

보험 계약 프로그램 프로젝트

박정환, 이호준

2025-04-03

목차

1 개요

- . 프로젝트 소개와 목적
- . 개발 동기
- . 개발 환경

2 시스템 설계

- . UI (User Interface) 설계
- . 기능 정의서
- . 전체 시스템 구조
- . 데이터베이스 ERD
- . UML

3 실행 화면

- . 관리자 로그인
- . 신규계약 등록 전체 과정
- . 기존 고객 검색
- . 신규 고객 등록
- . 보험상품 조회
- . 보험상품 결정, 보험료 산출
- . 계약내용 확인
- . 시연 영상

4 주요 기능

- . java GUI 기능
- . java 라이브러리 기능
- . 데이터베이스 기능

5 맺음말

- . 결론 및 향후 계획
- . 프로젝트 진행 경과도
- . 프로젝트 참여자

① 프로젝트 개요

프로젝트 소개와 목적

- . 보험 계약을 등록, 고객정보와 보험 상품을 조회하는
일원화된 프로그램 개발
- . 기존 보험계약 관리 프로그램의 경우, 여러 화면을 거쳐야 비로소 신규계약 등록을 완료할 수 있었음. 본 프로젝트는 이러한 과정을 줄여 한 화면에서 처리할 수 있는 프로그램을 목표로 제작.
- . 현업의 경험을 바탕으로 실제 사용자인 관리자의 편의성 제고를 목표로 프로그램 개발.

① 프로젝트 개요

개발 동기

- . 보험사의 시각에서 구성된 프로그램이 아닌, 보험 상품을 직접 다루는 관리자의 편의성에 초점을 맞춘 새로운 프로그램의 필요성.
- . 특히 피보험자의 월 보험료를 신규계약 등록 이후에야 인쇄물로 확인할 수 있는 체계에 큰 불편 발생. 이를 계약 등록 전 미리 확인하며 고객과 프로그램 사용자의 편의에 따른 더욱 효율적인 보험상품 설계 가능.

① 프로젝트 개요

개발 환경

- java JDK: 1.8
- java IDE: eclipse 4.8.0
(사용 라이브러리: ojdbc6, Lombok v1.18.36)
- DB: Oracle 11.2.0.1.0
- OS : Windows 10 pro 64비트 (10.0, 빌드 19045)



ORACLE®



② 시스템 설계

UI (User Interface) 설계



메인 화면

- 관리자 로그인
- 관리자 등록



보험상품 목록 화면

- 기존 고객, 상품 조회
- 신규 고객, 계약 등록
- 기존 계약 삭제
- 상품명, 주계약/특약 목록 조회
- 예상 월 보험료 계산

② 시스템 설계

기능 정의서

구분	분류	기능	설명
사용자 기능	보험 상품 관리	신규 계약 등록	신규 보험계약, 신규 고객(피보험자)의 정보를 동시에 등록.
		기존 계약 조회	기존 고객의 주민등록번호로 검색 가능. 고객 정보, 계약 정보 출력.
		기존 계약 삭제	데이터베이스에 저장된 기존 계약을 삭제.
		상품 조회	상품 검색 기능, 해당 상품에서 제공하는 주계약과 특약 조회 기능
		질병이력 조회	등록된 질병이력을 검색.
		월 보험료 계산	주계약과 특약을 추가하면 그 즉시 예상 월 보험료를 나타냄. 성별, 연령, 질병이력 등에 따라 변동 가능.
		가입금액 제한	상품과 주계약에 따라 최소-최대 가입금액 내에서만 등록 가능.
		가입기간 제한	가입기간은 1~30년만 가능.
	관리자 정보 관리	로그인/로그아웃	등록된 관리자의 계정과 비밀번호 확인 후 로그인. 종료 시 로그아웃 기능 제공.
		신규 관리자 등록	신규 관리자의 계정, 암호, 이름을 등록.
		중복 ID 방지 기능	신규 관리자의 계정과 기존 계정의 중복 등록 방지.
		ID, 비밀번호 글자 수 제한	ID와 비밀번호는 4~20자만 가능.
개발자 기능	GUI	배경화면	로그인 화면에 이미지 삽입
		한 화면에 복수의 기능 사용	한 화면 위에서 조회, 등록, 삭제 기능을 모두 사용할 수 있다.
		체크박스	사용자가 선택할 항목을 가시적으로 선택, 실행 가능
		주계약은 등록 제한	주계약은 1개만 등록 가능
		예상 보험료 초기값	피보험자 정보가 없는 상태에서는 40세 남성을 기준으로 보험료 계산
		등록/삭제 제한	등록은 신규계약만, 삭제는 기존계약만 가능.
	데이터베이스	데이터베이스 연동	java에서 입력한 내용을 데이터베이스에 입력, 검색, 삭제 기능.

② 시스템 설계

전체 시스템 구조

```

@Override
public void actionPerformed(ActionEvent e) {
    if(e.getSource() == btn1) {
        btn1();
    } else if(e.getSource() == btn2) {
        if(jt2.getSelectedRow() == -1) {
            JOptionPane.showMessageDialog(null, "먼저 주선등록번호를 선택해주세요.");
        } else {
            btn2();
        }
    } else if(e.getSource() == btn31) {
        btn31();
    } else if(e.getSource() == btn32) {
        btn32();
    } else if(e.getSource() == btn5) {
        btn5();
    } else if(e.getSource() == btn7) {
        if(jt7.getSelectedRow() == -1) {
            return;
        } else {
            int[] sr = jt7.getSelectedRows();
            for(int i : sr) {
                dtm7.removeRow(jt7.getSelectedRow());
            }
        }
    } else if(e.getSource() == btn70) {
        if(jt70.getSelectedRow() == -1) {
            return;
        } else {
            int[] sc = jt70.getSelectedRows();
            for(int i : sc) {
                dtm70.removeRow(jt70.getSelectedRow());
                calculatefee();
            }
        }
    } else if(e.getSource() == btn90) {
        calculatefee();
    }
}
    
```

```

SQL Plus
4 440424-2
   전라북도 무안군 고죽
5 550515-1
   강원도 춘천시 호반로
6 660606-2
   제주도 서귀포시 신록

M_ID      M_REGNUM      M_NAME
-----
MLADDR
7 770707-1
   충청북도 괴산군 괴산
8 880808-2
   경기도 수원시 장안
9 990430-1
   부산광역시 수영구 풍

9 개의 행이 선택되었습니다.

SQL> insert into Contract values (con_seq.NEXTVAL,2,'B00001',10000,30,'E00101','F101010','F104010',null,null,null,null,null,null);
    
```

