(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 2 shop_14 的部分

檔案及目錄結構如下圖


p0_14.png

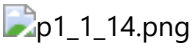
(15%) P1: 請用本地端的 pgAdmin 來建立 final_js_14 資料庫及 table menu_14

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)

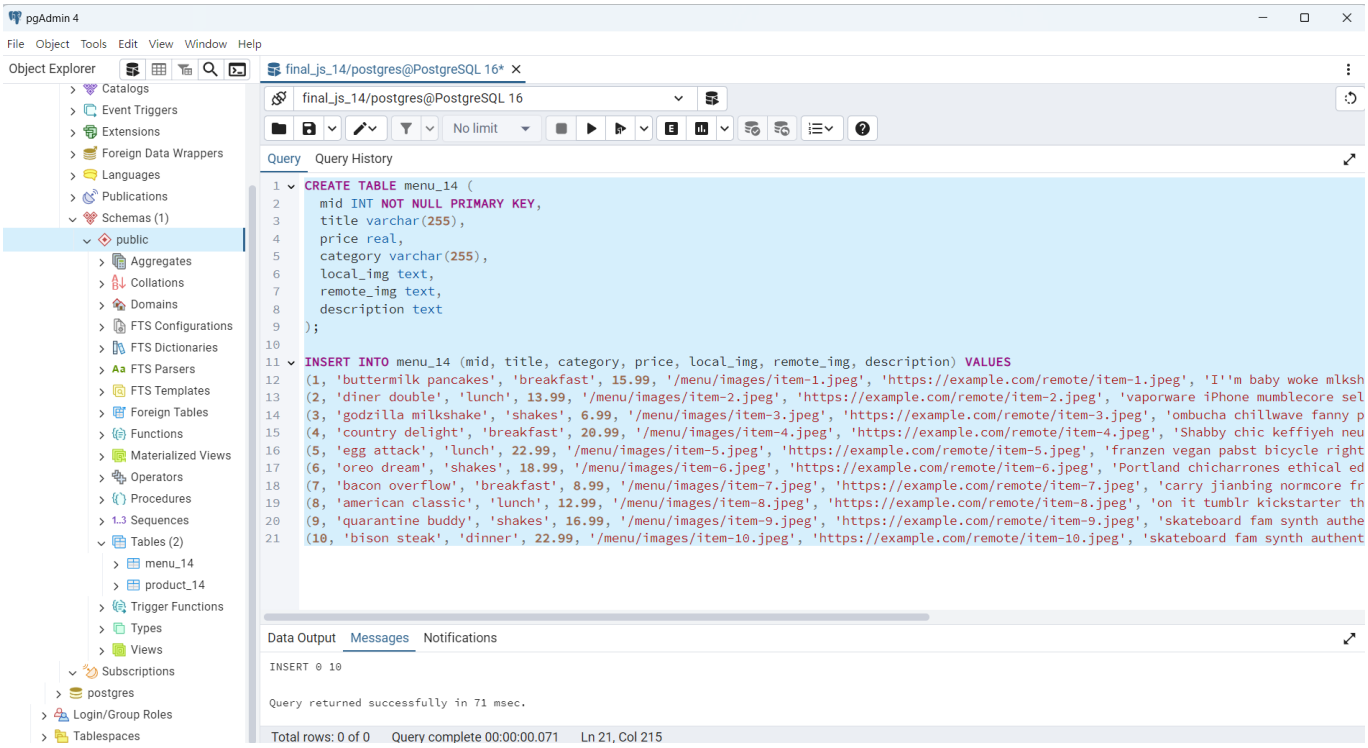
本次考試，老師題目顯示以 D=4,9 breakfast(4,7), dessert(9,3) 四筆資料 為主來解說。



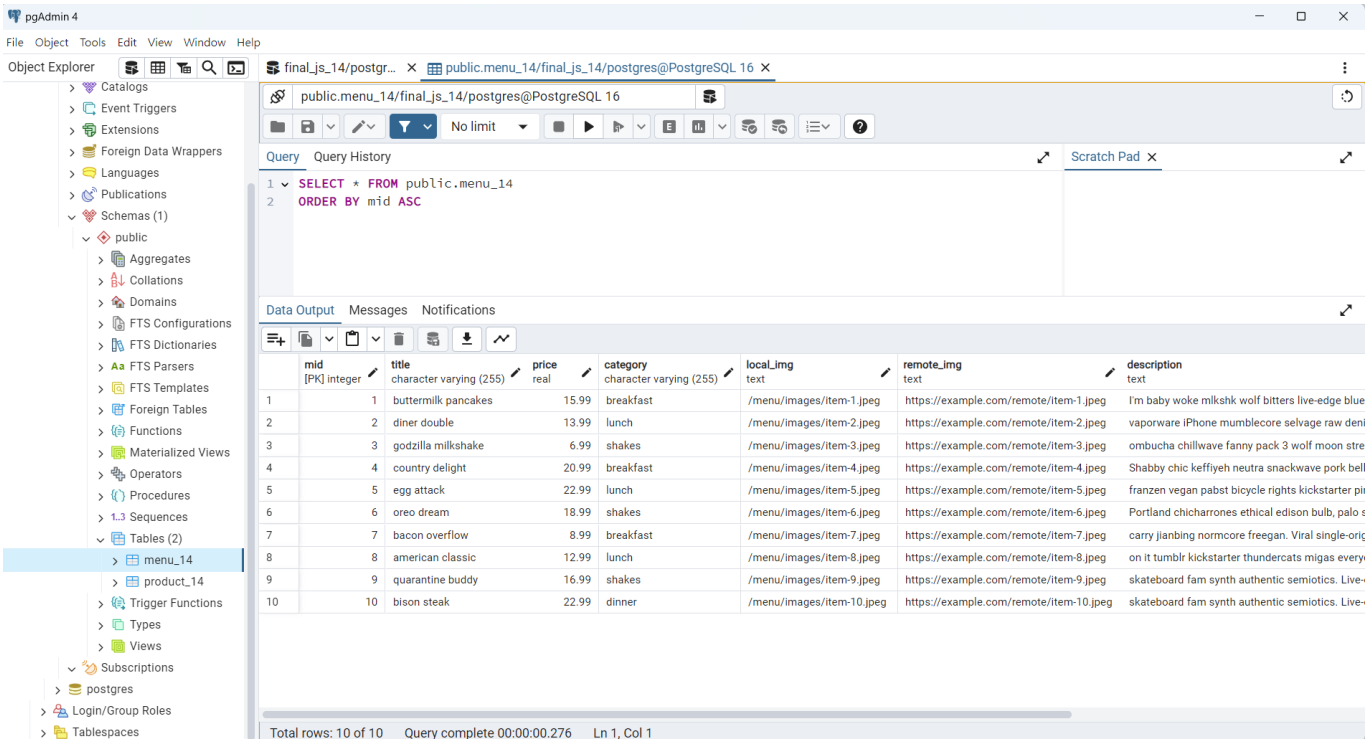
Your Answer

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

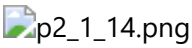


(15%) P2: 實作路由 /menu_14

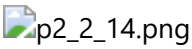
menu_14 theme 已經放在 /public/menu 目錄下 靜態頁面已經可以透過 /menu_14/static 顯現，有六筆

