



MÓDULO PROYECTO

CFGS Desarrollo de Aplicaciones
Multiplataforma
Informática y Comunicaciones

<Hands>

Tutor individual: <Silvia Pedrón Hermosa>

Tutor colectivo: <Cristina Silván Pardo>

Año: <2023/24 >

Fecha de presentación: (28/05/2024)

Nombre y Apellidos: Alonso Soriano Carrasco
Email: alonsosorianoc1@gmail.com

Tabla de contenido

1	Identificación proyecto	5
2	Organización de la memoria	5
3	Descripción general del proyecto	6
3.1	Objetivos	7
3.2	Cuestiones metodológicas	7
3.3	Entorno de trabajo	8
4	Descripción general del producto.....	9
4.1	Visión general del sistema.....	10
4.2	Descripción de métodos.....	12
4.3	Despliegue de la aplicación	13
5	Planificación y presupuesto	13
6	Documentación técnica	14
6.1	Especificación de requisitos	14
6.2	Análisis del sistema	15
6.3	Diseño del sistema.....	16
6.3.1	Almacenamiento y tratamiento de datos	18
6.3.2	Diseño de la interfaz de usuario	18
6.3.3	Diseño de la aplicación.....	21
6.4	Implementación	23
6.4.1	Entorno de desarrollo	24
6.4.2	Estructura del código	25
6.4.3	Cuestiones del diseño e implementación reseñables.....	32
6.5	Pruebas.....	34
7	Manuales	35
7.1	Manual de usuario.....	35
7.2	Manual de instalación	42

8	Conclusiones y posibles ampliaciones	43
9	Bibliografía	44
10	Anexos	44

TABLA DE ILUSTRACIONES

ILUSTRACIÓN 1 PRIMERA VERSIÓN DE LA APLICACIÓN	9
ILUSTRACIÓN 2 DISEÑO Y DISPOSICIÓN BOTONES WEB	19
ILUSTRACIÓN 3 DISEÑO INICIAL EN BASE A COLORES DE IMÁGENES	20
ILUSTRACIÓN 4 ESTRUCTURA, ESTILO Y COLORES HANDS	20
ILUSTRACIÓN 5 PANTALLA ABECEDARIO	20
ILUSTRACIÓN 6 ESQUEMA PANTALLAS HANDS	21
ILUSTRACIÓN 7 ESQUEMA ESTRUCTURA DETECCIÓN DE SIGNOS	22
ILUSTRACIÓN 8 ESQUEMA GENERAL DE LA APLICACIÓN HANDS	23
ILUSTRACIÓN 9 ESTRUCTURA DE CARPETAS	26
ILUSTRACIÓN 10 DECLARACIÓN COLORES EN CSS	27
ILUSTRACIÓN 11 ESTRUCTURA MENÚ EN HTML	28
ILUSTRACIÓN 12 ESPACIO RESERVADO PARA JAVASCRIPT	29
ILUSTRACIÓN 13 CÓDIGO JAVASCRIPT ACCIÓN DETECCIÓN	29
ILUSTRACIÓN 14 CÓDIGO CONDICIONALES	30
ILUSTRACIÓN 15 CÓDIGO OBTENER ÁNGULO	30
ILUSTRACIÓN 16 IMPORTACIONES FLASK Y FUNCIONES DE DETECCIÓN	31
ILUSTRACIÓN 17 LLAMADAS A LAS PANTALLAS HTML	32
ILUSTRACIÓN 18 INICIO WINDOWS10 CÁMARA1	33
ILUSTRACIÓN 19 CONFIGURACIÓN CÁMARA2	33
ILUSTRACIÓN 20 ESTRUCTURA Y ESTILO PANTALLA INICIO	36
ILUSTRACIÓN 21 ESTRUCTURA Y ESTILO PANTALLA INICIO 2	36
ILUSTRACIÓN 22 ESTRUCTURA Y ELEMENTOS PANTALLA ABECEDARIO	37
ILUSTRACIÓN 23 AMPLIACIÓN PANTALLA ABECEDARIO	38
ILUSTRACIÓN 24 ESTRUCTURA PANTALLA SALUDOS	39
ILUSTRACIÓN 25 ESTRUCTURA PANTALLA NÚMEROS	40
ILUSTRACIÓN 26 ESTRUCTURA PANTALLA DETECCIÓN 1	40
ILUSTRACIÓN 27 BARRA DE HERRAMIENTAS DE WINDOWS10	41
ILUSTRACIÓN 28 DETECCIÓN DE SIGNOS EN TIEMPO REAL 1	41
ILUSTRACIÓN 29 DETECCIÓN DE SIGNOS EN TIEMPO REAL 2	42
ILUSTRACIÓN 30 DESCARGA DE PYTHON	43

1 Identificación proyecto

Proyecto realizado por el alumno Alonso Soriano Carrasco como parte del módulo de proyecto del ciclo formativo de grado superior de desarrollo de aplicaciones multiplataforma.

El proyecto se identifica con el nombre de “Hands” y se desarrolla partiendo de la base de los conocimientos adquiridos por el alumno a lo largo de los dos años académicos del ciclo formativo.

Durante este periodo, el alumno ha adquirido y desarrollado una serie de conocimientos y aptitudes relacionadas con el desarrollo de aplicaciones. Campos asentados a través de una serie de módulos profesionales tales como “Programación”, “Base de datos”, “Sistemas informáticos” y demás asignaturas.

A mayores de todo lo aprendido, el alumno ha tenido que investigar, desarrollar y probar una serie de conceptos nuevos no impartidos por el centro. A modo de reto o prueba y recapitulación de este periodo académico.

Puesto de dicha forma y en posesión de ciertos conocimientos, el alumno propone la creación de una aplicación destinada al aprendizaje de la lengua de signos. Esta aplicación se desarrollará durante el periodo comprendido entre los meses de marzo a mayo. La aplicación contará con una serie de funcionalidades detalladas a lo largo de este documento.

Como resultado final del desarrollo del proyecto de desarrollo de aplicaciones multiplataforma de Alonso se cuenta con una aplicación tipo web con carácter didáctico, donde el usuario puede consultar una información en específico basada en imágenes y textos. Como funcionalidad a destacar se cuenta con la capacidad de detección de manos y gestos relacionados con la lengua de signos mediante algoritmos ligados a la inteligencia artificial y lógica aplicada a su entorno.

2 Organización de la memoria

En la organización de esta memoria, se desplegará un detallado recorrido por los diferentes aspectos que componen el proyecto "Hands". Desde su identificación inicial hasta las posibles ampliaciones futuras, se presentará una visión completa y estructurada de todo el proceso de desarrollo.

La memoria se inicia con una clara identificación del proyecto, donde se presenta el nombre del proyecto, su contexto en el marco educativo del ciclo formativo de grado superior de desarrollo de aplicaciones multiplataforma, así como una breve introducción al mismo.

A continuación, se ofrece una descripción general del proyecto, que aborda sus objetivos, las metodologías empleadas para alcanzarlos y el entorno de trabajo en el que se desenvuelve el desarrollo de la aplicación. Esta sección proporciona una visión panorámica de la dirección y los recursos involucrados en el proyecto.

Luego, se detalla la descripción del producto, incluyendo una visión general del sistema, los métodos empleados en su desarrollo y el despliegue de la aplicación. Esta sección es fundamental para comprender las características y funcionalidades del producto final, así como su alcance y potencial de implementación.

La planificación y presupuesto del proyecto se abordan en la siguiente sección, donde se establecen los cronogramas y recursos necesarios para llevar a cabo el desarrollo de la aplicación, así como los posibles costos asociados a su realización.

La documentación técnica ocupa un lugar central en la memoria, con secciones dedicadas a la especificación de requisitos, el análisis y diseño del sistema, así como la implementación y pruebas realizadas. Estas secciones ofrecen una visión detallada del proceso de desarrollo y validación del producto.

Los manuales de usuario e instalación proporcionan orientación práctica para los usuarios finales y administradores de la aplicación, facilitando su adopción y uso eficiente.

Finalmente, se presentan las conclusiones del proyecto, junto con posibles ampliaciones o desarrollos futuros, así como una bibliografía y anexos que respaldan y complementan la información presentada en la memoria.

En resumen, la organización de esta memoria sigue una estructura clara y coherente, que guía al lector a través de todas las etapas del proyecto "Hands", desde su concepción hasta su posible evolución futura.

3 Descripción general del proyecto

"Hands" es el proyecto desarrollado en busca del acercamiento y la difusión en forma de aprendizaje de la lengua de signos.

Este proyecto se lleva a cabo en un marco de una aplicación web basada en un servidor Flask de Python con pantallas estructuradas con HTML y con estilos, colores y acciones en CSS. Contando todo ello con eventos y funcionalidades bien definidas en JavaScript.

La aplicación cuenta con una serie de pantallas que muestran, en forma de galería de imágenes, diferentes tipos de datos útiles para el aprendizaje. Desde el abecedario o saludos hasta los números.

Este tipo de pantallas permiten la interacción de tal modo que el usuario puede acceder a las imágenes y verlas de forma detallada. En alguno de los casos, como ocurre con la pantalla de saludos, estas imágenes son móviles. Esto es debido a que gran parte de este tipo de comunicación se hace de forma dinámica y está basada en el movimiento de la persona y en la ejecución de varios gestos en un mismo marco comunicativo.

A mayores, se cuenta con la funcionalidad de la detección de los gestos que pueda hacer el usuario cuando practica la lengua de signos. Esta funcionalidad se realiza mediante la librería Mediapipe desarrollada por Google. Mediapipe es un marco de trabajo para construir pipelines aplicados de aprendizaje automático multimodales (como de video, audio, datos de series de tiempo, etc.), y

para diferentes plataformas como, por ejemplo, Android, iOS, Web, y dispositivos perimetrales. MediaPipe provee múltiples características, incluyendo detección de rostros, seguimiento de manos, detección de gestos, y detección de objetos.

A las funcionalidades ofrecidas por MediaPipe se aplica un procesamiento mediante lógica hecha en Python para la detección de ciertos gestos, como pueden ser las letras vocales de LSE (Lengua de Signos Española).

3.1 Objetivos

El proyecto “Hands” tiene como objetivos claros y definidos dos grandes pilares, el primero, es la difusión en forma de aprendizaje de la lengua de signos.

Desarrollando una aplicación didáctica de fácil manejo y fácil implantación es viable la puesta en escena en centros didácticos, como pueden ser colegios, asociaciones, academias y demás lugares destinados o propicios a estas prácticas.

El segundo objetivo fundamental, de cara esta vez para el alumno, es el aprendizaje de nuevas tecnologías, procesos y metodologías de trabajo aplicables al entorno laboral.

Esto puede ser, como ejemplo, el control y distribución del tiempo basándose en la relación “Trabajo-Avances-Tiempo”, aprendiendo así a tener control sobre estos tres factores importantes a la hora de llevar a cabo estos tipos de proyectos.

3.2 Cuestiones metodológicas

En el transcurso del desarrollo de la aplicación "Hands", se ha aplicado un enfoque altamente flexible y adaptable, que ha surgido de la necesidad de enfrentar los desafíos inherentes a un proyecto en solitario, sin la estructura formal de metodologías establecidas o la colaboración directa con compañeros. Este enfoque ha sido esencial para abordar las diversas etapas del proyecto de manera dinámica y eficiente, permitiendo ajustes y mejoras continuas en función de los hallazgos y resultados obtenidos en cada fase.

El proceso de desarrollo se ha caracterizado por una exploración iterativa y experimental de múltiples entornos de desarrollo y tecnologías. Desde el principio, se han llevado a cabo pruebas exhaustivas con diferentes plataformas, incluyendo opciones para dispositivos móviles, escritorio y web. Esta fase inicial de investigación y experimentación ha sido fundamental para comprender las fortalezas y limitaciones de cada entorno, así como para definir la dirección futura del proyecto.

Entre las alternativas consideradas, se evaluó el uso de Flutter para el desarrollo móvil, lo que inicialmente parecía prometedor. Sin embargo, a medida que se profundizaba en el desarrollo, se identificaron limitaciones significativas en cuanto a la capacidad de detección y procesamiento en tiempo real de imágenes, lo que llevó a descartar esta opción en favor de buscar alternativas más adecuadas para los requisitos del proyecto.

Asimismo, se exploraron opciones para la creación de una interfaz gráfica de usuario, incluyendo la utilización de Tkinter con PyCharm. Si bien esta combinación demostró ser funcional en algunos aspectos, se determinó que la interacción con la aplicación era demasiado lenta para cumplir con las expectativas de rendimiento y usabilidad establecidas para el proyecto.

La decisión final de utilizar Visual Studio Code como entorno de desarrollo para la implementación de la aplicación web se basó en una cuidadosa evaluación de las necesidades técnicas y funcionales del proyecto. La versatilidad y compatibilidad de Visual Studio Code con una amplia gama de tecnologías, incluyendo CSS, JavaScript, HTML y Python, proporcionaron la base necesaria para el desarrollo y diseño del producto.

Además, Visual Studio Code ofreció una serie de ventajas adicionales, como la flexibilidad y ampliación a través de plugins y extensiones, que facilitaron el proceso de desarrollo y la detección de errores de sintaxis. Estas capacidades adicionales fueron particularmente útiles para optimizar el flujo de trabajo y mejorar la productividad durante el desarrollo de la aplicación.

En cuanto a las pruebas locales, se optó por utilizar el navegador Google Chrome debido a su amplia adopción y compatibilidad con las tecnologías web. Esta elección permitió realizar pruebas exhaustivas y depuraciones en un entorno controlado antes de la implementación final, garantizando así la estabilidad y el rendimiento del producto.

El enfoque metodológico adoptado en el desarrollo de la aplicación "Hands" se ha caracterizado por su adaptabilidad y pragmatismo, permitiendo ajustes continuos en función de la viabilidad técnica y la eficacia observada en cada etapa del proyecto.

