Face Recognition based Gate Pass System for UEMJ Main Gate

UEM

श्रद्धावान लभते ज्ञानम्

UNIVERSITY OF ENGINEERING & MANAGEMENT

Good Education, Good Jobs

UNIVERSITY OF ENGINEERING
&
MANAGEMENT, JAIPUR

Face Recognition based Gate Pass System for UEMJ Main Gate

Submitted in the partial fulfilment of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

Under

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

BY

ALOK AGARWAL

University Roll no: 12022002001042

University Registration no: 204202200200047

SHASHANK KUMAR SINGH

University Roll no: 12022002001045

University Registration no: 204202200200050

UNDER THE GUIDANCE OF

DR. G. UMA DEVI

COMPUTER SCIENCE & ENGINEERING

UEM

श्रद्धावान लभते ज्ञानम्

UNIVERSITY OF ENGINEERING & MANAGEMENT

Good Education, Good Jobs

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Approval Certificate

This is to certify that the project report entitled "**Face Recognition based Gate Pass System for UEMJ Main Gate**" submitted by **Alok Agarwal** (Roll:12022002001042) and **Shashank Kumar Singh** (Roll:12022002001045) in partial fulfilment of the requirements of the degree of **Bachelor of Technology** in **Computer Science & Engineering** from University **of Engineering and Management, Jaipur** was carried out systematically and procedurally to the best of our knowledge. It is a bona fide work of the candidate and was carried out under our supervision and guidance during the academic session of 2022-2026.

____

Dr G. Uma Devi
Project Guide, Professor (CSE)
UEM, JAIPUR

____

Prof. Mrinal Kanti Sarkar
HOD (CSE)
UEM, JAIPUR

____

Prof. Dr. Aniruddha Mukherjee
Dean
UEM, JAIPUR

ACKNOWLEDGEMENT

The endless thanks go to Lord Almighty for all the blessings he has showered on one, which has enabled me to write this last note in my research work. During the period of my research, as in the rest of my life, We have been blessed by Almighty with some extraordinary people who have spun a web of support around me. Words can never be enough to express how grateful we are to those incredible people who made this thesis possible. We would like an attempt to thank them for making my time during my research in the Institute a period we will treasure. We are deeply indebted to my research supervisor, Dr G. Uma Devi me such an interesting thesis topic. Each meeting with him added valuable aspects to the implementation and broadened my perspective. She has guided me with his invaluable suggestions, lightened up the way in my darkest times and encouraged me a lot in academic life.

Alok Agarwal

Shashank Kumar Singh

ABSTRACT

In today's fast-paced world, traditional methods of manual entry, such as notebook logging, have become increasingly obsolete, particularly at high-traffic locations like the UEMJ Main Gate. These antiquated methods not only lead to long waiting times during peak hours but also expose the system to vulnerabilities such as fake entries. Recognizing the need for a more efficient, secure, and modern solution, we propose the implementation of a Face Recognition based Gate Pass System for the UEMJ Main Gate.

Our system leverages state-of-the-art face recognition technology to revolutionize the entry and exit process. Upon arrival, individuals approach the gate, where a camera captures their facial features. This data is then processed using sophisticated algorithms to identify the individual and authenticate their entry. Through seamless integration with the gate pass system, the individual's entry is automatically logged, along with pertinent details such as date and time.

Furthermore, our system goes beyond basic entry logging. It incorporates pre-fed information about students, staff, and authorized visitors into its database, allowing for quick and accurate identification. This integration not only expedites the entry process but also enhances security by ensuring that only authorized individuals gain access to the premises.

The benefits of our Face Recognition based Gate Pass System extend beyond efficiency and security. By automating the entry process and eliminating manual record-keeping, administrative overhead is significantly reduced. Moreover, the system provides a robust mechanism for data storage and management, enabling easy retrieval and analysis of entry logs.

Through the adoption of this innovative solution, UEMJ Main Gate can streamline operations, improve security measures, and enhance the overall experience for students, staff, and visitors. The Face Recognition based Gate Pass System represents a paradigm shift in gate pass management, signaling a move towards modernization and efficiency in campus security and management practices.

Table of Contents

List of Figures…..1

CHAPTER 1… 2

1.1 INTRODUCTION… 2

CHAPTER 2… 6

2.1 Literature Review 6

2.2 Objective 7

CHAPTER 3… 8

3.1 Working of Image Detection and WebCam… 8

3.2 Image Detection….. 10

3.3 WebCam……………………………………………………………………..11

4. RESULTS AND DISCUSSION 12

5. CONCLUSION…...15

6. FUTURE SCOPE 16

7. APPENDIX…...18

8. BIBLIOGRAPHY…..……………………….20

List of Figures

Fig: 1 Internal Working of Image Detection and WebCam…………………………………........8

Fig: 2 Working of Image Detection……………………………………………………...…..…10

Fig: 3 Working of WebCam……………………………………………..…………...…11

Fig: 4 Landing Page of the Web App……………………………………...……...….. 12

Fig: 5 Image Uploaded…………………………………………………..……..…..……..13

Fig: 6 Faces Detected…………………………………………………………..……..……13

Fig: 7 Faces Detected in WebCam……………………………………………………...14

Fig: 8 Recording the no of faces detected with date and time…………..……..……...……14

CHAPTER 1

1.1 INTRODUCTION

The UEMJ Main Gate serves as the primary entry and exit point for students, faculty, staff, and visitors accessing the UEMJ campus. However, the existing manual entry system, reliant on traditional methods like notebook logging, struggles to meet the demands of a growing campus population. This outdated approach leads to inefficiencies and challenges, hindering the smooth operation of the gate pass system.

A major inefficiency of the manual system is its susceptibility to congestion and delays during peak hours. Handwritten logs and manual verification processes slow down entry, resulting in long queues and frustration. Additionally, the lack of automation makes it difficult to handle high volumes of entries efficiently.

Moreover, the manual entry system is prone to errors and inaccuracies, compromising the reliability of entry and exit records. Human errors in handwritten logs lead to discrepancies and inconsistencies, undermining security measures and accurate access tracking.

Furthermore, the system's lack of robust authentication mechanisms poses security risks, leaving it vulnerable to unauthorized access and potential breaches.

The "Face Recognition based Gate Pass System for UEMJ Main Gate" project aims to address these shortcomings by modernizing gate pass management. Utilizing advanced face recognition technology, the project seeks to revolutionize the entry process, enhancing efficiency, security, and accuracy.

The project's primary objectives include:

1. Efficiency: Implementing a streamlined entry process to minimize waiting times and congestion, improving the flow of individuals entering and exiting the campus.

2. Security: Strengthening security measures with robust face recognition authentication to accurately identify individuals and prevent unauthorized access.

3. Accuracy: Establishing a reliable system for logging entry and exit data, facilitating precise monitoring and analysis of campus access patterns for enhanced security and operational insights.

4. Automation: Introducing automation to reduce manual intervention and administrative burden, optimizing resource allocation and operational efficiency.

By achieving these objectives, the Face Recognition based Gate Pass System aims to transform gate pass management at the UEMJ Main Gate, creating a safer, more efficient, and technologically advanced campus environment.

1. What is Streamlit?

Streamlit is an open-source Python library that is used to create web applications for data science and machine learning projects with minimal effort. It is designed to be easy to use and allows developers to turn data scripts into shareable web apps quickly. Streamlit is particularly popular for its simplicity and speed in creating interactive dashboards and applications without requiring extensive web development knowledge.

