

## ДОВІДКА

### про перевірку курсової роботи (проекту) на наявність плагіату

Відділення: Автоматизації та залізничного транспорту

Спеціальність: 5.123.1 Комп'ютерна інженерія

Освітньо-професійна програма: 123 Обслуговування комп'ютерних систем і мереж

Дисципліна: Програмування

Автор роботи: Харчов Юрій Юрійович

Назва роботи: Розробка програмного додатку для рекомендацій книг.

Керівник: Васильєв Микола Васильович

Обсяг матеріалів, сторінок 57

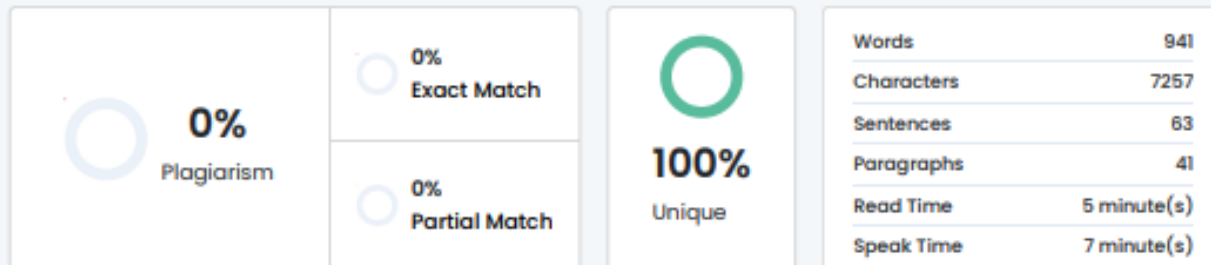
Обсяг матеріалів, який перевірявся на оригінальність, сторінок 38

Результати перевірки: робота перевірена програмою <https://www.duplichecker.com>, відсоток унікальності становить 99%.

Студент

Юрій ХАРЧОВ

## Plagiarism Scan Report



## Content Checked For Plagiarism

## 1. ВСТУП

Останнім часом інформаційні технології стали невід'ємною частиною повсякденного життя, і їх застосування охоплює всі сфери діяльності. Усе більше людей використовує програмне забезпечення для пошуку, вибору та управління інформацією, зокрема у сфері літератури. Сучасні цифрові рішення дозволяють автоматизувати процеси підбору книг відповідно до вподобань користувачів, що значно покращує їхній досвід читання.

Метою цього проекту є розробка програмного додатку для рекомендації книг, який дозволяє користувачам зручно переглядати, додавати та редагувати список книг, а також отримувати персоналізовані рекомендації на основі жанру, рейтингу та інших параметрів. Додаток буде використовувати базу даних для збереження інформації про книги та алгоритм відбору для формування рекомендацій.

Актуальність цього проекту зумовлена зростаючою популярністю цифрових бібліотек і необхідністю швидкого доступу до якісних літературних творів. Використання бази даних для збереження та управління інформацією дозволить забезпечити зручність роботи з великим обсягом даних і надійність доступу.

Метою даного проекту є створення ефективного інструменту для автоматизації процесу підбору книг, що допоможе користувачам знайти найкращі варіанти відповідно до їхніх уподобань, спростить процес вибору літератури та підвищить загальну якість взаємодії з книгами.

## 2. АНАЛІЗ ЗАДАЧІ, ЗАСОБІВ ТА МЕТОДІВ ЇЇ ВИРІШЕННЯ

В умовах стрімкого розвитку інформаційних технологій одним із ключових аспектів сучасного цифрового середовища є забезпечення ефективного доступу до інформації. У сфері літератури та книжкових рекомендацій користувачі часто стикаються з проблемою вибору серед великої кількості доступних книг. Тому актуальним є створення програмного забезпечення, яке дозволить автоматизувати процес підбору книг та надавати персоналізовані рекомендації.

Основною задачею даного проекту є розробка програмного додатку, що надасть можливість користувачам отримувати рекомендації книг на основі їхніх уподобань, зберігати інформацію про книги та редагувати власну бібліотеку. Додаток буде працювати з базою даних, у якій зберігатимуться відомості про книги, їх жанри, рейтинги та інші параметри.

Основні вимоги до програмного забезпечення:

- Збереження та організація інформації про книги.
- Можливість додавання, редагування та видалення записів у бібліотеці.
- Алгоритм формування рекомендацій на основі жанру, рейтингу або інших параметрів.
- Зручний та інтуїтивно зрозумілий графічний інтерфейс.
- Надійний та швидкий доступ до бази даних.

Засоби реалізації: для розробки програмного продукту будуть використані сучасні технології та програмні засоби:

- Мова програмування Python – забезпечить гнучкість у розробці додатку та його взаємодії з базою даних.
- Tkinter – бібліотека для створення графічного інтерфейсу, що дозволить користувачам зручно працювати з додатком.
- База даних SQL Server – для збереження інформації про книги та користувачів, що забезпечить швидкий доступ

до даних та їхню структурованість.

- `pyodbc` – бібліотека для інтеграції Python із SQL Server, яка дозволить ефективно обробляти запити до бази даних.

