

## Diagrama de flujo de navegación y datos

[Ver diagrama completo de navegación y flujo de datos](#)

Este diagrama muestra la **navegación** entre vistas y el **flujo de datos** desde los endpoints hacia el estado de la aplicación.

## Endpoints utilizados por componente

Componente	Endpoints
Legajos.tsx	GET /legajos
LegajosAbm.tsx	POST /legajos
	PUT /legajos
	PATCH /legajos
Matriceria.tsx	GET /matriceria (devuelve matrices, tiposDeMaquinas y centrosDeCosto)
MatriceriaAgregar.tsx	POST /matriceria
MatriceriaEditar.tsx	PUT /matriceria
LegajosTabla.tsx, MatriceriaTabla.tsx	(no invocan endpoints directos; consumen datos en props)

## Fallas en la lógica


- En **Legajos.tsx** la constante `legajoEditar` y su setter están comentados. Llamar a `setLegajoEditar` sin definir provoca error de compilación.
- Dentro de `useMemo` en **Legajos.tsx** se invoca `setPagina(1)`. Cambiar estado en un hook memoizado puede generar bucles de render.
- En **Matriceria.tsx** está comentada la declaración de `centrosDeCosto` y su setter. Llamar a `setCentrosDeCosto` sin definir provoca error.
- Al inicializar `useImprimirTabla(tablaRef?.current)`, se pasa `null`. Si el hook no actualiza la referencia, las acciones de impresión fallarán.
- En **LegajosTabla.tsx**, la columna "Usuario del Sistema" muestra un texto estático. Falta integrar datos reales.
- No se capturan errores de red con bloques `catch`, lo que impide manejar fallos de conexión.

## Legajos.tsx

Este **componente principal** orquesta la vista de Legajos Empleados.

Carga la lista desde el API y gestiona la búsqueda, exportación e impresión.

- Hooks y estados principales:
- **config**: controla el estado de carga global.
- **legajos** (legajoType[] | null): lista original.
- **muestra** (legajoType[]): lista filtrada vía useFiltrar.
- **pagina** (number): página activa.
- **mostrarModal** (boolean): controla apertura del modal ABM.
- Operaciones disponibles:
- **Alta/Edición** abre LegajosAbm.
- **Exportar** invoca useDescargarExcel.
- **Imprimir** usa useImprimirTabla.
- **Buscar** ajusta filtro con useFiltrar.

GINTOLO Waves

📅

sábado, 3 de enero de 2026

🕒

05:25:15

supervisor (Superadministrador)

➡

PRODUCCIÓN / MRP

INGENIERÍA

INVENTARIO

PISO DE PLANTA

CALIDAD

MANTENIMIENTO

COSTOS

ADMINISTRACIÓN

Administración

Depósitos

Centros de Costo

Etapas / Subetapas

Legajos Empleados

Proveedores y Clientes

Recursos

Máquinas

Instrumentales

Matrickería

Configuración

Usuarios

Roles

Opciones

Logs

Reglas de Negocio

Programas Auxiliares

Legajos Empleados

+ Alta de Empleado

📄 Exportar todo

🖨 Imprimir Tabla

🖨 Imprimir Tabla

🔍

Buscar por id, nombre, o sector

⏪

<

Página 1 de 15

>

⏩

710 registros

Número	Apellido y Nombre	Sector	Elaboración	Usuario del Sistema	ABM
1	DIAZ JOSE ROMAN	33	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
2	GONZALEZ, ANDRES	18	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
3	AMICO, GUILLERMO RAMON ANTONIO	6	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
4	GARTIGALE, FELIPE	8	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
5	SANCHEZ, ELEDORO	31	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor Mod.: 19/5/2025 - supervisor
6	GAITAN, MIGUEL	11	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
7	CURI JOSE ALBERTO	10	<div>🔴</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
8	IBARRA, RAUL ANTONIO	10	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
9	CASTAGNARO OSCAR JOSE	16	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
10	SOLOHAGA ARTURO BENILDO	71	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
11	DIAZ, ERNESTO SERAFIN	31	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
12	GIMENEZ RICARDO ANTONIO	2	<div>🔴</div>	(Pendiente)	Alta: 14/6/2024 - supervisor
13	LEQUIZAMON EULOGIO	71	<div>🟡</div>	(Pendiente)	Alta: 14/6/2024 - supervisor

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

## Fragmento clave: carga de datos

js

```
useEffect(() => {
  hacerPeticionApi().then(respuesta => {
    config.setCargando(false);
    const ls = respuesta.datos?.legajos as legajoType[];
    if (!ls) {
      mostrarError(respuesta);
      return;
    }
    setLegajos(ls);
  });
}, []);
```

## LegajosTabla.tsx

Este componente presenta la **tabla paginada** de legajos.

