Contents

[**Introduction** 2](#_Toc194004454)

[**System Requirements** 2](#_Toc194004455)

[**Setting Up the Project** 3](#_Toc194004456)

[**Installing PyCharm & Creating new project** 3](#_Toc194004457)

[**Project Structure Setup** 5](#_Toc194004458)

[**Installing Dependencies** 6](#_Toc194004459)

[**Installing Dlib** 8](#_Toc194004460)

[**Installing Project Requirements** 9](#_Toc194004461)

[**Setting Up Node.js Requirements** 10](#_Toc194004462)

[**Configuring MySQL Database** 11](#_Toc194004463)

[**Starting the Project & Server** 12](#_Toc194004464)

[**Accessing the Web App** 13](#_Toc194004465)

[**Setting Up Your IP Cameras** 14](#_Toc194004466)

[**Setting up your ESP32 CHIP (Optional)** 17](#_Toc194004467)

[**Guide to Project Usage** 20](#_Toc194004468)

[**Using the System** 20](#_Toc194004469)

[Admin Page 20](#_Toc194004470)

[**Faculty** 23](#_Toc194004471)

[How to start the System? 25](#_Toc194004472)

[**Additional Information** 27](#_Toc194004473)

[**Troubleshooting** 28](#_Toc194004474)

# **Introduction**

Welcome to the **AttendEase** setup guide! This document provides a high level, step-by-step guide on installing, setting up, and running the project.

The system supports [**IP Cameras**](#ipcamera) **and** [**ESP32 chips**](#esp32), but it can automatically adjust if these are not available. (More on this later)

# **System Requirements**

Before proceeding, ensure you have the following installed on your system:

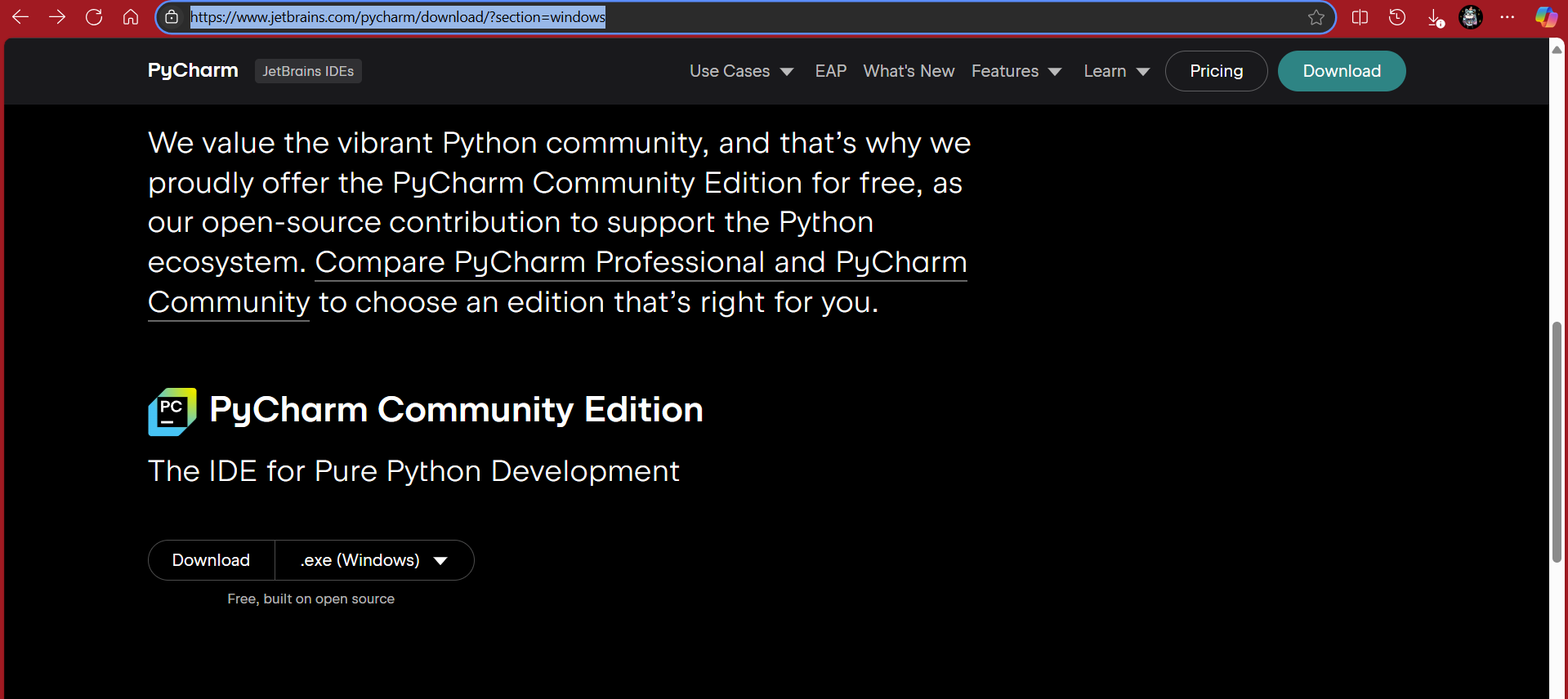
* **PyCharm IDE (Recommended)** – Download the free **Community Edition** from [JetBrains](https://www.jetbrains.com/pycharm/download/)
* **Python 3.10** (Recommended version for stability)
* **Node.js** (Required for the web server)
* **MySQL Database**

*Note:* You can use **Visual Studio Code** instead of PyCharm, but this guide focuses on PyCharm.