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

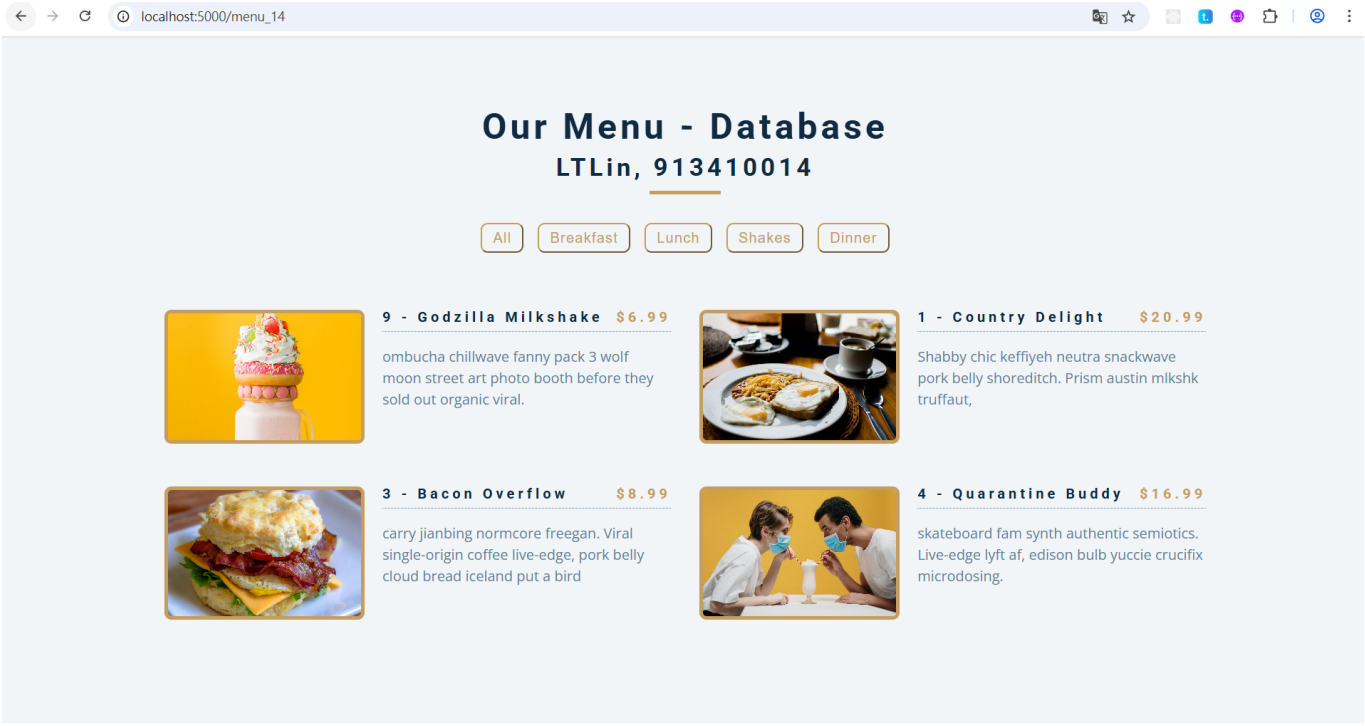


=> 將 menu_static.ejs 複製一份到 menu_14.ejs，從資料庫 menu_14 資料表中取得 4 筆資料顯示

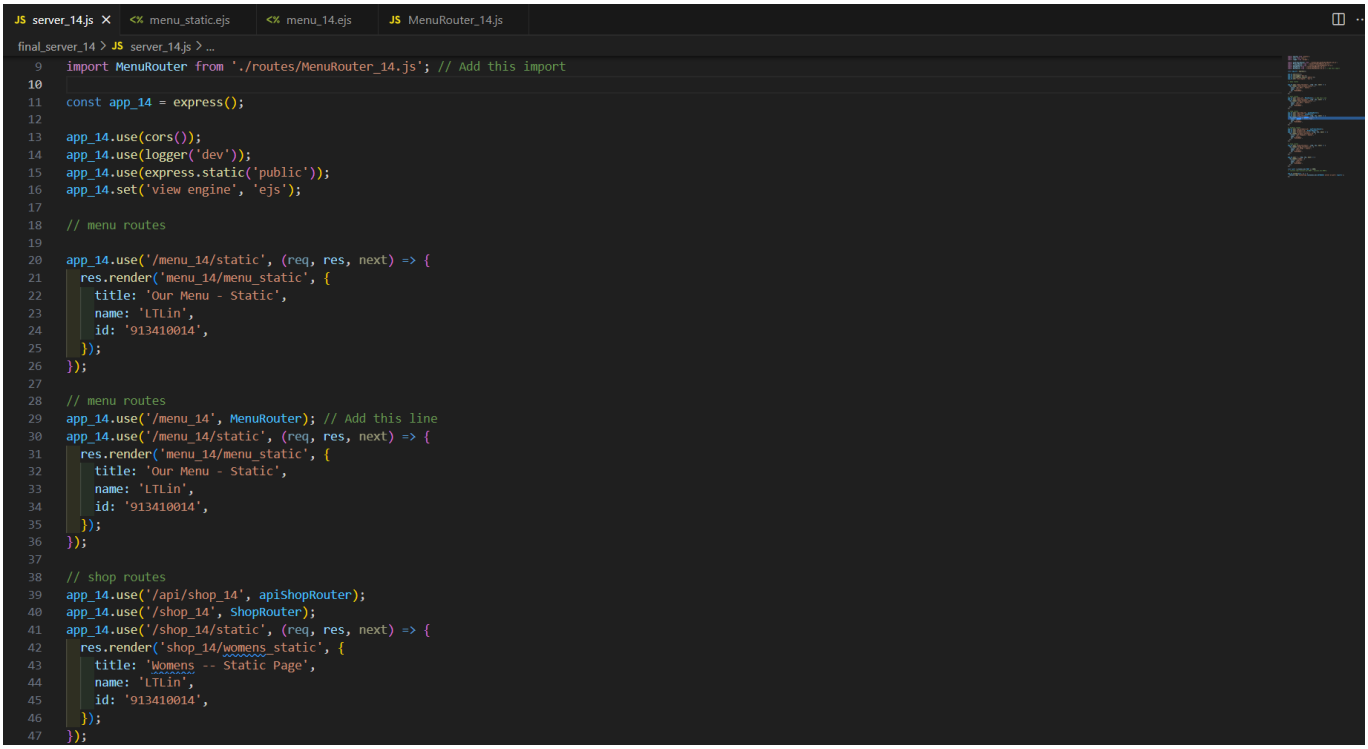


Your Answer

=> 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



```
import MenuRouter from './routes/MenuRouter_14.js'; // Add this import

const app_14 = express();

app_14.use(cors());
app_14.use(logger('dev'));
app_14.use(express.static('public'));
app_14.set('view engine', 'ejs');

// menu routes

app_14.use('/menu_14/static', (req, res, next) => {
  res.render('menu_14/menu_static', {
    title: 'Our Menu - Static',
    name: 'LTLin',
    id: '913410014',
  });
});

// menu routes
app_14.use('/menu_14', MenuRouter); // Add this line
app_14.use('/menu_14/static', (req, res, next) => {
  res.render('menu_14/menu_static', {
    title: 'Our Menu - Static',
    name: 'LTLin',
    id: '913410014',
  });
});

// shop routes
app_14.use('/api/shop_14', apiShopRouter);
app_14.use('/shop_14', ShopRouter);
app_14.use('/shop_14/static', (req, res, next) => {
  res.render('shop_14/womens_static', {
    title: 'Womens -- Static Page',
    name: 'LTLin',
    id: '913410014',
  });
});
```

