

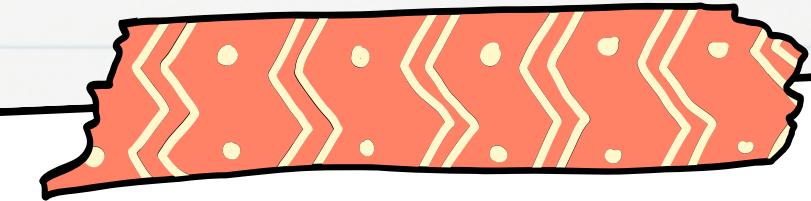
휴먼문고의 구매와 관리

Presented by: 이 찬

Overview

- 프로젝트 소개
- 유스케이스
- 요구사항
- ERD
- 클래스 다이어그램
- 기능 관련 코드 증
빙 및 해설
- 후기





프로젝트 소개

도서 구매 및 관리 프로그램

프로젝트 설명

- 종이책 / e-book 으로 나뉘어서 구매 및 관리 구현
- 비회원, 회원, 관리자로 등급을 나누고, 등급에 따라 기능을 달리하는 프로그램
- 데이터베이스와 연동하여 CRUD를 통해 관리
- **프로젝트 목표**
- 쉬운 사용법으로 어려움 없는 도서 구매 및 관리



