МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра економічної кібернетики та інформатики

МІЖДИСЦИПЛІНАРНА КУРСОВА РОБОТА

на тему:

"Інформаційна система замовлень у бібліотеці."

Студентки 4 курсу групи СА-41
спеціальності 124 «Системний аналіз»
Спяк У.В.,
(прізвище та ініціали)
Керівник
_____
_____
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____

Кількість балів: _____Оцінка: ECTS _____

Члени комісії: _____(_____)
прізвище та ініціали підпис
_____(_____)
прізвище та ініціали підпис
_____(_____)
прізвище та ініціали підпис

м. Тернопіль – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра економічної кібернетики та інформатики

ЗАВДАННЯ НА МІЖДИСЦИПЛІНАРНУ КУРСОВУ РОБОТУ

група СА -41 курс 4

Студентки Спяк У.В.

Тема міждисциплінарної курсової роботи:

Інформаційно-пошукова система замовлень у бібліотеці.

Основні розділи міждисциплінарної курсової роботи:

Вступ

1. Аналіз проблем управління біліотечними процесами
2. Інструментальні засоби для автоматицізації роботи біліотек
3. Розробка інформаційної системи замовлень у бібліотеці

Висновки

Рекомендована література:

1. Державна бібліотека України для юнацтва. Інформаційні технології у бібліотеках.
2. "Інформаційні технології в бібліотечній справі: методи та підходи". Навчальний посібник, Київ, 2021.
3. Політехніка Львів. "Розробка веб-додатків для бібліотек".

Додатки

Дата видачі завдання "____"_____ 2024р.

Термін представлення роботи на кафедру "____"_____ 2024р.

Керівник міждисциплінарної курсової роботи _____

Зміст

ВСТУП

Ручні процеси, які раніше були основою функціонування бібліотек, сьогодні є неефективними, що спричиняє затримки в обслуговуванні користувачів, помилки в обліку ресурсів та ускладнення у доступі до літератури. Це викликає необхідність автоматизації процесів, що дозволяє оптимізувати операції з каталогізацією, бронюванням та управлінням книжковими фондами. Автоматизація бібліотечних процесів включає створення інтегрованих інформаційних систем, які забезпечують зручність для користувачів і спрощують роботу персоналу. Основна мета таких систем полягає у зменшенні часу на обробку запитів, підвищенні точності ведення записів та забезпеченні безперебійного доступу до бібліотечних ресурсів. У даній роботі розглядається розробка інформаційної системи замовлень у бібліотеці, яка охоплює усі ключові аспекти автоматизації цього процесу.

Мета роботи полягає у розробці моделі для автоматизації процесів управління замовленнями книг у бібліотеці, що включає створення бази даних, серверної та клієнтської частин.

Результатом дослідження стане інтегрована інформаційна система, що забезпечить автоматизацію замовлення книг у бібліотеці, спростить взаємодію користувачів із бібліотекою та підвищить ефективність управління ресурсами.

РОЗДІЛ 1

АНАЛІЗ ПРОБЛЕМ УПРАВЛІННЯ БІБЛІОТЕЧНИМИ ПРОЦЕСАМИ

1.1 Проблеми бібліотечного управління в сучасному світі

Сучасне управління бібліотеками стикається з низкою проблем, що ускладнюють їх ефективність. Однією з основних є використання ручного управління та застарілих процесів, що призводить до помилок в обліку, дублювання записів і тривалого пошуку інформації. Відсутність централізованих баз даних ускладнює облік фондів, відстеження замовлень і співпрацю з іншими бібліотеками. Зміни в поведінці користувачів, які очікують цифрових сервісів і швидкого доступу до ресурсів, роблять традиційні бібліотеки менш актуальними.

Обмежені фінансові ресурси обмежують впровадження сучасних технологій, оновлення фондів і залучення кваліфікованого персоналу. Інформаційне перевантаження, характерне для наукових бібліотек, створює труднощі в організації матеріалів і забезпеченні доступу до них. Низький рівень автоматизації процесів,підвищує навантаження на персонал і сповільнює обслуговування.

Також бібліотеки стикаються з юридичними викликами, як-от захист авторських прав і конфіденційності даних користувачів, а також технічною застарілістю обладнання та програмного забезпечення. Ці проблеми вимагають впровадження сучасних інформаційних систем, що дозволять автоматизувати процеси, покращити управління фондами та забезпечити швидкий доступ до інформації. Інноваційні рішення, такі як інтегровані бази даних, веб-платформи та автоматизовані сервіси, здатні значно підвищити ефективність бібліотек і відповідати сучасним потребам користувачів.

Рисунок 1.1 – Одна з унікальних бібліотек світу

1.2 Сучасні інформаційні технології в управлінні бібліотеками

Сучасні інформаційні технології значно вдосконалили управління бібліотеками, дозволяючи автоматизувати процеси, підвищувати якість обслуговування та ефективно управляти ресурсами. Важливим напрямом є використання автоматизованих бібліотечних систем, таких як Aleph, Koha, Ex Libris та інші, які забезпечують облік фондів, каталогізацію, бронювання та контроль доступу. Також активно впроваджуються хмарні технології, що дозволяють зберігати великі обсяги даних, забезпечуючи доступ до них із будь-якого місця, що особливо важливо для академічних і наукових установ.

Цифрові платформи, включно з електронними каталогами та бібліотеками, відкривають користувачам можливості для дистанційного доступу до книг, статей і дослідницьких матеріалів, а також роблять ресурси доступними для віддалених регіонів. Інноваційні RFID-технології автоматизують облік і видачу книг, мінімізують втрати фонду та полегшують інвентаризацію. Мобільні

додатки, які надають доступ до каталогів, бронювання книг та нагадувань, підвищують зручність для користувачів і якість послуг.

Одночасно велика увага приділяється інформаційній безпеці, використовуючи методи шифрування, управління доступом і моніторингу для захисту даних. Усі ці інновації адаптують бібліотеки до вимог цифрового світу, сприяють підвищенню ефективності роботи та розширюють доступ до інформаційних ресурсів.

