# Integrating Quantum Image Processing with Computer Vision: A Streamlit Application for Real-Time Face Detection and Recognition

S. Rajarajeswari
*Computer Science and Engineering*
*Ramaiah Institute of Technology*

Hassan Srinivas Ranjitha
*Computer Science and Engineering*
*Ramaiah Institute of Technology*

***Abstract* -** This paper introduces a Streamlit-based web application that combines quantum processing concepts with traditional computer vision for face detection and recognition. Key libraries, including NumPy, OpenCV, and PIL, handle image processing, while Streamlit WebRTC enables real-time video streaming. The interface allows users to upload images, choose detection modes, and configure face detection settings. Using the Haar Cascade Classifier, faces in photos and video streams are detected and highlighted with bounding boxes. A built-in camera feature enables capturing images to build face datasets and train recognition algorithms. The application supports real-time attendance tracking, logging recognized faces with timestamps. Additionally, advanced quantum techniques like quantum edge detection are included for experimental image processing, expanding the application's versatility across various computer vision and machine learning tasks.

*Keywords –-Face Recognition, Quantum edge detection, OpenCV, LBPH, Streamlit.*

## 1. INTRODUCTION

Recent advancements in integrating quantum computing with computer vision focus on improving face detection and recognition accuracy and speed, particularly for applications requiring real-time processing. Quantum-enhanced facial recognition leverages the unique properties of quantum mechanics, such as superposition and entanglement, which allow faster and more complex data processing than classical systems. This results in improved accuracy and efficiency in face recognition tasks, which are essential for applications in security, surveillance, and data security.

One prominent study on quantum-enhanced face recognition used quantum convolutional neural networks (QCNNs) to improve facial recognition algorithms. These QCNNs enable faster and more efficient processing by utilizing quantum gates, allowing the system to better handle complex patterns in facial features with a high degree of accuracy. Such advancements are beneficial for real-time surveillance systems, as they allow faster recognition and response times compared to traditional methods

Furthermore, quantum computing's parallel processing capabilities reduce the time complexity associated with facial recognition models, making it feasible to analyze vast amounts of data simultaneously. Quantum encryption also adds an additional layer of security for biometric data, which is crucial for privacy in face recognition applications. Quantum encryption methods create highly secure, random keys, making unauthorized data access more challenging. These features make quantum-enhanced face recognition particularly appealing in industries that demand high security, such as law enforcement, healthcare, and finance

The goal of developing a quantum edge processing-based facial recognition attendance framework is to overcome the drawbacks of conventional techniques and existing biometric systems. The goal of this framework is to give attendance monitoring a more exact, safe, and efficient method—especially in big school contexts. Such a solution offers significant advantages over traditional methods by lowering the possibility of fraud and improving overall classroom management.

With the help of an intuitive Streamlit web application, this script is a potent tool for sophisticated face detection and identification. It makes use of traditional computer vision methods as well as investigates quantum computing capabilities, albeit the latter is not actively employed in the code that is displayed. Users may upload photos, record webcam footage in real time, and do real-time face identification using this program. It can identify faces in video streams, recognize faces in photos, and collect and label face data for model training, among other functions. An LBPH method facilitates face recognition by using grayscale photos to train the model and increase accuracy.

Real-time attendance tracking is available to users, who may log facial recognition data and manage it with a CSV file. For those interested in cutting edge methods, the script also incorporates experimental quantum image processing features. The Streamlit interface may be customized to seem neater and more polished. All in all, the script offers a flexible tool for both real-time monitoring and model creation by fusing strong face detection skills with cutting-edge quantum processing.

All things considered, the tracking and management of attendance is expected to undergo a complete transformation with the use of face recognition technology and quantum edge processing in attendance systems. This framework is a significant development in organizational and educational management methods because it combines the benefits of modern image processing techniques with the computational power of quantum computing to improve efficiency, accuracy, and security.

