



E2.2. Capa para la ingesta de datos de fuentes heterogéneas



VICEPRESIDENCIA
TERCERA DEL GOBIERNO
MINISTERIO
PARA LA TRANSICIÓN ECOLÓGICA
Y EL RETO DEMOGRÁFICO



Fundación Biodiversidad



Plan de Recuperación,
Transformación
y Resiliencia



Financiado por
la Unión Europea
NextGenerationEU

Contenido

E2.2. Capa para la ingesta de datos de fuentes heterogéneas	1
1 Introducción	3
1.1 Objetivo	3
1.2 Alcance.....	3
2 Conectores para ingestión de datos de fuentes externas y otros sistemas de detección3	
2.1 Diseño e Integración del Modelo de Datos de Información en IA4Birds	4
2.2 Modelo de Datos de Ingesta	5
2.2.1 Datos Externos (Ciencia Ciudadana y Contexto)	6
2.2.2 Ingesta desde Dispositivos (IoT)	6
2.2.3 Análisis y Agregación	7
2.2.4 Flujo de datos	7
2.3 Modelo de Datos de la Plataforma	7
2.3.1 Gestión de Usuarios y Entidades	8
2.3.2 Datos Biológicos y Observacionales	8
2.3.3 Relaciones y Cohesión del Modelo.....	9
2.3.4 Flujo de datos y operación (resumen).....	9
3 Fusión de información de fuentes heterogéneas en una base de conocimiento	9
3.1 Investigación de Tecnologías de Base de Datos.....	10
3.1.1 Arquitectura en la nube y microservicios	10
3.2 Diseño e implementación de interfaces responsivas.....	11
3.3 Estudio de diversas herramientas de software para la visualización de datos.....	11
3.3.1 OPENAPI/Swagger (interfaz documental/operativa)	11
3.3.2 API de Ingesta (ai4birds-ingest-service)	12
3.4 Diseño de técnicas de caché de visualizaciones para reducir consultas al Big Data y mejorar el rendimiento al acceder a la plataforma.	13
4 Conclusión.....	15

1 Introducción

1.1 Objetivo

Este documento corresponde al entregable:

E2.2 - Capa común de Inteligencia Artificial y técnicas de visualización.

Refleja los trabajos realizados y los resultados alcanzados durante la ejecución de las actividades:

A2.2. Conectores para la ingestión de datos de fuentes externas y otros sistemas de detección, cuyo objetivo es diseñar una herramienta de dispositivos IoT de bajo consumo y reducido tamaño que permitan monitorizar de forma acústica y visual las aves y su actividad en tiempo real.

Esta tarea está encuadrada dentro de la actividad **A2: Capa para la ingestión de datos de aves: red IoT, fuentes Big data y datos abiertos.** Cuyo objetivo es diseñar una plataforma que permita la recogida masiva de datos de aves desde fuentes heterogéneas e implementar una herramienta de sensores IoT para la monitorización acústica y visual de aves de cara a fusionar toda la información en una base única de conocimiento.

1.2 Alcance

Este documento se encuentra en la versión 1.0, la investigación aquí plasmada trata de presentar el trabajo realizado para la consecución de los objetivos del proyecto de diseñar una plataforma de IA explicable para optimizar la evaluación de espacios destinados a ser parques eólicos.

2 Conectores para ingestión de datos de fuentes externas y otros sistemas de detección

Este apartado presenta el diseño de los conectores que permiten integrar, de forma segura y trazable, los tres orígenes clave de IA4Birds (sensores IoT (cámaras/micrófonos), ciencia ciudadana (Xeno-canto y eBird) y contexto territorial (Mapa Eólico Ibérico e IDECYL) para alimentar de manera homogénea la plataforma y sus visualizaciones. Estos conectores se exponen como endpoints REST documentados en OpenAPI/Swagger, versionados y sometidos a rate-limiting; se encapsulan en el ai4birds-ingest-service (único punto de salida hacia terceros), que además se suscribe a Mosquito para recibir los eventos del Edge y programa tareas periódicas con Celery para sincronizar fuentes externas. La persistencia se organiza en una capa PostgreSQL por dominios (configuración, ingesta operativa y análisis), mientras la API de ingesta sirve recursos filtrables (segmentos, estadísticas, heatmaps, resultados combinados de ciencia ciudadana y capas de viento/exclusión) listos para el consumo por backend y frontend. La arquitectura mantiene una separación explícita de responsabilidades —Edge detecta y publica; Ingesta valida, persiste y sirve; Coordinación asegura; BFF compone; Frontend presenta—, reforzada por controles de seguridad (TLS, JWT, autorización por rol) y aislamiento de red. Para las fuentes de vídeo, MediaMTX convierte RTSP a HLS y expone un API para alta dinámica de orígenes, garantizando compatibilidad web y escalabilidad.