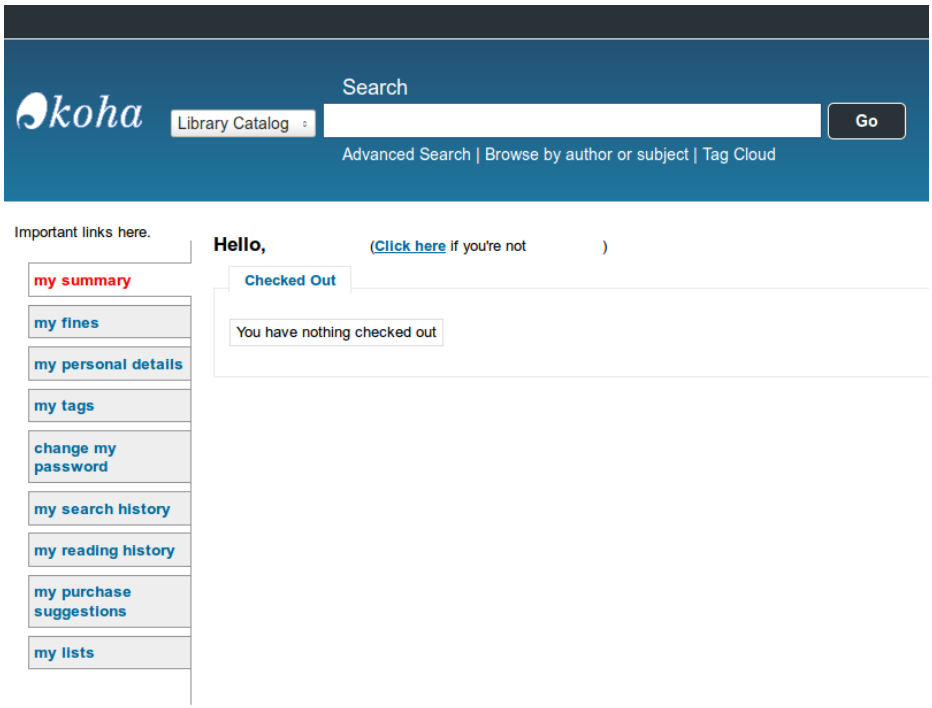

Рисунок 1.2 - Вигляд сайту Koha

1.3 Обґрунтування вибору підходу до автоматизації

Обґрунтування вибору підходу до автоматизації бібліотечних процесів ґрунтується на потребі забезпечення ефективного управління, зручного доступу до ресурсів та високої якості обслуговування користувачів. Основними критеріями для вибору є простота впровадження, функціональна гнучкість і масштабованість системи. Автоматизація має на меті оптимізувати процеси каталогізації, інвентаризації та видачі літератури, а також забезпечити безперебійний доступ до електронних ресурсів.Застосування хмарних платформ дозволяє скоротити витрати на апаратне забезпечення та забезпечити доступ до системи з будь-якого пристрою, тоді як RFID-технології спрощують контроль за

фондами та швидкість обслуговування користувачів. Обираючи підхід до автоматизації, також враховуються інтеграція з існуючими інформаційними системами та можливість розвитку у майбутньому, що гарантує відповідність сучасним технологічним вимогам. Таким чином, підхід орієнтований на створення зручного, функціонального й надійного середовища для управління бібліотечними процесами.

РОЗДІЛ 2
АНАЛІЗ ТЕХНОЛОГІЙ ДЛЯ АВТОМАТИЗАЦІЇ БІБЛІОТЕК

2.1 Загальний огляд методів роботи з інформаційними базами

Методи роботи з інформаційними базами є ключовим аспектом сучасного управління даними, забезпечуючи ефективну організацію, зберігання та доступ до великих обсягів інформації. Основними підходами є реляційні бази даних, які використовують табличну структуру для збереження даних і забезпечують швидкий доступ через мову запитів SQL, а також нереляційні бази даних, що підходять для зберігання неструктурованої інформації, наприклад, документів або мультимедійних файлів.

Особливу увагу приділяють технікам резервного копіювання та відновлення інформації, що дозволяє мінімізувати ризики втрати даних.

Автоматизація роботи з інформаційними базами включає створення програмного забезпечення для управління даними, використання API для інтеграції з іншими системами та застосування аналітичних інструментів для обробки великих обсягів інформації. Зокрема, сучасні хмарні платформи пропонують масштабовані рішення для роботи з базами даних, забезпечуючи гнучкість у розподілі ресурсів та доступ до інформації в режимі реального часу.

Загалом, ефективні методи роботи з інформаційними базами є основою для реалізації автоматизованих систем у різних сферах, включаючи управління бібліотечними процесами, що забезпечує швидкий доступ до даних і зручність роботи з інформацією.

2.2 Постановка задачі розробки бібліотечної системи замовлень

Розробка бібліотечної системи замовлень спрямована на автоматизацію ключових процесів бібліотеки, зокрема пошуку, резервування та обліку книг. Основна мета – створення інтегрованої платформи, яка забезпечить ефективний доступ до бібліотечних ресурсів, спростить взаємодію користувачів із системою та зменшить адміністративні витрати.

Запропонована система має вирішити основні проблеми бібліотечного управління: складність у пошуку необхідної літератури, відсутність актуальної інформації про доступність книг, затримки в обробці запитів і надмірне навантаження на персонал. В основі підходу лежить використання сучасних інформаційних технологій для створення функціонального інтерфейсу, який включатиме електронний каталог бібліотеки, пошук за заданими критеріями, управління замовленнями та бронюванням, а також реєстрацію користувачів і облік їхньої взаємодії із системою.Окрім базових функцій, система повинна мати модуль звітності, що дозволяє адміністраторам аналізувати статистику використання бібліотечних ресурсів. Це сприятиме оптимізації бібліотечних процесів і підвищенню задоволеності користувачів послугами бібліотеки.

2.3 Інструменти для розробки інформаційних систем

Інструменти для розробки інформаційних систем охоплюють різноманітні технології, які допомагають створювати, підтримувати та тестувати інформаційні системи. Вибір інструментів залежить від специфіки проєкту, цілей і вимог користувачів.

1. Мови програмування: Для розробки серверної та клієнтської частин інформаційних систем використовуються різноманітні мови програмування. Python є популярним вибором для серверних частин завдяки своїй простоті та потужності в обробці даних. Для веб-додатків також використовуються JavaScript (особливо в поєднанні з фреймворками React або Angular) для створення динамічних та інтерактивних інтерфейсів.
2. Бази даних: Інформаційні системи потребують надійної бази даних для зберігання і обробки великих обсягів інформації. Реляційні бази даних, такі як MySQL, PostgreSQL або Oracle, використовуються для зберігання структурованих даних, що включають замовлення, книги та користувачів. Для деяких специфічних випадків можуть

використовуватися NoSQL бази, такі як MongoDB, для зберігання неструктурованих даних.

3. Фреймворки: Фреймворки значно прискорюють розробку, оскільки вони надають готові рішення для створення веб-додатків. Django та Flask є популярними фреймворками для Python, які забезпечують все необхідне для створення серверної частини, роботи з базами даних і створення API. Для клієнтської частини широко використовуються фреймворки на основі JavaScript, такі як React або Angular, які дозволяють створювати складні та інтерактивні інтерфейси.

4. Інструменти тестування: Важливою частиною процесу розробки є тестування. Інструменти, такі як Postman, дозволяють тестувати API-запити та перевіряти правильність взаємодії між сервером і клієнтом. Крім того, для автоматичного тестування використовуються фреймворки на кшталт Selenium, який дозволяє автоматизувати тестування користувацьких інтерфейсів.

5. Інструменти для контролю версій: Для ефективної роботи в команді важливо мати систему контролю версій, яка дозволяє відслідковувати зміни в коді і співпрацювати над проектом. Git є найпопулярнішим інструментом для цього, а платформи на кшталт GitHub або GitLab дозволяють управляти репозиторіями та інтегрувати різні інструменти для автоматичного тестування та деплойменту.

Використання цих інструментів забезпечує ефективність, швидкість і надійність при розробці інформаційних систем, таких як бібліотечні системи, що автоматизують процеси управління та взаємодії з користувачами.

