

Vue 實務案例：整合 fetch 串接後端資料

1. 實務開發目標：抓取後端提供的資料，顯示在網頁上，
看右邊，產品資料列表，是透過網址，抓取後端資料，顯示在畫面上：



2. 流程：首先網頁上只有一個標題，是一個靜態的文字，然後我們有一個後端伺服器，首先還沒跟後端串接的時候，畫面會先顯示「資料載入中」，然後透過 JavaScript 內建的 fetch 函式去串接後端提供的資料網址，那我們就可以連線到後端的伺服器，接著我們就可以取後 JSON 格式的資料回應，最後再把 JSON 格式的資料渲染到網頁上

實務開發目標

運作概念說明

產品資料列表

資料載入中

實務開發目標

運作概念說明

音量

產品資料列表

資料載入中

1. fetch("

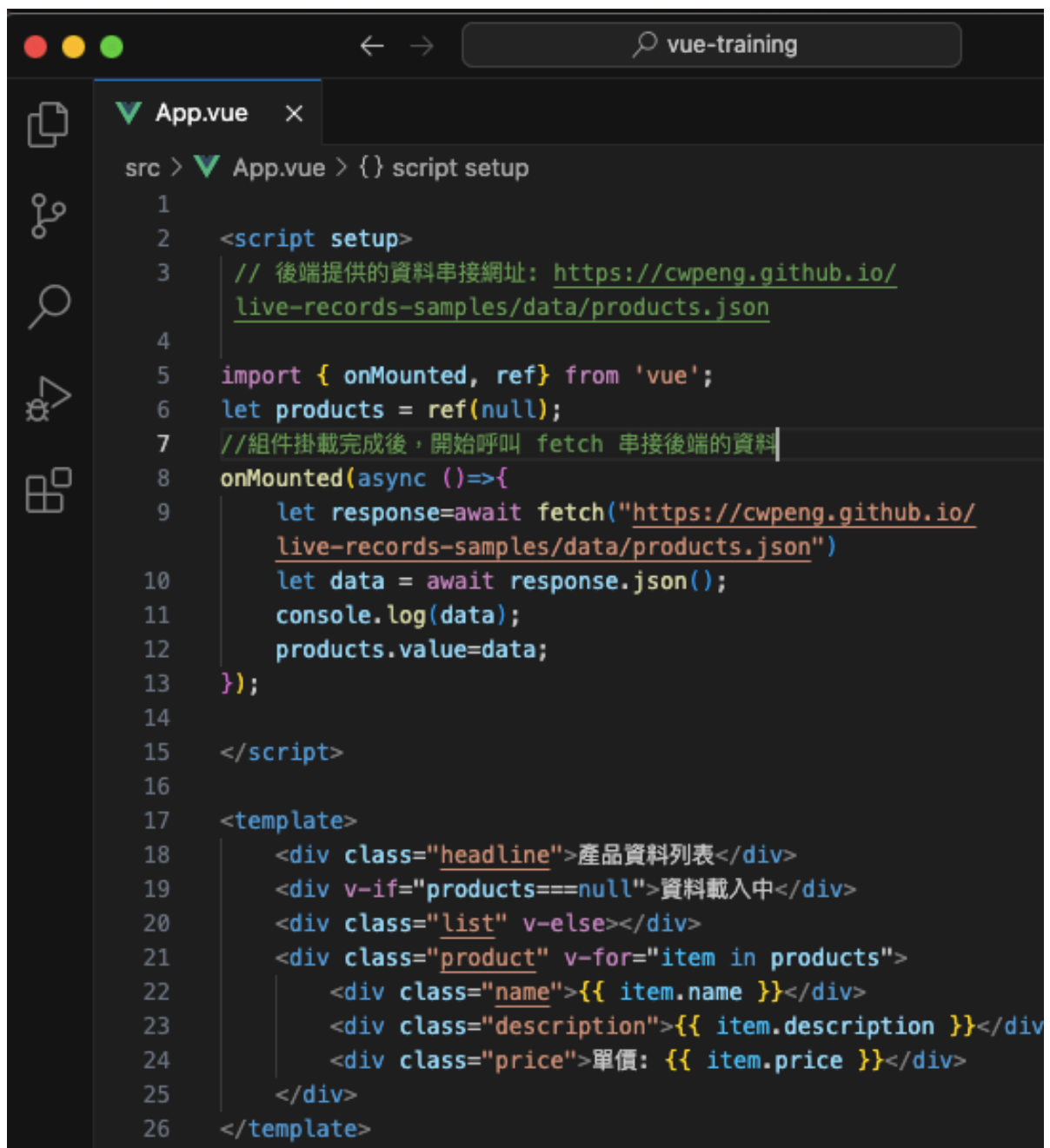
2. 取得 JS



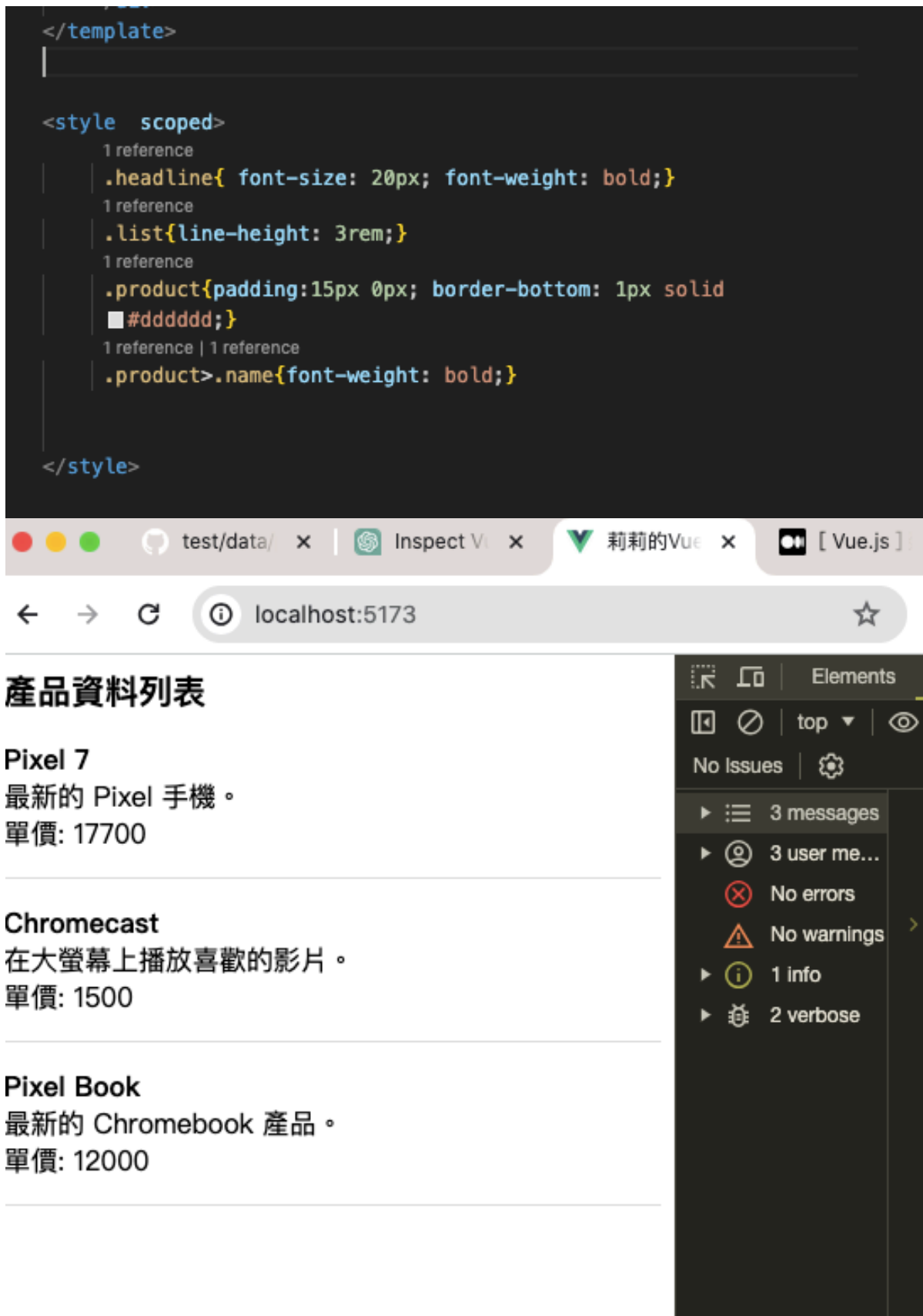
3. 開發準備事項：後端提供的資料網址，
JavaScript fetch 基本用法
Vue 樣板語法的流程控制
Vue 響應式狀態，生命週期
Vue 組件切割操作



