Project Proposal

By - Team050 - Produce101

Team members: jingyul9, yutaoz3, yuhaof3, jundas2

Project Title

An intelligent restaurant ordering and recommendation database system

Project Summary:

Our project is an application for dish ordering and food recommendations in restaurants. Customers can perform various functions such as booking tables, searching food, ordering dishes, viewing recommendations, paying for the bill in discount, and so on. The highlight of our system is that it can help customers make decisions on dishes according to information like history purchases, current dishes, and dishes in order. The system also displays a vivid interface and provides functions like fuzzy search, and discount calculation to enhance customer experiences.

We plan to collect real-world data with a web crawler and generate some data we need. The system realizes the basic CRUD database operations and advanced features like procedure and trigger. We also put a UI mockup and our tentative work distribution in our proposal.

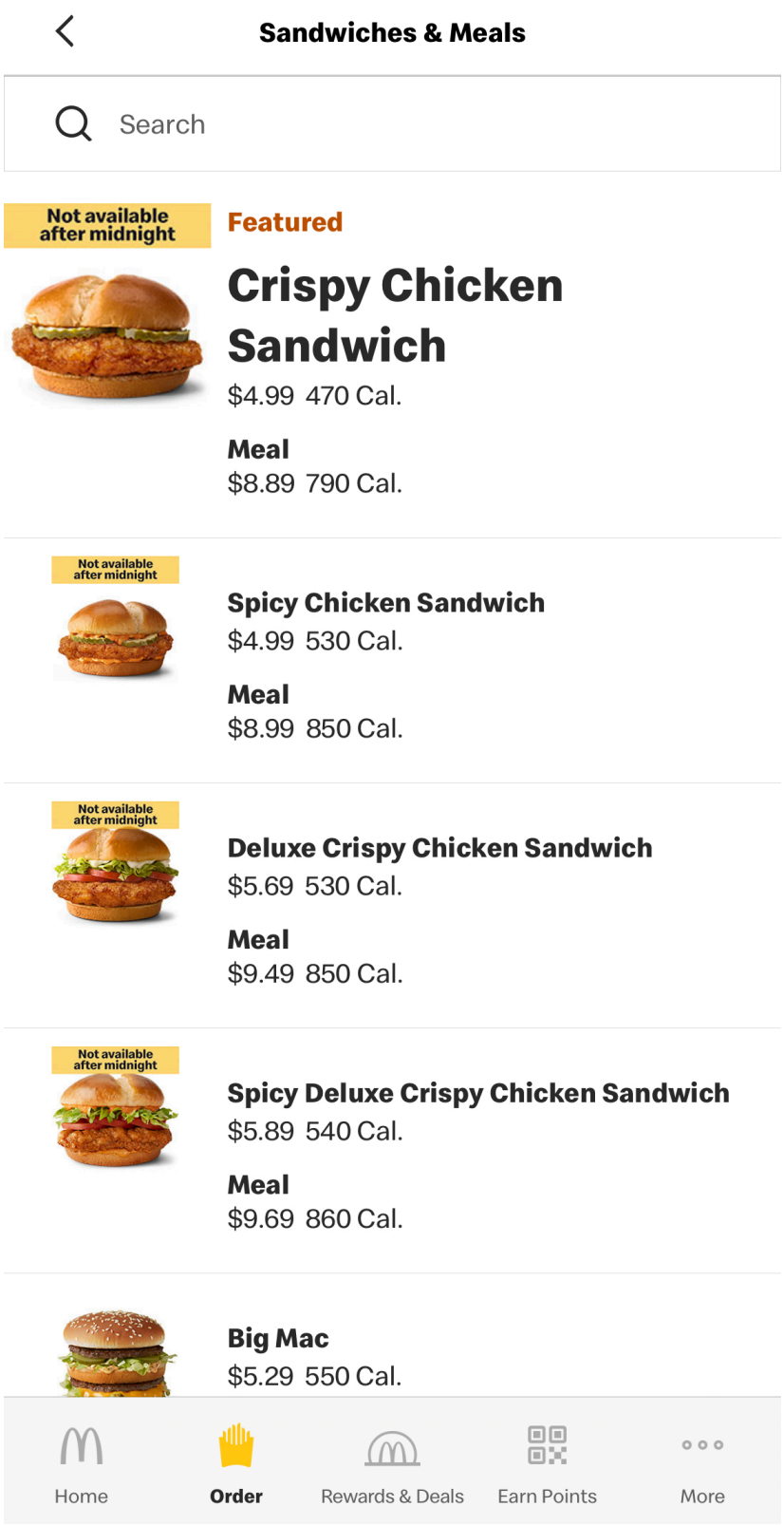
Description

Coming to restaurant for an enjoyable meal is a common activity for people. While customers used to order the food at the front desk, ordering automation is becoming more and more popular nowadays. Customers can scan the QR code on a certain table to view the menu and order the food they prefer. The ordering system significantly increases the efficiency of the restaurant and no long line is required anymore. However, the current system still has some defects. For example, some systems have no pictures of the food, and customers can only order the food by their name and imagine what they look like. Also, when customers have no preference for certain food and do not know how to decide, no recommendation generates. In addition, customers can only search for specific dish names instead of ingredients. Thus, we aim to design an intelligent restaurant ordering and recommendation system to solve these problems and improve customer experiences.

Usefulness

Our application is useful since it can help customers make decisions and accelerate the ordering process. When people walk into a restaurant for the first time, they are often unsure of what they want to order, and it is difficult for customers to find the right dish for their taste. Although there are certain ordering systems in some restaurants, they typically only offer the ability to make orders but lack the functionality to recommend food and help customers make decisions.

