

**Image Processing with Machine Learning (DD550)**



# **Facial Recognition Based Attendance Management System**

**Kaushik Deka (234209015)**

**Utpal Dutta (234209025)**

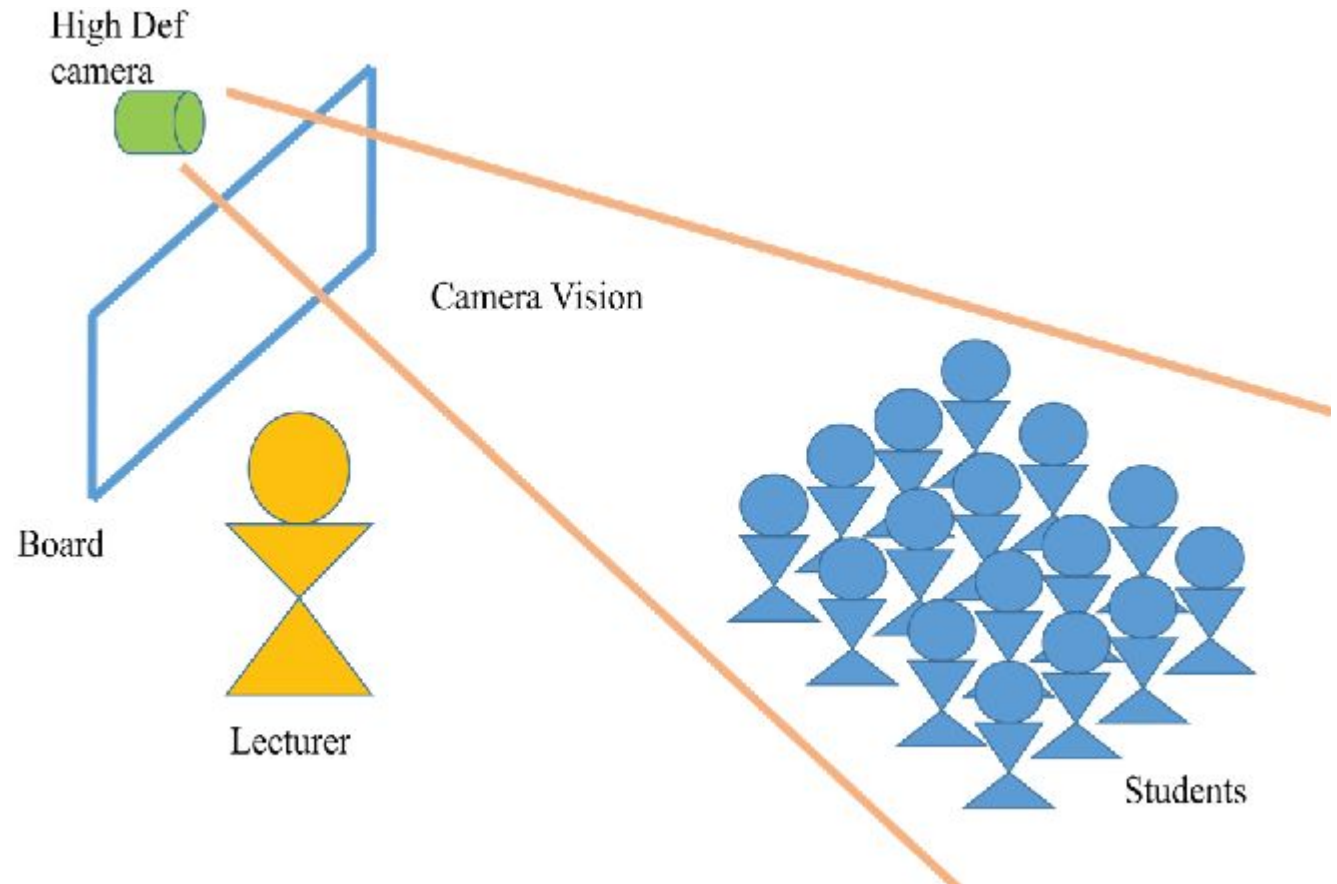
# Motivation

- To address the need for efficient attendance management in various settings.
- To replace time-consuming and error-prone traditional methods with a modern, automated solution.
- To foster a technology-driven approach to streamline administrative processes and enhance overall productivity.



