InstaBook

1차 프로젝트

SNS

안은종

목차

1) 개발 과정 보유 기술

Java SE 8u261

Product Version: 5.5.62

Community Edition 7.1.5

mysql-connector-java-5.1.46

1) 개발 과정 1-3) DB테이블

Database : instabook

필요한 테이블 수: 2개

회원 테이블: user_insta

칼럼명	컬럼 타입	컬럼 조건	기타	타입명
id_user	Varchar(50)	PK	아이디	String
pw_user	Varchar(30)	NN	패스워드	String
nickname_user	Varchar(30)	NN	별칭	String
img_user	Varchar(100)	Null	프로필사진	String
gender_user	char(1)	Null	성별	char
birth_user	Varchar(10)	Null	출생일	String

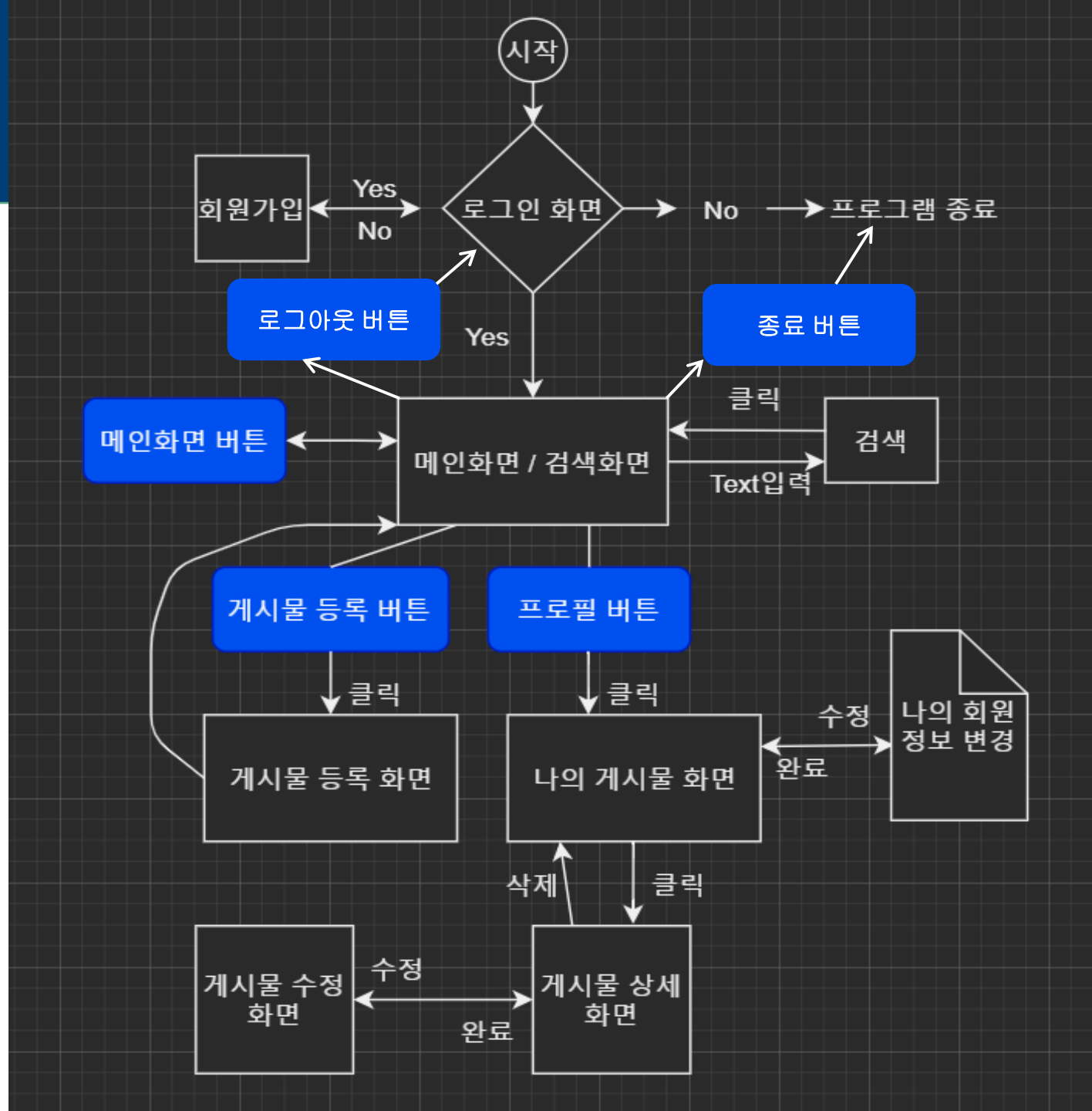
게시물 테이블 : post_insta

칼럼명	컬럼 타입	컬럼 조건	기타	타입명
id_post	Int(11)	PK	아이디	int
id_user	Varchar(50)	FK	게시자	String
date_post	Varchar(8)	NN	게시일	String
hash_post	Varchar(30)	NN	해쉬태그	String
img_post	Varchar(50)	NN	이미지	String
like_post	Int(5)	NN	좋아요	int

