PJT명	JS를 활용한 도서 보관함 기능 구현	
단계	[JavaScript 개발]	
진행일자	2025.11.21	
예상 구현 시간	필수기능	5H
	추가기능	2H
	심화기능	1H

1. 목표

- JavaScript를 이용하여 HTML의 내용을 조작할 수 있다.
- 사용자 동작에 따라 HTML의 내용을 변경할 수 있다.

2. 준비사항

1) 제공 파일

- 요구사항을 위한 UI 일부가 구현된 HTML 파일
  - index.html
- 데이터 제공을 위한 JS 파일
  - authors.js, categories.js, books.js
- 요구사항 구현을 위한 JS 파일
  - script.js
- 기타 제공 이미지 파일
  - stack-of-books.png

2) 개발언어 및 툴

- JavaScript
- Visual Studio Code
- Chrome Browser

3) 필수 라이브러리 / 오픈소스

- Bootstrap 5.3

3. 작업 순서

1) 팀원과 같이 요구사항을 확인하고, GitLab에 프로젝트를 생성한다.
   - 프로젝트 이름은 08-pjt로 지정한다.
   - 각 반 담당 강사님을 Maintainer로 설정한다.
2) 팀원과 합의하여 협업 방식을 결정하고, 요구사항을 구현한다.
3) 작성한 코드들을 정리하고, README를 작성한다.
   - .gitignore 파일을 활용하여 불필요한 파일 및 폴더는 제출하지 않는다.
4) README 작성이 완료되면 심화 학습을 진행한다.
5) 제출 기한에 맞춰 모든 산출물이 GitLab에 업로드 될 수 있도록 한다.

4. 요구사항

AI 기반 도서 분석과 창작 지원 통합 솔루션 서비스를 구축하려고 한다. 다양한 장르와 형태의 도서 데이터를 수집 및 관리하고, 이를 기반으로 AI 기반 도서 분석 기능을 설계 및 구현한다. 또한 작가들에게 아이디어 발상 지원, 플롯 구성 제안, 문장 교정 및 스타일 개선 등 창작 전반에 걸친 다양한 AI 기반 창작 지원 기능을 제공한다. 또한 도서에 대한 사용자 리뷰 및 감상평 공유 커뮤니티 기능을 제공하여, 사용자들이 활발하게 소통하고 정보를 교환할 수 있는 기능을 제공한다. 사용자는 자신이 읽은 도서를 평가하고, 다른 사용자의 리뷰를 참고하여 다음 도서를 선택하는데 도움을 받을 수 있다. 나아가, 관심 도서 목록을 맞춤형으로 구성하는 등 다양한 편의 기능을 제공한다. 팀원과 상의하여 아래 요구사항을 만족할 수 있도록 요구 사항 명세서를 작성 및 구현해보자.

사용자 경험 증대를 목표로 화면 개발을 진행하고자 한다. 주어진 데이터에서 특정 키워드를 바탕으로 도서를 검색하며, 데이터를 추가하고, 자주 보는 도서를 즐겨찾기로 추가할 수 있는 기능을 제공하기 위해 JavaScript를 도입해보자.

- 요구사항 예시(참고용)
  - 아래의 내용을 참고하여 추가적인 아이디어에 대해 요구사항을 추가 또는 수정하여 기능을 구현한다. 단, **필수 기능은 구현해야 하며, 수정할 수 없다**.

번호	분류	요구사항명	요구사항 상세	우선순위
기능적 요구사항				
F01	JavaScript	도서 목록 출력	주어진 도서 데이터에 따라 화면이 다르게 출력되도록 구현	필수
F02	JavaScript	검색 기능	사용자가 입력한 검색어를 바탕으로 특정 데이터만 출력되도록 구현	필수
F03	JavaScript	도서 데이터 입력 탭	데이터를 입력할 수 있는 UI로 전환하는 탭 기능을 구현	필수
F04	JavaScript	도서 데이터 추가	입력된 도서 데이터를 바탕으로 데이터 목록 갱신	필수
F05	JavaScript	즐겨찾기 기능	특정 도서를 쉽게 찾아볼 수 있도록 지정하고, 지정된 도서를 확인할 수 있는 탭으로 전환하는 기능 구현	필수
F06	AI 활용	Modal 구현	AI를 활용하여 도서의 상세정보를 확인할 수 있는 모달 창 구현	도전
F07	AI 활용	페이지 기능	AI를 활용하여 도서 목록에 페이지 기능 구현	도전
…	…	…	…	…
비기능적 요구사항				
NF01	프로젝트 관리	Git 활용	개발자 간 작업 충돌이 일어나지 않게끔 Git을 활용하여 프로젝트 관리	필수
…	…	…	…	…

1) 프로젝트 관리 (필수)

Git의 기능을 적극적으로 활용하여, 여러 개발자가 동시에 개발할 수 있는 환경을 조성한다.