GUI (Graphic User Interface)

- 사용자에게 필요한 정보와 기능을 화면에 표시
- 사용자가 입력한 명령을 java 코드로 넘김



java

- 프로그램의 기능을 GUI에 표시
- GUI에서 입력받은 명령을 DB 넘김
- DB에서 받은 데이터를 GUI로 넘김

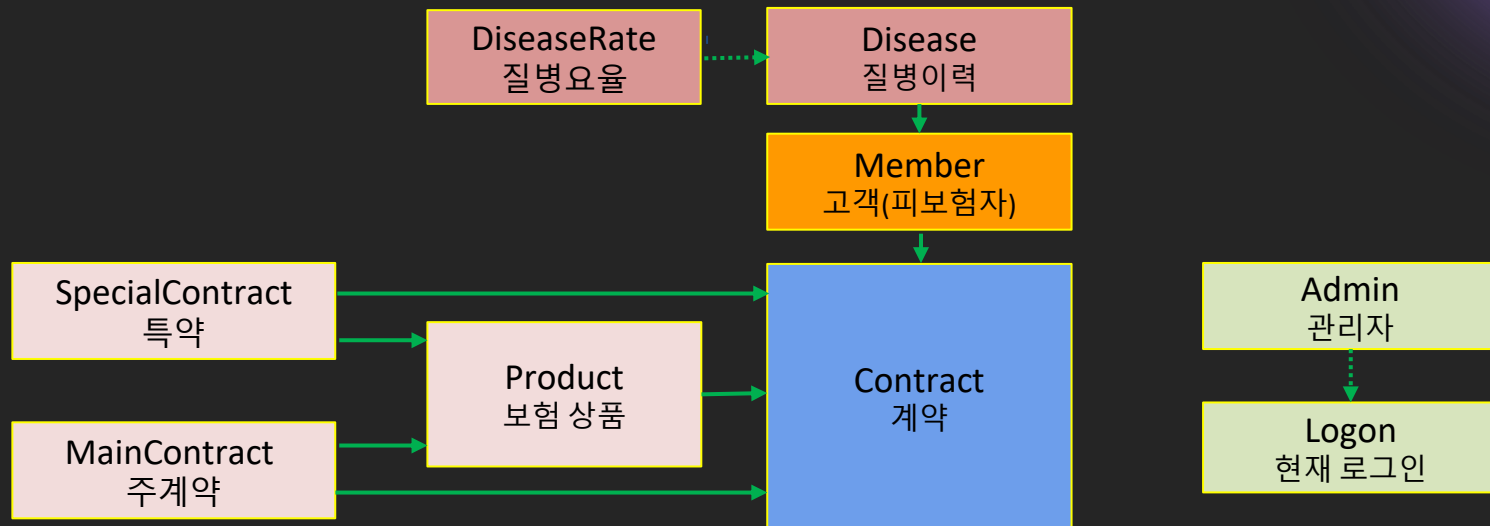


데이터베이스 (DB)

