Incrementar ID dinámicamente y validaciones

En esta sección, se resolverá el problema del ID duplicado en los ítems de la factura, y cómo incrementarlo dinámicamente para evitar conflictos en la consola. Además se realizará validaciones en los campos del formulario.

Pasos 70 y 71

Sigue estos pasos:

Paso 70: Solucionar el problema del ID duplicado

Anteriormente, cada nuevo ítem que se creaba tenía el ID 4, lo que generaba un conflicto debido a la duplicidad de keys. El error en la consola aparece cuando se agrega el quinto elemento desde el formulario a la tabla de items en el navegador.



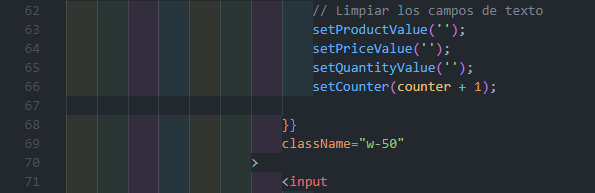
Para solucionarlo, se procede a utilizar el hook **useState** para manejar un contador de ID que se incrementará con cada nuevo ítem.

Primero se define un nuevo estado para el contador de ID, se establece el número **4** como valor inicial para el **useState** porque se tiene 3 elementos en el objeto **invoice** (desde la función **getInvoice**)

Luego se actualiza la función **onSubmit** del formulario, de tal manera que se asigne el valor de la variable **counter** como el **id** del nuevo ítem; asimismo se incrementa el contador en **1** después de añadir el ítem con la función **setCounter**.

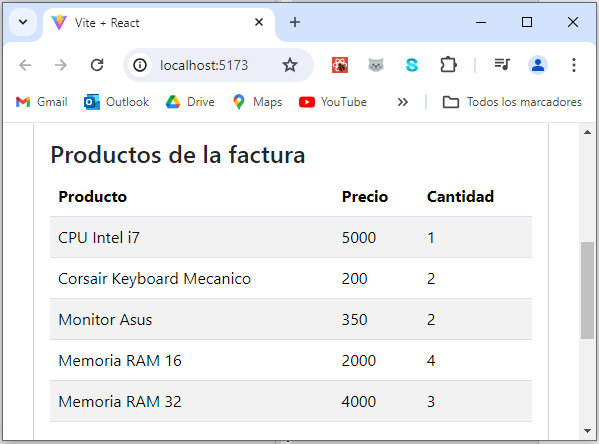
Todo esto se implementa de la siguiente manera:



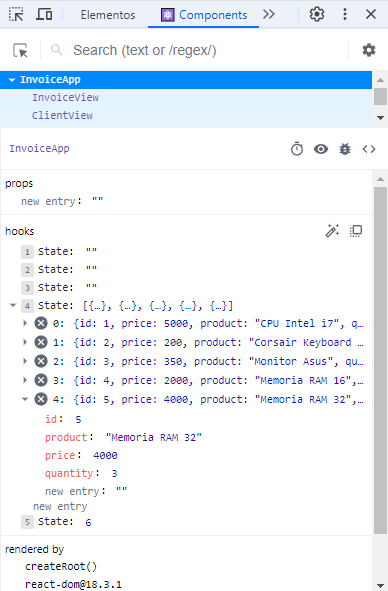


Al hacer una prueba en el navegador, se pueden agregar varios ítems sin que aparezcan errores de duplicación de **id** en la consola. Los valores de los inputs también se limpian correctamente después de añadir un nuevo ítem.

