Project Implementation Report (Group 5): Recipe Generator

Group Members: Jisu Baek, Marcus Loke, Maya Bayram, Omar Nesnas, Patrick Alvermann

1. Background

Our web application revolves around nutrition. A healthy lifestyle has become the forefront of people's agendas as a result of the pandemic. However, people usually do not have the skills nor time to prepare a meal at home. Additionally, one does not always have time to shop for a recipe that they have found online. Further, food allergies and dietary restrictions are on the rise, causing people to look for personalized dietary solutions. As a result, the personalized nutrition market is set to double from $8.2 Billion in 2020 to $16.4 Billion in 2025 (CAGR 15%). Thus, we aim to make the home cooking experience more convenient than ever with a recipe generator that works with whatever one has available in their kitchen.

2. Definition of Business

Our Recipe Generator aims to meet the above mentioned market demands by allowing users to search for recipes across the web. Essentially, the application will return recipes to users based on what ingredients they input into the search bar. Additionally, the application further filters to only return the recipes that use the prescribed ingredients.

3. Data Source

The recipe results will be queried through the Spoonacular Rapid API (https://rapidapi.com/spoonacular/api/recipe-food-nutrition/), which provides access to over 365,000 recipes and 86,000 food products. The results are then returned in JSON format as shown below.


```
▼ [ 5 items
  ▼ 0 : { 6 items
      "id" : 641803
      "title" : "Easy & Delish! ~ Apple Crumble"
      "image" : "https://spoonacular.com/recipeImages/Easy---Delish--Apple-Crumble-641803.jpg"
      "usedIngredientCount" : 3
      "missedIngredientCount" : 4
      "likes" : 1
  }
```

To demonstrate the application, we registered for a basic plan that allows 50 requests and 500 results daily for free. However, for full scale deployment, we would need to purchase the mega plan, which allows for 100,000 results daily for $999 per month.

4. Implemented Technologies and Rationale

The following three (3) technologies will be used in this project:

a. *Frontend Python Flask*

Flask is a small and lightweight Python framework that provides tools and features for creating web applications in Python. We can build an application quickly using only a single Python file, which is good for such exploratory projects. Also, Flask is extensible and does not force a particular directory structure, making it really easy to use and very scalable.

b. *REST API*

REST API allows us to interact with the food and recipes database by simply performing HTTP requests. It also allows us to get inputs from users to filter and return relevant search results.

c. *Backend MongoDB Stores*

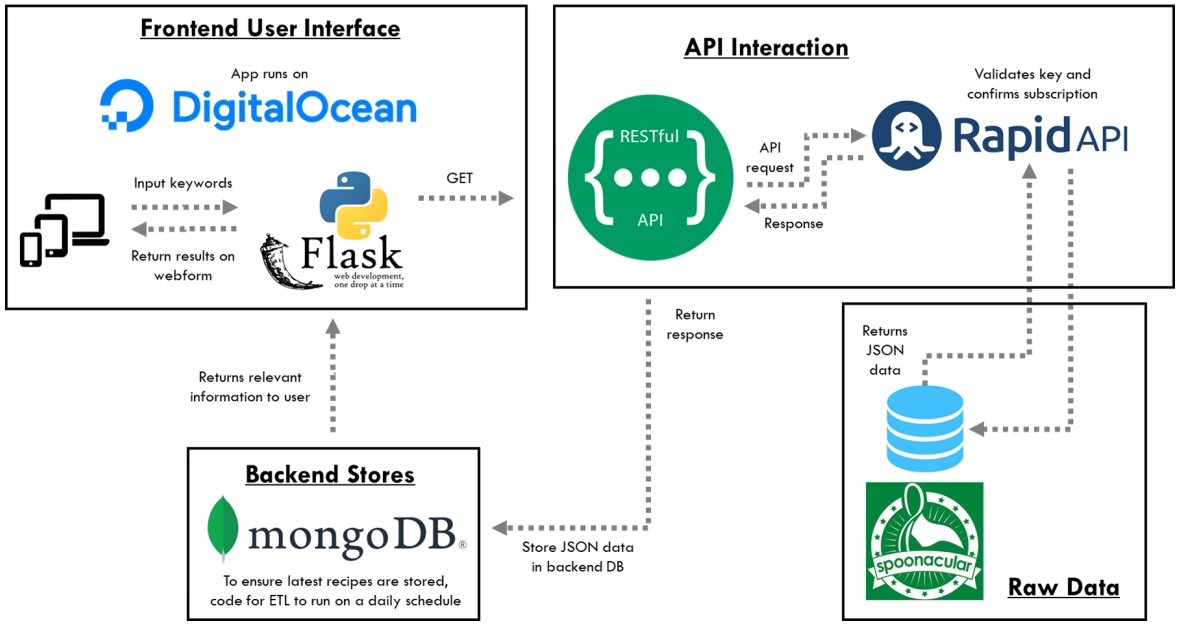
We want a consistent database because it is important to return relevant and latest information to the user, i.e., they must reflect the most recent completed write (in real time) of that data, no matter which server processed that write. We also want a database that can handle network breakdowns even if it means compromising on the availability of certain parts of the data. MongoDB is appropriate for these reasons and also because RapidAPI returns data in JSON format.

