PROJECT REPORT

On

“Face Detection and Recognition System Using Python & Arduino uno”

This project & Research Report has been independently developed by me and is not submitted to any college or university for academic evaluation

Developed by

Name:-Suzen Kumar Mohanty **(Reg. No.** 2421331063**)**

SKMP PROJECT MANAGEMENT

SKMP PROJECT MANAGEMENT

DEPARTMENT OF IOT

POLISHLINE.BALASORE,ODISHA, INDIA

MAY -2026

"Dedicated to my beloved parents, and teachers who have worked hard throughout my education"

CERTIFICATE

This is to certify that the project entitled " Face Detection and Recognition System Using Python & Arduino uno"
has been independently developed and completed by Suzen
Kumar Mohanty for the purpose of enhancing technical knowledge,
skills, and practical understanding.

This project is a self-initiated work carried out with personal interest and research. It has not been submitted to any
college, university, or institution for the award of any degree or
diploma.

The work presented in this project is genuine and has been completed by the developer himself.

Suzen Kumar Mohanty
(Project Developer)

Date: 14/ 05/ 2026
Place:Balasore,Odisha,India

SELF ATTESTATION

I hereby declare that I have personally worked on the project entitled " Face Detection and Recognition System Using Python & Arduino uno ".

All the data, analysis, and implementation presented in this project are the result of my own efforts and genuine work. Any external sources of information, if used, have been properly acknowledged.

This project is created solely for learning, skill development, and professional improvement, and has not been submitted to any institution for academic purposes.

Suzen Kumar Mohanty
(Project Developer)

Date: 14/ 05 / 2026
Place:Balasore,Odisha,India

DECLARATION

I hereby declare that the project titled “Face Detection and Recognition System Using Python & Arduino uno” is an original work carried out by me based on my own idea, research, and implementation.

This project has not been submitted previously, in part or in full, to any college, university, or institution for the award of any degree, diploma, or certification.

I confirm that all the information presented in this project is true to the best of my knowledge. Any references or sources used have been properly acknowledged.

I take full responsibility for the authenticity and originality of this work.

Name: Mr. Suzen Kumar Mohanty
Branch: Computer Science And Engineering
Institution : [Srinix Collage Of Engineering , Independent]
Date: [14/05/2026]
My PortPolio Link:- https://suzen-kumar-mohanty-8fdv.vercel.app/

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to everyone who directly or indirectly supported me in the successful completion of my personal project "Face Detection and Recognition System Using Python & Arduino uno".

This project is a result of my own idea, research, and continuous effort to enhance my technical knowledge and practical skills in the field of Internet of Things (IoT). Working on this project has provided me with valuable learning experiences and improved my understanding of real-world applications.

I would also like to thank my family and friends for their constant encouragement, motivation, and support throughout the development of this project.

Finally, I am thankful to all the online resources, tutorials, and communities that helped me gain knowledge and solve challenges during the project development.

Developed By:
Suzen Kumar Mohanty

ABOUT MY SELF

I am **Mr. Suzen Kumar Mohanty**, a passionate and dedicated **Web Developer** from the field of **Computer Science and Engineering**, with strong expertise in HTML, CSS, JavaScript, and Vue.js. I specialize in building scalable, responsive, and high-performance web applications that deliver seamless user experiences across all devices and screen sizes.

With a keen eye for detail, I focus on writing clean, maintainable, and well-structured code. My development approach is centered around creating intuitive user interfaces, ensuring both functionality and aesthetic quality. I follow modern front-end development practices such as component-based architecture, responsive design principles, and performance optimization techniques to build efficient and user-friendly applications.

Beyond web development, I actively work on **IoT-based solutions**, where I integrate hardware and software to create smart, real-world systems. I have hands-on experience with Arduino Uno and various sensors, enabling me to develop projects such as environmental monitoring systems, smart automation solutions, and real-time data tracking applications.

One of my key strengths lies in connecting IoT devices with web technologies. I build interactive dashboards that allow real-time data visualization, giving users the ability to monitor, analyse, and control devices remotely. My work focuses on bridging the gap between the physical and digital world by transforming raw sensor data into meaningful insights.

I am deeply interested in developing innovative, data-driven solutions that combine web technologies with embedded systems. My goal is to create impactful applications that solve real-world problems while improving efficiency, automation, and user experience.

With a strong passion for continuous learning and innovation, I consistently explore new tools, frameworks, and technologies to enhance my skills and deliver high-quality, future-ready solutions.

Name: Mr. Suzen Kumar Mohanty
Branch: Computer Science And Engineering
My PortPolio Link:- https://suzen-kumar-mohanty-8fdv.vercel.app/

ABSTRACT

With every passing day, we are becoming more and more dependent upon technology to carry out even the most basic of our actions. Facial detection and Facial recognition help us in many ways, be it sorting of photos in our mobile phone gallery by recognizing pictures with their face in them or unlocking a phone by a mere glance to adding biometric information in the form of face images in the country's unique ID database (Aadhaar) as an acceptable biometric input for verification.

This project lays out the basic terminology required to understand the implementation of Face Detection and Face Recognition using Intel's Computer Vision library called 'OpenCV'.

It also shows the practical implementation of the Face Detection and Face Recognition using OpenCV with Python embedding on both Windows as well as macOS platform.

The aim of the project is to implement Facial Recognition on faces that the script can be trained for. The input is taken from a webcam and the recognized faces are displayed along with their name in real time.

This project can be implemented on a larger scale to develop a biometric attendance system which can save the time-consuming process of manual attendance system

LIST OF FIGURES

INDEX

Chapter 1

INTRODUCTION

1.1 INTRODUCTION

A face recognition system is a technology that
can effectively match a person's face from a digital image or
a video frame, using it as
a reference to compare and identify against stored face data.
Researchers are currently exploring various methods through
which face recognition systems operate.The most advanced face recogni
tion technique, which is also used for user verification through
ID services, works by detecting and measuring facial features from
a given image. Initially developed as
a type of computer application, face recognition systems have expanded
their usage in recent years on smartphones and
in other types of technology, such as artificial intelligence.Because auto
mated face recognition involves measuring a person's physiological cha
racteristics, face recognition systems are classified as
a form of biometrics.Althoughthe accuracy of face recognition systems
as a biometric technology is lower than that of
iris recognition and fingerprint recognition, it is widely adopted due to
its contactless and non-
invasive approach.Facial recognition systems are used in advanced hum
an-computer interaction, video surveillance, and
the automatic categorization of images.Wehave developed a face recogn
ition technology capable of identifying faces.

