비대면을 위한 배달어플

저기요!

JEOGIYO
UNTACT CATERING

목차

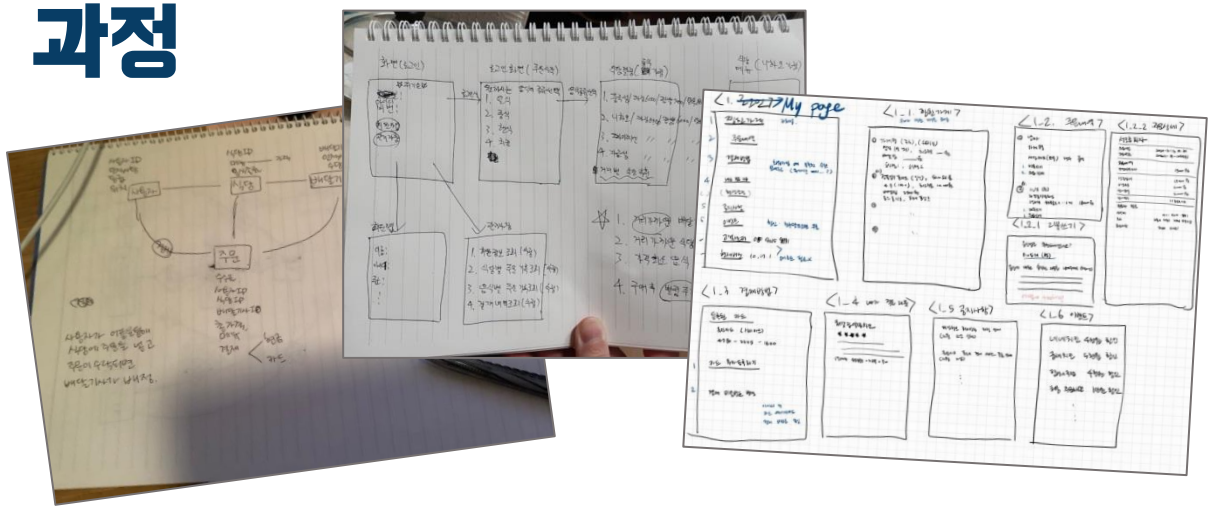
어플 제작 배경

코로나 바이러스로 인한 비대면 서비스의 필요성 대두
→코로나로 인해 비대면을 통한 결제가 17% 증가

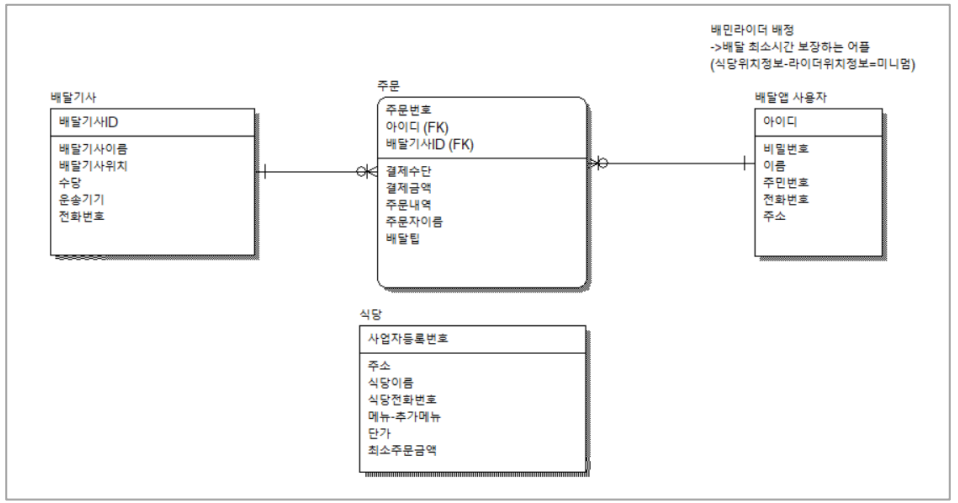
비대면 화상회의 중 배달 음식을 시켜 먹을 계획을
밝힌 팀원의 얘기를 듣고 “우리도 배달 어플을
만들어보자!”라는 의견의 합치를 보고 제작함!

그래서 배달 어플 ‘저기요!’를 만들게 되었습니다.

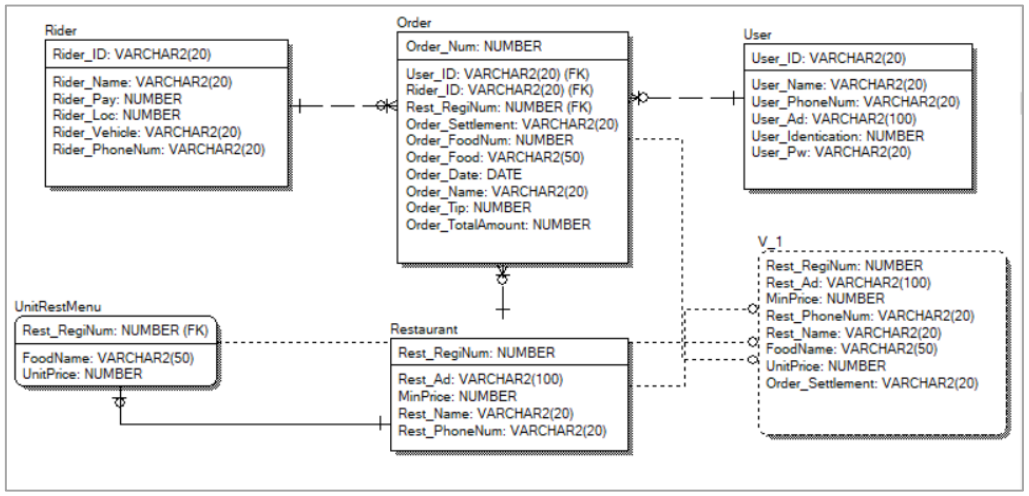
모델링 과정



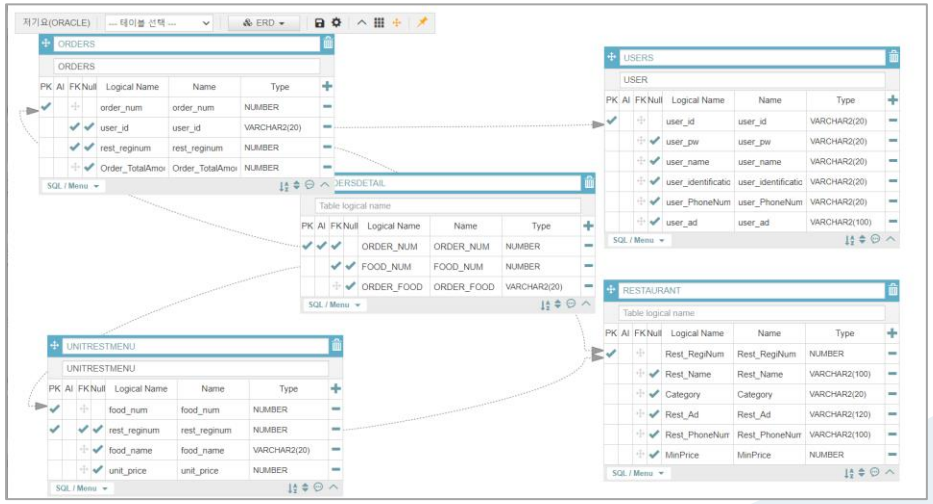
1. 초안



2. Logical



3. Physical



4. 최종안

데이터 상세표

RESTAURANT 테이블

컬럼명	Rest_RegiNum	Rest_Name	Category	Rest_Ad	Rest_PhoneNum	MinPrice
PK/UK/NOT NULL	PK/NOT NULL					
참조 테이블						
참조 컬럼						
CHECK						
데이터 타입	NUMBER	VARCHAR2	VARCHAR2	VARCHAR2	VARCHAR2	NUMBER
길이						
비고		100	20	120	100	

컬럼 설명

Rest_RegiNum	배달앱에 등록된 식당의 사업자등록번호를 의미한다.
Rest_Name	배달앱에 등록된 식당의 이름을 의미한다.
Category	배달앱 내의 식당 분류(한식,양식 등..)를 의미한다.
Rest_Ad	배달앱에 등록된 식당의 주소를 의미한다.
Rest_PhoneNum	배달앱에 등록된 식당의 전화번호를 의미한다.
MinPrice	배달앱에 등록된 식당의 최소배달금액을 의미한다.

USERS 테이블

컬럼명	user_id	user_pw	user_name	user_identification	user_PhoneNum	user_ad
PK/UK/NOT NULL	PK/NOT NULL					
참조 테이블						
참조 컬럼						
CHECK						
데이터 타입	VARCHAR2	VARCHAR2	VARCHAR2	VARCHAR2	VARCHAR2	VARCHAR2
길이						
비고	20	20	20	20	20	100

컬럼 설명

user_id	배달앱 사용자의 아이디를 의미한다.
user_pw	배달앱 사용자의 비밀번호를 의미한다.
user_name	배달앱 사용자의 이름을 의미한다.
user_identification	배달앱 사용자의 주민등록번호를 의미한다.
user_PhoneNum	배달앱 사용자의 전화번호를 의미한다.
user_ad	배달앱 사용자의 주소를 의미한다.

