



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

공학석사 학위논문

**OCR-Based Safety Check System  
of Packaged Food for Food Inconvenience Patients**

식품 불내성 환자를 위한 포장 식품의 OCR 기반  
안전확인 시스템

지도교수 신 옥 근

2020년 2월

한국해양대학교 대학원

컴퓨터공학과

**NURUL AZZAHRA PUTRI KAMIS**

본 논문을 NURUL AZZAHRA PUTRI KAMIS 의  
공학석사 인준함.

위원장 공학박사 박 휴 찬



위 원 공학박사 이 장 세



위 원 공학박사 신 옥 근



2019 년 12 월 26 일

한국해양대학교 대학원

# Contents

Abstract .....	iv
List of Abbreviations .....	vi
List of Tables.....	vii
List of Figures .....	viii
 <b>Chapter I: Introduction</b>	
1.1 Background of Research .....	1
1.2 Research Objectives.....	2
 <b>Chapter II: Literature Review</b>	
2.1 Review of Research Topics.....	4
2.1.1 Review of Optical Character Recognition (OCR) .....	4
2.1.2 Review of The Tesseract OCR Engine .....	6
2.1.3 Review of OCR Post-Processing .....	8

<b>2.1.4</b>	Review of Food Intolerance, Allergies and Auto-immune Diseases.....	9
<b>2.2</b>	Review of Related Work.....	10
<b>2.2.1</b>	Eatable.....	10
<b>2.2.2</b>	Food Allergy Scanner .....	12

### **Chapter III: System Design**

<b>3.1</b>	Overall Architecture of the System.....	14
<b>3.2</b>	Get Image of Product’s Ingredients.....	16
<b>3.2.1</b>	Client-Server Architecture .....	16
<b>3.3</b>	Perform OCR.....	17
<b>3.3.1</b>	OCR and Its Pre-Processing .....	18
<b>3.4</b>	Post-Processing of OCR.....	20
<b>3.4.1</b>	Tokenization .....	21
<b>3.4.2</b>	Extract Correct Ingredients Using Dictionaries .....	23
<b>3.5</b>	Search Harmful Ingredients for The User using Database .....	25
<b>3.6</b>	Notify Result to the user .....	27

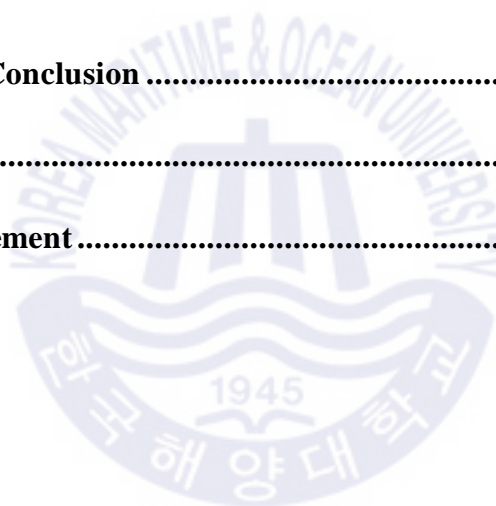
## **Chapter IV: System Implementation**

<b>4.1 Overall Explanation</b> .....	28
<b>4.1.1 Pre-Processing of OCR</b> .....	31
<b>4.1.2 Post-Processing of OCR</b> .....	32
<b>4.2 Database of the System</b> .....	36
<b>4.3 System Prototype using Android Studio</b> .....	39

<b>Chapter V: Conclusion</b> .....	42
------------------------------------	----

<b>References</b> .....	44
-------------------------	----

<b>Acknowledgement</b> .....	50
------------------------------	----



# **OCR-Based Safety Check System of Packaged Food for Food Inconvenience Patients**

*Nurul Azzahra Putri Kamis*

*Department of Computer Engineering,*

*Graduate School of Korea Maritime and Ocean University*

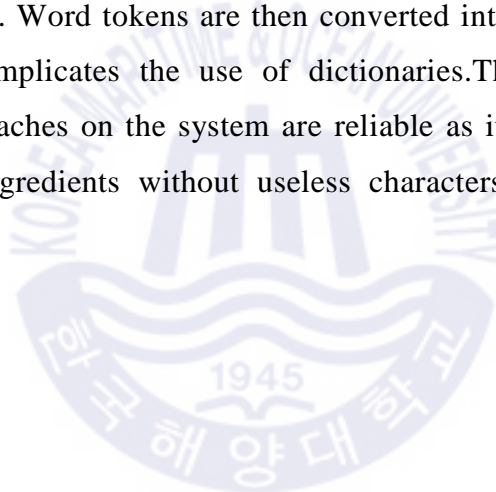
*Advised by:*

*Prof. Ok-Keun Shin*

## **Abstract**

These days, OCR and digital image processing technology are rapidly developing and there are many application areas in research and industry. This thesis presents a method to produce a better and reliable recognition by manipulating the output of OCR process in domain specific word recognition tasks. The output of OCR is improved by two post-processing steps: the tokenization and the extraction of correct word using dictionaries. The tokenization is a process where texts retrieved by OCR

are separated into word tokens. Then the tokens are compared with English and proprietary dictionaries in sequence. English dictionary is used to convert the word tokens into correct words candidates, while proprietary dictionary is used as a guide to select only meaningful words in the domain specific task. The practicality of the proposed approach was demonstrated in the task of text recognition of the ingredients list printed on the cover of the packaged foods. Based on the uploaded image of packaged food, the system performs OCR to get the editable texts. The editable texts are then tokenized into word tokens before the post-processing steps. Word tokens are then converted into correct words by the processes implicates the use of dictionaries. The result of these combined approaches on the system are reliable as it gives an accurate result of the ingredients without useless characters and nonessential ingredients





# List of Abbreviations

<b>API</b>	<b>Application Program Interface</b>
<b>C</b>	<b>C Programming Language</b>
<b>C#</b>	<b>C Sharp Programming Language</b>
<b>C++</b>	<b>C Plus Plus Programming Language</b>
<b>DB</b>	<b>Database</b>
<b>DOC</b>	<b>Microsoft Word Document</b>
<b>DPI</b>	<b>Dots Per Inch</b>
<b>HP</b>	<b>Hewlett-Packard</b>
<b>HTML</b>	<b>Hypertext Markup Language</b>
<b>PPTX</b>	<b>Microsoft PowerPoint Presentation</b>
<b>MWE</b>	<b>Multi Word Extraction</b>
<b>MySQL</b>	<b>MySQL Database</b>
<b>NLTK</b>	<b>Natural Language Toolkit</b>
<b>OCR</b>	<b>Optical Character Recognition</b>
<b>PDF</b>	<b>Portable Document Format</b>
<b>SLE</b>	<b>Systemic Lupus Erythematosus</b>
<b>TXT</b>	<b>Text File (extension)</b>
<b>UI</b>	<b>User Interface</b>
<b>US</b>	<b>United States</b>

# List of Tables

Table 1.1 OCR Engine Comparison Table .....5



# List of Figures

Fig. 2.1 The Tesseract-OCR Process Flow .....	7
Fig. 2.2 The interface of “Eatable” .....	12
Fig. 2.3 The interface of “Food Allergy Scanner” .....	13
Fig. 3.1 Flowchart of the Server.....	15
Fig. 3.2 Client-Server Architecture of this System .....	16
Fig. 3.3 Detail processes of OCR .....	18
Fig. 3.4 Flowchart on How to Extract Correct Ingredients .....	24
Fig. 3.5 Database Process Flowchart.....	25
Fig. 3.6 diseaseDB.....	26
Fig. 3.7 Entity-Relationship Diagram.....	26
Fig. 4.1 Flowchart of the Server System .....	30
Fig. 4.2 Original Image of Product.....	31
Fig. 4.3 Image after Pre-Processing .....	31
Fig. 4.4 Result of OCR .....	32
Fig. 4.5 Result of Segmentation on Tokenization Process .....	33
Fig. 4.6 Result of Character Removals and Lowercase Conversion .....	33
Fig. 4.7 Result of Multi-Word Tokenization .....	34
Fig. 4.8 Process and Result of Dictionaries using PyEnchant .....	35
Fig. 4.9 User Table Columns .....	37
Fig. 4.10 User Table and Its Information.....	37

Fig. 4.11 Disease’s Ingredients Table .....38

Fig. 4.12 User Interface of the Main Page and Health Profile.....39

Fig. 4.13 Example of Text Recognition Output .....39

Fig. 4.14 Mobile Application Output .....40



---

# CHAPTER I

## INTRODUCTION

### 1.1 Background of Research

Nowadays, the technology and the application area of digital image processing is in rapid expansion. One of the important areas of digital image processing is the Optical Character Recognition (OCR) which is necessary to recognize and convert the characters image into an editable text [1]. There are many OCR-based applications, to name a few, the application that converts image files to pdf files [2], or vehicle number plate recognition system [3] and etc.

There are also many OCR open source software in public domain that can be used and modified by software developers, for instance Tesseract, Google Cloud Vision and ABBYY FineReader. However, the OCR does not always give a perfect recognition result, which makes the post-processing important for an enhanced result.

The goal of this study is to find a method that improves the result of OCR by appending a post-processing module which makes use of dictionaries. The main purpose of these dictionaries are to identify errors in the OCR created words and to correct them into meaningful words.

Recently, food allergy, food intolerance and the auto-immune disease are not rare especially in young generations [4]. These abnormalities seem to be related with the consumption of packaged food [5].

Thus, in this study, the effectiveness of the proposed method is demonstrated in the area of food allergies and intolerances.

