**IMPLEMENTACIÓN**

PÁGINAS PHP



Además del esquema que aquí se muestra, existen tres módulos más:

1. functions.php: Fichero PHP que recoge funciones de utilidad que se emplean en todas las demás páginas, pero al cual no se puede acceder.
2. header.php: Módulo que se incluye en la cabecera de todas las páginas para generar un header común.
3. footer.php: Módulo que se incluye al final de todas las páginas para generar un footer común.

Es necesario mencionar que, en cualquier momento, el usuario puede realizar las siguientes operaciones, independientemente de donde se encuentre:

1. Volver al index.php haciendo clic en el icono del header.
2. Loguearse o registrarse haciendo clic en el botón apropiado del header.
3. Ver su cuenta, en caso de que esté logueado, usando el botón apropiado del header.
4. Ver su carrito, en caso de que contenga algún artículo, usando el botón apropiado del header.

Siempre que un usuario sea direccionado a login.php, tiene la opción de registrarse si no lo está. Una vez registrado, el sistema le devolverá a la operación que estuviera realizando:



ENTRADAS:

Una vez se haya logueado o registrado un usuario a través de login.php o register.php, el servidor web guardará su correo, user\_id, y si es un administrador, mediante una variable de sesión, que seguirá activa hasta que el usuario cierre la sesión o el navegador. Así, el resto de las páginas de la aplicación (en concreto, account.php, addresses.php, paymentMethods.php y orders.php) pueden hacer uso de este valor en cualquier momento para realizar sus consultas.

1. account.php: Esta página incorpora muchos intercambios de información, en concreto:
   1. A addresses.php se pasa, por medio de formulario post, si el usuario quiere modificar o borrar una dirección (junto con los datos de la misma, guardado en una variable de sesión), y, en otro formulario, si el usuario ha seleccionado crear una.
   2. A paymentMethods.php se pasa, por medio de formulario post, si el usuario quiere modificar o borrar un nuevo método de pago (junto con los datos del mismo, guardado en una variable de sesión), y, en otro formulario, si el usuario ha seleccionado crear uno.
   3. A sí misma, se pasa, por medio de formulario post, los nuevos datos del cliente cuando este los modifica. También con un formulario post, si el usuario elige borrar su cuenta.
2. addresses.php: Se pasa, a sí misma, un formulario post con los datos introducidos por el usuario para su validación.
3. paymentMethods.php: Se pasa, a sí misma, un formulario post con los datos introducidos por el usuario para su validación.
4. manageProduct.php: Tiene dos formularios post con los que el administrador puede interactuar con esta misma página:
   1. Un formulario post para crear un nuevo producto.
   2. Un formulario post para modificar un producto ya existente. La página genera uno por producto.
5. index.php: Contiene dos formularios post que indican a newProduct.php si el usuario desea construir un ordenador o un móvil.
6. newProduct.php: Contiene dos formularios post, uno para ordenadores y uno para móviles, aunque nunca se muestran los dos a la vez. Los datos de entrada de este formulario se guardan en una variable de sesión con la que se implementa la funcionalidad del carrito. Esta variable de sesión perdura hasta que el usuario cierre la sesión o el navegador.
7. orders.php: Emplea la variable de sesión creada en la página anterior para mostrar un resumen del pedido y pedir confirmación al usuario. Esta confirmación se envía por formulario get a sí misma para procesar los datos.
8. register.php: Emplea un formulario post, enviado a sí misma, para validar y procesar los datos introducidos por el usuario.
9. header.php: Emplea las variables de sesión de usuario y carrito para mostrar los botones apropiados con los que acceder a dichas funcionalidades. Aquí es donde se controla que un usuario no logueado no pueda acceder a su cuenta, o que no se pueda manipular un carrito vacío.

LIBRERÍAS

El fichero functions.php contiene una librería con funciones de utilidad empleadas en diversas páginas de la aplicación:

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Propósito | Argumentos | Valor devuelto |
| doesCountryExist | Comprueba que el país introducido por el usuario existe realmente | Código ISO ALPHA-3 del país | Verdadero o falso |
| getCountryName | Nombre completo de un país a partir de un código | Código ISO ALPHA-3 del país | Nombre del país. |
| printErrorMessage | Imprime un mensaje de error de servidor | Vector de mensajes de error en texto plano | Vector de mensajes de error formateados |
| createConnection | Crea una conexión a la base de datos | Ninguno | Enlace a la base de datos |
| addressDetails | Devuelve las direcciones asociadas al usuario logueado | Ninguno | Vector de direcciones |
| getAddressForOrder | Devuelve la dirección con un determinado id | Id de la dirección | Dirección con ese id |
| paymentMethodDetails | Devuelve los métodos de pago asociadas al usuario logueado | Ninguno | Vector de métodos de pago |
| validPaymentMethods | Comprueba si un método de pago es válido | Método de pago | Método de pago, si es válido. |
| checkSession | Comprueba si el usuario está logueado, y lo dirige al login si no lo está | Ninguno | Ninguno |
| printWelcome | Imprime la bienvenida y los botones apropiados del header según las variables de sesión | Ninguno | Ninguno |
| getAllComponents | Devuelve todos los componentes de la base de datos | Ninguno | Vector de componentes |

BASE DE DATOS:

La base de datos se ha implementado con el siguiente modelo relacional:



Cada entidad del diseño entidad-relación tiene su tabla, y cada atributo, una columna. Se han relacionado de la siguiente manera:

1. Relaciones muchos-a-muchos: se relacionan creando una tabla intermedia
   1. users y address se relacionan con la tabla user\_address, que guarda un id de cada una de las dos entidades.
   2. custom\_products y components se relacionan con la tabla custom\_products\_components del mismo modo.
2. Relaciones uno-a-muchos:
   1. users y payment\_methods: Un usuario tiene varios métodos de pago, por lo que cada método de pago tiene una clave foránea (FK) del usuario al que pertenece.
   2. orders y custom\_products: Un pedido tiene uno o más productos montables, por lo que estos últimos tienen una FK del pedido al que pertenecen.
   3. payment\_methods, addresses y orders: Un método de pago y una dirección son necesarios para un pedido, por lo que este guarda una FK de cada uno de los anteriores.

Esta implementación se ha desplegado sobre un servidor MySQL 5.3.