High Level Design Document

Introduction

This High Level Design (HLD) document outlines the architecture and core components for **EcoClassify - Wildlife Image Classifier**. EcoClassify is a Streamlit-based application leveraging transfer learning (ResNet) to classify wildlife images into multiple species, featuring data augmentation, model fine-tuning, and an explanation dashboard for predictions.

---

1. System Architecture Overview

Architecture Summary:
EcoClassify is a modular Python application with a Streamlit front-end, a PyTorch-based model backend, and supporting data processing and explanation modules.

Module	Description
User Interface	Streamlit app for image upload, results display, and explanation dashboard.
Image Preprocessing	Handles image validation, resizing, and augmentation (OpenCV, NumPy).
Model Inference	Loads fine-tuned ResNet model, performs classification (PyTorch).
Explanation Engine	Generates prediction explanations (e.g., Grad-CAM visualizations).
Data Management	Handles input/output, logging, and optional result storage (Pandas).

---

2. Component Interactions

Sequence Step	Interaction Description
1	User uploads image via Streamlit UI.
2	Image Preprocessing module processes and augments the image.
3	Processed image is passed to Model Inference for classification.
4	Model Inference returns predicted species and confidence scores.
5	Explanation Engine generates visual/textual explanation for the prediction.
6	Results and explanations are displayed in the Streamlit UI; optionally logged/stored.

---

3. Data Flow Overview

Data Source	Processed By	Output/Next Step
User Image Upload	Image Preprocessing	Augmented/validated image
Preprocessed Image	Model Inference	Prediction (species, confidence)

Prediction Result	Explanation Engine	Explanation visualization/text
All Results	Data Management	Display, logging, optional storage

---

4. Technology Stack

Layer/Function	Technology/Framework
Front-End UI	Streamlit
Model & Inference	PyTorch (ResNet)
Image Processing	OpenCV, NumPy
Data Handling	Pandas
Explanation/Visualization	Grad-CAM, Matplotlib (if needed)
Environment	Python 3.x

---

5. Scalability & Reliability

- **Scalability:**
  - Designed for single-user or small group research/education use; can be containerized for deployment.
  - Model and preprocessing are stateless per request, enabling horizontal scaling if needed.
- **Reliability:**
  - Input validation and error handling at each stage.
  - Modular design allows for independent testing and updates.
  - Security: Only local file uploads; no external data storage by default.

End of Document