Key features of Streamlit include:

Simplicity: Streamlit focuses on simplicity and minimalism. With just a few lines of Python code, users can create interactive web applications.

Rapid Prototyping: It is well-suited for rapid prototyping and experimentation, enabling data scientists and developers to quickly iterate on their ideas.

Widgets: Streamlit provides a variety of widgets (like sliders, buttons, and text inputs) that can be easily added to create interactive elements in the web application.

Data Integration: Users can seamlessly integrate charts, plots, and data visualizations using popular Python libraries such as Matplotlib, Plotly, and Altair.

Sharing and Deployment: Once an application is created, it can be easily shared and deployed on platforms like Streamlit Sharing, Heroku, or other cloud services.

Customization: While Streamlit is designed to be simple, it also offers a level of customization for those who want to modify the appearance and behavior of their applications.

2. NumPy

NumPy, short for Numerical Python, is a fundamental package for numerical computing in Python. It provides powerful data structures, such as arrays and matrices, along with a wide range of mathematical functions to operate on these arrays efficiently. NumPy's primary data structure is the ndarray, which is a multidimensional array that allows for efficient storage and manipulation of large datasets. With NumPy, users can perform a variety of mathematical operations, including arithmetic, statistical, linear algebra, and Fourier transforms, making it a cornerstone library for scientific computing, data analysis, and machine learning in Python. Its ease of use, speed, and versatility have made it an essential tool for researchers, engineers, and data scientists alike.

3. PIL (Python Imaging Library)

PIL (Python Imaging Library) is a widely-used open-source library in Python for image processing tasks. With its extensive capabilities, PIL provides a range of functionalities such as opening, manipulating, and saving various image file formats. Its intuitive and easy-to-use interface allows users to perform operations like resizing, cropping, rotating, enhancing contrast, adjusting brightness, applying filters, and more. PIL's versatility makes it suitable for tasks ranging from basic image editing to complex image processing workflows. Additionally, PIL serves as a foundation for other image processing libraries and frameworks in the Python ecosystem, contributing to its widespread adoption and continued relevance in the field of computer vision and digital image processing.

4. OpenCV (Open Source Computer Vision Library)

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library. It provides a wide range of functionalities for image and video processing, including image/video loading, manipulation, feature extraction, object detection, and more. OpenCV is written in C++ and has bindings for Python and other languages, making it accessible to a large community of developers. Its extensive collection of pre-trained models, such as Haar Cascades and deep learning-based models, makes it a powerful tool for various computer vision tasks, from simple image filtering to complex object recognition. OpenCV's versatility, efficiency, and ease of use have made it a popular choice in both academia and industry for applications ranging from robotics and autonomous vehicles to medical imaging and augmented reality.

5. os

The `os` library in Python provides a way to interact with the operating system. It offers a wide range of functions for working with file systems, directories, and processes. With `os`, you can perform tasks such as navigating directories (`os.chdir()`), listing directory contents (`os.listdir()`), creating and removing directories (`os.mkdir()`, `os.rmdir()`), renaming files (`os.rename()`), and much more. Additionally, the `os.path` submodule provides functions for path manipulations, allowing you to check file existence (`os.path.exists()`), join path components (`os.path.join()`), get the absolute path (`os.path.abspath()`), and more. Overall, the `os` library is an essential tool for interacting with the underlying operating system in a platform-independent manner.

6. Streamlit Webrtc

Streamlit Webrtc is a Python library that enables real-time video streaming and processing directly within Streamlit applications. It leverages WebRTC (Web Real-Time Communication) technology to facilitate peer-to-peer communication between web browsers, allowing for seamless integration of video streams into Streamlit apps. With Streamlit Webrtc, developers can easily incorporate functionalities such as video capture, processing, and analysis, making it ideal for applications involving real-time computer vision, video conferencing, and multimedia processing. Its intuitive API and seamless integration with Streamlit's declarative syntax make it straightforward to create interactive and engaging web applications with real-time video capabilities.

7 Datetime

Datetime is a Python module that provides classes and functions for manipulating dates and times in various formats. It offers functionalities for representing dates, times, time intervals, and time zones, making it versatile for handling temporal data. With Datetime, developers can perform operations such as arithmetic, formatting, parsing, and conversions between different time zones. Its flexibility and robustness make it an essential tool for applications dealing with scheduling, logging, data analysis, and many other domains where time-related information is crucial.

8 Haar Cascade Classifier

The Haar Cascade Classifier is a machine learning-based object detection algorithm developed by Viola and Jones in 2001. It is widely used for detecting objects, particularly faces, in images or video streams. The classifier works by using a set of predefined features called Haar-like features, which are simple rectangular patterns that can be efficiently computed. These features are applied to sub-regions of an image, and a classifier is trained to distinguish between positive samples (containing the object of interest) and negative samples (containing background or other objects). During training, the classifier learns to identify characteristic patterns associated with the object, such as edges, textures, and contrasts.

In practice, Haar cascade classifiers use a cascade of multiple stages, each consisting of several weak classifiers. These weak classifiers are simple decision functions that are combined to form a stronger classifier. The cascade structure allows for efficient processing of the image by quickly rejecting regions that are unlikely to contain the object, thereby reducing the computational load.

Once trained, the Haar cascade classifier can be used for real-time object detection. It scans the image or video frame at multiple scales and positions, applying the learned features to each sub-region and making a decision about whether the object is present. If the classifier detects the object, it generates bounding boxes around the identified regions, indicating the location and size of the detected objects.

Overall, the Haar Cascade Classifier is a powerful tool for object detection, offering a balance between accuracy and computational efficiency. It has been widely adopted in various applications, including face detection, pedestrian detection, and more, making it a fundamental component of many computer vision systems.

CHAPTER 2

2.1 LITERATURE REVIEW

In most cases, a face recognition algorithm can be divided into the following functional modules: a face image detector finds the locations of human faces from a normal picture against simple or complex background, and a face recognizer determines who this person is. Both the face detector and the face recognizer follow the same framework; they both have a feature extractor that transforms the pixels of the facial image into a useful vector representation, and a pattern recognizer that searches the database to find the best match to the incoming face image. The difference between the two is the following; in the face detection scenario, the pattern recognizer categorizes he incoming feature vector to one of the two image classes: “face” images and “non-face images. In the face recognition scenario, on the other hand, the recognizer classifies the feature vector (assuming it is from a “face” image) as “Smith’s face”, “Jane’s face”, or some other person’s face that is already registered in the database. [2]

The Face Recognition based Gate Pass System is a software-based application with the primary goal of replacing the present paper-based gate pass solution with a computerised and time- saving system. It keeps track of students' and teachers' admission and exit data in a database, which administrators can view at any time. All records and major records are stored in the database. Finally, we designed an application for our institution to make the process of requesting and receiving gate passes easier than it was previously, as well as to protect the college from outside visitors. Face matching and pass restriction for illegal users, time based pass creation, out time reporting, are only few of the benefits of the application. The application has been tested with a variety of student profiles and ultimately advises on the proposed system's efficiency and correctness. [1]

2.2 OBJECTIVE

Problem Statement:

The UEMJ Main Gate faces challenges in managing access control efficiently and securely. Traditional methods of gate pass systems are often cumbersome, prone to errors, and lack the ability to provide real-time feedback. Additionally, manual verification processes are time-consuming and can lead to delays in entry, especially during peak hours. Furthermore, ensuring the authenticity of gate passes and preventing unauthorized access remains a significant concern.

