Ramen Ratings API

By Roy Ang royangkr@gmail.com

Contents

Data cleaning

ramen-ratings.csv had missing/invalid data.

Missing/Weird Package

ID	Country	Brand	Type	Package	Rating
2167	CHN	Brand G	E Menm Chicken	NaN	3.8
2351	JPN	Brand D	Chicken Instant Noodle	NaN	4.8
2352	JPN	Brand E	Mushroom Instant Noodle	NaN	4.5
2353	THA	Brand F	Beef and Mushroom Instant Noodle	NaN	2
2354	USA	Brand G	Kimchi Ramen	NaN	4
2355	JPN	Brand A	Shoyu Ramen	NaN	4.5
2356	KOR	Brand B	Shoyu Ramen	NaN	1.8
2457	TWN	Brand C	100 Furong Shrimp	NaN	3

Review 2167, 2351-2356, 2457 are missing data in the Package column. I try to extrapolate from previous data.
Ideally, if the Country, Brand and Type matches, I would be able accurately extrapolate the missing Package. However, for each review missing Package, there was no other review that matched its Country, Brand and Type.
I decided that if the Brand and Type is the same, the package should probably be the same.

ID	Country	Brand	Type	Package	Rating
1268	CHN	Brand A	Shoyu Ramen	Pack	3
2355	JPN	Brand A	Shoyu Ramen	NaN	4.5
2441	USA	Brand A	Shoyu Ramen	Cup	1.8
2455	THA	Brand A	Shoyu Ramen	Pack	5
2481	KOR	Brand A	Shoyu Ramen	Pack	2.9
2508	THA	Brand A	Shoyu Ramen	Pack	2.8
2561	USA	Brand A	Shoyu Ramen	Pack	2
2601	USA	Brand A	Shoyu Ramen	Pack	2

> For example, most review that had "Brand A" and "Shoyu Ramen" were of "Pack" so I extrapolate that the Package for review 2355 is "Pack". For the reviews with missing Package, I extrapolate if confidence level is >80%. I could do so for 5/8 reviews that were missing Package. I also tried matching Country and Brand, and just Brand, but could not get >80% confidence so I still have 3 reviews with missing Package.

Review 82 has Package "Can" and Review 1440 has Package "Bar", which each occur once and does not seem like an applicable unit for ramen. I tried the above extrapolation but was not successful, so I left them as Can and Bar.

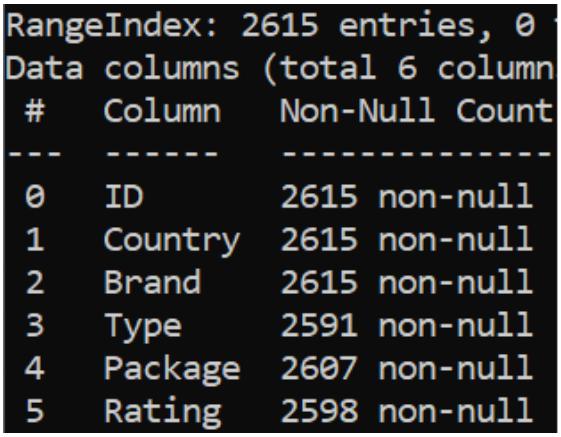
Missing Type

Reviews 2482-2494,2595-2606 are missing data in the Type column
I tried to extrapolate the Type for reviews missing Type, but was not successful because Type is not repeated much across the dataset.

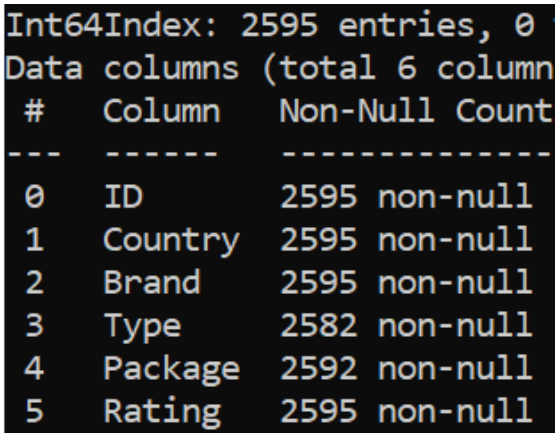
Missing Rating

Reviews 47,137,1008 have "#VALUE!" in the Rating column. Reviews 2430-2435,2595-2606 are missing data in the Rating column.
Lastly, I decided to remove all reviews that have missing/invalid rating, because this is supposed to be a ramen ratings database, the review is meaningless without a proper rating.

