

하이미디어아카데미

도서 대여 관리 프로그램

TEAM KIM 조

김민하, 김준혁, 김상희

목차

- 01. 프로젝트 개요
- 02. 프로젝트 팀 구성 및 역할
- 03. 프로젝트 수행 절차 및 방법
- 04. 프로젝트 수행 경과
- 05. 자체 평가 의견

프로젝트 개요

▶ 프로젝트 주제 및 선정 배경, 기획의도



프로젝트 주제

자바(Java)와 MySQL을 활용한 도서 대여 관리 시스템 개발



선정 배경

- 도서 대출 및 반납 내역을 체계적으로 관리할 수 있는 프로그램이 필요함
- Java와 MySQL을 연동함으로써 데이터베이스를 활용한 실용적인 도서 관리 시스템을 구현하고자 함

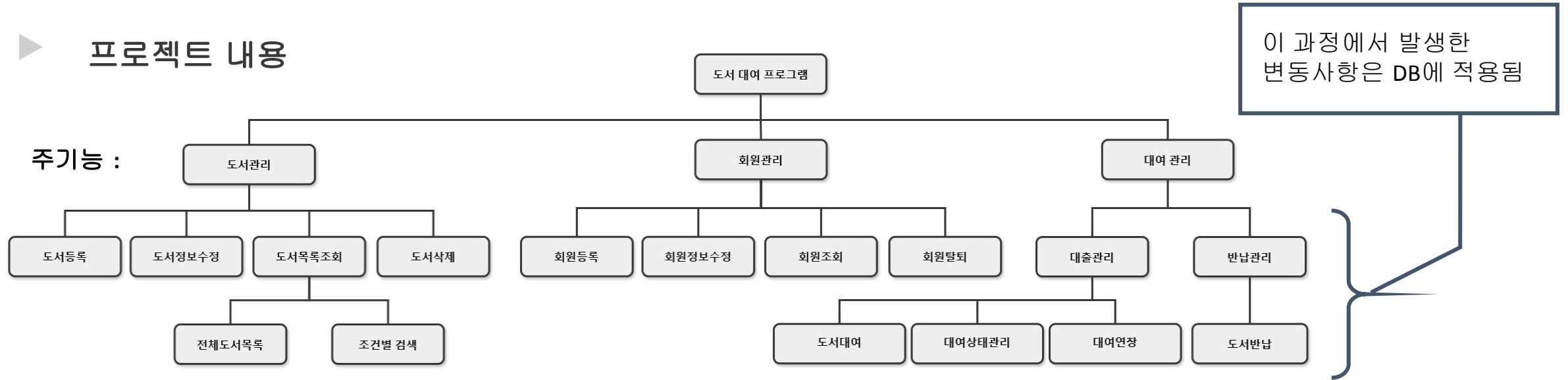


기획의도

- 관리자로서 수행할 수 있는 기능을 추가하여 관리자로 접근 시에만 회원정보 내 연체여부를 수정할 수 있다거나 전체 대여목록을 볼 수 있는 등의 추가작업이 가능한 시스템을 만들고자 함
- 즉, 관리자와 사용자가 복합적으로 이용가능한 시스템을 구축하고자 함

