Facial Geo Lock

Next Level Attendance Authentication with Location Tracking

SECURE SOFTWARE PROGRAMMING

SWE 681

PROJECT REPORT
(Professor Lisa Luo)

April 19, 2024

TEAM MEMBERS

Sai Subbarao Gunisetty – G01408517
Saaketh Matta – G01413120
Sushmitha Konduru – G01456225
Sushrutha Reddy Anireddy – G01446984

INDEX

INTRODUCTION

In today's fast-paced digital world, ensuring accurate attendance tracking while maintaining security and convenience is paramount for various institutions and organizations. Traditional methods of attendance tracking often prove to be cumbersome and prone to errors. To address these challenges, we present our innovative solution: Facial GeoLock – a next-level attendance authentication system with location tracking capabilities.

PROJECT OVERVIEW

Facial GeoLock revolutionizes attendance authentication by leveraging facial recognition technology combined with geolocation tracking. This system offers enhanced security, accurate attendance tracking, and scalability while ensuring efficiency and convenience for users. By integrating advanced biometric authentication and location-based verification, Facial GeoLock sets a new standard for attendance management systems.

COMPONENTS DEVELOPED BY OUR GROUP

1. Enhanced Facial Recognition Module

- **Description**: Our team developed an enhanced facial recognition module by integrating various components and customizing the base code obtained from existing works.

- **Development Process:** We began with a base facial recognition algorithm obtained from open-source repositories. We then enhanced this algorithm by incorporating the Haar cascades model for improved face detection accuracy.

- **Innovations:** Our enhanced facial recognition module incorporates advanced face detection techniques, resulting in higher accuracy and robustness in identifying faces from input images or video streams.

2. Geolocation Integration

- **Description**: We extended the functionality of our facial recognition system by integrating geolocation capabilities, enabling the identification and tracking of individuals based on their geographic location.

- **Development Process**: We leveraged Python libraries for geolocation services and seamlessly integrated them into our existing system architecture.

- **Innovations**: Our geolocation integration allows for the association of facial recognition data with specific geographical coordinates, facilitating applications such as location-based access control or attendance tracking.

COMPONENTS DERIVED FROM EXISTING WORKS

3. Base Facial Recognition Algorithm

- **Description**: The core facial recognition algorithm forms the foundation of our system and was derived from existing open-source projects and libraries.

- **Source**: The base code for facial recognition was obtained from Github sources.

- **Customizations**: While the base code provided the framework for facial recognition functionality, we customized it by integrating additional features such as the Haar cascades model for enhanced face detection.

KEY FEATURES

1) **Advanced Attendance Authentication System:**

Facial GeoLock leverages state-of-the-art facial recognition technology to accurately authenticate users' identities. This ensures that only authorized individuals can mark attendance, enhancing overall security and reliability. By analyzing unique facial features, Facial GeoLock provides a seamless and efficient authentication process, eliminating the need for traditional methods like swipe cards or PIN codes.

2) **Multi-Factor Authentication:**

Facial GeoLock employs a robust two-factor authentication mechanism by combining facial recognition with geolocation verification. This multi-layered approach significantly enhances security by requiring users to provide multiple forms of authentication. Geolocation verification adds an extra layer of security by confirming the user's physical presence at specified locations. This prevents unauthorized access attempts, even if someone tries to impersonate another user's facial identity.

3) **Prevents Buddy Punching:**

One of the key challenges in traditional attendance systems is buddy punching, where employees mark attendance on behalf of their colleagues. Facial GeoLock effectively addresses this issue by capturing unique biometric data – the user's facial features. By requiring each user to physically present their face for authentication, Facial GeoLock eliminates the possibility of fraudulent attendance marking, ensuring accuracy and integrity in attendance records.

4) **Location Tracking:**

Integrated geolocation tracking in Facial GeoLock adds an additional layer of security by restricting attendance marking to specific locations.By verifying the user's geographical location during the attendance process, Facial GeoLock ensures that attendance can only be marked from

designated areas. This prevents unauthorized access attempts from remote or unauthorized locations, enhancing overall security and compliance.