1.2 PROBLEM STATEMENT

Facial Detection and Facial
Recognition using Intel's open source Computer Vision Library (OpenCV) and Python dependencies
There are several scripts included in
the project that demonstrate various functionalities such as detecting fac es in static images, detecting faces in real-time video from
a webcam, capturing facial images and saving them in
a dataset, training a classifier for recognition,
and finally performing face recognition on the trained model.

All the scripts are written in Python 3.14.6 and come with well-documented code.
This project provides a comprehensive overview of
the essential tools and information required for face detection and face r ecognition, making it valuable for individuals interested in exploring fac ial recognition using OpenCV.
The project illustrates the implementation of multiple algorithms and re cognition methods, which will be further discussed in the project report.

Face recognition has significant applications in areas such as security, o rganization, marketing, surveillance, and robotics, among others.

Face detection can significantly enhance surveillance efforts, which can greatly assist in identifying individuals with
a criminal background, including criminals and terrorists who pose a ser ious threat to national and public security.
It also improves personal security, as there is nothing
for hackers to steal or alter, such as passwords.

.1.3 OBJECTIVES

This project is created so as to study the various means of recognizing faces with more
accuracy and reducing the error rates while recognition. The ideal condition for any
recognition project is to reduce the intra class variance of features and increase the
inter class variance of features to be detected or recognized.
Facial Recognition software is “Capable of uniquely identifying or verifying a person
by comparing and analyzing patterns based on the person’s facial contours. It is mostly
used for security purposes”. Many other areas of use.
Different recognizer approaches are used for recognition of faces. They are:

- Eigen Faces
- Fisher Faces
- Local Binary Pattern Histograms3

1.4 METHODOLOGY

TheProject makes use of several Python libraries, including OpenCV ve rsion 1.4.1. OpenCV, or the Open Source Computer Vision Library, is an opensource software library that focuses on computer vision and mac hine learning.Its main function is
to offer a common platform for developing computer vision application s.It was specifically built for such tasks and also helps in speeding up the use of machine perception in business products. Since it
is licensed under the BSD license, it
is easy for businesses to use and modify the code as needed.
In total, the library offers around 2500 optimized algorithms, which is a very impressive number.
These algorithms cover a wide range ofboth traditional and modern com puter vision and machine learning techniques.Theyare used for tasks lik e detecting and recognizing faces, identifying objects, classifying huma n actions in videos, tracking camera movements, following moving obje cts, creating 3D models of objects, generating 3D point clouds from ster eo cameras, combining images to form highresolution scenes, finding si milar images froma database, removing red eyes from photos taken with flash, tracking eye movements, recognizing landscapes,and placing mar kers for augmented reality applications, among others.
What makes this library particularly remarkable is
its large user community, which includes more than 47,000 members, and it has been downloaded over 18 million times.
The library is widely used by corporations, research teams,
and government organizations.

Alongside well-known companies such as Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, and Toyota, which make use of OpenCV, there are also many startups like Applied Minds, VideoSurf, and Zeitera that extensively use the library.

The application of OpenCV spans across a wide range of areas, from stitching together street view images, assisting in police work in Israel, monitoring mine activities in China, helping robots navigate and manipulate objects at Willow Garage, detecting drowning incidents in swimming pools in Europe, creating interactive art in Spain and New York, checking runways for debris in Turkey, scanning product labels in factories worldwide, and enabling fast face detection in Japan.

1.4.2 NumPy

NumPy (Numerical Python) is one of the most important libraries in Python for scientific computing and numerical analysis. Although Python was not originally developed for mathematical computing, it quickly became popular among researchers and engineers due to its simplicity and flexibility.

The development of NumPy began with the Numeric library in the mid-1990s. Later, Num array was introduced to improve performance for large datasets. In 2005, Travis Oliphant combined the strengths of both Numeric and Num array, resulting in the first official release of NumPy 1.0 in 2006 as part of the SciPy ecosystem. Since then, NumPy has become the standard library for numerical computing in Python, with support for Python 3 introduced in version 1.5.0.

NumPy provides a powerful N-dimensional array object, mathematical functions, linear algebra operations, statistical tools, random number generation, and Fourier transforms. It performs computations much faster than standard Python lists because its arrays are stored efficiently in memory.

Key Features of NumPy:-

- **High-performance multidimensional arrays.**
- **Fast mathematical and statistical operations.**
- **Linear algebra and matrix computations.**
- **Random number generation and scientific functions.**
- **Easy integration with libraries such as Pandas, SciPy, and Matplotlib.**

NumPy is widely used in data science, machine learning, artificial intelligence, image processing, scientific research, and engineering applications due to its speed, reliability, and extensive functionality.

1.4.3 Pillow

Pillow is an open-source Python library used for image processing and manipulation. It is the modern version of the original Python Imaging Library (PIL) and provides developers with an easy way to work with digital images.

The library supports a wide range of image file formats, including JPEG, PNG, BMP, GIF, TIFF, and WebP. Pillow enables users to open, edit, enhance, resize, crop, rotate, and save images efficiently. It also includes features for applying filters, adjusting colors, drawing text and shapes, and converting images between different formats.

Applications of Pillow:-

- **Image resizing and thumbnail generation.**
- **Image format conversion.**
- **Cropping, rotating, and flipping images.**
- **Applying filters and enhancements.**
- **Adding text, logos, and watermarks.**

- **Basic image analysis and preprocessing for machine learning.**

Advantages of Pillow:-

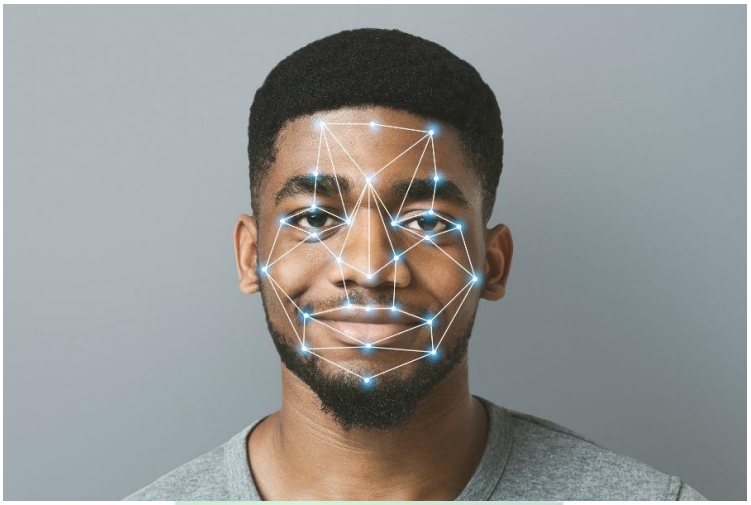
- **Simple and easy-to-use API.**
- **Supports multiple image formats.**
- **Fast image processing performance.**
- **Cross-platform compatibility.**
- **Frequently updated and actively maintained**.

