**UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA**

**UNAN – LEÓN**

****

**FACULTAD DE CIENCIAS Y TECNOLOGÍA**

**Carrera:** Ingeniería en Sistemas de Información

**Grupo:** GP1

**Componente:** Programación Orientada a la Web II

**Tema:** Mi primer Proyecto en React

**Elaborado por:** Br. Cristian José Oporta Cajina

**Enlace del proyecto:** https://bit.ly/3tpHmJ9

**Asignado por:** MSc. Juan Carlos Leyton Briones

**Fecha de inicio:** 23/09/2023

**“A La Libertad Por La Universidad”**

**23/09/2023**

**Introducción:**

La pastelería es un negocio que requiere llevar un control detallado de los productos, los clientes y las ventas. Por ello, es muy importante contar con una base de datos bien diseñada que permita almacenar y gestionar toda esta información de manera organizada y eficiente.

En este caso, se plantea crear una base de datos para una tienda de pastelería llamada "Bakery Store". El objetivo es poder registrar los distintos tipos de productos, los clientes, los pedidos y pagos, así como elementos adicionales como opiniones, listas de deseos y carritos de compra.

**Desarrollo:**

La base de datos "Bakery Store" consta de 11 tablas que permiten almacenar los datos necesarios. La tabla "categorías" sirve para clasificar los distintos tipos de productos. La tabla "productos" contiene todos los productos disponibles en la tienda con información como nombre, descripción, imagen, precio y categoría.

Para los clientes se crea la tabla "usuarios" con datos como nombre, email, contraseña y si es administrador o no. Los pedidos se almacenan en la tabla "pedidos" vinculada al usuario que realizó la compra. También existe una tabla "elementos\_pedido" que registra cada producto y cantidad incluidos en el pedido.

Otras tablas importantes son "carritos" y "elementos\_carrito" para llevar el control de lo que los usuarios agregan a su carrito antes de realizar la compra. Además, se crean tablas para almacenar opiniones, listas de deseos y los pagos realizados para cada pedido.

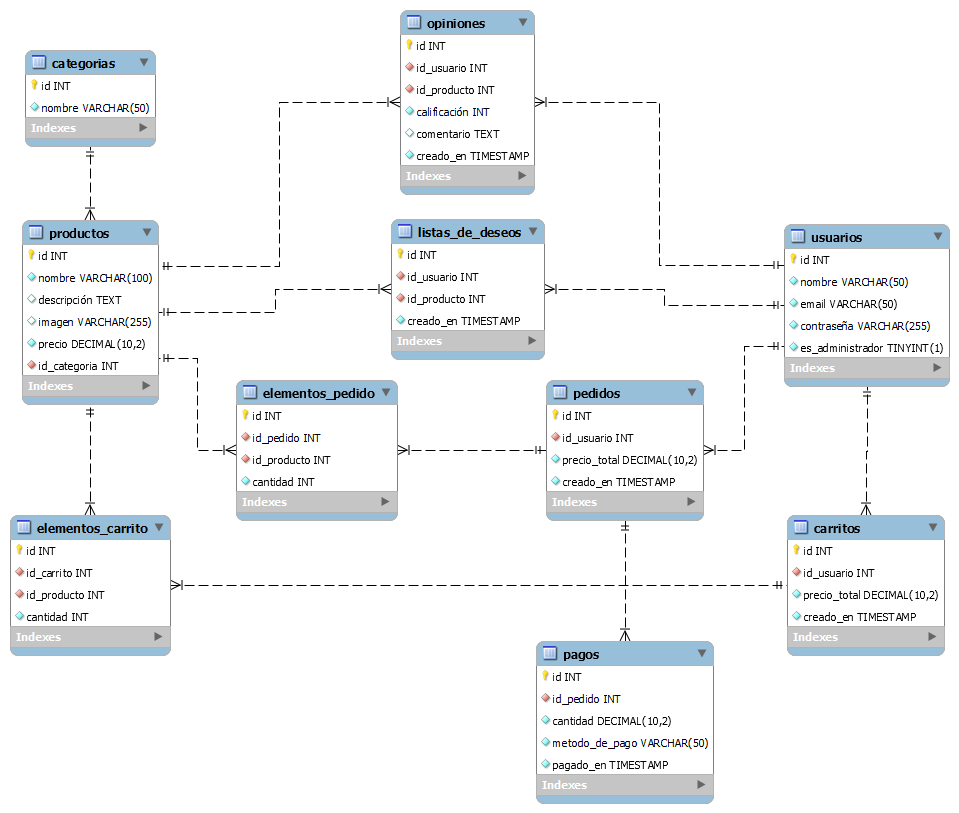
Cada tabla cuenta con llaves primarias para identificar los registros de manera única y llaves foráneas para vincular la información entre tablas. Por ejemplo, la tabla "pedidos" tiene una llave foránea que apunta al campo "id" de la tabla "usuarios" para saber qué usuario realizó cada pedido.

