PROJECT REPORT

Title: Library Book Management System

Submitted to:

Md. Rashid Al Asif

Assistant Professor

Department Of Computer Science & Engineering

University of Barisal

Submitted by:

Jakiya Jahan Soyna

Python Batch 06

Id: 03-006-18

Date of Submission: 18-12-2024

Project Report: Library Book Management System

1. Introduction

The Library Book Management System is designed to streamline the management of books in a library. It allows users to manage book records, track expenses, monitor returned and delivery books, and keep the library organized. The system is implemented in Python and provides a user-friendly interface for easy interaction.

2. Objectives

The primary objectives of the Library Book Management System are:

- To maintain a record of books along with their associated expenses.
- To provide an efficient way to add, delete, and view books.
- To track returned and delivery books separately.
- To calculate and display the total expense of all books.

3. Features

3.1 Core Features

1. Add a Book

   - Allows users to add a book along with its expense.

2. View Total Expense

   - Calculates and displays the total expense of all books in the library.

3. View Total Books and Names

   - Displays the total number of books and their names.

4. Delete a Book

   - Provides functionality to remove a book from the library database.

3.2 Additional Features

1. Add Returned Books

   - Maintains a list of books returned to the library.

2. Add Delivery Books

   - Tracks books delivered from the library.

3. View Returned Books

   - Displays all returned books in a structured format.

4. View Delivery Books

   - Lists all books delivered from the library.

4. Technologies Used

1. Programming Language: Python

2. Data Structures:

   - Dictionary for managing library books and their expenses.

   - Lists for tracking returned and delivery books.

3. Development Tools:

   - Python Integrated Development Environment (IDE)

   - Terminal for executing the program.

5. System Design

The system is menu-driven and provides the following options to the user:

1. Add a Book

2. View Total Expense

3. View Total Books and Names

4. Delete a Book

5. Add Returned Book

6. Add Delivery Book

7. View Returned Books

8. View Delivery Books

9. Exit

Data Flow:

1. Input:

   - User inputs book details such as name and expense.

   - Inputs for returned or delivery books.

2. Processing:

   - Adds book details to the dictionary or list.

   - Computes total expenses or counts of books as required.

3. Output:

   - Displays the requested information (total expense, book names, or list of returned/delivery books).

6. Code Implementation

Core Components

The implementation is divided into several functions:

1. add_book():

   - Adds a new book with its expense to the library database.

2. view_total_expense():

   - Sums and displays the total expenses of all books.

3. view_total_books_and_names() :

   - Displays the total number of books and their names.

4. delete_book() :

   - Deletes a specified book from the library.

5. add_returned_book() :

   - Adds a book to the returned books list.

6. add_delivery_book() :

   - Adds a book to the delivery books list.

7. view_books() :

   - Displays a list of books (returned or delivered).

7. Testing and Results

Test Cases

Action	Input	Expected Output
Add a book	Book: "Python"	Book added successfully.
View total expense	-	Displays the total expense
View total books and names	-	Displays book count and list of book names
Add returned book	Book: "Java"	Book added to returned books list.
Add returned book	-	Displays the returned books list.
Delete a book	Book: "Python"	Book deleted successfully.

Results

The system was tested with various inputs, and all functionalities performed as expected. Errors like invalid inputs (e.g., non-numeric expense values) were handled gracefully.

8. Conclusion

The Library Book Management System provides an efficient and straightforward solution for managing library books, tracking expenses, and organizing returned and delivery books. The modular structure ensures easy maintenance and scalability. This system can be enhanced further by integrating a database or adding a graphical user interface (GUI).

9. Future Enhancements

1. Database Integration:

   - Use a database like SQLite or MySQL to store book records persistently.

2. GUI:

   - Implement a graphical interface for better user interaction.

3. Search Functionality:

   - Add options to search for specific books.

4. Reporting:

   - Generate detailed reports of library transactions.

10. References

- Python Documentation: [https://docs.python.org] (https://docs.python.org)

- Stack Overflow: Community-driven Q&A for coding challenges.