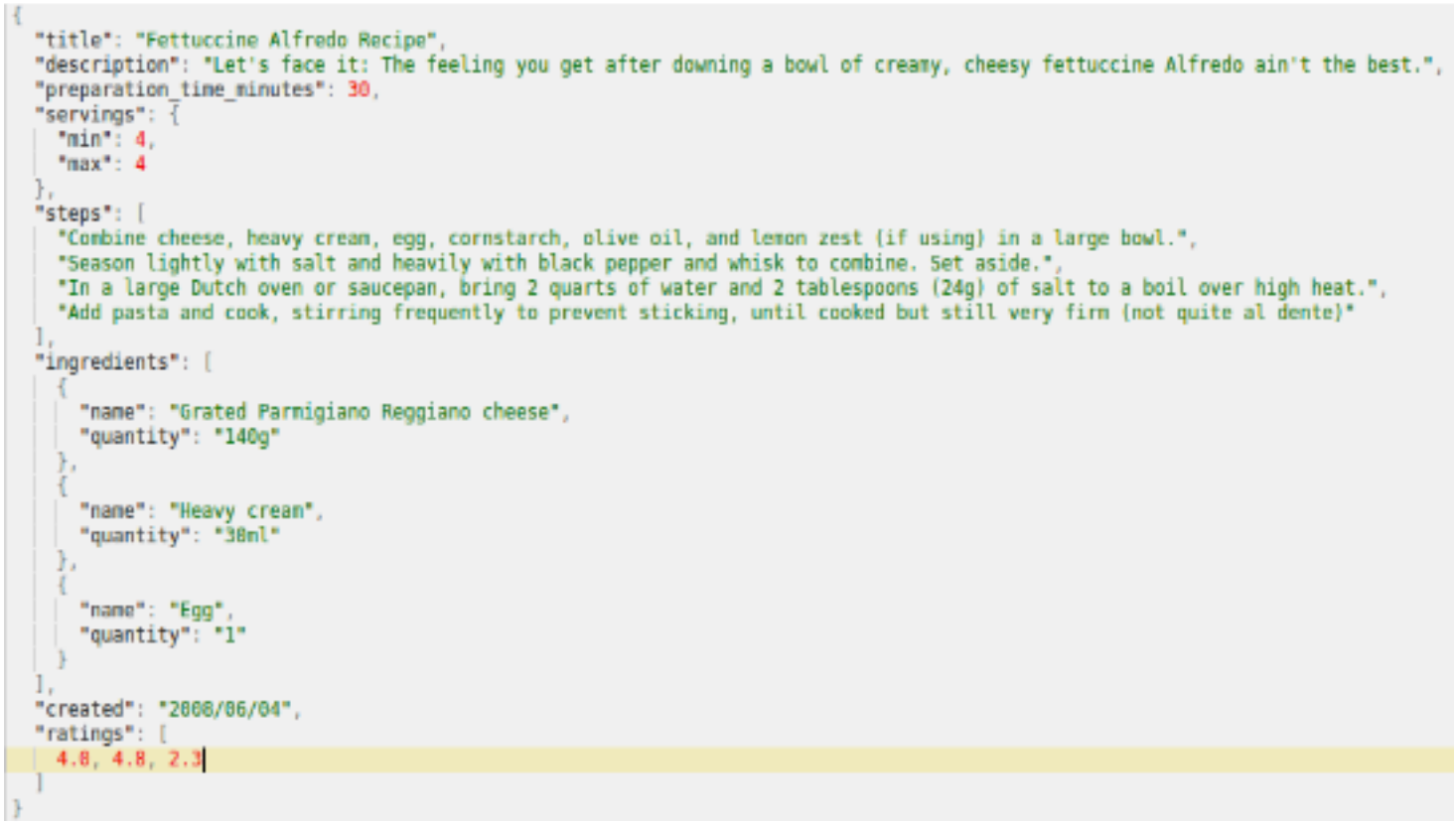
MAJOR ASSESSMENT

Related to recipe.json

1) Look at the sample json data in the picture below.




```
{
  "title": "Fettuccine Alfredo Recipe",
  "description": "Let's face it: The feeling you get after downing a bowl of creamy, cheesy fettuccine Alfredo ain't the best.",
  "preparation_time_minutes": 30,
  "servings": {
    "min": 4,
    "max": 4
  },
  "steps": [
    "Combine cheese, heavy cream, egg, cornstarch, olive oil, and lemon zest (if using) in a large bowl.",
    "Season lightly with salt and heavily with black pepper and whisk to combine. Set aside.",
    "In a large Dutch oven or saucepan, bring 2 quarts of water and 2 tablespoons (24g) of salt to a boil over high heat.",
    "Add pasta and cook, stirring frequently to prevent sticking, until cooked but still very firm (not quite al dente)"
  ],
  "ingredients": [
    {
      "name": "Grated Parmigiano Reggiano cheese",
      "quantity": "140g"
    },
    {
      "name": "Heavy cream",
      "quantity": "38ml"
    },
    {
      "name": "Egg",
      "quantity": "1"
    }
  ],
  "created": "2008/06/04",
  "ratings": [
    4.8, 4.8, 2.3
  ]
}
```


