

2025



FRESH&GO

DE LA TIENDA A TU MESA EN UN CLICK

FRESH&GO 2

MAR ALEJANDRA QUISPE ESCALANTE Y JESUS LACRUZ

Índice

1	Introducción.....	2
2	Cambios y ajustes	2
3	Estructura técnica.....	3
3.1	CRM Service (Node.js + Express)	3
3.2	IoT Service (FastAPI + Python).....	4
4	Funcionamiento General	5
5	Problemas encontrados.....	6
6	Simulación.....	7

1 Introducción

La Práctica 2 representa el paso de un diseño teórico (los JSON Schemas definidos en la Práctica 1) hacia la creación de microservicios reales y ejecutables. En esta fase se construyen dos servicios independientes:

- CRM Service → Node.js + Express
- IoT Service → FastAPI + Python

Ambos funcionan exclusivamente con datos simulados en archivos JSON, lo que permite probar la lógica de negocio sin necesidad de base de datos.

Esta práctica sirve como fase de simulación profesional del proyecto Fresh&Go.

2 Cambios y Ajustes Realizados

Durante la Práctica 2 se realizaron cambios clave respecto a la Práctica 1:

- Se eliminaron campos irrelevantes en sensores, lecturas y clientes.
- Se reorganizaron los JSON para evitar datos redundantes.
- Se separaron correctamente entidades (clientes, vehículos, sensores, lecturas).
- Se añadió validación (AJV en CRM, Pydantic en IoT).
- Se crearon controllers, routes y middlewares.
- Se estableció un sistema de manejo de errores unificado.
- Estructura modular en ambos servicios.

3 Estructura técnica

3.1 CRM Services(Node.js + Express)

```
crm/  
  data/  
    clientes.json      # Datos simulados de clientes  
    pedidos.json       # Listado de pedidos asociados a clientes  
    proveedores.json   # Información básica de proveedores  
    conductores.json   # Conductores y su disponibilidad  
  
routes/  
  clientes.js        # Define las rutas del módulo clientes  
  pedidos.js          # Rutas para pedidos  
  proveedores.js     # Rutas para proveedores  
  conductores.js      # Rutas para conductores  
  
controllers/  
  clientesController.js # Lógica de negocio de clientes  
  pedidosController.js # Lógica de pedidos  
  proveedoresController.js # Funciones del módulo proveedores  
  conductoresController.js # Gestión de conductores  
  
schemas/  
  cliente.schema.json # Validación AJV del cliente  
  pedido.schema.json  # Validación de pedido  
  conductor.schema.json # Validación de conductores  
  proveedor.schema.json # Validación de proveedores  
  
middleware/  
  errorHandler.js     # Formato y gestión de errores  
  
index.js             # Configuración del servidor Express.
```

```
3.2    IoT Service (FastAPI + Python)

iot/
  data/
    sensores.json      # Simulación de sensores instalados
    lecturas.json      # Lecturas de temperatura, humedad y ubicación
    vehiculos.json     # Vehículos en los que se instalan sensores
  routes/
    sensores_router.py # Define las rutas del módulo sensores
    lecturas_router.py # Rutas del módulo lecturas
    vehiculos_router.py # Rutas del módulo vehículos
  controllers/
    sensores_controller.py # Lógica del comportamiento de sensores
    lecturas_controller.py # Filtros y búsquedas de lecturas
    vehiculos_controller.py # Gestión de los vehículos y su estado
  schemas/
    sensor_schema.py    # Validación de sensores con Pydantic
    lectura_schema.py   # Validación de lecturas
    vehiculo_schema.py  # Validación de vehículos
  exceptions/
    error_handler.py    # Manejador de errores unificados
main.py          # Configuración y levantamiento del servidor FastAPI
```

4. Funcionamiento General

En esta práctica:

- Los servicios usan JSON como fuente de datos.
- Cada servicio se ejecuta por separado en su propio puerto.
- Se implementaron:
 - rutas básicas,
 - filtros,
 - paginación,
 - validación de entradas y salidas,
 - manejo de excepciones