=> routes/MenuRouter_14.js 之重點截圖

```
server_14.js  menu_static.ejs  menu_14.ejs  MenuRouter_14.js
final_server_14 > routes > MenuRouter_14.js > pool > password
import express from 'express';
import pkg from 'pg';
const { Pool } = pkg;

const router = express.Router();

const pool = new Pool({
  user: 'postgres',
  host: 'localhost',
  database: 'final_js_14',
  password: '0000', // Replace with your actual password
  port: 5432,
});

router.get('/', async (req, res) => {
  try {
    // Query to get 4 items based on student ID ending digit 4
    // D=4,9 => breakfast(4,7), dessert(9,3)
    const query =
      'SELECT * FROM menu_14 WHERE mid IN (4, 7, 9, 3) ORDER BY mid';
    const result = await pool.query(query);

    res.render('menu_14/menu_14', {
      title: 'Our Menu - Database',
      name: 'LTLin',
      id: '913410014',
      menuItems: result.rows,
    });
  } catch (err) {
    console.error('Database error:', err);
    res.status(500).send('Database error');
  }
});

export default router;
```

=> menu_14.ejs 有關 code 之重點截圖

```
server_14.js  menu_static.ejs  menu_14.ejs  MenuRouter_14.js
final_server_14 > views > menu_14 > menu_14.ejs > html > body > section.menu > div.section-center > ? > article.menu-item > div.item-info > header > h4 > ?
<html lang="en">
  <body>
    <section class="menu">
      <div class="btn-container">
        <button type="button" class="filter-btn" data-id="all">all</button
        ><button type="button" class="filter-btn" data-id="breakfast">
          breakfast</button
        ><button type="button" class="filter-btn" data-id="lunch">lunch</button
        ><button type="button" class="filter-btn" data-id="shakes">
          shakes</button
        ><button type="button" class="filter-btn" data-id="dinner">
          dinner
        </button>
      </div>
      <!-- menu items -->

      <div class="section-center">
        <% menuItems.forEach(function(item, index) { %>
        <article class="menu-item">
          <img
            src="<%= item.local_img %>"
            alt="<%= item.title %>"
            class="photo"
          />
          <div class="item-info">
            <header>
              <h4><%= id[index] %> - <%= item.title %></h4>
              <h4 class="price">$<%= item.price %></h4>
            </header>
            <p class="item-text"><%= item.description %></p>
          </div>
        </article>
        <% }); %>
      </div>
    </section>
  </body>
</html>
```