1) 개발 과정

1-3) 디렉토리 과정

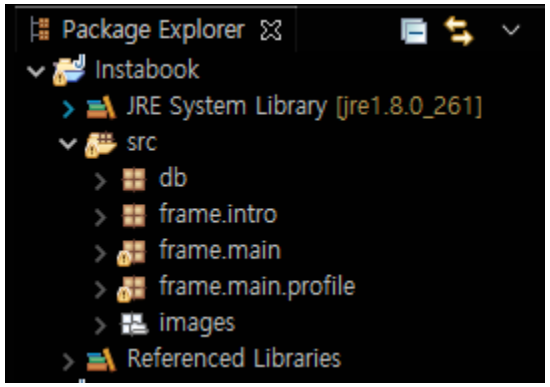
메인 화면 순서도



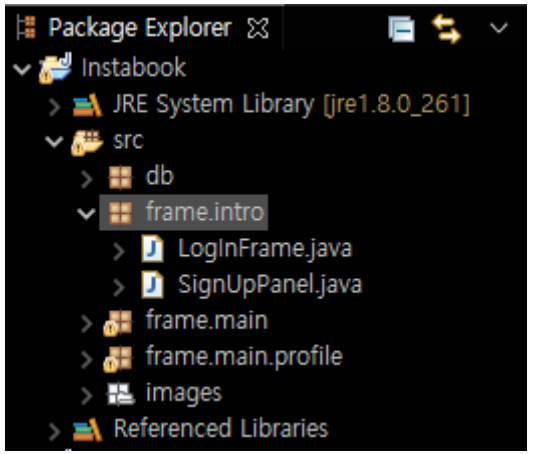
1) 개발 과정

1-4) 프로젝트 트리

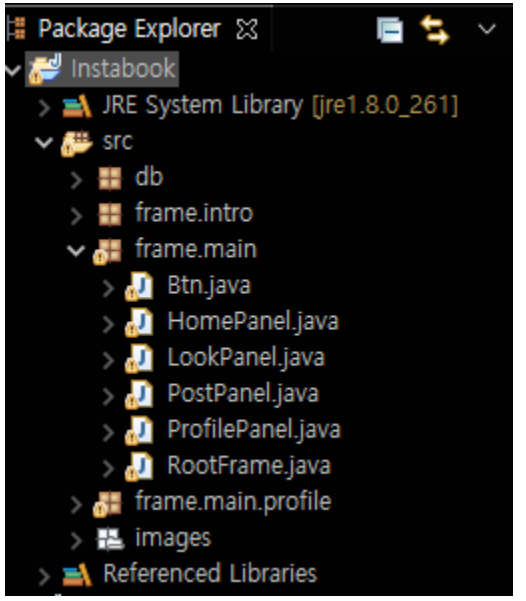
패키지 구성



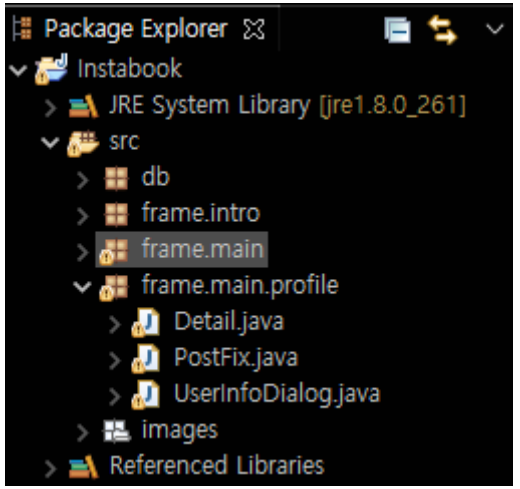
