МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ФАХОВИЙ БІЗНЕС-КОЛЕДЖ
Циклова комісія (кафедра) комп’ютерної інженерії та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему

АВТОМАТИЗОВАНИЙ ВЕБЗАСТОСУНОК З УПРАВЛІННЯ БАЗОЮ ДАНИХ БІБЛІОТЕКИ

Виконав: студент групи 2П-21
Спеціальності 121 Інженерія програмного забезпечення
Костянтин ІВАШИН
Керівник:
Наталя ФАЛЬЧЕНКО

Черкаси 2025

ЗМІСТ

ВСТУП

У сучасних умовах цифрової трансформації автоматизація бібліотечних процесів стає необхідністю. Традиційні методи управління бібліотечною базою даних є недостатньо ефективними, оскільки вимагають значних затрат часу та людських ресурсів. Вебзастосунки, що забезпечують зручний інтерфейс для роботи з базами даних, дозволяють значно спростити процес пошуку, сортування та обліку книг. Тому розробка автоматизованого вебзастосунку для управління бібліотечною базою даних є актуальним завданням, що сприяє підвищенню ефективності роботи бібліотекарів і покращенню користувацького досвіду відвідувачів бібліотек [1].

Метою даного дослідження є розробка вебзастосунку, який дозволить автоматизувати управління бібліотечною базою даних.

Об'єктом дослідження є система автоматизації управління бібліотечними базами даних.

Предметом дослідження виступають технології проєктування, реалізації та оптимізації реляційної бази даних, а також механізми розробки графічного інтерфейсу і підключення його до бази даних.

Практична значимість роботи

Вебзастосунок може бути використаний для ведення електронного обліку бібліотечного фонду у бібліотеках будь яких закладів.

Запропоноване рішення сприятиме:

- зменшенню навантаження на персонал;
- прискоренню пошуку літератури;
- покращенню якості обслуговування читачів;
- придатне для використання у бібліотеках, де немає ІТ-фахівців.

РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ВИЗНАЧЕННЯ ВИМОГ ДО СТВОРЕННЯ БАЗИ ДАНИХ

1.1 Проблематика ручного управління бібліотекою

У багатьох бібліотеках, особливо в школах, невеликих громадах або сільських населених пунктах, і досі використовуються традиційні, паперові методи ведення обліку. Хоча вони є найдешевшими, проте на практиці створюють величезну кількість проблем, які суттєво ускладнюють щоденну роботу бібліотекаря, знижують ефективність обслуговування читачів та призводять до втрат важливої інформації [2].

Найбільш поширена проблема — це ручне ведення читацьких формулярів, журналів видачі книг, інвентарних книг, а також картотек по авторам, жанрам і темам. Усі ці записи зберігаються у паперовому вигляді, часто — в єдиному екземплярі. Через це виникають такі ситуації:

- пошук книги займає багато часу, тому що — бібліотекар змушений переглядати картотеку або кілька журналів вручну;
- у разі втрати або пошкодження журналу — інформацію майже неможливо відновити;
- неможливо швидко отримати статистику: скільки книг видано, які найпопулярніші автори, які книги найчастіше не повертають тощо.

При звернені читача до бібліотеки, бібліотекар опрацьовує звернення вручну. Відсутність електронного фільтрування та сортування за параметрами (автор, рік видання, жанр, мова, серія тощо) значно ускладнює обслуговування читачів.

У паперовій системі бібліотекар фізично не встигає перевіряти терміни повернення для кожного читача. Це призводить до:

- затримок із поверненням книг;
- відсутності нагадувань для читачів;

- втрати книжок або невчасної інвентаризації;

Ручне внесення даних завжди супроводжується ризиком помилок, таких як:

- неправильно вписане прізвище;

- зміщення дати;

- подвійна реєстрація однієї й тієї ж книги;

- втрата сторінок або їх ушкодження.

З часом це ускладнює будь-який аналіз або звітність.

У більшості бібліотек потрібно регулярно подавати статистику про діяльність наприклад: кількість книг у фонді, кількість читачів, кількість книг видано за місяць/рік, кількість не повернутих книг.

У ручному режимі формування таких звітів займає багато годин або навіть днів. До того ж ці звіти часто містять неточності.

1.1.6. Відсутність обліку користувачів бібліотеки

Без автоматизованої системи важко вести повноцінний облік читачів:

- важко визначити, хто часто не повертає книги вчасно;

- відсутній захист даних, усе записується у відкритих журналах;

- зміна персоналу може призвести до втрати частини облікової інформації.

У випадку пожежі, затоплення чи фізичного знищення паперових носіїв вся база бібліотеки буде втрачена. Жодних копій не існує. Електронні системи, навіть найпростіші, дають можливість створювати резервні копії, що критично важливо для збереження інформації.

Усі описані вище проблеми зводяться до одного: бібліотекар втрачає надзвичайно багато часу на рутинну роботу, замість того щоб займатись розвитком бібліотеки, оновленням фондів, організацією заходів чи консультуванням читачів. Часто в навчальних закладах бібліотекарі працюють на пів ставки або навіть поєднують кілька посад, тому будь-яке

навантаження критичне. У результаті, відсутність автоматизованої бібліотечної системи значно ускладнює повсякденну роботу, знижує якість обслуговування читачів, збільшує кількість помилок та створює ризик втрати цінної інформації. Саме тому розробка доступного, простого і зручного вебзастосунку з управління бібліотекою є актуальним і необхідним кроком на шляху до цифровізації бібліотечної справи.

1.2 Огляд подібних програмних продуктів

В роботі були розглянуті такі програмні продукти якими користуються бібліотекарі українських установ: koha, alma, aleph.

У сучасних умовах автоматизація бібліотечних процесів стала однією з ключових складових ефективного функціонування бібліотек. Серед різноманітних програмних рішень особливої уваги заслуговує Koha — повнофункціональна інтегрована бібліотечна система з відкритим програмним кодом, яка стрімко набула популярності в усьому світізавдяки своїй гнучкості, функціональності та активній спільноті користувачів.

Система Koha була вперше розроблена в 1999 році в Новій Зеландії компанією Katipo Communications на замовлення місцевої бібліотеки Horowhenua Library Trust. З того часу вона перетворилася на один із найвідоміших відкритих бібліотечних проєктів, який використовується у понад 5000 бібліотеках у різних країнах світу, включаючи державні, університетські, шкільні та спеціалізовані бібліотеки.

Основні особливості Koha:

1. Відкритий вихідний код (open source): система розповсюджується за ліцензією GNU GPL, що дозволяє вільно використовувати, змінювати та поширювати програмне забезпечення.

2. Повноцінний веб-інтерфейс: як для бібліотекарів, так і для читачів, що дає можливість працювати з системою через будь-який сучасний браузер.
3. Підтримка MARC21 та UNIMARC: забезпечує високу сумісність з міжнародними стандартами бібліографічного опису.
4. Модульна структура: кожен розділ функціоналу винесено в окремий модуль, що дозволяє налаштовувати систему під потреби конкретної бібліотеки.
5. Гнучкий пошук: реалізовано можливості простого та розширеного пошуку за різними критеріями, а також підтримка індексації через Elasticsearch.
6. Підтримка багатомовності: включаючи українську мову, що робить систему зручною для використання в бібліотеках України.

Однією з головних переваг Koha є її повна відкритість і незалежність від комерційних постачальників. Це дозволяє установам значно зменшити витрати на впровадження та супровід програмного забезпечення, а також забезпечує високий рівень прозорості у роботі з бібліотечними даними [3].

