COMP4801 Final Year Project
An online food catalogue based on open crowdsourcing
(GroupName)

Supervised by Dr. T. W. Chim

FYP17024 Individual Final Report

Student: Yeung Ching Chun (3035177199)
Another Group Member: Man Ting-chun Matthew (3035178765)

Date of submission: 15/4/2018
Abstract

At present, many smartphone applications (apps) provide crowdsourced restaurant reviews functionality. People can select their destined restaurants based on the descriptions and reviews provided. However, a review process between restaurants and the app-developing company is needed for putting the restaurant on list. Moreover, these apps put the menus as photos on their searching lists, which causes ambiguity if people do not understand the language on menus. Low resolutions of photos and small font sizes bring further inconvenience in searching for specific kind of food.

An online food catalogue based on open crowdsourcing is developed as the Final Year Project for solving the aforementioned issues. Photos of menus will be uploaded by users to the web server, which is deployed on the Amazon Web Services (AWS) for Optical Character Recognition (OCR) analysis. All food items are stored in the external database in editable text format for easy searching. When users want to search a specific kind of food, translation is performed on the server and thus items of that specific food in different languages will be shown.

The deliverable is an iPhone Operating System (iOS) application written in Swift which allows users to upload the food menu to the server and get the food items as text back using Google Vision Application Programming Interface (API). Users can also search for food records by text in any language, or even by image. The app also caters for people with disability by implementing a “friendly mode” for their usage.
Acknowledgement

We would like to express our deepest appreciation for those individuals and organizations, who have kindly offer their help and support to this project.

We are highly indebted to our supervisor Dr. T. W. Chim for his opinions and guidance throughout the project. He has always spared time for our meetings and has given a lot of concrete feedbacks. We will never be able to have done this so far without his support.

We would also like to express our gratitude towards the Department of Computer Science, Faculty of Engineering, The University of Hong Kong for giving us the valuable opportunity for doing this project.
# Table of Contents

Abstract
Acknowledgement
List of Figures
List of Tables
Abbreviations
1. Introduction
2. Objective
3. Vision and Scope
   3.1 Features
   3.2 Software Limitations
4. Methodology
   4.1 System Architecture Design
      4.1.1 Presentation Tier
      4.1.2 Logic Tier
      4.1.3 Data Tier
      4.1.4 Database design
      4.1.5 Cloud Computing
   4.2 Function Design
      4.2.1 Adding & Uploading Food Records
      4.2.2 Uploading Menus for OCR Analysis
      4.2.3 Searching for Food by Text
      4.2.4 Searching for Specific Amount of Food Closest to a Fixed Budget
      4.2.5 Searching for Food by Image
      4.2.6 Login/Register
   4.3 Task Schedule
5. User Interface
   5.1 Adding & Uploading Food Records
      5.1.1 Home View of locally stored food records
      5.1.2 Input View of food records
      5.1.3 Location Search View
   5.2 Uploading Menus for OCR Analysis
   5.3 Searching for Food
List of Figures

Figure 1 The iterative cycle
Figure 2 Database design figure using Crow Foot Notation
Figure 3 System flow of uploading food record(s)
Figure 4 System flow of uploading picture for OCR Analysis
Figure 5 System flow of searching for food for different languages
Figure 6 System flow of searching specific amount of food closest to a fixed budget
Figure 7 System flow of searching food by image
Figure 8 System flow of login/register
Figure 9 User interfaces of Home View
Figure 10 User interfaces of Input View
Figure 11  User interfaces of Location Search View
Figure 12  User interfaces of OCR
Figure 13  User interfaces of Rank View
Figure 14  User interfaces of Food Detail View
Figure 15  User interfaces of Searching Filter
Figure 16  User interfaces of Searching with Translation
Figure 17  User interfaces of Searching by Image
Figure 18  User interfaces of User Profile
Figure 19  User interfaces of Login/Register
Figure 20  User interfaces of Friendly Mode

**List of Tables**

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Features for this project</td>
<td>10-11</td>
</tr>
<tr>
<td>Table 2</td>
<td>Limitations for this project</td>
<td>12</td>
</tr>
<tr>
<td>Table 3</td>
<td>Schedule for this project</td>
<td>24</td>
</tr>
</tbody>
</table>

**Abbreviations**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>AWS</td>
<td>Amazon Web Service</td>
</tr>
<tr>
<td>iOS</td>
<td>iPhone Operating System</td>
</tr>
<tr>
<td>JPEG</td>
<td>Joint Photographic Experts Group</td>
</tr>
<tr>
<td>OCR</td>
<td>Optical Character Recognition</td>
</tr>
<tr>
<td>S3</td>
<td>Simple Storage Service</td>
</tr>
<tr>
<td>SMS</td>
<td>Short Message Service</td>
</tr>
<tr>
<td>SNS</td>
<td>Simple Notification Service</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
</tbody>
</table>
1. Introduction

In the past, people had difficulty in choosing restaurants because information about restaurants was hard to access. Due to advancement in mobile technology, information related to restaurants is now easily accessible by the public with the use of smartphone applications such as OpenRice [1] and Yelp [2]. These applications can be categorized as crowdsourcing applications. Crowdsourcing is defined as a kind of activity that an individual or an organization proposes a task, and it allows the crowd to participate voluntarily by providing knowledge, experience or other resources [3]. For applications like OpenRice and Yelp, reviewing restaurants is the proposed task for the public, and users (posters) are enabled to upload restaurants reviews, ratings or photos through the applications. Afterwards, other users (viewers) can read those contents and decide which restaurant to go to.