프로젝트 개요

▶ 프로젝트 내용



▶ 활용 장비 및 재료



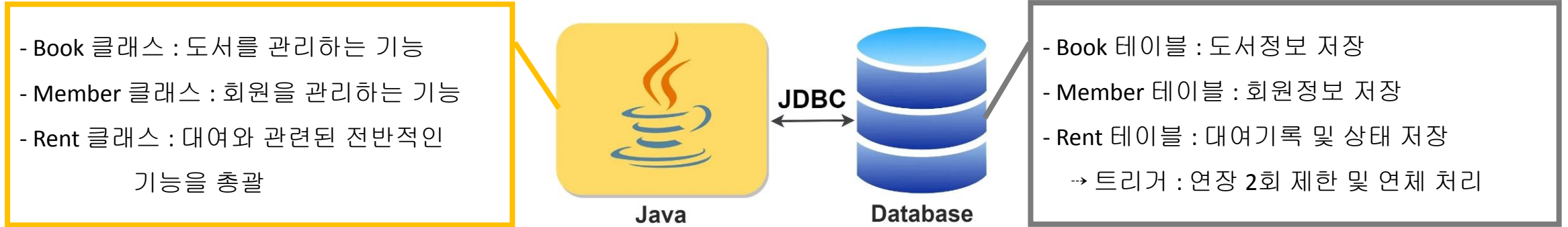
JDBC



개발용 PC

프로젝트 개요

▶ 프로젝트 구조



▶ 활용 방안 및 기대 효과

▶ 활용 방안

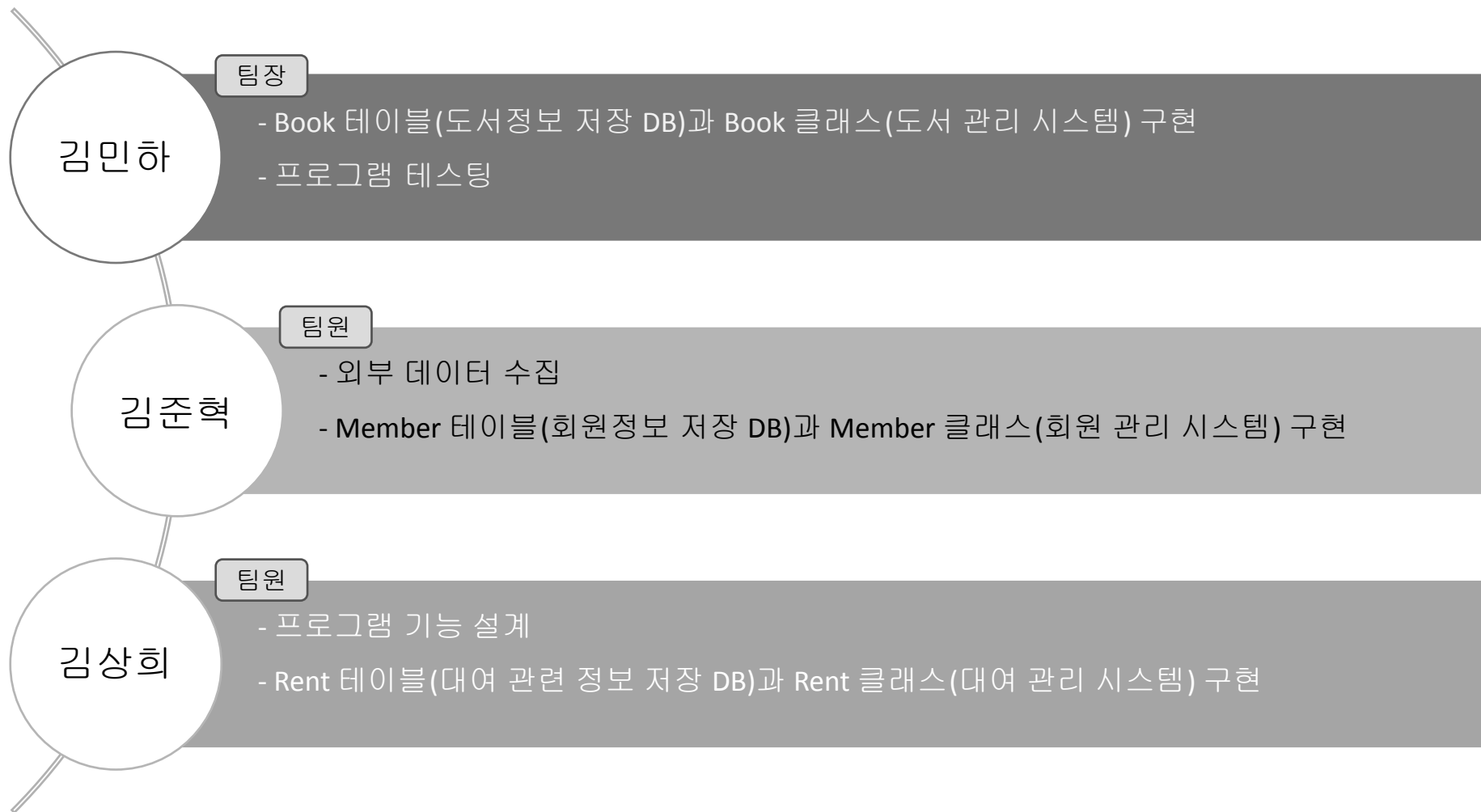
- 학교, 도서관, 기업 내 사내 도서관 등에서 활용 가능
- 소규모 도서 관리가 필요한 기관에서 유용하게 사용 가능

▶ 기대 효과

- 도서 대출 및 반납 절차의 자동화로 업무 효율 증가
- 데이터베이스를 활용한 체계적인 관리

프로젝트 팀 구성 및 역할

Team KIM



프로젝트 수행 절차 및 방법

▶ 프로젝트 기간별 활동 사항

구분	기간	활동	비고
기획 및 설계	▶ 2/12(수) ~ 2/13(목)	<ul style="list-style-type: none">▶ 각 팀원의 역할 분담 및 개발 방향 설정 (사용자를 누구에게 초점을 맞출 것인지)▶ 필요한 기능 정의 및 활용할 데이터 조사	
ERD 구현	▶ 2/13(목)	<ul style="list-style-type: none">▶ 프로그램 제작에 필요한 데이터 수집▶ 수집된 데이터 활용을 위한 테이블 구조 정의 및 생성▶ 협업을 위한 github 연동	▶ 공공데이터셋 포털 무료 제공 데이터 활용
코딩	▶ 2/14(금) ~ 2/16(일)	<ul style="list-style-type: none">▶ 기능 세분화 및 구현	
테스트	▶ 2/17(월) ~ 2/18(화)	<ul style="list-style-type: none">▶ 구현된 기능 취합 및 코드 통일화▶ 실행하며 오류 수정 및 시스템 작동 안정화	
총 개발기간	▶ 2/13(목) ~ 2/18(화)	-	-

