

냉장고 재고 관리 프로그램

프로그램의 목적

- 냉장고의 투명화
- 식재료의 선호도 관리
- 간단한 레시피 검색

주요 기능

- 사용자 가입 / 정보수정 / 탈퇴
- 사용자의 냉장고 정보 생성 / 삭제 / 기존 설정된 냉장고와의 연결
- 사용자가 선택한 냉장고 내부의 아이템 관리 – 등록 / 검색 / 수정 / 삭제
- 레시피 검색

사용자 관리부

<사용자에 대한 모든 관련 클래스들>

<포함 코드>

* user.view

- JFrameConstruct.java
- MembershipInsertDialog.java
- MemberJoiningEvent.java
- MemberLogin.java

* user.model.service

- UserDao.java

* user.controller

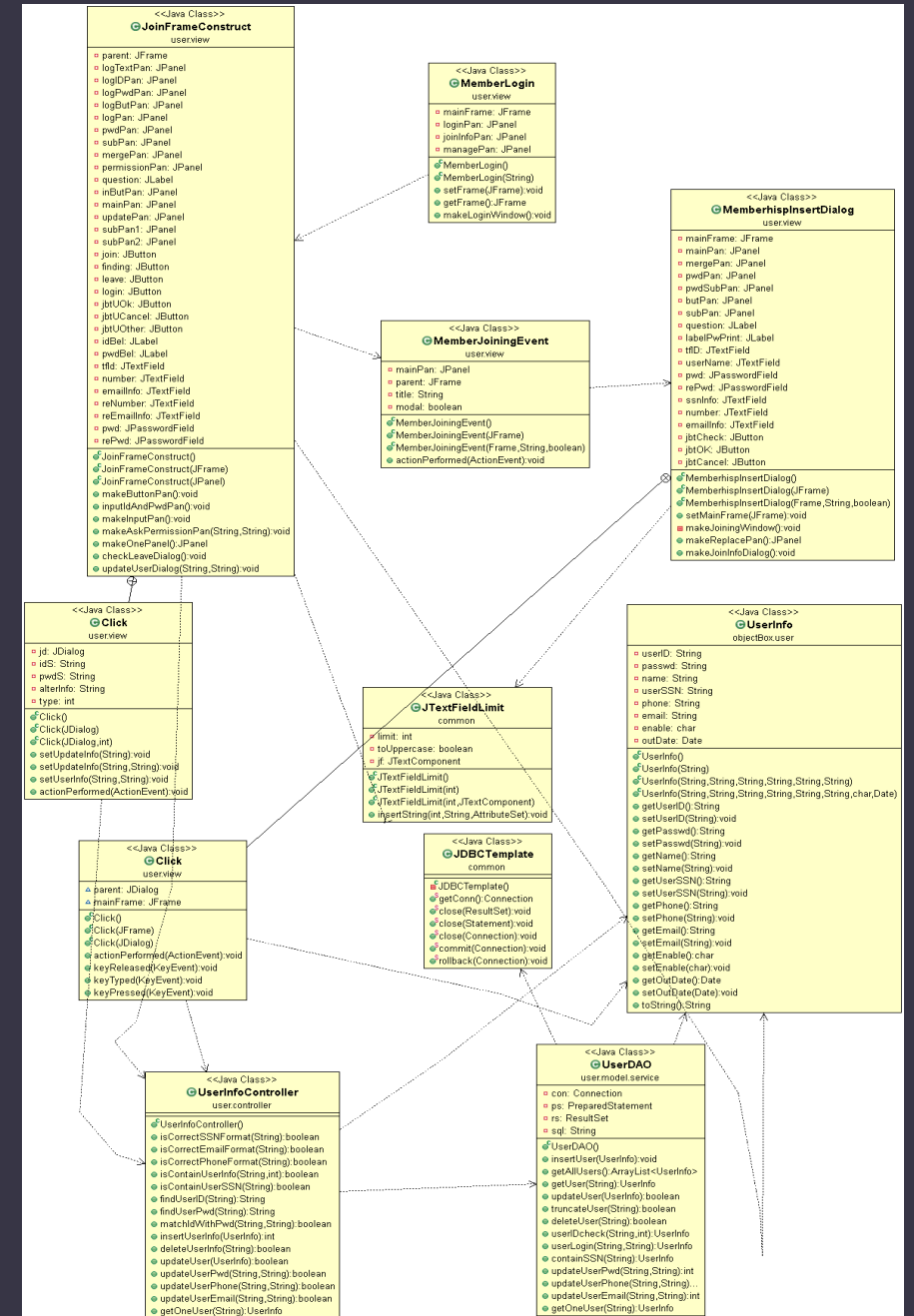
- UserInfoController.java

* objectBox.user

- UserInfo.java

* common

- JTextFieldLimit.java
- JDBCTemplate.java



사용자 정보 관리

· 로그인 화면

<아이디와 암호>

- 사용자에게 아이디와 암호를 입력 받기 위한 부분

The image shows a login form with the following elements:

- Input field for ID (아이디):
- Input field for Password (비밀번호):
- Login button (로그인):
- Sign Up button (회원가입):
- Find ID/Password button (아이디/암호찾기):
- Logout button (회원탈퇴):

<로그인>

- 아이디가 입력된 내용과,, 비밀번호 입력된 내용의 정보가 사용자 정보가 저장되어있는 DB와 매치 확인

<회원가입>

- 회원을 유일하게 구별 가능한 아이디와 주민번호를 입력받도록 구성
- 필수 입력 항목과 선택 입력 항목으로 구성
- 구별 가능한 식별자로 사용자 알려주기 (아이디 이외의 구별가능 식별자로 주민번호 이용)

<회원탈퇴>

- 회원의 정보 삭제
- ||서 삭제 트리거 구성 필요

회원가입

· 회원가입 화면

회원가입

(* 는 필수입력항목입니다)

* 사용하실 아이디를 입력해주세요

중복확인

* 사용하실 비밀번호를 입력하세요

* 사용하실 비밀번호를 다시 입력하세요

*당신의 이름을 입력하세요

*당신의 주민번호를 "-"를 포함하여 입력하세요

당신의 핸드폰 번호를 "-"를 포함하여 입력하세요

이메일을 입력하세요

가입완료 가입취소

BLACK.USERINFO

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 USER_ID	VARCHAR2 (15 BYTE)	No	(null)	1	사용자 아이디
2 USER_PWD	VARCHAR2 (15 BYTE)	No	(null)	2	사용자 암호
3 USER_NAME	VARCHAR2 (21 BYTE)	No	(null)	3	사용자 이름
4 USER_SSN	VARCHAR2 (14 BYTE)	No	(null)	4	사용자 주민번호
5 PHONE	VARCHAR2 (15 BYTE)	Yes	(null)	5	핸드폰번호
6 EMAIL	VARCHAR2 (30 BYTE)	Yes	(null)	6	이메일
7 ENABLE	CHAR (1 BYTE)	Yes	'Y'	7	사용자 탈퇴여부
8 OUTDATE	DATE	Yes	(null)	8	사용자 탈퇴날짜

PK_USERINFO (USER_ID)

UNI_USERINFO_SSN (USER_SSN)

< DB 구성 >

< DB 제약조건 >

CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1 NN_USERINFO_NAME	Check	"USER_NAME" IS NOT NULL
2 NN_USERINFO_PWD	Check	"USER_PWD" IS NOT NULL
3 NN_USERINFO_SSN	Check	"USER_SSN" IS NOT NULL
4 PK_USERINFO	Primary Key	(null)
5 UNI_USERINFO_SSN	Unique	(null)

