Efficient Management System for Small-Scale Restaurants

[1]Shen Wang, [2]FeiCao, [2]Douhao Ma
[1]Software Engineering Systems, Northeastern University
[2]Information Systems, Northeastern University

wang.shen3@northeastern.edu, cao.f@northeastern.edu, ma.dou@northeastern.edu

***Abstract*—Mr. Ping, the goose father of panda Po and the noodle restaurant owner in the famous 2008 dreamwork animation Kungfu Panda, is running the restaurant all on his own, with his son. It's a typical reflection of the small scale business in catering industry in China and many countries of the world. They are the majority of restaurants and the need of a efficient small scale restaurant management system is, therefore, ubiquitous. Utilizing java[1] and javafx[2], we developed an efficient management system for small-scale restaurants, where only 2 users roles and functions are implemented. This system can help any small scale catering business to increase their work efficiency with no burden on expenses.**

Keywords*—*small scale, catering, management system, java, javaFX,

I. Problem Description

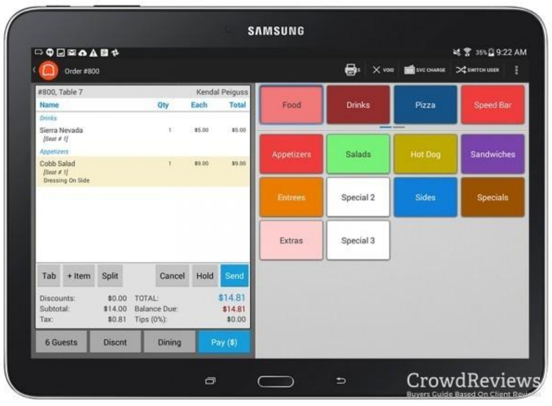
The catering industry, especially traditional small scale restaurants with only in person service, has been hit hard by COVID19 pandemic [3]. And now, as the pandemic faded out from our lives and all the industries are reviving, many restaurant owners will seek to continue their traditional way of "sit-in" service and restore the old-fashion enjoyment of gourmet experience. More and more small scale restaurants will emerge in the industry. Meanwhile, the economic stand of these kind of restaurants are not very optimistic under the current trend of fast development of food delivery service. There are already many restaurant management systems developed for the catering industry, such as Toast POS [4], Square POS [5] and many other applications of this kind. However, the price of these software packages are also not cheap, making it a burden for the small scale restaurants to purchase. For example, Toast POS's basic packages start at $79 per month, and premium packages may cost more. And the function of such software is also over powered for a "sit-in" style traditional restaurant. An affordable and efficient management system for small-scale restaurants will be meeting the growing need of such resturants.

The main function of these kind of management system will cover the fundamental aspect of running a traditional restaurant. Such as getting the order from the customer, transition between the waiters and the cook, and delivering the orders. And managing the menu of the restaurant is also an important part of every traditional restaurant. The software we designed focuses on these fundamental aspect of restaurant managing affairs and also have sufficient support for the financial aspect of the restaurant, such as the calculating the total sale, the total material cost, salary cost, taxes and the total profit, providing dependable aid to every single restaurant owner of small scale business.

II. Analysis (Related Work)

A restaurant management system (RMS) is a software package that helps restaurants management. Some famous restaurant management systems include Toast [4], touchbistro [6], Upserve [7], ShopKeep [8]。 All these software covers a lot of different functions, some even include a community or a user ecosystem. However, the most fundamental part of these software is the so-called POS system.

A point of sale (POS) system is a software and hardware system that is used to process payments and track sales in a retail store. It is also known as a retail management system. A POS system typically includes a cash register, a receipt printer, a credit card reader, and a barcode scanner. It may also include other features such as inventory management, customer loyalty programs, and marketing tools.

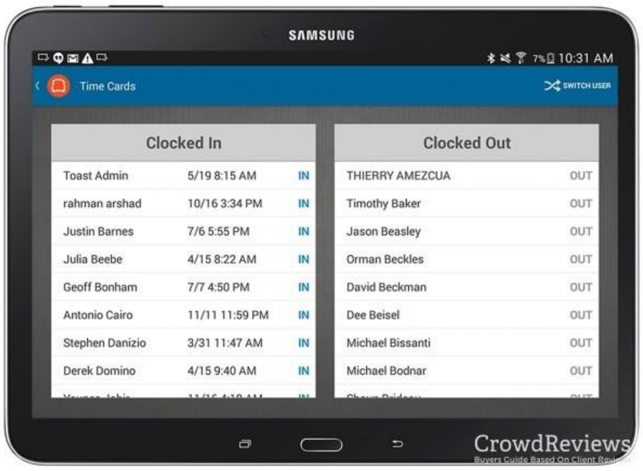


#800, Table 7 — Kendal Peiguss

Name	Qty	Each	Total
Drinks			
Sierra Nevada *[Seat # 1]*	1	$5.00	$5.00
Appetizers			
Cobb Salad *[Seat # 1]* Dressing On Side	1	$9.00	$9.00

Discounts:	$0.00	TOTAL:	$14.81
Subtotal:	$14.00	Balance Due:	$14.81
Tax:	$0.81	Tips (0%):	$0.00

Figure 1. order management page of Toast POS [9]