개발 도구



데이터베이스
오라클 11g



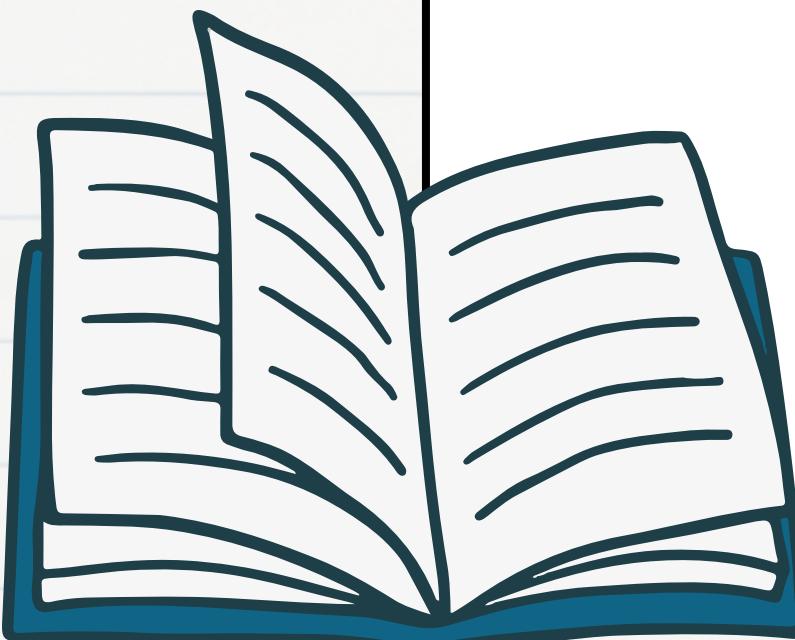
DB 관리 도구
DBeaver 버전
24.2.2



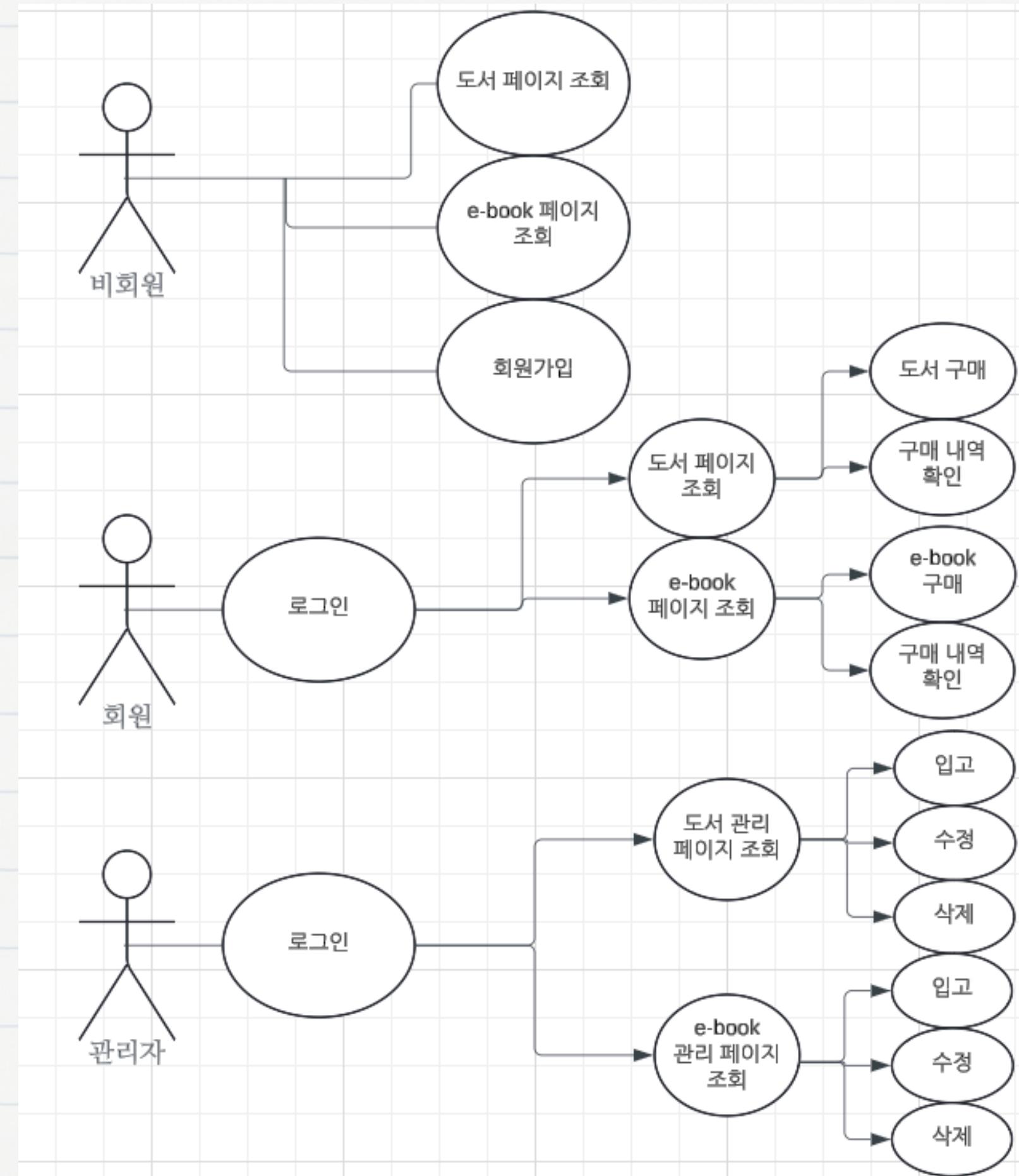
사용 언어
자바 버전 8 / 1.8



통합개발환경
이클립스 버전
4.32



유스케이스 다이어그램



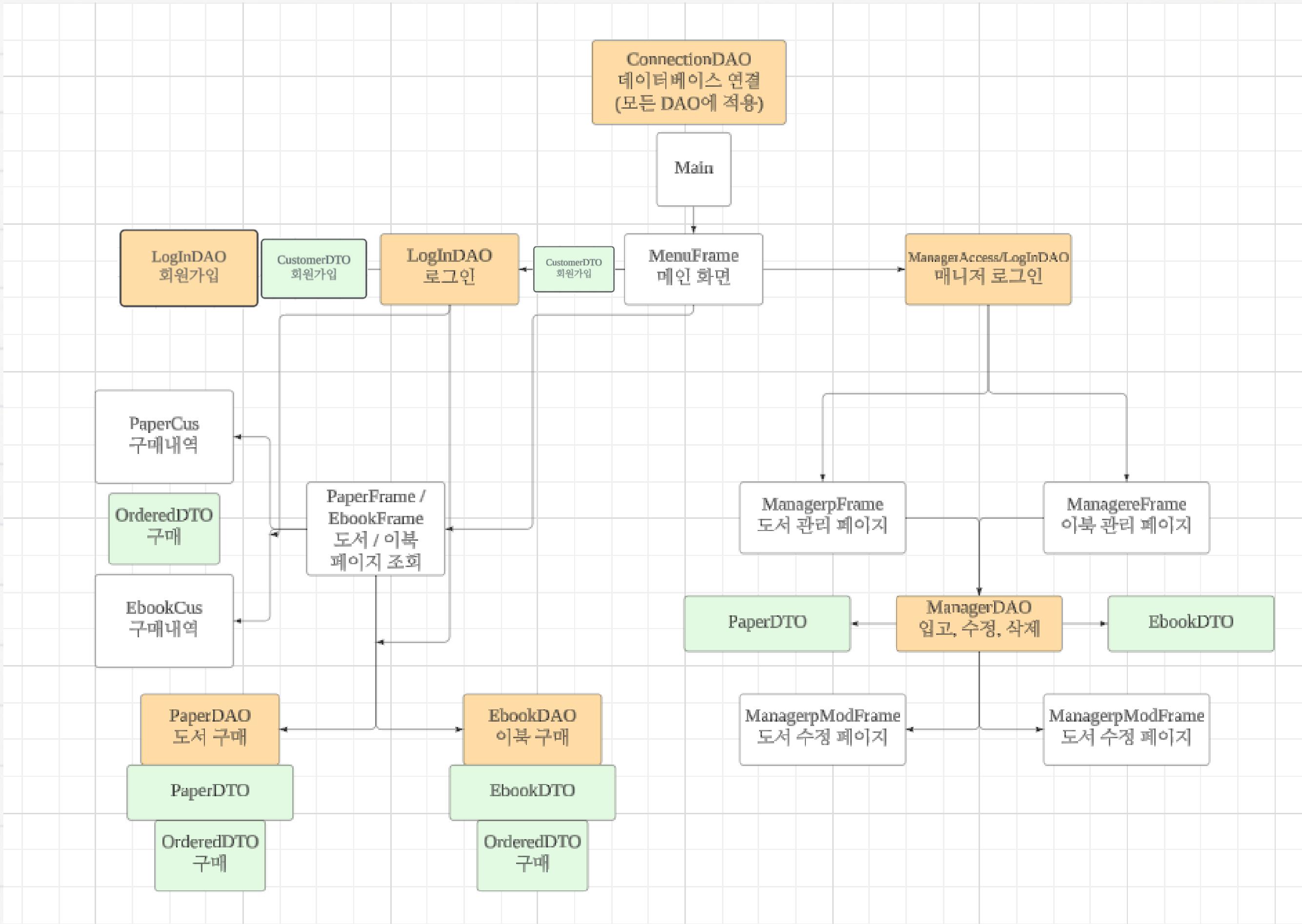
요구사항 정의서

요구사항 정의서	
요구사항 명	요구사항 내용
계급별 서비스 제공	1. 비회원, 회원, 관리자는 각자의 고유의 기능이 있고 이 기능들만 사용 가능하다.
비회원 기능	1. 비회원은 도서 및 e-book 페이지 및 책 리스트까지만 조회 가능하다. 2. 구매 및 자신의 구매목록 확인을 하려면 로그인을 해야 한다.
회원가입	1. 아이디, 이름, 비밀번호, 질문, 답변을 입력받는다. 2. 아이디는 중복되지 않아야 한다. 3. 비밀번호는 8~16글자이며 영어, 숫자, 특수문자가 전부 포함되어야 한다. 4. 비밀번호는 암호화되어 눈으로 식별할 수 없다. 5. 모든 정보를 입력해야만 가입이 완료된다.
로그인	1. 아이디와 비밀번호를 입력하여 로그인한다. 비밀번호 칸은 암호화되어 있다.
회원 기능	1. 리스트 위의 텍스트필드를 통해 원하는 책을 검색할 수 있다. 2. 리스트에서 도서를 선택, 원하는 만큼의 개수를 선택한 뒤 구매 가능하다. 3. 사용자의 구매 내역을 버튼을 눌러 확인할 수 있다.
구매 기능	1. 종이책은 재고 이상의 개수를 입력하면 경고 메시지가 뜨며 구매 할 수 없다. 2. 리스트를 클릭하면 해당 도서의 제목이 구매창에 표시된다.
구매 내역	1. 도서와 e-book, 각 페이지마다 구매 내역을 확인할 수 있다. 2. 각 구매 내역 버튼은 해당하는 종류(종이책은 종이책만, e-book은 e-book만)의 구매내역을 확인 가능하다. 3. 구매 내역에는 해당 회원의 구매내역만 표시된다. 4. 구매 내역에는 개수를 포함한 총 가격 또한 표시된다.
관리자 로그인	1. 관리자 로그인은 정해진 코드를 로그인 창에 입력해야만 로그인이 가능하다. 2. 관리자로 로그인하면 도서 관리, e-book관리, 로그아웃 창이 나오며 관리 창들을 누르면 각 독립된 창들이 나타난다.
관리자 CRUD	1. 관리자는 도서 명, 작가 명, 가격 및 재고를 입력하여 입고한다. 2. 리스트의 도서를 선택한 뒤 수정을 클릭하면 수정 전용 창으로 이동, 여기서 정보를 입력한 뒤 수정하기를 누르면 해당 도서의 정보는 수정된다. 3. 리스트의 도서를 선택한 뒤 삭제를 누르면 해당 도서는 삭제된다.

ERD



클래스 다이어그램



개발일정

주요 기능 관련 코드

증빙 및 해설



코드 증빙 및 해설 - 데이터베이스 연결

Connection 의 싱글톤화

구현 기술: Singleton 패턴 / JDBC API / 예외 처리

```