The post-processing of OCR is used to improve the recognition rate of the ingredients list printed on the envelope of the packaged food: The system first receives the image of the envelope of packaged food that contains ingredients list. Then, the OCR technique is used to convert the image into an editable text. Now, the converted texts undergoes two post-processing procedures: text tokenization and extraction of correct ingredients using dictionaries. The dictionaries are used as a guidance to get correct ingredients.

The extracted ingredients are compared with the disease database of a particular user to find matching ingredients. If the ingredients matched, the system tells that the product is not suitable for the user.

The approach presented in this thesis could also be used in fields such as drug substances and cosmetics etc.

## **1.2 Research Objectives**

The objectives of the research are as follows:

- **Improvement of OCR recognition rate**

- i. This system uses OCR technology to retrieve food ingredients.
- ii. The result of OCR is improved by means of post-processing method attached to OCR.

- **Safety Check System for Packaged Food**

- i. A system to check whether the packaged food contains harmful ingredients for the user.
- ii. System implemented in server-client application.

- **Construction of Dictionaries**

- i. There are two dictionaries: English dictionary and proprietary dictionary.
- ii. Proprietary dictionary contains lexical items related to food ingredients.
- iii. These two dictionaries are used in sequence to improve the results of OCR.

---

# CHAPTER II

## LITERATURE REVIEW

### 2.1 Review of Research Topics

#### 2.1.1 Review of Optical Character Recognition (OCR)

OCR literally stands for Optical Character Recognition [6]. It is a commonly used technology to recognize texts inside of images such as photos and digital documents. One of the main usages [7] of OCR is to convert hard copy of a document, such as printed paper documents into a softcopy documents which is a computer-readable texts documents for instance in formats of txt, word, pdf and etc. There are a lot of usage from the raw texts of OCR result. Although it is not always perfectly recognized, pre-processing method and post-processing methods can be used to help improvise the result to enhance the accuracy.

Currently, many powerful OCR engines are available for software developments. To name a few, there are OCR engines like Google Cloud Vision [8], Tesseract [9], ABBYY FineReader [10] and etc. As these engines are available on public domain, they are open for software developers to use the engines for their developments, but not all engines are free to be used. Some of them requires monthly or yearly subscriptions or sold at some price.



As shown on table 1.1 below, are the usages and various information comparison on a few of the most used OCR engines recently [11].

**Table 1.1 OCR Engine Comparison Table**

<b>OCR Engine Software</b>	<b>Release Year</b>	<b>OS System</b>	<b>Languages support</b>	<b>Output Format</b>	<b>Programming Language</b>
<b>Google Cloud Vision</b>	2015	Online (Browser)	200+	Text, TXT	Unknown
<b>Tesseract</b>	2018	Offline (Windows, Mac, Linux)	100+	Text, TXT, PDF, etc.	C, C++
<b>ABBYY FineReader</b>	2017	Online (Windows, Mac, Linux)	100+	DOC, XLS, PPTX, PDF, HTML, TXT, etc.	C, C++
<b>OmniPage</b>	2015	Online (Windows, Mac, Linux)	100+	DOC, XLS, PPTX, PDF, HTML, TXT, etc.	C, C++, C#
<b>OCROPUS</b>	2017	Offline (Mac, Linux)	Languages using Latin alphabets	TXT, PDF, etc.	Python

As shown previously on the table, there are the release year of each engines, the OS system used, language supports, output formats and the programming language used for the implementation of each engines.

In this study, Tesseract OCR engine is used for the system development. As it provides the latest technology of OCR and currently under further development by google, it is most probably a wise and safe decision to use this engine for the development of this system.

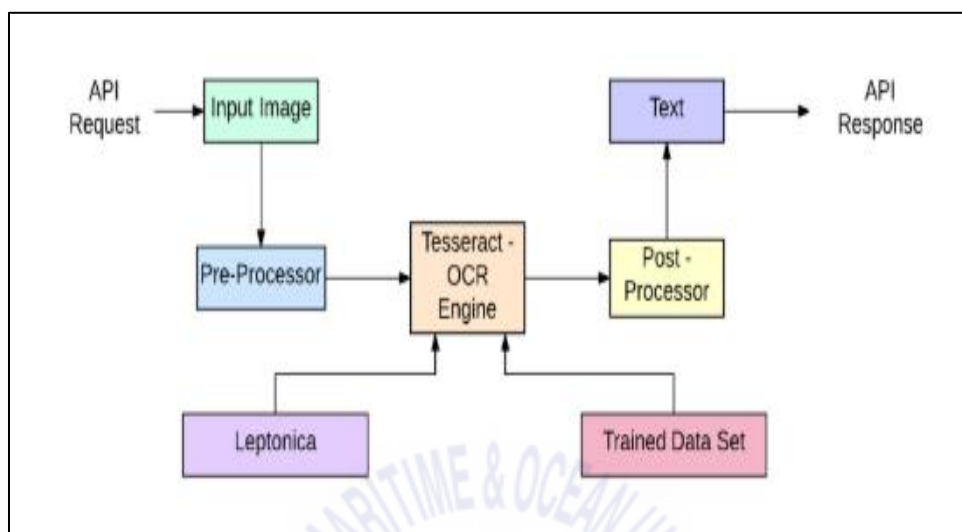
### **2.1.2 Review of The Tesseract OCR Engine**

The Tesseract OCR engine that is used in this system was first developed as a proprietary software at Hewlett Packard [12] at 1995 and now holds an Apache free software license that allows user to use, distribute and modify the software.

The development of Tesseract has been sponsored by Google from 2006 until now. Moreover, Tesseract is an offline OCR engine, free and open for open for any software development. It is also small in size and there are better documentation and examples available for this engine.

The Tesseract OCR engine is combined with Leptonica [13] Image Processing Library that can recognize varieties of image format and convert them to text in over 100 languages [14]. Tesseract is acknowledged as one of the most accurate open source OCR engines besides Google Cloud Vision, ABBYY FineReader, etc [15].

Fig. 2.1 shows the process flow of Tesseract OCR Engine.



**Fig. 2.1 The Tesseract-OCR Process Flow**

Basically, the input of the OCR process is an image that has been pre-processed to get an altered image form (in size and perspective of the image). Pre-processing of an image changes its format, corrects its position and resizes the image to get a clean and sharp image for the text recognizing process. Then, Leptonica acts as a library that is used by Tesseract, to binarize images, get position and page layout information.

After a clean and clear image is retrieved from post-processing, the image is processed through Tesseract OCR engine [16] to retrieve the texts from the image. The output of this software are texts in txt or pdf-based format. Then, post-processing of OCR is used to get a correct output in a text format [17].

### 2.1.3 Review of OCR Post-Processing

Generally, post-processing implicates as an enhancement method to something in order to improve the accuracy of its result [18]. There are a few most widely used post-processing techniques of OCR. One of the commonly used technique is the voting technique, which combines the recognition results of various OCR devices but usually without utilizing any contextual knowledge.

Voting technique [19] that are used in OCR has two approaches, first is where the data of the image is first pre-processed by different image filters. The other approach is by using a few different OCR engines on the same data input. Then the voting process chooses the results that gives the highest accuracy percentage and combines them to produce and improve the final result.

The voting technique falls into the broad approach of data extraction method which is Data-Based Extraction [20] approach. This approach is generic and usually used for materials like a book, files or newspapers as it contains a general and a large number of data.

In this study, the post-processing approach that is proposed is the implementation of a system to improve the accuracy of OCR result, which is to detect any misspelt words and correct the word based on the usage of proprietary dictionary. This approach that is proposed can be categorized as one of the Domain-Based extraction [21] approaches as the data extracted are known beforehand, thus the proprietary dictionary can be built to fix the output of OCR.

#### **2.1.4 Review of Food Intolerance, Allergies and Auto-immune Diseases**

A few years ago, cases that involves food intolerances and food allergies are often neglected and does not get as much attention like other diseases' cases. This could be because people do not see any serious effect as humans have been eating all kind of wheat and dairy products for millennia, apparently without any harm.

Last year, the number of food allergies rises and has been particularly remarkable because of a death-case in Western Australia that involves a six-year-old girl that had dairy allergy [22].

Food intolerance and food allergy contains the same meaning which is, reaction towards one or more type of food ingredients, but these two things would cause a different type of body's responses [23].

According to the British Allergy Foundation, food intolerance is a type of food ingredient that might cause a mild physical reaction to the patient and also more common than food allergies. Moreover, it produces a reaction towards the patient's body, but it does not involve the immune system of the patient. It causes digestive-related symptoms to the patients, such as gas and bloating, diarrhea, constipation, cramping, nausea and etc [24].

On the other side, food allergy causes a very serious reaction to the patient's body, it also affects numerous organs, and in some cases, a severe

food allergy reaction could be life-threatening, especially to patients that have both food allergy as well as auto-immune disease.

In extreme cases of food allergy, consuming or even touching a tiny amount of the allergen could cause a severe reaction, such as skin reaction like hives, swelling and painful itching. Other symptoms of food allergy include anaphylaxis which can cause breathing difficulty, wheezing, dizziness and death [25]

In contrast, food intolerance symptoms are generally less serious compared to food allergy. To generalize the terms of food intolerance and food allergies, these reactions will be categorized as **food inconvenience** in this thesis.

The diseases for instance are, Systemic Lupus Erythematosus (SLE) [26], Asthma [27], Anaphylaxis [28], Eczema [29] and etc. The patients of these diseases require precautions in their daily food intake. There are some ingredients that might trigger their symptoms even if a slightest amount of the allergen contained ingredients are consumed.

