ACKNOWLEDGEMENT

It gives us immense pleasure to present our group project**, "GPS Tracker for Animals,"** undertaken as part of the sixth semester curriculum at Purbanchal University.

We extend our sincere gratitude **to Khwopa Engineering College** for providing us with this valuable opportunity and the necessary platform to work on this innovative project. Our heartfelt thanks go to the Head of the Electronics, Communication and Automation Department, **Er. Yogesh Bajracharya**, for his guidance and encouragement, and to our respected supervisor **Er. Ganesh Ram Dhonju** for his unwavering support, insightful feedback, and mentorship throughout the development of this project.

We are also deeply thankful to all who contributed to the successful completion of this project proposal. Special appreciation goes to our team members for their dedication, collaboration, and relentless efforts in researching, analyzing, and developing the proposal. Each member's unique skillset and commitment played a vital role in shaping the final outcome.

We also acknowledge the valuable suggestions and feedback from our friends, which significantly enhanced the clarity and quality of our work. Our project was enriched by a wide range of research materials, academic literature, and online resources, which served as foundational references and sources of inspiration.

Lastly, we express our heartfelt gratitude to our families for their constant support, patience, and encouragement, which kept us motivated throughout this journey.

Group Members:

Dhiraj Mandal (780405)

Nisha Thapa Magar (780407)

Rabisha Machamasi (780410)

ABSTRACT

This report presents the progress on the "GPS Tracker for Animals," a project aimed at developing an affordable and efficient system for real-time monitoring of animals. The system utilizes GPS technology for location tracking and LoRa for long-range, low-power communication, with GSM as a backup. It addresses the challenges of traditional animal monitoring methods by offering continuous, real-time data on animal movement, behavior, and environmental conditions. The project incorporates features like geofencing and aims to integrate alarm system if any emergency occurs or the tracked animals went out of the dedicated range or area to enhance its capabilities for wildlife conservation, livestock management, and pet tracking. The current progress includes the successful implementation of both transmitter and receiver sides along with Initializing LoRa module code, verified GPS data transmission, and the development of a map-based interface with geofencing capabilities and data storage in a database. This system is expected to provide enhanced wildlife monitoring, improved disaster response, and data-driven conservation efforts, fostering community engagement and offering a scalable technological framework for broader ecological applications.

Keywords: *GPS, LoRa, Animal Tracking, Real-time Monitoring, Conservation, Geofencing, ESP32*

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER 1: INTRODUCTION

1.1 Background

GPS tracking for animals involves using Global Positioning System (GPS) technology to monitor and record the movements and behaviors of animals in real time. This is accomplished by attaching a compact, GPS-enabled device to the animal, which uses satellite signals to determine its precise location. The recorded data is then transmitted via lora to a central server, where it can be accessed and analyzed by researchers, conservationists, and wildlife managers. Recent advancements in GPS modules, wireless communication technologies (such as GSM, Wi-Fi, and LoRa), and microcontroller systems have enabled the development of cost-effective, compact, and energy-efficient tracking solutions. These systems often incorporate features like geofencing, which sets virtual boundaries and triggers alerts if an animal strays outside a predefined area—helping prevent wildlife from entering human settlements or hazardous zones.

LoRa (Long Range) communication technology has emerged as a game-changer in the Internet of Things (IoT) ecosystem. Designed for low-power, long-distance wireless communication, LoRa is ideal for scenarios where cellular or Wi-Fi connectivity is unavailable or impractical—such as deep forest environments, mountainous regions, or conservation zones. Unlike GSM-based systems that rely on existing telecom infrastructure (and incur data costs), LoRa operates over unlicensed radio frequencies (e.g., 433 MHz) and supports peer-to-peer or gateway-based communication with significantly lower power consumption, making it highly suitable for remote wildlife tracking**.**

Despite the growing use of GPS technology in wildlife research, several challenges persist, including battery life limitations, durability in harsh environmental conditions, and maintaining reliable data transmission in remote areas. These issues continue to drive innovation in tracker design to ensure minimal impact on animals while maximizing data accuracy and system longevity.

Nevertheless, GPS tracking has become an indispensable tool in modern conservation science. It enhances our ability to protect biodiversity, support species recovery programs, and manage animal populations sustainably. As the technology evolves, its applications are expanding—bridging the gap between scientific research, real-time monitoring, and effective wildlife management for the health of ecosystems worldwide.

1.2 Statement of Problems

Tracking the real-time location of objects, individuals, or animals has become increasingly important across numerous sectors such as transportation, wildlife conservation, personal safety, and asset management. In logistics, GPS tracking enhances efficiency through route optimization and fleet monitoring, while in conservation, it enables researchers to observe the movement and behavior of endangered species. Personal safety applications, such as tracking devices for children or the elderly, also rely on accurate location data. However, traditional GPS tracking systems depend heavily on cellular networks or Wi-Fi for data transmission. These systems work well in urban areas but struggle in remote, rural, or forested regions where network infrastructure is limited or nonexistent. This lack of reliable connectivity poses significant challenges for critical applications like wildlife tracking, disaster response, and remote area monitoring.

