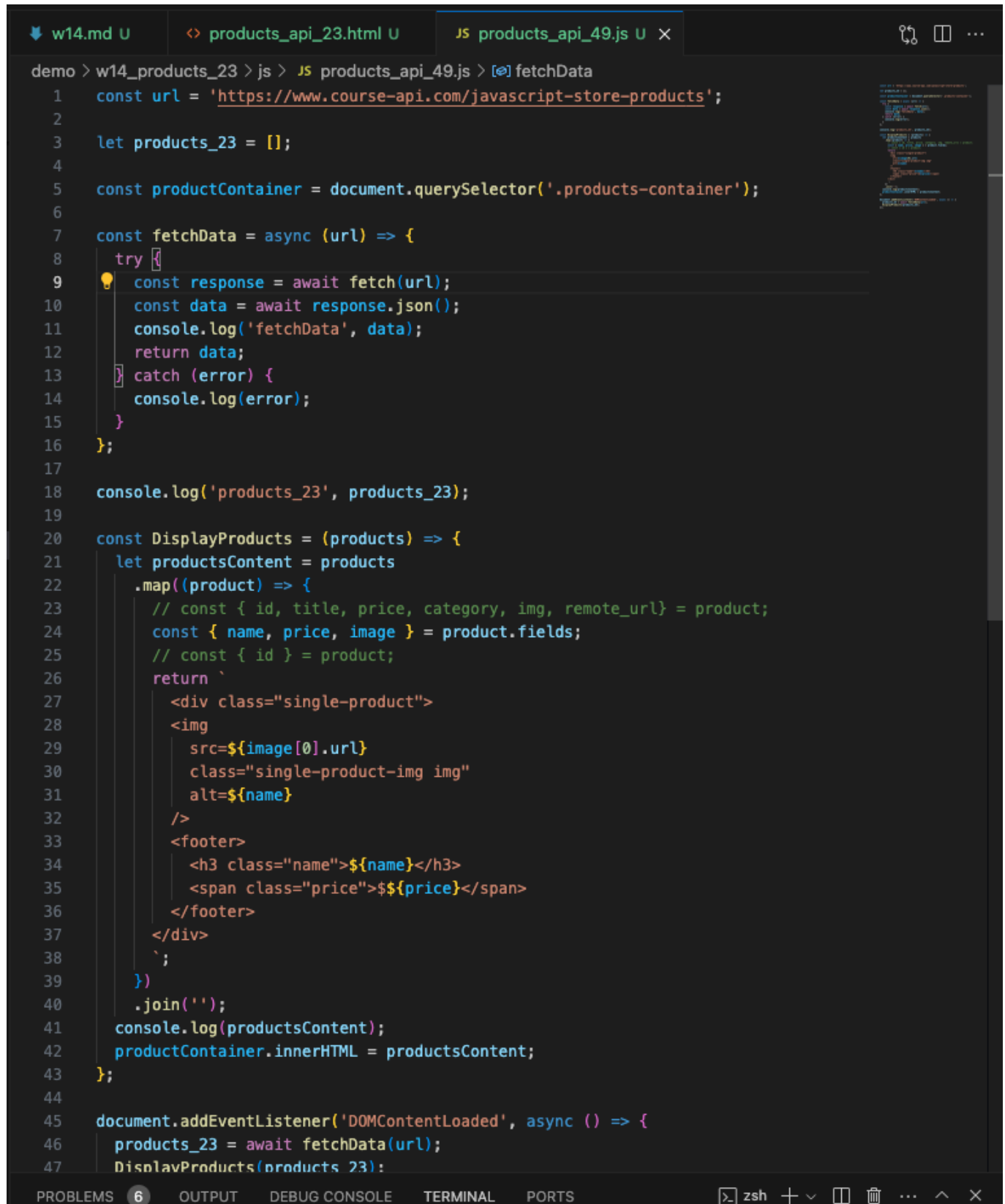


GITHUB :

[My github repo url 912410023](#)

[My Vercel url 912410023](#)