# Problem Statement

- **Develop a robust facial recognition attendance system using Python.**
- **Ensure accurate identification of individuals from images for secure and efficient attendance tracking.**
- **Overcome the limitations of manual attendance tracking, including potential errors and resource-intensive processes.**



# Solution using IPML

## 1. OpenCV:

### Face Detection:

- OpenCV provides pre-trained **Haar Cascade Classifiers**, a type of object detection method, which can be used for face detection.

### Image Capture and Processing:

- OpenCV allows you to capture video frames from a camera (**cv2.VideoCapture**) or load images from a file.
- extensive functionalities for image processing, manipulation, and analysis

### LBPH Face Recognition:

- OpenCV includes implementations of various face recognition algorithms. LBPH (Local Binary Pattern Histograms) is one such algorithm.

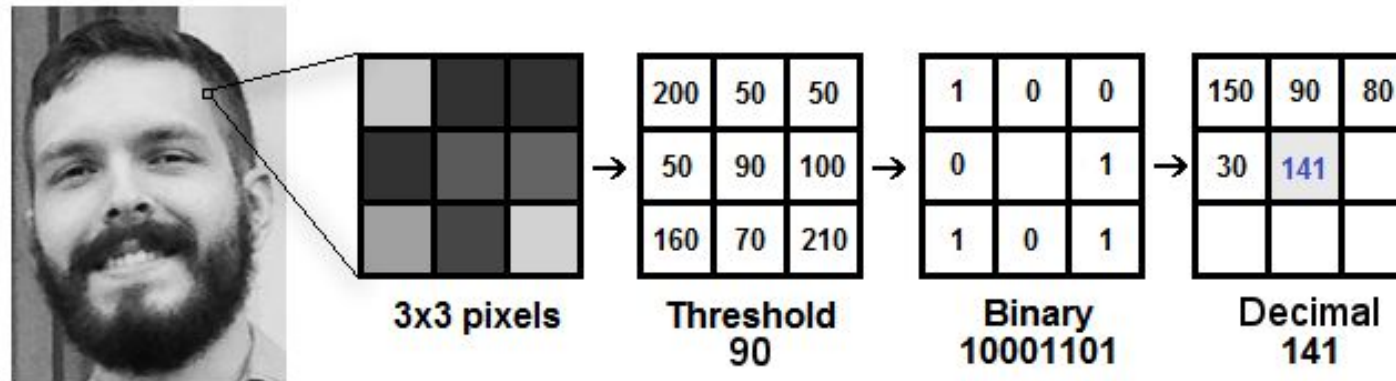
## 2. Pillow (PIL Fork):

### Image Handling and Processing:

- Pillow is a powerful library for working with images. It supports a wide range of image file formats and provides functionalities for opening, manipulating, and saving images.
- This can be useful for preprocessing images before feeding them into a face recognition system.

**LBPHFaceRecognizer** and **HaarCascadeClassifier** are used from OpenCV library for face recognition and face detection.

## 1. LBPHFaceRecognizer



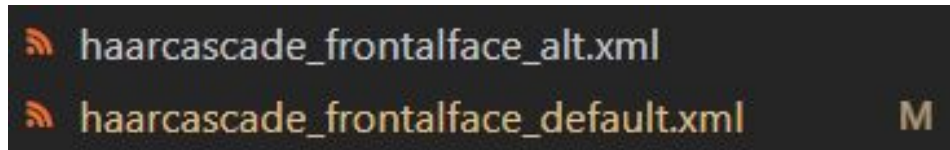
LBPH stands for **Local Binary Pattern Histogram**. It's a face recognition algorithm that's known for its performance and accuracy. LBPH can recognize a person's face from both the front and the side.