- 요구사항 번호: NF01
- 팀원과 합의하여 branch 생성 원칙 정리하기
  - Gitflow, GitHub Flow 등을 참고하여 원칙을 정할 수 있음
  - branch 운영 중 문제상황이 발생할 경우 원인에 대한 분석을 꼭 진행할 것
- 팀원과 합의하여 commit 내역 기록 원칙 정리하기
  - 기능 개발, 버그 수정 등 그 목적이 명확히 드러날 수 있도록 commit 메시지를 작성할 것
  - 어느 시점에 commit을 남길지에 대하여 충분한 합의를 이룰 것
- 프로젝트 종료 시 초기에 세운 원칙이 잘 지켜졌는지 점검
  - README에 초기에 세운 원칙을 설명하여 정리할 것
  - 생성했던 branch 들에 대한 용도를 정리할 것
  - 작성된 commit들을 GitLab에서 확인하여 스크린샷으로 README에 포함시킬 것

2) JavaScript (필수)

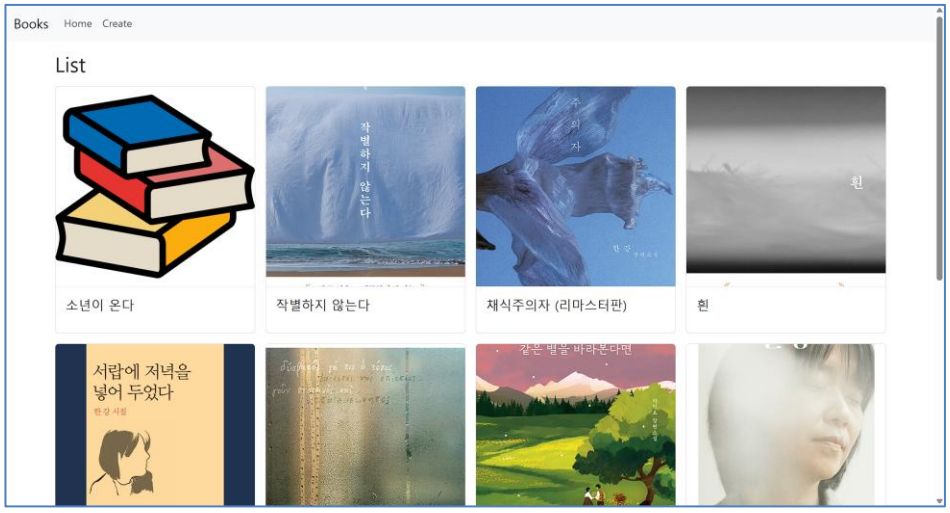
주어진 도서, 작가, 분류 데이터를 바탕으로 사용자의 행동에 따라 동작하는 UI를 구현해보자.

- 요구사항에 따라 index.html에 이미 코드가 작성된 경우가 존재한다.
- 원한다면 CSS, Bootstrap 등을 활용하여 추가 스타일링이 가능하다.
  - 출력 예시에서 사용된 UI 컴포넌트가 아니더라도 사용할 수 있다.
- 주어진 books.js, authors.js, categories.js 또는 script.js에 미리 구현된 코드를 수정할 경우 정상적으로 동작하지 않을 수 있다.
  - 기본적으로 모든 요구사항은 index.html, script.js만 수정해도 구현할 수 있지만, 원한다면 JS파일을 추가할 수 있다.
- 요구사항 구현을 위한 코드가 전부 독립적일 필요는 없다.
  - 새로운 요구사항 구현을 위해, 이전에 작성한 코드를 수정하여도 무방하다.
- HTML 요소를 추가하기 위한 목적의 **Element.innerHTML의 사용을 금지**한다.
  - 특정 HTML 요소의 content를 제거하는 용도로는 사용 가능하다.

A. 도서 목록 출력

주어진 도서 데이터를 화면에 출력하고자 한다. 각 도서의 정보를 그리드 형태로 출력해보자.

- 요구사항 번호: F01
- 'div#book-list-row'의 내부에 구현
  - script.js에 정의된 books의 데이터 활용
- Bookstrap의 Card Component로 구현
  - 커버 이미지, 제목, 내용 출력
  - 커버 이미지가 없거나 빈 문자열인 경우 주어진 stack-of-books.png 출력
- 화면 사이즈에 따라 한줄에 배치되는 도서 목록 조정
  - Bootstrap Grid System 활용
  - 'col-12', 'col-sm-6', 'col-md-4', 'col-lg-3'
- 출력 예시

