

Diagrama de flujo de navegación y datos

Este diagrama muestra la navegación y el flujo de datos entre los tres componentes y la capa API.

Endpoints por componente

A continuación se indican las llamadas clave a la API que realiza cada componente.

Componente	Método	Endpoint	Descripción
NormasFabricación	GET	/normasPf	Obtiene la lista completa de normas.
NormasFabricación Abm	POST	/normasPf	Crea una nueva norma de fabricación.
NormasFabricación Abm	PUT	/normasPf	Actualiza datos de una norma existente.
NormasFabricación Abm	PATCH	/normasPf	Anula o reactiva una norma.
NormasFabricaciónTabla	—	—	No realiza peticiones directas.

NormasFabricacionTabla.tsx

Este componente presenta la **tabla paginada** de normas de fabricación. Recibe datos filtrados y controla la visualización del botón de edición según permiso de usuario.

Props

ts

```
type propsType = {
  normasPf: normaPfType[] | null;
  pagina: number;
  setMostrarModal: Dispatch<SetStateAction<boolean>>;
  setNormaEditar: Dispatch<SetStateAction<normaPfType | null>>;
  tablaRef: RefObject<HTMLTableElement>;
}
```

Prop	Tipo	Descripción	
normasPf	`normaPfType[]` \	null`	Lista de normas para mostrar (puede

			estar vacía).
pagina	number	Página actual del paginado.	
setMostrarModal	Dispatch<SetStateAction<boolean>>	Controla apertura/cierre del modal ABM.	
setNormaEditar	`Dispatch<SetStateAction<normaPfType \	null>>`	Define la norma a editar en el modal.
tablaRef	RefObject<HTMLTableElement>	Referencia a la tabla para impresión/descarga.	

Lógica principal

- Aplica estilos dinámicos al encabezado con `generarEstilosDeCabezaDeTabla`.
- Paginación mediante `paginado(normasPf, pagina)`.
- Muestra mensaje o componentes cuando no hay elementos.
- Habilita botón de editar sólo si `tienePermiso(usuario, norma)`.

Integración de permisos y acciones

- Obtiene nivel de usuario con `useTraerUsuario`.
- Muestra columna de **Editar** sólo si `usuario.nivel > 1`.
- Al pulsar **Editar**, abre modal y asigna norma a `normaEditar`.

NormasFabricacion.tsx

Este componente agrupa la lógica de carga, filtrado y control de estados. Orquesta la presentación de la tabla y el formulario ABM.

Figura 1 – Pantalla principal de Normas de Fabricación con listado, filtros y acciones de alta, exportación e impresión.

Estados y referencias

- `tablaRef`: referencia para impresión/descarga.
- `normaEditar`: norma seleccionada para editar.
- `normasPf`: datos crudos desde la API.
- `pagina`: índice de paginado.

Hooks personalizados

- `useTraerConfiguracion`: controla indicador de carga global.
- `useDescargarExcel`: genera exportación de Excel.
- `useFiltrar`: maneja filtro de descripción.
- `useImprimirTabla`: prepara impresión Excel/PDF.
- `useMostrarError`: despliega errores de petición.
- `useTraerUsuario`: extrae datos de usuario.

Carga de datos

- En `useEffect`, llama `hacerPeticionApi()` (GET).
- Al recibir datos, asigna `normasPf` o muestra error.

Filtrado y paginación

- Usa `useMemo` para filtrar por descripción.
- Resetea pagina a 1 al cambiar filtro.
- Control de páginas con `ControlPaginas`.

Acciones y botones

- **Alta de Norma**: abre modal ABM si `usuario.nivel > 1`.
- **Exportar todo**: Excel completo.

- **Imprimir Tabla:** Excel o PDF de la vista actual.
- Filtro de descripción con <input>.

NormasFabricacionAbm.tsx

Formulario modal para **alta**, **edición** o **anulación** de normas de fabricación. Gestiona estados de campos y envíos a la API.

