**Group 1: Generating Nutrition Breakdown into (Macro/Micro nutrients)**

*Team Members: Jason, Maria*

## PROBLEM STATEMENT

When a consumer wants to keep track of their diet and nutrition when eating at a local restaurant, it is difficult and time-consuming for consumers to find and read through a list of ingredients and attempt to extract the macro/micro nutrients of each ingredient given it's measurements.

## MAIN GOAL/REQUIREMENTS

Our job is to automate the process of extracting the macro/micro nutrient information from restaurant foods given a list of ingredients and their measurements (pulled from the Nutripair database) to provide a human-readable nutrition breakdown for the consumer.

## ASSUMPTIONS

- We have access to a different restaurant databases and the ingredient lists (and measurements) for food items on their menu
- We're only grabbing data for restaurants we onboarded.
    - Future restaurants to be added if they onboard with us :).
- We are given three inputs:
    - Restaurant name
    - Food name
    - HashMap/JSON with (ingredient : measurement) mappings
- Only valid restaurants, foods, ingredients and measurements are being passed into the function ( NO restaurants that don't exist, foods that don't exist, ingredients that don't exist, negative measurements, etc)
- We have a reliable food API (USDA's FoodData Central Api)

- We have a way of differentiating different variation of similar ingredients (ex: “Salted Butter” vs “Unsalted Butter”)

## ROADMAP TO ACHIEVE GOAL/TECHNICAL DESIGN

def findIngredientsList(**string** restaurantName, **string** foodName)
Given the restaurant name and food item, we will query the ingredients list and measurements for that specific food item (using AWS OpenSeach/AWS DynamoDB)

1. Write a function that would take in two inputs:
    a. Name of restaurant
    b. Name of food item
2. Query the database (whatever we are using to extract JSON)
3. Parse and Construct on ingredients array, as well as an array for it’s measurements (we could also use a hashmap to map ingredient->amount)
4. Return a HashMap
    a. Key : ingredient
    b. Value : measurement

def findNutritionBreakDown(**HashMap** ingredientsMap)
1. Write a function that would take in one input:
    a. HashMap of ingredients -> measurements
        i. ingredientsMap
        ii. ingredientsMap[ingredient] (ex: the value for ingredients[“egg”] is the measurement for “egg” in (grams,ounces)?)
2. Iterate through the list of ingredients, extracting its micro/macro nutrient information using the FoodData Central API
    a. We will first need the FDC ID of the food by hitting the '/v1/foods/search' endpoint (we can request 4 food info at a time).
        i. We might need to make our own database that caches the info of the food so that we would not have to waste API requests for future references.
    b. We can query up to 25 nutrients of upto 20 FDC IDs in one API call
        i. Each nutrients are identified by an ID so we need to know exactly which nutrients information we need and get there ID from the Nutrient schema
3. For every ingredient, we will multiply its nutrition breakdown by its measurements:
    a. Let's say ingredient[i] = “Butter”

                i. Ex: nutritionMap[“carbs”] += findNutrientInfo(ingredient[i]).carbs * measurements[i]
                ii. Ex: nutritionMap[“fats”] += findNutrientInfo(ingredient[i]).fats * measurements[i]

4. Build a HashMap with:
    a. Keys -> micro/macro nutrient (ex: carbs)
    b. Values -> the breakdown of that micro/macro nutrient (ex: 4g)
5. Return the HashMap (giving the nutrition breakdown for each micro/macro)
6. PUT into our AWS database/backend infrastructure
7. Verify that the database meets the expected outcome

## POSSIBLE ALTERNATIVE SOLUTIONS

1. Assuming that we are doing this for preprocessing (not at run-time), the Time Complexity of our function shouldn’t be a major issue (O(n^2))
2. Is there a way to merge HashMaps together? Instead of having to iterate through every nutrient for every ingredient, can we just merge the JSON/HashMap that the API call gives us, with our result JSON/HashMap

## LOGISTICS

Estimated completion time: 2 weeks