UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

TÉCNICAS DE INTELIGENCIA ARTIFICIAL EN DISPOSITIVOS IOT

VILLACRES CANGO HJALMAR LEONEL
INGENIERO DE SISTEMAS

MACHALA
2021

UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

TÉCNICAS DE INTELIGENCIA ARTIFICIAL EN DISPOSITIVOS IOT

VILLACRES CANGO HJALMAR LEONEL
INGENIERO DE SISTEMAS

MACHALA
2021

UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

TRABAJO TITULACIÓN
PROPUESTAS TECNOLÓGICAS

TÉCNICAS DE INTELIGENCIA ARTIFICIAL EN DISPOSITIVOS IOT

VILLACRES CANGO HJALMAR LEONEL
INGENIERO DE SISTEMAS

HERNANDEZ ROJAS DIXYS LEONARDO

MACHALA, 27 DE ABRIL DE 2021

MACHALA
2021

Tesis

INFORME DE ORIGINALIDAD

1%	2%	0%	1%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	repositorio.espe.edu.ec Fuente de Internet	<1%
2	revistatecnologiadigital.com Fuente de Internet	<1%
3	scielo.conicyt.cl Fuente de Internet	<1%

Excluir citas	Activo	Excluir coincidencias	< 50 words
Excluir bibliografía	Activo		

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

El que suscribe, VILLACRES CANGO HJALMAR LEONEL, en calidad de autor del siguiente trabajo escrito titulado Técnicas de Inteligencia Artificial en dispositivos IoT, otorga a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tiene potestad para otorgar los derechos contenidos en esta licencia.

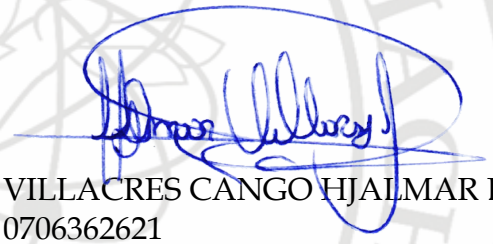
El autor declara que el contenido que se publicará es de carácter académico y se enmarca en las dispociones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

El autor como garante de la autoría de la obra y en relación a la misma, declara que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asume la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

Machala, 27 de abril de 2021



VILLACRES CANGO HJALMAR LEONEL
0706362621

DEDICATORIA

El presenta trabajo se lo dedico a toda mi familia los cuales han sido fuente de inspiración para lograr este peldaño más en mi formación académica, pero, sobre todo, agradezco a mis padres y hermanos, los cuales han sido siempre el pilar fundamental de mi vida y con su apoyo y amor incondicional me impulsan a seguir siempre adelante, lo cual me ha servido de aliento para cumplir mis metas y objetivos propuestos en el transcurso de mi formación Académica-Profesional.

De igual manera a mis maestros, los cuales dan todo para inculcar de conocimientos y valores al estudiante y saberme guiar por el camino correcto durante todo el transcurso de mi carrera universitaria.

Sr. Hjalmar Leonel Villacrés Cango

AGRADECIMIENTO

Agradezco principalmente a Dios, el cual, con su bendición me brinda la fortaleza necesaria para no dejarme vencer y superar todos los obstáculos que se me presentan en la vida. También por permitirme estar en compañía de mis padres y hermanos, los cuales, mediante sus enseñanzas me demostraron que nada es imposible con esfuerzo y dedicación.

También agradezco a todos los docentes de la Escuela de Informática de la Universidad Técnica de Machala por haber compartido sus conocimientos y por sus valiosos consejos, los cuales sirvieron de mucha ayuda en su momento. Además de amigos y compañeros que siempre estuvieron conmigo durante todo el transcurso de nuestra carrera universitaria.

Finalmente quiero expresar mi más grande y sincero agradecimiento a mi tutor, el Ing. Dixys Leonardo Hernández Rojas por su gran ayuda, colaboración y por su entereza al despejar todas mis interrogantes en el desarrollo de este trabajo.

Sr. Hjalmar Leonel Villacrés Cango

RESUMEN

Internet es una tecnología que va creciendo de manera exponencial, y de esta van apareciendo nuevas fuentes de información que dan paso al desarrollo de nuevas tecnologías. Una de ellas y que va ganando mucho terreno es el internet de las cosas o IoT, tecnología que ha crecido de manera muy rápida en los últimos años y esto es muy evidente debido a la gran conectividad que existe entre diversos dispositivos y la accesibilidad para obtenerlos. Actualmente son muchos los hogares que cuentan con acceso a internet y cada uno de ellos mantienen una conectividad entre varios dispositivos utilizando una misma red, dispositivos como celulares, tablets, laptops, televisores e incluso neveras inteligentes son las que recientemente permiten conectarse a Internet para compartir información. Los usuarios observan que la integración de estos dispositivos a la red permite una rápida manipulación de los mismos y a la vez sacan provecho de las tecnologías que estos presentan, creciendo más la conectividad entre dispositivos y aumentando el interés del Internet de las cosas aportando en el crecimiento del mismo.

Uno de los ámbitos en el cual nos estamos sumergiendo es en el Internet de las cosas, para ser más exactos, en la domótica debido a la automatización de dispositivos inteligentes. La domótica juega un papel muy importante en el IoT debido a que el mismo se lo considera como automatización, en la que intervienen dispositivos capaces de conectarse a la red y comunicarse entre sí compartiendo información fundamental para el usuario y a la vez una fácil manipulación en los dispositivos del hogar. Esta manipulación se logra por medio de herramientas y plataformas orientadas a la domótica, una de ellas y la más popular es Home Assistant capaz de alojar dispositivos domóticos para su fácil manipulación, además de que el mismo contiene un sinnúmero de herramientas que permiten operar de manera muy sencilla.

El otro ámbito que tratamos en el presente trabajo es la traducción de gestos originarios de la lengua de señas ecuatorianas que en el presente resulta ser una barrera entre las personas que entienden la lengua y las que no la comprenden, es decir, que el traductor permitirá la comunicación de ciertas

palabras, saludos y preguntas a texto y voz. Con esto generamos un sistema de bajo costo capaz de brindar una solución aceptable a este problema.

La automatización de dispositivos domóticos integrados en Home Assistant comúnmente se logra utilizando un dispositivo móvil o una laptop y esto incluye tiempo valioso que puede ser usado para realizar alguna otra actividad. Para ello, en el presente trabajo se plantea diseñar e implementar un sistema que permita reconocer diez gestos para su traducción a texto y voz, además de operar dispositivos domóticos integrados en Home Assistant por medio del protocolo de comunicación MQTT utilizando un gesto en particular. Para este proyecto se utilizarán componentes de bajo costo y de fácil accesibilidad.

Los resultados obtenidos en el proyecto fueron satisfactorios, debido a que el sistema cumplió con su objetivo con resultados muy favorables según las pruebas realizadas. Entre los gestos utilizados para la automatización de dispositivos los resultados fueron muy precisos, pero en la traducción de saludos y preguntas a texto y voz, este fue poco preciso con ciertos gestos debido a la similitud que presentaban y a su complejidad, recordando que el proyecto funciona con un solo sensor y con datos basados únicamente en un acelerómetro de 3 ejes.

Palabras claves:

IoT, Domótica, Machine Learning, Redes Neuronales Artificiales, Acelerómetro, Automatización, Home Assistant, Lengua de señas, LSEC.

ABSTRACT

The Internet is a technology that is growing exponentially, and from this new source of information are emerging that give way to the development of new technologies. One of them and that is gaining a lot of ground is the internet of things or IoT, a technology that has grown very quickly in recent years and this is very evident due to the great connectivity that exists between various devices and the accessibility to obtain them. Currently there are many homes that have internet access and each of them maintains connectivity between several devices using the same network, devices such as cell phones, tablets, laptops, televisions and even smart refrigerators are the ones that recently allow connecting to the Internet to share information. Users observe that the integration of these devices to the network allows a rapid manipulation of them and at the same time they take advantage of the technologies that they present, increasing the connectivity between devices and increasing the interest of the Internet of Things, contributing to the growth of the same.

One of the areas in which we are immersing ourselves is in the Internet of things, to be more exact, in home automation due to the automation of smart devices. Home automation plays a very important role in the IoT because it is considered as automation, in which devices capable of connecting to the network and communicating with each other intervene, sharing fundamental information for the user and at the same time easy manipulation in the household devices. This manipulation is achieved by means of tools and platforms aimed at home automation, one of them and the most popular is Home Assistant, capable of housing home automation devices for easy handling, in addition to containing a number of tools that allow operating in an easy way. very simple.

The other area that we deal with in this work is the translation of gestures originating in the Ecuadorian sign language, which at present turns out to be a barrier between people who understand the language and those who do not understand it, that is, that the translator will allow the communication of certain words, greetings and questions to text and voice. With this, we generate a low-cost system capable of providing an acceptable solution to this problem.

Automation of home automation devices integrated into Home Assistant is commonly accomplished using a mobile device or laptop and this includes valuable time that can be used for some other activity. For this, in the present work it is proposed to design and implement a system that allows to recognize ten gestures for their translation into text and voice, in addition to operating domotic devices integrated in Home Assistant through the MQTT communication protocol using a particular gesture. Low-cost and easily accessible components will be used for this project.

The results obtained in the project were satisfactory, because the system fulfilled its objective with very favorable results according to the tests carried out. Among the gestures used for the automation of devices, the results were very precise, but in the translation of greetings and questions to text and voice, this was not very precise with certain gestures due to the similarity they presented and their complexity, remembering that the project works with a single sensor and with data based solely on a 3-axis accelerometer.

Keywords:

IoT, Home Automation, Machine Learning, Artificial Neural Networks, Accelerometer, Automation, Home Assistant, Sign Language, LSEC.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En nuestro medio existe una gran barrera en la comunicación con personas privadas de hablar y escuchar, sobre todo si la comunicación se basa en una lengua de señas la cual no todos comprenden, pero si conocen de su existencia. Son varias las organizaciones que capacitan a estas personas para mantener activo este medio de comunicación y así generar una inclusión entre esta comunidad con la sociedad. La lengua de señas se encuentra incluida en varios ámbitos que requieren una simplicidad en la ejecución de actividades, como por ejemplo en la realidad virtual que a través de guantes se pueden simular el tacto en entornos virtuales, traductores de gestos comerciales que permiten la comunicación con la sociedad sordomuda (aunque no son completamente funcionales y su precio puede ser muy elevado), en la domótica para la manipulación de dispositivos, entre otros.

Este es el área en la que el presente proyecto se enfoca, en un sistema de reconocimiento de gestos realizados con una mano que permitan la traducción de una lengua de señas a texto y voz, asimismo permitirá la automatización de dispositivos, con esto solventamos los problemas de comunicación y se aporta en el ámbito de la domótica con un sistema capaz de manipular un dispositivo con un gesto utilizando algoritmos de redes neuronales y componentes de bajo costo como solución.

El presente documento se encuentra estructurado en tres capítulos detallados de la siguiente manera:

Capítulo I: Se describe el contexto en el que se desarrollará el proyecto, igualmente incluye la descripción y la necesidad por la cual se construye el reconocimiento de gestos, justificando la importancia por la cual el proyecto es relevante en el área de estudio seleccionada.

Capítulo II: Se describe la definición del prototipo, los objetivos y el proceso de desarrollo en el diseño y ensamblaje.

Capítulo III: El proyecto elaborado se somete a pruebas con diferentes usuarios, siguiendo métricas de evaluación para determinar su rendimiento; finalizando con las conclusiones y recomendaciones para trabajos futuros.

