

Manual de Usuario - HybridLogisticsHub

Sistema de Gestión Logística con Tracking en Tiempo Real

Jhordan Huamaní Huamán¹, Jorge Ortiz Castañeda¹, Miguel Flores León¹

¹Departamento de Ingeniería de Software, Universidad La Salle de Arequipa, Perú
{jhordanhh, jortizc, mfloresl}@ulasalle.edu.pe

Resumen—Este documento sirve como manual de usuario para el sistema HybridLogisticsHub. Se detallan los procedimientos de instalación, configuración y operación de la plataforma de gestión logística y visualización de rutas en tiempo real.

Index Terms—Logística, Tracking, Mapa Interactivo, Python, Docker, Manual de Usuario.

I. INTRODUCCIÓN

HybridLogisticsHub es un sistema integral de gestión logística diseñado para empresas de distribución y mensajería. La aplicación permite:

- **Gestionar órdenes de envío** con información detallada de clientes y destinos.
- **Visualizar rutas en tiempo real** sobre un mapa interactivo.
- **Simular entregas** con vehículos animados siguiendo rutas reales por calles.
- **Monitorear estadísticas** de órdenes pendientes, en tránsito y entregadas.

I-A. ¿Para quién es esta aplicación?

- Operadores logísticos.
- Supervisores de flotas.
- Empresas de delivery y mensajería.
- Personal de despacho y seguimiento.

II. REQUISITOS PREVIOS

II-A. Sistema Operativo

- **Windows 10/11** (recomendado)
- **Linux** (Ubuntu 20.04+)
- **macOS** (10.15+)

II-B. Software Necesario

Se requiere la instalación del software listado en la Tabla I.

Cuadro I
SOFTWARE NECESARIO

Software	Ver. Mínima	Fuente
Python	3.10+	python.org
Docker Desktop	4.0+	docker.com
Git	2.30+	git-scm.com
Navegador Web	Actualizado	-

II-C. Conexión a Internet

Se requiere conexión a internet para cargar el mapa (OpenStreetMap) y calcular rutas reales (OpenRouteService API).

III. INSTALACIÓN

III-A. Paso 1: Clonar el Repositorio

```
1 git clone https://github.com/JorLoRt/BaseDeGatos2Final.git
2 cd HybridLogisticsHub
```

III-B. Paso 2: Iniciar las Bases de Datos

```
1 docker-compose up -d
```

Nota: Espera aprox. 10-15 segundos para que los contenedores estén listos.

III-C. Paso 3: Instalar Dependencias Python

```
1 pip install -r requirements.txt
```

III-D. Paso 4: Inicializar la Base de Datos

Este comando crea las tablas y genera **100 órdenes de ejemplo** en Arequipa.

```
1 python init_db.py
```

III-E. Paso 5: Iniciar el Servidor

```
1 python -m uvicorn main:app --reload --port 8000
```

III-F. Paso 6: Acceder a la Aplicación

- **Tracking Visual:** http://localhost:8000/static/tracking_visual.html
- **API Docs:** <http://localhost:8000/docs>

IV. PANTALLA PRINCIPAL

IV-A. Vista General de la Interfaz

La vista general de la aplicación se presenta en la Fig. 1. La interfaz se divide en dos secciones principales: Panel Lateral y Mapa Interactivo.

El esquema visual se compone de:

1. **Panel Lateral:** Contiene estadísticas, lista de órdenes y panel de simulación.
2. **Mapa Interactivo:** Muestra marcadores de origen (verde), destino (rojo), rutas y vehículos.

IV-B. Componentes del Panel Lateral

IV-B1. Barra de Estadísticas: Como se aprecia en la Fig. 2, esta barra muestra en tiempo real las órdenes Pendientes, En Tránsito y Entregadas.

IV-B2. Lista de Órdenes: La lista de órdenes (ver Fig. 3) detalla en cada tarjeta el ID, Estado, Descripción, Dirección y el botón para iniciar la simulación.

