ToMEto Weekly Report 9

## Accomplishments

Jonathan – Polish up frontend, make visually appealing and ready for beta testing.

Matthew – Improve the objective algorithm testing and comparing.

Albert – Fixing issues in the backend: algorithm tweaking and Unicode encoding.

Charlie – Refactoring of codebase, overall improvement and streamlining of program execution.

## Work for next week

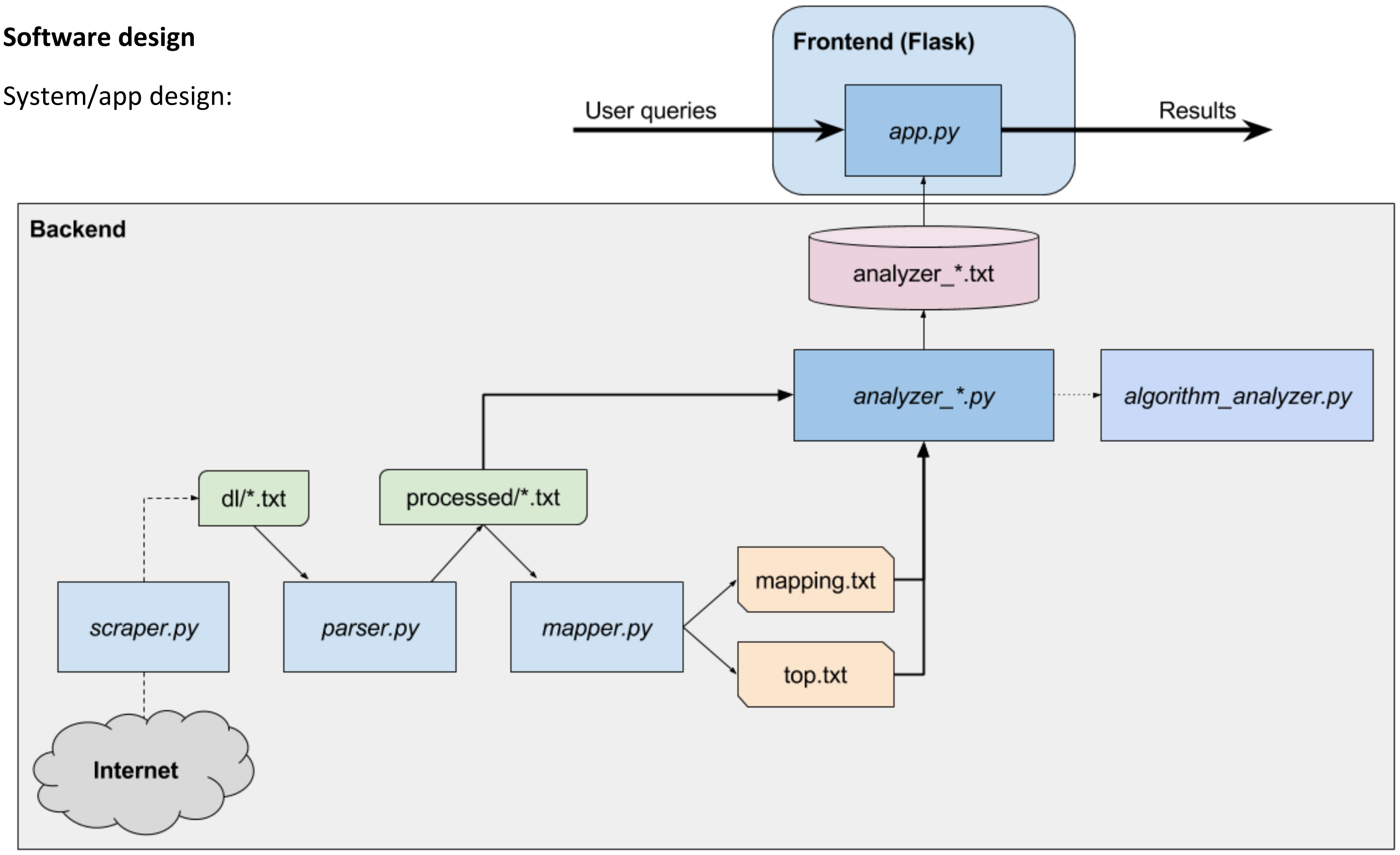
Prepare for beta testing/launch of service. Goal is to have it up and running so that end users can begin using our service starting next week. Hopefully can receive feedback on ways to improve our app.

