Logotipo, nombre de la empresa

Descripción generada automáticamente  
 Instituto Tecnológico de Costa Rica

Unidad de Computación

Base de datos distribuida: Monedero electrónico y servicio de cobros

Justin Alonso Martínez Molina 2019027054

Leiner Alvarado Rodríguez - 2022437759

Walter Lazo González - 2021081316

Sede San Carlos

12/05/2024

**Introducción.**

El proyecto que presentamos aborda la creación y configuración de una base de datos distribuida para un sistema de cobro basado en una plataforma de monedero electrónico. En un contexto donde la descentralización y la escalabilidad son fundamentales en los servicios de pago, esta iniciativa busca implementar un sistema que permita gestionar eficientemente las transacciones financieras en tiempo real.

La necesidad de registrar productos, gestionar ventas, aplicar descuentos y mantener el saldo de monederos electrónicos asociados a los clientes es cada vez más evidente en el panorama actual de las tecnologías financieras. Por ello, nuestro proyecto se centra en diseñar modelos de bases de datos tanto para el nodo central, que contendrá información sobre clientes, históricos de ventas y monederos electrónicos, como para los nodos locales, que gestionarán información sobre productos, inventario y ventas locales. Entre las funciones clave que abordamos se encuentra el registro de clientes, la aplicación de promociones, la recarga y descarga de monederos electrónicos, así como la validación de usuarios. Estas funcionalidades son esenciales para garantizar la operatividad y seguridad del sistema.

**Análisis del problema**

El problema principal que enfrentamos en este proyecto es la necesidad de crear un sistema de cobro basado en una plataforma de monedero electrónico que sea distribuido y escalable. Para lograr esto, debemos diseñar una arquitectura de base de datos que permita gestionar eficientemente la información relacionada con clientes, productos, ventas y transacciones financieras en tiempo real. Uno de los desafíos clave es garantizar la integridad y seguridad de los datos en un entorno distribuido, donde múltiples nodos locales deben comunicarse con un nodo central para realizar operaciones como el registro de clientes, la aplicación de promociones, la recarga y descarga de monederos electrónicos, entre otros. Además, debemos asegurarnos de que la plataforma sea fácil de usar tanto para los clientes como para los cajeros que operan en los puntos de venta. Esto implica desarrollar una interfaz web para los cajeros y una aplicación web (PWA) para los usuarios del monedero electrónico, que les permita acceder fácilmente a sus cuentas y realizar transacciones de manera segura.

**Solución del problema**

Se diseña la base central en PostgreSQL implementando la logística para llevar a cabo el monedero electrónico.

Para los nodos locales se utiliza una instancia en PostgreSQL y otra en SQL Server, que guardará la información de los productos y las ventas locales.

Se define las siguientes funcionalidades:

– Registro de Clientes: se implementa un sistema de registro en el servidor central en el que se solicita la información personal.

Recarga Monedero Electrónico: Permite recargar el monedero desde cualquier nodo solicitando únicamente la cédula, sin guardar el medio de pago.

Descarga Monedero Electrónico: Se permite aplicar una rebaja al monedero al fondo disponible.

Registro de Productos: Cada nodo puede agregar y vender sus productos disponibles.

Validación de compras: Se implementan códigos QR para validar la identidad del usuario y acceder a los fondos del monedero en cada transacción.

**Interfaz**

**Cajero**: El uso de Astro y React para la construcción del frontend de la página web ofrece numerosas ventajas. Astro es un framework estático y dinámico que permite generar páginas estáticas para un rendimiento óptimo, mientras que también ofrece la capacidad de agregar interactividad mediante React. Esta combinación ofrece una experiencia de usuario fluida y rápida, ya que las páginas se cargan de manera eficiente y luego se vuelven interactivas sin sacrificar la velocidad.

Integrar la biblioteca de componentes NextUI en el proyecto agrega una capa adicional de profesionalismo y cohesión visual. Los componentes predefinidos de NextUI permiten crear interfaces de usuario consistentes y visualmente atractivas con un mínimo esfuerzo de desarrollo. Además, la flexibilidad de personalización de estos componentes garantiza que el diseño de la página web pueda adaptarse fácilmente a las necesidades específicas del proyecto.

En el backend, la elección de Python junto con Flask proporciona un entorno de desarrollo ágil y eficiente. Flask es conocido por su simplicidad y su enfoque minimalista, lo que facilita la creación rápida de APIs y la gestión de solicitudes HTTP. Además, la amplia disponibilidad de extensiones y librerías para Flask simplifica tareas comunes como el manejo de autenticación, la conexión con bases de datos y la implementación de lógica de negocio.

Al combinar estas tecnologías tanto en el frontend como en el backend, se logra un ecosistema de desarrollo coherente y eficaz. Esto permite a los desarrolladores trabajar de manera más eficiente y concentrarse en crear una experiencia de usuario excepcional, mientras que también se asegura un rendimiento óptimo y una gestión eficiente de los recursos del servidor.

