

Python programming

Savvy plate

Progress Report : 2

Date : 2023/12/09

Name : 안수빈

ID : 214309

1. Introduction (16 pt)

This system is a Python program that recommends recipes based on the ingredients users have and helps manage groceries. When a user inputs the ingredients they currently have, it recommends possible recipes and creates a shopping list to assist with purchasing the necessary ingredients for the recipes. This system can contribute to reducing food waste and economical meal planning.

1) Background (14 pt)

In our busy lives, it is common to forget about purchased ingredients or let them go to waste after they pass their expiration date. Moreover, facing a variety of ingredients in the refrigerator without deciding what to make can cause significant stress for users. To solve this, we have recognized the need to develop a program that not only encourages users to effectively use the ingredients they already have but also optimizes the purchase and consumption of groceries.

This program will suggest recipes that utilize forgotten ingredients and features the ability to track the expiration dates of groceries, alerting users to use them within the timeframe to prevent waste. By managing a list of necessary groceries during shopping, it will promote an economical and environmentally sustainable diet.

2) Project goal

The goal is to develop a program that recommends personalized recipes based on the inventory of ingredients users currently have. This program aims to facilitate efficient management of ingredients and ease the shopping process, enabling users to effectively utilize their groceries, reduce waste, and make informed decisions regarding their ingredients.

3) Differences from existing programs

Most recipe recommendation services require users to actively search for specific recipes, demanding a passive approach to finding recipes that suit their dietary

habits. "Savvy Plate" differentiates itself in the following ways:

- Automatic recipe suggestions: Based on the ingredients users have, "Savvy Plate" automatically suggests recipes. This saves time and provides new cooking inspiration to users.
- Ingredient substitution suggestions: If a user doesn't have a specific ingredient, "Savvy Plate" recommends other ingredients that can be used as substitutes.

2. Functional Requirement

1) Function 1 (Ingredient-based recipe recommendation feature.)

- It analyzes the ingredients entered by the user to recommend possible recipes.

(1) Detailed function 1 (User ingredient input analysis.)

- When a user enters their ingredients, the program analyzes the list of entered ingredients to search a recipe database that can use those ingredients.

(2) Detailed function 2(Alternative ingredient suggestions.)

If some of the ingredients the user has do not exactly match those needed for a recipe, the program will suggest possible alternatives. This feature enhances the flexibility of recipes and helps users to cook easily.

2) Function 2(Ingredient Management and Shopping List Creation)

- Through the recommended recipes, users can identify additional ingredients needed and manage them as a shopping list.

(1) Detailed function 1 (Automatic Identification of Required Ingredients)

Based on the recommended recipes, the program automatically identifies ingredients

that the user does not currently have. It compares the list of entered ingredients with the ingredients list of the recommended recipe to find missing items.

3. Progress

1) Implementation of features

(1) Implemented Feature Name: 자동 레시피 추천 및 쇼핑 리스트 생성

Input/Output:

Input: Users input their available ingredients, separated by commas.

Output: The program displays a list of possible recipes and provides a list of missing ingredients and a shopping list for the selected recipe.

Description: This system takes an input list of ingredients from the user and searches the 'recipes_db' database to recommend recipes that can be made with those ingredients. If the user selects a specific recipe, the system identifies any ingredients that the user does not have and suggests possible substitutes if necessary. Additionally, it generates a shopping list based on this information, and offers the functionality to save this list if the user desires.

Applied Learning:

- **Loops:** The `for` loop is utilized to iterate through all recipes in the `recipes_db` and compare them with the user's ingredients.
- **Conditional Statements:** The `if` statement is employed to check if the user's ingredients match the ingredients of the recipe and to provide appropriate messages.
- **Functions:** Functions such as `find_recipes`, `suggest_alternatives`, `create_shopping_list`, and `print_recipe_instructions` have been defined to modularize the code and enhance reusability.
- **Modules:** The code has been organized into separate files like `main.py`, `function.py`, and `shopping_list_manager.py`, modularizing the code by

functionality.

- **Dictionaries:** The `recipes_db` and `alternatives_db` use dictionaries to manage the list of recipes and the list of ingredient substitutes, respectively.
- **Lists:** Ingredients input by the user are stored in a list, and the recipe ingredients as well as the shopping list are managed using list data structures.