En el formulario se agregaron 2 items tal y como se muestra:

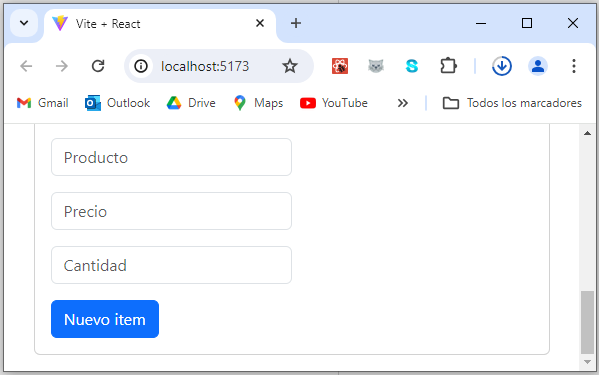


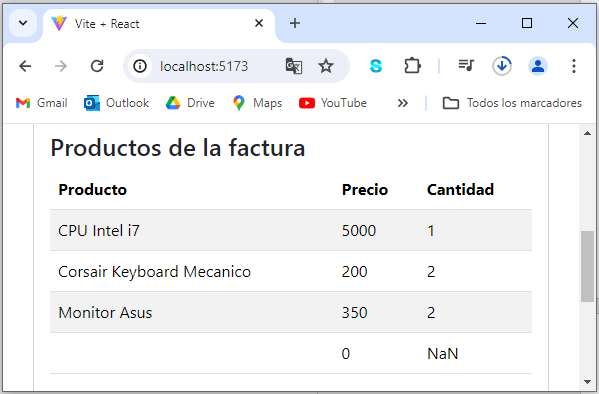
Ten en cuenta que el valor actual del counter (inicialmente tiene el valor 4) se asigna al nuevo ítem que se va a agregar y luego se suma 1 al counter para que pueda asignar ese nuevo valor del siguiente ítem que se va a agregar. Esto se puede notar en la consola de Chrome, en la sección **Components**.

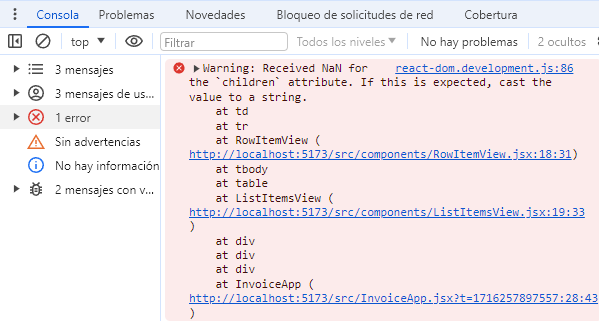


Paso 71: Agregar validaciones en el formulario

En la consola se generaba un error al intentar agregar ítems con campos vacíos en el formulario, el error indicaba que no se podía convertir el valor de string a entero, además si se agregaba el ítem con campos vacíos en el navegador se crea una nueva línea en la tabla con datos vacíos.







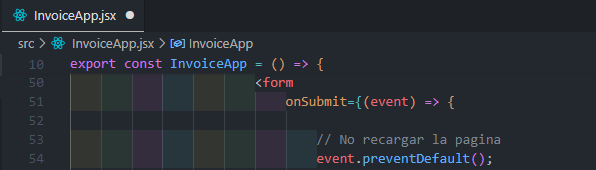
La solución es agregar validaciones en la función **onSubmit** para asegurar que los campos no se encuentren vacíos antes de agregar el ítem a la lista. Se tiene en cuenta lo siguiente:

Las validaciones se colocan después del método **preventDefault** aplicado al objeto **event**, pues las validaciones se aplican luego de enviar el formulario pero antes de agregar el nuevo ítem a la lista con la función **setItems** cuando se define los estados para los valores de los inputs (**productValue**, **priceValue** y **quantityValue**).

Se utiliza el método de JavaScript **trim** para eliminar espacios en blanco y la propiedad **length** para medir el largo del valor.

Se tiene en cuenta la siguiente lógica: si el valor de **productValue** o **priceValue** (el valor introducido en el campo del formulario correspondiente) es menor o igual a **1** carácter, o si el valor de **quantityValue** es menor a **1** carácter, no enviar el formulario (se utiliza un **return** vacío para finalizar la ejecución de la función).

Adicionalmente se puede añadir la clase de Bootstrap **my-3** al botón del formulario para aplicarle unos márgenes







Al intentar enviar el formulario con campos vacíos, este no se enviará. Si se coloca un carácter en el campo producto y/o precio (debe ser mayor a 1 carácter) o en el campo cantidad (debe tener al menos 1 carácter), el formulario no se enviará hasta que se completen correctamente los campos.