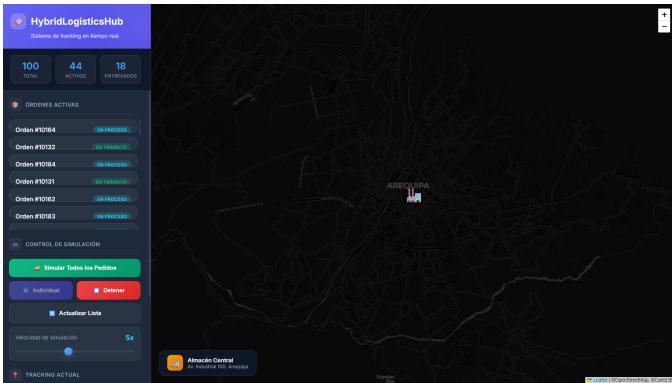


Figura 1. Pantalla Principal del Sistema

IV-B3. Panel de Simulación Múltiple: Este panel, mostrando en la Fig. 4, permite configurar y simular múltiples entregas simultáneamente.

IV-C. Estados de Órdenes

Los estados se diferencian por colores como se muestra en la Tabla II.

Cuadro II
ESTADOS DE LAS ÓRDENES

Estado	Color	Descripción
Pendiente	Amarillo	Creada, en espera
En Proceso	Cyan	Siendo preparada
En Tránsito	Verde	En camino
Entregado	Gris	Completada
Cancelado	Rojo	Cancelada

V. FUNCIONALIDADES PASO A PASO

V-A. Visualizar Órdenes en el Mapa

Para visualizar una orden:

1. Selecciona una orden de la lista.
2. El mapa se centrará en el destino.
3. Aparecerá un marcador rojo en la entrega, tal como se muestra en la Fig. 5.

V-B. Simular Entrega Individual

1. Ubica la orden en la lista.
2. Haz clic en el botón "Simular".
3. Aparecerá un marcador verde (origen) y se trazará la ruta (ver Fig. 6).
4. Un vehículo comenzará a moverse mostrando progreso y velocidad.
5. Al finalizar, el estado cambiará a Entregado".

V-C. Simular Múltiples Entregas

1. Ve a la sección "Simulación Múltiple".
2. Configura la cantidad de vehículos como se indica en la Fig. 7.
3. Haz clic en Iniciar Simulación.



Figura 2. Barra de Estadísticas

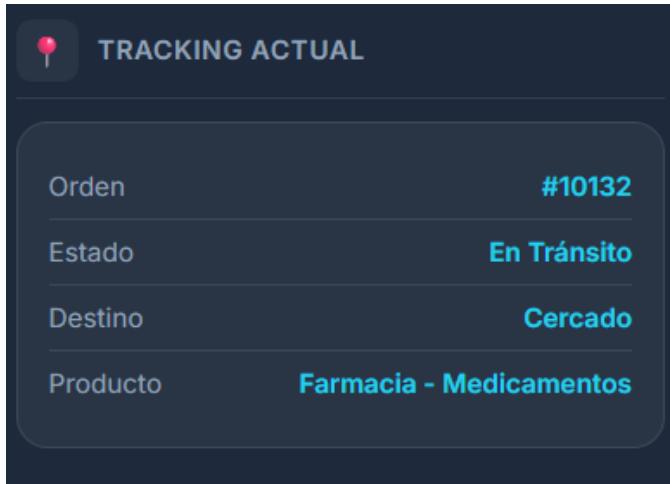


Figura 3. Lista de Órdenes

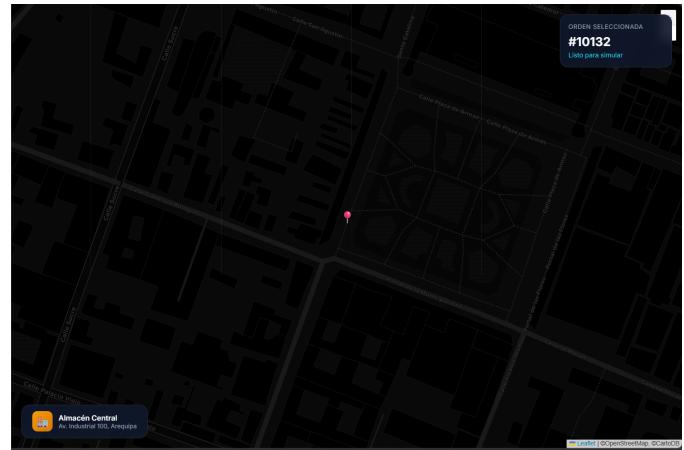


Figura 5. Selección de Orden



Figura 4. Panel de Simulación Múltiple

V-D. Usar la API REST

Accede a la documentación en [/docs](#).