private String username = "system";
private String password = "11111111";
private String url = "jdbc:oracle:thin:@localhost:1521:orcl";
private String driverName = "oracle.jdbc.driver.OracleDriver";
public Connection conn = null;

public static ConnectionDAO cdao = null;

public static ConnectionDAO getInstance() {
    if(cdao == null) {
        cdao = new ConnectionDAO();
    }
    return cdao;
}
public void init() {
    try {
        Class.forName(driverName);
        System.out.println("오라클 드라이버 로드 성공");
    }
    public boolean conn() {
        try {
            conn = DriverManager.getConnection(url,username,password);
            System.out.println("커넥션 자원 획득 성공");
            return true;
        }
```

```
if(cdao.conn()) {
    try {
        String sql = "insert into customer values (?,?,?,?,?)";
        PreparedStatement psmt = cdao.conn.prepareStatement(sql);
        psmt.setString(1, cd.getCid());
        psmt.setString(2, cd.getCname());
        psmt.setString(3, cd.getCPassword());
        psmt.setString(4, cd.getQuestion());
        psmt.setString(5, cd.getAnswer());
        int resultInt = psmt.executeUpdate();
        if(resultInt == 0) {
            cdao.conn.rollback();
        }else {
            cdao.conn.commit();
        }
    } catch(Exception e) {
        e.printStackTrace();
    } finally {
        if(cdao.conn != null) {
            try {
                cdao.conn.close();
            } catch(Exception e) {
```

