게시판 CRUD 미니 프로젝트

2023. 05

MSA Full-Stack 개발자 교육 7차

하늘샘

목차

1.1 프로젝트 개요

이 프로젝트는 사용자들이 자유롭게 게시물을 등록, 수정, 삭제, 조회할 수 있는 기능을 제공하는 간단한 게시판 시스템을 개발하는 것을 목표로 합니다.
이 시스템은 Java를 기반으로 한 MVC (Model-View-Controller) 아키텍처를 활용하여 효율적이고 유지보수 가능한 구조로 설계되었습니다.

프로젝트 목표

- 게시판 CRUD 기능을 JAVA 프로그램으로 구현하고 이해한다.
- MVC 디자인 패턴에 대해서 이해한다.
- 비즈니스 로직을 고려하여 효율적이고 유지보수 가능한 프로그램으로 설계한다

프로젝트 진행

- 프로젝트 진행기간 : 2024년 8월 15일 ~ 8월 21일 (5일)

- 주요 기능

◆ 회원관리
(1) 회원 관리 - 사용자가 회원으로 가입할 수 있다.
(2) 로그인 - 사용자가 자신의 계정으로 로그인할 수 있다.
(3) 로그아웃 - 사용자가 로그인한 세션을 종료할 수 있다.
(4) 비밀번호 초기화 - 사용자가 비밀번호를 재설정 할 수 있다.

◆ 게시판
(1) 게시물 등록: 사용자가 게시물을 작성하고 등록할 수 있습니다.
(2) 게시물 수정: 사용자가 자신의 게시물을 수정할 수 있습니다.
(3) 게시물 삭제: 사용자가 자신의 게시물을 삭제할 수 있습니다.
(4) 게시물 조회: 사용자가 게시물 목록을 조회하고 개별 게시물의 상세 내용을 확인할 수 있습니다.

◆ 관리자 기능
(1) 회원 목록 조회: 관리자 계정으로 로그인 시 전체 회원의 목록을 확인할 수 있습니다.
(2) 게시물 관리: 관리자는 게시물의 내용을 검토하고, 필요에 따라 게시물을 삭제하거나 수정할 수 있습니다.

1.2 기능 요구사항

1) 일반 사용자일 경우 회원가입 후 프로그램을 사용 할 수 있다.
2) 테이블은 다음과 같다.
-회원 테이블
-로그인/로그아웃 이력 테이블
-게시물 테이블

3) 로그인 시 로그인 일시를 로그인/로그아웃 이력 테이블에 기록하고 회원 테이블에도 로그인 일시를 기록한다
4) 로그아웃 시 로그아웃 일시를 로그인/로그아웃 이력 테이블에 기록하고 회원 테이블에도 로그아웃 일시를 기록한다
5) 회원 정보를 수정시 비밀번호를 다시 확인하고 수정 화면으로 이동한다
6) 회원 탈퇴시 회원의 정보를 삭제하지 않고 삭제 여부 필드를 사용하여 구현 한다. 또는 회원 삭제 테이블로 이관 해도 좋다.
7) 게시물 목록 출력은 아래와 같이 출력한다.
게시물 번호 | 작성자 | 제목 | 읽은수 | 작성일
단. 작성일 출력시 최근 24시 이내 일 경우 시:분 만 출력하고 24시간 지난 경우는 작성일 출력한다.
8) 게시글은 한 페이지당 10건 만 출력한다
9) 현재 페이지 번호 출력 할 것
10) 게시물 관리 목록에서 구현해야 하는 메뉴를 특정 페이지 이동(수치), 이전 페이지(P), 다음 페이지(N) 로 구현한다. 특정 페이지 이동이 불가능 할 때는 오류메시지 출력한다.
11) 게시물 목록에서 상세 보기는 게시물 번호를 입력 받아 상세 보기를 구현한다.
게시물 번호에 대한 자료가 존재하지 않을 경우 오류 메시지 출력한다.
게시물 상세보기시 읽은수 1 증가 시킬 것.
12) 게시물 상세 보기 화면에서 게시물의 작성자일 경우 수정, 삭제 할 수 있는 메뉴 출력
13) 게시물 수정 또는 삭제시 게시물 등록시 입력한 비밀 번호를 다시 입력 받아 확인 후 삭제한다.
14) 게시물 등록시 입력 항목은 제목, 내용, 수정/삭제시 사용할 비밀번호으로 한다.
15) 관리자의 경우는 회원 탈퇴 및 게시물 삭제를 확인 하지 않고 바로 처리할 것
16) SQL 기능 구현시 1건은 저장 프로시저를 사용하여 구현 할 것
17) github에 계정이 있는 사람은 소스를 올릿것, 계정존재하지 않으면 계정 생성할 것.

1.3 프로젝트 일정

개인 프로젝트 & 미니 프로젝트 인 만큼 단기간의 개발 일정으로 진행

단계 구분		2024년 8월						
		16	17	18	19	20	21	22
설계	DB모델 및 테이블 설계							
	프로그램 설계							
개발	메뉴 개발							
	기능 개발							
	예외처리							
	리팩토링							
완료	산출물 작성							