**Conclusión:**

El diseño propuesto para la base de datos "Bakery Store" permite almacenar de forma estructurada y relacionada toda la información necesaria para la gestión de una pastelería. Las 11 tablas cubren las necesidades de registro de productos, clientes, pedidos, opiniones, listas de deseos, carritos, pagos y más.

La implementación de esta base de datos en un sistema de información dará a la pastelería un mejor control y seguimiento de sus operaciones. Le permitirá tomar mejores decisiones comerciales y de inventarios, conocer mejor a sus clientes y optimizar procesos. Es una pieza clave para la administración exitosa de este tipo de negocio.

**1. Creación Diagrama Entidad-Relación**

****

**2. Script de la Base de Datos**

DROP DATABASE IF EXISTS bakery\_store;

CREATE DATABASE IF NOT EXISTS bakery\_store;

USE bakery\_store;

-- TABLA categorías

CREATE TABLE categorias (

id INT PRIMARY KEY AUTO\_INCREMENT,

nombre VARCHAR(50) NOT NULL

);

-- TABLA usuarios

CREATE TABLE usuarios (

id INT PRIMARY KEY AUTO\_INCREMENT,

nombre VARCHAR(50) NOT NULL,

email VARCHAR(50) NOT NULL UNIQUE,

contraseña VARCHAR(255) NOT NULL,

es\_administrador TINYINT(1) NOT NULL DEFAULT 0

);

-- TABLA productos

CREATE TABLE productos (

id INT PRIMARY KEY AUTO\_INCREMENT,

nombre VARCHAR(100) NOT NULL,

descripción TEXT,

imagen VARCHAR(255),

precio DECIMAL(10,2) NOT NULL,

id\_categoria INT NOT NULL,

FOREIGN KEY (id\_categoria) REFERENCES categorias(id)

);

-- TABLA pedidos

CREATE TABLE pedidos (

id INT PRIMARY KEY AUTO\_INCREMENT,

id\_usuario INT NOT NULL,

precio\_total DECIMAL(10,2) NOT NULL,

creado\_en TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (id\_usuario) REFERENCES usuarios(id)

);

-- TABLA elementos\_pedido

CREATE TABLE elementos\_pedido (

id INT PRIMARY KEY AUTO\_INCREMENT,

id\_pedido INT NOT NULL,

id\_producto INT NOT NULL,

cantidad INT NOT NULL,

FOREIGN KEY (id\_pedido) REFERENCES pedidos(id),

FOREIGN KEY (id\_producto) REFERENCES productos(id)

);

-- TABLA carritos

CREATE TABLE carritos (

id INT PRIMARY KEY AUTO\_INCREMENT,

id\_usuario INT NOT NULL,

precio\_total DECIMAL(10,2) NOT NULL,

creado\_en TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (id\_usuario) REFERENCES usuarios(id)

);

-- TABLA elementos\_carrito

CREATE TABLE elementos\_carrito (

id INT PRIMARY KEY AUTO\_INCREMENT,

id\_carrito INT NOT NULL,

id\_producto INT NOT NULL,

cantidad INT NOT NULL,

FOREIGN KEY (id\_carrito) REFERENCES carritos(id),

FOREIGN KEY (id\_producto) REFERENCES productos(id)

);

-- TABLA opiniones

CREATE TABLE opiniones (

id INT PRIMARY KEY AUTO\_INCREMENT,

id\_usuario INT NOT NULL,

id\_producto INT NOT NULL,

calificación INT NOT NULL,

comentario TEXT,

creado\_en TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (id\_usuario) REFERENCES usuarios(id),

