Web-Based Facial Recognition System

A Project Report Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

G.Saiteja (2010030055)

Md.Adnan (2010030236)

G.Samanth(2010030272)

M.Chaithanya (2010030366)

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING
K L DEEMED TO BE UNIVERSITY
AZIZNAGAR, MOINABAD , HYDERABAD-500 075

MARCH 2023

BONAFIDE CERTIFICATE

This is to certify that the project titled **Web-Based Facial Recognition System** is a bonafide record of the work done by

G.Saiteja (2010030055)

Md.Adnan (2010030236)

G.Samanth(2010030272)

M.Chaithanya (2010030366)

in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **COMPUTER SCIENCE AND ENGINEERING** of the **K L DEEMED TO BE UNIVERSITY, AZIZNAGAR, MOINABAD , HYDERABAD-500 075**, during the year 2022-2023.

P.Sree Lakshmi
Project Guide

Dr.Arpitha Gupta
Head of the Department

Project Viva-voce held on ________________________

Internal Examiner

External Examiner

ABSTRACT

A Web-Based Facial Recognition System is a software application that uses facial recognition technology to identify individuals from their facial features. This system is accessible through a web browser and can be used for various purposes, such as security, identification, and authentication. The system works by capturing an image of a person's face using a camera or uploading a pre- existing image.

The system then analyzes the facial features, such as the distance between the eyes, the shape of the nose, and the contours of the face, to create a unique biometric template. This template is then compared to a database of stored templates to identify the individual. Web-based facial recognition systems have a wide range of applications, including security, border control, access control, and customer service. They offer a quick and convenient way to authenticate individuals, without the need for physical identification cards or passwords. However, the use of facial recognition technology has also raised concerns around privacy and surveillance.

There have been instances of facial recognition technology being used without consent, and there is a risk of bias and discrimination against certain groups of people. Therefore, the development and deployment of web-based facial recognition systems require careful consideration of ethical and legal issues, and the implementation of appropriate safeguards to protect the privacy and rights of individuals.

Information and Communication Technology usage has witnessed rapid growth in the past decade all around the world. All web-based systems that have users and store personal information about the users require a mechanism to keep track of their users' information. commonly every user of the system is assigned an instance in the database that represents them (their identity). To protect the user identity, an authentication and authorization mechanism is implemented to control access to certain information. The most common method in web-based systems is authentication using passwords. passwords have well-known disadvantages in both usability and security. This leads to promotion of Face-based authentication.

ACKNOWLEDGEMENT

We would like to thank the following people for their support and guidance without whom the completion of this project in fruition would not be possible.

Dr.P Sree Lakshmi, our project guide, for helping us and guiding us in the course of this project .

Dr.Arpita Gupta, the Head of the Department, Department of Computer Scinece And Engineering.

Our internal reviewers, **Dr.P Sree Lakshmi** , **Dr.Sumit Hazra** , **Dr.SR Mugunthan** for their insight and advice provided during the review sessions.

We would also like to thank our individual parents and friends for their constant support.

TABLE OF CONTENTS

Title **Page No.**

List of Figures

Chapter 1
Introduction

1.1 Background of the Project

Information and Communication Technology usage has witnessed rapid growth in the past decade decade all around the world. A bigger percentage of the population has laptops, personal computers, and smart phones making it easy to access the internet and thus changing lives of millions of people [1]. All web-based systems that have users and store personal information about the users require a mechanism to keep track of their users' information. Most 1Corresponding author. katraxuk@gmail.com commonly every user of the system is assigned an instance in the database that represents them (their identity). To protect the user identity, an authentication and authorization mechanism is implemented to control access to certain information. The most common method in web-based systems is authentication using passwords. Users regularly provide a combination of their username and password through a form to access a remote account [9]. Passwords have well-known disadvantages in both usability and security. This leads to promotion of biometric-based authentication. The primary motivation of biometric authentication is usability: users are not required to remember the passwords, there is nothing for them to carry, biometric systems are generally easy to use, and scalable in terms of the burden exerted onto the users. Biometric technology can be used to control the risk of sharing, forgetting, losing, and embezzlement of passwords

1.1.1 About the Project

Face recognition is a popular and important technology integrated in many applications such as payment, door unlock, video monitoring systems, etc. A face recognition system is a technology that can identify and verify people from digital images and footage. The state-of-the-art face recognition approaches for web authentication include luxury and simplified from electronic Sid. These are both provided as commercial products. To recognize faces (human faces), Images are taken using a digital camera. the first step is to perform face detection, this is performed as a preprocessing step to locate the face in an image. Web-based facial recognition system is a software application that uses advanced biometric technology to identify individuals through their facial features. It is a rapidly growing field with applications in various sectors such as security, identification, and authentication. With the increase in demand for quick and accurate identification methods, web-based facial recognition systems have become a popular solution for many businesses and organizations.

1.2 Problem Statement

Attacks on a systems like information theft, DDoS attacks, ransomware, or other malicious activities can originate either from offine or online spheres and can crash a system.The consequences of information theft can be even worse when organizations store sensitive or confidential information like credit card numbers or customer information.1. Distributed Data2. Non-Relational Databases3. Endpoint Vulnerabilities4. Data Mining Solutions5. Access Controls

1.3 Objectives

Improve the security of the application by implementing facial authentication as a robust and reliable user verification method. Provide a user-friendly login experience by enabling users to access the application quickly and effortlessly through facial recognition. Enable users to securely and conveniently access online services using their facial features as authentication. Achieve a high level of accuracy in facial recognition to reduce false positives and negatives. Create a user-friendly interface for capturing facial images, making it easy for users to register and authenticate themselves. Develop an efficient and fast authentication process to provide a seamless user experience.

1.4 Scope of the Project

The need for facial recognition is multifold. I am trying to put up some points here in a concise way. Make sure to read the article to the end to get a complete conceptual understanding and a detailed implementation walkthrough.

Faster than traditional methods:

Facial authentication method is very fast than the traditional means of authentication. You just have to click on a button to start the authentication process and within a millisecond, it is done. In the traditional email password-based methods, you have to add your details line by line. Sometimes after successful log-in, you are greeted with a captcha. How irritating it is! The only requirement of the Facial authentication technique is a camera. All smartphones nowadays have a camera by default. All desktops also have some sort of webcams present. Therefore, users don't need any specialized hardware to use this service.

Reduce impersonation on social platforms:

The most important feature of facial authentication is that it can prevent impersonation. On social platforms, many people create fake accounts by impersonating someone. This can be very risky if the fake account holder commits some type of digital crime. With the help of facial recognition, social platforms can recognize if the account someone was trying to access, actually belongs to them.

Reduce bots and automated scripts:

Bots and automated scripts are introduced to help people get rid of repetitive tasks. But people also used them in a different way to spam others. Daily you come across many bots and automated scripts in your digital life that you don't even notice or realize. To prevent this, some websites use captcha. Using facial recognition, this problem can also be solved as bots have no face to authenticate .

Privacy-focused:

Privacy is a very sensitive topic for all of us. We all become a little bit concerned when someone asks you to authenticate using your face data. But as we are using FaceIO in this tutorial, the authentication process is purely end-to-end encrypted. In the backend, they store only the hash of your facial features. They are completely GDPR and CCPA-compliant. So, you can trust them to store your data safely.

Chapter 2

Literature Review

2.1 An Example On How To Add Pictures

The primary purpose of this paper review is to find the solutions provided by others author and consider the imperfection of the system proposed by them, give the best solutions. In [18] Kawaguchi introduced a lecture attendance system with a new method called continuous monitoring, and the student's attendance marked automatically by the camera which captures the photo of a student in the class. The architecture of the system is simple since two cameras equipped with the wall of the class. The first one is a capturing camera used to capture the image student in the class and the second camera is sensor camera is used to getting the seat of a student inside the class and the camera capturing will snap the image of the student. The system compares the picture taking from a camera capturing images and faces in the database done much time to perfect the attendance. Other paper proposed by [2] introduced a real-time computer vision algorithm in automatic attendance management system. The system installed the camera with non-intrusive, which can snap images in the classroom and compared the extracted face from the image of the camera capturing with faces inside the system.

