# Task 1: Fall Detection Feature (Edge ML – ESP32 Smart Pendant)

## Objective

Implement **fall detection** on the ESP32-based smart pendant using **accelerometer + gyroscope data** and a **TensorFlow Lite Micro ML model**. The system should reliably detect falls while minimizing false alarms.

## Requirements

### Hardware Setup

- **MCU:** ESP32-S3 (preferred for ML acceleration)
- **Motion Sensor:** MPU6050 / MPU6886 (accelerometer + gyroscope)
- **Feedback:** Vibration motor or buzzer for user notification
- **Battery-powered** for real-world wearable testing

### ML Model

- **Input:** 6-axis motion data (X, Y, Z acceleration + gyroscope) sampled at ~50Hz
- **Output:**
  - **Normal Activity** (walking, running, sitting, lying down)
  - **Fall Event**
- **Constraints:**
  - Model size < 50KB (quantized)
  - Inference time < 50ms on ESP32

### Functionality

1. **Data Collection Phase:**
    - Collect labeled motion data (normal vs. fall events).
    - Store as CSV/JSON for model training.
2. **Model Training:**
    - Train lightweight CNN/LSTM in TensorFlow.
    - Optimize (quantization + pruning).
    - Convert to `.tflite` (TensorFlow Lite Micro).
3. **Device Integration:**
    - Deploy model on ESP32-S3.
    - Real-time inference on motion data.
    - If a **fall is detected** → vibrate and wait for user cancel (button press).
    - If not cancelled → raise **Fall Alert event** (to be integrated with app in later tasks).
4. **Accuracy Goals:**
    - 90% fall detection accuracy in tests
    - <5% false alarm rate

## Deliverables (for this task)

- Data collection scripts & dataset (CSV format)
- Trained & quantized **.tflite model**
- ESP32 firmware running **real-time fall detection**
- Demo: vibration feedback when fall is detected
- Short documentation:
    - Model architecture
    - Dataset summary

    - Firmware usage instructions

## Timeline

- **Week 1–2:** Data collection & labeling
- **Week 3:** Model training & optimization
- **Week 4:** ESP32 firmware integration + testing