To address these limitations, there is a growing interest in integrating GPS with alternative wireless technologies such as LoRa (Long Range). LoRa enables low-power, long-distance communication without the need for cellular infrastructure, making it highly suitable for deployment in isolated or challenging environments. By combining GPS modules with LoRa and power-efficient microcontrollers like the ESP32, it becomes possible to design lightweight, energy-efficient tracking devices that can transmit location data across several kilometers. These systems offer reliable performance in areas lacking cellular coverage and are particularly beneficial for applications such as animal tracking in forests, equipment monitoring in rural farms, and safety systems for field personnel. As a result, LoRa-based GPS tracking solutions present a cost-effective, scalable, and sustainable alternative to conventional tracking methods, extending the reach of real-time monitoring to places previously considered unreachable.

1.3 Objective

1.3.1 Main Objective

To design and implement a low-cost, real-time GPS tracking system for monitoring animal locations over long distances using LoRa technology, with visual representation on a map and data logging capabilities.

1.3.2 Specific Objectives

- Interface ESP32 with NEO-6M GPS module to collect real-time location data.
- Establish long-range communication using LoRa modules for transmitting GPS data.

- Create a live tracking dashboard using Node-RED to display animal movements on a map.
- Implement geofencing to alert when animals cross predefined boundaries.
- Store GPS data in an SQL database for easy access and analysis.
- Optimize the system for low power use and real-time performance.

1.4 Scope and Limitations

The GPS tracker has become an indispensable tool for real-time location monitoring, offering critical functionality across a broad spectrum of applications. From personal safety devices that allow families to monitor children or elderly members, to sophisticated fleet management systems that optimize routes and track vehicle movements, GPS technology enhances situational awareness and operational efficiency. In wildlife conservation, GPS trackers enable researchers to study animal behavior, migration patterns, and habitat use with minimal intrusion. In the industrial sector, they play a key role in asset tracking and logistics by providing real-time updates on the movement of goods. With the integration of modern communication modules such as LoRa and ESP32 microcontrollers, GPS trackers can transmit location data wirelessly over long distances to centralized servers. This allows users to access real-time updates via mobile apps or computer dashboards. Moreover, advanced trackers can be equipped with additional sensors to monitor vital metrics such as temperature, humidity, speed, heart rate, or even motion—making them highly valuable in agriculture, healthcare, environmental monitoring, and smart city applications.

Despite their versatility and widespread adoption, GPS trackers are not without limitations. One major challenge is their reliance on satellite signals, which can be significantly weakened or blocked-in environments like tunnels, urban canyons, dense forests, or inside buildings—resulting in poor accuracy or total loss of signal. Another critical issue is power consumption; GPS modules and communication hardware require substantial energy, making it difficult to sustain long-term tracking in remote locations where recharging or battery replacement is not feasible. Security and privacy also present major concerns, as unauthorized interception or access to tracking data can lead to serious misuse or breaches of personal or sensitive information. These challenges highlight the need for ongoing innovation in energy-efficient design, signal processing, secure data transmission, and hybrid tracking approaches to ensure reliable, safe, and long-lasting GPS tracker performance.

CHAPTER 2: LITERATURE REVIEW

2.1 Literature Review

2.1.1 GPS based vehicle tracking and monitoring system

The rapid development and integration of Global Positioning System (GPS) technology have transformed various industries, enabling precise location tracking in real-time. GPS tracking systems have become ubiquitous in applications ranging from fleet management to personal safety devices, with considerable advancements in hardware and software technologies in recent years. One of the core technologies enabling such systems is the GPS module, which receives signals from satellites and determines the exact location of an object. These modules are typically low-cost, power-efficient, and compact, making them ideal for use in a wide range of applications, including transportation and animal tracking.

2.1.2 Tracking Wildlife for Conservation Research

A solution for public transportation. It provides solution for tracking and monitoring the public transportation vehicles using devices such as esp32 and GPS Antenna. ESP32 processing board can be used to receiving values and gives the result. This method can find a way to monitor the transportation vehicle from the location source to destination. In this paper, there is a use of GPS receiver module for receiving the latitude and longitude values of the present location of the vehicle continuously. A passenger of the vehicle will give different locations to the system between the source and destination locations. These values will be stored in the esp32 database and it further reciprocates accordingly. [1]

2.1.3 Android app-based vehicle tracking using GPS and GSM

Unlike the GPS in your mobile phone, tracking wildlife with GPS also requires location data to be transmitted back to a satellite. This two-way communication allows for real-time location data and tracking in remote areas. To make use of this technology, wildlife is fitted with a GPS system, usually a collar, so that their location and movements can be documented. Once downloaded onto a computer, the data can then be placed on maps. One of the advantages to GPS systems is that it can be set to collect points for a full 24-hour period and a biologist does not need to be in the field. Wildlife SOS is presently making use of GPS technology to track a herd of elephants in central Chhattisgarh. Wildlife SOS can track the herd remotely and in ‘real time’. This active monitoring allows our early warning alert system (EWAS) to be effective by warning locals when the elephants are approaching their farmlands. [2]

2.1.4 IOT based animal health monitoring and tracking system using ZIGBEE

Research paper has explained an embedded system, used to know the location of the vehicle using technologies like GSM and GPS. System needs closely linked GPS and GSM module with a microcontroller. Initially, the GPS installed in the device will receive the vehicle location from satellite and store it in a microcontroller ‘s buffer. In order to track location, the registered mobile number has to send request, once authentication of number get completed, the location will be sent to mobile number in the form of SMS. Then GSM get deactivated and GPS get activated again. The SMS consist of latitude and longitude value of vehicle. This value received in the SMS can be viewed via android app and this coordinate will be plotted in the app automatically. [3]