# **Setting Up the Project**

## **Installing PyCharm & Creating new project**

1. Download and install **PyCharm Community Edition** from [JetBrains](https://www.jetbrains.com/pycharm/download/).



1. Once installed, open PyCharm and create a **new project**.



1. Name your project and select Python **3.10** as the interpreter.

A screenshot of a program

AI-generated content may be incorrect.

1. Ensure the **Git repository** option is checked.
2. Click **Create** – the project will initialize, and necessary modules will install (this may take 3-5 minutes).

If setup is successful, you will see **Python 3.10 (your\_project\_name)** in the bottom-right corner.

A screenshot of a computer

AI-generated content may be incorrect.

## **Project Structure Setup**

1. Open the **terminal** in PyCharm.

A screenshot of a computer

AI-generated content may be incorrect.

1. Clone the project repository:

git clone https://github.com/GittyCandy/AttendEase.git

1. If prompted, log into **GitHub** to verify access.
2. The cloned project should appear in your main directory under the **AttendEase** folder.

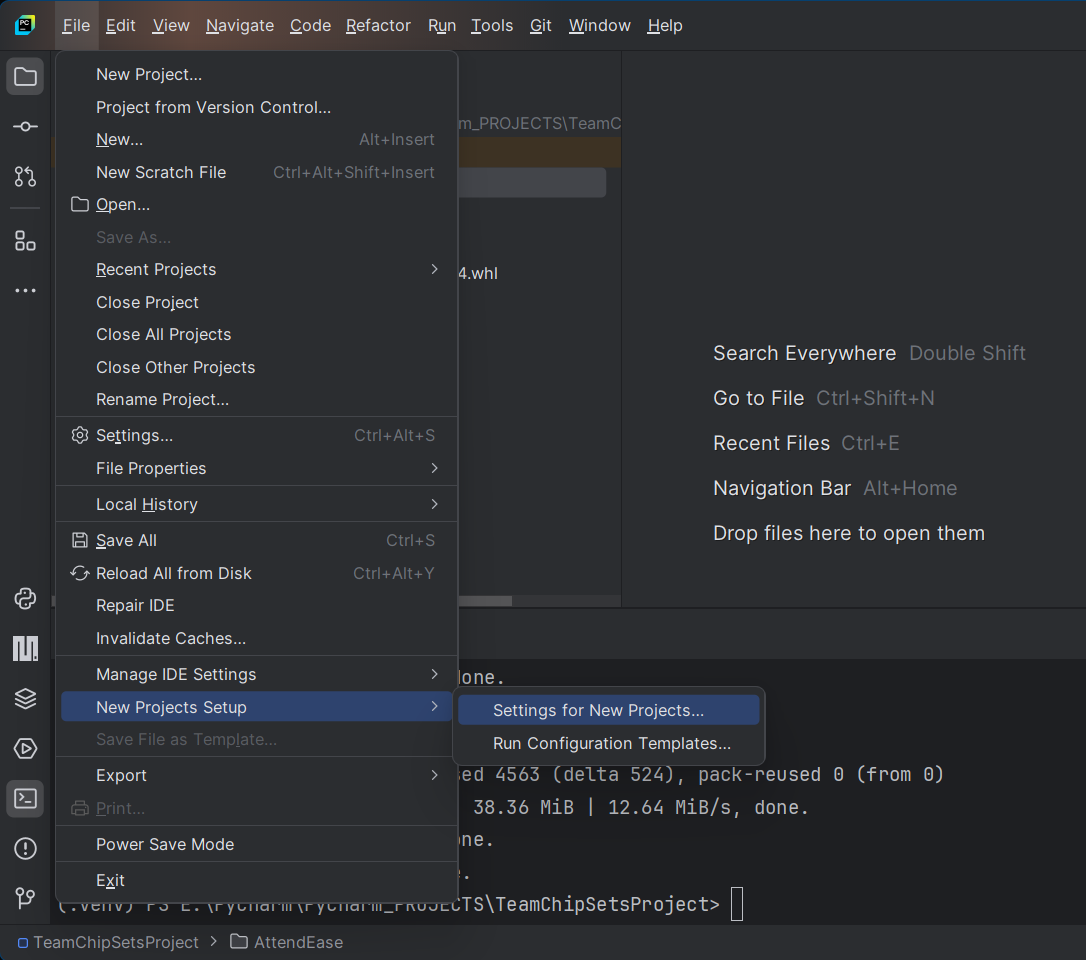
A screenshot of a computer

AI-generated content may be incorrect.