프로젝트 수행 경과

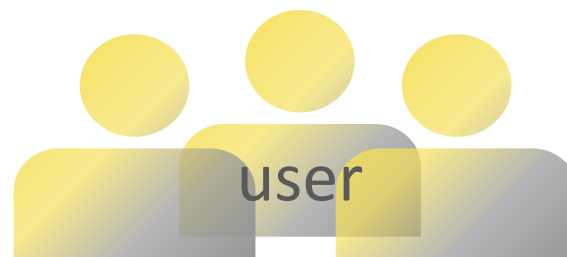
결과 제시 ① 기획 및 설계

기능정의서

Aa 구분	≡ 주 기능	≡ 상세기능	≡ 설명
1. 회원 관리	1.1 회원 등록		- 이름, 아이디, 비밀번호, 개인정보 등을 입력하여 회원을 등록할 수 있는 기능
	1.2 회원 정보 수정		- 회원의 개인정보를 수정할 수 있는 기능 - 관리자(admin)의 경우, 연체 여부도 수정 가능
	1.3 회원 조회		- 모든 회원 정보를 확인할 수 있는 기능
	1.4 회원 삭제		
2. 도서 관리	2.1 도서 등록		- 도서 정보(도서 제목, 저자, 출판사, ISBN, 수량 등)를 입력하여 도서를 등록하는 기능
	2.2 도서 정보 수정		- 도서의 제목, 저자, 출판사 등을 수정하는 기능
	2.3 도서 목록	2.3.1 전체 도서 목록	
		2.3.2 조건별 검색	- 제목, 저자, 출판사 별로 검색 가능
	2.4 도서 삭제		
3. 대여 관리	3.1 대출 관리	3.1.1 도서 대출	- 회원이 도서를 대출할 수 있는 기능 - 대출 날짜와 반납 날짜를 기록
		3.1.2 대출 상태 관리	- 회원: 자신이 대출 중인 도서 목록을 확인 - 관리자(admin): 전체 대출목록 확인 및 도서 상태별(도서번호, 반납완료도서, 미반납도서, 연체도서 등) 목록 확인 가능
		3.1.3 대출 연장	- 일주일씩 2회까지 연장 가능
	3.2 반납 관리	3.2.1 도서 반납	- 연체 여부를 확인 후 대출한 도서를 반납하고, 반납 일자를 기록하는 기능

개발 방향


(사용자를 누구에게 초점을 맞출 것인지)



프로젝트 수행 경과

결과 제시 ② 데이터 수집 및
구조화

▶ 학습 데이터 소개 (Book)

 제주국제자유도시개발센터_JDC문화공간 낭 도서 구매목록(어린이 등)_20181231

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	연번	도서명	저자 1	저자 2	ISBN	발행처(출판사)	발행연도	종수	책수	단가	금액	건축금액	비고
2	1	숨기 대장 카멜레온	문주영		979-11-253-0550-7(50)	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
3	2	주주 할머니의 주스 가게	정혜란		979-11-253-0552-1	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
4	3	큰 건 내거야	정순		979-11-253-0713-6	NEKIDS	2016	1	1	9500	9500	8550	상상수프통화(수학)
5	4	하양이의 나들이	신지영		979-11-253-0786-0	NEKIDS	2015	1	1	10000	10000	9000	상상수프통화(수학)
6	5	같은 것끼리 짝짝!	김자영		979-11-253-0818-8	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
7	6	씩씩씩둑 색종이 놀이	김정란		979-11-253-0835-5(6)	NEKIDS	2015	1	1	10000	10000	9000	상상수프통화(수학)
8	7	꾸리의 방귀 재주	신지영		979-11-253-0887-4	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
9	8	밤하늘에 톡톡톡	서동선		979-11-253-0951-2	NEKIDS	2015	1	1	10000	10000	9000	상상수프통화(수학)
10	9	나나와 똥지	조미희		979-11-253-0977-2	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
11	10	도도 공주의 생일 케이크	조미희		979-11-253-1067-9(10)	NEKIDS	2015	1	1	10000	10000	9000	상상수프통화(수학)
12	11	달나무 열매가 먹고 싶어!	조미희		979-11-253-1071-6	NEKIDS	2016	1	1	9500	9500	8550	상상수프통화(수학)
13	12	보름이의 서울여행	정순		979-11-253-1075-4	NEKIDS	2016	1	1	9500	9500	8550	상상수프통화(수학)
14	13	수박씨를 땀!	이지윤		979-11-253-0460-9	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
15	14	주머니 괴물의 선물	서보현		979-11-253-0513-2	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
16	15	잡을까? 말까?	이보미, 유하		979-11-253-0714-3	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
17	16	맛있는 게 없을까?	김정란		979-11-253-0787-7	NEKIDS	2015	1	1	10000	10000	9000	상상수프통화(수학)
18	17	도깨비는 뭘을 좋아해!	이선아		979-11-253-0819-5	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
19	18	훈이의 모험	이지윤		979-11-253-0836-2	NEKIDS	2015	1	1	10000	10000	9000	상상수프통화(수학)
20	19	아빠만 믿어	한정민		979-11-253-0888-1	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)
21	20	할아버지의 비밀 상자	이보미		979-11-253-0952-9	NEKIDS	2015	1	1	10000	10000	9000	상상수프통화(수학)
22	21	할아버지 절뚝을 적어	이선아		979-11-253-0978-9	NEKIDS	2015	1	1	9500	9500	8550	상상수프통화(수학)

1039개의 데이터 리스트

전처리

```
# book 테이블 데이터셋
select * from book;
commit;
INSERT INTO book(title,author,isbn,publisher,issue_year,total_qty,book_created,book_updated)
VALUES ('출둥! 슈퍼왕스 세계 국기놀이','권집부','8.82E+12','아이즐북스','2015',1,'2025-02-13 13:16:00','2025-02-13 13:16:00'),
('구석구석 세계 그림 지도','샘 레이크','8.83E+12','어스본코리아','2016',1,'2025-02-13 13:16:33','2025-02-13 13:16:33'),
('나무집 시리즈(전6권)(세트)','앤디 그리피스','978-15-09864-50-8','시공주니어','2017',6,'2025-02-13 13:20:08','2025-02-13 13:20:08'),
('뉴 곰돌곰돌 자연관찰(전84권)','권집부','978-15-09864-50-9','블루스미','2014',84,'2025-02-13 13:15:11','2025-02-13 13:15:11'),
```

SQL 명령어로 변환하여
데이터 삽입

프로젝트 수행 경과

결과 제시 ② 데이터 수집 및
구조화

▶ 학습 데이터 소개 (Member)

