**<프로젝트 주차 별 진행 상황>**

|  |
| --- |
| **1주차 (2009/1/31 ~ 2/1)** |
| 1. 기본 개발 환경 설정   1) Eclipse + Spring 2.5 + Maven  2) Eclipse Plug-ins 설치하기  - Eclipse 3.4+ JEE 버전  - Subversive  - M2Eclipse  - Spring IDE  3) MySQL 5.x  - Database는 UTF-8 인코딩을 사용  - Apache Commons DBCP 1.2.2이용  4) Tomcat 6.0 이상  - server.xml 파일에 DB Connection Pool 설정  - META-INF/context.xml 설정  - Spring 2.5 ApplicationContext 설정(Annotation 방식)  - Tomcat과 Eclipse 연동  5) 위의 환경을 자신의 개발 PC에 설정하고 Hello BookTogether 메시지 띄우기  2. 회의 내용 정리  1) 프로젝트에 대한 이해  - 프로젝트의 내용 정리  - 기존에 제안된 기능 이외에 추가할 사항  2) Spring MVC와 Service, Dao 개념 이해  3) 2주차 개발 계획 및 업무 분담 |

|  |
| --- |
| **2주차(2009/2/2 ~ 2/8)** |
| 1. 개발 환경 설정 추가  1) Dependency 추가  - JSTL : taglibs/standard 1.1.2 추가  - MySQL JDBC 드라이버  - Commons DBCP 커넥션 풀  - Log4J 1.2  2) Log4J 설정하기  - 기본적으로 화면으로만 로그 남기기  - 로그 찍을 때는 Commons Logging 사용하기    3) 위와 관련된 문서 읽기  - Log4J 설정 : http://kwon37xi.egloos.com/2176487  - Tomcat과 한글 : http://kwon37xi.egloos.com/1462013  - JSP 2.0 변경사항 : http://kr.geocities.com/kwon37xi/jsp20/WhatsNewJSP20-1.html  - JSTL 1.1  2. 코딩 규칙 정의  1) 패키지 구조   |  | | --- | | * com.google.code.booktogether.web : 웹 관련   + com.google.code.booktogether.web.controller : 컨트롤러들   + com.google.code.booktogether.web.interceptor : 인터셉터 * com.google.code.booktogether.service : 서비스 인터페이스   + com.google.code.booktogether.service.impl : 서비스 구현 * com.google.code.booktogether.dao : dao 인터페이스   + com.google.code.booktogether.dao.impl : dao 구현 |       2) 명명 규칙  - 서비스 인터페이스가 CurrentDateService라면 **구현체**는 CurrentDateService**Impl**  Dao 인터페이스가 UserDao라면 **구현체**는 UserDaoJdbc**Impl**  위와 같은 방식으로 파일명 정의  - 디렉토리 구분자는 유닉스 방식으로 "/"를 사용  3) 주석 달기  - 클래스, 메소드, 필드에 주석을 달 때는 아래와 같은 형태로 달 것   |  | | --- | | /\*\*  \* 주석처리 코드  \*/ |     \* Eclipse에서는 클래스, 메소드, 필드에 주석을 달 때 /\*\* 치고 Enter 누르면 자동으로 형태를 잡아준다.  3. 프로젝트 개발 내용  1) 도메인 클래스용 DB 테이블 설계  - User  - Book  2) 사용자 추가, 수정, 삭제  - User : 도메인 클래스  UserDao/UserDaoJdbcImpl  UserService/UserServiceImpl  UserController  - 사용자 로그인/로그아웃  사용자의 비밀번호를 저장할 때는 MD5와 salt로 암호화해서 저장  비밀번호 입력 정보를 로그로 남기지 말 것  회원가입, 리스트, 정보조회 등  3) 책 정보 추가, 수정, 삭제  - Book : 도메인 클래스  BookDao/BookDaoJdbcImpl  BookService/BookServiceImpl  BookController  - 책 검색 OpenAPI : daum 책 검색 API를 사용  4. 회의 내용 정리  1) 별점 및 리뷰 작성 및 구현  2) 추가로 구현하고 싶은 내용  3) 3주차 개발 계획 및 업무 분담 |