Below shows the ordering system of Mcdonald's. As we can see, in the sandwiches & meals bar, customers can only scroll up and down to see which to order but do not have a way to make a quick decision according to the user’s preference and history of purchases. Also, a fuzzy search for ingredients is not supported.



Our system is different in the way that it not only shows customers the style of each dish vividly (some ordering systems even do not have images for food) but also provides them with suggestions for ordering. For example, our proposed system can recommend matching dishes for customers based on their current tentative order and previous orders. To enhance the customer experience, our ordering system will deeply integrate the customer's information and data of previous orders (with the customer's consent) to provide certain discounts and better services. Our system would also support fuzzy search, recommending all dishes with a specific ingredient (e.g. beef, lettuce) when the customer wants to eat that ingredient.

Realness

We mainly have several schemas for the database system, which are food, order, table, ingredients, and customers. As for food, the information includes the price, image, calories, ingredients of each dish, and so on. The order table records customers' order information like time, total price, and food list. The table contains information like the table number, whether the table in the restaurant is full, table's order. The customer table just stores people's identity numbers, names, telephone, order histories, preferences, and so on.

The food will occupy the main storage space in our database. We will store plenty of real-world food information to make the system practical and authentic. We plan to use the Python web crawler to collect data from different restaurant websites (e.g. https://www.mcdonalds.com/us/en-us.html). For customers' orders, we will first generate some orders and then interact with the users for more orders, such as adding an order, modifying an order, and deleting an order (CRUD). For the information on dining tables, we will randomly generate a series of tables along with the number of customers that can sit at the tables. For the ingredients of the dishes, we will look for some real recipes and then record the amount of the ingredients required to make the dish, using a web crawler. As for the number of ingredients in stock, we will randomly generate some reasonable amounts and ensure that some dishes will be available for ordering. Lastly, for customers, the application will accept user registrations as well as store some random-generated customer information.

