Documento de Análisis y Diseño

de

ZDex

Fecha	07/11/25
Versión	2.0
Integrantes del grupo	Clemente Mujica Cristóbal Moraga Felipe Tapia Iván Weber Camilo Troncoso

1 Introducción

1.1 Diagnóstico e identificación del problema, desafío u oportunidad

En el pasado era muy común que las personas tuvieran o convivieran con animales de distintas especies y razas, existiendo un conocimiento general sobre la biodiversidad, alimentación, hábitos, etc. de estos. Sin embargo, con nuestra acelerada sociedad actual y su estilo de vida en las urbes, se ha perdido cercanía con la naturaleza y los conocimientos básicos sobre flora y fauna que antes existían.

Los nuevos alcances y tecnologías de aprendizaje de máquinas que han explotado en los últimos años abren una serie de oportunidades para crear nuevas formas de compensar la ignorancia sobre la biodiversidad de la sociedad moderna, e incluso, de aumentar el conocimiento general que antes existía.

1.2 Propuesta de solución

Nuestra propuesta, ZDex, consiste en una enciclopedia virtual de animales que funcione con detección de objetos en tiempo real.

Para lograr aquello se ocuparán técnicas de procesamiento de imágenes como Redes Neuronales Convolucionales diseñadas para detección de objetos (ej: YOLO), además de tecnologías de deep learning a partir de Tensorflow y/o Keras para rediseñar y ocupar redes neuronales y datasets ya existentes, con ello poder entrenar una máquina que identifique a un animal que se encuentre en cámara y se entregue información respecto a este. El proyecto se desarrollará en Python, ya que es un lenguaje de programación que ofrece soporte para todas las tecnologías mencionadas anteriormente.

1.3 Estado del Arte y de la Técnica

Seek: Es un proyecto de una organización sin fines de lucro llamada iNaturalist, consiste es una aplicación de teléfono móvil (Android y Ios) que permite identificar flora y fauna de distintas regiones del mundo a través de predicciones superpuestas en la cámara. También existió una primera versión donde era necesario tomar una fotografía para realizar la identificación, pero la dinámica era lenta y frustrante, ya que había que tomar una nueva fotografía cada vez que fallaba la identificación. El dataset ocupado proviene de la misma comunidad de iNaturalist que incluyen fotos e información de los animales presentes en la región del miembro, esto quiere decir que el reconocimiento funciona peor en las regiones donde hay pocos o no hay miembros de la comunidad. (iNaturalist, 2019)

SpeciesNet: Es un proyecto de Inteligencia Artificial desarrollado por organizaciones de conservación animal y Google, consiste en un modelo de código abierto con aprendizaje automático para clasificación de una amplia gama de especies con una extraordinaria precisión. Según sus evaluaciones, detecta el 99.4% de las imágenes que contienen animales, donde en un 98.7% de estos casos acierta. También, aseguran que obtienen una precisión del 94.5% en la predicción de la especie. El dataset se compone de 65 millones de imágenes provenientes de cámaras activadas por movimiento o “cámaras trampas” pertenecientes a Wildlife Insights. El proyecto realiza la clasificación con MegaDetector, el cual es otro modelo de inteligencia artificial basado en YOLO (modelo de Red Neuronal Convolucional) y COCO camera traps (formato JSON para subir imágenes a la base de datos), que detecta seres vivos (no identifica especies). (Hehmeyer, 2025) (Google, 2024) (Morris, 2025)

Google Lens: Es una aplicación de Google con un conjunto de funciones informáticas basados en la visión que le permiten comprender lo que estás mirando y usar esa información para copiar o traducir texto, **identificar plantas y animales**, explorar lugares o menús, descubrir productos, encontrar imágenes visualmente similares y llevar a cabo otras tareas útiles. Lens compara los objetos de tu foto con otras imágenes a las que clasifica por su grado de similitud y relevancia con respecto a los objetos de tu foto original. También analiza los objetos que aparecen en la foto para encontrar otros resultados relevantes en la Web. Además, por medio de otros indicadores útiles, como palabras, texto y otros metadatos del sitio que aloja la imagen, puede determinar la clasificación y la relevancia.

Cuando Lens analiza una imagen, suele generar varios resultados posibles y clasificarlos por la relevancia probable de cada uno. A veces acota esas posibilidades a un solo resultado. Supongamos que Lens está mirando a un perro que identifica con una probabilidad del 95% como pastor alemán y con una del 5% como corgi. En este caso Lens podría mostrar solo el resultado correspondiente al pastor alemán, por considerarlo más parecido desde el punto de vista visual. (Google, -)

1.4 Referencias

Google. (-). *¿Qué es Google Lens?* Google Lens. https://lens.google/intl/es/howlensworks/

Google. (2024, Diciembre 4). *AI models trained by Google to classify species in images from motion-triggered wildlife cameras.* GitHub. https://github.com/google/cameratrapai

Hehmeyer, A. (2025, Marzo 10). *Utilizando el poder de la IA para identificar y rastrear especies*. World Wildlife Fund. https://www.worldwildlife.org/descubre-wwf/historias/utilizando-el-poder-de-la-ia-para-identificar-y-rastrear-especies

iNaturalist. (2019, Abril 5). *Real-time Computer Vision predictions in Seek by iNaturalist version 2.0*. iNaturalist. https://www.inaturalist.org/blog/23075-real-time-computer-vision-predictions-in-seek-by-inaturalist-version-2-0

Morris, D. (2025, Julio 23). *MegaDetector*. Github. https://github.com/agentmorris/MegaDetector

2 Requisitos del Sistema

2.1 Requisitos Funcionales

RF1 El sistema debe funcionar con reconocimiento en tiempo real desde la cámara.

RF2 El sistema debe identificar por lo menos la especie del animal e idealmente la raza en específico.

