Leftover Recipe Generator: Version 1 Review

Overview

The Leftover Recipe Generator is a Python-based app that helps users turn leftover ingredients into new meals. It connects to the Spoonacular API to find recipes that best match what users already have at home, reducing food waste and saving time. Users input ingredients and the program returns recipes with ingredient matches, substitution suggestions, and Instacart links for missing items.

Current Implementation

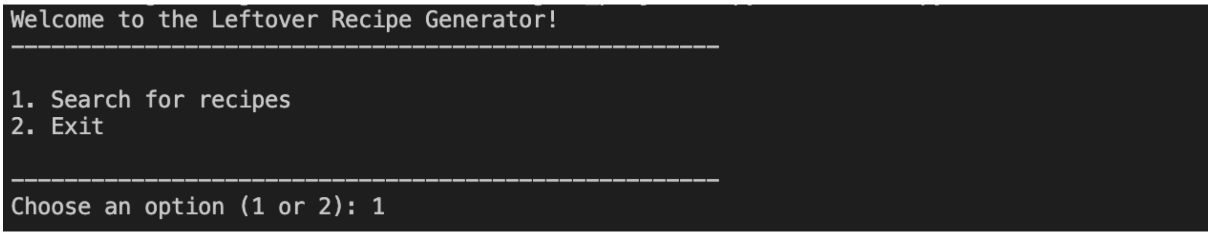
The current version runs in a command-line interface (CLI). It asks users to:

1. Enter a list of ingredients
2. Optionally specify a cuisine type
3. Choose how many recipe results to display.

The app then calls the Spoonacular API, ranks results by ingredient match, and displays:

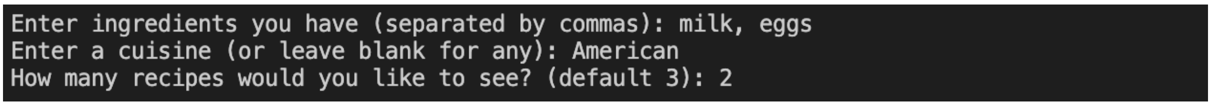
- Recipe titles and IDs
- Used and missing ingredients
- Instacart links for missing ingredients
- Suggested substitutes (from the Spoonacular substitutes endpoint)

Demonstration



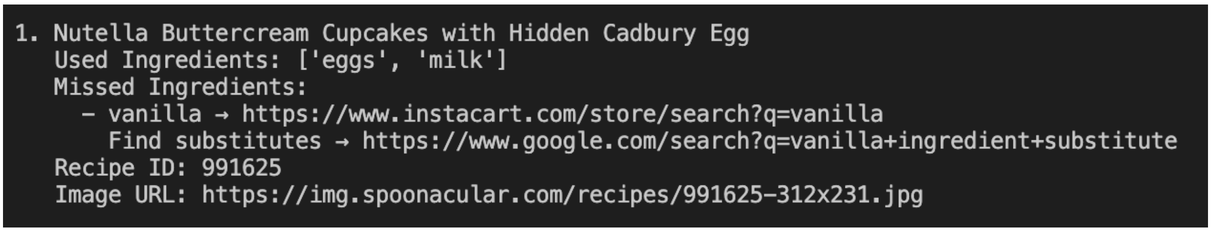
```
Welcome to the Leftover Recipe Generator!
-----------------------------------------------------

1. Search for recipes
2. Exit

-----------------------------------------------------
Choose an option (1 or 2): 1
```

The demo begins with the main menu, which welcomes the user and presents two options: to search for recipes or to exit the application.



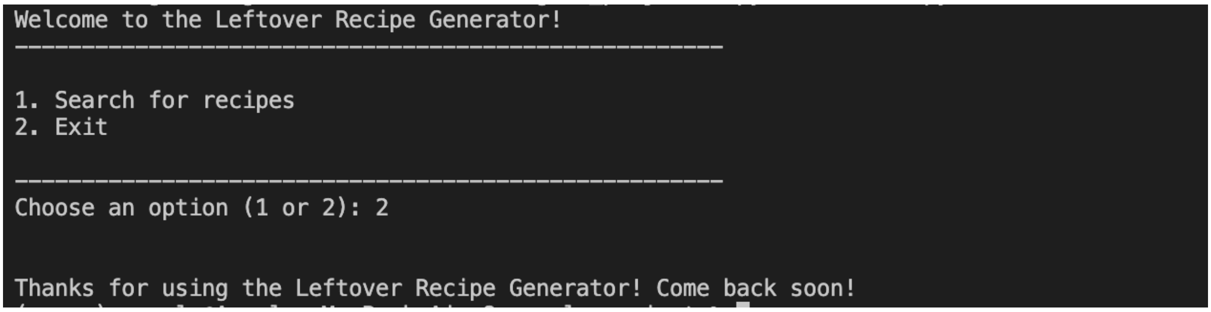
```
Enter ingredients you have (separated by commas): milk, eggs
Enter a cuisine (or leave blank for any): American
How many recipes would you like to see? (default 3): 2
```