▶ 임의로 설정한 10인의 인물데이터셋 삽입

초기 member 테이블 데이터셋(10개)

```
INSERT INTO member (name, user_id, password, birthday, gender, mobile, overdue)
VALUES
```

```
('John', 'user1', 'password1423', '19630514', 'M', '010-1234-5678', 'X'),
('Jane', 'user2', 'password3820', '19750220', 'F', '010-2345-6789', 'X'),
('Alice', 'user3', 'password4921', '19830315', 'F', '010-3456-7890', 'X'),
('Bob', 'user4', 'password2093', '19890910', 'M', '010-4567-8901', 'X'),
('Charlie', 'user5', 'password8374', '19900725', 'M', '010-5678-9012', 'X'),
('David', 'user6', 'password3047', '19950218', 'M', '010-6789-0123', 'X'),
('Eve', 'user7', 'password9754', '19991103', 'F', '010-7890-1234', 'X'),
('Frank', 'user8', 'password8745', '20050721', 'M', '010-8901-2345', 'X'),
('Grace', 'user9', 'password5620', '20100310', 'F', '010-9012-3456', 'X'),
('Hannah', 'user10', 'password6743', '20151225', 'F', '010-0123-4567', 'X');
```

프로젝트 수행 경과

결과 제시 ② 데이터 수집 및 구조화

데이터 테이블 구조

▶ book 테이블

```
create table book(  
  book_id int unsigned auto_increment primary key, # => join 시 활용  
  title varchar(128) not null, # 책 제목  
  author varchar(128) not null, # 작가  
  isbn varchar(24) not null, # isbn코드  
  publisher varchar(32), # 출판사  
  issue_year char(10) not null, # 출간년도  
  total_qty int unsigned default 1, # 총 수량  
  qty int unsigned default 0, # 빌린 수량  
  book_created datetime default current_timestamp, # 등록날짜  
  book_updated datetime default current_timestamp on update current_timestamp # 수정날짜  
);
```

데이터 정보를 저장할
테이블 생성

Field	Type	Null	Key	Default	Extra
book_id	int unsigned	NO	PRI	NULL	auto_increment
title	varchar(128)	NO		NULL	
author	varchar(128)	NO		NULL	
isbn	varchar(24)	NO		NULL	
publisher	varchar(32)	YES		NULL	
issue_year	char(10)	NO		NULL	
total_qty	int unsigned	YES		1	
qty	int unsigned	YES		0	
book_created	datetime	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED
book_updated	datetime	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED on update CURRENT_TI...

▶ member 테이블

```
create table member(  
  name varchar(32) not null, # 이름  
  user_id varchar(32) not null primary key, # 아이디  
  password varchar(24) not null, # 비밀번호  
  birthday char(8) not null, # 생년월일  
  gender char(1), # 성별 - M/F  
  mobile varchar(16) not null, # 전화번호  
  overdue char(1), # 연체 - 0/1  
  member_created datetime default current_timestamp, # 회원등록날짜  
  member_updated datetime default current_timestamp on update current_timestamp # 회원정보 수정날짜  
);
```

테이블 생성

Field	Type	Null	Key	Default	Extra
name	varchar(32)	NO		NULL	
user_id	varchar(32)	NO	PRI	NULL	
password	varchar(24)	NO		NULL	
birthday	char(8)	NO		NULL	
gender	char(1)	YES		NULL	
mobile	varchar(16)	NO		NULL	
overdue	char(1)	YES		NULL	
member_created	datetime	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED
member_updated	datetime	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED on update CURRENT_TI...

프로젝트 수행 경과

결과 제시 ② 데이터 수집 및
구조화

데이터 테이블 구조

▶ rent 테이블

```
create table rent (  
  rent_id int unsigned auto_increment primary key,  
  book_id int unsigned not null, foreign key(book_id) references book(book_id),  
  member_id varchar(32) not null, foreign key(member_id) references member(user_id),  
  rent datetime DEFAULT CURRENT_TIMESTAMP,  
  returned datetime,  
  expected datetime DEFAULT (CURRENT_TIMESTAMP + INTERVAL 2 WEEK),  
  extension tinyint(1) default false,  
  extension_count int default 0,  
  overdue boolean default false,  
  overdays int default 0  
);
```

book 테이블의 id
member 테이블의 user_id
빌린 날짜
실제 반납 날짜
반납 예정일 -> rent + 14days
연장 - 0/1 -> expected + 7days
연장횟수
연체여부 - 0/1
연체일수

데이터 정보를 저장할
테이블 생성

	Field	Type	Null	Key	Default	Extra
▶	rent_id	int unsigned	NO	PRI	NULL	auto_increment
	book_id	int unsigned	NO	MUL	NULL	
	member_id	varchar(32)	NO	MUL	NULL	
	rent	datetime	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED
	returned	datetime	YES		NULL	
	expected	datetime	YES		(now() + interval 2 week)	DEFAULT_GENERATED
	extension	tinyint(1)	YES		0	
	extension_count	int	YES		0	
	overdue	tinyint(1)	YES		0	
	overdays	int	YES		0	

▶ rent 테이블 – 트리거(Trigger)

데이터베이스에서 특정 조건에서 자동으로 실행되는 SQL 코드로,
특정 작업을 수행하도록 설정하는 데 사용됨.

```
-- 연장 2회 이상, 반납 선택, 연체 처리 포함  
DELIMITER $$  
CREATE TRIGGER before_return_update  
BEFORE UPDATE ON rent  
FOR EACH ROW  
BEGIN  
  IF OLD.extension_count >= 2 THEN  
    -- 연장이 2번 이상일 때, 반납만 가능  
    IF NEW.returned IS NOT NULL THEN  
      SET NEW.returned = current_timestamp;  
      -- 반납이 늦었을 경우 연체 처리  
      IF NEW.expected < NEW.returned THEN  
        SET NEW.overdue = TRUE;  
        SET NEW.overdays = DATEDIFF(NOW(), NEW.expected);  
      ELSE  
        SET NEW.overdue = FALSE;  
        SET NEW.overdays = 0;  
      END IF;  
    ELSE  
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = '연장이 2번 완료되어 더 이상 연장이 불가능합니다.';  
    END IF;  
  ELSE  
    -- 연장이 2번 미만일 때 연장 가능  
    IF NEW.extension = 1 THEN  
      SET NEW.expected = DATE_ADD(OLD.expected, INTERVAL 1 WEEK);  
      SET NEW.extension_count = OLD.extension_count + 1;  
    END IF;  
  END IF;  
END $$  
DELIMITER ;
```

