# SALUDO LDS - COLOMBIANA

Presentado por:
Dana Valentina Martinez Velosa
Anyi Shirley Lozada Aguilar

Materia:
Ingeniería de Software

Programa:
Tec. Desarrollo de software

Universidad:
San Buenaventura Sede Bogotá

Año:
2023

# Contenido

# Resumen

Este proyecto se enfoca en solucionar uno de los problemas más comunes que enfrentan las personas sordas en su interacción con las entidades públicas: la falta de una comunicación efectiva. En muchas ocasiones, las personas sordas encuentran dificultades para expresar sus necesidades y comprender las respuestas de los funcionarios públicos debido a la falta de capacitación en lenguaje de señas y otras herramientas de comunicación.

Para solucionar este problema, se ha propuesto la implementación de una aplicación que permita la comunicación entre personas sordas y los funcionarios de las entidades públicas de manera más eficaz. Esta herramienta tiene como función principal transcribir de manera precisa el lenguaje de señas en texto escrito, lo que permitirá a cualquier persona que no conozca el lenguaje de señas comprender lo que se está comunicando.

La implementación de esta herramienta no solo beneficiará a las personas sordas, sino también a los funcionarios públicos que, al no tener conocimientos en lenguaje de señas, se ven limitados en su capacidad de atender a todas las personas que requieren de sus servicios.

# Abstract

With the transcription tool, they will be able to communicate more effectively with deaf people and provide them with a more efficient and satisfying service.

It is important to highlight that, in many cases, the lack of effective communication can cause delays in conflict resolution and other processes in public entities. With the implementation of this tool, it is expected that the processes will be streamlined and that the waiting time in the care of deaf people will be reduced.

In short, this project represents an important advance in the social inclusion of deaf people, since it provides them with a tool to communicate effectively with public entities. In addition, it is also a step towards a more inclusive society that is aware of the needs of all people.

# Introducción

Los antecedentes teóricos y prácticos que dieron origen a este proyecto se relacionan con el reconocimiento de la importancia de garantizar el acceso a los servicios públicos a todas las personas, independientemente de sus capacidades físicas o sensoriales. En este sentido, se ha evidenciado que las personas sordas enfrentan barreras de comunicación que limitan su participación plena en la sociedad y su acceso a los servicios públicos.
Los objetivos de este proyecto son mejorar la comunicación entre las personas sordas y los funcionarios públicos, así como facilitar el acceso de las personas sordas a los servicios públicos y garantizar su inclusión en la sociedad. Los alcances de esta herramienta se limitan a la transcripción del lenguaje de señas a texto escrito, por lo que no sustituye la importancia del aprendizaje y uso del lenguaje de señas.

Las limitaciones de esta herramienta se relacionan con la necesidad de contar con un intérprete de lenguaje de señas que pueda realizar la interpretación en tiempo real. Además, esta herramienta requiere de un dispositivo tecnológico y conexión a internet para su funcionamiento.

La metodología empleada para la implementación de esta herramienta ha sido la investigación y desarrollo de tecnologías de reconocimiento de gestos y lenguaje de señas, así como el trabajo en conjunto con personas sordas y expertos en accesibilidad y tecnología. También se ha realizado una evaluación de la efectividad y usabilidad de la herramienta mediante pruebas piloto con personas sordas y funcionarios públicos.

# 1. Marco Teórico

La fundamentación sistémica, organizada y objetiva para afrontar el problema de la falta de comunicación efectiva entre personas sordas y funcionarios públicos se encuentra en la comprensión de la accesibilidad y la inclusión en el contexto de los servicios públicos. La Convención sobre los Derechos de las Personas con Discapacidad, ratificada por numerosos países, incluyendo aquellos donde se desarrolla este proyecto, reconoce el derecho de las personas con discapacidad a acceder a los servicios públicos en igualdad de condiciones que las demás personas.

En este sentido, se requiere una comprensión clara de los conceptos de accesibilidad y diseño universal, así como de las necesidades específicas de las personas sordas en términos de comunicación y acceso a la información. La implementación de esta herramienta se basa en el conocimiento especializado en tecnologías de reconocimiento de gestos y lenguaje de señas, así como en la colaboración con personas sordas y expertos en accesibilidad y tecnología.