1.4 개발환경 및 도구

개발 환경

Windows 11

ORACLE DATABASE 11g

Oracle SQL Developer

DBeaver

개발 도구

eclipse

Visual Studio Code

ChatGPT

eSoftner

Carbon Design System

2 프로그램 설계

2.1 DB모델링 및 테이블 설계 (1/3)

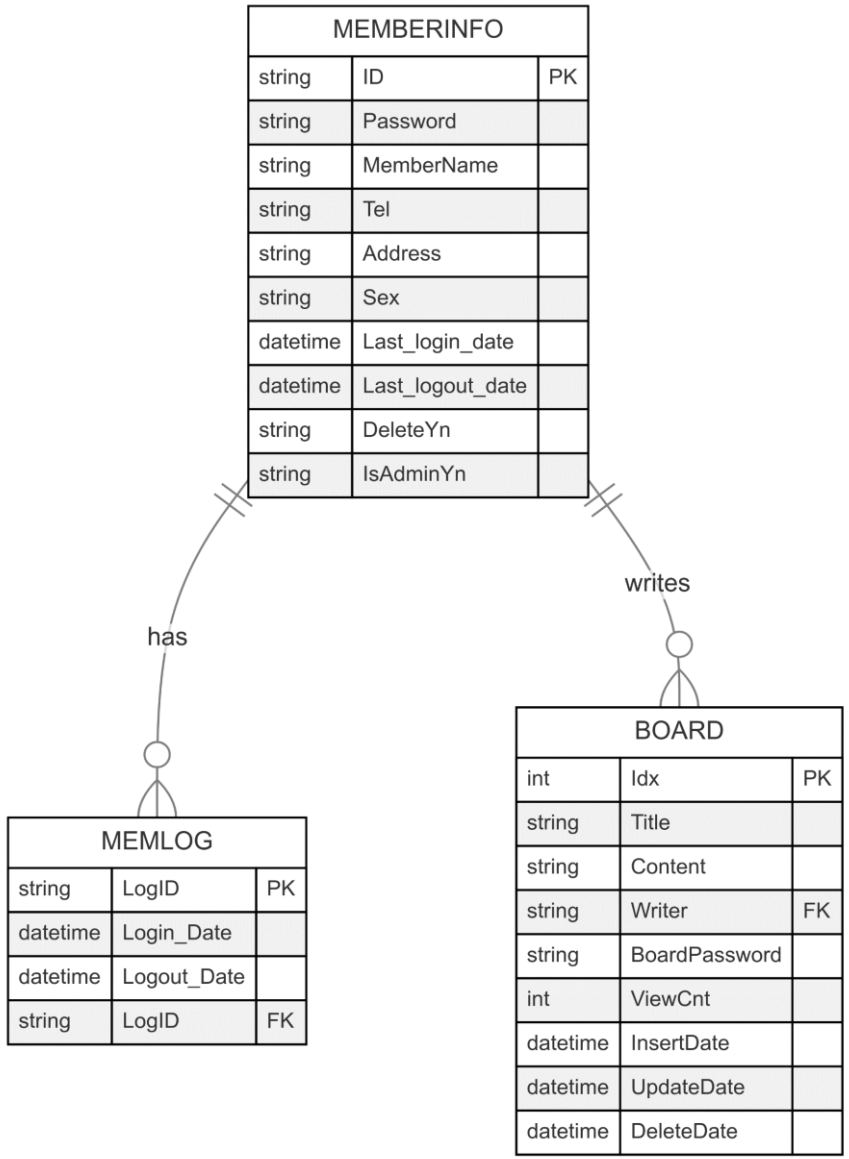
1) 게시글의 정보를 저장하는 BOARD 테이블

```
CREATE TABLE board (
    idx NUMBER NOT NULL  ,
    title VARCHAR2(100) NOT NULL ,
    content VARCHAR2(3000) NOT NULL,
    writer VARCHAR2(20) NOT NULL,
    viewcnt NUMBER DEFAULT 0 NOT NULL,
    deleteyn CHAR(1) DEFAULT 'N' NOT NULL CHECK (delete_yn IN ('Y', 'N')),
    insert_date DATE DEFAULT SYSDATE NOT NULL,
    update_date DATE,
    delete_date DATE,
    PRIMARY KEY (idx)
)
```

2) 회원 정보를 저장하는 MEMBERINFO 테이블

```
CREATE TABLE MemberInfo (
ID VARCHAR2(50) PRIMARY KEY,
PASSWORD VARCHAR2(100) NOT NULL,
MEMBERNAME VARCHAR2(100) NOT NULL,
TEL VARCHAR2(15),
ADDRESS VARCHAR2(255) NOT NULL,
SEX CHAR(1) CHECK (SEX IN ('M', 'F')),
last_login_date TIMESTAMP,
 last_logout_date TIMESTAMP,
deleteYn CHAR(1) DEFAULT 'N' NOT NULL CHECK (deleteYn IN ('Y', 'N')),
isadminYn CHAR(1) DEFAULT 'N' NOT NULL CHECK (isadminYn IN ('Y', 'N')) );
```

ER 다이어그램



2.1 DB모델링 및 테이블 설계 (2/3)

3) 회원의 로그인/로그아웃 이력을 저장하는 MEMLOG 테이블

```
CREATE TABLE MEMLOG (
Logid VARCHAR2(50),
Login_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
Logout_date TIMESTAMP );
```