РОЗДІЛ 3

РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ЗАМОВЛЕНЬ У БІБЛІОТЕЦІ

3.1 Архітектура та загальна характеристика системи

Архітектура інформаційної системи є основною складовою при розробці ефективної та масштабованої бібліотечної системи замовлень. Вона визначає, як різні компоненти системи взаємодіють між собою для забезпечення функціональності та досягнення поставлених цілей. Загальна характеристика системи охоплює декілька ключових аспектів:

1. Клієнт-серверна архітектура: Система використовує класичну клієнт-серверну архітектуру, де сервер виконує обробку даних і управління базою даних, а клієнт є інтерфейсом для взаємодії користувачів із системою. Клієнт (через веб-інтерфейс або додаток) відправляє запити до сервера, який обробляє їх, звертаючись до бази даних і повертаючи результати клієнту.
2. База даних: Основним компонентом є централізована база даних, що зберігає всі необхідні дані про книги, користувачів і їхні замовлення. База даних зазвичай використовує реляційну модель, що дозволяє організувати ефективне збереження і пошук даних, таких як інформація про книги, авторів, категорії тощо. Для забезпечення високої доступності та безпеки даних використовуються резервні копії та механізми відновлення.
3. Фронтенд (Клієнтська частина): Фронтенд забезпечує користувачам доступ до системи через веб-інтерфейс або спеціалізовані додатки. Він відповідає за відображення даних і взаємодію з користувачами, включаючи введення запитів на пошук книжок, оформлення замовлень і перегляд історії запитів. Використовуються сучасні фреймворки JavaScript (React, Angular) для створення динамічних і зручних інтерфейсів.

4. Бекенд (Серверна частина): Бекенд обробляє запити клієнтів, виконує логіку роботи з базою даних і повертає результати користувачам. Бекенд-частина містить функції для аутентифікації користувачів, управління замовленнями, перевірки доступності книг, а також для обробки будь-яких адміністративних функцій, таких як додавання нових книжок або оновлення статусів замовлень. Для розробки бекенду часто використовуються фреймворки Python (Django або Flask).

5. Інтеграція з іншими системами: Важливим аспектом є інтеграція з іншими системами, такими як платіжні шлюзи для оплати замовлень або зовнішні каталоги книг для оновлення даних про наявні видання. Інтеграція забезпечує гнучкість і масштабованість системи.

6. Безпека: Забезпечення безпеки є критичним аспектом, особливо при обробці персональних даних користувачів. Для цього використовуються методи шифрування даних, захист від SQL-ін'єкцій, а також аутентифікація та авторизація користувачів через безпечні протоколи, такі як HTTPS.

Загалом, система має бути масштабованою, забезпечуючи підтримку великої кількості користувачів та можливість зберігання та обробки великих обсягів даних. Вона також повинна бути адаптованою до змін і оновлень, щоб задовольняти нові потреби користувачів і враховувати зміни в бібліотечному процесі.

3.2 Розробка серверної частини

Розробка серверної частини бібліотечної системи є ключовим етапом у створенні функціональної та ефективної інформаційної системи для управління замовленнями книг. Серверна частина відповідає за обробку запитів від клієнтів (користувачів або адміністратора), виконання логіки додатку, а також взаємодію з базою даних. Основні етапи розробки серверної частини включають:

1. Вибір технологій для бекенду

Мова програмування та фреймворк: Вибір технології для реалізації серверної частини залежить від вимог до продуктивності, масштабованості та зручності в розробці. Для цього зазвичай використовуються мови програмування, такі як Python (з фреймворками Django або Flask), Node.js, Ruby (Ruby on Rails), або Java (Spring).

База даних: Для зберігання даних про книги, користувачів та замовлення використовується реляційна база даних (наприклад, MySQL, PostgreSQL) або NoSQL рішення (MongoDB), в залежності від специфіки проекту.

2. Розробка API (Application Programming Interface)

RESTful API: Серверна частина повинна забезпечувати клієнтську частину доступом до даних через API. Одним з популярних підходів є розробка RESTful API, яке використовує HTTP-методи (GET, POST, PUT, DELETE) для виконання операцій над ресурсами (наприклад, додавання книги, оформлення замовлення, перевірка доступності книги).

Авторизація та аутентифікація: Для забезпечення безпеки важливо реалізувати механізм аутентифікації користувачів. Для цього можна використовувати токенізацію (JWT — JSON Web Tokens) або традиційні сесії. Користувачі мають право доступу до конкретних ресурсів, таких як список доступних книг або статус замовлення, відповідно до рівня їх авторизації (користувач, адміністратор).

3. Обробка запитів до бази даних

ORM (Object-Relational Mapping): Для зручної роботи з базою даних використовується ORM, яке дозволяє взаємодіяти з базою даних через об'єкти, не пишучи SQL-запити вручну. Наприклад, Django використовує свій ORM для зручної роботи з моделями даних (наприклад, таблицями книг або замовлень).

CRUD операції: Серверна частина реалізує стандартні CRUD операції (Create, Read, Update, Delete) для управління даними в базі. Це включає створення нових замовлень, оновлення статусу книг, видалення записів тощо.

4. Обробка бізнес-логіки

Перевірка доступності книг: Серверна частина повинна виконувати бізнес-логіку перевірки наявності книг у бібліотеці при оформленні замовлення, обчислення штрафів за прострочені замовлення, управління чергою на популярні книги тощо.

Управління історією замовлень: Потрібно створити механізми для відстеження історії замовлень користувачів, що дозволяє їм переглядати поточні та завершені замовлення, а також отримувати інформацію про статус.

5. Тестування серверної частини

Юніт-тести: Для перевірки функціональності кожного модуля серверної частини розробляються юніт-тести. Наприклад, перевірка правильності обробки замовлень або точності обчислень штрафів.

Інтеграційні тести: Перевіряються взаємодії між різними модулями, такими як база даних, API, і клієнтська частина.

Тестування навантаження: Оскільки система повинна підтримувати роботу великої кількості одночасних користувачів, важливо перевірити, як серверна частина витримує навантаження.

6. Розгортання серверної частини

Хостинг і сервер: Для запуску серверної частини потрібно вибрати хостинг або серверну платформу. Це може бути локальний сервер або хмарні рішення, такі як Amazon Web Services (AWS), Google Cloud або Heroku.

Моніторинг та логування: Після розгортання сервера важливо налаштувати систему моніторингу та логування для відстеження проблем і продуктивності системи.

Розробка серверної частини бібліотечної системи замовлень є ключовим етапом, оскільки від неї залежить не лише збереження даних і обробка запитів, але й загальна ефективність і безпека роботи всієї системи.

3.3 Розробка клієнтської частини

Розробка клієнтської частини бібліотечної системи замовлень передбачає створення інтерфейсу, через який користувачі (клієнти) взаємодіють з серверною частиною системи. Цей етап включає проектування та реалізацію компонентів, що відповідають за взаємодію з користувачем, забезпечуючи зручний та інтуїтивно зрозумілий досвід для відвідувачів бібліотеки.

- Вибір технологій для клієнтської частини

HTML/CSS: Основою для розробки інтерфейсу є HTML для структури сторінок та CSS для їх оформлення. За допомогою CSS можна створювати адаптивні та привабливі інтерфейси, що забезпечують комфортне використання на різних пристроях.

