Actividad 9
(detalles de implementación)

Tecnologías Web

Profesor: Juan Carlos Conde Ramírez

Explorador de Soluciones

Los scripts del *backend* ubicados en:

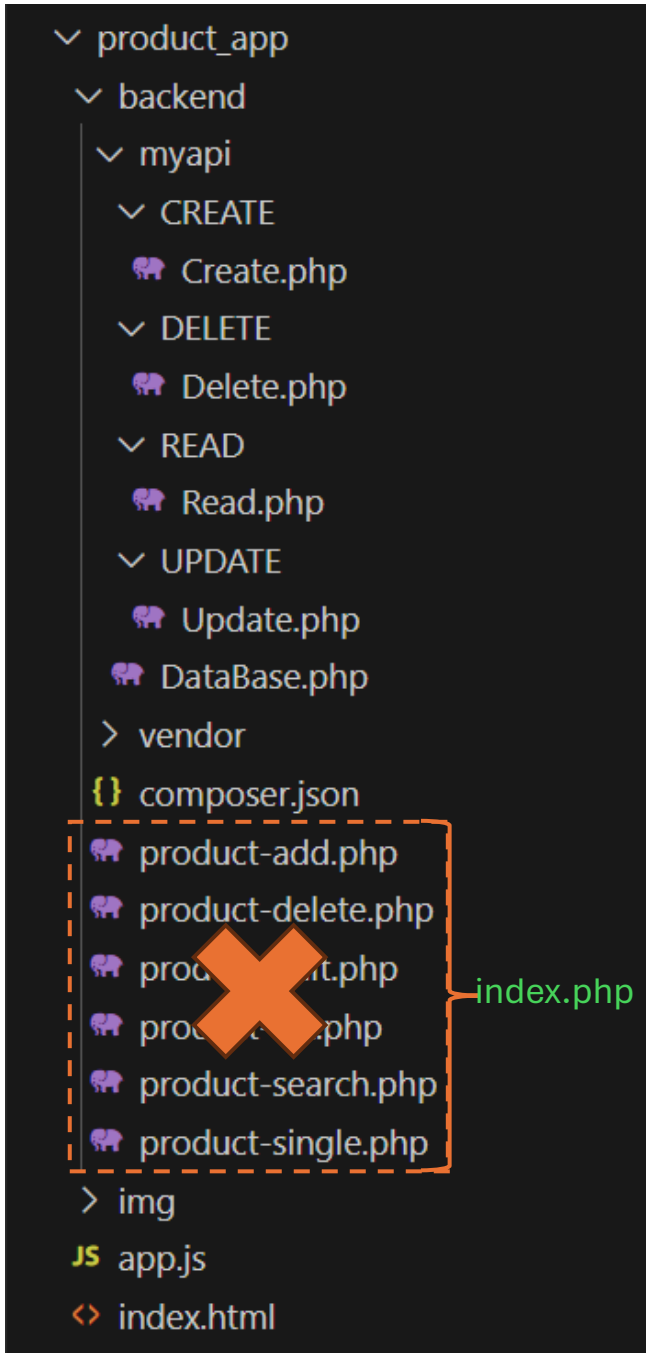
`a09/producto_app/backend`

, que originalmente son llamados desde el *frontend* (`app.js`)), deberán eliminarse y sustituirse por un único archivo que tenga la implementación del servicio Web (Rest API) de nombre `index.php` y que haga uso de dos cosas:

1. Las clases propias (Create, Read, Update y Delete) ubicadas en:

   `a09/producto_app/backend/myapi`

2. *Slim*, el micro-framework para implementar Rest APIs.



Routing para operaciones del API Rest

- Considera que para usar el servicio Web se debe acceder desde `http://localhost/...`

Routing	Método HTTP	Equivalencia (*backend/*)	Respuesta HTTP 20x
backend\product\<id>	GET	product-single.php	JSON con datos del producto cuyo id se pasa al final de la URL en <id>
backend\products	GET	product-list.php	Lista de productos no eliminados en JSON
backend\products\<search>	GET	product-search.php	Lista de productos no eliminados con coincidencias en id o nombre o marca o detalles en JSON

backend\product	POST	product-add.php	<pre>{
			"status": "success",
			"message": "operación realizada correctamente"
			}</pre> NOTA: el mensaje puede variar a tu gusto.

backend\product	PUT	product-edit.php	
backend\product	DELETE	product-delete.php	

IMPORTANTE: nota que la ruta o *routing* para “agregar”, “editar” o “borrar” ¡Es la MISMA!; sólo cambia el método HTTP a usar, el código HTTP y mensaje de respuesta.