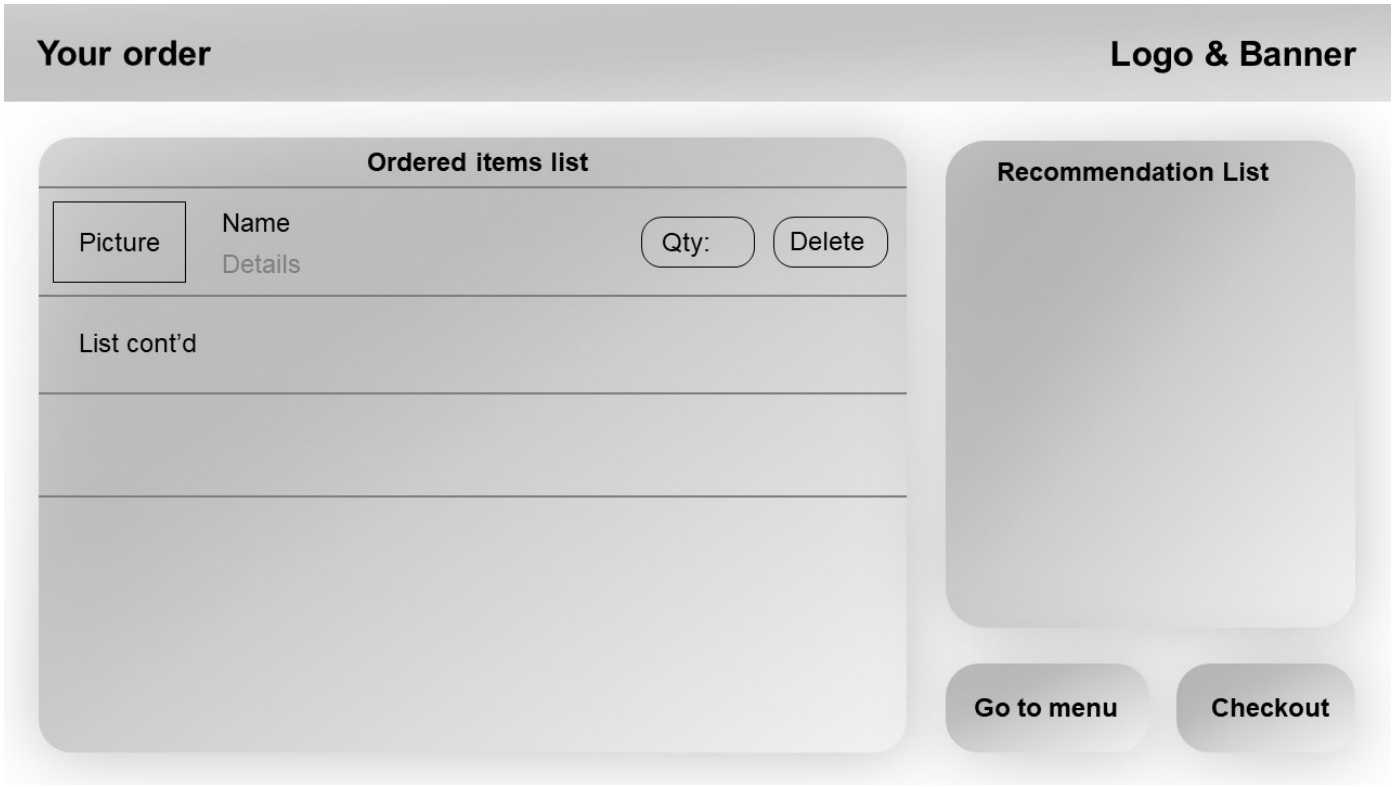
Functionality

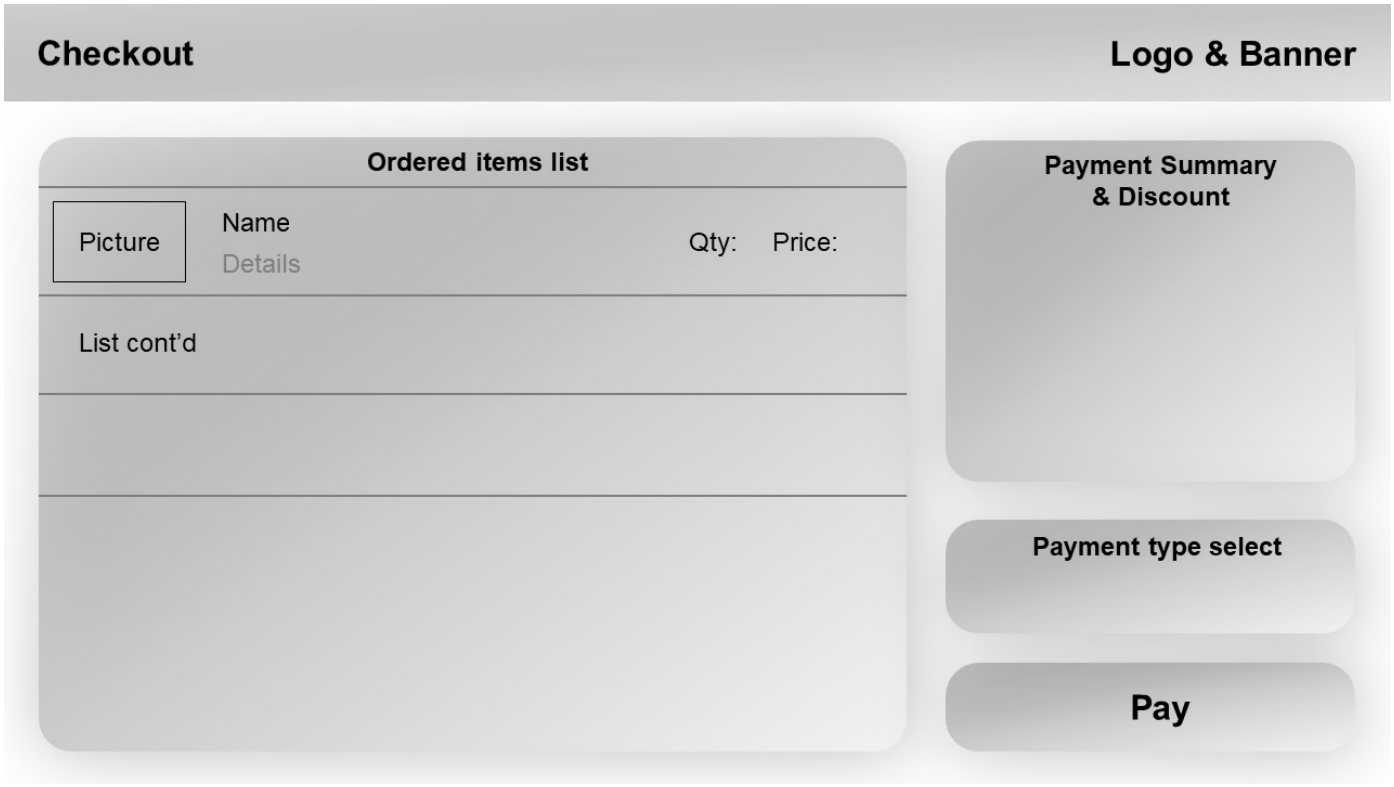
The food ordering application offers abundant functions for customers. We will illustrate the functionality in time order. Firstly, the customers need to select a table that could accommodate enough people. It will **update** the current dining table schema to full and attach it with the customers. They can also reserve seats beforehand. After their arrival, they can (fuzzy) **search** for the available dishes provided by the restaurant, which shows the basic **query** function of the database. During looking at the menu, the application will provide recommended dishes for a specific dish to help with customers' decisions. In this step, the advanced SQL **query** method should take effect and we would like to use a **procedure** for a recommendation. The system will generate the results with the information on the current dishes, dish list in order, customers' history preferences, and so on. When they decide what to eat, they can make orders that contain multiple dishes. This will **create** a new record in the order table. The order is not fixed but instead can be **updated** or **deleted** during the meal. For example, if the customers want more rice, then the amount of rice will be **updated** by adding some given value. The application also allows customers to cancel their dishes which has not been served, **deleting** the dish in the order. After the meal, the customers can pay the bill with either cash or their stored-value card. If it is a first-time customer, a new record will be **created**. During the transaction, the system will assign the corresponding discount for customers at different levels based on their total consumption amount. This process will be realized using a **trigger**. Then, their information regarding the history purchase and VIP score will be **updated** with the current meal.

In short, the core functionality is menu search and order dishes (CRUD). The advanced features are food recommendation (procedure) and discount calculation (trigger).

Low-fidelity UI mockup

Sample food videos as background.





Project work distribution

Our project will mainly involve 4 subtasks: Data preparation and database design, front-end development, back-end development (including SQL implementation), and testing.

To balance workload during development, each subtask will be assigned to two members to complete. The following is the tentative assignment:

Term	Member 1	Member 2
Data preparation and database design	jundas2	jingyul9
Front-end development	jingyul9	yuhaof3
Back-end development	yutaoz3	jundas2
Testing	yuhaof3	yutaoz3

Core functionality implementation distribution:

Term	Member 1	Member 2
Food Search	yutaoz3	jingyul9
Order Dish	yuhaof3	jundas2
Food recommendation	jingyul9	yuhaof3
Discount calculation	jundas2	yutaoz3