JavaScript: Для інтерактивності та динамічної взаємодії з сервером використовується JavaScript. За допомогою цього мови програмування можна реалізувати функціонал, наприклад, для обробки форм замовлення книг, перевірки наявності книг в бібліотеці в реальному часі, або для показу повідомлень про статус замовлення.

Фреймворки для JavaScript: Для полегшення розробки часто використовуються популярні бібліотеки та фреймворки, такі як React, Vue.js або Angular, які дозволяють створювати складні односторінкові додатки (SPA). Вони дають змогу організувати фронтенд на основі компонентів, що спрощує масштабування та підтримку коду.

- Функціональні можливості клієнтської частини

Авторизація та реєстрація користувачів: Клієнтська частина повинна включати форми для реєстрації нових користувачів та авторизації існуючих. Користувачі повинні мати можливість увійти в систему, щоб отримати доступ до своїх замовлень та особистих даних.

Перегляд доступних книг: Користувачі повинні мати можливість переглядати каталог книг, а також перевіряти їх доступність. Це може бути

реалізовано через таблиці або картки з інформацією про книгу, зображенням, описом, автором та іншими деталями.

Оформлення замовлень: Інтерфейс повинен дозволяти користувачам додавати книги до кошика, вибирати кількість екземплярів, вказувати необхідні дати та оформляти замовлення. Також потрібно реалізувати механізм для скасування або редагування замовлення.

Перегляд статусу замовлень: Після оформлення замовлення користувач повинен мати змогу відстежувати його статус (наприклад, "в обробці", "затверджено", "відправлено"). Це може бути реалізовано через персональну сторінку користувача або окремий розділ.

- Валідація та обробка даних

Валідація вводу: Для забезпечення коректності даних, що вводяться користувачами (наприклад, імені, контактних даних, дат), важливо реалізувати валідацію на клієнтському боці. Це дозволяє вчасно попередити користувача про помилки до того, як форма буде надіслана на сервер.

Обробка помилок: У разі неправильного введення даних або виникнення помилки на сервері, клієнтська частина повинна вивести повідомлення про помилку для користувача, щоб він міг виправити ситуацію.

- Зв'язок з серверною частиною

HTTP-запити: Клієнтська частина використовує HTTP-запити (через AJAX або Fetch API) для взаємодії з сервером. Це дозволяє здійснювати операції, такі як отримання даних про книги, створення нового замовлення або оновлення статусу замовлення.

Обробка відповіді від сервера: Після відправки запиту сервер повертає відповідь, яка містить необхідні дані або повідомлення про статус виконання запиту. Клієнтська частина повинна обробити цю відповідь і відповідно оновити інтерфейс (наприклад, відобразити нове замовлення або показати повідомлення про помилку).

- Інтерфейс користувача

Дизайн і зручність: Важливо створити інтуїтивно зрозумілий і привабливий інтерфейс, який буде зручним для користувачів різного рівня технічної підготовки. Це може включати мінімалістичний дизайн, чітке меню навігації, кнопки для швидкого доступу до основних функцій системи.

Адаптивність: Інтерфейс має бути адаптивним, тобто він повинен коректно відображатися на різних пристроях (комп'ютерах, планшетах, смартфонах). Це досягається за допомогою медіа-запитів у CSS або фреймворків, таких як Bootstrap.

- Тестування клієнтської частини

Юзабіліті-тести: Для перевірки зручності використання інтерфейсу важливо провести юзабіліті-тести з реальними користувачами. Це дозволяє виявити слабкі місця у дизайні та покращити взаємодію з системою.

Тестування інтерактивності: Оскільки клієнтська частина повинна бути інтерактивною, необхідно перевірити, чи правильно працюють всі елементи на сторінці (форми, кнопки, панелі навігації).

Розробка клієнтської частини є важливим етапом у створенні бібліотечної системи замовлень, оскільки саме через цей інтерфейс користувачі взаємодіють з системою. Тому важливо, щоб інтерфейс був функціональним, зручним і приємним у використанні.

3.4 Тестування розробленої системи

Тестування розробленої системи бібліотеки є важливим етапом для забезпечення її функціональності, безпеки та зручності користувачів. Оскільки система призначена для управління замовленнями книг та іншими бібліотечними процесами, необхідно ретельно перевірити всі аспекти її роботи. Ось основні етапи, які повинні бути виконані при тестуванні бібліотечної системи замовлень:

I. Функціональне тестування

- Перевірте основні функції системи, такі як реєстрація користувачів, авторизація, пошук книг, оформлення та скасування замовлень.
- Важливо перевірити, чи коректно обробляються всі типи запитів користувачів, чи відображається інформація про доступні книги, чи оновлюється статус замовлення, чи виконується пошук по запитах.
- Перевірте, чи система правильно працює з різними категоріями користувачів (адміністратор, бібліотекар, звичайний користувач).
- Тестування продуктивності
- Перевірте, як система працює під високим навантаженням, коли одночасно користуються кілька десятків або сотень користувачів. Це може бути перевірено шляхом стрес-тестування.
- Важливо оцінити швидкість обробки запитів, час відгуку серверів, а також продуктивність бази даних при великому обсязі даних.
- Тестування безпеки
- Перевірте захищеність даних користувачів. Для цього необхідно протестувати систему на вразливості, зокрема:
- Чи захищені персональні дані користувачів (паролі, адреси доставки).
- Чи застосовується шифрування даних, що зберігаються в базі даних.
- Чи є система захисту від SQL-ін'єкцій та інших видів атак.

II. Тестування зручності користування (юзабіліті)

- Оцінка зручності інтерфейсу є важливою для того, щоб користувачі могли без проблем взаємодіяти з системою.
- Проведіть юзабіліті-тестування, залучивши реальних користувачів для оцінки легкості пошуку книг, оформлення замовлень, навігації по системі.
- Важливо переконатися, що інтерфейс є інтуїтивно зрозумілим, а процес замовлення не викликає складнощів.

III. Тестування інтеграції

- Тестуйте інтеграцію між різними компонентами системи: інтерфейс користувача, сервер, база даних.
- Переконайтесь, що запити від користувачів коректно обробляються сервером і дані правильно передаються в базу даних.

IV. Тестування сумісності

- Перевірте, як система працює на різних пристроях та у різних браузерах, щоб переконатися, що вона підтримує різні платформи, зокрема мобільні та десктопні версії.
- Проведіть тестування на популярних браузерах, таких як Google Chrome, Mozilla Firefox, Safari, а також на мобільних пристроях.

V. Реєстрація та аналіз помилок

- Під час тестування реєструйте всі помилки, які виникають, і систематично їх виправляйте.
- Важливо створювати звіти про помилки та надавати їх розробникам для виправлення.

VI. Регресійне тестування

- Після виправлення помилок або внесення змін перевірте, чи не виникли нові проблеми в раніше перевірених функціях. Це дозволяє переконатися, що зміни не порушили роботу системи в інших частинах.

VII. Формування звітів і документування результатів тестування

- Ретельно документуйте результати тестування, включаючи виявлені помилки, вдалі тести і зауваження користувачів.
- Ці звіти будуть корисні для подальшого вдосконалення системи.

