	UNIVERSIDAD DON BOSCO **FACULTAD DE INGENIERÍA** **ESCUELA DE COMPUTACIÓN**	
CICLO: 02	**GUIA DE LABORATORIO #03**	
	Nombre de la Práctica:	**React, parte II**
	MATERIA:	**Diseño y Programación de Software Multiplataforma**

I. OBJETIVOS

Que el estudiante:

- Diseñe aplicaciones web utilizando funciones de React.
- Diseñe aplicaciones con navegación.
- Aprender hacer interfaz para que el usuario pueda interactuar con la aplicación.
- Aprender a crear componente personalizado.
- Aprender hacer interfaz para que el usuario pueda interactuar con la aplicación.
- Aprender a utilizar pipes dentro de la interfaz de usuario.
- Hacer uso de los componentes en el diseño de interfaz React.

II. INTRODUCCION TEORICA

¿Qué es Redux?

Redux es una herramienta para la gestión de estado en apps Javascript que nació en 2015. Aunque suele asociarse a React, lo cierto es que es una **librería *framework agnostic*.**

Cómo funciona Redux

Antes de entrar al detalle, déjame hacer hincapié en los 3 principios de Redux que lo convierten en un contenedor predecible de estados.

Los principios fundamentales de Redux

- **Fuente única de verdad**: En Redux hay un único objeto que almacena el estado de toda la aplicación. Esto ayuda a la hora de trabajar con apps universales, así como a la hora de debugar y de reiniciar el desarrollo en un punto concreto de ejecución.

- **Inmutabilidad, el estado es *read-only***. Ninguna interacción puede cambiarlo directamente. Lo único que puedes hacer para conseguirlo es emitir una **acción** que expresa su intención

de cambiarlo.

- **Funciones puras**: Usa funciones puras (a mismos inputs, mismos outputs) para definir cómo cambia el estado en base a una **acción**. En Redux estas funciones se conocen como **reducers** y al ser puras, su comportamiento es predecible.

Flujo de actualizaciones en Redux

Aquí tienes un diagrama con el flujo que sigue Redux desde una interacción, hasta que la aplicación actualiza la UI.

Es decir:

1. El **componente** recibe un evento (click, por ejemplo) y emite una acción.
2. Esta **acción**, se pasa a la *store*, que es donde se guarda el estado único.
3. La ***store*** comunica la acción junto con el estado actual a los reducers.
4. Los **reducers**, devuelven un nuevo estado, probablemente modificado en base a la acción.
5. Los componentes reciben el **nuevo estado** de la *store*.

Middleware

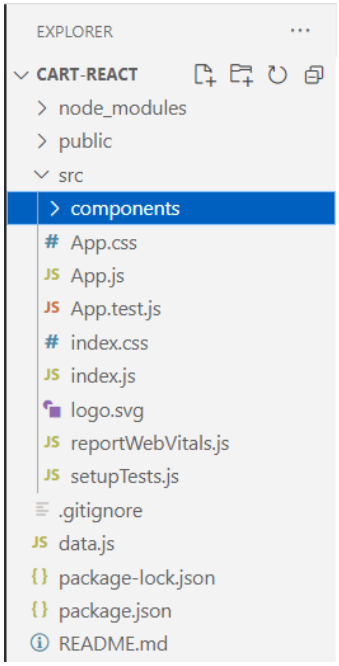
El *middleware* en React proporciona un punto de extensión para terceros entre el envío de una acción y el momento en que alcanza el reducer.

¿Que cosas se pueden hacer metiendo mano ahí en medio? Pues registrar eventos, generar informes de fallos, realizar llamadas a una API asíncrona, enrutamiento y básicamente lo que se te ocurra.