- java에서 넘겨받은 명령을 DB에 입력
- 새로운 데이터를 DB에 저장
- 호출받은 데이터를 java로 넘김

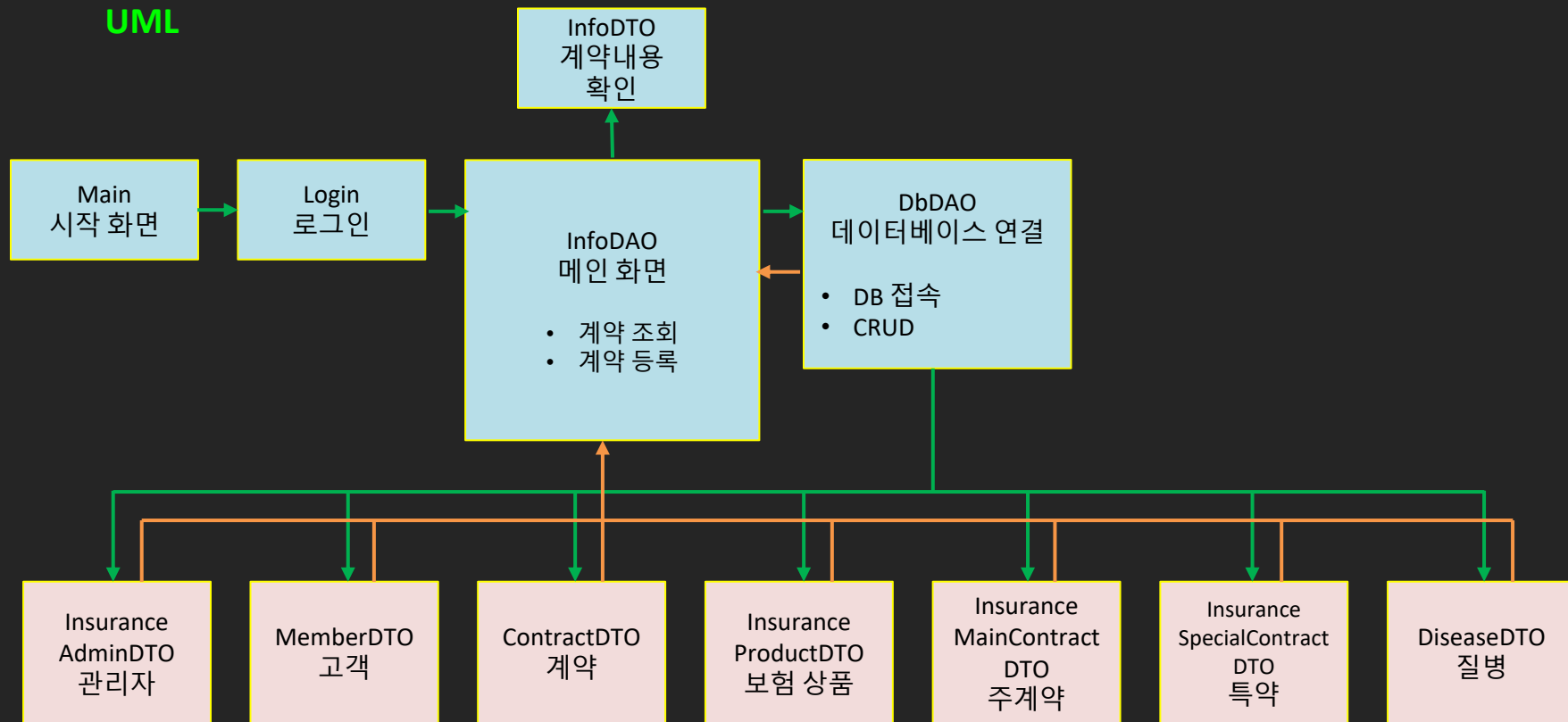
② 시스템 설계

ERD



② 시스템 설계

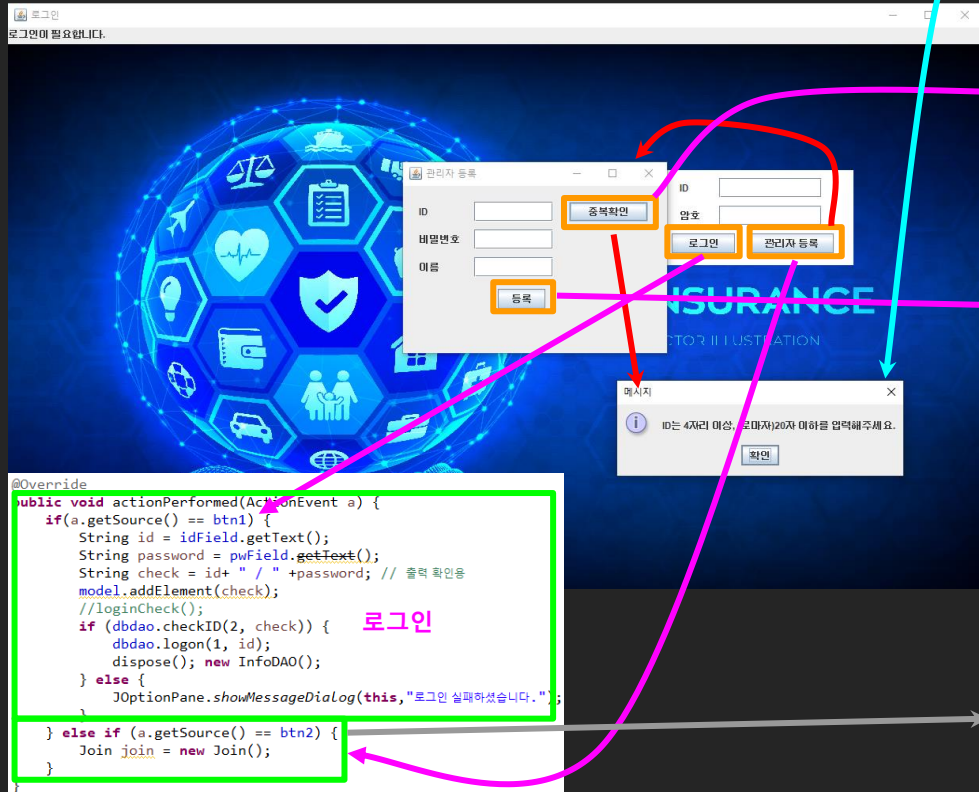
UML



③ 실행 화면

관리자 로그인

관리자 등록



```
@Override
public void actionPerformed(ActionEvent e) {
    String newID = jtf21.getText();
    if (newID.length() < 4) {
        JOptionPane.showMessageDialog(
            null, "ID는 4자리 이상, (로마자)20자 이하를 입력해주세요.");
        jtf21.setText(""); return;
    }

    if (e.getSource() == btn21) {
        if (newID.equals("")) {
            JOptionPane.showMessageDialog(null, "사용하실 ID를 입력하세요.");
            return;
        }
        if (dbdao.checkID(1, newID)) {
            JOptionPane.showMessageDialog(this, "ID가 중복됩니다. 다른 ID를 사용해주세요.");
            return;
        } else {
            JOptionPane.showMessageDialog(this, "사용가능한 ID입니다.");
        }
    } else if (e.getSource() == btn24) {
        if (newID == null) {
            JOptionPane.showInputDialog("사용하실 ID를 입력하세요.");
            return;
        }
        if (dbdao.checkID(1, newID)) {
            JOptionPane.showMessageDialog(this, "ID가 중복됩니다. 다른 ID를 사용해주세요.");
            return;
        } else {
            // System.out.println("ID 사용 가능");
            String newPW = jpf22.getText();
            String newAdminName = jtf23.getText();
            if (newPW.length() < 4) {
                JOptionPane.showMessageDialog(null, "비밀번호는 4자리 이상 입력해주세요.");
                return;
            } else if (newAdminName.length() < 1) {
                JOptionPane.showMessageDialog(null, "이름을 입력해주세요.");
                return;
            } else {
                InsuranceAdminDTO admdto = new InsuranceAdminDTO();
                admdto.setAdmin_id(newID);
                admdto.setAdmin_password(newPW);
                admdto.setAdmin_name(newAdminName);
                dbdao.insertAdmin(admdto);
                idField.setText(newID);
                JOptionPane.showMessageDialog(null, "관리자 등록에 성공했습니다.");
            }
        }
    }
}
```

④ 글자 수 제한

③ 중복 확인

② ID 중복확인,
비밀번호 4자리 이상
이름 입력 필수

① 관리자 등록

③ 실행 화면

신규계약 등록 전체 과정

계약 등록/조회

로그아웃

님 환영합니다.

