МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. І. Сікорського

КАФЕДРА АВТОМАТИЗОВАНИХ СИСТЕМ ОБРОБКИ ІНФОРМАЦІЇ ТА УПРАВЛІННЯ

**КУРСОВА РОБОТА**

з дисципліни «Структури даних та алгоритми»

на тему:

**«Бібліотека. Облік формулярів»**

Студента 1 курсу групи ІС-03

Спеціальності 126 «Інформаційні системи та технології»

Керівник: Лебідь Сергій Олександрович, асистент

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Національна оцінка:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії:

\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ – 2021

**Національний технічний університет України «КПІ ім. І. Сікорського»**

**Кафедра автоматизованих систем обробки інформації та управління**

Дисципліна «Структури даних та алгоритми»

**Спеціальність 126 «Інформаційні системи та технології»**

Курс 1 Група ІС-03 Семестр 2

**ЗАВДАННЯ**

**на курсову роботу студента**

**Діордєва Івана Андрійовича**

1. Тема проекту (роботи): Бібліотека. Облік формулярів
2. Строк здачі студентом закінченого проекту (роботи)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
3. Вихідні дані до проекту (роботи): мова програмування C#, технічна література, інтернет-джерела.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці): визначення важливості предметної області, розробка алгоритму роботи програми, архітектура програми.
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень): UML діаграма
6. Дата видачі завдання

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ЗМІСТ………………………………………………………………………………………...3**

**ВСТУП………………………………………………………………………………………..4**

**РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ…………………………………………5**

1.1 Обґрунтування та вибір алгоритму створення програми………………………5

1.2 Об’єктно-орієнтований аналіз області…………………………………………...6

1.3 Функціональні вимоги до програми……………………………………………...6

1.4 Use Case діаграма………………………………………………………………….8

**РОЗДІЛ 2. ОПИС ПРОГРАМНОГО ЗАЬЕЗПЕЧЕННЯ…………………………….....9**

2.1 Опис логічної структури програми……………………………………………....9

2.2 Діаграма класів програмного забезпечення……….……………………………13

**РОЗДІЛ 3. ІНСТРУКЦІЯ КОРИСТУВАЧА……………………………………………14**

3.1 Робота з програмою…………………………………………………………...…14

3.2 Системні вимоги………………………………………………………………….18

**РОЗДІЛ 4. ТЕСТУВАННЯ ПРОГРАМИ……………………………………………….19**

4.1 Опис методики тестування програмної системи………………………………19

4.2 Розробка плану тестування програми…………………………………………..19

4.3 Перевірка плану тестування програми…………………………………………19

**ВИСНОВОК………………………………………………………………………………..23**

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ………………………………………………..24**

**ДОДАТОК А………………………………………………………………………………..24**

**ВСТУП**

Ще декілька десятиліть тому книги були майже єдиним джерелом інформації. Але нині все більше і більше людей відмовляються від книг, надаючи перевагу інтернет-джерелам. Проте бувають випадки, коли потрібну інформацію можна знайти лише в літературі. Тому бібліотеки як систематизовані колекції різних друкованих джерел є актуальними і у наш час. Основна вимоги до них - це можливість швидкого та зручного пошуку потрібної літератури. Одним з простих і ефективних способів є створення електронного каталогу, оскільки сучасні комп’ютери дозволяють здійснювати пошук за лічені секунди.

Метою моєї курсової роботи є створення програми, яка є прототипом бібліотеки. Основною її функцією є надання потрібної літератури усім користувачам даної бібліотеки.

Програма повинна забезпечувати виконання наступних функцій:

1. Перегляд каталогу бібліотеки.
2. Пошук потрібної літератури за назвою, автором та тематикою.
3. Можливість реєстрації та отримання потрібної літератури.

Дана програма спрощує пошук потрібної літератури і допомагає обрати потрібну саме користувачу. Також вона має можливості для модернізації – в майбутньому можна додати графічний інтерфейс або зробити веб-версію та підключити до програми систему управління базами даних для більш зручного керування каталогом.

**РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ**

Основним завданням є створити систему взаємодії між користувачем та каталогом. Тому головними поняттями є книга, бібліотека (каталог) та користувач. Основними сутностями є зареєстрований користувач, адміністратор (бібліотекар), бібліотека та книга.

