



PROYECTO INTEGRADO  
Desarrollo de Aplicaciones Web  
**Raulmarket**

## INDICE

PLANTEAMIENTO Y BRIEFING DEL PROYECTO .....	3
HERRAMIENTAS Y/O TECNOLOGÍAS .....	4
FUNCIONALIDAD INTERNA .....	6
DESARROLLO DEL PROYECTO Y MOCKUP .....	9
CODIFICACIÓN Y EXPLOTACIÓN DE RAULMARKET .....	20
PASARELA DE PAGO .....	22
CONCLUSIONES FINALES .....	23
REFERENCIAS Y BIBLIOGRAFÍA .....	24

## PLANTEAMIENTO Y BRIEFING DEL PROYECTO

**Raulmarket** es una aplicación web que tiene el objetivo de mostrar una serie de productos (de todo tipo) de los cuales el cliente podrá ir eligiendo cada uno de ellos, almacenar en su carrito y posteriormente realizar la compra.

Dispone de una interfaz principal que pedirá introducir una cuenta que debe estar registrada en el sistema.

Aunque también nos ofrece la posibilidad de registrarnos en ella (siempre otorgando el rol de usuario normal).

Esta aplicación se compone de dos tipos de roles de usuario:

- **Usuario**, es el rol que se le otorgará al usuario predeterminado, que tendrá los permisos justos y necesarios para ver, almacenar en el carrito y realizar la compra de los productos.

- **Administrador**, es el rol que tienen ciertos usuarios privilegiados.

Podrán ver, añadir, editar y eliminar cada uno de los productos.

Además de poder añadir otro usuario al sistema (también con el rol de administrador) y eliminarlos.

## HERRAMIENTAS Y/O TECNOLOGÍAS

Para la realización de este proyecto se han requerido varias tecnologías para las distintas partes de la aplicación.

### DISEÑO

**CSS:** «hojas de estilo en cascada» (Cascading Style Sheets). Lenguaje que maneja el diseño y presentación de las páginas web.

**Bootstrap:** Framework CSS desarrollado por Twitter en 2010, para estandarizar las herramientas de la compañía.

Su principal objetivo es facilitar el proceso de desarrollo de los sitios web responsivos y orientados a los dispositivos móviles.

### FUNCIONALIDAD

**Javascript:** Lenguaje de programación que convierte nuestra aplicación en una web más dinámica.

**Ajax:** Tecnología utilizada para crear páginas web asíncronas.

**Php:** Lenguaje de programación interpretado del lado del servidor.

## BASE DE DATOS

**PhpMyAdmin:** Aplicación web que sirve para administrar bases de datos.

En **Raulmarket** no se utiliza una base de datos muy compleja, realmente se utilizan nada más que dos tablas:

Tabla	Acción
<input type="checkbox"/> <b>articulos</b>	Examinar            Estructura            Buscar            Insertar            Vaciar            Eliminar
<input type="checkbox"/> <b>usuarios</b>	Examinar            Estructura            Buscar            Insertar            Vaciar            Eliminar

En la tabla de **artículos** tendremos varias columnas, entre ellos el `id_articulo` (código único que hará referencia a ese artículo en concreto), su nombre, la sección a la que pertenece, precio, descripción y la ruta de la imagen (almacenada en una carpeta).

		<code>id_articulo</code>	<code>nombreakiculo</code>	<code>seccion</code>	<code>precio</code>	<code>descripcion</code>	<code>imagen</code>
<input type="checkbox"/>	Editar  Copiar  Borrar	15	Martillo	3	13	Martillo de alta calidad con mango antideslizante.	view/template/img/martillo.png
<input type="checkbox"/>	Editar  Copiar  Borrar	17	Barra de pan	1	1	Hecha con la mejor harina esta mañana	view/template/img/barra de pan.png
<input type="checkbox"/>	Editar  Copiar  Borrar	18	Bolsa de chuches	1	1	Escogidas por los mejores selectores de chuches.	view/template/img/bolsa de chuches.png

En la tabla de **usuarios**, podemos observar que tenemos ese id único para cada usuario, el rol (puede ser administrador o usuario), nombre, nombre de usuario y la contraseña encriptada (con MD5).

