**Name: Yash Gaikwad**
**Roll No: 381071**
**PRN: 22420141**

# Assignment 2

**Problem Statement**: Facial recognition using OpenCV and deep learning for binary classification.

## Introduction:

Facial recognition has become one of the most widely used applications of computer vision and deep learning, with uses ranging from security systems and attendance monitoring to personalized user experiences. It involves detecting and identifying human faces in images or videos. In this assignment, we focus on **binary classification**, where the system determines whether a given face belongs to a particular person (class 1) or not (class 0).
To achieve this, we integrate **OpenCV** for face detection and preprocessing with a **deep learning model** (using frameworks such as TensorFlow/Keras) for classification.

## Objective:

The main objectives of this assignment are:

- To understand the process of facial recognition using computer vision and deep learning techniques.
- To use **OpenCV** for face detection and image preprocessing.
- To build and train a **deep learning model** for binary classification of faces.
- To evaluate the accuracy of the model in distinguishing between the target face and other faces.

## Theory:

### 1 Facial Recognition Pipeline

Facial recognition generally involves two key stages:

1. **Face Detection** – Locating faces in an image or video. OpenCV's Haar cascades or DNN-based detectors are often used for this.
2. **Face Classification** – Using a trained deep learning model to identify whether the detected face belongs to a particular class (yes/no).

### 2 OpenCV for Face Detection

- **OpenCV (Open Source Computer Vision Library)** provides pre-trained models like Haar Cascade Classifier or DNN-based face detectors to locate faces.
- Detected faces are cropped, resized, and normalized before being passed to the deep learning model.

### 3 Deep Learning for Binary Classification

- A **Convolutional Neural Network (CNN)** is commonly used because of its effectiveness in extracting spatial features from images.

- Architecture typically includes:
    - **Convolutional layers** for feature extraction.
    - **Pooling layers** for dimensionality reduction.
    - **Fully connected layers** for decision-making.
    - **Sigmoid activation** in the final layer for binary classification (output between 0 and 1).

**4 Workflow**

1. **Data Preparation**
    - Collect positive samples (images of the target person).
    - Collect negative samples (images of other people).
    - Preprocess images (resize, normalize, grayscale if needed).
2. **Model Building**
    - Define CNN architecture using TensorFlow/Keras.
    - Use sigmoid activation for the final layer.
3. **Training and Evaluation**
    - Compile the model with **binary cross-entropy loss** and optimizers like Adam/SGD.
    - Train on training data and validate on unseen images.
    - Evaluate performance using accuracy, precision, recall, and F1-score.
4. **Integration with OpenCV**
    - Use OpenCV to detect faces in real-time (via webcam).
    - Pass the detected face to the trained CNN for prediction.
    - Display the classification result (e.g., "Person Identified" or "Unknown").

## Conclusion:

In this assignment, we successfully implemented facial recognition for binary classification using **OpenCV** and **deep learning techniques**. OpenCV handled the detection and preprocessing of faces, while the CNN-based model classified the input into two categories: belonging to the target person or not. The project demonstrated the power of combining computer vision with deep learning for real-world applications such as authentication and surveillance. This work can be further extended to **multi-class facial recognition systems** or integrated into real-time applications.