W14-P1: Do products_api_xx.js to get data from an api url




```
demo > w14_products_23 > js > JS products_api_49.js > [⌘] fetchData
1  const url = 'https://www.course-api.com/javascript-store-products';
2
3  let products_23 = [];
4
5  const productContainer = document.querySelector('.products-container');
6
7  const fetchData = async (url) => {
8    try {
9      const response = await fetch(url);
10     const data = await response.json();
11     console.log('fetchData', data);
12     return data;
13   } catch (error) {
14     console.log(error);
15   }
16 };
17
18 console.log('products_23', products_23);
19
20 const DisplayProducts = (products) => {
21   let productsContent = products
22     .map((product) => {
23     // const { id, title, price, category, img, remote_url } = product;
24     const { name, price, image } = product.fields;
25     // const { id } = product;
26     return `
27       <div class="single-product">
28         <img
29           src=${image[0].url}
30           class="single-product-img img"
31           alt=${name}
32         />
33         <footer>
34           <h3 class="name">${name}</h3>
35           <span class="price">${price}</span>
36         </footer>
37       </div>
38     `;
39   })
40   .join('');
41   console.log(productsContent);
42   productContainer.innerHTML = productsContent;
43 };
44
45 document.addEventListener('DOMContentLoaded', async () => {
46   products_23 = await fetchData(url);
47   DisplayProducts(products_23);
48 }
```

127.0.0.1:5500/demo/w14_products_23/products_api_2


Products_23 Demo

Kevin, 912410023



High-Back Bench

\$999



Albany Table

Elements

Console

Live reload enabled. products_api_23.html:59

products_23

Array (0)

No Properties

Array Prototype

fetchData

products_api_49.js:11

Object

fields: Object

colors: ["#f15025", "#222"] (2)

company: "ikea"

featured: true

image: Array (1)

{id: "attcvDDM1kF6G2iNi", width: 1000, height: 635}

Array Prototype

name: "high-back bench"

price: 999

Object Prototype

id: "rec43w3ipXvP28vog"

Object Prototype

1 {id: "rec4f2RIftFCb7aHh", fields: Object}

2 {id: "rec8kkCmSiMkbkiko", fields: Object}

3 {id: "recBohCqQsot4Q4II", fields: Object}

4 {id: "recDG1JRZnbpRHpoy", fields: Object}

5 {id: "recNWGyP7kjFhSqw3", fields: Object}

6 {id: "recZEougl5bbY4AE", fields: Object}

7 {id: "recjMK1jgTb2ld7sv", fields: Object}

8 {id: "recmg2a1ctaEJNZhu", fields: Object}

9 {id: "recvK0MNR3YFw0bEt3", fields: Object}

10 {id: "recxaXFy5IW539sgM", fields: Object}

11 {id: "recyqtRgLGNGt04Q5", fields: Object}

Array Prototype

DisplayProd... products_api_49.js:41

<div class= "single-product">

<img

src=https://course-api.com/images/store/product-1.jpeg

class="single-product-img img"

alt=high-back bench

>

<footer>

<h3 class="name">high-back bench</h3>

\$999

</footer>

</div>

W14-P2: Do products_localJson_xx.js to get local json data (/api/productsData.json)

The screenshot displays a web browser window with the URL `127.0.0.1:5500/demo/w14_products_23/products_localJson_49.js`. The browser's developer tools are open, showing the Network tab. A red box highlights the `0.json` file in the file list and the corresponding network request details.

The JavaScript code in `products_localJson_49.js` is as follows:

```
demo > w14_products_23 > js > JS products_localJson_49.js > fetchData
1  const url = './api/data/0.json';
2
3  let products_23 = [];
4
5  const productContainer = document.querySelector('.products-container');
6
7  const fetchData = async (url) => {
8    try {
9      const response = await fetch(url);
10     const data = await response.json();
11     console.log('fetchData', data);
12     return data;
13   } catch (error) {
14     console.log(error);
15   }
16 };
17
```

The Network tab shows the following details for the `0.json` request:

- Summary:**
 - URL: `http://127.0.0.1:5500/demo/w14_products_23/api/data/0.json`
 - Status: 200 OK
 - Source: Network
 - Address: 127.0.0.1:5500
 - Initiator: `products_localJson_49.js:9`
- Request:**
 - GET /demo/w14_products_23/api/data/0.json HTTP/1.1
 - Accept: */*
 - Accept-Encoding: gzip, deflate
 - Accept-Language: en-US,en;q=0.9
 - Connection: keep-alive
 - Host: 127.0.0.1:5500
 - Referer: `http://127.0.0.1:5500/demo/w14_products_23/products_localJson_49.js`
 - Sec-Fetch-Dest: empty
 - Sec-Fetch-Mode: cors
 - Sec-Fetch-Site: same-origin
 - User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Version/17.3 Safari/605.1.15
- Response:**
 - HTTP/1.1 200 OK
 - Accept-Ranges: bytes
 - Access-Control-Allow-Credentials: true

W14-P3: Use SQL to create schemas and data for company_xx and products_xx

=> company_xx schema and data

Table Editor

schema: public

+ New table

Search tables...

company

company_23

company_test

products

products_23

store_xx

0x55xx5's Org Free / tku_23_1122jstest / Enable branching

Filter Sort Insert

Add RLS policy role postgres Realtime off API Docs

	id int4	created_at timestamp	name varchar	email text	phone varchar
<input type="checkbox"/>	1	2024-05-23 12:21:49.796205	asdasd	AS@gmail.com	194541313
<input type="checkbox"/>	2	2024-05-23 12:22:26.243341	asdasd	AS@gmail.com	194541313
<input type="checkbox"/>	3	2024-05-23 12:22:26.96942	asdasd	AS@gmail.com	194541313
<input type="checkbox"/>	4	2024-05-23 12:22:27.383087	asdasd	AS@gmail.com	194541313
<input type="checkbox"/>	5	2024-05-23 12:22:27.744613	asdasd	AS@gmail.com	194541313
<input type="checkbox"/>	6	2024-05-23 12:22:28.170798	asdasd	AS@gmail.com	194541313

SQL Editor

+ New query

Templates

Quickstarts

Search queries...

YOUR QUERIES

Company and Store Tables

insert_product

Product Table

Products Table

w14

0x55xx5's Org Free / tku_23_1122jstest / Enable branching

Feedback

```
21
22
23 -- products_xx table
24
25 CREATE TABLE products_23 (
26   id serial NOT NULL PRIMARY KEY,
27   created_at timestamp NOT NULL DEFAULT NOW(),
28   title varchar(255),
29   price real,
30   "companyId" int4,
31   "localImg" text,
32   "remoteImg" text,
33   CONSTRAINT fk_1 FOREIGN KEY ("companyId") REFERENCES company_23(id) on DELETE SET NULL on UPDATE CASCADE
34 );
35
36 ALTER TABLE products_23 ENABLE ROW LEVEL SECURITY;
37 create policy "Allow SELECT access for all users" on "public"."products_23" as PERMISSIVE for SELECT to public using (true);
38 create policy "Allow INSERT access for all users" on "public"."products_23" as PERMISSIVE for INSERT to public with check
39 (true);
40 create policy "Allow UPDATE access for all users" on "public"."products_23" as PERMISSIVE for UPDATE to public using (true)
41 with check (true);
42 create policy "Allow DELETE access for all users" on "public"."products_23" as PERMISSIVE for DELETE to public using (true);
43
44 INSERT INTO products_23 (title, price, "companyId", "localImg", "remoteImg")
45 VALUES
46 ('Emperor Bed', 21.99, 2, './images/product-1.jpg', 'https://ifafquhvbvtxtzagzhf.supabase.co/storage/v1/object/public/demo_xx/products_xx/product-1.jpg'),
47 ('Accent Chair', 25.99, 4, './images/product-2.jpg', 'https://ifafquhvbvtxtzagzhf.supabase.co/storage/v1/object/public/demo_xx/products_xx/product-2.jpg'),
48 ('High-Back Bench', 9.99, 1, './images/product-3.jpg', 'https://ifafquhvbvtxtzagzhf.supabase.co/storage/v1/object/public/
```

Results Chart

role postgres Run

Click Run to execute your query.

=> products_xx schema and data

The image displays a web development workflow. On the left, a code editor shows the JavaScript code for a 'Products Demo' application. The code uses Supabase to fetch product data and renders it into HTML. The browser preview in the center shows the rendered page, which features two highlighted products: 'Emperor Bed' and 'Accent Chair'. The console on the right shows the product data array, which lists 12 items including chairs, tables, sofas, and bookshelves with their respective prices and creation timestamps.

