

Actividad evaluable UT4A1 – Documentación de aplicaciones

Manual de usuario

Desarrollo de Interfaces - 2º DAM - CIFP Villa de Agüimes

Moisés Antonio Pestano Castro

Enlaces:

- <https://github.com/MoisesAPC/proyectomapc/tree/informes>

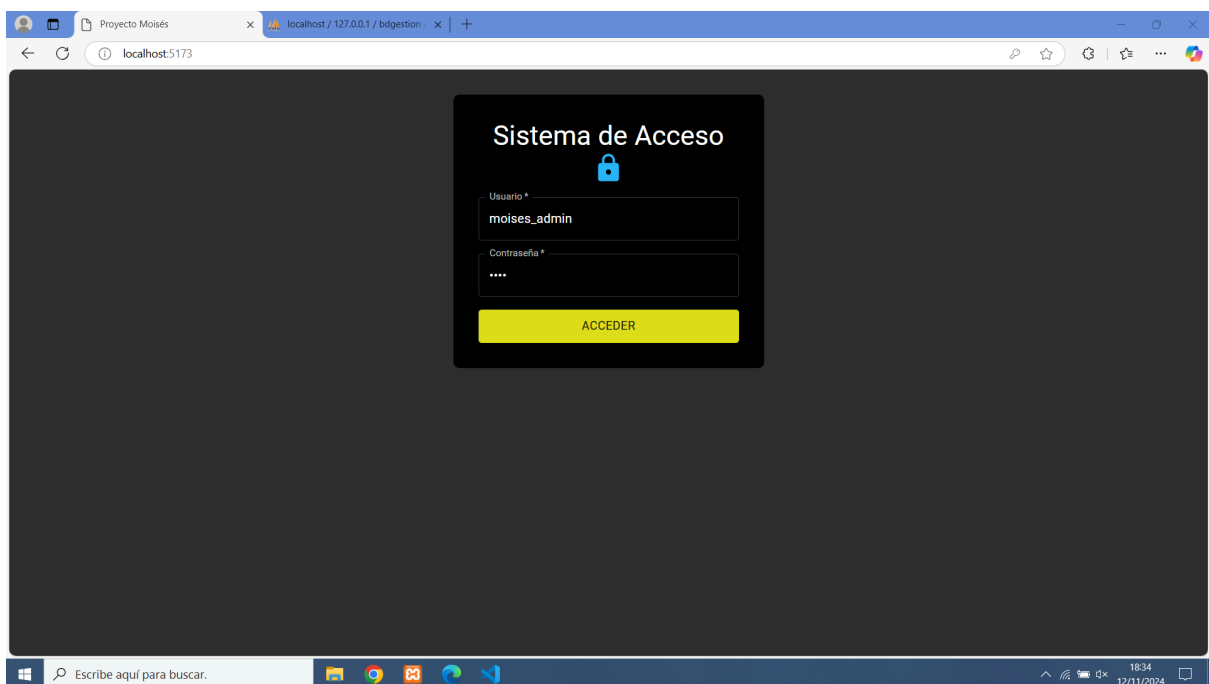
Índice

Introducción y modo de uso	3
Inserción de datos	5
Eliminación de datos	6
Roles: “admin” vs “user”	7
Generación de informes	10
Ayudas	15
Notas adicionales	16

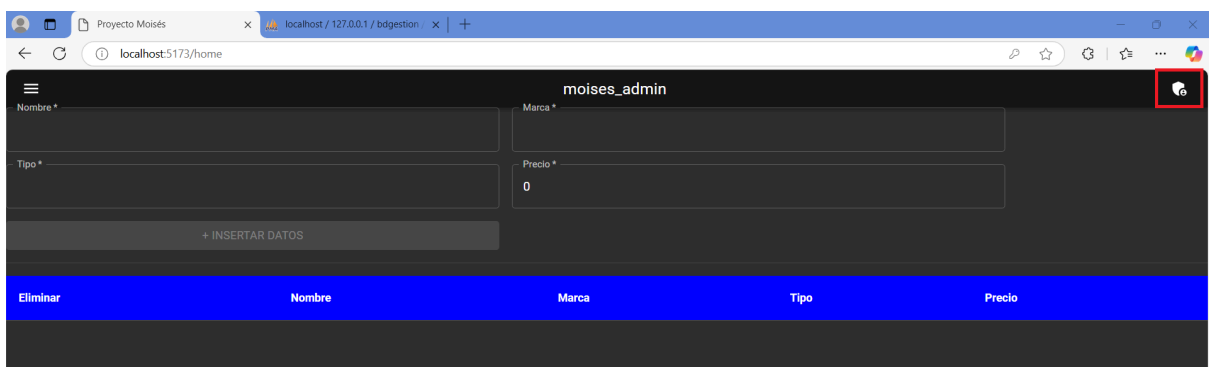
1. Introducción y modo de uso

En esta aplicación, podremos insertar y eliminar productos de una base de datos local.

Cuando entremos en la página principal de la aplicación (<http://localhost:5173/>), deberemos de introducir las credenciales:



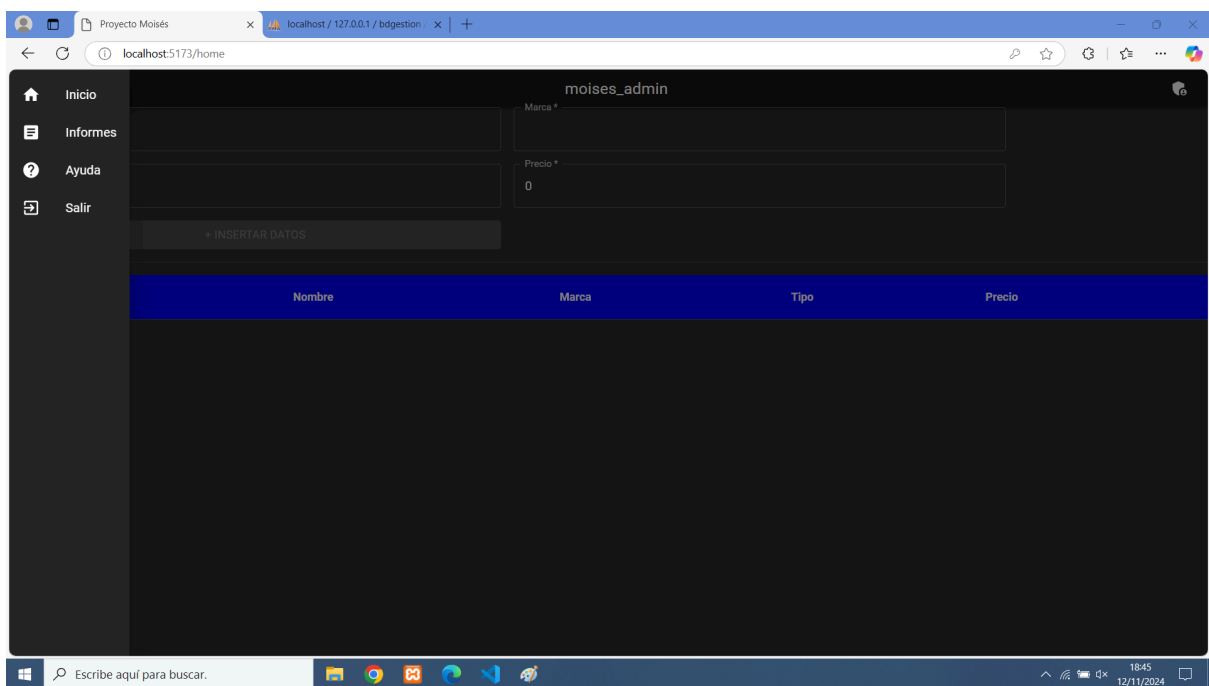
Una vez hayamos accedido a la página, llegaremos a la página **Home**. Aquí podremos ver el listado de todos los productos así como poder insertar nuevos productos a través del formulario. Si el usuario que accede tiene el rol de administrador, aparecerá un ícono en la parte superior derecha de la pantalla.



MANUAL DE USUARIO

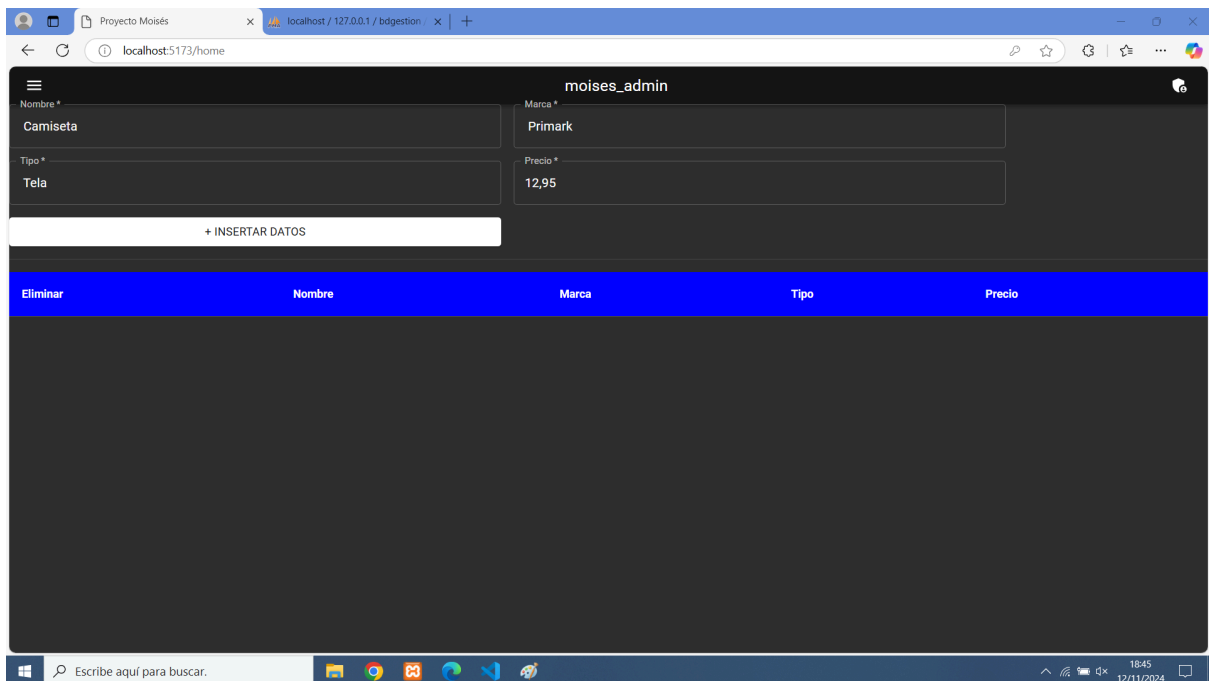
En la parte superior izquierda, disponemos de un menú hamburguesa con varias opciones:

- **Inicio:** Redirige a la pantalla Home. Esta pantalla es la que contiene el formulario y el listado de los productos.
- **Informes:** Redirige a la pantalla Reports.
- **Ayuda**
- **Salir:** Cierra sesión con el usuario actual y vuelve a la pantalla de login.