## 2. RELATED RESEARCH

The presented literature presents a novel method of tracking attendance through the integration of Quantum Edge Detection(QED) into facial recognition systems. It recognizes the shift in attendance tracking techniques toward sophisticated biometric solutions, emphasizing the growing importance of face recognition technology. By exploiting quantum computing's potential to transform image processing through parallelism and entanglement, the integration of quantum image processing marks a paradigm leap.

The review of the literature explores the state of the art in face recognition attendance framework research using Quantum Image Processing. It methodically examines the approaches used, theoretical underpinnings, and experimental results in this developing field. The review identifies important fields of study, including the use of quantum features and the integration of quantum principles into image processing systems.

A variety of approaches and advancements in the field of facial recognition-based attendance systems are highlighted in the collection of relevant work that is being presented. Thai-Viet Dang et al.'s [1] enhanced facial recognition model is the main tool used to improve the smart attendance system, and the results are impressive in terms of processing speed and accuracy. An Android-based attendance system is introduced by Dwi Sunaryono et al. [2], who highlight the system's improved classification accuracy and shorter training time when compared to other algorithms. In order to overcome issues with imperfect facial data, Baraa Adil Mahmood et al. [3] introduce a novel FW-MPM-LSTM technique that works better than current techniques. In order to automate face recognition in educational contexts, Muhammad Zeeshan Khan et al. [4] present a deep unified model that makes use of convolution neural networks and edge computing. In a Face Recognition Smart Attendance System, Khawla Alhanaeea et al. [5] investigate deep transfer learning and show good validation accuracies using several pre-trained CNN models. In their thorough evaluation of the literature, Budiman et al. [6] recommend CNN algorithms for class attendance because of their accuracy and stability. In order to overcome the difficulties posed by variations in facial expressions, Yogesh Kumar et al. [7] present a hybrid deep learning model for multi-pose facial expression identification. Ibrahim Al-Amoudi et al.'s automatic attendance system [8] emphasizes the necessity for advancements in non-frontal face recognition by utilizing the MTCNN and FaceNet algorithms. In their investigations of quantum face recognition protocols and neural networks, Vahid Salari et al. [9] and Yan Xu et al. [10] go deeper into quantum-based face recognition techniques. In order to address current issues, Roshan M. Thomas et al. [11] concentrate on real-time face mask detection and recognition using CNNs. A thorough analysis of digital image processing methods for facial identification is done by Yogalakshmi.S et al. [12], who stress the value of an integrated strategy.Liu et al. [13] developed a quantum signal processing framework for medical image edge detection. Tseng and Hwang introduced a quantum edge detection algorithm comparable to the Sobel method. Songlin et al. proposed a maximum quantum entropy criterion for image thresholding, demonstrating its superiority over Shannon entropy.

## 3. PROPOSED METHODOLOGY

Steps Followed by a Streamlit Application for Real-Time Face Detection and Recognition

1. Setup and Imports

- Import necessary libraries: streamlit, opencv-python, numpy, and any machine learning or deep learning libraries (like dlib, face_recognition, or OpenCV for detection).
- Load pre-trained models, such as Haar cascades, deep learning models, or LBPH face recognizers.

2. User Interface (UI) Design Using Streamlit

- Use st.title() and st.sidebar() to create a user-friendly interface.
- Provide options for the user to select between different modes, such as:
  - Real-Time Detection: Using the webcam.
  - Image Upload: Upload an image for detection and recognition.
- Create buttons using st.button() for starting or stopping detection.

3. Initializing Video Capture

- Use cv2.VideoCapture() to capture video from the webcam.
- Define a function to continuously read frames from the webcam using OpenCV:

```
def capture_video():
cap = cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()
    if not ret:
        break
    # Further processing and display
cap.release()
cv2.destroyAllWindows()
```

4. Face Detection

- Implement face detection using methods like Haar cascades or deep learning models:

```
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray_frame, 1.3, 5)
```

- Draw rectangles around detected faces using cv2.rectangle().

