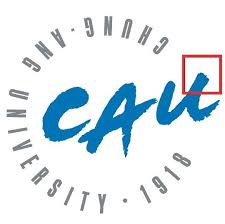
**NLP Project**

**Report**

Team Name: 자연어버억

Team leader: 박주현

|  |  |
| --- | --- |
| 학번 | 이름 |
| 20163704 | 박주현 |
| 20164245 | 홍진 |
| 20140313 | 양인수 |
| 20140490 | 이동규 |
| 20142748 | 이강희 |
| 20176058 | 박시현 |

**Table of Contents**

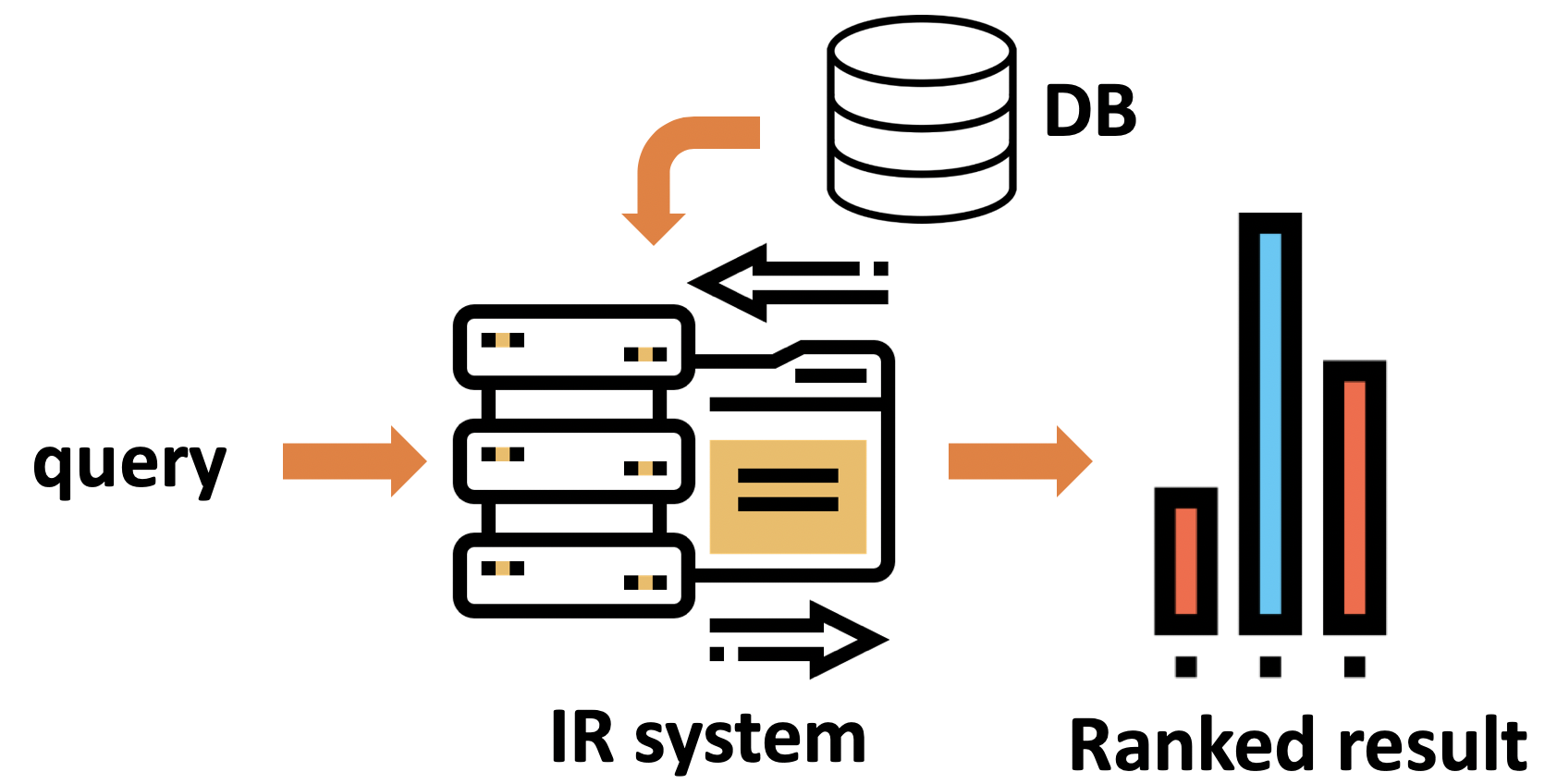
1. **Introduction**
2. **Workspace Environment**
3. **Implementation**
4. **Evaluation**
5. **Introduction**
6. **Project Topic**

The problem with traditional recipes is that if the user wishes to find out information about one particular dish, without knowing the name of the food, it is impossible to satisfy any user needs. For example, knowing the ingredients and the descriptions of the food does not help in finding out the information of the dish, since the title is unknown.