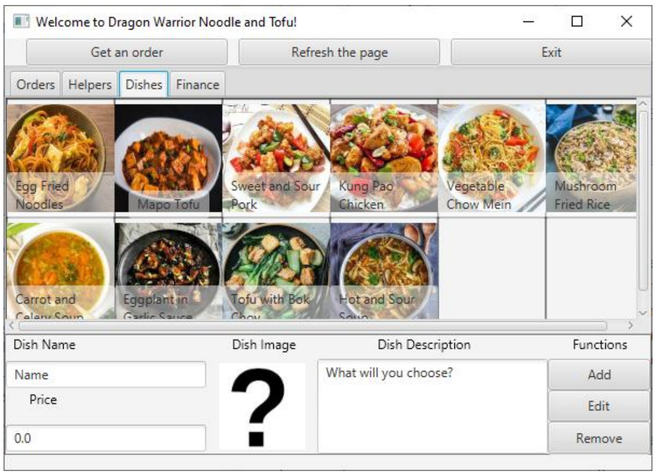
Clocked In			Clocked Out	
Toast Admin	5/19 8:15 AM	IN	THIERRY AMEZCUA	OUT
rahman arshad	10/16 3:34 PM	IN	Timothy Baker	OUT
Justin Barnes	7/6 5:55 PM	IN	Jason Beasley	OUT
Julia Beebe	4/15 8:22 AM	IN	Orman Beckles	OUT
Geoff Bonham	7/7 4:50 PM	IN	David Beckman	OUT
Antonio Cairo	11/11 11:59 PM	IN	Dee Beisel	OUT
Stephen Danizio	3/31 11:47 AM	IN	Michael Bissanti	OUT
Derek Domino	4/15 9:40 AM	IN	Michael Bodnar	OUT

Figure 2. employee management page of Toast POS [9]

POS systems are used by a variety of businesses, including retail stores, restaurants, and hotels. They can help businesses to improve their efficiency and profitability.

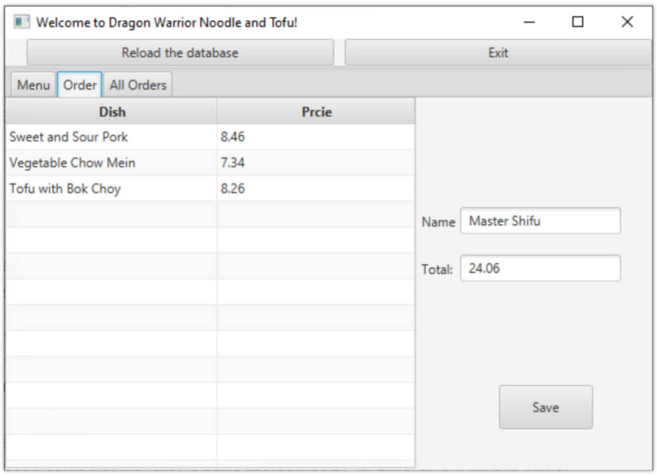
III. System Design

Duo to the simplicity nature of our restaurant management system, there are 2 main pages only. The owner page and the helper page. Given the small scale of the customer group, most of them are close in relationship. A considerable percent of the target customers could be running a family style restaurant, where the owner and the employees are family or relative relationship. That's the reason we used the term "helper", instead of the more formal term "employee". These kind of business resemble the noodle soup restaurant of Mr. Ping and Kungfu Panda which inspired our work.



Name
Price
0.0
What will you choose?

Figure 3. Screenshot for owner page



Dish	Prcie
Sweet and Sour Pork	8.46
Vegetable Chow Mein	7.34
Tofu with Bok Choy	8.26

Name: Master Shifu
Total: 24.06

Figure 4. Screenshot for helper page

Besides these two pages of main functionalities, there is one more page for user login. It was name "Welcome page":

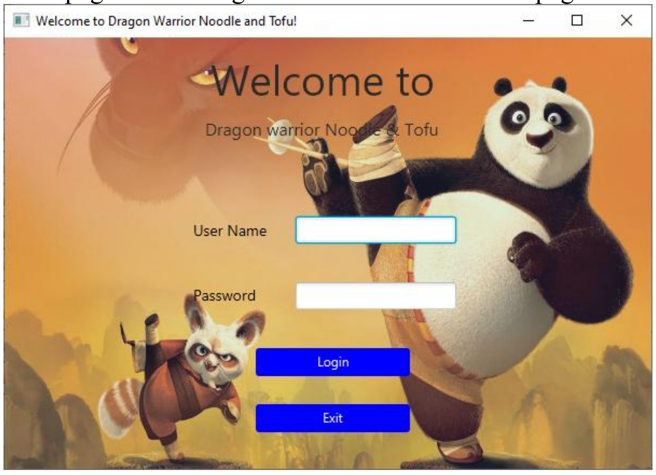

Welcome to Dragon Warrior Noodle and Tofu!
Welcome to
Dragon warrior Noodle & Tofu
User Name
Password
Login
Exit

Figure 5. Screenshot for welcome page

This software covers functionalities of 2 main characters: Owner and Helper.
Tentatively, we have assigned an owner account

User Name: Ping
Password: soup

(The name of the goose father in Kungfu Panda and his favorite food, inspired by his most well-known quote "*We are noodle folk. Broth runs through our veins!*".) [10]

And also a preset helper account is assigned:

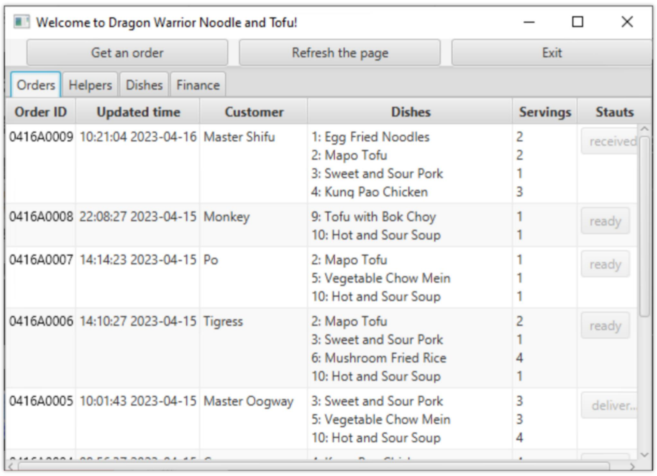
User Name: Po
Password: baozi

(The name of Kungfu Panda and his favorite food, as can be comprehended from the movie.) [11]

<u>Part I: The Owner Page:</u>

There are 4 tabs in the owner page, covering the 4 main functions of the owner:

Tab 1: Owner Page: Order management



Order ID	Updated time	Customer	Dishes	Servings	Stauts
0416A0009	10:21:04 2023-04-16	Master Shifu	1: Egg Fried Noodles 2: Mapo Tofu 3: Sweet and Sour Pork 4: Kung Pao Chicken	2 2 1 3	received
0416A0008	22:08:27 2023-04-15	Monkey	9: Tofu with Bok Choy 10: Hot and Sour Soup	1 1	ready
0416A0007	14:14:23 2023-04-15	Po	2: Mapo Tofu 5: Vegetable Chow Mein 10: Hot and Sour Soup	1 1 1	ready
0416A0006	14:10:27 2023-04-15	Tigress	2: Mapo Tofu 3: Sweet and Sour Pork 6: Mushroom Fried Rice 10: Hot and Sour Soup	2 1 4 1	ready
0416A0005	10:01:43 2023-04-15	Master Oogway	3: Sweet and Sour Pork 5: Vegetable Chow Mein 10: Hot and Sour Soup	3 3 4	deliver...

Figure 6. Screenshot for owner page, order tab

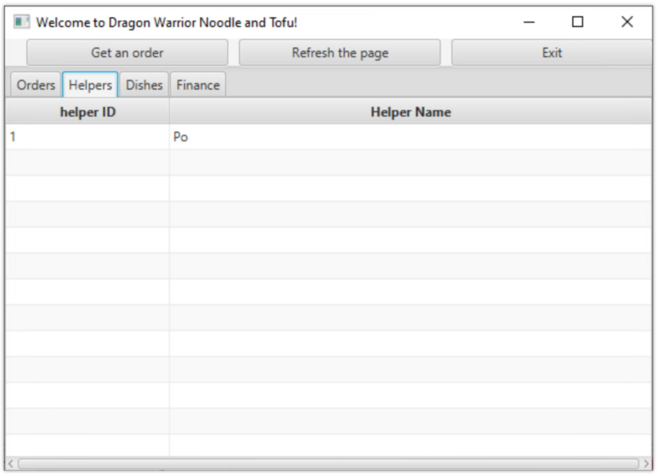
The main feature of this tab is a TableView, showing all the records of orders of the restaurant. The new records are on top while the old ones are at the bottom.
The columns of the TableView includes these information:

1. The order ID
2. The time and date when it was generated
3. The customer name
4. The list of dishes
5. The number of each dishes
6. The status of the order

It can be noted that the price of the order is not included in this view. It was due to the purpose of this page. The design of this page is to help the owner, usually the cook of the restaurant to handle the cooking. The cook can learn from this table which order is awaiting and which order is already taken care of. At this cooking stage, from the perspective of a cook, the price of the order is not the main concern.

It can also be noted that the last column is special and different from all other columns. There are buttons, instead of mere text information on the column. These buttons show the current status of each order, and also allow the owner to update the status when some action is taken to the order.

Tab 2: Owner Page: Employee management



helper ID	Helper Name
1	Po

Figure 7. Screenshot for owner page, helper tab

This view is simplified, as there is only 1 helper.

Tab 3: Owner Page: Menu management

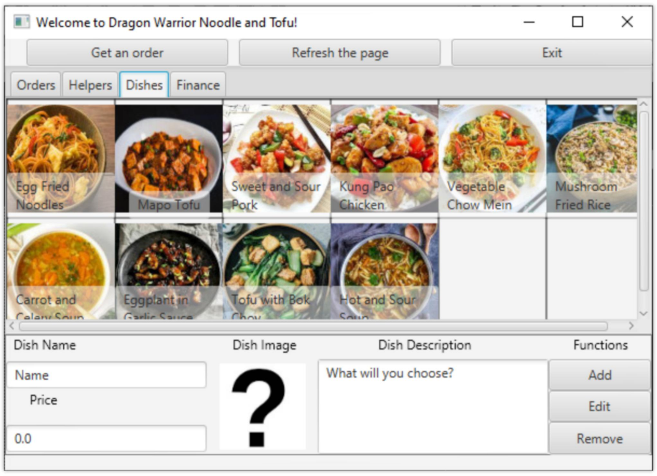


Figure 7. Screenshot for owner page, menu (dishes) tab

The main function of the owner is embodied in this tab.
The information about the dishes include

1. Images of dishes are shown in a gallery
2. Names of dishes on top, as a button
3. Dish information pops in the area below:
    1. Dish Name
    2. Dish Price
    3. Dish Image
    4. Dish Description
4. 3 main functions buttons are available depending on situation
    1. Add a dish
    2. Edit a dish
    3. Remove a dish

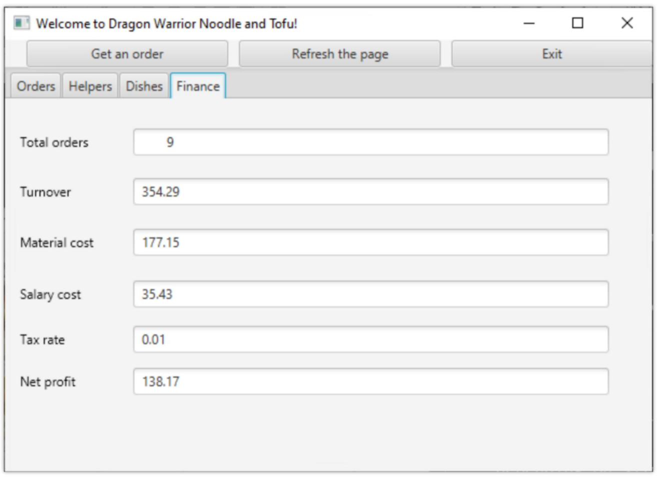
Instead of using a TableView, we chose a GridPane to show the dish. A StackPane is employed for every single dish, upon which there lay a ImageView holding an Image object and a

button showing the name text of the dish. The button is placed at the bottom of each cell of the GridView with a 72% transparency, making the whole image visible to the user.
The functionality of the buttons will be explained in the implementation part.