5. Implemented Design

The implemented design is shown in the figure below. There are four (3) major parts: (1) First, the frontend user interface is where the user would key in his/her ingredients on the webform. (2) Second, the API interaction portion will use REST API to query the data from RapidAPI, which fetches the raw data from the Spoonacular database. (3) Finally, the returned JSON results will

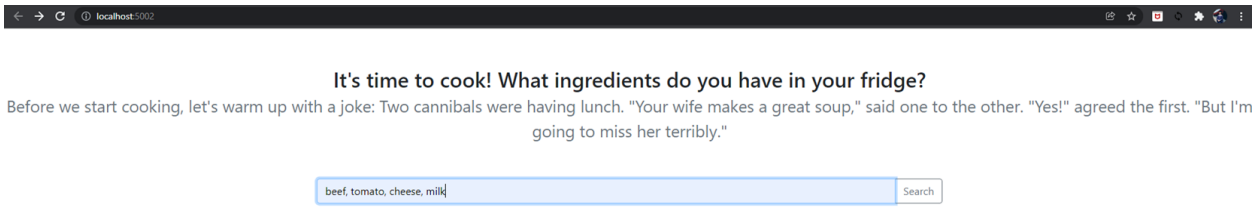
be stored in MongoDB where we intend to code the ETL to run on a daily schedule to ensure the latest recipes are stored. Then we will return the recipes to the webform by querying MongoDB.

To clarify, our app is not running on DigitalOcean for this proof of concept. We listed it here to illustrate the future possibility of using it as a server to host our app and for estimation of costs should we bring the app to full scale development.