로그인 트리



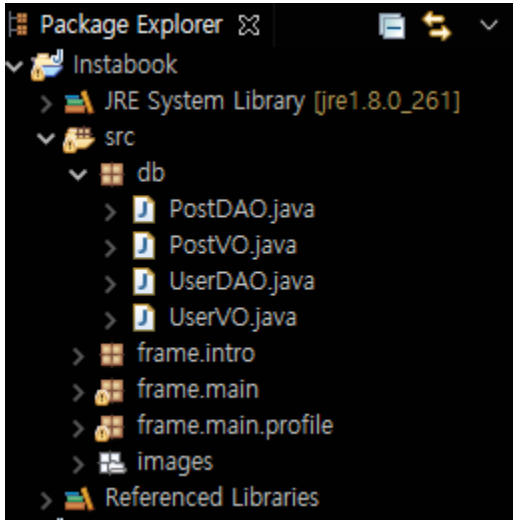
메인 게시판 트리



프로필 트리

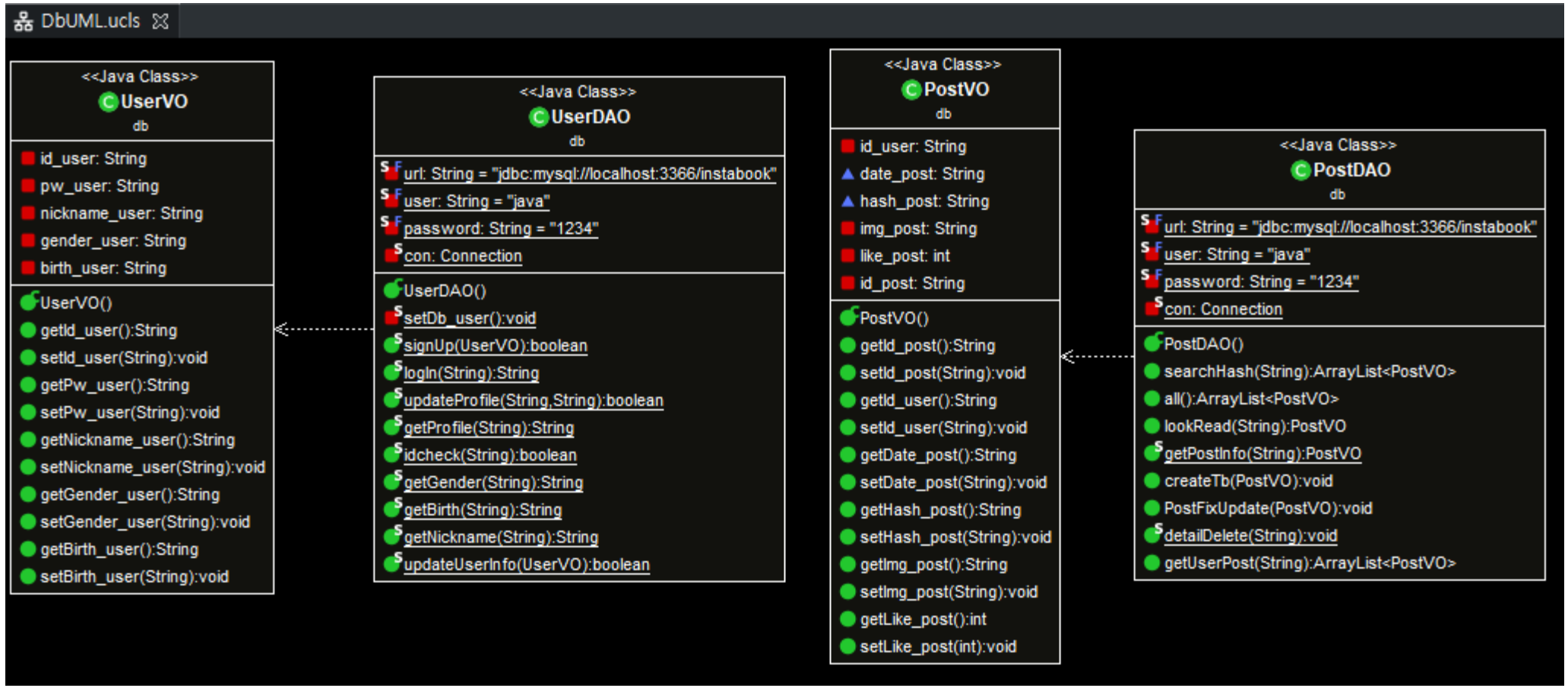


DB 트리



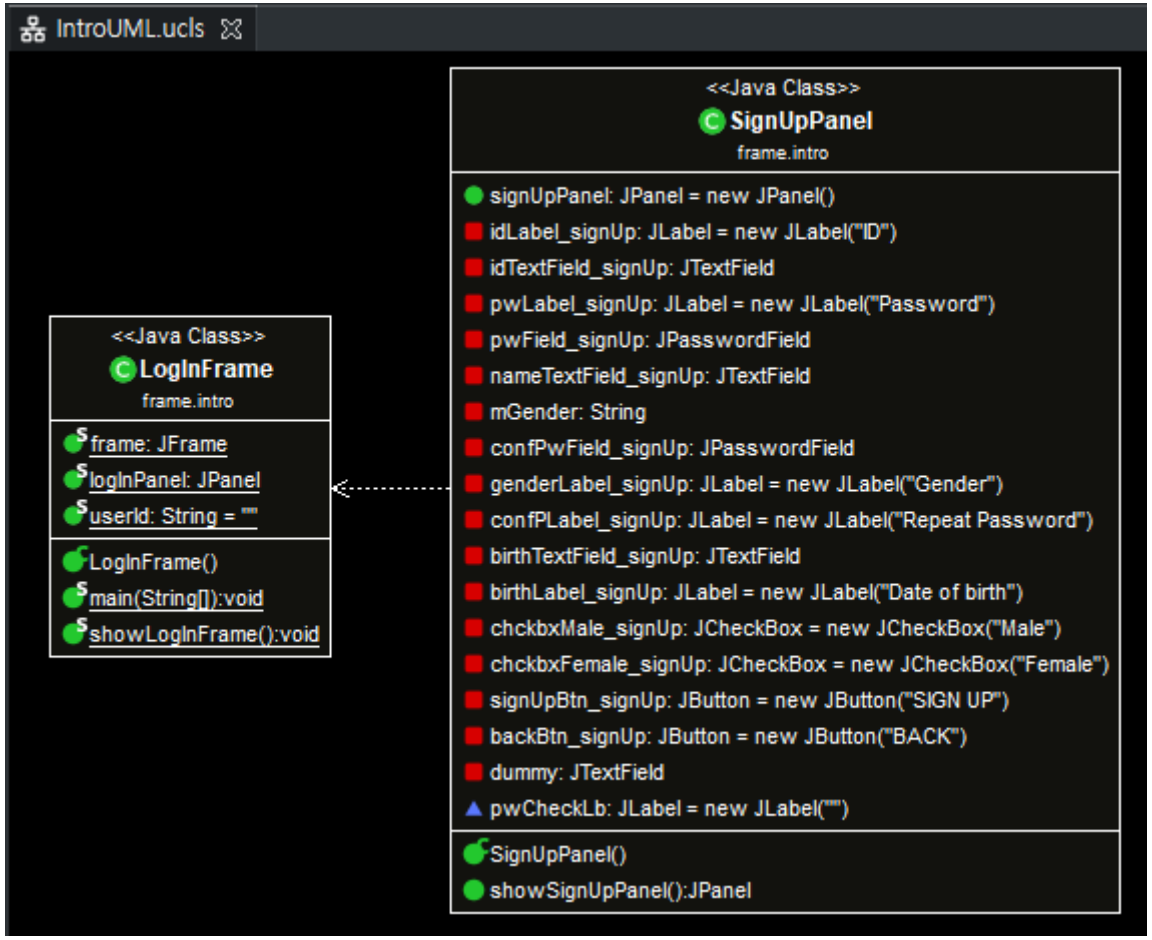
1) 개발 과정

1-5) UML / DAO , VO(DTO)



