


MustEatNow

Smart Management of Household Perishables Inventory

APPENDICES

Members:

Goh Kum Whye Leslie	A0229982M
Lim Jingcheng, Konchok	A0229969A
Low Ruo Qi	A0229977E
Low Yim Tong	A0229983L

Project Report for the Intelligent Reasoning Systems course at Institute of Systems Science, National University of Singapore

Table of Contents

Appendix A - Project Proposal

GRADUATE CERTIFICATE: Intelligent Reasoning Systems (IRS)

PRACTICE MODULE: Project Proposal

Date of proposal: 16 January 2021
Project Title: MustEatNow - Smart Management of Household Perishables Inventory
Sponsor/Client: *(Name, Address, Telephone No. and Contact Name)* Institute of Systems Science (ISS) at 25 Heng Mui Keng Terrace, Singapore NATIONAL UNIVERSITY OF SINGAPORE (NUS) Contact: Mr. GU ZHAN / Lecturer & Consultant Telephone No.: 65-6516 8021 Email: zhan.gu@nus.edu.sg
Background/Aims/Objectives: The proposed intelligent eco-system will make use of various advanced machine reasoning techniques and components to foster generic intelligent system adoption and agile implementation for business. It aims to support consumers in smarter management of household inventory through a three prong approach: inventory logging, timely

consumption and mindful purchasing.

Requirements Overview:

- Research ability
- Programming ability
- System integration ability

Resource Requirements (please list Hardware, Software and any other resources):

Hardware proposed for consideration:

- CPU with Windows / Linux OS

Software proposed for consideration:

- Software development: MySQL, HTML, CSS, JavaScript, PHP, Python
- Pre-trained machine learning models: Tesseract OCR engine
- Cloud server (for deployment): DigitalOcean Cloud Platform

Number of Learner Interns required: (Please specify their tasks if possible)

Team of four members

Methods and Standards:

Procedures	Objective	Key Activities
Requirement Gathering and Analysis	The team should meet with ISS to scope the details of the project and ensure the achievement of business objectives.	1. Gather & Analyze Requirements 2. Define internal and External Design 3. Prioritize & Consolidate Requirements 4. Establish Functional Baseline
Technical Construction	- To develop the source code in accordance to design. - To perform unit testing to ensure the quality before the components are integrated as a whole project	1. Setup Development Environment 2. Understand the System Context, Design 3. Perform Coding

		4. Conduct Unit Testing
Integration Testing and acceptance testing	To ensure interface compatibility and confirm that the integrated system hardware and system software meets requirements and is ready for acceptance testing.	1. Prepare System Test Specifications 2. Prepare for Test Execution 3. Conduct System Integration Testing 4. Evaluate Testing 5. Establish Product Baseline
Acceptance Testing	To obtain ISS user acceptance that the system meets the requirements.	1. Plan for Acceptance Testing 2. Prepare for Acceptance Test Execution 3. ISS Evaluate Testing
Delivery	To deploy the system into production (ISS standalone server)	1. Software must be packed by following ISS's standard

	environment.	2. Deployment guideline must be provided in ISS production (ISS standalone server) format 3. Production (ISS standalone server) support and troubleshooting process must be defined.

Team Formation & Registration

Team Name: L4
Project Title (repeated): MustEatNow - Smart Management of Household Perishables Inventory
System Name (if decided): MustEatNow
Team Member 1 Name: Goh Kum Whye Leslie
Team Member 1 Matriculation Number: A0229982M
Team Member 1 Contact (Mobile/Email): 90684493 / E0687390@u.nus.edu

Team Member 2 Name: Lim Jingcheng, Konchok
Team Member 2 Matriculation Number: A0229969A
Team Member 2 Contact (Mobile/Email): 96330372 / E0687377@u.nus.edu
Team Member 3 Name: Low Ruo Qi
Team Member 3 Matriculation Number: A0229977E
Team Member 3 Contact (Mobile/Email): 94576213 / E0687385@u.nus.edu

Team Member 4 Name: Low Yim Tong
Team Member 4 Matriculation Number: A0229983L
Team Member 4 Contact (Mobile/Email): 90107528 / E0687391@u.nus.edu

Appendix B - Mapped System Functionalities

S/N	System / Subsystem	Functionality	Course module / Concept
1	Smart OCR - Image Reader	Image preprocessing; Text extraction from image	Cognitive system - vision systems
2	Smart OCR - Text Parser	Text preprocessing; Rule-based model inferencing; Text information extraction	Machine Reasoning - logical reasoning; reasoning under uncertainty
3	Database System	Knowledge base management of word corpus	Knowledge acquisition / discovery - manual extraction
4	Database System	Data management	Knowledge representation - Relational DB
5	Database System	Purchase recommendations	Machine reasoning - rule-based logical reasoning
6	Frontend System	Expiry reminders	Machine Reasoning - rule-based logical reasoning
7	Overall System	Data transfer and manipulation; System integration	Knowledge representation - frames representation system

Appendix C - Installation and User Guide

Overall System Architecture

The MUSTEATNOW System was implemented as a web-based application residing on a cloud server architecture. The following technologies were used to deploy the application.

- A server running the Linux Ubuntu (v 20.04) operating system and serviced by the Apache HTTP Server system was deployed to host the core application.
- A MySQL (community edition) database was deployed on a separate dedicated server and serves as the knowledge base and data repository for the system.
- The web application was created using HTML, CSS and JavaScript for its frontend interface and the PHP language for its backend logic control.
- A Python (v.3) environment was also deployed on the web server for the Smart OCR component to interface with the PHP backend.
- A web domain (musteatnow.com) was purchased and setup to point to the IP Address of the web server which allow users to use a normal browser to access the application.

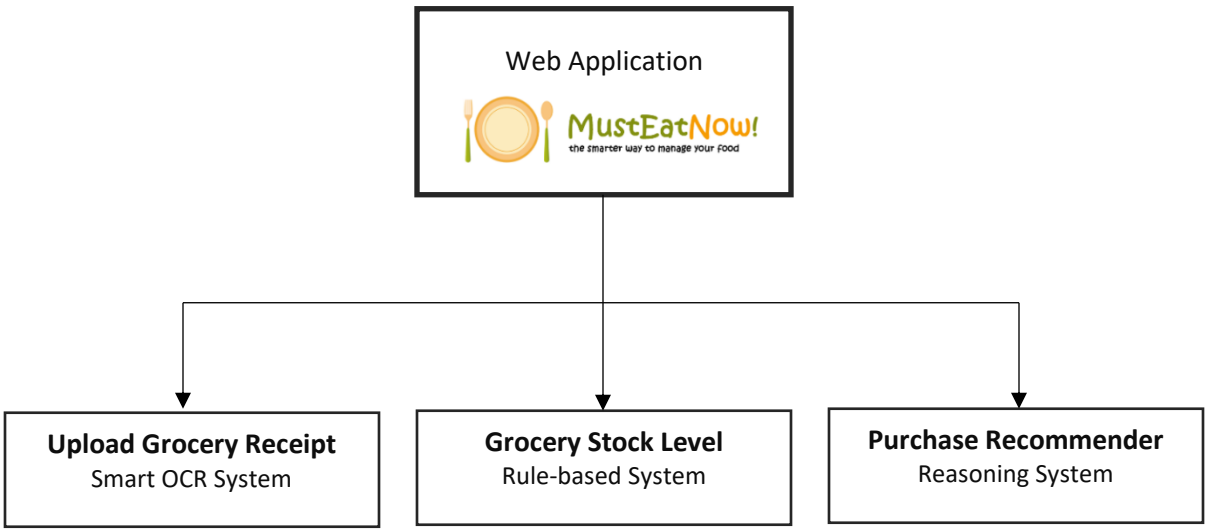
The entire server architecture was deployed on the DigitalOcean Cloud Platform.

Application Design

The MUSTEATNOW application was designed to solve a consumer's problem to food wastage. The entire system can be broadly broken down into three interconnected subsystems:

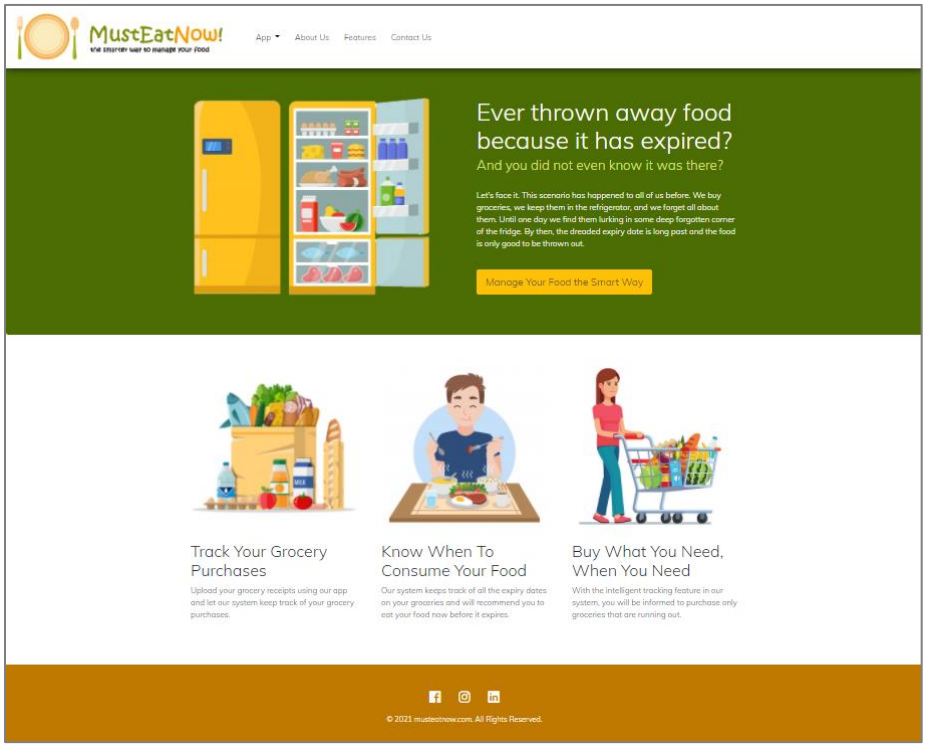
1. A smart OCR system to parse grocery receipt images as input data and to extract key information on items, prices and quantities purchased. Data extracted is matched against an evolving corpus to identify different food types.
2. A simple rule-based system to match the food type against a knowledge base of standard expiry information to determine when a user should consume a food or whether the food has crossed its expiry date.
3. A reasoning system that learns a consumer's consumption patterns and recommends the food to restock when the user visits a supermarket.