5. Face Recognition

- Extract facial features and use a trained model for recognition:
    - For LBPH (Local Binary Patterns Histograms):

```
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('trained_model.yml')  # Load the trained model
```

- Compare detected faces with known faces to recognize individuals.
- Display the name or label of the recognized face on the video frame.

6. Displaying Video Stream with Streamlit

- Use st.image() or st.video() to display the processed frames in the Streamlit app.
- Convert OpenCV's BGR image format to RGB for Streamlit:

```
st.image(cv2.cvtColor(frame, cv2.COLOR_BGR2RGB))
```

7. User Interactions and Control

- Allow the user to start or stop the video feed using Streamlit buttons.
- Use options to save recognized faces or log recognition events if needed.

8. Additional Features

- Option to upload and register new faces into the recognition database.
- Display statistics or recognition confidence scores in real time.
- Provide options to switch between different detection algorithms or change detection parameters.

9. Cleaning Up Resources

- Ensure proper release of the webcam and destruction of OpenCV windows when the application stops:

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

ALGORITHMS USED :

### 1. *Haar Cascade Classifier*

The Haar cascade classifier, used extensively in computer vision for tasks like face detection, leverages mathematical concepts around feature selection, weak classifiers, and ensemble learning. The primary mathematics involved:

#### a. Haar-like Features

Haar-like features are simple, rectangular patterns that detect contrasts in regions of an image. They work by comparing adjacent rectangular regions, typically using the difference in the sum of pixel intensities in these regions. A few common Haar-like features are:

Two-rectangle features: The difference between the sum of pixels in two adjacent regions.

Three-rectangle features: The difference between the sum of the pixels in the outer two regions and the center region.

Four-rectangle features: The difference between diagonal pairs

The mathematical expression for a Haar-like feature fi in an image I at a certain position (x,y) with a certain width and height is:

$$fi = \sum_{(x,y)\in R_1} I(x,y) - \sum_{(x,y)\in R_2} I(x,y)$$

where R1 and R2 represent the rectangular regions.

#### b. Integral Image (Summed Area Table)

To compute these features efficiently, an integral image S is used. The integral image at any point (x,y) contains the sum of the pixels above and to the left of (x,y) in the original image:

$$S(x,y) = \sum_{i=0}^{x} \sum_{j=0}^{y} I(i,j)$$

Using the integral image, the sum of pixels in any rectangular region can be computed in constant time.

#### c. Weak Classifiers

Each Haar-like feature acts as a weak classifier, often using a single threshold for decision-making. A weak classifier h based on a feature $f_i$, threshold θ, and polarity p (indicating the direction of inequality) is defined as:

$$h(x) = \begin{cases} 1 & \text{if} p f_i(x) < p\theta \\ 0 & \text{otherwise} \end{cases}$$

where x is the image region. If the classifier's output is 1, it indicates the presence of a face-like region; otherwise, it's considered non-face-like.

#### d. Boosting and AdaBoost Algorithm

To enhance the detection performance, many weak classifiers are combined using the **AdaBoost** algorithm, which assigns weights to each classifier. The classifier weights $\alpha_t$ are computed based on each weak classifier's accuracy, with higher weights given to more accurate classifiers:

$$\alpha_t = \ln\left(\frac{1-e_t}{e_t}\right)$$

where $e_t$ is the weighted error rate of the classifier $h_t$. AdaBoost adjusts the weights of the training samples to focus more on samples misclassified by the previous weak classifiers, improving overall performance.

The strong classifier H(x) is then a weighted sum of weak classifiers:

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

Where the following denotes a positive detection.

$$H(x) \geq \frac{1}{2}\sum_{t=1}^{T} \alpha_t$$

**e. Cascade Structure**

The Haar cascade classifier consists of a series of stages, where each stage contains a strong classifier trained to reject negative regions quickly. This cascade structure is designed to reject non-face regions early in the pipeline to improve efficiency.