		<code>id</code>	<code>rol</code>	<code>nombre</code>	<code>usuario</code>	<code>pass</code>
<input type="checkbox"/>	Editar  Copiar  Borrar	16	Administrador	Raul	raul	202cb962ac59075b964b07152d234b70
<input type="checkbox"/>	Editar  Copiar  Borrar	26	Usuario	ss	ss	3691308f2a4c2f6983f2880d32e29c84

## FUNCIONALIDAD INTERNA

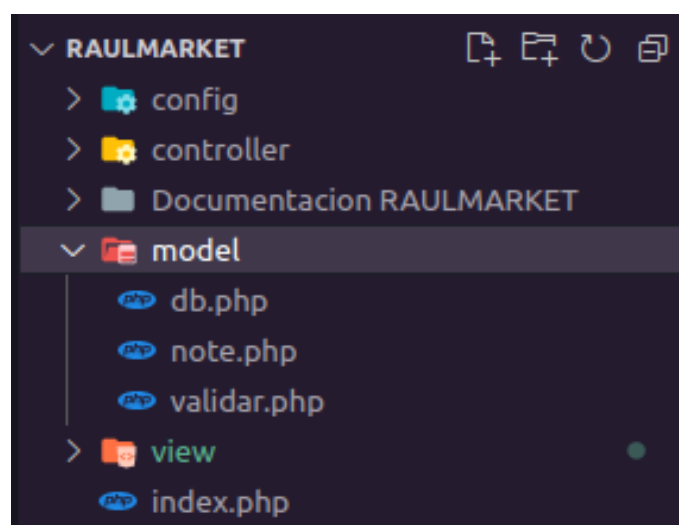
Este proyecto sigue el patrón MVC (Modelo Vista Controlador), la idea de este patrón es separar nuestro sistema en tres capas.

### MODELO

Se encarga de todo lo que tenga que ver con la persistencia e interacción con la base de datos.

En este caso, tenemos tres modelos:

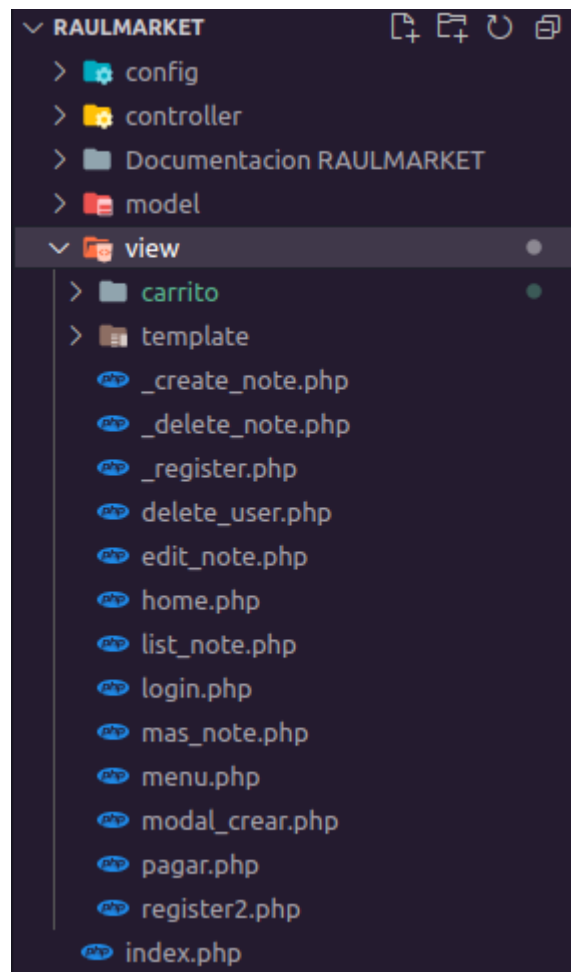
- db.php**: Contiene todos los datos relacionados con la interacción con la base de datos.
- **note.php**: Contiene todas las funciones de búsqueda, guardado, creación, etc, de los productos.
- **validar.php**: Contiene todas las funciones de gestión con los usuarios del sistema.



