7주차

나만의 냉장고

2020213311
신중근

목차

진행상황

식재료 데이터
DB에 저장될 식재료 정보를 미리 정의
식재료 정보: 남은 유통기한, 1인분 수량, 저장된 수량

```
// 각 식재료의 유통기한 (임의의 예시)
Map<String, Integer> ingredientExpirations = new HashMap<>();
ingredientExpirations.put(key:"고기", value:3);
ingredientExpirations.put(key:"당근", value:5);
ingredientExpirations.put(key:"감자", value:2);
ingredientExpirations.put(key:"양파", value:4);
ingredientExpirations.put(key:"토마토", value:3);
ingredientExpirations.put(key:"브로콜리", value:6);
ingredientExpirations.put(key:"두부", value:7);
ingredientExpirations.put(key:"계란", value:10);
```

```
// 각 식재료의 1인분 수량 (임의의 예시)
Map<String, Integer> ingredientPortions = new HashMap<>();
ingredientPortions.put(key:"고기", value:200);
ingredientPortions.put(key:"당근", value:100);
ingredientPortions.put(key:"감자", value:150);
ingredientPortions.put(key:"양파", value:80);
ingredientPortions.put(key:"토마토", value:120);
ingredientPortions.put(key:"브로콜리", value:100);
ingredientPortions.put(key:"두부", value:200);
ingredientPortions.put(key:"계란", value:50);
```

```
// 각 식재료의 저장된 수량 (임의의 예시)
Map<String, Integer> ingredientQuantities = new HashMap<>();
ingredientQuantities.put(key:"고기", value:400);
ingredientQuantities.put(key:"당근", value:300);
ingredientQuantities.put(key:"감자", value:500);
ingredientQuantities.put(key:"양파", value:200);
ingredientQuantities.put(key:"토마토", value:300);
ingredientQuantities.put(key:"브로콜리", value:150);
ingredientQuantities.put(key:"두부", value:300);
ingredientQuantities.put(key:"계란", value:120);
```

진행상황

DietNet 식생활정보센터
www.dietnet.or.kr

	분류	식품명	분량(g)	비고	섭취 횟수
곡류 (300kcal)	곡류	쌀,보리쌀,찹쌀,현미,혼합잡곡 쌀밥,보리밥	90 210	 1공기	1회 1회
	면류	삶은 면 생면-우동,칼국수 건면-국수용 냉면국수,메밀국수	300 200 100 100	1대접 국수묶음 두께2.5cm 국수묶음 두께2.5cm	1회 1회 1회 1회
	떡류	흰떡-떡국용 찹쌀떡,시루떡	130 130	1컵(썬 것)	1회 1회
	빵류	식빵,빵류	100	큰 것 2쪽,1개	1회
	씨리얼류	콘프레이크등	40		0.5회
	감자류	감자 고구마	130 130	중1개 중1/2	0.5회 0.5회
	기타	메밀묵 밤	200 100	큰 것 5개	0.5회 0.5회

	분류	식품명	분량(g)	비고	섭취 횟수
고기・생선 계란・콩류 (1000kcal)	육류	쇠고기 돼지고기 닭고기 햄	60 60 60 60		1회 1회 1회
	어패류	갈치, 삼치, 꽁치, 고등어, 동태, 가자미, 조기, 넙치, 참치, 참치통조림, 어묵, 미꾸라지, 민물장어	60	작은 것 한 토막	1회
		생굴,조갯살,꽃게 오징어,낙지,새우 건멸치,건조기,건오징어	80 80 15		1회 1회 1회
	난류	계란,메추리알	60	계란(중) 1개 메추리알 5개	1개
	콩류	검정콩,대두 두부 두유	20 80 200		1회 1회 1회
	견과류	땅콩 깨 호두	10 10 10	15알내외 1스푼	0.5회 0.5회 0.5회