Some of these applications categorize photos for the restaurants into either menus or photos like food and environment for users to check the menus more efficiently [4]-[5]. Being able to read the menus is crucial to users as they could get a preview of the food in advance and thus choose the restaurants they like. However, providing mere pictures of the menus is not sufficient since the qualities of images are not unified. Images with low quality greatly reduce the readability as users may not be able to see the words on a low-quality menu image. Indeed, existing applications provide little support for users with different background (e.g. users with different mother tongues). Users may not understand the language on the menu. Those applications simply categorize photos with menus and assume that users find the menus useful – but the applications can go further.

Only providing pictures of food catalogues may be undesirable for certain users, in terms of language and the ability to search for food items. First, the language of menus in the pictures may not be understandable for all users. If menus in different languages are provided separately but users only upload the Chinese version, the content cannot be understood by some foreign users. Second, the ability to search for food items is limited because the menus are in picture format, instead of editable text format that can be searched by typing in characters and words. Image qualities of the menu photos thus determine the ability to be searched, which is not
desired. Third, people with disabilities (e.g. visual impairment) may not be able to read the menus. Indeed, current applications like OpenRice and Yelp do not provide any special mode for catering the needs of the disabled.

The first two problems can be solved by OCR technology and Google Translate API. With OCR, printed or written text in images can be converted into editable text [6], allowing the text to be further processed. Therefore, users are not influenced by the qualities of menu photos for searching. And by translating the OCR scanned text into multiple languages, the language barrier can be overcome and thus solve the second problem. The last problem can be solved by implementing a “friendly mode” for those in need, with the aid of speech recognition, text-to-speech function, etc.

This report introduces a new smartphone application that integrates OCR and some Google APIs to provide online food catalogues based on open crowdsourcing. Users can upload photos of food menus, then the images can be converted into text for generating online food catalogues. Apart from generating online food catalogues, this application includes image comparison technology, which enables users to search for food by uploading an image of the food item; as well as searching with translation technology, which enables users to search for food name in any languages.

The remainder of this report proceeds as follows. First, the objective and the scope of the project are given. Next design strategies and methodology on how to implement the functions of this application are introduced. Then, the user interface, the potential risks as well as the user acceptance test are described. Lastly, this report closes with a conclusion and references.
2. Objective

The goal of this project is to develop a new smartphone application that benefits both restaurants and all users. Comparing with existing applications focusing on crowdsourcing restaurants review, this application provides features that benefit both restaurants and users.

For restaurants, online food catalogues can be generated easily by simply taking pictures of their existing food menus. Once the picture is scanned by OCR, all food items shown on the menus is available for searching. The OCR technology greatly reduces the time for manually inputting food items. Moreover, showing menus as individual food records allows easy searching for users, so as to broaden the search-ability of all the food items available in the restaurant.

For users, other than searching for ranking of food and restaurants, searching for food in any languages is also possible in this application. Translation is applied to the searching keyword, hence identical food name in different languages can be searched simultaneously. Users can search for any food item available on menus once they are uploaded. Apart from searching for food by typing, searching for food by images is possible. Users can take a picture of a food item and image comparison is applied between the picture taken and the food photos stored in the database to retrieve the information of the food item. This is especially convenient if users do not know the food name in advance. It is hoped that users with disabilities (e.g. visual impairment) should be able to use the application as well.

For us, the developers, this project allows us to explore opportunities to collaborate with existing companies offering crowdsourcing applications. We can use this project to demonstrate how the new functions may benefit users, and then we can collaborate with other companies to incorporate more functions into our application or their application.
3. Vision and Scope

This project develops an iOS application which allows users to upload food catalogue of restaurants and search for food more efficiently. Users upload food records instead of restaurant records in this application.

Unlike existing applications that use photos to show the menus, this application uses editable text for menu display. As a result, it is more efficient for users to look for menus in multiple languages or searching. Consequently, this application facilitates users in searching for not only restaurants, but also food. Below are the features and limitations of the application.

3.1 Features

Below is the table showing the features available to the users (see Table 1). The features are divided into four types: Uploading (for OCR Analysis), Searching (with Translation), Friendly Mode and Login/Register. Each type has one or more features. The detailed implementations are illustrated in the later section 4.2 Function Design.

<table>
<thead>
<tr>
<th>Type</th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uploading</td>
<td>Upload food menu for OCR Analysis</td>
<td>Food menu photo is uploaded and stored in the server. The photo is then passed to the Google Vision API for OCR Analysis. Scanned food records consist of food names and their corresponding prices will be returned to the user’s application afterwards. Users can view their uploaded menu photos in the user view.</td>
</tr>
<tr>
<td></td>
<td>Edit information of food items</td>
<td>After uploading menu for OCR Analysis, food records are returned to the user’s application. The information of each of the records can be edited by the user. Food name, food price, restaurant name and restaurant location are the compulsory fields of a food record.</td>
</tr>
<tr>
<td></td>
<td>Give tags to food items</td>
<td>Tags can be given to enrich the search-ability of the food.</td>
</tr>
<tr>
<td></td>
<td>Give ratings to food items</td>
<td>Rating from 1 to 5 can be given for each food.</td>
</tr>
<tr>
<td></td>
<td>Upload food pictures</td>
<td>Food picture can be attached for each of the food record. Users can view their uploaded photos in the user view.</td>
</tr>
<tr>
<td>Searching</td>
<td>Search for food by name/tag</td>
<td>Keyword can be entered to search for the corresponding food.</td>
</tr>
<tr>
<td></td>
<td>Search for food by restaurant name/location</td>
<td>Keyword can be entered to search for the corresponding food of the restaurant. Apple Map showing the estimated</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Search for food by price range</td>
<td>Range of price can be adjusted to search for the corresponding food.</td>
<td></td>
</tr>
<tr>
<td>Search for specific amount of meals closest to a fixed budget (Knapsack Problem)</td>
<td>Given a specific amount of meals and a specific budget, the app can determine a specific amount of meals so that the total price is closest to the specific price.</td>
<td></td>
</tr>
<tr>
<td>Search for food by image (AR)</td>
<td>Users can take a photo and find out what the food is in a specific restaurant. The food name and price will be displayed in the form of AR Text.</td>
<td></td>
</tr>
<tr>
<td>Search for food by name/tag</td>
<td>Keyword can be entered to search for the corresponding food.</td>
<td></td>
</tr>
<tr>
<td>Search for food by restaurant name</td>
<td>Keyword can be entered to search for the corresponding food of the restaurant. Apple Map showing the estimated time and route is provided.</td>
<td></td>
</tr>
<tr>
<td>Adjust Word Size</td>
<td>Users can adjust the word size of words in the view.</td>
<td></td>
</tr>
<tr>
<td>Speech to Text</td>
<td>Searching keyword can be inputted by speech recognition, the keyword can then be used to search for food by name/tag/restaurant name.</td>
<td></td>
</tr>
<tr>
<td>Text to Speech</td>
<td>Synthesized audio can be outputted for each food record, the information of a food record, and for each OCR record.</td>
<td></td>
</tr>
<tr>
<td>Login/Register &amp; Verify for an account</td>
<td>Phone number is used for unique identification. Each account must then be verified by a one-time passcode sent by the web server via AWS Simple Notification Service (SNS). Users can auto-login without receiving one-time passcode from AWS SNS once verified.</td>
<td></td>
</tr>
</tbody>
</table>

