WildGuard : AI-Powered Wildlife Conservation System

Satya Sheela D.[1]; Saniya S.[2]; Srinidhi R. Y.[3]; Umesh L.[4]; Vivin Vaibhav L. K.[5]

[1]Professor, [2, 3, 4, 5]Student at Department of Computer Science and Design Engineering,
K S Institute of Technology College, Bengaluru, India – 560109

Publication Date: 2025/12/19

Abstract: WildGuard is an AI-powered wildlife conservation system designed to identify animals, detect threats, and support rapid rescue responses. The platform uses machine learning, Internet of things (IOT) sensors, and geolocation to monitor habitats in real time and alert authorities to poaching risks, injured animals, and unusual activity. It also provides species information, conservation status, and connects users to nearby wildlife care centers. WildGuard offers a fast, scalable, and accessible solution for strengthening wildlife protection and improving conservation outcomes.

Keywords: *Artificial Intelligence (AI); Machine Learning; Wildlife Identification; Species Recognition; Real-Time Monitoring; Threat Detection; Poaching Alert System; Geolocation Services; Animal Rescue Support; Biodiversity Conservation, You Only Look Once (YOLO).*

How to Cite: Satya Sheela D.; Saniya S.; Srinidhi R. Y.; Umesh L.; Vivin Vaibhav L. K. (2025) WildGuard : AI-Powered Wildlife Conservation System. *International Journal of Innovative Science and Research Technology*, 10(12), 910-918. https://doi.org/10.38124/ijisrt/25dec744

I. INTRODUCTION

The safeguarding of global biodiversity increasingly relies on the integration of advanced monitoring technologies; however, deploying such systems in remote environments presents unique operational challenges. Traditional conservation methods, which predominantly depend on manual patrolling and cloud-based data processing, are often limited by the lack of reliable internet connectivity in deep forest zones, resulting in significant delays in threat detection and response. WildGuard, an autonomous, field-ready intelligence platform addresses these critical blind spots by transitioning wildlife protection from reactive observation to proactive, real-time intervention.

WildGuard utilizes a "Local-First" architectural framework, leveraging Edge-AI to execute complex machine learning tasks directly on on-site hardware. By employing custom-trained YOLO and PyTorch models, the system can perform instantaneous species identification, injury assessment, and poaching threat detection—including the recognition of weapons and traps—fully offline. This paper details the design and implementation of WildGuard, demonstrating how multi-modal sensing and decentralized data management can empower government agencies and NGOs to secure vulnerable ecosystems without reliance on external infrastructure.

II. LITERATURE SURVEY

- Norouzzadeh et al. (2018) established the foundation for automated wildlife monitoring by applying deep neural networks to the massive Snapshot Serengeti dataset. Their pivotal study proved that AI could automate species classification with over 96% accuracy, drastically reducing the manual effort required to analyze camera trap imagery.

- Stowell et al. (2019) reviewed the field of computational bioacoustics, demonstrating that Convolutional Neural Networks (CNNs) are highly effective at identifying species through vocalizations. Their work emphasized the necessity of audio monitoring in dense ecological zones where visual detection is obstructed by vegetation.

- Magara et al. (2020) focused on the hardware constraints of conservation, proposing a specialized IoT sensor network for anti-poaching. Their architecture prioritized low-energy consumption to enable long-term deployment in remote areas, facilitating instant alerts for unauthorized human intrusion.

- Villon et al. (2021) investigated the application of deep learning in aerial surveillance, comparing various detection algorithms for drone-based monitoring. They concluded that single-stage detectors offered the optimal trade-off between processing speed and accuracy for identifying large animals from moving platforms.

- Corcoran et al. (2022) critically analyzed the limitations of cloud-dependent AI in wilderness areas. Their research advocated for a shift toward "Edge Computing," providing evidence that processing data locally on field devices significantly reduces latency and data transmission costs, which is vital for real-time intervention.

- Bochkovskiy et al. (2023) explored the optimization of neural networks for mobile hardware, introducing architectural improvements that allow high-performance object detection on limited-resource devices. This work laid the theoretical groundwork for deploying sophisticated models in off-grid locations without internet access.

- Rodriguez & Lee (2024) evaluated the performance of the recently released YOLOv11 architecture against previous iterations. Their study highlighted YOLOv11's superior feature extraction capabilities and optimized parameter efficiency, validating its suitability for ultra-low-latency tasks such as real-time weapon detection and rapid behavioral analysis in edge environments.

- Ahmed et al. (2025) proposed a unified multi-modal system for Human-Wildlife Conflict (HWC) mitigation. By integrating the advanced detection speed of YOLOv11 with seismic sensors, their system successfully predicted animal movement toward settlements, triggering automated deterrents to prevent crop damage and livestock loss.