Головне вікно системи Koha представлено на рисунку 1.1

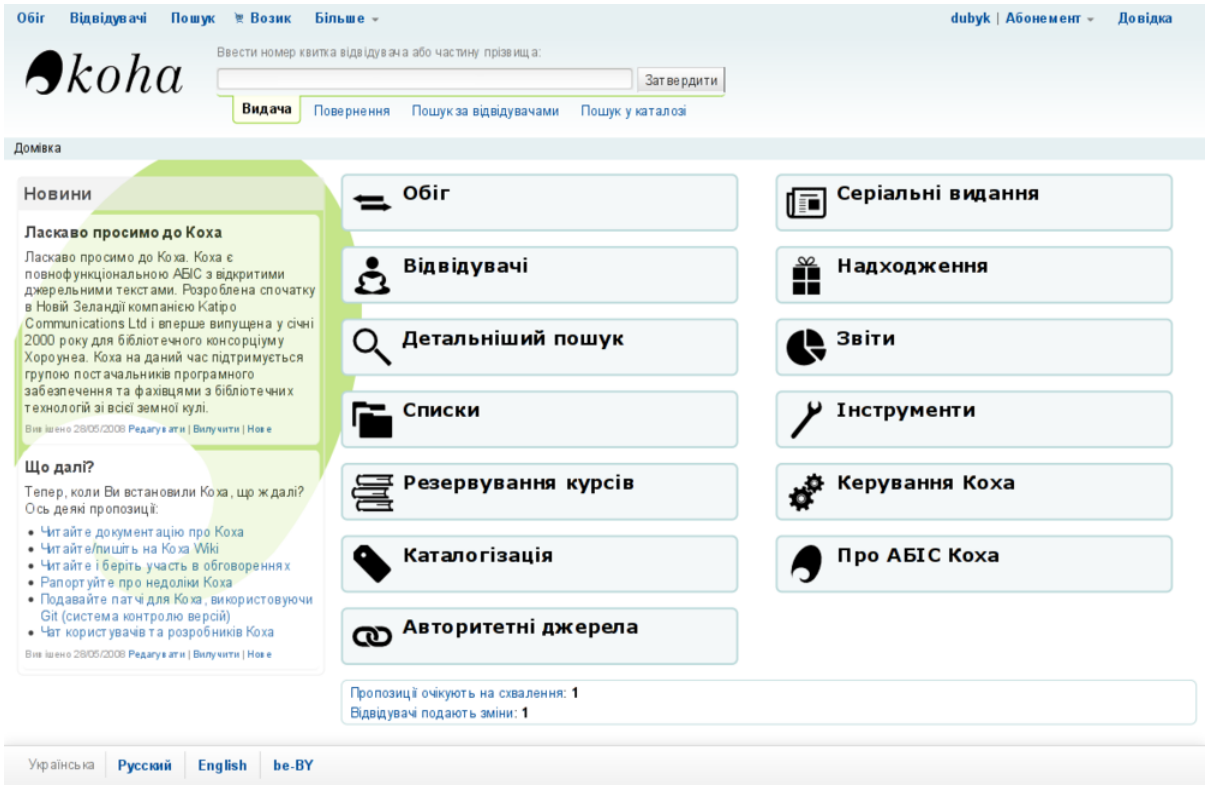

Рисунок 1.1 - Головна сторінка бібліотечної системи Koha.

В кваліфікаційній роботі також розглянута бібліотечна система Alma. Alma — це хмарна інтегрована бібліотечна система (ILS), розроблена ізраїльською компанією Ex Libris, яка є частиною корпорації Clarivate. Alma позиціонується як комплексне рішення для академічних та наукових бібліотек, орієнтоване на об’єднання всіх ключових бібліотечних процесів в одній платформі. Її функціональність охоплює управління фондами, електронними та друкованими ресурсами, ліцензіями, каталогізацією, звітністю, обслуговуванням користувачів тощо.

На відміну від традиційних бібліотечних систем, Alma створена з урахуванням потреб бібліотек у цифрову епоху та оптимізована для роботи з електронними ресурсами, інтеграції з базами даних, платформами відкритого доступу, системами дистанційного навчання тощо.

Основні характеристики Alma:

1. Хмарна інфраструктура: система повністю розміщена в хмарному середовищі, що знімає потребу в локальному хостингу або адмініструванні серверів.
2. Єдина платформа для всіх ресурсів: Alma дозволяє керувати друкованими, електронними та цифровими ресурсами з одного інтерфейсу.
3. Підтримка аналітики: вбудовані інструменти звітності та бізнес-аналітики для аналізу діяльності бібліотеки.
4. Інтеграція з іншими продуктами Ex Libris: зокрема з системою пошуку Primo, яка надає єдиний доступ до каталогу та електронних ресурсів.
5. Автоматизація робочих процесів: підтримка сценаріїв для спрощення закупівель, каталогізації, доступу до ліцензій та користувацьких запитів.
6. Масштабованість: Alma використовується великими бібліотечними консорціумами та університетами, зокрема Гарвардом, Стенфордом, Оксфордом.

Система також підтримує сучасні протоколи інтеграції (REST API, SAML, OAI-PMH) і має гнучку архітектуру, що дозволяє адаптувати її під потреби окремої бібліотеки або всієї бібліотечної мережі [4].

Головне вікно бібліотечної системи Alma представлено на рисунку 1.2

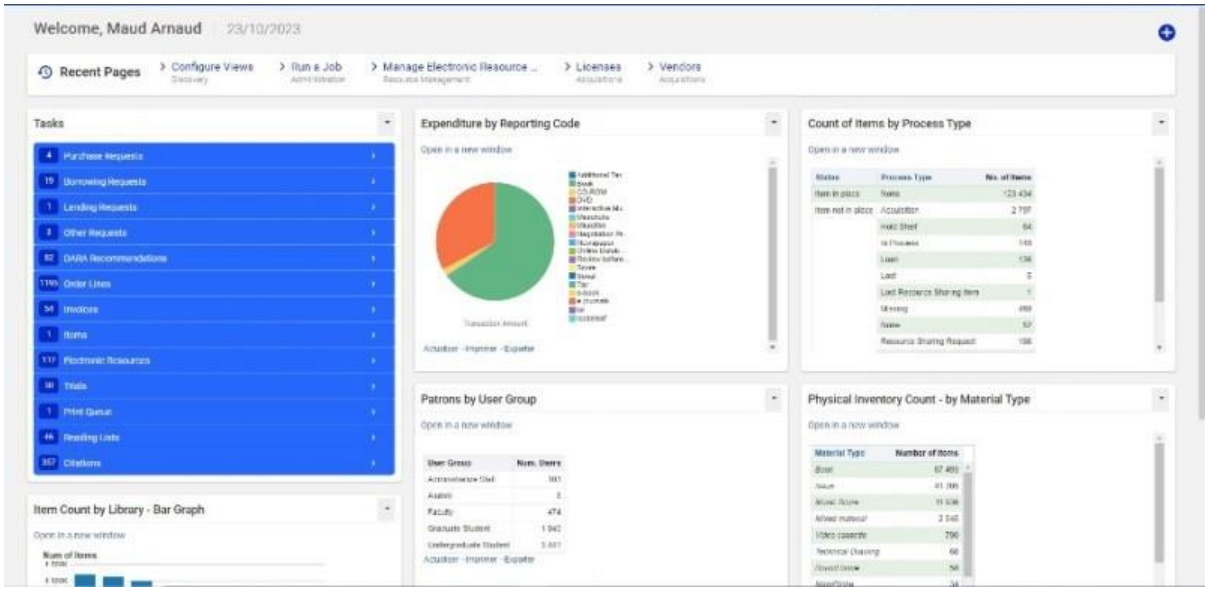

Рисунок 1.2 - Головна вікно бібліотечної системи Alma.

В кваліфікаційній роботі також розглянута бібліотечна система бібліотечна система Aleph. Aleph (Automated Library Expandable Program) — це одна з найвідоміших інтегрованих бібліотечних систем, розроблена компанією Ex Libris. Вона широко використовувалася у великих наукових, університетських і національних бібліотеках упродовж багатьох років. Система була створена ще у 1980-х роках, але з роками зазнала значної еволюції, перетворившись на багатофункціональне середовище для управління всіма аспектами бібліотечної роботи.

Aleph — це потужне рішення, орієнтоване насамперед на традиційні друковані фонди, але також підтримує електронні ресурси, інтеграцію з каталогами та системами пошуку. Вона створена як модульна система, що дозволяє масштабувати функціональність відповідно до потреб конкретної установи.

Основні характеристики Aleph:

1. Модульна архітектура: окремі модулі для каталогізації, циркуляції, управління фондами, серіальними виданнями, адміністрування користувачів тощо.
2. Підтримка міжнародних стандартів: MARC21, UNIMARC, Z39.50, OAI-PMH, що забезпечує сумісність з іншими бібліотеками та системами.
3. Розширена інтеграція: можливість взаємодії з пошуковими платформами (наприклад, Primo), системами дистанційного навчання, системами управління знаннями.
4. Стабільність і надійність: десятки років використання у великих бібліотеках доводять її надійність у продуктивному середовищі.
5. Гнучкість у налаштуванні: підтримка локалізації, специфічних політик обслуговування та обліку.

Aleph надає повний цикл обслуговування: від створення бібліографічного запису до списання ресурсу. Її інтерфейс побудований у вигляді клієнт-серверної архітектури, що вимагає встановлення клієнта на робочі станції бібліотекарів. Водночас це зумовлює меншу гнучкість у порівнянні з сучасними веборієнтованими системами [5].

Головне вікно системи Aleph представлено на рисунку 1.3

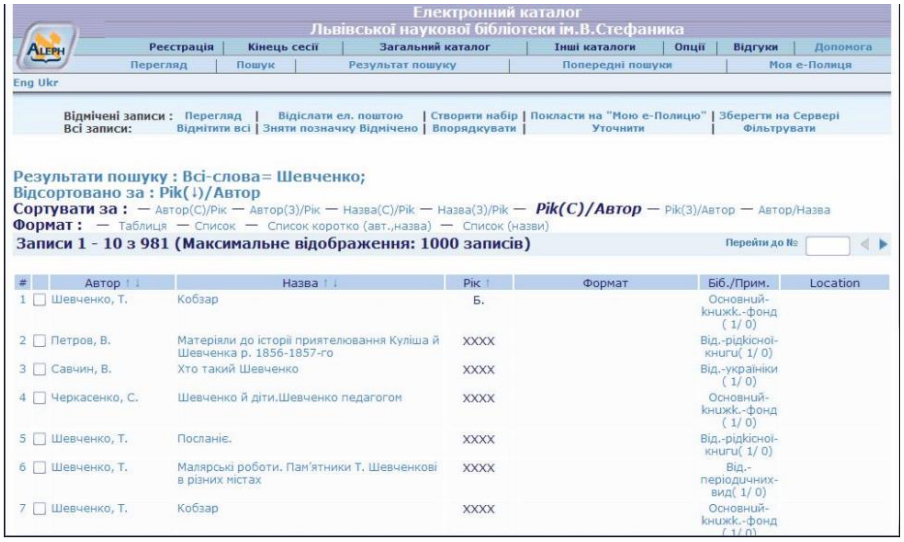

Рисунок 1.3 - Головне вікно бібліотечної системи Aleph.