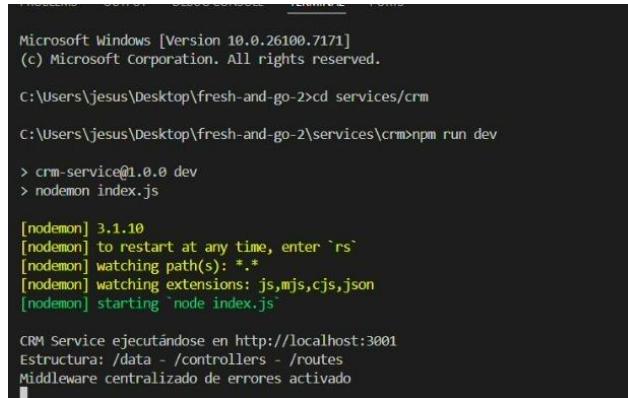
5. Problemas encontrados

- Algunos JSON devolvían demasiada información (ej. cliente → vehículo).
- Tuvimos que limpiar y normalizar varios campos.
- Se detectaron errores de validación que obligaron a ajustar schemas.
- En IoT, fue necesario instalar un paquete adicional para ejecutar FastAPI correctamente.
- En GitHub había muchos commits al inicio por desconocer la filosofía adecuada

5. Simulación: Carpeta raíz fresh-and-go-2

Nota: Esta práctica requiere la instalación de requirements.txt

En terminal



```
Microsoft Windows [Version 10.0.26100.7171]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jesus\Desktop\fresh-and-go-2>cd services/crm

C:\Users\jesus\Desktop\fresh-and-go-2\services\crm>npm run dev

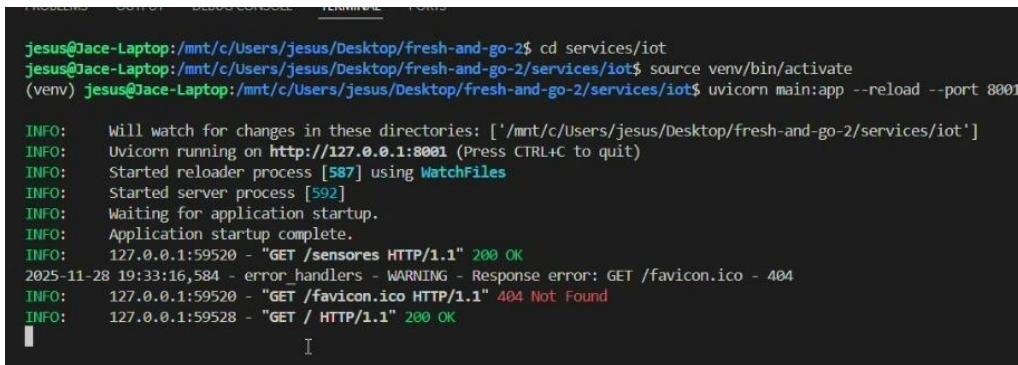
> crm-service@1.0.0 dev
> nodemon index.js

[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting node index.js

CRM Service ejecutándose en http://localhost:3001
Estructura: /data - /controllers - /routes
Middleware centralizado de errores activado
```

