NYU Technical Report
Diet Optimizer: providing a nutritionally sound diet while catering to the user's desires

Constantin Dedeyan, Marianne Rubat-Ciagnus, Aina Khan, Dennis Shasha

August 24, 2018

1 Abstract

The website allows users or dietitians to enter personal information (age, height, weight, activity level) and dietary preferences (diet; intolerances; optimization of calories, time, protein, etc.) to create a unique profile with the user's basal metabolic rate (BMR) and suggested nutrient intakes. Each time the user would like a diet plan, they enter their goal for this plan (optimize certain nutrients, favorite cuisines, food types, time limits, etc). All of this information is taken into account when the site determines, if possible, the quantities of at least some of those desired foods that would meet the nutritional constraints. If no recipes are found, ready-made foods are suggested and the user can regenerate recipes. The net result is an efficient diet plan that is tailored specifically to the user. This is further achieved by allowing the user to save their favorite recipe requirements, recommending popular recipes, and allowing them to plan for multiple days by saving recipes.

2 Reviews of Existing Websites

Many nutrition websites rely on the public nutrient database from the United States Department of Agriculture (USDA), which has common food items and their nutritional content. Often the websites further integrate their own food data, which may be more brand-specific. Nutritiondata.self.com, further complements that by allowing users to create their own recipes. The website's database also includes its own nutritional indices such as a caloric ratio pyramid, which shows the distribution of calories among carbohydrates, proteins and fats, and an inflammation index which predicts the inflammatory effects of foods.

Among food logging sites in which users record the food they consume, CalorieCount.com calculates the overall calories from the food logged, and offers nutritional advice as milestones for users to achieve. By contrast, CalorieCamp makes food logging social. Users can comment on the food logs, encourage other users, and

applaud other users' achievements. Paying users in CalorieCount.com can get nutritional advice from registered dietitions based on their logged diet. Another similar site is Shopwell.com which focuses on the grocery shopping experience. The user creates his/her shopping list and the website recommends healthier food options based on the condition of the users. Shopwell.com includes a bar code scanning mobile application to retrieve information of that food products while shopping. Shopwell then provides real-time feedback for similar but healthier food products to the user.

These commercial websites generally inform users and provide soft guidance for better food selections. However, they do not enforce any nutritional constraints for the users. Nor do they suggest specific amounts of each food item that should be consumed. The available nutrition websites may be helpful for people whose nutritional recommendations are not very strict. However, medical patients may have strict nutritional needs. Currently, there are no tools for quantitative food recommendation according to nutritional constraints. This leads to the second part of the report.

3 Need for this Diet Planner Website

Currently, nutritionists make diet recommendations based on hand-calculations. This first involves informed selection of food items based on a patient's desires. Then, further refinement for food quantities is necessary to cater to the patient's specific nutritional needs. Lastly, as medical patients' conditions change, continual adjustment of the food recommendation is required. Manual calculations in such processes are slow, error-prone, and may be limited by the range of foods with which a dietitian is familiar.

Another issue is patients' preference. Diets planned for medical patients often do not sufficiently consider the patient's personal taste. It is not always the case that preferred foods are unhealthy ones. Relatively healthy foods that the patients like should be included in the diet, although the diet should still be complemented with necessary food items to satisfy the nutritional needs of the patients.

This website therefore aims to address the two problems of inefficiency and lack of preferences being accounted for, by providing an accurate method to plan diets according to nutritional needs while adequately catering to food tastes of medical patients.

4 User Interface

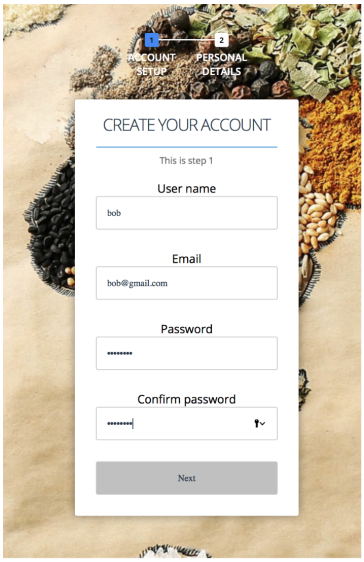
There are five major steps in the course of constructing a diet. First, the user fills in profile information as a new user or logs in to retrieve his or her profile. This information is necessary to create default nutritional constraints (carbohydrate, calorie, protein, and fat ranges), which are used in computing recipe options, unless the user wishes to alter them. In that case, secondly, the user (perhaps with the help of a dietitian) can manage his/her nutritional needs and objectives. Third, the user can browse foods and cuisines and select which of these preferences they would like to include or exclude in their diet plan. Fourth, the user gets the recommended diet plans, which can be saved for later viewing. Lastly, all recipes and foods can be accessed later and even be voted on. This

feedback helps with generating future recipes and recommending the most popular ones to all users. Also, the ability to access saved recipes at a later date is especially helpful for users who wish to diet plan for multiple days.