Ejemplo: Crear Orden (JSON)

```

1 {
2   "cliente": {
3     "nombre": "Maria Garcia",
4     "email": "maria@email.com",
5     "telefono": "+51 999 888 777"
6   },
7   "descripcion": "Paquete electronico",
8   "direccion_origen": "Centro Dist., Arequipa",
9   "direccion_destino": "Av. Ejercito 1200"
10 }

```

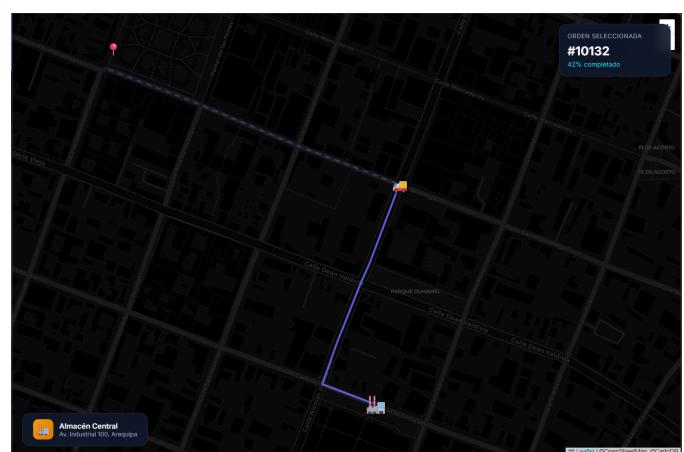


Figura 6. Simulación Individual

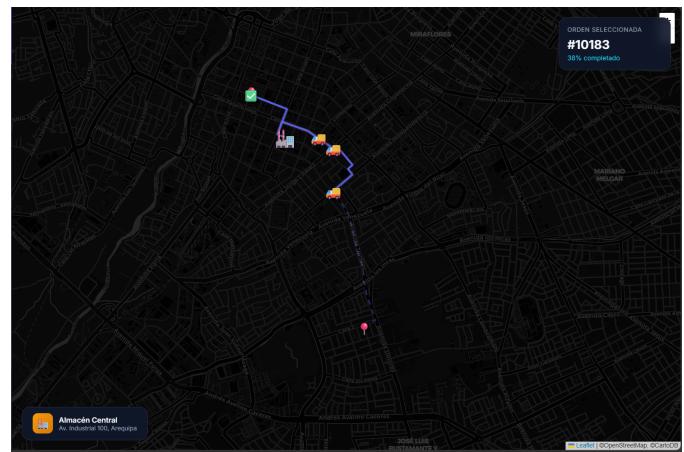


Figura 7. Simulación Múltiple

VI. REFERENCIA DETALLADA DE ENDPOINTS API

Esta sección describe cómo utilizar cada endpoint disponible en el sistema, detallando los parámetros requeridos y el

flujo de datos entre PostgreSQL y MongoDB.

VI-A. Sistema y Diagnóstico

Endpoints utilitarios para verificar la salud de la aplicación.

- **GET /**: Muestra un mapa de todos los endpoints disponibles y la versión del sistema, como se muestra en la Fig. 8.
 - **GET /health**: Verifica la conexión activa con PostgreSQL y MongoDB. Ver resultado esperado en la Fig. 9.

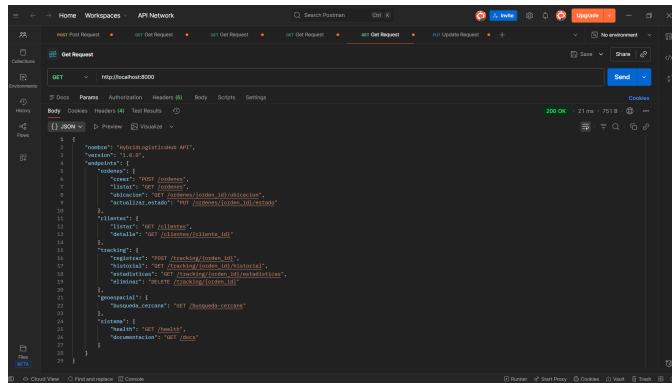


Figura 8. Respuesta del Endpoint Raíz (Info)