Ejemplo de uso del lado del *Frontend*

- Archivo: `app.js`
- Función : `listarProductos()`
- URL del servicio:

app.js

product_app > app.js > ready() callback > listarProductos > $.get("http://localhost/tecweb/actividades/a09/backend/products") callback

```
10  $(document).ready(function () {
17      function listarProductos() {
18          $.get("http://localhost/tecweb/actividades/a09/backend/products", function (data) {
19              console.log("Tipo de dato recibido:", typeof data);
20              console.log("Respuesta del servidor:", data);
21              let productos = JSON.parse(data);
22              let template = "";
23              productos.forEach(producto => {
24                  let descripcion = `
25                      <li>precio: ${producto.precio}</li>
26                      <li>unidades: ${producto.unidades}</li>
27                      <li>modelo: ${producto.modelo}</li>
28                      <li>marca: ${producto.marca}</li>
29                      <li>detalles: ${producto.detalles}</li>
30                  `;
31                  template += `
32                      <tr productId="${producto.id}">
33                          <td>${producto.id}</td>
34                          <td>
35                              <a href="#" class="product-item">${producto.nombre}</a>
36                          </td>
37                          <td><ul>${descripcion}</ul></td>
38                          <td><button class="product-delete btn btn-danger">Eliminar</button></td>
39                      </tr>
40                  `;
41              });
42              $("#products").html(template);
43          });
44      }
```

"http://localhost/tecweb/actividades/a09/backend/products"

Consideraciones, I

- Para consumir el método PUT o DELETE con jQuery, se debe utilizar la función `$.ajax()` y se define el valor del parámetro `type` como PUT. También se debe proporcionar la URL y los datos que se enviarán en la solicitud. El código base para esto es el siguiente:

JavaScript

```javascript
$.ajax({
    url: 'tu_url',
    type: 'PUT',
    data: { sus_datos },
    success: function(respuesta) {
        console.log(respuesta);
    },
    error: function(xhr, estado, error) {
        console.log(xhr.responseText);
    }
});
```

Consideraciones, II

Explicación:

1. `$.ajax()`: Esta función de jQuery se utiliza para realizar solicitudes HTTP asíncronas.
2. `url`: Especifica la URL del *endpoint* al que se envía la solicitud PUT.
3. `type`: Se establece a PUT para indicar que la solicitud es de tipo PUT.
4. `data`: Un objeto que contiene los datos que se enviarán en el cuerpo de la solicitud.
5. `success`: Una función de devolución de llamada que se ejecuta si la solicitud es exitosa.
6. `error`: Una función de devolución de llamada que se ejecuta si ocurre un error en la solicitud.
7. `console.log(respuesta)`: Imprime la respuesta del servidor en la consola.
8. `console.log(xhr.responseText)`: Imprime la respuesta del error en la consola.

Ejemplo con datos tipo JSON:

- Supongamos que tienes un formulario con algunos campos que quieres actualizar. Puedes obtener los valores de estos campos usando jQuery y luego enviarlos en la solicitud PUT:

```
$(document).ready(function() {
     $('#actualizarForm').submit(function(event) {
          event.preventDefault(); // Evitar que el formulario se envíe de forma normal
          var data = {
               nombre: $('#nombre').val(),
               apellido: $('#apellido').val(),     // ... otros campos
               };

          $.ajax({
               url: 'https://tu_api/actualizar_usuario',
               type: 'PUT',
               data: data,
               success: function(respuesta) {
                    console.log(respuesta);        // Actualizar la interfaz de usuario
               },
               error: function(xhr, estado, error) {
                    console.log(xhr.responseText); // Mostrar un mensaje de error
               }});
     });
});
```

Ejemplo con datos tipo JSON:

En este ejemplo:

1. Se captura el evento de envío del formulario (`#actualizarForm`).
2. Se obtienen los valores de los campos del formulario y se crean un objeto data.
3. Luego, se utiliza `$.ajax()` para enviar los datos al servidor mediante una solicitud PUT.

Documentación

Para una mejor comprensión se recomienda visitar la documentación de la función `$.ajax()`:

Sitio oficial de jQuery (Documentación)
📎 https://api.jquery.com/jQuery.ajax/

TutorialesPoint (Guía paso a paso en español)
📎 https://www.tutorialspoint.com/jquery/ajax-jquery-ajax.htm

W3Schools (Ejemplo sencillo)
📎 https://www.w3schools.com/jquery/ajax_ajax.asp