The bottom left area is the “information area” of the menu management. The attributes of each dish will be shown here once it is selected.

The ImageView, holding the image of a question mark when the page is initialized, is a button as well. Upon click, it will allow the user to choose a new image for the dish, by calling out a pop-up dialogue.

Tab 4: Owner Page: Financial report tab



Total orders	9
Turnover	354.29
Material cost	177.15
Salary cost	35.43
Tax rate	0.01
Net profit	138.17

Figure 8. Screenshot for owner page, Finance tab

The purpose and design of these page is to show the financial status of the restaurant.
Seeing the financial report, including these aspects:

1. Total number of orders
2. Total amount of sale (turnover)
3. Total material cost
4. Total salary cost
5. tax rate
6. Total profit

<u>Part II: The helper page:</u>

Tab 1: Helper Page: Menu

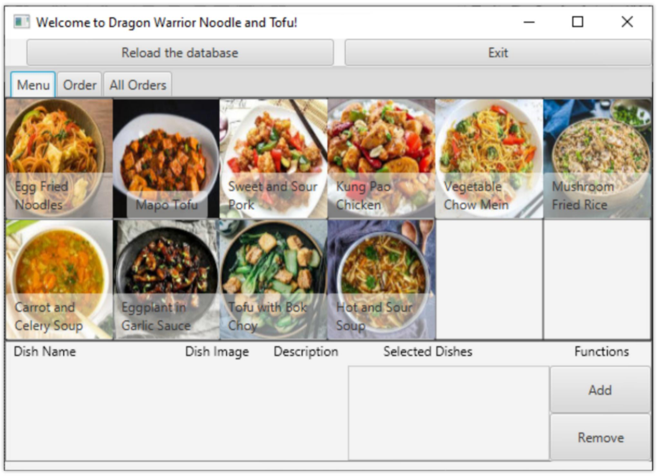


Figure 9. Screenshot for helper page, menu tab

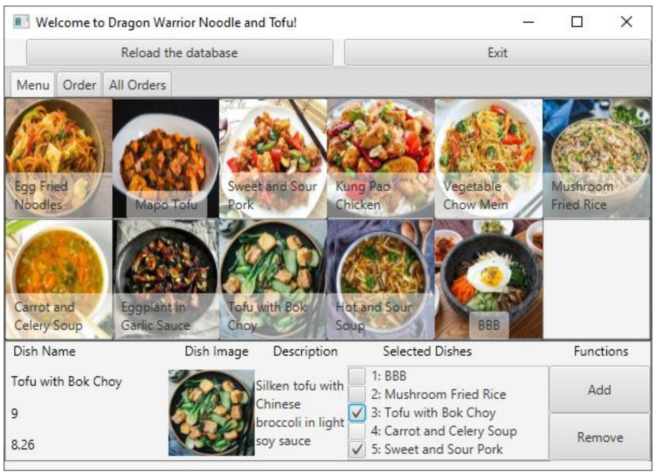
The same gallery design with the dishes tab on the owner page is employed for this page, showing the reusability of the code.

Upon clicking on the dish picture, the dish info will pop up at the bottom area, including:

Dish Name
Dish ID
Dish Price
Dish Image
Dish Description

Note that it is different from the owner page, since the purpose of this page is for the helper to assist the customers to place order by selecting dishes. There is no need for editing any dish. The information area is therefore set to read only, meaning that all the text fields are not editable, only for showing the information, without writing function and accessibility.

There is a special area under the “selected Dishes” area. It is for showing the list of selected dishes. When a dish was added to selected list, it will be shown here, with its index in the list and a checkbox in front.



Dish Name	Dish Image	Description	Selected Dishes
Tofu with Bok Choy 9 8.26		Silken tofu with Chinese broccoli in light soy sauce	☐ 1: BBB ☐ 2: Mushroom Fried Rice ☑ 3: Tofu with Bok Choy ☐ 4: Carrot and Celery Soup ☑ 5: Sweet and Sour Pork

Figure 10. Screenshot for helper page, menu tab, with “Selected Dishes” box populated

Figure 10 shows the screenshot when the ““Selected Dishes” box” is populated. If there are more than these dishes, the scroll bar will be activated as the vertical box (vbox) is lay upon a ScrollPane.

The helper can check the checkbox in front of the food name to mark the dishes and delete them from the “Selected Dishes” list.

“Add” button: Add the dish to the “Selected Dish” box, along with a check box. Will appear in “Order” page as well.
“Remove” button: Remove the selected (with a checked box) dishes from the “Selected Dish” box. Will affect “Order” page as well.

“Remove” button: Remove the selected (with a checked box) dishes from the “Selected Dish” box. Will affect “Order” page as well.

Tab 2: Helper Page: Order

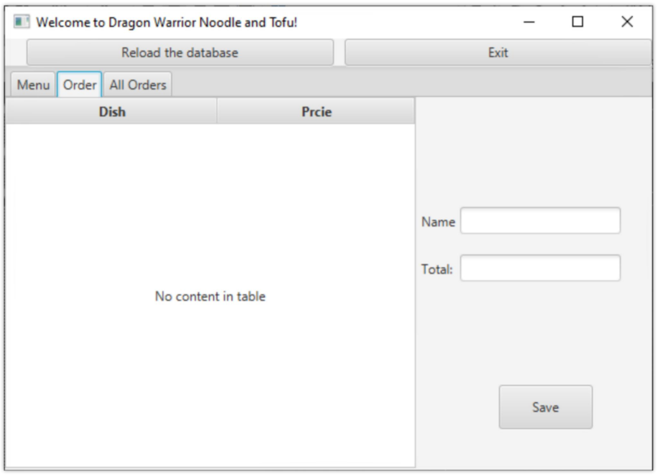
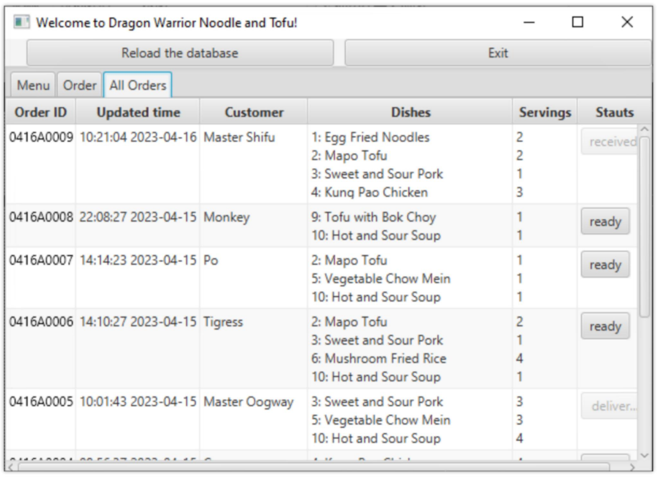