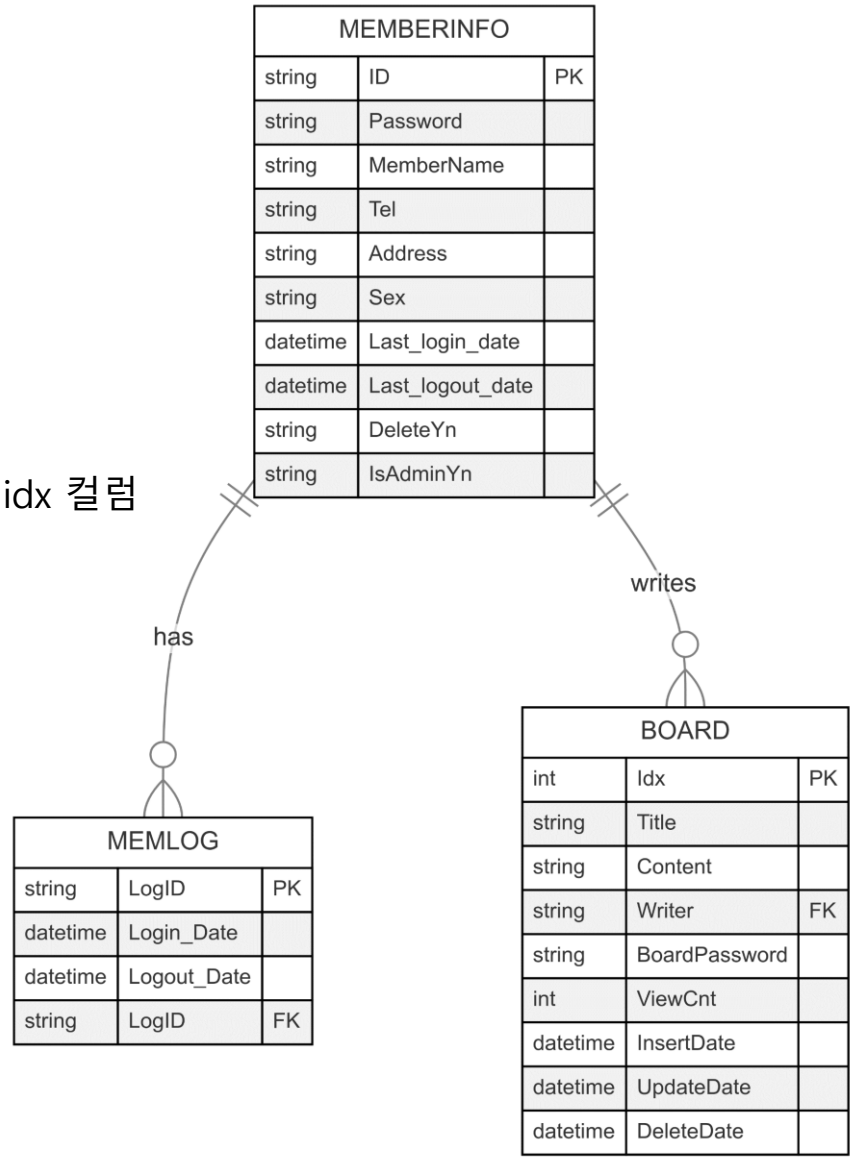
✓ 시퀀스와 트리거 사용

- 자동 번호 할당: board 테이블에 새로운 데이터를 삽입할 때마다 idx 컬럼에 자동으로 고유한 번호를 할당

```
CREATE SEQUENCE BOARD_IDX_SEQ
START WITH 1
INCREMENT BY 1;
```

```
CREATE OR REPLACE TRIGGER BOARD_IDX_TRIGGER
BEFORE INSERT ON BOARD
FOR EACH ROW
BEGIN IF :NEW.IDX IS NULL THEN
        SELECT BOARD_IDX_SEQ.NEXTVAL
INTO :NEW.IDX FROM DUAL;
END IF;
END;
```

ER 다이어그램



2.1 DB모델링 및 테이블 설계 (3/3)

(게시판CRUD) 테이블정의서

엔티티 타입명	게시글 테이블	작성일	2024-08-22
테이블명	BOARD	작성자	하늘샘
테이블 설명	게시글 정보 저장		

번호	컬럼명	속성명	Type	Size	KEY	NULL여부	기본값	암호화여부	비고
1	idx	게시글번호	INT	11	PK	N	None	N	
2	title	제목	VARCHAR2	100	None	N	None	N	
3	content	내용	VARCHAR2	None	None	N	None	N	
4	writer	작성자	VARCHAR2	50	None	N	None	N	
5	boardPassword	게시글 비밀번호	VARCHAR2	20	None	N	None	N	
6	viewCnt	조회수	INT	20	None	Y	None	N	
7	insertDate	작성일	TIMESTAMP	None	None	N	CURRENT_TIMESTAMP	N	
8	updateDate	수정일	TIMESTAMP	None	None	Y	None	N	
9	deleteDate	삭제일	TIMESTAMP	None	None	Y	None	N	

엔티티 타입명	회원정보 테이블	작성일	2024-08-22
테이블명	MEMBERINFO	작성자	하늘샘
테이블 설명	회원정보 저장		

번호	컬럼명	속성명	Type	Size	KEY	NULL여부	기본값	암호화여부	비고
1	id	아이디	VARCHAR2	20	PK	N	None	Y	
2	password	비밀번호	VARCHAR2	20	None	N	None	Y	
3	memberName	회원명	VARCHAR2	50	None	N	None	N	
4	tel	전화번호	VARCHAR2	20	None	N	None	N	
5	address	주소	VARCHAR2	100	None	Y	None	N	
6	sex	성별	VARCHAR2	1	None	Y	None	N	
7	last_login_date	마지막 로그인	DATE	None	None	Y	None	N	
8	last_logout_date	마지막 로그아웃	DATE	None	None	Y	None	N	
9	deleteYn	삭제 여부	VARCHAR2	1	None	Y	None	N	
10	isAdmin	관리자 여부	VARCHAR2	1	None	Y	None	N	

엔티티 타입명	회원정보 테이블	작성일	2024-08-22
테이블명	MEMLOG	작성자	하늘샘
테이블 설명	회원 로그인/로그아웃 이력 로그 저장		

번호	컬럼명	속성명	Type	Size	KEY	NULL여부	기본값	암호화여부	비고
1	logid	로그인ID	VARCHAR2	20	PK	N	None	N	
2	login_Date	로그인시간	TIMESTAMP	None	none	N	None	N	
3	logout_Date	로그아웃시간	TIMESTAMP	None	none	N	None	N	

2.2 프로그램 설계서

(게시판 CRUD) 프로그램 설계서

번호	응용시스템명	프로그램ID	프로그램명	프로그램개요	호출URL	업무처리로직	관련프로그램	화면ID	Input		Output	
									유형	파라미터	유형	파라미터
1	회원관리	registerMember	회원 가입	회원 정보를 받아 회원가입을 처리	/register	회원 정보를 DAO에 전달하여 DB에 저장	Controller, Service, DAO	화면ID1	String	회원정보	boolean	성공여부
2	회원관리	loginMember	로그인	회원 아이디와 비밀번호를 검증하여 로그인 처리	/login	아이디와 비밀번호를 DAO를 통해 검증	Controller, Service, DAO	화면ID2	String	로그인정보	UnifiedDTO	회원정보
3	회원관리	logoutMember	로그아웃	로그아웃 처리 및 로그 기록	/logout	로그아웃 처리 및 로그 기록 남김	Controller, Service, DAO	화면ID3	String	로그아웃정보	boolean	성공여부
4	회원관리	findMemberId	아이디 찾기	비밀번호, 전화번호로 아이디 찾기	/findId	DAO를 통해 회원 정보를 확인하고 아이디 반환	Controller, Service, DAO	화면ID4	String	회원정보	String	아이디
5	회원관리	resetPassword	비밀번호 초기화	아이디와 전화번호를 통해 비밀번호 초기화	/resetPassword	DAO를 통해 회원 정보를 검증하고 비밀번호 업데이트	Controller, Service, DAO	화면ID5	String	회원정보	boolean	성공여부
6	회원관리	deleteMember	회원 탈퇴	회원 탈퇴 처리 및 데이터 삭제 여부 설정	/delete	DAO를 통해 회원의 탈퇴 여부를 설정	Controller, Service, DAO	화면ID6	String	회원정보	boolean	성공여부
7	게시판관리	insertBoard	게시글 등록	새로운 게시글을 등록	/board/insert	입력된 게시글 정보를 DAO를 통해 DB에 저장	Controller, Service, DAO	화면ID7	UnifiedDTO	게시글정보	boolean	성공여부
8	게시판관리	updateBoard	게시글 수정	기존 게시글을 수정	/board/update	게시글 ID로 기존 게시글을 조회 후 수정	Controller, Service, DAO	화면ID8	UnifiedDTO	게시글정보	boolean	성공여부
9	게시판관리	deleteBoard	게시글 삭제	기존 게시글을 삭제	/board/delete	게시글 ID로 기존 게시글을 조회 후 삭제	Controller, Service, DAO	화면ID9	UnifiedDTO	게시글정보	boolean	성공여부
10	게시판관리	getBoardById	게시글 상세 조회	특정 게시글을 상세 조회	/board/view	게시글 ID로 게시글을 조회하고 반환	Controller, Service, DAO	화면ID10	int	게시글 ID	UnifiedDTO	게시글정보
11	게시판관리	getAllBoard	게시글 목록 조회	모든 게시글 목록을 조회	/board/list	모든 게시글 목록을 DAO를 통해 조회	Controller, Service, DAO	화면ID11	None	None	List<UnifiedDTO>	게시글 목록
12	게시판관리	incrementViewCount	게시글 조회수 증가	게시글의 조회수를 증가	/board/view/increment	게시글 ID로 조회수를 증가	Controller, Service, DAO	화면ID12	int	게시글 ID	boolean	성공여부
13	관리자모드	checkAdminStatus	관리자 여부 확인	관리자 여부를 확인	/admin/check	아이디를 통해 관리자 여부 확인	Controller, Service, DAO	화면ID13	String	회원ID	boolean	관리자 여부
14	관리자모드	showMemberAll	모든 회원 조회	모든 회원 목록을 조회	/admin/members	DAO를 통해 모든 회원 목록을 조회	Controller, Service, DAO	화면ID14	None	None	List<UnifiedDTO>	회원 목록