**1.1 Обґрунтування та вибір алгоритму створення програми**

Основним інструментом для написання даної курсової роботи було обрано мову програмування C#, оскільки вона є однією з передових мов програмування. Вона є об’єктно-орієнтованою, має нескладний синтаксис, тому її досить легко вивчати та використовувати. У якості середовища розробки було обрано JetBrains Rider, оскільки воно є досить потужним і в той же час легким у використанні. Також однією з переваг Rider є наявність ReSharper – інструменту, який значно полегшує написання коду. Розробка програми здійснювалась у .NET 5.0.

.NET 5.0 – платформа, заснована на .NET Framework. .NET поділяється на дві основні частини — середовище виконання (по суті віртуальна машина) та інструментарій розробки. Як і технологія Java, середовище розробки .NET створює байт-код, призначений для виконання віртуальною машиною. Вхідна мова цієї машини в .NET називається CIL (Common Intermediate Language), також відома як MSIL (Microsoft Intermediate Language), або просто IL. Застосування байт-коду дозволяє отримати крос-платформність на рівні скомпільованого проєкту (в термінах .NET: *збірка*), а не на рівні сирцевого тексту, як, наприклад, в C. Перед запуском збірки в середовищі виконання (CLR) байт-код перетворюється вбудованим в середовище JIT-компілятором (just in time, компіляція на льоту) в машинні коди цільового процесора. [2]

Інтерфейс користувача реалізований у консолі. Введення усієї потрібно інформації відбувається з клавіатури. Проект містить невелику кількість класів, кожен з яких займає певне місце у програмі. Були використані парадигми ООП – абстракція, інкапсуляція, наслідування та поліморфізм.

**1.2 Об’єктно-орієнтований аналіз області**

Дана програма буде використовуватися для пошуку та використання потрібної літератури серед каталогу бібліотеки.

Основні завдання створеної програми:

* Надати користувачу можливість створити обліковий запис та увійти в нього.
* Надати можливість пошуку необхідної книги за назвою, автором або тематикою.
* Надати можливість переглянути всі книги у даній бібліотеці.
* Надати користувачу можливість взяти потрібну книгу.

Оскільки дана програма працює з книжками, кількість яких обмежена, то необхідно передбачити наступні проблеми, які можуть виникнути у процесі виконання:

* Користувач повинен отримувати лише ті книжки, які відповідають його пошуковому запиту.
* Якщо підходящих книжок немає, то користувач має бути сповіщений про це.
* Користувач має право отримати книжку лише у випадку наявності її у бібліотеці та за умови, що користувач зараз має не більше 10 книжок.

На початку роботи програми у бібліотеці зареєстрований лише адміністратор. Каталог бібліотеки порожній. Щоб додати книжки до бібліотеки, необхідно зайти в обліковий запис адміністратора та додати книжки одним з двох способів: читанням з .csv файлу або вручну. Користувач має створити свій акаунт самостійно.

**1.3 Функціональні вимоги програми**

Дана програма значно спрощує роботу користувачів з бібліотекою. Використовуючи дану програму, необхідно зробити вибір потрібної літератури максимально простим та зрозумілим. Тому можна виокремити наступні функціональні вимоги програми:

1. Користувач може створити обліковий запис.
2. Користувач може переглядати весь каталог бібліотеки.

*Зауваження: якщо каталог порожній – виводиться відповідне повідомлення.*

1. Користувач може здійснювати пошук відповідно до своїх потреб: за назвою, автором та тематикою.

*Зауваження: якщо книжок, що відповідають пошуковому запиту, не знайдено – виводиться відповідне повідомлення.*

1. Користувач може переглянути список своїх книжок.

*Зауваження: якщо користувач не взяв жодної книжки – виводиться відповідне повідомлення.*

1. Користувач може взяти книгу з бібліотеки.

*Зауваження: користувач не може взяти книгу, якщо він її вже взяв, в нього вже знаходиться 10 книг або книга відсутня в бібліотеці.*

1. Користувач може повернути книгу у бібліотеку.

*Зауваження: користувач не може повернути книгу, якщо він її не брав.*

**1.4 Use Case діаграма**

На рисунку 1.1 наведена Use Case діаграма програми.

