

KKJ CONO KIOSK!

연수영 이상원 이문관 이용제 김주선

끌까지 간다 PRESENTS

CONTENTS

- 01 작업 일정
- 02 팀원 소개
- 03 프로그램 목적
- 04 화면 및 기능

SCHEDULE

11/28-12/3



11/28

화면 구현
SQL 테스트

11/29-12/1

기능 구현

12/2

오류 점검 및 수정

12/3

최종 점검
테스트

TEAM



연수영

간식 결제 화면



이상원

관리자 화면



이용제

노래방 결제 화면



이문관

로그인, 회원가입 외
각종 팝업창



김주선

인트로 화면
대기(예약)

프로그램 목적

대기 예약부터 결제까지
코인노래방 키오스크의 전 과정 구현

주요 기능

인트로 페이지

로그인
회원 가입

대기(예약)

방 선택 및 결제

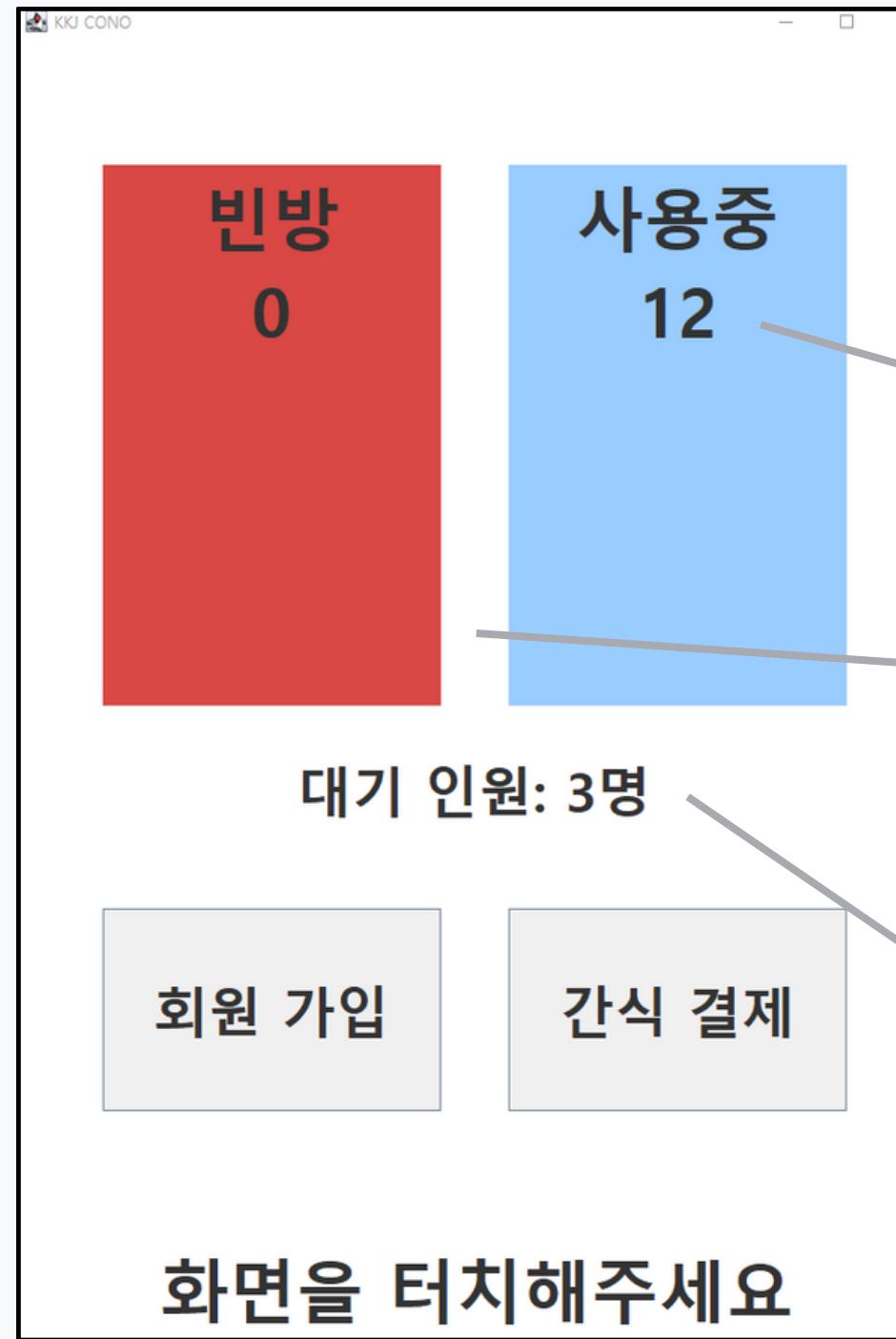
간식 선택 및 결제

관리자 페이지

1. INTRO PAGE



<빈 방 있을 때>



<빈 방 없을 때 >

사용중인 방 개수 실시간 업데이트

화면 터치 시 회원/비회원 선택창으로 연결
빈 방 0일 때는 대기(예약)화면으로 연결

빈 방 0일 때 자동으로 대기인원 수 표시

1.INTRO PAGE

주요 코드

- 1초마다 사용중인 방 개수와 대기인원 업데이트해 화면에 표시

```
// 타이머를 통한 주기적인 업데이트
Timer timer = new Timer(1000, new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        UpdateUsingandVacantRoom();
        UpdateWaiting(); //사용중인 방 개수가 12

    }
});
timer.start();
```

2. WAITING PAGE

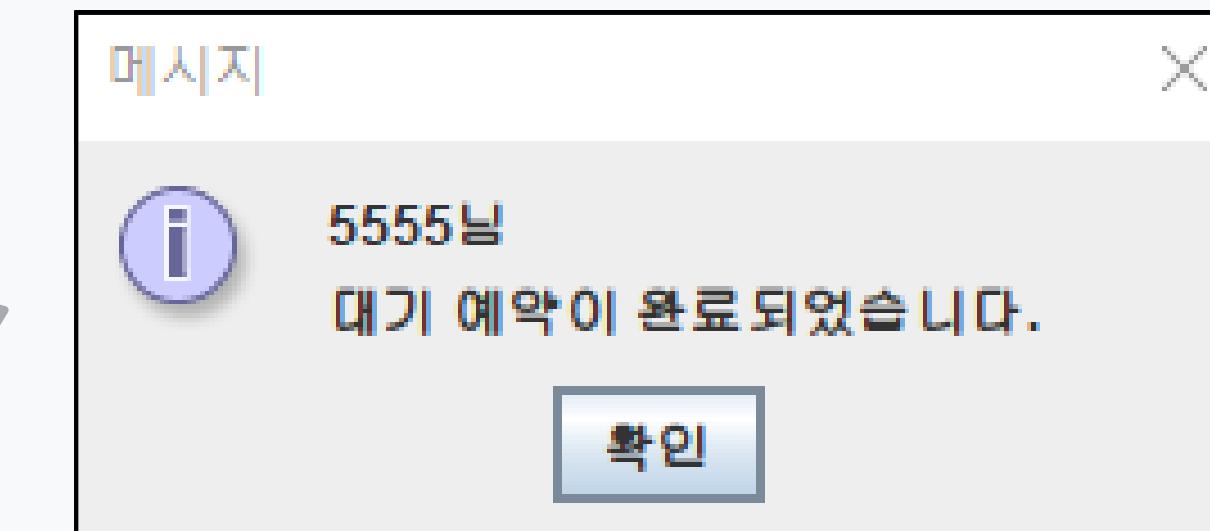
대기 예약

휴대폰 번호를 입력하세요.

