

## Documentación del módulo de gestión de materiales

Esta documentación cubre los archivos principales del módulo de gestión de materiales, que se encarga de listar, agregar y administrar materias primas en un sistema de inventario. Las siguientes secciones incluyen un diagrama de flujo de navegación y datos, mapeo de endpoints, descripciones detalladas de los archivos y análisis lógico para detectar posibles fallas.

### Diagrama de navegación y flujo de datos

A continuación se presenta un diagrama de flujo que resume la navegación entre las pantallas/componentes principales y cómo fluye la información por la aplicación:

- **Lista de Materias Primas** es la pantalla principal de listado, filtro y alta.
- **Agregar Materia Prima** es el formulario para registrar nuevas materias primas.
- **Detalle de Materiales** muestra el detalle completo, incluyendo costos, almacenamiento, política de stock y datos contables.
- **API** representa los endpoints de backend usados para operaciones CRUD.

### Uso de endpoints por componente

A continuación se muestra un mapeo de los endpoints (llamadas a API) que utiliza cada componente, según el análisis del código:

Componente / Archivo	Endpoints de API usados	Descripción
MateriasPrimas.tsx	GET /materiasprimas	Trae todas las materias primas y datos relacionados
MateriasPrimasAgregar.tsx	POST /materiasprimas	Da de alta una nueva materia prima
DetalleMP.tsx	GET /materiasprimas/:id PATCH /materiasprimas/:id	Trae y actualiza una materia prima específica
DetalleMPAlmacenamiento.tsx	(ninguno directo)	Recibe datos por props
DetalleMPCostos.tsx	(ninguno directo)	Recibe datos por props
DetalleMPPoliticaDeStock.tsx	(ninguno directo)	Recibe datos por props

DetalleMPDatosContables.tsx	(ninguno directo)	Recibe datos por props
MateriasPrimasTabla.tsx	(ninguno directo)	Recibe datos por props

#### Claves:

- Todas las llamadas a API se realizan mediante `hacerPeticionApi`, con el endpoint implícito según método y contexto.
- Más abajo se detalla cada endpoint mediante bloques `api`.

## Descripción de archivos y análisis

### MateriasPrimas.tsx

#### Listar y gestionar materias primas

Este es el componente principal de la gestión de materias primas. Proporciona una tabla filtrable, acciones para alta de nuevos artículos y opciones para exportar o imprimir.

- **Gestión de estado:** Maneja listas de materias primas, filtros y estado de UI para el modo de alta.
- **Uso de API:** Al montar, realiza `hacerPeticionApi` para obtener materias primas, depósitos, tipos y unidades.
- **Interacción de usuario:**
  - Filtrar por descripción, código o ID.
  - Alta de nueva materia prima (muestra `MateriasPrimasAgregar`).
  - Exportar (Excel) e imprimir (Excel/PDF).
  - Paginación (mediante `ControlPaginas`).

Figura 1 – Pantalla principal de Gestión de Materias Primas con listado, filtros y acciones de alta, exportación e impresión.

## GET /materiasprimas

### Análisis de lógica

- **Uso de useMemo para filtrado:** Filtra eficientemente la lista cuando cambian el filtro o los datos.
- **Posible falla:** Se llama a `setCalidadesDeArticulos` pero no está definido en el fragmento, lo que puede provocar error.
- **Acceso de usuario:** Los botones de alta solo se muestran si `usuario.nivel > 1`.

## MateriasPrimasAgregar.tsx

### Alta de una nueva materia prima

Este componente es el formulario para crear materias primas nuevas. Maneja entrada de datos, validación y envío al backend.

- **Estado de formulario:** Usa estado local para todos los campos (descripción, dimensiones, stock, etc.).
- **Validación:** Utiliza `useMemo` para calcular si el formulario es válido antes de habilitar el envío.
- **Envío:** Al aceptar, llama a `hacerPeticionApi` con método POST y actualiza el estado del padre al recibir respuesta.
- **UX:** Ajusta unidades, depósito y campos mostrados dinámicamente según el tipo seleccionado.
- **Campos solo lectura:** Algunos campos (por ejemplo, código) se calculan automáticamente y no se pueden editar.

