

/api/reclamo/reclamar

- Método: POST.
- Ruta: /api/reclamo/reclamar.
- Autenticación: requerida. Router aplica `authMiddleware` (JWT Bearer). Controlador exige rol `medico`.
- Controlador: `ReclamoController.reclamarProximoPaciente` → Servicio: `UrgenciaService.reclamarPaciente`.

Request

- Headers: `Authorization: Bearer <jwt>`, `Content-Type: application/json`.
- Body opcional:
 - `ingresoCuil`: string para reclamar un paciente específico; si se omite, toma el primero de la lista ordenada.
 - `nombre, apellido`: strings opcionales del médico; defaults derivados del email o "medico"/"Cuenta".
 - `cuil`: string opcional del médico; default "20123456789", validado como VO Cuil.
 - `matricula`: string opcional; default "MAT-DEFAULT".

Validaciones y reglas

- Rechaza con 403 si el usuario autenticado no tiene `rol === "medico"`.
- VO Cuil y Email del médico se validan; `ingresoCuil` se normaliza dentro del servicio.
- Servicio verifica:
 - Que exista al menos un ingreso pendiente.
 - Que el médico no tenga ya un ingreso en proceso.
 - Que el CUIL solicitado exista en la lista (si se pasa `ingresoCuil`).
 - Asigna el doctor y mantiene estado `EN_PROCESO`.

Flujo interno

1. Construye `Doctor` con datos del JWT y body (aplicando defaults).
2. Llama a `reclamarPaciente` con CUIL objetivo o toma el primero en cola.
3. Devuelve el ingreso asignado serializado.

Respuestas

- 200 { "message": "Paciente reclamado exitosamente", "ingreso": { ... } }.
- 400 si no hay pacientes, si el médico ya tiene uno en proceso o si no se encuentra el CUIL solicitado.
- 403 si el JWT no corresponde a rol médico.
- 401 si falta o es inválido el token.

Consideraciones

- La lista de espera es compartida con `/api/urgencias/registrar`; el orden respeta nivel de emergencia y antigüedad.