Task Title: Building a Library Management System in Python

Task Description:

You are tasked with creating a Python program that simulates a basic **Library Management System**. The system will allow users (Library Members) to borrow and return books, while the system tracks borrowed books for each member. Additionally, there will be different types of members (e.g., regular and premium members), and premium members will have privileges like borrowing more books. The program will incorporate concepts like classes, objects, inheritance, method overriding, and collections.

Deliverables:

1. **Python Script 1** (`library_system.py`):
    - This script will handle user interaction.
    - Users can borrow books, return books, and check their borrowed books.
    - It should import functions and classes from the second script (`library.py`).
    - Use a loop to continuously interact with the user and handle possible errors (e.g., borrowing limits, invalid book selections).
2. **Python Script 2** (`library.py`):
    - This script will contain the core logic for managing books and members.
    - Implement classes for `Book`, `LibraryMember`, and `PremiumMember`.
    - Include methods to borrow books, return books, and display the list of borrowed books.
    - Use inheritance for different member types (regular and premium).
3. **Optional Script** (`utils.py` - Optional):
    - This script can include utility functions like input validation, handling edge cases, and other helper functions to keep the code modular.

Requirements List:

1. **Classes and Objects**:
    - Define classes such as `Book`, `LibraryMember`, and `PremiumMember`.
    - Use attributes to store details like the name, ID, and borrowed books for each member.
    - Implement the `borrow_book()` and `return_book()` methods to manage borrowing and returning books.
2. **Inheritance and Method Overriding**:
    - Inherit from the `LibraryMember` class to create `PremiumMember` class with enhanced borrowing privileges.
    - Override the `borrow_book()` method in the `PremiumMember` class to allow borrowing more books than regular members.

3. **Collections**:
    - Use a list to store available books and borrowed books for each member.
    - Use a dictionary to map member IDs to their respective member objects.
4. **Error Handling**:
    - Use try-except blocks in the `library_system.py` script to handle cases like exceeding the borrowing limit or trying to return a book that wasn't borrowed.
    - Handle invalid inputs (e.g., when selecting a book that doesn't exist).
5. **Functions**:
    - `borrow_book(book)` – Adds a book to the member's borrowed list (in `library.py`).
    - `return_book(book)` – Removes a book from the member's borrowed list (in `library.py`).
    - `check_borrowed_books()` – Displays the list of books currently borrowed by the member (in `library.py`).

File Structure:

bash
CopyEdit

```
library_management/
│
├── library_system.py    # Handles user interaction
├── library.py           # Contains core library functions and member
classes
└── utils.py             # Optional: Contains helper functions (e.g.,
validation)
```

Example User Test:

User interaction example in `library_system.py`:

```
Enter 'b' to borrow a book, 'r' to return a book, 'c' to check
borrowed books, or 'exit' to quit: b
Enter book title to borrow: 1984
Borrowed: 1984

Enter 'b' to borrow a book, 'r' to return a book, 'c' to check
borrowed books, or 'exit' to quit: c
Books borrowed: 1984
```

```
Enter 'b' to borrow a book, 'r' to return a book, 'c' to check
borrowed books, or 'exit' to quit: r
Enter book title to return: 1984
Returned: 1984

Enter 'b' to borrow a book, 'r' to return a book, 'c' to check
borrowed books, or 'exit' to quit: exit
Exiting... Goodbye!
```