1	2	3
4	5	6
7	8	9
←	0	확인

대기 예약

휴대폰 번호가
010으로 시작하고 11글자 이내인 올바른 형식인지 확인하여
조건을 만족한다면 대기자 정보를 등록



대기 예약 성공 시 팝업

2.WAITING PAGE

주요 코드

- 전화번호 형식이 맞는지 확인(010으로 시작, 11자리)

```
public void actionPerformed(ActionEvent e) {  
    phone = jtf_phone.getText();  
    if(phone.substring(0, 3).equals("010") && phone.length()==11) { //올바른 전화번호인지 확인  
        waitinsertmethod(phone);  
        dispose();  
    }else {  
        JOptionPane.showMessageDialog(null, "휴대폰 번호를 알맞게 작성했는지 확인해 주세요");  
        jtf_phone.setText("");  
    }  
}
```

3. MEMBER JOIN PAGE

The image shows a 'Member Join' page interface. At the top left is a logo with the text '회원 가입'. Below it are three input fields: '휴대폰 번호' (Phone Number), '비밀번호' (Password), and '비밀번호 확인' (Password Confirmation). To the right of these fields is a numeric keypad grid with rows labeled 1-3, 4-6, 7-9, and a bottom row with a backspace key ('←'), a zero ('0'), and a pink '확인' (Confirm) button.

