Library Management System

Author: Winston Pawitra Lystanto

Contents

Overview

This report highlights the purpose and functionality of creating a library management application, along with descriptions and explanations of the features available in this application.

Purpose and Functionality

The application's purpose is to enable a librarian to view, add, update, and delete books in a library database. For each book, the following information must be stored: ISBN (International Standard Book Number), Book Title, Book Type (Hard Cover, Paper Back or eBook), and Quantity Available.

Additional features are required to complement the main functions (view, add, update and delete) to give the librarian a better user experience. As such, the application is designed such that the librarian could:

1. Search specific books by ISBN
2. Filter books by ISBN, Title, Type or Quantity
3. Enter the appropriate inputs through validation
4. Exit from the application

The following are the main types of inputs validated in the system (disclaimer - some are not listed here):

- ISBN
- Option (e.g. 'y'/'n', multiple choices)
- Page number (will be covered in 'View all books' section)

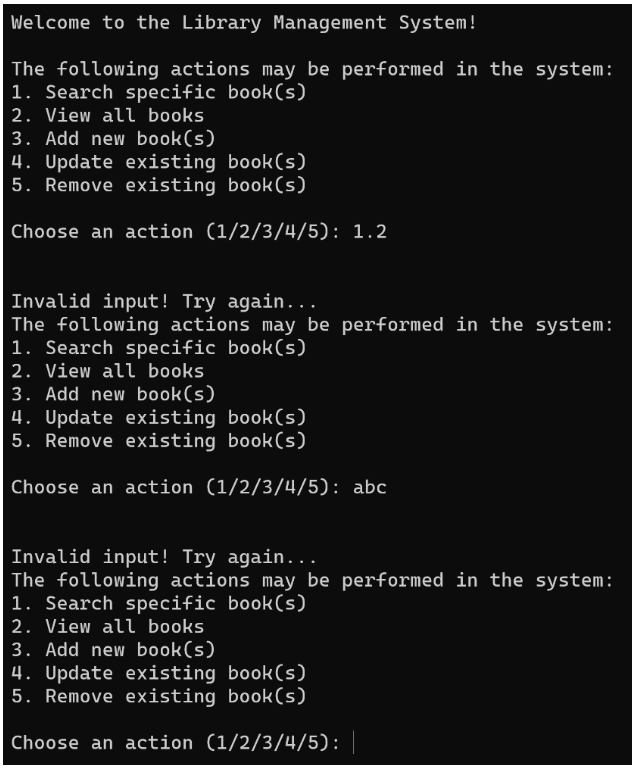
Description and Explanation of Features

Validation

In the program, the 'isbnlib' module is imported to aid in ISBN validation. The program accepts ISBNs with 10/13 digits, checking whether the check-digit (last digit) is correct.

Sometimes, the program will prompt the user to key in 'y'/'n' or choose a number in a range of numbers. The logic of the validator works the same in both scenarios. The program will check whether the input is the same as one of the elements in a list of accepted values. In the case of 'y'/'n' prompts, the program will make the input lowercase to enable a certain degree of error in the input provided by the user.

In either validation, if the validator returns false, the program will show 'Invalid input! Try again…' before reshowing the prompt. This is an example:

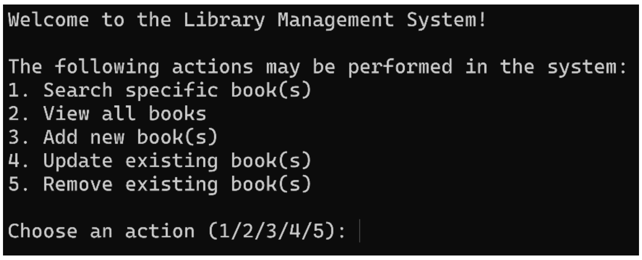


```
Welcome to the Library Management System!

The following actions may be performed in the system:
1. Search specific book(s)
2. View all books
3. Add new book(s)
4. Update existing book(s)
5. Remove existing book(s)

Choose an action (1/2/3/4/5): 1.2


Invalid input! Try again...
The following actions may be performed in the system:
1. Search specific book(s)
2. View all books
3. Add new book(s)
4. Update existing book(s)
5. Remove existing book(s)

Choose an action (1/2/3/4/5): abc


Invalid input! Try again...
The following actions may be performed in the system:
1. Search specific book(s)
2. View all books
3. Add new book(s)
4. Update existing book(s)
5. Remove existing book(s)

Choose an action (1/2/3/4/5):
```

Fig. 1: Examples of Input Validation in Menu

Menu

When the user opens the program, a menu will be shown, prompting the user to choose an action to execute.