OBJECTIVES

The primary objectives of Facial GeoLock are:

1) **Enhance Security and Reliability:**

The primary objective of Facial GeoLock is to enhance the security and reliability of attendance authentication systems. By leveraging advanced biometric technologies and multi-factor authentication, Facial GeoLock ensures that only authorized individuals can access attendance records, minimizing the risk of unauthorized access or fraudulent activities.

2) **Prevent Fraudulent Attendance Practices:**

Facial GeoLock aims to prevent fraudulent attendance practices such as buddy punching, where employees may attempt to manipulate attendance records. By capturing unique biometric data and implementing strict authentication measures, Facial GeoLock ensures the integrity and accuracy of attendance records, promoting fairness and transparency in attendance management.

3) **Ensure Compliance and Accountability:**

Another objective of Facial GeoLock is to ensure compliance with organizational policies and regulations regarding attendance management. By implementing features such as location tracking and multi-factor authentication, Facial GeoLock helps organizations maintain compliance with attendance policies and promotes accountability among employees.

4) **Enhance User Experience and Convenience:**

Facial GeoLock strives to provide a seamless and user-friendly experience for both administrators and users. By automating the attendance authentication process and eliminating the need for manual intervention, Facial GeoLock enhances convenience and efficiency, allowing users to focus on their core tasks without unnecessary administrative burdens.

BENEFITS:

1) **Enhanced Security:** By leveraging advanced biometric authentication and geolocation tracking, Facial GeoLock enhances the security and integrity of attendance management systems.
2) **Accurate Attendance Tracking:** The system provides accurate and reliable attendance tracking, minimizing errors and discrepancies associated with traditional methods.
3) **Efficiency and Convenience:** Facial GeoLock streamlines the attendance authentication process, offering a convenient and user-friendly experience for both administrators and users.
4) **Compliance**: Facial GeoLock helps organizations maintain compliance with regulatory requirements and internal policies related to attendance management and data privacy.

FEATURES AND TECHNOLOGIES IMPLEMENTED

Facial GeoLock incorporates several key technologies and features:

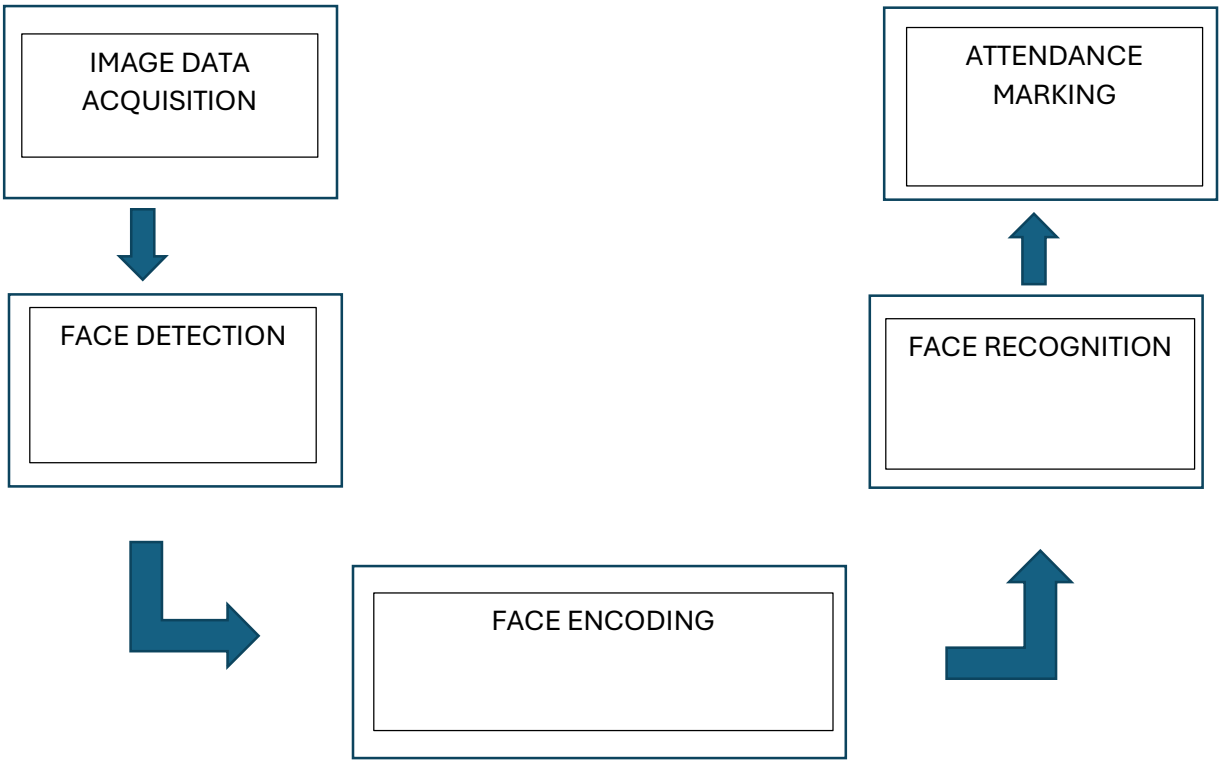
1) **OpenCV**: Utilized for face detection and encoding, OpenCV plays a crucial role in processing facial data within the system.
2) **Flask**: As a lightweight web framework, Flask facilitates the development of the system's backend and API endpoints, enabling seamless communication between components.
3) **Face_recognition Library**: This library powers facial recognition tasks within Facial GeoLock, allowing for accurate identification and authentication of users.
4) **Geocoder**: Integrated for geolocation tracking, the Geocoder library ensures that users can only mark attendance from designated locations.
5) **Multi-Factor Authentication**: Implemented to enhance security, Facial GeoLock combines facial recognition with geolocation verification for robust authentication.