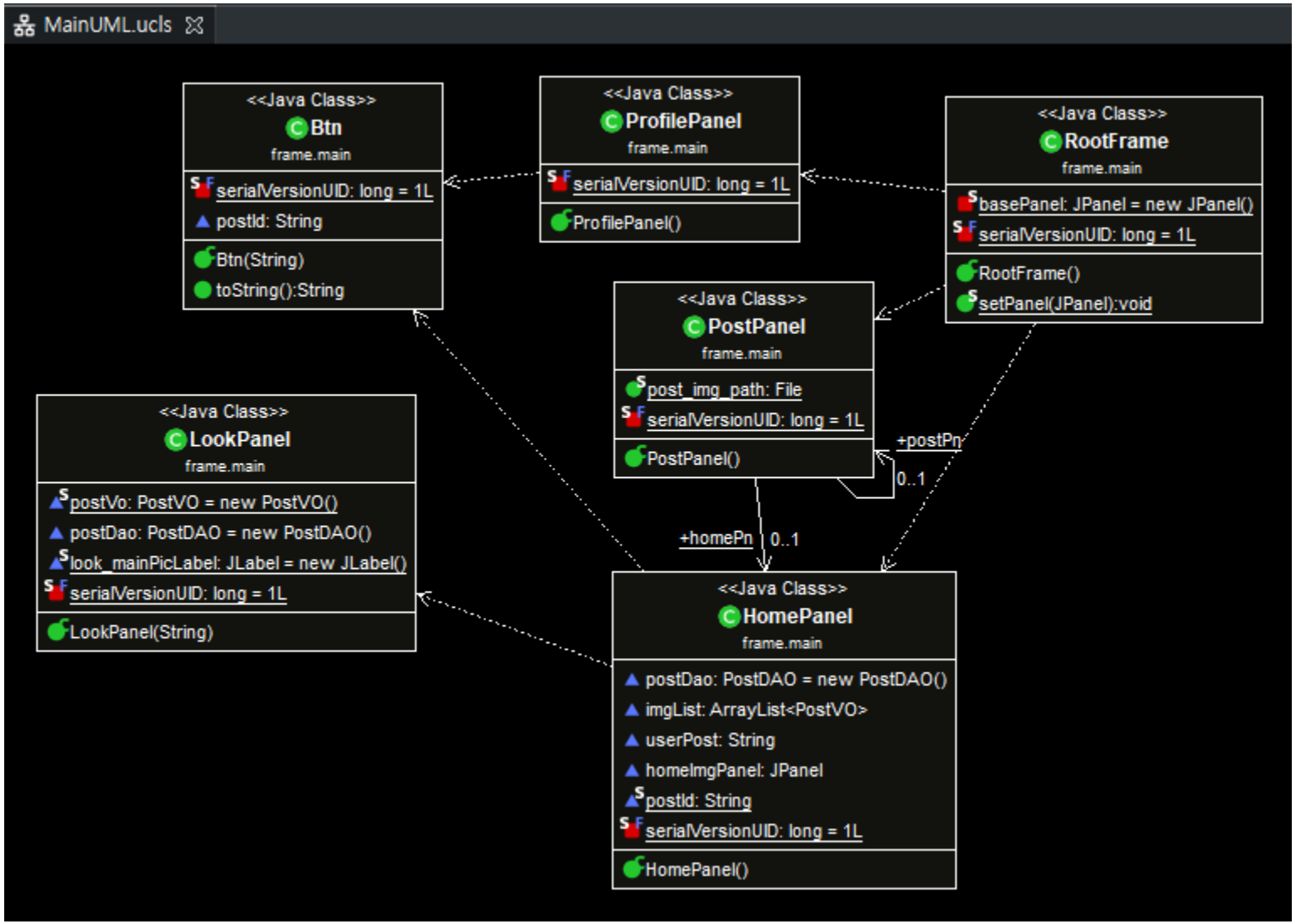
1) 개발 과정

1-5) UML / frame.main



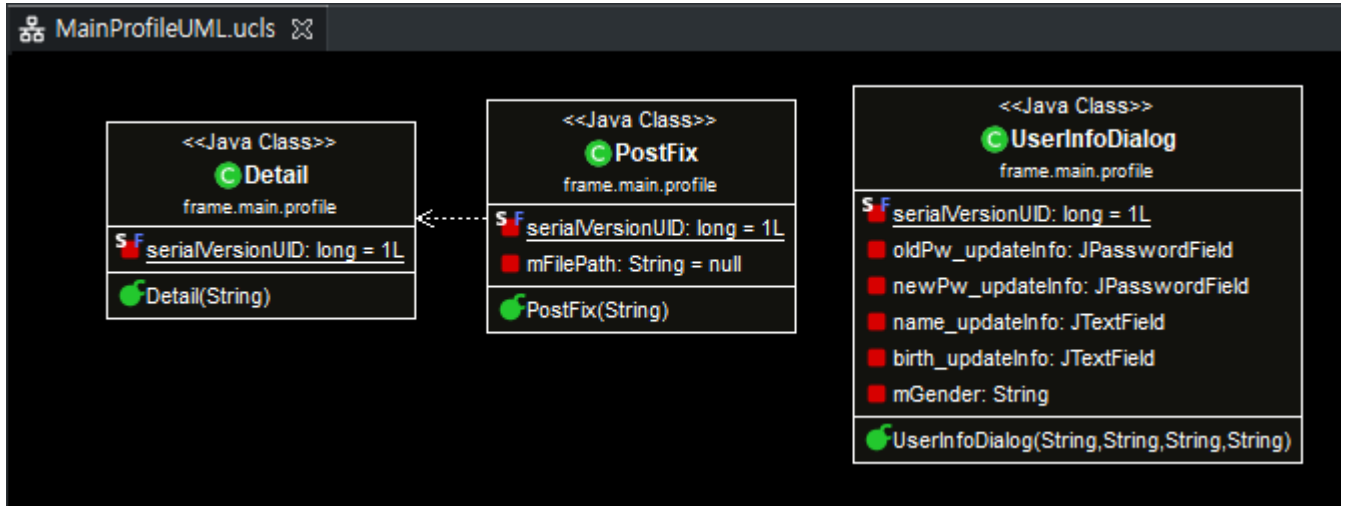
1) 개발 과정

1-5) UML / frame.intro



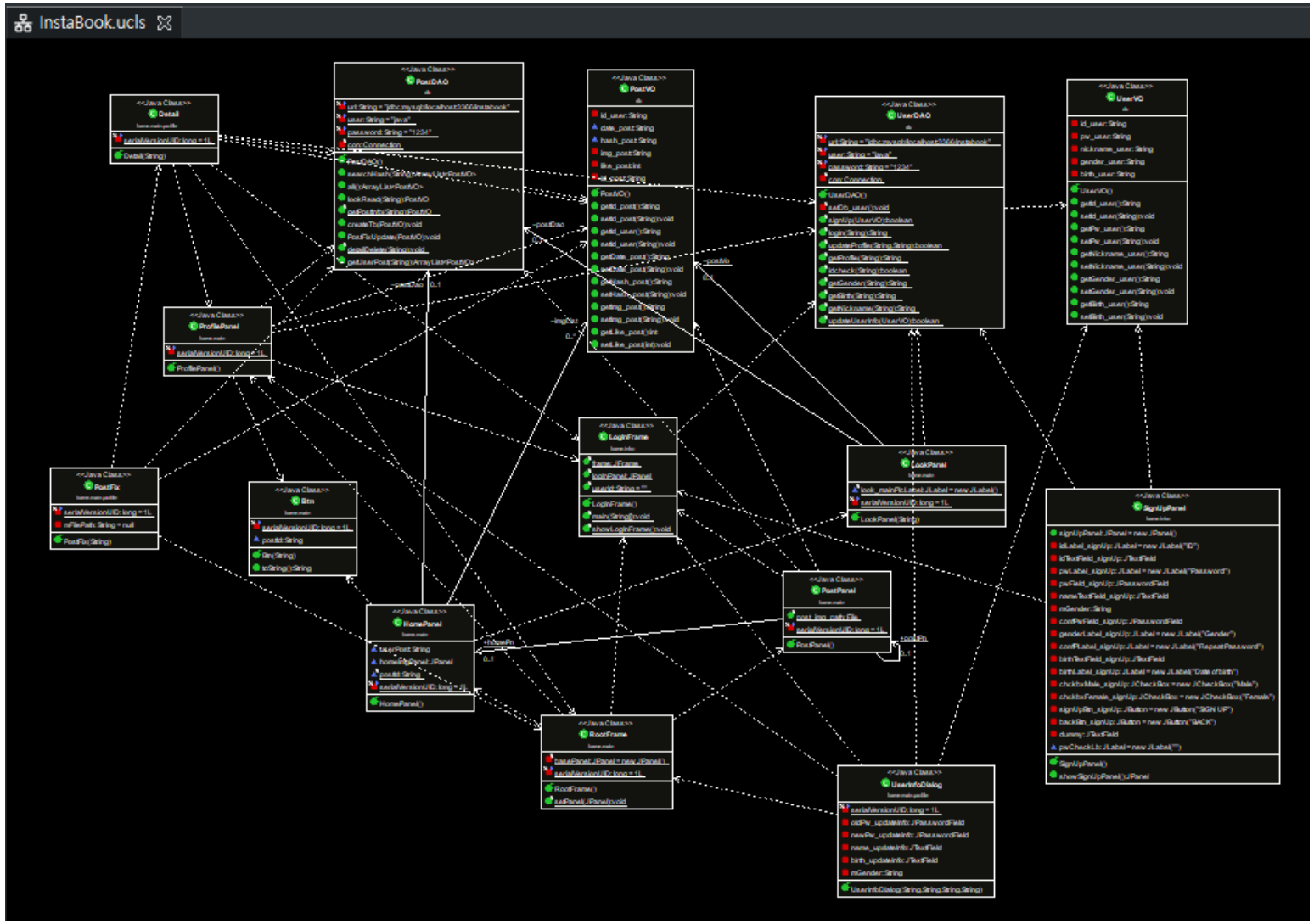
1) 개발 과정

1-5) UML / frame.main.profile



1) 개발 과정

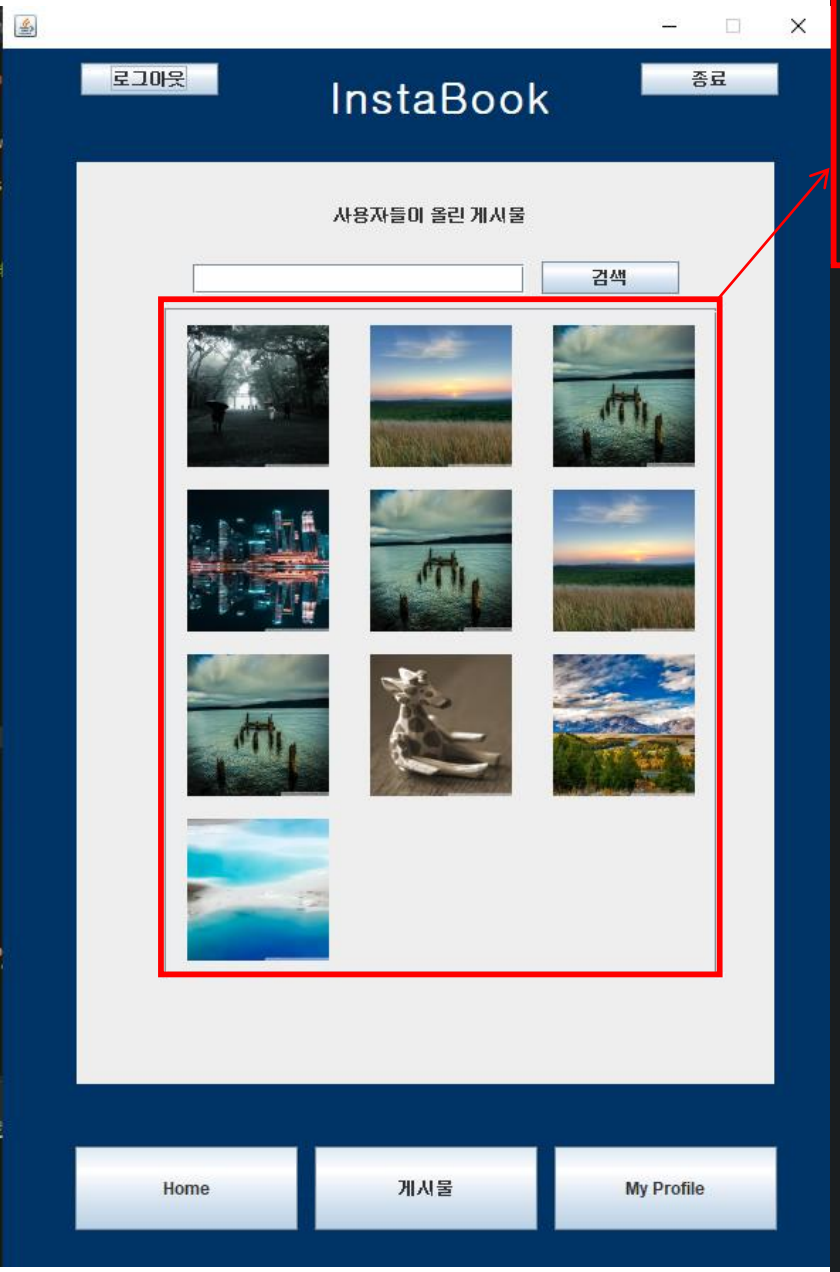
1-5) 종합 UML



2) 기술설명

2) 메인화면 CRUD

안은종




```
// 다른 사용자의 이미지 불러와서 보여주기
try {
    postDao = new PostDAO();
    imgList = postDao.all();

    for (int i = 0; i < imgList.size(); i++) {
        PostVO bag = imgList.get(i);

        Btn btn = new Btn(bag.getId_post());

        String homeImg = bag.getImg_post();
```

```
public ArrayList<PostVO> all() throws Exception {

    ArrayList<PostVO> list = new ArrayList<PostVO>();

    //1. connector설정
    Class.forName("com.mysql.jdbc.Driver");

    // 2. db연결
    con = DriverManager.getConnection(url, user, password);

    // 3. sql문을 만든다.
    String sql = "select * from post_insta";
    PreparedStatement ps = con.prepareStatement(sql);

    // 4. sql문은 전송
    // select의 결과는 검색결과가 담긴 테이블(항목+내용)
    // ResultSet: 데이터베이스의 데이터 집합을 나타내는 데이터 표로, 데이터베이스에 쿼리문을 실행하여 생성된다.
    // 복수의 데이터를 가져올때 사용