Однак, попри переваги таких систем, не всі з них підходять для малих бібліотек через складність розгортання, потребу у хостингу чи ліцензіях. Наприклад, Koha вимагає знань адміністрування Linux, а ALMA — комерційна та потребує підписки. Саме тому доцільним є створення простого вебзастосунку з базою на PostgreSQL, який реалізує ключову функціональність без надмірної складності. Такий підхід дозволяє адаптувати рішення під конкретні потреби невеликих установ і не вимагає серйозних ресурсів для впровадження.

Порівняльна характеристика розглянутих середовищ представлена у таблиці 1.1

Таблиця 1.1 – Порівняльна характеристика бібліотечних систем Koha, Alma, Aleph.

Назва	Відкрите ПЗ	Наявність української локалізації	Ліцензія	Можливість локального впровадження
Koha (використовується в University of Alberta, Канада)	+	+	Open Source	+
Alma (використовується в (Harvard University, США)	-	+	Платна	- (тільки хмара)
Aleph (використовується в Library of Congress, США)	-	-	Платна	+

При розробці за стосунку враховуються результати порівняльної характеристики бібліотечних систем.

1.3 Завдання на розробку проєкту

В результаті аналізу подібних програмних продуктів сформовано завдання на розробку проєкту :

1. Розробити архітектуру системи.

2. Розробити інтерфейс користувача, який зручний, зрозумілий і адаптований під потреби бібліотекаря;

3. Виконати підключення до бази даних PgAdmin.

4.Протестувати розроблену систему різними методами.

Висновки до першого розділу

В першому розділі кваліфікаційної роботи було проведено аналіз проблематики ручного ведення що показало, що ручний метод ведення бібліотечної системи не є ефективним.

Також було розглянуто аналогічні програмні системи Koha, Alma, Aleph було складено таблицю порівняння характеристик для вибору подальшої стратегії роботи.

Проведений аналіз показав, що традиційні методи управління бібліотекою не відповідають сучасним вимогам, щодо ефективності, точності та швидкості обслуговування. Відсутність автоматизації створює серйозне навантаження на бібліотекарів, ускладнює процес пошуку та обліку літератури, спричиняє втрату інформації та збільшує ризик помилок.

Тому в першому розділі роботи було розроблено сформовано завдання на розробку проекту.

РОЗДІЛ 2 ПРОЄКТУВАННЯ СУБД ТА РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ

2.1 Аналіз вимог до програмного забезпечення, що розробляється

У процесі розробки вебзастосунку важливу роль відіграє попереднє визначення функціональних та нефункціональних вимог до системи. Це дозволяє вже на етапі проєктування врахувати всі ключові аспекти майбутнього застосунку, побудувати оптимальну архітектуру та реалізувати логіку роботи без критичних помилок. Завдяки цьому програмне забезпечення, що створюється, відповідає вимогам щодо надійності, зручності використання та ефективності виконання основних бібліотечних операцій [6].

Під час формування вимог проводиться зіставлення з функціоналом наявних професійних бібліотечних систем, таких як Koha, Aleph, Alma тощо. Проте, на відміну від них, розроблюваний застосунок не потребує складного налаштування або серверного середовища. Він орієнтований на автономну роботу в межах одного комп’ютера або локальної мережі, що значно спрощує його використання [7].

Для створення застосунку буде використано середовище розробки Visual Studio, що дозволяє зручно керувати проєктом, перевіряти код, налагоджувати програму і створювати графічний інтерфейс.

Завдяки поєднанню PgAdmin і Python буде створено гнучкий і зрозумілий застосунок, який дозволить бібліотекарям швидко працювати з базою даних, не вникаючи в технічні деталі.

Для розробки вебзастосунку було обрано поєднання перевірених технологій, які забезпечують баланс між простотою, доступністю, функціональністю та можливістю підтримки в умовах обмежених

ресурсів. Основна вимога — забезпечити повноцінну роботу системи без необхідності встановлення складного серверного середовища.

У ролі системи управління базою даних (СУБД) використано PgAdmin який має зручний графічний інтерфейс для проєктування структури таблиць, встановлення зв’язків і перегляду вмісту бази. Вибір PostgreSQL зумовлений його простотою, підтримкою реляційної моделі, а також можливістю локального зберігання даних без потреби у мережевому сервері. База даних PgAdmin зберігається у вигляді одного .sql, що спрощує резервне копіювання і перенесення між машинами.

Для розробки програмної частини вебзастосунку було використано мову програмування Python та середовище розробки Microsoft Visual Studio. Python інтегрується з базами даних через спеціальні бібліотеки (psycopg2 для PostgreSQL). Це дозволяє програмістам зручно працювати з даними: здійснювати підключення, виконувати запити, зберігати та обробляти інформацію. Python використовується для автоматизації, аналізу даних, розробки веб-додатків і створення скриптів для роботи з базами даних.

Комунікація між Python і базою даних PostgreSQL, яка керується через pgAdmin, здійснюється за допомогою бібліотеки psycopg2. Це дуже зручний інструмент, який дозволяє без проблем підключатися до бази даних, виконувати SQL-запити та обробляти результати в Python. Завдяки цьому інтеграція між мовою програмування та базою даних стає простою та ефективною [8].

Для створення сайту на Python через Visual Studio використані наступні інструменти та технології:

- Python — основна мова програмування для розробки серверної частини сайту.

- Flask — обрано цей фреймворк через його легкість і гнучкість, що ідеально підходить для створення простих та середніх проектів. Він дозволяє швидко налаштувати сервер та додавати потрібні функції.
- Visual Studio — середовище розробки, яке надало всі необхідні інструменти для написання коду, налагодження та тестування. Завдяки підтримці Python через розширення, робота з проектом стала значно зручнішою.
- HTML, CSS, JavaScript — ці технології використані для створення фронтенду сайту, оформлення сторінок і взаємодії з користувачем.
- SQL (PostgreSQL) — для зберігання даних сайту використано PostgreSQL, оскільки ця реляційна база даних є потужною і надійною для роботи з великою кількістю структурованих даних.
- psycopg2 — бібліотеку для взаємодії Python з PostgreSQL використано для роботи з базою даних, що дозволяє ефективно виконувати SQL-запити та обробляти результати.

Ці технології дозволили створити функціональний, масштабований і ефективний веб-додаток.

Обрані ці технології з кількох причин. Python — універсальна і зручна мова програмування, яка дозволяє швидко створювати серверну частину веб-додатків з мінімумом коду. Це значно прискорює розробку і полегшує підтримку проекту. Flask був обраний завдяки його легкості та гнучкості, що дає змогу швидко створювати невеликі та середні веб-додатки з можливістю кастомізації за потреби. Завдяки мінімалістичному підходу Flask дозволяє зосередитись на реалізації основного функціоналу, без зайвих складнощів [9].

Для роботи в середовищі Visual Studio було вибрано саме це IDE, оскільки воно містить усі необхідні інструменти для написання, налагодження та тестування Python-коду. Крім того, воно забезпечує

інтеграцію з іншими технологіями, що значно спрощує процес розробки та тестування.

Щодо фронтенду, використано HTML, CSS та JavaScript. Ці технології є стандартом для сучасної веб-розробки, і вони дозволяють створювати адаптивні, інтерактивні веб-сторінки з привабливим дизайном. Це покращує взаємодію користувачів з сайтом і робить його доступним на різних пристроях.

Для зберігання даних використовувалось PostgreSQL — надійну і потужну реляційну базу даних, здатну ефективно працювати з великими обсягами інформації, що важливо для сучасних веб-додатків. psycopg2 використовувалась як бібліотека для роботи з PostgreSQL, оскільки вона забезпечує швидку і зручну інтеграцію з Python, дозволяючи без проблем виконувати SQL-запити і працювати з даними з бази.

Загалом, ці технології дають змогу створювати потужні, масштабовані та ефективні веб-додатки, поєднуючи зручність Python із потужними інструментами для роботи з даними і розробки фронтенду.

2.2. Архітектура системи

Розроблений вебзастосунок реалізовано за клієнт-серверною архітектурою. Бібліотекар взаємодіє з клієнтською частиною через браузер. Клієнська частина надсилає HTTP-запити до сервера, реалізованого за допомогою фреймворку Flask. Сервер, у свою чергу, обробляє запити, звертається до бази даних PostgreSQL, отримує або змінює інформацію, після чого повертає результат клієнту у вигляді HTML-сторінки або JSON-даних [10].

Схема взаємодії компонентів системи представлена на рисунку 2.3

Рисунок 2.3 – Схема взаємодії системи.

2.3 Логічна структура бази даних

База даних складається з п’яти основних таблиць:

- authors — зберігає дані про авторів (public, authors, id serial, full_name);
- genres — жанри книг (public, genres, id serial, name);
- books — містить книги, прив’язані до авторів та жанрів (public, books, id serial, title, authors_id, genre_id, publish_year, isbn);
- readers — зареєстровані читачі бібліотеки (public, readers, id serial, full_name, email, phone);
- loans — інформація про видачу книг (яка книга, кому, на який термін) (public, loans, id serial, book_id, reader_id, loan_date, return_date).