SECURITY MEASURES IMPLEMENTED

To ensure the security and privacy of user data, Facial GeoLock implements the following measures:

1) **Multi-Factor Authentication** (MFA): By requiring multiple forms of verification, including facial recognition and geolocation tracking, Facial GeoLock enhances security and mitigates the risk of unauthorized access.
2) **Secure Communication Protocols**: Facial GeoLock utilizes HTTPS (Hypertext Transfer Protocol Secure) to encrypt data transmitted between client devices and the server. HTTPS ensures that data exchanged over the network is encrypted and protected from interception, safeguarding data integrity and confidentiality.
3) **Access Control**: Strict access controls are enforced to regulate system access and prevent unauthorized users from tampering with attendance records.

SYSTEM ARCHITECTURE:



PROJECT DESCRIPTION

1)Image Data Acquisition:

- When the capture button is clicked, a popup appears prompting the user to enter their name.
- The processed frame, containing the detected face, is then sent to the Flask server along with the associated name**.**

```
function captureImage(name) {
    const video = document.getElementById('video');
    const canvas = document.createElement('canvas');
    const context = canvas.getContext('2d');
    canvas.width = video.videoWidth;
    canvas.height = video.videoHeight;
    context.drawImage(video, 0, 0, canvas.width, canvas.height);
    const imageData = canvas.toDataURL('image/jpeg');

    // Send captured image data along with name to the server
    fetch('/capture_image', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            name: name,
            image_data: imageData
        })
    })
        .then(response => response.json())
        .then(data => {
            console.log('Image captured:', data.filename);
            // You can do something with the response if needed
        })
        .catch(error => {
            console.error('Error capturing image:', error);
        });
}
```

2)Face Detection:

- Upon receiving the image data and name, the server decodes the image and utilizes the Haar cascade classifier for face detection.
- If a face is detected, it is cropped from the image and stored, along with the associated name, for further use.

```python
@app.route('/capture_image', methods=['POST'])
def capture_image():
    # Get the image data from the request
    data = request.json
    name = data['name']
    image_data = data['image_data']

    # Decode base64-encoded image data
    image_bytes = base64.b64decode(image_data.split(',')[1])

    # Convert image bytes to OpenCV format
    nparr = np.frombuffer(image_bytes, np.uint8)
    img = cv2.imdecode(nparr, cv2.IMREAD_COLOR)

    # Perform face detection
    face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
    faces = face_cascade.detectMultiScale(img, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))
```

3)Face Encodings:

- The load_images function loads user images from a directory and extracts class names (filenames).
- It then generates face encodings for each image using the findEncodings function.
- findEncodings converts images to RGB format and encodes faces using the face_recognition library.
- The encoded faces are stored for recognition purposes.

```python
def findEncodings(images):
    encodeList = []
    for img in images:
        # Debugging: Print out the shape of the image
        print("Image shape:", img.shape if img is not None else "None")

        # Check if the image is empty or None
        if img is None:
            print("Error: Empty image encountered")
            continue  # Skip this image and proceed to the next one

        # Convert the color space if the image is not empty
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

        # Encode the face
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
    return encodeList
```

4)Face Recognition:

- To capture the attendance, the user clicks on the take_attendance button, the model then calculates the geolocation of the user using their IP address. Only if the user is within the mentioned radius of the mentioned location will the user be allowed to give their attendance.

- The recognize_faces method decodes the image data received from the client.

- It then pre-processes the image by resizing and converting it to RGB format.
- Using the face_recognition library, it detects faces in the image and calculates their encodings.
- These encodings are compared with the known encodings stored in encodeListKnown.
- If a match is found based on a predefined threshold, the recognized face is marked for attendance.

```
def recognize_faces(self, frame_data, threshold=0.6):
    global classNames, encodeListKnown

    # Decode the image data from the request
    image_bytes = base64.b64decode(frame_data)

    # Convert the image bytes to a NumPy array
    nparr = np.frombuffer(image_bytes, np.uint8)

    # Decode the image using OpenCV's imdecode function
    img = cv2.imdecode(nparr, cv2.IMREAD_COLOR)

    # Check if the input image is valid
    if img is None:
        print("Error: Invalid input image")
        return None
```

```
def calculate_distance(lat1, lon1, lat2, lon2):
    from math import radians, sin, cos, sqrt, atan2

    # Convert latitude and longitude from degrees to radians
    lat1 = radians(lat1)
    lon1 = radians(lon1)
    lat2 = radians(lat2)
    lon2 = radians(lon2)

    # Calculate the differences in latitude and longitude
    dlat = lat2 - lat1
    dlon = lon2 - lon1

    # Calculate the Haversine formula parameters
    a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon / 2)**2

    # Ensure that the value of 'a' is within the valid range [-1, 1]
    if a > 1:
        a = 1
    elif a < -1:
        a = -1

    # Calculate the distance using the Haversine formula
    c = 2 * atan2(sqrt(a), sqrt(1 - a))
    distance = 6371 * c * 1000  # Radius of the Earth in meters

    return distance
```

5) Attendance Marking:

- The model accesses the given path and records the attendance of the recognized person in the file. The attendance is stored in the format:" NAME-DATE-TIME"

```
class AttendanceSystem:
    def mark_attendance(self, name):
        with open(r"C:\Users\Administrator\Desktop\New Text Document (2).txt", 'a+') as f:
            myDataList = f.readlines()
            nameList = [entry.split(',')[0] for entry in myDataList]
            if name not in nameList:
                time_now = datetime.now()
                tString = time_now.strftime('%H:%M:%S')
                dString = time_now.strftime('%d/%m/%Y')
                f.write(f'\n{name},{tString},{dString}')
```

PROJECT DEMO:

STEP 1: Capture the face of the student.