Pillow is widely used in computer vision, web development, automation, AI applications, and image editing projects. It is installed using the following command:
pip install pillow
After installation, the library can be imported into Python programs using:
from PIL import Image
Pillow provides an efficient and flexible solution for handling image-related tasks, making it one of the most popular image processing libraries in Python.

1.4.4 FACE RECOGNITION

The Face Recognition library is a Python library that enables developers to detect, recognize, and compare human faces in images and videos. It is built on top of the dlib machine learning library and uses deep learning techniques to provide accurate facial recognition capabilities. This library simplifies the implementation of face recognition systems by offering easy-to-use functions for detecting face locations, generating facial feature encodings, comparing faces, and identifying individuals. It can process images from files, webcams, or video streams with only a few lines of Python code.
The Face Recognition library uses pre-trained deep neural network

models that achieve high recognition accuracy under different lighting conditions and facial expressions. It is widely used in applications that require reliable identity verification and face matching.



(Fig-01 Face recognization)

Key Features:-

- **Detects one or multiple faces in an image.**
- **Generates unique facial encodings for each detected face.**
- **Compares facial features to identify matching individuals.**
- **Supports real-time face recognition using webcam input.**
- **Easy integration with OpenCV and other Python libraries.**
- **High accuracy using deep learning-based models.**

Applications:-

- **Attendance management systems.**
- **Security and access control.**
- **Surveillance and monitoring.**
- **User authentication.**
- **Smart home and IoT systems.**

- **Photo organization and tagging.**

1.5 Organization

Biometric authentication is a security method that identifies or verifies individuals using their unique biological or behavioral characteristics. Common biometric traits include fingerprints, facial features, iris patterns, voice, and signatures. Since these characteristics are unique to each individual, biometric systems provide a reliable and secure method of authentication.

Before a biometric system can authenticate a user, the individual's biometric data must be collected and stored in a database through a process known as enrollment. During authentication, the system compares the captured biometric data with the stored records to determine the user's identity.

Biometric authentication generally operates in two modes:

Identification:-

Identification is the process of comparing an individual's biometric data with all records in the database to determine their identity. This is commonly referred to as one-to-many (1:N) matching.

Verification:-

Verification confirms whether a person is who they claim to be by comparing the captured biometric data with a single stored template. This is known as one-to-one (1:1) matching.

Types of Biometrics:-

Biometric technologies are broadly classified into two categories:

1. **Physiological Biometrics**

2. Behavioral Biometrics

1. ***Physiological Biometrics:-***

Physiological biometrics rely on the physical characteristics of an individual that remain relatively stable over time.

Fingerprint Recognition:-

Fingerprint recognition analyzes the unique ridge patterns and minutiae points present on a person's fingertips. It is one of the most widely used biometric techniques because of its high accuracy, speed, and reliability. Fingerprint authentication is commonly used in smartphones, attendance systems, and access control applications.

Facial Recognition:-

Facial recognition identifies individuals by analyzing facial features such as the distance between the eyes, nose, mouth, jawline, and overall facial structure. It is widely used in surveillance systems, mobile device security, and attendance management. Recognition accuracy may be affected by changes in lighting, facial expressions, aging, or accessories such as masks and glasses.

Iris and Retina Recognition:-

Iris and retina recognition use the unique patterns present in the human eye for authentication. These methods provide very high accuracy and are commonly used in high-security environments. However, specialized hardware is required, making them more expensive than other biometric technologies.

Voice Recognition:-

Voice recognition identifies users based on vocal characteristics such as pitch, tone, pronunciation, and speech patterns. Although convenient for

remote authentication, its accuracy may decrease due to background noise, illness, or changes in a person's voice.

DNA Recognition:-

DNA-based authentication offers one of the highest levels of identification accuracy because every individual's DNA profile is unique. Due to its cost and processing time, it is mainly used in forensic science, criminal investigations, and medical applications rather than routine authentication.

2. Behavioral Biometrics:-

Behavioral biometrics analyze patterns in a person's actions rather than physical characteristics.

Signature Recognition:-

Signature recognition verifies a person's identity by analyzing handwriting style, writing speed, pressure, stroke order, and signature dynamics. It is widely used in banking and legal document verification.

Keystroke Dynamics:-

Keystroke dynamics measures an individual's typing behavior, including typing speed, key press duration, typing rhythm, and error frequency. This technique provides an additional layer of authentication in cybersecurity applications.

Chapter 2

LITERATURE SURVEY

2.1 Face Tracking:-

Face tracking is a computer vision technique that continuously detects and follows a person's face across images or video frames. It identifies important facial landmarks such as the eyes, nose, mouth, and jawline to monitor facial movement and position. Modern face tracking systems are widely used in surveillance, biometric authentication, augmented reality, video conferencing, and human-computer interaction. They improve recognition accuracy by maintaining continuous tracking even when the face moves or changes orientation.

2.2 Mechanisms of Human Facial Recognition:-

Human facial recognition involves detecting facial features and converting them into meaningful representations for identification. Modern recognition systems use machine learning and deep learning algorithms to extract distinctive facial characteristics and generate numerical feature vectors. These feature vectors are then compared with stored templates to recognize or verify an individual's identity. Facial recognition systems are designed to handle variations in pose, lighting conditions, facial expressions, and aging while maintaining high recognition accuracy.

2.3 Eye Spacing Measurement for Facial Recognition:-

Eye spacing measurement is an important facial feature extraction technique used in biometric recognition. It calculates the distance between the eyes and combines this measurement with other facial landmarks to improve identification accuracy. Image processing techniques such as edge detection, contour analysis, and circular feature detection help locate the iris and eye boundaries accurately. This feature

is commonly integrated with multiple facial measurements to create robust face recognition systems.

2.4 Linear Discriminant Analysis (LDA) for Face Recognition:-

Linear Discriminant Analysis (LDA) is a dimensionality reduction technique widely used in face recognition and pattern classification. It transforms high-dimensional facial data into a lower-dimensional feature space while maximizing the separation between different classes. By reducing redundant information and emphasizing distinguishing facial characteristics, LDA improves recognition accuracy and computational efficiency. It is commonly combined with machine learning algorithms in facial recognition, image classification, and biometric authentication systems.