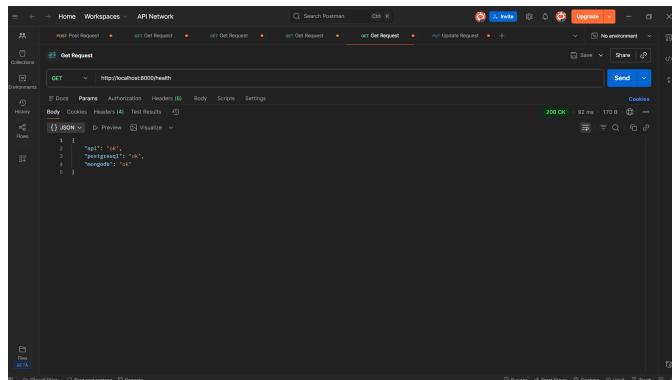


Figura 9. Chequeo de Estado (Health Check)

VI-B. Gestión de Clientes (PostgreSQL)

- **GET /clientes:** Obtiene una lista de clientes registrados junto con el conteo de sus órdenes. Permite paginación mediante el parámetro `límite` (ver Fig. 10).
 - **GET /clientes/{cliente_id}:** Obtiene la información detallada de un cliente específico y la lista de sus órdenes asociadas. Si no existe, retorna 404 (ver Fig. 11).

VI-C. Gestión de Órdenes (PostgreSQL + Híbrido)

- **POST /ordenes:** Crea una nueva orden. Aplica lógica transaccional ACID para crear o reutilizar el cliente basado en el email. Ver ejemplo de Body y respuesta en la Fig. 12.

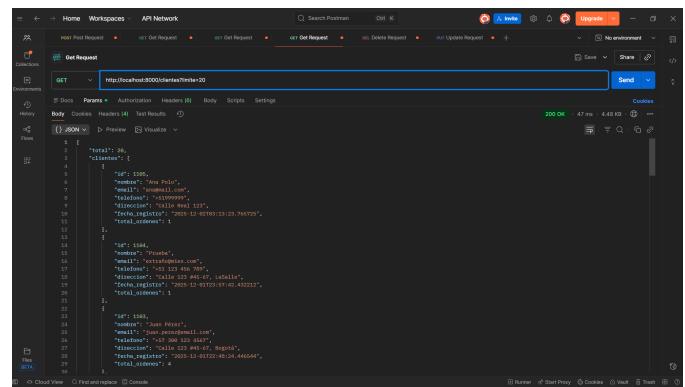


Figura 10. Listado de Clientes

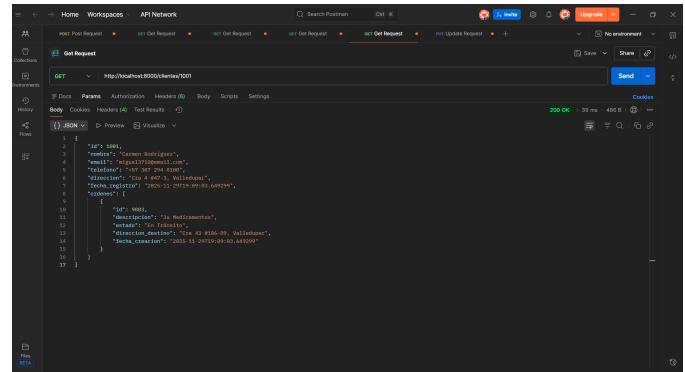


Figura 11. Detalle de Cliente Específico

```
1 {  
2   "cliente": {  
3     "nombre": "Ana Polo",  
4     "email": "ana@mail.com",  
5     "telefono": "+519999999",  
6     "direccion": "Calle Real 123"  
7   },  
8   "descripcion": "Laptop Gamer",  
9   "direccion_origen": "Almacen A",  
10  "direccion_destino": "Av. Sol 456"  
11 }
```

- **GET /ordenes:** Lista órdenes registradas permitiendo filtrar por estado (ej. En Tránsito) como se aprecia en la Fig. 13.
 - **GET /ordenes/{orden_id}:** Obtiene los datos administrativos de una orden específica desde PostgreSQL (ver Fig. 14).
 - **GET /ordenes/{orden_id}/ubicacion: Consulta Híbrida.** Combina los datos de la orden (SQL) con la última posición registrada (NoSQL). Ver estructura unificada en la Fig. 15.
 - **PUT /ordenes/{orden_id}/estado:** Cambia el estado de la orden. Si el estado es Entregado, finaliza el tracking en MongoDB (ver Fig. 16).