- **LBPHFaceRecognizer\_create():** Creates an instance of the LBPH face recognizer.
- **train(training\_images, labels):** Trains the recognizer on a set of labeled training images.
- **predict(test\_image):** Predicts the label and confidence for a given test image.

## 2. Haar Cascade Classifier

The Haar Cascade Classifier is a machine learning object detection method used for identifying objects in images or video. It is particularly well-known for face detection.

- **CascadeClassifier(haarcascade\_path):** Creates an instance of the Haar Cascade Classifier, where `haarcascade_path` is the file path to a pre-trained XML classifier file (e.g., for face detection).



- **detectMultiScale(image, scaleFactor, minNeighbors):** Detects objects (faces, in this case) in the input image. The `scaleFactor` and `minNeighbors` parameters control the sensitivity and robustness of the detection.

These components are often used together in a face recognition system:

- The **Haar Cascade Classifier** is used for detecting faces in images or video frames.
- The detected faces are then processed (e.g., converted to grayscale) and passed to the **LBPFaceRecognizer** for recognition and labeling.

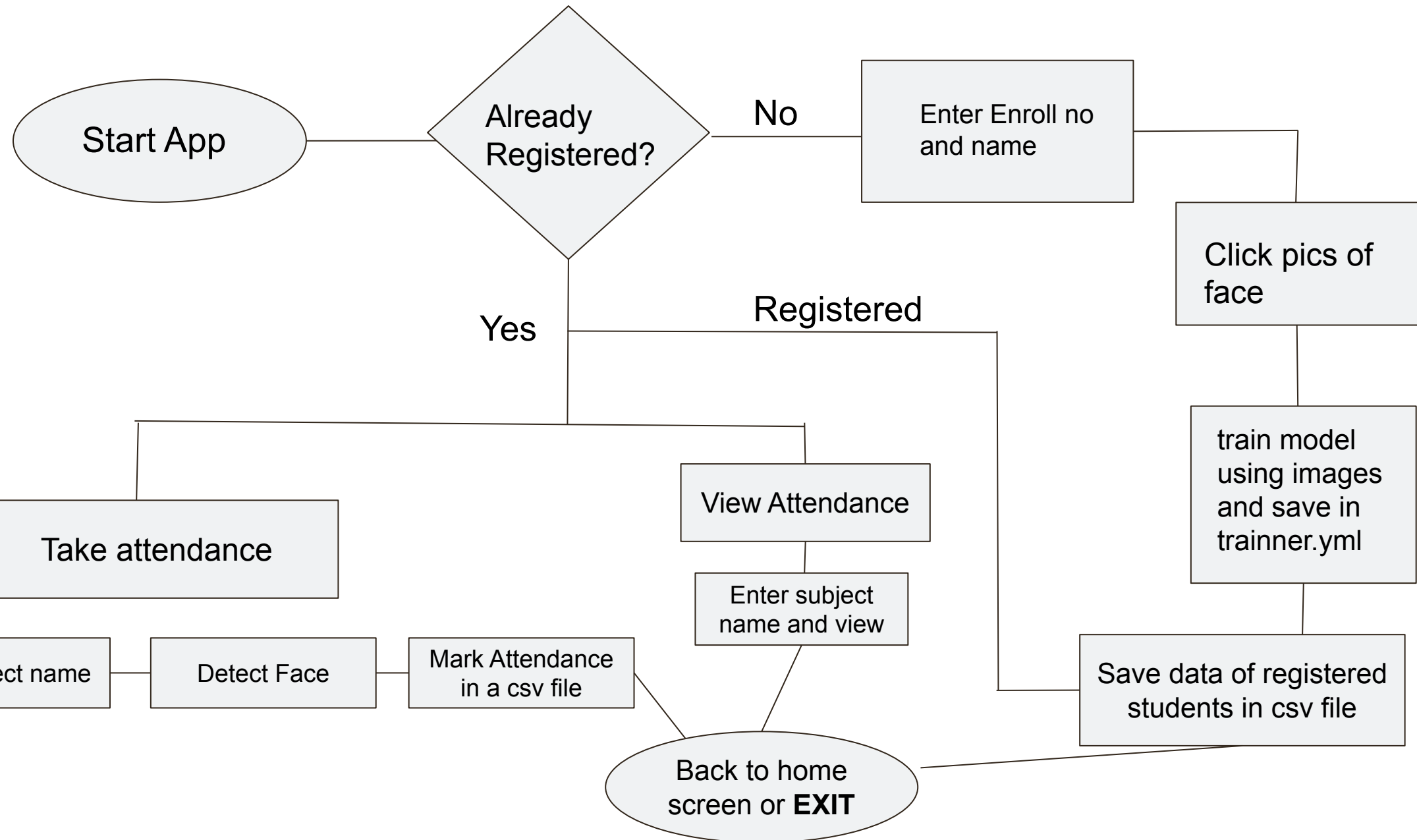
# Tkinter

We used Tkinter toolkit for better usability and user interface of our Attendance system



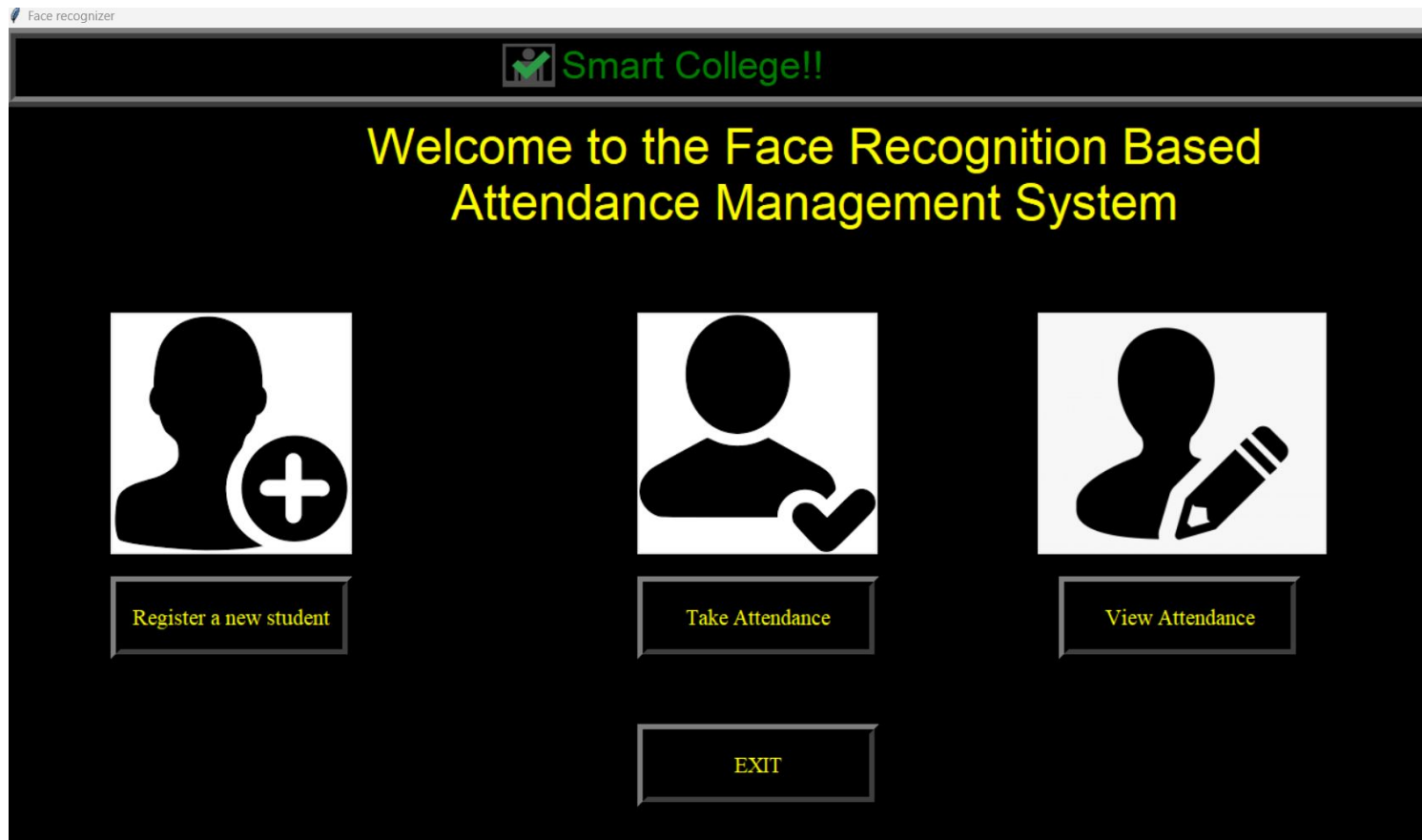
1. **Tkinter** is a Python GUI toolkit that can be used to create **graphical user interfaces (GUIs)**. It is the standard GUI library for Python, and it is included in all standard Python distributions.
2. Tkinter is a thin object-oriented layer on top of **Tk**. Tk is a cross-platform collection of graphical control elements, or widgets, for building application interfaces.
3. Tkinter module helps in creating **GUI applications** in a fast and easy way. Tkinter provides **15 types** of widgets. Some common ones are **Button, Label, Frame, Menu**.