Our project’s objective is to build a recipe search system that recognizes natural language queries.

1. **Project Objective**
2. Implement a natural language recognizing recipe search IR system.
3. Implement a Boolean model recognizing ingredients
4. Make a web interface for user inquiries.
5. **Workspace Environment**
6. Github: For project management
7. Python: All the models are implemented in Python language.
8. Django: Web environment implemented by Django
9. Jupyter Notebook: Python workspace
10. Microsoft Excel: database files were saved in xlsx format.
11. **Implementation**

**SUB. Overall Project Structure**



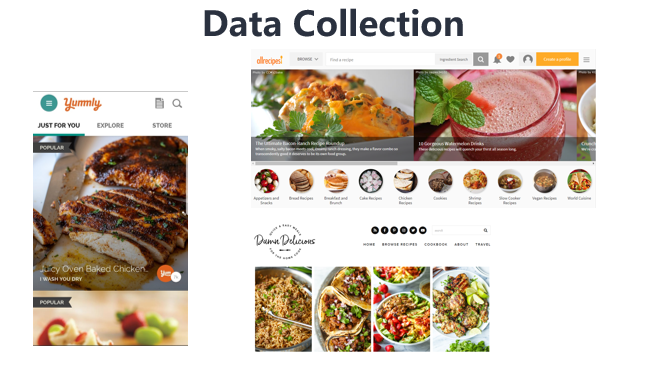
The overall project flow is as follows:

1. User inputs query.
2. IR system compares the query with the existing database.
3. IR System returns top matching results to the user.

However, there are several detailed processes inside each step in order for the IR system to recognize natural language queries. We have solved this problem by preprocessing queries in a special way, and using a language model (Dirichlet Smoothing Model). We will explain further details in the following sections.

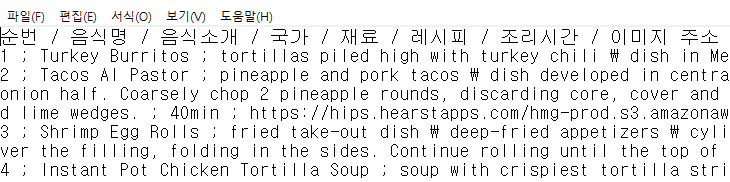
1. **Data Mining**

For data collection, we gathered data from open sources (Yummly, Damn Delicious, Allrecipes).