## **Installing Dependencies**

Before installing project requirements, configure the settings:

1. Go to **File > New Project Setup > Settings for New Project**.



1. In the new window, install the following prerequisites:
   * **pip** (23.2.1)
   * **setuptools** (68.2.0)
   * **wheel** (0.41.2)

A screenshot of a computer

AI-generated content may be incorrect.

**Click the add icon to install the pre-requisite lib**A screenshot of a computer

AI-generated content may be incorrect.**raries**

*Recommended to use these versions for compatibility.*

*The final window should look like this:* A screenshot of a computer

AI-generated content may be incorrect.

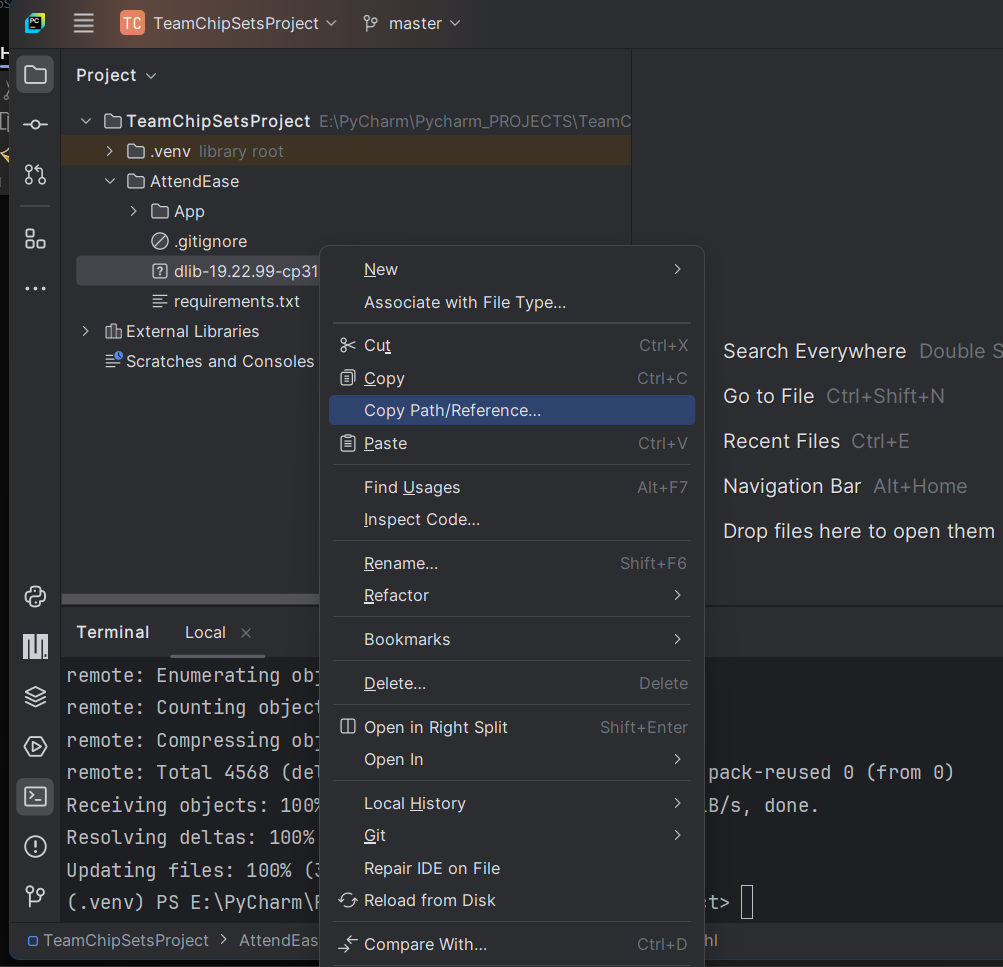
## **Installing Dlib**

1. Change the **terminal directory** to the AttendEase folder:

cd .\AttendEase



1. Locate the **dlib package** in the project.



1. Copy the **absolute path** of the dlib package.

A screenshot of a computer

AI-generated content may be incorrect.

1. In the terminal, run:

pip install The\_Absolute\_Path

A screen shot of a computer

AI-generated content may be incorrect.

## **Installing Project Requirements**

Once Dlib is installed, install all project dependencies:

pip install -r ./requirements.txt

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

This process may take **4-9 minutes** depending on internet speed. (let the indexing finish on the bottom left)

## **Setting Up Node.js Requirements**

1. Verify if **Node.js** is installed by running:

node

A screen shot of a computer

AI-generated content may be incorrect.

If installed, you should see **Welcome to Node.js**. Press Ctrl+C twice to exit.