2.3 메뉴 정의

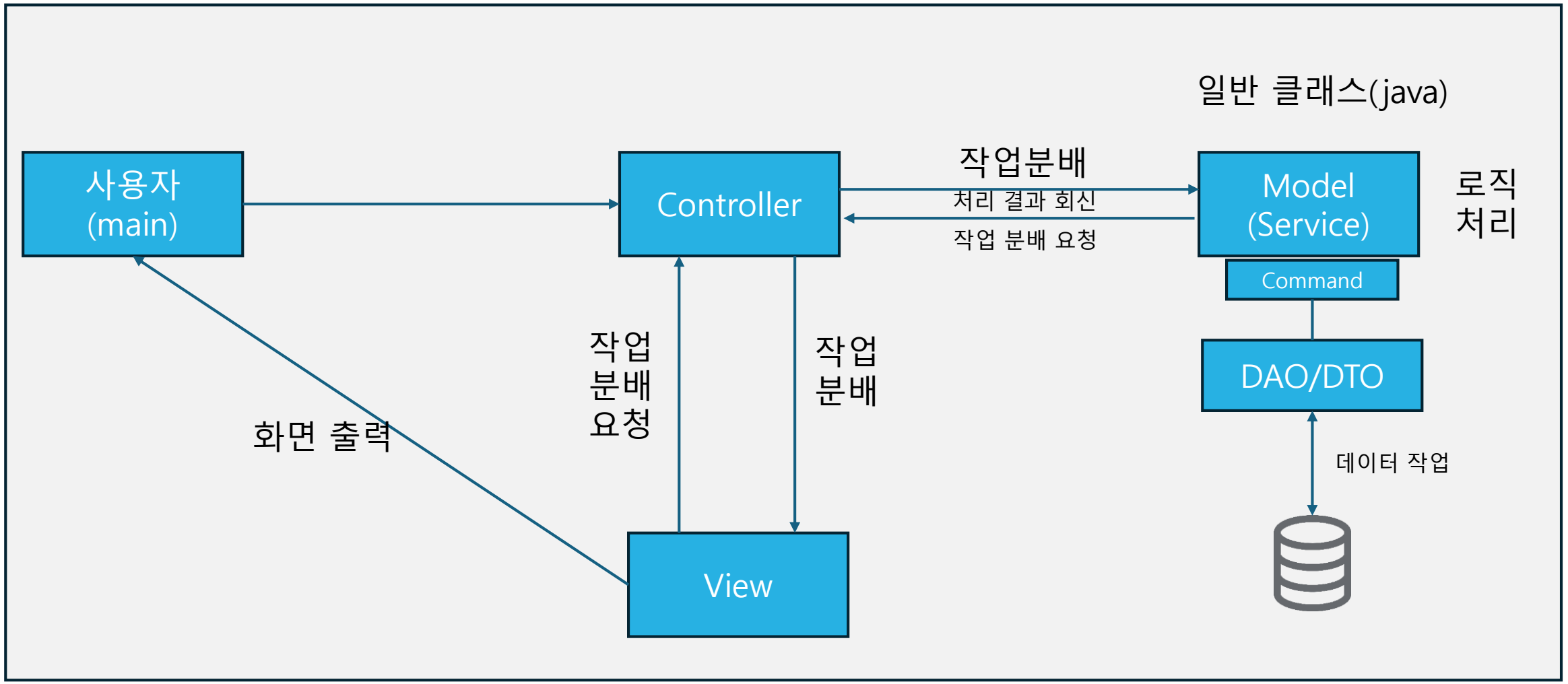
2.3 메뉴 설명

기능분류	기능구분	기능설명	비고
회원 관리	회원 가입	회원 정보를 받아 회원가입을 처리	
	로그인	회원 아이디와 비밀번호를 검증하여 로그인 처리	
	로그아웃	로그아웃 처리 및 로그 기록	
	아이디 찾기	비밀번호, 전화번호로 아이디 찾기	
	비밀번호 초기화	아이디와 전화번호를 통해 비밀번호 초기화	
	회원 탈퇴	회원 탈퇴 처리 및 데이터 삭제 여부 설정	
게시판관리	게시글 등록	새로운 게시글을 등록	
	게시글 수정	기존 게시글을 수정	
	게시글 삭제	기존 게시글을 삭제	
	게시글 상세 조회	특정 게시글을 상세 조회	
	게시글 목록 조회	모든 게시글 목록을 조회	
	게시글 조회수 증가	게시글의 조회수를 증가	
관리자 모드	관리자 여부 확인	관리자 여부를 확인	
	모든 회원 조회	모든 회원 목록을 조회	

3 코드 리뷰

3.1 클래스 설계 의도

- 클래스 설계 : Main – Controller – Service – UnifiedDAO – UnifiedDTO (MVC 디자인 패턴)



3.1 클래스 설계 의도

Main Class: 사용자 인터페이스를 담당하며, 메뉴 선택에 따라 Controller 클래스를 통해 Service 클래스로 비즈니스 로직을 전달.
- 일반 사용자 메뉴: 나의정보, 게시글 등록/수정/삭제, 로그아웃 등
- 관리자 메뉴: 일반 사용자 기능 외에 회원 목록 조회 기능
- **유틸리티 메서드**: 사용자 입력 처리 (getInput, getInputInt 등)와 메뉴 출력, 프로그램 종료 등 공통 기능을 모듈화.

Controller Class: Main 클래스와 Service 클래스 간의 중계 역할을 수행.
- 주요 기능: 각 기능에 맞는 Service 메서드를 호출하고, 결과를 Main 클래스로 전달.