## **2.2 Review of Related Work**

### **2.2.1 Eatable**

‘Eatable’ [30] is an application that helps people with food allergies check and locate allergen-free food products. This application is made out of a research that proposed a method to help users locate a store selling suitable

products from allergens chosen by the user.

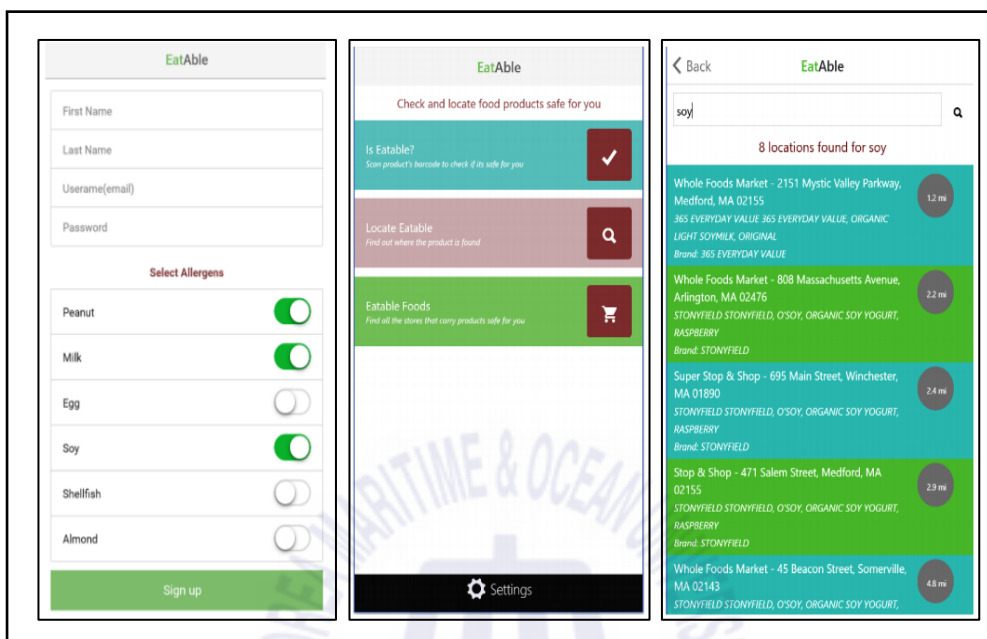
This research is inspired by many providers, one of them is Allergyeats.com which provides a similar service but runs on online website. Both of the applications share the same method which uses the allergens as an input from other users to locate the store. The difference is that 'Eatable' service focused on grocery products while Allergyeats.com helps user to find an allergy-friendly restaurant.

The Eatable application implemented a few requirements, such as user's mobile access to provide product barcode and geo-location data to the system. Then it needs user management to collect user data such as allergens, the application supports only 6 types of allergens which are milk, egg, peanut, soy, almond and shellfish.

There are also a few more requirements such as mapping data collection that collects the store data using Google Services API, product location that is showed on Google Maps by determining the user's location and also the store's database.

This application is a cloud-based hybrid mobile application that needs a crowd-sourced data from other users, which means the database needs an input of the store, product and ingredients from the users to work. It is also connected to a third-party web services to build a database.

Fig. 2.2 shows the interface of the product, “Eatable”.



**Fig. 2.2 The interface of “Eatable”**

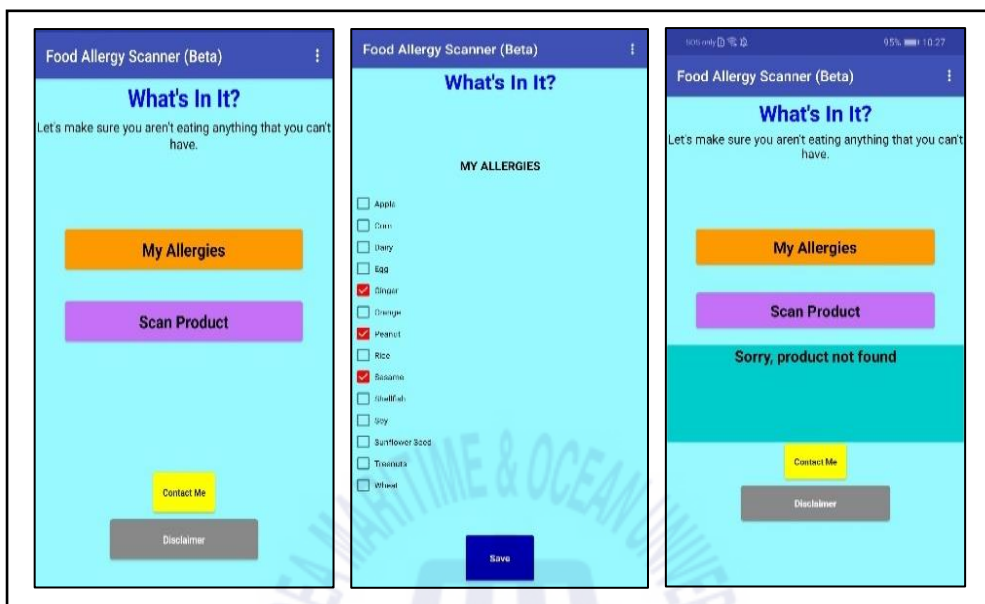
Shown on Fig. 2.2 is the user interface of the mobile application. The first figure shows the user profile and allergens selection part. Then the nearby food products and stores are shown as the result of the search.

## 2.2.2 Food Allergy Scanner

A mobile application, ‘Food Allergy Scanner’ [31] is an application which gives allergy information of a product from by choosing allergens and scanning its QR code. This application uses the online information of products.



Fig. 2.3 shows the interface of the mobile application.



**Fig. 2.3 The interface of “Food Allergy Scanner”**

As shown on the figure, this mobile application possesses an identical concept of collecting ingredients data to ‘Eatable’ application. Both of the application is using product barcode to search and obtain product information from a food products online database that are available on the internet. But undoubtedly, not all products are available and some of the products are not supplied with their ingredient’s information.

---

# CHAPTER III

## SYSTEM DESIGN

### 3.1 Overall Architecture of the System

The next figure shows the server receives the image of the product's ingredients list from the user's smartphone.

The server starts by performing the optical character recognition (OCR) to convert the image of the list of ingredients to text format.

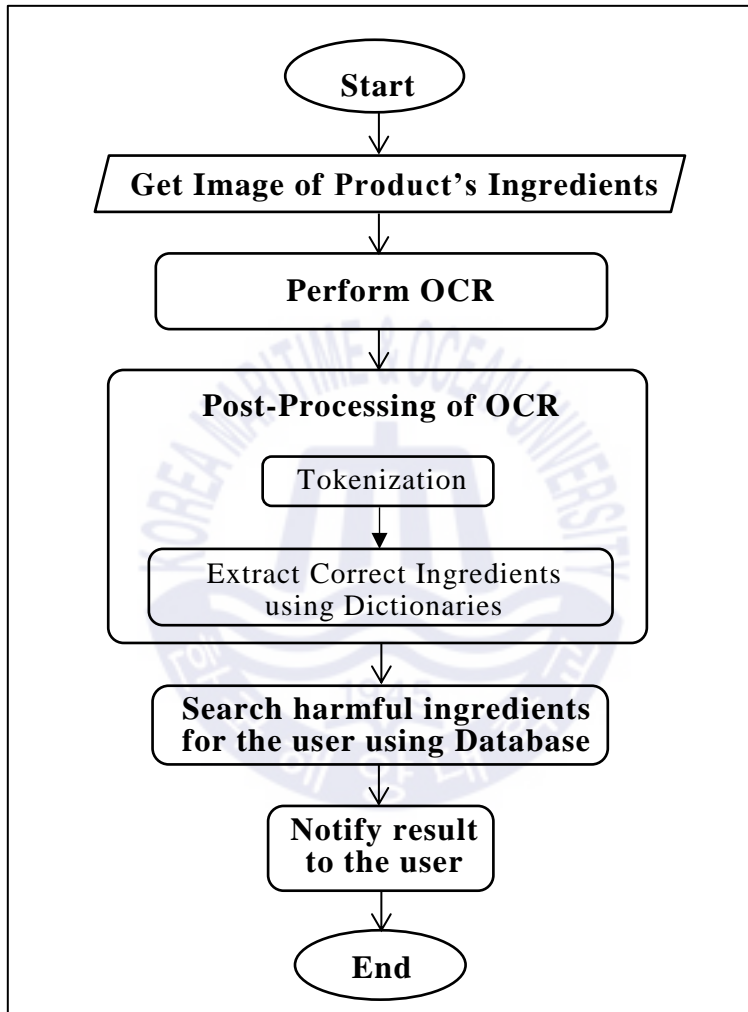
However, the output of the OCR process cannot be directly used for further processing because what is contained in the output are the strings of characters which are supposed to be ingredients, but with many misrecognitions.

Moreover, the strings of characters contain not only the ingredients, but also may contain symbols and numbers, which usually describes the portion of each ingredients. These require a sophisticated post processing to the OCR.

First of all, the system adopts post-processing steps of text recognition to convert strings into meaningful words: tokenization. The results of these post-processing are called ingredient candidates, which might still contain misrecognized words. The candidates then go through a series of processes

which makes use of proprietary dictionaries to be converted into correct ingredients.