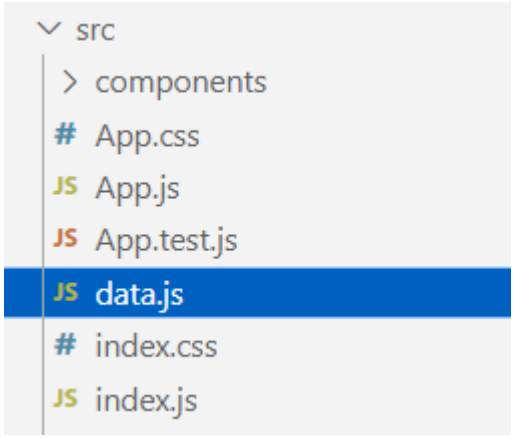
Crear un *middleware* desde cero es un paso más avanzado y no te lo detallaré ahora, pero ya te avanzo que pocas veces tienes que hacerlo de cero: Hay cantidad de proyectos *open source* que proporcionan *middlewares* para todo lo que imagines en Redux.

IV. PROCEDIMIENTO

1. realizar la instalación de React: **npm install -g create-react-app**
2. Crear nuestro Proyecto: **create-react-app cart-react**
3. En nuestra carpeta de proyecto src, vamos crear una nueva carpeta con el nombre components



4. En nuestra carpeta src, agregar un nuevo archivo de nombre data.js

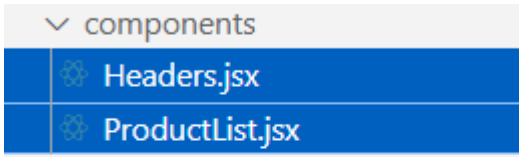


5. Agregar el siguiente código en el archivo data.js

```
export const data=[
    {
        id:1,
        title:'Cien años de soledad',
        price:100,
        urlImage:'https://images.penguinrandomhouse.com/cover/9780525562443',
        quantity:1
    },
    {
        id:2,
        title:'El señor de los anillos(Trilogía)',
```

```
        price:190,

urlImage:'https://proassetspdlcom.cdnstatics2.com/usuaris/libros/fotos/358/original/portada_pack-trilogia-el-senor-de-los-anillos_j-r-r-tolkien_202206071544.jpg',
        quantity:1
    },
        {
        id:3,
        title:'Cuentos de Barro',
        price:30,

urlImage:'https://www.librosdelaballena.com/wp-content/uploads/2020/05/cuentos-barro-244x300.png',
        quantity:1
    },
       {
        id:4,
        title:'Tierra de Infancia',
        price:30,

urlImage:'https://marketsvstg.blob.core.windows.net/marketsv/0011705_tierra-de-infancia_510.jpeg',
        quantity:1
    },   {
        id:5,
        title:'Harry Potter Pack',
        price:390,

urlImage:'https://contentv2.tap-commerce.com/cover/large/9789878000473_1.jpg?id_com=1113',
        quantity:1
    }
];
```

6. Vamos a crear dos archivos en nuestra carpeta componentes: Header.jsx y ProductList.jsx



7. Agregar el siguiente código en Header.jsx

```
import React from "react";

export const Headers=()=>{
    return (
    <header>
    <h1>Tienda de Libros</h1>

    <div className="container-icon">
        <div className="container-cart-icon">
```

```
<img
src=”https://e7.pngegg.com/pngimages/833/426/png-clipart-black-shopping-cart-icon
-for-free-black-shopping-cart.png” alt="carrito" className="icon-cart" />
            <div className="count-products">
                <span id="contador-productos">0</span>
            </div>
        </div>

        <div className="container-cart-products hidden-cart">
            <div className="row-product hidden">
                <div className="cart-product">
                    <div className="info-cart-product">
                        <span className="cantidad-producto-carrito">1</span>
                        <p className="titulo-producto-carrito">Don Quijote</p>
                        <span className="precio-producto-carrito">$80</span>
                    </div>
                          <img
src="https://static.vecteezy.com/system/resources/previews/018/887/462/original/s
igns-close-icon-png.png"  alt="cerrar"  className="icon-close"     />
                </div>
            </div>

            <div className="cart-total hidden">
                <h3>Total:</h3>
                <span className="total-pagar">$200</span>
            </div>
            <p className="cart-empty">El carrito está vacío</p>
        </div>
    </div>
</header>);
}
```