Figure 11. Screenshot for helper page, Order tab

Selected dish and its price will be shown on this page.
The helper needs to fill in the customer name
The total price text box will be populated automatically with the calculated value
Upon clicking on “Save”, these will done.
1. A new order will be created and added to the all order list
2. A new entry will be saved in the order data base
3. The All Orders page will be affected

Tab 3: Helper Page: Order

Same design with the orders tab in owner’s page
When a new order is put down
it will appear here
and the new order will come on top
The status buttons toggle status from ready to delivered.



Order ID	Updated time	Customer	Dishes	Servings	Stauts
0416A0009	10:21:04 2023-04-16	Master Shifu	1: Egg Fried Noodles 2: Mapo Tofu 3: Sweet and Sour Pork 4: Kung Pao Chicken	2 2 1 3	received
0416A0008	22:08:27 2023-04-15	Monkey	9: Tofu with Bok Choy 10: Hot and Sour Soup	1 1	ready
0416A0007	14:14:23 2023-04-15	Po	2: Mapo Tofu 5: Vegetable Chow Mein 10: Hot and Sour Soup	1 1 1	ready
0416A0006	14:10:27 2023-04-15	Tigress	2: Mapo Tofu 3: Sweet and Sour Pork 6: Mushroom Fried Rice 10: Hot and Sour Soup	2 1 4 1	ready
0416A0005	10:01:43 2023-04-15	Master Oogway	3: Sweet and Sour Pork 5: Vegetable Chow Mein 10: Hot and Sour Soup	3 3 4	deliver...

Figure 12. Screenshot for helper page, “All order” tab

IV. Implementation

A. *Outline*

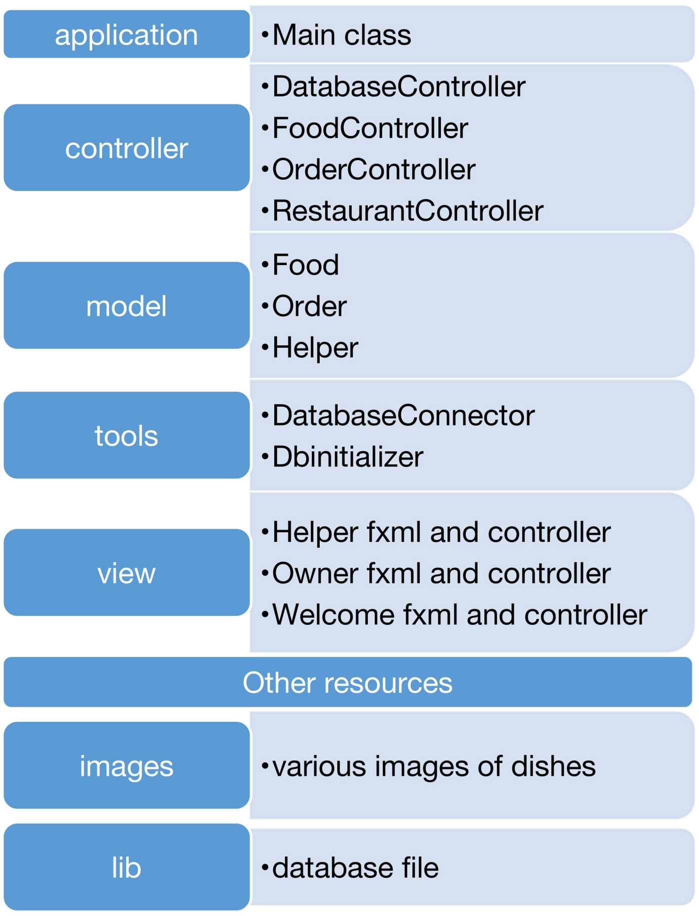
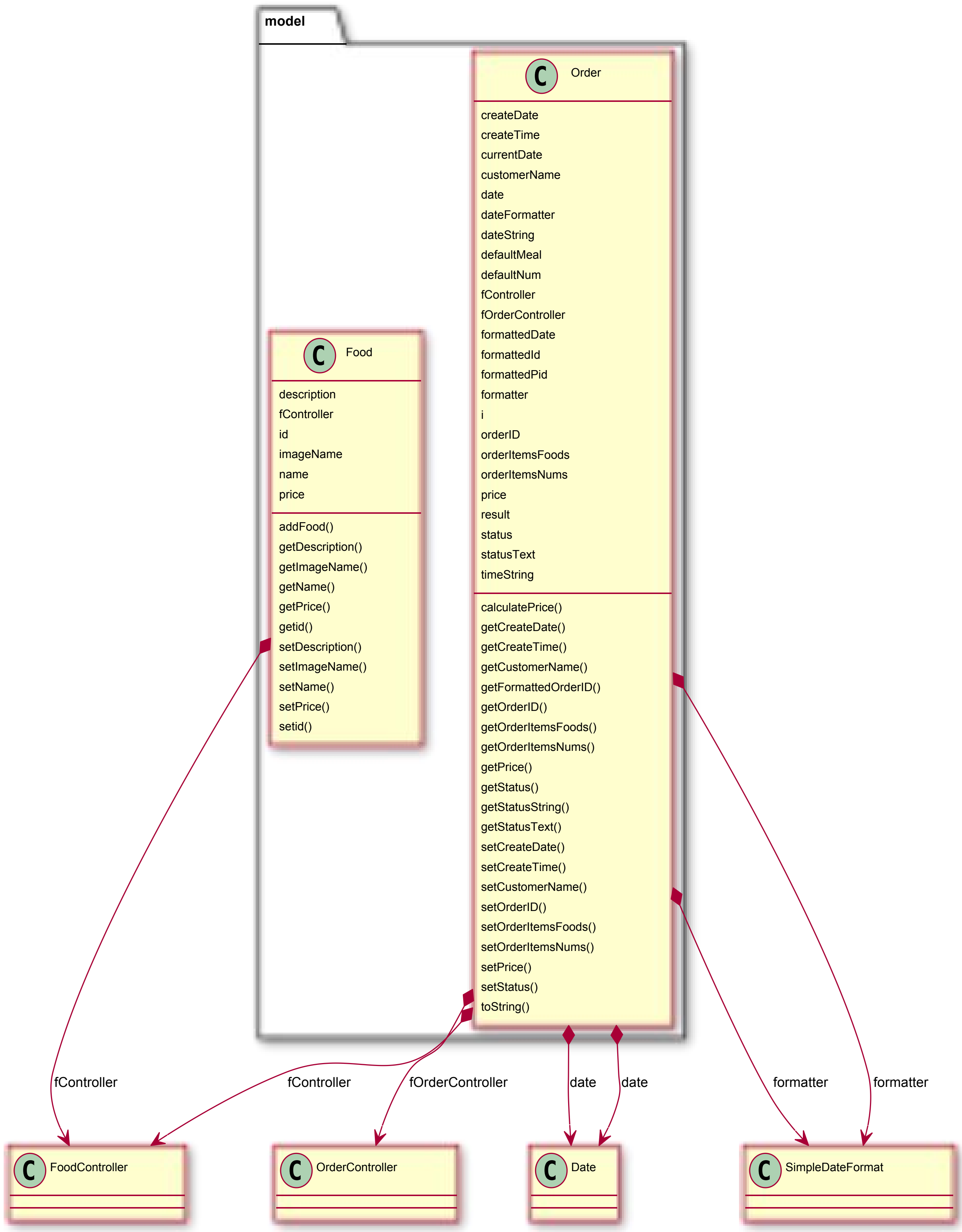


