🧑‍⚕️ Real-Time AI Fall Detection and Alert System

This is a multi-layered computer vision system designed to detect human falls in real-time, trigger a local audio alarm, record the incident, and send an immediate email notification with the video attachment and location details to a designated caregiver or emergency contact.

✨ Features

- **Multi-Person Detection:** Uses the **YOLOv8n** (nano) model to efficiently identify all people present in the frame.
- **Real-Time Pose Estimation:** Leverages **MediaPipe Pose** for accurate keypoint tracking (shoulders, hips, knees) to determine posture.
- **Fall Classification:** Classifies a person's state as `Standing`, `Moving`, or `Fallen` based on calculated joint angles and sudden movement patterns.
- **Emergency Alerting:**
  - Triggers a **local audio alarm** upon fall detection using `pygame`.
  - Sends an **email alert** with the current location, timestamp, and the recorded video of the incident.
- **Location Geocoding:** Automatically determines the public IP location using `geocoder` and converts coordinates to a human-readable address using `geopy`.
- **Continuous Recording:** Records the video feed, saving the file locally in the `recordings` directory.
- **Cooldowned Alerts:** Implements a cooldown period and limit on email alerts to prevent spamming.

🛠️ Prerequisites

Before running the script, ensure you have **Python 3.x** installed.

Critical Security Step: You **MUST** generate a **Google App Password** for your sending email account. Using your regular password with `smtplib` is no longer supported by most email providers and is a major security risk.

🚀 Installation

1. Install Dependencies

Install all required Python libraries using pip:

```
pip install opencv-python mediapipe pygame requests geocoder geopy python-dotenv ultral
```

2. Configure Environment Variables (`.env`)

The system uses environment variables to securely store your email password.

Create a file named `.env` in the root directory of the project and add your email configuration. **The variable MUST be named** `EMAIL_PASSWORD` **.**

```
# .env file
# Replace the placeholder with your actual 16-character Google App Password:
EMAIL_PASSWORD=your_16_character_google_app_password
```

3. Configure Hardcoded Variables in `main.py`

You must adjust the following variables inside the `main.py` script before running:

Variable	Location	Description
`EMAIL_SENDER`	Lines 16/17	Your sending Gmail address (must match the account linked to the password in `.env`).
`EMAIL_RECEIVER`	Line 19	The recipient's email address (e.g., caregiver or emergency contact).
`ALARM_FILE`	Line 61	**MUST BE UPDATED.** The absolute path to your `.wav` alarm file (e.g., `C:\Users\User\siren.wav`).

4. Adjust Alert Parameters (Optional)

You can adjust the following alerting logic variables in `main.py` as needed:

Variable	Default Value	Description
`MOVEMENT_THRESHOLD`	`0.03`	Sensitivity threshold for detecting movement between frames.
`MAX_EMAIL_LIMIT`	`2`	Maximum number of emails to send per runtime session.
`EMAIL_COOLDOWN_SECONDS`	`30`	Time (in seconds) that must pass between sending consecutive alerts.
`FRAME_SKIP`	`6`	Number of frames to skip before running the heavy YOLO detection (improves FPS).

🏃 Usage

1. Ensure your webcam is connected and initialized (the script defaults to camera index `0`).
2. Run the script from your terminal:

```
python your_script_name.py
```

3. A window titled **"Multi-Person Posture Detection"** will open, displaying the live video feed.

4. The system will draw pose landmarks and labels (`Standing`, `Fallen`, `Moving`) on detected individuals.
5. Press the `q` key to safely exit the application. The final recording file will be saved in the `recordings` directory.

⚙️ How It Works (The Pipeline)

The script executes a powerful, multi-stage detection pipeline:

1. **Object Detection (YOLOv8):** The video frame is processed by the **YOLOv8n** model to quickly identify the bounding boxes of all objects classified as `person` (class ID 0).
2. **Region of Interest (ROI) Cropping:** The system crops the image to the bounding box of each detected person, maximizing the efficiency of the next step.
3. **Pose Estimation (MediaPipe):** The cropped ROI is processed by **MediaPipe Pose** to find 33 key body landmarks.
4. **Posture Classification:** The key landmarks are used to calculate the angle between the shoulder, hip, and knee. A significantly small angle (less than 170 degrees) combined with lack of movement typically indicates a **Fallen** state.
5. **Alerting Logic:** If a **Fallen** state is confirmed, the local audio alarm is triggered, and the asynchronous email alert process begins (respecting cooldowns).
6. **Recording:** Every processed frame is written to an `.avi` video file in the `recordings` folder for incident verification.