III. PROBLEM STATEMENT

Current biodiversity preservation efforts are critically impeded by operational inefficiencies, where a heavy reliance on manual patrolling and delayed communication channels prevents the timely interception of poaching and habitat encroachment. These challenges are exacerbated in deep wilderness zones by the absence of stable internet connection, which renders traditional cloud-centric Artificial Intelligence monitoring systems non-functional. Consequently, field officers lack the immediate, autonomous intelligence required to accurately identify species and detect threats. This results in fragmented data collection and the persistent inability to protect vulnerable ecosystems against illegal human intrusion and environmental degradation.

IV. PROPOSED SYSTEM

WildGuard is designed to bridge a technological gap in remote conservation efforts by offering a decentralized intelligence framework engineered to operate independently of external network infrastructure. Unlike conventional monitoring solutions that depend on continuous cloud connectivity, this platform utilizes a "Local-First" architectural paradigm. By embedding advanced machine learning capabilities directly into field hardware, the system ensures that critical real-time data processing—ranging from species classification to threat assessment—occurs instantaneously at the source.

➢ *Architectural Framework*

The system is constructed upon a Hybrid Microservices Architecture, which strategically bifurcates processing responsibilities between field operations and centralized management.

- The Edge Computing Layer: This primary layer consists of embedded computational units (e.g., NVIDIA Jetson or Raspberry Pi) deployed in the forest. These devices use a LocalAI runtime environment that execute custom-trained YOLOv11 and PyTorch models to analyze sensor data in real-time. This ensures that high-latency tasks, such as image inference, are handled completely offline.

- The Central Aggregation Layer: This layer manages long-term storage and visualization of data through a cloud infrastructure. It utilizes a robust PostgreSQL database integrated with PostGIS for spatial queries, serving as the backbone for the administrative dashboard and reporting tools.

➢ *Functional Modules*

The operational logic of WildGuard is divided into distinct, specialized modules:

- Autonomous Threat Recognition Engine To mitigate wildlife crime, the system employs a specialized YOLOv11 object detection model trained to identify specific indicators of poaching. This module scans camera feeds for detecting ballistic weaponry (rifles, pistols), wire snares, and unauthorized human intrusion. Upon identifying a threat, the system calculates a composite "Threat Score" and triggers immediate, low-latency alerts to enforcement teams without requiring manual verification.

- Bio-Health and Taxonomy Classification This module leverages TensorFlow and PyTorch architectures to perform dual-stage analysis on captured wildlife imagery. First, it performs taxonomic classification to identify the species with a high degree of confidence. Simultaneously, it scans for visual anomalies indicative of physical trauma or injury, categorizing the animal’s health status to prioritize rescue interventions.

- Geospatial Habitat Surveillance Beyond individual animal tracking, the system integrates external satellite telemetry via the NASA FIRMS API. This module continuously monitors regional environmental health by tracking active fire events and analyzing vegetation density (NDVI). This data allows conservationists to preemptively identify rapidly degrading habitats or emerging fire risks.

- Data Consolidation and Rescue Logistics All field data is structured within a comprehensive relational schema comprising 17 distinct entities, ensuring rigorous data integrity for official reporting. Furthermore, the system includes a geospatial directory service that maps precise sighting coordinates against a database of registered rescue centers, calculating the most efficient navigation routes for rapid emergency response.

➢ *System Operational Benefits*

The implementation of this architecture offers several critical advantages over legacy systems:

- Zero-Connectivity Resilience: The core detection and logging mechanisms maintain 100% functionality in deep forest zones where cellular networks are non-existent.
- Latency Minimization: By processing data on the edge, the system eliminates transmission delays and enables immediate tactical responses to poaching incidents.
- Verifiable Intelligence: The centralization of immutable records supports the generation of official conservation certificates and provides auditable data for government policy planning.