2.1 Diseño e Integración del Modelo de Datos de Información en IA4Birds

A continuación se detalla y amplía el alcance de la investigación y diseño de la fusión de información en IA4Birds, orientada a integrar grandes volúmenes de datos heterogéneos con garantías de calidad, trazabilidad y rendimiento.

1) Datos IoT (Visual/Acústico)

- Naturaleza y cadencia: flujos de alta frecuencia procedentes del Edge (ventanas de frames y espectrogramas, conteos por clase, telemetría de dispositivo). Se trabaja con *event time* (marca de tiempo del sensor) y *processing time* (ingesta), gestionando *jitter*, datos fuera de orden y pérdidas intermitentes.
- Normalización de metadatos: todo evento llega con *device_id*, *camera_id*, *segment_idx*, georreferenciación/actitud del sistema (colatitud, azimut, zoom/FOV), y cronología (*received_at*, reloj del dispositivo y offset de sincronía).
- Preprocesamiento en borde: compresión/agrupación por ventana, filtrado de ruido, *tracking* ligero y umbrales de confianza para reducir ancho de banda.
- Validación en ingesta: esquemas rígidos y validaciones (idempotencia por clave compuesta, duplicados, rangos físicos de FOV/ángulos, tamaños de *payload*), con colas y *backpressure* para absorber picos.
- Persistencia: separación entre descriptores en BD (p.ej., *segment_data*, *device_status*) y artefactos binarios (imágenes/recortes, audio) en *object storage* con URLs firmadas. Índices temporales y espaciales (PostGIS) para consultas subsegundo por área/intervalo.

2) Datos Externos (Ciencia Ciudadana y Contexto)

- Fuentes y acceso: conectores a eBird/Xeno-canto (observaciones y grabaciones), y a capas territoriales (mapas de sensibilidad, IDECYL, Mapa Eólico) vía APIs/servicios OGC (WMS/WFS/WMTS).
- Armonización semántica: catálogo maestro de especies (sinónimos, nombre común/científico, taxonomía, *authority*), normalización de unidades/formatos y *country/region codes*.
- Calidad y deduplicación: *fingerprints* espacio-temporales, umbrales de proximidad (espacial/temporal/especie), *rate limiting* y cacheado para evitar sobrecarga y re-ingesta innecesaria.
- Geo-fusión: *spatial join* de observaciones con cuadrículas/zonas de sensibilidad, buffers de paso migratorio y capas de viento/exclusión para enriquecer consultas analíticas y mapas.

3) Datos del Modelo (Predicciones y Estadísticas)

- Trazabilidad de inferencias: cada predicción conserva versión de modelo, *checkpoint*, hiperparámetros relevantes y *confidence/uncertainty* (p.ej., entropía, intervalos bayesianos).
- Agregados operativos: acumulados por especie/cámara/periodo (*bird_statistics*), detecciones recientes (*last seen*), *rollups* horarios/diarios y *tiles* de calor (*heatmap_image*) listos para visualización.
- Curación y feedback: almacenamiento de falsos positivos/negativos etiquetados, muestreo estratificado para *retraining* y *drift detection* (cambios en distribución de clases/ruido).

Estrategias de fusión multimodal

- Alineación temporal-espacial: resamplio a ventanas comunes, *watermarks* para cerrar ventanas con datos tardíos, corrección de *clock drift* y proyección de FOV a cielo/territorio (sistemas esféricos y UTM).
- Fusión temprana/tardía/híbrida:
 - *Temprana*: combinación de *features* (p.ej., embeddings de audio + descriptores visuales) en modelos conjuntos.
 - *Tardía*: combinación de salidas (votación ponderada por calidad de fuente y calibración de probabilidades).
 - *Híbrida*: verificación cruzada (audio confirma visión en umbrales bajos; visión desambigua especies acústicamente próximas).

Incertidumbre y conflicto: propagación de *uncertainty* desde sensores a agregados; resolución por reglas (prioridad de precisión/recencia), puntuaciones de calidad de fuente y *Bayesian model averaging*.

2.2 Modelo de Datos de Ingesta

El modelo de datos propuesto actúa como punto central de integración entre las distintas fuentes de información del sistema, permitiendo almacenar, relacionar y transformar tanto datos externos como registros generados por dispositivos de campo y procesos analíticos. Su estructura se organiza en tres bloques funcionales principales.

