перезавантаження сторінки. Це суттєво покращить користувацьку взаємодію веб-застосунку, роблячи її більш плавною і швидкою [2].

Для реалізації серверної частини веб-застосунку було вибрано базу даних MySQL. Вона добре оптимізована для роботи з великим обсягами даних. Завдяки підтримці кешування, індексації та розподіленого зберігання MySQL може обробляти великий потік робіт, що критично для онлайн-опитувань із великою кількістю користувачів. Також ця база даних підтримує реплікацію та шардінг, що дозволяє розподілити навантаження між кількома серверами та підвищення продуктивності. Крім того, MySQL забезпечує високу надійність та цілісність даних. Це означає, що всі операції з базою даних виконуються коректно, а в разі збою системи гарантує відновлення даних без втрати цілісності [2].

MySQL також має гнучкі механізми керування доступом, які дозволяють налаштувати рівні прав для користувачів. Адміністратор бази може обмежити доступ до певних таблиць, полів або навіть конкретних запитів, що значно підвищує рівень безпеки та зменшує ризик несанкціонованого доступу. Окрім цього, MySQL підтримує вбудовані механізми шифрування даних як на рівні з’єднання, так і безпосередньо у сховищі. Це дозволяє захистити конфіденційну інформацію, зокрема персональні дані користувачів та результати опитувань, від стороннього втручання.

Загалом завдяки своїй доступності та універсальності веб-застосунки широко використовуються у сферах електронної комерсії, фінансових послуг, освітніх проєктах, соціальних мережах і в багатьох інших сферах діяльності, забезпечуючи гнучкість, зручність використання та менших грошових витрат.

1.2 Аналіз застосунків конкурентів

Сучасний ринок пропонує велику кількість сервісів, що дозволяють проводити онлайн опитування. Умовно їх можна поділити на універсальні платформи, орієнтовані на широке коло користувачів, та професійні дослідницькі

інструменти з розширеними можливостями аналітики та кастомізації. Кожен з них має свої переваги та недоліки, що демонструють їхню потребність використання у конкретних ситуаціях.

Google Forms – це веб-застосунок мережі Google Workspace, призначений для створення онлайн-форм, опитувальників і реєстраційних анкет. Сервіс працює у браузері та не потребує інсталяції, оскільки інтегрований з хмарним сховищем Google Drive. Доступ до форм, а також їх спільне редагування, здійснюється через обліковий запис Google [3].

Рисунок 1.2.1 – Логотип Google Forms

Під час проєктування форми автор може обирати різні типи запитань: одинарний або множинний вибір, випадаючі списки, шкали лінійного рейтингу, сітки, поля для відкритих відповідей тощо. Передбачено функцію логічних розгалужень, що дозволяє будувати адаптивний сценарій проходження опитування залежно від відповідей респондента.

Система автоматично збирає дані в окрему вкладку «Відповіді» та, за потреби, синхронізує їх із Google Sheets для подальшого аналізу. У візуальному інтерфейсі відображаються діаграми розподілу, кількісні підсумки та деталізовані записи кожної форми. Параметри доступу дають змогу налаштувати публічний або приватний режим, обмежити кількість відповідей чи вимагати авторизацію користувачів домену.

Для оформлення доступні готові шаблони та галерея тем з можливістю змінити колірну схему, шрифти й фонове зображення. Опубліковану форму можна вбудувати на веб-сайт за допомогою HTML-фрейму, поширити посиланням або надіслати електронною поштою. Підтримується адаптивне відображення на мобільних пристроях, що забезпечує коректну роботу на смартфонах і планшетах.

SurveyMonkey – це хмарне програмне забезпечення для розробки, розповсюдження й аналізу онлайн-опитувань, орієнтований на бізнес, освітні заклади та дослідницькі організації. Доступ до платформи здійснюється через браузер після реєстрації облікового запису; створені анкети зберігаються у власному акаунті та синхронізуються на всіх пристроях [4].

Рисунок 1.2.2 – Логотип SurveyMonkey

Конструктор форм пропонує великий набір елементів: запитання з одинарним і множинним вибором, ранжування, матриці, сітки зі шкалами Лайкерта, поля для відкритих відповідей, завантаження файлів, а також логічні правила переходів і умовне відображення блоків. Автор може налаштовувати порядок сторінок, випадковизувати варіанти відповіді та додавати підказки чи обов’язковість заповнення.

Зібрані дані відображаються в інтегрованому модулі аналітики, де автоматично будуються діаграми, таблиці та сумарні показники. Є можливість експорту результатів у формати CSV, XLS чи PDF та передавати їх до сторонніх систем через API або інтеграції з інструментами бізнес-аналітики. Користувачі мають доступ до функцій керування правами: можна запрошувати колег до

спільного редагування, обмежувати перегляд або встановлювати ролі для перегляду результатів [4].

Опитування публікується за універсальним посиланням, через вбудований HTML-код, електронну пошту чи соціальні мережі. Доступні налаштування дизайну дають змогу застосовувати кольорові схеми, шрифти й логотипи компанії, щоб узгодити зовнішній вигляд анкети з корпоративним стилем. Сервіс підтримує адаптивну верстку, тож анкети коректно відображаються як на десктопах, так і на мобільних пристроях.

Typeform – це компанія, що надає програмне забезпечення для створення інтерактивних форм і опитувань, побудованих за принципом «запитання-за-запитанням». Сервіс доступний через браузер після авторизації в обліковому записі та зберігає усі форми у хмарному сховищі. Основна концепція полягає в послідовному відображенні кожного елемента опитування на окремому екрані, що імітує діалог і сприяє зосередженню респондента на конкретній відповіді [5].

Рисунок 1.2.3 – Логотип Typeform

Конструктор Typeform підтримує різноманітні типи запитань: текстові поля, варіанти вибору, шкали, оцінки, завантаження файлів, платежі, а також логіку переходів із умовними гілками. Дизайнери можуть додавати зображення, відео та анімації до окремих блоків, налаштовувати шрифти, кольори й вирівнювання, аби сформувати цілісну візуальну айдентику. Зібрані відповіді відображаються в інтегрованій аналітиці або експортуються до CSV, Google Sheets чи сторонніх

сервісів через Webhooks і REST API. Публікувати форму можна за прямим посиланням, через iframe-вбудову на сайт чи автоматичні сервіси розсилок, а адаптивна верстка забезпечує коректне відображення на десктопах і мобільних пристроях.

Microsoft Forms – це хмарний сервіс від Microsoft 365, призначений для створення онлайн-опитувань, вікторин і реєстраційних форм. Працює без інсталяції у будь-якому сучасному браузері, а всі створені форми зберігаються у хмарному сховищі та синхронізуються з обліковим записом Microsoft. Доступ можливий як з комп'ютерів, так і з мобільних пристроїв, а також інтегрується в Microsoft Teams і OneDrive [6].

Рисунок 1.2.4 – Логотип Microsoft Forms

Конструктор пропонує різноманітні типи запитань: одинарний і множинний вибір, текстові поля, рейтингові шкали, дати, ранжування й файл-завантаження. Передбачено логічне розгалуження, що змінює послідовність питань залежно від відповіді респондента, а також таймер і автоматичне оцінювання для тестових робіт.

Зібрані відповіді відображаються у вбудованій аналітиці: діаграми в реальному часі, підсумкові таблиці та можливість переглядати кожне подання окремо. Дані можна експортувати до Excel Online або завантажити у форматі .xlsx для подальшої обробки. Форму можна поширити через URL-посилання, QR-код, кнопку «Поділитися в Teams» чи вбудований HTML-фрейм на веб-сайті.

Інструмент підтримує базове оформлення: вибір кольорової теми, фонових зображень та шрифтів із колекції Microsoft. Крім того, через Power Automate доступно налаштування робочих процесів, наприклад автоматичне надсилання сповіщень або копіювання відповідей у сторонні системи.

1.3 Порівняльний аналіз характеристик аналогів

У попередньому розділі розглянуто основні конкурентні застосунки для мого проєкту: Google Forms, SurveyMonkey, Typeform та Microsoft Forms. Проаналізуємо їх сильні та слабкі сторони, що допоможе знайти підходи, які слід використовувати під час розробки власної платформи і, водночас, рішення, яких слід уникати.

Для початку розглянемо переваги веб-застосунку Google Forms:

— Повністю безкоштовний базовий функціонал, достатній для більшості освітніх і внутрішніх задач.

— Відповіді автоматично надходять у Google Sheets; це дає змогу в реальному часі будувати власні зведені таблиці, графіки та ділитися даними з колегами.

— Простий інтерфейс і готові шаблони дозволяють створити опитування за кілька хвилин без спеціальних знань.

— Підтримка одночасного співавторства та історії редагувань, кілька редакторів можуть паралельно вносити правки, а журнал версій дає змогу відкотитися до попереднього стану, якщо щось пішло не так.

Далі розпишемо недоліки, які присутні в Google Forms:

— Обмежений візуальний кастом: доступні лише базові кольорові схеми та шрифт.

— Брак просунутої логіки: A/B-тестування та розподіл блоків випадковим чином, обмежує складні маркетингові чи наукові дослідження.

— Відсутність захисту паролем: форма може бути відкритою за посиланням або лише для користувачів домену, але задати пароль не вийде без сторонніх рішень.

Отже, Google Forms є оптимальним вибором для безплатних освітніх та внутрішніх опитувань з базовою логікою й швидким стартом, але з мінімальним функціоналом [3].

Наступним сервісом для розгляду переваг та недоліків буде SurveyMonkey [4]. Серед них можна виділити наступні.

Переваги:

— Змога задання правил відсікання учасників після досягнення потрібної кількості відповідей певного типу, налаштування рандомізації варіантів і створення складних сценарії з умовними переходами.

— Розвинена аналітика: інструмент пропонує фільтри, перехресні таблиці, різні види графіків, щоб оцінити ваші результати у контексті ринку.

— 200+ інтеграцій та REST API: є готові конектори для Salesforce, HubSpot, Tableau тощо, а через API можна автоматизувати експорт даних у власні системи BI.

— Персоналізована e-mail-розсилка: надсилання опитування з унікальними токенами дозволяє відслідковувати, хто саме відповів, і нагадувати тим, хто ще не пройшов анкету.

Недоліки:

— Безкоштовний план дає лише прості опитування й обмежену аналітику; професійні можливості коштують відчутно дорожче, що не підходить малим некомерційним проєктам.

— Ліміти відповідей у free-версії:у безкоштовному режимі доступно лише 10 питань і 40 відповідей на опитування, після чого сервіс блокує перегляд нових результатів.

Отже, SurveyMonkey сильний у складних дослідженнях завдяки потужній логіці й звітності, однак дорогий у преміум-сегменті [4].

Наступним конкурентом для нашої системи є TypeForm. Розглянемо його сильні і слабкі сторони [5].

Переваги:

— Респондент бачить одне запитання за раз, що зменшує когнітивне навантаження й підвищує завершуваність опитування.

— Можна додати власні шрифти, фонові GIF чи відео, зробити форму інтерактивною й цілком у корпоративному стилі без коду.

— Вбудовані конектори з Slack, HubSpot, Airtable та Zapier дозволяють моментально передавати відповіді у CRM, таблиці чи автоматичні ланцюжки e-mail.

Недоліки:

— Лише 100 відповідей на місяць і максимум 10 запитань на форму; для більшого обсягу доведеться одразу переходити на платний тариф.

— Якщо опитування містить 50+ питань, користувачеві доведеться багато разів клікати «Next», що може збільшити відтік.

— Висока ціна масштабування: вартість швидко зростає зі збільшенням відвідуваності або необхідності командної роботи, що може стати бар’єром для стартапів.

Отже, Typeform виділяється найкращим користувацьким досвідом і візуальною привабливістю, проте має строгі ліміти у free-плані та підвищену вартість масштабування [5].

І останнім для розгляду є онлайн сервіс Microsoft Forms, серед його переваг можна виокремити наступні:

— Результати можна одночасно виводити в Teams-чат, зберігати в OneDrive та обробляти в онлайн-Excel без експорту-імпорту вручну.

— Побудова тестів із автоперевіркою: створення вікторин, призначення кількості балів і автоматичне надсилання респондентам оцінок відразу після відправлення відповідей.

Серед недоліків даної платформи є:

— Скромні можливості дизайну: усі форми виглядають схоже і дуже просто.

— Необхідність ліцензії Microsoft 365: повний функціонал відкривається лише в середовищі корпоративного пакета.

Отже, Microsoft Forms є зручним всередині екосистеми Microsoft 365 для швидких опитувань і тестів, але поступається у гнучкості дизайну та у потрібності платної підписки [6].

1.4 Формування вимог до власної системи

В сучасному цифровому світі онлайн-опитування значно спрощують процес збору даних завдяки автоматизованим платформам. Це дозволяє швидко створювати, поширювати їх серед цільової аудиторії та аналізувати результати опитувань без потреби у фізичної присутності опитуючих. Однак організація таких опитувань має низку особливостей, які безпосередньо впливають на якість зібраної інформації та правильність отриманих результатів.

Проаналізувавши характеристики аналогів для проведення онлайн-опитувань було обрано необхідну стратегію, що відповідає дослідженню. Визначивши цільову аудиторію, яка буде брати участь в опитуваннях, було сформульовану чіткі, зрозумілі та неупередженні вимоги до власної системи.

Якість отриманих відповідей також багато в чому залежить від правильного структурування питань. По-перше, опитування має містити вступ в якому коротко пояснюється мету опитування та інструкцію щодо заповнення. По-друге, основну частину де буде знаходитись набір питань з різними варіантами відповідей(закритими, відкритими і змішаними). По-третє, завершальний блок, а саме надсилання результатів та отримання результатів. Крім того, важливо, щоб

дизайн опитування був простим, адаптивним під мобільні пристрої та не містив зайвої інформації, яка може відволікати.

Для запобігання неякісних або випадкових відповідей варто використовувати контрольні запитання для перевірки уважності учасників. Також обмежити кількість відповідей з однієї IP-адреси. Аналіз часу, витраченого на проходження опитування, для виявлення спаму і некомпетентних користувачів.

Одним з головних факторів, що впливає на відповіді користувачів, є довіра учасників до безпеки їх даних. Важливо під час проведення онлайн-опитуваня вказати, як будуть використовуватись зібрані дані. У разі потреби слід гарантувати анонімність відповідей. І в особливому порядку дотримуватись законодавчих норм щодо збору персональних даних, особливо з урахуванням актуальних правових та соціальних норм у країні.

Вирішальною і найбільш важливою дією в проведені онлайн-опитувань є аналіз даних після збору відповідей. Початковим кроком є очищення даних від пустих варіантів та спаму це забезпечеть правильність подальшої перевірки з використанням різних статичних методів, та візуалізацію результатів. На основі цього є створення висновків на основі отриманих даних. Дотримання правильного підходу, дозволить отримати коректні та об’єктивні результати опитувань, що в подальшому сприятимуть в прийнятті фінальних рішень у різних галузях.

1.5 Пошук та опис ключових відношень між акторами та прецедентами

Для представлення можливостей веб-застосунку застосовується діаграма варіантів використання (Use Case Diagram), що графічно ілюструє взаємодію між користувачами(акторами) та системою [7, 8].

На момент створення система передбачає у собі 4 актори:

Респондент:

— має змогу реєструватись на сайті, входити та виходити з системи;

— налаштовує власний профіль зі змогою зміни паролю;

— проходить опитування заповнюючи бланки відповідей;

— змінює варіанти обраних відповідей;

— здійснює пошук опитувань;

— має змогу поширювати опитування до яких має доступ.

Створювач опитувань:

— доступні усі і ті ж функції, що і респонденту;

— створює опитування з можливістю його зміни;

— створює питання до кожного опитування;

— створює варіанти відповідей для кожного з питань.

Аналітик опитувань:

— доступні усі і ті ж функції, що і респонденту;

— має змогу переглядати варіанти відповідей до опитувань;

— підраховує та додає результати опитування.

Адміністратор:

— доступні усі і ті ж самі функції, що і респонденту, створювачу та аналітику опитувань;

— вхід в панель адміністрування всього сайту;

— створює, редагує та видаляє користувачів системи;

— додає, редагує та видає опитування від усіх користувачів;

— редагує, видаляє та додаває питання з самого опитування від усіх користувачів;

— додає, редагує та видаляє варіантів відповідей з опитування від усіх користувачів;

— переглядає та змінює результати опитувань.

Було знайдено та описано зв’язки між акторами, респондентом, створювачем та аналітиком опитувань, та адміністратором з їхніми прецедентами. Тепер

зобразимо ці дані у вигляді діаграми варіантів використання системи зображену на рисунку 1.5.1.

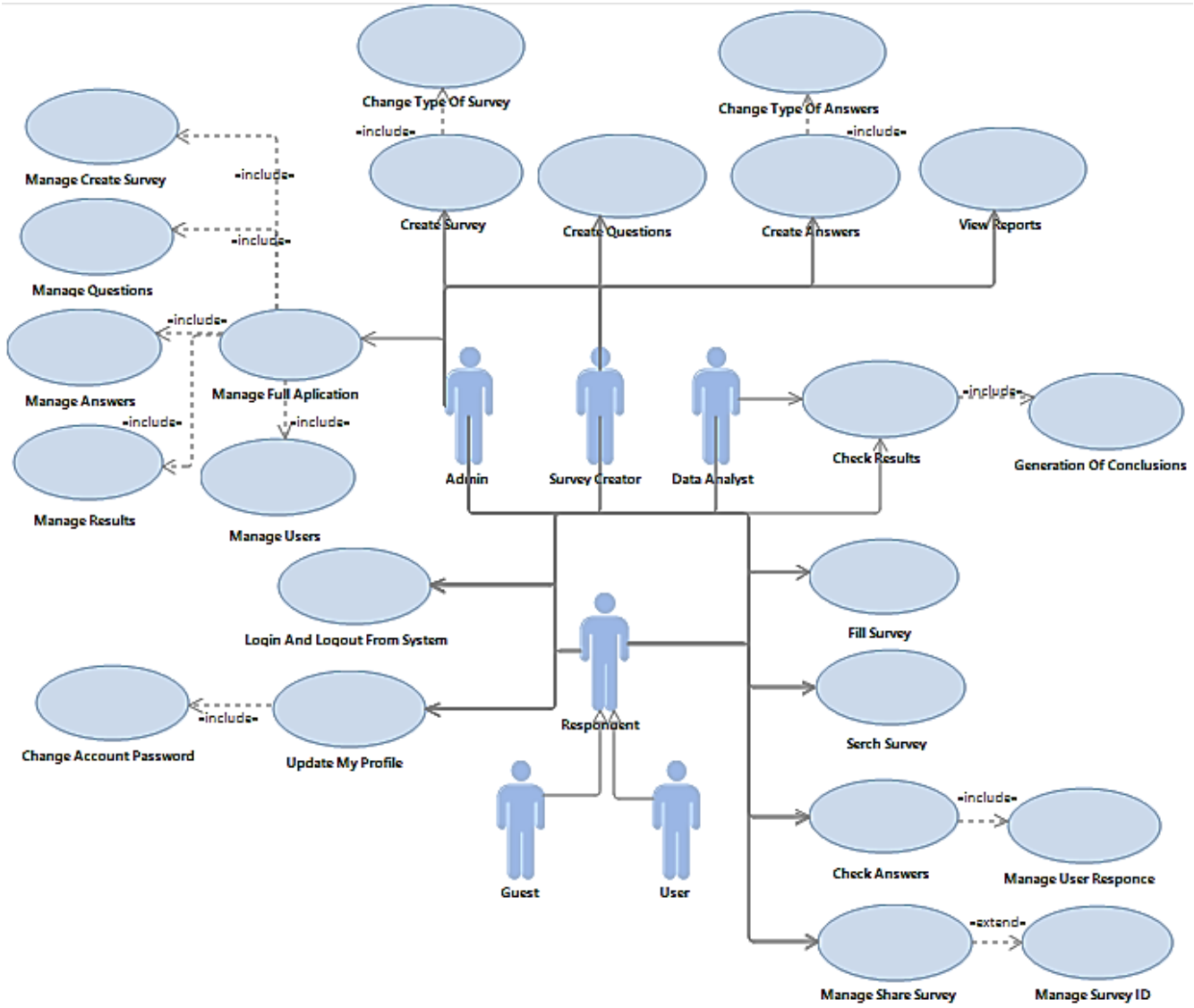


Рисунок 1.5.1 – Діаграма варіантів використання(use case diagram)

2 ПРОЄКТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ

2.1 Визначення архітектури системи

Проєктування веб-застосунку для створення та проведення онлайн-опитувань потребує чіткого опису структури та функціоналу.

Структурне моделювання базується на клієнт-серверній архітектурі, яка забезпечує ефективну взаємодію між користувачем, сервером та базою даних. Через це обробка даних розподіляється між клієнтською та серверною частинами, оптимізуючи продуктивні та забезпечуючи безпеку даних. Розглянемо та розпишемо основні компоненти системи [9].

Користувачі, для яких у системі передбачено категорії:

— адміністраця системи – здійснюють адміністрування сайтом та змінюють налаштування доступу;

— створювачі опитувань – мають можливості створювати, редагувати та керувати опитуваннями;

— аналітики даних – аналізує відповіді користувачів на питання опитувань, створює їх результати та надсилає їх у профіль респондента.

— учасники системи – проходять опитування, надсилаючи відповіді у визначеному форматі, переглядати результати та їх графіки аналізу. Залежно від налаштувань опитування, вони можуть бути анонімні та авторизовані.

Кожен користувач взаємодія із системою через веб-інтерфейс, надсилаючи та отримуючи інформацію за допомогою HTTP-запитів. Тому розглянемо наступний компонент системи, а саме веб-інтерфейс.

Веб-інтерфейс – набір веб-сторінок, які забезпечують зручну роботу користувача із системою [9].

