UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE ELÉCTRONICA Y AUTOMATIZACIÓN

BOT DE SERVICIOS PARA CUMPLEAÑOS CON RECONOCIMINETO FACIAL E INTERACCIÓN HUMANO MÁQUINA PARA EL RESTAURANTE "LA TABERNA"

Trabajo de titulación previo a la obtención del

Título de Ingeniero Electrónica y Automatización

AUTOR: Alisson Tamara Guaño Clavijo

TUTOR: Andrés Sebastián Calero Calero

Quito-Ecuador

2024

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Yo, Alisson Tamara Guaño Clavijo con documento de identificación N°1719909838 manifiesto que:

Soy la autora y responsable del presente trabajo; y, autorizo a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de maneratotal o parcial el presente trabajo de titulación.

Quito, 08 de agosto del año 2024

Atentamente,

Alisson Tamara Guaño Clavijo
1719909838

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Yo, Alisson Tamara Guaño Clavijo con documento de identificación No. 1719909838, expreso mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del Proyecto Técnico: "Bot de servicio para cumpleaños con reconocimiento facial e interacción humano máquina para el restaurante La Taberna", el cual ha sido desarrollado para optar por el título de: Ingeniero en Electrónica y Automatización, enla Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que hago la entrega del trabajo final en formato digital a la Biblioteca de la Universidad PolitécnicaSalesiana.

Quito, 08 de agosto del año 2024

Atentamente,

Alisson Tamara Guaño Clavijo
1719909838

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Andrés Sebastián Calero Calero con documento de identificación N° 1719252346, docente de la Universidad Politécnica Salesiana, declaro quebajo mi tutoría fue desarrollado el trabajo de titulación: BOT DE SERVICIO PARA CUMPLEAÑOS CON RECONOCIMIENTO FACIAL E INTERACCIÓN HUMANO MÁQUINA PARA EL RESTAURANTE LA TABERNA, realizado por Alisson Tamara Guaño Clavijo con documento de identificación N° 1719909838, obteniendo como resultado final el trabajo de titulaciónbajo la opción Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 08 de agosto del año 2024

Atentamente,

Ing.Andrés Sebastián Calero Calero MsC
1719252346

ÍNDICE DE CONTENIDOS

RESUMEN X

ABSTRACT XI

INTRODUCCIÓN XII

CAPÍTULO 1 1

1. ANTECEDENTES 1
   1.1. PROBLEMA 1
      1.1.1. Antecedentes 1
      1.1.2. Importancia y alcances 3
      1.1.3. Delimitación 4
   1.2. Objetivos del proyecto 4
      1.2.1. Objetivo general 4
      1.2.2. Objetivos específicos 4

CAPÍTULO 2 5

2. FUNDAMENTOS TEÓRICOS 5
   2.1. Bots y la interacción humana-máquina 5
   2.2. Inteligencia Artificial 5
   2.3. API 6
   2.4. Reconocimiento Facial 8
   2.5. Interfaz Gráfica 9
   2.6. Jetson Orin Nano 9
   2.7. Python 11
   2.8. OpenCV 12
   2.9. gTTS 12
   2.10. Whisper OpenAI 13
   2.11. Librería Threading 15

CAPITULO 3 17

3. MARCO METODOLÓGICO 17
   3.1. Propuesta del proyecto 17
   3.2. Estructura del bot 19
   3.2.1. Estructura 19
   3.2.2. UPS 21
   3.2.3. Iluminación 23

3.2.4. Sonido y micrófono 23
3.2.5. Pantallas 25
3.3. Programación del bot 26
3.3.1. Librería para la interfaz gráfica 26
3.3.2. Librería de reconocimiento facial 28
3.3.3. Librería de texto a voz y reproducción de canciones. 29
3.3.4. Librería de voz a texto 30
3.3.5. Programación 30
CAPITULO 4 33
4. PRUEBAS Y RESULTADOS 33
4.1. Pruebas con la Interfaz de Usuario 33
4.2. Pruebas de registro de Usuarios 37
4.3. Pruebas de reconocimiento de Usuario 40
CONCLUSIONES 45
RECOMENDACIONES 47
REFERENCIAS BIBLIOGRAFICAS 48
ANEXOS 53

ÍNDICE DE TABLAS

Tabla 3.2.2.1: Tabla de consumo.....22
Tabla 4.2.1: Intento de registro.....40
Tabla 4.3.1: Tabla de reconocimiento facial 41
Tabla 4.3.2:Tabla de funcionamiento de las canciones 43
Tabla 4.3.3:Tabla de selección de canciones.....43

ÍNDICE DE FIGURAS

Figura 2.10.1: Lista con las ubicaciones en el servidor. 14
Figura 2.10.2:Código para la transcripción de la voz a texto. 15
Figura 3.1.1:Diagrama de flujo 18
Figura 3.1.2: Diagrama de interacción humano-máquina 19
Figura 3.2.1.1:Idea de la distribución de la base 20
Figura 3.2.1.2:Estructura del robot 20
Figura 3.2.1.3: Shaya, bot asistente de cumpleaños 21
Figura 3.2.2.1: UPS Powest 22
Figura 3.2.3.1: Tira led Inteligente RGB 23
Figura 3.2.4.1: Módulo Bluetooth 24
Figura 3.2.4.2: Parlante de 6W 24
Figura 3.2.4.3: Micrófono USB 25
Figura 3.2.5.1: Diagrama de conexión entre el panel y la Raspberry Pico 25
Figura 3.2.5.2: Rostros diseñados para el panel LED 26
Figura 3.3.1.1: Imagen base para la creación de la ventana de canciones de cumpleaños. 27
Figura 3.3.1.2: Diseño de botones 27
Figura 3.3.2.1: Fragmento del código encargado de codificar imágenes 29
Figura 3.3.2.2:Código para la codificación de rostros de una cámara y la comparación con las imágenes guardadas 29
Figura 3.3.5.1: Código de la interfaz 30
Figura 3.3.5.2:Código de la ventana secundaria. 31
Figura 3.3.5.3:Cñodigo para cerrar la ventana y liberar la cámara. 31
Figura 3.3.5.4: Código para la transcripción de audio a texto 32
Figura 4.1.1: Pantalla principal 33
Figura 4.1.2: Imagen del resultado cuando se intenta identificar un rostro desconocido 34
Figura 4.1.3: Ventana de Registro 35
Figura 4.1.4: Pantalla de Canciones 36
Figura 4.1.5: Pantalla previa a las recomendaciones 36
Figura 4.2.1: Foto realizada cuando se hacia el proceso de registro con una persona desconocida. 37
Figura 4.2.2: Captura realizada de la distribución de las carpetas en la Jetson Orin Nano

.....38
Figura 4.2.3: Archivo txt que contiene los datos de la persona registrada 39
Figura 4.2.4: Captura de pantalla de la carpeta que contiene las fotos de las personas registradas.....39
Figura 4.3.1: Captura de las carpetas que se necesitan para el reconocimiento facial 40
Figura 4.3.2: Fotos de las personas registradas y los códigos respectivos a cada uno. ...41
Figura 4.3.3: Carpeta de datos.....42

RESUMEN

Como búsqueda de innovación y adaptación a la era digital, se ha planteado la idea de la integración de las nuevas tecnologías en restaurantes de Ecuador, ya que actualmente no se puede ver implementada en negocios de este estilo. El uso de IA (Inteligencia Artificial) como complemento a las funcionalidades y servicios que ya ofrece el establecimiento permite crear una experiencia más relevante para el cliente, además de beneficiar en muchos aspectos al negocio. El uso de herramienta que permita potenciar y personalizar la experiencia de los comensales sin tener que sobrecargar la demanda laboral de los trabajadores, es lo que motiva a empresarios hacer uso de este tipo de herramientas.

Se diseña un sistema que implementa primeramente un programa realizado en el lenguaje Python con una Nvidia Jetson Orin Nano como controlador, la cual contiene un modelo de reconocimiento facial para almacenar datos de clientes que posteriormente serán detectados. Se incorpora también scripts que permiten al bot, que previamente fue diseñado, interactuar con el usuario mediante la visualización de una interfaz gráfica en una pantalla, permitiendo también el uso de la voz para dar comandos y la capacidad de comunicarse bidireccionalmente con el bot, todo esto diseñado en una estructura física que asemeje un asistente de camarero.

Mediante la conexión con OpenAI se obtuvo la transcripción de la voz a texto y mediante una herramienta de Google se consiguió convertir el texto a voz.

Palabras clave: Inteligencia Artificial, reconocimiento facial, bot, interacción humano máquina.

ABSTRACT

As a search for innovation and adaptation to the digital era, the idea of integrating new technologies in restaurants in Ecuador has been raised, since it cannot currently be seen implemented in businesses of this style. The use of AI (Artificial Intelligence) as a complement to the functionalities and services already offered by the establishment allows creating a more relevant experiences for costumer, in addition to benefiting the business in many aspects. The use of a tool that allows the diners experience to be enhanced and personalized without having to overload the labor demand of workers is what motivates business owners to use this type of tools.

A system is designed that first implements a program made in the Python language with an Nvidia Jetson Orin Nano as a controller, which contains a facial recognition moder to store customer data will later be detected. Scripts are also incorporated that allow the bot, which was previously designed, to interact with the user by displaying a graphical interface on a screen, also allowing the use if voice to give commands an the ability to communicate bidirectionally with the bot, all of it. This designed in a physical structure that resembles a waiter's assistant.

By connecting with OpenAI, the transcription of the voice to text was obtained and using a Google tool the text was converted to speech.

Key Words: Artificial Intelligence, facial recognition, bot, human-machine interaction.

INTRODUCCIÓN

La integración de nuevas tecnologías en la cotidianidad de la vida se ha vuelto cada vez más relevante y notorio. La manera de relacionase con otras personas y con el entorno ha ido cambiando por el desarrollo constante de la era digital, la cual se ha visto impulsada por el acceso a tecnología que actualmente se tiene.

Todo tipo de negocio, tanto del sector secundario como terciario, se han visto en la necesidad de adaptarse a estos avances para continuar siendo competentes. Las empresas como restaurantes, parte fundamental para el sector terciario, busca maneras de innovar con el fin de brindar un mejor servicio, y por ende una mayor satisfacción a los clientes. La Inteligencia Artificial es una de las herramientas que pueden estar más presentes en un restaurante, complementando y facilitando las labores que hasta el momento tiene.

El uso de la Inteligencia Artificial en un restaurante dependiendo de donde se aplique, permite obtener una mayor eficiencia, optimizar gestiones, reducir errores y en este caso, mejorar la experiencia del cliente mediante una atención más personalizada.

Esta fue la principal motivación que se tuvo para implementar un sistema de inteligencia artificial que permita personalizar la experiencia de los comensales, en especial el día del cumpleaños, reconociendo a la persona y dándole opciones a escoger, con el fin conseguir eventos diferentes a otros dependiendo de las elecciones que el usuario haga.

El desarrollo de este proyecto se centrar en cuatro apartados, siendo el primero la problemática, los antecedentes, la importancia y los objetivos. En el siguiente apartado se redacta el marco teórico. En tercer lugar, se indica de forma detallada, el desarrollo del proyecto. Se continua con mostrando las pruebas realizadas y los resultados, y se finaliza con las conclusiones y recomendaciones.

CAPÍTULO 1

1. ANTECEDENTES

1.1. PROBLEMA

1.1.1. Antecedentes

Los modelos de negocios que actualmente existen son diferentes a los que había años atrás. La era digital ha provocado que las empresas se vean en la obligación de innovar constantemente e incluir en sus procesos o servicios, recursos tecnológicos para seguir siendo competentes en el mercado, demostrando así el valor añadido que como comercio tienen y sobretodo demostrando a los clientes el aporte adicional que los distingue de los demás de la misma categoría (David L. Rogers, 221 C.E.).

Se han registrado en numerosos países el uso de inteligencia artificial y robots que complementan los servicios que ofrecen los restaurantes, incluso colaboran en procesos de producciones, por ejemplo robots baristas, robots que realicen determinadas comidas o asistentes con inteligencia artificial encargados de analizar al cliente y dar recomendaciones en base a las preferencias anteriormente obtenidas (Neilenko & Rusavska, 2021), tareas que puede llegar a ser más complicada hacerla por un ser humano.

Cabe recalcar que el sector económico encargado de la atención al público es extremadamente volátil y depende en gran parte al gusto y preferencia del consumidor, y dado que la tecnología es una herramienta que está entrando con fuerza en el mercado, es natural entender que sea necesario adaptarse e incorporar dichas herramientas para el trato de los consumidores (Peter D. Nyheim, 2019). Señalando lo anteriormente expuesto, y conociendo la preocupación de ciudadanos de todo el mundo en la disminución de empleo por culpa de la integración de robots o tecnología que complementen los servicios que tiene un restaurante, es necesario aclarar que hoy por hoy la intención de los empresarios en innovar es conseguir la integración entre el trato humano y el avance futurista que proporciona la tecnología (Martinez Jaume, 2019), pero siempre respetando el valor humano que proporciona el trato de una persona. Estudios realizados arrojan como conclusión que a pesar de la atracción visual que genera un robot en un establecimiento, la población actual aún no está preparada para experimentar la interacción netamente con robots (Kaur et al., 2023).

Restaurantes de todo el mundo sobre todo en Asia, Europa y Estados Unidos dan testimonios de como la tecnología ha ayudado en los procesos, además de ser un gran atrayente para los clientes lo que se ha reflejado en mejores ventas o menores perdidas. Destacan sobretodo la influencia de las redes sociales. Los robots, en este caso particular, llaman mucho la atención de los comensales, los cuales destacan el agrado de verlos y además de la interacción que consiguen cuando los saludan o se despiden de ellos, para muchos resultan llamativo y curioso, y por consecuencia comparten su experiencia promocionando el negocio gratuitamente, y ayudando al mismo a crecer ("Los robots meseros llegan para quedarse: ¿Cómo cambiará la industria de restaurantes?," 2023).

Con la IA pasa exactamente lo mismo, el sector hotelero comprende los beneficios de la aplicación en varios ámbitos, desde la atención al cliente hasta BigData pasando por el reconocimiento de voz y de rostros (Rock, 2024). Empresas de desarrollo se han dado cuenta de la necesidad a futuro de este tipo de herramientas, obteniendo como resultado el acceso a API's de Google o OpenAI, por ejemplo.

OpenAI, es la empresa detrás del desarrollo de ChatGPT, destacada en el ámbito de Inteligencia Artificial, proporcionando recursos a desarrolladores para acceder a sus herramientas e integrándolas en varios proyectos, el desarrollo que ellos tienen sus aplicaciones es de código abierto asi que eso promueve el desarrollo ("Open AI lanza la API de ChatGPT y de Whisper para poder integrar estas herramientas en las 'apps' de los desarrolladores," 2023).

Whisper es un sistema ASR, las siglas en ingles de un sistema de reconocimiento automático de voz, el cual está basado en un modelo Transformer codificador-decodificador (Radford et al., 2022). Es indispensable saber y conocer, ya que es precisamente la implementación de este tipo API lo que permite que un bot pueda conseguir la interacción humano- máquina. Es fundamental, conseguir que una persona sea capaz de hablar con normalidad y que la maquina consiga comprender instrucciones, palabras o sentencias(Rusan & Mocanu, 2022), por lo tanto, es claro que la herramienta que proporciona OpenAI es completamente útil para conseguir ese cometido, ya que con el modelo Transformer que utiliza, está en la capacidad de volver a texto a texto las palabras que expresa un ser humano, para que mediante programación se pueda conseguir utilizarlo para realizar una acción determinada.

El objetivo principal cuando se quiere implementar un sistema que pueda interactuar con un humano, es la facilidad que tenga el usuario para hacer las acciones para lo cual está programada la máquina. Por lo tanto, es indispensable incluir herramientas que ayuden a esta interacción, como interfaces gráficas y que la máquina "hable".

Por este motivo, para crear un sistema que sea amigable con el usuario es necesario ajustarse a sus preferencias, y siempre pensando en su comodidad. Por lo tanto, es indispensable incluir un sistema de voz que permita al bot comunicar la acción que espera de una persona, un sistema que permita al usuario expresarse sin que amerite un exceso de trabajo, y finalmente una interfaz sencilla que muestre lo suficiente sin sobrecargar de información, puede hacer que la persona que interactúa se sienta comoda y pueda aceptar con agrado herramientas tecnológicas.

1.1.2. Importancia y alcances

Dado los avances en IA y robotica que está sucediendo alrededor del mundo, no era de extrañar que alcanzara el sector de los servicios. En Europa y Asia, está incorporando este tipo de tecnología para potenciar la experiencia que viven los consumidores, y sobre todo como herramienta de innovación y con ello acompaña la posibilidad de un crecimiento en la productividad.

La importancia de esto en este sector es la potencialidad y el beneficio que tiene para el establecimiento, ya que las tareas más repetitivas y monótonas son las que podrían ser ocupadas por robots e IA, beneficiando a los trabajadores ya que estos podrán hacerse cargo de otro tipo de ocupaciones que requieran de habilidades empáticas, características que por el momento no es posible incorporar en robots(Blöcher & Alt, 2021).

Ya que alrededor del mundo cada día existen más países que se suman al cambio y la experimentación de este tipo de recursos, es crucial que, en el continente americano, en específico, los países del sur empiecen analizar el impacto social que esto puede tener en la ciudadanía latinoamericana, como es su comportamiento ante el avance de la inteligencia artificial y de la robotica en el día a día.

1.1.3. Delimitación

El proyecto planteado es un sistema de interacción con el humano, el cual, mediante el almacenamiento de datos como el rostro, el nombre y la fecha de cumpleaños de diferentes personas, debe tener la capacidad de reconocerlas y haciendo una comparativa entre todos los rostros, determinar cuál de todos se asemeja más a la persona a reconocer. Adicional a eso debe comparar la fecha actual con la fecha de cumpleaños y realizar una acción dependiendo de si esa condición se cumple o no con el fin de conseguir una interacción humano-máquina se añadirá librerias que permita que el bot se exprese y que de igual manera entienda instrucciones o palabas dichas por una persona.

Por lo tanto, el proyecto va dirigido al restaurante la Taberna y a todos los clientes que lo visitan, ya que se pretende que el bot consiga personalizar la experiencia de cumpleaños de los consumidores sin la necesidad de aumentar las labores de los actuales empleados. Todo esto con el fin de que las nuevas tecnologías vayan de a poco integrándose en el servicio al cliente, esperando como resultado un mayor atractivo al restaurante y animando al mismo a involucrar las nuevas tecnologías en sus procesos y servicios.

1.2. Objetivos del proyecto

1.2.1. Objetivo general

- Desarrollar un bot asistente para cumpleaños para el restaurante “La Taberna” mediante el reconocimiento facial y la interacción humano máquina

1.2.2. Objetivos específicos

- Investigar el funcionamiento e implementación del reconocimiento facial y como se consigue la interacción entre humano y un bot.
- Modificar atributos de un modelo de identificación de objetos y de rostros mediante técnicas de aprendizaje
- Diseñar la arquitectura de comunicación entre bot y la nube (OpenAI) para la interacción humano-maquina mediante protocolos de comunicación web.
- Desarrollar una interfaz humano-máquina para el ingreso de parámetros de comportamiento del bot mediante hardware y software especializado.
- Verificar los skills de asistente de cumpleaños e interacción humana del bot para su validación mediante pruebas de campo.

CAPÍTULO 2

2. FUNDAMENTOS TEÓRICOS

2.1. Bots y la interacción humana-máquina

Una de las características más notorias entre los seres humanos es la interacción que se genera entre individuos, es decir, la vida social que existe entre personas. Con el fin de conseguir una comunicación más natural entre máquinas y personas, se han desarrollados herramientas con el objetivo de replicar aspectos para lograr que la interacción sea más cómoda y sencilla.

Conseguir la comunicación entre individuos y las máquinas permite a la sociedad ser más receptiva ante la interacción con las nuevas tecnologías y la inteligencia Artificial. Es necesario integrar la tecnología en la sociedad, ya que, con los últimos avances(Awais et al., 2020), y observando como esta se desarrolla, es inevitable pensar que, con el paso de los años, se incorporaran más en la vida diaria de los individuos.

El termino bot se refiere a una interfaz generalmente inteligente, con el cual un usuario tendrá que interactuar principalmente en forma de conversación. La función más destacada de este tipo de programas es que son capaces de realizar tareas que pueden ser tediosas, repetitivas o que no necesite de habilidades sociales. Gracias a los avances existentes en cuanto inteligencia artificial, estos bots cada vez son más comunes en diferentes campos, ya que tienen funcionalidades aprovechables para la industria y la vida diaria.(Lebeuf et al., 2018).

2.2. Inteligencia Artificial

La Inteligencia Artificial es una herramienta computacional que se encarga de crear sistemas inteligentes que sean capaces de trabajar con humanos o como ellos. Entre las múltiples herramientas que se han realizado se encuentra la visión por computadora, el procesamiento del lenguaje natural, los bots y la robótica, etc. El área de aplicación de este tipo de herramientas es completamente interdisciplinaria ya que se puede incluir desde las finanzas o el diseño, hasta en la medicina o la ingeniería. En otras palabras, este recurso es tan versátil y adaptable que dependiendo de la funcionalidad que se le dé, puede integrarse sin problemas en cualquier ámbito que se requiera(A, 2023).

Un sector bastante destacado en este documento es el uso en restauraste y hotelería, el cual permite mejorar la atención al cliente, e incluso automatizar procesos que involucren en la experiencia que puedan tener los personas que visitan los establecimientos.

A pesar de que la Inteligencia Artificial es una herramienta que está permitiendo un gran avance tecnológico en esta era digital, existe la contraparte, que intenta utilizarla para fines maliciosos o perjudiciales, como ataques a la integridad, resultados no deseados, manipulación en el mercado de valores, desinformación, phishing (fraude cibernético para engañar a la gente y obtener datos confidenciales).