The entire application interfaces with a database system which serves as its knowledge base as well as its data repository.



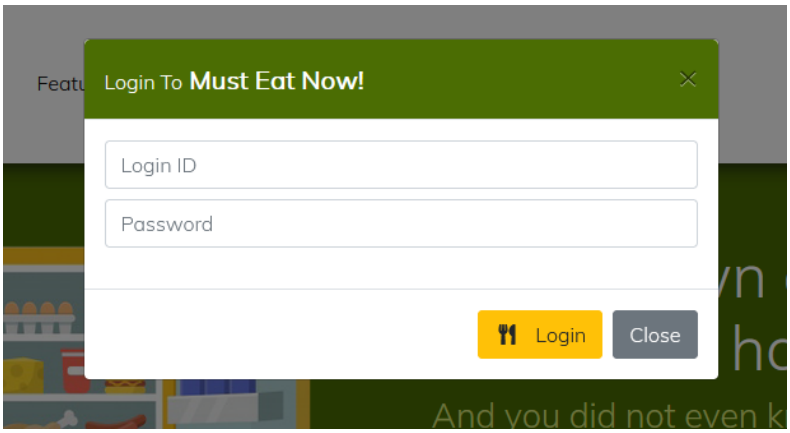
User Guide

As the MUSTEATNOW application is web-based, users can access it by typing www.musteatnow.com in their browser. On the first landing page, the user can read a bit more on the features of the app and how to sign up for an account.



Login

To begin using the application, users will need to have an account and must login before they can access any of the functions.



Upon login, the user will have access to the other functional web pages in the application.

Log in: my_family@example.com

Password: irs2021

Upload Grocery Receipt

To begin, the user must first upload the latest receipt from the supermarket.

FAIRPRICE XTRA
NEX HYPERMART
23 Serangoon Central #03-42
Singapore 556083
TEL: 6634 4947/4948

UEN No: S83CS0191L
GST No :M4-0004578-0

A PACKHAM PEAR 800G	6.75
COWHEAD CHEESE 3X250	7.95
Normal Price 9.85	
D T/C YOG SBERY150G	
2X$2.75/Unit	5.50
2 FOR $3.95	-1.55
F GREEK STRAWBERRY90	
3X$1.20/Unit	3.60
3 FOR $2.85	-0.75
FP HNY B/HAM200G	2.95
FRESH BLUEBERRIES	
3X$3.50/Unit	10.50
3 FOR $6.95	-3.55
FRESH FRUITS PROMO H	
2X$3.95/Unit	7.90
KF BABY SPINACH 200G	1.10
KORU NZ/US APPLE800G	5.95
Normal Price 7.90	
M R.TOPPING-N.KOMI40	4.90
M R.TOPPING-S.FUMI40	4.90
NAT.SHABU SHABU250G	7.00
P CHSE SMKY CHDR125G	5.55
S G/C TB-REG/SOFT 6S	11.85
SH PKT BREAD/BUN 350	
2X$3.50/Unit	7.00
USA R'BERRY(A/FLOWN)	
2X$6.95/Unit	13.90
LINK CARD 8105000001824397	
TOTAL	**$101.45**
MASTER	$101.45
ACNT NO.xxxxxxxxxxxx0987	
Exchange ID 8105000001824397	
Exchange ID for rebates	
REBATE AMT:	101.45

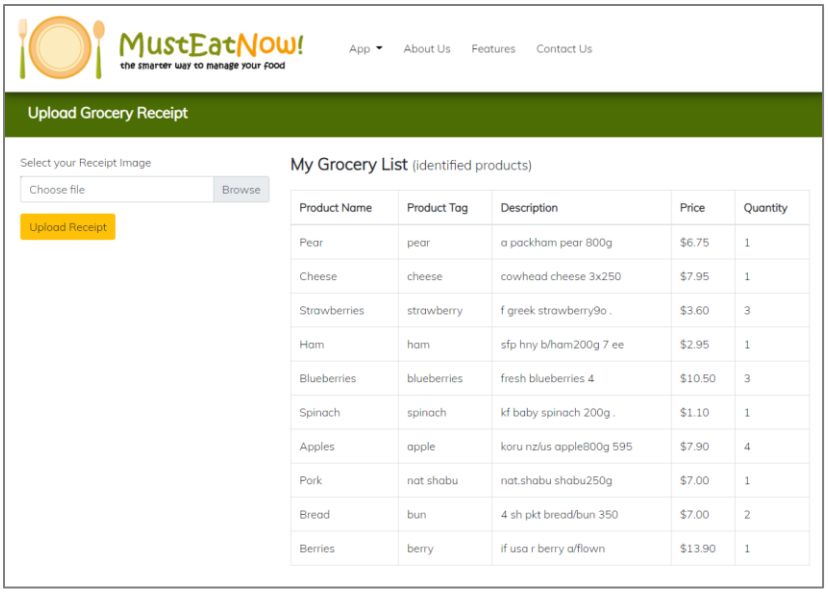
CS BUGIS JUNCTION
230 VICTORIA STREET #B1-16 S188024
GST MR85002257 Tax Invoice BIZ 22661900A

CARROT AU	1.52 V
0.724 X 2.10 /KG	
CARROT AU	1.39 V
0.662 X 2.10 /KG	
CARROT AU	1.54 V
0.733 X 2.10 /KG	
SAV MINI CHEESE 360G	12.95
1.000 X 12.95	
HORMEL SMKD TURKEY BRST 8OZ	14.95
1.000 X 14.95	
H/D COFFEE 473ML	14.50
MCVITIES DIGESTIVE 500G	3.50
SUBTOTAL	50.35
TOTAL(SGD)INCL.GST	50.35
GST AMT	3.29
Redeem TapForMore	0.49
BALANCE DUE (SGD)	49.86
VISA	49.86

PA CARD NO: 1009710029402442
Redeem TapForMore Pt 73.50
Earn TapForMore Pt 50.35 Bal Point: 51.00
TERMINAL: 50581175 APPR CODE:003046
BATCH: 00265 CARD NO:0295
REF.NO: 118963 053178
NO SIGNATURE REQUIRED

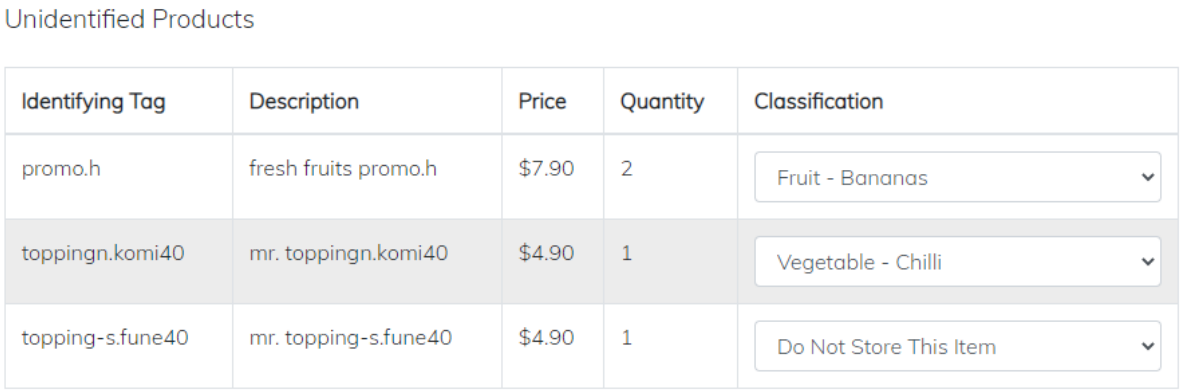
V - Save up to 20% on HealthyFood(TM)
Visit AIAVitality.com.sg now!

Receipt required for refund or exchange
Rg:14 Tr:61135 Ch:14 21:08 12/04/21



The grocery list will be identified by the Smart OCR System and a table of recognized food types will be displayed on the page.

There will also be a list of items that the OCR System cannot identify (either through ambiguity or nonstandard phrases).