# Flow Chart





# Output



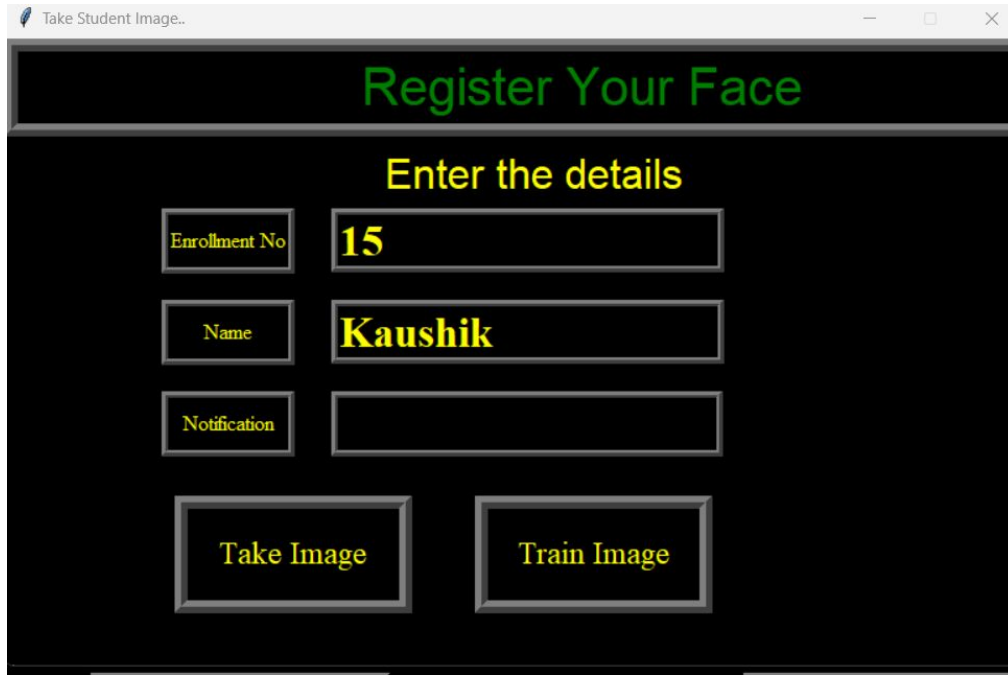
**Tkinter** is used to create the python GUI application.