|  |
| --- |
| **3주차 (2009/2/9 ~ 2/15)** |
| 1. Code Review  - 구현한 코드를 함께 보며 수정 사항과 차후 개발할 내용을 정리  1) 기능별 구현 내용   |  |  |  | | --- | --- | --- | | 구분 | 내용 | | | 사용자(User) | - 로그인/로그아웃  - 회원정보 수정  - 비밀번호 수정  - 아이디 중복확인  - 아이디/비밀번호 찾기 | - 회원목록  - 사용자 조회  - 사용자 탈퇴  - 기본정보 등록  - 생활반경 등록/검색 | | 책(Book) | - 책 등록(직접 입력)  - 책 등록(API정보 이용)  - DB에 등록된 책 목록 | - 책 검색(API 이용)  - 책 조회/수정/삭제 | | 별점(Grade) | - 별점 등록/목록/삭제 | - 별점 조회 | | 리뷰(Review) | - 리뷰 등록/수정/삭제  - 리뷰 목록 | - 리뷰 조회  - 리뷰 추천하기 | | 인용구(BookMark) | - 인용구 등록/수정/삭제  - 인용구 목록 | - 인용구 공감하기 |     2) 공통 구현 내용   |  | | --- | | * SQL관리   + XMLUtil클래스를 이용하여 XML문서로 관리 * RESULT SET 관리   + com.google.code.booktogether.dao.rowmapper 패키지로 관리 * Scripts, Styles   + /scripts/ 안에 common, book, user 별 폴더 생성   + /Styles/ 안에 common, book, user 별 폴더 생성 * DB Table |   3) 미개발 내용 및 기타 아이디어  - 비밀번호 이메일 전송  - 자신의 블로그로 리뷰와 인용구 포스팅하기  - 나를 기준으로 책 상태 분류  - 내가 읽은 책/읽고 있는 책/읽고 싶은 책 구분  - 책 소유 여부  - 사용자의 생활 반경에 존재하는 책 소유자를 조회하여 책 돌려보기  - 관심서재 등록하기  - 특정 책을 좋아하는 사람들이 좋아하는 다른 책들 추천해주기  - 내 서재에 왔을 때 내가 별점 높게 준 것을 더 부각시켜 주기  - 특정 그룹끼리 함께 보면 좋을 만한 책들 추천해주기  4. 회의 내용 정리  1) 작명규칙 추가  - 직관적인 메소드명, 변수명, 파일명으로 변경하기  (어떤 기능을 하는지 이름만으로도 알 수 있도록 직관적으로 명명하기)  - 변수명과 메소드명에 ‘\_(underbar)’ 삭제  2) 패키지의 인터페이스/구현을 분리  - 기능에 따라 패키지 분리  - 인터페이스는 public으로 접근 가능하도록 하고, 구현은 private으로 숨기도록 함  - 책 검색 부분 인터페이스로 빼고 bean으로 객체 한번만 생성하도록 변경  3) XmlUtil.java 수정사항  - 직관적으로 파일명 수정 Ex)XmlSqlParser.java  - sql.xml 값을 코드로 고정시키지 말고 생성자에서 문자열의 List로 받는 방법  (ApplicationContext에서 객체를 생성하는 방법을 찾아보기)  4) Transaction 처리  - @Transactional(....)을 클래스 정의 위에 적어주면 공통으로 적용  (readOnly=true가 default, write가 발생할 수 있는 것에만 readOnly=false로 따로 지정)  - Transactional은 RuntimeException이 default.  5) Primitive data type은 사용하지 말 것  - Primitive data type에는 NULL 개념이 없다.  (""와 NULL은 다르기 때문에 IS NULL 조건으로 DB를 조회할 경우  ""은 조회되지 않는 문제 발생)  - int 🡪 Integer  6) Exception 처리  - 모든 Exception은 BooktogetherException?을 상속하여 사용  - 이유 없는 Exception을 발생시키지 말 것(반드시 message 넣어주기)  - Checked Exception의 Try catch 부분에서 error를 잡아먹는 실수를 하지 말 것  (Try catch finally 부분에서 return()은 절대로 해서는 안된다!)  - printStackTrace()는 절대 쓰지 말 것  7) 주소 검색  - Type0 사용(읍,면,동까지 DB에 저장)  - 나와 일치하는 생활반경 검색 시 읍면동까지 일치하는 경우를 가장 위로 정렬  8) 이클립스에서 배포가 잘 되지 않는 문제  - project > properties > java build path > source 탭 선택 > Excluded 삭제하기    9) 4주차 개발 계획 및 업무 분담  - Refactoring 작업  - Interceptor의 적용(RequestContextHolder 클래스 참고)  - 내 서재 만들기(책 상태 분류 기능 포함)  - 페이지 처리  (이전 화면으로 돌아갈 때 이전에 가지고 있던 모든 파라미터를 가지고 가도록)  - 스프링 공부(고급 기능을 사용하기 위하여)  - 비밀번호 암호화(MD5, Salt를 이용하여 이중으로 보안)    10) 차후 추가 개발할 기능 생각해보기  - 관리자 페이지  - 스프링에서 제공하는 링크 방식의 주석 처리(현재는 인터페이스에만 주석처리)  - Validator에 대한 고려  어떻게 하면 key 값만으로 주소(생활반경)를 비교할 수 있을지 생각해보기  5. 참고사항   |  | | --- | | * 개발자를 위한 font - DejaVu Sans mono (다운로드: <http://blog.naver.com/empireclan?Redirect=Log&logNo=140062917666>) * 핵심기능에 집중할 것 - 부가적인 기능은 차후에 생각하자. * Formatting 수행(Ctrl + Shift + f) - 저장하기 전에 잊지 말고 할 것 * System.out.println 구문은 소스에 절대 포함되지 않도록 할 것 * session은 메모리와 연결되므로, 남용하지 않고 잘 지워주도록 할 것 * 참고 도서 - Effective Java, Java5, JQuery(위키북스) | |

|  |
| --- |
| **4주차 (2009/2/16 ~ 2/22)** |
| 1. Code Review  - 구현한 코드를 함께 보며 수정 사항과 차후 개발할 내용을 정리  1) 기능별 구현 내용   |  |  |  | | --- | --- | --- | | 구분 | 내용 | | | 서재(Library) | - DB Table 생성  - Controller 생성  - Service/Impl 생성 | - SQL.xml 추가  - RowMapper 추가  - 기본 jsp 틀 생성 | | 책(Book) | - 책 리스팅 관련 기능  - DB Table 추가  - Controller 추가  - Service 추가  - DAO 추가 | - 내가 보유한 책 CRUD  - DB Table 생성  - Controller 추가  - Service 추가  - DAO 추가 |       2) 수정사항 확인  - 불필요한 코드 삭제 ( System.out.println(), try{}catch(){} )  - 변수 및 메소드 이름 작명규칙에 따라 수정  - Eclipse Formatter로 정렬  - Primitive data type 수정 (int 🡪 Integer로 변경)  - String형 변수 “”을 NULL로 변경  - ModelMap으로 변경  - jsp 상대 경로 변경      3) 디버깅 작업  - 세션 관리 문제  - 로그인 문제  - 페이지 표시 및 오류 문제  - DB insert시 이중삽입 문제    2. 회의 내용 정리  1) ServiceImple if ~ else 구문 정리  - 가독성 높이기    2) BaseObject 부분 abstract로 변경  3) requestContextHolder 이용하여 코드 간단하게 변경  4) xss(Cross Site Scripting) 보안 처리  - {fn: } 이용해서 처리  - server 단에서 기본 사용자에게 기본 tag만 허용하도록 처리  - 사용자가 직접 작성하는 리뷰, 인용구의 경우에만 기본 tag 제공  - 자바 정규표현식을 이용하여 필터링하는 방법  (정규식 알기 - 참고 : <http://pupustory.tistory.com/132>  htlm 태그 및 금지어 필터링을 적절히 믹스  - 참고 : <http://www.java2go.net/blog/tag/Regular%20Expression>)  5) Primitive data type은 사용하지 말 것  - 공통되는 부분은 그대로 두고 API 종류에 따라 달라지는 것들만 Map으로 구현  6) JQuery 공부하기  - JQuery 강좌 링크 :  <http://blog.naver.com/guruby?Redirect=Log&logNo=140058679107>  7) 기타 구현할 기능 마무리  - 내 생활반경에 등록된 책 리스트 조회  - 내 관심서재 등록  - 그 외의 사소한 이슈들  8) Logging 처리시 레벨 체크 메서드 사용이유  - 비용계산문제에서 발생 - 참고 : <http://okcode.egloos.com/1892085> |