1. CAPÍTULO I: DIAGNÓSTICO DE NECESIDADES Y REQUERIMIENTOS

1.1. Ámbito de Aplicación: Descripción del contexto y hechos de interés

El Consejo Nacional para la igualdad de Discapacidades del Ecuador (CONADIS) estima que existen 65821 personas en el país con algún tipo de discapacidad auditiva y que la lengua oficial para esta comunidad se la denomina Lengua de Señas Ecuatorianas (LSEC) [1]. Además, existen otras formas de comunicación alternativas para estas comunidades como el sistema braille, el oralismo, entre otros. Con esto se puede resumir que el País posee su propio lenguaje de señas y que el mismo se encuentra respaldado por el gobierno.

Partiendo de la lengua de señas podemos determinar su importancia para la comunicación entre comunidades sordomudas y a la vez este puede ser utilizado para la automatización de dispositivos. El proyecto propuesto integra el uso de gestos de la LSEC, garantizando de que el mismo puede ser entendido perfectamente por esta comunidad y a la vez servirá como inicio para que diferentes usuarios conozcan y aprendan de esta lengua.

Smart Home es considerada una de las aplicaciones más prominentes de la IoT, teniendo como característica principal la automatización y como objetivo principal reducir el esfuerzo humano. Por lo tanto, en los sistemas de automatización el uso de tecnologías inalámbricas proporciona ventajas a diferencia de la red cableada [2]. Uno de los principales propósitos del sistema a desarrollar, es reducir el tiempo en la automatización de dispositivos utilizando gestos para su manipulación.

1.2. Establecimiento de requerimientos

Para la administración de dispositivos domóticos existe Home Assistant (Hassio) como principal alternativa, siendo esta la idónea tanto para expertos como para aficionados de la domótica. La misma permite la interacción con los dispositivos en tiempo real y observar su comportamiento, pero, esta interacción necesita de un dispositivo como intermediario, sea un teléfono Smart o un computador. El

sistema propuesto no requerirá de estos, solo bastará con realizar un gesto y un microcontrolador se encargará de recibir el mensaje y realizar la automatización deseada. Asimismo, permitirá la traducción del gesto a texto y a voz.

Los requerimientos de la presente propuesta tecnológica se apoyan de una serie de gestos pertenecientes a la LSEC, los cuales servirán como medio de comunicación con el dispositivo ESP32 (seleccionado como procesador). Por otro lado, existen requerimientos adicionales para que el microcontrolador funcione y genere los resultados solicitados. En fin, todos estos requerimientos permitirán culminar con un prototipo capaz de recibir señales de un sensor para posteriormente generar un resultado, sea para la automatización de dispositivos o para la traducción a voz y texto.

1.3. Justificación del requerimiento a satisfacer

La integración de dispositivos a la red y su manipulación desde cualquier punto del hogar genera que a varias personas les interese la domótica, además de que proporciona flexibilidad a la hora de realizar esta actividad y reduce el tiempo al no hacerlo de manera manual. Sobre todo, cuando vivimos en una era completamente tecnológica y donde la IoT se encuentra presente en diversas áreas sin olvidar el hogar. Según [3], los avances recientes permiten a los usuarios controlar y consumir información de varios dispositivos Smart, debido a esto, una de las tendencias de investigación se centra en la conectividad permitiendo a los usuarios acceder y controlar objetos desde cualquier rincón del hogar (con acceso a internet) y en cualquier momento.

La contribución que brinda el proyecto tecnológico es la divulgación de la LSEC, utilizándose como medio de comunicación entre el usuario y el sistema, además de que el uso de esta lengua permite una mayor accesibilidad. Por otra parte, el sistema contiene 10 gestos dinámicos entre los que se dividen 8 para su traducción a texto y voz entre los que incluye gestos como saludos y preguntas (todas orientados a la comunicación); y 2 gestos que se utilizarán para la automatización de dispositivos domóticos, tanto para encender y apagar un enchufe y un bombillo integrado en Hassio (la comunicación se la realizará a través de MQTT). El proyecto es una alternativa eficiente y de bajo costo para la traducción de gestos y automatización de dispositivos Smart.

2. CAPÍTULO II: DESARROLLO DEL PROYECTO

2.1. Definición del prototipo tecnológico

A lo largo de la historia se van integrando nuevas tecnologías capaces de solventar problemas o facilitar actividades a la sociedad, estas van apareciendo en diversos ámbitos y a la vez fortaleciéndolos con la automatización de ciertas actividades que anteriormente tomaban tiempo realizarlas. Actualmente estas se encuentran a un clic o a un gesto de distancia. El ámbito en el cual nos estamos sumergiendo es en el internet de las cosas, para ser más exactos, en la domótica debido a la automatización de dispositivos inteligentes.

El internet de las cosas juega un papel muy importante en la vida humana y en el campo educativo, debido a que son capaces de proporcionar información y completar tareas mientras el usuario realiza otra actividad [4]. La arquitectura propuesta en la Figura 1 describe al sistema desarrollado, utilizando como tecnología principal las técnicas de inteligencia artificial en dispositivos IoT. Esta describe el ingreso de gestos por medio de señales realizadas por un sensor MPU6050 a un dispositivo ESP32 que las procesa utilizando redes neuronales artificiales reconociendo el gesto y realizando una actividad programada, entre las que se destacan la manipulación de dispositivos domóticos alojados en la plataforma Home Assistant a través de MQTT y la traducción de los mismos a texto utilizando un display y voz por medio de un amplificador de sonido.

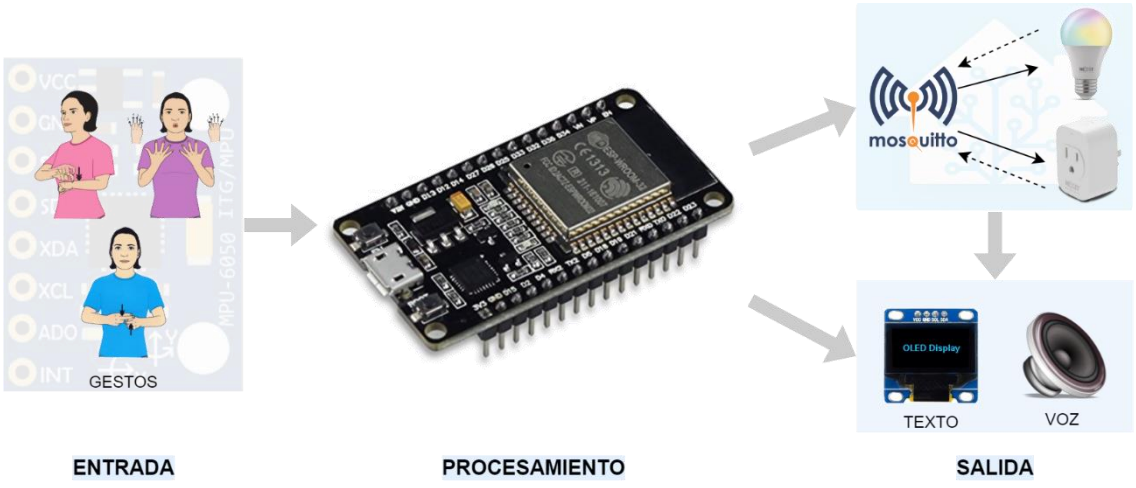


Figura 1: Arquitectura del sistema propuesto

Fuente: Elaboración del Autor

2.2. Fundamentación teórica del prototipo

La fundamentación teórica de la propuesta tecnológica del presente documento se describe en la Figura 2 como mapa conceptual. Cada uno de los temas mencionados poseen subtemas que profundizan los aspectos importantes a conocer para entender el proceso del proyecto.

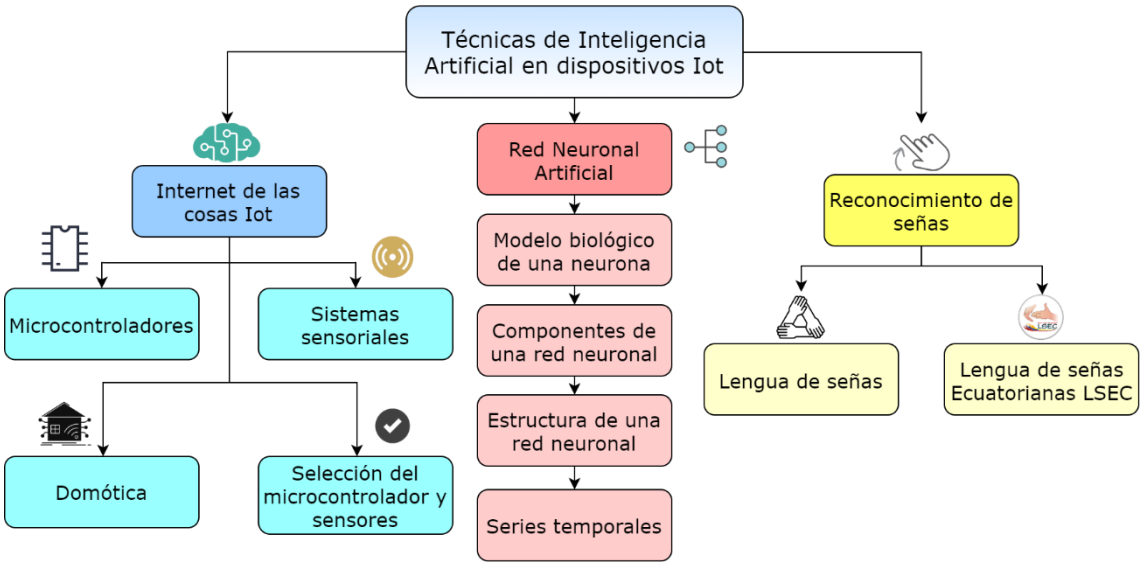


Figura 2: Mapa conceptual de la fundamentación teórica del prototipo.

Fuente: Elaboración del Autor

2.2.1. Internet de las cosas (IoT)

Internet es una tecnología que va creciendo de manera exponencial, y de esta van apareciendo nuevas fuentes de información que dan paso al desarrollo de nuevas tecnologías. Según [5], el Internet de las cosas es una tecnología que está incursionando en diversos campos como la salud, logística, ciudades inteligentes, hogar inteligente, además de nuevos campos como la agricultura de precisión, como por ejemplo [6], cuyo trabajo propone un panel de control de riego inteligente basado en IoT, como parte de un sistema SCADA para la automatización del riego de cultivos de banano.

El internet de las cosas se trata de dispositivos conectados de baja potencia y bajo consumo para diseñar una infraestructura de conectividad escalable para detectar y analizar datos provenientes de diversas fuentes [7]. Mientras que [8] lo describe como una extensión de internet al integrar redes móviles, redes sociales y cosas inteligentes para proporcionar mejores servicios o aplicaciones

a los usuarios. En la actualidad la IoT se caracteriza por el uso de dispositivos con recursos limitados, de poca memoria, con capacidades de comunicación limitada, con baja potencia informática y alta dependencia a las baterías [9].

Los dispositivos como celulares, tabletas, relojes inteligentes, televisores, neveras, entre otros, actualmente nos brindan la opción de integrarlos en nuestra red, y la tecnología IoT los hace capaces de compartir, comunicarse y transferir información a través de internet, pero estos no son los únicos dispositivos, existen otros compatibles con IoT como Arduino, Raspberry PI, entre otros [10]. El objetivo de la IoT es controlar objetos o dispositivos eléctricos que nos rodean de una manera más fácil y fluida [11]. Esto nos garantiza una vida más rentable y ecológica utilizando dispositivos que proveen un bajo consumo de recursos.

2.2.1.1. Domótica

Existen muchas maneras de definir a la domótica, si revisamos diferentes trabajos observaremos la definición planteada por varios autores, pero siempre utilizando una palabra, automatización. Este concepto siempre va acompañado por IoT y la comunicación entre dispositivos. Según [12], es una aplicación de internet de las cosas que permite a los usuarios controlar y monitorear los dispositivos domésticos en tiempo real a través de internet. Así mismo lo describe [13], el cual menciona que implica el control y la monitorización en tiempo real de varios electrodomésticos.

