

CSC3170 Project Report
EAT WHAT: An databse for takeaway in
CUHK(SZ)

Chen Hanzhi 118010011
Chen Jiaqi 118010014
Liu Jianing 118010182
Yang Yudi 118010373
Pan ao117010206

April 2021

Contents

1	Abstract	3
2	Introduction and Motivation	4
2.1	Motivation	4
2.2	Introduction	4
2.2.1	Review System	4
2.2.2	Pick-up Bounty Mission System	5
2.3	Result of Survey	5
3	Hardware Design	7
3.1	Objectives	7
3.2	Implementation	8
4	Software Design	10
4.1	UML Diagrams	10
4.2	Database Design	12
4.2.1	ER Diagram	12
4.2.2	Explanation for Tables	13
4.2.3	Highlights for the Database Design	16
4.3	Web Design	17
4.3.1	Frontend	17
4.3.2	Backend	17
4.3.3	Test Frame	18
5	Software Functionalities	18
5.0.1	Login and Register	18
5.0.2	Interface	19
5.0.3	Comments	20
5.0.4	Category	20
5.0.5	View new comments	21
5.0.6	Labels	22
6	Development Log and Testing Document	22
7	Conclusion, Self-evaluation and Future Work	22
7.1	Conclusion	22
7.2	Self-evaluation	22
7.3	Future Work	23

1 Abstract

We often hear voices like "Can you recommend a delicious takeaway" or "I don't know what to eat at noon today" on campus. Although the evaluations on the takeaway platform like "Meituan" or "Eleme" is open to students, everyone still encounters various troubles when choosing a restaurant to order their takeaways. Therefore, we did a survey, and the results showed that most students think that the evaluations' reference value on the existing food delivery platform is low, and they all need a more authentic food delivery evaluation platform for them to understand these restaurants. There are various needs for take-out food that have not been met for the time being. Therefore, we decided to use the knowledge of the database and website architecture, coupled with our own server, to build an on-campus takeaway evaluation platform for everyone to use. After a month of hard work, we completed the construction of the front-end web page and the back-end database and its connection. Our website has also been used by many internal beta users, and has accumulated a large amount of data on food delivery and evaluation around the school. At the same time, our platform has also been widely praised. Now that the platform can be accessed online through the wireless network, we will continue to improve the construction of the platform and promote it to achieve more practical functions and help more users.

2 Introduction and Motivation

2.1 Motivation

Today, what to eat for a meal has become a frequent and disturbing problem for modern people. And students in CUHK(SZ) campus also face the same problem as others. Our campus locates in a rather suburban area of Shenzhen, which set a frustrating limit to our daily dining option. Not only so, recently the safety of the food in our campus canteen has also become worrying. Therefore, many students choose to order food delivery from nearby restaurants on MEITUAN or ELEMA food delivery platform. While there are some good restaurants nearby, there are also some restaurants with only good appearance but bad quality. What's worse, restaurants can pay money to get large amount of high ratings and positive reviews that were supposed to be reflecting the actual opinions of customers.

We observed that, some of the students, with the intention to cope with this kind of problem, has formed WeChat groups to exchange comments on the nearby restaurants' delivery food. However, it is obvious that this method is quite inefficient. For example, the comments of individual students are unstructured data, and therefore make it difficult to form a general impression on a restaurant. And more importantly, the newcomers can hardly retrieve the history comments that others have already made. In the light of this, we decide to make a delivery-food reviewing system for students and teachers in campus which will only have sincere comments from our own people.

Another phenomenon we noticed is how much labor are wasted on the way of picking up our delivery. Because delivery guys are not allowed to enter the campus, students and teachers have to walk for 3 to 12 minutes to get the delivery. But actually, it is usually easy for one to take the deliveries of 3 to 4 people in one trip. Therefore, we also want to introduce a delivery pick-up service, provided by our users and enjoyed by our users.