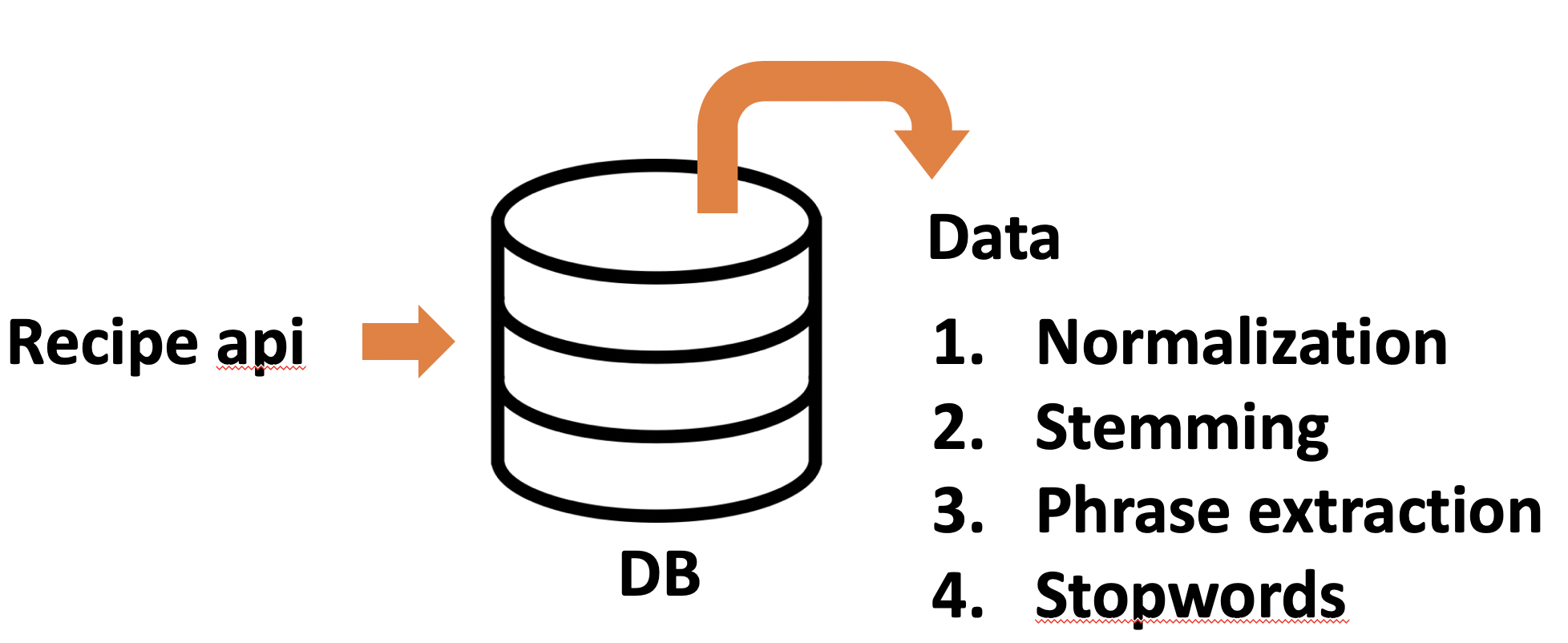
The data schema we collected is as follows.

We collected 100 number of dishes, from 11 countries.



For efficient usage and parsing, we categorized data by using ‘;” character between each data description.