UNITRESTMENU 테이블

컬럼명	food_num	rest_reginum	foodname	unitprice
PK/UK/NOT NULL	PK/NOT NULL	PK		
참조 테이블		RESTAURANT		
참조 컬럼		rest_reginum		
CHECK				
데이터 타입	NUMBER	NUMBER	VARCHAR2	NUMBER
길이				
비고			50	

컬럼 설명

food_num	메뉴의 고유 번호를 의미한다.
rest_reginum	배달앱에 등록된 식당의 사업자등록번호를 의미한다.
foodname	메뉴의 이름을 의미한다.
unitprice	메뉴의 개당 가격을 의미한다.

ORDERS 테이블

컬럼명	order_num	user_id	rest_reginum	Order_TotalAmount
PK/UK/NOT NULL	PK/NOT NULL			
참조 테이블		USERS	RESTAURANT	
참조 컬럼		user_id	rest_reginum	
CHECK				
데이터 타입	NUMBER	VARCHAR2	NUMBER	NUMBER
길이				
비고		20		

컬럼 설명

order_num	배달앱을 통한 주문번호를 의미한다.
user_id	배달앱 사용자의 아이디를 의미한다.
rest_reginum	배달앱에 등록된 식당의 사업자등록번호를 의미한다.
Order_TotalAmount	해당 주문의 총 결제금액을 의미한다.

ORDERSDETAIL 테이블

컬럼명	order_num	food_num	order_food
PK/UK/NOT NULL	NOT NULL		
참조 테이블	ORDERS	UNITRESTMENU	
참조 컬럼	order_num	food_num	
CHECK			
데이터 타입	NUMBER	NUMBER	VARCHAR2
길이			
비고			50

컬럼 설명

order_num	배달앱을 통한 주문번호를 의미한다.
food_num	메뉴의 고유 번호를 의미한다.
order_food	배달앱을 통해 주문한 메뉴의 이름을 의미한다.

기능 개괄

1. 회원가입 - 유효성검사
2. 로그인 - 유효성검사
3. 마이페이지　내 정보, 주문내역
4. 광고
5. 인기음식추천
6. 식당 선택 후 장바구니에 메뉴 담기
7. 결제하기
8. 결제완료창
9. 관리자모드
10. 뒤로가기

반갑습니다 고객님,

조기요를 경험해보시겠어요 ?

코드 설명

회원가입 -유효성검사

```java
// 1.ID
//= ""를 써야 null값이 안들어간다!!
String userId = "";
boolean loop = true;
while (loop) {
    System.out.println("   1.ID :"); // DAO에서 중복검사
    userId = sc.next();
    for (int i = 0; i < userId.length(); i++) {
        if (Character.isLetter(userId.charAt(i)) || ('0' <= userId.charAt(i) && userId.charAt(i) <= '9')) {
            loop = false;
        } else {
            System.out.println("ID에 특수문자와 공란은 입력할 수 없습니다.");
            loop = true;
            break;
        }
    }
}

System.out.println("");

// 2.PW
String userPw = "";
while (true) {
    String pattern = "^[0-9]*$";
    System.out.println("   2.PASSWORD :");
    userPw = sc.next();
    if (userPw.matches(pattern) && userPw.length() <= 10) {
        break;
    } else {
        System.out.println("숫자만 입력 가능하고 10글자 이하로만 입력가능합니다.");
        continue;
    }
}
System.out.println("");
```

회원가입 -유효성검사

```java
// 3.name
String userName = "";
while (true) {
    String pattern = "^[가-힣]*$";
    System.out.println("   3.이름 :"); // 한글
    userName = sc.next();
    if (userName.matches(pattern) && userName != "") {
        break;
    } else {
        System.out.println("이름에 특수문자, 영어, 공란은 입력할 수 없습니다.");
        continue;
    }
}
System.out.println("");

// 5.userPhoneNum
String userPhoneNum = "";
while (true) {
    String pattern = "^[0-9]*$";
    System.out.println("   5.전화번호 :"); // 글자수
    userPhoneNum = sc.next();
    if (userPhoneNum.matches(pattern) && userPhoneNum.length() == 11) {
        break;
    } else {
        System.out.println("전화번호는 11글자 입니다!");
        continue;
    }
}
System.out.println("");

// 6.주소
System.out.print("6.주소 : ");
String userAd = sc.next();
sc.nextLine();
```

JEOGIYO
UNTACT CATERING

회원가입 -유효성검사

```java
// 4.identification
String identification = "";
boolean boo = true;
do {
    System.out.println("   4.주민번호 :"); // 13글자 형태
    identification = sc.next();
    if (identification.length() == 14) {
        if (identification.charAt(6) == '-') {
            boolean check = true;
            for (int i = 0; i < 14; i++) {
                if (i != 6) {
                    char ch = identification.charAt(i);
                    if (ch < '0' || ch > '9') {
                        check = false;
                    }
                }
            }
            if (check) {
                boo = false;
            } else {
            }
        } else {
            System.out.println("주민번호형식은 123456-124567입니다");
        }
    } else {
        System.out.println("주민번호형식은 123456-124567입니다");
    }
} while (boo);// 주민번호 형식판단
System.out.println("");
```

결제완료창

```
void end() {
    //배달소요시간
    Random rd = new Random();
    int time = rd.nextInt(31)+30;
    View.end(time,basket, OrderNum);

    System.out.println("시작화면으로 돌아가시겠습니까?(y/n)");
    String yn = sc.next();
    if(yn.equals("y")) {
       view.showStart();
    }else {
       System.out.println("이용해주셔서 감사합니다!");
    }
}
```

Class Controller

Class View

```
public static void end(int time, ArrayList<VO> basket, int OrderNum) {

    System.out.println("[redacted]  저 기 요  [redacted]");
    System.out.println("");
    System.out.println("\t\t결  제  완  료                    ");
    System.out.println("");
    System.out.println("\t\t배달 예상 소요시간  : " + time + " 분                ");
    System.out.println("");
    System.out.println("\t\t주문번호 : " + OrderNum);
    System.out.println("");
    System.out.println("\t\t식당 : " + basket.get(0).getRestName());
    for (int j = 0; j < basket.size(); j++) {
       System.out.println("\t\t메뉴 : " + basket.get(j).getFoodName());
    }
    System.out.println("");
    System.out.println("");
    System.out.println("[redacted]");
    System.out.println();
}
```

JEOGIYO
UNTACT CATERING

DAO 로그인

*Controller.java | *DAO.java | *LoginVO.java | View.java | jeogiyo_Main.java

```
39      // 1.로그인
40      public List listMembers1() {
41          List list = new ArrayList();
42
43          try {
44              conn(); // 오라클 연결
45              String userSelect = "SELECT * FROM users ";      //쿼리문
46              preStmt = con.prepareStatement(userSelect); //오라클에 접속해서 쿼리문 실행 (순서 맨 마지막 필수)
47
48              rs = preStmt.executeQuery();  //오라클 연결, 명령된 쿼리문의 내용들을 rs에 담기
49
50              while (rs.next()) { // 테이블의 행단위로 하나씩 가져옵니다
51                  String UserId = rs.getString(1);
52                  String UserPw = rs.getString(2);
53
54                  LoginVO loginvo = new LoginVO();
55                  loginvo.setUserId(UserId);
56                  loginvo.setUserPw(UserPw);
57
58                  list.add(loginvo);
59
60              }
61              rs.close();
62              preStmt.close();
63              con.close();
64
65          } catch (SQLException e) {
66              e.printStackTrace();
67          }
68          return list;
69      }
```

DAO 마이페이지(개인 정보조회)

*Controller.java | *DAO.java | *LoginVO.java | View.java | jeogiyo_Main.java

```
71      // 2. 마이페이지 조회
72      public List listMember(String USER_ID) {     //매개변수로 컨트롤러에 연결
73          List list = new ArrayList();
74
75          try {
76              conn(); // 오라클 연결
77              String userSelect = "SELECT * FROM users where user_id = ?";     //해당되는 id의 전체내역을 가져오기
78              preStmt = con.prepareStatement(userSelect);      //쿼리실행
79              preStmt.setString(1, USER_ID);       // 첫번째 물음표에 필요한 user_id의 내용이 가져오게 됨
80              rs = preStmt.executeQuery();         //조건들 다 담은 데이터를 rs에 담기
81
82              while (rs.next()) { // // 테이블의 행단위로 하나씩 가져옵니다
83                  String UserId = rs.getString(1);
84                  String UserPw = rs.getString(2);
85                  String UserName = rs.getString(3);
86                  String UserPhoneNum = rs.getString(5);
87                  String UserAd = rs.getString(6);
88
89                  LoginVO loginvo = new LoginVO();
90                  loginvo.setUserId(UserId);
91                  loginvo.setUserPw(UserPw);
92                  loginvo.setUserName(UserName);
93                  loginvo.setUserPhoneNum(UserPhoneNum);
94                  loginvo.setUserAd(UserAd);
95
96                  list.add(loginvo);
97              }
98              rs.close();
99              preStmt.close();
100             con.close();
101
102         } catch (SQLException e) {
103             e.printStackTrace();
104         }
105         return list;
106     }
```