4. 寫程式，我們要串接後端提供的網址，把後端抓到的資料渲染到畫面上，我們載入 `ref`, `unmounted` 從 `vue` 套件，當需要一個響應式狀態 (`products`) 來記錄產品資料，他就是要對應我們從後端抓到的資料，即若要對應後端抓到的資料，第一個時間點就是空值(`null`)，所以畫面呈現「資料載入中」，`vue` 的組件被畫到畫面上後，就是「組件掛載完成」，然後開始啟動串接後端的程序，我們就使用 `JavaScript` 內建的函式 `fetch()` 來串接後端，為了使用 `fetch` 函式，我們就使用 `unMounted` 這個 `HOOKS`，代表掛載完成之後要做的事情，`function` 我們用 `async` 搭配函式內部的 `await`，利用 `fetch` 連接後端網址，得到一個回應之後，到用 `json` 格式解讀得到一個 `data`，最後我們把 `data` 更新到響應式狀態 (`products`)，更新響應式狀態，我們的樣版就要重畫，我們的響應式狀態 (`products`) 就不是 `null`，他是一個陣列，在 `v-else`，所以就用 `for` 迴圈，把陣列資料全部渲染成一個 `div` 結構，再搭配 `css` 設定，讓它看起來相當的舒服。



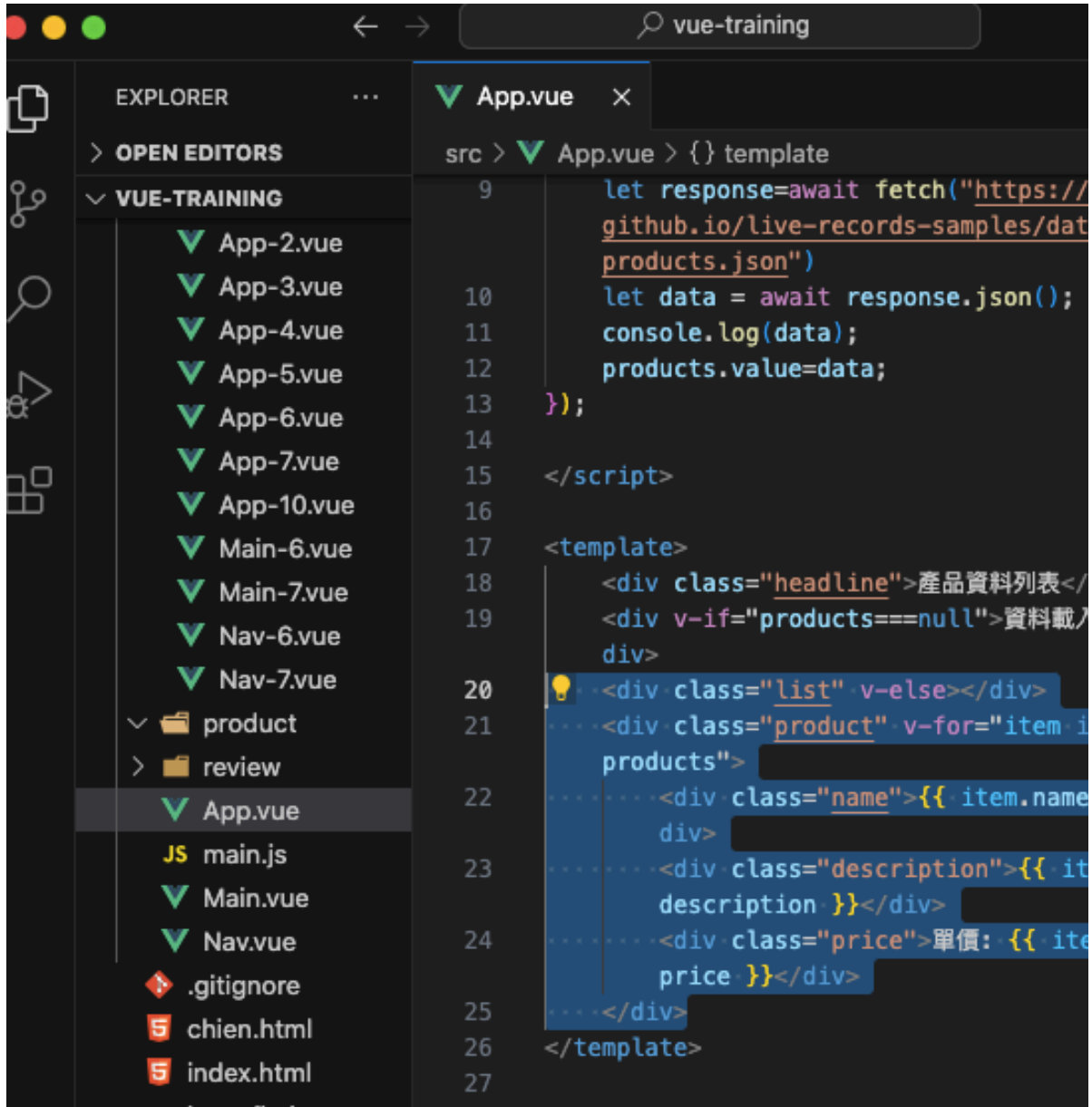
```
src > App.vue > {} script setup
1
2 <script setup>
3   // 後端提供的資料串接網址: https://cwpeng.github.io/
4   live-records-samples/data/products.json
5   import { onMounted, ref } from 'vue';
6   let products = ref(null);
7   //組件掛載完成後，開始呼叫 fetch 串接後端的資料
8   onMounted(async ()=>{
9     let response=await fetch("https://cwpeng.github.io/
10    live-records-samples/data/products.json")
11    let data = await response.json();
12    console.log(data);
13    products.value=data;
14  });
15 </script>
16
17 <template>
18   <div class="headline">產品資料列表</div>
19   <div v-if="products===null">資料載入中</div>
20   <div class="list" v-else></div>
21   <div class="product" v-for="item in products">
22     <div class="name">{{ item.name }}</div>
23     <div class="description">{{ item.description }}</div>
24     <div class="price">單價: {{ item.price }}</div>
25   </div>
26 </template>
```