3.3 Entorno de trabajo

A lo largo del proceso por el cual ha pasado esta aplicación, se han probado diferentes entornos de desarrollo. Cabe destacar que, en el proceso de análisis de requisitos, búsqueda de información y diseño se ha ido cambiando de plataforma destinada (Móvil, escritorio o web) así como variando entre frameworks y lenguajes hasta llegar al punto final.

Se ha probado el desarrollo de la aplicación móvil utilizando el framework multiplataforma Flutter para codificación en Dart. Esta idea se desestimó al no ser compatible con la detección y el procesamiento por lógica de las imágenes en tiempo real.

También se hicieron intentos con la herramienta GUI de Python Tkinter codificada con el IDE PyCharm de la compañía JetBrains siendo útil y funcional de una forma directa y efectiva para enlazar con MediaPipe. Esta idea se desestimó debido a que la interacción usuario-app era demasiado lenta.

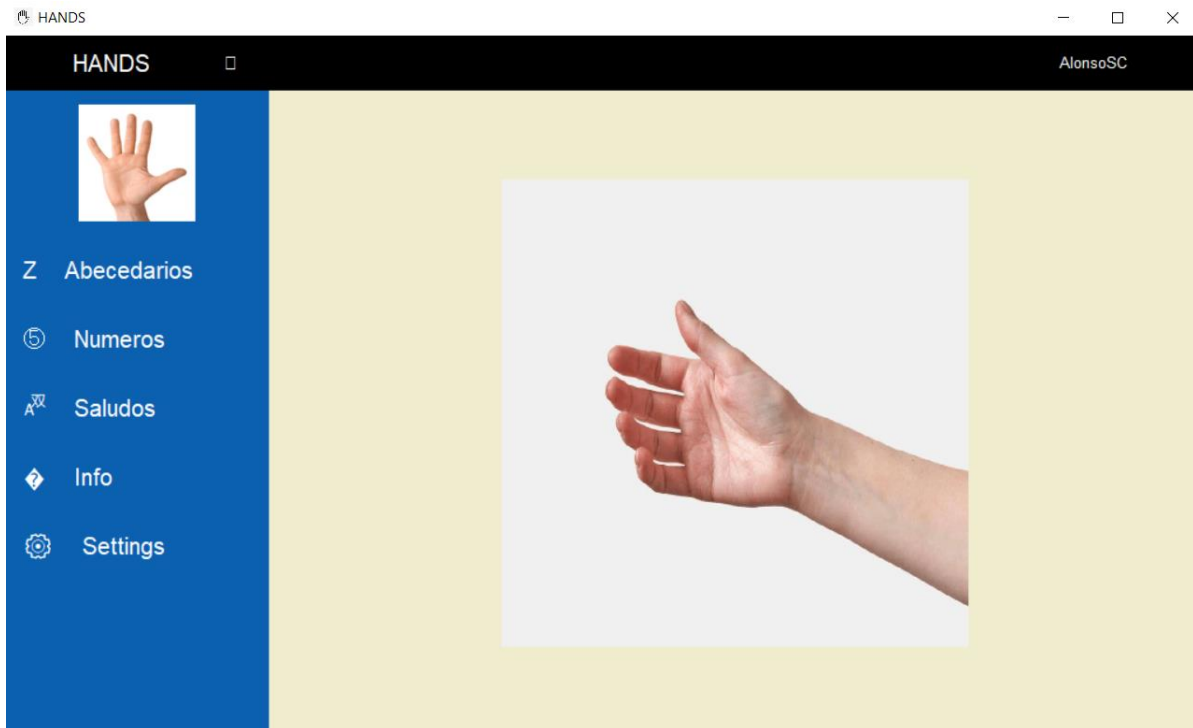


Ilustración 1 Primera versión de la aplicación

Para la elección de entorno de desarrollo del proyecto final, la aplicación web, se tiene en cuenta que se van a utilizar diferentes tecnologías. Desde CSS y JavaScript para el desarrollo de acciones y estilos, hasta HTML para estructurar la información. Sabiendo que se necesita un IDE que soporte estos lenguajes y frameworks se escoge el uso de Visual Studio Code. Este entorno de desarrollo integrado permite el uso de todas estas tecnologías nombradas. Pudiendo ampliar funcionalidades mediante plugins o extensiones que facilitan el trabajo. Algunas de estas extensiones pueden ser aplicadas para la ayuda en la detección de errores sobre lenguajes y sintaxis. En este caso se ha aplicado para la codificación de la lógica en Python.

A mayores de la codificación de la aplicación con Visual Studio Code se utiliza como entorno de pruebas a modo local el navegador de búsqueda Google Chrome.

El proyecto se lanza desde el equipo del alumno en modo local utilizando el puerto 5000

`http://127.0.0.1:5000` o `http://localhost:5000`

4 Descripción general del producto

El proyecto "Hands", busca promover el aprendizaje de la lengua de signos a través de una aplicación web interactiva. Utilizando un enfoque innovador, "Hands" ofrece una plataforma didáctica y accesible que combina tecnologías modernas con los conocimientos adquiridos durante el programa educativo. La aplicación, diseñada con un servidor Flask de Python y herramientas estándar web como HTML, CSS y JavaScript, presenta una serie de pantallas que muestran

contenido relevante para el aprendizaje de la lengua de signos, como el abecedario, saludos y números, en forma de galería de imágenes.

Una característica distintiva de “Hands” es su capacidad para detectar y reconocer gestos de las manos del usuario mientras practica la lengua de signos. Esta funcionalidad se logra gracias a la integración de la librería Mediapipe de Google, que ofrece herramientas para la detección de manos y gestos. Además, se aplica lógica en Python para procesar estos gestos, como la identificación de letras en la lengua de signos española (LSE).

“Hands” representa un proyecto innovador y ambicioso que fusiona la enseñanza de la lengua de signos con la aplicación de tecnologías modernas, con el objetivo de fomentar la inclusión y facilitar el aprendizaje de una habilidad comunicativa esencial. Además de suponer un reto técnico y capacitivo al alumno que explota sus capacidades.

4.1 Visión general del sistema

Este sistema se basa en el desarrollo de una aplicación web que utiliza la biblioteca Flask de Python para el servidor y HTML para la interfaz de usuario. La funcionalidad principal de la aplicación es la detección y reconocimiento de gestos de las manos del usuario utilizando la librería Mediapipe de Google.

Dentro del proyecto existe la limitación y, a su vez oportunidad de desarrollar y mejorar a futuro, la posibilidad de hacer la detección de gestos de dos formas definidas. Ambas basadas en MediaPipe de Google.

En este caso, la primera opción se aplica en el proyecto. El sistema utiliza la cámara web del dispositivo del usuario para capturar imágenes en tiempo real, las cuales son procesadas por la biblioteca Mediapipe para detectar las manos y sus movimientos. Posteriormente, se aplican ciertas condiciones y cálculos para identificar gestos específicos, como la formación de letras en el lenguaje de signos.

Para lograr esto, se implementa un bucle que continuamente captura fotogramas de la cámara, procesa las imágenes para la detección de manos y gestos, y muestra los resultados en una ventana de visualización en tiempo real. Además, se utiliza threading para ejecutar la detección de manos en un hilo separado, lo que permite que la aplicación siga siendo receptiva y funcional mientras se lleva a cabo la detección.

En el caso de la segunda opción es la detección de gestos directamente sobre el navegador. Esto quiere decir que el usuario desde la interfaz HTML puede practicar la lengua de signos y visualizarse en tiempo real mientras obtiene información de si está haciendo los gestos correctamente. Hasta el momento esta funcionalidad está implementada con JavaScript basándose en los preentrenamientos ofrecidos por la librería de MediaPipe por lo que tiene la limitación de detección de seis gestos significativos. Esto no cumple las exigencias del proyecto, pero abre el camino una gran oportunidad. La posibilidad a futuro de mejora de la aplicación basándose en el entrenamiento de visión artificial para la detección de el abecedario completo pudiendo detectar

varios idiomas o inclusive pudiendo detectar formas de comunicación más complejas que requieren una movilidad de tronco, cabeza y extremidades. Un ejemplo puede ser la detección de saludos, la implementación de juegos interactivos o la traducción en tiempo real enfocada a diferentes posibles entornos.

La aplicación también incluye varias rutas que permiten al usuario navegar por diferentes páginas web, como la página de inicio, una página con el abecedario en lengua de signos, una página de saludos y una página de números. Estas páginas están diseñadas con HTML y se renderizan utilizando templates de Flask.

Las pantallas del abecedario, números y saludos en la aplicación "Hands" constituyen herramientas fundamentales para el aprendizaje y la práctica de la lengua de signos de manera interactiva y accesible. La pantalla del abecedario presenta una galería de imágenes que muestran cada una de las letras del alfabeto en lengua de signos. Cada imagen está acompañada de su correspondiente letra, lo que permite a los usuarios familiarizarse con la forma de los signos y su relación con las letras convencionales del alfabeto. Además, esta pantalla puede servir como una referencia visual útil para aquellos que deseen aprender y practicar la comunicación en lengua de signos, proporcionando un recurso accesible y fácil de usar.

Por otro lado, la pantalla de números ofrece una representación visual de los números en lengua de signos. Cada número está asociado con su respectivo signo, lo que permite a los usuarios aprender y practicar la expresión numérica en lengua de signos de manera efectiva.

Finalmente, la pantalla de saludos proporciona ejemplos de saludos comunes expresados en lengua de signos, permitiendo a los usuarios aprender y practicar cómo saludar a otros de manera inclusiva y respetuosa. Cada saludo está acompañado de su representación visual en lengua de signos, lo que facilita la comprensión y la práctica de estos gestos comunicativos.

La aplicación "Hands" interactúa con varios sistemas para proporcionar su funcionalidad completa.

- Sistema de Cámara Web: La aplicación utiliza la cámara web del dispositivo del usuario para capturar imágenes en tiempo real. Estas imágenes son procesadas por la biblioteca Mediapipe para la detección y reconocimiento de gestos de las manos.
- Sistema Operativo: La aplicación es compatible con diferentes sistemas operativos, como Windows, macOS y Linux, ya que se ejecuta en un entorno de servidor Flask de Python que es compatible con múltiples plataformas.
- Sistema de Detección de Gestos (Mediapipe): Interactúa con la librería Mediapipe de Google, que proporciona las herramientas necesarias para la detección y seguimiento de manos en tiempo real, así como para el reconocimiento de gestos específicos.
- Sistema de Renderizado de Páginas Web: La aplicación renderiza páginas HTML utilizando Flask, un framework web de Python. Estas páginas son servidas al cliente a través de un navegador web, como Google Chrome, Firefox o Safari.
- Sistema de Procesamiento de Imágenes (OpenCV): Utiliza la biblioteca OpenCV (Open Source Computer Vision Library) para el procesamiento de imágenes, como la conversión de imágenes de la cámara web al formato requerido por Mediapipe.

- Sistema de Interacción Cliente-Servidor: La aplicación utiliza un servidor Flask para recibir solicitudes HTTP desde el cliente y enviar respuestas, lo que permite la comunicación entre la aplicación web y el navegador del usuario.

4.2 Descripción de métodos

Durante el proceso de desarrollo de la aplicación "Hands", se emplean diversos métodos para la detección de manos y el reconocimiento de gestos asociados a la lengua de signos. Estos métodos se implementan mediante la integración de diversas bibliotecas y herramientas, entre las que se destacan OpenCV, MediaPipe y Flask, entre otras plataformas.

La detección de manos se lleva a cabo utilizando la biblioteca MediaPipe, específicamente el módulo Hands, que proporciona funcionalidades avanzadas para detectar y seguir la posición de las manos en tiempo real a través de una cámara. Este proceso permite capturar imágenes del entorno y procesarlas para identificar la presencia y ubicación de las manos en el campo de visión.

El proceso de detección se realiza en un bucle continuo, donde se capturan los fotogramas de la cámara y se procesan utilizando el modelo de detección de manos de MediaPipe. Este modelo utiliza técnicas de aprendizaje automático para identificar puntos clave en las manos, como las puntas de los dedos y las articulaciones, lo que proporciona información crucial para determinar la posición y la orientación de las manos.

Una vez detectadas las manos en cada fotograma, se aplican algoritmos para analizar la posición y configuración de los dedos, lo que permite reconocer gestos específicos asociados a letras del abecedario y otros signos de la lengua de signos. Por ejemplo, se calculan ángulos entre los puntos clave de los dedos para determinar la posición de los mismos y su relación con la formación de letras.

Además, se implementan funciones específicas para cada letra del abecedario, donde se definen condiciones basadas en la posición y configuración de los dedos para identificar cada letra de manera individual. Estas funciones se activan cuando se cumplen ciertas condiciones predefinidas, lo que permite mostrar la letra correspondiente en la interfaz de usuario y realizar acciones específicas asociadas a cada letra.

Para mejorar la experiencia de interacción con la aplicación "Hands", se ha implementado el framework Flask para crear una interfaz web altamente accesible y versátil. Esta interfaz web no solo facilita el acceso a las funciones de la aplicación, sino que también sirve como un medio interactivo para conectar a los usuarios con el rico mundo de la lengua de signos.

La estructura de la interfaz web se ha diseñado para ofrecer secciones claramente definidas, abarcando una variedad de funciones y actividades. Destacan entre estas secciones el abecedario, los saludos y los números, cada una aportando una experiencia nueva para los usuarios.

En la sección del abecedario, los usuarios pueden explorar y familiarizarse con las letras del alfabeto a través de una presentación educativa. Cada letra se representa con una imagen que

ilustra su forma y configuración en la lengua de signos, permitiendo a los usuarios comprender y asimilar esta forma de comunicación.

La sección de saludos ofrece a los usuarios la oportunidad de aprender y practicar una variedad de saludos comunes en la lengua de signos. Desde saludos simples como "hola" y "adiós" hasta expresiones más elaboradas de cortesía y gratitud, esta sección ofrece una plataforma para fomentar la comunicación inclusiva y el intercambio cultural.