Chapter 3

SYSTEM DEVELOPMENT

3.1 Applications of Face Recognition Systems:-

Face recognition is a computer vision technology that detects, analyzes, and identifies human faces from digital images or video streams. It is a specialized application of object detection that focuses on locating facial regions and comparing them with stored facial templates. Modern face recognition systems use machine learning and deep learning algorithms to recognize individuals accurately, even under varying lighting conditions, facial expressions, and viewing angles.
The technology has become an essential component of biometric authentication systems due to its high accuracy, contactless operation, and ease of deployment.
Applications of Face Recognition

1. Facial Motion Capture:-

Facial motion capture records a person's facial movements using cameras or sensors and converts them into digital facial data. This information is widely used in computer animation, video games, virtual reality, augmented reality, and film production to create realistic facial expressions for digital characters.

2. Biometric Authentication:-

Face recognition is widely used for identity verification and access control. The system compares facial features with a stored database to authenticate users. Common applications include smartphone unlocking, attendance management, airport security, banking, and secure building access.

3. Photography and Smart Cameras:-

Modern smartphones and digital cameras use face detection to improve autofocus, exposure control, smile detection, and portrait photography. Face recognition also helps organize and categorize photo collections automatically.

4. Video Surveillance:-

Security agencies use face recognition in surveillance systems to identify individuals in real time. The technology assists in monitoring public places, detecting suspicious activities, and locating missing or wanted persons.

5. Human-Computer Interaction:-

Face recognition enables computers and smart devices to interact naturally with users. Applications include virtual assistants, personalized user interfaces, driver monitoring systems, and emotion recognition.

6. *Marketing and Customer Analytics:-*

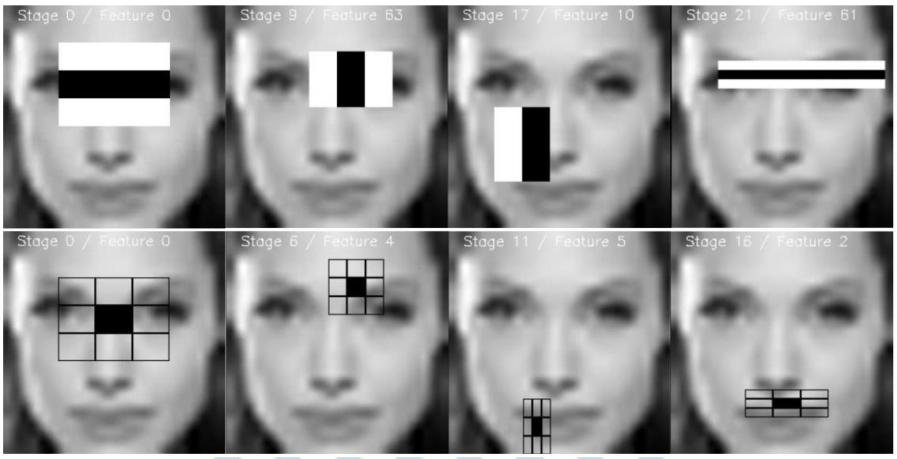
Retail stores and advertising platforms use face recognition to estimate customer demographics, analyze engagement, and deliver personalized advertisements while respecting applicable privacy regulations.

3.2 Viola–Jones Algorithm:-

The Viola–Jones algorithm is one of the earliest and most successful real-time object detection algorithms, particularly for face detection. Developed by Paul Viola and Michael Jones in 2001, it introduced a fast and efficient approach for detecting frontal human faces in images and video.

The algorithm combines Haar-like features, integral images, AdaBoost,

and a cascade classifier to achieve high detection speed while maintaining good accuracy.



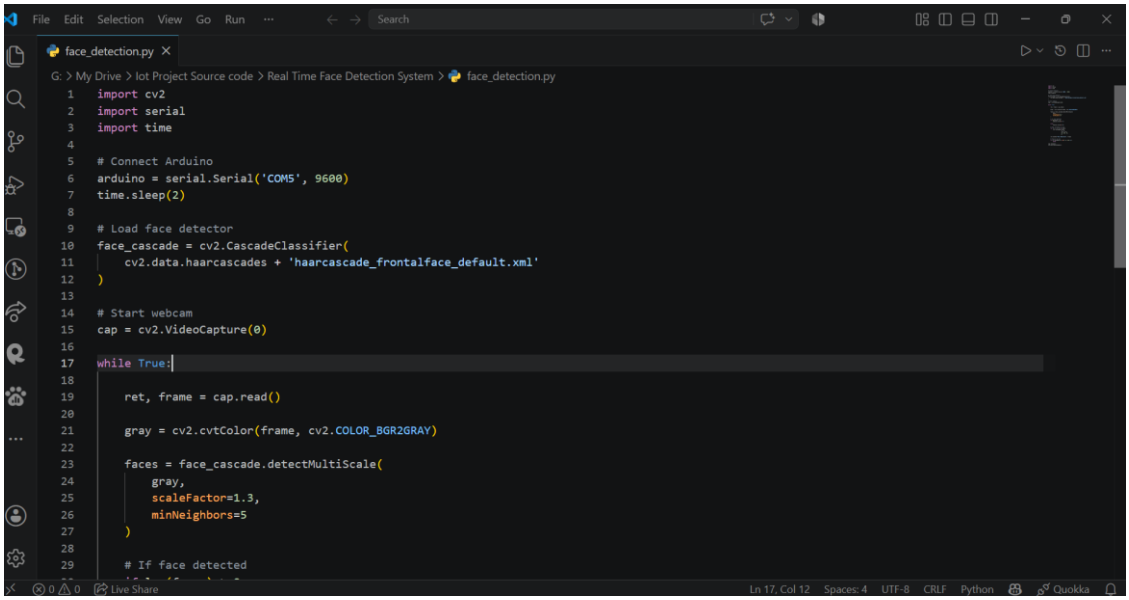
(Fig-2 Viola–Jones Algorithm)

Working Principle:-

The algorithm scans an image using a sliding detection window of different sizes. At each position, the system checks whether the selected region contains a face. If the region passes all classification stages, it is identified as a face; otherwise, it is rejected immediately.

Step-by-Step Process

1. **Convert the input image into an integral image.**
2. **Slide a detection window across the image at multiple scales.**
3. **Extract Haar-like features from each window.**
4. **Evaluate features using AdaBoost-trained weak classifiers.**
5. **Pass the window through multiple cascade classifier stages.**
6. **Declare the region as a face only if it passes every stage.**



(Fig-03 Live Cam Feed)

Key Components of the Viola–Jones Algorithm:-

1. Haar-like Features:-

hear-like features measure differences in pixel intensities between adjacent rectangular regions. These features capture important facial characteristics such as the darker eye region, brighter cheeks, and the bridge of the nose.