Además, la fundamentación sistémica, organizada y objetiva para afrontar este problema se relaciona con la comprensión de la importancia del lenguaje de señas como lengua natural de las personas sordas, reconocida por la Convención sobre los Derechos de las Personas con Discapacidad. Es necesario comprender la estructura gramatical y las características del lenguaje de señas para poder desarrollar una herramienta efectiva que permita la transcripción precisa de los gestos y signos en texto escrito.

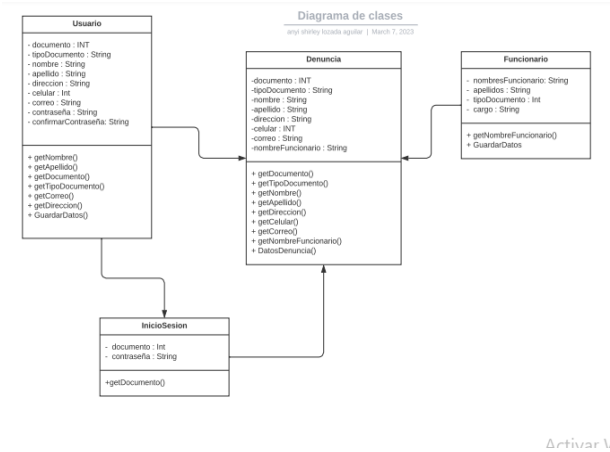
Asimismo, se requiere una comprensión de los procesos de atención al público en las entidades públicas, así como de las barreras de comunicación que enfrentan las personas sordas en estos contextos. La implementación de esta herramienta no solo depende de la tecnología, sino también de la capacitación y sensibilización de los funcionarios públicos en materia de accesibilidad e inclusión.

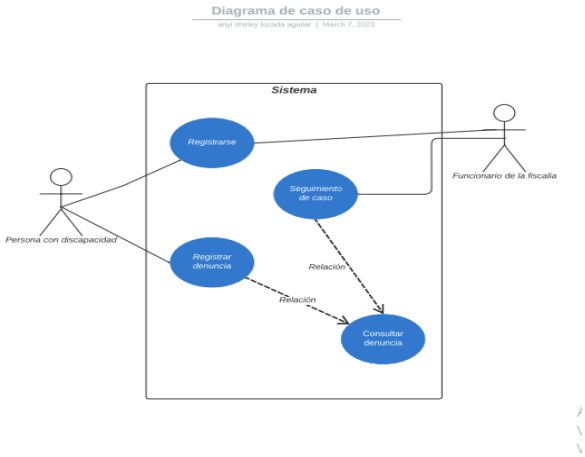
En resumen, la base fundamentación sistémica, organizada y objetiva para afrontar el problema de la falta de comunicación efectiva entre personas sordas y funcionarios públicos se encuentra en la comprensión de la accesibilidad y la inclusión en los servicios públicos, el conocimiento especializado en tecnologías de reconocimiento de gestos y lenguaje de señas, la comprensión del lenguaje de señas como lengua natural de las personas sordas, la comprensión de los procesos de atención al público en las entidades públicas y la capacitación y sensibilización de los funcionarios públicos en materia de accesibilidad e inclusión.

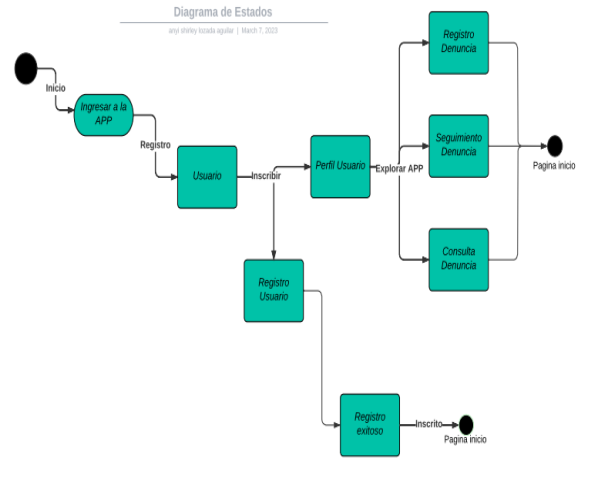
# 2. Desarrollo de Ingeniería

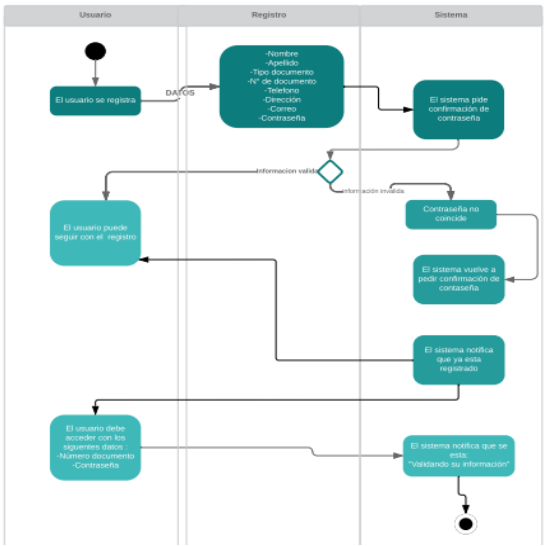
Se desarrollaron las siguientes pautas en ingeniería:

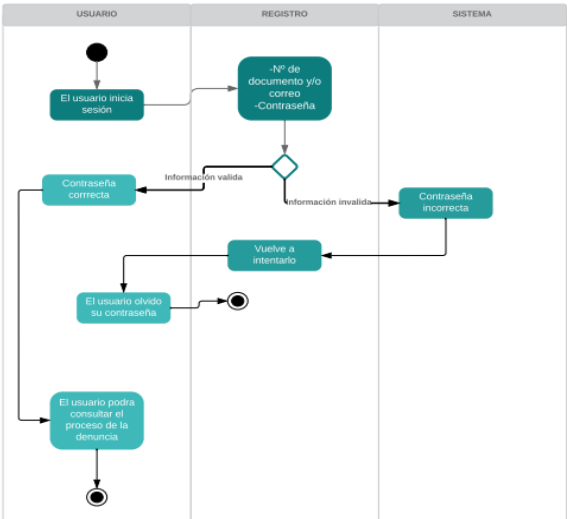
- Diagramas de: casos de uso, clases, actividades y estados. Estos diagramas se realizan para saber cómo se relacionan cada uno de estos.

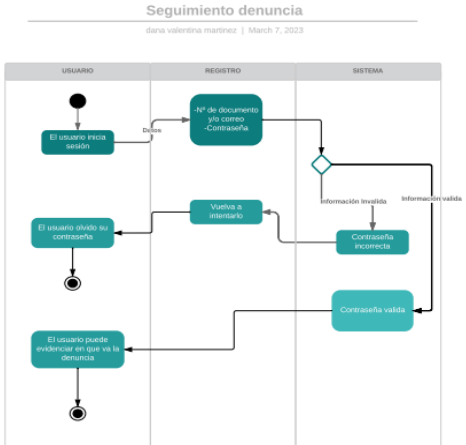


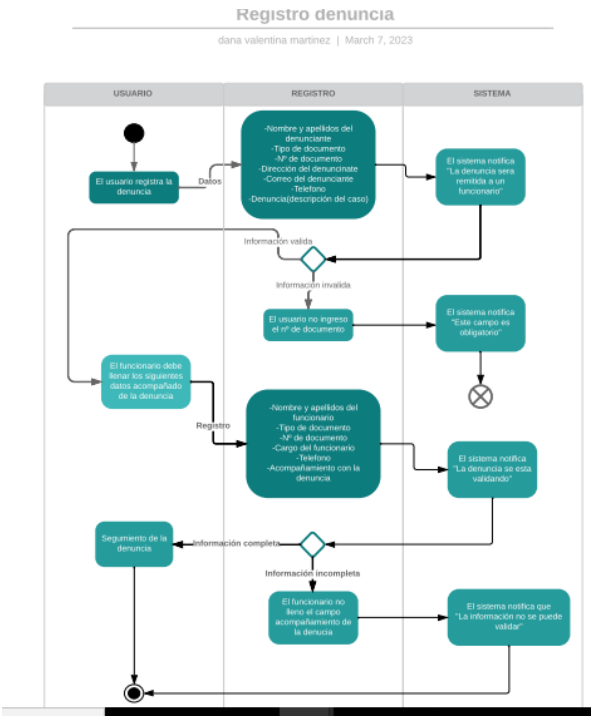



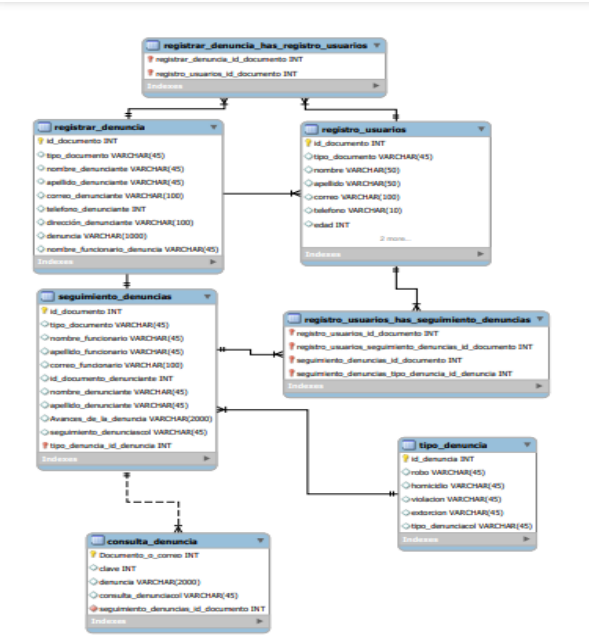



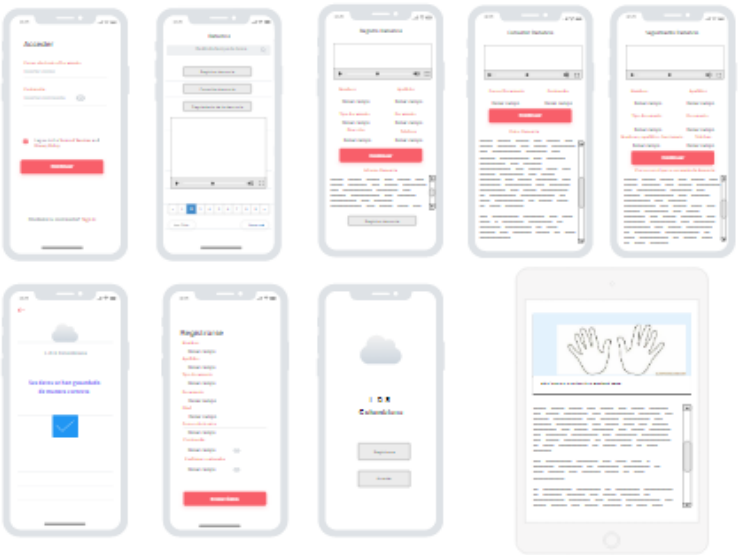





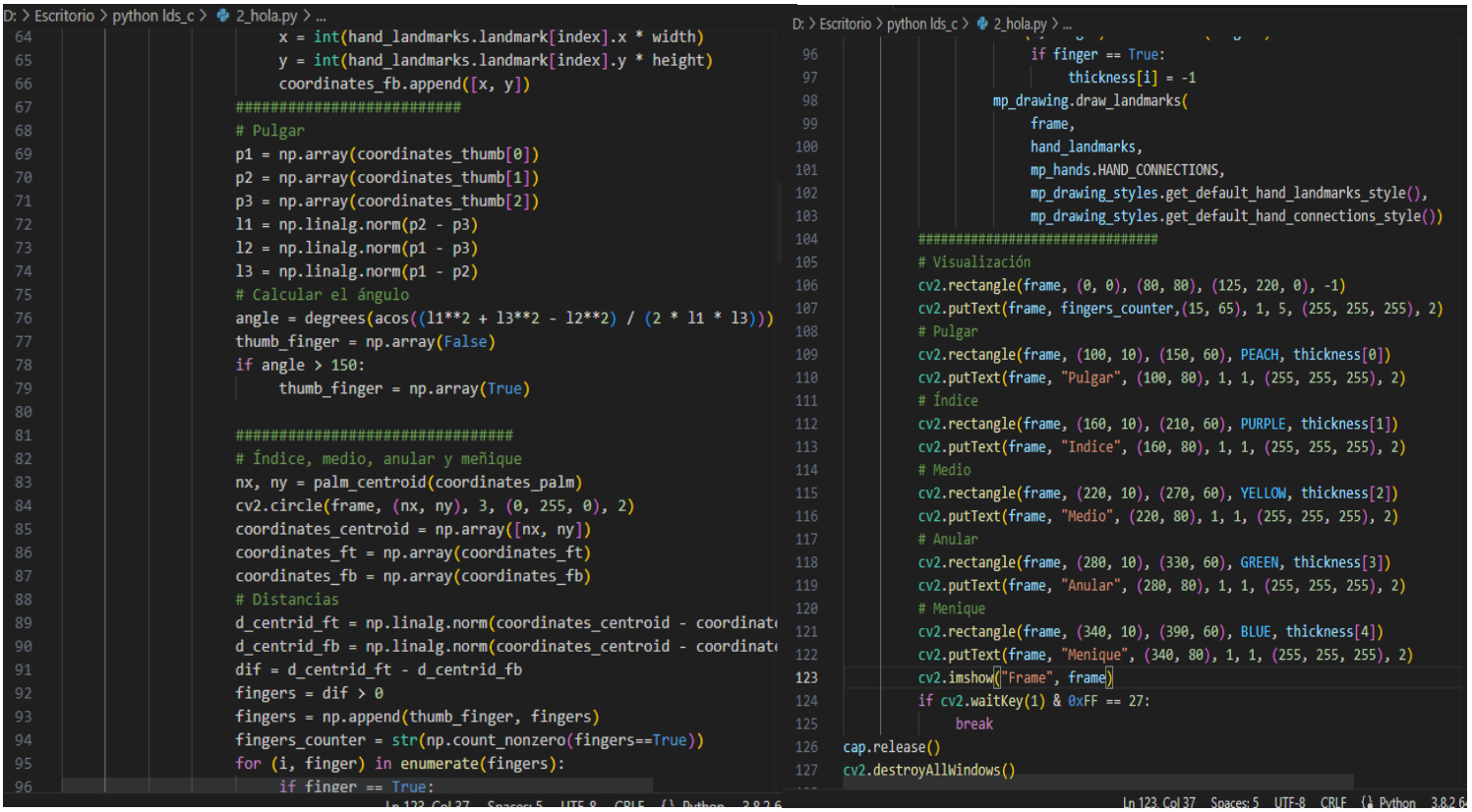

- Entidad Relación




- Mockups: Se desarrolla un diseño de cómo va hacer la aplicación.



- Código del programa: Se realiza un código en python versión 3.8 y 3.11 a esta se le aplica el OpenCV. Estos son avances que se tienen de la aplicación, con este ya reconoce las manos.



```
D: > Escritorio > python lds_c > 2_hola.py > ...
64                      x = int(hand_landmarks.landmark[index].x * width)
65                      y = int(hand_landmarks.landmark[index].y * height)
