Lightweight Attendance System on Raspberry Pi Zero 2 W

# Project Overview

This is a lightweight version of an AI-powered attendance system using the Raspberry Pi Zero 2 W. It performs simple face detection and manual verification to log attendance with minimal hardware resources.

# 1. Hardware Requirements

- Raspberry Pi Zero 2 W  
- Raspberry Pi Camera Module v2 or USB Webcam (e.g., Logitech C270)  
- MicroSD card (16GB+)  
- Power Supply  
- Heat sink or passive cooling (recommended)  
- Optional: USB hub, keyboard, mouse for setup

# 2. Software & Libraries

- OS: Raspberry Pi OS Lite (or Full for desktop UI)  
- Python 3  
- OpenCV  
- Flask (for the local web interface)  
- SQLite  
- Optional: face\_recognition (if performance allows)

# 3. Performance Optimization Tips

- Use low resolution (e.g., 320x240) and low FPS  
- Increase swap memory:  
 sudo nano /etc/dphys-swapfile  
 Set CONF\_SWAPSIZE=1024  
 sudo dphys-swapfile setup && sudo dphys-swapfile swapon  
- Avoid real-time video stream processing  
- Capture image once every few seconds or on button press

# 4. Basic Face Detection & Approval Code

import cv2  
from flask import Flask, render\_template, request  
from datetime import datetime  
import sqlite3  
import os  
  
app = Flask(\_\_name\_\_)  
face\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml')  
cap = cv2.VideoCapture(0)  
  
@app.route('/')  
def home():  
 ret, frame = cap.read()  
 if ret:  
 frame = cv2.resize(frame, (320, 240))  
 gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)  
 faces = face\_cascade.detectMultiScale(gray, 1.1, 4)  
 for (x, y, w, h) in faces:  
 face\_img = frame[y:y+h, x:x+w]  
 face\_path = 'static/face.jpg'  
 cv2.imwrite(face\_path, face\_img)  
 break  
 return render\_template('index.html', face\_path='static/face.jpg')  
  
@app.route('/approve', methods=['POST'])  
def approve():  
 name = request.form['name']  
 worker\_id = request.form['id']  
 time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")  
 conn = sqlite3.connect("attendance.db")  
 c = conn.cursor()  
 c.execute("INSERT INTO attendance (name, id, time) VALUES (?, ?, ?)", (name, worker\_id, time))  
 conn.commit()  
 conn.close()  
 return "Attendance Logged."  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(host='0.0.0.0', port=5000)

# 5. HTML Template (templates/index.html)

<!DOCTYPE html>  
<html>  
<head><title>Approval</title></head>  
<body>  
 <h2>Manual Attendance Approval</h2>  
 <img src="{{ face\_path }}" width="300">  
 <form method="POST" action="/approve">  
 <label>Name:</label><input type="text" name="name" required><br>  
 <label>ID:</label><input type="text" name="id" required><br>  
 <button type="submit">Approve</button>  
 </form>  
</body>  
</html>

# 6. Database Initialization Script

import sqlite3  
conn = sqlite3.connect("attendance.db")  
c = conn.cursor()  
c.execute('''  
 CREATE TABLE IF NOT EXISTS attendance (  
 record\_id INTEGER PRIMARY KEY AUTOINCREMENT,  
 name TEXT NOT NULL,  
 id TEXT NOT NULL,  
 time TEXT NOT NULL  
 )  
''')  
conn.commit()  
conn.close()