|  |
| --- |
| **5주차 (2009/2/23 ~ 3/1)** |
| 1. Code Review  - 구현한 코드를 함께 보며 수정 사항과 차후 개발할 내용을 정리  1) 기능별 구현 내용   |  |  | | --- | --- | | 구분 | 내용 | | 서재(Library) | - 관심서재 기능 추가  - 등록/삭제/조회/목록  - 서재 검색기능 추가(공개 서재만 허용)  - 비공개 서재 설정 기능 추가  - 소유책 등록 방법 추가 | | 책(Book) | - 책 조회 페이지 수정  - 리뷰, 별점, 인용구 목록 표시 수정 | | 방명록(Board) | - 개인서재에 방명록 추가하기  - 로그인 한 사용자만 글쓰기 권한 부여  - 글 삭제 권한 (서재주인/글 등록한 사용자)  - 비공개일 경우 등록, 조회, 목록, 삭제 불가  (서재 주인만 허용)  - 서재 순위 기능  (활동을 많이 하는 10명의 사용자를 리스트로 출력)  - 서재 내에 존재하는 책 검색 기능 추가 |   2) 비공개 서재 설정 기능  - 비공개 서재에 접근하는 경우 비공개 메시지 띄우고 이전 화면으로 돌아가기  - 리뷰, 별점, 인용구는 검색가능(서재가 아닌 개인 정보만 비공개)  - 읽고 싶은 책, 읽고 있는 책, 읽은 책은 서재의 정보이므로 조회할 수 없음  - 서재 주인일 경우만 조회가 가능함  3) 소유책 등록 방법 추가  - 기존에는 서재에 등록된 책을 소유하고 있으면 정보를 입력하는 방식  - 책 조회 시 바로 소유책 등록 화면으로 이동하여 등록이 가능하도록 함      4) 수정사항 확인  - ServletRequestAttributes 클래스 사용을 변경  ( 변경후 RequestContextHolder.getRequestAttributes() 해당 클래스는  Controller에서 사용함으로 Controller에서 부분 수정  - 책 검색시 <b></b>태그 삭제  (HtmlInputFilter 클래스로 처리)  - 검색 인기 순위 기능 추가  ( booktogether에서 검색 count를 계산하여 1위~10위까지 표시)  - BaseObject 수정  ( 기존의 BaseObject클래스를 Abstract Class로 변경하여 상속하는 클래스에서 구현  equal(), hash(), toString() 재정의 )  5) 디버깅  - 별점 삭제 후 페이지 이동 오류 수정  (별점이 삭제된 책이 아닌 다른 책의 정보가 보여짐)  - 소유 책 정보 수정 오류  (읽고 싶은 책, 읽은 책, 읽고 있는 책 중에 소유중하지 않은 책에서  소유로 수정할 경우 내 소유책 목록에 나타나지 않는 오류)  - 줄바꿈 문제 수정  ( 인용구, 리뷰, 내 서재 인사말 대상으로 입력 시 입력문자열 그대로 삽입하고,  출력 시 \r\n문자를 replaceAll()함수로 "<br/>"로 변경)  2. xss 문제 해결   |  | | --- | | \* 짧은 문장의 내용 ex) : 제목, 이름, 인용구  - 출력 : {fn:}으로 출력  - server-side에선 정규화 표현식으로 html삭제  \* 긴 문장의 내용 ex) 리뷰내용, 자기 서재 소개(인사말)  - 일정 태그 허용  ex)img, b, p, br,  \* OPEN API를 이용하여 DB에 Insert되는 부분은 필터링 하지 않음  - 표시부분은 {fn:}으로 출력 |   3. 회의 내용 정리  1) 페이징 처리  - 현재 페이지 기능은 보여주기만 할뿐 페이지간 이동은 없음  - JSP 페이지 안에 페이지 기능 추가  - Controller페이지 정보 파라미터로 수신  - 이동한 JSP페이지에선 URL에 페이지 정보를 표시(페이지 정보 가지고 있어야 함)  2) Findbugs로 소스 수정  - finbug plug-in으로 기존 소스의 버그를 찾고 디버깅  3) Thumbnail 수정  - java include 오류 발생하는 부분  참고 : <http://schmidt.devlib.org/java/creating-thumbnails-with-java.html>  **4) UI 정하기**  **- 메뉴 정하기**  **- HTML 기본 레이아웃 정하기**  **- CSS로 기본 스타일 정하기**  **5) DB ForeignKey 지정**    6) sitemesh 이용 또는 jsp 파일 include 여부 결정 |
