The Foodies

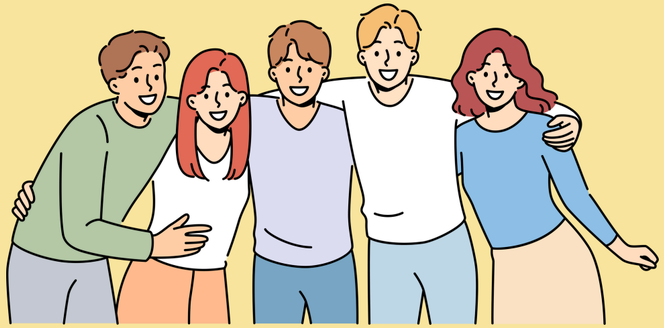
by Anouk de Brouwer
and Lisa Levasseur

Project Recap

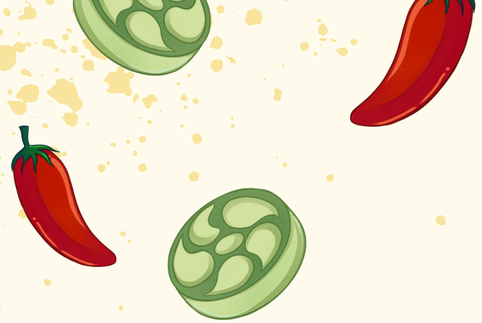
why Recipes ?