2.1.5 Animal Tracking System using LoRa Technology

The LoRa end node, tied to the animal sends the location information and temperature values to the server via gateway. The end node consists a rechargeable LoRa end node tied to the animal ISSN End node. Experiment was carried out to check the range and measure the surrounding temperature. During the experiment, the gateway was kept at certain height of a building and end node was moved away from gateway keeping Spreading Factor (SF) value fixed at a time. Various range values for different spreading factors with delays are depicted from the test cases it was conferred that as SF increases the range increases and also a delay increase. In this project SF value of 7 is used as it has the minimum delay of 1 minute. The Android app is used for displaying the location and temperature of the animals. The location gets updated when the animal moves to other location as well as temperature also gets updated. Practically, it was observed from the test cases that in urban area, when the gateway was kept at nearly 30 m above the ground, total range of 250 m radius was achieved with maximum delay of 4 minutes. [4]

CHAPTER 3: METHODOLOGY

3.1 Block Diagram

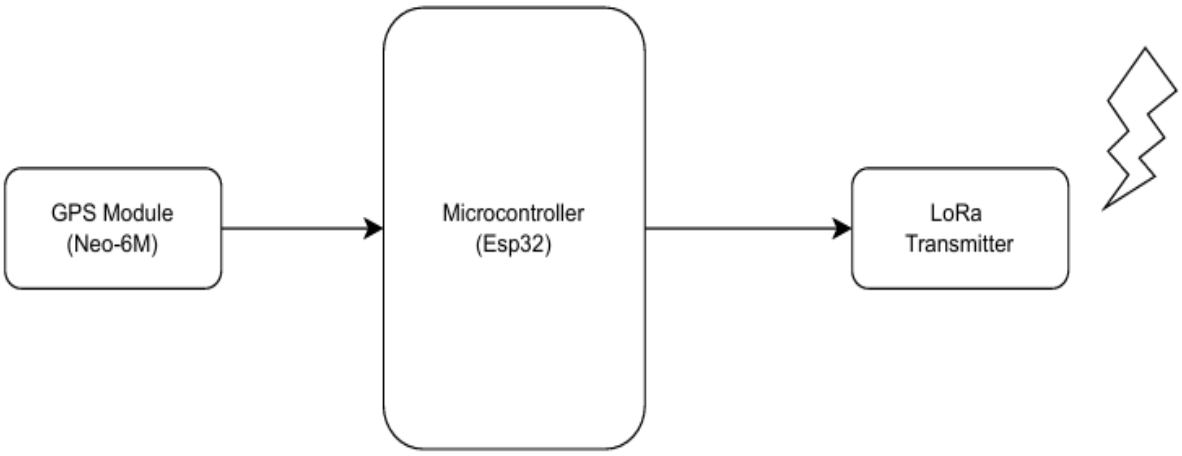


Fig 3.1.1: Transmitter

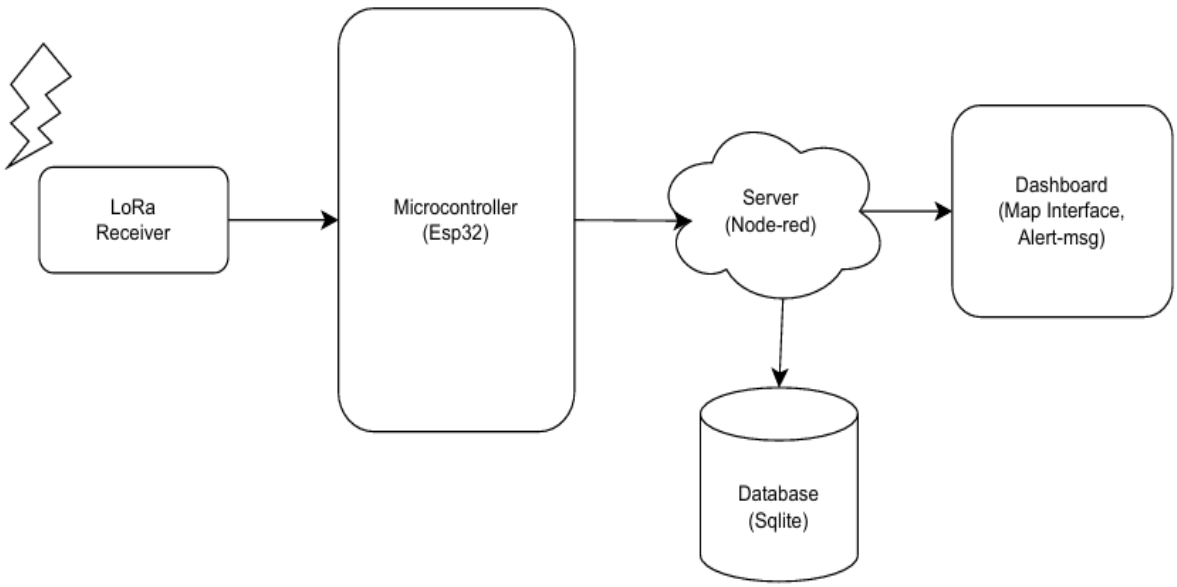


Fig3.1.2: Receiver

Transmitter Side: -

- **GPS Signal**

The GPS antenna receives signals from satellites and passes them to the GPS module (Neo-6M), which processes this signal to generate location data (latitude, longitude, etc.). The GPS module sends the formatted data to the esp32 microcontroller using SPI protocol.