Common hear features include:

- **Edge features**
- **Line features**
- **Four-rectangle features**

These simple features enable fast face detection while requiring minimal computational resources.

2. Integral Image:-

An integral image is a preprocessing technique that allows the sum of pixel values within any rectangular region to be calculated using only

four array references.

Advantages:

- **Significantly reduces computation time.**
- **Enables rapid calculation of Haar features.**
- **Supports real-time object detection.**

3. AdaBoost Algorithm:-

AdaBoost (Adaptive Boosting) is a machine learning algorithm used to select the most informative hear features from thousands of possible candidates.

Its main functions include:

- **Selecting the best discriminative features.**
- **Combining weak classifiers into a strong classifier.**
- **Improving detection accuracy while reducing computational complexity.**

Instead of evaluating hundreds of thousands of features, AdaBoost retains only the most useful ones for face detection.

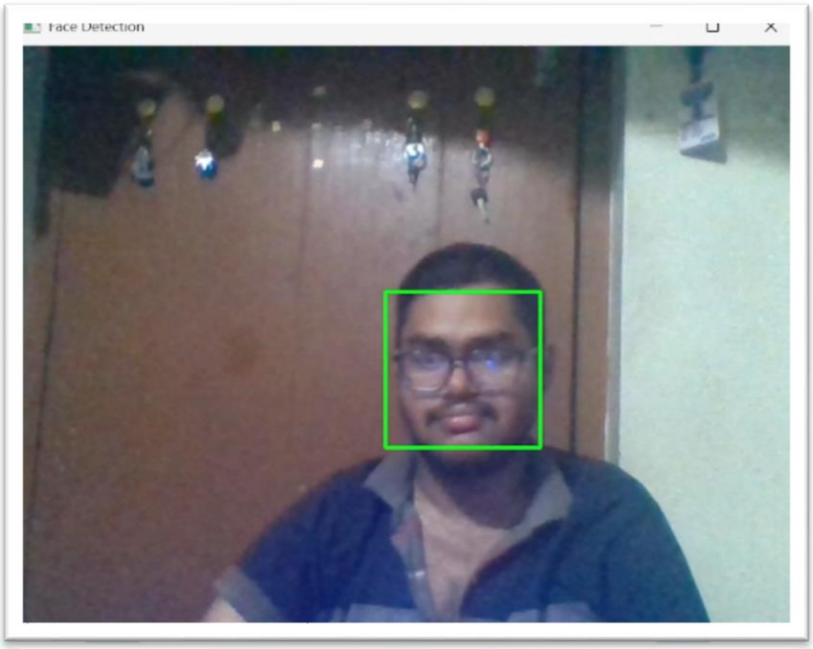
4. *Cascade Classifier:-*

The cascade classifier organizes multiple classifiers into sequential stages.

- **Early stages quickly eliminate regions that clearly do not contain faces.**
- **Later stages perform more detailed analysis only on promising regions.**
- **A detection window must pass every stage before being classified as a face.**

This cascading approach dramatically increases detection speed by

avoiding unnecessary computations.



(Fig-04 Face Detected)

Advantages of the Viola–Jones Algorithm:-

- **Fast real-time face detection.**
- **High detection accuracy for frontal faces.**
- **Low computational requirements.**
- **Efficient use of Hear features and integral images.**
- **Suitable for embedded and resource-constrained systems.**
- **Easy implementation using OpenCV.**

Limitations:-

- **Performs best on frontal faces.**
- **Accuracy decreases for rotated or partially occluded faces.**
- **Sensitive to extreme lighting conditions.**
- **Less effective than modern deep learning-based face detectors for complex real-world scenarios.**

Chapter 4

TRAINING AND TESTING

4.1 Training in OpenCV:-

Training is an essential stage in a face recognition system where the algorithm learns to identify individuals using a collection of labeled facial images. In this project, the Local Binary Pattern Histogram (LBPH) algorithm is used because of its high recognition accuracy and robustness under varying lighting conditions.

The training process begins by collecting multiple facial images of each person and assigning a unique ID to every individual. These images are converted into grayscale and processed to extract facial features. The extracted features are transformed into histograms, which are combined to form feature vectors representing each face. During recognition, the input face is processed similarly and compared with the stored feature vectors to determine the closest match.

4.2 Training the Classifier:-

OpenCV provides the Face Recognizer class to train and save facial recognition models. The training procedure consists of the following steps:

1. **Collect face images and assign labels.**
2. **Convert images into grayscale format.**
3. **Resize images if required.**
4. **Extract facial features.**
5. **Train the LBPH recognizer.**
6. **Save the trained model as a YAML/XML file for future recognition.**

The trained model stores the extracted facial features, enabling fast and accurate face identification without retraining the system each time**.**

4.3 Training Function:-

The. train() function is responsible for learning facial patterns from the training dataset.

Input Parameters

- **Collection of facial images.**
- **Corresponding labels (unique IDs).**

The function analyzes each labeled image and generates a recognition model that can later identify unknown faces.

4.4 Dataset Creation:-

A high-quality dataset is essential for improving recognition accuracy.

The dataset creation process includes:

- **Capturing multiple face images using a webcam.**
- **Detecting facial regions automatically.**
- **Saving cropped facial images into the dataset folder.**
- **Assigning a unique ID to each image.**

Increasing the number of training images generally improves recognition performance because the model learns more facial variations.

4.5 FaceRecognizer Class:-

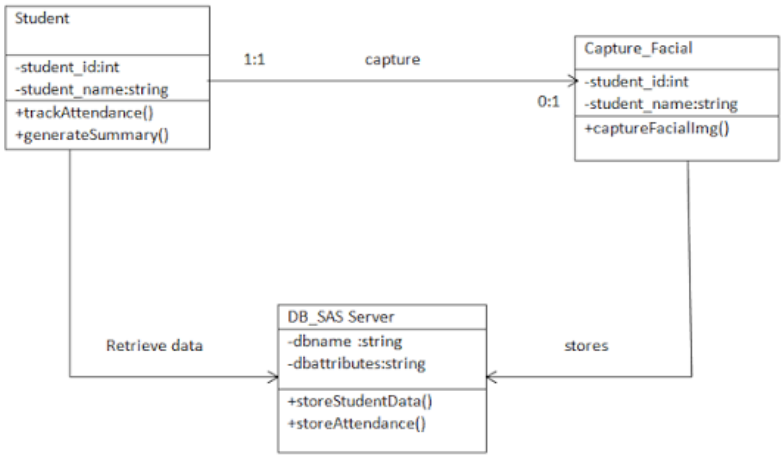
The Face Recognizer class in OpenCV provides a unified interface for different face recognition algorithms such as:

- **Eigenfaces**
- **Fisher faces**
- **LBPH**

Its primary functions include:

- **Training recognition models.**
- **Predicting identities.**
- **Saving and loading trained models.**
- **Managing labels associated with recognized individuals.**

This class simplifies the implementation of biometric face recognition systems.



