/api/auth/register

- Método: POST.
- Ruta: `/api/auth/register`.
- Autenticación: requerida. Router aplica `authMiddleware` (JWT Bearer) y `adminMiddleware` (rol `administrativo`).
- Controlador: `AuthController.registrar` → Servicio: `AuthService.registrarUsuario`.

Request

- Headers: `Authorization: Bearer <jwt>`, `Content-Type: application/json`.
- Body:
    - `email`: string con formato válido.
    - `password`: string, longitud mínima 8.
    - `rol`: string opcional. Se aceptan `"MEDICO"` o `"ENFERMERA"`; cualquier otro valor o ausencia define rol `administrativo`.

Validaciones y reglas

- VO `Email` normaliza a minúsculas y valida regex.
- `password` < 8 caracteres lanza error.
- Verifica que el email no exista ya en el repositorio de usuarios.
- Usa bcrypt para hashear la contraseña y guarda usuario con UUID.
- Payload del JWT de entrada debe incluir `rol: "administrativo"`; de lo contrario responde 403.

Respuestas

- 201 `{ "message": "Usuario registrado exitosamente", "id": "<uuid>" }`.
- 400 por email inválido, contraseña corta o email duplicado.
- 401 si falta/expira token; 403 si el token no es rol administrativo.

Consideraciones

- El rol almacenado se guarda en minúsculas según enum `Rol` (`medico`, `enfermero`, `administrativo`).
- El token que devuelve `login` se firma con expiración por defecto 1h.