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

(Group 1)

Supervised by Dr. T. W. Chim

FYP17024 Interim Report

Man Ting-chun Matthew (3035178765)
Yeung Ching Chun (3035177199)

Date of submission: 21/1/2017
Abstract

At present, many smartphone applications (apps) provide crowdsourced restaurant reviews functionality. People can select their favourite restaurants based on the descriptions and reviews provided. However, a review process 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.

Currently, 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 type in keyword in any language to filter the uploaded food records. In later stage, searching for restaurant, searching by food image, and navigation to restaurants will be implemented. The user interface will be further developed to enhance user experience.
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 2
Acknowledgement 3
List of Figures 5
List of Tables 6
Abbreviations 6
1. Introduction 7
2. Objective 9
3. Vision and Scope 10
   3.1 Features 10
   3.2 Software Limitations 11
4. Methodology 12
   4.1 System Architecture Design 13
      4.1.1 Presentation Tier 14
      4.1.2 Logic Tier 14
      4.1.3 Data Tier 14
      4.1.4 Table design 15
      4.1.5 Cloud Computing 16
   4.2 Function Design 17
      4.2.1 Adding & Uploading Food Records 17
      4.2.2 Uploading Menus for OCR Analysis 18
      4.2.3 Searching for Food by Text 19
      4.2.4 Login/Register 20
5. Current Progress 22
   5.1 Adding & Uploading Food Records 22
      5.1.1 Home View of locally stored food records 22
      5.1.2 Input View of food records 23
      5.1.3 Location Search View 24
   5.2 Uploading Menus for OCR Analysis 25
   5.3 Searching for Food by Text 26
      5.3.1 Rank View 26
5.3.2 Search with Translation 27
5.3.3 Search with Tags 28
5.4 Login/Register 29

6. Encountered Challenges 31
   6.1 OCR Accuracy 31
   6.2 Map Usage 31
   6.3 Response Rate on OCR & Searching 32

7. Future Planning 33
   7.1 Searching for Menus for Restaurants 34
   7.3 Navigation to Restaurants 35
   7.4 Boost Business Potential 36

8. Conclusion 37

9. References 38

List of Figures

Figure 1 The iterative cycle 12
Figure 2 Database design figure using Crow Foot Notation 15
Figure 3 System flow of uploading food record(s) 17
Figure 4 System flow of uploading picture for OCR Analysis 18
Figure 5 System flow of searching for food for different languages 19
Figure 6 System flow of login/register 20
Figure 7 User interfaces of Home View 22
Figure 8 User interfaces of Input View 23
Figure 9 User interfaces of Location Search View 24
Figure 10 User interfaces of OCR 25
Figure 11 User interfaces of Rank View 26
Figure 12 User interfaces of Searching with Translation 27
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</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>11</td>
</tr>
<tr>
<td>Table 3</td>
<td>Future Working Plan</td>
<td>34</td>
</tr>
</tbody>
</table>

Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<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>JIMP</td>
<td>Javascript Image Manipulation Program</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 frequently had difficulty in choosing restaurants because information about restaurants was hard to access. Due to advancement in mobile technology, information relating to restaurants is 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 may not be sufficient since the quality of images is not unified and images with low quality greatly reduce the readability of users. Indeed, existing applications provide little support for users with different background (e.g. users with different mother tongues). 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 understood by 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. If only a
few menu photos for a restaurant are uploaded, the restaurant cannot be found by searching for food name because only the pictures are added to the restaurant.

The aforementioned problems can be solved by Optical Character Recognition(OCR) technology. With OCR, printed or written text in images can be converted into editable text [6], allowing the text to be further processed, by translating the scanned text into multiple languages or searching for the scanned text from the images.

This interim report introduces a new smartphone application that integrates OCR 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 for restaurants. Apart from generating online food catalogues, this application includes image comparison technology, which enables users to search for food name by uploading an image of the food item.

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 this application are introduced. Then, the current progress and planning, as well as the potential risks 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 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 using this application, 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. In the past, only food items with reviews from customers were available for searching. This application helps restaurants to promote all food items, instead of the most renowned ones that other users reviewed.