VI-D. Tracking GPS (*MongoDB*)

- **POST /tracking/{orden_id}**: Registra una nueva coordenada GeoJSON para una orden activa. Usado por dispositivos GPS (ver Fig. 17).

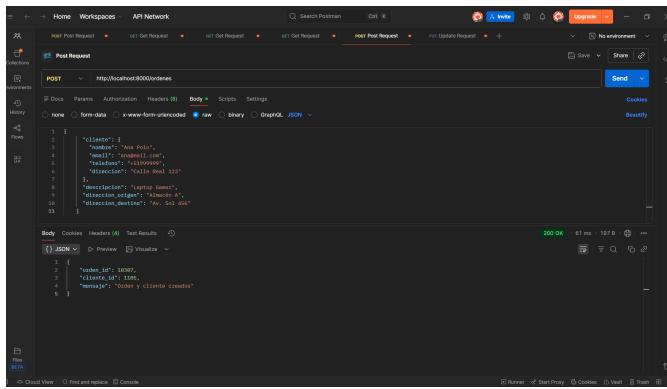


Figura 12. Creación de Orden (POST)

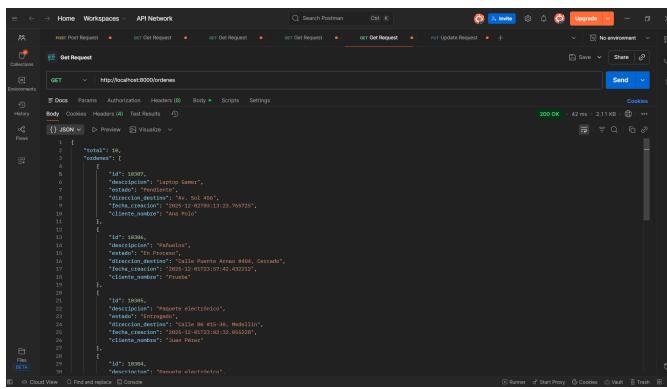


Figura 13. Listado de Órdenes Filtrado

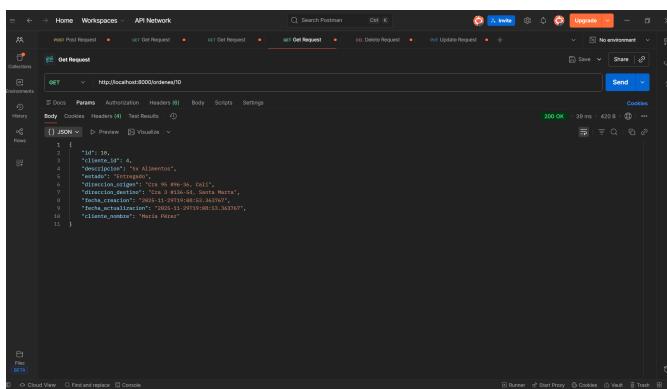


Figura 14. Detalle Administrativo de Orden

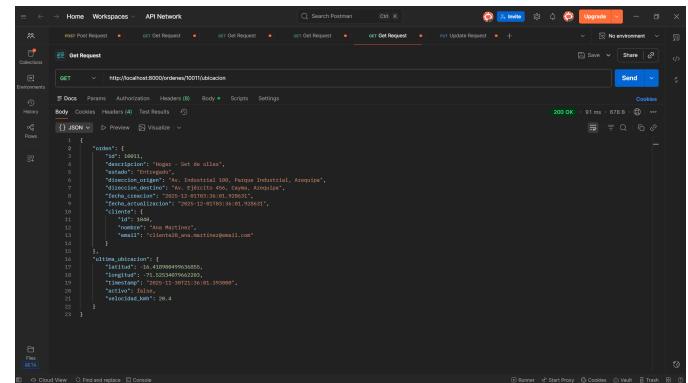


Figura 15. Consulta Híbrida de Ubicación

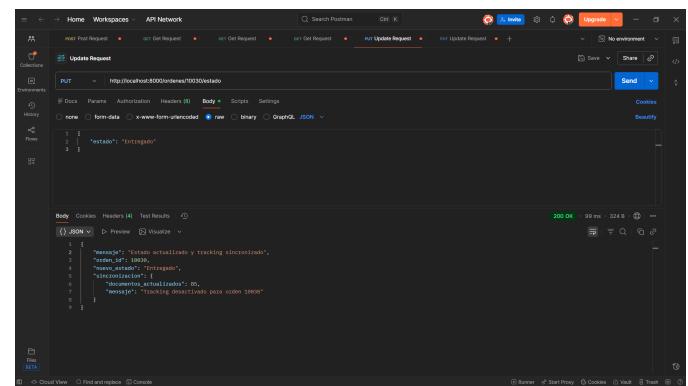


Figura 16. Actualización de Estado

- **GET /tracking/{orden_id}/historial:** Recupera la lista completa de coordenadas para dibujar la ruta en el mapa (ver Fig. 18).
 - **GET /tracking/{orden_id}/estadisticas:** Calcula métricas (velocidad promedio, tiempos) basándose en documentos de MongoDB (ver Fig. 19).
 - **DELETE /tracking/{orden_id}:** Elimina todos los registros de ubicación de una orden (ver Fig. 20).