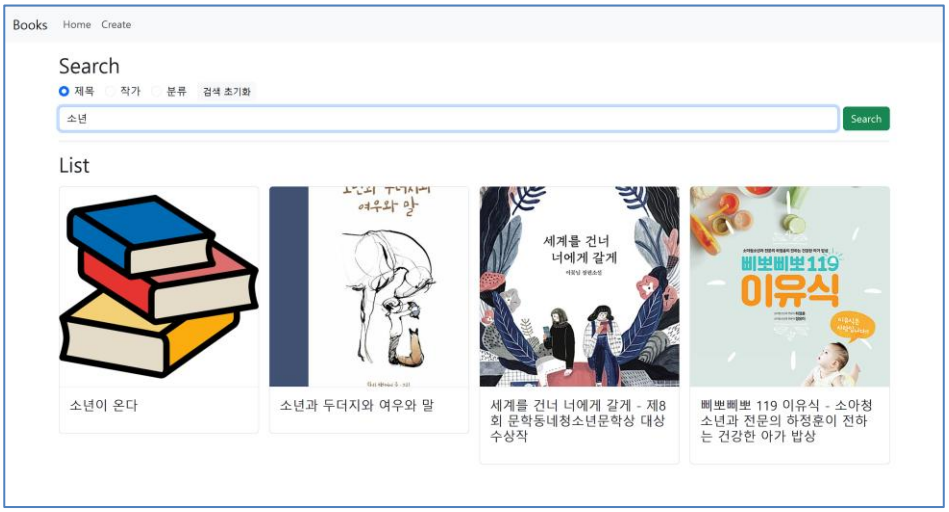


B. 검색 기능

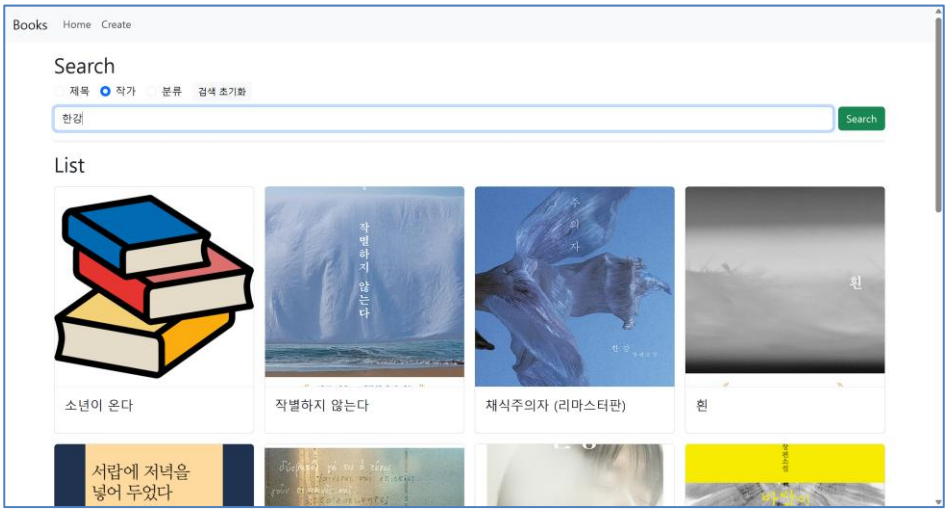
도서 목록을 사용자의 검색어에 따라 필터링 하고 싶다. 검색창을 구현하고, 검색창에 입력한 검색어를 포함한 도서 정보만 필터링하여 보여주는 기능을 구현해보자.

- 요구사항 번호: F02
- 검색 양식의 경우 HTML에 직접 구현
  - 버튼 클릭이나 엔터 키 등을 통해 동작
  - 데이터를 추가하는 기능은 JS로 구현
  - '필터링'의 기준은 "입력한 검색어가 선택한 기준에 포함되었는지"만 기준으로 삼아도 무방
  - '제목', '작가', '분류'를 기준으로 검색 가능
- 검색어를 입력하지 않았을 경우 alert 창으로 '검색어를 입력하세요.' 출력
- 도서의 작가, 분류 데이터의 경우 개별 도서의 'authorId', 'categoryId' 속성으로 저장되어 있음.
  - 해당 속성의 값은 각각 'authors', 'categories' 배열의 동일한 값의 'id'를 가진 장르를 상징함
- 검색된 도서 목록은 요구사항 F01과 동일한 디자인으로 출력하여도 무방
  - 검색 결과가 없을 경우에 대한 처리는 자유롭게 구성
- 검색 초기화 UI를 추가
  - 초기화 버튼을 누른 경우 전체 목록을 다시 출력

