Tabla de contenido

Datos generales

Código del sitio: HTML5, CSS3, librería Bootstrap, Javascript y Jquery

Llamadas a la API del profe: Todas son a través de Javascript usando funciones de flecha, las tomamos de postman y las adaptamos a lo que necesitamos.

Modificar elementos html como botones, carrito, nombre de usuario: Usamos JAvascript y Jquery para modificar el DOM de la página

Requerimiento 01 – Crear cuenta

Formulario: Creamos un formulario en HTML5

Validar formato correo: Lo hacemos declarando en el input type=”email”, html5 valida automáticamente el formato y marca si hay errores

crearCuenta.html - línea 108

Validar formato de Contraseña (mínimo 8 caracteres, al menos un número, una mayúscula y una minúscula): Lo hacemos por medio de **expresiones regulares de HTML5** usando las propiedades **minlenght**=”8” y **pattern**="^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[a-zA-Z\d]{8,}$"

crearCuenta.html - línea 114 y 118

Validar campos contraseña y repetir contraseña son iguales: Lo hacemos mediante HTML5 y javascript comparando los valores de los inputs al momento en que se va tecleando en el evento **Oninput**, lo pusimos como parámetro en la etiqueta donde abre <form> con **oninput**='password_confirmation.**setCustomValidity**(password_confirmation.value != password.value || password.value.length == 0 ? "Passwords no coinciden." : "")'

crearCuenta.html - línea 73

El usuario no se podrá registrar dos veces con el mismo correo: Lo validamos a través de la respuesta de la API, leemos el mensaje de error y si es “DuplicatedAccount”, le informamos al usuario

Js/ crearCuenta.js línea 39

Uso de Captcha: Bajamos una librería de Google y la implementamos validando mediante sus propias funciones

crearCuenta.html - línea 126 y Js/crearCuenta.js línea 09

Redirigir al usaurio: Lo hacemos con javascript con window.location = "login.html";

Js/crearCuenta.js línea 57

Requerimiento 02 – Recuperar contraseña

Solicitar cuenta de correo: En un form pedimos la contraseña

recupass.html línea 81

Pasar correo a la siguiente ventana donde pide código de recuperación: Lo hacemos pasando una variable local concatenándola en la url window.location = codrecupass.html?email="+formCorreo;

Js/recupass.js línea 35

Recuperar el correo en la ventana dónde pide código de recuperación: PARa recuperar cualquier variable local mandada por url creamos la función **getvarlocal()** que obtiene la url completa y mediante expresiones regulares busca la variable con el nombre que le hayamos pasado de parámetro a la función

Js/general.js línea 41

Validar la clave de recuperación: Lo hacemos mediante la respuesta que recibimos de la API, si nos manda el error nosotros mostramos al usuario "El correo o la clave son incorrectas"

Js/general.js línea 29

Si la clave es correcta permite modificar contraseña: Si la contraseña es correcta deshabilitamos el input de la clave y mostramos al usuario un div oculto que tiene un form que le pide contraseña nueva y confirmarla.

Js/general.js línea 34

Las reglas de validación de las contraseñas: Usamos expresiones regulares y HTML5 pusimos **minlenght**=”8” y **pattern**="^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[a-zA-Z\d]{8,}$"

codrecupass.html línea 90

Requerimiento 03 – Iniciar sesión

Inicio de sesión: Hicimos un formulario en HTML y al momento del submit primero evaluamos que los campos estén llenos y si es así enviamos la solicitud a la API

login.html línea 70 y js/login.js línea 30

Recordar cuenta: Guardamos el correo y la contraseña en local storage usando:

localStorage.setItem("UTemail", formData.get("email"));

localStorage.setItem("UTpassword", formData.get("password"));

js/login.js línea 39

Mantener sesión inicada si puso recuérdame y vuelve a entrar a login automático inicia sesión y redirige a inicio: Con JavaScript preguntamos al momento que la página si en localstorage existe el correo y la contraseña si sí pasmos los datos a la llamada a la API que inicia sesión automáticamente y redirige a inicio

js/login.js línea 2

Guardar nombre de la persona logueada: Al momento de iniciar sesión guardamos con javascript en localstorage el nombre del usuario concatenando los valores de nombre y primer apellido que nos regresa la respuesta de la API con:

localStorage.setItem("nombre_usuario", midata.data.customer.first_name +" "+midata.data.customer.last_name);

js/ login.js línea 48

Mostrar nombre de la persona logueda en barra menú: Con Javascript al cargar la página revisamos si está la variable declarada en localstorage, si sí la mostramos manipulando el DOM

Js/general.js línea 10

Mostrar botones de inicio de sesión y Cerrar sesión: Usamos un mismo elemento html el cual modificamos con javascript según si el usuario está logueado o no, cambiamos el dom para mostrar diferente texto y cambiamos el comportamiento de su función onClick

Js/general.js línea 13

Mantener sesión inicada en todas las páginas así se cierre o habrá una nueva: Utilizamos javascript apara ver si está declarada la sesión en localstorage, si sí la página válida para mantenerla activa.

Js/general.js línea 32

Requerimiento 04 – Pantalla búsqueda

Cargar productos en HTML: nos basamos en el ejemplo del profe, creamos una plantilla de cómo se acomodan los productos, hacemos la llamada a la API y modificamos buscando coincidencias para sustituir en el contenido del HTML

Js/rederizado_productos.js línea 43

Buscar por texto al clic en botón: creamos una función javascript que llama a la API con la búsqueda y ejecuta el mismo proceso para “pintar” el html usando la plantilla

Js/rederizado_productos.js línea 30