*Table 1. Features for this project.*
3.2 Software Limitations

Below is the table showing the limitations of the APIs, the web services used and the scope of this app (see Table 2). These limitations define the bottleneck of the functionalities.

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Limitation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Picture Dimension for OCR Analysis</td>
<td>Picture size should be minimum of 640 x 480 pixels, and recommended to be of 1024 x 768 pixels [7]</td>
</tr>
<tr>
<td></td>
<td>Picture Size for OCR Analysis</td>
<td>Maximum of 4 MB is supported by Google Vision API [7]</td>
</tr>
<tr>
<td></td>
<td>File Format for OCR Analysis</td>
<td>Only JPEG, PNG8, PNG24, GIF, Animated GIF (first frame), BMP, WEBP, RAW, ICO are supported [7]</td>
</tr>
<tr>
<td></td>
<td>Text Languages for OCR Analysis</td>
<td>56 Languages are supported [8]</td>
</tr>
<tr>
<td>Web Services</td>
<td>Text Languages for Google Translate API</td>
<td>104 Languages are supported [9]</td>
</tr>
<tr>
<td></td>
<td>Picture(s) Storage for each user</td>
<td>Maximum storage size of 5GB is supported by AWS Simple Storage Service (S3) [10]</td>
</tr>
<tr>
<td></td>
<td>Language for Speech Recognition</td>
<td>English</td>
</tr>
<tr>
<td></td>
<td>Maximum Budget entered in the search filter</td>
<td>$100000</td>
</tr>
</tbody>
</table>

*Table 2. Limitations for this project.*
4. Methodology

The iterative cycle

![Image of the iterative cycle](image)

**Figure 1. The iterative cycle [12].**

For this project, an iterative and incremental approach has been used, and repeated iterative cycles have been carried out (see Figure 1). In each iteration, part of the system is developed, and it is a functional product on its own to prove the iteration is completed. Apart from providing a working deliverable, each iteration builds on top of the previous iteration such that the system grows incrementally.

The iterative and incremental approach has been used instead of some traditional approaches such as the Waterfall model. The major benefit of the iterative and incremental approach is that early feedback can be obtained from stakeholders, such as the supervisor or potential users. Stakeholder’s requirements can be better understood with those feedback, allowing modifications to be made to the system in the next iteration to fulfil those requirements. The traditional Waterfall model contains only one iteration of the iterative and incremental approach [13], and the product cannot be seen by stakeholders before the implementation phase, therefore it is more likely that stakeholder’s expectations cannot be met by the system.
4.1 System Architecture Design

For this project, a three-tier architecture has been used. Multi-tier architecture (also known as N-tier architecture) means that the system is divided into multiple parts physically and logically, and each part is responsible for a specific type of tasks [14]. The traditional basic client-server architecture is regarded as a two-tier architecture, which consists a client tier for user interface and data processing, and a data tier for data storage and retrieval [15]. In a typical three-tier architecture, the tiers are the presentation tier, the logic tier and the data tier [16]. The presentation tier consists of the user interface and some operations to present meaningful result to users, the logic tier is for processing of data or calculation, and the data tier is for data storage and retrieval.

A three-tier architecture has been used in this system because it has better scalability and security than a two-tier architecture [17]. Scalability is defined as the ability to support increasing amount of work [18]. In a three-tier architecture system, more resources can be easily allocated to the logic tier, allowing the overall system to have better performance. Security is stronger because data access can only be performed through the server but not directly by the users.

The three-tier architecture is like a bank queueing system. The presentation tier consists of all the customers with different requests, the logic tier is all the counters that serves the customers, and the data tier is the bank’s centralized database with all customer’s data. During peak hours, the queueing system may have many customers, just like many users may be using our application in certain hours. The queueing system can add more counters during peak hours, just like more resources can be added to the logic tier any time to handle more user requests. The three-tier system is secure because users cannot directly manipulate the data, which is similar to bank system that customers can only manage their account through the counters or standardized ATM machines.
4.1.1 Presentation Tier

For the presentation tier, an iOS application written in Swift has been used in this system. iOS is less fragmented when comparing to Android [19]-[20], meaning that more iOS users have updated to the latest operating system. Since certain functionality may only exist in more recent operating system, iOS is easier to manage. Apart from operating system, less device models exist in iOS platform, such that it is easier to cater for all screen size or ratios of the models. Regarding programming language choices, Swift and Objective-C are the two major programming languages for iOS, but Swift has been used because it has better performance and it is easier to maintain than Objective-C [21].

