Composición, dividiendo componente en subcomponentes I

La aplicación de factura actualmente tiene un problema de legibilidad y mantenibilidad debido a la mezcla de detalles de factura, datos del cliente, empresa y productos en un solo componente.

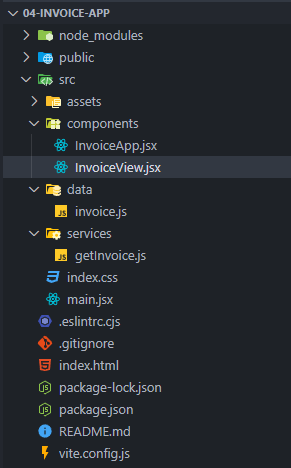
La idea de los componentes de React es mantener la simplicidad, modularidad y reutilización del código. La solución propuesta en esta sección es dividir el componente en subcomponentes más pequeños y especializados, para facilitar la comprensión y la reutilización.

Pasos 31 al 39

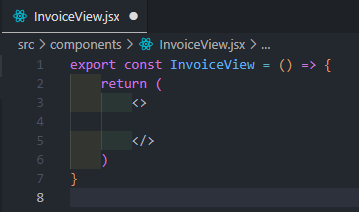
Sigue estos pasos:

Paso 31: Crear un componente para los datos de la factura

Crea un nuevo archivo en la carpeta de componentes llamado **InvoiceView.jsx** (o **InvoiceDetail.jsx**) para manejar los datos de la factura.



Dentro de este componente, define una función que va a devolver un fragmento de código HTML vacío por ahora.



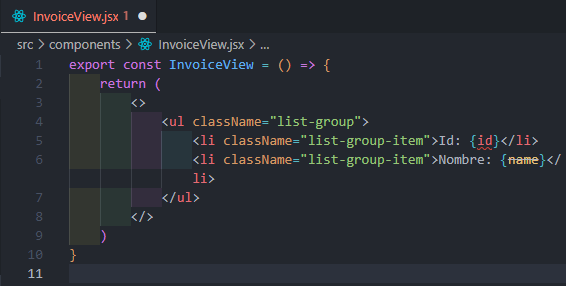
Paso 32: Trasladar los detalles de la factura al nuevo componente

Es importante tener en cuenta que los elementos **<div>** con las clases **container**, **card** **my-3** y **card-header** pertenecen a **InvoiceApp.jsx**, no al detalle de la factura. Además, la clase **card-body** pertenece a la clase **card** y no debe estar dentro del nuevo componente.



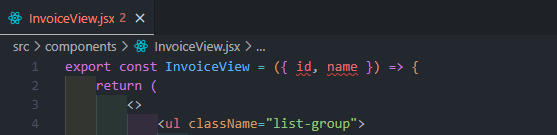


Los detalles de la factura se representan mediante los elementos **<ul>** y **<li>** que contienen el ID y el nombre de la factura. Estos detalles se seleccionan, cortan y pegan en el nuevo componente **InvoiceView**.



Paso 33: Implementar el componente InvoiceView

En el componente **InvoiceView**, se pasan las props **id** y **name** como un parámetro de la función. Se utiliza la desestructuración para obtener **id** y **name** de las props.

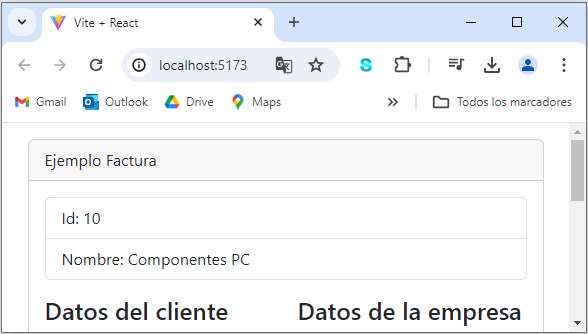


En el archivo **InvoiceApp.jsx**, se importa y utiliza el componente **InvoiceView**. Se asegura de pasar los valores del objeto **invoice** (mediante la función **getInvoice**) a las props **id** y **name** del componente **InvoiceView**, el cual se renderiza dentro de **InvoiceApp**.

