Automatic Meal Planner

Callista Christ, Katya Gozman, Melissa Merz, Dylan Sukay

Project Description and Goals:

- When you're a busy college student living off campus, trying to figure out what meals to make every week can be a real hassle. That's why we were inspired to create a website where a user can get recommendations for weekly meals within their price range and get an automatic grocery list of items to buy next time they go shopping for those recipes.
  - Users create an account, input their zip code, specified weekly price range, allergies and meal preferences (vegan? kosher?), disliked foods, favored cuisines, and what kinds of meals they want recommendations for (breakfast? lunch?). In the beginning, a user will also be shown a series of random meals that we'll have them rate to see what kinds of meals they favor and possibly use a machine learning method to recommend meals.
  - We generate a list of recipes for them to make, along with ingredient lists that is exported to a text file for easy access and printing, and an estimated price.
  - If a user is not happy with one or more of the recipes, they can reject it and be offered a new one. The website will keep track of recipes a user rejects so that it knows not to offer them the recipe again.
  - Once a user makes a meal, they can rate it on a scale of 1-5. The higher they rate it, the more likely the system is to recommend the same meal to them again.

Data Sources:

- **www.allrecipes.com**
- **www.bonappetit.com** (maybe)
- We need to learn the format of HTML for at least one website

Task List:

- **Melissa**: Crawl all recipes on www.allrecipes.com and store url, name, ingredients, cuisine, image
  - Save in SQL database
  - Are images too large to store in SQL?

- **Dylan**: Scrape jewel-osco or hpp data for grocery prices to add to database to estimate the weekly cost of the recipes. [optional: have a nation-wide application using walmart/whole-foods amazon. Cheap way of doing this would be factoring chicago prices by COL difference between zip codes.]

- Katya and **Callista**: [optional: machine learning component that learns what kinds of food the user likes based on how they rate past recipes, and recommends recipes based on their previous preferences]
- Katya and **Callista**: User data:
  - Zip code
  - Budget (cap?)
  - Food restrictions (allergies, vegan, vegetarian, kosher, halal, ingredients they don't like)
    - Possible allergies: vegetarian, vegan, gluten free, pescatarian, halal, kosher, lactose-intolerant, nuts,
  - Rank cuisines
  - Rank recipes they've already eaten (if we don't do machine learning, make this affect how often they see that meal again).
  - blacklisted recipes
  - average amount of time they are willing to spend per meal in an avg. week, can adjust at the start of each week by scoring busy-ness. (Some recipes indicate active time vs. waiting time, how do we want to deal with that? Account for meal-prepping?)
  - Foods they have so we can use those up. Deduce what they still have and ask if they bought anything else at the start of the following weeks.
  - How do we record "time", i.e. when do we know the week is up and the user needs new recipes?
  - How do we allow the instructors to generate several "weeks" in a sitting?
- All GUI:
  - How the user will input information and interact with our code
  - They will have an option to deselect randomly chosen recipes
  - They can rate the recipes from the previous week (which can indicate whether the recipes will show up again, either through machine learning or a > 3 star threshold
  - Outputs:
    - A text file of ingredients
      - Sorted on refrigerated and unrefrigerated
    - Some type of "calendar" that displays the link for the meal, possibly an image of that meal
    - Deselect button to remove a recipe

Timeline:

- **Week 4-6:** All web scraping (recipes and groceries) and database (recipes, groceries, and user data) tasks.
- **Week 6-8:** GUI.

- **Week 6+:** Machine learning, time allowing.