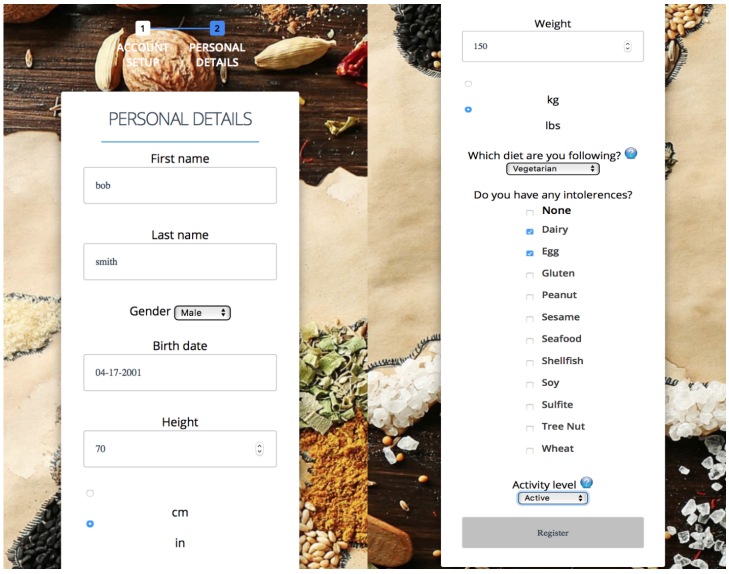
4.1 Fill in profile

There are two steps to creating an account, both of which include retrieving basic personal information from the user. This information is used for calculating nutritional constraints.

Step 1:



Step 2:



First, calorie needs are calculated by Harris Benedict Equation, which measures [1]:

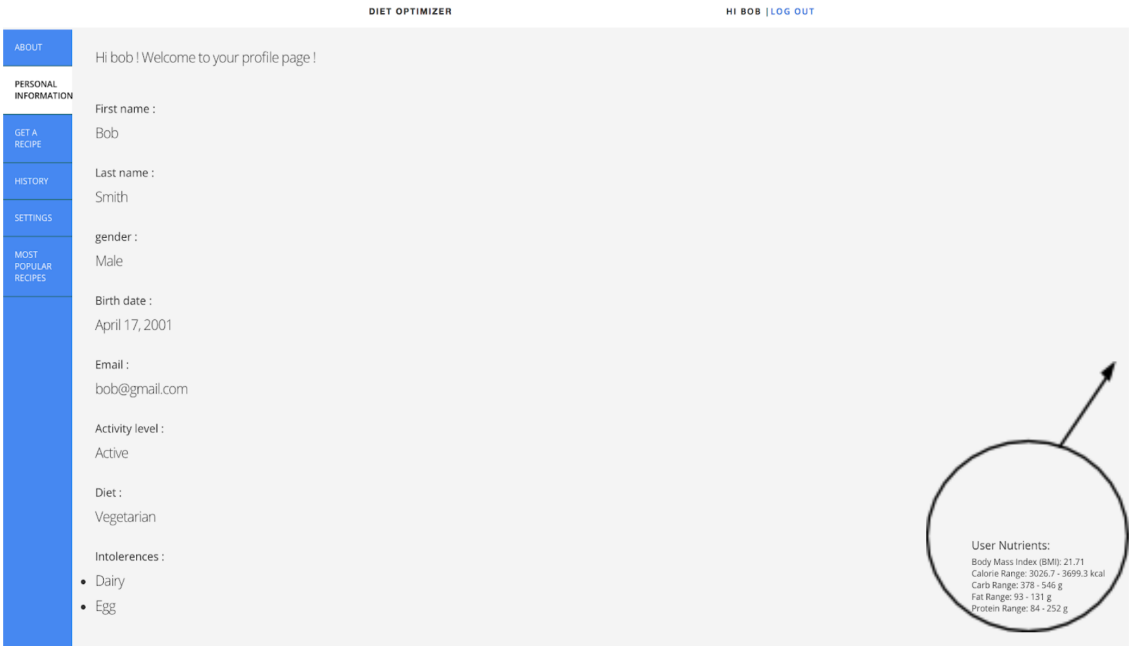
Women: BMR = 655 + (9.6 x Weight[kg]) + (1.8 x Height[cm]) - (4.7 x age in years)
Men: BMR = 66 + (13.7 x Weight[kg]) + (5 x Height[cm]) - (6.8 x age in years)