El phishing es un bot que puede interactuar de forma muy natural con el ser humano, proporcionando datos personales y muy específicos con el fin de que la persona atacada proporcione datos bancarios o contraseñas de cuentas personales. Los medios que se usa para este tipo de fraudes pueden ser extremadamente variables, desde llamadas telefónicas hasta mensajes por redes sociales.

Deepfake se refiere a la modificación de imágenes o videos mediante inteligencia artificial con fines maliciosos. Aunque al inicio puede sonar ventajoso el realizar modificación, ciberdelincuentes han encontrado la forma de aprovecharlo para objetivos perjudiciales. Esto se debe a que estas modificaciones, puede provocar que en un video aparezca alguien diciendo cosas que realmente no han dicho, o mostrando imágenes que realmente no han sucedido. Inclusive han logrado hacer este tipo de falsificaciones en vivo simulando que una persona se ha reunido o ha hecho una llamada en vivo con alguien totalmente diferente sin darse cuenta(Blauth et al., 2022).

2.3. API

API son las siglas en ingles de interfaz de programación de aplicaciones. Es un software que permite que exista una comunicación entre un servidor y el usuario. Uno de los beneficios que tiene utilizar una API es el uso de una herramienta que ya creada y utilizarla a conveniencia del programador. Es decir que un proveedor permitirá que utilicen una herramienta y un consumidor la ocupara para la aplicación que este en desarrollo(De, 2017).

Existen API’s sobre temas distintos, y una de las plataformas que proporciona este tipo de servicios es OpenAi, además permite observar las múltiples herramientas que tienen, entre ellas se encuentra, la que se utiliza en el presente proyecto, encargada de pasar el audio a texto, también tiene API’s que se encarga de pasar un texto a voz o de traducir un texto, normalmente al inglés (OpenAi, n.d.). se puede observar en la Figura 2.3 como es la conexión entre la aplicación que la usa, en el caso de este proyecto, los programas de Python, con las empresas que ofrecen las API.

Figura 2.3: Servicios API

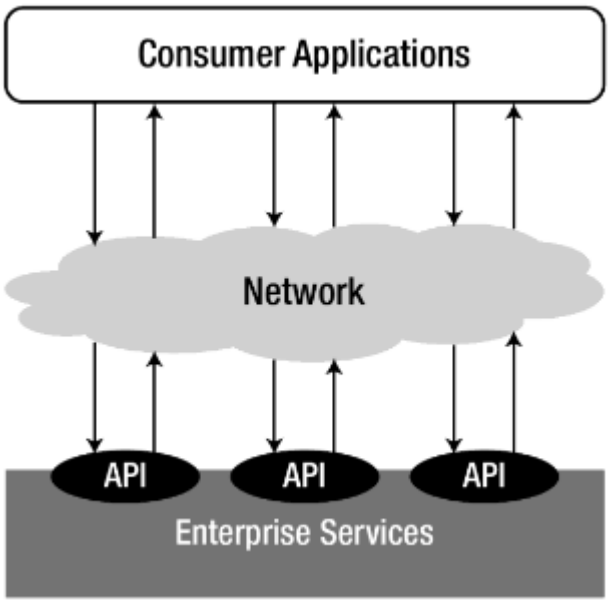


Diagrama de la interfaz de conexión entre una API con el consumo de una aplicación realizada por un desarrollador. Fuente: OpenAI (“Open AI lanza la API de ChatGPT y de Whisper para poder integrar estas herramientas en las ‘apps’ de los desarrolladores,” 2023)

El principal motivo para utilizar este tipo de recursos es aprovechar las herramientas que servidores existentes proporcionen y ocuparlas para el beneficio de la aplicación en proceso de desarrollo, por lo tanto es compresible que usuarios de todo el mundo decidan aprovechar y ocupar en sus sistemas API’s ya desarrolladas, en la actualidad existen numerosas API’s disponibles, tanto para aplicar en robots, como asistentes, o para el manejo de redes sociales o empresas(Gao et al., 2022) incluso de diferentes empresas, no solo de OpenAI.

2.4.Reconocimiento Facial

Los seres humanos tienen la capacidad de observar un rostro, recordarlo, y en caso de encontrarlo en otro lugar identificarlo, es un proceso que se realiza de forma inmediata sin ser prácticamente conscientes. Sin embargo, conseguir eso en una máquina, es un proceso más laborioso ya que requiere que se comprenda desde la base cómo funciona el cerebro humano. Con el fin de replicar el mismo proceso, se ha generado, mediante inteligencia artificial, algoritmos de identificación de rostros, que se encargan de extraer los rasgos característicos, y compararlos con imágenes previamente almacenadas y etiquetadas, y como resultado entregar aquella que se asemeje más al rostro por reconocer (Li et al., 2020).

En la actualidad los avances de este tipo de tecnología han sido enormes, a tal punto que el que las aplicaciones principales están inmersas en muchos campos como en el marketing. En este campo el reconocimiento facial está siendo de gran ayuda para la personalización de campañas publicitarias ya que es posible mostrar el anuncio dependiendo de las emociones, el estado de ánimo, el género y sobre todo dependiendo de la edad, eso ayuda enormemente a segmentar correctamente la población, permitiendo mostrar anuncios que realmente puedan llegar al público objetivo.(Vlačić et al., 2021)

Como toda herramienta que se desarrolla, existe un grupo de personas, que buscan utilizarlas para fines egoístas y perjudiciales, en este caso el reconocimiento facial al necesitar una base de datos con que comparar, se crea una gran vulnerabilidad de la cual las empresas se encargan de protegerse de gran manera, ya que al hackear la base de datos, se puede modificar esta información y el reconocimiento facial dejaría de funcionar correctamente, y permitiría que otras personas ingresara a la información o incluso que pueda extraerla(Latika Kharb & Deepak Chahal, 2023).

Una gran preocupación que puede tener la gente ante la inteligencia artificial es la vigilancia y el seguimiento, ya que con la información suficiente, es posible rastrear una persona con facilidad. En determinadas condiciones puede ser beneficioso por ejemplo en secuestros o trata de blancas, ya que se podría localizar a las víctimas fácilmente, sin embargo, el rastreo es un término que a muchas personas puede preocupar incluso en manos equivocadas puede llevar a consecuencias desagradables(Lai & Patrick Rau, 2021).

El tener la información en la nube, es extremadamente ventajoso y cómodo para todo el mundo porque eso implica que se tiene más facilidades para ingresar a nuestra propia información sin dificultades, sin embargo, es completamente vulnerable. Esto puede causar innumerables delitos y sobre todo daño a las personas a las cuales se les roba la información. Con ese acceso gente indebida puede hacer uso, para fraudes, estafas, entre otros, y no asumir culpa de ningún delito.

2.5.Interfaz Gráfica

Existen numerosas herramientas que sirven para hacer interfaces gráficas, la elección de cual ocupar depende de los requisitos que necesite la librería, el lenguaje de programación que se ocupe, de las preferencias del programador y de la apariencia final que se quiera conseguir. En el presente proyecto se utilizó la librería de Tkinter por su facilidad al utilizar los diferentes widgets que contiene la librería. Además de ser una librería que viene instalada por defecto con Python, lo que facilita el acceso a ella, ya que no requiere de la instalación de dependencia adicionales para su funcionamiento. Sin embargo, la simpleza de su programación unido con su limitada posibilidad de configuración de widgets dificulta realizar una interfaz llamativa para el usuario.(Seetha et al., 2023)(Alberto Cuevas Àlvarez, 2018).

2.6.Jetson Orin Nano

El sistema embebido utilizado en este proyecto es una Jetson Orin Nano de 8GB. Posee un rendimiento de 40 TOPS (Unidad que cuantifica el rendimiento de la inteligencia artificial), lo que permite un rápido procesamiento de librerías de IA y de machine learning lo que es recomendable para desarrollar proyectos de robótica, reconocimiento facial, o librerías para transformación de texto a voz y de voz a texto, entre otros (NVIDIA, n.d.-a).

Figura 2.6.1 Kit de Desarrollo Jetson Orin Nano de Nvidia

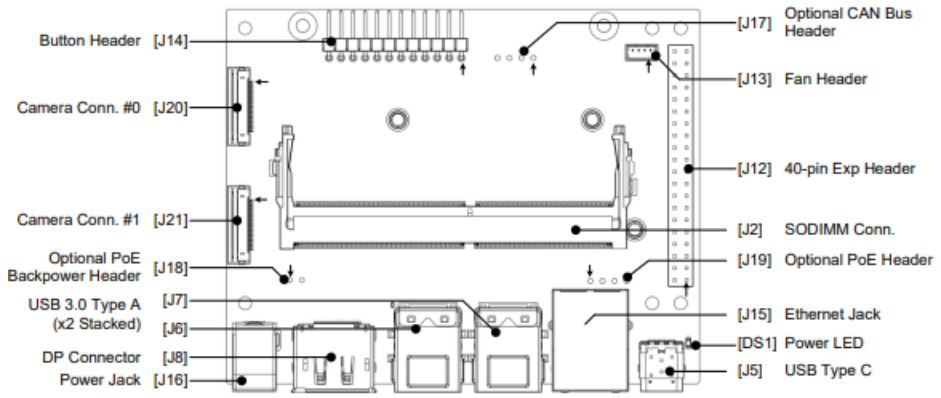
Kit de desarrollo con 40 pines y una GPU Jetson Orin Nano de 8GB de memoria RAM.
Fuente: Nvidia (NVIDIA, n.d.-a)

El sistema operativo que se instala en estas tarjetas está basado en Ubuntu de Linux, pero con una pequeña modificación en la arquitectura, ya que en este caso utiliza NVIDIA Ampere, junto a un optimizador proporcionado por CUDA que permite un mayor procesamiento en inteligencia artificial, machine learning y robótica (NVIDIA, n.d.-b).

El kit de desarrollo contiene conectores comunes en la industria lo cual beneficia enormemente para aumentar las funciones que en si vienen integradas, adicional a eso contiene pines de expansión que pueden funcionar como GPIO (NVIDIA, 2023), los cuales pueden ayudar enormemente en el control de componentes electrónicos, como luces, motores, sensores, etc. Entre las múltiples ventajas que tiene este desarrollo es la completa compatibilidad con objetos diseñados para Raspberry Pi, lo que proporciona más herramientas para aplicar, sin tener que gastar demasiado y encontrando elementos

bastante económicos.

Figura 2.6.2: Vista superior de la placa Jetson Orin Nano



J2	Jetson Orin Nano Conn. (SODIMM, 260-pin)	**J15**	RJ45 Ethernet Socket, 18-pins, RA, Female
J5	USB Type C	**J16**	Power Jack
J6	USB Type A Dual Stacked Connector	**J17**	Optional: CAN Bus Header (1x4, 2.54 mm pitch, RA)
J7	USB Type A Dual Stacked Connector	**J18**	Optional: PoE Backpower Header (1x2, 2.54 mm pitch)
J8	DisplayPort Connector	**J19**	Optional: PoE Header (1x4, 2.54 mm pitch)
J12	40-pin Expansion Header (2x20, 2.54 mm pitch)	**J20**	Camera (#0) Connector (22 pos, 0.5 mm pitch)
J13	Fan Header (4-pin, 1.25 mm pitch)	**J21**	Camera (#1) Connector (22 pos, 0.5 mm pitch)
J14	Button Header (1x12, 2.54 mm pitch, RA)	**DS1**	Power LED (Green)

Explicación de los conectores del kit de desarrollo Fuente: Nvidia (NVIDIA, 2023)

En el caso práctico de este proyecto, se utilizaron precisamente los puertos USB que contiene, para conectar una cámara, un micrófono y la pantalla donde se visualizará la interfaz. Otro puerto utilizado es el DisplayPort el cual es necesario para ocupar la pantalla Touch. De los pines GPIO se ocupan aquellos que sirven para el encendido, el Reset y las luces.

2.7.Python

Python es unos de los lenguajes de alto nivel más utilizados a nivel mundial. Uno de los beneficios que tiene es la facilidad de aplicación y programación, además de tener una extensa comunidad que continuamente desarrolla códigos y los comparte mediante plataformas como GitHub, permitiendo que todo aquel que lo desee contribuya en el desarrollo, con esto se puede conseguir avanzar en Inteligencia artificial con librerias muy buenas para reconocimiento de objetos, procesamiento de lenguaje natural, Machine

Learning, Deep learning, entre otros.

Aunque en este documento, está más centrado precisamente en el reconocimiento de objetos, en especial en el reconocimiento de rostros, y el procesamiento de lenguaje natural tanto TTS (Text-To-Speech) y ASR(Automatic speech Recognition) , por eso el usar este lenguaje de programación es un gran avance en el ámbito de la IA ya que los sistemas se están volviendo compatible con este tipo de lenguaje. Es tanta la utilidad que aporta Python en la inteligencia artificial y la robótica que cada vez más usuarios están volviendo compatibles librerias de otros lenguajes con Python, y en el área de la investigación está obteniendo grandes avances, ya que actualmente están desarrollando múltiples librerias para conseguir un procesamiento rápido y más certero (Raschka et al., 2020).

2.8.OpenCV

OpenCV es una librería de gratuita desarrollada para la creación de software de inteligencia artificial y visión por computadora. Entre las muchas aplicaciones que tiene esta librería esta la clasificación de poses, detección de objetos y clasificación de imágenes.

En el caso práctico de este proyecto se utiliza para la apertura de la cámara que posteriormente se utilizara para el reconocimiento facial, y para la segmentación de los rostros cuando se realiza la foto de la cara de las personas. Esta empresa ha desarrollado bastante el tema de inteligencia artificial, sin embargo, la más eficiente fue la de fase-recognition que se aplicó.

2.9.gTTS

gTTS es una librería de Python que tiene la capacidad de interactuar con la API de Google Translete (Pierre Nicolas Durette, 2024). El funcionamiento de esta libre ya en el ámbito práctico consiste en la escritura de un texto en un código realizado en Python, una vez que se haya ejecutado, la librería creará un archivo mp3 en la ubicación designada por el usuario. Mediante otra librería que permita la reproducción de audios, se hace el llamado

de este audio y empieza a sonar.

La facilidad de instalación, que sea totalmente gratuito y que no se necesite iniciar cuenta en ninguna plataforma es lo que hace que esta aplicación sea interesante instalar. Por supuesto, dependiendo del presupuesto y de los requerimientos que se necesite se pueden utilizar otras, sin embargo, a pesar de que es una librería bastante básica, posee muchas opciones configurables que lo hacer totalmente funciona, como la configuración del acento y del idioma. En este caso, se quería configurar para que "hablara" en español, pero con un acento español, sin embargo, si se deseaba se podía configurar para que su acento fuera el español neutro que se utiliza en Latinoamérica.

Como se puede observar en la página donde se muestra la documentación de esta librería "*gTTS aprovecha la función de voz no documentada de GoogleTranslate y es diferente de Google Cloud Text-To-Speech*" (pndurette, 2024), lo que significa que, aunque tengan el nombre que haga referencia a Google, no es la herramienta de Google Cloud.

2.10. Whisper OpenAI

Para conseguir, que lo que una persona hable, se pueda transcribir a texto, se utilizó una herramienta proporcionada por OpenAI, que se llama Whisper. Este tipo de API's reciben un audio como entrada y lo transforman a texto. En específico, Whisper ha sido entrenada con una gran variedad de idiomas, por lo tanto, es completamente capaz de distinguir con claridad más de 10 idiomas teniendo un error bastante bajo además acepta gran variedad de formatos de entrada, por lo tanto, se puede ajustar a los gustos del desarrollador(OpenAI, 2023).

La instalación de esta herramienta es bastante sencilla, la principal desventaja es que se debe pagar. El precio es de $0.006 por minuto de transcripción(OpenAI, n.d.) y para poder utilizarla se debe generar una clave que única para cada cliente, y es esencial que no se comparta.

Para la comunicación con OpenAI, es necesario conexión a internet, y la programación es sencilla, ya que los códigos base los proporciona OpenAi en su documentación, y la

comunicación entre la plataforma y Python está incorporada en la instalación, así que no requiere ningún paso adicional, lo que sí es indispensable es utilizar una librería que grabe el audio que se quiere transcribir, para que ese parámetro sea pasado a Whisper y pueda devolver la transcripción realizada. Los formatos compatible con esta herramienta son mp3, mp4, mpeg, mpga, m4a, wav, webm.El contener esta amplia variedad de formatos compatibles, permite que la transcripción sea muy accesible ya que no se limita a un solo formato y facilita en el tema de transformaciones de audio, lo cual puede ser tedioso al momento de programar.

Para conseguir que esta transcripción funcione es necesario conectar con el servidor de OpenAI que contiene la información del procesamiento de Whisper. Para esta conexión dentro de la librería, se obtiene la librería urllib, la cual conecta el programa en un sistema embebido con la ubicación en el servidor. En La Figura 2.10.1 se puede observar las direcciones que contiene el programa y que posteriormente se llamaran según sea necesario en el programa fuente.

Figura 2.10.1: Lista con las ubicaciones en el servidor.

```
_MODELS = {
    "tiny.en": "https://openaipublic.azureedge.net/main/whisper/models/d3dd57d32accea0b295c96e26691aa14d8822fac7d9d27d5dc00b4ca2826dd03/tiny.en.pt",
    "tiny": "https://openaipublic.azureedge.net/main/whisper/models/65147644a518d12f04e32d6f3b26facc3f8dd46e5390956a9424a650c0ce22b9/tiny.pt",
    "base.en": "https://openaipublic.azureedge.net/main/whisper/models/25a8566e1d0c1e2231d1c762132cd20e0f96a85d16145c3a00adf5d1ac670ead/base.en.pt",
    "base": "https://openaipublic.azureedge.net/main/whisper/models/ed3a0b6b1c0edf879ad9b11b1af5a0e6ab5db9205f891f668f8b0e6c6326e34e/base.pt",
    "small.en": "https://openaipublic.azureedge.net/main/whisper/models/f953ad0fd29cacd07d5a9eda5624af0f6bcf2258be67c92b79389873d91e0872/small.en.pt",
    "small": "https://openaipublic.azureedge.net/main/whisper/models/9ecf779972d90ba49c06d968637d720dd632c55bbf19d441fb42bf17a411e794/small.pt",
    "medium.en": "https://openaipublic.azureedge.net/main/whisper/models/d7440d1dc186f76616474e0ff0b3b6b879abc9d1a4926b7adfa41db2d497ab4f/medium.en.pt",
    "medium": "https://openaipublic.azureedge.net/main/whisper/models/345ae4da62f9b3d59415adc60127b97c714f32e89e936602e85993674d08dcb1/medium.pt",
    "large-v1": "https://openaipublic.azureedge.net/main/whisper/models/e4b87e7e0bf463eb8e6956e646f1e277e901512310def2c24bf0e11bd3c28e9a/large-v1.pt",
    "large-v2": "https://openaipublic.azureedge.net/main/whisper/models/81f7c96c852ee8fc832187b0132e569d6c3065a3252ed18e56effd0b6a73e524/large-v2.pt",
    "large-v3": "https://openaipublic.azureedge.net/main/whisper/models/e5b1a55b89c1367dacf97e3e19bfd829a01529dbfdeefa8caeb59b3f1b81dadb/large-v3.pt",
    "large": "https://openaipublic.azureedge.net/main/whisper/models/e5b1a55b89c1367dacf97e3e19bfd829a01529dbfdeefa8caeb59b3f1b81dadb/large-v3.pt",
}
```

Lista con las ubicaciones en el servidor.

Cuando se instala la librería de OpenAI viene por defecto instalado todas esas librerias y archivos que componen la aplicación y que ayudan a la conexión con el servidor. Asi cuando se quiere escribir un programa utilizando esta librería, el código es simple como se puede observar en la Figura 2.10.2, pero detrás de todo eso hay una extensa

programación que facilita la aplicación

Figura 2.10.2:Código para la transcripción de la voz a texto.

```
audio_file= open("/home/tammyai/Documents/RoBi/Speech-To-Text/grabacion.wav", "rb")
transcription = openai.audio.transcriptions.create(
  model="whisper-1",
  file=audio_file
)
texto=transcription.text
```

Código utilizado para la transcripción de audio a texto mediante la API de OpenAI.

Fuente: Alisson Tamara Guaño Clavijo

El código utilizado es realmente sencillo, ya que lo principal es hacer el llamado del archivo de audio que se quiere transcribir, para eso se debe poner la ubicación exacta del archivo, posteriormente se especifica el modelo que se quiere utilizar. En cuanto a modelos para escoger, se tiene dos opciones, whisper-1 y whisper-1hd. De las dos opciones, según la documentación encontrada en la página web de OpenAI, es más estable la versión estándar, ya que se ha realizado más pruebas y tiene comprobado que es más fiable la transcripción de esta opción.

Las siglas STT se refiere a una IA que puede pasar el audio a texto, sin embargo, whisper tiene una pequeña diferencia a un STT tradicional. Por lo general un modelo entrenado para la función de Speech-To-Text suele ser entrenado con datos manuales, es decir, para que el modelo entienda, es necesario que se ingrese el audio y el texto para de esta forma la red neuronal entiendo como es el proceso de transcripción, pero en el caso de Whisper, el entrenamiento es mayormente no supervisado, es decir, que con pocos datos de entrenamiento, el modelo debe descubrir los patrones por sí mismo. A este tipo de desarrollo de voz, se le llama reconocimiento automático del Habla, o lo que es lo mismo ASR por sus siglas en ingles.