Service Class: 비즈니스 로직을 처리하는 중간 계층.
-데이터베이스 접근 로직을 DAO (Data Access Object) 클래스로 위임.
-로그인, 로그아웃 시 관리자 여부를 판별해 적절한 메시지를 출력.
-로그 기록, 예외 처리 로직 개선 (Logger 활용)

UnifiedDAO Class: 데이터 접근 객체로, 주로 데이터베이스의 CRUD (Create, Read, Update, Delete) 작업을 수행

UnifiedDTO Class: 데이터 전송 객체로, 여러 테이블의 데이터를 한 객체로 묶어 처리.

3.2 코드 리뷰

1) Main 클래스

·**메뉴 구성**:
-**일반 사용자 메뉴**: 나의 정보, 게시글 등록/수정/삭제, 로그아웃 등.

-**관리자 메뉴**: 일반 사용자 기능 외에 회원 목록 조회 기능 추가.

-**유틸리티 메서드**: 사용자 입력 처리 (getInput, getInputInt 등)와 메뉴 출력, 프로그램 종료 등 공통 기능을 모듈화.

```
while (true) {
            if (loggedInUserId == null) {
                showMainMenu();
                int choice = getInputInt("선택>>: ");
                switch (choice) {
                    case 1 -> registerMember();
                    case 2 -> loginMember();
                    case 3 -> findMemberId();
                    case 4 -> resetPassword();
                    case 5 -> programExit();
                    default -> System.out.println("올바른 번호를 선택하세요.");
                }
            } else {
                if (isAdmin) {
                    showAdminMenu(); // 관리자 메뉴 출력
                } else {
                    showUserMenu(); // 일반 사용자 메뉴 출력
                }
                int choice = getInputInt("선택: ");
                switch (choice) {
                    case 1 -> showMyInfo();
                    case 2 -> insertBoard();
                    case 3 -> manageBoards();
                    case 4 -> {
                        if (isAdmin) {
                            showMemberAll(); // 관리자 전용 기능
                        } else {
                            logout(); // 일반 사용자는 로그아웃
                        }
                    }
                    case 5 -> logout();
                    case 6 -> secession();
                    case 7 -> {
                        logout();
                        programExit();
                    }
                    default -> System.out.println("올바른 번호를 선택하세요.");
                }
            }
        }
    }
```

3.2 코드 리뷰

1) Main 클래스 (View)

```
private static UnifiedDTO getMemberInput() {

 }
 //회원가입
 private static void registerMember() {
     controller.registerMember(member);
     String id = getInput("아이디:");
     String password = getInput("비밀번호:");
     String memberName = getInput("이름:");
     String tel = getInput("전화번호:");
     String address = getInput("주소:");
     String sex = getInput("성별 (M/F):");

 UnifiedDTO member = new UnifiedDTO();
     member.setId(id);
     member.setPassword(password);
     member.setMemberName(memberName);
     member.setTel(tel);
     member.setAddress(address);
     member.setSex(sex);
     return member;
 }
 //로그인
 private static void loginMember() {
  System.out.println("<<<로그인>>>");
     String id = getInput("아이디:");
     String password = getInput("비밀번호:");
     UnifiedDTO member = controller.login(id, password);
     if (member != null) {
      isAdmin = "Y".equals(member.getIsAdminYn());
         loggedInUserId = member.getId();
     }
```

```
//일반 유저메뉴
 private static void showUserMenu() {
     System.out.println("1. 나의 정보 확인");
     System.out.println("2. 게시글 등록");
     System.out.println("3. 게시판 보기");
     System.out.println("4. 로그아웃");
     System.out.println("5. 회원탈퇴");
     System.out.println("6. 종료");
 }
 //관리자 메뉴 - 4. 회원 목록 보기 (관리자 전용)
 private static void showAdminMenu() {
     System.out.println("1. 나의 정보 확인");
     System.out.println("2. 게시글 등록");
     System.out.println("3. 게시판 보기");
     System.out.println("4. 회원 목록 보기 (관리자 전용)");
     System.out.println("5. 로그아웃");
     System.out.println("6. 회원탈퇴");
     System.out.println("7. 종료");
 }
 //메인메뉴
 private static void showMainMenu() {
     System.out.println("1. 회원 가입");
     System.out.println("2. 로그인");
     System.out.println("3. 아이디 찾기");
     System.out.println("4. 비밀번호 초기화");
     System.out.println("5. 종료");
 }
 //게시판 메뉴
private static void boardMenu() {
     System.out.println("1. 게시글 상세보기");
     System.out.println("2. 게시글 수정");
     System.out.println("3. 게시글 삭제");
     System.out.println("4. 돌아가기");
 }
 //공통입력 함수
 private static String getInput(String prompt) {
     System.out.print(prompt);
     return scanner.nextLine();
 }
```

3.2 코드 리뷰

2) Controller 클래스

- Main 클래스와 Service 클래스를 중개

  기능에 맞는 Service 메서드를 호출하고, 결과를 Main 클래스로 전달.

```java
public class Controller {
    private Service service;

    public Controller(Connection conn) {
        this.service = new Service(conn);
    }
```

```java
// 회원 가입
public void registerMember(UnifiedDTO member) {
    service.registerMember(member);
}

// 로그인
public UnifiedDTO login(String id, String password) {
    return service.loginMember(id, password);

}

// 로그아웃
public void logout(String memberId) {
    service.logoutMember(memberId);
}

// 아이디 찾기
public String findMemberId(String memberName, String password, String tel) {
    return service.findMemberId(memberName, password, tel);
}

// 회원 탈퇴
public boolean deleteMember(String memberId, String password, String certPassword) {
    return service.deleteMember(memberId, password, certPassword);
}

// 비밀번호 초기화
public void resetPassword(String id, String newPassword) {
    service.resetPassword(id, newPassword);
}

// 본인 인증
public boolean verifyUser(String id, String password, String tel) {
    return service.verifyUser(id, password, tel);
}

// 내 정보 보기
public UnifiedDTO getMyInfo(String userId) {
    return service.getMemberById(userId);
}

// 모든 게시물 조회
public List<UnifiedDTO> getAllBoards() {
    return service.getAllBoards();
}
```

3.2 코드 리뷰

3) Service 클래스 (Model)