## VISTA

Muestra al usuario todo lo que se pueda visualizar.

En este caso, disponemos de muchas vistas para las distintas funcionalidades de nuestra aplicación. Incluido las vistas que ya no utilizamos (tienen un “\_” delante).

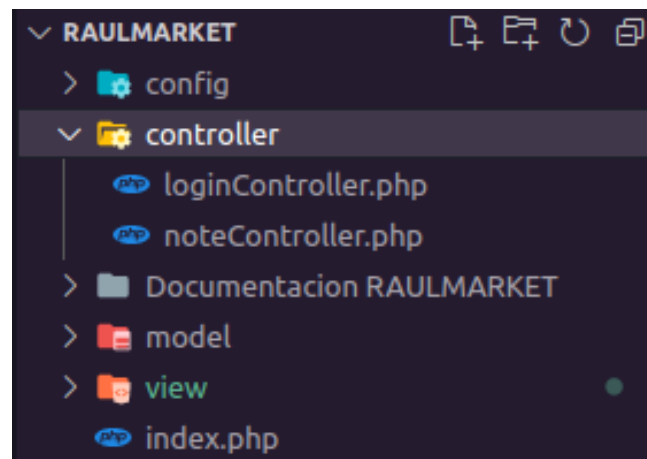


## CONTROLADOR

Pide información al modelo y la implanta sobre la vista.

En este caso tenemos dos controladores:

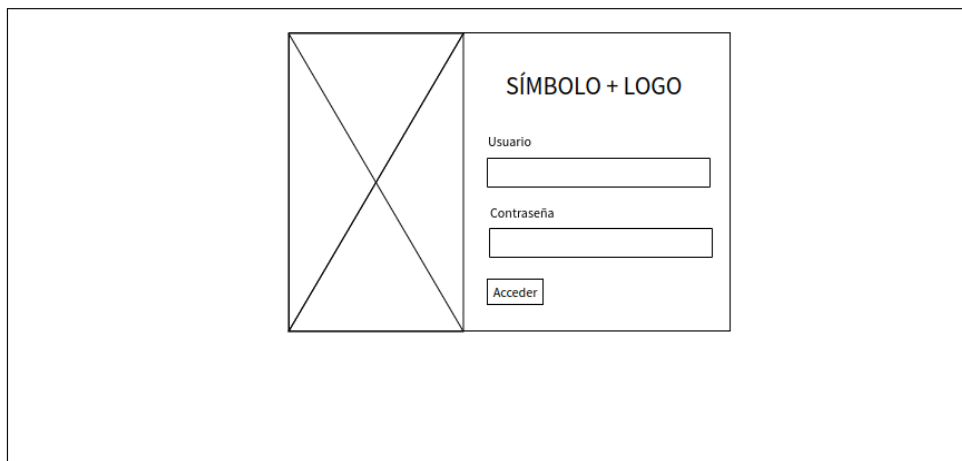
- **loginController.php**: Controla todo lo relacionado con la gestión de usuarios del login y register.
- **noteController.php**: Controla todo lo relacionado con la gestión de productos del sistema.





