**Face Recognition Based Attendance System**

MUZAMMAL BILAL



BS(CS)-6A

Artificial Intelligence

Computer Science Department

BAHRIA UNIVERSITY ISLAMABAD

# Project Report

Face Recognition-Based Attendance System

1. **Introduction:**

The project presents a robust Face Recognition-based Attendance System developed using machine learning techniques.

Aim: To enhance traditional attendance methods by integrating face recognition technology with liveness detection for improved accuracy and security.

1. **Problem Statement:**

Traditional attendance systems are plagued by issues like inaccuracy, scalability, and susceptibility to fraudulent activities.

The proposed solution targets these challenges by leveraging advanced AI techniques to revolutionize attendance management.

1. **Objectives:**

* + - Develop a custom face recognition algorithm utilizing the K Nearest Neighbors (KNN) approach.
    - Prototype a user-friendly attendance system with seamless integration of face recognition.
    - Evaluate the system's performance, reliability, and security.

1. **Methodology**:

* 1. **Face Recognition Techniques:**

* + - Utilize OpenCV for face detection and KNN algorithm for recognition.
    - Extract facial features and train the model using pre-stored data.

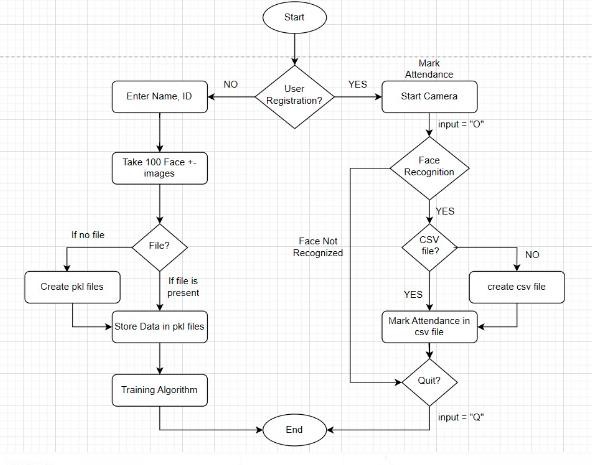
* 1. **System Implementation:**

Develop two Python files:

* + - add\_faces.py for adding faces to the dataset and
    - test.py for recognizing faces and marking attendance.

Utilize Haar Cascade Classifier for face detection.

* 1. **Flow Chart Diagram**



* 1. **Data Management:**

Store face data, IDs, and names in pickle files for efficient retrieval.

Manage attendance records using CSV files, creating new files for each day's attendance.

1. **Working of the Project:**

* 1. **Adding Faces (add\_faces.py):**

Capture face images using a webcam.

Resize and store face data along with corresponding IDs and names.

Update pickle files for storage.

**Explanation**

**Initialization:**

* + - The required libraries (cv2, pickle, numpy, os) are imported.
    - A video capture object video is created to capture frames from the default camera (index 0).
    - A cascade classifier object facedetect is initialized using the Haar cascade XML file for face detection.

**Data Collection:**

* + - The user is prompted to input their name and ID.
    - Inside a while True: loop, the code continuously captures frames from the video feed.
    - Each frame is converted to grayscale to simplify processing.
    - The cascade classifier is used to detect faces in the grayscale frame.
    - For each detected face:
    - The face region is cropped from the original frame.
    - The cropped face image is resized to a fixed size of 50x50 pixels.
    - The resized face image is appended to the faces\_data list.
    - The loop continues until 100 face images are collected or until the user presses 'q' to quit.

**Data Storage:**

* + - The collected face images are converted into a NumPy array (faces\_data) and stored in a pickle file (faces\_data.pkl).
    - The user's name and ID are stored in separate pickle files (names.pkl and ids.pkl).
    - If the pickle files do not exist, they are created; otherwise, the data is appended to the existing files.

* 1. **Recognizing Faces and Marking Attendance (test.py):**

Utilize KNN classifier to recognize faces in real-time.

Retrieve stored IDs and names for recognized faces.

Record attendance with ID, name, and timestamp in a CSV file.

**Explanation**

**Initialization and Setup:**

Libraries are imported, a video capture object is created, and a cascade classifier for face detection is initialized.

**Loading Preprocessed Data:**

Preprocessed face images and associated labels are loaded from pickle files.

**Model Training:**

A K-nearest neighbors classifier is trained using the loaded data.

1. **Expected Outcome:**

Development of a user-friendly attendance system with high accuracy.

Provide a viable alternative to traditional attendance methods, ensuring security and efficiency.

1. **Resources Required:**

Software: VS Code (Python Environment)

Datasets: Personal images( images of family members, BS(CS)-6A students)

Total 2000 images

Collected from 20 people (100 images from each)

Note

Before utilizing the personal images collected for training data in this project, explicit permission was obtained from all individuals involved. Each participant provided informed consent for the use of their facial images for research and educational purposes. This ethical consideration ensures compliance with privacy regulations and respects the rights of individuals regarding the handling of their personal data. Any images used in this project were done so with full consent and confidentiality, and no personal information will be disclosed or shared outside the scope of this project.

1. **Potential Challenges:**

Optimization for real-time processing.

Handling variations in lighting conditions.

Ensuring robustness against spoofing attacks.