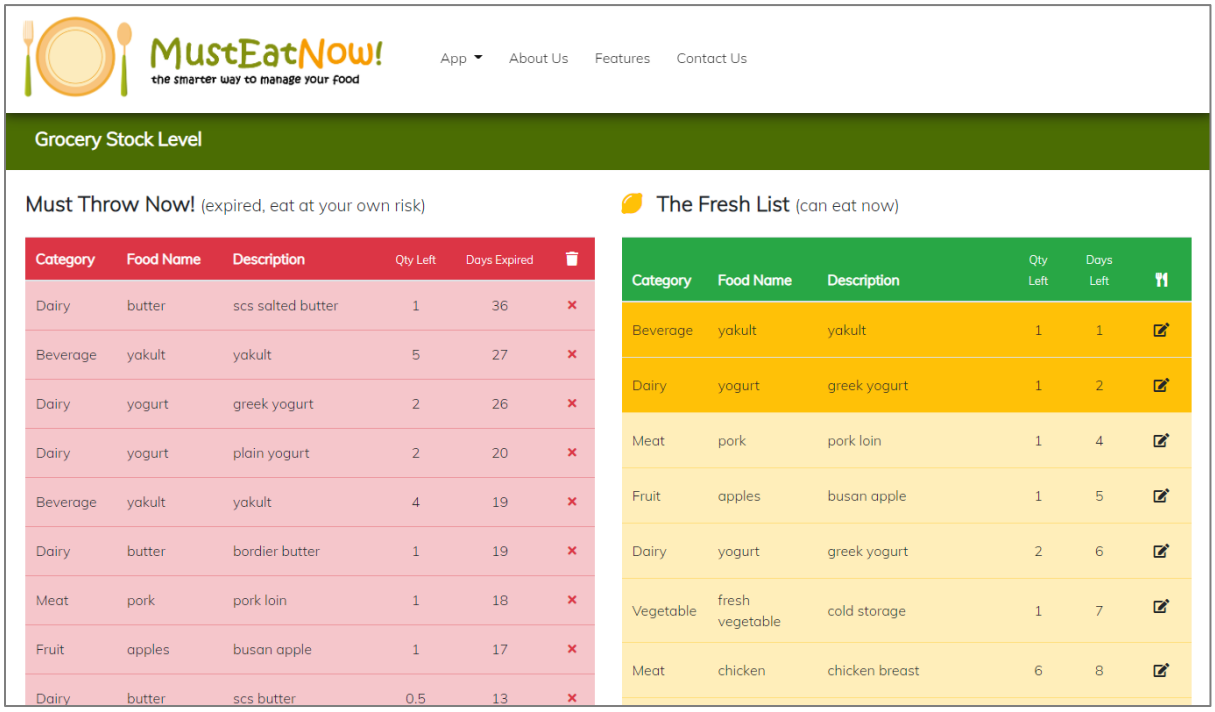
The user has a choice to select the correct classification of the item and whether to add the item to his grocery list. Upon selection and classification, the unknown item is tagged and inserted to the corpus of text identifiers. The screenshot below shows the different words that relate to a single food keyword.

13	12	vegetables	vegetables	2021-04-28 16:39:33
14	12	vegetable	vegetable	2021-04-28 16:39:33
15	12	vege	vege	2021-04-28 16:39:33
16	12	veggie	veggie	2021-04-28 16:39:33
17	12	veggies	veggies	2021-04-28 16:39:33
18	13	garlic	garlic	2021-04-28 16:39:33
19	13	garlics	garlics	2021-04-28 16:39:33
20	14	onion	onion	2021-04-28 16:39:33
21	14	onions	onions	2021-04-28 16:39:33
22	15	parsley	parsley	2021-04-28 16:39:33

So, each time an unidentified word is displayed and classified, the original corpus of keywords and tags is increased. And if you upload a later receipt which contains the newly added keyword, the OCR system is now able to tag and identifier the said keyword.

Grocery Stock Level

The Grocery Stock Level page shows the existing food stock amount for the current user. The entire list is broken into two sections: expired and fresh food items.



Category	Food Name	Description	Qty Left	Days Expired
Dairy	butter	scs salted butter	1	36
Beverage	yakult	yakult	5	27
Dairy	yogurt	greek yogurt	2	26
Dairy	yogurt	plain yogurt	2	20
Beverage	yakult	yakult	4	19
Dairy	butter	bordier butter	1	19
Meat	pork	pork loin	1	18
Fruit	apples	busan apple	1	17
Dairy	butter	scs butter	0.5	13

Category	Food Name	Description	Qty Left	Days Left
Beverage	yakult	yakult	1	1
Dairy	yogurt	greek yogurt	1	2
Meat	pork	pork loin	1	4
Fruit	apples	busan apple	1	5
Dairy	yogurt	greek yogurt	2	6
Vegetable	fresh vegetable	cold storage	1	7
Meat	chicken	chicken breast	6	8

The expiry data is based on a standard set of rules which are stored in the database. The screenshot below shows some of these rules,

food_id	food_cat_id	food_id_name	default_expiry_days	default_serving	insert_dt
10	2	Crab	8	1	2021-04-28 09:14:16
11	2	Processed Se...	30	1	2021-04-28 09:14:16
12	3	Fresh Vegeta...	8	1	2021-04-28 09:14:16
13	3	garlic	10	1	2021-04-28 09:14:16
14	3	onion	10	1	2021-04-28 09:14:16
15	3	parsley	10	1	2021-04-28 09:14:16
16	3	cilantro	10	1	2021-04-28 09:14:16
17	3	potato	10	1	2021-04-28 09:14:16
18	3	spinach	10	1	2021-04-28 09:14:16
19	3	chive	10	1	2021-04-28 09:14:16
20	3	ginger	10	1	2021-04-28 09:14:16
21	3	chilli	10	1	2021-04-28 09:14:16
22	4	Apples	10	1	2021-04-28 09:14:16
23	4	Bananas	10	10	2021-04-28 09:14:16
24	4	pineapple	10	1	2021-04-28 09:14:16
25	4	Pear	10	1	2021-04-28 09:14:16
26	4	mango	10	1	2021-04-28 09:14:16
27	4	strawberries	10	1	2021-04-28 09:14:16

Food items that are expired will be totally disposed of once the user clicks on the delete button. All expired foods are highlighted in red.

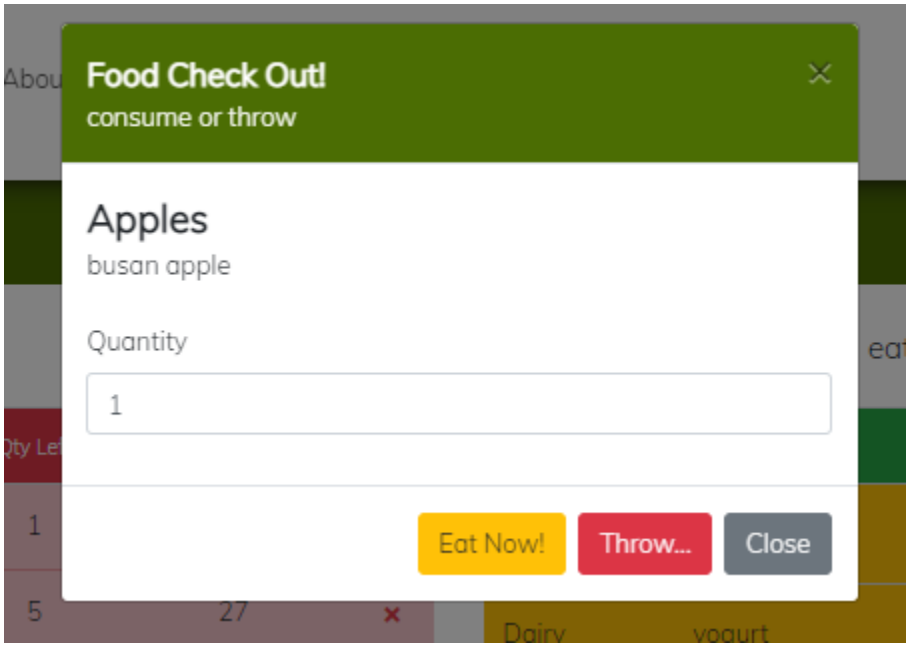
The Fresh List contains food that have not exceeded its expiry date. In the Fresh List, there is another set of rules that determine the color of the row in order to provide a clearer visual representation to the user. The rules are:

- expiry date within 0 – 3 days dark yellow
- expiry date within 4 – 8 days light yellow
- expiry date 9 days and beyond white

The Fresh List (can eat now)

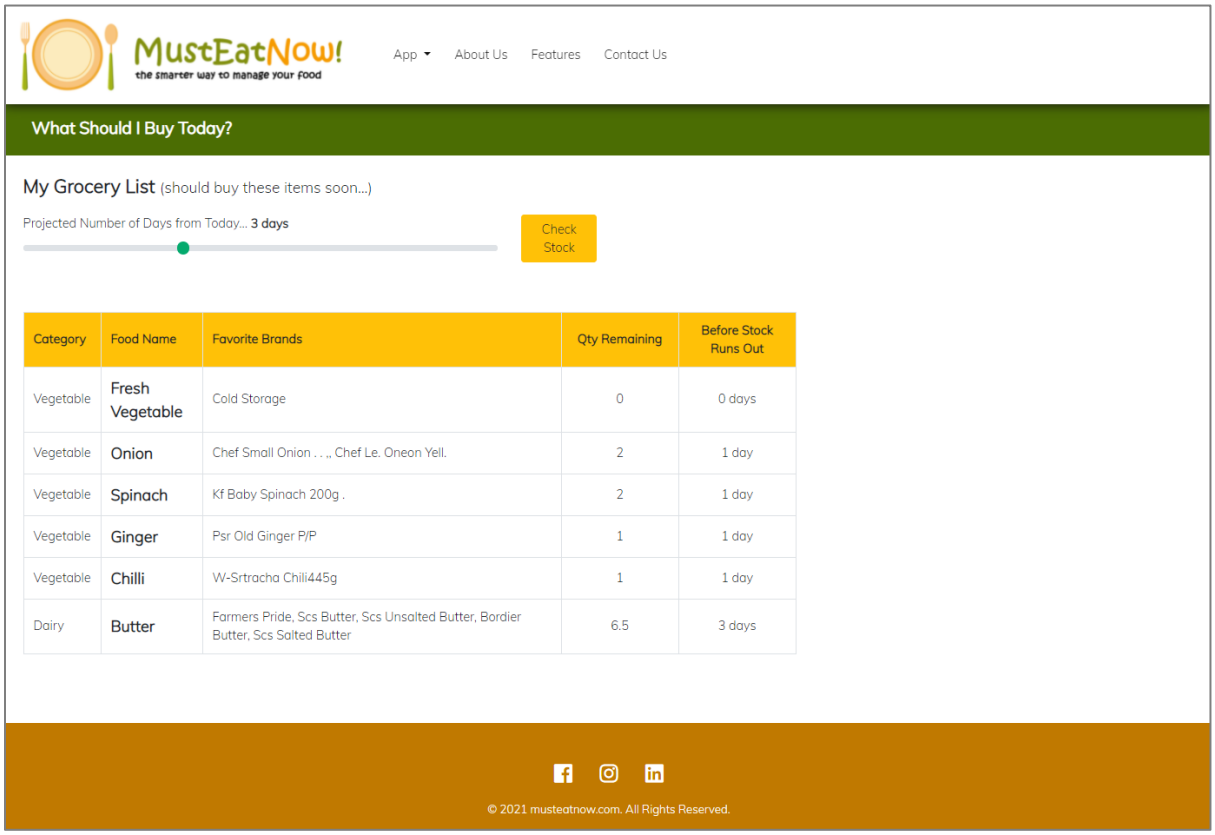
Category	Food Name	Description	Qty Left	Days Left	
Beverage	yakult	yakult	1	1	
Dairy	yogurt	greek yogurt	1	2	
Meat	pork	pork loin	1	4	
Fruit	apples	busan apple	1	5	
Dairy	yogurt	greek yogurt	2	6	
Vegetable	fresh vegetable	cold storage	1	7	
Meat	chicken	chicken breast	6	8	
Grain	bread	4 sh pkt bread/bun 350	4	8	
Grain	bread	4 sh pkt bread/bun 350	4	8	
Meat	pork	nat.shabu shabu250g	1	9	
Meat	ham	sfp hny b/ham200g 7 ee	1	9	
Meat	ham	sfp hny b/ham200g 7 ee	1	9	
Dairy	butter	bordier butter	1	9	

To manage the Fresh List, the user can decide whether to consume or dispose of the food and how much quantity to check out at one time.