La palabra Smart Home se ha convertido en un término frecuentemente utilizado, debido a que vivimos en un mundo cada vez más automatizado. En la domótica no puede faltar este término, al cual se lo define como el proceso de automatizar los diversos dispositivos electrónicos de un hogar convirtiéndolo así en un hogar inteligente [14]. El mismo autor describe que este concepto es adoptado con el fin de reducir la intervención humana, mejorar la eficiencia energética (uso de dispositivos de bajo consumo) y la productividad.

Hacer que la casa sea inteligente es permitir la ejecución automática de varios comandos después de analizar los datos recopilados de diferentes dispositivos. La automatización se puede lograr mediante el uso del internet de las cosas (IoT) [15]. La IoT es uno de los pilares fundamentales de la domótica, debido a que

estos facilitan la interacción entre todos los dispositivos que integran el hogar, permitiendo la automatización de los mismos.

Otro de los conceptos muy importantes en la domótica es la comunicación entre dispositivos, en el que se encuentra MQTT, siendo el idóneo para trabajar en conjunto con dispositivos IoT, debido a que el mismo es un protocolo de mensajería de baja potencia y de poca memoria. MQTT se lo define como un protocolo de capa de aplicación con una sobrecarga muy ligera, adecuado para IoT, M2M y WSN, este funciona según el principio de publicar/suscribir [16]. Su protocolo de mensajería ligero lo convierte en el adecuado para comunicarse con dispositivos de borde de IoT con recursos limitados [17]. Desde el punto de vista de [18], MQTT es la mejor alternativa por ser un protocolo concebido para ambientes de recursos restringidos. Por esta razón MQTT es muy popular entre los desarrolladores de IoT, siendo utilizado en aplicaciones como robótica, smartwatch, atención médica y domótica [19].

2.2.1.2. Microcontroladores

Existen muchas empresas dedicadas al hardware libre como los son Arduino, Pycom, Artik, Xbee, Espressif, entre otros, que facilitan la conexión e implementación de hardware en placas impresas que contienen diferentes sensores y sobre todo un microprocesador, además brindan el software necesario para facilitar su programación [20]. Es decir, son muchas las opciones que se tiene en cuanto a microcontroladores, pero todos obedecen un mismo término de bajo consumo, lo cual lo convierte en idóneos para la tecnología IoT.

Para la selección del mismo es importante entender el significado de un microcontrolador, el cual, según [21], es un sistema económico y programable que generalmente incluye memoria e interfaces de E/S en un solo chip. [14] lo define como un dispositivo de control que viene integrado con periféricos, memoria y un procesador.

La facilidad en su programación y la cantidad de componentes que este integra, adicionando el bajo coste y bajo consumo, lo convierte en una tecnología accesible para el desarrollo de aplicaciones IoT. Los componentes que este integra serán de vital importancia en su elección, debido a que el mismo será programado para procesar la respuesta producida por el sensor.

2.2.1.3. Sistemas sensoriales

Actualmente son muchos los sensores que permiten la medición de ángulos de inclinación y que proveen de señales que permiten determinar la posición de un objeto, uno de ellos y más utilizado es el acelerómetro, el cual, permite detectar fuerzas dinámicas como vibraciones o movimientos.

Existen otros dispositivos capaces de brindar señales que permitan determinar los movimientos de una mano, una de ellas son las señales EMG las cuales contienen información sobre la intención de un movimiento. Existen otras señales como las que arrojan los sensores de flexión cuya señal varía dependiendo la flexión en la que se encuentre.

En ámbitos, como el planteado en este proyecto, se utilizan sensores cuyas señales permitan determinar el movimiento de una mano. Actualmente, la aplicación de estos sistemas son muy importantes y se utilizan en varias aplicaciones como prótesis inteligente, control de dispositivos, dispositivos de rehabilitación y reconocimiento de lengua de señas [22].

2.2.1.4. Justificación de la selección del microcontrolador y sensores

El microcontrolador es la pieza clave del proyecto, debido a que el mismo será el encargado de ejecutar todas las actividades. Por ende, su selección es importante. Con una revisión a varios dispositivos, el módulo ESP32 se perfila como el candidato perfecto por las características que este trae, además de su bajo costo. La combinación de un solo chip Wi-Fi y Bluetooth, un procesador de dos núcleos y un rico conjunto de periféricos lo hace líder en su segmento y el idóneo para proyectos más complejos [23].

La mayoría de dispositivos que se encuentran actualmente son bastante costosos o grandes en términos de tamaño y peso, mientras que el ESP32 goza de ser bastante pequeño y poderoso, además, el módulo ESP32 fue diseñado para ser una solución perfecta para dispositivos IoT [24].

El módulo seleccionado para este proyecto es el ESP-WROOM-32, integrado de antena, oscilador y flash, además de otras características, las cuales se describen en la Tabla 1.

Tabla 1: Características del microcontrolador ESP32 (ESP-WROOM-32)

Detalle	**Chip (módulo):** ESP32 (ESP-WROOM-32)
CPU	Tensilica Xtensa LX6 32 bit Dual-Core at 160/240 MHz
SRAM	520 KB
FLASH	2 MB (máximo 64 MB)
Voltaje	2.2v a 3.6v
Corriente de funcionamiento	Promedio 80 mA
Programación	C, C++, Lua, etc.
Código abierto	Si
Conectividad:	
Bluetooth	4.2 BR/EDR + BLE
UART	3
Wi-Fi	802.11 b/g/n
I/O:	
GPIO	32
SPI	4
12c	2
PWM	8
ADC	18 (12 bits)
DAC	2 (8 bits)
Otros:	
Tamaño	25.5 x 18.0 x 2.8 mm
Precio	£8,00 ≈ $10,98

Fuente: Tomado de [24]

Debido a las características mencionadas en la Tabla 1, la placa es la mejor opción para aficionados y profesionales de la electrónica [25]. Sin duda, Espressif lanzó un gran producto en el 2016.

Para la selección del sensor que proveerá las señales a procesar para la detección de gestos, se tomó en cuenta los tipos de movimientos a realizar, los cuales son dinámicos y muy diferentes entre sí, además de que el movimiento de las manos es más frecuente que el de los dedos. Por tal motivo, entre la cantidad de sensores que existen se optó por el MPU6050, el cual posee acelerómetro y giroscopio en un solo dispositivo, dejando de lado otros más sofisticados, pero con un precio más elevado, los cuales según [26], son caros y no son adecuados para un uso prolongado, además estos pueden causar irritación en la piel y a menudo son muy ruidosos cuando se usan a largo plazo.

Otro motivo principal por el que se hará uso del MPU6050 es el bajo costo, la fácil programación y su fácil manipulación. Evitando así el consumo excesivo de recursos al utilizar una mayor cantidad de sensores.

2.2.2. Red Neuronal Artificial

El paradigma de redes neuronales artificiales (RNA) es muy popular dentro de la Inteligencia Artificial, este es muy utilizado en la solución de problemas de clasificación supervisada. Según [27] son modelos que procesan información y hacen predicciones, es decir, una vez entrenada sirve como una herramienta analítica para pronosticar resultados previamente controlados. Estos modelos son diseñados para clasificar datos, predecir valores y ayudar en el proceso de toma de decisiones.

Las redes neuronales artificiales son modelos matemáticos muy avanzados que pretenden simular el funcionamiento del sistema nervioso, estos están constituidos por un conjunto de neuronas o nodos conectados unos con otros. En la actualidad es muy común el uso de este algoritmo en diversas áreas, según [28], estos modelos fueron introducidos con éxitos en diferentes campos como en la medicina con la clasificación de imágenes hasta la conducción autónoma. El mismo autor indica que estos modelos son procesados en servidores en la nube debido al alto consumo computacional que requiere.

En el algoritmo de redes neuronales existen varios métodos que influyen en su entrenamiento y validación que pueden producir un alto nivel de precisión en su predicción. Para obtener un buen modelo es necesario ajustar Hiper parámetros hasta conseguir un nivel de precisión capaz de cumplir con el objetivo.

2.2.2.1. Modelo biológico de una Neurona.

De acuerdo con [29], El modelo de las redes neuronales artificiales fue inspirado en el sistema nervioso y el comportamiento biológico del cerebro humano. En este proceso, una neurona recibe, a través de las dendritas, impulsos nerviosos procesados por el cuerpo y luego transmitidos a otras neuronas, todo esto en un conjunto inmenso de conexiones e información.

Un criterio similar describe [30], el cual menciona que las RNA están inspiradas en las redes neuronales biológicas del cerebro humano, es decir una RNA está

constituida por elementos que se comportan de forma similar a la neurona biológica en sus funciones más comunes, simulando al cerebro humano. A partir de este concepto se puede describir a una red neuronal artificial, el cual por medio de un modelo matemático intenta imitar el comportamiento de una red neuronal biológica, como podemos observar en la Figura 3, en el cual, el axón que es la salida de la neurona se conecta con las dendritas que son las entradas por medio de una sinapsis, la cual varía en el proceso de aprendizaje.

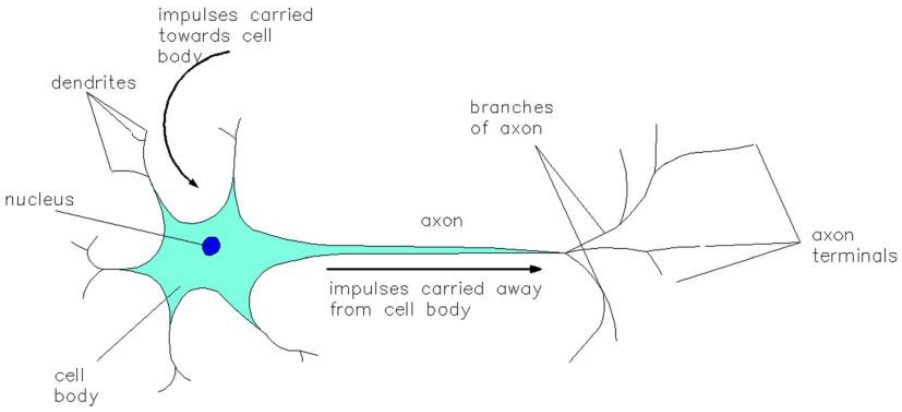


Figura 3: Estructura de una neurona biológica

Fuente: Tomado de [27]

2.2.2.2. Componentes de una Red Neuronal.

Son varios los componentes que interfieren en el modelo de una red neuronal artificial, pero esta generalmente está compuesta por tres partes, las cuales son: Entradas, núcleo, salidas. En el modelo de una RNA, [29] describe que cada neurona recibe entradas, que por medio de una función obtienen una salida producto de la suma ponderada de todas sus entradas, este, con la función de activación a cada entrada le correspondería un factor de ponderación que influye en el cálculo en su valor de salida, a este valor se le conoce como peso, siguiendo con el entrenamiento en el cual los pesos se ajustan para que la salida sea la esperada.

[31] Indica que las RNA se componen de nodos de entrada (también denominada como variable independiente o predictora) y salida (también denominado como variable dependiente o resultado), estos utilizan pesos de conexión, pesos de sesgo que son los parámetros de intercepción y multi-entropía para aprender o entrenar (en esta etapa se estiman parámetros) el modelo.

Con lo descrito anteriormente podemos resumir que, en una RNA, la entrada recibe los datos o parámetros que le permiten decidir a la neurona si estará activa o no, siguiendo por los pesos que representan la memoria de la red (son introducidos para simular la naturaleza aleatoria de la neurona biológica) y en el núcleo se encuentran todas las operaciones matemáticas necesarias para determinar la salida de la neurona.

2.2.2.3. Estructura de una Red Neuronal.

Según [32], las redes neuronales artificiales son muy populares en el campo de aprendizaje automático y a la vez son varias las estructuras que posee, una de ellas y la más utilizada es la perceptrón multicapa, la cual está completamente conectada, esta consta de una capa de entrada, una o más capas ocultas y una capa de salida, como podemos observar en la Figura 4.

