



FACULDADE IMPACTA DE TECNOLOGIA
CURSO TECNOLÓGICO DE BANCO DE DADOS

Marcelo Gomes de Sousa

Walysson de Brito Soares Ferreira

Maria Darlyanne Rangel de Barros Reis

Raylane do Nascimento Pereira

Felipe Julio do Val

RECONHECIMENTO FACIAL – FERRAMENTA ESCOLAR

SÃO PAULO

2022

Marcelo Gomes de Sousa	RA: 1600348
Walysson de Brito Soares Ferreira	RA: 1904508
Maria Darlyanne Rangel de Barros Reis	RA: 1905165
Raylane do Nascimento Pereira	RA: 1905164
Felipe Julio do Val	RA: 1904893

4A

RECONHECIMENTO FACIAL – FERRAMENTA ESCOLAR

SÃO PAULO

2022

1. ENTENDIMENTO DO NEGÓCIO

A voz é o principal meio de comunicação entre os seres humanos. Por meio dela as pessoas conseguem passar informações sobre sua história de vida, descrever seus pensamentos, seus sonhos, seus gostos e, principalmente, sua personalidade. Tendo isto em vista é evidente a importância da oratória para o mundo e a vida.

Todavia alguns profissionais estão perdendo essa habilidade por causa da exaustão vocal causada pelo uso excessivo dela e o aumento de volume. Um estudo foi realizado em 1989 por M. Calas que mostrou: “que 96% dos Professores entrevistados sofriam de fadiga vocal, 86% tinham lesões (frequentemente *nódulos*) e 85% usavam técnica vocal falha.”

Analisando informação acima, lembramos do profissional professor que perde tempo realizando diversas chamadas durante suas aulas, pensando em minimizar e automatizar esse problema em sala de aulas, desenvolvemos uma ferramenta para realizar a chamada por meio detecção facial.

A ferramenta detecta a face, envia para o banco de dados e para o e-mail do professor informando horário de chegada, turma e dados pessoais (nome, R.A.).

2. FERRAMENTA E TECNOLOGIAS

2.1. Biblioteca Dlib

A biblioteca Dlib realiza a detecção de faces, podendo ser utilizada no formato ao vivo.

A principal característica da biblioteca dlib é sua capacidade de identificar um ponto de referência (landmark).

2.2. Face Recognition

É uma API (Application Programming Interface) de reconhecimento facial, baseada no estado na biblioteca Dlib, desenvolvida em Python.

Realizando tarefas de localizar o rosto na imagem e tratar os frames para as etapas de reconhecimento.

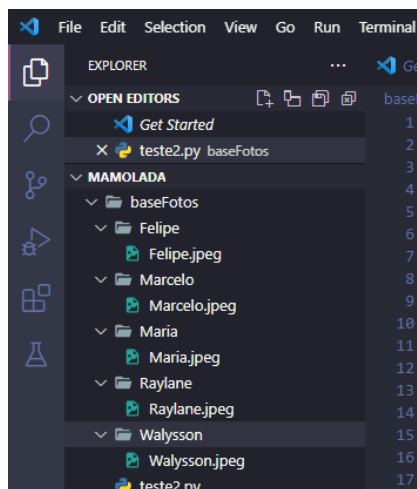
2.3. OpenCV

A biblioteca serve como infraestrutura para o desenvolvimento de aplicativos voltados a visão computacional, contendo funções de manipulação de imagens e vídeos, entre as funções encontradas na biblioteca, estão há as de conversão de imagens coloridas para a escala de cinza.

3. BASE DE DADOS

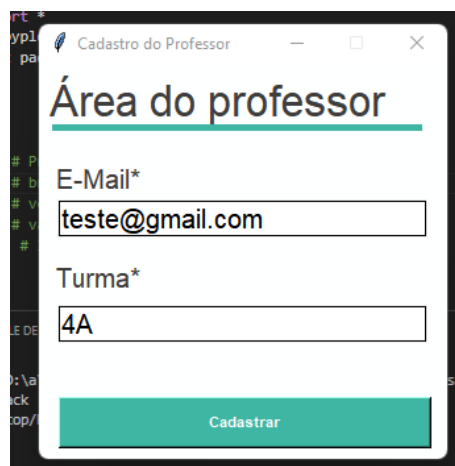
Foi anexado na entrega uma base de dados do grupo contendo apenas uma foto de cada, não realizamos treinamento, pois apenas com 1 foto o código obteve ótimo desempenho.

Realizamos testes com 20 pessoas e obtivemos sucesso no reconhecimento, mas não iremos anexar neste trabalho a imagem delas pois não temos autorização.



4. DESENVOLVIMENTO

Utilizando a biblioteca tkinter, criamos uma interface para cadastro, por trás da interface acontece uma validação de e-mail, caso o e-mail não seja válido o código é automaticamente interrompido.



5. PREPARAÇÃO DOS DADOS

Estamos realizando reconhecimento facial ao vivo, para isso foi necessário a criação de uma base com imagens de integrantes da equipe.

Criamos uma função chamada “detect_face” com o parâmetro “path_foto”, com o objetivo de reconhecer na imagem os rostos. Utilizamos da biblioteca Face Recognition os módulos load_image_file e face_encodings

- load_image_file: Delimita de rostos humanos em uma imagem
- face_encodings: Dada uma imagem, retorne à codificação daquela imagem.

Aplicamos uma lógica para retornar apenas se detectar uma face na foto.

```
def detect_face(path_foto):  
    foto = fr.load_image_file(path_foto)  
    rostos = fr.face_encodings(foto)  
    if (len(rostos)>0):  
        return True, rostos  
  
    return False, []
```

Utilizamos método get para acessar a codificação das imagens do dicionário, e armazenar nas variáveis “ra”, “rostos_conhecidos” e “nomes_dos_rostos”

```
def get_rostos():  
    rostos_conhecidos = []  
    nomes_dos_rostos = []  
    ra = []  
  
    maria = reconhece_face(r'.\baseFotos\Maria.jpeg')  
    if (maria[0]):  
        rostos_conhecidos.append(maria[1][0])  
        nomes_dos_rostos.append('Maria Darlyanne Rangel de Barros Reis')  
        ra.append(1905165)  
  
    Walysson = reconhece_face(r'.\baseFotos\Walysson.jpeg')  
    if (Walysson[0]):  
        rostos_conhecidos.append(Walysson[1][0])  
        nomes_dos_rostos.append('Walysson de Brito Soares Ferreira')  
        ra.append(1904508)
```

```

Felipe = reconhece_face(r'.\baseFotos\Felipe.jpeg')
if (Felipe[0]):
    rostos_conhecidos.append(Felipe[1][0])
    nomes_dos_rostos.append('Felipe Julio do Val')
    ra.append(1904893)

Marcelo = reconhece_face(r'.\baseFotos\Marcelo.jpeg')
if (Marcelo [0]):
    rostos_conhecidos.append(Marcelo[1][0])
    nomes_dos_rostos.append('Marcelo Gomes de Sousa')
    ra.append(1600348)

Raylane = reconhece_face(r'.\baseFotos\Raylane\Raylane.jpeg')
if (Raylane [0]):
    rostos_conhecidos.append(Raylane[1][0])
    nomes_dos_rostos.append('Raylane do Nascimento Pereira')
    ra.append(1905164)

nomes_dos_rostos.append('Desconhecido')
return      rostos_conhecidos, nomes_dos_rostos, ra

```

6. MODELAGEM

Utilizamos a biblioteca CV2 para abrir câmera e escolher qual das webcams atreladas na minha máquina iria ser utilizada.

Utilizamos a lógica para o código rodar enquanto a câmera estiver ligar e apenas encerrar quando apertamos a tecla 27 (Esc).

Dentro da condição utilizamos o modulo “read()”, que retorna 2 atributos, frame e se o quadro for lido corretamente True (ret).

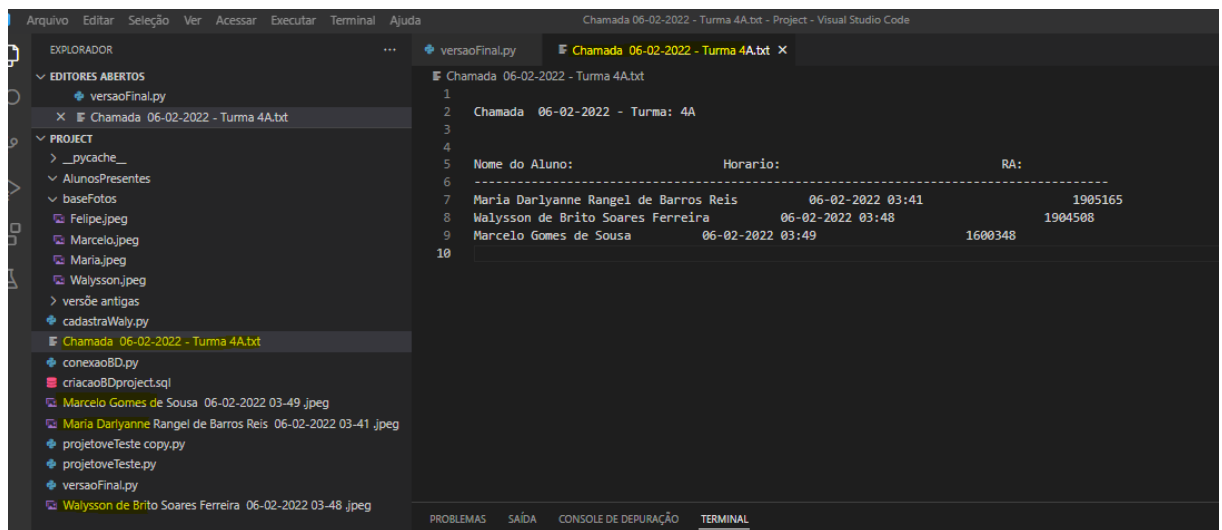
OpenCV cv2.VideoCapture.read() retorna o frame no formato BGR, então frame[:, :, ::-1] o converte no formato RGB que é o formato das imagens em nosso banco.

Após isso é utilizado logica para enquadrar as faces detectadas em live, utilizado o modulo “argmin” da biblioteca numpy que suporta o processamento dimensional para

identificar no atributo `face_distances` que está no formato `rgb` para localiza seu índice e atribuir na o nome da pessoa identificada na imagem.

Essa parte dodigo tem 2 principais atividades, que são:

- Quando a câmera é iniciada ela automaticamente captura uma imagem, salva a imagem com a nome, data e hora.
- Quando a câmera é finalizada ela automaticamente exporta um relatório de todos as faces detectadas



```
rostos_conhecidos, nome_dos rostos, ra1 = get_rostos()

lista_chegadas = []
relatorio = []
preseSalaAula = nome_dos rostos.copy()
video_capture = cv2.VideoCapture(0)

while video_capture.isOpened():
    ret, frame = video_capture.read()

    rgb_frame = frame[:, :, :-1]

    localizacao_dos rostos = fr.face_locations(rgb_frame)
    rosto_desconhecido = fr.face_encodings(rgb_frame, localizacao_dos rostos)

    for (top, right, botton, left), face_ecoloding in zip(localizacao_dos rostos, rosto_desconhecido):
        resultados = fr.compare_faces(rostos_conhecidos, face_ecoloding)
```



```

        face_distances = fr.face_distance(rostos_conhecidos, face_ecolo-
ding)

        melhor_id = np.argmin(face_distances)
        if resultados[melhor_id]:
            nome = nome_dos rostos[melhor_id]
            ra = ra1[melhor_id]

        else:
            nome = "Desconhecido"

        #print(nome)
        # Ao redor do rosto
        cv2.rectangle(frame, (left,top), (right,botton), (0,0,255),2)

        # #Emabaixo
        cv2.rectangle(frame, (left,botton-35), (right,botton), (0,0,255))
        font = cv2.FONT_HERSHEY_SIMPLEX

        # #texto
        cv2.putText(frame, nome, (left + 6, botton -6 ), font, 1.0,
(255,255,255), 1)

        cv2.imshow("WebCam_facerecognition", frame)

        lista = nome[0:]

        #print(ra)

        if lista != "Desconhecido" and len(preseSalaAula)>1:

            if lista in preseSalaAula:

                rel = datetime.now().strftime("%m-%d-%Y %H:%M")
                relatorio_now = [lista,rel,ra]
                relatorio.append(relatorio_now)

            def gerar_relatorio():
                print(f"""\nNome do Aluno:\t\tHorario: """)
                print("-----")
                for x in range(len(relatorio)):
                    print(f"""\t\t\t{relatorio[x][0][0]}\t\t\t{relato-
rio[x][1][1]}\t\t\t{relatorio[x][2][1]} """)

```

```

        lista_cheg = (datetime.now().strftime("%m-%d-%Y %H:%M"))
        lista_chegadas.append(lista_cheg)
        print(lista_chegadas)

        now = datetime.now().strftime("%m-%d-%Y %H:%M").re-
place(':', '-')
        teste = str(now)
        x = (nome[0:]+ ' ' + teste+ ' '+ '.jpeg')
        preseSalaAula.remove(lista)
        cv2.imwrite(x, frame)

    if cv2.waitKey(5) == 27:

        dataChamada = datetime.now().strftime("%m-%d-%Y")
        arquivo = open(f'Chamada {dataChamada} - Turma {turmaCha-
mada}.txt', 'w')
        arquivo.write(f"\nChamada {dataChamada} - Turma: {turmaCha-
mada}\n")
        arquivo.write("\n\n")
        arquivo.write("Nome do Aluno:                               Hora-
rio:                               RA:\n")
        arquivo.write("-----\n")
        for i in range(len(relatorio)):
            arquivo.writelines(str(relatorio[i][0] )+" " +str(re-
latorio[i][1] )+" " +str(relatorio[i][2] )+'\n')

        arquivo.close()

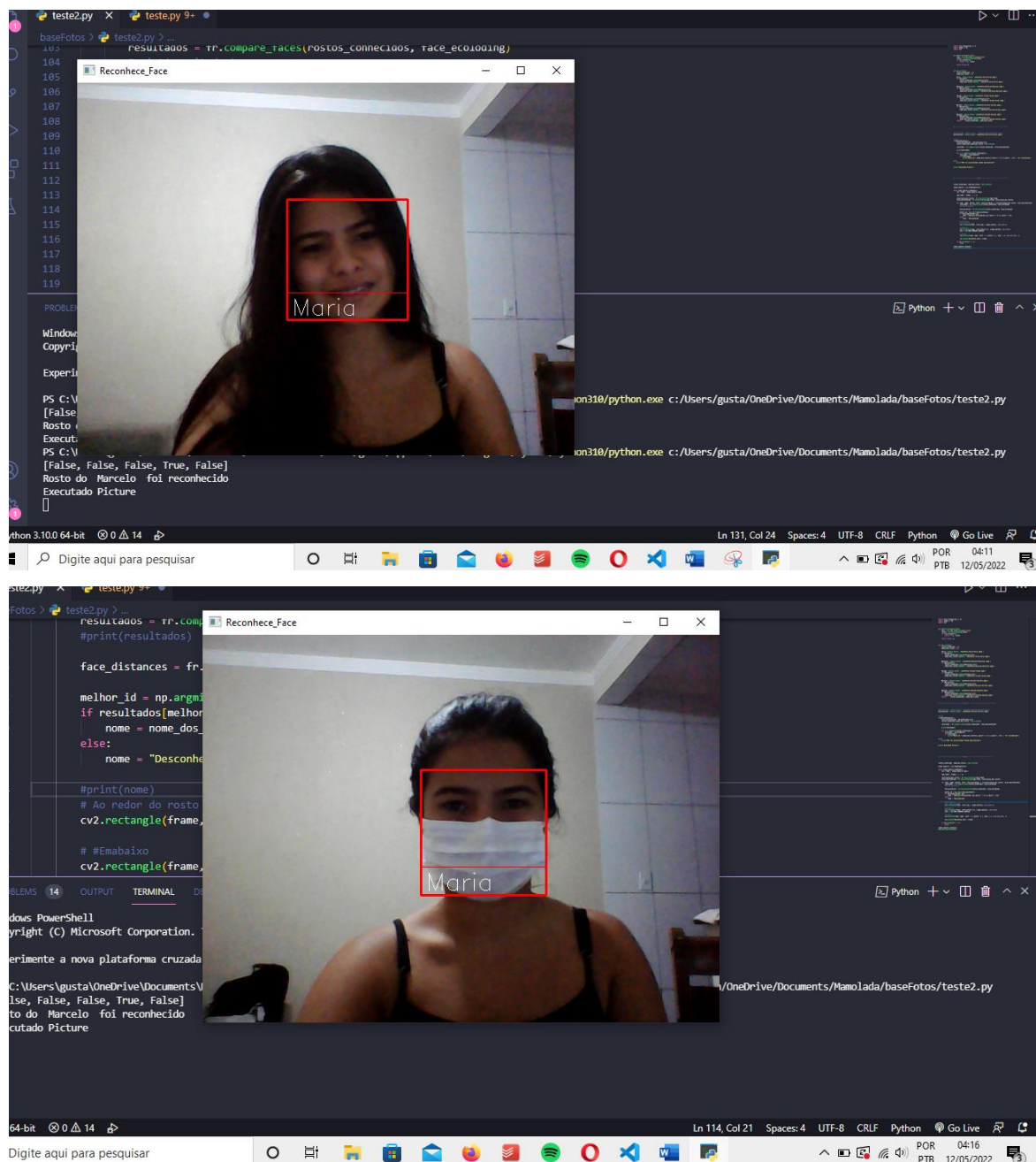
        break

video_capture.release()
cv2.destroyAllWindows()

```

7. EVIDÊNCIA

Abaixo esta demonstração da detecção, estando de perfil ou com máscara.

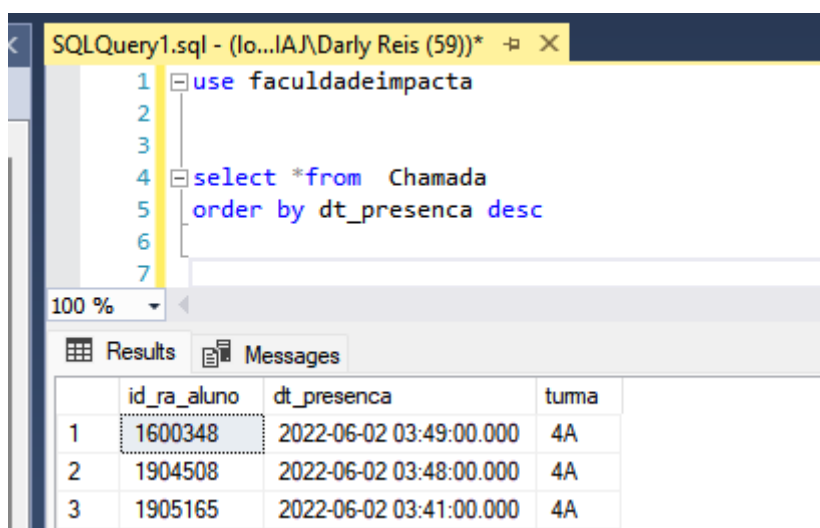


8. CONEXÃO COM O BANCO DE DADOS

Com o intuito de simular uma situação em uma instituição, realizamos conexão com banco de dados local, por meio da biblioteca pyodbc.

Todos os alunos que foram identificados, são enviados automaticamente para o banco de dados local.

O arquivo para a criação do banco de dados está em anexo da atividade.



The screenshot shows a SQL query window titled 'SQLQuery1.sql - (lo...IAJ\Darly Reis (59))'. The query is as follows:

```
1 use faculdadeimpacta
2
3
4 select *from Chamada
5 order by dt_presenca desc
6
7
```

Below the query window, the 'Results' tab is active, displaying a table with the following data:

	id_ra_aluno	dt_presenca	turma
1	1600348	2022-06-02 03:49:00.000	4A
2	1904508	2022-06-02 03:48:00.000	4A
3	1905165	2022-06-02 03:41:00.000	4A

```
#----- Conexão com o Banco -----
import pyodbc
def retornar_conexao_sql():
    server = "LAPTOP-SV6AVIAJ"
    database = "FACULDADEIMPACTA"
    string_conexao = 'Driver={SQL Server Native Client 11.0};Server='+server+';Database='+database+';Trusted_Connection=yes;'
    conexao = pyodbc.connect(string_conexao)
    return conexao.cursor()

for x in range(len(relatorio)):
    id_ra_aluno = str(relatorio[x][2])
    dt_presenca = str(lista_chegadas[x])
    turmapre = turmaChamada

    cursor = retornar_conexao_sql()
    comando = (f""" set dateformat ymd
```

```

        insert into Chamada values({id_ra_aluno},
cast('{dt_presenca}' as datetime) , '{turmapre}' ) "")
        cursor.execute(comando)
        cursor.commit()

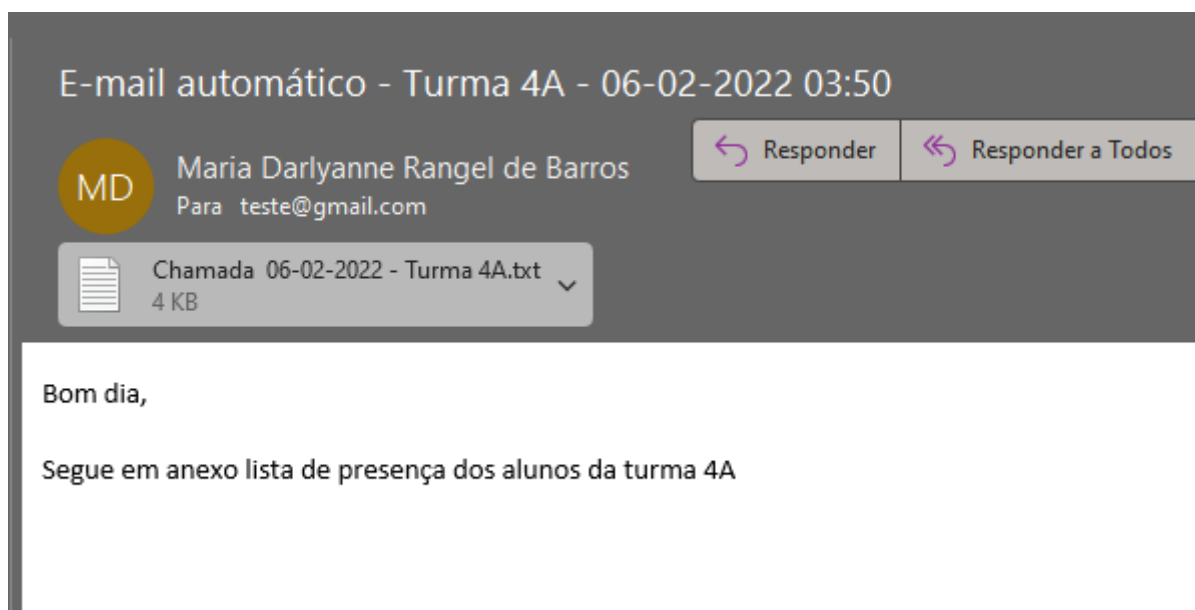
        #cursor.execute(""" SELECT * FROM alunos  """)
        #row = cursor.fetchall()
        #for x in row:
        #    print(x)

print("Salvo dados no banco")

```

9. ENVIO DE EVIENCIA PARA O E-MAIL

Como forma de auditar informações além de envio das informações para o banco de dados é disparo um e-mail para o professor com a lista de chamada em anexo.



```

#----- Enviar email -----

if turmaChamada != '' and emailPrestador != '':

    # criar a integração com o outlook
    outlook = win32.Dispatch('outlook.application')

```

```
# criar um email
email = outlook.CreateItem(0)

DataHoraChamada = datetime.now().strftime("%m-%d-%Y %H:%M")

# configurar as informações do seu e-mail
email.To = f"{emailPrestador}"
#email.To = "maria.reis@aluno.faculdadeimpacta.com.br"
email.Subject = f"E-mail automático - Turma {turmaChamada} - {DataHoraChamada}"
email.HTMLBody = f""""Bom dia,<br><br>Segue em anexo lista de presença dos alunos da turma {turmaChamada}""""

dataChamada = datetime.now().strftime("%m-%d-%Y")

anexo = ('C:/Users/Darly Reis/Desktop/Project/Chamada ' + dataChamada + ' - Turma ' + turmaChamada + '.txt')
print(anexo)
email.Attachments.Add(anexo)

email.Send()
print("Email Enviado")
```

10. AVALIAÇÃO

Nos testes realizados quando à uma ótima iluminação tivemos acerto de 100%, mas em momentos em que há iluminação considerada boa de 20 pessoas ele errou uma média de 2 pessoas e em baixa iluminação não reconhece