Everyone is eating food, everyday topic

Concrete applications



Dein Abschnittstext



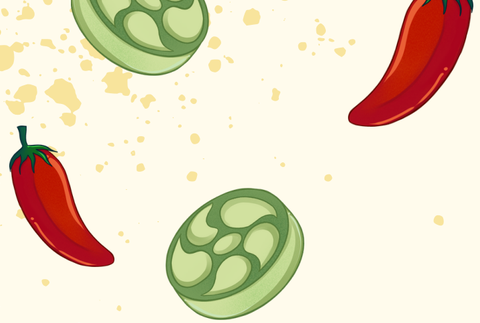
Project Recap

Dataset

Kaggle → Food.com

200 000 recipes

Dein Abschnittstext



Project Recap

Graph Structure

Nodes

Recipes

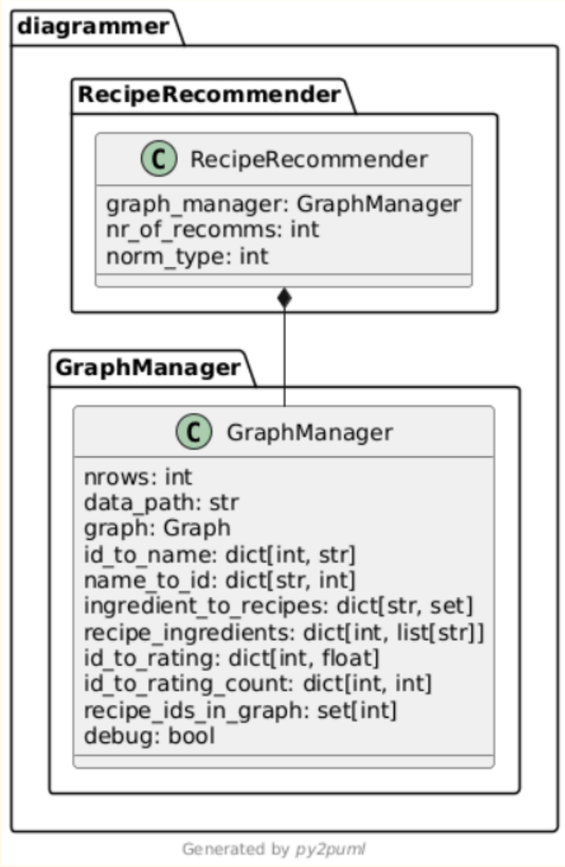
Edges

Recipes sharing same ingredients

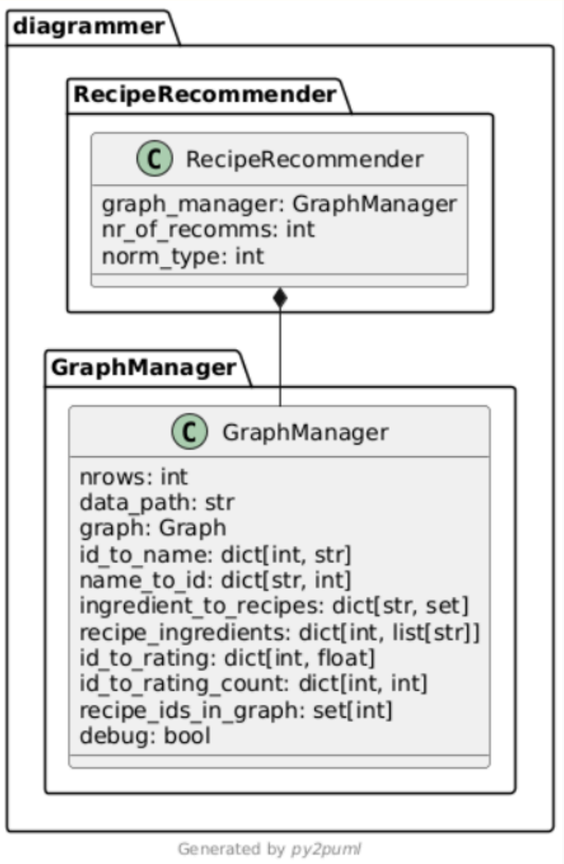
Weights

of shared ingredients

First Interactive Script

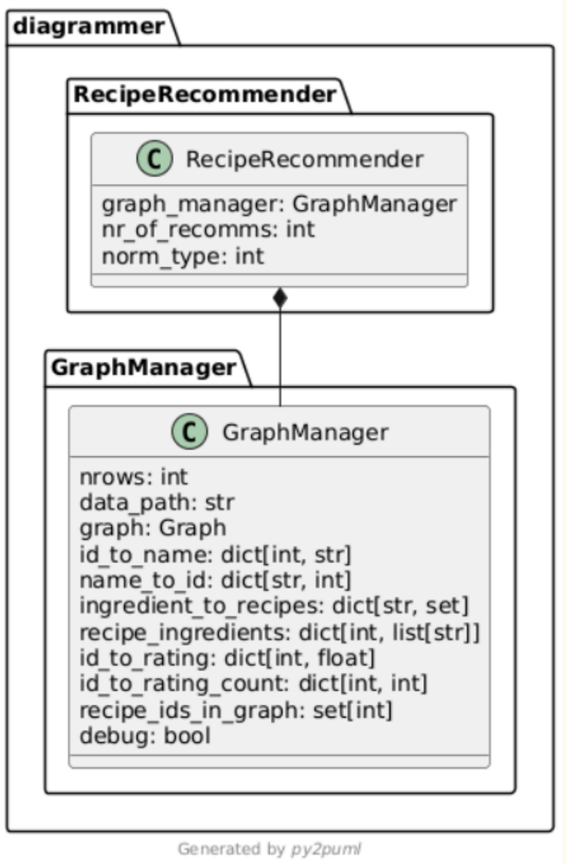


First Interactive Script



← number of recommendations
← normalization type

First Interactive Script



← number of recommendations
← normalization type

```python
def calculate_similarity_score(self, normalization_type, neighbor, recipe_id):
    weight = self.graph[recipe_id][neighbor]['weight']

    match normalization_type:
        case 0:  # weight -> nr of shared ingredients
            return weight

        case 1:  # normalized by total nr of ingredients
            neighbor_ingredients = len(self.get_recipe_ingredients(neighbor))
            return weight / neighbor_ingredients

        case 2:  # total nr of ingredients + normalization of rating
            neighbor_ingredients = len(self.get_recipe_ingredients(neighbor))
            similarity = weight / neighbor_ingredients
            rating = self.get_avg_recipe_rating(neighbor)
            return similarity + (rating / 5.0)  # /5 is normalization for rating

        case _:
            return weight
```

First Interactive Script

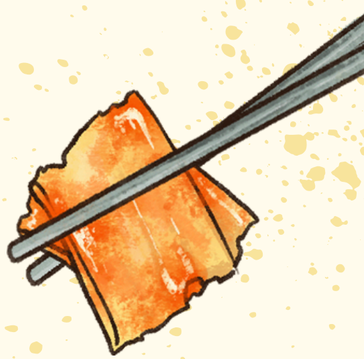
0. $shared\ ingredients$

1. $$\frac{shared\ ingredients}{all\ ingredients}$$

2. $$\frac{shared\ ingredients}{all\ ingredients} + \frac{rating}{5}$$

```python
def calculate_similarity_score(self, normalization_type, neighbor, recipe_id):
    weight = self.graph[recipe_id][neighbor]['weight']

    match normalization_type:
        case 0:  # weight -> nr of shared ingredients
            return weight

        case 1:  # normalized by total nr of ingredients
            neighbor_ingredients = len(self.get_recipe_ingredients(neighbor))
            return weight / neighbor_ingredients

        case 2:  # total nr of ingredients + normalization of rating
            neighbor_ingredients = len(self.get_recipe_ingredients(neighbor))
            similarity = weight / neighbor_ingredients
            rating = self.get_avg_recipe_rating(neighbor)
            return similarity + (rating / 5.0)  # /5 is normalization for rating

        case _:
            return weight
```