주면등록번호 조회

① 기존고객 조회/
신규고객 정보 입력

주민등록번호 피보험자 생년월일 주소

880808-2 아아아 D00505 경기도 수원시 장안구 장안동

등록

② 고객정보 등록

피보험자 연령 성별 생년월일

아아아 36 F D00505

계약코드 상품코드 상품명 가입금액 가입기간 신규

1 B00008 암질환보험 5000만원 15년 X

신규 B00001 기본종신보험 10000만원 30년 O

선택 취소

③ 보험상품 조회

상품코드 상품명

B00001 기본종신보험

B00002 특수종신보험

④ 보험상품 등록 +
주계약/특약 검색

상품명 조회

가입금액 만원

가입기간 년 등록

⑤ 주계약/특약
등록

계약코드 계약명 가입금액 보험료 신규

E00101 기본종신주계약 10000만원 200000 O

F104010 수술특약 4000만원 2500 O

F107010 통원치료특약 1000만원 1200 O

⑥ 예상 월 보험료
계산

계약코드 구분 계약코드 계약명 가입금액 보험료

✓ 주계약 E00101 기본종신주계약 10000만원 200000

✓ 특약 F101010 입원치료특약 2000만원 1500

✓ 특약 F104010 수술특약 4000만원 2500

✓ 특약 F107010 통원치료특약 1000만원 1200

추가

예상 월 보험료 201,663 원

예상 등록 삭제

신규계약 등록 기존계약 삭제(해지)

③ 실행 화면

기존 고객(피보험자) 검색

주민등록번호 조회

주민등록번호	피보험자	병력	주소
880808-2	아아아	D00505	경기도 수원시 장안구 장안동

등록

피보험자	연령	성별	병력
아아아	36	F	D00505

계약코드	상품코드	상품명	가입금액	가입기간	선유
1	B00008	암질환보험	5000 만원	15 년	X
신규	B00001	기본중신보험	10000 만원	30 년	O

선택 취소

```
// 기존 고객이 가지고 있는 보형계약 정보 불러오기
private void btn21(String memberCode) {
    for(int i=0; i<dtm7.getRowCount(); i++){
        dtm7.removeRow(i); i=-1;
    }
    for(int i=0; i<dtm70.getRowCount(); i++){
        dtm70.removeRow(i); i=-1;
    }
    ArrayList<ContractDTO> cdto = dbdao.selectContract(1, memberCode);
    beCode = "X"; int cdtoCount = 0; int j = 0;
    for(ContractDTO cdtoFor : cdto) {
        String prodCode = cdtoFor.getP_ID(); // 상품 코드
        String prodName = dbdao.selectProduct(prodCode); // 상품명 검색
        Vector<Object> v2 = new Vector<Object>(); // 상품 등록 파일(jp7)
        prodPrice = cdtoFor.getC_amount(); // 가입금액
        v2.add(cdtoFor.getSC_ID1()); v2.add(cdtoFor.getSC_ID2()); v2.add(prodName);
        v2.add(cdtoFor.getSC_ID3()); v2.add(cdtoFor.getSC_ID4()); v2.add(cdtoFor.getSC_ID5());
        v2.add(cdtoFor.getSC_ID6()); v2.add(cdtoFor.getSC_ID7()); v2.add(cdtoFor.getSC_ID8()); v2.add(cdtoFor.getSC_ID9());
        dtm7.addRow(v2);
        msCont[0] = cdtoFor.getMC_ID1(); msCont[1] = cdtoFor.getSC_ID1();
        msCont[2] = cdtoFor.getSC_ID2(); msCont[3] = cdtoFor.getSC_ID3();
        msCont[4] = cdtoFor.getSC_ID4(); msCont[5] = cdtoFor.getSC_ID5();
        msCont[6] = cdtoFor.getSC_ID6(); msCont[7] = cdtoFor.getSC_ID7();
        msCont[8] = cdtoFor.getSC_ID8(); msCont[9] = cdtoFor.getSC_ID9();
    }
}
```

③ 기존 계약 또한
동시에 출력

```
private void btn1() {
    String searchNumber = jtf1.getText();
    for(int i=0; i<dtm2.getRowCount(); i++){
        dtm2.removeRow(i); i=-1;
    }
    if(searchNumber.equals("")) {
        JOptionPane.showMessageDialog(null, "주민등록번호를 입력해주세요.");
    } else if (searchNumber.length() < 7 || searchNumber.charAt(7) < 49 || searchNumber.charAt(7) > 52) {
        JOptionPane.showMessageDialog(null, "주민등록번호 입력 오류입니다.");
        return;
    } else {
        ArrayList<MemberDTO> memlist = dbdao.memberselectAll(searchNumber);
        Vector<Object> v = new Vector<Object>();
        if (memlist.isEmpty()) {
            JP20 jp20 = new JP20(searchNumber);
        } else {
            for(MemberDTO memFor : memlist) {
                v.add(memFor.getM_registrationnumber());
                v.add(memFor.getM_name());
                v.add(memFor.getM_diseasehistory());
                v.add(memFor.getM_addr());
                dtm2.addRow(v);
                //memFor.prt();
            }
        }
        jtf1.setText("");
    }
}
```

① 주민번호로 조회하면
해당 고객 명단이 표시

```
private void btn2() {
    for(int i=0; i<dtm6.getRowCount(); i++){
        dtm6.removeRow(i); i=-1;
    }
    String code = (String)jt2.getValueAt(jt2.getSelectedRow(), 0);
    ArrayList<MemberDTO> memlist2 = dbdao.memberselectAll2(code);
    Vector<Object> v = new Vector<Object>();
    if (memlist2.isEmpty()) {
        calculateAge();
        if (memberAge == -1) {
            JOptionPane.showMessageDialog(null, "주민등록번호 입력 오류입니다.");
            return;
        } else {
            v.add(dtm2.getValueAt(0,1)); v.add(memberAge);
            v.add(memberSex); v.add(dtm2.getValueAt(0,2));
            dtm6.addRow(v);
        }
    } else {
        for(MemberDTO memFor2 : memlist2) {
            calculateAge();
            v.add(memFor2.getM_name()); v.add(memberAge); v.add(memberSex);
            //v.add(memFor2.getM_age());
            //v.add(memFor2.getM_gendertype());
            if(memFor2.getM_diseasehistory() != null) {
                v.add(memFor2.getM_diseasehistory());
            } else {
                v.add("없음");
            }
            dtm6.addRow(v);
            //memFor2.prt2();
        }
    }
    memberAge=0; memberSex=null;
}
```

② 고객 명단을 등록하면
등록된 고객 정보 출력

③ 실행 화면

신규 고객(피보험자) 등록

계약 등록/조회

■■■■ 님 환영합니다.