2.11. Librería Threading

Threading es un módulo que permite trabajar con subprocesos en paralelo e independiente. El uso de esta librería es bastante sencillo, ya que solo es necesario crear una función que cree un proceso, y esa función hacer el llamado desde el hilo de

ejecución(*Threading- Therad -Based Parallelism*, 2024).

Normalmente este tipo de librería se utiliza en la ejecución varias librerías que necesitan mucha ejecución, pero al mismo tiempo se necesita que se visualicen al mismo tiempo, por ejemplo, si se quiere mostrar una interfaz, abrir la cámara de video y escuchar un audio a la vez es necesario hacer uso de este tipo de librerías para que todo se pueda visualizar, si no habría fallos en la ejecución y uno de ellos tardaría en ejecutarse o no se ejecutaría.

CAPITULO 3

3. MARCO METODOLÓGICO

3.1. Propuesta del proyecto

Con el fin de conseguir llamar la atención de clientes hacia algo novedoso y sobretodo práctico para el restaurante La Taberna, se buscó la posibilidad de crear una herramienta que consiguiera interactuar con los comensales para crear un ambiente más amigable. Continuando con esa línea de planteamiento, se decidió limitar la interacción, para que se centrará básicamente en un evento particular que suele ocurrir en el establecimiento, la celebración de cumpleaños.

Para ellos se pretende realizar un Bot que se encargue de proporcionar una dinámica distinta. El proceso consiste en dar una bienvenida. Y sugerir al usuario pulsar "Empezar", en caso de hacerlo, seguirá reconociendo si una persona está cerca y le saludará.

En caso de pulsar "Empezar" comenzará con el reconocimiento facial, en el caso de ser reconocido comparará la fecha actual con la fecha de cumpleaños, si existe coincidencia, permitirá que el usuario pase a la siguiente pantalla donde podrá elegir una canción para que le canten en su cumpleaños, en esta parte tendrá 4 opciones diferentes a elegir. Una vez que se escogió la canción, esta sonará y finalmente dará un agradecimiento y terminará el proceso, haciendo que se regrese a la pantalla inicial.

En caso de pulsar "Empezar" y no coincidir la fecha del cumpleaños con la fecha actual, igualmente saludará a la persona reconocida y se le mostrará otra pantalla diferente, en la cual se le preguntará si desea unas sugerencias de comida, de ser positiva la respuesta, se le preguntará con quien visita el restaurante, las opciones son amigos, familia y pareja, y dependiendo de la elección se mostrará un mensaje u otro. En caso de escoger que no, se le agradecerá la visita y volverá a la página principal.

Finalmente, en caso de que al pulsar "Empezar" no reconozca a la persona, se intentaría reconocer 3 veces más, y si no está registrada, el programa la pasaría a una nueva pantalla

donde mediante la respuesta obtenidas del usuario obtendrá los datos y el rostro para reconocer en una próxima ocasión. Después del registro, igual se compararía si la fecha del cumpleaños concuerda con la actual, y ocurriría los eventos descritos anteriormente dependiendo de si la fecha de cumpleaños coincide con la actual o no, tal y como se muestra en el diagrama de flujo que se encuentro en la Figura 3.1.1.

Figura 3.1.1:Diagrama de flujo

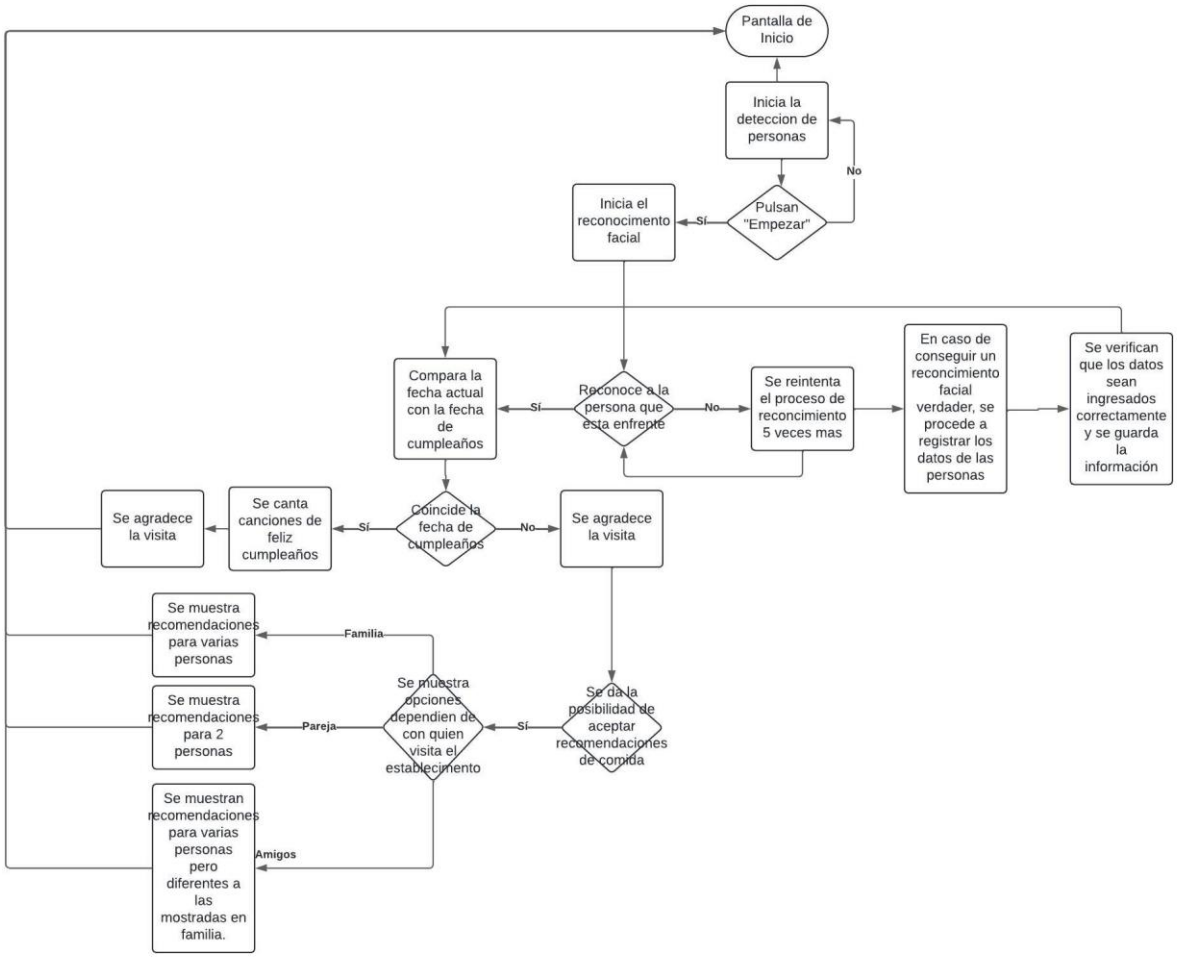
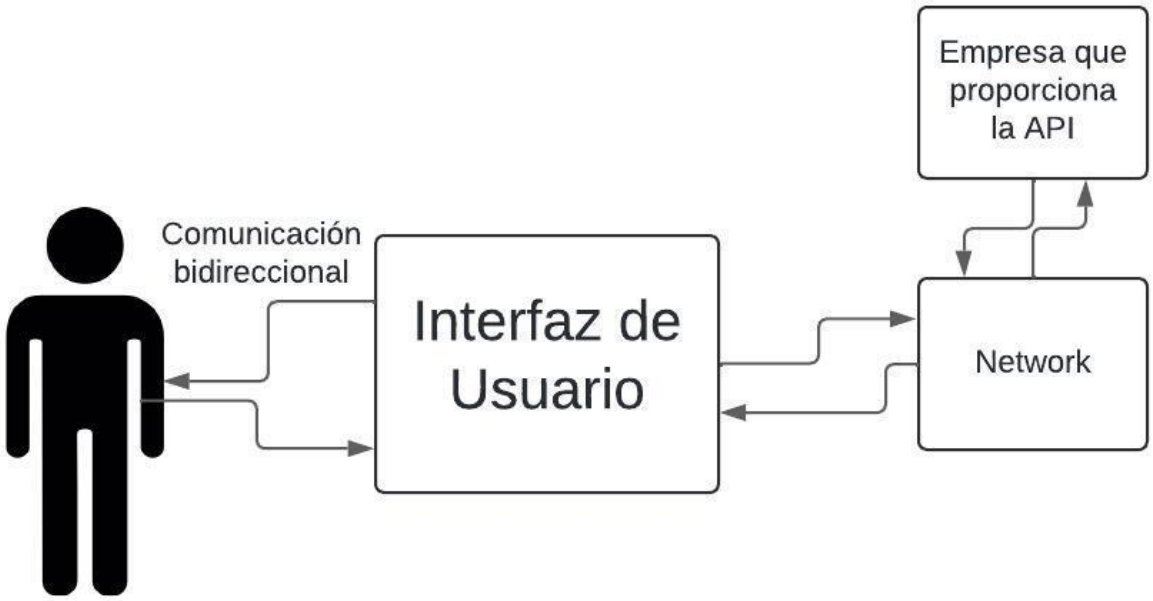


Diagrama explicativo del proceso que realiza el programa en conjunto. Fuente: Alisson Tamara Guaño Clavijo

Una de las grandes ventajas presentes en este sistema es la posibilidad de la comunicación bidireccional, es decir, el robot puede “hablar” para que la persona entienda y realice alguna acción, y de la misma manera, cuando la persona le habla es capaz entender y realizar alguna acción o ejecutar algún evento. Esto no solo aporta en establecer una

comunicación más natural, sino que también tiene la capacidad de que esta comunicación se vuelva más comoda y sencilla.

Figura 3.1.2: Diagrama de interacción humano-máquina



El diagrama muestra cómo se consigue la interacción entre persona que habla, y la respuesta que obtiene por parte del bot. Fuente: Alisson Tamara Guaño Clavijo

3.2.Estructura del bot

3.2.1. Estructura

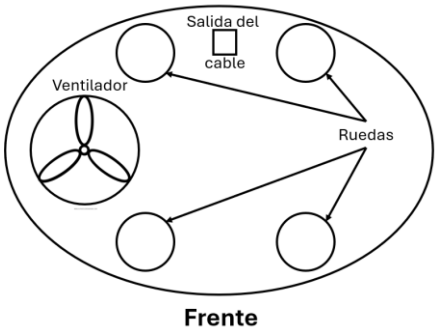
Se realizó una figura para incorporar el bot, ya que de esta forma las personas que interactúan con él se sienten más cómodas que tuvieran una apariencia más humanoide que a uno con aspecto más de máquina. Por este motivo se decidió darle una forma física al bot y no solo hacerlo como un asistente parecido a Alexa u otros de otras marcas.

El gran dilema detrás de realizar esta estructura es principalmente que aspecto debe tener, en que parte debe ir cada uno de los elementos que se le va a incorporar, y otra parte fundamental, que aspectos se debe tener en cuenta para conseguir una estabilidad de la estructura y evitar que esta se caiga.

Con esa premisa en mente, se pensó en realizar una base ancha que mantuviera todo el

peso de la estructura además en la base debía ir unas ruedas para poder mover el bot sin dificultad, pero garantizando estabilidad. Adicional a eso se pensó la distribución de algunos elementos más, como la ventilación y la alimentación del sistema.

Figura 3.2.1.1:Idea de la distribución de la base



Idea de la base con las ruedas para mantener la estabilidad de la estructura y el orificio de ventilación. Fuente: Alisson Tamara Guaño Clavijo

Para los materiales, se buscó uno que fuera resistente pero no muy pesado, y sobretodo que no fuera tan caro para evitar que saliera demasiado costoso el proyecto. Por ello se utilizó fibra de vidrio para moldear la estructura y adaptarla a la idea que se tenía. Con el fin de darle un aspecto más semejante a robots que comerciales ya existente, se utilizó pintura metalizada, con unas capas de barniz, para conseguir un brillo similar.

Figura 3.2.1.2:Estructura del robot

Estructura del robot ya con la pintura metalizada. Fuente: Alisson Tamara Guaño Clavijo

Se colocó un acrílico en los brazos, el cual hace la función de bandeja para poder llevar objetos encima, tales como pasteles o postres. Tanto la base como la bandeja de los brazos contiene tiras LED RGB con el fin de hacerlo visualmente más tractivo y llamativo.

Figura 3.2.1.3: Shaya, bot asistente de cumpleaños

Estructura del bot finalizada con todos los elementos necesarios para su funcionamiento.

Fuente: Alisson Tamara Guaño Clavijo

3.2.2. UPS

Para conseguir el desplazamiento de un robot es necesario que disponga de baterías que le mantengan encendido cuando no esté conectado a la corriente. Para eso es necesario investigar el consumo de cada uno de los elementos y dependiendo de eso escoger una batería que satisfaga el consumo total. En este caso, como la Jetson Orin Nano es una GPU costosa, se busca que el voltaje de entrada estable para evitar sobrecargas y que se pueda quemar. Por este motivo una de las soluciones que mejor se planteó al momento de buscar un equipo que mantuviera encendido el bot aunque no esté conectado es un UPS.

Tabla 3.2.2.1: Tabla de consumo

Equipos	**Voltaje**	**Corriente**	**Consumo**
Kit de desarrollo Jetson Orin Nano	19V	2.37 A	45 W
Tira Led	5V	2A	10 W
Matriz de 64x32 Led	5V	3A	15 W
Ventilador	5V	1A	5 W
Modulo Bluetooth	12V	5A	60 W
		Total	135W

Tabla de consumo total de los elementos que están conectados al UPS

Una de las características que tiene el UPS powest precisamente la estabilidad a la entrada que se busca como medida de protección de la Jetson Orin Nano, adicional a eso tiene una protección contra sobrecargas lo que hace que sea más seguro conectarlo.

Dependiendo de los elementos que se utilice, funcionan a diferentes voltajes, por lo tanto el utilizar un UPS facilita la conexión, ya que no es necesario realizar ningún tipo de conexión que pueda poner en peligro algún elemento, si no que solo se debe conseguir los conectores con los voltajes y corrientes necesarios para cada elemento, y ya que el UPS contiene interruptores en la parte posterior, solo se debe conectar y empieza a funcionar todo correctamente.

Figura 3.2.2.1: UPS Powest

UPS que funciona como batería para cuando se desconecte el bot de la alimentación

Fuente: Google Images (*Powest UPS*, n.d.)

Adicional a eso, se investigó algunas baterías con el fin de encontrar una que satisfaga el consumo que necesita este proyecto, pero el costo llegaba a ser más alto, por lo tanto, términos de eficiencia y monetario la opción más coherente era el ocupar un UPS como

fuente de alimentación.

3.2.3. Iluminación

El aspecto físico de un objeto es muy importante, ya que de eso depende de que la gente se sienta curiosa de conocerlo y observarlo. En base a eso y observando los robots actuales que existen en la industria, se quiso incluir en la estructura una tira led que añadiera atractivo visual al bot.

Figura 3.2.3.1: Tira led Inteligente RGB

Imagen de referencia de la tira led Inteligente que se utilizó para la estructura del bot.
Fuente: Google Images (*Tira LED RGB* , n.d.)

En este caso se buscó una tira Led RGB inteligente que tuviera un controlador incorporado para poder modificar y ajustar a gusto la secuencia de colores que se quería obtener y de esta forma conseguir lo planeado, hacer que la estructura se vuelva llamativa. Adicional a eso se decidió utilizar la misma tira led en la parte superior, en el acrílico que se utilizó como bandeja.

3.2.4. Sonido y micrófono

En cuestión de sonido, la primera configuración que se realizó fue en la Jetson, ya que era necesario otorgarle permisos al bluetooth que venía incluido en la placa y que por defecto estaba desconectado. El motivo de utilizar un módulo bluetooth es la falta de puertos USB de la tarjeta, entre la cámara, el micrófono, la pantalla superior, y la pantalla touch que tiene el pecho, no había espacio para colocar una tarjeta de sonido, y sin esa tarjeta no es posible conectar el plug de unos parlantes. Además, las tarjetas compatibles con la Jetson Orin Nano no eran lo suficiente potentes como para poder utilizarlas con unos parlantes con buen sonido, por lo tanto, la opción más recomendable era utilizar un Módulos Bluetooth.

Figura 3.2.4.1: Módulo Bluetooth

Módulo bluetooth utilizado para la transmisión de audio. Fuente: Google Images(*Módulo Bluetooth*, n.d.)

En el caso de los parlantes se utilizó 2 de 6W, que están conectados al módulo anteriormente mencionado. Se comprobó que la potencia que requerida por los parlantes necesitaba mínimo 12W ya que de utilizar uno más sencillo provocaba que se desconectar continuamente, y para la aplicación que tiene este bot, era contraproducente que eso ocurriera continuamente, pero con el módulo que se colocó y los parlantes de 6Wse obtiene un sonido nítido, y potente, perfecto para este tipo de aplicación.

Figura 3.2.4.2: Parlante de 6W

Parlante utilizado y colocado en la estructura del bot. Fuente: Google Images (*Parlante de 6w*, n.d.)

Finalmente, para completar esta sección es indispensable comentar el micrófono utilizado. La principal característica que tiene este micrófono es que su funcionamiento no es a través de plug si no se conecta a la Jetson mediante USB. Es utilizado para que las personas puedan hablar y se transcriba a texto la información para continuar con el funcionamiento normal del programa.

Figura 3.2.4.3: Micrófono USB

Micrófono utilizado para la transcripción de voz a texto. Fuente: Google Images(*Micrófono Por USB*, n.d.)

3.2.5. Pantallas

Para conseguir una cara más animada que pueda realizar varios gestos, se decidió incorporar un panel de matriz LED de 64x32 pixeles, el cual es controlado por una Raspberry Pico y al ser RGB se puede colocar poner colores diferentes dependiendo de las preferencias del desarrollador. Dicho panel necesita ser alimentado por 5V y casi 3A

Figura 3.2.5.1: Diagrama de conexión entre el panel y la Raspberry Pico

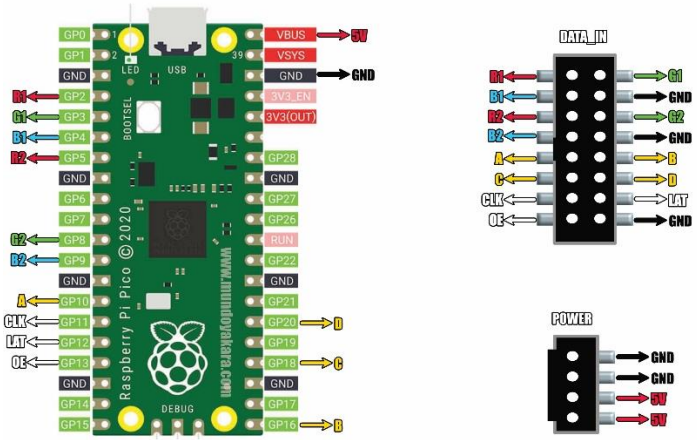
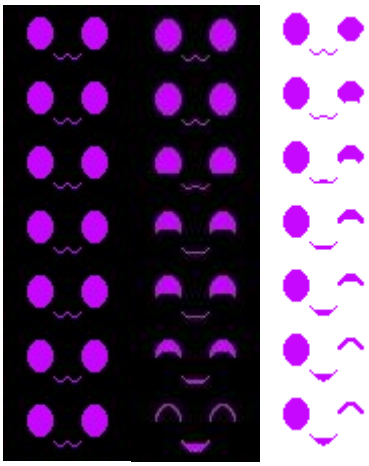


Diagrama de conexiones entre una raspberry Pico y el Panel LED 64x32

Para conseguir la visualización de los rostros en el panel es necesario crear una imagen en formato bmp, para ello previamente se diseñó la animación en páginas web, en especial piskelapp.com y se descargó la imagen completa, se carga en raspberry y se programa para que este se pueda observar en el panel.

Figura 3.2.5.2: Rostros diseñados para el panel LED



Rostros diseñados para que se visualicen el panel LED de64x32. Fuente: Alisson Tamara Guaño Clavijo

Para conseguir una interacción entre el usuario y el bot, se decidió incorporar una pantalla touch de 10,1 pulgadas la cual está pensada para ser utilizada con una raspeberry pero es completamente compatible con Nvidia y con Ubuntu, asi que funciona completamente bien con la Jetson Orin Nano.

3.3.Programación del bot

3.3.1. Librería para la interfaz gráfica

La librería utilizada para el desarrollo de las diferentes páginas que tiene es Tkinter. La principal desventaja que se puede encontrar al utilizar este tipo de librería es los widgets básicos que contiene, ya que puede llegar a ser muy simples y cuadrados, haciendo que la interfaz se vea aburrida y simple. Sin embargo, existen alternativas para poder utilizar los widgets de esta librería y consiguiendo una interfaz más amigable, aunque requiere de más trabajo realizarlo, y es utilizando herramientas de diseño y creando todos los labels y botones como imágenes y sobreponiendo a una imagen base como la creada en canva y que se muestra en la Figura 3.3.1.1.

Figura 3.3.1.1: Imagen base para la creación de la ventana de canciones de cumpleaños.

Imagen utilizada en el fondo de la ventana que corresponde a la elección de la canción de cumpleaños. Fuente: Alisson Tamara Guaño Clavijo

Para la configuración de los botones se realiza el mismo procedimiento. Se debe crear las botones en canva y guardarlos como imagen y al momento de crear el botón se debe incluir la imagen diseñada. Se debe aclarar que, para darle un aspecto un poco más moderno, en este caso se creó dos diseños para el mismo botón, uno que corresponde al aspecto del botón cuando esta desactivado y otro cuando el botón este pulsado o el mouse pase por encima el curso.