- **Microcontroller**

The Neo-6M GPS module outputs NMEA sentences over SPI. Using the *TinyGPS++* library it extracts clean structured data. The ESP32 then packages the parsed data into a json format.

- **LoRa Transmitter**

The formatted data is transmitted via LoRa (SX1278) module wirelessly using radio waves (433 MHz).

Receiver Side: -

- **LoRa Receiver**

The formatted data is transmitted via LoRa (SX1278) module wirelessly using radio waves (433 MHz).

- **Microcontroller**

It receives GPS data from LoRa receiver via SPI. It forwards the data to a local server (Node-red) over WIFI using http.

- **Server (Node-red)**

It acts as a middleware platform. It receives GPS data from receiver esp32, parses and processes the data. It sends data to database (SQLite) for storing data and dashboard for visualization and alerts.

- **Database (SQLite)**

It is light weighted local database. It stores timestamped GPS data for retrieval and history tracking.

- **Dashboard**

It is built with node-red's dashboard nodes. It shows: -

SNNR: - It shows signal to noise ratio of the transmitted and received signal over the LoRa. This is showed in gauze.

RSSI: - It measures the power level of a received radio signal. The connection is said to be strong if the RSSI is from 30db to 70db.

Altitude: - The altitude chart is shown in dashboard tracking change in altitude.

Alert: - Alert message is shown if the tracker exits the geofence.

Map: - The world map is shown with the received latitude, longitude in map. A geofence is also mapped here.

GPS Table: - It includes received longitude, latitude, altitude, SNR, RSSI.

3.2 Tools and Platforms

3.2.1 Tools Used

1. Arduino IDE:

Used to write, compile, and upload C++ code to the ESP32 microcontroller. It supports various libraries required for GPS and LoRa functionalities. Also used to configure serial communication for debugging. Essential for integrating all hardware components programmatically. Code and configure ESP32, GPS, LoRa.

2. Node-Red:

A flow-based development tool for wiring together hardware and software. Handles backend logic, data processing, and event-driven alerts. Visual interface simplifies complex logic without deep coding. Integrates well with sensors, serial ports, and databases.

3. SQLITE:

Lightweight, serverless database for local storage of GPS data. Efficient and reliable for embedded or edge applications. Used to log historical tracking data. Easily integrates with Node-RED through dedicated nodes.

4. Arduino Serial Monitor:

Debugging tool within Arduino IDE to view real-time serial output. Helps verify GPS data, LoRa messages, and sensor readings. Crucial for testing communication between modules. Enables easy troubleshooting during development.

3.2.2 Platforms

Web Application:

Provides a visual interface for real-time GPS tracking. Allows users to interact with the map and view device status. Built using dashboards and world map nodes in Node-RED. Enables remote monitoring from any device with internet access.

3.2.3 Programming Languages

1. C++:

Used for programming the ESP32 microcontroller. Handles GPS parsing, LoRa transmission, and sensor interfacing. Efficient and low-level, suitable for embedded systems. Offers access to hardware-level operations and control.

2. JavaScript:

Used in Node.js environment for server-side scripting. Processes incoming GPS data and handles database interactions. Enables creation of dynamic, interactive web content. Plays a key role in backend logic and dashboard updates.

3.2.4 Libraries

TinyGPS++:

Parses raw NMEA data from GPS modules. Provides latitude, longitude, speed, and other details. Lightweight and efficient for embedded applications. Simplifies GPS data extraction for use in code

LoRa.h:

Facilitates communication between LoRa modules. Provides functions to send and receive packets. Supports long-range, low-power data transmission. Eases integration of LoRa into ESP32 projects.

Hardware Serial:

Allows multiple serial port handling on ESP32. Used for communication with GPS or LoRa modules. Supports simultaneous debugging and data transfer. Essential for managing devices over different serial channels.

SQLite:

Database library for managing local GPS data logs. Used within both embedded and Node-RED environments. Supports fast queries and structured data storage. Ideal for lightweight tracking systems without internet reliance.

Node-red-contrib-we-worldmap:

Displays real-time GPS data on an interactive world map. Allows tracking of movement and geolocation visually. Simple integration with Node-RED data flows. Useful for fleet or animal tracking applications.

Node-red-dashboard:

Creates interactive UI elements like charts, maps, and buttons. Used for real-time status display and control panels. Customizable layout for different user needs. Integrates directly into Node-RED environment.

Node-red-node-serialport:

Enables reading/writing data from serial ports in Node-RED. Facilitates communication with GPS and LoRa modules. Useful for collecting real-time data from microcontrollers. Supports multiple serial devices simultaneously.

Node-red-node-sqlite:

Manages SQLite database interactions within Node-RED. Allows storing, querying, and updating GPS logs. Supports historical data tracking and analytics. Ensures smooth flow from device data to database.