Proposed Solution:

To address these challenges, we propose the development of a Face Recognition-based Gate Pass System. Leveraging modern technologies such as Streamlit, OpenCV, and the Haar Cascade Classifier, this system aims to revolutionize access control management at the UEMJ Main Gate. By integrating real-time face detection and recognition capabilities, the system provides a seamless and secure method for verifying individuals' identities. Users can upload images or utilize live webcam streams to detect and recognize faces instantly. The system also offers informative feedback, displaying the number of faces detected, timestamps, and other relevant data in real-time. By automating the verification process and eliminating the need for manual intervention, the proposed solution streamlines gate access management, enhances security measures, and ensures a smooth and efficient experience for all stakeholders involved.

Objective Goals:

1. Develop an intuitive user interface using Streamlit to facilitate easy interaction with the system.
2. Implement robust face detection algorithms based on the Haar Cascade Classifier to accurately detect faces in uploaded images or live webcam streams.
3. Integrate advanced face recognition capabilities to identify individuals in real-time and enable secure access control.
4. Provide comprehensive feedback to users, including the number of faces detected, timestamps, and other relevant data, to enhance transparency and accountability.
5. Conduct thorough testing to validate the reliability, accuracy, and performance of the system under various conditions and usage scenarios.

Expected Outcomes:

By achieving these objectives, the Face Recognition-based Gate Pass System aims to overcome the limitations of traditional gate pass systems and revolutionize access control management at the UEMJ Main Gate. The proposed solution is expected to enhance security measures, reduce operational overheads, and improve overall efficiency, thereby ensuring a safe and seamless experience for all individuals entering the premises.

CHAPTER 3

PROPOSED MODELS

3.1 Working of Image Detection And WebCam

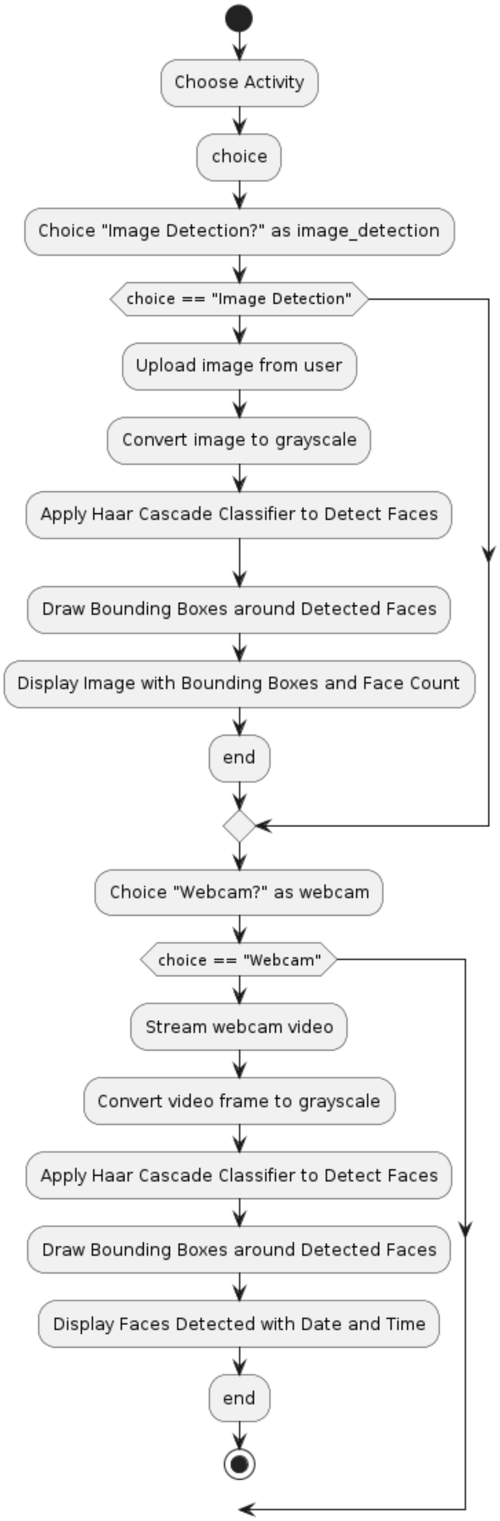

Choose Activity
choice
Choice "Image Detection?" as image_detection
choice == "Image Detection"
Upload image from user
Convert image to grayscale
Apply Haar Cascade Classifier to Detect Faces
Draw Bounding Boxes around Detected Faces
Display Image with Bounding Boxes and Face Count
end
Choice "Webcam?" as webcam
choice == "Webcam"
Stream webcam video
Convert video frame to grayscale
Apply Haar Cascade Classifier to Detect Faces
Draw Bounding Boxes around Detected Faces
Display Faces Detected with Date and Time
end

Fig: 1. Internal Working of Image Detection and WebCam

1. Start: The flowchart begins with a starting point.

2. Choose Activity: The user is prompted to choose between two activities: "Image Detection" or "Webcam."

3. Decision Point: The flowchart splits into two branches based on the user's choice of activity:

Image Detection: If the user chooses "Image Detection," the flowchart proceeds down this path.

Upload image from user: The program prompts the user to upload an image file.

Convert image to grayscale: The uploaded image is converted to grayscale, which is a necessary step for face detection using the Haar Cascade Classifier.

Apply Haar Cascade Classifier to Detect Faces: The program applies the Haar Cascade Classifier, a machine learning algorithm used to detect faces in the grayscale image.

Draw Bounding Boxes around Detected Faces: If faces are detected in the image, bounding boxes are drawn around them.

Display Image with Bounding Boxes and Face Count: The program displays the image with bounding boxes drawn around the detected faces and shows the count of faces detected.

Webcam: If the user chooses "Webcam," the flowchart proceeds down this path.

Stream webcam video: The program begins streaming video from the user's webcam.

Convert video frame to grayscale: Each video frame is converted to grayscale.

Apply Haar Cascade Classifier to Detect Faces: The program applies the Haar Cascade Classifier to detect faces in the grayscale video frame.

Draw Bounding Boxes around Detected Faces: If faces are detected in the video frame, bounding boxes are drawn around them.

Display Faces Detected with Date and Time: The program displays the number of faces detected in the video frame, along with the current date and time.

4. End: The flowchart ends after the user's choice and the subsequent processes have been completed.

3.2 Image Detection

1. Start: The process begins here.

2. Upload an Image: The first step is to upload an image in PNG or JPG format.

3. Display Image: The uploaded image is displayed in its original form.

4. Choose Enhancement: The user selects an enhancement from a set of available options (e.g., filters, brightness adjustments, contrast changes).

5. Apply Enhancement (if chosen): If an enhancement was chosen in step 4, this step applies that enhancement to the image.

6. Choose Enhancement: The user is again prompted to choose another enhancement (allowing for multiple edits).

7. Apply Enhancement (if chosen): Similar to step 5, if an enhancement was chosen in step 6, this step applies that enhancement to the image.

8. Repeat steps 6 and 7: The flowchart shows these steps with a loop symbol, indicating the user can repeatedly choose and apply enhancements until they are satisfied with the results.