ILUSTRACIÓN 1: EJECUCIÓN DE CRM EN CMD

La guía QUICK_START.md nos facilitará los comandos necesarios para iniciar cada uno de los servicios. En el caso del CRM, después de movernos a la carpeta correspondiente (cd services/crm), ejecutamos el comando npm run dev, que arranca Nodemon y permite que el servicio se reinicie automáticamente ante cualquier cambio en los archivos relevantes (como se aprecia en la Ilustración 1: "watching path(s): ." y "starting node index.js"). Esto pone en marcha el CRM Service en el puerto http://localhost:3001, mostrando además la estructura de directorios principales (/data, /controllers, /routes) y la activación de su middleware centralizado para el manejo de errores. En este contexto, el servidor valida los esquemas AJV y se habilita el entorno ideal para desarrollar y testear funcionalidades del CRM, todo respaldado por los comandos sugeridos en la guía y visible en el output presentado en la terminal



```
PROBLEMS    CONSOLE    DEBUG CONSOLE    TERMINAL    PORTS

jesus@Jace-Laptop:/mnt/c/Users/jesus/Desktop/fresh-and-go-2$ cd services/iot
jesus@Jace-Laptop:/mnt/c/Users/jesus/Desktop/fresh-and-go-2/services/iot$ source venv/bin/activate
(venv) jesus@Jace-Laptop:/mnt/c/Users/jesus/Desktop/fresh-and-go-2/services/iot$ unicorn main:app --reload --port 8001

INFO:     Will watch for changes in these directories: ['/mnt/c/Users/jesus/Desktop/fresh-and-go-2/services/iot']
INFO:     unicorn running on http://127.0.0.1:8001 (Press CTRL+C to quit)
INFO:     Started reloader process [587] using WatchFiles
INFO:     Started server process [592]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     127.0.0.1:59520 - "GET /sensores HTTP/1.1" 200 OK
2025-11-28 19:33:16,584 - error_handlers - WARNING - Response error: GET /favicon.ico - 404
INFO:     127.0.0.1:59520 - "GET /favicon.ico HTTP/1.1" 404 Not Found
INFO:     127.0.0.1:59528 - "GET / HTTP/1.1" 200 OK
```

ILUSTRACIÓN 2: EJECUCIÓN IoT EN WSL

La guía QUICK_START.md también nos proporciona los pasos para la ejecución del servicio IoT, el cual se corre en el entorno WSL usando FASTAPI y Python. Después de acceder a la carpeta correspondiente (cd services/iot), activo el entorno virtual con source venv/bin/activate y lanza el servidor con el comando unicorn main:app --reload --port 8001, lo que inicia el servicio escuchando en el puerto http://127.0.0.1:8001. Como se aprecia en la Ilustración 2, se informa del inicio tanto del proceso recargador como del servidor, y la API queda lista para recibir peticiones (por ejemplo, /sensores), facilitando el desarrollo y prueba de endpoints IoT. Además, el sistema monitorea cambios en los archivos para recargar automáticamente, optimizando el flujo de trabajo y permitiendo que todas las funcionalidades y validaciones estén disponibles tal como indica la guía y se muestra en el terminal

Endpoints CRM

```

{
  "total": 8,
  "page": 1,
  "pageSize": 25,
  "data": [
    {
      "id": "PED0001",
      "clienteId": "CLI0001",
      "proveedorId": "PRO0001",
      "conductorId": "CON0001",
      "fechaPedido": "2024-10-25",
      "fechaEntrega": "2024-10-26",
      "productos": [
        {
          "id": "PRO0001",
          "nombre": "Tomates Cherry",
          "cantidad": 50,
          "precio": 2.5,
          "total": 125
        },
        {
          "id": "PRO0002",
          "nombre": "Lechugas Frescas",
          "cantidad": 30,
          "precio": 1.8,
          "total": 54
        }
      ],
      "estado": "En transito",
      "totalPedido": 179,
      "comentarios": "Entregar antes de las 12:00"
    },
    {
      "id": "PED0002",
      "clienteId": "CLI0002",
      "proveedorId": "PRO0002",
      "conductorId": "CON0002",
      "fechaPedido": "2024-10-24",
      "fechaEntrega": "2024-10-25",
      "productos": [
        {
          "id": "PRO0003",
          "nombre": "Leche Integral 1L",
          "cantidad": 100,
          "precio": 0.95,
          "total": 95
        },
        {
          "id": "PRO0004",
          "nombre": "Yogur Natural 125g",
          "cantidad": 50,
          "precio": 0.75,
          "total": 35
        }
      ]
    }
  ]
}