Code screenshot

[main.py]

```
from function import find_recipes, suggest_alternatives, recipes_db, create_shopping_list, print_recipe_instructions
from shopping_list_manager import save_shopping_list

while True:
    # 사용자로부터 재료를 입력받기
    input_ingredients = input("가지고 있는 재료를 쉼표로 구분하여 입력해주세요: ")
    user_ingredients = [ingredient.strip() for ingredient in input_ingredients.split(',')]

    # 가진 재료로 만들 수 있는 레시피 추천
    available_recipes = find_recipes(user_ingredients, recipes_db)
    if available_recipes:
        print("\n가지고 있는 재료로 만들 수 있는 레시피:")
        for idx, recipe in enumerate(available_recipes, 1):
            print(f" - {idx}. {recipe['name']}")
    else:
        print("\n가진 재료로 만들 수 있는 레시피가 없습니다.")

    # 사용자가 만들고 싶은 레시피를 선택할 수 있도록 안내
    selected_recipe_idx = input("\n만들고 싶은 레시피 번호를 입력해주세요: ")
    if selected_recipe_idx.isdigit():
        selected_recipe = available_recipes[int(selected_recipe_idx) - 1]

    # 선택한 레시피가 있다면, 필요한 재료를 체크
    if selected_recipe:
        print(f"\n{selected_recipe['name']} 레시피에 필요한 재료:")
        missing_ingredients = []
        for ingredient in selected_recipe['ingredients']:
            # 재료가 사용자가 가지고 있는 재료인지 확인
            if ingredient in user_ingredients:
                print(f" - {ingredient} (가지고 있음)")
            else:
                missing_ingredients.append(ingredient)
                alternatives = suggest_alternatives(ingredient)
                alternative_text = f" (대체재료: {' , '.join(alternatives)})" if alternatives else ""
                print(f" - {ingredient} (필요함){alternative_text}")
```

```

35         print(f" - {ingredient} (필요함){alternative_text}")
36
37     # 쇼핑 리스트를 생성
38     shopping_list = create_shopping_list(selected_recipe, user_ingredients)
39     if shopping_list:
40         print("\n쇼핑 리스트:")
41         for item in shopping_list:
42             print(f" - {item}")
43     else:
44         print("\n추가로 구매할 재료가 없습니다.")
45
46     # 사용자에게 쇼핑 리스트를 저장할지 질문
47     if shopping_list:
48         save_option = input("쇼핑 리스트를 저장하시겠습니까? (yes/no): ")
49         if save_option.lower() == 'yes':
50             filename = save_shopping_list(shopping_list)
51             print(f"쇼핑 리스트가 {filename}에 저장되었습니다.")
52
53     # 레시피 지침을 출력하는 코드를 추가
54     print_recipe_instructions(selected_recipe['name'], recipes_db)
55     break
56 else:
57     print("선택한 레시피를 찾을 수 없습니다.")
58 else:
59     # 숫자가 아닌 값을 입력한 경우, 다시 레시피 선택 단계로 돌아감
60     continue
61

```

Recipe Recommendation:

Using the find_recipes function from the function module, the system compares user ingredients against the recipes_db to determine possible recipes. It displays a list of recipes the user can prepare, with each recipe assigned a unique number to simplify selection.

Recipe Selection:

The user is then prompted to select a desired recipe by entering its corresponding number. If a valid number is entered, the system proceeds to ingredient verification for the selected recipe.

Ingredient Verification and Shopping List Generation:

For the chosen recipe, the script checks which ingredients are missing from the user's inventory. It calls suggest_alternatives to provide substitutions for any missing ingredients, displaying this information alongside the required ingredients. The create_shopping_list function generates a shopping list of these missing items.

Shopping List Storage:

The user is then asked whether they would like to save the shopping list. If they choose to do so, the save_shopping_list function from the shopping_list_manager module is invoked to save the list, and the filename where it is stored is displayed to the user.

Recipe Instructions Display:

The script calls `print_recipe_instructions` to output the cooking instructions for the selected recipe, providing the user with step-by-step guidance on how to prepare the dish.

Loop Control and User Experience:

If the user inputs a non-digit when selecting a recipe, the script will prompt them to try again, ensuring a robust user experience that handles incorrect inputs gracefully. The loop will continue until a valid recipe selection is made and processed.