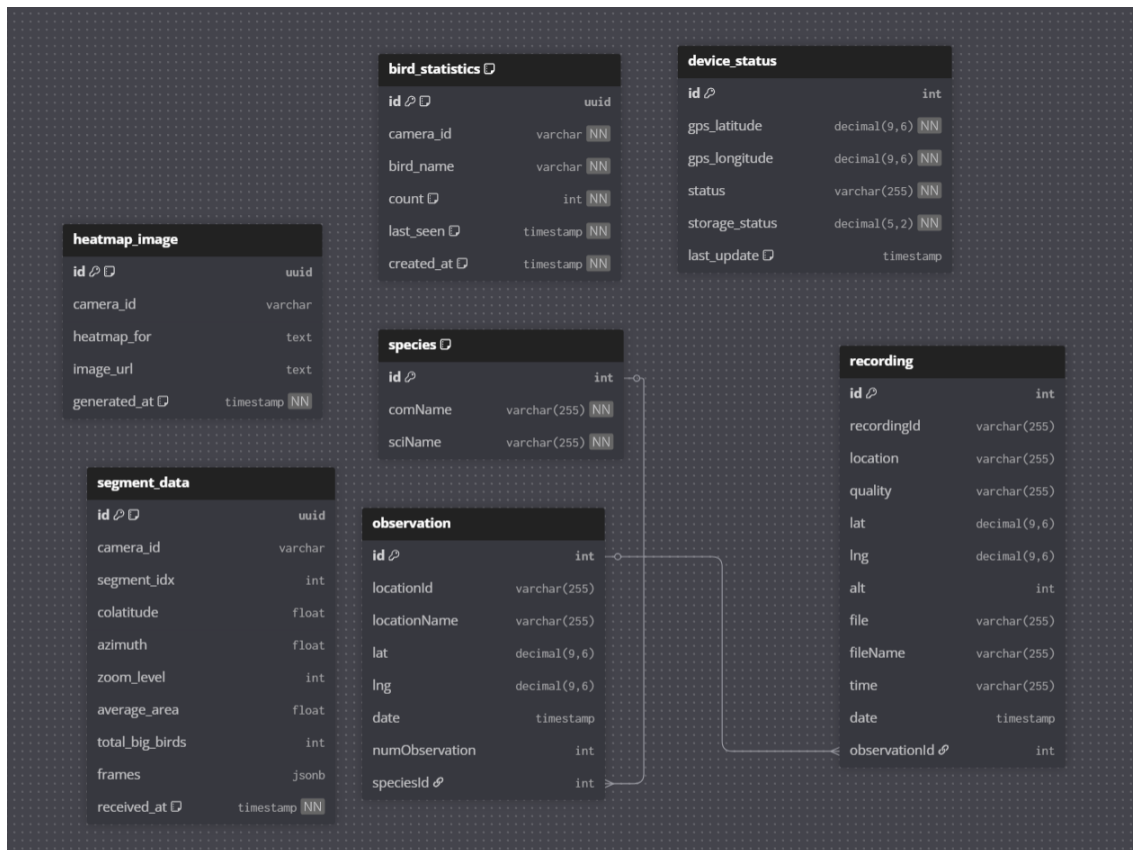


Fig.1. Modelos de datos de ingesta

2.2.1 Datos Externos (Ciencia Ciudadana y Contexto)

Este bloque agrupa las entidades que representan la información proveniente de fuentes externas, como observaciones y grabaciones de fauna.

- **species:** Tabla maestra que define las especies y unifica los nombres científicos y comunes, sirviendo como referencia central para el resto del modelo. (id, comName, sciName) normaliza el catálogo.
- **observation:** Registra avistamientos y observaciones georreferenciadas, vinculadas a una especie. Incluye datos como la localización, la fecha y la cantidad observada. lat, lng, date, numObservation, speciesId representa observaciones (p.ej., eBird) enlazadas a species.
- **recording:** Contiene los metadatos de las grabaciones asociadas a las observaciones, incluyendo información sobre ubicación, calidad y archivo. lat, lng, alt, file, fileName, date, time, observationId) recoge grabaciones (p.ej., Xeno-canto) ligadas a cada observación.

2.2.2 Ingesta desde Dispositivos (IoT)

Este bloque gestiona los datos generados por dispositivos distribuidos, como cámaras o unidades de procesamiento en el borde (*Edge*), que publican información en tiempo real, el edge detecta y publica resultados ligeros por MQTT, un servicio de ingesta los valida y persiste en PostgreSQL para explotar después.