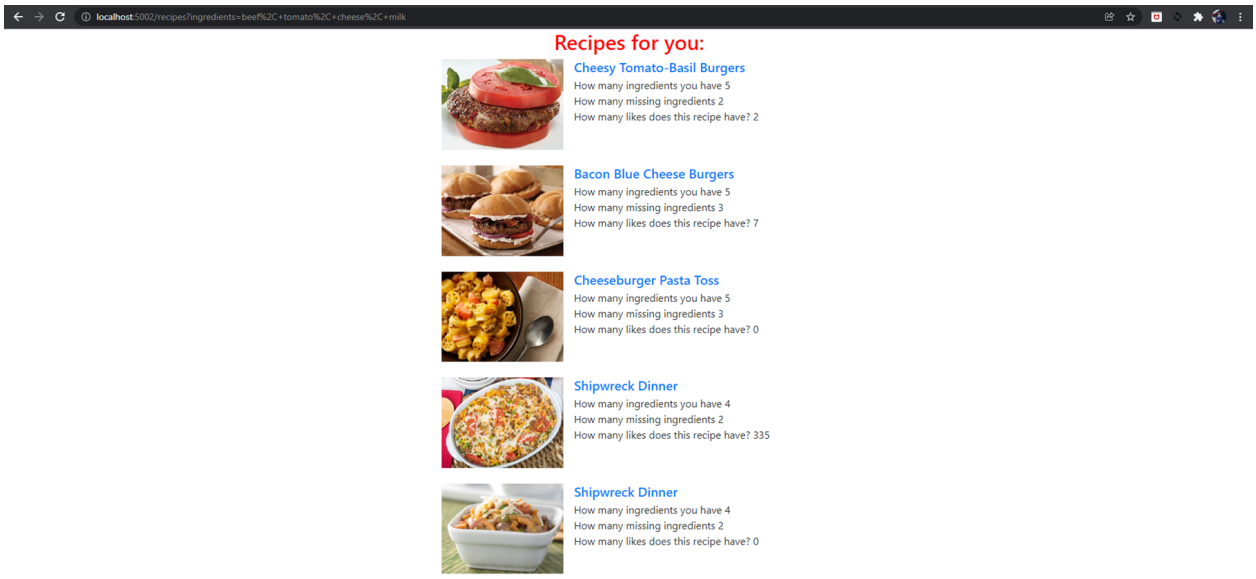


6. Demonstration

The app launch page is shown in the figure below. At the top, we coded the app to request a random joke from RapidAPI to spice things up before the user keys in the ingredients. The joke will be different each time the app refreshes. To start, we typed in “beef, tomato, cheese, milk” as a hypothetical list of ingredients that we have (you can add in more ingredients, but they must be comma separated).



Once we click “search”, the whole implemented design as shown in the previous section would kick in and return the top 5 results in a nicely formatted view as shown below.



The backend MongoDB stores will look like the following image below.

	_id ObjectId	id Int32	title String	image String	imageType String	usedIngredientCount Int32	missedIngredientCount Int32	missedIngredients
41	61b153a6f2d05c9eaacc5fc9	72922	"Homemade Butter Shortcrust Pas	"https://spoonacular.com/recipe	"jpg"	2	1	[] 1 elements
42	61b153a6f2d05c9eaacc5fca	802307	"Zucchini Bread Coffee Cake"	"https://spoonacular.com/recipe	"jpg"	2	2	[] 2 elements
43	61b153a6f2d05c9eaacc5fcb	38922	"Garlic Cheddar Soft Pretzel Di	"https://spoonacular.com/recipe	"jpg"	2	2	[] 2 elements
44	61b153a6f2d05c9eaacc5fcc	1058181	"Ferrero Rocher Christmas Cooki	"https://spoonacular.com/recipe	"jpg"	2	2	[] 2 elements
45	61b153a6f2d05c9eaacc5fcd	1071605	"3-Ingredient Buttermilk Biscui	"https://spoonacular.com/recipe	"jpg"	2	2	[] 2 elements
46	61b3a1988f66b79dc1bccc0a	274085	"Cheesy Tomato-Basil Burgers"	"https://spoonacular.com/recipe	"jpg"	5	2	[] 2 elements
47	61b3a1988f66b79dc1bccc0b	278233	"Bacon Blue Cheese Burgers"	"https://spoonacular.com/recipe	"jpg"	5	3	[] 3 elements
48	61b3a1988f66b79dc1bccc0c	173776	"Cheeseburger Pasta Toss"	"https://spoonacular.com/recipe	"jpg"	5	3	[] 3 elements
49	61b3a1988f66b79dc1bccc0d	269837	"Shipwreck Dinner"	"https://spoonacular.com/recipe	"jpg"	4	2	[] 2 elements
50	61b3a1988f66b79dc1bccc0e	100454	"Shipwreck Dinner"	"https://spoonacular.com/recipe	"png"	4	2	[] 2 elements
51	61b3a1b88f66b79dc1bccc0f	226222	"Migas de Arepa: A Hearty Break	"https://spoonacular.com/recipe	"jpg"	2	2	[] 2 elements
52	61b3a1b88f66b79dc1bccc10	513258	"Corn Casserole"	"https://spoonacular.com/recipe	"jpg"	2	2	[] 2 elements
53	61b3a1b88f66b79dc1bccc11	526063	"Cheesy Cornbread Pudding {}"	"https://spoonacular.com/recipe	"jpg"	2	3	[] 3 elements
54	61b3a1b88f66b79dc1bccc12	798699	"5-Ingredient Baked Eggs in Tom	"https://spoonacular.com/recipe	"jpg"	2	3	[] 3 elements
55	61b3a1b88f66b79dc1bccc13	267014	"Fresh Corn Custards"	"https://spoonacular.com/recipe	"jpg"	2	3	[] 3 elements
56	61b3a26a8f66b79dc1bccc14	274085	"Cheesy Tomato-Basil Burgers"	"https://spoonacular.com/recipe	"jpg"	5	2	[] 2 elements
57	61b3a26a8f66b79dc1bccc15	278233	"Bacon Blue Cheese Burgers"	"https://spoonacular.com/recipe	"jpg"	5	3	[] 3 elements
58	61b3a26a8f66b79dc1bccc16	173776	"Cheeseburger Pasta Toss"	"https://spoonacular.com/recipe	"jpg"	5	3	[] 3 elements
59	61b3a26a8f66b79dc1bccc17	269837	"Shipwreck Dinner"	"https://spoonacular.com/recipe	"jpg"	4	2	[] 2 elements
60	61b3a26a8f66b79dc1bccc18	100454	"Shipwreck Dinner"	"https://spoonacular.com/recipe	"png"	4	2	[] 2 elements

The Jupyter Notebook containing the codes and instructions have also been submitted along with this report.