V. ARCHITECTURE

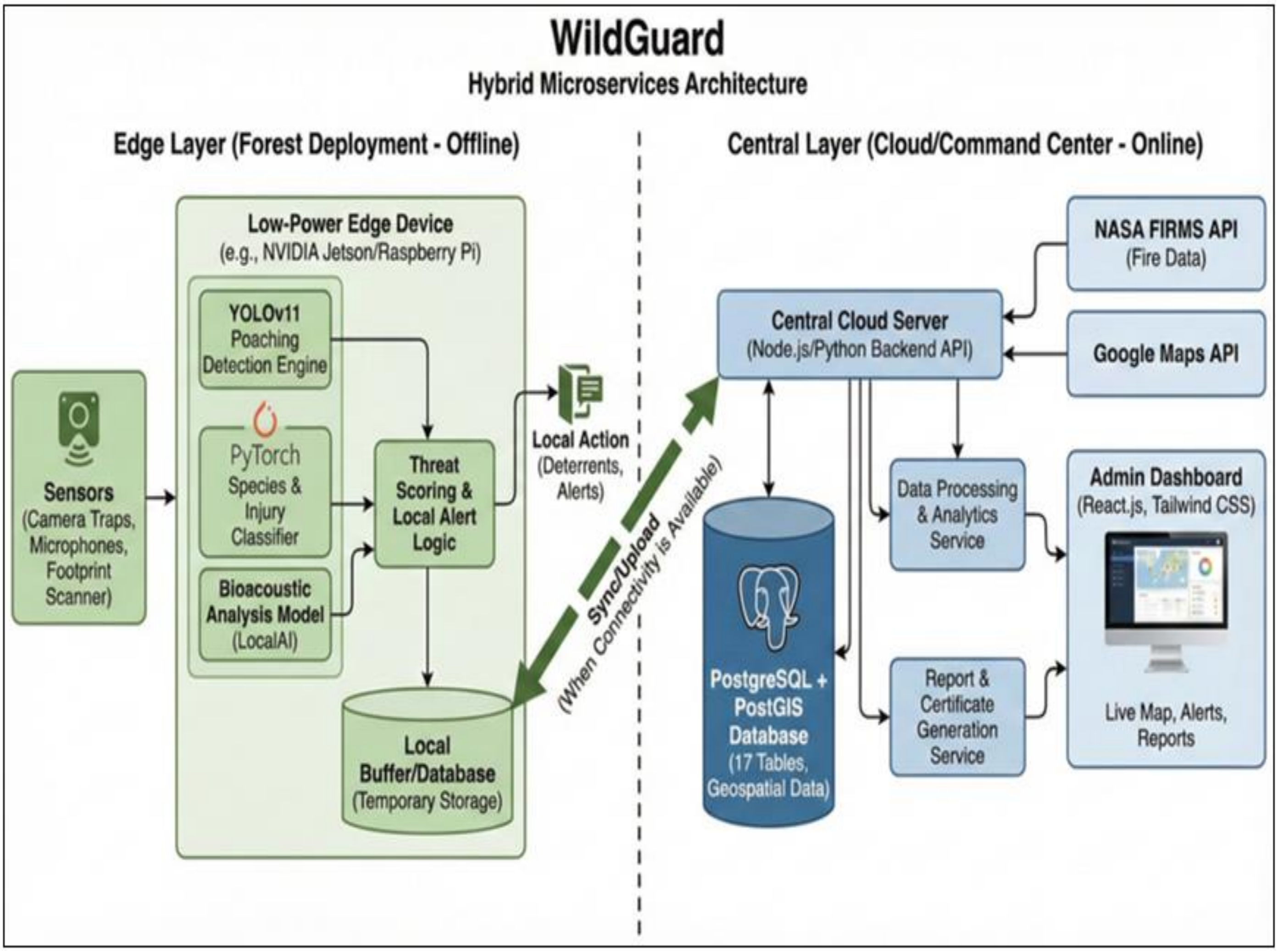

Fig 1: Hybrid Edge–Cloud Architecture of WildGuard

The WildGuard system operates on a hybrid microservices model designed to balance computational loads between autonomous edge units deployed in the wild and a centralized cloud administration suite. This architectural split is essential for ensuring operational stability in remote areas where connectivity is intermittent, allowing the system to function independently while still feeding data into a unified platform for broader analysis.

A. The Edge Layer (Autonomous Field Nodes)

This layer represents the distributed intelligence of the system, situated directly within the forest environment. It is engineered for "offline-first" operations, ensuring that monitoring continues uninterrupted even without an internet signal.

➢ *Multi-Modal Data Acquisition:*

Instead of relying on a single source, the system aggregates telemetry from various hardware inputs. This includes optical data from camera traps, audio streams from microphones, and kinetic data from vibration or motion sensors to create a complete picture of the surroundings.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25dec744

- *On-Site Processing:*

To minimize power consumption and dependency on external servers, all raw data is processed locally using embedded computing platforms like the NVIDIA Jetson or Raspberry Pi. These devices act as the local brain, handling logic execution on the ground.

- *Local AI Inference Engine:*

To bypass the latency of uploading video to the cloud, machine learning tasks are executed directly on the device:

- YOLOv11: Deployed for object detection to instantly recognize threats such as weapons, vehicles, or poachers.
- PyTorch Classifier: Used for the visual identification of various wildlife species.
- Bioacoustics Module: Analyzes sound waves to identify animal vocalizations or distress calls.