In each stage Sj, if the detection score Hj(x) from that stage exceeds a threshold, the region proceeds to the next stage. Otherwise, it is immediately classified as non-face:

$$S_j(x) = \begin{cases} \text{proceed to next stage} & \text{if } H_j(x) \geq \tau_j \\ \text{reject as non-face} & \text{otherwise} \end{cases}$$

This cascading approach minimizes computation by quickly eliminating non-facial regions.

2. ***LBPH (Local Binary Patterns Histograms) Algorithm***

Face recognition systems use the LBPH algorithm to extract characteristics from facial photographs. In order to create a binary pattern depending on whether the nearby pixels are more or less than the intensity of the central pixel, it compares each pixel with its neighbouring pixels. Following that, these patterns are transformed into histograms, which represent the frequency at which each pattern appears in the picture. Through the comparison of these histograms between a recently taken picture and a database of faces that has been saved, LBPH is able to identify faces even in situations with different lighting and emotions on their faces.

a. Local Binary Patterns (LBP) Calculation:

- LBP is a simple and efficient texture operator that labels the pixels of an image by thresholding the neighborhood of each pixel and converting the result into a binary number.
- For a given pixel at $(x_c,y_c)$ with intensity $I(x_c,y_c)$, consider its neighborhood of P equally spaced pixels on a circle of radius R. Let $I(x_p,y_p)$ be the intensity of the $p^{th}$ neighboring pixel.
- The LBP value for the pixel $(x_c,y_c)$ is given by:

$$\mathrm{LBP}_{P,R} = \sum_{p=0}^{P-1} s(I(x_p,y_p) - I(x_c,y_c)) \times 2^p$$

Where

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

b. Histogram Calculation:

- Once the LBP values are calculated for every pixel in the image, a histogram is constructed to represent the frequency of each LBP pattern in the image.
- If there are P neighbors, the resulting histogram will have $2^P$ bins, each bin corresponding to a possible LBP pattern: H(i)=Number of pixels with LBP value i,

  for $i=0,1,\ldots,2^P-1$

c. Concatenating Histograms for Face Recognition:

- The image is divided into several regions, and the LBP histograms are calculated for each region. These histograms are then concatenated into a single, feature vector.
- If the image is divided into m×n regions, the final feature vector F is: $F=[H_1,H_2,\ldots,H_{m\times n}]$
- where $H_k$ represents the histogram of the $k^{th}$ region.

d. Distance Measure for Face Recognition:

- To compare two histograms (e.g., for face recognition), a distance metric such as the Euclidean distance or Chi-square distance is used. The Chi-square distance between two histograms $H_1$ and $H_2$ is given by:

$$2(H1,H2) = \sum_{i=1}^{N} \frac{(H_1(i) - H_2(i))^2}{H_1(i) + H_2(i) + \epsilon}$$

- where N is the number of bins in the histograms, and ϵ is a small constant to avoid division by zero.

3. Quantum Edge Detection Algorithm

The Quantum Edge Detection Algorithm utilizes principles from quantum mechanics and quantum computing to enhance edge detection capabilities. The concepts like quantum image representation, quantum Fourier transform, and edge detection operators are deployed in face edge detection.

### a. Quantum Image Representation

- **Qubit Representation**: Images are encoded into quantum states to leverage the parallel processing power of quantum computing. One common method is the *Quantum Image Representation* using *Quantum Image Processing (QIP)*, which employs qubits to represent pixel intensity and spatial information.
- For an image with dimensions N×N times, a quantum state representing the image can be described as:

$$I = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |i, j\rangle |p_{ij}$$

where $|i,j\rangle$ encodes the pixel position and $|p_{ij}\rangle$ encodes the pixel intensity.

### b. Quantum Fourier Transform (QFT)

- **QFT** is a quantum version of the classical Fourier transform and is used for frequency analysis of the image. The QFT of a quantum state $|x\rangle$ is given by:

$$QFT|x\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i x k/N} |k\rangle$$

- QFT helps in identifying edges by analyzing the frequency components of the image.