주면등록번호

주면등록번호 피보험자 병력 주소

피보험자 연령 성별 병력

계약코드 상품코드 상품명 가입금액 가입기간 신규

계약코드 계약명 가입금액 보험료 신규

피보험자 등록

주면등록번호 871014-■■■■

피보험자 홍길동

병력 D00000

주소 서울특별시 송파구 석촌로 36

질병코드	질병명
<input checked="" type="checkbox"/> D00000	없음
<input type="checkbox"/> D00101	당뇨
<input type="checkbox"/> D00202	위암
<input type="checkbox"/> D00301	폐암
<input type="checkbox"/> D00401	심근경색
<input type="checkbox"/> D00402	간경화
<input type="checkbox"/> D00501	화상
<input type="checkbox"/> D00502	루게릭병
<input type="checkbox"/> D00601	장폐(상체)
<input type="checkbox"/> D00602	장폐(상체)
<input type="checkbox"/> D00701	중이염
<input type="checkbox"/> D00702	녹내장

```
@Override
public void actionPerformed(ActionEvent e) {
    if(e.getSource() == btn21) {
        for(int i=0; i<dtm2.getRowCount(); i++){
            dtm2.removeRow(i); i=-1;
        }
        String[] memberInfo = new String[4];
        memberInfo[0] = jtf21.getText(); memberInfo[1] = jtf22.getText();
        memberInfo[2] = jtf23.getText(); memberInfo[3] = jtf24.getText();
        dtm2.addRow(memberInfo); memberInfo = null;
        dispose(); return;
    } else if(e.getSource() == btn22) {
        dispose();
    } else if(e.getSource() == btn23) {
        for(int i=0; i<dtm23.getRowCount(); i++){
            dtm23.removeRow(i); i=-1;
        }
        String searchD = jtf23.getText();
        ArrayList<DiseaseDTO> dlist
            = dbdao.selectDiseaseDTO(searchD);
        for(DiseaseDTO dFor : dlist) {
            Vector<Object> v = new Vector<Object>();
            v.add(false); v.add(dFor.getD_id()); v.add(dFor.getD_name());
            dtm23.addRow(v);
        }
        return;
    } else if(e.getSource() == btn24) {
        Boolean checked = Boolean.valueOf
            (jtf23.getValueAt(jtf23.getSelectedRow(),0).toString());
        if(checked)
            jtf23.setText(jtf23.getValueAt(jtf23.getSelectedRow(),1).toString());
        // 복주의 것을 선택할 때
        /*for(int i=0; i<jtf23.getRowCount(); i++){
            dtm23.removeRow(i); i=-1;
        }
        return;
    }
}
```

③ 등록/취소

① 병력(病歷) 검색

② 검색한 병력 선택

③ 실행 화면

보험상품 조회

상품코드	상품명
B00001	기본종신보험
B00002	특수종신보험

상품명	<input type="text" value="종신"/>	<input type="button" value="조회"/>
가입금액	<input type="text"/>	만원
가입기간	<input type="text"/>	년 <input type="button" value="등록"/>

```
private void btn31() {  
    for(int i=0; i<dtm3.getRowCount(); i++){  
        dtm3.removeRow(i);  
        i=-1;    // 기존에 잔존하던 테이블 값을 완전히 지우기 위해 일부러 i값을 0으로  
    }  
    String searchProd = jtf31.getText();  
    if(searchProd.equals("")) {  
        JOptionPane.showMessageDialog(null, "상품을 입력해주세요.");  
    } else {  
        ArrayList<InsuranceProductDTO> prodlist  
        = dbdao.selectInsuranceProductDTO(searchProd);  
        for(InsuranceProductDTO prodFor : prodlist) {  
            Vector<Object> v = new Vector<Object>();  
            v.add(prodFor.getIP_id());  
            v.add(prodFor.getIP_name());  
            dtm3.addRow(v);  
            // prodFor.prt();  
        }  
    }  
}
```

③ 실행 화면

보험상품 결정, 보험료 산출

계약코드	계약명	가입금액	보험료	신규
E00301	기본단체계약	10000 만원	20000	O
F103030	산입재해특약	2000 만원	500	O
F104030	수술특약	7000 만원	3500	O
F107030	통원치료특약	1000 만원	710	O

← 취소

v	구분	계약코드	계약명	가입금액	보험료
<input checked="" type="checkbox"/>	주계약	E00301	기본단체계약	10000 만원	20000
<input type="checkbox"/>	특약	F101030	입원치료특약	1000 만원	500
<input checked="" type="checkbox"/>	특약	F103030	산입재해특약	2000 만원	500
<input type="checkbox"/>	특약	F104030	수술특약	7000 만원	3500
<input checked="" type="checkbox"/>	특약	F107030	통원치료특약	1000 만원	710