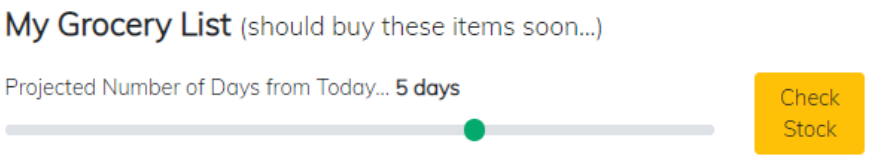


Grocery Purchase Recommendations

The Grocery Purchase List page recommends the food items to replenish - for food that are either going to expire or going to be fully consumed.



The user is able to view the recommended purchase list from 1 to 7 days in the future. Default is set to 3 days in advanced.

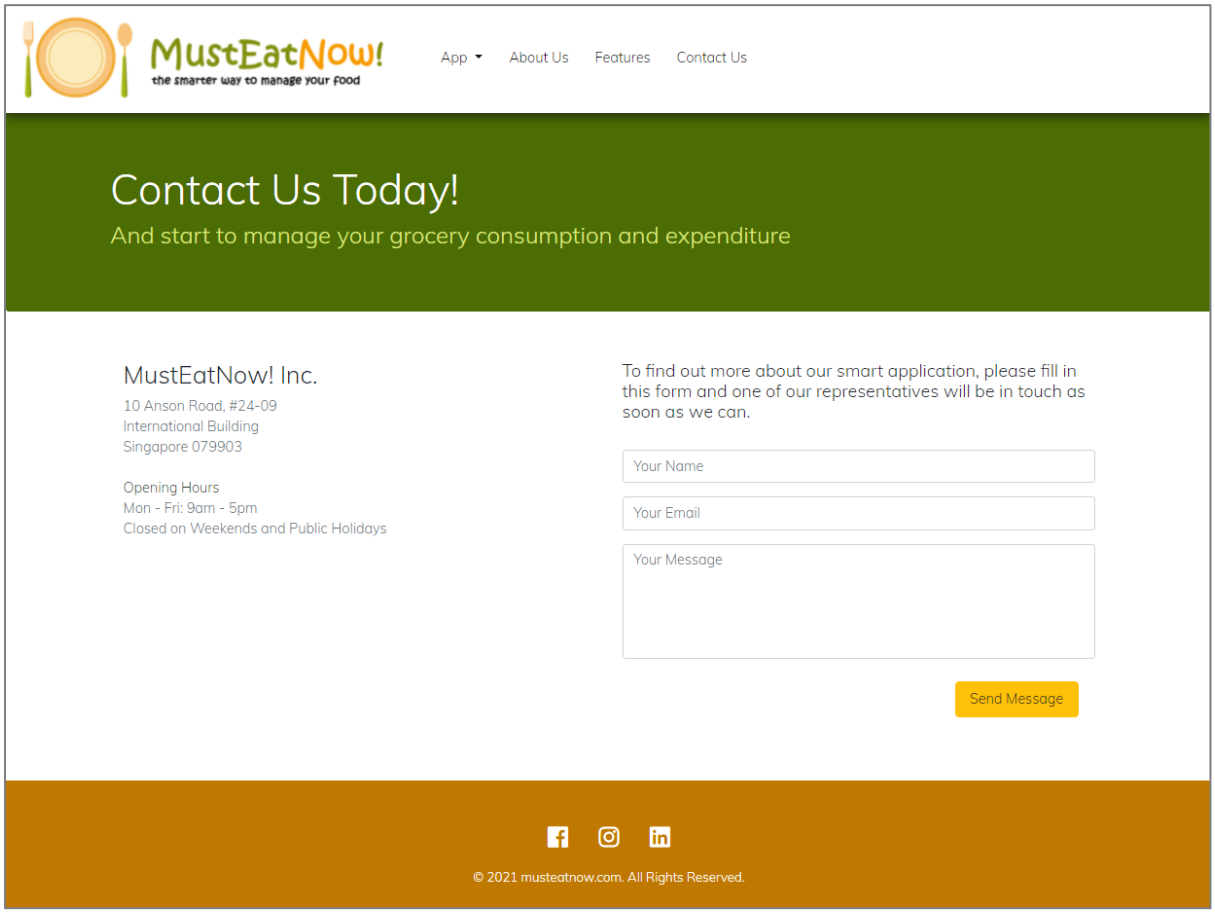


Purchase recommendations are based on a user's consumption pattern, so each individual user will have a different shopping list.

Category	Food Name	Favorite Brands	Qty Remaining	Before Stock Runs Out
Vegetable	Fresh Vegetable	Cold Storage	0	0 days
Vegetable	Onion	Chef Small Onion . . „ Chef Le. Oneon Yell.	2	1 day
Vegetable	Spinach	Kf Baby Spinach 200g .	2	1 day
Vegetable	Ginger	Psr Old Ginger P/P	1	1 day
Vegetable	Chilli	W-Srtracha Chili445g	1	1 day
Dairy	Butter	Farmers Pride, Scs Butter, Scs Unsalted Butter, Bordier Butter, Scs Salted Butter	6.5	3 days
Dairy	Milk	Green Fields, Farmers Union	2	4 days
Dairy	Yogurt	Meiji Yogurt, Yogurt, Plain Yogurt, Greek Yogurt	7	4 days
Dairy	Cheese	Cowhead Cheese 3x250	2	4 days
Fruit	Pear	A Packham Pear 800g	2	4 days
Fruit	Berries	If Usa R Berry A/Flown	2	4 days
Fruit	Tomato	Fresh Tomato... .	1	4 days
Grain	Bread	4 Sh Pkt Bread/Bun 350	8	4 days

Further Information

For further information on the MustEatNow! application, please do not hesitate to contact us.



Installation Guide

This Installation Guide will detail the necessary steps to install and setup the MUSTEATNOW! application on your local machine (Windows).

Local Installation (Windows)

Setting up the WAMP environment. As this is a web-based application, we need to setup a local web server to run the entire application.

1. Download the **Apache Web Server** from this location:
   https://www.apachehaus.com/downloads/httpd-2.4.46-o111j-x86-vc15.zip.sha

   Before installing Apache, you might need to install the Visual C++ 2008 Redistributable Package. You can download the package from this location: https://aka.ms/vs/15/release/vc_redist.x64.exe

   Once the Visual C++ package has been installed, it is often a good idea to restart the system to ensure any remaining changes requiring a restart are completed.

   Extract the Apache ZIP folder to a destination (e.g. C:/Program Files/Apache 2.4) on your computer. Navigate to the BIN folder and run the httpd.exe command.

   You will likely notice a dialogue box from the Windows Firewall noting that some features are being blocked. If this appears, place a checkmark in "PrivateNetworks…" as well as "PublicNetworks…", and then click "Allowaccess."

2. Installing **Apache as a Service**.
   In your Command Prompt window, enter (or paste) the following command:

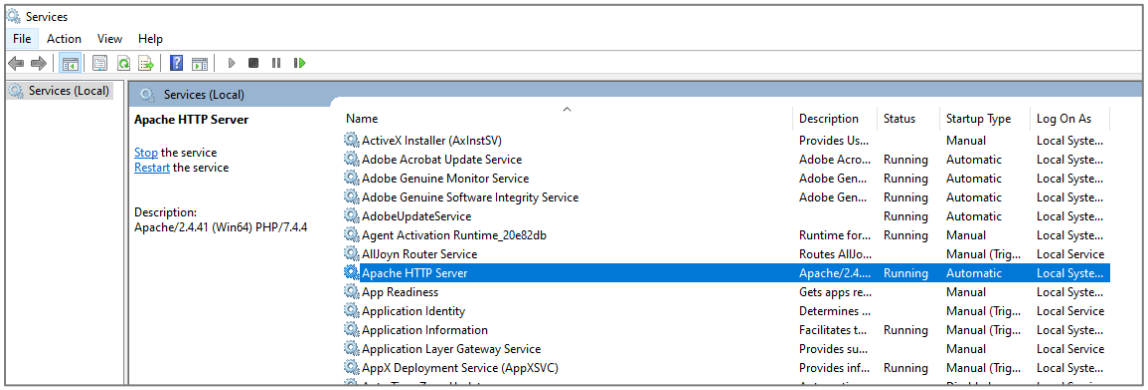
   httpd.exe -k install -n "Apache HTTP Server"

   You will get the following output:

```
Installing the 'Apache HTTP Server' service
 The 'Apache HTTP Server' service is successfully installed.
 Testing httpd.conf....
 Errors reported here must be corrected before the service can be started.
```

   Open Windows Services and check that the Apache HTTP Server has been installed as a service and that its startup type is set to "Automatic".