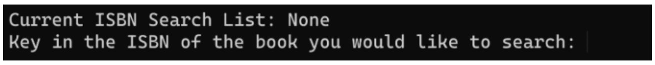
```
Welcome to the Library Management System!

The following actions may be performed in the system:
1. Search specific book(s)
2. View all books
3. Add new book(s)
4. Update existing book(s)
5. Remove existing book(s)

Choose an action (1/2/3/4/5):
```

Fig. 2: Menu of application

Upon entering the appropriate input, the screen is cleared.

Search specific book(s)

The screen is cleared upon keying in '1' as the input in the Menu. The program will show the current ISBN list and prompt the user to key in the book's ISBN to search for his/her desired book.



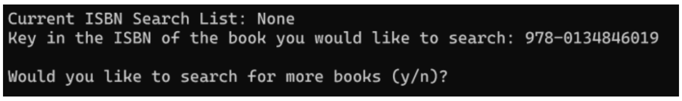
```
Current ISBN Search List: None
Key in the ISBN of the book you would like to search:
```

Fig. 3.1: Showcase of ISBN Search List and ISBN-search Prompt

The rationale for adding an ISBN search list is that the user may want to search multiple books. By adding such a list, the user knows whether he/she is missing out on any ISBN which he/she wants to search.

Note that the ISBN validator runs here, checking if the ISBN exists in the library system. Otherwise, the system shows 'Invalid input! Please try again…' and re-prompt the user for an ISBN.

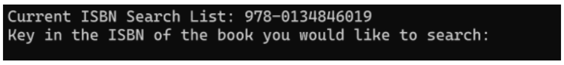
Suppose the user wants to search for the following ISBN, 978-0134846019 and 978-1587147029. The user keys in the first ISBN. After that, the program will prompt whether the user wants to search for another book by ISBN.



```
Current ISBN Search List: None
Key in the ISBN of the book you would like to search: 978-0134846019

Would you like to search for more books (y/n)?
```

Fig. 3.2: Prompt to search for more books

The program uses an option validator to accept only uppercase/lowercase 'y'/ 'n'. If the user inputs 'y', the following is shown:

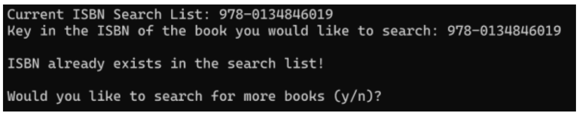


```
Current ISBN Search List: 978-0134846019
Key in the ISBN of the book you would like to search:
```

Fig. 3.3: Updated ISBN Search List

If there is more than one ISBN in the search list, a coma and space will be added before the next ISBN (e.g. 978-0134846019, 978-1587147029 etc.).

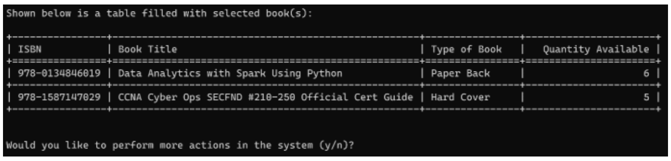
It is also possible that the user may search for the same ISBN again. The program will check for such an occurrence. If so, the program displays the following:



```
Current ISBN Search List: 978-0134846019
Key in the ISBN of the book you would like to search: 978-0134846019

ISBN already exists in the search list!

Would you like to search for more books (y/n)?
```