← 추가

예상 월 보험료 24,525 원

← 계산 등록 삭제

```
private void calculatefee() {
    fee = 0; int age; int rateAge = 0; int rateSex = 0; String ill = null;
    for(int i = 0; i<dtm70.getRowCount(); i++) {
        int tempFee = Integer.parseInt(dtm70.getValueAt(i,3).toString());
        fee += tempFee;
    }
    if (dtm6.getRowCount() == 0) {
        JOptionPane.showMessageDialog(
            null, "피보험자 정보가 없으면 40% 납성을 기준으로 월 보험료를 계산합니다.");
        age = 40; rateSex = 25;
    } else {
        age = (int) dtm6.getValueAt(0,1);
        if (dtm6.getValueAt(0,2).equals("M")) rateSex = 25;
        ill = dtm6.getValueAt(0,3).toString(); // 피보험자 병력
    }
    int rateIll = dbdao.selectDisease(ill);
    NumberFormat nf = NumberFormat.getInstance(); // 숫자 3자리마다 콤마 붙이기
    int ageEra = (age/10)*10; // 연령대(20대, 30대, 40대, ...)
    if (ageEra < 30) rateAge = -200;
    else if (ageEra >= 30 && ageEra < 40) rateAge = -100;
    else if (ageEra >= 40 && ageEra < 50) rateAge = 0;
    else if (ageEra >= 50 && ageEra < 60) rateAge = 100;
    else if (ageEra >= 60 && ageEra < 70) rateAge = 200;
    int amountCons = (fee +
        (fee * (rateAge + rateSex + rateIll) / 10000));
    String amountWon = nf.format(amountCons);
    t182.setText(amountWon);
}
```

② 보험료 계산 후 GUI에 표기

```
private void btn5() {
    if (jt5.getSelectedRow() == -1) {
        JOptionPane.showMessageDialog(null, "주계약/특약을 선택하세요.");
        return;
    }
    Vector<Object> selectRow = new Vector<>();
    for(int i=0; i<jt5.getRowCount(); i++) {
        Boolean checked = Boolean.valueOf(jt5.getValueAt(i, 0).toString());
        if(checked) {
            selectRow.add(i);
        }
    }
    for (int j=0; j<dtm70.getRowCount(); j++) {
        String existCode = dtm70.getValueAt(j,0).toString(); // 0번 열이 계약코드
        for (int k=0; k<selectRow.size(); k++) {
            if (existCode.equals
                (jt5.getValueAt((int) selectRow.get(k),2).toString())) {
                JOptionPane.showMessageDialog(null, "이미 추가된 계약입니다.");
                return;
            }
        }
    }
    for(int i=0; i<jt5.getRowCount(); i++) {
        Boolean checked = Boolean.valueOf(jt5.getValueAt(i, 0).toString());
        if(checked) {
            Vector<Object> row = new Vector<Object>();
            row.add(jt5.getValueAt(i, 2).toString());
            row.add(jt5.getValueAt(i, 3).toString());
            row.add(jt7.getValueAt(0, 2).toString());
            row.add(jt5.getValueAt(i, 4).toString());
            dtm70.addRow(row);
        }
    }
    calculatefee();
}
```

① 주계약/특약을 추가하는 동시에 보험료도 계산

③ 실행 화면

계약내용 확인

약관내용 확인

계약코드: 25 / 피보험자: 장길산 / 상품코드: B00001 / 상품명: 기본종신보험

구분	계약코드	계약명	가입금액	가입기간
주계약	E00101	기본종신주계약	12000 만원	30 년
특약	F104010	수술특약	4000 만원	30 년
특약	F107010	통원치료특약	1000 만원	30 년

예상 보험료

나이	금액
27 세	200,441 원
30 세	202,478 원
40 세	204,514 원
50 세	206,551 원

확인

200,441 원

매상 등록 삭제

```
public InfoDTO(String[] dto, DefaultTableModel dtm10, ArrayList<String[]> arr2) {
    this.dtm10 = dtm10; this.arr2 = arr2;
    this.period = Integer.parseInt(dto[4].replaceAll("[^0-9]", "")); // 가입기간
    this.setBounds(200, 200, 720, 450);
    this.setTitle("계약내용 확인");
    JLabel jld0 = new JLabel(" 계약코드: " + dto[0] + " / 피보험자: " + dto[1] +
        " / 상품코드: " + dto[2] + " / 상품명: " + dto[3]);
    jld0.setFont(new Font("맑은 고딕", Font.BOLD, 15));
    this.add("North", jld0);
    jld1(); jld2(); // jld3(); jld4();
    jpD1(); jpD2(); /* jpD3(); jpD4(); */ jpD5();
    jpD10.add(jpD1); jpD10.add(jpD2); jpD10.add(jpD3); jpD10.add(jpD4); jpD10.add(jpD5);
    jpD10.setLayout(null);
    this.add(jpD10);
    this.setVisible(true);
}
```

계약내용 확인
전체 패널

```
int amount = (int) dtm10.getValueAt(0, 0); // 보험료
int age = (int) dtm10.getValueAt(0, 1); // 피보험자 연령
int rateAge = 0; // 피보험자 연령에 따른 조정 요율 *10000
int rateSex = 0; // 피보험자 성별에 따른 조정 요율 *10000
if (dtm10.getValueAt(0, 2).equals("M")) rateSex = 25; // 남성 +0.25%
String ill = (String) dtm10.getValueAt(0, 3); // 피보험자 병력
int rateIll = dbdao.selectDisease(ill);
// 피보험자 병력에 따른 조정 요율 *10000
NumberFormat nf = NumberFormat.getInstance(); // 숫자 3자리마다 콤마 붙이기
int ageEra;
for (int iage: i < 80; i+=10) {
    if (i > period+age) break;
    ageEra = (i/10)*10; // 연령대(20대, 30대, 40대, ...)
    if (ageEra == (age/10)*10) {
        ageEra = age;
    }
    if (ageEra < 30) {
        rateAge = -200; // 20대 이하 -2%
    } else if (ageEra >= 30 && ageEra < 40) {
        rateAge = -100;
    } else if (ageEra >= 40 && ageEra < 50) {
        rateAge = 0;
    } else if (ageEra >= 50 && ageEra < 60) {
        rateAge = 100;
    } else if (ageEra >= 60 && ageEra < 70) {
        rateAge = 200;
    }
}
int amountCons = (amount +
    (amount * (rateAge + rateSex + rateIll) / 10000));
String amountWon = nf.format(amountCons);
Vector<Object> v = new Vector<>();
v.add(ageEra + " 세");
v.add(amountWon + " 원");
dtmD1.addRow(v);
}
```

예상 보험료 출력.
나이+가입기간까지만
출력한다.