→ 연장 횟수에 따라 반납과 연장 가능 여부를 결정하고,
연체 처리를 관리하는 역할

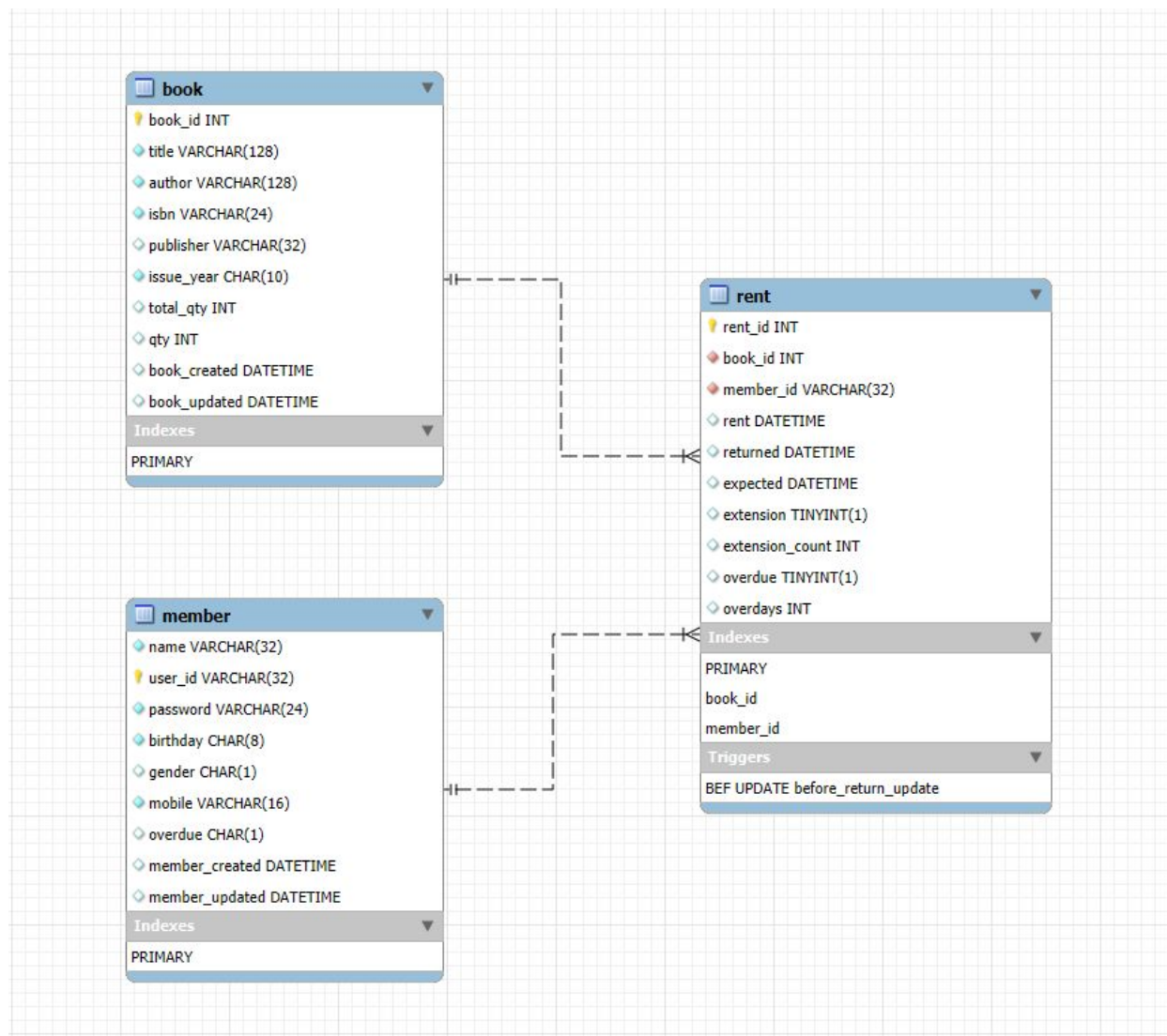
프로젝트 수행 경과

결과 제시 ② 데이터 수집 및
구조화

데이터 테이블 구조

▶ ER 다이어그램

: 'Entity 개체'와 'Relationship 관계'를 중점적으로
표시하는 데이터베이스 구조를 한 눈에 알아보기
위해 그려놓는 다이어그램



프로젝트 수행 경과

결과 제시 ③ 기능 세분화 및
구현



시스템 구현 알고리즘

< Book클래스 : 도서관리 시스템 >

Control() : 각 기능을 총괄하는 메소드.
도서 관리 작업을 선택할 수 있는 메뉴를 제공.