RF3 El sistema debe entregar nombre, nombre científico, especie, raza (si se tiene entrenado), imagen de referencia y hábitat del animal identificado.

RF4 El sistema debe rodear al animal con un bounding box en tiempo real.

RF5 Panel lateral con información detallada del animal.

2.2 Requisitos no funcionales

RNF1 Clasificación de especies con precisión mayor al 80%.

RNF2 Tiempo de respuesta < 5 s.

RNF3 Consumo de memoria RAM < 8 GB en operación normal.

RNF4 Uso de TensorFlow/Keras para modelos de ML.

2.3 Requisitos de Interfaces

Tabla 1: Eventos externos

Evento	Descripción	Iniciador	Parámetros	Respuesta
InicioCámaraSolicitado	Usuario solicita activar la cámara para detección.	Usuario	-	InicializaCámara
FrameCapturado	Se obtiene un nuevo frame de la cámara,	Hardware Cámara	Frame Resolución	DetecciónAnimales
AnimalEncontrado	En el frame se detecta un animal,	Hardware Cámara	Frame	IdentificaciónAnimal
InfoDetallada	Usuario presiona botón para ver info. detallada,	Usuario	NombreAnimal	PanelInfo

La Tabla 2 muestra las respuestas del sistema frente a eventos externos.

Tabla 2: Respuestas del sistema

Respuesta	Descripción	Parámetros
InicializaCámara	El sistema solicita el uso de la cámara al SO.	Nombre cámara

Respuesta	Descripción	Parámetros
DetecciónAnimales	El sistema inicia su algoritmo de detección para identificar animales,	Frame
IdentificaciónAnimal	El sistema entrega el frame a la red neuronal para identificar el animal,	Frame
PanelInfo	El sistema debe desplegar un panel lateral con información del animal identificado.	NombreAnimal

2.4 Requisitos de Ambiente

Hardware:

- Computador.
- GPU.
- Webcam.

Software:

- Python
- Tensorflow
- Keras
- OpenCV
- Tkinter

2.4.1 Hardware de Desarrollo

- Computador: Dispositivo donde se realizará el procesamiento de imagen, creación de red neuronal y uso de esta.
- GPU: Unidad de procesamiento que mejorará la capacidad de cómputo del computador en las tareas de análisis de imagen.
- Webcam: Periférico necesario para obtener imágenes del entorno y los animales, puede ser cámara propia o externa (ej: Celular).

2.4.2 Software de Desarrollo

- Python: Lenguaje de programación que se utilizará para desarrollar el proyecto.
- Tensorflow: Biblioteca multiplataforma de código abierto para construir y entrenar modelos de redes neuronales que identifiquen y clasifiquen a los animales.
- Keras: Biblioteca de Python de código abierto para simplificar el diseño de nuevas redes neuronales o implementar redes neuronales ya existentes.
- OpenCV: Biblioteca multiplataforma de código abierto que provee herramientas de segmentación y lectura de modelos CNN para detectar a los animales.
- Tkinter: Interfaz por defecto de Python para el kit de herramientas de GUI Tk, se utilizará para desarrollar la GUI de ZDex.

2.4.3 Datasets e Imágenes/Videos de Prueba

Para nuestra solución ocuparemos el dataset con el que se entrena al proyecto “SpeciesNet”, ya que será nuestra base para desarrollar ZDex, estos datos se componen de aproximadamente 65 millones de imágenes de cámaras trampas ya clasificadas: 42 millones de un dataset no público de Wildlife Insights, 19.3 millones de un dataset público de un repositorio llamado The Labelled Information Library of Alexandria: Biology and Conservation y 180k de un dataset público llamado iWildcam.

2.5 Perfiles de Usuario

Perfil	Socioeconómico y cultural	Ocupacional	Etario	Características físicas, fisiológicas, psicológicas.	Otros
Niños/Estudiantes de primaria	Nivel socioeconómico medio-bajo (requieren de acceso a computadores/celulares)	Estudiantes de educación básica	6-12 años	Tiempo de atención corto. Mentalidad curiosa, lúdica y con aprendizaje basado en descubrimiento	Desarrollo de habilidades digitales y conocimientos de biodiversidad
Docentes y Monitores	Contexto laboral	Docentes de ciencias o monitores de excursiones	21-65 años	En ocasiones, poca creatividad para enseñar. Conocimiento limitado respecto a las especies (no conocen todas).	Desarrollo de habilidades digitales, de enseñanza y conocimientos de biodiversidad
Animalistas	Variable, pero con acceso a computadores o celulares	Voluntarios de reservas naturales, activistas o guías de turismo ecológico	18-65 años	Empatía con la naturaleza y propósitos de conservación. Estado físico medio (uso de aplicación en intemperie)	Desarrollo de habilidades digitales y conocimientos de biodiversidad

3 Planificación del Proyecto

3.1 Objetivo General (Goal)

El objetivo general del proyecto ZDex es desarrollar una aplicación de escritorio que permita la identificación en tiempo real de animales mediante técnicas de visión por computador e inteligencia artificial, proporcionando al usuario información relevante y accesible sobre cada especie reconocida.

La aplicación busca facilitar el acceso a conocimientos sobre fauna, combinando detección automática con modelos de deep learning y una interfaz intuitiva y amigable, que entregue datos de interés como nombre común, nombre científico, hábitat y características principales.

De esta forma, ZDex integra tecnologías de procesamiento de imágenes, redes neuronales convolucionales y un sistema de visualización interactivo, con el fin de ofrecer una experiencia educativa y entretenida para aprender sobre fauna, especialmente en contextos donde la información no está fácilmente disponible.