Figura 3.3.1.2: Diseño de botones

Imagen de los botones en activación y desactivación. Fuente: Alisson Tamara Guaño Clavijo

En cuanto a los bordes se les pone blanco para evitar que se vea en el fondo y de esta

forma simular que es una interfaz más moderna, y visualmente se consigue eso. En cuanto a los labels, se configura similar, poniendo fondo blanco con letras negras en las partes en las que el fondo es blanco, de esta forma se camufla con el fondo y los bordes cuadrados y simples no se notan

3.3.2. Librería de reconocimiento facial

La librería utilizada se llama Face-Recognition, basada en la biblioteca de dlib. Dlib es una herramienta desarrollada en C++ con algoritmos de aprendizaje automático, dicha librería ha sido adaptada de forma que pueda ser utilizada en el lenguaje de programación Python aunque inicialmente este desarrollada para C, además de eso, es de código abierto, lo que quiere decir que está en constante mejora(Adam Geitgey, 2022).

Para conseguir el reconocimiento facial es necesario primero realizar la codificación de cada una de las imágenes y eso almacena en una lista. Esa codificación es la que se compara con la imagen que se recibe desde la cámara, mediante la librería de OpenCv de Python.
En caso de reconocer el rostro proporcionara las coordenadas en 2D de donde está ubicada el rostro detectado.

Dependiendo de las preferencias de las personas se puede configurar algunos parámetros, como la tolerancia, la cual determina que tan estricto debe ser el modelo de reconocimiento. Esto es necesario configurar en caso de que el reconocimiento no esté funcionando correctamente, y que de muchos errores al clasificar los diferentes rostros(Adam Geitgey, 2020).

El algoritmo de reconocimiento facial sigue unos pasos muy específicos. Lo primero que debe realizar el programa es extraer las características faciales de los rostros ya registrados, como posición de las cejas, ojos, boca. Eso se consigue mediante las líneas de códigos que se muestran en la Figura 3.3.2.1. Se especifica la ubicación de la carpeta donde se pueden encontrar las fotos de los clientes, y mediante el método append() se va añadiendo la codificación de los rostros uno detrás de otro.

Figura 3.3.2.1: Fragmento del código encargado de codificar imágenes

```
for filename in os.listdir("/home/tammyai/Documents/RoBi/Usuarios/Fotos"):
        if filename.endswith(".jpg") or filename.endswith(".png"):
                imagen_conocida=face_recognition.load_image_file(os.path.join("/home/tammyai/Documents/RoBi/Usuarios/Fotos",filename))
                codificacion_cara_conocida=face_recognition.face_encodings(imagen_conocida)[0]
                imagenes.append(codificacion_cara_conocida)
```

Proceso de codificación de los rostros almacenados en una carpeta local. Fuente: Alisson Tamara Guaño Clavijo

El siguiente paso necesario es localizar un rostro en un video transmitido desde una cámara web. Al encontrar el rostro, la librería se encarga de codificarlo igual que se hizo con las imágenes. Una vez que se consiguió se debe comparar los puntos obtenidos con la lista que se creó en el primer paso, todos estos pasos se pueden comprobar en el código de la Figura 3.3.2.2.

Figura 3.3.2.2:Código para la codificación de rostros de una cámara y la comparación con las imágenes guardadas

```
if ret:
        face_location=face_recognition.face_locations(frame)#para detectar un rostro por la camara
        face_encoding=face_recognition.face_encodings(frame,face_location)

        for (top,rigth,bottom,left),face_encoding in zip(face_location,face_encoding):
                matches=face_recognition.compare_faces(imagenes,face_encoding)
```

Código en Python que se encarga de la codificación de rostros en una cámara y la comparación con las imágenes de los clientes registrados. Fuente: Alisson Tamara Guaño Clavijo

En caso de que las imágenes de los rostros y de la cámara coincida, entrará en una condición que ejecutara una función, donde extraerá los datos que posteriormente se utilizará, sin embargo si no coincide, el proceso anteriormente descrito lo intentará 3 veces más.

3.3.3. Librería de texto a voz y reproducción de canciones.

Para generar la voz es necesario el uso de una librería que transforme el texto a audio, para eso se utilizó la librería de gTTS, la cual permitía escribir en el código el mensaje que se quería dar y la misma librería creaba un audio en .mp3, posteriormente se llamaba a través del código al audio y se reproduce. Y de esta forma se consigue que el bot “hable”.

Para la reproducción de la música primeramente se debe almacenar en una carpeta todas

las canciones que se desea que suene, posteriormente se crean los botones que permitirán la preproducción y se vincula con la función que controla la reproducción de la música. En esta ocasión esta función está utilizando la librería de PyGame para hacer la reproducción. Con el fin de evitar hacer un código repetitivo, se utilizó listas para almacenar los nombres de las canciones y la ubicación, de esta forma con una sola función se puede reproducir varias canciones.

3.3.4. Librería de voz a texto

Con la instalación de la librería de Whisper de OpenAI, ya incluye la comunicación entre la nube y la aplicación realizada en Python, sin embargo si es necesario generar la clave que se utiliza para verificar la conexión. Para ellos es necesario, ingresar los datos bancarios de una tarjeta y proporcionar un presupuesto para que la API pueda empezar a funciona.

Antes de hacer la ejecución de esta librería es necesario, crear un código que permita grabar las respuestas por voz de la persona, eso se debe almacenar en un lugar conocido, ya que esa misma dirección se tiene que ingresar en el modelo de ASR que va a transcribir la voz a texto

3.3.5. Programación

Para conseguir la interfaz y utilizando la librería de Tkinter, para ello es necesario tener una ventana base como se ve en la imagen de la Figura 3.3.5.1. El método tk.Tk provoca que la ventana se comporte como ventana principal, en otras palabras, esa es la ventana base de todo la aplicación, las demás ventanas deben configurarse para que se creen encima de esta o a continuación de esta. En una aplicación, solo se puede tener una ventana Principal, en caso de intentar abrir otra ventana principal entraría en conflicto y no funcionaría las interfaces, por lo tanto es un aspecto que se debe tener mucho en cuenta ya que siempre se debe realizar una ventana principal y múltiples ventanas secundarias

Figura 3.3.5.1: Código de la interfaz

```
class FormularioMaestroDesign(tk.Tk):

        def __init__(self):
                super().__init__()
```

Ventana principal realizada en tkinter. Fuente: Alisson Tamara Guaño Clavijo

Para las ventanas secundarias, la declaración es similar, sin embargo en vez de poner el

método tk.Tk se utiliza el método tk.Toplevel, como esta en la Figura 3.3.5.2. este método lo que hace es llamar a la librería de Tkinter y crear una ventana nueva encima de la principal.

Figura 3.3.5.2:Código de la ventana secundaria.

```
class ReconocimientoDeClientes(tk.Toplevel):

        def __init__(self):
                super().__init__()
```

Ventana secundaria que se utiliza para el resto de las ventanas. Fuente: Alisson Tamara Guaño Clavijo

De esta forma se consigue tener una ventana inicial, la cual siempre está abierta, y las ventanas secundarias llamadas por la ventana principal u otras secundarias, y que inmediatamente cuando terminan su ejecución se cierran. Sobre todo en esta aplicación es fundamental tener en cuenta las herramientas de hardware que utiliza el programa, ya que si no se cierra correctamente algunos elementos puede entrar en conflicto, en especial la cámara, por lo tanto, en el ejemplo especifico de la cámara, es indispensable, liberar la cámara antes de pasar a la siguiente ventana, como se realiza en la Figura 3.3.5.3. Entonces el orden es, mediante la sentencia self.camera.release(), liberar la cámara, abrir la nueva ventana que se quiere visualizar, y cerrar la ventana en la que se encuentra actualmente.

Figura 3.3.5.3:Cñodigo para cerrar la ventana y liberar la cámara.

```
def pantalla_usuarios(self):
        self.camera.release()
        RegistroUsuarios()
        self.destroy()
```

Código para la interacción entre las diferentes ventanas. Fuente: Alisson Tamara Guaño Clavijo

En el caso de que se quiera pasar información de un código a otro es necesario crear la función pero siempre retornando un dato. Un ejemplo de esto es el código de Grabar, el cual ejecuta la grabación de audio y la transcripción de el audio en texto, como se puede observar, devuelve el texto a la función el cual en el programa se utiliza.

Figura 3.3.5.4: Código para la transcripción de audio a texto

```
audio_file= open("/home/tammyai/Documents/RoBi/Speech-To-Text/grabacion.wav", "rb")
transcription = openai.audio.transcriptions.create(
  model="whisper-1",
  file=audio_file
)
texto=transcription.text

return texto
```

Código utilizado para la transcripción de audio a texto. Fuente: Alisson Tamara Guaño Clavijo

En los anexos se puede apreciar por completo el código y como interactúa cada uno de los códigos entre ellos.

CAPITULO 4

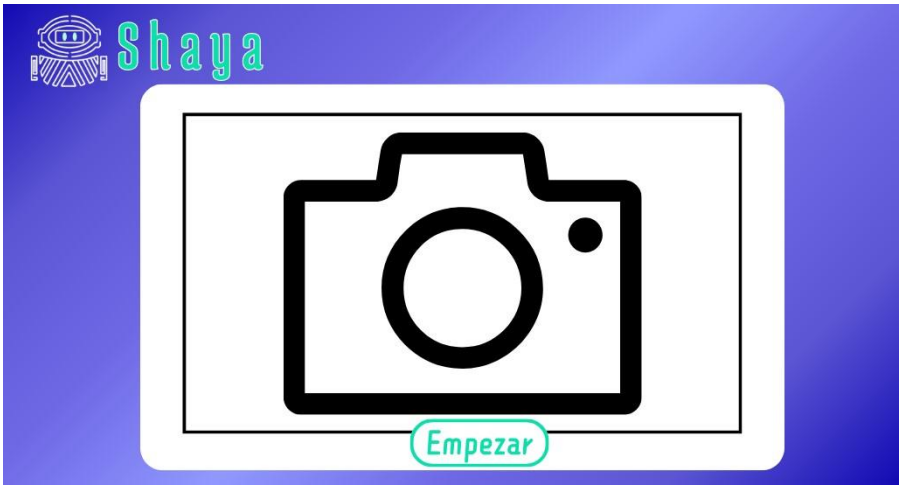
4. PRUEBAS Y RESULTADOS

Para verificar los resultados obtenidos es necesario dividir en diferentes partes. El primer punto es necesario explicar la interfaz de usuario que se creó para el bot y como esta ayuda en la interacción con humanos. El segundo punto explica cómo se realiza el reconocimiento de rostro y cómo funciona el almacenamiento la información de las personas registradas. Por último, se explica cómo se realiza el registro de las personas desconocidas, y como es el proceso de registro.

4.1.Pruebas con la Interfaz de Usuario

Continuando con el diagrama planteado, se ha realizado una pantalla inicial, la cual se encarga de dar un saludo al cliente que se pone en frente de él. Y espera hasta que un cliente pulse el botón de Empezar. Ésta, está vinculada con la siguiente ventana que se encarga del reconocimiento facial. Como mecanismo de protección y para evitar que siga reconociendo infinitamente, al volver nuevamente a la pantalla principal, después de hacer toda la ejecución del programa, se mantendrá todo sin funcionamiento hasta que el encargado pulse el logo que aparece en pantalla.

Figura 4.1.1: Pantalla principal



Pantalla inicial diseñada para el inicio del proceso. Fuente: Alisson Tamara Guaño Clavijo

Una vez que el usuario ha pulsado empezar, el reconocimiento facial comenzará, haciendo que después de unos segundos, mencione el nombre de la persona registrada en caso de tener almacenada su información y compara la fecha de cumpleaños con la fecha actual. En caso de ser igual la fecha actual con la fecha de cumpleaños, se abrirá la pantalla de canciones, caso contrario se verá una ventana de transición que consultará al usuario si quiere algunas sugerencias el cual tendrá que responder con Si o No, según la preferencia del usuario. En caso de ser Desconocida la persona, pasará a la ventana de Registro.

En caso de que la persona sea desconocida se abrirá una ventana nueva la cual mostrará al individuo desconocido, posteriormente se escuchará un saludo y le indicará que su rostro no ha sido registrado con anterioridad.

Figura 4.1.2: Imagen del resultado cuando se intenta identificar un rostro desconocido

Imagen realizada del suceso que ocurre cuando delante de la cámara esta una persona desconocida. Fuente: Alisson Tamara Guaño Clavijo

Abrirá la nueva ventana de Registro, la cual informará a la persona que va a registrar su información. Para empezar con ese proceso, es necesario pulsar el botón de Registrar. Cuando se realice esa acción mediante la voz, se podrá decir el nombre de la persona a registrar y su fecha de cumpleaños. En caso de que la fecha de cumpleaños no cumpla

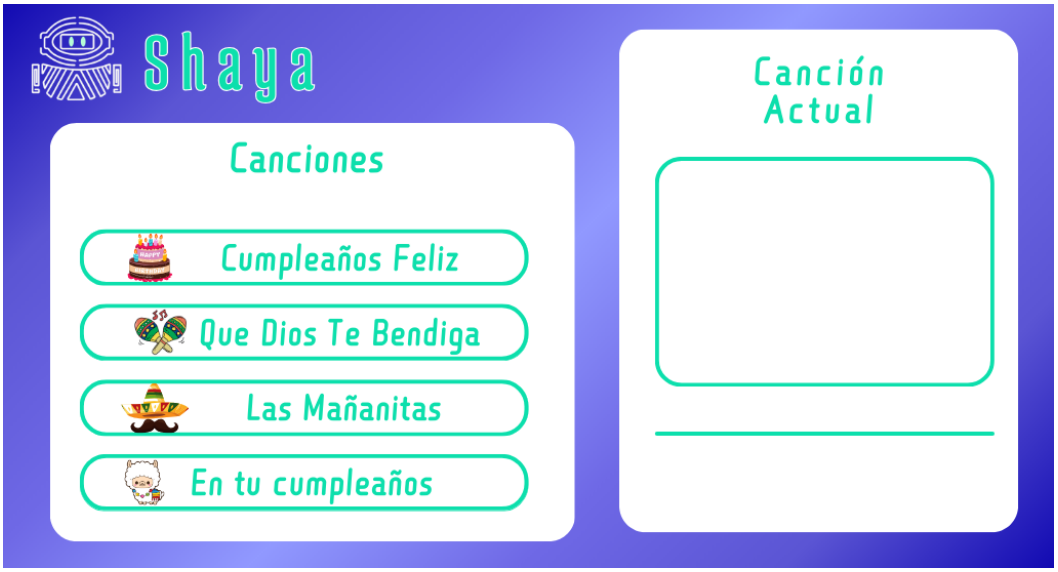
con la condición de ser una fecha posible, pedirá al usuario que repita su fecha de cumpleaños hasta que este diga una fecha que sea posible. Si la información es correcta, se realizará la captura del rostro de la persona y se guardaran en un archivo txt para ser utilizada más adelante.

Figura 4.1.3: Ventana de Registro

Ventana diseñada para el registro de personas que han resultado ser desconocidas por el reconocimiento facial previamente realizado. Fuente: Alisson Tamara Guaño Clavijo

Una vez registrada la persona, se volverá a verificar la fecha de cumpleaños con la fecha actual, y en caso de coincidir, como en el ejemplo anterior, se abrirá la página de canciones y si no, la de transición anteriormente mencionadas. La pantalla de Canciones muestra las diferentes opciones de canciones a elegir, actualmente se incorporaron 4 opciones diferentes y bastante conocidas las cuales pueden elegir libremente la que más le guste

Figura 4.1.4: Pantalla de Canciones



Pantalla diseñada para la elección de canciones. Fuente: Alisson Tamara Guaño Clavijo

Otra de las pantallas notorias de mencionar es la de la elección de los acompañantes. En esta el usuario podrá escoger quien le acompaña, tiene 3 opciones: Amigos, Pareja y Familia, y dependiendo de la elección las recomendaciones que dará serán completamente distintas.

Figura 4.1.5: Pantalla previa a las recomendaciones

Pantalla para que los usuarios escuchen las recomendaciones dependiendo de con quien está. Fuente: Alisson Tamara Guaño Clavijo

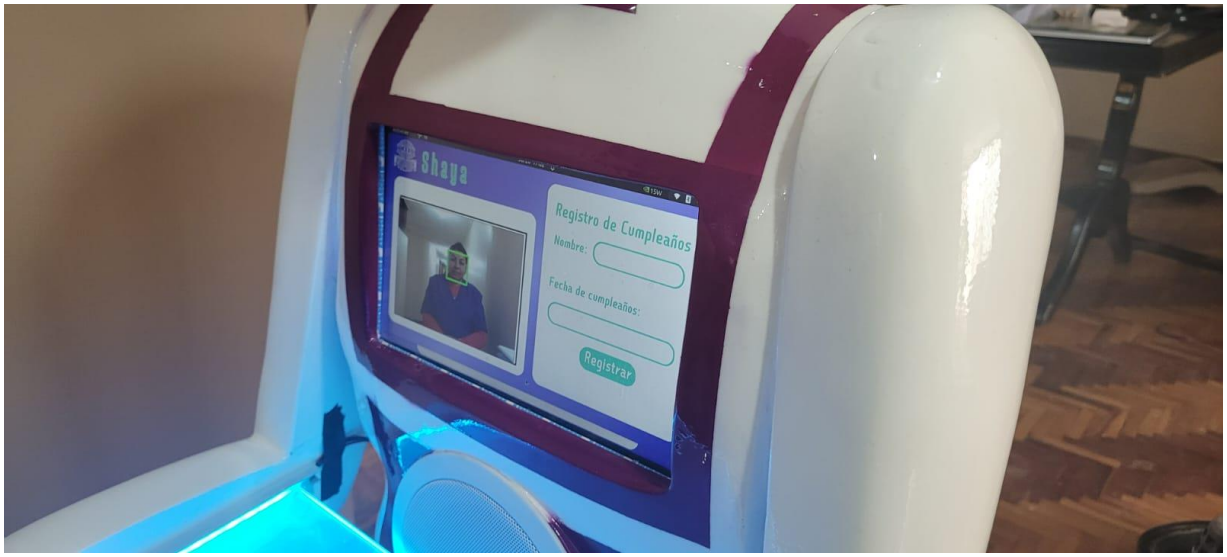
El usuario podrá escoger entre una de las tres opciones y dependiendo de la elección que haga.

Finalmente, se obtiene la pantalla que muestra las sugerencias y se puede visualizar los platos de los que se habla, para terminar, se despide y regresa a la pantalla inicial que se describió anteriormente

4.2.Pruebas de registro de Usuarios

En este apartado se describe el proceso de registro de usuarios nuevos. Previamente se realiza el reconocimiento facial, y al no detectar un rostro, este abrirá una nueva pantalla que se encargará de registrar a los usuarios mediante la voz, es decir, el bot preguntará el nombre gracias a la librería de TTS que permite que el "hable" e inmediatamente después la persona que debe ser registrada. Los datos que se registran son, el nombre y la fecha de su cumpleaños. Eso se verifica previamente antes de ser guardado.

Figura 4.2.1: Foto realizada cuando se hacia el proceso de registro con una persona desconocida.

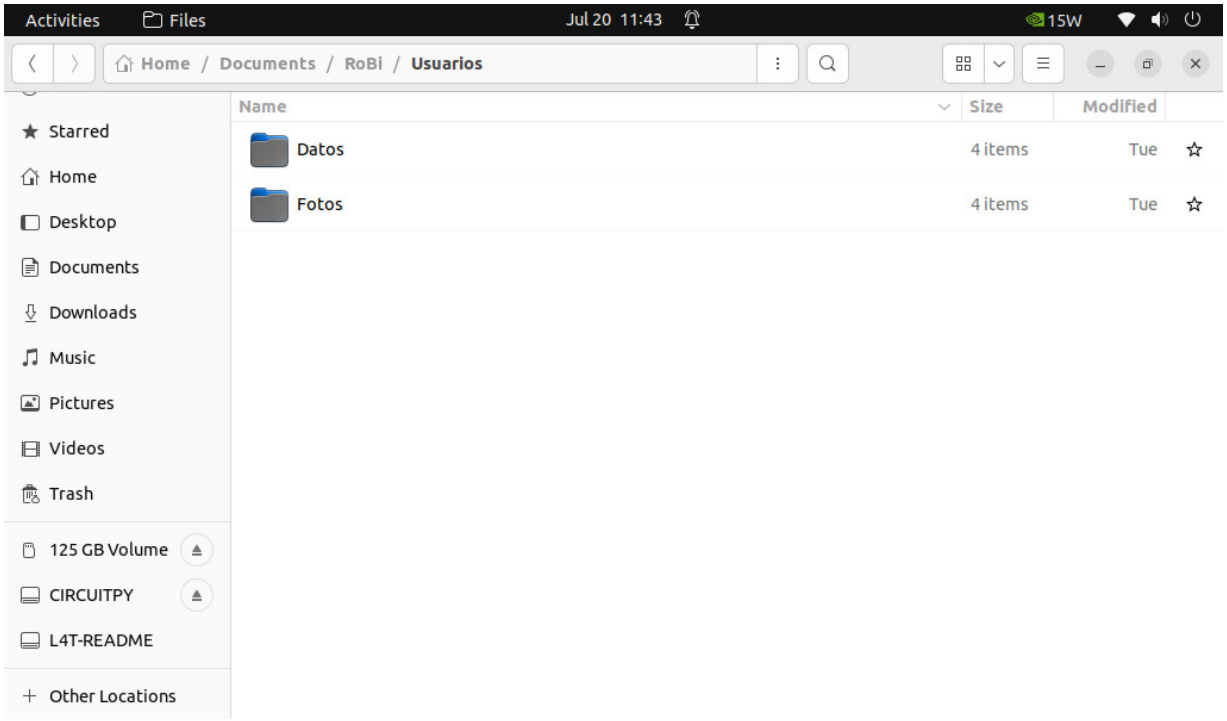


La persona no estaba previamente guardada, por lo tanto, el reconocimiento facial no la reconoció, asi que abre la pantalla que permite que se almacene. Fuente: Alisson Tamara Guaño Clavijo

La información obtenida es almacenada en un archivo txt, el nombre del archivo es un número de 5 dígitos completamente aleatorio. Tanto la imagen del rostro de la persona registrada como los datos tienen el mismo número aleatorio. Se puede observar en la

imagen de la Figura 4.2.2, con el fin de mantener organizado el programa, se decidió dividir en dos carpetas distintas, una que contenga el nombre de los datos de las personas y otra que contenga solo las imágenes.