휴대폰 번호가 ID 역할

기존 회원 정보와 중복되지 않는
휴대폰번호일 경우 회원가입 성공

비밀번호 확인

두 번 입력한 비밀번호가 일치할 경우
회원 정보 저장

3. MEMBER JOIN PAGE

주요 코드

- 두 번 입력한 비밀번호가 일치하는지 확인

```
if (jpf_password.getText().equals(jpf_passwordCheck.getText()) && !jpf_password.getText().equals("")) {
```

- 비밀번호가 일치하지 않을 경우 문구 띄우고 비밀번호 입력 필드 초기화

```
else {  
    JOptionPane.showMessageDialog(null, "비밀번호를 확인해주세요");  
    jpf_password.setText("");  
    jpf_passwordCheck.setText("");
```

4.LOGIN PAGE

The image shows a login interface window titled "로그인". It contains two input fields: "휴대폰 번호" (Phone Number) and "비밀번호" (Password). Below these is a 4x3 grid numeric keypad. The grid is as follows:

1	2	3
4	5	6
7	8	9
←	0	확인

At the bottom is a "뒤로가기" (Back) button.

로그인

휴대폰 번호, 비밀번호 입력받아
회원 데이터베이스와 비교 후 로그인

노래방 결제 시 로그인 정보 함께 저장

5. ROOM PAY PAGE



남은 시간

이용중인 방은 방마다 실시간으로 남은 시간 표시

방 옵션

방 번호, 방 크기, 마이크 유선/무선 여부

이용중인 방은 선택 불가

요금 선택

방 클릭 시 하단에 요금 선택 버튼 나타남
회원/비회원인지에 따라 요금 차이

결제

버튼 클릭 시 결제 완료

결제 내역 데이터베이스에 저장

5. ROOM PAY PAGE

주요 코드

- 비회원의 경우 회원의 1.2배인 요금 버튼 표시

```
public void actionPerformed(ActionEvent e) {  
  
    int startindex = gottext.indexOf("-");  
    String won = gottext.substring(startindex+1);  
  
    minu = min[a];  
    sale = (int)(pay[a]*1.2);  
  
    jlb_payment.setText(gottext);  
}
```

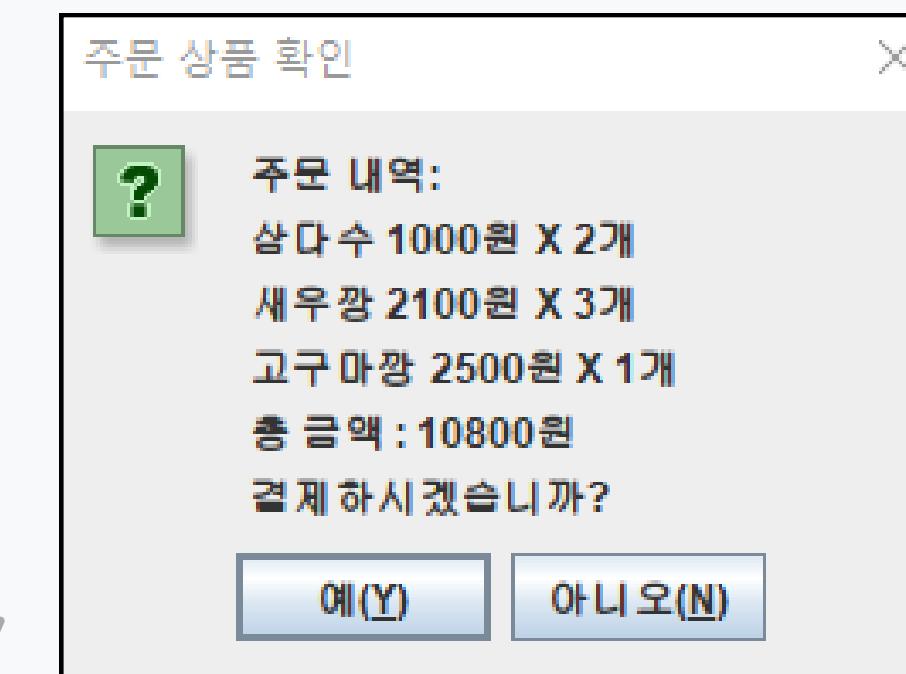
6. SNACK PAY PAGE

The screenshot shows a grid of nine snack items with their prices and quantity selection buttons. Arrows from the text descriptions point to the respective sections of the page.

