**📚 API Documentation - Gestión Escáneres**

**Microservicio:** gestion\_escaneres  
**Versión:** 1.0  
**Fecha:** Septiembre 2025

**📖 Tabla de Contenidos**

1. [🔗 Endpoints REST](https://claude.ai/chat/809a4b70-9497-49e0-a3c0-a766f590af5a#endpoints-rest)
2. [📦 DTOs de Petición](https://claude.ai/chat/809a4b70-9497-49e0-a3c0-a766f590af5a#dtos-de-peticion)
3. [📤 DTOs de Respuesta](https://claude.ai/chat/809a4b70-9497-49e0-a3c0-a766f590af5a#dtos-de-respuesta)
4. [🏷️ Enumeraciones](https://claude.ai/chat/809a4b70-9497-49e0-a3c0-a766f590af5a#enumeraciones)
5. [❌ Manejo de Errores](https://claude.ai/chat/809a4b70-9497-49e0-a3c0-a766f590af5a#manejo-de-errores)
6. [🌍 Internacionalización](https://claude.ai/chat/809a4b70-9497-49e0-a3c0-a766f590af5a#internacionalizacion)

**🔗 Endpoints REST**

**🏪 MercadoRestController**

**Base Path:** /api/escaner/mercado

| **Método** | **Endpoint** | **Descripción** | **Parámetros** | **Retorna** |
| --- | --- | --- | --- | --- |
| GET | /api/escaner/mercado | Lista todos los mercados | Ninguno | List<MercadoDTORespuesta> |

**🔍 EscanerRestController**

**Base Path:** /api/escaner

| **Método** | **Endpoint** | **Descripción** | **Parámetros** | **Retorna** | **Validación** |
| --- | --- | --- | --- | --- | --- |
| POST | /api/escaner | Crear escáner | @RequestBody EscanerDTOPeticion | EscanerDTORespuesta (201) | @Valid @NotNull |
| GET | /api/escaner/{id} | Obtener escáner por ID | @PathVariable Long id | EscanerDTORespuesta | @NotNull @Positive |
| GET | /api/escaner | Listar todos los escáneres | Ninguno | List<EscanerDTORespuesta> | - |
| GET | /api/escaner/archivados | Listar escáneres archivados | Ninguno | List<EscanerDTORespuesta> | - |
| PUT | /api/escaner/{id} | Actualizar escáner | @PathVariable Long id<br>@RequestBody EscanerDTOPeticion | EscanerDTORespuesta | @NotNull @Positive<br>@Valid @NotNull |
| DELETE | /api/escaner/{id} | Eliminar escáner | @PathVariable Long id | Boolean | @NotNull @Positive |

**⚡ EstadoEscanerRestController**

**Base Path:** /api/escaner/estado

| **Método** | **Endpoint** | **Descripción** | **Parámetros** | **Retorna** | **Validación** |
| --- | --- | --- | --- | --- | --- |
| POST | /api/escaner/estado/{id}/iniciar | Iniciar escáner | @PathVariable Long id | EstadoEscanerDTORespuesta | @NotNull @Positive |
| POST | /api/escaner/estado/{id}/detener | Detener escáner | @PathVariable Long id | EstadoEscanerDTORespuesta | @NotNull @Positive |
| POST | /api/escaner/estado/{id}/archivar | Archivar escáner | @PathVariable Long id | EstadoEscanerDTORespuesta | @NotNull @Positive |
| POST | /api/escaner/estado/{id}/desarchivar | Desarchivar escáner | @PathVariable Long id | EstadoEscanerDTORespuesta | @NotNull @Positive |

**🎯 FiltroRestController**

**Base Path:** /api/escaner/filtro

| **Método** | **Endpoint** | **Descripción** | **Parámetros** | **Retorna** | **Validación** |
| --- | --- | --- | --- | --- | --- |
| GET | /api/escaner/filtro/categorias | Obtener categorías | Ninguno | List<CategoriaDTORespuesta> | - |
| GET | /api/escaner/filtro?categoria={categoria} | Filtros por categoría | @RequestParam EnumCategoriaFiltro | List<FiltroDtoRespuesta> | @NotNull |
| GET | /api/escaner/filtro/defecto?filtro={enumFiltro} | Filtro por defecto | @RequestParam EnumFiltro | FiltroDtoRespuesta | @NotNull |
| GET | /api/escaner/filtro/escaner/{idEscaner} | Filtros de escáner | @PathVariable Long idEscaner | List<FiltroDtoRespuesta> | @NotNull @Positive |
| POST | /api/escaner/filtro/escaner/{idEscaner} | Guardar filtros | @PathVariable Long idEscaner<br>@RequestBody List<FiltroDtoPeticion> | List<FiltroDtoRespuesta> (201) | @NotNull @Positive<br>@Valid @NotNull |
| DELETE | /api/escaner/filtro/{idFiltro} | Eliminar filtro | @PathVariable Long idFiltro | Boolean | @NotNull @Positive |