- **device_status:** Registra la telemetría y el estado de los dispositivos, incluyendo su posición, estado operativo y nivel de almacenamiento.
- **segment_data:** Almacena los datos visuales detectados en segmentos de vídeo, junto con sus metadatos (cámara, orientación, zoom) y la información cruda de detección en formato estructurado.
(*id, camera_id, segment_idx, colatitude, azimuth, zoom_level, average_area, total_big_birds, frames(jsonb), received_at*) almacena los “segmentos” que publica el edge (detecciones por ventana temporal). Los campos de colatitud/azimut/zoom son los que permiten reconstruir el mapeado celeste y los barridos PAN/TILT/HFOV–VFOV, base para generar heatmap
- **heatmap_image:** Guarda los resultados derivados del procesamiento visual, como los mapas de calor generados, junto con su referencia a la cámara y la ubicación del archivo resultante.

2.2.3 Análisis y Agregación

Este bloque se orienta a la consolidación y síntesis de información, transformando los datos operativos en conocimiento analítico.

- **bird_statistics:** Representa una capa de agregación de datos, generada a partir de las detecciones individuales. Permite mantener conteos acumulados y fechas de última detección por especie y cámara, optimizando las consultas analíticas y de visualización. (*camera_id, bird_name, count, last_seen, created_at*) es el agregado por especie/cámara y facilita paneles temporales.
- **heatmap_image:** (*camera_id, heatmap_for, image_url, generated_at*) es el descriptor publicable del heatmap generado a partir de *segment_data* (la imagen se sirve en el frontend con *bounds/URL*).

2.2.4 Flujo de datos

1. **Detección en Edge (visión y audio):** publica en MQTT resultados (detecciones, conteos, descriptores).
2. **Ingesta en la nube:** persiste en *segment_data* y *device_status*, consulta Xeno-canto/eBird y guarda en *observation/recording/species*.
3. **Analítica:** jobs agregan a *bird_statistics* y generan artefactos *heatmap_image* que el frontend muestra.

Este patrón (Edge detecta y publica; Ingesta valida/persiste; Coordinación asegura; Frontend presenta) es el corazón de la versión alpha y del caso de uso controlado.

2.3 Modelo de Datos de la Plataforma

El modelo de datos de la plataforma está diseñado para gestionar la información estructural y operativa del sistema, integrando usuarios, cámaras, especies y registros de observación. Su arquitectura se orienta a garantizar la trazabilidad de los datos, la integridad de las relaciones y la interoperabilidad entre los distintos módulos funcionales.



Fig.2. Modelo de datos de la plataforma

2.3.1 Gestión de Usuarios y Entidades

Este bloque define la estructura de los usuarios y su relación con los recursos que gestionan dentro de la plataforma.

- **users:** Contiene la información principal de los usuarios registrados, incluyendo datos de contacto, organización, ocupación y tipo de entidad. Incorpora un sistema de activación y control de estado.
- **cameras:** Registra las cámaras asociadas a cada usuario, incluyendo el tipo de fuente (RTSP, RTMP, HLS, YouTube), las URL de origen y reproducción, la ubicación, los metadatos asociados y el estado operativo. El campo *is_public* permite distinguir entre cámaras públicas y privadas.

2.3.2 Datos Biológicos y Observacionales

Este bloque representa la información relacionada con las especies y los registros de observación y grabación, tanto internos como integrados desde fuentes externas.

- **species:** Tabla maestra que almacena los nombres comunes y científicos de las especies, sirviendo como referencia para observaciones y grabaciones.

- **observation:** Registra observaciones georreferenciadas, vinculadas a una especie concreta. Incluye información de localización, fecha y número de individuos observados.
- **recording:** Almacena los metadatos de las grabaciones asociadas a las observaciones, incluyendo la ubicación, calidad, fecha y archivo correspondiente.

2.3.3 Relaciones y Cohesión del Modelo

El modelo mantiene la consistencia referencial mediante relaciones definidas entre las distintas entidades:

- Cada **observation** se asocia a una **species**, permitiendo la clasificación y el análisis taxonómico.
- Cada **recording** se vincula a una **observation**, estableciendo una relación directa entre los registros de campo y los datos audiovisuales.
- Cada **camera** pertenece a un **user**, lo que asegura la trazabilidad y control de acceso a los dispositivos registrados.

2.3.4 Flujo de datos y operación (resumen)

- Edge → nube por MQTT; Ingest suscrita a Mosquitto recibe detecciones/estado y lanza tareas periódicas (Celery/Redis).
- Streaming: MediaMTX RTSP→HLS para visualización web.
- Paneles y administración: visor de mapa, analíticas de cámaras, blog, panel admin para validar conectores y supervisar datos crudos.