Selecting the search option brings up a prompt asking the user to enter the ingredients they have on hand, separated by commas. For example, “milk, eggs.” Next, the app optionally asks the user to specify a cuisine, allowing them to filter results based on preferences, though this can be left blank if they want any cuisine. The user is then prompted to choose the number of recipe results to display, with a default of three, giving them control over how many options they see. \



```
1. Nutella Buttercream Cupcakes with Hidden Cadbury Egg
   Used Ingredients: ['eggs', 'milk']
   Missed Ingredients:
     - vanilla → https://www.instacart.com/store/search?q=vanilla
       Find substitutes → https://www.google.com/search?q=vanilla+ingredient+substitute
   Recipe ID: 991625
   Image URL: https://img.spoonacular.com/recipes/991625-312x231.jpg
```

After entering this information, the app processes the input and displays the recipe outputs. Each recipe includes the title, ingredients that are already available, and missing ingredients. For missing ingredients, the app now provides Instacart links for quick purchasing as well as substitution links either from Spoonacular or a Google search for alternatives. Additionally, each recipe shows the recipe ID and an image URL.



```
Welcome to the Leftover Recipe Generator!
-----------------------------------------------------

1. Search for recipes
2. Exit

-----------------------------------------------------
Choose an option (1 or 2): 2


Thanks for using the Leftover Recipe Generator! Come back soon!
```

If the user selects option **2** from the main menu, the application immediately ends the session. A friendly message is displayed thanking the user for using the Leftover Recipe Generator and encouraging them to come back later. After this message, the program exits, stopping any further input or API calls. This ensures that users have a clear and predictable way to leave the app without errors or confusion.

Issues & Challenges

- Instacart Links: I'm improving how missing ingredients link to Instacart searches since at times the URLs are not correct.
    - Proposed Solution: I need some more time to think about how I want to approach this, I might try a different strategy
- Multiple Searches: Currently, users must restart the program to perform a new search.
    - Proposed Solution: I plan to add a loop or main menu refresh to support continuous use.
- Output Formatting: The CLI layout works, but it becomes cluttered when many ingredients are entered.
    - Proposed Solution: Transitioning to a GUI would fix this.
- Available Cuisine Display: I plan to show the user a list of available cuisines based on the ingredients entered. This prevents selecting a cuisine that would return no results and improves usability.

- Proposed Solution: After fetching recipes for the entered ingredients, I will generate a list of cuisines that actually have matching recipes. This list will be displayed to the user before they choose a cuisine, reducing the chance of empty results.

Milestones for the Next Weeks

1. Refine API & Search Logic: Clean up Instacart link formatting to handle special characters, improve error handling for failed API requests, and ensure missing ingredient substitutions display correctly. This will make the core search experience reliable.

2. Add Multiple Search Support: Modify the CLI so users can perform multiple searches in a single session without restarting the program. Include an updated main menu that allows repeated searches and optional cuisine selection from available results.

3. Display Available Cuisines: After analyzing recipes for the entered ingredients, generate and show a list of cuisines that have matching recipes. This ensures users don't select cuisines with no results.
4. Add Dietary Restrictions & Cooking Time Filters: Allow users to specify dietary preferences and maximum cooking time when searching for recipes. These filters will be applied in the API query and reflected in the displayed results.

5. Data Persistence (Optional / Time Permitting): Implement local storage for bookmarking recipes (JSON or lightweight database). Bookmarks will be available if time allows, letting users revisit previously found recipes.

6. React Web Interface: Build a full React-based GUI that interacts with the Python backend. Include interactive recipe cards, filters, recipe images, clickable Instacart links, and optional bookmarking. This is the main long-term milestone.

Self-Reflection

I'm satisfied with my current progress. The core functionality works as intended, and I'm now focusing on improving usability. My progress has been a bit slower than expected, but I plan to dedicate more time to this project for the rest of the semester to stay on track.

Easier than expected: Integrating the Spoonacular API and parsing JSON responses.

Harder than expected: Handling edge cases (like missing ingredient data) and formatting clean output in the CLI.

I believe I can finish the project within the remaining weeks by completing the CLI improvements first and then developing the React interface as the next major milestone.