3.2.5 Hardware Used

ESP32:

The **ESP32** is a powerful microcontroller with dual-core processing, built-in Wi-Fi, Bluetooth, and multiple serial interfaces, making it highly suitable for IoT and real-time communication projects. It was chosen over alternatives like Arduino Uno or Nano due to its higher processing speed, larger memory, and built-in wireless capabilities, which eliminate the need for extra communication modules. Unlike Raspberry Pi, which is more power-hungry and overqualified for simple embedded tasks, the ESP32 offers a perfect balance between performance and power efficiency. Its versatility and cost-effectiveness make it ideal for compact GPS tracker applications.

NEO-6M GPS Module:

The **NEO-6M** GPS module provides reliable and accurate geolocation data with a quick fix time under open skies. It was selected over more advanced (and expensive) options like the NEO-M8N because it meets the accuracy and update rate requirements of this project at a lower cost. It also has lower power consumption and is widely supported with libraries like TinyGPS++, making development and integration straightforward. Its performance is sufficient for applications like animal tracking, vehicle monitoring, or environmental sensing where sub-meter precision is not critical.

LoRa Module:

The **LoRa module** enables low-power, long-range wireless data transmission, ideal for remote tracking where cellular networks may not be available. It was chosen over GSM modules (like SIM800L) because LoRa does not require SIM cards or monthly fees and works in unlicensed ISM bands. Compared to Zigbee or Wi-Fi, LoRa offers significantly longer range and better penetration in outdoor or forested environments. Its low energy consumption also makes it suitable for battery-operated devices deployed in the field for long periods.

CHAPTER 5: WORK COMPLETED

5.1 Work Done

1. Research and Findings: Extensive research on LoRa and GPS modules led to the selection of SX1278 LoRa module (cost-effective, 10-15km range, available in Nepal) and NEO-6M GPS module (affordable, ~2.5m accuracy, low power consumption).

2. Procurement of Products: SX1278 LoRa Module, NEO 6m GPS Module, and ESP-32 have been procured.

3. Initializing LoRa Module Code: Basic LoRa module code has been initialized and tested, demonstrating packet transmission with output showing "Packet Sent!". The LoRa module was initialized at 915MHz (with a note for region-based frequency change) and configured with low transmit power (5 dBm) for testing.

4. Initializing 6m GPS Module Code: GPS module code has been initialized and tested, successfully acquiring latitude, longitude, speed, altitude, satellite count, and UTC time.

5. Transmitter Side Development: The transmitter side has been successfully built using the LoRa module. It combines GPS data acquisition and LoRa transmission, sending data like "Lat:27.668390Lng:85.466171".

Transmitted Packet: -

Sending Lora Packet:

Packet sent #1000 Lat:27.668087 Lon:85.466110 Alt:1366.80m

Packet Sent! Size:49 bytes

6. Receiver Side Development: The receiver side has been successfully built using the LoRa module. The ESP32-based receiver was configured to send received GPS data. Received data such as "Pkt#232 Lat: 27.668087 Lon: 85.466110 Alt:1366.80m" has been verified.

Received Packet: -

Waiting for packets...

Received Packet (Size: 49 bytes): Packet sent #1000 Lat:27.668087 Lon:85.466110 Alt:1366.80m (RSSI: -84 dBm, SNR: 10.25 dB)

7. Map Based Interface: To provide a user-friendly and real-time visualization of the GPS data, a map-based interface was developed using **Node-RED**. The incoming GPS coordinates (latitude and longitude) received via LoRa are parsed and forwarded through a Node-RED dashboard. These coordinates are dynamically plotted on an embedded map using the **World Map** node, allowing users to track the live location of the animal or asset on a geographical

interface. This interface not only improves usability but also facilitates immediate understanding of movement patterns and position history without requiring manual coordinate interpretation.

8. Database Integration: To ensure that all tracking data is preserved for analysis, historical logging, and offline access, the received location data is stored in a **SQLite** database. Node-RED processes the incoming packets, extracts relevant fields such as latitude, longitude, altitude, timestamp, and packet number, and inserts them into the database. This implementation enables the development of additional features like playback of historical paths, trend analysis, and performance tracking. The use of SQLite provides a lightweight yet powerful embedded solution that requires no separate server setup, making it ideal for edge devices.

9. Geofencing Implementation: A geofencing feature was implemented by defining a virtual square boundary on the map. The application continuously checks whether the received coordinates fall within or outside the predefined area. When the tracked object or animal crosses this boundary, a trigger is generated in Node-RED, which can be configured to send alerts such as pop-up notifications, logs, or even SMS/email messages using third-party services. This geofencing mechanism enhances security and control, making the system suitable for use cases such as preventing wildlife intrusion into human settlements or securing valuable mobile assets.