8. Agregar el siguiente código en ProductList.jsx

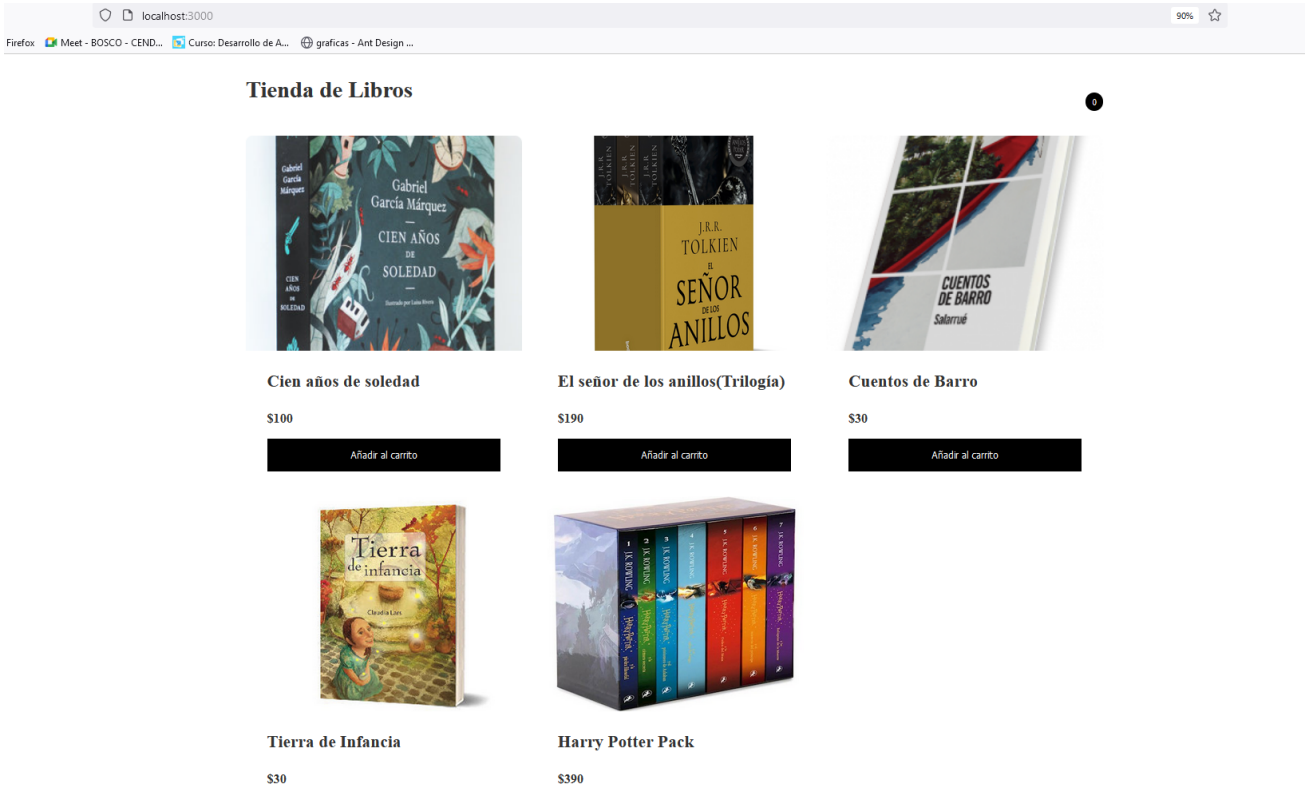
```
import React from "react";
import {data} from "../data";
export const ProductList=()=>{
    return (
        <div className="container-items">
           {data.map(product =>(
                <div className="item" key={product.id}>
                <figure>
                    <img src={product.urlImage} alt={product.title}/>
                </figure>
                <div className="info-product">
                    <h2>{product.title}</h2>
                    <p className="price">${product.price}</p>
                    <button className="btn-add-cart">Añadir al carrito</button>
                </div>
            </div>
            ))}
        </div>
```

```
    );
}
```