**📦 DTOs de Petición**

**🏪 MercadoDTOPeticion**

{

"enumMercado": "NYSE" // @NotNull(message = "escaner.mercado.enum.empty")

}

**⚡ TipoEjecucionDTOPeticion**

{

"enumTipoEjecucion": "DIARIA" // @NotNull(message = "tipoEjecucion.empty")

}

**🔍 EscanerDTOPeticion**

{

"nombre": "Mi Escáner", // @NotBlank, @Size(min=3, max=100)

"descripcion": "Descripción", // @NotBlank, @Size(min=5, max=255)

"horaInicio": "09:30:00", // @NotNull

"horaFin": "16:00:00", // @NotNull

"objTipoEjecucion": { // @NotNull

"enumTipoEjecucion": "DIARIA"

},

"mercados": [ // @NotNull

{

"enumMercado": "NYSE"

}

]

}

**🎯 FiltroDtoPeticion**

{

"enumFiltro": "VOLUME", // @NotNull(message = "filtro.enumFiltro.empty")

"parametros": [ // @NotNull, @Valid

{

"enumParametro": "CONDICION",

"objValorSeleccionado": {

"enumTipoValor": "CONDICIONAL",

"enumCondicional": "MAYOR\_QUE",

"valor1": 1000000

}

}

]

}

**🔧 ParametroDTOPeticion**

{

"enumParametro": "CONDICION", // @NotNull

"objValorSeleccionado": { // @NotNull, @Valid

"enumTipoValor": "INTEGER",

"valor": 100

}

}

**💰 Valores DTOPeticion (Polimórfico)**

**Base:** ValorDTOPeticion

{

"enumTipoValor": "INTEGER" // @NotNull

}

**Integer:**

{

"enumTipoValor": "INTEGER",

"valor": 100 // @NotNull

}

**Float:**

{

"enumTipoValor": "FLOAT",

"valor": 10.5 // @NotNull

}

**String:**

{

"enumTipoValor": "STRING",

"valor": "texto" // @NotBlank

}

**Condicional:**

{

"enumTipoValor": "CONDICIONAL",

"enumCondicional": "ENTRE", // @NotNull

"valor1": 10, // @NotNull

"valor2": 100 // Opcional

}

**📤 DTOs de Respuesta**

**🏪 MercadoDTORespuesta**

{

"etiqueta": "Mercado de Nueva York",

"enumMercado": "NYSE"

}

**🔍 EscanerDTORespuesta**

{

"idEscaner": 1,

"nombre": "Mi Escáner",

"descripcion": "Descripción del escáner",

"horaInicio": "09:30:00",

"horaFin": "16:00:00",

"fechaCreacion": "2025-09-15",

"mercados": [

{

"etiqueta": "Mercado de Nueva York",

"enumMercado": "NYSE"

}

],

"objEstado": {

"enumEstadoEscaner": "DETENIDO",

"fechaRegistro": "2025-09-15"

},

"objTipoEjecucion": {

"etiqueta": "Diaria",

"enumTipoEjecucion": "DIARIA"

}

}

**⚡ EstadoEscanerDTORespuesta**

{

"enumEstadoEscaner": "INICIADO",

"fechaRegistro": "2025-09-15"

}

**🎯 FiltroDtoRespuesta**

{

"enumFiltro": "VOLUME",

"etiquetaNombre": "Volumen",

"etiquetaDescripcion": "Filtro por volumen de transacciones",

"objCategoria": {

"enumCategoriaFiltro": "VOLUMEN",

"etiqueta": "Volumen"

},

"parametros": [

{

"enumParametro": "CONDICION",

"etiqueta": "Condición",

"objValorSeleccionado": {

"etiqueta": "Mayor que 1,000,000",

"enumTipoValor": "CONDICIONAL",

"enumCondicional": "MAYOR\_QUE",

"valor1": 1000000

},

"opciones": [

{

"etiqueta": "Mayor que",

"enumTipoValor": "CONDICIONAL",

"enumCondicional": "MAYOR\_QUE"

}

]

}

]

}

**🏷️ Enumeraciones**

⚠️ **IMPORTANTE:** El backend espera las representaciones string de los enums (ej. "VOLUME", "NYSE"), no valores numéricos. Por eso usamos string enums en TypeScript.

