

목차

1. 프로젝트 설명
 - a. 프로젝트 폴더 계층 구조
 - b. 사용 기술 및 언어
2. **ERD**
3. 메인 페이지
4. 책
 - a. 상세정보
 - b. 검색
5. 장바구니
6. 주문

1-a. 프로젝트 폴더 계층 구조

설명 : sg폴더가 본인이 한거임.

- **src/main/java/**
 - **com/bookmarket/app**
 - **controller** : MVC 컨트롤러.
 - **dto** : MVC DTO.
 - **mapper** : MVC DAO.
 - **restcontroller** : API 용 컨트롤러.
 - **service** : MVC 모델 서비스 레이어.
 - **BookMarketApplication.java** : 프로그램 진입점.
- **src/main/resources/**
 - **mapper** : SQL 코드들.
- **src/main/webapp/**
 - **resources** : css, js, Multipart 등
 - **WEB-INF/views** : MVC View (JSP)

1-b. 사용 기술 및 언어

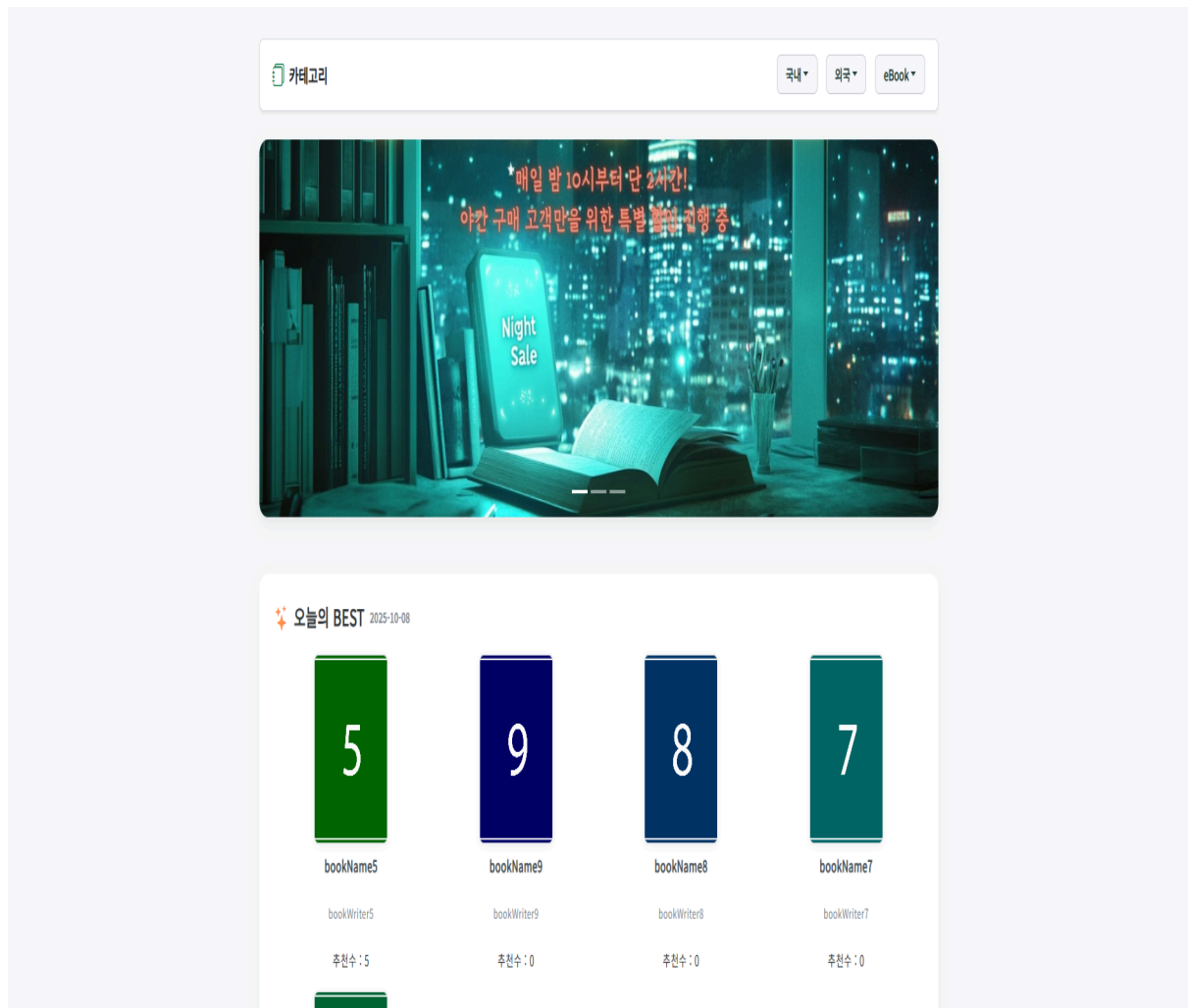
- **기술** : Spring boot, mybatis, oracle sql, java, ajax
- **언어** : SQL, JAVA, JS

2-a. ERD

ERD 주소 : <https://www.erdcloud.com/d/CptyXbQ9G42t5ZCfi>

설명 : 출판사가 책 등록을 해서 판매하고 사용자가 사는 상황.