(Fig-05 Face Recognizer Flow Diagram)

4.6 LBPH Recognizer:-

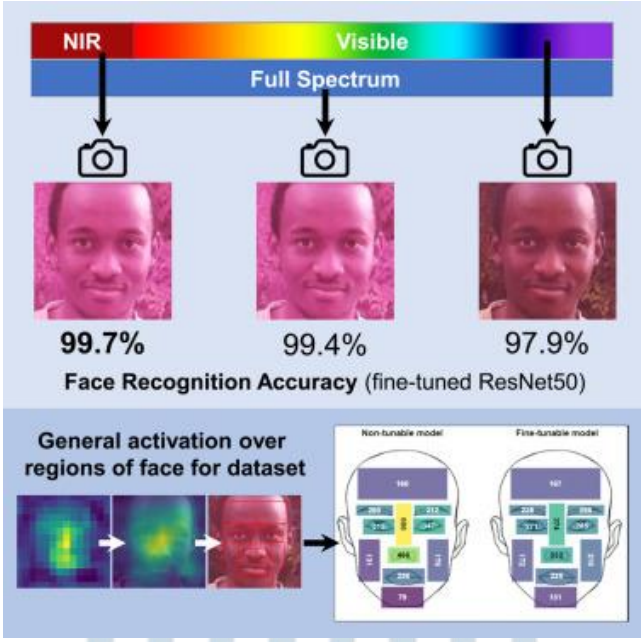
The Local Binary Pattern Histogram (LBPH) algorithm is used in this project because it performs well even with changes in lighting and facial expressions.

Working Process:-

The LBPH algorithm performs the following operations:

1. **Define parameters such as radius, neighbors, Grid X, and Grid Y.**
2. **Generate Local Binary Patterns by comparing neighboring pixel values.**
3. **Divide the image into multiple grids.**
4. **Compute histograms for each grid.**
5. **Merge all histograms into a single feature vector.**
6. **Store feature vectors along with corresponding labels.**

These feature vectors form the recognition model used during prediction.



(Fig-06 Face Recognizer Color Pattern)

LBPH Parameters

Parameter Description:-

Radius	**Radius used to calculate Local Binary Patterns**
Neighbors	**Number of neighboring pixels considered**
Grid X	**Number of horizontal grids**
Grid Y	**Number of vertical grids**

4.7 Prediction Function:-

The .predict() function identifies an unknown face by comparing its extracted features with the trained database.

The prediction process includes:

- **Capturing the input face.**
- **Extracting LBPH features.**
- **Comparing with stored histograms.**
- **Calculating the similarity score.**
- **Returning the predicted label and confidence value.**

Lower distance values indicate a better match and higher recognition

confidence.

4.8 Face Recognition Implementation:-

After training, the recognition program loads the saved trainer.yml model and processes live webcam images.

The system performs the following operations:

- **Detects faces using the Viola–Jones algorithm.**
- **Extracts facial features.**
- **Compares them with the trained model.**
- **Displays the recognized person's name.**
- **Marks attendance automatically if a valid match is found.**

This enables real-time facial recognition with minimal user interaction.

4.9 Applications:-

The developed face recognition system has numerous practical applications, including:

- **Automatic attendance management.**
- **Smartphone face unlock.**
- **Secure access control systems.**
- **Banking and digital payment authentication.**
- **Law enforcement and surveillance.**
- **Healthcare patient identification.**
- **Smart office security.**
- **Airport immigration systems.**

These applications improve security, reduce manual effort, and enhance user convenience.

Chapter 5

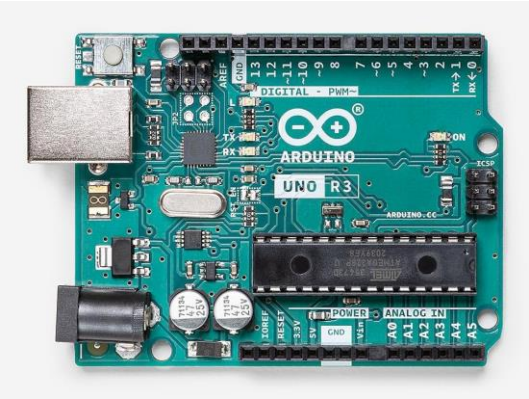
COMPONENTS REQUIRED

➢ Ultrasonic Sensor

(FIG-07 ULTRASONIC SENSOR)

(The HC-SR04 Ultrasonic Distance Sensor is a sensor used for detecting the distance to an object using sonar).

➢ Arduino UNO



(FIG-08 ARDUINO UNO BOARD)

(The Arduino UNO is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino cc.The board is equipped with sets of digital and analog input/output pins).

➢ Buzzer-

(FIG-09 BUZZER)

(A piezo buzzer is basically a tiny speaker that you can connect directly to an Arduino; from the arduino you can make sounds with a buzzer by using tone).

➢ 9V Battery –

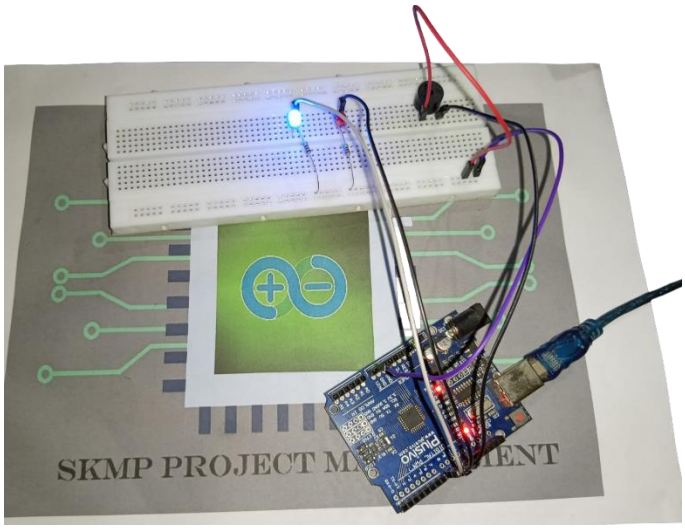
(FIG-10 BATTERY)

The battery is used to provide power supply to the system for its operation.