Він включає такі пункти:

— головна сторінка – відображає список доступних опитувань, які доступні для користувачів та віджети інших веб-сторінок;

— сторінка авторизації та реєстрації – використовується для входу в веб-застосунок, визначення прав користувача та для коректного проходження опитувань з отриманням результатів;

— сторінка створення опитувань – дозволяє адміністраторам додавати нові опитування, визначати типи питань;

— сторінка проходження опитування – забезпечує зручне введення відповідей учасників опитувань;

— сторінка користувача – дозволяє респондентам отримувати дані про опитування та їх результати, в свою чергу адміністраторам ж переглядати відповіді та формувати аналіз відповідей для надання доступу до них учасникам опитувань.

Серверна частина теж вважається компонентом системи, тому що вона відповідає за обробку запитів від клієнтів та взаємодію з базою даних. Обробляє HTTP-запити користувачів такі як: надсилання відповідей, отримання списку опитувань та учасників. Звідки далі відбувається генерація аналітичних звітів та статистики на основі отриманих відповідей. Також автентифікація, авторизація та обробка даних самих користувачів перед збереженням в базі даних проходить через серверну частину [9].

Тепер, розглянувши більшість компонентів системи, опишемо центральну та найголовнішу серед них – базу даних. Вона забезпечує збереження та доступ до інформації, та містить такі основні таблиці:

— користувачі – зберігає дані про адміністраторів та учасників (логін, пароль, електронна пошта, рівень доступу);

— опитування – містить інформацію про створенні опитування (назва, опис, автор, дата створення);

— питання – зберігає питання, пов’язані з опитуваннями;

— відповіді – містить відповіді учасників на конкретні запитання;

— результати – оброблені та згенеровані дані для статичного аналізу.

Для системи веб-застосунку було обрано базу даних MySQL через її просту взаємодію з мовами програмування, які використовуються в розробці проєкту, та її ефективність працювати з великими обсягами даних.

Загалом описавши основні компоненти системи, також важливо розібрати їх взаємодію, а саме між клієнтською та серверною частинами. Для початку, користувач заходить на веб-застосунок та відправляє HTTP-запит на сервер. Сервер обробляє запит, звертається до бази даних і повертає відповідь клієнту. Далі клієнтська частина отримує відповідь та відображає її у вигляді веб-інтерфейсу. Якщо користувач проходить опитування, відповіді надсилаються на сервер, обробляються та зберігаються у базі даних. Адміністратор чи аналітик даних переглядає результати через веб-інтерфейс, отримує статичні дані з бази та формує по ним аналіз опитування [2, 9].

Таким чином, структурне моделювання веб-застосунку визначає логічні зв’язки між його основними складовими та здійснює ефективне управління процесами системи.

2.2 Проєктування моделі баз даних

База даних системи опитувань складається з таких основних таблиць: Users, Surveys, Questions, Answers, Surveys_with_Users та зв’язків між полями цих таблиць [10].

Позначення PK (Primary Key) означає «первинний ключ» – стовпець, який унікально ідентифікує рядок таблиці [10].

FK (Foreign Key) означає «зовнішній ключ» – стовпець, що посилається на первинний ключ іншої або тієї ж таблиці для забезпечення цілісності даних [10].

Спроєктована модель бази даних має наступний вигляд (рисунок 2.2.1).

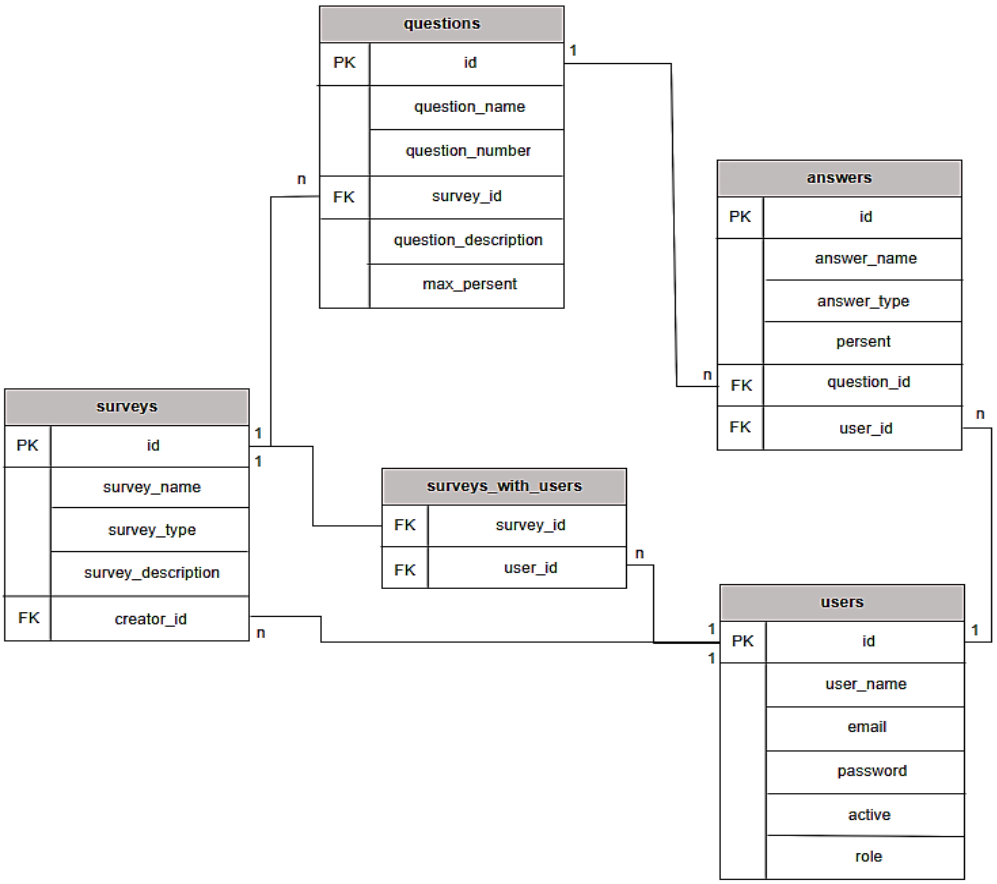


Рисунок 2.2.1– Модель основних таблиць бази даних

Структура таблиці «Users» представлена наступним чином (таблиця 2.2.1).

Таблиця 2.2.1 – Структура таблиці «Users»

Ім’я поля	**Тип і розмір поля**	**Опис поля**
id	int	індентифікатор
user_name	varchar(32)	ім’я
email	varchar(64)	електронна пошта
password	varchar(255)	пароль
active	varchar(255)	тип користувача
role	varchar(32)	роль користувача

В даній таблиці розміщуються поля, як ідентифікатор користувача, його ім’я, електронна пошта та пароль, також активність користувача чи є він

онлайн/офлайн, та ролі, які мають такі варіанти: «Респондент», «Створювач опитувань», «Аналітик даних» та «Адміністратор».

Структура таблиці «Surveys» представлена в таблиці (таблиця 2.2.2).

Таблиця 2.2.2 – Структура таблиці «Surveys»

Ім’я поля	**Тип і розмір поля**	**Опис поля**
id	int	ідентифікатор
survey_name	varchar(255)	назва опитування
survey_type	varchar(255)	тип опитування
survey_description	varchar(512)	опис опитування
creator_id	int	ідентифікатор створювача

Таблиця «Surveys» складається з полів індетифікатор, назва опитування, опис опитування та індентифікатор створювача опитувань, що містить зв’язок з таблицею «Users», так як виражає інформацію про користувача, що створив опитування.

Структура таблиці «Surveys_with_Users» представлено нижче (таблиця 2.2.3).

Таблиця 2.2.3 – Структура таблиці «Surveys_with_Users»

Ім’я поля	**Тип і розмір поля**	**Опис поля**
survey_id	int	ідентифікатор опитування
user_id	int	ідентифікатор користувача

Таблиця «Surveys_with_Users» складається з полів ідентифікатор опитувань та індентифікатор користувачів. Дана таблиця реалізовує зв’язок між таблицями «Users» та «Surveys». Цей зв’язок вказує на те, що один респондент може проходити багато опитувань, а одне опитування може містити багато респондентів. Також це допоможе з приватними видами опитувань, які будуть поширюватись за

посиланнями. Тому була необхідна була таблиця в базі даних «Surveys_with_Users» для реалізації цього зв’язку.

Структура таблиці «Questions» представлена в таблиці (таблиця 2.2.4).

Таблиця 2.2.4 – Структура таблиці «Questions»

Ім’я поля	**Тип і розмір поля**	**Опис поля**
id	int	індентифікатор
question_name	varchar(255)	назва завдання
description	varchar(512)	опис завдання
survey_id	int	ідентифікатор опитування
answer_name	varchar(255)	варіанти відповідей
max_persent	double	максимальна оцінка

Таблиця «Questions» складається з полів ідентифікатор, назва завдання, опис завдання, варіанти відповідей, ідентифікатор опитування та максимальна збіжність відповідей, по якій буде проводитись аналітика опитувань. Поле ідентифікатору опитувань мають зв’язок з таблицею «Questions», так як виражає до якого саме опитування відносяться завдання.

Структура таблиці «Answers» наведено в таблиці (таблиця 2.2.5).

Таблиця 2.2.5 – Структура таблиці «Answers»

Ім’я поля	**Тип і розмір поля**	**Опис поля**
id	int	ідентифікатор
answer_name	varchar(255)	назва відповіді
answer_type	varchar(512)	тип відповіді
persent	double	збіжність відповідей
question_id	int	ідентифікатор питання
user_id	int	ідентифікатор користувача

Таблиця «Answers» містить в собі поля ідентифікатор, назва відповіді, тип відповіді, збіжність відповідей в відсотках, ідентифікатор питання та індентифікатор респондента. Таблиця об’єднує зв’язок між «Questions» та «Users», тому що завдання як і опитування, може вирішуватись багатьма респондентами.

2.3 Побудова UML-діаграми класів системи

Проєктування головних класів відображає статичну структуру системи, тобто сукупність об’єктів предметної області, їхні атрибути, операції та взаємозв’язки. А побудова суцільної діаграми класів є критичним етапом перед реалізацією коду, оскільки саме в ній на основі варіантів використання виділяються класи та їхня роль у системі, також для кожного класу задаються ключові поля, операції та встановлюються зв’язки [11].

Одним з найголовніших елементів системи буде клас «Roots», який відповідає за керування правами доступу у системі (рисунок 2.3.1).

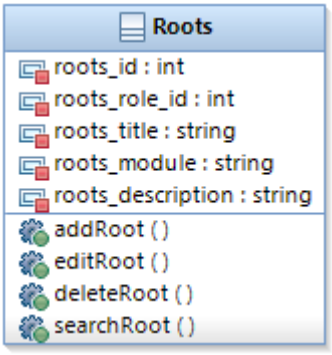


Рисунок 2.3.1 – Клас «Roots»

Він зберігає інформацію про те, яка роль яким функціональним модулем може користуватися. В даному класі будуть наступні поля:

— root_id – поле типу IntegerField; первинний ключ, унікально ідентифікує кожен запис про право доступу.

— root_role_id – поле типу ForeignKey (багато-до-одного) на модель Role; вказує, якій ролі в системі належить це право.

— root_title – поле типу CharField; зберігає коротку назву дозволу, наприклад «Edit Survey» чи «View Reports».

— root_module – поле типу CharField; містить технічну назву або код модуля/функційної області (наприклад survey, analytics), до якої належить це право.

— root_description – поле типу TextField; детальний опис того, що саме дає це право, щоб адміністратори швидко розуміли його призначення.

А також такі методи:

— addRoot() – створює новий запис про право доступу та зберігає його в базі даних.

— editRoot() – змінює атрибути наявного дозволу (наприклад, назву чи опис) і фіксує зміни.

— deleteRoot() – видаляє право доступу; зазвичай викликається лише адміністраторами.

— searchRoot() – повертає перелік дозволів, що відповідають заданим критеріям пошуку, полегшуючи навігацію у великій системі прав.

Іншим не менш важливим класом є «User», який буде являти собою зареєстрованого користувача системи онлайн-опитувань (рисунок 2.3.2).

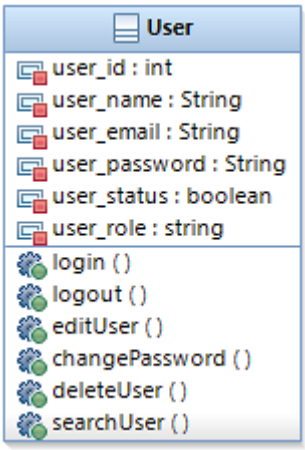


Рисунок 2.3.2 – Клас User

Даний клас буде мати наступні поля:

— user_id – поле типу IntegerField; первинний ключ, який унікально ідентифікує кожного користувача.

— user_name – поле типу String; зберігає унікальне відображуване ім’я або логін користувача, що використовується для входу в систему чи відображення в інтерфейсі.

— user_email – поле типу String; містить електронну адресу користувача, яка слугує контактним каналом і може використовуватися для відновлення пароля та сповіщень.

— user_password – поле типу String; хешований пароль користувача .

— user_status – поле типу BooleanField; позначає чи активовано обліковий запис.

— user_role – поле типу string; назва ролі, до якої належить користувач. Використовується для перевірки прав доступу.

Методи:

— login() – автентифікує користувача за введеними обліковими даними; встановлює сесію або видає токен доступу.

— logout() – завершує поточну сесію користувача, анулюючи токен або cookie та очищаючи тимчасові дані.

— editUser() – дозволяє користувачеві змінити власні дані профілю (ім’я, email, аватар тощо) та зберегти їх у базі даних.

— changePassword() – реалізує зміну пароля: перевіряє старий пароль, хешує новий і оновлює значення поля password.

Наступний клас досить важливий в системі, так як він визначає ролі користувача і надає йому доступ до системи (рисунок 2.3.3).

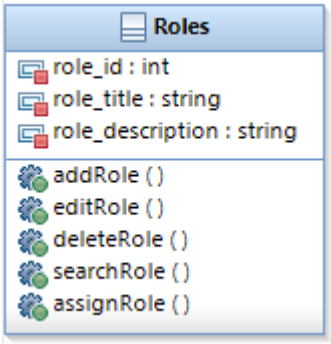


Рисунок 2.3.3 – «Клас Roles»

Клас «Roles» буде містити в собі наступні поля:

— role_id – поле типу IntegerField. Є первинним ключем, який унікально ідентифікує кожну роль у базі даних.

— role_title – поле типу StringField. Зберігає коротку та унікальну назву ролі.

— role_description – поле типу . Містить докладний опис прав і призначення ролі, щоб адміністратор швидко розумів її функцію.

Основні методи класу:

— addRole() – створює нову роль, перевіряє унікальність назви, додає запис у БД.

— editRole() – дозволяє змінити назву чи опис наявної ролі та зберегти зміни.

— deleteRole() – видаляє роль; перед видаленням зазвичай перевіряє, чи не призначено її активним користувачам.

— searchRole() – виконує пошук ролей за заданими критеріями.

— assignRole() – прив’язує роль до конкретного користувача, оновлюючи відповідне поле role_title або зовнішній ключ.

Основний клас «Survey» представляє опитування у системі; містить загальні відомості про нього та забезпечує ключові операції життєвого циклу створення, публікація, закриття, отримання завдань і відповідей (рисунок 2.3.4).

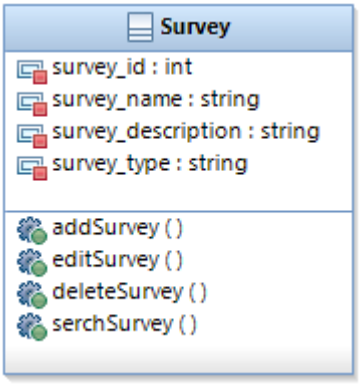


Рисунок 2.3.4 – Клас «Survey»

Даний клас буде мати наступні поля:

— survey_id – IntegerField. Первинний ключ; унікально ідентифікує кожне опитування.

— survey_name – CharField. Коротка назва опитування.

— survey_description – TextField. Розгорнутий опис мети та змісту опитування.

— survey_type – CharField / Enum. Відображає категорію опитування анонімне, обов'язкове, внутрішнє, зовнішнє, що допомагає фільтрувати та керувати опитуваннями.

Методи класу:

— addSurvey() – створює новий запис опитування: перевіряє коректність полів, записує у БД та повертає ID щойно створеного об'єкта.

— editSurvey() – дозволяє змінити назву, опис або тип існуючого опитування та зберегти зміни.

— deleteSurvey() – видаляє опитування; перед видаленням зазвичай перевіряє, чи немає вже зібраних відповідей, щоб уникнути втрати даних.

— searchSurvey() – виконує пошук опитувань за заданими критеріями ключові слова в назві/описі та повертає перелік відповідників.

Окреме завдання всередині опитування демонструє клас «Questions» та містить усю потрібну інформацію для відображення й оцінювання (рисунок 2.3.5).

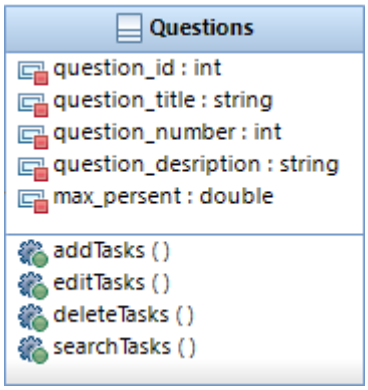


Рисунок 2.3.5 – Клас «Questions»

Поля:

— question _id – IntegerField. Первинний ключ, що унікально ідентифікує кожне питання.

— question _title – CharField. Коротка назва або заголовок питання.

— question _number – IntegerField. Порядковий номер у межах поточного опитування; гарантує правильну послідовність відображення.

— question_description – TextField. Повний текст питання або інструкції до нього.

— max_persent – DoubleField. Максимальний бал/відсоток, який може дати це питання у загальному підсумку.

Методи:

— addQuestions () – створює новий запис завдання, додає його до відповідного опитування та повертає ID.

— editQuestions () – дозволяє змінити назву, порядковий номер, тип, опис чи максимальний бал, після чого зберігає зміни у БД.

— deleteQuestions () – видаляє завдання; зазвичай перевіряє, чи немає вже відповідей, аби запобігти втраті даних.

— searchQuestions() – виконує пошук завдань за критеріями (ключові слова в назві/описі, тип, номер) і повертає перелік відповідників.

Клас «Answers» зберігає конкретні відповіді респондентів на окремі завдання опитування та дозволяє виконувати базові операції керування цими даними (рисунок 2.3.6).

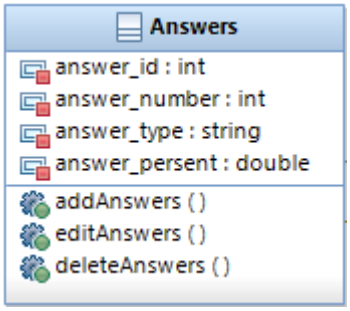


Рисунок 2.3.6 – Клас «Answers»

Він містить такі поля та методи:

— answer_id – IntegerField. Первинний ключ, унікально ідентифікує кожну відповідь.

— answer_number – IntegerField. Порядковий номер варіанта відповіді для конкретного завдання.

— answer _type – CharField/Enum. Вказує тип відповіді одиничний вибір, множинний вибір або відкрита відповідь тощо.

— answer_persent – DoubleField. Числовий бал або відсотковий вклад цієї відповіді у підсумкову оцінку 0 – 100 %.

Методи:

— addAnswers() – створює новий запис відповіді та зберігає його у базі даних.

— editAnswers() – дозволяє змінити номер або вагу/відсоток уже наявної відповіді.

— deleteAnswers() – видаляє відповідь; зазвичай перевіряє, чи не використано її у підрахунку результатів.

Також було створено суцільну діаграму класів, де подано класи системи та зображено зв’язки між ними. Повна діаграма класів зображено на рисунку 2.3.7.

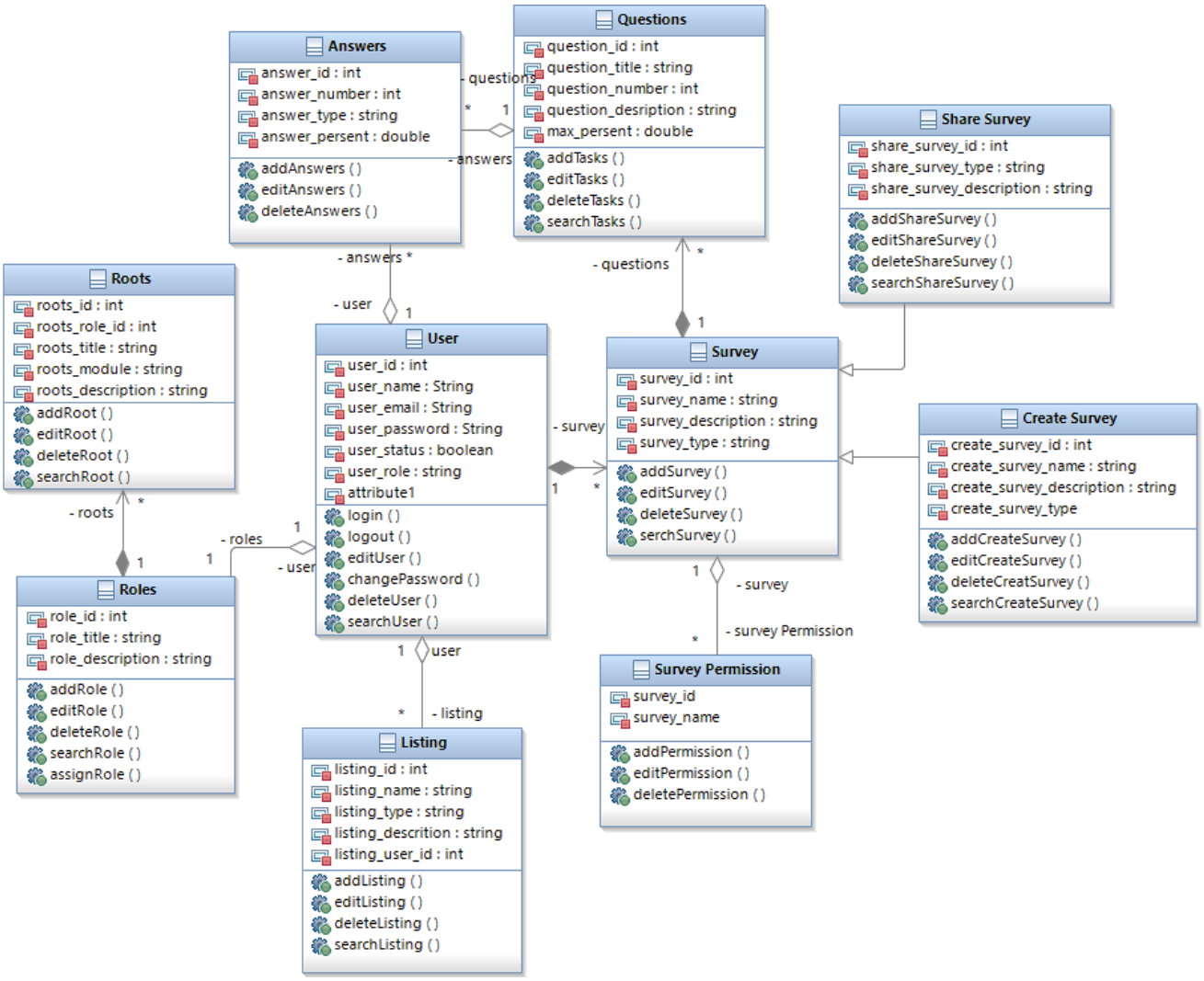


Рисунок 2.3.7 – UML-діаграми класів

2.4 Проєктування структури веб-застосунку

Згідно з потребами користувачів структура веб-застосунку для створення та проведення онлайн-опитувань та їх аналізу складається з головних сторінок (рисунок 2.4.1 та додаток Б), які мають різну функціональність для користувачів з різними ролями (вищого або нищого статусу) [12].

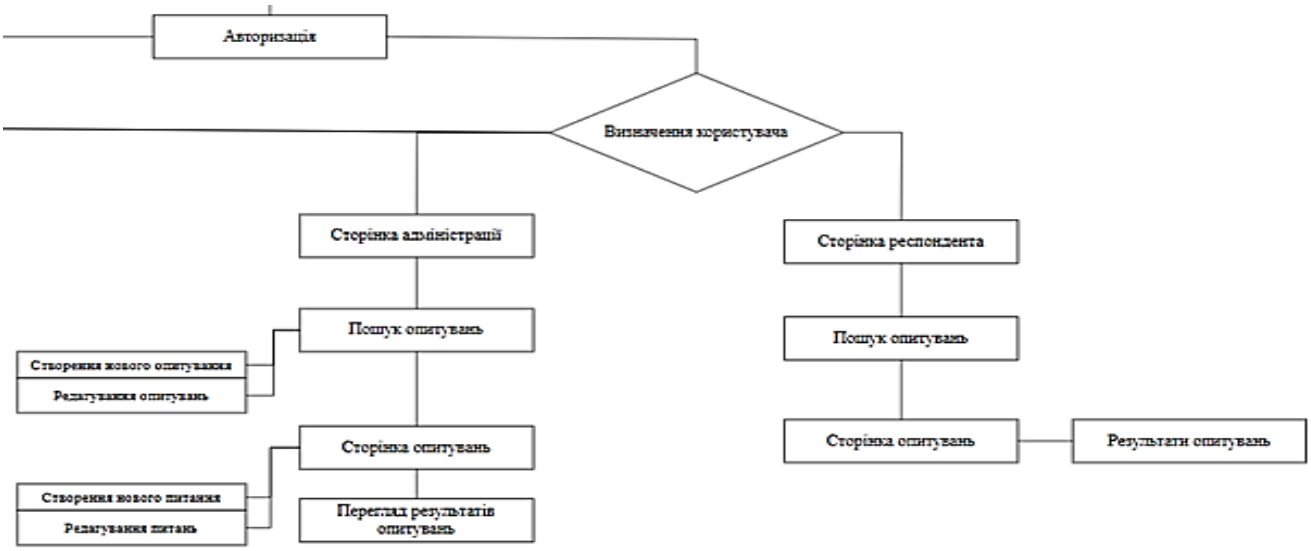


Рисунок 2.4.1 – Структура веб-застосунку

Початок роботи веб-застосунку стартує з головної сторінки сайту і етапу визначення пристрою з якого користувач входить у систему за допомогою фреймворку Bootstrap, що аналізує ширину дисплею пристрою і підлаштовується під його розмір.

Перша сторінка – Авторизація (Вхід). Сторінка авторизації дозволяє увійти в особостий кабінет облікового запису системи опитувань шляхом вводу особистих даних, а саме за допомогою логіну або електронної пошти та паролю. У разі якщо користувач не зареєстрований у системі, то у нього є можливість перейти на сторінку реєстрації.

Друга сторінка – Реєстрація, де користувач зможе заповнити потрібні поля для створення облікового запису. Дана сторінка, як і сторінка авторизації містить перевірку даних, що вводить користувач. Після того, як користувач буде зареєстрований його перенаправить на сторінку авторизації для входу у систему. Далі користувач вводить коректні дані для входу в систему і визначається, якою роллю володіє користувач. Виходячи із значення «роль» користувачу буде надано права на користування сторінкою. Якщо користувач володіє ролями «Адміністратор», «Створювач опитувань» або «Аналітик даних», то він зможе керувати та адмініструвати дані веб-застосунку, коли ж «Респондент» він лише буде мати змогу на пересування по сторінках та проходження опитувань.

Третя сторінка – Мій Профіль (адміністрації або ж респондента). Профіль користувача залежить від ролі, яку надано користувачу при реєстрації облікового запису. Якщо користувач з роллю «Адміністратор», то на його особистій сторінці він має змогу керувати новими та старими видами опитувань, та їх даними. Якщо користувач є «Створювачем опитувань», то він має право на створення нових видів опитувань, та перегляду опитувань, що вже створені ним попередньо. Якщо ж користувач є «Аналітиком даних», то він має доступ до результатів опитувань та створює підсумки результатів. Якщо роль буде «Респондент», то для нього є можливість переглянути опитування та знайти потрібне опитування за допомогою сторінки «Пошук Опитувань», яка знайде збіги з наявними опитуваннями та відобразить їх.

Четверта сторінка – Опитування. Сторінка опитувань не сильно відразняється для користувача з ролями «Респондент» та «Адміністратор» чи інших. В усіх випадках користувачі бачать перелік опитувань, які можна переглянути натиснувши на на назву опитування та можливістю перегляду результатів.

П’ята сторінка – Створення Опитувань. На даній сторінці для «Адміністратора» та «Створювача опитувань» відкрита можливість створювати нові опитування з різними видами вибору відповідей. Відповідні дії не будуть доступні для простих «Респондентів», вони будуть бачити дану сторінку, яка буде працювати для них, але без відповідних прав, вони не зможуть створювати опитування, що забезпечить чистоту сайту, без різних дивних опитувань.

Шоста сторінка – Пошук Опитувань. Конкретна сторінка доступна для всіх ролей у веб-застосунку. Вона дозволяє користувачам знаходити різні опитування за ключовими словами та дає змогу проходити їх, відповідаючи на питання та в кінці отримуючи результати опитувань.

Сьома сторінка – Аналіз Опитувань. Сторінка результатів доступна лише для ролей «Адміністратора» з «Аналітиком даних», на ній будуть доступні відповіді респондентів до всіх опитувань.

3 РОЗРОБКА ТА ТЕСТУВАННЯ ВЕБ-ЗАСТОСУНКУ ДЛЯ ПРОВЕДЕННЯ ОНЛАЙН-ОПИТУВАНЬ

3.1 Розробка веб-застосунку

У цьому підрозділі описано етапи розробки серверної та користувацької частин веб-застосунку для створення, проведення та аналізу онлайн-опитувань. Основною метою було створити функціональну та розширену систему, яка забезпечує зручний інтерфейс для користувачів різних ролей: респондент, створювач опитувань, аналітик даних, адміністратор та підтримує базові операції з опитуваннями, питаннями й відповідями [2].

3.1.1 Розробка серверної частини

Серверна частина реалізована на чистому PHP з використанням Composer, MVC-підходу та гнучкої модульної архітектури. Сам ж застосунок побудовано на основі архітектурного підходу MVC (Model-View-Controller), що забезпечує розділення логіки між обробкою запитів, бізнес-логікою та роботою з базою даних для полегшення роботи з опитуваннями, користувачами, завданнями та відповідями (рисунок 3.1.1.1).

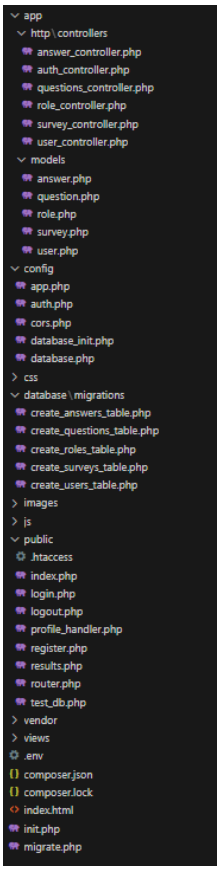


Рисунок 3.1.1.1 – Серверна структура проєкту

У структурі проєкту реалізовано наступні компоненти:

Моделі (`app/models/`) – описують сутності, відповідні до таблиць бази даних (User, Survey, Questions, Answer, Role) та їх поля.

Контролери (`app/http/controllers/`) – реалізують логіку обробки дій над сутностями, зокрема створення, отримання, видалення та редагування опитувань і відповідей.

Конфігураційні файли (`config/`) – містять налаштування підключення до бази даних (`database.php`), змін середовища (`app.php`), політики безпеки (`cors.php`) та ролей авторизації (`auth.php`).

Міграції (`database/migrations/`) – SQL-структура у вигляді PHP-скриптів, що створюють таблиці бази даних.

У проєкті використано СУБД MySQL, підключення до якої здійснюється через об’єкт `mysqli`[13]. Параметри підключення винесені у `.env` файл для зручності зміни конфігурації без редагування коду (рисунок 3.1.1.2).

```
APP_NAME="Survey Application"
APP_ENV=local
APP_DEBUG=true
APP_URL=http://localhost:8080/main

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=8808
DB_DATABASE=surveyhub
DB_USERNAME=root
DB_PASSWORD=123456
```

Рисунок 3.1.1.2 – Лістинг файлу `.env`

Зчитування `.env` реалізовано через бібліотеку `vlucas/phpdotenv`, яка підключається через інструмент управління залежностями в проєктах PHP – Composer (рисунок 3.1.1.3).

```
PS D:\ТНТУ\program> git --version
git version 2.49.0.windows.1
PS D:\ТНТУ\program> composer install
Installing dependencies from lock file (including require-dev)
Verifying lock file contents can be installed on current platform.
Package operations: 6 installs, 0 updates, 0 removals
  - Downloading symfony/polyfill-php80 (v1.32.0)
  - Downloading symfony/polyfill-mbstring (v1.32.0)
  - Downloading symfony/polyfill-ctype (v1.32.0)
  - Downloading phpoption/phpoption (1.9.3)
  - Downloading graham-campbell/result-type (v1.1.3)
  - Downloading vlucas/phpdotenv (v5.6.2)
  - Installing symfony/polyfill-php80 (v1.32.0): Extracting archive
  - Installing symfony/polyfill-mbstring (v1.32.0): Extracting archive
  - Installing symfony/polyfill-ctype (v1.32.0): Extracting archive
  - Installing phpoption/phpoption (1.9.3): Extracting archive
  - Installing graham-campbell/result-type (v1.1.3): Extracting archive
  - Installing vlucas/phpdotenv (v5.6.2): Extracting archive
Generating autoload files
6 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
```

Рисунок 3.1.1.3 – Встановлення Composer

Механізм конфігурації дозволяє централізовано керувати середовищем (назва застосунку, режим роботи, URL тощо).

Створення структури бази даних винесено в окремі PHP файли-міграції, розміщені в директорії `database/migrations/`. Це дає змогу швидко розгорнути проєкт на будь-якому сервері [14]. Усі таблиці мають зовнішні ключі, які зберігають цілісність даних. Для запуску використовується файл `init.php` або `migrate.php` (рисунок 3.1.1.4 і 3.1.1.5).

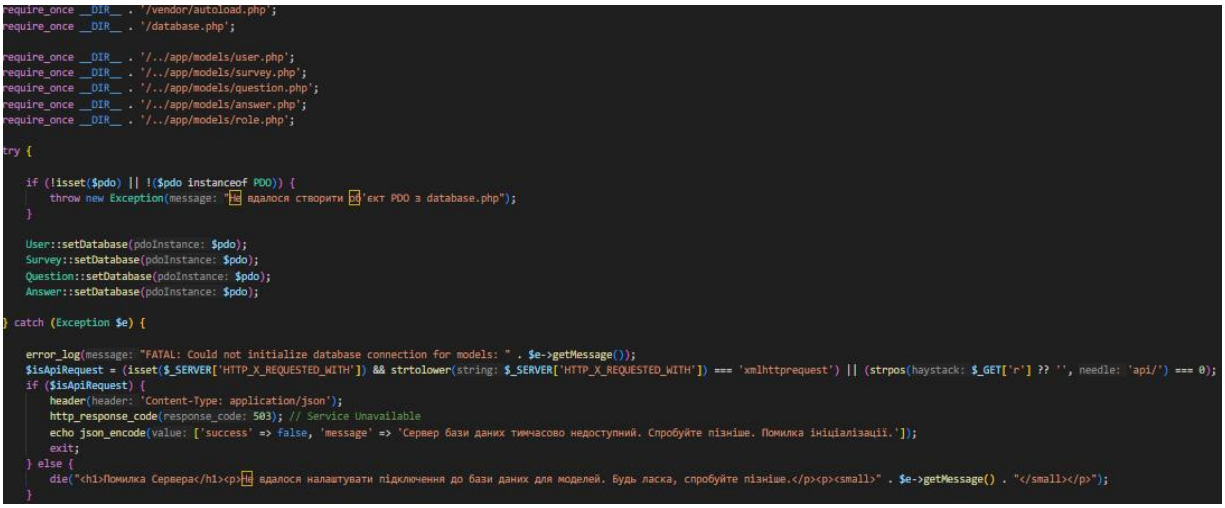


Рисунок 3.1.1.4 – Частковий лістинг коду `init.php`

```php
require_once 'vendor/autoload.php';

try {
    $dotenv = Dotenv\Dotenv::createImmutable(paths: __DIR__);
    $dotenv->load();
} catch (Dotenv\Exception\InvalidPathException $e) {
    error_log(message: "Could not load .env file for migrations: " . $e->getMessage());
    die("Configuration error for migrations: .env file not found or path is incorrect. Make sure .env is in the project root.");
}

$host = $_ENV['DB_HOST'] ?? 'localhost';
$username = $_ENV['DB_USERNAME'] ?? 'root';
$password = $_ENV['DB_PASSWORD'] ?? '';
$database = $_ENV['DB_DATABASE'] ?? null;
$port = $_ENV['DB_PORT'] ?? 3306;

if (!$database) {
    die("DB_DATABASE not set in .env file. Cannot run migrations.");
}
```

Рисунок 3.1.1.5 – Частковий лістинг коду `migrate.php`

Також розглянемо найважливіші функцій на сайті, які забезпечують пересування сайтом та надають доступ до функціоналу. У файлі `register.php` та `login.php` реалізовано логіку реєстрації та авторизації користувачів. Паролі хешуються функцією `password_hash()`, валідація проводиться через перевірку відповідності email та ролі (рисунок 3.1.1.6 та 3.1.1.7).

```php
$name = $_POST['name'];
$email = $_POST['email'];
$password = password_hash(password: $_POST['password'], algo: PASSWORD_DEFAULT);

// За замовчуванням роль – user
$role = 'user';

try {
    $sql = "INSERT INTO users (name, email, password, role) VALUES (:name, :email, :password, :role)";
    $stmt = $pdo->prepare(query: $sql);
    $stmt->execute(params: [
        ':name' => $name,
        ':email' => $email,
        ':password' => $password,
        ':role' => $role
    ]);

    // Після успішної реєстрації – перенаправити на сторінку входу
    header(header: "Location: ../login.html");
    exit;

} catch (PDOException $e) {
    if ($e->getCode() == 23000) {
        echo " Користувач з таким email вже існує.";
    } else {
        echo "Помилка: " . $e->getMessage();
    }
}
```

Рисунок 3.1.1.6 – Реалізація логіки реєстрації

```php
$sql = "SELECT * FROM users WHERE name = :name LIMIT 1";
$stmt = $pdo->prepare(query: $sql);
$stmt->execute(params: [':name' => $name]);
$user = $stmt->fetch();

if ($user && password_verify(password: $password, hash: $user['password'])) {

    $_SESSION['user_id'] = $user['id'];
    $_SESSION['user_name'] = $user['name'];
    $_SESSION['user_role'] = $user['role'];


    switch ($user['role']) {
        case 'admin':
            header(header: "Location: ../profile_admin.html");
            break;
        case 'survey_creator':
            header(header: "Location: ../profile_survey_creator.html");
            break;
        case 'data_analyst':
            header(header: "Location: ../profile_data_analyst.html");
            break;
        default:
            header(header: "Location: ../profile.html");
    }
    exit;
} else {
    echo "Невірний логін або пароль.";
}
```

Рисунок 3.1.1.7 – Реалізація логіки авторизації

У `config/auth.php` описані всі дозволені ролі (рисунок 3.1.1.8).

```php
return [
    'table' => 'users',

    'login_field' => 'email',
    'password_field' => 'password',
    'role_field' => 'role',

    'roles' => [
        'admin',
        'user',
        'data_analyst',
        'survey_creator',
    ],
];
```

Рисунок 3.1.1.8 – Ролі веб-застосунку

Для реалізації обробки HTTP-запитів у веб-застосунку було створено файл `router.php`, розміщений у публічній директорії `public/`. Цей файл виконує функцію простого маршрутизатора, який аналізує вхідні запити, визначає який контролер має їх обробити, та повертає відповідь у форматі JSON (рисунок 3.1.1.9).

```php
if ($apiScope === 'admin') {

    switch ($resource) {
        case 'users':
            require_once __DIR__ . '/../app/http/controllers/user_controller.php';
            $controller = new UserController();

            if ($method === 'GET' && !$id) $actionMethod = 'ajaxListUsers';
            elseif ($method === 'GET' && $id) { $actionMethod = 'ajaxGetUser'; $params['id'] = $id; }
            elseif ($method === 'POST' && !$id) $actionMethod = 'ajaxCreateUser';
            elseif ($method === 'POST' && $id) { $actionMethod = 'ajaxUpdateUser'; $params['id'] = $id; }
            elseif ($method === 'DELETE' && $id) { $actionMethod = 'ajaxDeleteUser'; $params['id'] = $id; }

            break;

        case 'surveys':
            require_once __DIR__ . '/../app/http/controllers/survey_controller.php';
            $controller = new SurveyController();
            if ($method === 'GET' && !$id) $actionMethod = 'ajaxListSurveys';
            elseif ($method === 'GET' && $id) { $actionMethod = 'ajaxGetSurvey'; $params['id'] = $id; }
            elseif ($method === 'POST' && !$id) $actionMethod = 'ajaxCreateSurvey';
            elseif ($method === 'POST' && $id) { $actionMethod = 'ajaxUpdateSurvey'; $params['id'] = $id; }
            elseif ($method === 'DELETE' && $id) { $actionMethod = 'ajaxDeleteSurvey'; $params['id'] = $id; }
            break;

        case 'answers':
            require_once __DIR__ . '/../app/http/controllers/answer_controller.php';
            $controller = new AnswerController();

            if ($method === 'POST' && $id === 'deleteBySurvey') {
                $actionMethod = 'ajaxDeleteAllAnswersForSurvey';
            }
            break;

    }
} else {

    switch ($resource) {
        case 'surveys':
            require_once __DIR__ . '/../app/http/controllers/survey_controller.php';
            $controller = new SurveyController();
            if ($method === 'GET' && !$id) { $actionMethod = 'getPublicSurveys'; }
            elseif ($method === 'GET' && $id) {  $actionMethod = 'getPublicSurveyDetails'; $params['id'] = $id;  }
            break;

        case 'questions':
            require_once __DIR__ . '/../app/http/controllers/question_controller.php';
            $controller = new QuestionController();
            if ($method === 'GET' && isset($_GET['survey_id'])) {
                $actionMethod = 'getQuestionsForSurvey'; $params['survey_id'] = $_GET['survey_id'];
                exit;
            }
            break;

        case 'answers':
            require_once __DIR__ . '/../app/http/controllers/answer_controller.php';
            $controller = new AnswerController();
            if ($method === 'POST' && !$id) {
                $actionMethod = 'ajaxSubmitAnswers';
            }
            break;
    }
}
```

Рисунок 3.1.1.9 – Частковий лістинг `router.php`

Задля забезпечення коректної роботи маршрутизації, так як використовується програма AMPPS для локального серверного середовища, де Apache виступає веб-сервером для обробки запитів, було створено спеціальний конфігураційний файл `.htaccess`, розміщений у директорії `public/` [14]. Він

використовується для перенаправлення всіх запитів до одного вхідного файлу `index.php` з метою реалізації «чистих» URL-ів (рисунок 3.1.1.10).

```
RewriteEngine On

RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^ index.php [QSA,L]

<IfModule mod_headers.c>
    Header set Access-Control-Allow-Origin "*"
    Header set Access-Control-Allow-Methods "GET, POST, PUT, DELETE, OPTIONS"
    Header set Access-Control-Allow-Headers "Content-Type, Authorization"
</IfModule>
```

Рисунок 3.1.1.10 – Конфігураційний файл `.htaccess`

Також додано підтримку CORS через файл `config/cors.php`, що забезпечує можливість доступу до API з frontend-додатків, які працюють на інших портах або доменах. Запити обробляються з урахуванням preflight OPTIONS, а також перевірки прав доступу (рисунок 3.1.1.11).

```
$allowedOrigins = [
    'http://localhost',
    'http://localhost:3000',
    'http://127.0.0.1:5500',
];


$origin = $_SERVER['HTTP_ORIGIN'] ?? '';

if (in_array(needle: $origin, haystack: $allowedOrigins)) {
    header(header: "Access-Control-Allow-Origin: $origin");
    header(header: "Access-Control-Allow-Credentials: true");
}

header(header: "Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS");

header(header: "Access-Control-Allow-Headers: Content-Type, Authorization");


if ($_SERVER['REQUEST_METHOD'] === 'OPTIONS') {
    http_response_code(response_code: 200);
    exit;
}
```

Рисунок 3.1.1.11 – Реалізація безпеки в системі

Звідси можна приступати до реалізації клієнтської частини, тому що серверна частина веб-застосунку вже є структурованою та готовою до використання як у локальному, так і в хмарному середовищі. Вона повністю підтримує зберігання, редагування та обробку даних користувачів, та опитувань із можливістю їх розширення.