Es importante notar que **InvoiceApp.jsx** y **InvoiceView.jsx** están en la misma carpeta, por lo que la importación es directa (recuerda mantener abierto ambos archivos para que la importación sea automática cuando se escriba el nombre del componente).

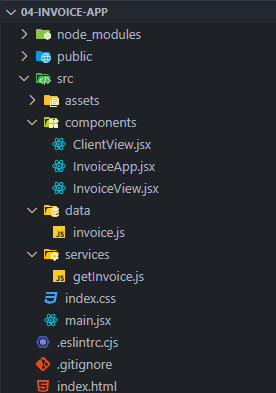


Se guarda el archivo y se verifica que la aplicación web funcione correctamente, sin cambios ni errores en la consola.

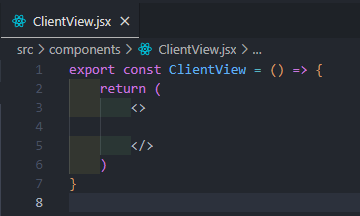


Paso 34: Crear un componente para los datos del cliente

Crea un archivo en la carpeta de componentes llamado **ClientDetails.jsx** o **ClientView.jsx**.



Dentro de este archivo, se define una función exportable que devuelve un fragmento vacío (por el momento).



Paso 35: Trasladar los datos del cliente al nuevo componente

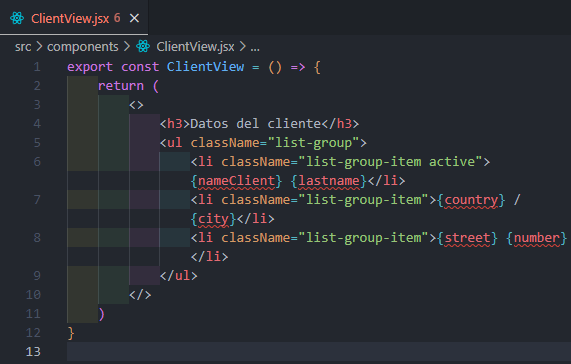
Selecciona el título y los datos del cliente, se encuentran dentro de las etiquetas **<h3>** y **<ul>** en el componente **InvoiceApp**.

Recordar que con bootstrap se han definido los datos del cliente y los datos de la empresa dentro de una fila (con la clase **row**) y cada uno de sus elementos contenidos se encuentran dentro de una columna (con la clase **col**), en el navegador se muestran uno al lado del otro.





Corta los datos seleccionados y pegalo en el componente **ClientView**.



Paso 36: Asignar el objeto client como propiedad al componente ClientView

En el componente **ClientView**, se pasa el objeto **client** como una prop desestructurada.

export const ClientView = ({ client }) => {
    return (
        <>
            <h3>Datos del cliente</h3>
            <ul className="list-group">
                <li className="list-group-item active">{nameClient} {lastname}</li>
                <li className="list-group-item">{country} / {city}</li>
                <li className="list-group-item">{street} {number}</li>
            </ul>
        </>
    )
}


Dentro de **ClientView**, se procede a desestructurar el objeto anidado **client** para acceder a sus props cómo **nameClient**, **lastname**, **address**, etc. Corta la desestructuración del objeto **client** desde **InvoiceApp** y pegalo dentro del componente **ClientView**.





Paso 37: Agregar la dirección del cliente al componente ClientView

En el componente **ClientView**, se puede desestructurar el objeto **address** para acceder a sus detalles, incluida la dirección.

Corta y pega la desestructuración de la dirección del cliente desde **InvoiceApp** al componente **ClientView**.

