



Información:

Observaciones:

Api -> Utilizaremos la apiRest <https://fakestoreapi.com/>

1.- (1 puntos) Nombre del archivo: examenLogin.html (Realizará una demostración de lo siguiente):

a) Nos permitirá realizar un logeo en nuestro sistema. Para ello dispondremos de un formulario con dos campos, login y password, y un botón validar . Al pulsar en validar capturaremos los datos introducidos y realizaremos la validación del usuario con los datos de la solicitud <https://fakestoreapi.com/users>. Usaremos una función valida(objUsuario) [Donde objUsuario será un objeto con los atributos user y pass] que validará si los datos son correcto o no usando la correspondiente llamada a la API.

La función valida(objUsuario) devolverá un objeto Usuario o null si no existe dicho usuario.

El objeto Usuario contendrá los siguientes atributos:

id,

email:

username:

b) Si la validación es correcta nos redireccionará a la página datosExamen.html , si no es correcta aparecerá un mensaje indicando que el usuario es incorrecto, y almacenará una clave en el almacenamiento local con el objeto Usuario

2.- (5,5 puntos) Nombre del archivo datosExamen.html , en esta página mostraremos los datos de los productos. *Comprobará que existe la clave de sesión objeto Usuario [en caso de no existir la clave redireccionará la página examenLogin.html]*

a) Con la solicitud a la API <https://fakestoreapi.com/products> tenemos los datos referentes a los productos donde cada card podrá tener un aspecto similar la siguiente:

Utilizaremos los siguientes métodos:

CreaCard(): Recibirá un objeto producto y devolverá un nodo de DOM que representará cada producto individual. Utilizaremos esta función para la representación de todos los productos. (0,5 punto)

WD 2TB Elements



USB 3.0 and USB 2.0
64
Rating: Rate:3.3 - Count:203

CreaCards(arrayProductos) : Recibirá un array de productos creará la página (usando DOM) utilizando la función CreaCard() para cada producto. (0,5 punto)

Fjallraven - Fo



Your perfect pack fo
109.95
Rating: Rate:3.9 - Count:120

Mens Casual Pre



Slim-fitting style,
22.3
Rating: Rate:4.1 - Count:259

Mens Cotton Jac



great outerwear jack
55.99
Rating: Rate:4.7 - Count:500

Mens Casual Sli



The color could be s
15.99
Rating: Rate:2.1 - Count:430

John Hardy Wome



From our Legends Col
695
Rating: Rate:4.6 - Count:400

Solid Gold Peti



Satisfaction Guarant
168
Rating: Rate:3.9 - Count:70

White Gold Plat



Classic Created Wedd
9.99
Rating: Rate:3 - Count:400

Pierced Owl Ros



Rose Gold Plated Dou
10.99
Rating: Rate:1.9 - Count:100

WD 2TB Elements



USB 3.0 and USB 2.0
64
Rating: Rate:3.3 - Count:203

Filtrar(objFiltro): Devolverá un array de productos según las condiciones del objeto objFiltro pasado como argumento. (2 puntos)

El objeto objFiltro tendrá la siguiente estructura:

{ atributo, valor }

Atributo será el atributo por que deseamos filtrar los datos, este podrá ser: Price, rate, count

Valor: será el valor máximo de los datos a filtrar.

Ejemplo { “price”, 101 } -> nos devolvería todos los productos cuyo precio sea inferior a 101

DevuelveGeneros(arrObjBDProductos): Esta función recibirá el array de productos y devolverá un array con las distintas categorías que existen, es decir, un array con los distintos valores posibles del atributo category de los productos. (1 puntos)



Diseña y usa un formulario que permite la utilización de las funciones de filtro
(0,5 puntos)

Filtros

Atributo Valor

Buscador: Sobre al Buscador: Implementar un buscador al ir pulsando las teclas me irá mostrando los posibles artículos cuyo nombre coincide con el texto introducido.
(1 puntos)

3.- (3,5 puntos) Sobre la página datosExamen.html vamos a añadir la funcionalidad asociada a un carrito. Para ello crearemos los siguientes elementos:

a) Clase Carrito, con un atributo artículos que será un array con los artículos (podemos poner sólo el título de los artículos). (0,5 punto)

Y los siguientes métodos:

add(elemento): Añadirá un elemento al carrito

dibujaCarrito(): Retornará un nodo DOM (por ejemplo fragment) que contendrá todos los elementos del carrito, y nos permitirá representar el carrito añadiendo dicho nodo al DOM de nuestra página.

b) Añadiremos la funcionalidad pertinente para que al hacer click en una imagen del producto de la página, añadirá dicho producto a nuestro carrito. Y repintará el carrito para que aparezcan los elementos seleccionados. indicando en todo momento el total de productos (1,5 puntos)

CARRITO

Mens Casual Premium Slim Fit T-Shirts

White Gold Plated Princess

Pierced Owl Rose Gold Plated Stainless Steel Double

c) En la parte inferior de la página aparecerá un Contador con el número total de artículos incluidos en el array del almacenamiento local. (0,5 puntos)

d) Al incluir un artículo al carrito aparecerá el icono asociado al artículo, de la forma:



Al pulsar el icono del check dejará de estar seleccionado dicho elemento y por tanto se retirará del carrito de la compra. (1 punto)