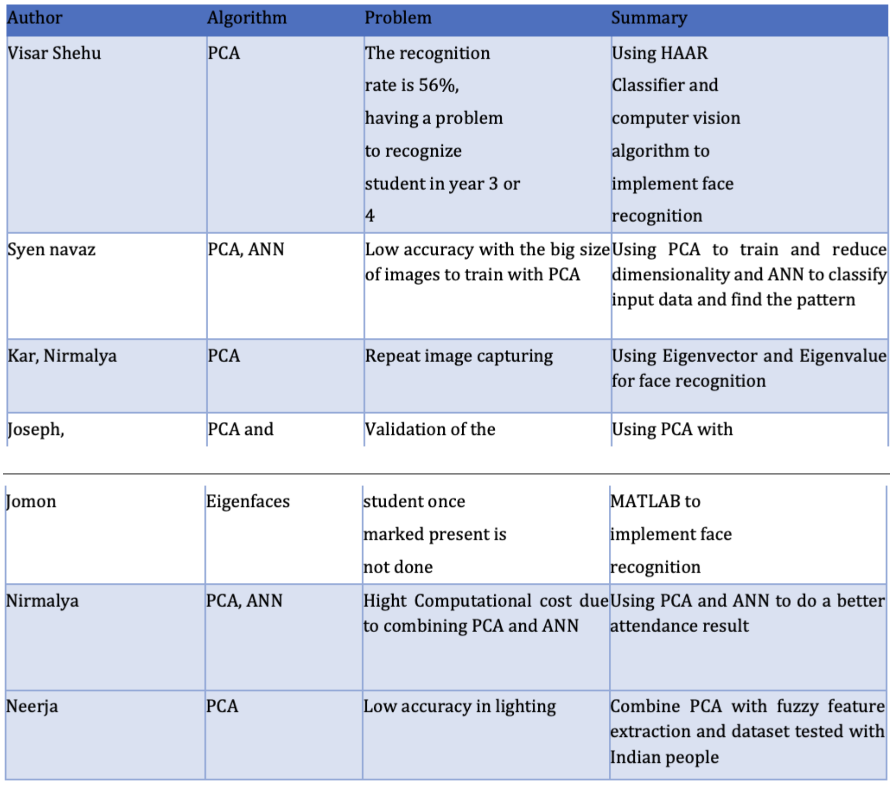
This system also used machine learning algorithm which are usually used in computer vision. Also, HAAR CLASSIFIERS used to train the images from the camera capturing. The face snap by the camera capturing will convert to grayscale and do subtraction on the images; then the image is transferred to store on the server and processing later. In 2012 N. Kar [19] introduced an automated attendance management system using face recognition technique which used the Principal Component Analysis

To implementation the system, use two libraries such OpenCV is a computer vision library and FLTK(Light Tool Kit. Both of this libraries helped the development such as OpenCV support algorithm[20] and FLTK [21] used to design the interface. In the system, there are Request Matching and Adding New fact to Database. In Request Matching, the first step is open the camera and snap the photo after the extraction the frontal face. The next step is recognizing the face with the training data and project the extracted face onto the Principal Component Analysis. The final step displays the nearest face with the acquired images. Apart from that, adding a new face into the database is snap the photo after that extract the frontal face images and then perform the Haar cascade Method to find the perform the Principal Component Analysis Algorithm. The final step is storing the information inside the face XML file. The system is focused on the algorithm to improve the face detection from acquired images or videos. In [3] the author also proposed a system which implements automatic attendance using face recognition. The system which can extract the object in the face such nose, mouth by using MATLAB with Principal Component Analysis (PCA). The system [7] designed to resolve the issues of attendance marking system such as timeconsuming. As the result of the experiment show that this paper, the system can recognize in case the dark background or difference view of the face in the classroom. Jyotshana Kanti [4] proposed a smart attendance marking system which combines two differencing algorithms such Principal Component Analysis and Artificial Neural Network. The purpose of the author is to solve the traditional attendance marking system and to resolve the time-consuming. In the system implement with Principal Component Analysis, it does an extraction and identify the similarities of the face database and acquire images. Artificial Neural Network is used to solve the problem of the input data or learn from the input data, and the expect value. In the system implemented by the author using back propagation algorithm and combines with mathematical function to perform in that system. As a result, written by the author research, it shows that the system can use to recognize in a different environment.

In [22] Priyanka Thakare proposed a method using Eigenface and Principal Component Analysis which has the architecture as the following step. The camera needs to install in the front which can capture an entire face of the student inside the class. The first phase after the camera has been captured; the captured image was transferred into the system as an input. The image capture from the camera sometimes come with the darkness or brightness which need to do an enhancement on it such as convert to gray image. The next step, Histogram Normalization is used in this system remove the contrast of the image. It is easy to recognize when has the student sit in the back row. The Median filter is used to remove noise from the image in case the camera is high definition camera, but sometimes it still contains the noise. The author also implements with skin classification which changes all the pixel to black except the pixel are close to the skin. [1] Student Attendance System using Face Recognition: Samridhi Dev, Tushar Patnaikb(2020) In this paper the system was tested on three different algorithms out of which the KNN algorithm proved to be better with the accuracy of 99.27 expressions, the distance of students from the camera. The system stands up to the expectations even when the image contains faces with beards and spectacles and without beard and spectacles. proposed system evinced to be magnificent to recognize faces having two years of difference. [2] AUTOMATED SMART ATTENDANCE SYSTEM USING FACE RECOGNITION: Kolipaka Preethi,swathy vodithala (2021) The proposed method consists of different stages to mark the attendance live A. Face Detection B. DataSet Creation and Training C. Face Recognition and Updating attendance [3] FAREC - CNN Based Efficient Face Recognition Technique using Dlib: Sharma S, Karthikeyan Shanmugasundaram, Sathees Kumar Ramasamy(2016) The paper used trained feature models from Convolutional Neural Network; model has the features of the entire labels of the face recognition systems. The test images are validated against these models and provide the maximum probability value among the labels and claims that to be the person. FAREC takes 20 epoch for converging learning rate from 0.01 and produce 96accuracy for FRGC and False acceptance rate of 0.1very soon as before 5th epoch. The following figure 9 and figure 10 showing the learning rate convergence and accuracy of FAREC. [4] FaceTime – Deep Learning Based Face Recognition Attendance System: Marko Arsenovic, Srdjan Sladojevic, Andras Anderla, Darko Stefanovic (2017) The model was trained based on a small number of images per employee and using the proposed method of augmentation. This led to the enlargement of the initial dataset and the improvement of the overall accuracy. By analyzing the images stored in the database during the acquisition period, it could be seen that the light conditions influenced the recognition process. Most of the images predicted incorrectly were exposed to the daylight while the door was open. This could potentially be corrected by applying gradient transformation on the images. A small number of images affected by noise of the unknown cause were predicted correctly. The overall accuracy could be improved by applying on time interval automatic re-training of the embedding deep CNN together with the newly gathered images predicted by the model with the high accuracy rate. [5] Real Time Attendance System Using Face Recognition Technique: Mayank Srivastava, Amit Kumar, Aditya Dixit, Aman Kumar (2020) In this, project experimented with 30 faces as a training set of 7 people for measurement of accuracy of the system. The Extract () function shows a sample binary image obtained with the help of face extracting frame work detection method by Paul – Viola. The results shows that with respect to face detection and recognition rate, on increasing the face angle, camera decreases. Introducing entry and exit times, the authors intend to develop an attendance management system for colleges which is based on facial recognition technology. Every student's attendance is collected by the system through constant observation at the entry and exit points. The results of our initial experiment performed better in performance assessment than traditional black and white display systems. This system is mainly developed for face recognition from images or video frames.

2.2 Example on Table Usage

Various technical papers have shown the studies of number each algorithm. This table covers five papers that are related to this area, and an overview of all the papers survey is given here we give mainly their problems and limitations we also give a summary and conclusion every researcher has proposed in those papers.



Author	Algorithm	Problem	Summary
Visar Shehu	PCA	The recognition rate is 56%, having a problem to recognize student in year 3 or 4	Using HAAR Classifier and computer vision algorithm to implement face recognition
Syen navaz	PCA, ANN	Low accuracy with the big size of images to train with PCA	Using PCA to train and reduce dimensionality and ANN to classify input data and find the pattern
Kar, Nirmalya	PCA	Repeat image capturing	Using Eigenvector and Eigenvalue for face recognition
Joseph, Jomon	PCA and Eigenfaces	Validation of the student once marked present is not done	Using PCA with MATLAB to implement face recognition
Nirmalya	PCA, ANN	Hight Computational cost due to combining PCA and ANN	Using PCA and ANN to do a better attendance result
Neerja	PCA	Low accuracy in lighting	Combine PCA with fuzzy feature extraction and dataset tested with Indian people

Figure 2.1: Literature Review

Chapter 3

Proposed System

3.1 System Requirements

System Specifications Specifications are standard data sets that summarize system requirements. Business analysts, also known as systems analysts, assess the business needs of customers and stakeholders to identify and recommend solutions to business problems.

