

Library Management System

Python Console Application

Manas (main.py, utils.py)

Piyush (book_operations.py, manage_books.py)

Smit (helpers.py, storage.py)

November 3, 2025

Abstract

This document describes a simple Library Management System implemented in Python. The program is modularized across six files so that three team members can work in parallel. It supports adding, viewing, searching, deleting, and persisting book records (JSON). Example charts illustrate simple dataset summaries.

Contents

1	Introduction	3
2	Project Structure	3
3	How to Run	3
4	Code Listings	3
4.1	main.py	3
4.2	book_operations.py	4
4.3	manage_books.py	5
4.4	helpers.py	6
4.5	storage.py	6
4.6	utils.py	7
5	Sample Dataset (Example)	7
6	Data Visualization (Example Figures)	7
6.1	Bar Chart: Books by Year	7
6.2	Pie Chart: Distribution by Author	8
7	Sample Console Output	8
8	Future Enhancements	8
9	Acknowledgements	8

1 Introduction

The Library Management System is a lightweight console application that enables basic CRUD operations (Create, Read, Update, Delete) for book records, and persists data into a JSON file. The project demonstrates modular programming, file I/O, unique ID generation, and simple team collaboration via Git.

2 Project Structure

File	Purpose
main.py	Program entry point; menu and control flow.
book_operations.py	Add and view books.
manage_books.py	Search and delete books.
helpers.py	Small helper functions (confirmation, display).
storage.py	Load and save JSON data.
utils.py	Utility functions (ID generator).

3 How to Run

1. Upload all Python files into one folder (local or repository).
2. Ensure Python 3 is installed.
3. Run from terminal:

```
python main.py
```

4. Follow the on-screen menu.

4 Code Listings

Below are the source files. (You can copy these into ‘.py’ files.)

4.1 main.py

```
1 from book_operations import add_book, view_books
2 from manage_books import search_book, delete_book
3 from storage import save_data, load_data
4
5 def main():
6     books = load_data()
7
8     while True:
9         print("\n=== Library Management System ===")
10        print("1. Add Book")
11        print("2. View Books")
12        print("3. Search Book")
13        print("4. Delete Book")
14        print("5. Save & Exit")
15
16        choice = input("Enter your choice: ")
17
18        if choice == '1':
19            add_book(books)
20        elif choice == '2':
21            view_books(books)
22        elif choice == '3':
23            search_book(books)
24        elif choice == '4':
25            delete_book(books)
26        elif choice == '5':
27            save_data(books)
28            print("    Data saved! Exiting...")
29            break
30        else:
31            print("    Invalid choice, try again!")
32
33 if __name__ == "__main__":
34     main()
```

Listing 1: main.py

4.2 book_operations.py

```
1 from utils import generate_id
2
3 def add_book(books):
4     title = input("Enter book title: ")
5     author = input("Enter author name: ")
6     year = input("Enter publication year: ")
7     book_id = generate_id()
8
9     book = {"id": book_id, "title": title, "author": author, "
10            year": year}
11     books.append(book)
12     print("        Book added successfully!")
13
14 def view_books(books):
15     if not books:
16         print("No books found in the library.")
17         return
18     print("\n--- Library Books ---")
19     for b in books:
20         print(f"ID: {b['id']} | {b['title']} by {b['author']} ({b
21            ['year']})")
```

Listing 2: *book_operations.py*

4.3 manage_books.py

```
1 def search_book(books):
2     keyword = input("Enter title or author to search: ").lower()
3     results = [b for b in books if keyword in b["title"].lower()
4                or keyword in b["author"].lower()]
5     if results:
6         print("\n--- Search Results ---")
7         for b in results:
8             print(f"{b['id']} - {b['title']} by {b['author']} ({b
9                ['year']})")
10    else:
11        print("        No books found matching your search.")
12
13 def delete_book(books):
```

```
12     book_id = input("Enter book ID to delete: ")
13     for b in books:
14         if b["id"] == book_id:
15             books.remove(b)
16             print("          Book deleted successfully!")
17             return
18     print("      No book found with that ID.")
```

Listing 3: *manage_books.py*

4.4 helpers.py

```
1 def confirm_action(message):
2     choice = input(f"{message} (y/n): ").lower()
3     return choice == 'y'
4
5 def display_line():
6     print("=" * 40)
```

