Hola Miguel!

Ya estoy revisando el código, veo que pudiste usar websockets y aplicaste los conocimientos de la clase, bien!

Mencionas que no sabes como aplicar **fetch,** no te preocupes, el uso de fetch lo considero opcional para este desafío, pero sería bueno que lo investigues. Te brindaré igualmente un ejemplo al final del presente feedback.

Entonces te sugiero que no pienses en usar fetch inicialmente, la solución lo puedes hacer todo con **websockets**,

1. El contenido principal de html no debe estar en public, estaría en views.

2. Los archivos formProductos y tablaPRoductos no serían necesarios en este caso.

3. La tabla lo tendrás que armar en front, ejemplo:

En tu index tendrías algo como esto:

<table id="productsContainer" class="table">

</table>

y luego en tu js lo podrías pintar de la siguiente forma: recuerda obtener el elemento por su id

socket.on('producto', productos => {

const productosHTML = productos

``` .map(pro => ````

<tr class="tr2">

<td class="td1">${pro.title}</td>

<td class="td2">${pro.price}</td>

<td class="td3"><img src="${pro.thumbnail}"></td>

</tr>`)

.join('')

productsContainer.innerHTML = productosHTML

})

Como ves, inicialmente no usarás handlebars para tratar los datos, solo para enviar el index. Con esto ya tienes aprobado el desafío.

Ahora el siguiente nivel es usar handlebars para tratar los datos, en este caso te brindaré un ejemplo punto a punto:

1. En la carpeta **public** tendrías que guardar la estructura en handlebars, por ejemplo para el listado: (Estoy usando bootstrap para los estilos)

Archivo en: **/public/productList.handlebars**

{{#if products}}

<div class='table-responsive'>

<table class='table table-hover'>

<thead>

<tr>

<th>NOMBRE</th>

<th>PRECIO</th>

<th>FOTO</th>

</tr>

</thead>

<tbody>

{{#each products}}

<tr>

<td>{{{title}}}</td>

<td>{{{price}}}</td>

<td width='100px'>

<img src='{{{thumbnail}}}' alt='{{{title}}}' width='100px' />

</td>

</tr>

{{/each}}

</tbody>

</table>

</div>

{{else}}

<div class='alert alert-danger'>No hay productos</div>

{{/if}}

2. Obtener la plantilla del listado con fetch en el front, y usar en este caso handlebars con la CDN para hacer el render.

* 2.1 Usar el CDN de handlebars dentro del index.html: (El código es con PUG, pero lo importante es la ruta. Igualmente puedes revisar sus docs)

script(src="[cdn.jsdelivr.net/npm/handlebars@latest/dist/handlebars.js")](https://cdn.jsdelivr.net/npm/handlebars@latest/dist/handlebars.js%22)%60%60%60)

[CDN.JSDELIVR.NET](https://cdn.jsdelivr.net/npm/handlebars@latest/dist/handlebars.js)

2.2 Usar websocket para pasar la data de los productos al método para hacer render con handlebars.

socket.on(keys.PRODUCTS, renderProductList\_handlebars)

2.3 El método que trabajaría con fetch y haría el render con handlebars.

function renderProductList\_handlebars(data) {

fetch(`${API}/productList.handlebars`)

.then((res) => res.text())

.then((res) => {

const template = Handlebars.compile(res)

const html = template({ products: data })

tableProducts.innerHTML = html

})

}

Ahí API = localhost:8080 en este caso.

Para que entiendas mejor, primero investiga como funciona fetch, espero te sirva!  
  
Recuerda que no estoy considerando fetch obligatorio!  
  
Creo que lo podriamos ver en un after junto a otras dudas de tus compañeros, estaba coordinando con los tutores ya que me gustaría dictar un after, espero pueda hacerlo luego de las fiestas