

(75%) 113-2 JavaScript 期末考 1 -- 斷網考試

2025-06-12, at E201, from 18:20~21:00

Note:

1. 本期末考分兩部分，斷網考試及開放網路考試。
2. 請不要發揮同學愛，作弊雙方除了本次考試 0 分外，平常分數另扣 20 分，情節嚴重者會送校。
3. iClass 上請繳交 md1_14.pdf, final_md1.zip, final_client_14.zip, final_server_14.zip，壓縮前請將 node_modules 刪除
4. 請直接將答案寫在 md1_14.md 上，老師出題及圖片放在 md1_htc.pdf 上，請依照老師所給的圖片來實作並標註
5. 跟小考相關的檔案及目錄名稱有 xx 時，必須要改成學號後 2 碼，沒有修改時，會視違犯情況扣分。
6. 每一張圖片要有機房左側背景，圖片上要有你的學號(或後兩碼)，圖片標註要跟老師所標註的類似。違者會依情節扣分。
7. 請自評分數，將每一題的？填入分數，沒有填者，不會批改，以 0 分計算。

Your (Name, ID): (林亮廷, 913410014)

Part 1: 斷網考試 (75%)

- P1 (15%): 15 分
- P2 (15%): 15 分
- P3 (15%): 15 分
- P4 (10%): 10 分
- P5 (10%): 10 分
- P6 (10%): 10 分

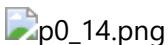
總分: 75 分

Part 2: 開放考試 (25%)

- P7 (15%): 15 分
- P8 (10%): 10 分

Note: 本次考試前端及後端主要以 **menu theme** 為主來實作，但考題涵蓋了上課 **product_14 demo** 及小考 **shop_14** 的部分

檔案及目錄結構如下圖



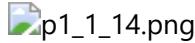
(15%) P1: 請用本地端的 pgAdmin 來建立 final_js_14 資料庫及 table menu_14

請以你的學號最後一個數字 (D) · 來決定要實作哪一個分類的頁面。

分類 : breakfast: mid = 1, 4, 7, 10 lunch: mid = 2, 5, 8 dessert: mid = 3, 6, 9

D=1,6 => breakfast(1,4), dessert(6,9) D=2,7 => lunch(2,5), breakfast(7,10) D=3,8 => dessert(3,6), lunch(8,2),
D=4,9 => breakfast(4,7), dessert(9,3) D=5,0 => lunch(5,8), breakfast(10,1)

本次考試 · 老師題目顯示以 D=4,9 breakfast(4,7), dessert(9,3) 四筆資料 為主來解說 。



Your Answer

=> SQL code

參考上圖 · 老師定義 SQL 的欄位名稱及資料型別。將指定的 menu 資訊 · 寫成 SQL 指令 · 執行時可以新增資料。SQL 指令要包含 CREATE TABLE, INSERT 指令

```

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer final_js_14/postgres@PostgreSQL 16* X
Query Query History
1 v CREATE TABLE menu_14 (
2   mid INT NOT NULL PRIMARY KEY,
3   title varchar(255),
4   price real,
5   category varchar(255),
6   local_img text,
7   remote_img text,
8   description text
9 );
10
11 v INSERT INTO menu_14 (mid, title, category, price, local_img, remote_img, description) VALUES
12 (1, 'buttermilk pancakes', 'breakfast', 15.99, '/menu/images/item-1.jpeg', 'https://example.com/remote/item-1.jpeg', 'I''m baby woke mlksh
13 (2, 'diner double', 'lunch', 13.99, '/menu/images/item-2.jpeg', 'https://example.com/remote/item-2.jpeg', 'vaporware iPhone mumblecore sel
14 (3, 'godzilla milkshake', 'shakes', 6.99, '/menu/images/item-3.jpeg', 'https://example.com/remote/item-3.jpeg', 'ombucha chillwave fanny p
15 (4, 'country delight', 'breakfast', 20.99, '/menu/images/item-4.jpeg', 'https://example.com/remote/item-4.jpeg', 'Shabby chic keffiyeh neu
16 (5, 'egg attack', 'lunch', 22.99, '/menu/images/item-5.jpeg', 'https://example.com/remote/item-5.jpeg', 'franzen vegan pabst bicycle right
17 (6, 'oreo dream', 'shakes', 18.99, '/menu/images/item-6.jpeg', 'https://example.com/remote/item-6.jpeg', 'Portland chicharrones ethical ed
18 (7, 'bacon overflow', 'breakfast', 8.99, '/menu/images/item-7.jpeg', 'https://example.com/remote/item-7.jpeg', 'carry jianbing normcore fr
19 (8, 'american classic', 'lunch', 12.99, '/menu/images/item-8.jpeg', 'https://example.com/remote/item-8.jpeg', 'on it tumblr kickstarter th
20 (9, 'quarantine buddy', 'shakes', 16.99, '/menu/images/item-9.jpeg', 'https://example.com/remote/item-9.jpeg', 'skateboard fam synth authen
21 (10, 'bison steak', 'dinner', 22.99, '/menu/images/item-10.jpeg', 'https://example.com/remote/item-10.jpeg', 'skateboard fam synth authent

```

Data Output Messages Notifications

INSERT 0 10

Query returned successfully in 71 msec.