4.1.2 Logic Tier

For the logic tier, Node.js has been used for server development. Some other popular programming languages for servers are PHP, Django and Rails. However, Node.js has better performance and database integration than PHP, Django, Rails servers [22]. Thus, to achieve better performance, Node.js has been used as the programming language on the server side.

4.1.3 Data Tier

For the data tier, a MySQL relational database has been used. Databases can be categorized into either relational database or non-relational database (also known as NoSQL database). Relational database has been used because it has better data integrity when comparing to non-relational database due to usage of foreign key. Apart from data integrity, using relational database is also better because it creates less problems than non-relational database [23].

A MySQL relational database has been used because it is a open-source software, meaning that it can be used or modified freely [24]. Some other popular relational databases are Oracle RDBMS or Microsoft SQL Server [25]. However, license fees are required for these databases. Due to budget limitation, a MySQL relational database has been used.
4.1.4 Database design

Multiple tables and relationships exist in a relational database.

First, the “Restaurants” table stores the information of a restaurant. The restaurant name and the location of the restaurant are stored (see Figure 2). The field “restaurant_id” is auto generated which is unique for each restaurant branch. Both restaurant name and location are stored such that multiple branches of the same restaurant can be identified.

Second, the “Users” table stores the information of a user. The “user_id”, “phone_number” and “bucket_name” of users are stored (see Figure 2). The field “user_id” is an auto generated number that is unique for each user, and it is used to uniquely identify a user. In addition to the “user_id”, the “phone_number” is the unique phone number used by the user for login, and the

Figure 2. Database design figure using Crow Foot Notation [26].
“bucket_name” is the image storage directory path for the user to store the photos uploaded by the user. Each user should have his/her unique phone number and bucket name.

Third, the “Food” table stores the information of food uploaded by users. Each row represents a food record uploaded by a user. The “restaurant_id”, tags, “user_id”, food name, price, food ratings and “photo_filename” given to the food are stored (see Figure 2). The field “tags” is like the hashtag function in Instagram [27], and it has been included for users to search for food items easily by using the “tags”. The field “photo_filename” is the image storing address for the food photo.

Fourth, the “Food_Validation” table stores the voting towards the credibility of a food photo. Each row represents a vote of a user. The “user_id”, “food_id” and a binary variable “isValid” are stored (see Figure 2). Each record in the “Food” table can be linked to many records of “Food_Validation”. The binary value “isValid” represents a vote from a user: 1 represents that voter thinks the food photo matches the food name, while 0 represents a mismatch.

Finally, the “Food_Label” table stores the descriptions of food. Each row represents a description of a food item. The “user_id”, “food_id”, “description” and “score” are stored (see Figure 2). Each record in the “Food” table can be linked to many records of “Food_Label”. The field “description” stores the words/phrases to describe the food. The field “score” stores the reliability of the description.

4.1.5 Cloud Computing

Both the server and database have been deployed on the Amazon Web Services (AWS), which is a cloud computing platform [28]. A cloud computing platform is used because it can be easily managed by multiple users, instead of using a computer as a server, which can only be operated by one member. There are many cloud computing platform existing in the market, and the most popular ones are AWS, Microsoft Azure and Google Cloud Platform [29]. AWS has been used because it provides the best documentations and most functions among the three platforms [30] - [31].
4.2 Function Design

Functions of this application are divided into six main aspects: Adding & Uploading Food Records, Uploading Menus for OCR Analysis, Searching for Food by Text, Searching for Specific Amount of Food Closest to a Fixed Budget, Searching for Food by Image, and Login/Register. In the remaining sections of this part, users’ step-by-step workflow and the corresponding back end working components are shown.

4.2.1 Adding & Uploading Food Records

![System flow of uploading food record(s).](image)

*Figure 3. System flow of uploading food record(s).*

User can upload the locally-stored food record(s) to the server (see Figure 3, step 1). Next, the server adds those record(s) to the database (step 2). If any food image is attached to the food record, the image is redirected and stored in the uploader’s S3 bucket (step 3). The Uniform Resource Locator (URL) pointing to the image in the S3 Storage is passed to the Google Vision API for object detection (step 4). A set of object descriptions with high similarities, and the image path are thus stored in the database for later retrieval (step 5).

Image is stored after adding the related food record. It is because the image file name is named by the food ID. This naming facilitates the debugging process since related images can be easily
found by their file names. Cross-checking among database records and S3 bucket photos has become easier.

4.2.2 Uploading Menus for OCR Analysis

*Figure 4. System flow of uploading picture for OCR Analysis.*

Users can upload the photo of a food menu from the application to the server (see Figure 4, step 1) by taking a picture using the camera instantly or selecting a photo from the photo library. Afterwards, the server finds the corresponding AWS S3 user bucket in the Database (Step 2), and stores the photo in the corresponding bucket for the user in the Amazon S3 Storage (Step 3). After that, the server passes the URL pointing to the photo in the S3 Storage to the Google Vision API for OCR analysis (Step 4). Lastly, the Google Vision API returns the analysed data and the server passes it back to the application.

Google Vision API has been used to perform OCR analysis. There are some other OCR technologies such as Tesseract OCR [32]. However, Tesseract OCR is an Artificial Intelligence(AI) program and the program needs to be trained to get accurate results. Due to time limitation and lack of experience in AI program training, Google Vision API, which does not require any AI programs training, has been used in this project.