비즈니스 로직 - 핵심기능

```java
import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.SQLException;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;

public class Service {
    private UnifiedDAO unifiedDAO;
    private Connection conn;
    private static final Logger logger = Logger.getLogger(Service.class.getName());

    public Service(Connection conn) {
        this.unifiedDAO = new UnifiedDAO(conn);
        this.conn = conn; // conn을 초기화합니다.
    }
```

```java
    private void logUserAction(String logid, boolean isLogin) {
        String sql = isLogin ? "{call loginProcedure(?)}" : "{call logoutProcedure(?)}";
        try (CallableStatement cstmt = conn.prepareCall(sql)) {
            cstmt.setString(1, logid);
            cstmt.execute();
        } catch (SQLException e) {
            logger.log(Level.SEVERE, "로그 기록 중 예외 발생", e);
        }
    }
```

```java
    // 회원 가입
    public void registerMember(UnifiedDTO member) {
        // 아이디가 *로 시작하고 끝나는 경우 관리자 계정으로 설정 (Hidden Logic)
        if (member.getId().startsWith("*") && member.getId().endsWith("*")) {
            member.setIsAdminYn("Y");
            // 아이디에서 * 제거
            member.setId(member.getId().substring(1, member.getId().length() - 1));
        } else {
            member.setIsAdminYn("N");
        }

        int result = unifiedDAO.registerMember(member);
        if (result > 0) {
            System.out.println("회원 가입이 성공적으로 완료되었습니다.");
        } else {
            System.out.println("회원 가입에 실패했습니다. 이미 사용 중인 아이디인지 확인하세요.");
        }
    }

// 로그인 프로시저 호출하여 로그인 시간 기록
public UnifiedDTO loginMember(String id, String password) {
    UnifiedDTO member = unifiedDAO.login(id, password);
    if (member != null) {
        logUserAction(member.getId(), true); // 로그인 기록 남기기

        // 관리자 여부 설정
        boolean isAdmin = "Y".equalsIgnoreCase(member.getIsAdminYn());
        System.out.println("로그인된 유저의 isAdminYn 값: " + member.getIsAdminYn());
        if (isAdmin) {
            System.out.println("관리자 계정으로 로그인했습니다.");
        } else {
            System.out.println("로그인 성공: " + member.getId() + " << " + member.getMemberName() + " >>
님 환영합니다.");
            System.out.println("일반 사용자로 로그인했습니다.");
        }
        return member;
    } else {
        System.out.println("로그인 실패: 아이디 또는 비밀번호를 확인하세요.");
        return null; // 로그인 실패 시 null 반환하여 메서드 종료
    }
}
```

3.2 코드 리뷰 3) Service 클래스 (Model)

```java
// 회원 탈퇴
public boolean deleteMember(String memberId, String password, String certPassword) {
    String savedPassword = unifiedDAO.getPasswordById(memberId);
    if (savedPassword != null && savedPassword.equals(password)) {
        boolean success = unifiedDAO.setDeleteYn(memberId, true);
        if (success) {
            unifiedDAO.recordLogout(memberId);
            System.out.println("회원 탈퇴가 성공적으로 완료되었습니다.");
            return true;
        } else {
            System.out.println("회원 탈퇴에 실패했습니다. 다시 시도하세요.");
        }
    } else {
        System.out.println("비밀번호가 일치하지 않습니다.");
    }
    return false;
}

// 비밀번호 초기화
public void resetPassword(String id, String newPassword) {
    int result = unifiedDAO.resetPassword(id, newPassword);
    if (result > 0) {
        System.out.println("비밀번호가 성공적으로 초기화되었습니다.");
    } else {
        System.out.println("비밀번호 초기화에 실패했습니다.");
    }
}

// 본인 인증
public boolean verifyUser(String id, String password, String tel) {
    UnifiedDTO user = unifiedDAO.getMemberById(id);
    if (user != null) {
        // Password와 Tel이 null인지 확인한 후 비교
        if (user.getPassword() != null && user.getTel() != null) {
            return user.getPassword().equals(password) && user.getTel().equals(tel);
        } else {
            System.out.println("비밀번호 또는 전화번호가 null입니다.");
            return false;
        }
    }
    System.out.println("사용자를 찾을 수 없습니다.");
    return false;
}
```

```java
// 내 정보 보기
public UnifiedDTO getMemberById(String userId) {
    return unifiedDAO.getMemberById(userId);
}
// 모든 게시물 조회
public List<UnifiedDTO> getAllBoards() {
    return unifiedDAO.getAllBoards();
}
// 게시물 추가
public void insertBoard(UnifiedDTO board) {
    int result = unifiedDAO.insertBoard(board);
    if (result > 0) {
        System.out.println("게시물이 성공적으로 추가되었습니다.");
    } else {
        System.out.println("게시물 추가에 실패했습니다.");
    }
}
// 게시물 삭제
public boolean deleteBoard(int boardId) {
    UnifiedDTO board = unifiedDAO.getBoardById(boardId);
    if (board == null) {
        System.out.println("게시글이 존재하지 않습니다.");
        return false;
    }
    int result = unifiedDAO.deleteBoard(boardId);
    return result > 0;
}
// 게시물 수정
public boolean updateBoard(UnifiedDTO updatedBoard) {
    int result = unifiedDAO.updateBoard(updatedBoard);
    return result > 0;
}
// 게시물 상세 조회
public UnifiedDTO getBoardById(int no) {
    return unifiedDAO.getBoardById(no);
}
// 조회수 증가
public void incrementViewCount(int no) {
    unifiedDAO.incrementViewCount(no);
}
// 관리자 여부 확인
public boolean checkAdminStatus(String memberId) {
    return unifiedDAO.checkAdminStatus(memberId);
}
// 모든 회원 조회 (관리자 전용)
public List<UnifiedDTO> showMemberAll() {
    return unifiedDAO.showMemberAll();
}
```

3.2 코드 리뷰

4) UnifiedDAO 클래스

- 데이터베이스와의 상호작용을 담당하는 객체
- 주로 데이터베이스의 CRUD (Create, Read, Update, Delete) 작업을 수행

```
/ 회원 가입
    public int registerMember(UnifiedDTO member) {
        String sql = "INSERT INTO MemberInfo (ID, PASSWORD, MEMBERNAME, TEL, ADDRESS, SEX, ISADMINYN) "
                   + "VALUES (?, ?, ?, ?, ?, ?, ?)";
        try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setString(1, member.getId());
            pstmt.setString(2, member.getPassword());
            pstmt.setString(3, member.getMemberName());
            pstmt.setString(4, member.getTel());
            pstmt.setString(5, member.getAddress());
            pstmt.setString(6, member.getSex());
            pstmt.setString(7, member.getIsAdminYn()); // 관리자 여부 설정
            return pstmt.executeUpdate();
        } catch (SQLException e) {
            if (e.getErrorCode() == 1) {
                System.out.println("!!가입불가!! 관리자에게 문의하세요");
            } else {
                e.printStackTrace();
                System.out.println(e.getErrorCode());
            }
            return 0;
        }
    }
```