③ 실행 화면

시연 영상

https://youtu.be/9S_TslrFXbk

④ 주요 기능

java GUI 기능

사용자가 화면을 통해 정보를 인지하고,
명령을 내리는 체계를 구현
(JFrame 상속, ActionListener 인터페이스 사용)

The screenshot shows a Java GUI application titled "계약 등록/조회" (Contract Registration/Search). The interface is composed of several panels labeled jp1 through jp10. jp1 is a login panel with fields for ID and password. jp2 is a member registration panel with fields for name, ID, password, and email. jp3 is a contract registration panel with fields for contract ID, name, amount, and term. jp5 is a contract search panel with a table of contracts. jp6 is a member search panel with a table of members. jp7 is a contract details panel with a table of contract details. jp8 is a contract list panel with a table of contracts. jp9 is a contract details panel with a table of contract details. jp10 is the main frame, which contains a table of contracts and a sidebar with buttons for "계상" (Account), "등록" (Register), and "삭제" (Delete). The main frame also has a "전체 패널" (Main Panel) label and a "전체 패널 위에 작은 패널들 올리기" (Move small panels to the top of the main panel) label.

```
public InfoDAO() {
    this.setBounds(0,0,1250,850);
    this.setTitle("계약 등록/조회");
```

```
adminName = dbdao.getAdminName();
JLabel title = new JLabel(" "+adminName+"님 환영합니다.");
title.setFont(new Font("맑은 고딕", Font.BOLD, 15));
JButton logout = new JButton("로그아웃");
JPanel jp0 = new JPanel();
jp0.add(title); jp0.add(logout);
this.add("North", jp0);
logout.addActionListener(this);
```

관리자 이름 표기,
로그아웃 버튼

```
jp1(); jp2(); jp3(); jp5(); jp6(); jp7(); jp70(); jp8(); jp9();
jp10.setLayout(null);
jp10.add(jp1); jp10.add(jp2); jp10.add(jp3); jp10.add(jp5);
jp10.add(jp6); jp10.add(jp7); jp10.add(jp70); jp10.add(jp8); jp10.add(jp9);
this.add(jp10);
this.setVisible(true);
this.setDefaultCloseOperation(this.EXIT_ON_CLOSE);
```

전체 패널 위에
작은 패널들 올리기

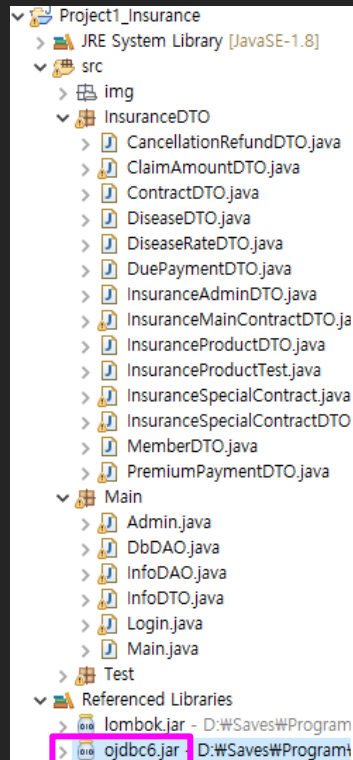
```
dtr = new DefaultTableCellRenderer() {
    public Component getTableCellRendererComponent
        (JTable table, Object value, boolean isSelected,
        boolean hasFocus, int row, int column) {
        JCheckBox box= new JCheckBox();
        box.setSelected(((Boolean)value).booleanValue());
        box.setHorizontalAlignment(JLabel.CENTER);
        return box;
    }
};
```

표에 체크박스 넣기

④ 주요 기능

java 라이브러리 기능 - ojdbc6

ojdbc6 를 이용하여
java에서 데이터베이스에 접근



```
public void insertContract(ContractDTO cdto) {
    PreparedStatement pstmt = null;
    try {
        if(getConnection() != null) {
            String sql = "insert into Contract values "
                + "(con_seq.NEXTVAL,?,?,?,?,?,?,?,?,?,?,?,?,?)";
            pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, cdto.getM_ID());
            pstmt.setString(2, cdto.getP_ID());
            pstmt.setInt(3, cdto.getC_amount());
            pstmt.setInt(4, cdto.getC_period());
            pstmt.setString(5, cdto.getMC_ID());
            pstmt.setString(6, cdto.getSC_ID1());
            pstmt.setString(7, cdto.getSC_ID2());
            pstmt.setString(8, cdto.getSC_ID3());
            pstmt.setString(9, cdto.getSC_ID4());
            pstmt.setString(10, cdto.getSC_ID5());
            pstmt.setString(11, cdto.getSC_ID6());
            pstmt.setString(12, cdto.getSC_ID7());
            pstmt.setString(13, cdto.getSC_ID8());
            pstmt.setString(14, cdto.getSC_ID9());
            pstmt.executeUpdate();
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, "계약 등록에 실패했습니다. 다시 시도해주세요.");
    } finally {
        try {
            pstmt.close();
            conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

SQL Plus

M_ID	M_REGNUM	M_NAME	D_ID
7	770707-1		
8	880808-2		
9	990430-1		

SQL> select * from contract;

C_ID	M_ID	P_ID	C_AMOUNT	C_PERIOD	MC_ID	SC_ID1
SC_ID2	SC_ID3	SC_ID4	SC_ID5	SC_ID6	SC_ID7	SC_ID8
SC_ID9						
1	8	B00008	5000	15	E00801	F101080
F104081	F104082					

8번 고객의 계약이 신규등록됨

④ 주요 기능

데이터베이스 기능

SQL Plus

SQL> desc Product;

이름	널?	유형
P_ID	NOT NULL	VARCHAR2(10)
P_NAME		VARCHAR2(30)

SQL> desc MainContract;

이름	널?	유형
MC_ID	NOT NULL	VARCHAR2(10)
P_ID		VARCHAR2(10)
MC_NAME		VARCHAR2(30)
MC_MIN		NUMBER(38)
MC_MAX		NUMBER(38)
MC_FEE		NUMBER(38)

SQL>

보험상품 데이터를 저장하는 테이블과,
해당 상품에 부가되는 주계약 목록을 저장하는
데이터베이스 테이블.

[보험상품]
보험 상품ID -- PK (B)
보험 상품명 ----- 상품명 이름 (VARCHAR2)

