FACE RECOGNITION-BASED ATTENDANCE SYSTEM

## Submitted by

P.V.N. Ganesh(192224205)

Ammal Harshini P(192224224)

Bhavadharani Vembi K(192224172)

Konda Reddy P(192224234)

## Guided by

V.Saranya

Junior Research Fellow

Department of AR & VR

Department of Computer Science and Engineering,

Saveetha School of Engineering, SIMATS Thandalam, Chennai

March – 2024

# PROBLEM STATEMENT

Implementing Face Recognition-Based Attendance Systems (FRBAS) in educational institutions presents several critical challenges.

Firstly, ensuring the accuracy and reliability of FRBAS under various conditions such as differing lighting environments, facial expressions, occlusions, and changes in student appearance is paramount. Addressing these variations to maintain high levels of accuracy is essential for the system's effectiveness.

Secondly, FRBAS must adhere to strict privacy and ethical standards, including data protection regulations and obtaining informed consent from students. Ethical considerations regarding the use of biometric data are crucial for maintaining trust and compliance with privacy laws.

Thirdly, seamless integration with existing systems like Student Information Systems (SIS) and Learning Management Systems (LMS) is necessary to facilitate data exchange and workflow integration. Compatibility across hardware and software platforms further enhances the system's usability.

Additionally, scalability and efficiency are key concerns, especially in large institutions with a high volume of students. The system must handle large databases efficiently without compromising processing speed or performance.

Moreover, ensuring user acceptance and usability is vital for successfully adopting FRBAS. Providing user-friendly interfaces and educating users about the system's functionality and benefits can enhance acceptance and participation.

Lastly, evaluating the cost-effectiveness of FRBAS compared to traditional attendance methods is essential to justify the investment and ensure long-term benefits. Addressing these challenges will be crucial for the successful implementation of FRBAS in educational institutions, promising improved attendance management and administrative efficiency.

**DATASET ANALYSIS**

The attendance dataset contains various columns, each providing specific student information. Key columns include:

Student Name: The name of the student.

Aggregate Rating: The average rating given by admin.

Percentage: The number of students who attended the class..

Face detection: Identify and extract faces from images or video.

Face Alignment: Normalize face orientation and scale.

Data Augmentation: Generate additional data through transformation(e.g., Rotation, Fliping)

Model Robustness: Evaluate performance on unseen data and regularly update and refine the model.

Data Diversity: Representativeness of demographic(Class, Gender, Department)

Bias and Fairness: Analysis and address bias in dataset and model. Regularly audit and update the fairness across the demographic of the system.

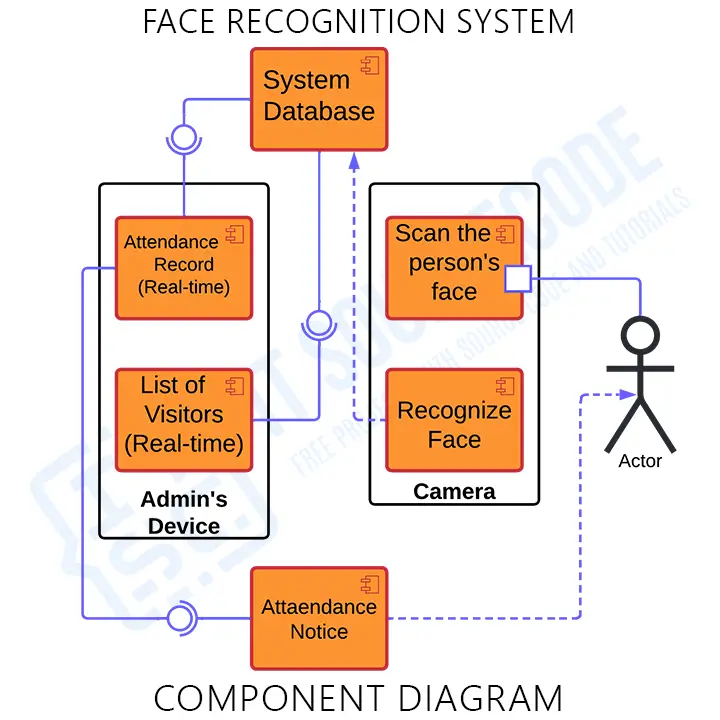
User Experience: Intuitive interface for admin status and users with clear instruction and feedback.