Por otro lado, la pestaña de números permite a los usuarios explorar y aprender los números en la lengua de signos, facilitando así la comunicación de información numérica de manera efectiva y precisa. Desde los números básicos hasta los más complejos, esta sección ofrece recursos visuales y educativos para fortalecer la comprensión y el uso de los números.

Además de estas secciones principales, la interfaz web también incluye un acceso directo a la funcionalidad de detección de manos, que constituye el corazón operativo de la aplicación. Este acceso rápido y conveniente permite a los usuarios iniciar y detener la detección de manos con un simple clic, facilitando aún más la interacción y el control de la aplicación.

4.3 Despliegue de la aplicación

La ejecución de la aplicación por el momento se realiza de forma local sobre el equipo que contiene el proyecto. Iniciando el fichero Python con el código de arranque del servidor se activa todo el sistema y pasa a estar en modo funcional. Para acceder a las diferentes pantallas que ofrece la aplicación se debe de buscar la siguiente ruta en el navegador de preferencia.

`http://127.0.0.1:5000` o `http://localhost:5000`

De esta forma se accede a la pantalla de inicio o índice. En ella se puede hacer la navegación a las otras pestañas.

Como mejora a futuro, se podría implementar la capacidad de despliegue y acceso remoto de la aplicación. Esto podría lograrse mediante la configuración de un servidor web en la nube, lo que permitiría a los usuarios acceder a la aplicación desde cualquier lugar con conexión a internet

5 Planificación y presupuesto

Se planifica la gestión y producción del proyecto destinando un porcentaje del tiempo bastante elevado y considerable al propio desarrollo de la aplicación. Dentro de este desarrollo van incluidas las tareas de análisis, diseño, producción, pruebas y demás procesos necesarios.

Por otra parte, otro porcentaje considerable es destinado a la redacción de la memoria y manuales.

El tiempo de producción comenzando por el análisis, recopilación de información, diseño e investigación, es la parte fundamental y de mayor peso en este proyecto.

El desarrollo completo a constado de 62 días de producción y alrededor de unas 250-260 horas útiles de trabajo.

A continuación, se muestra en forma de tabla el tiempo en días destinado al desarrollo del proyecto y la redacción de la memoria.

DIAS	TAREAS
4-17 MARZO	Investigación, pruebas preliminares con MediaPipe y desarrollo de primeras interfaces.
18 MARZO – 7 ABRIL	Desarrollo de lógica para procesado de imagen, Desarrollo de interfaz y aplicación en Tkinter.
8-11 ABRIL	Migración a aplicación Flet para desarrollo de escritorio (tecnología Python con Flutter)
12-15 ABRIL	Desarrollo de aplicación en Flutter
15-30 ABRIL	Desarrollo e investigación de producto final (Aplicación web con Flask de Python)
MEMORIA	MEMORIA
1 ABRIL	Redacción de puntos generales y especificaciones de entorno (descripción de proyecto y herramientas utilizadas)
2-5 ABRIL	Redacción detallada memoria

Tabla 1 Tabla Tiempo desarrollo

6 Documentación técnica

6.1 Especificación de requisitos

Los requisitos físicos (Hardware) para el desarrollo de este proyecto no son costosos. Cualquier equipo con capacidad de procesamiento de dos hilos de trabajo y un entorno de desarrollo integrado como es Visual Studio Code puede desempeñar sin problema esta tarea.

Cómo requisito indispensable el desarrollo es la disposición de una cámara con capacidad de conectividad al ordenador. Si se utiliza un portátil, lo habitual es contar con una webcam integrada, cumpliendo perfectamente con ese requisito.

En cuanto a requisitos para la navegación a nivel usuario, solo se requeriría en caso de despliegue en remoto un equipo con conexión a internet y disposición de una webcam para el uso completo de los servicios que ofrece la página. Si, por el contrario, la aplicación tiene funcionamiento en modo local, solo se requeriría la capacidad de ejecución y un navegador. Es importante contar con puertos de acceso disponibles. En este caso se utiliza en modo local ocupando el puerto 5000.

6.2 *Análisis del sistema*

En este punto se tratan las funciones principales del sistema y como operan con su entorno, así como sus particularidades.

- Funcionamiento de la detección de manos:

La detección de manos se lleva a cabo mediante el empleo de la biblioteca Mediapipe, que ofrece un conjunto de modelos de aprendizaje automático preentrenados diseñados específicamente para identificar y localizar puntos clave en las manos en tiempo real. Este proceso implica una serie de etapas clave:

- Captura de video:

La aplicación accede a la cámara del dispositivo para obtener un flujo de video en tiempo real. Esto proporciona la entrada necesaria para el proceso de detección y análisis de manos.

- Detección de puntos clave:

Utilizando el modelo de Mediapipe, se identifican y localizan los puntos clave en la mano del usuario. Estos puntos incluyen las puntas de los dedos, las articulaciones de las falanges y otros puntos anatómicos relevantes. La precisión y velocidad de este proceso son fundamentales para una detección efectiva y una interacción fluida.

- Análisis de gestos:

Basándose en la posición, orientación y relación espacial de los puntos clave detectados, se realiza un análisis de los gestos de la mano del usuario. Este análisis puede incluir la identificación de gestos, como abrir y cerrar la mano, hacer gestos con los dedos, y movimientos de desplazamiento y rotación.

- Procesamiento en tiempo real:

Todo el proceso de detección y análisis se lleva a cabo en tiempo real mientras la aplicación está en uso. Esto garantiza una interacción fluida y una respuesta instantánea a los movimientos de la mano del usuario, lo que contribuye a una experiencia de usuario satisfactoria y envolvente.

Este proceso de detección y análisis de manos se ejecuta de manera continua mientras la aplicación está en uso, lo que permite una interacción natural y en tiempo real con el usuario. La combinación de la precisión del modelo de Mediapipe y la implementación cuidadosa de la lógica de interacción garantiza una experiencia de usuario intuitiva y satisfactoria. Otras funcionalidades en HTML y su diseño:

- Interfaz de usuario intuitiva:

La interfaz de usuario (UI) está diseñada con un enfoque centrado en el usuario, priorizando la usabilidad y la accesibilidad. Se utilizan elementos HTML como botones, menús e imágenes para proporcionar una experiencia de usuario intuitiva y familiar. El diseño se basa en principios de diseño responsivo para garantizar que la interfaz sea compatible con una amplia gama de dispositivos y tamaños de pantalla.

- Navegación y estructura del sitio:

La navegación se facilita mediante la inclusión de enlaces y botones que permiten a los usuarios moverse entre las diferentes secciones y páginas del sitio web de manera rápida y sencilla. Se utiliza una estructura de navegación clara y jerárquica, con menús desplegables o barras de navegación fijas para facilitar la exploración del contenido.

- Contenido dinámico y multimedia:

Además de texto estático, el sitio web puede incorporar contenido dinámico y multimedia para enriquecer la experiencia del usuario. Esto puede incluir imágenes, videos, animaciones y elementos interactivos como mapas incrustados o gráficos interactivos. Se utilizan etiquetas HTML como , <video>, <audio> y <canvas> para integrar estos elementos de manera efectiva en el diseño de la página.

- Accesibilidad y compatibilidad con estándares:

Se presta atención a la accesibilidad del sitio web, asegurándose de que sea compatible con dispositivos de asistencia para usuarios con discapacidades. Se siguen las mejores prácticas de diseño web y se adhieren a los estándares HTML y CSS para garantizar la compatibilidad con una amplia variedad de navegadores y dispositivos.

6.3 Diseño del sistema

El diseño del sistema se estructura en base a los siguientes componentes principales, que trabajan en conjunto para lograr la funcionalidad deseada:

- Módulo de Detección de Manos:

Este módulo es el núcleo de la aplicación, encargado de detectar y analizar los gestos de las manos del usuario en tiempo real.

Utiliza la biblioteca Mediapipe para identificar y localizar puntos clave en las manos.

Procesa el flujo de video capturado por la cámara del dispositivo.

Analiza la posición y orientación de los puntos clave para identificar gestos específicos, como abrir y cerrar la mano, movimientos de los dedos, y gestos de desplazamiento y rotación.

Proporciona una interfaz para la interacción con otros módulos de la aplicación.

- **Módulo de Interfaz de Usuario (UI):**

Este módulo gestiona la interfaz de usuario de la aplicación, mostrando la retroalimentación visual correspondiente a los gestos detectados. Utiliza HTML, CSS y JavaScript para crear una interfaz intuitiva y receptiva.

Recibe información del Módulo de Detección de Manos para actualizar la interfaz en tiempo real, reflejando los gestos del usuario. Incluye elementos como botones, menús desplegables y campos de entrada para permitir la interacción del usuario con la aplicación.

- **Módulo de Control y Gestión de Eventos:**

Este módulo se encarga de interpretar los gestos detectados y traducirlos en acciones específicas dentro de la aplicación. Recibe información del Módulo de Detección de Manos sobre los gestos identificados. Asocia cada gesto con una acción correspondiente, como seleccionar opciones en menús, navegar entre pantallas o activar comandos específicos. Coordina la comunicación entre el Módulo de Detección de Manos y la interfaz de usuario para garantizar una interacción fluida y coherente.

- **Módulo de Comunicación con la Cámara:**

Este módulo gestiona el acceso a la cámara del dispositivo para capturar el flujo de video en tiempo real. Utiliza tecnologías web como WebRTC para acceder a la cámara desde el navegador.

Proporciona una interfaz para iniciar y detener la captura de video, así como para ajustar la resolución y configuración de la cámara según sea necesario.

- **Módulo de Compatibilidad y Optimización:**

Este módulo se encarga de garantizar la compatibilidad y rendimiento óptimo de la aplicación en diferentes dispositivos y navegadores. Implementa técnicas de diseño responsivo para adaptar la interfaz de usuario a diferentes tamaños de pantalla y orientaciones.

Realiza pruebas de compatibilidad cruzada para asegurar que la aplicación funcione de manera consistente en una variedad de dispositivos y plataformas.

Este diseño del sistema proporciona una arquitectura modular y escalable que permite una fácil expansión y mantenimiento de la aplicación. Cada módulo cumple un propósito específico y trabaja en conjunto para ofrecer una experiencia de usuario satisfactoria y envolvente.

6.3.1 Almacenamiento y tratamiento de datos

En "Hands" por el momento no existe almacenamiento de datos. No se requiere de ninguna estructura de datos compleja, ni de ningún sistema de almacenamiento en base de datos. Esto es debido a que la aplicación carece de la necesidad de identificación y registro de usuarios. Tampoco cuenta con la posibilidad de recopilación de datos como pueden ser formularios de contacto o similares.

Esto ofrece una gran ventaja librando de la tarea técnica que corresponde al desarrollo e implantación de esas tecnologías de almacenamiento y tratamiento de datos.

En el proyecto "Hands", las imágenes se obtienen principalmente de un repositorio público donde están almacenadas. Estas imágenes son accesibles a través de URLs públicas, lo que permite a la aplicación cargarlas directamente desde el repositorio sin necesidad de almacenarlas localmente o en una base de datos.

El proceso de obtener las imágenes se realiza mediante solicitudes HTTP a las direcciones URL de las imágenes específicas en el repositorio. Una vez que la aplicación obtiene el contenido de las imágenes a través de estas solicitudes, puede procesarlas y mostrarlas en las páginas web correspondientes. Por ejemplo, en la pantalla del abecedario, la aplicación itera sobre cada letra del alfabeto y genera la URL de la imagen correspondiente a esa letra en lengua de signos. Luego, utiliza esta URL para cargar la imagen en la página HTML y mostrarla al usuario.

Este enfoque de cargar imágenes directamente desde un repositorio público es eficiente y práctico, ya que elimina la necesidad de gestionar un almacenamiento local de imágenes y simplifica el proceso de obtención de recursos visuales para la aplicación. Además, garantiza que las imágenes estén siempre actualizadas y accesibles para todos los usuarios de la aplicación.

6.3.2 Diseño de la interfaz de usuario

El diseño para la interfaz de usuario está basado en un cuadro interactivo de tipo Dashboard, con menú de navegaciones lateral situado a la izquierda si se toma de referencia el eje central de la pantalla. Esta distribución, se mantiene constante para todas las pantallas siguiendo el mismo estilo y la misma paleta de colores.

El menú lateral está compuesto por el título de la página en la parte superior seguido de los 4 botones principales para la navegación entre las diferentes pantallas que forman la web. Dichos botones contienen la propiedad de cambio de color al pasar el cursor por encima, haciendo así una experiencia más agradable e interactiva. La distribución de estos botones y el contenido de ellos varía dependiendo de la página en la que estemos. Desapareciendo la opción de navegar a la propia pantalla en la que se encuentre el usuario y sustituyendo el hueco de ese botón por el correspondiente, normalmente el botón de inicio que lleva a la pestaña principal o índice.

- **Botón 1:** Detección → se realiza la navegación hasta la pantalla de detección. Se inicia la ejecución del código de detección de gestos con las manos.
- **Botón 2:** Abecedario → se realiza la navegación hasta la pantalla de abecedario.
- **Botón 3:** Saludos → se realiza la navegación hasta la pantalla de saludos.
- **Botón 4:** Números → se realiza la navegación hasta la pantalla de números.