    ResultSet rs = ps.executeQuery();
    System.out.println("4. 홈 이미지 SQL문 전송 성공.!!");
    while (rs.next()) { // 결과가 있는지 없는지 체크해주는 메서드
        // 가방을 여러개 만들어야 해서 while문 안에 넣는다.
        PostVO bag = new PostVO();// 가방만들어서,
        String img_post = rs.getString("img_post");
        String id_user = rs.getString("id_user");
        String id_post = rs.getString("id_post");
        bag.setImg_post(img_post);
        bag.setId_user(id_user);
        bag.setId_post(id_post);
        // 컨테이너에 넣는다.
        list.add(bag);
    } // while
    return list;
}
```

```
// 다른 사용자의 게시물(이미지) 버튼
btn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        postId = btn.postId;

        // 다른 사용자의 게시물 보기
        LookPanel lookPn = new LookPanel(postId);
        RootFrame.setPanel(lookPn);
```

- HomePanel(메인화면) 다른 사용자의 이미지 DB에서 가져와서 출력
- PostDao의 값을 all()메서드로 컨터이너 처리를 하여 post_insta 테이블의 값을 하나씩 bag(가방)에 담아 list(컨테이너) 에 넣어서 HomePanel 반환 해준다.
- 그 반환된 값을 가지고 imgList에 넣어서 for문을 통해 list에 담긴 bag의 값에 해당 imgList.get(i) 의 인덱스 값에 만큼 데이터를 꺼내서
- Btn 클래스에 담겨있는 Id_post를 가져와 다시 btn에 담아서 버튼으로 반복 출력한다.
- All()메서드 에서 담겨져온 데이터를 look으로 넘기기