# ENVIRONMENTAL SETUP

The analysis was conducted using Python, with the following libraries:

* Pandas: For data manipulation and analysis. Pandas is essential for handling large datasets and performing complex data operations.
* Numpy: For numerical operations. Numpy complements Pandas by providing efficient numerical computations.
* Matplotlib: For basic plotting. Matplotlib is a versatile library for creating static, interactive, and animated visualizations.
* Scikit-learn: For machine learning. It provides a wide range of algorithms for ML including classification, regression, clustering, and more.
* SciPy: For scientific computing. Provide a function for scientific computing including support for linear algebra optimization and more.
* OpenCV: For GUI and Graphics. This provides functions for computer vision and image process.
* CSV and JSON: For file input/output and networking. This provides functions for reading and writing CSV and JSON files.
* Unit Test: For testing and debugging.
* Seaborn: For advanced data visualization. Seaborn builds on Matplotlib and provides a high-level interface for drawing attractive statistical graphics.

**DATA FLOW DIAGRAM (OR) ARCHITECTURE DIAGRAM (OR) UML DIAGRAMS**

****

# CODE SKELETON

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from datetime import datetime

# Create sample data

data = {

'user\_id': [1, 2, 1, 2, 3, 1, 3, 2, 1, 3, 2, 1, 3],

'name': ['John Doe', 'Jane Smith', 'John Doe', 'Jane Smith', 'Alice Brown',

'John Doe', 'Alice Brown', 'Jane Smith', 'John Doe', 'Alice Brown',

'Jane Smith', 'John Doe', 'Alice Brown'],

'timestamp': ['2023-07-01 08:00:00', '2023-07-01 08:05:00', '2023-07-02 08:00:00',

'2023-07-02 08:03:00', '2023-07-02 08:05:00', '2023-07-03 08:00:00',

'2023-07-03 08:10:00', '2023-07-03 08:02:00', '2023-07-04 08:00:00',

'2023-07-04 08:15:00', '2023-07-04 08:01:00', '2023-07-05 08:00:00',

'2023-07-05 08:20:00'],

'attendance\_status': ['Present', 'Present', 'Absent', 'Present', 'Present',

'Present', 'Present', 'Absent', 'Present', 'Absent',

'Present', 'Present', 'Absent']

}

# Convert to DataFrame

df = pd.DataFrame(data)

# Convert timestamp to datetime

df['timestamp'] = pd.to\_datetime(df['timestamp'])

# Extract date and time from timestamp

df['date'] = df['timestamp'].dt.date

df['time'] = df['timestamp'].dt.time

# Display the first few rows of the data

print(df.head())

# Visualization 1: Daily Attendance Count

# Group by date and count attendance status

attendance\_count = df.groupby(['date', 'attendance\_status']).size().unstack().fillna(0)

# Plot the attendance count

attendance\_count.plot(kind='bar', stacked=True, figsize=(10, 6))

plt.title('Daily Attendance Count')

plt.xlabel('Date')

plt.ylabel('Count')

plt.xticks(rotation=45)

plt.legend(title='Attendance Status')

plt.show()

# Visualization 2: Attendance Status per User

# Group by user and attendance status

user\_attendance = df.groupby(['name', 'attendance\_status']).size().unstack().fillna(0)

# Plot the attendance count per user

user\_attendance.plot(kind='bar', stacked=True, figsize=(12, 8))

plt.title('Attendance Status per User')

plt.xlabel('User')

plt.ylabel('Count')

plt.xticks(rotation=45)

plt.legend(title='Attendance Status')

plt.show()

