**Đề bài:** Trong một dự án phần mềm, khi có sự thay đổi về tình trạng hoặc trạng thái công việc (task), các thành viên trong nhóm sẽ nhận được thông báo tự động để theo dõi tiến độ. Hãy áp dụng Observer Design Pattern vào các trường hợp trên   
Yêu cầu: vẽ sơ đồ trước khi viết code.

**<<interface>>**

**Observer**

**--------------------------------**

**+ update(task : Task)**

**▲**

**|**

**----------------------------------------------**

**| |**

**TeamMember TeamMember**

**- name : String - name : String**

**+ update(task) + update(task)**

**<<interface>>**

**Subject**

**--------------------------------**

**+ attach(o : Observer)**

**+ detach(o : Observer)**

**+ notifyObservers()**

**▲**

**|**

**Task**

**---------------------------------------**

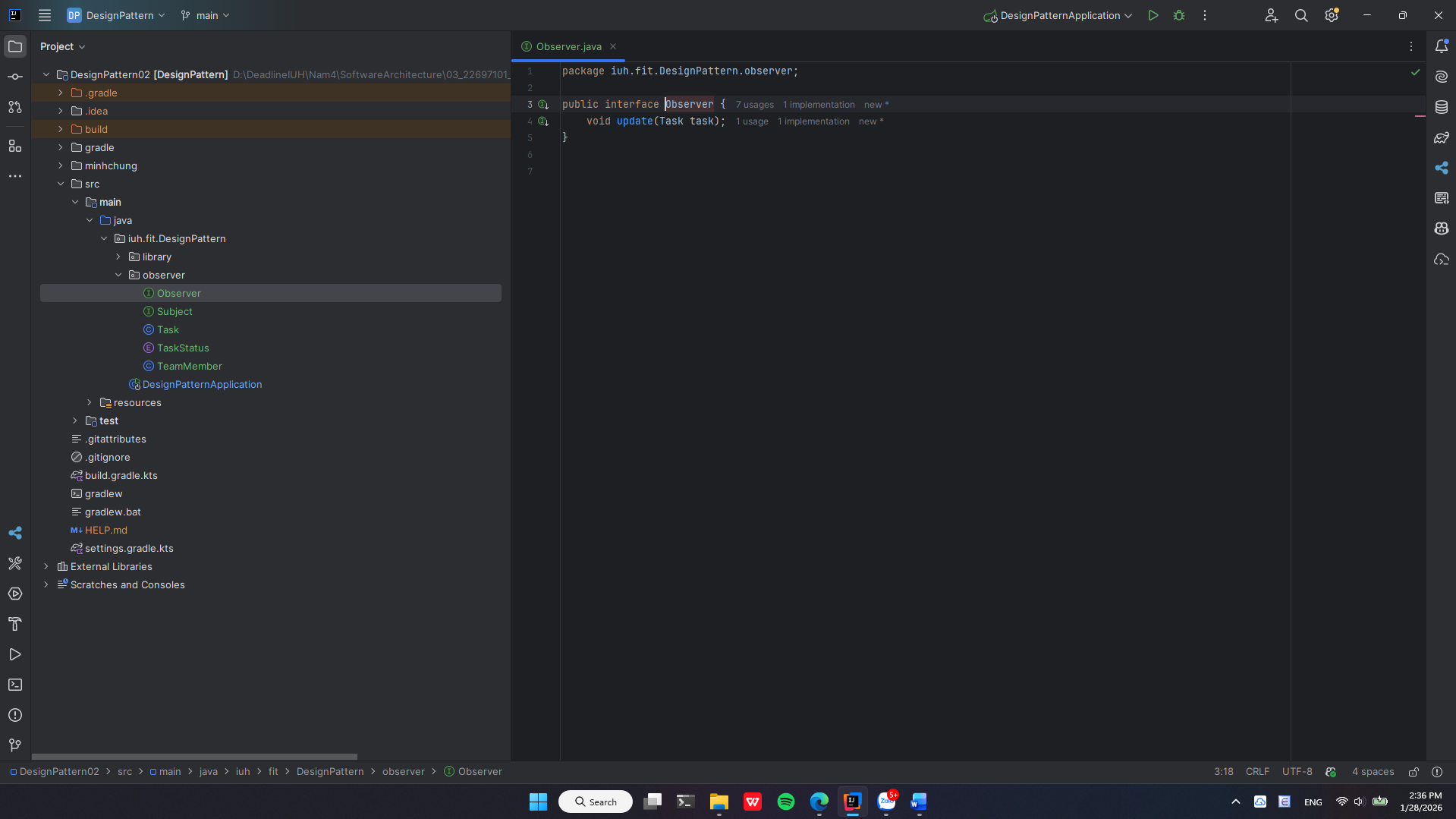
**- title : String**

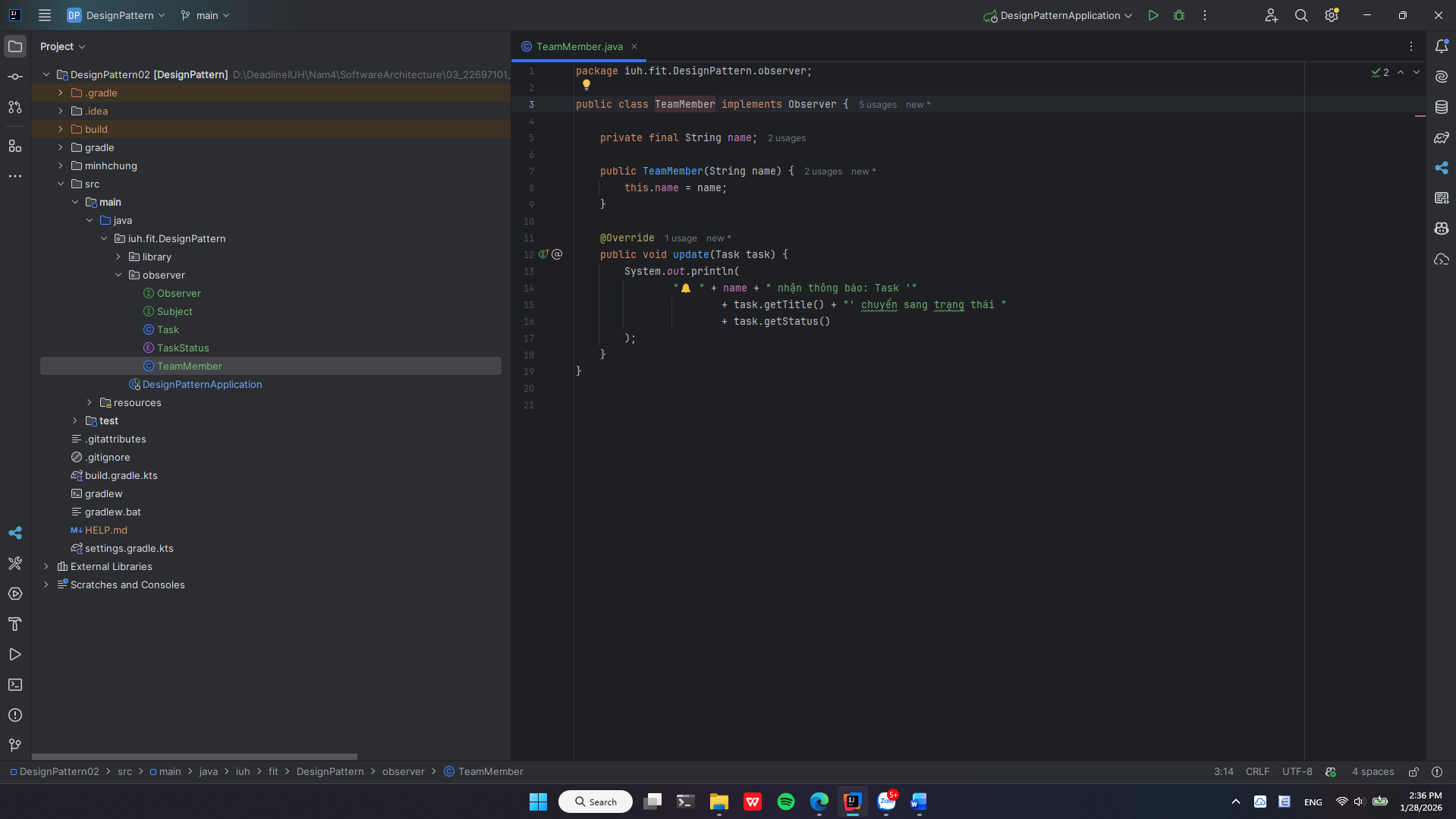
**- status : TaskStatus**

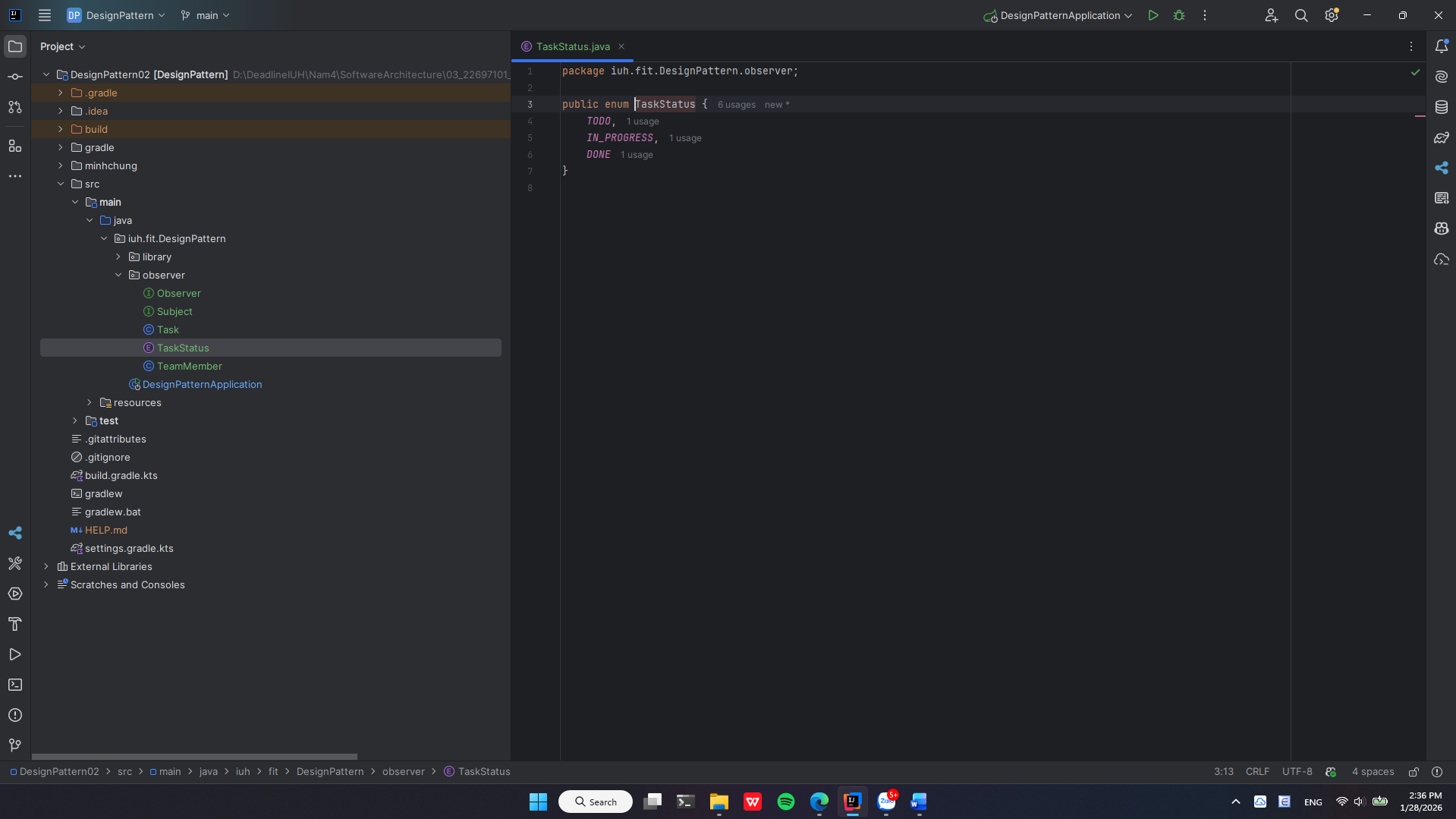
**- observers : List<Observer>**

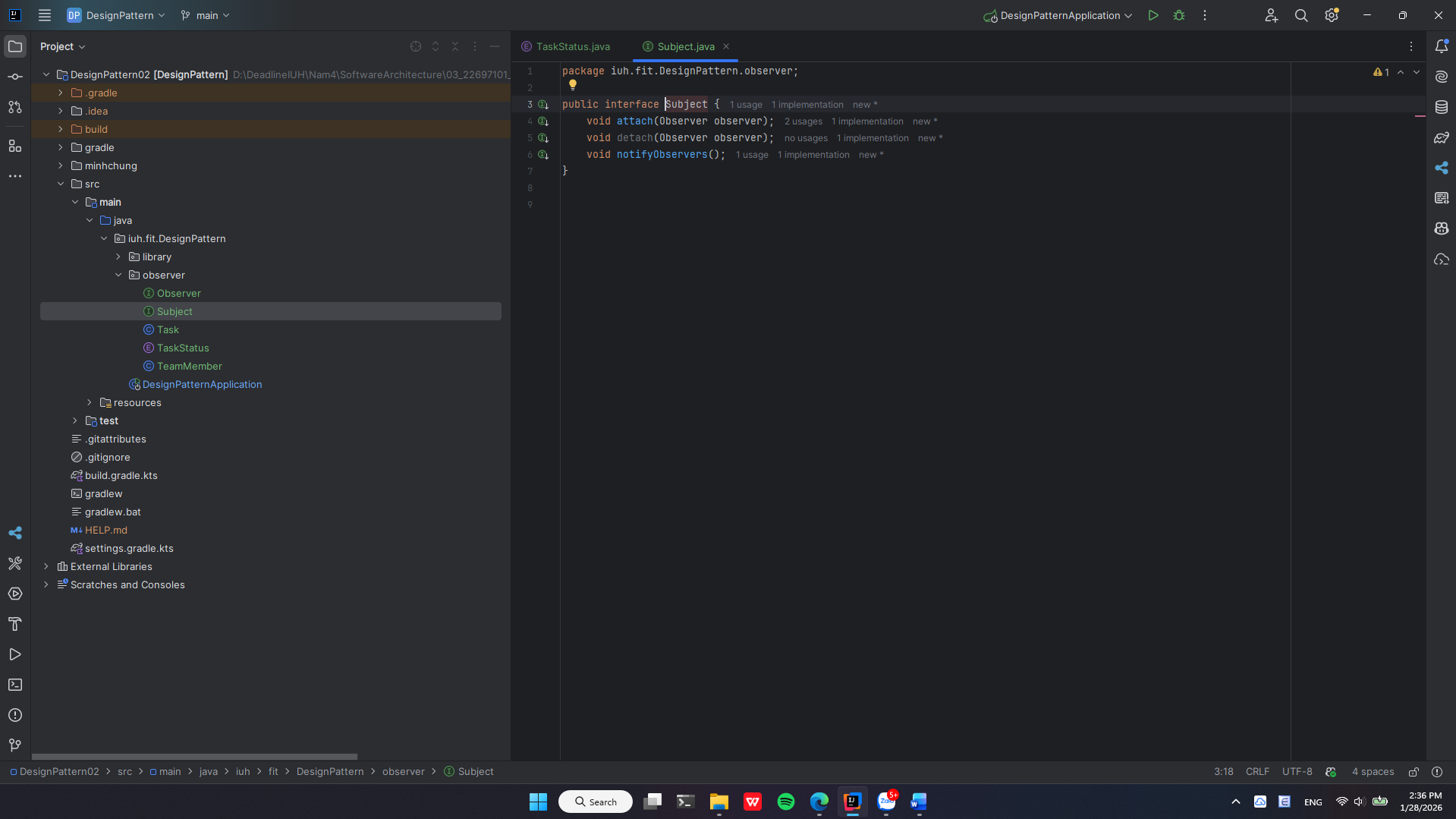
**+ setStatus(status)**

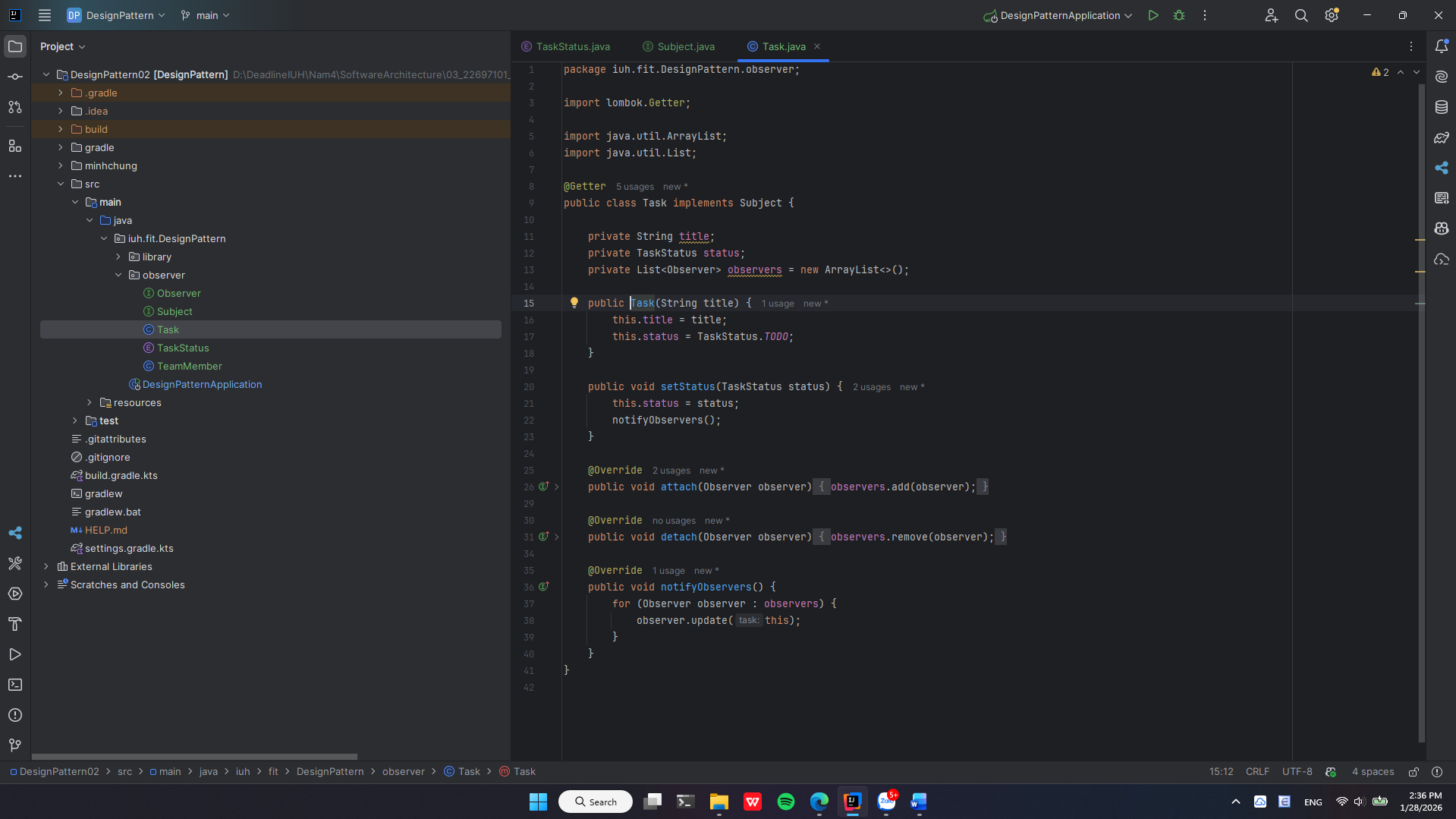
**+ notifyObservers()**

****

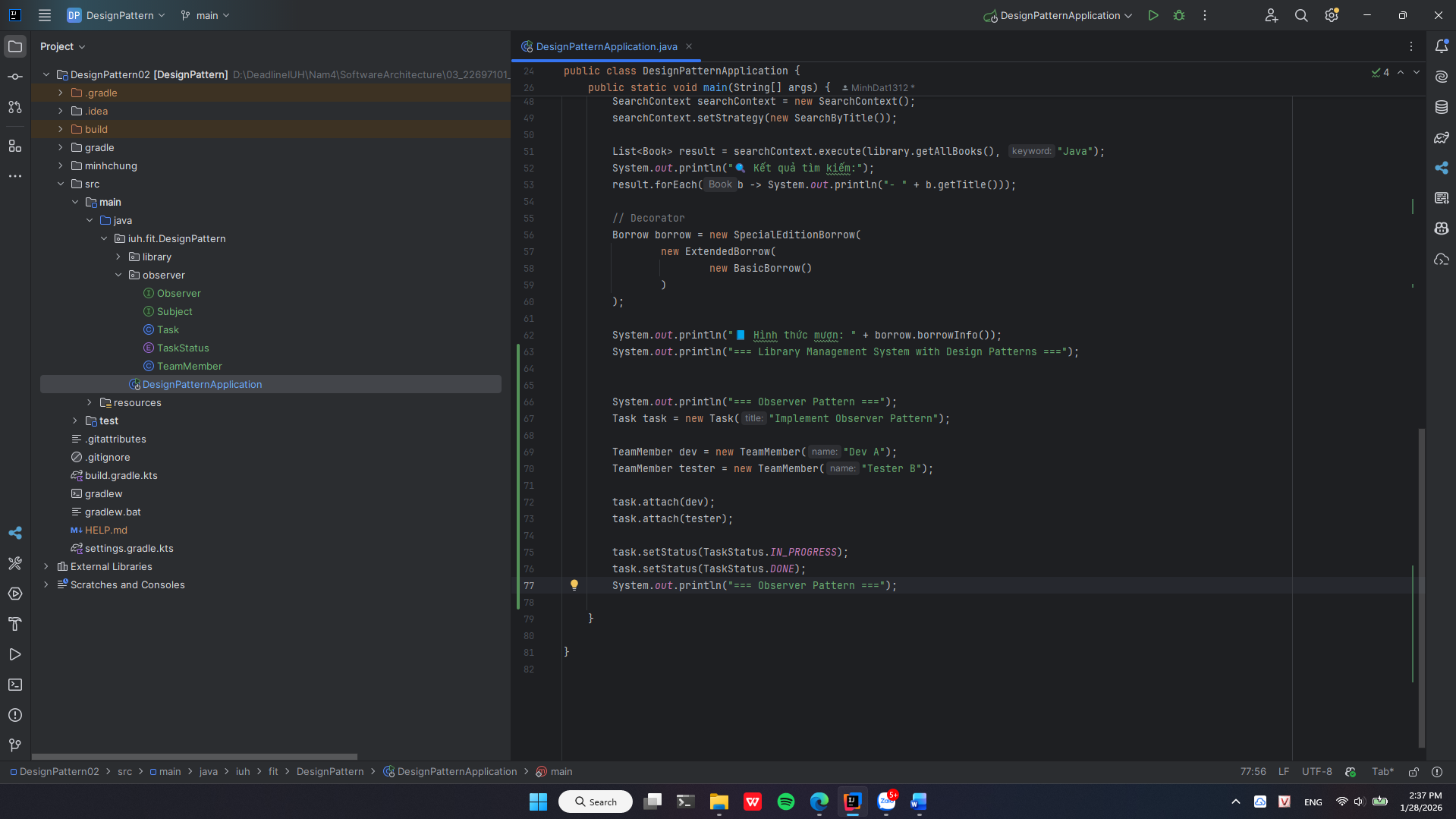