상품명	단가	수량	합계
삼다수	1000원	2	2000원
새우깡	2100원	3	6300원
고구마깡	2500원	1	2500원
			합계 : 10800원
			전체 삭제
			결제

각 상품의 +,-버튼 눌렀을 때, 표에 수량 업데이트

0개가 되면 아래 목록에서 삭제



결제 버튼 클릭 시 확인 팝업

6. SNACK PAY PAGE

주요 코드

- 상품 여러 개 한번에 결제 시 하나의 주문번호를 갖도록 sql 설정

- FoodPayDAO

```
String sql = "insert into pay values ("  
    + "(select max(pno) + 1 from pay), "  
    + "order+", "  
    + "'p2', "  
    + "100, "  
    + "null, "  
    + "(select fno from foodinfo where f_name = '" + foodname + "'), "  
    + "null, "  
    + sum+, "  
    + count+, "  
    + "sysdate, "  
    + "null);";
```

PNO	ORDER_NUM	PT_C	CNO	RNO	FNO	P_MINUTE	SALEPRICE	P_CNT	PAYTIME	ENDTIME
23	16	p2	100	9			7500	3	23/12/03	
30	19	p2	100	1			1000	1	23/12/03	
31	19	p2	100	5			4000	2	23/12/03	
32	19	p2	100	8			2100	1	23/12/03	
24	17	p2	100	7			10000	5	23/12/03	
25	17	p2	100	8			10500	5	23/12/03	
26	17	p2	100	9			12500	5	23/12/03	

결제내역 테이블에 insert된 모습

- OrderNumDAO

```
String sql = "select max(order_num) from pay";  
int order = 0;
```

6. SNACK PAY PAGE

주요 코드

- 표에 수량 업데이트하고, 0개 이하로 내려가면 표에서 삭제

```
// "-" 버튼 눌렀을 때
btn_minus[i].addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        foodCount = Integer.parseInt(jtf_num[j].getText().trim()); // 현재 수량

        if (foodCount > 0) {
            foodCount = foodCount - 1;
            jtf_num[j].setText(Integer.toString(foodCount));

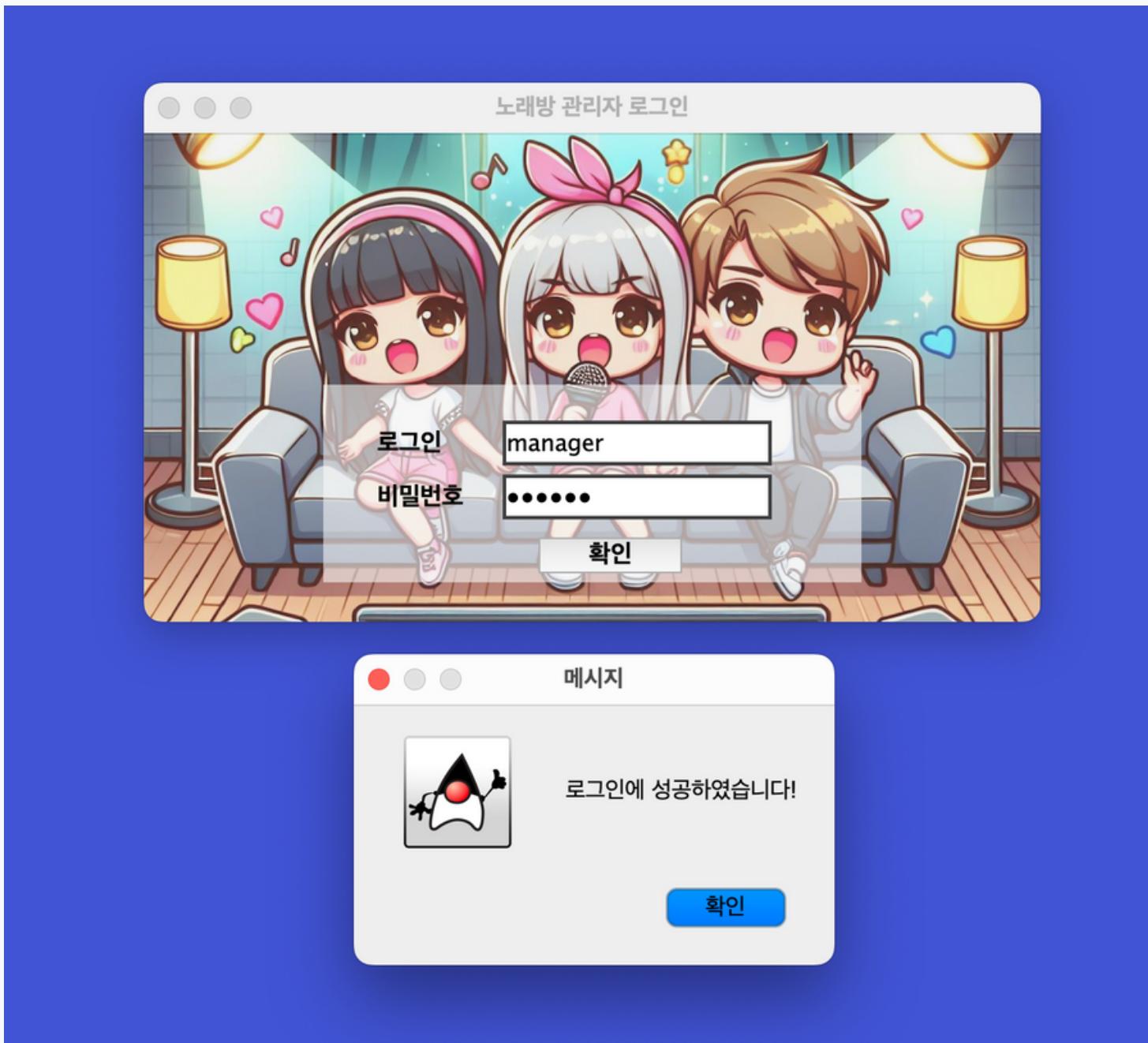
            if (foodCount == 0) {
                // 수량이 0이 되었을 때 표에서 삭제.
                for (int foodIndex = 0; foodIndex < foodSelect.getRowCount(); foodIndex++) {
                    if (foodSelect.getValueAt(foodIndex, 0).equals(foodName[j])) {
                        foodSelect.removeRow(foodIndex);
                        break;
                    }
                }
                btn_minus[j].setEnabled(false);
            } else {
                // 기존상품이 담겨있으면 행을 추가하지 않고 수량, 가격 변경
                int foodIndexToUpdate = -1;
                for (int foodIndex = 0; foodIndex < foodSelect.getRowCount(); foodIndex++) {
                    if (foodSelect.getValueAt(foodIndex, 0).equals(foodName[j])) {
                        foodIndexToUpdate = foodIndex;
                        break;
                    }
                }
            }
        }
    }
});
```

```
if (foodIndexToUpdate != -1) {
    foodTotal = foodTotal - foodPrice[j];
    foodSelect.setValueAt(Integer.toString(foodCount), foodIndexToUpdate, 2);
    // 수량 업데이트
    foodSelect.setValueAt(Integer.toString(foodPrice[j] * foodCount) + "원",
    foodIndexToUpdate, 3); // 합계 업데이트
}
jlb_totalPrice.setText("합계 : " + foodTotal + "원");
});
```