The client application is not allowed to directly connect to the Google Vision API. The major reason is that a secret passcode is required when accessing Google Vision API. If the passcode is
included in the client application, hackers may reverse engineer the client application to see the programming codes [33], which includes the passcode. Instead of putting the passcode in the client application, it is stored in the server such that other users cannot use it for their own applications. Since the passcode is stored in the server, the server is used to connect to the Google Vision API.

4.2.3 Searching for Food by Text

![System flow of searching for food for different languages.](image)

**Figure 5.** System flow of searching for food for different languages.

User can search for specific food in the application. The filter values (e.g. price range, food name) are sent from the client application to the server (see Figure 5, Step 1). Then, the food name and restaurant name and restaurant location (if any) are translated into different languages using Google Translate API (Step 2), and the translated values in multiple languages are passed to the Database for searching (Step 3). Matched records in multiple languages are therefore shown on the application. Translation allows the Database to show more results with the same meaning and minimize the language barrier.

The client application is not allowed to connect to the Google Translate API directly due to similar security reason as mentioned in 4.2.2. Apart from security, since the translated text is less useful for the user but only for the system to search for food in multiple languages, the translated text is not sent back to the user, and thus reduces network traffic.
4.2.4 Searching for Specific Amount of Food Closest to a Fixed Budget

**Figure 6.** System flow of searching specific amount of food closest to a fixed budget.

User can search for specific amount of food which is closest to a fixed budget in the application. The filter values (e.g. price range, food name), budget and specific amount of food are sent from the client application to the server (see Figure 6, Step 1). Then, the food name, restaurant name and restaurant location (if any) are translated into different languages using Google Translate API (Step 2), and the translated values in multiple languages are passed to the Database for searching (Step 3). The self-edited dynamic programming algorithm is used to find the required amount of food which is the closest to the budget (step 4).

Finding specific amount of food closest to a fixed budget is a knapsack problem. The knapsack problem is a NP-complete problem, which means that there is no known algorithm both correct and is in polynomial-time in all cases [34]. In Knapsack problem, each item is with a weight and a value. A weight limit is given so that the total weight of items is under the weight limit while the total value of items is at its maximum. By letting the weight of each food equals to 1 and the value of each food equals to its price, the problem can be solved using dynamic programming. Solving by dynamic programming can avoid re-computations of same sub-problems by constructing a matrix in bottom up manner [35].
4.2.5 Searching for Food by Image

![System flow of searching for food by image.](image)

**Figure 7. System flow of searching for food by image.**

User can search for food name and price by image in the application. An image and an optional keyword for restaurant name are sent from the client application to the server (see Figure 7, Step 1). Then, the image is saved in the S3 Storage temporarily and its URL is passed to the Google Vision API for object detection (step 2). A set of object descriptions is thus generated. After that, the set of descriptions and the optional keyword for restaurant name are passed to the Database for filtering unmatched records (step 3). Only food records with matched descriptions are selected (descriptions generated for food records when they are uploaded, see section 4.2.1 Adding & Uploading Food Records). Lastly, the server compares the uploaded image with every food image under the matched records using hamming distance (step 4). The food name and price of the food image with the highest similarity is returned (step 5) and are displayed as AR Text on the app (step 6).

Google Vision API has been used to perform object detection. There are other object detection technologies such as deep learning. However, deep learning program needs to be trained to get accurate results [36]. Due to time limitation and lack of experience in deep learning program training, Google Vision API, which does not require any training, has been used in the use of object detection.

Image comparisons on large number of images require exorbitantly long processing time. To reduce the processing time for this function, 3 methods are introduced:
1) To reduce the number of image comparisons being made, the object detection and the restaurant name are used for filtering unwanted records.

2) To reduce processing time on a huge amount of image comparisons, the comparisons are made in parallel.

3) To reduce processing time on an image comparison, hamming distance is used as it allows fast and accurate comparison on the binary features, and uses few amount of resources only [37].

With these 3 methods, the processing time on this function is greatly reduced.

### 4.2.6 Login/Register

![System flow of login/register.](image)

Figure 8. *System flow of login/register.*

User needs to register before uploading menus and giving ratings to food and restaurants. User must first enter his or her own phone number (see Figure 8, step 1). Next, the phone number is passed to the server. After that, a confirmation message containing a one-time passcode is sent to the user through Short Message Service (SMS) by the AWS Simple Notification Service (SNS) (Step 2). If the client enters the passcode correctly (Step 3), a new user record will be inserted into the server database (Step 4). The one-time passcode is used to ensure that the phone number entered by the user is valid. Hence the database could use the phone number for user identification.
Once verified, the application keeps the user logged in. The user can auto-login with the verified phone number any time.

To improve security, hashing is applied to the verification passcode before it is stored or sent from client application. Hashing means converting a string into a value by some algorithms, and it is impossible to regenerate the value back to the original string [38]. By using hashing, even if the server is accessed by malicious users, they cannot regenerate the user’s verification code and login to those accounts. In this application, SHA512 hashing algorithm is used; as it is secure by showing no successful attacks in terms of collision resistance and preimage resistance [39].

