Solution to Library Management System:

Here is a Python program that implements the case study, demonstrating the basic OOP concepts.

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

    def __str__(self):
        return f"Title: {self.title}, Author: {self.author}, Available Copies: {self.copies}"

    def is_available(self):
        return self.copies > 0


class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        self.books.append(book)
        print(f"Book '{book.title}' added to the library.")

    def borrow_book(self, title):
        for book in self.books:
            if book.title == title:
                if book.is_available():
                    book.copies -= 1
                    print(f"You have successfully borrowed '{title}'.")
                else:
                    print(f"Sorry, '{title}' is currently unavailable.")
                return
        print(f"Book '{title}' not found in the library.")
    def return_book(self, title):
        for book in self.books:
            if book.title == title:
                book.copies += 1
                print(f"Thank you for returning '{title}'.")
                return
        print(f"Book '{title}' does not belong to this library.")

    def list_books(self):
        print("\nLibrary Collection:")
        if not self.books:
            print("No books available in the library.")
            return
        for book in self.books:
            print(book)
        print()


# Simulate the Library Management System
if __name__ == "__main__":
    # Create a Library
    library = Library()

    # Add Books
    library.add_book(Book("The Great Gatsby", "F. Scott Fitzgerald", 3))
    library.add_book(Book("1984", "George Orwell", 5))
    library.add_book(Book("To Kill a Mockingbird", "Harper Lee", 2))

    # List Books
    library.list_books()

    # Borrow a Book
    library.borrow_book("1984")

    # List Books Again
    library.list_books()
    # Return a Book
    library.return_book("1984")

    # List Books Again
    library.list_books()
```

Sample Output:

```
Book 'The Great Gatsby' added to the library.
Book '1984' added to the library.
Book 'To Kill a Mockingbird' added to the library.

Library Collection:
Title: The Great Gatsby, Author: F. Scott Fitzgerald, Available Copies: 3
Title: 1984, Author: George Orwell, Available Copies: 5
Title: To Kill a Mockingbird, Author: Harper Lee, Available Copies: 2

You have successfully borrowed '1984'.

Library Collection:
Title: The Great Gatsby, Author: F. Scott Fitzgerald, Available Copies: 3
Title: 1984, Author: George Orwell, Available Copies: 4
Title: To Kill a Mockingbird, Author: Harper Lee, Available Copies: 2

Thank you for returning '1984'.

Library Collection:
Title: The Great Gatsby, Author: F. Scott Fitzgerald, Available Copies: 3
Title: 1984, Author: George Orwell, Available Copies: 5
Title: To Kill a Mockingbird, Author: Harper Lee, Available Copies: 2
```

Explanation:

- The program uses **classes and objects** to model the real-world entities of `Book` and `Library`.
- **Encapsulation** is demonstrated by bundling attributes and methods together within classes.
- **Abstraction** is achieved through meaningful methods like `borrow_book`, `return_book`, and `list_books`, which hide implementation details.
- The example demonstrates OOP principles effectively while providing clear and reusable code.