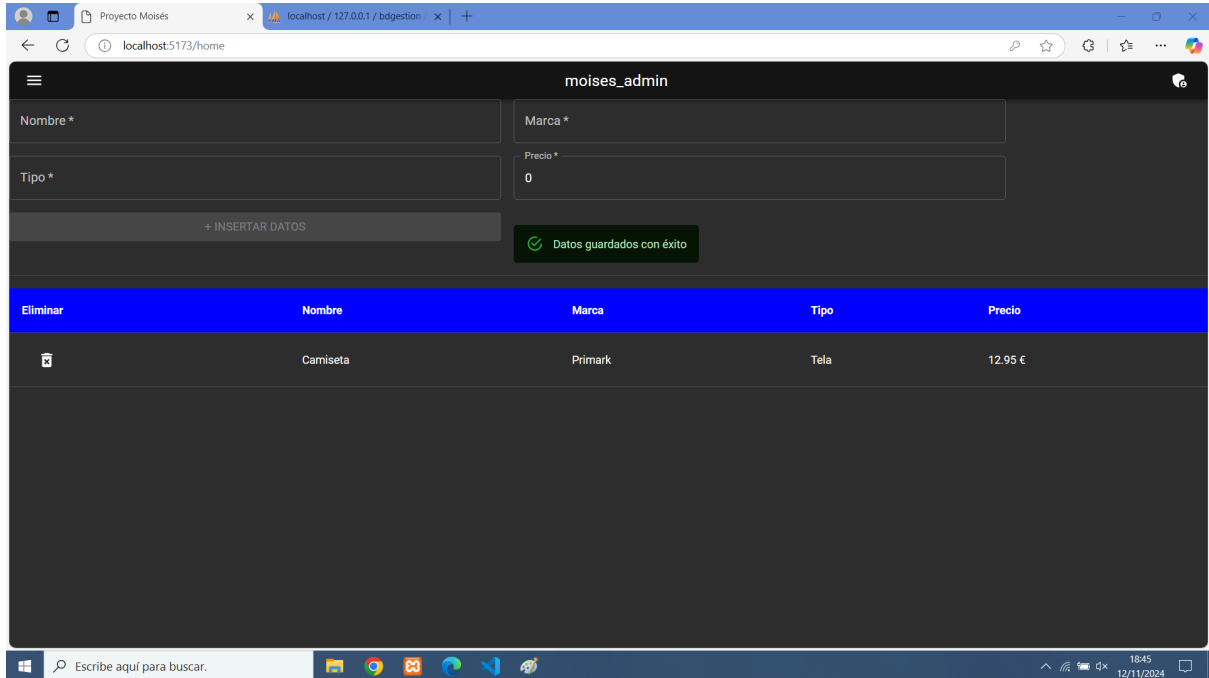
2. Inserción de datos

Para insertar productos, rellenamos los campos correspondientes del formulario. Nótese que en el campo del precio solamente se pueden introducir números positivos (pueden ser decimales). Cuando hayamos introducido los campos, le damos al botón de INSERTAR DATOS.



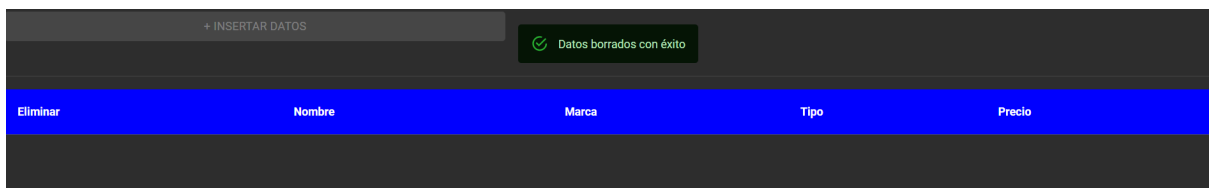
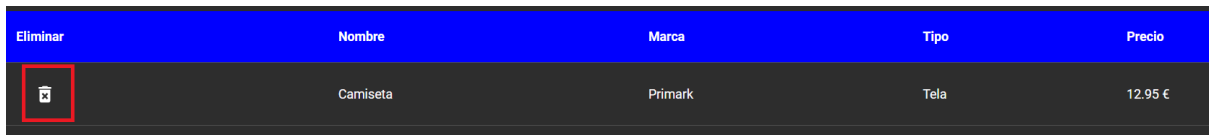
The screenshot shows a web browser window with the address bar displaying 'localhost:5173/home'. The application interface is titled 'moises_admin' and features a dark theme. It contains a form with four input fields: 'Nombre *' (containing 'Camiseta'), 'Marca *' (containing 'Primark'), 'Tipo *' (containing 'Tela'), and 'Precio *' (containing '12.95'). Below the form is a white button labeled '+ INSERTAR DATOS'. At the bottom of the form area, there is a table header with five columns: 'Eliminar', 'Nombre', 'Marca', 'Tipo', and 'Precio'. The table body is currently empty. The Windows taskbar at the bottom shows the search bar with the text 'Escribe aquí para buscar.' and the system clock indicating 18:45 on 12/11/2024.

Si la inserción fue correcta, veremos el producto en la lista:



3. Eliminación de datos

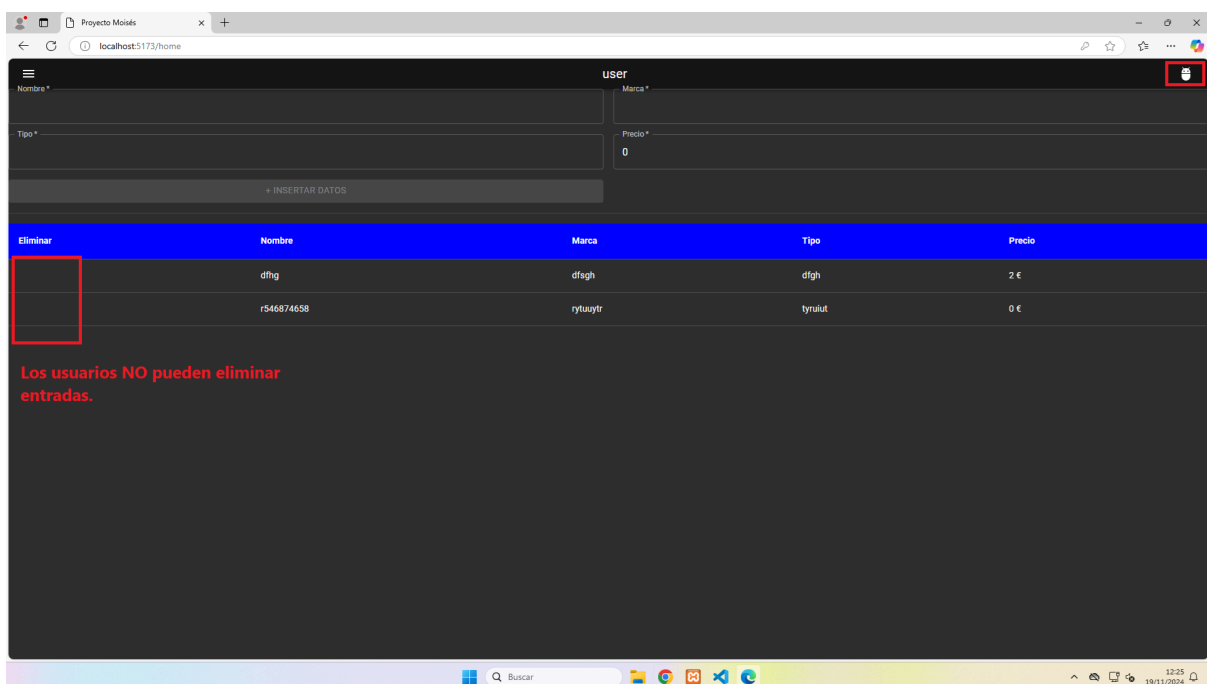
Para borrar un producto, basta con hacer click en el ícono del cubo de basura y el producto se eliminará:



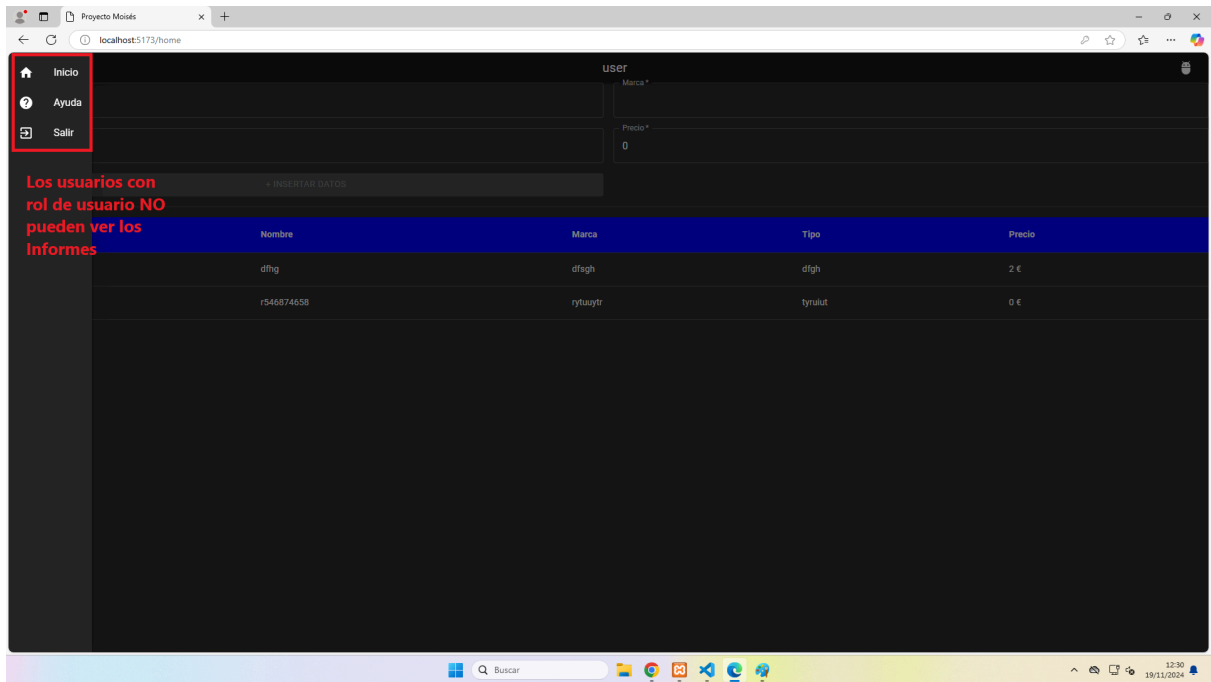
4. Roles: “admin” vs “user”

En nuestra aplicación, existen dos roles: administrador (“admin”) y usuario (“user”). Cada uno tiene su propio ícono, el cuál podremos ver en la parte superior derecha.

Los usuarios con rol usuario **NO** pueden borrar entradas de la tabla colecciones, ni generar informes nuevos. Dichas opciones solo están disponibles para los administradores.