Second, this calculation and the user's activity level are considered, as default nutritional constraints are taken from dietary reference intakes report [2].

For macronutrients, the lower default limits are based on Recommended Dietary Allowances (RDAs). "RDA is the average daily dietary intake level; sufficient to meet the nutrient requirements of nearly all (97-98 percent) healthy individuals in a group." The upper default limits are based on the upper values of Acceptable Macronutrient Distribution Ranges.

For micronutrients, the lower default limits are based on Estimated Average Requirements (EARs). "EAR is the average daily nutrient intake level estimated to meet the requirements of half of the healthy individuals in a group." The upper limit default limits are based on Tolerable Upper Intake Levels, which shows "the highest level of daily nutrient intake that is likely to pose no risk of adverse health effects to almost all individuals in the general population."

Once the profile is created, the user can log in and access their home page, which will summarize their personal information and default nutritional ranges. An example can be seen below:



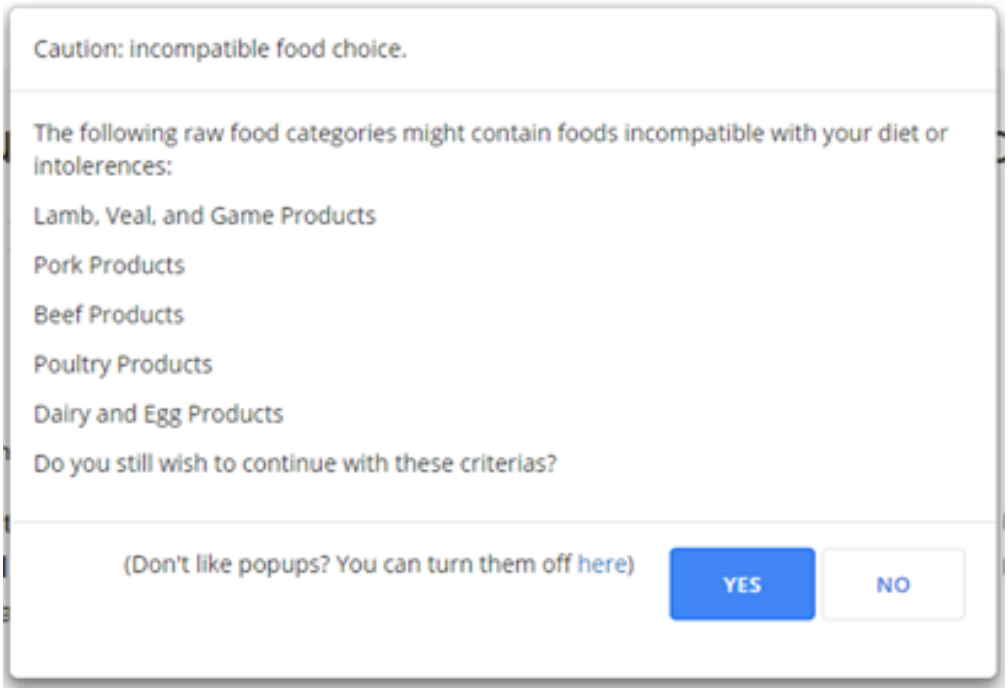
Default user nutritional ranges, calculated using the strategies seen above.

4.2 Manage Diet

When signing up, the site asks users what diet they follow and which intolerances they have. This key information shapes the types of recipes the website generates. At any time, the user can update these settings and try different diets.

The website heavily considers this information when managing the user's diet. With this notion in mind, if the user chooses a certain diet or intolerances and asks for conflicting foods (ex: a vegetarian asks for meat products), the website asks for a confirmation, in order to ensure that their diet is being evaluated properly.