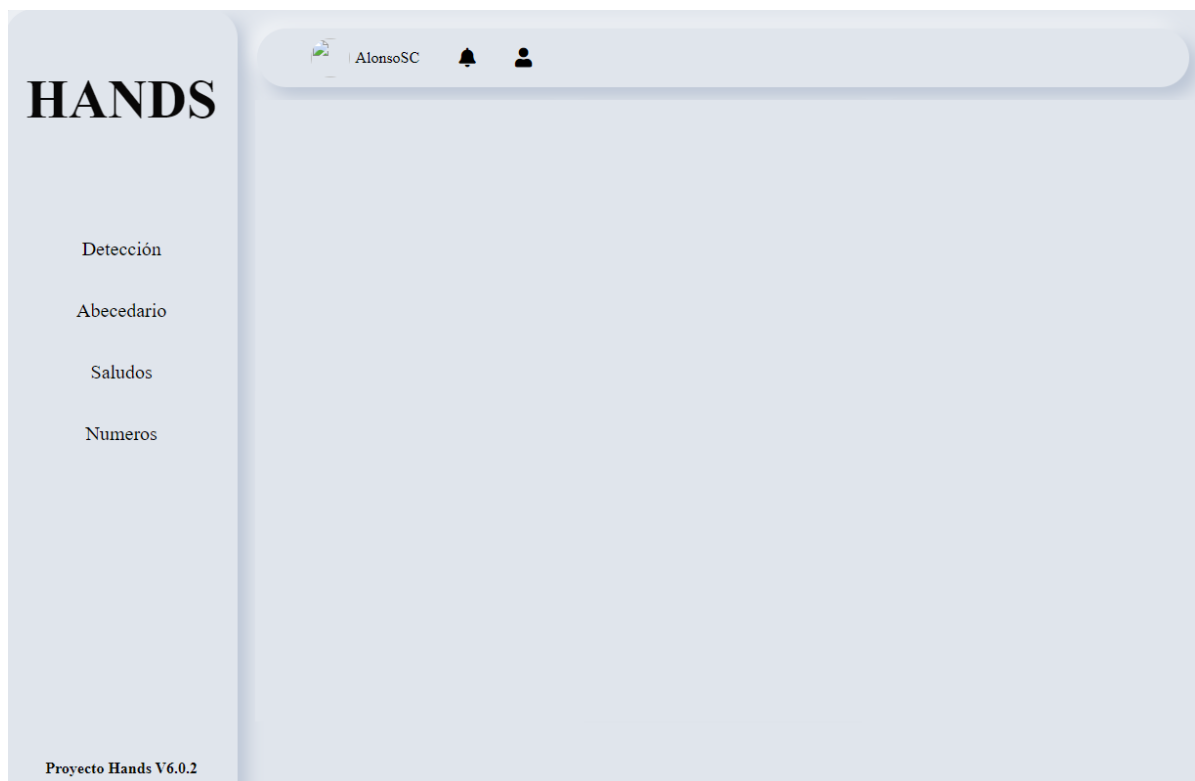


Ilustración 2 Diseño y disposición botones web

El diseño basado en una temática suave y carga visual ligera facilita al usuario la estancia y navegación entre páginas. El esquema gráfico está pensado para el uso continuado. Esto es debido a que el usuario requiere de un tiempo prolongado para visualizar las imágenes, asentar los conocimientos y practicar lo aprendido. En un primer momento, se plantea homogeneizar y estandarizar los colores de la web en base a las imágenes, esta idea queda descartada debido a la carga visual que supone.

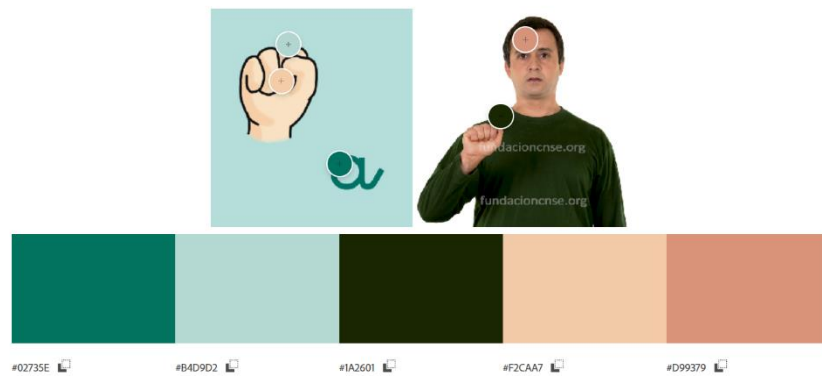


Ilustración 3 Diseño inicial en base a colores de imágenes



Ilustración 4 Estructura, estilo y colores Hands

En cuanto al contenido de la web, está basado en las imágenes y GIFs que contienen la información relevante para el usuario. En este caso, son las imágenes de gestos para la lengua de signos en español. Estas imágenes se muestran en forma de galerías en el cuerpo principal de cada pestaña. Todas ellas están contenidas en cuadros blancos simulando un marco comprendido como cartas. Con un diseño ligero y sin bordes haciendo destacar así el cuerpo central, las imágenes.

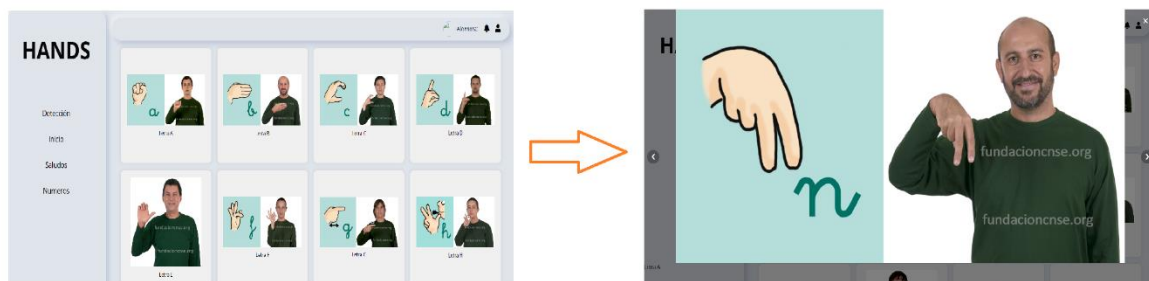


Ilustración 5 Pantalla Abecedario

6.3.3 Diseño de la aplicación

La aplicación consta de una organización de pantallas estructuradas en HTML con relación e interacción entre ellas. Desde cualquiera de las pantallas se puede llegar por ejemplo a la pantalla de inicio o de detección.

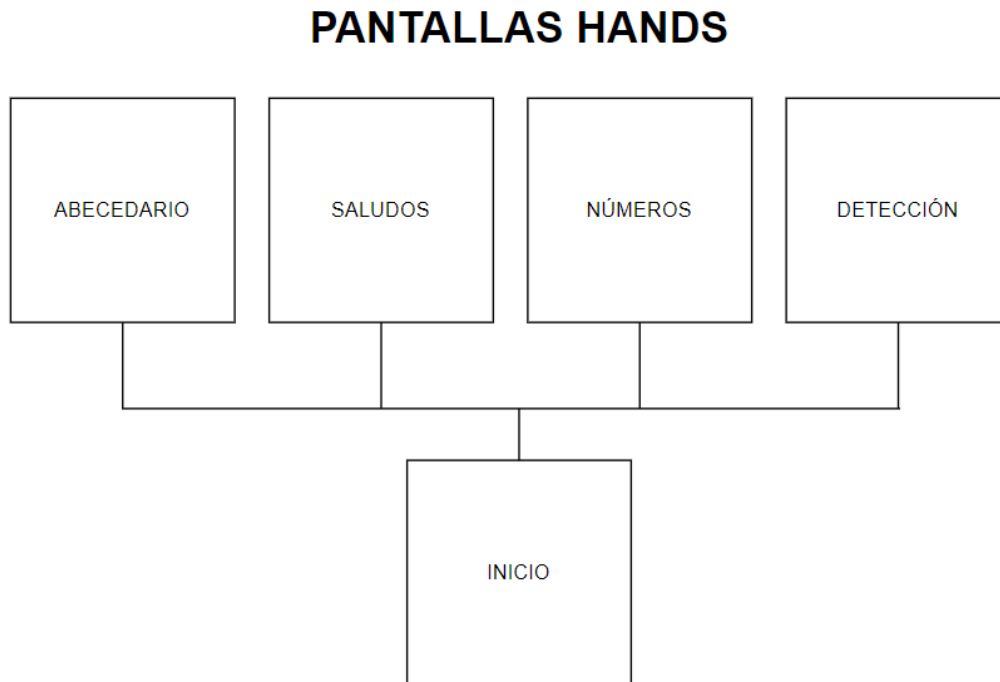


Ilustración 6 Esquema pantallas Hands

En el diagrama se pueden ver las 4 pantallas principales que componen la aplicación, relacionadas entre si. Y la principal, inicio, destacada por ser la primera a la que accede el usuario.

A mayores, se realiza la detección de los gestos de las manos en la aplicación. Esto opera de la

siguiente forma según el diagrama.

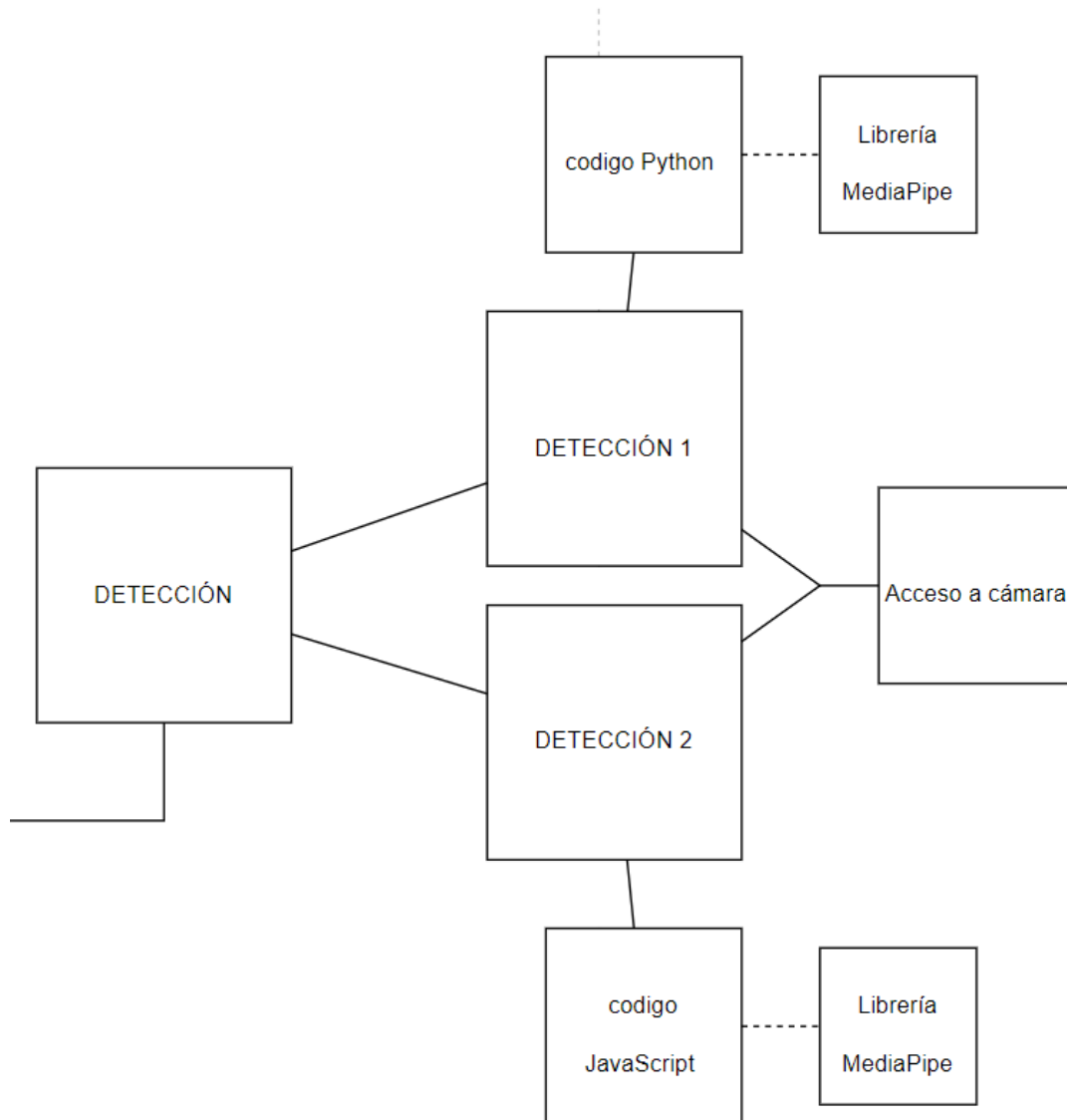


Ilustración 7 Esquema Estructura detección de signos

Visión general de la aplicación con esquema gráfico de los componentes que forman la aplicación web “Hands”.

Elementos esquema:

- Pantallas de detección, inicio, números, abecedario y saludos.
- Código de detección con MediaPipe en JavaScript y en Python.
- Uso de librería MediaPipe

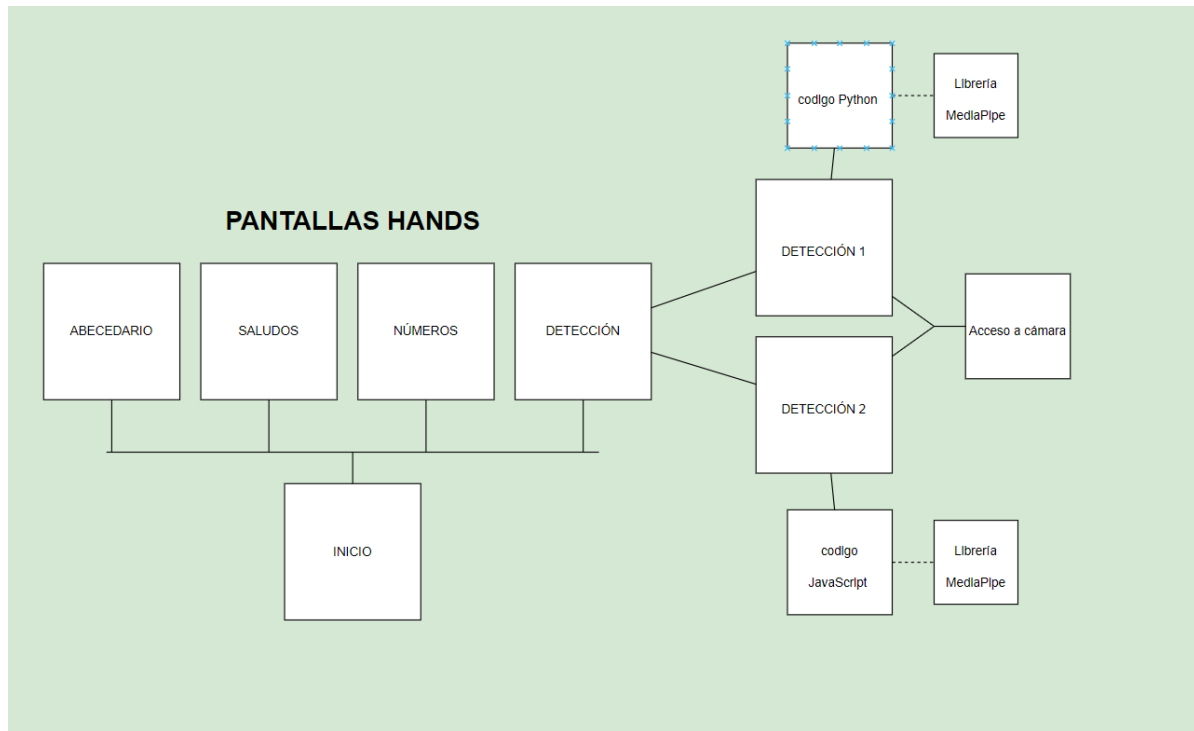


Ilustración 8 Esquema general de la aplicación Hands

6.4 Implementación

En este caso la aplicación no está subida en ningún servidor ni tampoco asociada a un dominio. Por lo tanto, su uso es completamente en forma local.

