Python Code

Face Recognition Attendance Marking Script

```
import cv2

import face_recognition

import pandas as pd

from datetime import datetime

import numpy as np

import time

import os


def mark_attendance(name):

    file_name = 'attendance.csv'

    if not os.path.exists(file_name):

        print("File does not exist. Creating new file.")

        df = pd.DataFrame(columns=['Name', 'Date', 'Time'])

    else:

        try:

            df = pd.read_csv(file_name)

            print("File loaded successfully.")

        except pd.errors.EmptyDataError:

            print("File is empty. Creating new DataFrame.")

            df = pd.DataFrame(columns=['Name', 'Date', 'Time'])


    now = datetime.now()

    date_string = now.strftime('%Y-%m-%d')
```

Python Code

```
    time_string = now.strftime('%H:%M:%S')

    if not ((df['Name'] == name) & (df['Date'] == date_string)).any():
        new_row = pd.DataFrame({'Name': [name], 'Date': [date_string], 'Time': [time_string]})
        df = pd.concat([df, new_row], ignore_index=True)
        df.to_csv(file_name, index=False)
        print(f"Attendance marked for {name} at {time_string}")
    else:
        print(f"{name} already marked for today.")

# List of known face encodings and names
known_face_encodings = []
known_face_names = []

# Paths to known face images
known_faces_paths = [
    r"C:\Users\shrav\OneDrive\Desktop\student1.jpg",
    r"C:\Users\shrav\OneDrive\Desktop\student2.jpg"
]

# Corresponding names for known faces
known_face_names = [
    "Ansh",
    "Darshan"
]
```

Python Code

```
# Load and encode known faces
for index, face_path in enumerate(known_faces_paths):
    try:
        image = face_recognition.load_image_file(face_path)
        image_encoding = face_recognition.face_encodings(image)[0]
        known_face_encodings.append(image_encoding)
        print(f"Loaded and encoded face for {known_face_names[index]}")
    except FileNotFoundError:
        print(f"Error loading {face_path}: File not found.")
    except IndexError:
        print(f"Error encoding face for {face_path}: No faces detected.")

# Initialize video capture
video_capture = cv2.VideoCapture(0)

# Time tracking for attendance marking
last_detection_time = time.time()
detection_interval = 10  # seconds
confidence_threshold = 0.6  # Adjust confidence threshold as needed

while True:
    ret, frame = video_capture.read()
    if not ret:
        print("Failed to grab frame")
        break


    rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)


    face_locations = face_recognition.face_locations(rgb_frame)
    print(f"Detected {len(face_locations)} faces")


    if face_locations:
        face_encodings = face_recognition.face_encodings(rgb_frame, face_locations)
        print(f"Detected {len(face_encodings)} face encodings")


        for face_encoding, face_location in zip(face_encodings, face_locations):
            matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
            face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
            best_match_index = np.argmin(face_distances)


            if face_distances[best_match_index] <= confidence_threshold:
                name = known_face_names[best_match_index]
            else:
                name = "Unknown"


            # Mark attendance only if it has been a while since the last detection
            if name != "Unknown" and (time.time() - last_detection_time) > detection_interval:
                mark_attendance(name)
                last_detection_time = time.time()
```

Python Code

```
            top, right, bottom, left = face_location
            cv2.rectangle(frame, (left, top), (right, bottom), (0, 255, 0), 2)
             cv2.putText(frame, name, (left + 6, bottom - 6), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,
255, 255), 1)
    else:
        print("No faces detected in the current frame.")

    cv2.imshow('Video', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

video_capture.release()
cv2.destroyAllWindows()
```