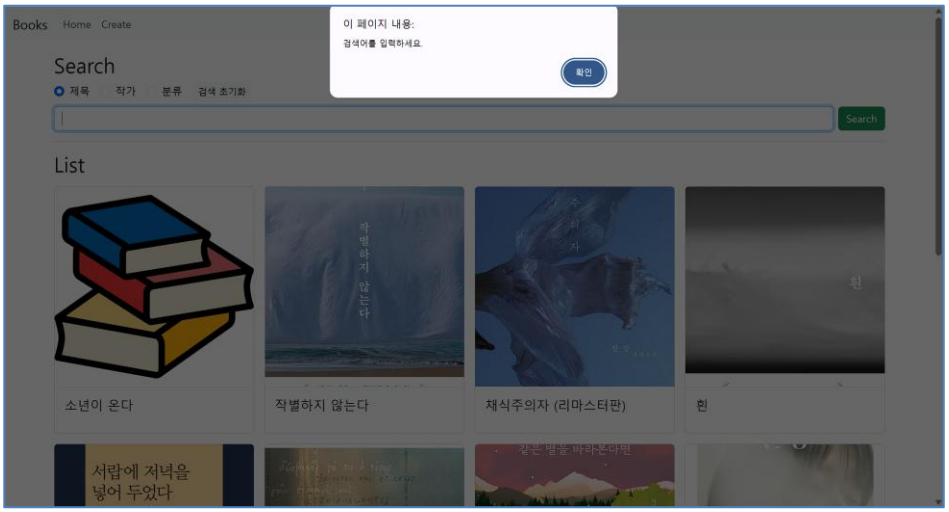
- 출력 예시
  - '제목'으로 '소년' 검색



  - '작가'로 '한강' 검색



  - 검색어가 없을 경우

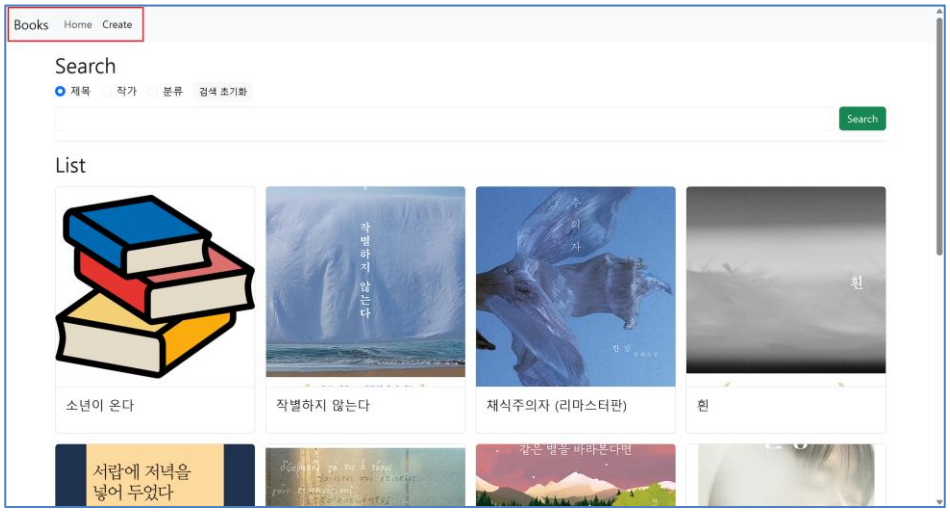


C. 도서 데이터 입력 탭

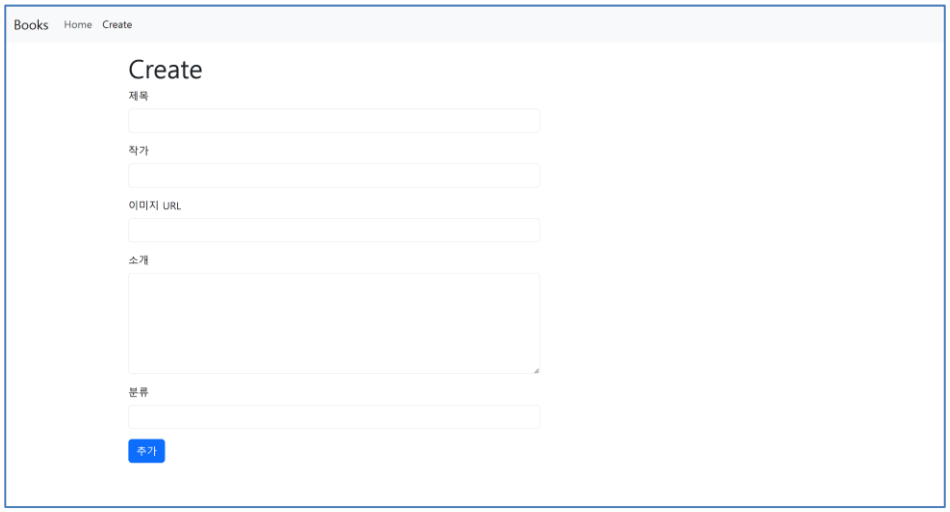
도서 추가 화면으로 전환하는 기능을 구현해보자.