**OpenCV** is used for Image processing. **Pillow** library is used for added image processing features.

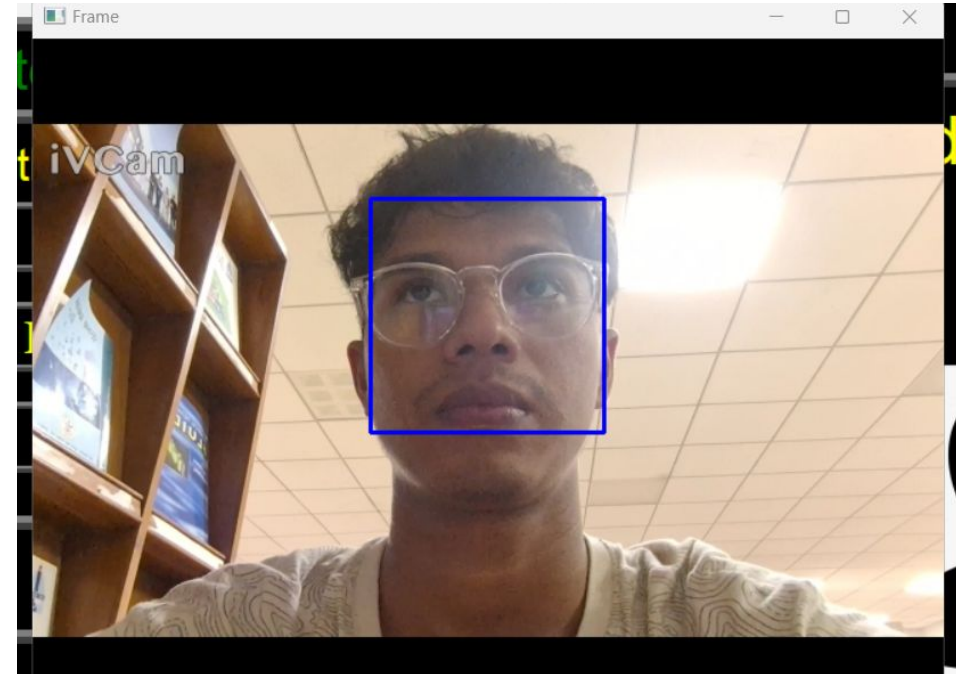
The app has **three** feature:

1. To Register as a new Student
2. Take attendance of a particular subject
3. View Attendance of a particular subject

# 1. Register new Student



The screenshot shows a software window titled 'Take Student Image..'. It has a black background with green and yellow text. At the top, it says 'Register Your Face' in green. Below that, 'Enter the details' is written in yellow. There are three input fields: 'Enrollment No' with the value '15', 'Name' with the value 'Kaushik', and 'Notification' which is empty. At the bottom, there are two buttons: 'Take Image' and 'Train Image', both with yellow text.



OpenCV library is used for Face recognition using system Camera module

```
cam = cv2.VideoCapture(0)
```

It uses OpenCV (cv2) to open the default camera (index 0).

```
detector = cv2.CascadeClassifier(harcascade_path)
```

It initializes a face detector using the Haar Cascade Classifier. This is basically a XML (extensible markup language) file containing generic information about features of a face. The following code uses this information to draw a rectangle around the detected face.

```
while True:
    ret, img = cam.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = detector.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
        sampleNum = sampleNum + 1
```



saved image

- Inside a continuous loop, it captures frames from the camera using `cam.read()`.
- Converts the captured frame (`img`) to grayscale using `cv2.cvtColor`.
- Detects faces in the grayscale image using the `detectMultiScale` function. It returns a list of rectangles, where each rectangle represents a detected face.