7. MANAGER PAGE

매니저 페이지 전체 화면 - 방, 회원, 결제 관리

매니저 계정 로그인



노래방 관리자

경수증 프린트

「방관리」

방번호:	결제시각:	종료시각:	회원번호:	남은시간:	
Room 1	Room 2	Room 3	Room 4	Room 5	Room 6
Room 7	Room 8	Room 9	Room 10	Room 11	Room 12

「회원관리」

회원번호	회원전화	비밀번호	총이용시간
100	99999999999	1234	1500
101	01011110000	1234	120
102	01022211111	4567	0
103	01033332222	6666	0
105	01055554444	3333	60
106	01066665555	1111	60
107	01088889999	1234	0

** 회원조회결과 **

회원번호	회원전화	비밀번호	총이용시간

회원전화: 비밀번호:
신규가입 회원찾기 정보변경 회원탈퇴

「결제관리」

결제번호	주문번호	결제타입	회원번호	음식명	수량	방번호	이용시간	결제시각	종료시각	결제금액
35	30	노래방결제	101		0	12	60	2023-12-01 00:40:00	2023-12-01 01:40:00	12000
34	29	노래방결제	105		0	2	60	2023-12-01 00:30:00	2023-12-01 01:30:00	6000
33	28	노래방결제	100		0	3	120	2023-12-01 20:30:00	2023-12-01 22:30:00	12000
32	27	노래방결제	100		0	12	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000
31	26	노래방결제	100		0	11	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000
30	25	노래방결제	100		0	10	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000
29	24	노래방결제	100		0	9	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000
28	23	노래방결제	100		0	8	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000
27	22	노래방결제	100		0	7	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000
26	21	노래방결제	100		0	6	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000
25	20	노래방결제	100		0	5	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000

** 결제조회결과 **

결제순서	회원번호	방번호	이용시	음식명	수량	결제금	결제시각
1	100	2	60		0	6000	2023-11-09 20:30:00

(날짜 입력 형식: 11월 23일 2023년 입력시, 23/11/2023) 조회 시작: 조회 마침: 결제 찾기
회원 결제 찾기(전화번호): 회원 찾기
로그아웃