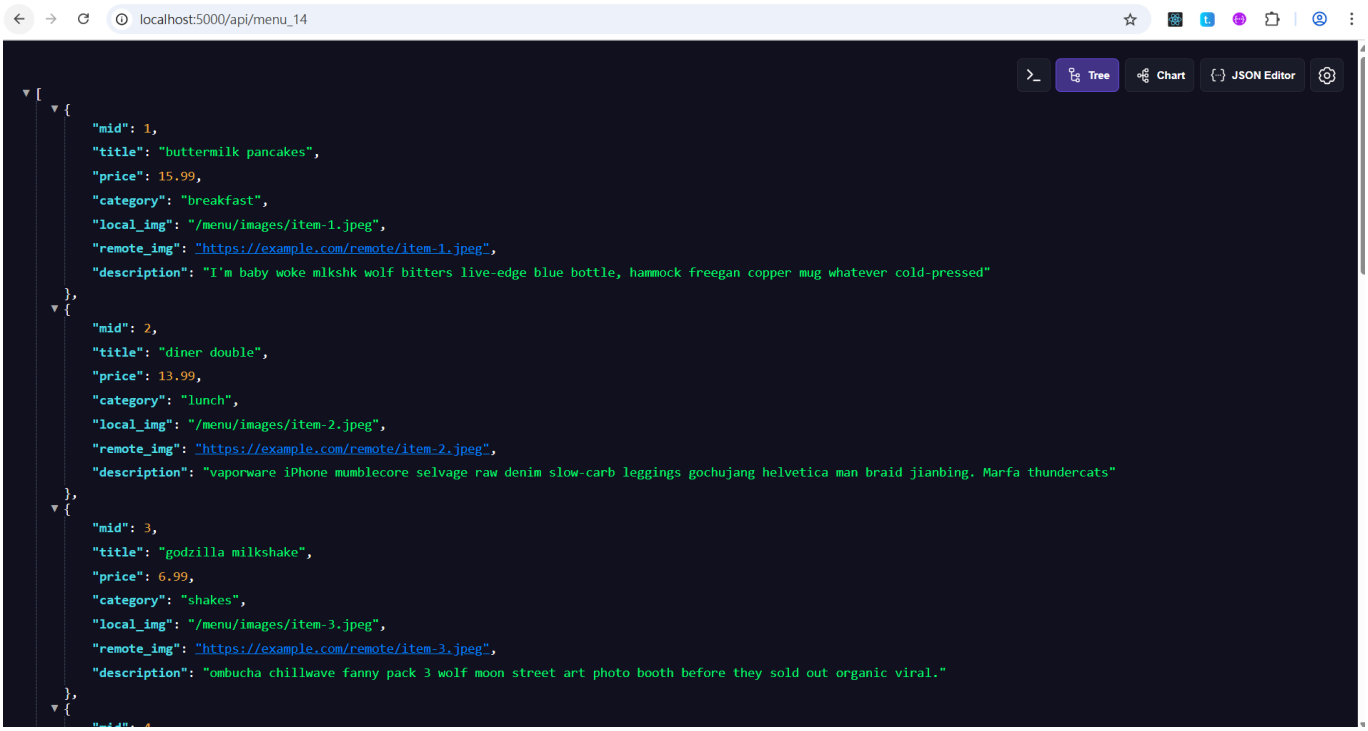
(15%) P3: 實作 路由 /api/menu_14

=> Chrome 顯示

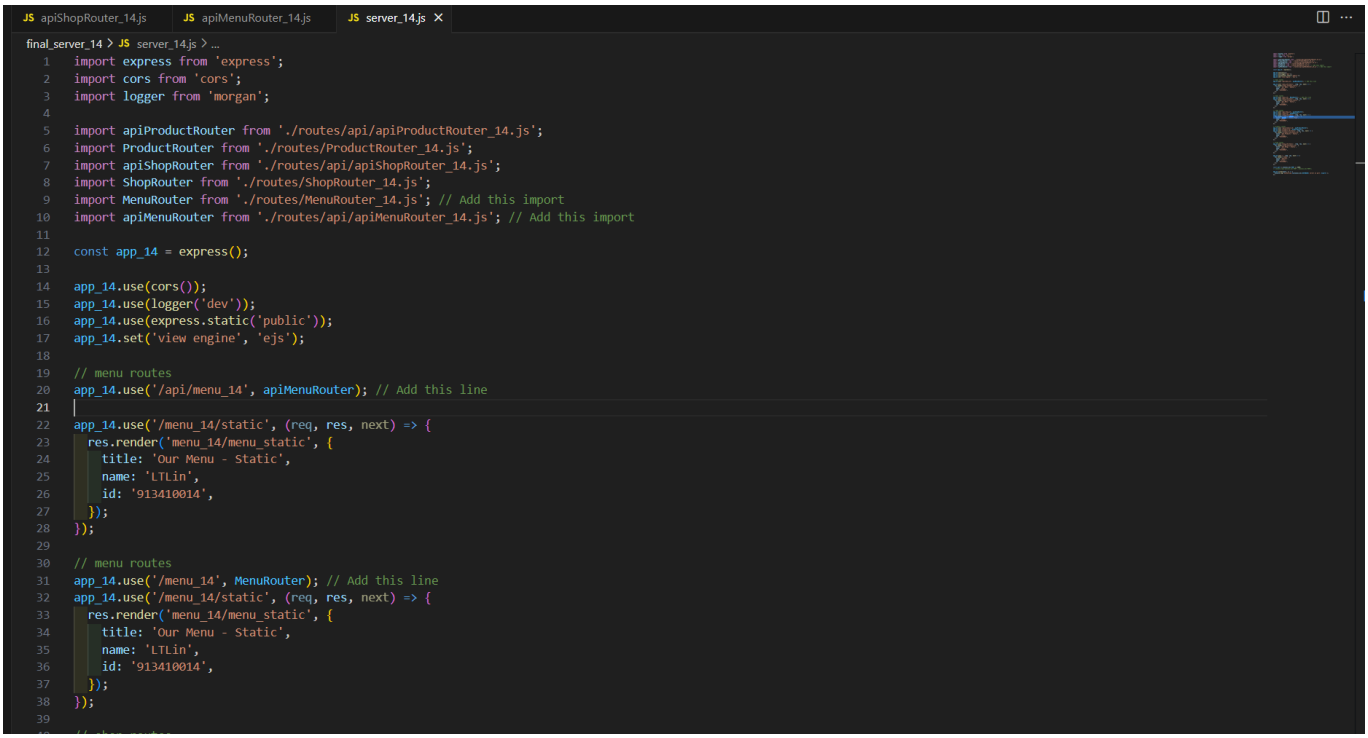
p3_1_14.png

Your Answer

=> Chrome 顯示



=> server_14.js code 重點



=> apiMenuRouter_14.js code 重點

```
JS apiShopRouter_14.js   JS apiMenuRouter_14.js ×   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

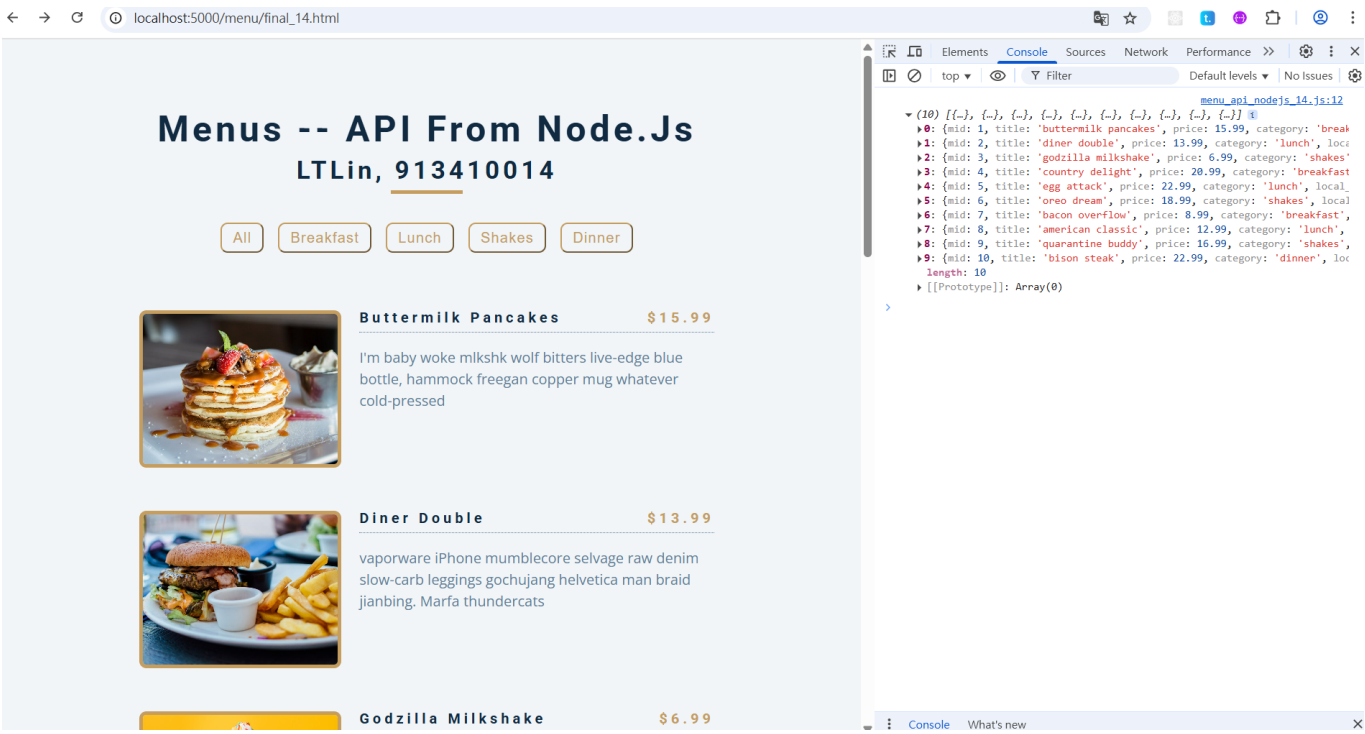
在 menu_api_nodejs_14.js 中儲存 menu 的變數名稱為 menu_14，不可以改變

=> Chrome 顯示

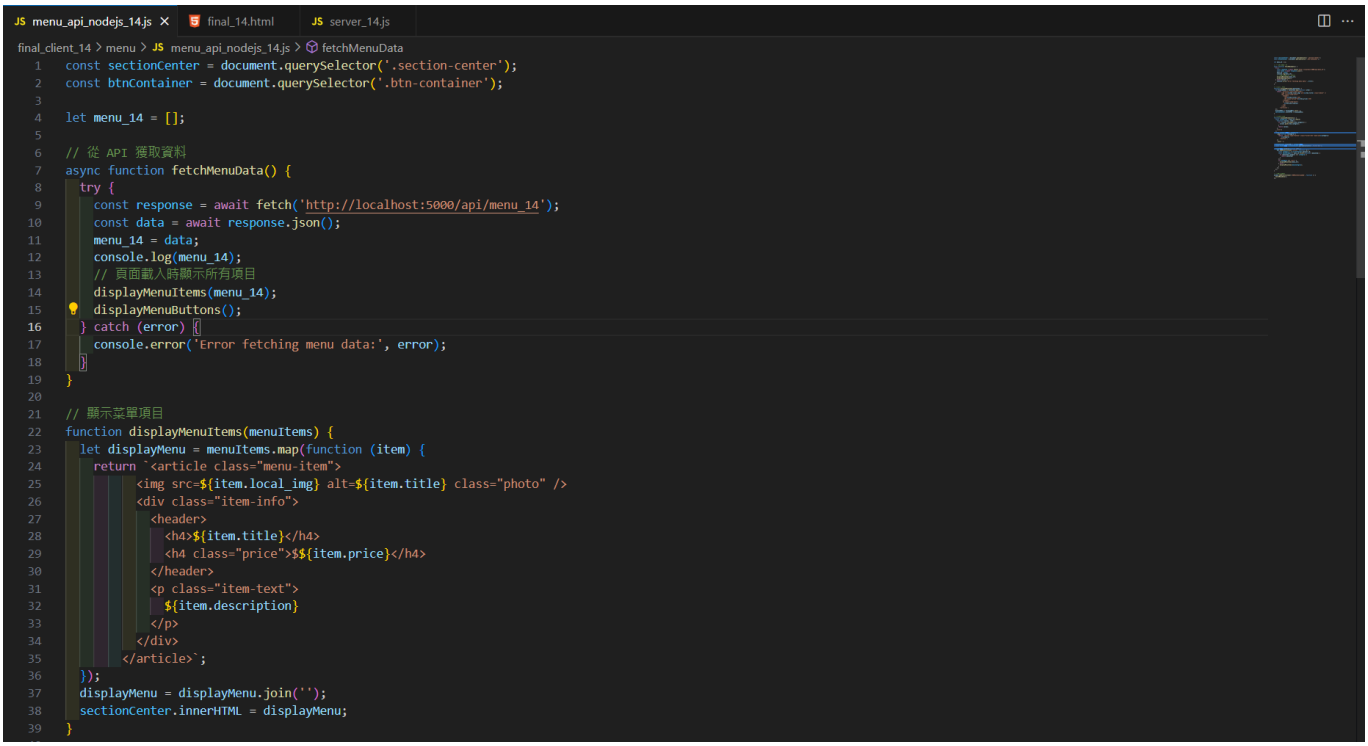
p4_1_14.png

Your Answer

=> Chrome 顯示



=> 相關 js code 顯示



相關 js code 顯示

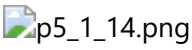
