

## 스프링 프레임워크 개발자 매뉴얼

프로젝트 정보	
프로젝트명	MinGW's 도서 관리 프로젝트
프로젝트 소개	<p>2인 1조 팀원으로 중간고사 프로젝트에서 구현했던 기존의 콘솔 도서 관리 프로그램이 아닌 뷰 형식의 사용자 친화적 디자인을 추구하여 처음 프로그램을 실행하는 사용자가 쉽게 사용 할 수 있도록 구현하였습니다.</p> <p>중간고사와 달리 하나의 도서에 대한 재고 비축과, 도서 연장 서비스, 지속적 연체 회원 문제에 대한 해결법 등 실제 도서 사이트로 이용될 수 있을 만큼 다양한 서비스를 준비하였습니다.</p>
구성도	이미지가 너무 커서 따로 파일로 별첨하겠습니다.
특 · 장점	<p>실제 도서 사이트와 같이 비회원도 도서 대여와 희망 도서 신청, 게시글 작성 이외의 도서 검색, 자유 게시판 열람, 공지 사항을 읽는 다양한 서비스를 받을 수 있다는 특징이 있습니다.</p> <p>또한 실제 도서관처럼 같은 도서가 여러개 존재할 수 있음을 인지하고 재고 수를 속성으로 추가하여 여러 사람이 동일한 도서를 한정된 수량에 한하여 동시에 대여할 수 있도록 제공합니다.</p> <p>또한 습관적 연체 회원과 반납을 잘 하는 회원을 랭크로 구분하여 대여 횟수에 차등을 줌으로써 연체라는 행위를 의식적으로 지양하도록 유도하는 특별한 서비스를 제공합니다.</p>
주요 기능	<ul style="list-style-type: none"> <li>- 기본 기능 : 회원 가입, 회원 정보(비밀 번호) 수정, 다중 사용자의 대여와 반납, 도서 검색</li> <li>- 사용자(비, 회원) : 자유 게시판을 통한 후기 나눔, 대여 도서 기간 연장, 비치되길 원하는 도서에 대한 희망 도서 등록 등</li> <li>- 관리자 : 도서 등록, 수정, 삭제 와 도서 관리 및 공지 사항 등록, 삭제 와 추천 도서 등록, 삭제 및 회원 정보(블랙리스트 관리) 수정, 희망 도서 관리, 자유 게시판 관리 등</li> </ul>

## 목차

I. 프로젝트 개요

II. 프로젝트 내용

III. 프로젝트 수행내용

VI. 팀원 별 역할 표 및 기타 개발에서의 노력

## I. 프로젝트 개요

### 1. 프로젝트 소개

- 국립 중앙 도서관 사이트를 벤치마킹하여 구현한 도서 관리 프로그램
- 비회원, 회원, 관리자 등 **각자의 역할에 따라** 페이지 혹은 기능을 달리 하여 각자에게 필요한 서비스를 달리 제공받을 수 있도록 구현하였습니다.
- 습관적 연체자와 같이 실제 도서관을 운영하면서 있을 수 있는 문제 사항을 연구하였고 이에 대한 해결책을 구현하였습니다.

### 2. 팀원별 역할

- 김민지 : Main (Front) + Sub (Back)
  - JSP, JS, CSS 등 실제 화면에서 보이는 Web content 디자인들을 설계 및 구현하였습니다.
  - 데이터 베이스의 구조를 설계 몇몇 Controller 자바 파일 및 DTO 파일 구현
- 정민지 : Main (Back) + Sub (Front)
  - 실제 프로그램에서 기능이 동작하기 위한 Service 자바 파일과 SQL 합쳐야 하는 DAO 패키지 속 자바 파일을 제작
  - 실제 데이터베이스 레코드와 연결하기 위한 데이터를 매핑하는 DTO를 구현 및 Controller에 필요한 예외처리 자바 파일 제작
  - JSP 파일 속 몇몇 JSTL 코드 구현

### 3. 벤치마킹

- 국립 중앙 박물관의 메인 화면 속 네비게이션과 각 섹션 별 기능에서 영감을 받았습니다.
- 경기 도서관의 도서관 소개 페이지를 참고하여 도서관의 특징 중 하나인 도서 소개 서비스를 구현하였습니다.