[function.py]

```
# 레시피 데이터베이스의 모의 데이터
recipes_db = [
    {
        'name': '토마토 파스타',
        'ingredients': ['토마토', '파스타', '소금', '올리브 오일', '마늘'],
        'instructions': [
            "파스타를 끓는 물에 소금을 약간 넣고 삶아주세요.",
            "토마토와 마늘을 잘게 다진 후 올리브 오일로 볶아 토마토 소스를 만듭니다.",
            "삶은 파스타를 토마토 소스와 잘 섞어주세요."
        ]
    },
    {
        'name': '갈릭 브레드',
        'ingredients': ['빵', '마늘', '버터'],
        'instructions': [
            "마늘을 잘게 다져 버터와 섞습니다.",
            "빵에 마늘 버터를 바르고 오븐에서 바삭하게 구워냅니다."
        ]
    },
    {
        'name': '브루스케타',
        'ingredients': ['토마토', '마늘', '올리브 오일', '소금', '바질', '페퍼'],
        'instructions': [
            "토마토는 주사위 모양으로 잘게 썰고, 마늘은 다져 준비합니다.",
            "빵을 오븐에서 바삭하게 구운 후 마늘을 문지릅니다.",
            "구운 빵 위에 토마토를 올리고 올리브 오일, 소금, 바질, 후추로 양념합니다."
        ]
    }
]

# 대체 재료 데이터베이스의 모의 데이터
alternatives_db = {
    '토마토': ['캔 토마토', '토마토 페이스트'],
    '파스타': ['스파게티', '페투치니'],
    '올리브 오일': ['카놀라 오일', '해바라기 오일'],
    '마늘': ['마늘 분말', '마늘 페이스트'],
    '빵': ['크루통', '바게트'],
    '버터': ['마가린', '식물성 버터'],
    '계란': ['두부'],
    '간장': ['타마리', '코코넛 아미노스', '우스터 소스', '소야 소스'],
    '참기름': ['들기름', '호두 오일'],
    '김': ['노리', '해초 샐러드'],
    '통깨': ['깨소금', '참깨'],
    '크림': ['코코넛 밀크', '캐시넛 크림'],
    '파마산 치즈': ['베지테리언 파마산'],
    '후추': ['흰 후추', '카옌 페퍼'],
    '돼지고기': ['닭고기', '쇠고기'],
    '두부': ['치즈', '유부'],
    '양파': ['샬롯', '대파'],
    '대파': ['쪽파', '양파'],
    '고추장': ['고춧가루', '미소 페이스트'],
    '오이': ['피클', '호박'],
    '식초': ['레몬 주스', '라임 주스'],
    '설탕': ['꿀', '메이플 시럽'],
    '소고기': ['돼지고기', '양고기'],
    '다시마': ['멸치 가루', '해초 가루'],
    '멸치': ['새우 가루', '다시다'],
    '콩나물': ['무', '숙주'],
    '당면': ['우동 면', '라이스 누들'],
    '당근': ['파프리카', '호박'],
    '시금치': ['케일', '청경채'],
    '피망': ['파프리카', '애호박'],
    '양파': ['샬롯', '대파'],
}
```

```

# 사용자가 가진 재료를 기반으로 레시피를 찾는 함수
✓ def find_recipes(user_ingredients, recipes_db):
    # 사용자가 가진 재료를 기반으로 레시피를 찾아 리스트로 반환합니다.
    recommended_recipes = []
    ✓ for recipe in recipes_db:
    ✓     if any(ingredient in user_ingredients for ingredient in recipe['ingredients']):
        recommended_recipes.append(recipe) # 레시피 전체를 추가
    return recommended_recipes

# 대체 재료를 제안하는 함수
✓ def suggest_alternatives(ingredient):
    return alternatives_db.get(ingredient, [])

# 쇼핑 리스트 만드는 함수
✓ def create_shopping_list(selected_recipe, user_ingredients):
    ✓ shopping_list = [ingredient for ingredient in selected_recipe['ingredients']
        if ingredient not in user_ingredients]
    return shopping_list

# 레시피 지침을 출력하는 함수
✓ def print_recipe_instructions(recipe_name, recipes_db):
    recipe = next((r for r in recipes_db if r['name'] == recipe_name), None)
    ✓ if not recipe:
        print("레시피를 찾을 수 없습니다.")
        return

    print(f"\n{recipe['name']} 레시피 지침:")
    ✓ for instruction in recipe.get('instructions', []):
        print(instruction)

```

find_recipes function)

Functionality: This function takes a list of ingredients provided by the user (user_ingredients) and searches the recipes_db database for recipes that can be made with those ingredients.

Operation: It examines each recipe's list of ingredients to check if they are included in the user's available ingredients, adding any matching recipes to the recommended_recipes list.

Return Value: Returns a list of recipes that the user can make with the ingredients they have.

suggest_alternatives function)

Functionality: This function provides alternative ingredients for a specified ingredient (ingredient).

Operation: It searches the alternatives_db database using the input ingredient as a key to find a list of substitutes.

Return Value: Returns a list of possible substitutes for the ingredient. If there are no alternatives available, it returns an empty list.

create_shopping_list function)

Functionality: This function creates a shopping list for ingredients that are needed for a selected recipe (selected_recipe) but are not currently possessed by the user.