➢ LED

(FIG-11 LED LIGHT)

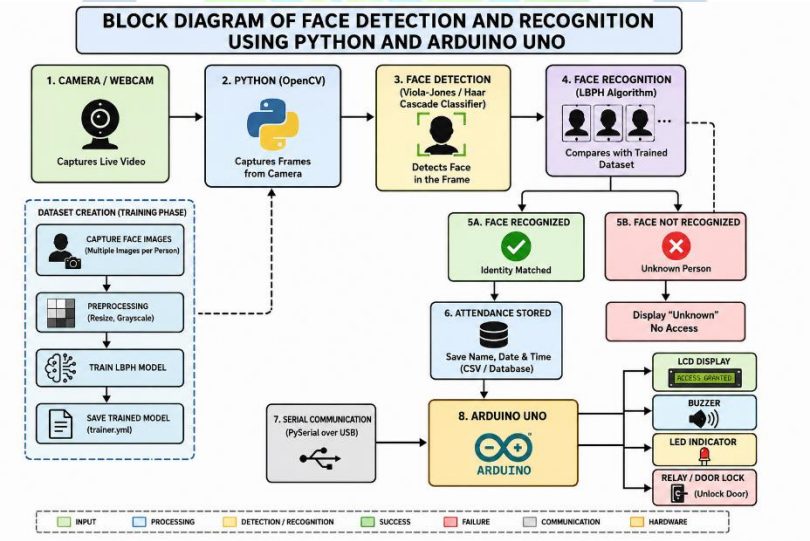
(A light-emitting diode is a semiconductor light source that emits light when current flows through it).



(Fig-12 Final Hardware Connection

Chapter 5

CODE & BLOCK DIAGRAM



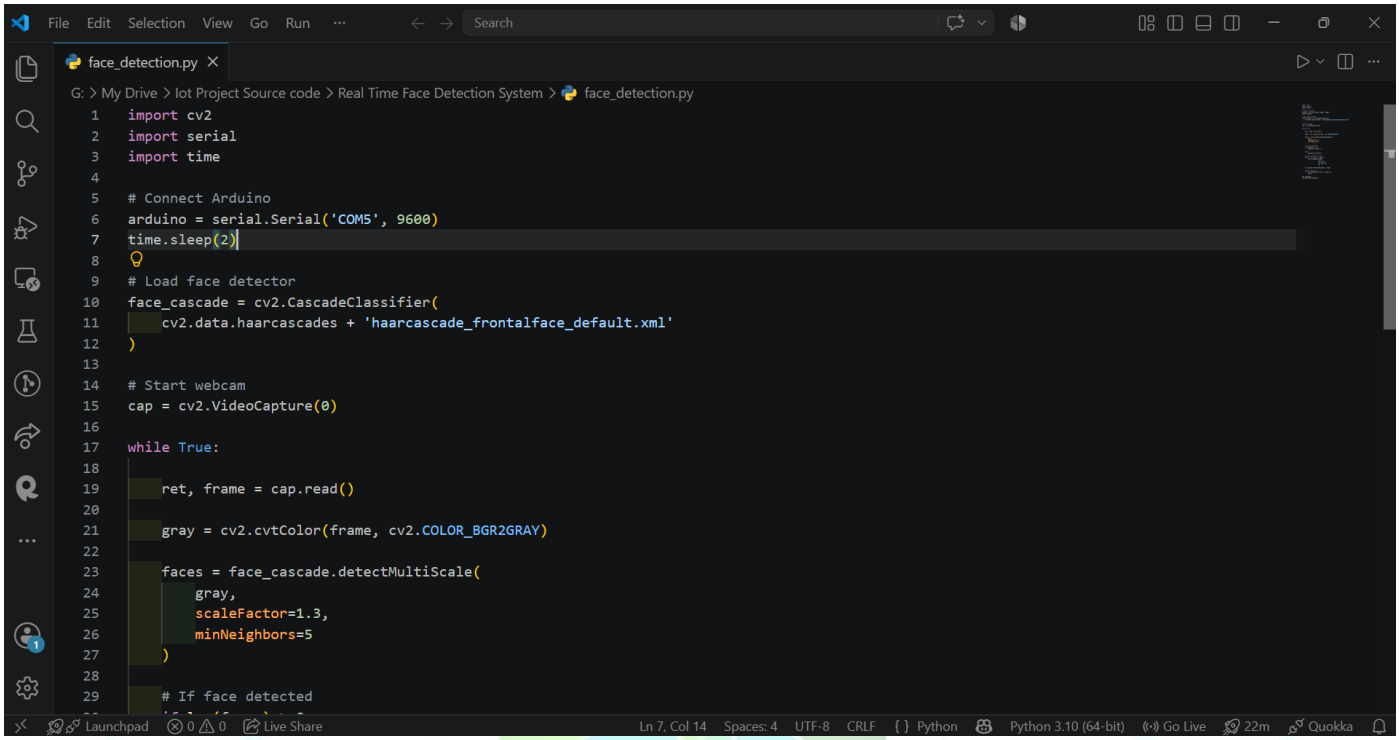
(Fig-13 Block Diagram of Face Detection and Recognition System)