**🔑 Enums Principales**

**EnumMercado**

export enum EnumMercado {

NYSE = "NYSE",

NASDAQ = "NASDAQ",

AMEX = "AMEX",

ETF = "ETF",

OTC = "OTC",

}

**EnumEstadoEscaner**

export enum EnumEstadoEscaner {

ARCHIVADO = "ARCHIVADO",

INICIADO = "INICIADO",

DETENIDO = "DETENIDO",

DESARCHIVADO = "DESARCHIVADO",

}

**EnumTipoEjecucion**

export enum EnumTipoEjecucion {

UNA\_VEZ = "UNA\_VEZ",

DIARIA = "DIARIA",

}

**EnumCategoriaFiltro**

export enum EnumCategoriaFiltro {

TODOS = "TODOS",

VOLUMEN = "VOLUMEN",

PRECIO\_Y\_MOVIMIENTO = "PRECIO\_Y\_MOVIMIENTO",

VOLATILIDAD = "VOLATILIDAD",

MOMENTUM\_E\_INDICADORES\_TECNICOS = "MOMENTUM\_E\_INDICADORES\_TECNICOS",

CARACTERISTICAS\_FUNDAMENTALES = "CARACTERISTICAS\_FUNDAMENTALES",

TIEMPO\_Y\_PATRONES\_DE\_PRECIO = "TIEMPO\_Y\_PATRONES\_DE\_PRECIO",

}

**EnumTipoValor**

export enum EnumTipoValor {

INTEGER = "INTEGER",

FLOAT = "FLOAT",

STRING = "STRING",

CONDICIONAL = "CONDICIONAL",

}

**🎯 Enums de Valores Específicos**

**EnumCondicional**

export enum EnumCondicional {

MAYOR\_QUE = "MAYOR\_QUE",

MENOR\_QUE = "MENOR\_QUE",

ENTRE = "ENTRE",

FUERA = "FUERA",

IGUAL\_A = "IGUAL\_A",

}

**EnumTimeframe**

export enum EnumTimeframe {

\_1M = "\_1M",

\_2M = "\_2M",

\_3M = "\_3M",

\_5M = "\_5M",

\_10M = "\_10M",

\_15M = "\_15M",

\_30M = "\_30M",

\_45M = "\_45M",

\_1H = "\_1H",

\_2H = "\_2H",

\_3H = "\_3H",

\_4H = "\_4H",

\_12H = "\_12H",

\_1D = "\_1D",

\_2D = "\_2D",

\_3D = "\_3D",

\_1W = "\_1W",

\_1MO = "\_1MO",

\_3MO = "\_3MO",

\_6MO = "\_6MO",

\_1Y = "\_1Y",

}

**Otros Enums Importantes**

export enum EnumTipoPatron {

BEARISH = "BEARISH",

BULLISH = "BULLISH",

}

export enum EnumModoPromedioMovil {

EMA = "EMA",

SMA = "SMA",

VMA = "VMA",

RMA = "RMA",

}

export enum EnumEstadoNoticia {

COMPRA = "COMPRA",

VENTA = "VENTA",

NINGUNA = "NINGUNA",

}

export enum EnumCondicionFirstCandle {

ALCISTA = "ALCISTA",

BAJISTA = "BAJISTA",

}

export enum EnumDireccionRompimiento {

ABOVE = "ABOVE",

BELOW = "BELOW",

}

export enum EnumLineaCruce {

VWAP = "VWAP",

EMA = "EMA",

}

export enum EnumLineaReferencia {

VWAP = "VWAP",

MA = "MA",

EMA = "EMA",

}

export enum EnumNivelCruce {

OPEN = "OPEN",

CLOSE = "CLOSE",

VWAP = "VWAP",

EMA = "EMA",

}

export enum EnumOpcionExtremo {

HIGH = "HIGH",

LOW = "LOW",

}

export enum EnumPuntoReferencia {

OPEN = "OPEN",

CLOSE = "CLOSE",

CLOSE\_POST\_MARKET = "CLOSE\_POST\_MARKET",

CLOSE\_PRE\_MARKET = "CLOSE\_PRE\_MARKET",

}

export enum EnumTipoRol {

RESISTENCIA = "RESISTENCIA",

SOPORTE = "SOPORTE",

}

export enum EnumTipoValorMedida {

PRECIO = "PRECIO",

PORCENTAJE = "PORCENTAJE",

}

export enum EnumTipoVolumen {

CIERRE = "CIERRE",

APERTURA = "APERTURA",

}

**💡 Ejemplo de Uso Correcto**

// ✅ Correcto - Envía "NYSE" al backend

const payload = {

enumMercado: EnumMercado.NYSE // Valor: "NYSE"

