CIT 594 Project Documentation: CocktailPedia

Team Algoholic

Team Members

Name	Email
Ruichen Zhang	ruichenz@seas.upenn.edu
Rui Zhang	ruizhan@seas.upenn.edu
Wenrui Zhang	wrzhang@sas.upenn.edu
Yuxin Hu	yuxinhu@seas.upenn.edu

Introduction

CocktailPedia is a Java-based cocktail recommender system. It uses cocktail datasets on the web as the data source to provide recipe lookup and recommendations for users. It also allows users to customize their own recipes, and export them in a text file format.

Project Features

The project provides the following features:

1. Recipe lookup
A user can request the recipe of a cocktail by the cocktail name. The recipe contains information such as drink name, category, glassware, ingredients, taste and preparation instructions.

2. Recommendation
If a user is new or does not have a specific preference, the user will be recommended one of the classic recipes by default.

If a user has a taste preference, the user will be recommended one of the cocktail recipes with the user's favorite taste.

After a recipe is recommended to a user, we increase the popularity point of the recipe, so that the most popular cocktails will be recommended first instead of randomly chosen.

Based on these basic building blocks of recommendations by a single feature, we will design a more complex recommendation algorithm based on multiple factors, such as both popularity and user preference.

3. Recipe customization

The user can select from a few customization options to get a customized recipe in a text file format. The options include ingredients, preparation style and taste.

The features mentioned above are achieved via interactions with the user in the console. After the program is manually run by a user, it will accept user input, execute corresponding functions, and print the recipes and helper information in the console based on different user options. Users can also locate the recipe files they created under the directory of their username.

Languages and Data Structures

This project is Java-based, but we will use very limited Python to assist us in cleaning the dataset. Our project will involve data structures such as maps, arrays, heaps. For example, we will use a hash table to store cocktail recipes imported from the dataset, and a sorted map to keep track of the popularity of cocktail recipes.

Interface Design

Constants

CLASSIC
Constant: public static final List<String> CLASSIC = Arrays.asList   ("Old Fashioned", "Negroni", "Daiquiri", "Dry Martini", "Whiskey Sour"); **Description:** A default list of classic cocktails.

DEFAULT_RECOMMENDATION_LIMIT
Constant: public static final int DEFAULT_RECOMMENDATION_LIMIT = 1; **Description:** A default number of recommendation limit.

Abstract Methods

loadDataset

Method:
public Map<String, Cocktail> loadDataset(String path);

Description:
Load the csv format cocktail dataset, and store the cocktail recipes using a map.

Parameter:
path: file path of the dataset.

Returns:
A map storing cocktails recipes, where the key is the name of the drink, the value is the recipe of the cocktail.

---

initializePopularity

Method:
public Map<String, Integer> initializePopularity(Map<String, Cocktail> recipeMap);

Description:
Use the number of queries of the drink to represent popularity. Initialize the popularity map of all drinks with 0s.

Parameter:
recipeMap: the map storing cocktails recipes.

Returns:
A sorted map storing cocktail popularity, where the key is the name of the drink, the value is the number of queries of the drink (0 at first).

---

buildIndexByPreference

Method:
public Map<String, String> buildIndexByPreference(Map<String, Cocktail> recipeMap);

Description:
Group cocktail drinks by taste preference, and store into a new preference map.

Parameter:
recipeMap: the map storing cocktails recipes.

Returns:
A map storing different categories of cocktail tastes, where the key is the taste, the value is the list of cocktails of that taste.

---

queryByDrink

Method:
public Cocktail queryByDrink(String drink,
                            Map<String, Cocktail> recipeMap,
                            Map<String, Integer> popularityMap);

Description:
Return the recipe of the drink name the user queries for, and update the query count of this drink in the popularity map.

Parameter:
drink: the name of the drink.
recipeMap: the map storing cocktails recipes.
popularityMap: the map storing the popularity of cocktails.

Returns:
The recipe for the cocktail.

---

recommendByClassic

Method:
public String recommendByClassic();

Description:
Recommend classic cocktails to users, and return the name of the recommended drink.

Parameter:
None.

Returns:
The name of the drink to recommend.

---

recommendByPopularity

Method:
public String recommendByPopularity(Map<String, Integer> popularityMap);

Description:
Recommend cocktails with top query counts in the popularity map to users, and return the name of the recommended drink.

Parameter:
popularityMap: the map storing the popularity of cocktails.

Returns:
The name of the drink to recommend.

---

recommendByPreference

Method:
public String recommendByPreference(String taste,
                                      Map<String, String> preferenceMap);

Description:
Recommend cocktails with the users' preferred taste in the preference map to users, and return the name of the recommended drink.

Parameter:
taste: the users' preferred taste.
preferenceMap: the map storing different categories of cocktail tastes.

Returns:
The name of the drink to recommend.

---

recommend

Method:
public Cocktail recommend(String taste,
                        Map<String, String> preferenceMap,
                        Map<String, Integer> popularityMap,
                        Map<String, Cocktail> recipeMap);

Description:
Return the recommended cocktail recipe to users based on users' options:
   Option 1: Recommend by classic
   Option 2: Recommend by popularity
   Option 3: Recommend by preference
   Option 4: Recommend by preference and popularity

Parameter:
taste: the users' preferred taste.
preferenceMap: the map storing different categories of cocktail tastes.
popularityMap: the map storing the popularity of cocktails.
recipeMap: the map storing cocktails recipes.

Returns:
The cocktail recipe to recommend.

---

customizeRecipe

Method:
public void customizeRecipe(String username,
                           Map<String, Cocktail> recipeMap);

Description:
Allow a user to create a customized cocktail recipe. The user can choose from a list of options we provide, including type of the liquor/spirit, taste, preparation style. Then we assemble the recipe based on the choices made by the user, generate a txt file under the directory named by the username, and store it into a user map.

Parameter:
username: name of the user.
recipeMap: the map storing cocktails recipes.

Returns:
None.

Project Management

Milestones

Milestone	Description	Timeline
M1	Project scope, feature and interface design	4/17 - 5/4
M2	Feature implementation and testing	5/4 - 5/8
M3	Project demo	5/9

Story Points

Feature	Story Point	Assignee
Project design & documentation	4	Ruichen
Data cleaning	2	Ruichen
Load dataset and support user queries	2	Yuxin
Build popularity map	1	Yuxin
Build index preference	1	Yuxin
Recommend by a single feature	3	Rui
Recommend by multiple features	2	Rui

Customize recipe	2	Wenrui
Recommender main class (boot up)	3	Wenrui
Testing	4	All
Video demo	4	All

References

1. https://www.kaggle.com/datasets/ai-first/cocktail-ingredients