7. MANAGER PAGE

방 관리 & 회원 관리

「 방관리 」

방번호	결제	종료	회원번호	남은시간
3	05:20:00	07:20:00	102	292

Room 1 Room 2 Room 3 Room 4 Room 5 Room 6

Room 7 Room 8 Room 9 Room 10 Room 11 Room 12

방 번호를 선택하여 해당 방의 현재 이용정보 조회

「 회원관리 」

회원번호	회원전화	비밀번호	총이용시간
100	99999999999		1500
101	01011110000	1234	120
102	01022221111	4567	0
103	01033332222	6666	0
105	01055544444	3333	60
106	01066665555	1111	60
107	01088889999	1234	0

** 회원조회결과 **

회원번호	회원전화	비밀번호	총이용시간
102	01022221111	4567	0

회원전화 비밀번호

[신규가입](#) [회원찾기](#) [정보변경](#) [회원탈퇴](#)

신규 회원 가입, 회원 검색, 정보 변경, 탈퇴

7. MANAGER PAGE

주요 코드

- 회원검색시 이용시간이 0인 회원까지 검색되도록 한후 HAVING을 이용하여 특정 멤버를 선택

```
String sql = "SELECT c.cno, c_phone, c_password, SUM(p.p_minute) "
    + "FROM customer c "
    + "LEFT OUTER JOIN pay p "
    + "ON c.cno = p.cno "
    + "GROUP BY c.cno, c_phone, c_password "
    + "HAVING c_phone = '" + phone + "'";
```

- 회원탈퇴시 PAY테이블(자식테이블)에 남아 있는 Customer의 정보를 업데이트해주는 트리거 생성

```
CREATE OR REPLACE TRIGGER update_pay_before_customer_delete
BEFORE DELETE ON customer
FOR EACH ROW
BEGIN
    UPDATE pay p
        SET p.cno = 100
    WHERE p.cno = :OLD.cno;
END;
```

7. MANAGER PAGE

결제 관리

「 결제관리 」

결제번호	주문번호	결제타입	회원번호	음식명	수량	방번호	이용시간	결제시각	종료시각	결제금액
22	17	노래방결제	100		0	2	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000
21	16	노래방결제	106		0	3	60	2023-11-30 21:30:30	2023-11-30 22:30:30	3000
20	15	간식결제	100	사이다	3	0	0	2023-11-26 21:30:00		3600
19	14	간식결제	100	콜라	4	0	0	2023-11-26 20:30:00		6000
18	13	간식결제	100	포카리	2	0	0	2023-11-26 20:30:00		2000
17	13	간식결제	100	양파링	1	0	0	2023-11-26 20:30:00		1500
16	13	노래방결제	100		0	1	60	2023-11-26 20:30:00	2023-11-26 21:30:00	6000
15	12	간식결제	101	사이다	1	0	0	2023-11-26 12:30:00		1200
14	12	간식결제	101	포카칩	3	0	0	2023-11-26 12:30:00		6000
13	12	간식결제	101	새우깡	2	0	0	2023-11-26 12:30:00		2000
12	12	노래방결제	101		0	1	60	2023-11-26 12:30:00	2023-11-26 13:30:00	6000

** 결제조회결과 **

결제순서	회원번호	방번호	이용시	음식명	수량	결제금	결제시각
1	101	1	60		0	6000	2023-11-26 12:30:00
2	101	0	0	새우깡	2	2000	2023-11-26 12:30:00
3	101	0	0	포카칩	3	6000	2023-11-26 12:30:00
4	101	0	0	사이다	1	1200	2023-11-26 12:30:00