3.1.2 Розробка клієнтської частини

Клієнтська частина веб-застосунку реалізована з використанням стандартного стеку веб-технологій – HTML, CSS та JavaScript. Основна мета клієнтської частини полягає в забезпеченні інтуїтивного інтерфейсу для користувача. Мова розмітки HTML (HyperText Markup Language) є основою клієнтської частини веб-застосунку. У цьому проєкті HTML використовується для структурування та відображення інформації на екрані користувача. Кожна веб-сторінка реалізована як окремий .html-файл, що відповідає певній функціональності системи [15]. Приклад використання її в проєкті (рисунок 3.1.2.1).

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Онлайн-Опитування</title>
    <link rel="stylesheet" href="css\styles.css">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
    <link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;500;600;700&display=swap" rel="stylesheet">
    <link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600&family=Poppins:wght@600&display=swap" rel="stylesheet">
    <script src="https://kit.fontawesome.com/a076d05399.js" crossorigin="anonymous"></script>
</head>
<body>
<header>
    <div class="logo"><b>SurveyHub</b></div>
    <nav>
        <ul>
            <li><a href="/index.html"><i class="fas fa-home"></i> Головна</a></li>
            <li><a href="/about.html"><i class="fas fa-info-circle"></i> Про нас</a></li>
            <li><a href="/surveys.html"><i class="fas fa-poll"></i> Опитування</a></li>
            <li><a href="/serch_survey.html"><i class="fas fa-search"></i> Пошук опитувань</a></li>
            <li><a href="/create_survey.html"><i class="fas fa-plus-circle"></i> Створення опитувань</a></li>
            <li><a href="/profile.html"><i class="fas fa-user-circle"></i> Мій Профіль</a></li>
            <li><a href="/login.html">Вхід</a></li>
            <li><a href="/register.html" class="btn btn-success">Реєстрація</a></li>
        </ul>
    </nav>
    <label class="theme-switch">
        <input type="checkbox" id="themeToggle">
        <span class="slider"></span>
    </label>
```

Рисунок 3.1.2.1 – Використання HTML в розробці головної сторінки

Також у створенні інтерфейсів було використано фреймворк Bootstrap, він використовуватиметься для прискорення розробки клієнтської частини [17]. Bootstrap підключено через CDN (Content Delivery Network) у <head> HTML-документів (рисунок 3.1.2.2).

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
<link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;500;600;700&display=swap" rel="stylesheet">
<link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600&family=Poppins:wght@600&display=swap" rel="stylesheet">
<script src="https://kit.fontawesome.com/a076d05399.js" crossorigin="anonymous"></script>
```

Рисунок 3.1.2.2 – Підключення Bootstrap та шрифтів

У веб-застосунку CSS використано для візуального оформлення HTML-елементів та створення адаптивного, зручного для користувача інтерфейсу [16]. Каскадні таблиці стилів дозволили відокремити структуру від представлення вигляду, що якраз і надало вигляду сучасної веб-розробки (рисунок 3.1.2.3)

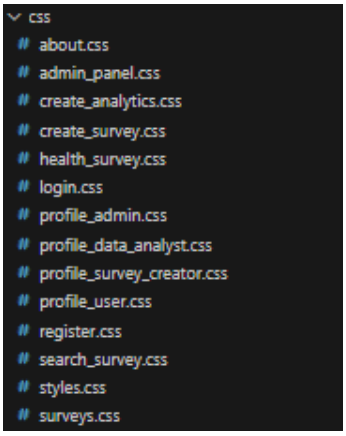


Рисунок 3.1.2.3 – Вигляд структури CSS файлів в проєкті

В кожному з цих файлів детально описано налаштування, задано стиль хедера та футера, та вигляд для кожної основної частини HTML файлу. Також задано адаптивність інтерфейсу, що забезпечує оптимізацію для кожної системи та екрану, через який буде здійснено вхід на сайт (рисунок 3.1.2.4).

```css
/* Адаптивність */
@media (max-width: 768px) {
    header {
        flex-direction: column;
        text-align: center;
    }
    nav ul {
        flex-direction: column;
        margin-top: 10px;
    }
    nav ul li {
        margin: 5px 0;
    }
    main {
        padding: 30px 16px;
    }
    .about-content {
        padding: 30px 20px;
    }
    .about-content h1 {
        font-size: 24px;
    }
    .about-content p, .about-content li {
        font-size: 15px;
    }
    .cta-text {
        font-size: 16px;
    }
}
@media (max-width: 480px) {
    .about-content {
        padding: 20px 16px;
    }
    .about-content h1 {
        font-size: 22px;
    }
    .about-content p, .about-content li {
        font-size: 14px;
    }
}
```

Рисунок 3.1.2.4 – Лістинг коду адаптивності сайту

JavaScript використовується в застосунку для динамічної взаємодії між клієнтською частиною (HTML) та серверною частиною (PHP) [18]. Основна роль JavaScript – обробка подій, надсилання даних без перезавантаження сторінки, валідація форм і реалізація асинхронних запитів до API за допомогою AJAX [19]. Це досить зручно для онлайн-опитувань, тому що кількість опитувань може бути великою, а використання таких запитів спростить задачу в їх пошуку та показу (рисунок 3.1.2.5 та 3.1.2.6).

```javascript
async function loadSurvey() {
    if (!questionsContainer || !surveyTitleElement) return;

    questionsContainer.innerHTML = '<p class="loading-questions">Завантаження питань...</p>';
    surveyTitleElement.textContent = 'Завантаження назви...';
    if (surveyDescriptionElement) surveyDescriptionElement.textContent = '';

    try {
        const response = await fetch(surveyLoadUrl);
        if (!response.ok) {
            throw new Error(`HTTP error! status: ${response.status}`);
        }
        const data = await response.json();

        if (data.success && data.survey) {
            const survey = data.survey;
            surveyTitleElement.textContent = escapeHTML(survey.survey_name || "Назва опитування");
            if (surveyDescriptionElement && survey.description) {
                surveyDescriptionElement.textContent = escapeHTML(survey.description);
            }
            document.title = `${escapeHTML(survey.survey_name)} | Проходження опитування`;
```

Рисунок 3.1.2.5 – Використання JavaScript для відображення опитувань

```
document.getElementById("searchForm").addEventListener("submit", function (e) {
  e.preventDefault();

  const keyword = document.getElementById("searchInput").value;

  fetch(`router.php?r=surveys&search=${encodeURIComponent(keyword)}`)
    .then(res => res.json())
    .then(data => {
      const container = document.getElementById("resultsContainer");
      container.innerHTML = ""; // Очистити попередні результати

      if (data.length === 0) {
        container.innerHTML = "<p>Нічого не знайдено.</p>";
      } else {
        data.forEach(survey => {
          const div = document.createElement("div");
          div.innerHTML = `<h3>${survey.survey_name}</h3><p>${survey.description}</p>`;
          container.appendChild(div);
        });
      }
    });
});
```

Рисунок 3.1.2.6 – Використання AJAX для пошуку опитувань

Технологія AJAX дозволяє браузеру обмінюватися даними з сервером без особливих зусиль. У проєкті AJAX-запити реалізовано через fetch API, який надсилає дані на сервер (`router.php`) та отримує JSON-відповідь [19]. Розглянемо на прикладі створення опитувань в системі. з надсиланням одразу всіх даних опитування, а саме назву, питання і варіанти відповідей на сервер JSON через AJAX (рисунок 3.1.2.6).

```
function addQuestion() {
  questionCount++;
  const container = document.getElementById('questionsContainer');
  const questionBlock = document.createElement('div');
  questionBlock.classList.add('question-block');

  questionBlock.innerHTML = `
    <label for="question${questionCount}">Питання ${questionCount}:</label>
    <input type="text" id="question${questionCount}" name="questions[]" required>

    <label for="type${questionCount}">Тип відповіді:</label>
    <select id="type${questionCount}" name="types[]" onchange="handleTypeChange(this, ${questionCount})">
      <option value="multiple">Множинний вибір</option>
      <option value="yesno">Так / Ні</option>
      <option value="open">Відкрита відповідь</option>
    </select>

    <div id="options${questionCount}" class="options">
      <label>Варіанти відповіді:</label>
      <input type="text" name="option${questionCount}[]" placeholder="Варіант 1" required>
      <input type="text" name="option${questionCount}[]" placeholder="Варіант 2" required>
      <button type="button" onclick="addOption(${questionCount})">+ Додати варіант</button>
    </div>
  `;

  container.appendChild(questionBlock);
}

// Додає варіант відповіді
function addOption(questionId) {
  const optionsDiv = document.getElementById(`options${questionId}`);
  const input = document.createElement('input');
  input.type = 'text';
  input.name = `option${questionId}[]`;
  input.placeholder = `Варіант ${optionsDiv.querySelectorAll('input').length + 1}`;
  input.required = true;
  optionsDiv.insertBefore(input, optionsDiv.lastElementChild);
}
```

Рисунок 3.1.2.6 – Часковий лістинг коду JS

У цьому випадку форма знаходиться в `create_survey.html`, запит відправляється на `router.php?r=surveys` через JavaScript і в кінці все обробляє `SurveyController` на бекенді (рисунок 3.1.2.7).

```
document.getElementById("createSurveyForm").addEventListener("submit", function (e) {
  e.preventDefault();

  const surveyData = {
    survey_name: document.getElementById("title").value,
    description: document.getElementById("description").value,
    creator_id: 1,
    questions: []
  };

  for (let i = 1; i <= questionCount; i++) {
    const questionText = document.getElementById(`question${i}`).value;
    const type = document.getElementById(`type${i}`).value;
    const options = Array.from(document.querySelectorAll(`input[name='option${i}[]']`)).map(input => input.value);

    surveyData.questions.push({
      text: questionText,
      type: type,
      options: options
    });
  }

  fetch("router.php?r=surveys", {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify(surveyData)
  })
    .then(response => response.json())
    .then(response => {
      if (response.success) {
        alert("Опитування створено!");
        window.location.href = "surveys.html";
      } else {
        alert("Помилка під час створення.");
      }
    });
```

Рисунок 3.1.2.7 – Використання AJAX для створення опитувань

Звідси можна приступати до тестування функціоналу клієнтської частини та серверної частина, так як веб-застосунок вже є структурованим та готовим до використання як у локальному, так і в хмарному середовищі.

3.2 Тестування веб-застосунку

Для тестування веб-застосунку використаємо браузер Google [20]. Спочатку відкриємо головну сторінку, ввівши у адресному рядку браузера домен хоста клієнта, який хоститься на 8080 порті, адреса має вигляд: http://localhost:8080/main. Головна сторінка буде мати вигляд (рисунок 3.2.1).

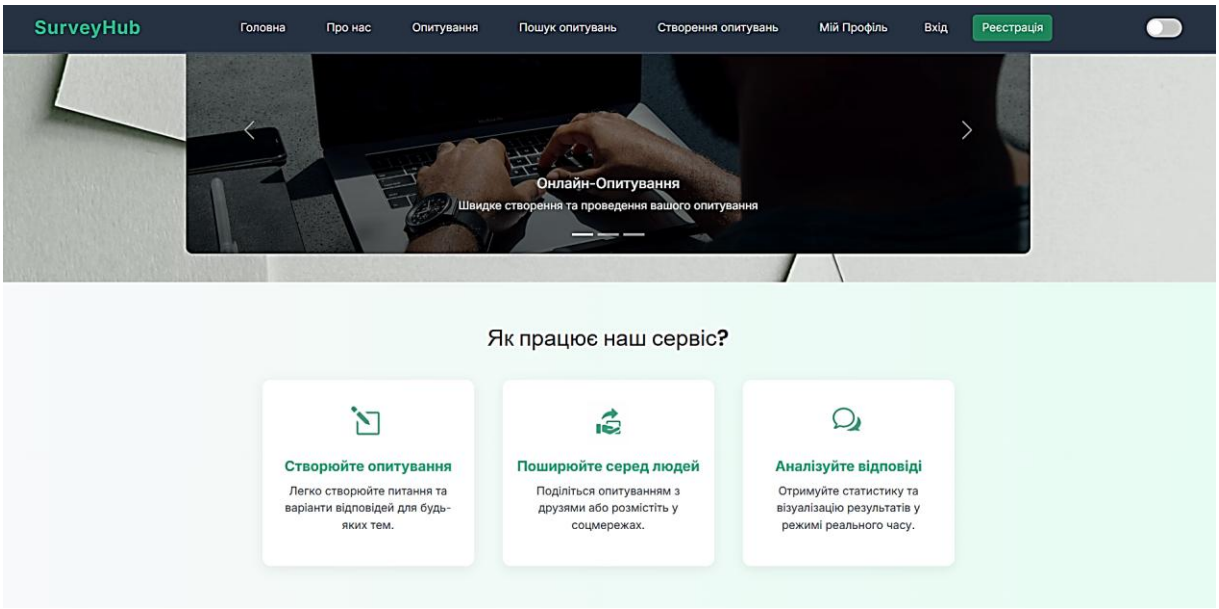


Рисунок 3.2.1 – Головна сторінка веб-застосунку

Дана сторінка виступає візитною карткою для веб-застосунку, так як на ній знаходиться весь функціонал, опис та правила користування застосунком. Основною її задачею є зацікавити відвідувача залишитись на сайті та скористатись сервісом. Вона поєднує в собі інформативність і зручність, що є важливим для першого враження про платформу.

У верхній частині сторінки знаходиться навігаційне меню (рисунок 3.2.2).

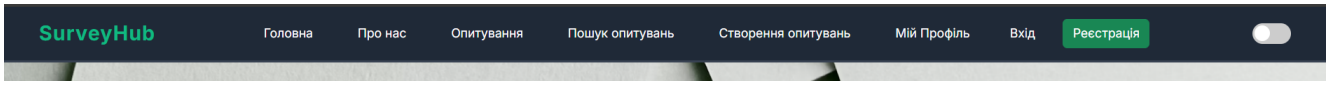


Рисунок 3.2.2 – Вигляд навігаційного меню

Меню дозволяє швидко перейти до основних функцій системи: сторінки «Про нас», перегляду опитувань і їх пошуку, створення опитувань, профілю користувача, реєстрації, входу та зміни теми сторінки(темна чи світла теми). Вигляд сторінки «Про Нас» в якій надано інформацію про веб-застосунок, його функціонал та з можливістю приєднатись в систему подано на рисунку 3.2.3.

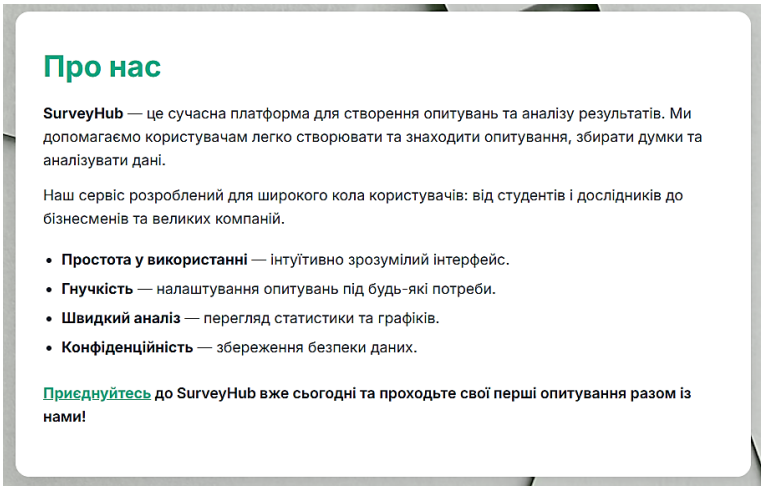


Рисунок 3.2.3 – Вигляд сторінки «Про Нас»

Також у нижній частині головної сторінки розміщено окремий вітальний блок, який виконує роль заклику до дій, а саме початку свторення опитувань, зображено на рисунку 3.2.4.

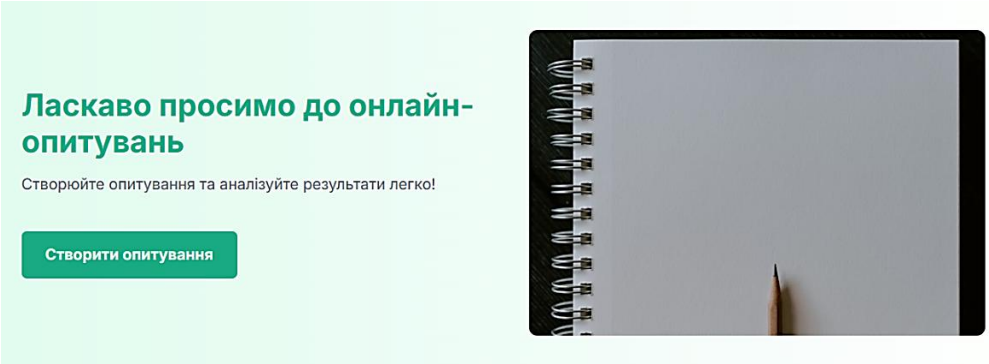


Рисунок 3.2.4 – Вигляд вітального блоку

Цей блок містить коротке і чітке повідомлення. Поряд із текстом розташоване тематичне зображення. А знизу кнопка «Створити опитування», що дозволяє направити користувача до головного функціоналу застосунку. Однак перед тим як створити опитування, потрібно визначити роль користувача від респондента(людини яка лише проходитиме опитування) до адміністатора. Цей процес відбувається на сторінці авторизації (рисунок 3.2.5).

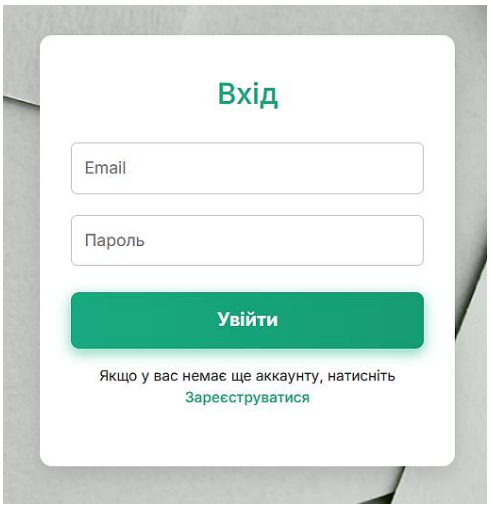


Рисунок 3.2.5 – Сторінка авторизації

Якщо користувач не має облікового запису в системі, то у цьому випадку передбачено кнопку «Зареєструватися», натиснувши на яку користувач автоматично перенаправляється на сторінку реєстрації (рисунок 3.2.6).

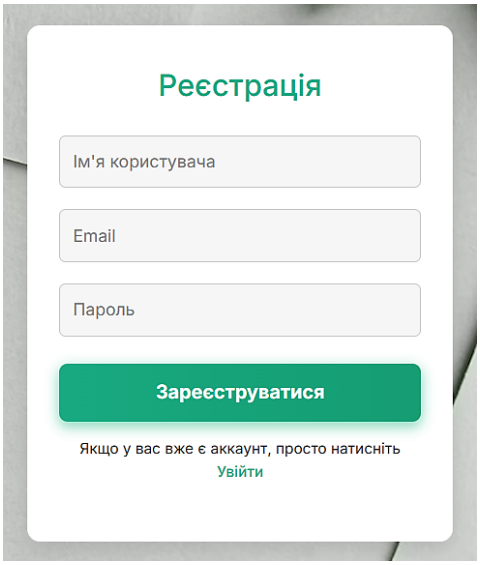


Рисунок 3.2.6 – Сторінка реєстрації

Після реєстрації та авторизації у систему користувача переводить на його особисту сторінку. Так як зареєстрований користувач є респондентом, то його переводить на сторінку респондента (рисунок 3.2.7).

