***Foodlicious***

**What the project accomplished**

1.We created a recipe share website where users can upload , search and review other people's’ recipes.

2.Each user has his own user profile where he can see his own recipes, reviews as well as recommended recipes.

3.Users can search recipes through various criteria including title, category, tags, ingredients and calories.

4.Users can give reviews to others' recipes including rating, tags and comments and  receive feedback for their own recipes in the same way.

5.Users can update/delete his own recipes, view/delete his own reviews, view other recipes, and add/view reviews.

6. A ranking system and recommendation system were established.

**Project Usefulness**

Recipe sharing is a very popular idea these days. We'd like to create a website where users can upload, get feedback on their self-published recipes from other users, search recipes, and also be recommended with recipes.

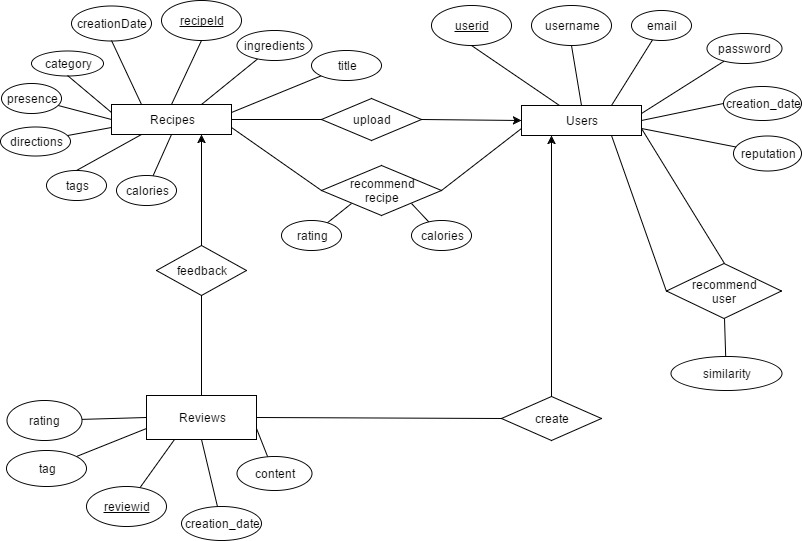
The site is similar to StackOverflow, but a food version. This allows users to get feedback from other users as well get recommendations for food to make. We combined multiple functions together so that we can automatically rank current recipes in our database, find the most similar user like you and recommend high-rated, low calories recipes for you.

**Data in our Database**

Our database mainly consists of three relations: Recipes, Reviews, and Users. The three types of data are closely related to each other. Each recipe and review is linked to the user who created the respective review/recipe by having a field for the user ID. Each review is uniquely linked to a recipe and an user. Each recipe has an unique user who uploaded the recipe.

**ER Diagram and Schema**

ER Diagram



Schema

Recipes(recipeId, creationDate, ingredients, category, directions, tags, title, userid, calories, presence)

Users(userid, username, e-mail, password, creation\_date, reputation)

Reviews(reviewid, recipeid, userid, creation\_date, rating, tag, content)

recommend user(userid, recommended userid, similarity)

recommend recipe(userid, recipeid, rating, calories)

**Data Source**

Our “main” data is the recipe so we created a Python script that crawled through the popular recipe website Tastemade. Using the web scraping API, Beautiful Soup, we were able to crawl across about 110 recipe webpages and parse them into a format that CPanel could understand and populate the database. In addition, we also have some user-generated recipes uploaded by ourselves.

For the other two relations: reviews and users. Users are all created when we registered in the webiste and the corresponding data is stored in the Users relation. The reviews are mostly user-generated by different users and for different recipes.

**Features and Functionality**

-Registration, Log in/out

-User profile display containing own recipes, reviews and recommended recipes

-Create, update, delete own recipes and search for recipes

-Create and delete own reviews

-View others' recipes and reviews

-Create a ranking system

-Create a user based collaborative filtering recommendation system

**Basic Function Example**

One of our basic function is search recipes. We can search for recipes based on multiple criteria including title, category, tags, ingredients and calories.  We will return all the results corresponding to the search criteria and display related reviews.