### c. Edge Detection Operators

- **Quantum Edge Operators**: Analogous to classical edge detection filters (like the Sobel or Laplacian operators), quantum edge detection algorithms use quantum operators. These operators can identify regions in an image with high-intensity gradients, which correspond to edges.
- The mathematical operation can be represented using quantum gates and circuits that apply transformations based on the intensity differences between neighboring pixels.

### d. Quantum Measurement and Gradient Calculation

- The quantum system is measured to extract information about edges. The outcome of the measurement provides the gradient magnitude at each pixel location.
- **Gradient Magnitude Calculation**: Using quantum circuits, the difference between pixel intensities is calculated. For example, if $p_{ij}$ and $p_{i+1,j}$ are the intensities of neighboring pixels, the gradient G can be expressed as:

$$G = \sqrt{(p_{i+1,j} - p_{ij})^2 + (p_{i,j+1} - p_{ij})^2}$$

- This calculation is efficiently performed using quantum parallelism.

### e. Quantum Superposition and Parallelism

One of the advantages of using quantum algorithms is the ability to process all pixels simultaneously due to quantum superposition. The entire image is processed in parallel, significantly reducing computational time.

### f. Thresholding and Edge Detection

- After calculating the gradients, a threshold is applied to determine the presence of an edge. The thresholding operation can be formulated as:

$$E(i,j) = \begin{cases} 1 & \text{if} G(i,j) \geq \text{Threshold} \\ 0 & \text{otherwise} \end{cases}$$

- Quantum algorithms apply this thresholding process using quantum gates and measurements.

## 4. RESULT AND OBSERVATION

The Face Recognition Attendance System effectively integrates various technologies for seamless functionality. It uses OpenCV’s Haar cascade classifier for face detection and the LBPH algorithm for accurate recognition from webcam footage. The system’s web-based interface, built using a web application framework, allows users to interact via browsers, enabling features like capturing images, uploading photos, and training models. Despite importing a quantum computing library, the system primarily relies on image processing and computer vision techniques. It supports real-time video streaming for dynamic user engagement and offers an intuitive setup with a customizable layout. Users can choose detection modes, upload images, and view real-time detection with highlighted faces, making the experience simple and efficient.

The system includes a component for training a facial recognition model, which involves capturing and processing images to classify faces. Its primary focus is developing an optimized model for real-time face recognition. The attendance feature logs presence and timestamps while removing duplicate entries for accuracy. The system efficiently manages resources by closing windows and releasing the camera when not in use, ensuring smooth operation.