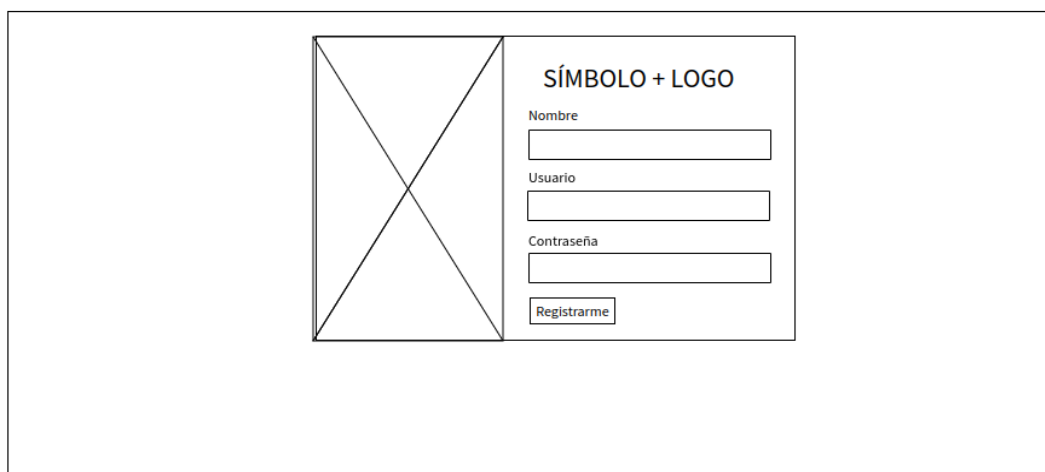
## DESARROLLO DEL PROYECTO Y MOCKUP

### LOGIN



A login form mockup enclosed in a large rectangular container. On the left side of the container is a square placeholder for a logo, marked with a large 'X'. To the right of this placeholder is a smaller rectangular box containing the text 'SÍMBOLO + LOGO' at the top. Below this text are two input fields: the first is labeled 'Usuario' and the second is labeled 'Contraseña'. At the bottom of this box is a button labeled 'Acceder'.

### REGISTER

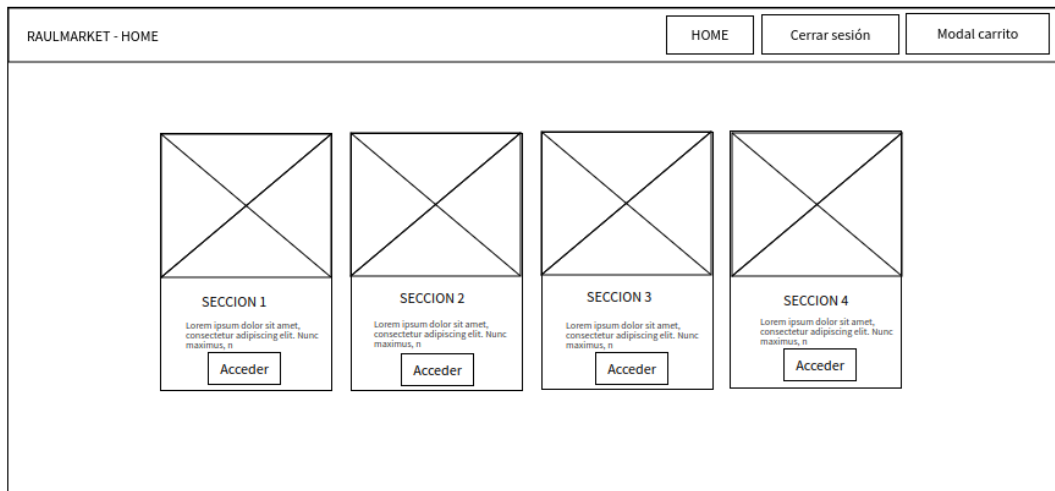


A register form mockup enclosed in a large rectangular container. On the left side of the container is a square placeholder for a logo, marked with a large 'X'. To the right of this placeholder is a smaller rectangular box containing the text 'SÍMBOLO + LOGO' at the top. Below this text are three input fields: the first is labeled 'Nombre', the second is labeled 'Usuario', and the third is labeled 'Contraseña'. At the bottom of this box is a button labeled 'Registrarme'.

La vista (login.php) muestra dos tipos de formularios, el de loguearse y la de registrarse.

## VISIÓN DEL USUARIO

### HOME - USUARIO



Una vez accedido, el usuario contará con un menú sencillo con varios botones y se situará en el **Home**:

- **Home**: Al hacer click redirige hacia la interfaz principal donde se muestran las distintas secciones de productos.
- **Cerrar sesión**: Botón que cierra la sesión del usuario y dirige a login.php.
- **Modal carrito**: Modal que se actualizará automáticamente según se agreguen productos al carrito, mostrando también el total.

