WWW Technologies and Applications 2025

Assignment 4 ─ AJAX/JSON/Web API

TA: Maggie (g13410050@ccu.edu.tw)

Deadline: 11:59pm, May 29, 2025

1. Attention

- The **backend** of this homework **must be implemented** in **Node.js Express framework**(https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm) or **PHP**.
- You **must** use **Ajax** to **request JSON data** from the backend server.

2. Introduction

In this assignment, you need to create a single-page web application that allows users to search for food nutrition information using the FatSecret Platform API (https://platform.fatsecret.com/platform-api) and display the results. The webpage consists of four sections: a search bar, a results table, a nutrition information pie chart, and a favorite list. You must use the FatSecret Platform API and Ajax to complete this task. In addition, all the implementation details and the requirements will be explained in the following sections.

Regarding quick start guides for the FatSecret Platform API (For more information, please go to the following website: https://platform.fatsecret.com/docs/guides):

Step 1. Registering a free FatSecret Platform API Account ➢ Go to the FatSecret Platform API: https://platform.fatsecret.com/ ➢ Click on "Start Free" to create a developer account
Step 2. Obtaining API key ➢ After logging in, you can retrieve your API key from the "API key" page: https://platform.fatsecret.com/my-account/api-key ➢ FatSecret API supports both OAuth 1.0 and OAuth 2.0 authentication methods. You can choose one to use. If you use OAuth 2.0, **please add the IP range 0.0.0.0/0 or 140.123.0.0/16 to the allowlist** (https://platform.fatsecret.com/my-account/ip-restrictions) to ensure the TA can execute your assignment.
Step 3. Testing API ➢ If you are using OAuth 1.0, you can refer to this program to test the API: https://colab.research.google.com/drive/1nCYewurs8-0WUCftaXO1nNBxnCoKGluo?usp=sharing ➢ If you use OAuth 2.0, you can refer to this program to test the API: https://colab.research.google.com/drive/1wysEjhKfgeJf67ac6T1FL1aGmm6gYVFN?usp=sharing

3. High level description

3.1 Search bar

When a user initially opens your webpage, the page is illustrated in Figure 1. It must include two components: a navigation bar and a search bar. The navigation bar must include a navigation bar title and a "Favorite List" button. The user can enter the name of the food they want to search for in the search box and click the "Search" button to retrieve the results. Please note that an error message must be displayed when the required fields are empty to remind the user that this field cannot be left blank, as shown in Figure 2.