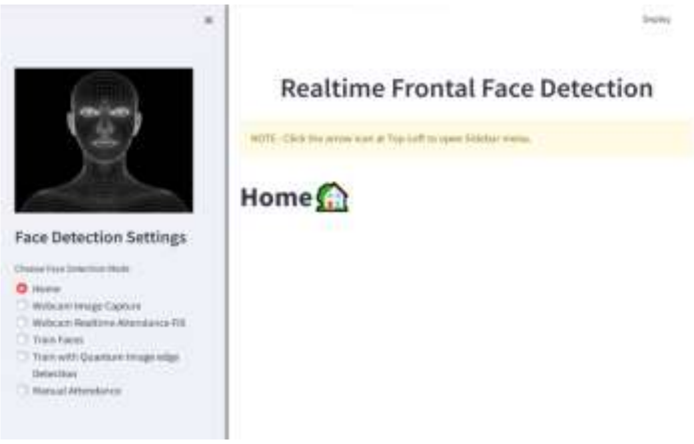

Fig. 1: Home page

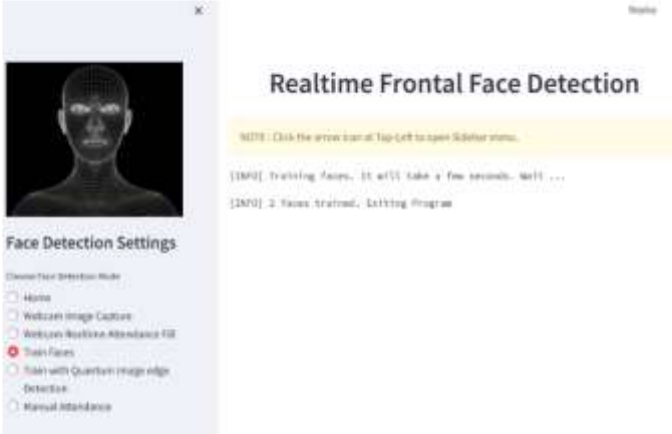

Fig.4: Train Faces

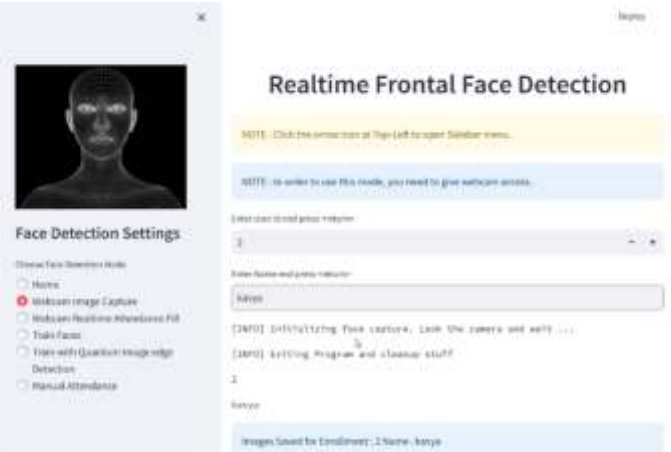

Fig. 2: Filling Student Information

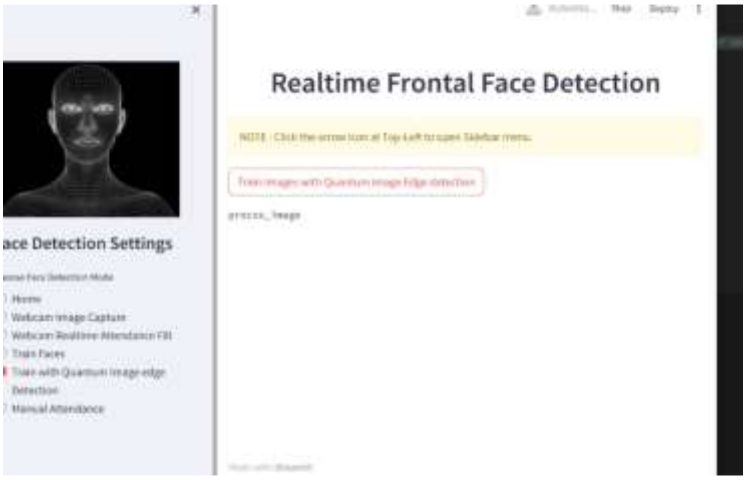

Fig.5: Train Images with QED

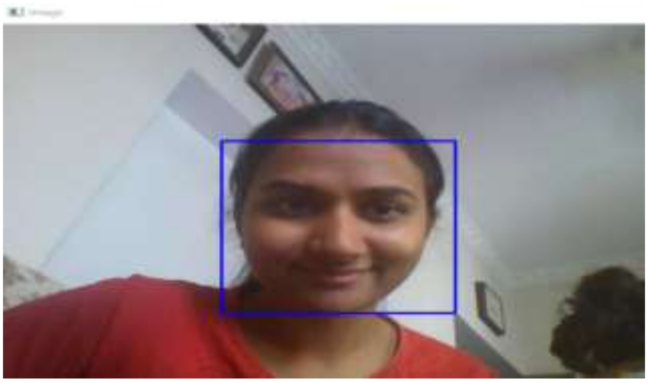
Fig. 3: Training Images

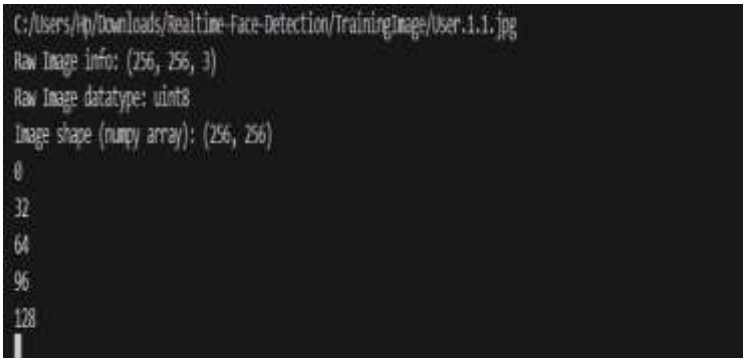

Fig.6: Processing Quantum edge Detection

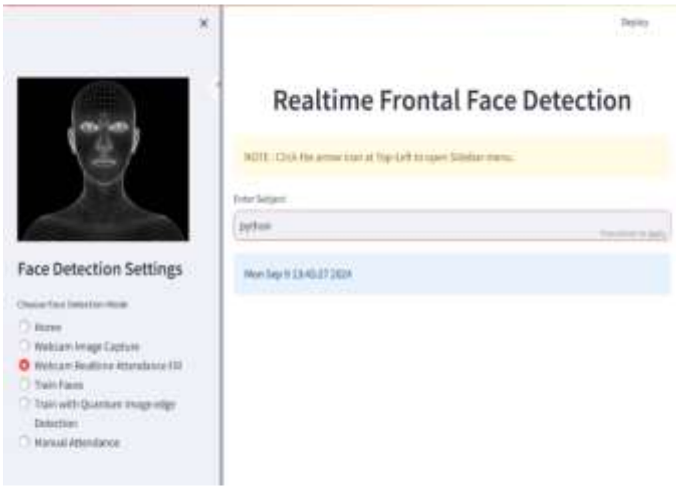


Fig. 7: marking attendance for a particular subject

Fig.8: Marking Attendance of the Students

Fig. 9 : edge detection of the face

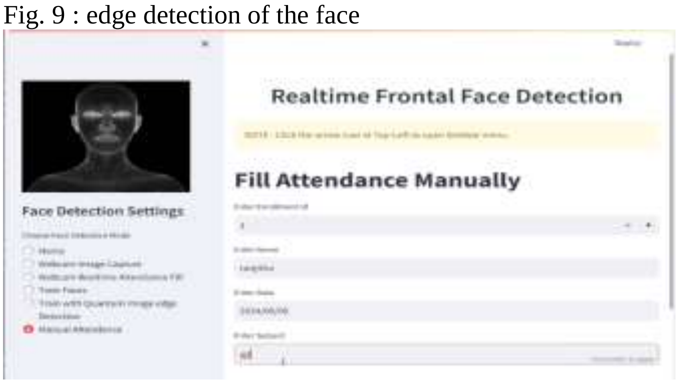


Fig10: manually filling attendance for particular subject

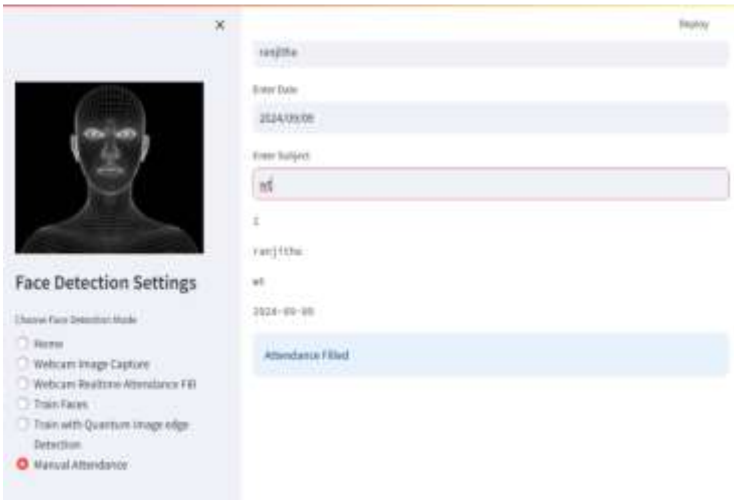


Fig.11: attendance entered for particular subject

## 6. CONCLUSION

The code integrates various libraries and functions to create a complete face detection and identification system. It uses computer vision and image processing libraries, Qiskit for quantum computing tasks, and Streamlit for building a web-based interface. The setup with Streamlit allows for a user-friendly interface, with sidebars for face detection modes and customizable settings. Haar Cascade Classifier and the LBPH algorithm are used for detecting faces in static images, and the app offers options to download or view processed images. Additionally, the system can capture photos from a camera, train a recognition model, and detect faces in real-time video streams, useful