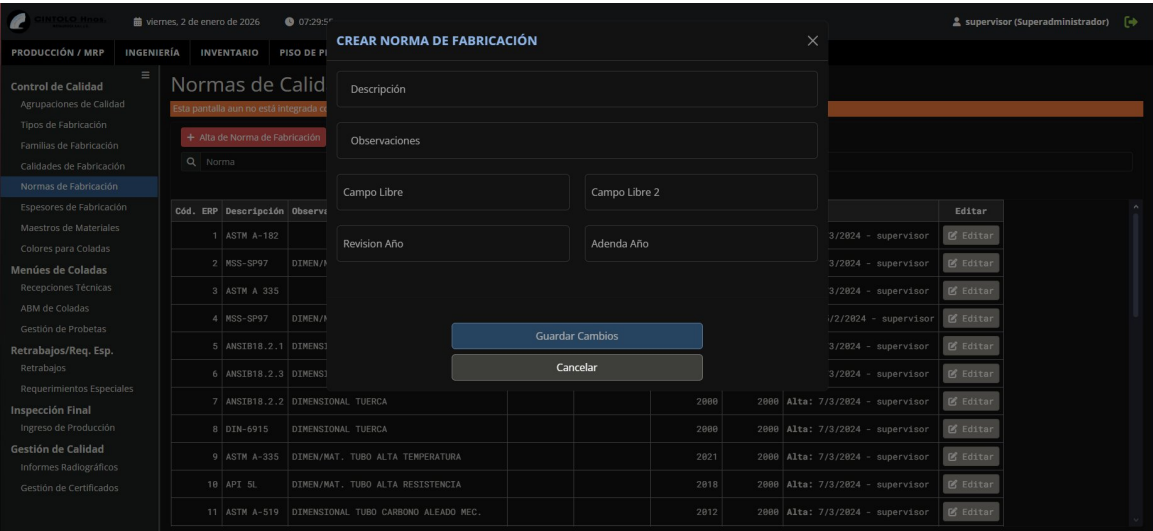


Figura 2 – Formulario de alta de una nueva Norma de fabricación con carga de descripción, observaciones, campos libres, revisión año y adenda año.

Props

ts

```
type propsType = {
  cerrarModal: () => void;
  normaEditar: normaPfType \ | null;
  setNormasPf: Dispatch<SetStateAction<normaPfType[] \ | null>>;
}
```

Prop	Tipo	Descripción	
	cerrarModal	() => void	Función para cerrar el modal.
normaEditar	`normaPfType \	null`	Norma a editar; null para un alta.
setNormasPf	`Dispatch<SetStateAction<normaPfType[] \	null>>`	Actualiza lista de normas en el padre.

Formularios y estados locales

- Campos: descripcion, observaciones, campoLibre, campoLibre2, revisionAño, adendaAño.

- Control de longitud máxima en inputs (<101 caracteres).
- Estado guardando para deshabilitar botones.

Validación y envío

- esValido: activa botón **Aceptar** sólo si descripcion no está vacío.
- Función aceptar(anular = false):
- Elige método: POST, PUT o PATCH.
- Envía body con datos según operación.
- Al recibir respuesta, actualiza setNormasPf y cierra modal.
- Muestra error y cierra modal si la API falla.

Botones de acción

- Componente AceptarCancelarBtn gestiona:
- Confirmar (alta/editar).
- Anular o reactivar.
- Cancelar.
- Indicador de carga.

Fallas en la lógica

Estas son las inconsistencias detectadas en la implementación:

Estado modal indefinido

En NormasFabricacion.tsx se comentó la definición de mostrarModal, provocando referencia a variable no declarada.

- Uso de setPagina dentro de useMemo, lo que genera efectos secundarios fuera de useEffect.
- titulosDeTabla en NormasFabricacionTabla no coincide con los <th> reales.
- paginado(normasPf, pagina) asume que normasPf es siempre array y no maneja null.