Connection의 인스턴스는 단 하나만 존재하게 된다

코드 증빙 및 해설 - 회원가입

MenuFrame > signup(J버튼) > duplicate(J버튼) > signupcomplete(J버튼)

구현 기술: Interface ActionListener / JPasswordField / JComboBox / 정규식 RegPattern / JFrame / Timer

MenuFrame뿐만 아닌 모든 ~Frame 클래스는 JFrame을 상속받고, 이는 GUI 애플리케이션에서 윈도우를 만드는데 사용된다.

MenuFrame은 ActionListener와 MouseListener 인터페이스를 구현하였다.

signup 버튼을 클릭하면 actionPerformed 메서드가 실행된다.

로그인 시 비밀번호 패턴은 정규식을 통해 지정되었다.
정규식: 8~16글자; 영어, 숫자, 특수문자 사용

나타나는 메세지들은 모두 Timer를 통해 3~5초 이후에 사라진다.

```
public class MenuFrame extends JFrame implements ActionListener, MouseListener, KeyListener {
    ...
    if(e.getSource() == signup) {
        if(e.getSource() == duplicate) {
            String sid = stextid.getText();
            boolean isDuplicate = ldboa.duplicateId(sid);

            String message;
            if(isDuplicate) {
                message = "중복된 아이디입니다. 다시 입력하세요.";
                contentPane.remove(signupcomplete);
            } else {
                message = "해당 아이디는 사용 가능합니다.";
                signupcomplete.setBounds(270, 350, 150, 25);
                contentPane.add(signupcomplete);
                signupcomplete.addActionListener(this);
            }
        }
    }
}
```

```
PRIVATE STRING PW_PATTERN = "^(?=.*[A-Z])(?=.*\\D)(?=.*[!@#$%^&*()+=])[A-Z\\D~!@#$%^&*()+=]{8,16}$";

if(e.getSource() == signupcomplete) {
    String sid = stextid.getText();
    String sname = stextName.getText();
    String spw = String.valueOf(stextpw.getPassword());
    String sqt = String.valueOf(questiontext.getSelectedItem());
    String sanwer = answertext.getText();
    if (!spw.matches(pwpattern)) {
        JLabel errorLabel = new JLabel("비밀번호는 8~16자, 숫자, 문자, 특수문자 포함해야 합니다.");
    }
}
```

코드 증빙 및 해설 - 회원가입

MenuFrame > signup(J버튼) > duplicate(J버튼) > signupcomplete(J버튼)

CustomerDTO / LogInDAO

아이디 중복 체크는 LogInDAO의 duplicateId 메서드를 통해 진행된다. 해당 메서드는 MenuFrame의 duplicate 버튼을 클릭하면 진행된다.

정규식과 일치하는 등 모든 조건을 만족하면 회원가입이 진행된다.

CustomerDTO객체 생성, 회원 정보를 설정한 뒤 LogInDAO를 호출하여 해당 객체의 signUp메서드에 당장의 CustomerDTO객체 정보를 전달한다. 그러면 SignUp메서드에서 회원가입을 완료한다.

```

public boolean duplicateId(String id) {
    boolean dup = true;
    if(cdao.conn()) {
        try {
            String sql = "select count(*) from customer where cid = ?";
            PreparedStatement psmt = cdao.conn.prepareStatement(sql);
            psmt.setString(1, id);
            ResultSet rs = psmt.executeQuery();

            if(rs.next() && rs.getInt(1) > 0) {
                System.out.println("중복된 아이디입니다. 다시 입력하세요.");
                dup = true;
            } else {
                System.out.println("해당 아이디를 사용합니다.");
                dup = false;
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    } return dup;
}

cdto = new CustomerDTO();
cdto.setCid(sid);
cdto.setCname(sname);
cdto.setCpassword(spw);
cdto.setCquestion(sqt);
cdto.setCanswer(sanwer);
ldboao.signUp(cdto);

initializeMenuFrame();
setBounds(100,100,1080,720);