66                      coordinates_fb.append([x, y])
67                  ##########################
68                  # Pulgar
69                  p1 = np.array(coordinates_thumb[0])
70                  p2 = np.array(coordinates_thumb[1])
71                  p3 = np.array(coordinates_thumb[2])
72                  l1 = np.linalg.norm(p2 - p3)
73                  l2 = np.linalg.norm(p1 - p3)
74                  l3 = np.linalg.norm(p1 - p2)
75                  # Calcular el ángulo
76                  angle = degrees(acos((l1**2 + l3**2 - l2**2) / (2 * l1 * l3)))
77                  thumb_finger = np.array(False)
78                  if angle > 150:
79                      thumb_finger = np.array(True)
80
81                  ##############################
82                  # Índice, medio, anular y meñique
83                  nx, ny = palm_centroid(coordinates_palm)
84                  cv2.circle(frame, (nx, ny), 3, (0, 255, 0), 2)
85                  coordinates_centroid = np.array([nx, ny])
86                  coordinates_ft = np.array(coordinates_ft)
87                  coordinates_fb = np.array(coordinates_fb)
88                  # Distancias
89                  d_centrid_ft = np.linalg.norm(coordinates_centroid - coordinat
90                  d_centrid_fb = np.linalg.norm(coordinates_centroid - coordinat
91                  dif = d_centrid_ft - d_centrid_fb
92                  fingers = dif > 0
93                  fingers = np.append(thumb_finger, fingers)
94                  fingers_counter = str(np.count_nonzero(fingers==True))
95                  for (i, finger) in enumerate(fingers):
96                      if finger == True:
```