2) 기술설명

2) 다른 사용자 게시물 상세화면 CRUD

안은종


```
public LookPanel(String postId) {

    PostVO postVo = PostDAO.getPostInfo(postId);
    String userImg = UserDAO.getProfile(postVo.getId_user());
    String name = UserDAO.getNickname(postVo.getId_user());
    String postImg = postVo.getImg_post();
    int like = postVo.getLike_post();
    String hash = postVo.getHash_post();
    String date = postVo.getDate_post();
```

```
// look패널 id_post 기준으로 게시물의 모든 데이터 검색
public static PostVO getPostInfo(String postId) {
    PostVO bag = null;
```

```
String sql = "select * from post_insta where id_post =?";
ps = con.prepareStatement(sql);
ps.setString(1, postId);
System.out.println("3.searchHash sql 생성 성공.!!");
```

```
ResultSet rs;
try {
    rs = ps.executeQuery();
    System.out.println("4. sql문 전송 성공.!!");
    bag = new PostVO();// 3.묶음 값을 꺼내주기 위해 객체 생성.
    if (rs.next() == true) {
        System.out.println("게시물 검색결과가 있어요.");
        // 4.쿼리를 통해 DB에서 받은 값을
        // 5. MemberVO 클래스를 통해 가방에 넣기
        bag.setImg_post(rs.getString("img_post"));
        bag.setId_user(rs.getString("id_user"));
        bag.setDate_post(rs.getString("date_post"));
        bag.setHash_post(rs.getString("hash_post"));
        bag.setLike_post(rs.getInt("like_post"));
    } else {
        System.out.println("searchHash 검색결과가 없어요.");
    }
} catch (SQLException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}

return bag;
```

- HomePanel에서 LookPanel로 값을 넘겨 받은 postId에서 값을 하나씩 해당 라벨에 맞추어서 넣어주고

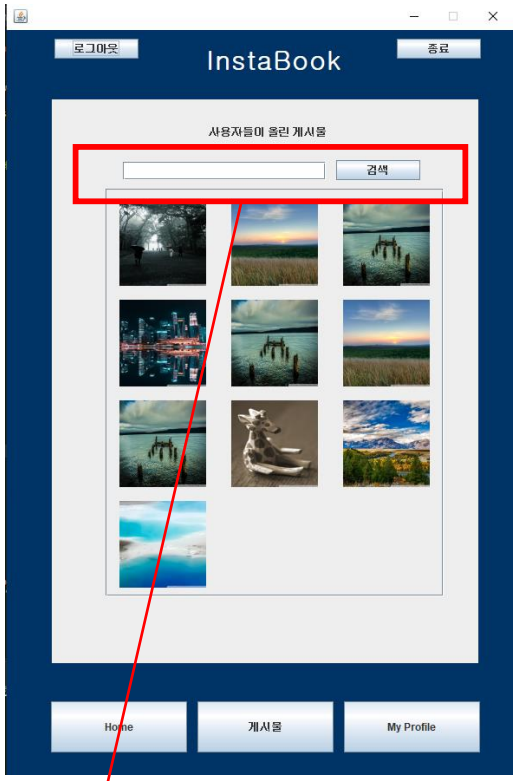
- getPostInfo메서드에서 값을 id_post기준으로 검색하여 해당 LookPanel에 set 적용하기 위해 값을 bag에 담아주고 return시켜서 넘겨준다.

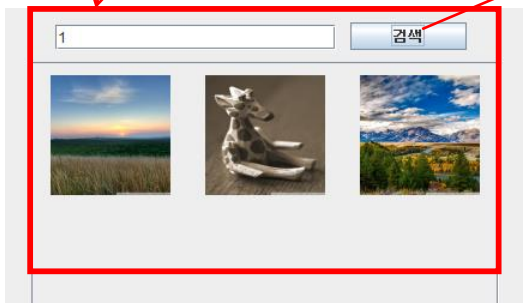
*그 넘겨준 bag값을 기준으로 다른 데이터에서 찾아서 쓸 수 있다.

2) 기술설명

2) 메인화면 검색 CRUD

안은종






```
try {
    ArrayList<PostVO> list = postDao.searchHash(hashTag);
    for (int i = 0; i < list.size(); i++) {// 1번째 for
        System.out.println("for문 인덱스 : " + i);
        PostVO bag = list.get(i);
        System.out.println("bag주소값: " + bag);

        Btn btn = new Btn(bag.getId_post());

        String homeImg = bag.getImg_post();
        System.out.println("검색시 이미지 경로 : " + homeImg);
```

```
// 다른 사용자의 게시물(이미지) 버튼
btn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        postId = btn.postId;
        // 다른 사용자의 게시물 보기
        LookPanel lookPn = new LookPanel(postId);
        RootFrame.setPanel(lookPn);

    }
});
btn.setIcon(iconCh);
// 스크롤패널 버튼적용
homeImgPanel.add(btn);
```

- 검색된 버튼을 누를시 LookPanel로 postId값 전달.

- 검색 클릭시 searchHash()메서드에서 검색창에 입력한 시작값을 기준으로 list에 묶음을 담아서 리턴, 리턴 받은 list을 for에 해당 크기의 size()만큼 반복 출력하여 btn에서 담은 값 가지고 이미지 버튼에 적용하여 출력

```
// 해쉬태그 전용 검색 메서드
public ArrayList<PostVO> searchHash(String hashTag) throws Exception {
    ArrayList<PostVO> list = new ArrayList<PostVO>();
    // 1. connector설정
    Class.forName("com.mysql.jdbc.Driver");
    System.out.println("1.searchHash connector 연결 성공.!!");

    con = DriverManager.getConnection(url, user, password);
    System.out.println("2.searchHash db 연결 성공.!!");

    String sql = "select * from post_insta where hash_post like '" + hashTag + "%'";
    PreparedStatement ps = con.prepareStatement(sql);
    System.out.println("3.searchHash sql 생성 성공.!!");

    ResultSet rs = ps.executeQuery();
    System.out.println("4. sql문 전송 성공.!!");
    while (rs.next()) {
        PostVO bag = new PostVO();// 3.묶음 값을 꺼내주기 위해 객체 생성.
        System.out.println("searchHash 검색결과가 있어요.");
        // 4.쿼리를 통해 DB에서 받은 값을
        String date_post = rs.getString("date_post");
        String hash_post = rs.getString("hash_post");
        String img_post = rs.getString("img_post");
        int like_post = rs.getInt("like_post");
        String userId = rs.getString("id_user");
        String postId = rs.getString("id_post");
        // 5. PostVO 클래스를 통해 가방에 넣기
        bag.setDate_post(date_post);
        bag.setHash_post(hash_post);
        bag.setImg_post(img_post);
        bag.setLike_post(like_post);
        bag.setId_user(userId);
        bag.setId_post(postId);

        list.add(bag);
    }
    // 6.bag은 참조형 변수에 넣은 값, 주소를 전달!
    return list;// 7.회원CURDUI에 전달.
```