For users, other than searching for ranking of food and restaurants, searching for food name in any languages is also possible in this application. Translation is applied to the searching keyword that the user input, hence identical food name in different languages can be searched simultaneously. Instead of waiting for reviews on the specific food items, 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 and the images stored in the database to retrieve the information of the food item.

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 in order 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. For simplicity, only English and Chinese are supported for OCR Analysis at the current stage.

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 three types: Uploading (for OCR Analysis), Searching (with Translation), and Login/Register. Each type could have one or more features. The features highlighted in red are the unimplemented 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.</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 searchability 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.</td>
</tr>
</tbody>
</table>
Searching

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search for food by name</td>
<td>Keyword can be entered to search for the corresponding food</td>
</tr>
<tr>
<td>Search for food by tags</td>
<td>Tags starting with “#” can be given by user for each food. Each food can have many tags. A tag can be entered to search for the corresponding food with the corresponding tag.</td>
</tr>
<tr>
<td>Search for food by image</td>
<td>Users can take a photo and find out what the food is in a specific restaurant.</td>
</tr>
<tr>
<td>Search for food menu of restaurant</td>
<td>Users can search for a specific restaurant and the restaurant’s food menu will be shown.</td>
</tr>
<tr>
<td>Search for location of restaurant</td>
<td>Navigation is provided if users search for specific restaurant.</td>
</tr>
</tbody>
</table>

Login/Register

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>
</tr>
</tbody>
</table>

Table 1. Features for this project.

3.2 Software Limitations

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

<table>
<thead>
<tr>
<th>Limitation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<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>Picture Size for OCR Analysis</td>
<td>Maximum of 4 MB is supported by Google Vision API [7]</td>
</tr>
<tr>
<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>Text Languages for OCR Analysis</td>
<td>Only 56 Languages are supported [8]</td>
</tr>
<tr>
<td>Picture(s) Storage for each user</td>
<td>Maximum storage size of 5GB is supported by AWS Simple Storage Service (S3)[9]</td>
</tr>
</tbody>
</table>

Table 2. Limitations for this project.
4. Methodology

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 modification 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 [11], 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.

Figure 1. The iterative cycle [10].
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 [12]. 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 [13]. In a typical three-tier architecture, the tiers are the presentation tier, the logic tier and the data tier [14]. 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 [15]. Scalability is defined as the ability to support increasing amount of work [16]. 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 [17]-[18], 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 [19].

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 [20]. 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 [21].

A MySQL relational database has been used because it is a open-source software, meaning that it can be used or modified freely [22]. Some other popular relational databases are Oracle RDBMS or Microsoft SQL Server [23]. However license fees are required for these databases. Due to budget limitation, a MySQL relational database has been used.
4.1.4 Table design

![Database design figure using Crow Foot Notation](image)

**Figure 2.** Database design figure using Crow Foot Notation [24].

Multiple tables and relationships exist in a relational database.

First, the restaurant name and the location of the restaurant are stored in the “Restaurants” table (see Figure 2). The field restaurant_id is autogenerated 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 user_id, phone_number and bucket_name of users are stored in the “Users” table as shown in the figure. The field user_id is an autogenerated number that is unique for each user,
and it is used to uniquely identify a particular user. In addition to the user_id, the phone_number is the phone number used by the user for login, and the bucket_name is the image storage directory path for the user to store the photos uploaded by the user.

Finally, the restaurant_id, tags, user_id, name, price, ratings and photo_filename given to the food are stored in the “Food” table in the figure. The field “tags” is similar to the hashtag function in Instagram [25], 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.

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 [26]. 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 [27]. AWS has been used because it provides the best documentations and most functions among the three platforms [28] - [29].
4.2 Function Design

Functions of this application are divided into four main aspects: Adding & Uploading Food Records, Uploading Menus for OCR Analysis, Searching for Food By Text, 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

![Figure 3. System flow of uploading food record(s).](image)

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); while the corresponding path is stored in the database for later retrieval.