[31] Argumenta, que una RNA puede tener capas simples o múltiples y que generalmente esta se divide en tres capas de neuronas como son: Entradas (reciben los datos), capas ocultas (extrae patrones y se encargan del procesamiento interno) y salidas (presentan resultados).

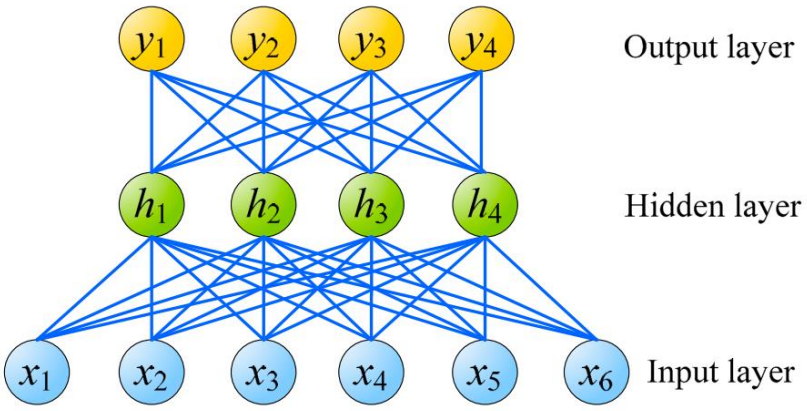


Figura 4: Estructura de una red neuronal multicapa (posee una capa oculta)

Fuente: Tomado de [32]

Una vez definida la estructura de la red neuronal, es necesario determinar su topología, en la cual se incluye: el número de capas, cantidad de neuronas por capas, grado de conectividad y el tipo de conexión entre neuronas, esto, para dar inicio al proceso de redes neuronales en el que se incluyen tres fases.

2.2.2.3.1. Fase de aprendizaje o entrenamiento.

En este proceso se va refinando iterativamente la solución hasta alcanzar un modelo lo suficientemente bueno. En esta fase, [33] argumenta que durante el aprendizaje o entrenamiento los pesos y términos independientes se actualizan para asegurar que la red se dirija en una aproximación óptima a la salida esperada por el usuario.

2.2.2.3.2. Fase de validación.

Para la ejecución de esta fase, los datos ingresados se dividen en datos de entrenamiento y datos de pruebas, mediante este proceso se evalúa el modelo generado en el proceso de entrenamiento ingresando datos nuevos para evaluar el rendimiento del modelo. Según [31], esta práctica es utilizada para minimizar la mala generalización del modelo. Mientras que, [33] afirma que este método es utilizado en proyectos donde se requiere estimar la precisión de un modelo que se llevará a cabo a la práctica.

2.2.2.3.3. Fase de inferencia o predicción.

Fase final del proceso de redes neuronales, en este se prueba el modelo para determinar si funciona correctamente o cumple con los objetivos planeados. Es decir, determinar si el modelo distingue las clases, basadas en el conjunto de datos de formación [31].

2.2.2.4. Series temporales.

[34] Define a las series temporales como una colección secuencial de observaciones que se registra en periodos de tiempo consecutivos.

2.2.2.4.1. Extracción de características.

El proceso de extracción de características reduce en gran medida el volumen de datos; este proceso se realiza debido a que no podemos ingresar todo el segmento a la red neuronal. Según [22], las técnicas de extracción de características mapean los datos en un conjunto en características, por otra parte, estas técnicas extraen las características en diferentes dominios como frecuencia, tiempo, tiempo-frecuencia, fractal y el espacio.

En resumen, esta etapa consiste en extraer las características más relevantes de un segmento para minimizar la complejidad de implementación, esto lo realizamos debido a que el hardware en el que se alojará el sistema posee bajos recursos a nivel computacional y a nivel de memoria. Se pueden extraer varias características, pero en este trabajo extraeremos las características en dominio del tiempo, como son: valor medio absoluto (MAV), valor eficaz (RMS) y longitud de forma de onda (WL).

Valor medio absoluto (MAV). Permite calcular el promedio de los valores absolutos de N muestras tomadas.

$$MAV = \frac{1}{N}\sum_{k=1}^{N}|p_k| \quad (1)$$

Valor eficaz (RMS). Similar a la anterior, es un valor escalar correspondiente a la media cuadrática.

$$RMS = \sqrt{\frac{1}{N}\sum_{k=1}^{N}{p_k}^2} \quad (2)$$

Longitud de forma de onda (WL). Es una variación acumulativa que puede indicar el grado de variaciones de las señales.

$$WL = \sum_{k=1}^{N}|p_k - p_{k-1}| \quad (3)$$

2.2.3. Reconocimiento de señas

2.2.3.1. Lengua de señas

Según [35], es una forma estructurada de gestos que implican movimientos y señas visuales utilizadas como sistema de comunicación, estos métodos permiten que personas de una comunidad (comúnmente personas con deficiencia auditiva y del habla) en específica interactúen entre sí; el mismo autor indica que estos métodos de comunicación implican el uso de diferentes partes

del cuerpo (dedos, manos, ojos, cabeza, expresiones faciales, entre otros) para transmitir el mensaje. Mientras que [1] lo describe como un lenguaje natural de expresión y configuración de gestos espaciales y percepción visual, con el cual, personas con discapacidad pueden comunicarse.

Según [1], existe un estándar internacional para el lenguaje de señas, pero existen variaciones para cada país en particular, es decir, de las 300 lenguas que signos que estiman la Federación Mundial de Sordos, en todos los países existe alguna variación en particular.

Un punto importante a mencionar, es que no existe una lengua de señas universal, es decir, este puede variar en diferentes regiones, para este trabajo se seleccionaron gestos del diccionario de lengua de señas ecuatorianas.

2.2.3.2. Lengua de señas ecuatorianas (LSEC)

El lenguaje de señas utilizado en el Ecuador se lo denomina Lenguaje de señas ecuatoriano (LSEC), aunque este es reconocido oficialmente por el gobierno, no lo es en diferentes comunidades, debido a que no existe un sistema de difusión para el mismo. Desde la integración en la constitución en 1998 y su cambio en el 2008, estos métodos de comunicación garantizan la integración de las personas con discapacidad a la sociedad [1].

El lenguaje de señas ecuatoriano es el medio de comunicación más utilizado por la comunidad sordomuda en el Ecuador [1].

2.2.3.2.1. Diccionario de lengua de señas

Es una herramienta con información cultural e ideológica de las personas sordas que contribuye al fortalecimiento y desarrollo de su lengua natural. Este diccionario fue desarrollado por la Federación Nacional de Personas Sordas del Ecuador (FENASEC) con el fin de desarrollar un diccionario oficial de lengua de señas ecuatorianas, en este se recopilaron todas las señas más usadas por la comunidad Sorda a nivel nacional. La FENASEC considera este proyecto fundamental para fortalecer esta lengua e integrar a esta comunidad en todos los ámbitos de la vida [36].

En el 2014 la CONADIS y la FENASEC presentaron el Diccionario Virtual De Lengua De Señas Ecuatorianas De Gabriel Román en el cual constan cerca de 5000 palabras extraídas del Diccionario Oficial de Lengua de Señas. Estas fueron de gran ayuda para el desarrollo del proyecto, debido al material ilustrativo de cada una de sus señas.

Según [1], esta herramienta ha demostrado ser una alternativa robusta y rápida para establecer una comunicación con personas con discapacidad auditiva y del habla y sin conocimiento previo de la LSEC.

2.2.3.2.2. Glosario básico de la lengua de señas ecuatorianas

Instrumento que permite mantener una comunicación con personas con discapacidad auditiva y del habla [37]. En el glosario se describen aspectos básicos para entender la lengua de señas. Según [38] la implementación de la lengua de signos básica ecuatoriana, el cual, fue basado en el diccionario estandarizado por CONADIS, es una herramienta que permite la comunicación y el aprendizaje de personas con discapacidad auditiva y del habla.

2.3. Objetivos del prototipo

2.3.1. Objetivo General

Desarrollar un sistema de reconocimiento de gestos mediante técnicas de inteligencia artificial y dispositivos IoT para la traducción de lengua de señas y automatización de dispositivos domóticos.

2.3.2. Objetivos Específicos

- Integrar técnicas de Inteligencia artificial utilizando algoritmos de Machine Learning como son las Redes Neuronales Artificiales.
- Instalar y configurar las herramientas domóticas para la integración y control de dispositivos domóticos comerciales.
- Desarrollar un prototipo utilizando dispositivos IoT para la implementación de la Red Neuronal Artificial.
- Realizar pruebas al modelo generado en el proceso de Redes Neuronales utilizando métricas de evaluación para medir su funcionamiento.

2.4. Diseño del prototipo

En el presente capítulo se describe el diseño del prototipo para llevar a cabo el proyecto para la ejecución de actividades basadas en el reconocimiento de gestos mediante señales obtenidas de un sensor y procesadas en una red neuronal, entre estas fases necesarias para el diseño del prototipo tenemos:

Diseño de Hardware. En el presente se realiza el diseño de los módulos para la ejecución de algoritmos**,** en los cuales es necesario una serie de módulos tanto en el proceso de redes neuronales (paquetes, librerías, entre otros.) como en el proceso de inferencia, en el cual se determinan las salidas a presentar cada vez que se detecta un gesto previamente entrenado. La principal salida para todos los gestos entrenados es la traducción a voz, la cual, una vez reconocido el movimiento, este, por medio de una librería reproduce el sonido perteneciente a dicho gesto. Lo mismo sucede de manera impresa, pero esta vez mostrando el significado en un display.

Otra salida interesante para estos gestos es la automatización de dispositivos domóticos comerciales que se encuentren integrados en la plataforma Home Assistant, es decir, el usuario realiza un gesto y activa o desactiva uno de estos dispositivos, entre los que utilizaremos están un bombillo y un enchufe (ambos desarrollados por la empresa Nexxt Solutions).

Implementación de las Redes Neuronales Artificiales. Este proceso es el más extenso, debido a que se deben estudiar varios conceptos relacionados a las redes neuronales con el fin de comprender y aplicar cada uno de ellos, además de que el mismo conlleva varias fases, comenzando desde la adquisición y procesamiento de datos, desde el cual se realizan las conexiones necesarias entre el microcontrolador y el sensor MPU6050 para obtener señales provenientes del acelerómetro y del giroscopio para su manipulación en Python, seguido por la generación de un conjunto de datos, el entrenamiento, hasta concluir con un modelo que sea capaz de satisfacer las necesidades del sistema.

Diseño del Software y prototipo final. Una vez culminado el proceso de entrenamiento se procede a definir un prototipo final con el resultado de cada una de las actividades realizadas anteriormente. En este último proceso se

determinan la salida de las clases (gesto inferido) obtenidas de la red neuronal, para ello se seleccionan los componentes necesarios a incorporar en el microcontrolador y la codificación necesaria para su incorporación, además de la conexión con Home Assistant para la manipulación del bombillo y del enchufe a través de MQTT.

Antes de iniciar el proceso de desarrollo se debe tener a la mano ciertos recursos y la selección definitiva de gestos. Entre los requisitos de hardware y software más fundamentales tenemos los siguientes:

2.4.1. Herramientas de desarrollo

Hardware.

Tabla 2: Requisitos de Hardware

Hardware	Definición
Módulo ESP32.	Para este proyecto hacemos uso del módulo ESP-WROOM-32.
Módulo MPU6050.	Sensor con giroscopio y acelerómetro, fuente principal de datos.
Pantalla Oled I2c 128x64 0.96 pulgadas.	Mostrar información necesaria.
Módulo amplificador digital PAM8403 de 5v con control de volumen (potenciómetro).	Fundamental para la salida de audio, integra un control de volumen para una fácil manipulación.
Mini parlante de 8 Ohmios.	Utilizado para la reproducción de sonido.
Cables de conexión dupont.	Facilita la conexión entre el microcontrolador y los componentes.
Protoboard.	Esencial para pruebas del prototipo.
Botón pulsador.	Permite el ingreso de datos.
Resistencia de 10k	Trabaja en conjunto con el pulsador.
Bombillo Smart Wifi	Para este proyecto se utilizó un bombillo de la marca Nexxt Solution.
Enchufe Smart Wifi	Para este proyecto se utilizó un enchufe de la marca Nexxt Solution.
Guante de motocicleta.	Esencial para realizar los gestos a inferir, en este se introducirá el sensor MPU6050.