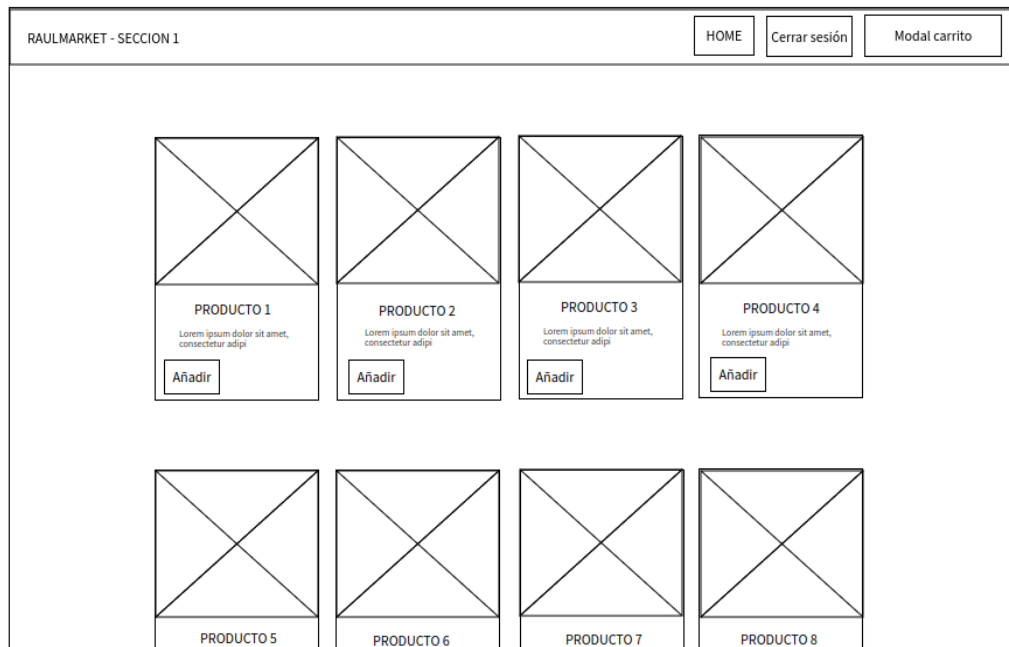
## MODAL CARRITO - USUARIO

RAULMARKET - SECCION 1	HOME	Cerrar sesión	Modal carrito								
			<table border="1"><tr><td>QUESO x 2</td><td>14 €</td></tr><tr><td>Alicates x 1</td><td>2€</td></tr><tr><td colspan="2">Total (EUR) 16€</td></tr><tr><td>Vaciar</td><td>Pagar</td></tr></table>	QUESO x 2	14 €	Alicates x 1	2€	Total (EUR) 16€		Vaciar	Pagar
QUESO x 2	14 €										
Alicates x 1	2€										
Total (EUR) 16€											
Vaciar	Pagar										

El modal del carrito muestra una lista de los productos que se van añadiendo, el recuento total de todos ellos y además dos botones:

- **Vaciar carrito:** Al hacer click se vaciará por completo el carrito.
- **Pagar:** Redirige a la página de pago donde se realiza el mismo.

## SECCION 1 - USUARIO



Al acceder a cualquier sección se mostrarán los distintos tipos de productos con la opción de poder añadirlos al carrito.