Picture I: Sample JSON Data

Create an **index with any name** that you prefer (example: recipes) along with **mapping**. When creating mapping:

- decide on what are the fields required and their respective types that should be available in the mapping of the index by looking at the sample json data in the picture above (Picture I: Sample JSON Data).
- create **custom analyzer** (you can provide it any name, example: new_analyzer) that can handle english stop-words and synonyms available in a file created inside the config directory of elasticsearch (example, synonyms_recipes.txt). In order to add synonyms, you can download the file in **this link**, open it in notepad or any other text editor to see what kind of synonyms can be created and add the synonyms that you have thought of in synonyms_recipes.txt file following proper format. The custom analyzer should be provided to **description** and **steps** fields.

2) **Bulk insert** the data downloaded from **this link** into the newly created index. The data has different recipes as documents.

3) For documents having preparation_time_minutes of less than of equal to 15, the difference between min and max servings should be at least 1 and at most 3 (i.e. the difference between min and max servings should be between 1-3, inclusive). Write update by query using script to make sure that this case prevails.

4) Search for all the documents in the index. View the nested field **ingredients** of the documents. You can see that in some documents, some ingredients are missing quantity field. First, find the count of such documents. Then, update such documents in which at least one of the ingredients is missing quantity field by adding the quantity field with value 'Per Choice'.
Hint: ingredients is a nested field (array of JSON objects), so you need to perform nested query along with update by query and script to do this question.
Side Note: First perform nested query to see how many such documents exist. After executing nested query, you can see that each document/hit that is returned has at least one JSON object that match the nested query condition in the array of JSON objects of the nested field. Only then, try performing update by query. Following this approach helps you to verify your work properly.

5) Delete unrated documents, that is, documents that have empty array in ratings field.

6) Find all the recipes that use Egg as one of the ingredients. Display only title, ratings, steps, number of steps. Note that number of steps is a derived field. The documents should be ordered by average rating.

7) Execute at least 10 different aggregations for performing analysis on the data.

- You need to prepare at least 3 different metric aggregations
- You need to prepare at least 5 different bucket aggregations
- You need to prepare at least 2 different sub aggregations
    - At least 1 sub aggregation having metric aggregation within bucket aggregation
    - At least 1 sub aggregation having bucket aggregation within another bucket aggregation

8) Design your own search using compound bool query. The bool query should have at least 2 musts, at least 2 filters and at least 1 should. Showcase the use of synonyms, proximity (slop) and fuzziness parameters where possible in your search query.

9) Practice using cut-off frequency to handle domain specific stop-words in match query and common-terms query. You can make use of **steps** field for this purpose.

Terms Lookup Mechanism and Geo Queries

10)

A. Create an index **items** with mapping having following fields:
   item_id: integer field
   name: text field
   stock: integer field
   vendor: object with properties name (text field), contact (keyword), address(geo_point)

B. Bulk insert at least 5 documents into **items** index.

C. Try performing geo bounding box and geo distance queries in the vendor's address.

D. Create an index **category_items** with mapping having following fields:
   category: text field
   Items: array of item_ids, that means, array of integer

E. Insert two documents in category_items: one related to cosmetic category and next related to household category. When inserting documents in category_items, please note that each category should have at least one item_id inserted in items index.

F. Using terms query with terms lookup mechanism, find the items from items index that belong to cosmetic category.

Related to orders-bulk.json

*Note: I had told everybody to bulk insert orders-bulk.json into an index **orders** in class. So, I hope everybody has already created that index with mapping as instructed in class, and bulk inserted orders-bulk.json into it. If not, please do so.*

11) Create filtered alias of documents of orders index fulfilling the following conditions:

- status: processed **or** completed
- sales_channel: phone **and** app
- Having total_amount >=100

12) Perform the following queries in the filtered alias. Design your search conditions yourself.

- Term query
- Range query
- Prefix query
- Wildcard Query

- Match
- Fuzzy Match

Creativity Check

- Create an index of your own choice with mapping having fields of different types. Consider using all the types that we have learnt so far. Prefer creating analyzer and try using synonyms and stop-words filters in the analyzer.
- Bulk insert at least 10 documents creating a json file yourself and by using curl.
- Perform as many different searches and aggregations as you would like to for analyzing various aspects of your data.