The flowchart of the server's core process is shown in Fig. 3.1.



**Fig. 3.1 Flowchart of the Server**

Finally, the extracted ingredients are compared with the user's disease database to confirm whether it contains harmful ingredients. The

system then returns the safety information of the product to the user. Each of the server's processes are explained in more detail in the following sections.

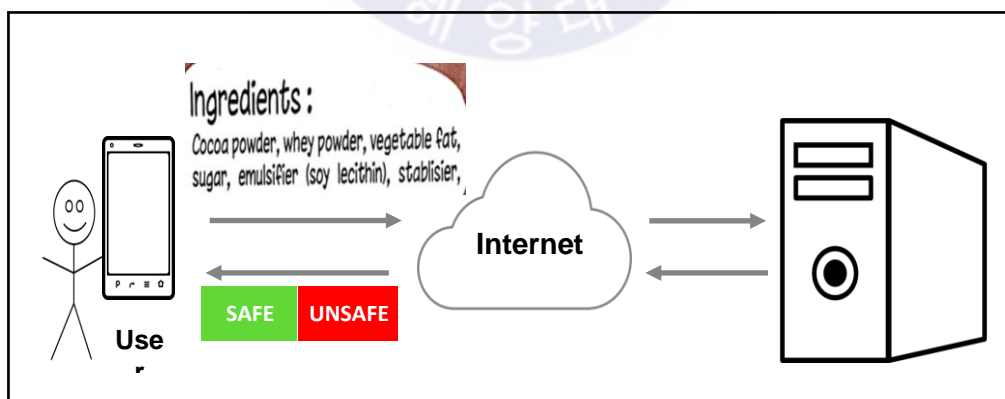
### 3.2 Get Image of Product's Ingredients

The image of product's ingredients should be provided by the user. For this system, the image of the product is sent by the user through mobile application.

On the application, the user's information, diseases details that are secured in the database and the uploaded image can be accessed by the system. To use the system, the image should be uploaded to the application before the image are sent and processed on the back end.

#### 3.2.1 Client - Server Architecture

In Fig. 3.2, the client-server architecture of the system is shown.



**Fig. 3.2 Client-Server Architecture of this System**

First of all, users register their personal information, including name, age, sex, and in particular, their disease conditions, on the server by making use of the client systems which are usually smartphones. When the user with a disease enquires if the packaged food in hand is safe to consume, the user then takes picture of the product's cover where the ingredients list is printed.

Then, the picture of the product's ingredients list is sent to the server to process. The server in its turn, first finds out each ingredient from the image of the list, and then evaluates the safeness of the food by comparing each ingredients of the food with the harmful ingredients for the user's disease. The server finally returns the safeness information to the user.

To make a correct safeness decision of the product, it is crucial to find out the exact ingredients list from the image.

### **3.3 Perform OCR**

The ingredient candidates are texts that are recognized by OCR. The texts are called the ingredient candidates as they might contain misrecognized words as well as texts that are not ingredients.

To get the correct ingredient, the candidates must go through a few post-processing methods after the OCR process. The OCR and its post processes are explained below.

### 3.3.1 OCR and Its Pre-Processing

The OCR technology are used in this system to extract ingredients information from image. OCR technology adopts a few pre-processing steps before the main process, which in its turn includes segmentation, feature extraction, and classification. Then, the post-processing follows.

On Fig. 3.3 the detail process of OCR is shown.

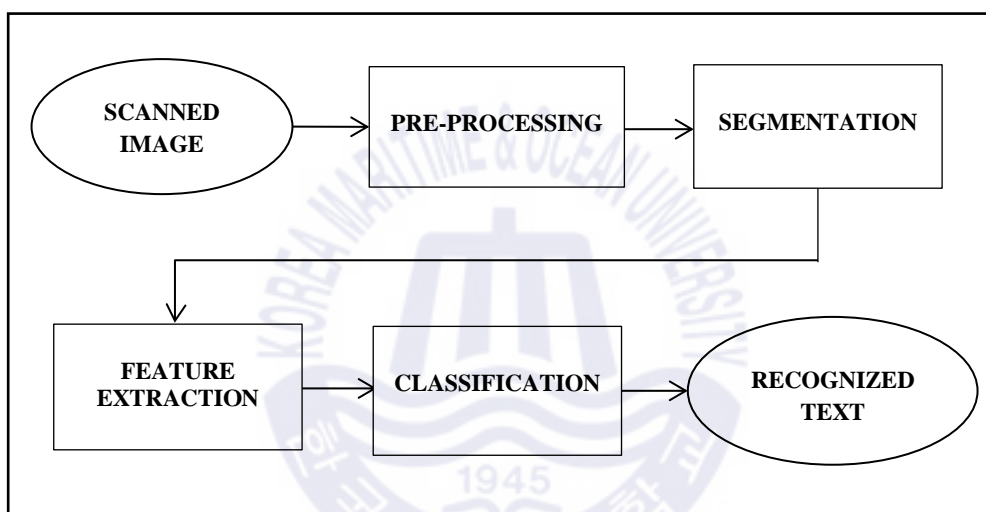


Fig. 3.3 Detail processes of OCR

#### 1) Pre-Processing:

The **pre-processing** involves the following steps [32].

##### a) Image rescaling:

Image rescaling is very important for image analysis. It means to resize the image to satisfy the qualification of OCR process. The image is

recommended to be rescaled if it has less than 300 DPI, as most of OCR engine gives an accurate output if the image has at least 300 DPI.

**b) Skew correction:**

Skew correction is used for a skewed, slanted or image that is not straight. The skewed image gives a direct impression to the OCR line segmentation which reduces its accuracy. To correct the skewed text, it is necessary first to detect the text block with skew in the image, then to calculate the angle of the rotation, before rotating the image to correct the skew.

**c) Binarization:**

OCR machines does binarization internally as they usually work on black and white images. The simplest way is to calculate the threshold value and convert all pixels above the threshold value to white, and the remaining pixels to black.

**d) Noise removal:**

Noise removal is also known as denoise. Noise is a random variation of the color or brightness between image pixels. Noise reduces the readability of the text from the image. There are two main types of noise - salt and pepper noise and Gaussian noise.

**2) Segmentation:**

After these pre-processing steps, the segmentation follows to partition digital image into multiple segments (sets of regions). Image

segmentation is used for text-based images aiming in retrieval of specific information from the entire image. This information can be a sentence or a word or even a character [33].

### **3) Feature extraction technique and classification technique:**

These techniques play a major role in the performance of the OCR. The characteristics of the feature extraction techniques must be independent of the scalable font characteristics such as type, size, style, tilt, rotation and should be able to describe the complex, distorted, broken characters effectively. A feature vector should be simple, reliable, complete, and compact to recognize any input character with high accuracy similar to human perception [34].

The OCR process follows after retrieving processed image that adopted the pre-processing methods. The OCR are processed to get the recognized texts before further processing.

### **3.4 Post-Processing of OCR**

The Post-processing method involves two processes which make use of natural language processing technology. They are used for more accurate ingredient extraction. The approach of OCR post-processing that are used in this system are explained in the next sections.



### 3.4.1 Tokenization

The general meaning of tokenization is the act of splitting a sequence of strings into pieces called tokens, which can be words, phrases and symbols. Tokenization is one of the natural language processing tasks and is commonly used in computer science and it is also one of the important process of lexical analysis.

During tokenization, tokens and words are separated by whitespaces, punctuation marks or line breaks. On some cases when white spaces or punctuation marks are not needed, these marks may be thrown away during the tokenization. The tokens can be made up of all alphabet characters, alphanumeric or numeric characters [35].

There are a few steps that are needed in this system's tokenization process [36];

#### 1) **Sentence segmentation:**

This segmentation which is also one of the crucial are performed based on punctuations such as ‘.’, ‘?’ and ‘!’ as they tend to mark the boundaries of the sentence.

#### 2) **Words segmentation:**

The tokenizer needs to segment those text into words. The separation of the word is via unique space character.

### 3) **Characters removal and convert text to lowercase:**

In this study, the presence of non-alpha numeric characters within the word is one of the indicators of an unwanted word. Characters and specifiers like ‘%’, ‘\$’, punctuation marks ‘!’,’?’), hyphens, comma and period signs are removed together, as this system does not need any other information than the name of the ingredients.

Then, to make the recognition easier, any uppercase alphabets are converted into lowercase alphabet before continuing.

### 4) **Multi-word tokenization:**

Another important step for the tokenizer for this system is, the process of multi-word tokenization which is needed as the ingredients’ name might be formed in more than one word, for instance, ‘green tea’, ‘sour cream’, or ‘maple syrup’. The multi-word tokenizer takes a string of a tokenized words and re-tokenize it to merge a multi-word expression into a single token [37].

The pre-processing of this system is using the word tokens as it is without further processing under stemming and lemmatization. This is a way to preserve the original tokens as those processes cuts a lot of characters from the tokens thus makes it meaningless.

### **3.4.2 Extract Correct Ingredients using Dictionaries**

In this system, there are two types of dictionaries that are used. These dictionaries play a very important role in this system as it is used for a post-processing of this approach. Listed below are the dictionaries used in this system.

