INDEX

1 Introduction

2 Objectives of the Project

3 Scope of the Project

4 Algorithm

5 Flow chart

6 Tools and Technologies Used

7 System Requirements

8 Project Description

9 Advantages of the System

10 Future Enhancements

11 Source code

12 screen short of running Project

13 Conclusion

1. Introduction

The Library Management – User Query Log System is a Python-based application designed to maintain a record of user search queries in a text file. This system helps librarians analyze user interests by tracking searches and identifying frequently searched topics such as Python programming.

The system supports viewing, adding, searching, and analyzing user queries using file handling and string processing.

---

2. Objectives of the Project

The main objectives of this project are:

- To maintain a log of user search queries
- To store queries permanently using file handling
- To perform case-insensitive search operations
- To count how many users searched for Python
- To provide a menu-driven interface for easy use

---

3. Scope of the Project

The scope of this project includes:

- Storing user queries in a text file
- Searching keywords in stored queries

- Counting Python-related searches
- Viewing complete search history
- Expanding later with database integration

This project is useful for libraries, online book portals, and learning platforms.

---

4. Algorithm

Algorithm for Library Query System:

1. Start
2. Load the query file
3. Display menu options
4. Accept user choice
5. Perform selected operation
6. Repeat until exit
7. Stop

---

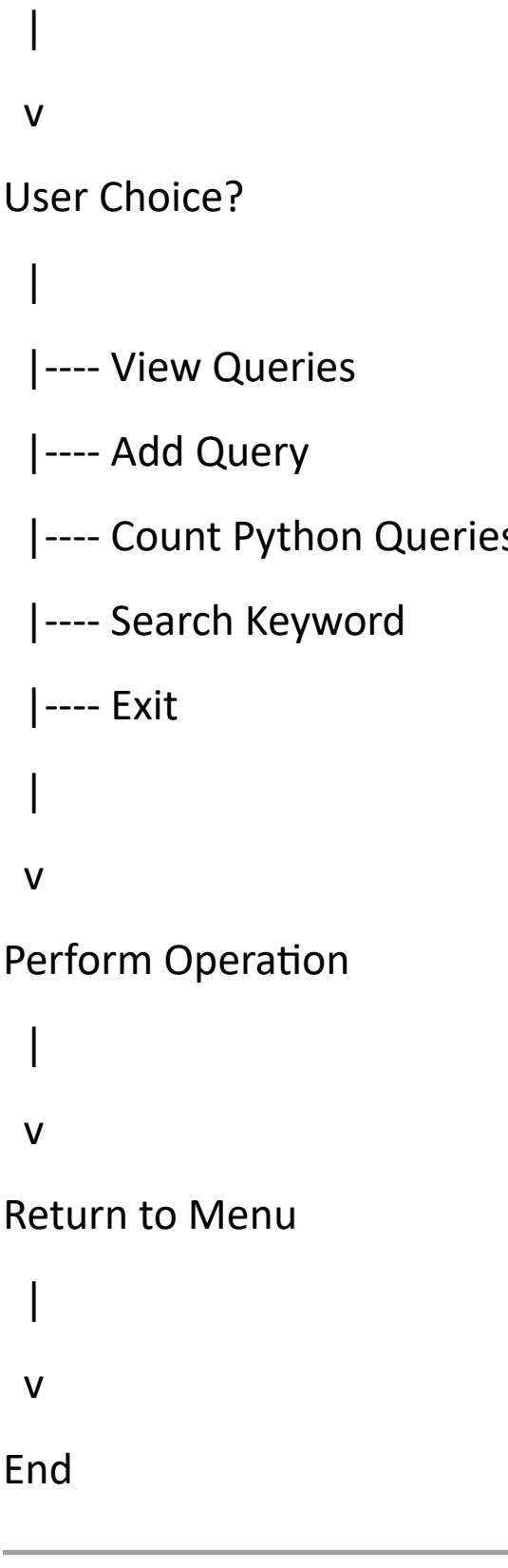
5. Flow Chart

Start

|

v

Display Menu


|
v
User Choice?
|
|---- View Queries
|---- Add Query
|---- Count Python Queries
|---- Search Keyword
|---- Exit
|
v
Perform Operation
|
v
Return to Menu
|
v
End

6. Tools and Technologies Used

- Programming Language: Python 3
- IDE: VS Code / PyCharm
- Operating System: Windows
- File Handling Module
- Command Line Interface

---

7. System Requirements

Hardware Requirements:

- Minimum 2GB RAM
- Intel i3 or above
- 10GB Free Disk Space

Software Requirements:

- Windows 10 or above
- Python 3.x
- Text Editor / IDE

---

8. Project Description

This project reads user search queries from a text file and performs operations such as:

- Adding new queries
- Searching keywords
- Counting Python-related searches
- Viewing all queries

It uses file handling, loops, functions, and string operations.