Методи вирішення задачі:

1. Модульна архітектура – розробка окремих модулів для роботи з базою даних, формування рекомендацій та управління графічним інтерфейсом.
2. Графічний інтерфейс – створення інтуїтивно зрозумілого GUI, що забезпечить простоту взаємодії з додатком.
3. Розробка алгоритму рекомендацій – використання фільтрації за жанром, рейтингом та іншими критеріями для надання персоналізованих рекомендацій.
4. Оптимізація роботи з базою даних – забезпечення швидкої обробки запитів для ефективного доступу до інформації про книги.

Запропонований додаток дозволить користувачам легко знаходити книги відповідно до своїх інтересів, автоматизуючи процес вибору літератури та підвищуючи зручність використання цифрової бібліотеки.

## 2.1 Python

Python є однією з найпопулярніших мов програмування завдяки своїй простоті, гнучкості та широкому спектру можливостей. Вона активно використовується у сфері розробки програмного забезпечення, обробки даних, автоматизації процесів, створення веб-додатків, а також у таких напрямках, як штучний інтелект та машинне навчання.

Однією з ключових причин вибору Python для цього проекту є його зручність у розробці та велика кількість бібліотек, що спрощують виконання завдань, зокрема роботу з базами даних, створення графічного інтерфейсу та взаємодію з іншими технологіями.

Основні переваги Python для реалізації проекту:

1. Легкість у вивченні та читабельність коду

Python має простий і зрозумілий синтаксис, що дозволяє розробникам швидко освоїти мову та зосередитися на вирішенні основних завдань. Це особливо корисно для розробки програм, які потребують швидкого створення та тестування функціональності.

2. Розвинена екосистема бібліотек завдяки широкому вибору готових бібліотек Python значно спрощує роботу над проектами. Наприклад, для взаємодії з базами даних у цьому проекті буде використано бібліотеку `pyodbc`, яка забезпечує зручне підключення до SQL Server. Для побудови графічного інтерфейсу користувача використовується `Tkinter`, що дозволяє легко створювати інтуїтивно зрозумілий GUI.

3. Кросплатформність програми, написані на Python, можуть працювати на різних операційних системах без значних змін у коді. Це забезпечує високу гнучкість і можливість використання розробленого програмного забезпечення на пристроях із різними ОС, такими як Windows, Linux та macOS.

4. Швидка розробка та зручне тестування Python дозволяє суттєво скоротити час на створення програмного забезпечення завдяки своїй простоті та великій кількості інструментів для автоматизації тестування. Це допомагає швидко знаходити та виправляти помилки, що є особливо важливим у процесі розробки.

5. Гнучкість та розширюваність Python є потужною мовою, яка дозволяє реалізовувати складні алгоритми та інтегрувати додатковий функціонал. Наприклад, за допомогою бібліотек для роботи з мережею можна реалізувати відправлення повідомлень через електронну пошту, SMS або месенджери.

6. Об'єктно-орієнтований підхід та підтримка об'єктно-орієнтованого програмування (ООП) дає можливість організувати код у вигляді класів і об'єктів, що робить його більш структурованим і легким для розширення. Це важливий аспект при розробці програмного забезпечення, яке має потенціал для подальшого вдосконалення та додавання нових можливостей.

Завдяки цим перевагам Python є ідеальним вибором для реалізації проекту, забезпечуючи високу продуктивність, легкість у розробці та зручність у використанні.

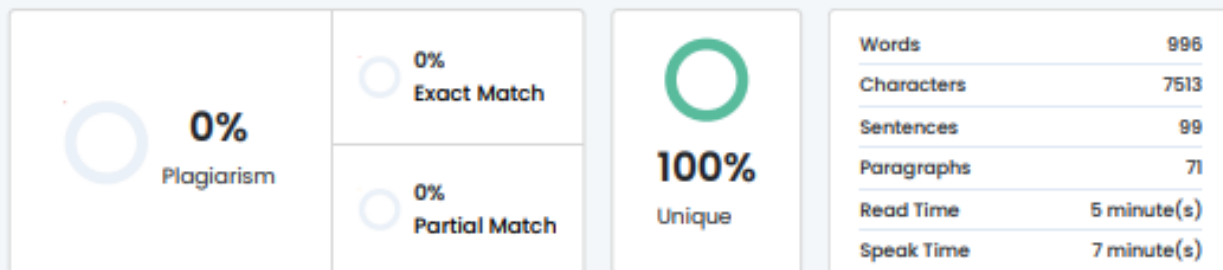
## 2.2 Історія створення мови програмування Python

Python — це високорівнева мова програмування, яка була створена наприкінці 1980-х років і офіційно випущена в 1991 році. Її розробником став Гвідо ван Россум (Guido van Rossum), голландський програміст, який на той час працював у дослідницькому центрі CWI (Centrum Wiskunde & Informatica) в Нідерландах.

Передумови створення:

Гвідо ван Россум працював над проектом ABC — мовою програмування, розробленою для навчальних цілей, яка мала простий синтаксис і була орієнтована на легкість використання. Однак ABC мала певні недоліки, зокрема обмежені можливості розширення. Це спонукало ван Россума створити нову мову, яка б поєднувала простоту синтаксису з потужністю та гнучкістю.