Usuarios: Se crea una PWA (progressive web app) para la solicitud de códigos QR que son necesarios para cada compra.

**Análisis de resultados**

|  |  |  |
| --- | --- | --- |
| Aspecto. | Resultado | Comentarios |
| Diseño y creación de base de datos | 100% |  |
| Plataforma: Servidor Central | 100% |  |
| Plataforma: Nodos Locales | 80% | El nodo realizado en SQL SERVER no se conecta en la aplicación web y hay un tipo de dato incompatible en insertar ventas |
| Desarrollo Web y PWA | 80% | La PWA no se conecta a las bases de datos |
| Documentación Interna | 100% |  |
| Documentación Externa | 100% |  |

***Conclusiones***

A lo largo de la gestión de las bases de datos se ofrece una solución integral y robusta para la creación y gestión de un sistema de cobro basado en una plataforma de monedero electrónico distribuida. Al abordar la necesidad de descentralización y escalabilidad en los servicios de pago, hemos diseñado una arquitectura de base de datos que permite gestionar eficientemente la información relacionada con clientes, productos, ventas y transacciones financieras.

La creación de una interfaz de usuario intuitiva y eficiente, tanto para los cajeros como para los usuarios de la PWA, es crucial para la adopción y el éxito del sistema. La elección de tecnologías como Astro, React y Flask permite ofrecer una experiencia de usuario fluida y rápida, garantizando que las transacciones se realicen de manera eficiente y segura.

Finalmente, se comprende que garantizar la integridad y seguridad de los datos en un entorno distribuido es fundamental para la confianza de los usuarios y la protección de la información sensible.

**Recomendaciones**

A medida que el sistema crece, es fundamental garantizar su capacidad para escalar horizontalmente y manejar un aumento en la carga de trabajo sin degradación del rendimiento.

El sistema debe estar sujeto a un proceso continuo de mejora basado en el feedback de los usuarios y las necesidades cambiantes del mercado. Esto podría implicar la incorporación de nuevas funcionalidades, optimizaciones de rendimiento, o ajustes en las interfaces para mejorar la experiencia.

Mantener la documentación detallada del sistema, permitirá realizar cambios y ofrecer un mantenimiento adecuado tanto de la base central como de los nodos.

**Bibliografía**

*PostgreSQL 16.3 Documentation*. (2024, May 9). PostgreSQL Documentation. <https://www.postgresql.org/docs/16/index.html>

Diagrama

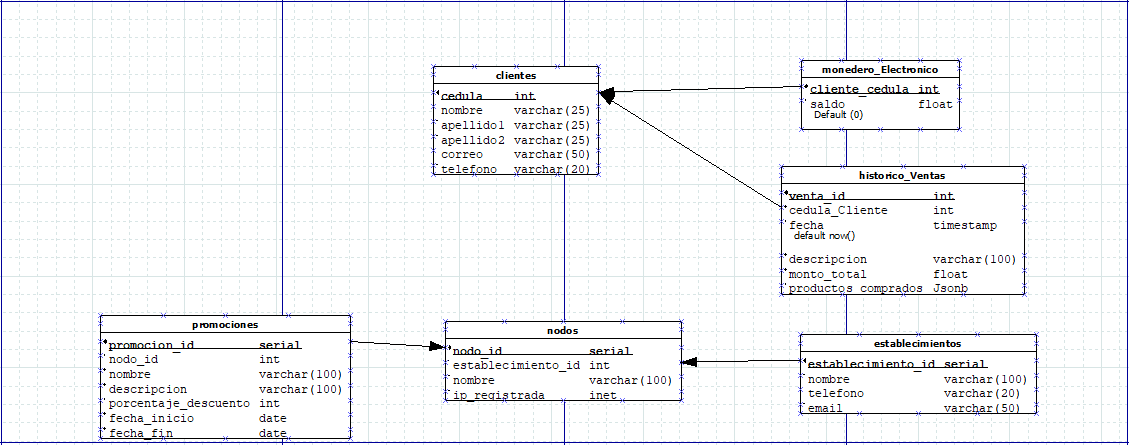


Figura 1 - Base Central - PostgreSQL

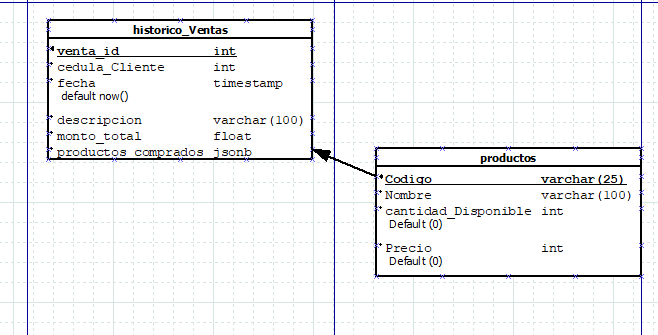


Figura 2 - Nodo Local