Result from cleaning



```
RangeIndex: 2615 entries, 0
Data columns (total 6 column
 #   Column   Non-Null Count
---  ------   --------------
 0   ID       2615 non-null
 1   Country  2615 non-null
 2   Brand    2615 non-null
 3   Type     2591 non-null
 4   Package  2607 non-null
 5   Rating   2598 non-null
```



```
Int64Index: 2595 entries, 0
Data columns (total 6 column
 #   Column   Non-Null Count
---  ------   --------------
 0   ID       2595 non-null
 1   Country  2595 non-null
 2   Brand    2595 non-null
 3   Type     2582 non-null
 4   Package  2592 non-null
 5   Rating   2595 non-null
```

Because I dropped reviews with missing/invalid ratings, I am down to 2595 reviews from 2615. I have 13 reviews missing Type, down from 24. I have 3 reviews missing Package, down from 8.

Design considerations

I want all the data provided in the sample dataset to be included in the database. As such, I assume all the rows in the sample dataset are valid i.e.

- ID need not be unique, but must not be empty or NULL
- Country must be found in the ISO 3166-1
- Type can be NULL, but must not be empty
- Package can be NULL, but must not be empty
- Rating must be between 0.0-5.0

Importantly, I decided to not use ID as the primary key and instead use the unique rowid. This gave me the opportunity to use the ID as the “ID/token of the reviewer”. When submitting reviews, the reviewer is asked to include an ID. If someone wants to modify/delete the review, he has to enter the correct ID.

(Note: some reviews imported from the sample dataset will have the same ID and rowid. If you think this defeats the purpose of using the ID as a secret token, you can uncomment line 38 of init_db.py "random.shuffle(to_db)", but when you want to delete any of them, you would have to manually search for their ID in ramen-ratings.csv :laughing:)

Installation for Windows

Install Python 3
Enter the project directory, create a virtual env and enter it

```
cd ramen_ratings
py -m pip install --upgrade pip
py -m pip install --user virtualenv
py -m venv env
.\env\Scripts\activate
```

Install dependencies

```
py -m pip install -r requirements.txt
```

Running the server

```
python init_db.py
set FLASK_APP=app.py
set FLASK_ENVIRONMENT=development
flask run
```

On the use of curl and Postman

For the rest of this documentation, I will be demonstrating using Windows Command Prompt and Postman. The commands will work in Windows Powershell if use curl.exe instead of curl and --% to stop parsing as powershell commands:

```
curl.exe --%
```

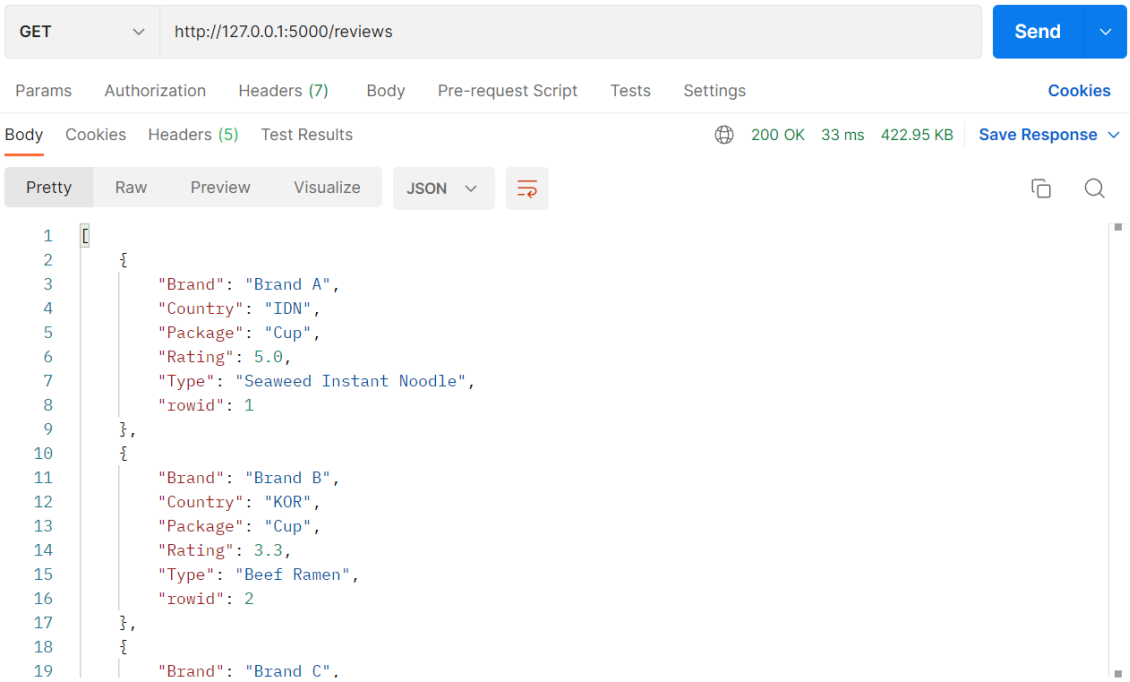
List all reviews

Endpoint: `http://127.0.0.1:5000/reviews`, HTTP Method: GET Since I am using ID as a token, I hide it when listing reviews and show rowid instead.