- Company specifications define what needs to be delivered or completed in order to have value.
- Product specifications define a system's or a product's characteristics
- The aim of a feasibility study is to determine whether adding new modules and debugging an existing device is technically, operationally, and economically feasible. If there are limitless resources and time, every system is feasible. There are some elements of the preliminary investigation's feasibility report.

Hardware requirements:

Processer : Min i5

Ram : min 4GB

Hard Disk : min 200 GB

Software requirements:

Operating System OS : Windows

Tools/Technologies : opencv, numpy, pandas

IDE : PyCham/Jupiter

Programming language: python

3.2 Design of the System

A project description is a high-level overview of why you're doing a project. The document explains a project's objectives and its essential qualities. Think of it as the elevator pitch that focuses on what and why without delving into how.

3.2.1 Subsection Title

A project description is a high-level overview of why you're doing a project. The document explains a project's objectives and its essential qualities. Think of it as the elevator pitch that focuses on what and why without delving into how.

3.3 Algorithms and Techniques used

There are two main parts to a CNN architecture A convolution tool that separates and identifies the various features of the image for analysis in a process called as Feature Extraction. The network of feature extraction consists of many pairs of convolutional or pooling layers. A fully connected layer that utilizes the output from the convolution process and predicts the class of the image based on the features extracted in previous stages. This CNN model of feature extraction aims to reduce the number of features present in a dataset. It creates new features which summarises the existing features contained in an original set of features. There are many CNN layers as shown in the CNN architecture diagram

CNNs are particularly effective in handling visual data because they are structured to mimic the human visual system. They consist of multiple layers, including convolutional layers, pooling layers, and fully connected layers. Here's a brief overview of these components:

1. Convolutional Layer:

Convolution is a mathematical operation that involves sliding a small filter (also known as a kernel) over the input image to extract local features. This operation helps the network learn features like edges, textures, and patterns in the input data. This layer is the first layer that is used to extract the various features from the input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size MxM. By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter (MxM). The output is termed as the Feature map which gives us information about the image such as the corners and edges. Later, this feature map is fed to other layers to learn several other features of the input image. The convolution layer in CNN passes the result to the next layer once applying the convolution operation in the input. Convolutional layers in CNN benefit a lot as they ensure the spatial relationship between the pixels is intact.

2. Pooling Layer:

Pooling layers downsample the feature maps produced by the convolutional layers, reducing their spatial dimensions. This helps reduce the computational load and provides some degree of translational invariance, making the network more robust to small changes in object position within the input. In most cases, a Convolutional Layer is followed by a Pooling Layer. The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs. This is performed by decreasing the connections between layers and independently operates on each feature map. Depending upon method used, there are several types of Pooling operations. It basically summarises the features generated by a convolution layer. In Max Pooling, the largest element is taken from feature map. Average Pooling calculates the average of the elements in a predefined sized Image section. The total sum of the elements in the predefined section is computed in Sum Pooling. The Pooling Layer usually serves as a bridge between the Convolutional Layer and the FC Layer. This CNN model generalises the features extracted by the convolution layer, and helps the networks to recognise the features independently. With the help of this, the computations are also reduced in a network.

3. Fully Connected Layer:

After several convolutional and pooling layers, a CNN typically ends with one or more fully connected layers. These layers are similar to the ones in a traditional neural network and are used for making final predictions based on the high-level features learned by the previous layers. The Fully Connected (FC) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers. These layers are usually placed before the output layer and form the last few layers of a CNN Architecture. In this, the input image from the previous layers are flattened and fed to the FC layer. The flattened vector then undergoes few more FC layers where the mathematical functions operations usually take place. In this stage, the classification process begins to take place. The reason two layers are connected is that two fully connected layers will perform better than a single connected layer. These layers in CNN reduce the human supervision.

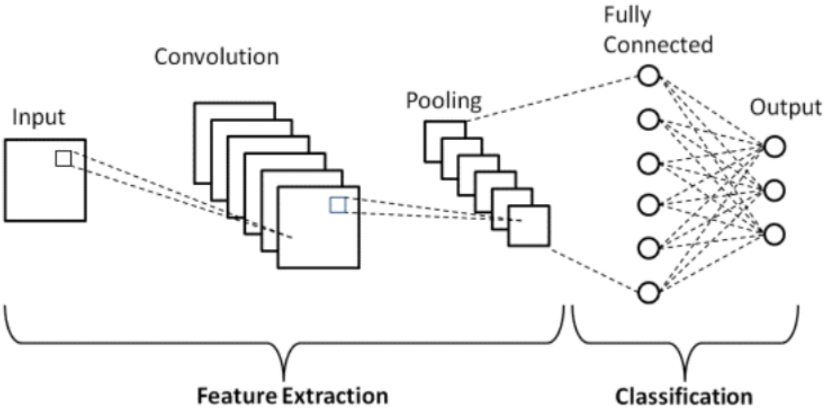


Figure 3.1: CNN architecture

Chapter 4

Implementation

4.1 Tools and Technologies used

Pythonis a very powerful programming language used for a variety of applications. Over time, a large community centered around this open source language has created many tools that work effectively with Python. Several tools have been created specifically for computer science in recent years. As a result, analyzing data with Python is easier than ever. Python is a programming language that helps you to work more easily and efficiently with programmes. Python is divided into two versions: Python 2 and Python 3. The two are diametrically opposed. Python is a programming language that can be used in a variety of ways. Many functions allow functional and side-oriented programming, and object-oriented and structured programming are completely supported'.[2]. Many other designs, including contract design and logical programming, are supported by add-ons. Python uses 'a combination of dynamic typing 'and a garbage collector' to define links for reference accounts and memory management. It also has a variable mode and dynamic name resolution (late binding) that binds names' when driving. Python design provides some support for functional programming in the Lisp tradition'. It has filtering, display and collapsing functions. The last set of vocabulary, dictionary, and generator expressions. The standard library contains two modules (IT tools and functional tools) that operate functional tools borrowed from Haskell and Standard ML. Python is designed as an easy-to- read language. The format is not visually cluttered and often uses English keywords, while other languages use punctuation. Unlike many other languages'[2], you don't use curly braces to separate blocks, and the semicolon after the statement is optional. There are fewer syntax exceptions and special cases than C or Pascal'[2].

Artificial Intelligence

Al allows machines to think, which means they will be able to make decisions without the need for human interference. It's a wide field of computer science that simulates human intelligence in computers. So it's not about training a computer to drive a car by obeying traffic signals; it's also when the computer learns to show human-like signs of road rage.

Machine Learning

Machine Learning is a subset of artificial Intelligence that employs mathematical learning algorithms to create systems that can learn and develop on their own without being specifically programmed. Supervised, Unsupervised, and Reinforcement learning are the three types of machine learning algorithms.

Supervised Learning

In supervised learning, you already know what a set of data will look like, with an idea of the relationship between the inputs and the outputs.

Unsupervised Learning

This allows you to deal with problems with little or no idea how the results will appear. You can collect data structures that don't need to know the effect of the variable. You can achieve this structure by grouping data based on the relationship between variables in the data. Reinforcement Learning Reinforcement learning is taking appropriate steps to maximize rewards in a given situation. It is used by various programs and machines to find the optimal motion or path for a given situation. Reinforcement learning differs from supervised learning in that supervised learning has a solution key in the training data, so the model is trained with the correct solution. You have done this.Without a training data set, he has to learn from his own experience.

Deep Learning

Deep learning, which mimics the way the human brain processes data and generates patterns for use in decision making, is the function of artificial intelligence (Al). Deep learning is an artificial intelligence branch of machine learning that uses networks to monitor learning based on unstructured or unstructured data. Deep neural networks or deep neural learning are other terms for the same thing. Deep learning is a machine learning methodology that is based on how the human brain filters information and learns from examples. It aids a computer model's ability to process input data across layers in order to predict and classify data. Deep learning is often used in applications that people do because it handles knowledge in the same way that a human brain does. It's the technology that allows self-driving cars to identify a stop sign and differentiate between a pedestrian and a lamp post. Deep neural networks are the name given to the majority of deep learning approaches that use neural network architectures. Deep Learning is a multi-neural network architecture that contains a large number of parameters and layers that is designed to simulate the human brain.