Code:-(Python Code)

```
import cv2
import serial
import time

# Connect Arduino
arduino = serial.Serial('COM5', 9600)
time.sleep(2)

# Load face detector
face_cascade = cv2.CascadeClassifier(
    cv2.data.haarcascades + 'haarcascade_frontalface_default.xml'
)

# Start webcam
cap = cv2.VideoCapture(0)

while True:

    ret, frame = cap.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = face_cascade.detectMultiScale(
        gray,
        scaleFactor=1.3,
        minNeighbors=5
    )
```

```
    # If face detected
    if len(faces) > 0:
        arduino.write(b'1')

    else:
        arduino.write(b'0')

    # Draw rectangle on face
    for (x, y, w, h) in faces:
        cv2.rectangle(frame,
                      (x, y),
                      (x+w, y+h),
                      (0, 255, 0),
                      2)

    cv2.imshow("Face Detection", frame)

    # Press Q to exit
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

Code Snippets-



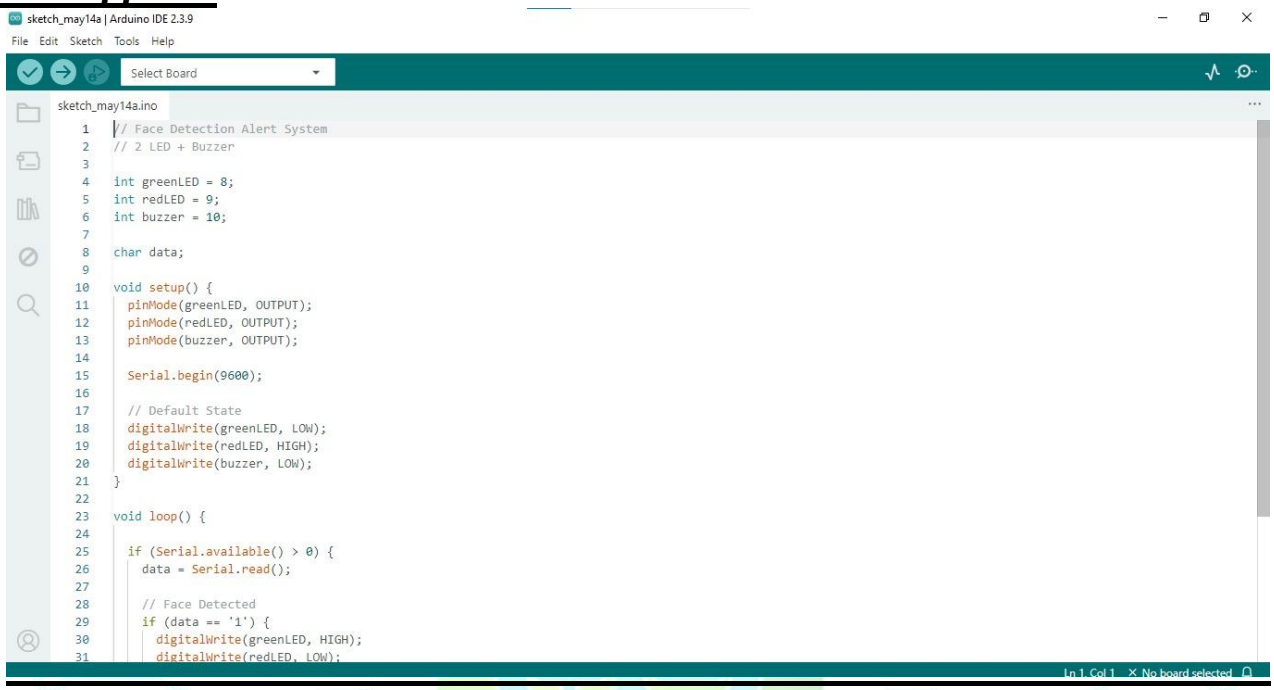
(Fig-13 Code Snippet Of Code Python)

Code (Arduino Uno):-

```
// Face Detection Alert System
// 2 LED + Buzzer

int greenLED = 8;
int redLED = 9;
int buzzer = 10;

char data;

void setup() {
  pinMode(greenLED, OUTPUT);
```

```
  pinMode(redLED, OUTPUT);
  pinMode(buzzer, OUTPUT);

  Serial.begin(9600);

  // Default State
  digitalWrite(greenLED, LOW);
  digitalWrite(redLED, HIGH);
  digitalWrite(buzzer, LOW);
}

void loop() {

  if (Serial.available() > 0) {
    data = Serial.read();

    // Face Detected
    if (data == '1') {
      digitalWrite(greenLED, HIGH);
      digitalWrite(redLED, LOW);
      digitalWrite(buzzer, HIGH);
    }

    // No Face Detected
    else if (data == '0') {
      digitalWrite(greenLED, LOW);
      digitalWrite(redLED, HIGH);
      digitalWrite(buzzer, LOW);
    }
  }
}
```

Code Snippets-



(Fig-14 Code Snippet Arduino uno Processing Software)

Last Chapter

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion:-

This project successfully demonstrates the design and implementation of a real-time face recognition system using Python, OpenCV, and the LBPH algorithm. The system accurately detects human faces, extracts facial features, and compares them with a trained database to recognize authorized individuals.

The integration of the Viola–Jones face detection algorithm with the LBPH recognizer provides a fast, reliable, and cost-effective biometric authentication solution. Compared to traditional attendance methods, the proposed system eliminates manual errors, prevents proxy attendance, and improves efficiency.

Overall, the project highlights the effectiveness of computer vision and machine learning techniques in developing secure and automated face recognition applications**.**

5.2 Future Scope:-

The future scope of this project includes several enhancements:

- **Integration with cloud databases for large-scale deployments.**
- **Deep learning-based face recognition using CNNs.**
- **Real-time attendance through mobile applications.**
- **Multi-camera surveillance systems.**
- **Mask-aware face recognition.**
- **Emotion and expression analysis.**

- **Smart city surveillance.**
- **AI-powered identity verification.**
- **Integration with IoT-based security systems.**
- **Contactless authentication for banking and healthcare.**

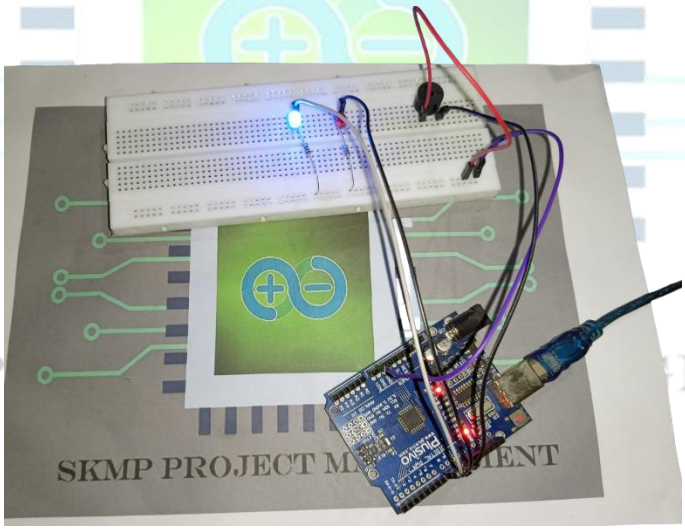
These improvements can significantly enhance system accuracy, scalability, and real-world usability.

PROJECT LINKS

- Project Video Link:- https://youtu.be/lYO5gyBTMs4?si=vXUu3zoXeuSXXVha

- Project Folder Link:- https://drive.google.com/drive/folders/1xqrG5NUDxCtHQTSE4MdZPcqb8i3wPJga?usp=sharing

- Project Website Link:- https://skmp-project-management.vercel.app/#hero



References

[1] Schneiderman.United States of America Patent U.S. Patent No. 8,457,367, 2013.

[2] R. J. Baron, "Mechanisms of human facial recognition," International Journal of ManMachine Studies.

[3] M. Nixon, ""Eye Spacing Measurement for Facial Recognition"," International Society for Optics and Photonics., vol. (Vol. 575), (19 December 1985).

[4] H. &. Y. J. Yu, "A direct LDA algorithm for high-dimensional data—with application to face recognition," 2001.

SKMP PROJECT MANAGEMENT

==========

SKMP PROJECT MANAGEMENT

DEPARTMENT OF IOT