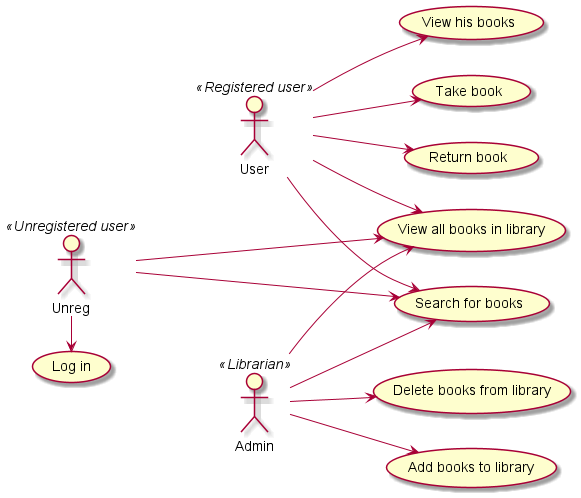


Рис. 1.1

**РОЗДІЛ 2. ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

**2.1 Опис логічної структури програми**

Програма має відносно нескладну структуру. Вона реалізується за допомогою класів та інтерфейсу. Кожен клас має поля та методи, які виконують певну функцію. Далі будуть перераховані використані класи з описом усі компонентів.

Користувацький інтерфейс реалізований за допомогою класу Program. Він не має полів. У ньому присутні наступні методи:

1. Main – точка входу у програму. Відбувається ініціалізація бібліотеки та виклик стартового меню.
2. ShowStartMenu – метод виклику стартового меню (меню незареєстрованого користувача).
3. ShowUserMenu – метод виклику меню зареєстрованого користувача.
4. ShowAdminMenu – метод виклику меню адміністратора (бібліотекаря).
5. ShowAllBooks – метод, що виводить на екран всі книги, які містяться у бібліотеці.
6. FindBookMenu – метод, що викликає меню пошуку потрібної книги.
7. AccountEventHandler – метод-обробник подій, що відбуваються з акаунтами.
8. StorageEventHandler – метод-обробник подій, що відбуваються з бібліотекою.
9. ExceptionHandler – метод-обробник виключних ситуацій.

Клас Library – це головний клас, що реалізує бібліотеку. Він має наступні поля:

* Name – назва бібліотеки
* Users – список зареєстрованих користувачів
* Admin – акаунт адміністратора (бібліотекаря)
* \_totalAccounts – загальна кількість зареєстрованих користувачів
* Books – список книг
* \_totalBooks – загальна кількість книжок у бібліотеці

Також цей клас має наступні методи:

1. AddAccount – метод, що створює новий акаунт користувача та додає його у список Users.
2. RemoveAccount – метод, що видаляє акаунт зі списку користувачів.
3. FindAccount – метод, що знаходить акаунт користувача за вказаним ID.
4. Login – метод, що повертає акаунт користувача у випадку, якщо він ввів правильний ID.
5. LoginAsAdmin – метод, що повертає акаунт адміністратора у випадку, якщо він ввів правильний пароль.
6. AddBook – метод, що створює нову книгу та додає її до списку.
7. RemoveBook – метод, що видаляє вказану книгу зі списку.
8. FindBook – метод, що повертає екземпляр книги або список книг, які підходять за пошуковим запитом. Пошук може здійснюватися за назвою книги, автором або тематикою.
9. FindBookById – метод, що повертає екземпляр книги за вказаним ID.
10. GiveBook – метод, що видає користувачу книгу з вказаним ID.
11. TakeBook – метод, що забирає у користувача книгу з вказаним ID.

Інтерфейс IAccount містить методи, які мають бути реалізовані у кожному похідному класі, що описують акаунти. Це методи LogIn та LogOut.

Клас Account – базовий абстрактний клас. Він містить поле Id, що містить індивідуальний номер облікового запису. Також він містить наступні методи:

1. Конструктор з параметром, що приймає число. Це число – ID.
2. LogIn – метод, що викликає подію LoggedIn.
3. LogOut – метод, що викликає подію LoggedOut.