Unique
(아이디/암호 찾기)

Java와 DB와
동일하게 null 허용

항목의 유효성 확인

· 유효성 검사



- 필수 입력 항목은 비어있지 않도록
- 비밀번호의 더블 확인
- 아이디와 주민등록번호의 중복
- 이름 길이
- 주민등록번호 형식 } <주민등록번호는 숫자>
- 핸드폰 번호 형식
- 이메일 입력 형식

*당신의 주민번호를 "-"를 포함하여 입력하세요

*사용하실 비밀번호를 입력하세요

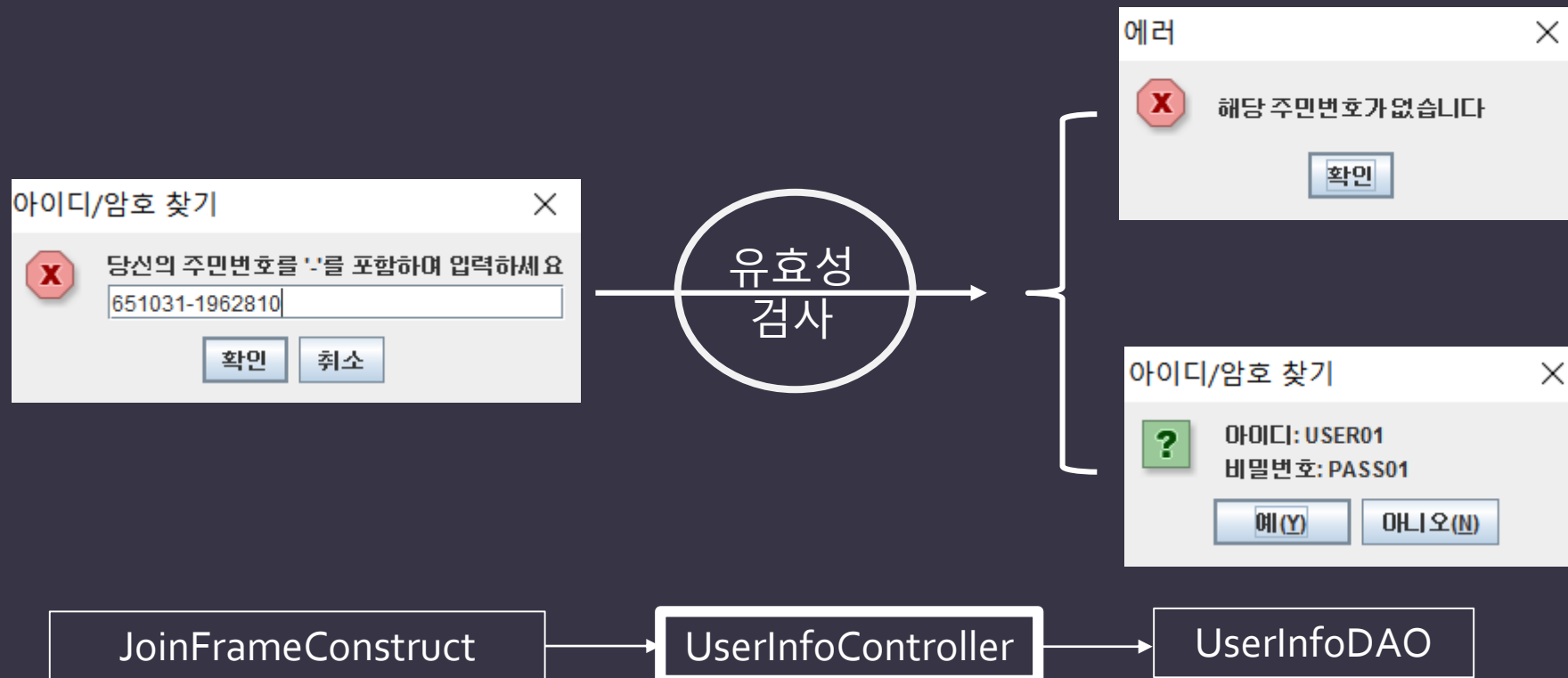
*사용하실 비밀번호를 다시 입력하세요

비밀번호가 일치하지 않습니다.

확인

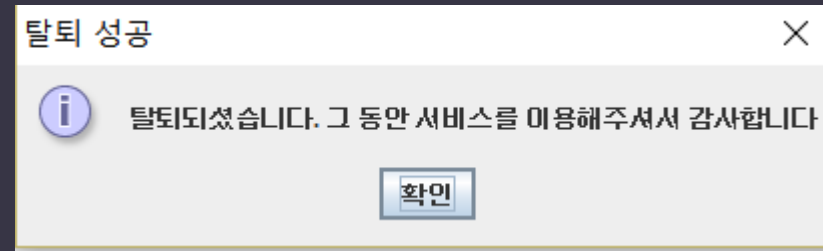
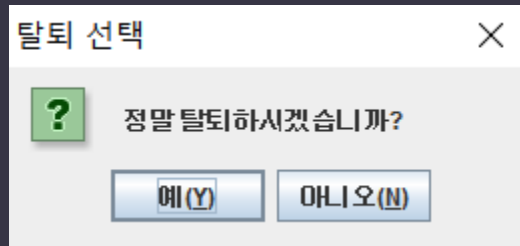
아이디/암호 찾기

- DB의 UNIQUE 항목을 이용하여 아이디/암호 찾기
 - DB에서 정의한 Unique 항목: USER_ID & USER_SSN



회원 탈퇴

- 회원 탈퇴 - 3개월의 유예 기간을 주기 위해 바로 DB에서 삭제하지 않음



	USER_ID	USER_PWD	USER_NAME	USER_SSN	PHONE	EMAIL	ENABLE	OUTDATE
	10	USER11	PASS11	김유진	860112-2410019	(null)	(null)	(null)
	10	USER11	PASS11	김유진	860112-2410019	(null)	N	16/01/28

아이디/암호 입력

· 아이디/암호 입력 필드

```
public void inputIdAndPwdPan() {
    logTextPan = new JPanel(new GridLayout(2,2));

    idBel = new JLabel("아이디: ");
    tfId = new JTextField(15);
    tfId.setDocument(new JTextFieldLimit(15, tfId));

    logIDPan = new JPanel();
    logIDPan.setLayout(new FlowLayout());
    logIDPan.add(idBel);
    logIDPan.add(tfId);
    logTextPan.add(logIDPan);

    pwdBel = new JLabel("비밀번호: ");
    pwd = new JPasswordField(15);
    pwd.setDocument(new JTextFieldLimit(15, pwd))
    pwd.setEchoChar('*');

    logPwdPan = new JPanel();
    logPwdPan.setLayout(new FlowLayout(FlowLayout
    logPwdPan.add(pwdBel);
    logPwdPan.add(pwd);
    logTextPan.add(logPwdPan);
}
```

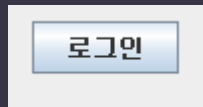
* 입력값 글자 수 제한
- JTextField의 크기가 곧 입력 받는 글자 수가 되도록 제한
(byte 크기를 통해 제한하는 방법 이용)

```
@Override
public void insertString(int offset, String str, AttributeSet attr)
    throws BadLocationException {
    if(str == null){
        return;
    }

    if(jf.getText().getBytes().length + str.getBytes().length <= limit){
        super.insertString(offset, str, attr);
    }
}
```