3.2 Objetivos Específicos

O1. Captura y preprocesamiento de imágenes

Desarrollar el componente que permite obtener imágenes en tiempo real desde la cámara y aplicar un preprocesamiento básico (ajuste de tamaño, normalización de intensidades y conversión de color) para preparar los frames de entrada a la red neuronal.

O2. Modelo de detección y clasificación

Seleccionar un modelo de deep learning ya entrenado y adaptarlo a las necesidades del proyecto, configurando los parámetros para identificar animales en tiempo real con una precisión aceptable.

O3. Interfaz gráfica de usuario

Diseñar e implementar una GUI intuitiva que muestre la transmisión en vivo con la detección resaltada mediante bounding boxes y que permita la interacción del usuario de forma simple.

O4. Panel informativo de especies

Integrar un panel lateral que despliegue información complementaria de cada animal identificado, incluyendo nombre común, nombre científico, hábitat, imagen de referencia y otros datos relevantes.

O5. Validación y pruebas del sistema

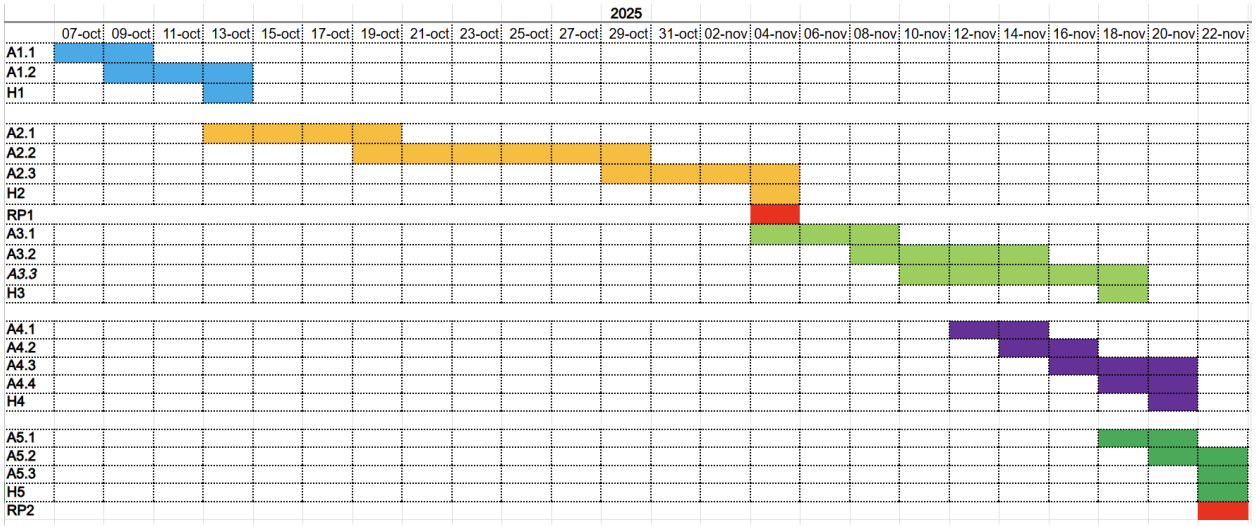
Realizar pruebas con imágenes y videos reales para evaluar el rendimiento del sistema, verificando tiempos de respuesta menores a 5 segundos y una precisión superior al 80%.

3.3 Actividades y Milestones

	Responsable	Participación equipo	Fecha comienzo	Fecha término	% de avance
A1.1 Configuración de la cámara	Ivan	5%	07/10	09/10	0%
A2.1 Preprocesamiento básico	Ivan	5%	09/10	13/10	0%
Milestone 1 Módulo de captura funcional	Ivan	-	13/10	13/10	0%
A2.1 Selección del modelo	Camilo	10%	13/10	19/10	0%
A2.2 Adaptación del modelo	Camilo	20%	19/10	29/10	0%
A2.3 Integración con flujo de video	Camilo	15%	29/10	04/11	0%
Milestone 2 Modelo integrado y detectando	Camilo	-	04/11	04/11	0%
A3.1 Diseño de GUI	Cristobal	5%	04/11	08/11	0%
A3.2 Implementación vista de cámara	Cristobal	10%	08/11	14/11	0%
A3.3 Eventos de usuario	Cristobal	10%	10/11	18/11	0%
Milestone 3 Prototipo GUI funcional	Cristobal	-	18/11	18/11	0%
A4.1 Conexión con API externa	Clemente	5%	12/11	14/11	0%
A4.2 Procesamiento de respuesta	Clemente	5%	14/11	16/11	0%
A4.3 Integración del panel lateral	Clemente	5%	16/11	20/11	0%
A4.4 Optimización de visualización	Clemente	5%	18/11	20/11	0%

Milestone 4 Panel integrado con API	Clemente	-	20/11	20/11	0%
A5.1 Pruebas unitarias, Test por módulo.	Felipe	5%	18/11	20/11	0%
A5.2 Pruebas de integración.	Felipe	5%	20/11	22/11	0%
A5.3 Evaluación de rendimiento.	Felipe	5%	22/11	22/11	0%
Milestone 5 Validación del sistema	Felipe	-	22/11	22/11	0%