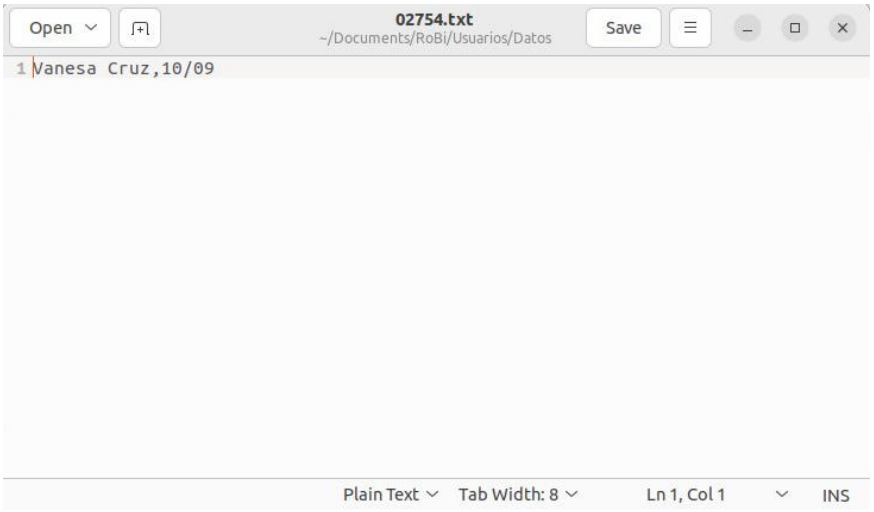
Figura 4.2.2: Captura realizada de la distribución de las carpetas en la Jetson Orin Nano



Para conseguir una mejor organización se dividieron las carpetas, una va a almacenar solo los datos de todos los clientes y la otro solo las fotos de todos los rostros. Fuente: Alisson Tamara Guaño Clavijo

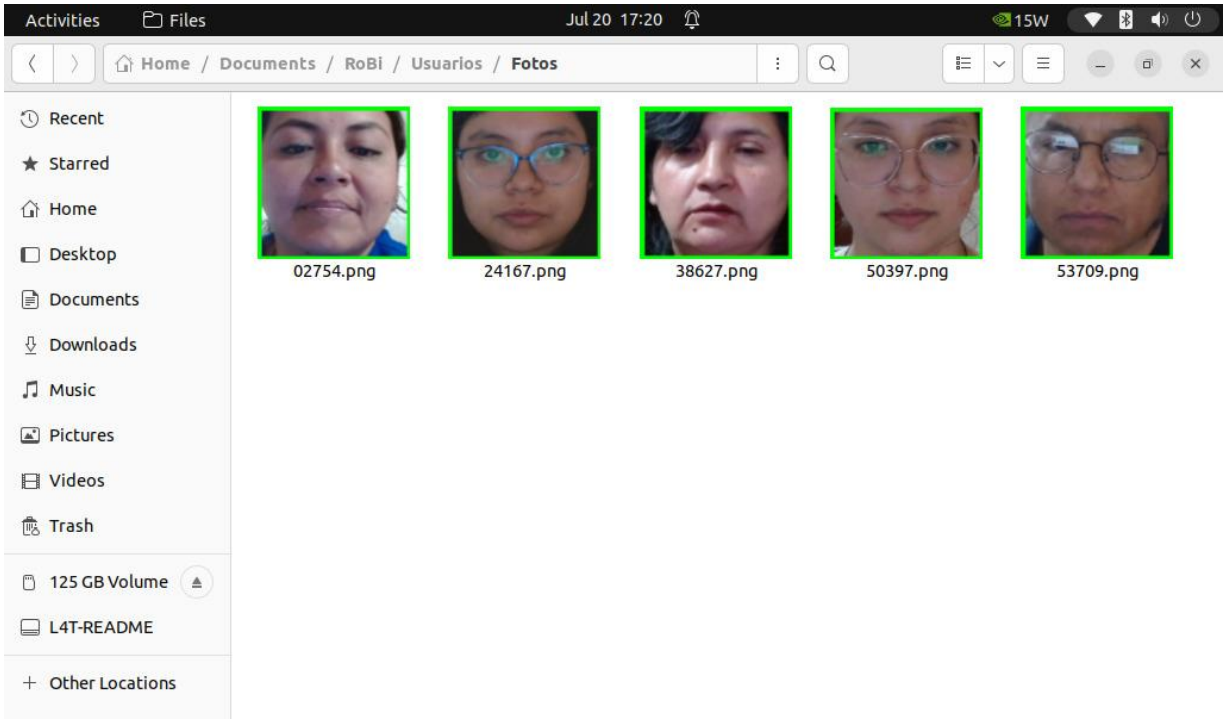
En la Figura 4.2.3 se puede observar los datos que se guardaron en el archivo generado, además en la parte superior de la misma se puede observar un código, el cual va a coincidir con el numero identificativo que tiene la imagen del rostro de la Figura 4.2.4

Figura 4.2.3: Archivo txt que contiene los datos de la persona registrada



La imagen muestra los datos proporcionados por la persona registrada, tanto la fecha de cumpleaños como el nombre. Fuente: Alisson Tamara Guaño Clavijo

Figura 4.2.4: Captura de pantalla de la carpeta que contiene las fotos de las personas registradas



Imágenes de las personas registradas. Fuente: Alisson Tamara Guaño Clavijo

Como se puede observar el nombre del archivo es el mismo que el de la foto, esto ayuda en otra etapa del programa igualmente importante que es el reconocimiento facial

Tabla 4.2.1: Intento de registro

Registro	Personas					
	1	2	3	4	5	
Desconocido	✓	✓	✓	✓	✖	✓

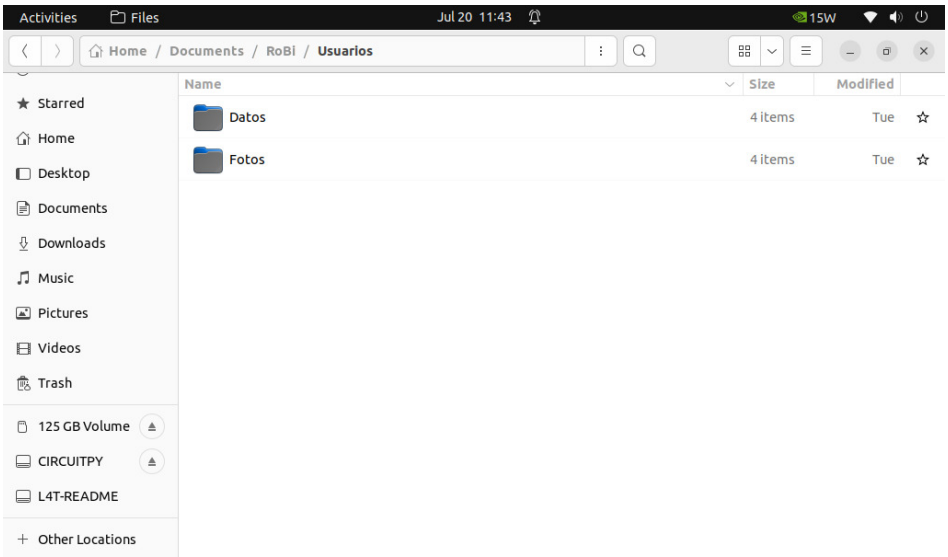
Pruebas de registro de clientes realizadas para su verificación

Se intentó realizar el registro con 5 personas. En esta prueba se pudo verificar que el reconocimiento facial detecta cuando una persona es conocida o desconocida, y permite el paso a la pantalla de registro de personas. Como resultado, se obtuvo que, en una sola ocasión, el reconocimiento facial confundió a la persona enfrente y no permitió el registro. El resto de las ocasiones funciono correctamente.

4.3.Pruebas de reconocimiento de Usuario

Para comprender la distribución de las carpetas es indispensable entender como es el funcionamiento del reconocimiento. Para conseguir resultados es esencial codificar las imágenes que se tienen guardadas una vez, según el apartado 4.2, que se completa el registro de las personas. Por lo tanto, se necesita una carpeta que disponga solo las imágenes para poder codificarlas y evitar que puedan surgir errores.

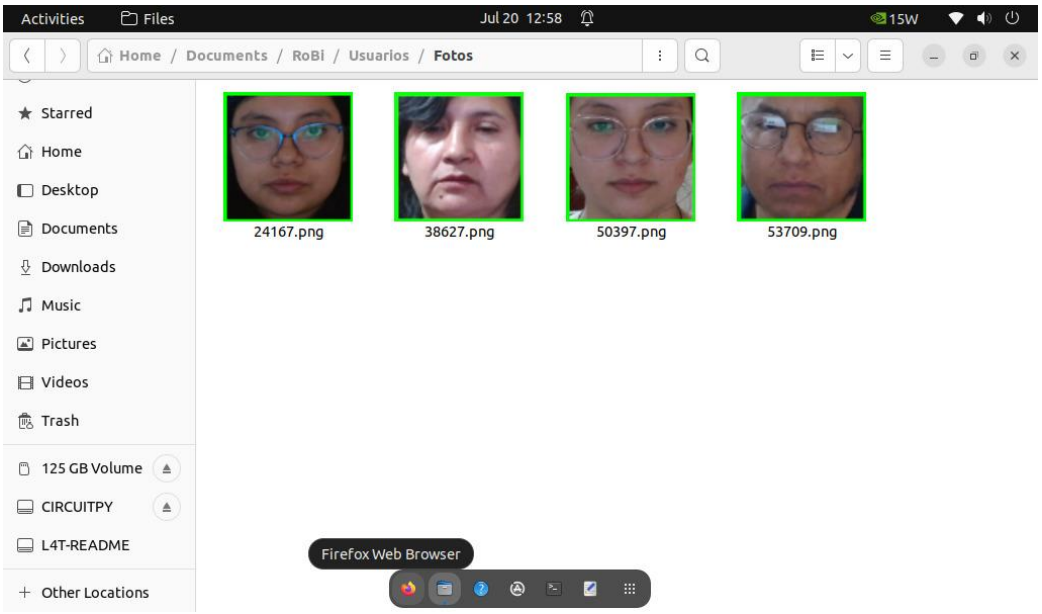
Figura 4.3.1: Captura de las carpetas que se necesitan para el reconocimiento facial



En la siguiente imagen se puede observar las dos carpetas que contienen la información necesaria para el reconocimiento facial. Fuente: Alisson Tamara Guaño Clavijo

Cada una de las imágenes que se pueden observar en la carpeta de Fotos, se importan al código que se realizó en Python, se codifican y se guardan en una lista para posteriormente poder utilizar la imagen de la cámara web, y que, al codificarse los rostros en vivo, se compare con las fotos, dando como resultado una de las imágenes del usuario ingresado. Como resultado se obtendrá el código del rostro que más se pareció y ese código quedará almacenado en una variable.

Figura 4.3.2: Fotos de las personas registradas y los códigos respectivos a cada uno.



Se puede observar las fotos de las personas que se han registrado y que han sido almacenadas en la carpeta de Fotos.

Tabla 4.3.1: Tabla de reconocimiento facial

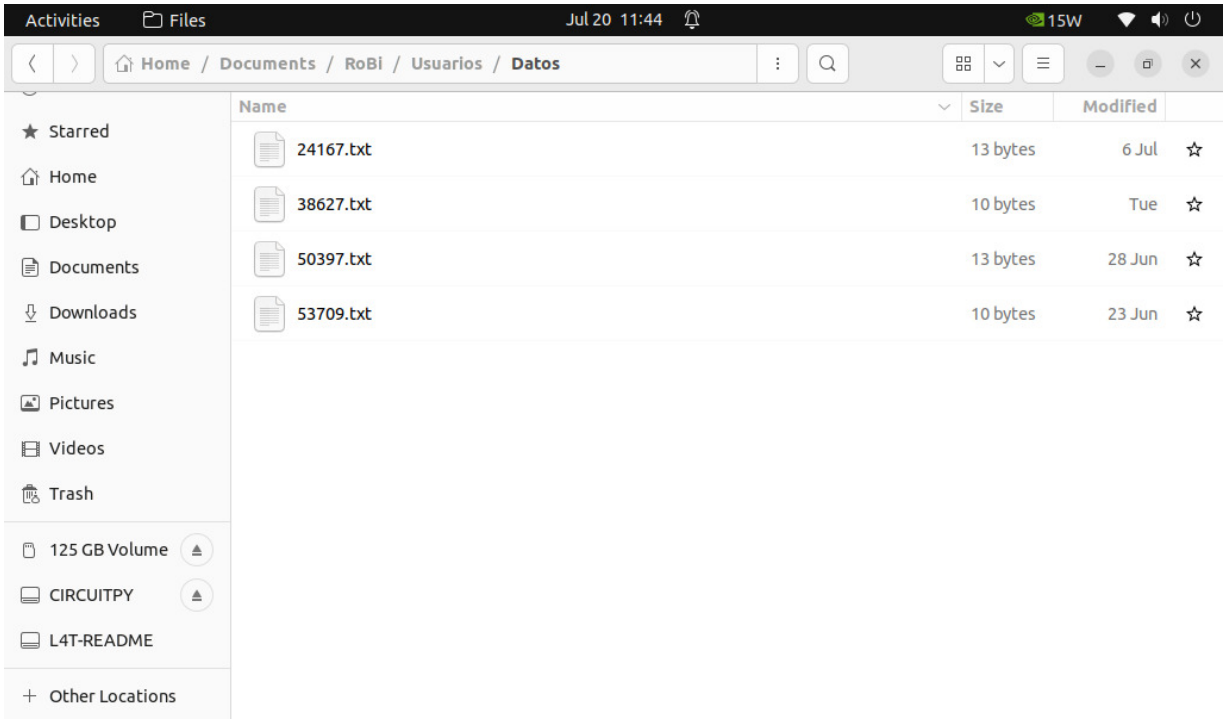
Código Cliente	Prueba										Porcentaje
	1	2	3	4	5	6	7	8	9	10	
24167	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100%
38627	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	90%
50397	✗	✓	✓	✓	✗	✓	✓	✓	✗	✓	70%
53709	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100%
										Total del porcentaje	72%

La tabla muestra las veces que reconoció a las personas

Como se muestra en la tabla 4.3.1 se realizaron 40 pruebas totales, 10 por cada usuario. En este caso se pudo observar que en la persona con el código 50397 tiene más errores al ser reconocido, esto se debe a que la persona con el código 24167 y la persona con el código 50397 son hermanas, y los rasgos faciales de las dos son similares, por lo tanto, el reconocimiento facial tiene fallos al reconocerla correctamente. Sin embargo, si una de ellas se retira los lentes, el reconocimiento facial funciona muy bien.

Una vez que se verifico los rostros y con el código guardado en la variable, solo se procede a extraer los datos del archivo .txt que corresponde al código que quedó guardado.

Figura 4.3.3: Carpeta de datos



Carpeta donde se guardan los datos de los clientes, cada archivo contiene nombre y fecha de cumpleaños de cada una de las personas.

En cuanto a la verificación de las skills, que en este caso se refiere al funcionamiento las canciones, se verifico de 2 formas. La primera ejecutando solo esa parte del código el cual dio como resultado la Tabla

Tabla 4.3.2:Tabla de funcionamiento de las canciones

Canción	Intentos									
	1	2	3	4	5	6	7	8	9	10
Feliz cumpleaños	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Lejos de ti	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Las mañanitas	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Que dios te bendiga	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Para la configuración de la música, ejecutando solo el programa que contenía la reproducción de las canciones, se probaron cada canción 10 veces y las 40 veces funcionó correctamente y se reprodujo la canción sin complicaciones.
La otra verificación que se realizó fue mediante el reconocimiento de personas. Para esto se tuvo que modificar las fechas reales de cumpleaños para que coincida con el día actual que se realizó la verificación.

En este caso se hizo la comprobación 1 vez por persona, ya que para llegar a este punto se tiene que modificar algunos datos y puede ser cansado y tedioso para las personas registradas, tener que hacer muchas pruebas. Los clientes registrados escogían una canción, por este motivo no se pudo verificar completamente con todas las canciones ya que las personas registradas son solo 5.

Tabla 4.3.3:Tabla de selección de canciones

Canción	Persona				
	02754	24167	38627	50397	53709
Feliz cumpleaños	✓	✓	✓		
Que dios te bendiga					
Las Mañanitas					✓
Lejos de ti				✓	

En este caso la tabla nuestra la elección que tuvo la gente, cuando se les mostró las opciones de canciones que tenían por escoger.

Con la Tabla 4.3.3: se puede observar las preferencias de las personas por una canción o

por otra. Se puede observar que el 60% de las personas que tuvo la oportunidad de escoger, prefirió escuchar el feliz cumpleaños. En segundo lugar, y siendo el 20% de preferencia cada opción, se tiene “Las Mañanitas” y “Lejos de Ti”, finalmente se observa que de lo usuarios que tuvieron acceso al bot, nadie tuvo interés en escuchar “Que dios te bendiga”. Con esto se comprueba que aceptación tiene las opciones escogidas, es posible buscar otras alternativas si una de las skills no es del gusto del consumir. Sin embargo, para determinar esto, se debe realizar la encuesta a más persona,

Finalmente, para saber la aceptación de las personas ante el bot, se le realizó una pequeña encuesta a las personas que pudieron interactuar, en este caso fueron a 5, y como se observa en el Anexo 11, la mayoría de las personas que probaron el bot les gusto, mientras que el 40% no se sintió muy convencido ante eso.
A pesar de que algunas de las personas disfrutaron de la interacción con el bot, aumentarían las funcionalidades que ofrece el bot como muestras los resultados del anexo 12. Por lo tanto, es un gran detalle a tener en cuenta para continuar con el desarrollo del bot a futuro que complazca y mejor más la experiencia de los clientes.

En el grafico se puede observar que aunque las personas encuestadas, no estuvieron totalmente complacidas con la experiencia con el robot si tiene mucho interés en este tipo de tecnología y siente interés en continuar teniendo experiencias relacionas con el bot, y eso se ve reflejado en los resultados del anexo 13

CONCLUSIONES

Con el presente proyecto se pudo observar algunas de las herramientas de inteligencia artificial que sería necesario para la construcción de un bot y sobretodo se aprendió como utilizarlas para esta aplicación y algunas restricciones que tienen. La aplicación realizada en este proyecto depende completamente de la conexión a internet ya que las librerias de voz y texto que se usa necesitan mantener una comunicación con las respectivas plataformas que proporciona el servicio.

Con el antecedente de la aceptación por parte de las personas sobre la inteligencia artificial mediante las encuestas realizadas a los clientes con las que se hicieron las pruebas de campo, se proyecta un interés por parte del consumidor, y sobretodo ocupar este tipo de herramientas tanto en el presente como en el futuro para mejorar la experiencia que tengan al venir a consumir en el restaurante La Taberna.

Previo a la construcción del proyecto, se realizó una extensa investigación de la aplicación de este tipo de recursos y sobretodo del impacto ante los avances que actualmente tiene. Además de las herramientas que se podían utilizar para poder llevar a cabo este proyecto y de los materiales a utilizar, buscando entre ellos el más económico pero sin perder calidad, por lo tanto, el conocimiento obtenido de este proyecto no es solo en el ámbito de desarrollo de IA si no construcción, de materiales, entre otros. Como resultado de esta investigación y con el objetivo de realizar algo diferente a lo que ya estaba construido, se utilizó la inteligencia artificial, para que el trato fuera personalizado, es decir que tuviera la capacidad de interactuar directamente con una persona y que no fuera algo general.

Al desarrollar un proyecto que requiere tener en consideración muchos aspectos, se pudo observar el gran trabajo interdisciplinario que este requiere, ya que no solo se necesita el conocimiento en electrónica, por la parte del consumo y conexiones, si no también fue indispensable un amplio conocimiento en programación, ya que fue lo más necesario para aplicar las librerias de inteligencia artificial para las diferentes herramientas que se utilizó, incluso tener conocimientos básicos sobre sistemas operativos, ya que todo fue desarrollado en una Jetson Orin Nano, la cual tiene un sistema operativo basado en Linux que es muy diferente al que tradicionalmente se utilizan en las computadoras, esto

provoca que el conocimiento que se tiene de compatibilidad entre librerias y la instalación no sea de gran ayuda y se necesite consultar mucho antes incorporar algo nuevo.

Finalmente, para concluir este proyecto fue indispensable ponerlo en práctica y obtener datos, como los casos en los que el reconocimiento facial podría o no fallar, como se observó en las tablas de arriba se pudo observar que el programa tuvo un acierto del 72%. Las skills aplicadas están completamente operativas ya que de los 40 intentos que se realizó, el 100% de las veces funcionaron correctamente. Aunque al momento de probarlo con el programa entero solo se pudo comprobar 1 vez por persona y en todos los casos funcionó, sería necesario aumentar las pruebas para comprobar que el funcionamiento de todas las opciones es correcto.