Total rows: 0 of 0 Query complete 00:00:00.071 Ln 21, Col 215

=> pgAdmin 顯示 database final_js_14, table menu_14 的四筆資料

The screenshot shows the pgAdmin 4 interface. On the left is the Object Explorer tree, which includes Catalogs, Event Triggers, Extensions, Foreign Data Wrappers, Languages, Publications, Schemas (1), public (with Aggregates, Collations, Domains, FTS Configurations, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and Tables (2)), menu_14, product_14, Trigger Functions, Types, Views, Subscriptions, postgres, Login/Group Roles, and Tablespaces. The Tables (2) node under public is expanded, showing menu_14, product_14, Trigger Functions, Types, Views, and Subscriptions.

The main window shows a query editor with the following SQL code:

```
1 SELECT * FROM public.menu_14
2 ORDER BY mid ASC
```

The results pane displays a table with 10 rows of data from the menu_14 table. The columns are: mid [PK] integer, title character varying (255), price real, category character varying (255), local_img text, remote_img text, and description text. The data includes various food items like buttermilk pancakes, diner double, godzilla milkshake, etc., with their descriptions and image URLs.

mid [PK] integer	title character varying (255)	price real	category character varying (255)	local_img text	remote_img text	description text
1	buttermilk pancakes	15.99	breakfast	/menu/images/item-1.jpeg	https://example.com/remote/item-1.jpeg	I'm baby woke mlkshk wolf bitters live-edge blue vaporware iPhone mumblecore selvage raw denim
2	diner double	13.99	lunch	/menu/images/item-2.jpeg	https://example.com/remote/item-2.jpeg	Shabby chic keffiyeh neutra snackwave pork belly
3	godzilla milkshake	6.99	shakes	/menu/images/item-3.jpeg	https://example.com/remote/item-3.jpeg	ombucha chillwave fanny pack 3 wolf moon streetwear
4	country delight	20.99	breakfast	/menu/images/item-4.jpeg	https://example.com/remote/item-4.jpeg	Portland chicharrones ethical edison bulb, palo s
5	egg attack	22.99	lunch	/menu/images/item-5.jpeg	https://example.com/remote/item-5.jpeg	franzen vegan pabst bicycle rights kickstarter
6	oreo dream	18.99	shakes	/menu/images/item-6.jpeg	https://example.com/remote/item-6.jpeg	skateboard fam synth authentic semiotics. Live-
7	bacon overflow	8.99	breakfast	/menu/images/item-7.jpeg	https://example.com/remote/item-7.jpeg	carry jianbing normcore freegan. Viral single-origin
8	american classic	12.99	lunch	/menu/images/item-8.jpeg	https://example.com/remote/item-8.jpeg	on it tumblr kickstarter thundercats migas every
9	quarantine buddy	16.99	shakes	/menu/images/item-9.jpeg	https://example.com/remote/item-9.jpeg	skateboard fam synth authentic semiotics. Live-
10	bison steak	22.99	dinner	/menu/images/item-10.jpeg	https://example.com/remote/item-10.jpeg	skateboard fam synth authentic semiotics. Live-

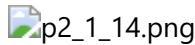
Total rows: 10 of 10 Query complete 00:00:00.276 Ln 1, Col 1

(15%) P2: 實作路由 /menu_14

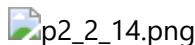
menu_14 theme 已經放在 /public/menu 目錄下 靜態頁面已經可以透過 /menu_14/static 顯現，有六筆