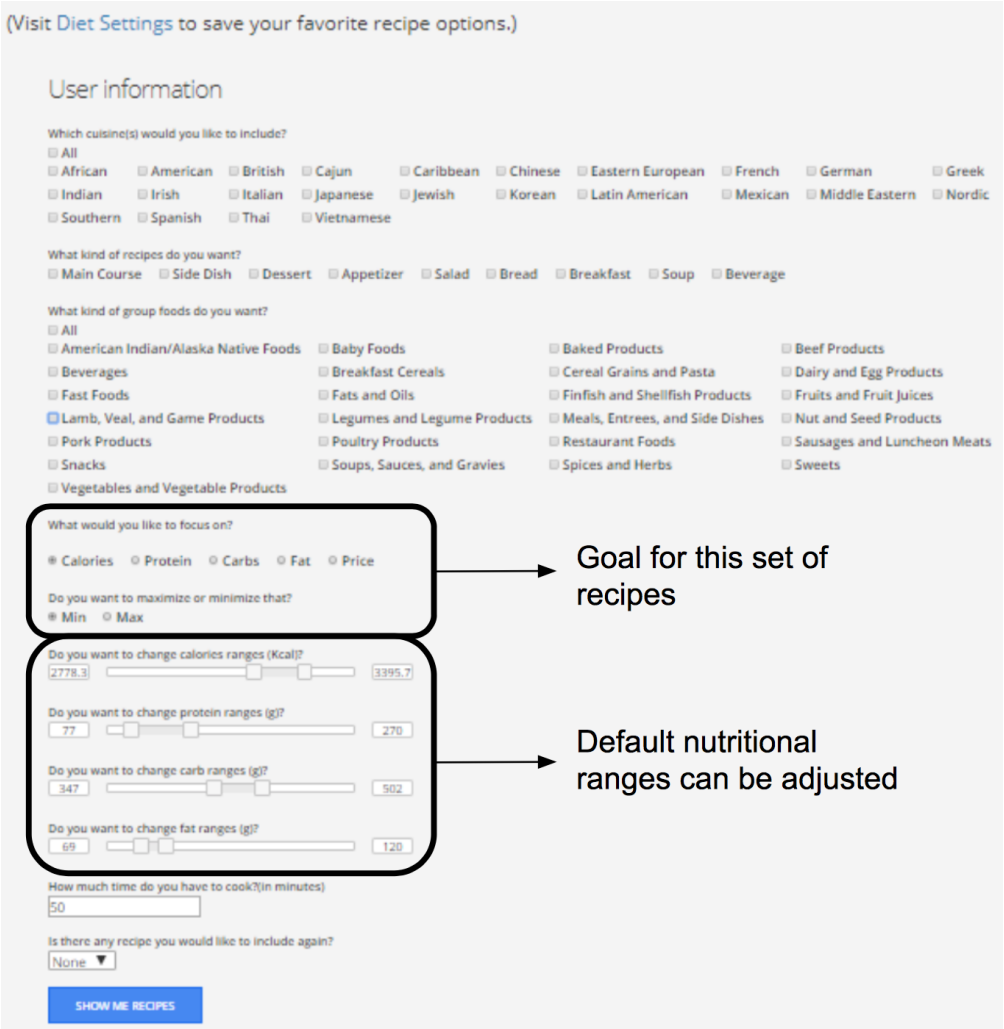
Below is an example of a "pop up". In this example, our user, Bob, is a vegetarian with egg and dairy intolerances. However, when asking for a recipe, he asked for meat, egg, and dairy products. This prompted the following message:



Additionally, when asking for recipes, the user can choose which nutrient to minimize or maximize (or the user can maximize/minimize price), as well as adjust nutrient ranges directly. This will generate a diet that is low or high in that chosen nutrient.

The website is made to adapt to the flexibility of the user, as the nutrient ranges can be altered from recipe to recipe. For example, the user may want a diet plan to focus on minimizing calories one day, and a plan that maximizes protein consumption the next day. Also, in order to keep results as accurate as possible, the user can update their personal information (age, height, weight, etc.) and diet preferences (diet, intolerances, favorite cuisines and foods) as frequently as they choose.

An example template that the user would use to request recipes is shown below. The diet managing tools discussed above are indicated (see next page).