DAO 마이페이지(주문 내역 조회)

Controller.java | *DAO.java | *LoginVO.java | View.java | jeogiyo_Main.java

```
107
108     // 2. 주문내역
109     public List listOrders(String USER_ID) {
110
111         List list = new ArrayList();
112         try {
113             conn();
114             String userSelect = "SELECT orders.order_num, FOOD_NUM, order_food, user_id, rest_reginum, order_totalamount \r\n"
115                     + "FROM ordersdetail, orders\r\n" + "WHERE orders.order_num = ordersdetail.order_num\r\n"
116                     + "         and user_id = ? ";
117             preStmt = con.prepareStatement(userSelect);
118             preStmt.setString(1, USER_ID);
119             rs = preStmt.executeQuery();
120
121             while (rs.next()) { //  테이블의 행단위로 하나씩 가져옴
122                 int Order_Num = rs.getInt(1);
123                 int Order_Foodnum = rs.getInt(2);
124                 String Order_Food = rs.getString(3);
125                 String user_id = rs.getString(4);
126                 int rest_reginum = rs.getInt(5);
127                 int Order_TotalAmount = rs.getInt(6);
128
129                 LoginVO loginvo = new LoginVO();
130
131                 loginvo.setOrder_Num(Order_Num);
132                 loginvo.setOrder_Foodnum(Order_Foodnum);
133                 loginvo.setOrder_Food(Order_Food);
134                 loginvo.setUserId(user_id);
135                 loginvo.setRest_reginum(rest_reginum);
136
137                 loginvo.setOrder_TotalAmount(Order_TotalAmount);
138
139                 list.add(loginvo);
140
141             }
142             rs.close();
143             preStmt.close();
144             con.close();
145
146         } catch (SQLException e) {
147             e.printStackTrace();
148         }
149         return list;
150     }
151
152 }
153
```

Controller 로그인

*Controller.java | *DAO.java | *LoginVO.java | View.java | jeogiyo_Main.java

```
29      // 2. 로그인
30      void showStart() {
31          DAO dao = new DAO();
32          List list = dao.listMembers1(); // 로그인메소드
33
34          System.out.print("아이디를입력하세요 ");
35          USER_ID = sc.next(); // 아이디 입력. 전역변수 USER_ID에 저장
36          System.out.print("비밀번호를 입력하세요 ");
37          USER_PW = sc.next(); // 비밀번호 입력. 전역변수 USER_PW에 저장
38          ;
39
40          for (int i = 0; i < list.size(); i++) { //연결된 데이터베이스에 존재하는지 비교
41              LoginVO loginvo = (LoginVO) list.get(i);
42
43              if(loginvo.getUserId().equals(USER_ID) && loginvo.getUserPw().equals(USER_PW)) {// 입력한 값과 동일한 데이터 찾기
44                  System.out.println("로그인되었습니다! ");
45              } else if (loginvo.getUserId().equals(USER_ID)||loginvo.getUserPw().equals(USER_PW)) {
46                  System.out.println("로그인 실패! ");
47                  System.out.println("회원가입 하시겠습니까? y / n ");
48                  String yn = sc.next();
49                  View view = new View() ;
50                  view.showStart();
51
52                  System.out.println("-------------------------");
53                  // menu(ID); //로그인이 된 경우, 메인페이지
54                  }else {
55
56                  }
57              }
58          }
```

```
[redacted]  저 기 요  [redacted]

˚+o。◈。o+˚+o。◈。o+  LOGIN  +o。◈。o+˚+o。◈。o+

        1. ID :

        2. PASSWORD :

[redacted]
아이디를입력하세요 yein2
비밀번호를 입력하세요 2222
로그인되었습니다!
[redacted]  저 기 요  [redacted]
```

```
[redacted]  저 기 요  [redacted]

˚+o。◈。o+˚+o。◈。o+  LOGIN  +o。◈。o+˚+o。◈。o+

        1. ID :

        2. PASSWORD :

[redacted]
아이디를입력하세요 yein2
비밀번호를 입력하세요  9999
가입되지 않은 아이디입니다.
회원가입 하시겠습니까? y / n
```

Controller - 마이페이지

DAO.java | *Controller.java | JeoKiYo.java | ArrayList.class | ManagerDAO.java

```
102
103     // 2. 마이페이지(예인)
104     public void mypage() {
105         LoginDAO dao = new LoginDAO();
106         List list = dao.listMember(UserId); // 매개변수로 연결
107         List Orders = dao.listOrders(UserId);
108
109         System.out.println("[redacted]  저 기 요  [redacted]");
110         System.out.println("                                          ");
111         System.out.println("------------------------------------------");
112         System.out.println("                내 정보                   ");
113         System.out.println("------------------------------------------");
114
115         for (int i = 0; i < list.size(); i++) {
116             LoginVO loginvo = (LoginVO) list.get(i); // 멤버 정보 //데이터베이스화 한 내용을 에 전달, 그 안에 있는 내용 출력됨
117             System.out.println("아이디는 [" + loginvo.getUserId() + "] 입니다");
118             System.out.println("이름은 [" + loginvo.getUserName() + "] 입니다");
119             System.out.println("전화번호는 [" + loginvo.getUserPhoneNum() + "] 입니다");
120             System.out.println("주소는 [" + loginvo.getUserAd() + "] 입니다");
121         }
122         System.out.println("                                          ");
123         System.out.println("------------------------------------------");
124         System.out.println("              지난 주문 내역              ");
125         System.out.println("------------------------------------------");
126
127         if (Orders.isEmpty()) {
128             System.out.println("아직 주문한 내역이 없습니다");
129         } else {
130             for (int i = 0; i < Orders.size(); i++) {
131                 LoginVO loginvo = (LoginVO) Orders.get(i); // 주문내역
132
133                 System.out.println("--------------" + i + "번 주문 입니다 --------------");
134                 System.out.println("      주문건의 번호 [" + loginvo.getOrder_Num() + "] 입니다       ");
135                 System.out.println("      음식번호는 [" + loginvo.getOrder_Foodnum() + "] 입니다     ");
136                 System.out.println("       음식이름은 [" + loginvo.getOrder_Food() + "] 입니다       ");
137                 System.out.println("     총 금액은 [" + loginvo.getOrder_TotalAmount() + "] 입니다   ");
138             }
139         }
140         System.out.println("카테고리 화면으로 돌아가시겠습니까?(y/n)");
141         String yn = sc.next();
142         if (yn.equals("y"))
143             categoryRun();
144     }
```

```
[redacted]  저 기 요  [redacted]

------------------------------------------
            내 정보
------------------------------------------
아이디는 [yeiny] 입니다
이름은 [예인] 입니다
전화번호는 [01042158888] 입니다
주소는 [서울시 은평구 불광 ] 입니다

------------------------------------------
          지난 주문 내역
------------------------------------------
아직 주문한 내역이 없습니다
카테고리 화면으로 돌아가시겠습니까?(y/n)
y
[redacted]  저 기 요  [redacted]
```

```
[redacted]  저 기 요  [redacted]

------------------------------------------
            내 정보
------------------------------------------
아이디는 [yeiny] 입니다
이름은 [예인] 입니다
전화번호는 [01042158888] 입니다
주소는 [서울시 은평구 불광 ] 입니다

------------------------------------------
          지난 주문 내역
------------------------------------------
--------------0번 주문 입니다 --------------
    주문건의 번호 [210] 입니다
    음식번호는 [1102] 입니다
     음식이름은 [삼겹살김치찜] 입니다
   총 금액은 [31100] 입니다
카테고리 화면으로 돌아가시겠습니까?(y/n)
```

코드리뷰 카테고리

```
void categoryRun(){
    //카테고리 화면 출력
    Random rd = new Random();
    int num = rd.nextInt(5);
    View.foodCategories(num);

    int choice = VO.getSetMenu();
    switch(choice) {
    case 1:
        Category = "한식";
        break;
    case 2:
        Category = "중식";
        break;
    case 3:
        Category = "일식";
        break;
    case 4:
        Category = "일품";
        break;
    default :
        System.out.println("잘못 선택하셨습니다.");

        }

    if(Category.equals("")) {
        System.out.println("아무것도 입력되지 않았습니다.");
    }else {
        restInfoRun();
    }

}   //식당 카테고리 선정 -> 식당 리스트로 들어가기
```

```
public static void foodCategories(int num) {

    System.out.println("▒▒▒▒▒▒▒▒▒▒▒▒  저 기 요  ▒▒▒▒▒▒▒▒▒▒▒▒");
    System.out.println("                                          ");
    System.out.println("------------------------------------------");
    System.out.println();
    switch (num) {
    case 0:
        System.out.println(" {\\___/} ৬ ৬ ৬ ৬ ৬ ৬ ৬ ৬ ৬ ৬ ৬ ৬ ");
        System.out.println(" ( • ▽•)          '비트 치킨'                ");
        System.out.println(" /っ৬       월요일은 허니 콤보 먹는 날          ");
        break;
    case 1:
        System.out.println("  @▷□◁@  🏙 🏙 🏙 🏙 🏙 🏙 🏙 🏙 🏙 🏙");
        System.out.println(" (๑◕‿‿◕๑)          '달콤한 디저트 데이'");
        System.out.println(" /   っ 🏙       화요일은 달콤한 케이크 어떠세요?");
        break;
    case 2:
        System.out.println(" ▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬");
        System.out.println("( ◉_◉ )수요일은 코딩하는날 비트컴퓨터로 코딩하러오세요");
        System.out.println("/っ💀▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬");
        break;
    case 3:
        System.out.println(" ♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡");
        System.out.println("(๑•ㅂ•๑)✧ 목요일에는 비트 떡튀이 먹는날!!!!!!!!!!!");
        System.out.println("/っ 🍥  ♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡♥♡");
        break;
    case 4:
        System.out.println(" 🍺🍺🍺**************************");
        System.out.println("( ◉_◉ )**************************");
        System.out.println("/っ💀금요일밤에는 치킨엔 맥주!**************** //광고
        break;
    }
```