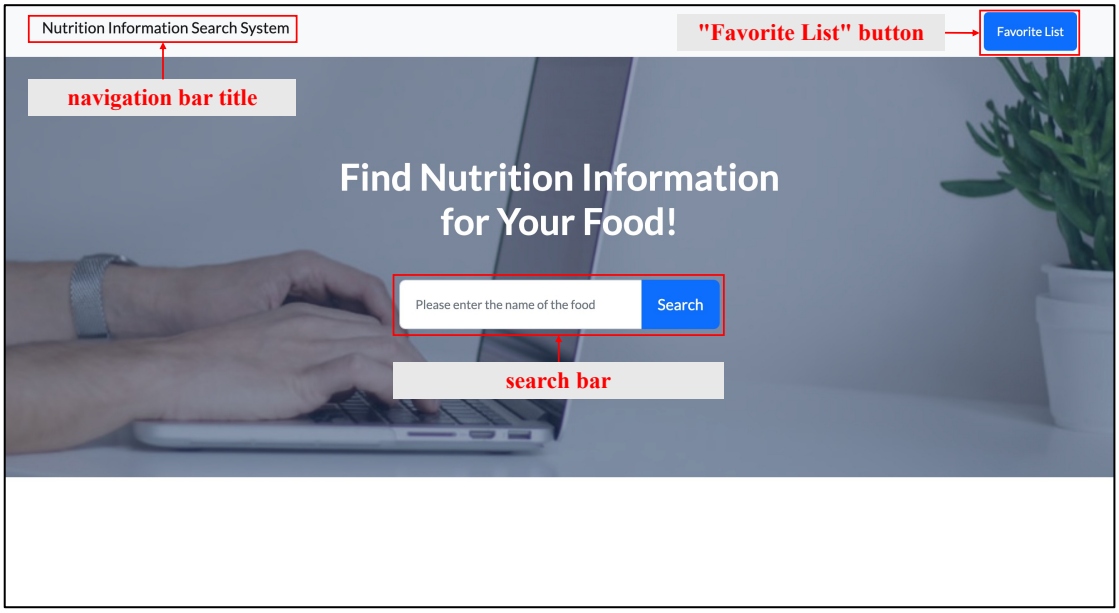


Figure 1

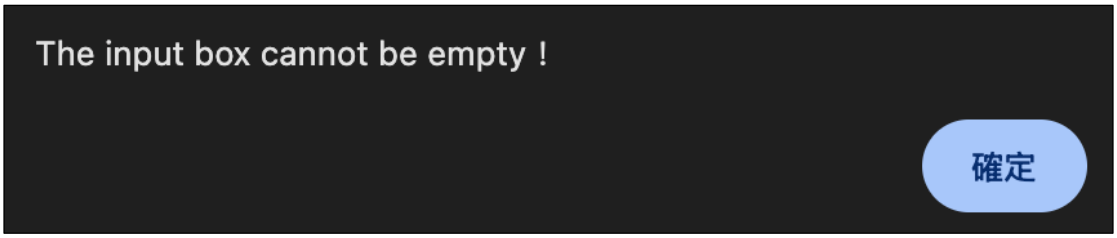


Figure 2

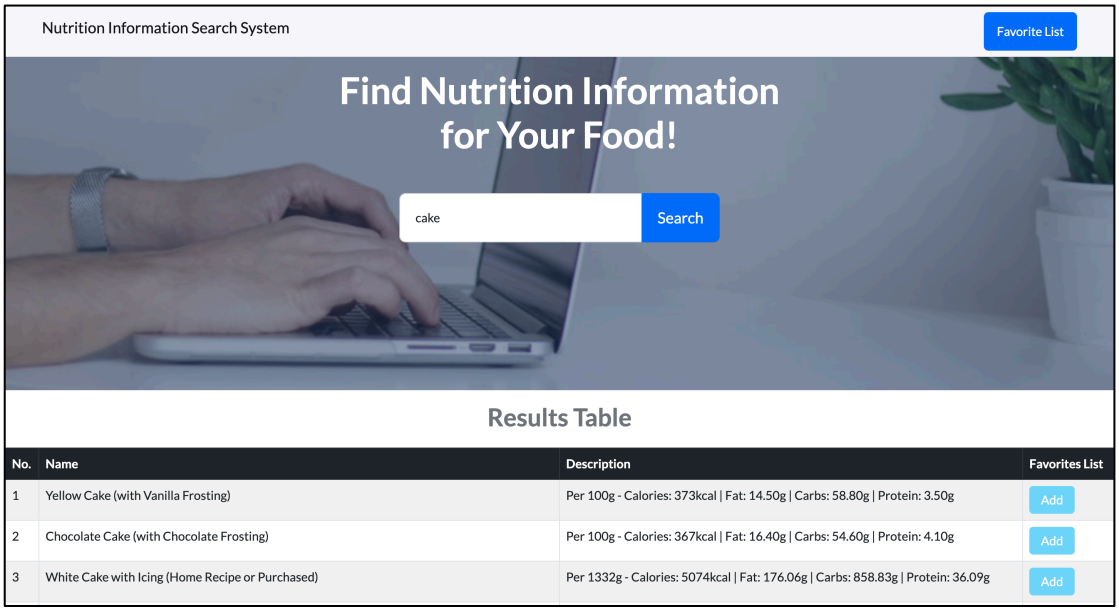
📖 **Grading Criteria**:

- (2%) Navigation bar
  - ➢ The navigation bar must include a navigation bar title and a "Favorite List" button.
- (2%) Input and search
  - ➢ Users can enter the food name in the search box and click the "Search" button.
- (1%) Error message
  - ➢ When the input box is empty, an error message should be displayed.

3.2 Results table

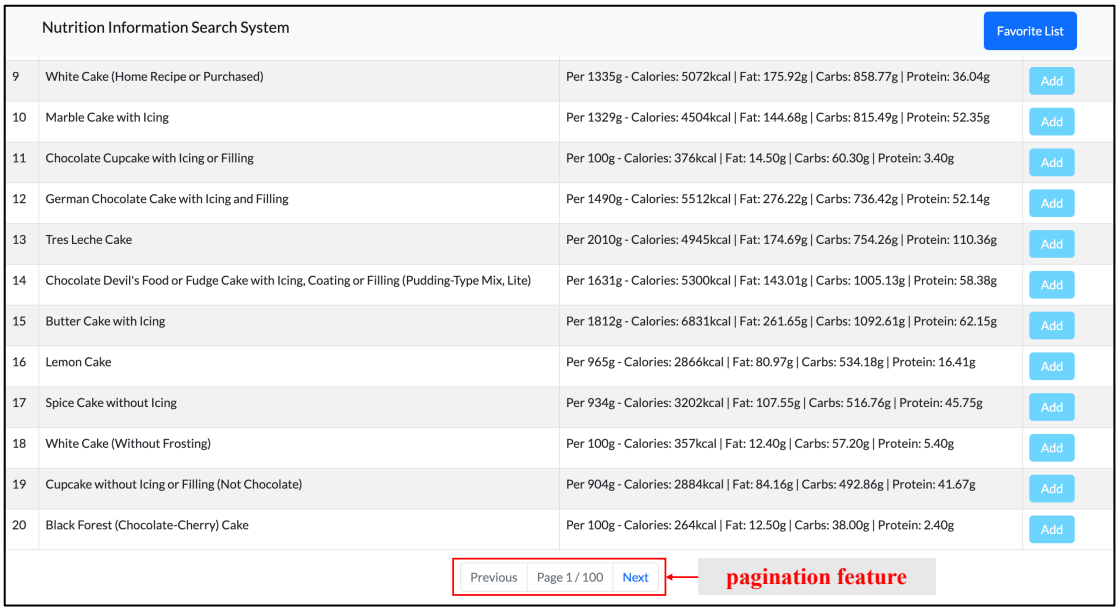
When the user clicks the "Search" button, you must display the food information in a tabular format below the search bar. AJAX should be used to prevent the page from refreshing. A sample output is shown in Figure 3. For example, when the user enters "cake" to query, the results table will expand below the search bar. As shown in Figure 4, below the results table, there is a pagination feature that is used to navigate to the next set of foods. There are a maximum of 20 foods on a page. The user can navigate to a different page directly using the "Previous" and "Next" buttons.