```

ILUSTRACIÓN 4: ENDPOINT PEDIDOS

```

{
  "data": [
    {
      "id": "COND001",
      "nombre": "Karla Martinez Ruiz",
      "email": "karla.martinez@freshgo.com",
      "telefono": "+34 600 111 222",
      "vehiculoAsignado": "VEH001",
      "licenciaConducir": "0012345678",
      "fechaExpiracionLicencia": "2026-05-15",
      "disponibilidad": true,
      "estado": "activo",
      "experiencia": 12,
      "calificacion": 4.8,
      "rutas": [
        "Sevilla Norte",
        "Alcala de Guadaira"
      ]
    },
    {
      "id": "COND002",
      "nombre": "Ana Rodriguez Lopez",
      "email": "ana.rodriguez@freshgo.com",
      "telefono": "+34 600 333 444",
      "vehiculoAsignado": "VEH002",
      "licenciaConducir": "0087654321",
      "fechaExpiracionLicencia": "2025-11-30",
      "disponibilidad": false,
      "estado": "activo",
      "experiencia": 8,
      "calificacion": 4.9,
      "rutas": [
        "Sevilla Centro",
        "Triana"
      ]
    },
    {
      "id": "COND003",
      "nombre": "Miguel Angel Torres",
      "email": "miguel.torres@freshgo.com",
      "telefono": "+34 600 555 666",
      "vehiculoAsignado": "VEH003",
      "licenciaConducir": "0011223344",
      "fechaExpiracionLicencia": "2027-03-20",
      "disponibilidad": true,
      "estado": "activo",
      "experiencia": 15,
      "calificacion": 5,
      "rutas": [
        "Sevilla Este",
        "Dos Hermanas",
        "Utrera"
      ]
    }
  ]
}

```

ILUSTRACIÓN 3: ENDPOINT CONDUCTORES

El endpoint /pedidos devuelve la colección de todos los pedidos realizados en el sistema, mostrando campos como el identificador del pedido, el cliente asociado, el proveedor, el conductor responsable, fechas de pedido y entrega, estado del pedido y una lista detallada de productos solicitados con cantidad, precio y totales. Además, incluye el campo de comentarios para instrucciones especiales y el total global del pedido, gestionando eficazmente el seguimiento y estado actual de cada envío (Ilustración 3)

Por último, el endpoint /conductores proporciona un listado de todos los conductores registrados en el CRM con detalles completos como nombre, email, teléfono, vehículo asignado, número y fecha de expiración de la licencia de conducir, disponibilidad, estado, experiencia, calificación y rutas habituales. Esta estructura facilita la asignación y seguimiento de conductores, asegurando la trazabilidad tanto de la documentación como del desempeño de cada profesional dentro de la plataforma (Ilustración 4)

```

{
  "total": 5,
  "page": 1,
  "pageSize": 25,
  "data": [
    {
      "id": "CLI001",
      "nombre": "Restaurante El Buen Sabor",
      "email": "pedidos@elbuensabor.com",
      "direccion": "Calle Mayor 45, 41001 Sevilla",
      "telefono": "+34 954 123 456"
    },
    {
      "id": "CLI002",
      "nombre": "Supermercado Fresh Market",
      "email": "compras@freshmarket.es",
      "direccion": "Avenida de la Constitución 23, 41002 Sevilla",
      "telefono": "+34 954 234 567"
    },
    {
      "id": "CLI003",
      "nombre": "Hotel Andalucía Palace",
      "email": "cocina@andaluciapalace.com",
      "direccion": "Plaza Nueva 8, 41003 Sevilla",
      "telefono": "+34 954 345 678"
    },
    {
      "id": "CLI004",
      "nombre": "Cafetería La Esquina",
      "email": "info@laesquina.es",
      "direccion": "Calle Sierpes 12, 41004 Sevilla",
      "telefono": "+34 954 456 789"
    },
    {
      "id": "CLI005",
      "nombre": "Panadería Los Arcos",
      "email": "ventas@losarcos.com",
      "direccion": "Plaza del Salvador 3, 41005 Sevilla",
      "telefono": "+34 954 567 890"
    }
  ]
}

```

ILUSTRACIÓN 6 ENDPOINT CLIENTES

```

{
  "data": [
    {
      "id": "PROV001",
      "nombre": "Frutas y Verduras Andaluzas S.L.",
      "contacto": "Juan García Martínez",
      "email": "ventas@frutassandaluzas.com",
      "telefono": "+34 955 111 222",
      "direccion": "Polígono Industrial Los Remedios, Sevilla",
      "cif": "A12345678",
      "productos": [
        "Frutas",
        "Verduras",
        "Mortalizas"
      ],
      "horarioEntrega": "06:00-22:00",
      "diasEntrega": [
        "Lunes",
        "Martes",
        "Miércoles",
        "Jueves",
        "Viernes",
        "Sábado"
      ]
    },
    {
      "id": "PROV002",
      "nombre": "Lácteos del Sur",
      "contacto": "María López Fernández",
      "email": "pedido@lacteosdelsur.es",
      "telefono": "+34 955 222 333",
      "direccion": "Calle Industrial 15, Dos Hermanas",
      "cif": "B12345678",
      "categorias": "Lácteos y derivados",
      "productos": [
        "Leche",
        "Yogur",
        "Queso",
        "Mantequilla"
      ],
      "horarioEntrega": "07:00-20:00",
      "diasEntrega": [
        "Lunes",
        "Miércoles",
        "Viernes"
      ]
    }
  ]
}