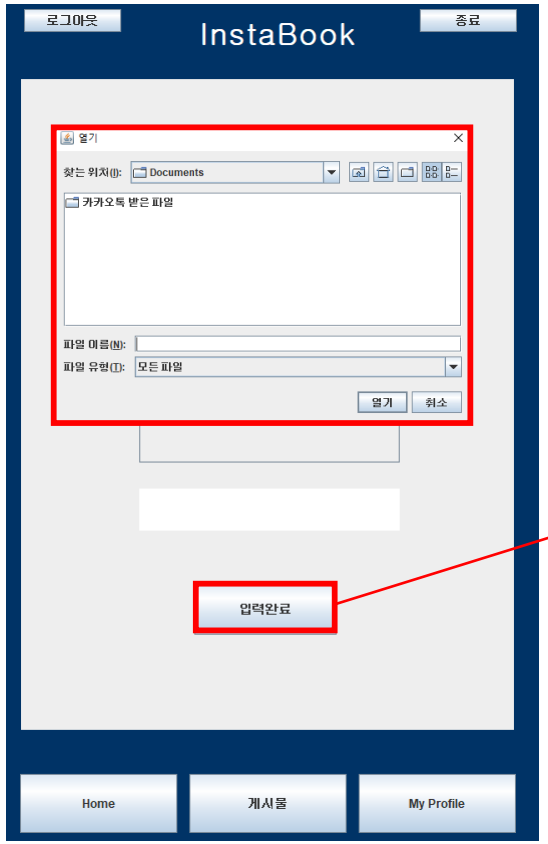
2) 기술설명

2) 게시물 등록화면 CRUD

안은종

DB에 등록할 아이디 확인

```
public PostPanel() {
    // 로그인 된 아이디 값 확인.
    String id_user = LogInFrame.userId;
```




```
post_mainPic_btn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        JFileChooser post_img_file = new JFileChooser();
        int img_file = post_img_file.showOpenDialog(post_mainPic_btn);
        if (img_file == JFileChooser.APPROVE_OPTION) {
            // 선택한 파일의 경로 반환
            post_img_path = post_img_file.getSelectedFile();
            // 경로 출력
            System.out.println(post_img_path);

            ImageIcon img = new ImageIcon(post_img_path.getPath());
            Image pic = img.getImage(); // ImageIcon을 Image로 변환.(객체를 돌려준다.)
            Image picCh = pic.getScaledInstance(265, 255, java.awt.Image.SCALE_SMOOTH);// 이미지 사이즈 조정
            ImageIcon iconCh = new ImageIcon(picCh); // Image로 ImageIcon 생성

            post_mainPic_btn.setIcon(iconCh);
            add(post_mainPic_btn);
```

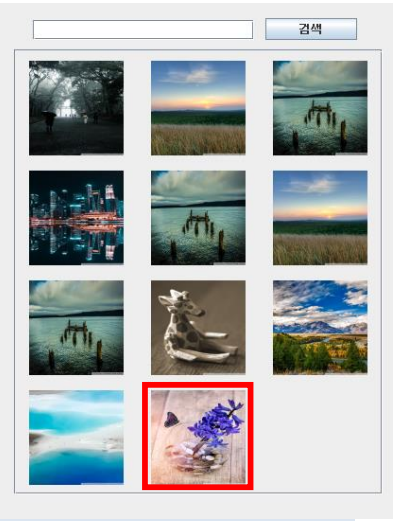
- 버튼 클릭시 파일을 열어서 이미지를 클릭시 그 경로의 값을 이미지로 반환하여 파일를 부른 버튼에 이미지 적용.

```
// post하기 위한 메서드
public void createTb(PostVO postSet) throws Exception {

    String id_user = postSet.getId_user();
    String date_post = postSet.getDate_post();
    String hash_post = postSet.getHash_post();
    String img_post = postSet.getImg_post();
    int like_post = postSet.getLike_post();

    // 1. connector 설정
    Class.forName("com.mysql.jdbc.Driver");
    con = DriverManager.getConnection(url, user, password);
    String sql = "insert into post_insta values (null,?,?,?,?,?)";// 어떤 값을 받을지 아직 모른다.
    PreparedStatement ps = con.prepareStatement(sql);
    ps.setString(1, id_user);
    ps.setString(2, date_post);
    ps.setString(3, hash_post);
    ps.setString(4, img_post);
    ps.setInt(5, like_post);

    ps.executeUpdate();
}
```



14	a	20.09.16	2	C:₩Users₩Administrator₩Desktop₩img₩11.jpg
17	a	20.09.16	test2	C:₩Users₩Administrator₩Desktop₩img₩2.jpg

*해당 id에 맞는 게시물 DB등록 확인.

2) 기술설명

2) 나의 게시물 모아보는 화면 CRUD

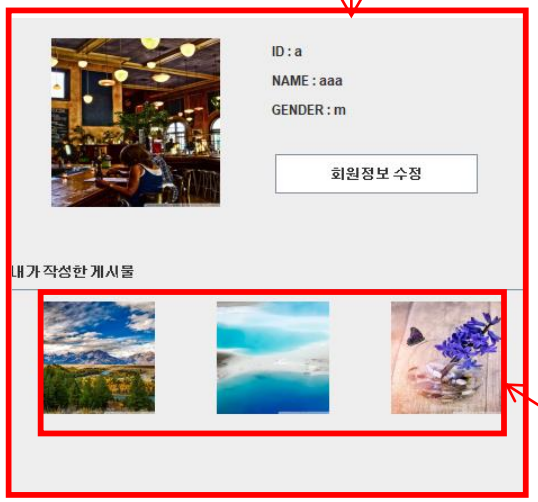
안은종

```
public Detail(String postId) {
    // 로그인 된 아이디 값 확인.
    String id_user = LogInFrame.userId;

    String name = UserDAO.getNickname(id_user);
    String profileImg = UserDAO.getProfile(id_user);

    PostVO postVo = null;
    postVo = PostDAO.getPostInfo(postId);
```