2.2 Introduction

To provide above 2 major services, we decided to implement a web application. And to fulfill the requirements of functionality, we chose the traditional 3-layer structure of application design. We used a Raspberry Pi 3b+ to build a server to store the database and process admin and user requests. Finally, we chose MySQL as the database system. Our service consists of two parts: review system and a pick-up bounty mission system. The detail of the two services are introduced below:

2.2.1 Review System

This system is designed for users to browse and write reviews of nearby food deliveries. The database pre-stores some of the restaurants. If a user wants to write a review or browse the reviews for one particular restaurant, they can

directly search for the name of the restaurant, the web application would make a search in our database. But when users are not sure about what they want to find, they can also look for restaurants according to some requirements. We assign each restaurant with one type and several tags to represent their features. For example, when a user wants to eat Sichuan food, they can click on the type of “Sichuan Food”. Then only Sichuan food restaurants will be displayed. And if the user simply wants to have something spicy, they can click on the tag of “spicy”. Then every restaurant matching that tag would be shown. If the restaurant has not been recorded in our database just yet, user can also request to establish an entry for it. When giving a review, users can give rankings, write comments and add tags for their review. Although this review system is similar to that on the food delivery apps, most of our reviews will be authentic ratings and sincere comments because all users are required to verify their student or stuff identity before being able make comments.

2.2.2 Pick-up Bounty Mission System

This part helps our users to save time and labor picking up their delivery through a bounty system. In this system, students can pay a small amount to post missions that ask for somebody to pick up their delivery for them. And if a student is going to pick up their own delivery, they can accept a few missions posted on the website and take other people’s deliveries along. In this way, not only the students who are willing to go to pick up the delivery can earn money that may counter their delivery fee, but also those who wish to save some time or are unavailable to pick up their delivery can benefit from this trade.

2.3 Result of Survey

In order to verify our observations, we conducted several surveys among 200 students in CUHK(SZ). The result shows that there is a market demand for our application. The following are the result of our survey.

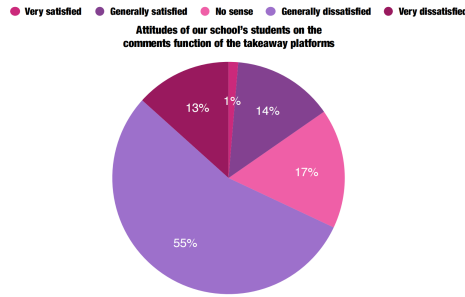


Figure 1: Attitudes of our school’s students on the comments funtion of the takeaway platdorms