顯示頁面 womens_static.ejs 放在 views/shop_14 下 如果你是實作 hats，靜態頁面請用 hats_static.ejs

=> Chrome 顯示/menu_14/static 靜態頁面

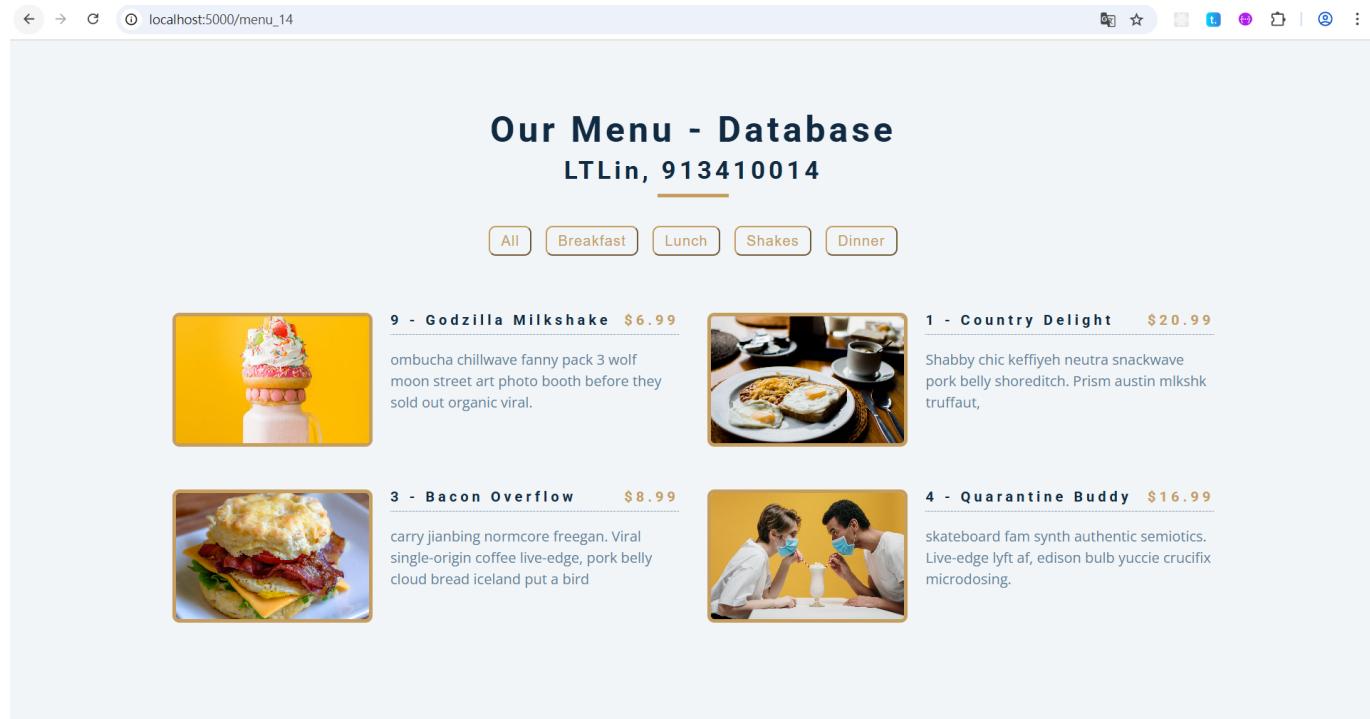


=> 將 menu_static.ejs 複製一份到 menu_14.ejs，從資料庫 menu_14 資料表中取得 4 筆資料顯示



Your Answer

=> 將 menu_static.ejs 複製一份到 menu_14.ejs，從資料庫 menu_14 資料表中取得 4 筆資料顯示



=> server_14.js 有關 code 之重點截圖

```

JS server_14.js ✘ <x> menu_static.ejs ✘ menu_14.ejs ✘ JS MenuRouter_14.js
final_server_14 > JS server_14.js > ...
9 import MenuRouter from './routes/MenuRouter_14.js'; // Add this import
10
11 const app_14 = express();
12
13 app_14.use(cors());
14 app_14.use(logger('dev'));
15 app_14.use(express.static('public'));
16 app_14.set('view engine', 'ejs');
17
18 // menu routes
19
20 app_14.use('/menu_14/static', (req, res, next) => {
21   res.render('menu_14/menu_static', {
22     title: 'Our Menu - Static',
23     name: 'LTLin',
24     id: '913410014',
25   });
26 });
27
28 // menu routes
29 app_14.use('/menu_14', MenuRouter); // Add this line
30 app_14.use('/menu_14/static', (req, res, next) => {
31   res.render('menu_14/menu_static', {
32     title: 'Our Menu - Static',
33     name: 'LTLin',
34     id: '913410014',
35   });
36 });
37
38 // shop routes
39 app_14.use('/api/shop_14', apiShopRouter);
40 app_14.use('/shop_14', ShopRouter);
41 app_14.use('/shop_14/static', (req, res, next) => {
42   res.render('shop_14/womens_static', {
43     title: 'Womens -- Static Page',
44     name: 'LTLin',
45     id: '913410014',
46   });
47 });

```

=> routes/MenuRouter_14.js 之重點截圖

```

JS server_14.js | <x> menu_static.ejs | <x> menu_14.ejs | JS MenuRouter_14.js X
final_server_14 > routes > JS MenuRouter_14.js > pool > password
1 import express from 'express';
2 import pkg from 'pg';
3 const { Pool } = pkg;
4
5 const router = express.Router();
6
7 const pool = new Pool({
8   user: 'postgres',
9   host: 'localhost',
10  database: 'final_js_14',
11  password: '0000', // Replace with your actual password
12  port: 5432,
13});
14
15 router.get('/', async (req, res) => {
16   try {
17     // Query to get 4 items based on student ID ending digit 4
18     // D=4,9 => breakfast(4,7), dessert(9,3)
19     const query =
20       'SELECT * FROM menu_14 WHERE mid IN (4, 7, 9, 3) ORDER BY mid';
21     const result = await pool.query(query);
22
23     res.render('menu_14/menu_14', {
24       title: 'Our Menu - Database',
25       name: 'LTLin',
26       id: '913410014',
27       menuitems: result.rows,
28     });
29   } catch (err) {
30     console.error('Database error:', err);
31     res.status(500).send('Database error');
32   }
33 });
34
35 export default router;
36