Клас UserAccount – похідний клас від Account, який представляє зареєстрованого користувача. Він містить поля MyBooks – список книг, які взяв користувач, та Available – число книг, які користувач може взяти. Також він містить наступні методи:

1. Конструктор з параметром, що приймає число. Це число – ID.
2. Create – метод, що викликається при створенні акаунту і викликає подію Created.
3. Delete – метод, що викликається при видаленні акаунту і викликає подію Deleted.
4. LogIn – метод, що викликає подію LoggedIn. Він перевизначає метод, описаний у базовому класі.
5. LogOut – метод, що викликає подію LoggedOut. Він перевизначає метод, описаний у базовому класі.
6. TakeBook – метод, що дає можливість взяти книжку в бібліотеці. Викликає подію TakenBook.
7. ReturnBook – метод, що дає можливість повернути книжку в бібліотеку. Викликає подію ReturnedBook.

Клас LibrarianAccount описує акаунт адміністратора, є похідним класом від Account. Має одне поле \_password, у якому міститься пароль для входу. Також містить наступні методи:

1. Конструктор з параметром, що приймає число. Це число – ID.
2. LogIn – метод, що викликає подію LoggedIn. Він перевизначає метод, описаний у базовому класі.
3. LogOut – метод, що викликає подію LoggedOut. Він перевизначає метод, описаний у базовому класі.
4. AddBook – метод, що додає книгу у вказану бібліотеку.
5. RemoveBook – метод, що видаляє вказану книгу з бібліотеки.

Клас Book описує книгу. Має наступні поля:

1. Name – назва книги.
2. Author – автор.
3. Theme – тематика.
4. ID – унікальний номер.
5. Quantity – загальна кількість книжок.
6. Available – кількість книжок, що залишилася.

Також має конструктор з параметрами, що встановлює вище вказані поля.

AccountEventDelegate – делегат, що відповідає за обробку подій, які відбуваються з акаунтом.

StorageEventDelegate – делегат, що відповідає за обробку подій, які відбуваються з бібліотекою.

BookAlreadyTakenException – клас-виключення, що викликається у випадку, якщо користувач вже має книгу та хоче взяти її ще раз.

BookLimitReachedException – клас-виключення, що викликається у випадку, якщо користувач вже взяв максимальну кількість книг.

BookNotAvailableException - клас-виключення, що викликається у випадку, якщо книг вже не залишилося.

BookNotTakenException - клас-виключення, що викликається у випадку, якщо користувач хоче повернути книгу, яку він не брав.

BookNotReturnedException - клас-виключення, що викликається у випадку, якщо користувач вже має книгу та хоче взяти її ще раз.

WrongIdException - клас-виключення, що викликається у випадку, якщо користувач вказав неправильний ID книги або облікового запису і такого ID не існує в системі.

WrongPasswordException - клас-виключення, що викликається у випадку, якщо вказав неправильний пароль при вході як адміністратор.