Table 1: the outline of all the resources of source code

Figure 13. UML diagram of Food and Order classes, Generated by GitUML www.gituml.com [12]



B. *Class definition*

In this work, there are 2 main classes that are fully featured. These classes are stored in the "model" package.

Class: Order

Class: Food

Figure 13 on the previous page is the UML diagram for these 2 classes.

C. *TableView population*

One difficult part of the implementation of the system is the TableView population.

Different data types / objects are passed to the TableView for demonstration.

In javaFX, in order to populate a TableView, both the TableView object, as well as the Column objects in them shall have reference id, in order to be referred to.

```
// Widgets on the Tab "Orders"
@FXML
private TableView<Order> ordersTableView; //
Point to the whole table, and below are each
colomn
@FXML
private TableColumn<Order,String> Order_ID;
@FXML
private TableColumn<Order, String>
Updated_time;
@FXML
private TableColumn<Order, String>
Customer_Name;
@FXML
private TableColumn<Order, String>
Order_Dishes;
@FXML
private TableColumn<Order, String> Servings;
@FXML
private TableColumn<Order, String> Status;
```

A sample code is as below:

And the "setCellValueFactory() " method of the column object is called to point the content of the column to a specific field of the Class shown in the TableView.

Taking the definition of the Servings Column as an example:

```
// 5th column, all the dishes' servings
Servings.setCellValueFactory(new
Callback<CellDataFeatures<Order,String>,
ObservableValue<String>>() {
@Override
public ObservableValue<String>
call(CellDataFeatures<Order, String> data) {
Order currentOrder = data.getValue();
String servingString = "";
for (Integer num:
currentOrder.getOrderItemsNums()) {
servingString += num+"\n";
}
return new SimpleStringProperty(servingString);
}
});
```

This code is used to set the value of the 5th column in a table, which is used to display the number of servings for each dish. The code first gets the current order from the data object. It then creates a string variable called servingString. The code then iterates through the getOrderItemsNums() method of the order object. For each number in the list, the code adds the number to the servingString variable, followed by a newline character. The code then returns a SimpleStringProperty object with the value of the servingString variable.

This code is an example of how to use JavaFX to bind the value of a cell in a table to the value of a property in an object.

In this process, an ObservableList data type is required as the input for the setCellValueFactory method.

And after the definition of all the columns are done, the function of

```
orderList = fOrderController.getOrders();
orderShowList.addAll(orderList);
FXCollections.reverse(orderShowList);
```

D. *Event listeners*

One of the requirement for the managing systems is that the functional buttons shall be active to users at given condition. If this is not implemented, then the users will be uninformed and tend to make erroneous operations.

An event listener can be implemented like the example below:

```
/****Setting the menu page buttons'
availability******/
//This is done by adding textfield listeners
dishName.textProperty().addListener((observable
, oldValue, newValue) ->
{checkAndUpdateButton();});
dishDescription.textProperty().addListener((obser
vable, oldValue, newValue) ->
{checkAndUpdateButton();});
price.textProperty().addListener((observable,
```

With a well-defined checkAndUpdateButton() method, the buttons on the page can be set active/inactive conditionally.

V. EVALUATION

The result of the project.

Below is a flowchart of a test run of the project.