Er-діаграма бази даних представлена на рисунку 2.4

Між таблицями бази даних встановлені такі зв’язки:

- один автор може написати багато книг;
- один жанр може відповідати багатьом книгам;
- одна книга може бути видана багато разів різним читачам;
- один читач може взяти багато книг.

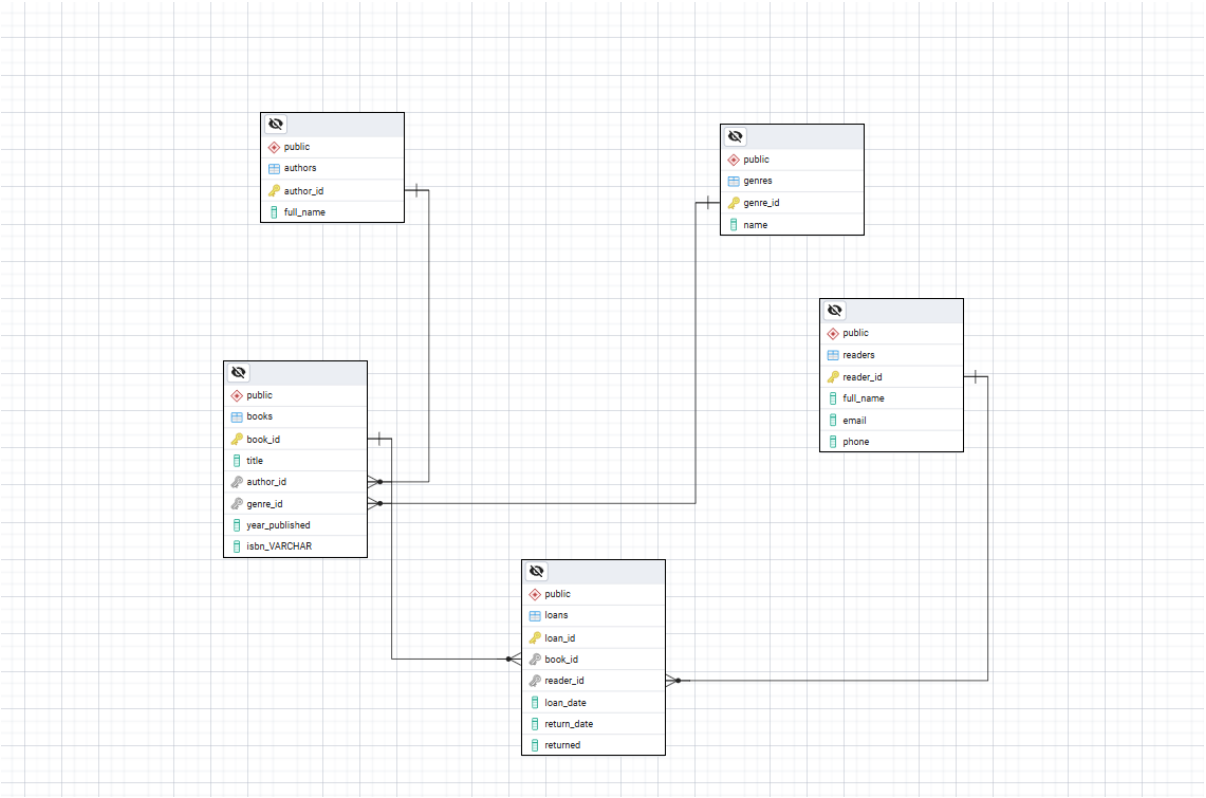

Рисунок 2.4 - ER-діаграма бази даних.

2.4 Функціональність веб-сайту

При розробці веб-сайту потрібно враховувати такий функціонал:

- перегляд списку книг з фільтрацією за автором або жанром;
- додавання нової книги (вибір автора, жанру, введення ISBN);
- реєстрація читача (ПІБ, email, телефон);
- оформлення видачі книги (вибір книги, читача, дати);
- сторінка активних позик, де показано, які книги не повернуто.

Такий фунціонал можна роеалізувти за домогою наступних інструментів:

Для реалізації автоматизованого вебзастосунку з управління бібліотекою у межах цієї кваліфікаційної роботи було використано поєднання двох основних технологій: PgAdmin для зберігання даних

та мову програмування Python для створення логіки вебінтерфейсу та взаємодії з базою даних.

PgAdmin — це графічний інтерфейс для керування базами даних PostgreSQL. Він дозволяє користувачам працювати з базами даних, створювати таблиці, виконувати SQL-запити та налаштовувати сервери. Це потужний інструмент для адміністрування PostgreSQL, який спрощує управління та аналіз даних.

У межах цього проекту PgAdmin буде використовуватись як основа для зберігання таких даних:

- список книг (назва, автор, жанр, рік видання, інвентарний номер);
- інформація про читачів (ПІБ, клас або група, контактні дані);
- журнал видачі книг (хто, коли і яку книгу взяв, дата повернення);

Для розробки інтерфейсу використовуються бібліотеки мови Python.

Python — це високорівнева мова програмування, відома своєю простотою та універсальністю. Вона використовується для розробки веб-додатків, аналізу даних, штучного інтелекту, автоматизації задач та інтеграції з базами даних. Python має зрозумілий синтаксис, що дозволяє швидко освоїти мову, та величезну кількість бібліотек, що робить її популярною серед розробників по всьому світу.

2.5 Програмна реалізація проекту

Розробку вебзастосунку для управління бібліотекою реалізовано з використанням мови програмування Python та мікрофреймворку Flask, що забезпечує зручне створення маршрутизованої логіки та обробку HTTP-запитів. У якості системи керування базами даних обрано PostgreSQL, яка забезпечує надійне зберігання та обробку інформації про книги, читачів, авторів, жанри та видачі [11].

При підключенні бази даних до вебзастосунку було використано ORM-бібліотеку SQLAlchemy, яка підтримується фреймворком Flask. Підключення здійснюється за допомогою конфігураційних параметрів, які задаються у файлі app.py.

Фрагмент коду з підключенням бази даних зображено на рисунку 2.6

```
app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'postgresql://postgres:123456@localhost/librarydb'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
```

Рисунок 2.5 - Підключення бази даних до сайту.

Опис структури проекту відображена на рисунку 2.7

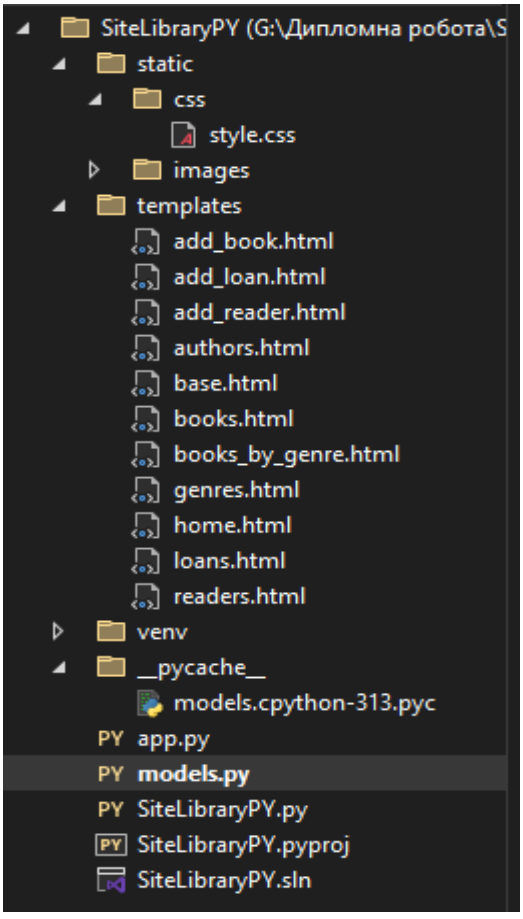

Рисунок 2.6 - Структура файлової системи вебзастосунку має такий вигляд.

Елементи структури наведені нижче:

- SiteLibraryPY - коренева папка
- Static - містить статичні ресурси, які не змінюються під час роботи сайту (CSS, зображення, JS-файли тощо).
- style.css – відповідає за загальне оформлення сайту: кольори, шрифти, таблиці, кнопки, відступи. Застосовується до всіх сторінок для забезпечення єдиного стилю.
- images - зберігається зображення — логотипи, обкладинки книг, іконки тощо.
- templates - це папка з шаблонами HTML.
- base.html – базовий шаблон сайту. містить спільні для всіх сторінок елементи (меню, шапка, футер). усі інші HTML-шаблони наслідують його.

Сторінки для перегляду:

- home.html – головна сторінка сайту. Містить привітання та навігацію до основних розділів (книги, читачі, позики тощо).
- books.html – виводить повний список книг із бази даних. включає назву, автора, жанр, рік видання та ISBN.
- books_by_genre.html – сторінка, яка показує книги за обраним жанром. дозволяє користувачеві переглядати книги конкретного напрямку.
- authors.html – список авторів. виводить усіх авторів, які є у базі, разом із кнопками перегляду книг певного автора.
- genres.html – виводить список жанрів, які використовуються в системі. Може також містити посилання для фільтрації книг за жанром.
- readers.html – показує усіх зареєстрованих читачів. Виводить ПІБ, email та номер телефону.

- loans.html – таблиця всіх поточних і минулих позик. Відображає, хто яку книгу взяв, на який термін і чи була вона повернена.