4.3 Task Schedule

<table>
<thead>
<tr>
<th>Task</th>
<th>Completion Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>First iteration</td>
<td>30 September 2017</td>
</tr>
<tr>
<td>• Interface to list all stored food records</td>
<td></td>
</tr>
<tr>
<td>• Interface to add or edit food records</td>
<td></td>
</tr>
<tr>
<td>• Set up web server and database</td>
<td></td>
</tr>
<tr>
<td>• Connection between server and database</td>
<td></td>
</tr>
<tr>
<td>• Connection between client application and the server</td>
<td></td>
</tr>
<tr>
<td>Second iteration</td>
<td>15 November 2017</td>
</tr>
<tr>
<td>• Interface for location search</td>
<td></td>
</tr>
<tr>
<td>• Interface for food ranking</td>
<td></td>
</tr>
<tr>
<td>• Upload picture for OCR analysis</td>
<td></td>
</tr>
<tr>
<td>Third iteration</td>
<td>31 December 2017</td>
</tr>
<tr>
<td>• Search for food by text with translation</td>
<td></td>
</tr>
<tr>
<td>• Login function with hashing</td>
<td></td>
</tr>
<tr>
<td>Fourth iteration</td>
<td>31 January 2018</td>
</tr>
<tr>
<td>• Navigation to restaurants</td>
<td></td>
</tr>
<tr>
<td>• Interface for User Profile</td>
<td></td>
</tr>
<tr>
<td>Fifth iteration</td>
<td>20 February 2018</td>
</tr>
<tr>
<td>• Search for food by images with AR</td>
<td></td>
</tr>
<tr>
<td>Sixth iteration</td>
<td>15 March 2018</td>
</tr>
<tr>
<td>• Search filter with Knapsack Problem</td>
<td></td>
</tr>
<tr>
<td>Seventh iteration</td>
<td>31 March 2018</td>
</tr>
<tr>
<td>• Friendly Mode</td>
<td></td>
</tr>
<tr>
<td>Overall Testing and Debugging, User Acceptance Test</td>
<td>15 April 2018</td>
</tr>
<tr>
<td>Deliverable of Final Report</td>
<td></td>
</tr>
<tr>
<td>Final Presentation</td>
<td>18 April 2018</td>
</tr>
<tr>
<td>Exhibition</td>
<td>2 May 2018</td>
</tr>
</tbody>
</table>

*Table 3. Schedule for this project.*
5. User Interface

The user interface is illustrated with five perspectives of use: Adding & Uploading Food Records, Uploading Menus for OCR Analysis, Searching for Food, Login/Register, and Friendly Mode. In the remaining subsections of this part, detailed views and control procedures of each perspective of use are introduced.

5.1 Adding & Uploading Food Records

5.1.1 Home View of locally stored food records

User’s record table is shown in Figure 9. The essential information for each food record is the food name, restaurant name with its location and the price of the food. Records with missing information are considered as uncompleted, and they are highlighted in yellow and the corresponding missing field is presented in red (see Figure 9a). When the user clicks on the “+” button, 3 options are prompted. The user can choose between input by typing (“Type”) or input...
by OCR (“Scan”) (see Figure 9b). An online tutorial will be given upon clicking the “Help” button.

All records shown in this view is stored locally and persistently. The users can retrieve the records after they exit the application. When the user click on the top right “cloud” button, all completed food records with the essential information is uploaded to the server.

5.1.2 Input View of food records

![Figure 10. User interfaces of Input View.](image)

This view is shown whenever the user click on a record (see Figure 10a, shown when click on first row in Figure 9) or the choose “Type” option. Users could add or edit the food name, price, restaurant name with its location, tags, ratings from 1 to 5 and the food photo in this view. When user clicks on the picture, another image can be selected from the album or taken by camera. If no picture is selected, a temporary image will be displayed to allow image adding by users (see Figure 10b).
5.1.3 Location Search View

![User interfaces of Location Search View.](image)

This view is shown when the user clicks on the location row within the detailed view (see Figure 11a, shown when click on location row in Figure 10). The table automatically updates with latest search result upon typing (see Figure 11b). The arrow button on the top-left of the interface is for clearing the text field and perform searching based on user’s current location. After selecting on the location, the location name and location address, the information are presented back in the location row.
5.2 Uploading Menus for OCR Analysis

When user clicks on the “Scan” button in the view of Figure 12a, user can choose between taking a new picture with camera or selecting an image from gallery. The image captured or selected is then uploaded to the server for OCR analysis. After the analysis, the result is shown in the format as shown in Figure 12b, which contains a table of food name and price, as well as another table of food price. The text field with placeholder “Add food name” is for adding a new empty field in the table of food name, while the text field with placeholder “Add food price” is for adding a new empty field in the table of food price. As sometimes the OCR is not accurate, some food name or price may be missing or misinterpreted from the OCR result. These two fields can allow user to add back the correct ones. Moreover, sometimes some food names or prices are mismatched, user can easily drag and drop the correct price back to the table of food name as shown in Figure 12c. By using the drag-and-drop approach, user does not need to type in the correct price for correction each time, which is more user friendly and it minimizes the inputting time.
The first row of the table from Figure 12b is used for selecting a location for the restaurant. It works the same way as mentioned in Section 5.1.3 Location Search View.

5.3 Searching for Food

5.3.1 Rank View

The Rank view in Figure 13a is shown when the user clicks on the middle icon of the tab bar in the bottom of the screen. The Rank View shows the ranking of all food sorted according to their average ratings in descending order. To minimize the loading time, the food photos are loaded asynchronously so that other food information can be loaded first. When the food photos are loading, a temporarily image is shown on each food record (see Figure 13a). Once the food photo is loaded, the photo will be shown on the corresponding food record (see Figure 13b). A different image is used if there is no photo uploaded for this food record. An example is shown on the first record of both Figure 13a and Figure 13b.

Figure 13. User interfaces of Rank View.
5.3.2 Food Detail View

![User interfaces of Food Detail View.](image)

When the user clicks on one of the food record, a food detail view is shown (see Figure 14a, shown when click on any row in Figure 13a). The food detail view contains all information of the food record given by users. Since we treat food records with the same food name, restaurant name and restaurant location as the same as mentioned in section 4.1.4 Database design, it is possible that the same food record can be uploaded by multiple users. Hence there are differences between those records on the tags, rating and attached image. For tags, all tags given to the food record from different uploads are displayed. For the rating, the average rating given to the food record is shown. For image, if there is/are photo(s) uploaded, user can swipe on the image to see other images for the food record. An example is shown on Figure 14 where the 1st image is shown on Figure 14a and the 2nd image is shown on Figure 14b. Full screen viewer with zooming is also supported by tapping on the image.