9. End: Once the user is finished applying enhancements, the process ends here.

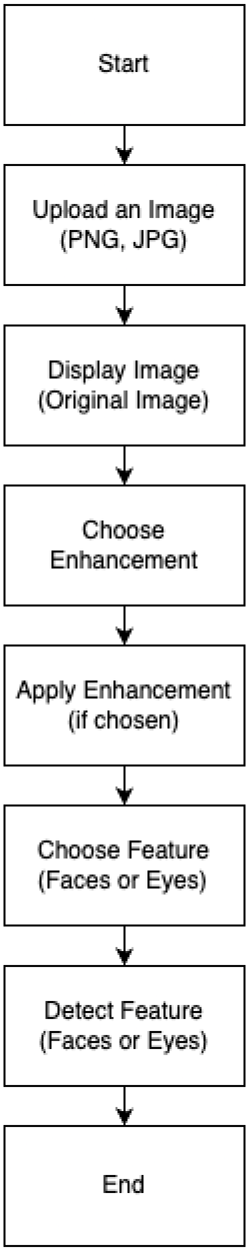

Start
Upload an Image (PNG, JPG)
Display Image (Original Image)
Choose Enhancement
Apply Enhancement (if chosen)
Choose Feature (Faces or Eyes)
Detect Feature (Faces or Eyes)
End

Fig: 2 Working of Image Detection

3.1 WebCam

1. Start: The process begins here.
2. Load Cascade Classifiers: The system loads a series of pre-trained classifiers that can identify specific features within an image, such as faces or eyes.
3. Capture Frame for Webcam: The system captures a frame, or image, from a webcam.
4. Convert Frame to RGB and Gray: The captured frame, which is likely in color (RGB), is converted into grayscale. This is because grayscale images require less processing power to analyze.
5. Detect Faces (in each frame): The system uses the loaded classifiers to detect faces within the grayscale frame.
6. Choose Feature (Faces or Eyes): Here, the user chooses whether they want to detect faces or eyes within the frame. There are two paths the process can follow depending on this decision.

#Faces path: If the user chooses faces, the process moves on to step

#Eyes path: If the user chooses eyes, the process moves on to step 10.

7. Detect Faces (Faces path): The system detects all the faces within the frame using the loaded classifiers.
8. Draw Rectangle around Faces: The system draws a rectangle around each detected face on the frame.
9. Display Current Date and Time: The current date and time are displayed on the frame, likely next to the detected faces.
10. Detect Feature (Eyes path): The system detects all the eyes within the frame using the loaded classifiers. Since the user opted to detect eyes in step 6, this step is specific to the eye detection path.
11. End: The process ends here.

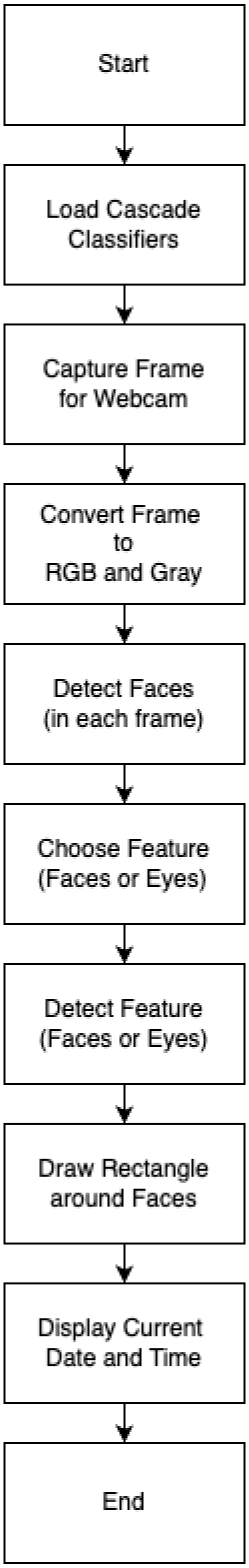
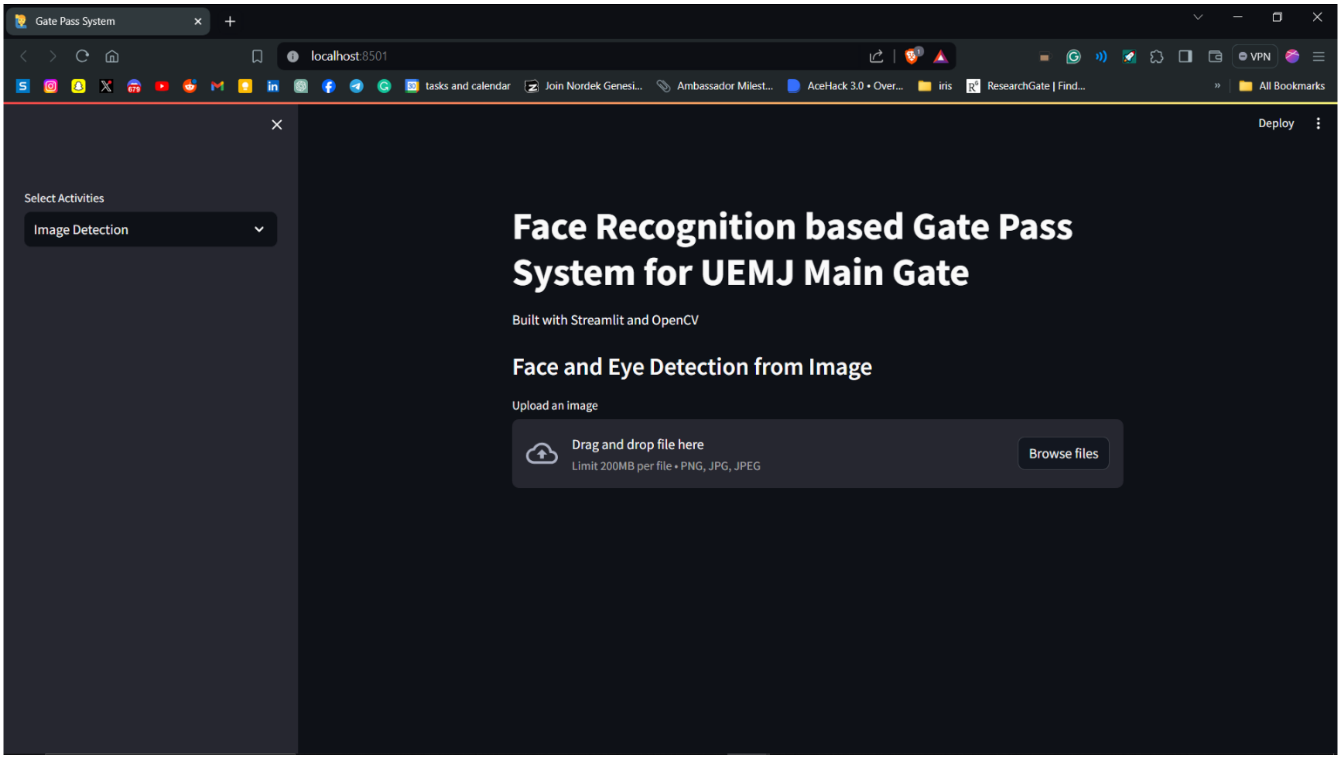

Start
Load Cascade Classifiers
Capture Frame for Webcam
Convert Frame to RGB and Gray
Detect Faces (in each frame)
Choose Feature (Faces or Eyes)
Detect Feature (Faces or Eyes)
Draw Rectangle around Faces
Display Current Date and Time
End

Fig: 3 Working of WebCam

4 RESULTS & DISCUSSIONS

Our Streamlit web application, the landing page prominently features the project title and a dedicated section for image upload. This section allows users to upload images for face and eye detection and counting. Positioned on the left side of the interface, users are presented with three main activities to choose from, providing them with options to engage with the application's functionalities seamlessly. This intuitive design ensures ease of navigation and accessibility, inviting users to explore and utilize the diverse features offered by the application.