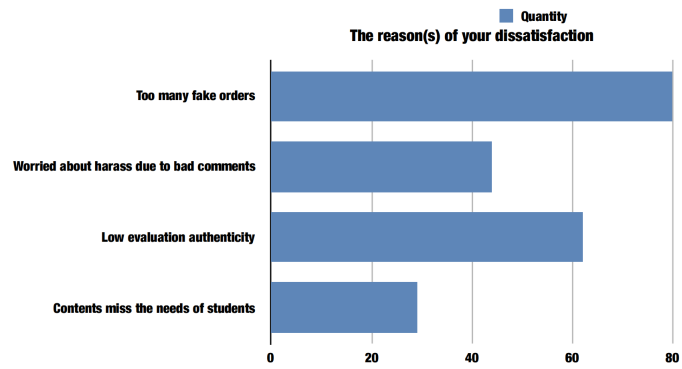


Figure 2: The reason of dissatisfaction

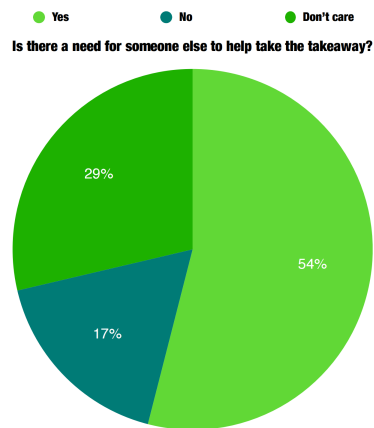


Figure 3: Need for someone to help take the takeaway

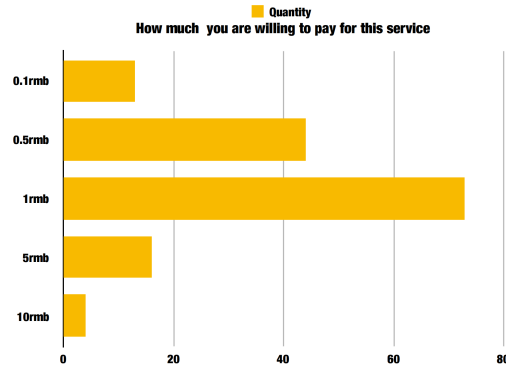


Figure 4: How much you are willing to pay for the "help takeaway" service

3 Hardware Design

For the convenience of maintenance and development, we choose to build our own server for the project. Now we use a Raspberry Pi 3B+ as our server hardware, and we use many other tools and services to make it a complete server for our project.

3.1 Objectives

At the very first period of our project, we decided that the server should be available at low cost, and have high extension malleability. Therefore, a single-chip computer with full-fledged software support and active developer communities is a better choice. The initial choices include the Raspberry Pi and the Arduino. Considering that the Arduino needs an extra Ethernet shield to work as a web server, and its low computing power, the Raspberry pi became our final choice.

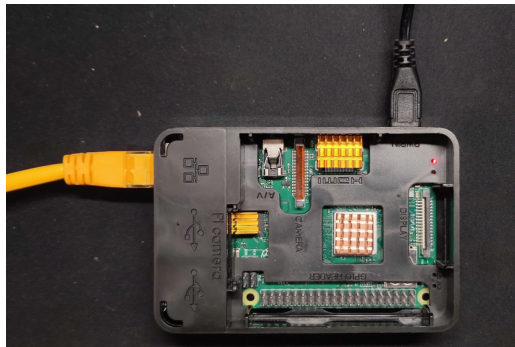


Figure 5: The Raspberry Pi Server

The Raspberry Pi community officially uses a Debian OS as Raspberry Pi's operating system. This version of Debian is based on ARM Linux, which makes further development more convenient.

3.2 Implementation

The installation of the operating system is very simple. The Raspberry Pi uses a TF card as storage and the installation is to write the required data into the card. After the installation of the operating system, the server then is accessed with a security shell. Then we go on to build other functions on our server.

- Intranet Penetration: Since our group members don't live on the same campus, we need intranet penetration to make the server accessible for every group member. The tool we use is called phddns (""). Phddns is a tool developed by Oray Technology, which supports ARM Linux and provides the DDNS service. We use the service to implement the remote SSH connection through a public network IP. Besides, we also use this service to make our website accessible from a domain name and a public network IP.

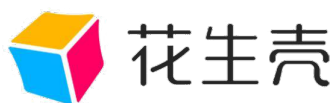


Figure 6: phddns service provided by Oray Technology

- Database: Our design of the project uses MySQL to build our database. Since MySQL stopped the support for ARM architecture after version 5.0, we change to use MariaDB to build the database. MariaDB is developed by the same group of developers of MySQL, it supports the syntax of MySQL and supports the InnoDB as the storing engine.
- HTTP Server: Despite that Django provides an integrated HTTP server for testing and development, it is far from enough for our project. The integrated server does not support concurrency and is not stable.



Figure 7: APACHE HTTP Server

We choose the Apache server and mod-wsgi to build our HTTP server, as is the recommended by Django community. However, during the test, we found that the apache server sometimes takes too many hardware resources. Therefore, we are going to change it with Nginx and the Gunicorn.