(Hint: You will need to use the **food.search** method of the FatSecret Platform API. For more information, please go to the following website: https://platform.fatsecret.com/docs/v1/foods.search)



No.	Name	Description	Favorites List
1	Yellow Cake (with Vanilla Frosting)	Per 100g - Calories: 373kcal | Fat: 14.50g | Carbs: 58.80g | Protein: 3.50g	Add
2	Chocolate Cake (with Chocolate Frosting)	Per 100g - Calories: 367kcal | Fat: 16.40g | Carbs: 54.60g | Protein: 4.10g	Add
3	White Cake with Icing (Home Recipe or Purchased)	Per 1332g - Calories: 5074kcal | Fat: 176.06g | Carbs: 858.83g | Protein: 36.09g	Add

Figure 3



9	White Cake (Home Recipe or Purchased)	Per 1335g - Calories: 5072kcal | Fat: 175.92g | Carbs: 858.77g | Protein: 36.04g	Add
10	Marble Cake with Icing	Per 1329g - Calories: 4504kcal | Fat: 144.68g | Carbs: 815.49g | Protein: 52.35g	Add
11	Chocolate Cupcake with Icing or Filling	Per 100g - Calories: 376kcal | Fat: 14.50g | Carbs: 60.30g | Protein: 3.40g	Add
12	German Chocolate Cake with Icing and Filling	Per 1490g - Calories: 5512kcal | Fat: 276.22g | Carbs: 736.42g | Protein: 52.14g	Add
13	Tres Leche Cake	Per 2010g - Calories: 4945kcal | Fat: 174.69g | Carbs: 754.26g | Protein: 110.36g	Add
14	Chocolate Devil's Food or Fudge Cake with Icing, Coating or Filling (Pudding-Type Mix, Lite)	Per 1631g - Calories: 5300kcal | Fat: 143.01g | Carbs: 1005.13g | Protein: 58.38g	Add
15	Butter Cake with Icing	Per 1812g - Calories: 6831kcal | Fat: 261.65g | Carbs: 1092.61g | Protein: 62.15g	Add
16	Lemon Cake	Per 965g - Calories: 2866kcal | Fat: 80.97g | Carbs: 534.18g | Protein: 16.41g	Add
17	Spice Cake without Icing	Per 934g - Calories: 3202kcal | Fat: 107.55g | Carbs: 516.76g | Protein: 45.75g	Add
18	White Cake (Without Frosting)	Per 100g - Calories: 357kcal | Fat: 12.40g | Carbs: 57.20g | Protein: 5.40g	Add
19	Cupcake without Icing or Filling (Not Chocolate)	Per 904g - Calories: 2884kcal | Fat: 84.16g | Carbs: 492.86g | Protein: 41.67g	Add
20	Black Forest (Chocolate-Cherry) Cake	Per 100g - Calories: 264kcal | Fat: 12.50g | Carbs: 38.00g | Protein: 2.40g	Add

Figure 4

📖 **Grading Criteria**:

- (20%) Results table
  - ☑ Using Ajax to correctly display No., Name, and Description returned by the API response. (15%)
    - ➢ When the search result contains at least one record, you need to map the data extracted from the API results to the columns to render the HTML result table as described in Table 1.

Table 1

HTML Table Column	Details
No.	The value of the **"No."** starts at 1 and increments sequentially.
Name	The value of the **"Name"** is "food_name" inside the "food" object in the JSON data.
Description	The value of the "**Description**" is "food_description" inside the "food" object in the JSON data.

    - ➢ For example, a request URL searching for "cake" with a maximum of 1 result on the first page would be:

https://platform.fatsecret.com/rest/server.api?method=foods.search&search_expression=cake&format=json&max_results=1&page_number=0

You can retrieve the **food_name** and **food_description** from the JSON data (see Figure 5) and display them in the table under the **Name** and **Description** columns, respectively.

```
food: {
    food_description: 'Per 100g - Calories: 373kcal | Fat: 14.50g | Carbs: 58.80g | Protein: 3.50g',
    food_id: '38869',
    food_name: 'Yellow Cake (with Vanilla Frosting)',
    food_type: 'Generic',
    food_url: 'https://www.fatsecret.com/calories-nutrition/usda/yellow-cake-(with-vanilla-frosting)'
}
```

Figure 5

  - ☑ "Add" button (5%)
- (20%) Pagination feature
  - ☑ A maximum of 20 foods on a page. (5%)
  - ☑ Display the current and the total page number (see Figure 6). (5%)

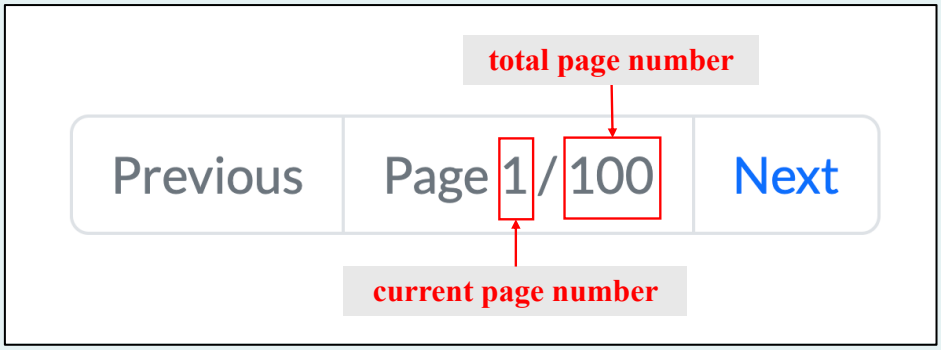


Figure 6

☑ The user can navigate to a different page directly by using the "Previous" and "Next" buttons. (10%)

➢ For example, page 1 contains the 1st to 20th food items. When the user clicks the "Next" button to go to page 2, they can see the 21st to 40th food items.