On Windows Command Prompt

```
curl -i http://127.0.0.1:5000/reviews
```

On Postman



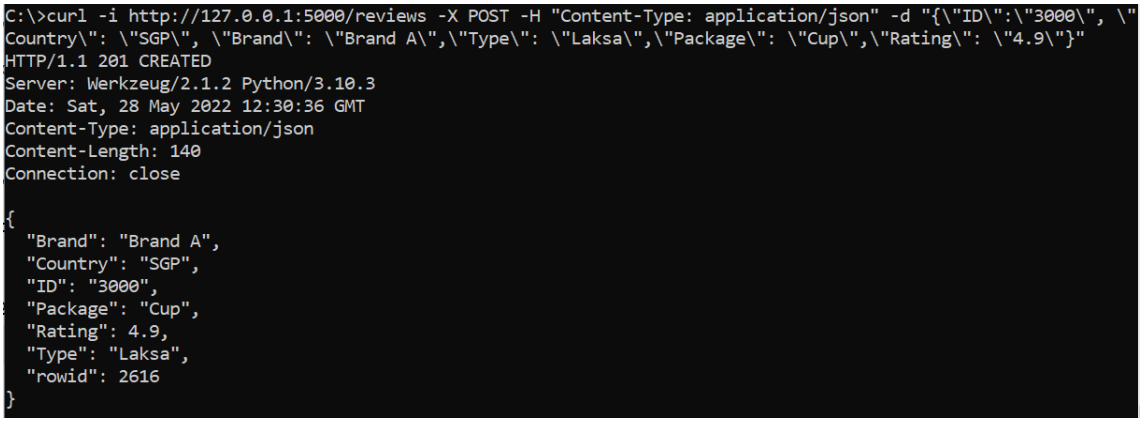
Create a new review

Endpoint: `http://127.0.0.1:5000/reviews`, HTTP Method: POST
ID, Country and Brand are required and cannot be empty. Country must be found in the ISO 3166-1
To create a review with the following values | ID | Country | Brand | Type | Package | Rating | | ------ | ------ | ------ | ------ | ------ | ------ | | 3000 | SGP | Brand A | Laksa | Cup | 4.9 |

On Postman, send POST to `http://127.0.0.1:5000/reviews` and use the following as Body input: `{"ID":"3000","Country":"SGP","Brand":"Brand A","Type":"Laksa","Package":"Cup","Rating":4.9}`

On Windows Command Prompt

```
curl -i http://127.0.0.1:5000/reviews -X POST -H "Content-Type: application/json" -d "
{\"ID\":\"3000\", \"Country\": \"SGP\", \"Brand\": \"Brand A\",\"Type\":
\"Laksa\",\"Package\": \"Cup\",\"Rating\": \"4.9\"}"
```



Get specific review

Endpoint: `http://127.0.0.1:5000/reviews/<rowid>`, HTTP Method: GET

On Postman, send GET to `http://127.0.0.1:5000/reviews/1`

On Windows Command Prompt

```
curl -i http://127.0.0.1:5000/reviews/1
```

Modify review

Endpoint: `http://127.0.0.1:5000/reviews/<rowid>`, HTTP Method: PUT or PATCH
ID must match the one used when review was created.

PUT (replaces all columns)

Country and Brand are required and cannot be empty. [Country must be found in the ISO 3166-1]

On Postman, send PUT to `http://127.0.0.1:5000/reviews/1` and use the following as Body input: `{“ID”:“1”,“Country”:“SGP”,“Brand”:“Brand A”,“Type”:“Laksa”,“Package”:“Cup”,“Rating”:4.9}`