Add()	Updated()	Read()	Delete()
<div>1. 사용자가 도서 정보를 입력합니다 (제목, 저자, 출판사, 출간년도, ISBN, 보유수량).</div> <div>2. 입력된 정보가 유효한지 체크합니다.</div> <div>3. 모든 값이 유효하면, INSERT INTO SQL 문을 사용하여 데이터를 데이터베이스에 저장합니다.</div> <div>4. DB 연결 및 오류 처리 후, 성공적인 등록을 출력합니다.</div>	<div>1. 사용자가 수정할 도서 제목을 입력합니다.</div> <div>2. 수정할 항목을 선택합니다 (제목, 저자, 출판사, 출간년도, ISBN, 수량).</div> <div>3.수정할 항목에 따라 새로운 값을 입력받고, 해당 값으로 UPDATE SQL 문을 실행합니다.</div> <div>4.수정이 성공적으로 이루어지면 수정된 결과를 출력합니다. 실패할 경우 오류 메시지를 출력합니다.</div>	<div>1. 사용자가 검색할 항목(제목, 저자, 출판사)을 선택합니다.</div> <div>2. 해당 항목에 대한 값을 입력받습니다.</div> <div>3. 입력된 값으로 SELECT SQL 문을 실행하여 검색된 결과를 ResultSet으로 가져옵니다.</div> <div>4. 결과가 있으면 도서 정보를 출력하고, 없다면 “해당 도서를 찾을 수 없습니다”라는 메시지를 출력합니다.</div>	<div>1.사용자가 삭제할 도서 제목을 입력합니다.</div> <div>2.해당 제목을 DELETE FROM SQL 문을 사용하여 삭제합니다.</div> <div>3.삭제가 성공하면 “도서 삭제 완료” 메시지를 출력합니다. 도서가 없으면 실패 메시지를 출력합니다.</div>
		<div>readAll()</div> <div>1. DB에서 모든 도서를 SELECT SQL 문으로 가져옵니다.</div> <div>2. 가져온 데이터를 ResultSet을 사용하여 출력합니다. 도서 정보 (제목, 저자, 출판사 등)를 출력하여 사용자에게 보여줍니다.</div>	

프로젝트 수행 경과

결과 제시 ③ 기능 세분화 및
구현

▶ 시스템 구현 알고리즘

< Member클래스 : 회원관리 시스템 >

Control() : 각 기능을 총괄하는 메소드.
회원 관리 작업을 선택할 수 있는 메뉴를 제공.

registerMember()

1. 사용자가 입력한 회원 정보를 받습니다 (이름, 아이디, 비밀번호, 생년월일, 성별, 전화번호).
2. 입력 값에 대해 유효성 검사(길이 체크, 형식 체크 등)를 진행합니다.
3. 유효한 값이면, **INSERT INTO SQL** 문을 사용하여 회원 정보를 데이터베이스에 저장합니다.
4. 성공적인 회원가입 후, 완료 메시지를 출력합니다.

updateMember()

1. 사용자가 수정할 회원의 ID를 입력받습니다.
2. 관리자인 경우 다른 회원의 정보도 수정할 수 있습니다. 관리자가 아닌 경우 본인만 수정 가능합니다.
3. 수정할 항목(이름, 비밀번호, 전화번호 등)을 선택합니다.
4. 선택된 항목에 대해 새로운 값을 입력받고, **UPDATE SQL** 문을 사용하여 정보를 수정합니다.
5. 수정이 완료되면 결과 메시지를 출력합니다

researchMember()

1. 데이터베이스에서 모든 회원 정보를 조회하는 **SELECT SQL** 문을 실행합니다.
2. 결과를 **ResultSet**으로 받아와 회원 목록을 출력합니다

deleteMember()

1. 탈퇴할 회원의 ID를 입력받습니다.
2. 해당 회원을 **DELETE FROM SQL** 문을 사용하여 삭제합니다.
3. 성공적으로 삭제되면 “회원 탈퇴 완료” 메시지를 출력합니다. 존재하지 않는 회원이라면 실패 메시지를 출력합니다.

프로젝트 수행 경과

결과 제시 ③ 기능 세분화 및
구현

▶ 시스템 구현 알고리즘

< Rent클래스 : 대여-반납 관리 시스템 >

Control() : 각 기능을 총괄하는 메소드
대여 관리 작업을 선택할 수 있는 메뉴를 제공.

Add()	Updated()	Read()	extend()	returned ()
<p>1. 사용자가 대여하고자 하는 책의 ID를 입력받고, 해당 책의 재고가 충분한지 확인합니다.</p> <p>2. 재고가 부족하면 대여 불가능 메시지 출력, 그렇지 않으면 rent 테이블에 대여 기록을 추가합니다.</p> <p>3. 책의 재고는 book 테이블에서 업데이트되어야 하며, 대여가 완료되면 재고가 증가합니다.</p> <p>4. 대여 후, 사용자가 대여 내역을 확인할 수 있도록 read 메서드를 호출하여 대여 내역을 보여줍니다.</p>	<p>1. 사용자가 수정할 도서 제목을 입력합니다.</p> <p>2. 수정할 항목을 선택합니다 (제목, 저자, 출판사, 출간년도, ISBN, 수량).</p> <p>3. 수정할 항목에 따라 새로운 값을 입력받고, 해당 값으로 UPDATE SQL 문을 실행합니다.</p> <p>4. 수정이 성공적으로 이루어지면 수정된 결과를 출력합니다. 실패할 경우 오류 메시지를 출력합니다.</p>	<p>1. 사용자가 자신의 대여 현황을 조회할 수 있도록 해당 사용자의 대여 목록을 보여줍니다.</p> <p>2. 대여된 책의 제목, 대여일, 반납 예정일, 연장 여부, 연체 여부 등을 출력합니다.</p> <p>3. admin 계정이면 모든 회원의 대여 현황을 조회할 수 있습니다.</p> <div>readAll()</div> <p>1. 관리자만 사용할 수 있는 기능으로, 전체 대여 현황을 관리할 수 있는 메뉴를 제공합니다.</p> <p>2. 관리자는 전체 대여 현황, 도서별 대여 현황, 반납 완료 도서 목록, 미반납 도서 목록, 연체 도서 목록 등을 확인할 수 있습니다.</p> <p>3. 각 작업에 맞는 SQL 쿼리로 해당 데이터를 조회하여 출력합니다.</p>	<p>1. 사용자가 대여한 책을 연장할 수 있는 기능.</p> <p>2. 연장이 가능한 책들을 출력하고, 사용자가 연장하고자 하는 대여 번호를 입력받습니다.</p> <p>3. 대여 번호에 해당하는 도서의 연장 여부를 업데이트하고, 연장된 대여의 반납 예정일을 1주일 연장합니다.</p>	<p>1. 사용자가 대여한 도서를 반납할 수 있는 기능.</p> <p>2. 반납되지 않은 대여 목록을 출력하고, 사용자가 반납할 대여 번호를 입력받습니다.</p> <p>3. 반납된 도서에 대해서는 rent 테이블의 returned 컬럼을 현재 시간으로 업데이트합니다.</p> <p>4. 책의 재고는 book 테이블에서 증가시켜 반환된 도서 수량을 반영합니다.</p>