4.3 Select Foods

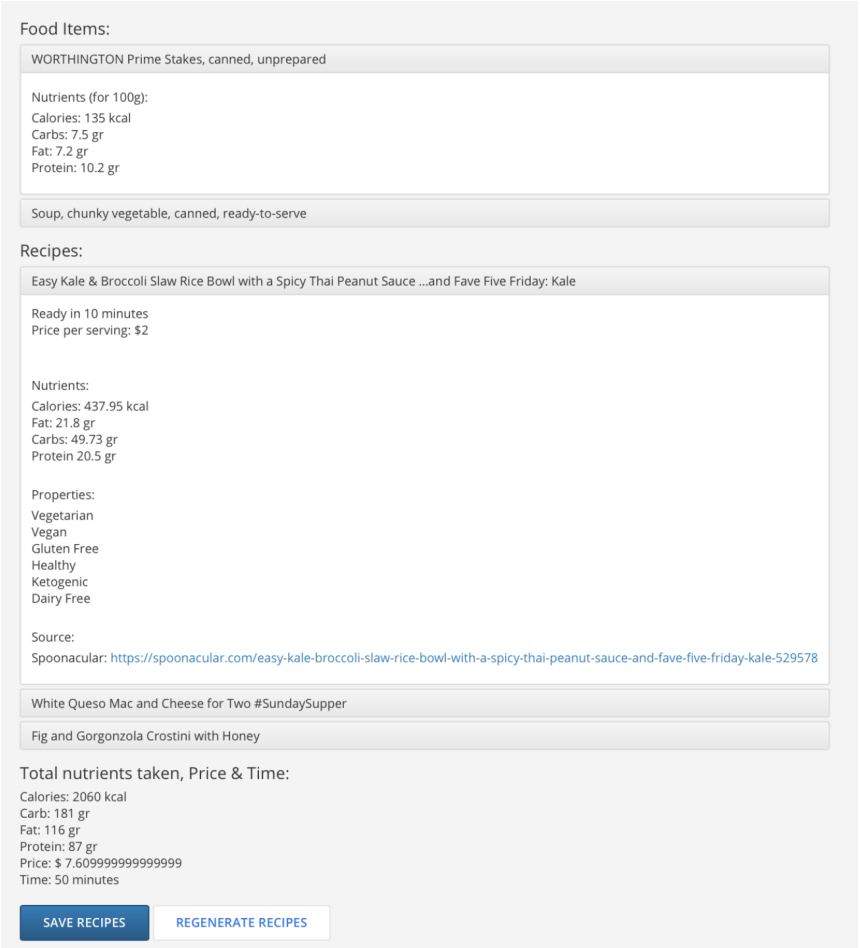
When asking for recipes, the user can choose between a variety of diverse cuisines, meal types (main dish, appetizer, beverage, etc.), and food groups (poultry, legumes, sweets, etc.). The website then randomly generates recipes using Spoonacular and foods using the USDA database.

Additionally, the website factors in the user's feedback. For example, a user can request to have a specific recipe they may have liked in the past in the recipe plan. The website will then include this recipe in the total nutrients and diet plan. Conversely, if the user dislikes a recipe, the website makes sure not to include it in future recipes.

4.4 Get Diet Plan

There are two possible outcomes for the results of requesting a diet plan. In both outcomes, ready-made foods will always be generated, and will include nutritional information for each individual food. The two results vary in the two possible recipe outcomes. If the user's constraints are too narrow and specific, the website will likely not find any recipes to satisfy the user's requests. The user will see this message: "We are sorry, no recipes were found to match your requirements. Try to change your research criterias. You might have more results." Otherwise, the website works properly and generates a list of recipes that meet the given requests. Along with each recipe is the link to more in-depth instruction, as well as the nutritional information for each recipe. Also, the site calculates and outputs the total nutrients for the diet plan of recipes (not including the foods).