로그인 아이디를 확인하고 그 값을 기준으로
닉네임 및 프로필 이미지 UserDAO로 불러오기

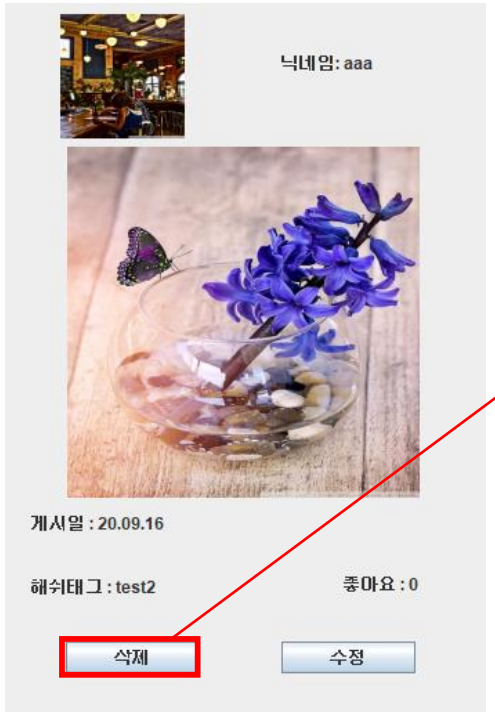


UserDAO로 가져온 프로필 적용.

```
// 프로필 이미지
JButton detail_header_pic_btn = new JButton();
ImageIcon img = new ImageIcon(profileImg);
```

postId 로 받아온 값을 postVO에 넣어 호출하여 버튼에 받아 반복 출력

```
// 게시물 이미지
ImageIcon mainimg = new ImageIcon(postVo.getImg_post());
Image mainpic = mainimg.getImage(); // ImageIcon을 Image로 변환.(객체를 돌려준다.)
```




```
// 삭제버튼
JButton deleteBtn = new JButton("삭제");
deleteBtn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        try {
            PostDAO.detailDelete(postId);
            ProfilePanel pp = new ProfilePanel();
            RootFrame.setPanel(pp);
        } catch (Exception e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }
```

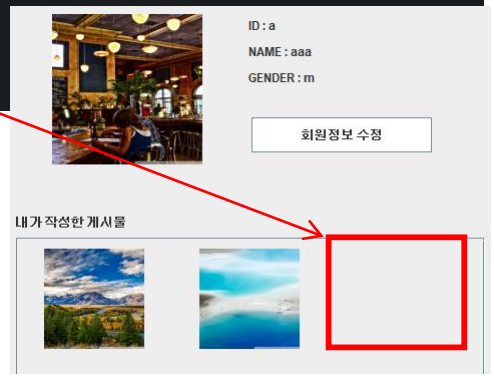
- 삭제 버튼 클릭시 detailDelete() 메서드로 profilePanel 에서 수정 사진 클릭시 받아온 postId로 DB에 가서 id_post에 맞는 값을 확인 후 삭제.

```
// db에서 삭제 처리 전담 메서드
public static void detailDelete(String id_post) throws Exception {
    // 디비 사용을 위해 driver class를 불러온다.
    // swing --- connector설정(driver) ---- mySQL
    Class.forName("com.mysql.jdbc.Driver");

    // db연결
    con = DriverManager.getConnection(url, user, password);

    // db에 있는 데이터를 삭제하기 위한 sql 명령문
    String sql = "delete from post_insta where id_post = ?";
    PreparedStatement ps = con.prepareStatement(sql);
    // UI에서 받은 파라미터값을 받기 위한 설정.
    ps.setString(1, id_post);

    // sql문 전송
    ps.executeUpdate();
}
```



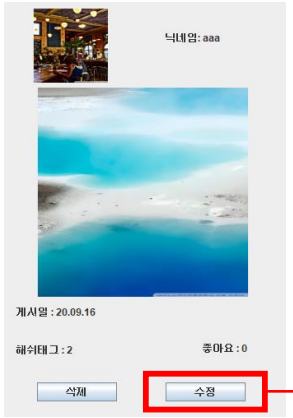
2) 기술설명

2) 나의 게시물 수정화면 CRUD

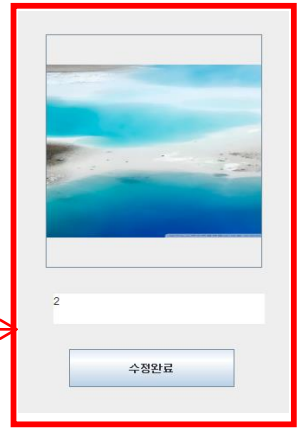
안은종

```
public PostFix(String postId) {

    // 패널에 보여줄 데이터가 담긴 VO 객체
    PostVO postVo = PostDAO.getPostInfo(postId);
    String img = postVo.getImg_post();
    String hash = postVo.getHash_post();
    mFilePath = PostDAO.getPostInfo(postId).getImg_post();
```



PostFix 화면



=


```
// 수정완료 버튼
JButton postFix_btn = new JButton("수정완료");
postFix_btn.setBounds(97, 420, 191, 44);
postFix_btn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
            String hash = postFix_hash_input.getText();
            PostDAO PostTb = new PostDAO();
            PostVO postSet = new PostVO();
            // userDAO의 id_user값 기준으로 임의값 입력
            postSet.setId_post(postId);
            // 오늘 날짜 계산 부품
            SimpleDateFormat sdf = new SimpleDateFormat("yy" + "." + "MM" + "." + "dd");
            Calendar c1 = Calendar.getInstance();
            String strToday = sdf.format(c1.getTime());
            postSet.setDate_post(strToday);
            postSet.setImg_post(mFilePath);
            postSet.setHash_post(hash);

            PostTb.PostFixUpdate(postSet);

            // 이전(Detail) 화면으로 이동
            Detail detailPn = new Detail(postSet.getId_post());
            RootFrame.setPanel(detailPn);
        } catch (Exception e1) {
            e1.printStackTrace();
        }
```

```
public void actionPerformed(ActionEvent e) {
    JFileChooser postFix_img_file = new JFileChooser();
    int img_file = postFix_img_file.showOpenDialog(imgUpdateBtn);
    if (img_file == JFileChooser.APPROVE_OPTION) {
        // 선택한 파일의 경로 반환
        File postFix_img_path = postFix_img_file.getSelectedFile();
        mFilePath = postFix_img_path.getPath();// 수정완료 클릭시 파일 경로 얻기위해 저장
        ImageIcon img = new ImageIcon(postFix_img_path.getPath());
        Image pic = img.getImage(); // ImageIcon을 Image로 변환.(객체를 돌려준다.)
        Image picCh = pic.getScaledInstance(270, 270, java.awt.Image.SCALE_SMOOTH);// 이미지 사이즈 조정
        ImageIcon iconCh = new ImageIcon(picCh); // Image로 ImageIcon 생성
        imgUpdateBtn.setIcon(iconCh);
    }
```

- Detail패널에서 받아온 postId값으로 수정할 이미지 값을 불러와주고, 수정할 이미지 파일에서 불러와서 이미지 변환하여 btn에 추가, hashTag 입력란에 수정할 텍스트 입력 수정입력 클릭시 DB로 전송 데이터 적용.

*수정된 DB 값과 화면

13	a	20.09.16	1	C:₩Users₩Administrator₩Desktop₩img₩10.jpg
14	a	20.09.16	img4	C:₩Users₩Administrator₩Desktop₩img₩4.jpg

감사합니다.