1.랜덤함수를 통해 광고가 출력

2. VO의 getSetMenu 메소드를 호출해서
   Choice에 카테고리값을 입력받음

```
public static int getSetMenu() {
    System.out.println("번호를 선택해주세요 ☞");
    return sc.nextInt();
}//번호선택메소드
```

```
▒▒▒▒▒▒▒▒▒▒▒▒  저 기 요  ▒▒▒▒▒▒▒▒▒▒▒▒

------------------------------------------

▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬
( ◉_◉ )수요일은 코딩하는날 비트컴퓨터로 코딩하러오세요
/っ💀▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬

------------------------------------------
*+†+*——*+†+*——*+†+*—*+†+*——*+†+*——*+†+*
<인기 음식 추천!!> 웰빙김치찜의 삼겹살김치찜
<인기 음식 추천!!> 스시노하나의 연어초밥
<인기 음식 추천!!> 큰맘할매순대국의 순대국
*+†+*——*+†+*——*+†+*—*+†+*——*+†+*——*+†+*
------------------------------------------


        1. 한식              2. 중식

        3. 일식              4. 일품

------------------------------------------
                5.마이페이지 가기
                0.뒤로가기
------------------------------------------

▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒

번호를 선택해주세요 ☞
1
```

코드리뷰 - 식당선택

```java
VO[] restList(String Category, VO[] infoCate) {
    List list = dao.restList(Category);
    for (int i = 0; i < list.size(); i++) {
        infoCate[i] = (VO) list.get(i);
    }
    return infoCate;
}//category를통한 식당 정보 불러오기
```

```java
void restInfoRun() {

    infoCate = restList(Category, infoCate);
    View.restaurantsList(infoCate);

    int choice = VO.getSetMenu();

    switch(choice) {
    case 0:
        categoryRun();
        restInfoRun();
        break;
    case 1:
        RestName = infoCate[choice-1].getRestName();
        break;
    case 2:
        RestName = infoCate[choice-1].getRestName();
        break;
    case 3:
        RestName = infoCate[choice-1].getRestName();
        break;
    case 4:
        RestName = infoCate[choice-1].getRestName();
        break;
    default :
        System.out.println("잘못 선택하셨습니다.");
    }

    orderRun();
}//식당 리스트에서 식당 메뉴로 넘어가기
```

```java
public List restList(String category) {
    List list = new ArrayList();

    try {
        conn();
        // 셀렉트문 입력
        String userSelect = "SELECT rest_name, minprice, category" +
                            " FROM restaurant " +
                            " WHERE category = ?";

        preStmt = con.prepareStatement(userSelect);
        preStmt.setString(1, category);
        rs = preStmt.executeQuery();

        while (rs.next()) {

            String RestName = rs.getString("rest_name");
            String Category = rs.getString("category");
            int MinPrice = rs.getInt("minprice");

            VO vo = new VO();
            vo.setRestName(RestName);
            vo.setMinPrice(MinPrice);
            vo.setCategory(Category);

            list.add(vo);

        }
        rs.close();
        preStmt.close();
        con.close();

    } catch (SQLException e) {
        e.printStackTrace();
    }
    return list;
}//각 카테고리 식당 리스트 select 검색
```

```java
public static void restaurantsList(VO[] infoCate) {
    String emo = null;
    switch (infoCate[0].getCategory()) {
    case "한식":
        emo = "🍚";
        break;
    case "중식":
        emo = "🥟";
        break;
    case "일식":
        emo = "🍣";
        break;
    case "일품":
        emo = "🍗";
        break;
    }

    System.out.println("████████████   저 기 요  ████████████");
    System.out.println("-------------------------------------------");
    System.out.println("\t\t      " + infoCate[0].getCategory() + emo + "\t\t   ");
    System.out.println("-------------------------------------------");
    System.out.println();
    for (int i = 0; i < infoCate.length; i++) {
        System.out.print(" " + (i + 1) + ". " + infoCate[i].getRestName() + "\t\t");
        System.out.print("최소 주문금액 : " + infoCate[i].getMinPrice());
        System.out.println();
        System.out.println();
    }
    System.out.println();
    System.out.println("                0.뒤로가기                ");
    System.out.println();
    System.out.println("██████████████████████████████████");
    System.out.println();
} // 식당리스트
```

```
████████████   저 기 요   ████████████
-------------------------------------------
                한식🍚
-------------------------------------------

 1. 웰빙김치찜          최소 주문금액 : 10000

 2. 엄마식당            최소 주문금액 : 10000

 3. 쫄면주는삼겹본능          최소 주문금액 : 10000

 4. 큰맘할매순대국        최소 주문금액 : 10000


            0.뒤로가기

██████████████████████████████████

번호를 선택해주세요 👉
3
```

1. DAO의 restList를 통해 입력된 카테고리의 식당정보를 받아옴
2. View단을 통해서 출력
3. 스위치문을 통해 식당선택

JEOGIYO

코드리뷰 - 메뉴리스트

```
VO[] restInfo(String RestName, VO[] infoRest) {
    List list = dao.restInfo(RestName);
    for (int i = 0; i < list.size(); i++) {
        infoRest[i] = (VO) list.get(i);

    }
    return infoRest;
}//restInfo를 통한 해당 식당의 총 정보 불러오기
```

```
void orderRun() {
    int MinPrice = infoRest[0].getMinPrice();
    infoRest = restInfo(RestName, infoRest);
    View.restaurantMenu(infoRest);
    Total = 0;
    int cnt = 0;
    basket.clear();
    sameFood.clear();
    //메뉴선택 시작
    System.out.println("주문하실 메뉴를 선택해주세요 ☞");
    while(true) {
        choice = VO.getOrder();
        int same = 1;
        int key = 0;

        if(choice == 0) {

            System.out.println("다른 식당보기로 이동하시면 장바구니가 초기화 됩니다 ");
            restInfoRun();
            orderRun();
        }
```

```
public List restInfo(String Rest) {
    List list = new ArrayList();

    try {
        conn();
        // 셀렉트문 입력
        String userSelect = "SELECT  r.rest_reginum, r.rest_name, r.rest_ad, r.rest_phonenum, u.food_num, u.foodname, u.unitprice, r.minprice" +
                            " FROM restaurant r, unitrestmenu u" +
                            " WHERE r.rest_reginum = u.rest_reginum" +
                            " AND r.rest_name = ?";
        preStmt = con.prepareStatement(userSelect);
        preStmt.setString(1, Rest);
        rs = preStmt.executeQuery();
        while (rs.next()) {
            String RestName = rs.getString("rest_name");
            String RestAddress = rs.getString("rest_ad");
            String FoodName = rs.getString("foodname");
            String Phone = rs.getString("rest_phonenum");
            int RestRegiNum = rs.getInt("rest_reginum");
            int MinPrice = rs.getInt("minprice");
            int UnitPrice = rs.getInt("unitprice");
            int FoodNum = rs.getInt("food_num");
            VO vo = new VO();
            vo.setRestName(RestName);
            vo.setRestAddress(RestAddress);
            vo.setFoodName(FoodName);
            vo.setRestRegiNum(RestRegiNum);
            vo.setPhone(Phone);
            vo.setMinPrice(MinPrice);
            vo.setUnitPrice(UnitPrice);
            vo.setFoodNum(FoodNum);
            list.add(vo);
        }
        rs.close();
        preStmt.close();
        con.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return list;
}//각 카테고리 식당 메뉴 select 검색
```

```
public static void restaurantMenu(VO[] infoRest) {

    System.out.println("[redacted]  저 기 요  [redacted]");
    System.out.println("-----------------------------------------");
    System.out.println("\t\t" + infoRest[0].getRestName() + "\t\t     ");
    System.out.println("-----------------------------------------");
    System.out.println(" 최소 주문금액 : " + infoRest[0].getMinPrice());
    System.out.println(" 주소 : " + infoRest[0].getRestAddress());
    System.out.println(" 전화번호 : " + infoRest[0].getPhone());
    System.out.println("-----------------------------------------");
    System.out.println("                메뉴                ");
    System.out.println();
    for (int i = 0; i < infoRest.length; i++) {
        System.out
                .println((i + 1) + ". " + infoRest[i].getFoodName() + "\t\t\t" + infoRest[i].getUnitPrice() + "원");
    }

    System.out.println();
    System.out.println("-----------------------------------------");
    System.out.println("\t\t 11. 장바구니로 가기 ");
    System.out.println("\t\t 0. 뒤로가기");
    System.out.println();
    System.out.println("[redacted]");
    System.out.println();
}
```

```
[redacted]  저 기 요  [redacted]
-----------------------------------------
          쫄면주는삼겹본능
-----------------------------------------
최소 주문금액 : 10000
주소 : 서울 서대문구 창천동 33-43
전화번호 : 5033334444
-----------------------------------------
          메뉴

1. 구이삼겹                14000원
2. 간장구이삼겹            15000원
3. 대창구이삼겹            14000원
4. 김치구이삼겹            14500원
5. 매운구이삼겹            14000원
6. 구이삼겹                14000원
7. 대창덮밥                7000원
8. 참치마요밥              8000원
9. 깍두기볶음밥            4000원
10. 음료수                 1000원

-----------------------------------------
          11. 장바구니로 가기
          0. 뒤로가기

[redacted]

주문하실 메뉴를 선택해주세요 ☞
1
```