On Windows Command Prompt

```
curl -i http://127.0.0.1:5000/reviews/1 -X PUT -H "Content-Type: application/json" -d "{\"ID\":\"1\", \"Country\": \"SGP\", \"Brand\": \"Brand A\",\"Type\": \"Laksa\",\"Package\": \"Cup\",\"Rating\": \"4.9\"}"
```

```
C:\>curl -i http://127.0.0.1:5000/reviews/1 -X PUT -H "Content-Type: application/json" -d "{\"ID\":\"1\", \"Co
untry\": \"SGP\", \"Brand\": \"Brand A\",\"Type\": \"Laksa\",\"Package\": \"Cup\",\"Rating\": \"4.9\"}"
HTTP/1.1 200 OK
Server: Werkzeug/2.1.2 Python/3.10.3
Date: Sat, 28 May 2022 12:34:46 GMT
Content-Type: application/json
Content-Length: 134
Connection: close

{
  "Brand": "Brand A",
  "Country": "SGP",
  "ID": "1",
  "Package": "Cup",
  "Rating": 4.9,
  "Type": "Laksa",
  "rowid": 1
}
```

PATCH (changes some columns)

Country and Brand are not required but cannot be empty. [Country must be found in the ISO 3166-1]

On Postman, send PATCH to `http://127.0.0.1:5000/reviews/1` and use the following as Body input: `{“ID”:“1”,“Rating”:4.8}`

On Windows Command Prompt

```
curl -i http://127.0.0.1:5000/reviews/1 -X PATCH -H "Content-Type: application/json" -
d "{\"ID\":\"1\",\"Rating\": \"4.8\"}"
```

```
C:\>curl -i http://127.0.0.1:5000/reviews/1 -X PATCH -H "Content-Type: application/json" -d "{\"ID\":\"1\",\"R
ating\": \"4.8\"}"
HTTP/1.1 200 OK
Server: Werkzeug/2.1.2 Python/3.10.3
Date: Sat, 28 May 2022 12:44:14 GMT
Content-Type: application/json
Content-Length: 134
Connection: close

{
  "Brand": "Brand A",
  "Country": "SGP",
  "ID": "1",
  "Package": "Cup",
  "Rating": 4.8,
  "Type": "Laksa",
  "rowid": 1
}
```

Delete review

Endpoint: `http://127.0.0.1:5000/reviews/<rowid>`, HTTP Method: DELETE
ID must match the one used when review was created.

On Postman, send DELETE to `http://127.0.0.1:5000/reviews/1` and use the following as Body input: `{“ID”:“1”}`

On Windows Command Prompt

```
curl -i http://127.0.0.1:5000/reviews/1 -X DELETE -H "Content-Type: application/json"
-d "{\"ID\":\"1\"}"
```

```
C:\>curl -i http://127.0.0.1:5000/reviews/1 -X DELETE -H "Content-Type: application/json" -d "{\"ID\":\"1\"}"
HTTP/1.1 200 OK
Server: Werkzeug/2.1.2 Python/3.10.3
Date: Sat, 28 May 2022 14:22:50 GMT
Content-Type: application/json
Content-Length: 3
Connection: close

{}
```

Get a list of ramen reviews filtered by a country

Endpoint: `http://127.0.0.1:5000/reviews?countr=<country>`, HTTP Method: GET

Restriction on Country value

When reviews are created, the Country value must be found in the ISO 3166-1, a standard defining codes for the names of countries.
For example, Singapore in the ISO 3166-1 is `Country(name='Singapore', alpha2='SG', alpha3='SGP', numeric='702')`. The name or codes are accepted and when filtered by country, any row with County that matches name or code (no need to match case) will be shown.