## Plagiarism Scan Report



## Content Checked For Plagiarism

Початок розробки:

У 1989 році Гвідо ван Россум почав працювати над новою мовою у вільний час, використовуючи напрацювання проекту ABC та додаючи до них такі важливі функції, як робота з винятками, модульність та взаємодію з операційною системою.

Назву «Python» він обрав не на честь змії, а через своє захоплення британським комедійним шоу Monty Python's Flying Circus. Він прагнув, щоб нова мова була легкою та зрозумілою, а її використання приносило задоволення.

Офіційний випуск:

Перша версія Python 0.9.0 була випущена в 1991 році. Вона вже мала ключові особливості, які визначають мову до сьогодні:

- Динамічну типізацію
- Автоматичне керування пам'яттю
- Підтримку об'єктно-орієнтованого програмування
- Вбудовані типи даних (списки, словники тощо)

У 1994 році вийшла Python 1.0, яка стала популярною серед розробників завдяки своїй простоті та потужності.

Подальший розвиток:

- Python 2.0 (2000 р.) — додано збірку сміття та підтримку спискових включень (list comprehensions).
- Python 3.0 (2008 р.) — значне оновлення, яке принесло покращення в синтаксисі та управлінні пам'яттю, проте не зберегло зворотну сумісність із Python 2.x.

На сьогодні Python є однією з найпопулярніших мов програмування у світі, активно використовується в розробці веб-додатків, наукових дослідженнях, аналізі даних, машинному навчанні та багатьох інших сферах.

Рис.2.2.1 – Середовище розробки Visual Studio

### 2.3 База даних SQL Server

SQL Server — це потужна реляційна система керування базами даних (РСКБД), розроблена компанією Microsoft. Вона забезпечує зберігання, обробку та управління великими обсягами даних, а також підтримує мову структурованих запитів SQL (Structured Query Language) для взаємодії з даними.

Історія та розвиток:

Перша версія SQL Server була випущена в 1989 році як спільний продукт компаній Microsoft, Sybase та Ashton-Tate. Однак у подальшому Microsoft самостійно розвивала систему, зробивши її ключовим компонентом екосистеми своїх корпоративних рішень.

З роками SQL Server зазнав численних удосконалень, серед яких підтримка хмарних технологій, розширені засоби безпеки, покращена продуктивність та оптимізація запитів.

Основні можливості SQL Server:

1. Реляційна модель даних — SQL Server використовує таблиці, що мають зв'язки між собою, що дозволяє ефективно зберігати й обробляти великі обсяги структурованої інформації.



2. Підтримка транзакцій – забезпечує цілісність даних завдяки використанню механізмів ACID (атомарність, узгодженість, ізоляція, надійність).
3. Розширена система безпеки – містить механізми аутентифікації, авторизації, шифрування та аудитів для захисту даних.
4. Оптимізація продуктивності – завдяки кешуванню запитів, індексуванню та аналітичним інструментам SQL Server дозволяє швидко обробляти складні запити.
5. Підтримка хмарних технологій – інтеграція з Microsoft Azure дає змогу працювати з базами даних у хмарному середовищі.
6. Вбудовані засоби резервного копіювання – дозволяють автоматично створювати резервні копії для захисту від втрати даних.

Використання SQL Server у проекті:

У цьому проекті SQL Server використовується для зберігання інформації про книги, їхні характеристики (автор, жанр, рейтинг), а також управління даними користувачів та рекомендаціями. Основні завдання, які виконує база даних у програмі:

- Збереження списку книг із відповідними атрибутами.
- Фільтрація та пошук книг за жанром, рейтингом та іншими параметрами.
- Зберігання інформації про користувачів і їхні вподобання.
- Формування рекомендацій на основі вподобань користувачів.

Для взаємодії Python з SQL Server у проекті використовується бібліотека `pyodbc`, яка забезпечує виконання SQL-запитів та отримання даних з бази.

SQL Server є оптимальним вибором для реалізації цього програмного додатку, оскільки забезпечує високу продуктивність, безпеку та надійність збереження інформації.

Рис.2.3.1 – База даних BooksDB в SQL Server

### 3. ПРОЕКТУВАННЯ ЗАГАЛЬНОГО АЛГОРИТМУ РОБОТИ ПРОГРАМИ

#### 3.1 Загальна структура програми

Програмний додаток для рекомендації книг має модульну структуру, що забезпечує зручність у розробці, підтримці та розширенні функціоналу. Основні компоненти програми включають:

- Графічний інтерфейс користувача (GUI) – реалізований за допомогою Tkinter, забезпечує взаємодію користувача з додатком.
- База даних (SQL Server) – використовується для зберігання інформації про книги, користувачів та рекомендації.
- Модуль управління базою даних – клас для взаємодії з SQL Server через бібліотеку `pyodbc`.
- Модуль обробки рекомендацій – алгоритм вибору книг на основі жанру, рейтингу та вподобань користувача.
- Модуль керування користувачами та книгами – реалізує додавання, редагування та видалення книг, а також управління профілями користувачів.

#### 3.2 Алгоритм роботи програми

- Запуск програми – відкривається головне вікно з інтерфейсом.
- Завантаження даних – підключення до SQL Server, отримання списку книг.
- Додавання, редагування та видалення книг – адміністратор може змінювати базу книг.
- Вибір параметрів користувачем – користувач обирає жанр, рейтинг тощо.
- Генерація рекомендацій – алгоритм аналізує вибрані параметри та пропонує відповідні книги.
- Відображення результату – користувач отримує список рекомендованих книг.