1. DAO에서 선택된 식당의 정보를 받아옴
2. View를 통해 출력
3. 뒤로가면 장바구니를 초기화 시키기위해 clear()메소드 사용

JEOGIYO
UNTACT CATERING

코드리뷰 메뉴 선택

```
    if(choice == 11) //장바구니로 넘어가기
        break;

    if(cnt == 0) {
        basket.add(infoRest[choice-1]);
        sameFood.put(0, 1);
    }else {
        for (int i = 0; i < basket.size(); i++) {      //선택된 메뉴 중복인지 검사
            if(basket.contains(infoRest[choice-1])) {
                if(infoRest[choice-1].getFoodName().equals(basket.get(i).getFoodName())) {
                    System.out.println("i : "+ i);
                    System.out.println("전에시킨 갯수 : " + sameFood.get(i));
                    same += sameFood.get(i);
                    sameFood.put(i, same);
                    System.out.println("총 시킨 갯수 : " + sameFood.get(i));
                    key = i;
                    break;
                }
            }else { //중복이 아니라면 추가
                basket.add(infoRest[choice-1]);
                sameFood.put(basket.size()-1, same);
                System.out.println("새로추가된 메뉴 시킨갯수 : "+ sameFood.get(basket.size()-1));
                key = basket.size()-1;
                break;
                }
            }
        } //중복검사 끝

        Total += infoRest[choice-1].getUnitPrice();
        System.out.println(infoRest[choice-1].getFoodName() + " " + sameFood.get(key) + "개가 장바구니에 추가됐습니다.");
        System.out.println("총 금액 : " + Total);
        System.out.println("-----------------------------------------");
        cnt++; //총 몇개시키는지 셈
        if( Total < infoRest[choice-1].getMinPrice()) {  //주문금액이 주문최소금액보다 작으면 더골르라는 팝업
            View.orderPopUp();
        }else{  //최소금액보다 많이 나오면 결제하겠냐고 물어보기
            System.out.println("장바구니로가려면 11번을 눌러주세요☞");
            System.out.println();
            System.out.println("계속 고르려면 추가 할 메뉴번호를 눌러주세요 ☞");
            System.out.println("-----------------------------------------");
        }
    }
    myBasket();
//결제창
```

```
public static void orderPopUp() {
    System.out.println("-----------------------------");
    System.out.println("|                           |");
    System.out.println("| 최소주문 금액이 넘어야 결제가 됩니다 |");
    System.out.println("|       ㅠㅠ 더 추가해주세요 ㅠㅠ      |");
    System.out.println("|                           |");
    System.out.println("-----------------------------");
    System.out.println();
}
```

```
주문하실 메뉴를 선택해주세요 ☞
5
김치추가 1개가 장바구니에 추가됐습니다.
총 금액 : 6000
-----------------------------------------
-----------------------------
|                           |
| 최소주문 금액이 넘어야 결제가 됩니다 |
|       ㅠㅠ 더 추가해주세요 ㅠㅠ      |
|                           |
-----------------------------
```

```
구이삼겹 1개가 장바구니에 추가됐습니다.
총 금액 : 14000
-----------------------------------------
장바구니로가려면 11번을 눌러주세요☞

계속 고르려면 추가 할 메뉴번호를 눌러주세요 ☞
-----------------------------------------
2
새로추가된 메뉴 시킨갯수 : 1
간장구이삼겹 1개가 장바구니에 추가됐습니다.
총 금액 : 29000
-----------------------------------------
장바구니로가려면 11번을 눌러주세요☞

계속 고르려면 추가 할 메뉴번호를 눌러주세요 ☞
-----------------------------------------
3
새로추가된 메뉴 시킨갯수 : 1
대창구이삼겹 1개가 장바구니에 추가됐습니다.
총 금액 : 43000
-----------------------------------------
장바구니로가려면 11번을 눌러주세요☞

계속 고르려면 추가 할 메뉴번호를 눌러주세요 ☞
-----------------------------------------
1
총 시킨 갯수 : 2
구이삼겹 2개가 장바구니에 추가됐습니다.
총 금액 : 57000
-----------------------------------------
장바구니로가려면 11번을 눌러주세요☞
```

1. Basket배열에는 사용자가 고른 메뉴를 담고 sameFood배열에는 수를 입력
2. Cnt를 통해서 처음 등록하는 메뉴는 무조건 basket배열에 입력
3. 그다음부터는 basket에 담긴메뉴면 sameFood배열에서 숫자만 더해줌
4. 새로 등록하는 메뉴면 basket배열에 입력해준다

JEOGIYO
UNTACT CATERING

코드리뷰

```
int tip = rd.nextInt(20)*100+2000;
```

```
void myBasket() {
    //장바구니창 시작
    View.orderList(basket, sameFood, Total, tip);
    infoUser = logUserInfo(UserId, infoUser);
    System.out.println("주문하시려면 1번을 눌러주세요☞ ");
    choice = VO.getOrder();
    if(choice == 0) {
        orderRun();
        myBasket();
    }
    payment();
}
```

```
VO[] logUserInfo (String UserId, VO[] infoUser) {
    List list = dao.logUserInfo(UserId);
    for (int i = 0; i < list.size(); i++) {
        infoUser[i] = (VO) list.get(i);

    }
    return infoUser;
}//logUserInfo를 통한 해당 로그인 유저의 정보 불러오기
```

```
public List logUserInfo(String UserId) {
    List list = new ArrayList();

    try {
        conn();
        // 셀렉트문 입력
        String userSelect = "SELECT user_phonenum, user_ad" +
                            " FROM users" +
                            " WHERE user_id = ?";

        preStmt = con.prepareStatement(userSelect);
        preStmt.setString(1, UserId);
        rs = preStmt.executeQuery();

        while (rs.next()) {

            String UserPhoneNum = rs.getString("user_phonenum");
            String UserAddress = rs.getString("user_ad");

            VO vo = new VO();

            vo.setUserPhoneNum(UserPhoneNum);
            vo.setUserAddress(UserAddress);

            list.add(vo);

        }
        rs.close();
        preStmt.close();
        con.close();

    } catch (SQLException e) {
        e.printStackTrace();
    }
    return list;
}//로그인한 사람의 정보 검색
```

```
public static void orderList(ArrayList<VO> basket, HashMap<Integer, Integer> sameFood, int Total, int tip) {

    System.out.println("[redacted]  저 기 요  [redacted]");
    System.out.println("-----------------------------------------------");
    System.out.println("\t\t" + basket.get(0).getRestName() + "\t\t\t");
    System.out.println("-----------------------------------------------");
    for (int i = 0; i < basket.size(); i++) {
        System.out.println(basket.get(i).getFoodName());
        System.out.println(sameFood.get(i) + " 개");
        System.out.println(basket.get(i).getUnitPrice() * sameFood.get(i));
        System.out.println("-----------------------------------------------");
    }

    System.out.println("-----------------------------------------------");
    System.out.println(" 주문금액 : " + (Total) + "원");
    System.out.println(" 배달팁  : " + tip + "원 ");
    System.out.println("-----------------------------------------------");
    System.out.println("\t" + "1." + (Total + tip) + " 원 주문 하기 🛍 ");
    System.out.println("                0. 뒤로 가기                ");
    System.out.println("[redacted]");
    System.out.println();
} // 장바구니
```

```
[redacted]  저 기 요  [redacted]
-----------------------------------------------
                쫄면주는삼겹본능
-----------------------------------------------
구이삼겹
2 개
28000
-----------------------------------------------
간장구이삼겹
1 개
15000
-----------------------------------------------
대창구이삼겹
1 개
14000
-----------------------------------------------
-----------------------------------------------
주문금액 : 57000원
배달팁  : 3400원
-----------------------------------------------
        1.60400 원 주문 하기 🛍
              0. 뒤로 가기
[redacted]

주문하시려면 1번을 눌러주세요☞
1
```

1. 고른 메뉴와 개수 총 금액, 배달팁을 View단을 통해서 출력
2. 로그인한 유저 아이디를 받아서 DAO에서 유저의 정보를 infoUser에미리 받아옴