Fuente: Elaboración del Autor

Cabe mencionar que cada uno de estos componentes son esenciales para la traducción de gestos, sin mencionar los requisitos computacionales necesarios para la etapa de entrenamiento del modelo, para los cuales no son necesarios equipos de gran capacidad, debido a que el entrenamiento se lo realiza en Google Colab aprovechando los recursos de este, pero, existe una etapa previa al entrenamiento, necesaria para observar en tiempo real las señales provenientes del sensor y tomar decisiones sobre las señales a analizar. Este proceso requiere como requisitos mínimos los siguientes:

- Procesador Intel Core i5 o superior.
- Mínimo 8GB de RAM.
- Tarjeta de video NGVIDIA GeForce 940M (Opcional).

Estos requisitos son suficientes también para la codificación del firmware que será introducido en nuestro microcontrolador ESP32.

Software.

Los módulos de software necesarios para la ejecución de algoritmos se encuentran presentes en todas las fases de desarrollo mencionadas anteriormente, así mismo, en la Tabla 3 se describen los entornos de desarrollo utilizados en el proyecto.

Tabla 3: Requisitos de Software

Software	Definición
PlatformIO	Entorno de desarrollo para dispositivos IoT, este IDE se integra en Visual Studio Code y es el idóneo para la codificación del firmware.
Python	Es el lenguaje de programación seleccionado para el desarrollo de las redes neuronales, entre los IDEs que utilizaremos tenemos IDLE y Google Colab para la fase de entrenamiento.
Home Assistant	Plataforma domótica en la cual integraremos nuestros dispositivos domóticos comerciales para su manipulación. Su instalación es por medio de una Máquina virtual usando Oracle Virtual Box.

Fuente: Elaboración del Autor

2.4.2. Selección de gestos

El aprendizaje automático (Machine learning) nos permite utilizar redes neuronales para que un dispositivo (ESP32) reconozca patrones de movimiento (previamente entrenado) y realice una acción posterior a esta. En este proyecto se traducen movimientos obtenidos del diccionario oficial de lengua de señas ecuatorianas (LSEC). Estos gestos fueron estrictamente seleccionados teniendo en cuenta un nivel de complejidad bajo, debido a que el diccionario incluye movimientos en el que participan ambas manos, ojos, labios, lengua, entre otros, y el sistema solo recoge datos de un sensor presente en una sola mano (derecha). Es decir, los movimientos deben contener una mayor frecuencia en la mano derecha.

Por este motivo se optó por seleccionar 10 movimientos sencillos en los que se incluyen saludos, preguntas y estados. Entre estos gestos unos son representados por un solo movimiento, mientras que otros requieren de una serie de movimientos con diferentes significados para llegar a una pregunta. Entre los movimientos seleccionados para su traducción tenemos los siguientes:

Tabla 4: Selección de gestos

Código	Gesto, Lengua de señas	Significado
G1	Buenos días	Buenos días
G2	Buenas tardes	Buenas tardes
G3	Buenas noches	Buenas noches
G4	¿Cómo te sientes?	¿Cómo te sientes?
G5	Encender, Iluminar	Encender, Iluminar
G6	Activar, activo (a), conectar	Activar, activo (a), conectar
G7	¿Qué hora es?	¿Qué hora es?
G8	¿Cómo llego?, dirección	¿Cómo llego?, dirección
G9	Llamarse, nombre; Tú	¿Cómo te llamas?
G10	Años; Cuántos; Tú	¿Cuántos años tienes?

Fuente: Elaboración del Autor

El entrenamiento de estos movimientos se los realizó con un sensor MPU6050 ubicado en la mano derecha, esto quiere decir que solo obtendrá y aprenderá movimientos realizados con esta mano. Por ende, la inferencia solo funcionará con valores obtenidos de una mano y estos a la vez realizados de manera

dinámica, es decir, cada uno tiene un patrón de movimiento único que lo diferencia del otro.

El sistema no reconoce movimientos estáticos como letras o números, debido a los recursos limitados del sensor, para esto se necesitaría sensores de flexión, los cuales no se incluyen en el prototipo. A continuación, se mencionan a detalle cada gesto seleccionado:

G1: Buenos días.

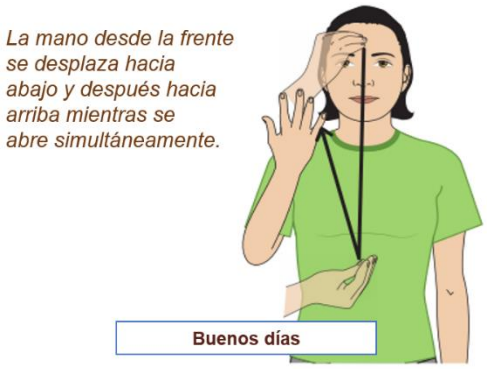


Figura 5: G1: Buenos días

Fuente: Tomado de [36]

G2: Buenas tardes.

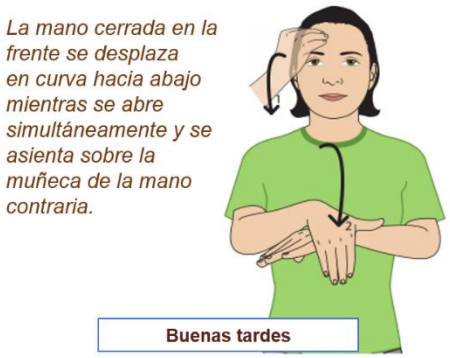


Figura 6: G2: Buenas tardes

Fuente: Tomado de [36]

G3: Buenas noches.

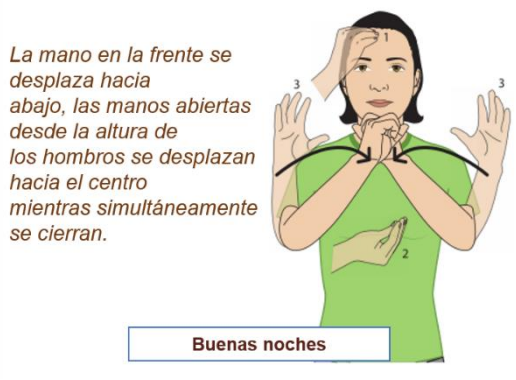


Figura 7: G3: Buenas noches

Fuente: Tomado de [36]

G4: ¿Cómo te sientes?

Figura 8: G4: ¿Cómo te sientes?

Fuente: Tomado de [36]

G5: Encender, Iluminar.

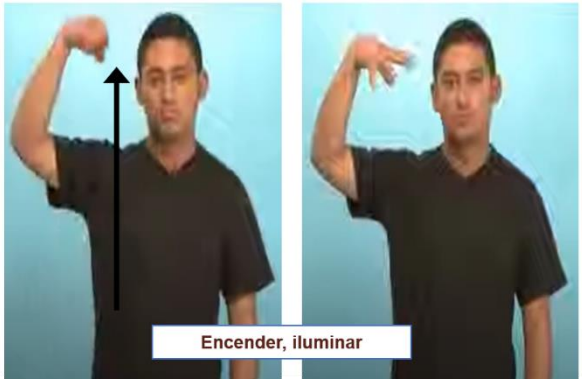


Figura 9: G5: Encender, iluminar

Fuente: Tomado de [36]

G6: Activar, activo (a), conectar.

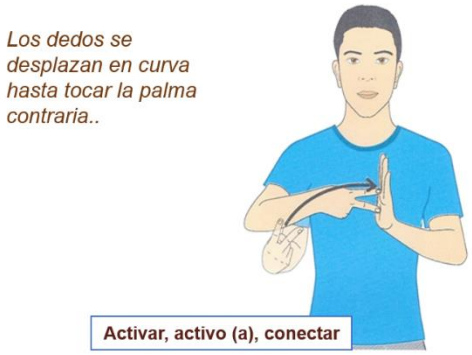


Figura 10: G6: Activar, activo (a), conectar

Fuente: Tomado de [36]

G7: ¿Qué hora es?

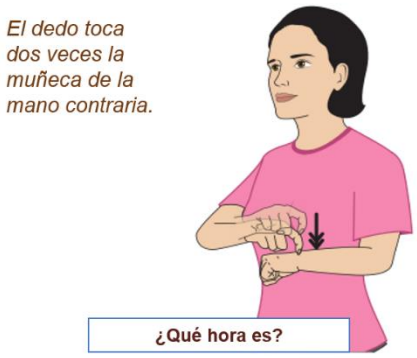


Figura 11: G7: ¿Qué hora es?

Fuente: Tomado de [36]

G8: ¿Cómo llego?, dirección.

Figura 12: G8: ¿Cómo llego?, dirección

Fuente: Tomado de [36]

G9: ¿Cómo te llamas?

Figura 13: G9: ¿Cómo te llamas?

Fuente: Tomado de [36]

G10: ¿Cuántos años tienes?

Figura 14: G10: ¿Cuántos años tienes?

Fuente: Tomado de [36]

Cada uno de estos gestos poseen patrones de movimiento diferentes a excepción de los gestos G1, G2 y G3 los cuales poseen movimientos similares diferenciándose únicamente del último paso. Pero su selección fue fundamental debido a que esta es una frase muy común y en su mayoría este se lo realiza con la mano derecha. Los gestos G5 y G6 fueron los indicados para la comunicación con la plataforma Home Assistant ya que estos nos permitirán controlar los dispositivos domóticos integrados en esta plataforma.

El G5 nos servirá para encender y apagar el bombillo, es decir, el sistema reconoce el movimiento y envía una publicación al bróker (Mosquitto) de MQTT indicando el cambio de estado de On a Off o viceversa. Lo mismo sucede con G6, pero este permitirá activar o desactivar el enchufe que de igual manera se

encuentra integrado en Home Assistant. Con esto realizamos la manipulación a dispositivos domóticos por medio de gestos utilizando técnicas de Inteligencia Artificial en dispositivos IoT.

2.4.3. Diseño de Hardware

En la Tabla 2 se describieron los recursos necesarios para armar el prototipo y a continuación se especifica la conexión de cada uno de ellos y la función que este realiza iniciando por la ubicación del sensor (fuente de datos).

2.4.3.1. Ubicación del sensor MPU6050

El sensor MPU6050 nos permite obtener una orientación espacial en términos angulares, también nos permite detectar fuerzas dinámicas como vibraciones o movimientos, por tal motivo se optó por posicionarlo en el dorso de la mano, además de que los gestos en su mayoría son realizados con estos movimientos, mientras que para su posición se prefirió la derecha, aunque sin ningún problema este pudo haber funcionado perfectamente en la otra mano, pero los gestos seleccionados deben adaptarse a la mano con la que se lo realiza.

Para una mejor flexibilidad el sensor fue adherido a un Guante corto de motocicleta, esto para una mayor comodidad y un libre movimiento, tal y como se aprecia en la Figura 15.

Figura 15: Ubicación del sensor MPU6050 en el guante

Fuente: Elaboración del Autor

En la Figura 15 se puede observar que el sensor se encuentra conectado con un cable tipo bus, este tiene un tamaño de aproximadamente 130 cm garantizando un rango más amplio de espacio para efectuar los movimientos y una distancia suficiente con el microcontrolador que es el encargado de obtener y de procesar las señales que arroje el mismo. En cuanto a las conexiones entre el ESP32 y el MPU6050 estas se comunican por medio de I2c utilizando los pines SCL (Pin de señal de reloj) y SDA (Pin de señal de datos), además de los pines de VCC y GND los cuales corresponden a corriente y tierra como se observa en el siguiente diagrama, Figura 16, el cual fue realizado utilizando la herramienta Fritzing.