Figura 2 – Formulario de alta de una nueva Materia Prima con carga de datos generales, unidades y parámetros de stock.

## POST /materiasprimas

### Análisis de lógica

- **Validación:** Es completa y cubre los campos relevantes.
- **Autoconfiguración de campos:** Ajusta unidades y depósitos automáticamente para ciertos tipos.
- **Posibles problemas:**
- `setCalidadesDeArticulos` está comentado en el padre; si se usa, debe existir.
- `useTeclaEnter` maneja la tecla Enter; si no está bien implementado, podría generar envíos dobles.

## MateriasPrimasTabla.tsx

### Tabla de materias primas

Este componente muestra las materias primas en una tabla sin estado propio. Muestra columnas de dimensiones, descripción, proveedor y unidad de medida.

- **Paginación:** Usa la utilidad `paginado` para dividir los resultados por página.
- **Navegación a detalle:** Al hacer clic en una fila navega al detalle si el usuario tiene acceso.
- **Indicadores visuales:** Muestra advertencia para ítems dados de baja o no enviados al ERP.
- **Sin datos:** Usa `NoHayElementos` cuando la lista está vacía.

### Análisis de lógica

- **Chequeo de acceso:** Usa `useAcceder` para restringir la navegación según permisos.

- **Posibles fallas:** No se detectan problemas lógicos significativos; la tabla es sencilla.

## DetalleMP.tsx

### Vista de detalle de materia prima

Muestra todos los datos de una materia prima seleccionada. Se compone de varios paneles de detalle.

- **Cabecera:** Botones de navegación (volver, movimientos de stock).
- **Datos generales:** Código, tipo, descripción, factor de conversión, unidades, proveedor y parámetros.
- **Paneles:**
  - Política de Stock (PoliticaDeStock)
  - Datos Contables (DatosContables)
  - Costos (Costos)
  - Almacenamiento (Almacenamiento)
- **Acciones:** Botones de Aceptar/Cancelar y Dar de baja/Reactivar materia prima.

Figura 3 – Vista de detalle de una Materia Prima con información general, política de stock, costos y datos contables.

### GET /materiasprimas/:id

### PATCH /materiasprimas/:id

### Análisis de lógica

- **Carga de datos:** Carga todos los datos relacionados al montar el componente.
- **Manejo de estado:** Muestra un spinner de carga (Cargando) mientras se reciben los datos.
- **Posibles fallas:**

- `editarMateriaPrima` llama `hacerPeticionApi` sin indicar el ID en el propio componente. Si `hacerPeticionApi` no infiere el ID por ruta o contexto, podría afectar al recurso incorrecto.
- El botón `AceptarCancelarBtn` está deshabilitado (`disabled={true}`), por lo que no se pueden guardar cambios; puede ser intencional pero confuso.

## DetalleMPAlmacenamiento.tsx

### Panel de almacenamiento

Muestra (actualmente en forma muy limitada) información de almacenamiento y embalaje de la materia prima.

- **Campos:** Peso, tipo de embalaje y unidad de medida (muchos comentados para futura implementación).
- **Props:** Recibe tipos de embalaje, la materia prima y unidades de medida.

### Análisis de lógica

- **Solo lectura:** No hace llamadas a API, solo muestra valores que llegan por props.
- **Posibles fallas:**
  - La mayoría de los campos y la lógica están comentados o como placeholders.
  - Los inputs de peso y unidad tienen `value={ ' ' }`, por lo que hoy no muestran datos reales.

## DetalleMPCostos.tsx

### Panel de costos

Muestra información de costos para la materia prima.

- **Campos:** Cuenta contable, costo estándar, costo real, última compra y costo promedio.
- **Props:** Recibe un objeto de costos (`costosType`).
- **Formato:** Usa `formatoMoneda` para mostrar valores de manera legible.

### Análisis de lógica

- **Interactividad:** El usuario solo puede cambiar el rango de cálculo del promedio.
- **Posibles fallas:**
  - Todos los campos son de solo lectura y el botón de historial está deshabilitado.
  - No existe edición ni refresco de datos; el panel se marca como "en desarrollo".