## 4. 개발목표 및 내용

### ○ 최종 개발 기능

#### - 비회원, 회원 사용자

- 사서 추천 도서, 인기 도서, 신간 도서 확인 - 공통
- 도서 : 도서 검색 (공통), 대여 (회원), 반납 (회원), 연장 (회원)
- 자유 게시판 후기 게시, 댓글 추가 -회원 전용
- 회원 가입, 본인 정보 찾기, 회원 정보(비밀 번호) 수정 - 회원 전용
- 비치되지 않은 도서를 관리자에게 비치하도록 요구하는 희망도서 신청 - 회원 전용

#### - 관리자(사서) 사용자

- 회원 사용자에게 신청 받은 희망 도서 목록 확인 가능
- 도서 : 도서 추가, 수정, 삭제
- 연체 도서 확인, 블랙리스트 (습관적 연체자) 랭크 관리
- 비치되어 있는 도서에 한해 사서 추천 도서 게시글 추가 및 삭제
- 비회원 및 회원 모두에게 알리는 공지 사항 게시글 추가 및 삭제
- 자유 게시판에서 불량 게시글을 비공개 처리함으로써 자유 게시판 관리

### ○ 주요 개발내용 (기능 특이사항)

(BACK) - 본인 정보 찾기 : 실제로 사용하는 E-mail을 입력하여 본인 정보 (비밀 번호)를 찾을 시 정규식을 통해 랜덤으로 생성한 문자열을 통해 비밀번호를 초기화 하고 실제 E-mail 로 전송합니다.

(FRONT) - 글자 수 제한 : DB 테이블의 속성들에 대하여 각각의 크기를 회원, 관리자가 알기 어렵기 때문에 자유 게시판의 게시글, 댓글 및 관리자가 도서, 공지사항 등을 추가하면서 입력하는 글자가 DB Table 에 추가될 때 에러가 나지 않도록 미리 최대 글자 입력 한계를 정하여 시각적으로 보여줌과 동시에 자동적으로 한계에 벗어난 글자를 삭제합니다.

(DB TABLE) - Member 테이블의 속성 중 ‘정상 반납 횟수’ 속성과 ‘랭크’ 속성, ‘대여 한계 횟수’ 속성을 연계하여 정상 반납과 연체를 나누어 랭크를 변경하고, 이 랭크에 따라 대여 한계 횟수 또한 변경 되게 구현함으로써 회원 본인의 행동에 따라 동적으로 서비스 제공 한계를 변경합니다.

또한 계속된 연체로 대여하지 못하는 회원은 영구적으로 대여를 못하는 상황 또한 발생 가능성을 가정 하였고, 이를 해결하기 위해 연체 회원이 직접 관리자에게 용서를 구함과 동시에 랭크 복구 요청을 할 시 관리자가 직접 연체자의 랭크를 일반 회원으로 다시 상승 시킬 수 있도록 예외 처리를 하였습니다.

## II. 프로젝트 내용

### 1. 구성도

- 이미지의 크기가 커서 따로 파일로 첨부하겠습니다.

