

FASE I — BUSINESS (RUTAS ABIERTAS)

ISS-05 — Swagger / OpenAPI (Feature + Registry)

Documentación interactiva de la API Client con OpenAPI 3.0.3, Registry centralizado y endpoints /api/docs

Laboratorio Backend Express 5 + TypeScript 5.9 + Sequelize

Propósito del ISS-05 e Integración Arquitectónica

Objetivos Principales

- Documentar el contrato HTTP completo de la entidad Client bajo la especificación OpenAPI 3.0.3.
- Exponer la documentación de manera interactiva vía Swagger UI en /api/docs y en JSON crudo en /api/docs.json.
- Anotar los esquemas y endpoints bajo la etiqueta 'SIN AUTH' (modalidad abierta de la Fase I).
- Implementar un patrón Registry modular que escale sin acoplar la configuración central a cada feature.

Dependencias y Habilitaciones

- Depende de ISS-03 (CRUD de Clientes): requiere que las rutas /api/clientes y sus contratos DTO existan.
- Habilita ISS-06 (ProductType): establece la plantilla estándar para documentar todos los features subsiguientes.
- Establece la infraestructura de documentación que en la Fase II integrará la seguridad con Bearer Auth.

Arquitectura Modular de Documentación: Feature + Registry

1. Feature Specific

`clients.swagger.ts`

Cada feature define sus propias rutas, esquemas (DTOs) y respuestas en su propia carpeta sin tocar el núcleo.

2. Central Registry

`src/swagger/index.ts`

Consolida las especificaciones Swagger de todos los features y configura swagger-ui-express.

3. App Integration

`src/config/index.ts`

El método `App.docs()` invoca `setupSwagger(this.app)` exponiendo los endpoints `/api/docs` y `/api/docs.json`.

Anatomía de clients.swagger.ts: Tags, Paths y Schemas

Estructura de Paths y Tags

- Tag asignado: 'Clientes (SIN AUTH)' para denotar explícitamente el acceso público de la Fase I.
- Path /api/clientes: Documenta GET (listar/paginar) y POST (crear cliente).
- Path /api/clientes/{id}: Documenta GET (detalle), PUT (reemplazo), PATCH (parcial/status) y DELETE (borrado físico/lógico).
- Parámetros de ruta: ID validado como número entero positivo.

Esquemas y Respuestas de Error

- CreateClientDto: Esquema de entrada para registro con campos obligatorios (first_name, last_name, email, password).
- ClientResponseDto: Esquema de salida sanitizado que EXCLUYE estrictamente el password_hash.
- Manejo de Respuestas HTTP: 200 OK, 201 Created, 400 Bad Request, 404 Not Found, 409 Conflict.
- Estándar de Errores: Objeto JSON unificado { status, message, timestamp }.

El Registry Centralizado: src/swagger/index.ts

Consolidación de Especificaciones OpenAPI

- **Combinación de Paths y Schemas:** El Registry importa clientsSwagger (y subsiguientes) y los fusiona en el objeto principal swaggerDocument.
- **Configuración del Servidor Base:** Define la información global: title: 'Aprendizaje Express API', version: '1.0.0', y servers: [{ url: 'http://localhost:4000' }].
- **Exposición de Endpoints:** Usa swagger-ui-express para montar la interfaz gráfica en /api/docs y un handler JSON en /api/docs.json.
- **Independencia de Features:** Permite incorporar nuevos features (ej. product-types, products) agregando una única línea en el archivo index.ts del registry.

Integración en el Ciclo de Vida: App.docs()

Método docs() en la Clase App

- Invocación centralizada: App.docs() ejecuta `setupSwagger(this.app)` durante la construcción de la aplicación.
- Ubicación en el constructor: Se registra después de los middlewares base (`express.json`, `cors`) y antes/junto al agregador de rutas.
- Desacoplamiento limpio: La clase App no conoce la sintaxis detallada de Swagger, solo delega al módulo especializado.

Endpoints Expuestos al Cliente

- `http://localhost:4000/api/docs`: UI gráfica e interactiva de Swagger para probar peticiones directamente desde el navegador.
- `http://localhost:4000/api/docs.json`: Especificación cruda en JSON consumible por herramientas externas (Postman, Insomnia, generadores de cliente).
- Sin autenticación: Accesibles públicamente durante la Fase I.

Criterios de Cierre (GATE) para ISS-05

Checklist de Verificación GATE

- ✓ **Compilación TypeScript Limpia:** Ejecución de 'npx tsc --noEmit' finaliza con cero errores de compilación.
- ✓ **Endpoint JSON Crudo Correcto:** Petición 'curl -s http://localhost:4000/api/docs.json' retorna el objeto JSON con openapi: '3.0.3'.
- ✓ **Navegación UI Swagger Funcional:** Apertura de http://localhost:4000/api/docs rinde la interfaz gráfica con el tag 'Clientes (SIN AUTH)'.
- ✓ **Ejecución Interactiva Exitosa:** Prueba de peticiones GET /api/clientes y POST /api/clientes directamente desde Swagger UI funcionando correctamente.

Cierre y Siguiente Paso: Hacia ISS-06

ISS-05 Completado con Éxito

La infraestructura de documentación OpenAPI 3.0.3 está desplegada y conectada al ciclo de vida de la aplicación Express.

¿Qué habilita el siguiente módulo? (ISS-06 — ProductType)

En el ISS-06 replicaremos la arquitectura completa (Modelo, CRUD por capas, Seeder con Faker y especificación Swagger en `product-types.swagger.ts`) para la entidad Tipos de Producto sobre `/api/tipos-producto`.