3.4 Carta Gantt



4 Diseño de Arquitectura del Sistema

4.1 Diagrama de Contexto

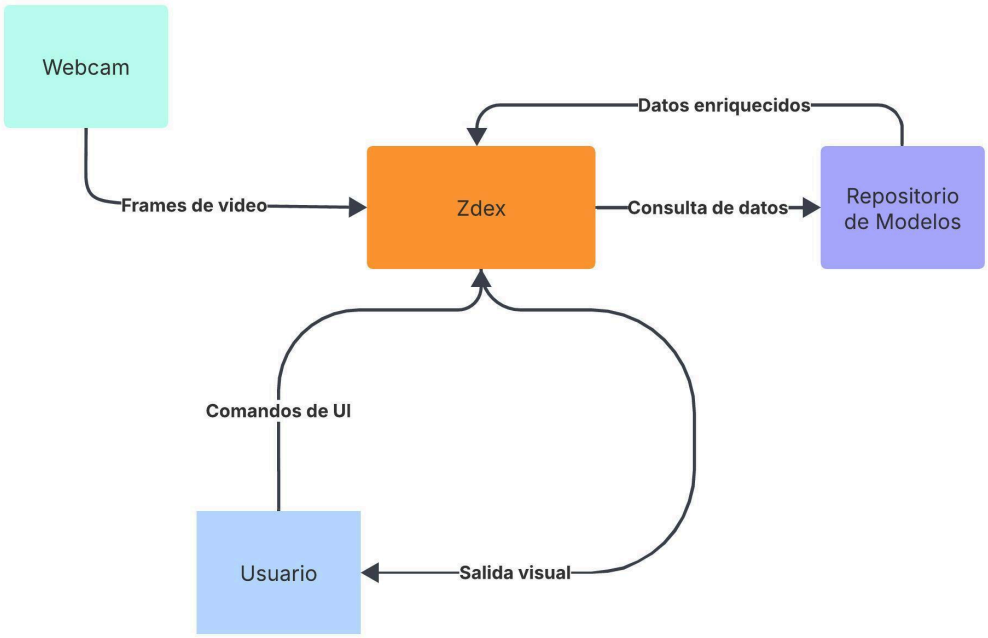
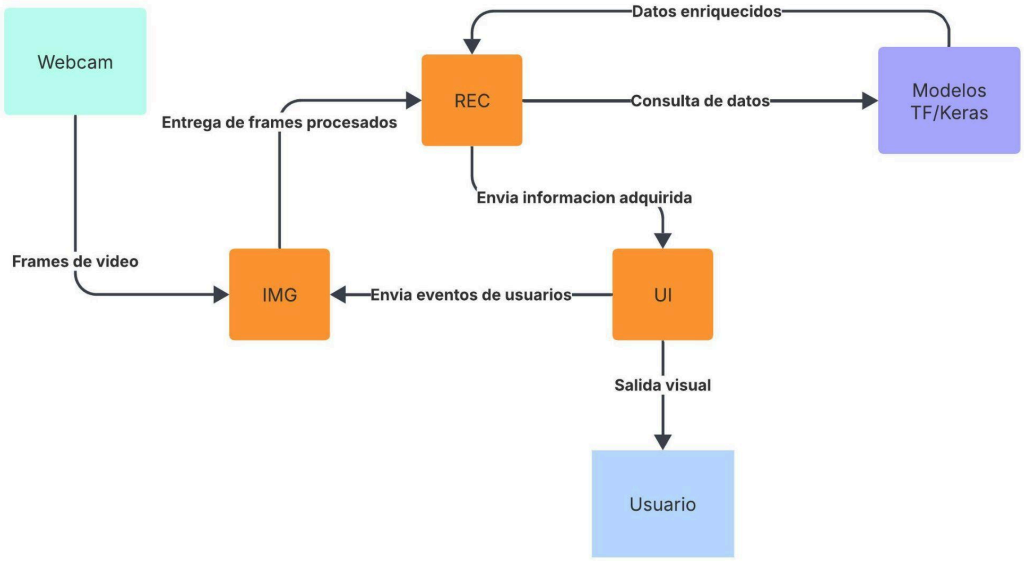


Figura 1: Diagrama de contexto del sistema

4.2 Diagrama de Arquitectura



Breve descripción de el diagrama de arquitectura:

Webcam→IMG: Entrega de frames crudos (Entrada en tiempo real).

IMG→REC: Entrada de frames pre procesados para detección y clasificación.

REC→MODELOS TF/KERAS: Uso de modelos online.

MODELOS TF/KERAS→REC: Modelos actualizados para la detección de las especies y razas con precisión.

REC→UI: Envía las especies/razas para la visualización.

UI→IMG: Envía eventos como iniciar/detener cámara, captura de imagen, etc.

UI→Usuario: Entrega de información al usuario de forma visual.

Figura 2: Diagrama de arquitectura del sistema

4.3 Enumeración de Módulos

La Tabla 3 muestra los módulos de la Figura 2. Por cada módulo se entrega un breve párrafo descriptivo de su propósito, además de la sección en donde se especifica el módulo en detalle.

Tabla 3: Módulos de la arquitectura del sistema

Módulo	**Propósito**	**Sección**
Módulo de Captura y Procesamiento de Imagen (IMG)	Este módulo se encarga de inicializar la cámara, gestionar la captura de imágenes en tiempo real y realizar un preprocesamiento básico de cada frame, como la normalización de tamaño, color y segmentación de regiones relevantes. Su función principal es preparar los datos de entrada para que el módulo de reconocimiento pueda operar con mayor precisión y eficiencia.	5
Módulo de Reconocimiento y Clasificación (REC)	La función principal de este módulo es la aplicación de técnicas para la detección e identificación de animales en los frames entregados del módulo anterior. Uso de redes neuronales para la clasificación de la especie y raza del animal.	5
Módulo de Interfaz y Presentación de Resultados (UI)	Este módulo se encarga de mostrar el video en tiempo real con el animal identificado y su panel lateral con la información detallada proveniente del módulo REC. Además, administra los eventos de usuario, como iniciar la cámara o capturar una imagen, y se relaciona directamente con el módulo IMG para la gestión de la cámara.	5

4.4 Matriz de Requisitos Funcionales y Módulos

Reescriba los requisitos funcionales, con su abreviatura. La siguiente matriz muestra qué módulos del sistema implementan qué requisitos funcionales.