아이디/암호 매치 확인

- 아이디/암호 매치를 USERINFO DB에서 확인
-> 결과가 있으면 YES / 없으면 NO



```
//아이디와 암호가 매치되는지 확인
public boolean matchIdWithPwd(String userID, String pwd){
    boolean flag = false;

    UserInfo user = new UserDAO().userLogin(userID, pwd);
    if(user != null){
        flag = true;
    }

    return flag;
}
```

```
//아이디와 암호가 저장되어 있는가 확인: 저장되어 있으면 저장된 user 정보 리턴
public UserInfo userLogin(String userID, String userPwd){
    UserInfo user = null;
    try {
        con = getConn();
        sql = "SELECT * FROM USERINFO "
            + "WHERE USER_ID = ? AND USER_PWD = ?";

        ps = con.prepareStatement(sql);
        ps.setString(1, userID);
        ps.setString(2, userPwd);
        rs = ps.executeQuery();

        if(rs.next()){
            user = new UserInfo();

            user.setUserID(rs.getString("USER_ID"));
            user.setPasswd(rs.getString("USER_PWD"));
            user.setName(rs.getString("USER_NAME"));
            user.setUserSSN(rs.getString("USER_SSN"));
            user.setPhone(rs.getString("PHONE"));
            user.setEmail(rs.getString("EMAIL"));
            user.setEnable(rs.getString("ENABLE").charAt(0));
            user.setOutDate(rs.getDate("OUTDATE"));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        close(rs);
        close(ps);
        close(con);
    }

    return user;
}
```

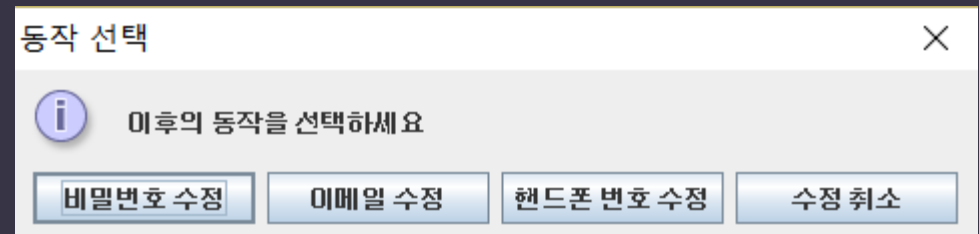
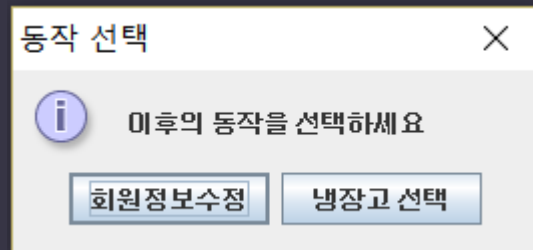
VIEW

CONTROLLER

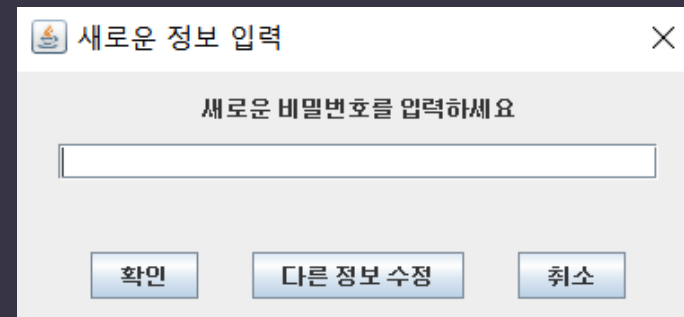
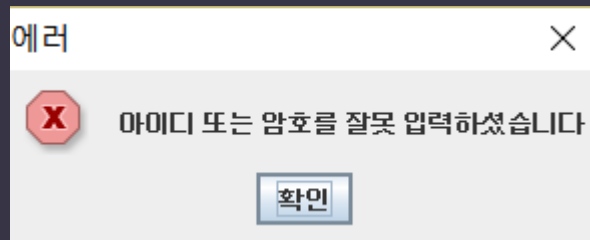
MODEL

로그인 성공 시 이후 동작 선택

- 회원 수정 항목을 따로 로그인 화면에 구성하지 않음
 - 아이디/암호 매치 확인이 YES일 경우



- 아이디/암호 매치 확인이 NO인 경우



냉장고 관리부

<냉장고에 대한 모든 관련 클래스들>

< 포함 코드 >

* fridge.view

- ActionSelectDialog.java
- FridgeSelectDialog.java
- StorageDialog.java

* fridge.model

- AskPermissionDAO.java
- MatchFridgeDAO.java

* fridge.control

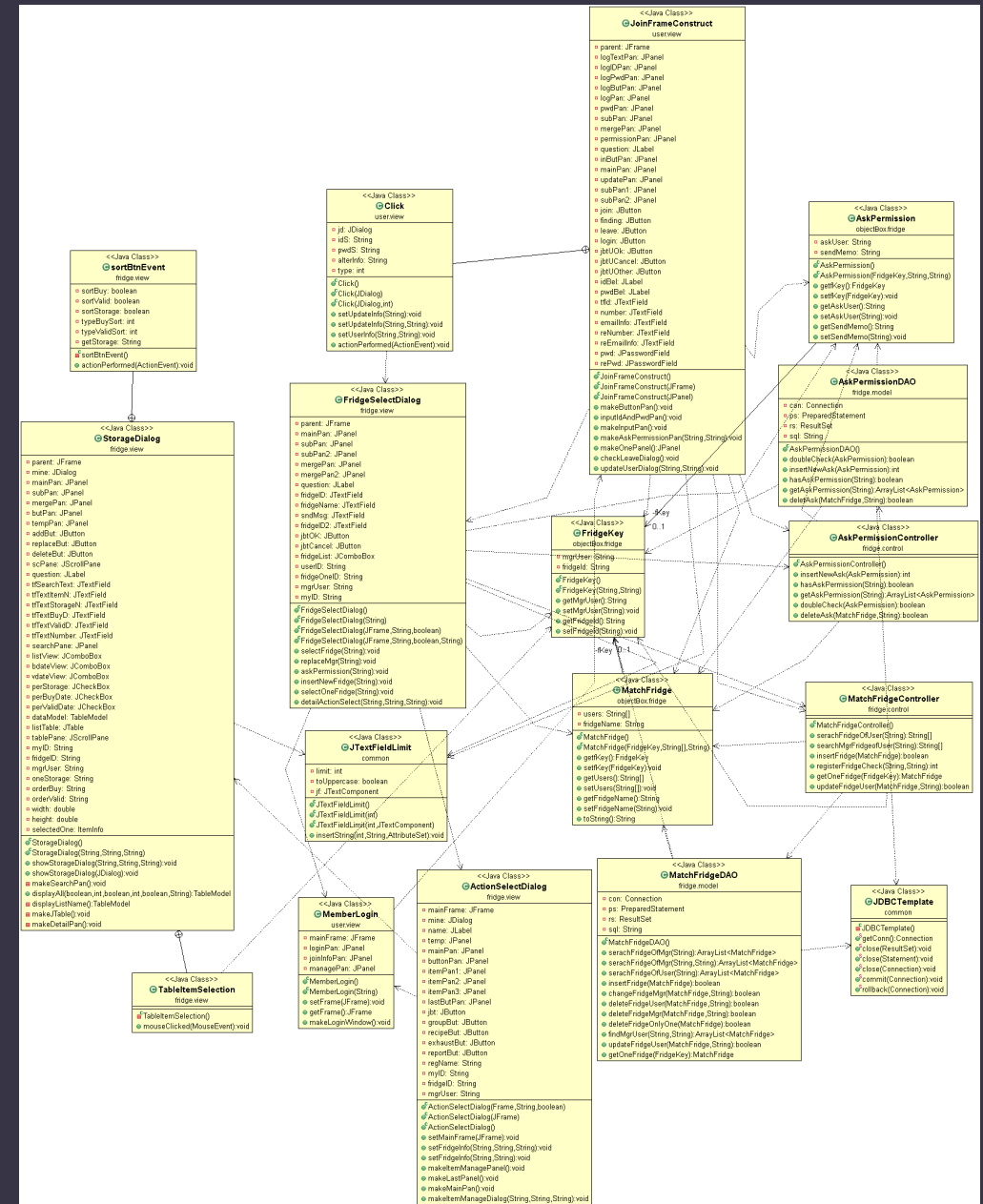
- AskPermissionController.java
- MatchFridgeController.java