```
CREATE TABLE Product(
P_ID varchar2(10) primary key,
P_name varchar2(30)
);
```

```
insert into Product values ('B00001','기본종신보험');
insert into Product values ('B00002','특수종신보험');
insert into Product values ('B00003','기본단채보험');
insert into Product values ('B00004','특수단채보험');
insert into Product values ('B00007','입원실비보험');
insert into Product values ('B00008','암질환보험');
```

[주계약]

주계약 코드 PK [E]
상품 코드 FK (B)
주계약 이름
주계약 최소 가입금액 (단위: 만원)
주계약 최대 가입금액
주계약 기준 보험료 (40세 기준)

```
CREATE TABLE MainContract(
MC_ID varchar2(10) primary key,
P_ID varchar2(10),
MC_name varchar2(30),
MC_min int,
MC_max int,
MC_fee int,
foreign key(P_ID) references Product(P_ID)
);
```

```
insert into MainContract values ('E00101','B00001','기본종신주계약',2000,200000,200000);
insert into MainContract values ('E00201','B00002','특수종신주계약',10000,500000,250000);
insert into MainContract values ('E00301','B00003','기본단채주계약',1000,40000,20000);
insert into MainContract values ('E00401','B00004','특수단채주계약',2000,70000,24000);
insert into MainContract values ('E00701','B00007','입원치료주계약',500,10000,50000);
insert into MainContract values ('E00702','B00007','통원치료주계약',300,7000,30000);
insert into MainContract values ('E00801','B00008','암질환주계약1형',1000,12000,120000);
insert into MainContract values ('E00802','B00008','암질환주계약2형',1000,8000,80000);
```

⑤ 맺음말

결론 및 향후 계획

- . 실무에 바로 투입 가능한 프로그램 완성
- . 실무자의 편의성 제고라는 목표 달성
- . 부수적으로 관리자 등록, 중복확인, 로그인과 로그아웃 등 기본적인 프로그램 보안 기능도 함께 구현
- . 보험 상품 자체를 등록, 관리하는 기능 추가 가능
(이미 보험 상품에 관한 데이터베이스와 연동되어 있음)

⑤ 맷음말

프로젝트 진행 경과도

날짜 (2025년 3월, 4월)	난이도	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2
1주차: 전체 계획 입안, GUI 틀 구축,																								
기안, 목표 설정	★																							
프로젝트에 필요한 자료 조사	★																							
이호준 활용	★																							
로그인 화면에 이미지 삽입	★★																							
java 클래스 구분, UML 작성	★★																							
계약 등록/조회 패널 구현	★★																							
표 안에 체크박스 구현	★★★																							
java GUI 구성 설계도 작성/변경	★																							
DB 테이블 설정	★★★★																							
UML 작성	★																							
복수의 패널 구현	★★★★																							
MouseListener 인터페이스 적용 (개발과정에만 사용)	★★																							
예상 보험료, 해지환급금 등의 GUI 설계도	★★																							
DB 테이블과 java 데이터 연동	★★																							
2주차: DB와 java 연계, 자료 입력																								
팝업창 구현	★																							
로그인, 주민등록번호 입력	★★																							
중복ID 방지 기능	★★																							
등록된 계정만 로그인 가능	★★																							
보험상품 검색	★																							
피보험자 검색 기능	★★																							
한 패널 위에 여러 기능을 모두 사용	★★★★																							
등록한 상품 정보 다른 표에 표시	★★																							
조회 시 표에 남아있던 이전 검색결과 삭제	★★																							
상품 등록 시 주계약/특약 목록 표시	★★																							
주계약/특약 추가 시 다른 표에 표시	★★																							
월 예상 보험료 계산	★																							
개발 작업 병합, 재작업 시 오류 해결																								
나이 계산 기능	★																							
신규 피보험자 등록 기능 (java 중첩클래스 사용)	★★																							

1 강이영 氏 도움. 개발과정에만 사용.
2 서호진 氏 도움

날짜 (2025년 3월, 4월)	난이도	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2
3주차: 기능 보완, 발표 자료 작성																								
주계약/특약 추가 시 중복방지 기능	★★																							
예상 보험료, 예상 보험금 기능	★★★																							
가입기간 입력칸 추가	★																							
예상 버튼 추가	★																							
계약 조회/등록 버튼 삭제	★																							
병력 검색 기능	★★																							
예상 해지환급금, 예상 만기지급금 기능	☆☆☆																							
신규계약 기본 정보 등록 기능	★★																							
발표자료(PPT) 작업	★★																							
관리자 등록 개편 (별도의 정보입력 패널 생성)	★★																							
로그아웃 기능	★★																							
로그아웃 하지 않을 경우, 다음 로그인 시 메시지 출력	★★																							
로그인 시 관리자 이름 출력	★★																							
가입금액, 가입기간 제한 기능	★★																							
주계약 하나만 등록 가능	★★																							
피보험자 정보가 없는 상태에서도 예상 보험료 출력	★★																							
4주차: 추가 보완 작업, 발표 자료 작성																								
기존 고객을 검색하면 계약정보 표기	★★																							
기존 계약 삭제 버튼	★★																							
예상 패널 활성화	★																							
최종 발표 연습, 대본 작성	★★																							

★ 달성 난이도 상/중/하
☆ 미완료
박정환 (파란색)
이호준 (초록색)
공동작업 (주황색)

⑤ 맺음말

프로젝트 참여자

- 프로젝트 기획: 박정환
- 프로젝트 구성: 박정환
- java 총괄: 박정환
- 데이터베이스 총괄: 이호준
- 발표자료 구성: 이호준
- 기술제휴: 강성대
- 기술고문: 강이영, 서호진

감사합니다

GitHub 주소: https://github.com/jpminlak/Insurance_Project