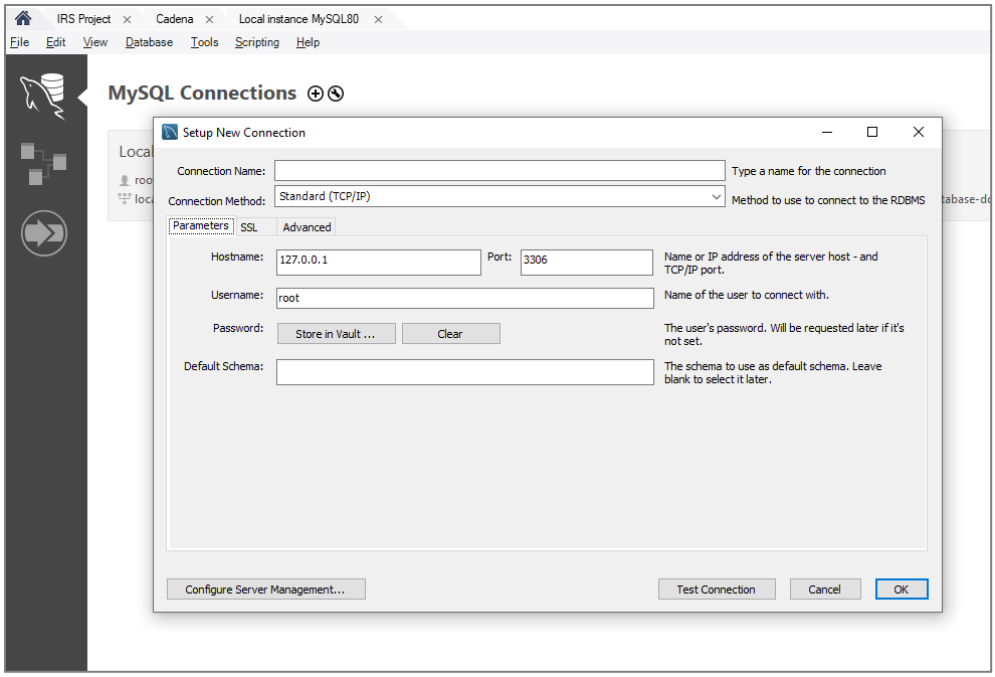
   Restart the Apache service and open a web browser. To test your setup, type http://localhost in the browser's URL bar.



3. Installing and configuring **MySQL**
   We will be using the free open-sourced database known as MySQL. Download the MySQL Installer for Windows here: https://dev.mysql.com/downloads/windows/installer/8.0.html

   - Run the installer
   - Select the “Developer Default” option
   - Set a root account and password
   - Select to start the MySQL server as a Windows service
   - Install MySQL Workbench as part of the current installation process

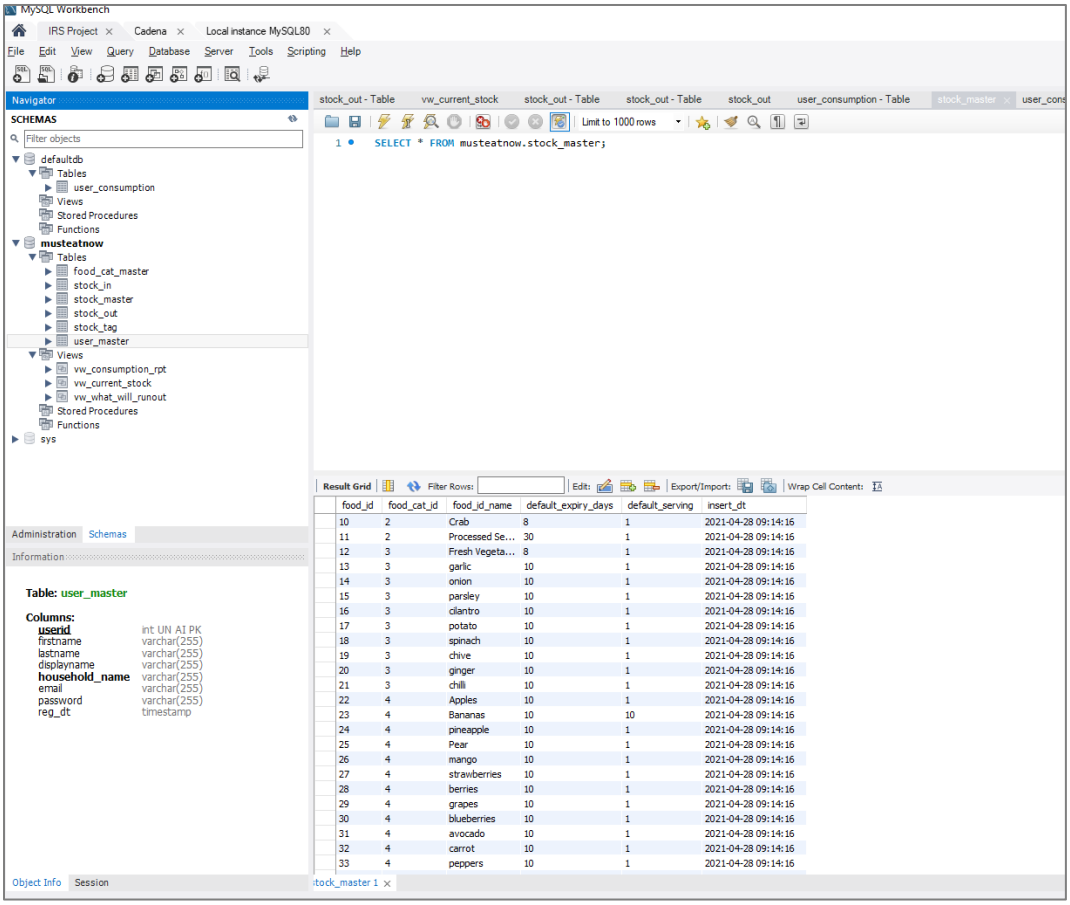
4. Connecting to **MySQL using Workbench**
   Upon completing the installation of the MySQL database, open MySQL Workbench. Workbench provides a GUI to connect to and manage the databases.



Setup a new connection to the database using the root account and password that you have set during the installation process. Test the connection by pressing the TEST CONNECTION button.

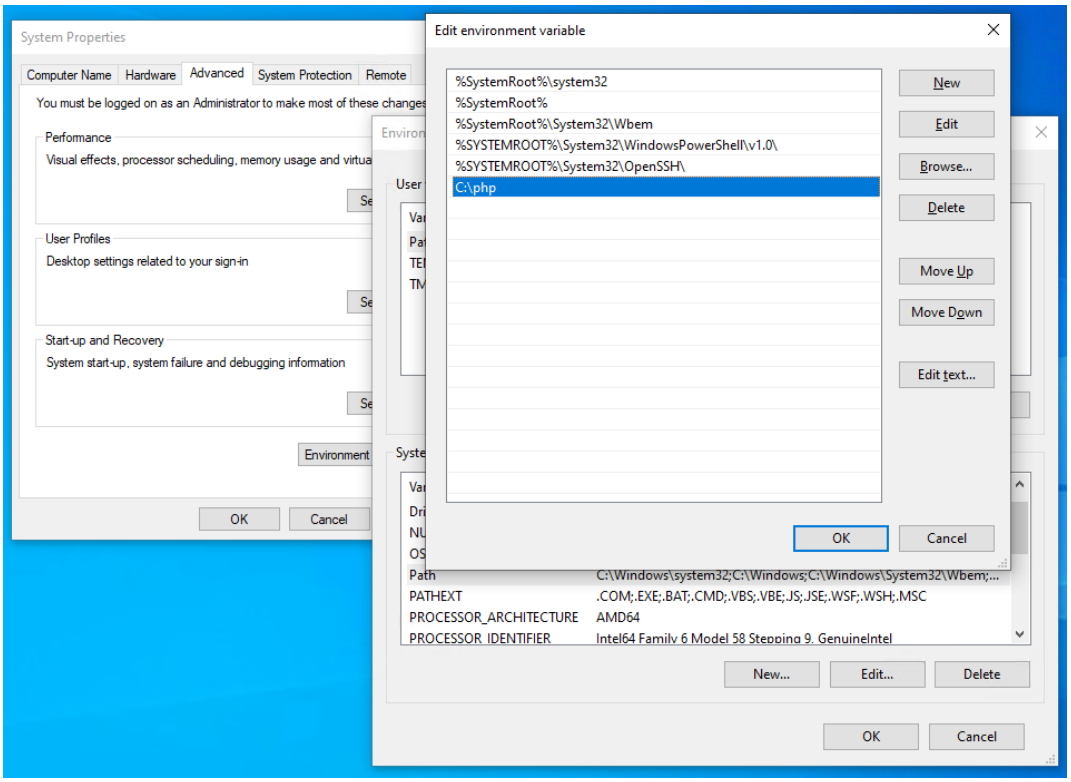
If the connection is successful, name the connection and save it.

On the home screen, click on the newly saved connection and you will be directed to the MySQL dashboard which shows all the available schemas in the database.



5. Installing **PHP**
   Download the latest thread safe PHP Windows version here: https://windows.php.net/download#php-7.4

   Extract the contents of the ZIP folder to a suitable location on your local machine (e.g., C:/Program Files/php). Once this is done, we will need to configure Windows environment variables to be able to access php from the command line.



Copy the file php.ini-development available at the path where we have extracted PHP and save it as php.ini at the same location.

In the php.ini file, you have to enable some extensions by removing the semi-colon symbol from the extension packages.

```
....
;extension=openssl
;extension=pdo_firebird
extension=pdo_mysql
;extension=pdo_oci
;extension=pdo_odbc
....
```

You must also enable to extensions directory by updating the php.ini file as shown below. Please name the extension directory according to your own folder pathing.

```
; Directory in which the loadable extensions (modules) reside.
; http://php.net/extension-dir
;extension_dir = "./"
; On windows:
;extension_dir = "ext"
extension_dir = "E:\tools\php\php-7.3.1\ext"
```

6. Configuring **Apache**. Apache needs to be configured to
   - recognize the installed php module, and
   - point a pseudo domain to a home directory.