1. If not installed, download and install **Node.js** from [nodejs.org](https://nodejs.org/).
2. Install the required Node.js packages by running:

npm install express csv-parser mysql2 body-parser bcryptjs express-session multer dotenv fs nodemailer

A black screen with white text

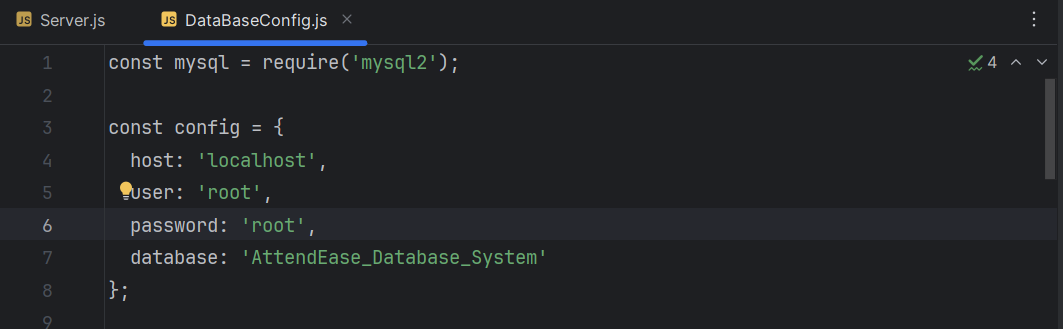
AI-generated content may be incorrect.

## **Configuring MySQL Database**

1. Navigate to the **Config** folder inside AttendEase.
2. Open the **DataBaseConfig.js** file.
3. Update the **username** and **password** fields to match your MySQL credentials.

username: "root",

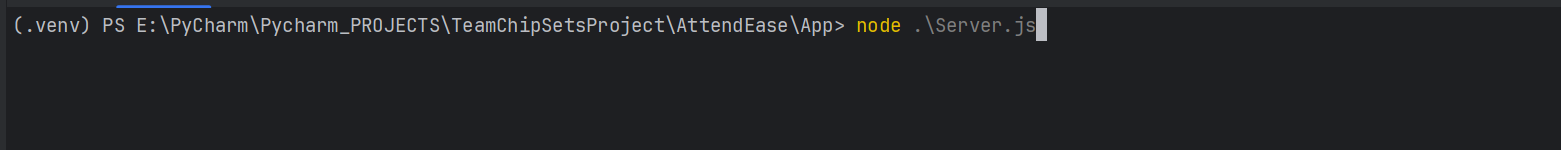
password: "your\_mysql\_password"



## **Starting the Project & Server**

Run the following command to start the main server:

node ./Server.js



A screenshot of a computer program

AI-generated content may be incorrect.

This will:

* Automatically create the database and tables.
* Set up the default **admin account** in MySQL.

To restart the project, run this command again:

node ./Server.js

A screen shot of a computer

AI-generated content may be incorrect.

## **Accessing the Web App**

Once the server is running, go to: <http://localhost:3030> and scroll down for the information about the project and team

A screenshot of a computer

AI-generated content may be incorrect.

# **Setting Up Your IP Cameras**

Before using the face recognition system, it’s important to set up your IP cameras properly. This guide will walk you through setting up **Hikvision IP cameras** for use with the system as it was used throughout the test phase. Links to user manuals for other popular IP camera brands will also be provided.

**Hikvision IP Cameras Setup**

1. **Unbox and Connect the Camera**:  
   * Unbox your Hikvision camera and connect it to your network using an Ethernet cable. Ensure that the camera has power (either via Power over Ethernet (PoE) or an external power supply).
2. **Access Camera's Web Interface**:  
   * Using the default IP address which can be found on the box of the camera (usually 192.168.1.64), access the camera's web interface from a browser on the same network. You may need to use **Hikvision SADP Tool** to find the camera’s IP if it’s not set to a static address.
   * Download the SADP Tool from Hikvision’s official site: https://www.hikvision.com/en/support/tools/hitools/clea8b3e4ea7da90a9/.
3. **Login to the Camera**:  
   * Once you’ve entered the camera’s IP address in your browser, you’ll be prompted to log in. Use the default credentials (typically admin as the username, with 12345 or admin12345 as the default password) and then change the password immediately for security reasons.
4. **Configure Network Settings**:  
   * Open the **Hikvision Web Interface** by entering the camera’s IP address in a web browser.
   * In the camera interface, go to **Configuration** > **Network** to assign a static IP address or configure the camera to work with DHCP, the Facial recognition system will be able to discover this automatically.
   * Ensure that **RTSP (Real-Time Streaming Protocol)** is enabled, which will allow the camera feed to be accessed by the face recognition system. Typically, RTSP can be enabled under **Configuration** > **Network** > **Advanced Settings** > **RTSP**.
   * Go to **Configuration** > **Network** > **Advanced Settings** > **Integration Protocol**.
   * Enable **ONVIF** and create an ONVIF user if required.
5. **Set Up Camera Streams**:  
   * Navigate to **Video/Audio** > **Video Settings** to configure the stream resolution, frame rate, and bitrate for optimal performance. Use the best available settings, the Facial Recognition system can adjust to this based on camera capabilities.
   * Enable the **H.264** codec to ensure compatibility with the face recognition system.
6. **Test the Camera**:  
   * Test the camera feed to ensure everything is working properly by accessing the RTSP stream URL through a video player like VLC or using a simple browser stream if necessary.