3.3 Nutrition information pie chart

When the user clicks on a table row, your website must display the food details (fat, carbohydrate, and protein) in a pie chart format. AJAX should be used to prevent the page from refreshing. A sample output is shown in Figure 7. When the user clicks on the first food item "Yellow Cake (with Vanilla Frosting)" in the results table, the page will render and display the nutrition pie chart for "Yellow Cake (with Vanilla Frosting)." More specifically, when nutritional information for the food exists, you need to map the data extracted from the API results to the corresponding categories to render an HTML pie chart, as shown in Table 2.

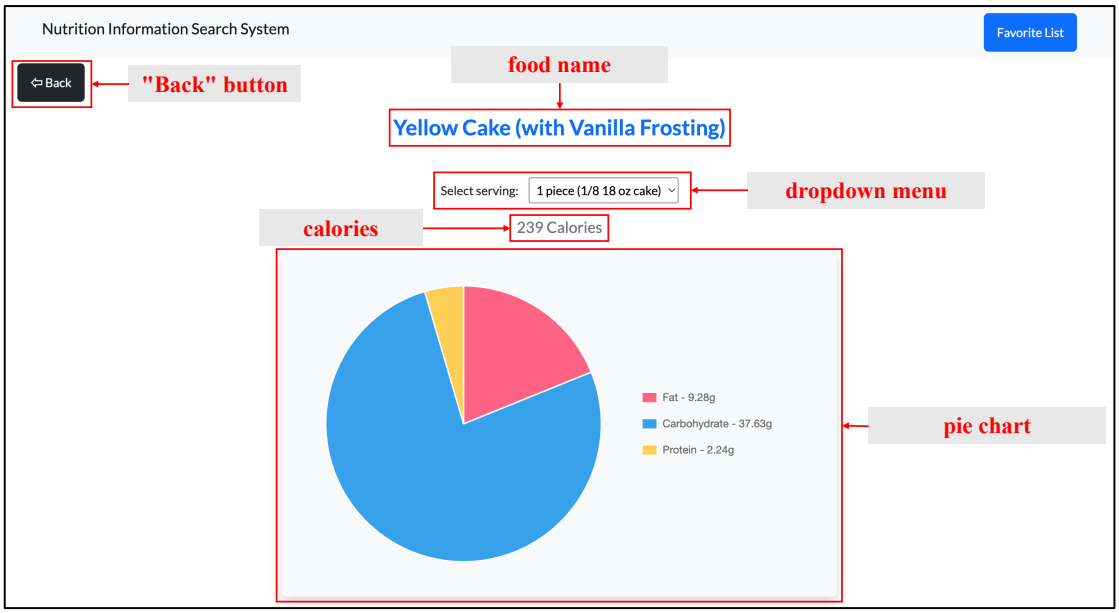


Figure 7

Table 2

HTML Pie Chart Slice	Details
Fat	The value of the "**Fat**" is "fat" inside the "serving" object in the JSON data.
Carbohydrate	The value of the "**Carbohydrate**" is " carbohydrate " inside the "serving" object in the JSON data.
Protein	The value of the "**Protein**" is "protein" inside the "serving" object in JSON data.

For example, a request URL to search for detailed information about the food with ID 33869 would be:

```
https://platform.fatsecret.com/rest/server.api?method=food.get&food_id=38869&format=json
```

You can retrieve the **calories, fat, carbohydrate, protein,** and **serving_description** from the responded JSON data (see Figure 8) to complete this part. (Hint: You will need to use the **food.get** method of the FatSecret Platform API. For more information, please go to the following website: https://platform.fatsecret.com/docs/v4/food.get)

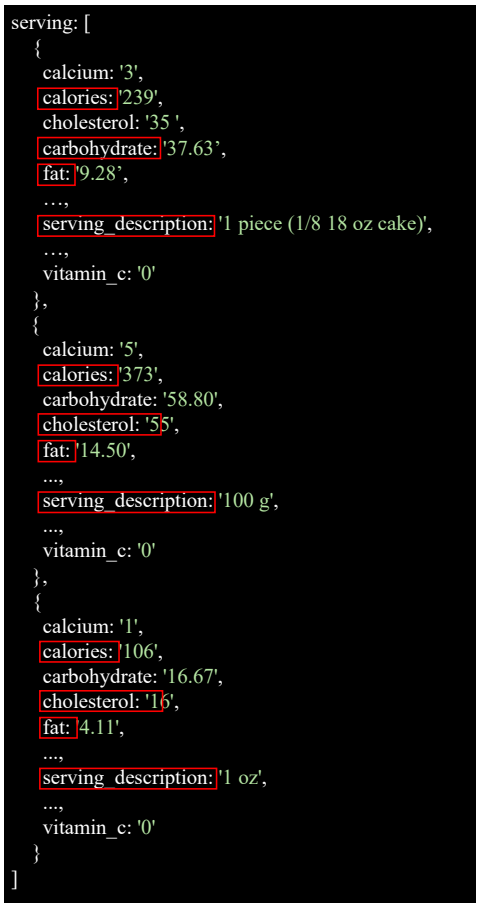


Figure 8

📖 **Grading Criteria**:

- (5%) Food name
  - ➢ Correctly display the name corresponding to the food item clicked by the user.
- (5%) Dropdown menu
  - ➢ You can retrieve the serving_description from the JSON data and update the options of the dropdown menu (see Figure 9).