4 Software Design

4.1 UML Diagrams

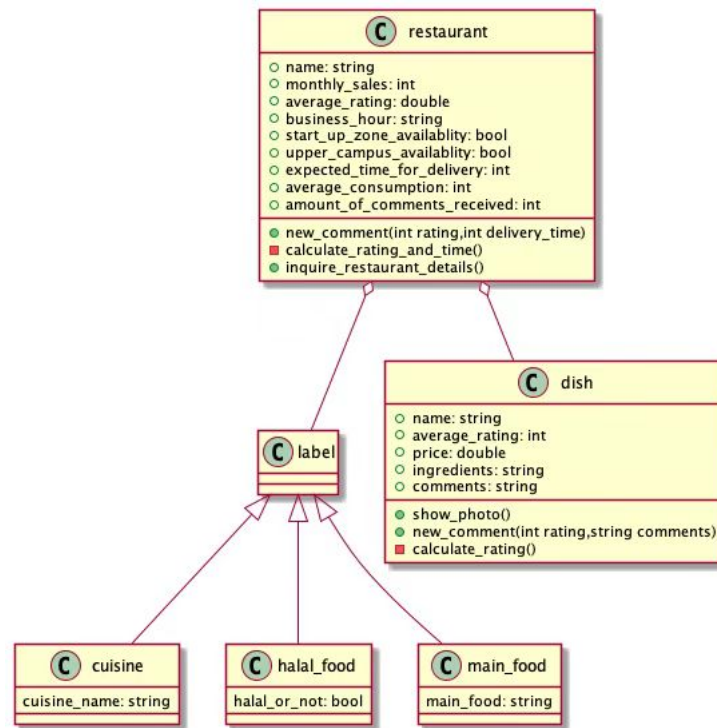


Figure 8: Comment on restaurants

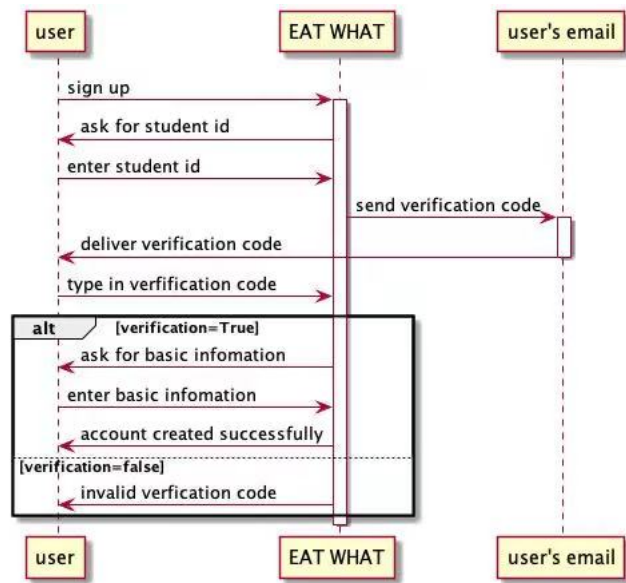


Figure 9: Sign in Sign up

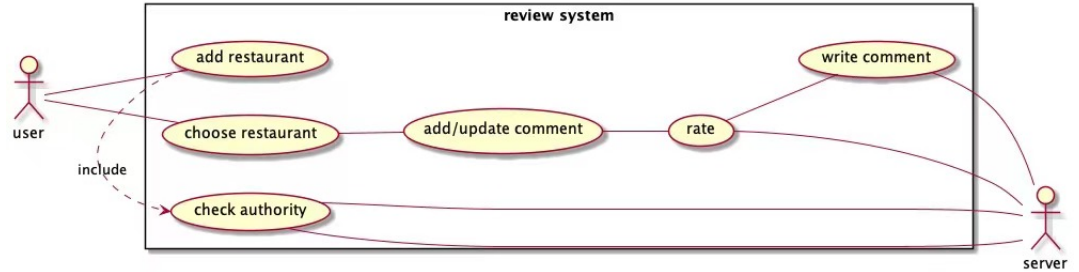


Figure 10: Comment on restaurants

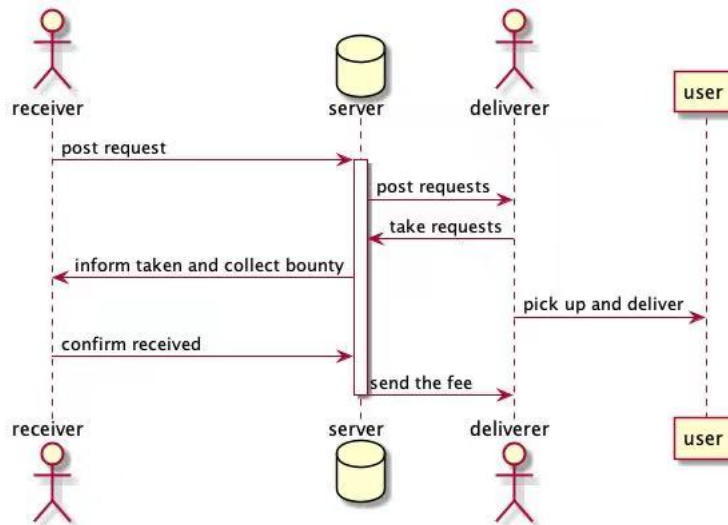


Figure 11: "Help to pick takeaway"

4.2 Database Design

Our application can be majorly divided into two parts: review system and a pick-up service system. We will introduce them separately.