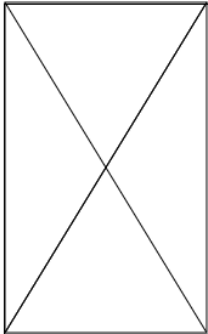
# PAGAR - USUARIO

RAULMARKET - PAGO

HOME

Cerrar sesión

Modal carrito



Seleccione el método de pago

Pagar con PayPal

Pagar con SOFORT

Tarjeta de débito o crédito

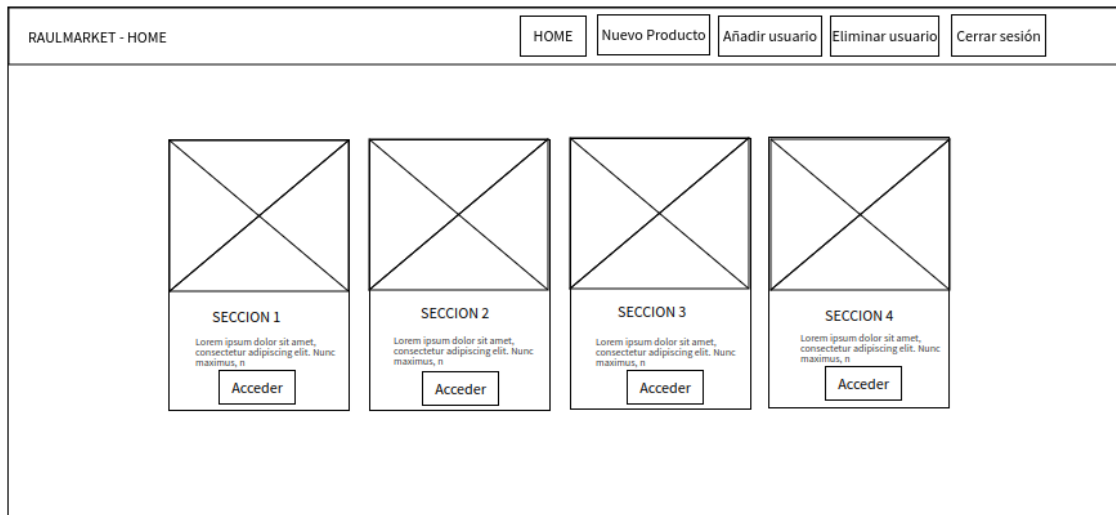
Total: 16€

Vista (pagar.php) donde muestra al usuario el total a pagar con las distintas opciones de pago disponibles, todas ellas desarrollado por PayPal:

- PayPal
- SOFORT
- Tarjeta de débito o crédito

## VISIÓN DEL ADMIN

### HOME - ADMIN



Una vez accedido, el admin contará con un menú más complejo que el usuario normal, dispondrá de las opciones mencionadas anteriormente (menos el carrito) junto a otras tres más y se situará en el **Home**:

- **Crear producto:** Abre un modal con un formulario que añade productos.
- **Añadir usuario:** Redirige a “agregar2.php” que permite agregar usuarios con cualquier tipo de rol.
- **Eliminar usuario:** Redirige a “delete\_user.php”, donde muestra una tabla con todos los usuarios con la posibilidad de eliminarlos.

# CREAR PRODUCTO - ADMIN

The image shows a web application interface for an administrator. At the top, there is a navigation bar with the text "RAULMARKET - SECCION 1" on the left and a series of buttons: "HOME", "Nuevo producto", "Añadir usuario", "Eliminar usuario", and "Cerrar sesión". The "Nuevo producto" button is highlighted with a yellow dot. Below the navigation bar, there is a grid of product cards. A modal form titled "CREAR PRODUCTO" is open in the center, overlaying the grid. The modal contains a form with the following fields: "Nombre Artículo", "Seccion", "Precio", and "Descripción". There are also "Añadir" and "Cerrar" buttons at the bottom of the modal. The background grid shows eight product cards, each with a placeholder image (a square with an 'X'), a title (e.g., "PRODUCTO 1"), a description (Lorem ipsum), and "Editar" and "Eliminar" buttons.

En la opción del menú de “Nuevo producto” nos muestra un modal que contiene un formulario con los datos del producto a incluir, entre ellos la sección, que muestra una lista de las secciones disponibles y donde añadir el producto.

# AÑADIR USUARIO - ADMIN

RAULMARKET - REGISTRO

HOME Nuevo producto Añadir usuario Eliminar usuario Cerrar sesión

SÍMBOLO + LOGO

Nombre

Usuario

Rol

Contraseña

Registrar

En la opción del menú de “Añadir usuario”, muestra un formulario con los datos a rellenar de ese usuario, entre ellos destaca la casilla de “Rol”, que ofrece la posibilidad de que el usuario sea tanto **“usuario”** como **“administrador”**. Por lo que un usuario administrador solamente lo podrá ser gracias a otro administrador.

