Bookshop Management System

Mausumi Bhuyan

December 2024

Abstract:

This is a mini-bookshop management system project developed in the C++ language using file handling techniques to manage bookshop operations. The system provides two modes, Admin and Guest, where administrators can do operations such as adding new books, viewing, searching, deleting, and updating the information of the books, and guest can search for books by title or price range and purchase it.

Objectives:

The objective of this project is to create a simple and efficient way to manage book inventories. The system uses file-based data storage to store details that can be easily accessed and updated. It is designed to implement features such as adding, viewing, deleting, and updating, making management easy and effective.

Requirements:

- **C++ Compiler:** It will need a C++ compiler to run the code.
- **IDE(Integrated Development Environment):** An IDE is not strictly necessary to compile and run C++ code, but it can make the process easier. IDEs often include a text editor with features such as syntax highlighting and automatic identification, making the code easier to read and write.
- **C++ Standard Libaries:** This code uses several C++ standard libraries, including iostream, fstream, windows, string, etc.

Steps of the Entire Process:

- Created a text file to store the admin credentials and a csv file to store the book records.
- Declared a login() for both admin and guest users. For the admin log-in, the admin need to type their username and password for authentication, and for the guest log-in, guests can log in directly without credentials.
- Initially, the password entered by the administrator was visible, which raised privacy issues. To prevent others from seeing the password, I implemented a password asterisk(*) so that the password will not be visible.
- Then I declared adminmenu(), after logging in, the admin has the following operations.

  - Add a book.
  - Display all books.
  - Search for a book.
  - Delete a book.
  - Update the book details.

- These operations are defined in a separate file BookStucture.cpp where the book structure and functions are implemented.

- For addbook(), the admin can enter details for a new book: ID, Title, Author name, Category, Price and Quantity. The details are then added to the csv file.

- For displaybook(), it will display all the books currently present in the inventory and show the details of ID, title, author, category, price, and quantity of all the books.

- For searchbook(), it allows the admins to search for a book in the inventory.Initially, the search required exact matching of characters, including captilization and whitespace. To improve usability, i implemented a normalizestring() and computesimilarity(). Converts the input and stored data to lowercase and removes extra spaces for consistent matching. It iterates through the characters of both strings, counts the number of matching characters, and computes the similarity. This ensures that books can be found even if the input has minor differences in case or formatting. The admins can search for the book by title, category, or price range.

- For deletebook(), it allows the admins to enter the title of the book to be deleted. It also uses normalizestring() and computesimilarity() so that the book can be easily found in the case of formatting. By typing the title, the book details will be shown. If they want to delete the book ,by typing Y the book will automatically be removed from the inventory.

- For updatebook(), it allows the admins to update the details of any book in the inventory. Admins can modify fields such as quantity, price, or any other detail. If the admin finishes their tasks or doesn't want to perform additional operations, they can choose to exit.

- After declaring all the admin operations, I declared the guestmenu(), after logging in, the guest has the following operations for purchasing the books.
  - Search for a book by BookId.
  - Search for a book by price.

- By selecting the search for a book by BookId, it will first display all books present in the inventory. The guest is prompted to enter the bookid of which books they want to purchase and also the quantity. Suppose that the number of copies is not the number of copies they want, then it will show the text "Sorry only this much copies are available". The books which are selected will be added to the cart if they want to purchase any other books, they can type the bookid and it will be added to the cart. A bill will be generated showing the details of their purchase.

- By selecting the search for a book by price, they can choose the price range.
  - under 300
  - under 500
  - above 500

- By selecting the price range, it shows the books that fall under the selected price range. The guest can enter the bookid of the desired book and specify the quantity they wish to purchase. A bill is automatically generated with the purchase details.
- After completing their purchases, guests can exit if they do not want to purchase additional books. Similarly, the admin can exit after completing their operations.

Project Goals:

The system aims to provide user-friendly services to administrators and guests, ensuring ease of use. The project utilizes file handling to store and manage book records in a structured format, allowing for reliable and efficient retrieval and updates. Core functionalities include adding, searching, updating, and deleting book records along with handling guest purchases and generating detailed bills. To enhance the search experience, the system supports case-insensitive and whitespace-tolerant book searches, ensuring flexibility and accuracy.