****

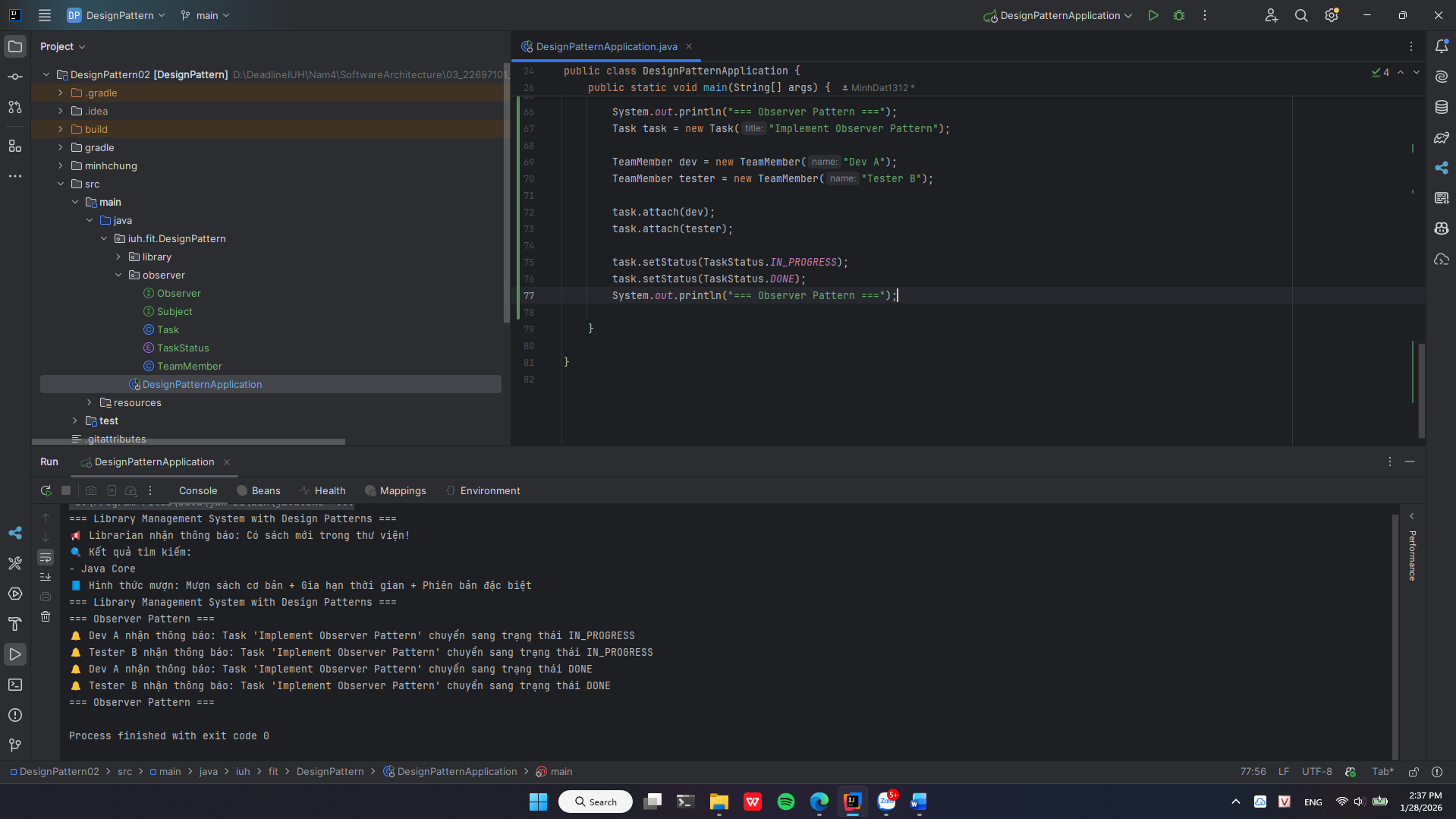
****

****

****

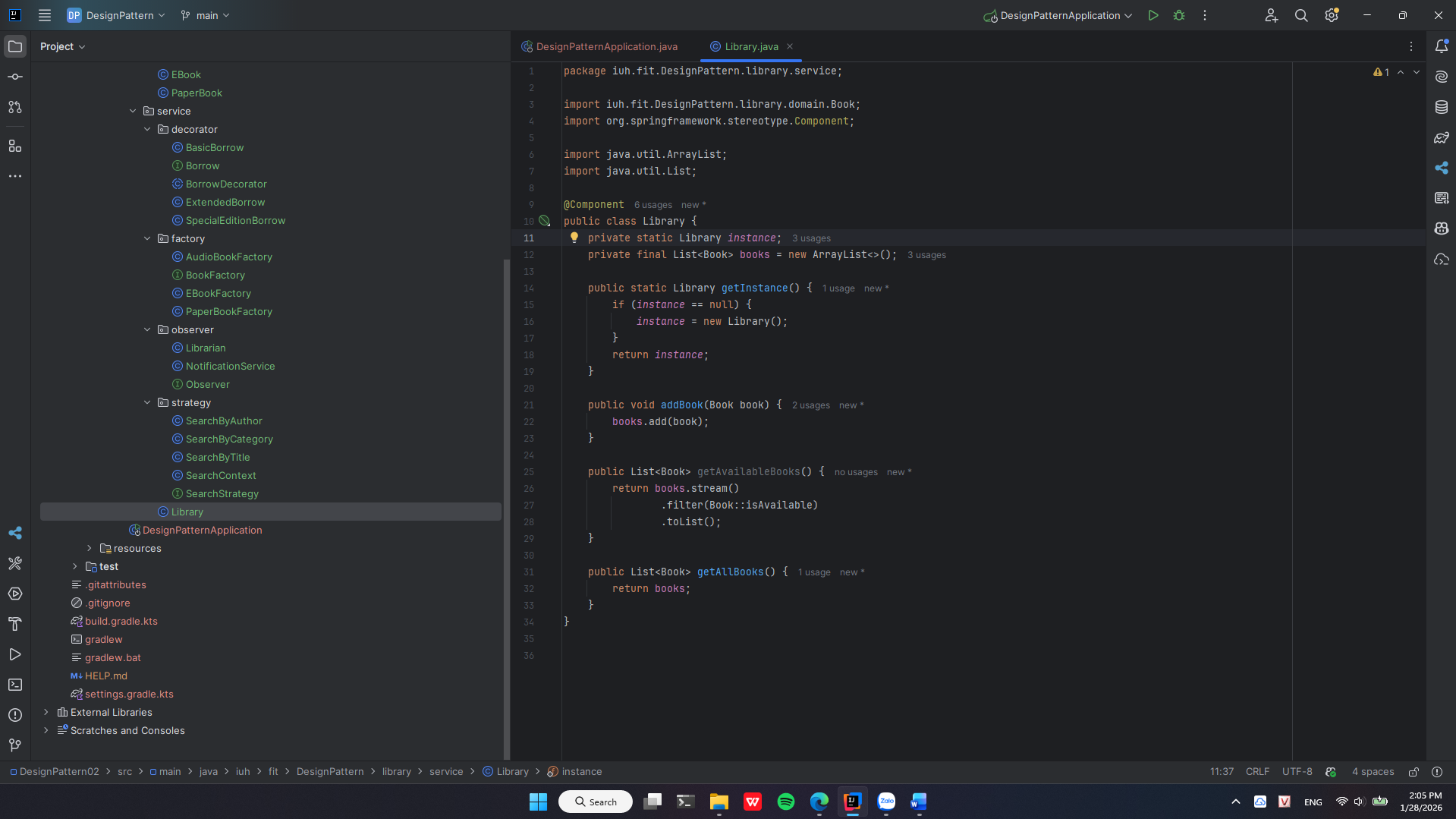
**Kết quả chạy:**

****

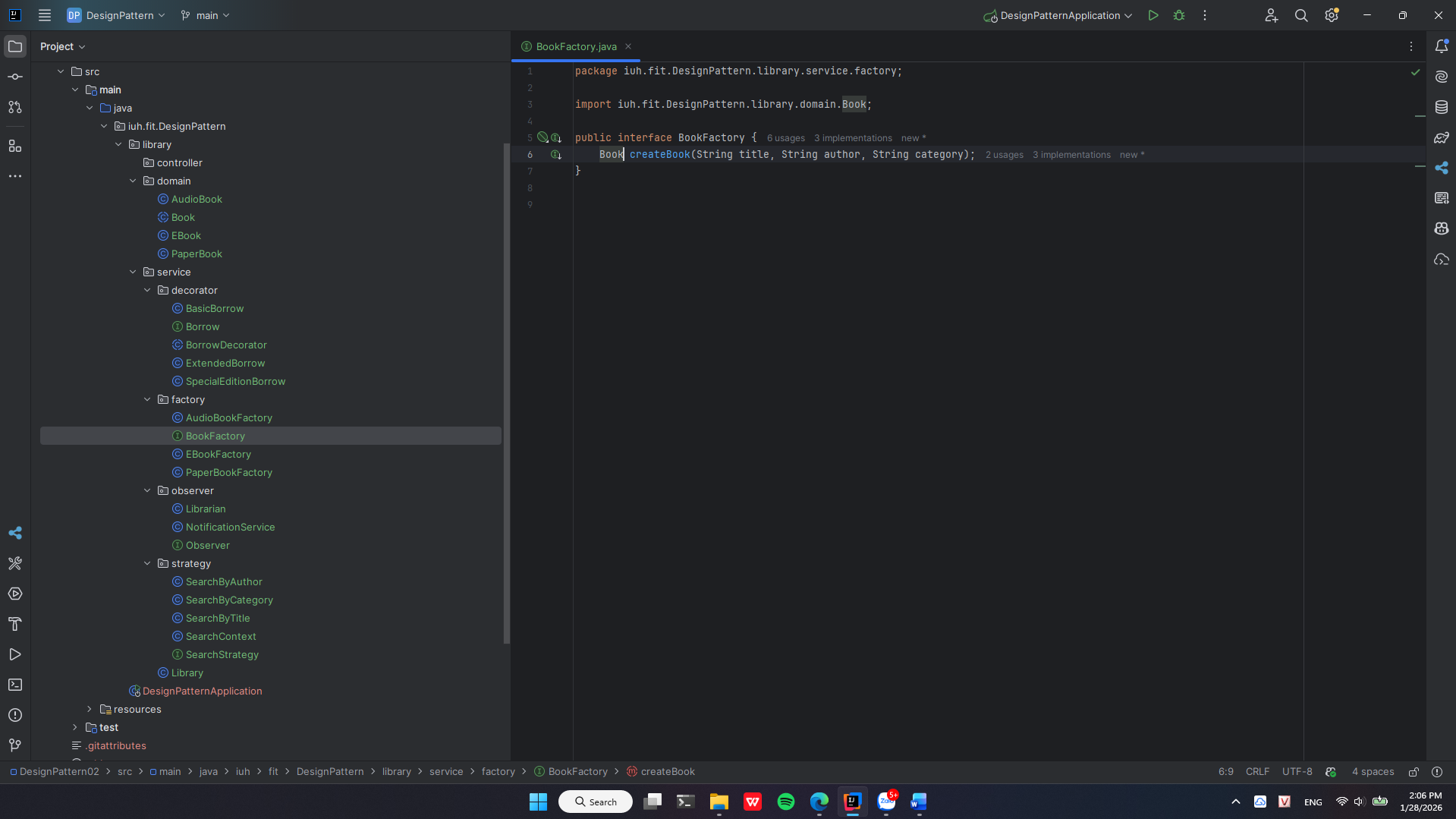
****

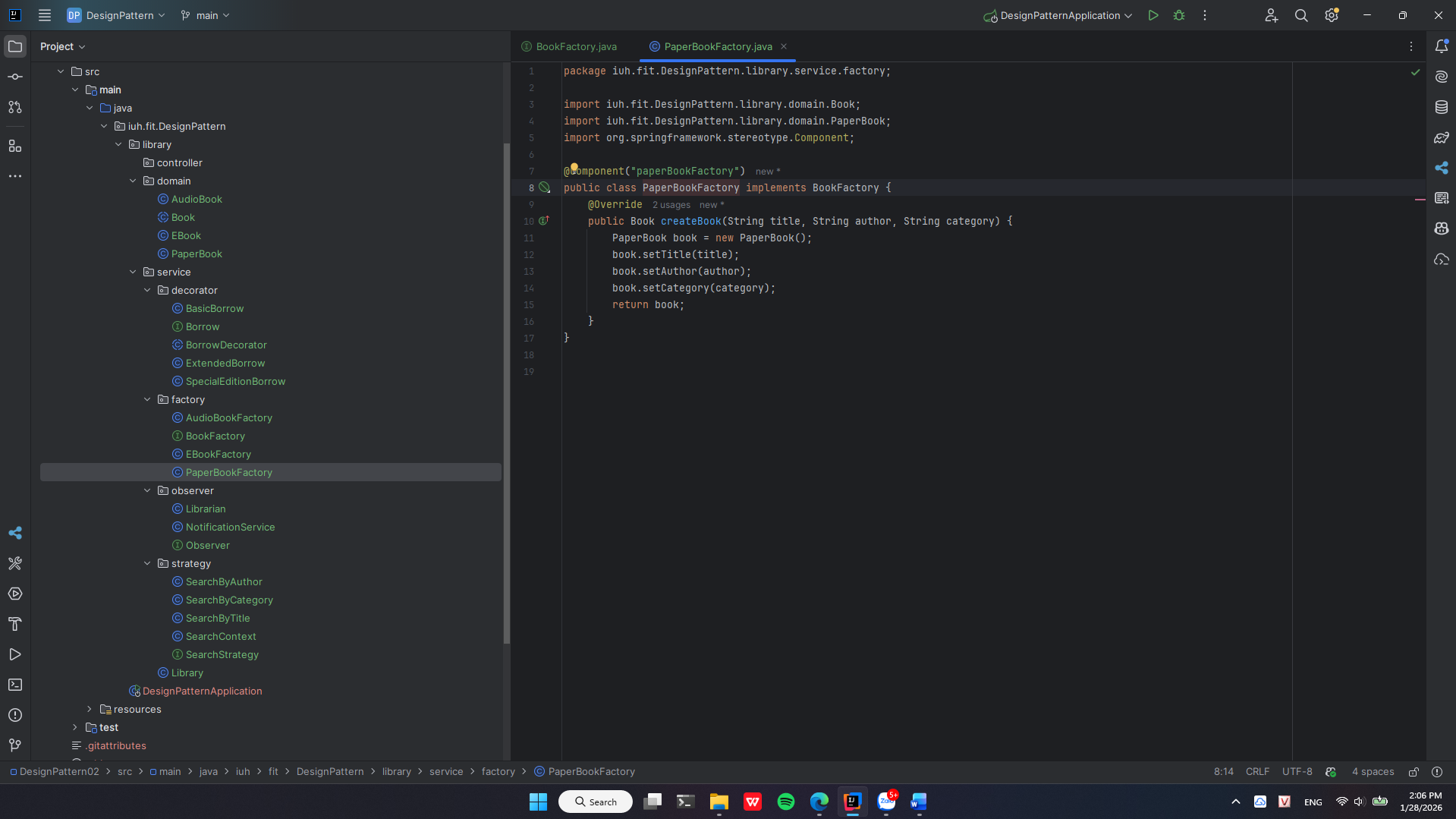
**Đề bài:** Bạn đang xây dựng một hệ thống quản lý thư viện. Hệ thống này sẽ cho phép người dùng thực hiện các chức năng cơ bản như: mượn sách, trả sách, thêm sách mới vào thư viện, xem danh sách sách có sẵn, tìm kiếm sách theo tên, tác giả và thể loại. Hệ thống sẽ được phát triển với nhiều thành phần và có thể