3 Fusión de información de fuentes heterogéneas en una base de conocimiento

La fusión de información en IA4Birds articula, de extremo a extremo, la integración de tres corrientes principales —IoT (visual/acústico), fuentes externas (ciencia ciudadana y capas territoriales) y salidas del modelo (predicciones/estadísticas)— para convertir datos masivos y heterogéneos en conocimiento operativo y reproducible. Este capítulo describe cómo esa fusión se sostiene sobre: (i) una base de datos relacional robusta (PostgreSQL) capaz de combinar relaciones complejas con datos semiestructurados; (ii) una arquitectura cloud de microservicios (ingesta, coordinación, backend y streaming) que asegura seguridad, trazabilidad y escalabilidad; (iii) interfaces responsivas y herramientas de visualización interactivas que exponen mapas, dashboards y APIs documentadas (OpenAPI/Swagger); y (iv) un conjunto de técnicas de cacheo y pre-cómputo que reducen la latencia y evitan accesos costosos a datos brutos. En conjunto, estas piezas alinean temporal y espacialmente los eventos, armonizan la semántica (taxonomía de especies y metadatos de dispositivo), gestionan la incertidumbre de detección y proporcionan vistas agregadas listas para la toma de decisiones y la planificación óptica con criterios de conservación.

3.1 Investigación de Tecnologías de Base de Datos

La fase de investigación evaluó las tecnologías NoSQL (MongoDB, Cassandra, etc.) frente a las bases de datos relacionales tradicionales (SQL).

Se decidió utilizar PostgreSQL por las siguientes razones:

1. **Naturaleza Relacional de los Datos:** La información de IA4Birds, aunque diversa, es fundamentalmente relacional. Las detecciones (`segment_data`) pertenecen a una cámara. Las observaciones (`observation`) se refieren a una especie. Las estadísticas (`bird_statistics`) agregan datos de una cámara para un tipo de ave en concreto. La capacidad de realizar consultas complejas entre estas entidades es crucial para el análisis y la visualización.
2. **Integridad y Transacciones (ACID):** Para un sistema que genera estadísticas y datos analíticos, la consistencia de los datos es primordial. PostgreSQL garantiza transacciones ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad), asegurando que las estadísticas y los registros de detección sean fiables.
3. **Potencia Híbrida:** PostgreSQL, en su versión moderna, integra de manera equilibrada las ventajas de los enfoques relacional y semiestructurado, ofreciendo una solución versátil y eficiente para la gestión de datos. Esta capacidad resulta especialmente valiosa para manejar información masiva y heterogénea proveniente del *Edge*, como los *frames* de detección, ya que permite combinar la estructura y consistencia de un esquema relacional con la flexibilidad necesaria para almacenar y consultar datos complejos de forma ágil y escalable.
4. **Madurez y Ecosistema:** PostgreSQL es un sistema de gestión de bases de datos extremadamente robusto, maduro y con un amplio soporte de herramientas para análisis y administración.

En resumen, se seleccionó PostgreSQL por su capacidad para combinar la flexibilidad de los datos semiestructurados propios del entorno IoT con la potencia relacional y la integridad necesarias para el resto de la plataforma.

3.1.1 Arquitectura en la nube y microservicios

La plataforma se despliega como microservicios contenerizados coordinados por redes internas. Nginx actúa como puerta de entrada, termina TLS y enruta al frontend (React, construida con Vite) y al backend (Node.js). El backend autentica, autoriza por rol y compone respuestas agregando datos de servicios internos.

El `ai4birds-coordinate-service` (Flask) aplica control de acceso: valida JWT, registra acceso y reenvía llamadas autorizadas al `ai4birds-ingest-service`.

El `ingest-service` escucha MQTT para recepcionar detecciones y heatmaps, persiste en la base operativa y ejecuta tareas periódicas con Celery para consultar Xeno-canto y eBird, obtener el Mapa Eólico Ibérico para coordenadas y cargar capas de IDECYL. La API está documentada con Flask-RestX/Swagger.

El almacenamiento se estructura en tres instancias de PostgreSQL: configuración (usuarios, cámaras, roles), ingesta operativa (segmentos, estadísticas, observaciones, referencias de

heatmap) y análisis (vistas materializadas y agregados temporoespaciales). Redis hace de broker de Celery y de caché de resultados. MediaMTX recibe RTSP y publica HLS; su API permite dar de alta orígenes sin tocar componentes adyacentes.