9. Sustituir el archivo de index.css por el proporcionado en los recursos de la guía.
10. Modificar el archivo app.jsx con el siguiente código:

```
import {Headers} from './components/Headers';
import {ProductList} from './components/ProductList';

function App() {
  return (
    <>
     <Headers></Headers>
     <ProductList></ProductList>
    </>
  );
}

export default App;
```

11. Con estas modificaciones la aplicación debería verse así:



12. Modificar el App.js para agregar funcionalidad del carrito de compra:

```
import { useState } from 'react';
import { Header } from './components/Headers';
import { ProductList } from './components/ProductList';

function App() {
```

```
  const [allProducts, setAllProducts] = useState([]);
  const [total, setTotal] = useState(0);
  const [countProducts, setCountProducts] = useState(0);

  return (
    <>
      <Header
        allProducts={allProducts}
        setAllProducts={setAllProducts}
        total={total}
        setTotal={setTotal}
        countProducts={countProducts}
        setCountProducts={setCountProducts}
      />
      <ProductList
        allProducts={allProducts}
        setAllProducts={setAllProducts}
        total={total}
        setTotal={setTotal}
        countProducts={countProducts}
        setCountProducts={setCountProducts}
      />
    </>
  );
}

export default App;
```

13. Modificaremos el Headers.jsx

```
import { useState } from 'react';

export const Header = ({
    allProducts,
    setAllProducts,
    total,
    countProducts,
    setCountProducts,
    setTotal,
}) => {
    const [active, setActive] = useState(false);

    const onDeleteProduct = product => {
        const results = allProducts.filter(
            item => item.id !== product.id
        );

        setTotal(total - product.price * product.quantity);
        setCountProducts(countProducts - product.quantity);
        setAllProducts(results);
    };

    const onCleanCart = () => {
```

```
        setAllProducts([]);
        setTotal(0);
        setCountProducts(0);
    };

    return (
        <header>
            <h1>Tienda de Libros</h1>

            <div className='container-icon'>
                <div
                    className='container-cart-icon'
                    onClick={() => setActive(!active)}
                >
                    <img
src="https://e7.pngegg.com/pngimages/833/426/png-clipart-black-shopping-cart-icon
-for-free-black-shopping-cart.png"
                    alt="carrito"
                    className="icon-cart" />




                    <div className='count-products'>
                        <span id='contador-productos'>{countProducts}</span>
                    </div>
                </div>

                <div
                    className={`container-cart-products ${
                        active ? '' : 'hidden-cart'
                    }`}
                >
                    {allProducts.length ? (
                        <>
                            <div className='row-product'>
                                {allProducts.map(product => (
                                    <div className='cart-product'
key={product.id}>
                                        <div className='info-cart-product'>
                                            <span
className='cantidad-producto-carrito'>
                                                {product.quantity}
                                            </span>
                                            <p
className='titulo-producto-carrito'>
                                                {product.title}
                                            </p>
                                            <span
className='precio-producto-carrito'>
                                                ${product.price}
                                            </span>
```