Fig. 3.4: Outcome of duplication in ISBN Search

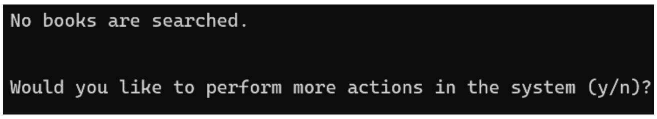
Suppose the user has finished keying in 978-0134846019 and 978-1587147029, and he/she does not want to search for more books. The screen will clear after 'n' or 'N' is keyed in the system, showing the following:



```
Shown below is a table filled with selected book(s):

+----------------+----------------------------------------------------+----------------+----------------------+
| ISBN           | Book Title                                         | Type of Book   |   Quantity Available |
+================+====================================================+================+======================+
| 978-0134846019 | Data Analytics with Spark Using Python             | Paper Back     |                    6 |
+----------------+----------------------------------------------------+----------------+----------------------+
| 978-1587147029 | CCNA Cyber Ops SECFND #210-250 Official Cert Guide | Hard Cover     |                    5 |
+----------------+----------------------------------------------------+----------------+----------------------+


Would you like to perform more actions in the system (y/n)?
```

Fig. 3.5: Output of the Search action (if there are books searched)

However, if no books are searched, the following is shown:



```
No books are searched.

Would you like to perform more actions in the system (y/n)?
```

Fig 3.6: Output of Search Action (if there are no books searched)

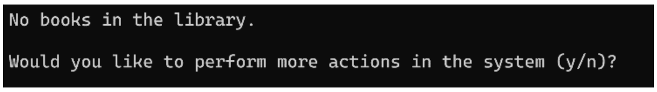
The program will prompt the user to choose between exiting the system and performing more actions in the system. The program will use the 'y'/'n' validator to validate the user input.

If 'y' or 'Y' is entered, the program goes back to the Menu. The program exits if 'n' or 'N' is entered.

View all books

P.s. For the purpose of illustration, two additional books were added to the system.

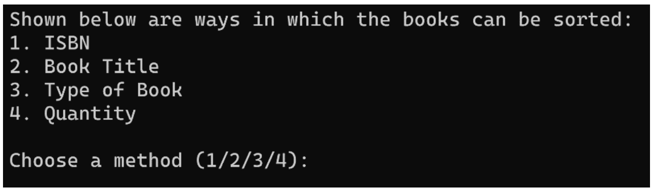
Upon keying in '2' as the input in the Menu, the screen is cleared. If there are books in the library, the system will display 'No books in the library.' and prompt whether the user wants to perform more actions in the system.



```
No books in the library.

Would you like to perform more actions in the system (y/n)?
```

Fig. 4.1: No books in System

If there are books in the library, the following is shown:

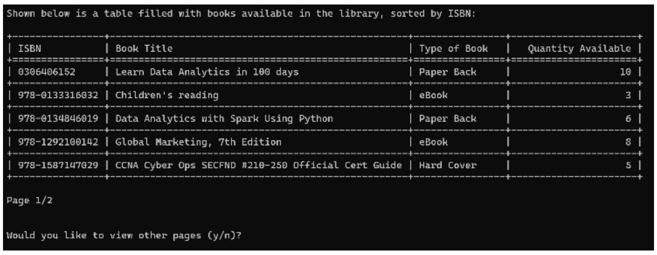


```
Shown below are ways in which the books can be sorted:
1. ISBN
2. Book Title
3. Type of Book
4. Quantity

Choose a method (1/2/3/4):
```

Fig. 4.2: Filter by Option

An option validator will run to validate if the input is 1, 2, 3 or 4. If the input is valid, the screen is cleared.

If the user inputs ‘1’, the books are categorised by ISBN in ascending order. If the user inputs ‘2’, books will be arranged in alphabetical order. If the user inputs ‘3’, the books will be categorised by book type. If the user inputs ‘4’, the books will be sorted by ascending quantity.



```
Shown below is a table filled with books available in the library, sorted by ISBN:

+----------------+----------------------------------------------------+---------------+----------------------+
| ISBN           | Book Title                                         | Type of Book  |   Quantity Available |
+================+====================================================+===============+======================+
| 0306406152     | Learn Data Analytics in 100 days                   | Paper Back    |                   10 |
+----------------+----------------------------------------------------+---------------+----------------------+
| 978-0133316032 | Children's reading                                 | eBook         |                    3 |
+----------------+----------------------------------------------------+---------------+----------------------+
| 978-0134846019 | Data Analytics with Spark Using Python             | Paper Back    |                    6 |
+----------------+----------------------------------------------------+---------------+----------------------+
| 978-1292100142 | Global Marketing, 7th Edition                      | eBook         |                    8 |
+----------------+----------------------------------------------------+---------------+----------------------+
| 978-1587147029 | CCNA Cyber Ops SECFND #210-250 Official Cert Guide | Hard Cover    |                    5 |
+----------------+----------------------------------------------------+---------------+----------------------+

Page 1/2

Would you like to view other pages (y/n)?
```