1. **Data Processing**

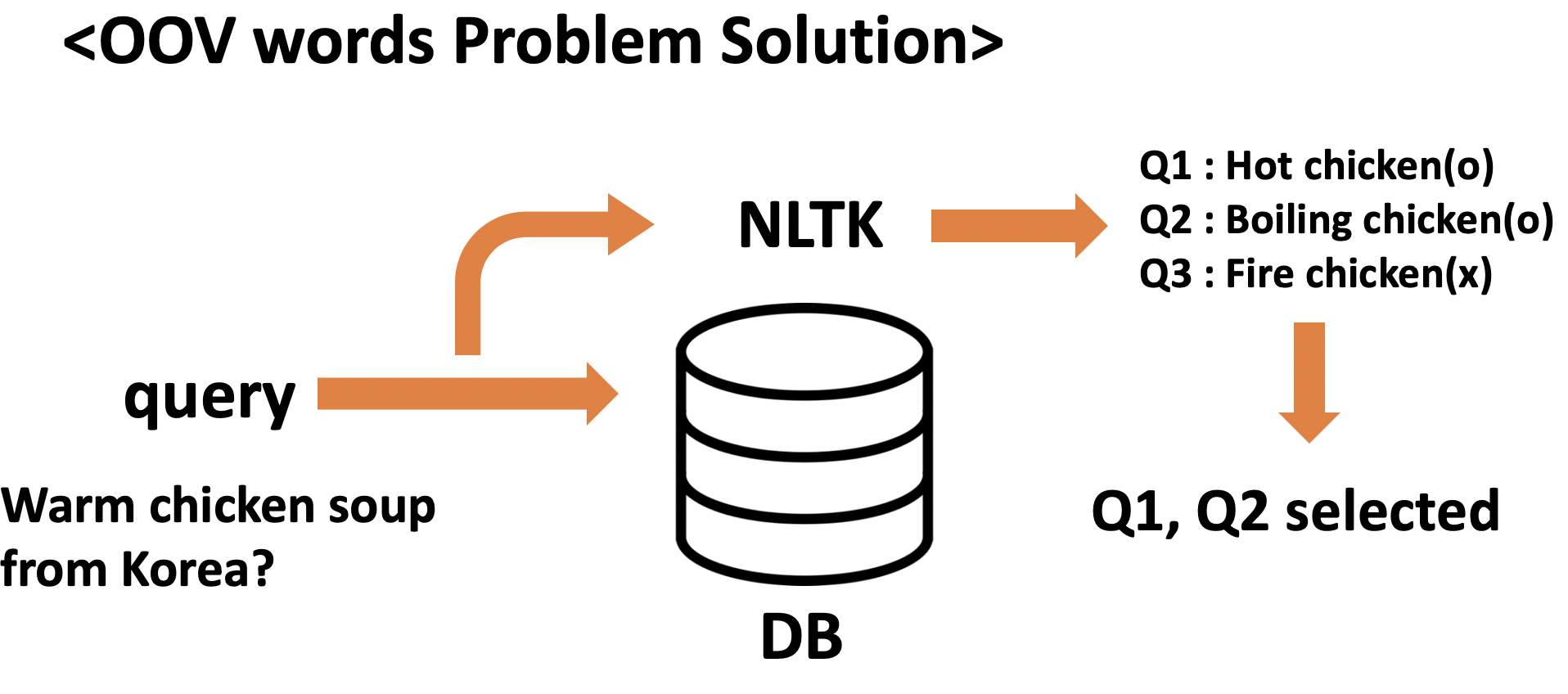


We processed the collected database in order to match with query. The DB underwent 4 steps of processing:

1. Normalization: this process corrects typos.
2. Stemming: transforms raw vocabulary to its morpheme
3. Phase extraction: extracts useful vocabulary
4. Stopwords: erases useless vocabularies.
5. **Query Processing**

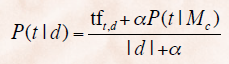
One of the major problem facing natural language query is that if words in the query do not exist in the existing database. This is called the OOV problem. We solved this OOV problem by using NLTK library, which returns a list of synonyms of the input vocabulary.

1. If user inquires a query containing a word that does not exist in the database,
2. Preprocessor makes a list of synonyms of the non-existing word
3. Preprocessor searches for matches between list of synonyms and database.
4. If matches are found, preprocessor makes 10 random queries using the matched vocabularies. As a result, the new 10 queries are all composed of words existing in the database.
5. If no matches are found, the unknown vocabulary is disregarded.



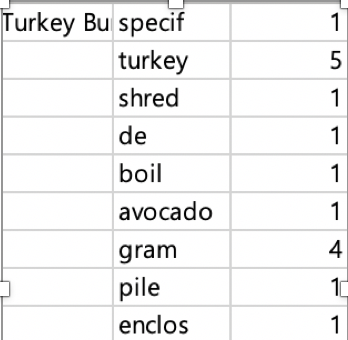
1. **Modeling**

We made our IR system as a language model, using Dirichlet Smoothing.

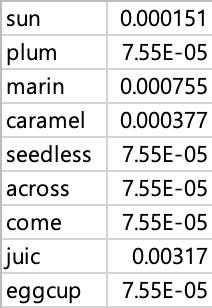


We calculated each components of the P(t|d).

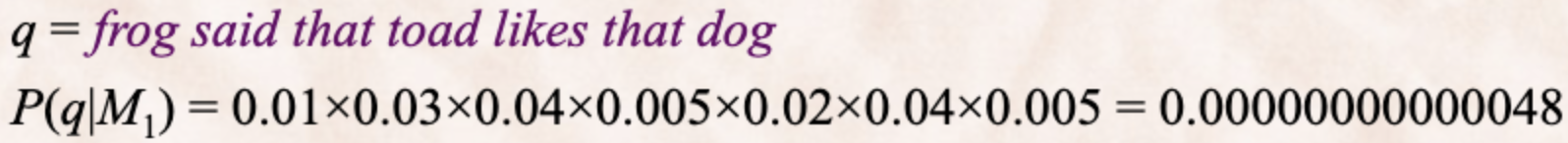
1. Tf: we calculated the tf same as the previous VSM models.