Listing 4: *helpers.py*

4.5 storage.py

```
1 import json
2 import os
3
4 FILE_NAME = "books.json"
5
6 def load_data():
7     if not os.path.exists(FILE_NAME):
8         return []
9     with open(FILE_NAME, "r") as f:
10         return json.load(f)
11
12 def save_data(books):
13     with open(FILE_NAME, "w") as f:
14         json.dump(books, f, indent=4)
```

Listing 5: *storage.py*

4.6 utils.py

```
1 import uuid
2
3 def generate_id():
4     return str(uuid.uuid4())[:8]
```

Listing 6: utils.py

5 Sample Dataset (Example)

Below is an example dataset (this is what might appear in `books.json`):

```
[
  {"id": "a1b2c3d4", "title": "Python Basics", "author": "John Doe", "year": "2023"},
  {"id": "e5f6g7h8", "title": "Data Science 101", "author": "Alice Smith", "year": "2023"},
  {"id": "i9j0k1l2", "title": "Intro to ML", "author": "Robert Lee", "year": "2022"}
]
```

6 Data Visualization (Example Figures)

These charts use the example dataset above and are included for report visuals.

6.1 Bar Chart: Books by Year

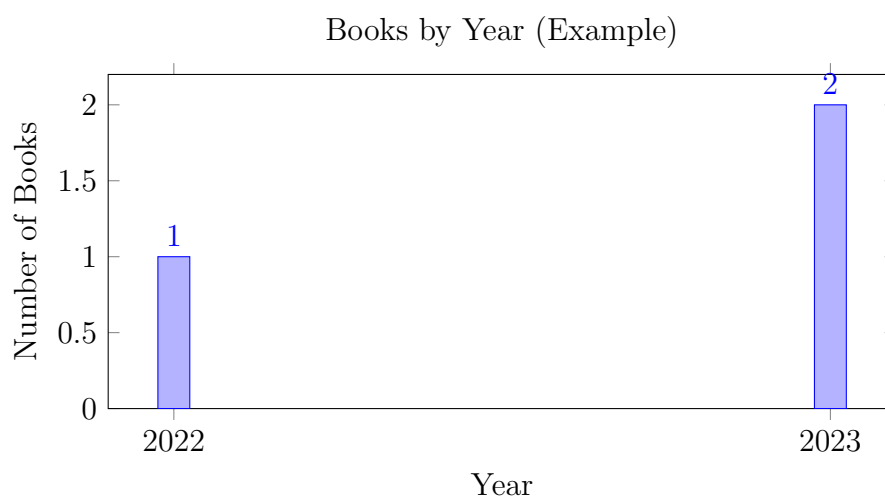


Figure 1: Example bar chart showing number of books by year.

6.2 Pie Chart: Distribution by Author

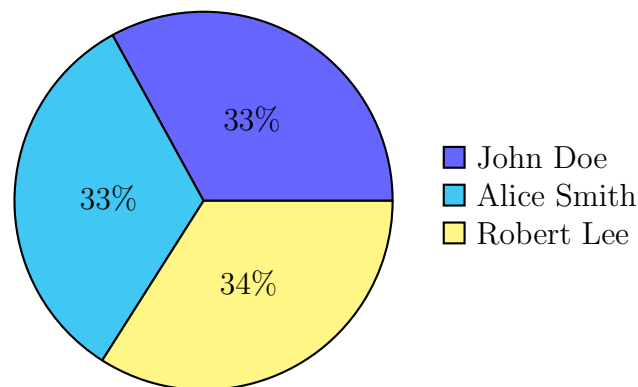


Figure 2: Example pie chart showing distribution of books by author.

7 Sample Console Output

```
=== Library Management System ===
1. Add Book
2. View Books
3. Search Book
4. Delete Book
5. Save & Exit
Enter your choice: 1
Enter book title: Python Basics
Enter author name: John Doe
Enter publication year: 2023
Book added successfully!
```

8 Future Enhancements

- Add GUI using Tkinter or PyQt.
- Implement user roles (admin, student).
- Add loan/return tracking and due dates.
- Add CSV/Excel import-export.

9 Acknowledgements

Thanks to Manas, Piyush, and Smit for collaborating on code and documentation.

Notes for Overleaf:

- This file uses `pgfplots` and `pgf-pie`. Overleaf includes these packages by default.
- If charts do not render, ensure the project compiles with PDFLaTeX and check package availability.