5. 最後 要做組件的切割：

為什麼要做組件切割，因為程式碼越來越多，需要分工，所以要切割

首先要把「產品列表」給切割出去：

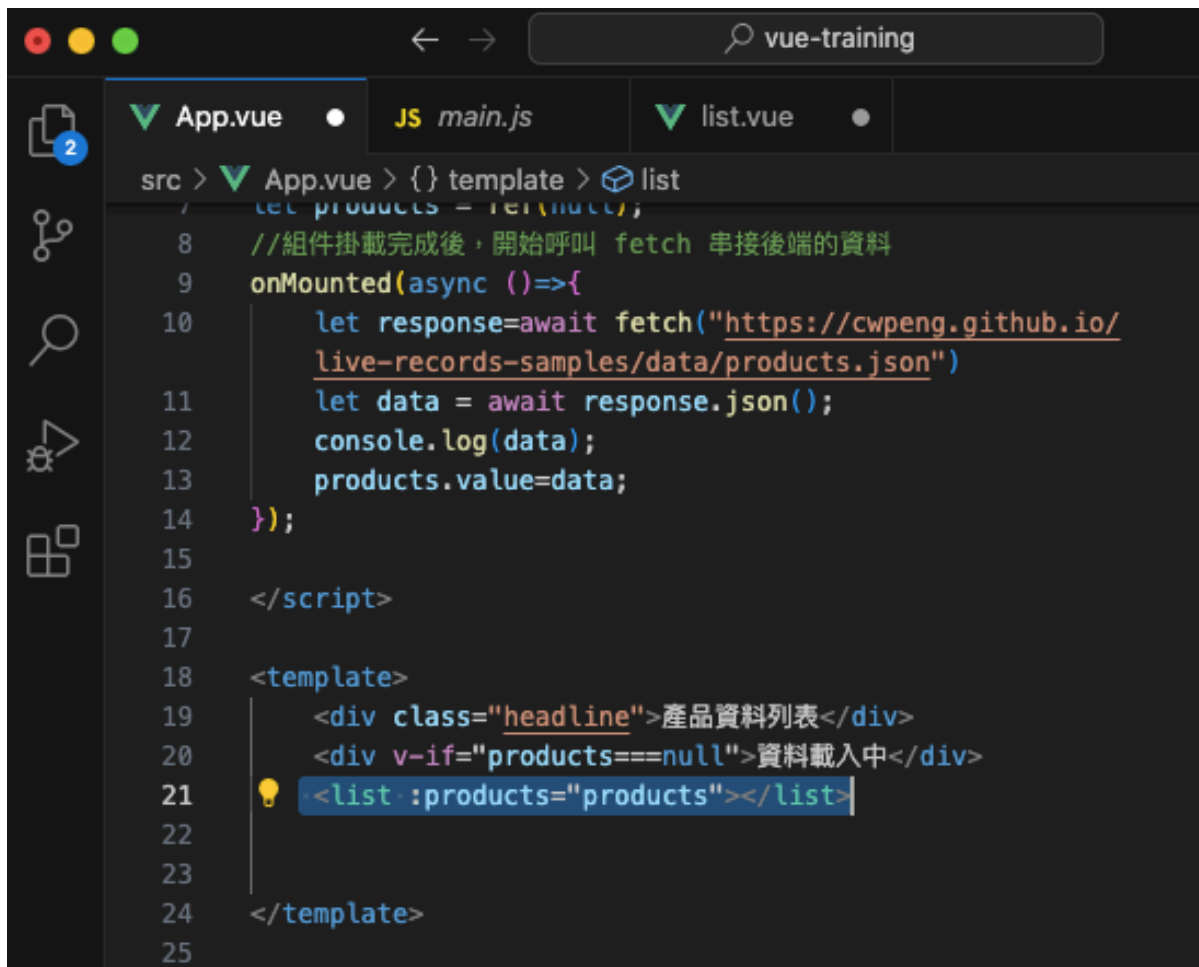


先建立一個 list.vue 的檔案

在 list 組件中的 products，這個響應式狀態是放在 App 組件中，那要怎樣傳遞到 list 組件，就要使用自訂的屬性來做傳遞，在自訂標籤，自訂義一個屬性叫 products，它要對應到響應式狀態，然後要把響應式狀態的資料傳遞到 list.vue，

