El function service

En esta sección, se procederá a crear una nueva función correspondiente al servicio, la cual permitirá obtener la factura definida en **invoice.js**.

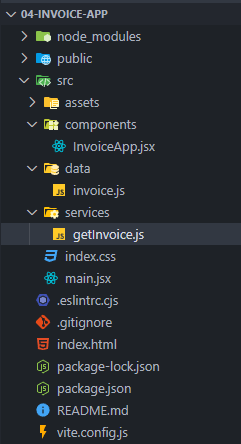
Esa función o helper va a ser utilizada en el componente **InvoiceApp** para mostrar toda la información relevante, como el id, nombre, cliente, empresa y detalles de la factura, incluyendo los productos comprados.

Pasos 8 al 11

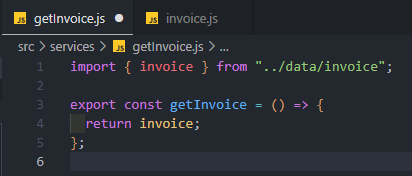
Sigue estos pasos:

Paso 8: Crear el archivo de servicio

1. Crea una carpeta dentro de **src** con el nombre **services** o **helpers**, dependiendo de tu preferencia.
2. Dentro de esa carpeta, crea un archivo llamado **getInvoice.js**, ya que este archivo contendrá funciones relacionadas con la obtención de facturas.



1. En este archivo, proceda a definir una función de flecha exportable que retorne el objeto **invoice**, el cual fue definido previamente en el archivo **invoice.js** ubicado dentro de la carpeta **data** (por lo cual se debe importar).



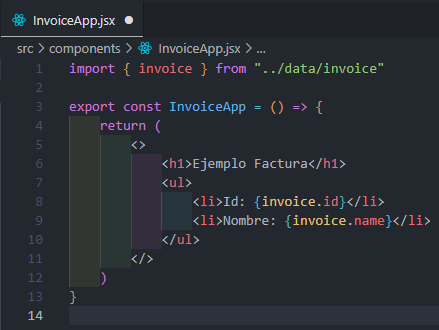
**Nota:** Para facilitar la importación del archivo, es importante mantener abierto el archivo **invoice.js** para que se importe automáticamente en el entorno de desarrollo al momento de escribir la constante **invoice**.

Paso 9: Integrar la data de factura en el componente InvoiceApp

Es importante tener en cuenta que en el archivo **invoice.js** se exporta el objeto **invoice**, lo cual permite su importación en otros archivos para su utilización. Procede a importar el archivo **invoice.jsx** en el componente **invoiceApp**.

En el componente **InvoiceApp**, se obtiene la información de la factura en el navegador, como el ID y el nombre (se escribe el nombre del objeto **invoice** seguido por un punto y su atributo correspondiente).

Es importante recordar que las etiquetas HTML **<ul>** se utilizan para definir una lista y **<li>** define cada uno de los elementos de la lista.



import { invoice } from "../data/invoice"

export const InvoiceApp = () => {
    return (
        <>
            <h1>Ejemplo Factura</h1>
            <ul>
                <li>Id: {invoice.id}</li>
                <li>Nombre: {invoice.name}</li>
            </ul>
        </>
    )
}

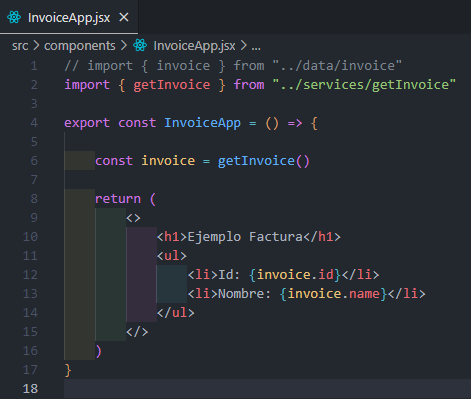

Además, se recomienda tener abierto el archivo **invoice.js** para facilitar la importación rápida.