```

=> menu_14.ejs 有關 code 之重點截圖

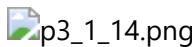
```

JS server_14.js | <x> menu_static.ejs | <x> menu_14.ejs X | JS MenuRouter_14.js
final_server_14 > views > menu_14 > <x> menu_14.ejs > body > section.menu > div.section-center > ? > article.menu-item > div.item-info > header > h4 > ?
1 <html lang="en">
2   <body>
3     <section class="menu">
4       <div class="btn-container">
5         <button type="button" class="filter-btn" data-id="all">all</button>
6         <button type="button" class="filter-btn" data-id="breakfast">
7           breakfast</button>
8         <button type="button" class="filter-btn" data-id="lunch">lunch</button>
9         <button type="button" class="filter-btn" data-id="shakes">
10          shakes</button>
11         <button type="button" class="filter-btn" data-id="dinner">
12           dinner</button>
13       </div>
14       <!-- menu items -->
15
16       <div class="section-center">
17         <% menuitems.forEach(function(item, index) { %>
18           <article class="menu-item">
19             "
22               class="photo"
23             />
24             <div class="item-info">
25               <header>
26                 <h4><%- id[index] %>- <%- item.title %></h4>
27                 <h4 class="price">$<%- item.price %></h4>
28               </header>
29               <p class="item-text"><%- item.description %}</p>
30             </div>
31           </article>
32         <% }); %>
33       </div>
34     </section>
35   </body>
36 </html>
37

```

(15%) P3: 實作 路由 /api/menu_14

=> Chrome 顯示



Your Answer

=> Chrome 顯示

```

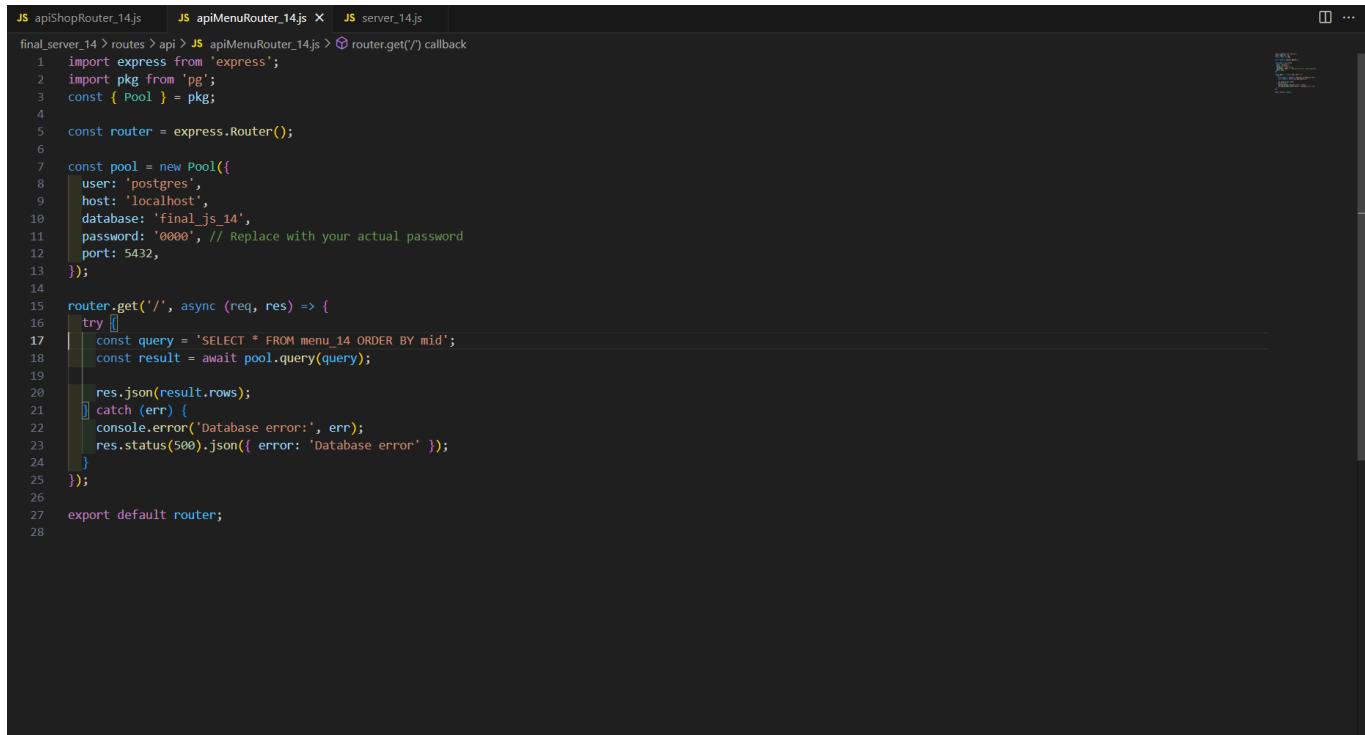
[{"mid": 1, "title": "buttermilk pancakes", "price": 15.99, "category": "breakfast", "local_img": "/menu/images/item-1.jpeg", "remote_img": "https://example.com/remote/item-1.jpeg", "description": "I'm baby woke mlkshk wolf bitters live-edge blue bottle, hammock freegan copper mug whatever cold-pressed"}, {"mid": 2, "title": "diner double", "price": 13.99, "category": "lunch", "local_img": "/menu/images/item-2.jpeg", "remote_img": "https://example.com/remote/item-2.jpeg", "description": "vaporware iPhone mumblecore selvage raw denim slow-carb leggings gochujang helvetica man braid jianbing. Marfa thundercats"}, {"mid": 3, "title": "godzilla milkshake", "price": 6.99, "category": "shakes", "local_img": "/menu/images/item-3.jpeg", "remote_img": "https://example.com/remote/item-3.jpeg", "description": "ombucha chillwave fanny pack 3 wolf moon street art photo booth before they sold out organic viral."}]
  