for applications like attendance monitoring. Quantum computing enhances edge detection through amplitude encoding, adding innovative quantum elements to image processing. The Streamlit UI is customized to be cleaner, and the code incorporates memory management to optimize performance, blending classical and quantum approaches effectively.

## REFERENCES

[1]Dang, T. V. (2023). Smart attendance system based on improved facial recognition. Journal of Robotics and Control (JRC), 4(1), 46-53.

[2]Sunaryono, D., Siswantoro, J., & Anggoro, R. (2021). An android based course attendance system using face recognition. Journal of King Saud University-Computer and Information Sciences, 33(3), 304-312..

[3]Mahmood, B. A., & Kurnaz, S. (2023). An investigational FW-MPM-LSTM approach for face recognition using defective data. Image and Vision Computing, 132, 104644.

[4]Khan, M. Z., Harous, S., Hassan, S. U., Khan, M. U. G., Iqbal, R., & Mumtaz, S. (2019). Deep unified model for face recognition based on convolution neural network and edge computing. IEEE Access, 7, 72622-72633..

[5]Alhanaee, K., Alhammadi, M., Almenhali, N., & Shatnawi, M. (2021). Face recognition smart attendance system using deep transfer learning. Procedia Computer Science, 192, 4093-4102.

[6]Budiman, A., Yaputera, R. A., Achmad, S., & Kurniawan, A. (2023). Student attendance with face recognition (LBPH or CNN): Systematic literature review. Procedia Computer Science, 216, 31-38.