- 요구사항 번호: F03
- 도서 추가 양식의 경우 'form#create-form' 요소에 구현되어 있음
  - 조상 요소 중 'div#create-container'의 class로 포함된 d-none 클래스로 인해 출력되지 않고 있는 상태
- 미리 구현된 Navbar 요소를 이용해 화면 전환
  - 자손 요소 중 '.nav-link' 요소를 클릭 시 화면의 내용이 변경되도록 구현
  - Home을 클릭한 경우 다시 영화 목록 페이지로 이동
  - 요소를 변경하거나 id를 추가하는 등 구현 방식은 자유롭게 구성할 수 있음

- 출력 예시
  - 미리 구현된 Navbar 요소



  - Create 클릭 시



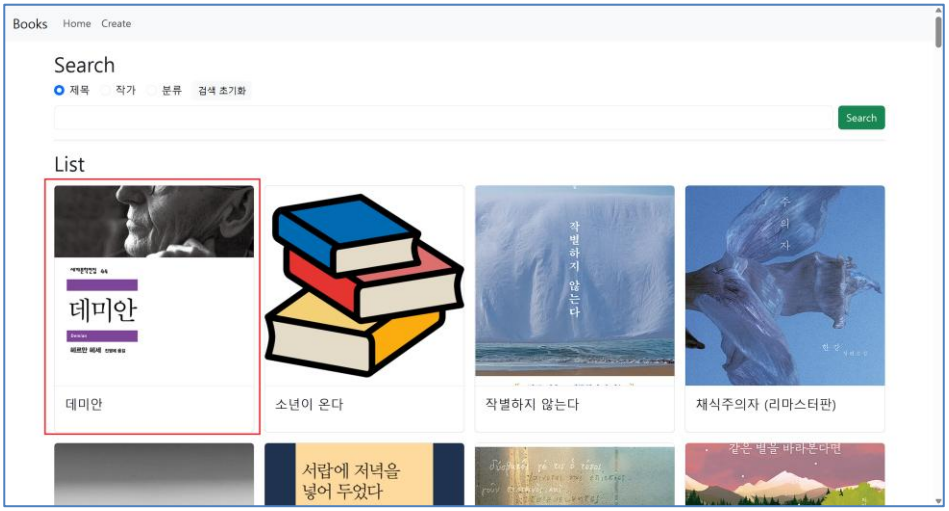
D. 도서 데이터 추가

도서 추가를 위해 구현한 UI에 입력한 데이터를, 도서 목록에 추가해보자.

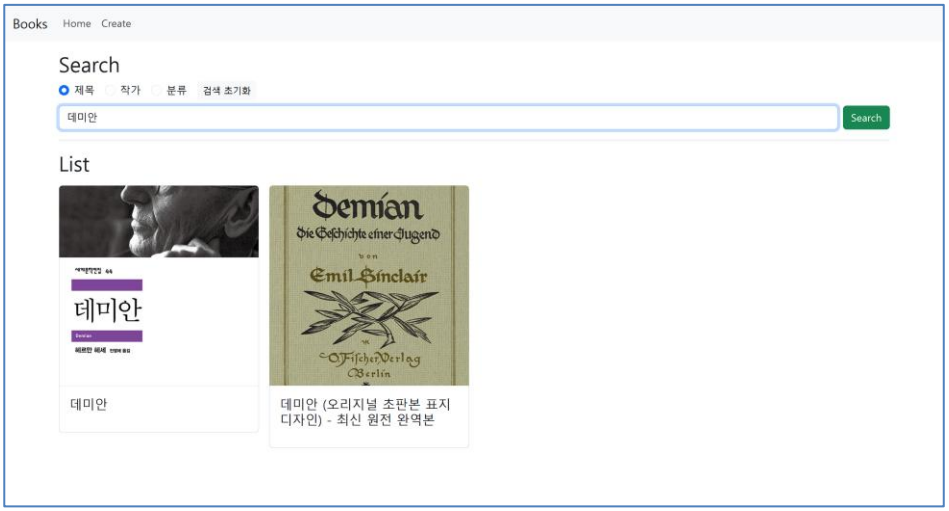
- 요구사항 번호: F04
- 제공된 양식의 기능 구현
  - 버튼 클릭이나 엔터 키 등을 통해 동작
  - 제목, 작가, 소개, 분류는 입력 필수
  - 필수 내역이 입력되지 않았을 경우 'ul#create-book-errors'에 적절한 에러 메시지 표시
- 이미지의 경우 URL 만 입력되어야 함
  - URL 판단의 기준은 http로 시작하는지만 검증
  - URL이 아닌 경우 빈 문자열로 대체
- 입력된 작가, 분류 데이터가 이미 저장된 데이터가 아닌 경우, 각각 authors, categories 배열에 추가
- 전체 도서 목록 제일 앞에 추가
  - 도서 추가 후 화면을 도서 목록 탭으로 변경
- 추가된 도서는 브라우저 **새로고침 시 사라짐**에 유의