1. P(t|Mc): similar to df



1. alpha: default value was 0.5, but after evaluation, this constant was modified where the best results were made.
2. **Matching**

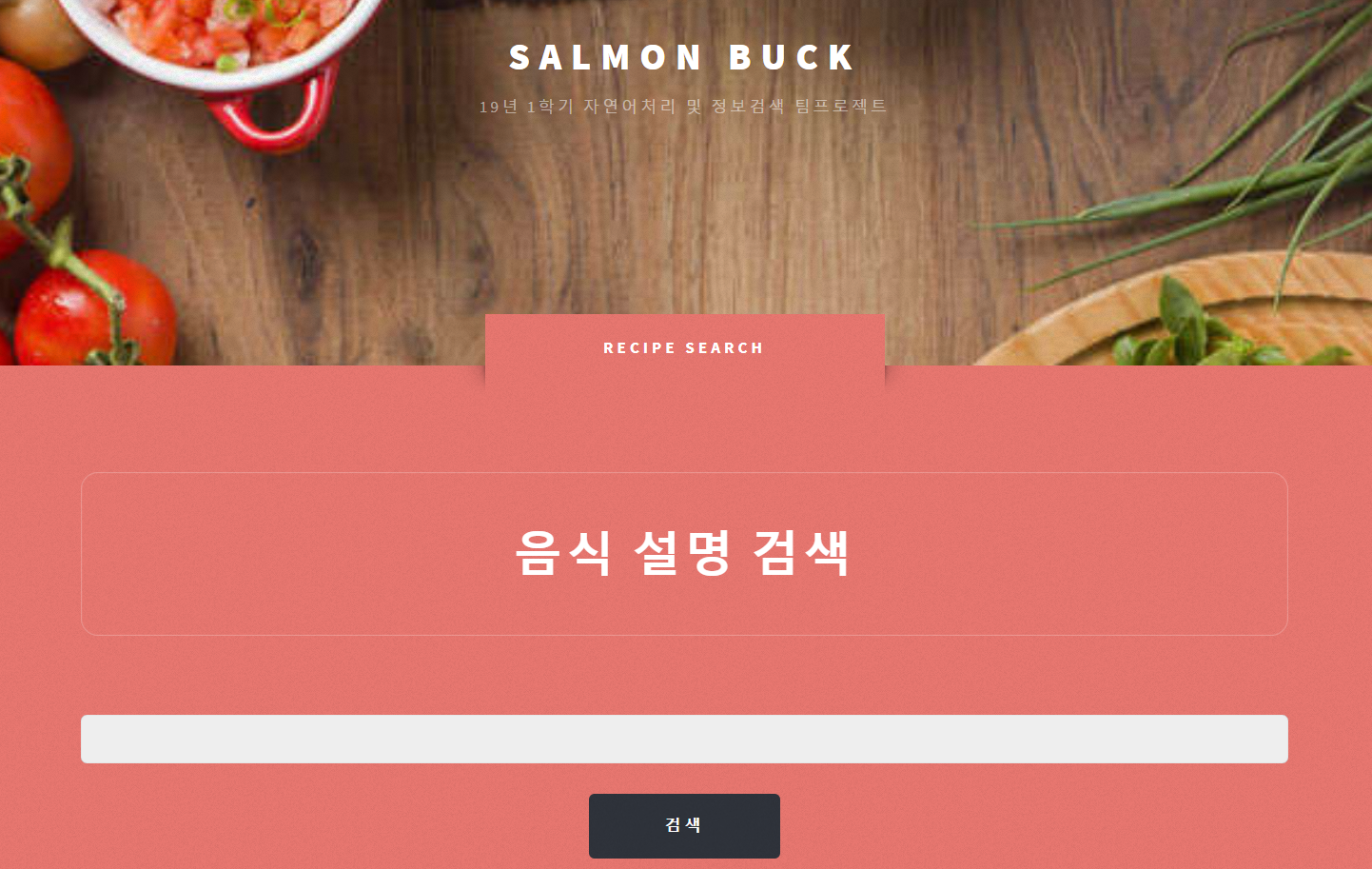
We calculated the total probability by the following method:

1. **Web Interface**



We implemented the web page using python-based Django web framework. The following web page is composed of 2 pages, and supports 2 basic features.