Tabla 4: Matriz de requisitos funcionales y módulos

	IMG	REC	UI
RF1	X	X	
RF2		X	
RF3		X	X
RF4	X		
RF5			X

5 Módulos

5.1 Definición del Módulo IMG — Captura y Procesamiento de Imagen

Propósito

Gestionar la adquisición de frames desde la(s) cámara(s) y preparar frames listos para inferencia: lectura, decodificación, normalización, ajuste de resolución, (opcional) detección de movimiento o ROI y envío a la cola de inferencia.

General

- Productor de datos: IMG lee datos entregados por hardware y publica Frame en la cola compartida queue_frames.
- Consumo: REC toma frames desde queue_frames.
- Control desde UI: comandos START_CAMERA, STOP_CAMERA, SET_RESOLUTION, CAPTURE_IMAGE.
- Eventos emitidos: FrameCapturado(), CamaraError(), FramePreprocesado().

Marco Técnico del Módulo

- Lenguaje: Python 3.10+.
- Librerías: OpenCV (cv2), numpy, queue, threading.
- Formato de frame: numpy.ndarray dtype uint8, shape (H,W,3) BGR (OpenCV). Cuando vaya al modelo, convertir a float32 y normalizar según el modelo (p.ej. /255.0 o mean/std)
- Opciones de optimización: leer siempre el *último* frame disponible (drop older) para reducir latencia.

Estructura General

- Entradas: Comandos desde UI.
- Salidas: queue_frames.put(FrameEnvelope) donde FrameEnvelope = {frame, ts, device_id, seq_id, metadata}.
- Configuración: device_id, target_resolution, fps_limit, buffer_policy (latest o fifo).

5.2 Definición del Módulo REC — Reconocimiento y Clasificación (detección en tiempo real)

Propósito

Ejecutar la inferencia en cada frame recibido, aplicar postprocesado (NMS, filtrado por umbral), mapear class_id → metadata (nombre común, nombre científico, link) y emitir Detections al UI.

General

- Consume FrameEnvelope desde queue_frames.
- Produce DetectionPackage hacia queue_detections y eventos DeteccionLista.
- Puede consultar modelo local o servicio remoto (fallback configurable).

Marco Técnico del Módulo	• Frameworks de entrenamiento/inferencia: PyTorch (YOLOv5/YOLOv8) o TensorFlow (EfficientDet, TF Object Detection API). • Formato de modelo para despliegue: ONNX (portabilidad) o TensorRT (NVIDIA). Para dispositivos ARM, exportar a TFLite si se requiere. • Pre/postprocesos: redimensionar al input del modelo, normalizar, aplicar NMS (IoU threshold 0.4–0.6), filtrar por confidence threshold (ej. 0.25–0.5). • Librerías auxiliares: onnxruntime, torch, numpy, opencv, pycocotools (evaluación), tqdm (entrenamiento/benchmarks).
Estructura General	• Entradas: FrameEnvelope (frame, ts, id). • Salidas: DetectionPackage = {frame_id, ts_infer, detections:[{bbox:[x1,y1,x2,y2],class_id,label,confidence}], model_version, metrics}.

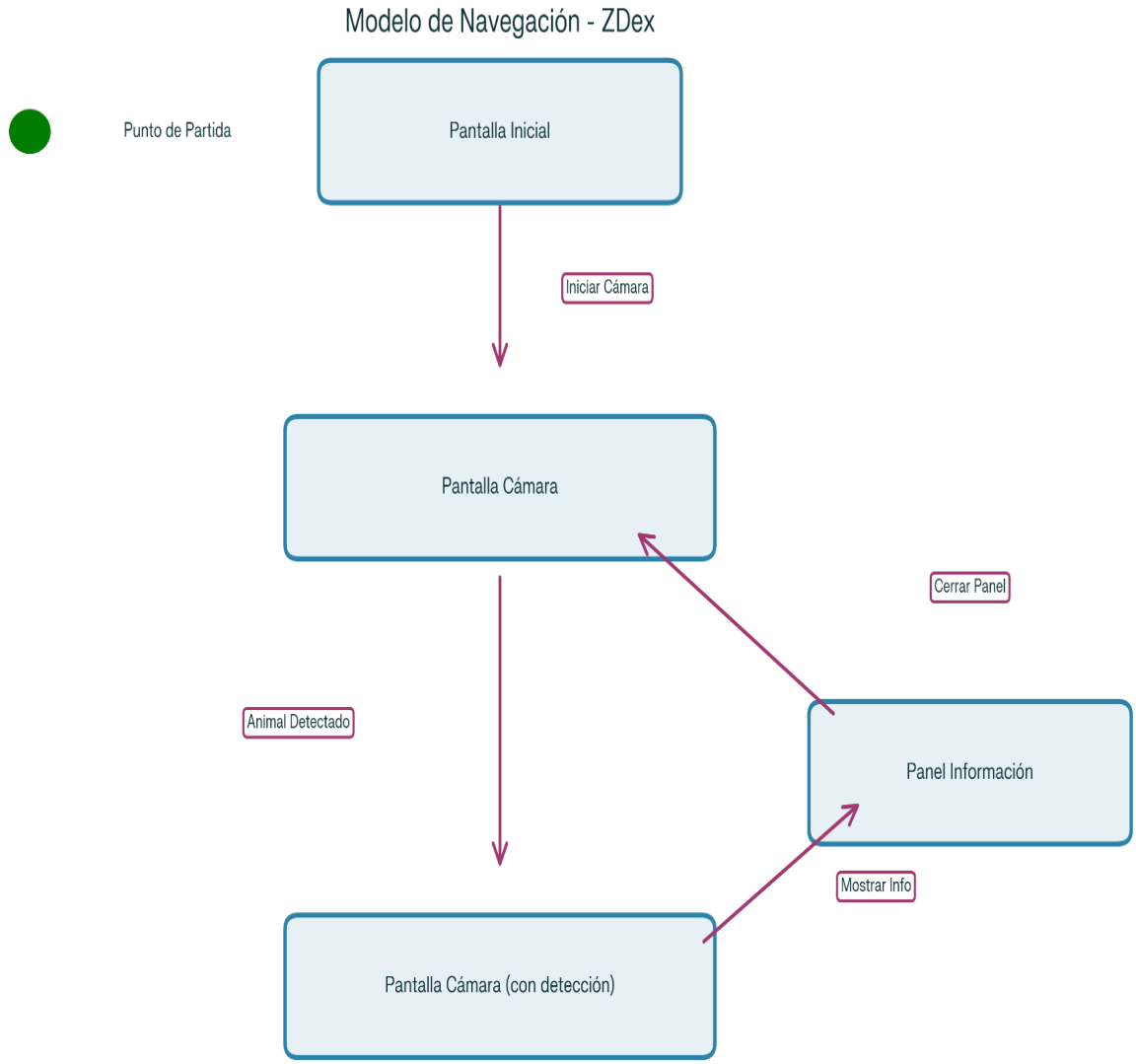
5.3 Definición del Módulo UI — Interfaz y Presentación de Resultados