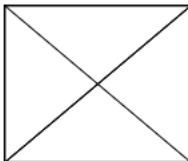
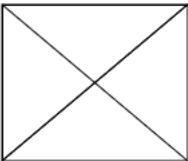
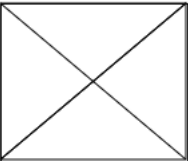
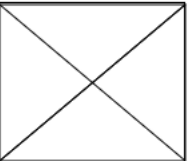
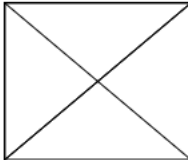
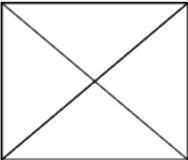
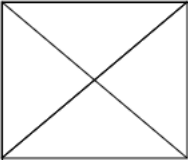



# ELIMINAR USUARIO - ADMIN

RAULMARKET - ELIMINAR USUARIO					HOME	Nuevo producto	Añadir usuario	Eliminar usuario	Cerrar sesión
ID	Nombre	Usuario	Rol	Eliminar					
16	Raul	raul	Administrador	Borrar					
25	Marcos	marcos	Usuario	Borrar					
28	Marta	marta	Usuario	Borrar					

En la opción del menú de “Eliminar usuario”, muestra una tabla con todos los usuarios del sistema, ofreciendo a través de un botón el borrado de ese usuario.

# SECCION 1 - ADMIN

RAULMARKET - SECCION 1		HOME	Nuevo producto	Añadir usuario	Eliminar usuario	Cerrar sesión
 PRODUCTO 1 Lorem ipsum dolor sit amet, consectetur adipiscing <input type="button" value="Editar"/> <input type="button" value="Eliminar"/>	 PRODUCTO 2 Lorem ipsum dolor sit amet, consectetur adipiscing <input type="button" value="Editar"/> <input type="button" value="Eliminar"/>	 PRODUCTO 3 Lorem ipsum dolor sit amet, consectetur adipiscing <input type="button" value="Editar"/> <input type="button" value="Eliminar"/>	 PRODUCTO 4 Lorem ipsum dolor sit amet, consectetur adipiscing <input type="button" value="Editar"/> <input type="button" value="Eliminar"/>			
 PRODUCTO 5	 PRODUCTO 6	 PRODUCTO 7	 PRODUCTO 8			

En la sección ubicada, el administrador tendrá toda la gestión sobre esos productos:

- **Editar:** Redirige a “edit\_note.php”, donde nos muestra un formulario (con los datos rellenados de ese producto a editar) y ofreciendo la posibilidad de cambiarlos.
- **Eliminar:** A través de la tecnología AJAX, nos elimina ese producto de forma inmediata y mostrando el cambio realizado sin la necesidad de tener que recargar la página.

# EDITAR PRODUCTO - ADMIN

Editar producto

HOME

Nuevo producto

Añadir usuario

Eliminar usuario

Cerrar sesión

Martillo

3

13

Descripción

Examinar ..

Guardar

Cancelar

Al pulsar “editar” nos redirige a “edit\_note.php”, donde nos muestra un formulario con los datos rellenados de ese producto y con posibilidad de añadir una imagen.

## CODIFICACIÓN Y EXPLOTACIÓN DE RAULMARKET

Como se ha explicado anteriormente, utilizamos el protocolo MVC y esto da pie a que nuestro archivo “**index.php**” actúe como emisor y receptor del controlador.

Este archivo ejecuta datos por defecto (página de login) y a raíz de aquí, recibirá otros datos que ejecutarán las distintas funciones del controlador junto a sus vistas.

### Controlador

Un ejemplo podría ser la función “list()” de “noteController.php”. Se puede observar como:

- Continuamos la sesión con “session\_start()”.
- Establece el título de la página.
- Devuelve el resultado de la función “getNotes()” (ubicada en el modelo note.php).

```
/*
-FUNCIÓN LISTA
Título de página: RAULMARKET - PRODUCTOS
Devuelve un new Note() con la función "getNotes()" que muestra todos los productos.
*/
0 references | 0 overrides
public function list()
{
    session_start();
    $this->page_title = 'RAULMARKET - HOME';
    return $this->noteObj->getNotes();
}
```

## Modelo

Podemos ver que la función “getNotes()” realiza la interacción con la base de datos.