NumPy

NumPy is an open source Python digital library. It contains multidimensional matrices and matrix data structures. It Can be used to perform several mathematical operations on matrices Such as trigonometric, statistical and algebraic procedures. Therefore, the library contains a large number of math, algebra and transformation functions. NumPy is an extension of Numeric and Numarray. It also includes a random number generator. It is a cover for C-based libraries.

Matplotlib

Matplotlib is a popular Python library used for creating static, animated, and interactive visualizations in a wide range of formats. It provides a high-level interface for drawing attractive and informative graphs, charts, and plots. Matplotlib is often used in data analysis, scientific research, and data visualization projects. Some of the key features and capabilities of Matplotlib include the ability to create line plots, scatter plots, bar charts, histograms, and more. You can customize almost every aspect of a plot, from the colors and labels to the axes and legends. Matplotlib is highly extensible, allowing you to create complex and customized visualizations. It can also be used in combination with other libraries such as NumPy for numerical operations and Pandas for data manipulation.

OpenCV

OpenCV, which stands for ”Open Source Computer Vision Library,” is an open-source computer vision and machine learning software library. It is designed to provide tools and functions for a wide range of computer vision tasks, including image and video analysis, object recognition, machine learning, and image processing. OpenCV has interfaces for Python, Java, and other languages, making it a versatile choice for computer vision projects. Some of the key features and capabilities of OpenCV include:

1.Image Processing

2.Object Detection and Recognition

3.Machine Learning

4.Camera Calibration

5.Video Analysis

AES -Advanced Encryption Standard

AES is known for its strong security and efficient performance and is used in a wide range of applications, including data encryption, secure communications, and file protection.

Here is an overview of how the AES algorithm works:

Key Expansion:

AES operates on blocks of data, typically 128 bits (16 bytes) at a time. The encryption key is also 128, 192, or 256 bits in length, depending on the desired security level. The key expansion process generates a set of round keys from the original encryption key. These round keys are used in the encryption and decryption processes. Initial Round:

The plaintext block is XORed with the first round key. Rounds:

AES operates on multiple rounds, with the number of rounds depending on the key size: 10 rounds for a 128-bit key, 12 rounds for a 192-bit key, and 14 rounds for a 256-bit key. In each round, the following transformations are applied: SubBytes: A byte-by-byte substitution of the data using a predefined S-box (substitution box).

ShiftRows: The rows of the state are shifted cyclically. MixColumns: Each column is mixed to provide diffusion and confusion. AddRoundKey: The round key for that round is XORed with the state data. Final Round:

The final round is similar to the other rounds but lacks the MixColumns operation. Output:

The resulting state at the end of the final round is the ciphertext. To decrypt data encrypted with AES, the process is reversed. The ciphertext is XORed with the round keys in reverse order, and the inverse of the operations (InvSubBytes, InvShiftRows, InvMixColumns) is applied in each round.

Key points to note about AES:

AES is a symmetric-key encryption algorithm, meaning the same key is used for both encryption and decryption. The security of AES is based on the key length, and it is considered highly secure. AES-128, AES-192, and AES-256 use key lengths of 128 bits, 192 bits, and 256 bits, respectively. AES provides a good balance between security and efficiency, making it suitable for a wide range of applications, including secure communications and data storage. AES encryption and decryption operations can be efficiently implemented in hardware and software.

4.2 Modules and their descriptions

Our Approach on face recognition consists of three stages, that is given an image, we perform face localization, face embeddings generation using Google's FaceNet pre-trained model, and classification using the linearSVC classifier.

A. Data Collection

In a case study on 100 different people, a total of 1424 images from the volunteers. Captured using a 12MP dual-lens camera of the tecno camon 12 air smartphone, the images appeared in different orientations and lightings. The number 1424 is that which was obtained after performing data cleaning.

B. Hardware Setup

The training was done on a personal computer. A HP ProBook 450 G3, with

8192MB RAM, Intel(R) Core(TM)i5-6200U CPU @ 2.30GHz(4 CPUs) 2.4GHz processor, and AMD Radeon (TM) R7 M340 graphics card with 2GB dedicated VRAM.

For the case of face detection, two of the most commonly used ways of face detection were compared;

1) The Viola-Jones Haarcascade classifier:

It is based on the Viola-Jones Object Detection framework. This frame work has a quiet high (true positive rate) detection rate and a very low false positive rate making the algorithm a robust one that also processes the images quickly. The main objective is face detection not recognition: it distinguishes faces from non-faces which is the first (preprocessing) step recognition. The framework follows four main steps: Haar Feature Selection, Creating an Integral Image, Adaboost Training and the Cascading Classifiers.

2) The Multi-Task Cascaded Convolutional Neural Network:

that is based on Neural NetworkBased Face Detection. A Neural Network inspired by human brain composed of simple artificial neurons also known a perceptrons are connected to each other in multiple layer. The MTCNN is comprised of three layers. Layer 1: The Proposal Network (P-Net): This is a fully convolutional network which is used to obtain the candidate windows and their bounding box regression vectors as shown in figure 1. Fig. 1. The Proposal Network Layer Layer 2: The Refine Network (R-Net): The output of the P-Net is input into the R-Net which is a Convolution Neural Network and hence denser than the P-Net layer. The R-Net further reduces the number of candidates (which further rejects a large number of false candidates), performs calibration with bounding box regression and employs Non-Maximum Suppression (NMS) to merge overlapping candidates. Fig. 2. The Refine Network Layer [20] Layer 3: The Output Network (O-Net): Similar to the R-Net the O-Net aims to describe the face in detail outputting five facial landmarks' positions for the main face features like the eyes, nose and mouth. Face detection and localization was performed on the faces using the MTCNN algorithm. The localized face is resized into a 160×160 RGB image. A face dataset is created.

4.3 Flow of the System

Capture Image: The system starts by capturing an image of a person's face through a webcam or an image upload.

Facial Detection: A CNN-based facial detection model is used to identify and locate the face within the captured image.

Facial Recognition: Another CNN-based model is employed for facial recognition. It verifies the detected face against a database of known faces.

AES Encryption: If the facial recognition is successful and the user is authenticated, the system encrypts the facial data using the AES algorithm.

Store Encrypted Data: The encrypted facial data is stored securely, possibly in a database.

Web-Based Access: Users can access the system via a web interface, where they can input their credentials and request access to the encrypted facial data.

Web Interface: The web interface allows users to access and interact with the system.

Display Encrypted Data: Users can view the encrypted facial data if they have the proper authorization.

Decryption: If authorized, users can request decryption of the facial data.

Display Decrypted Facial Data: The decrypted facial data is displayed to the authorized user.

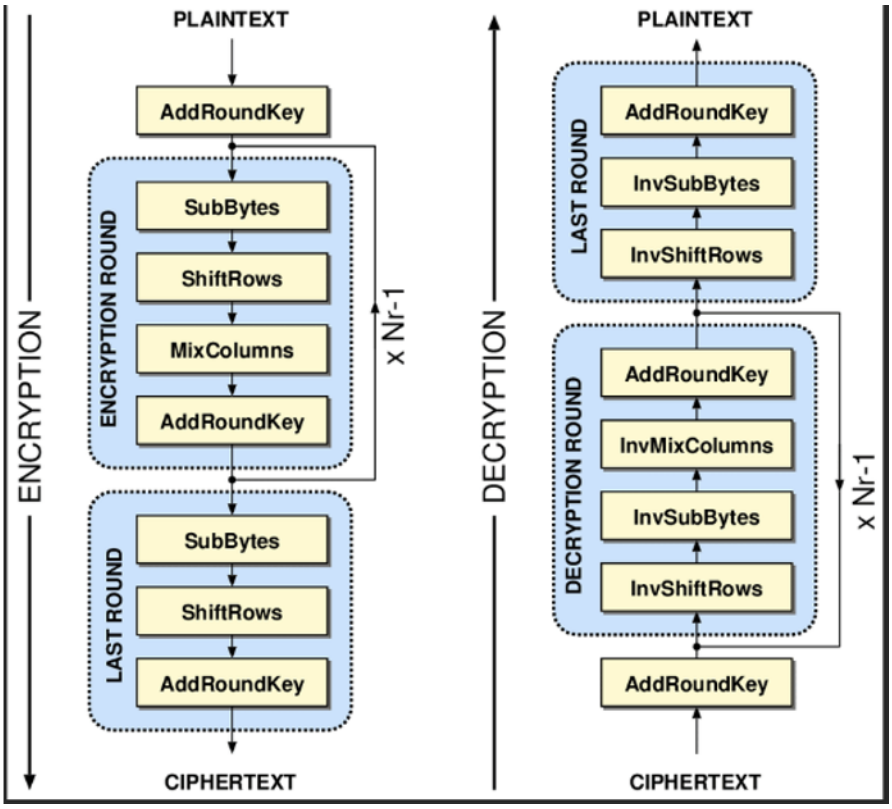


Figure 4.1: AES Work Flow

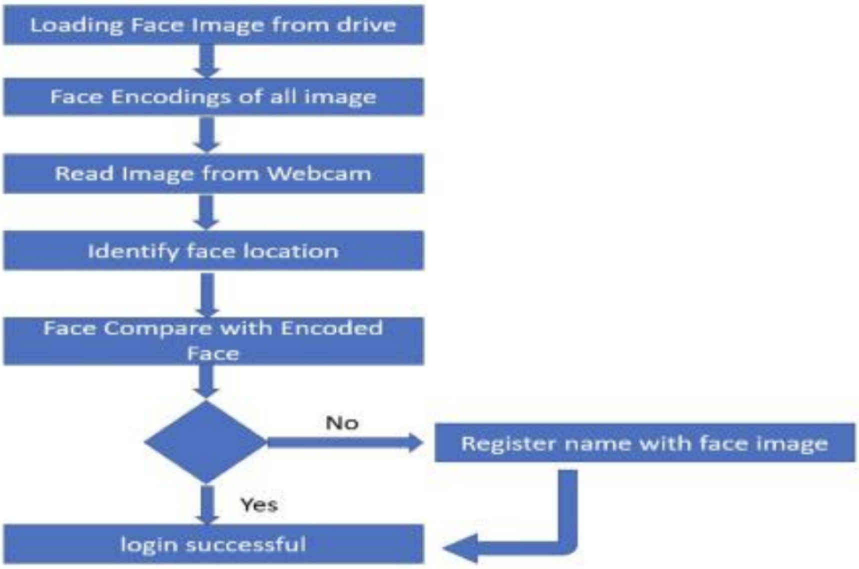

Loading Face Image from drive
Face Encodings of all image
Read Image from Webcam
Identify face location
Face Compare with Encoded Face
No
Register name with face image
Yes
login successful

Figure 4.2: Workflow

Chapter 5

Results and Analysis

5.1 Performance Evaluation

I can provide an overview of how a web-based facial encryption system using the AES algorithm and CNN for facial recognition might work, but I cannot provide actual results or code, as I do not have access to real-time data or the ability to execute code. However, I can describe the process and the expected outcomes

System Overview: A web-based facial encryption system using AES and CNN typically involves several components, including a web interface, image capture, facial detection, facial recognition, AES encryption, and database storage. The system allows users to capture and encrypt facial data and later decrypt it based on their authentication.

Expected Results:

Facial Detection: The CNN-based facial detection model should accurately locate and identify faces within the captured images. The expected result is bounding boxes or regions around detected faces.

Facial Recognition: The CNN-based facial recognition model should be able to recognize individuals' faces and authenticate users. The expected result is a match or a confidence score indicating the likelihood of a match.

AES Encryption: The facial data should be successfully encrypted using the AES algorithm. The expected result is encrypted data that is secure and protected from unauthorized access.

Database Storage: The encrypted facial data should be stored securely in a database. The expected result is a well-structured and protected database containing the encrypted facial data.

Web Interface: The web interface should allow users to input their credentials, request access to encrypted facial data, and interact with the system. The expected result is a user-friendly interface for easy navigation.

Decryption: When authorized, users should be able to decrypt the facial data. The expected result is the original facial data in its recognizable form.

Authentication: Users should be authenticated based on their facial features, and the system should grant or deny access accordingly.

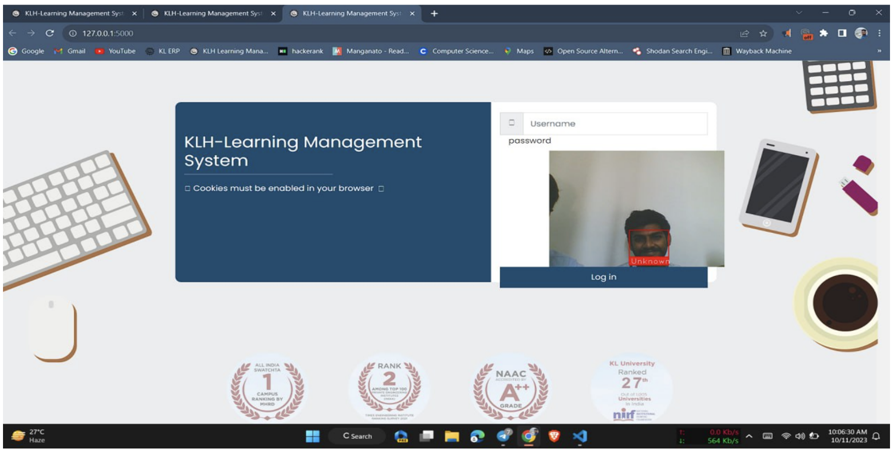
Challenges and Considerations:

Accuracy: The accuracy of facial recognition using CNNs depends on the quality of training data and the model's architecture. High accuracy is essential to ensure correct user authentication.

Security: AES encryption is a strong security measure, but the system must also address potential vulnerabilities such as key management and secure data transmission.

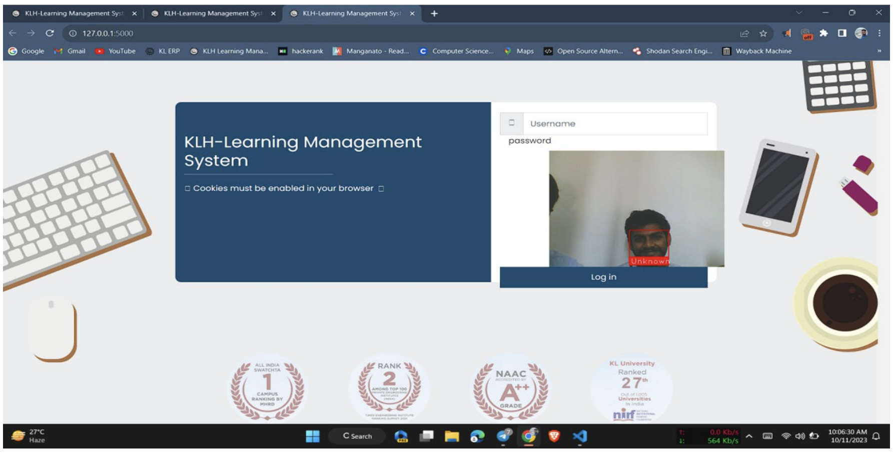
Privacy: Consider privacy concerns related to storing and processing facial data. Ensure compliance with data protection regulations.

User Experience: A user-friendly web interface is crucial to ensure a positive user experience. To obtain actual results, you would need to implement the system, train the CNN models on relevant facial datasets, and perform tests with real users. Monitoring and evaluating the system's performance and security are critical for assessing its effectiveness in a real-world context.


KLH-Learning Management System

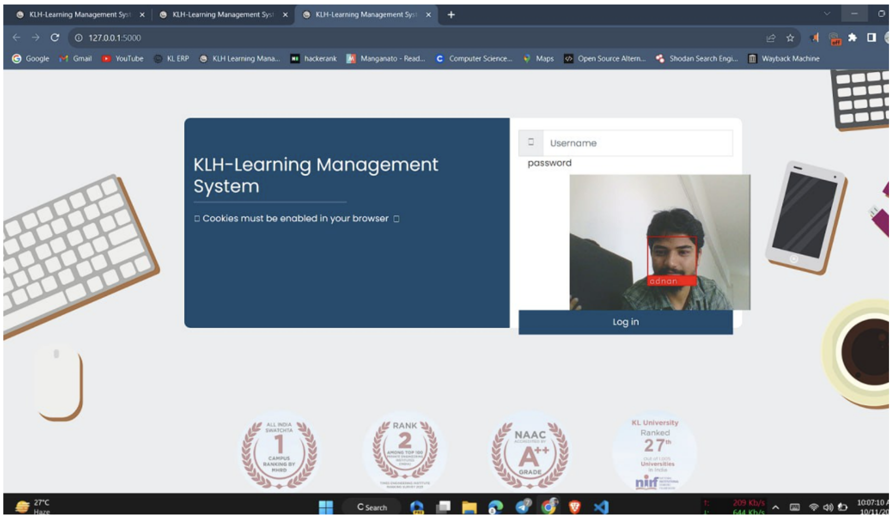
□ Cookies must be enabled in your browser □

Figure 5.1: Image Example1


KLH-Learning Management System

□ Cookies must be enabled in your browser □

Figure 5.2: Image Example2


KLH-Learning Management System

□ Cookies must be enabled in your browser □

Figure 5.3: Image Example3

Chapter 6

Conclusion and Recommendations

6.1 Summary of the Project

In conclusion, web-based facial recognition systems are a rapidly growing technology that offers many benefits for various sectors, including security, identification, and authentication. These systems use advanced biometric algorithms to analyze unique facial features and create a biometric template that can be used for identification purposes. However, the use of facial recognition technology also raises concerns around privacy and surveillance.