| **6주차 (2009/3/2 ~ 3/8)** |
| 1. Code Review  - 구현한 코드를 함께 보며 수정 사항과 차후 개발할 내용을 정리  1) 내 생활반경의 책 목록 가져오기   |  | | --- | | \* 내 생활반경의 책 목록 가지고 오기 구현  - 표시 정보(책 이미지, 책이름, 사용자 아이디, 지역명)  \* 책 목록 정보 중에 사용자 아이디 클릭하면 사용자 서재로 이동  \* 책 이미지나 책 이름 선택시  - 소유 책 정보 조회  \* 지역명 클릭시 지역명의 소유 목록 가지고 오기  \* 생활반경이 정확히 일치하는 사람만 검색되도록 구현  - 생활반경을 포함하여 그보다 더 큰 구역도 검색되도록 수정  - 우편번호를 이용한 SQL문으로 변경할 것을 고려 |       2) 수정사항 확인  - Image Resize 수정하기  - DB ForeignKey 지정 하기  - 아이디 중복 확인 시 새창 열기를 팝업으로 수정  (아이디 입력 필드를 클릭했을 때 중복확인 팝업이 바로 뜰 수 있도록 수정,  하나의 팝업에서 아이디 중복을 체크하기 위해 JSON 객체를 이용하여 처리)  2. UI 설계  1) 메인 페이지 UI 설계  - 메인 페이지의 레이아웃을 기본으로 다른 페이지를 구성  2) 메인 페이지 CSS 구성  - 기존의 구현된 기능을 접목하여 CSS로 구성하기  3. 회의 내용 정리  1) Quartz를 이용한 스케줄링 처리      그림 UI 설계 그림 |

|  |
| --- |
| **7주차 (2009/3/9 ~ 3/15)** |
| 1. Code Review  - 구현한 코드를 함께 보며 수정 사항과 차후 개발할 내용을 정리  1) 기능별 구현 내용   |  |  | | --- | --- | | 구분 | 내용 | | 레이아웃(Layout) | - Top 메뉴의 기능 구현  - 메뉴 네비게이션 바  - 미구현 페이지 ‘준비중’ 이미지 삽입  (About us, Help, FAQ, Sitemap) | | 공통(Common) | - No Image 이미지 추가  (이미지 없는 경우 default 이미지가 출력되도록 수정) | | 사용자(User) | - 사용자 썸네일 이미지 변경 | | 책(Book) | - DB Table 수정(Author의 name 필드 사이즈 변경) |   2) 사용자 썸네일 이미지 변경  - 사용자가 썸네일 이미지를 변경 후 기본 썸네일 이미지로 변경하고자 할 때  변경 할 수 있도록 구현  - 개인정보 수정 시 기본 이미지로 변경 버튼을 등록하여 변경 가능하게 함  3) Quartz를 이용한 스케줄링 처리  - 활동이 많은 서재를 DB에서 조회해오는 것을 스케쥴링으로 처리  4) Input Field 유효성 검사  - Javascript 이용  - Controller에서도 검사    5) 디버깅  - UserController에서 아이디 중복 체크의 res.setContentType() mime타입 수정  🡪 application/json으로 변경  - Top의 책/서재 검색 부분 default 조건 지정  - 데이터량이 많은 경우 페이지 에러 발생  - 전체적인 페이지 목록 수정   |  |  | | --- | --- | | - 별점 목록  - 인용구 목록  - 읽고 있는 책 목록  - 읽은 책 목록  - 읽고 싶은 책 목록 | - 리뷰 목록  - 소유책 목록  - 내가 쓴 리뷰 목록  - 내가 입력한 인용구 리뷰 목록  - 내가 매긴 별점 목록 |   2. 프로젝트 최종 제출물 정리 작업  3. 회의 내용 정리  1) 핵심기능에 집중하기!  - 블로그에 포스팅하기 기능이 빠져있다.  - Meta Weblog API 이용  ( 멘토님 스프링노트 참고 : <http://kwon37xi.springnote.com/pages/871502> )  2) Top의 책/서재 검색 부분 default 표시  3) 주소 DB의 자동화  - 데이터가 갱신되면 명령어를 통해 DB가 자동으로 update될 수 있도록 하기  - Maven/Ant 스크립트 이용 또는 batch 파일 만들기  4) 화면 구성  - 정보를 보여주는 부분/입력이 필요한 부분을 분리하기  책의 정보가 중요 🡪 정보를 보여주는 부분을 위로, 입력을 아래쪽으로 배치  별점목록 등 핵심이 아닌 부분이 차지하는 size 줄이기  - 책을 검색했을 때, 해당 책의 소유자 목록을 간단히 보여주기  다른 서재로 쉽게 접근할 수 있도록 링크 걸기  - 내 소유책 등록화면에서 책 가격은 삭제하거나 기본적으로 정가가 입력되도록 수정  - 필수입력과 