**Evaluación Next Js - Web Sockets**

**Tema 2**

* La entrega se realiza a través de un archivo RAR con el formato de nombre. 5A\_REACT\_APELLIDO\_NOMBRE.rar o

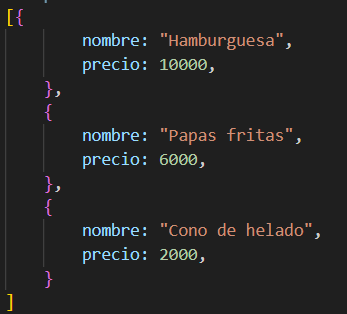
5B\_REACT\_APELLIDO\_NOMBRE.rar Por ejemplo: 5X\_REACT\_MARCHESI\_MATIAS.rar

* Si bien se pueden consultar páginas web y código previamente realizado para la resolución del examen, **cualquier intento de comunicación con un compañero** (vía Classroom, WhatsApp Web, Mail, Replit, Discord, celular, personalmente, etc.) **será motivo de anulación del examen**. A su vez, está sumamente prohibido el uso de inteligencias artificiales (chatgpt, etc)
* Recuerde apoyarse en el uso de la consola para la detección de errores.
* La web entregada puede ser lo más fea posible. No pierda tiempo intentando embellecerla. Eso sí, debe poder ser entendible y cumplir con los requisitos que se le solicitan.
* ***Comentar el código, indicar qué se esperaba hacer, qué se hizo y qué diferencia hay entre lo que se esperaba y lo que se logró hacer (esto último en caso de no haber finalizado).***
* ***En caso de finalizar el examen en tiempo y forma, comuníqueselo a los docentes previo a entregar.***

**Consigna:**

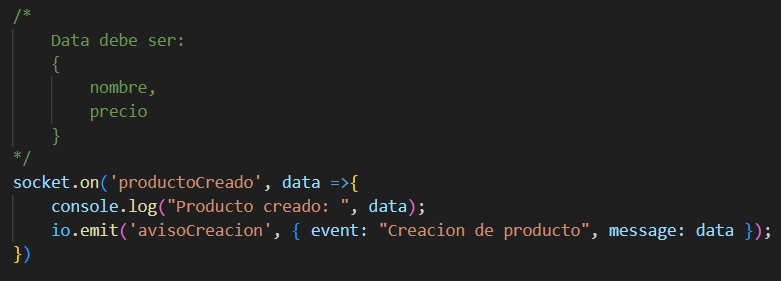
Realizar un programa capaz de organizar Productos para vender en una tienda de comida.

1. Realizar un componente llamado **Producto** que consta de 2 campos: **Nombre del producto** y **Precio del producto**.
2. Realizar una página home donde la primera vez que se carga la página ***(use effect de dependencias vacías)*** se llame por medio de un fetch a un pedido **GET “/productos”**, la cual va a traer una lista con todos los productos en el Backend. Una vez que se tenga la lista en el Front mostrar en la página web las mismas ***(hacer un mapeo)***. La lista de los productos va a tener el siguiente formato:



1. Los dos inputs y el botón deberían estar en el mismo componente para que el buttonHandler tome el contenido de los inputs facilmente  
     
   Sumarle a la página dos inputs ***(textInputComponents modificados)*** y un botón para agregar un producto a la lista. Para lo cual, al pulsar el botón se debe enviar un fetch al pedido **POST “/crearProducto”** con los dos datos en un objeto ***(req.body del post)*** cuyas parámetros sean nombre y precio. Previo a eso, se debe verificar en esta función que no exista un producto con ese nombre, ya que el nombre es el identificador único de los productos.
2. Cuando el usuario hace un POST de “/crearProducto”, en el back (presumiblemente) se añade el producto a la lista. Ahí es cuando se “crea”. Inmediatamente después de eso, hacete el io.emit que dentro tenga la data del array entero  
     
   Al crearse, enviar un aviso por socket ***(io.emit en back)*** para que le llegue a los otros usuarios al instante así pueden actualizar su lista.

**Pista**: Tener en cuenta que en todos los puntos que se pide hacer un socket se debe realizar tanto el emit (socket.emit()) como la función correspondiente para la recepción (socket.on()). En este caso se tiene que actualizar la lista de productos. En el back esto está hecho de esta forma:



1. Acá tenes que buscar en la lista usando algún método. Buscate un método de búsqueda en internet que no se me ninguno de memoria. Tenés que encontrar la posición en el array en la cual estas buscando el elemento. Una vez que lo encontras haces array[posición] = a lo que quieras  
     
   Sumarle a la página otros elementos: **dos inputs y un botón** para cambiar el precio de un producto ***(para cambiar el precio del producto lo tenes que buscar en el array por su nombre)***. En el primer input el usuario deberá escribir el nombre del producto a modificar y en el otro input elegirá el nuevo precio. Al presionar el botón se debe enviar por socket un aviso al back ***(front hace socket.on y cuando el back lo recibe hace io.emit)*** y a todos los usuarios con el nuevo precio del producto.

Recordar:

* Hacer la función recepción del socket (socket.on()) para que al momento de recibir un cambio de precio en una de ellas, la misma se modifique en la lista de productos.
* El backend estará a la espera de un evento **“productoModificado”** y emitirá un evento **“avisoModificación”**. Armen el objeto data de forma tal que tenga el nombre del producto y el precio modificado.