First Interactive Script



Which recipe do you want to replace today? (Press 'Enter' to confirm or 'Escape' to cancel)

First Interactive Script

```
Found recipe: almond cow

Ingredients (4):
amaretto, ice, kahlua, milk

Top 3 Similar Recipes:
--------------------------------------------------------------------------------
 1. almond latt
    Number of shared ingredients: 3
    Shared ingredients: amaretto, kahlua, milk
    Rating: 4.0/5.0
    Total ingredients: 4

 2. adult chocolate milk
    Number of shared ingredients: 3
    Shared ingredients: ice, kahlua, milk
    Rating: 5.0/5.0
    Total ingredients: 7

 3. amaretto iced coffee
    Number of shared ingredients: 3
    Shared ingredients: amaretto, ice, milk
    Rating: 5.0/5.0
    Total ingredients: 8
```

```
neighbors.append({
    'id': neighbor,
    'name': recipe_name,
    'similarity_score': similarity,
    'ingredients': ingredients,
    'shared_ingredients': shared_ingredients,
    'rating': rating
})
```

```
neighbors.sort(key=lambda x: x['similarity_score'], reverse=True)
return neighbors[:top_k]
```

First Interactive Script

```
❓ Recipe 'almond' not found exactly. Did you mean:
1. Almond Tea
2. Almond Joy
3. Almond Dip
4. Almond Cow
5. Almond Tofu
```

```python
from difflib import get_close_matches
```

```python
# Exact match
if query_lower in self.name_to_id:
    return self.name_to_id[query_lower], query

# Fuzzy match
recipe_names = list(self.name_to_id.keys())
matches = get_close_matches(query_lower, recipe_names, n=max_suggestions)
```

UI library

Simple interface

Users could interact with our recipe recommender

Suggestions of similar recipes

UI library

Comparison

Tkinter ?

Included by default in Python
Basic
Interfaces often look outdated
Simple layouts require more code

UI library

Comparison

Dash ?

Data visualization

Powerful tool

Callbacks

UI library

Comparison

Streamlit !