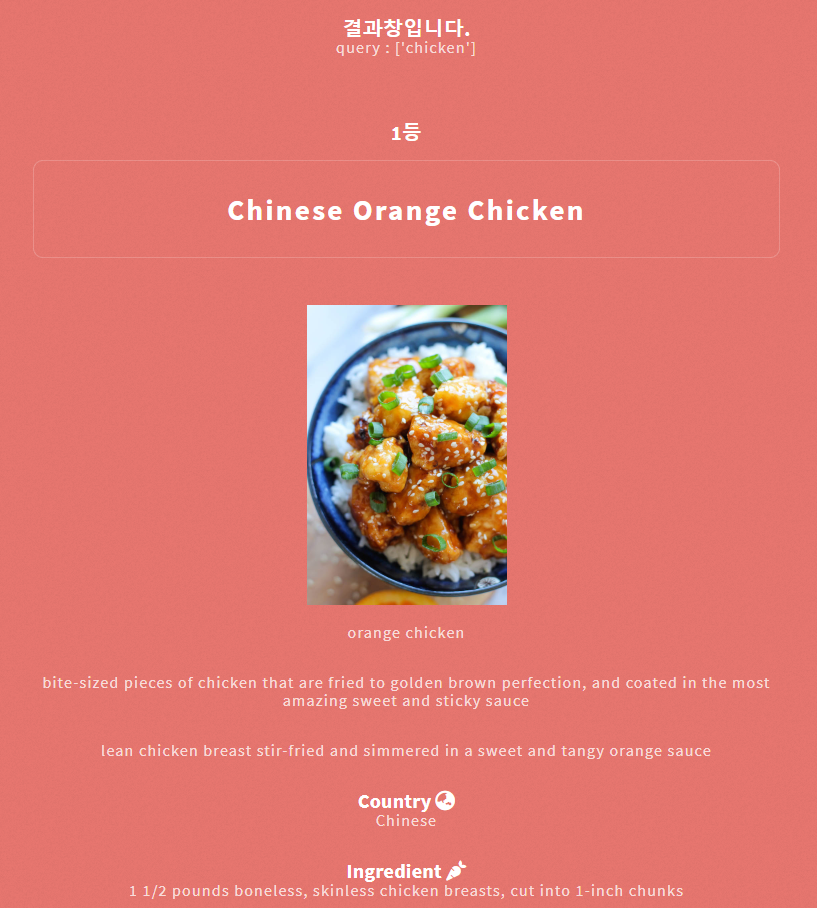
1. Main Page



The main web page supports 2 search features. The first is searching food by any natural language, and the other searching by ingredients.



1. Result Page



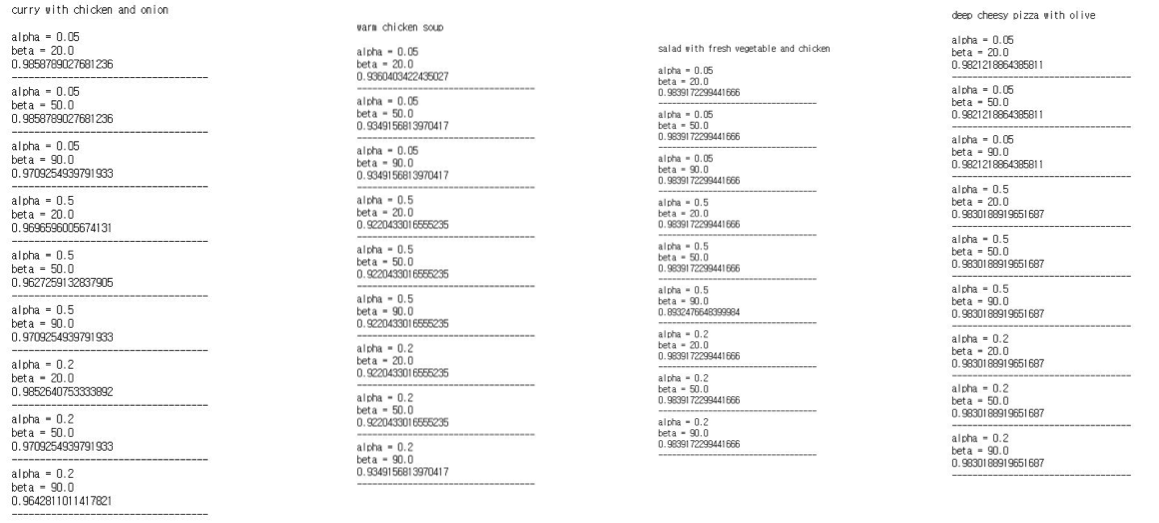
The first entity shows the input query. The search result is shown by the top 5 ranking results.

After matching, the web shows the matching result’s representative components in a user-friendly format.

1. Boolean Model

We implemented a Boolean model as an additional feature. This works by searching queries by the ingredients, and the IR system retrieves the top matches according to the requested ingredients.

1. **Evaluation**
2. NDCG

We evaluated our IR system by making 20 test queries of the IDCG. We used NDCG system to score the relevance of each documents per test query.