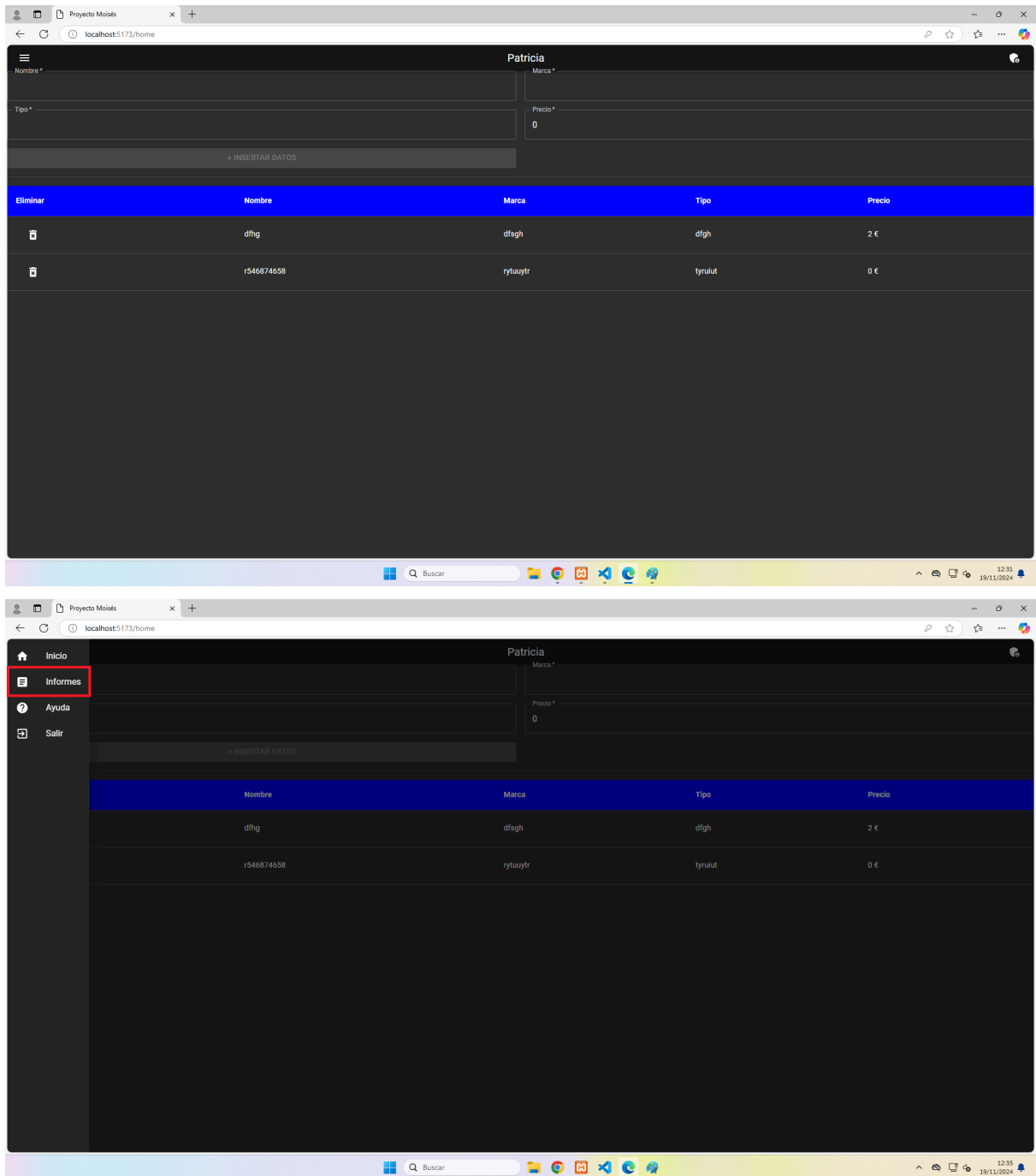


MANUAL DE USUARIO



Así es como se ve la aplicación cuando estamos logueados con el rol de usuario.

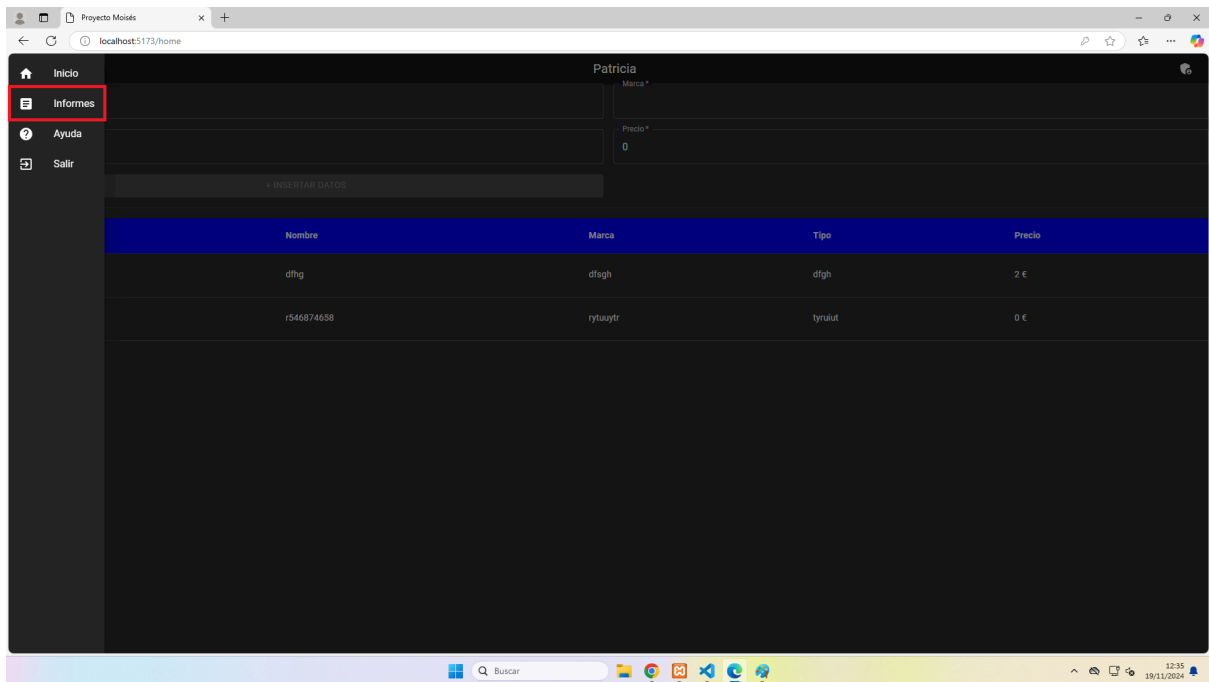
MANUAL DE USUARIO



Así es como se ve la aplicación cuando estamos logueados con el rol de administrador.