Форми для додавання:

- add_book.html – форма для додавання нової книги до бази: назва, автор, жанр, рік, ISBN.
- add_reader.html – форма реєстрації нового читача: ПІБ, email, номер телефону.
- add_loan.html – форма видачі книги читачеві. Вибирається книга, читач, дати позики та повернення.
- app.py - головний файл програми. Запускає сервер, створює маршрути (@app.route), з'єднується з PostgreSQL, обробляє HTML-форми та SQL-запити. Відповідає за логіку застосунку.
- models.py - містить допоміжні функції для роботи з базою даних: отримання книг, додавання читача, створення позики тощо. Виносить частину логіки з app.py для зручності й повторного використання.
- venv - віртуальне середовище Python, де встановлені всі залежності проєкту (Flask, psycopg2 тощо). не входить до основного коду.
- sitelibrarypy.pyproj / SiteLibraryPY.sln - файли налаштування проєкту для Visual Studio. використовуються середовищем розробки для структурування та збирання проєкту.

Маршрутизатор системи описано описано у файлі app.py який реалізує файл add_book.html на сторінці «Додати книгу».

```python
@app.route('/add_book', methods=['GET', 'POST'])
def add_book():
    if request.method == 'POST':
        title = request.form.get('title')
        author_id = request.form.get('author_id')
        new_author_name = request.form.get('new_author', '').strip()
        genre_id = request.form.get('genre_id')
        publish_year = request.form.get('publish_year')
        isbn = request.form.get('isbn')

        if new_author_name:
            new_author = Author(full_name=new_author_name)
            db.session.add(new_author)
            db.session.flush()
            author_id = new_author.id
        else:
            if not author_id:
                author_id = None

        if publish_year and publish_year.isdigit():
            publish_year = int(publish_year)
        else:
            publish_year = None

        new_book = Book(
            title=title,
            author_id=author_id,
            genre_id=genre_id,
            publish_year=publish_year,
            isbn=isbn if isbn else None
        )
        db.session.add(new_book)
        db.session.commit()
        return redirect(url_for('books'))

    authors_list = Author.query.all()
    genres_list = Genre.query.all()
    current_year = datetime.now().year
    return render_template('add_book.html', authors=authors_list, genres=genres_list, current_year=current_year)
```

Рисунок 2.7 – Фрагмент коду, який реалізує додавання книги в довідник.

Висновки до другого розділу

У цьому розділі було здійснено проєктування та програмну реалізацію вебзастосунку для автоматизації діяльності бібліотеки. На основі попередньо визначених функціональних вимог було створено архітектуру системи, розроблено структуру бази даних, а також обрано відповідне програмне забезпечення для реалізації проєкту.

Застосунок реалізовано з використанням мови програмування Python та мікрофреймворку Flask, що забезпечило гнучкість, простоту реалізації маршрутизації та інтеграцію з базою даних PostgreSQL.

Для розділення коду на логічні частини було створено окремі файли шаблонів, стилів та модулів логіки. Підключення до бази даних

здійснюється з використанням бібліотеки psycopg2, а обробка форм і вивід даних реалізовані за допомогою HTML-шаблонів [12].

У межах програмної реалізації було розроблено функціонал для роботи з основними сутностями системи: книги, автори, жанри, читачі та видача книг. Забезпечено можливість додавання нових записів, перегляду списків і взаємодії між ними.

Таким чином, результатом цього етапу є повноцінний вебзастосунок, який дозволяє бібліотекарю ефективно управляти бібліотечним фондом та обліком читачів, що створює основу для подальшого тестування і впровадження системи.

РОЗДІЛ 3 ТЕСТУВАННЯ РОЗРОБЛЕНОГО ЗАТОСУНКА БІБЛІОТЕКАРЯ

3.1 Запуск веб-застосунку у Visual studio 2022 через Developer PowerShell

Для запуску веб-застосунку, створеного на мові Python у середовищі Visual studio 2022, було використано вбудовану консоль Developer PowerShell for Visual studio, зображена на рисунку 3.1. Ця консоль є зручною для розробника, оскільки дозволяє працювати у межах контексту Visual Studio з доступом до всіх необхідних інструментів та SDK.

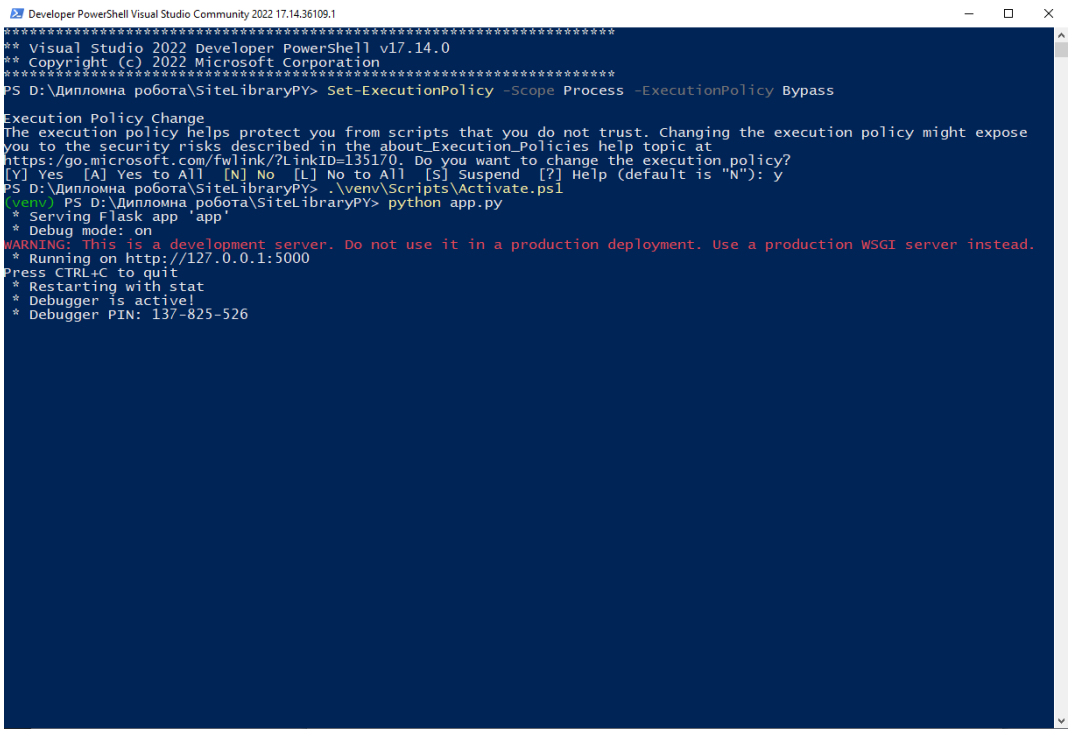

Рисунок 3.1 - Активація віртуального середовища.

У PowerShell за замовчуванням встановлена політика безпеки, яка забороняє виконання локальних скриптів. Щоб дозволити їх тимчасове виконання, вводиться наступна команда, зображена на рисунку 3.2

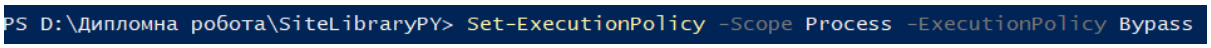

Рисунок 3.2 – Зміна політики виконання скриптів у PowerShell.

Ця команда змінює політику лише на час поточної сесії терміналу. Це безпечний спосіб використання скриптів, який не впливає на глобільні налаштування системи.

Активація віртуального середовища дозволяє ізольовати встановлювати бібліотеки, потрібні лише для цього проекту, не змінюючи глобальні налаштування Python в системі. Активація середовища відбувається через наступну команду, яка зображена на рисунку 3.3

Рисунок 3.3 – Активація віртуального середовища Python.

Після активації зліва у командному рядку з’являється префікс (venv) , що свідчить про успішне завантаження середовища.

Для запуску веб-застосунку використовується стандартна команда запуску Python-скрипту зображена на рисунку 3.4

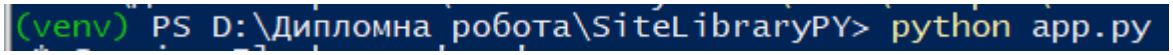

Рисунок 3.4 – Запуск локаного сервера веб-застосунку.

Файл app.py є головним файлом за стосунку, що містить ініціалізацію сервера, підключення до бази даних PostgreSQL, маршрутизацію сторінок та запуск веб-сервера.

У результаті виконання команди виводиться повідомлення, яке зображене на рисунку 3.5

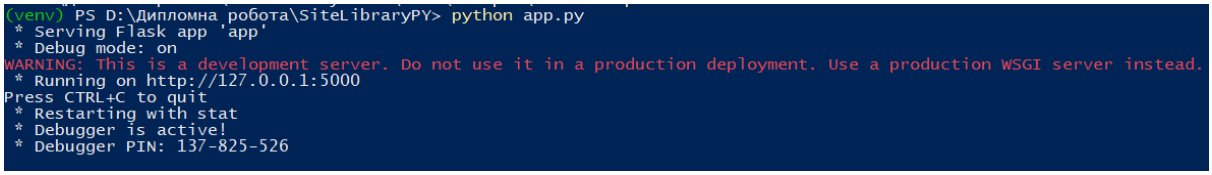

Рисунок 3.5 – Успішний запуск сервера.

Це означає, що веб-застосунок успішно запущено і доступ у локальній мережі.