Propósito	Visualizar el video con bounding boxes y mostrar el panel con información de la detección (nombre común, nombre científico, confianza, imagen de referencia, botón para confirmar/corregir). Proveer controles (start/stop, settings, exportar) y registrar feedback del usuario.
General	• Interactúa con IMG (comandos start/stop) y con REC (consume queue_detections). • Produce feedback (correcciones humanas) que se guardan en DM para reentrenamiento. • Debe ser reactiva y no bloquear procesos de inferencia.
Marco Técnico del Módulo	• Stack inicial: Tkinter para prototipo (ligero). • Render de frames: convertir numpy → PIL.Image → PhotoImage (Tkinter). • Cache: almacenar localmente imágenes de referencia en ./cache/species_images/{class_id}.jpg para evitar latencias por red.
Estructura General	• Eventos entrantes: DeteccionLista (detections), FrameCapturado (meta). • Comandos salientes: START_CAMERA, STOP_CAMERA, SET_CONF_THRESHOLD, EXPORT_LOGS, USER_FEEDBACK({frame_id, detection_idx, correct/label}).

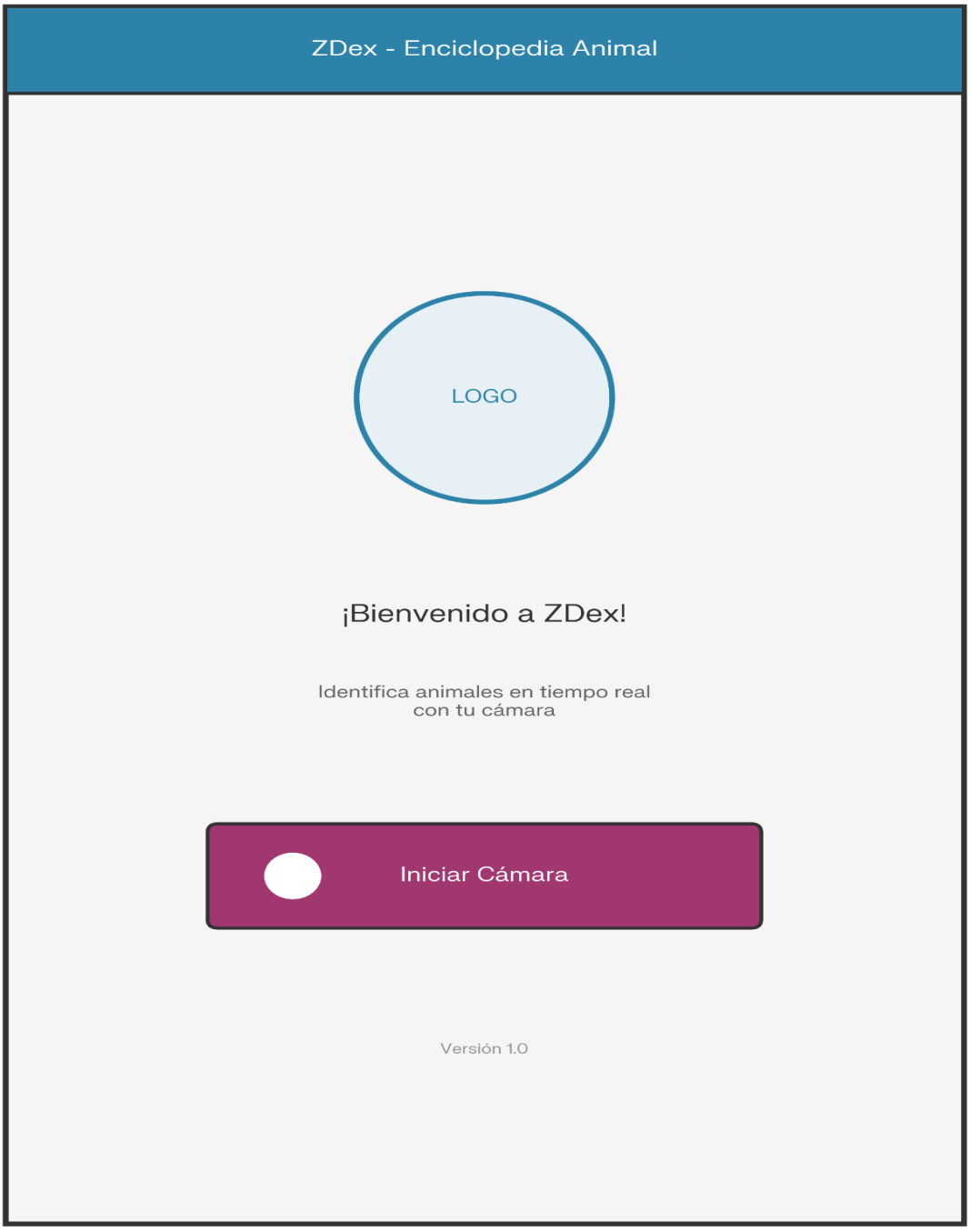
6 Diseño de Interfaces

6.1 Modelo de Navegación