5.2 Snapshots

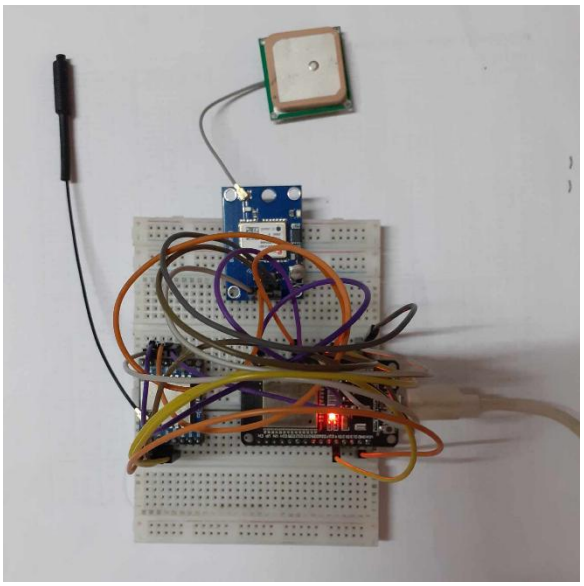


Fig 5.2.1: Transmitter

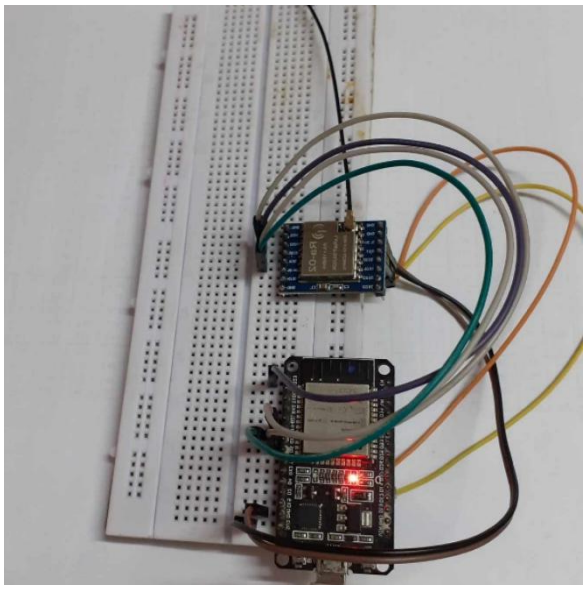


Fig 5.2.2: Receiver

```
11:11:52.688 -> Packet Sent! Size: 49 bytes
11:11:52.688 ->
11:11:52.688 -> Sending LoRa packet:
11:11:52.688 -> Pkt#3099 Lat:27.668087 Lon:85.466110 Alt:1366.80m
11:11:52.771 -> Packet Sent! Size: 49 bytes
11:11:52.771 ->
11:11:52.771 -> Sending LoRa packet:
11:11:52.771 -> Pkt#3100 Lat:27.668087 Lon:85.466110 Alt:1366.80m
11:11:52.841 -> Packet Sent! Size: 49 bytes
11:11:52.841 ->
```