Після запуску серверу потрібно відкрити браузер і перейти за вказаною адресою http://127.0.0.1:5000/

Відкриється головна сторінка веб-застосунку бібліотекаря, як на рисунку 3.6

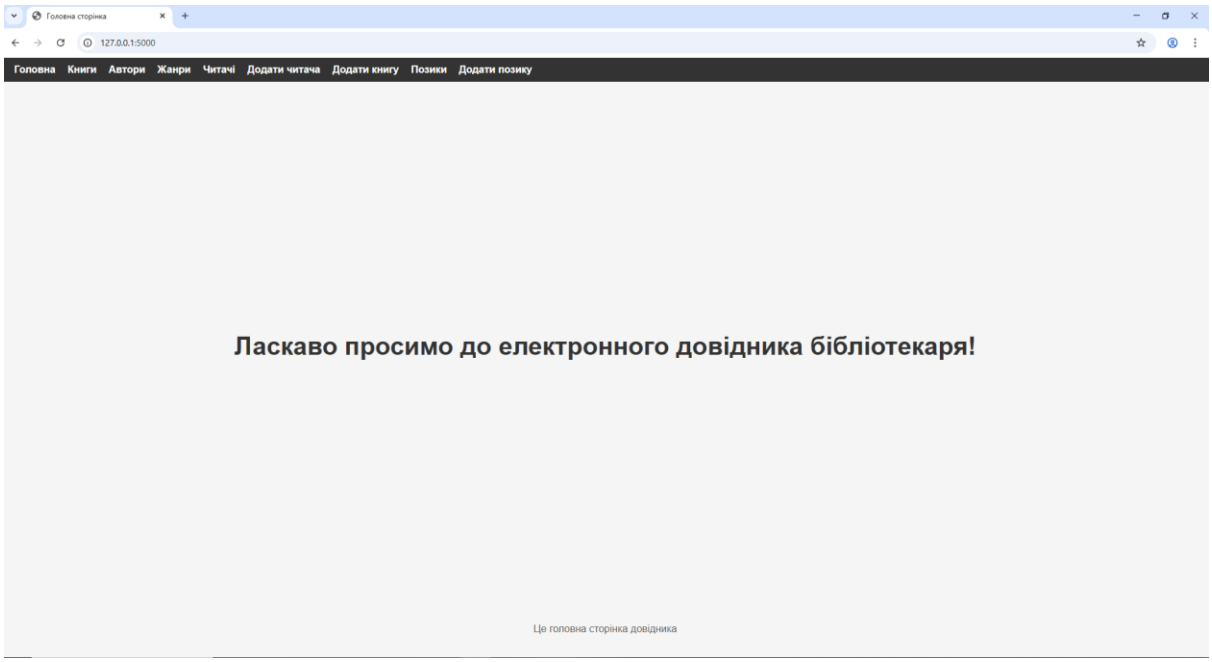

Рисунок 3.6 – Головна сторінка веб-застосунку в браузері.

З неї можна перейти до перегляду бази книг, додавання чи редагування записів тощо, як на рисунку 3.7, 3.8

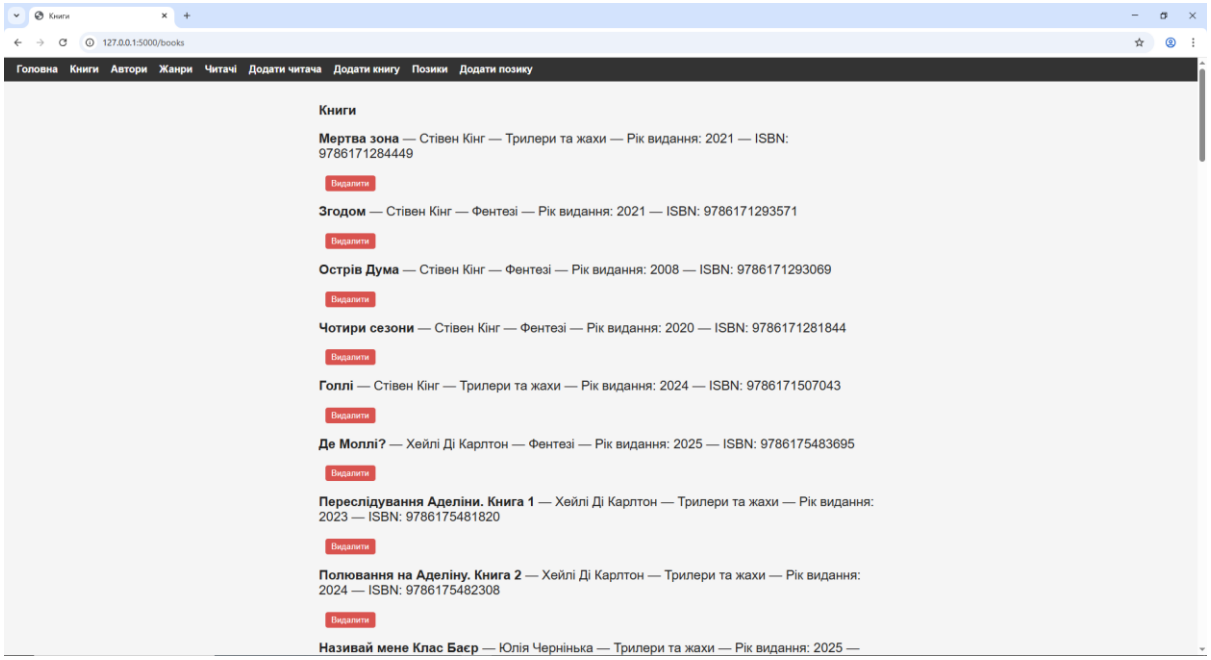

Рисунок 3.7 – Розділ перегляду і редагування книг.

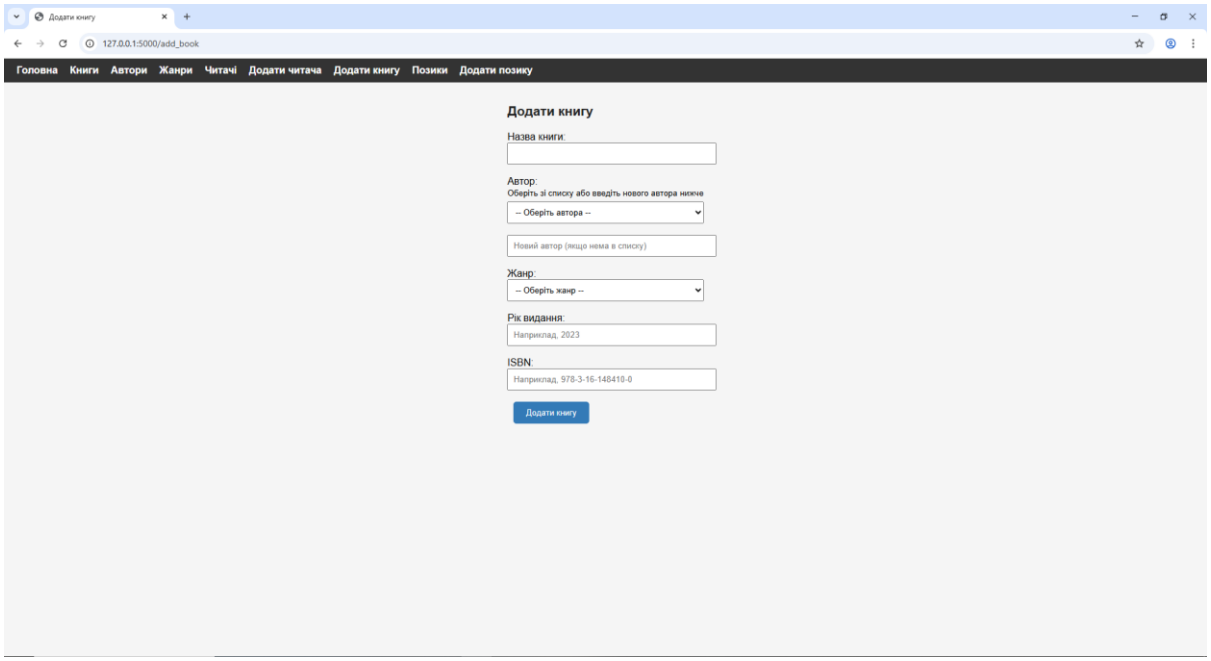

Рисунок 3.8 - Розділ додавання книг.

Тестування модуля додавання книги на рисунку 3.9

Функція додавання протестована таким чином.

3.2 Тестування вебзастосунку

Для цього відкриємо сторінку «Додати книгу» та заповнимо всі необхідні поля. У якості тестового прикладу додамо книгу Тараса Шевченка під назвою «ТестоваКнига».

