

Quick Start Guide - Get Running in 5 Minutes

What You Need

- 1 Master ESP32 with LoRa, SD card, GPS, Buzzer
 - 1-3 Slave ESP32 with LoRa, BMP280, GPS, LED, Button, Battery
 - Laptop with WiFi
 - Arduino IDE installed
-


5-Minute Setup

Step 1: Install Libraries (2 min)

Open Arduino IDE → Tools → Manage Libraries, install:

- LoRa
- ArduinoJson
- Adafruit BMP280
- Adafruit BMP085
- TinyGPSPlus

Step 2: Flash Master Node (1 min)

1. Open `Master_GPS_WiFi_Cloud.ino`
2. Select: Tools → Board → ESP32 Dev Module
3. Select your COM port
4. Click Upload 

5. Wait for "Done uploading"

Step 3: Flash Slave Nodes (1 min each)

1. Open `Slave_GPS_Enhanced.ino`
2. Change `#define NODE_ID "NODE1"` to NODE1, NODE2, NODE3
3. Upload to each slave
4. Repeat for additional nodes

Step 4: Power On & Connect (30 sec)

1. Power master + slaves with batteries/USB
2. Master creates WiFi hotspot: **LoRa_Network**
3. Connect laptop to this WiFi (password: **lora12345**)
4. Open browser: <http://192.168.4.1>

Step 5: See It Work! (30 sec)

- Dashboard loads showing all nodes
- Map displays GPS locations
- Real-time sensor data updates
- Press button on slave → Alert triggers!

Verification Checklist

Master Node:

- ✓ Buzzer beeps twice on startup
- ✓ Serial Monitor shows "Master Ready"

- ✓ WiFi LED (if connected) is solid
- ✓ SD card detected message

Slave Nodes:

- ✓ LED blinks 3 times on startup
- ✓ Serial Monitor shows "Node Ready"
- ✓ Shows "REQ:NODE1" when polled
- ✓ Sends JSON data

Dashboard:

- ✓ Shows node cards with data
- ✓ Map displays markers
- ✓ Data refreshes every 5 seconds
- ✓ CSV download works

First Test

Test 1: Manual Alert

1. Press button on any slave
2. LED should flash
3. Master buzzer sounds
4. Dashboard shows alert
5. All slaves blink for 10 seconds

Test 2: Fall Detection

1. Lift slave node up 3 meters
2. Drop it quickly (10m altitude change)
3. System triggers fall alert
4. Master buzzes rapidly
5. Alert logged to SD card

Test 3: Low Battery

1. Disconnect battery briefly
2. Battery reading drops
3. System warns at <15%
4. Alert appears on dashboard

Common Issues & Fixes

Issue: "LoRa init failed"

Fix: Check antenna, verify pins, ensure 3.3V power

Issue: "SD Card init failed"

Fix: Format SD as FAT32, check CS pin (GPIO 15)

Issue: Can't connect to WiFi

Fix:

- Master Serial Monitor shows IP: 192.168.4.1
- WiFi name: LoRa_Network

- Password: lora12345

Issue: No GPS fix

Fix:

- System uses Bangalore fallback automatically
- Takes 30-60 sec outdoors for GPS lock
- Check GPS TX→16, RX→17

Issue: Nodes not responding

Fix:




- Check antenna on both master & slave
- Verify frequency match (433MHz)
- Ensure nodes have unique IDs
- Check Serial Monitor for "REQ" messages



Understanding the Data

Dashboard Indicators

-  **Green border** = Node healthy
-  **Yellow border** = Battery <15%
-  **Red border** = Node offline
-  **Offline icon** = No data 15+ min

Alert Types

Icon	Meaning	Threshold
	Manual button	User pressed
	Fall detected	10m altitude drop
	Low battery	<15% remaining

Icon	Meaning	Threshold
	Movement	>100m GPS change
	Node offline	15+ min silence

Cloud Upload (Optional)

Ubidots Setup (3 min)

1. Go to ubidots.com → Sign up free
2. Get token: Account → API Credentials → Copy token
3. Edit master code:

```
cpp
#define UBIDOTS_TOKEN "paste_token_here"
```

4. Re-upload to master
5. Data auto-uploads every 60 seconds!

View on Ubidots

1. Devices → See your nodes (node1, node2, node3)
2. Create dashboard with charts
3. Set up email alerts

Advanced: Python Dashboard

Want a fancy laptop dashboard? Run this:

```
bash
```

```
# Install Python packages
```

```
pip install flask folium requests pandas plotly
```

```
# Configure ESP32 IP
```

```
# Edit python script: ESP32_IP = "192.168.4.1"
```

```
# Run dashboard
```

```
python dashboard.py
```

```
# Open: http://localhost:5000
```

Features:

- Real-time charts
- Historical data analysis
- Auto CSV export
- Cloud sync button

Next Steps

Expand Your Network

- Add more nodes (up to 10+)
- Increase range with external antennas
- Add more sensors (humidity, CO2, etc.)

Improve Battery Life

- Use deep sleep mode

- Increase poll interval to 10-30 seconds
- Solar panel charging

Custom Alerts

- Add email notifications via Ubidots
- SMS alerts for emergencies
- Telegram bot integration

Advanced Features

- GPS tracking paths
- Predictive maintenance
- Machine learning anomaly detection

Support Resources

Arduino Reference:

- LoRa: <https://github.com/sandeepmistry/arduino-LoRa>
- ESP32: <https://docs.espressif.com>

Troubleshooting:

- Check Serial Monitor (115200 baud)
- Verify pin connections
- Test components individually

Community:

- Arduino Forum

- ESP32 Discord
 - LoRa enthusiast groups
-

Success Indicators

You're all set when you see:

- ✓ Master shows "Master Ready" on Serial Monitor
- ✓ Slaves respond with JSON data
- ✓ Dashboard displays all nodes with green borders
- ✓ Map shows node locations
- ✓ Manual alert test works
- ✓ SD card logs data
- ✓ CSV download contains sensor readings

Congratulations! Your LoRa network is operational! 

Quick Reference

Default Credentials:

- WiFi: LoRa_Network / lora12345
- Dashboard: <http://192.168.4.1>

File Locations:

- SD Card: /data.txt (sensor log)
- SD Card: /alerts.txt (alert history)
- Local: local_data.csv (Python dashboard)

Pin Summary:

LoRa: 5,14,26,18,19,23

SD: 15 (+ shared SPI)

GPS: 16,17

LED: 25

Button: 4

Battery: 34

Buzzer: 27 (master only)

Timing:

- Poll interval: 3 seconds
- Response timeout: 1.5 seconds
- Cloud upload: 60 seconds
- Offline threshold: 15 minutes

System ready for deployment! 🚀