```

=> server_14.js code 重點

```

final_server_14 > JS server_14.js > ...
1 import express from 'express';
2 import cors from 'cors';
3 import logger from 'morgan';
4
5 import apiProductRouter from './routes/api/apiProductRouter_14.js';
6 import ProductRouter from './routes/ProductRouter_14.js';
7 import apiShopRouter from './routes/api/apiShopRouter_14.js';
8 import ShopRouter from './routes/ShopRouter_14.js';
9 import MenuRouter from './routes/MenuRouter_14.js'; // Add this import
10 import apiMenuRouter from './routes/api/apiMenuRouter_14.js'; // Add this import
11
12 const app_14 = express();
13
14 app_14.use(cors());
15 app_14.use(logger('dev'));
16 app_14.use(express.static('public'));
17 app_14.set('view engine', 'ejs');
18
19 // menu routes
20 app_14.use('/api/menu_14', apiMenuRouter); // Add this line
21
22 app_14.use('/menu_14/static', (req, res, next) => {
23   res.render('menu_14/menu_static', {
24     title: 'Our Menu - static',
25     name: 'LTlin',
26     id: '913410014',
27   });
28 });
29
30 // menu routes
31 app_14.use('/menu_14', MenuRouter); // Add this line
32 app_14.use('/menu_14/static', (req, res, next) => {
33   res.render('menu_14/menu_static', {
34     title: 'Our Menu - Static',
35     name: 'LTlin',
36     id: '913410014',
37   });
38 });
39
  
```

=> apiMenuRouter_14.js code 重點



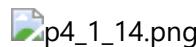
```
JS apiShopRouter_14.js JS apiMenuRouter_14.js X JS server_14.js
final_server_14 > routes > api > JS apiMenuRouter_14.js > router.get('/') callback
1 import express from 'express';
2 import pkg from 'pg';
3 const { Pool } = pkg;
4
5 const router = express.Router();
6
7 const pool = new Pool({
8   user: 'postgres',
9   host: 'localhost',
10  database: 'final_js_14',
11  password: '0000', // Replace with your actual password
12  port: 5432,
13 });
14
15 router.get('/', async (req, res) => {
16   try {
17     const query = 'SELECT * FROM menu_14 ORDER BY mid';
18     const result = await pool.query(query);
19
20     res.json(result.rows);
21   } catch (err) {
22     console.error('Database error:', err);
23     res.status(500).json({ error: 'Database error' });
24   }
25 });
26
27 export default router;
28
```

(10%) P4: 請以老師提供的框架，寫前端 JavaScript 程式

請以老師提供的 final_14.html，實作 menu_api_nodejs_14.js，透過 api 呼叫後端 Node.js server 端程式，可以取得 json 資料，套用在 final_14.html 頁面中

在 menu_api_nodejs_14.js 中儲存 menu 的變數名稱為 menu_14，不可以改變

=> Chrome 顯示



Your Answer

=> Chrome 顯示

localhost:5000/menu/final_14.html

Menus -- API From Node.Js

LTLin, 913410014

All Breakfast Lunch Shakes Dinner

	Buttermilk Pancakes	\$ 15.99
I'm baby woke milkshk wolf bitters live-edge blue bottle, hammock freegan copper mug whatever cold-pressed		
	Diner Double	\$ 13.99
vaporware iPhone mumblecore selvage raw denim slow-carb leggings gochujang helvetica man braid jianbing. Marfa thundercats		
	Godzilla Milkshake	\$ 6.99