Therefore, it is essential to carefully consider ethical and legal issues when developing and deploying web-based facial recognition systems. The implementation of appropriate safeguards is crucial to protect the privacy and rights of individuals. The responsible development and use of this technology can help to enhance security and convenience, but it must be done in a way that does not violate individual rights and freedoms.

As this technology continues to advance, it is important for businesses, organizations, and policymakers to keep up with the latest developments and ensure that facial recognition technology is used in a way that is transparent, fair, and respectful of individuals' privacy and rights. By doing so, we can continue to benefit from the convenience and security provided by this technology while avoiding its potential pitfalls.

Bibliography

[1] Abbadi, I.M., Alawneh, M., 2012. A framework for establishing trust in the Cloud. Comput. Electr. Eng. 38 (5), 1073–1087.

[2] Abbadi, I.M., Martin, A., 2011. Trust in the Cloud. Inf. Secur. Techn. Rep. 16, 108–114.

[3] Adjei, J.K., Blackman, C., Blackman, C., 2015. Explaining the role of trust in cloud computing services. Info 17.

[4] Afroz, S., Navimipour, N.J., 2017. Memory designing using quantum dot cellular automata: systematic literature review, classification, and currenttrends. J. Circuits Syst. Comput. 26 (12), 1730004 (2017) [34 pages].

[5] Alhanahnah, M., Bertok, P., Tari, Z., 2017. Trusting cloud service providers: trust phases and a taxonomy of trust factors. IEEE Cloud Comput. 4,44–54. 8. REFERENCES

[6] rancis Galton, “Personal identification and description,” In Nature. University of Zurich, Zurich, 2017

[7] . Zaho, “Robust image based 3D face recognition,” Ph.D. Thesis, Maryland University, 2019.

[8] . Chellappa, C.L. Wilson and C. Sirohey, ”Humain and machine recognition of faces: A survey,” Proc. IEEE, vol. 83, no. 5, pp. 705-740, may 2010.

[9] . Fromherz, P. Stucki, M. Bichsel, “A survey of face recognition,” MML Technical Report, No 97.01, Dept. of Computer Science, University of Zurich, Zurich, 2007.

[10] . Riklin-Raviv and A. Shashua, “The Quotient image: Class based recognition and synthesis under varying illumination conditions,” In CVPR, P. II: pp. 566-571,2019.

[11] .j. Edwards, T.f. Cootes and C.J. Taylor, “Face recognition using active appearance models,” In ECCV, 1998.

[12] . Sim, R. Sukthankar, M. Mullin and S. Baluja, “Memory- based face recognition for vistor identification,” In AFGR, 2010.

[13] . Sim and T. Kanade, “Combing models and exemplars for face recognition: An illuminating example,” In Proceeding Workshop on Models Versus Exemplars in Computer Vision, CUPR 2001.

Appendices

Appendix A

Source code

A project description is a high-level overview of why you're doing a project. The document explains a project's objectives and its essential qualities. Think of it as the elevator pitch that focuses on what and why without delving into how.

```
from flask import Flask, render template, Response

import cv2

import face_recognition

Import numpy as np

app Flask (_name__)

Load a sample picture and learn how to recognize it.

camera cv2.VideoCapture(0)

varun_image face_recognition.load_image_file(”photos/varun.jpg”)
    varun_face_encoding face_recognition. face_encodings (varun_image)
    [0]

saiteja image face_recognition.load_image_file(”photos/saiteja.jpg”)
    saiteja_face_encoding face_recognition. face_encodings (
    saiteja_image)[0]

balaji_sir_image = face_recognition.load_image_file(”photos/
    balaji_sir.jpg”) balaji_sir_encoding face_recognition.
    face_encodings (balaji_sir_image)[0]

Create arrays of known face encodings and their names

known_face_encodings [

varun_face_encoding,

saiteja_face_encoding,

balaji_sir_encoding,

known_face_names = [

”varun”,

”saiteja”,

”balaji sir”

Initialize some variables

face_locations = []

face_encodings = []

face_names []

process_this_frame = True

def gen_frames():

while True:
uccess, frame = camera.read() # read the camera Frame if not success:

break

else:

#Resize frame of video to 1/4 size for faster face recognition

processing

small_frame = cv2.resize(frame, (0, 0), fx-0.25, fy-8.25)

# Convert the image from BGR color (which OpenCV uses) to RGB color
    face recognition uses)

rgb_small_frame = small_frame[:, :, ::-1]

Only process every other frame of video to save time

#Find all the faces and face encodings in the current frame of video
    face_locations face_recognition. face_locations (rgb_small_frame)

face_encodings face_recognition. face_encodings (rgb_small_frame,
    face_locations)

face_names = [] for face_encoding in face_encodings:

# See if the face is a match for the known face(s) matches
    face_recognition.compare_faces (known_face_encodings,

face_encoding)

name ”Unknown”

# Or instead, use the known face with the smallest distance to

the new face
```

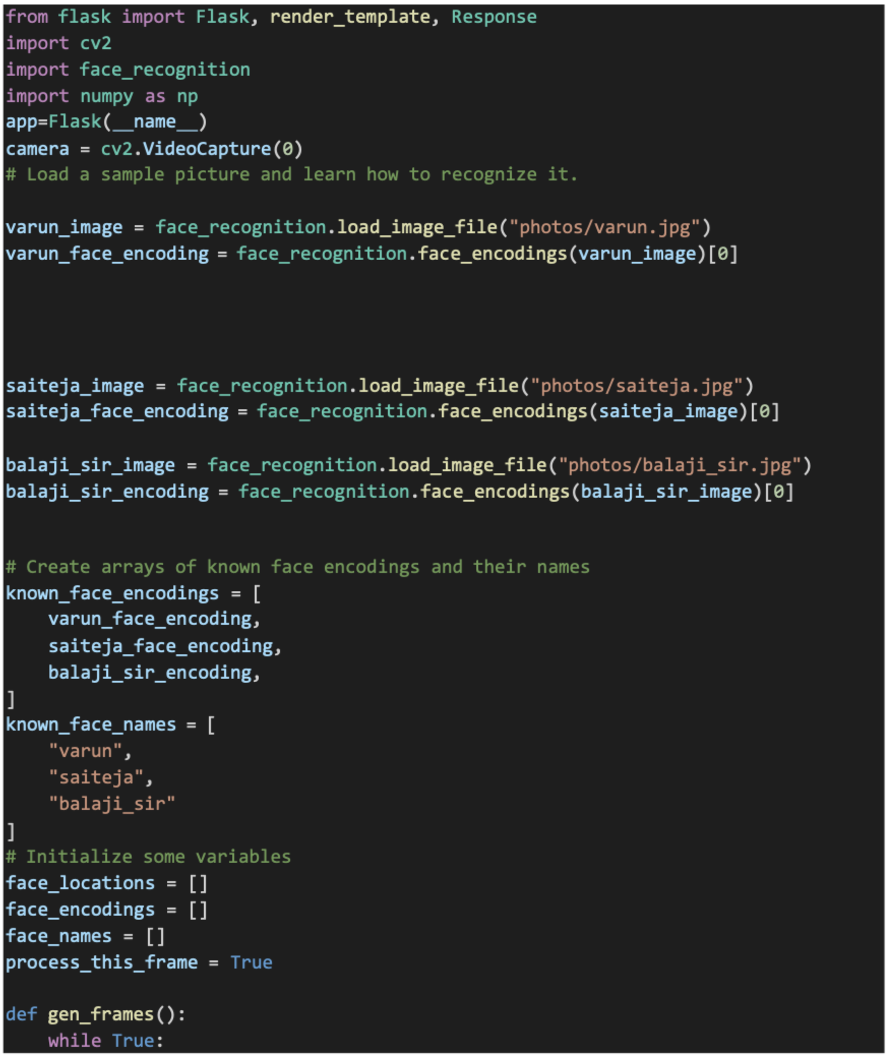
```

face_distances =

. face_distance (known_face_encodings, face_encoding)

Face_recognition

(which

best_match_index = np.argmin(face_distances)

if matches[best_match_index]:

name= known_face_names[best_match_index]

face_names.append(name)

# Display the results

for (top, right, bottom, left), name in zip(face_locations,

face_names):

# Scale back up face locations since the frame we detected in was

scaled to 1/4 size

top*= 4

right *=4

bottom *= 4

left *= 4

# Draw a box around the face cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255),132)

#Draw a label with a name below the face

cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0,

font cv2.FONT HERSHEY DUPLEX

cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.8, (255,

ret, buffer = cv2.imencode('.jpg', frame)

frame buffer.tobytes()

yield (b'--frame\r\n'

b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')

@app.route('/')

def index():