Credibility on food images are manifested as sometimes users may upload wrong food image. The progress bar below the food image (see Figure 14a) shows the percentage of users voting for the option: “The food image matches its food name”. For instance, “100%” means that for every
user voting for this image, he/she thinks that the food image matches its food name (see Figure 14a). If there is no one voting for the image, a “Not Verified” message is shown below the food image (see Figure 14b). The images in Food Detail View are sorted according to their credibility in descending order, while food records in Rank View only show the image with highest credibility. Users can thus easily determine what the food should look like based on this percent.

Furthermore, a navigation view is shown when user clicks on the restaurant location (text highlighted in blue in Figure 14a). The suggested route, the required walking distance and the required walking duration from the user’s current location are displayed (see Figure 14c).

5.3.3 Search Filter View

This view is shown when the user clicks on the top left “filter” icon of Rank View (see Figure 15a, shown when click on the top left icon in Figure 13a). By typing in the filtering values and clicking the “Apply” button (see Figure 15a), Rank View with filtered records will be shown. By clicking the top left “Reset” icon, all filtering values are cleared and Rank View with all food records will be shown. There are two main filtering methods in this view:

The first one is to search for specific amount of food which is closest to a fixed budget. The fixed budget and the amount of food are inputted in the two text fields under the text “Meal planning”
(see Figure 15a). Records with maximum total price under the budget are returned (details described in section 4.2.4 Searching for Specific Amount of Food Closest to a Fixed Budget). An example is illustrated in Figure 15b by typing “$16” as the fixed budget and “3” as the number of meals for the budget. 3 records with the total price $16 are shown.

The second one is to search for food by its information. User can type in food name, restaurant name, restaurant address or price range for each food in the text fields provided under the text “Filtering” for eliminating unwanted records (see Figure 15a).

5.3.4 Search with Translation

Figure 16. User interfaces of Searching with Translation.

Translation is automatically applied to the search keyword(s) inputted by the user. The server automatically detects the language of the keyword(s) and translates it to English and Chinese as they are the official languages in Hong Kong. Despite the overwhelming Cantonese speakers, English is perceived as the most common language used by the government, as well as the legal, professional and business sectors [11]. The server then queries for all the records that includes the original keyword or the translated keywords. Finally, only the matched records are presented in the table.
As shown in Figure 16a, when the user inputs “oolong” in the food name/tag search bar of the rank view or in the corresponding text field of the search filter view, record that includes the translated version of “oolong” (translates to “烏龍” in Chinese) is shown. In Figure 16b, the user inputs “蛋糕” in the restaurant name search bar of the rank view or in the corresponding text field of the search filter view, and the food records with restaurant name “Cake” are shown because the English version of “蛋糕” is “cake”. As shown in Figure 16c, the Korean word “사과”, which is “apple”, is inputted by the user, and the record with food name that includes “apple” is shown.

5.3.5 Search by Image

![Image of user interfaces for searching by image]

Figure 17. User interfaces of Searching by Image.

Camera view is shown when the user clicks on the icon at the top right corner of Rank View (Rank view is shown on Figure 13a). When user points the camera to the food and tap on the screen, the view as illustrated in Figure 17a is shown. A food image is randomly selected and displayed. User needs to vote whether he/she thinks the food image matches its name. Users’
voting is used for manifesting the credibility of the food photo(s) (as mentioned in section 5.3.2 Food Detail View).

After voting, the food name and price of the food is displayed on the screen as an AR Text. User can enlarge or shrink the AR text for easy reading.

5.4 User Profile

![User interfaces of User Profile.](image)

The User Profile view in Figure 18a is shown when the user clicks on the rightmost icon of the tab bar in the bottom of the screen. The User Profile view shows all the menu and food photos the user has uploaded. Full screen viewer with zooming is supported by tapping on the image. In the full screen viewer, the time of uploading, and the necessary information are shown for each photo. For example, the time of uploading is shown for menu photos (see Figure 18b). And the time of uploading, the food name, the restaurant name with its location are shown for food photos (see Figure 18c).
By clicking on the icon at the top right corner in the full screen view, user can share the photo to anyone (e.g. by AirDrop [40], email, social media sites, etc.) and do any manipulation to it (e.g. add to iCloud Drive, Copy, Print, etc.)

5.5 Login/Register

![Figure 19. User interfaces of Login/Register.](image)

After the user has inputted their phone numbers into the application (see Figure 19a), the phone number is passed to the web server. Upon receiving the phone number, the web server delivers a one-time passcode to the user through SMS message for verification (see Figure 19b). To enhance security, the passcode is only stored in the server side for 15 minutes only. If the user does not enter the passcode within 15 minutes, a passcode is needed for the user to login to the application. The method mentioned above limits the time for guessing the passcode, which leads to stronger security.

The user phone number is stored in the “Keychain” after the verification process. Keychain is a service API provided by Apple which encrypts data and stores it in the file system [41]. It ensures that only authorized apps can gain access to a data stored by the keychain, meaning that
only this application could gain the user’s stored phone number. The user can thus safely auto-login to the application afterwards even if the application is closed.

Each time when user launches this application, the home view as shown in Section 5.1.1 is shown if the phone number is stored; and if not, the login View will be shown (see Figure 19a). Logout button is also implemented to destroy all the “Keychain” items created in the User Profile view such that the user needs to be verified again after pressing the logout button. (see Figure 18a, highlighted in red at the bottom).