};

// ✅ Correcto - Comparación

if (response.enumMercado === EnumMercado.NYSE) {

// lógica

}

// ✅ Correcto - En peticiones HTTP

const filtro = {

enumFiltro: "VOLUME", // El backend recibe "VOLUME"

// o usando el enum:

enumFiltro: EnumFiltro.VOLUME, // También envía "VOLUME"

};

**❌ Manejo de Errores**

**📋 Estructura de Error Principal**

{

"codigoError": "GC-0003",

"mensaje": "La entidad solicitada no fue encontrada, El escáner con ID 123 no existe",

"codigoHttp": 404,

"url": "/api/escaner/123",

"metodo": "GET"

}

**🚨 Códigos de Error y Excepciones**

| **Código** | **Descripción** | **HTTP Status** | **Excepción** |
| --- | --- | --- | --- |
| GC-0001 | Error genérico | 500 | Exception |
| GC-0002 | Entidad ya existe | 406 | EntidadYaExisteException |
| GC-0003 | Entidad no encontrada | 404 | EntidadNoExisteException |
| GC-0004 | Violación de regla de negocio | 400 | ReglaNegocioException |
| GC-0005 | Estado denegado | 400 | EstadoDenegadoException |
| GC-0006 | Tipo de argumento inválido | 400 | MethodArgumentTypeMismatchException |

**🔍 Tipos de Respuestas de Error Detalladas**

**1. Error Genérico (500)**

{

"codigoError": "GC-0001",

"mensaje": "Ha ocurrido un error interno del servidor",

"codigoHttp": 500,

"url": "/api/escaner/123",

"metodo": "GET"

}

**2. Entidad Ya Existe (406)**

{

"codigoError": "GC-0002",

"mensaje": "La entidad ya existe, Ya existe un escáner con el nombre 'Mi Escáner'",

"codigoHttp": 406,

"url": "/api/escaner",

"metodo": "POST"

}

**3. Entidad No Encontrada (404)**

{

"codigoError": "GC-0003",

"mensaje": "La entidad solicitada no fue encontrada, El escáner con ID 999 no existe",

"codigoHttp": 404,

"url": "/api/escaner/999",

"metodo": "GET"

}

**4. Regla de Negocio (400)**

{

"codigoError": "GC-0004",

"mensaje": "Violación de regla de negocio, No se puede eliminar un escáner que está en ejecución",

"codigoHttp": 400,

"url": "/api/escaner/123",

"metodo": "DELETE"

}

**5. Estado Denegado (400)**

{

"codigoError": "GC-0005",

"mensaje": "Violación de regla de negocio, No se puede iniciar un escáner que ya está en ejecución",

"codigoHttp": 400,

"url": "/api/escaner/estado/123/iniciar",

"metodo": "POST"

}

**6. Tipo de Argumento Inválido (400)**

{

"codigoError": "GC-0006",

"mensaje": "Tipo de argumento inválido, El campo 'id' debe ser un número válido",

"codigoHttp": 400,

"url": "/api/escaner/abc",

"metodo": "GET"

}

**📝 Errores de Validación**

**Validación de DTOs (MethodArgumentNotValidException)**

{

"nombre": "El nombre del escáner no puede estar vacío",

"descripcion": "La descripción debe tener entre 5 y 255 caracteres",

"horaInicio": "La hora de inicio es requerida"

}

**Validación de Filtros (ValidacionFiltroException)**

{

"CONDICION": "La condición es requerida para el filtro de volumen",

"VALOR\_MINIMO": "El valor mínimo debe ser mayor que cero",

"TIMEFRAME": "El timeframe seleccionado no es válido"

}

**🎯 Ejemplos de Casos de Error Específicos**

**Error al crear escáner duplicado**

POST /api/escaner

**Respuesta:** 406 Not Acceptable

{

"codigoError": "GC-0002",

"mensaje": "La entidad ya existe, Ya existe un escáner con el nombre 'Análisis Diario'",