La implantación del sistema en un entorno local garantiza un funcionamiento eficiente y seguro, directamente en el equipo del usuario. Este enfoque ofrece una serie de ventajas significativas en términos de rendimiento, privacidad y accesibilidad. A continuación, se detallan los aspectos clave de la implantación en entorno local:

- **Instalación y Configuración:**

La implantación en entorno local comienza con la instalación del sistema en el equipo del usuario. Esto implica descargar e instalar los archivos necesarios, que incluyen el código fuente de la aplicación, bibliotecas externas como MediaPipe, y cualquier otro recurso requerido. Una vez instalado, el sistema puede ser configurado según las preferencias del usuario, ajustando parámetros como la resolución de la cámara, la sensibilidad de la detección de gestos, y otras opciones personalizables.

- **Acceso a Recursos Locales:**

Al ejecutarse en el equipo del usuario, el sistema tiene acceso directo a los recursos locales, incluyendo la cámara integrada en el dispositivo. Esto permite una captura de video en tiempo real sin necesidad de depender de conexiones externas o servicios en la nube. Además, el sistema

puede aprovechar otros recursos locales, como el procesador y la memoria, para un rendimiento óptimo.

- Privacidad y Seguridad:

La implantación en entorno local ofrece un mayor control sobre la privacidad y seguridad de los datos del usuario. Como la aplicación no depende de servicios externos, los datos de video y gestos permanecen en el dispositivo del usuario en todo momento. Esto reduce el riesgo de exposición de datos sensibles y protege la privacidad del usuario frente a posibles amenazas externas.

- Rendimiento Optimizado:

Al ejecutarse directamente en el equipo del usuario, el sistema puede aprovechar al máximo los recursos disponibles, lo que se traduce en un rendimiento optimizado y una respuesta rápida a las acciones del usuario. La detección de gestos se realiza en tiempo real, sin retrasos perceptibles, lo que contribuye a una experiencia de usuario fluida y envolvente.

6.4.1 Entorno de desarrollo

El proyecto implica el desarrollo de una aplicación web interactiva que utilice tecnologías como HTML, CSS, JavaScript y Python. Esta aplicación tiene como objetivo principal la detección y análisis de gestos de manos en tiempo real, utilizando la biblioteca Mediapipe para identificar y localizar puntos clave en las manos de los usuarios.

- Consideraciones Técnicas:

Antes de tomar una decisión sobre el entorno de desarrollo, se llevaron a cabo varias consideraciones técnicas para asegurar que el entorno elegido fuera adecuado para las necesidades específicas del proyecto. Estas consideraciones incluyeron:

Compatibilidad Tecnológica: Se requiere un entorno de desarrollo que admita todas las tecnologías necesarias para el proyecto, incluyendo HTML, CSS, JavaScript y Python. Esto garantiza que los desarrolladores puedan trabajar de manera efectiva con todas las partes del código base.

Herramientas de Detección de Errores: Dada la complejidad del proyecto y la integración de múltiples tecnologías, es crucial contar con herramientas que faciliten la detección de errores de sintaxis y lógica. Esto ayuda a mantener la calidad del código y a reducir el tiempo dedicado a la depuración.

Entorno de Pruebas: Se necesita un entorno que permita realizar pruebas locales en un ambiente controlado antes de implementar cambios en un servidor en vivo. Esto es fundamental para garantizar la estabilidad y fiabilidad de la aplicación durante el desarrollo y después de su implementación.

La decisión de utilizar Visual Studio Code como entorno de desarrollo para la implementación de la aplicación web se fundamentó en una evaluación exhaustiva de las necesidades técnicas y funcionales del proyecto. La versatilidad y compatibilidad de Visual Studio Code con una amplia

gama de tecnologías, incluyendo CSS, JavaScript, HTML y Python, ofrecieron la base necesaria para el desarrollo y diseño del producto.

- **Ventajas de Visual Studio Code:**

Flexibilidad y Ampliación: La capacidad de Visual Studio Code para admitir plugins y extensiones permite ampliar su funcionalidad según las necesidades del proyecto. Esto resulta especialmente útil para optimizar el flujo de trabajo y mejorar la productividad durante el desarrollo de la aplicación.

Detección de Errores: Las extensiones disponibles en Visual Studio Code facilitan la detección de errores de sintaxis y ofrecen sugerencias para mejorar la calidad del código. Esto contribuye a un desarrollo más eficiente y a la creación de un producto final más robusto y fiable.

Soporte para pruebas Locales: Visual Studio Code permite realizar pruebas locales en un entorno controlado antes de implementar cambios en un servidor en vivo. Esto ayuda a identificar y solucionar problemas antes de que afecten a la experiencia del usuario final.

- **Entorno de Desarrollo Integrado:**

El entorno de desarrollo integrado se completa con el uso del navegador Google Chrome como herramienta de prueba. Esta combinación permite realizar pruebas exhaustivas del código HTML, CSS y JavaScript en un entorno realista y familiar para los desarrolladores.

- **Requisitos Físicos:**

Los requisitos físicos (hardware) para el desarrollo de este proyecto no son costosos, ni exigentes. Cualquier equipo con capacidad de procesamiento de hilos separados y un entorno de desarrollo integrado como Visual Studio Code puede desempeñar esta tarea sin problemas, lo que garantiza una accesibilidad amplia.

6.4.2 Estructura del código

EL código de la aplicación se basa en una estructura de directorios con diferentes nombres identificativos. “Funciones”, “static” y “templates” son las principales. Dentro de ellos, se encuentran ficheros JavaScript, Python, HTML y Css. Así como imágenes con diferentes extensiones en función de su cometido bien sean iconos (.ico) o fondos o imágenes auxiliares (.png o .jpg)

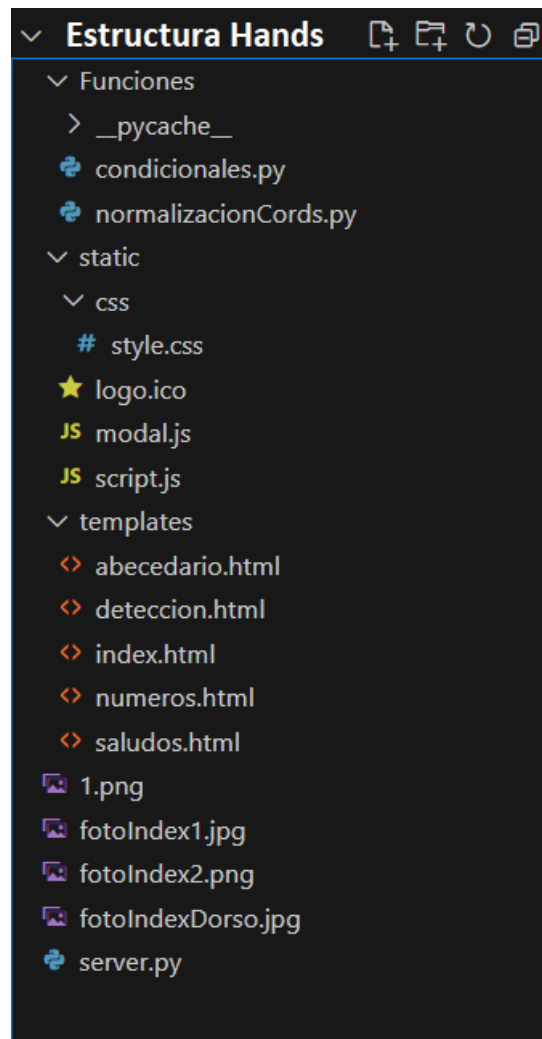


Ilustración 9 Estructura de carpetas

Static: Esta carpeta almacena archivos estáticos como hojas de estilo CSS y código JavaScript. En este caso el archivo style.css, logo.ico, modal.js y script.js. Cada uno de estos ficheros cumple una función específica. modal.js por ejemplo es un fichero que contiene el código necesario para toda la parte de la operación con las tarjetas que contienen las imágenes, desde visualizarlas hasta su función de activación con los Gifs al paso del cursor por encima. Otra funcionalidad a destacar sería la de script.js que contiene el código JavaScript destinado a lanzar el hilo en segundo plano que activa el código para la detección de los signos mediante el código Python.

```
static > css > # style.css > .prev
1  * {
2      padding: 0px;
3      margin: 0px;
4      -webkit-box-sizing: border-box;
5      -moz-box-sizing: border-box;
6      box-sizing: border-box;
7  }
8
9  :root {
10     --main-bg-color: #e0e5ec;
11     --main-font-to-color: #6a7786;
12     --main-color-alt: #1787e0;
13     --primary: #6d5dfc;
14     --min-font-color: rgba(0, 0, 0, 0.582);
15     --main-transition: 0.3s;
16     --font-title: 'Roboto Slab', serif;
17 }
```

Ilustración 10 Declaración Colores en Css

Css: En el directorio se encuentran los archivos CSS que definen el estilo visual de la web. Los estilos vienen definidos por una estructura suave y ligera.

En cuanto a los colores aplicados en la web son los siguiente:

- #e0e5ec: Se aplica al fondo principal (background-color) del cuerpo (body), menú (menu), y contenedor de tarjetas (card-container).
- #6a7786: Se utiliza como color principal para el texto (color) en el menú (menu).
- #1787e0: Se utiliza como color de fondo (background-color) al pasar el ratón (hover) sobre los enlaces (a) del menú (menu).
- #6d5dfc: Se utiliza como color primario para el fondo (background-color) del menú (menu).
- rgba(0, 0, 0, 0.582): Se usa como color de fuente (color) para un tono de gris más oscuro.
- rgb(8, 7, 7): Se aplica como color de texto (color) en el menú (menu).
- rgba(163, 177, 198, 0.6): Se aplica como sombra (box-shadow) al menú (menu).
- rgba(255, 255, 255, 0.5): Se utiliza como sombra (box-shadow) en el menú (menu).
- #f1f1f1: Se utiliza como color de fondo (background-color) para el contenido del menú desplegable (menu-content) y sombra (box-shadow) en el menú desplegable.

- #5a6268: Se usa como color de fondo (background) para el botón de búsqueda (search-button).
- #f0f0f0: Se utiliza como color de fondo (background-color) para las tarjetas (card).
- rgba(0, 0, 0, 0.1): Se aplica como sombra (box-shadow) en las tarjetas (card).
- rgba(0, 0, 0, 0.3): Se utiliza como color de fondo (background-color) para los botones de navegación (prev y next) y sus estados de hover.
- rgba(0, 0, 0, 0.5): Se utiliza como color de fondo (background-color) para los botones de navegación (prev y next) y sus estados de hover.

Templates: En esta carpeta, se encuentran los archivos HTML que constituyen las diferentes vistas de la aplicación. Estos archivos HTML pueden contener código HTML estático y también pueden integrar código dinámico utilizando plantillas de servidor.

Cada pantalla cuenta con su menú lateral con la siguiente disposición:

```
<body>
  <div class="minet">
    <div class="menu">
      <h3 class="logo">HANDS</h3>
      <div class="linkes">
        <a href="/deteccion" id="start_detection_button" class="button">Detección</a>
        <a href="/abecedario" class="button">Abecedario</a>
        <a href="/saludos" class="button">Saludos</a>
        <a href="/numeros" class="button">Numeros</a><br>
        <br>
        <p>Proyecto Hands V6.0.2 </p>
      </div>
    </div>
  </div>
```

Ilustración 11 Estructura menú en HTML

Dependiendo de en qué pestaña se encuentre el usuario, el menú varía cambiando las opciones. Si el usuario está en la pestaña de abecedario, no aparecerá el botón de abecedario y este será reemplazado por el de inicio. En este caso la imagen muestra el menú del índice, se puede identificar debido a que no tiene el botón de inicio.