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 Uniform Resource Locator (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 [30]. 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 lacking 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 [31], 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

![Diagram of system flow](image)

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

User can search for specific food or restaurant in the application. The keyword is sent from the client application to the server (see Figure 5, Step 1). Then, the keyword is translated into different languages using Google Translate API (Step 2), and the translated keywords 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 Login/Register

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

**Figure 6. 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 6, 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 [32]. By using hashing, even if
the server is accessed by some malicious users, they cannot regenerate the user’s verification code and login to those accounts.
5. Current Progress

There are three major functions fully implemented by now: Adding & Uploading Food Records, Uploading Menus for OCR Analysis, Searching for Food By Text, which are the same as mentioned in section 4.2 Function Design. The Login/Register function is partially implemented, except the hashing algorithm. In the remaining subsections of this part, detailed progresses of each function 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 7. 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 incomplete, and they are highlighted in yellow and the corresponding missing field is presented in red (Figure 7a). 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”) (Figure 7b). 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

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

This view is shown whenever the user click on a record (Figure 8a, shown when click on first row in Figure 7) 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 from camera. If
no picture is selected, a temporary image will be displayed to allow image adding by users (Figure 8b).

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 9a, shown when click on location row in Figure 8). The table automatically updates with latest search result upon typing (Figure 9b). 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 of the location, the location name and location address, these information are presented in the location row.
5.2 Uploading Menus for OCR Analysis

![User interfaces of OCR.](image)

When user clicks on the “Scan” button in the view of Figure 10a, 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 table format as shown in Figure 10b, which contains one column of food name and one column of the corresponding food price. The “+” button on each of the rows is for adding a new empty field, which is for helping users to match the food name and food price. The “+” on the middle is for adding an extra food name, and the “+” on the right is for adding an extra food price. This button is added because sometimes the OCR is not accurate, hence some food name or price may be missing from the OCR result. This button allow user to add extra record while keeping the original order of other records.
The first row of the table is used for selecting a location for the restaurant. It works the same way as mentioned in Section 5.1.3.

5.3 Searching for Food by Text

5.3.1 Rank View

![User interfaces of Rank View.](image)

The view in Figure 11a is shown when the user clicks on the middle icon of the tab bar in the bottom of the screen. It is the Rank View, which shows the ranking of all food, sorted according to their average ratings in descending order. When the user clicks on one of the food item, a food detail view is shown(Figure 11b). The food detailed 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 the database design, it is possible that the same food record can be uploaded by multiple users. Hence the 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 no photo uploaded for this food record, the image with Red Cross is shown (Figure 11a); else if there is/are photo(s) uploaded, user can swipe on the image to see other images for the food record. Full screen viewer with zooming is also supported by tapping on the image.

5.3.2 Search with Translation

![Figure 12. User interfaces of Searching with Translation.](image)

Translation is automatically applied to the search keyword inputted by the user. The server automatically detects the language of the keyword and translates it to English and Chinese as they are the official languages in Hong Kong. In spite of 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 [33]. The server then queries for all the records that includes the original keyword or the translated keywords within the food name. Finally only the matched
records are presented in the table. As shown in Figure 12a, when the user inputs “tofu” in the search bar of the rank view, record that includes the translated version of “tofu” (translates to “豆 腐” in Chinese) is shown. In Figure 12b, the user inputs “蛋”, and the food record with food name “EGG FRIED RICE” is shown because the English version of “蛋” is “egg”. As shown in Figure 12c, the Korean word “치킨”, which is “chicken”, is inputted by the user, and the two records with food name that includes “CHICKEN” are shown.

5.3.3 Search with Tags

![Figure 13. User interfaces of Searching with Tags.](image)

Food records can be searched by inputting the keyword with the symbol “#” at the beginning. The server finds all the records that includes the keyword as the tag. The matched records are therefore shown on the client application. As shown in Figure 13a, the user inputs “#xxx” in the search bar of the rank view, then the record that includes the tag “xxx” is shown. The food detailed view of the matched record is shown in Figure 13b, proving that there is a tag “#xxx” for the food record.
5.4 Login/Register

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

After the user has inputted their phone numbers into the application (see Figure 14a), 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 14b). 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 [34]. 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 14a). Logout button is also implemented to destroy all the “Keychain” items created (see Figure 14c). Then the user need to verify again upon pressing the logout button.
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. Testings 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. Future Planning

This section covers the future planning of unimplemented functions. The scheduled deadlines are allocated according to their necessity and ease of completion, with reference to the PICK Chart (see Figure 15). The PICK chart is a Lean Six Sigma tool which helps organizing grouping ideas for process amelioration [35]. The priorities are described as follows: tasks with higher impact and easier implementing process are planned to do first, then to the easier tasks with lower payoff, and lastly to the harder tasks with higher impact.