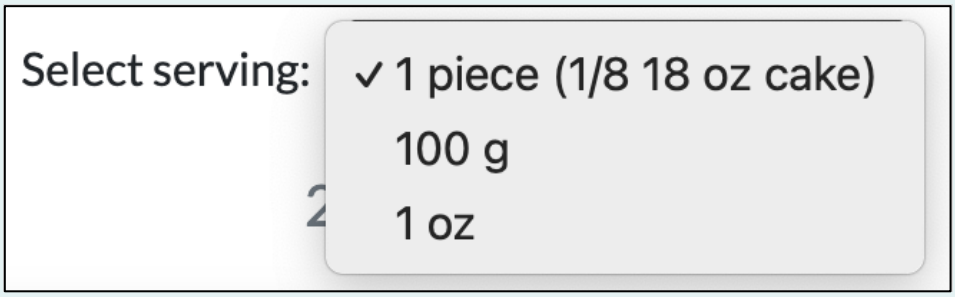


Figure 9

- (5%) Calories
  - ➢ When the user selects a different serving size, the displayed calorie value should update accordingly, as shown in Figure 10 and Figure 11.

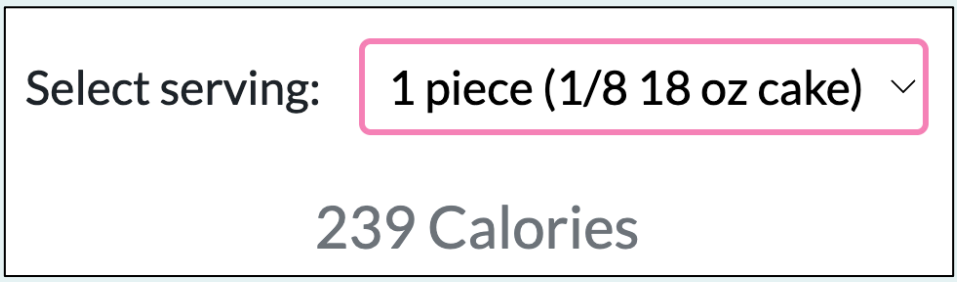


Figure 10

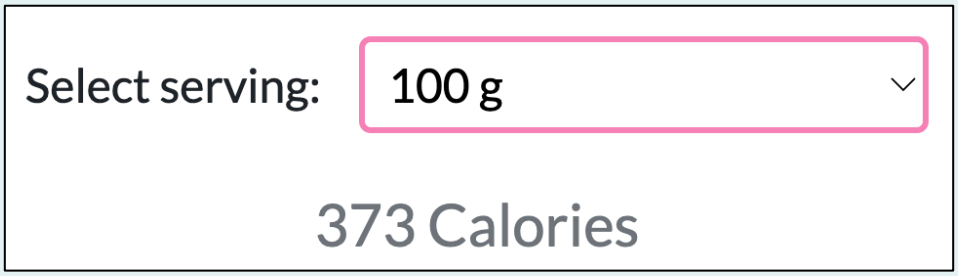


Figure 11

- (15%) Pie chart (Hint: You can use Chart.js to implement pie charts)
  - ☑ Using Ajax, correctly display the value of each slice by the API response. (10%)
    - ➢ For example, if the user selects 100g of Yellow Cake (with Vanilla Frosting), the displayed values will be Fat: 14.5g, Carbohydrate: 58.8g, and Protein: 3.5g (see Figure 12).

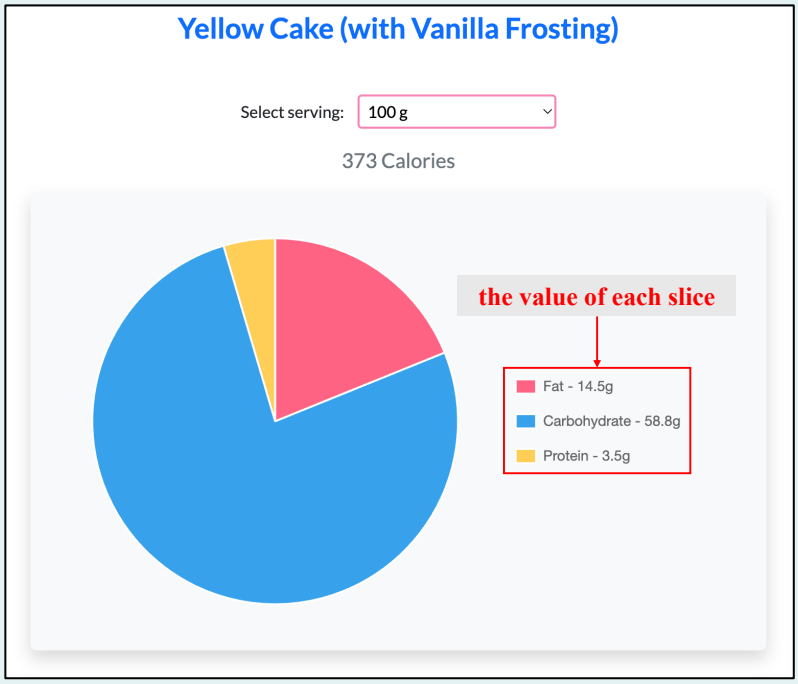


Figure 12

☑ Hover effect (5%)

➢ When the mouse hovers over the pie chart, it should display the value of the corresponding nutrient (see Figure 13).