프로젝트 수행 경과

결과 제시 ③ 기능 세분화 및 구현

시스템 코드

▶ Rent클래스 – 연장 기능



트리거 작동

- 연장이 2번 이상일 때, 반납만 가능
- 2번 미만이면 연장가능

```
public void extend(String memId) {
    // SQL 데이터 읽어오기 > select 멤버아이디, 책 이름>from book, 대출일, 반납예정일, 연장여부, 연장횟수, 연체여부 반복문 + 반납일(returned) is null인 값만 show

    DBConnection.setConnection();
    try {
        Statement st = DBConnection.conn.createStatement();
        String sql = "select r.rent_id,r.member_id,b.title,r.rent,r.expected,IF(r.extension = 1, 'O', 'X') extension,"
            + "r.extension_count,IF(r.overdue = 1, 'O', 'X') overdue from rent r,book b "
            + "where r.member_id='"+memId+"'and r.book_id=b.book_id and r.returned is null";
        ResultSet rs = st.executeQuery(sql);

        while (rs.next())
        {
            System.out.println("대여 번호: "+rs.getInt("rent_id")+", 회원 ID: "+rs.getString("member_id")+", 책 제목: "+rs.getString("title")+
                ", 대여일: "+rs.getString("rent")+", 반납 예정일: "+rs.getString("expected")+", 연장 여부: "+rs.getString("extension")+
                ", 연장 횟수: "+rs.getInt("extension_count")+", 연체 정보: "+rs.getString("overdue"));
        }

        st.close(); rs.close();

        while(true) {
            System.out.println("> 연장하실 도서의 대여 번호를 입력해 주세요. 종료 희망 시:Enter"); // rent_id 입력 받고 연장(update)하는 sql문
            String sRentId = sc.nextLine();
            if(sRentId.equals("")) break;
            if(!Main.isNumber(sRentId)) break;
            int RentId = Integer.parseInt(sRentId);

            st = DBConnection.conn.createStatement();
            sql = "update rent set extension = true, extension_count = extension_count + 1, expected = date_add(expected, INTERVAL 1 WEEK) where rent_id ='"+RentId;
            st.executeUpdate(sql);
            String sql2 = "select expected from rent where rent_id ='"+RentId;
            ResultSet rs2 = st.executeQuery(sql2);

            while(rs2.next()) {
                System.out.println("> 연장이 완료되었습니다. 반납 기한: "+rs2.getString("expected"));
            }

            st.close(); rs2.close();
        }

    } catch (SQLException e) {
        System.out.println(e.getMessage());
    }
    DBConnection.disconnect();
}
```