For more detailed information on Hikvision camera setup, please refer to the Hikvision User Manual: https://assets.hikvision.com/prd/public/all/doc/m000051606/UD15501B-C\_Baseline\_User-Manual-of-Network-Camera\_V5.5.90\_20221223.pdf

**Setting Up Other Prominent IP Camera Brands**

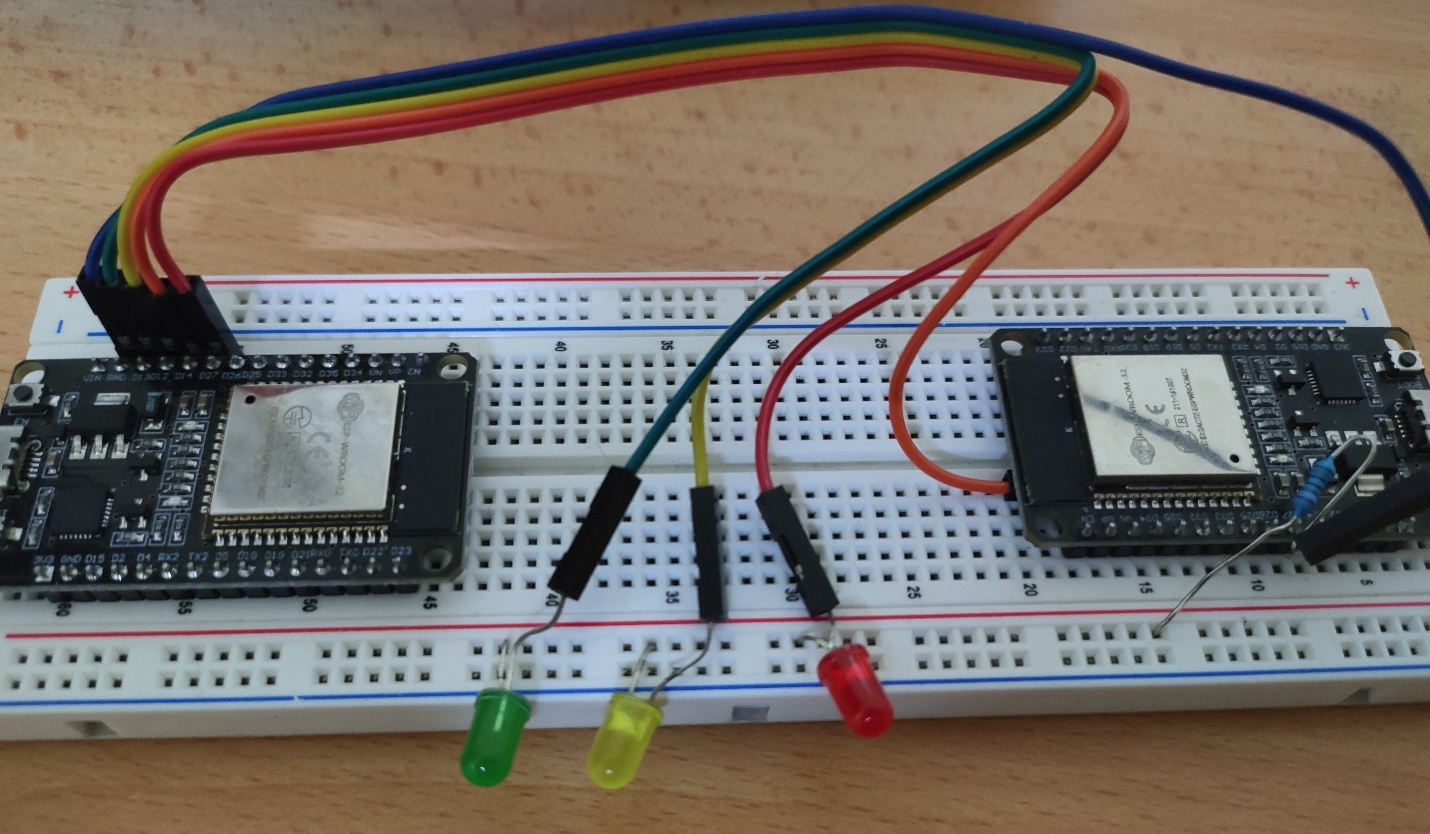
If you're using a different brand of IP camera, the process will be somewhat similar. Below are links to user manuals for other popular camera brands:

* **Amcrest**: https://support.amcrest.com/hc/en-us
* **Axis Communications**: https://www.axis.com/en-ae/products
* **D-Link**: https://www.dlink.com/middle-east/en/for-home/cameras
* **Foscam**: https://www.foscam.com/downloads/index.html
* **Reolink**: https://support.reolink.com/hc/en-us/

Make sure to enable ONVIF and WS Discovery and RTSP streaming to obtain the RTSP stream URL for use in the system.