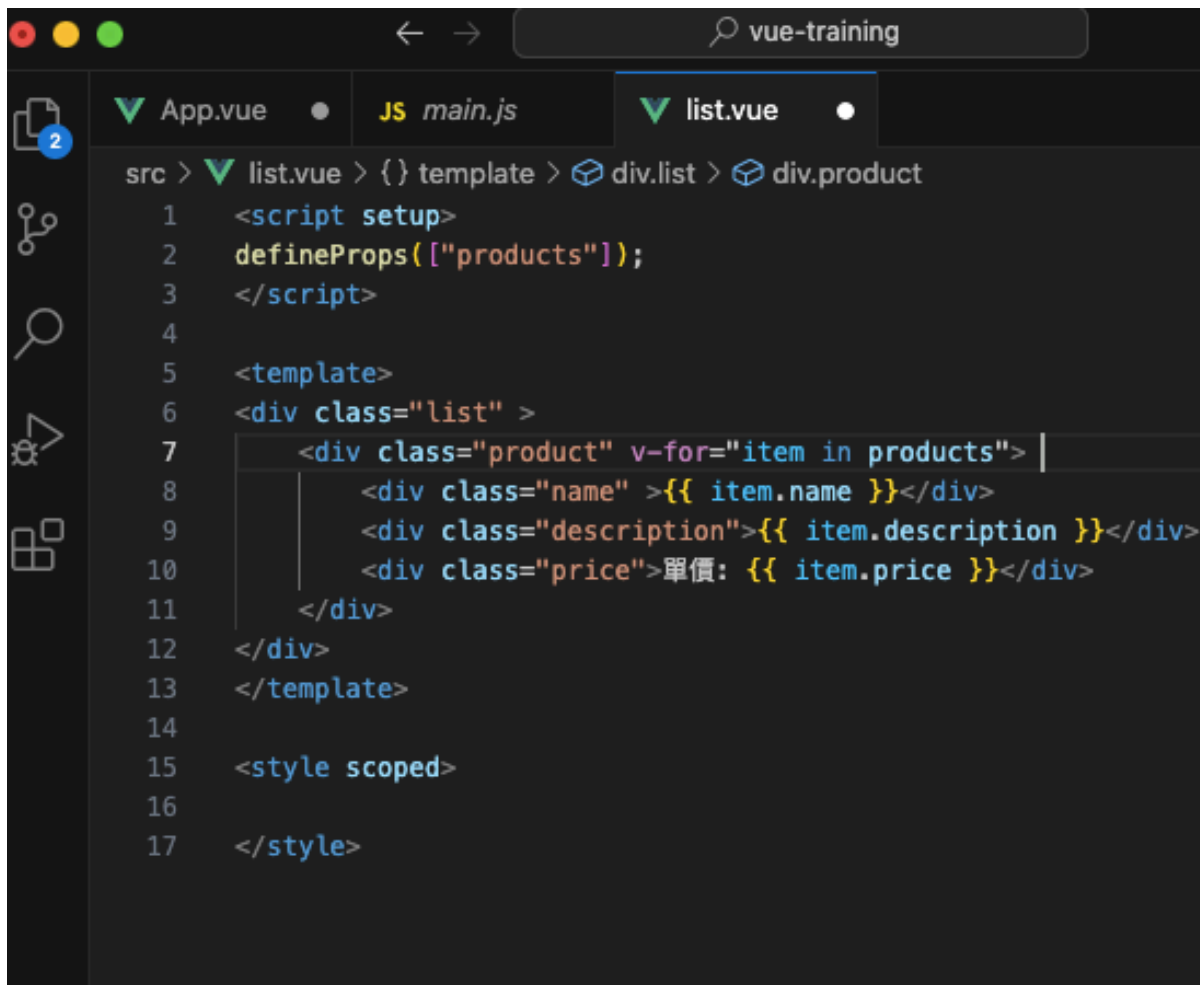
所以在屬性前面要加一個冒號 v-bind 的簡寫(:products; “products”)，這代表綁定一個動態資料的概念。

然後在 list.vue 用 defineprops([products])，把products 訂義起來，接收上層組件傳進來的資料，



The image shows a web browser window at the top with the address bar displaying "vue-training". Below the browser is a code editor with three tabs: "App.vue" (active), "main.js", and "list.vue". The code editor shows the following code:

```
src > App.vue > {} template > list
7   let products = ref(null);
8   //組件掛載完成後，開始呼叫 fetch 串接後端的資料
9   onMounted(async ()=>{
10     let response=await fetch("https://cwpeng.github.io/
11     live-records-samples/data/products.json")
12     let data = await response.json();
13     console.log(data);
14     products.value=data;
15   });
16   </script>
17
18   <template>
19     <div class="headline">產品資料列表</div>
20     <div v-if="products===null">資料載入中</div>
21     ⚡ <list :products="products"></list>
22
23   </template>
24
25
```



```
src > list.vue > {} template > div.list > div.product
1  <script setup>
2    defineProps(["products"]);
3  </script>
4
5  <template>
6    <div class="list" >
7      <div class="product" v-for="item in products"> |
8        <div class="name" >{{ item.name }}</div>
9        <div class="description">{{ item.description }}</div>
10       <div class="price">單價: {{ item.price }}</div>
11     </div>
12   </div>
13 </template>
14
15 <style scoped>
16
17 </style>
```

切割目的：可分工合作。
運作成功：

產品資料列表

Pixel 7

最新的 Pixel 手機。

單價: 17700

Chromecast

在大螢幕上播放喜歡的影片。

單價: 1500

Pixel Book

最新的 Chromebook 產品。

單價: 12000

Elements

top

No Issues

3 messages

3 user me...

No errors

No warnings

1 info

2 verbose