Introducción al desarrollo Web con Ant Design

Create

Para crear la última solicitud para el CRUD vamos a construir la misma en el archivo services/products.js recuerda que este también es un end point protegido

```
export const createProduct = async (productData) => {
    try {
        const url = `${ENV.API_URL}/${ENV.ENDPOINTS.PRODUCTS}`;
        const token = storageController.getToken();
        if (!token) {
            throw new Error('No se encontró el token de autenticación');
        }
        const response = await axios.post(url, productData, {
            headers: {
                Authorization: `Bearer ${token}`,
                'Content-Type': 'application/json'
            }
        });
        return response.data;
    } catch (error) {
        console.error('Error al crear el producto', error);
        throw error;
    }
};
```

Una vez que creamos la solicitud vamos a crear el componente **componentes/FormNewProduct** éste contendrá el formulario para enviar los datos necesarios para crear un producto, como puedes observar el formulario manda los Ítem necesarios según la estructura de la Api pasando por Props **onSubmit** que será utilizado en el componente CrudButton que tiene el botón de acción para crear un nuevo producto, este botón abrirá el modal que contendrá este formulario.

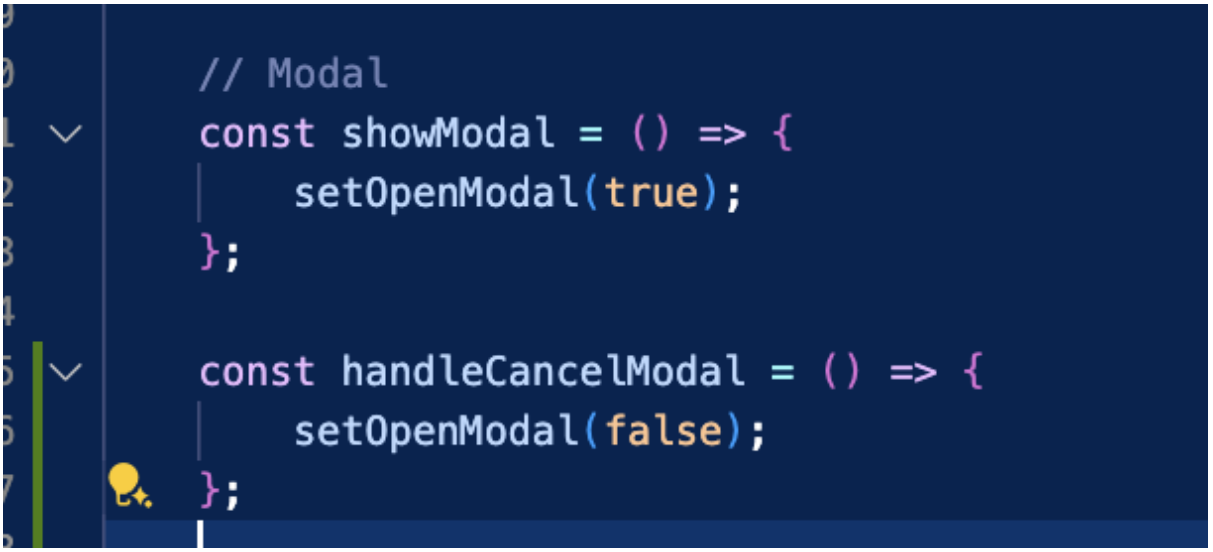
```jsx
import React from 'react';
import { Form, Input, Button } from 'antd';

const FormNewProduct = ({ onSubmit }) => {


    const onFinish = (values) => {
        onSubmit(values);
    };

    return (
        <Form

            onFinish={onFinish}
            labelCol={{ span: 8 }}
            wrapperCol={{ span: 14 }}
            layout="horizontal"
            initialValues={{}}
            style={{ maxWidth: 800, marginTop: 40 }}
        >
            <Form.Item
                name="name"
                label="Nombre"
                rules={[{ required: true, message: 'Por favor ingrese el nombre del producto' }]}
            >
                <Input />
            </Form.Item>
            <Form.Item
                name="price"
                label="Precio"
                rules={[{ required: true, message: 'Por favor ingrese el precio del producto' }]}
            >
                <Input type="number" />
            </Form.Item>
            <Form.Item
                name="category"
                label="Categoría"
                rules={[{ required: true, message: 'Por favor ingrese la categoría del producto' }]}
            >
                <Input />
            </Form.Item>
            <Form.Item
                name="imgURL"
                label="URL de la imagen"
                rules={[{ required: true, message: 'Por favor ingrese la URL de la imagen del producto' }]}
            >
                <Input />
            </Form.Item>
            <Form.Item
                wrapperCol={{ offset: 8, span: 14 }}
                style={{ display: 'flex', justifyContent: 'center', }}
            >
                <Button type="primary" htmlType="submit" style={{ width: 160, fontSize: 16, height: 35 }}>
                    Guardar
                </Button>
            </Form.Item>
        </Form>
    );
};

export default FormNewProduct;
```