# **Setting up your ESP32 CHIP (Optional)**

This is an optional addition to get indicator feedback via led on valid student entries its highly customizable and can add according to the organizational setting, (the system will automatically adjust if no chip is found)



How to add the esp32 chip in the project

Requirements

* ESP32 Chip
* Wires, leds, 220 om resistor
* Ardurino ide

Upload this code to your chip and the system will automatically, wirelessly link to the project

#include <WiFi.h>

#include <WebServer.h>

// WiFi credentials

const char\* ssid = "your\_wifiname";

const char\* password = "your\_wifipassword";

// LED GPIO Pins

#define GREEN\_LED 13

#define YELLOW\_LED 14

#define RED\_LED 27

#define BUILTIN\_LED 2  // Built-in LED (GPIO2)

WebServer server(80);

unsigned long ledTimer = 0;

unsigned long blinkTimer = 0;

bool redOn = false;

bool greenOn = false;

bool redBlinking = false;

bool redState = false;  // To toggle red LED while blinking

void handleLED() {

  if (server.hasArg("cmd")) {

    String command = server.arg("cmd");

    command.trim();

    // Blink built-in LED when receiving a new command

    digitalWrite(BUILTIN\_LED, HIGH);

    delay(100);

    digitalWrite(BUILTIN\_LED, LOW);

    if (command == "green") {

      turnOffLEDs();

      digitalWrite(GREEN\_LED, HIGH);

      greenOn = true;

      ledTimer = millis();

      server.send(200, "text/plain", "Green LED is ON (Real Face)");

    }

    else if (command == "yellow") {

      turnOffLEDs();

      digitalWrite(YELLOW\_LED, HIGH);

      server.send(200, "text/plain", "Yellow LED is ON (Computing)");

    }

    else if (command == "red") {

      turnOffLEDs();

      redBlinking = true;

      ledTimer = millis();

      blinkTimer = millis();

      server.send(200, "text/plain", "Red LED is BLINKING (Spoof Detected)");

    }

    else if (command == "stop") {

      turnOffLEDs();

      server.send(200, "text/plain", "All LEDs are OFF");

    }

    else {

      server.send(400, "text/plain", "Invalid command. Use 'green', 'yellow', 'red', or 'stop'");

    }

  }

  else {

    server.send(400, "text/plain", "Missing command parameter");

  }

}

void turnOffLEDs() {

  digitalWrite(GREEN\_LED, LOW);

  digitalWrite(YELLOW\_LED, LOW);

  digitalWrite(RED\_LED, LOW);

  redOn = false;

  greenOn = false;

  redBlinking = false;

  redState = false;

}

// Setup function

void setup() {

  Serial.begin(115200);

  // Initialize LED pins as outputs

  pinMode(GREEN\_LED, OUTPUT);

  pinMode(YELLOW\_LED, OUTPUT);

  pinMode(RED\_LED, OUTPUT);

  pinMode(BUILTIN\_LED, OUTPUT);

  // Ensure all LEDs are off at startup

  turnOffLEDs();

  // Connect to WiFi

  WiFi.begin(ssid, password);

  Serial.println("Connecting to WiFi...");

  while (WiFi.status() != WL\_CONNECTED) {

    delay(500);

    Serial.print(".");

  }

  Serial.println();

  Serial.print("Connected! ESP32 IP address: ");

  Serial.println(WiFi.localIP());

  // Define route for LED control

  server.on("/led", handleLED);

  server.begin();

  Serial.println("HTTP server started");

}

void loop() {

  server.handleClient();

  // Handle red LED blinking if in spoof mode

  if (redBlinking) {

    if (millis() - blinkTimer >= 500) {  // Toggle every 500ms

      redState = !redState;

      digitalWrite(RED\_LED, redState);

      blinkTimer = millis();

    }

  }

  // Auto turn-off Green LED and Red Blinking after 2 seconds

  if ((greenOn || redBlinking) && millis() - ledTimer >= 2000) {

    turnOffLEDs();

  }

}

# **Guide to Project Usage**

## **Using the System**

### Admin Page

Log in with the default admin acc

Email: admin@example.com

password: passwordA screenshot of a computer

AI-generated content may be incorrect.

1. **Uploading Face Images for Training**
   * Format: YourName\_UserID (e.g., Ahsan Latif\_7838220)
   * The uploaded images are stored in a dedicated folder.

A screenshot of a computer

AI-generated content may be incorrect.

Face will be stored in this folderA screenshot of a computer

AI-generated content may be incorrect.

1. **Testing Attendance System (Without Class Setup)**
   * To quickly test the system without setting up a class environment:

Run attendancelogfacemanual.py

A screenshot of a computer program

AI-generated content may be incorrect.

* + This runs face recognition and anti-spoofing in real time.
  + This will use the webcam of ur laptop or pc, to test the anti-spoofing AI, use an **image**, or **video** of the face that is stored
  + Multiple faces are processed using an algorithm that temporarily stores them in the face folder. (this is to help understand how it works better)

A screenshot of a computer

AI-generated content may be incorrect.

In case of multiple faces

### **Faculty**

To access the faculty page, u must register a new account, click login/signup

A screenshot of a computer

AI-generated content may be incorrect.