**2.2 UML діаграма класів**

UML діаграма класів наведена на рисунку 2.2

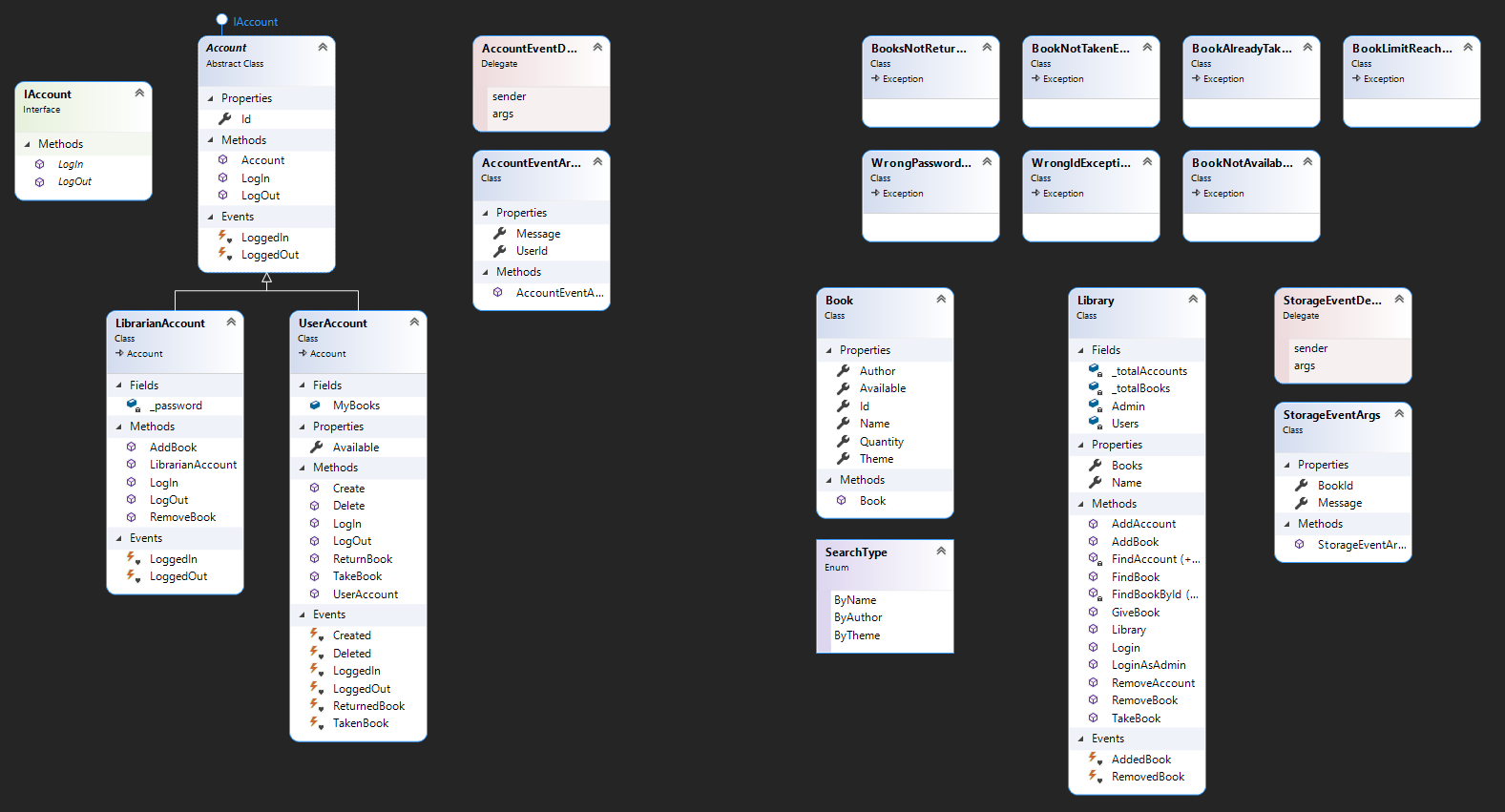


Рис. 2.2

Код програми можна знайти у додатку А.

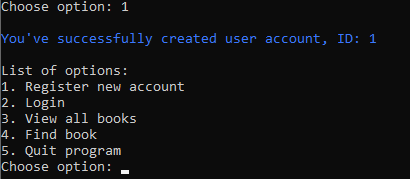
**РОЗДІЛ 3. ІНСТРУКЦІЯ КОРИСТУВАЧА**

**3.1 Робота з програмою**

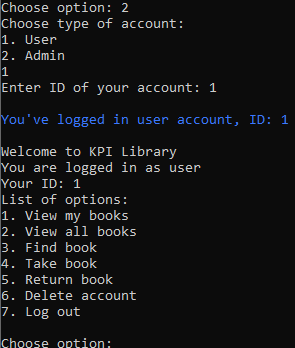
Після запуску програми відкривається меню незареєстрованого користувача.



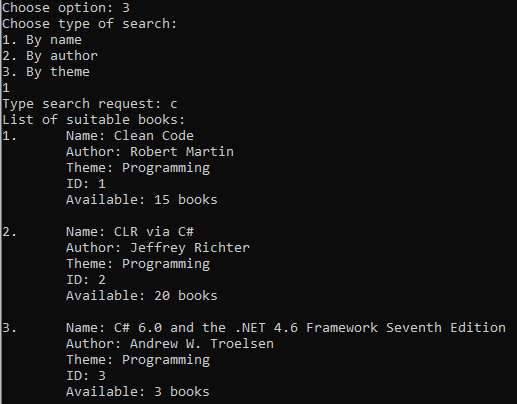
Далі необхідно зареєструвати акаунт, що мати змогу увійти до нього.