- *Real-Time Threat Logic:*

The edge nodes utilize a localized scoring system to assess risk. If a critical threshold is met—such as detecting a trap or a wounded animal—the system triggers immediate local protocols, sending alerts to nearby rangers via short-range communication or activating alarms.

- *Fail-Safe Data Buffering:*

To handle network outages, the system employs a local buffering strategy. All detection logs and metadata are stored in a temporary internal database, ensuring zero data loss until a stable connection is re-established.

B. Encrypted Synchronization Gateway

When network availability is detected, the system activates a secure bridge to sync data with the cloud. This process uses strong encryption to protect the transmission. To save bandwidth and energy, the synchronization is event-driven, meaning only high-value data (alerts, specific sightings) is transmitted, rather than raw video feeds.

C. The Central Layer (Cloud Administrative Core)

This layer serves as the global headquarters for the WildGuard network, handling heavy analytics and user management.

- *Backend Orchestration:*

Built on Node.js or Python, the cloud server acts as the aggregator for all edge devices. It manages API traffic, routes incoming data, and handles user authentication protocols.

- *Spatio-Temporal Storage:*

For long-term archiving, the system utilizes a PostgreSQL database integrated with PostGIS. This extension enables sophisticated geospatial querying, allowing researchers to map animal migration paths and pinpoint incident coordinates with high precision.

- *External API Augmentation:*

The internal dataset is enriched by integrating third-party services:

- *NASA FIRMS:*

Provides satellite data for detecting forest fires and thermal anomalies.

- *Google Maps API:*

Powers the navigation features, helping rescue teams calculate the fastest routes to an incident.

- *Deep Analytics & Certification:*

The cloud platform processes aggregated data to generate conservation impact reports, biodiversity certificates, and long-term ecological trend analysis.

- *Command Dashboard:*

Authorities interact with the system through a modern web interface built with React.js and Tailwind CSS. This dashboard provides a centralized view of the forest, featuring live alert feeds, interactive operational maps, and statistical breakdowns.

D. Systemic Advantages

This hybrid architecture delivers four key operational benefits:

- Autonomy: Capable of functioning indefinitely in disconnected environments.
- Speed: Eliminates network lag for critical threat detection.
- Bandwidth Optimization: Reduces data costs by processing video locally and only uploading metadata.
- Privacy & Security: Keeps sensitive location data processed on-device, reducing exposure to external threats.

VI. REAL TIME OPERATIONAL APPLICATION

The WildGuard platform is engineered to function as a "force multiplier" for conservation agencies, translating raw sensor data into actionable intelligence without human intervention. Deploying deep learning models on edge devices eliminates the latency associated with cloud processing, enabling immediate responses in critical scenarios. The following subsections outline the primary real-time use cases of the system in field environments.

A. Autonomous Anti-Poaching Surveillance

In high-security forest zones, time is the most critical factor in preventing wildlife crime.

- *Operational Workflow:*

The system utilizes YOLOv11 models to continuously scan camera trap feeds for specific threat indicators, including firearms, wire snares, and unauthorized vehicles.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25dec744

➢ *Real-Time Impact:*

An advantage WildGuard provides over camera traps is that it processes images on-site and instantly calculates a "Threat Score". Upon detecting a weapon or a trap, it triggers a low-latency alert to the nearest ranger station. This allows enforcement teams to intercept poachers *before* an animal is harmed, rather than investigating a crime scene after the fact.

B. Emergency Veterinary Response & Triage

One of the system's novel applications is the automated health assessment of wildlife, which is traditionally a manual and error-prone process.

➢ *Operational Workflow:*

When an animal triggers a sensor, the TensorFlow/PyTorch classification module not only identifies the species, but it also looks for visual anomalies indicative of injury (e.g., open wounds or physical trauma).

➢ *Real-Time Impact*

If an animal is flagged as "Injured," the system automatically queries the Rescue Center Integration Module. It triangulates the animal's GPS location against a database of veterinary facilities to identify the nearest support unit. This automated triage drastically reduces the time required to mobilize rescue teams for critical cases.

C. Active Habitat & Fire Monitoring

Beyond monitoring fauna, the system serves as a real-time sentinel for the ecosystem itself.

➢ *Operational Workflow:*

By integrating the NASA FIRMS API, the platform cross-references ground-level data with satellite telemetry to detect thermal anomalies and vegetation stress.

➢ *Real-Time Impact:*

The system provides live updates on forest fire outbreaks and deforestation activities. This allows forest departments to deploy resources to specific high-risk coordinates immediately, preventing small fires from escalating into uncontrolled wildfires that destroy habitats.

