Week 2 Final Project: Library Management System 📚 Now that you've learned OOP, File Handling, Exception Handling, and Data Structures, let's apply these concepts in a real-world project: Library Management System. ✅ Project Goals: •Manage a library's book collection. •Allow users to borrow and return books. •Store book details using OOP and File Handling. •Handle errors (exception handling) and data persistence (files). 💻 Full Code Implementation

In [1]:
```
import os

class Book:
    """Class representing a book in the library."""
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.is_borrowed = False  # Track if the book is borrowed

    def display(self):
        """Returns book details as a string."""
        status = "Available" if not self.is_borrowed else "Borrowed"
        return f"📖 {self.title} by {self.author} ({status})"

class Library:
    """Class to manage the library system."""
    def __init__(self, filename="library.txt"):
        self.books = []  # List to store book objects
        self.filename = filename
        self.load_books()  # Load books from file

    def load_books(self):
        """Load books from a file (persistent storage)."""
        if os.path.exists(self.filename):
            with open(self.filename, "r") as file:
                for line in file:
                    title, author, status = line.strip().split(", ")
                    book = Book(title, author)
                    book.is_borrowed = (status == "Borrowed")
                    self.books.append(book)

    def save_books(self):
        """Save books to a file for persistence."""
        with open(self.filename, "w") as file:
            for book in self.books:
                status = "Borrowed" if book.is_borrowed else "Available"
                file.write(f"{book.title}, {book.author}, {status}\n")

    def add_book(self):
        """Add a new book to the library."""
        title = input("Enter book title: ")
        author = input("Enter author: ")
        self.books.append(Book(title, author))
        self.save_books()
        print(f"📚 '{title}' by {author} added successfully!")

    def display_books(self):
        """Show all available books in the library."""
        if self.books:
            print("\n📚 Library Books 📚")
```

```
            for i, book in enumerate(self.books, start=1):
                print(f"{i}. {book.display()}")
        else:
            print("No books in the library.")

    def borrow_book(self):
        """Allow a user to borrow a book."""
        self.display_books()
        if self.books:
            try:
                choice = int(input("\nEnter the book number to borrow: ")) - 1
                if 0 <= choice < len(self.books) and not self.books[choice].is_borr
                    self.books[choice].is_borrowed = True
                    self.save_books()
                    print(f"You borrowed '{self.books[choice].title}'. Enjoy readin
                else:
                    print("Invalid choice or book already borrowed!")
            except ValueError:
                print("Invalid input! Please enter a number.")

    def return_book(self):
        """Allow a user to return a borrowed book."""
        borrowed_books = [book for book in self.books if book.is_borrowed]
        if not borrowed_books:
            print("No borrowed books to return.")
            return

        print("\n📚 Borrowed Books 📚")
        for i, book in enumerate(borrowed_books, start=1):
            print(f"{i}. {book.display()}")

        try:
            choice = int(input("\nEnter the book number to return: ")) - 1
            if 0 <= choice < len(borrowed_books):
                borrowed_books[choice].is_borrowed = False
                self.save_books()
                print(f"Thank you for returning '{borrowed_books[choice].title}'!")
            else:
                print("Invalid choice!")
        except ValueError:
            print("Invalid input! Please enter a number.")

# Main Menu
library = Library()

while True:
    print("\n📖 Library Management System 📖")
    print("1. Add Book")
    print("2. View Books")
    print("3. Borrow Book")
    print("4. Return Book")
    print("5. Exit")

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

    if choice == "1":
```

```
            library.add_book()
        elif choice == "2":
            library.display_books()
        elif choice == "3":
            library.borrow_book()
        elif choice == "4":
            library.return_book()
        elif choice == "5":
            print("Exiting Library System. Goodbye!")
            break
        else:
            print("Invalid choice! Please enter a number between 1-5.")
```

```
📖 Library Management System 📖
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Exit
📚 'Wings of fire' by A P J Abdul kalam added successfully!

📖 Library Management System 📖
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Exit
📚 Library Books 📚
1. 📖 Wings of fire by A P J Abdul kalam (Available)
You borrowed 'Wings of fire'. Enjoy reading! 📖

📖 Library Management System 📖
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Exit
📚 Borrowed Books 📚
1. 📖 Wings of fire by A P J Abdul kalam (Borrowed)
Thank you for returning 'Wings of fire'!

📖 Library Management System 📖
1. Add Book
2. View Books
3. Borrow Book
4. Return Book
5. Exit
Exiting Library System. Goodbye!
```

🔍 Step-by-Step Explanation

📌 Step 1: Storing Books in a Class

python Copy Edit

```python
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.is_borrowed = False
```

- Each **Book** object stores its **title, author, and borrow status.**

📌 Step 2: Using a Library Class

python Copy Edit

```python
class Library:
    def __init__(self, filename="library.txt"):
        self.books = []
        self.filename = filename
        self.load_books()
```

- `Library` manages books using a **list**.
- Loads books from a **file** to keep data even after the program exits.

📌 Step 3: Adding a Book

python Copy Edit

```python
def add_book(self):
    title = input("Enter book title: ")
    author = input("Enter author: ")
    self.books.append(Book(title, author))
    self.save_books()
    print(f"📚 '{title}' by {author} added successfully!")
```

- Takes **user input** and adds a new **Book object** to the library.

📌 Step 4: Displaying Books

python Copy Edit

```python
def display_books(self):
    if self.books:
        print("\n📚 Library Books 📚")
        for i, book in enumerate(self.books, start=1):
            print(f"{i}. {book.display()}")
    else:
        print("No books in the library.")
```

- **Loops through the book list** and prints available books.

📌 Step 5: Borrowing a Book

python Copy Edit

```python
def borrow_book(self):
    self.display_books()
    if self.books:
        try:
            choice = int(input("\nEnter the book number to borrow: ")) - 1
            if 0 <= choice < len(self.books) and not self.books[choice].is_borrowed:
                self.books[choice].is_borrowed = True
                self.save_books()
                print(f"You borrowed '{self.books[choice].title}'. Enjoy reading! 📖")
            else:
                print("Invalid choice or book already borrowed!")
        except ValueError:
            print("Invalid input! Please enter a number.")
```

- Allows users to **borrow a book** and updates the borrow status.

📌 Step 6: Returning a Book

python Copy Edit

```python
def return_book(self):
    borrowed_books = [book for book in self.books if book.is_borrowed]
    if not borrowed_books:
        print("No borrowed books to return.")
        return
```

- Finds **borrowed books** and allows users to return them.

📖 Library Management System 📖 1. Add Book 2. View Books 3. Borrow Book 4. Return Book 5. Exit Enter your choice (1-5): 1 Enter book title: Python Basics Enter author: John Doe 🎉 'Python Basics' by John Doe added successfully! Enter your choice (1-5): 2 📚 Library Books 📚 1. 📖 Python Basics by John Doe (Available) Enter your choice (1-5): 3 Enter the book number to borrow: 1 You borrowed 'Python Basics'. Enjoy reading! 📖 📌 Summary of Week 2 Final Project ✔ Applied all Python concepts learned this week. ✔ Built a real-world application using OOP & File Handling. ✔ Practiced Exception Handling & Data Persistence.