// import { invoice } from "../data/invoice"
import { getInvoice } from "../services/getInvoice"
import { InvoiceView } from "./InvoiceView";

export const InvoiceApp = () => {

    const { id, name, client, company, items } = getInvoice()
    const { country, city, street, number } = address;

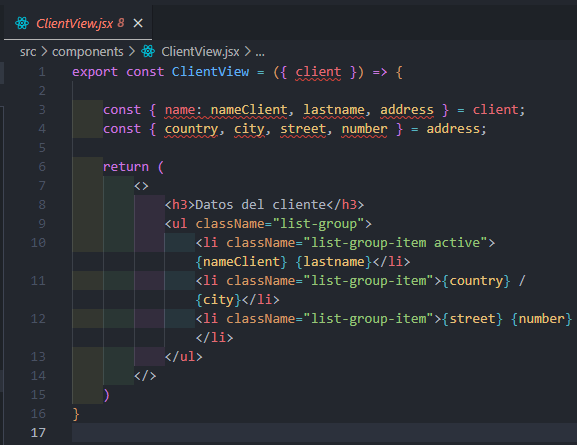
    return (
        <>
            <div className="container">
                <div className="card my-3">
                    <div className="card-header">
                        Ejemplo Factura
                    </div>
                    <div className="card-body">
                        <InvoiceView id={id} name={name} />

                        <div className="row my-3">
                            <div className="col">

                            </div>

                            <div className="col">
                                <h3>Datos de la empresa</h3>
                                <ul className="list-group">
                                    <li className="list-group-item active">{company.name}</li>
                                    <li className="list-group-item">{company.fiscalNumber}</li>
                                </ul>
                            </div>
                        </div>

                        <h3>Productos de la factura</h3>
                        <table className="table table-striped table-hover">
                            <thead>
                                <tr>
                                    <th>Producto</th>
                                    <th>Precio</th>
                                    <th>Cantidad</th>
                                </tr>
                            </thead>
                            <tbody>
                                {items.map(({ id, product, price, quantity }) => (
                                    <tr key={id}>
                                        <td>{product}</td>
                                        <td>{price}</td>
                                        <td>{quantity}</td>
                                    </tr>
                                ))}
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </>
    )
}



En el componente **InvoiceApp.jsx**, importa el subcomponente **ClientView** y llamalo dentro de una de las columnas. Luego asigna el objeto anidado **client** como un valor para la prop **client** que se va a renderizar en el componente **ClientView**.

// import { invoice } from "../data/invoice"
import { getInvoice } from "../services/getInvoice"
import { ClientView } from "./ClientView";
import { InvoiceView } from "./InvoiceView";

export const InvoiceApp = () => {

    const { id, name, client, company, items } = getInvoice()

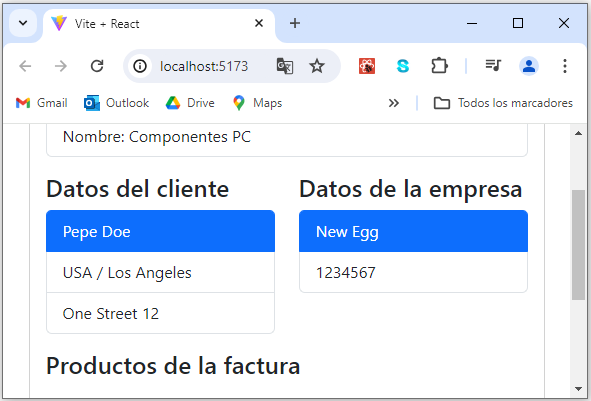
    return (
        <>
            <div className="container">
                <div className="card my-3">
                    <div className="card-header">
                        Ejemplo Factura
                    </div>
                    <div className="card-body">
                        <InvoiceView id={id} name={name} />

                        <div className="row my-3">
                            <div className="col">
                                <ClientView client={client} />
                            </div>

                            <div className="col">
                                <h3>Datos de la empresa</h3>
                                <ul className="list-group">
                                    <li className="list-group-item active">{company.name}</li>
                                    <li className="list-group-item">{company.fiscalNumber}</li>
                                </ul>
                            </div>
                        </div>