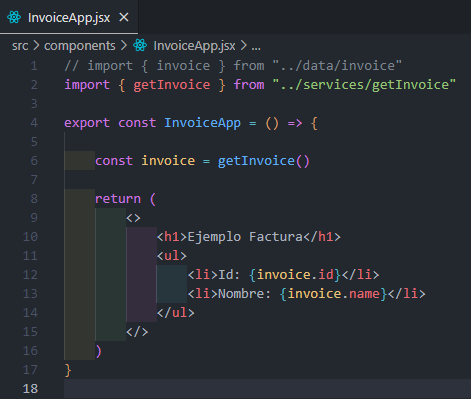
También es posible realizar manipulaciones en el objeto **invoice**, cómo devolver un objeto acortado con ciertos atributos o realizar búsquedas utilizando los operadores **filter** o **find**. Sin embargo, dado que en este caso solo se tiene una factura, es suficiente para los propósitos de esta aplicación.

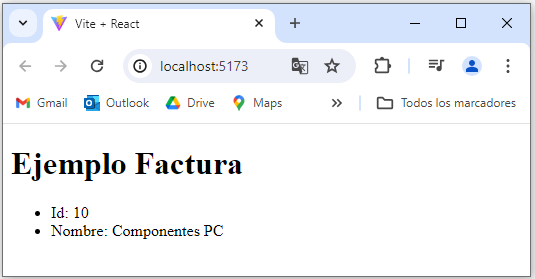
Paso 10: Implementar el servicio de factura en el componente InvoiceApp

El servicio creado solamente tiene la función de devolver la factura y será utilizado en el componente **InvoiceApp.jsx**. Por lo cual:

1. Importa la función **getInvoice** del archivo correspondiente en la carpeta de servicios. Por otro lado, se elimina la importación del archivo **invoice.js**.
2. Defina una constante **invoice**, el cual va a obtener el resultado de la función **getInvoice** (el objeto **invoice**)
3. Utilice la constante **invoice** para mostrar la información de la factura en el componente **InvoiceApp**.



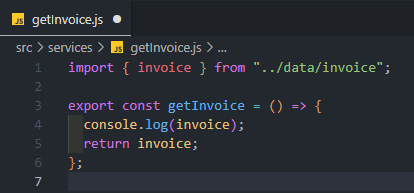




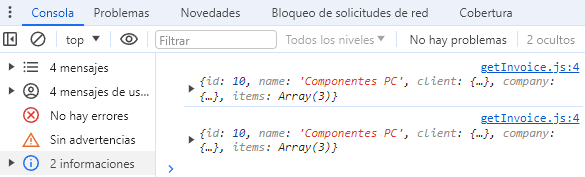
Además, se destaca que aunque actualmente se llama directamente a la función en el componente, esto no es una buena práctica, ya que cada vez que se modifica el componente se vuelve a realizar la llamada a la función. En futuras mejoras, se implementará un hook que se encargue de inicializar el objeto **invoice** solo una vez, al comienzo de la creación del componente, para evitar llamadas innecesarias al servicio.

Paso 11: Verificar la impresión de la factura en la consola

Una vez que se haya obtenido el título y los datos de la factura, es importante asegurar de que no haya errores en la consola del navegador. Para aquello se procede a definir un mensaje de registro en la consola en el archivo **getInvoice.js** para imprimir la factura.



Si aparece un error en la consola, se puede intentar guardar el componente padre **InvoiceApp.jsx** aunque no tenga cambios para aplicar todos los cambios realizados.



Es importante tener en cuenta que en la consola se imprimirá el objeto **invoice** dos veces debido a la configuración de **<React.StrictMode>** en el archivo **main.jsx**.

Aunque ese modo estricto es importante durante el desarrollo para realizar varios controles en el código, al entrar en producción, el segundo mensaje en la consola no aparecerá (Más información en la documentación de React: <https://legacy.reactjs.org/docs/strict-mode.html>).

Por lo tanto, no se elimina **<React.StrictMode>**, ya que su ejecución adicional es para verificar que el código funcione correctamente y para detectar posibles problemas.