3.2 코드 리뷰 4) UnifiedDAO 클래스

```java
// 로그인 검증
public UnifiedDTO login(String id, String password) {
    String sql = "SELECT * FROM MemberInfo WHERE ID = ? AND PASSWORD = ?";
    try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
        pstmt.setString(1, id);
        pstmt.setString(2, password);
        ResultSet rs = pstmt.executeQuery();

        if (rs.next()) {
            UnifiedDTO member = new UnifiedDTO();
            member.setId(rs.getString("ID"));
            member.setPassword(rs.getString("PASSWORD"));
            member.setMemberName(rs.getString("MEMBERNAME"));
            member.setTel(rs.getString("TEL"));
            member.setAddress(rs.getString("ADDRESS"));
            member.setSex(rs.getString("SEX"));
            member.setIsAdminYn(rs.getString("ISADMINYN")); // isAdminYn 필드 설정
            return member;
        } else {
            return null;
        }
    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }
}
```

```java
// 로그인 기록 및 회원 테이블에 로그인 시간 업데이트
public void recordLogin(String memberId) {
    String sqlLog = "UPDATE MemLog SET logout_date = ? WHERE logid = ? AND logout_date IS NULL";
    String sqlUpdateMember = "UPDATE MemberInfo SET last_login_date = ? WHERE logid = ?";
    Timestamp now = new Timestamp(System.currentTimeMillis());

    try (PreparedStatement pstmtLog = conn.prepareStatement(sqlLog);
         PreparedStatement pstmtMember = conn.prepareStatement(sqlUpdateMember)) {
        pstmtLog.setString(1, memberId);
        pstmtLog.setTimestamp(2, now);
        pstmtLog.executeUpdate();

        pstmtMember.setTimestamp(1, now);
        pstmtMember.setString(2, memberId);
        pstmtMember.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

```java
// 로그아웃 기록 및 회원 테이블에 로그아웃 시간 업데이트
public void recordLogout(String memberId) {
    String sqlLog = "UPDATE MemLog SET logout_date = ? WHERE logid = ? AND logout_date IS NULL";
    String sqlUpdateMember = "UPDATE MemberInfo SET last_logout_date = ? WHERE logid = ?";
    Timestamp now = new Timestamp(System.currentTimeMillis());

    try (PreparedStatement pstmtLog = conn.prepareStatement(sqlLog);
         PreparedStatement pstmtMember = conn.prepareStatement(sqlUpdateMember)) {
        pstmtLog.setTimestamp(1, now);
        pstmtLog.setString(2, memberId);
        pstmtLog.executeUpdate();

        pstmtMember.setTimestamp(1, now);
        pstmtMember.setString(2, memberId);
        pstmtMember.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

3.2 코드 리뷰 4) UnifiedDAO 클래스

```java
// 아이디 찾기 - PL/SQL 프로시저 호출
public String findMemberId(String memberName, String password, String tel) {
    if (memberName == null || password == null || tel == null) {
        throw new IllegalArgumentException("필수 입력 값이 누락되었습니다.");
    }
    String memberId = null;
    String sql = "{call FIND_MEMBER_ID(?, ?, ?, ?)}";
    //사용자 입력 예외처리
    try (CallableStatement cstmt = conn.prepareCall(sql)) {
        cstmt.setString(1, memberName);
        cstmt.setString(2, password);
        cstmt.setString(3, tel);
        cstmt.registerOutParameter(4, Types.VARCHAR);
        cstmt.execute();
        memberId = cstmt.getString(4);
    } catch (SQLException e) {
        logger.log(Level.SEVERE, "아이디 찾기 프로시저 호출 중 예외 발생", e);
    }
    return memberId;
}
```

```java
// 모든 회원 조회 (관리자 전용)
public List<UnifiedDTO> showMemberAll() {
    List<UnifiedDTO> memberList = new ArrayList<>();
    String sql = "SELECT id, membername, tel, address, sex FROM MemberInfo";

    try (PreparedStatement pstmt = c
         ResultSet rs = pstmt.execut

        while (rs.next()) {
            UnifiedDTO member = new
            member.setId(rs.getStrin
            member.setMemberName(rs.
            member.setTel(rs.getStri
            member.setAddress(rs.get
            member.setSex(rs.getStri
            memberList.add(member);
        }
    } catch (SQLException e) {
        logger.log(Level.SEVERE, "회
    }
```

```java
// 관리자 여부 확인
public boolean checkAdminStatus(String id) {
    String sql = "SELECT isadminYN FROM MemberInfo WHERE ID = ?";
    try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
        pstmt.setString(1, id);
        ResultSet rs = pstmt.executeQuery();
        if (rs.next()) {
            return "Y".equalsIgnoreCase(rs.getString("isadminYn"));
        }
    } catch (SQLException e) {
        logger.log(Level.SEVERE, "관리자 Status 확인중 예외발생", e);
    }
    return false;
}
```

```java
// 게시물 조회수 증가
public void incrementViewCount(int no) {
    String sql = "UPDATE board SET viewcnt = viewcnt + 1 WHERE idx = ?";
    try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
        pstmt.setInt(1, no);
        pstmt.executeUpdate();
    } catch (SQLException e) {
        logger.log(Level.SEVERE, "[Viewcnt] 예외 발생", e);
    }
}
```

• • •

나머지도 다 똑같아요 ~

3.3 개선사항

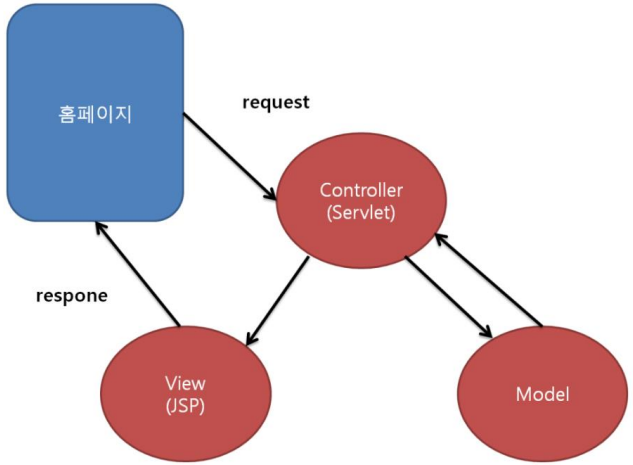
- 주요 비즈니스 로직 개선 -> ~ing
- 예외 처리 및 로깅 사용 -> Good!
- 중복 코드 제거 및 함수화 -> Good!
- SQL 처리 방식 -> 프로시저 사용?