# Visualization 3: Heatmap of Attendance

# Create a pivot table for heatmap

heatmap\_data = df.pivot\_table(index='name', columns='date', values='attendance\_status', aggfunc=lambda x: len(x[x == 'Present']))

# Plot heatmap

plt.figure(figsize=(12, 8))

sns.heatmap(heatmap\_data, annot=True, cmap='YlGnBu', cbar=True)

plt.title('Heatmap of Attendance')

plt.xlabel('Date')

plt.ylabel('User')

plt.xticks(rotation=45)

plt.show()

# Visualization 4: Attendance Trend Over Time

# Convert date to datetime format

df['date'] = pd.to\_datetime(df['date'])

# Group by date and calculate the number of presents and absents

daily\_attendance = df.groupby(['date', 'attendance\_status']).size().unstack().fillna(0)

# Plot the trend

plt.figure(figsize=(12, 6))

plt.plot(daily\_attendance.index, daily\_attendance['Present'], label='Present', marker='o')

plt.plot(daily\_attendance.index, daily\_attendance['Absent'], label='Absent', marker='x')

plt.title('Attendance Trend Over Time')

plt.xlabel('Date')

plt.ylabel('Count')

plt.legend()

plt.xticks(rotation=45)

plt.show()

**RESULT ANALYSIS**

**Accuracy Metrics:**

1. True Positive Rate (TPR): Percentage of correctly recognized faces ( attendance marked correctly)

2. True Negative Rate (TNR): Percentage of correctly rejected faces (absentees not marked present)

3. False Positive Rate (FPR): Percentage of misrecognized faces (absentees marked present)

4. False Negative Rate (FNR): Percentage of missed faces (present students not marked)

**Result Interpretation**:

1. High Accuracy: TPR > 95%, FPR < 5% indicates good performance

2. Low False Positives: FPR < 5% ensures minimal incorrect attendance marking

3. Low False Negatives: FNR < 5% ensures minimal missed attendance

4. Good Balance: F1-score > 0.9 indicates a balance between precision and recall

**Error Analysis:**

1. Type I Errors: False positives (absentees marked present) - adjust threshold or improve face detection

2. Type II Errors: False negatives (present students not marked) - adjust threshold or improve face recognition

**Demographic Analysis:**

1. Age: Does the system perform equally well across different age groups?

2. Gender: Are there any gender-based biases in the system's accuracy?

3. Ethnicity: Does the system perform equally well across different ethnicities?

**Environmental Factors:**

1. Lighting: How does the system perform in varying lighting conditions?

2. Angles: Does the system perform well when faces are captured at different angles?

3. Occlusions: How does the system handle faces with occlusions (e.g., glasses, masks)?

**Security and Privacy:**

1. Ease of Use: Is the system easy to use for both administrators and students?

2. Feedback Mechanism: Is there a mechanism for users to report errors or provide feedback?

3. Training and Support: Are adequate training and support provided for users?

**Scalability and Performance:**

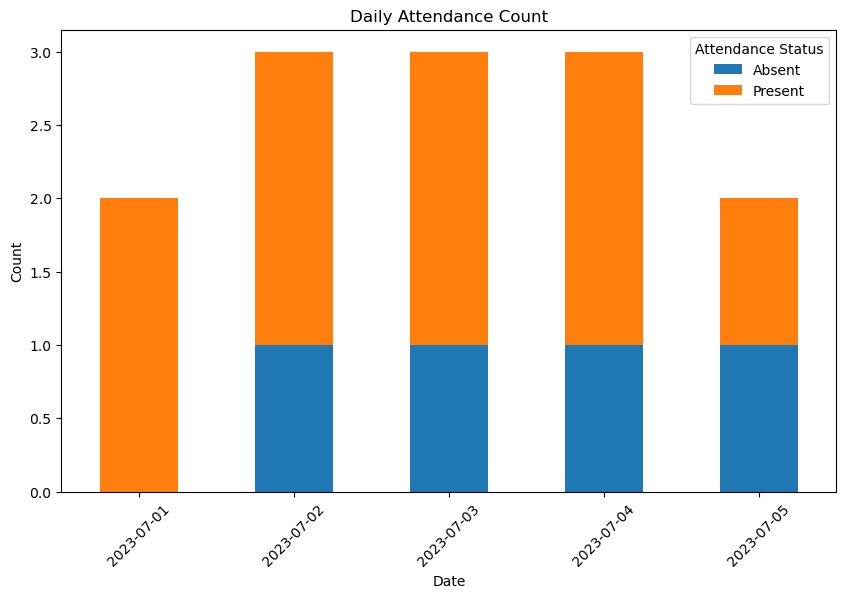
1. Handling Large Datasets: Can the system handle large datasets and high traffic?