## cv2.imwrite

It saves the detected face as a grayscale image in the specified directory using



	A	B	C
1	Enrollmen	Name	
2	15	Kaushik	
3	13	Hrishitaa	
4			
5	12	manu	

```
def TrainImage(haarcascade_path, trainimage_path, trainimagelabel_path):
```

Function to train images

```
recognizer = cv2.face.LBPHFaceRecognizer_create()  
detector = cv2.CascadeClassifier(haarcascade_path)
```

- It initializes a face recognizer using the **LBPH algorithm** with **cv2.face.LBPHFaceRecognizer\_create()**.
- It also initializes a face detector using the Haar Cascade Classifier specified by **haarcascade\_path**.

```
faces, Id = getImagesAndLables(trainimage_path)  
recognizer.train(faces, np.array(Id))  
recognizer.save(trainimagelabel_path)
```

It trains the recognizer using the faces and corresponding labels obtained from the training images. Saves the trained model to the file specified by **trainimagelabel\_path**. This file will be used later for face recognition.

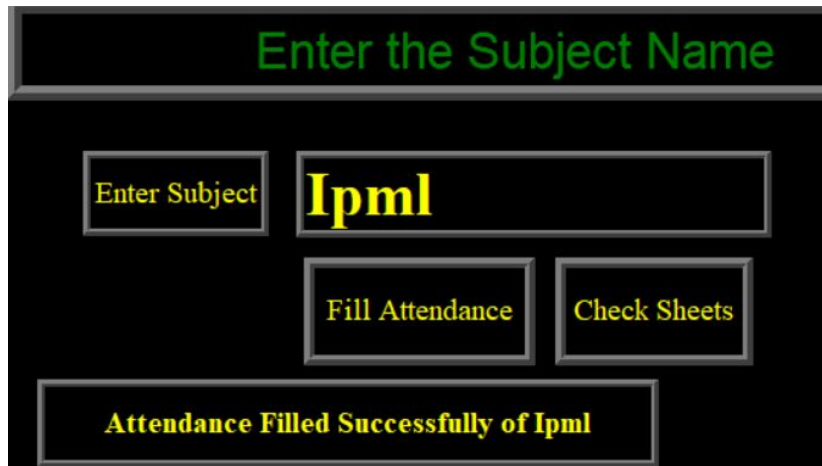
Notification

Image Trained successfully

Images are trained successfully

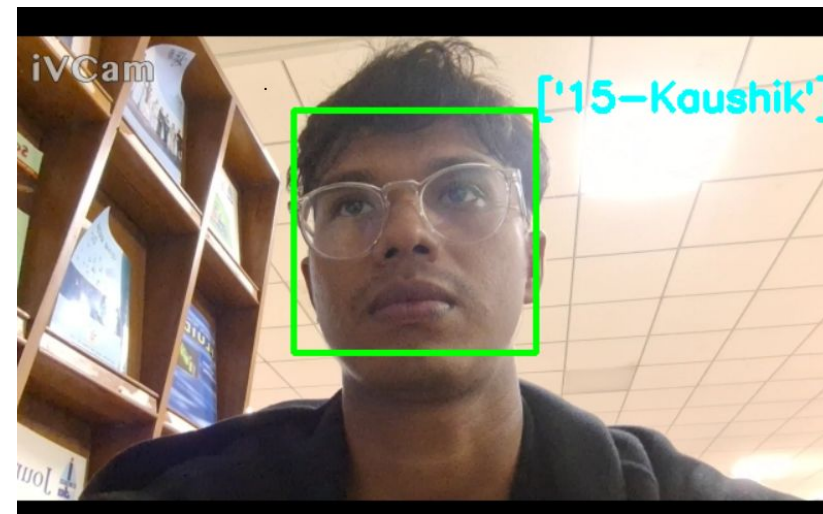


## 2. Take Attendance



Clicking on **Take Attendance** will open another window where one can enter subject name and fill attendance.

This will open the camera module and it will automatically try to detect the face as shown in the image below.