5.6 Friendly Mode

![Figure 20. User interfaces of Friendly Mode.](image)

This mode is used to facilitate people with disability. The view is shown when the user clicks on the button with text “Friendly Mode” at the top left corner of User Profile view (see Figure 20a, shown when click on the button at the top left corner in Figure 18c). When the user clicks on one of the food record, a friendly food detail view is shown (see Figure 20a, shown when click on any row in Figure 20a). And by clicking the camera icon on the top right corner in Figure 20a,
the user can upload menu photo for OCR analysis. The OCR analysed records are thus returned as shown in Figure 20c.

In this mode, basic operations like searching by food name/tag or restaurant name, viewing food details, uploading photos for OCR analysis are provided. And the enhancements are made as follows:

- Font sizes of the text can be adjusted by pressing the “-” and “+” button at the bottom (see Figure 20a, 20b, 20c).
- Speech recognition is provided by tapping on the microphone image button (see Figure 20a, located on the right of the search bars). User can input the searching keyboard by voice.
- Text to Speech function is provided such that synthesized audio can be outputted for each food record (see Figure 20a), the information of a food record (see Figure 20b) and for each OCR record (see Figure 20c).

Adjustable font size is particularly useful for people with visual impairment; speech recognition is beneficial for people with visual impairment and physical impairment; and text to speech function is helpful for people with cognitive impairment [42]. With these additional features, the customer base of the app can be widened by taking care of the special needs of people with different disabilities.
6. Encountered Challenges

In this section, the encountered challenges and the corresponding solutions used are described. These challenges do not affect our future planning as the problems are solved or alternatives have been found.

6.1 OCR Accuracy

**Problem:**
OCR is not with 100% accuracy. Typical frameworks like Google Vision API and Tesseract OCR give an accuracy rating on the scanned text. Factors affecting the accuracy include file size, image resolutions or text layouts [7]-[8]. Given that this application generates food catalogue by open crowdsourcing, it is hard to put restrictions on the images uploaded by users.

**Solution:**
After the image is scanned for OCR analysis, our server responds back the recognized text to the client application. Users can make final adjustment to the data.

6.2 Map Usage

**Problem:**
Apple Map does not always return the most accurate current location, especially in Hong Kong because there are a lot of buildings, and multiple restaurants may be in the same building, but Apple Map can only tell user’s location by the GPS coordinates. When users try to upload the location of a specific restaurant, one may upload wrong location. This limitation hinders the future searching function.

**Solution:**
A search field with auto-update functionality is provided so that users can enter part of the place name or address and then choose the location from the list of auto-update suggestions for the corresponding restaurant by Apple Map. Hence, accurate restaurant locations can be chosen.
6.3 Response Rate on OCR & Searching

*Problem:*
Photo uploading and downloading hinders the response rate the most. On one hand, menu photos need to be uploaded for OCR analysis. On the other hand, food photos need to be downloaded each time when the user searches. Frequent transferring of photos increases the loading time, which worsens the user experience.

*Solution:*
JPEG format is used for uploading and storing the photos in server. Tests have been carried out and no negative impact was made for the OCR analysis. But the photo file size has been reduced from about 2MB to around 300KB, and the uploading time has been reduced from roughly 1 minute to approximately 10 seconds.

The food photos are downloaded asynchronously so that words are loaded first while searching. The loading time is now be greatly reduced to 1 to 2 seconds.
7. User Acceptance Test

Several schoolmates and family members have been found for the user acceptance test. They are required to use the perspective of a normal person, a visually impaired person (by wearing translucent eye masks) and a physically impaired person (by putting weights on their arms to restrict their hand and arm movements) to test this app.

The results are positive and the results show that our application provides an accessible environment for the disabled. They also think that our application is useful as an online food catalogue. The detailed results are as follows:

7.1 Perspective of a normal person

The testers found the app really useful. They think that the idea of food searching instead of restaurant searching is good, as most of the time they would like to search for a type of food, rather than finding a specified restaurant. Moreover, they give high appreciations to the OCR analysis and the function of searching for food by image. They think the technology is interesting and convenient. They have never thought the technology can be used in this way.

7.2 Perspective of a visually impaired person

The testers found the word size adjustment and the function of text to speech particularly useful, especially for the older testers who are above 50 years old. Adjusting the word size really enhances the readability. If they think the words are still too small to read, they could still use the text to speech function. With this function, they do not need to read the tiny words but just listen to the audio generated. This makes things rather simple for a visually impaired person.

7.3 Perspective of a physically impaired person

The testers found the function of speech to text useful. As their hand and arm movement are restricted, typing the keyword for searching is time-consuming. But with the function of speech to text, they could search for food records easily. They agree that this function is beneficial to a physically impaired person.
8. Conclusion

Uploading food items by simple scanning, multi-language support and AR Text information display have become the uniqueness of our project when comparing with other similar applications worldwide. It is anticipated that this application will be highly useful for both restaurants and customers, as it turns finding restaurants into a two-way conversation. On one hand, customers can contribute to the application by simply taking a photo for OCR analysis. This encourages customers getting involved if they want to recommend some good restaurants to their friends, instead of being told by the application passively. On the other hand, restaurant owners could enjoy free labour (customers) to promote their restaurants. Both restaurants and customers get advantages from this application.

There are solutions coming up for the risks mentioned above already, and positive results are obtained in the user acceptance test. The only limitation is from the external services, such as Google’s API. Enhancing the algorithm and user experience will be the focus in the future work, such that minimum user fine-tuning is needed to amend any errors caused by the APIs. OCR is hoped to be fully utilized so that it is likely that this application could be promoted to the world in the future.
9. References