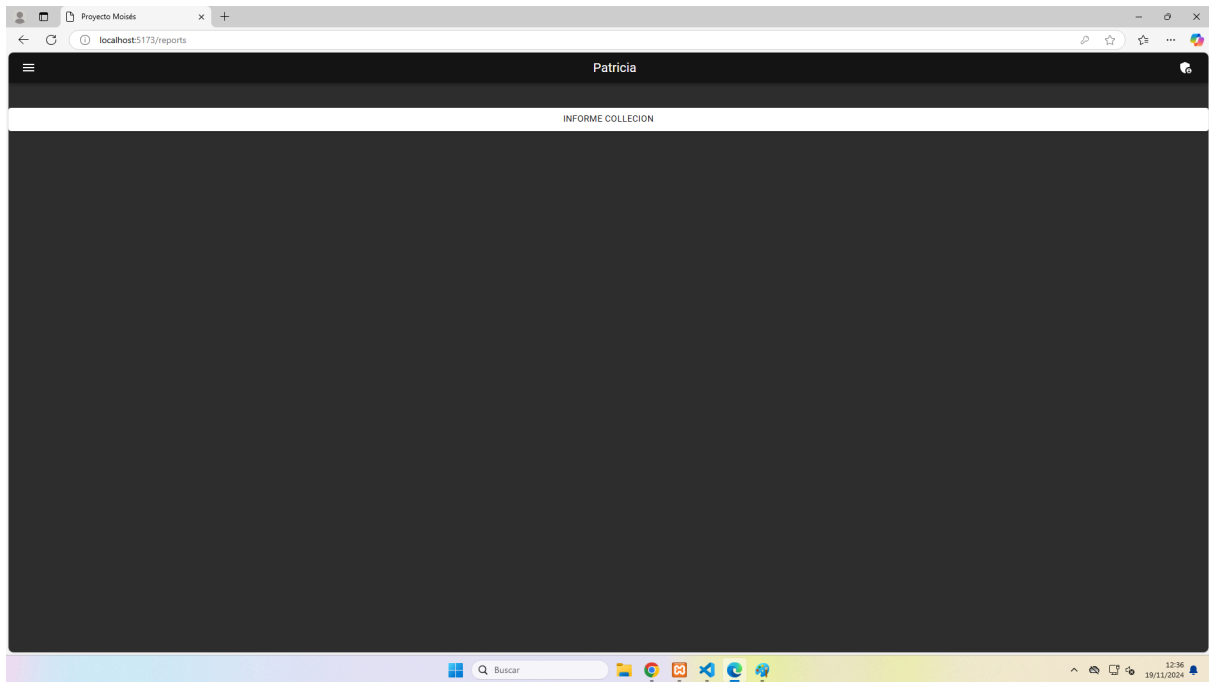
5. Generación de informes

Para generar informes, vamos a loguearnos como un usuario con rol de Administrador. Una vez logueado, hacemos click en el menú hamburguesa y seleccionamos la opción “Informes”.



MANUAL DE USUARIO

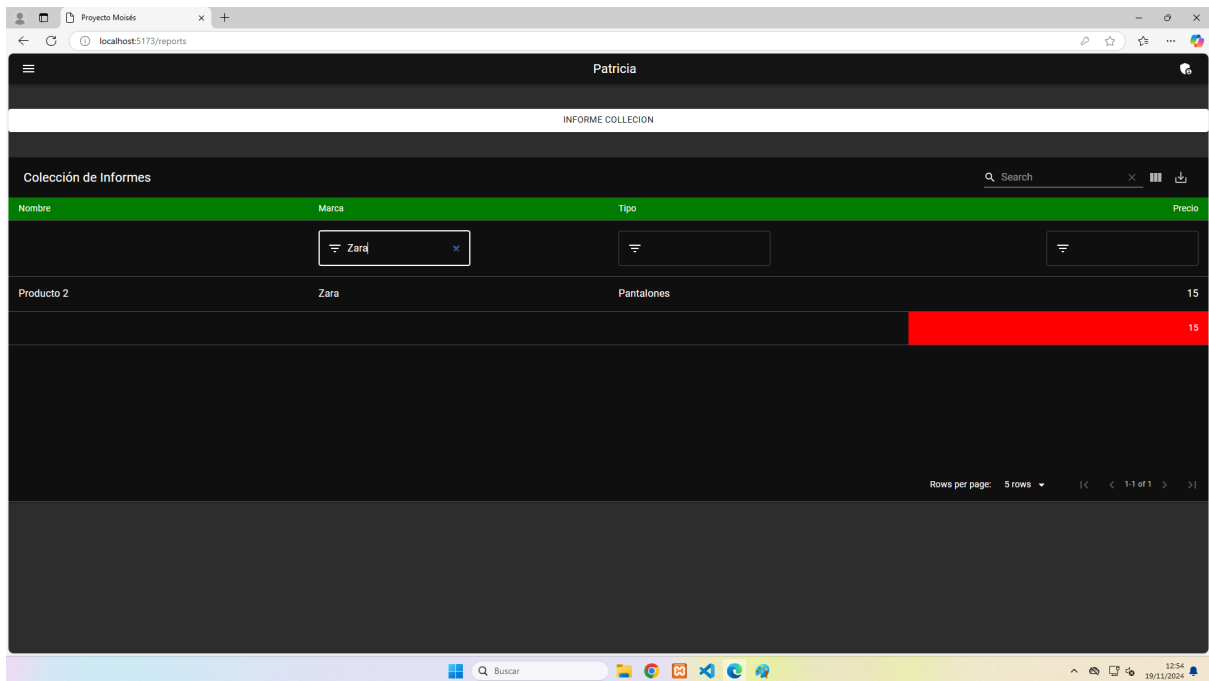
Ahora, hacemos click en el botón “**INFORME COLECCION**” para generar el informe con la lista de los productos. En rojo se nos mostrará, además, la suma de los precios de todos los productos de la base de datos.