코드리뷰 - 결제창

```
void payment() {
    //결제창 시작
    if(choice == 1)
        View.payment(infoUser, Total, tip); //결제창 출력

    String pay[] = {"현금/카드 ", "휴대폰 결제", "까까오페이"};
    System.out.println("결제 방식을 입력해주세요 ☞ ");

    boolean boo = true;
    while(boo) {
        choice = VO.getOrder();
        switch (choice) {
        case 0 :
            myBasket();
            payment();
            break;
        case 1 :
            System.out.println(pay[choice-1] +" 결제를 선택하셨습니다");
            break;
        case 2 :
            System.out.println(pay[choice-1] +" 결제를 선택하셨습니다");
            break;
        case 3 :
            System.out.println(pay[choice-1] +" 결제를 선택하셨습니다");
            break;
        default :
            System.out.println("결제방식을 다시 선택해주세요");
        }
        break;
    }
    paySuccess();
}
```

```
public static void payment(VO[] infoUser, int Total, int tip) {

    System.out.println("[redacted]   저 기 요   [redacted]");
    System.out.println("-----------------------------------------");
    System.out.println("\t\t🏦 배달 정보 🏦");
    System.out.println();
    System.out.println("  주소 : " + infoUser[0].getUserAddress());
    System.out.println();
    System.out.println("  연락처 : " + infoUser[0].getUserPhoneNum());
    System.out.println();
    System.out.println("-----------------------------------------");
    System.out.println("-----------------------------------------");
    System.out.println("\t\t💳 결제수단 💳 ");
    System.out.println("  1. 신용 / 체크카드 ");
    System.out.println("  2. 휴대폰결제 ");
    System.out.println("  3. 까까오페이 ");
    System.out.println("-----------------------------------------");
    System.out.println("-----------------------------------------");
    System.out.println("\t\t💰 결제금액 💰 ");
    System.out.println("  주문금액 \t\t\t" + Total + " 원");
    System.out.println("  배달팁  \t\t\t " + tip + "원 ");
    System.out.println("-----------------------------------------");
    System.out.println("              4. " + (Total + tip) + "원 결제하기 📝 ");
    System.out.println("              0. 뒤로 가기                ");
    System.out.println();
    System.out.println("[redacted]");
    System.out.println();
}
```

```
[redacted]   저 기 요   [redacted]
-----------------------------------------
          🏦 배달 정보 🏦

주소 : 서울시 마포구 신수동 63-14 거구장 지하1층 5호실 4번째 줄

연락처 : 010-2331-2830

-----------------------------------------
-----------------------------------------
          💳 결제수단 💳
  1. 신용 / 체크카드
  2. 휴대폰결제
  3. 까까오페이
-----------------------------------------
-----------------------------------------
          💰 결제금액 💰
주문금액                        57000 원
배달팁                          3400원
-----------------------------------------
              4. 60400원 결제하기 📝
              0. 뒤로 가기

[redacted]

결제 방식을 입력해주세요 ☞
1
현금/카드  결제를 선택하셨습니다
결제하시려면 4번을 눌러주세요
4
```

1. 받아온 유저정보와 총금액 배달팁을 View단에 출력
2. 스위치문을 통해서 결제 방법을 선택

JEOGIYO
UNTACT CATERING

코드리뷰 결제 정보넘기기

```
void paySuccess() {

    System.out.println("결제하시려면 4번을 눌러주세요");
    choice = VO.getOrder();
    switch(choice) {
    case 0 :
        myBasket();
        payment();
    case 4 :
        View.payment2(); //결제를 시작합니다 출력
        System.out.println();
        View.payment3(); //결제중입니다 출력
        //주문 데이터 DB로 넘기기
        Total += tip;
        int RestRegiNum = basket.get(0).getRestRegiNum();
        OrderNum = dao.lastOrderNum()+1;
        dao.addOrder(OrderNum, UserId, RestRegiNum, Total);
        dao.addOrderDetail(basket, OrderNum);
        System.out.println("결제완료");
        break;
    }

    end();
}
```

```
public int lastOrderNum() {
    //List list = new ArrayList();
    int OrderNum = 0;
    try {
        conn();
        // 셀렉트문 입력
        String userSelect = "SELECT MAX(order_num)" +
                            " FROM orders";


        preStmt = con.prepareStatement(userSelect);
        //preStmt.setString(1, "orders");
        rs = preStmt.executeQuery();

        while (rs.next()) {

            OrderNum = rs.getInt("MAX(order_num)");
            VO vo = new VO();
            vo.setOrderNum(OrderNum);
            System.out.println();
            //list.add(vo);

        }
        rs.close();
        preStmt.close();
        con.close();

    } catch (SQLException e) {
        e.printStackTrace();
    }
    //return list;
    return OrderNum;
}//로그인한 사람의 정보 검색
```

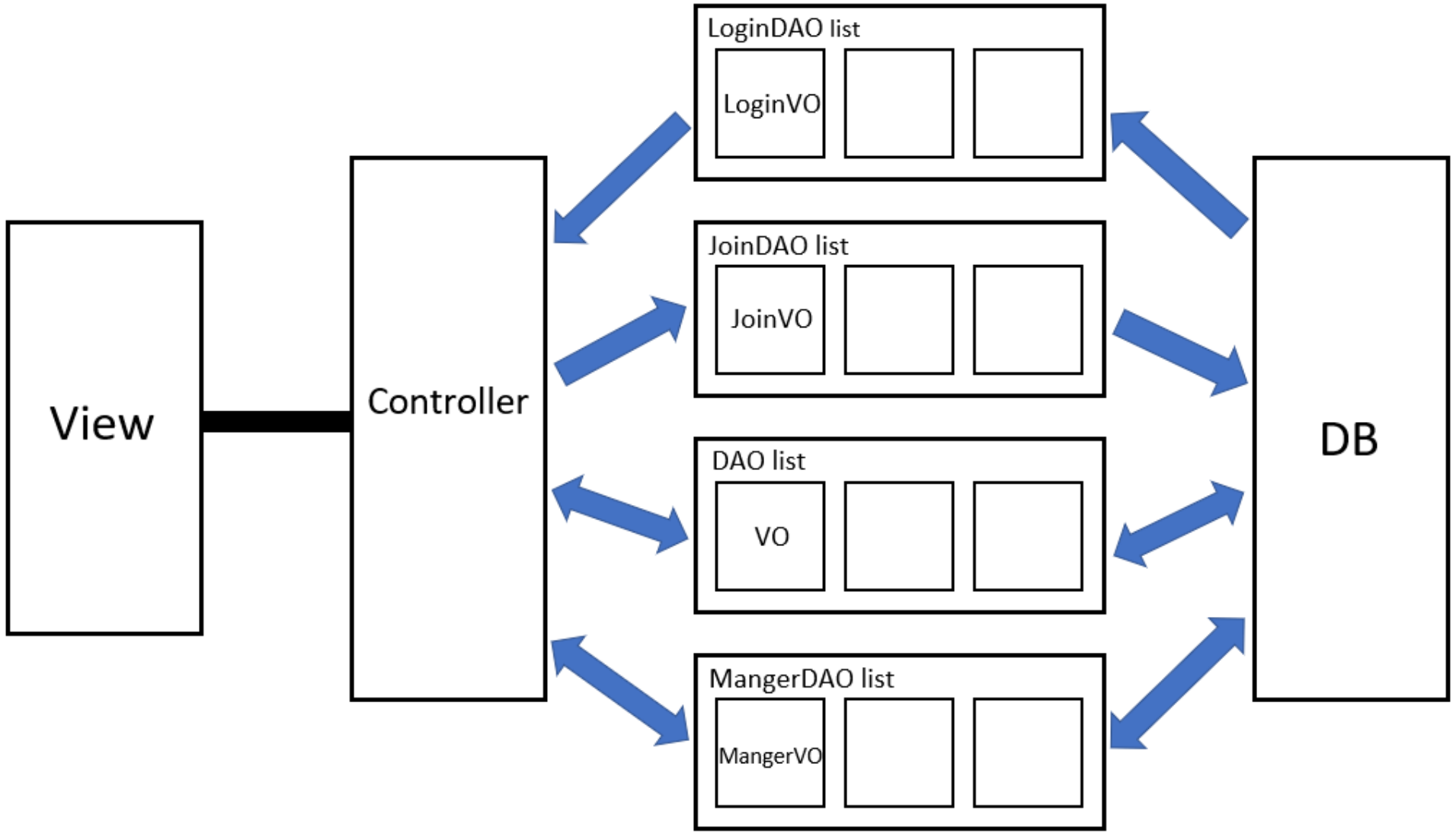

```
public void addOrderDetail(ArrayList<VO> basket, int LastOrder) {
    try {
        for (int i = 0; i < basket.size(); i++) {
        conn();
        String userInsert = "INSERT INTO ordersdetail(order_num, food_num, order_food) VALUES(?,?,?)";
        preStmt = con.prepareStatement(userInsert); //pre는 statement에 쿼리문을 넣는다.


            preStmt.setInt(1, LastOrder );
            preStmt.setInt(2, basket.get(i).getFoodNum());
            preStmt.setString(3, basket.get(i).getFoodName());


        preStmt.executeUpdate(); //insert 와 update는 이걸로 그냥은 여기에 쿼리문을 넣는다.
        preStmt.close();
        con.close();
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }

} //주문정보 데베로넘기기
```

```
public void addOrder(int LastOrder, String UserId, int RestRegiNum,  int Total ) {
    try {
        //for (int i = 0; i < basket.size(); i++) {
        conn();
        String userInsert = "INSERT INTO orders(order_num, user_id, rest_reginum, order_totalamount) VALUES(?,?,?,?)";
        preStmt = con.prepareStatement(userInsert); //pre는 statement에 쿼리문을 넣는다.


            preStmt.setInt(1, LastOrder);
            preStmt.setString(2, UserId);
            preStmt.setInt(3, RestRegiNum);
            preStmt.setInt(4, Total);

        preStmt.executeUpdate(); //insert 와 update는 이걸로 그냥은 여기에 쿼리문을 넣는다.
        preStmt.close();
        con.close();
    //  }
    } catch (SQLException e) {
        e.printStackTrace();
    }
} //orders 테이블에 추가
```