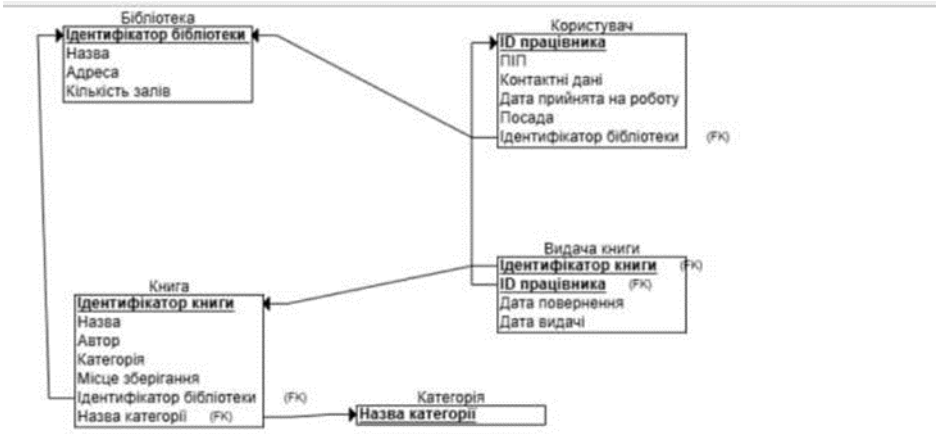

Рисунок 3.1 - Реляційна схема бібліотеки

1. Таблиця "Бібліотека"
    - Призначення: Містить загальну інформацію про бібліотеки.
    - Атрибути:
        - Ідентифікатор бібліотеки (Primary Key): Унікальний ідентифікатор для кожної бібліотеки.
        - Назва: Назва бібліотеки.
        - Адреса: Локація бібліотеки.
        - Кількість залів: Інформація про інфраструктуру бібліотеки.
    - Зв’язки:
        - Зв’язана із таблицями "Книга" та "Користувач".
2. Таблиця "Книга"
    - Призначення: Зберігає інформацію про всі книги, доступні в бібліотеці.
    - Атрибути:
        - Ідентифікатор книги (Primary Key): Унікальний номер кожної книги.
        - Назва: Назва книги.
        - Автор: Автор книги.
        - Категорія: Тип книги (художня, наукова тощо).
        - Місце зберігання: Локація книги у бібліотеці.

        - Ідентифікатор бібліотеки (Foreign Key): Зв’язок з таблицею "Бібліотека".
    - Зв’язки:
        - Пов’язана з таблицею "Видача книги" та "Категорія".
3. Таблиця "Категорія"
    - Призначення: Зберігає перелік категорій книг.
    - Атрибути:
        - Назва категорії (Primary Key): Назва жанру чи типу книг.
    - Зв’язки:
        - Використовується у таблиці "Книга".
4. Таблиця "Користувач"
    - Призначення: Зберігає інформацію про бібліотечних працівників чи користувачів.
    - Атрибути:
        - ID працівника (Primary Key): Унікальний ідентифікатор користувача чи працівника.
        - ПІП: Прізвище, ім’я, по батькові.
        - Контактні дані: Телефон чи email.
        - Дата прийняття на роботу: Для працівників.
        - Посада: Для працівників (наприклад, бібліотекар, адміністратор).
        - Ідентифікатор бібліотеки (Foreign Key): Зв’язок із бібліотекою, в якій працює користувач.
    - Зв’язки:
        - Зв’язана з таблицею "Видача книги".
5. Таблиця "Видача книги"
    - Призначення: Містить інформацію про книги, видані користувачам.
    - Атрибути:
        - Ідентифікатор книги (Foreign Key): Посилання на конкретну книгу.

- ID працівника (Foreign Key): Посилання на працівника чи користувача.
- Дата видачі: Дата, коли книга була видана.
- Дата повернення: Очікувана або фактична дата повернення книги.

- Зв’язки:
  - Зв’язана з таблицями "Книга" та "Користувач".

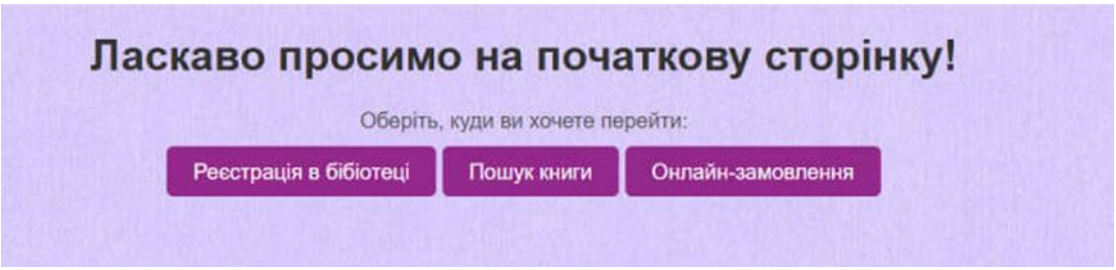

Рисунок 3.2 – Початкова сторінка сайту

На головній сторінці ми бачимо лише три кнопки, але це на цей момент і є самі основні.Коли ви вибираєте реєстрацію ми можемо або сам користувач зареєструватися, щоб в майбутньому був невеличкий відсоток знижки або мав карту постійного читача і міг перший дізнаватися про всі нові книги.Наступна кнопка меню відповідає за пошук книги там на цей момент є тільки п'ять книг але вони популярні можна побачити скільки є книг, а також забронювати якщо ви боїтеся що її хтось купить.Якщо книги немає в наявності навіть коли користувач натиснути забронювати йому вискочить повідомлення що бронювати не можна.Третя кнопка відповідає за онлайн замовлення.