- 출력 예시
  - 데이터 입력

  - 추가 결과



  - 검색 결과

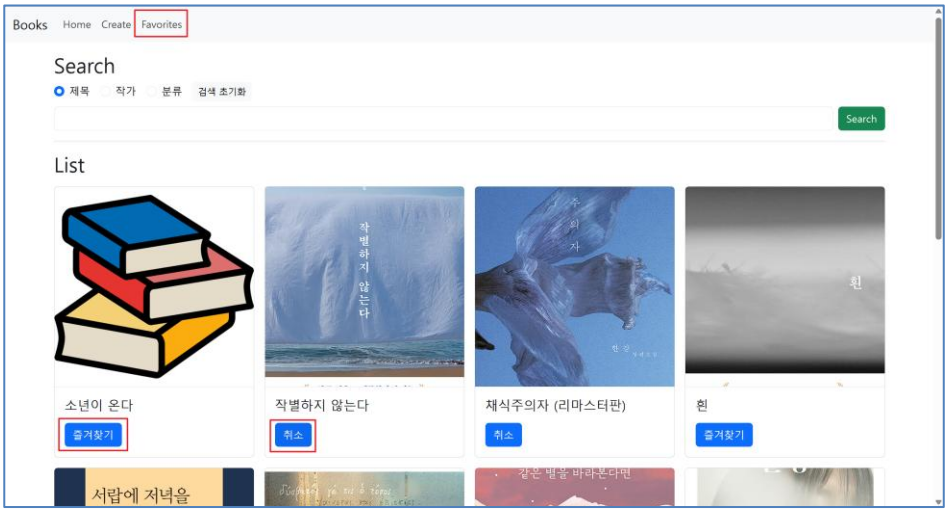


E. 즐겨찾기 기능

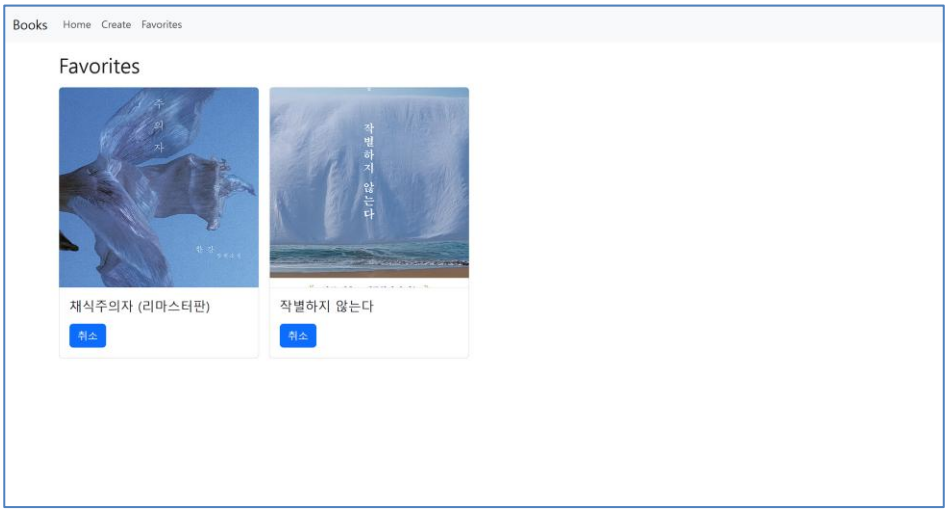
도서 목록에서 원하는 도서를 즐겨찾기 하는 기능을 만들어보자.

- 요구사항 번호: F05
- 탭 목록에 'Favorites' 추가
  - HTML에 직접 구현 가능
  - 실제 '즐겨찾기' 탭의 UI의 경우, 'Home', 'Create' 탭의 구현 참고
- 도서 Card Component에 버튼 형태의 UI로 구현
  - 사용자가 특정 도서가 '즐겨찾기'에 추가되었는지, 추가되지 않았는지 구분하기 쉽도록 구현
  - '즐겨찾기'에 추가된 도서의 경우 취소할 수 있도록 구현
- '즐겨찾기'에 추가된 도서가 없을 경우 적절한 메시지 노출
  - '즐겨찾기' 탭으로 전환된 상태에서, 모든 도서가 '즐겨찾기'에서 제거될 경우 'Home' 탭으로 이동

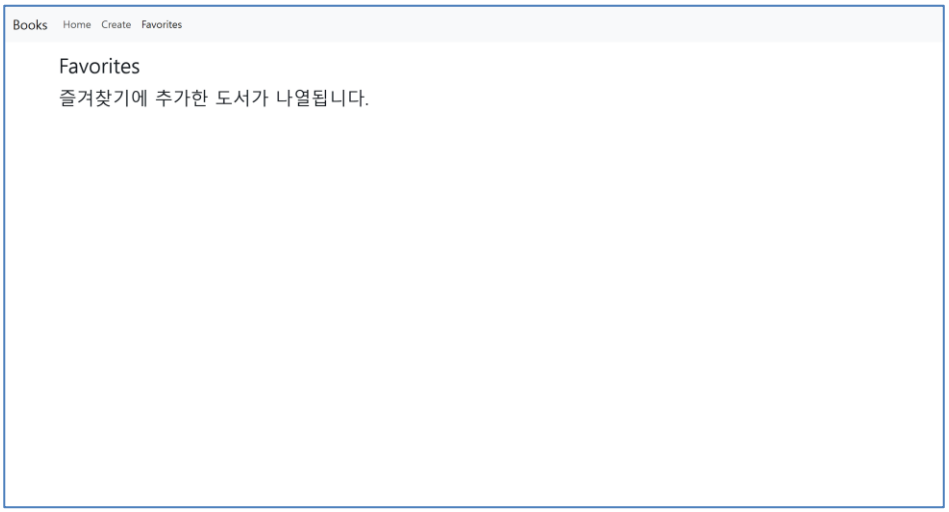
- 출력 예시
  - 즐겨찾기 탭 및 추가 버튼



  - 즐겨찾기 탭 전환



  - 즐겨찾기 도서가 없는 경우



3) 도전 과제

생성형 AI 도구를 활용하여 도전과제 요구사항을 해결해보자.