Searching for a recipe based on title/ingredients/any relevant steps. This example shows searching for any recipe related to “Taco”.   
  
SELECT title, ingredients, directions   
FROM `Recipes`   
WHERE ingredients LIKE '%Taco%'   
OR  directions LIKE '%Taco%'   
OR title LIKE '%Taco%'   
ORDER BY  
CASE   
WHEN title = "Taco" THEN 0   
WHEN title LIKE "Taco%" THEN 1  
ELSE 2  
END ASC

**Actual SQL code snippet**

Search Recipes:

$sql="SELECT \* FROM Recipes WHERE title LIKE '%$title%' AND category LIKE '%$category%' AND tags LIKE '%$tags%' AND calories <= $calories";

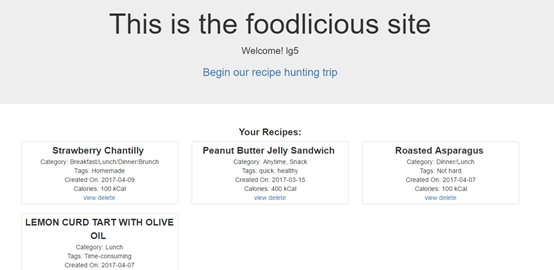
**Data Flow**

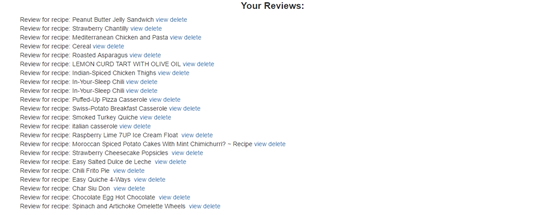


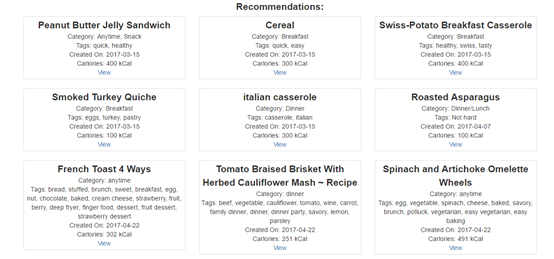
The first time we go to this website, we will see this interface. Firstly, we can click the sign up here button for registration:



When we finished all the information. All the corresponding data will be inserted to Users relation in our database and automatically generate an userid for current user. Now we can go back to sign in page and sign in.





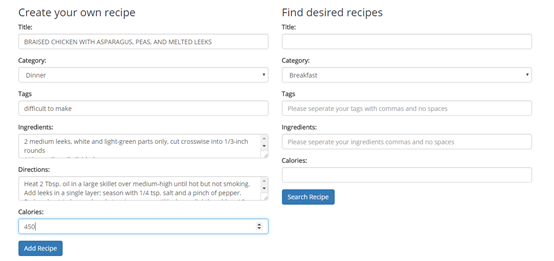


After signing in, we will see the user profile page with your own username. We will see all your own recipes, reviews as well as recommended recipes for you. We can click the view/delete button near recipes/reviews to view recipe/reviews or delete them. For example, if we click view near a recipe, we will see results like:



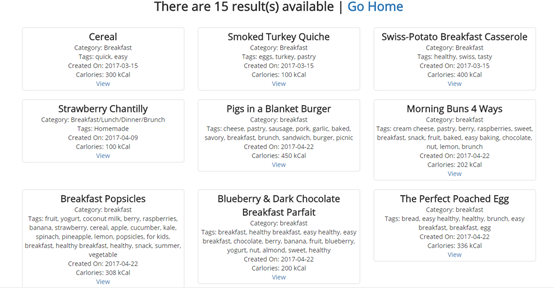
Now go home and click the Begin our recipe hunting trip button and we will go to the home page.