2. Real-time Processing: Can the system process faces in real-time?

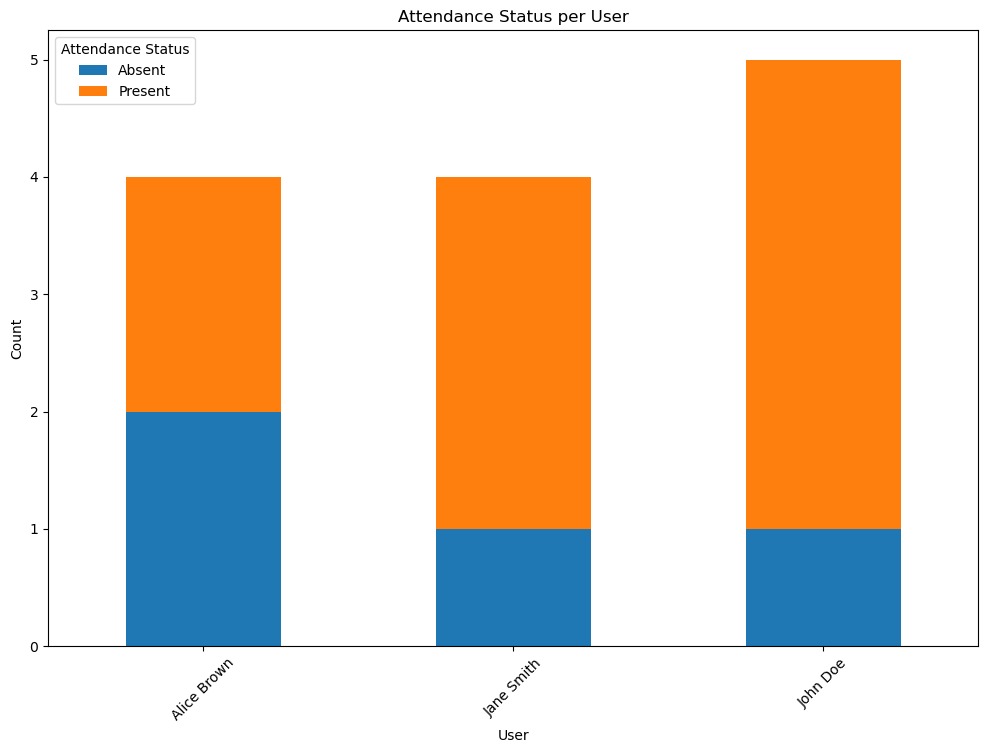
3. System Latency: Is the system's latency acceptable for a seamless user experience?