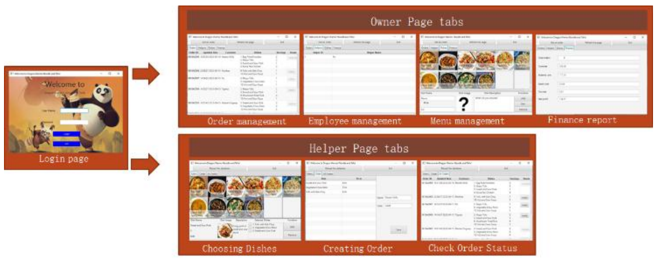


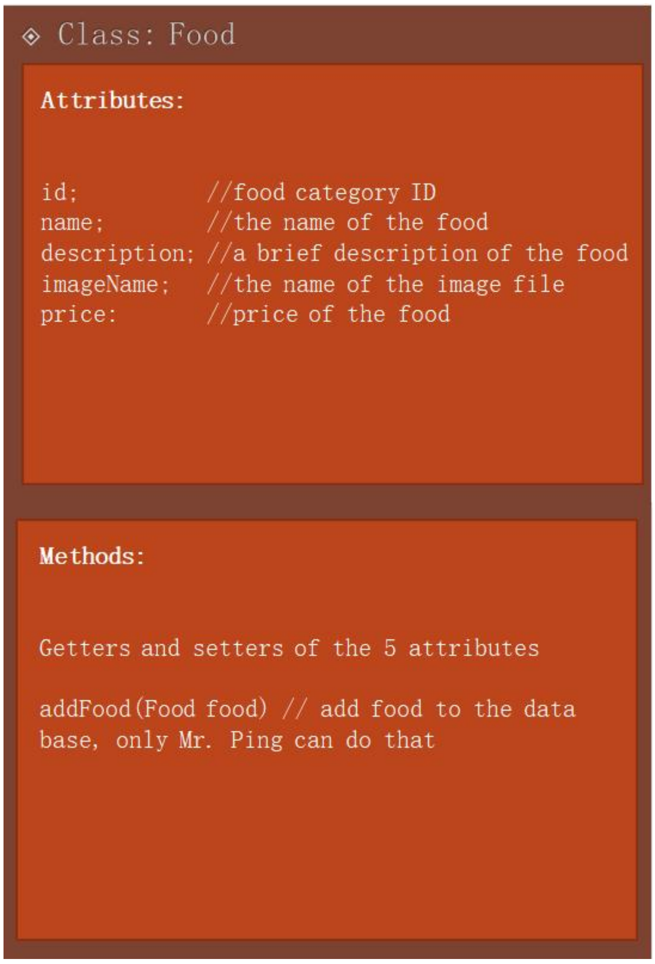
Figure 14. a flowchart of a test run of the project

VII. DISCUSSION (REFLECTION)

The topics in the course CSYE6200 are covered in this work. Below is some discussion about these covered topics.

Topic 1: Class Definition

2 main classes are involved in this project, Food and Order.



◈ Class: Food

Attributes:

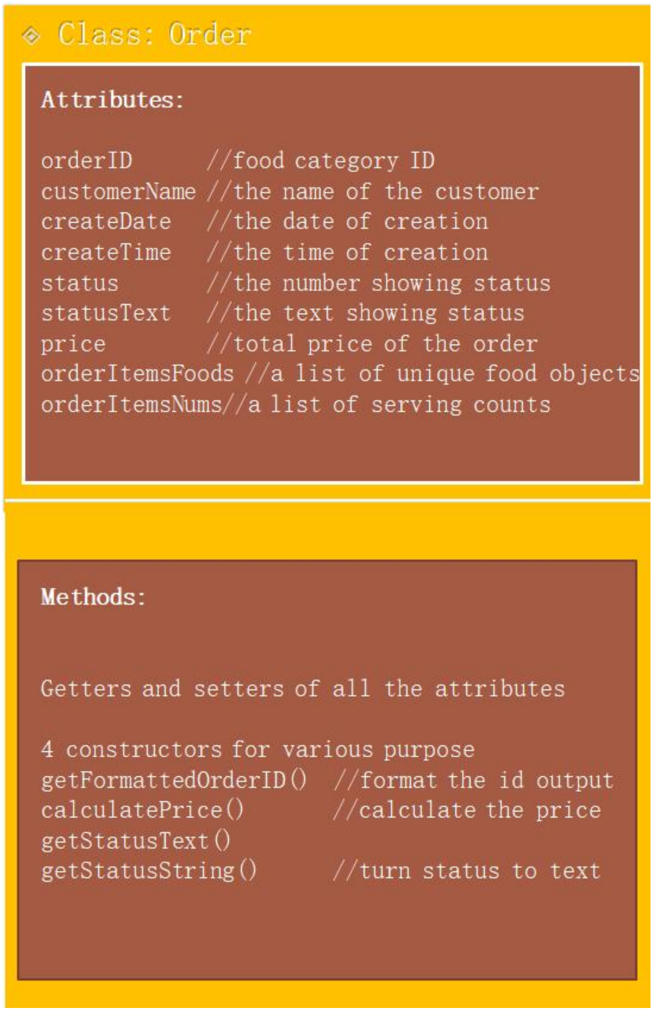
```
id;          //food category ID
name;        //the name of the food
description; //a brief description of the food
imageName;   //the name of the image file
price:       //price of the food
```

Methods:

Getters and setters of the 5 attributes

addFood(Food food) // add food to the data base, only Mr. Ping can do that

Figure 14. Food Class



```
◇ Class: Order

Attributes:

orderID       //food category ID
customerName //the name of the customer
createDate    //the date of creation
createTime    //the time of creation
status        //the number showing status
statusText    //the text showing status
price         //total price of the order
orderItemsFoods //a list of unique food objects
orderItemsNums//a list of serving counts

Methods:

Getters and setters of all the attributes

4 constructors for various purpose
getFormattedOrderID()  //format the id output
calculatePrice()       //calculate the price
getStatusText()
getStatusString()      //turn status to text
```