Console What's new

```
menu_api_nodejs_14.js:12
  ↵ (10) [{}]
  ↵ 0: {mid: 1, title: 'buttermilk pancakes', price: 15.99, category: 'breakfast'}
  ↵ 1: {mid: 2, title: 'diner double', price: 13.99, category: 'lunch', local: true}
  ↵ 2: {mid: 3, title: 'godzilla milkshake', price: 6.99, category: 'shakes'}
  ↵ 3: {mid: 4, title: 'country delight', price: 20.99, category: 'breakfast'}
  ↵ 4: {mid: 5, title: 'egg attack', price: 22.99, category: 'lunch', local: true}
  ↵ 5: {mid: 6, title: 'oreo dream', price: 18.99, category: 'shakes', local: true}
  ↵ 6: {mid: 7, title: 'bacon overflow', price: 8.99, category: 'breakfast'}
  ↵ 7: {mid: 8, title: 'american classic', price: 12.99, category: 'lunch'}
  ↵ 8: {mid: 9, title: 'quarantine buddy', price: 16.99, category: 'shakes'}
  ↵ 9: {mid: 10, title: 'bison steak', price: 22.99, category: 'dinner', local: true}
  ↵ length: 10
  ↵ [[Prototype]]: Array(0)
```

=> 相關 js code 顯示

final_client_14 > menu > JS menu_api_nodejs_14.js > fetchMenuData

```
final_client_14 > menu > JS menu_api_nodejs_14.js > fetchMenuData
1 const sectionCenter = document.querySelector('.section-center');
2 const btnContainer = document.querySelector('.btn-container');
3
4 let menu_14 = [];
5
6 // 從 API 獲取資料
7 async function fetchMenuData() {
8   try {
9     const response = await fetch('http://localhost:5000/api/menu_14');
10    const data = await response.json();
11    menu_14 = data;
12    console.log(menu_14);
13    // 頁面載入時顯示所有項目
14    displayMenuItems(menu_14);
15    displayMenuButtons();
16  } catch (error) {
17    console.error('Error fetching menu data:', error);
18  }
19}
20
21 // 顯示菜單項目
22 function displayMenuItems(menuItems) {
23   let displayMenu = menuItems.map(function (item) {
24     return `article class="menu-item">
25       <img src=${item.local_img} alt=${item.title} class="photo" />
26       <div class="item-info">
27         <header>
28           <h4>${item.title}</h4>
29           <h4 class="price">$${item.price}</h4>
30         </header>
31         <p class="item-text">
32           ${item.description}
33         </p>
34       </div>
35     </article>`;
36   });
37   displayMenu = displayMenu.join('');
38   sectionCenter.innerHTML = displayMenu;
39 }
```

相關 js code 顯示

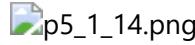
(10%) P5: 請針對 P4 題目，加入 filter 篩選功能

請參考在 server 端的 menu theme，該範例是可以執行 menu 之篩選功能的。其 filter 按鈕會根據所讀入的 menu 之 category 資訊自動建立，但本題是固定的，因此要做調整。

在 menu_api_nodejs_14.js 中儲存 menu 的變數名稱為 menu_14，不可以改變

=> Chrome 顯示

按照順序，點選按鈕 Breakfast, Lunch, Shakes, Dessert，然後用 console.log 顯示筆數，出現的筆數請展開



Your Answer

=> Chrome 顯示

按照順序，點選按鈕 Breakfast, Lunch, Shakes, Dessert，然後用 `console.log` 顯示筆數，出現的筆數請展開

localhost:5000/menu/final_14.html

Menus -- API From Node.Js

LTLin, 913410014

All Breakfast Lunch Shakes Dinner

	Godzilla Milkshake	\$ 6.99
	ombucha chillwave fanny pack 3 wolf moon street art photo booth before they sold out organic viral.	
	Oreo Dream	\$ 18.99
	Portland chicharrones ethical edison bulb, palo santo craft beer chia heirloom iPhone everyday	
	Quarantine Buddy	\$ 16.99

localhost:5000/menu/final_14.html

Menus -- API From Node.Js

LTLin, 913410014

All Breakfast Lunch Shakes Dinner

	Bison Steak	\$ 22.99
	skateboard fam synth authentic semiotics. Live-edge lyft af, edison bulb yuccie crucifix microdosing.	