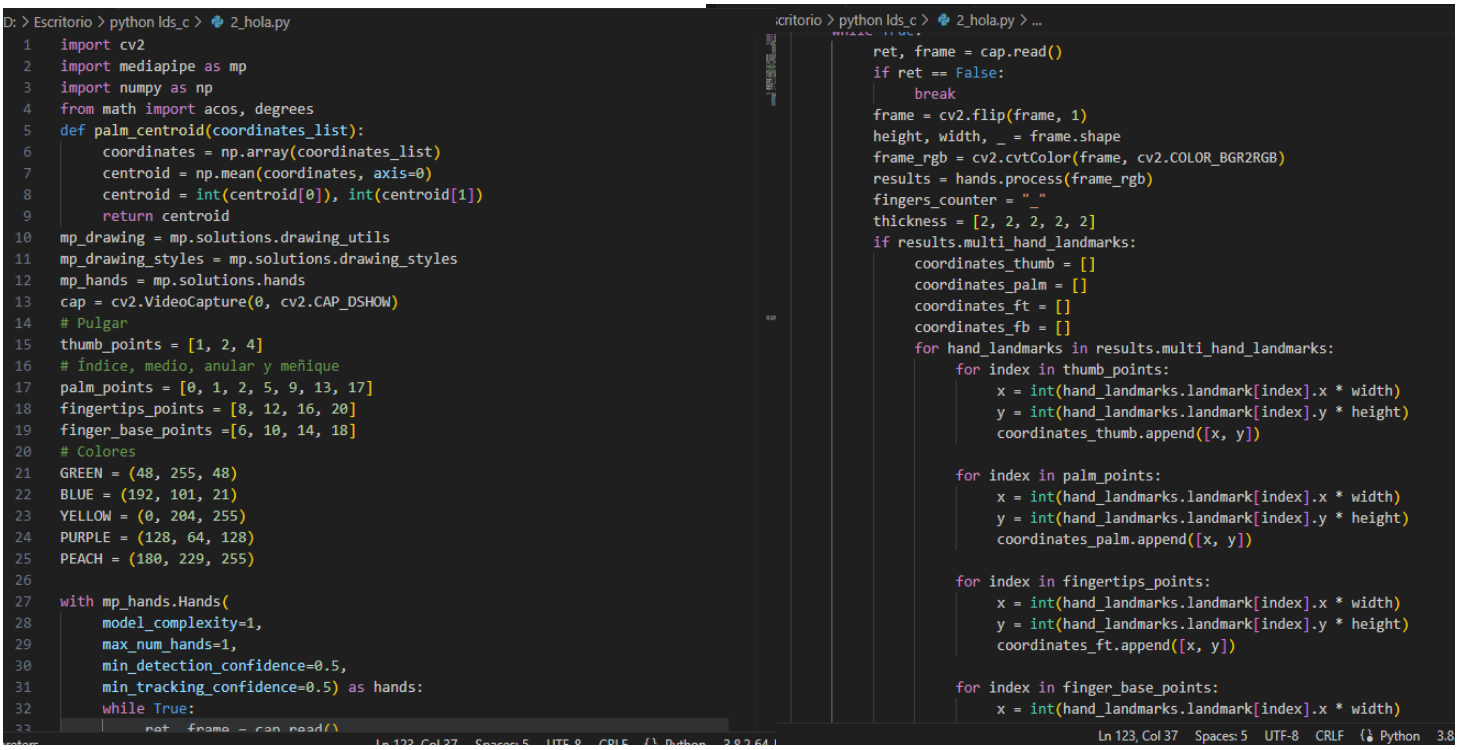
```
D: > Escritorio > python lds_c > 2_hola.py > ...
96                      if finger == True:
97                          thickness[i] = -1
98                  mp_drawing.draw_landmarks(
99                      frame,
100                     hand_landmarks,
101                     mp_hands.HAND_CONNECTIONS,
102                     mp_drawing_styles.get_default_hand_landmarks_style(),
103                     mp_drawing_styles.get_default_hand_connections_style())
104             ##############################
105             # Visualización
106             cv2.rectangle(frame, (0, 0), (80, 80), (125, 220, 0), -1)
107             cv2.putText(frame, fingers_counter,(15, 65), 1, 5, (255, 255, 255), 2)
108             # Pulgar
109             cv2.rectangle(frame, (100, 10), (150, 60), PEACH, thickness[0])
110             cv2.putText(frame, "Pulgar", (100, 80), 1, 1, (255, 255, 255), 2)
111             # Índice
112             cv2.rectangle(frame, (160, 10), (210, 60), PURPLE, thickness[1])
113             cv2.putText(frame, "Indice", (160, 80), 1, 1, (255, 255, 255), 2)
114             # Medio
115             cv2.rectangle(frame, (220, 10), (270, 60), YELLOW, thickness[2])
116             cv2.putText(frame, "Medio", (220, 80), 1, 1, (255, 255, 255), 2)
117             # Anular
118             cv2.rectangle(frame, (280, 10), (330, 60), GREEN, thickness[3])
119             cv2.putText(frame, "Anular", (280, 80), 1, 1, (255, 255, 255), 2)
120             # Menique
121             cv2.rectangle(frame, (340, 10), (390, 60), BLUE, thickness[4])
122             cv2.putText(frame, "Menique", (340, 80), 1, 1, (255, 255, 255), 2)
123             cv2.imshow("Frame", frame)
124             if cv2.waitKey(1) & 0xFF == 27:
125                 break
126     cap.release()
127     cv2.destroyAllWindows()
Ln 123, Col 37  Spaces: 5  UTF-8  CRLF  {} Python  3.8.2 64
```