Fig. 4.3: Example of sorted books by ISBN

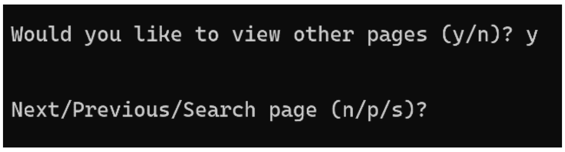
Note that the program shows the way in which the books are sorted (in Fig. 4.2, by ISBN).

As shown, each page consists of 5 books per page. The reason for this is there are many books in a library- we cannot display all of them on a single page, lest the system crashes.

The system prompts the user whether he/she wants to view more pages. A ‘y’/‘n’ validator runs here.

If ‘n’ or ‘N’ is keyed in as input, the program will prompt the user to choose between exiting the system and performing more actions in the system, where the ‘y’/‘n’ validator is used to validate the user input. If ‘y’ or ‘Y’ is entered, the program goes back to the Menu. The program exits if ‘n’ or ‘N’ is entered.

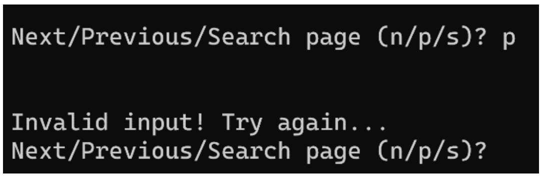
If ‘y’ or ‘Y’ is keyed in as an input in the ‘view more pages’ prompt, the following is shown:



```
Would you like to view other pages (y/n)? y


Next/Previous/Search page (n/p/s)?
```

Fig. 4.4: Page Function

An option validator runs to check whether the input is uppercase/lowercase ‘n’/‘p’/‘s’. Upon keying in ‘n’ or ‘p’, the program will check if moving to the next/previous page respectively is possible. If it is not possible, the program shows an error message as such:

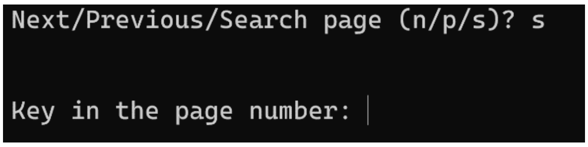


```
Next/Previous/Search page (n/p/s)? p


Invalid input! Try again...
Next/Previous/Search page (n/p/s)?
```

Fig 4.5: Checking page number with Action

Fig 4.3 occurs when the user is at page 1 and wants to move to the previous page, which is impossible as there is no page 0.

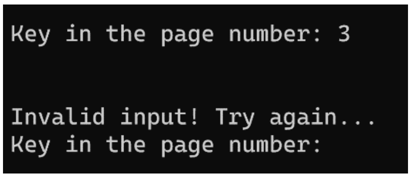
If ‘s’ is keyed in as the input, the program will prompt the user to key in a page number.



```
Next/Previous/Search page (n/p/s)? s


Key in the page number:
```

Fig. 4.6: Search Page Function

Here, the program will validate whether the page number inputted by the user is within the valid page range (in this case, 1-2).



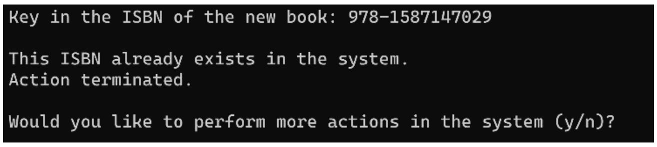
```
Key in the page number: 3


Invalid input! Try again...
Key in the page number:
```

Fig 4.7: Example of Page Number Validation

The program shows the appropriate page after keying in a valid page number.