#### **1) English dictionary:**

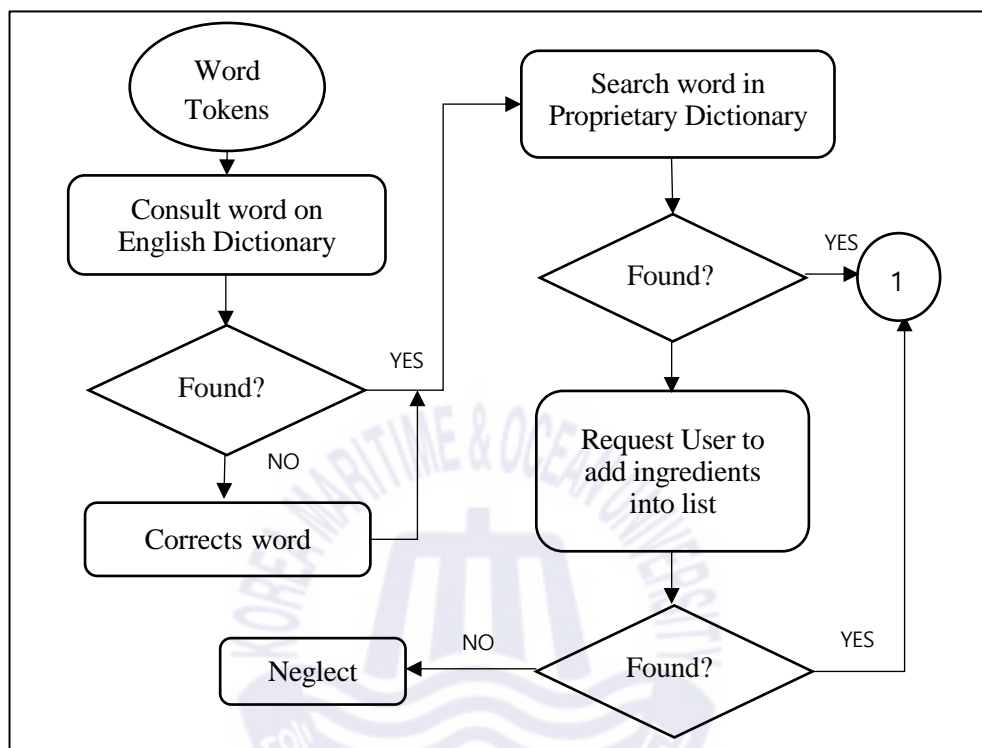
English dictionary is retrieved from an English dictionary database which includes more than 370,000 words. The English dictionary database is obtained from a trusted website of many software developers community.[38] To ensure that the ingredients are well-recognized, the lists of ingredients are also added to the dictionary.

#### **2) Proprietary dictionary:**

The proprietary dictionary is the custom-made ingredients list prepared for this study. The data is collected from various sources. They are usually included in the envelope or the packaging of packaged food. Also, some ingredients are obtained by the U.S food and drug administration for the list of food ingredients, additives and colors list, as well as food ingredients dictionary list which provides more than 5000 types of ingredients.

The ingredients list contains a large number of redundancy, and needed to be excluded from the list. Most of the packaged food ingredients are obtained through internet, but the dictionary can always be updated. The proprietary dictionaries are used as a guide to select the real ingredients out of the candidates of ingredients extracted from the OCR process.

Fig. 3.4 below shows the flowchart on how to extract real ingredients.



**Fig. 3.4 Flowchart on How to Extract Correct Ingredients**

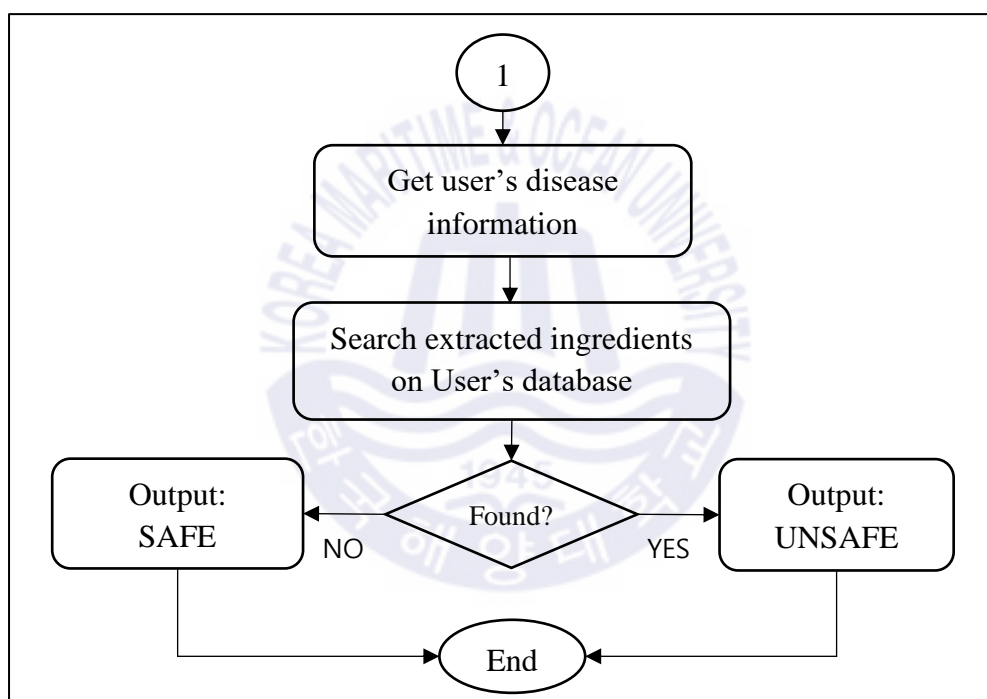
In this study, the system retrieves the word tokens from previous process, which is the tokenization, then extracts the ingredient candidates using English dictionary to assure that the word tokens retrieved are meaningful words and that they are retrieved in correct spellings.

Then, the candidates are processed using proprietary dictionary as a guide to make sure that only the correct ingredients are selected, the unnecessary words are neglected from the queue of the ingredients list.

### 3.5 Search Harmful Ingredients for The User using Database

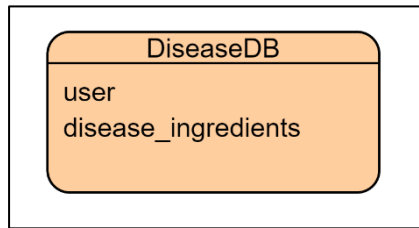
The database of this system is used to record the user's disease information. Then, the extracted ingredients from previous process are matched with the User's database to find out if there are any matches. If matched, it means that the product is not safe for the user.

Shown below is Fig. 3.5 which shows the database's flowchart.



**Fig. 3.5 Database Process Flowchart**

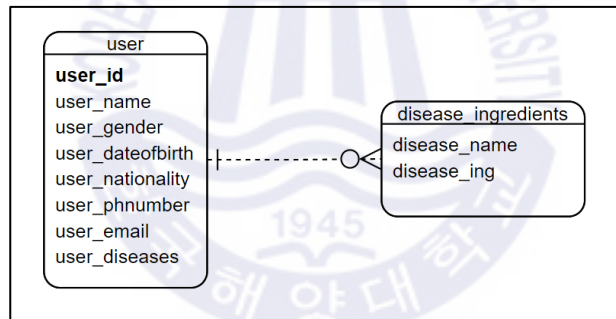
The database of this system is named as 'diseaseDB'. This database contains two tables to record all possible data from patients and the admin. The Fig. 3.6 shows the 'diseaseDB' and its tables. The database is made simple for this system.



**Fig. 3.6 diseaseDB**

It is built first of all, to record the data of the user, and the diseases. Furthermore, it is also used to match the string between the retrieved ingredients from the photo with patient’s diseases which contains the harmful ingredients of the disease.

Fig. 3.7 below shows the entity-relationship diagram of the system’s database.



**Fig. 3.7 Entity-Relationship Diagram**

As shown in the Fig. 3.7, the database of this system includes two tables: ‘user’ table and ‘disease\_ingredients’ table. The user table contains a few columns that are filled with the user information including the food intolerance of the user. The ‘disease\_ingredients’ table contains list of the disease names, and its ingredients which causes trouble to the disease.

Once the list of ingredients of the product is obtained in text format, the server first finds out the user's diseases from the user table and obtains the ingredients harmful to the user.

Then, the system compares each ingredient in the product ingredient list with the harmful ingredients. If one or more of the ingredients acquired are matched with the chosen diseases' ingredients, the user is notified that the product is harmful for the user.

### **3.6 Notify Result to The User**

Corresponds to 'Get Image of Product's Ingredients', the result of the system is also shown on mobile application. The mobile application retrieves either 'UNSAFE' or 'SAFE' notification from the system after the database process.

To get the result, the product's ingredients extracted are processed by matching them with the user's database ingredients. The system indicates that the product contains harmful ingredients if there are matched ingredients between database and extracted ingredients. Hence, the application notifies the user that the product is 'UNSAFE' and the list of harmful ingredients are provided together.

In case of no matched ingredients are found, the application notifies the user that the product is 'SAFE', and that the product is suitable for the user's diet.

---

# CHAPTER IV

## SYSTEM IMPLEMENTATION

### 4.1 Overall Explanation

The user sends image of ingredients from the client terminal to the server and the server produces result to the client terminal at the end of the process.

The front-end client system is an android mobile application that provides the UI to the application. The client system is fully dependent on the backend system for the system processes and the data collection.

For the backend sever system, appropriate third-party open source software is chosen and assembled in a python environment.

The open source Tesseract's OCR library is chosen to implement the system's main function, the text recognition. This program recognizes the input text image and convert them into editable text.

Then the output text is used as the input to the post-processing processes, which are tokenization, segmentation and multi-tokenization. The open source software Natural Language Tool Kit (NLTK) [39] is adopted to perform these functions.