Below shows the scheduled tasks and their corresponding deadlines (see Table 3). Elaborations of each task and justifications for their corresponding deadlines will be illustrated below from section 7.1 to 7.4.

![PICK Chart](image)

*Figure 15. PICK Chart [35].*
<table>
<thead>
<tr>
<th>Task</th>
<th>Deadline</th>
<th>Progress</th>
</tr>
</thead>
<tbody>
<tr>
<td>First iteration</td>
<td>30 September 2017</td>
<td>Completed</td>
</tr>
<tr>
<td>● Interface to list all stored food records</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Interface to add or edit food records</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Set up server and database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Connection between server and database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Connection between the client application and the server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Second iteration</td>
<td>15 November 2017</td>
<td>Completed</td>
</tr>
<tr>
<td>● Interface for location search</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Interface for food ranking</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Upload picture for OCR analysis</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Third iteration</td>
<td>31 December 2017</td>
<td>Completed</td>
</tr>
<tr>
<td>● Search for food by text with translation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>● Login function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fourth iteration</td>
<td>31 January 2018</td>
<td>In Progress</td>
</tr>
<tr>
<td>● Search for menus of restaurants</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fifth iteration</td>
<td>20 February 2018</td>
<td>Not started</td>
</tr>
<tr>
<td>● Search for food by images</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sixth iteration</td>
<td>15 March 2018</td>
<td>Not started</td>
</tr>
<tr>
<td>● Navigation to Restaurants</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Seventh iteration</td>
<td>31 March 2018</td>
<td>Not started</td>
</tr>
<tr>
<td>● Boost Business Potential</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall Testing and Debugging</td>
<td>15 April 2018</td>
<td>Not started</td>
</tr>
<tr>
<td>Deliverable of Final Report</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Presentation</td>
<td>16-20 April 2018</td>
<td>Not started</td>
</tr>
</tbody>
</table>

*Table 3. Future Working Plan.*

### 7.1 Searching for Menus for Restaurants

Searching for food in restaurants is the fundamental function provided by most of the existing online food catalogue application, therefore it is necessary to complete this function before implementing other add-on functions.
With simple SQL queries, this function can be easily completed. Thus it is put in the next iteration due to its greatest necessity and high ease of completion.

### 7.2 Searching for Food By Image

This function is used to find out the food name of an unknown dish. This happens when users discovering stunning dish or dish photo without knowing any information about it.

The flow is described as follows: after the user has taken the photo or selected a photo of the unknown dish, the server performs image comparisons on related images using Javascript Image Manipulation Program (JIMP) library and returns the corresponding food name to the client application; the returned food name is then presented in AR 3D Text form using ARKit.

This function provides a more convenient way in searching which brings great benefits to users. With experiences in handling AR objects and studies on existing frameworks in previous projects, the ease of implementation has been increased. Hence this function is going to be implemented in the fifth iteration.

### 7.3 Navigation to Restaurants

This function is planned to provide means of transportation, estimated time and suggested routes to the destined restaurant. It is perceived to be more challenging than functions described in section 7.1 and 7.2 as related studies on map navigation have not been carried out. Methodology of this function has not yet been designed. However navigation is definitely effective in guiding users to restaurants. This function is therefore put in the iteration after the implementation of searching for food by image, with its great benefits to users but slightly challenging implementation process.
7.4 Boost Business Potential

To boost business potential, reward points are given to users for uploading food records. The reward points can be used to get discounts in some restaurants in order to attract more customers. Nevertheless, communications with restaurants for discounts are time-consuming. The business potential is thus put in the last iteration.
8. Conclusion

The major functions of this application have been implemented. The application can now support uploading food records from menu photos, as well as searching for food in various languages.

Uploading food items by simple scanning and multi-language support has 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. The only limitation is from the external services, such as Google’s API. Enhancing the algorithm and user experience will be the focus in a later scope, 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