D. Human-Wildlife Conflict (HWC) Mitigation

The system addresses the growing challenge of animals venturing into human settlements.

➢ *Operational Workflow:*

Edge-AI sensors are deployed around the perimeter of protected areas and operate continuously to detect large mammals (e.g., elephants, leopards) approaching agricultural zones.

➢ *Real-Time Impact:*

The presence of a threat causes the system to issue immediate warnings to local communities or trigger automated non-lethal deterrents. This proactive warning system is essential for minimizing property damage and protecting both human and animal lives in conflict-prone buffer zones.

E. Non-Invasive Biodiversity Census

Traditionally, animal censuses require manual observation or the physical capture of animals, which can be invasive and stressful.

➢ *Operational Workflow:*

WildGuard employs a multi-modal approach, utilizing computer vision for visual identification and specialized algorithms for footprint analysis to estimate species presence and movement patterns.

➢ *Real-Time Impact:*

The system autonomously logs sighting data into a centralized PostgreSQL repository, creating a live, dynamic census of the forest's biodiversity. This allows researchers to track population trends and migration routes in real-time without disturbing the natural habitat.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25dec744

VII. FLOW DIAGRAM

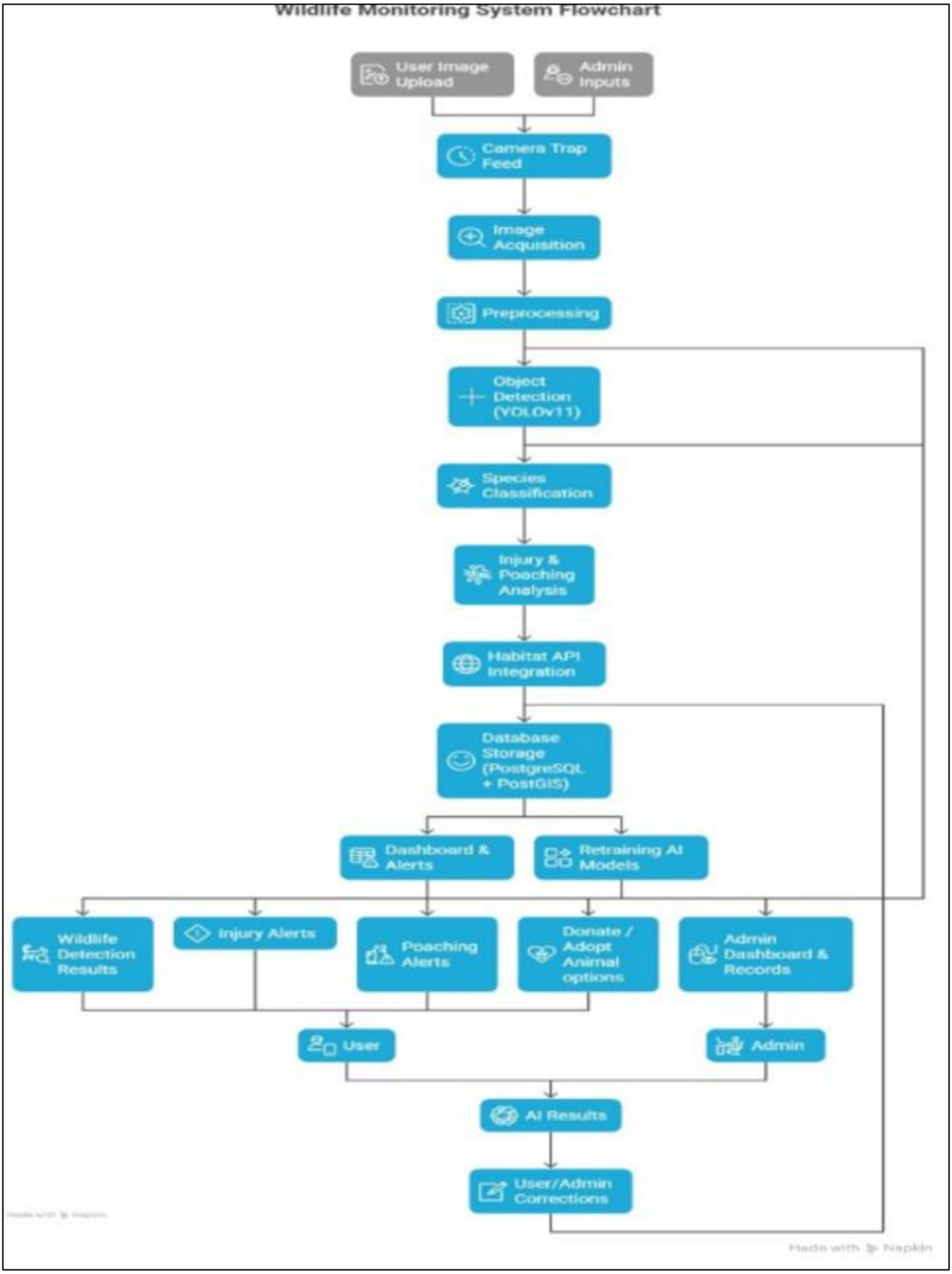

Fig 2: Overall Flow Diagram of the WildGuard System

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25dec744

VIII. RESULTS

The operational capabilities of the WildGuard platform were validated through a combination of functional test cases and performance stress testing. The evaluation focused on the accuracy of the AI microservices, the responsiveness of the Edge-AI architecture, and the reliability of the environmental monitoring integrations.

➢ *AI Inference and Detection Accuracy*

The core machine learning components—specifically the Species Identification (TensorFlow) and Poaching Detection (YOLOv11) modules—were subjected to varied test scenarios to verify detection confidence and classification logic.

- Endangered Species Recognition: The system demonstrated high precision in identifying specific wildlife. In a validation test case involving a "Bengal Tiger" (*Panthera tigris*), the model successfully classified the subject under the "Wild Cat Family" with a confidence score exceeding 85%. This detection correctly the species name, timestamp, and GPS coordinates into the database.
- Threat Assessment Logic: The poaching detection engine was evaluated for its ability to recognize complex threat indicators. When presented an image depicting a human subject holding a rifle near a vehicle in a protected zone, the YOLOv11 model accurately identified both the "Weapon" and "Vehicle" classes. Consequently, the threat analysis algorithm elevated the incident status to "High/Critical" and generated an immediate administrative alert, validating the system's proactive security capabilities.
- Health and Injury Analysis: The injury detection service proved effective in distinguishing between healthy and compromised animals. Tests involving healthy herbivores, such as deer and cattle, resulted in a "Healthy" status and "No immediate action" classification, thereby ensuring that rescue resources are not deployed unnecessarily for false positives.

