Descripción de la Solución Implementada y Decisiones Técnicas

Introducción

Este documento describe la solución implementada para la simulación de un sistema de gestión de almacenes (WMS) utilizando Vue 3. El proyecto está diseñado para manejar productos en un almacén, permitiendo agregar nuevos productos, actualizar cantidades existentes, y visualizar un resumen del inventario. Toda la información se gestiona en memoria y no persiste después de recargar la página.

Estructura del Proyecto

El proyecto se organiza en tres componentes principales que se integran en el archivo `App.vue`. Estos componentes son:

- `ProductForm.vue`: Permite agregar nuevos productos al sistema.
- `UpdateProduct.vue`: Facilita la actualización de las cantidades de los productos existentes.
- `WarehouseSummary.vue`: Muestra un resumen del inventario actual, incluyendo el número total de productos y la cantidad total almacenada.

Descripción de los Componentes

ProductForm.vue

Este componente se encarga de agregar nuevos productos al sistema. El formulario incluye campos para ingresar el nombre del producto, el código SKU y la cantidad disponible. Una vez que se ingresa la información, el nuevo producto se añade a la lista de productos existentes mediante un evento emitido al componente padre.

UpdateProduct.vue

El componente `UpdateProduct.vue` permite actualizar la cantidad de un producto existente en el inventario. El usuario selecciona un producto de la lista desplegable y especifica la nueva cantidad que se sumará a la cantidad existente. Esta actualización se realiza mediante la emisión de un evento al componente padre, que se encarga de gestionar la lógica de la suma.

WarehouseSummary.vue

El componente `WarehouseSummary.vue` proporciona un resumen del inventario del almacén. Muestra el número total de productos y la cantidad total almacenada, utilizando los productos que se pasan como `props` desde el componente padre.

Decisiones Técnicas

1. **Manejo del Estado en `App.vue`:**
El estado de los productos se gestiona en el componente principal `App.vue`. Este enfoque centralizado asegura que todos los componentes reciban datos actualizados y puedan interactuar correctamente. El estado se pasa como `props` a los componentes hijos, y estos emiten eventos cuando es necesario realizar actualizaciones, como agregar un nuevo producto o actualizar una cantidad existente.

2. **Reactividad y Gestión de Cantidades:**
El uso de la reactividad de Vue permite que cualquier cambio en los datos del inventario se refleje automáticamente en la interfaz de usuario. Una decisión clave fue implementar la actualización de cantidades sumando la cantidad ingresada a la cantidad existente en lugar de reemplazarla. Esto asegura que las operaciones de inventario sean precisas y reflejen correctamente las acciones del usuario.

3. **Simplicidad en la UI:**
La interfaz de usuario se diseñó para ser simple y funcional, cumpliendo con los requisitos de claridad y facilidad de uso. No se implementaron características avanzadas de diseño, lo que permitió centrarse en la funcionalidad principal y en la correcta gestión del inventario.

Conclusión

La solución implementada cumple con los requerimientos funcionales del proyecto, proporcionando una forma eficiente de gestionar productos en un almacén sin necesidad de persistencia en base de datos. Las decisiones técnicas tomadas, como la centralización del estado y la reactividad de Vue, aseguran que la aplicación sea robusta y fácil de mantener. Este proyecto puede servir como base para futuros desarrollos que requieran una gestión más avanzada de inventarios.