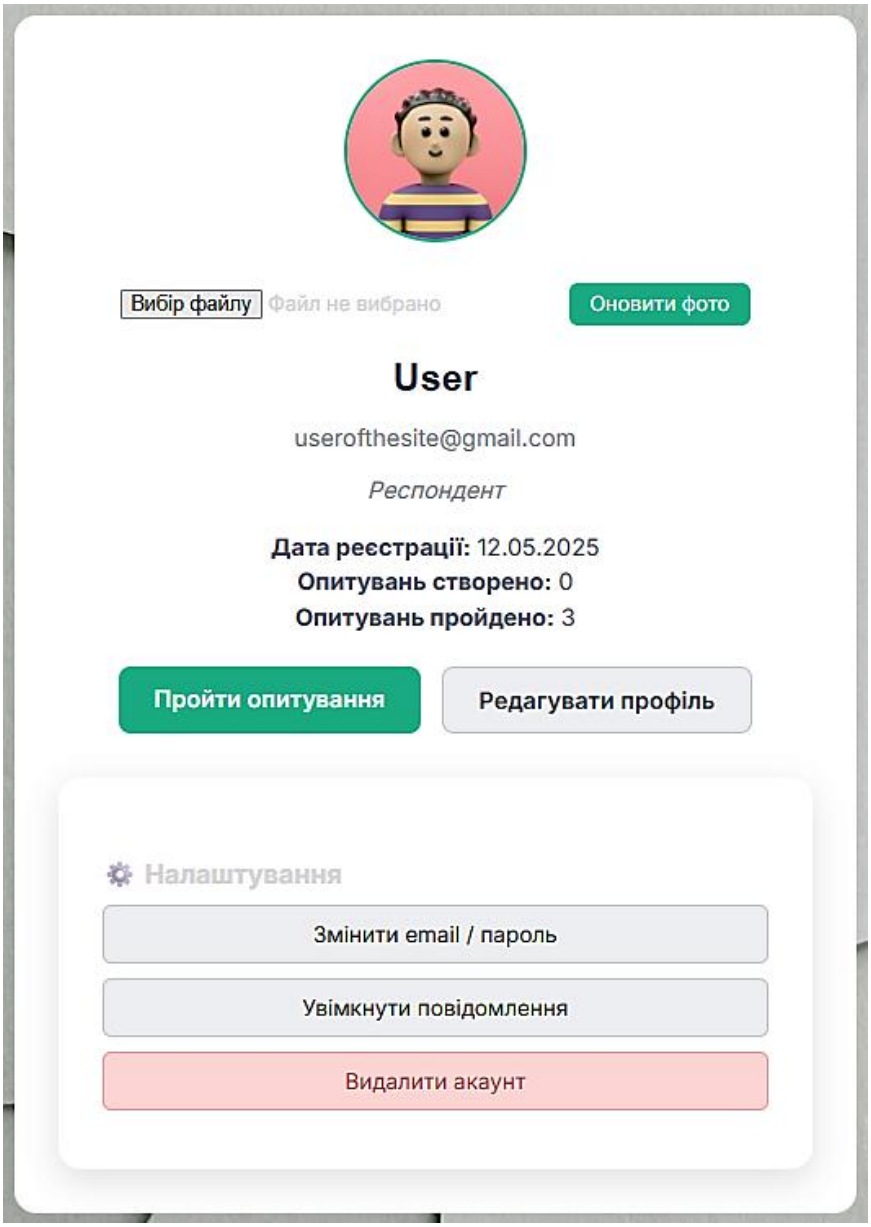


Рисунок 3.2.7 – Сторінка респондента

Ця сторінка є профілем користувача, який зареєстрований у системі як респондент, тобто особа, яка проходитиме опитування. Тут знаходиться основна інформація про обліковий запис, а саме: ім’я користувача, електронна адреса, біографія користувача, дата реєстрації, а також статистика щодо створених та пройдених опитувань. На його сторінці присутня кнопка «Пройти опитування», за допомогою якої його перекидує на сторінку опитувань, а також редагувати особисту інформацію за допомогою кнопки «Редагувати профіль».

У блоці «Налаштування» доступні опції зміни електронної пошти або паролю, увімкнення/вимкнення повідомлень, а також можливість видалити обліковий запис.

На сторінці також присутній аватар, який користувач може змінити, завантаживши новий файл за допомогою кнопки «Оновити фото».

Оскільки користувач зареєстрований у системі як респондент, то його можливості в системі обмежені, основною і найважливішою його функцією є проходження опитувань, список вже створених опитувань зображено на рисунку 3.2.8.

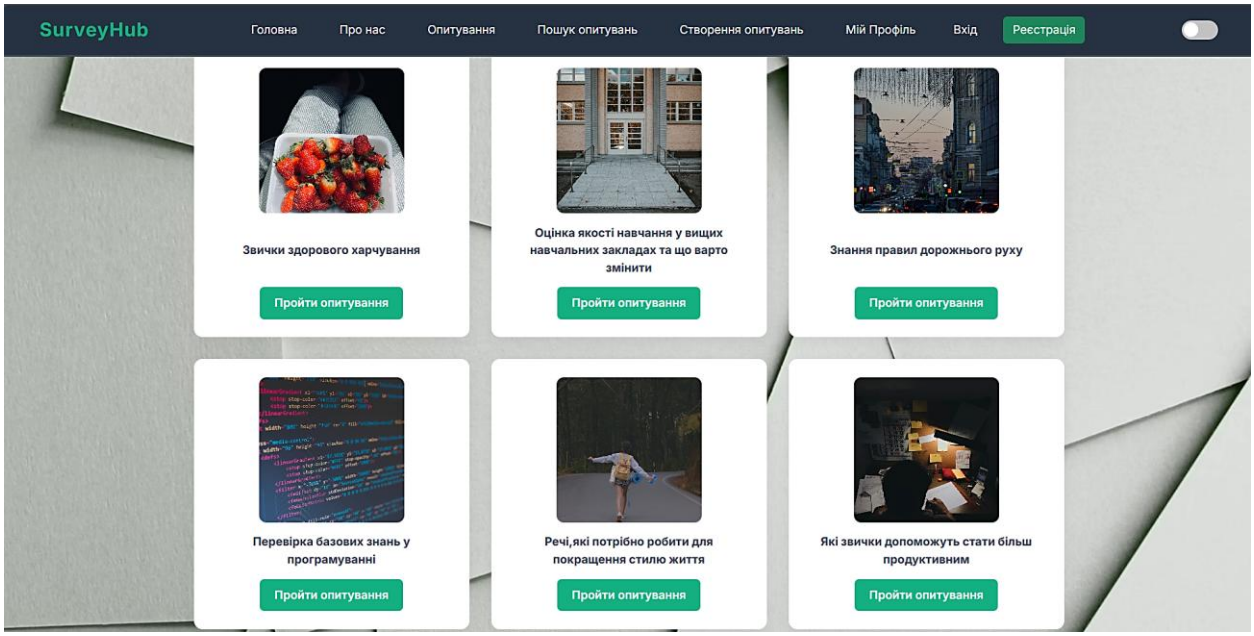


Рисунок 3.2.8 – Сторінка опитувань

Дана сторінка відображає наявні опитування, які вже є у застосунку у вигляді блоків, розташованих у колонках і розподілених по сітці. Кожна картка опитувань містить зображення,яке візуально відображає тему опитування, саму назву та кнопку «Пройти опитування», яка перенаправляє на сторінку заповнення форм опитувань на надсилання відповідей (рисунок 3.2.9).

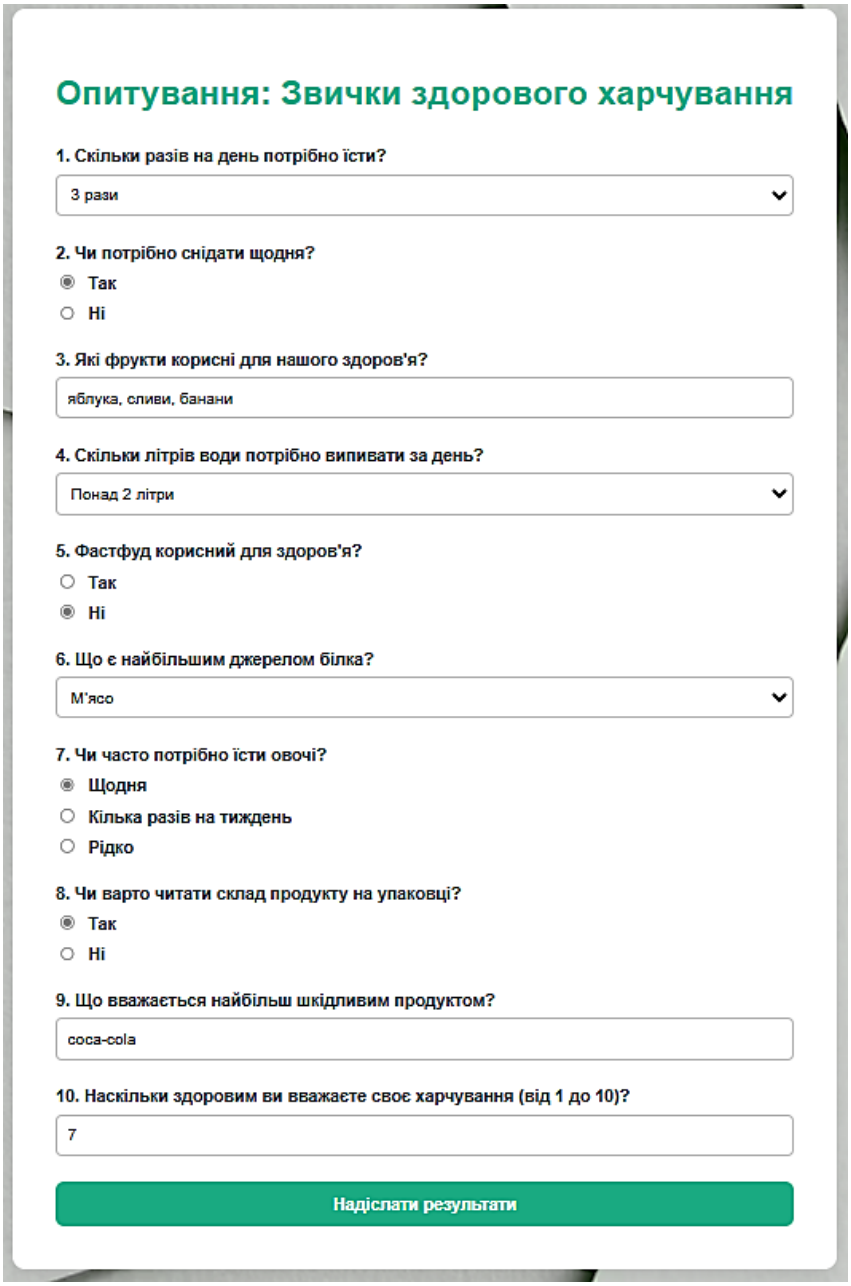


Рисунок 3.2.9 – Сторінка форми опитування

Конкретно ця сторінка дозволяє респонденту проходити опитування, надаючи відповіді на різні види питань. В кінці є змога надіслати результати опитування, які далі будуть передані аналітику даних, який проаналізувавши їх відобразить відсоток правильності в профілі користувача (рисунок 3.2.10).

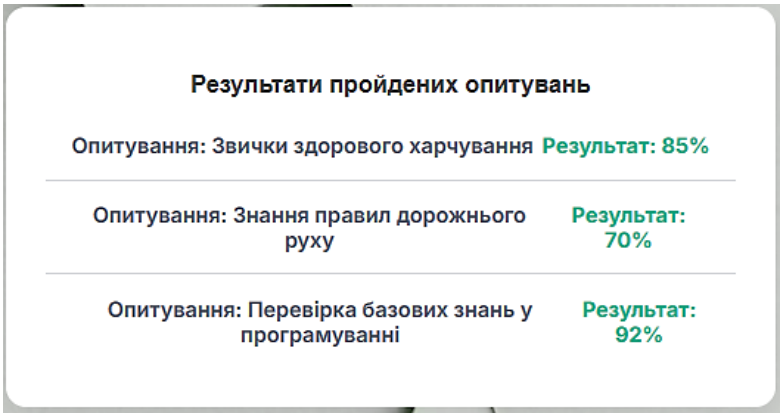


Рисунок 3.2.10 – Результати опитувань

Також при збільшенні кількості опитувань на сайті створена сторінка «Пошук опитувань», яка за ключовими словами знаходить потрібні з усіх, що значно заощаджує час респондента (рисунок 3.2.11).

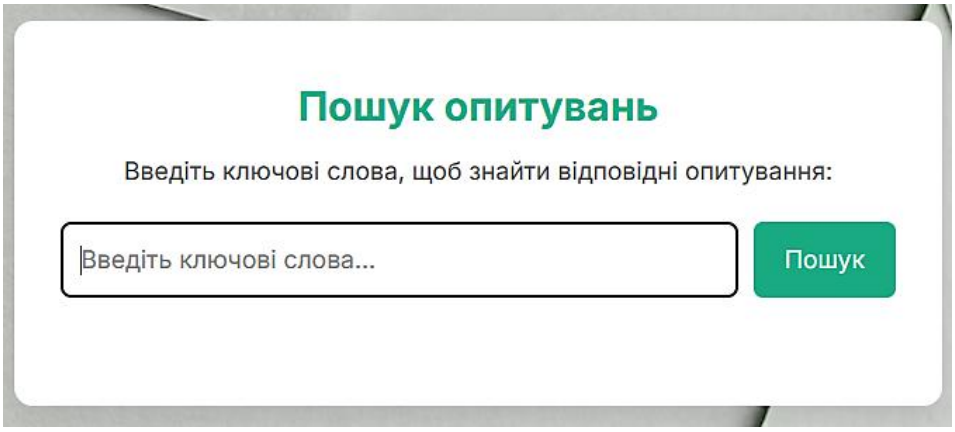


Рисунок 3.2.11 – Сторінка пошуку опитувань

Нехай користувачу потрібно знайти опитування, яке містить у собі слово «перевірка». Результат даного пошуку продемонстровано на рисунку 3.2.12.

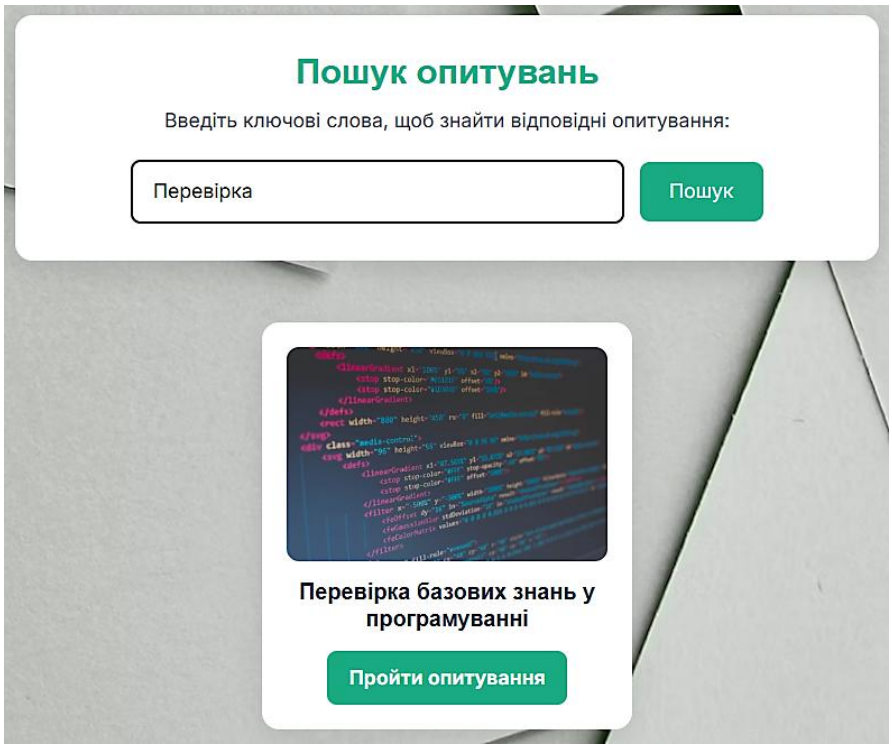


Рисунок 3.2.12 – Результат пошуку

Розглянемо варіант, коли користувач входить в систему під ролюю «Створювача опитувань» іншими словами «Соціолога». Дана роль надає доступ до розширеного функціоналу, який дозволяє створювати, редагувати та публікувати опитування. Це один з ключових типів користувачів системи, між респондентами та адміністраторами. Профіль «Соціолога» зображено на рисунку 3.2.13.

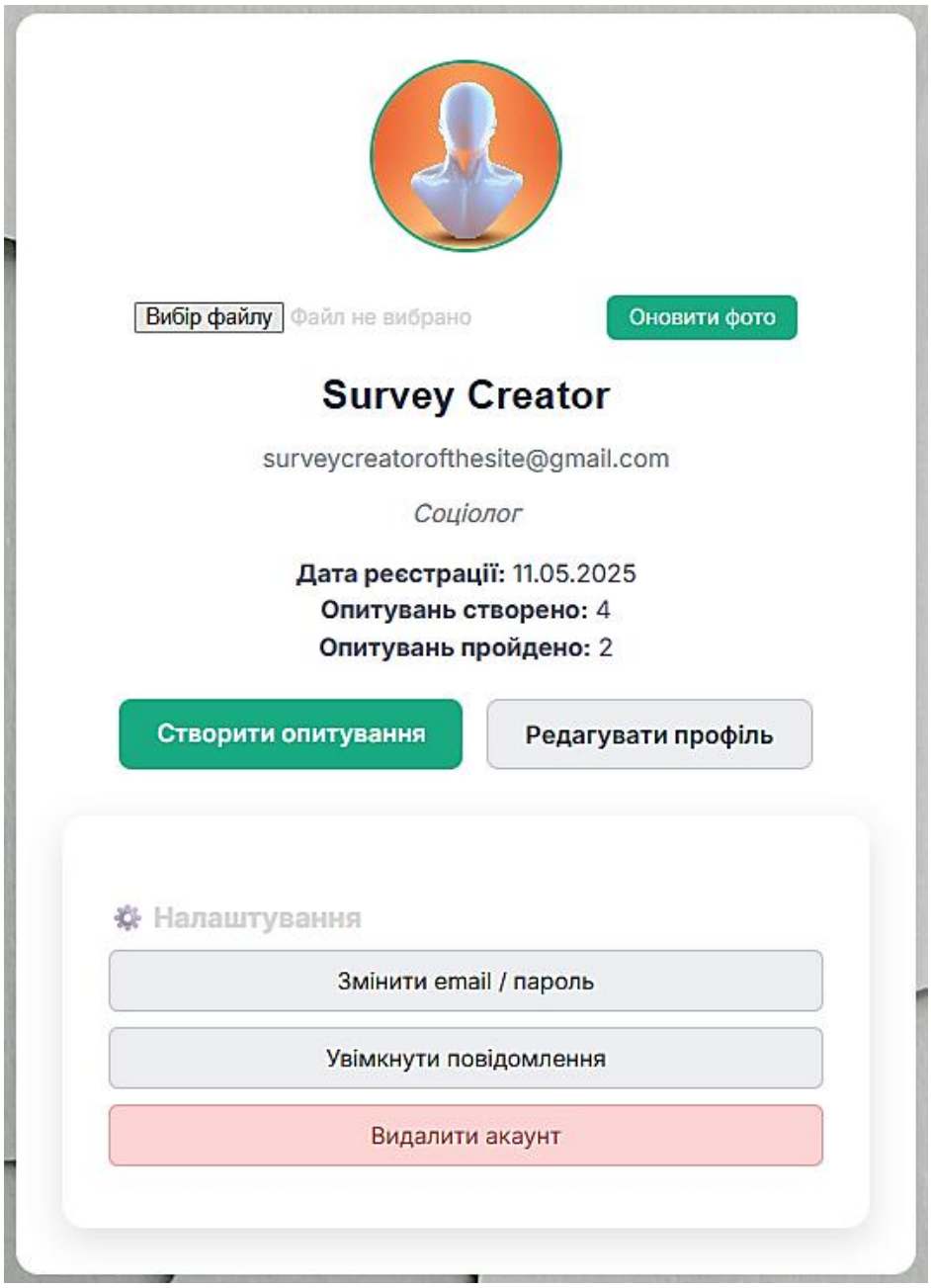


Рисунок 3.2.13 – Вигляд профілю «Створювача опитувань»

Ця сторінка дозволяє переглядати та змінювати персональну інформацію, створювати опитування, а також керувати налаштуваннями облікового запису. Однак на відміну від просто респондента, користувач який зареєстрований у системі як «Створювач опитувань» має більше можливостей. На його сторінці присутня кнопка «Створити опитування», натиснувши на яку перекидує на сторінку створення опитувань, зображену на рисунку 3.2.14.

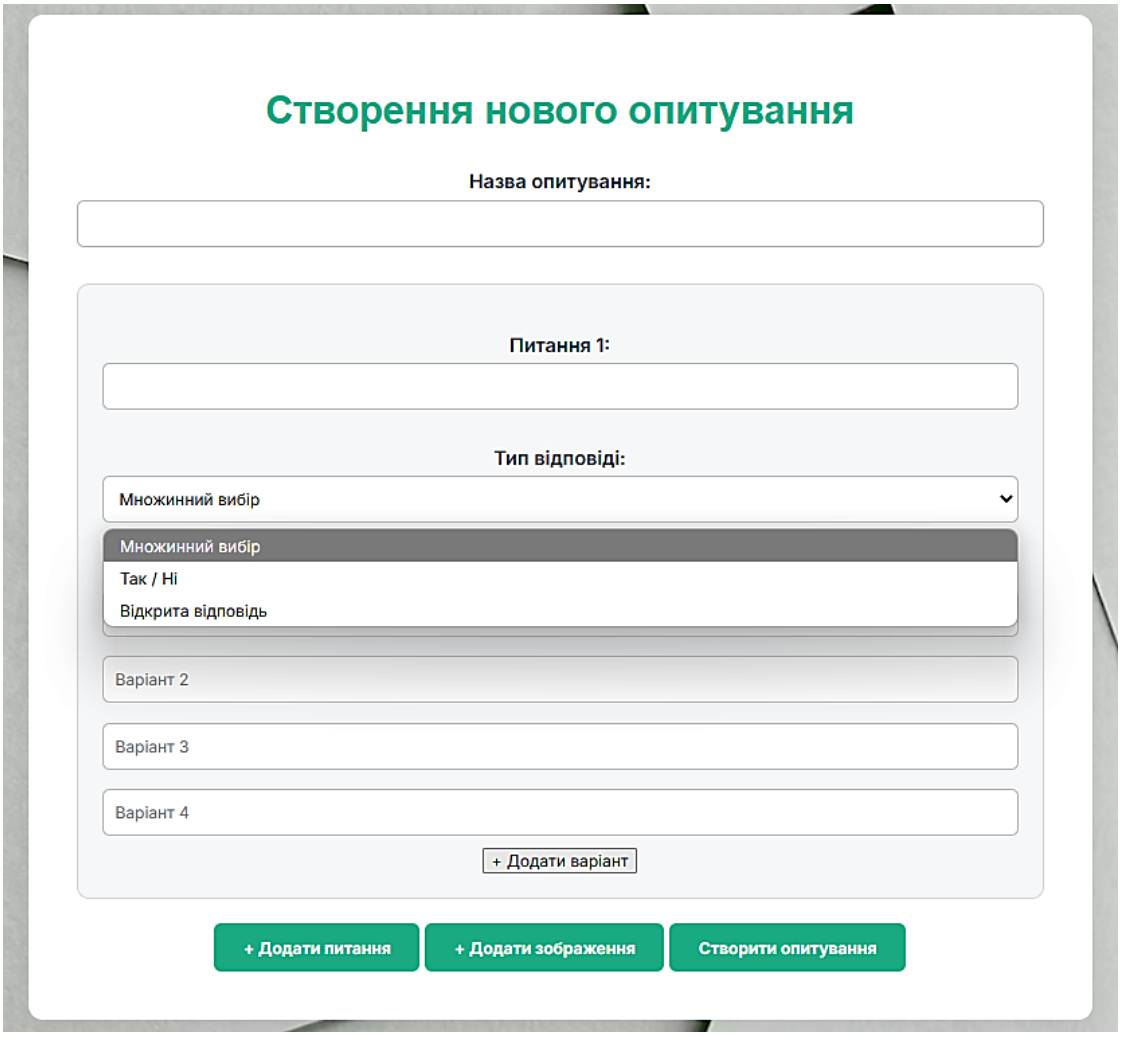


Рисунок 3.2.14 – Вигляд сторінки «Створення нового опитування»

Сторінка слугує для створення нового опитування. Структура блоку передбачає в собі поля для введення заголовку опитування, додавання запитань та вписування варіантів відповідей. Створювач опитувань може самостійно обрати формат відповідей від варіантів із кількома виборами, простих «Так або Ні» до відкритих форм, яка дає змогу респонденту вільно висловлювати власну думку.

Після налаштування основного змісту опитування є можливість прикріпити зображення за допомогою кнопки «Додати зображення». А для підтвердження публікації створеного опитування треба скористатись кнопкою «Створити опитування», після чого воно стане доступним на окремій сторінці з усіма наявними опитуваннями.

Також кожен «Соціолог» у власному профілі може переглянути кількість створених опитувань, їх назви та число питань (рисунок 3.2.15).

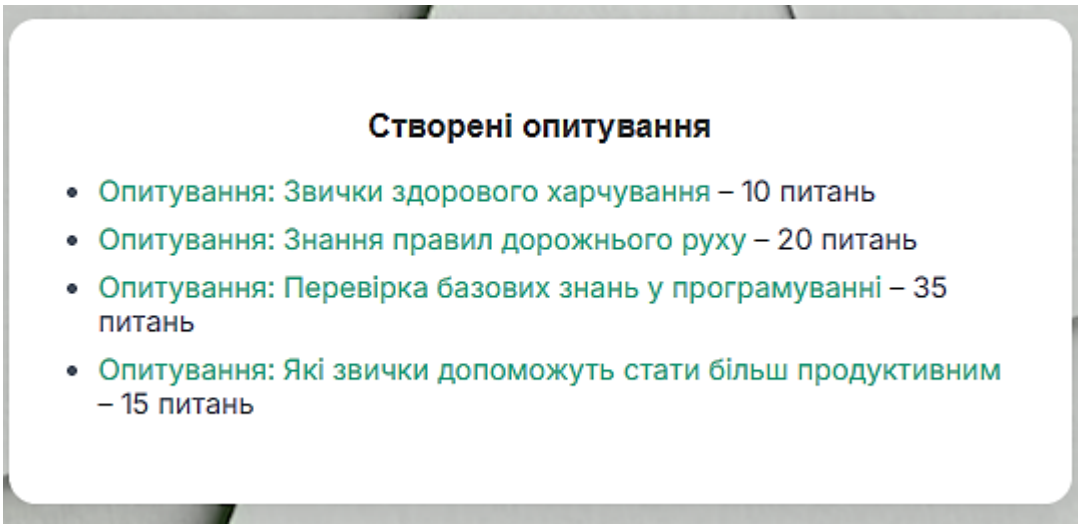


Рисунок 3.2.15 – Вигляд створених опитувань в профілі

Розглянемо наступну роль веб-застосунку, а саме «Аналітик опитувань». Користувач з цією роллю має доступ до розширених функцій, пов’язаних із переглядом та інтерпретацією результатів опитувань. Після натискання на кнопку «Проаналізувати опитування», користувач буде перенаправлений на аналітичну сторінку веб-застосунку. Ця сторінка включає список опитувань з можливістю перегляду варіантів відповідей та внесення відстотків правильності кожного з них в профілі респондентів. Аналітик не має права редагувати зміст опитувань, додавати нові запитання або змінювати ролі інших користувачів, вигляд профілю зображено на рисунку 3.2.16.

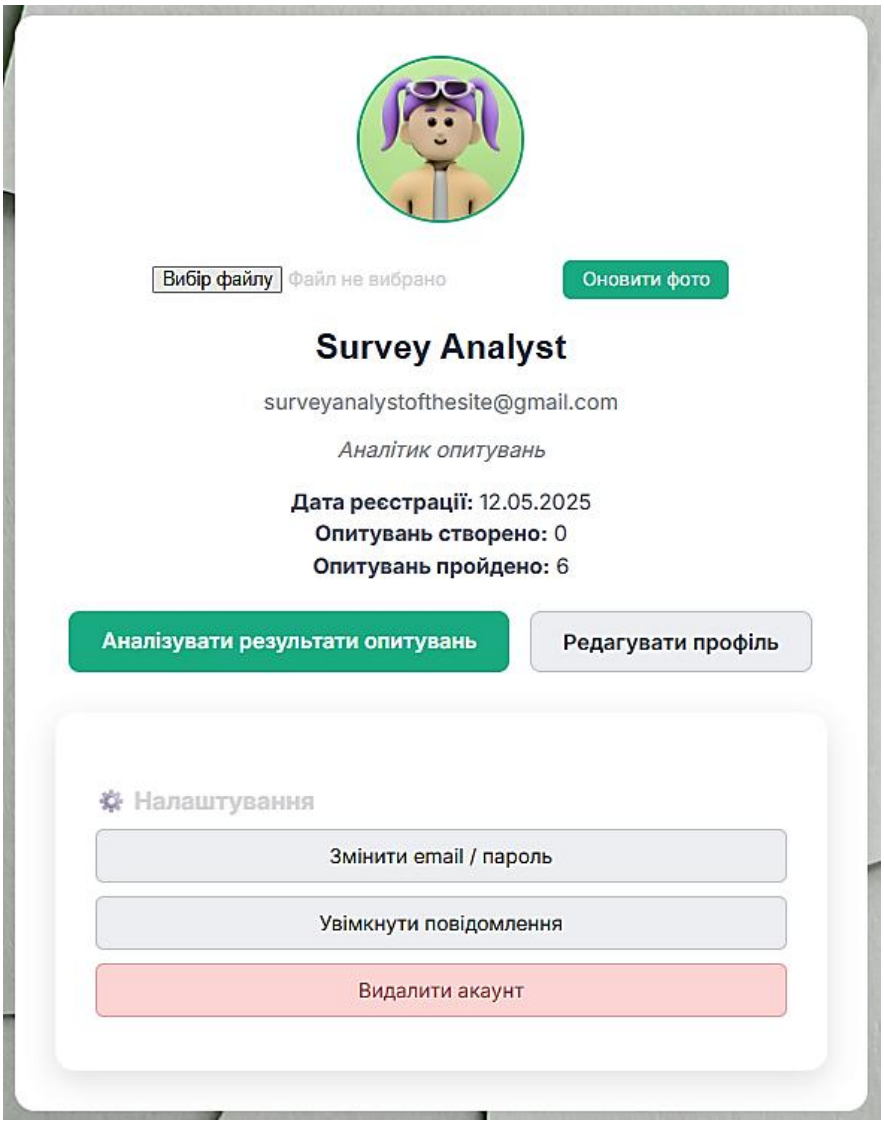


Рисунок 3.2.16 – Вигляд профілю «Аналітика Опитувань»

Завершуючи огляд ролей користувачів у веб-застосунку, розглянемо, як оформлений профіль «Адміністратора». Візуальна частина мало чим відрізняється від інших ролей — аватар, електронна пошта, дата реєстрації і т.д. Також доступні функції, як «Проаналізувати опитування» та «Створити опитування». Однак «Панель користування системою» надає доступ до повного списку зареєстрованих користувачів з можливістю редагування їхніх ролей, а також може деактивовувати або остаточно видалити будь-яке опитування. Ця функція є свого роду міні-версія phpMyAdmin, але призначена для керування не всією базою даних, а лише тими аспектами, які критично важливі для роботи опитувань (рисунок 3.2.17).

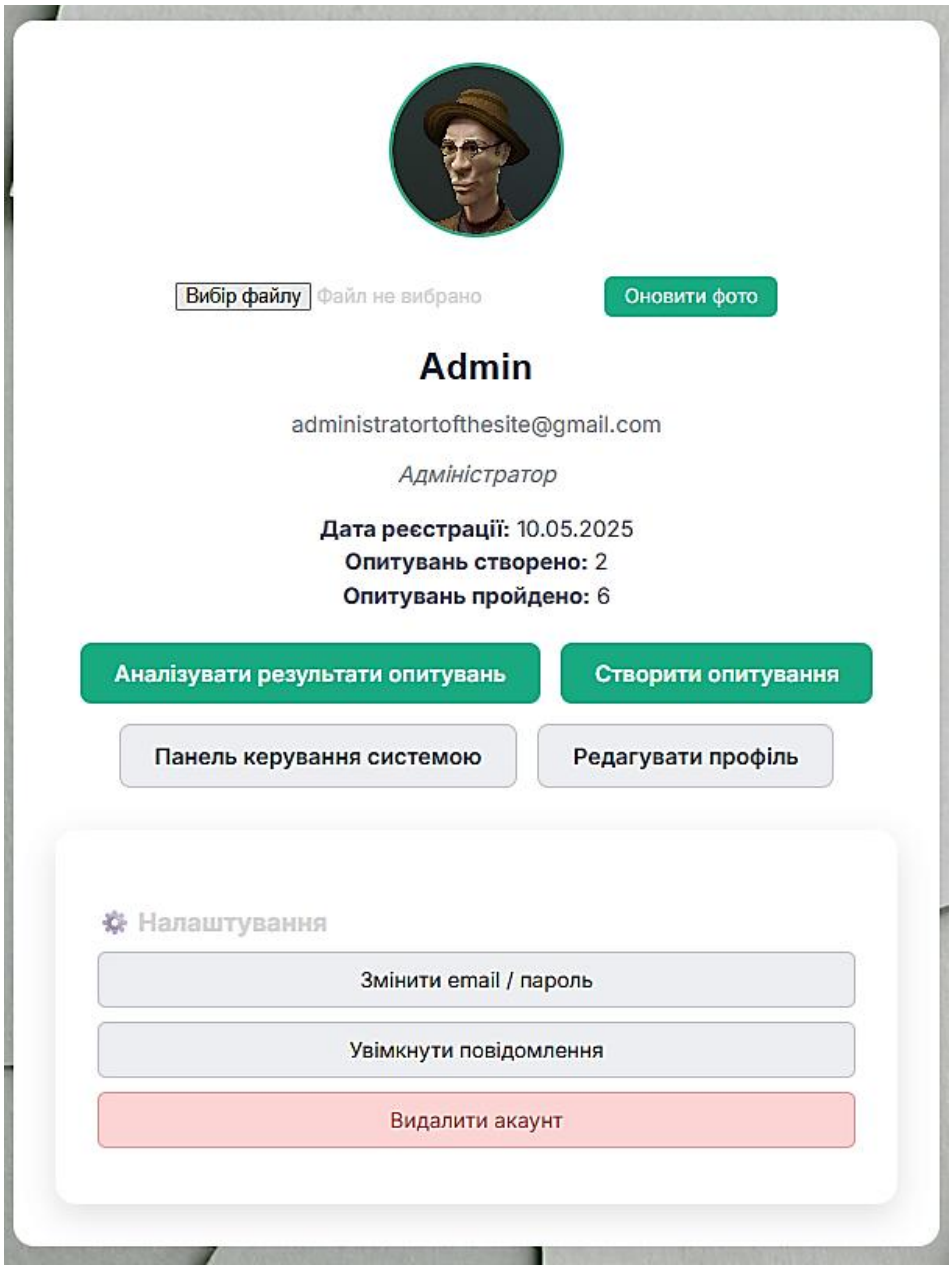


Рисунок 3.2.17 – Вигляд профілю «Адміністратора»

Таким чином функціонал та роботу веб-застосунку було успішно протестовано та доведено коректність його роботи.

4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Соціально-політичні небезпеки, їхні види та характеристики

Соціально-політичні небезпеки часто виникають у контексті суспільно-політичних конфліктів. Конфлікт найчастіше описують як протистояння двох чи більше сил із протилежними позиціями, які намагаючись реалізувати власні інтереси, стикаються з активною протидією [28].

Конфлікти можна класифікувати за способом їх урегулювання на дві великі групи: антагоністичні (насильницькі) та компромісні (ненасильницькі).

Насильницькі конфлікти полягають у силовому розв'язанні суперечностей, коли йдеться про руйнування структур усіх учасників або примусовий вихід із протистояння всіх сторін, окрім однієї, що в підсумку й здобуває перемогу

Ненасильницькі чи компромісні конфлікти передбачають кілька сценаріїв розв'язання через взаємні поступки: учасники коригують свої цілі, строки та умови взаємодії, шукаючи прийнятне для всіх рішення.

Сфери, у яких проявляються конфлікти, є надзвичайно різноманітними – від політики та економіки до суспільних відносин і світоглядних установок. Політичні конфлікти протистояння щодо розподілу владних повноважень і форм боротьби за владу. Соціальні конфлікти виникають у системі міжособистісних та міжгрупових стосунків, коли посилюється зіткнення інтересів і ціннісних орієнтацій соціальних спільнот або окремих осіб. Економічні конфлікти охоплюють суперечності між економічними інтересами осіб чи груп і стосуються боротьби за ресурси, сфери впливу й розподіл власності. Такі конфлікти трапляються на всіх рівнях управління – від локального до національного.

За помітністю конфліктні протистояння поділяють на відкриті та приховані.

Відкритий конфлікт виявляється у прямому зіткненні сторін: це можуть бути сварки, публічні суперечки, силові чи вербальні сутички. Поведінку опонентів у

таких ситуаціях регламентують соціальні норми й рольові очікування, що відповідають статусу учасників та характеру події.

Прихований конфлікт не супроводжується явно агресивними діями. Сторони застосовують опосередковані способи впливу – інформаційні витоки, домовленості, дипломатичний тиск тощо. Подібний формат протистояння обирають, коли один із учасників побоюється сил суперника або не володіє достатнім ресурсом для відкритої боротьби.

За суб'єктним складом конфлікти поділяють на міжгрупові та міжособистісні.

Міжгрупові конфлікти виникають між окремими колективами, підрозділами чи соціальними спільнотами, інтереси яких у певний момент зіштовхуються. Під час протистояння члени групи демонструють високу згуртованість, що може зникнути одразу після врегулювання конфлікту.

Міжособистісні конфлікти охоплюють зіткнення окремих осіб. Це найпоширеніший різновид протистоянь, у якому сторони відстоюють власні інтереси та потреби.

Крайні форми соціально-політичних небезпек та пов'язані з ними загрози [29].

Війна – це збройне протистояння між державами та між соціальними, етнічними спільнотами. У ширшому політичному значенні це крайня форма боротьби, коли ворожі сили переходять до відкритого насильства, аби реалізувати свої інтереси. В результаті яких втрати людського ресурсу несуть дві сторони.

Також не менш серйозна небезпека є тероризм. Він став невід'ємною частиною сучасної реальності й набув великих масштабів. Він проявляється як погроза застосування крайнього насильства, зокрема фізичного знищення людей та залякування населення для досягнення трьох базових цілей політичних, релігійних чи корисливих.

Екстремальні небезпеки кримінального характеру. Значний виклик безпеці людства становить злочинність. В Україні криміногенна ситуація ускладнюється

економічною нестабільністю, різким падінням рівня життя та прогалинами у правовому регулюванні. За таких умов зростає кількість осіб, схильних до протиправної діяльності, що підсилює загрозу тяжких злочинів.

Соціальні небезпеки, такі як: алкоголізм, тютюнокуріння та наркоманія [29].

Алкоголізм має певну закономірність. Перший прийом викликає захисну реакцію – адже організм прийняв отруту. Це можуть бути нудота, блювання, головний біль, запаморочення і таке інше. Ніяких приємних відчуттів при цьому не виникає. Однак при повторних прийомах алкоголю настає ейфорія, а захисна реакція поступово слабшає. З часом стан ейфорії стає для людини потребою, і вона вже не може обходитись без алкогольних напоїв.

Аналогічна ситуація з тютюнокурінням, основною шкідливою звичкою передчасної смерті, яку навідмінно від наркотиків можна побороти.

Наркоманія, природні або синтетичні речовини, потрапляючи в організм, змінюють його функції й за багаторазового вживання формують психічну або фізичну залежність. Психічна залежність виникає тоді, коли очікуваний емоційний стан досягається лише за наявності наркотику. Фізична залежність проявляється як синдром відміни: різке припинення вживання спричиняє тяжкі фізіологічні розлади, оскільки організм адаптувався до постійної присутності речовини. Також внаслідок безконтрольного вживання наркотиків, поширюється ряд хвороб, одна з них – ВІЛ/СНІД.

Синдром набутого імунодефіциту (СНІД) спричиняється вірусом імунодефіциту людини (ВІЛ). Основні шляхи передачі інфекції:

— Незахищені статеві контакти з інфікованою особою.

— Переливання зараженої крові чи трансплантація тканин і органів.

— Повторне використання нестерильних голок і шприців та татуювання нестерильним інструментом.

— Пошкодження шкіри або слизових оболонок забрудненими ВІЛ медичними інструментами.

— Вертикальна передача: від матері до дитини під час вагітності, пологів або грудного вигодовування.

Інфекція не поширюється через побутові контакти – рукостискання, поцілунки, спільну їжу, предмети домашнього вжитку, воду в басейні чи укуси комах. Однак кожен, хто залучений до поведінкових ризиків, перебуває в зоні потенційної небезпеки.

Загалом, соціально-політичні конфлікти, глобальна злочинність і масові соціальні патології алкоголізм, тютюнокуріння та наркоманія, утворюють комплекс взаємопов'язаних загроз, що вимагають профілактики та співпраці для мінімізації їхнього руйнівного впливу на суспільство.

4.2 Соціальне значення охорони праці

Соціальна цінність системи охорони праці полягає у забезпеченні зростання ефективності суспільного виробництва через постійне удосконалення умов праці і підвищення її безпеки. Такий підхід дає змогу зменшити виробничий травматизм, а також поступово ліквідовувати робочі місця, що не відповідають вимогам охорони праці [30]. Відповідно до ст. 43 Конституції України та ст. 4 Закону України «Про охорону праці» право працівника на безпечні й здорові умови праці є невід'ємним, а його пріоритет над результатами виробничої діяльності закріплений на законодавчому рівні [31, 32].

Ці фактори виявляється у трьох головних показниках:

— підвищення продуктивності праці, внаслідок зростання робочого часу, який збільшується завдяки скороченню внутрішніх простоїв;

— зменшення цілоденних втрат робочого часу та збереження трудового потенціалу, поліпшення умов праці сприяє кращому стану здоров'я працівників і підвищує їхню професійну активність;

— зростання професійної кваліфікації, досягається через покращення попередніх показників і їхніх складових.

Охорона праці має вагоме соціальне значення, адже жодні виробничі показники не здатні компенсувати людині втрату здоров'я, а тим більше життя. За статистикою, у випадках виробничих аварій та нещасних випадків гине не просто працівник, у якого держава інвестувала час і кошти на підготовку – це передусім, годувальник сім'ї, батько чи мати дітей [30].

Тому соціальна складова охорони праці спрямована на всебічний розвиток кожного працівника та пріоритизацію життя і здоров'я у процесі трудової діяльності. Вона передбачає профілактику негативних наслідків, що виникають через недотримання вимог безпеки та виробничої гігієни. Згідно із ст. 20 Закону в підрозділі колективного договору про охорону праці [33]. Мають бути обов'язково відображені наступні заходи:

— забезпечення працівникам соціальних гарантій в області охорони праці на рівні, не нижче передбаченого законодавством;

— комплексні заходи для досягнення нормативів безпеки праці й виробничої санітарії;

— заходи з підвищення існуючого рівня охорони праці, попередження випадків виробничого травматизму, професійних захворювань, аварій і пожеж.

Наведені чинники створюють як моральний, так і матеріальний стимул для роботодавця системно опікуватися питаннями безпеки. Тому що жоден зовнішній тиск не матиме тривалого ефекту, якщо керівництво не забезпечить гідні умови праці та атмосферу взаємної довіри, поваги до закону й неписаних правил порядності у взаєминах з персоналом.

Отже, питання охорони праці складне і досить відповідальне. Адже за ним життя і здоров'я людей, які своєю працею створюють для держави матеріальні блага.

ВИСНОВКИ

Під час виконання бакалаврської кваліфікаційної роботи було проведено аналіз предметної області та визначено особливості в управлінні та проведенні онлайн-опитувань, що дало змогу визначити ключові сценарії користувачів. Аналіз існуючих аналогів дозволив зрозуміти, яких саме можливостей сьогодні бракує більшість сервісів або перенавантажені зайвими функціями, або ж обмежені в аналітичних інструментах. Саме ця інформація стала початком для формулювання роботи платформи, де кожен етап роботи з опитуваннями виконується просто й без особливих зусиль.

У ході вибору технологічного стеку зроблено акцент на перевірені рішення, що забезпечують злагоджену роботу в різноманітних середовищах. Використання HTML і CSS дозволило створити чітку структуру для всіх сторінок, а застосування адаптивності гарантує комфортне відображення на екранах різної роздільної здатності. JavaScript задав логіку інтерфейсу, підтвердивши свою спроможність миттєво реагувати на дії користувача. PHP-скрипти, реалізовані в рамках обраного фреймворка, забезпечили надійну обробку запитів із дотриманням принципів безпеки, а MySQL із правильною нормалізацією даних і побудованими індексами став гарантом того, що базу можна розгортати як на локальному сервері, так і в хмарному середовищі.