Figura 16: Esquema de conexión del sensor MPU6050 con el ESP32

Fuente: Elaboración del Autor

La conexión es muy sencilla debido a que solo se utilizan 4 de los 8 pines que presenta el MPU6050. Este apartado describe el prototipo final de esta comunicación entre este componente con nuestro microcontrolador, lo siguiente sería realizar la codificación necesaria para obtener los datos del mismo.

2.4.3.2. Pantalla Oled SSD1306

Otro de los módulos necesarios para integrar al prototipo es la conexión con un display Oled de 128x64, este nos permitirá mostrar información relevante con el fin de que exista una comunicación más amigable entre el sistema y el usuario. El Display que se está utilizando es un dual color de 128x64 pixeles, en los cuales los primeros 16 pixeles son amarillos y los 48 siguientes son azules, pantalla de 0.96 pulgadas, modelo SSD1306. Esta consume pocos recursos de energía,

pero hay que tener en cuenta que mientras más grande es la pantalla, más consumirá. Además de que posee 4 pines (SCL, SDA, VCC y GND).

La conexión es muy sencilla debido a que el display y el MPU6050 utilizan diferentes direcciones I2c y podemos conectarlo al mismo bus I2c, es decir, ambos comparten las mismas conexiones como se muestra en la Figura 17.

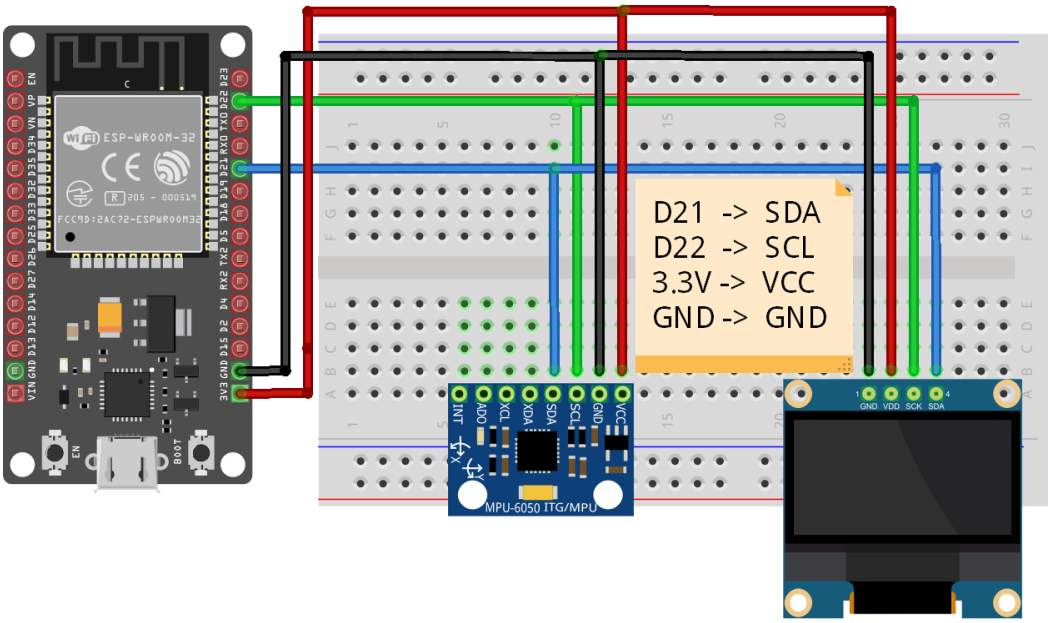


Figura 17: Esquema de conexión entre el display y el ESP32

Fuente: Elaboración del Autor

Nótese que en la Figura anterior los dos dispositivos, tanto el display como el sensor comparten los mismos pines del ESP32 sin ningún problema. Este apartado describe el prototipo final de esta comunicación entre este componente con nuestro microcontrolador, lo siguiente sería realizar la codificación necesaria para mostrar información en el display como texto con diferentes fuentes, imágenes de mapas de bits e incluso los datos del sensor en tiempo real.

2.4.3.3. Amplificador digital PAM8403

El proceso para obtener audio desde el dispositivo ESP32 es muy sencillo actualmente a diferencia de años anteriores, en los que obtener audio a través de estos dispositivos era casi imposible; Para este proceso hacemos uso del DAC (Convertidor analógico digital), el cual es una forma de reproducir sonidos utilizando equipos digitales, como podemos observar en el esquema representado en la Figura 18.

El ESP32 posee dos salidas DAC (en nuestro caso el ESP32-WROOM-32), el GPIO25 y el GPIO26. Un punto importante en este proceso es el sonido, el cual no es de una buena calidad, esto debido a que la resolución del dispositivo es de 256 y tiene un DAC de 8 bits (valores de 0 a 255). Entre los dispositivos que utilizaremos tenemos un PAM8403 y un par de parlantes pequeños de 8 Ohmios.

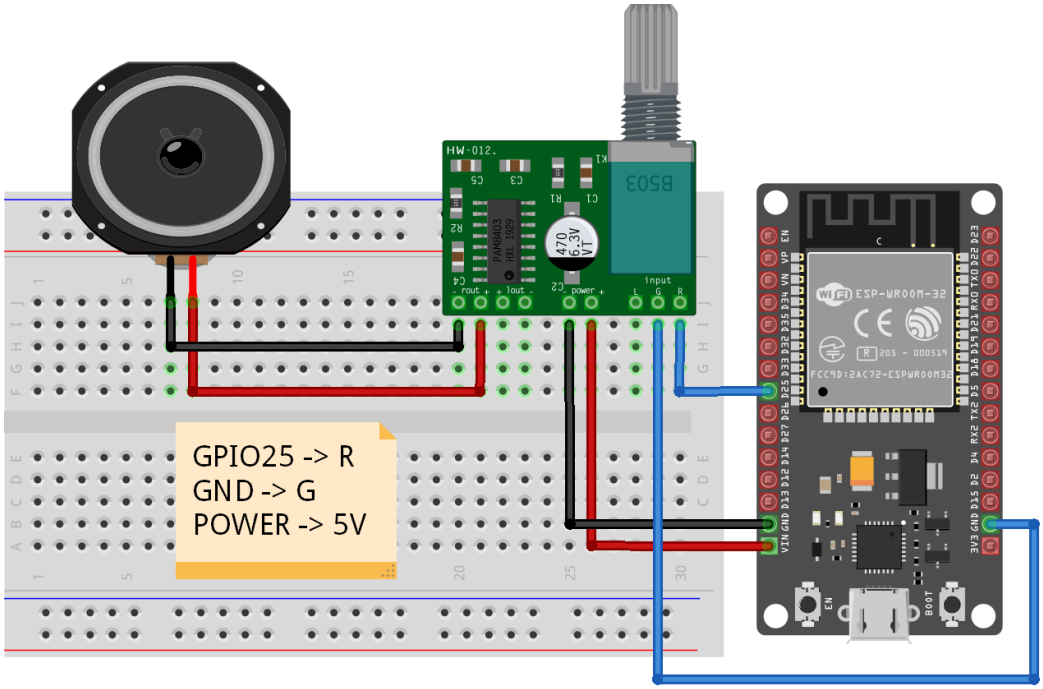


Figura 18: Esquema de conexión entre el amplificador digital PAM8403 y el ESP32.

Fuente: Elaboración del Autor

La conexión es muy sencilla esto debido a que solo utilizamos una salida DAC de nuestro ESP32 que es la GPIO25, ahora, nótese que este amplificador recibe corriente de nuestro ESP32 por medio de VIN, esto lo hace debido a que este necesita una conexión de 5V y VIN los provee.

2.4.4. Redes Neuronales Artificiales

Las redes neuronales no son más que modelos matemáticos capaces de reproducir el funcionamiento del cerebro humano, esto lo adquieren por medio de un gran proceso que inicia desde la adquisición de datos, su procesamiento, el entrenamiento y la extracción de características, las cuales son utilizadas en el dispositivo para su inferencia.

Este proceso requiere un grado muy alto de conocimiento en cálculo diferencial. Pero actualmente se lo puede realizar con cualquier lenguaje de programación,

además de que existen librerías capaces de llevar a cabo todo este proceso, una de ellas y la más utilizada es TensorFlow, la cual permite el desarrollo y entrenamiento de modelos de Aprendizaje Automático, esta cuenta con una serie de herramientas, bibliotecas y recursos provenientes de una gran comunidad muy activa. El proceso a seguir para el desarrollo de nuestra red neuronal es el descrito en siguiente diagrama, Figura 19.

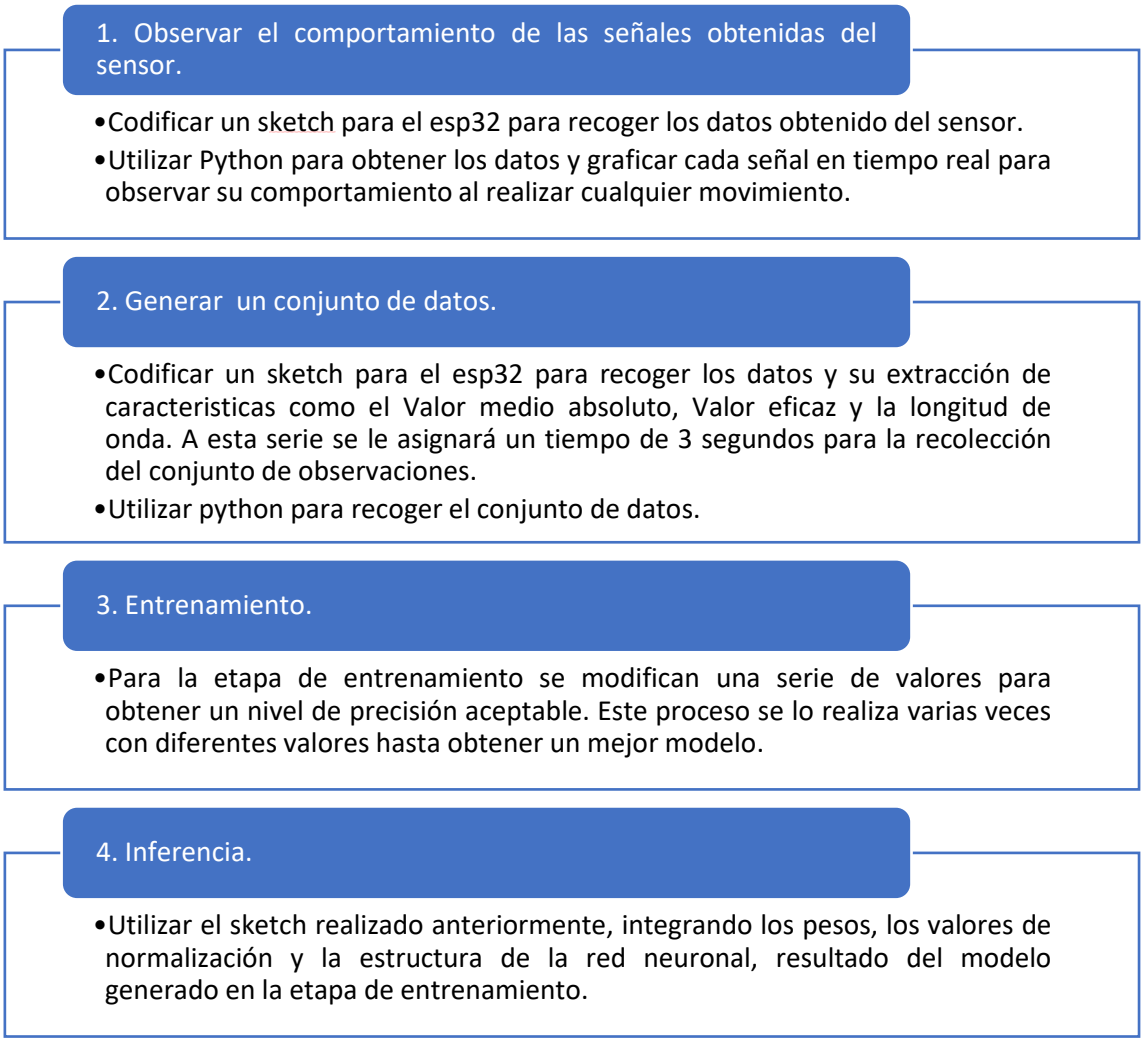


Figura 19: Flujo de desarrollo del proceso de redes neuronales

Fuente: Elaboración del Autor

2.4.5. Diseño del software y prototipo final

Una vez determinado el diseño de cada uno de los módulos se procede a integrarlos para posteriormente poseer un prototipo funcional que cumpla con las actividades programadas anteriormente. El prototipo final se lo realizó en un Protoboard, el cual facilitó las conexiones de cada uno de los componentes

concluyendo con el esquema de conexión definido en la Figura 20. Nótese que cada componente conectado al microcontrolador cumple una función. A este prototipo se le agregó un botón pulsador, el cual indicará al sensor el momento en el que debe comenzar a obtener el conjunto de observaciones para las cuales se estableció un tiempo de 3 segundos, es decir, el usuario tendrá 3 segundos una vez presionado el botón para comenzar a realizar un movimiento que represente a un gesto y posterior a este el sistema aplicará las respuestas necesarias para cada uno de ellos.