[7]Kumar, Y., Verma, S. K., & Sharma, S. (2022). Multi-pose facial expression recognition using hybrid deep learning model with improved variant of gravitational search algorithm. Int. Arab J. Inf. Technol., 19(2), 281-287.

[8]Al-Amoudi, I., Samad, R., Abdullah, N. R. H., Mustafa, M., & Pebrianti, D. (2022). Automatic attendance system using face recognition with deep learning algorithm. In Proceedings of the 12th National Technical Seminar on Unmanned System Technology 2020: NUSYS'20 (pp. 573-588). Springer Singapore..

[9]Salari, V., Paneru, D., Saglamyurek, E., Ghadimi, M., Abdar, M., Rezaee, M., ... & Karimi, E. (2023). Quantum face recognition protocol with ghost imaging. Scientific Reports, 13(1), 2401.

[10]Xu, Y., Zhang, X., & Gai, H. (2011). Quantum neural networks for face recognition classifier. Procedia Engineering, 15, 1319-1323.

[11]Thomas, R. M., Sabu, M., Samson, T., Mol, S., & Thomas, T. (2021). Real Time Face Mask Detection and Recognition using Python. International Journal of Engineering Research & Technology (IJERT), 9(07), 57-62.

[12]Yogalakshmi, S., Megalan, L. L., & Simla, A. J. (2020, July). Review on digital image processing techniques for face recognition. In 2020 International Conference on Communication and Signal Processing (ICCSP) (pp. 1633-1637). IEEE.

[13] Abdel-Khalek, S., Abdel-Azim, G., Abo-Eleneen, Z. A., & Obada, A.-S. F. (2016). New Approach to Image Edge Detection Based on Quantum Entropy. Journal of Russian Laser Research, 37(2), 141–154.