* Sign up

A screenshot of a purple and white website

AI-generated content may be incorrect.

* Register account (any unique number for userid)

A screenshot of a computer

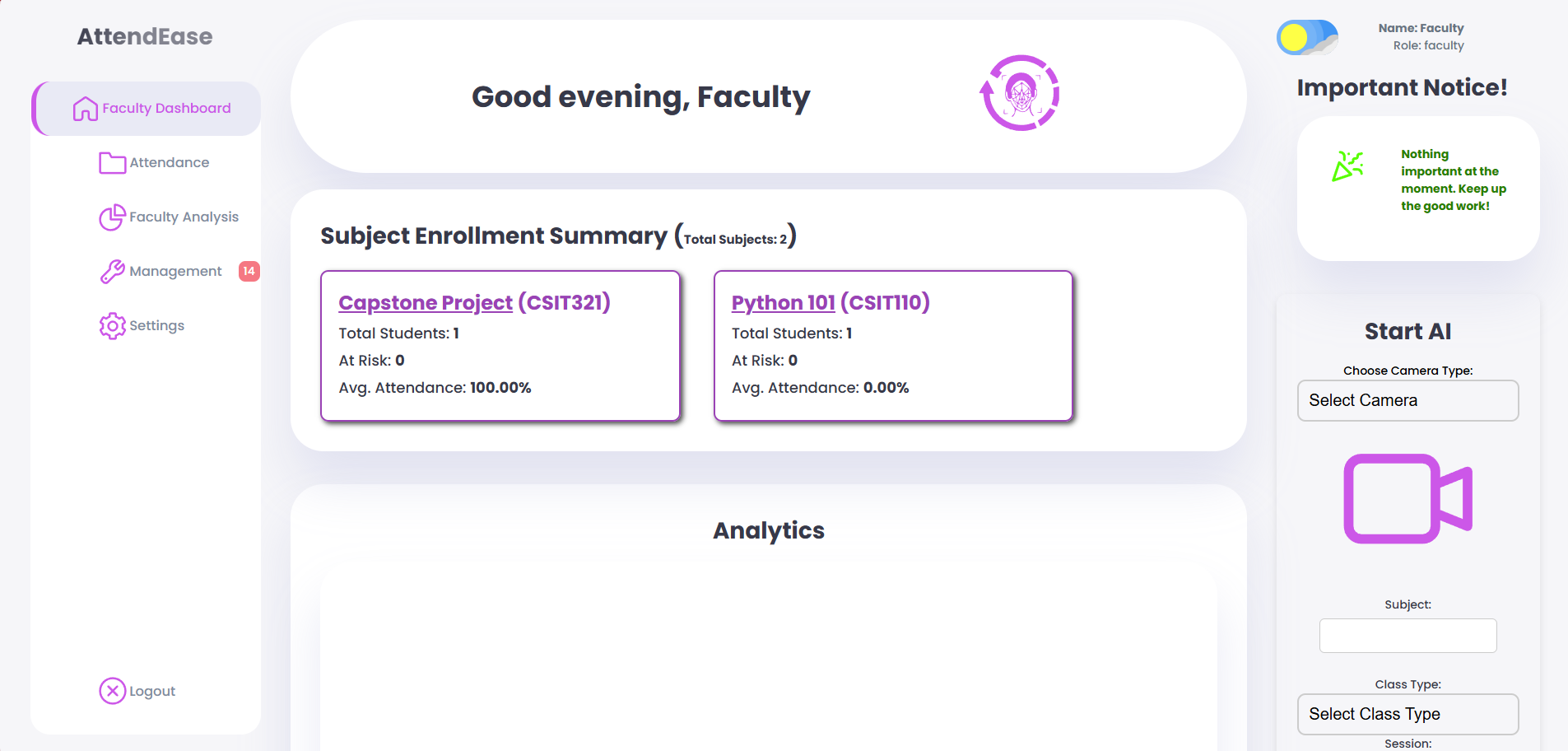
AI-generated content may be incorrect.

Once logged into that new account, you will be given a demo account, we need to give this account faculty role in order to make it a faculty account

* Go to the admin page, under user management, filter that new account and change the role to faculty



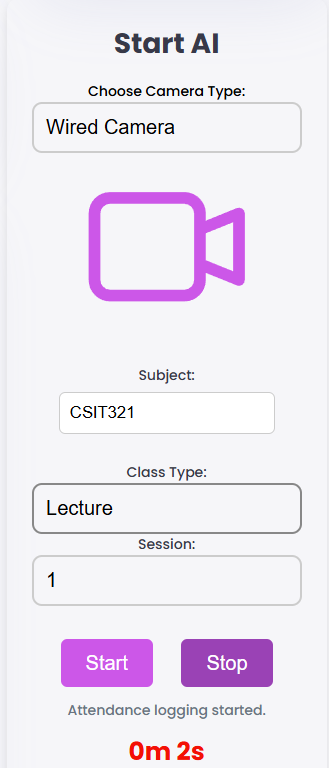
This is done in real time, if u log in with that new account, u will be given faculty perms.