Figure 15. Order Class

Topic 2. Inheritance/Polymorphism

In the code, we can easily found definition as such:

List<Food> foodList = new ArrayList<>();

List<Helper> helpersList = new ArrayList<Helper>();

private ObservableList<Order> orderShowList = FXCollections.observableArrayList();

private List<Order> orderList = new ArrayList<Order>();

These 4 statements are very common in our project. They are showing the usage of polymorphism, where the variables of super classes refer to sub-class objects

Topic 3. Abstract Classes/Interfaces

We have 2 main Interfaces defined in our project

public interface IFoodController{...}
public interface IOderController {...}

The 2 main classes' database controllers are implementations of them

public class FoodController **implements** IFoodController{...}
public class OrderController **implements** IOderController{...}

Topic 4. Lists

The 2 main classes also correspond to 2 main Lists

private ObservableList<Food> addedDishesShow= FXCollections.*observableArrayList();*
private ObservableList<Order> allOrderShow= FXCollections.*observableArrayList*();
private List<Food> addedDishes = **new** ArrayList<Food>();
private List<Order> allOrder = **new** ArrayList<Order>();

We also incorporated List when we need to show a structured data type with javaFX
List<CheckBox> checkboxes = **new** ArrayList<CheckBox>();

Topic 5: Set/Map

Maps were incorporated twice in our project

1. In database controller

Map<Integer, Integer> map = **new** HashMap<>();
map = orderController.getFoodIdsbyOrderid(id);

2. In turning a foodList into an Order compatible unique foodNameList and foodNumList

```
Map<Integer, Integer> dishCount = new
HashMap<>();
for (Food food:addedDishes) {
int id = food.getid();
dishCount.put(id,dishCount.getOrDefault(id,
0)+1}
int[] dishNameList = new int[dishCount.size()];
int[] dishCountList = new int[dishCount.size()];
int foodNum = 0;
for (Map.Entry<Integer, Integer>
```

VIII. Conclusions and Future Work

Conclusion:

This project covers the 4 aspects of a small scale restaurant owner, who is also the cook of the restaurant:

1. Operating on orders
   a. view the order list
   b. prepare the received order
   c. pass the ready order to helper
2. Operating on dishes
   a. view the dish list
   b. create a new dish
   c. edit an existing dish
   d. delete a dish
3. Operating on users
   a. View User List
   b. Upcoming
4. Getting financial report

Also, 2 main function of the helper (waiter) of the restaurant is also covered:

1. Get order from the customer
2. Deliver order to the customer and inform the owner

It's simple and self-explanatory and fully featured to meet the fundamental need of a small scale restaurant owner.

Some problems of current version:

1. The owner and helper cannot get connected since it is not a web app yet. If there are 2 instances of the program running simultaneously, both users are using the same local database, causing an accessibility conflict to the database.

2. The users fields in the database are not used yet, and need more information.

3. The basic AnchorPane size is fixed. When the app window is stretched or maximized, it does not adjust.

Future Work:

1. Tidy up the code

Through this work, we learned that it is more important to communicate and to comply on the agreement between team members than writing large amount of code.
Lack of communication causes redundancy of the code and repeated work, leading to low efficiency and messy code.

2. Beautify the GUI

javaFX is great for GUI design. It's worth the time and energy to learn more about its details in order to design GUI that satisfies the aesthetics standards.

3. Deploy this project online, so that owners and helpers can access the database from anywhere simultaneously

4. Enrich the employee management functions for the owner

5. Enhance the owner – helper interaction functions, timely updating the order status

6. Complete owner's "get an order" function into "get an order online"

IX. Job Assignment

Below is a list of the team members' contributions to the project. This job's load is equally distributed between the 3 members:

- Shen Wang:
  1. general project design
  2. owner page GUI design
  3. owner page backend functionality
  4. database maintenance
  5. report & presentation composing
- Fei Cao:
  1. welcome & helper page GUI design
  2. welcome & helper page backend functionality
  3. background investigation
  4. report & presentation composing
- Douhao Ma
  1. general project structure design
  2. database design and maintenance
  3. model design

References

[1] Java. (n.d.). Home. Oracle. https://www.java.com/en/

[2] OpenJFX. (n.d.). Home. OpenJFX. https://openjfx.io/

[3] McKinsey & Company. (n.d.). How restaurants can thrive in the next normal. McKinsey & Company. https://www.mckinsey.com/industries/retail/our-

insights/how-restaurants-can-thrive-in-the-next-normal

[4] Toast. (n.d.). Restaurant POS & management system. Toast. https://pos.toasttab.com/

[5] Square. (n.d.). Point of Sale (POS) Systems. Square. https://squareup.com/us/en/point-of-sale

[6] TouchBistro. (n.d.). Restaurant POS & management system. TouchBistro. https://www.touchbistro.com/

[7] Lightspeed. (n.d.). Upserve by Lightspeed. Lightspeed. https://www.lightspeedhq.com/upserve/

[8] Lightspeed. (n.d.). ShopKeep by Lightspeed. Lightspeed. https://www.lightspeedhq.com/shopkeep/

[9] Reviews on Toast POS . (n.d.). by CROWDREVIEWS.COM, LLC https://www.crowdreviews.com/toast-pos/product-info

[10] Fandom. (n.d.). Mr. Ping. *Kung Fu Panda Wiki*. Retrieved [2023/4/18], from https://kungfupanda.fandom.com/wiki/Mr._Ping

[11] Fandom. (n.d.). Po. *Kung Fu Panda Wiki*. Retrieved [2023/4/18], from https://kungfupanda.fandom.com/wiki/Po

[12] GitUML. (n.d.). Retrieved from www.gituml.com

[13] Bing. (2023, April 18). How to populate the TableView in javafx with a List of object String typed attributes [Chat transcript]. Microsoft Bing search. https://www.bing.com/