mở rộng trong tương lai. Hãy sử dụng các Design Patterns phù hợp để xây dựng hệ thống này. Các yêu cầu chi tiết như sau:

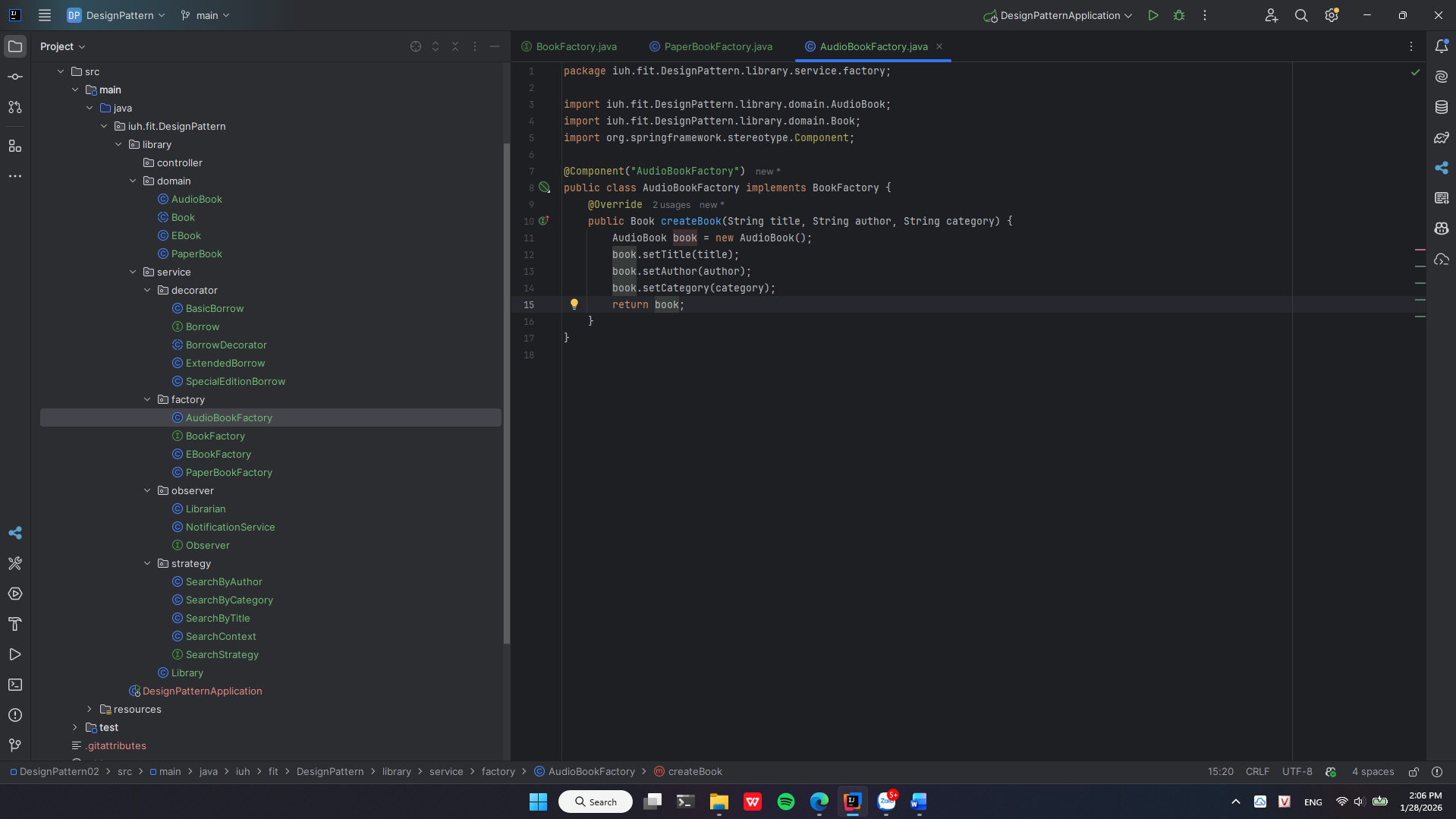
**1. Singleton Pattern:** Xây dựng một đối tượng Library để quản lý tất cả các sách trong thư viện. Đảm bảo rằng chỉ có một đối tượng duy nhất của Library trong hệ thống (chỉ có một thư viện duy nhất).



**2. Factory Method Pattern:** Khi thêm sách mới vào thư viện, bạn có thể chọn loại sách (sách giấy, sách điện tử, sách nói, v.v.). Hãy sử dụng Factory Method để tạo ra các loại sách khác nhau.