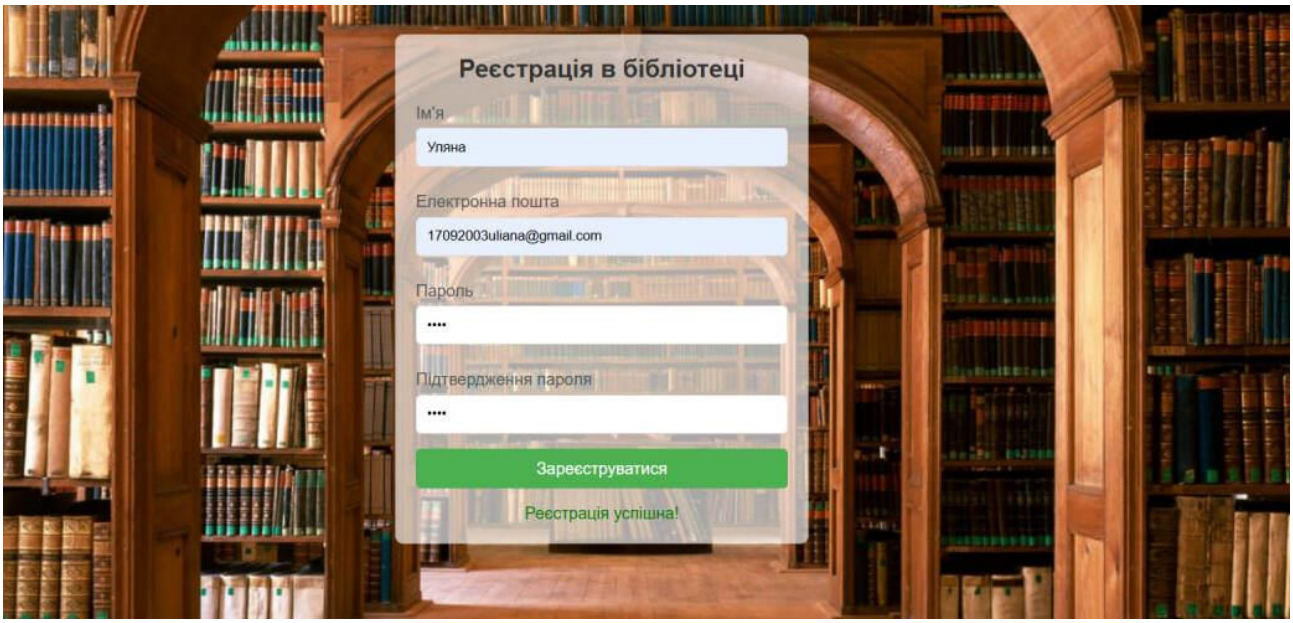

Рисунок 3.3 – Реєстрація в бібліотеці

Використовується стандартна структура HTML для створення форми. Основні елементи включають:

- Текстові поля для введення імені, електронної пошти, пароля та підтвердження пароля.
- Кнопка для надсилання форми ("Зареєструватися").
- Повідомлення про успіх після реєстрації.

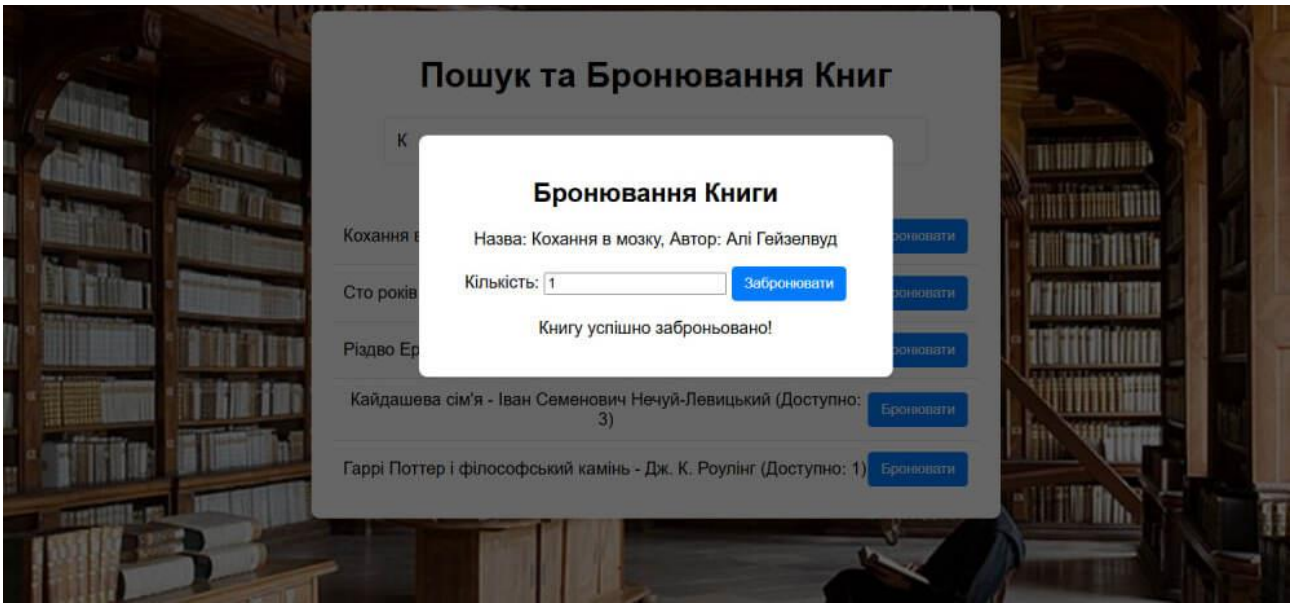

Рисунок 3.4 – Пошук та бронювання книги

На зображенні показаний інтерфейс функції бронювання книг у бібліотечній інформаційній системі. Ось основні етапи, які були реалізовані, і що відбувається на кожному з них:

1. Інтерфейс пошуку книг :
    - Верхня частина вікна містить поле для пошуку книги за назвою, що дозволяє користувачам швидко знайти потрібну книгу з доступного каталогу.
    - Після введення запиту в пошукове поле відображатиметься список книг із відповідними назвами, авторами, кількістю доступних екземплярів та кнопкою "Бронювати".
2. Форма підтвердження бронювання :
    - Коли користувач натискає кнопку "Бронювати", відкривається модальне вікно, яке підтверджує обраний запит.
    - У цьому вікні виводиться:
        - Назва книги (наприклад, "Кохання в мозку").
        - Автор книги.
        - Поле для введення кількості примірників, які користувач хоче забронювати.
    - Користувач може змінити кількість примірників у відповідному полі, що дозволяє здійснити гнучку бронювання залежно від наявності.
3. Підтвердження успішного бронювання :
    - Після натискання кнопки "Забронювати" в модальному вікні системи здійснюється перевірка доступності вибраного елемента примірників.
    - Якщо бронювання успішне, з'являється повідомлення "Книгу успішно заброньовано!".
    - Успішне завершення бронювання означає, що інформація передається до баз даних, де оновлюється статус книги (кількість доступних екземплярів зменшується).

Технічна реалізація:

1. Фронтенд :
    - Використовується HTML/CSS для структури та стилізації форм.
    - JavaScript або інший фреймворк (React/Vue.js) для динамічного відображення та інтерактивності модального вікна.
2. Бекенд :
    - Реалізовано API-метод для обробки запиту бронювання:
        - Перевіряється кількість доступних примірників у базі даних.
        - Внесені зміни в запис бази даних після успішного бронювання.
3. База даних :
    - Таблиця книг зберігає інформацію про доступність (кількість).
    - Таблиця користувачів/бронювання записує, хто, скільки і скільки екземплярів забронював.