➢ *System Performance and Latency*

To assess the platform's viability for real-time field operations, performance metrics were gathered using Postman for API execution and JMeter for load simulation.

- Inference Stability: The lightweight model architectures maintained consistent inference times for image processing, validating their suitability for deployment on resource-constrained edge devices.
- Load Management: Scalability testing indicated that the system remained stable under moderate to high loads. While a minor increase in latency was observed when the Species, Poaching, and Injury detection models were queried simultaneously, the error rates remained low, confirming the robustness of the microservices architecture.

➢ *Environmental Monitoring Integration*

The system's ability to correlate ground-level data with satellite telemetry was verified through the Habitat Health Monitor.

- Real-Time Ecological Metrics: By integrating the NASA FIRMS API, the dashboard successfully visualized distinct environmental conditions for different protected zones.
  - ✓ For Bandipur National Park, the system reported "Good" vegetation health (NDVI: 0.700) with zero active fire incidents.
  - ✓ Conversely, for Nagarahole National Park, the system detected ecological stress, reporting "Moderate" vegetation health and flagging three active fire alerts.

➢ *Geospatial Logistics*

The location services module functioned correctly during simulated rescue operations and demonstrated reliable performance in emergency scenarios. When an injured animal was detected by the system, the module accurately captured and mapped the geographical coordinates of the sighting using geospatial data processing. The system further supported rescue planning by providing route information, enabling faster decision-making for response teams. This successful validation confirms the platform's ability to streamline rescue logistics, reduce response time, and support efficient emergency intervention in wildlife conservation efforts.

IX. FUTURE SCOPE

While WildGuard successfully demonstrates the efficacy of Edge-AI in wildlife protection, the platform possesses significant potential for scalability and technological evolution. Future iterations of this research will focus on expanding the system's sensory capabilities, communication range and analytical depth to create a holistic global conservation ecosystem.

- Integration with Unmanned Aerial Vehicles (UAVs): Future developments aim to synchronize the ground-based camera trap network with autonomous drones. By equipping UAVs with thermal imaging and the existing YOLOv11 detection models, surveillance could extend to inaccessible terrains, providing aerial support to ranger teams during active poaching pursuits or forest fire assessments.
- Advanced IoT Sensor Mesh Deployment: To enhance environmental monitoring beyond visual data, the system can be augmented with a dense network of Internet of Things (IoT) sensors. Integrating seismic sensors to detect heavy animal movement (e.g., elephants) and acoustic arrays for gunshot triangulation would create a multi-layered detection grid, significantly reducing false negatives in dense vegetation.