	분류	식품명	분량(g)	비고	섭취 횟수
채소류 (15kcal)	채소류	고구마줄기, 고사리, 시금치, 풋고추, 근대, 깻잎, 무청, 부추, 돌미나리, 배추, 상추, 쑥, 쑥갓, 아욱,취나물, 애호박, 두릅, 오이, 콩나물, 숙주나물, 머위, 무, 배추, 양배추, 양파, 가지, 당근, 늙은 호박, 토마토	70	1접시	1회
		나박김치,오이소박이,동치미 갓김치,깍도기,배추김치,열무김치,종각김치,단무지	60 40		1회 1회
		우엉,도라지,파,파김치 마늘 토마토주스	25 10 100		1회 1회 1회
	해조류	다시마,미역,파래(생것) 김	30 2		1회
	버섯류	느타리,양송이,팽이,표고(생것)	30		1회

	분류	식품명	분량(g)	비고	섭취 횟수
과일류 (50kcal)	과일류	딸기, 수박, 참외 귤, 감, 바나나, 망고, 키위, 사과, 배, 복숭아, 오렌지, 포도, 파인애플	100	딸기 10개 귤 중1개 사과(중)1/2개	1회
	주스류	오렌지주스, 귤주스, 사과주스, 포도주스	100		1회

	분류	식품명	분량(g)	비고	섭취 횟수
우유・ 유제품류 (125kcal)	우유	우유	200	1컵 (1개)	1회
	유제품	치즈 요구르트(호상) 요구르트(호상) 아이스크림	20 100 150 100	1장 1/2컵(1개) 3/4컵(1개) 1/2컵(1개)	0.5회 1회 1회 1회

진행상황

가중치 계산
각 식재료에 가중치를 부여하는 로직
유통기한과 수량을 고려하여 계산
1순위: 유통기한, 2순위: 수량

```java
// 확률을 계산할 변수 초기화
double totalWeight = 0.0;

// 유통기한과 수량을 기반으로 가중치 부여
Map<String, Double> ingredientWeights = new HashMap<>();
for (String ingredient : userIngredients) {
    int expiration = ingredientExpirations.getOrDefault(ingredient, Integer.MAX_VALUE);
    int portion = ingredientPortions.getOrDefault(ingredient, defaultValue:100);
    int quantity = ingredientQuantities.getOrDefault(ingredient, defaultValue:0);

    double expirationWeight = 1.0 / Math.pow((1 + expiration), b:2); // 유통기한을 기반으로 하는 가중치
    double quantityWeight = quantity / (double) portion; // 수량을 기반으로 하는 가중치
    double weight = expirationWeight * quantityWeight; // 유통기한과 수량을 고려한 가중치

    ingredientWeights.put(ingredient, weight);
    totalWeight += weight; // 총 가중치 계산
}
```

$$\text{가중치} = \frac{1}{(1+\text{유통기한})^2} \times \frac{\text{남은 수량}}{\text{1인분 수량}}$$

진행상황

식재료 추천 알고리즘
0부터 총 가중치 사이의 난수 생성
각 식재료의 가중치를 누적하여 더함
누적된 가중치가 생성된 난수를 넘으면 해당 식재료를 선택

```java
// 레시피 추천 및 결과 리스트 초기화
List<List<String>> recommendedRecipesList = new ArrayList<>();
Random random = new Random();

// 레시피별로 선택된 식재료 기록을 위한 Map
Map<List<String>, Integer> recipeFrequency = new HashMap<>();

for (int i = 0; i < numIterations; i++) {
    List<String> recommendedRecipes = new ArrayList<>();
    Set<String> selectedIngredients = new HashSet<>();

    while (selectedIngredients.size() < numReturnedIngredients) {
        double randomValue = random.nextDouble() * totalWeight; // 0부터 총 가중치 사이의 난수 생성
        double cumulativeWeight = 0.0;

        for (Map.Entry<String, Double> entry : ingredientWeights.entrySet()) {
            cumulativeWeight += entry.getValue();
            if (cumulativeWeight >= randomValue) {
                String ingredient = entry.getKey();
                selectedIngredients.add(ingredient);
                break;
            }
        }
    }

    recommendedRecipes.addAll(selectedIngredients);
    recommendedRecipesList.add(recommendedRecipes);

    // 레시피별로 선택된 식재료 기록 업데이트
    if (recipeFrequency.containsKey(recommendedRecipes)) {
        recipeFrequency.put(recommendedRecipes, recipeFrequency.get(recommendedRecipes) + 1);
    } else {
        recipeFrequency.put(recommendedRecipes, value:1);
    }
}
```