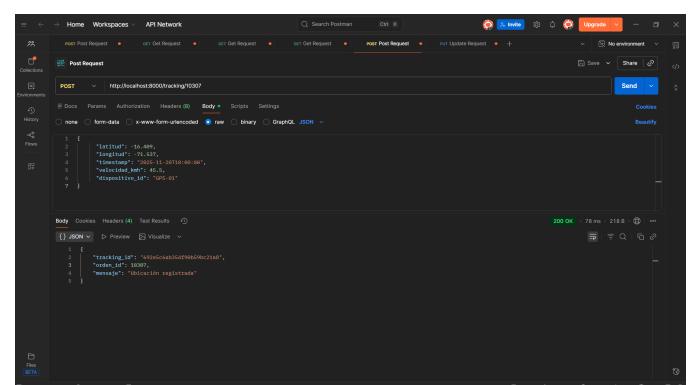


Figura 17. Registro de Punto GPS

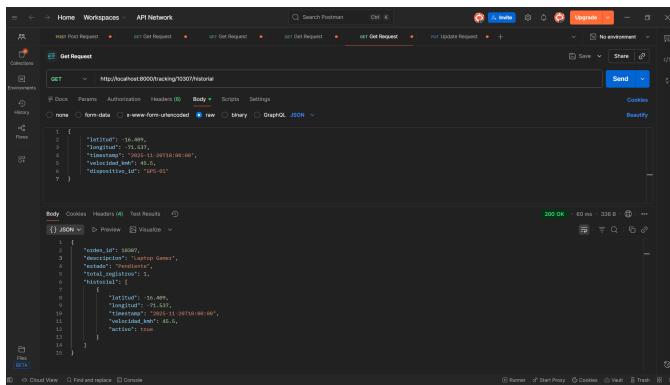


Figura 18. Historial de Ruta GPS

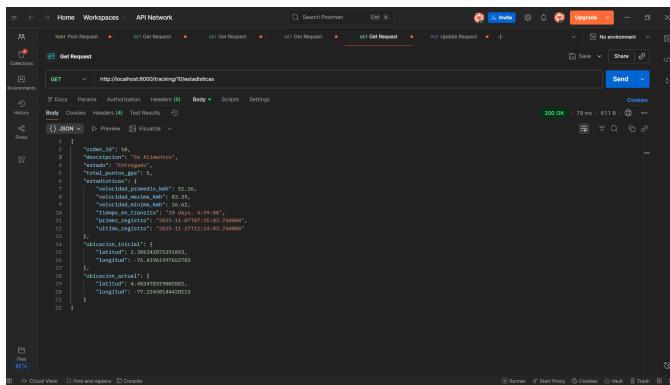


Figura 19. Estadísticas de Envío

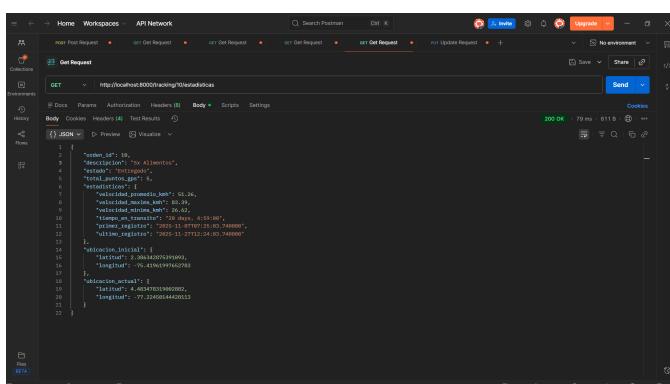


Figura 20. Eliminación de Tracking

VI-E. Geoespacial (MongoDB - 2dsphere)

- **GET /busqueda-cercana:** Encuentra órdenes activas dentro de un radio específico (metros) desde un punto central (latitud/longitud). Utiliza el índice \$near de MongoDB. Ver resultado en la Fig. 21.

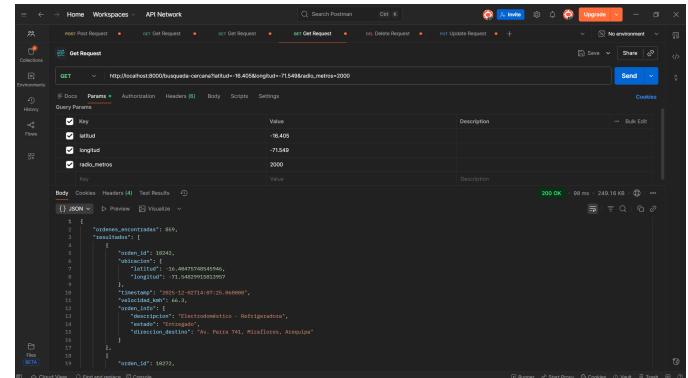


Figura 21. Radar de Órdenes Cercanas

VII. CASOS DE USO COMUNES

VII-A. Monitoreo de Entregas del Día

Escenario: Un supervisor necesita ver el estado de todas las entregas activas.

Pasos: Filtrar por En Tránsito en la interfaz (Fig. 2) y revisar la ubicación en el mapa.

VII-B. Simular Ruta de Entrega

Escenario: Verificar la ruta que tomará un repartidor.

Pasos: Usar la función "Simular"(Fig. 6) en una orden específica para ver el trazado por calles reales.

VII-C. Prueba de Carga

Escenario: Probar el sistema con múltiples entregas.