   Open the httpd.conf file from your Apache folder. It is usually found in the conf folder of your Apache root folder. Search for the line having LoadModule php7_module. Now comment out the line and add the path to your PHP as shown below. Please remember to use your own folder paths.

```
#LoadModule php7_module "${INSTALL_DIR}/bin/php/php7.1.26/php7apache2_4.dll"
LoadModule php7_module "e:/tools/php/php-7.3.1/php7apache2_4.dll"
PHPIniDir "e:/tools/php/php-7.3.1"
```

   Add the following lines to the bottom of your httpd.conf file:

   <VirtualHost *:80>

   ServerAlias testmusteatnow.com

   ServerAlias www.testmusteatnow.com

   DocumentRoot C:/musteatnow

   </VirtualHost>

   <IfModule dir_module>

   DirectoryIndex index.php index.html

   </IfModule>

7. Configuring the **Host file**
   Find your host file at this location: C:\Windows\System32\Drivers\etc\hosts
   Right click the file and user Notepad to edit it.

   Add in these lines to the bottom of the Host file:

   127.0.0.1 testmusteatnow.com
   127.0.0.1 www.testmusteatnow.com

   Save and exit the file.

8. Download the Application from its **Git Repo**
   The MustEatNow! Application can be found here: [rqlow/IRS: IRS project (github.com)]

   Create a new folder on your local machine (C:/musteatnow) and unzip the contents of the Git to this new location. Please note that if you setup the application on a different location on your local machine, you need to put in the correct path in the DocumentRoot in step 6 above.

9. Creating the **Database Schema**
   Open MySQL Workbench using the connection that was setup in step 4. Under the schema folder in your application, there is an sql file called musteatnow.sql. Use Workbench to open and execute this SQL file.

   The entire *musteatnow* schema, complete with data tables, sample data and knowledge base will be created for you in MySQL.

10. **Configuring the MustEatNow! Application**
    Open the configuration file from this location: C:/musteatnow/common/config.php. Setup the configurations accordingly. Please remember to use your own pathing.

```php
//----- database functions
function SetDatabaseConnection() {
    $host = '127.0.0.1';
    $dtbs = 'musteatnow';
    $user = 'root';
    $pass = 'password';
    $char = 'utf8';

    $dsn = "mysql:host=$host;dbname=$dtbs;charset=$char;port=3306";

    $options = array(
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
        PDO::ATTR_EMULATE_PREPARES => false
    );

    $conn = new PDO($dsn, $user, $pass, $options);
    return $conn;
}

//----- file path functions
function PATH_Server()
{
    return "";
}

function PATH_Root() {
    return "C:/musteatnow/";
}

function PATH_Host() {
    return "http://www.testmusteatnow.com/";
}


//----- file size
function UploadFileSize()
{
    return 5120000;
}

?>
```

11. Setting up a **Python Environment**

    Python for Windows can be downloaded directly from the Microsoft Store package, or as a full MSI installer from this location: https://www.python.org/downloads/release/python-394/

    Once Python is installed, as with PHP, you will need to setup the Python command as a Windows Environment Variable. Follow step 5 above.

    Remember to include Python's package manager, PIP, as an environment variable too.

12. Installing the **Python Libraries**
    Use PIP to install the following libraries for Python.

    - mysql-connector-python
    - pyspellchecker
    - pytesseract
    - opencv-python
    - scikit-image
    - nltk
    - matplotlib

    Install the tesseract library using the windows executable file at this location:
    Home · UB-Mannheim/tesseract Wiki (github.com)

Server Installation (Ubuntu)

Setting up the LAMP environment. As this is a web-based application, we need to setup a web server to run the entire application.

1. Installing **Apache** and Updating the Firewall
   Install Apache using Ubuntu's package manager, apt:

```
$ sudo apt update
$ sudo apt install apache2
```

   Once the installation is finished, you'll need to adjust your firewall settings to allow HTTP traffic. To only allow traffic on port 80, use the Apache profile:

```
$ sudo ufw allow in "Apache"
```

   You can verify the change with:

```
$ sudo ufw status
```

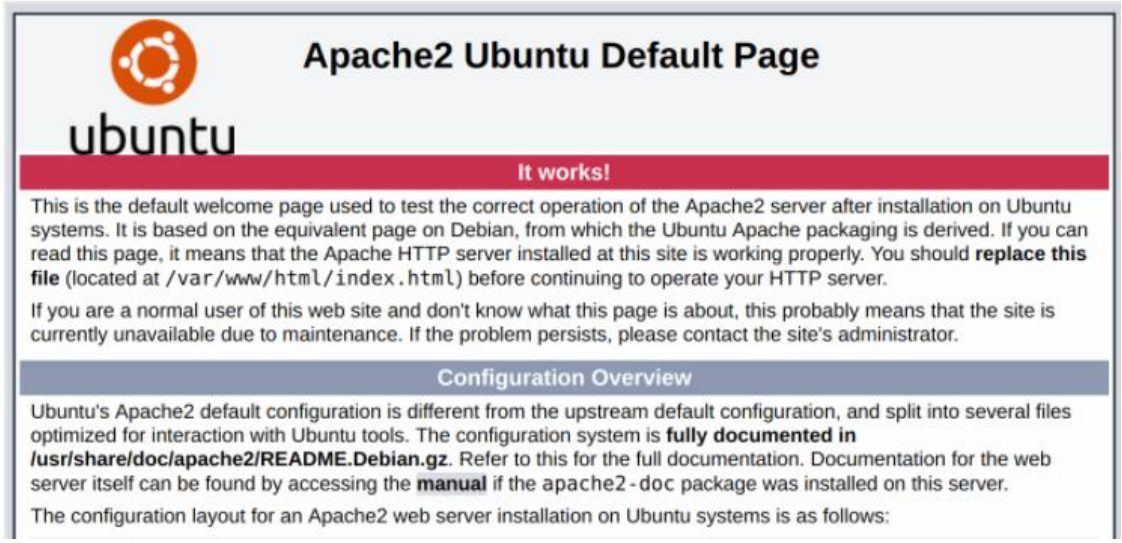
```
Output
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
Apache                     ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
Apache (v6)                ALLOW       Anywhere (v6)
```

Traffic on port 80 is now allowed through the firewall. You can do a spot check right away to verify that everything went as planned by visiting your server's public IP address in your web browser (see the note under the next heading to find out what your public IP address is if you do not have this information already):

```
http://your_server_ip
```

You'll see the default Ubuntu 20.04 Apache web page, which is there for informational and testing purposes. It should look something like this:



2. Installing **MySQL**
   Use apt to acquire and install this software:

```
$ sudo apt install mysql-server
```

When the installation is finished, it's recommended that you run a security script that comes pre-installed with MySQL. This script will remove some insecure default settings and lock down access to your database system. Start the interactive script by running:

```
$ sudo mysql_secure_installation
```

Follow the steps to create database users and the access password for the root user.

When you're finished, test if you're able to log in to the MySQL console by typing:

```
$ sudo mysql
```

This will connect to the MySQL server as the administrative database user root, which is inferred by the use of sudo when running this command. You should see output like this:

```
Output
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 22
Server version: 8.0.19-0ubuntu5 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Type "exit" to leave the MySQL command line.

3. Installing **PHP**
   To install PHP, run the following command:

```
$ sudo apt install php libapache2-mod-php php-mysql
```

4. Setting up **Git**
   To install Git on the server, run the following command:

```
$ sudo apt install git
```

5. Creating a **Virtual Host** for the MustEatNow! Application
   Create the directory for the MustEatNow! application as follows:

   $ sudo mkdir /home/musteatnow

   Clone the MustEatNow! Git Repo to this folder that you have created. And change the ownership of the storage folder

   $ git clone https://github.com/rqlow/IRS.git /home/musteatnow
   $ chown -R www-data:www-data /home/musteatnow/storage

6. Configuring a **Virtual Domain in Apache**
   Use nano or vim to modify the Apache configuration file on the server. If you do not have a domain name, then you need to use the IP address that is associated with your server (or 127.0.0.1 if you are running a local Ubuntu installation).

   sudo vim /etc/apache2/sites-enabled/musteatnow.conf

   This will create a blank config file. Paste the Virtual Host configuration below to the config file, taking care of the IP address or domain name that is going to be associated with the application.

   <VirtualHost *:80>
   DocumentRoot "/home/musteatnow"
   ServerName 127.0.0.1
   ServerAlias 127.*
   ErrorLog ${APACHE_LOG_DIR}/error.log
   CustomLog ${APACHE_LOG_DIR}/access.log combined
   </VirtualHost>

Save and exit the file.

You can now use a2ensite to enable the new virtual host:

```
$ sudo a2ensite your_domain
```

Once this is done, restart the Apache server:

$ service apache2 restart

7. Setting up the **Python Environment**
   Python for Ubuntu can be setup by typing in the following command:

   $ sudo apt install python3.8
   $ sudo apt install -y python3-pip

   Install the necessary libraries for our application:

   $ sudo apt-get install tesseract-ocr

   Use PIP (or pip3) to install the following Python libraries:

   - mysql-connector-python
   - pyspellchecker
   - pytesseract
   - opencv-python
   - scikit-image
   - nltk
   - matplotlib

8. Creating the **Database Schema**
   At the command line, execute the SQL file found in the schema directory:

   $ mysql -u user -p < /home/musteatnow/schema/musteatbow.sql

   The entire *musteatnow* schema, complete with data tables, sample data and knowledge base will be created for you in MySQL.

9. **Configuring the MustEatNow! Application**

Open the configuration file from this location using nano or vim:
/home/musteatnow/common/config.php.

Setup the configurations accordingly. Please remember to use your own pathing.

