**Meal Planning Application for Custom and Specialized Dietary Needs**

**Advanced Python CIS 4930  
spring 2014**

**Team Members:**  
Mark Brophy  
James Robe  
Vi Tran  
Jason Maynard  
Erik Hendrickson

# Introduction

Yummly is an advanced, food-based search engine that has pointers to approximately 1 million recipes and food related resources. According to Wikipedia:

***Yummly*** *is a semantic* [*web search engine*](http://en.wikipedia.org/wiki/Web_search_engine) *for* [*food*](http://en.wikipedia.org/wiki/Food)*,* [*cooking*](http://en.wikipedia.org/wiki/Cooking) *and* [*recipes*](http://en.wikipedia.org/wiki/Recipes)*. It ‘understands’ food on a variety of levels, allows users to search by ingredient, diet, allergy, nutrition, price, cuisine, time, taste, meal courses and sources, and ‘learns’ about users based on their likes and dislikes. Yummly uses this information to categorize food for search and make recommendations.[[1]](#footnote-1)*

Yummly’s mission is to be the world’s largest, most powerful, and most helpful food site in the world.[[2]](#footnote-2)

Yummly provides a developer site and API that enables a wide variety of options for developers (Figure 1).



Figure 1: Potential Yummly API options

Yummly also provides a Python library on GitHub

<https://github.com/dgilland/yummly.py>

# Goals

There are an infinite number of applications that can be built from the Yummly API. However, our team has been able to narrow the focus down to a targeted subset. We propose building a cross-platform Python application targeted for the needs of college students.[[3]](#footnote-3)

While this segment is a small subset of the overall population, there are still a wide variety of needs within the student population. The app will provide a resource for students such that they can select meals based on three main criteria:

1. Dietary needs
2. Cost
3. Availability

## Dietary needs

Dietary needs can be as simple as a food preference or religion but may also be driven by specific health concerns such as diabetes. To illustrate, Dr. Nathanael Paul recently presented a set of needs to our class to help those suffering from diabetes. In terms of specific dietary needs, this application could be customized by user such that the user could select a proper diabetic diet based on blood sugar, time of day, and availability of ingredients. As an example, the application could make several recommendations for meals or snacks based on carbohydrate count. Dr. Paul mentioned that, as a diabetic, he needed to eat 30 grams of carbs at particular intervals during the day. A breakfast bar may be one source to provide this calorie count but the Yummly app could also make other recommendations to serve the same requirement.

The Yummly API supports the inclusion or exclusion of specific ingredients. For example, if I really like lasagna but don’t like ricotta cheese, I could specify that requirement in my search for lasagna recipes. Perhaps the user has an allergy to peanuts. The user could then exclude peanuts from the search result. Dr. Jeanty said:

*The nice thing about this API is that you can request recipes with just tomatoes and mozzarella for instance. Or you could ask for recipes that “DO NOT” contain peanuts. Or you can ask for recipes that are appropriate for Christmas or Hanukah, etc… If you check out their available plans, there is a free plan that will allow for 500 API calls/day.*

## Cost

College students typically do not have large budgets for meals. This application would enable students to quickly select meals based on cost per meal. For example, the user could select from several meal plans that are derived from a price per meal. For a weekly plan, the student could select a plan for $3, $5, or $7 meals during the week as the main source of nutrition, but could also select a pricier $10, $15, or $20 meal for weekends or special occasions while still being compliant to the nutritional needs of the user.

## Availability

Finally, the Yummly API enables the user to pick meals or plans based on the current inventory of ingredients on hand or locally available. Imagine being able to quickly find a recipe based on what is in your refrigerator at that moment!

# Design components (i.e., tools required to complete project)

## API

The team will research and learn to work with the Yummly API (Figure 1 ) such that we can accomplish the three design goals listed above.

## Web Server

A web server will be required to host the application. The team is performing trade studies to determine the best solution.

## GUI

A stylish Python-based GUI will be developed using TkInter, WxPython or other framework depending on the current best practices, research, and project schedule. A list of potential GUI frameworks is found here:

<https://wiki.python.org/moin/GuiProgramming>

The team will also research the feasibility of incorporating the design into a mobile app platform. New tools are being introduced to enable mobile app interfaces. The team will explore the feasibility of porting the design to a mobile app. If this is not feasible in the project timeframe, or proves to be too complex, we propose to create a design in the spirit of future porting to mobile platform. This will be accomplished by researching mobile app design for Python and attempting to create a Phase I design that is oriented toward future mobile app design.

## Database

In addition to an API, and GUI, the team will develop a database to store user profiles as well as the user’s history and ratings of recipes. (*Expand ideas here*.. )

# Future enhancements

* Link to coupon site
* Mobile app version
* Link to diabetic tracking or monitoring system
* Link to shopping service (<http://www.wired.com/business/2014/02/next-big-thing-missed-future-groceries-really-online/>)
* Other?

1. http://en.wikipedia.org/wiki/Yummly [↑](#footnote-ref-1)
2. http://www.yummly.com/about/ [↑](#footnote-ref-2)
3. Dr. Jeanty suggested Yummly as a project idea and the potential to link the resulting application to a coupon site such as [*http://www.coupons.com/*](http://www.coupons.com/) [↑](#footnote-ref-3)