Los HTML pueden contar con secciones que contienen referencias a scripts de JavaScript para realizar ciertas acciones, en este caso son para abrir el modo carrusel y para lanzar la ejecución de la detección de manos en segundo plano.

```

72     </div>
73     <div id="myModal" class="modal">
74         <span class="close">&times;</span>
75         <img class="modal-content" id="img01">
76         <div id="caption"></div>
77         <button id="prevBtn" class="prev">&#10094;</button>
78         <button id="nextBtn" class="next">&#10095;</button>
79     </div>
80
81     <script src="{{ url_for('static', filename='modal.js') }}"></script>
82     <script src="{{ url_for('static', filename='script.js') }}"></script>
83 </body>

```

Ilustración 12 Espacio reservado para JavaScript

Los ficheros modal.js y script.js son código JavaScript destinado a acciones concretas de las pantallas HTML. script.js está compuesto por una función específica que espera el evento clic sobre el botón de detección de cualquiera de las pantallas. Esta función hace un “POST” para iniciar la detección en segundo plano de la detección de los gestos del abecedario.

```

static > JS script.js > ...
1  document.addEventListener("DOMContentLoaded", function () {
2      document.getElementById("start_detection_button").addEventListener("click", function () {
3          fetch("/start_detection", {
4              method: "POST"
5          }).then(response => response.json())
6              .then(data => {
7                  console.log(data.message);
8              })
9              .catch(error => console.error("Error:", error));
10     });
11
12 });

```

Ilustración 13 Código JavaScript acción detección

Funciones: Contiene código Python necesario para el procesamiento de la lógica aplicada a la detección de las letras. Contiene dos ficheros esenciales. normalizacionCord.py y condicionales.py

```

Funciones > condicionales.py > condicionalesLetras
1  import cv2
2
3  def condicionalesLetras(dedos, frame):
4      font = cv2.FONT_HERSHEY_SIMPLEX
5      if dedos == [1, 1, 0, 0, 0, 0]:
6          cv2.rectangle(frame, (0, 0), (100, 100), (255, 255, 255), -1)
7          cv2.putText(frame, 'A', (20, 80), font, 3, (0, 0, 0), 2, cv2.LINE_AA)
8          print("A")
9
10     if dedos == [0, 0, 0, 0, 0, 0]:
11         cv2.rectangle(frame, (0, 0), (100, 100), (255, 255, 255), -1)
12         cv2.putText(frame, 'E', (20, 80), font, 3, (0, 0, 0), 2, cv2.LINE_AA)
13         print("E")
14

```

Ilustración 14 Código Condicionales

En condicionales se desarrolla una función principal que recibe los dedos y el frame (la imagen) y en base a eso se comprueba mediante sentencias de control la posición de cada coordenada de los puntos clave (landmarks). Si los parámetros de entrada corresponden a la comprobación, se dibuja en un cuadro la letra correspondiente.

```
Funciones > normalizacionCords.py > ...
1  import mediapipe as mp
2  from math import degrees, acos
3  import numpy as np
4
5  #actualizacion
6  mp_hands = mp.solutions.hands
7
8  def obtenerAngulos(results, width, height):
9      for hand_landmarks in results.multi_hand_landmarks:
10
11          # COORDENADAS MEÑIQUE
12          x1, y1 = [int(hand_landmarks.landmark[mp_hands.HandLandmark.PINKY_TIP].x * width),
13                  int(hand_landmarks.landmark[mp_hands.HandLandmark.PINKY_TIP].y * height)]
14
15          x2, y2 = [int(hand_landmarks.landmark[mp_hands.HandLandmark.PINKY_PIP].x * width),
16                  int(hand_landmarks.landmark[mp_hands.HandLandmark.PINKY_PIP].y * height)]
17
18          x3, y3 = [int(hand_landmarks.landmark[mp_hands.HandLandmark.PINKY_MCP].x * width),
19                  int(hand_landmarks.landmark[mp_hands.HandLandmark.PINKY_MCP].y * height)]
20
```

Ilustración 15 Código Obtener Ángulo

En este bloque de código, se define una función llamada obtenerAngulos. Esta función recibe tres parámetros: results, width y height. Los parámetros width y height representan el ancho y la altura de la imagen en la que se están detectando las manos, mientras que results contiene los resultados de la detección de manos realizada por la biblioteca MediaPipe.

La función obtenerAngulos itera a través de los resultados proporcionados por MediaPipe para cada mano detectada en la imagen. Para cada mano, se extraen las coordenadas de los puntos clave (landmarks) relevantes, como las puntas de los dedos, las articulaciones y la muñeca. Estas coordenadas se calculan multiplicando las coordenadas normalizadas proporcionadas por MediaPipe por el ancho y la altura de la imagen, para obtener las coordenadas en píxeles.

Luego, se calculan las longitudes de los segmentos de los dedos y se utilizan para calcular los ángulos entre los segmentos de los dedos. Se emplea la fórmula del coseno para calcular los ángulos a partir de las longitudes de los lados de un triángulo.

$$\text{Cos} = \frac{\text{Cateto Contiguo}}{\text{Hipotenusa}}$$

Sin embargo, se toman precauciones para evitar casos de indeterminación y para asegurar que los valores de coseno estén en el rango [-1, 1]. Finalmente, se almacenan los ángulos calculados en una lista llamada angulosid.

```
server.py > ...
1  from flask import Flask, render_template, request, jsonify
2  import cv2
3  import mediapipe as mp
4  from Funciones.condicionales import condicionalesLetras
5  from Funciones.normalizacionCords import obtenerAngulos
6  import threading
7
8  app = Flask(__name__)
9
```

Ilustración 16 Importaciones Flask y funciones de detección

El código de server.py define una aplicación web utilizando el framework Flask que implementa la detección de manos en tiempo real utilizando la biblioteca MediaPipe. Aquí hay un resumen de lo que hace el código y para qué sirve:

Importaciones de bibliotecas: Se importan las bibliotecas necesarias, incluyendo Flask para la creación de la aplicación web, OpenCV para el procesamiento de imágenes, MediaPipe para la detección de manos, y threading para ejecutar la detección en un hilo separado.

Configuración de la aplicación Flask: Se crea una instancia de la aplicación Flask llamada app.

Variables globales y configuraciones de MediaPipe: Se declara una variable global llamada lectura_actual que se utiliza para almacenar la posición actual del dedo meñique y se inicializa en 0. También se configuran algunas variables relacionadas con MediaPipe, como mp_hands para la detección de manos y mp_drawing_styles para el estilo de dibujo de los resultados.

Función detect_hands: Esta función realiza la detección de manos en tiempo real utilizando la cámara. Utiliza un bucle infinito que captura continuamente fotogramas de la cámara, y los procesa con MediaPipe. Con ello, detecta las manos, y realiza acciones basadas en los resultados, como calcular ángulos entre los dedos y dibujar los resultados en los fotogramas.

Ruta /start_detection: Esta ruta permite iniciar la detección de manos en un hilo separado cuando se realiza una solicitud POST. La detección se inicia llamando a la función detect_hands en un nuevo hilo.

Rutas para las páginas web: Se definen varias rutas para diferentes páginas web de la aplicación, como la página principal ("/"), una página para el abecedario ("/abecedario"), una página para saludos ("/saludos"), una página para números ("/numeros"), una página para la detección de manos ("/deteccion"), y una página de inicio ("/index").

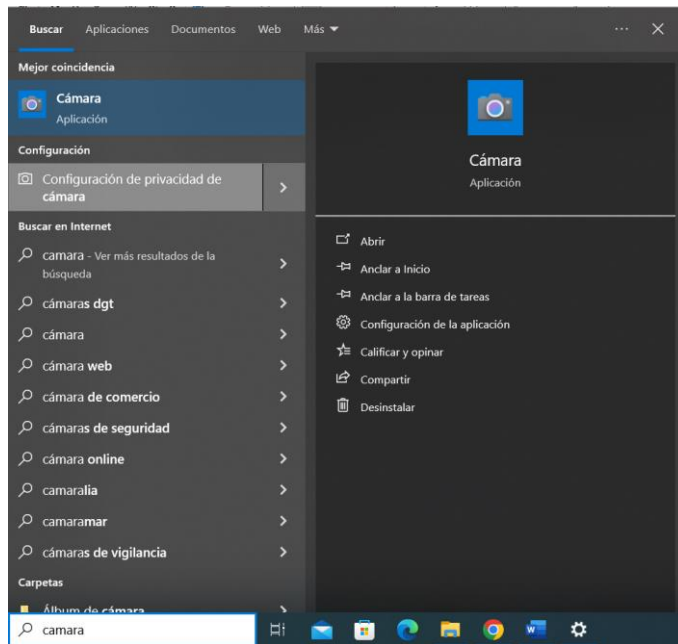
Función principal: Finalmente, se verifica si el script se está ejecutando directamente (__name__ == "__main__") y en ese caso se inicia el servidor web de Flask en modo de depuración.

```
102 @app.route("/")
103 def index():
104     return render_template("index.html")
105
106 @app.route("/abecedario")
107 def abecedario():
108     return render_template("abecedario.html")
109
110 @app.route("/saludos")
111 def saludos():
112     return render_template("saludos.html")
113
114 @app.route("/numeros")
115 def numeros():
116     return render_template("numeros.html")
117
118 @app.route("/deteccion")
119 def deteccion():
120     return render_template("deteccion.html")
121
122 @app.route("/index")
123 def inicio():
124     return render_template("index.html")
125
126 if __name__ == "__main__":
127     app.run(debug=True)
128
```

Ilustración 17 Llamadas a las pantallas HTML

6.4.3 Cuestiones del diseño e implementación reseñables

A la hora de operar con la detección de manos por visión artificial, el usuario debe de conceder los permisos necesarios al navegador para el acceso a la cámara. Esto es una práctica común de seguridad y privacidad. Para acceder a esta configuración los usuarios de Windows 10 pueden llegar a ello de la siguiente manera:



Pasos a seguir:

1. Barra de búsqueda de Windows.
"cámara" en el campo de texto.
2. Acceder a "configuración de privacidad de cámara"
3. Buscar sección "Permitir que las aplicaciones de escritorio accedan a la cámara"
4. Activar botón. Por defecto, desactivado.

Ilustración 18 Inicio Windows10 cámara1



Cámara

Visor web de aplicación de escritorio ☒ Activado

Permitir que las aplicaciones de escritorio accedan a la cámara


Algunas aplicaciones y características de Windows necesitan tener acceso a la cámara para funcionar correctamente. Si desactivas este ajuste aquí, podrías limitar lo que las aplicaciones de escritorio y Windows pueden hacer.

☒ Activado

Cámara activada o desactivada

Es posible que algunas aplicaciones de escritorio no aparezcan en la siguiente lista o que no se vean afectadas por esta configuración.

[Averiguar por qué](#)

 Google Chrome
Último acceso: 04/05/2024 12:01:26

 Python
Último acceso: 27/04/2024 20:15:47

 Python
Último acceso: 04/05/2024 12:15:22


 Zoom Meetings
Último acceso: 01/10/2023 12:29:22

Ilustración 19 Configuración cámara2

En caso de no tener webcam incorporada, se debería de conectar una de forma manual. Normalmente esta operación se suele hacer por medio de un puerto USB.

6.5 Pruebas

En la aplicación, se llevan a cabo una serie de pruebas destinadas a garantizar su funcionalidad y usabilidad para los usuarios. A continuación, se detallan algunas de las pruebas realizadas junto con sus resultados favorables:

- Prueba de Interfaz de Usuario (UI):

Se verifica que la interfaz de usuario sea intuitiva y fácil de navegar. Por ejemplo, se asegura de que los botones y controles estén ubicados de manera lógica y que los menús sean accesibles.

Se prueba la disposición de los elementos en la pantalla para asegurar una experiencia visualmente atractiva. Esto implica comprobar que los colores, fuentes y tamaños de los elementos sean coherentes y estén en línea con la identidad visual de la aplicación.

- Prueba de Navegación:

Se verifica la navegación entre diferentes secciones de la aplicación para asegurar que sea fluida y sin problemas. Por ejemplo, se comprueba que los enlaces y botones de navegación lleven al usuario a las páginas correspondientes de manera rápida y eficiente.

Se prueba la funcionalidad de los botones de navegación para garantizar que lleven al usuario a las secciones correctas. Se realizan pruebas de clics para asegurarse de que los enlaces funcionen correctamente en todas las áreas de la aplicación.

- Prueba de Funcionalidad Básica:

Se verifica que todas las funciones básicas de la aplicación funcionen correctamente, como la captura de imágenes, la detección de gestos y la visualización de resultados. Se llevan a cabo pruebas exhaustivas de cada función para identificar posibles errores o problemas de rendimiento.

Se prueba la capacidad de la aplicación para reconocer gestos simples de manera precisa y oportuna. Por ejemplo, se realizan gestos de deslizamiento y toque en la pantalla para verificar la respuesta de la aplicación en tiempo real.

- Prueba de Respuesta en Tiempo Real:

Se prueba la capacidad de la aplicación para procesar y responder a las acciones del usuario en tiempo real, especialmente durante la captura de imágenes y la detección de gestos. Se realizan pruebas de latencia para garantizar que la aplicación responda de manera rápida y precisa a las interacciones del usuario.

- Prueba de Estabilidad:

Se verifica la estabilidad de la aplicación durante un uso prolongado para asegurar que no haya fallos o cierres inesperados. Se realizan pruebas de resistencia en las que la aplicación se ejecuta durante largos períodos de tiempo bajo condiciones de carga pesada.

Se prueba la aplicación en diferentes dispositivos y condiciones para garantizar su estabilidad en diferentes entornos. Se realizan pruebas de compatibilidad en dispositivos móviles, tabletas y computadoras de escritorio para asegurarse de que la aplicación funcione correctamente en todas las plataformas.

- Prueba de Usabilidad:

Se realizan pruebas con usuarios reales para evaluar la facilidad de uso y comprensión de la aplicación. Se solicita a los usuarios que realicen tareas específicas dentro de la aplicación, y se observa su interacción para identificar áreas de confusión o dificultad.

Se recopilan comentarios y sugerencias de los usuarios para identificar áreas de mejora en la usabilidad.

Estas pruebas permiten validar la funcionalidad básica y la experiencia del usuario de la aplicación. Los resultados obtenidos de estas pruebas proporcionan información valiosa para realizar ajustes y mejoras en el diseño y el funcionamiento de la aplicación.

7 Manuales

7.1 *Manual de usuario*

El manual de usuario proporciona una guía detallada que permite a los usuarios explorar los métodos de navegación y las funcionalidades específicas de cada pantalla de la aplicación "Hands". Esta herramienta está diseñada para facilitar el aprendizaje de la lengua de signos, ofreciendo desde la navegación básica hasta funciones más avanzadas. Con esta guía, los usuarios pueden sacar el máximo provecho de la aplicación y mejorar sus habilidades en el uso y comprensión de la lengua de signos.