У форму вводимо такі дані, як на рисунку 3.9:

- назва книги: ТестоваКнига
- автор: Тарас Шевченко
- жанр: Класична література
- рік видання: 1845
- isbn: 978111111111

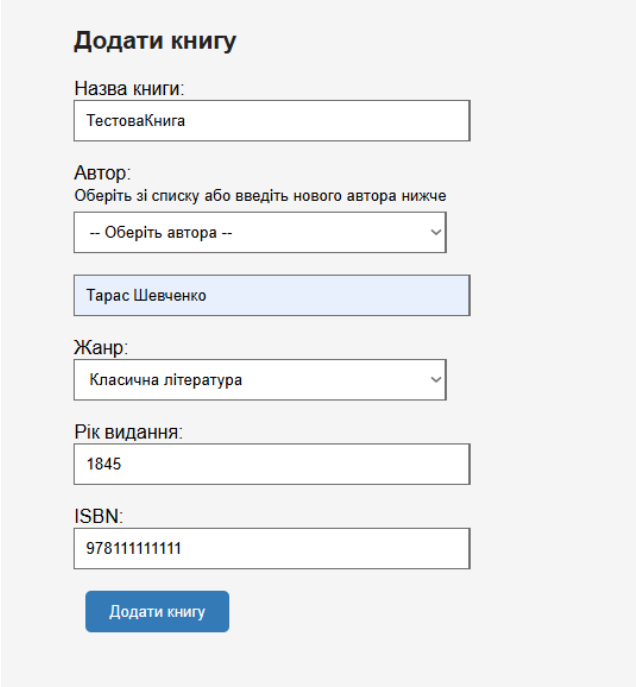

Рисунок 3.9 - Форма додавання книги.

Після цього переходимо до розділу «Книги», рисунок 3.10 .

Виконалось додавання що доданий запис «ТестоваКнига» з’явився у загальному списку. Це підтверджує, що функція додавання нової книги працює коректно, а інформація успішно зберігається у базі даних і відображається на сторінці з доступом до повного списку літератури.

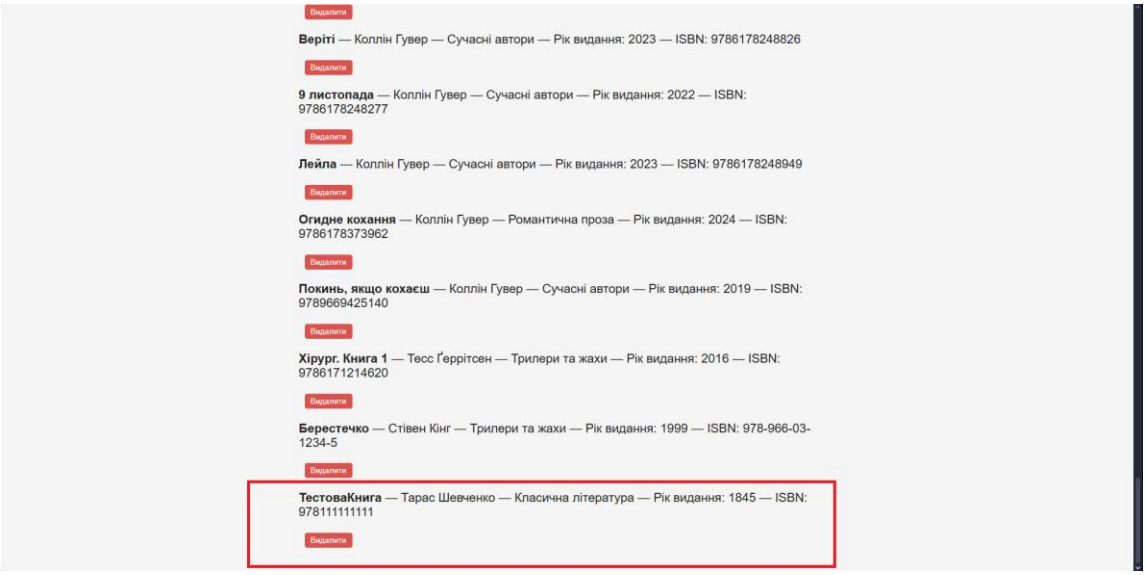

Рисунок 3.10 - Відображення вікна у розділі ”Книги” з’явилась тестова книга.

Для цього переходимо до розділу «Книги», де у загальному списку знаходимо щойно доданий запис. Під назвою книги «ТестоваКнига» розміщена кнопка «Видалити», яка зображена на рисунку 3.11.

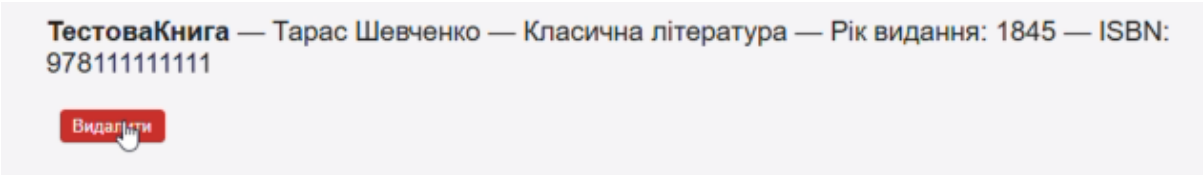

Рисунок 3.11 – Процес видалення книги.

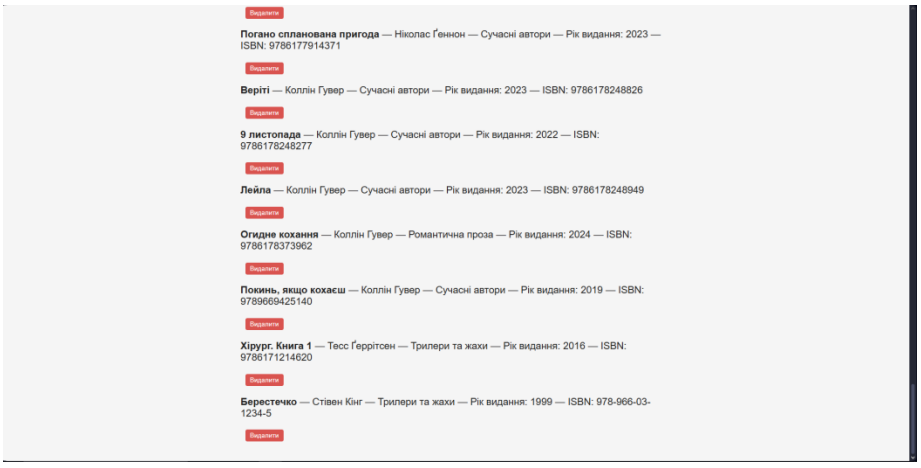

Рисунок 3.12 – Результат виконання команди видалення книги.

Функція видалення книги працює стабільно та без помилок. Після натискання кнопки «Видалити» запис видаляється як з інтерфейсу користувача, так і з бази даних PostgreSQL. Повідомлення про успішне видалення відображається правильно, що дозволяє користувачу чітко розуміти результат виконаної дії.

Для повноцінної роботи бібліотеки, окрім роботи з книгами, необхідно забезпечити можливість додавання нових читачів та оформлення їм книжкових позик. Проведемо тестування цього процесу.

Для перевірки працездатності функції додавання нового читача протестуємо створення читацького запису. У якості прикладу додамо тестового читача з наступними даними:

- ім’я: Андрій
- прізвище: Стрикало
- по батькові: Олегович
- електронна пошта: Andrii999@gmail.com
- телефон: +380967231153

Для цього переходимо до розділу «Читачі» та натискаємо кнопку «Додати читача». У відповідну форму вводимо зазначені дані й натискаємо кнопку «Додати», як на рисунку 3.13.

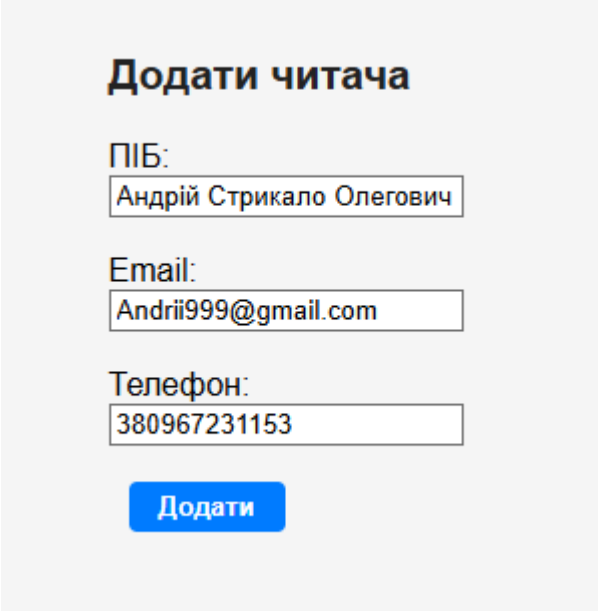

Рисунок 3.13 – Форма додавання нового читача.

Після цього система повідомляє про успішне додавання нового читача, а запис Андрій Стрикало Олегович з'являється у загальному списку читачів, як зображено на рисунку 3.14.

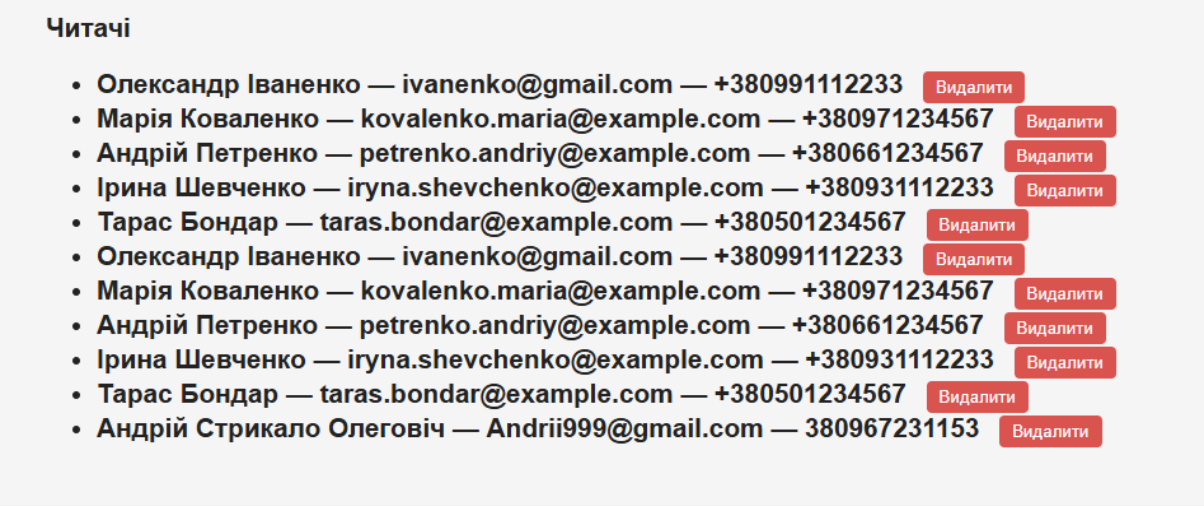

Рисунок 3.14 - Читач Андрій був добавлений до бази Читачів.

