E2_FASEIMPLEMPRUE

Ejercicio 1:

Ficheros con funcionalidades/principales generados:

verificarSesionIniciada.php:
Simplemente inicia la sesión y en caso de que exista la variable de sesión user, devuelve true y el usuario, de lo contrario devuelve false.

verificarCorreo.php:
Trabaja de dos maneras, en caso de que se trate de una petición POST comprueba que el código de verificación almacenado en la BBDD coincida con el introducido por el usuario en el formulario, si es así, actualiza el campo "verificado" de la BBDD a True, de lo contrario muestra que no es correcto.

En caso de que no sea una petición POST, comprobará si el usuario ya está verificado, si lo está lo manda a la página de inicio, en caso contrario generará un código de verificación y lo enviará al usuario por correo y mostrará el formulario.

sendmail.php:

Contiene una función que recibe por parámetro el destinatario, asunto y mensaje y envia un correo con esos datos.

search.php:
Recibe por método GET el parámetro de búsqueda, realiza una consulta a la BBDD y todos los resultados que coincidan los devuelve en formato JSON.

saveLike.php:

Recibe el producto y el usuario y registra el "like" en la bbdd, devuelve si ha sido registrado correctamente o en caso de fallar, el por qué.

register.php:
Muestra un formulario de registro con los campos correo, nombre, apellido y contraseña. Cuando se envia, hace una consulta a la bbdd para ver si existe algún usuario con ese correo, en caso de que exista muestra un error y no continua, de lo contrario valida el email, si no es correcto muestra un error también, en caso de que sea valido registra al usuario en la bbdd y lo lleva a la página de verificación de correo.

productos.php (posible refactorización de código):

Página que muestra los productos de la tienda y puede filtrar por sexo, tipo de prenda o por parámetro de búsqueda recibidos por el método GET. También recupera de la BBDD los productos a los que el usuario (en caso de que haya

iniciado sesión) le ha dado like. Contiene una función usada en los productos para en caso de que esté entre los productos likeados por el usuario, devolver el estilo css correcto.

Contiene un formulario con un cuadro de búsqueda que al enviarlo llama a este mismo archivo y pasa el parámetro por GET, y unos filtros (todavía en desarrollo).

Para mostrar los productos, según los parámetros recibidos por GET, realiza una consulta SQL u otra a la BBDD (esto hay que refactorizarlo y mejorarlo, mucho código repetido) y los muestra junto a las tallas disponibles y las unidades por talla en cada producto.

procesarPedido.php:

Recibe por método POST los datos del pedido y los registra en la BBDD, tanto la cabecera del pedido (fecha, usuario, id de pedido), como el cuerpo del pedido, es decir, cada producto y su información.

logout.php:

Simplemente destruye la sesión y redirige a la página de inicio.

login.php:
Formulario de inicio de sesión. Muestra un formulario con los campos para correo y contraseña y al enviar el formulario se llama a si mismo, verifica con la BBDD si los datos son correctos y si lo son, inicia una sesión con el correo del usuario como identificador del usuario en la variable de sesión “USER”. En caso de ser erróneos, muestra un error según qué has introducido mal.

database.php:

Crea la conexión con la BBDD y se reutiliza allá donde hace falta.

carrito.php:

Extrae mediante JavaScript los productos del carrito de las cookies y los muestra junto a un resumen del subtotal, gastos de envio, iva y total del pedido y cuando realizas el pedido, llama a procesarPedido.php.

header.php:

Contiene lo común en varias páginas, para no repetir código. Por ejemplo, conexión a BBDD, inicialización de sesión, conexiones con CSS, JS..., el menú de la web.

register.js/login.js (está separado aunque hagan lo mismo para futuras implementaciones de funcionalidades separadas):

JavaScript sencillo para manejar el evento de click para mostrar la contraseña del formulario de registro.

productos.js:

Método habilitarCarrouseles:
habilita todos los carrouseles de imágenes de los productos en la página "productos.php", permitiendo tener varias imágenes del producto y pasar de una a otra.

Método habilitarLikes:

Habilita los "botones" de like en cada producto, es decir, controla el evento de "click" en los botones de me gusta (los corazoncitos), al darle like a un producto, verifica si la sesión esta iniciada mediante verificarSesionIniciada.php, en caso de haber sesión, los manda a registrar mediante saveLike.php, de lo contrario manda al usuario a la página de inicio de sesión.

Método habilitarBotonesCompra:

Habilita los botones de añadir al carrito, para que cuando le des click a uno de ellos, compruebe si el producto ya está en el carrito y las unidades en stock del producto que hay en la talla elegida. En caso de estar en el carrito y estar dentro del rango de unidades disponibles, aumenta el numero de unidades en el carrito, si no hay mas stock, no las aumenta. En caso de no estar en el carrito, añade el producto al carrito. Si actualiza información del carrito a nivel visual, también lo hace en la cookie.

header.js:

Método traductorGoogle:

Simplemente elimina elementos visuales del elemento que implementa el traductor de Google en la web, para dejarlo más simple visualmente.