(10%) P5: 請針對 P4 題目，加入 filter 篩選功能

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

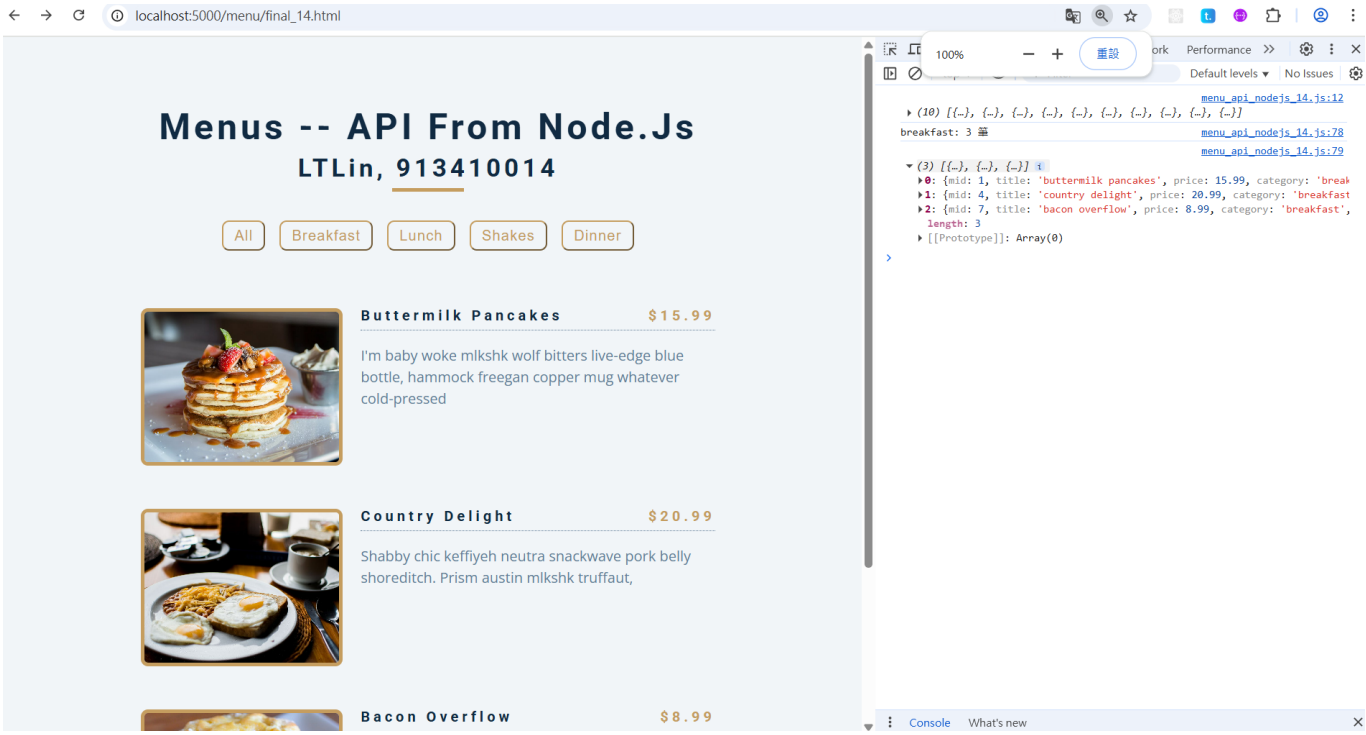
按照順序，點選按鈕 Breakfast, Lunch, Shakes, Dessert，然後用 console.log 顯示筆數，出現的筆數請展開

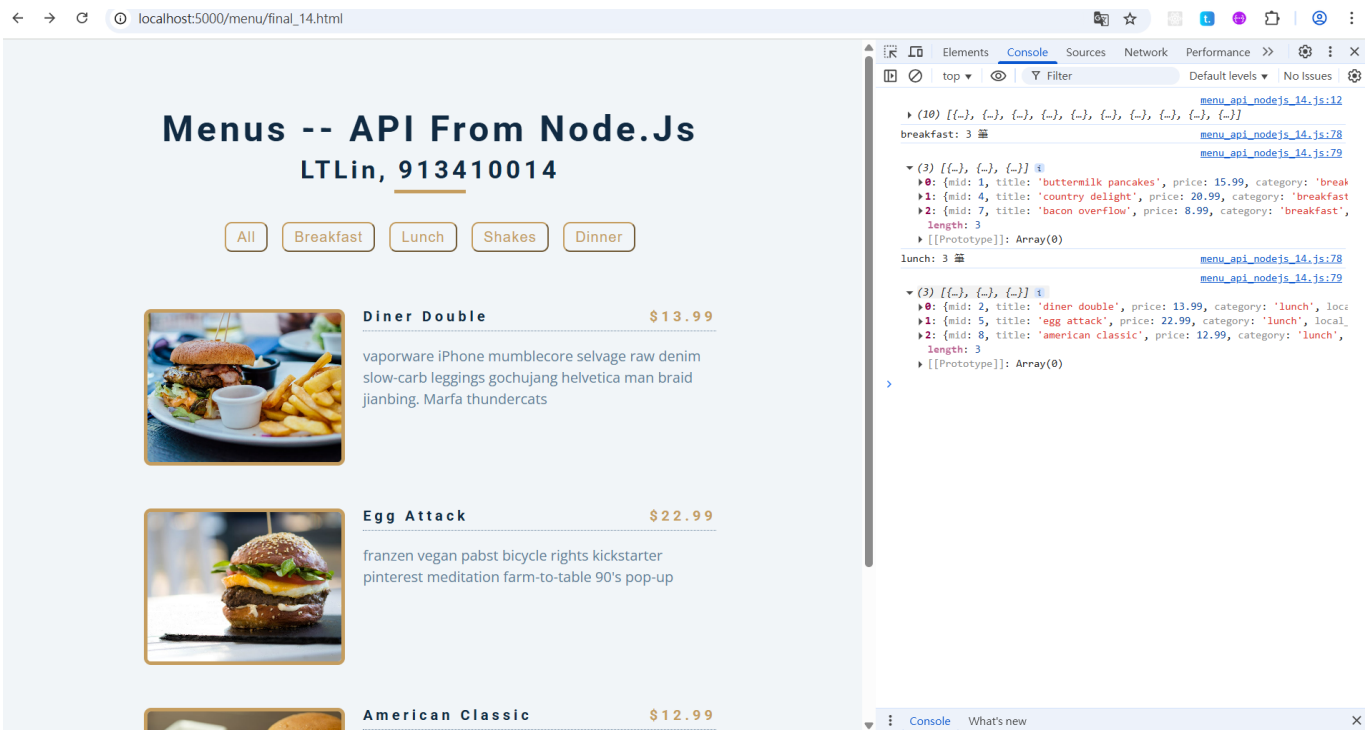


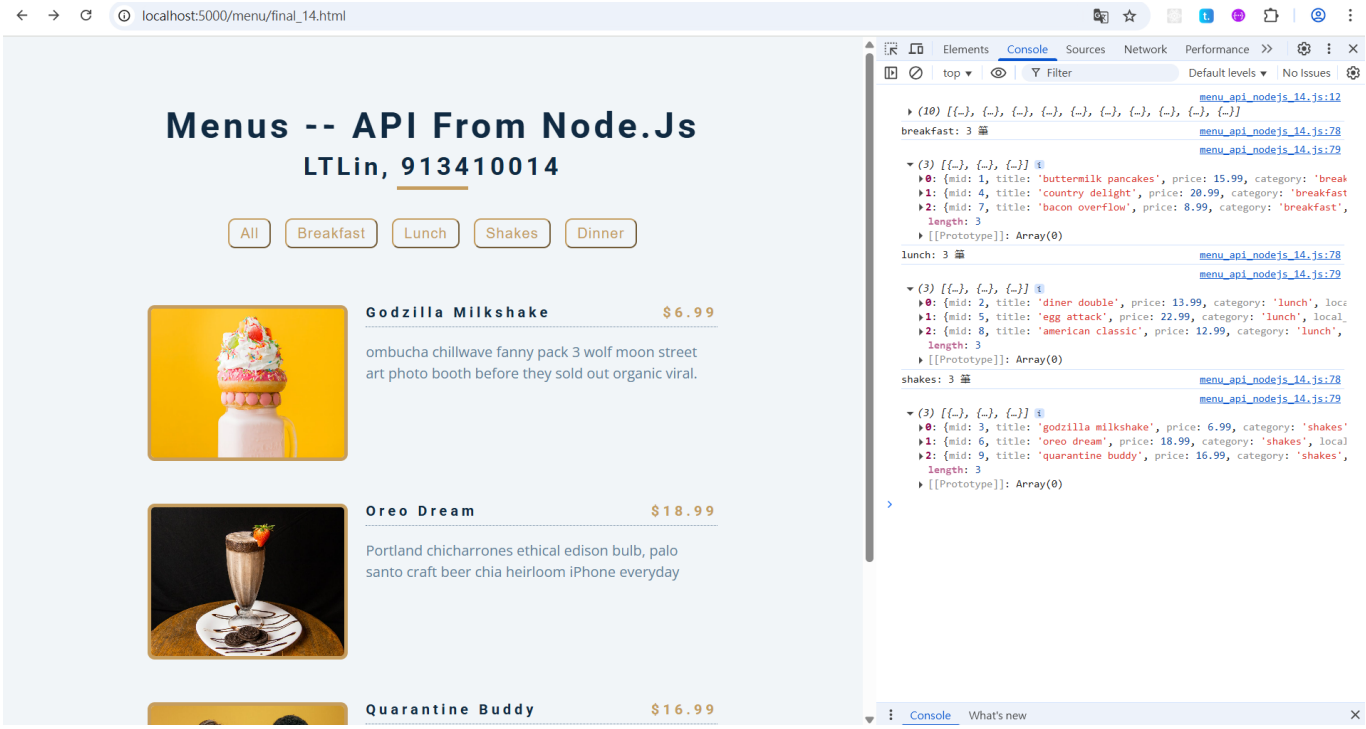
Your Answer

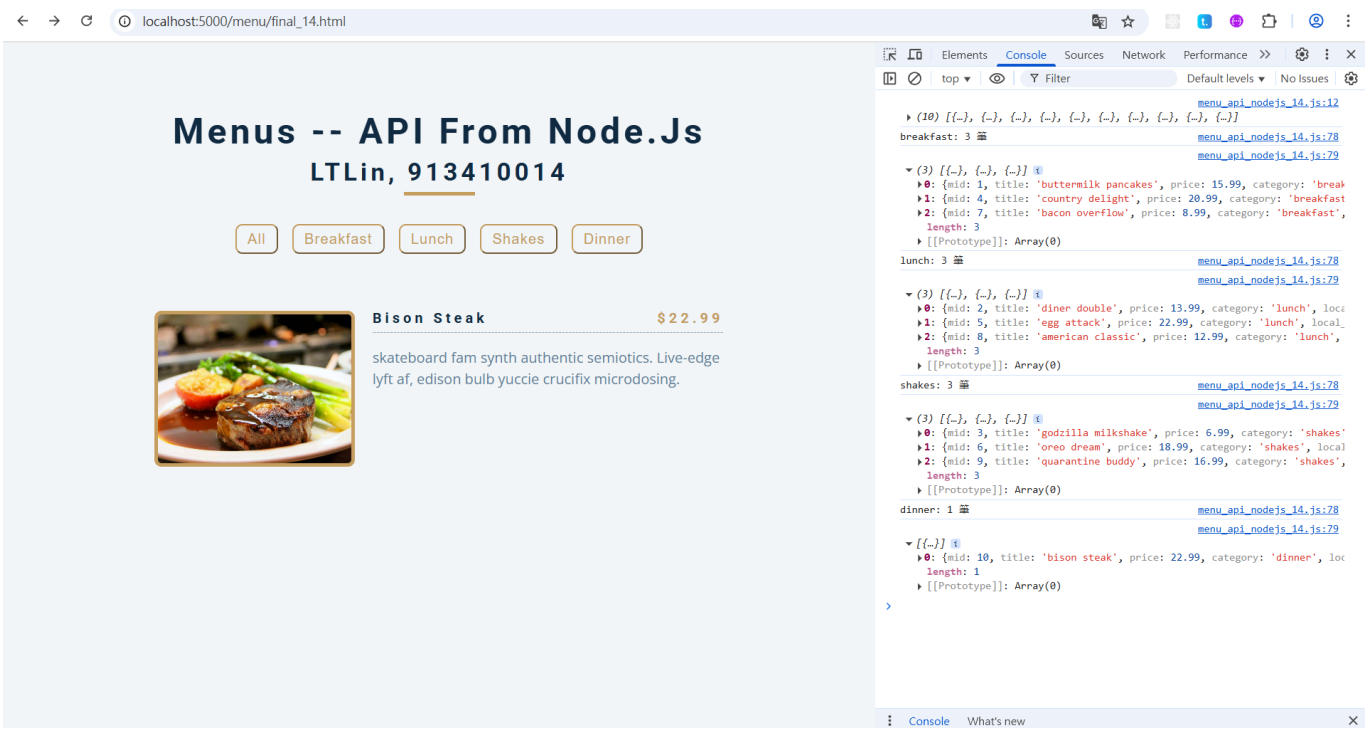
=> Chrome 顯示

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

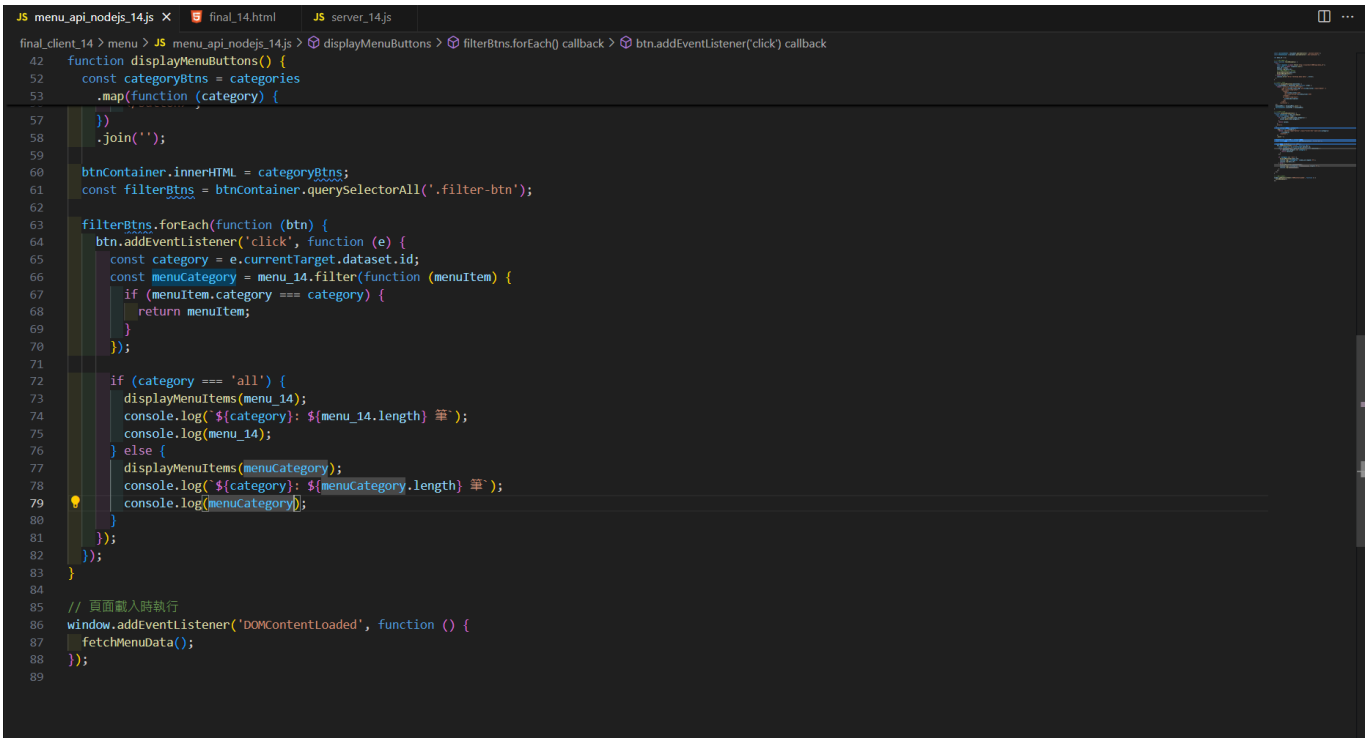





localhost:5000/menu/final_14.html
Menus -- API From Node.Js
LTLin, 913410014
All
Breakfast
Lunch
Shakes
Dinner
Godzilla Milkshake
$6.99