7. Data Governance

We strive to ensure that data quality is maintained by defining responsibilities at each level of the team:

a. *Steering Committee*

   This is the Recipe's Generator's advisory body that will ensure delivery of the project goals and success measures (e.g., providing advice on budget, timelines and risks).

b. *Governance Council*

   A team who regulates and gives strategic guidance for the overall care of data (e.g., guiding the use of DBaaS to allow for resource elasticity, faster deployments and rapid provisioning).

c. *Lead Data Stewards*

   Functional group responsible for what is stored in the Recipe Generator's data fields (e.g., ensuring credit is given to the original recipe searches in our web app).

d. *Lead Data Custodians*

   Functional group responsible for the technical environments and database structure of the Recipe Generator (e.g., ensuring a dedicated application encryption layer)

Next, the five (5) categories where we will have policies in place so that internal accountability and external transparency is maintained:

a. *Privacy*

   We will create a legal data privacy document that lives on our website and details how personal data may be used.

b. *Access*

   We want to preserve the confidentiality and availability of our data resources. This policy will improve the ability of our team to properly manage data in compliance with Federal and State laws and regulations.

c. *Usage*

We will be public about which aspects of user data are private and which we are able to use and control.

d. *Integrity*

To maintain reliable and trustworthy data we will put the following checks in place: validate input, validate data, remove duplicate data, backup data, instill data access controls and keep an audit history.

e. *Licenses*

All packages produced will be licensed under the Apache License, Version 2.0, unless otherwise explicitly stated.

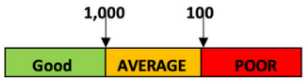
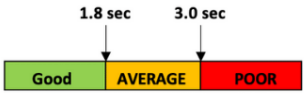
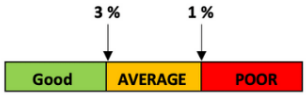
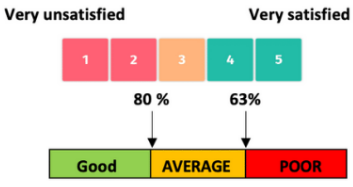
8. Cost Implications

Our project consists of recurrent and non-recurrent expenditures. The non-repetitive cost, around $5000, includes the Flask API development using Python Flask where we estimated a one (1) man-month worth of effort for developing the app and UI/UX design. As for the recurrent cost, approximately $3150/month, it consists of hosting the app on DigitalOcean ($95/month), MongoDB cloud ($57/month) which allows the database to run on a different server with a dedicated cloud storage, RapidAPI subscription ($999/month) consisting of 100,000 results/day, 200,000 “tiny requests”/day, 30,000 requests/day, maintenance of infrastructure ($1000/month) with a part-time data steward and a part-time business analyst to track metrics’ success ($1000/month).

S/N	Item	Cost	Details
1	Developing Flask API using Python Flask	$5,000	• Estimated as 1 man-month worth of effort for app development and UI/UX designer • Non-recurrent expenditure
2	Hosting app on DigitalOcean for about 50K unique users	$95/month	• App to run on DigitalOcean server
3	MongoDB cloud (DB-as-a-service)	$57/month	• DB to run on a different server • Dedicated MongoDB cloud storage
4	RapidAPI subscription	$999/month	• 100,000 results/day • 200,000 "tinyrequests"/day • 30,000 requests/day
5	Maintenance of infrastructure	$1000/month	• Estimate cost for data stewards
6	Tracking of success metrics	$1000/month	• Estimate for part-time business analysts

9. Criteria for Success

We have identified four (4) main areas to track the success of our app: traffic, performance, engagement and satisfaction. The success criteria are summarized in the table below.