Console

```

menu_api_nodejs_14.js:12
(10) [{}, {}, {}, {}, {}, {}, {}, {}, {}, {}]
breakfast: 3
menu_api_nodejs_14.js:78
menu_api_nodejs_14.js:79
(3) [{}, {}, {}]
> 0: {mid: 1, title: 'buttermilk pancakes', price: 15.99, category: 'breakfast'}
> 1: {mid: 4, title: 'country delight', price: 20.99, category: 'breakfast'}
> 2: {mid: 7, title: 'bacon overflow', price: 8.99, category: 'breakfast', length: 3}
> [[Prototype]]: Array(0)
lunch: 3
menu_api_nodejs_14.js:78
menu_api_nodejs_14.js:79
(3) [{}, {}, {}]
> 0: {mid: 2, title: 'diner double', price: 13.99, category: 'lunch', loca
> 1: {mid: 5, title: 'egg attack', price: 22.99, category: 'lunch', local
> 2: {mid: 8, title: 'american classic', price: 12.99, category: 'lunch', length: 3
> [[Prototype]]: Array(0)
shakes: 3
menu_api_nodejs_14.js:78
menu_api_nodejs_14.js:79
(3) [{}, {}, {}]
> 0: {mid: 3, title: 'godzilla milkshake', price: 6.99, category: 'shakes'}
> 1: {mid: 6, title: 'oreo dream', price: 18.99, category: 'shakes', loca
> 2: {mid: 9, title: 'quarantine buddy', price: 16.99, category: 'shakes', length: 3
> [[Prototype]]: Array(0)
dinner: 1
menu_api_nodejs_14.js:78
menu_api_nodejs_14.js:79
(1) [{}, {}]
> 0: {mid: 10, title: 'bison steak', price: 22.99, category: 'dinner', loca
length: 1
> [[Prototype]]: Array(0)

```

=> 相關 js code 顯示

```

JS menu_api_nodejs_14.js X final_14.html JS server_14.js
final_client_14 > menu > JS menu_api_nodejs_14.js > displayMenuButtons > filterBtns.forEach() callback > btn.addEventListener('click') callback
42 function displayMenuButtons() {
52 const categoryBtns = categories
53 .map(function (category) {
54   })
55 .join('');
56
57 btnContainer.innerHTML = categoryBtns;
58 const filterBtns = btnContainer.querySelectorAll('.filter-btn');
59
60 filterBtns.forEach(function (btn) {
61   btn.addEventListener('click', function (e) {
62     const category = e.currentTarget.dataset.id;
63     const menuCategory = menu_14.filter(function (menuItem) {
64       if (menuItem.category === category) {
65         return menuItem;
66       }
67     });
68
69     if (category === 'all') {
70       displayMenuItems(menu_14);
71       console.log(`${category}: ${menu_14.length} 筆`);
72       console.log(menu_14);
73     } else {
74       displayMenuItems(menuCategory);
75       console.log(`${category}: ${menuCategory.length} 筆`);
76       console.log(menuCategory);
77     }
78   });
79 })
80
81 });
82 }
83
84 // 頁面載入時執行
85 window.addEventListener('DOMContentLoaded', function () {
86   fetchData();
87 });
88
89

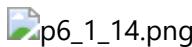
```

相關 js code 顯示

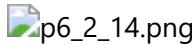
(10%) P6: 在前端顯示 product, shop 資訊

請直接以老師給的 code 匯入資料庫，不需要再改學號後兩碼，從資料庫取得 json 資料，顯示在前端，唯一要改的姓名，學號

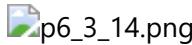
=> 資料庫顯示 shop_14 資料表資料



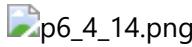
=> Chrome 顯示 shop 資訊



=> 資料庫顯示 product_14 資料表資料



=> Chrome 顯示 product 資訊



Your Answer

請直接以老師給的 code 匯入資料庫，不需要再改學號後兩碼，從資料庫取得 json 資料，顯示在前端，唯一要改的是姓名，學號

=> 資料庫顯示 shop_14 資料表資料

The screenshot shows the pgAdmin 4 interface. In the Object Explorer on the left, under the 'public' schema, there are tables: menu_14, product_xx, and shop_xx. The shop_xx table is selected. The main pane displays a query result table with 7 rows of data:

pid [PK] integer	title character varying (255)	category_id integer	price real	local_img text	remote_img text
1	Blue Tanktop	4	25	/shop/img/womens/blue-tank.png	https://i.ibb.co/7CQVJNm/blue-tank.png
2	Floral Blouse	4	20	/shop/img/womens/floral-blouse.png	https://i.ibb.co/4W2DGKm/floral-blouse.png
3	Floral Dress	4	80	/shop/img/womens/floral-skirt.png	https://i.ibb.co/KV18Ysr/floral-skirt.png
4	Red Dots Dress	4	80	/shop/img/womens/red-polka-dot-dress.png	https://i.ibb.co/N3BN1bh/red-polka-dot-dress.png
5	Striped Sweater	4	45	/shop/img/womens/striped-sweater.png	https://i.ibb.co/KmSKMBH/striped-sweater.png
6	Yellow Track Suite	4	135	/shop/img/womens/yellow-track-suit.png	https://i.ibb.co/v1cvwNf/yellow-track-suit.png
7	White Blouse	4	20	/shop/img/womens/white-vest.png	https://i.ibb.co/qBcrsJg/white-vest.png

=> Chrome 顯示 shop 資訊

The screenshot shows a Chrome browser window at localhost:5000/shop/quiz2_p5_14.html. The page title is "Womens -- API From Node.js" and the address bar shows "localhost:5000/shop/quiz2_p5_14.html". The page content displays a grid of seven product images with their titles and IDs:

- Blue Tanktop (pid: 1)
- 25 Floral Blouse (pid: 2)
- 20 Floral Dress (pid: 3)
- 80 Red Dots Dress (pid: 4)
- Stripped Sweater (pid: 5)
- 45 Yellow Track Suite (pid: 6)
- 135 White Blouse (pid: 7)

On the right side of the browser, the developer tools' Console tab is open, showing the JSON response from the API call:

```

shop_14 json from shop_api_nodejs_14.js:11
Node.js server
  ↴ (7) [{}]
    ↴ 0: {pid: 1, title: 'Blue Tanktop', cate
    ↴ 1: {pid: 2, title: 'Floral Blouse', cate
    ↴ 2: {pid: 3, title: 'Floral Dress', cate
    ↴ 3: {pid: 4, title: 'Red Dots Dress', cat
    ↴ 4: {pid: 5, title: 'Striped Sweater', c
    ↴ 5: {pid: 6, title: 'Yellow Track Suite',
    ↴ 6: {pid: 7, title: 'White Blouse', cate
      length: 7
    > [[Prototype]]: Array(0)
  
```

=> 資料庫顯示 product_14 資料表資料

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

Servers (1) PostgreSQL 16

Databases (2) final_js_14

Casts Catalogs Event Triggers Extensions Foreign Data Wrappers Languages Publications Schemas (1) public

Aggregates Collations Domains FTS Configurations FTS Dictionaries FTS Parsers FTS Templates Foreign Tables Functions Materialized Views Operators Procedures Sequences Tables (3) menu_14 product_xx shop_xx Triggers Functions

Query Query History

```
1 SELECT * FROM public.product_xx
2 ORDER BY id ASC LIMIT 100
3
```

Data Output Messages Notifications

	id [PK] integer	title character varying (255)	price real	category character varying (255)	img text	remote_img text
1	1	Emperor Bed	21.99	Liddy	/product/images/product-1.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
2	2	Accent Chair	25.99	Caressa	/product/images/product-2.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
3	3	High-Back Bench	9.99	Ikea	/product/images/product-3.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
4	4	Wooden Table	19.99	Ikea	/product/images/product-4.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
5	5	Dining Table	69.99	Caressa	/product/images/product-5.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
6	6	Entertainmint Center	25.99	Liddy	/product/images/product-6.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
7	7	Albany Sectional	10.99	Ikea	/product/images/product-7.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
8	8	Sofa Set	69.99	Liddy	/product/images/product-8.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
9	9	Utopia Sofa	69.99	Liddy	/product/images/product-9.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
10	10	Modern Bookshelf	8.99	Marcos	/product/images/product-10.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
11	11	Albany Table	79.99	Marcos	/product/images/product-11.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ
12	12	Leather Sofa	39.95	Liddy	/product/images/product-12.jpg	https://loxixwyirrqkjruruqbq.supabase.co/storage/v1/object/public/hchung-1n-xx/produ

Total rows: 12 of 12 Query complete 00:00:00.173 Ln 1, Col 1

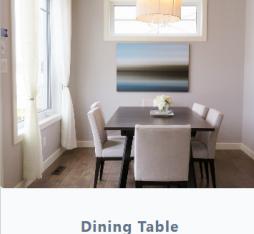
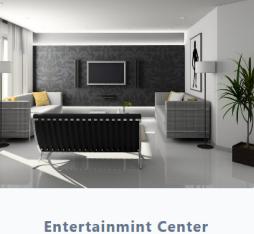
Successfully run. Total query runtime: 173 msec. 12 rows affected.

=> Chrome 顯示 product 資訊

localhost:5000/product/product_api_nodejs_14.html

Get Products from Local Node.js PostgreSQL server
Liang-Ting Lin, 913410014

		
Emperor Bed	Accent Chair	High-Back Bench
\$21.99	\$25.99	\$9.99

		
Wooden Table	Dining Table	Entertainmint Center

Console

```
product_14  product_api_nodejs_14.js:11
json from Node.js server
> Array(12) []
  ↪ 0: {id: 1, title: 'Emperor Bed', price: 21.99}
  ↪ 1: {id: 2, title: 'Accent Chair', price: 25.99}
  ↪ 2: {id: 3, title: 'High-Back Bench', price: 9.99}
  ↪ 3: {id: 4, title: 'Wooden Table', price: 19.99}
  ↪ 4: {id: 5, title: 'Dining Table', price: 69.99}
  ↪ 5: {id: 6, title: 'Entertainmint Center', price: 25.99}
  ↪ 6: {id: 7, title: 'Albany Sectional', price: 10.99}
  ↪ 7: {id: 8, title: 'Sofa Set', price: 69.99}
  ↪ 8: {id: 9, title: 'Utopia Sofa', price: 69.99}
  ↪ 9: {id: 10, title: 'Modern Bookshelf', price: 8.99}
  ↪ 10: {id: 11, title: 'Albany Table', price: 79.99}
  ↪ 11: {id: 12, title: 'Leather Sofa', price: 39.95}
  length: 12
[[Prototype]]: Array(0)
```