```

ILUSTRACIÓN 5: ENDPOINT PROVEEDORES

El endpoint /clientes del CRM muestra un listado completo de los clientes registrados, en este caso un total de cinco. Para cada cliente se presentan datos esenciales como el nombre, dirección, teléfono y email, junto a un identificador único. Por ejemplo, aparecen el “Restaurante El Buen Sabor”, el “Supermercado Fresh Market” y otros negocios de Sevilla, lo que facilita la gestión y consulta inmediata de la información básica de cada uno directamente desde la API (Ilustración 5)

Al consultar /proveedores, el CRM despliega la información de los proveedores, mostrando detalles como nombre de la empresa, persona de contacto, email, teléfono y dirección, además del CIF, categoría de productos y el listado de productos concretos que ofrece cada proveedor. También se incluye el rango de horario de entrega y los días específicos en los que realiza entregas, permitiendo así un control claro sobre la programación logística y la relación comercial con cada proveedor (Ilustración 6)

Endpoints IoT

```

{
  "data": [
    {
      "id": "VEH001",
      "matricula": "7234BXJ",
      "capacidadKg": 2500,
      "gps": "37.3887,-5.9920",
      "temperatura": 4.5,
      "valor": "Activo"
    },
    {
      "id": "VEH002",
      "matricula": "4567XYZ",
      "capacidadKg": 3000,
      "gps": "37.3920,-5.9850",
      "temperatura": 4.2,
      "valor": "Activo"
    },
    {
      "id": "VEH003",
      "matricula": "8901DEF",
      "capacidadKg": 2800,
      "gps": "37.3950,-5.9900",
      "temperatura": 4.8,
      "valor": "Activo"
    },
    {
      "id": "VEH004",
      "matricula": "2345GHI",
      "capacidadKg": 2200,
      "gps": "37.3880,-5.9880",
      "temperatura": 5.1,
      "valor": "Activo"
    }
  ]
}
  
```

ILUSTRACIÓN 7: ENDPOINT VEHICULOS

```

{
  "data": [
    {
      "id": "LECT001",
      "sensorId": "SENS001",
      "ubicacionId": "VEH001",
      "timestamp": "2025-11-22T08:00:00Z",
      "temperatura": -19.5,
      "gps": {
        "latitud": 37.3886,
        "longitud": -5.9845,
        "altitud": 12
      },
      "estado": "normal",
      "alertaActiva": false,
      "tiempoFueraRango": 0,
      "cadenRota": false
    },
    {
      "id": "LECT002",
      "sensorId": "SENS001",
      "ubicacionId": "VEH001",
      "timestamp": "2025-11-22T08:05:00Z",
      "temperatura": -18.2,
      "gps": {
        "latitud": 37.389,
        "longitud": -5.985,
        "altitud": 13
      },
      "estado": "normal",
      "alertaActiva": false,
      "tiempoFueraRango": 0,
      "cadenRota": false
    },
    {
      "id": "LECT003",
      "sensorId": "SENS001",
      "ubicacionId": "VEH001",
      "timestamp": "2025-11-22T08:10:00Z",
      "temperatura": -14.8,
      "gps": {
        "latitud": 37.3895,
        "longitud": -5.9855,
        "altitud": 14
      },
      "estado": "normal",
      "alertaActiva": false,
      "tiempoFueraRango": 0,
      "cadenRota": false
    }
  ]
}
  