3. 메인페이지



책 추천 설명 : 현재날짜 - 7 부터 현재날짜까지 출간된 책들중 추천순으로 표시.

4-a. 책 상세정보

1	book-desc-container	
	bookId : 1	
	추천수 : 0	
	수량 : 1	1000 원
	장바구니	추천하기

book-info-container

설명 :

- **수량 표시** : **if**문으로 작성한게 아닌 효율성 및 가독성을 위해 정규식 사용. 초기 수량을 **1**로 표시를 해 수량의 변화가 발생하고 해당 변화가 정규식에 부합하면 해당 값으로 표시하고 정규식에 부합하지 않으면 롤백처리를 통해 이전 값을 표시. 이전 값은 정상처리된 값이기 때문에 롤백처리.
- **가격 표시** : 정규식 사용. 수량 표시와 동일한 원리.
- **추천하기** : 추천테이블에서 오늘날짜에 해당 데이터가 있는지 없는지를 확인하여 구현. 있으면 실패 없으면 성공. **1일 1회** 제한.
- **장바구니** : 물건 넣기 기능. 추천하기 기능과 비슷하게 해당 데이터가 있는지 없는지에 대한 판별을 기준으로 구현. 중복 책 넣기 안됨.

4-b. 책 검색



검색어를 입력하세요



로그인

검색 결과: book



bookName9

저자: bookWriter9

9000원

상세보기



bookName8

저자: bookWriter8

8000원

상세보기



bookName7

저자: bookWriter7

7000원

상세보기



bookName6

저자: bookWriter6

6000원



bookName5

저자: bookWriter5

5000원



bookName4

저자: bookWriter4

4000원

설명 : 판매신청된 책들중에서 판매신청완료된 책들을 **WHERE %keyword%** 필터링을 통한뒤, 페이징 처리를 하여 보여줌.

5. 장바구니

3	bookName3				삭제
			단가 : 3,000원 총 합계 : 3,000원		
			1개		
2	bookName2				삭제
			단가 : 2,000원 총 합계 : 2,000원		
			1개		
1	bookName1				삭제
			단가 : 1,000원 총 합계 : 1,000원		
			1개		
		1			
총 주문 금액 : 6,000원					
		주문하기			

설명 : 1페이지당 5개, 장바구니 물건 최대 15개로 제한. 총 3페이지. 장바구니 페이지에 있는 상품들은 JS쪽에서 배열로 저장되어 있음.

AJAX 사용해서 동적처리함. 이벤트 변화에 따른 장바구니 데이터가 서버에 저장됨. 이벤트가 시작되면 서버로 요청을 해서 데이터를 저장함.

- **총 합계** : 개수 올리면 자동으로 변경. 이벤트 발생시 배열안에 있는 특정요소를 탐색하여 해당 요소의 총합을 구함.
- **총 주문 금액** : 모든 페이지의 주문 합계 계산됨. 2, 3 페이지의 가격또한 포함. 배열을 통한 탐색이므로 뒤쪽 페이지의 대한 금액또한 쉽게 구할 수 있음.
- **최대 개수 제한 이유** : 장바구니에 제한없이 계속해서 넣는 것은 이상할 뿐더러 제한을 너무 폭넓게 잡으면 장바구니에 있는 상품을 주문시 너무 많은 입력이 발생하므로 제한을 하였음.

6. 주문

메인

주문신청목록

주문완료목록

주문반려목록

주문번호: 23

2025-05-11

주문 도서 목록

3

bookName3 주문수량 : 1권 단가 : 3,000원 총 합계 : 3,000원

2

bookName2 주문수량 : 1권 단가 : 2,000원 총 합계 : 2,000원

1

bookName1 주문수량 : 1권 단가 : 1,000원 총 합계 : 1,000원

총 주문신청금액 : 6,000원

주문신청취소

주문번호: 22

2025-05-11

주문 도서 목록

주문처리과정 : 장바구니에서 주문을 하면 주문데이터를 입력하게 되는데 주문데이터를 입력시 주문과 주문상세를 INSERT ALL을 이용해서 하나의 트랜잭션으로 묶어서 처리함. 이렇게 하나로 묶지 않고 따로 처리하게 되면 주문과 주문상세의 데이터가 일치하지 않는 경우가 발생할 수 있음.

주문확인 : 주문한 것을 확인할 때마다 총주문금액 컬럼을 추가하지 않고 주문에 있는 상품의 총합을 구하는 연산을 통해 구할 수 있음. 하지만 이렇게 하면 주문을 확인할 때마다 연산을 해야하는 비효율적인 과정이 생김. 따라서 주문 테이블에 총주문금액 컬럼을 추가하여 이러한 연산과정을 없앴.