MANUAL DE USUARIO

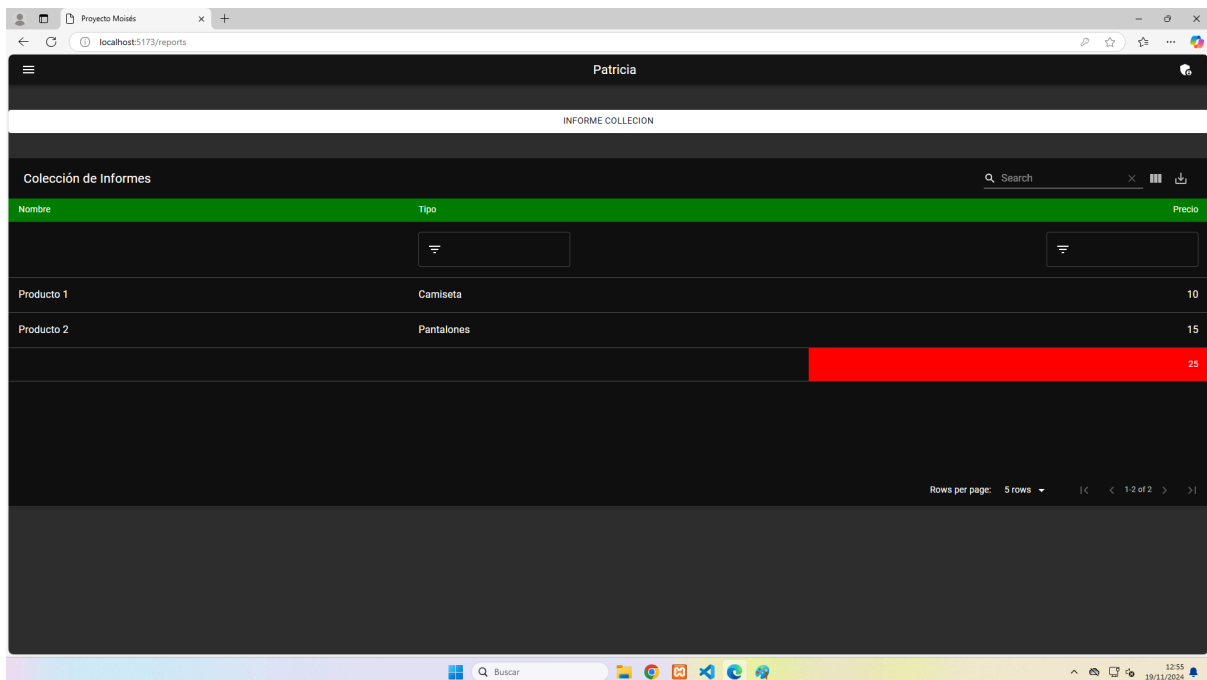
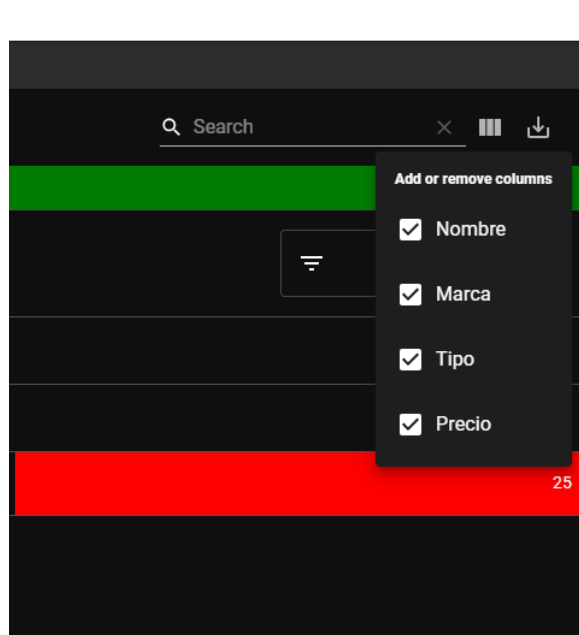
Una vez generado, dispondremos de varias opciones:

- **Opción de búsqueda (Search)**: Con esta opción podremos introducir un término de cualquiera de los valores que tengamos en la tabla y se nos mostrará las filas con dicho valor
- **Opciones de filtrado**: En los campos de Marca, Tipo y Precio, podremos introducir un término para poder filtrar solamente las entradas que se encuentren en dichas columnas. Funciona como la opción de búsqueda, pero solamente teniendo en cuenta esas columnas.



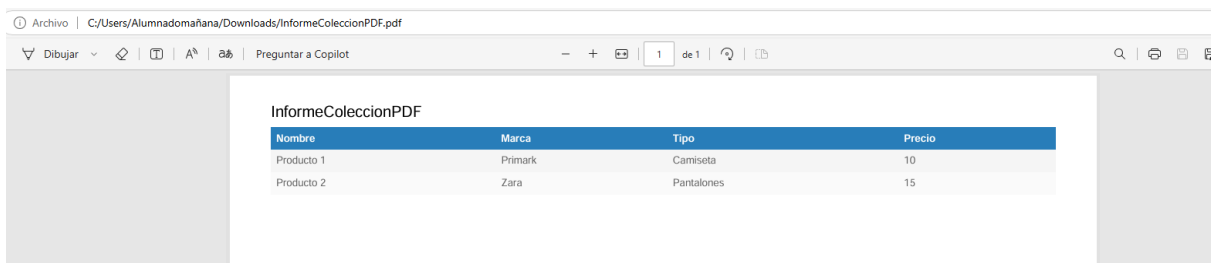
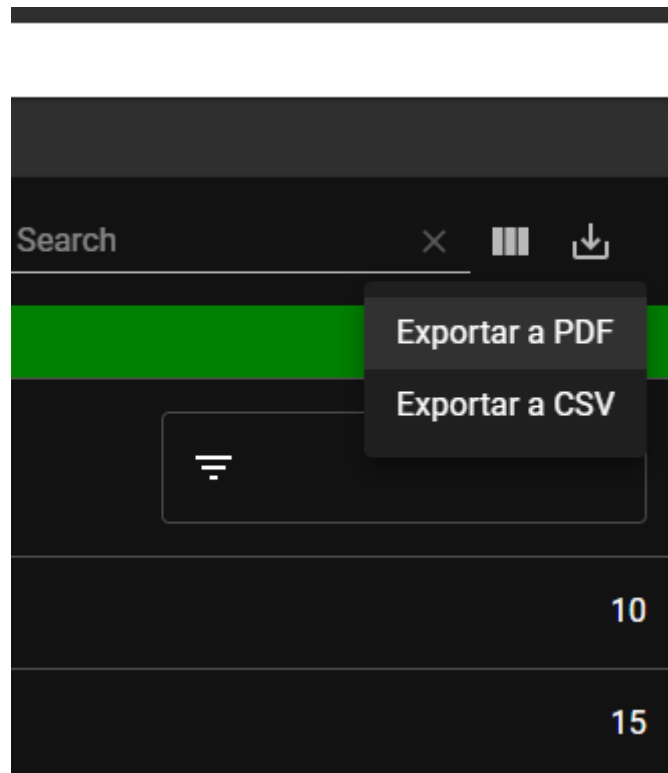
MANUAL DE USUARIO

- **Mostrar columnas:** Con esta opción, podremos mostrar columnas específicas de la tabla.