3.2 Diseño e implementación de interfaces responsivas

Para alcanzar los objetivos de la plataforma IA4Birds y garantizar una interacción eficiente entre los usuarios y los datos generados por el sistema, se ha desarrollado un conjunto de

El mapa es completamente interactivo: el usuario puede desplazarse y hacer zoom (funciones pan/zoom), consultar parámetros eólicos y visualizar en tiempo real las celdas recomendadas para la instalación de turbinas. Al seleccionar una celda, se despliega un popup informativo con datos específicos como la velocidad media del viento, cobertura de exclusión, especies observadas, viabilidad del aerogenerador (WTG) y porcentaje de área afectada.

Esta interfaz cumple la función de integrar información ambiental, climática y biológica en un entorno visual unificado, permitiendo la toma de decisiones informada en el contexto de la planificación eólica y la conservación de aves.

3.3 Estudio de diversas herramientas de software para la visualización de datos

Con el fin de garantizar la eficiencia, interoperabilidad y calidad visual de los resultados mostrados en la plataforma IA4Birds, se ha realizado una investigación exhaustiva de herramientas de software y entornos de desarrollo orientados a la visualización, documentación y explotación de datos procedentes de las distintas capas del sistema.

Este estudio tuvo como objetivo identificar las soluciones tecnológicas más adecuadas para representar y compartir grandes volúmenes de información ambiental y biológica de forma dinámica, segura y estandarizada. Se evaluaron plataformas de desarrollo, librerías de visualización, frameworks de documentación de APIs y sistemas de gestión de datos geoespaciales, valorando aspectos como la escalabilidad, la modularidad, la facilidad de integración y la adopción de estándares abiertos.

Entre los criterios de selección se consideraron especialmente la compatibilidad con arquitecturas de microservicios, el soporte para formatos JSON y GeoJSON, la capacidad de generar visualizaciones interactivas de tipo dashboard o mapa, y la posibilidad de automatizar la documentación de los endpoints y la validación de las peticiones API.

Como resultado de esta investigación, se definió una infraestructura tecnológica de referencia para IA4Birds basada en herramientas de documentación activa de APIs (OpenAPI/Swagger), servicios RESTful y frameworks geoespaciales y analíticos capaces de manejar datos tanto en tiempo real como históricos. Esta combinación asegura la trazabilidad de las fuentes, la transparencia de los procesos y la facilidad de integración con los módulos de análisis y predicción basados en inteligencia artificial desarrollados en las fases anteriores del proyecto.

3.3.1 OPENAPI/Swagger (interfaz documental/operativa)

Naturaleza: Es la especificación estándar (formalmente OAS 3) para describir y definir APIs REST. Swagger es el conjunto de herramientas más popular que implementa esta especificación, incluyendo la interfaz de usuario (Swagger UI).

Función

- Documentación viva: Documenta automáticamente todos los endpoints de las APIs de Ingesta y Coordinación.
- Interfaz operativa: Facilita pruebas interactivas directamente desde el navegador, permitiendo a los desarrolladores validar la lógica de la API.
- Herramienta de desarrollo: Permite la validación automática de peticiones y la generación de código cliente en diferentes lenguajes.

Características

- Define formalmente los esquemas de petición/respuesta (qué datos enviar y esperar).
- Documenta los esquemas de seguridad (ej. cómo usar el JWT).
- Especifica todos los códigos de estado posibles (200, 401, 404, etc.) y provee ejemplos.

3.3.2 API de Ingesta (ai4birds-ingest-service)

En el stack de la nube, MediaMTX convierte el RTSP a HLS para el consumo web. El servicio ai4birds-ingest-service (Flask) recibe los mensajes MQTT y sirve la API REST principal. La seguridad se delega en ai4birds-coordinate-service (Flask), que actúa como pasarela validando tokens JWT. Finalmente, un backend en Node.js y TypeScript orquesta la lógica de aplicación y la autenticación, mientras que un frontend en React (Vite) presenta los datos al usuario.

La API de Ingesta es el motor de datos de la plataforma; es la que se comunica con la base de datos y con los conectores de terceros. Sus endpoints están organizados por espacios de nombres (namespaces) para agrupar funcionalidades:

- ns_xenocanto (GET /xenocanto): Consulta y recupera registros de grabaciones de aves de la base de datos de XenoCanto[7].
- ns_ebird (GET /ebird): Consulta y recupera registros de avistamientos de aves de la base de datos de eBird[8].
- ns_dataBird (GET /dataBird): Expone un endpoint que combina los resultados de XenoCanto y eBird en una única respuesta.
- ns_windmap (POST /windmap): Recibe coordenadas (lat, lon, z) y devuelve datos del Mapa Eólico Ibérico, como el perfil de viento, la rosa de vientos y datos Weibull.
- ns_exclusionmap: Ofrece varios endpoints para consumir los datos de exclusión eólica de IDECYL:
 - o POST /: Devuelve los datos en un formato JSON paginado.
 - o GET /all: Devuelve el conjunto de datos completo en un solo JSON.
 - o GET /zip: Proporciona un fichero .zip descargable con los datos.
 - o POST /stream-exclusion-data: Permite el streaming de datos por lotes.
- ns_sensitivity (GET /sensitivity): Recupera la información de las capas de sensibilidad de aves en Castilla y León.