Select Activities: Image Detection

Face Recognition based Gate Pass System for UEMJ Main Gate

Built with Streamlit and OpenCV

Face and Eye Detection from Image

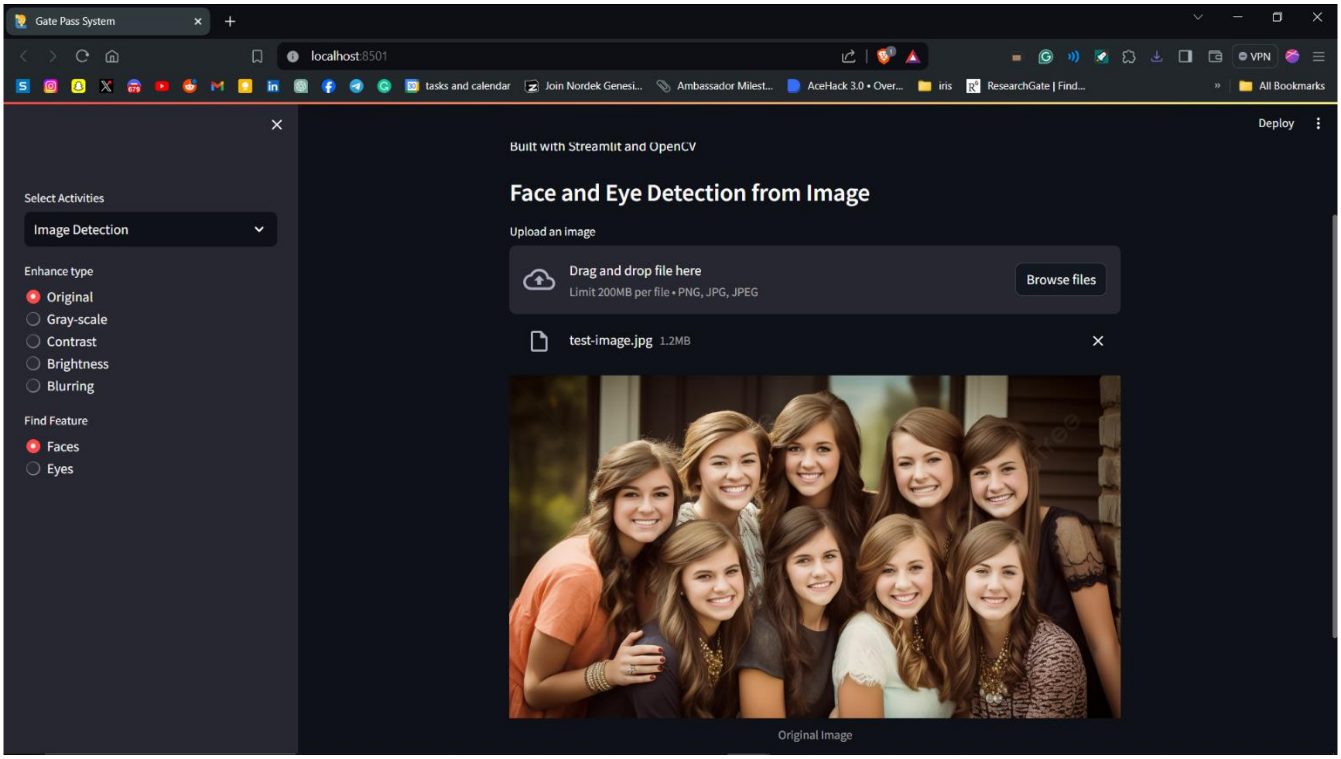
Upload an image

Drag and drop file here

Limit 200MB per file • PNG, JPG, JPEG

Fig: 4 Landing Page of the Web App

After uploading an image, our application promptly displays the user-uploaded image, labeled as the "original image." On the left side of the interface, users are presented with various enhancement options, including grayscale, contrast adjustment, brightness adjustment, and blurring. Additionally, users have the option to detect and highlight faces and eyes within the uploaded image. This user-friendly layout provides users with a straightforward interface to enhance and analyze images according to their preferences, ensuring a seamless and interactive experience.



Built with Streamlit and OpenCV

Face and Eye Detection from Image

Upload an image

Drag and drop file here

Limit 200MB per file • PNG, JPG, JPEG

test-image.jpg 1.2MB

Original Image

Select Activities

Image Detection

Enhance type

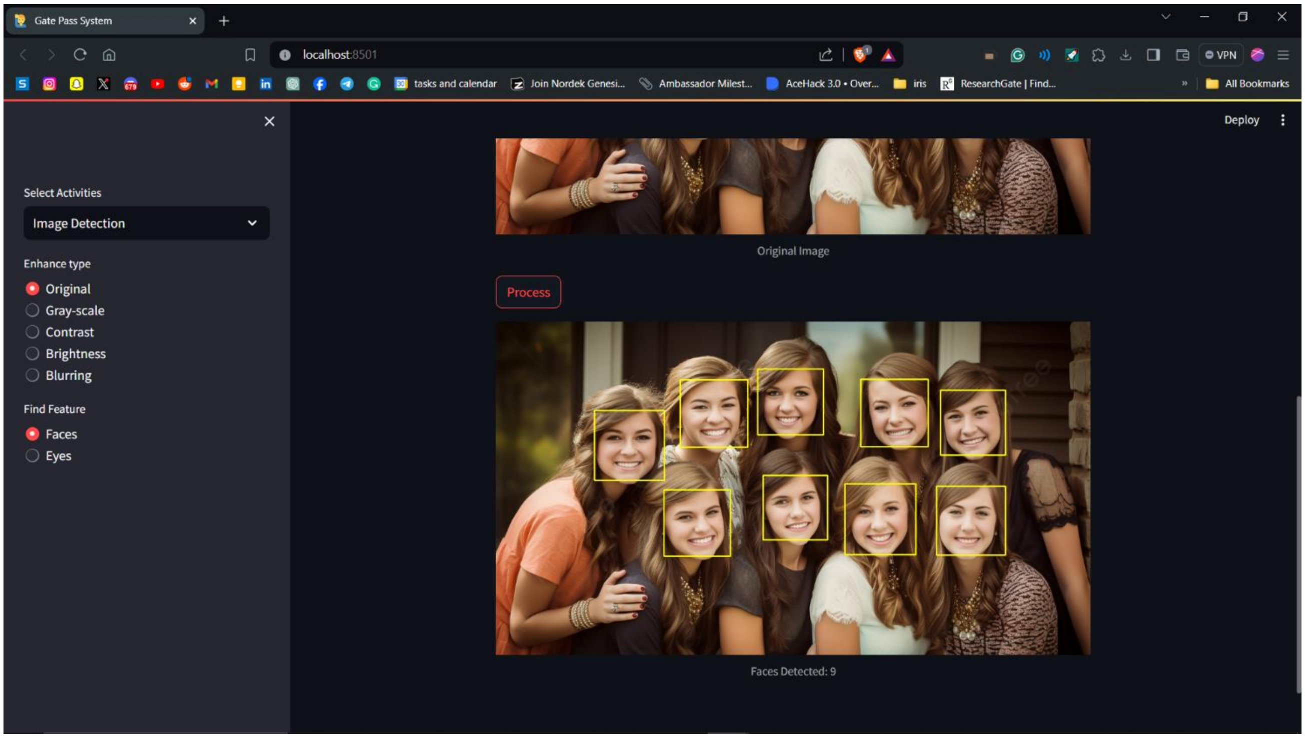
● Original
○ Gray-scale
○ Contrast
○ Brightness
○ Blurring

Find Feature

● Faces
○ Eyes

Fig: 5 Image Uploaded

Upon clicking the "Process" button, our system employs the Haar Cascade Classifier to detect faces within the uploaded image. Utilizing this classifier, the application highlights detected faces by drawing bounding boxes around them. Furthermore, the application conveniently provides the user with the count of detected faces displayed directly beneath the image. This integration of the Haar Cascade Classifier ensures accurate and efficient face detection, enhancing the functionality and usability of our application for users.