## How to start the System?

On the right-hand side, select a camera, (choose wired, unless u have an external IP camera)

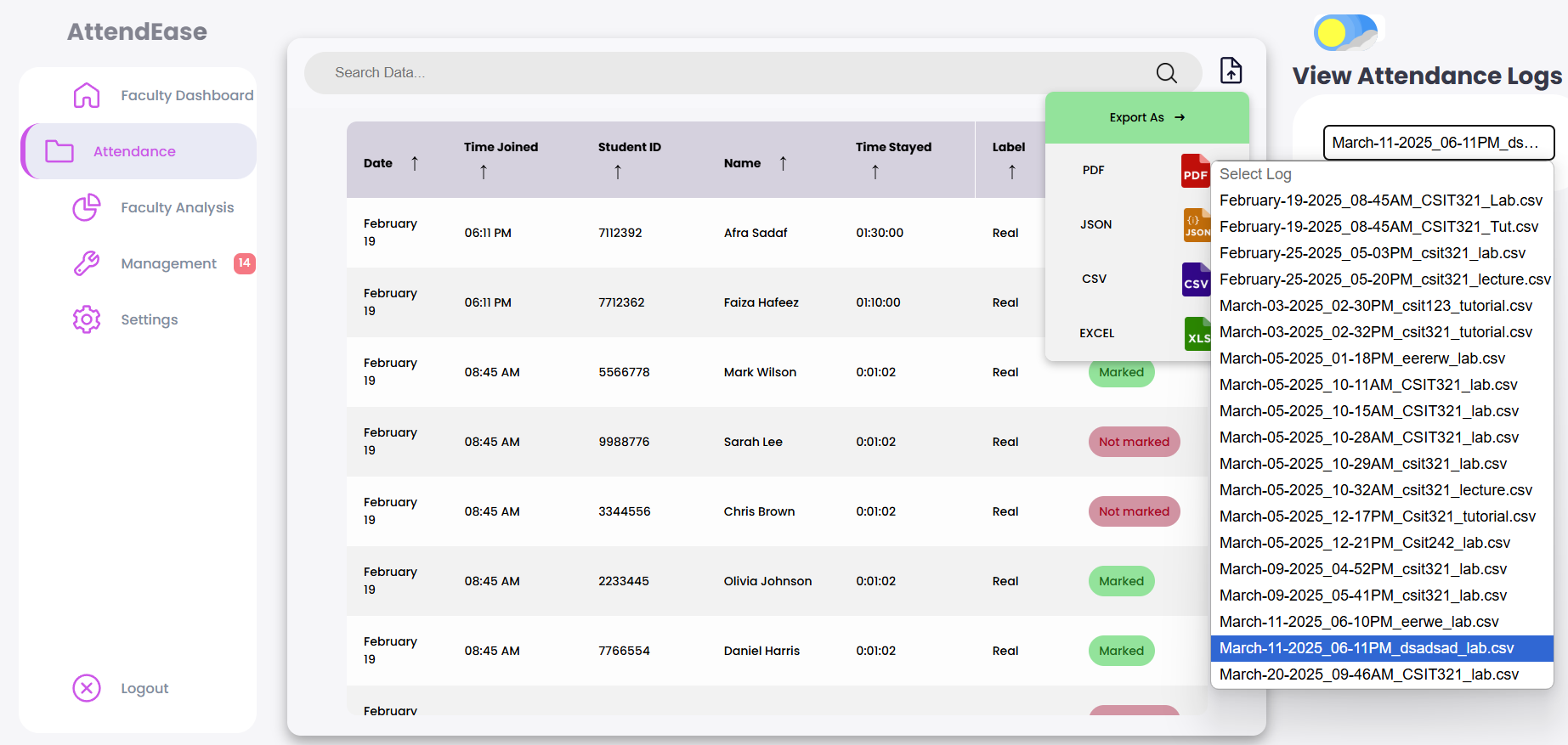
* Enter subject name, class type, and session by default its one, if u have multiple batches, u may choose a different one depending on the use case.



This will start the whole system; u may notice your camera is running now.

Use case, once the class has ended, u press the stop button and it’ll generate the complete analysis of the attendance

It should look like this, on an actual class, u may choose it download it as a pdf or any and select previous class via the logs



It’ll automatically update the database and generate analysis

A screenshot of a graph

AI-generated content may be incorrect.

To manually update attendance, filter by student ID, subject and class type

A screenshot of a computer

AI-generated content may be incorrect.

# **Additional Information**

* **Refer to the Report Documentations** for more details on web usage and functionalities.
* The system is designed for **organizations** where students and faculty are in a structured class environment, but for general testing, an isolated program has been made available

# **Troubleshooting**

* **Error: Python version not found**
  + Ensure Python **3.10** is installed and properly linked in PyCharm.
* **Node.js command not recognized**
  + Reinstall **Node.js** and restart your terminal.
* **Database connection issues**
  + Ensure MySQL is running and credentials in databaseConfig.js are correct.