Pasos: Usar "Simulación Múltiple"(Fig. 7) con 5-10 vehículos.

VII-D. Consultar Historial

Escenario: Cliente pregunta por su envío.

Pasos: Usar el endpoint GET /tracking/{orden_id}/historial.

VIII. PREGUNTAS FRECUENTES (FAQ)

P: ¿Por qué no carga el mapa?

R: Verifica tu conexión a internet. OpenStreetMap requiere conexión activa.

P: ¿Por qué la ruta no se muestra?

R: OpenRouteService tiene límites de uso gratuito. Espera unos minutos si has excedido el límite.

P: ¿Cómo reinicio los datos?

R: Ejecuta:

```
1 docker-compose down -v
2 docker-compose up -d
3 python init_db.py
```

P: ¿Cómo cambio la ubicación inicial?

R: En `tracking_visual.html`, modifica las coordenadas en la inicialización de Leaflet:

```
1 const map = L.map('map').setView([-16.409, -71.537], 13);
```

IX. CONTACTO Y SOPORTE

IX-A. Desarrolladores

Este proyecto fue desarrollado para el curso de **Sistemas de Bases de Datos II** en la **Universidad de La Salle**.

IX-B. Soporte Técnico

Para reportar problemas, utilice GitHub Issues en el repositorio oficial.

X. HISTORIAL DE VERSIONES

Cuadro III
HISTORIAL DE VERSIONES

Versión	Fecha	Cambios
1.0.0	Nov 2025	Versión inicial con tracking visual