```php
//----- database functions
function SetDatabaseConnection() {
    $host = '127.0.0.1';
    $dtbs = 'musteatnow';
    $user = 'root';
    $pass = 'password';
    $char = 'utf8';

    $dsn = "mysql:host=$host;dbname=$dtbs;charset=$char;port=3306";

    $options = array(
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
        PDO::ATTR_EMULATE_PREPARES => false
    );

    $conn = new PDO($dsn, $user, $pass, $options);
    return $conn;
}
```

```php
function PATH_Root() {
    return "/home/musteatnow/";
}

function PATH_Host() {
    return "http://127.0.0.1/";
}
```

Appendix D1 - Individual Project Report (Goh Kum Whye Leslie)

Personal Contributions

For this project, my overall contributions were in:

- setting up the web (Apache, PHP) and database (MySQL) architecture on a cloud server (Ubuntu)
- designing the frontend application interface using HTML, CSS, JavsScript and jQuery
- integrating with the Python based Smart OCR System and its knowledge base
- creating the Grocery Stock Level page that is governed by a simple rule-based reasoning system
- interfacing with the logical view created in the database to determine the items to purchase

Subsystem 1: Integration with the Smart OCR System

The integration between the web application and the Python based OCR component was done through a shell level script execution. My task at this point was to:

- receive the user uploaded receipt image from the application interface
- save the image onto the web server
- execute the Python code containing the OCR logic to read the image file
- receive a JSON data array from the OCR component
- parse the data array into a format readable by the web environment
- display the results into two HTML tables (*identified* and *unknown* food items)

For unknown items, I created an interface that allowed the user to select from a predefined class

of known food types from a dropdown selection. Upon confirmation, the application will then:

1. upload the entire grocery list to the database
2. insert the previously unknown keyword tags into the corpus of keywords for future identification

Subsystem 2: Grocery Stock Level management using a Rule-based Reasoning System

The rules are stored as a knowledge base within the database. My task for this subsystem was to:

- match existing food items against the expiry date rules found in the knowledge base
- create an HTML table displaying all expired food products and allowing the user to dispose of them from the system
- create a second HTML table to display a "Fresh" list of food products and their projected expiry dates
- users are then able to checkout each "Fresh" food item by keying in a quantity and selecting whether the food item is consumed or disposed

I created a second rule set for visualising the "Fresh" list. The rules are:

- expiry date within 0 – 3 days dark yellow
- expiry date within 4 – 8 days light yellow
- expiry date 9 days and beyond white

Subsystem 3: Grocery Purchase Recommendations from a Logical View created in the Database

The recommendations for purchasing grocery food items were done by integrating a regression based model stored as a logical view in the database. My task was to:

- query the logical view and retrieve the data set it returns
- display the result on an HTML table in the frontend
- allow the user to change an input parameter (projected days to purchase) and re-query the data view to get a new set of results, which were then displayed to the user again

Useful Takeaways

This was an interesting project that involved integrating several different software platforms, a few of which I was initially unfamiliar with.

My working experience as a web developer has never exposed me to Python, much less integrating a web application with a Python software component. It took a while to be able to understand how to successfully call, pass and retrieve parameters from PHP and Python. There were some things (like writing to a file) that were not allowed when the Python script was called from PHP and it took me a while to troubleshoot the entire OCR component before I could successfully interface with it.

It was also an experience to work with a teammate who is very proficient with manipulating data in MySQL using data views. The database was a central model that all subcomponents of the system revolve around and I was able to see how each of my teammates query the database using different software platforms and models.

Applications in Other Situations

As mentioned earlier, all of my working experience has been around designing and developing web applications and one of my reasons for attending this course is to further my knowledge in A.I. and intelligent systems and to see how such learning and reasoning models can be incorporated in web applications.

From this project, I can see that it is possible to interface the web programming environment with

a reasoning system environment written in Python. It will take some more trial and error to be able to fully understand how to interface the two software layers, but this has been a great start.

OCR is one component that I believe has great potential in a web application. For example, receipts could be scanned and identified in a petty cash management system which might link to a bigger HR or finance module in a Management Information System.

All in all, I am glad to be given the opportunity to be able to work with my teammates who are all from very different backgrounds and to learn from them their expertise and experience.

Appendix D2 - Individual Project Report (Lim Jingcheng, Konchok)

Personal contributions

In this project, my main contributions were project ideation, exploring use-cases from my professional work, and offering inputs from technical feasibility and implementation perspective. For actual implementation of the project, my role was to design the schema, communicating with both Python and PHP developer on the system logic from a database perspective. I also proposed the use of logical views over stored procedures to reduce overheads and response latency between application and database. The initial table population was also created by me, and with inputs from the Python developer, populated the database with the first corpus to start our project.

After the initial setup of the database and schema, I made the logical views for the current stock availability and regression based solution to forecast the consumption of available stocks in the user's inventory. This was done together with the frontend developer to ensure the insights are communicated to the end user in way that drives a desired response from the user. In this case, a desired response would be to consume food which are expiring first, not eat and dispose expired food, and lastly help users decide on the grocery purchases items in the inventory which will be running out soon.

Once the development of the project complete, I contributed to the database, and purchase recommender portions of the report. The technical video was also recorded by me, in collaboration with the frontend developer for preparation of the presentation materials.

Useful takeaways

I have always been a Data Analyst by training, and later picked up some use of Python libraries for model-fitting, and learned from my colleagues who majored in statistics, the basics of forecasting. Starting on this program I had wanted to use more of Python programming, however the requirements of the project meant my practical contribution to the team would be at the database level. It was still a learning opportunity as I had previously been much more familiar with Microsoft products, include SQL Server, however we implemented this in a LAMP stack, meaning I had to learn MySQL Server and rediscover how all the features used in my professional work can be replicated here. It was also an opportunity for me to work with a full-stack developer and

observe how all the components were integrated together. While previously I have used Python professionally, it was also an opportunity for me to observe how Python programmers from other industries use this popular tool.

Applications in other situations

Where previously I viewed the database primarily for data analysis and means to enforce consistent logic throughout the system, this project also taught me that a database could grow as a knowledge base over time with more user input. Seeing first hand an implementation of a computer vision feature, also impressed upon me the myriad of use-cases in industrial and office applications. It also showed me that a combination of rule-based decision making and ML-model driven decision making can result in targeted outputs for the user, and there is a human component in designing the output in a way that drives the user to the desired action. I can foresee applying this in logistical applications, or situations where there is physical stock. As a matter of fact, we are exchanging ideas about how we can employ this for commercial customers in the market.

Appendix D3 - Individual Project Report (Low Ruo Qi)

Personal contributions

In this project, my main contributions were in the system design, development and implementation of the Smart OCR system. For the other deliverables, I was responsible for creating the business use case video demo and contributed to the report write-up for the Smart OCR system.

During the design stage, I researched some methods for receipt OCR that have been implemented by others. Most of the literature had a focus on image preprocessing as this is a core step in ensuring a good output from the OCR engine. Through this literature review, I also identified some shortfalls in these methods. For example, most applications implement a standalone rule set, or require extensive labelled training data to build a neural network model. Hence, the team tried to overcome these shortfalls through other system features and implementation.

During the development stage, I experimented with different image preprocessing techniques to try and identify the image preprocessing pipeline that would be best suited for our use case. I also analyzed the various receipts to identify the different patterns and experimented with different rules to build and fine tune the model. I developed and implemented the model inferencing process and the observation state estimation process to try and overcome some of the challenges due to the inaccuracies in the input data. The rule set was built based on manual extraction and repeatedly fine tuning and testing the model to try and find optimal threshold values for the rules. During the implementation stage, I worked together with other team members to ensure the proper integration of the full system.

Useful takeaways

One of my most important takeaways was the importance of properly scoping the task required to be performed by the model. By being clear on the task that the model is designed for, I can then curate the proper training data and do the necessary preprocessing to ensure that the data is suitable for the problem that the model is trying to solve. I learnt about the importance of ensuring proper training data because the model can only output based on what it has seen in the

training dataset.

Besides that, I also learnt about the advantages of building a system that crosses different domains and integrating them together effectively. Usually, a single standalone model or system will have its own advantages and disadvantages. By making use of different subsystems, these subsystems can compensate for the shortfalls of each other and integrate together to form an effective solution. For example, in our project, we make use of the rules during text parsing to compensate for some OCR inaccuracies, and we also make use of the database and frontend system for continuous training to compensate for the model inaccuracies. As such, when thinking about system design, I learnt to try and think across different domains when solutioning for a problem.

Lastly, learning to make the business use case video was an interesting experience and a new software skill that I learnt. It was a fun experience learning how to frame our business problem and making a video demo on the use case.

Applications in other situations

In my work area, I work on robotics perception problems such as identification and classification problems. This project experience in developing rule sets for well-defined tasks was valuable and applicable to me. Through this experience, I have learnt how to create better rules and how to represent the data effectively with the different knowledge representation methods. I learnt how to choose a suitable method for modelling, and how simple rules can be very effective. Not all problems need to employ neural networks to be solved. For example, in parts of my work, there are areas where I have to manually verify some data output or manually combine and generate new training data during development. I could apply the knowledge and skills from the course to create more effective automated data pipelines that can help me do automated data verification and data generation that integrates with the existing system.

Appendix D4 - Individual Project Report (Low Yim Tong)

Personal contributions

My main contributions were in the ideation and preliminary design stages of our project. I was responsible for crafting the commercial case, market research, product positioning and report write-up in those aspects, and supporting the research and design logic of the text parser subsystem (part of the Smart OCR) and the purchase recommender system.

Useful takeaways

During the design phase, I experimented with text processing techniques applied in receipt digitisation and found the more relevant functions to be (i) short form to full form conversions, (ii) regular expressions, (iii) difflib library to get close matches to extract the specific items required given a fixed starting database of items (Figure D4.1). I laid out a single item rule-based logic model for expiry and purchase recommendations (Figure D4.2); and a takeaway was that the ability to translate that to databases was key to multiply the model logic through larger datasets.

On the business case, it was important to distill the functions existing applications in the market had when conceptualising the value proposition of our proposed product. On the technical side, I learnt a lot from the other team members who carried out the development and deployment of the system and gained exposure to databases, web-based applications, image processing and system integration. I have gained a better understanding of the appropriate tools available and their limitations for implementation of a system concept/ model.

Applications in other situations

In my work area, I analyse and value businesses across a range of industries. I find that the inventory management system created could potentially be developed and used in back-of-house for small-medium enterprises (e.g., single outlet where perishable items are stored on site). The OCR capability can be further developed to read purchase orders and delivery orders and help to replace manual processes of data input.