STEP 2: The detected face is cropped and stored with the name of the student

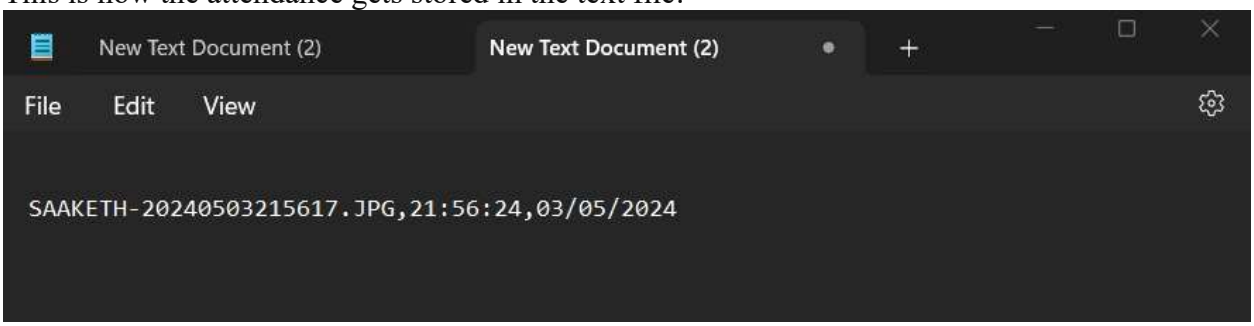
STEP 3: The unique features of the face are encoding into numerical values and stored

```
Encoding Complete
[array([-1.43842295e-01,  5.81688285e-02,  4.07476835e-02, -7.00023351e-03,
       -4.54129688e-02, -4.62599769e-02, -6.64031580e-02, -5.76800667e-02,
        2.04386726e-01, -1.15114629e-01,  1.21034101e-01, -3.38430963e-02,
       -2.27935120e-01, -6.18670089e-03, -5.25740683e-02,  1.66227579e-01,
       -1.67388499e-01, -1.42660767e-01, -6.61636218e-02, -5.74312918e-02,
        1.02826700e-01,  6.53017908e-02,  6.21671043e-02,  1.36926085e-01,
       -1.95829466e-01, -3.02572340e-01, -7.01400712e-02, -6.13782965e-02,
       -2.75934264e-02, -8.80390033e-02, -9.54251662e-02,  4.49476764e-02,
       -2.36476228e-01, -6.02768734e-02,  3.42245772e-02,  2.05898657e-01,
        5.91248535e-02, -4.26832624e-02,  1.18462026e-01, -7.43394718e-04,
       -2.52134502e-01, -1.16219155e-01,  1.29978210e-01,  2.48963252e-01,
        1.77989081e-01,  1.43590361e-01, -3.75710949e-02,  9.10409726e-04,
```

STEP 4: For recognition, the stored encodings are compared with the new calculated encoding and if they match and the student is within the given radius then the attendance is marked fro the specific student.

```
127.0.0.1 - - [18/Apr/2024 21:01:26] "POST /video_feed HTTP/1.1" 200 -
Local IP address: 10.151.149.67
[38.8462, -77.3064]
1546.824860407172
here
Attendance marked for: SAAKETH-20240418210006.JPG
```

This is how the attendance gets stored in the text file:



If the student image is not in the database, then the output message is as follows:

```
127.0.0.1 - - [03/May/2024 21:54:03] "POST /capture_image HTTP/1.1" 200 -
Local IP address: 192.168.56.1
[38.8462, -77.3064]
1492.2924674101494
here
Face not recognized
127.0.0.1 - - [03/May/2024 21:54:07] "POST /video_feed HTTP/1.1" 200 -
```

CHALLENGES FACED:

During the development of Facial GeoLock, several challenges were encountered, ranging from technical complexities to implementation hurdles. These challenges posed significant obstacles to the project team but were ultimately overcome through collaborative efforts and innovative solutions. Below are the key challenges faced during the development of Facial GeoLock:

1) **Facial Recognition Accuracy:**

   One of the primary challenges faced was achieving high accuracy in facial recognition. While facial recognition technology has advanced significantly in recent years, ensuring accurate detection and recognition of faces across various conditions (e.g., different lighting conditions, facial expressions, occlusions) proved to be a daunting task. The project team had to experiment with different algorithms, parameter settings, and preprocessing techniques to improve the accuracy of facial recognition in diverse environments.