(날짜 입력 형식: 11월 23일 2023년 입력시, 23/11/2023) 조회 시작 조회 마침 결제 찾기

회원 결제 찾기(전화번호) 회원 찾기

로그아웃

테이블의 레코드 클릭 시
동일한 주문번호로 묶어서 조회

결제 기간별 조회

결제자 핸드폰 번호로 조회

7. MANAGER PAGE

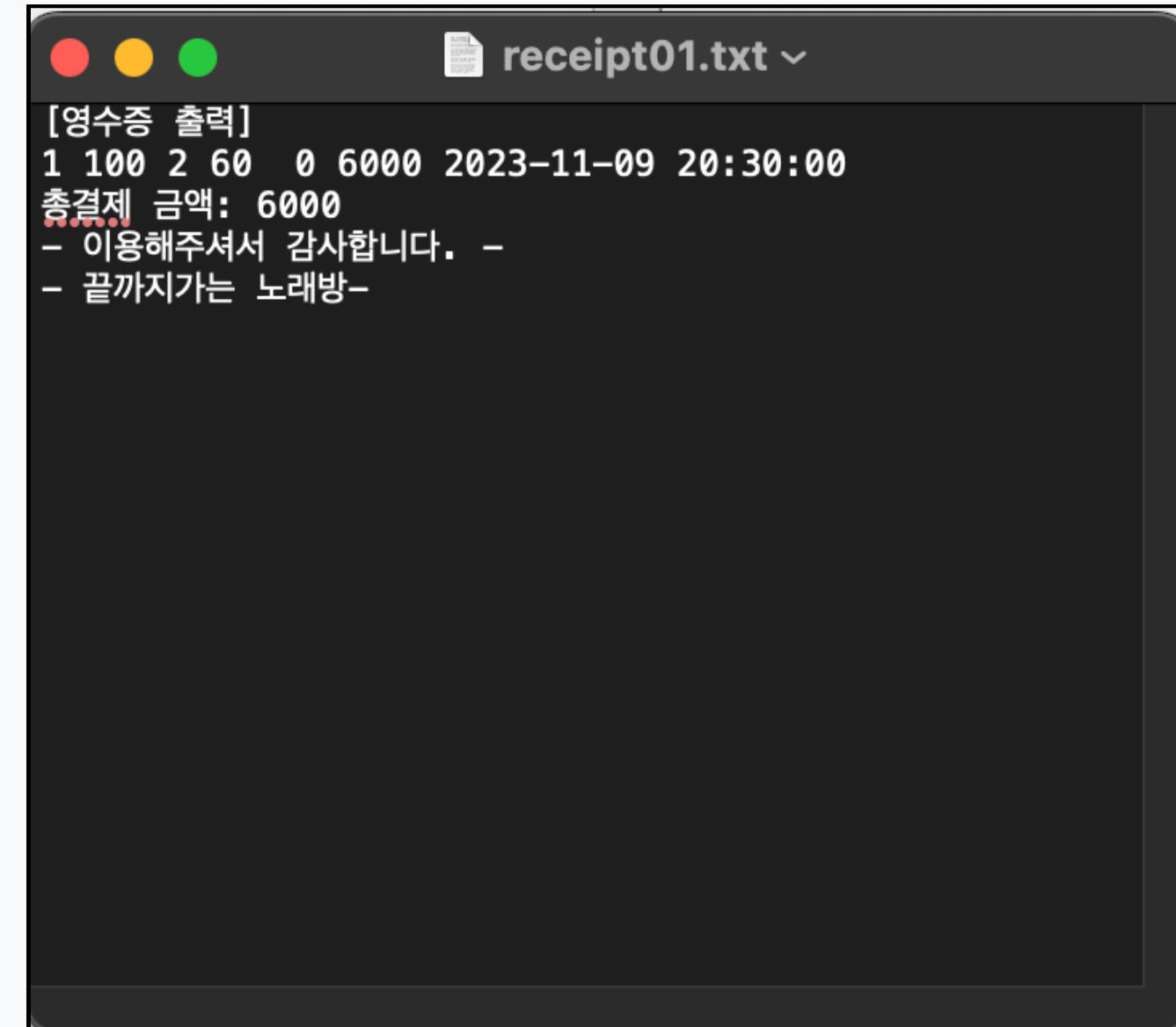
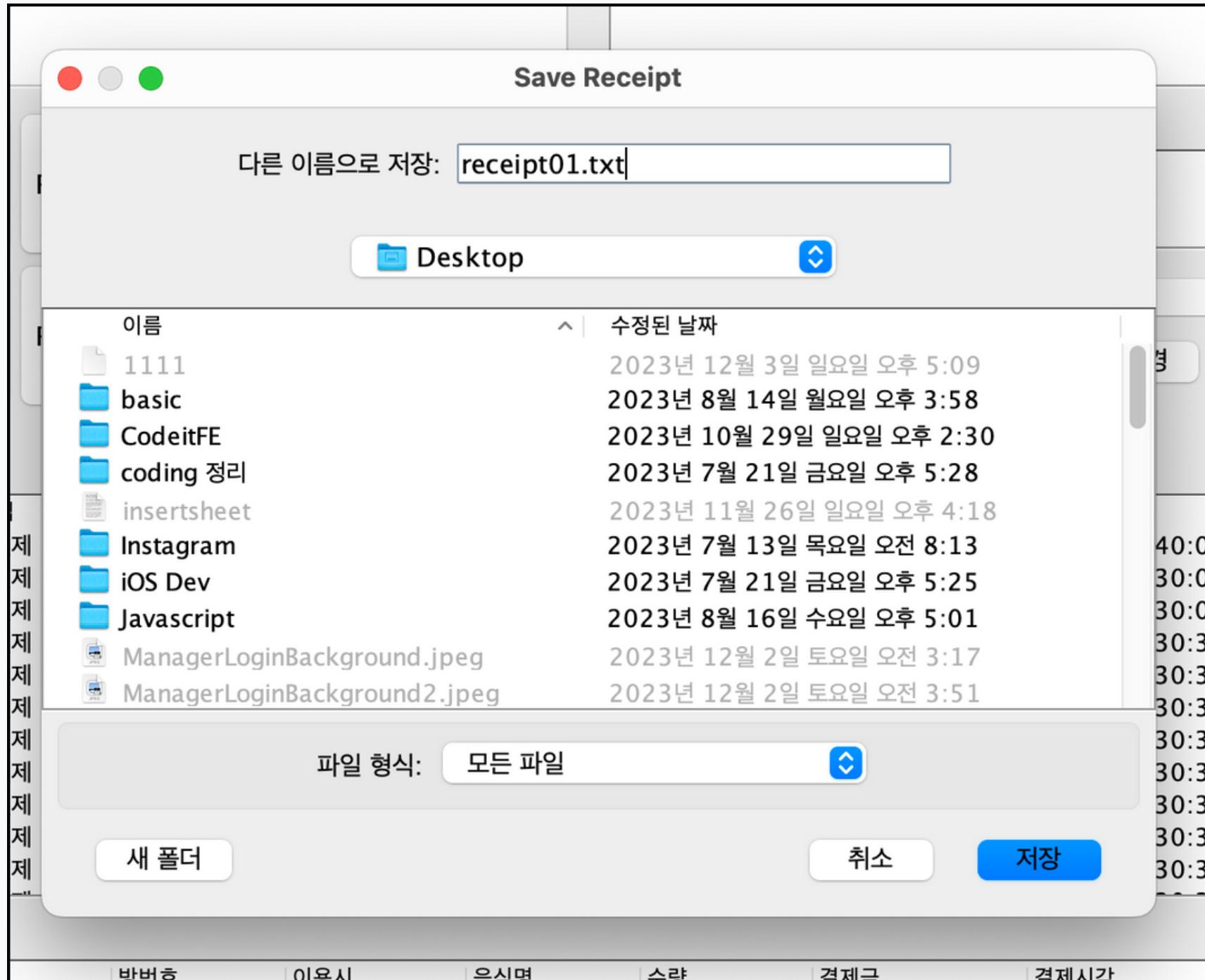
주요 코드