---

9. Advantages of the System

- Easy to use
- Stores data permanently
- Fast searching
- Case-insensitive matching
- Low memory usage
- No database required

---

10. Future Enhancements

- Database integration (MySQL / MongoDB)
- GUI interface using Tkinter
- Web version using Flask
- User login system
- Graphical analytics dashboard

---

11. Source Code

- **Project Files**

LibraryQuerySystem/

│

├── queries.txt

└── library_system.py

- **Sample queries.txt**

Python programming basics

PYTHON data science

Intro to java

advanced PYTHON

- **File: library_system.py**

```
# Library Management - User Query Log System

FILE_NAME = "queries.txt"

# Function to display all queries
def view_queries():
    print("\n--- User Search Queries ---")
    with open(FILE_NAME, "r") as file:
        queries = file.readlines()

    for i, q in enumerate(queries, start=1):
        print(f"{i}. {q.strip()}")

# Function to add a new query
def add_query():
    query = input("Enter new search query: ")
```

```
    with open(FILE_NAME, "a") as file:
        file.write("\n" + query)

    print(" Query added successfully!")

# Function to count python related queries
def count_python_queries():
    count = 0

    with open(FILE_NAME, "r") as file:
        queries = file.readlines()

    for q in queries:
        if "python" in q.lower():
            count += 1

    print(" 📊 Total queries containing 'python':", count)
```

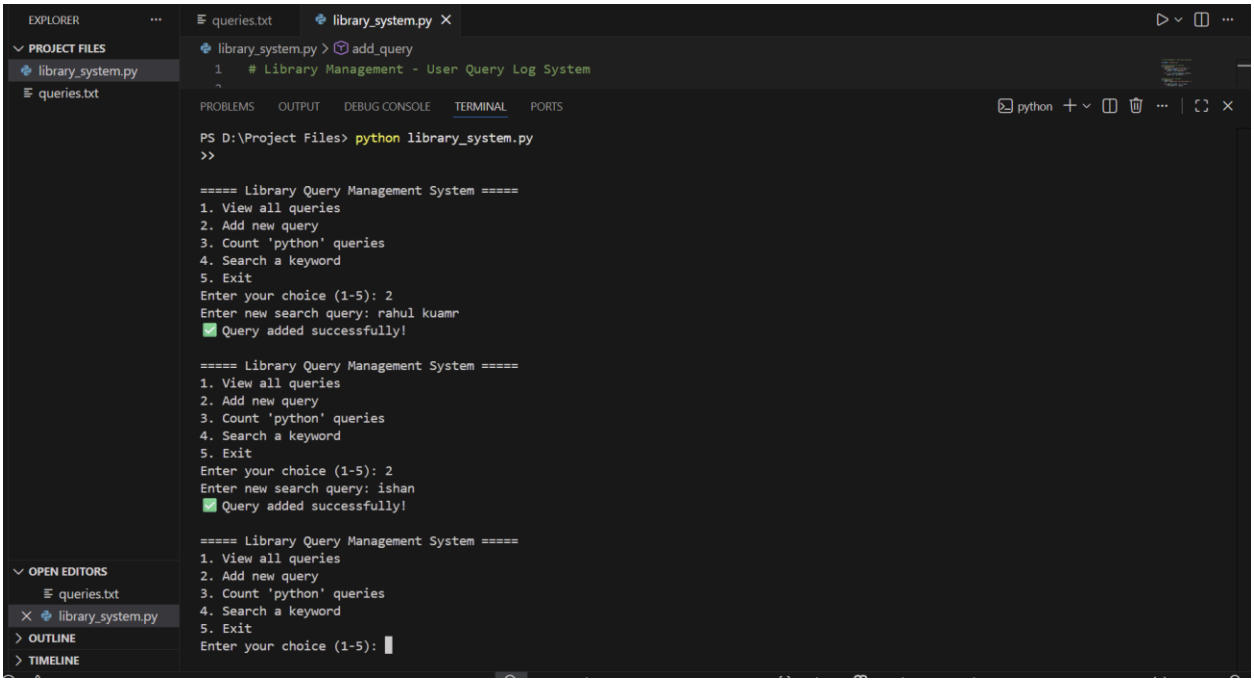
```
# Function to search any keyword
def search_keyword():
    keyword = input("Enter keyword to search: ").lower()

    print("\n🔍 Search Results:")
    found = False

    with open(FILE_NAME, "r") as file:
        for q in file:
            if keyword in q.lower():
                print("-", q.strip())
                found = True

    if not found:
        print(" No matching queries found.")

# Main Menu
def main_menu():
    while True:
```

```
    print("\n===== Library Query Management System =====")
    print("1. View all queries")
    print("2. Add new query")
    print("3. Count 'python' queries")
    print("4. Search a keyword")
    print("5. Exit")

    choice = input("Enter your choice (1-5): ")

    if choice == "1":
        view_queries()
    elif choice == "2":
        add_query()
    elif choice == "3":
        count_python_queries()
    elif choice == "4":
        search_keyword()
    elif choice == "5":
```

```
        print(" Exiting system. Thank you!")
        break
    else:
        print(" Invalid choice! Try again.")

# Run program
main_menu()
```

---

12. screen short of running Project



13. Conclusion

The Library Management – User Query Log System is a simple and efficient application for maintaining and analyzing library search data. It demonstrates the use of file handling, string manipulation, and menu-driven programming in Python. The system can be expanded into a full-scale library analytics platform.