```
FAIRPRICE XTRA
NEX HYPERMART
23 Serangoon Central #03-42
Singapore 596083
TEL: 6634 4947/4948

UEN No: S83CS0191L
GST No :M4-0004578-0

A PACKHAM PEAR 800G 6.75

COMHEAD CHEESE 3x250 7.95
Normal Price 9.85

D T/C YOG SBERY150G

QN$2,75/Uni t 5,50
2 FOR $3.95 -1.55

F GREEK STRAWBERRY9O

3X$1.20/Unit 3.60
3 FOR $2.85 -0.75

FP HNY B/HAM200G 2.95

FRESH BLUEBERRIES

3X$3.50/Uni t 10.50
3 FOR $6.95 -3,55

FRESH FRUITS PROMO H

2X$3 .95/Uni t 7.90

KF BABY SPINACH 200G 4.10

KORU NZ/US APPLE800G 5.95
Normal Price 7.90

MR. TOPPING-N.KOMI40 4.90

M R.TOPPING-S .FUMI40 4.90

NAT. SHABU SHABU250G 700

P CHSE SMKY CHDR125G 5*55

$ G/C TB-REG/SOFT 65 11135

SH PKT BREAD/BUN 350 ,
```

Receipt result in text file post OCR

```
<re.Match object; span=(689, 697), match='20/03/21'>
2021-03-20
```

```
Counter({'yoghurt': 2, 'strawberry': 2, 'apple': 2,
```

	0
yoghurt	2
strawberry	2
blueberry	1
apple	2
cheese	2
bread	1
spinach	1
pear	1

Date, Item and quanity extracted into pandas

Figure D4.1: Information extracted from a receipt based on a defined list of perishable items

Template model for one item over 5 week period

userid 1, stock_id 1

Note: to expand to two userIDs, and multiple items

Assumptions

default_expiry_days		3	
servings consumed per occasio	*between:*	0	4

	servings bought per occassion	Probability	Cumulative probability
Frequency of grocery trips (0 – did not go grocery shopping; >=1 represents servings bought in one grocery trip	0	70%	0%
	1	3%	70%
	2	3%	73%
	3	3%	76%
	4	3%	79%
	5	3%	82%
	6	3%	85%
	7	3%	88%
	8	3%	91%
	9	3%	94%
	10	3%	97%
		100%	

Summary of model

servings bought	58
servings disposed	(12)
servings consumed	(46)
servings left in fridge	0

	No. of grocery trips	Servings consumed
Week 1	2	11
Week 2	2	10
Week 3	1	12
Week 4	3	9
Week 5	0	4

BoP – 'beginning of period'

EoP – 'end of period'

	Inventory movement						Rules			OUTPUT	
date	serving_BoP	serving_out_obsolete	serving_in	serving_out_consumed	serving_out_disposed	serving_left EoP	servings bought within last 3 days	servings bought 3 days ago	servings consumed in last 3 days	**musteatnow** (expires next day)	**mustbuynow** (not enough for next day demand)
17/4/2021	0	0	0	0	0	0	0	0	0	0	1
18/4/2021	0	0	0	0	0	0	0	0	0	0	2
19/4/2021	0	0	0	0	0	0	0	0	0	0	0
20/4/2021	0	0	7	(4)	0	3	7	0	(4)	0	1
21/4/2021	3	0	0	(3)	0	0	7	0	(7)	0	1
22/4/2021	0	0	0	0	0	0	7	0	(7)	0	0
23/4/2021	0	0	4	(4)	0	0	11	7	(11)	0	1
24/4/2021	0	0	0	0	0	0	4	0	(7)	0	3
25/4/2021	0	0	0	0	0	0	4	0	(4)	0	0
26/4/2021	0	0	8	(3)	0	5	12	4	(7)	0	0
27/4/2021	5	0	0	(1)	0	4	8	0	(4)	0	0
28/4/2021	4	0	0	(1)	0	3	8	0	(5)	0	0
29/4/2021	3	0	0	(3)	0	0	8	8	(8)	0	0
30/4/2021	0	0	10	(2)	0	8	10	0	(7)	0	0
1/5/2021	8	0	0	(1)	0	7	10	0	(7)	0	0
2/5/2021	7	0	0	(4)	0	3	10	0	(10)	2	0
3/5/2021	3	0	0	(1)	(2)	0	10	10	(8)	0	0
4/5/2021	0	0	0	0	0	0	0	0	(6)	0	0
5/5/2021	0	0	8	(4)	0	4	8	0	(9)	0	0
6/5/2021	4	0	0	0	0	4	8	0	(5)	0	0
7/5/2021	4	0	0	(2)	0	2	8	0	(6)	0	0
8/5/2021	2	0	0	(2)	0	0	8	8	(8)	0	4
9/5/2021	0	0	0	0	0	0	0	0	(4)	0	0
10/5/2021	0	0	8	(2)	0	6	8	0	(6)	0	0
11/5/2021	6	0	5	(4)	0	7	13	0	(8)	0	0
12/5/2021	7	0	8	0	0	15	21	0	(6)	2	0
13/5/2021	15	0	0	0	(2)	13	21	8	(6)	0	0
14/5/2021	13	0	0	(1)	0	12	13	5	(5)	7	0
15/5/2021	12	(4)	0	(4)	(3)	1	8	8	(5)	1	4
16/5/2021	1	(1)	0	0	0	0	0	0	(5)	0	1
17/5/2021	0	0	0	0	0	0	0	0	(5)	0	3
18/5/2021	0	0	0	0	0	0	0	0	(4)	0	3
19/5/2021	0	0	0	0	0	0	0	0	0	0	2
20/5/2021	0	0	0	0	0	0	0	0	0	0	4
21/5/2021	0	0	0	0	0	0	0	0	0		
		(5)	58	(46)	(7)	0					

Figure D4.2: Single item model logic for stock balance, expiry reminder and purchase recommender

Appendix E - References

Market Research

1. *Food Waste Management*. (n.d.). National Environment Agency. Retrieved May 1, 2021, from https://www.nea.gov.sg/our-services/waste-management/3r-programmes-and-resources/food-waste-management

2. *Food Waste*. (n.d.). Zero Waste Masterplan. Retrieved May 1, 2021, from https://www.towardszerowaste.gov.sg/foodwaste/

3. Ng, G. (2018, September 10). *How supermarkets fight food waste in Singapore*. The Straits Times. https://www.straitstimes.com/singapore/how-supermarkets-fight-food-waste-in-singapore

4. Yeap, S., & Yeap, S. (2020, March 5). *These mobile apps combat food wastage in Singapore*. Lifestyle Asia Singapore. https://www.lifestyleasia.com/sg/food-drink/mobile-apps-combating-food-wastage-singapore/

5. *Fridge Pal*. (n.d.). AppPicker. Retrieved May 1, 2021, from https://www.apppicker.com/apps/496451091/fridge-pal

6. Assistant Secretary for Public Affairs (ASPA). (2020, July 30). *FoodKeeper App*. FoodSafety.Gov. https://www.foodsafety.gov/keep-food-safe/foodkeeper-app

7. *Epicurious – Recipes, Menu Ideas, Videos & Cooking Tips*. (2021, April 30). Epicurious. https://www.epicurious.com/

8. Bring! (n.d.). *Bring! Shopping List App for iOS & Android*. Retrieved May 1, 2021, from https://www.getbring.com/en/home

9. *Grocery Shopping List - rShopping - Apps on Google Play*. (n.d.). Google Play. Retrieved May 1, 2021, from https://play.google.com/store/apps/details?id=com.RayDarLLC.rShopping&hl=en

10. *Out of Milk - Shopping List*. (2012, December 23). App Store. https://apps.apple.com/us/app/grocery-shopping-list-out/id564974992

11. Kit, T. S. (2020, March 3). *Price comparison app for consumers to be rolled out across Singapore*. CNA. https://www.channelnewsasia.com/news/singapore/price-kaki-comparison-app-bargains-supermarket-food-12491564

12. *treatsure*. (n.d.). Treatsure. Retrieved May 1, 2021, from https://www.treatsure.co/

13. OLIO. (n.d.). *Helping to stop food waste in Singapore*. Retrieved May 1, 2021, from https://olioex.com/food-waste-in/singapore/

14. *Makan Rescue*. (n.d.). Save Money, Save Food. Retrieved May 1, 2021, from https://makanrescue.com/

15. Chia, L. (2020, December 22). *Rayner Loi hit the pause button on school to become the CEO of a technology start-up building AI trackers that make dustbins smart enough to achieve a step change in reducing food waste.* CNA. https://www.channelnewsasia.com/news/cnainsider/rayner-loi-food-

waste-warrior-made-in-singapore-smart-dustbins-11565270

16. Lumitics. (2020, November 6). *Waste Management Solution For Kitchen | Lumitics Singapore*. https://lumitics.com/waste-management-solutions/

Literature Review

1. Babu, S. C. (2021, April 22). *Automating Receipt Digitization with OCR and Deep Learning*. AI & Machine Learning Blog. https://nanonets.com/blog/receipt-ocr/#a-traditional-receipt-digitization-pipeline

2. D. (2020, October 7). *Receipt OCR Part 1: Image segmentation by OpenCV*. Kaggle. https://www.kaggle.com/dmitryyemelyanov/receipt-ocr-part-1-image-segmentation-by-opencv/

3. R.U., A.S., A.R., F.A., & R.M. (2018). OCR Engine to extract food-items, prices, quantity, units from receipt images, heuristics rules based approach. *International Journal of Scientific & Engineering Research*, *9*(2), 1334–1341. https://www.ijser.org/researchpaper/OCR-Engine-to-Extract-Food-Items-Prices-Quantity-Units-from-Receipt-Images-Heuristics-Rules-Based-Approach.pdf