Ця структура дозволяє зробити програму зручною, гнучкою та легкою у підтримці, а використання SQL Server та Python забезпечує надійну та швидку роботу додатка.

Рис.3.2.1 – Алгоритм роботи застосунку від початку до дій користувача

Рис.3.2.2 – Алгоритм роботи завдання "Додати книгу"

Рис.3.2.3 – Алгоритм роботи завдання "Редагувати книгу"

Рис.3.2.4 – Алгоритм роботи завдання "Видалити книгу"

Рис.3.2.5 – Алгоритм роботи завдання "Отримати рекомендації"

Рис.3.2.6 – Алгоритм роботи завдання "Отримати список книг"

Проектування алгоритму є важливим етапом розробки програмного забезпечення, що дозволяє структурувати логіку роботи та оптимізувати процеси. Запропонована схема забезпечує гнучкість, надійність та зручність у використанні, а також гарантує ефективний підбір та рекомендацію книг відповідно до вподобань користувача. Використання бази даних дозволяє зберігати та обробляти інформацію про книги, жанри, рейтинги та користувачів, що сприяє точності та персоналізації рекомендацій.

#### 4. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка програмного забезпечення для рекомендації книг включає кілька основних етапів: створення бази даних, реалізацію основних модулів та розробку графічного інтерфейсу. База даних містить інформацію про книги, жанри, рейтинги та вподобання користувачів. Основні модулі забезпечують обробку даних та генерацію рекомендацій, а графічний інтерфейс надає користувачам зручний доступ до функціоналу додатку.

##### 4.1 Створення бази даних

Створення бази даних є одним із ключових етапів розробки програмного додатку для рекомендації книг. База даних забезпечує збереження та управління інформацією про книги, жанри, рейтинги, відгуки користувачів та їхні вподобання.

Процес розробки бази даних включає:

- Проектування структури даних та взаємозв'язків між таблицями.
- Використання SQL Server для зберігання даних.
- Реалізацію взаємодії додатку з базою даних через мову Python та бібліотеку pyodbc.

Правильна організація бази даних дозволяє швидко обробляти інформацію, створювати

Для роботи з базою даних використовується бібліотека pyodbc, яка дозволяє взаємодіяти з SQL Server через ODBC (Open Database Connectivity).

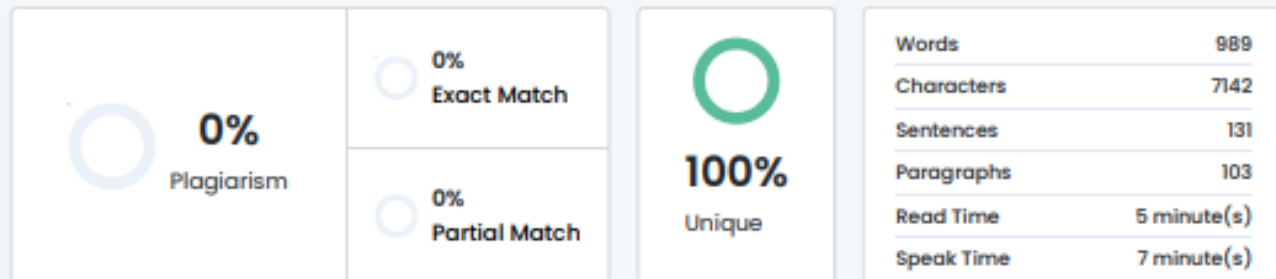
ODBC – це стандартний інтерфейс програмування, який дозволяє Python підключатися до різних СУБД (систем управління базами даних), зокрема Microsoft SQL Server.

Рис.4.1.1 – Імпорт pyodbc

#### Matched Source

No plagiarism found

## Plagiarism Scan Report



## Content Checked For Plagiarism

## 4.2 Налаштування підключення до бази даних

- `server` – змінна, яка має містити назву сервера бази даних (може бути локальний сервер або хмарний).
- `database` – назва бази даних, у якій зберігається інформація про книги.

Рис.4.2.1 – Назва бази даних та серверу

## 4.3 Функція для встановлення підключення до бази даних

Функція `get_connection()` створює та повертає об'єкт підключення до бази даних.

Основні параметри підключення:

- `DRIVER={{ODBC Driver 17 for SQL Server}}` – вказує драйвер ODBC для SQL Server.
- `SERVER={server}` – змінна, яка містить адресу або ім'я сервера.
- `DATABASE={database}` – змінна, яка містить назву бази даних (BooksDB).
- `Trusted_Connection=yes` – дозволяє використовувати автентифікацію Windows для підключення.
- `Encrypt=yes` – активує шифрування з'єднання для підвищення безпеки.
- `TrustServerCertificate=yes` – дозволяє використовувати самопідписані сертифікати при підключенні.

Ця функція використовується в усіх інших частинах програми для встановлення зв'язку з базою даних.

Рис.4.3.1 – Підключення до бази даних

## 4.4 Функція для додавання книги до бази даних

Перевірка введених даних:

- Перед виконанням запиту перевіряється, чи всі необхідні дані передані у функцію.
- Якщо хоча б одне поле не заповнене, програма виведе повідомлення про помилку і припинить виконання функції.