```

```
return render_template('index.html')

@app.route('/video_feed')

def video feed();

return Response(gen_frames(), mimetype='multipart/x-mixed-replace;

boundary-frame')

Af

name

OCT

main

Banjara Hills Rdno 12 Model SAITEJA, [07-Nov-2023 at 8:58:56 PM ]:
...rom flask import Flask, render_template, Response
import cv2
import face_recognition
import numpy as np
app=Flask( name )
camera = cv2.VideoCapture(0)
# Load a sample picture and learn how to recognize it.
varun_image = face_recognition.load_image_file("photos/varun.jpg")
varun_face_encoding = face_recognition.face_encodings(varun_image)[0]
saiteja_image = face_recognition.load_image_file("photos/saiteja.jpg"
    )
saiteja_face_encoding = face_recognition.face_encodings(saiteja_image
    )[0]
balaji_sir_image = face_recognition.load_image_file("photos/
    balaji_sir.jpg")
balaji_sir_encoding = face_recognition.face_encodings(
    balaji_sir_image)[0]
# Create arrays of known face encodings and their names
known_face_encodings = [
varun_face_encoding,
saiteja_face_encoding,
balaji_sir_encoding,
]
known_face_names = [
"varun",
"saiteja",
"balaji_sir"
]
# Initialize some variables
face_locations = []
face_encodings = []
face_names = []
process_this_frame = True
def gen_frames():
while True:
5. IMPLEMENTATION
13
success, frame = camera.read() # read the camera frame
if not success:
```

```
break
else:
# Resize frame of video to 1/4 size for faster face recognition
processing
small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)
# Convert the image from BGR color (which OpenCV uses) to RGB color
(which face_recognition uses)
rgb_small_frame = small_frame[:, :, ::-1]
# Only process every other frame of video to save time
# Find all the faces and face encodings in the current frame of video
face_locations = face_recognition.face_locations(rgb_small_frame)
face_encodings = face_recognition.face_encodings(rgb_small_frame,
face_locations)
face_names = []
for face_encoding in face_encodings:
# See if the face is a match for the known face(s)
matches = face_recognition.compare_faces(known_face_encodings,
face_encoding)
the new face
name = "Unknown"
# Or instead, use the known face with the smallest distance to
face_distances =
face_recognition.face_distance(known_face_encodings, face_encoding)
best_match_index = np.argmin(face_distances)
if matches[best_match_index]:
name = known_face_names[best_match_index]
face_names.append(name)
# Display the results
for (top, right, bottom, left), name in zip(face_locations,
face_names):
# Scale back up face locations since the frame we detected in was
scaled to 1/4 size
top *= 4
right *= 4
bottom *= 4
left *= 4
# Draw a box around the face
cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255),
2)

cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0,
255), cv2.FILLED)
font = cv2.FONT_HERSHEY_DUPLEX
cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255,
255, 255), 1)
ret, buffer = cv2.imencode('.jpg', frame)
frame = buffer.tobytes()
yield (b'--frame\r\n'
b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')
@app.route('/')
def index():
return render_template('index.html')
@app.route('/video_feed')
def video_feed():
return Response(gen_frames(), mimetype='multipart/x-mixed-replace;
boundary=frame')
if name ==' main__':
app.run(debug=True
```