RECOMENDACIONES

Conseguir desarrollar este tipo de proyecto requiere de una gran inversión de tiempo, ya que es necesario tener en cuenta muchas variables, como la compatibilidad del sistema, el lenguaje de programación que se usa, el sistema operativo que se instala, los beneficios y restricciones que tiene, por lo tanto, antes de realizar este tipo de proyecto se recomienda leer a fondo las especificaciones técnicas del sistema embebido que se va a utilizar, y sobretodo investigar si existe algún tipo de servicio técnico accesible con el que consultar peculiaridades, errores o daños que puedan salir en el proceso.

Como la inteligencia artificial es en gran parte programación, es indispensable conocer sobre lenguajes de programación o por lo menos uno que sea aplicado para desarrollar este tipo de proyectos, por este motivo se recomienda realizar cursos, no solo de inteligencia artificial si no de Python, interfaces gráficas, programación orientada objetos o temas relacionadas. Esto va a ser muy útil a larga por que hará que la programación se haga más sencilla y reducirán los errores que se produzcan.

Una de las recomendaciones más valiosas y que ayudó en la realización de este proyecto es utilizar herramientas de desarrollo, como lo es Github. En esta plataforma se puede encontrar códigos de todo tipo, y mucha gente ha desarrollado previamente algunas de las ideas que se tiene en mente, y es de gran ayuda observar diferentes lógicas de programación y entender el proceso que se hizo y extraer ideas para concluir con el proyecto que se tiene en mente.

Al ser un proyecto que requiere mucho la interacción con la gente, es recomendable finalizar un código base e ir modificando según el comportamiento que se observa de la gente, con eso se corrige lo necesario y se implementa solo lo indispensable.

REFERENCIAS BIBLIOGRAFICAS

A, P. (2023). Artificial Intelligence and its Applications. *International Scientific Journal of Engineering and Management*, *02*(04). https://doi.org/10.55041/ISJEM00301

Adam Geitgey. (2020). *Face Recognition package*. https://face-recognition.readthedocs.io/en/latest/face_recognition.html

Adam Geitgey. (2022). *Dlib C++ Library*. http://dlib.net/

Alberto Cuevas Àlvarez. (2018). *Aplicaciones gráficas con Python 3*. RA-MA.

Awais, M., Saeed, M. Y., Malik, M. S. A., Younas, M., & Rao Iqbal Asif, S. (2020). Intention Based Comparative Analysis of Human-Robot Interaction. *IEEE Access*, *8*, 205821–205835. https://doi.org/10.1109/ACCESS.2020.3035201

Blauth, T. F., Gstrein, O. J., & Zwitter, A. (2022). Artificial Intelligence Crime: An Overview of Malicious Use and Abuse of AI. *IEEE Access*, *10*, 77110–77122. https://doi.org/10.1109/ACCESS.2022.3191790

Blöcher, K., & Alt, R. (2021). AI and robotics in the European restaurant sector: Assessing potentials for process innovation in a high-contact service industry. *Electronic Markets*, *31*(3), 529–551. https://doi.org/10.1007/s12525-020-00443-2

David L. Rogers. (221 C.E.). *Digital Transformation Playbook* (Columbia University Press, Ed.; 1st ed.). Ediciones Urano.

De, B. (2017). Introduction to APIs. In *API Management* (pp. 1–14). Apress. https://doi.org/10.1007/978-1-4842-1305-6_1

Gao, H., Qin, X., Barroso, R. J. D., Hussain, W., Xu, Y., & Yin, Y. (2022). Collaborative Learning-Based Industrial IoT API Recommendation for Software-Defined Devices: The Implicit Knowledge Discovery Perspective. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *6*(1), 66–76. https://doi.org/10.1109/TETCI.2020.3023155

Kaur, N., Mahajan, N., Singh, V., & Gupta, A. (2023). Artificial Intelligence Revolutionizing The Restaurant Industry - Analyzing Customer Experience Through Data Mining and Thematic Content Analysis. *2023 3rd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, 1–5. https://doi.org/10.1109/ICIPTM57143.2023.10117897

Lai, X., & Patrick Rau, P.-L. (2021). Has facial recognition technology been misused? A public perception model of facial recognition scenarios. *Computers in Human*

Behavior, *124*, 106894. https://doi.org/10.1016/j.chb.2021.106894

Latika Kharb, & Deepak Chahal. (2023). Privacy Threats in Facial Recognition-Based Identity Verification. *International Journal of Advanced Research in Science, Communication and Technology*, 569–572. https://doi.org/10.48175/IJARSCT-11686

Lebeuf, C., Storey, M.-A., & Zagalsky, A. (2018). Software Bots. *IEEE Software*, *35*(1), 18–23. https://doi.org/10.1109/MS.2017.4541027

Li, L., Mu, X., Li, S., & Peng, H. (2020). A Review of Face Recognition Technology. *IEEE Access*, *8*, 139110–139120. https://doi.org/10.1109/ACCESS.2020.3011028

Los robots meseros llegan para quedarse: ¿Cómo cambiará la industria de restaurantes? (2023, April 8). *El Vocero de Puerto Rico*. https://www.proquest.com/newspapers/los-robots-meseros-llegan-para-quedarse-cómo/docview/2798611179/se-2?accountid=32861

Martinez Jaume. (2019). *Industria 4.0* (1st ed.). Editorial UOC.

Micrófono por USB. (n.d.). Retrieved July 24, 2024, from https://www.google.com/search?sca_esv=00cbfd697922c9db&hl=en&sxsrf=ADLYWILK69tEilFny4Rrv-kTL4uLIfK00w:1721916472282&q=microfono+lavalier&tbm=isch&source=lnms&fbs=AEQNm0C5IIOpSDYVh51ut_zQFgH5DNv_pTVgUFFjqHIROnWawIGVx0E6ObWKkhgJ1A_fnEQWRGH1n7mpHWnN6UNr_Gm3zX6YU52_oeBIXhbaguYxZuLgU8vpKE6m6-wMC179lPvkI-jk8et3sM1ra3TNp1coYh9wygIKmVxCI0NsnOuNyQPPFyh6m9Rx8pIVFjqcOIwqHJzJ&sa=X&ved=2ahUKEwip96bdrsKHAxVeRTABHRD8DZ8Q0pQJegQIFRAB&biw=1536&bih=703&dpr=1.25

Módulo bluetooth. (n.d.). Retrieved July 24, 2024, from https://www.google.com/search?q=Amplificador+Mini+Usb+Sd%2Ffm%2Fblt+200w+2ch+Yw-(191l)&&tbm=isch&ved=2ahUKEwi9jc_ZrcKHAxWjobAFHR5UAb0Q2-cCegQIABAA&oq=Amplificador+Mini+Usb+Sd%2Ffm%2Fblt+200w+2ch+Yw-(191l)&gs_lp=EgNpbWciMkFtcGxpZmljYWRvciBNaW5pIFVzYiBTZC9mbS9ibHQgMjAwdyAyY2ggWXctKDE5MWwpSLMJUJYHWJYHcAF4AJABAJgBngGgAZ4BqgEDMC4xuAEDyAEA-AEBigILZ3dzLXdpei1pbWeIBgE&sclient=img&ei=JFuiZr3rBqPDwt0PnqiF6As&bih=703&biw=1536#imgrc=TA6AV9qqnT2KzM

Neilenko, S., & Rusavska, V. (2021). Implementation of Artificial Intelligence in Restaurants. *Restaurant and Hotel Consulting. Innovations*, *4*(1), 73–86. https://doi.org/10.31866/2616-7468.4.1.2021.234831

NVIDIA. (n.d.-a). *NVIDIA Jetson Orin*. Retrieved July 16, 2024, from https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/

NVIDIA. (n.d.-b). *Technologies*. NVIDIA CUDA-X. Retrieved July 16, 2024, from https://www.nvidia.com/en-us/technologies/cuda-x/

NVIDIA. (2023). *Jetson Orin Nano Developer Kit Carrier Board*.

Open AI lanza la API de ChatGPT y de Whisper para poder integrar estas herramientas en las "apps" de los desarrolladores. (2023, March 2). *Estrategia*. https://www.proquest.com/newspapers/open-ai-lanza-la-api-de-chatgpt-y-whisper-para/docview/2782135392/se-2?accountid=32861

OpenAi. (n.d.). *API Reference*. Retrieved July 16, 2024, from https://platform.openai.com/docs/api-reference/assistants-streaming-v1/events

OpenAI. (n.d.). *Pricing*. Retrieved July 19, 2024, from https://openai.com/api/pricing/

OpenAI. (2023). *Whisper*. https://github.com/openai/whisper?tab=MIT-1-ov-file

Parlante de 6w. (n.d.). Retrieved July 24, 2024, from https://www.google.com/search?q=parlante+de+6w&tbm=isch&ved=2ahUKEwiSvqmNrsKHAxXp1ckDHWDAIh4Q2-cCegQIABAA&oq=parlante+de+6w&gs_lp=EgNpbWciDnBhcmxhbnRlIGRlIDZ3MgQQABgeMgQQABgeMgQQABgeMgQQABgeMgQQABgeMgYQABgFGB4yBhAAGAgYHjIGEAAYCBgeMgYQABgIGB5I3h5QoAdY8B1wAHgAkAEAmAHfAaABnRGqAQYwLjEzLjG4AQPIAQD4AQGKAgtnd3Mtd2l6LWltZ8ICBBAjGCfCAgUQABiABMICChAAGIAEGEMYigWIBgE&sclient=img&ei=kFuiZpK2Iemrp84P4ICL8QE&bih=703&biw=1536#imgrc=W7S2Wk6GlM02pM

Peter D. Nyheim. (2019). *Estrategias Tecnológicas para la Industria de la Hostelería* (1st ed.). Editorial de la Universidad Catolica de Salta, EUCaSa.

Pierre Nicolas Durette. (2024). *gtts*. https://gtts.readthedocs.io/en/latest/

pndurette. (2024). *GTTS*. https://pypi.org/project/gTTS/

Powest UPS. (n.d.). Retrieved July 24, 2024, from https://www.google.com/search?sca_esv=00cbfd697922c9db&sxsrf=ADLYWILn28Wpc434SposjJg7zgvyM2p8yQ:1721914397540&q=powest+ups&tbm=isch&so

urce=lnms&fbs=AEQNm0Ct5BGqt6h9_ZXsXikdzlxSYpuxWKtPjQ1k43zoSC7iY3idU2FSH7o-NaQYajlunfmo70vQSVo4x85eYaZUmN0mAtiNZl4YeEmyJuHpfhfoXT-RM6sqldqMHaCfH6-rQSyF1dUNXvz4G3YtyQnpRCqrMwyXiXlASXAKWLB710_HSZvpzOba8bPZW9rj-rHZDMLXZwDf&sa=X&ved=2ahUKEwjt2f7_psKHAxU1RTABHd-JAAMQ0pQJegQIDRAB&biw=1536&bih=703&dpr=1.25#imgrc=ZygWK4-irMFTYM&imgdii=1l4YQ1-3IJawPM

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., & Sutskever, I. (2022). *Robust Speech Recognition via Large-Scale Weak Supervision*.

Raschka, S., Patterson, J., & Nolet, C. (2020). Machine Learning in Python: Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence. *Information*, *11*(4), 193. https://doi.org/10.3390/info11040193

Rock, R. (2024, May 20). Turismo tecnológico: Mucho más que robots que sirven los tragos. *El Sol de Mexico*. https://www.proquest.com/newspapers/turismo-tecnológico-mucho-más-que-robots-sirven/docview/3057592356/se-2?accountid=32861

Rusan, H.-A., & Mocanu, B. (2022). Human-Computer Interaction Through Voice Commands Recognition. *2022 International Symposium on Electronics and Telecommunications (ISETC)*, 1–4. https://doi.org/10.1109/ISETC56213.2022.10010253

Seetha, H., Tiwari, V., Anugu, K. R., Makka, D. S., & Karnati, D. R. (2023). A GUI Based Application for PDF Processing Tools Using Python & CustomTkinter. *International Journal for Research in Applied Science and Engineering Technology*, *11*(1), 1613–1618. https://doi.org/10.22214/ijraset.2023.48848

Threading- therad -based parallelism. (2024). https://docs.python.org/3/library/_thread.html#module-_thread

Tira LED RGB . (n.d.). Retrieved July 24, 2024, from https://www.google.com/search?q=Tira+de+luces+LED+RGB+con+WiFi%2C+12V%2C+con+aplicaci%C3%B3n+Tuya+Smart+Life%2C+se%C3%B1al+de+ne%C3%B3n+Flexible%2C+cinta+de+luz+de+fondo+impermeable&tbm=isch&ved=2ahUKEwjBw8SKrcKHAxVTi7AFHaQMAbAQ2-cCegQIABAA&oq=Tira+de+luces+LED+RGB+con+WiFi%2C+12V%2C+con+a

plicaci%C3%B3n+Tuya+Smart+Life%2C+se%C3%B1al+de+ne%C3%B3n+Flexible%2C+cinta+de+luz+de+fondo+impermeable&gs_lp=EgNpbWcigQFUaXJhIGRlIGx1Y2VzIExFRCBSR0IgY29uIFdpRmksIDEyViwgY29uIGFwbGljYWNpw7NuIFR1eWEgU21hcnQgTGlmZSwgc2XDsWFsIGRlIG5lw7NuIEZsZXhpYmxlLCBjaW50YSBkZSBsdXogZGUgZm9uZG8gaW1wZXJtZWFibGVIrwNQAFgAcAB4AJABAJgBAKABAKoBALgBA8gBAPgBAYoCC2d3cy13aXotaW1niAYB&sclient=img&ei=flqiZsGMENOWwt0PpJmEgAs&bih=703&biw=1536#imgrc=LM1EOw-XiXXvFM

Vlačić, B., Corbo, L., Costa e Silva, S., & Dabić, M. (2021). The evolving role of artificial intelligence in marketing: A review and research agenda. *Journal of Business Research, 128,* 187–203. https://doi.org/https://doi.org/10.1016/j.jbusres.2021.01.055

ANEXOS

Anexo 1. Código de transición

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERP
O_PRINCIPAL
from Formulario.recomendaciones import Recomendaciones
import Util.util_ventana as util_ventana
import cv2
from PIL import Image,ImageTk
import sys
import datetime
import os
import face_recognition
from gtts import gTTS
import threading
import time


class Transicion(tk.Toplevel):

    def __init__(self):
        super().__init__()
        self.fondo=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/Recomendacion
es.png")
        self.si_on=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/sion.png")
        self.si_off=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/sioff.png")
        self.no_on=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/noon.png")
        self.no_off=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/nooff.png")


        self.pantalla()
        self.fondo_pantalla()
        self.Botones()
        self.escoge()
        self.hilo_audio()


    def pantalla(self):
        self.title("RoBi")
        self.resizable(0,0)
        util_ventana.centrar_ventana(self,1024,550)
        self.overrideredirect(True)

    def fondo_pantalla(self):
        self.miframe=tk.Frame(self,bg=COLOR_BARRA_SUPERIOR, height=50)
        self.miframe.pack(side=tk.TOP)

        self.background=tk.Label(self.miframe, image=self.fondo)
        self.background.pack()

    def Botones(self):
```

```
boton_si=tk.Button(self.miframe,image=self.si_off,bd=0,highlightthickness=0,
bg="#ffffff",command=self.si)
        boton_si.place(x=295,y=415)


boton_no=tk.Button(self.miframe,image=self.no_off,bd=0,highlightthickness=0,
bg="#ffffff",command=self.no)
        boton_no.place(x=580,y=415)

        self.bind_hover_events1(boton_si)
        self.bind_hover_events2(boton_no)

    def bind_hover_events1(self,boton_si):
        boton_si.bind("<Enter>", lambda event:self.on_enter1(event,boton_si))
        boton_si.bind("<Leave>", lambda event:self.on_leave1(event,boton_si))

    def on_enter1(self,event,boton_si):

boton_si.config(image=self.si_on,bd=0,highlightthickness=0,bg="#ffffff")

    def on_leave1(self,event,boton_si):

boton_si.config(image=self.si_off,bd=0,highlightthickness=0,bg="#ffffff")

    def bind_hover_events2(self,boton_no):
        boton_no.bind("<Enter>", lambda event:self.on_enter2(event,boton_no))
        boton_no.bind("<Leave>", lambda event:self.on_leave2(event,boton_no))

    def on_enter2(self,event,boton_no):

boton_no.config(image=self.no_on,bd=0,highlightthickness=0,bg="#ffffff")

    def on_leave2(self,event,boton_no):

boton_no.config(image=self.no_off,bd=0,highlightthickness=0,bg="#ffffff")

    def escoge(self):
        escogesiono=f"Pero no pasa nada, si quieres puedo hacerte unas
sugerencias de nuestros platos dependiendo de con quien nos visites. ¿Te
gustaría que lo hiciera?"
        tts = gTTS(text=escogesiono, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-
Speech/escogesiono.mp3")

    def si(self):
        escogesiono=f"Genial! será un placer darte unas sugerencias!"
        tts = gTTS(text=escogesiono, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-
Speech/escogesiono.mp3")
        self.hilo_audio()
        time.sleep(6)
        Recomendaciones()
        self.destroy()

    def no(self):
        escogesiono=f"No hay ningun problema, igualmente ha sido un gusto
verte. Muchas gracias por visitarnos. Recuerda seguirnos en nuestras redes
sociales para conocer increibles promociones. ¡Hasta la proxima!"
        tts = gTTS(text=escogesiono, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-
Speech/escogesiono.mp3")
        self.hilo_audio()
```

```
        time.sleep(17)
        self.destroy()

    def hilo_audio(self):
        hilo_reproduccion = threading.Thread(target=self.reproducir_audio)
        hilo_reproduccion.start()

    def reproducir_audio(self):
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/escogesiono.mp3")
```

Anexo 2. Script de registro de usuarios

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERP
O_PRINCIPAL
import Util.util_ventana as util_ventana
import Util.reconocimiento_facial as reconocimiento
from Formulario.transicion import Transicion
from Formulario.grabar import grabar
from Formulario.canciones import CancionesAEscoger
import cv2
from PIL import Image,ImageTk
import sys
import random
import datetime
import os
import face_recognition
from gtts import gTTS
import threading
import time
import openai

class RegistroUsuarios(tk.Toplevel):

    def __init__(self):
        super().__init__()

        self.fondo=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/Fondo.png")

self.registrar_on=ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imag
enes/registraron.png")

self.registrar_off=ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Ima
genes/registraroff.png")

self.si=ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/si.pn
g")

self.blanco=ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/b
lanco.png")



        self.pantalla()
        self.fondo_pantalla()
```

```
        self.camara()
        self.complementos()
        self.botones()
        self.registro()
        self.hilo_audio_bienvenida()



    def pantalla(self):
        self.title("RoBi")
        self.resizable(0,0)
        self.overrideredirect(True)
        util_ventana.centrar_ventana(self,1024,550)

    def fondo_pantalla(self):
        self.miframe=tk.Frame(self,bg=COLOR_BARRA_SUPERIOR, height=50)
        self.miframe.pack(side=tk.TOP)

        self.background=tk.Label(self.miframe, image=self.fondo)
        self.background.pack()

    def camara(self):
        self.video=tk.Label(self.miframe, bg=COLOR_BARRA_SUPERIOR)
        self.video.place(x=58,y=164)
        self.camera=cv2.VideoCapture(0)
        self.image=tk.PhotoImage(width=449,height=326)
        self.video.config(image=self.image)
        self.actualizar_frame()


#control de la videocamara con cv2
    def actualizar_frame(self):
        (capture,frame)=self.camera.read()
        frame=cv2.flip(frame,1)
        frame=cv2.cvtColor(frame,cv2.COLOR_BGR2RGB)
        frame=cv2.resize(frame,(449,326))
        face_locations=face_recognition.face_locations(frame)

        for face_location in face_locations:
            x1,y1=face_location[3],face_location[0]
            x2,y2=face_location[1],face_location[2]

self.imagen_rostro=cv2.rectangle(frame,(x1,y1),(x2,y2),(0,255,0),3)
            self.recorte=frame[y1:y2,x1:x2]
            self.recorte_si=cv2.cvtColor(self.recorte,cv2.COLOR_BGR2RGB)
        im=Image.fromarray(frame)
        self.imagen=ImageTk.PhotoImage(image=im)
        self.video.config(im=self.imagen)
        self.video.after(10,self.actualizar_frame)

    def save_frame(self):
        valor=self.random()
        ret,frame=self.camera.read()
        frame=cv2.flip(frame,1)

        print(valor)
        if ret:
            cv2.imwrite('Usuarios/Fotos/'+f"{valor}.png",self.recorte_si)
        #para poder guardar los datos

            f=open('Usuarios/Datos/'+f"{valor}.txt","w")
            f.write(self.texto+',')
            f.write(self.fecha_completa)
```

```
            f.close()

    def random(self):
        numeros="1234567890"
        longitud=5
        extension=random.sample(numeros,longitud)
        valor="".join(extension)
        return valor

    def complementos(self):
        self.entrada_nombre=tk.Label(self.miframe, text=" ",
bd=0,bg="#ffffff",width=18,highlightthickness=0 )
        self.entrada_nombre.place(x=735,y=139)

        self.entrada_fecha=tk.Label(self.miframe, text=" ",
bd=0,bg="#ffffff",width=18,highlightthickness=0 )
        self.entrada_fecha.place(x=645,y=311)

    def fecha_valida(self):
        fecha_por_verificar=self.fecha_completa
        try:
            datetime.datetime.strptime(fecha_por_verificar,'%d/%m')
            return True
        except ValueError:
            return False

    def botones(self):

boton_registro=tk.Button(self.miframe,image=self.registrar_on,bd=0,highlight
thickness=0,bg="#ffffff",command=self.iniciar_registro)
        boton_registro.place(x=700,y=359)
        self.bind_hover_events(boton_registro)

    def bind_hover_events(self,boton_registro):
        boton_registro.bind("<Enter>", lambda
event:self.on_enter(event,boton_registro))
        boton_registro.bind("<Leave>", lambda
event:self.on_leave(event,boton_registro))

    def on_enter(self,event,boton_registro):

boton_registro.config(image=self.registrar_on,bd=0,highlightthickness=0,bg="
#ffffff")

    def on_leave(self,event,boton_registro):

boton_registro.config(image=self.registrar_off,bd=0,highlightthickness=0,bg=
"#ffffff")

    def iniciar_registro(self):

        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/nombre.mp3")
        self.preguntar_nombre()

    def preguntar_nombre(self):

        self.texto=grabar()
        print (self.texto)

        self.entrada_nombre=tk.Label(self.miframe,text=self.texto,
font=("Arial",18),bd=0,bg="#ffffff",width=18,highlightthickness=0 )
        self.entrada_nombre.place(x=735,y=139)
```

```
        time.sleep(1)

        self.preguntar_fecha()

    def registro(self):
        registro=f"Ahora vamos a registrar algunos datos, por favor pulsa
Registrar."
        tts = gTTS(text=registro, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-Speech/registro.mp3")

    def hilo_audio_bienvenida(self):
        hilo_reproduccion =
threading.Thread(target=self.reproducir_audio_bienvenida)
        hilo_reproduccion.start()

    def reproducir_audio_bienvenida(self):
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/registro.mp3")


    def preguntar_fecha(self):
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/fecha.mp3")
        self.fecha=grabar()
        print (self.fecha)
        print(self.texto)
        self.entrada_fecha=tk.Label(self.miframe,text=self.fecha,
font=("Arial",18),bd=0,bg="#ffffff",width=18,highlightthickness=0 )
        self.entrada_fecha.place(x=645,y=311)
        fecha_cumpleaños=self.fecha
        separado=fecha_cumpleaños.split()
        print(separado)
        Dia=separado[0]
        Mes=separado[2]
        mes_a_numero={
            "enero":"01",
            "febrero":"02",
            "marzo":"03",
            "abril":"04",
            "mayo":"05",
            "junio":"06",
            "julio":"07",
            "agosto":"08",
            "septiembre":"09",
            "octubre":"10",
            "noviembre":"11",
            "diciembre":"12"
        }

caracteres_permitidos="abcdefghijklmnopkrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
        cadena_original=Mes

cadena_sin_especiales=self.eliminar_caracteres_especiales(cadena_original,ca
racteres_permitidos)
        mes_minuscula=cadena_sin_especiales.lower()
        mes_numero=mes_a_numero[mes_minuscula]
        print(mes_numero)
        self.recibir_dia=Dia
        self.recibir_mes=mes_numero
        self.fecha_completa=self.recibir_dia+'/'+self.recibir_mes
        print(self.fecha_completa)
        fecha_validada=self.fecha_valida()
        if fecha_validada==True:
```

```
            print("fecha correcta")

fecha_completa1=datetime.datetime.strptime(self.fecha_completa,'%d/%m')

fecha_completa_modificada=datetime.datetime.strftime(fecha_completa1,'%d/%m'
)
            self.save_frame()
            hoy=datetime.datetime.now()
            hoy_modificado=datetime.datetime.strftime(hoy,'%d/%m')
            if fecha_completa_modificada==hoy_modificado:
               print("es tu cumpleaños")
               print("es tu cumpleaños")
               time.sleep(10)
               self.hablar_cumpleanos()
               self.hilo_audio()
               time.sleep(5)
               self.pantalla_canciones()

            else:
               print("no es tu cumpleaños")
               self.hablar_no_cumpleanos()
               time.sleep(5)
               self.hilo_audio()
               time.sleep(5)
               Transicion()
               self.camera.release()
               self.destroy()
         else:
            print("fecha incorrecta")
            self.preguntar_fecha()

    def hablar_no_cumpleanos(self):
         saludo=f"Hola {self.texto}, es un gusto conocer. Parece que el dia de
hoy no es tu cumpleaños"

         tts = gTTS(text=saludo, lang="es", tld="es")

         tts.save("/home/tammyai/Documents/RoBi/Text-To-Speech/saludo.mp3")

    def hablar_cumpleanos(self):
         saludo=f"Hola {self.texto}, es un gusto conocerte. ¡Genial! hoy es tu
cumpleaños, tenemos una sopresa increible preparada para ti"

         tts = gTTS(text=saludo, lang="es", tld="es")

         tts.save("/home/tammyai/Documents/RoBi/Text-To-Speech/saludo.mp3")
    def hilo_audio(self):
         hilo_reproduccion = threading.Thread(target=self.reproducir_audio)
         hilo_reproduccion.start()

    def reproducir_audio(self):
         os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/saludo.mp3")

    def pantalla_canciones(self):
         self.camera.release()
         CancionesAEscoger()
         print("cerrar ventana despues de cancion")
         self.destroy()

    def eliminar_caracteres_especiales(self,cadena,caracteres_permitidos):
         cadena_sin_especiales=""
         for caracter in cadena:
```

```
            if caracter in caracteres_permitidos:
                cadena_sin_especiales+=caracter
        return cadena_sin_especiales
```

Anexo 3. Script para las recomendaciones a los clientes

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERP
O_PRINCIPAL
from Formulario.familia import Familia
from Formulario.amigos import Amigos
from Formulario.pareja import Pareja
import Util.util_ventana as util_ventana
import cv2
from PIL import Image,ImageTk
import sys
import datetime
import os
import face_recognition
from gtts import gTTS
import threading
import time


class Recomendaciones(tk.Toplevel):

    def __init__(self):
        super().__init__()
        self.fondo=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/Recomendacion
es.png")
        self.familia_on=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/familiaon.png
")
        self.familia_off=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/familiaoff.pn
g")
        self.amigos_on=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/amigoson.png"
)
        self.amigos_off=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/amigosoff.png
")
        self.pareja_on=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/parejaon.png"
)
        self.pareja_off=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/parejaoff.png
")
        self.recognized_name = None
        self.recognition_attempts = 0
        self.codigo=".."
        self.reconocimiento_thread=None
        self.pantalla()
        self.fondo_pantalla()
        self.Botones()
        time.sleep(1)
        self.acompanantes()
        self.hilo_audio()
```

```
    def pantalla(self):
        self.title("RoBi")
        self.resizable(0,0)
        self.overrideredirect(True)
        util_ventana.centrar_ventana(self,1024,550)

    def fondo_pantalla(self):
        self.miframe=tk.Frame(self,bg=COLOR_BARRA_SUPERIOR, height=50)
        self.miframe.pack(side=tk.TOP)
        self.background=tk.Label(self.miframe, image=self.fondo)
        self.background.pack()

    def Botones(self):

boton_familia=tk.Button(self.miframe,image=self.familia_off,bd=0,highlightth
ickness=0,bg="#ffffff",command=self.familia)
        boton_familia.place(x=110,y=268)


boton_amigos=tk.Button(self.miframe,image=self.amigos_off,bd=0,highlightthic
kness=0,bg="#ffffff",command=self.amigos)
        boton_amigos.place(x=390,y=268)


boton_pareja=tk.Button(self.miframe,image=self.pareja_off,bd=0,highlightthic
kness=0,bg="#ffffff",command=self.pareja)
        boton_pareja.place(x=679,y=268)

        self.bind_hover_events1(boton_familia)
        self.bind_hover_events2(boton_amigos)
        self.bind_hover_events3(boton_pareja)

    def bind_hover_events1(self,boton_familia):
        boton_familia.bind("<Enter>", lambda
event:self.on_enter1(event,boton_familia))
        boton_familia.bind("<Leave>", lambda
event:self.on_leave1(event,boton_familia))

    def on_enter1(self,event,boton_familia):

boton_familia.config(image=self.familia_on,bd=0,highlightthickness=0,bg="#ff
ffff")

    def on_leave1(self,event,boton_familia):

boton_familia.config(image=self.familia_off,bd=0,highlightthickness=0,bg="#f
fffff")


    def bind_hover_events2(self,boton_amigos):
        boton_amigos.bind("<Enter>", lambda
event:self.on_enter2(event,boton_amigos))
        boton_amigos.bind("<Leave>", lambda
event:self.on_leave2(event,boton_amigos))

    def on_enter2(self,event,boton_amigos):

boton_amigos.config(image=self.amigos_on,bd=0,highlightthickness=0,bg="#ffff
ff")
```

```
    def on_leave2(self,event,boton_amigos):

boton_amigos.config(image=self.amigos_off,bd=0,highlightthickness=0,bg="#fff
fff")


    def bind_hover_events3(self,boton_pareja):
        boton_pareja.bind("<Enter>", lambda
event:self.on_enter3(event,boton_pareja))
        boton_pareja.bind("<Leave>", lambda
event:self.on_leave3(event,boton_pareja))

    def on_enter3(self,event,boton_pareja):

boton_pareja.config(image=self.pareja_on,bd=0,highlightthickness=0,bg="#ffff
ff")

    def on_leave3(self,event,boton_pareja):

boton_pareja.config(image=self.pareja_off,bd=0,highlightthickness=0,bg="#f5f
aff")

    def acompanantes(self):
        acompanantes=f"EL dia de hoy, ¿Con quién nos visitas?. ¿Con tu
familia, con tu pareja o con amigos? Escoge una de las opciones que ves en
pantalla."
        tts = gTTS(text=acompanantes, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-
Speech/acompanantes.mp3")

    def hilo_audio(self):
        hilo_reproduccion = threading.Thread(target=self.reproducir_audio)
        hilo_reproduccion.start()

    def reproducir_audio(self):
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/acompanantes.mp3")

    def familia(self):
        Familia()
        time.sleep(1)
        self.destroy()

    def amigos(self):
        Amigos()
        time.sleep(1)
        self.destroy()

    def pareja(self):
        Pareja()
        time.sleep(1)
        self.destroy()
```

Anexo 4. Código de Parejas

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERP
O_PRINCIPAL
```

```
import Util.util_ventana as util_ventana
import cv2
from PIL import Image,ImageTk
import sys
import os
import face_recognition
from gtts import gTTS
import threading
import time


class Pareja(tk.Toplevel):

    def __init__(self):
        super().__init__()
        self.fondo=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/final.png")
        self.amigos_nombre=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/amigos.png")
        self.amigos_foto=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/amigosfoto.pn
g")
        self.cerveza=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/cerveza.png")
        self.chuleta=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/chuleta.png")
        self.cocteles=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/cocteles.png"
)
        self.michelada=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/michelada.png
")

        self.pantalla()
        self.fondo_pantalla()
        self.imagenes()
        self.comida_amigos()
        self.hilo_audio()
        #time.sleep(34)
        #self.destroy()




    def pantalla(self):
        self.title("RoBi")
        self.resizable(0,0)
        self.overrideredirect(True)
        util_ventana.centrar_ventana(self,1024,550)

    def fondo_pantalla(self):
        self.miframe=tk.Frame(self,bg=COLOR_BARRA_SUPERIOR, height=50)
        self.miframe.pack(side=tk.TOP)

        self.background=tk.Label(self.miframe, image=self.fondo)
        self.background.pack()

    def imagenes(self):

imagen_amigos=tk.Label(self.miframe,image=self.amigos_nombre,bd=0,highlightt
hickness=0,bg="#ffffff")
        imagen_amigos.place(x=70,y=130)
```

```
foto_amigos=tk.Label(self.miframe,image=self.amigos_foto,bd=0,highlightthick
ness=0,bg="#ffffff")
        foto_amigos.place(x=70,y=270)


foto_chuleta=tk.Label(self.miframe,image=self.chuleta,bd=0,highlightthicknes
s=0,bg="#ffffff")
        foto_chuleta.place(x=430,y=80)


foto_cocteles=tk.Label(self.miframe,image=self.cocteles,bd=0,highlightthickn
ess=0,bg="#ffffff")
        foto_cocteles.place(x=700,y=300)


foto_cerveza=tk.Label(self.miframe,image=self.cerveza,bd=0,highlightthicknes
s=0,bg="#ffffff")
        foto_cerveza.place(x=450,y=280)


foto_michelada=tk.Label(self.miframe,image=self.michelada,bd=0,highlightthic
kness=0,bg="#ffffff")
        foto_michelada.place(x=750,y=75)


    def comida_amigos(self):
        familia=f"si lo que quieres es disfrutar con tu pareja, tenemos unas
copas de sangria que te encantaran y que complementaría perfectamente con
una deliciosa parrillada. Espero que te gusten estan recomendaciones y que
sirvan para que tu visita sea más agradable, ha sido un gusto verte. ¡Hasta
la Próxima!"
        tts = gTTS(text=familia, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-Speech/familia.mp3")

    def hilo_audio(self):
        hilo_reproduccion = threading.Thread(target=self.reproducir_audio)
        hilo_reproduccion.start()


    def reproducir_audio(self):
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/familia.mp3")
        time.sleep(5)
        self.destroy()
```

Anexo 5. Código Gracias

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERP
O_PRINCIPAL

import Util.util_ventana as util_ventana
import cv2
from PIL import Image,ImageTk
import sys
import datetime
import os
import face_recognition
```

```
from gtts import gTTS
import threading
import time



class Gracias(tk.Toplevel):

    def __init__(self):
        super().__init__()
        self.fondo=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/Fondo_final.p
ng")

        self.rostro_detectado=False
        self.reproduciendo_audio=False
        self.deteccion_facial_thread=None
        self.empezar_thread=None
        self.pantalla()
        self.fondo_pantalla()
        time.sleep(2)
        self.despedida()
        self.hilo_audio()


    def pantalla(self):
        self.title("RoBi")
        self.resizable(0,0)
        self.overrideredirect(True)
        util_ventana.centrar_ventana(self,1024,550)

    def fondo_pantalla(self):
        self.miframe=tk.Frame(self,bg=COLOR_BARRA_SUPERIOR, height=50)
        self.miframe.pack(side=tk.TOP)

        self.background=tk.Label(self.miframe, image=self.fondo)
        self.background.pack()

    def despedida(self):
        despedida=f"De parte de las personas que te queremos y de La Taberna,
te deseamos un feliz cumpleaños y esperamos que hayas disfrutado esta
sorpresa. No olvides recibir tu cortesía. Muchas Gracias por visitarnos, y
esperamos verles pronto. Adiooooos!"

        tts = gTTS(text=despedida, lang="es", tld="es")

        tts.save("/home/tammyai/Documents/RoBi/Text-To-Speech/despedida.mp3")
    def hilo_audio(self):
        hilo_reproduccion = threading.Thread(target=self.reproducir_audio)
        hilo_reproduccion.start()

    def reproducir_audio(self):
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/despedida.mp3")
        time.sleep(5)
        self.destroy()
```

Anexo 6. Codigo para grabar y pasa el texto a voz

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
```

```
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERPO_PRINCIPAL
import Util.util_ventana as util_ventana
from PIL import Image,ImageTk
import pyaudio
import wave
import threading
import time
import openai


def grabar():
    chunk = 1024  # Tamaño del fragmento de audio en bytes
    formato = pyaudio.paInt16  # Formato de audio
    canales = 2  # Número de canales (1 para mono, 2 para estéreo)
    frecuencia = 44100  # Frecuencia de muestreo (Hz)
    tiempo_grabacion = 2  # Tiempo de grabación en segundos

    # Inicializar PyAudio
    audio = pyaudio.PyAudio()

    # Abrir flujo de entrada
    stream_entrada = audio.open(format=formato,
                        channels=canales,
                        rate=frecuencia,
                        input=True,
                        frames_per_buffer=chunk)

    # Abrir archivo de salida
    archivo_salida = wave.open("/home/tammyai/Documents/RoBi/Speech-To-Text/grabacion.wav", "wb")
    archivo_salida.setnchannels(canales)
    archivo_salida.setsampwidth(audio.get_sample_size(formato))
    archivo_salida.setframerate(frecuencia)

    # Grabar audio
    datos = []
    for i in range(int(tiempo_grabacion * frecuencia / chunk)):
        datos_audio = stream_entrada.read(chunk)
        datos.append(datos_audio)

    # Detener y cerrar flujo de entrada
    stream_entrada.stop_stream()
    stream_entrada.close()

    # Escribir datos grabados en archivo de salida
    archivo_salida.writeframes(b''.join(datos))
    archivo_salida.close()

    # Imprimir mensaje de confirmación
    print("Grabación finalizada!")
    openai.api_key="sk-DN8feCbZhfl7J7xwmjxeT4BlbkFJWQE4k78xrU2GxmNJqo10"


    audio_file= open("/home/tammyai/Documents/RoBi/Speech-To-Text/grabacion.wav", "rb")
    transcription = openai.audio.transcriptions.create(
      model="whisper-1",
      file=audio_file
    )
    texto=transcription.text

    return texto
```

Anexo 7.

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERP
O_PRINCIPAL
import Util.util_ventana as util_ventana
import cv2
from PIL import Image,ImageTk
import sys
import os
import face_recognition
from gtts import gTTS
import threading
import time


class Familia(tk.Toplevel):

    def __init__(self):
        super().__init__()
        self.fondo=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/final.png")
        self.familia_nombre=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/familia.png")
        self.familia_foto=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/familiafoto.p
ng")
        self.parrillada=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/parrillada.pn
g")
        self.hamburguesa=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/hamburguesa.p
ng")
        self.alitas=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/alitas.png")
        self.michelada=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/michelada.png
")
        self.granizado=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/granizado.png
")

        self.pantalla()
        self.fondo_pantalla()
        self.imagenes()
        self.comida_familia()
        self.hilo_audio()




    def pantalla(self):
        self.title("RoBi")
        self.resizable(0,0)
        self.overrideredirect(True)
        util_ventana.centrar_ventana(self,1024,550)

    def fondo_pantalla(self):
```

```
        self.miframe=tk.Frame(self,bg=COLOR_BARRA_SUPERIOR, height=50)
        self.miframe.pack(side=tk.TOP)

        self.background=tk.Label(self.miframe, image=self.fondo)
        self.background.pack()

    def imagenes(self):

imagen_familia=tk.Label(self.miframe,image=self.familia_nombre,bd=0,highligh
tthickness=0,bg="#ffffff")
        imagen_familia.place(x=70,y=130)


foto_familia=tk.Label(self.miframe,image=self.familia_foto,bd=0,highlightthi
ckness=0,bg="#ffffff")
        foto_familia.place(x=70,y=270)


foto_parrillada=tk.Label(self.miframe,image=self.parrillada,bd=0,highlightth
ickness=0,bg="#ffffff")
        foto_parrillada.place(x=430,y=80)


foto_michelada=tk.Label(self.miframe,image=self.michelada,bd=0,highlightthic
kness=0,bg="#ffffff")
        foto_michelada.place(x=710,y=80)


foto_granizado=tk.Label(self.miframe,image=self.granizado,bd=0,highlightthic
kness=0,bg="#ffffff")
        foto_granizado.place(x=820,y=80)


foto_alitas=tk.Label(self.miframe,image=self.alitas,bd=0,highlightthickness=
0,bg="#ffffff")
        foto_alitas.place(x=430,y=295)


foto_hamburguesa=tk.Label(self.miframe,image=self.hamburguesa,bd=0,highlight
thickness=0,bg="#ffffff")
        foto_hamburguesa.place(x=700,y=300)


    def comida_familia(self):
        familia=f"Para compartir en familia, te recomendamos esta deliciosa
parrillada que incluye limonada y ensalada. Puedes acompañarlo con una
michelada real, y para los más pequeños te ofrecemos un exquisito granizado
goloso.          Si te gustan las alitas, podemos ofrecerte esta magnífica
promoción de 2 alitas más la bebida incluida. Y si lo que más te gusta son
las hamburguesas, la promoción 1 te encantará ya que incluye un combo de 6
alitas y una hamburguesa más la bebida.          Espero que mis
recomendaciones te ayuden.         Ha sido un gusto verte. Espero verte
pronto"
        tts = gTTS(text=familia, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-Speech/familia.mp3")

    def hilo_audio(self):
        hilo_reproduccion = threading.Thread(target=self.reproducir_audio)
        hilo_reproduccion.start()

    def reproducir_audio(self):
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/familia.mp3")
```

```
        time.sleep(5)
        self.destroy()
```

Anexo 8.

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERP
O_PRINCIPAL
from Formulario.reconocimiento_facial import ReconocimientoDeClientes
import Util.util_ventana as util_ventana
import cv2
from PIL import Image,ImageTk
import sys
import datetime
import os
import face_recognition
from gtts import gTTS
import threading
import time



class FormularioMaestroDesign(tk.Tk):

    def __init__(self):
        super().__init__()
        self.fondo=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/Fondo1.png")

self.empezar_on=ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagen
es/empezar_on.png")

self.empezar_off=ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Image
nes/empezar_off.png")

self.logo=ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/log
o.png")
        self.rostro_detectado=False
        self.reproduciendo_audio=False
        self.deteccion_facial_thread=None
        self.empezar_thread=None
        self.pantalla()
        self.fondo_pantalla()
        self.botones()

    def pantalla(self):
        self.title("RoBi")
        self.resizable(0,0)
        util_ventana.centrar_ventana(self,1024,550)
        #self.overrideredirect(True)

    def fondo_pantalla(self):
        self.miframe=tk.Frame(self,bg=COLOR_BARRA_SUPERIOR, height=50)
        self.miframe.pack(side=tk.TOP)

        self.background=tk.Label(self.miframe, image=self.fondo)
        self.background.pack()
```

```
    def botones(self):

boton_registro=tk.Button(self.miframe,image=self.empezar_on,bd=0,highlightth
ickness=0,bg="#ffffff",command=self.iniciar_funcion_empezar)
       boton_registro.place(x=450,y=440)
       self.bind_hover_events(boton_registro)


boton_iniciar=tk.Button(self.miframe,image=self.logo,bd=0,highlightthickness
=0,bg="#5b55d4",command=self.iniciar_deteccion_facial)
       boton_iniciar.place(x=59, y=6)

       #self.deteccion_rostros()

    def bind_hover_events(self,boton_registro):
       boton_registro.bind("<Enter>", lambda
event:self.on_enter(event,boton_registro))
       boton_registro.bind("<Leave>", lambda
event:self.on_leave(event,boton_registro))

    def on_enter(self,event,boton_registro):

boton_registro.config(image=self.empezar_on,bd=0,highlightthickness=0,bg="#f
5faff")

    def on_leave(self,event,boton_registro):

boton_registro.config(image=self.empezar_off,bd=0,highlightthickness=0,bg="#
f5faff")

    def iniciar_deteccion_facial(self):
       self.estado=True
       print(self.estado)
       print(self.deteccion_facial_thread)
       if not self.deteccion_facial_thread:
          self.rostro_detectado=False

self.deteccion_facial_thread=threading.Thread(target=self.deteccion_rostros)
          self.deteccion_facial_thread.start()

    def iniciar_funcion_empezar(self):
       self.camara.release()
       #self.deteccion_facial_thread=None
       time.sleep(1)
       if self.deteccion_facial_thread:
          print("hola")
          self.estado=False
          self.deteccion_facial_thread=None
          print(self.deteccion_facial_thread)
          ReconocimientoDeClientes()


    def deteccion_rostros(self):
       self.camara=cv2.VideoCapture(0)
       print(self.estado)
       while self.estado:
          if not self.estado:
             break
          capture,frame=self.camara.read()
          frame=cv2.cvtColor(frame,cv2.COLOR_BGR2RGB)

          face_locations=face_recognition.face_locations(frame)
          print (f"rostros detectado = {self.rostro_detectado}")
```

```
            print(f"audio {self.reproduciendo_audio}")
            if len(face_locations)>0 and not self.reproduciendo_audio:
                rostro_detectado=True
                print("se detecto un rostro")
                self.mensaje()
                self.hilo_audio()

            elif not len(face_locations) and self.reproduciendo_audio:

                print("no se detecto un rostro")
                self.reproduciendo_audio = False
                rostro_detectado=False

    def mensaje(self):
        bienvenida=f"Hola, Bienvenido a La Taberna. Soy Shaya. Pulsa Empezar"
        tts = gTTS(text=bienvenida, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-
Speech/bienvenida.mp3")

    def hilo_audio(self):
        self.reproduciendo_audio = True
        hilo_reproduccion = threading.Thread(target=self.reproducir_audio)
        hilo_reproduccion.start()

    def reproducir_audio(self):
        self.reproduciendo_audio = True
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/bienvenida.mp3")
        time.sleep(5)
        self.reproduciendo_audio = False
```

Anexo 9.