Método habilitarBotonesHeader:

Controla el evento de "click" en los botones de la cabecera de la web y los menú. Cuando das click a uno de los botones, cierra el menú que este abierto, si lo hay, y abre el menú seleccionado.

Método habilitarMain:

Básicamente controla que si das click en algo que no pertenezca a la cabecera de la página (ósea, menús o la propia cabecera), se cierren los menús activos.

Método cargarCarrito:

Carga los productos desde las cookies y los muestra si los hay, si no, muestra un mensaje.

Método habilitarSearch:

Habilita el cuadro de búsqueda de la cabecera de la página y controla la entrada por teclado al escribir en el para que cada vez que introduces una tecla, haga una petición al archivo search.php, que devuelve los productos encontrados y en caso de haberlos, los muestra en la zona de resultados.

Método habilitarInteraccionesProductos:

Gestiona la interacción con los productos del carrito (en caso de que los haya) para que puedas eliminar productos, modificar el número de unidades de un producto (siempre respetando el limite máximo en base al stock disponible de ese producto en la talla elegida y el mínimo, que es 1) y actualiza la información en la cookie y en el carrito visualmente.

Método getCookie:

Simplemente es un método para reutilizar código y no tener que escribirlo todo el rato. Recupera la cookie con el nombre que le pasas por parámetro y te devuelve su contenido.

Método asíncrono comprobarSesion:

Llama a verificarSesionIniciada.php y en caso de que no haya sesión iniciada, te redirige a la página de inicio de sesión, si no, devuelve el usuario actual.

Método actualizarInfoCesta:

Actualiza la información numérica de la cesta calculándola en tiempo real, tanto subtotal como numero de unidades.

carrito.js:

Archivo que maneja la funcionalidad de la página de resumen del carrito en la que haces el pedido.

Método asíncrono comprobarSesion:

Llama a verificarSesionIniciada.php y en caso de que no haya sesión iniciada, te redirige a la página de inicio de sesión, si no, devuelve el usuario actual.

Método cargarProductos:

Carga los productos desde la cookie y si no hay, muestra un mensaje, pero si hay, los muestra junto a toda su información (nombre, talla, unidades, precio unitario y total del producto) y luego habilita el botón de realizar el pedido.

Método calcularResumen:

Calcula el resumen del pedido en base a los productos del carrito (unidades y precio por unidad), sacando el subtotal (unidades de cada producto x precio unitario), el IVA de ese total del producto, los costes de envio de ese producto y el total, que es la suma de todo.

Método actualizarDatos:

Recibe por parámetro el producto al que se le ha modificado la cantidad, lo actualiza en la cookie y modifica visualmente la información de ese producto, luego llama a calcularResumen para actualizar el resumen del pedido.

Método eliminarProducto:

Recibe por parámetro el producto a eliminar, lo elimina de la cookie y de la interfaz, dándole una animación, y calcula de nuevo el resumen del pedido.

Método realizarPedido:

Crea un formulario invisible para el usuario con todos los datos del pedido y lo envía automáticamente a procesarPedido.php.

Ejercicio 2:

Las pruebas básicas que podrían hacerse serían intentar registrar dos usuarios con el mismo correo, probar correos que no sean válidos, inyección SQL, probar si la verificación del correo funciona correctamente, si controla que un usuario haya iniciado sesión antes de poder realizar acciones restringidas a quien si haya iniciado sesión, si los productos respetan el stock de cada talla, si se añaden y eliminan correctamente al carrito en la interfaz y la cookie (cada producto con su talla elegida y las unidades de ese producto en esa talla, sin duplicados ni cosas extrañas). También se podrían probar los cuadros de búsqueda para ver que funcionan correctamente.

En general, probar formularios y todo con lo que pueda interactuar y modificar el usuario para comprobar que no puedan introducir datos erróneos o crear

cualquier tipo de conflicto en la app. Comprobar que los datos que se le sirven al usuario son correctos en todo momento en base a los parámetros de búsqueda o filtros / el paso o momento en el que se encuentre el usuario y que el flujo de la app es el correcto, que el usuario pasa por las pantallas que debería, las comprobaciones necesarias y que las respuestas que da la app son correctas en todas las circunstancias y controla los posibles fallos de la BBDD correctamente.

Ejercicio 3:

Las posibles mejoras / ampliaciones previstas y/o contempladas son la implementación de la pasarela de pago, manejo de direcciones de entrega del usuario, permitir cambiar la contraseña, desarrollo de las secciones incompletas de la web (como la pantalla de inicio, que ahora solo muestra un banner, las secciones de información sobre quienes somos y contacto), la implementación de un formulario de contacto, los filtros de la página de productos, implementar un sistema de opiniones y valoraciones de los usuarios de la tienda.

Y sobre todo, refactorizar el código para hacerlo mas modular y quizás separar la parte visual de la parte lógica de la app mediante archivos html y php separados y utilizar JavaScript como lógica en el lado del cliente para realizar las peticiones pertinentes a los archivos php necesarios.