- Ejecuta otra función (la cual es el conector con nuestra base de datos).
- Guarda en una variable la sentencia a realizar (en este caso mostrar todos los datos).

```
/*  
-FUNCIÓN QUE OBTIENE LOS PRODUCTOS  
Ejecuta la función que conecta con la base de datos.  
Selecciona todos los productos de la tabla (articulos).  
Devuelve $stmt que contiene un array con todos los resultados.  
*/  
3 references | 0 overrides  
public function getNotes()  
{  
    $this->getConnection();  
    $sql = "SELECT * FROM " . $this->table;  
    $stmt = $this->connection->prepare($sql);  
    $stmt->execute();  
  
    return $stmt->fetchAll();  
}
```

- Prepara la sentencia y se ejecuta.
- Devuelve un array con todas las filas de la búsqueda realizada.

## PASARELA DE PAGO

Para la realización de la pasarela de pago me he apoyado en el kit de herramientas (SDK) que ofrece el propio PayPal.

Primeramente se incluye el script :

```
<!-- Enlace que inserta el script de PayPal -->
<script src="https://www.paypal.com/sdk/js?client_id=AYJWx7jCybBGUA4B9DShmLwh0h4NVnj1sgDTIAU8Llkfdg2rWrGdp1TPSCZzRjp2lt727CDFmLics0LL&currency=EUR" ></script>
```

Después, utilizamos el id “paypal-button-container”, que nos crea la sección con las distintas opciones de pago que nos ofrece PayPal.

```
<div id="paypal-button-container">
  <h4 class="text-center text-white mb-5 pb-3">Seleccione el método de pago</h4>
</div>
```

Y lo demás consiste en darle su funcionalidad y estilo.

```
paypal.Buttons({
  //Estilo de los botones de paypal.
  style: {
    color: 'blue',
    label: 'pay',
    shape: 'pill'
  },

  //Función que se ejecuta para crear el pedido
  createOrder: function(data, actions) {
    return actions.order.create({
      purchase_units: [{
        amount: {
          //Valor del pedido
          value: precioTotal
        }
      }]
    });
  },

  //Función que se ejecuta cuando se ha realizado el pedido
  onApprove: function(data, actions) {
    actions.order.capture().then(function(detalles) {
      var miElemento = document.getElementById('aviso');
      miElemento.style.backgroundColor = "#5cb85c";
      miElemento.innerHTML = '<h3>Pago Realizado con éxito</h3>';
    });
  },
});
```

## CONCLUSIONES FINALES

Para la realización del proyecto tenía pensado en un principio extender y mejorar “Raulapp”, el proyecto realizado el curso anterior.

Sobretudo por el hecho de haber obtenido algo de experiencia en Laravel y Tailwind CSS.

Pero debido a la forma de trabajar en mi empresa (Área 14), usando código de forma nativa y usando el protocolo MVC, decidí hacerlo de esta manera y afrontar un nuevo proyecto por completo.

En relación a la funcionalidad, he incluido multitud de tecnologías y herramientas utilizadas hoy en día, como pueden ser el uso de modales, tecnología Ajax (para definir comunicaciones con el servidor sin recargar páginas), uso de roles de usuario, carrito de compra, o incluso la utilización de un SDK para la realización del pago por parte de un cliente.

Optando también por una interfaz intuitiva y visual, he usado Bootstrap, como framework (el más utilizado en la actualidad).

Por último, he perfeccionando el diseño para que llegue a ser “responsive”, de modo que sería apto para entorno móvil y Tablet.

## REFERENCIAS Y BIBLIOGRAFÍA

- <https://www.jairogarciarincon.com/clase/programacion-orientada-a-objetos-en-php/patron-mvc>
- <https://www.php.net/manual/es/index.php>
- <https://www.hostinger.es/>
- <https://stackoverflow.com/>
- <https://www.youtube.com/>
- <https://www.lawebdelprogramador.com/foros/JavaScript/>
- <https://getbootstrap.com/docs/4.0/utilities/>
- <https://www.w3schools.com/>
- <https://lenguajejs.com/javascript/peticiones-http/ajax/>