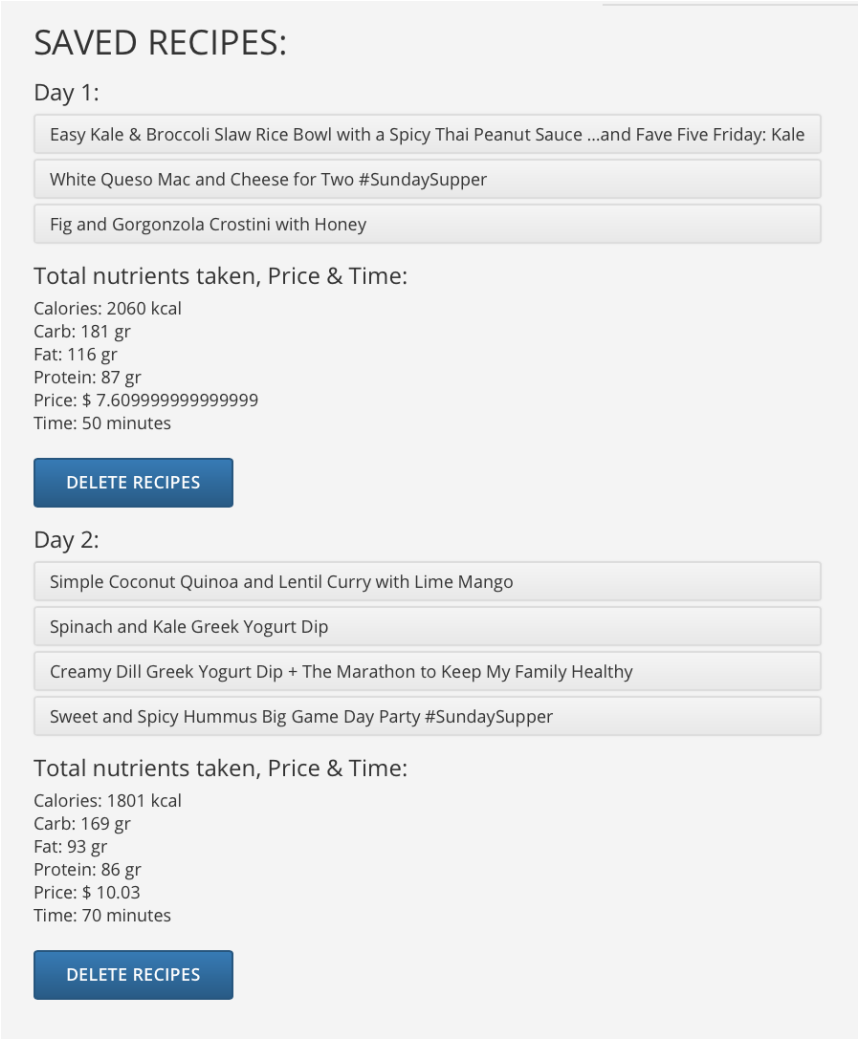
Below is an example of the results a user may see. The user can learn more about each food and recipe by simply clicking on it.



4.5 Access Later

If the user wishes to get a diet plan for multiple days, or in general wishes to save recipe plans for later use, they can choose to save recipes when they are generated. Otherwise, each individual recipe and food are saved in history, which can also be accessed at all times. This is especially crucial, as this is where the user can submit feedback on the recipes and foods. This helps with the user's future recipes (they can specifically request their favorite recipes and exclude recipes they don't like), as well as all the users. This is because each like, dislike, or no opinion is taken into account when calculating the website's most popular recipes. This is just another way for the user to manage their diet plan and try some new recipes they may not have otherwise seen.

An example of an output for the Saved Recipes page can be seen below.



5 Internal Implementation

The implementation is separated into three parts: getting the data, database development, and web application development. The code can be accessed at https://gitlab.com/Khnil/diet_optimizer.git.

5.1 Getting the Data

This part of the site was not the focus of this project. We implemented previous code to accomplish this step.

5.2 Database Development

MySQL is used as our choice of relational database management system with the intention to exploit its full-text search. We also took full advantage of the already built-in features of the Django framework for the web development of the site. For example, Django takes care of transforming Python objects into acceptable data for the database through its built-in Object Relational Mapper (ORM). Additionally, we never have to write any MySQL code, as Django does this for us.

The database has mainly ten tables that are used in the code:
- auth_user
- feedback
- feedbackrawfood
- intolerences
- preferencecuisines
- preferencerawfoods
- recipes
- saved_recipes
- usda
- users

Auth_user and **users** contain basic information of all users, such as username, age, height, gender, etc. These are required for calculating default nutritional constraints. **Feedback** stores the information for past recipes in each user's history, such as the recipe name and the user's 'mark' (like, dislike, no opinion, didn't try). **Feedbackrawfood** stores the same information, but for the foods. **Intolerences** stores all the intolerances for every user in the database. **Preferencecuisines** has all the favorite cuisines for each user. If a user doesn't have any favorite cuisines, they are not included in this table. **Preferencerawfoods** works the same way, but for the users' favorite types of foods (beverages, poultry, fast foods, etc.). **Recipes** stores important recipe information for the developer, such as the URL to the website and the recipe's ID on Spoonacular (the site we use to get recipes). **Saved_recipes** has all the information for the saved recipes

for each user. **Usda** is a very extensive list of ready-made foods that the website uses to generate foods for the user.