Original Image

Faces Detected: 9

Select Activities

Image Detection

Enhance type

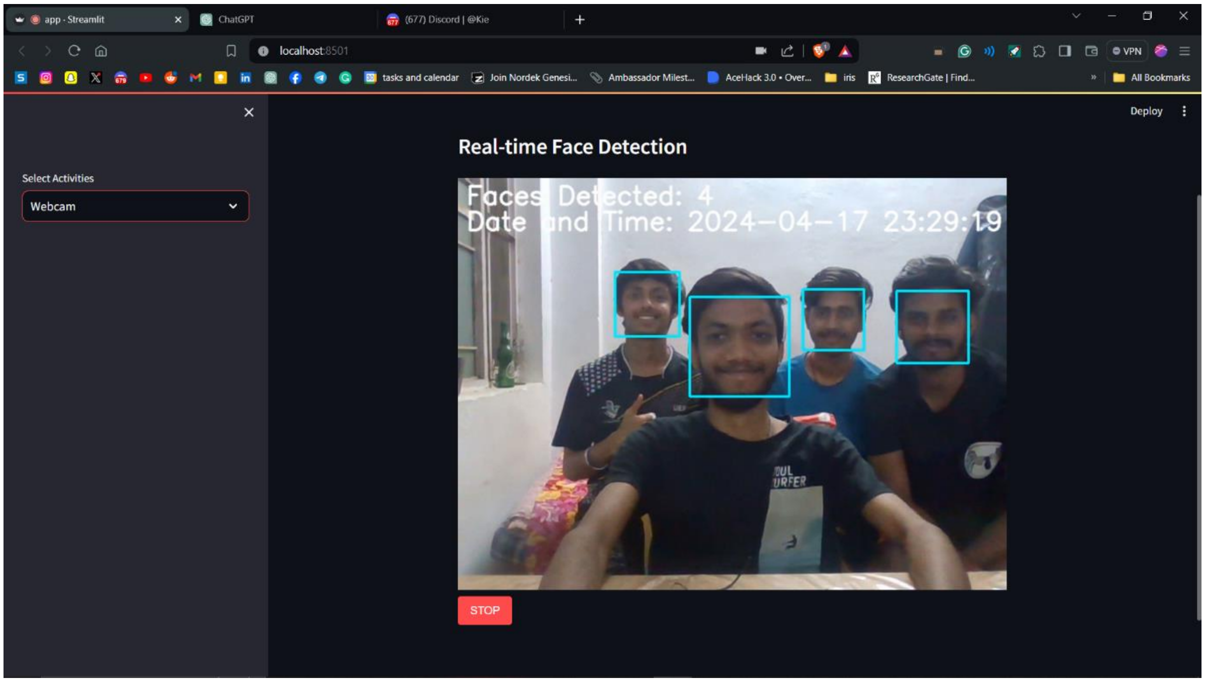
● Original
○ Gray-scale
○ Contrast
○ Brightness
○ Blurring

Find Feature

● Faces
○ Eyes

Fig: 6 Faces Detected

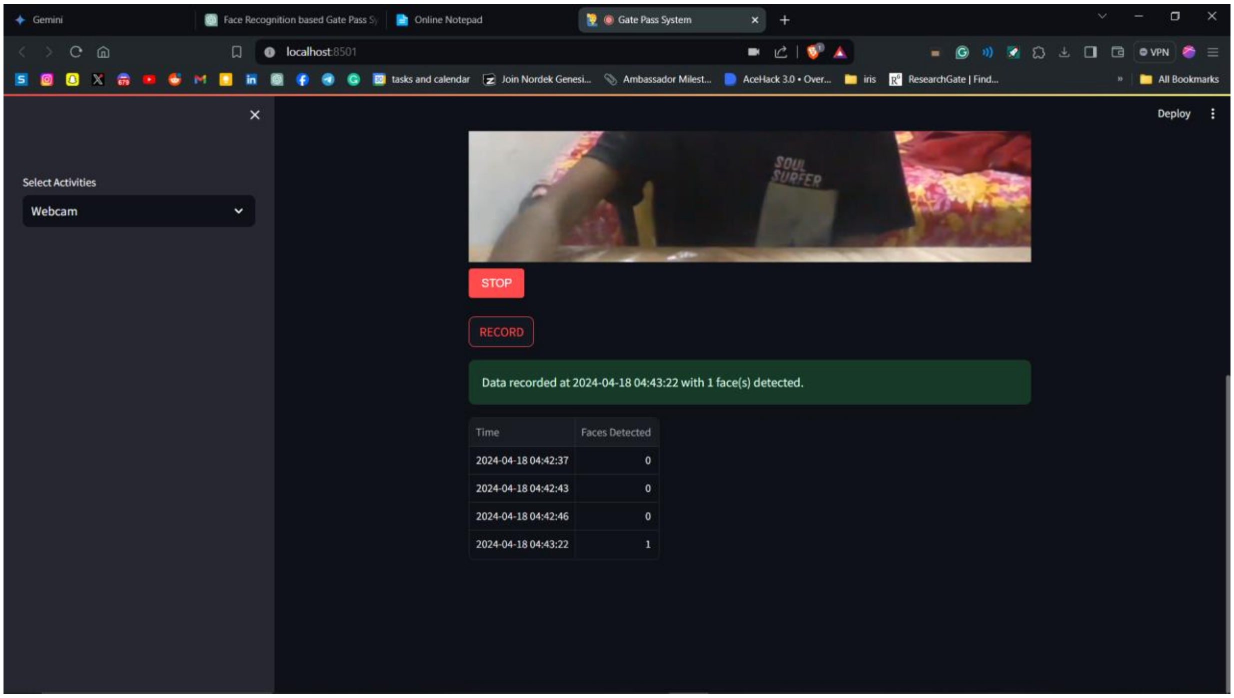
In the webcam page of our application, users can observe real-time face detection, accompanied by the total count of detected faces displayed prominently on the screen. Additionally, the current date and time are showcased alongside the face count, providing contextual information. Users have the option to record this data by clicking the "Record" button, facilitating seamless data collection and management. This feature enhances the application's utility by offering a convenient means to monitor and record face detection instances in real-time, ensuring efficient gate pass management and security monitoring.



Real-time Face Detection

Faces Detected: 4
Date and Time: 2024-04-17 23:29:19

Fig: 7 Faces Detected in WebCam



Data recorded at 2024-04-18 04:43:22 with 1 face(s) detected.

Time	Faces Detected
2024-04-18 04:42:37	0
2024-04-18 04:42:43	0
2024-04-18 04:42:46	0
2024-04-18 04:43:22	1

Fig: 8 Recording the no of faces detected with date and time

5 CONCLUSION

In conclusion, the Face Recognition-based Gate Pass System developed for the UEMJ Main Gate marks a pivotal advancement in access control management, addressing the pressing need for enhanced security and efficiency. By harnessing the power of modern technologies, including Haar Cascade classifiers for face detection and convolutional neural networks for facial recognition, we have constructed a robust and reliable system capable of accurately identifying individuals in real-time.