Рис.4.4.1 – Функція додавання книги

Виконання SQL-запиту:

- Використовується контекстний менеджер `with`, щоб забезпечити автоматичне закриття підключення після виконання операції.
- Формується SQL-запит для вставки нової книги до таблиці Books.
- Використовуються параметризовані запити (?), які допомагають уникнути SQL-ін'єкцій.
- `conn.commit()` – підтверджує внесення змін у базу даних.

Рис.4.4.2 – Вигляд SQL-запиту

Обробка помилок:

- Якщо виникає помилка, вона виводиться у консоль.

Рис.4.4.3 – Обробка помилок

## 4.5 Функція для отримання рекомендованих книг із бази даних

Імпорти та клас:

- Ця функція шукає книги у базі за вказаним жанром.
- Якщо вказано `min_rating`, запит доповнюється фільтром за мінімальним рейтингом.

Рис.4.5.1 – Функція отримання рекомендованих книг

Фільтрація за рейтингом:

- Якщо `min_rating` не передано, беруться всі книги вказаного жанру.

Рис.4.5.2 – Фільтрація за рейтингом

Виконання запиту та отримання результату:

- `cursor.fetchall()` повертає всі знайдені книги у вигляді списку.

Рис.4.5.3 – Отримання результату

4.6 Функція для отримання всіх книг із бази даних

Функція `get_all_books_from_db()` виконує запит до бази даних і повертає список усіх книг, які є в таблиці `Books`.

Алгоритм роботи:

1. Встановлення підключення до бази даних:
  - Використовується контекстний менеджер `with get_connection() as conn`, який гарантує, що з'єднання буде автоматично закрито після завершення роботи.
2. Відправлення SQL-запиту:
  - Виконується запит `SELECT`, який отримує всі записи з таблиці `Books`.
  - У запиті вибираються лише конкретні поля: `Title` (назва), `Author` (автор), `Genre` (жанр), `Year` (рік) і `Rating` (рейтинг).
3. Отримання та повернення результатів:
  - Метод `fetchall()` повертає список усіх рядків, отриманих у результаті запиту.
  - Функція повертає цей список, щоб можна було використовувати його в інших частинах програми.
4. Обробка помилок:

Функція включає два блоки `except` для перехоплення можливих помилок

- `pyodbc.Error` – для обробки помилок, пов'язаних із SQL-запитами та підключенням до бази.
- 5. У разі помилки виводиться відповідне повідомлення, а функція повертає порожній список (`[]`).
- `Exception` – загальний обробник для інших можливих винятків, не пов'язаних безпосередньо з `pyodbc`.

Функція `get_all_books_from_db()` дозволяє отримати список усіх книг із бази даних та забезпечує стабільність роботи завдяки обробці можливих помилок. Вона використовується для виведення повного каталогу книг у графічному інтерфейсі або в консольному виведенні.

Рис.4.6.1 – Функція для отримання всіх книг із бази даних

4.7 Функція для оновлення інформації про книгу в базі даних

Функція `update_book_in_db(book_id, title=None, author=None, genre=None, year=None, rating=None)`

дозволяє оновлювати інформацію про книгу в базі даних. Вона приймає `book_id` як обов'язковий параметр, а всі інші параметри є необов'язковими (`None` за замовчуванням), що дозволяє оновлювати лише певні поля.

Алгоритм роботи:

1. Встановлення підключення до бази даних:
  - Використовується контекстний менеджер `with get_connection() as conn`, щоб автоматично закрити з'єднання після завершення операції.
2. Формування SQL-запиту:
  - Спочатку створюється базовий запит.
  - Створюється список параметрів `params = []`, який буде містити значення для оновлення.
3. Додавання змінних у SQL-запит:
  - Якщо передано значення для `title`, `author`, `genre`, `year`, `rating`, вони додаються до запиту.
  - Аналогічно додаються інші поля.
4. Видалення зайвої коми:
  - Оскільки додаємо змінні динамічно, у кінці запиту може бути зайва кома.
  - Додавання умови `WHERE` для конкретного запису
  - Вказуємо, що оновлення має стосуватися лише книги з певним `id`:



#### 5. Виконання SQL-запиту

- Передаємо сформований запит і список параметрів у `cursor.execute()`.
- Використовується підстановка `?`, що захищає від SQL-ін'єкцій.

#### 6. Збереження змін:

- Виконується `conn.commit()`, щоб зміни були застосовані в базі даних.

#### 7. Обробка помилок:

- Якщо виникає помилка SQL-запиту, виводиться відповідне повідомлення.
- Якщо виникає інша невідома помилка, вона теж обробляється.

Функція `update_book_in_db()` дозволяє оновлювати дані про книги в базі без необхідності змінювати всі поля. Завдяки цьому можна, наприклад, змінити тільки рейтинг книги або її автора, залишивши інші дані без змін.

Рис.4.7.1 – Функція для оновлення інформації про книгу в базі даних

#### 4.8 Функція для видалення книги з бази даних

Функція `delete_book_from_db(book_id)` видаляє книгу з бази даних за її унікальним ідентифікатором `book_id`. Вона встановлює з'єднання з базою, виконує SQL-запит для видалення відповідного запису, перевіряє успішність операції та обробляє можливі помилки.