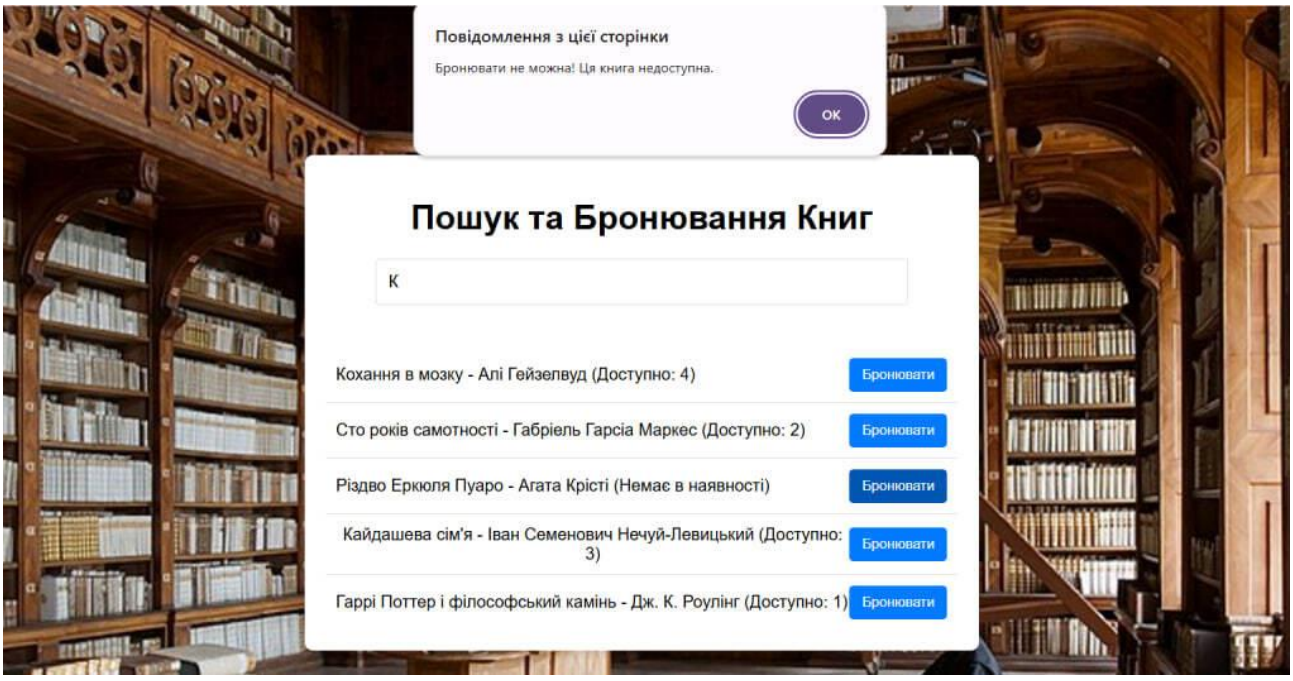

Рисунок 3.5 – Якщо користувач хоче забронювати книгу якої немає

Книги немає в наявності і забронювати в даний момент її не можна.

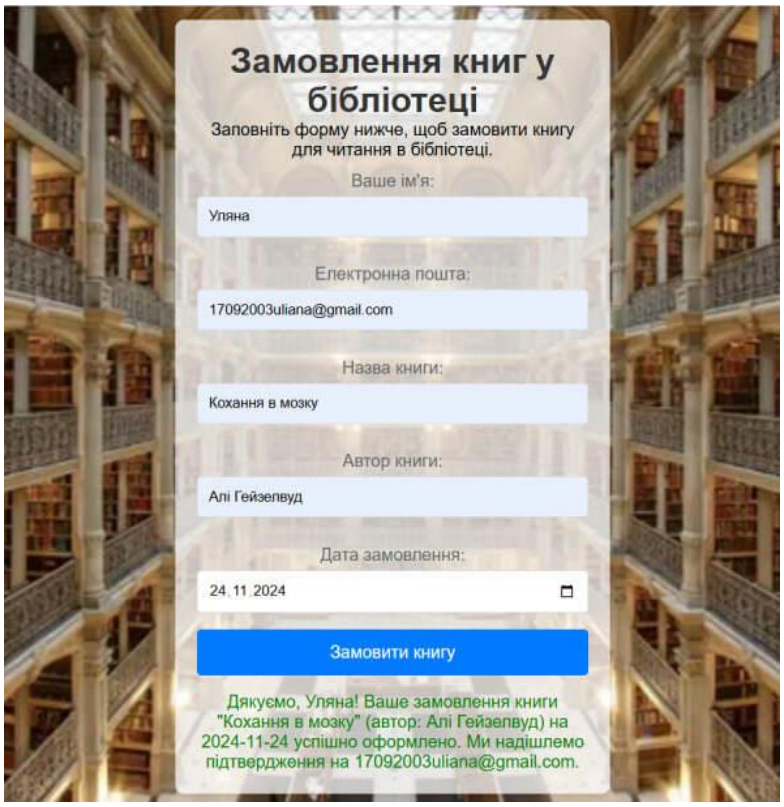

Рисунок 3.6 – Оформлення замовлення у бібліотеці

Ви заповнюєте форму для замовлення книги в бібліотеці. Ось що ви робите:

- Заповнення форми: Ви вводите своє ім'я ("Уляна") та електронну пошту ("17092003uliana@gmail.com").
- Вказуєте назву книги ("Кохання в мозку") та автора ("Ані Гейзельвуд").
- Вибираєте дату замовлення ("24.11.2024").
- Підтвердження замовлення: Після натискання кнопки "Замовити книгу" система підтверджує, що замовлення було успішно оформлене. З'являється повідомлення з подякою, вказується деталі книги (назва та автор), а також дата замовлення. Окрім того, вам надійде підтвердження на вказану електронну пошту.

Цей процес дозволяє користувачам зручно замовляти книги для читання в бібліотеці через онлайн-форму.

ВИСНОВКИ

У результаті дослідження було проаналізовано вимоги до бібліотечних інформаційних систем, визначено ключові функції, необхідні для забезпечення автоматизації процесів у бібліотеці, зокрема реєстрація користувачів, пошук книг і оформлення замовлень.

В основі розробки лежить використання сучасних веб-технологій, таких як HTML, CSS та JavaScript, які забезпечили ефективність і гнучкість реалізації. Застосовано адаптивний дизайн, що дозволяє користувачам комфортно працювати із системою на різних пристроях, включаючи мобільні телефони, планшети та комп’ютери. Створено багатосторінковий веб-додаток, який інтегрує різні функції: реєстрація дозволяє бібліотеці зберігати інформацію про користувачів, пошук книг здійснюється за ключовими словами або автором, а онлайн-замовлення забезпечує зручний доступ до бібліотечних ресурсів.

Особливу увагу приділено інтерфейсу користувача, який реалізовано інтуїтивно зрозумілим і дружнім до користувача. Система розроблена з урахуванням можливостей її подальшого розширення. У майбутньому можлива інтеграція з серверною частиною та базами даних для зберігання інформації про книги, користувачів і замовлення, що зробить систему більш потужною та ефективною.

Таким чином, розроблена інформаційна система є важливим кроком у напрямку автоматизації бібліотечних процесів, сприяє підвищенню зручності обслуговування користувачів та оптимізації роботи бібліотекарів. Вона демонструє потенціал для використання в реальних умовах, а також може бути вдосконалена з урахуванням потреб конкретної бібліотеки або установи.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Ісаєнко О.О., Стамбол І.І. Сучасні інформаційні системи та технології в бібліотечній діяльності та архівах. Навчальний ресурс Київського університету імені Бориса Грінченка.

2. Державна бібліотека України для юнацтва. Інформаційні технології у бібліотеках.http://4uth.gov.ua/bibliotekar-rekomenduye/nashi-vydannya/#

3. Журнал "Бібліотечний вісник". Різні випуски присвячені автоматизації процесів у бібліотеках та використанню сучасних IT-інструментів у цій сфері.

4. "Інформаційні технології в бібліотечній справі: методи та підходи". Навчальний посібник, Київ, 2021.

5. Бібліотека імені Вернадського (НАН України). Річний звіт 2023 року, розділ про впровадження електронних каталогів та замовлення.