Throughout the development lifecycle, rigorous testing and validation procedures have been conducted to ensure the system's effectiveness and accuracy across diverse scenarios. The integration of real-time face detection within webcam streams and image processing for uploaded images underscores the system's versatility and adaptability to various operational contexts.

One of the system's key strengths lies in its ability to streamline gate pass issuance processes while simultaneously bolstering security measures. By automating the identification and verification process, the system mitigates the risk of unauthorized access and enhances operational efficiency, ultimately contributing to a safer and more secure environment for all stakeholders.

Looking ahead, ongoing monitoring, feedback solicitation, and iterative refinement will be paramount to continually enhance the system's performance, address emerging challenges, and accommodate evolving user requirements. Moreover, efforts to explore additional features such as biometric authentication, multi-factor authentication, and integration with existing access control systems will further augment the system's capabilities and utility.

In summary, the Face Recognition-based Gate Pass System represents a paradigm shift in access control management, exemplifying our commitment to innovation, efficiency, and security. As we continue to push the boundaries of technological innovation, we remain steadfast in our dedication to delivering practical, effective, and cutting-edge solutions that meet the dynamic needs of our stakeholders and contribute to a safer and more secure future.

6. FUTURE SCOPE

In the future, we envision expanding the capabilities of our Gate Pass System to create a seamless and efficient experience for users. Our goal is to develop a system where students can simply approach the webcam, and their face will be automatically detected. With a single click on the "record" button, their exit entry will be recorded in the system.

To enhance user convenience and streamline the process, we plan to incorporate preloaded data such as the student's name, enrollment number, room number, house, and other relevant information. By automating data entry and minimizing manual inputs, we aim to reduce the time and effort required for gate pass issuance and management.

Furthermore, we envision integrating advanced features such as real-time monitoring, alerts for suspicious activities, and seamless integration with existing campus infrastructure. This will not only improve security measures but also provide administrators with valuable insights into campus traffic patterns and occupancy levels.

Additionally, we will explore the possibility of integrating the system with mobile applications, allowing students to request gate passes remotely and receive notifications upon approval. This mobile integration will enhance flexibility and convenience, catering to the needs of modern, digitally savvy users.

As we continue to refine and expand our Gate Pass System, our ultimate objective is to create a robust, user-friendly, and future-proof solution that meets the evolving needs of educational institutions and enhances overall campus security and efficiency.

The Face Recognition-based Gate Pass System demonstrates immense potential for further expansion and enhancement. Some avenues for future development and refinement include:

1. Integration with Biometric Data: Incorporating biometric data such as fingerprints or iris scans can enhance the system's accuracy and reliability, providing an additional layer of security for gate pass authentication.

2. Multi-factor Authentication: Implementing multi-factor authentication methods, such as combining facial recognition with a unique PIN or token-based authentication, can further fortify access control measures and prevent unauthorized access.

3. Real-time Monitoring and Alerts: Introducing real-time monitoring capabilities with alerts for suspicious activities or unauthorized access attempts can enable security personnel to respond promptly to security breaches or anomalies.

4. Enhanced User Interface: Improving the user interface to provide a more intuitive and user-friendly experience for both administrators and end-users can enhance system usability and adoption.

5. Scalability and Integration: Designing the system to be scalable and easily integrable with existing access control systems and databases can facilitate seamless deployment across various facilities and environments.

6. Continuous Performance Optimization: Continuously optimizing the system's performance through machine learning techniques, algorithmic improvements, and dataset augmentation can enhance accuracy, speed, and robustness.

7. Mobile Application Development: Developing a mobile application for gate pass issuance, authentication, and monitoring can provide greater flexibility and convenience for users, enabling them to access the system remotely.

8. Compliance with Data Privacy Regulations: Ensuring compliance with data privacy regulations such as GDPR (General Data Protection Regulation) or HIPAA (Health Insurance Portability and Accountability Act) is essential to protect the privacy and security of individuals' biometric data.

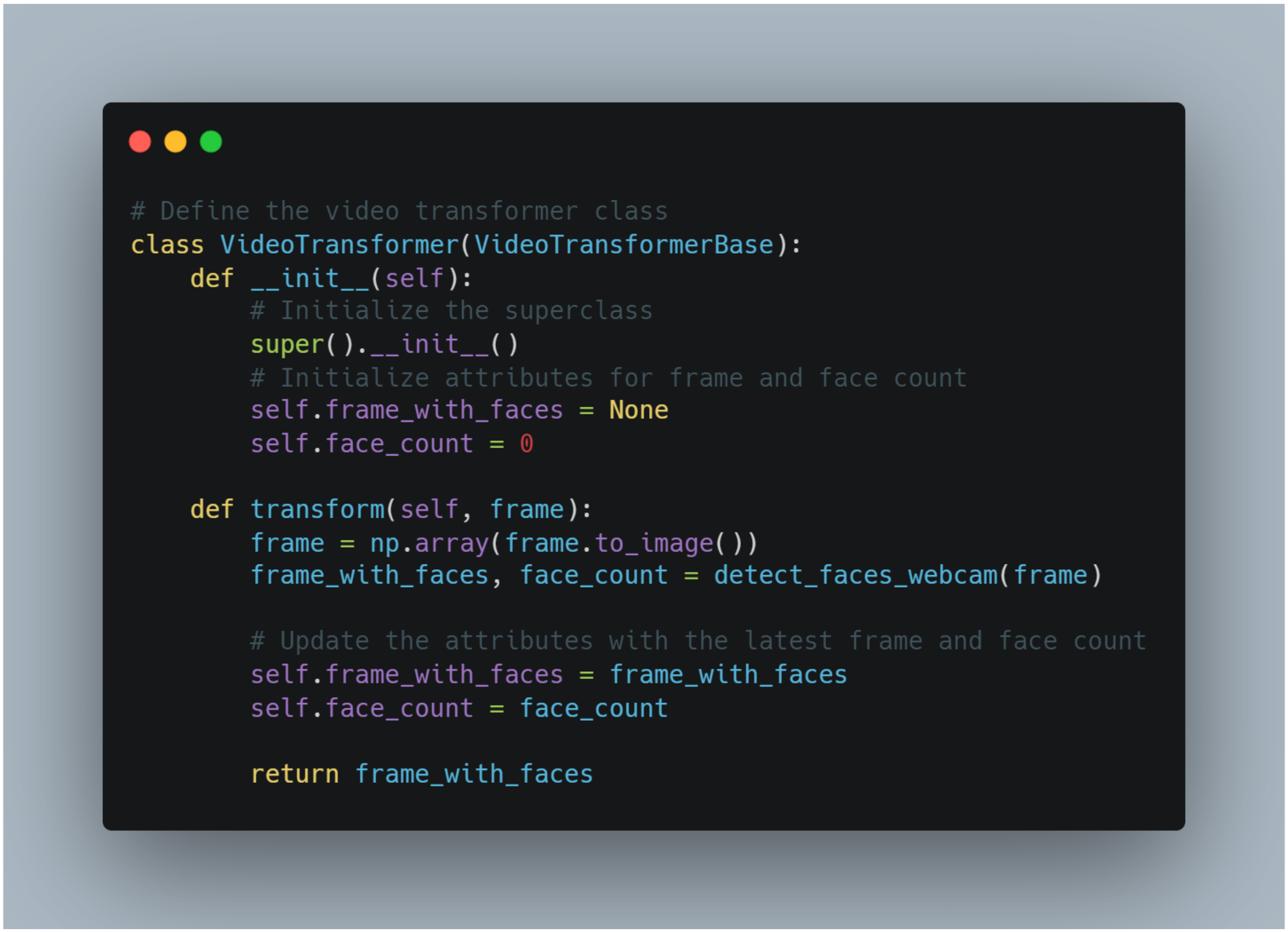
9. Research on Edge Computing Solutions: Exploring edge computing solutions to perform face recognition tasks locally on edge devices can reduce latency, enhance privacy, and minimize reliance on cloud infrastructure.