```

```

if(e.getSource() == duplicate) {
    String sid = stextid.getText();
    boolean isDuplicate = ldbao.duplicateId(sid);

    String message;
    if(isDuplicate) {
        message = "중복된 아이디입니다. 다시 입력하세요.";
        contentPane.remove(signupcomplete);
    } else {
        message = "해당 아이디는 사용 가능합니다.";
        signupcomplete.setBounds(270, 350, 150, 25);
        contentPane.add(signupcomplete);
        signupcomplete.addActionListener(this);
    }
}

```

```

if(cdao.conn()) {
    try {
        String sql = "insert into customer values (?,?,?,?,?)";
        PreparedStatement psmt = cdao.conn.prepareStatement(sql);
        psmt.setString(1, cd.getCid());
        psmt.setString(2, cd.getCname());
        psmt.setString(3, cd.getCPassword());
        psmt.setString(4, cd.getQuestion());
        psmt.setString(5, cd.getAnswer());
        int resultInt = psmt.executeUpdate();
    }
}

```

코드 증빙 및 해설 - 도서/e-book 페이지 조회

MenuFrame > paperLabel(그림) > PaperFrame(JFrame)

구현 기술: Interface MouseListener / CollectionFrameWork ArrayList/
DefaultTableModel / JTable/ScrollPane / ScrollBar / Timer

```
ImageIcon originalpaper = new ImageIcon(getClass().getResource("/images/usedbook.png"));
Image paperbook = originalpaper.getImage();

Image scaledpaper = paperbook.getScaledInstance(200, 200, Image.SCALE_SMOOTH);
ImageIcon scaledp = new ImageIcon(scaledpaper);

JLabel paperLabel = new JLabel(scaledp);

ImageIcon originalebook = new ImageIcon(getClass().getResource("/images/ebook.png"));
Image ebook = originalebook.getImage();

Image scaledebook = ebook.getScaledInstance(200, 200, Image.SCALE_SMOOTH);
ImageIcon scalede = new ImageIcon(scaledebook);

JLabel ebookLabel = new JLabel(scalede);

if(e.getSource() == buybtn && lg.isloggedIn() == true)
if(e.getSource() == mypbtn && realid != null) {
```

위의 코드를 보시다시피, 책 구매(buybtn) 및 구매 내역
(mypbtn) 보기는 로그인이 되어있지 않다면 사용 불가
능하다.

MenuFrame > ebookLabel(그림) > EbookFrame(JFrame)

MouseClicked을 통해 이미지를 클릭해서 새로운 JFrame을 인스턴스화

```
public void mouseClicked(MouseEvent e) {
    if(e.getSource() == paperLabel) {
        PaperFrame2 pframe = new PaperFrame2(pdao, odto, this, lg, pc);
        pframe.setVisible(true);
    }
    if(e.getSource() == ebookLabel) {
        EbookFrame eframe = new EbookFrame(edao, odto, this, lg, ec);
        eframe.setVisible(true);
```

비회원은 아래의 searchbtn(책 검색)까지만 사용이 가능

```
if(e.getSource() == searchbtn) {
    String namep = null;
    namep = searchText.getText();
    pbao.oneSelect(namep);
    ArrayList<PaperDTO> results = pbao.oneSelect(namep);
    porderedList.setRowCount(0);
    for (PaperDTO paper : results) {
        porderedList.addRow(new Object[] { paper.getPcode(), paper.getPname(),
                                         paper.getPAuthor(), paper.getPPrice(), paper.getPQuantity() });
    }
    contentPane.revalidate();
    contentPane.repaint();
}
```

코드 증빙 및 해설 - 도서 구매

MenuFrame > login(J버튼) > PaperFrame > buybtn / mypbtn

구현 기술: Interface ActionListener / MouseListener / JSpinner / Logginginterface / ArrayList / OrderedDTO / Timer

리스트의 도서를 클릭하면, 해당 도서의 코드를 pcode,
재고를 inventory라는 전역 변수에 저장한다.

해당 전역 변수를 이용하여, 만약 내가 입력한 구매량이 재고보다 높
으면 재고가 부족하다는 메시지가 뜨고, 아니라면 데이터베이스의
ordered 테이블에 저장이 된다.

구매량*가격으로 총 가격을 구한다

```
if(e.getSource() instanceof JTable) {
    JTable table = (JTable) e.getSource();
    int selrow = table.getSelectedRow();
    pcode = String.valueOf(table.getValueAt(selrow, 0));
    inventory = Integer.parseInt(String.valueOf(table.getValueAt(selrow, 4)));
    // str = str.substring(0, str.length() - 1);
    String pname = String.valueOf(table.getValueAt(selrow, 1));
    textpname.setText(pname);

if(e.getSource() == buybtn && lg.isloggedIn() == true) {
    String pcode2 = pcode;
    String pname = textpname.getText();
    int quantity = Integer.parseInt(String.valueOf(textpquantity.getValue()));
    if(inventory<quantity) {
        howmuch.setText("재고가 부족합니다.");
    } else {
        odto = new OrderedDTO();
        odto.setCid(mf.getRealid());
        odto.setPcode(pcode2);
        odto.setBname(pname);
        odto.setQuantity(quantity);
        pbao.insert(odto);
        howmuch.setText("총 " + odto.getQuantity() + "권 구매하셨습니다.");
    }
    contentPane.revalidate();
    contentPane.repaint();

String sql =
"insert into ordered values ('o p' || ordered_seq.NEXTVAL, ?,?,?,?,?,"
+ "(SELECT pname FROM paperbook WHERE pcode = ?),?,?"
+ "(SELECT pprice FROM paperbook where pcode = ?)*?)";
```