4.2.1 ER Diagram

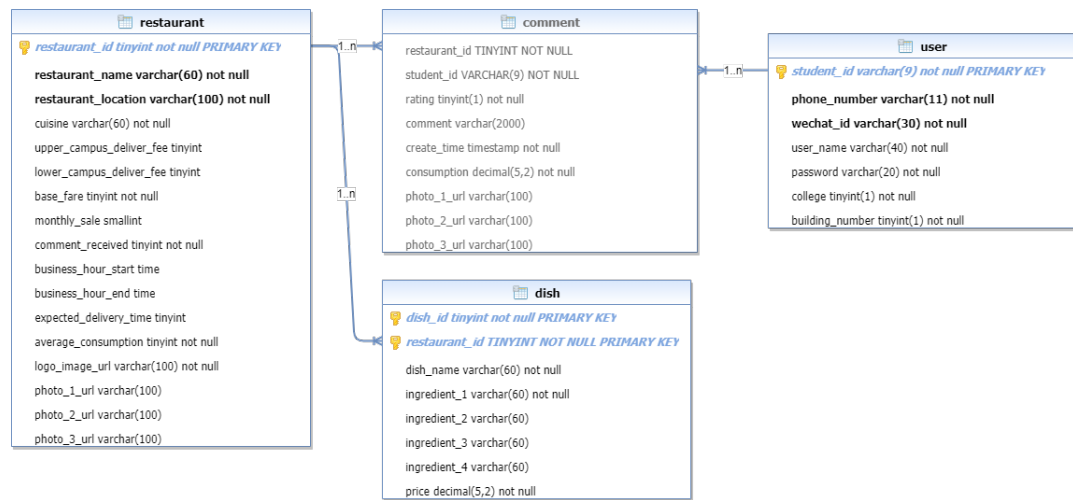


Figure 12: ER diagram for the takeaway database

4.2.2 Explanation for Tables

For our database, there are five tables: restaurant, dish, user, comment, and request.

For the ‘restaurant’ table, there are 23 attributes, having ‘restaurant-id’ as the primary key. Other basic information like its name and availability on different platforms are recorded as well.

```
mysql> desc restaurant;
```

Field	Type	Null	Key	Default	Extra
restaurant_id	tinyint	NO	PRI	NULL	
restaurant_name	varchar(60)	NO		NULL	
cuisine	varchar(60)	NO		NULL	
restaurant_location	varchar(100)	NO		NULL	
upper_campus_availability	tinyint(1)	NO		NULL	
lower_campus_availability	tinyint(1)	NO		NULL	
upper_campus_deliver_fee	tinyint	YES		NULL	
lower_campus_deliver_fee	tinyint	YES		NULL	
base_fare	tinyint	NO		NULL	
monthly_sale	smallint	YES		NULL	
comment_received	tinyint	NO		NULL	
business_hour_start	time	YES		NULL	
business_hour_end	time	YES		NULL	
halal	tinyint(1)	YES		NULL	
expected_delivery_time	tinyint	YES		NULL	
average_consumption	tinyint	NO		NULL	
logo_image_url	varchar(100)	NO		NULL	
photo_1_url	varchar(100)	YES		NULL	
photo_2_url	varchar(100)	YES		NULL	
photo_3_url	varchar(100)	YES		NULL	
meituan_availability	tinyint(1)	NO		NULL	
wechat_availability	tinyint(1)	NO		NULL	
eleme_availability	tinyint(1)	NO		NULL	

Figure 13: ‘restaurant’ table

For the ‘dish’ table, there are 4 attributes, having ‘restaurant-id’ and ‘dish-id’ as the primary key, where ‘restaurant-id’ is a foreign key referencing the ‘restaurant’ table. Other basic information like its name and price are recorded as well.

```
[mysql> desc dish;
```