선택입력 필드에 대한 구분이 명확하도록 표시하기  - 내 서재 수정사항  Content 부분: 사용자가 최근에 등록한 책의 리뷰, 인용구 등을 보여주기  자신의 최근 활동을 알 수 있도록 하기  인사말은 사용자가 적지 않았으면 보여주지 말기  사용자가 보유하고 있는 책의 권수를 출력, 등록된 책 권수를 list에 보여주기  - 서재의 왼쪽 리스트에서 책검색 빼기  5) form의 method 변경  - DB가 변경되는 action이 있는 경우, post가 원칙  - search에서 이전 단계로 돌아가면 만료된 페이지로 나옴 🡪 get으로 변경  6) 기타  - alert창의 남용은 UI를 해치는 일  결과화면을 바로 보여주거나 일정시간이 지나면 자동으로 페이지가 이동하도록 수정  - 다른 서재로 쉽게 접근할 수 있도록 하기  공유에 초점을 맞추자  - href와 같은 불필요한 javascript 사용하지 않기  링크를 걸어 처리할 수 있는 것은 링크를 걸어서 처리  - Exception 페이지 만들기  사용자는 자세한 오류 메세지를 확인할 수 없도록 해야 함  개발자만 확인할 수 있도록 간단한 로그만을 남길 것  - 중복되는 코드 줄이기  반복되는 코드에 대해서는 함수 만들기  책표지/사용자 default 이미지 보여주는 부분  - html의 표준 코드 삽입  - UserController에서 아이디 중복체크 부분  res.setContentType() 부분 mime 수정 -> application/json  - Author DB의 name 필드 사이즈 늘리기  **-** 소스 제출할 때 WAR 파일로 만들어서 실행하는 방법을 알려주는 방식으로 하기 |
| **8주차 (2009/3/16 ~ 3/18)** |
| 1. Code Review  - 구현한 코드를 함께 보며 수정 사항과 차후 개발할 내용을 정리  1) 기능별 구현 내용   |  |  | | --- | --- | | 구분 | 내용 | | 서재(Library) | - 등록한 리뷰 블로그에 포스팅 기능 추가  (MetaWeblog API 이용)  - 서재의 왼쪽 리스트에서 책검색 빼기  - 내 소유책 등록 화면에서 가격 입력 삭제  - 서재 정보 표시 수정 | | 공통(Common) | - 필수/선택 입력 필드에 대한 구분이 명확하게 표시하기 | | 책(Book) | - 책 정보 화면 수정  (정보를 보여주는 부분/사용자가 입력하는 부분 분리)  - 별점 목록 작게 표시  - 책 소유자 목록 표시 | | 사용자(User) | - 로그인시 Password 입력필드에 focus가 있으면  enter 눌렀을 때 로그인 가능하도록 하기  (JQuery trigger 구현) |   2) 서재 조회시 변경될 정보  - 읽고 싶은 책, 읽고 있는 책, 읽은 책이 등록된 권수 표시  - 소유책, 리뷰목록 등 표시할 수 있는 정보는 다 표시하기  - 서재 메인 content부분에 서재의 활동 내용을 표시하기  (읽고 싶은 책, 읽고 있는 책, 읽은 책, 소유책, 인용구 목록, 리뷰목록 등)  3) 예외처리 페이지 완성  - 예외처리에 맞게 표시되게  - 505 페이지 에러  - 404 페이지 에러  - 기타 예외페이지  4) Form의 method 변경  - DB가 변경 되는 action이 있는 경우, post가 원칙  - search에서 이전 단계로 돌아가면 만료된 페이지로 나옴  🡪 get으로 변경      5) <a href="javascript:"/> 내용 삭제  - javscript:함수명() 되도록이면 제거  - submit이나 confirm을 사용할 경우를 제외한 나머지는 수정  6) HTML의 표준 코드 추가   |  | | --- | | <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">  <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ko"> |   7) alert() 삭제  - alert()창의 남용은 UI를 해치는 일  - 결과 화면을 바로 보여주도록 수정  8) 중복되는 코드 줄이기  - 반복되는 부분에 대해서는 함수 만들기  - 책표지/사용자 default이미지 보여주는 부분  - alert()창 표시하는 중복 코드 삭제    9) reflushLibraryRank 수정  - 두 개의 메소드를 하나로 합치기  - deleteLibraryRank() + reflushLibraryRank() 🡪 reflushLibraryRank()로 합치기    2. 프로젝트 최종 결과물 정리 |