- 코드 생성, 아이디어 구상, 문제 해결 방법 탐색 등 다양한 방식으로 활용할 수 있다.
- 사용할 생성형 AI 서비스는 자유롭게 선택한다.
  - 제공되는 GPT API Key를 활용할 경우, 기능별로 정해진 모델을 활용한다.
- 최종 결과물은 AI 생성 내용을 바탕으로 직접 수정 및 개선하여 적용해본다.

보조 수단으로 활용하되, 능동적인 자세로 학습에 임할 것.

- 최종적인 이해와 적용은 자기 주도적 학습을 통해 이루어지며, 배운 내용을 스스로 기록하고 정리하며 학습 효과를 높일 것.

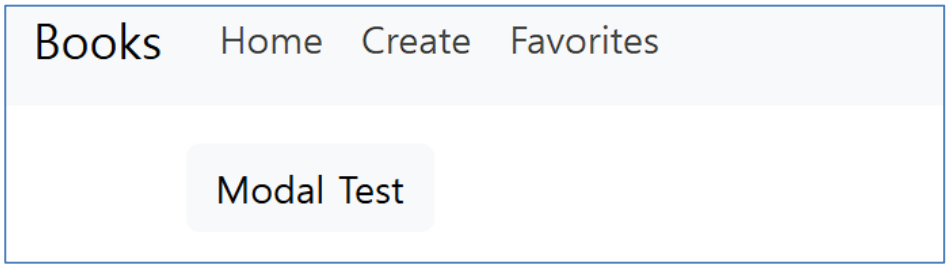
A. 상세보기 모달

선택된 도서의 상세 정보를 확인할 수 있는 모달 창을 구현해보자.

- 요구사항 번호: F06
- 모달은 Bootstrap으로 구현되어 있음
- 주어진 index.html에 작성된 모달 및 모달 열기 버튼을 통해 모달의 모습과 동작 방식을 학인
  - 모달 열기 버튼의 경우 주석처리 되어 있음
- 도서 Card Component의 버튼 등의 UI를 클릭할 시 모달 노출
  - 이 때 클릭한 도서의 제목, 작가, 커버 이미지, 줄거리가 모달에 채워져야 함
- 모달의 Close 버튼이나 모달 바깥을 클릭할 시 모달이 제거
- 그 외의 스타일 요소는 자유롭게 구성

- 데모 모달 요소
  - 모달 버튼

```
<!-- Home 탭 -->
<div id="home-container" class="container mb-3">
  <!-- 모달 버튼 예시 -->
  <!-- <div class="my-2">
    <button class="class btn btn-light" data-bs-toggle="modal"
    data-bs-target="#book-details-modal">Modal Test</button>
  </div> -->
```