Після успішного додавання читача Андрія Стрикала Олеговича протестуємо функцію видачі книги цьому читачу.

Для цього переходимо до розділу «Додати позику». У формі заповнюємо наступні поля:

- книга: Переслідування Аделіни. Книга 1
- читач: Стрикало Андрій Олегович
- дата позики: 18.06.2025
- дата повернення: 18.07.2025

Після заповнення форми натискаємо кнопку «Додати позику», як це зображено на рисунку 3.15.

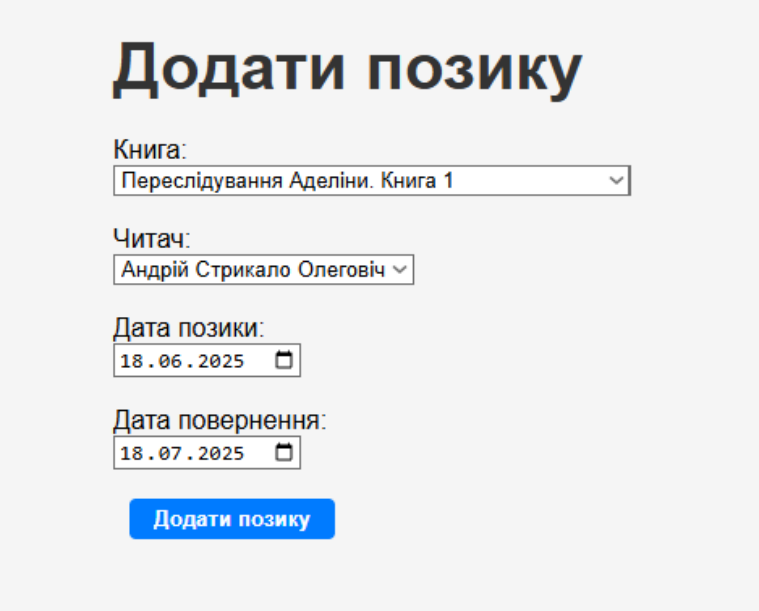

Рисунок 3.15 - Заповнення форми додавання позики.

Позики книг

Книга	Читач	Дата позики	Дата повернення	Дії
Мертва зона	Марія Коваленко	2025-06-01	Не повернено	Видалити
Елантріс	Олександр Іваненко	2025-05-20	2025-06-05	Видалити
Ребекка	Ірина Шевченко	2025-06-10	Не повернено	Видалити
Веріті	Тарас Бондар	2025-06-08	Не повернено	Видалити
Останній лист	Андрій Петренко	2025-05-15	2025-05-30	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Переслідування Аделіни. Книга 1	Андрій Стрикало Олеговіч	2025-06-18	2025-07-18	Видалити

Рисунок 3.16 - Результатна таблиця виконання команди взяття позики.

Після успішного оформлення позики тестується функціональність видалення запису, якщо книга була повернута достроково. Така можливість дозволяє бібліотекарю оперативно оновлювати інформацію у базі даних.

Кроки дій:

1. Переходимо до розділу «Позики».

2. У списку знаходимо запис, де зазначено, що книга «Переслідування Аделіни. Книга 1» була видана читачу Андрій Стрикало Олеговіч.
3. Натискаємо кнопку **«Видалити»,** як на рисунку 3.17

Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Переслідування Аделіни. Книга 1	Андрій Стрикало Олеговіч	2025-06-18	2025-07-18	Видалити

Рисунок 3.17 - Видалення читача з позик який ввідав книгу достроково.

Позики книг

Книга	Читач	Дата позики	Дата повернення	Дії
Мертва зона	Марія Коваленко	2025-06-01	Не повернено	Видалити
Елантріс	Олександр Іваненко	2025-05-20	2025-06-05	Видалити
Ребекка	Ірина Шевченко	2025-06-10	Не повернено	Видалити
Веріті	Тарас Бондар	2025-06-08	Не повернено	Видалити
Останній лист	Андрій Петренко	2025-05-15	2025-05-30	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити
Острів Дума	Олександр Іваненко	2025-06-01	Не повернено	Видалити
Голлі	Марія Коваленко	2025-06-02	2025-06-10	Видалити
Згодом	Андрій Петренко	2025-06-05	Не повернено	Видалити

Рисуснок 3.18 - Чатач який ввідав книгу достроково успішно видалено.

В результаті виконання функції видалення позики довідник працює стабільно. Після підтвердження дії запис повністю видаляється з інтерфейсу та бази даних, що дозволяє підтримувати актуальність інформації у системі.

Висновок до третього розділу

У розділі було проведено повне функціональне тестування основних можливостей веб-застосунку для управління бібліотекою. Зокрема протестовано: додавання та видалення книг, створення читацьких записів, оформлення та видалення позик. Усі перевірені функції працюють стабільно, без збоїв, а результати дій відображаються коректно в інтерфейсі користувача та базі даних PostgreSQL.

Завдяки зрозумілому і логічно побудованому інтерфейсу, веб-застосунок є зручним у використанні для бібліотекаря. Система значно полегшує виконання щоденних завдань, таких як ведення електронного каталогу книг, облік читачів та контроль за позиками.

Проведене тестування підтвердило, що веб-застосунок готовий до практичного використання та може бути впроваджений у роботу реальної бібліотеки для автоматизації її основних процесів.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було розроблено автоматизований вебзастосунок для управління базою даних бібліотеки, що дозволяє ефективно вирішити проблеми, пов’язані з ручним обліком літератури, читачів та позик. Проведений аналіз предметної області та існуючих програмних рішень підтвердив актуальність створення власного, простого у використанні програмного продукту, адаптованого під потреби невеликих бібліотек.

Розроблений застосунок реалізований на основі сучасних технологій — Python, Flask та PostgreSQL, що забезпечує його надійність, гнучкість і зручність у використанні. У результаті тестування підтверджено стабільну роботу всіх функціональних модулів, включно з додаванням, редагуванням і видаленням даних, що свідчить про готовність системи до практичного впровадження.

Таким чином, поставлена мета кваліфікаційної роботи — створення функціонального вебзастосунку для автоматизації бібліотечної системи — повністю досягнута, а отримані результати мають практичне значення для закладів освіти та інших установ, що потребують простого рішення для обліку бібліотечних ресурсів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Литвиненко, Н. А. (2021). Автоматизація бібліотечних процесів у сучасних умовах цифровізації. Вісник Книжкової палати, Бібліотекознавство. Документознавство. Інформологія : наук. журн. / М-во культури та інформ. політики України, Нац. акад. керів. кадрів культури і мистецтв. Київ : НАКККіМ, 2021. № 1. 86 с. URL: https://nakkkim.edu.ua/images/Instytuty/nauka/vydannia/bibliotek_dok_inform/Bibliotekoznavstvo_1_2021_DOI.pdf(дата звернення 19.06.2025)
2. Бондаренко, Г. П. (2018). Проблеми автоматизації бібліотек в Україні. // Наукові праці НДІ інформатики і права. URL: https://dnpb.gov.ua/wp-content/uploads/2017/11/3-1.pdf
3. Koha Library Software — URL: https://koha-community.org
4. Ex Libris (2024). Alma Library Services Platform Overview. URL: https://exlibrisgroup.com/products/alma-library-services-platform
5. Ex Libris. (2022). Aleph Integrated Library System – Product Information. URL: https://exlibrisgroup.com/products/aleph-integrated-library-system
6. Pressman, R. S. Software Engineering: A Practitioner’s Approach. McGraw-Hill Education, 2014. URL : https://mlsu.ac.in/econtents/16_EBOOK-7th_ed_software_engineering_a_practitioners_approach_by_roger_s._pressman_.pdf
7. Koha Integrated Library System Documentation. URL: https://koha-community.org/documentation/
8. Psycopg2 Documentation. URL: https://www.psycopg.org/docs/

9. Husayn Fakher, Unveiling Flask: A Versatile Python Microframework for Web Development, Medium, 16 травня 2024 року. URL: https://medium.com/@speaktoharisudhan/flask-a-web-framework-435de255e81b
10. Wikipedia contributors. (2024). Client–server model. Wikipedia. https://en.wikipedia.org/wiki/Client–server_model
11. Flask. The Python microframework for building web applications. Офіційний сайт: https://flask.palletsprojects.com
12. Psycopg – PostgreSQL database adapter for Python. https://www.psycopg.org/