### Project Summary

This is a complete **Face Recognition Attendance System** built in Python. The project is split into three main parts:

1. **encode_faces.py (The Trainer):** This script scans your dataset/ folder, finds all the images of people, computes their facial "encodings" (a unique mathematical representation), and saves them all into the encodings.pickle file. This file acts as your "known faces" database.
2. **app.py (The Main System):** This is the core application. It connects to your **ESP32-CAM** video stream, detects faces in real-time, compares them to the faces in encodings.pickle, and identifies them. When a known person is identified, it logs their attendance in attendance.csv with a status ("Present," "Late," or "Early") based on the time window defined in config.json.
3. **web_interface.py (The Dashboard):** This is a small web server using **Flask**. It provides a simple webpage (index.html) where you can see the attendance data from the attendance.csv file in real-time, displayed in your web browser.

---

### 💻 Hardware & Software Requirements

These are the things you need installed or set up on your devices.

#### Hardware

- **ESP32-CAM:** A low-cost microcontroller with a camera that streams video over your Wi-Fi network.
- **Host Computer:** A computer (like your laptop) that runs the Python scripts. It must be on the same Wi-Fi network as the ESP32.

#### Software

- **Python 3:** The programming language used for the entire project.
- **C++ Compiler & CMake:** These are system-level tools. The face_recognition library (which uses dlib underneath) **must** be compiled from source, so these are required *before* you can install the Python library.
- **A Web Browser:** To view the attendance dashboard (e.g., Chrome, Firefox).

---

### 📚 Key Python Libraries & Modules Used

This is a breakdown of all the import statements in your code and what they do.

#### Core AI & Computer Vision

- **face_recognition:** This is the most important library. It does all the heavy lifting:
  - Finding faces in images (face_locations).
  - Creating the 128-point facial encodings (face_encodings).
  - Comparing faces (compare_faces and face_distance).
- **opencv-python (imported as cv2)**: The primary computer vision library.
  - Connects to and reads the video stream from the ESP32 (cv2.VideoCapture).
  - Reads and processes images for encoding (cv2.imread, cv2.cvtColor).
  - Displays the live video feed in a window (cv2.imshow).
  - Draws the rectangles and names on the video feed (cv2.rectangle, cv2.putText).
- **numpy (imported as np)**: A fundamental library for numerical calculations. It's used by face_recognition and to efficiently find the best match (using np.argmin).

#### Web Server & Dashboard

- **Flask**: A lightweight web server framework used in web_interface.py.
  - Flask: The main class to create the web application.
  - render_template: To show the index.html file.
  - jsonify: To send data (like the attendance list) from Python to the web browser.

### Data Handling & Utilities

- **pickle**: Used to save your knownEncodings and knownNames into the encodings.pickle file and load them back into memory.
- **csv**: Used to read from and write to the attendance.csv log file.
- **json**: Used to read the start and end times from your config.json file.

### Python Standard Libraries (Built-in)

- **os**: Used for interacting with the operating system, mainly to find all the image files in your dataset/ folder and build their file paths.
- **datetime**: Used to get the current date and time, check if it's a new day, and compare the current time to your attendance window.
- **time (imported as py_time)**: Used in app.py to create a short delay (py_time.sleep(2)) if the ESP32 stream disconnects, before trying to reconnect.