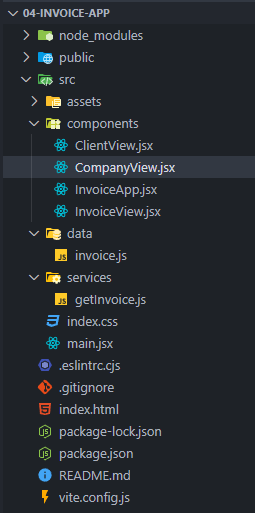
                        <h3>Productos de la factura</h3>
                        <table className="table table-striped table-hover">
                            <thead>
                                <tr>
                                    <th>Producto</th>
                                    <th>Precio</th>
                                    <th>Cantidad</th>
                                </tr>
                            </thead>
                            <tbody>
                                {items.map(({ id, product, price, quantity }) => (
                                    <tr key={id}>
                                        <td>{product}</td>
                                        <td>{price}</td>
                                        <td>{quantity}</td>
                                    </tr>
                                ))}
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </>
    )
}

Se guarda el archivo y se verifica que la aplicación web funcione correctamente, sin cambios ni errores en la consola.

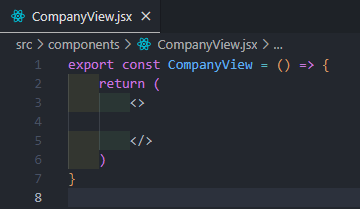


Paso 38: Incorporar los detalles de la empresa en el componente CompanyView

Crea el componente **CompanyView.jsx** dentro de la carpeta de **components**.



Define una función de flecha exportable que va a retornar un fragmento vacío.



En **InvoiceApp.jsx**, corta todo el código que contiene el título y los detalles de la empresa y pegalo en el componente **CompanyView**.

// import { invoice } from "../data/invoice"
import { getInvoice } from "../services/getInvoice"
import { ClientView } from "./ClientView";
import { InvoiceView } from "./InvoiceView";

export const InvoiceApp = () => {

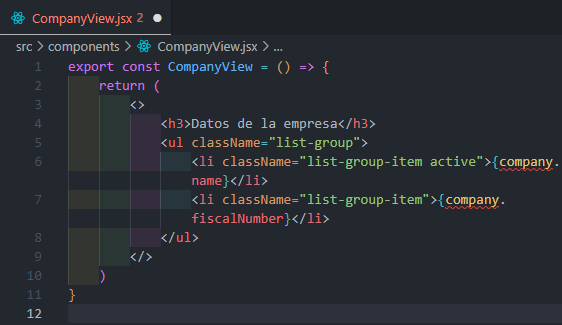
    const { id, name, client, company, items } = getInvoice()

    return (
        <>
            <div className="container">
                <div className="card my-3">
                    <div className="card-header">
                        Ejemplo Factura
                    </div>
                    <div className="card-body">
                        <InvoiceView id={id} name={name} />

                        <div className="row my-3">
                            <div className="col">
                                <ClientView client={client} />
                            </div>

                            <div className="col">
                                <h3>Datos de la empresa</h3>
                                <ul className="list-group">
                                    <li className="list-group-item active">{company.name}</li>
                                    <li className="list-group-item">{company.fiscalNumber}</li>
                                </ul>
                            </div>
                        </div>

                        <h3>Productos de la factura</h3>
                        <table className="table table-striped table-hover">
                            <thead>
                                <tr>
                                    <th>Producto</th>
                                    <th>Precio</th>
                                    <th>Cantidad</th>
                                </tr>
                            </thead>
                            <tbody>
                                {items.map(({ id, product, price, quantity }) => (
                                    <tr key={id}>
                                        <td>{product}</td>
                                        <td>{price}</td>
                                        <td>{quantity}</td>
                                    </tr>
                                ))}
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </>
    )
}

En **InvoiceApp.jsx**, importa el componente **CompanyView** y asígnale el objeto anidado **company** a la prop **company** dentro del componente que se va a renderizar.