Add new book(s)

Upon keying in '3' as the input in the Menu, the screen is cleared and the program prompts the user to key in the ISBN of the new book. Here, the ISBN validator runs. If the input is a valid ISBN, the program checks if the ISBN exists in the system. If it exists, the following is shown:

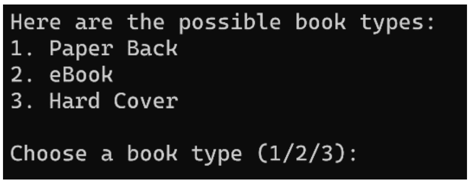


```
Key in the ISBN of the new book: 978-1587147029

This ISBN already exists in the system.
Action terminated.

Would you like to perform more actions in the system (y/n)?
```

Fig. 5.1: Example of Checking if inputted ISBN exists

This aims to ensure there're no duplicate ISBNs in the program. The user may exit the program or return to the menu page in the last prompt in Fig. 5.1.

Back to the first prompt which states to key in the ISBN of the new book. Suppose the user inputs a valid ISBN which does not exist within the system, the screen clears and prompts the user to key in a book title.. Upon keying in the book title, the screen clears and shows the following:

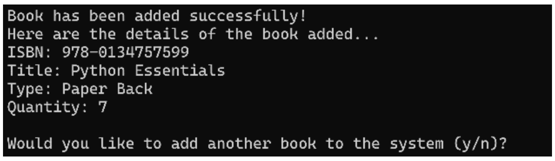


```
Here are the possible book types:
1. Paper Back
2. eBook
3. Hard Cover

Choose a book type (1/2/3):
```

Fig. 5.2: Prompting to Choose a Book Type

The choice validator runs to enter the input is either 1/2/3.

If the input passes the validation, the screen clears and the program prompts the user to 'Key in the number of books with this ISBN'. A validator runs to ensure the value is a numerical input. After clicking 'Enter', the program adds the book details to the system. The screen clears and shows the following:



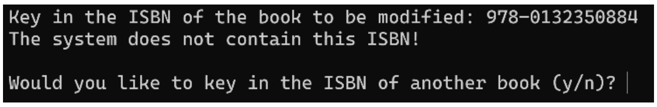
```
Book has been added successfully!
Here are the details of the book added...
ISBN: 978-0134757599
Title: Python Essentials
Type: Paper Back
Quantity: 7

Would you like to add another book to the system (y/n)?
```

Fig 5.3: Output of Adding a New Book

The choice validator runs to ensure the input is 'y'/'n'. If 'y' or 'Y' is keyed in as the input, the screen clears, and the program reruns the 'add new book(s)' function. If 'n' or 'N' is keyed in as input, the program will prompt the user to choose between exiting the system and performing more actions in the system, where the 'y'/'n' validator is used to validate the user input. If 'y' or 'Y' is entered, the program goes back to the Menu. The program exits if 'n' or 'N' is entered.

Update existing book(s)

Upon keying in '4' as the input in the Menu, the screen is cleared and the program prompts the user to "Key in the ISBN of the book to be modified: ". The ISBN validator runs here. If the ISBN does not exist in the system, the screen clears and shows the following:



```
Key in the ISBN of the book to be modified: 978-0132350884
The system does not contain this ISBN!

Would you like to key in the ISBN of another book (y/n)? |
```

Fig. 6.1: Non-existent ISBN validation

Option validator runs here. If 'Y' or 'y' is inputted, the screen will clear and the 'Update existing book(s)' function will rerun. If 'N' or 'n' is inputted, the program will ask whether the user wants to perform more actions in the system.

Suppose a valid ISBN within the system is inputted, then the screen will clear and ask the user "Would you like to modify the book's ISBN (y/n)? ". The option validator for 'y' /'n' runs here.

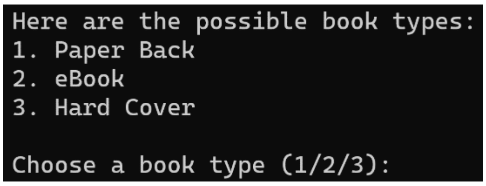
If 'N' or 'n' is inputted, the screen is cleared and shows the next prompt. If 'Y' or 'y' is inputted, the screen will clear and the program will ask the user to "Key in the new ISBN: " The ISBN validator checks the input for a valid ISBN. Upon successful input of ISBN, the screen clears and shows the next prompt.

The program then asks the user "Would you like to modify the book's Title (y/n)? ". The option validator for 'Y' / 'n' runs here.