* objectBox.fridge

- AskPermission.java
- FridgeKey.java
- MatchFridge.java

* common

- JTextFieldLimit.java
- JDBCTemplate.java



로그인 이후의 냉장고 선택

- 사용자와 냉장고의 연결에 대한 DB

< DB 구성 >

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
USER_ID	VARCHAR2(100 BYTE)	No	(null)	1	냉장고 사용자 아이디
FRIDGE_ID	VARCHAR2(15 BYTE)	No	(null)	2	냉장고 아이디
FRIDGE_NAME	VARCHAR2(20 BYTE)	Yes	(null)	3	냉장고 이름
MGR_USER	VARCHAR2(15 BYTE)	No	(null)	4	냉장고 관리자 아이디

BLACK.MATCH_FRIDGE	
* USER_ID	VARCHAR2(100)
P * FRIDGE_ID	VARCHAR2(15)
FRIDGE_NAME	VARCHAR2(20)
PF * MGR_USER	VARCHAR2(15)
PK_MATCHFRIDGE (MGR_USER, FRIDGE_ID)	
FK_MATCHFRIDGE_USERINFO (MGR_USER)	
PK_MATCHFRIDGE (MGR_USER, FRIDGE_ID)	

USER_ID	FRIDGE_ID	FRIDGE_NAME	MGR_USER
USER01:USER02:	F01	HOME	USER01
USER07:USER04:	F01	WORK	USER07

< DB 제약조건 >

CONSTRAINT_NAME	CONSTRAINT_TYPE
FK_MATCHFRIDGE_USERINFO	Foreign_Key
PK_MATCHFRIDGE	Primary_Key
SYS_C007322	Check

2개의 조합된 COLUMN을
Primary Key로 지정

COLUMN_NAME	COLUMN_POSITION
MGR_USER	1
FRIDGE_ID	2

2개의 열을 KEY로 사용

- OBJECT 자체를 Key와 Value로 나누어서 구현
 - Key: 냉장고 관리자 ID 와 냉장고 ID의 2개의 String 값을 가진 클래스
 - Value: Key를 포함한, DB의 모든 열에 대응되는 값을 가지는 클래스

```
public class FridgeKey{
    private String mgrUser;
    private String fridgeId;

    public FridgeKey() { }
    public FridgeKey(String mgrUser, String fridgeId){
        this.mgrUser = mgrUser;
        this.fridgeId = fridgeId;
    }
    public String getMgrUser() {
        return mgrUser;
    }
    public void setMgrUser(String mgrUser) {
        this.mgrUser = mgrUser;
    }
    public String getFridgeId() {
        return fridgeId;
    }
    public void setFridgeId(String fridgeId) {
        this.fridgeId = fridgeId;
    }
}
```

<Key>

```
public class MatchFridge {
    private FridgeKey fKey;
    private String[] users;
    private String fridgeName;

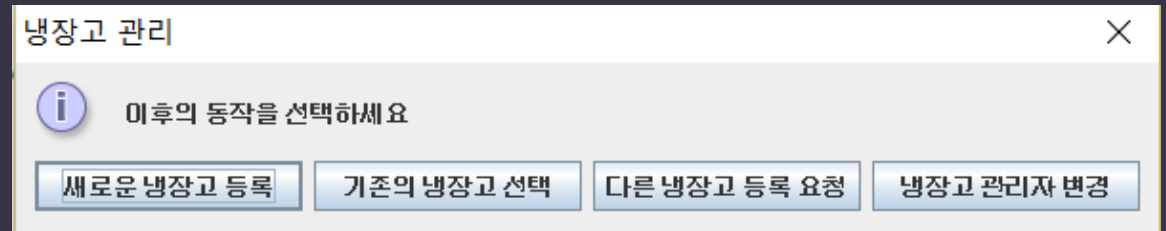
    public MatchFridge() { }

    public MatchFridge(FridgeKey fKey, String[] users, String fridgeName) {
        this.fKey = fKey;
        this.users = users;
        this.fridgeName = fridgeName;
    }
    public FridgeKey getfKey() {
        return fKey;
    }
    public void setfKey(FridgeKey fKey) {
        this.fKey = fKey;
    }
    public String[] getUsers() {
        return users;
    }
    public void setUsers(String[] users) {
        this.users = users;
    }
    public String getFridgeName() {
        return fridgeName;
    }
    public void setFridgeName(String fridgeName) {
        this.fridgeName = fridgeName;
    }
    public String toString() {
        return "FridgeKey: " + fKey + ", users: " + users + ", fridgeName: " + fridgeName;
    }
}
```

<Value>

냉장고 관련 세부 동작 선택

- “냉장고 선택”이 이루어진 후..
 - 새로운 냉장고 등록: INSERT
 - 기존의 냉장고 선택: SELECT
 - 다른 냉장고 등록 요청: INSERT
 - 냉장고 관리자 변경: UPDATE



새로운 냉장고 등록

- 새로운 냉장고 등록 수행 시, 새로운 냉장고를 등록하는 사용자가 관리자가 됨
- USER_ID의 시작은 항상 관리자

새로운 냉장고 등록

사용하실 냉장고 아이디를 입력하세요
(영문 혹은 숫자 5글자 ~ 12글자로 입력하세요)

NEW ONE

쉽게 알아볼 수 있도록 냉장고 이름을 입력하세요
(한글 입력 가능, 빈 칸 가능)

TEST(연습)

확인취소



등록 성공!

i

새로운 냉장고 등록이 성공했습니다.