# 3. Resultados

Se realizaron varias actividades de desglose para saber cómo se realizará la aplicación para ello se desarrollaron diagramas de caso de uso para saber que va realizar la app, diagrama de estados es una paso a paso de cómo se ingresara, diagramas de clases este muestra la relación entre los objetos que están programados. Se llevó a cabo un diseño de la entidad relación para saber que tablas se van a implementar en nuestra base de datos. También se diseñan los mockups para saber el contenido que va a tener la aplicación.

Los resultados que se han obtenido en el código hasta el momento es el reconocimiento de las manos, este hace un conteo de los dedos que el usuario muestre y se tiene una barra en donde los dedos de la mano tienen un color diferente, para identificar qué dedo se utilizó.

Se harán cambios los cuales ya no identificara el conteo de los dedos, si no que ahora este realizara la identificación de las señas de las personas sordas y se traduce a texto la lengua de señas. La app también reconocerá la emoción que la persona exprese en el momento. Para ello se realizarán los respectivos cambios y se implementará un reconocimiento facial de las emociones en el código.



```
D: > Escritorio > python lds_c > 2_hola.py
import cv2
import mediapipe as mp
import numpy as np
from math import acos, degrees
def palm_centroid(coordinates_list):
    coordinates = np.array(coordinates_list)
    centroid = np.mean(coordinates, axis=0)
    centroid = int(centroid[0]), int(centroid[1])
    return centroid
mp_drawing = mp.solutions.drawing_utils
mp_drawing_styles = mp.solutions.drawing_styles
mp_hands = mp.solutions.hands
cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
# Pulgar
thumb_points = [1, 2, 4]
# Índice, medio, anular y meñique
palm_points = [0, 1, 2, 5, 9, 13, 17]
fingertips_points = [8, 12, 16, 20]
finger_base_points =[6, 10, 14, 18]
# Colores
GREEN = (48, 255, 48)
BLUE = (192, 101, 21)
YELLOW = (0, 204, 255)
PURPLE = (128, 64, 128)
PEACH = (180, 229, 255)

with mp_hands.Hands(
    model_complexity=1,
    max_num_hands=1,
    min_detection_confidence=0.5,
    min_tracking_confidence=0.5) as hands:
    while True:
        ret, frame = cap.read()
```

```
ret, frame = cap.read()
if ret == False:
    break
frame = cv2.flip(frame, 1)
height, width, _ = frame.shape
frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
results = hands.process(frame_rgb)
fingers_counter = "_"
thickness = [2, 2, 2, 2, 2]
if results.multi_hand_landmarks:
    coordinates_thumb = []
    coordinates_palm = []
    coordinates_ft = []
    coordinates_fb = []
    for hand_landmarks in results.multi_hand_landmarks:
        for index in thumb_points:
            x = int(hand_landmarks.landmark[index].x * width)
            y = int(hand_landmarks.landmark[index].y * height)
            coordinates_thumb.append([x, y])

        for index in palm_points:
            x = int(hand_landmarks.landmark[index].x * width)
            y = int(hand_landmarks.landmark[index].y * height)
            coordinates_palm.append([x, y])

        for index in fingertips_points:
            x = int(hand_landmarks.landmark[index].x * width)
            y = int(hand_landmarks.landmark[index].y * height)
            coordinates_ft.append([x, y])

        for index in finger_base_points:
            x = int(hand_landmarks.landmark[index].x * width)
```