Datos Operativos (Ingesta desde el Edge):

- ns_device_status: Gestiona el estado de los dispositivos Edge.
 - o POST /: Permite al Edge reportar un nuevo estado (almacenamiento, GPS, etc.).
 - o GET /latest: Obtiene el último estado reportado por un dispositivo.

- GET /health: Comprueba el estado de salud del sistema.
- ns_segment_data (GET /segment-data): Es uno de los endpoints principales; recupera los datos de detecciones de aves (segmentos), permitiendo filtrar por camera_id.
- ns_heatmap_data (GET /heatmap-data): Recupera los metadatos del mapa de calor (heatmap) más reciente generado para una camera_id específica.
- ns_bird_statistics: Proporciona estadísticas agregadas:
 - GET /: Devuelve estadísticas filtradas por camera_id y por bird_name (especie).
 - GET /by-camera: Devuelve todas las estadísticas agrupadas por una camera_id.
- ns_heatmap_files: Gestiona el acceso a los ficheros de imagen de los heatmaps.
 - GET /files: Lista todos los ficheros de heatmap disponibles.
 - GET /download/<filename>: Permite la descarga directa de un fichero de imagen de heatmap específico (ej. un .png).

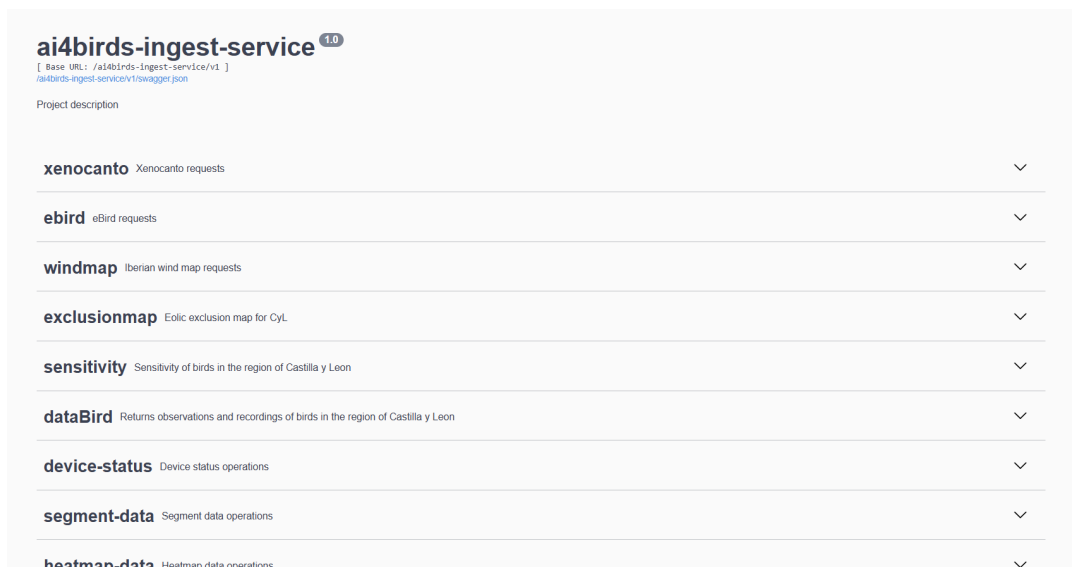


Fig.3. Servicio de ingesta

Para emplear estos endpoints desde la interfaz de Swagger, solo es necesario seleccionar el que se desea usar y presionar “Try it out”, seguido de “Execute”. Dentro de esta visual se encuentran ejemplos de uso y las posibles respuestas. El usuario puede modificar el valor de los “Parameters” (como camera_id o bird_name) para personalizar la petición.

3.4 Diseño de técnicas de caché de visualizaciones para reducir consultas al Big Data y mejorar el rendimiento al acceder a la plataforma.