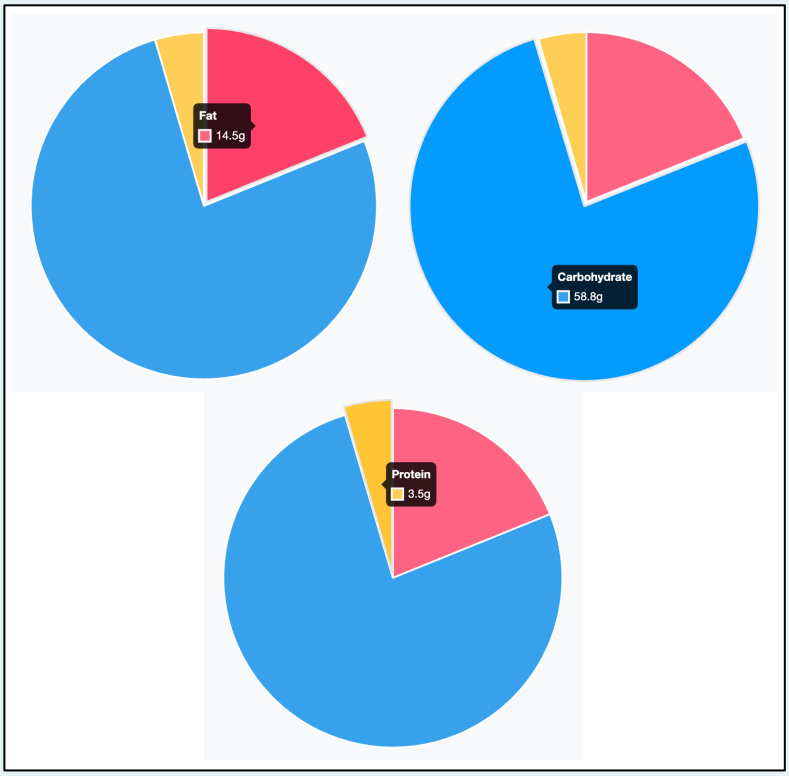


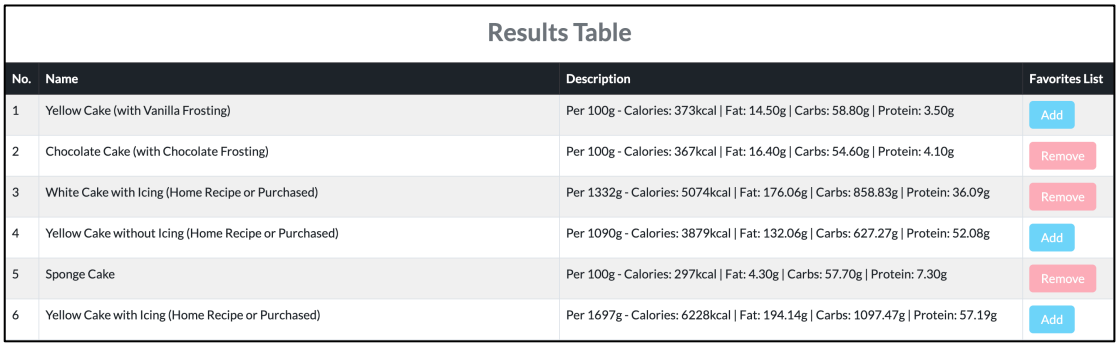
Figure 13

- (3%) "Back" button

➢ When the users click the "Back" button, they can return to the results table.

3.4 Favorite list

In the results table, clicking the "Add" button toggles the selected food in the favorites list, as shown in Figure 14. (Hint: You can use local storage to implement this function.) When the user clicks the "Favorite List" button, your webpage must display the favorite list in a tabular format, and AJAX should be used to prevent the page from refreshing, as shown in Figure 15.