### 2. 프로젝트 구조

- 전체 프로젝트 구조

구분	기능	설명
src/main/java	libraryManageApplication	전체 프로젝트를 실행하는 메인
	libraryManageConfig	Bean 객체를 만들어 Spring JDBC 를 이용하여 MySQL 데이터 베이스에 접근하도록 세팅
	libraryManage DAO	JDBC를 이용해 MySQL 과 직접 연결되어 구현한 SQL 문을 실행
	libraryManage DTO	MySQL 내에 존재하는 DB Table 의 구조를 java 로 구현하여 setter,getter 등으로 실제 존재하는 데이터에 접근할 수 있도록 함
	libraryManage SERVICE	사용자가 이용하는 기능들이 실행될 수 있도록 알고리즘 및 코드 구현
	libraryManage EXECPTION	예외 처리
src/main/resources	STATIC	프로젝트 기반을 위해 존재하는 이미지 뿐만 아니라 JSP 파일에서 사용되는 특정 기능을 구현해주는 JS 파일 등
src/main/webapp	WEB-INF VIEWS	실제로 사용자가 만나는 화면을 구성하는 JSP 파일이 존재
src/main/webapp	BOOK IMAGE STORAGE	관리자 사용자가 도서 등을 추가하면서 업로드 하는 이미지가 저장되는 경로

○ DTO

구분	기능	설명
DTO	BoardDTO.java	자유 게시판 테이블과 맞췄는 구조체 파일
	BookDTO.java	도서 테이블과 맞췄는 구조체 파일
	CheckOutDTO.java	대여 테이블과 맞췄는 구조체 파일
	CommentDTO.java	댓글 테이블과 맞췄는 구조체 파일
	GoodDTO.java	추천 도서 테이블과 맞췄는 구조체 파일
	HopeDTO.java	희망 도서 테이블과 맞췄는 구조체 파일
	MemberDTO.java	회원 테이블과 맞췄는 구조체 파일
	NoticeDTO.java	공지 사항 테이블과 맞췄는 구조체 파일
	PhraseDTO.java	독서 명언 테이블과 맞췄는 구조체 파일

○ DAO

구분	기능	설명
DAO	BoardDAO.java	자유 게시판 테이블에 직접 접근하여 업데이트하는 파일
	BookDAO.java	도서 테이블에 직접 접근하여 업데이트하는 파일
	CheckOutDAO.java	대여 테이블에 직접 접근하여 업데이트하는 파일
	CommentDAO.java	댓글 테이블에 직접 접근하여 업데이트하는 파일
	GoodDAO.java	추천 도서 테이블에 직접 접근하여 업데이트하는 파일
	HopeDAO.java	희망 도서 테이블에 직접 접근하여 업데이트하는 파일
	MemberDAO.java	회원 테이블에 직접 접근하여 업데이트하는 파일
	NoticeDAO.java	공지 사항 테이블에 직접 접근하여 업데이트하는 파일
	PhraseDAO.java	독서 명언 테이블에 직접 접근하여 업데이트하는 파일

○ Service

구분	기능	설명
Service	BoardService.java	BoardDAO 와 BoardController를 연결해주는 파일
	BookService.java	BookDAO 와 BookController를 연결해주는 파일
	GoodService.java	GoodDAO 와 GoodController를 연결해주는 파일
	MemberService.java	MemberDAO 와 MemberController를 연결해주는 파일
	NoticeService.java	NoticeDAO 와 NoticeController를 연결해주는 파일

○ Controller

기능		설명
Controller	AdminBoardController.java	관리자 게시판 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	AdminBookController.java	관리자 도서 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	AdminController.java	관리자 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	AdminGoodController.java	관리자 추천 도서 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	AdminMemberController.java	관리자 회원 관리 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	AdminNoticeController.java	관리자 공지 사항 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	BoardController.java	게시판 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	BookController.java	도서 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	GoodController.java	추천 도서 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	MainController.java	메인 화면 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	MemberController.java	회원 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스
	NoticeController.java	공지사항 기능 주소 매핑 및 사용자가 요청한 기능을 구현하는 클래스

○ 기타

기능		설명
	LibraryManageApplication.java	실제 프로젝트를 실행하는 파일
	JavaConfig.java	Bean 객체 생성
예외 처리	AlreadyExistingException.java	데이터가 이미 존재할 시 예외 처리
	BlackListException.java	회원 등급 처리 시 예외 처리
	ConfirmNotMatchingException.java	확인용 문자열이 동일하지 않을 시 예외 처리
	FillOutInformationException.java	모든 정보를 입력하지 않았을 시 예외 처리
	NotAvailableException.java	불가능한 요청 시 예외 처리
	NotExistingException.java	회원 정보가 존재하지 않을 시 예외 처리
	NotLoginException.java	로그인 실패 처리
	NotMatchingException.java	세션 정보와 동일하지 않은 계정에 대한 처리 요청 시 예외 처리

○ Index JSP with Introduce JSP

구분	기능	설명
Index JSP	index.jsp	로그인을 하지 않은 페이지
	member_index.jsp	로그인을 하고 난 후의 페이지
	admin_index.jsp	관리자용 페이지
	library_introduce.jsp	도서 관리 프로젝트를 소개하는 페이지

○ Admin JSP

기능		설명
Admin JSP	admin_alarm_board_update.jsp	게시판의 게시글의 공개 여부를 설정해주는 페이지
	admin_alarm_good_add.jsp	추천 도서 게시글을 등록하는 페이지
	admin_alarm_good_delete.jsp	추천 도서 게시글을 삭제하는 페이지
	admin_alarm_notice_add.jsp	공지사항 게시글을 등록하는 페이지
	admin_alarm_notice_delete.jsp	공지사항 게시글을 삭제하는 페이지
	admin_book_add.jsp	도서를 등록하는 페이지
	admin_book_delete.jsp	등록된 도서를 삭제하는 페이지
	admin_book_overdue.jsp	연체된 도서 목록을 확인하는 페이지
	admin_book_update.jsp	도서의 정보를 수정해주는 페이지
	admin_member_black_show.jsp	블랙리스트 회원의 랭크를 수정해주는 페이지
	admin_member_hope.jsp	회원 사용자가 신청한 희망 도서 목록을 확인하는 페이지
	admin_member_show.jsp	회원 가입한 회원들의 목록을 확인하는 페이지



○ Member JSP

기능		설명
Member JSP	member_register.jsp	회원가입 페이지
	member_login.jsp	로그인 페이지
	member_my_page.jsp	회원 전용 마이 페이지
	member_hope.jsp	희망 도서 신청 페이지
	member_forgotPwd.jsp	비밀 번호 초기화 요청 페이지
	member_change_password.jsp	회원 정보 수정 페이지

○ 공통 JSP

기능		설명
공통 JSP	new_book_unified_search.jsp	신작 도서 10권만 출력해주는 페이지
	hit_book_unified_search.jsp	대여가 5회 이상 되어야 입성할 수 있는 인기 도서 페이지
	notice_unified_search.jsp	공지사항 목록과 검색을 진행시킬 수 있는 페이지
	notice_detail.jsp	공지사항의 상세 내용을 출력해주는 페이지
	good_unified_search.jsp	추천 도서 목록과 검색을 진행시킬 수 있는 페이지
	good_detail.jsp	추천 도서의 상세 내용을 출력해주는 페이지
	book_unified_search.jsp	전체 도서 목록과 검색을 진행시킬 수 있는 페이지
	book_detail.jsp	도서의 상세 내용을 출력해주는 페이지
	board_unified_search.jsp	자유 게시판 목록과 검색을 진행시킬 수 있는 페이지
	board_detail.jsp	자유 게시판 속 게시글의 상세 내용을 출력해주는 페이지
	board_write.jsp	자유 게시판에 글을 작성할 수 있도록 지원해주는 페이지

○ CCS & JS

구분	기능	설명
CSS	Style.css	테이블이 존재하지 않는 일반 페이지들의 페이지 디자인
	Style2.css	테이블이 존재하는 페이지들의 페이지 디자인
	Style3.css	도서관 소개 페이지만의 디자인
JS	dataTables.js	각 페이지에 목록 테이블을 구현해주는 기능을 하는 파일
	scripts.js	글자수를 제한하는 기능을 하는 파일