## DetalleMPPoliticaDeStock.tsx

### Panel de política de stock

Muestra parámetros de control de stock asociados a la materia prima.

- **Campos:** Control por lotes, stock mínimo, stock crítico, lote óptimo de compra y depósito.
- **Props:** Recibe la política de stock y la información de depósitos.

### Análisis de lógica

- **Solo lectura:** Todos los inputs son de solo lectura; se usa `materiaPrima` y `politicaDeStock` como fuente.
- **Posibles fallas:**
- El checkbox de Control por Lotes tiene `onChange={() => {}}`, por lo que no realiza acciones.
- No hay forma de editar estos valores desde este componente.

## DetalleMPDatosContables.tsx

### Panel de datos contables

Muestra información contable relacionada con la materia prima.

- **Campos:** Cuenta de inventario, ventas y compras (selects con única opción, valor actual).
- **Props:** Recibe un objeto `contables` con estas cuentas.

### Análisis de lógica

- **Solo lectura:** Todos los select están forzados a `value={1}` y solo tienen una opción.
- **Posibles fallas:**
- El campo de IVA está comentado y no se muestra.
- La lógica actual no permite modificar ninguna cuenta desde este panel.

## Fallas lógicas y de diseño

Se identifican varias posibles fallas y áreas de mejora dentro del módulo:

- **Vistas incompletas o solo lectura:** Muchos paneles de detalle (almacenamiento, costos, política de stock, contables) son solo lectura o están marcados "en desarrollo". El usuario no puede editar estas secciones.
- **Acciones deshabilitadas:** En `DetalleMP.tsx`, el botón de Aceptar/Cancelar está permanentemente deshabilitado, impidiendo actualizaciones. Puede ser intencional para solo consulta, pero genera ambigüedad.
- **Contexto de llamadas API:** Al actualizar o dar de baja una materia prima, el código no indica explícitamente el ID en las llamadas, confiando en que `hacerPeticiónApi` lo maneje. Esto es arriesgado si la función no tiene ese contexto.
- **Variables sin definir o sin usar:**

- En `MateriasPrimas.tsx` se usa `setCalidadesDeArticulos` pero no se ha declarado en el fragmento mostrado.
- **Funcionalidades sin completar:** Varios textos "en desarrollo" y código comentado indican que faltan partes por implementar o refactorizar.

### Áreas a mejorar

Varios paneles son solo lectura o están incompletos. Asegure IDs claros en las APIs y habilite edición donde sea necesario.

## Resumen general

Componentes	Uso de API	¿Editable?	Posibles problemas
<code>MateriasPrimas.tsx</code>	GET /materiasprimas	No	<code>setCalidadesDeArticulos</code> indefinido
<code>MateriasPrimasAgregar.tsx</code>	POST /materiasprimas	Sí	Ninguno evidente
<code>MateriasPrimasTabla.tsx</code>	Ninguno	No	Ninguno evidente
<code>DetalleMP.tsx</code>	GET/PATCH /materiasprimas/:id	Parcial	Botón deshabilitado, contexto PATCH
<code>DetalleMPAlmacenamiento.tsx</code>	Ninguno	No	Campos sin datos reales
<code>DetalleMPCostos.tsx</code>	Ninguno	No	Solo lectura, sin historial activo
<code>DetalleMPPoliticaDeStock.tsx</code>	Ninguno	No	Solo lectura, sin handlers
<code>DetalleMPDatosCon tables.tsx</code>	Ninguno	No	Solo lectura, IVA comentado

## Recomendaciones

- Completar los subpaneles que aún están "en desarrollo" para obtener una vista de detalle totalmente funcional.
- Eliminar código comentado o sin uso, o bien terminar su implementación.
- Asegurar que todas las llamadas de actualización/baja a la API indiquen explícitamente el ID de la materia prima.
- Revisar y corregir setters o estados no definidos en los componentes.

- Mejorar la experiencia de usuario dejando claro qué campos son solo lectura y cuáles se pueden editar.