코드 증빙 및 해설 - 구매 내역 보기

MenuFrame > login(J버튼) > PaperFrame > mybtn

로그인 상태에서 구매 내역을 클릭하면 PaperCus 객체가 생성(로그
인하지 않는다면 경고메세지만 표시된다)

저장 된 전역 변수에 입력되어있는 아이디에 해당하는 주문건수만 회
원의 구매 내역으로 가져온다.

for 루프를 통해 각 주문(order)을 순회하면서, mypage라는 데이터
모델에 주문 정보를 행(row) 형태로 추가, 이후 이 데이터 모델을 기
반으로 한 mytable이라는 주소를 가진 JTable을 생성한다.

JScrollPane을 이용하여 테이블 스크롤 또한 가능하다.

```
if(e.getSource() == mybtn && realid != null) {
    pc = new PaperCus(pbao, odto, this);
    pc.setVisible(true);
} else if(e.getSource() == mybtn && realid == null) {
    JOptionPane.showMessageDialog(this, "로그인 후 이용하세요.", "경고", JOptionPane.WARNING_MESSAGE);
    return;
}

mypage = new DefaultTableModel(
    new Object[][] {},
    new String[] {"주문 번호", "아이디", "도서 코드", "도서 제목", "주문 수량", "총 가격"}
);

ArrayList<OrderedDTO> orders = pdao.selectOne(cid);
for (OrderedDTO order : orders) {
    mypage.addRow(new Object[]{
        order.getTid(),
        order.getCid(),
        order.getPcode(),
        order.getBname(),
        order.getQuantity(),
        order.getFullprice()+"원"
    });
}
mytable = new JTable(mypage);
JScrollPane scrollPane = new JScrollPane(mytable);
contentPane.add(scrollPane, BorderLayout.CENTER);

setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

"select * from ordered where cid = ? AND pcode is not null";
```

코드 증빙 및 해설 - 관리자 로그인

MenuFrame > managerLogin

구현 기술: Interface ActionListener / ManagerAccess / Timer

매니저 코드를 입력하는 TextField에서 입력된 코드를 가져온다.

getMcode 메서드를 통해 해당 코드가 ManagerAccess의 mcode
와 같다면 매니저 로그인이 진행된다.

로그인이 완료되면, 로그인 TextField와 버튼은 없고
종이책 관리 화면 / 이북 관리 화면의 주소가 있는 menumanager,
menuemanager가 표시되는 UI로 업데이트 된다.

표시되는 안내 메세지들은 너무 Timer에 의해 3초 후에 사라진다.

```
if(e.getSource() == managerLogin) {  
    String mcode = String.valueOf(textmanager.getPassword());  
    if(mcode.equals(ma.getMcode())) {  
        lg.managerIn();  
        System.out.println("관리자 계정으로 로그인하였습니다.");  
        ismanager = new JLabel("관리자 계정으로 로그인하였습니다.");  
        managerLogin.setVisible(false);  
        textmanager.setVisible(false);  
        menumanager.setVisible(true);  
        menuemanager.setVisible(true);  
        notmanager.setVisible(true);  
    } else {  
        lg.managerOut();  
        System.out.println("관리자 계정 코드가 틀렸습니다.");  
        ismanager = new JLabel("관리자 코드가 틀렸습니다.");  
    }  
    ismanager.setFont(new Font("한국외대체 L", Font.PLAIN, 14));  
    ismanager.setBounds(472, 525, 300, 21);  
    contentPane.add(ismanager);  
    contentPane.revalidate();  
    contentPane.repaint();  
    Timer timer = new Timer(3000, sec3 -> {  
        contentPane.remove(ismanager);  
        contentPane.revalidate();  
        contentPane.repaint();  
    });  
    timer.setRepeats(false);  
    timer.start();  
}
```

코드 증빙 및 해설 - 도서 관리 메뉴 / 이북 관리 메뉴 열기

MenuFrame > managerLogin

구현 기술: Interface ActionListener

```
if(e.getSource() == menumanager) {  
    ManagerpFrame mpframe = new ManagerpFrame(ldbao, lg, pdao, edao, mdao, this, mmframe);  
    mpframe.setVisible(true);  
}  
if(e.getSource() == menumanager) {  
    ManagereFrame meframe = new ManagereFrame(ldbo, lg, pdao, edao, mdao, this, mmeframe);  
    meframe.setVisible(true);  
}
```

사용자가 menumanager 메뉴 항목을 클릭하면, ManagerpFrame 클래스의 인스턴스를 생성하고, 여러 데이터 접근 객체(dao)를 포함하여 mpframe이라는 이름으로 프레임이 만들어진다.