○ MySQL

구분	기능	설명
DAO	book	도서 테이블
	member	회원 테이블
	notice	공지사항 테이블
	checkout	대여 테이블 (book 과 member PK를 FK)
	good	추천 도서 테이블 (book PK를 FK)
	board	자유 게시판 테이블 (member PK를 FK)
	comment	댓글 테이블 (board 와 member PK를 FK)
	request	희망 도서 테이블

### 3. 특이 기능에 대한 코드 설명

코드 이미지	설명
<pre>public void sendEmail(MemberDTO memberDTO, String div) throws Exception {     // Mail Server 설정     String charset = "utf-8";     String hostSMTP = "smtp.gmail.com"; // 지메일 이용시 smtp.gmail.com     String hostSMTPid = ""; // "서버 이메일 주소(보내는 사람 이메일 주소)";     String hostSMTPpwd = ""; // "서버 이메일 비번(보내는 사람 이메일 비번)";      // 보내는 사람 EMail, 제목, 내용     String fromEmail = "admin@admin"; // "보내는 사람 이메일주소(받는 사람 이메일에 표시됨)";     String fromName = "MinGW"; // "프로젝트이름 또는 보내는 사람 이름";     String subject = "";     String msg = "";      if (div.equals("forgotPwd")) {         subject = "MinGW's Library 임시 비밀번호입니다.";         msg += "&lt;div align='center' style='border:1px solid black; font-family:verdana'&gt;";         msg += "&lt;h3 style='color: blue;'&gt;";         msg += memberDTO.getMemberName() + "님의 임시 비밀번호입니다. 비밀번호를 변경하여 사용하세요.&lt;/h3&gt;";         msg += "&lt;p&gt;임시 비밀번호 : ";         msg += memberDTO.getMemberPassword() + "&lt;/p&gt;&lt;/div&gt;";     }      // 받는 사람 E-Mail 주소     String mail = memberDTO.getMemberEmail();     try {         HtmlEmail email = new HtmlEmail();         email.setDebug(true);         email.setCharset(charset);         email.setSSL(true);         email.setHostName(hostSMTP);         email.setSmtpport(465); // 지메일 이용시 587          email.setAuthentication(hostSMTPid, hostSMTPpwd);         email.setTLS(true);         email.addTo(mail, charset);         email.setFrom(fromEmail, fromName, charset);         email.setSubject(subject);         email.setHtmlMsg(msg);         email.send();     } catch (Exception e) {         System.out.println("메일발송 실패 : " + e);     } }</pre>	<p>MemberService 에 구현되어 있으며 SMTP 를 사용해서 임시 비밀번호를 생성해 입력한 이메일로 발급해주는 기능을 구현했습니다.</p> <p>메일 서버를 Gmail 로 설정해주고 Gmail 아이디와 비밀번호를 입력합니다.</p> <p>subject 는 메일의 제목, msg 는 메일의 내용을 입력하는 문자열입니다.</p> <p>msg 와 subject 에 들어갈 내용을 입력한 후에 HtmlEmail 을 사용해서 인코딩 방식을 UTF-8 로 설정, 전송 계층 보안 설정 (setSSL, setTLS) 을 true 로 세팅해줍니다.</p> <p>해당 계정의 보안 수준을 최저로 낮춘 후에 실행해야 메일이 보내집니다.</p>