## Challenges

Objective testing is difficult to do in a way that captures the goal of our application. Our previous testing methodology favors algorithms that finds the most common ingredients (e.g. “salt” or “flour”). However, this is generally not what we want for recommendations since they are not creative or exciting at all. This explains the poor performance of the algorithm that we subjectively think is the best at recommendations (weighted PMI). We are still working on improving the testing methodology to more accurately assess the quality of each algorithm’s recommendations.

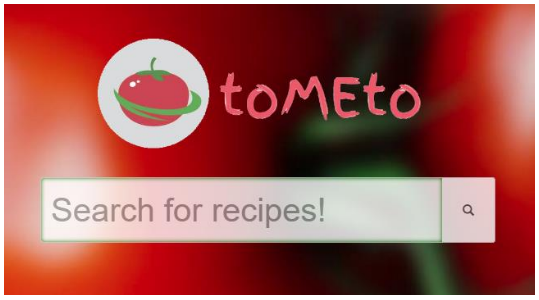
## Software design

System/app design:

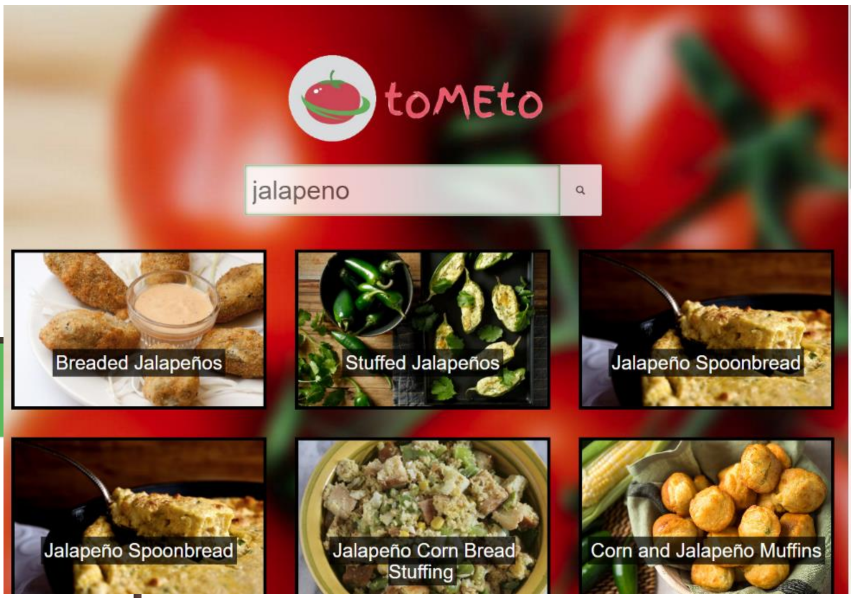


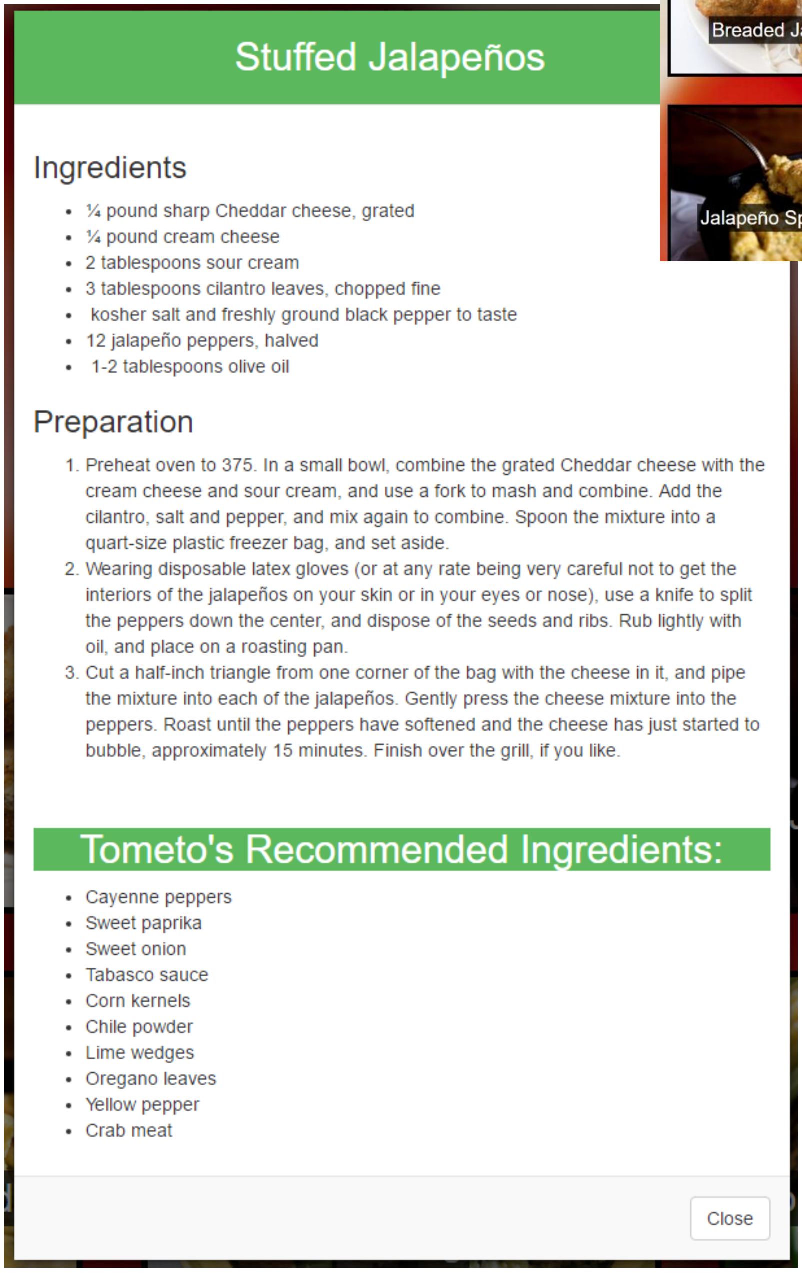
Explanation:

We launch our service through *app.py*. This brings up the main page for ToMEto →



The user enters a search query, which gets passed to *app.py* and then *search.py* in the backend. The results are returned to the user through *app.py*, displayed as a webpage →





Stuffed Jalapeños

Ingredients

- ¼ pound sharp Cheddar cheese, grated
- ¼ pound cream cheese
- 2 tablespoons sour cream
- 3 tablespoons cilantro leaves, chopped fine
- kosher salt and freshly ground black pepper to taste
- 12 jalapeño peppers, halved
- 1-2 tablespoons olive oil

Preparation

1. Preheat oven to 375. In a small bowl, combine the grated Cheddar cheese with the cream cheese and sour cream, and use a fork to mash and combine. Add the cilantro, salt and pepper, and mix again to combine. Spoon the mixture into a quart-size plastic freezer bag, and set aside.
2. Wearing disposable latex gloves (or at any rate being very careful not to get the interiors of the jalapeños on your skin or in your eyes or nose), use a knife to split the peppers down the center, and dispose of the seeds and ribs. Rub lightly with oil, and place on a roasting pan.
3. Cut a half-inch triangle from one corner of the bag with the cheese in it, and pipe the mixture into each of the jalapeños. Gently press the cheese mixture into the peppers. Roast until the peppers have softened and the cheese has just started to bubble, approximately 15 minutes. Finish over the grill, if you like.

Tometo's Recommended Ingredients:

- Cayenne peppers
- Sweet paprika
- Sweet onion
- Tabasco sauce
- Corn kernels
- Chile powder
- Lime wedges
- Oregano leaves
- Yellow pepper
- Crab meat

Each is a recipe that the user can click for the full recipe and recommendations.

← For "Stuffed Jalapeños"

The user can then use this recipe to make some delicious and creatively new dishes!

**Software modules**

| Module Name | Description/Functionality | Author | Lines of code |
|---|---|---|---|
| scraper.py | Scrapes for recipes off NYT and Allrecipes websites | Charlie Tong | ~100 |
| parser.py | Extracts relevant information (title, ingredients, body) from scraped recipes | Charlie Tong | ~150 |
| mapper.py | Finds top ingredients and maps uncommon ingredients to real ones (improves data quality) | Charlie Tong | ~400 |
| analyzer_*.py | Different algorithms for recommendations | Albert Ge, Charlie Tong, Matthew Jin | ~600 |
| algorithm_analyzer.py | Objectively tests differences between algorithms | Matthew Jin | ~100 |
| utils.py | Utility functions used by different files | Albert Ge, Charlie Tong, Matthew Jin | ~200 |
| search.py | Handles user search queries (forward to NYT search) | Matthew Jin | ~20 |
| app.py + CSS/HTML | Entry point to the program, used to start up the ToMEto service (built using Flask Framework) | Jonathan Joo | ~600 |

Each module uses text files as basic input/output. For more details, see architecture diagram above.

Everything is coded in Python. Flask Framework was used to frontend (app.py).

Fun fact: the combined size of the project (all code, scraped files, parsed files, and images) totals 260K files and 16GB.