Field	Type	Null	Key	Default	Extra
restaurant_id	tinyint	NO	PRI	NULL	
dish_id	tinyint	NO	PRI	NULL	
dish_name	varchar(60)	NO		NULL	
ingredient_1	varchar(60)	NO		NULL	
ingredient_2	varchar(60)	YES		NULL	
ingredient_3	varchar(60)	YES		NULL	
ingredient_4	varchar(60)	YES		NULL	
price	decimal(5,2)	YES		NULL	

```
8 rows in set (0.01 sec)
```

Figure 14: ‘dish’ table

For the ‘user’ table, there are 7 attributes, having ‘student-id’ as the primary key. Other basic information like user’s name and college are recorded as well. One student id is assigned to one account, for which this app can assure the authenticity of the comments and ratings. Once the student sign up on our app, an E-mail with verification code would be sent to the corresponding address.

```
mysql> desc user;
```

Field	Type	Null	Key	Default	Extra
student_id	varchar(9)	NO	PRI	NULL	
user_name	varchar(40)	NO		NULL	
password	varchar(20)	NO		NULL	
phone_number	varchar(11)	NO		NULL	
wechat_id	varchar(30)	NO		NULL	
college	tinyint(1)	NO		NULL	
building_number	tinyint(1)	NO		NULL	

```
7 rows in set (0.00 sec)
```

Figure 15: ‘user’ table

For the ‘comment’ table, there are 9 attributes, having ‘restaurant-id’ and ‘student-id’ as the primary key. Both ‘restaurant-id’ and ‘student-id’ are foreign key referencing the ‘restaurant’ table and the ‘user’ table. Other basic information like rating and consumption are recorded as well. In our app, one user is supposed to have at most one comment and one rating for one restaurant to ensure the ratings are objective and credible. This principle is the reason for the 2 pick of primary key. The users are allowed to update their comments and ratings, by which they can evaluate the restaurant more than once.

```
mysql> desc comment;
```

Field	Type	Null	Key	Default	Extra
restaurant_id	tinyint	NO	PRI	NULL	
student_id	varchar(9)	NO	PRI	NULL	
rating	tinyint(1)	NO		NULL	
comment	varchar(2000)	YES		NULL	
create_time	timestamp	NO		NULL	
consumption	decimal(5,2)	NO		NULL	
photo_1_url	varchar(100)	YES		NULL	
photo_2_url	varchar(100)	YES		NULL	
photo_3_url	varchar(100)	YES		NULL	

9 rows in set (0.00 sec)

Figure 16: 'comment' table

For the 'request' table, there are 7 attributes, having 'request-date' and 'request-number' as the primary key. 'Deliver-id' and 'receiver-id' are both referenced from 'user'. Other basic information like request time and delivery location are recorded as well. Once an order of delivery is released and a user respond to the order, a request is created.

```
MariaDB [PROJECT]> desc request;
```

Field	Type	Null	Key	Default	Extra
request_date	date	NO	PRI	NULL	
request_time	time	NO		NULL	
request_number	tinyint(4)	NO	PRI	NULL	
deliverer_id	varchar(9)	NO	MUL	NULL	
receiver_id	varchar(9)	NO	MUL	NULL	
deliver_detail	varchar(500)	NO		NULL	
upper_or_lower	tinyint(1)	NO		NULL	

7 rows in set (0.002 sec)

Figure 17: 'request' table

Multiple views are created for different uses. For various reasons, the complete table are hidden from users. For example, to protect user's privacy, the password is not supposed to be accessed by anyone but the user and the system.

```
MariaDB [PROJECT]> desc view_from_deliverer_117020003_2021_4_26;
```

Field	Type	Null	Key	Default	Extra
request_time	time	NO		NULL	
receiver_name	varchar(40)	NO		NULL	
college	tinyint(1)	NO		NULL	
building_number	tinyint(1)	NO		NULL	
deliver_detail	varchar(500)	NO		NULL	

5 rows in set (0.003 sec)

Figure 18: ‘View-for-deliverer=or-117020003-2021-4-25y

‘View-for-deliverer=or-117020003-2021-4-25’ is a view created for user with student id ‘117020003’ to check the orders of deliveries he takes on 25th of April, 2021. He could deliver the takeaways to the corresponding location.