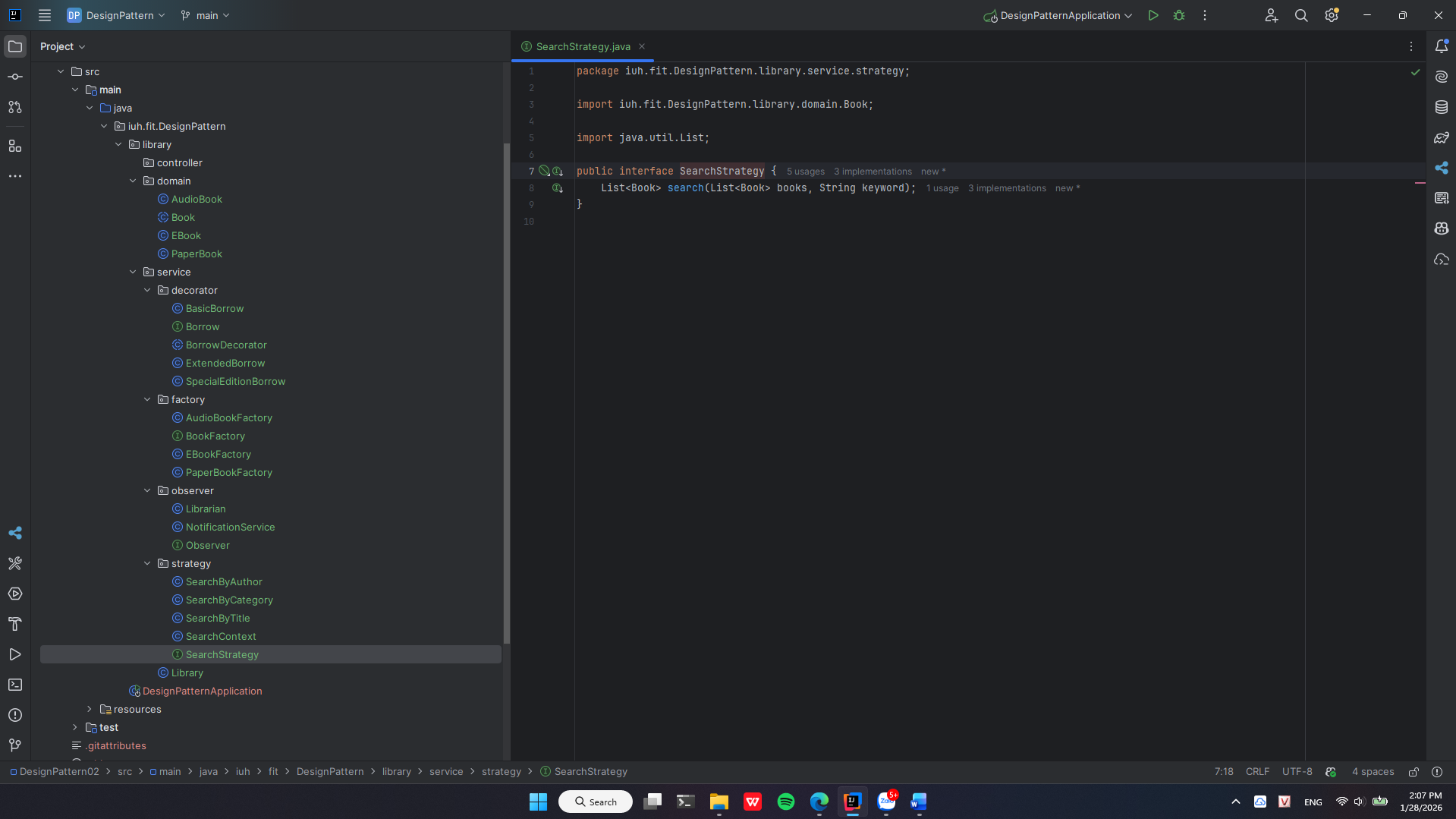


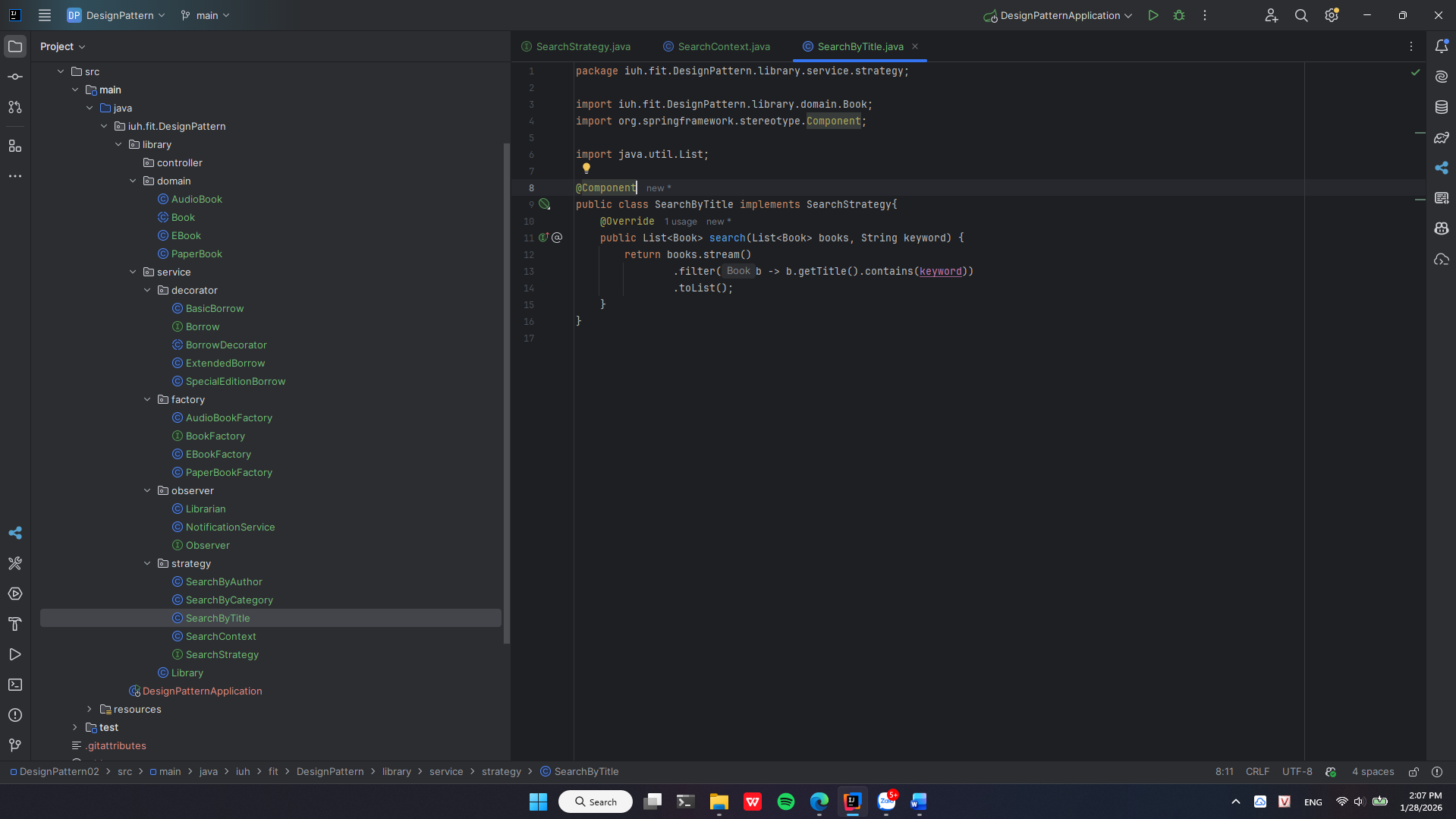


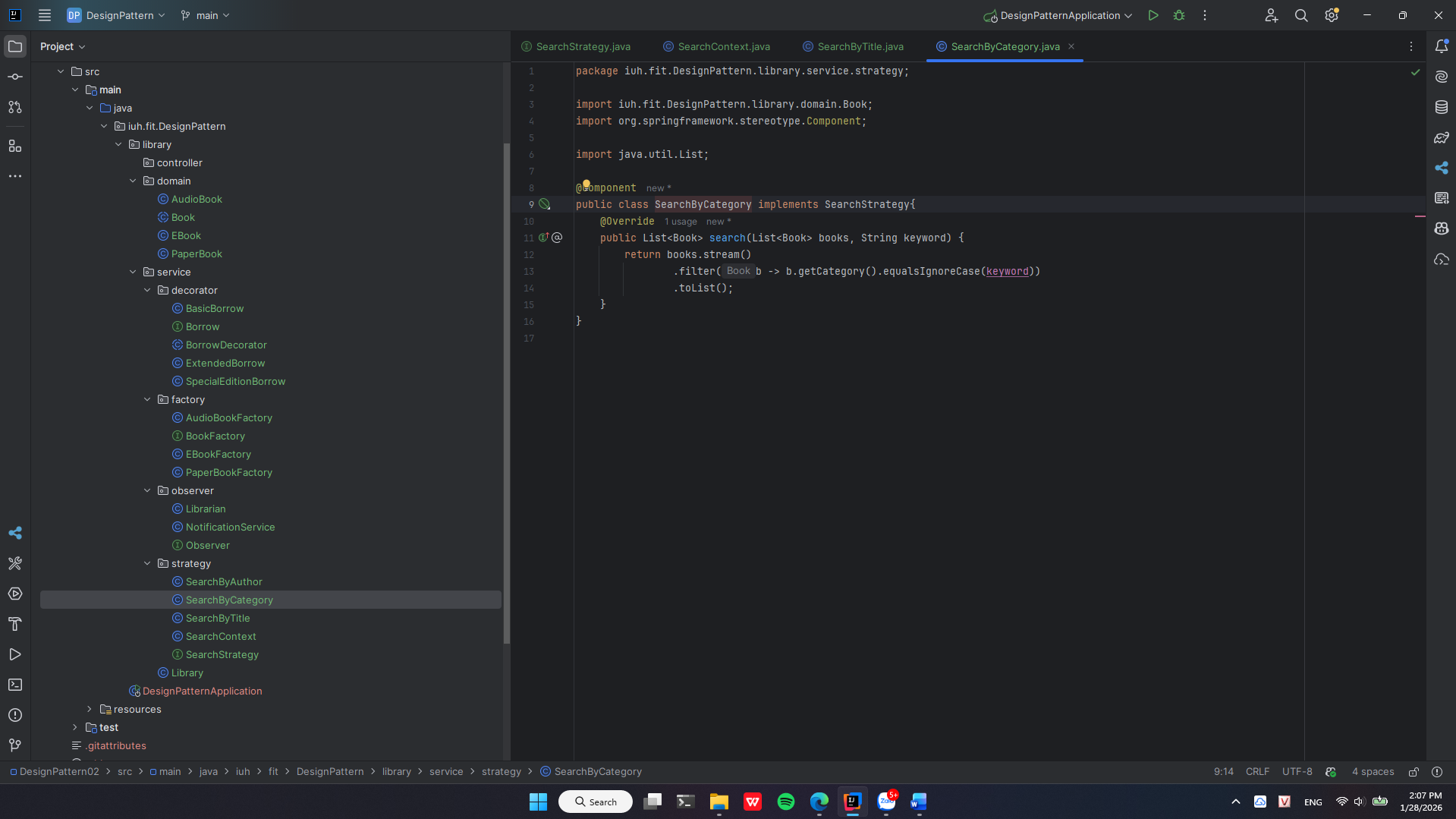


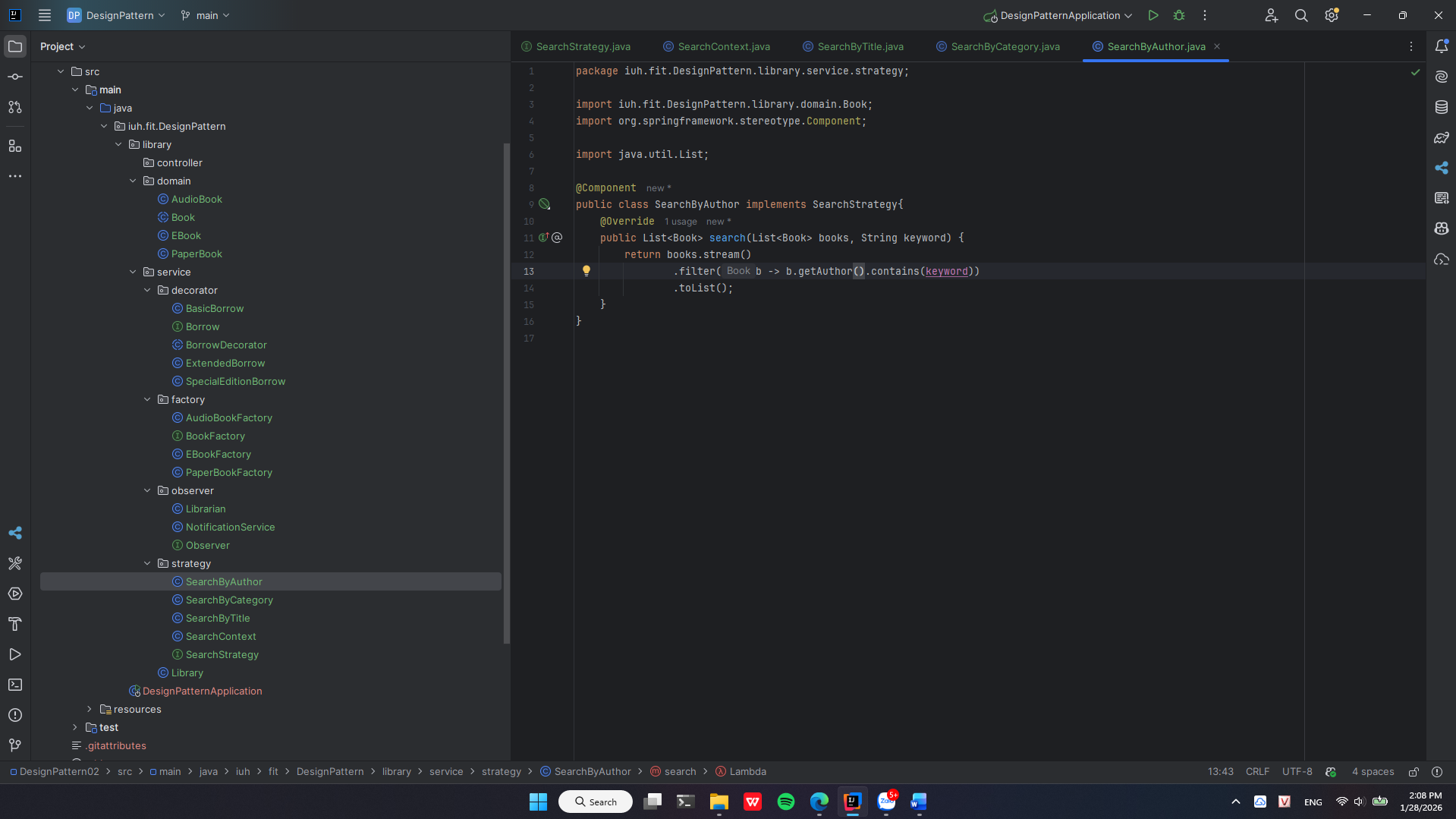


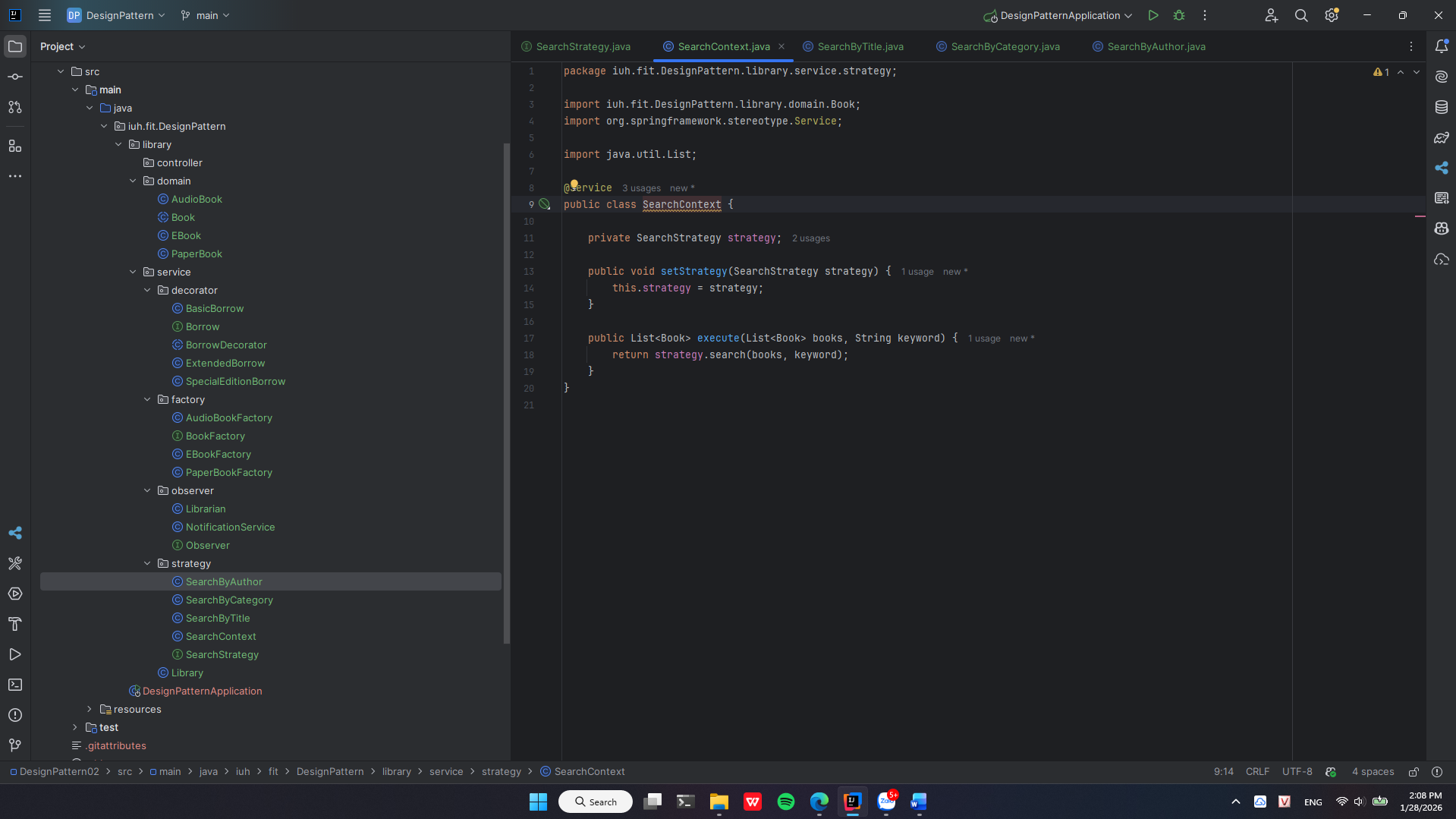
**3. Strategy Pattern:** Xây dựng các chiến lược tìm kiếm sách khác nhau (tìm kiếm theo tên, theo tác giả, theo thể loại). Dựa vào lựa chọn của người dùng, chiến lược tìm kiếm sẽ được thay đổi.