Here we can add and search for recipes and see relevant reviews:



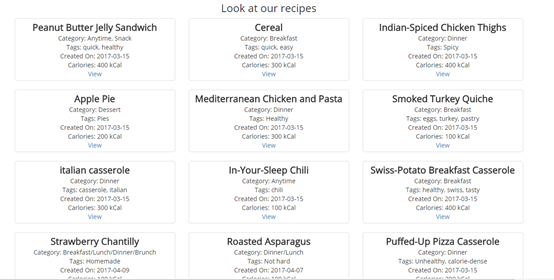
On the left hand side, we can add our own recipe to Recipe relation and all the data will be stored to the relation. After insertion, we will stay in this page.

On the right hand side, we can search for recipes through different criteria. For example, we can search breakfast and we wll get results like:

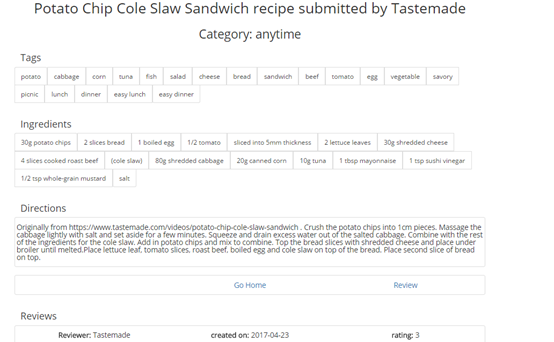


Again, we can click view to see each recipe or click go home to our home page and continue other operations.

On the bottom of our home page, we can see all of our recipes in our database:



We can view them and retrieve all the information related to the recipe and corresponding reviews:



**Advanced Functions**

1. Recommendation System

We built a recommender system using user based collaborative filtering method. The basic idea we employed is to find the user who is most similar to the current user, and use the ratings from this most similar user to make recommendations, i.e. we recommended high-rating recipes with lowest calories of the user most similar to current user.

It is advanced because firstly we need to link all relations in our database to gather information as a pivot table containing all the recipe ratings, indexed by the userid and columned by the recipeid.

This allows us to find recipes to recommend, given a particular user. Now, every user, or every row in the pivot table can be regarded as a vector containing ratings to all the recipes.

Secondly, we implied a general distance measurement  method as cosine similarity to calculate distance between two vectors. Current user’s vector is compared to vectors of all the other users by this method  to derive the distance, i.e. the similarity, between current user and other users and find the user with minimum distance, i.e. most similar one.

Then, we recommend current user with high-rating recipes reviewed by the most similar user. And we return these recommended recipes with lowest calories and highest ratings.

**Technical Challenges**  
  
Development Plan Challenges/Changes  
We initially planned to create a resume review site, but due to one member dropping the class and the unfeasible specs, such as using OCR to automatically detect text off of resume PDF files, we decided to change it into something that the rest of the team would be able to achieve within the development period. However we kept a lot of the initally proposed functions from the resume idea to our current idea.

PHP/Python Pass Variable Challenge

Since we used some python files, one of the challenge is to pass variables between php and python. Before that, we need to first create a virtualenvironment and install necessary python modules. Then after different trials to pass variables between php and python, we concluded that we could use passthru/exec(" python path/to/python.py $variable...") in php and receive the variable in python as 'import sys, variable=sys.argv[n]). From python to php, we can use json to dump and decode. Be sure to notice where the variable is passed and in which command to avoid mistake!

**Comparison with initial development plan**

Basically, we followed the steps and schedules in our initial development plan and proposed specifications.

**Labor Division/Teamwork**

We set deadlines for certains aspects of the project throughout the semester and reminded each other to reach them. We tried to divide the work for each stage into 3 parts for everyone through the semester.  
  
Dehua Chen:  Wrote some of the basic SQL queries for some functions.  Constructed an advanced function:  Merit Ranking for Recipes, configured most of the UI for the websites, improved upon the existing search engine  
  
Patrick Wang: Wrote some of the SQL Queries for various functions, crawled data from other websites for recipe data.   
  
Lingsong Gao: Wrote some basic SQL functionalities as search recipes, insert/view/delete reviews, log in/out and registration and assisted in creating the user profile. Constructed an advanced function: recommendation system