진행상황

식재료 추천 결과
유통기한이 짧을수록 더 높은 확률로 추천
유통기한이 길더라도 수량이 많으면 더 높은 확률로 추천

식재료	유통기한	남은 수량/1인분
고기	3	2
토마토	3	2.5
브로콜리	6	1.5
우유	7	5

```
각 식재료의 빈도수:
감자: 8013회
토마토: 4885회
고기: 4213회
양파: 3403회
당근: 2908회
우유: 2808회
브로콜리: 1150회
두부: 838회
계란: 738회
치즈: 615회
마늘: 205회
파프리카: 199회
설탕: 6회
소금: 5회
간장: 4회
면: 4회
스파게티: 2회
밥: 2회
후추: 1회
올리브 오일: 1회
```

진행상황

레시피 검색
DB에서 식재료를 가진 레시피 검색
SELECT * FROM 테이블 WHERE 열 LIKE '%식재료%';

```java
// 식재료를 입력받고, 해당 식재료를 가진 행의 id를 리턴하는 함수
private static List<Integer> getRecipeIdsByIngredient(Connection connection, String ingredient) throws SQLException {
    List<Integer> recipeIds = new ArrayList<>();
    String sql = "SELECT id FROM recipes WHERE ingredient LIKE ?";
    try (PreparedStatement statement = connection.prepareStatement(sql)) {
        statement.setString(parameterIndex:1, "%" + ingredient + "%");
        ResultSet resultSet = statement.executeQuery();
        while (resultSet.next()) {
            int id = resultSet.getInt(columnLabel:"id");
            recipeIds.add(id);
        }
    }
    return recipeIds;
}
```

진행상황

레시피 추천 결과

식재료 a, b, c를 추천한다고 가정, a, b, c를 차례로 DB에 검색한 후 해당 식재료가 포함된 레시피 추출. 레시피가 중복 추출된 횟수를 구하여 가장 많이 추출된 레시피를 리턴