4.2.3 Highlights for the Database Design

Third Normal form:

For our database, there are several “one-to many” and “many-to-many” relationships. For example, one restaurant may have multiple dishes, one user can write comment to different restaurants and respond to other users. The database may have some issues when performing query and decomposition if all comments and dishes are put in one single restaurant table. Therefore, we design our database in third normal form with divided the whole database into four main tables: restaurant (with primary key restaurant id), dish (with primary key restaurant iddish id), user (with primary key student id), comment (with primary key restaurant iddish id, one user can have at most one comment for one restaurant to avoid redundancy comments). This design can avoid redundancy and guarantee lossless composition and decomposition for further use.

Hashing:

For the security of the password storage, our project uses a hash function to encrypt the password. The function employs the SHA-256 algorithm developed by NIST.

SHA256 is a standard that belongs to the SHA-2 algorithms. SHA represents the Secure Hash Algorithm. For a SHA256 algorithm, it will generate a 256-bit Hash value for a piece of information with any length. The procedure of encryption is as below:

- Add some certain characters to the target information to make the length of the information divisible by 512. Since some certain strings have a certain hash value, we add “salt” to the password string to increase the security. We use the unique user ID as salt.

- 2. Divide the message into blocks that are 512 bits long (Since we have a length limit with the password input, we only have one block in this step,). Then do the calculate:

$$H^i = H^{i-1} + C_{M^i}H^{i-1}$$

Since we only have one block to calculate, the calculate is:

$$H = C_M$$

Where C is the compression function of SHA256, M is the message block.

4.3 Web Design

This is a light-weight web-based application. Our goal is to help building a platform for students in campus to share information about food and provide take-away delivery services in a crowdsourcing manner. The application can be divided into two major parts, the information sharing platform and the delivery information platform. Both platforms include the frontend program and backend database.

4.3.1 Frontend

The current plan of the project is to use php to develop the frontend software, including the whole user interface and all the functions and module that translate the users' requirement into SQL codes. The main page of this application includes a search box and a space for pushing the restaurants with top rankings. The searching mechanism of this application highly rely on the labels. The users can search for restaurants in two ways, either directly search for a restaurant via its name or use labels to narrow down the available range of restaurants. This mechanism can both ensure the simplicity of the procedure of finding a restaurant, especially when the user have little idea about what to eat, and minimize the amount of query to the database. Besides, through establishing connections between labels, it is easy to push more restaurants out of the range that the user sets, but with more similarity to the users' initial needs. The main page of the application offers an entrance to the delivery service platform. The delivery service platform automatically refreshes every 15 minutes. Instead of using a backend MySQL server to store data, the delivery demand is stored directly in the server for quicker change of its status.

4.3.2 Backend

For the information platform, we use a MySQL serve to store the information of the restaurants. The frontend application sends query to the server and the server returns the result. As for the delivery service platform, the order is

taken as an object. A list is used to record the real time information of delivery demand.

4.3.3 Test Frame

Since our application is a web-based application, the test frame is implemented with python and the selenium library. Test to be considered includes the test on the algorithm on pushing and ranking strategy, and the pressure test on the delivery service platform.

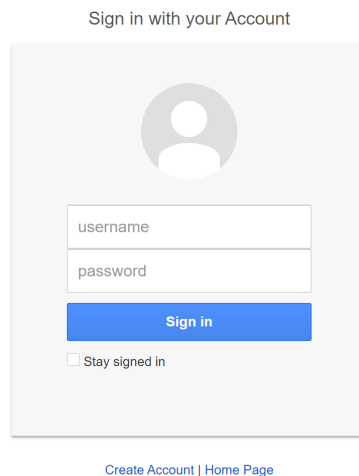
5 Software Functionalities

For details of functionalities, please refer to the user document.

A brief introduction to some basic interfaces and functionalities are shown below.

