In [ ]:

```
pip install face_recognition
```

In [ ]:

```
import face_recognition
```

In [ ]:

```
import os
import shutil

def compare_faces(face_image_path, leader_folder):
    face_image = face_recognition.load_image_file(face_image_path)
    face_encodings = face_recognition.face_encodings(face_image)

    # Check if any face encoding is available
    if len(face_encodings) == 0:
        return False

    # Iterate over the face images associated with the same leader
    for file in os.listdir(leader_folder):
        if file.endswith(".jpg") or file.endswith(".png"):
            leader_face_image_path = os.path.join(leader_folder, file)
            leader_face_image = face_recognition.load_image_file(leader_face_image_path)
            leader_face_encodings = face_recognition.face_encodings(leader_face_image)

            # Check if any face encoding is available
            if len(leader_face_encodings) == 0:
                continue

            # Compare the current face with the faces associated with the leader
            results = face_recognition.compare_faces(leader_face_encodings, face_encodings[0], tolerance=0.35)

            # Calculate the percentage of matching faces
            matching_percentage = sum(results) / len(results) * 100

            # If matching percentage is greater than 50%, categorize the face
            if matching_percentage > 50:
                return True

    return False

def process_leaders(folder_path, output_folder_path):
    for leader_folder in os.listdir(folder_path):
        leader_folder_path = os.path.join(folder_path, leader_folder)
        if os.path.isdir(leader_folder_path):
            leader_processed_folder_path = os.path.join(output_folder_path, "leader_pictures_processed", leader_f
            os.makedirs(leader_processed_folder_path, exist_ok=True)

            for face_file in os.listdir(leader_folder_path):
                if face_file.endswith(".jpg") or face_file.endswith(".png"):
                    face_image_path = os.path.join(leader_folder_path, face_file)
                    if compare_faces(face_image_path, leader_folder_path):
                        destination_path = os.path.join(leader_processed_folder_path, face_file)
                        shutil.copy(face_image_path, destination_path)
                        print(f"Face {face_file} categorized for leader {leader_folder}")

# Specify the input folder path where the extracted faces are stored
extracted_faces_folder_path = "/Users/kei/Desktop/Extracted2"

# Specify the output folder path on your desktop
output_folder_path = os.path.expanduser("/Users/kei/Desktop/Processed2")

# Create the leader_pictures_processed folder if it doesn't exist
processed_folder_path = os.path.join(output_folder_path, "leader_pictures_processed")
os.makedirs(processed_folder_path, exist_ok=True)

print("Processing leaders...")
process_leaders(extracted_faces_folder_path, output_folder_path)

print("Leader processing completed!")
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js