5.3 Web Application Development

Django is modeled around a Model-View-Controller (MVC) framework. MVC is a software design pattern that aims to separate a web application into three interconnecting parts: 1. The **model**, which provides the interface with the database containing the application data 2. The **view**, which decides what information to present to the user and collects information from the user 3. The **controller**, which manages the bulk of the application's data processing, application logic, and messaging

However, Django uses slightly different terminology: **model, template, view**. So by calling to a "view" (controller according to MVC framework), all the necessary calculations are completed and a template (html) is rendered for display. Django's easy-to-use nature also came in handy here, as we made use of its bootstrapping tools for the display.

The flow of the application, as explained in the user interface, is the following: fill in profile, manage diet, select foods, generate diet, and access later. These five steps can be achieved by the use of the tabs on the left-hand side of the page, which include: About (views.about_logged_in), Personal Information (views.profile), Get a Recipe (views.get_recipe), History (views.history_not_tried, views.history_tried, views.saved_recipes), Settings (views.personal_settings, views.user_profile, views.account_settings), and Most Popular Recipes (views.most_popular_recipes_foods).

The first step is creating a profile (views.signup and views.personal_details). The user will then be prompted to log in each subsequent time it visits the website (Django login imported view).

The next step is managing the user's diet. This is mainly done through finding recipes that follow a certain diet and intolerances (this information is specified in views.personal_details and can be updated in views.user_profile). The site does its best to manage the recipes it generates to cater to these needs, especially through its "popup" feature that asks for confirmation if, let's say, a vegetarian asks for meat products (views.get_recipe_confirm). Also, the user can choose to prioritize a certain nutrient and adjust the ranges of all nutrients (forms.GetRecipeForm > variables: objectiveNutrition and objectiveMinMax). And, the user can update their diet preferences (diet, intolerances, favorite cuisines, etc.) at any time through the diet settings page (views.user_profile).

Once the user's settings and information are updated, they are ready to request a diet plan (views.get_recipe). They can choose from several cuisine and food category options (forms.GetRecipeForm > variables: cuisines, recipeTypes, and rawGroups). If the user would like to include a specific recipe they had liked previously, this can also be done (forms.GetRecipeForm > __init__()). On the other hand, recipes that the user dislikes will automatically be excluded from future diet plans (models.py > class RecipeHandler > get_recipes()). This will then prompt the views.results, which leads us to the next step: generating the user's diet.

The process of getting a diet is computed in the views.results. In this view, the site randomly generates two ready-made foods from the filtered USDA database, taking into account the food types the user had requested. These foods and their details are stored in the raw_foods list, with each element being a food and its details (in the form of a dictionary). Each food also has its individual total nutrients calculated and stored in the views.results as well. As for the recipes, the process is much less random. After obtaining all the user's requests (cuisines, food types, etc.), the website creates a URL using the recipe search tool Spoonacular (models.py > class RecipeHandler > get_URL()). This URL results in a list of all possible recipes, without taking into account the nutritional objectives and ranges, nor the time and/or price constraints. So, a LinearProgrammingSolver object is created, which then narrows down the list generated from Spoonacular to optimize the user's nutritional, price, and/or time constraints [1]. It is also in views.results that any specifically asked-for recipes are incorporated and factored in. Also, for both the food and recipes, this view also takes care of creating a feedback for each food/recipe if it doesn't have one already. The importance of feedback can be seen in the next step: access later.