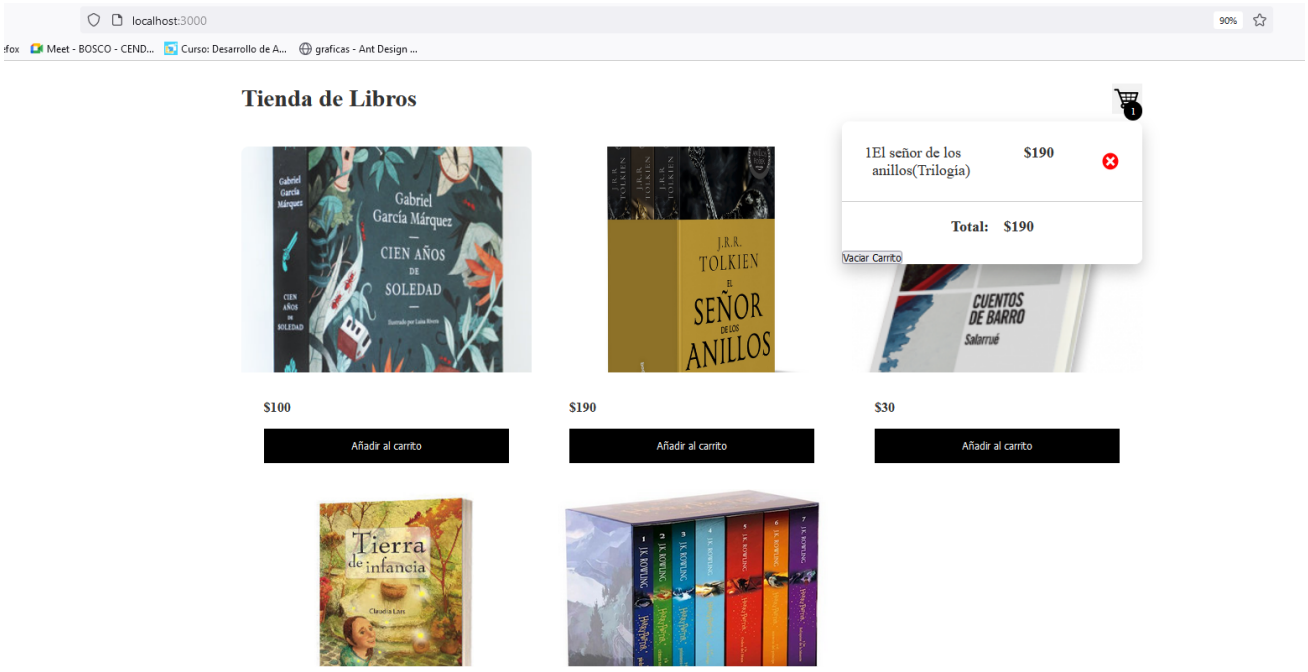
```
                                        </div>
                                        <img
src="https://static.vecteezy.com/system/resources/previews/018/887/462/original/signs-close-icon-png.png"
                                        alt="cerrar"
                                        className="icon-close"
                                        onClick={() => onDeleteProduct(product)}
                                        />

                                    </div>
                                ))}
                            </div>

                            <div className='cart-total'>
                                <h3>Total:</h3>
                                <span className='total-pagar'>${total}</span>
                            </div>

                            <button className='btn-clear-all'
onClick={onCleanCart}>
                                Vaciar Carrito
                            </button>
                        </>
                    ) : (
                        <p className='cart-empty'>El carrito está vacío</p>
                    )}
                </div>
            </div>
        </header>
    );
};
```

14. Modificaremos el ProductList.jsx

```
import React from "react";
import {data} from "../data";

export const ProductList = ({
    allProducts,
    setAllProducts,
    countProducts,
    setCountProducts,
    total,
    setTotal,
}) => {
    const onAddProduct = product => {
        if (allProducts.find(item => item.id === product.id)) {
            const products = allProducts.map(item =>
                item.id === product.id
                    ? { ...item, quantity: item.quantity + 1 }
                    : item
            );
            setTotal(total + product.price * product.quantity);
```

```
            setCountProducts(countProducts + product.quantity);
            return setAllProducts([...products]);
        }

        setTotal(total + product.price * product.quantity);
        setCountProducts(countProducts + product.quantity);
        setAllProducts([...allProducts, product]);
    };

    return (
        <div className='container-items'>
            {data.map(product => (
                <div className='item' key={product.id}>
                    <figure>
                        <img src={product.urlImage} alt={product.title} />
                    </figure>
                    <div className='info-product'>
                        <h2>{product.nameProduct}</h2>
                        <p className='price'>${product.price}</p>
                        <button onClick={() => onAddProduct(product)}>
                            Añadir al carrito
                        </button>
                    </div>
                </div>
            ))}
        </div>
    );
};
```

15. La aplicación debería verse de la siguiente forma:



V. DISCUSION DE RESULTADOS

Modificaremos el ejercicio de la practica para que cuando se agreguen los productos también se pueda apreciar una imagen junto al nombre del producto.

VII. BIBLIOGRAFIA

- React, una biblioteca de JavaScript. (2013-2020). Facebook, Inc. Jordan Walke. Recuperado de https://es.reactjs.org/docs/getting-started.html