1. DAO를 통해서 마지막 주문번호를 받아온다음 1을 더해서 새로운 주문의 주문번호로 사용
2. DAO를 사용해 주문정보(주문번호, 사용자아이디, 식당번호, 총금액)와 상세주문번호(주문번호, 음식번호, 음식명)를 DB로 넘겨줌

JEOGIYO
UNTACT CATERING

어플리케이션 구조


LoginDAO list
LoginVO
JoinDAO list
JoinVO
DAO list
VO
MangerDAO list
MangerVO
View
Controller
DB

코드리뷰 (관리자 창)
: 모든 회원조회

```java
ManagerDAO dao = new ManagerDAO();
List list = dao.listMember();
String name = "";

System.out.print("모든회원 or 특정회원> ");
String Allone = sc.next(); // 아이디나 비밀번호
if (Allone.equals("모든회원")) {
    for (int i = 0; i < list.size(); i++) {
        ManagerVO membervo = (ManagerVO) list.get(i);
        System.out.println("--------------------[" + (i + 1) + "]----------
        System.out.println("회원 이름: " + membervo.getUserName());
        System.out.println("회원 아이디:" + membervo.getUserId());
        System.out.println("회원 비밀번호: " + membervo.getUserPw());
        System.out.println("회원 주민번호: " + membervo.getUserIdentication());
        System.out.println("회원 전화번호: " + membervo.getUserPhoneNum());
        System.out.println("회원 주소: " + membervo.getUserAd());
        System.out.println("----------------------------------------");
```

```java
public class ManagerVO {
    //유저테이블의 정보
    String UserId;
    String UserPw;
    String UserName;
    String UserIdentication;
    String UserPhoneNum;
    String UserAd;


    public String getUserId() {
        return UserId;
    }
    public void setUserId(String userId) {
        UserId = userId;
    }
    public String getUserPw() {
        return UserPw;
    }
    public void setUserPw(String userPw) {
        UserPw = userPw;
    }
```

```java
//1.조회(회원정보)
public List listMember() {
    List list = new ArrayList();
    try {
        conn();
        String query="SELECT * FROM users";
        preStmt = con.prepareStatement(query);
        rs = preStmt.executeQuery();
        while(rs.next()) {
            String UserId = rs.getString(1);
            String UserPw = rs.getString(2);
            String UserName = rs.getString(3);
            String UserIdentication = rs.getString(4);
            String UserPhoneNum = rs.getString(5);
            String UserAd = rs.getString(6);
            ManagerVO mangersvo = new ManagerVO();
            mangersvo.setUserId(UserId);
            mangersvo.setUserPw(UserPw);
            mangersvo.setUserName(UserName);
            mangersvo.setUserIdentication(UserIdentication);
            mangersvo.setUserPhoneNum(UserPhoneNum);
            mangersvo.setUserAd(UserAd);
            list.add(mangersvo);
```

DB

Return list;

코드리뷰 (관리자 창)
:특정회원 조회

```
if (Allone.equals("특정회원")) {
    System.out.println("어떤 회원을 조회하시겠습니까? "); // 아이디 비번
    String inputName = sc.next();

    for (int i = 0; i < list.size(); i++) {
        ManagerVO membervo = (ManagerVO) list.get(i);
        if (membervo.getUserName().equals(inputName)) {
            System.out.println("==================회원<" + (i + 1) + ">============
            System.out.println("회원 이름: " + membervo.getUserName());
            System.out.println("회원 아이디:" + membervo.getUserId());
            System.out.println("회원 비밀번호: " + membervo.getUserPw());
            System.out.println("회원 주민번호: " + membervo.getUserIdentication());
            System.out.println("회원 전화번호: " + membervo.getUserPhoneNum());
            System.out.println("회원 주소: " + membervo.getUserAd());
         name = inputName;
        }
    }
```

```
List orderslist = dao.listOrders(name);
for (int i = 0; i < orderslist.size(); i++) {
    ManagerVO ordersvo = (ManagerVO) orderslist.get(i);
    System.out.println("-----------------주문 정보<" + (i + 1) + ">--------------
    System.out.println("주문 번호: " + ordersvo.getORDER_NUM());
    System.out.println("주문 식당 사업자등록번호: " + ordersvo.getREST_REGINUM());
    System.out.println("주문 식당이름: " + ordersvo.getREST_NAME());
    System.out.println("주문 음식이름: " + ordersvo.getORDER_FOOD());
    System.out.println("주문 총 금액: " + ordersvo.getORDER_TOTALAMOUNT());
    System.out.println("----------------------------------------");
}
System.out.println("========================================");
System.out.println("[redacted]");

if(name.equals("")) {
    System.out.println("등록된 회원이 아닙니다.");
}
```

```
String query="SELECT order_num, rest_reginum, rest_name, order_food, order_totalamount\r\n" +
        "FROM restaurant \r\n" +
        "JOIN orders USING(rest_reginum)\r\n" +
        "JOIN ordersdetail USING(order_num)\r\n" +
        "WHERE order_num IN (SELECT order_num "
        + "FROM users JOIN orders USING(user_id) WHERE user_name = ?)";
preStmt = con.prepareStatement(query);
preStmt.setString(1, name);
```

JEOGIYO
UNTACT CATERING

코드리뷰(관리자 창)
:회원정보 수정

```java
//5-3.회원정보 수정
public void updateOneUser() {
    System.out.print("정보를 수정할 아이디 입력: ");
    String id = sc.next();
    ManagerDAO dao = new ManagerDAO();

    List list = dao.listMember();
    for (int i = 0; i < list.size(); i++) {
        ManagerVO membervo = (ManagerVO) list.get(i);
        if (membervo.getUserId().equals(id)) {
            System.out.print("회원이름: ");
            String UserName = sc.next();
            System.out.print("회원비밀번호: ");
            String UserPw = sc.next();
            System.out.print("회원 전화번호: ");
            String UserPhoneNum = sc.next();
            System.out.print("회원 주민번호: ");
            String UserIdentication = sc.next();
            System.out.print("회원 주소: ");
            String UserAdd = sc.next();

            ManagerVO vo = new ManagerVO();
            vo.setUserName(UserName);
            vo.setUserId(id);
            vo.setUserPw(UserPw);
            vo.setUserPhoneNum(UserPhoneNum);
            vo.setUserIdentication(UserIdentication);
            vo.setUserAd(UserAdd);
            dao.updateMember(vo);
```

```java
// 3. 회원정보 수정(ok)
public void updateMember(ManagerVO vo) {
    try {
        conn();
        String query="UPDATE users SET user_id=?, user_pw=?, "
                + "user_name=?, user_identication=? ,"
                + "user_phonenum=?, user_ad=? WHERE user_id=?";
        preStmt = con.prepareStatement(query);
        preStmt.setString(1, vo.getUserId());
        preStmt.setString(2, vo.getUserPw());
        preStmt.setString(3, vo.getUserName());
        preStmt.setString(4, vo.getUserIdentication());
        preStmt.setString(5, vo.getUserPhoneNum());
        preStmt.setString(6, vo.getUserAd());
        preStmt.setString(7, vo.getUserId());
        int cnt = preStmt.executeUpdate();
        System.out.println("<정보 수정이 완료되었습니다.>");
        rs.close();
        preStmt.close();
        con.close();
```

코드리뷰(관리자 창)
:회원 별 구매액 순위

```
ManagerDAO dao = new ManagerDAO();
List list = dao.listUserAmountRank();

System.out.println("            회원 아이디    |      누적 주문액        ");
System.out.println("-----------------------------------------");
for (int i = 0; i < list.size(); i++) {
    ManagerVO ordersvo = (ManagerVO) list.get(i);
    System.out.println("        " + ordersvo.getSUM_RANK() + "위: " + ordersvo.getUSER_ID() + " | "
            + ordersvo.getSUM_TOTALAMOUNT() + "원");
}
```

```
String query="SELECT user_id, sum(order_totalamount),"
                + "RANK() OVER (ORDER BY sum(order_totalamount)DESC) 구매순위 "
                + "FROM orders "
                + "GROUP BY user_id "
                + "ORDER BY 2 DESC";
```