2) **Geolocation Tracking Precision:**

   Another challenge was ensuring precision in geolocation tracking. Geolocation tracking relies on GPS coordinates or IP-based geolocation services to verify the physical location of users. However, factors such as signal interference, network latency, and inaccuracies in GPS data could affect the precision of location tracking. The project team had to implement robust error handling mechanisms and fine-tune the geolocation tracking algorithms to minimize errors and ensure accurate verification of user locations.

3) **Integration of Multi-Factor Authentication:**

   Integrating multi-factor authentication (MFA) into Facial GeoLock posed a significant integration challenge. MFA requires seamless coordination between facial recognition and geolocation verification components to ensure that users undergo dual authentication during the attendance marking process. The project team had to develop custom integration solutions and establish secure communication channels between the different authentication modules to facilitate smooth MFA workflows.

4) **Scalability and Performance Optimization:**

   Scalability and performance optimization were critical challenges, especially considering the potential deployment of Facial GeoLock in large-scale organizational environments. The system needed to accommodate a large number of users and locations while maintaining optimal performance and response times. The project team conducted extensive performance testing and optimization efforts to fine-tune the system architecture, database schema, and resource allocation to meet scalability requirements and ensure consistent performance under varying load conditions.

User Acceptance and Training:

User acceptance and training were challenges associated with the adoption of Facial GeoLock in organizational settings. Introducing a new attendance authentication system required user training and change management efforts to familiarize users with the system's features, interface, and authentication procedures. The project team collaborated with stakeholders to develop comprehensive training materials, conduct user workshops, and provide ongoing support to address user concerns and ensure a smooth transition to Facial GeoLock.

FUTURE ENHANCEMENTS:

While Facial GeoLock already offers robust functionality and security features, there are several areas for potential enhancement and development:

1) **Integration with Advanced Biometrics:**
   - Explore the integration of additional biometric modalities such as iris recognition or voice recognition to enhance the multi-factor authentication mechanism.
   - By incorporating multiple biometric traits, Facial GeoLock can further strengthen security and reduce the likelihood of unauthorized access.
2) **Enhanced Machine Learning Algorithms:**
   - Invest in research and development to improve the accuracy and efficiency of facial recognition algorithms.
   - Implement advanced machine learning techniques such as deep learning to enhance the system's ability to recognize faces in various lighting conditions and poses.
3) **Mobile Application Development:**
   - Develop a mobile application for Facial GeoLock to provide users with convenient access to attendance authentication features on their smartphones.
   - The mobile app could offer additional functionalities such as push notifications for attendance reminders and real-time access to attendance records.
4) **Geofencing Capabilities:**
   - Enhance the geolocation tracking system with geofencing capabilities to create virtual boundaries around designated locations.
   - Implement alerts or notifications to administrators when users attempt to mark attendance outside of predefined geofenced areas.
5) **Data Analytics and Reporting:**
   - Implement data analytics tools to analyze attendance patterns and trends over time.
   - Generate customizable reports and insights to help administrators make informed decisions about workforce management and resource allocation.
6) **Integration with Access Control Systems:**

- Integrate Facial GeoLock with existing access control systems to provide seamless access to facilities based on authenticated attendance.
- Enable automatic door access control based on successful attendance verification, enhancing security and convenience for users.

7) **Cloud-Based Deployment:**
   - Explore the migration of Facial GeoLock to a cloud-based deployment model to improve scalability and accessibility.
   - Cloud deployment would enable easy deployment across multiple locations and facilitate remote access for administrators.

By focusing on these future enhancements, Facial GeoLock can continue to evolve and adapt to the changing needs of organizations, further enhancing security, efficiency, and user experience in attendance management.

REFERENCES:

1) https://github.com/neha01/FaceRecognition/blob/master/faceRecognition.py
2) https://docs.opencv.org/4.x/da/d60/tutorial_face_main.html
3) https://www.analyticsvidhya.com/blog/2021/06/learn-how-to-implement-face-recognition-using-opencv-with-python/
4) https://geocoder.readthedocs.io/