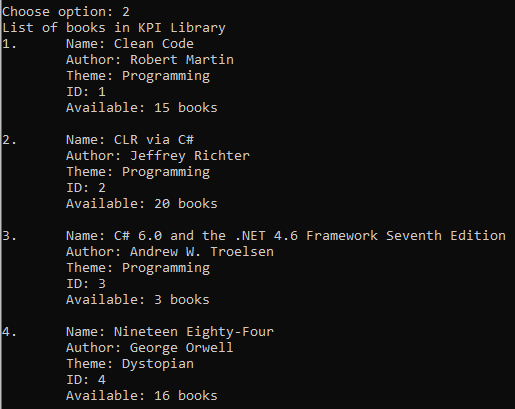


Після цього можна увійти в акаунт, використовуючи ID.

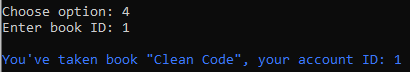


У меню, що з’явилося, можна здійснити пошук за назвою, автором, або тематикою, або відобразити всі книги, що є в бібліотеці.

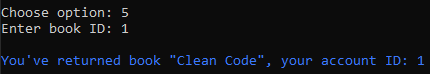




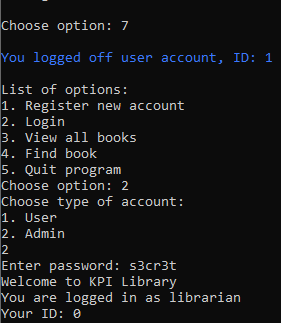
Після цього можна взяти книгу за вказаним ID.



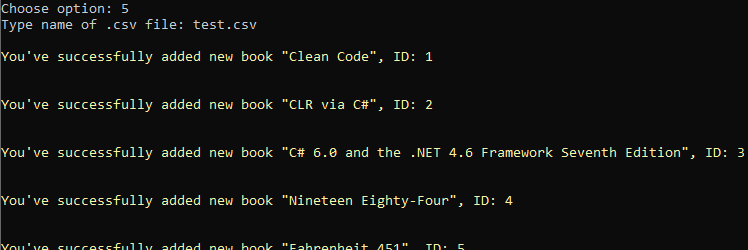
Щоб повернути книгу, необхідно вибрати потрібний пункт у меню та вказати ID книги.

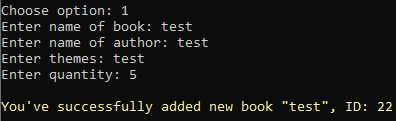


У випадку, якщо необхідно увійти в акаунт адміністратора, необхідно вийти з поточного акаунта та увійти до акаунта адміністратора, вказавши пароль.

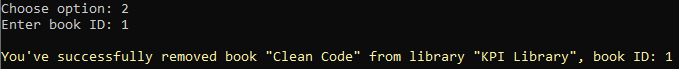


Після цього можна зчитати каталог з .csv файлу, щоб додати кілька книжок за раз, або додати книгу вручну.

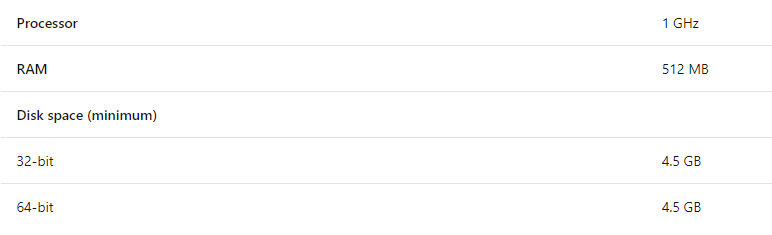




Також можна видалити книгу з бібліотеки, вибравши відповідний пункт у меню та вказавши її ID.



**3.2 Системні вимоги**



**РОЗДІЛ 4. ТЕСТУВАННЯ РОБОТИ ПРОГРАМИ**

**4.1 Опис методики тестування програмної системи**

Тестування програми виконується за допомогою введення неправильних, некоректних або неіснуючих значень. Також програма перевіряється на виникнення виключних ситуацій та правильну їх обробку.

**4.2 Розробка плану тестування програмної системи**

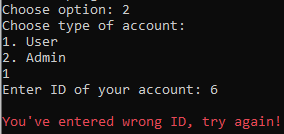
1. Тестування меню стартового меню: введення неіснуючого ID, введення символів типу char замість int, введення неправильного паролю при авторизації як адміністратор.