### 3. 특이 기능에 대한 코드 설명

코드 이미지	설명
<pre>&lt;table id="datatablesSimple"&gt;   &lt;thead&gt;     &lt;tr&gt;       &lt;th&gt;ISBN10&lt;/th&gt;       &lt;th&gt;제목&lt;/th&gt;       &lt;th&gt;저자&lt;/th&gt;       &lt;th&gt;출판사&lt;/th&gt;       &lt;th&gt;장르&lt;/th&gt;       &lt;th&gt;해당 도서 페이지&lt;/th&gt;     &lt;/tr&gt;   &lt;/thead&gt;   &lt;!-- &lt;tfoot&gt;     &lt;tr&gt;       &lt;th&gt;ISBN10&lt;/th&gt;       &lt;th&gt;제목&lt;/th&gt;       &lt;th&gt;저자&lt;/th&gt;       &lt;th&gt;출판사&lt;/th&gt;       &lt;th&gt;장르&lt;/th&gt;     &lt;/tr&gt;   &lt;/tfoot&gt; --&gt;   &lt;tbody&gt;     &lt;c:forEach var="bookDTO" items="\${bookList}"&gt;       &lt;tr&gt;         &lt;td&gt;\${bookDTO.bookISBN}&lt;/td&gt;         &lt;td&gt;\${bookDTO.bookTitle}&lt;/td&gt;         &lt;td&gt;\${bookDTO.bookAuthor}&lt;/td&gt;         &lt;td&gt;\${bookDTO.bookPublisher}&lt;/td&gt;         &lt;td&gt;\${bookDTO.bookGenre}&lt;/td&gt;         &lt;td&gt;&lt;input type="button" value="자세히" onclick="Location.href='/book/book_detail?bookISBN=\${bookDTO.bookISBN}&amp;bookGenre=\${bookDTO.bookGenre}'"/&gt;&lt;/td&gt;       &lt;/tr&gt;     &lt;/c:forEach&gt;   &lt;/tbody&gt; &lt;/table&gt;</pre>	<p>종속성이 없는 자바스크립트 HTML 테이블 플러그인 simpleDatatables을 이용하여 쉽게 검색 및 정렬이 가능한 테이블을 생성합니다.</p> <p>이후 c:forEach 태그를 이용하여 원하는 도서 구조체의 속성과 속성에 들어있는 값을 구해옵니다.</p> <p>이러한 구조로 도서 테이블 뿐만 아니라 현재 프로젝트에서 사용된 목록 테이블들을 구현하였습니다.</p>

### III. 프로젝트 수행내용

## 1. 프로젝트 수행일정

[illegible]

## VI. 팀원 별 역할 표 및 기타 개발에서의 노력

### 1. 팀원 별 역할 표

구분	기능	설명
src/main/java	libraryManage DAO	정민지
	libraryManage DTO	김민지, 정민지
	libraryManage SERVICE	정민지
	libraryManage CONTROLLER	김민지, 정민지
	libraryManage EXECPTION	정민지
src/main/resources	STATIC	김민지
src/main/webapp	WEB-INF VIEWS	김민지

### 2. 동영상 제작

- 김민지 : 동영상 데모 플레이 촬영, 동영상 자막 제공
- 정민지 : 동영상 편집, 발표

### 3. 기타 개발에서의 노력

- 2인 1팀으로써 프로젝트의 형상 관리가 가장 중요한 문제였습니다.  
따라서 GitHub 계정을 이용해 프로젝트 Repository를 생성하여 김민지 팀원이 Fork 후 Dev Branch 로 commit merge를 하는 방식을 이용해 서로 서로 각자의 **코드**의 **변경점만을 업데이트하는 방식으로 프로젝트를 공동 진행**하였습니다.
- README.md 파일의 **목차를 만들어** 구현해야할 기능, Database Table 구조, PUSH 목록 등을 업데이트하며 진행하였습니다.

이를 통해 각자 어떤 기능을 우선하여 구현해야 하는지를 알 수 있었습니다.

- 2021-06-07 현재 총 **335번의 commit**을 하여 완성 하였습니다.

서로 커밋한 소스 파일들이 기록되어 있는 깃허브 주소:

<http://www.github.com/17mirinae/LibraryManage>

<http://www.github.com/jee00609/LibraryManage>

May 7, 2021 – June 7, 2021

Period: 1 month ▾


#### Overview


60 Active Pull Requests

0 Active Issues

 60  
Merged Pull Requests

 0  
Open Pull Requests

 0  
Closed Issues

 0  
New Issues

Excluding merges, **2 authors** have pushed **266 commits** to master and **266 commits** to all branches. On master, **0 files** have changed and there have been **0 additions** and **0 deletions**.



 60 Pull requests merged by 1 person