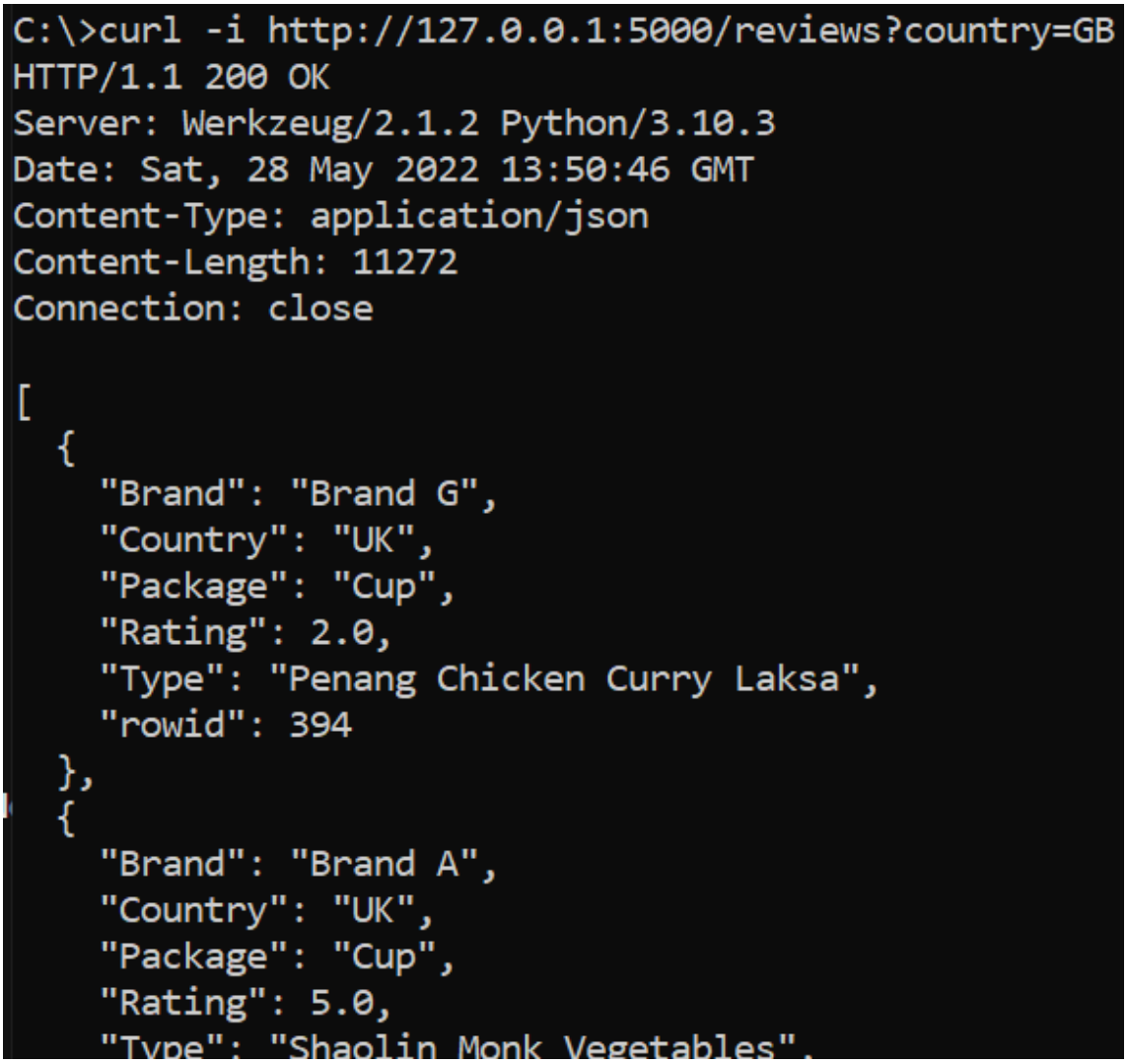
- Accepted values: `SGP, sgp, sGp, Singapore, siNgapore, SG, sg, 702` etc
- Unaccepted/Wrong values: `Singapo, ingapore, s, gp(points to Guadeloupe)` etc

Notably, in the sample dataset, only Country "UK" does not match any of the countries in ISO 3166-1. That is becuase United Kingdom is `Country(name='United Kingdom of Great Britain and Northern Ireland', alpha2='GB', alpha3='GBR', numeric='826')`. I manually added UK as an acceptable value of the same country as GB.

On Postman, send GET to `http://127.0.0.1:5000/reviews?country=GB`

On Windows Command Prompt

```
curl -i http://127.0.0.1:5000/reviews?country=GB
```



Get a list of ramen reviews based on a partial text

Endpoint: `http://127.0.0.1:5000/reviews?q=<partial text>`, HTTP Method: GET

On Postman, send GET to `http://127.0.0.1:5000/reviews?q=Seaweed`

On Windows Command Prompt

```
curl -i http://127.0.0.1:5000/reviews?q=Seaweed
```

```
C:\>curl -i http://127.0.0.1:5000/reviews?q=Seaweed
HTTP/1.1 200 OK
Server: Werkzeug/2.1.2 Python/3.10.3
Date: Sat, 28 May 2022 13:52:54 GMT
Content-Type: application/json
Content-Length: 2616
Connection: close

[
  {
    "Brand": "Brand C",
    "Country": "TWN",
    "Package": "Pack",
    "Rating": 3.5,
    "Type": "Noodle Snack Wheat Cracks Seaweed Flavor",
    "rowid": 123
  },
  {
    "Brand": "Brand F",
    "Country": "KOR",
    "Package": "Cup",
    "Rating": 0.5,
    "Type": "Seaweed Instant Noodle"
```