Traffic	**Number of visits on the app** Traffic for startups	**1) Number of Visit per day** - Unique Visit ≥ 1,000: Good - 100 ≤ Unique Visit < 1,000: Average - Unique Visit < 100: Poor	
Performance	**Page loading time** 1) First Contentful Paint(FCP) 2) Largest Contentful Paint(LCP)	**1) FCP** - FCP time < 1.8 sec: Good - 1.8sec ≤ FCP time < 3.0 sec: Average - FCP time ≥ 3.0 sec: Poor **2) LCP** - LCP time < 2.5 sec: Good - 2.5sec ≤ LCP time < 4 sec: Average - LCP time ≥ 4.0 sec: Poor	 
Engagement	**Average rate of visitors who become subscribers** - Subscriber conversion rate: $\frac{\#\ of\ new\ subscribers}{total\ unique\ Visitors}$	**1) Subscriber conversion rate** - Conversion rate ≥ 2%: Good - 1% ≤ Conversion rate < 2%: Average - Conversion rate < 1%: Poor	
Satisfaction	**Customer Satisfaction Score(CSAT)** - CSAT for search engine $\frac{Total\ response\ scores\ Given}{Total\ possible\ Response\ scores}$	**1) Customer satisfaction score:** - CSAT ≥ 80%: Good - 63% ≤ CSAT < 80%: Average - CSAT < 63%: Poor	

10. Future Recommendations

We have been searching for recipes by ingredients only thus far. Next, we can expand the feature to include searching for recipes by allergies, nutrients, time to cook, etc. RapidAPI supports these searches, but we did not include these features in this proof of concept because of the limited number of requests we can make a day.

Also, we hope to develop a feature to automatically analyze the success/performance metrics so that we can have continuous improvement.

Last, we aim to release the completed work under Apache License 2.0 after successful implementation of the Minimum Viable Product (MVP) and safety of the entire architecture.

11. References

Food Allergies & Intolerances in the United States 2019. (n.d.). Statista. Retrieved October 25, 2021, from https://www-statista-com.ezproxy.cul.columbia.edu/forecasts/1093511/food-allergies-and-or-food-intolerances-in-the-us

Changing nutrition in the United States 2019. (n.d.). Statista. Retrieved October 27, 2021, from https://www-statista-com.ezproxy.cul.columbia.edu/forecasts/1093580/intention-to-change-own-nutrition-in-the-us

David. (n.d.). Recipe - Food - Nutrition API documentation (spoonacular). RapidAPI. Retrieved December 2021, from https://rapidapi.com/spoonacular/api/recipe-food-nutrition/.

Dyouri, Abdelhadi. “How to Create Your First Web Application Using Flask and Python 3.” DigitalOcean, DigitalOcean, 18 Aug. 2021, https://www.digitalocean.com/community/tutorials/how-to-create-your-first-web-application-using-flask-and-python-3

Personalized Nutrition Market Growth | Analysis, Trends & Forecasts | MarketsandMarkets. (n.d.). Retrieved October 25, 2021, from https://www.marketsandmarkets.com/Market-Reports/personalized-nutrition-market-249208030.html

Study: Most U.S. consumers to stick with eating at home post-pandemic. (2021, May 13). Supermarket News.

https://www.supermarketnews.com/consumer-trends/study-most-us-consumers-stick-eating-home-post-pandemic

Successful Website Statistics. (n.d.). Retrieved December 7, 2021, from https://www.mediacollege.com/internet/statistics/successful-sites.html

Improve Largest Contentful Paint (LCP) on Your Website With Ease. (2021, September 9). CSS-Tricks. https://css-tricks.com/improve-largest-contentful-paint-lcp-on-your-website-with-ease/

How Fast Should My Website Load? | Average Page Load Time. (n.d.). Retrieved December 7, 2021, from https://www.bluecorona.com/blog/how-fast-should-website-be/

What is a Good Conversion Rate and How to Improve It. (2020, August 10). Adoric Blog. https://adoric.com/blog/what-is-a-good-conversion-rate-2020/

Page Views Per Session. (n.d.). Retrieved December 7, 2021, from https://www.klipfolio.com/metrics/marketing/page-views-per-session

How to Attract Enough Visitors to Your Website to Earn a Living. (n.d.). Fizzle. Retrieved December 7, 2021, from https://fizzle.co/attract-enough-visitors-to-your-website-to-earn-a-living/

What Is a Good CSAT Score? (2021, January 29). MonkeyLearn Blog. https://monkeylearn.com/blog/what-is-a-good-csat-score/

Benchmarks By Industry. (n.d.). Retrieved December 7, 2021, from https://www.theacsi.org/index.php?option=com_content&view=article&id=148&Itemid=213