// import { invoice } from "../data/invoice"
import { getInvoice } from "../services/getInvoice"
import { ClientView } from "./ClientView";
import { CompanyView } from "./CompanyView";
import { InvoiceView } from "./InvoiceView";

export const InvoiceApp = () => {

    const { id, name, client, company, items } = getInvoice()

    return (
        <>
            <div className="container">
                <div className="card my-3">
                    <div className="card-header">
                        Ejemplo Factura
                    </div>
                    <div className="card-body">
                        <InvoiceView id={id} name={name} />

                        <div className="row my-3">
                            <div className="col">
                                <ClientView client={client} />
                            </div>

                            <div className="col">
                                <CompanyView company={company}/>
                            </div>
                        </div>

                        <h3>Productos de la factura</h3>
                        <table className="table table-striped table-hover">
                            <thead>
                                <tr>
                                    <th>Producto</th>
                                    <th>Precio</th>
                                    <th>Cantidad</th>
                                </tr>
                            </thead>
                            <tbody>
                                {items.map(({ id, product, price, quantity }) => (
                                    <tr key={id}>
                                        <td>{product}</td>
                                        <td>{price}</td>
                                        <td>{quantity}</td>
                                    </tr>
                                ))}
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </>
    )
}


// import { invoice } from "../data/invoice"
import { getInvoice } from "../services/getInvoice"
import { ClientView } from "./ClientView";
import { CompanyView } from "./CompanyView";
import { InvoiceView } from "./InvoiceView";

export const InvoiceApp = () => {

    const { id, name, client, company, items } = getInvoice()

    return (
        <>
            <div className="container">
                <div className="card my-3">
                    <div className="card-header">
                        Ejemplo Factura
                    </div>
                    <div className="card-body">
                        <InvoiceView id={id} name={name} />

                        <div className="row my-3">
                            <div className="col">
                                <ClientView client={client} />
                            </div>

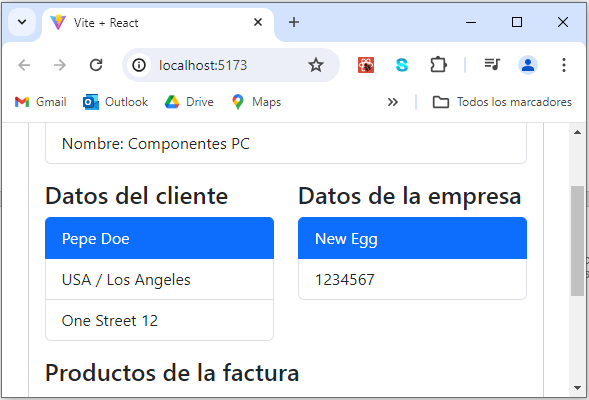
                            <div className="col">
                                <CompanyView company={company}/>
                            </div>
                        </div>

                        <h3>Productos de la factura</h3>
                        <table className="table table-striped table-hover">
                            <thead>
                                <tr>
                                    <th>Producto</th>
                                    <th>Precio</th>
                                    <th>Cantidad</th>
                                </tr>
                            </thead>
                            <tbody>
                                {items.map(({ id, product, price, quantity }) => (
                                    <tr key={id}>
                                        <td>{product}</td>
                                        <td>{price}</td>
                                        <td>{quantity}</td>
                                    </tr>
                                ))}
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </>
    )
}


Pasa el objeto **company** como una prop dentro del componente **CompanyView** para acceder a los detalles de la empresa (no se utiliza la desestructuración).

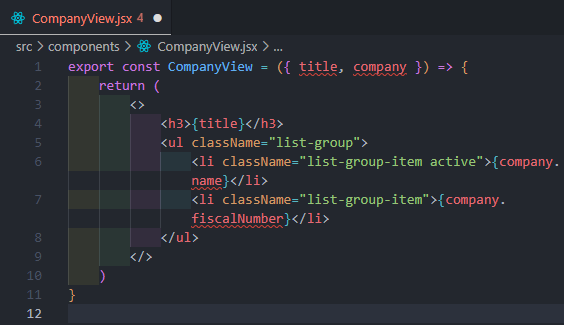


Como resultado, el componente **CompanyView** va a mostrar los detalles de la empresa en el navegador.