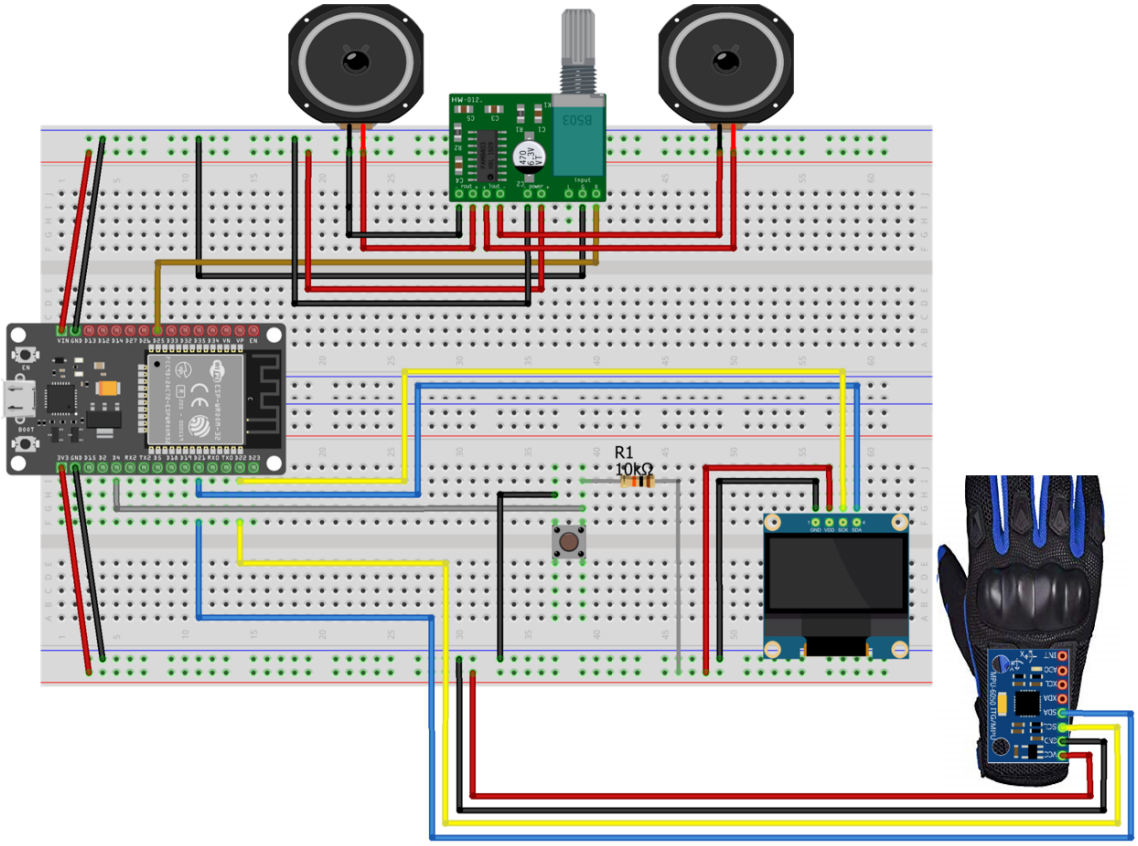


Figura 20: Esquema de conexión del prototipo final.

Fuente: Elaboración del Autor

Algo adicional en el prototipo es el uso de 2 parlantes, el cual, un segundo puede ser opcional, además del ingreso de una resistencia de 10 kΩ la cual es muy importante para el uso del botón o pulsador. Una vez entendido el funcionamiento de cada uno de los módulos que componen el hardware, es necesario explicar los módulos que componen el software en los que participan el firmware del dispositivo ESP32, MQTT y Home Assistant, los cuales mediante un diagrama de actividades (Figura 21) se representa la interacción entre estos.

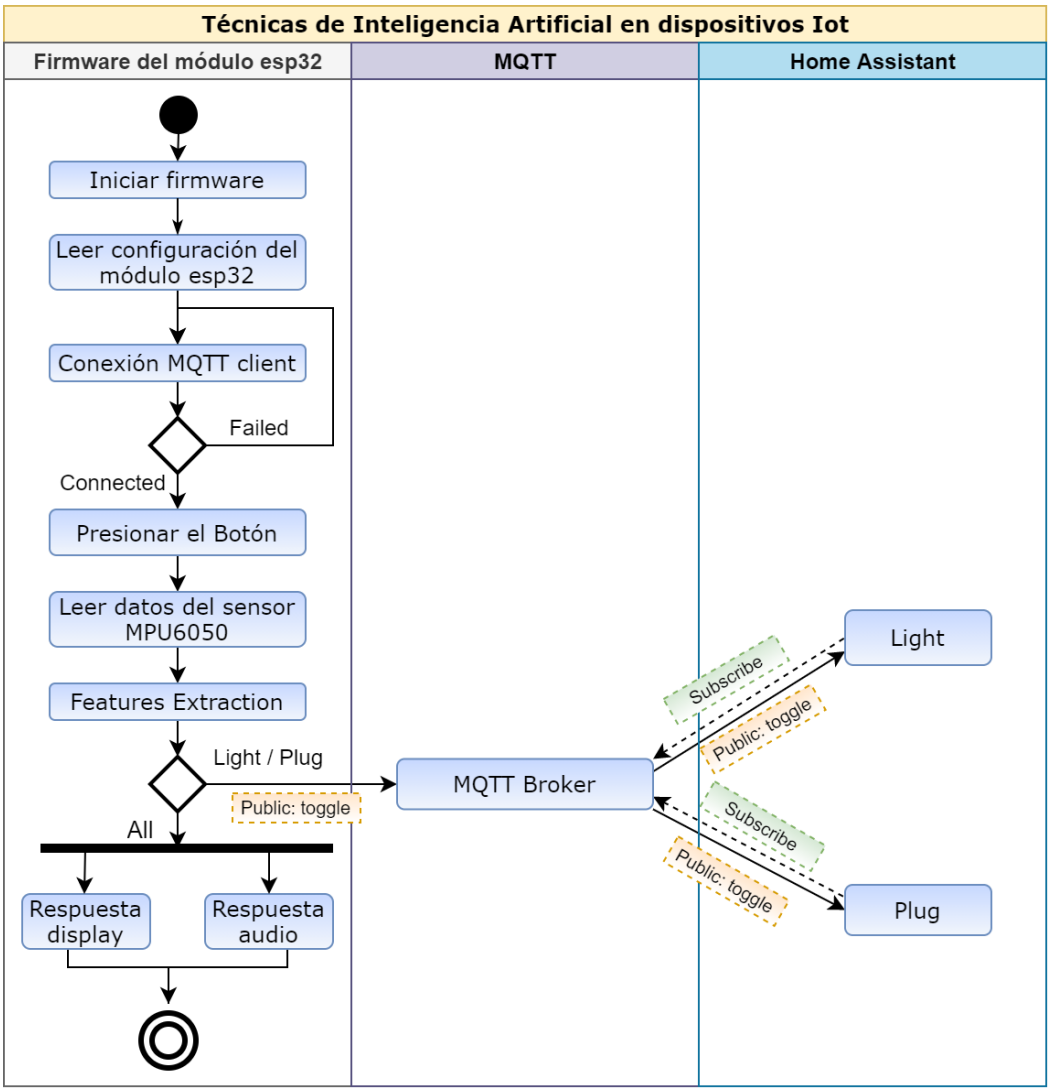


Figura 21: Diagrama de actividades del prototipo.

Fuente: Elaboración del Autor

En el diagrama de actividades expuesto se observa que el proceso inicia con las configuraciones del firmware, en las cuales tenemos la importación de bibliotecas, creación e inicialización de variables y la inicialización de funciones seguido por los métodos necesarios para la ejecución de algoritmos, entre los que se encuentra el método connection, encargado de verificar la conexión con MQTT y de las publicaciones. Una vez inicializado estos métodos el sistema estará listo para el reconocimiento de gestos, los cuales serán iniciados al presionar el botón. Para entender a detalle el proceso, a continuación, se lo representa por medio de un diagrama de flujo en la Figura 22.

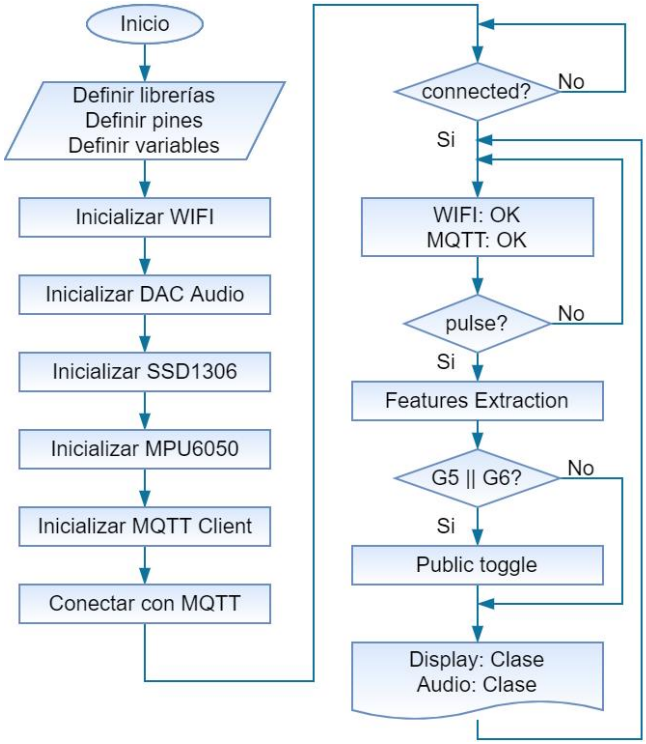


Figura 22: Diagrama de flujo del proceso realizado por el firmware

Fuente: Elaboración del Autor

2.5. Ejecución y/o ensamblaje del prototipo

2.5.1. Instalación y configuración de la pantalla Oled SSD1306

Para mostrar gráficos e información en la pantalla Oled necesitamos hacer uso de una librería, para ello existen varias alternativas, pero haremos uso de la librería Adafruit. Para su instalación nos dirigimos a Librerías y buscamos Adafruit SSD1306 y la instalamos. Como se había mencionado en la sección requisitos necesarios, el IDE en cual se realizará la codificación del firmware es PlatformIO, este se ejecuta en el editor de código fuente Visual Studio Code.