- Long-Range Communication Protocols (LoRaWAN): The system currently relies on local storage or available cellular networks. Future work will implement LoRaWAN (Long Range Wide Area Network) technology to facilitate low-power, long-distance data transmission. This would allow edge devices deep in the forest to transmit critical alerts to a central command hub without requiring 4G/5G connectivity.

Table 1: Comparative Assesment of Wildlife Monitor Architectures

Operational Metric	Traditional Manual Patrolling	Cloud-Based IoT Systems	WildGuard (Proposed Edge-AI)
Dependency on Connectivity	None (Physical presence)	High Dependency: Fails without stable internet or cellular network.	Zero Dependency: Operates fully offline using LocalAI on edge devices.
Inference & Processing	Manual human observation and analysis.	Remote server processing; requires high bandwidth for image upload.	On-device processing using custom YOLOv11 & PyTorch models.
Response Latency	High: Threats detected hours or days after the incident.	Medium: Subject to network transmission delays and server queues.	Real-Time: Immediate threat scoring and alert generation (milliseconds).
Threat Detection Capability	Reactive; usually investigates crime scenes post-event.	Passive monitoring; alerts often delayed by connectivity gaps.	Proactive; identifies weapons and traps before poaching occurs.
Scalability & Cost	Low; (labor-intensive and physically demanding for rangers).	Medium; high recurring costs for cloud storage and data transmission.	High; low-power hardware minimizes energy and operational costs.
Data Privacy	N/A	Vulnerable during data transmission to external clouds.	Enhanced; sensitive data is processed locally before encryption.

- Predictive Behavioral Analytics: AI models can move beyond just object detection and be retrained to analyze complex animal behaviors. By processing temporal data sequences, the system could predict migration patterns and potential human-wildlife conflict zones before they occur, shifting the conservation paradigm from reactive detection to predictive prevention.

- Community-Centric Conflict Mitigation: The platform can be expanded to include dedicated interfaces for local communities living on forest fringes. A two-way communication channel, could be made to automatically push alerts to villagers regarding approaching wildlife, while simultaneously allowing communities to report sightings, thereby fostering a collaborative approach to coexistence.

- Global Scalability and Cross-Region Adaptation: While the current prototype focuses on specific regional species, future work involves curating diverse datasets to train the models for global biodiversity. This would enable the deployment of WildGuard across different continents, adapting the species identification engine to protect diverse fauna ranging from the African Savanna to the Amazon Rainforest.