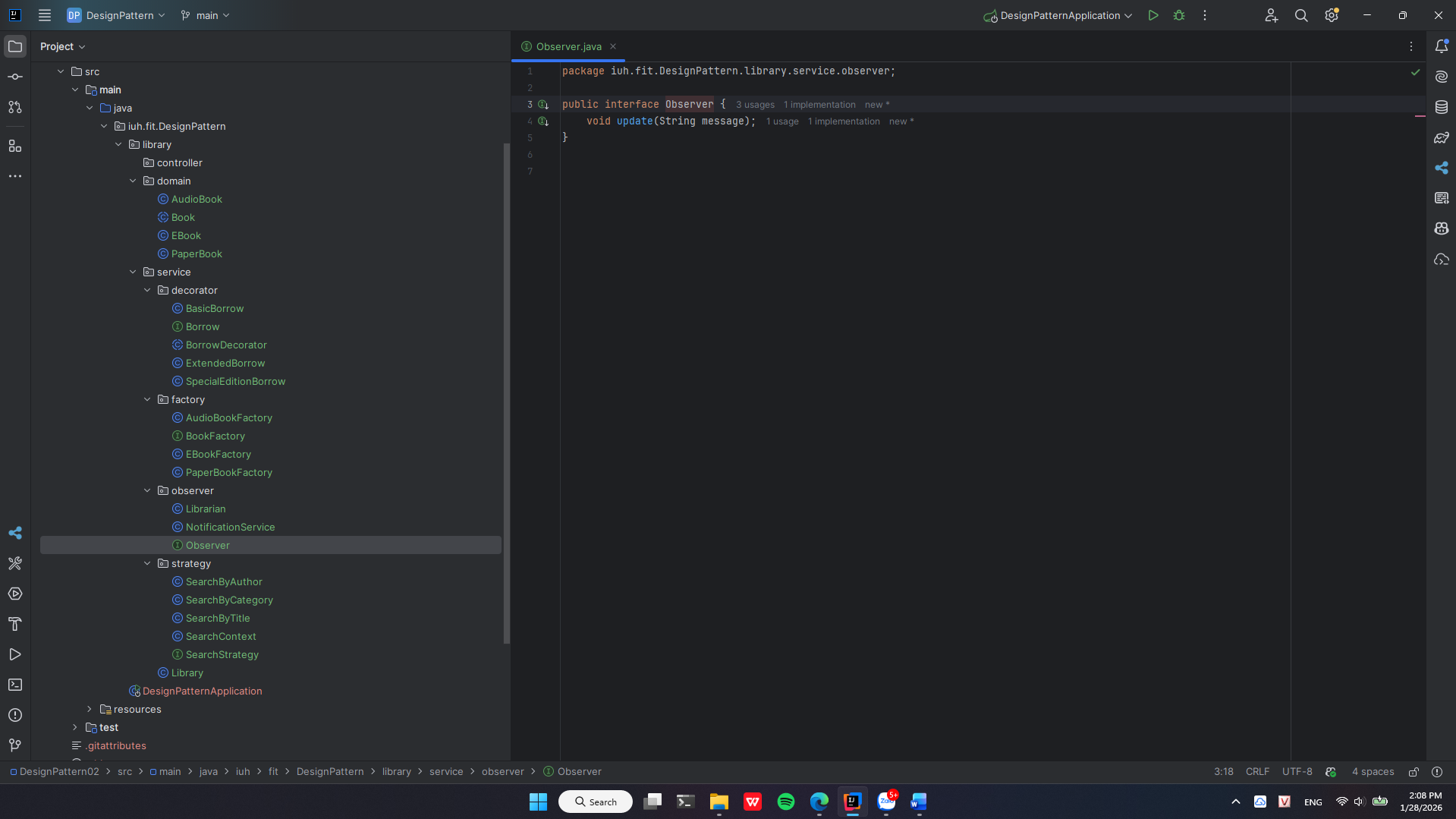


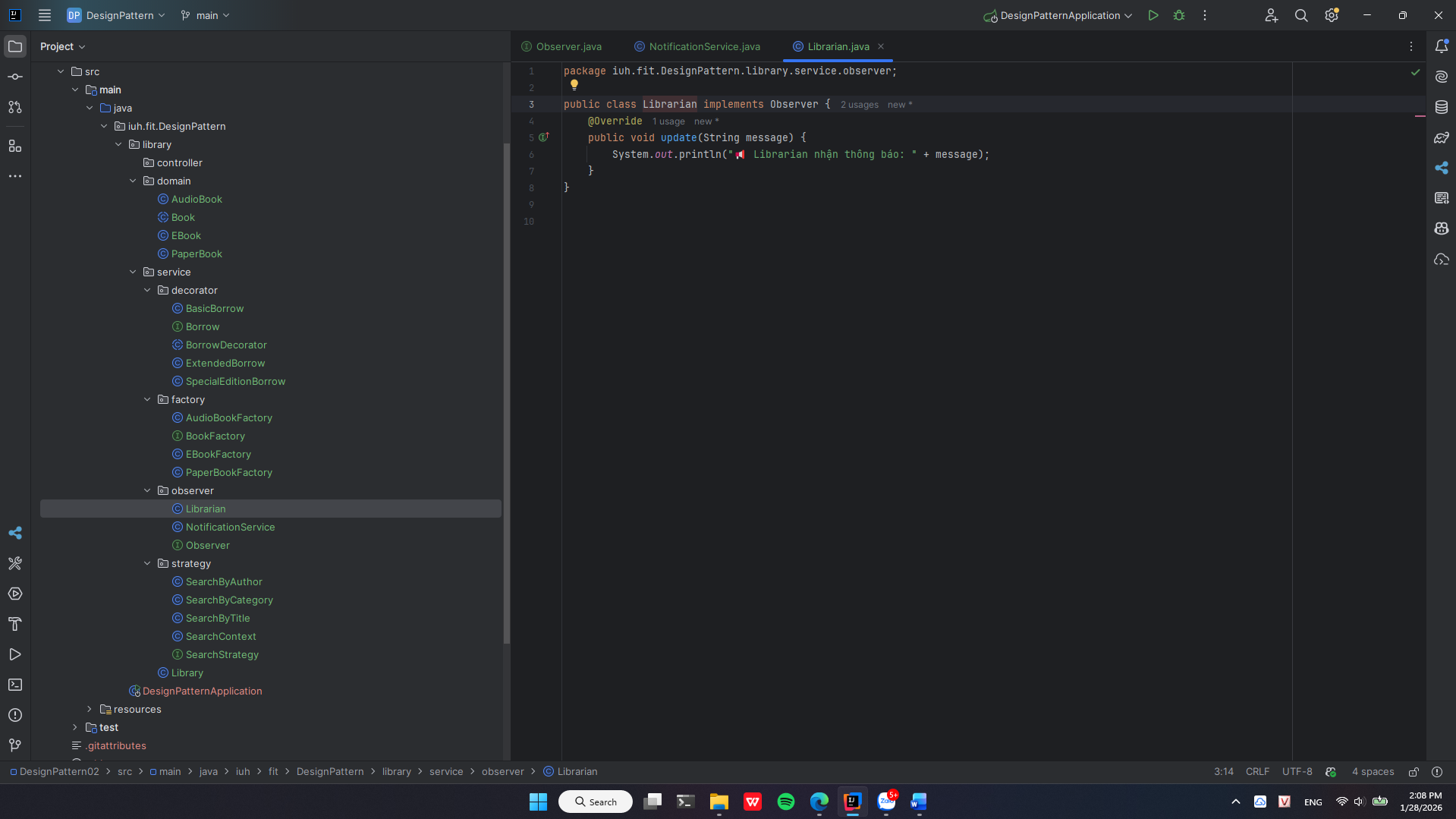


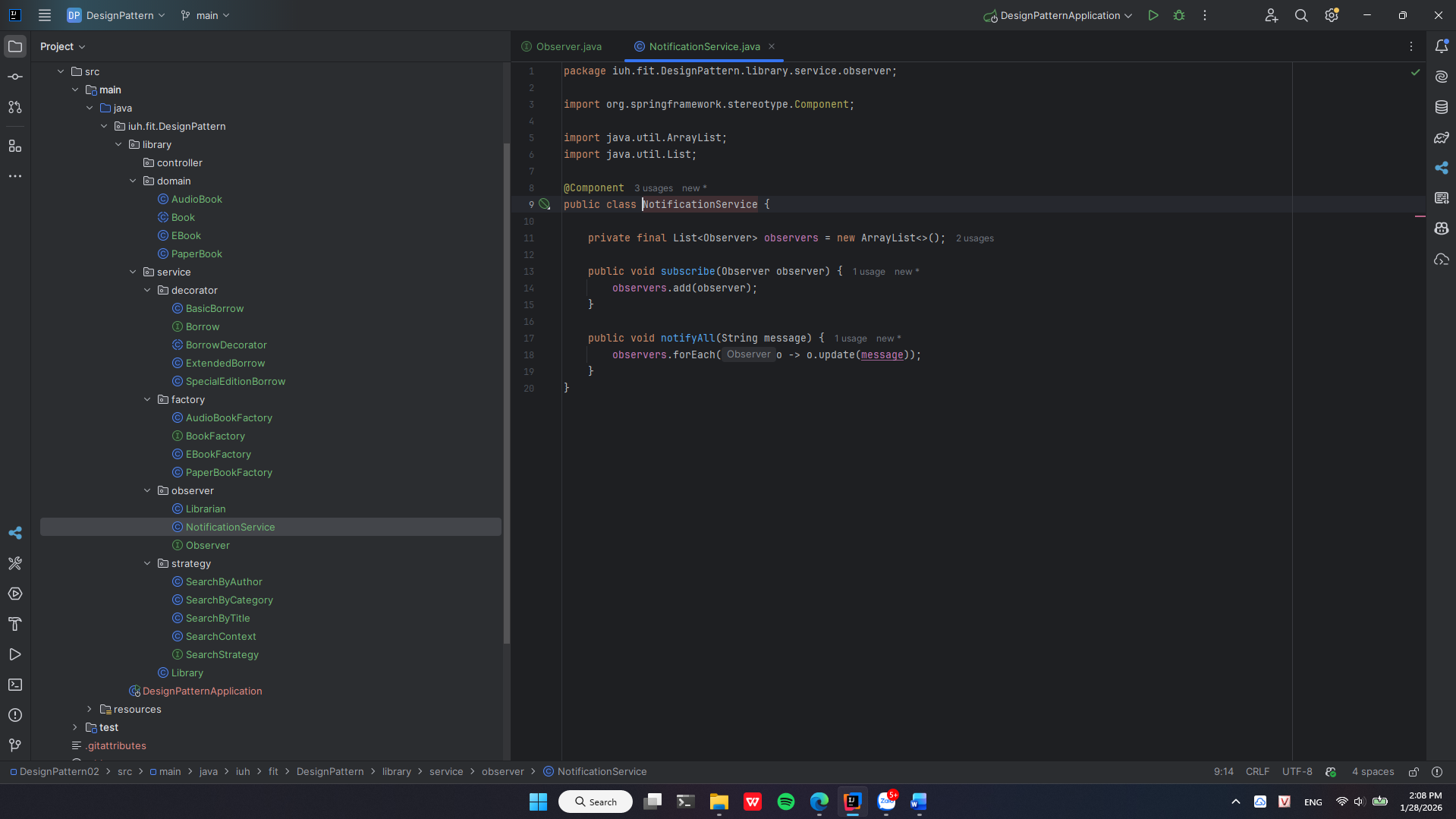




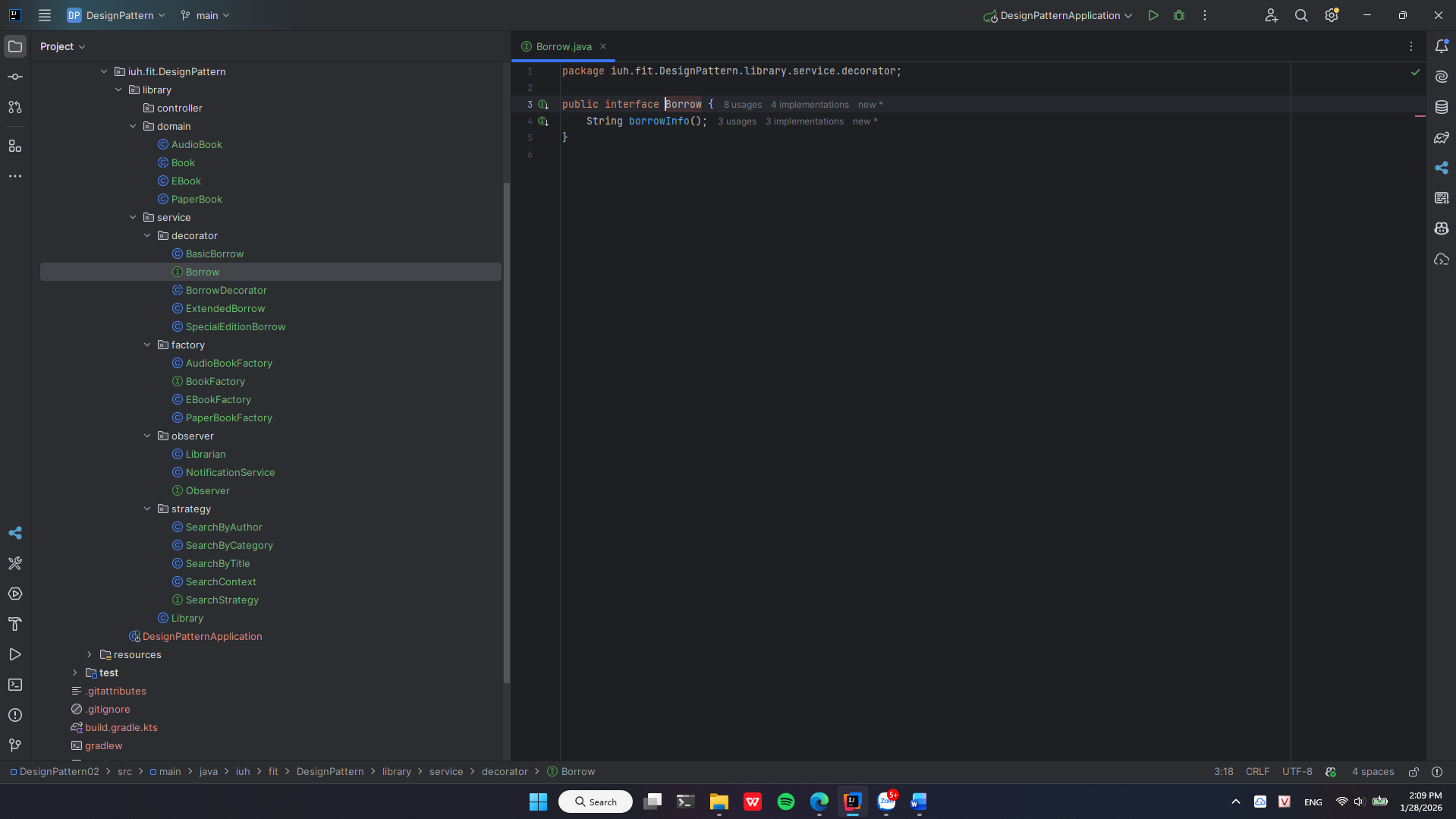
**4. Observer Pattern:** Khi có sách mới hoặc sách đã hết hạn mượn, hệ thống sẽ gửi thông báo cho những người quan tâm (ví dụ: các nhân viên thư viện hoặc những người dùng đã đăng ký theo dõi). Hãy sử dụng Observer Pattern để xử lý việc này.