The retrieved word tokens are compared with two dictionaries in sequence by means of PyEnchant [40], a python spell checker tool.

First of all, the ingredient candidates are found by comparing the tokens with an English dictionary. The dictionary corrects any misspelt word to the most similar word suggestion made by the spell checker. Then, the proprietary dictionary is used to select only ingredients components which are registered in this dictionary.

Other words that are not recognized by the proprietary dictionary will be listed as 'ingredient not found'. The example is shown on the Post-Processing sub-section in this chapter.

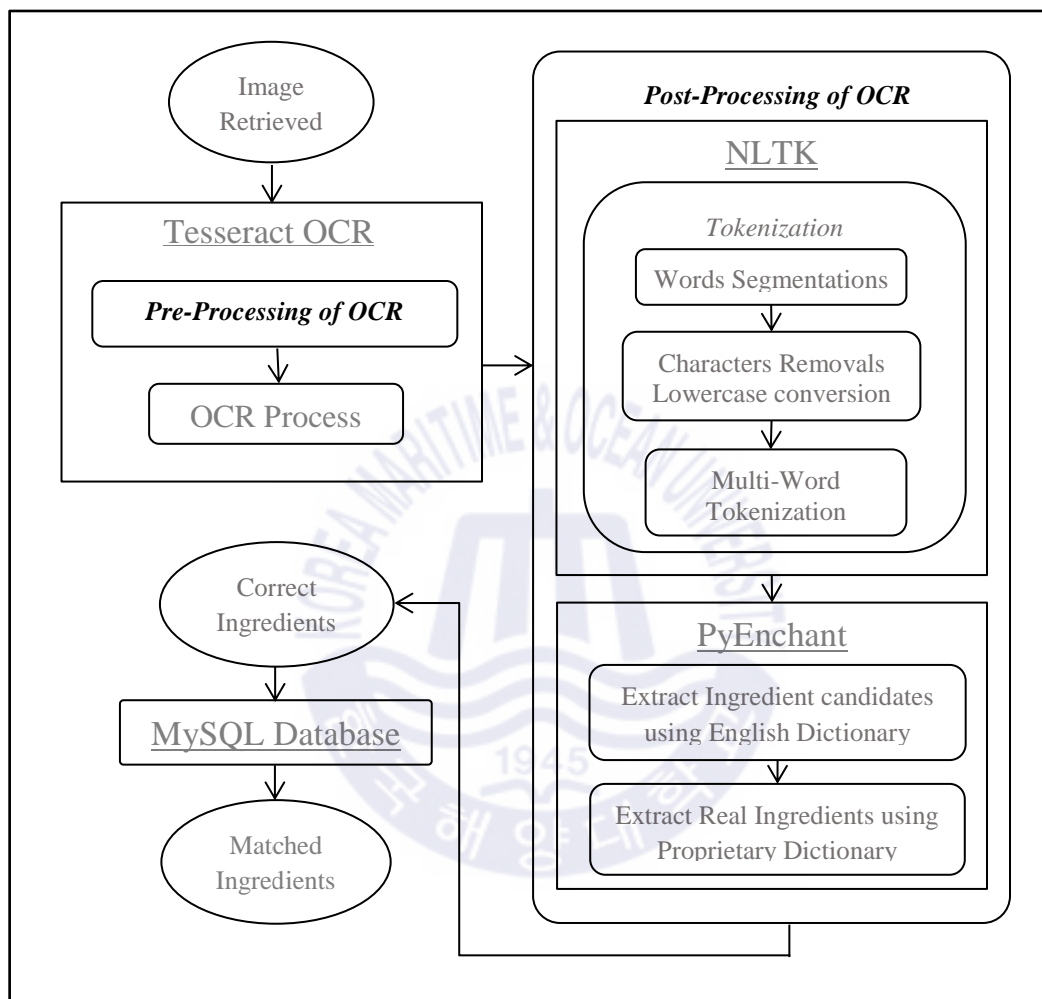
In this study, the proprietary dictionary is assumed incomplete and the dictionary are extensible where administrators can add new words as needed.

For the persistence of the system and user's personal data, MySQL [41] is used.

Assuming that the correct ingredients are retrieved from the image of the ingredients by the system, the list of ingredients is sent to the database for string match to find out any harmful ingredients according to the user's data.

If a string thus found is included in the user's database, the response is 'UNSAFE', otherwise, the 'SAFE' decision is returned to the user's terminal.

Flowchart in Fig. 4.1 shows the flow of the program and the software tools used to implement the system.



**Fig. 4.1 Flowchart of the Server System**

The following sub-sections show the examples of input and output of processes executed on this system. The image that is used as an example of this system's input is a food product 'Hershey's Milk Chocolate'.

### 4.1.1 Pre-Processing of OCR

Tesseract OCR library is used to perform OCR. In Tesseract OCR library, pre-processing functions of OCR, such as binarization, skew correction, noise removal and etc. are also included. These functions are based on Numpy and OpenCV utilities and wrapped by python.

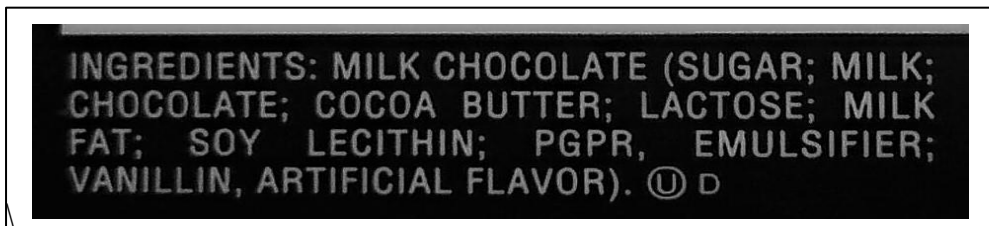
In Fig.4.2, the original image captured from the envelope of a packaged food is shown.



**Fig. 4.2 Original Image of Product**

The image goes through pre-processing steps such as, image rescaling, skew correction, binarization and etc.

The result of the pre-processing is shown in Fig. 4.3 below.



**Fig. 4.3 Image after Pre-Processing**

The final image acquired from pre-processing steps of OCR are then sent for OCR process to retrieve the editable texts from the image.

Fig. 4.4 shows the OCR Result of the food product.



**Fig. 4.4 Result of OCR**

The result of OCR is not suitable for further processing, because of many unnecessary components are included such as symbols, numbers and etc. Therefore, the processes of getting a clean output is very important.

After the pre-processing and OCR output is retrieved, the editable texts are used for post-processing. There are two post-processing steps: first of all, the tokenization process, then extraction of correct ingredients using dictionaries.

#### **4.1.2 Post-Processing of OCR**

##### **1. Tokenization Process**

The implementation of the tokenization the OCR result is achieved by one of NLTK modules which is `nltk.tokenize.API` module. This tokenizer divides a string into words by splitting the strings.

Before other processes of tokenization takes place, shown on Fig. 4.5 below is the result of the first step of tokenization process which are the sentences and words segmentation step.

```
['INGREDIENTS', ',', 'MILK', 'CHOCOLATE', '(', 'SUGAR', ',', 'MILK', ',', 'CHOCO  
LATE', ',', 'COCOA', 'BUTTER', ',', 'LACTOSE', ',', 'MILK', 'FAT', ',', 'SOY',  
LECITHIN', ',', 'PGPR', ',', 'EMULSIFIER', ',', 'VANILLIN', ',', 'ARTIFICIAL',  
FLAVOR', ')', ',', '@', '0']
```

**Fig. 4.5 Result of Segmentation on Tokenization Process**

By taking the result of the OCR, the system then uses the tokenizer, to remove the unwanted characters, numbers, convert the uppercase alphabets into lowercase alphabets.

Shown in the Fig. 4.6 below is the result of these steps.

```
milk chocolate sugar milk chocolate cocoa butter lactose milk fat soy lecithin p  
gpr emulsifier vanillin artificial flavor
```

**Fig. 4.6 Result of character removals and lowercase conversion**

After previous processes, the final step of tokenization is to recognize whether the ingredients candidates are formed by more than one word. If the candidates happen to be a form of a multi-word candidate, the system then re-tokenizes the single word tokens to their original form(multi-word).

In this step, multi-word tokenization is used to re-tokenize them before other processes to get ingredients candidates.

The result of multi word tokenization are shown on the Fig. 4.7 below.

```
['milk_chocolate', 'sugar', 'milk_chocolate', 'cocoa_butter', 'lactose', 'milk',  
'fat', 'soy_lecithin', 'pgpr', 'emulsifier', 'vanillin', 'artificial_flavor']
```

**Fig. 4.7 Result of Multi-Word Tokenization**

The result of tokenized multi word are separated by ‘\_’ to indicate that there are whitespaces between the word. The result is then used on next step of getting the ingredients candidates.

## **2. Get Correct Ingredients using Dictionaries**

From the tokenizer result, the tokens are used to extract correct ingredient candidates from English dictionary, then real ingredients from the proprietary dictionary of ingredients.

The basic process is first of all, it uses PyEnchant to run the spell checker by using the English dictionary to retrieve the ingredient candidates. Then, the candidates are used as an input and process through proprietary dictionary of ingredients to extract the real ingredients.

In some cases where there might be spelling error, English dictionary is used to extract correct spelling of an English word.