2. Тестування меню зареєстрованого користувача: спроба переглянути порожній каталог, пошук книги за назвою\автором\тематикою, якої немає в каталозі, введення неправильного ID книги при спробі взяти або повернути книгу, спроба взяти книгу, яка вже є у користувача, спроба повернути книгу, якої немає у користувача, спроба видалити акаунт при наявності книжок.

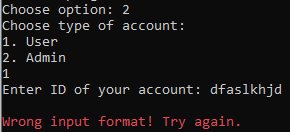
3. Тестування меню адміністратора: введення символів типу char замість int при введенні кількості нової книги, видалення книги, які ще залишилися у користувача, видалення неіснуючої книги, імпортування бібліотеки з неіснуючого .csv файлу.

**4.3 Перевірка плану тестування програмної системи**

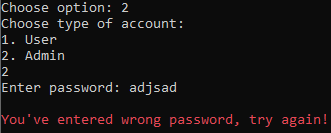
1. Введення неіснуючого ID:



Введення символів типу char замість int:



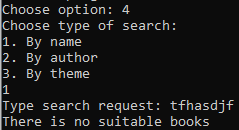
Введення неправильного паролю при авторизації як адміністратор:



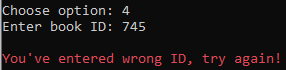
2. Спроба переглянути порожній каталог:



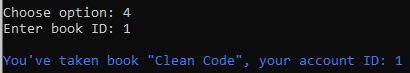
Пошук книги за назвою\автором\тематикою, якої немає в каталозі:

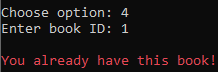


Введення неправильного ID книги при спробі взяти або повернути книгу:

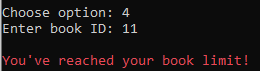


Спроба взяти книгу, яка вже є у користувача:

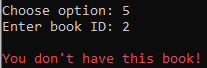




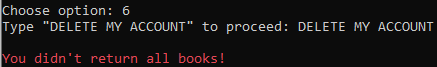
Спроба взяти книгу при наявності 10 книг у користувача:



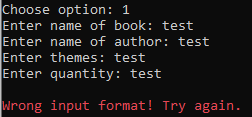
Спроба повернути книгу, якої немає у користувача:



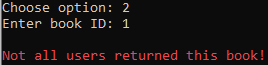
Спроба видалити акаунт при наявності книжок:



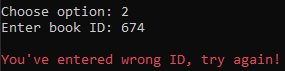
3. Введення символів типу char замість int при введенні кількості нової книги:



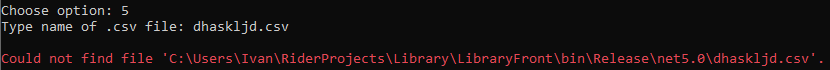
видалення книги, які ще залишилися у користувача:



видалення неіснуючої книги:



імпортування бібліотеки з неіснуючого .csv файлу:



**ВИСНОВОК**

Підсумовуючи, можна сказати, що створена програма працює коректно і виконує всі поставлені задачі, а саме: дає можливість переглянути каталог бібліотеки, здійснити пошук за автором, назвою та тематикою, та взяти необхідну книгу за умови наявності її в бібліотеці та наявності не більше 10 найменувань у користувача.

Програма була розроблена у середовищі JetBrains Rider, мовою програмування C#, інтерфейс реалізований у консольному додатку. Застосовувалися основні парадигми ООП: абстракція, наслідування, інкапсуляція та поліморфізм.

Програма була протестована. Критичних ситуацій виявлено не було. Передбачені виняткові ситуації (наприклад вхід у неіснуючий акаунт, невірний формат вводу тощо), вони коректно обробляються та не заважають роботі програми.

У майбутньому програму можна буде покращити доданням нових функцій. Наприклад, замінити консольний інтерфейс на веб-інтерфейс та підключити її до бази даних.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Полное руководство по языку программирования C# 9.0 и платформе .NET 5 [Електронний ресурс] – Режим доступу до ресурсу: <https://metanit.com/sharp/tutorial/>
2. .NET Framework — Вікіпедія [Електронний ресурс] – Режим доступу до ресурсу: <https://uk.wikipedia.org/wiki/.NET_Framework>
3. Герберт Шилдт - C# 4.0: полное руководство. Жанр: Программирование, издательство ООО "И.Д. Вильямс", год 2011.

**ДОДАТОК А**

<https://github.com/i-diordiev/Library>