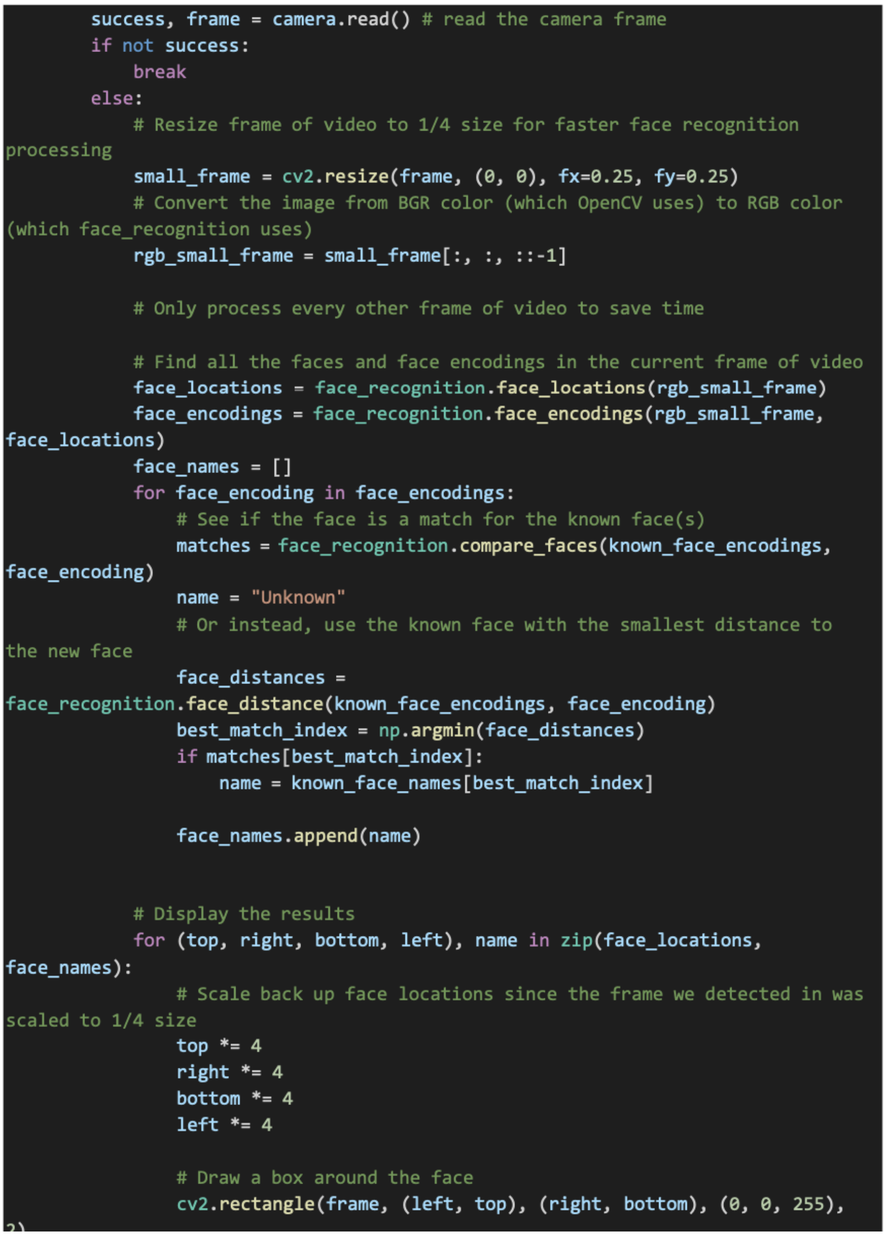
Appendix B

Screen shots



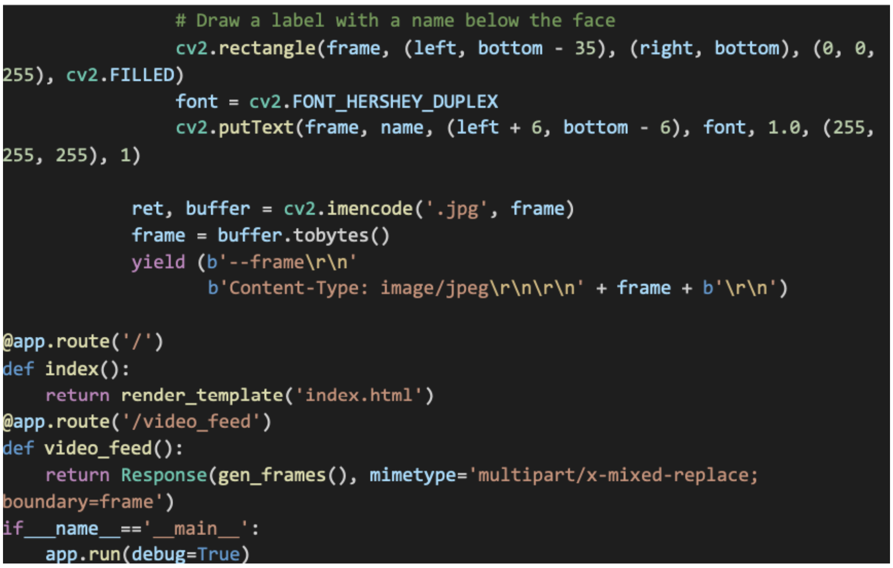
```
from flask import Flask, render_template, Response
import cv2
import face_recognition
import numpy as np
app=Flask(__name__)
camera = cv2.VideoCapture(0)
# Load a sample picture and learn how to recognize it.

varun_image = face_recognition.load_image_file("photos/varun.jpg")
varun_face_encoding = face_recognition.face_encodings(varun_image)[0]




saiteja_image = face_recognition.load_image_file("photos/saiteja.jpg")
saiteja_face_encoding = face_recognition.face_encodings(saiteja_image)[0]

balaji_sir_image = face_recognition.load_image_file("photos/balaji_sir.jpg")
balaji_sir_encoding = face_recognition.face_encodings(balaji_sir_image)[0]


# Create arrays of known face encodings and their names
known_face_encodings = [
    varun_face_encoding,
    saiteja_face_encoding,
    balaji_sir_encoding,
]
known_face_names = [
    "varun",
    "saiteja",
    "balaji_sir"
]
# Initialize some variables
face_locations = []
face_encodings = []
face_names = []
process_this_frame = True

def gen_frames():
    while True:
```

Figure 6.1: Source Code1



```
        success, frame = camera.read() # read the camera frame
        if not success:
            break
        else:
            # Resize frame of video to 1/4 size for faster face recognition
processing
            small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)
            # Convert the image from BGR color (which OpenCV uses) to RGB color
(which face_recognition uses)
            rgb_small_frame = small_frame[:, :, ::-1]

            # Only process every other frame of video to save time

            # Find all the faces and face encodings in the current frame of video
            face_locations = face_recognition.face_locations(rgb_small_frame)
            face_encodings = face_recognition.face_encodings(rgb_small_frame,
face_locations)
            face_names = []
            for face_encoding in face_encodings:
                # See if the face is a match for the known face(s)
                matches = face_recognition.compare_faces(known_face_encodings,
face_encoding)
                name = "Unknown"
                # Or instead, use the known face with the smallest distance to
the new face
                face_distances =
face_recognition.face_distance(known_face_encodings, face_encoding)
                best_match_index = np.argmin(face_distances)
                if matches[best_match_index]:
                    name = known_face_names[best_match_index]

                face_names.append(name)


            # Display the results
            for (top, right, bottom, left), name in zip(face_locations,
face_names):
                # Scale back up face locations since the frame we detected in was
scaled to 1/4 size
                top *= 4
                right *= 4
                bottom *= 4
                left *= 4

                # Draw a box around the face
                cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255),
2)
```

Figure 6.2: Source Code2



```
                # Draw a label with a name below the face
                cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0,
255), cv2.FILLED)
                font = cv2.FONT_HERSHEY_DUPLEX
                cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255,
255, 255), 1)

            ret, buffer = cv2.imencode('.jpg', frame)
            frame = buffer.tobytes()
            yield (b'--frame\r\n'
                   b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')

@app.route('/')
def index():
    return render_template('index.html')
@app.route('/video_feed')
def video_feed():
    return Response(gen_frames(), mimetype='multipart/x-mixed-replace;
boundary=frame')
if__name__=='__main__':
    app.run(debug=True)
```

Figure 6.3: Source Code3

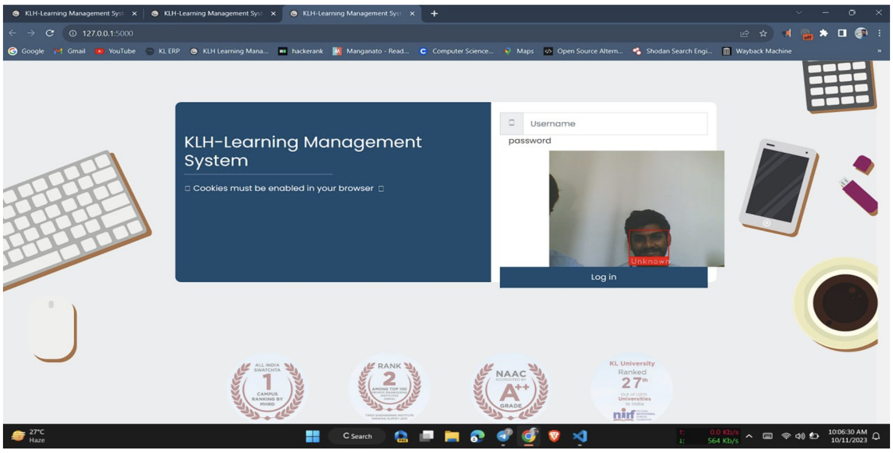

KLH-Learning Management System
127.0.0.1:5000
KLH-Learning Management System
□ Cookies must be enabled in your browser □
Username
password
Unknown
Log in

Figure 6.4: Result Image1

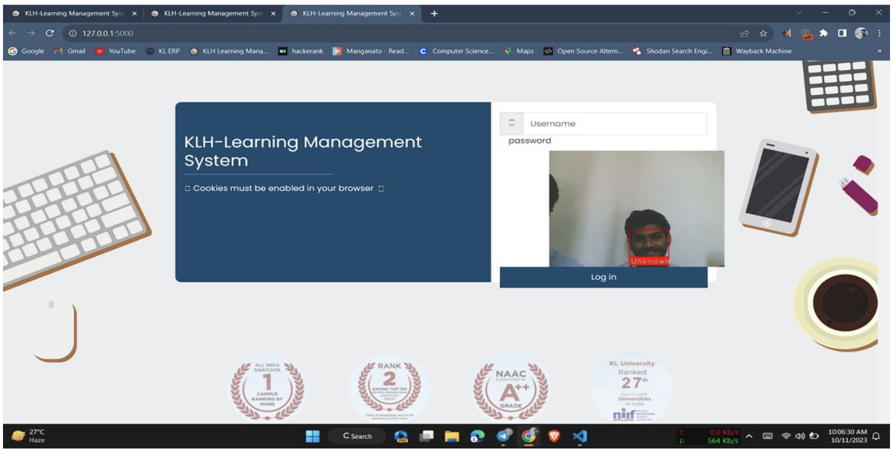
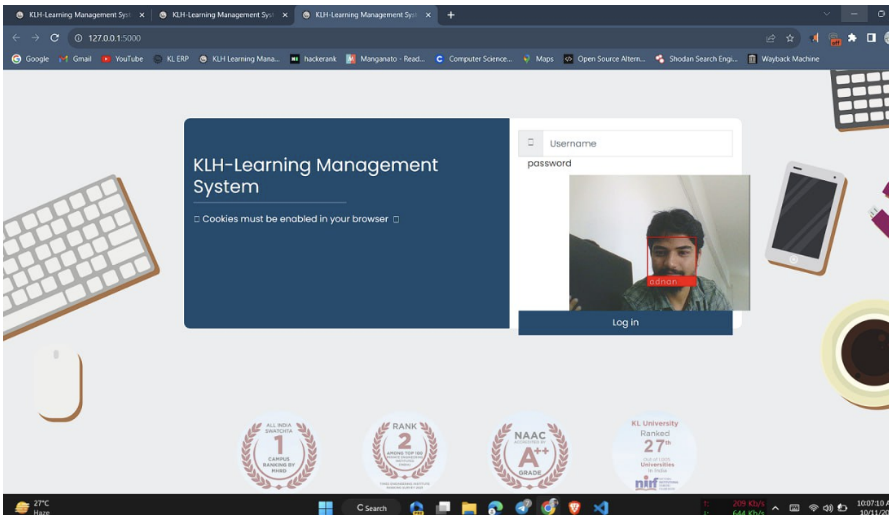

KLH-Learning Management System
127.0.0.1:5000
KLH-Learning Management System
□ Cookies must be enabled in your browser □
Username
password
Unknown
Log in

Figure 6.5: Result Image2



KLH-Learning Management System

□ Cookies must be enabled in your browser □

Figure 6.6: Result Image3