Алгоритм роботи:

##### 1. Встановлення з'єднання з базою даних:

- Викликається функція `get_connection()`, яка повертає підключення до бази.
- Використовується `conn.cursor()`, що дозволяє виконувати SQL-запити.

##### 2. Формування та виконання SQL-запиту:

- Запит `DELETE FROM Books WHERE id = ?` використовується для видалення книги, де `?` замінюється на значення `book_id`.

- Передача параметрів окремо (`(book_id,)`) захищає від SQL-ін'єкцій.

##### 3. Підтвердження змін у базі:

- Викликається `conn.commit()`, щоб зміни набули чинності.

##### 4. Перевірка успішності операції:

- Якщо `cursor.rowcount == 0`, тобто жоден запис не був видалений, виникає помилка `ValueError`, що означає, що книги з таким `id` у базі не існує.

##### 5. Закриття курсору та з'єднання:

- `cursor.close()` та `conn.close()` закривають об'єкти, щоб звільнити ресурси бази даних.

##### 6. Обробка помилок:

- Якщо під час роботи функції виникає будь-яка помилка, вона перехоплюється у блоці `except`.
- Створюється новий виняток із повідомленням про помилку, щоб користувач отримав зрозумілу інформацію про проблему.

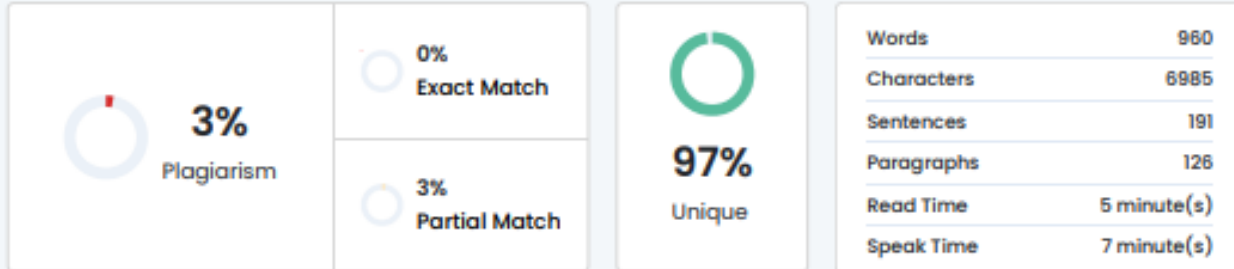
Функція забезпечує безпечне видалення книги з бази даних і гарантує, що користувач отримає повідомлення у випадку, якщо книгу не знайдено або виникла інша помилка.

Рис.4.8.1 – Функція для видалення книги з бази даних

## Matched Source

No plagiarism found

## Plagiarism Scan Report



## Content Checked For Plagiarism

## 4.9 Візуальна частина

Імпорт бібліотек:

- `tkinter` – основна бібліотека для створення GUI в Python.
  - `messagebox` – модуль для відображення повідомлень (підтвердження успіху, помилки тощо).
  - `ttk` – модуль для розширених віджетів інтерфейсу.
  - `database_connection` – імпорт функцій для роботи з базою даних.
- o `add_book_to_db()` – додає книгу до бази даних.
  - o `get_recommendations_from_db()` – отримує рекомендації книг.
  - o `update_book_in_db()` – оновлює дані книги в базі.
  - o `delete_book_from_db()` – видаляє книгу з бази.

Рис.4.9.1 – Імпорт бібліотек

## 4.10 Функція для додавання книги

Призначення: отримує дані з полів вводу та передає їх у базу.

Алгоритм роботи:

1. Отримує назву, автора, жанр, рік та рейтинг книги.
2. Перевіряє, чи всі поля заповнені.
3. Пробоє конвертувати рік у `int`, а рейтинг у `float`.
4. Викликає функцію `add_book_to_db()` для додавання книги.
5. Відображає повідомлення про успішне додавання або помилку.
6. Очищає поля вводу після додавання.

Рис.4.10.1 – Функція для додавання книги

## 4.11 Функція для редагування книги

Призначення: оновлює дані книги у базі даних.

Алгоритм роботи:

1. Отримує ID книги та нові значення полів.
2. Перевіряє, чи всі поля заповнені.
3. Конвертує рік і рейтинг у відповідні числові формати.
4. Викликає функцію `update_book_in_db()` для оновлення запису в базі.
5. Показує повідомлення про успіх або помилку.
6. Очищає поля вводу.

Рис.4.11.1 – Функція для редагування книги

## 4.12 Функція для видалення книги

Призначення: видаляє книгу з бази даних за її ID.

Алгоритм роботи:

1. Отримує ID книги.
2. Перевіряє, чи введено ID.
3. Викликає `delete_book_from_db()` для видалення книги.
4. Показує повідомлення про успіх або помилку.
5. Очищає поля вводу.

Рис.4.12.1 – Функція для видалення книги

#### 4.13 Функція очищення форми

Функція `clear_add_book_fields()` потрібна для того, щоб після виконання операцій з книгами всі поля ставали порожніми та були готові до введення нових даних.