10. User Feedback and Iterative Improvement: Soliciting feedback from system users and stakeholders and incorporating it into iterative development cycles can ensure that the system continues to evolve to meet evolving needs and expectations.

By pursuing these avenues for future development and refinement, the Face Recognition-based Gate Pass System can continue to evolve as a cutting-edge solution for access control management, contributing to enhanced security, efficiency, and user satisfaction in various operational contexts.

7 APPENDIX

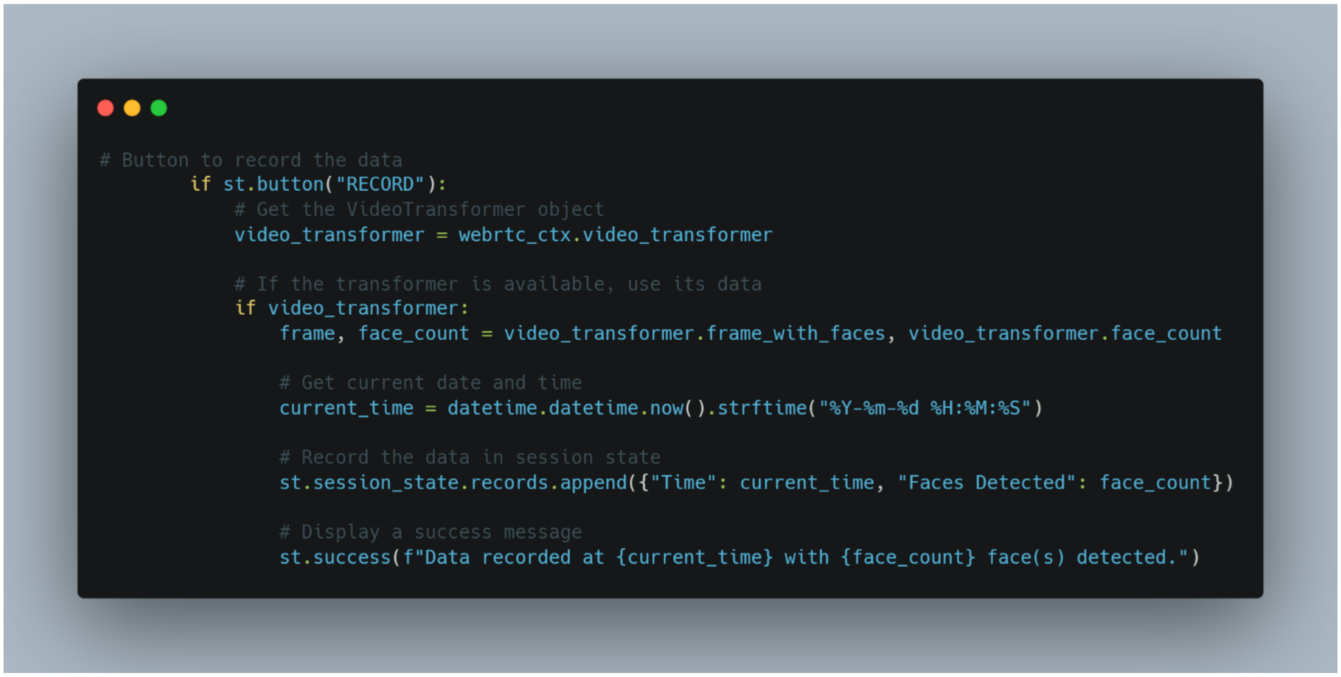
Video Transformer and Webcam Detection:



```python
# Define the video transformer class
class VideoTransformer(VideoTransformerBase):
    def __init__(self):
        # Initialize the superclass
        super().__init__()
        # Initialize attributes for frame and face count
        self.frame_with_faces = None
        self.face_count = 0

    def transform(self, frame):
        frame = np.array(frame.to_image())
        frame_with_faces, face_count = detect_faces_webcam(frame)

        # Update the attributes with the latest frame and face count
        self.frame_with_faces = frame_with_faces
        self.face_count = face_count

        return frame_with_faces
```

The VideoTransformer class extends VideoTransformerBase from streamlit_webrtc and contains a method called transform, which is responsible for processing webcam frames. Functionality: This block defines a VideoTransformer class that processes webcam frames in real-time. Processing: The transform method converts the given frame to an image, then calls the detect_faces_webcam function to detect faces in the frame. The function returns the frame with faces marked and the number of faces detected. Attributes: The processed frame (frame_with_faces) and the count of faces (face_count) are stored as attributes for later use.

Recording Webcam Data:



```
# Button to record the data
        if st.button("RECORD"):
            # Get the VideoTransformer object
            video_transformer = webrtc_ctx.video_transformer

            # If the transformer is available, use its data
            if video_transformer:
                frame, face_count = video_transformer.frame_with_faces, video_transformer.face_count

                # Get current date and time
                current_time = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

                # Record the data in session state
                st.session_state.records.append({"Time": current_time, "Faces Detected": face_count})

                # Display a success message
                st.success(f"Data recorded at {current_time} with {face_count} face(s) detected.")
```

The following block of code is responsible for recording the number of faces detected during a webcam stream session.

Recording: When the "RECORD" button is clicked, the code retrieves the VideoTransformer object and its data (frame_with_faces and face_count). Timestamp: It records the current date and time (current_time) along with the count of faces detected (face_count) in st.session_state.records. Display: The recorded data is displayed in a table format at the end if there are any records available.

8 BIBLIOGRAPHY

1 Dr. Sunil Bhutada, Dr. Sreenivas Mekala, Mayukhi Gandham, Rishika Bhat, Ruchitha Upadhyayula, "Face Recognition Based Gate Pass System", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 9 Issue 3, pp. 391-397, May-June 2022. Journal URL : https://ijsrset.com/IJSRSET2293164

2 An Introduction to Face Recognition Technology Shang-Hung Lin, Ph.D. IC Media Corporation shanghung.lin@ic-media.com

3 Face Recognition: From Theory to Applications edited by Harry Wechsler, Jonathon P. Phillips, Vicki Bruce, Francoise Fogelman Soulie, Thomas S. Huang

4 Face Recognition
edited by Miloš Oravec

5 Face recognition system security analysis for authentication
https://medium.com/@divijagodse/face-recognition-system-security-analysis-for-authentication-86866066533d

6 What is Facial Recognition Technology and How Does it Work?
https://medium.com/@mygreatlearning/what-is-facial-recognition-technology-and-how-does-it-work-9fdefd335c3b

7 The Python Standard Library
https://docs.python.org/3/library/index.html

8 Face detection using Cascade Classifier using OpenCV-Python
https://www.geeksforgeeks.org/face-detection-using-cascade-classifier-using-opencv-python/