Una vez el usuario accede a la primera pantalla, Índice o inicio, se le presenta una interfaz clara y accesible. En esta pantalla principal, destacan elementos como el logotipo de la aplicación "HANDS" y una serie de botones de navegación que ofrecen acceso rápido a diferentes secciones de la aplicación. Los botones disponibles son "Detección", "Abecedario", "Saludos" y "Números".

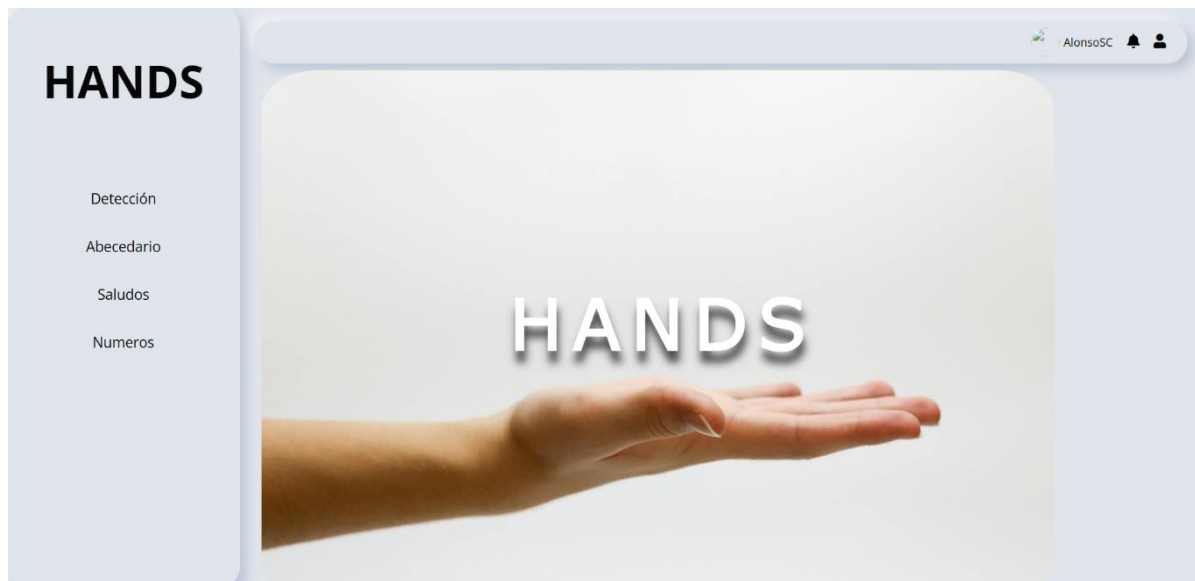


Ilustración 20 Estructura y estilo Pantalla Inicio

La pantalla también muestra una imagen principal, que se puede cambiar haciendo clic sobre ella, proporcionando una experiencia interactiva al usuario. Esta combinación de elementos facilita la navegación y el acceso a las diversas funcionalidades de la aplicación.



Ilustración 21 Estructura y estilo Pantalla inicio 2

Para lograr la navegación el usuario debe de hacer clic en cualquiera de los botones situados en el menú lateral izquierdo. Al pulsar el botón, el usuario podrá pasar a cualquiera de las 5 pantallas.

Una vez el usuario accede a la pantalla "Abecedario", se encuentra con una disposición similar a la pantalla de inicio, con una interfaz clara y accesible. En esta pantalla, el logotipo de la aplicación "HANDS" sigue siendo prominente, junto con el menú lateral izquierdo que contiene los botones de navegación.

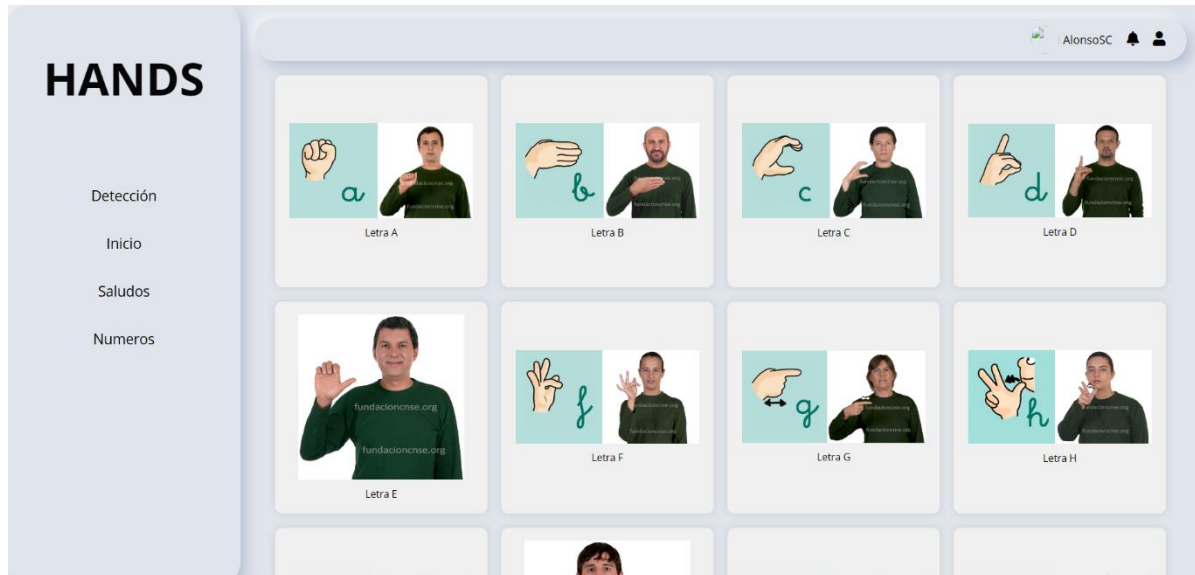


Ilustración 22 Estructura y elementos pantalla Abecedario

Los botones disponibles en este caso son "Detección", "Inicio", "Saludos" y "Números", lo que permite al usuario navegar fácilmente entre las diferentes secciones de la aplicación. Al igual que en la pantalla de inicio, estos botones ofrecen acceso rápido a las funcionalidades correspondientes.

En cuanto al contenido específico de la pantalla "Abecedario", se muestran tarjetas que representan cada letra del abecedario acompañadas de una imagen y el nombre de la letra en mayúsculas. Estas tarjetas son interactivas y pueden seleccionarse haciendo clic sobre ellas, lo que permite al usuario explorar y aprender el alfabeto en lengua de signos de manera intuitiva.

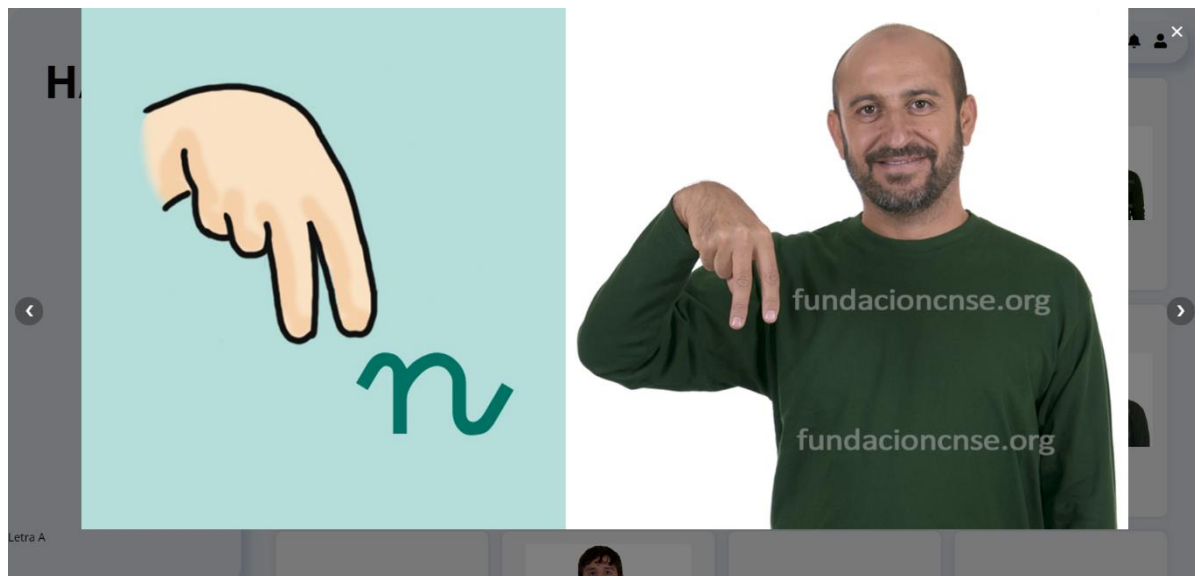


Ilustración 23 Ampliación pantalla Abecedario

Si el usuario pulsa una de las imágenes, esto provocará que pasen a un tamaño mayor y con ello la posibilidad de ir pasando fotografías mediante las flechas para lograr ver el catálogo completo a modo de galería interactiva.

Una vez el usuario accede a la pantalla "Saludos", se encuentra con una disposición similar a las pantallas anteriores, manteniendo la interfaz clara y accesible característica de la aplicación "Hands". En esta pantalla, el logotipo de la aplicación sigue siendo visible en la parte superior, junto con el menú lateral izquierdo que ofrece acceso rápido a otras secciones de la aplicación, como "Detección", "Abecedario", "Números" e "Inicio".

El contenido principal de la pantalla "Saludos" consiste en una serie de tarjetas que representan diferentes expresiones de saludo en lengua de signos. Cada tarjeta contiene una imagen animada (GIF) que representa el saludo correspondiente, junto con el texto descriptivo del saludo en la parte inferior.

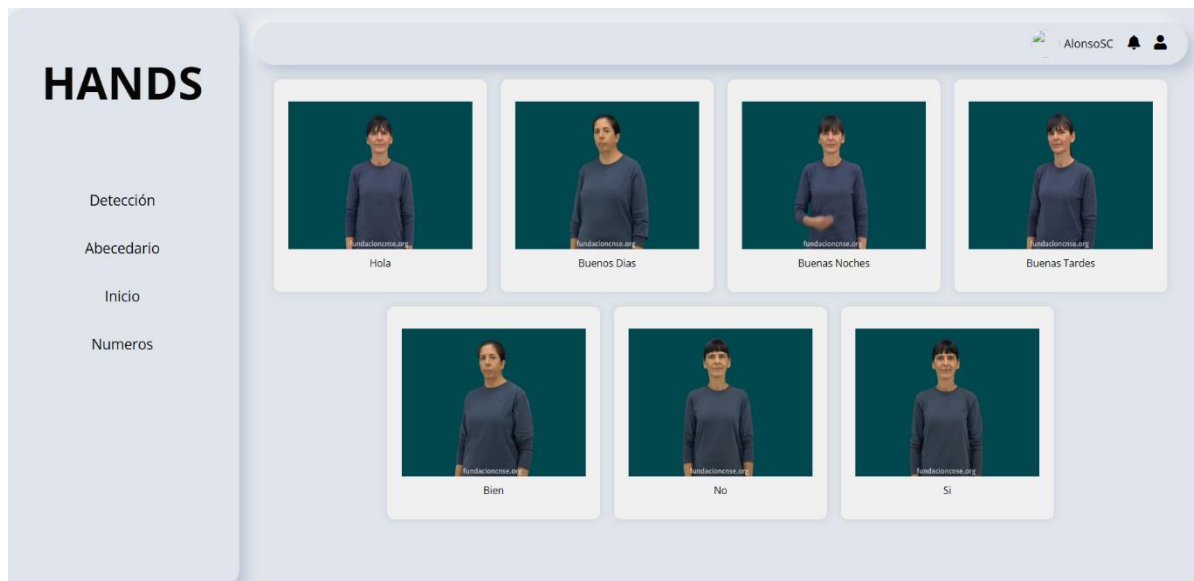


Ilustración 24 Estructura pantalla Saludos

Es importante destacar que los GIFs no se reproducen automáticamente al cargar la página; en cambio, permanecen estáticos hasta que el usuario pasa el cursor sobre ellos. Al hacerlo, los GIFs se reproducen para ofrecer una representación dinámica de los saludos en lengua de signos. Esta característica proporciona al usuario control sobre la interacción con el contenido multimedia de la aplicación.

Además, cada tarjeta es interactiva y puede seleccionarse haciendo clic sobre ella, lo que permite al usuario explorar y aprender las diferentes expresiones de saludo de manera intuitiva y atractiva.

Una vez que el usuario accede a la pantalla "Números", se encuentra con una disposición similar a las anteriores en la aplicación "HANDS". La interfaz sigue siendo clara y accesible, con el logotipo de la aplicación en la parte superior y un menú lateral izquierdo que proporciona acceso rápido a otras secciones de la aplicación, como "Detección", "Abecedario", "Inicio" y "Saludos".

El contenido principal de la pantalla "Números" consiste en una serie de tarjetas que representan los números en lengua de signos. Cada tarjeta contiene una imagen que representa un número, permitiendo al usuario aprender y practicar los números en lengua de signos de manera visual y efectiva.