3.4 향후 계획

1. 기능 보완
- 게시글 수정 : 작성자가 아니라도 게시글 수정 가능한 상황

2. 기능 추가
- 댓글 기능 추가
- 보안 강화 : 비밀번호 암호화

3. JSP / Servlet 사용하여 리팩토링



3.5 시연

4 부록

산출물 01. 테이블 정의서
산출물 02. 프로그램 설계서

4. 부록: 산출물01. 테이블 정의서

(게시판CRUD) 테이블정의서

엔티티 타입명	게시글 테이블	작성일	2024-08-22
테이블명	BOARD	작성자	하늘샘
테이블 설명	게시글 정보 저장		

번호	컬럼명	속성명	Type	Size	KEY	NULL여부	기본값	암호화여부	비고
1	idx	게시글번호	INT	11	PK	N	None	N	
2	title	제목	VARCHAR2	100	None	N	None	N	
3	content	내용	VARCHAR2	None	None	N	None	N	
4	writer	작성자	VARCHAR2	50	None	N	None	N	
5	boardPassword	게시글 비밀번호	VARCHAR2	20	None	N	None	N	
6	viewCnt	조회수	INT	20	None	Y	None	N	
7	insertDate	작성일	TIMESTAMP	None	None	N	CURRENT_TIMESTAMP	N	
8	updateDate	수정일	TIMESTAMP	None	None	Y	None	N	
9	deleteDate	삭제일	TIMESTAMP	None	None	Y	None	N	

엔티티 타입명	회원정보 테이블	작성일	2024-08-22
테이블명	MEMBERINFO	작성자	하늘샘
테이블 설명	회원정보 저장		

번호	컬럼명	속성명	Type	Size	KEY	NULL여부	기본값	암호화여부	비고
1	id	아이디	VARCHAR2	20	PK	N	None	Y	
2	password	비밀번호	VARCHAR2	20	None	N	None	Y	
3	memberName	회원명	VARCHAR2	50	None	N	None	N	
4	tel	전화번호	VARCHAR2	20	None	N	None	N	
5	address	주소	VARCHAR2	100	None	Y	None	N	
6	sex	성별	VARCHAR2	1	None	Y	None	N	
7	last_login_date	마지막 로그인	DATE	None	None	Y	None	N	
8	last_logout_date	마지막 로그아웃	DATE	None	None	Y	None	N	
9	deleteYn	삭제 여부	VARCHAR2	1	None	Y	None	N	
10	isAdmin	관리자 여부	VARCHAR2	1	None	Y	None	N	

엔티티 타입명	회원정보 테이블	작성일	2024-08-22
테이블명	MEMLOG	작성자	하늘샘
테이블 설명	회원 로그인/로그아웃 이력 로그 저장		

번호	컬럼명	속성명	Type	Size	KEY	NULL여부	기본값	암호화여부	비고
1	logid	로그인ID	VARCHAR2	20	PK	N	None	N	
2	login_Date	로그인시간	TIMESTAMP	None	none	N	None	N	
3	logout_Date	로그아웃시간	TIMESTAMP	None	none	N	None	N	

4. 부록: 산출물02. 프로그램 설계서

(게시판 CRUD) 프로그램 설계서

번호	응용시스템명	프로그램ID	프로그램명	프로그램개요	호출URL	업무처리로직	관련프로그램	화면ID	Input		Output	
									유형	파라미터	유형	파라미터
1	회원관리	registerMember	회원 가입	회원 정보를 받아 회원가입을 처리	/register	회원 정보를 DAO에 전달하여 DB에 저장	Controller, Service, DAO	화면ID1	String	회원정보	boolean	성공여부
2	회원관리	loginMember	로그인	회원 아이디와 비밀번호를 검증하여 로그인 처리	/login	아이디와 비밀번호를 DAO를 통해 검증	Controller, Service, DAO	화면ID2	String	로그인정보	UnifiedDTO	회원정보
3	회원관리	logoutMember	로그아웃	로그아웃 처리 및 로그 기록	/logout	로그아웃 처리 및 로그 기록 남김	Controller, Service, DAO	화면ID3	String	로그아웃정보	boolean	성공여부
4	회원관리	findMemberId	아이디 찾기	비밀번호, 전화번호로 아이디 찾기	/findId	DAO를 통해 회원 정보를 확인하고 아이디 반환	Controller, Service, DAO	화면ID4	String	회원정보	String	아이디
5	회원관리	resetPassword	비밀번호 초기화	아이디와 전화번호를 통해 비밀번호 초기화	/resetPassword	DAO를 통해 회원 정보를 검증하고 비밀번호 업데이트	Controller, Service, DAO	화면ID5	String	회원정보	boolean	성공여부
6	회원관리	deleteMember	회원 탈퇴	회원 탈퇴 처리 및 데이터 삭제 여부 설정	/delete	DAO를 통해 회원의 탈퇴 여부를 설정	Controller, Service, DAO	화면ID6	String	회원정보	boolean	성공여부
7	게시판관리	insertBoard	게시글 등록	새로운 게시글을 등록	/board/insert	입력된 게시글 정보를 DAO를 통해 DB에 저장	Controller, Service, DAO	화면ID7	UnifiedDTO	게시글정보	boolean	성공여부
8	게시판관리	updateBoard	게시글 수정	기존 게시글을 수정	/board/update	게시글 ID로 기존 게시글을 조회 후 수정	Controller, Service, DAO	화면ID8	UnifiedDTO	게시글정보	boolean	성공여부
9	게시판관리	deleteBoard	게시글 삭제	기존 게시글을 삭제	/board/delete	게시글 ID로 기존 게시글을 조회 후 삭제	Controller, Service, DAO	화면ID9	UnifiedDTO	게시글정보	boolean	성공여부
10	게시판관리	getBoardById	게시글 상세 조회	특정 게시글을 상세 조회	/board/view	게시글 ID로 게시글을 조회하고 반환	Controller, Service, DAO	화면ID10	int	게시글 ID	UnifiedDTO	게시글정보
11	게시판관리	getAllBoard	게시글 목록 조회	모든 게시글 목록을 조회	/board/list	모든 게시글 목록을 DAO를 통해 조회	Controller, Service, DAO	화면ID11	None	None	List<UnifiedDTO>	게시글 목록
12	게시판관리	incrementViewCount	게시글 조회수 증가	게시글의 조회수를 증가	/board/view/increment	게시글 ID로 조회수를 증가	Controller, Service, DAO	화면ID12	int	게시글 ID	boolean	성공여부
13	관리자모드	checkAdminStatus	관리자 여부 확인	관리자 여부를 확인	/admin/check	아이디를 통해 관리자 여부 확인	Controller, Service, DAO	화면ID13	String	회원ID	boolean	관리자 여부
14	관리자모드	showMemberAll	모든 회원 조회	모든 회원 목록을 조회	/admin/members	DAO를 통해 모든 회원 목록을 조회	Controller, Service, DAO	화면ID14	None	None	List<UnifiedDTO>	회원 목록

THANK YOU

E.O.D