ombucha chillwave fanny pack 3 wolf moon street art photo booth before they sold out organic viral.
Oreo Dream
$18.99
Portland chicharrones ethical edison bulb, palo santo craft beer chia heirloom iPhone everyday
Quarantine Buddy
$16.99
Elements
Console
Sources
Network
Performance
top
Filter
Default levels
No Issues
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'}
1: {mid: 5, title: 'egg attack', price: 22.99, category: 'lunch'}
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'}
2: {mid: 9, title: 'quarantine buddy', price: 16.99, category: 'shakes'}
length: 3
[[Prototype]]: Array(0)
Console
What's new


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.
Elements
Console
Sources
Network
Performance
top
Filter
Default levels
No Issues
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'}
1: {mid: 5, title: 'egg attack', price: 22.99, category: 'lunch'}
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'}
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
[{…}]
0: {mid: 10, title: 'bison steak', price: 22.99, category: 'dinner'}
length: 1
[[Prototype]]: Array(0)
Console
What's new

=> 相關 js code 顯示

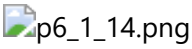


相關js code 顯示

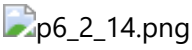
(10%) P6: 在前端顯示 product, shop 資訊

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

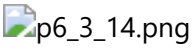
=> 資料庫顯示 shop_14 資料表資料



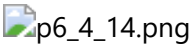
=> Chrome 顯示 shop 資訊



=> 資料庫顯示 product_14 資料表資料



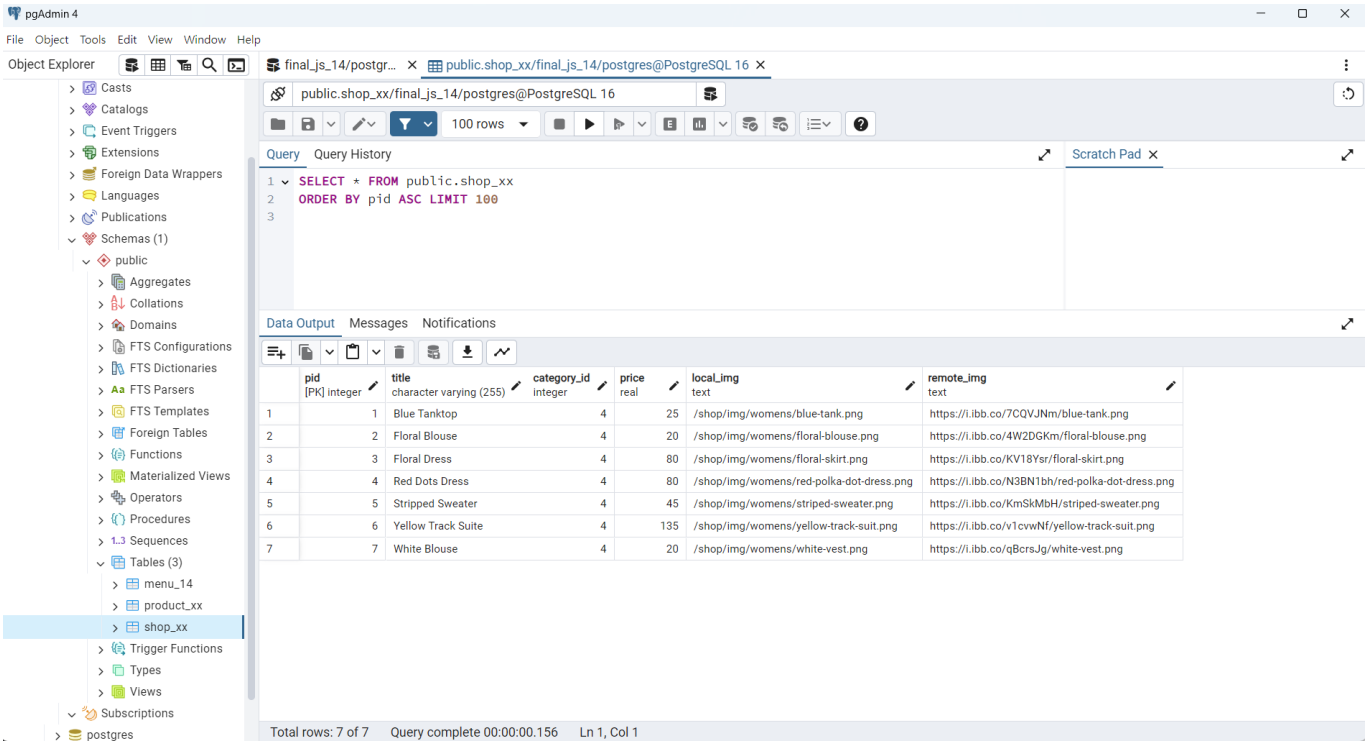
=> Chrome 顯示 product 資訊



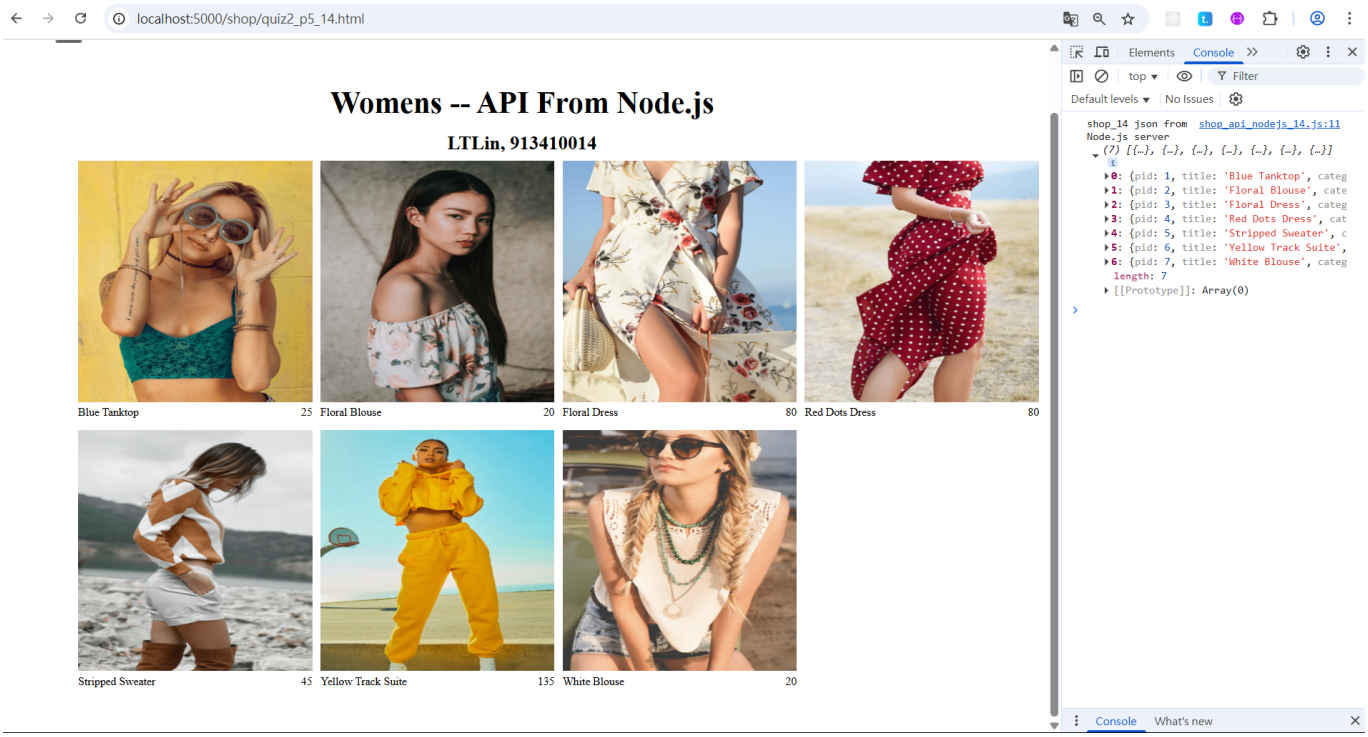
Your Answer

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

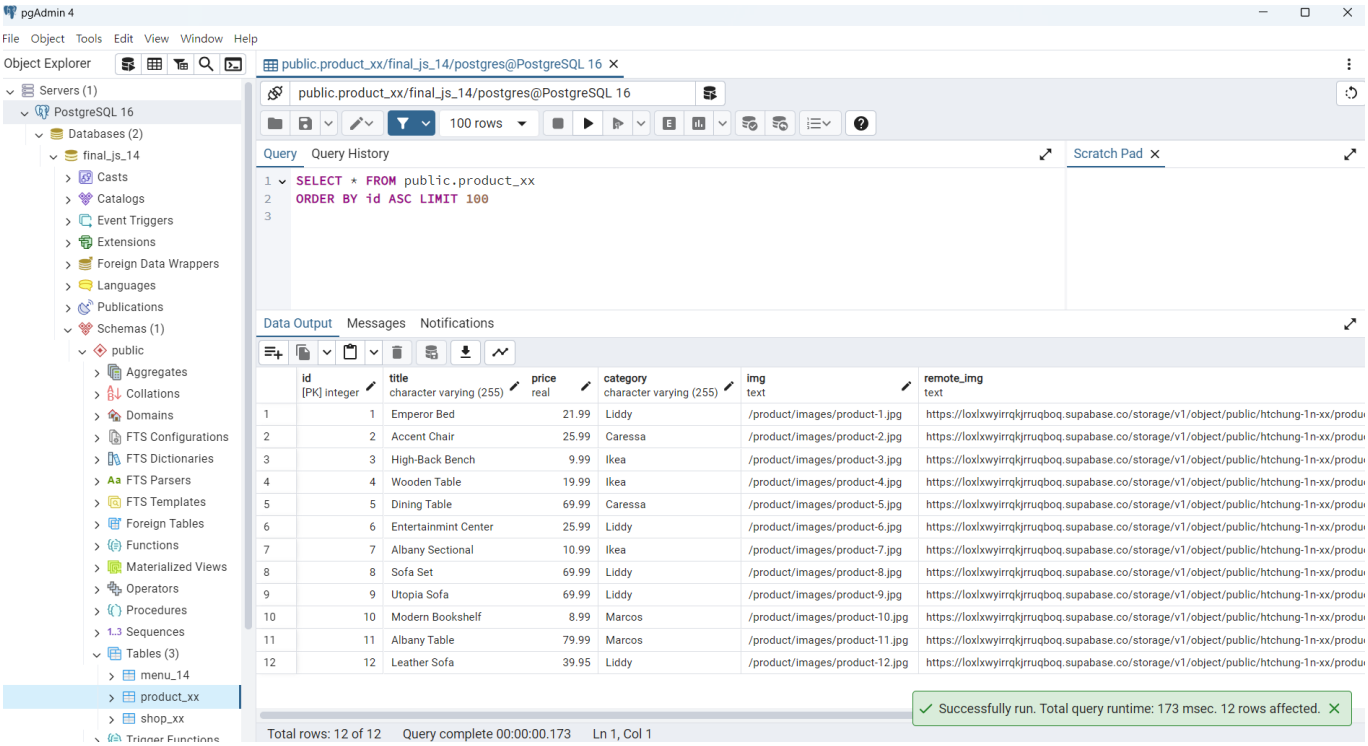
=> 資料庫顯示 shop_14 資料表資料



=> Chrome 顯示 shop 資訊



=> 資料庫顯示 product_14 資料表資料



=> Chrome 顯示 product 資訊