If 'N' or 'n' is inputted, the screen is cleared and shows the next prompt. If 'Y' or 'y' is inputted, the screen will clear and the program will ask the user to "Key in the new Title: ". Upon entering the new title of the book, the screen clears and shows the next prompt.

The program then asks the user "Would you like to modify the book's Type (y/n)? "The option validator for 'Y' / 'n' runs here.

If 'N' or 'n' is inputted, the screen is also cleared and shows the next prompt. If 'Y' or 'y' is inputted, the screen will clear and show the following:



```
Here are the possible book types:
1. Paper Back
2. eBook
3. Hard Cover

Choose a book type (1/2/3):
```

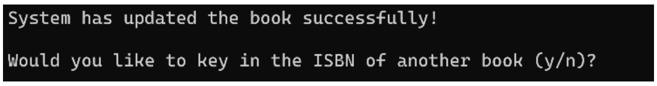
Fig. 6.2: Book Type Option

The option validator runs to check if the input is 1, 2 or 3. Upon entering the new type of book, the screen clears and shows the next prompt.

The program then asks the user "Would you like to modify the book's Quantity (y/n)? "The option validator for 'Y' / 'n' runs here.

If 'N' or 'n' is inputted, the screen is cleared and shows the next prompt. If 'Y' or 'n' is inputted, the screen clears and the program will ask the user to "Key in the new Quantity: ". Here, the program only accepts whole numbers (0, 1, 2, …). If a valid number is inputted, the screen clears and shows the next prompt.

Upon making the necessary updates, the program shows the following:



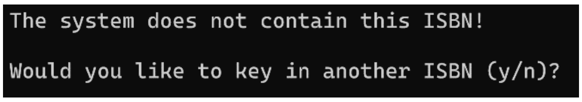
```
System has updated the book successfully!

Would you like to key in the ISBN of another book (y/n)?
```

Fig. 6.3: Update Successful Notification

The option validator runs here to check if the input is 'y'/'n'. If the user wants to key in another ISBN, the program clears the screen and replays the 'Update existing book(s)' action, prompting the user to key in another ISBN. If 'n' or 'N' is keyed in as input, the program will prompt the user to choose between exiting the system and performing more actions in the system, where the 'y'/'n' validator is used to validate the user input. If 'y' or 'Y' is entered, the program goes back to the Menu. The program exits if 'n' or 'N' is entered.

<u>Remove existing book(s)</u>

Upon keying in '5' as the input in the Menu, the screen is cleared and the program prompts the user to "Key in the ISBN of the book that you wish to remove: ". The ISBN validator runs here. If the ISBN does not exist in the system, the screen clears and shows the following:

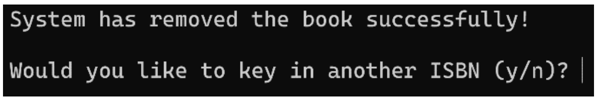


```
The system does not contain this ISBN!

Would you like to key in another ISBN (y/n)?
```

Fig. 7.1: Error Message when ISBN is non-existent

The option validator runs here to check if the input is 'y'/'n'. If the user wants to key in another ISBN, the program replays the 'Remove existing book(s)' action, prompting the user to key in another ISBN. If 'n' or 'N' is keyed in as input, the program will prompt the user to choose between exiting the system and performing more actions in the system, where the 'y'/'n' validator is used to validate the user input. If 'y' or 'Y' is entered, the program goes back to the Menu. The program exits if 'n' or 'N' is entered.

Back to the prompt which asks the user to key in the ISBN of the book he/she wants to remove. If the inputted ISBN exists within the system, the program deletes the book and clears the screen, showing the following:



```
System has removed the book successfully!

Would you like to key in another ISBN (y/n)?
```

Fig. 7.2: Successful removal of ISBN

The option validator runs here to check if the input is 'y'/'n'. If the user wants to key in another ISBN, the program clears the screen and replays the 'Remove existing book(s)' action, prompting the user to key in another ISBN. If 'n' or 'N' is keyed in as input, the program will prompt the user to choose between exiting the system and performing more actions in the system, where the 'y'/'n' validator is used to validate the user input. If 'y' or 'Y' is entered, the program goes back to the Menu. The program exits if 'n' or 'N' is entered.