Коли ця функція викликається, вона:

1. Очищає поле ID книги → `entry_book_id.delete(0, tk.END)`
2. Очищає поле назви книги → `entry_title.delete(0, tk.END)`
3. Очищає поле автора → `entry_author.delete(0, tk.END)`
4. Очищає поле жанру → `entry_genre.delete(0, tk.END)`
5. Очищає поле року видання → `entry_year.delete(0, tk.END)`
6. Очищає поле рейтингу → `entry_rating.delete(0, tk.END)`

Після додавання, редагування або видалення книги потрібно очистити поля вводу, щоб користувач міг вводити нові дані без необхідності видаляти старі вручну.

Рис.4.13.1 – Функція очищення форми

#### 4.14 Функція для отримання рекомендацій

Призначення: відображає список рекомендованих книг за жанром та рейтингом.

Алгоритм роботи:

1. Отримує вибраний жанр і мінімальний рейтинг.
2. Конвертує рейтинг у число або залишає `None`.
3. Викликає `get_recommendations_from_db()` для отримання списку книг.
4. Відображає список рекомендацій у текстовому полі.

Рис.4.14.1 – Метод `send_reminder`

#### 4.15 Ініціалізація головного вікна

1. Ініціалізація головного вікна:

- `root = tk.Tk()` – створює головне вікно додатка.
- `root.title("Програмний додаток для рекомендацій книг")` – задає заголовок вікна.
- `root.geometry("650x750")` – встановлює розмір вікна (ширина 650 пікселів, висота 750 пікселів).
- `root.config(bg="#e0f7fa")` – задає фон вікна (блідо-блакитний колір).

2. Налаштування стилю кнопок:

- Використовується `ttk.Style()` для зміни стилю кнопок (`TButton`).
- `style.configure("TButton", font=("Arial", 12), padding=10, relief="flat", background="#4CAF50", foreground="red")` – змінює вигляд кнопок (шрифт, колір, розмір).
- `style.map("TButton", background=[('active', '#388E3C')])` – змінює колір кнопки при наведенні.

3. Фрейм для роботи з книгами:

- `frame_add_book=tk.LabelFrame(root,text="Додати/Редагувати/Видалити книгу", ...)` – створює контейнер для віджетів, пов'язаних з додаванням, редагуванням і видаленням книг.

• Додаються поля вводу:

- o `entry_book_id` – для введення ID книги при редагуванні або видаленні.
- o `entry_title` – для введення назви книги.
- o `entry_author` – для введення імені автора.
- o `entry_genre` – для введення жанру книги.
- o `entry_year` – для введення року видання.
- o `entry_rating` – для введення рейтингу книги.

• Додаються кнопки:

- o Додати книгу (викликає `add_book`).
- o Редагувати книгу (викликає `edit_book`).

- o Видалити книгу (викликає `delete_book`).

4. Фрейм для рекомендацій книг:

- `frame_recommend = tk.LabelFrame(root, text="Рекомендації", ...)` – створює контейнер для рекомендацій.
- `combo_genre` – випадючий список жанрів.
- `entry_recommend_rating` – поле для введення мінімального рейтингу.
- Отримати рекомендації – кнопка, яка викликає `recommend_books`.
- `result_text` – текстове поле для відображення списку рекомендованих книг.

5. Запуск головного циклу програми:

- `root.mainloop()` – запускає головний цикл обробки подій, щоб вікно програми залишалося відкритим та реагувало на дії користувача.

Рис.4.15.1 – Ініціалізація головного вікна

Рис.4.15.2 – Налаштування фреймів та кнопок

Рис.4.15.3 – Налаштування рекомендацій

## 5. КЕРІВНИЦТВО КОРИСТУВАЧА

Програмний додаток для рекомендацій книг має зручний графічний інтерфейс, що дозволяє користувачам легко керувати базою даних книг та отримувати рекомендації.

Запуск програми:

Щоб розпочати роботу з додатком, необхідно запустити файл `book_recommendation_app.py`. Після запуску відкриється головне вікно, яке містить основні інструменти для керування книгами та отримання рекомендацій.

Головне вікно містить два основні розділи:

- Блок для управління книгами (додавання, редагування, видалення записів).
- Блок рекомендацій (отримання списку рекомендованих книг).

Рис.5.1 – Загальний вигляд програмного додатку

Функціональні можливості програми:

1) Додавання нової книги:

Щоб додати книгу до бази даних, користувачеві необхідно:

1. Ввести назву книги у відповідне поле.
2. Вказати автора книги.
3. Обрати або ввести жанр книги.
4. Ввести рік видання книги.
5. Вказати рейтинг (оцінку книги за шкалою, наприклад, від 1 до 10).
6. Натиснути кнопку "Додати книгу".

Після натискання кнопки книга буде збережена в базі даних, і користувач отримає повідомлення про успішне додавання.

Рис.5.2 – Додавання нової книги

Рис.5.3 – Повідомлення про успішне додавання нової книги до бази даних

2) Редагування книги:

Щоб змінити інформацію про вже існуючу книгу, необхідно:

1. Ввести ID книги, яку потрібно відредагувати (ID можна знайти у списку або при перегляді даних).



2. Ввести нові значення для змінюваних полів (назва, автор, жанр, рік видання, рейтинг).
3. Натиснути кнопку "Редагувати книгу".