For instance, the example taken from Fig. 4.8, ‘*ingredientss*’ are recognized as a false English word. Therefore, it is changed to ‘*ingredients*’. Another example, ‘*cocoa\_butteer*’ which in this example there are ‘\_’ obtained through the multi word tokenization.

```

===== RESTART: C:\Users\Administrator\Desktop\diseaseSC\dictionaries.py =====
ENGLISH DICTIONARY
('INPUT = ', 'ingredients')
('word exists? = ', False)
('CORRECT WORD = ', 'ingredients')
INGREDIENTS DICTIONARY
('INPUT = ', 'ingredients')
('word exists? ', False)
ingredient not found
>>>
===== RESTART: C:\Users\Administrator\Desktop\diseaseSC\dictionaries.py =====
ENGLISH DICTIONARY
('INPUT = ', 'milk_chocolate')
('word exists? = ', False)
('CORRECT WORD = ', 'milk chocolate')
INGREDIENTS DICTIONARY
('INPUT = ', 'milk chocolate')
('word exists? ', True)
('milk chocolate', 'is an ingredient')
>>>
===== RESTART: C:\Users\Administrator\Desktop\diseaseSC\dictionaries.py =====
ENGLISH DICTIONARY
('INPUT = ', 'sugarr')
('word exists? = ', False)
('CORRECT WORD = ', 'sugar')
INGREDIENTS DICTIONARY
('INPUT = ', 'sugar')
('word exists? ', True)
('sugar', 'is an ingredient')
>>>
===== RESTART: C:\Users\Administrator\Desktop\diseaseSC\dictionaries.py =====
ENGLISH DICTIONARY
('INPUT = ', 'cocoa_butteer')
('word exists? = ', False)
('CORRECT WORD = ', 'cocoa butter')
INGREDIENTS DICTIONARY
('INPUT = ', 'cocoa butter')
('word exists? ', True)
('cocoa butter', 'is an ingredient')
>>>
===== RESTART: C:\Users\Administrator\Desktop\diseaseSC\dictionaries.py =====
ENGLISH DICTIONARY
('INPUT = ', 'soy_lecithin')
('word exists? = ', False)
('CORRECT WORD = ', 'soy lecithin')
INGREDIENTS DICTIONARY
('INPUT = ', 'soy lecithin')
('word exists? ', True)
('soy lecithin', 'is an ingredient')
>>>
===== RESTART: C:\Users\Administrator\Desktop\diseaseSC\dictionaries.py =====
ENGLISH DICTIONARY
('INPUT = ', 'pgpr')
('word exists? = ', False)
('CORRECT WORD = ', 'pps')
INGREDIENTS DICTIONARY
('INPUT = ', 'pps')
('word exists? ', False)
ingredient not found

```

**Fig. 4.8 Process and Result of Dictionaries using PyEnchant**

However, in this process it will convert ‘\_’ to a whitespace. Thus, the process produces correct ingredient which is ‘*cocoa butter*’. After

extracting correct ingredient candidates (which is in this example are ‘*ingredients*’ and ‘*cocoa butter*’) after both of the candidates are processed through the proprietary dictionary of ingredients, it will recognize ‘*cocoa butter*’ as an ingredient.

On a side note, as ‘*ingredients*’ are not recognized by the proprietary dictionary of ingredients, it is neglected and removed from the queue of the real ingredients result.

Shown in Fig. 4.8 are the examples of result from the use of proprietary dictionaries using PyEnchant spell checker module.

The system then proceeds to collect the result of the proprietary dictionary to be added to the list of correct ingredients before progressing further.

## **4.2 Database of the System**

The database of the system is built using MySQL in python. The database is appointed as ‘*diseaseDB*’ and it is used to log the data information of the patients as well as the data of the disease ingredients.

First of all, Fig. 4.9 shows the implementation of user table which consists of the ‘*user\_name*’ and ‘*user\_disease*’ column. The information is composed of the users’ personal data and their disease types.



Field	Type	Null	Key	Default	Extra
user_name	varchar(255)	NO		NULL	
user_disease	varchar(255)	NO		NULL	

**Fig. 4.9 User Table Columns**

Shown in Fig. 4.10 below is the sample of the users and their diseases collected and stored in the database for further processing.

user_name	user_disease
신속근	Lupus
김민관	Coeliac
자라관	Asthma
최원국	Histamine Intolerance
김정래	Eczema
김정민	Anaphylaxis
야나근	Birch Pollen
천변근	Ragweed Pollen
이카	Timothy and Orchard Grass
우미	Mugwort Pollen

**Fig. 4.10 User Table and Its Information**

Fig. 4.11 is the implementation of the ‘disease\_ingredients’ table, which consists of ‘disease\_name’ column for the names of the diseases and ‘disease\_ingredients’ column for the ingredients of the diseases.

disease_name	disease_ingredients
Lupus	alfalfa, garlic, saturated fat, trans fat, white potato, tomato,
Coeliac	bread, pasta, cereal, biscuit, cracker, oat, gluten, barley, ry
Asthma	egg, cow milk, peanut, soy, wheat, fish, shrimp, shellfish, tre
Eczema	citrus, dairy, egg, gluten, wheat, vanilla, cloves, cinnamon, t
Anaphylaxis	egg, peanut, walnut, pecan, soy, wheat, fish, shellfish, cow mi
Birch Pollen	almond, apple, apricot, carrot, celery, cherry, hazelnut, kiwi,
Ragweed Pollen	banana, cucumber, melon, cantaloupe, honeydew, watermelon, whit
Timothy and Orchard Grass	cucumber, watermelon, orange, peach, peanut, tomato, white pota
Mugwort Pollen	apple, bell pepper, broccoli, cabbage, carrot, celery, cauliflo
Histamine Intolerance	alcohol, pickle, cheese, salami, ham, sausages, shellfish, chic

hot pepper, eggplant, alcohol, green tea
e, wheat, farina, graham flour, semolina, durum, cous cous, spelt
e nut, dried fruit, potato, wine, beer, line, lemon
onato, beans, black tea, chocolate, cocoa, nut, pea, seed, shellfish, soybean, soy, green apple, carrot, hazelnut, pear
lk, wheat, shrimp, lobster, crab
peach, peanut, pear, plum, potato, soybean, anise, caraway, coriander, fennel, parsley
e potato, zucchini
to, zucchini
wer, garlic, onion, peach, aniseed, black pepper, caraway seed, coriander, fennel, mustard, parsley, chard
kpea, soy bean, peanut, walnut, cashew nut, chocolate, cocoa, vinegar

**Fig. 4.11 Disease's Ingredients table**

The disease ingredients are placed in a row and separated by commas to indicate different strings.

For instance, when the user chose a disease 'Eczema', then 'chocolate', 'milk' and 'wheat' are retrieved from list of real ingredients, the list is then matched by adopting the list of ingredients on 'disease\_ingredient' of 'Eczema' disease.

### 4.3 System Prototype using Android Studio

Subsequently, the composition of this section are user interfaces of the mobile application implementation built using the system's approach.

Fig. 4.12 below shows the main page of the application which contains button for camera, gallery, health profile and verify button to run the system.



**Fig. 4.12 User Interface of the Main Page and Health Profile**

First of all, the user needs to register their disease on the health profile page by ticking the disease's names available on the list. Then, the user can provide the image of the product's ingredients by either capturing or uploading the image from the gallery.

When the image are uploaded on the application, the text recognized by the OCR library are shown beneath the image uploaded. The examples of the output are shown on Fig. 4.13 below.



**Fig. 4.13 Example of Text Recognition Output**

The text recognized by the OCR process are then sent to the system to operate other processes resulting to the output as shown on Fig. 4.14 below.



**Fig. 4.14 Mobile Application Output**

Output with ‘UNSAFE’ are shown together with the matched strings from the real ingredients list, which is telling the user that the product are not suitable for the disease’s diet. Otherwise, the output is ‘SAFE’.

---

# CHAPTER V

## CONCLUSION

Recently, the number of packaged food products are increasing to suffice the demand of the consumers. The packaged food's industry accommodates a wide range of food products. As the number of food intolerance patients is also increasing, a lot of studies and mobile applications are developed to help those people in need.

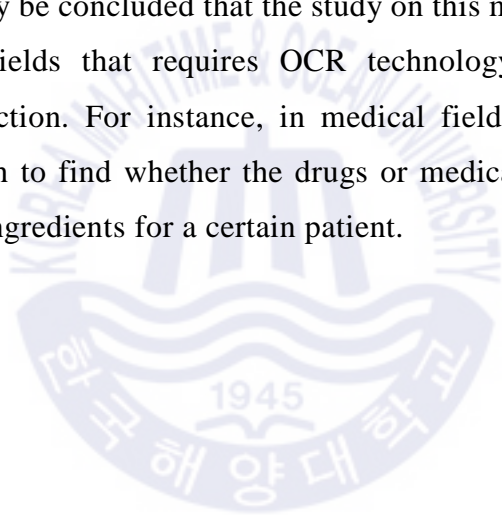
In summary, this paper describes a software system, which helps people with a certain food intolerance, such as allergies or immune disorder, by informing them if the packaged food in question contains harmful ingredients for the user.

As the main feature of this system is using OCR, sometimes the text recognition comes out incorrect. To reduce the possibility of incorrect words, the system uses the dictionaries implemented based on the system needs, which is in this thesis are described as 'food ingredients'. The implementation of proprietary dictionary of this system are crucial in order to get the correct ingredients.