The "access later" portion of the website consists of some of the newest features added to the website (see Diet Optimizer Tests: First Priority Improvements). These additions were added to make the website more user-friendly and make diet planning for multiple days more feasible. More specifically, this includes the save recipes feature (views.save_recipes_in_db, views.saved_recipes, views.delete_recipes_in_db), the separation of 'to be tried' and 'tried' recipes, as well as their ranking (views.history_not_tried and views.history_tried), and the most popular recipes page, (views.most_popular_recipes). When a user saves a recipe, the save_recipes_in_db view is prompted. Then, the saved_recipes view works just like the results view in the sense that all the same information for each recipe is displayed for the user, but each set of recipes is distinguishable by day. This is nice because otherwise, only the individual recipes would be saved in history and the user couldn't plan diets in advance. After, if the user wishes to unsave a set of recipes, they can delete them, which prompts the delete_recipes_in_db view. The history_not_tried view was created to make it easier for the user to see which recipes they haven't tried yet. This, and views.history_tried are sorted first by rank and then alphabetically (models.py > class Feedback > class Meta). Similarly, in order to calculate the five most popular recipes, all the recipes are ordered by their percentage, which is calculated in the history tried and not tried views (models.py > class Recipe > class Meta).

See separate documentation information which summarizes all the functions used in the web development.

[1] It should be noted that this optimization process has proven to be quite slow, and too often, narrows down the list of recipes to a point where no recipes can be found. See **Future Work** for more on this.

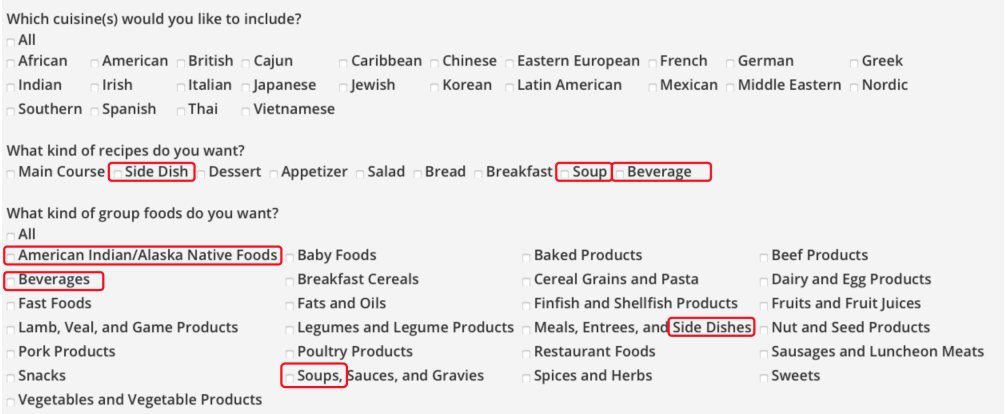
5.4 Future Work

We would like the user to be able to input his/her own recipes, calculate its overall nutritional content based on those ingredients, and save them as preferred food items. This would further advance the user's ability to customize and manage their diet.

As of now, the algorithm used to generate recipes can be quite slow. Often times, the user must wait at least one minute to get a list of recipes, and sometimes, the site can't even find recipes. Ideally, the algorithm should be optimized to shorten this waiting process and more recipes should be added to the pool of recipes.

As for the ready-made food items suggested from the USDA database, while they are reliable in that the website will always be able to suggest a couple that suit the user's requests, they are unreliable in the sense that they can be very vague (ex: mozzarella cheese sticks, DENNY S). And, because these food names can be vague, it makes it nearly impossible to ensure that these foods truly do suit a user's specific diet, since we don't always know the specific name of the food/brand or the ingredients it is made from. So, possibly clarifying some of the unclear food names descriptions would be beneficial.

Also, the categorization of the foods can be a bit unclear, due to the structure of the USDA food database. For example, even though there is a separate section for cuisines, the database has a "group foods" category (which is typically types of foods, not cuisines) called "American Indian/Alaska Native Foods". To avoid confusion, moving this category to the cuisines section (which includes cuisines such as Irish, Chinese, American, etc.) would make more sense. There is also confusion in the "type of foods" and "group foods" categories, as there is some overlap. For example, "Beverages", along with a few others, is repeated in both sections, which is again due to how the USDA database is structured.



Overall, it would be beneficial to reorganize this database to better suit the website's needs. This can and will be very time consuming, as this database is extremely large.

For more future plans, see Diet Optimizer Tests (Second Priority Improvements.

6 Works Cited

[1] J. Arthur, Harris, and Benidict Francis G. *A biometric study of basal metabolism in man* . Washington Carnegie Institution of Washington, 1919. Print.

United States. Institute of Medicine of the National Academies. *Dietary Reference Intakes DRI The Essential Guide to Nutrient Requirements*. 2006. Web. <http://www.nal.usda.gov/fnic/DRI/Essential_Guide/DRIEssentialGuideNutReq.pdf>.