Permite editar un registro si el usuario tiene permiso.

- Props:
- legajos: lista filtrada.
- pagina: página activa.
- editar: callback para abrir modal.
- tablaRef: ref para impresión.
- Funciones destacadas:
- **paginado**: segmenta la lista.
- **tienePermiso**: condiciona la edición.
- **generarEstilosDeCabezaDeTabla**: ajusta estilos.

js

```
<thead>
  <tr>
    <th>Número</th>
    <th>Apellido y Nombre</th>
    <th>Sector</th>
    <th>Elaboración</th>
    <th>Usuario del Sistema</th>
    <th>ABM</th>
  </tr>
</thead>
```

## LegajosAbm.tsx

Modal para **crear, editar o anular** un legajo.

Centraliza validaciones y notificaciones.

- Estados internos:
- apellidoNombre, sector, elaboracion: campos de formulario.
- guardando (boolean): estado de petición.
- Validación (esValido):
- **apellidoNombre** no vacío.
- **sector** mayor que cero.
- Botones:
- **Aceptar/Cancelar**
- **Anular/Reactivar** (según bajaUsuarioId)
- Notificación Redux tras guardar:

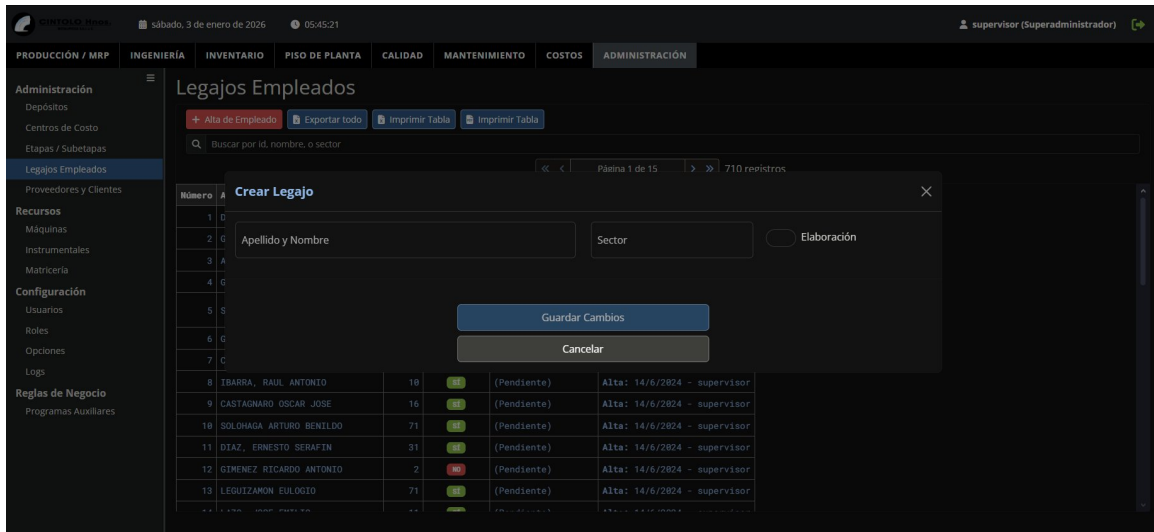


Figura 2 – Formulario de alta de un nuevo legajo con carga de apellido y nombre, sector y elaboración.

js

```
dispatch(abrirModalReducer({
  titulo: 'Logrado',
  mensaje: `Se ${anular ? 'anuló' : 'creó'} legajo de ${apellidoNombre}`,
  animacion: 1,
  ejecucionAceptar: cerrarModal
})));
```

## Matriceria.tsx

Vista principal de **Matriceria**.

Gestiona listado, búsqueda, exportación e impresión.

- Carga inicial:
- Invoca GET /matriceria para matrices, tiposDeMaquinas y centrosDeCosto.
- Almacena en estados: matrices, tiposMatrices, centrosDeCosto.
- Funciones de interfaz:
- **Nueva Matriz** abre MatriceriaAgregar.
- **Exportar/Imprimir** con hooks personalizados.
- **Buscar** filtra descripción.
- **Paginación** con ControlPaginas.