Ahora en el componente CrudButton asegurate de importar los componentes necesarios de Antd ***import { Col, Row, Button, Popconfirm, Modal, message } from 'antd';*** así como la solicitud a la Api de services ***import { createProduct } from '../../services/products';***

Crea el estado para abrir y cerrar el Modal

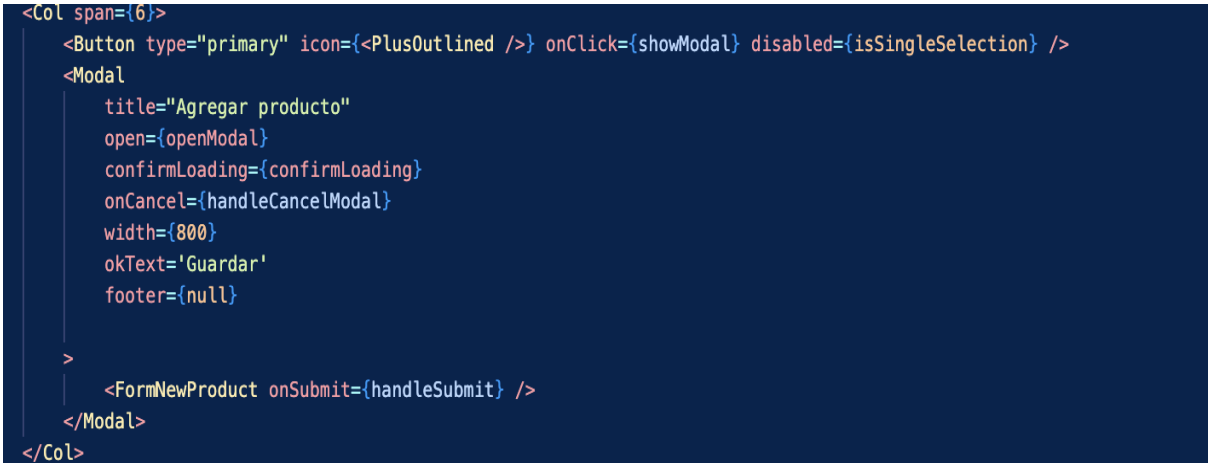
const [openModal, setOpenModal] = useState(false);

Crea las funciones para mostrar y cerrar el modal



```
// Modal
const showModal = () => {
    setOpenModal(true);
};

const handleCancelModal = () => {
    setOpenModal(false);
};
```

Ahora en el botón de agregar modificar las propiedades de onclick, que accionara showModal, también estoy deshabilitando el botón cuando exista un elemento seleccionado, esta función ya la creamos anteriormente, observa que dentro del modal está el componente FormNewProduct que tiene como propiedad onSubmit que ya trajimos por props, esta acciona la función **handleSubmit** que será la encargada de recibir los datos del formulario y realizar la solicitud a la Api.



```
<Col span={6}>
    <Button type="primary" icon={<PlusOutlined />} onClick={showModal} disabled={isSingleSelection} />
    <Modal
        title="Agregar producto"
        open={openModal}
        confirmLoading={confirmLoading}
        onCancel={handleCancelModal}
        width={800}
        okText='Guardar'
        footer={null}

    >
        <FormNewProduct onSubmit={handleSubmit} />
    </Modal>
</Col>
```

La función handleSubmit se compone de la obtención de formData (los datos del formulario FormNewProduct) se asegura que el campo price sea de tipo number, actualiza el estado y hace la solicitud a createProduct, mandando como argumento la data, envía un mensaje de confirmación y además hace la solicitud a fetchProducts para asegurarnos que el nuevo

producto creado se muestre automáticamente en la Tabla de productos. Dentro del catch se maneja el error en la solicitud. No te olvides de pasar fetchProducts por props al componente CrudButton

```
const CrudButton = ({ isSingleSelection, onDelete, onEdit, fetchProducts }) => {
```

```
    const handleSubmit = async (formData) => {
        console.log('Formulario', formData);
        const productData = {
            ...formData,
            price: Number(formData.price),
        };
        try {
            setConfirmLoading(true);
            await createProduct(productData);
            setOpenModal(false);
            setConfirmLoading(false);
            message.success('Producto creado correctamente');
            fetchProducts();  // Actualizar los productos después de crear uno nuevo
        } catch (error) {
            console.error('Error al crear el producto', error.response?.data || error.message);
            setConfirmLoading(false);
        }
    };
```