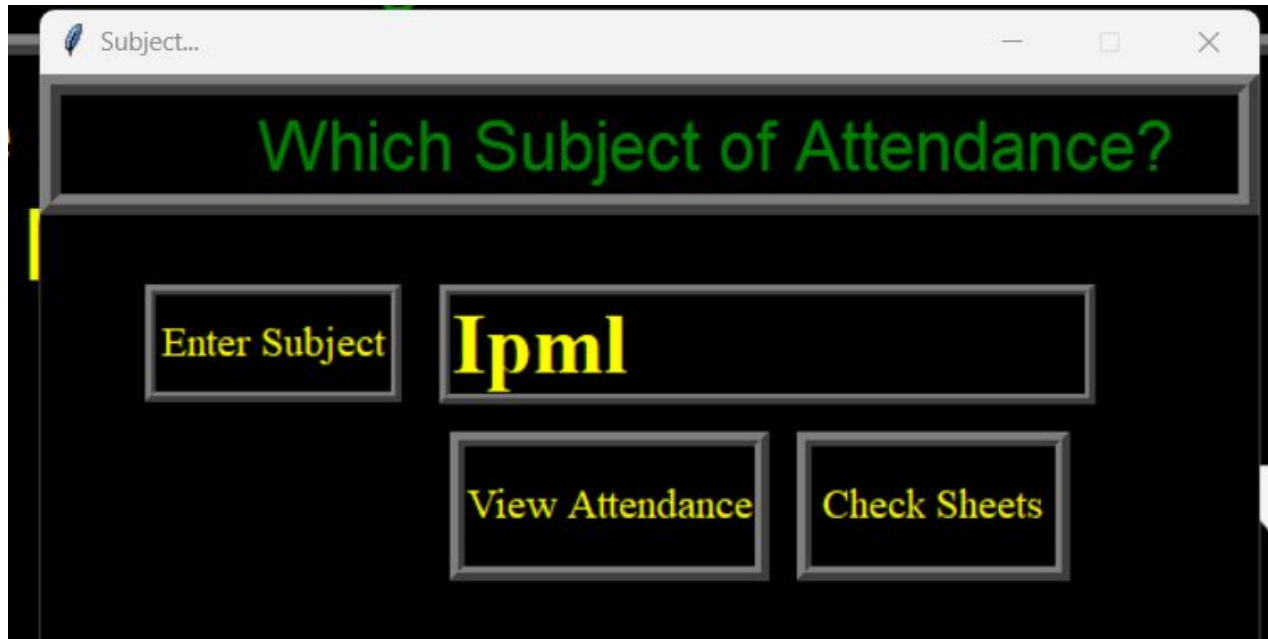


```
recognizer = cv2.face.LBPHFaceRecognizer_create()
facecasCade = cv2.CascadeClassifier(harcascade_path)
df = pd.read_csv(studentdetail_path)
cam = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_SIMPLEX
col_names = ["Enrollment", "Name"]
attendance = pd.DataFrame(columns=col_names)
while True:
    __, im = cam.read()
    gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
    faces = facecasCade.detectMultiScale(gray, 1.2, 5)
```

Attendance is marked and shown in a pop up window.  
It is saved in a csv file

Attendance of Ipml		
Enrollment	Name	2023-11-14
15	['Kaushik']	1

### 3. View Attendance



A screenshot of a web application window titled "Subject...". The window has a dark background. At the top, there is a green text prompt: "Which Subject of Attendance?". Below this, there is a yellow button labeled "Enter Subject". To the right of the button is a text input field containing the text "Ipml". Below the input field, there are two yellow buttons: "View Attendance" and "Check Sheets".



A screenshot of a web application window titled "Attendance of Ipml". The window displays a table with the following data:

Enrollment	Name	2023-11-14	Attendance
15	['Kaushik']	1	100%

One can enter the subject and view the attendance of all the registered students.

# Future Direction

- **Enhanced Security:**
  - Implement anti-spoofing measures to prevent unauthorized access.
- **Database integration:**
  - Implement a sql database in the backend to store and fetch attendance information.
- **Cloud Integration:**
  - Securely store attendance data in the cloud for accessibility and analysis.
- **Continuous Learning:**
  - Regularly train the model with new data to adapt to facial appearance changes.
- **Mobile Integration:**
  - Develop a mobile application for convenient on-the-go attendance tracking

**THANK YOU**