D: > Escritorio > python lds_c > 2_hola.py > ...

```
                x = int(hand_landmarks.landmark[index].x * width)
                y = int(hand_landmarks.landmark[index].y * height)
                coordinates_fb.append([x, y])
            ##########################
            # Pulgar
            p1 = np.array(coordinates_thumb[0])
            p2 = np.array(coordinates_thumb[1])
            p3 = np.array(coordinates_thumb[2])
            l1 = np.linalg.norm(p2 - p3)
            l2 = np.linalg.norm(p1 - p3)
            l3 = np.linalg.norm(p1 - p2)
            # Calcular el ángulo
            angle = degrees(acos((l1**2 + l3**2 - l2**2) / (2 * l1 * l3)))
            thumb_finger = np.array(False)
            if angle > 150:
                thumb_finger = np.array(True)

            ##############################
            # Índice, medio, anular y meñique
            nx, ny = palm_centroid(coordinates_palm)
            cv2.circle(frame, (nx, ny), 3, (0, 255, 0), 2)
            coordinates_centroid = np.array([nx, ny])
            coordinates_ft = np.array(coordinates_ft)
            coordinates_fb = np.array(coordinates_fb)
            # Distancias
            d_centrid_ft = np.linalg.norm(coordinates_centroid - coordinat
            d_centrid_fb = np.linalg.norm(coordinates_centroid - coordinat
            dif = d_centrid_ft - d_centrid_fb
            fingers = dif > 0
            fingers = np.append(thumb_finger, fingers)
            fingers_counter = str(np.count_nonzero(fingers==True))
            for (i, finger) in enumerate(fingers):
                if finger == True:
```

> Escritorio > python lds_c > 2_hola.py > ...

```
                    if finger == True:
                        thickness[i] = -1
                mp_drawing.draw_landmarks(
                    frame,
                    hand_landmarks,
                    mp_hands.HAND_CONNECTIONS,
                    mp_drawing_styles.get_default_hand_landmarks_style(),
                    mp_drawing_styles.get_default_hand_connections_style())
        ##############################
        # Visualización
        cv2.rectangle(frame, (0, 0), (80, 80), (125, 220, 0), -1)
        cv2.putText(frame, fingers_counter,(15, 65), 1, 5, (255, 255, 255), 2)
        # Pulgar
        cv2.rectangle(frame, (100, 10), (150, 60), PEACH, thickness[0])
        cv2.putText(frame, "Pulgar", (100, 80), 1, 1, (255, 255, 255), 2)
        # Índice
        cv2.rectangle(frame, (160, 10), (210, 60), PURPLE, thickness[1])
        cv2.putText(frame, "Indice", (160, 80), 1, 1, (255, 255, 255), 2)
        # Medio
        cv2.rectangle(frame, (220, 10), (270, 60), YELLOW, thickness[2])
        cv2.putText(frame, "Medio", (220, 80), 1, 1, (255, 255, 255), 2)
        # Anular
        cv2.rectangle(frame, (280, 10), (330, 60), GREEN, thickness[3])
        cv2.putText(frame, "Anular", (280, 80), 1, 1, (255, 255, 255), 2)
        # Menique
        cv2.rectangle(frame, (340, 10), (390, 60), BLUE, thickness[4])
        cv2.putText(frame, "Menique", (340, 80), 1, 1, (255, 255, 255), 2)
        cv2.imshow("Frame", frame)
        if cv2.waitKey(1) & 0xFF == 27:
            break
cap.release()
cv2.destroyAllWindows()
```

```
import os
import pickle

import mediapipe as mp
import cv2
import matplotlib.pyplot as plt


mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
mp_drawing_styles = mp.solutions.drawing_styles

hands = mp_hands.Hands(static_image_mode=True, min_detection_confidence=0.3)

DATA_DIR = './data'

data = []
labels = []
for dir_ in os.listdir(DATA_DIR):
    for img_path in os.listdir(os.path.join(DATA_DIR, dir_)):
        data_aux = []

        x_ = []
        y_ = []
```

# 5. Conclusiones

Este proyecto se enfoca en una herramienta que facilite la comunicación de las personas sordas y los funcionarios que se encuentran en las oficinas de las entidades públicas. Los resultados de la app es una versión beta ya que se quiere que esta pueda llegar a brindar el apoyo a las personas sordas ya que esta identificara la lengua de señas y su traducción a texto, así mismo identificará la voz de los funcionarios y se traduce a lengua de señas.

Este proyecto quiere llegar a MinCiencias ya que es un ente que patrocina proyectos en las diferentes convocatorias y pautas que este pide. Está propuesta tendría que ser aprobada por dicho Ministerio. Esta entidad continuamente tendría que ser informada sobre los avances y gastos a lo largo de la ejecución del proyecto.

# Referencias

Mockups:

https://app.moqups.com/LpkYFbJaSFaMfHCUJ140mPz6eRbIby9N/view/page/a880590a1

Jira:

https://dana-valen.atlassian.net/jira/software/projects/LDSC/boards/3