Results Table

No.	Name	Description	Favorites List
1	Yellow Cake (with Vanilla Frosting)	Per 100g - Calories: 373kcal | Fat: 14.50g | Carbs: 58.80g | Protein: 3.50g	Add
2	Chocolate Cake (with Chocolate Frosting)	Per 100g - Calories: 367kcal | Fat: 16.40g | Carbs: 54.60g | Protein: 4.10g	Remove
3	White Cake with Icing (Home Recipe or Purchased)	Per 1332g - Calories: 5074kcal | Fat: 176.06g | Carbs: 858.83g | Protein: 36.09g	Remove
4	Yellow Cake without Icing (Home Recipe or Purchased)	Per 1090g - Calories: 3879kcal | Fat: 132.06g | Carbs: 627.27g | Protein: 52.08g	Add
5	Sponge Cake	Per 100g - Calories: 297kcal | Fat: 4.30g | Carbs: 57.70g | Protein: 7.30g	Remove
6	Yellow Cake with Icing (Home Recipe or Purchased)	Per 1697g - Calories: 6228kcal | Fat: 194.14g | Carbs: 1097.47g | Protein: 57.19g	Add

Figure 14

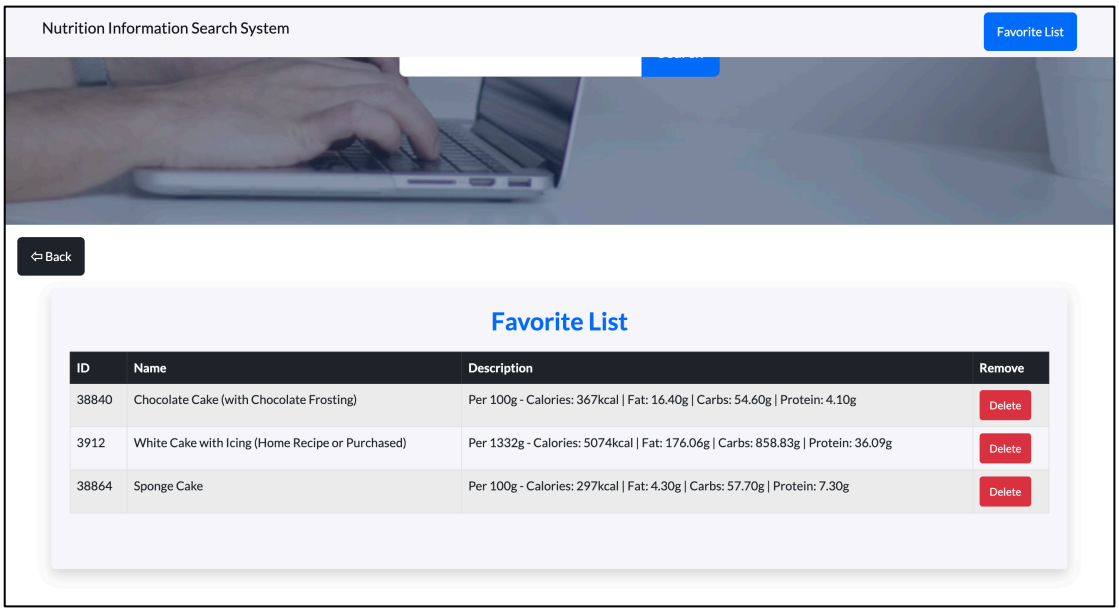


Figure 15

📖 **Grading Criteria**:

- (15%) Favorite list
  - ➢ Using Ajax to correctly display ID, Name, and Description.
- (3%) "Add" button and "Remove" button in the results table
  - ➢ When the user clicks the "Add" button, the food item is added to the favorite list and the "Add" button is replaced by a "Remove" button; when the user clicks the "Remove" button, the food item is removed from the Favorite list and the "Remove" button is changed back to an "Add" button. More specifically, foods that have been added to the favorite list should

have a pink "Remove" button, while foods that are not in the favorite list should have a blue "Add" button.

- (3%) "Delete" button in the favorite list
  - ➢ When the user clicks the "Delete" button in the favorite list, the food item is immediately removed from the favorite list without requiring a page refresh. Subsequently, when switching back to the results table, an "Add" button should be appear for the food item.
- (1%) "Back" button
  - ➢ When the user clicks the "Back" button, it returns to the results table.
- (5% bonus) Nutrition table
  - ➢ Using Ajax to correctly display serving description and calories.
  - ➢ When the user clicks a food item in the favorite list, the corresponding nutrition table is displayed below the list. For example, when the user clicks on "Chocolate Cake (with Chocolate Frosting)" in the favorite list, the nutrition table will be displayed as shown in Figure 16.

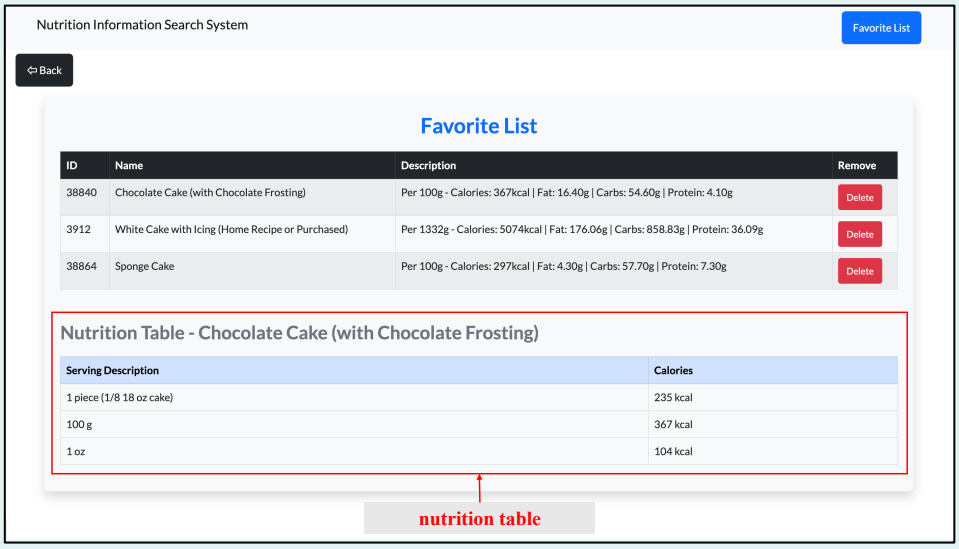


Figure 16

- (5% bonus) Progress bar
  - ➢ A progress bar should be displayed while waiting for the data to be retrieved (see Figure 17).