To use the system, the user captures the ingredients list printed on the envelope of the product. Then, the server receives image and executes OCR processes to transform the ingredients list to text format. The strings in the

text are separated into word tokens by using tokenizer. The word tokens are then used to execute the dictionaries processes to find correct ingredients. Once the list of correct ingredients of the product is obtained in text format, the system compares each ingredient in the product ingredient list with the harmful ingredients stored in the user's database. If one or more of the ingredients acquired are matched with the chosen diseases' ingredients, the user is notified that the product is harmful for the user.

Finally, it may be concluded that the study on this method can be used in many other fields that requires OCR technology and proprietary dictionary to function. For instance, in medical field they can use the system's approach to find whether the drugs or medical substances used contains certain ingredients for a certain patient.



---

## REFERENCES

1. Krishna, M. (2017, Nov). Digital Image Processing Techniques in Character Recognition, International Journal of Scientific Research in Computer Science, Engineering and Technology (IJSRCSEIT), pp 1-5
2. Patel, C. (2012, Oct). Optical Character Recognition by Open Source OCR Tool Tesseract, Charotar University of Science and Technology, pp 4-8
3. Kaur, K. (2013, Sept) Number Plate Recognition Using OCR Technique, International Journal of Research in Engineering and Technology, pp 1-11
4. Santos, A. (2019, Sept). Why the world is becoming more allergic to food, King's College London, Retrieved From <https://www.bbc.com/news/health-46302780>
5. Oria, M.P. (2016, Nov) Finding a Path to Safety in Food Allergy, National Academies of Sciences, Engineering and Medicine
6. Wikipedia (2019, Dec) Optical character recognition, Retrieved From [https://en.wikipedia.org/wiki/Optical\\_character\\_recognition](https://en.wikipedia.org/wiki/Optical_character_recognition)
7. Rouse, M. (2019, Apr) OCR (Optical Character Recognition), Retrieved From <https://searchcontentmanagement.techtarget.com/definition/OCR-optical-character-recognition>



8. Google Cloud, (2019) Google Cloud Vision documentation, Retrieved From <https://cloud.google.com/vision/docs/>
9. Wikipedia, (2019, Sept) Tesseract (software), Retrieved From [https://en.wikipedia.org/wiki/Tesseract\\_\(software\)](https://en.wikipedia.org/wiki/Tesseract_(software))
10. ABBYY FineReader, (2019) ABBYY Cloud OCR SDK, Retrieved From <https://www.ocrsdk.com/?ga=2.102026945.1780483114.1576588947-1127810322.1576588947>
11. Wikipedia, (2019, Oct) Comparison of OCR Software, Retrieved From [https://en.wikipedia.org/wiki/Comparison\\_of\\_optical\\_character\\_recognition\\_software](https://en.wikipedia.org/wiki/Comparison_of_optical_character_recognition_software)
12. Wikipedia, (2019, Oct) Hewlett-Packard, Retrieved From <https://en.wikipedia.org/wiki/Hewlett-Packard>
13. Bloomberg, D. (2019, Mar) Leptonica, Retrieved From <http://www.leptonica.org/>
14. Wikipedia, (2019, Oct) Comparison of OCR Software, Retrieved From [https://en.wikipedia.org/wiki/Comparison\\_of\\_optical\\_character\\_recognition\\_software](https://en.wikipedia.org/wiki/Comparison_of_optical_character_recognition_software)
15. Wikipedia, (2019, Oct) Comparison of OCR Software, Retrieved From [https://en.wikipedia.org/wiki/Comparison\\_of\\_optical\\_character\\_recognition\\_software,](https://en.wikipedia.org/wiki/Comparison_of_optical_character_recognition_software)
16. Wikipedia, (2019, Sept) Tesseract (software), Retrieved From [https://en.wikipedia.org/wiki/Tesseract\\_\(software\)](https://en.wikipedia.org/wiki/Tesseract_(software))

17. Parthasarathy, B. (2018, Feb) Build your own Optical Character Recognition, Retrieved From <https://medium.com/@balaajip/optical-character-recognition-99aba2dad314>
18. Dengel, A. *et al.* (1997) Techniques for improving OCR Results, Handbook of Recognition and Document Image pp 227-258
19. Boiangiu, C.A *et al.* (2016, Dec) Voting-Based OCR System, Polytechnic University of Buchares, pp 2-11
20. Mathews, M. (2018, Oct) Data-based extraction, Retrieved From <https://dzone.com/articles/what-is-the-best-ocr-extraction-method-on-printed-1>
21. Mathews, M. (2018, Oct) Domain-based extraction, Retrieved From <https://dzone.com/articles/what-is-the-best-ocr-extraction-method-on-printed-1>
22. Santos, A. (2019, Sept) Why the world is becoming more allergic to food, King's College London, Retrieved From <https://www.bbc.com/news/health-46302780>
23. Li, J.T.C. (2019, May) Food Allergy vs. Food Intolerance: What's the Difference? (Expert Opinion), Retrieved From <https://www.mayoclinic.org/diseases-conditions/food-allergy/expert-answers/food-allergy/faq-20058538>

24. Schaefer, A. (2018, Oct) Food Allergy vs. Sensitivity: What's the Difference? Retrieved From <https://www.healthline.com/health/allergies/food-allergy-sensitivity-difference#food-allergies>
25. Schaefer, A. (2018, Oct) Food Allergy vs. Sensitivity: What's the Difference? Retrieved From <https://www.healthline.com/health/allergies/food-allergy-sensitivity-difference#food-allergies>
26. Watson, S. (2018, Feb) Diet Tips for Lupus, Retrieved From <https://www.healthline.com/health/lupus/diet-tips>
27. Brennan, D. (2018, May) Foods That Can Trigger Asthma Attacks, Retrieved From <https://www.webmd.com/asthma/guide/food-allergies-and-asthma#1>
28. Food Allergy Research & Education, (2019, June) About Anaphylaxis Retrieved From <https://www.foodallergy.org/life-with-food-allergies/anaphylaxis/about-anaphylaxis>
29. Nall, R. (2018, Feb) Eczema elimination diet and foods to eat, Retrieved From <https://www.medicalnewstoday.com/articles/320855.php>
30. Prajapati, S. (2017, May) Eatable: An Application That Helps People with Food Allergies Check and Locate Allergen-Free Food Products, Harvard University pp 1-40

31. Avani, G. (2018) Food Allergy Scanner (Android Mobile App) Retrieved From [https://play.google.com/store/apps/details?id=appinvento.aiavnig2005.FoodAllergyScannerv1\\_checkpoint1&hl=en](https://play.google.com/store/apps/details?id=appinvento.aiavnig2005.FoodAllergyScannerv1_checkpoint1&hl=en)
32. Gupta, B. (2018, Sept) Improve Accuracy of OCR using Image Pre-processing, Retrieved From <https://medium.com/cashify-engineering/improve-accuracy-of-ocr-using-image-preprocessing-8df29ec3a033>
33. Mehul *et al*, (2014, March) Text-Based Image Segmentation Methodology, International Conference on Innovations in Automation and Mechatronics Engineering
34. Soora, N.R. *et al*, (2017, July) Review of Feature Extraction Techniques for Character Recognition, IEEE Journal of Research, pp 1-4
35. Techopedia, (2019) Tokenization, Retrieved From <https://www.techopedia.com/definition/13698/tokenization>
36. Trim, C. (2013, Jan) The art of Tokenization, Retrieved From <https://www.ibm.com/developerworks/community/blogs/nlp/entry/tokenization?lang=en>
37. Michelbacher, L. (2013) Multi-Word Tokenization for Natural Language Processing, University of Stuttgart pp 113-117
38. Infochimps (2014) Dataset English-Words List, Retrieved From <https://github.com/dwyl/english-words>

39. Team NLTK, (2019, Aug) NLTK tokenize package, Retrieved From [http://www.nltk.org/api/nltk.tokenize.html#module-nltk.tokenize\\_mwe](http://www.nltk.org/api/nltk.tokenize.html#module-nltk.tokenize_mwe)
40. Wikipedia (2019, Oct) MySQL – Wikipedia, Retrieved From <https://en.wikipedia.org/wiki/MySQL>
41. Norvig, P. (2016, Aug) How to write a spelling corrector, Retrieved From <http://norvig.com/spell-correct.html>



# Acknowledgement

Above everything else, thanks and praises to God for His unconditional love and blessings in my life.

First of all, I would like to record my million thanks to my thesis-cum-life advisor, Prof. Ok-Keun Shin, whose maddening attention to detail, which drove me to finally learn and write better. Not to forget his countless advices and nags that kept me going, and his selfless time and care to proofread my work eventhough he must be sick of it by now. Thank you and thank you, for your supervision until this very moment.

Not the least, to fellow examiners of my thesis, Prof. Hyu-Chan Park and Prof. Jang-Se Lee. My sincere thanks for the insightful comments and every brilliant feedbacks given for me to improve my writings.

Most importantly, to my parents, Kamis and Yanti; and my younger siblings, Sofiyya and Nawwaf; I am forever indebted and eternally grateful for their continuous prayers and definite love. Thank you for giving me enormous amount of supports needed in every possible way and for being behind every reason that made me who I am today.

Special thanks to my backbone, whose a devoted listener to my dilemmas and daily rants and one that always brings out the best in me, Won-Guk Choi. I can never thank you enough for the love and support given. To my dearest friends; Anna Shaidi, Eyka, Ummy, my dearest labmates; Jong-Rae Kim and Yana, thank you for the laughs and joys throughout our friendship and to everyone who gives amazing supports, thank you!

This thesis is exclusively dedicated to my little sister, thank you for being a fighter and the source of my spirit to complete my journey.

Zara Kamis signing out.