확인



USER_ID	FRIDGE_ID	FRIDGE_NAME	MGR_USER
USER07:	NEW ONE	TEST(연습)	USER07

```
String[] users = inMf.getUsers();
for(int i = 0; i < users.length; i++){
    unite += users[i] + ":";
}
```

기존의 냉장고 선택

- 관리자로 등록된 냉장고나 일반 사용자로 등록된 냉장고 중 하나 선택

USER_ID	FRIDGE_ID	FRIDGE_NAME	MGR_USER
USER01:USER02:	F01	HOME	USER01
USER04:USER06:	F02	HONE	USER04
USER03:USER04:USER05:	F03	HOME	USER03

```
USER(
FridgeKey fk = new FridgeKey();
USER(

fk.setMgrUser(uId);
fk.setFridgeId(rs.getString("FRIDGE_ID"));

MatchFridge mf = new MatchFridge();
mf.setfKey(fk);
String[] users = rs.getString("USER_ID").split(":");
mf.setUsers(users);
mf.setFridgeName(rs.getString("FRIDGE_NAME"));
```

<USER
- 등록

하나의 냉장고 선택

하나의 냉장고를 선택하세요

관리자

F01

<USER
- USER
- USER

F01
F05

다른 냉장고 등록 요청 (1/2)

- 다른 냉장고에 사용자로 등록하기 위해서는 등록 요청이 이루어져야 함
- “냉장고 관리자:냉장고 아이디” 형태로 입력 – 분리하여 냉장고 정보 유무 판단

<입력 예>

등록된 냉장고와 연결

이미 등록된 냉장고 아이디를 입력하세요
(관리자 아이디:등록된 냉장고 아이디 형식으로 입력하세요)

USER02:F01

관리자에게 보낼 메시지를 입력하세요
(100글자 까지 입력 가능합니다)

허가를 요청합니다

확인

취소

<세 가지 에러 타입>

에러

X

형식에 맞지 않습니다

확인

에러

X

해당 관리자가 관리하는 냉장고가 아닙니다. 다시 확인해주세요

확인

에러

X

관리자 정보가 없습니다. 다시 확인해주세요

확인

다른 냉장고 등록 요청 (2/2)

- 등록 요청을 저장하는 별도의 DB를 구성

BLACK.ASK_PERMISSION	
P * MGR_USER	VARCHAR2 (15 BYTE)
P * FRIDGE_ID	VARCHAR2 (15 BYTE)
P * REQ_USER	VARCHAR2 (15 BYTE)
REQ_MSG	VARCHAR2 (100 BYTE)
PK_ASKPERMISSION (MGR_USER, FRIDGE_ID, REQ_USER)	
PK_ASKPERMISSION (MGR_USER, FRIDGE_ID, REQ_USER)	

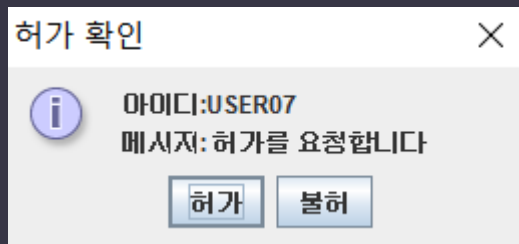
⚡ COLUMN_NAME	⚡ DATA_TYPE	⚡ NULLABLE	DATA_DEFAULT	⚡ COLUMN_ID	⚡ COMMENTS
1 MGR_USER	VARCHAR2 (15 BYTE)	No	(null)	1	등록 요청한 냉장고 관리자
2 FRIDGE_ID	VARCHAR2 (15 BYTE)	No	(null)	2	등록 요청한 냉장고
3 REQ_USER	VARCHAR2 (15 BYTE)	No	(null)	3	등록 요청한 사용자
4 REQ_MSG	VARCHAR2 (100 BYTE)	Yes	(null)	4	사용자의 요청 메시지

MGR_USER & FRIDGE_ID & REQ_USER 묶음을 Primary Key로 지정
- 하나의 냉장고에 대해 사용자가 하나의 요청만 가능하도록 하기 위해

요청에 대해 Primary Key로 묶어서 입력 시 중복을 막지만,
별도의 중복체크 메소드로 중복에 대한 에러 메시지 출력

다른 냉장고 등록 요청 (3/3)

- 등록을 요청받은 관리자는 로그인 시 곧바로 메시지로 요청에 대한 확인 가능
 - 관리자가 허가를 선택하면, ASK_PERMISSION 테이블에서 관련 정보 삭제
MATCH_FRIDGE 테이블에서 사용자 리스트에 추가



	USER_ID	FRIDGE_ID	FRIDGE_NAME	MGR_USER
1	USER01:USER02:	F01	HOME	USER01
2	USER04:USER06:	F02	HONE	USER04
3	USER03:USER04:USER05:	F03	HOME	USER03
4	USER06:	F04	MINE	USER06
5	USER01:USER04:USER07:	F05	WORK	USER01

	USER_ID	FRIDGE_ID	FRIDGE_NAME	MGR_USER
1	USER01:USER02:	F01	HOME	USER01
2	USER04:USER06:	F02	HONE	USER04
3	USER03:USER04:USER05:	F03	HOME	USER03
4	USER06:USER07:	F04	MINE	USER06
5	USER01:USER04:USER07:	F05	WORK	USER01

냉장고 관리자 변경

- 관리자 변경에 대한 요구를 처리하기 위한 코드는 작성
 - MATCH_FRIDGE 테이블에서 USER_ID 리스트에서 새로운 관리자를 맨 앞으로 두고, 이전 관리자를 리스트의 2번째에 오도록 처리
- 관리자 탈퇴 시 활성화 되도록 구현 필요
 - 새롭게 관리자가 된 사용자에게 알림을 주기 위한 메시지 DB가 미 구현 상태

```
sql = "UPDATE MATCH_FRIDGE"
      + " SET MGR_USER = ?, USER_ID = ?"
      + " WHERE MGR_USER = ? AND FRIDGE_ID = ?";

ps = con.prepareStatement(sql);
ps.setString(1, nMgr);

String[] users = mf.getUsers();
uList = nMgr + ":";
for(int i = 0; i < users.length; i++){
    if(!users[i].equals(nMgr)){
        uList += users[i] + ":";
    }
}
ps.setString(2, uList);
ps.setString(3, mf.getfKey().getMgrUser());
ps.setString(4, mf.getfKey().getFridgeId());
```


냉장고 아이템 관리

- 사용자에게 등록된 냉장고를 하나 선택하면 4개의 메인 동작을 선택할 창이 발생
 - 냉장고 재고 관리
 - 패턴 분석
 - 소진해야 할 재료
 - 추천 레시피



아이템 관리부

<아이템에 대한 모든 관련 클래스들>

<포함 코드>

```
* fridge.view
```

- ActionSelectDialog.java
- ItemManageDialog.java
- NeedToExhaustDialog.java
- PatternAnalysisDialog.java
- RecipeBook.java

```
* fridge.model
```

- ItemDAO.java
- UsedItemDAO.java

* fridge.control

- ItemController.java
- UsedItemController.java

```
* objectBox.fridge
```

- ItemInfo.java
- FridgeKey.java
- UsedItemInfo.java

```
* recipe.model.service
```