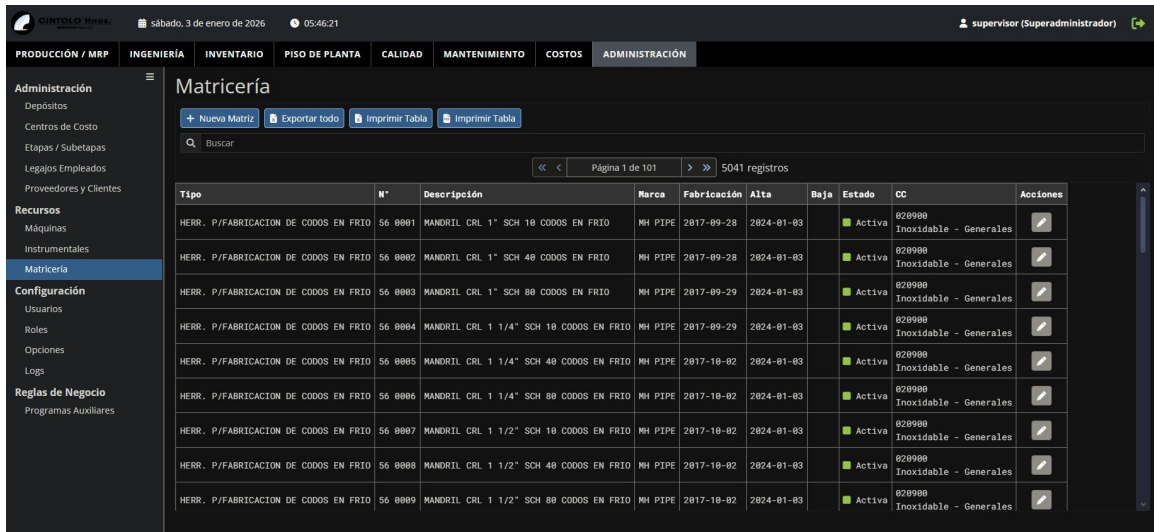


Figura 3 – Pantalla principal de Matricería con listado, filtros y acciones de Nueva Matriz, exportación e impresión.

js

```
const muestra = useMemo(() =>
  matrices?.filter(m =>
    m.descripcion.toLowerCase()
      .includes(filtrar.filtro.descripcion1.toLowerCase())
  ) || [],
  [filtrar.filtro, matrices]
);
```

## MatriceriaTabla.tsx

Tabla dinámica de matrices con opción de edición in-line.

Muestra estado, fechas y centro de costo.

- Props clave:
- muestra: datos filtrados.
- idMostrarEditar: controla fila de edición.
- tiposMatrices, centrosDeCosto: referencias para edición.
- Renderizado condicional:
- Fila normal vs. fila de edición (MatriceriaEditar).
- Botón de editar habilitado según permiso.

## MatriceriaEditar.tsx

Formulario in-line para editar una matriz existente.

Permite modificar descripcion, marca, fechaBaja y fechaFabricacion.

- Validaciones:
- Descripción  $\geq 7$  caracteres.

- Marca  $\geq 3$  caracteres.
- Fecha de fabricación obligatoria.
- No duplicar combinación tipoId + numero.
- Al guardar:
- Llama PUT /matrickeria con campos actualizados.
- Actualiza estado global setMatrices.

js

```
const respuesta = await hacerPeticionApi({
  method: 'PUT',
  body: { /* ... */ }
});
```

## MatrickeriaAgregar.tsx

Modal para **crear** una nueva matriz.

Solicita datos básicos y evita duplicados.

- Flujo:
- Selección de **centro de costo** y **tipo de matriz**.
- Ingreso de **número**, **descripción**, **marca** y **fechaFabricación**.
- Validación de existencia (yaExiste).
- Envío con POST /matrickeria.
- Actualiza estado global y cierra formulario.

Figura 4 – Formulario de alta de una nueva matriz con carga de Centro de Costo, tipo, Número, descripción, Marca y fecha de fabricación.

js

```
const mh = respuesta?.datos?.matrickeria as maquinaType[];
if (!mh) {
  mostrarError(respuesta);
  return;
}
```

```
setMatrices(mh);  
setMostrarAgregar(false);
```

Esta documentación detalla la estructura, los flujos de datos y las posibles mejoras de lógica en cada componente.