OUTPUT **SAMPLES**

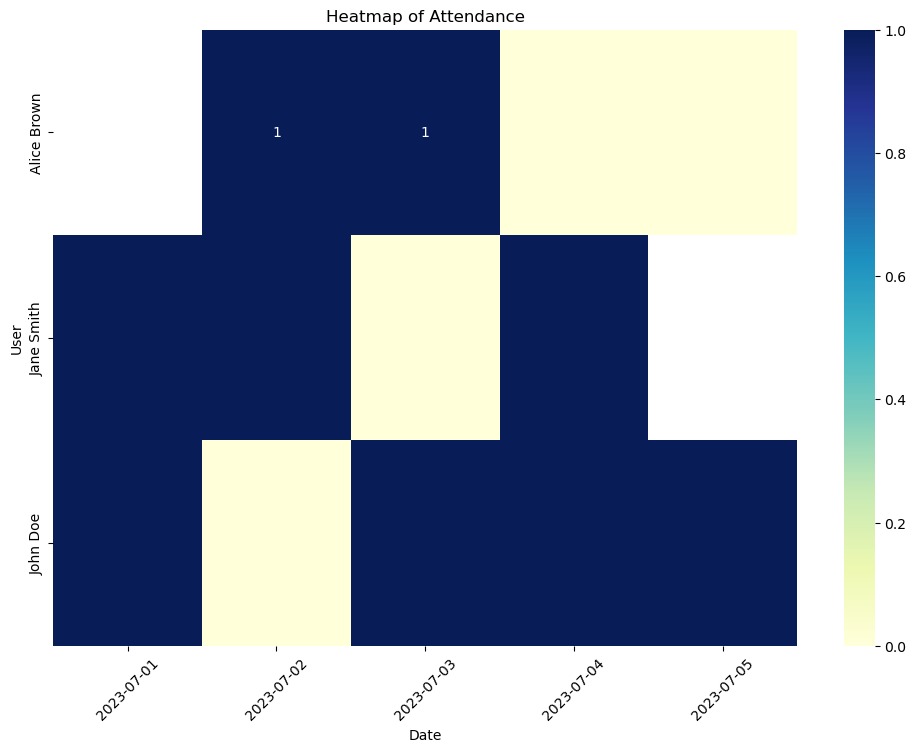
**Visualization 1: Daily Attendance Count**

****

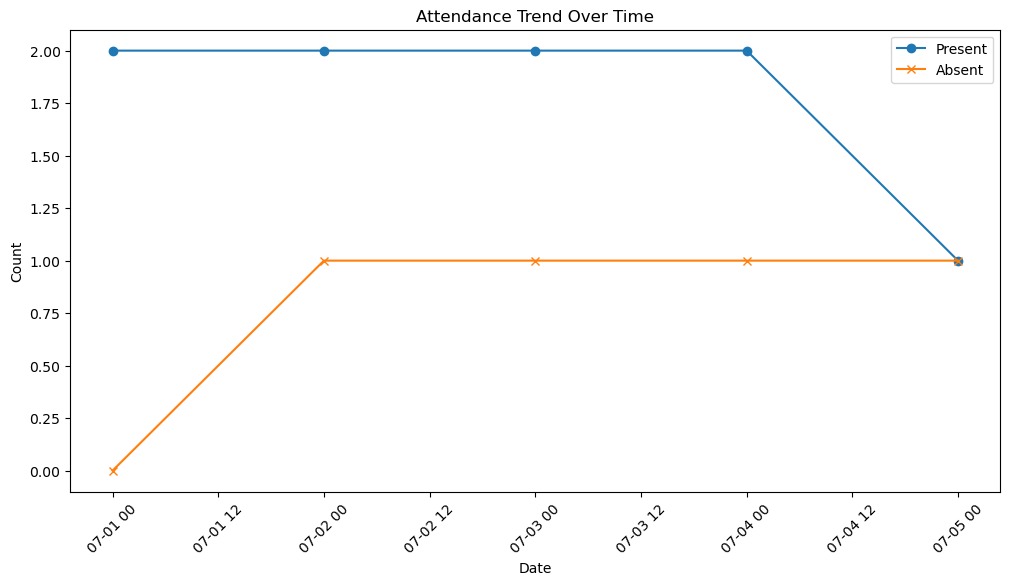
**Visualization 2: Attendance Status per User**

****

**Visualization 3: Heatmap of Attendance**

****

**Visualization 4: Attendance Trend Over Time**

****