이 코드는 사용자가 선택한 버튼에 따라 서로 다른 프레임을 열어주는 기능을 수행한다.

코드 증빙 및 해설 - 관리자 메뉴: 입고

MenuFrame > managerLogin > menumanager > storebtn(J버튼)

구현 기술: ActionListener / ArrayList/ DefaultTableModel / JTable/ScrollPane / ScrollBar / Timer / ManagerDAO

text~로 입력받은 책 이름, 작가이름, 가격, 재고를 PaperDTO에 설정한다.

mdao는 PaperDTO로, pinsert 메서드를 통해 pdto에 담긴 상품 정보를 데이터베이스에 저장한다.

ManagerDAO의 pinsert 메서드를 실행하여 데이터베이스에 pdto 데이터를 저장한다.

이후 테이블에 시퀀스를 통해 생성 된 주문코드를 받아오고 주문코드를 포함한 데이터를 테이블에 추가한다.

이후, TextField를 초기화하고 UI업데이트 후 완료 메시지를 표시한다.

```
if(e.getSource() == storebtn) {  
    String pname = txtpname.getText();  
    String pauthor = txtpauthor.getText();  
    int pprice = Integer.parseInt(textpprice.getText());  
    int pquantity = Integer.parseInt(String.valueOf(txtpquantity2.getValue()));  
    PaperDTO pdto = new PaperDTO();  
    pdto.setPname(pname);  
    pdto.setPauthor(pauthor);  
    pdto.setPprice(pprice);  
    pdto.setPquantity(pquantity);  
    mdao.pinsert(pdto);  
    getPorderedlist().fireTableDataChanged();  
  
    String pcode = mdao.pcodereturn(pdto);  
    getPorderedlist().addRow(new Object[] {pcode, pname, pauthor, pprice, pquantity});  
  
    txtpname.setText("");  
    txtpauthor.setText("");  
    textpprice.setText("");  
    txtpquantity2.setValue(0);  
  
    contentPane.revalidate();  
    contentPane.repaint();  
  
    JLabel footerp = new JLabel("입고 완료");  
    southP.add(footerp);  
    Timer timer = new Timer(3000, sec3 -> {  
        contentPane.remove(footerp);  
        contentPane.revalidate();  
        contentPane.repaint();  
    });  
    timer.setRepeats(false);  
    timer.start();  
}
```

코드 증빙 및 해설 - 관리자 메뉴: 입고

```
public void pinsert(PaperDTO pdto) {
    if(cdao.conn()) {
        try {
            String sql = "INSERT INTO paperbook VALUES ('p' || paper_seq.NEXTVAL, ?,?,?,?,?,?)";
            PreparedStatement psmt = cdao.conn.prepareStatement(sql);
            psmt.setString(1, pdto.getPname());
            psmt.setString(2, pdto.getPauthor());
            psmt.setInt(3, pdto.getPprice());
            psmt.setInt(4, pdto.getPquantity());
            int resultInt = psmt.executeUpdate();
            if(resultInt > 0) {
                cdao.conn.commit();
            }else {
                cdao.conn.rollback();
            }
        } catch(SQLException e) {
        }
    }
}
```

시퀀스를 통해 도서 코드를 생성, 이후 책이름, 작가, 가격, 재고가 설정된다.

코드 증빙 및 해설 - 관리자 메뉴: 수정

MenuFrame > managerLogin > menumanager > modbtn(J버튼)

구현 기술: ActionListener / ArrayList/ DefaultTableModel / JTable/ScrollPane / ScrollBar / Timer / ManagerDAO

```
if(e.getSource() == modbtn) {
    int selrow = pordered.getSelectedRow();
    if (selrow != -1) {
        String pcode = String.valueOf(getPorderedlist().getValueAt(selrow, 0));
        String pname = String.valueOf(getPorderedlist().getValueAt(selrow, 1));
        String pauthor = String.valueOf(getPorderedlist().getValueAt(selrow, 2));
        double pprice = (Double) getPorderedlist().getValueAt(selrow, 3);
        int pquantity = (Integer) getPorderedlist().getValueAt(selrow, 4);

        mmframe = new ManagerModFrame(this, pdao, mdao, pcode, pname, pauthor, pprice, pquantity);
        mmframe.setVisible(true);

    } else {
        System.out.println("수정할 서적을 선택하지 않았습니다.");
    }
}
if(e.getSource() == backbtn) {
    this.dispose();
```

도서를 선택한 뒤 modbtn을 누르면 actionPerformed 메소드가 실행된다.

선택한 도서의 행 데이터를 가져온다. 지금 가져온 데이터는 수정 될 데이터.