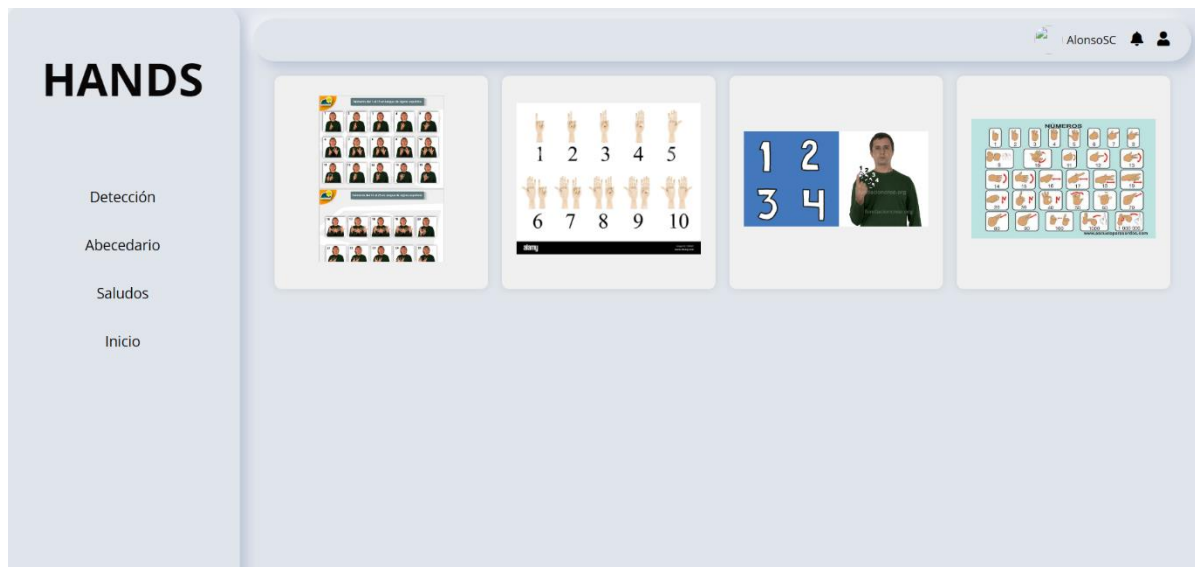


Ilustración 25 Estructura Pantalla Números

Las tarjetas son interactivas y pueden seleccionarse haciendo clic sobre ellas, lo que facilita la exploración de los diferentes números y su representación en lengua de signos.

Una vez que el usuario accede a la pantalla "Detección", se encuentra con una disposición familiar y accesible, característica de la aplicación "HANDS". La interfaz mantiene la claridad visual y la coherencia con otras secciones de la aplicación. En la parte superior, el logotipo de la aplicación es prominente, acompañado de un menú lateral izquierdo que ofrece acceso rápido a diferentes secciones, como "Inicio", "Abecedario", "Números" y "Saludos".

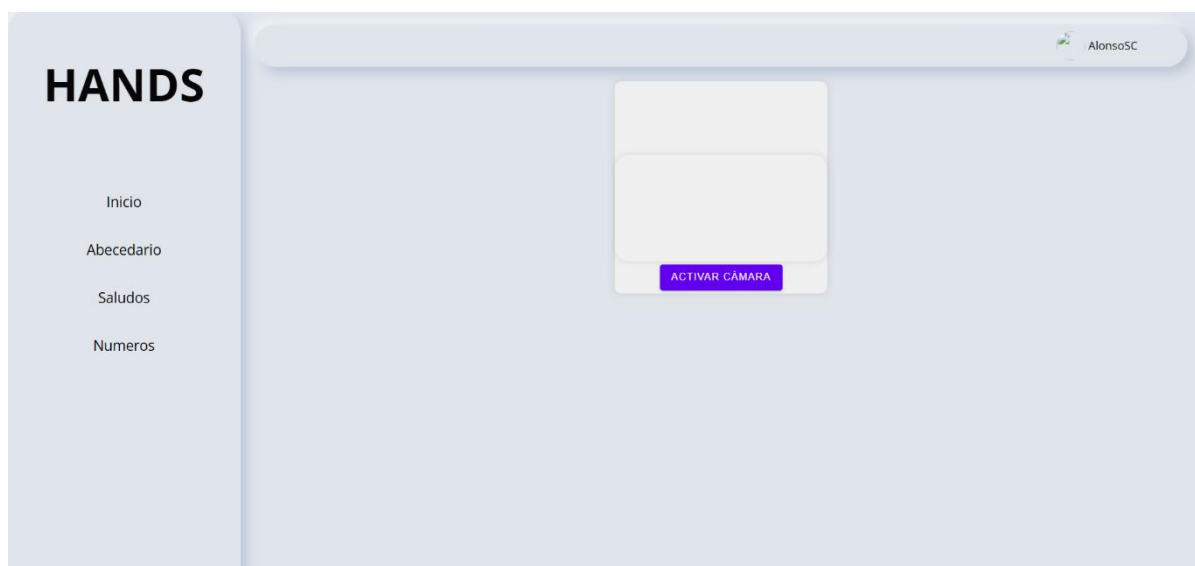


Ilustración 26 Estructura pantalla Detección 1

El contenido principal de la pantalla "Detección" se centra en las opciones para realizar la detección de gestos de lengua de signos. El usuario tiene dos alternativas para llevar a cabo esta función:

La primera y principal, la más desarrollada en este caso, que cuenta con una detección de las letras del abecedario de lengua de signos.

Esta opción ocurre cuando el usuario desde cualquier pantalla anterior a la actual (detección) hace clic sobre el botón de detección. Dicha acción lanza un hilo en segundo plano que abre una nueva pestaña, normalmente aparece en la barra de herramientas inferior en el SO de Windows.



Ilustración 27 Barra de herramientas de Windows10

En esta nueva pantalla, el usuario puede verse en tiempo real y realizar gestos con las manos a modo de comunicación siguiendo el alfabeto LSE. Esta funcionalidad capta la posición de las manos y en función a las formas hechas con las manos determina la letra que se está formando. Para detener esta funcionalidad el usuario solo debe pulsar la letra "Q" en su teclado.



Ilustración 28 Detección de signos en tiempo real 1

La segunda opción, la menos desarrollada, es la posibilidad de hacer una detección de gestos predeterminados haciendo clic en el botón de "Activar Cámara". Esto inicia la acción de la webcam del dispositivo y el usuario puede hacer estos gestos viéndose en tiempo real. Para detener esta acción solo se debe volver a hacer clic en el mismo lugar, que en este caso se llama "Desactivar Cámara"

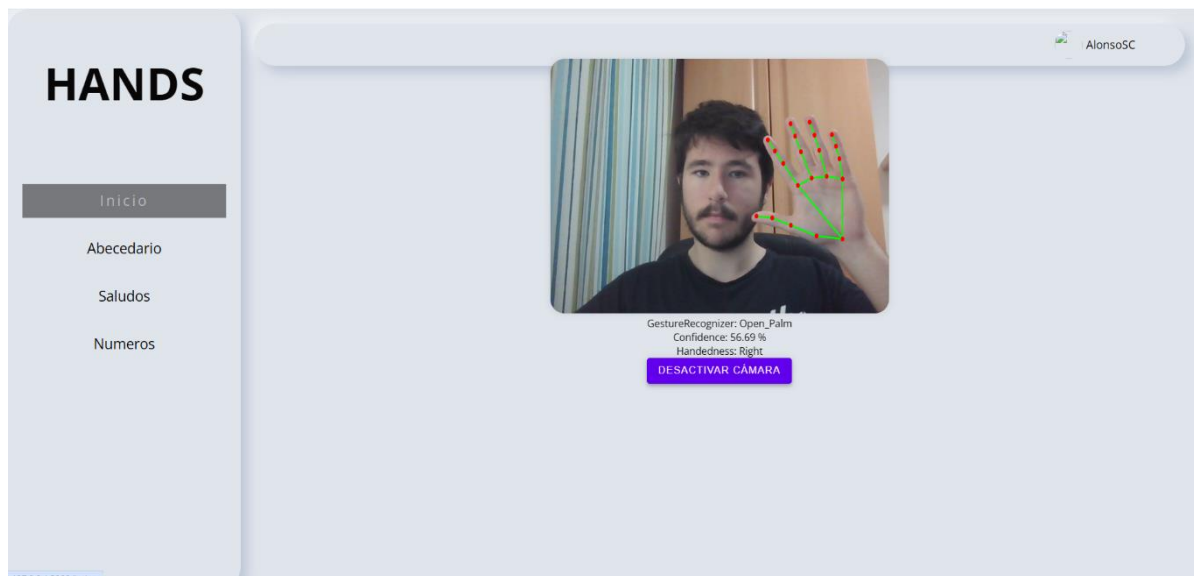


Ilustración 29 Detección de signos en tiempo real 2

7.2 Manual de instalación

“Hands” no requiere de instalación al uso como en un software comercial. Para poder utilizar la versión actual de la app, es necesario tener un IDE como Visual Studio Code o PyCharm, tener instalado Python 3.0 y hacer la instalación de las librerías que hay en el fichero requirements.txt. Para hacer esto se requiere tener acceso a internet.

Con la siguiente sentencia se realiza la instalación:

```
pip install -r requirements.txt
```

El fichero requirements contiene diversas librerías comunes en el desarrollo de aplicaciones de este estilo. Si algunas de esas librerías ya están instaladas en el equipo del usuario, solo se descargarán las necesarias.

Algunas de estas librerías son MediaPipe para la detección de las manos, numpy para el manejo de matrices y operaciones numéricas o OpenCv para el procesamiento de imágenes y la detección de manos.

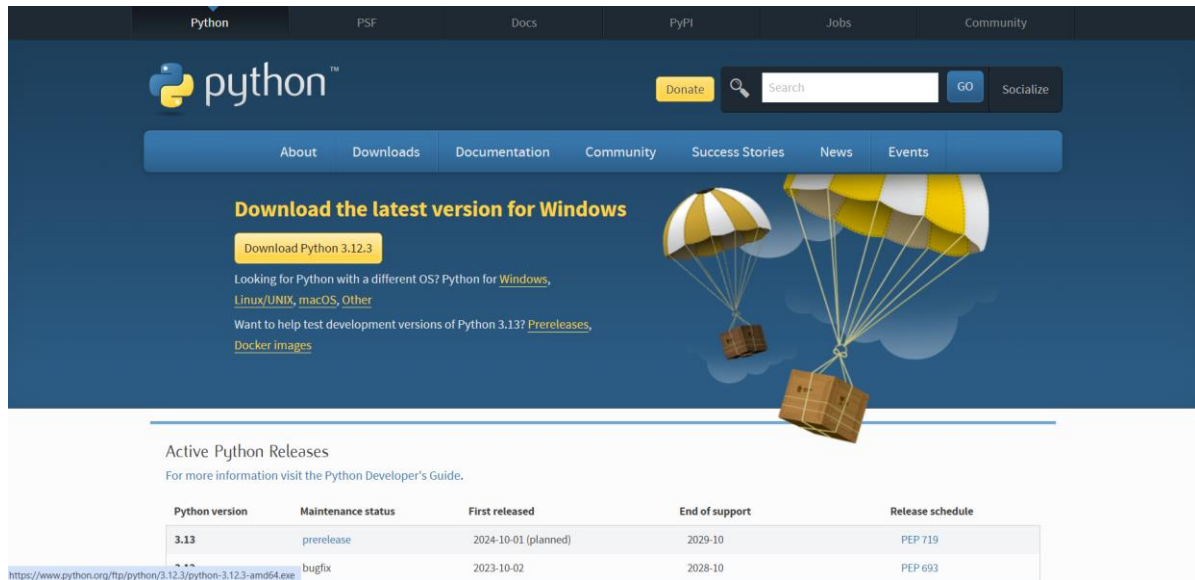


Ilustración 30 Descarga de Python

8 Conclusiones y posibles ampliaciones

Conclusiones: Un reto superado.

El desarrollo de la aplicación "Hands" ha supuesto un reto personal y técnico de gran envergadura para el alumno, quien ha logrado superar con éxito las dificultades inherentes a un proyecto en solitario. A lo largo del proceso, ha adquirido valiosas habilidades en el ámbito del desarrollo de aplicaciones multiplataforma, desde la elección de las herramientas adecuadas hasta la implementación de las funcionalidades deseadas.

Más allá del aprendizaje técnico, el proyecto "Hands" puede tener un impacto significativo en la promoción de la lengua de señas y la inclusión social. La aplicación ha demostrado ser una herramienta útil para el aprendizaje de este idioma.

Posibles ampliaciones:

- Ampliar la base de datos de signos: Se podrían incluir más signos en la aplicación, abarcando un vocabulario más amplio y completo.
- Desarrollar un sistema de reconocimiento de gestos más avanzado: Se podría implementar un sistema de reconocimiento de gestos más preciso y robusto, que permita una mejor interacción con la aplicación.
- Crear una versión móvil de la aplicación: Se podría desarrollar una versión móvil de la aplicación para que los usuarios puedan acceder a ella desde sus smartphones y tablets.

- Incorporar gamificación: Se podrían incorporar elementos de gamificación a la aplicación para hacerla más atractiva y motivante para los usuarios.
- Desarrollar un sistema de tutoría virtual: Se podría desarrollar un sistema de tutoría virtual que proporcione a los usuarios retroalimentación y orientación personalizada durante su aprendizaje.
- Traducir la aplicación a otros idiomas: Se podría traducir la aplicación a otros idiomas para que pueda ser utilizada por personas de todo el mundo.

El proyecto "Hands" tiene un gran potencial para seguir creciendo y mejorando. Con el desarrollo continuo, la aplicación "Hands" puede convertirse en una herramienta aún más valiosa para el aprendizaje de la lengua de señas y la promoción de la inclusión social.

9 Bibliografía

adobe. (2024). *ColorAdobe*. Obtenido de <https://color.adobe.com/es/>

AprendelIngenia. (2024). *AprendelIngenia*. Obtenido de Proyectos de visión artificial:
<https://github.com/AprendelIngenia>

developer.mozilla.org. (2024). *Documentación JavaScript*. Obtenido de
<https://developer.mozilla.org/es/docs/Web/JavaScript>

developers.google. (2024). *LandMarks de MediaPipe*. Obtenido de
https://developers.google.com/mediapipe/solutions/vision/hand_landmarker

Martín, F. P. (2021). *Lenguaje de Marcas*. Paraninfo.

pypi.org. (2024). *pypi mediapipe*. Obtenido de <https://pypi.org/project/mediapipe/>

python.org. (2024). *Documentación oficial de Python 3.11*. Obtenido de
<https://docs.python.org/es/3.11/>.

python.org. (2024). *python downloads*. Obtenido de <https://www.python.org/downloads/>

10 Anexos

Enlace del proyecto Hands en GitHub

<https://github.com/AlonsoSorCar/ProyectoHands.git>