Enhancing Accuracy

1. **Output Screens**

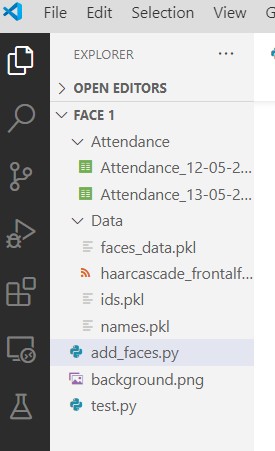
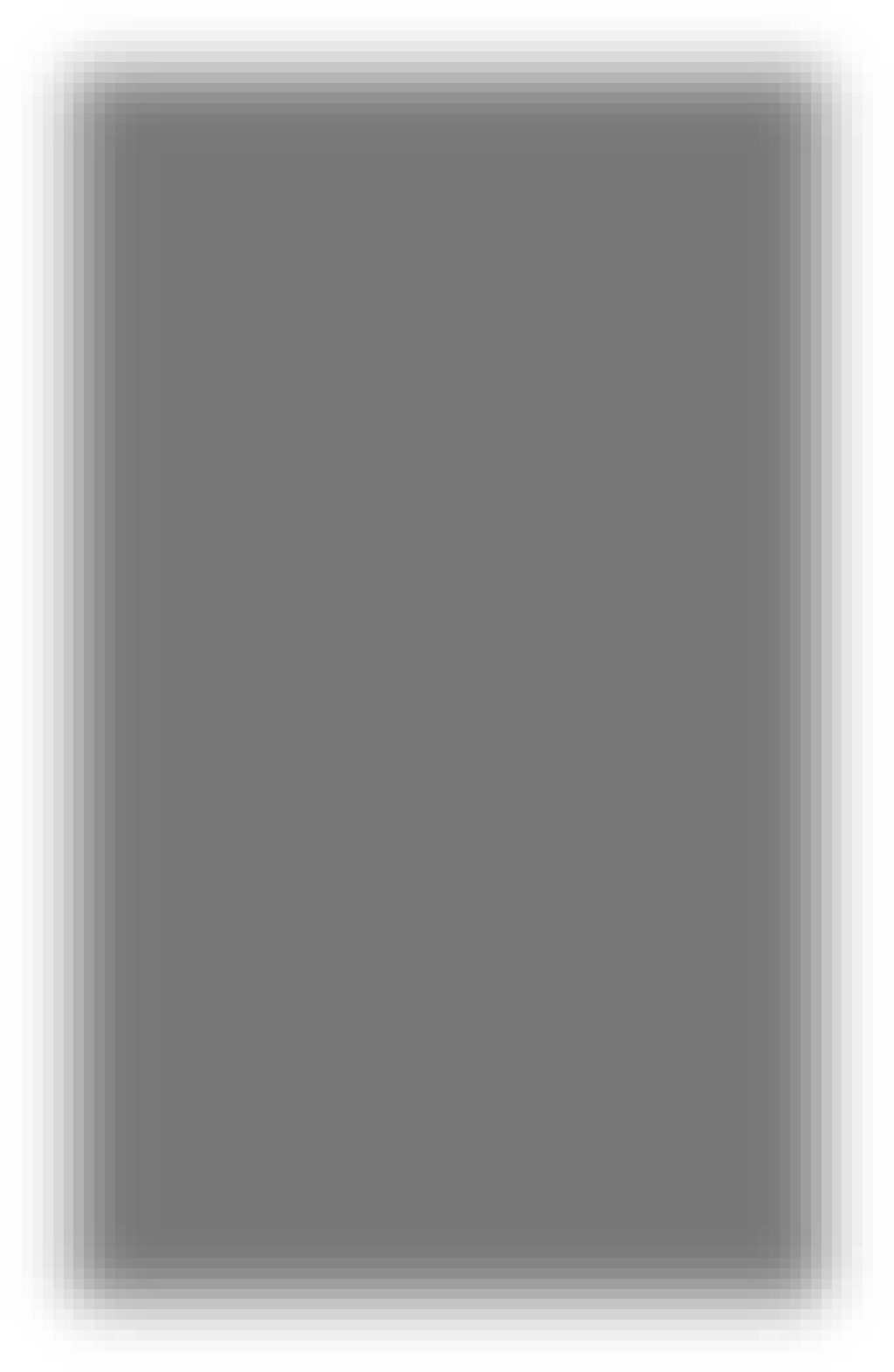
First to get registered, a file “add\_faces.py” needs to run.

It will take username, Id.

It will take 100 images of a person and will be stored in pickle files.

Then to test, or to mark attendance, a file “test.py” needs to run.

On-clicking ‘O’ it will mark attendance and will store the data in a csv file with name, id and time.



**10.Model Architecture**

There are three layers

* + Input Layer: Represents the input data in the form of preprocessed face images.
  + Hidden Layer (Training Layer): Represents the training step of the K-nearest neighbors classifier.
  + Output Layer: Produces the predicted labels (IDs) based on the input data using the trained K-nearest neighbors classifier.

1. **Software Engineering Model**

We employed the Agile software development approach to ensure flexibility and teamwork in our project. By breaking down tasks into manageable cycles, we continuously enhanced our face recognition-based attendance system. This method enabled us to quickly adapt to changes in requirements and incorporate feedback from stakeholders. Additionally, it fostered close collaboration among team members, promoting efficiency and responsiveness throughout the development process. As a result, we successfully delivered incremental improvements to the system while meeting user expectations.

1. **Drawbacks/Limitations**

* + Limited to facial recognition: Doesn't account for other forms of biometric identification.
  + Relies on sufficient lighting conditions for accurate detection.
  + Requires a substantial amount of training data for reliable performance.

1. **Conclusion:**

The project demonstrates the feasibility of using machine learning for attendance management.

By integrating face recognition, the system offers enhanced security and efficiency.