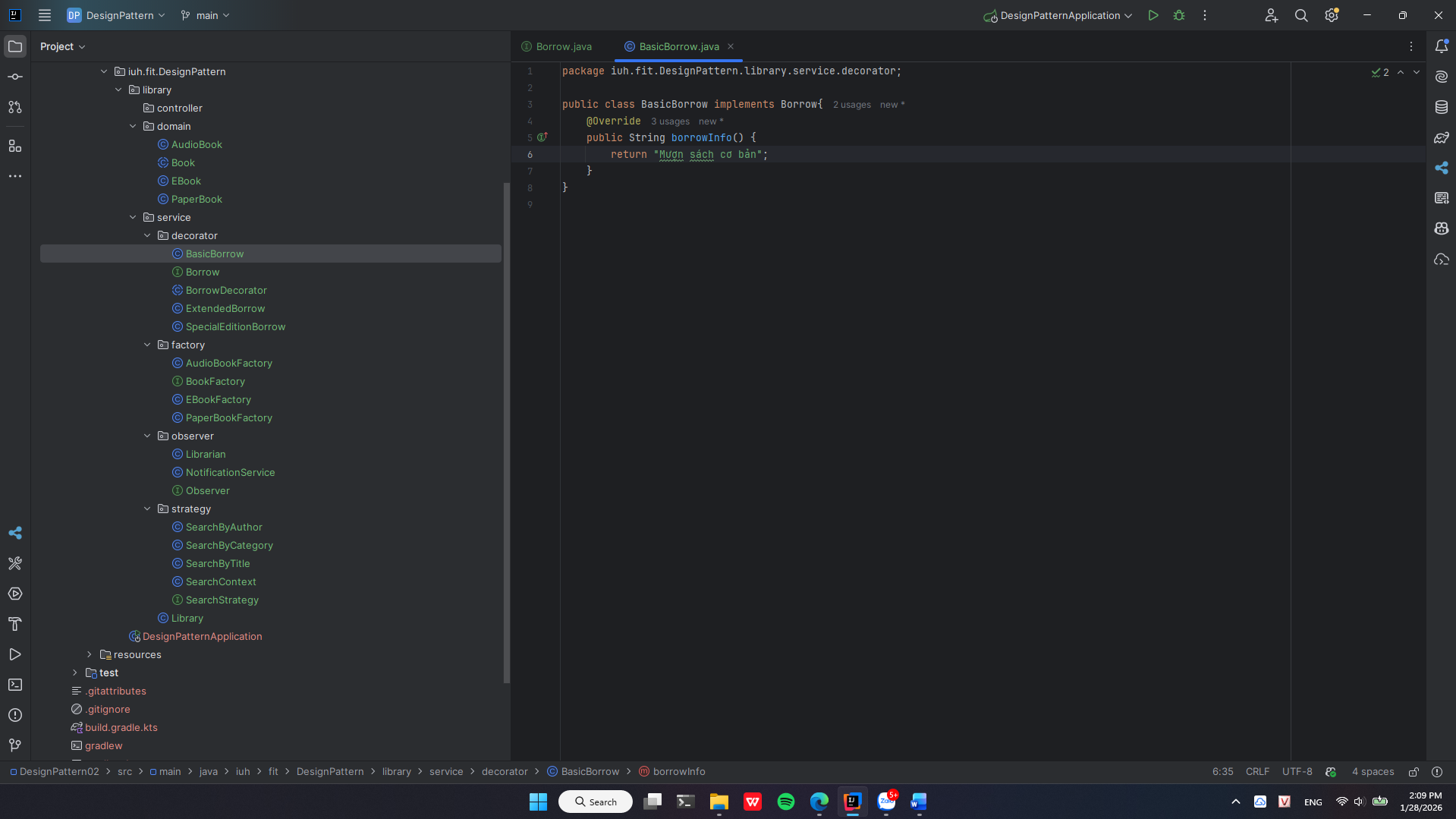


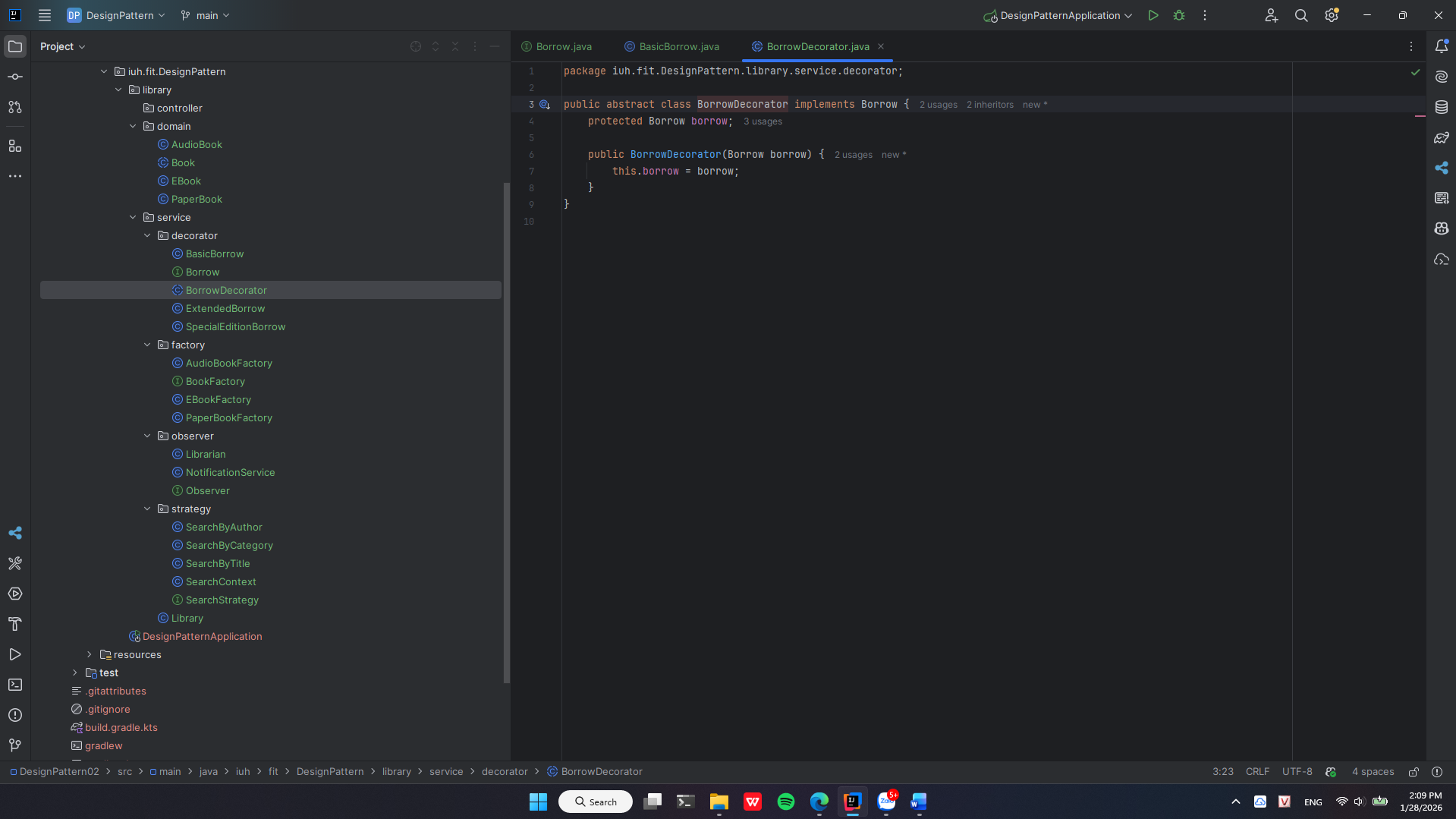


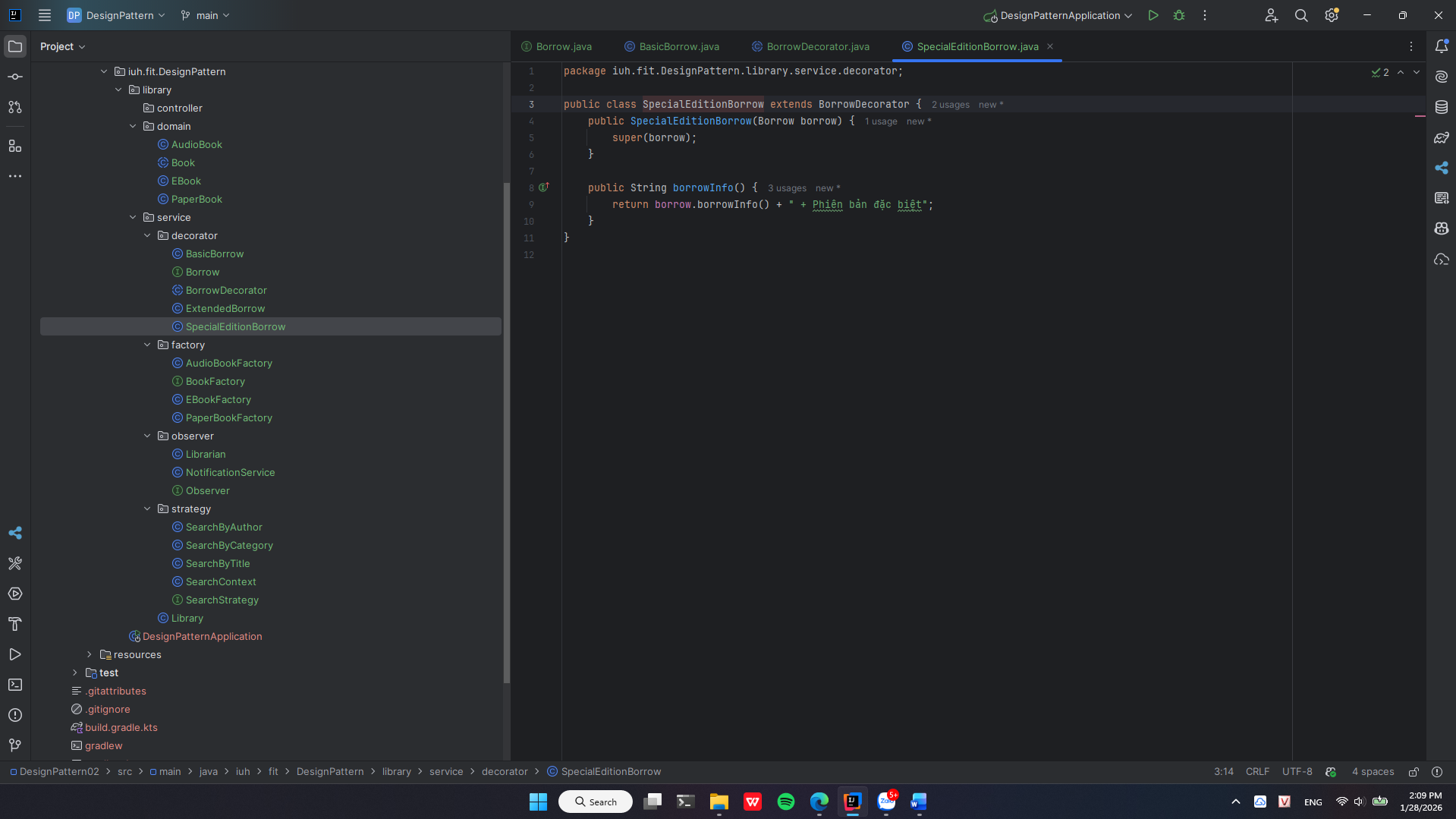


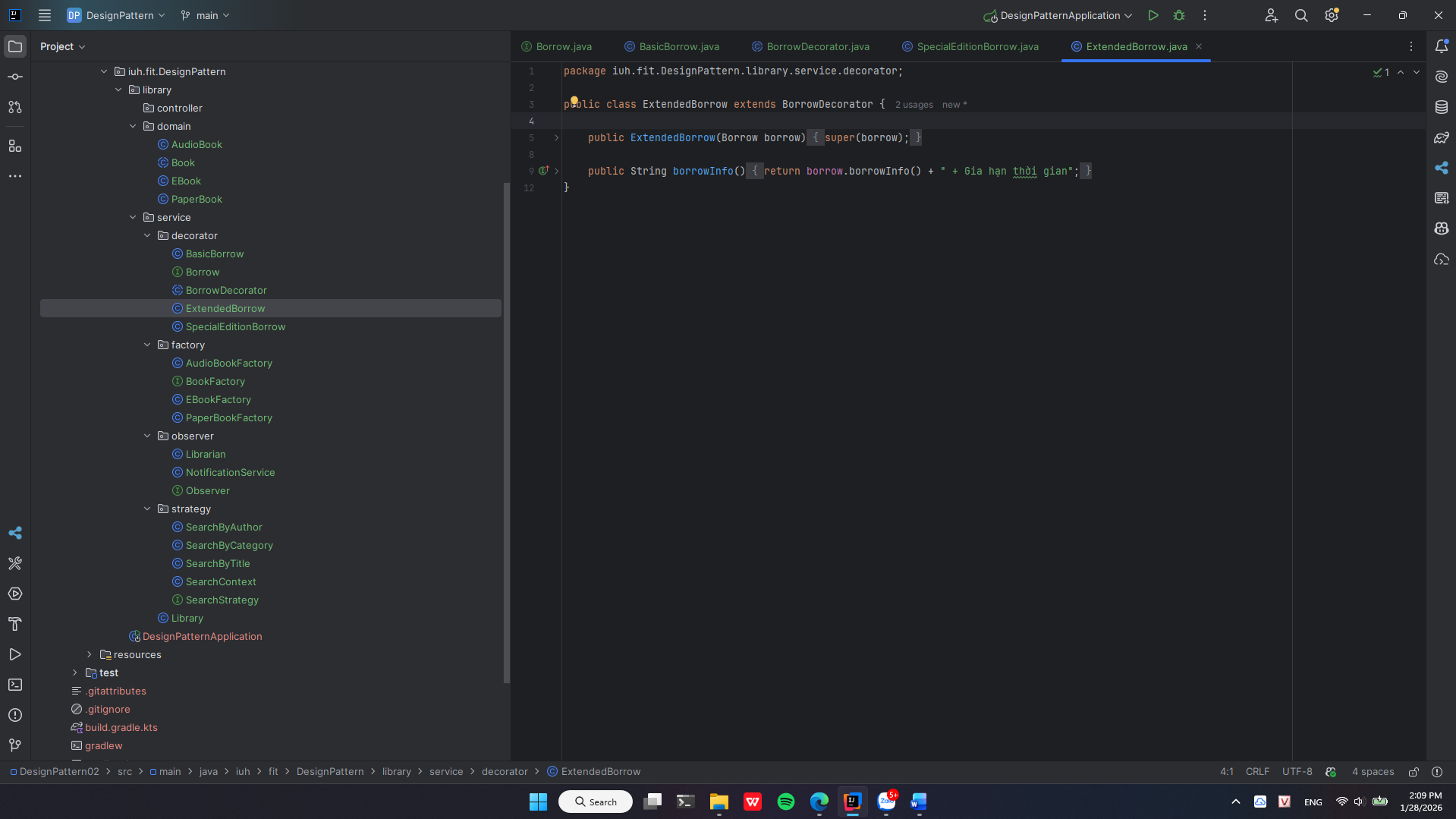
**5. Decorator Pattern:** Cho phép người dùng mượn sách với các tính năng bổ sung như gia hạn thời gian mượn, hay yêu cầu sách với phiên bản đặc biệt (sách có chữ nổi, sách có bản dịch, v.v.). Sử dụng Decorator Pattern để mở rộng các tính năng của việc mượn sách mà không thay đổi các lớp sách cơ bản.











**Kết quả chạy:**