Operation: It iterates through the list of ingredients for the selected recipe and adds those that the user does not have to the shopping list.

Return Value: Returns a list of ingredients that need to be purchased.

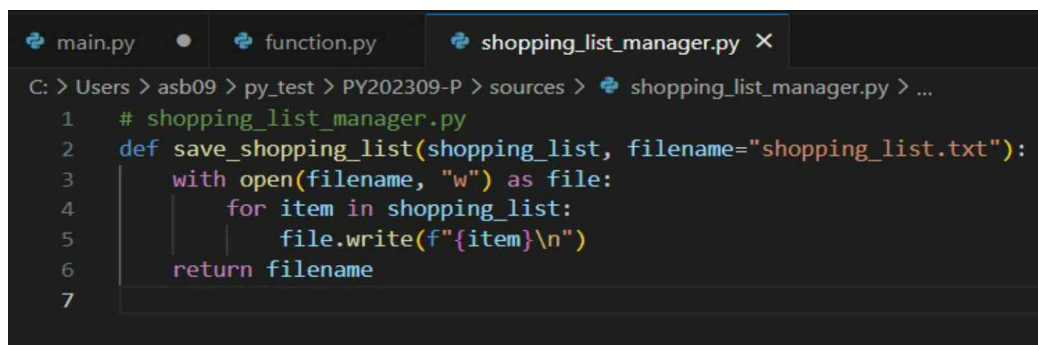
print_recipe_instructions function)

Functionality: Based on the name of the recipe chosen by the user (recipe_name), this function finds and prints out the instructions for that recipe from the recipes_db.

Operation: It iterates through the recipes_db to find a matching recipe by name and then prints out the recipe's instructions in order.

Return Value: None, as it directly prints the instructions to the console.

[shopping_list_manager.py]



```
main.py • function.py shopping_list_manager.py X
C: > Users > asb09 > py_test > PY202309-P > sources > shopping_list_manager.py > ...
1 # shopping_list_manager.py
2 def save_shopping_list(shopping_list, filename="shopping_list.txt"):
3     with open(filename, "w") as file:
4         for item in shopping_list:
5             file.write(f"{item}\n")
6     return filename
7
```

2) Test results

(1)Ingredient-Based Recipe Recommendation Function

Description:

This function takes the input of ingredients from the user and recommends possible

recipes from the `recipe_db` based on the entered ingredients.

Alternative Ingredient Suggestions: If some ingredients do not exactly match the requirements of the recipe, the system proposes alternative ingredients.

(2)Shopping List Generation-Manages the additional required ingredients as a shopping list

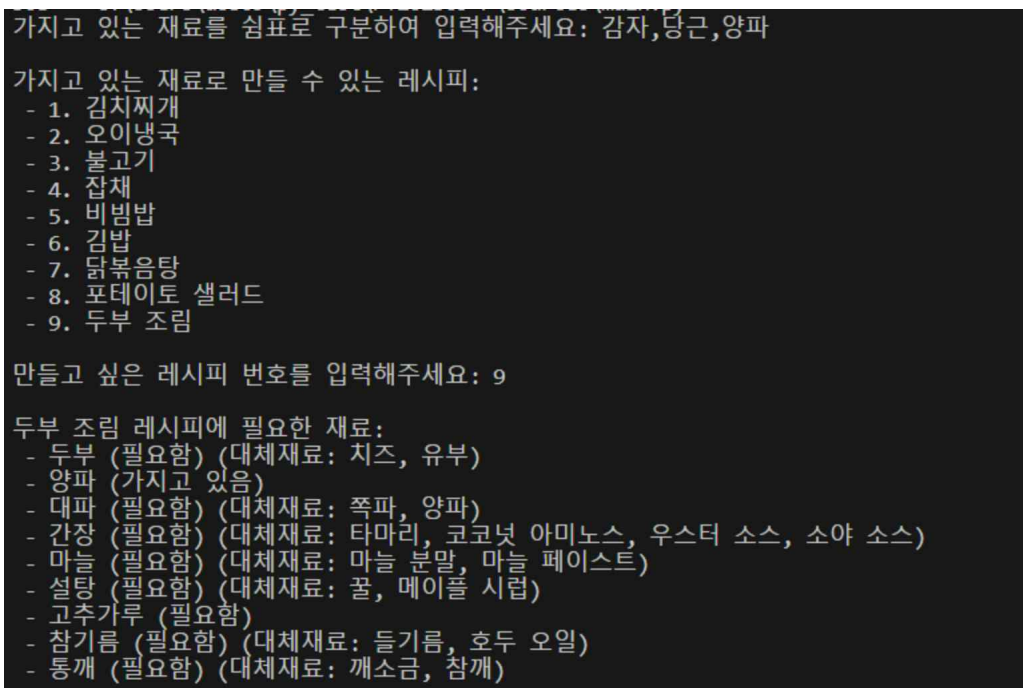
Description:

It identifies ingredients that the user does not have and asks whether to save the missing items as a shopping list.