"codigoHttp": 406,

"url": "/api/escaner",

"metodo": "POST"

}

**Error al intentar iniciar escáner archivado**

POST /api/escaner/estado/123/iniciar

**Respuesta:** 400 Bad Request

{

"codigoError": "GC-0005",

"mensaje": "Violación de regla de negocio, No se puede iniciar un escáner archivado",

"codigoHttp": 400,

"url": "/api/escaner/estado/123/iniciar",

"metodo": "POST"

}

**Error de validación en filtros**

POST /api/escaner/filtro/escaner/123

**Respuesta:** 400 Bad Request

{

"VOLUMEN\_MINIMO": "El volumen mínimo debe ser mayor que 1000",

"PRECIO\_MAXIMO": "El precio máximo no puede ser negativo",

"TIMEFRAME": "El timeframe es requerido para indicadores técnicos"

}

**🔧 Manejo Recomendado en Frontend**

// Tipo para errores de la API

interface ApiError {

codigoError: string;

mensaje: string;

codigoHttp: number;

url: string;

metodo: string;

}

// Tipo para errores de validación

interface ValidationErrors {

[key: string]: string;

}

// Función para manejar errores

const handleApiError = (error: any) => {

if (error.response) {

const status = error.response.status;

const data = error.response.data;

switch (status) {

case 400:

// Puede ser Error o ValidationErrors

if (data.codigoError) {

// Es un Error estructurado

showError(`${data.codigoError}: ${data.mensaje}`);

} else {

// Es un objeto de errores de validación

Object.entries(data).forEach(([field, message]) => {

showFieldError(field, message as string);

});

}

break;

case 404:

showError('Recurso no encontrado');

break;

case 406:

showError('El recurso ya existe');

break;

case 500:

showError('Error interno del servidor');

break;

default:

showError('Error desconocido');

}

}

};

**🌍 Internacionalización**

**🎯 Campo etiqueta**

* Los DTOs de respuesta incluyen campos etiqueta con textos traducidos
* El frontend puede mostrar estos textos directamente
* No requiere lógica de traducción adicional

**🌐 Solicitar Idioma**

**Header requerido:**

Accept-Language: es // Español

Accept-Language: en // Inglés

Accept-Language: fr // Francés

**📝 Ejemplo de Petición**

GET /api/escaner/mercado

Accept-Language: es

Content-Type: application/json

**📤 Ejemplo de Respuesta**

[

{

"etiqueta": "Mercado de Nueva York",

"enumMercado": "NYSE"

},

{

"etiqueta": "NASDAQ",

"enumMercado": "NASDAQ"

}

]

**🔧 Mensajes de Error Internacionalizados**

**Español (Accept-Language: es):**

{

"codigoError": "GC-0003",

"mensaje": "La entidad solicitada no fue encontrada, El escáner con ID 123 no existe",

"codigoHttp": 404

}

**Inglés (Accept-Language: en):**

{

"codigoError": "GC-0003",

"mensaje": "The requested entity was not found, Scanner with ID 123 does not exist",

"codigoHttp": 404

}

**📋 Notas Importantes**

**✅ Buenas Prácticas**

1. **Siempre incluir** Accept-Language en las peticiones
2. **Validar** los DTOs de petición antes de enviar
3. **Manejar todos** los códigos de error posibles
4. **Usar** los campos etiqueta para mostrar textos al usuario
5. **Implementar** manejo de errores específico por tipo de excepción

**⚠️ Consideraciones**

* Los IDs deben ser números positivos (@Positive)
* Los campos requeridos no pueden ser null (@NotNull)
* Las cadenas no pueden estar vacías (@NotBlank)
* Respetar los límites de tamaño (@Size)
* Los errores pueden ser objetos Error o mapas de validación

**🔄 Flujo Típico de Uso**

1. **Obtener** categorías de filtros disponibles
2. **Seleccionar** filtros por categoría
3. **Configurar** parámetros para cada filtro
4. **Crear/actualizar** escáner con filtros
5. **Gestionar** estado del escáner (iniciar/detener)
6. **Manejar** errores apropiadamente según su tipo

**🚨 Manejo de Errores Críticos**

* **Error 500:** Reportar al equipo de desarrollo
* **Error 406:** Verificar unicidad de datos antes de enviar
* **Error 404:** Verificar existencia de recursos
* **Error 400:** Validar datos y reglas de negocio
* **Errores de validación:** Mostrar mensajes específicos por campo