Fig 5.2.3: Transmitted Data

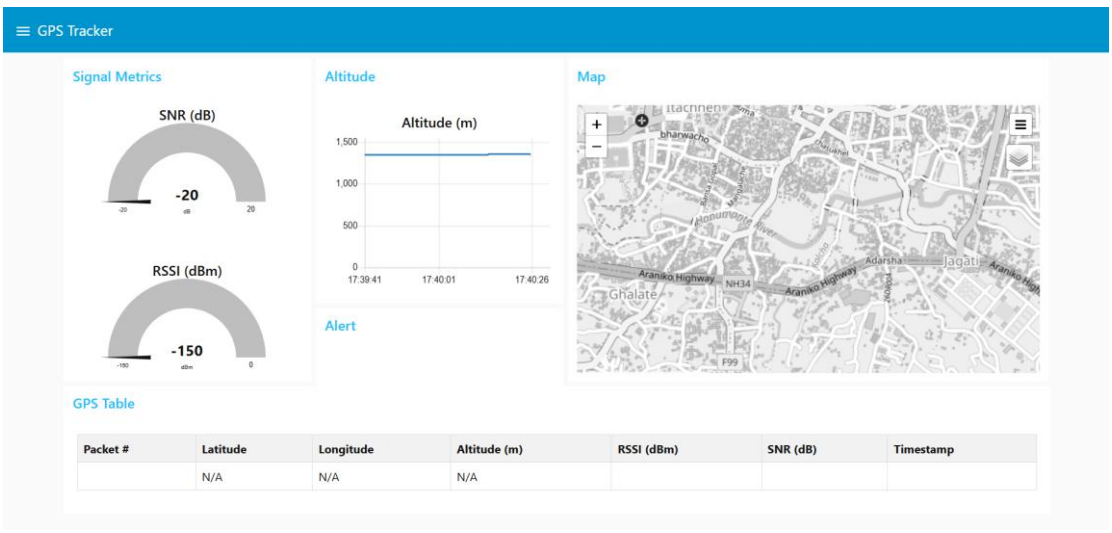


Fig 5.2.4: Map Interface

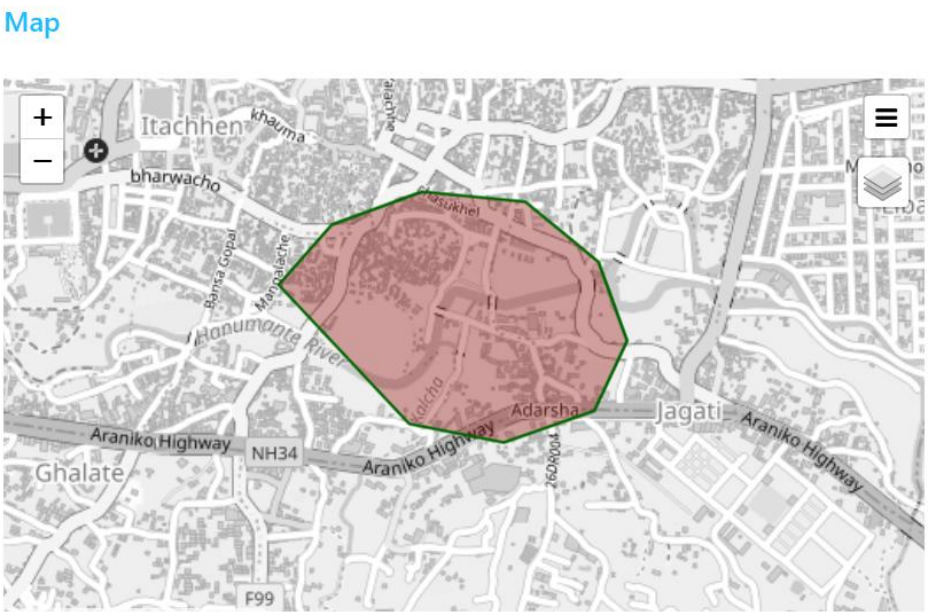


Fig 5.2.5: Geofence

Fig 5.2.6: Node-Red Dashboard

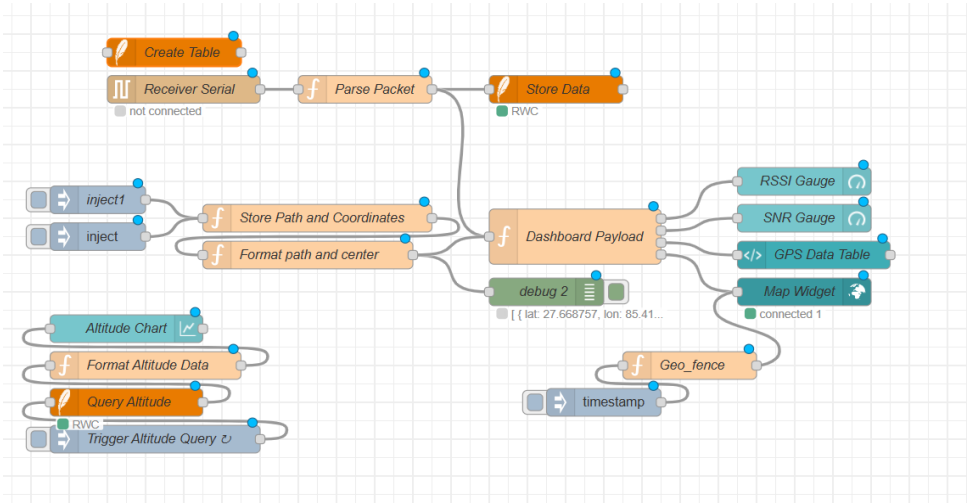
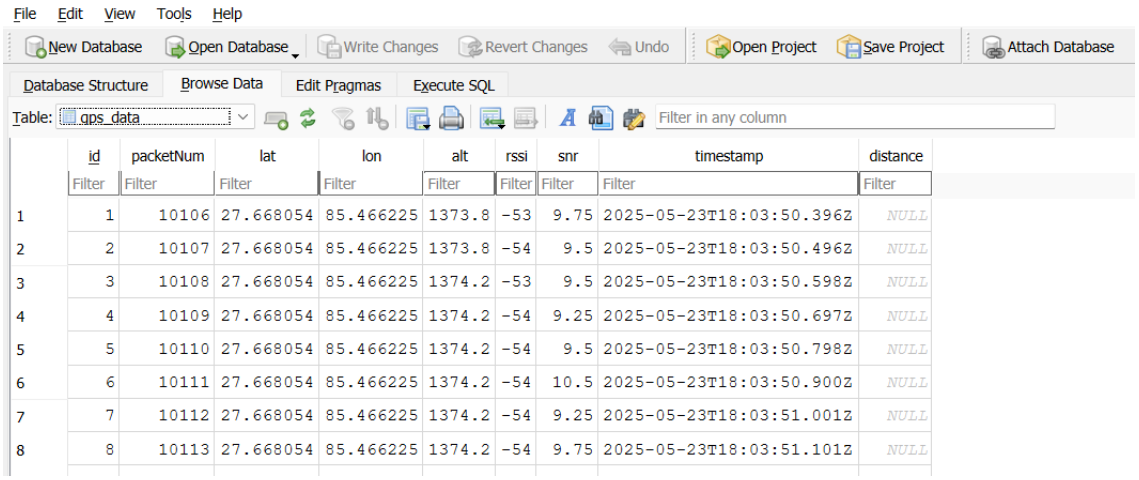


Fig 5.2.7: Node-Red Flow Diagram



	id	packetNum	lat	lon	alt	rssi	snr	timestamp	distance
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	10106	27.668054	85.466225	1373.8	-53	9.75	2025-05-23T18:03:50.396Z	NULL
2	2	10107	27.668054	85.466225	1373.8	-54	9.5	2025-05-23T18:03:50.496Z	NULL
3	3	10108	27.668054	85.466225	1374.2	-53	9.5	2025-05-23T18:03:50.598Z	NULL
4	4	10109	27.668054	85.466225	1374.2	-54	9.25	2025-05-23T18:03:50.697Z	NULL
5	5	10110	27.668054	85.466225	1374.2	-54	9.5	2025-05-23T18:03:50.798Z	NULL
6	6	10111	27.668054	85.466225	1374.2	-54	10.5	2025-05-23T18:03:50.900Z	NULL
7	7	10112	27.668054	85.466225	1374.2	-54	9.25	2025-05-23T18:03:51.001Z	NULL
8	8	10113	27.668054	85.466225	1374.2	-54	9.75	2025-05-23T18:03:51.101Z	NULL