Проєктування архітектури, яке включало побудову діаграм варіантів використання з чітко описаними варіантами взаємодії та діаграм класів з обгрунтованими зв’язками між об’єктами. За допомогою ER-діаграм вдалося глибоко продумати взаємозв’язки між, а роль зовнішніх і первинних ключів допомогла забезпечити цілісність даних і уникнути дублювання.

Реалізація клієнтської частини за методикою прогресивного покрокового налаштування компонентів дозволила швидко отримувати перші прототипи інтерфейсу. Запровадження AJAX-запитів мінімізувало затримки, тоді як серверна

логіка PHP відповідала за перевірку прав доступу, валідацію вхідних даних та формування об’єктів.

Проведено успішні тестування, які продемонстрували працездатність та коректність функціоналу веб-застосунку відповідно до поставлених вимог.

У результаті платформа набула таких властивостей, як гнучкість налаштувань, масштабованість і безпека. Гнучкість полягає у підтримці різних типів питань – від простого вибору одного варіанта до відкритих текстових полів та рейтингів – з можливістю комбінувати їх у будь-якому порядку. Масштабованість досягається завдяки ретельному розподілу логіки між клієнтською й серверною частинами, а також оптимізованим запитам до бази даних, що гарантує швидке завантаження сторінок навіть при великій кількості збережених відповідей. Безпека забезпечується через багаторівневу систему авторизації, шифрування конфіденційних даних та регулярні резервні копії, що дозволяє захистити інформацію як самих авторів опитувань, так і їхніх респондентів.

Таким чином, реалізовано не просто веб-застосунок, а комплексне середовище для управління опитуваннями, яке гармонійно поєднує зручний інтерфейс і потужні інструменти аналітики. З огляду на досягнуті результати, можна стверджувати, що обрана методологія й технологічний стек повністю відповідають поставленим цілям, а сам проєкт має всі передумови для подальшої комерціалізації та масштабування в умовах сучасних інформаційних запитів. Серед напрямків розвитку варто виокремити інтеграцію з зовнішніми аналітичними сервісами, розширення можливостей кастомізації результатів, а також впровадження модулів мультимовності та адаптивної локалізації. Крім того, наступними кроками можуть стати розробка мобільних додатків-оболонок з використанням існуючого API та впровадження механізмів машинного навчання для автоматичного аналізу відкритих відповідей.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Визначення поняття веб-застосунку[Електронний ресурс] – Режим доступу до ресурсу: https://foxminded.ua/chym-vidrizniaietsia-sait-vid-veb-dodatku/ (дата звернення 01.05.2025)

2. Основи розробки веб-додатків : посібник / П’юрівал С. – Київ: Україна ТД, 2020. – 320 с.

3. Офіційна сторінка Google Forms[Електронний ресурс] – Режим доступу до ресурсу: https://workspace.google.com/products/forms/(дата звернення 05.05.2025)

4. Офіційна сторінка SurveyMonkey[Електронний ресурс] – Режим доступу до ресурсу: https://www.surveymonkey.com/ (дата звернення 05.05.2025)

5. Офіційна сторінка TypeForm[Електронний ресурс] – Режим доступу до ресурсу: https://www.typeform.com/ (дата звернення 05.05.2025)

6. Офіційна сторінка Microsoft Forms[Електронний ресурс] – Режим доступу до ресурсу: https://forms.office.com/Pages (дата звернення 05.05.2025)

7. Розробка діаграм з використанням IBM Rational Architect Designer [Електронний ресурс] – Режим доступу до ресурсу : https://www.ibm.com/products/rational-software-architect-designer (дата звернення 08.05.2025)

8. Про методику побудови діаграм варінтів використання[Електронний ресурс] – Режим доступу до ресурсу: https://www.maxzosim.com/use-cases-and-scenarios/ (дата звернення 08.05.2025)

9. Основні принципи проєктування системи[Електронний ресурс] – Режим доступу до ресурсу: https://blog.ithillel.ua/articles/web-application-architecture (дата звернення 08.05.2025)

10. Про методику побудови баз даних [Електронний ресурс] – Режим доступу до ресурсу: https://foxminded.ua/skhemy-bazy-danyh/ (дата звернення 09.05.2025)

11. Методика побудови діаграм класів[Електронний ресурс] – Режим доступу до ресурсу: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/ (дата звернення 09.05.2025)

12. Побудова роботи системи[Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua/forums/topic/40575/ (дата звернення 09.05.2025)

13. Використання баз даних для серверної розробки[Електронний ресурс]– Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side (дата звернення 12.05.2025)

14. Програмування мовою PHP : книга / Васильєв О. – Київ: Ліра-К, 2022. – 368 с.

15. HTML і CSS. Розробка та дизайн веб-сайтів : книга / пер. з англ. / Дакетт Дж. – Київ: Yakaboo, 2023. – 512 с.

16. Head First. HTML і CSS : книга / пер. з англ. / Фримен Е., Робсон Е. – Харків: Ранок, 2021. – 720 с.

17. Про особливості фреймворку Bootstrap[Електронний ресурс] – Режим доступу до ресурсу: https://getbootstrap.com/ (дата звернення 15.05.2025)

18. Використання JavaScript у розробці Frontend частини[Електронний ресурс] – Режим доступу до ресурсу: https://uk.javascript.info/ (дата звернення 15.05.2025)

19. Про використання AJAX методів та запити до API [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@francesco.saviano87/how-to-use-javascript-fetch-api-for-ajax-requests-858a7a7b729b (дата звернення 15.05.2025)

20. Основні процеси тестування веб-застосунків для перевірки справності їх роботи[Електронний ресурс] – Режим доступу до ресурсу: https://qalight.ua/baza-znaniy/testuvannya-veb-proektiv-osnovni-etapi-ta-poradi/ (дата звернення 18.05.2025)

21. Дистанційний курс «Кваліфікаційні роботи бакалаврів» сайту дистанційного навчання ТНТУ [Електронний ресурс]. – Режим доступу: URL: https://dl.tntu.edu.ua/bounce.php?course=5329 (дата звернення 20.05.2025)

22. Information System for Design of Thin Multilayer Film Processes Parameters Management based on Diffusion Mykhaylo Petryk, Vitalii Chyzh, Halyna Tsupryk, Oksana Petryk, ITTAP'2024: 4th International Workshop on Information Technologies: Theoretical and Applied Problems, October 23-25, 2024, Ternopil, Ukraine, Opole, Poland, 2024, pp. 486-493 (Scopus) – Режим доступу до ресурсу: https://ceur-ws.org/Vol-3896/. (ТНТУ)

23. Methods of constructing algorithms for comparative test statistical verification of mathematical models of bioobject responses to low-intensity stimuli / Bohdan Yavorskyy, Evhenia Yavorska, Halyna Tsupryk, Roman Kinash // Scientific Journal of TNTU. — Tern.: TNTU, 2023. — Vol 112. — No 4. — P. 82–90. (фахове видання України)

24. Методичні вказівки до виконання дипломної роботи освітнього рівня — бакалавр студентами усіх форм навчання для напряму підготовки 121 – Інженерія програмного забезпечення / Укладачі : Петрик М.Р., Михалик Д.М., Кінах Я.І., Гладьо С.В., Цуприк Г.Б. – Тернопіль: Вид-во ТНТУ імені Івана Пулюя, 2016 – 28 с.

25. Є.Б. Яворська, Р.В. Кінаш, Г.Б. Цуприк, В.І. Николайчук, Проблеми виявлення біосигналів у медичних інформаційних системах/ Є.Б. Яворська, Р.В. Кінаш, Г.Б. Цуприк, В.І. Николайчук// IV Міжнародна науково-практична конференція «Інформаційні системи та технології в медицині» (ІСМ–2021)[Текст]: зб. наук. пр. – Харків: Нац. аерокосм. ун-т ім. М.Є. Жуковського «Харків. авіац. ін-т», 2021. – 25-26 с.

26. Н. Доскоч Г.Б. Цуприк, Розробка системи забезпечення безпечного обміну інформацією з використаннм технологій веб програмування. Збірник тез доповідей X науково-технічної конференції «Інформаційні моделі, системи та технології» – Тернопіль 7 грудня 2022 року. – Т. : ТНТУ, 2022 – 114 с.

27. О. Остапчук, Г. Цуприк, Технічні особливості взаємодії між клієнтом та сервером у реальному часі. Збірник тез доповідей X науково-технічної

конференції «Інформаційні моделі, системи та технології» – Тернопіль 7 грудня 2022 року. – Т. : ТНТУ, 2022 – 124 с.

28. Соціально-політичні конфлікти – теорія та практика : підручник / І. О. Кресіна ; Київ: «Наукова думка», 2020. – 312 с.

29. Безпека життєдіяльності та охорона праці: підручник : у 2 ч. / Я. О. Сєріков, Л. Ф. Коженевські, М. В. Хворост ; Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова [та ін.]. – Харків : ХНУМГ ім. О. М. Бекетова ; Краків : ЄАС, 2021.

30. Охорона праці в галузі: підручник / І. М. Герасименко ; Київ: Ліра-К, 2022. — 464 с.

31. Про право на безпечні та здорові умови праці. [Електронний ресурс] Конституція України – 2020 – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/254к/96-вр#Text (дата звернення 02.06.2025)

32. Про державну політику в галузі охорони праці. [Електронний ресурс] Закон України – 2025 – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/main/2694-12#Text (дата звернення 02.06.2025)

33. Про регулювання охорони праці у колективному договорі, угоді. [Електронний ресурс] Закон України – 2025 – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/main/2694-12#Text (дата звернення 02.06.2025)

ДОДАТКИ

ДОДАТОК А. Тези для VIII Міжнародної студентської науково-технічної конференції

Міністерство освіти і науки України
Тернопільський національний технічний університет
імені Івана Пулюя
Маріборський університет (Словенія)
Технічний університет в Кошице (Словаччина)
Каунаський технологічний університет (Литва)
Львівський національний університет
імені Івана Франка,
Гірничо-металургійна академія ім. Станіслава Сташиця (Польща)
Луцький національний технічний університет,
Чернівецький національний університет
імені Юрія Федьковича,
Вроцлавський економічний університет (Польща)
Університет технологій та економіки
імені Хелени Ходковської (Польща)
Донбаська державна машинобудівна академія

Студентське наукове товариство

VIII МІЖНАРОДНА

студентська науково - технічна конференція

"ПРИРОДНИЧІ ТА ГУМАНІТАРНІ НАУКИ.

АКТУАЛЬНІ ПИТАННЯ"

24-25 квітня 2025 р.

(збірник тез конференції)

Тернопіль 2025

УДК 004.738.5
Лунак О. –ст. гр. СП-42
Тернопільський національний технічний університет імені Івана Пулюя

РОЗРОБКА ВЕБ-ЗАСТОСУНКУ ДЛЯ ЗБОРУ ТА ОПРАЦЮВАННЯ ДАНИХ З ВИКОРИСТАННЯМ СУЧАСНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Науковий керівник: к.т.н., доцент Цуприк Г.Б.

Lunak O.
Ternopil Ivan Puluj National Technical University

DEVELOPMENT OF A WEB APPLICATION FOR DATA COLLECTION AND PROCESSING USING MODERN INFORMATION TECHNOLOGIES

Supervisor: Tsupryk H.

Ключові слова: онлайн-опитування, веб-застосунок, респондент, фронтенд, бекенд
Keywords: online survey, web application, respondent, frontend, backend

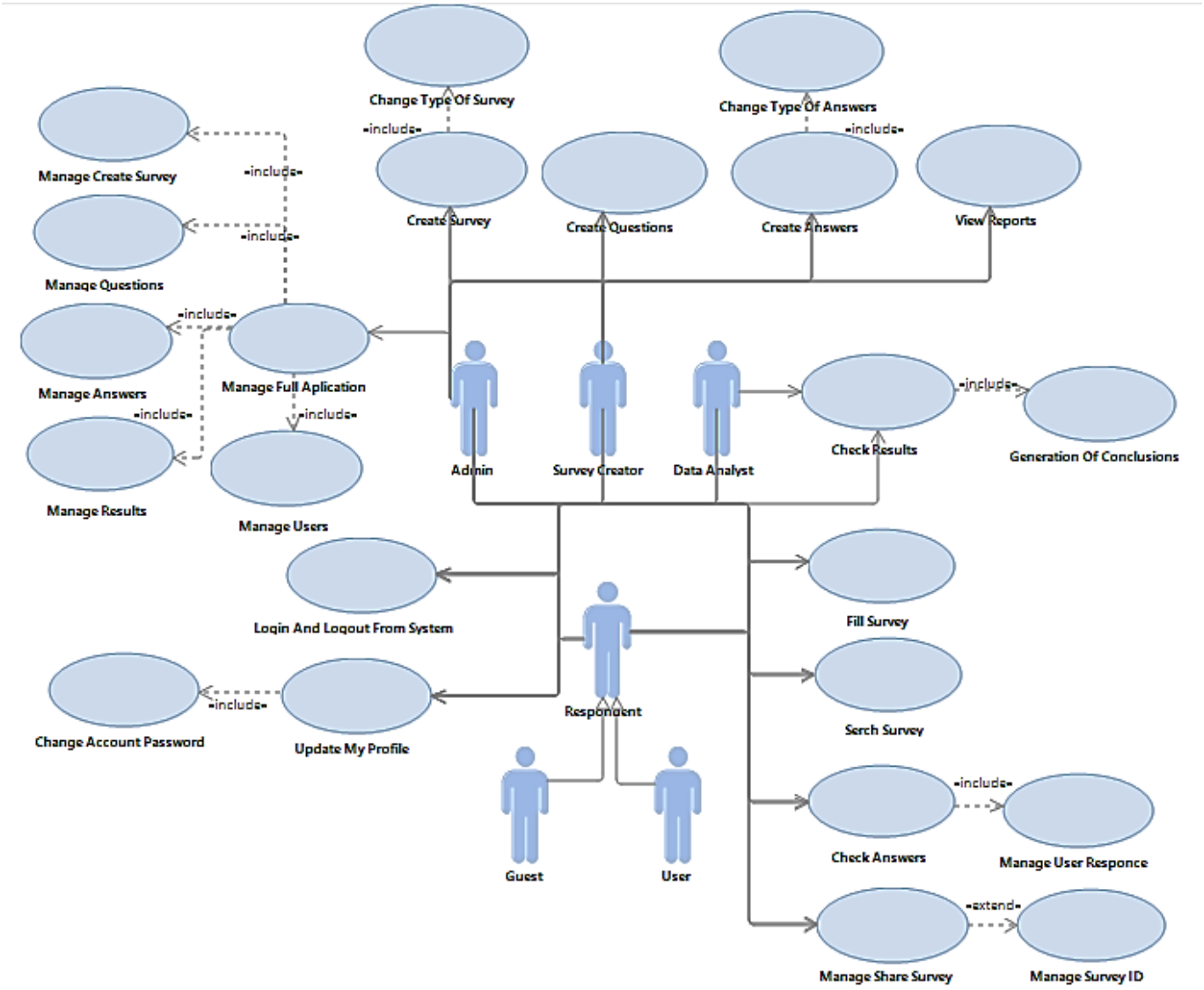
У статті розглядається веб-застосунок, призначений для створення, проведення та аналізу онлайн-опитувань. Цей застосунок створено з метою автоматизації процесу збору даних, забезпечення зручності взаємодії між адміністраторами, які формують опитування, та користувачами, які беруть у них участь. Адміністративна частина веб-застосунку надає змогу створювати опитування з довільною кількістю різних видів питань, редагувати існуючі та переглядати результати респондентів. Користувачі можуть проходити опитування у режимі реального часу, що підвищує якість збору даних та зменшує витрати часу на обробку результатів.

Однією з важливих переваг є функція аналізу результатів. Система генерує статистичні звіти та візуалізує дані у вигляді діаграм та графіків, що спрощує сприйняття результатів і дає змогу швидко робити висновки. Веб-застосунок має адаптивний дизайн, тому його зручно використовувати як з комп'ютера, так і з мобільного пристрою, що розширює аудиторію потенційних респондентів. Завдяки підтримці авторизації можна проводити як публічні опитування, доступні кожному, так і приватні опитування лише для зареєстрованих користувачів або за посиланнями.

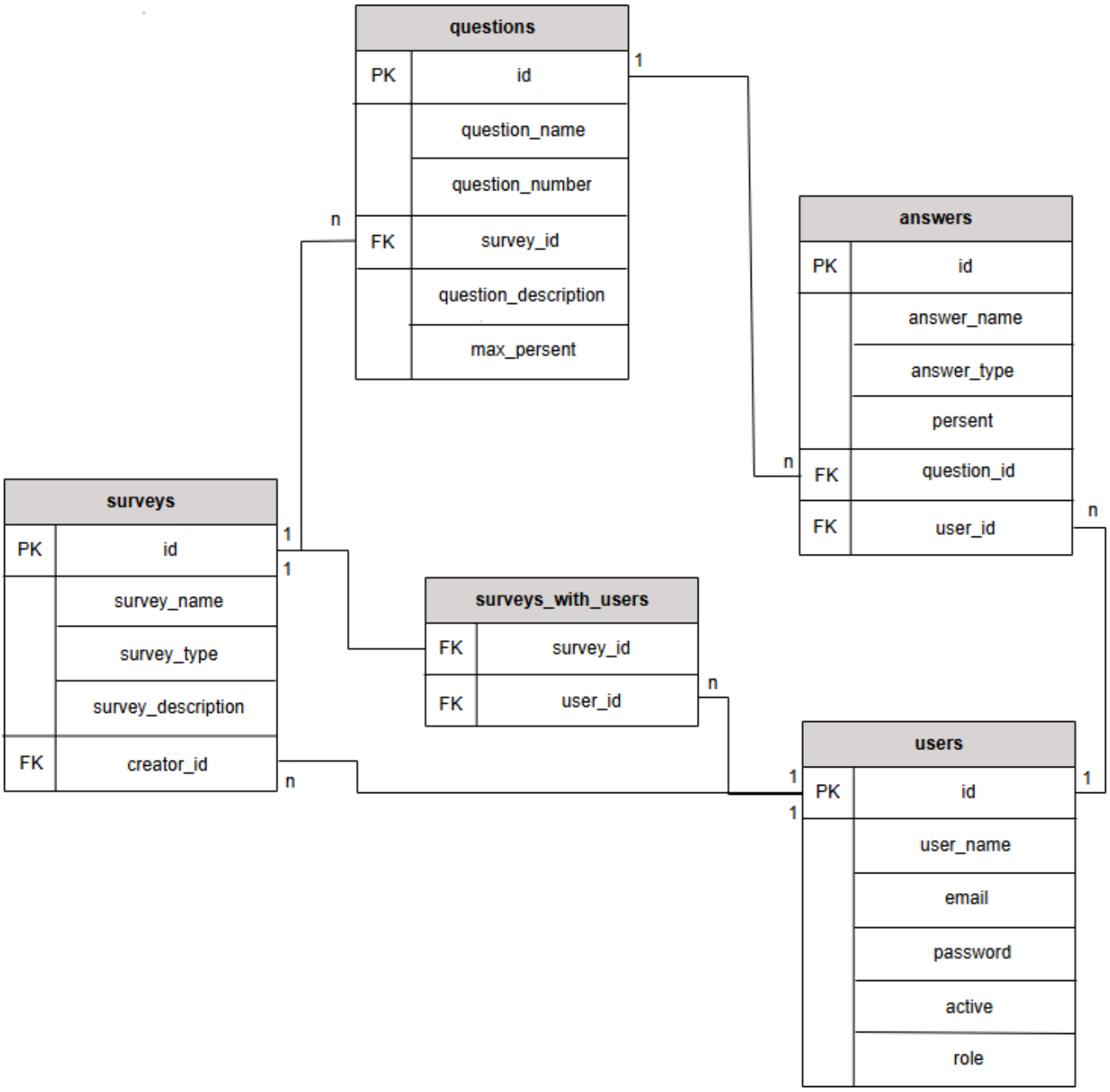
До недоліків можна віднести залежність від інтернет-з'єднання та обмеження щодо використання складних аналітичних методів, які потребують додаткового програмного забезпечення. Однак у більшості типових завдань, пов'язаних зі збором та обробкою даних, функціоналу системи цілком достатньо. Фронтенд-застосунку реалізовано з використанням HTML, CSS і JavaScript, що забезпечує сучасний, адаптивний та інтуїтивно зрозумілий інтерфейс. Серверна частина побудована на мові програмування PHP, а для зберігання всієї інформації використовується реляційна база даних MySQL. Таке технологічне рішення дає змогу гнучко масштабувати застосунок, а також забезпечити його стабільну роботу при зростанні кількості користувачів.

Даний веб-застосунок є актуальним у сучасному інформаційному суспільстві. Він може застосовуватись в освітніх установах для оцінювання знань студентів, у бізнес-середовищі для аналізу клієнтської думки, внутрішніх опитувань робітників або у соціальних дослідженнях, де важливе швидке отримання великої кількості відповідей. У контексті інформаційних технологій дана система демонструє приклад ефективної реалізації веб-сервісу, який об'єднує сучасні інструменти фронтенд- і бекенд-розробки, забезпечуючи при цьому масштабованість, безпеку, зручність та універсальність використання.