- 주문번호 일치하는 결과만 뽑아내 동시에 결제한 주문들을 모아보기 (음식번호대신 음식이름 표시)

```
String sql = "SELECT ROWNUM, p.CNO, p.RNO, p.P_MINUTE, f.F_NAME, p.P_CNT, p.SALEPRICE, p.PAYTIME "
+ "FROM pay p "
+ "LEFT JOIN foodinfo f ON p.FNO = f.FNO "
+ "WHERE p.ORDER_NUM = "+ orderNumber + "";
```

7. MANAGER PAGE

영수증 출력



영수증
선택한 주문의 결제 내역을 파일로 출력

7. MANAGER PAGE

주요 코드

- 파일로 영수증 출력

파일 저장 경로 지정
(FileChooser 이용)

영수증을 파일로 저장
테이블 데이터를 파일에 기록
(BufferedWriter 이용)

```
try (BufferedWriter writer = new BufferedWriter(new FileWriter(fileToSave))) {  
    writer.write("[영수증 출력]");  
    writer.newLine();  
    int totalSalePrice = 0;  
    for (int i = 0; i < model.getRowCount(); i++) {  
        for (int j = 0; j < model.getColumnCount(); j++) {  
            Object cellValue = model.getValueAt(i, j);  
            String value = (cellValue != null) ? cellValue.toString() : "";  
            writer.write(value + " ");  
            if (j == 6) {  
                try {  
                    totalSalePrice += Integer.parseInt(value);  
                } catch (NumberFormatException e) {  
                    System.out.println("Not a valid integer: " + value);  
                }  
            }  
        }  
        writer.newLine();  
    }  
    writer.write("총결제 금액: " + totalSalePrice + "");  
    writer.newLine();  
    writer.write("- 이용해주셔서 감사합니다. -");  
    writer.newLine();  
    writer.write("- 끝까지가는 노래방 -");  
} catch (Exception e) {  
    e.printStackTrace();  
}
```

**THANK
YOU!**