Así se ve cuando ocultamos la columna “Marca”

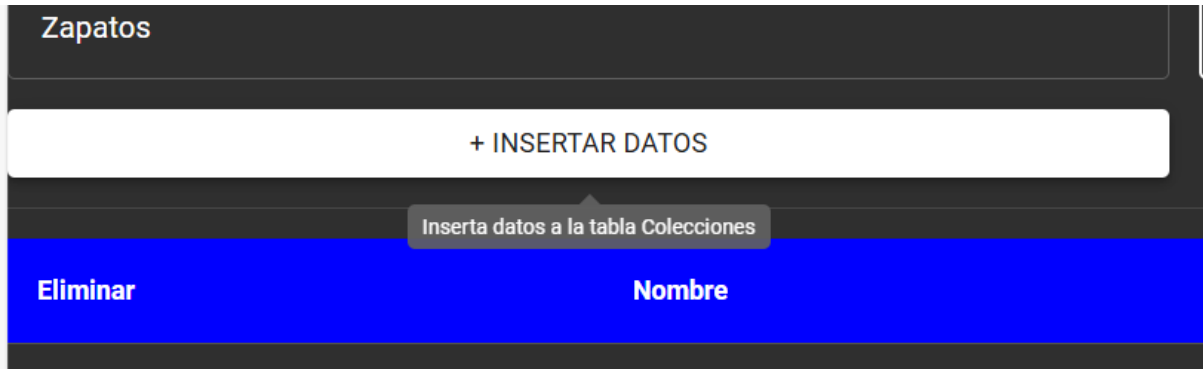
- **Exportar:** Podremos exportar la tabla en dos formatos distintos: PDF y CSV.



Así se ve cuando exportamos la tabla a PDF

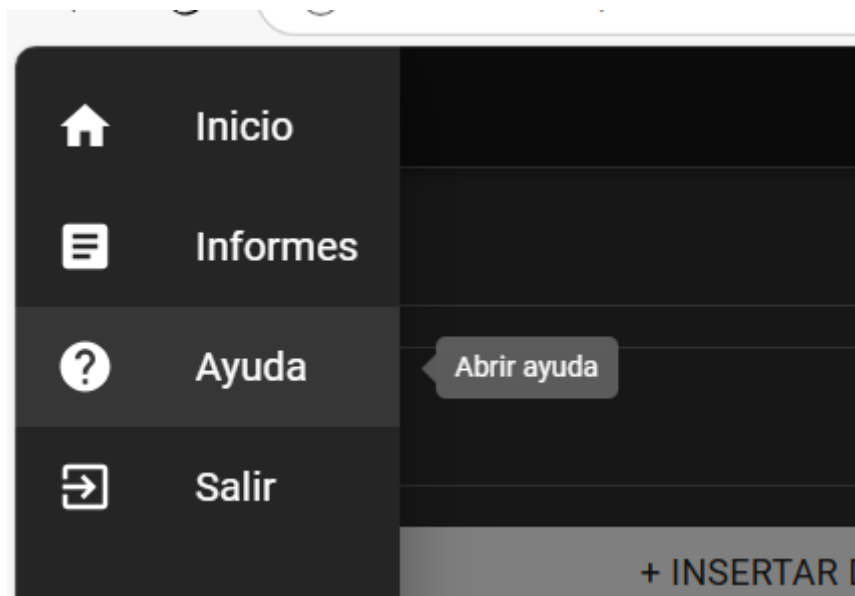
6. Ayudas

Cada botón de la aplicación dispone de un pequeño mensaje de ayuda que se muestra cuando el cursor se deja sobre el botón en cuestión.



Ejemplo con el botón de “Insertar Datos” de la página “Home”

Además, puedes acceder a este manual de usuario haciendo click en el botón de **Ayuda** del menú hamburguesa. El fichero PDF con el manual se abrirá en una pestaña aparte.



7. Notas adicionales

El front-end de la base de datos se ejecuta en el puerto **5173** (usando **npm run dev**), mientras que el back-end se ejecuta en el puerto **3030** (usando **npm run start**).

¿En qué parte del frontend hacemos **fetch()** al endpoint que hace el **SELECT**?
¿Cómo lo hacemos?

El **fetch()** al endpoint de **SELECT** se realiza en la función **obtenerDatosEnTablaColeccion()** del fichero **frontend\src\components\Dashboard.tsx**.

En él, hacemos una llamada al endpoint **getItems**, que a su vez llama al método **getData**, el cual realiza la consulta SQL **SELECT** para obtener los datos.

```
async function obtenerDatosEnTablaColeccion () {
  fetch(`http://localhost:3030/getItems`)
  .then(response => response.json())
  .then (response => {

    console.log('Lo que nos llega de la base de datos (SELECT tabla
coleccion): ')
    console.log(response.data)

    setTableData(response.data)
  })
}
```

```
app.get('/getItems', async function(req, res, next) {
  try {
    res.json(await items.getData(req, res))
  } catch (err) {
    console.error(`Error while getting items `, err.message);
    next(err);
  }
})
```



```
async function getData (req, res) {  
    // La variable rows almacena los datos obtenidos de la consulta  
select.  
    const rows = await db.query(  
        `SELECT * FROM coleccion`  
    )  
    /* Los datos obtenidos de la consulta del select los paso por la  
función helper para que  
    en el caso de que no haya datos devueltos, me devuelva un array  
vacío. */  
    const data = helper.emptyOrRows(rows)  
    return {  
        // Devolvemos el resultado del Select, que está almacenado en  
la variable data  
        data  
    }  
}
```