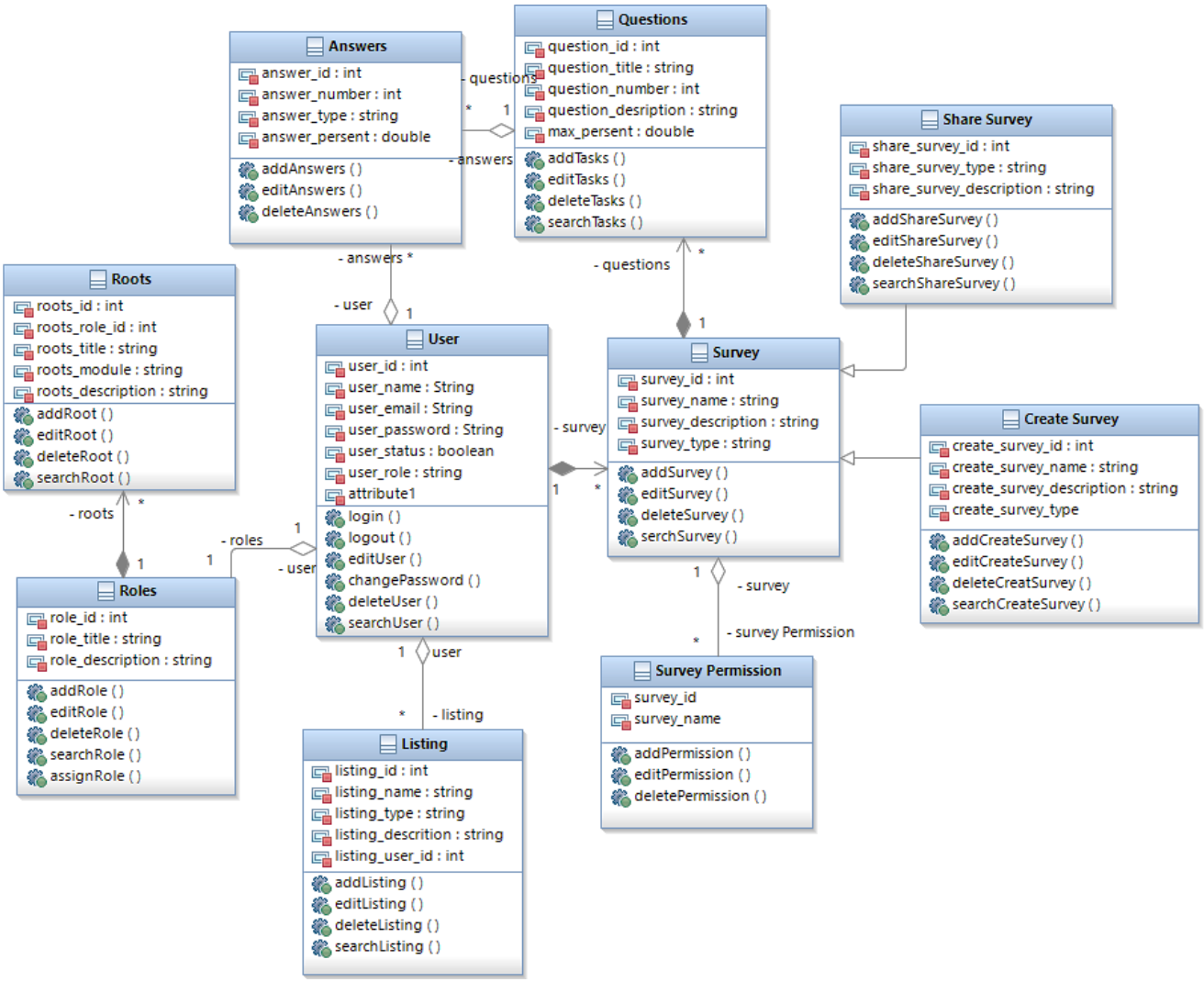
ДОДАТОК Б. Діаграми та рисунки



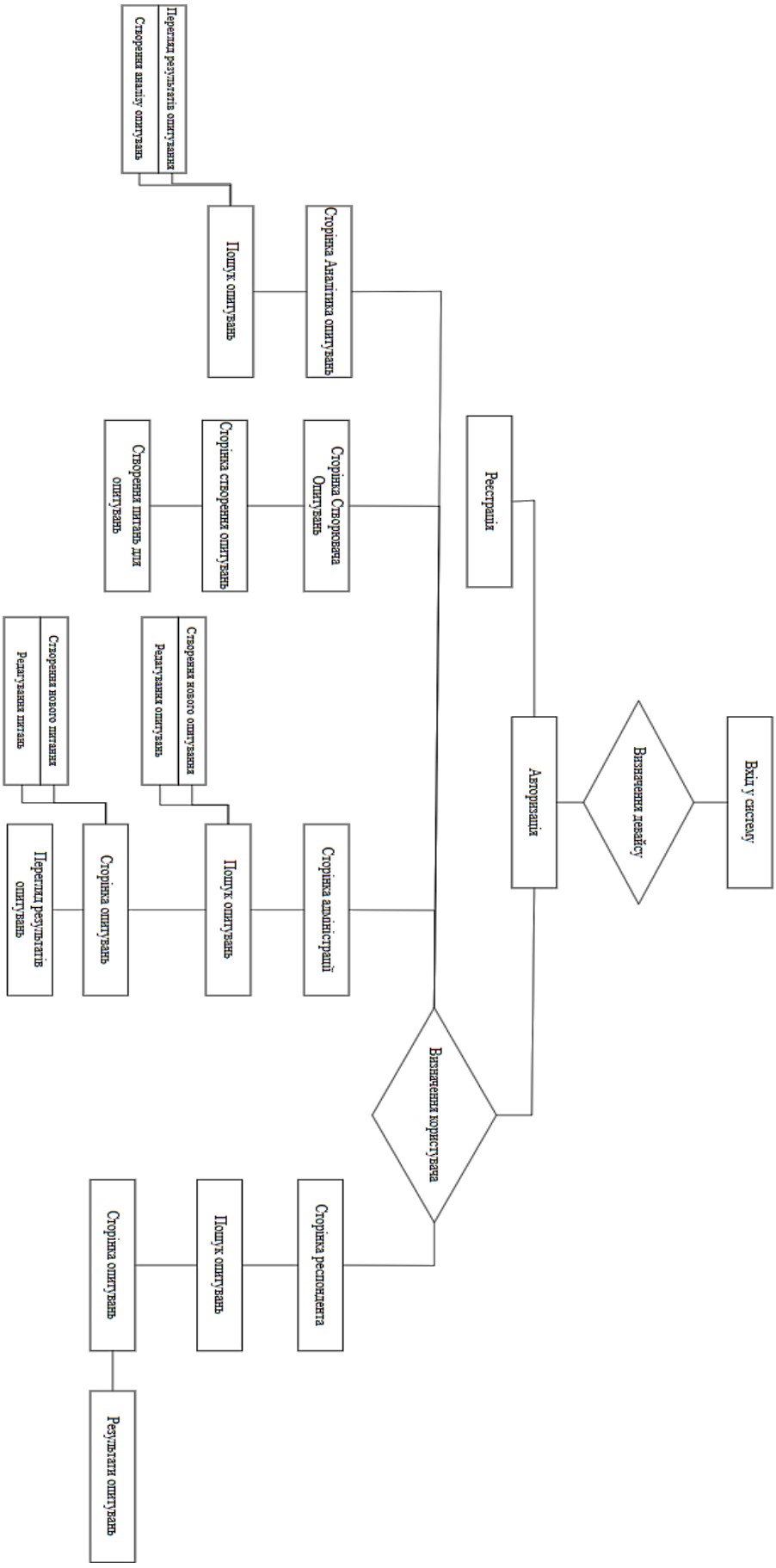
Діаграма варіантів використання



ER-діаграма бази даних



Діаграма класів



Діаграма структури веб-застосунку

ДОДАТОК В. Лістинг коду

main.html:

```
    </header>
    <div class="carousel-wrapper">
    <div class="container mt-4">
        <div  id="carouselExampleCaptions"  class="carousel  slide"
data-bs-ride="carousel">
            <div class="carousel-indicators">
                <button            type="button"            data-bs-
target="#carouselExampleCaptions" data-bs-slide-to="0" class="active"
aria-current="true" aria-label="Slide 1"></button>
                <button            type="button"            data-bs-
target="#carouselExampleCaptions"     data-bs-slide-to="1"     aria-
label="Slide 2"></button>
                <button            type="button"            data-bs-
target="#carouselExampleCaptions"     data-bs-slide-to="2"     aria-
label="Slide 3"></button>
            </div>
            <div class="carousel-inner">
                <div class="carousel-item active">
                    <img src="images/image_1.avif" class="d-block w-
100" alt="...">
                    <div class="carousel-caption d-none d-md-block">
                        <h5>Онлайн-Опитування</h5>
                        <p>Швидке створення та проведення вашого
опитування</p>
                    </div>
                </div>
                <div class="carousel-item">
                    <img src="images/image_2.avif" class="d-block w-
100" alt="...">
                    <div class="carousel-caption d-none d-md-block">
```

```
                        <h5>Різноманітність Тем</h5>
                        <p>Створення опитувань на будь-яку вашу тему
з великим вибором відповідей.</p>
                    </div>
                </div>
                <div class="carousel-item">
                    <img src="images/image_3.avif" class="d-block w-
100" alt="...">
                    <div class="carousel-caption d-none d-md-block">
                        <h5>Аналіз Результатів</h5>
                        <p>Виведення Ваших результатів відповідей
після завершення опитування</p>
                    </div>
                </div>
            </div>
            <button   class="carousel-control-prev"   type="button"
data-bs-target="#carouselExampleCaptions" data-bs-slide="prev">
                <span    class="carousel-control-prev-icon"    aria-
hidden="true"></span>
                <span class="visually-hidden">Попередній</span>
            </button>
            <button   class="carousel-control-next"   type="button"
data-bs-target="#carouselExampleCaptions" data-bs-slide="next">
                <span    class="carousel-control-next-icon"    aria-
hidden="true"></span>
                <span class="visually-hidden">Наступний</span>
            </button>
        </div>
    </div>
    </div>
    <section class="info-section">
        <h2>Як працює наш сервіс?</h2>
        <div class="info-cards">
          <div class="info-card">
            <img src="images/icon_1.png" alt="Іконка опитування" />
```

```
            <h3><b>Створюйте опитування</b></h3>
            <p>Легко створюйте питання та варіанти відповідей для будь-яких тем.</p>
          </div>
          <div class="info-card">
            <img src="images/icon_2.png" alt="Іконка поширення" />
            <h3><b>Поширюйте серед людей</b></h3>
            <p>Поділіться опитуванням з друзями або розмістіть у соцмережах.</p>
          </div>
          <div class="info-card">
            <img src="images/icon_3.png" alt="Іконка аналітики" />
            <h3><b>Аналізуйте відповіді</b></h3>
            <p>Отримуйте статистику та візуалізацію результатів у режимі реального часу.</p>
          </div>
        </div>
      </section>
    <main>
        <section class="hero">
            <div class="hero-text">
                <h1><b>Ласкаво просимо до онлайн-опитувань</b></h1>
                <p>Створюйте опитування та аналізуйте результати легко!</p>
                <a href="#create" class="cta-button">Створити опитування</a>
            </div>
            <div class="hero-image">
                <img src="images/image_4.avif" alt="Ілюстрація для опитування">
            </div>
        </section>
    </main>
```

styles.css:

```
header {
    background: #1f2937;
    padding: 12px 50px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    border-bottom: 3px solid #374151;
    position: sticky;
    top: 0;
    z-index: 1000;
}
.logo {
    font-size: 26px;
    font-weight: 700;
    color: #10b981;
    letter-spacing: 1px;
}
nav ul {
    list-style: none;
    padding: 0;
    margin: 0;
    display: flex;
    align-items: center;
}
nav ul li {
    margin: 0 10px;
    display: flex;
    align-items: center;
    gap: 5px;
}
nav ul li a {
```

```
            color: white;
            text-decoration: none;
            font-size: 16px;
            padding: 10px 16px;
            border-radius: 5px;
            transition: all 0.3s ease;
            display: flex;
            align-items: center;
            gap: 6px;
        }
        nav ul li a:hover {
            background: rgba(255, 255, 255, 0.2);
            transform: scale(1.05);
        }
        .btn {
            background: #10b981;
            color: white;
            padding: 10px 18px;
            border-radius: 6px;
            font-weight: bold;
            text-decoration: none;
            transition: all 0.3s ease;
        }
        .btn:hover {
            background: #0ea86c;
        }


        /* Адаптивність */
    @media (max-width: 768px) {
        header {
            flex-direction: column;
            text-align: center;
        }
        nav ul {
```

```
        flex-direction: column;
        margin-top: 10px;
    }
    nav ul li {
        margin: 5px 0;
    }
    main {
        padding: 30px 16px;
    }
    .about-content {
        padding: 30px 20px;
    }
    .about-content h1 {
        font-size: 24px;
    }
    .about-content p, .about-content li {
        font-size: 15px;
    }
    .cta-text {
        font-size: 16px;
    }
}
@media (max-width: 480px) {
    .about-content {
        padding: 20px 16px;
    }
    .about-content h1 {
        font-size: 22px;
    }
    .about-content p, .about-content li {
        font-size: 14px;
    }
```

scripts.js:

```
    function applyTheme(theme) {
        if (theme === 'light-theme') {
            body.classList.add('light-theme');
            if (themeToggle) themeToggle.checked = true;
        } else {
            body.classList.remove('light-theme'); // Default is dark
            if (themeToggle) themeToggle.checked = false;
        }
    }
    if (currentTheme) {
        applyTheme(currentTheme);
    } else if (window.matchMedia && window.matchMedia('(prefers-color-
scheme: light)').matches) {
        applyTheme('light-theme');
        localStorage.setItem('theme', 'light-theme');
    } else {
        applyTheme('dark-theme');
    }
    if (themeToggle) {
        themeToggle.addEventListener('change', function() {
            const selectedTheme = this.checked ? 'light-theme' :
'dark-theme';
            applyTheme(selectedTheme);
            localStorage.setItem('theme', selectedTheme);
        });
    }
    const navLinks = document.querySelectorAll('header nav ul li a');
    let currentPath = window.location.pathname;
    if (currentPath.endsWith('/index.html')) {
        currentPath = currentPath.substring(0, currentPath.length -
'index.html'.length);
```

```
    }
    if (currentPath === '') currentPath = '/';
    navLinks.forEach(link => {
        let linkPath = new URL(link.href).pathname;
        if (linkPath.endsWith('/index.html')) {
            linkPath = linkPath.substring(0, linkPath.length -
'index.html'.length);
        }
        if (linkPath === '') linkPath = '/';

        if (linkPath === currentPath) {
            link.classList.add('active-link');
        } else {
            link.classList.remove('active-link');
        }
    });
    const yearSpan = document.getElementById('currentYear');
    if (yearSpan) {
        yearSpan.textContent = new Date().getFullYear();
    }
});
```

init.php:

```
require_once __DIR__ . '/vendor/autoload.php';
require_once __DIR__ . '/database.php';
require_once __DIR__ . '/../app/models/user.php';
require_once __DIR__ . '/../app/models/survey.php';
require_once __DIR__ . '/../app/models/question.php';
require_once __DIR__ . '/../app/models/answer.php';
require_once __DIR__ . '/../app/models/role.php';
try {
```

```
        if (!isset($pdo) || !($pdo instanceof PDO)) {
            throw new Exception("Не вдалося створити об'єкт PDO з
database.php");
        }
        User::setDatabase($pdo);
        Survey::setDatabase($pdo);
        Question::setDatabase($pdo);
        Answer::setDatabase($pdo);
    } catch (Exception $e) {
        error_log("FATAL: Could not initialize database connection
for models: " . $e->getMessage());
        $isApiRequest = (isset($_SERVER['HTTP_X_REQUESTED_WITH']) &&
strtolower($_SERVER['HTTP_X_REQUESTED_WITH']) === 'xmlhttprequest')
|| (strpos($_GET['r'] ?? '', 'api/') === 0);
        if ($isApiRequest) {
            header('Content-Type: application/json');
            http_response_code(503); // Service Unavailable
            echo json_encode(['success' => false, 'message' =>
'Сервер бази даних тимчасово недоступний. Спробуйте пізніше. Помилка
ініціалізації.']);
            exit;
        } else {
            die("<h1>Помилка Сервера</h1><p>Не вдалося налаштувати
підключення до бази даних для моделей. Будь ласка, спробуйте
пізніше.</p><p><small>" . $e->getMessage() . "</small></p>");
        }
    }
```

migrate.php:

```
require_once 'vendor/autoload.php';
try {
```

```
        $dotenv = Dotenv\Dotenv::createImmutable(__DIR__);
        $dotenv->load();
    } catch (Dotenv\Exception\InvalidPathException $e) {
        error_log("Could not load .env file for migrations: " . $e-
>getMessage());
        die("Configuration error for migrations: .env file not found
or path is incorrect. Make sure .env is in the project root.");
    }
    $host = $_ENV['DB_HOST'] ?? 'localhost';
    $username = $_ENV['DB_USERNAME'] ?? 'root';
    $password = $_ENV['DB_PASSWORD'] ?? '';
    $database = $_ENV['DB_DATABASE'] ?? null;
    $port = $_ENV['DB_PORT'] ?? 3306;
    if (!$database) {
        die("DB_DATABASE not set in .env file. Cannot run
migrations.");
    }
    mysqli_report(MYSQLI_REPORT_ERROR | MYSQLI_REPORT_STRICT);
    try {
        $mysqli = new mysqli($host, $username, $password, $database,
(int)$port);
        $mysqli->set_charset("utf8mb4");
    } catch (mysqli_sql_exception $e) {
        die("Migration connection failed: " . $e->getMessage());
    }
    echo "Підключено до бази даних: " . $database . PHP_EOL;
    $createMigrationsTableSQL = "
    CREATE TABLE IF NOT EXISTS migrations (
        id INT AUTO_INCREMENT PRIMARY KEY,
        migration VARCHAR(255) NOT NULL UNIQUE,
        batch INT NOT NULL,
        applied_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    );";
    if (!$mysqli->query($createMigrationsTableSQL)) {
```

```
        die("Помилка створення таблиці міграцій: " . $mysqli->error
. PHP_EOL);
    }
    echo "Таблиця 'migrations' перевірена/створена." . PHP_EOL;
    $appliedMigrations = [];
    $result = $mysqli->query("SELECT migration FROM migrations");
    if ($result) {
        while ($row = $result->fetch_assoc()) {
            $appliedMigrations[] = $row['migration'];
        }
        $result->free();
    }
    $batchResult = $mysqli->query("SELECT MAX(batch) as max_batch
FROM migrations");
    $currentBatch = $batchResult ? ($batchResult-
>fetch_assoc()['max_batch'] ?? 0) + 1 : 1;
    $migrationsPath = __DIR__ . '/database/migrations/';
    $migrationFiles = glob($migrationsPath . '*.php');
    sort($migrationFiles);
    $newMigrationsApplied = 0;
    foreach ($migrationFiles as $file) {
        $migrationName = basename($file);
        if (!in_array($migrationName, $appliedMigrations)) {
            echo "Застосування міграції: " . $migrationName . "... ";
            try {
                $mysqli->begin_transaction();
                require $file;
                $stmt = $mysqli->prepare("INSERT INTO migrations
(migration, batch) VALUES (?, ?)");
                if (!$stmt) {
                    throw new Exception("Помилка підготовки запиту
для запису міграції: " . $mysqli->error);
                }
                $stmt->bind_param("si", $migrationName,
$currentBatch);
```

```
                if (!$stmt->execute()) {
                    throw new Exception("Помилка запису міграції в
таблицю: " . $stmt->error);
                }
                $stmt->close();
                $mysqli->commit();
                echo "Успішно." . PHP_EOL;
                $newMigrationsApplied++;
            } catch (Exception $e) {
                $mysqli->rollback();
                echo "Помилка: " . $e->getMessage() . PHP_EOL;
            }
        }
    }
    if ($newMigrationsApplied > 0) {
        echo "Виконано " . $newMigrationsApplied . " нових
міграцій." . PHP_EOL;
    } else {
        echo "Немає нових міграцій для застосування." . PHP_EOL;
    }
    $mysqli->close();
    echo "Процес міграції завершено." . PHP_EOL;
```

router.php

```
switch ($mainResource) {
    case 'api':
        handleApiRoutes($subResource, $action, $id, $method);
        break;
    case 'login':
        require_once __DIR__ .
'/../app/http/controllers/auth_controller.php';
```

```
        $controller = new AuthController();
       /*                echo                json_encode($controller-
>login(json_decode(file_get_contents("php://input"), true)));*/
        break;
    case 'register':
        require_once                    __DIR__                    .
'/../app/http/controllers/auth_controller.php';
        $controller = new AuthController();
        /*echo                              json_encode($controller-
>register(json_decode(file_get_contents("php://input"), true)));*/
        break;
    case 'logout':
        require_once                    __DIR__                    .
'/../app/http/controllers/auth_controller.php';
        $controller = new AuthController();
        echo json_encode($controller->logout());
        break;
    case 'admin_panel.html':
    case 'admin':
        serveHtmlFile('admin/admin_panel.html');
        break;
    case 'manage_users.html':
        serveHtmlFile('admin/manage_users.html');
        break;
    case 'manage_roles.html':
        serveHtmlFile('admin/manage_roles.html');
        break;
    case 'manage_surveys.html':
        serveHtmlFile('admin/manage_surveys.html');
        break;
    case 'create_analytics.html':
        serveHtmlFile('admin/create_analytics.html');
        break;
    case 'listing.html': // Логи
        serveHtmlFile('admin/listing.html');
```

```
        break;
    case '':
    case 'index.html':
        serveHtmlFile('index.html');
        break;
    case 'about.html':
        serveHtmlFile('about.html');
        break;
    case 'surveys.html':
        serveHtmlFile('surveys.html');
        break;
    case 'serch_survey.html':
        serveHtmlFile('serch_survey.html');
        break;
    case 'create_survey.html':
        serveHtmlFile('create_survey.html');
        break;
    default:
        $publicFilePath = __DIR__ . '/' . $routePath;
        if (file_exists($publicFilePath) && is_file($publicFilePath)
&& $routePath !== 'index.php' && $routePath !== 'router.php') {
            serveStaticFile($publicFilePath);
        } else {
            http_response_code(404);
            if ($isApiRequest) {
                echo json_encode(['success' => false, 'message' =>
'API маршрут не знайдено']);
            } else {
                echo "<h1>404 Сторінку не знайдено</h1><p>Маршрут:
{$routePath}</p>";
            }
        }
        break;
}
exit;
```

ДОДАТОК Г. Диск