+)

The recipe recommendation process has been simplified by assigning numbers to each recipe, allowing users to make selections using numerical input rather than entering names.

[Test results screenshot]



가지고 있는 재료를 심bolo로 구분하여 입력해주세요: 감자,당근,양파

가지고 있는 재료로 만들 수 있는 레시피:

- 1. 김치찌개
- 2. 오이냉국
- 3. 불고기
- 4. 잡채
- 5. 비빔밥
- 6. 김밥
- 7. 닭볶음탕
- 8. 포테이토 샐러드
- 9. 두부 조림

만들고 싶은 레시피 번호를 입력해주세요: 9

두부 조림 레시피에 필요한 재료:

- 두부 (필요함) (대체재료: 치즈, 유부)
- 양파 (가지고 있음)
- 대파 (필요함) (대체재료: 쪽파, 양파)
- 간장 (필요함) (대체재료: 타마리, 코코넛 아미노스, 우스터 소스, 소야 소스)
- 마늘 (필요함) (대체재료: 마늘 분말, 마늘 페이스트)
- 설탕 (필요함) (대체재료: 꿀, 메이플 시럽)
- 고춧가루 (필요함)
- 참기름 (필요함) (대체재료: 들기름, 호두 오일)
- 통깨 (필요함) (대체재료: 깨소금, 참깨)

```

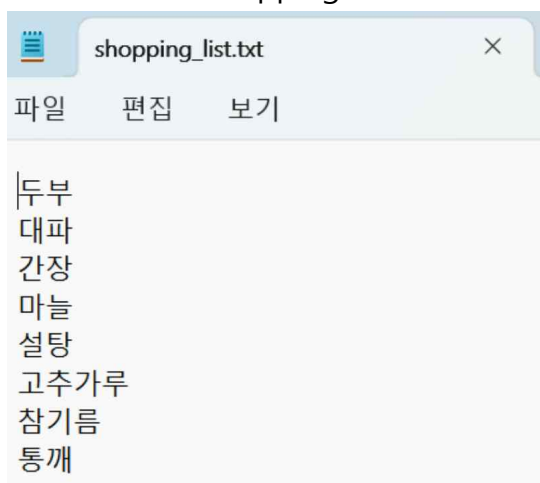
쇼핑 리스트:
- 두부
- 대파
- 간장
- 마늘
- 설탕
- 고추가루
- 참기름
- 통깨
쇼핑 리스트를 저장하시겠습니까? (yes/no): yes
쇼핑 리스트가 shopping_list.txt에 저장되었습니다.

두부 조리 레시피 지침:
두부는 물기를 제거한 후 적당한 크기로 썰어줍니다.
양파와 대파는 채썰고, 마늘은 다져줍니다.
팬에 참기름을 두르고 양념 재료들을 넣어 볶다가 두부를 넣고 조리합니다.
양념이 두부에 배면 완성입니다. 위에 통깨를 뿌려서 서빙합니다.

(base) c:\Users\asb09\py test\PY202309-P\sources>

```

When users input the ingredients they have, the system presents recipes that can be made with those ingredients. Upon entering the desired recipe number, the system informs the user of the necessary ingredients and compiles a shopping list for any missing items, then inquires if the user wishes to save this list. If saved, it is stored in the shopping list.



4. Changes in Comparison to the Plan

1) Change title

Before:

Ingredient Optimization: The program assists in the optimal utilization of refrigerator contents. By suggesting recipes that prioritize ingredients nearing their expiration, it helps reduce food waste and maintain the freshness of groceries.

After:

Even if not by the exact date of manufacture, assign an order to indicate which items should be consumed soon.

Reason:

I initially wanted to differentiate my program by including an ingredient optimization

feature that would assign expiration dates to each grocery item and suggest recipes based on items nearing expiry. However, my current skill level is not sufficient to develop such a program. Instead, I'd like to focus on refining the other features to ensure they are programmed with higher quality. I want to address potential issues that may arise with these other functionalities. Additionally, I aim to review what I've learned in class and use that knowledge as a foundation to develop a more cohesive and complete codebase.

5.Schedule