이후 수정 할 데이터를 입력할 ManagerModFrame이 생성된다.

코드 증빙 및 해설 - 관리자 메뉴: 수정

ManagerModFrame > modbtn2(J버튼)

구현 기술: ActionListener / ArrayList/ DefaultTableModel / JTable/ScrollPane / ScrollBar / Timer / ManagerDAO

이후, 바꿀 TextField에 데이터를 입력한 뒤 '수정하기(modbtn2)' 버튼을 누르면 actionPerformed 메소드가 실행된다.

입력 된 데이터를 가져온 뒤 PaperDTO에 해당 데이터들을 설정, 이후 ManagerDAO의 pmod를 실행하여 데이터베이스를 업데이트한다.

이후, 테이블의 선택한 행을 mpframe.getPorderedlist()... 코드들을 통해 업데이트한다.

마지막으로 테이블을 갱신하고 현재 프레임을 종료, 도서 관리 프레임은 다시 표시하고 해당 메서드가 끝이 난다.

```
if(e.getSource() == modbtn2) {  
    String modname = name.getText();  
    String modauthor = author.getText();  
    double modprice = Integer.parseInt(price.getText());  
    int modquantity = Integer.parseInt(quantity.getText());  
    PaperDTO pdto = new PaperDTO();  
    pdto.setPcode(pcode);  
    pdto.setPname(modname);  
    pdto.setPauthor(modauthor);  
    pdto.setPprice((int)modprice);  
    pdto.setPquantity(modquantity);  
  
    mdao.pmod(pdto);  
  
    int selectedRow = mpframe.pordered.getSelectedRow();  
    mpframe.getPorderedlist().setValueAt(modname, selectedRow, 1);  
    mpframe.getPorderedlist().setValueAt(modauthor, selectedRow, 2);  
    mpframe.getPorderedlist().setValueAt(modprice, selectedRow, 3);  
    mpframe.getPorderedlist().setValueAt(modquantity, selectedRow, 4);  
    mpframe.getPorderedlist().fireTableDataChanged();  
    mpframe.pordered.repaint();  
    this.dispose();  
    mpframe.setVisible(true);
```

코드 증빙 및 해설 - 관리자 메뉴: 삭제

MenuFrame > managerLogin > menumanager > delbtn(J버튼)

구현 기술: ActionListener / ArrayList/ DefaultTableModel / JTable/ScrollPane / ScrollBar / Timer / ManagerDAO

도서를 선택한 뒤 delbtn을 누르면 actionPerformed 메소드가 실행된다.

선택한 도서의 코드를 가져온다.

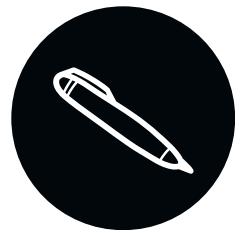
먼저, 데이터베이스에서 해당 코드를 가진 튜플을 삭제, 이후 테이블에서도 삭제한다.

이후 삭제 완료 메시지를 하단에 3초 동안 띄운 뒤 해당 메서드는 끝이 난다.

```
if(e.getSource() == delbtn) {  
    int selrow = pordered.getSelectedRow();  
    String delcode = String.valueOf(getPorderedlist().getValueAt(selrow, 0));  
    mdao.pdel(delcode);  
    getPorderedlist().removeRow(selrow);  
    contentPane.revalidate();  
    contentPane.repaint();  
  
    JLabel footerp = new JLabel("삭제 완료");  
    southP.add(footerp);  
    Timer timer = new Timer(3000, sec3 -> {  
        contentPane.remove(footerp);  
        contentPane.revalidate();  
        contentPane.repaint();  
    });  
    timer.setRepeats(false);  
    timer.start();  
}
```

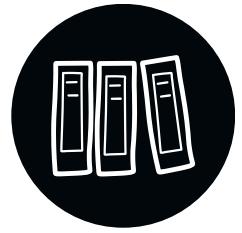


후기



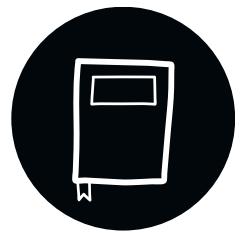
GUI구성

- 프레임을 나누고 UI를 구성하는 것에 큰 어려움을 느꼈다. 이러한 문제에 익숙해지기 위해 많은 시간을 쏟아냈고 완성해나가면서 끈기있게 결과물을 만들어나갔다.



DB Connection에 대한 고민

- 커넥션 코드를 계속 사용해야하는 것에 큰 번거로움을 느낌
- ConnectionDAO를 싱글톤 패턴으로 만들어서 코드를 간소화 시켜보았다.



처음으로 혼자 완성해본 프로젝트

- 남의 도움 없이 혼자 원하는 기능에 대해 생각하고 이를 시간을 들여 구현하는 것에 큰 성취감을 느꼈다.



감사합니다!