- ## - RecipeService.java

```
* recipe.model.vo
```

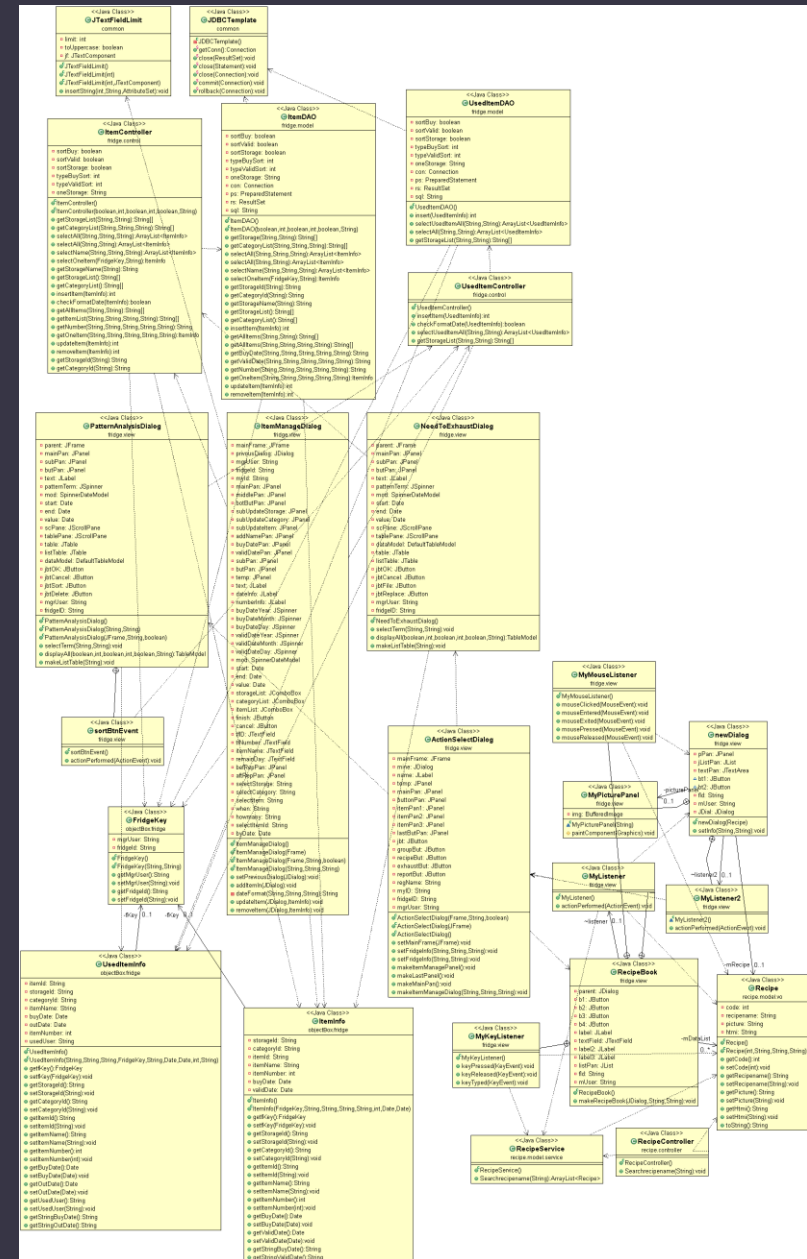
- ## - Recipe.java

```
* recipe.controller
```

- ## - RecipeController.java

* common

- JTextFieldLimit.java
- JDBCTemplate.java



냉장고 재고 관리

냉장고 재고 관리

아이템 이름:

품목	구입일자	유통기한	남은 기한
소고기	2016년 01월 27일	2016년 02월 21일	25
배추	2016년 01월 27일	2016년 02월 25일	29
상추	2016년 01월 27일	2016년 03월 02일	35
표고버섯	2016년 01월 27일	2016년 03월 02일	35
11	2016년 01월 27일	2016년 03월 02일	35
222	2016년 01월 27일	2016년 03월 02일	35
라면	2016년 01월 27일	2016년 03월 02일	35
우유	2016년 01월 27일	2016년 03월 02일	35
김치	2016년 01월 27일	2016년 03월 02일	35
111	2016년 01월 27일	2016년 03월 02일	35

보기 옵션 (다중 선택 가능)

☒ 저장공간 기준

```
ArrayList<ItemInfo> list
= new ItemController(sortBuy, typeBuySort, sortValid, typeValidSort, sortStorage, oneStorage).
    selectAll(mgrUser, fridgeID);
data = new Object[list.size()][4];
for(int i = 0; i < list.size(); i++)
{
    data[i] = new Object[4];
    ItemInfo p = list.get(i);

    data[i][0] = p.getItemName();
    data[i][1] = p.getStringBuyDate();
    data[i][2] = p.getStringValidDate();

    long day = (p.getValidDate().getTime() - p.getBuyDate().getTime()) /
        (24 * 60 * 60 * 1000);
    data[i][3] = (int)day;
}
```

< JTable >

: ArrayList로 읽어온 내용을
이중 배열로 삽입

냉장고 재고 관리: 보기옵션 - 정렬

냉장고 재고 관리

아이템 이름:

품목	구입일자	유통기한	남은 기한
소고기	2016 년 01 월 27 일	2016 년 02 월 21 일	25
배추	2016 년 01 월 27 일	2016 년 02 월 25 일	29
상추	2016 년 01 월 27 일	2016 년 03 월 02 일	35
표고버섯	2016 년 01 월 27 일	2016 년 03 월 27 일	60
11	2016 년 01 월 28 일	2016 년 01 월 30 일	2
222	2016 년 01 월 28 일	2016 년 03 월 28 일	60
라면	2016 년 01 월 28 일	2016 년 03 월 28 일	60
우유	2016 년 01 월 28 일	2016 년 04 월 02 일	65
김치	2016 년 01 월 28 일	2017 년 02 월 28 일	397
111	2016 년 02 월 27 일	2016 년 04 월 27 일	60

보기 옵션 (다중 선택 가능)

☒ 저장공간 기준
☐ 구입기한 기준
☐ 유통기한 기준

상세 정보 (품목 선택) ----

< 보기 옵션 >: 다중 선택 가능을 위해 체크박스 구성

: JTable 내의 항목 정렬을 위해 위치

-> 저장공간 기준)

=> 아이템이 저장된 공간들을 DB로부터 읽어와서 JComboBox로 옵션 제공

-> 구입기한/유통기한: 오름차순/내림차순의 두 순서 정렬

=> SQL 문에서 ORDER BY 옵션을 통해 읽어온 결과로 보이기

냉장고 재고 관리: 상세보기/수정/삭제

냉장고 재고

아이템 수정

아이템

수정할 아이템 이름: 소고기

삭제 가능) 보기

품목

구입일자: 2016년 01월 27일

구입일자: 2016년 01월 27일

유통기한: 2016년 02월 21일

유통기한: 2016년 02월 21일

아이템의 개수/중량: 100

수정할 아이템의 개수/중량을 입력하세요

완료 취소

개수/중량:

준

I준

I준

! (품목 선택) ---

< 상세 보기 >: 테이블 내의 항목 선택
: 테이블의 품목 선택
-> 테이블의 품목 선택
수정 / 삭제에

냉장고 재고 관리: 아이템 추가

냉장고 재고 관리

아이템 이름: 검색

품목	구입일자	유통기한	남은 기한
소고기	2016년 01월 27일	2016년 02월 21일	25
배추	2016년 01월 27일	2016년 02월 25일	29
상추	2016년 01월 27일	2016년 03월 02일	35
표고버섯	2016년 01월 27일	2016년 03월 27일	60
11	2016년 01월 28일	2016년 01월 30일	2
222	2016년 01월 28일	2016년 03월 28일	60
라면	2016년 01월 28일	2016년 03월 28일	60
우유	2016년 01월 28일	2016년 04월 02일	65
김치	2016년 01월 28일	2017년 02월 28일	397
소고기	2016년 01월 29일	2016년 01월 31일	2
111	2016년 02월 27일	2016년 04월 27일	60

보기 옵션 (다중 선택 가능) 보기

☒ 저장공간 기준

☐ 구입기한 기준

☐ 유통기한 기준

--- 상세 정보 (클릭) ---

저장공간:

상품명:

구입기일:

유통기한:

개수/종량:

추가 수정 삭제

아이템 입력

추가할 아이템의 저장공간을 선택하세요

추가할 아이템의 카테고리를 선택하세요

추가할 아이템을 입력하세요

구입 날짜를 선택하세요 2016년 01월 29일

유통 기한을 선택하세요 2016년 01월 29일

추가할 아이템의 개수/종량을 입력하세요

완료 취소

로그아웃 시스템 종료

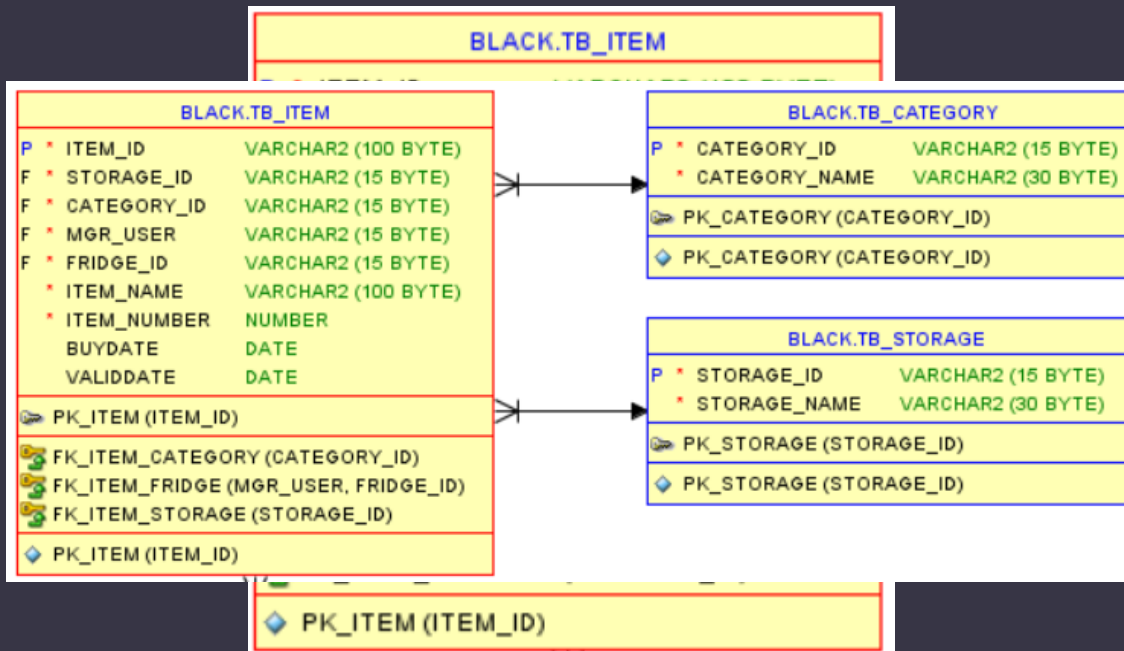
```
> MemberLogin.java
냉장고 관리.ucls
사용자.ucls
아이템 관리.ucls
전체.ucls
> JRE System Library [JavaSE-1.8]
> ojdbc6.jar - C:\Woraclexe\app\Woracle\product\11.2\...

// 주성일
updatePa
//update
UserInfo
if(choos
mergeran = new JPanel(new GridLayout(3,1));
```

아이템 관리 테이블

입력 (TB_ITEM)

- 새롭게 추가되는 아이템 저장용



사용 (USED_ITEM)

- 아이템 수정 시, 개수/용량이 감소한 아이템은 사용했다는 전제조건
- 아이템 삭제 시, 아이템이 사용되었다는 전제 조건

BLACK.USED_ITEM	
* ITEM_ID	VARCHAR2 (15 BYTE)
* STORAGE_ID	VARCHAR2 (15 BYTE)
* CATEGORY_ID	VARCHAR2 (15 BYTE)
* MGR_USER	VARCHAR2 (15 BYTE)
* FRIDGE_ID	VARCHAR2 (15 BYTE)
* ITEM_NAME	VARCHAR2 (20 BYTE)
* BUYDATE	DATE
	USEDDATE DATE
* USED_NUMBER	NUMBER
* USED_USER	VARCHAR2 (15 BYTE)

패턴 분석 / 소진해야 할 재료

패턴 분석

품목	구입일자	소비날짜	소비기간	소비개수
QQQQ	2016년 01월 27일	2016년 01월 28일	1	8
QQQQ	2016년 01월 27일	2016년 01월 28일	1	10
QQQQ	2016년 01월 27일	2016년 01월 28일	1	8
QQQQ	2016년 01월 27일	2016년 01월 28일	1	19
111	2016년 01월 27일	2016년 01월 28일	1	11
QQQQ	2016년 01월 27일	2016년 01월 28일	1	6
QQQQ	2016년 01월 27일	2016년 01월 28일	1	8
QQQQ	2016년 01월 27일	2016년 01월 28일	1	9
QQQQ	2016년 01월 27일	2016년 01월 28일	1	8
111	2016년 01월 27일	2016년 01월 28일	1	23
QQQQ	2016년 01월 27일	2016년 01월 28일	1	3
소고기	2016년 01월 27일	2016년 01월 28일	1	1000
소고기	2016년 01월 27일	2016년 01월 28일	1	100
표고버섯	2016년 01월 27일	2016년 01월 28일	1	200
ASD	2016년 01월 28일	2016년 01월 28일	0	22
111	2016년 01월 27일	2016년 01월 28일	1	2
111	2016년 01월 27일	2016년 01월 28일	1	8
돼지고기	2016년 01월 27일	2016년 01월 28일	1	1000

파일로..
확인

소진해야 할 재료

품목 이름	저장 공간	남은개수	구입 날짜
상추	냉장실	200	2016년 01월 27일
배추	냉장실	1	2016년 01월 27일
표고버섯	냉동실	100	2016년 01월 27일
111	냉장실	500	2016년 02월 27일
소고기	냉동실	100	2016년 01월 27일
222	그 외	900	2016년 01월 28일
라면	푸드박스	5	2016년 01월 28일
11	냉동실	11	2016년 01월 28일
김치	김치냉장고	555	2016년 01월 28일
우유	그 외	3	2016년 01월 28일

파일로..
확인

추천 레시피

오늘의 추천요리

재료입력검색

아래의 목록이 검색되었습니다.*****
원하시는 항목을 선택 후 확인 버튼을 누르세요.****

- 감자전
- 허니버터감자튀김
- 감자옹심이
- 감자탕
- 감자볶음

<이벤트 처리>

- 버튼 클릭
- 키보드: 엔터 키
- 마우스리스너

```
textPan = new JTextArea();  
//라인 밑으로 떨어지는메소드  
textPan.setLineWrap(true);  
  
//개행 관련  
String desc = mRecipe.getHtml();  
desc = desc.replace("\\n", "\n");  
textPan.append(desc);
```

얼마나 맛있게요




■재료 (옹심이 2인분)
주재료 - 감자 4개
부재료 - 당근 1/2개, 애호박 1/2개, 다진 마늘 1/2T, 달걀 1개, 국간장 1T, 소금 약간
육수 재료 - 다시마, 국물용 멸치 15마리, 대파 1대, 무 한 토막

■만드는 법?
· 냄비에 물 5~6컵과 육수 재료를 넣고 끓으면 다시마를 건져내고 10분 정도 더 끓인 후 육수만 걸러준다.
· 감자는 껍질을 벗겨 강판에 갈고, 면포로 물기를 짜고, 나온 물은 양금을 가라앉힌다.
· 물기를 뺀 감자와 가라앉힌 양금, 소금 약간을 넣고 반죽해 동그랗게 옹심을 만들어준다.
· 끓는 육수에 옹심을 넣고 익으면 국간장과 다진마늘을 넣고 간을 맞춘다.
· 옹심이 위로 떠오르면 다 익었다는 거예요~)