  - 데모 모달 출력

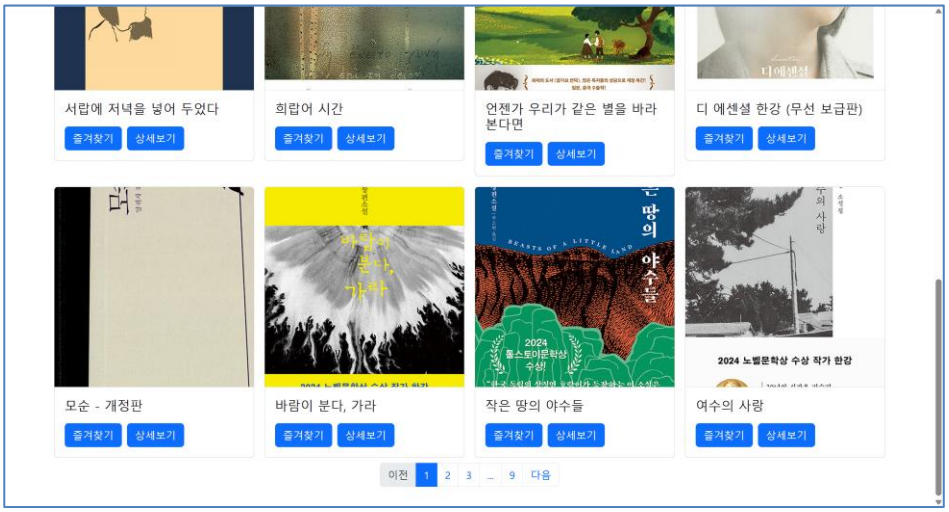
- 출력 예시

B. 페이지 기능

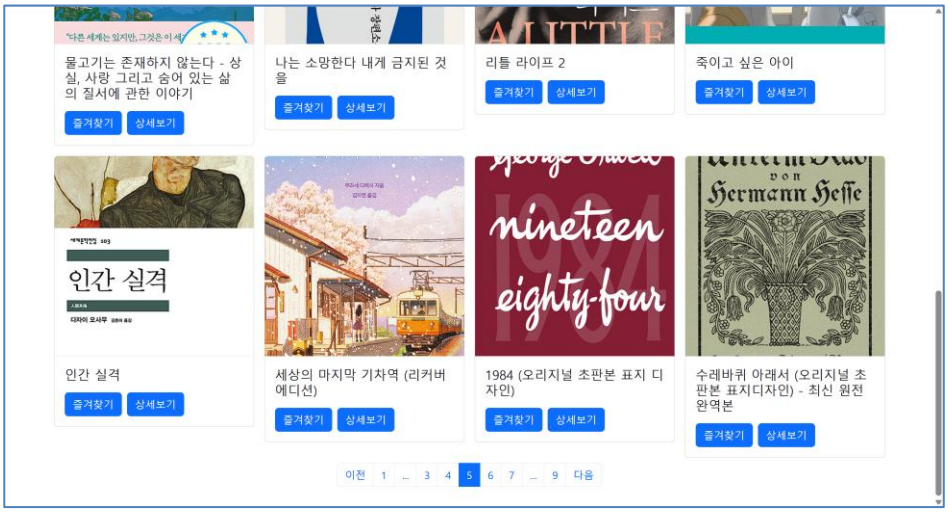
도서 목록에 페이지 기능을 추가해보자.

- 요구사항 번호: F07
- 출력할 도서 데이터가 12권 이상인 경우, 전체 도서 목록에 대해서만 동작
  - 한 페이지에 12권씩 만 출력되도록 구현
- 페이지 링크를 구현하기 위한 UI는 자유롭게 구성
  - HTML 직접 수정 가능
  - Bootstrap 활용 가능
- 페이지 이동을 위해 새로고침이 이뤄져도 무방
  - 이로 인해 **추가된 도서가 삭제되어도 상관없음**

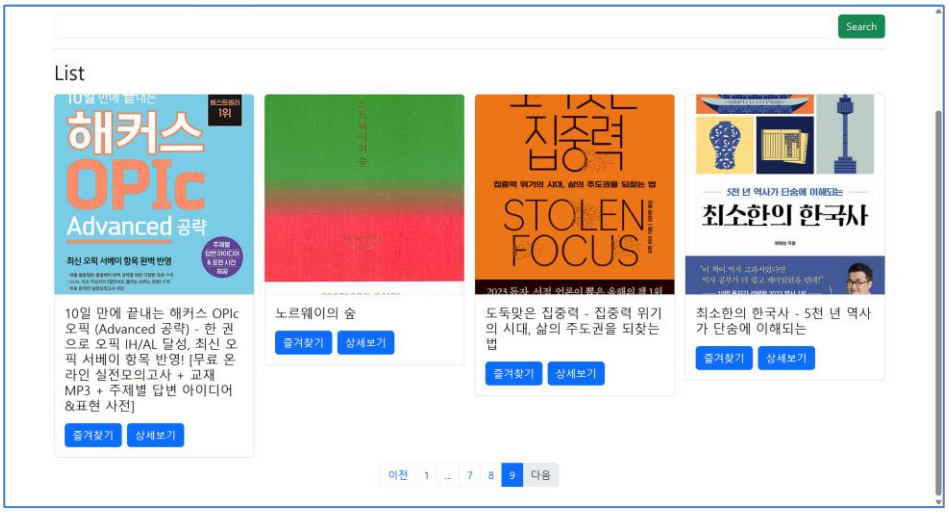
- 출력 예시
  - 첫 페이지



  - 중간 페이지



  - 마지막 페이지



5. 참고자료

- MDN

  https://developer.mozilla.org/en-US/
- Bootstrap

  https://getbootstrap.com/

6. 결과

제출 기한은 진행일 18시까지이므로 제출 기한을 지킬 수 있도록 한다. 제출은 GitLab을 통해서 이뤄진다.

- **산출물과 제출**
  - 단계별로 구현 과정 중 학습한 내용, 어려웠던 부분, 새로 배운 것들 및 느낀 점을 상세히 기록한 README.md
  - 완성된 각 문제 별 소스코드 및 실행 화면 캡쳐본
  - 프로젝트 이름은 08-pjt로 지정 및 각자 제출
    - 한 명의 GitLab 계정에 Git 저장소 Push 및 작업
    - 나머지 인원은 제출 시 해당 저장소를 Fork 하여 제출
  - 각 반 담당 강사님을 Maintainer로 설정

- 끝 -