Fig 5.2.8: Fetched database from the receiver

5.3 Remaining Works

One of the key remaining tasks is optimizing power consumption on the transmitter side by implementing **sleep mode functionality**. Since GPS trackers deployed in remote or outdoor environments often rely on battery power, reducing energy consumption is essential for prolonged operation. The plan involves programming the microcontroller (ESP32) to enter deep sleep mode between transmissions, thereby significantly lowering its current draw when idle. In this mode, the GPS module and LoRa transceiver will remain off or in low-power standby, waking periodically at predefined intervals to acquire and transmit location data. This enhancement is crucial for increasing the tracker's operational lifespan, especially in scenarios where frequent battery replacement or recharging is impractical.

Another important area of development is **manual location control**, which involves configuring the system to allow the user to trigger location transmissions manually when needed. This functionality can be useful during field testing or in situations where automatic periodic updates are not necessary or need to be overridden.

Additionally, the design and fabrication of a custom **PCB (Printed Circuit Board)** remains a primary objective for the next phase of the project. Transitioning from a breadboard-based prototype to a dedicated PCB will not only improve reliability and durability but also reduce size and enhance mechanical stability. The PCB will be designed to accommodate all essential components, including the ESP32, LoRa module, GPS module, and any required sensors, with proper considerations for power regulation, RF signal integrity, and future expansion. This step is critical to transforming the current prototype into a deployable and scalable solution suitable for field use.

CHAPTER 6: GANTT CHART

6.1 Gantt Chart

SN	Job Description	1st Week	2nd Week	3rd Week	4th Week	5th Week	6th Week	7th Week	8th Week
1.	Problem Identification								
2.	Analysis								
3.	Circuit Design								
4.	Coding								
5.	Implementing & Testing								
6.	Documentation								

Fig 6.1.1: Gantt Chart

REFERENCES

[1] A. R. Kismat Pradhan, Yogesh Limboo, "*A study on gps and telemetry technologies for wildlife conservation*," 2017. [Online]. Available: https: //www.ijariit.com/manuscripts/v4i3/V4I3-1478.pdf

[2] W. SOS, "*Tracking wildlife for conservation research: Gps vs teleme- try technologies*," 2021. [Online]. Available: https://wildlifesos.org/ tracking-wildlife-for-conservation-research-gps-vs-telemetry-technologies/

[3] B. V. P. Keertana, "*A study on gps and telemetry technologies for wildlife conservation*," 2017. [Online]. Available: https://www.ijrti.org/papers/ IJRTI1704054.pdf

[4] M. B.C, "*Animal tracking system using lora technology*," 2019. [Online]. Available: https://ijarbest.com/journal/v5i7/1950

ANNEX

Circuit Diagram

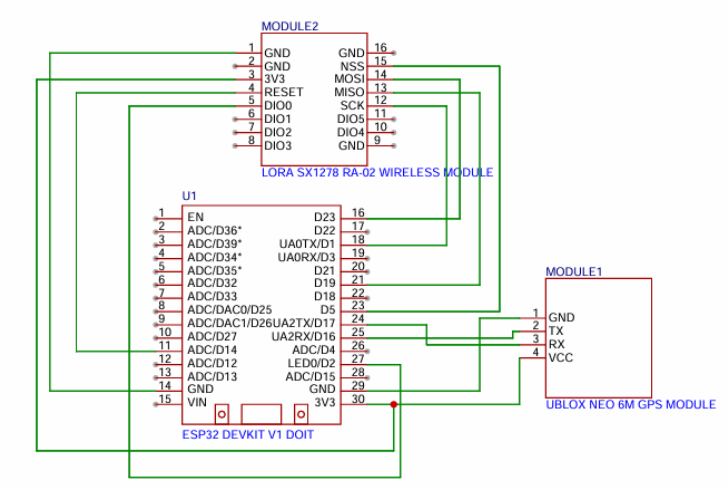


Fig: - Circuit diagram of transmitter

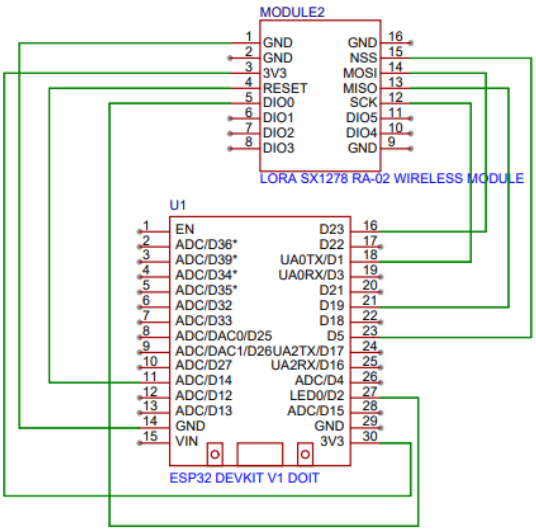


Fig: - Circuit diagram of receiver