```

ILUSTRACIÓN 8: ENDPOINT LECTURAS

El endpoint /vehiculos del sistema IoT muestra un listado con la información detallada de todos los vehículos conectados. Para cada uno se especifica el identificador, matrícula, capacidad en kilogramos, posición GPS (latitud y longitud), temperatura y estado actual. Por ejemplo, vemos vehículos como "VEH001", "VEH002" y otros, donde cada uno tiene asociados valores propios de localización y condiciones operativas, lo que permite controlar y supervisar en tiempo real aspectos logísticos críticos para la flota (Ilustración 7)

El endpoint /lecturas proporciona las lecturas tomadas por los sensores instalados en los vehículos, incluyendo datos como el identificador de la lectura, el sensor que la tomó, el vehículo en que se encuentran, la fecha y hora exacta (timestamp), temperatura registrada, estado y valores GPS (latitud, longitud, altitud). También indica si hay alerta activa, si la lectura está fuera de rango y otros parámetros operativos, lo que facilita el monitoreo preciso de las condiciones en las que viajan los productos y la respuesta ante cualquier incidencia detectada por los sensores (Ilustración 8).

```

{
  "data": [
    {
      "id": "SENS001",
      "nombre": "Sensor Vehiculo 1 - Zona Congelados",
      "ubicacionId": "VEH001",
      "tipoAlimento": "congelado",
      "rangoMin": -22,
      "rangoMax": -18,
      "umbralAlerta": -15,
      "umbralCritico": -12,
      "intervaloLectura": 300
    },
    {
      "id": "SENS002",
      "nombre": "Sensor Vehiculo 1 - Zona Refrigerados",
      "ubicacionId": "VEH001",
      "tipoAlimento": "refrigerado",
      "rangoMin": 0,
      "rangoMax": 4,
      "umbralAlerta": 4,
      "umbralCritico": 7,
      "intervaloLectura": 300
    },
    {
      "id": "SENS003",
      "nombre": "Sensor Vehiculo 1 - Zona Delicados",
      "ubicacionId": "VEH001",
      "tipoAlimento": "delicado",
      "rangoMin": 15,
      "rangoMax": 25,
      "umbralAlerta": 28,
      "umbralCritico": 32,
      "intervaloLectura": 600
    },
    {
      "id": "SENS004",
      "nombre": "Sensor Vehiculo 2 - Zona Congelados",
      "ubicacionId": "VEH002",
      "tipoAlimento": "congelado",
      "rangoMin": -22,
      "rangoMax": -18,
      "umbralAlerta": -15,
      "umbralCritico": -17
    }
  ]
}

```

ILUSTRACIÓN 10: ENDPOINT SENsoRES

```

{
  "id": "SENS003",
  "nombre": "Sensor Vehiculo 1 - Zona Delicados",
  "ubicacionId": "VEH001",
  "tipoAlimento": "delicado",
  "rangoMin": 15,
  "rangoMax": 25,
  "umbralAlerta": 28,
  "umbralCritico": 32,
  "intervaloLectura": 600
}

```

ILUSTRACIÓN 9: ENDPOINT ID SENsoRES

Desde /sensores se obtiene toda la información relativa a los sensores desplegados en los vehículos IoT. Para cada sensor se muestran campos como el identificador del sensor, nombre descriptivo, vehículo al que está asignado, tipo de alimento monitoreado (congelado, refrigerado, delicado), rangos mínimos y máximos de operación, umbrales de alerta y críticos, e intervalo entre lecturas. Así, por ejemplo, el sistema puede diferenciar sensores para zonas congeladas, refrigeradas o delicadas, y configurar alertas automáticas para garantizar la correcta conservación de los productos transportados (Ilustración 9)

Por último, el endpoint /sensores/SENS003 sirve para consultar la información específica de un sensor determinado, en este caso el sensor "SENS003" instalado en el vehículo "VEH001" para la zona de alimentos delicados. Aquí se muestran todos los parámetros relevantes como tipo de alimento, rangos y umbrales configurados, así como el intervalo de actualización de las lecturas. Este nivel de detalle permite la gestión y verificación de cada sensor individual, asegurando el cumplimiento de los requisitos de conservación y el control exhaustivo de las variables críticas durante el transporte (Ilustración 10)



FRESH&GO

DE LA TIENDA A TU MESA EN UN CLICK