6. Політехніка Львів. "Розробка веб-додатків для бібліотек". URL: [https://directory.lpnu.ua/majors/subject/ikni/6.122.00.12/8/2024/ua/full/3/21006]

7. Київський університет імені Тараса Шевченка. Практичні дослідження інформаційних систем замовлення в бібліотеках.

ДОДАТОК А:

Початкова сторінка

```
<!DOCTYPE html> <html lang="uk"> <head> <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Початкова сторінка</title>
    <link rel="stylesheet" href="style2.css/style2.css"> </head>  <body>
 <h1>Ласкаво просимо на початкову сторінку!</h1>
    <p>Оберіть, куди ви хочете перейти:</p>
    <a href="index.html" class="button">Реєстрація в бібіотеці</a>
    <a href="c:\Users\Ulyana\Desktop\пошук книг\index2.html"
class="button">Пошук книги</a>
    <a href="c:\Users\Ulyana\Desktop\онлайн-замовлення\index.html"
class="button">Онлайн-замовлення</a>
</body> </html>
```

Реєстрація в бібліотеці

```
<!DOCTYPE html> <html lang="uk"> <head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Реєстрація в бібліотеці</title>
  <link rel="stylesheet" href="style.css/style.css"> </head> <body>
```

```
<div class="registration-container"> <h2>Реєстрація в бібліотеці</h2>

<form id="registrationForm"> <label for="name">Ім'я</label> <input type="text" id="name" name="name" required>

<label for="email">Електронна пошта</label>

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

<input type="password" id="password" name="password" required>

<label for="confirmPassword">Підтвердження пароля</label>

<input type="password" id="confirmPassword" name="confirmPassword" required>

<button type="submit">Зареєструватися</button>

<p id="message"></p> </form> </div> <script src="script.js"></script> </body> </html>
```

Пошук книги

```
<!DOCTYPE html> <html lang="uk"> <head> <meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Пошук та Бронювання Книг</title> <link rel="stylesheet" href="style.css/style.css"> </head> <body>

<div class="search-container"> <h1>Пошук та Бронювання книг</h1>

<input type="text" id="searchInput" placeholder="Введіть назву книги..." onkeyup="searchBooks()">
```

×

<h2>Бронювання Книги</h2> <label for="quantity">Кількість:</label>

<input type="number" id="quantity" min="1" value="1">

<button onclick="reserveBook()">Забронювати</button>   <script src="script.js"></script> </body> </html>

Онлайн-замовлення книг

<!DOCTYPE html> <html lang="uk"> <head> <meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Онлайн-замовлення книг</title> <link rel="stylesheet" href="styles.css/style.css"> </head> <body> <h1>Замовлення книг у бібліотеці</h1>

Заповніть форму нижче, щоб замовити книгу для читання в бібліотеці. <form id="order-form">

<label for="name">Ваше ім'я:</label> <input type="text" id="name" name="name" required>

<label for="email">Електронна пошта:</label> <input type="email" id="email" name="email" required>

<label for="book-title">Назва книги:</label> <input type="text" id="book-title" name="book-title" required>

<label for="author">Автор книги:</label> <input type="text" id="author" name="author" required>

<label for="pickup-date">Дата замовлення:</label> <input type="date" id="pickup-date" name="pickup-date" required> <button type="submit">Замовити книгу</button> </form>

  <script src="scripts.js"></script> </body> </html>

SCRIPT SRC

Реєстрація

```
document.getElementById("registrationForm").addEventListener("submit",
function(event) {
    event.preventDefault();
    const name = document.getElementById("name").value;
    const email = document.getElementById("email").value;
    const password = document.getElementById("password").value;
    const confirmPassword = document.getElementById("confirmPassword").value;
    const message = document.getElementById("message");
    if (password !== confirmPassword) {
      message.textContent = "Паролі не співпадають.";
      return;
    }
    message.style.color = "green";
    message.textContent = "Реєстрація успішна!";
  });
```

Пошук книги

```
const books = [
  { title: "Кохання в мозку", author: "Алі Гейзелвуд", available: 5 }, // Додано поле
available
  { title: "Сто років самотності", author: "Габріель Гарсіа Маркес", available: 2 },
  { title: "Різдво Еркюля Пуаро", author: "Агата Крісті", available: 0 },
```

```
  { title: "Кайдашева сім'я", author: "Іван Семенович Нечуй-Левицький",
available: 3 },
  { title: "Гаррі Поттер і філософський камінь", author: "Дж. К. Роулінг",
available: 1 },
];
let selectedBook = null;
function searchBooks() {
  const input = document.getElementById("searchInput").value.toLowerCase();
  const resultsDiv = document.getElementById("results");
  resultsDiv.innerHTML = "";
  const filteredBooks = books.filter(book =>
    book.title.toLowerCase().includes(input) ||
book.author.toLowerCase().includes(input)
  );
  if (filteredBooks.length > 0) {
    filteredBooks.forEach(book => {
      const bookItem = document.createElement("div");
      bookItem.classList.add("book-item");
      const bookInfo = document.createElement("span");
      bookInfo.textContent = `${book.title} - ${book.author} (${book.available > 0 ?
`Доступно: ${book.available}` : 'Немає в наявності'})`;
      const reserveButton = document.createElement("button");
      reserveButton.textContent = "Бронювати";
      reserveButton.onclick = () => handleReservation(book);
      bookItem.appendChild(bookInfo);
      bookItem.appendChild(reserveButton);
      resultsDiv.appendChild(bookItem);
    });
  } else {
    resultsDiv.innerHTML = "<p>Нічого не знайдено</p>";
  }
```

```
}
function handleReservation(book) {
  if (book.available === 0) {
    alert("Бронювати не можна! Ця книга недоступна.");
  } else {
    openModal(book);
  }
}
function openModal(book) {
  selectedBook = book;
  document.getElementById("bookDetails").textContent = `Назва: ${book.title}, Автор: ${book.author}`;
  document.getElementById("quantity").value = 1;
  document.getElementById("reservationMessage").textContent = "";
  document.getElementById("reservationModal").classList.remove("hidden");
}
function closeModal() {
  document.getElementById("reservationModal").classList.add("hidden");
}
function reserveBook() {
  const quantity = parseInt(document.getElementById("quantity").value, 10);
  if (quantity > selectedBook.available) {
    document.getElementById("reservationMessage").textContent = "Недостатньо книг у наявності!";
    return;
  }
  selectedBook.available -= quantity;
  document.getElementById("reservationMessage").textContent = "Книгу успішно заброньовано!";
  setTimeout(() => {
    closeModal();
```

```
    searchBooks();
  }, 2000);
}
```

Онлайн-замовлення

```
document.getElementById('order-form').addEventListener('submit', function(event) {
    event.preventDefault();
    const name = document.getElementById('name').value;
    const email = document.getElementById('email').value;
    const bookTitle = document.getElementById('book-title').value;
    const author = document.getElementById('author').value;
    const pickupDate = document.getElementById('pickup-date').value;
    document.getElementById('message').innerText =
        `Дякуємо, ${name}! Ваше замовлення книги "${bookTitle}" (автор:
${author}) на ${pickupDate} успішно оформлено. Ми надішлемо підтвердження
на ${email}.`;
});
```