프로젝트 수행 경과

결과 제시 ③ 기능 세분화 및
구현

시스템 코드

▶ Rent클래스 – 반납 기능

```
public void returned(String memId) {
    // SQL 데이터 읽어오기 > select 멤버아이디, 책 이름>from book, 대출일, 반납예정일, 연장여부, 연장횟수, 연체여부를 반납일(returned) is null인 값만 show
    DBConnection.setConnection();
    try {
        Statement st = DBConnection.conn.createStatement();
        String sql = "select r.rent_id,r.member_id,b.book_id,b.title,r.rent,r.expected,IF(r.extension = 1, 'O', 'X') extension,"
            + "r.extension_count,IF(r.overdue = 1, 'O', 'X') overdue from rent r,book b "
            + "where r.member_id='"+memId+"'and r.book_id=b.book_id and r.returned is null";
        ResultSet rs = st.executeQuery(sql);

        while (rs.next())
        {
            System.out.println("대여 번호: "+rs.getInt("rent_id")+", 회원 ID: "+rs.getString("member_id")+", 도서 번호: "+rs.getInt("book_id")+", 도서 제목: "+rs.getString("title")+
                ", 대여일: "+rs.getString("rent")+", 반납 예정일: "+rs.getString("expected")+", 연장 여부: "+rs.getString("extension")+
                ", 연장 횟수: "+rs.getInt("extension_count")+", 연체 정보: "+rs.getString("overdue"));
        }
        st.close(); rs.close();

        while(true) {
            // SQL rent> returned insert, overdue check(?)>penalty, Book qty update(외부로부터 반납되는 수량)
            System.out.println("> 반납하실 도서의 대여 번호를 입력해 주세요. 종료 희망 시:Enter"); // rent_id 입력 받고 연장(update)하는 sql문
            String sRentId = sc.nextLine();
            if(sRentId.equals("")) break;
            if(!Main.isNumber(sRentId)) break;
            int RentId = Integer.parseInt(sRentId);

            try {
                Statement st1 = DBConnection.conn.createStatement();
                String sql2 = "UPDATE rent SET returned = CURRENT_TIMESTAMP() WHERE rent_id = " + RentId + " AND returned IS NULL";
                st1.executeUpdate(sql2);
                st1.close();

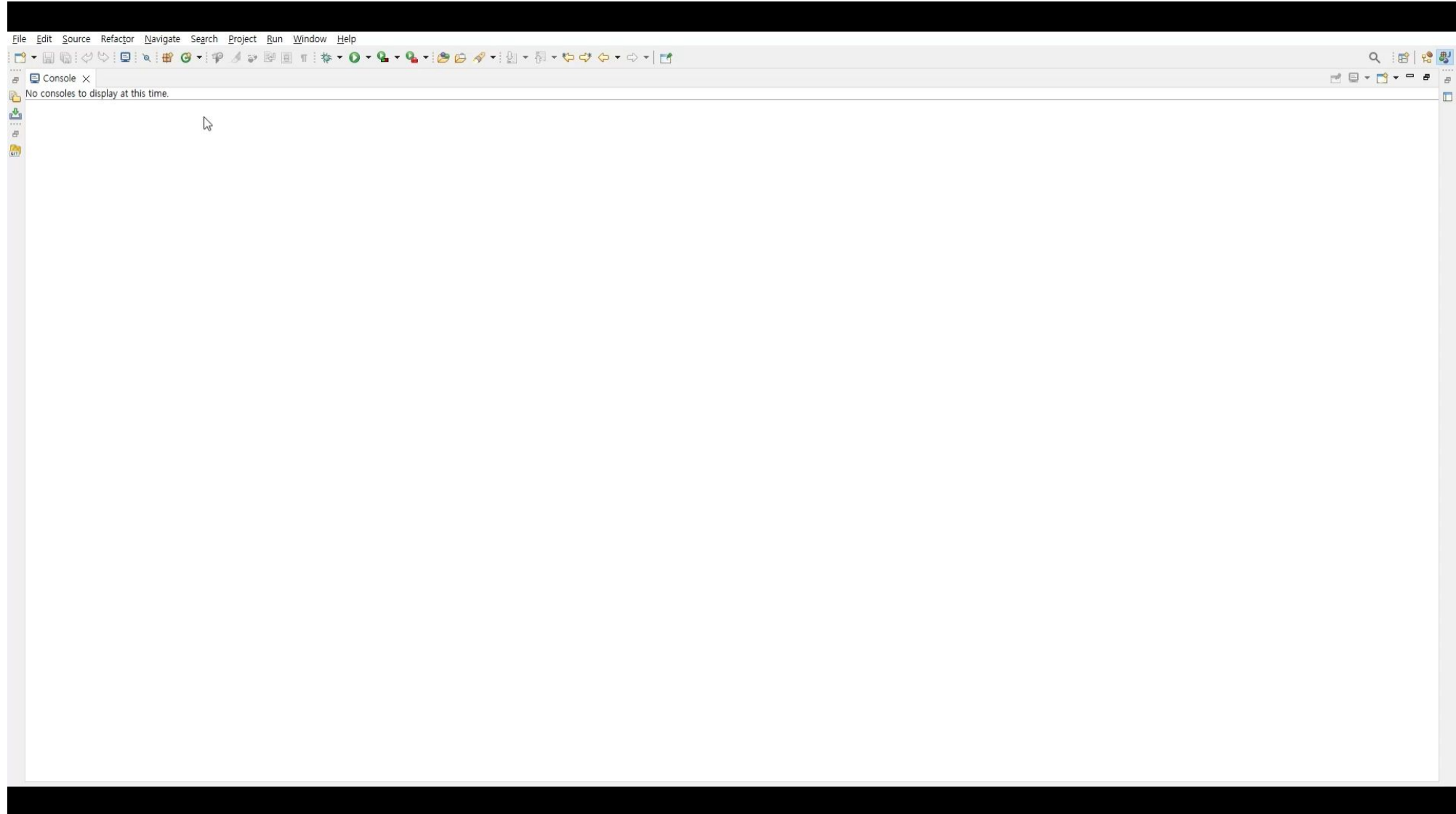
                Statement st2 = DBConnection.conn.createStatement();
                String sql3 = "update book b join rent r on r.book_id=b.book_id set b.qty = b.qty - 1 where r.rent_id = "+RentId;
                st2.executeUpdate(sql3);
                System.out.println("> 반납이 완료되었습니다.");
                st2.close();

            }catch(SQLException e) {
                System.out.println(e.getMessage());
            }
        }
    }catch(SQLException e) {
        System.out.println(e.getMessage());
    }
    DBConnection.disconnect();
}
```

프로젝트 수행 경과

※ 별도 첨부 가능

결과 제시 ④ 시연 동영상



자체 평가 의견

▶ 프로젝트 결과물에 대한 프로젝트 기획 의도와의 부합 정도 및 실무 활용 가능 정도, 달성도, 완성도 등 훈련생의 자체적인 평가 의견과 느낀 점을 작성한다.

- 사전 기획의 관점에서 **프로젝트 결과물에 대한 완성도 평가**(10점 만점)

⇒ 9점

- 개인 또는 우리 팀이 **잘한 부분과 아쉬운 점**

⇒ 협력하는데 있어 원활한 의사소통으로 좀 더 수월하게 프로젝트를 진행할 수 있었다. 다만, 시스템 구현 과정에서 기능을 정의할 때 좀 더 세분화하여 작업을 분담했다면, 구현 과정이 더 빠르게 수행되었을 것 같다는 아쉬움이 남음.

- 프로젝트 결과물의 **추후 개선점이나 보완할 점** 등 내용 정리

⇒ 연체 기능을 구현할 때 대어나 반납을 모두 현재시각을 저장하게 설정하여 해당 기능이 잘 수행되는지 명확하게 확인하지 못해 이 부분을 추후 확인 및 보완해야 함.

- 프로젝트를 수행하면서 **느낀 점이나 경험한 성과**(경력 계획 등과 연관)

⇒ 강의시간에 배운 내용들을 토대로 프로젝트를 진행하면서 강의 중 배운 것들을 직접 코드로 구현하고 응용하는 과정에서 더 많은 것을 알고 습득할 수 있게 됨.