5.0.1 Login and Register

Sign in with your Account



The login form is centered on the page. It features a light gray background with a white border. At the top, there is a gray circular icon representing a user profile. Below the icon are two input fields: one for 'username' and one for 'password'. A blue 'Sign in' button is positioned below the password field. At the bottom of the form, there is a checkbox labeled 'Stay signed in'. Below the form, there are two links: 'Create Account' and 'Home Page'.

username

password

Sign in

☐ Stay signed in

[Create Account](#) | [Home Page](#)

Figure 19: Login

5.0.2 Interface



Figure 20: Website interface 1

5.0.3 Comments



Figure 21: Website interface 2

5.0.4 Category



Figure 22: Select certain category

5.0.5 View new comments

新评价

tox 发表在《[周记汕头牛肉（牛肉粿条）](#)》

SynthMob 发表在《[韩式烤冷面（爱联店）](#)》

SynthMob 发表在《[学活二楼自选](#)》

SynthMob 发表在《[粤记煲仔饭](#)》

pa 发表在《[粤记煲仔饭](#)》

新店铺

[一点点](#)

[周记汕头牛肉（牛肉粿条）](#)

[喜茶：深圳龙岗COCO Park店](#)

[麦当劳麦乐送（深圳大运得来速店）](#)

[王大爷重庆小面](#)

[韩式烤冷面（爱联店）](#)

[粤记煲仔饭](#)

[学活二楼自选](#)

[肯德基](#)

[木屋烧烤](#)

Figure 23: View new comments

5.0.6 Labels

标签云

免配送费 0起送费 配送
时间短 品牌商家 跨天
预订

Figure 24: Labels for restaurants

6 Development Log and Testing Document

See the attached file "Development Log and Testing Document".

7 Conclusion, Self-evaluation and Future Work

7.1 Conclusion

In order to solve the students' trouble with the evaluation of the take-away platforms, we finally built a new platform built by Python Django and Mysql. In this progress, we built a sound database, which contains restaurants, dishes, users, comments and requests, then we use python Django to connect the database in Mysql to the website which is programmed in HTML. And as for the Web front end, there are several parts such as login, browse, keyword search, classification and labeling, comment, reply, which are connected to the SQL statements in the databases, saving, changing or getting the data. This Website based on the Takeaways database integrate all takeaway resources and provides real evaluation information. Till now, tens of users have been registered in to the website, and tens of comments has been uploaded. We have visited the internal beta users who have already used the website. The vast majority of users think that the website's evaluation is very useful for them, and believe that as time and users grow, the number of comments increases. The usefulness of the website will also become higher and higher, and it will become an indispensable part of school users when ordering takeaways.

7.2 Self-evaluation

Our website and database basically cover the students' need for take-out information. At the same time, the framework can be used in any school. As long as our server has a large enough capacity, it can be used by major schools across the country. And establish a database for each school, because the restaurant and other data are uploaded by the users themselves, so we only

need to ensure the stability of the server, so the practicability and expansion benefits of the project are very high. At the same time, the security of our website is also very reliable. The user's password is encrypted multiple times by a hash function and iterated to a very safe state for storage. And a campus account can only register one account, so our user quality is obtained in order to ensure sufficient protection, this also directly determines the high quality of the evaluation on the platform. Meanwhile, our solid database structure can ensure the stability of website data. Furthermore, our website can be used on any device. We have made functions that adapt to various resolutions so that users can easily access our webpage and use it on mobile phones, tablets, or computers, and it will maintain its consistent beauty.

7.3 Future Work

In the future, we will continue to develop Eat-What food-delivery comment system. This web application is highly extendable. We would add some new features in the upcoming updates. The first one would be the rating system, which will provide a simpler overview of the quality of an eatery. And we would like to allow our users to upload pictures in their reviews. Finally, we will add a Pick-up Bounty system, which allow our students to help each other pick up food-delivery by posting and accepting bounty missions. This feature would save much time and labor for those students and teachers that are in short of time. And this would be the new tables added to the database once we upgrade this feature on the website. Up to now, a lot of our work has achieved very good results in this project. In the future, we will further improve the platform based on the above methods and put it to more users to provide convenience for more users.