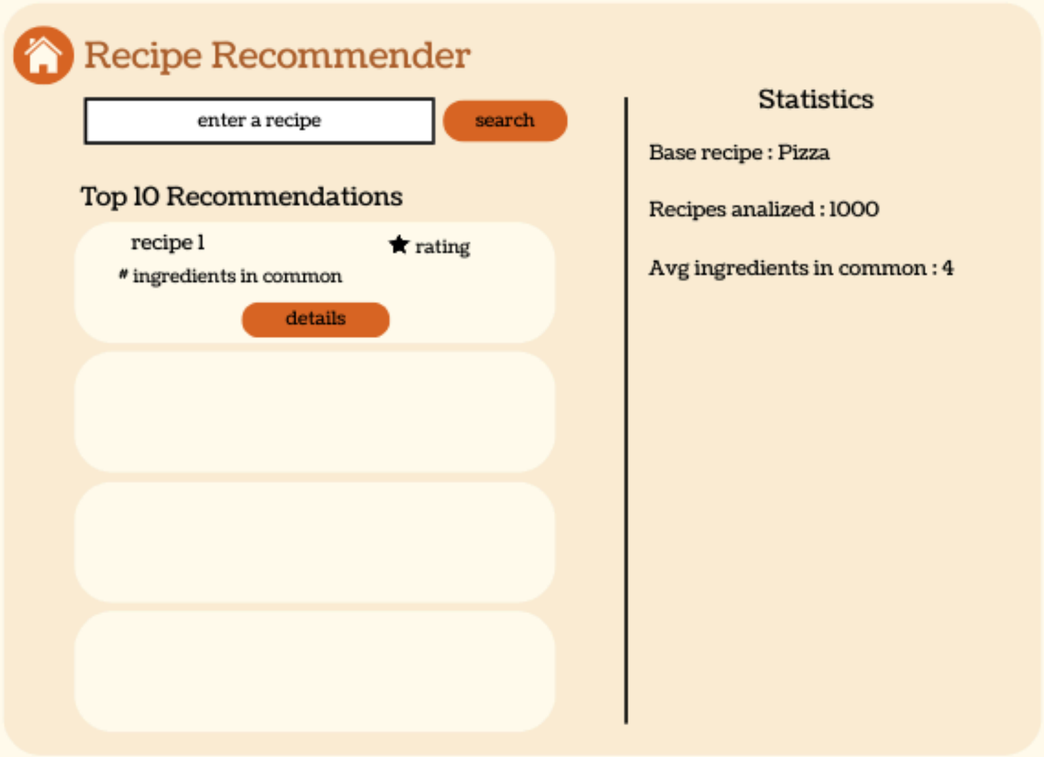
Very simple code

Interactive visualizations

UI library

Idea

Home page

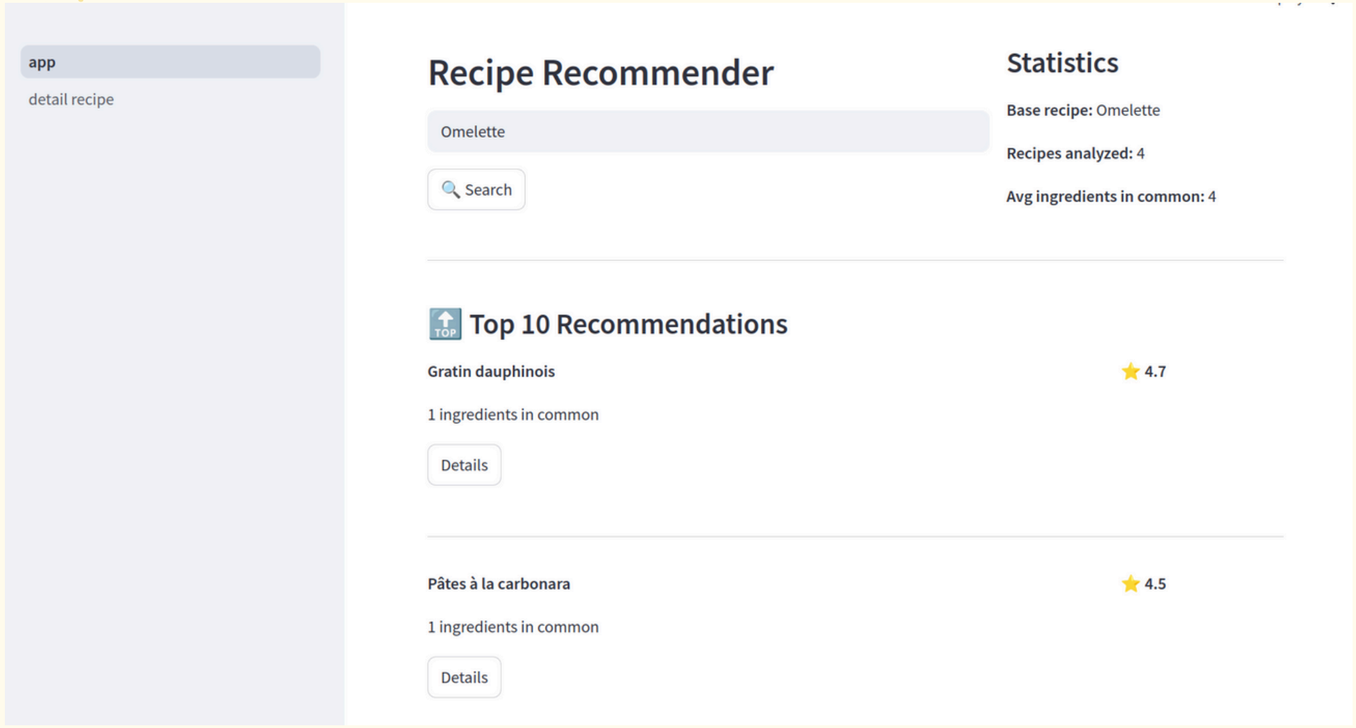


Recipe page

UI library

Mockup

Home page



Recipe page

Next schedule

Project Planning

May First Half

Preparation

- Brainstorming
- Graph

May Second Half

Simple Implementation

- Recommending logic
- Command Line
- UI Mockup

June First Half

Final Implementation

- Final UI
- Additional Logic

Distribution of tasks

References

Kaggle Dataset https://www.kaggle.com/datasets/shuyangli94/food-com-recipes-and-user-interactions/data?select=PP_recipes.csv

ChatGPT : https://chatgpt.com/

https://docs.python.org/3/library/tkinter.html
https://streamlit.io/
https://dash.plotly.com/
https://docs.python.org/3/library/difflib.html

Thank you for your attention