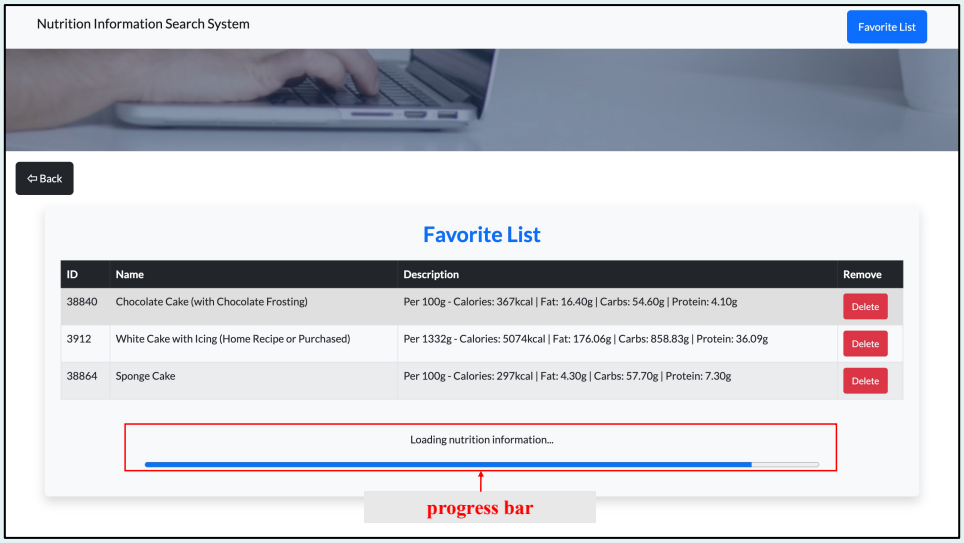


Figure 17

4. How and what to submit

4.1 How to submit your assignment

The procedures for submitting your assignment can be found on our document website (http://wwwdmplus2025.csie.io:81/).

4.2 What should be submitted

1. You should hand in your assignment to Gitlab and deploy your website on your server.
2. You should record a demo video to show each function in your assignment and upload it to YouTube.

Note: If your project uses Node.js, it's recommended to use **PM2** to manage your backend process. This ensures your backend server stays active even after you close the terminal. You may refer to the instructions below for a basic setup. For more advanced configuration and options, please refer to the official PM2 documentation (https://pm2.keymetrics.io/docs/usage/quick-start/).

Step 1: SSH into your server `ssh root@wwweb2025.csie.io -p [your_ssh_port]`
Step 2: Install PM2 globally `npm install pm2 -g`
Step 3: Change into your project directory `cd /var/www/html/hw4`

Step 4: Start your backend with PM2 and save the process
`pm2 start app.js --name hw4`
`pm2 save`

5. Grading policy

Graders will test your homework only on **Google Chrome in 1920*1080**. Homework submitted late will be accepted for up to 7 days after the due date and will receive an automatic 30% penalty. Homework submitted more than 7 days after the due date will not be accepted. Only one final submission (either one on-time or one late submission) will be accepted. Your assignment will not be graded without a demo video. You should explain all the functions of your assignment in the demo video. The link to your demo video should be submitted to eCourse2.

The TA(s) will mark and give points according to the following grading policy: Different from homework 1 and homework 2, your website does not need to look the same as the examples. The TA(s) will focus on the required functions only.

Function	Description	Weight (%)
search bar	Navigation bar must include a navigation bar title and a "Favorite List" button	2%
	Users can enter the food name in the search box and click the "Search" button	2%
	When the input box is empty, an error message should be displayed	1%
result table	Using Ajax to correctly display No., Name, and Description returned by the API response	15%
	Implement an "Add" button	5%
	Implement a pagination feature (A maximum of 20 foods on a page)	5%
	Implement a pagination feature (Display the current page number and total page number)	5%
	Implement a pagination feature (Users can navigate to a different page directly by using "Previous" and "Next" buttons and view different food data)	10%
pie chart	Correctly display the name corresponding to the food item clicked by the user	5%
	Render the dropdown menu options using serving_description from the JSON data	5%
	When the user selects a different serving size, the displayed calorie value should update accordingly	5%
	Using Ajax to correctly display the value of each slice by the API response	10%

	When the mouse hovers over the pie chart, it should display the value of the corresponding nutrient information	5%
	When the user clicks the "Back" button, they can return to the results table.	3%
favorite list	Using Ajax to correctly display ID, Name, and Description	15%
	Implement "Add" button and "Remove" button	3%
	Implement "Delete" button (Update favorite list, "Add" button, and "Remove" button to reflect their toggled state immediately)	3%
	When the user clicks the "Back" button, they can return to the results table	1%
	(bonus) Using Ajax to correctly display serving descriptions and calories for the selected food	5%
	(bonus) A progress bar should be displayed while waiting for the data to be retrieved	5%