El modelo de navegación de ZDex presenta un flujo simple e intuitivo que permite al usuario interactuar con el sistema de identificación de animales. El punto de partida es la Pantalla Inicial, desde donde el usuario puede activar la cámara. Una vez en la Pantalla de Cámara, el sistema detecta animales en tiempo real y permite visualizar información detallada mediante un Panel Información lateral. El usuario puede cerrar este panel y regresar a la detección en cualquier momento.



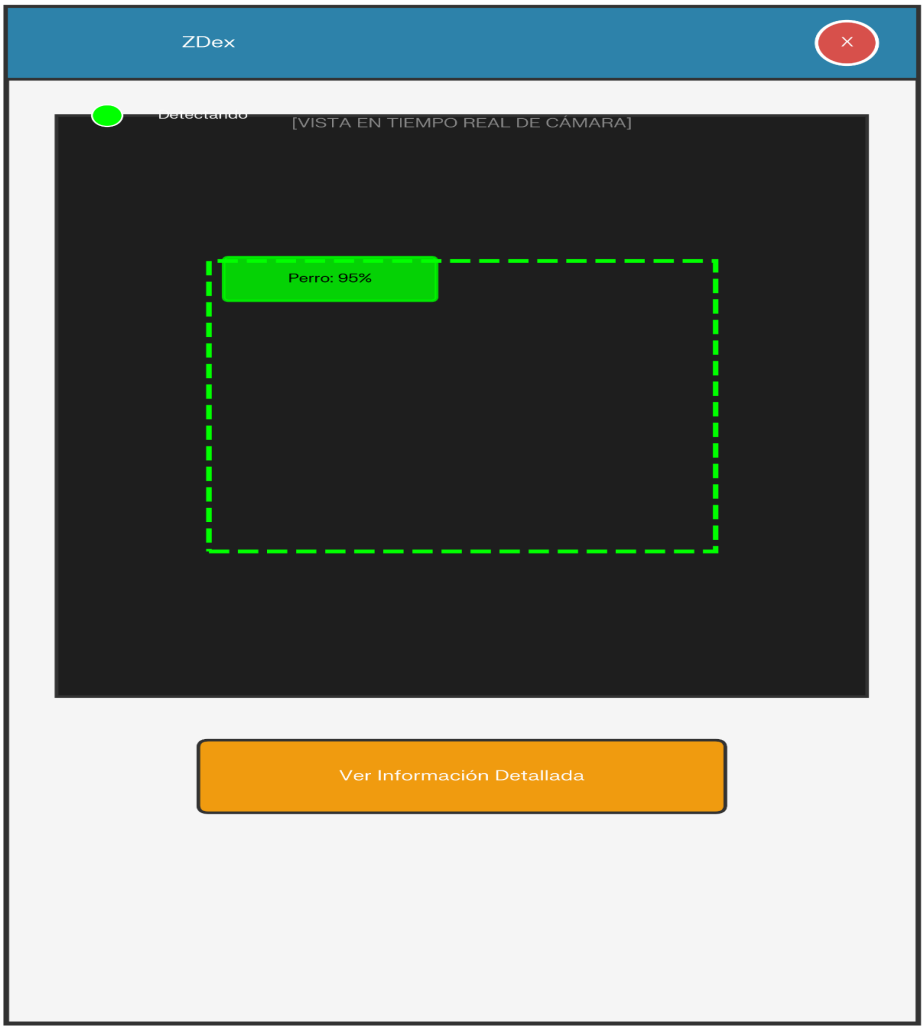
6.2 Prototipos de Interfaces Usuarias



6.2.1 Prototipo 01: Pantalla Inicial

Esta interfaz constituye el punto de entrada a la aplicación ZDex. Presenta un diseño limpio y minimalista que comunica claramente el propósito de la aplicación y permite al usuario iniciar la funcionalidad principal con un solo clic.

Evento	Interacción	Acción	Objeto afectado
Iniciar Cámara	¡El usuario presiona el botón "Iniciar Cámara"	Se activa el hardware de la cámara, se solicita permiso al sistema operativo, y se navega a la Pantalla de Cámara	Sistema de Cámara, Pantalla de Cámara
Cerrar Aplicación	El usuario cierra la ventana de la aplicación mediante el botón de cierre del sistema operativo	La aplicación se cierra completamente y se liberan los recursos	Sistema completo