코드리뷰(관리자 창)
:식당 별 주문금액 순위

```
ManagerDAO dao = new ManagerDAO();
List list = dao.listRestAmount();

System.out.println("     순위 | 사업자등록번호 | 식당이름 | 누적 주문 금액     ");
System.out.println("----------------------------------------");
for (int i = 0; i < list.size(); i++) {
    ManagerVO ordersvo = (ManagerVO) list.get(i);
    System.out.println("   " + ordersvo.getSUM_RANK() + "위| "
            + ordersvo.getREST_REGINUM() + " | "
            + ordersvo.getREST_NAME() + " | " + ordersvo.getSUM_TOTALAMOUNT() + "원");
}
```

```
String query="SELECT rest_reginum, rest_name, SUM(order_totalamount), "
        + "RANK() OVER(ORDER BY SUM(order_totalamount) DESC) \"총 주문액 순위\"\r\n" +
        "FROM restaurant\r\n" +
        "JOIN orders USING(rest_reginum)\r\n" +
        "GROUP BY rest_reginum, rest_name";
preStmt = con.prepareStatement(query);
```

코드리뷰(관리자 창)
:음식 별 주문건수

```
ManagerDAO dao = new ManagerDAO();
List list = dao.FoodCountRank();

System.out.println(" 순위 | 사업자등록번호 | 식당이름 | 주문된 음식 | 누적 주문건 ");
System.out.println("-----------------------------------------");
for (int i = 0; i < list.size(); i++) {
    ManagerVO ordersvo = (ManagerVO) list.get(i);
    System.out.println(" " + ordersvo.getCOUNT_RANK() + "위 |    "
            + ordersvo.getREST_REGINUM() + "    | "
            + ordersvo.getREST_NAME() + " | " + ordersvo.getORDER_FOOD()
            + " | " + ordersvo.getCOUNT_FOOD()
            + "건    ");
}
```

```
String query="SELECT rest_reginum, rest_name, order_food, count(order_num),"
        + "RANK() OVER (ORDER BY count(order_food) DESC) "
        + "FROM restaurant "
        + "JOIN unitRestMenu USING(rest_reginum) "
        + "JOIN ordersdetail USING(food_num) "
        + "GROUP BY rest_reginum, rest_name, order_food";
preStmt = con.prepareStatement(query);
```

코드리뷰(관리자 창)
:음식 별 주문금액 순위

```
ManagerDAO dao = new ManagerDAO();
List list = dao.FoodAmountRank();

// 수정하기
System.out.println(" 순위 | 사업자등록번호 | 식당이름 | 주문된 음식 | 누적 주문액 ");
System.out.println("-----------------------------------------");
for (int i = 0; i < list.size(); i++) {
    ManagerVO ordersvo = (ManagerVO) list.get(i);
    System.out.println(" " + ordersvo.getSUM_RANK() + "위 |    "
            + ordersvo.getREST_REGINUM() + "    | "
            + ordersvo.getREST_NAME() + " | " + ordersvo.getORDER_FOOD() + " | "
            + ordersvo.getSUM_TOTALAMOUNT()
            + "원");
}
```

```
String query = "SELECT rest_reginum, rest_name, order_food, SUM(unitPrice), "
        + "RANK() OVER(ORDER BY SUM(unitPrice) DESC) "
        +"FROM restaurant "
        +"JOIN unitRestMenu USING(rest_reginum) "
        +"JOIN ordersdetail USING(food_num) "
        +"GROUP BY rest_reginum, rest_name,order_food";
```

코드리뷰
:맛집 추천

```
// 5-8.맛집추천
public void recommendFood() {
    ManagerDAO dao = new ManagerDAO();
    List list = dao.recommendFood();

    for (int i = 0; i < 3; i++) {
        ManagerVO ordersvo = (ManagerVO) list.get(i);
        System.out.println("<인기 음식 추천!!> " + ordersvo.getREST_NAME()
                                + "의 " + ordersvo.getORDER_FOOD());
    }
}
```

```
String query = "SELECT rest_name, order_food, COUNT(food_num)\r\n" +
        "FROM ordersDetail\r\n" +
        "JOIN unitrestmenu USING(food_num)\r\n" +
        "JOIN restaurant USING(rest_reginum)\r\n" +
        "GROUP BY rest_name, order_food\r\n" +
        "ORDER BY 3 DESC";
preStmt= con.prepareStatement(query);
```

감사합니다 고객님,
혹시 질문 있으실까요 ?





프로젝트를 마치며...

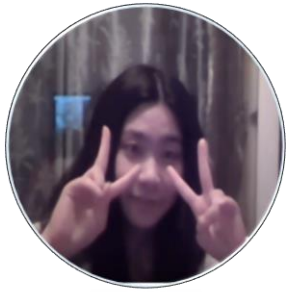
기간 : 2020.12.17.
~ 2020.12.27.

매일 8시~11시까지는 고정
회의,
그래. 우린 회의에 미친 조...
크리스마스? **예외는 없다...**

크리스마스 이브도 물론...
2020 12 25 00:00에
메리크리스마스를 외쳤고...

주말? 당연...

JEOGIYO
UNTACT CATERING



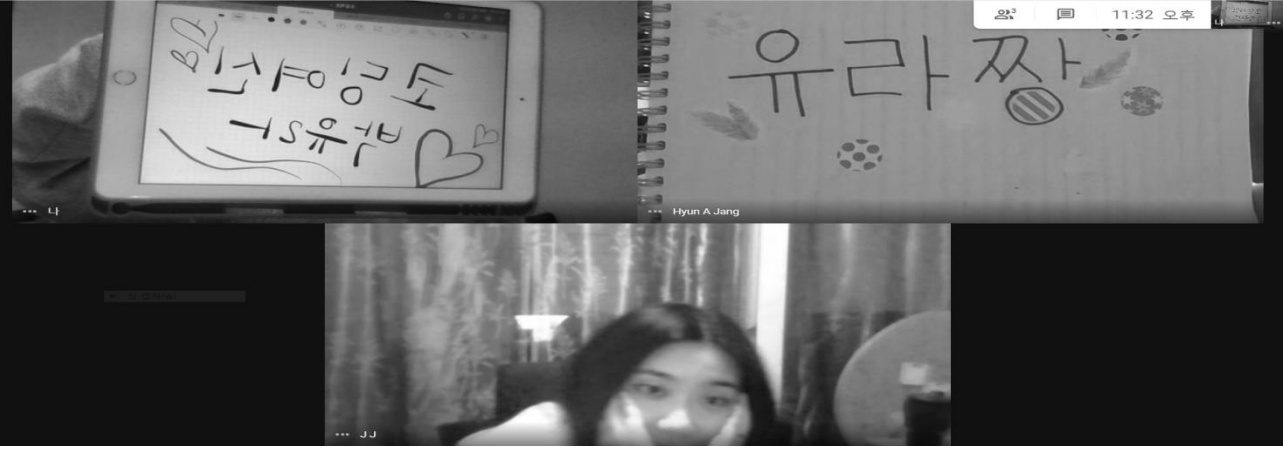
박은지 : 처음 제대로 된 기획을 하면서 프로젝트를 했는데, 마음이 잘 맞는 팀원들과 함께해서 더 인상깊었던 프로젝트였습니다. 실력이 많이 부족하다고 느꼈는데 다같이 앞에서 끌어주고 뒤에서 밀어주며 서로 도움을 많이 주고 받은 거 같아서 감사했습니다.
그리고 생각했던 기능을 구현했다는 것에 대해서 성취감을 느낄 수 있던 좋은 기회였습니다!

손예인 : 이번 프로젝트는 지난 기간동안 배웠던 자바와 데이터베이스로 프로그램을 구현할 수 있었던 좋은 기회였고, 하나부터 열까지 배우지 않은 부분이 없었습니다! 그리고 바랬던 기능구현들을 위해서는 더 열심히 공부해야겠다는 다짐을 크게 할 수 있었습니다.
그리고 무엇보다 끝까지 마무리할 수 있도록 최선을 다해 함께해주신 팀원분들께 진심으로 감사드립니다!

임상우 : erd와 테이블을 짜는 것이 생각보다 정말 어렵고 오래걸려서 처음에 구현하고자 했던 기능들을 많이 구현해보지 못해서 아쉬웠지만 팀원들과 서로 많은 의견도 주고받고 아는 것과 모르는 것도 교환하면서 더 발전할 수 있는 시간이 되었던 것 같습니다.

장현아 : 프로젝트 진행하면서 좋았던 점은 더디지만 조금씩 나아가는 제 자신을 발견했던 것이고 진행하는 내내 팀원들끼리 서로 다독이며 프로젝트를 진행하는 모습에 이 친구들과 같은 팀이 되어 다행이라는 생각을 했습니다. 서로의 실력과 생각들이 다르지만 서로 맞춰가려고 노력하는 모습이 프로젝트를 진행하는 큰 이유 중 하나일 것이라고 생각을 했습니다. 프로젝트 기간 동안 정말 감사했습니다!

JEOGIYO
UNTACT CATERING



감사합니다!

Thank you!

JEOGIYO
UNTACT CATERING