다양한 요리로 맛있게 식사하세요. 다들 즐겨주세요. (2인분)

추천 레시피 관리 테이블

BLACK.RECIPE	
P * CODE	NUMBER
RECIPENAME	VARCHAR2 (4000 BYTE)
PICTURE	VARCHAR2 (4000 BYTE)
HTMI	VARCHAR2 (4000 BYTE)
 RECIPE_PK (CODE)	

“이미지”는 DB 내부에 경로로 저장
-> 메뉴가 호출되면 경로 상의 이미지가 업로드

IPENAME	PICTURE	HTML
! {태감자튀김	./이워지/감자전.png	재료감자(중간크기) 5개, 소금 약간, 포도씨유 들기름 적당량씩만드는 법1. 감자 4개는 껍질을 벗겨 곱게 쳐 >감자 2개, 버터 1조각, 꿀 2큰술, 건파슬리 약간, ?물 2컵, 소금 2작은술, 식용유<순서>1. ?먼저,
!심이	./이워지/감자옹심이.png	■재료 (옹심이 2인분)주재료 - 감자 4개부재료 - 당근 1/2개, 애호박 1/2개, 다진 마늘 1/2t, 달걀 1개, -돼지 등뼈 손질반-1. 돼지 등뼈 6개(1,300g) 준비 2. 냄비에 등뼈를 넣고 등뼈가 잠길 정도로 물을 넣고
!}	./이워지/감자탕.png	<재료>감자 250g, 청피망 20g (약1/5개), 홍피망 20g (약1/5개),양파 70g (약1/4개), 당근 20g (약1/6
5 감자볶음	./이워지/감자볶음.png	재료 : 꿀병이 통조림 1캔, 통조림 국물 6큰 술, 다진마늘 1큰 술, 초장 1큰 술, 대우포 1줌, 소금 1/2큰
6 (간식)꿀병이무침	./이워지/꿀병이무침.png	*재료김 15장, 쌀 3컵, 물 4컵+2큰 술, 소금 1+2/5큰 술참기름 3큰 술, 단무지 10개, 햄 20개, 오이 2개
7 (간식)김밥	./이워지/김밥.png	* 주재료: 밀가루 약 4줌 (400g), 어묵 3장 * 양념장 재료: 고운 고춧가루 18 큰 술, 설탕 10 큰 술, 물
8 (간식)떡볶이	./이워지/떡볶이.png	재료 : 물, 소면 200g(500원 동전 크기), 고추장 5큰 술, 고춧가루 5큰 술, 간 마늘 1큰 술, 설탕 5큰 술
9 (면요리)비빔국수	./이워지/비빔국수.png	* 반죽 (4인분 기준)재료 : 식용유 4 큰 술, 물 1컵(200cc), 밀가루 4컵(800cc) 조리과정 :1.식용유 4
10 (간식)수제비	./이워지/수제비.png	재료 : 열무 한 단, 참쌀가루 1컵(200cc), 물 17컵(3,400cc), 소금 5큰 술, 고춧가루 9큰 술, 다진 마늘
11 (면요리)열무국수	./이워지/열무국수.png	* 면 재료: 생면 3인분(450g), 물 3,000cc(200cc 맥주컵 15개 분량), 소금 반 큰 술* 시중에 판매되는
12 (면요리)잔치국수	./이워지/잔치국수.png	주재료고구마 200g, 우유 1 1/2컵 양념 및 소스재료꿀 1큰술, 연유 1작은술, 소금 약간?선택재료생크림 적
13 (차)고구마라떼	./이워지/고구마라떼.png	- 주재료 : 고구마 2개(300g) · 부재료 : 식용유(튀김용) 4컵(800ml), 시럽(설탕 1큰술(10g), 물엿 3큰술
14 (간식)고구마맛탕	./이워지/고구마맛탕.png	- 주재료 : 고구마 1개(200g), 밀가루 1/2컵(50g)+ 3큰술(15g) · 부재료 : 식용유 3컵(600ml), 달걀(달걀
15 (간식)고구마튀김	./이워지/고구마튀김.png	- 주재료 : 고구마 2와 1/2개(500g) · 부재료 : 대파(대파 흰 부분) 20cm(40g), 양파 1/2개(100g), 닭육
16 고구마스프	./이워지/고구마스프.png	재료 : 등심(280g) 4개, 안심(240g) 4개, 달걀 2개, 밀가루 2큰 술, 전분 2큰 술,물 6큰 술, 식용유, 소
17 (고기)돈가스	./이워지/돈가스.jpg	레시피-준비 과정-1. 돼지 목전지 600g, 2mm 두께로 준비 전지, 목살도 사용 가능-----
18 (고기)돼지불고기	./이워지/돼지불고기.jpg	재료 쇠고기(갈비살) 400g, 간장 2큰술, 다진 마늘 1작은술, 다진 파 2작은술, 소금·후춧가루·통깨 약간
19 (고기)떡갈비	./이워지/떡갈비.jpg	* 주재료: 불고기용 고기 4줌(600g), 물에 불린 당면 4줌, 양파 한 개, 대파 반쪽, 당근 반 개, 간 마늘(
20 (고기)떡배기불고기	./이워지/떡배기불고기.jpg	<우밍치> (무 1개 기준) 재료: 무 1개, 소금 5큰 술, 고춧가루 8큰 술, 미나리 1줌, 쪽파 1줌, 생들 2
21 (고기)보쌈	./이워지/보쌈.jpg	재료 : 닭 2마리(5호, 약 500g), 물 10컵 반(2,100cc), 참쌀 1공기, 수삼 2뿌리, 마늘 4쪽, 대추 4알,
22 (고기)삼계탕	./이워지/삼계탕.jpg	재료 : 물 4컵, 닭 1마리, 간장 13큰 술, 커피 1큰 술,설탕 3큰 술, 물엿 6큰 술,감자 2개, 당근 1/3개,
23 (고기)찜닭	./이워지/찜닭.jpg	재료 : 닭 1마리(9호, 851-950g), 기름 2L, 간 양파 5큰 술, 간 마늘 2+1큰 술, 간 생강 2큰 술, 설탕 2
24 (고기)치킨	./이워지/치킨.jpg	*소스 만들기(3인분 기준) 재료 : 물 1컵 반(300cc), 설탕 8큰 술, 2배 식초 5큰 술, 중국 간장(노추) 1
25 (고기)탕수육	./이워지/탕수육.jpg	재료돼지고기 항정살 300g, 마늘 5쪽, 대파 1/2대 양념 재료된장 1큰술, 물리고당 3/4큰술, 설탕 1/2큰술
26 (고기)항정살된장구이	./이워지/항정살된장구이.jpg	-닭 손질반-1. 닭 1.2kg 1마리 기준2. 꼬리, 가슴, 목의 지방 제거3. 물 1,200cc(맥주컵 6컵) 녹차 가루
27 (김치) (고기)김치닭볶음탕	./이워지/김치닭볶음탕.png	

향후 발전 고려 사항

- 관리자 변경 기능
- 패턴 분석 / 소진해야 할 아이템 부분의 정렬 기능 & 파일로 내 보내는 기능
- DB의 문제 발생을 대비하여 백업 파일 준비 필요
- 디자인적인 부분.....