Якщо книга з вказаним ID є в базі даних, програма оновить її дані та покаже повідомлення про успішне редагування. Якщо ID не знайдено, з'явиться відповідне повідомлення про помилку.

Рис.5.4 – Редагування існуючої книги

Рис.5.5 – Повідомлення про успішне оновлення книги

### 3) Видалення книги:

Для видалення книги з бази даних потрібно:

1. Ввести ID книги, яку потрібно видалити.
2. Натиснути кнопку "Видалити книгу".

Програма видалить запис із бази даних та виведе повідомлення про успішне видалення. Якщо книга з таким ID відсутня, програма повідомить про помилку.

## Matched Source

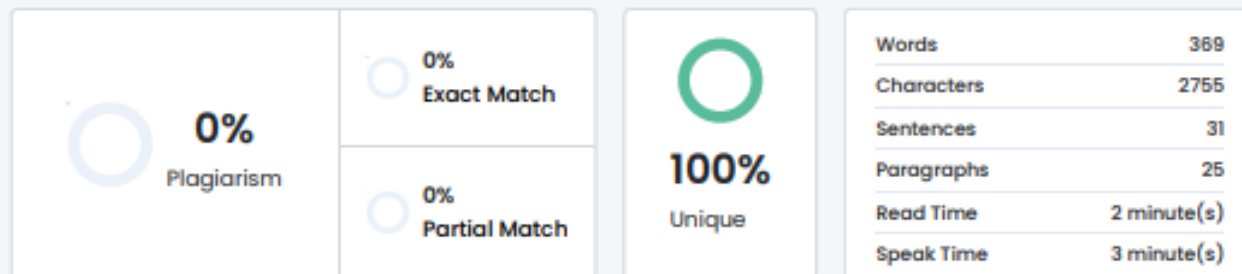
### Similarity 15%

**Title:** Python. Програми з графічним інтерфейсом — урок. Інформатика, 8 клас.

```
root = Tk() root.title('Графічна програма') # Встановити заголовок вікна root.geometry('250x150') # Встановити розміри вікна (в пікселях) root.mainloop() # команда запуску головного циклу обробки подій
```

<https://www.miyklas.com.ua/p/informatica/8-klas/algoritmi-ta-programi-394917/programi-z-grafichnim-interfeisom-395197/re-6038e80b-3201-49cd-a207-7518a6aab2c0>

## Plagiarism Scan Report



## Content Checked For Plagiarism

Рис.5.6 – Повідомлення про успішне видалення книги з бази даних

4) Отримання рекомендацій:

Користувач може отримати список рекомендованих книг за обраним жанром та мінімальним рейтингом. Для цього потрібно:

1. Обрати бажаний жанр зі списку доступних жанрів.
2. Ввести бажаний мінімальний рейтинг (наприклад, 7 або 8).
3. Натиснути кнопку "Отримати рекомендації".

Після натискання кнопки в нижній частині вікна з'явиться список книг, які відповідають заданим критеріям.

Відображення результатів:

Результати роботи програми, такі як список рекомендованих книг або повідомлення про успішні дії, відображаються в текстовому полі у нижній частині головного вікна.

Рис.5.7 – Отримання рекомендацій за жанром "Фантастика" та рейтингом не нижче 4.6

Рис.5.8 – Отримання рекомендацій за жанром "Поезія" та рейтингом не нижче 4.9

Користувач може переглянути додані книги, редагувати записи або видаляти застарілу інформацію, що робить програму зручною для організації особистої бібліотеки.

Збереження даних:

Усі дані про книги автоматично зберігаються в базі даних SQL Server, що забезпечує збереження інформації навіть після закриття програми. Це дозволяє користувачеві продовжити роботу з книгами без втрати введених даних.

Рис.5.9 – Збережені книги в базі даних

## 6. ВИСНОВКИ

Метою даного курсового проекту було створення програмного додатка для рекомендації книг. Робота над проектом була розділена на кілька етапів, що дозволило чітко визначити завдання та послідовно їх реалізувати:

- Аналіз предметної області та визначення вимог до додатка;
- Проектування архітектури програми та бази даних;
- Реалізація функціоналу додавання, редагування та видалення книг;

- Розробка графічного інтерфейсу для взаємодії з користувачем;
- Тестування додатка та усунення помилок.

На початковому етапі було проведено аналіз функціональних вимог до системи, визначено необхідні можливості, такі як збереження інформації про книги, пошук та видача рекомендацій на основі жанру та рейтингу. Для реалізації додатка було обрано мову програмування Python у поєднанні з бібліотекою Tkinter для створення графічного інтерфейсу та базою даних для збереження інформації.

У ході розробки було створено зручний інтерфейс, що дозволяє користувачам легко керувати списком книг та отримувати рекомендації відповідно до своїх уподобань. Завдяки використанню бази даних забезпечено надійне збереження даних і швидкий доступ до них.

Таким чином, у результаті виконання курсової роботи я отримав практичні навички проектування програмного забезпечення, розробки баз даних, створення графічних інтерфейсів і роботи з алгоритмами фільтрації та рекомендацій. Розроблений додаток може бути корисним для книжкових магазинів, бібліотек або просто читачів, які шукають нові книги за своїми вподобаннями.

## Matched Source

No plagiarism found