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERP
O_PRINCIPAL
import Util.util_ventana as util_ventana
#from Util.reconocimiento_facial import main
#import Util.reconocimiento_facial as reconocimiento
from Formulario.gracias import Gracias
import cv2
from PIL import Image,ImageTk
import sys
import pygame
import pygame.mixer
import os
from gtts import gTTS
import threading
import time

class CancionesAEscoger(tk.Toplevel):

    def __init__(self):
        super().__init__()
        self.fondo=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/Fondo_cumplea
ños.png")
        self.feliz_cumpleaños_off=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/felizcumpleañ
```

```
osoff.png")
        self.feliz_cumpleaños_on=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/felizcumpleañ
oson.png")
        self.que_dios_off=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/quediostebend
igaoff.png")
        self.que_dios_on=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/quediostebend
igaon.png")
        self.las_mañanitas_off=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/lasmañanitaso
ff.png")
        self.las_mañanitas_on=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/lasmañanitaso
n.png")
        self.en_tu_cumpleaños_off=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/entucumpleaño
soff.png")
        self.en_tu_cumpleaños_on=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/entucumpleaño
son.png")
        #self.pastel=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/pastel.png")
        self.en_reproduccion=False
        self.current_song=None
        self.audio_files={
            "FELIZ
CUMPLEAÑOS":"/home/tammyai/Documents/RoBi/Canciones/cumpleaños_feliz.mp3",
            "QUE DIOS TE
BENDIGA":"/home/tammyai/Documents/RoBi/Canciones/Que_dios_te_bendiga.mp3",
            "LAS
MAÑANITAS":"/home/tammyai/Documents/RoBi/Canciones/Las_mañanitas.mp3",
            "EN TU
CUMPLEAÑOS":"/home/tammyai/Documents/RoBi/Canciones/lejos_de_ti.mp3"
        }
        self.image_path={
            "FELIZ
CUMPLEAÑOS":"/home/tammyai/Documents/RoBi/Imagenes/pastel.png",
            "LAS
MAÑANITAS":"/home/tammyai/Documents/RoBi/Imagenes/mexico.png",
            "QUE DIOS TE
BENDIGA":"/home/tammyai/Documents/RoBi/Imagenes/colombia.png",
            "EN TU
CUMPLEAÑOS":"/home/tammyai/Documents/RoBi/Imagenes/ecuador.png"

        }

        self.tiempo_path={
            "FELIZ CUMPLEAÑOS":47,
            "LAS MAÑANITAS":66,
            "QUE DIOS TE BENDIGA":131,
            "EN TU CUMPLEAÑOS":106

        }

        self.pantalla()
        self.fondo_pantalla()
        self.botones()
        time.sleep(5)
        self.escoge()
        self.hilo_audio()
```

```
    def pantalla(self):
        self.title("RoBi")
        self.resizable(0,0)
        self.overrideredirect(True)
        util_ventana.centrar_ventana(self,1024,550)

    def fondo_pantalla(self):
        self.miframe=tk.Frame(self,bg=COLOR_BARRA_SUPERIOR, height=50)
        self.miframe.pack(side=tk.TOP)

        self.background=tk.Label(self.miframe, image=self.fondo)
        self.background.pack()

    def botones(self):
    #Boton 1:FELIZ CUMPLEAÑOS

boton_feliz_cumpleaños=tk.Button(self.miframe,image=self.feliz_cumpleaños_of
f,bd=0,highlightthickness=0,bg="#f5faff",
command=lambda:self.play_song_and_show_image("FELIZ CUMPLEAÑOS"))
        boton_feliz_cumpleaños.place(x=73,y=209)

    #Boton 2:QUE DIOS TE BENDIGA

boton_que_dios_te_bendiga=tk.Button(self.miframe,image=self.que_dios_off,bd=
0,highlightthickness=0,bg="#f5faff",command=lambda:self.play_song_and_show_i
mage("QUE DIOS TE BENDIGA"))
        boton_que_dios_te_bendiga.place(x=73,y=281)

    #Boton 3:LAS MAÑANITAS

boton_las_mañanitas=tk.Button(self.miframe,image=self.las_mañanitas_off,bd=0
,highlightthickness=0,bg="#f5faff",command=lambda:self.play_song_and_show_im
age("LAS MAÑANITAS"))
        boton_las_mañanitas.place(x=73,y=353)

    #Boton 4:EN TU CUMPLEAÑOS

boton_en_tu_cumpleaños=tk.Button(self.miframe,image=self.en_tu_cumpleaños_of
f,bd=0,highlightthickness=0,bg="#f5faff",command=lambda:self.play_song_and_s
how_image("EN TU CUMPLEAÑOS"))
        boton_en_tu_cumpleaños.place(x=73,y=425)

        self.bind_hover_events1(boton_feliz_cumpleaños)
        self.bind_hover_events2(boton_que_dios_te_bendiga)
        self.bind_hover_events3(boton_las_mañanitas)
        self.bind_hover_events4(boton_en_tu_cumpleaños)


    #Boton 1:FELIZ CUMPLEAÑOS
    def bind_hover_events1(self,boton_feliz_cumpleaños):
        boton_feliz_cumpleaños.bind("<Enter>", lambda
event:self.on_enter1(event,boton_feliz_cumpleaños))
        boton_feliz_cumpleaños.bind("<Leave>", lambda
event:self.on_leave1(event,boton_feliz_cumpleaños))

    #Boton 2:QUE DIOS TE BENDIGA
    def bind_hover_events2(self,boton_que_dios_te_bendiga):
        boton_que_dios_te_bendiga.bind("<Enter>", lambda
event:self.on_enter2(event,boton_que_dios_te_bendiga))
        boton_que_dios_te_bendiga.bind("<Leave>", lambda
event:self.on_leave2(event,boton_que_dios_te_bendiga))
```

```
    #Boton 3:LAS MAÑANITAS
    def bind_hover_events3(self,boton_las_mañanitas):
       boton_las_mañanitas.bind("<Enter>", lambda
event:self.on_enter3(event,boton_las_mañanitas))
       boton_las_mañanitas.bind("<Leave>", lambda
event:self.on_leave3(event,boton_las_mañanitas))

    #Boton 4:EN TU CUMPLEAÑOS
    def bind_hover_events4(self,boton_en_tu_cumpleaños):
       boton_en_tu_cumpleaños.bind("<Enter>", lambda
event:self.on_enter4(event,boton_en_tu_cumpleaños))
       boton_en_tu_cumpleaños.bind("<Leave>", lambda
event:self.on_leave4(event,boton_en_tu_cumpleaños))

    #Boton 1:FELIZ CUMPLEAÑOS
    def on_enter1(self,event,boton_feliz_cumpleaños):

boton_feliz_cumpleaños.config(image=self.feliz_cumpleaños_on,bd=0,highlightt
hickness=0,bg="#f5faff")

    #Boton 2:QUE DIOS TE BENDIGA
    def on_enter2(self,event,boton_que_dios_te_bendiga):

boton_que_dios_te_bendiga.config(image=self.que_dios_on,bd=0,highlightthickn
ess=0,bg="#f5faff")

    #Boton 3:LAS MAÑANITAS
    def on_enter3(self,event,boton_las_mañanitas):

boton_las_mañanitas.config(image=self.las_mañanitas_on,bd=0,highlightthickne
ss=0,bg="#f5faff")

    #Boton 4:EN TU CUMPLEAÑOS
    def on_enter4(self,event,boton_en_tu_cumpleaños):

boton_en_tu_cumpleaños.config(image=self.en_tu_cumpleaños_on,bd=0,highlightt
hickness=0,bg="#f5faff")

    #Boton 1:FELIZ CUMPLEAÑOS
    def on_leave1(self,event,boton_feliz_cumpleaños):

boton_feliz_cumpleaños.config(image=self.feliz_cumpleaños_off,bd=0,highlight
thickness=0,bg="#f5faff")

    #Boton 2:QUE DIOS TE BENDIGA
    def on_leave2(self,event,boton_que_dios_te_bendiga):

boton_que_dios_te_bendiga.config(image=self.que_dios_off,bd=0,highlightthick
ness=0,bg="#f5faff")

    #Boton 3:LAS MAÑANITAS
    def on_leave3(self,event,boton_las_mañanitas):

boton_las_mañanitas.config(image=self.las_mañanitas_off,bd=0,highlightthickn
ess=0,bg="#f5faff")

    #Boton 4:EN TU CUMPLEAÑOS
    def on_leave4(self,event,boton_en_tu_cumpleaños):

boton_en_tu_cumpleaños.config(image=self.en_tu_cumpleaños_off,bd=0,highlight
thickness=0,bg="#f5faff")
```

```
    def play_song_and_show_image(self,button_name):

        audio_file=self.audio_files[button_name]
        image_path=self.image_path[button_name]
        tiempo_path=self.tiempo_path[button_name]

        pygame.mixer.init()
        try:
            pygame.mixer.music.load(audio_file)
            #image=Image.open(image_path)
        except pygame.error:
            print(f"Error al cargar el archivo de audio o la imagen")
            return

        image_tk=ImageTk.PhotoImage(file=image_path)


image_label=tk.Label(self.miframe,bd=0,highlightthickness=0,bg="#f5faff")
        image_label.config(image=image_tk)
        image_label.image=image_tk
        image_label.place(x=655,y=145)

        def reproducir_en_hilo():
            pygame.mixer.music.play()
            print(tiempo_path)
            time.sleep(tiempo_path)
            print("termina la cancion")
            Gracias()
            self.destroy()

        hilo_reproducciones=threading.Thread(target=reproducir_en_hilo)
        hilo_reproducciones.start()

    def escoge(self):
        escoge=f"por tu cumpleaños seleciona la canción que más te guste"
        tts = gTTS(text=escoge, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-Speech/escoge.mp3")

    def hilo_audio(self):
        hilo_reproduccion = threading.Thread(target=self.reproducir_audio)
        hilo_reproduccion.start()

    def reproducir_audio(self):
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/escoge.mp3")
```

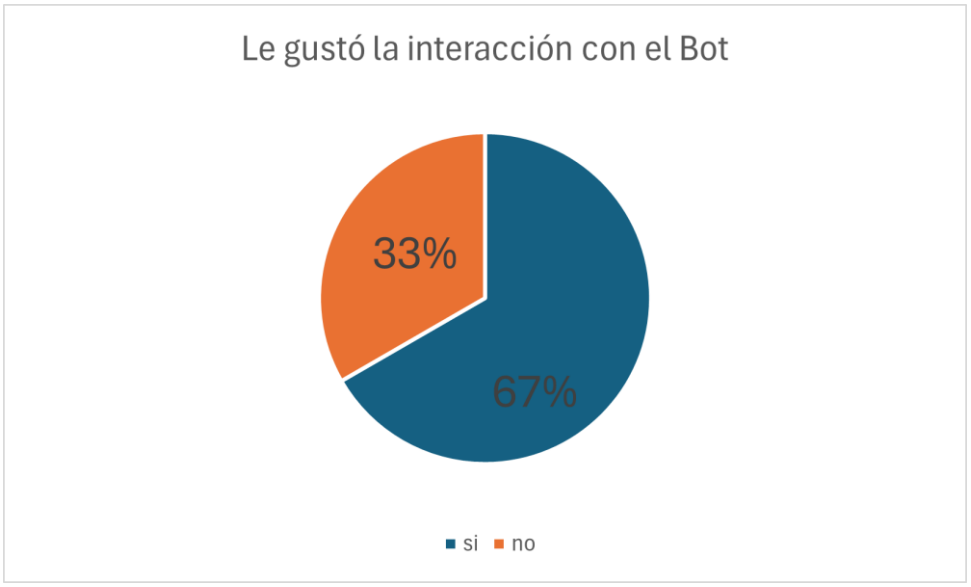
Anexo 10.

```
import tkinter as tk
from tkinter import *
from tkinter import font
from config import
COLOR_MENU_LATERAL,COLOR_MENU_CURSOR_ENCIMA,COLOR_BARRA_SUPERIOR,COLOR_CUERP
O_PRINCIPAL
import Util.util_ventana as util_ventana
import cv2
from PIL import Image,ImageTk
import sys
import os
import face_recognition
from gtts import gTTS
```

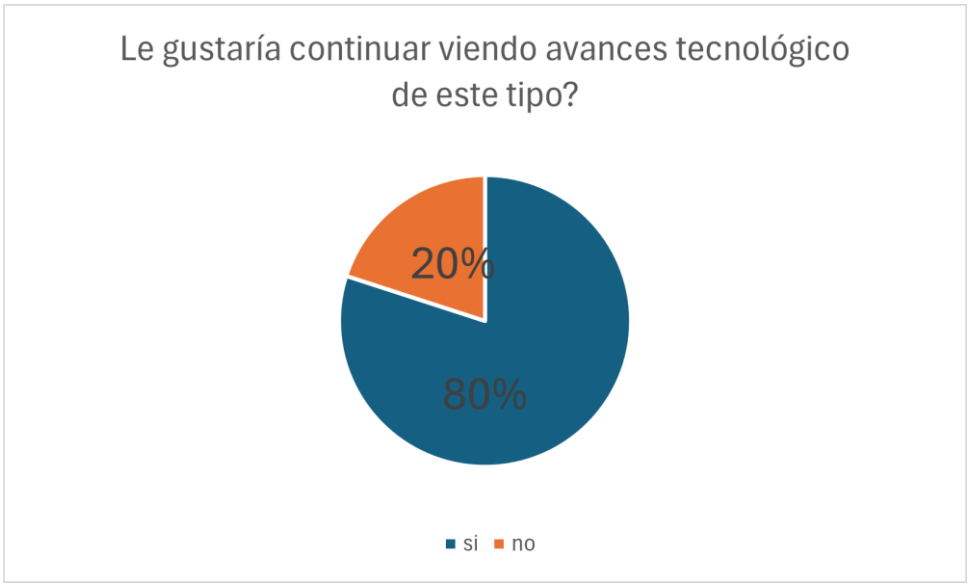
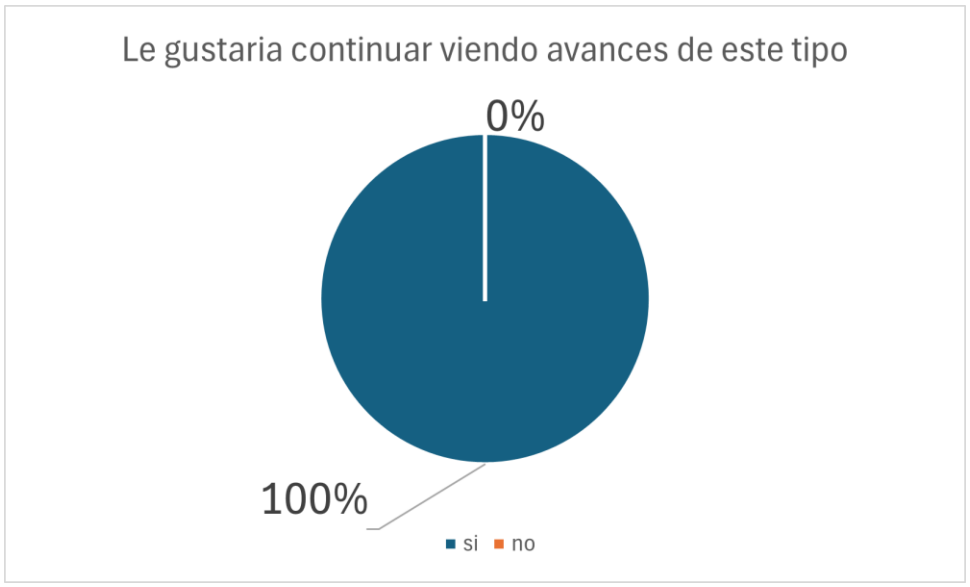
```
import threading
import time


class Amigos(tk.Toplevel):

    def __init__(self):
        super().__init__()
        self.fondo=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/final.png")
        self.amigos_nombre=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/amigos.png")
        self.amigos_foto=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/amigosfoto.pn
g")
        self.cerveza=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/cerveza.png")
        self.chuleta=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/chuleta.png")
        self.cocteles=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/cocteles.png"
)
        self.michelada=
ImageTk.PhotoImage(file="/home/tammyai/Documents/RoBi/Imagenes/michelada.png
")

        self.pantalla()
        self.fondo_pantalla()
        self.imagenes()
        self.comida_amigos()
        self.hilo_audio()




    def pantalla(self):
        self.title("RoBi")
        self.resizable(0,0)
        self.overrideredirect(True)
        util_ventana.centrar_ventana(self,1024,550)

    def fondo_pantalla(self):
        self.miframe=tk.Frame(self,bg=COLOR_BARRA_SUPERIOR, height=50)
        self.miframe.pack(side=tk.TOP)

        self.background=tk.Label(self.miframe, image=self.fondo)
        self.background.pack()

    def imagenes(self):

imagen_amigos=tk.Label(self.miframe,image=self.amigos_nombre,bd=0,highlightt
hickness=0,bg="#ffffff")
        imagen_amigos.place(x=70,y=130)


foto_amigos=tk.Label(self.miframe,image=self.amigos_foto,bd=0,highlightthick
ness=0,bg="#ffffff")
        foto_amigos.place(x=70,y=270)


foto_chuleta=tk.Label(self.miframe,image=self.chuleta,bd=0,highlightthicknes
s=0,bg="#ffffff")
```

```
        foto_chuleta.place(x=430,y=80)


foto_cocteles=tk.Label(self.miframe,image=self.cocteles,bd=0,highlightthickn
ess=0,bg="#ffffff")
        foto_cocteles.place(x=700,y=300)


foto_cerveza=tk.Label(self.miframe,image=self.cerveza,bd=0,highlightthicknes
s=0,bg="#ffffff")
        foto_cerveza.place(x=450,y=280)


foto_michelada=tk.Label(self.miframe,image=self.michelada,bd=0,highlightthic
kness=0,bg="#ffffff")
        foto_michelada.place(x=750,y=75)


    def comida_amigos(self):
        familia=f"Lo mejor para pasar un buen momento con los amigos es
compartir una deliciosas micheladas. Tienes una gran variedad de cocteles y
cervezas artesanales que seguro te encantarán. Nada mejor que acompañar tus
bebidas favoritas con una exquisita chuleta o pollo al grill. Espero que
estas recomendaciones te ayuden a ti y a tus amigos a disfrutar tu visita
con nosotros, ha sido un gusto verte. ¡Hasta la Próxima!"
        tts = gTTS(text=familia, lang="es", tld="es")
        tts.save("/home/tammyai/Documents/RoBi/Text-To-Speech/familia.mp3")

    def hilo_audio(self):
        hilo_reproduccion = threading.Thread(target=self.reproducir_audio)
        hilo_reproduccion.start()


    def reproducir_audio(self):
        os.system("mpg321 /home/tammyai/Documents/RoBi/Text-To-
Speech/familia.mp3")
        time.sleep(5)
        self.destroy()
```

Anexo 11: Gráfico de aceptación



Anexo 12: Mejoras para el bot



Anexo 13: Interés de los clientes ante la IA