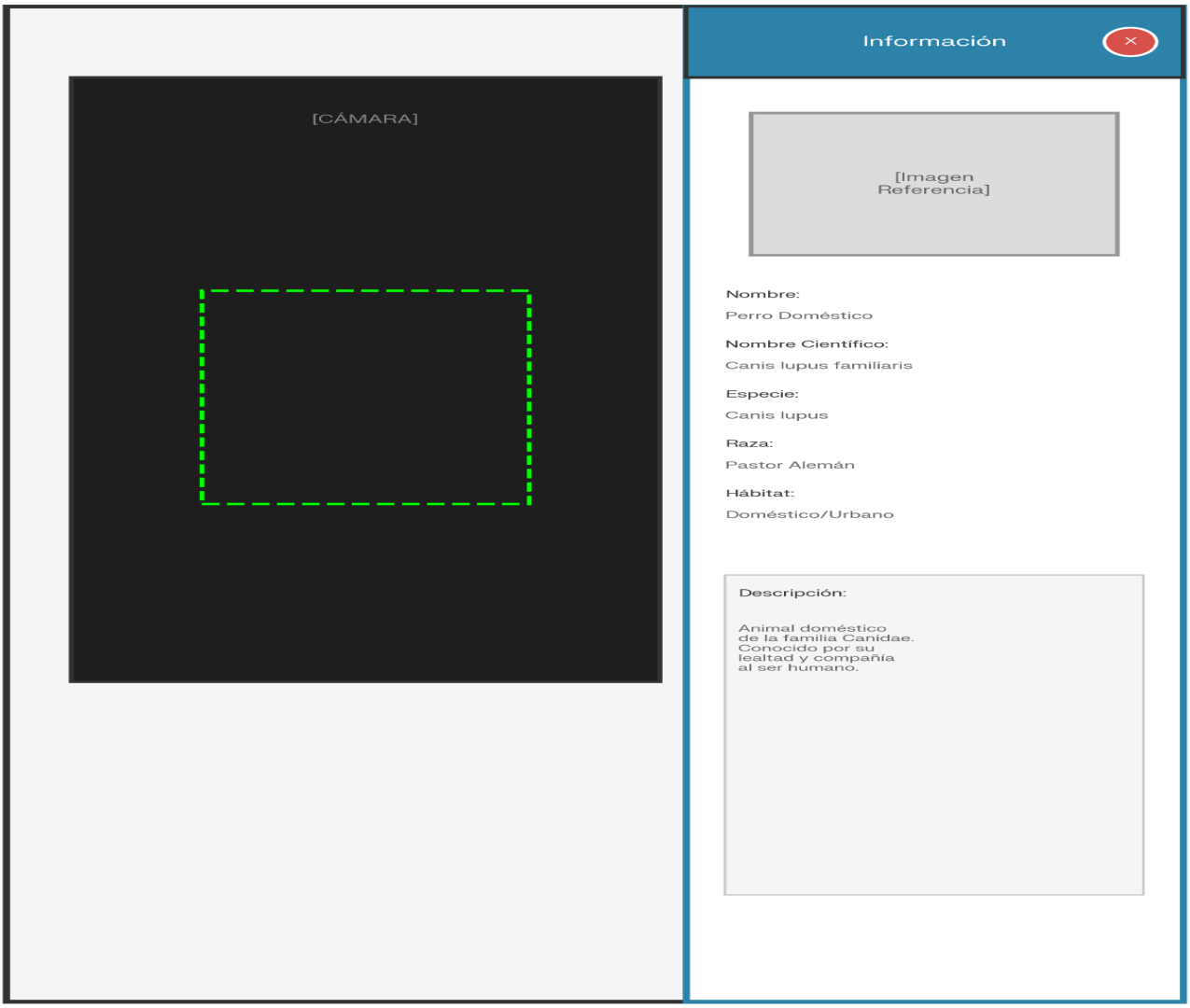


6.2.2 Prototipo 2: Pantalla de Cámara con Detección

Esta interfaz muestra la vista en tiempo real de la cámara con capacidades de detección de animales. Cuando el sistema identifica un animal, se dibuja un bounding box verde alrededor del mismo y se muestra la clasificación con su nivel de confianza.

Evento	Interacción	Acción	Objeto afectado
Detección Animal	El sistema detecta un animal en el frame de la cámara mediante el modelo de ML	Se dibuja un bounding box verde alrededor del animal detectado, se muestra la etiqueta con el nombre del animal y el porcentaje de confianza, y se activa el botón "Ver Información Detallada"	Vista de Cámara, Bounding Box, Etiqueta de Clasificación, Botón de Información

Ver Información	El usuario presiona el botón "Ver Información Detallada"	Se abre el Panel de Información lateral mostrando los detalles completos del animal detectado	Panel de Información, Layout de la pantalla
Cerrar Cámara	El usuario presiona el botón "X" en la esquina superior derecha	Se detiene el stream de la cámara, se liberan los recursos del hardware y se retorna a la Pantalla Inicial	Sistema de Cámara, Pantalla Inicial
Actualización Frame	El hardware de la cámara captura un nuevo frame automáticamente (30-60 fps)	El sistema procesa el nuevo frame, ejecuta el modelo de detección y actualiza la vista	Vista de Cámara, Modelo de ML



6.2.3 Prototipo 3: Pantalla de Cámara con Panel de Información

Esta interfaz combina la vista reducida de la cámara con un panel lateral que presenta información detallada sobre el animal identificado. El diseño permite al usuario continuar viendo la detección en tiempo real mientras consulta los datos del animal.

Evento	Interacción	Acción	Objeto afectado
Cerrar Panel	El usuario presiona el botón "X" en la cabecera del Panel de Información	El panel lateral se cierra con una animación, la vista de cámara se expande a tamaño completo	Panel de Información, Vista de Cámara

Cargar Imagen	El sistema obtiene una imagen de referencia del animal desde la base de datos	Se muestra la imagen de referencia en el área designada del panel	Área de Imagen de Referencia
Mostrar Datos	El sistema recupera y muestra los datos del animal (nombre, nombre científico, especie, raza, hábitat, descripción)	Los datos se cargan y se muestran en los campos correspondientes del panel	Campos de Información del Panel
Desplazar Panel	El usuario hace scroll en el panel para ver más información (si el contenido es extenso)	El contenido del panel se desplaza verticalmente	Contenido del Panel de Información
Detección Continua	El sistema continúa detectando animales en la vista reducida de la cámara	Si se detecta un animal diferente, se actualiza el bounding box y opcionalmente se puede actualizar el panel	ista de Cámara Reducida, Bounding Box, Panel de Información