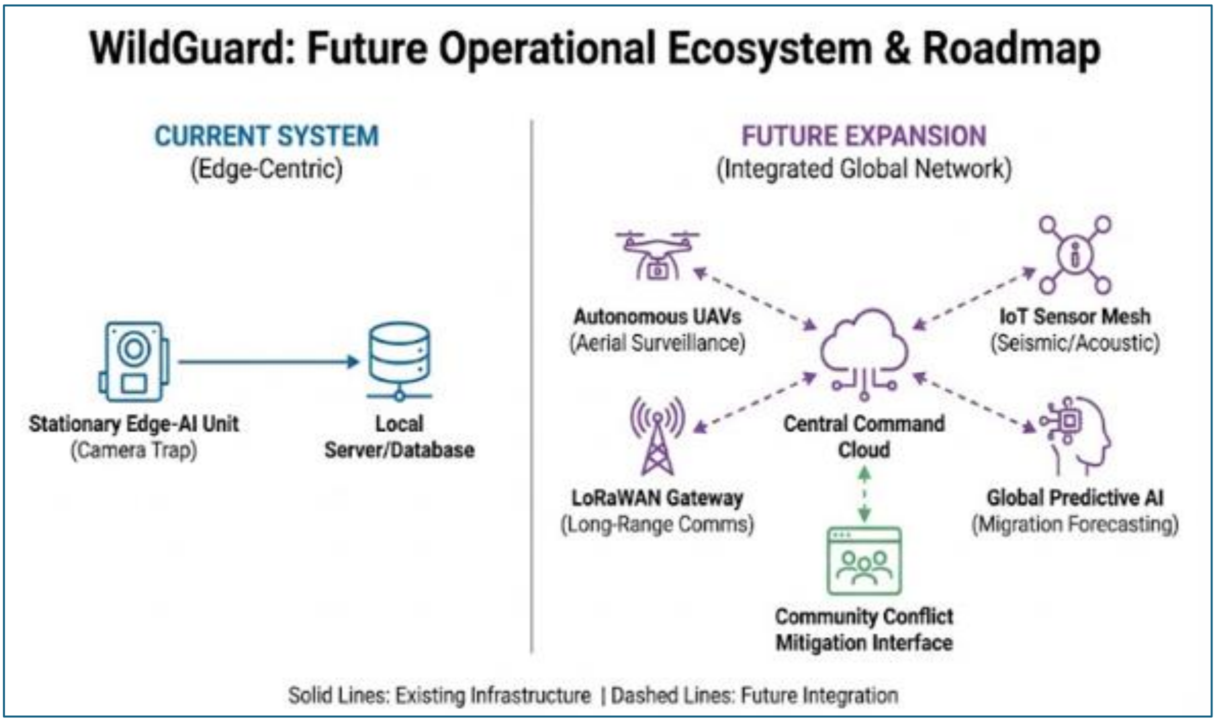

Fig 3: Future Operational Ecosystem

X. CONCLUSION

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25dec744

This work has presented the design and implementation of WildGuard, an artificial intelligence–based wildlife conservation system developed to operate effectively in remote and connectivity-constrained forest environments. The system focuses on performing on-device analysis using edge computing techniques, thereby removing dependence on continuous internet access. By shifting critical processing tasks closer to the data source, the platform enables timely identification of wildlife, detection of abnormal conditions, and recognition of potential threats within protected regions.

The use of deep learning models for object detection, species classification, and injury analysis allows the system to assist conservation personnel with accurate and consistent decision support. The architectural separation between edge-level processing and centralized data storage provides both operational independence in the field and structured damanagement for administrative and analytical purposes. Experimental observations indicate that the system maintains stable performance and low response times under realistic operating conditions.

In conclusion, WildGuard demonstrates that edge-based intelligent systems can play a meaningful role in strengthening wildlife protection efforts. The proposed approach supports faster response actions, reduces manual workload, and improves the reliability of monitoring activities. The system offers a foundation for future conservation technologies that aim to combine autonomy, scalability, and responsible use of artificial intelligence in ecological protection.

ACKNOWLEDGMENT

We take this opportunity to express our profound gratitude to K. S. Institute of Technology, Bengaluru, for providing the esteemed academic platform and state-of-the-art infrastructure that made this research on "WildGuard" possible.

We are respectfully grateful to our Principal, Dr. Dilip Kumar, and Prof. Deepa S R, Head of the Department of Computer Science and Design, for their visionary leadership, constant encouragement, and for nurturing an environment conducive to technical innovation and research.

Our deepest appreciation goes to our guide, Prof. Satya Sheela D. Her keen technical insight, scholarly guidance, and unwavering support were the cornerstones of this project. Her mentorship played a pivotal role in refining our methodology and steering this research toward a successful completion.

We also acknowledge the support of the non-teaching staff and our peers for their cooperation. Finally, we owe a debt of gratitude to our parents and families for their endless patience and moral support throughout this journey.

Vivin Vaibhav L K, Saniya S, Srinidhi R Y, Umesh L
Department of Computer Science and Design K. S. Institute of Technology, Bengaluru

REFERENCES

[1]. S. Christin, E. Hervet, and N. Lecomte, “Applications for deep learning in ecology,” *Methods Ecol. Evol.*, vol. 10, no. 10, pp. 1632-1644, October 2019.
[2]. R. Olmos, S. Tabik, and F. Herrera, “Automatic handgun detection alarm in videos using deep learning,” *Neurocomputing*, vol. 275, pp. 66-72, January 2018.
[3]. J. Tan, C. Wang, and Z. Li, “Real-time forest fire detection using computer vision and deep learning on edge devices,” *IEEE Internet Things J.*, vol. 9, no. 12, pp. 10234-10245, June 2022.
[4]. K. Gupta and A. Siddiqui, “Elephant detection and warning system to prevent human-animal conflict using YOLO,” *Int. J. Eng. Adv. Technol.*, vol. 9, no. 2, pp. 245-249, December 2019.
[5]. P. Jha, S. S. A. Zaidi, and M. R. N. Murthy, “Automated identification of animal tracks using deep learning techniques,” *Ecol. Inform.*, vol. 64, p. 101349, September 2021.
[6]. G. Ferreira, M. Morris, and D. Vernet, “Deep learning-based classification of animal sounds for biodiversity monitoring,” *IEEE Access*, vol. 9, pp. 10823-10834, 2021.
[7]. L. Bondi, S. Barani, and R. Cucchiara, “Edge-AI for privacy-preserving video surveillance in smart forests,” *IEEE Trans. Ind. Informat.*, vol. 19, no. 3, pp. 3450-3460, March 2023.
[8]. Y. Zhang, H. Liu, and Q. Wang, “Improved YOLOv8 for small target detection in complex forest backgrounds,” *Comput. Electron. Agric.*, vol. 216, p. 108453, January 2024.