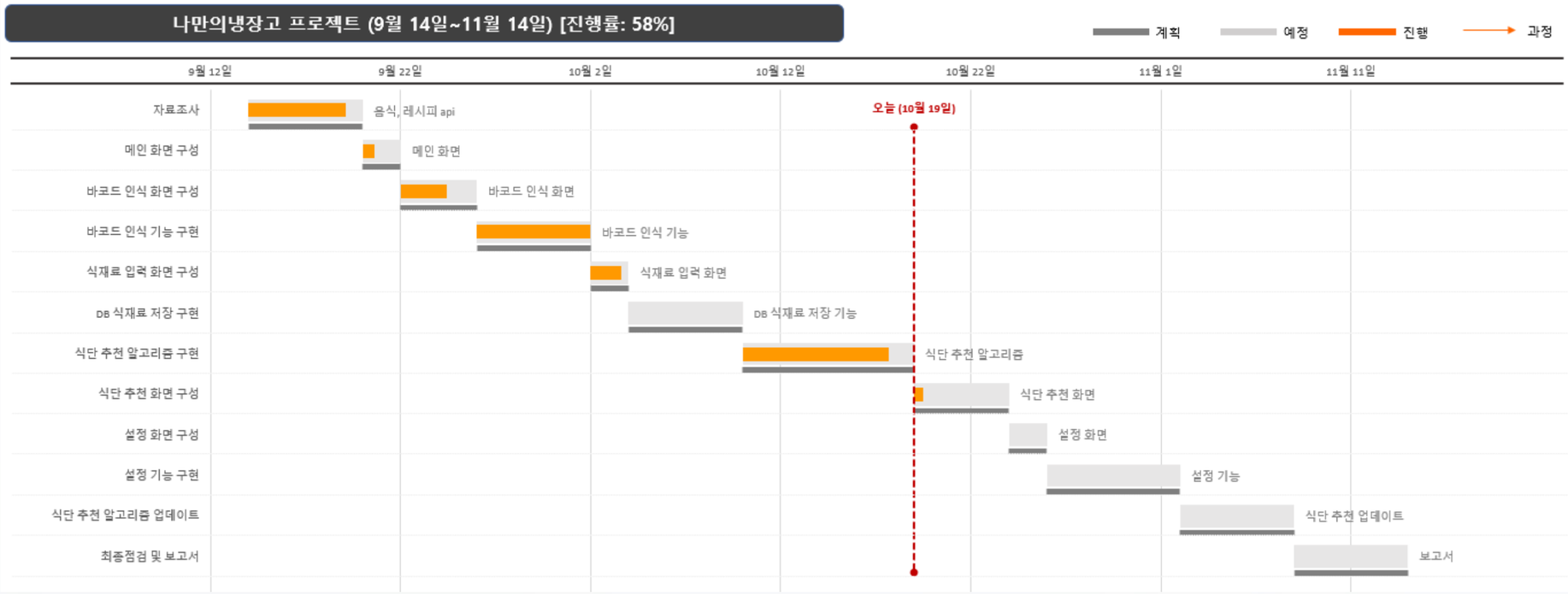
```
url = "url";
id = "id";
password = "password";
Connection = DriverManager.getConnection(url, id, password);

// 레시피별로 불러온 횟수를 저장하는 Map
Map<Integer, Integer> recipeLoadFrequency = new HashMap<>();

// 식재료 결과 제공
System.out.println(x:"추천된 식재료:");
for (List<String> recommendedRecipes : recommendedRecipesList) {
    System.out.println(recommendedRecipes);
    for (String ingredient : recommendedRecipes) {
        System.out.print(ingredient + " ");
        List<Integer> recipeIds = getRecipeIdsByIngredient(connection, ingredient);
        for (int id : recipeIds) {
            System.out.println("레시피 ID: " + id);
            // 레시피 ID가 이미 Map에 있으면 값을 증가시키고, 없으면 1로 초기화
            recipeLoadFrequency.put(id, recipeLoadFrequency.getOrDefault(id, defaultValue:0) + 1);
        }
    }
    System.out.println(x:"");
}

// 가장 많이 불러온 횟수
int maxLoadFrequency = Collections.max(recipeLoadFrequency.values());

// 가장 많이 불러온 레시피들을 저장하는 리스트
List<Integer> mostFrequentlyLoadedRecipes = new ArrayList<>();

// 가장 많이 불러온 레시피들을 찾아서 리스트에 추가
for (Map.Entry<Integer, Integer> entry : recipeLoadFrequency.entrySet()) {
    if (entry.getValue() == maxLoadFrequency) {
        mostFrequentlyLoadedRecipes.add(entry.getKey());
    }
}

// 가장 많이 불러온 레시피들 출력
System.out.println(x:"\n가장 많이 불러온 레시피들:");
for (int id : mostFrequentlyLoadedRecipes) {
    System.out.println("레시피 ID " + id + ": " + maxLoadFrequency + "회");
}
```

다음 주 할 일

DB **생성, 연동**

레시피 DB**에 저장**

간트차트


나만의냉장고 프로젝트 (9월 14일~11월 14일) [진행률: 58%]
계획
예정
진행
과정
9월 12일
9월 22일
10월 2일
10월 12일
10월 22일
11월 1일
11월 11일
오늘 (10월 19일)
자료조사
음식, 레시피 api
메인 화면 구성
메인 화면
바코드 인식 화면 구성
바코드 인식 화면
바코드 인식 기능 구현
바코드 인식 기능
식재료 입력 화면 구성
식재료 입력 화면
DB 식재료 저장 구현
DB 식재료 저장 기능
식단 추천 알고리즘 구현
식단 추천 알고리즘
식단 추천 화면 구성
식단 추천 화면
설정 화면 구성
설정 화면
설정 기능 구현
설정 기능
식단 추천 알고리즘 업데이트
식단 추천 업데이트
최종점검 및 보고서
보고서

감사합니다