El Ingest Service funciona como el núcleo de recepción y procesamiento de datos. Este componente escucha en tiempo real mensajes a través del protocolo MQTT, lo que le permite integrar de forma continua los flujos de datos provenientes de los sensores desplegados —como cámaras IoT, micrófonos y dispositivos de detección ambiental—. A partir de esta información, el servicio gestiona la persistencia de detecciones, metadatos y heatmaps en la base de datos

operativa. Además, mediante la integración del sistema de tareas distribuidas Celery, el servicio ejecuta procesos periódicos automatizados que incluyen:

- La consulta y sincronización con fuentes externas de ciencia ciudadana como XenoCanto y eBird, que aportan observaciones acústicas y visuales de aves en la región de estudio.
- La obtención dinámica del Mapa Eólico Ibérico, con el fin de asociar datos de recurso eólico (velocidad y dirección del viento) a las coordenadas geográficas de cada observación.
- La actualización automática de capas geospaciales de IDECYL, incorporando información ambiental relevante como zonas de exclusión eólica y áreas protegidas.

Toda la API del sistema está documentada mediante Flask-RESTX, integrando una interfaz Swagger que facilita la exploración, prueba y validación de los endpoints. Esto permite a los desarrolladores e investigadores interactuar fácilmente con los servicios del sistema, consultar la estructura de los datos y automatizar procesos analíticos o de ingestión.

En cuanto a la infraestructura de almacenamiento, el sistema se apoya en una arquitectura PostgreSQL distribuida en tres instancias independientes, cada una optimizada para un propósito específico:

1. Instancia de Configuración: almacena la información estructural de la plataforma, incluyendo usuarios, roles, credenciales y metadatos de cámaras, garantizando una gestión segura y segmentada de los accesos.
2. Instancia de Ingesta Operativa: contiene los datos brutos generados por los sensores (segmentos de vídeo, estadísticas, observaciones acústicas, detecciones visuales y referencias a mapas de calor), sirviendo como la base de datos activa para la explotación inmediata.
3. Instancia de Análisis: se especializa en la generación de vistas materializadas y agregados temporoespaciales, que permiten realizar análisis avanzados sobre la distribución de especies, frecuencia de avistamientos, comportamiento de vuelo y patrones de migración.

El sistema emplea Redis como broker de Celery y como caché de resultados, optimizando la latencia en la ejecución de tareas distribuidas y acelerando las consultas frecuentes sobre datos de gran volumen. Este enfoque de almacenamiento en memoria reduce significativamente el tiempo de respuesta y aumenta la eficiencia del sistema frente a operaciones concurrentes o de alta demanda.

Por otro lado, el componente MediaMTX gestiona la recepción y retransmisión de flujos de vídeo desde los dispositivos IoT desplegados en campo. Este servicio recibe señales RTSP (Real Time Streaming Protocol) desde las cámaras y las convierte en flujos HLS (HTTP Live Streaming), garantizando la compatibilidad con navegadores y reproductores web. Además, su API expone endpoints REST que permiten registrar o eliminar orígenes de vídeo dinámicamente, sin necesidad de reiniciar servicios ni modificar otros componentes del sistema. Esta característica facilita la escalabilidad del ecosistema IA4Birds, permitiendo incorporar nuevos sensores o cámaras sin interrumpir el funcionamiento operativo de la plataforma.

4 Conclusión

IA4Birds consolida un espacio de datos interoperable que fusiona de forma consistente flujos IoT (visual/acústico), fuentes externas (ciencia ciudadana y capas territoriales) y salidas de modelos (predicciones/estadísticas). La elección de PostgreSQL como núcleo transaccional—combinada con almacenamiento de artefactos en objeto, microservicios desacoplados (ingesta, coordinación, backend y streaming), APIs documentadas (OpenAPI/Swagger) y caché en memoria (Redis)—garantiza trazabilidad, integridad ACID y rendimiento bajo carga. El ingest service y la orquestación con Celery permiten sincronizaciones confiables y pre-cómputo (vistas materializadas/aggregados), mientras que MediaMTX asegura compatibilidad web en tiempo real para las fuentes de vídeo.

El resultado es una plataforma escalable y reproducible que reduce la latencia de las visualizaciones, facilita el análisis temporoespacial y habilita decisiones informadas para la planificación eólica compatible con la conservación. La alineación temporal/espacial, la normalización taxonómica y el manejo explícito de la incertidumbre convierten datos heterogéneos en evidencia operativa, preparada para auditoría y reutilización.

Impactos medibles

- Menor tiempo de respuesta en paneles y mapas gracias a caché y pre-cómputo.
- Mayor calidad y cobertura de datos por sincronización periódica con eBird/Xeno-Canto e integración IDECYL/Mapa Eólico.
- Mantenibilidad y seguridad mejoradas por separación de responsabilidades, control de acceso y observabilidad extremo a extremo.