Graze

Making food easy

Michael Barlow, Gregory Benton, Rasheeq Jahan, Sofia Mehrotra, Erin Ruby

CSCI 3308
University of Colorado
July 2017

- Project Tracker: Our main project tracking tool was Trello, our team's board can be found here (https://trello.com/b/SjEkaJvM).
- The video can be found here(https://youtu.be/kY6gEQqslb4), and the link is in the Github repository under the title Milestone6_video.
- VCS: We used Github as our version control system, our public repository can be found here (https://github.com/g-benton/3308-Project). Note that our main branch is titled working-branch, and that represents the final stage of our project.
  - Source Code: Contained in all the folders found, primarily within the app directory.
  - Test Cases: Use case test documents are in the documents folder in the Milestone5_Use_Case_Testing file
  - Auto testing: Python scripts to conduct automated testing are located in the auto-test directory
  - Video: here (https://youtu.be/kY6gEQqslb4) (link also on Github under the file Milestone6_video
- Github Contributions: shown below in figure 1.
- Deployment: The website can be found here(http://h-django.herokuapp.com/). Deployment instructions for a local server version are included in the Github readme.
- Repository Structure:
  - app: main Django folder, contains all necessary Django documents and modules
  - auto-test: folder containing scripts to test the database against an array of inputs as well as the ChromeDriver application
  - database-models: contains the backend model and database information used by Django and Docker (allowed everyone to have same database across machines)
  - populate-db-yummly: scripts and methods used to populate the database initially using the Yummly API
  - presentation-writeup: Just a few image files needed for some of the milestone write-ups
  - recipe_db_hard_backups: SQL backup files just in case we had any database errors.
  - The files in the main folder are files needed for the milestone write-ups as well as files necessary for everyone to build their docker containers at the highest level of our directories.
- Database Design:

  We constructed a database built out of two tables with simple entries with a large relation table. The first table of entries is the `recipe` table that contains the recipes as records with attributes of ID, name, and yummly_id which is the id that Yummly does get calls with. The next table is the `ingredients` table which holds ID's and ingredient names. These tables both share a one to many relationship with the `ingredient_recipes` table, in which each record relates an ingredient to a recipe. This played well with Django's model structure, allowing us to have easy access to the database using python scripts.

An ER diagram is shown below in figure 2.

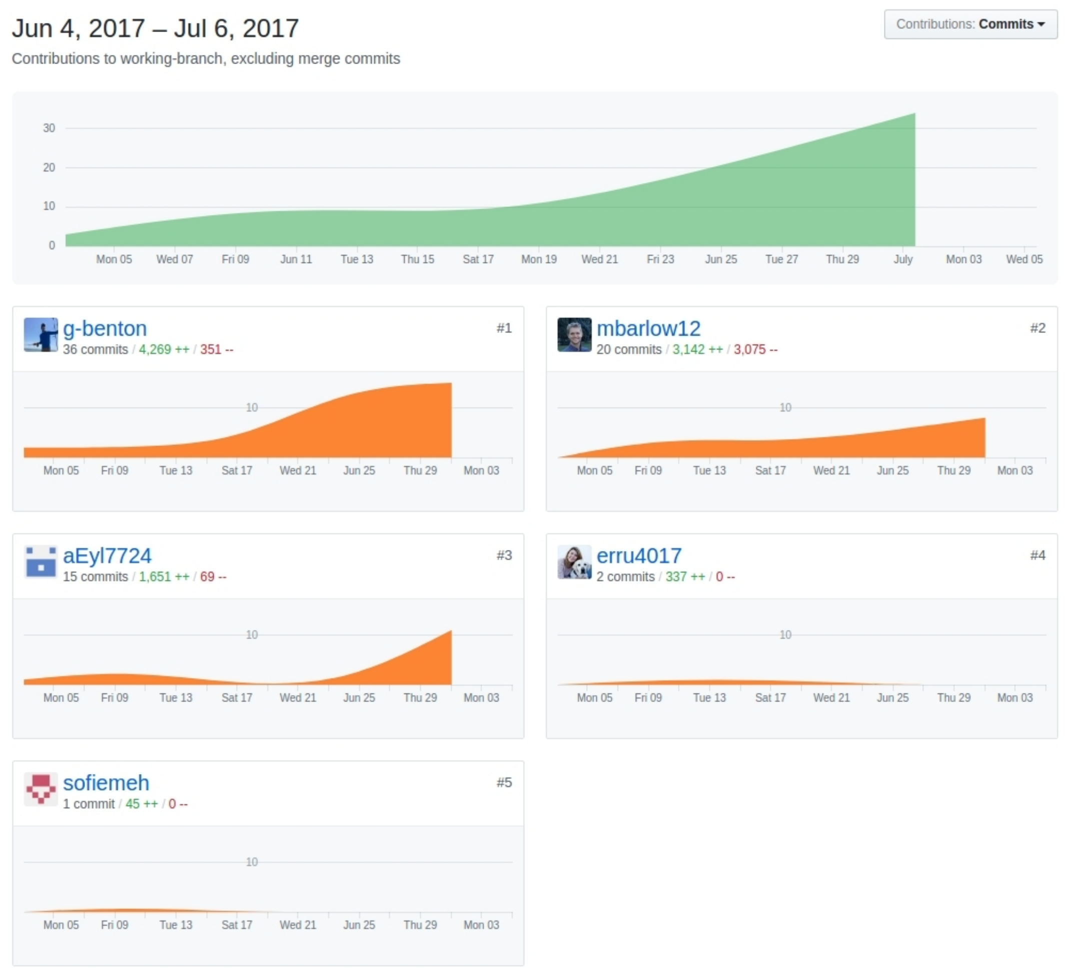
Jun 4, 2017 – Jul 6, 2017
Contributions to working-branch, excluding merge commits
Contributions: Commits
g-benton #1
36 commits / 4,269 ++ / 351 --
mbarlow12 #2
20 commits / 3,142 ++ / 3,075 --
aEyl7724 #3
15 commits / 1,651 ++ / 69 --
erru4017 #4
2 commits / 337 ++ / 0 --
sofiemeh #5
1 commit / 45 ++ / 0 --

Figure 1: Michael = mbarlow12, Greg = g-benton, Rasheeq = aEyl7724, Sofie = sofiemeh, Erin = erru4017

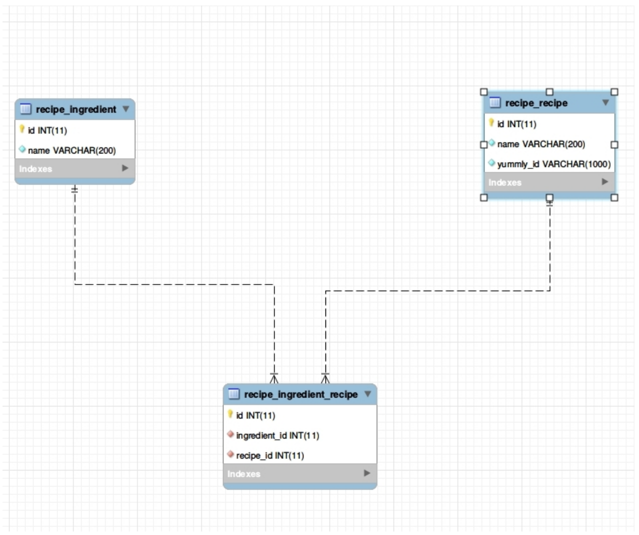

Figure 2: ER Diagram of the database