Paso 39: Desacoplar los títulos de los componentes ClientView y CompanyView

En el componente **CompanyView.jsx**, elimina el título **"Datos de la empresa"**, que está contenido en las etiquetas **<h3>**. Luego define una nueva prop llamada **title**, el cual va a servir para mostrar el título contenido en el elemento **<h3>**.



En **InvoiceApp.jsx**, defina un valor para el título como un string para asignarlo al componente **CompanyView** al momento de llamarlo.

// import { invoice } from "../data/invoice"
import { getInvoice } from "../services/getInvoice"
import { ClientView } from "./ClientView";
import { CompanyView } from "./CompanyView";
import { InvoiceView } from "./InvoiceView";

export const InvoiceApp = () => {

    const { id, name, client, company, items } = getInvoice()

    return (
        <>
            <div className="container">
                <div className="card my-3">
                    <div className="card-header">
                        Ejemplo Factura
                    </div>
                    <div className="card-body">
                        <InvoiceView id={id} name={name} />

                        <div className="row my-3">
                            <div className="col">
                                <ClientView client={client} />
                            </div>

                            <div className="col">
                                <CompanyView title="Datos de la empresa" company={company} />
                            </div>
                        </div>

                        <h3>Productos de la factura</h3>
                        <table className="table table-striped table-hover">
                            <thead>
                                <tr>
                                    <th>Producto</th>
                                    <th>Precio</th>
                                    <th>Cantidad</th>
                                </tr>
                            </thead>
                            <tbody>
                                {items.map(({ id, product, price, quantity }) => (
                                    <tr key={id}>
                                        <td>{product}</td>
                                        <td>{price}</td>
                                        <td>{quantity}</td>
                                    </tr>
                                ))}
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </>
    )
}

Repite el mismo proceso para el componente **ClientView.jsx**, crea la prop **title**, elimina el texto contenido en las etiquetas **<h3>**, y reemplazalo con el valor que se asignará desde la prop **title**. Por otro lado, se define un valor para el título en **InvoiceApp.jsx**.

export const ClientView = ({ title, client }) => {

    const { name: nameClient, lastname, address } = client;
    const { country, city, street, number } = address;

    return (
        <>
            <h3>{title}</h3>
            <ul className="list-group">
                <li className="list-group-item active">{nameClient} {lastname}</li>
                <li className="list-group-item">{country} / {city}</li>
                <li className="list-group-item">{street} {number}</li>
            </ul>
        </>
    )
}


// import { invoice } from "../data/invoice"
import { getInvoice } from "../services/getInvoice"
import { ClientView } from "./ClientView";
import { CompanyView } from "./CompanyView";
import { InvoiceView } from "./InvoiceView";

export const InvoiceApp = () => {

    const { id, name, client, company, items } = getInvoice()

    return (
        <>
            <div className="container">
                <div className="card my-3">
                    <div className="card-header">
                        Ejemplo Factura
                    </div>
                    <div className="card-body">
                        <InvoiceView id={id} name={name} />

                        <div className="row my-3">
                            <div className="col">
                                <ClientView title="Datos del cliente" client={client} />
                            </div>

                            <div className="col">
                                <CompanyView title="Datos de la empresa" company={company} />
                            </div>
                        </div>

                        <h3>Productos de la factura</h3>
                        <table className="table table-striped table-hover">
                            <thead>
                                <tr>
                                    <th>Producto</th>
                                    <th>Precio</th>
                                    <th>Cantidad</th>
                                </tr>
                            </thead>
                            <tbody>
                                {items.map(({ id, product, price, quantity }) => (
                                    <tr key={id}>
                                        <td>{product}</td>
                                        <td>{price}</td>
                                        <td>{quantity}</td>
                                    </tr>
                                ))}
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </>
    )
}



Con esta refactorización, los componentes **ClientView** y **CompanyView** tienen títulos más flexibles y reutilizables.