FOREIGN KEY (id\_producto) REFERENCES productos(id)

);

-- TABLA listas\_de\_deseos

CREATE TABLE listas\_de\_deseos (

id INT PRIMARY KEY AUTO\_INCREMENT,

id\_usuario INT NOT NULL,

id\_producto INT NOT NULL,

creado\_en TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (id\_usuario) REFERENCES usuarios(id),

FOREIGN KEY (id\_producto) REFERENCES productos(id)

);

-- TABLA pagos

CREATE TABLE pagos (

id INT PRIMARY KEY AUTO\_INCREMENT,

id\_pedido INT NOT NULL,

cantidad DECIMAL(10,2) NOT NULL,

metodo\_de\_pago VARCHAR(50) NOT NULL,

pagado\_en TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (id\_pedido) REFERENCES pedidos(id)

);

**3. Creación del proyecto “bakery\_store”**

Por el momento se creo el proyecto, se utilizó el siguiente comando:

npx create-react-app bakery\_store

**UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA**

**UNAN – LEÓN**

****

**FACULTAD DE CIENCIAS Y TECNOLOGÍA**

**Carrera:** Ingeniería en Sistemas de Información

**Grupo:** GP1

**Componente:** Programación Orientada a la Web II

**Tema:** Creación de componentes

**Elaborado por:** Br. Cristian José Oporta Cajina

**Asignado por:** MSc. Juan Carlos Leyton Briones

**Fecha de inicio:** 03/10/2023

**“A La Libertad Por La Universidad”**

**03/10/2023**

**Introducción:**

React es una biblioteca de JavaScript open-source creada por Facebook para construir interfaces de usuario. Utiliza un enfoque basado en componentes que permite dividir la interfaz en partes independientes, reutilizables y pensadas en función de su estado. Los componentes de React son la principal forma de encapsular código reutilizable.

**Desarrollo:**

Los componentes en React son como funciones JavaScript que aceptan entradas arbitrarias (llamadas "props") y devuelven a React los elementos que describen lo que debería aparecer en la pantalla. Los componentes permiten separar la interfaz en piezas independientes, reutilizables y pensar en cada pieza de forma aislada.

Hay dos tipos de componentes en React:

* **Componentes de función:** Son funciones JavaScript simples que aceptan props como entrada y devuelven los elementos de React que describen lo que se debe renderizar. No tienen estado propio.
* **Componentes de clase:** Son clases de JavaScript que extienden React.Component y deben definir un método render() que devuelve los elementos a renderizar. Los componentes de clase mantienen su propio estado interno que se actualiza a través de métodos como setState().

Algunas características importantes de los componentes en React:

* Encapsulan la lógica y el markup de partes de la interfaz.
* Pueden ser reutilizados y combinados para crear UIs complejas.
* Permiten separar las preocupaciones en partes más pequeñas e independientes.
* Se actualizan de forma eficiente gracias al sistema de renderizado declarativo de React.
* Permiten pensar en el estado de cada componente de forma aislada.

**Conclusión:**

Los componentes son la piedra angular de React. Permiten dividir la interfaz en piezas independientes, reutilizables y pensadas en función de su estado local. Existen componentes de función sin estado y componentes de clase con estado. La composición y reutilización de componentes es clave para construir interfaces de usuario de forma modular, escalable y eficiente en React. El uso de componentes bien diseñados facilita enormemente el desarrollo y mantenimiento de aplicaciones complejas.

**Referencias:**

Facebook Inc. (2023). Components and Props. React documentation. <https://reactjs.org/docs/components-and-props.html>

Refsnes Data. (2023). React Components. W3Schools. <https://www.w3schools.com/react/react_components.asp>

Gackenheimer, C. (2015). Introduction to React. O'Reilly Media, Inc.

**Seguimiento de la aplicación:**

Para este punto he creado una carpeta “components” ubicada en la carpeta “src” para alojar todos los componentes que voy a utilizar en el archivo principal de la SPA “App.js”.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

También he creado por el momento un componente “NavBar.js” para la barra de navegación y un “Footer.js” para la información de la compañía en el pie de la página.

Una captura de pantalla de un celular

Descripción generada automáticamente con confianza media

Por el momento la aplicación luce de este estilo:

