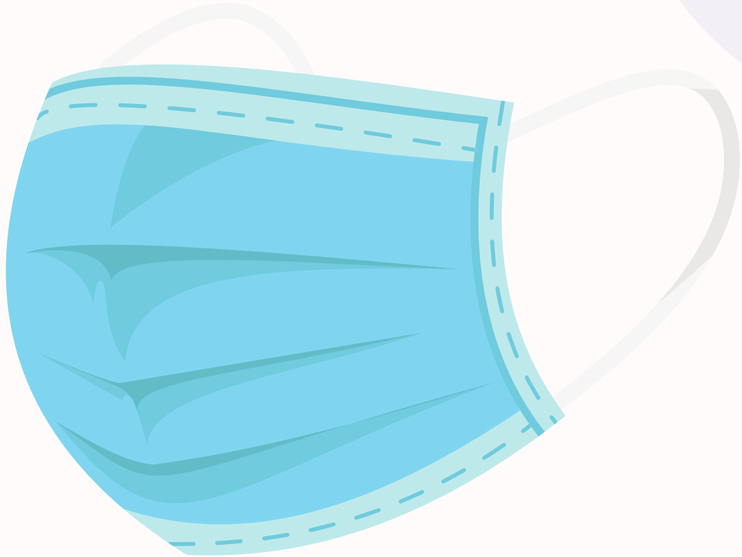
PORT'ON MASK



(french pun intended)

By Dylan Dray

TABLE OF CONTENT

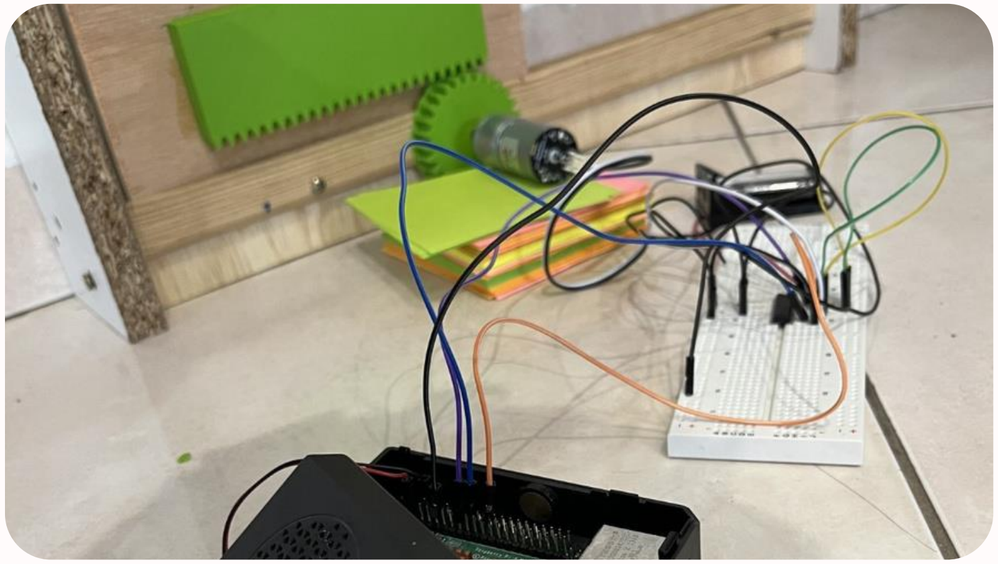
CONTEXT

CLASSES PRÉPARATOIRES AUX GRANDES ÉCOLES (CPGE)

HÔPITAL SAINT-JOSEPH
Marseille, France

SUBJECT OF STUDY: HEALTH AND PREVENTION

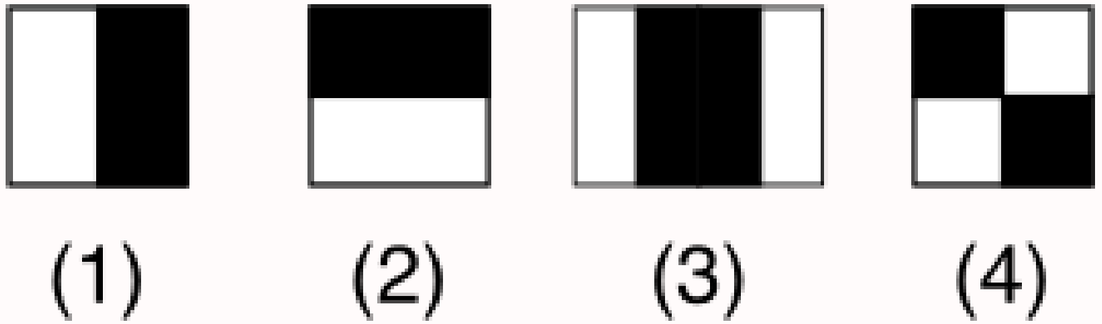
V1-MASK DETECTION &DOOR OPENING



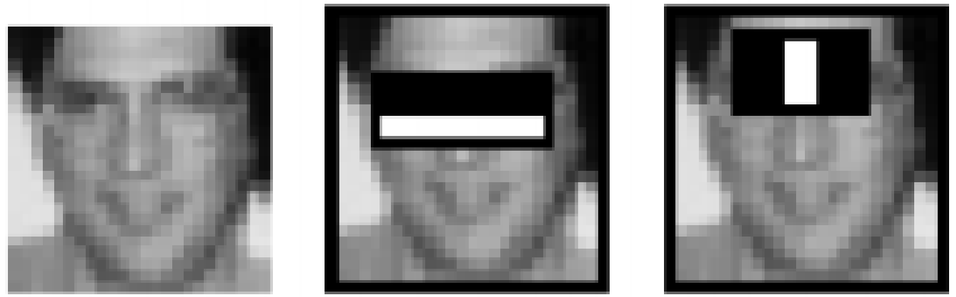
Raspberry PI (4B) with its Picamera
Electronic circuit controlling the
motor, which turns a pinion in a rack
in order to open the door.

Me sitting proudly next to
the Port'on Mask prototype

HAAR-LIKE FEATURES
VIOLA AND JONES ALGORITHMS



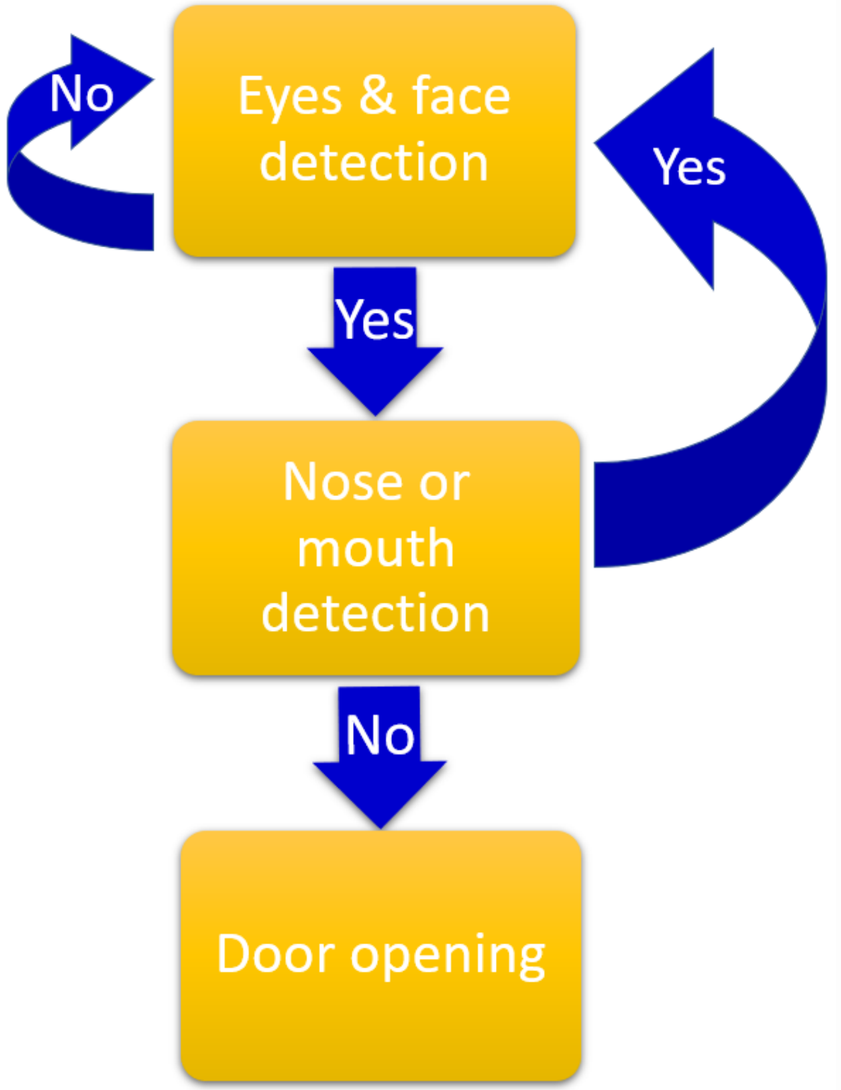
HAAR FEATURES



$$f(x,y) = \sum_i p_b(i) - \sum_i p_w(i)$$

The more f(x,y) is close to 1, the more it is accurate

FLOWCHART


No
Eyes & face detection
Yes
Yes
Nose or mouth detection
No
Door opening

SOME STATISTICS

Success rates	Near (70cm)	Far (120cm)
Mask worn	84%	68%
Mask not well-worn / not worn	86%	88%

by considering the indecision as mask badly worn








SOURCES OF ERRORS

Sunglasses

Distinction
Hand/Mask

Bangs

Backlight
Luminosity

STEP 2

IMPROVEMENTS

"CREATE A TOOL THAT SOLVES A PROBLEM THAT YOU'VE NOTICED IN YOUR LIFE"

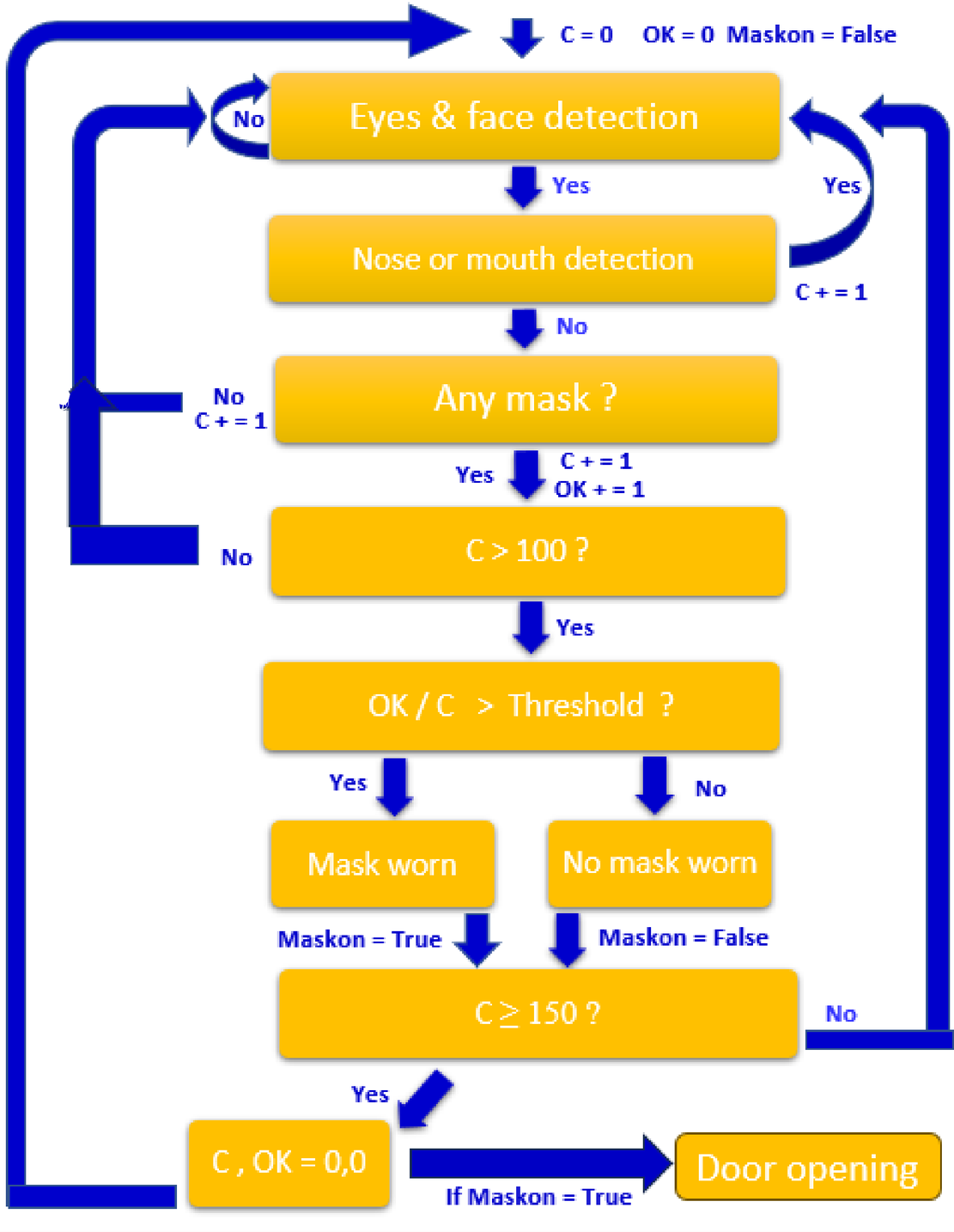
V2 IMPROVEMENTS

IMPROVED MASK DETECTION SYSTEM

FACE RECOGNITION WITH MASK ON

DATABASE SYSTEM

FLOWCHART

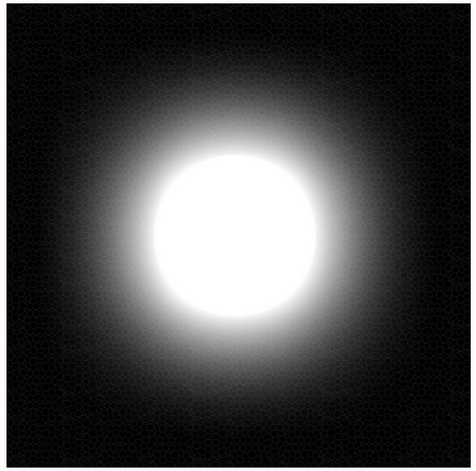

C = 0 OK = 0 Maskon = False
Eyes & face detection
No
Yes
Nose or mouth detection
Yes
C + = 1
No
Any mask ?
No
C + = 1
Yes
C + = 1
OK + = 1
C > 100 ?
No
Yes
OK / C > Threshold ?
Yes
No
Mask worn
No mask worn
Maskon = True
Maskon = False
C ≥ 150 ?
No
Yes
C , OK = 0,0
If Maskon = True
Door opening

MASK DETECTION SYSTEM HYPOTHESIS

Face must always be with the same angle

Must always stand at the same predefined distance

Does not wear an original mask



Always the same luminosity

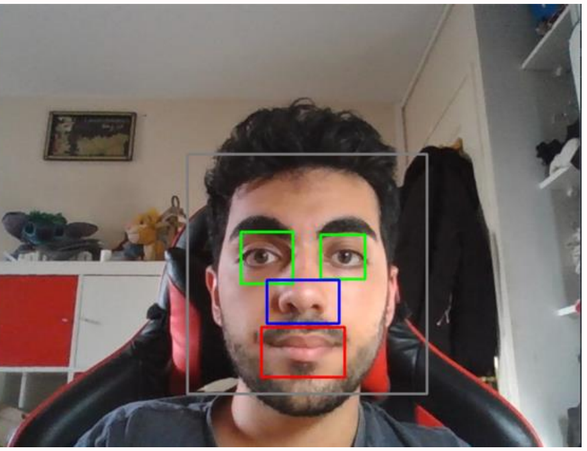
MASK DETECTION SYSTEM IMPROVEMENTS

- NOW CAPABLE OF DIFFERENTIATING RANDOM OBSTACLES (SUCH AS HAND IN FRONT OF THE MOUTH) FROM MASKS
- NOW CAPABLE TO DETERMINE WITH ACCURACY EYES, EVEN WITH GLASSES (NOT TINTED)

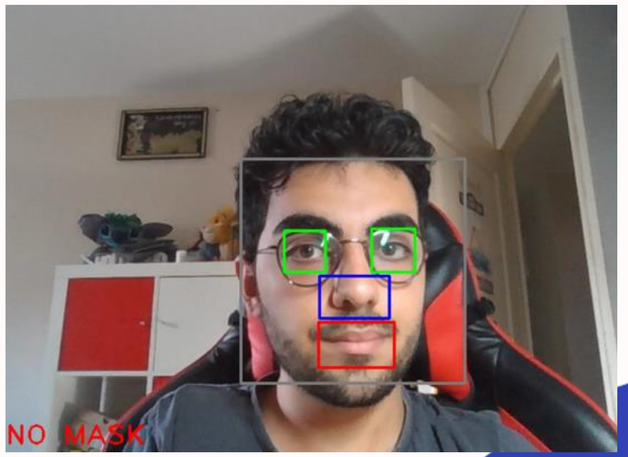
AND WITH THE PREVIOUS HYPOTHESIS:

- ACCURACY OF FACE/EYES/NOSE/MOUTH RECOGNITION INCREASED → ACCURACY OF MASK DETECTION INCREASED
- NO MORE ISSUES WITH LUMINOSITY

WITHOUT MASK WITHOUT GLASSES



WITHOUT MASK WITH GLASSES



MASK WITHOUT GLASSES

MASK WITH GLASSES

DATABASE 1: IMAGES

ZOOM ON THE FACE

PLACE NODAL POINTS ON EYEBROWS/EYES

CALCULATIONS FROM NODAL POINTS

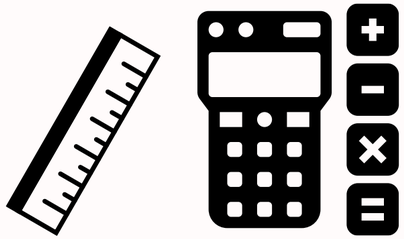
Valuable data

400 unprocessed photos

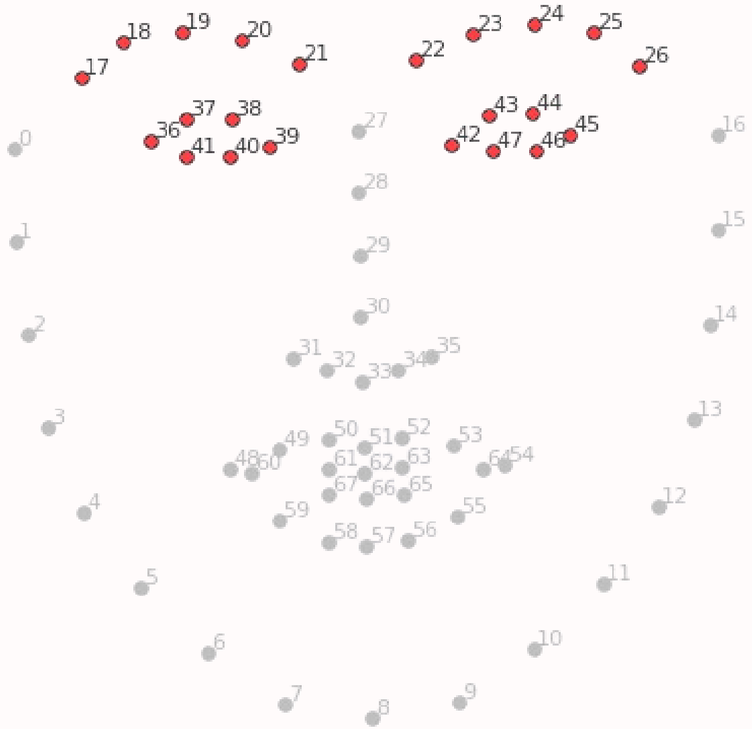
PROCESS_IMAGESBDDPHOTO( )

~130 processed photos

DATABASE 1:
IMAGES - VALUABLE DATA



MEASURES, RATIOS BETWEEN POINTS POSITION



DLIB PRETRAINED MODEL: 22 NODAL POINTS FOR EYEBROWS AND EYES

[LEFT EYEBROW CURVE, RIGHT EYEBROW CURVE]

DATABASE 2: MYSQL

```sql
DROP TABLE IF EXISTS Patient;
CREATE TABLE Patient (
    FileVisage VARCHAR(500) PRIMARY KEY,
    Nom VARCHAR(50),
    Prenom VARCHAR(50),
    Ratios VARCHAR(1000)
);
```

CREATEBDD( )

```sql
SELECT * FROM Patient;
SELECT * FROM Rdv ORDER BY Daterdv, Heure;
SELECT * FROM Docteur;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

FileVisage	Nom	Prenom	Ratios
C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	Lastname0	Firstname0	0.6818857329980162,0.6434382168114536,1....
C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	Lastname1	Firstname1	0.6818823239017383,0.6880099303306433,1....
C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	Lastname10	Firstname10	0.6624390150144968,0.6386159585991316,1....
C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	Lastname100	Firstname100	0.6814784012629456,0.6801716341904412,1....

NewPatient()

```sql
DROP TABLE IF EXISTS Docteur;
CREATE TABLE Docteur (
    ID VARCHAR(50) PRIMARY KEY,
    Nom VARCHAR(50),
    Prenom VARCHAR(50),
    Fonction VARCHAR(100)
);
```

CREATEBDD( )

```sql
36  SELECT * FROM Docteur;
```

Result Grid | Filter Rows: | Exp

ID	Nom	Prenom	Fonction
1001	Nom1	Prenom1	Radiologist
1002	Nom2	Prenom2	Endocrynologist
1003	Nom3	Prenom3	Radiologist
1004	Nom4	Prenom4	Cardiologist
1005	Nom5	Prenom5	General Practioner
1006	Nom6	Prenom6	Cardiologist

```sql
DROP TABLE IF EXISTS Rdv;
CREATE TABLE Rdv (
    Id VARCHAR(50) PRIMARY KEY,
    Emplacement VARCHAR(50),
    Daterdv DATE,
    Patient_FileVisage VARCHAR(500),
    Heure VARCHAR(10),
    ID_Doctor VARCHAR(50),
    FOREIGN KEY (ID_Doctor) REFERENCES Docteur (ID),
    FOREIGN KEY (Patient_FileVisage) REFERENCES Patient (FileVisage)
);
```

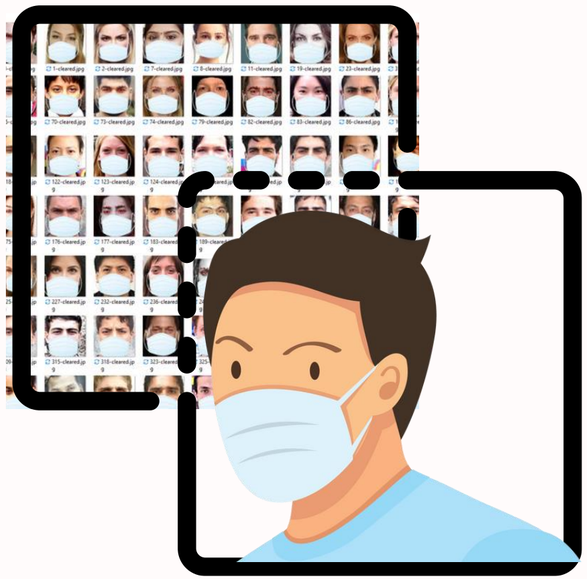
CREATEBDD( )

```sql
35  SELECT * FROM Rdv ORDER BY Daterdv, Heure;
36  SELECT * FROM Docteur;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	Emplacement	Daterdv	Patient_FileVisage	Heure
08:34:008342388600	Floor 1, Room 11,Left elevator	2023-06-24	C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	08:34:00
09:11:00911966774	Floor 6, Room 18,Right elevator	2023-06-24	C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	09:11:00
09:58:009581087284	Floor 4, Room 22,Left elevator	2023-06-24	C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	09:58:00
10:11:0010111745409	Floor 4, Room 1,Right elevator	2023-06-24	C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	10:11:00
10:22:0010222060371	Floor 8, Room 29,Left elevator	2023-06-24	C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	10:22:00
10:33:0010331125938	Floor 1, Room 4,Left elevator	2023-06-24	C:\Users\dylan\OneDrive\Bureau\ProjetMask\B...	10:33:00

FACE RECOGNITION: COMPARISONS

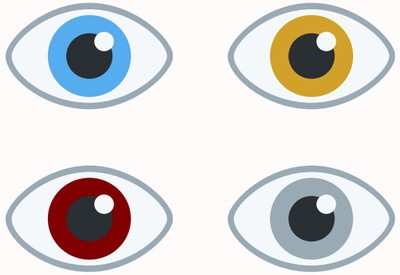


comparison of eyebrow curves

Top50

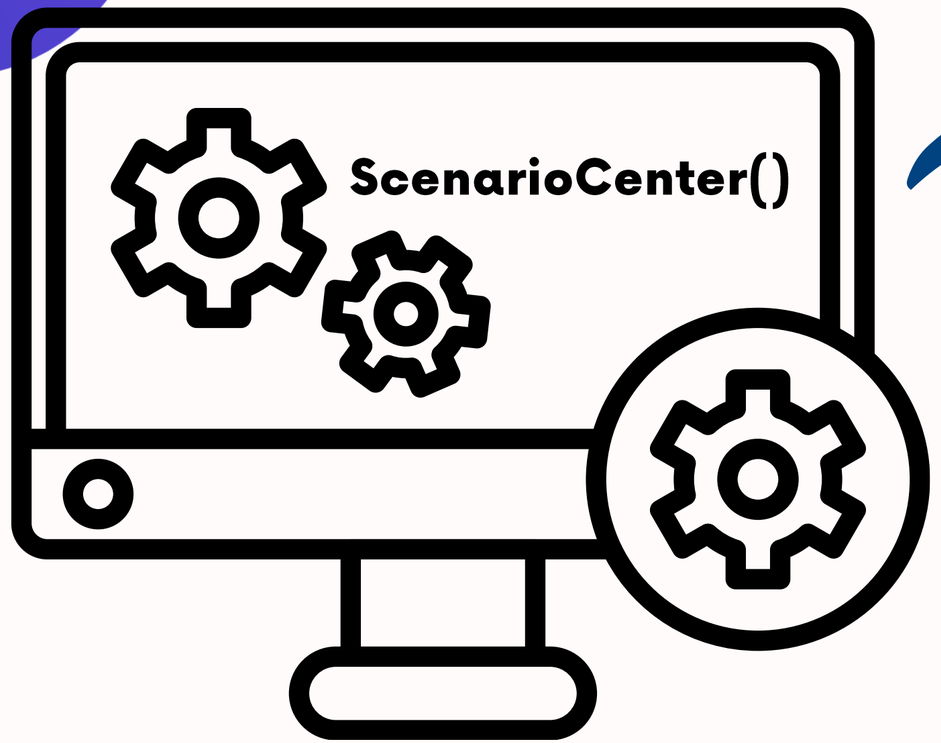
comparison of differences between characteristic distance ratios

Top20

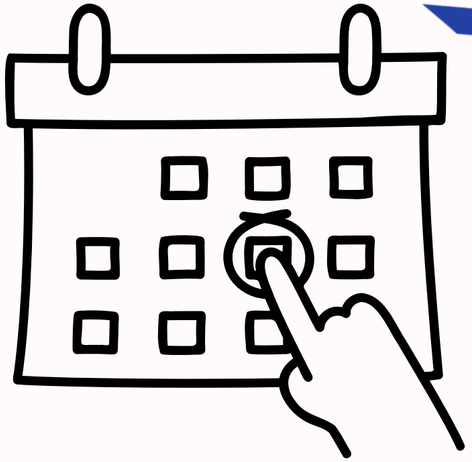


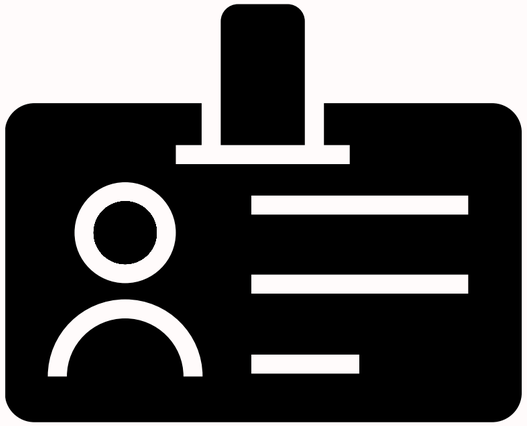
comparison of eye colors

Top5



Top5





SCENARIO

HEALTH CENTER CASE

THERE IS A VIDEO ABOUT THE APPLICATION CASE ON MY WEBSITE !

THERE IS A VIDEO ABOUT THE HEALTH CENTER CASE ON MY WEBSITE !

```python
# Haar Classifiers
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
mouth_cascade = cv2.CascadeClassifier('haarcascade_mouth.xml')
nose_cascade = cv2.CascadeClassifier('haarcascade_nose.xml')

threshold = 0.45  # threshold for moving average

c = 0
ok = 0
cap = cv2.VideoCapture(0)
mouths=[]
noses=[]
maskon = False
while True:
    ret, img = cap.read()
    img_nude = img.copy()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)  #for the masks
    normalized_gray = cv2.equalizeHist(gray)

    # Define color ranges for blue, black or white surgical masks
    lower_blue = np.array([110, 50, 50])
    upper_blue = np.array([130, 255, 255])
    lower_black = np.array([0, 0, 0])
    upper_black = np.array([180, 255, 30])
    lower_white = np.array([0, 0, 200])
    upper_white = np.array([180, 255, 255])
    mask = cv2.inRange(hsv, lower_blue, upper_blue) + cv2.inRange(hsv, lower_black, upper_black) + cv2.inRange(hsv, lower_white, upper_white)

    faces = face_cascade.detectMultiScale(normalized_gray, 1.1, 15)
    eyes = eye_cascade.detectMultiScale(normalized_gray, 1.2, 15)

    for (ex, ey, ew, eh) in eyes:
        cv2.rectangle(img, (ex, ey), (ex + ew, ey + eh), (0, 255, 0), 2)

    for (x, y, w, h) in faces:
```

```
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x + w, y + h), (125, 125, 125), 2)
        roi_gray = normalized_gray[y:y + h, x:x + w]
        roi_color = img[y:y + h, x:x + w]
        roi_mask = mask[y:y + h, x:x + w]
        has_maskon = cv2.countNonZero(roi_mask) > w * h * 0.235
        has_no_nose = len(nose_cascade.detectMultiScale(roi_gray, 1.1, 35)) == 0
        has_no_mouth = len(mouth_cascade.detectMultiScale(roi_gray, 1.3, 40)) == 0
        has_eyes = len(eyes) != 0

        if has_maskon and has_no_nose and has_no_mouth and has_eyes:
            ok += 1
        mouths = mouth_cascade.detectMultiScale(roi_gray, 1.3, 40)

        for (sx, sy, sw, sh) in mouths:
            cv2.rectangle(roi_color, (sx, sy), (sx + sw, sy + sh), (0, 0, 255), 2)

        noses = nose_cascade.detectMultiScale(roi_gray, 1.1, 35)
        for (nx, ny, nw, nh) in noses:
            cv2.rectangle(roi_color, (nx, ny), (nx + nw, ny + nh), (255, 0, 0), 2)


    c+=1

    if c >= 100:
        (height, width) = img.shape[:2]

        if ok / c > threshold:
            cv2.putText(img, "MASK OK", (10, height - 10), cv2.FONT_HERSHEY_PLAIN, 2, (0, 255, 0), 2)
            cv2.imwrite(path_newsave, img_nude)
            maskon = True
        else:
            cv2.putText(img, "NO MASK", (10, height - 10), cv2.FONT_HERSHEY_PLAIN, 2, (0, 0, 255), 2)

    if c>=150:
        c,ok=0,0

    cv2.imshow('img', img)
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break
    print(c,ok)

cap.release()
cv2.destroyAllWindows()
```

```python
#%% facial recognition while masked
identity= 0
def calculate_ratio(value1, value2):
    if value2 != 0:
        return value1 / value2
    else:
        return 0

def calculate_distance(point1, point2):
    return distance.euclidean((point1.x, point1.y), (point2.x, point2.y))

def get_filevisage_by_ratios(ratios):
    conn = mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="root",
        database="donneespatient")
    cursor = conn.cursor()
    query = f"SELECT FileVisage FROM Patient WHERE Ratios IN ({ratios});"
    cursor.execute(query)
    results = cursor.fetchall()
    file_visage_list = []

    for row in results:
        file_visage = row[0]
        file_visage_list.append(file_visage)

    cursor.close()
    conn.close()

    return file_visage_list
```

```
    return file_visage_list

def get_name_by_filevisage(filevisage):
    conn = mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="root",
        database="donneespatient")
    cursor = conn.cursor()
    new_filevisage =filevisage.replace('\\', '\\\\')
    query = f"SELECT Nom, Prenom FROM Patient WHERE FileVisage IN ({new_filevisage});"
    cursor.execute(query)
    results = cursor.fetchall()
    identity_list = []

    for row in results:
        nom, prenom = row[0], row[1]
        identity_list.append([nom, prenom])

    cursor.close()
    conn.close()

    return identity_list

def get_filevisage_by_name(lname, fname):
    conn = mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="root",
        database="donneespatient")
    cursor = conn.cursor()
    query = f"SELECT FileVisage FROM Patient WHERE Nom IN ('{lname}') AND Prenom IN ('{fname}');"
    cursor.execute(query)
    results = cursor.fetchall()
    return results[0]
    cursor.close()
    conn.close()
```

```python
def process_imagesbddphoto():    #This function takes the entire image database, places points on the photos, calculates the ratios and returns the photo, ratios(distances and ratios) and curves
    ratios = []
    for k in range(1010):
        image_path = base_image_path + str(k) + "-with-mask.jpg"
        if os.path.isfile(image_path):
            img = cv2.imread(image_path)
            img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = face_cascade.detectMultiScale(img_gray, 1.3, 5)
            roi_hsv = None
            for (x, y, w, h) in faces:
                roi_hsv = img_gray[y:y + h, x:x + w]
                face = dlib.rectangle(x, y, x+w, y+h)
                shape = predictor(img, face)
                ratioint = []
                left_eyebrow_points= []
                right_eyebrow_points= []
                curves = []
                if roi_hsv is not None:
                    for j in range(22):
                        point = shape.part(j)
                        cv2.circle(img, (point.x, point.y), 1, (0, 0, 255), -1)
                    cv2.imwrite(output_path + str(k) + "-cleared.jpg", img[y:y + h, x:x + w])
                    interocular_distance = calculate_distance(shape.part(13), shape.part(16))
                    left_eye_width = calculate_distance(shape.part(10), shape.part(13))
                    right_eye_width = calculate_distance(shape.part(16), shape.part(19))
                    left_eyebrow_width = calculate_distance(shape.part(0), shape.part(4))
                    right_eyebrow_width = calculate_distance(shape.part(5), shape.part(9))
                    eye_height_to_eyebrow_height = []
                    for i in range(0,5):
                        k = shape.part(i).x
                        l = shape.part(i).y
                        left_eyebrow_points.append((k,l))
                    for i in range(5,10):
                        m = shape.part(i).x
                        p = shape.part(i).y
                        right_eyebrow_points.append((m,p))
                    distancesG = [distance.euclidean(left_eyebrow_points[i], left_eyebrow_points[i+1]) for i in range(len(left_eyebrow_points)-1)]
                    curves.append(np.mean(distancesG))
                    distancesD = [distance.euclidean(right_eyebrow_points[i], right_eyebrow_points[i+1]) for i in range(len(right_eyebrow_points)-1)]
                    curves.append(np.mean(distancesD))
                    for i in range(10, 16):
                        eye_height = shape.part(i).y - shape.part(i-6).y
                        eyebrow_height = shape.part(i-6).y - min([shape.part(j).y for j in range(10, 16)])
                        eye_height_to_eyebrow_height.append(calculate_ratio(eyebrow_height, eye_height))

                    for i in range(16, 22):
                        eye_height = shape.part(i).y - shape.part(i-11).y
                        eyebrow_height = shape.part(i-11).y - min([shape.part(j).y for j in range(16, 22)])
                        eye_height_to_eyebrow_height.append(calculate_ratio(eyebrow_height, eye_height))

                    ratioint.extend([
                        calculate_ratio(left_eye_width, interocular_distance),
                        calculate_ratio(right_eye_width, interocular_distance),
                        calculate_ratio(left_eyebrow_width, interocular_distance),
                        calculate_ratio(right_eyebrow_width, interocular_distance)
                        ])
                    ratioint.extend(eye_height_to_eyebrow_height)
                    ratios.append((image_path, ratioint,curves))
                else:
                    print("No face detected")
    return ratios
```

```python

def process_imagesolo(image_path):  #This function returns the calculation data for a single image (img path, ratios and curves)

    ratiosolo = []
    if os.path.isfile(image_path):
        img = cv2.imread(image_path)
        img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(img_gray, 1.3, 5)
        roi_hsv = None
        for (x, y, w, h) in faces:
            roi_hsv = img_gray[y:y + h, x:x + w]
            face = dlib.rectangle(x, y, x+w, y+h)
            shape = predictor(img, face)
            ratioint = []
            left_eyebrow_points= []
            right_eyebrow_points= []
            curves = []

            if roi_hsv is not None:
                for j in range(22):  # There are 22 landmarks in total
                    point = shape.part(j)
                    cv2.circle(img, (point.x, point.y), 1, (0, 0, 255), -1) #I want to keep the colored version in the database bc I will compare colors
                cv2.imwrite(path_cleared , img[y:y + h, x:x + w])
                interocular_distance = calculate_distance(shape.part(13), shape.part(16))
                left_eye_width = calculate_distance(shape.part(10), shape.part(13))
                right_eye_width = calculate_distance(shape.part(16), shape.part(19))
                left_eyebrow_width = calculate_distance(shape.part(0), shape.part(4))
                right_eyebrow_width = calculate_distance(shape.part(5), shape.part(9))
                eye_height_to_eyebrow_height = []
                for i in range(0,5):
                    k = shape.part(i).x
                    l = shape.part(i).y
                    left_eyebrow_points.append((k,l))
                for i in range(5,10):
                    p = shape.part(i).x
                    m = shape.part(i).y
                    right_eyebrow_points.append((p,m))
                distancesG = [distance.euclidean(left_eyebrow_points[i], left_eyebrow_points[i+1]) for i in range(len(left_eyebrow_points)-1)]
                curves.append(np.mean(distancesG))
                distancesD = [distance.euclidean(right_eyebrow_points[i], right_eyebrow_points[i+1]) for i in range(len(right_eyebrow_points)-1)]
                curves.append(np.mean(distancesD))
                for i in range(10, 16):
                    eye_height = shape.part(i).y - shape.part(i-6).y
                    eyebrow_height = shape.part(i-6).y - min([shape.part(j).y for j in range(10, 16)])
                    eye_height_to_eyebrow_height.append(calculate_ratio(eyebrow_height, eye_height))

                for i in range(16, 22):
                    eye_height = shape.part(i).y - shape.part(i-11).y
                    eyebrow_height = shape.part(i-11).y - min([shape.part(j).y for j in range(16, 22)])
                    eye_height_to_eyebrow_height.append(calculate_ratio(eyebrow_height, eye_height))

                ratioint.extend([
                calculate_ratio(left_eye_width, interocular_distance),
                calculate_ratio(right_eye_width, interocular_distance),
                calculate_ratio(left_eyebrow_width, interocular_distance),
                calculate_ratio(right_eyebrow_width, interocular_distance)
                ])
                ratioint.extend(eye_height_to_eyebrow_height)
                ratiosolo.append((image_path, ratioint, curves))
            else:
                print("No face detected")
    return [(ratiosolo[0][0], ratiosolo[0][1], ratiosolo[0][2])]
```

```python
def compare_curves(curvesbdd, curvsolo):
    distances = []
    for index, sublist in enumerate(curvesbdd):
        sublist = np.array(sublist)
        relative_gap = np.abs((sublist - curvsolo) / curvsolo) * 100
        relative_gap = relative_gap.mean()
        distances.append((relative_gap, index))
    distances.sort()
    top_50_indexes = [index for _, index in distances[:50]]
    return top_50_indexes

def compare_differences(ratio1tot, image1, ratio2, image2):
    ratio1tot = np.array(ratio1tot)
    ratio2 = np.array(ratio2)

    differences = np.sum(np.abs(ratio1tot - ratio2), axis=1)

    indexes = np.argsort(differences)[:20]

    return indexes.tolist()

def compare_eye_color(image1_paths, image2_path):
    image2 = cv2.imread(image2_path)
    gray_image2 = cv2.cvtColor(image2, cv2.COLOR_BGR2GRAY)
    eye_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_eye.xml')
    eyes2 = eye_cascade.detectMultiScale(gray_image2, scaleFactor=1.1, minNeighbors=5)

    best_matches = []

    for image1_path in image1_paths:
        image1 = cv2.imread(image1_path)
        gray_image1 = cv2.cvtColor(image1, cv2.COLOR_BGR2GRAY)
        eyes1 = eye_cascade.detectMultiScale(gray_image1, scaleFactor=1.1, minNeighbors=5)

        eye_matches = []
        for (x1, y1, w1, h1) in eyes1:
            best_match = None
            best_similarity = float('inf')

            for (x2, y2, w2, h2) in eyes2:
                eye_region1 = gray_image1[y1:y1+h1, x1:x1+w1]
                eye_region2 = gray_image2[y2:y2+h2, x2:x2+w2]

                eye_region1 = cv2.resize(eye_region1, (w2, h2))
                similarity = cv2.absdiff(eye_region1, eye_region2).mean()

                if similarity < best_similarity:
                    best_similarity = similarity
                    best_match = (x2, y2, w2, h2)

            eye_matches.append(best_match)

        best_matches.append(eye_matches)

    sorted_images = sorted(zip(image1_paths, best_matches), key=lambda x: np.mean([similarity for similarity in x[1] if similarity is not None]))

    top_5_images = [image for image, _ in sorted_images[:5]]
    return top_5_images #top 5 almost 100%, top 4 ~80%, top 3 ~40%, top 2 ~30%, top 1 ~15% (100 tests done)
```

```python
def Globale_Case(): #This function returns the top5 paths
    resultatbdd = process_imagesbddphoto()
    imagesbdd, ratiobdd,curvesbdd = zip(*resultatbdd)
    resultatsolo = process_imagesolo(path_newsave)
    imagesolo, ratiosolo,curvsolo = zip(*resultatsolo)
    top_50_curves_i = compare_curves(curvesbdd,curvsolo)
    triratiobdd = []
    for i in top_50_curves_i:
        triratiobdd.extend([ratiobdd[i]])
    top_10_indexes = compare_differences(triratiobdd,imagesbdd,ratiosolo,imagesolo)
    triratio = []
    for j in top_10_indexes:
        triratio.extend([triratiobdd[j]])
    str_list = [",".join(map(str, sublist)) for sublist in triratio]
    result = ",".join('"' + sublist + '"' for sublist in str_list)
    TOP10_VISAGES= get_filevisage_by_ratios(result)
    TOP5 = compare_eye_color(TOP10_VISAGES, path_newsave)
    return TOP5

def GlobalTopChoice(TOP):   #This function takes top5 paths and print last and first names of the top5.
    TOP = ','.join(['"{}"'.format(path) for path in TOP])
    identity_list = get_name_by_filevisage(TOP)
    for i, identite in enumerate(identity_list):
        print(f"Press {i} if your identity is: "+identite[0]+" "+identite[1]+" !")
        get_filevisage_by_name(identite[0], identite[1])        #for practical purposes
        print("\n")
    print("Please press any other button if your name does not appear.")
    x = input()
    if x.isdigit() and int(x) <= len(identity_list)-1:
        identite_selectionnee = identity_list[int(x)]
        print(f"Welcome, {identite_selectionnee[0]} {identite_selectionnee[1]} !")
    else:
        print("Your name does not appear, please make sure you are correctly positioned. You must repeat the process.")


def ApplicationCase():
    TOP = Globale_Case()
    GlobalTopChoice(TOP)
```

```python
def Health_Center():  #This function searches for individuals in the top5 who have an appointment on the date specified in the code, to improve the probability of finding the right face.
    CreateBDD()
    liste_file_visage = Globale_Case()
    conn = mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="root",
        database="donneespatient")
    cursor = conn.cursor()
    date = "2023-06-29"
    query = """
        SELECT Patient.Nom, Patient.Prenom, Rdv.Emplacement, Docteur.Nom, Docteur.Prenom, Docteur.Fonction, Rdv.Heure
        FROM Patient
        JOIN Rdv ON Rdv.Patient_FileVisage = Patient.FileVisage
        JOIN Docteur ON Rdv.ID_Doctor = Docteur.ID
        WHERE Rdv.Daterdv = %s AND Patient.FileVisage = %s
        """
    Top = []
    for file_visage in liste_file_visage:
        cursor.execute(query, (date, file_visage))
        results = cursor.fetchall()
        for row in results:
                nom_patient, prenom_patient, emplacement_rdv, nom_docteur, prenom_docteur, fonction_docteur, heure_rdv = row
                Top.append([nom_patient, prenom_patient, emplacement_rdv, nom_docteur, prenom_docteur, fonction_docteur, heure_rdv])
        cursor.nextset()
    if len(Top) == 0:
        GlobalTopChoice(liste_file_visage)
    elif len(Top) == 1:
        print("Welcome " + Top[0][0] + Top[0][1] + ". Please find below the details of your appointment:")
        print("Location:", Top[0][2])
        print("Doctor: Dr." + Top[0][3] + " " + Top[0][4] + " ," + Top[0][5])
        print("Appointment time", Top[0][6])
        print("-----------------------------------")
    else:
        for i, identite in enumerate(Top):
            print(f"Press {i} if your identity is: "+identite[0]+" "+identite[1]+" !")
        print("Please press any other button if your name does not appear.")
        x = int(input())
        if int(x) <= len(Top)-1: #REPRENDRE ça
            for i in range(len(Top)):
                if Top[i][0] != Top[x][0] and Top[i][1] != Top[x][1]:
                    Top[i] = []
                else:
                    print("Welcome " + Top[i][0] +" "+ Top[i][1] + ". Please find below the details of your appointment:")
                    print("Location:", Top[i][2])
                    print("Doctor: Dr." + Top[i][3] + " " + Top[i][4] + " ," + Top[i][5])
                    print("Appointment time", Top[i][6])
                    print("-----------------------------------")
        else:
            print("Your name does not appear, please repeat the process, making sure you are correctly positioned.")
    cursor.close()
    conn.close()
```

```python
def NewPatient(file_visage,ratios): #This function adds a new Patient in the Mysql database
    conn = mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="root",
        database="donneespatient")
    cursor = conn.cursor()
    print("Last name of the patient?")
    nom = str(input())
    print("First name of the patient?")
    prenom = str(input())

    ratios_str = ','.join(str(ratio) for ratio in ratios)
    insert_query = "INSERT INTO Patient (Nom, Prenom, Ratios, FileVisage) VALUES (%s, %s, %s, %s)"
    values = (nom, prenom, ratios_str, file_visage)

    cursor.execute(insert_query, values)

    conn.commit()
    cursor.close()
    conn.close()

def ScenarioCenter():   #Scenario case

    print("Hello, please choose an option: ")
    print("Press 1 if you are a new visitor \nPress 2 if you already came in this establishment before")
    t = int(input())
    while t != 1 and t !=2:
        print("Error, you must press 1 if you are a new visitor, or press 2 if you already came in this establishment before")
    if t == 1:
        ratio = process_imagesolo(path_newsave)
        NewPatient(base_image_path+str(num)+"-with-mask.jpg", ratio[0][1])
    if t == 2:
        Health_Center()
```

```python
#%%  MySQL DB creation
def RemplissagePatientsbdd():   #create random patients for each images in the photodb
    conn = mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="root",
        database="donneespatient")
    cursor = conn.cursor()
    ratios = process_imagesbddphoto()
    ratios_str = ','.join(str(ratio) for ratio in ratios)

    insert_query = "INSERT INTO Patient (Nom, Prenom, Ratios, FileVisage) VALUES (%s, %s, %s, %s)"
    Listenom = ["Lastname"+ str(i) for i in range(1000)]
    Listeprenom = ["Firstname"+str(i) for i in range(1000)]
    k = 0
    for ratio_data in ratios:
        nom = Listenom[k]
        prenom = Listeprenom[k]
        k+=1
        file_visage = ratio_data[0]
        ratios_str = ','.join(str(ratio) for ratio in ratio_data[1])

        values = (nom, prenom, ratios_str, file_visage)

        cursor.execute(insert_query, values)

    conn.commit()
    cursor.close()
    conn.close()
def RemplissageMedecinsbdd():   #create random doctors
    conn = mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="root",
        database="donneespatient")
    cursor = conn.cursor()

    start_id = 1001
    num_doctors = 50
    fonctions = ["General Practioner","Dermatologist","Radiologist","Cardiologist","Endocrynologist"]

    for i in range(start_id, start_id + num_doctors):
        id_doctor = str(i)
        nom = f"Nom{i - start_id + 1}"
        prenom = f"Prenom{i - start_id + 1}"
        fonction = random.choice(fonctions)

        sql = "INSERT INTO Docteur (ID, Nom, Prenom, Fonction) VALUES (%s, %s, %s, %s)"
        values = (id_doctor, nom, prenom, fonction)
        cursor.execute(sql, values)

    conn.commit()

    cursor.close()
    conn.close()
```

```python
def RemplissageRDVbdd():    #create random appointments
    conn = mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="root",
        database="donneespatient")
    cursor = conn.cursor()

    start_date = datetime.now().date()
    end_date = start_date + timedelta(days=30)

    etages = list(range(1, 11))
    salles = list(range(1, 31))
    ascenseurs = ["Right", "Left"]

    current_date = start_date
    while current_date <= end_date:
        cursor.execute("SELECT FileVisage FROM Patient")
        file_visages = cursor.fetchall()
        cursor.execute("SELECT ID FROM Docteur")
        id_docteur = cursor.fetchall()


        for _ in range(30):
            hour = random.randint(8, 18)
            minute = random.randint(0, 59)
            appointment_time = datetime(current_date.year, current_date.month, current_date.day, hour, minute).time()
            appointment_id = str(appointment_time) +str(hour) + str(minute) + str(_)+ str(random.randint(1,99999))
            etage = random.choice(etages)
            salle = random.choice(salles)
            ascenseur = random.choice(ascenseurs)
            emplacement = f"Floor {etage}, Room {salle},{ascenseur} elevator"
            patient_file_visage = random.choice(file_visages)[0]
            id_doctor = random.choice(id_docteur)[0]

            sql = "INSERT INTO Rdv (Id, Emplacement, Daterdv, Patient_FileVisage, Heure, ID_Doctor) VALUES (%s, %s, %s, %s, %s, %s)"
            values = (appointment_id, emplacement, current_date, patient_file_visage, appointment_time, id_doctor)
            cursor.execute(sql, values)

        current_date += timedelta(days=1)
    conn.commit()

    cursor.close()
    conn.close()

def CreateBDD():
    RemplissagePatientsbdd()
    RemplissageMedecinsbdd()
    RemplissageRDVbdd()
```

```python
#%% Opening of the door
if maskon == True and identity != 0:
    GPIO.setmode(GPIO.BOARD)
    Moteur1A = 16
    Moteur1B = 18
    Moteur1E = 22
    GPIO.setup(Moteur1A,GPIO.OUT)
    GPIO.setup(Moteur1B,GPIO.OUT)
    GPIO.setup(Moteur1E,GPIO.OUT)
    pwm = GPIO.PWM(Moteur1E,50)
    #Cycle d'ouverture
    pwm.start(34)
    print("Rotation sens direct")
    GPIO.output(Moteur1A,GPIO.HIGH)
    GPIO.output(Moteur1B,GPIO.LOW)
    GPIO.output(Moteur1E,GPIO.HIGH)

    sleep(3.6)

    GPIO.output(Moteur1E,GPIO.LOW)
    pwm.stop()
    sleep(5)
    #Cycle de fermeture
    pwm.start(34)
    print("Rotation sens inverse")
    GPIO.output(Moteur1A,GPIO.HIGH)
    GPIO.output(Moteur1B,GPIO.LOW)
    GPIO.output(Moteur1E,GPIO.HIGH)

    sleep(3.6)

    print("Arrêt du moteur")
    GPIO.output(Moteur1E,GPIO.LOW)

    pwm.stop()
    GPIO.cleanup()
```