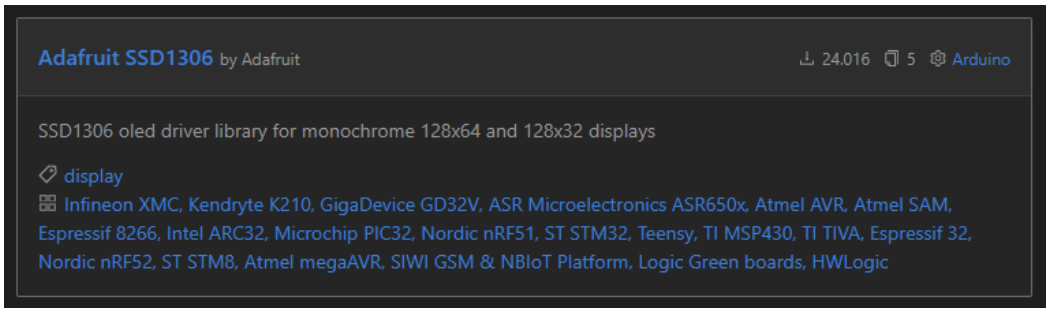


Figura 23: Instalación de la librería Adafruit SSD1306

Fuente: Elaboración del Autor

De igual manera se trabaja en conjunto con otras dos, estas son Adafruit GFX y Adafruit BusIO, las cuales de igual manera se las busca e instala.

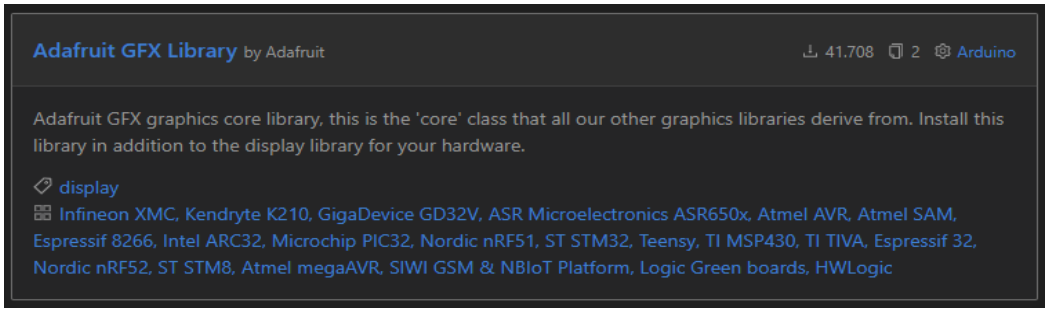


Figura 24: Instalación de la librería Adafruit GFX

Fuente: Elaboración del Autor

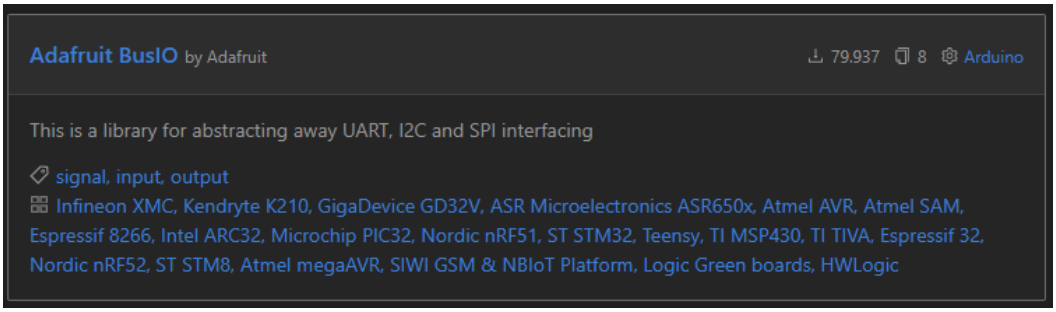


Figura 25: Instalación de la librería Adafruit BusIO

Fuente: Elaboración del Autor

Una vez instaladas, aparecerán en nuestro Platformio.ini. El proceso para mostrar información en el display es muy sencillo, solo se debe tener en cuenta la capacidad de información que se puede mostrar en el display (128 x 64). Lo primero que debemos hacer es importar las librerías en nuestro código, tanto para usar I2C como para escribir en pantalla.

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
```

Luego procedemos a inicializar el display y el objeto de visualización con el ancho y alto definidos anteriormente.

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
```

```
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
```

En el setup () inicializamos la pantalla OLED.

```
if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println("SSD1306 allocation failed");
    for(;;);
}
```

Eliminamos cualquier contenido presente en la pantalla (Borramos el búfer de pantalla) con el método:

```
display.clearDisplay();
```

Establecemos un tamaño de la fuente.

```
display.setTextSize(1);
```

Establecemos el color de fuente.

```
display.setTextColor(WHITE);
```

Establecemos la ubicación.

```
display.setCursor(20,26);
```

Escribimos el contenido a mostrar en el display.

```
display.println("WIFI...");
```

Por último, enviamos el contenido al display.

```
display.display();
```

Como podemos observar el proceso es muy sencillo, incluso la librería trae ejemplos de cómo obtener el máximo provecho al display. Los mismos pasos son para imprimir texto con otro tipo de fuente, solo debemos especificar la fuente e incluirlo en el boceto.

Imágenes de mapas de bits en el Oled.

Lo primero a realizar es seleccionar la imagen y establecerle un tamaño de 128 x 64 (tamaño del display) y guardamos la imagen como mapa de bits monocromático. Esto lo podemos tranquilamente realizar en Paint.

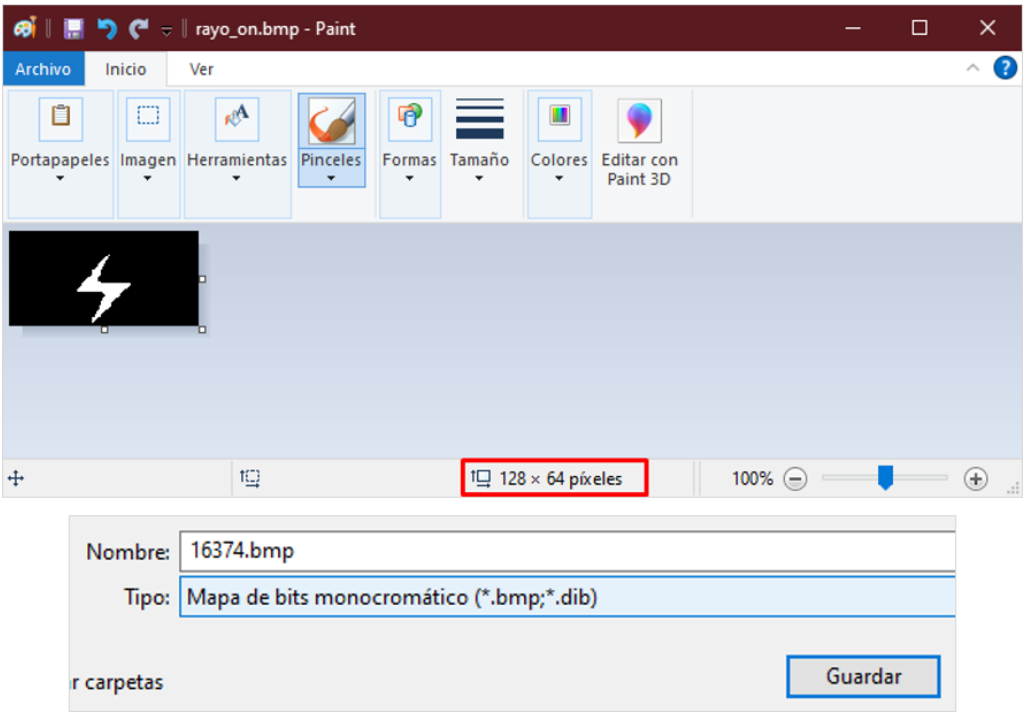


Figura 26: Cambiar el tamaño y formato de imagen en mapa de bits utilizando Paint.

Fuente: Elaboración del Autor

Una vez seleccionada la imagen, utilizamos el software LCD-Image-Converter [39] para modificarla y generar un archivo C.

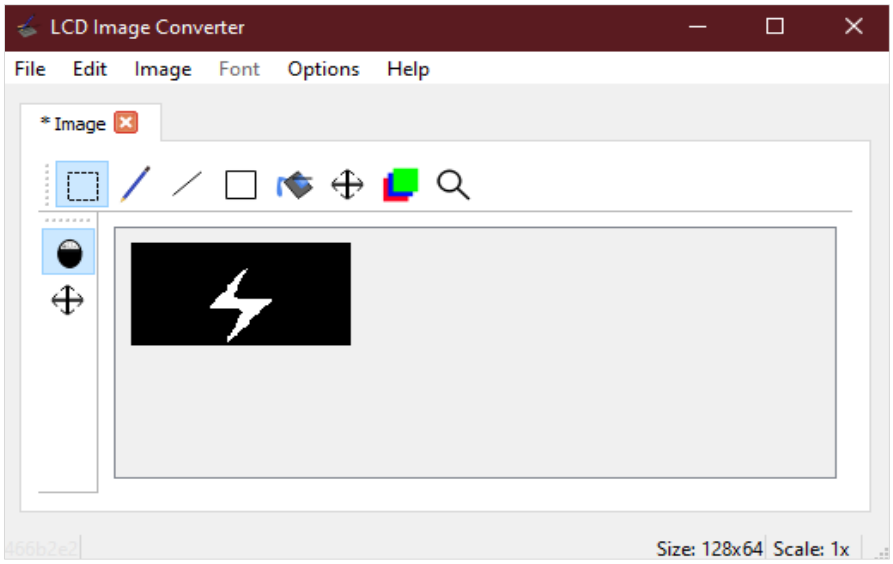


Figura 27: LCD-Image-Converter para generar un archivo C de una imagen

Fuente: Elaboración del Autor

En “opciones > Conversión” realizamos las siguientes configuraciones:

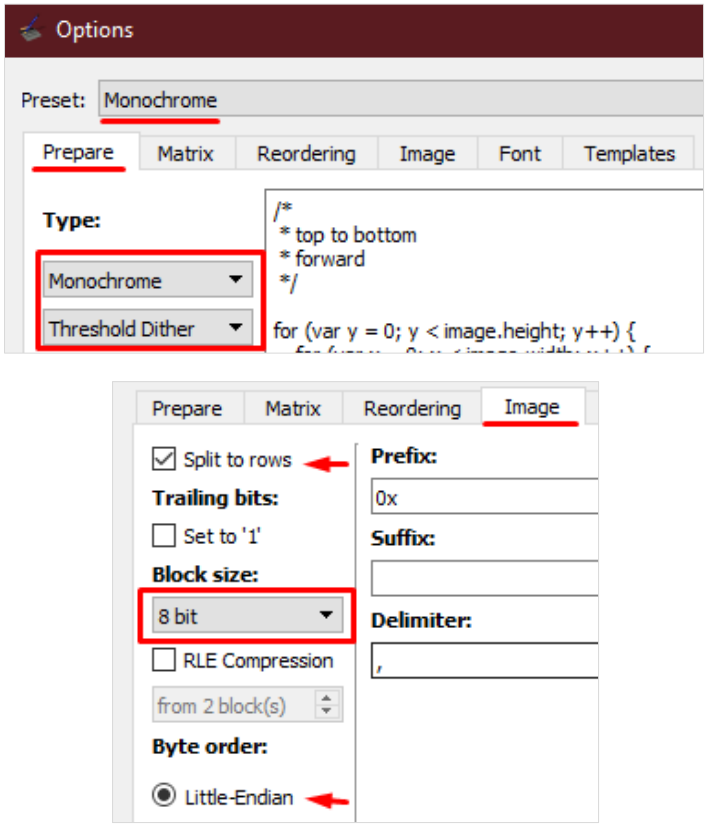


Figura 28: Configuraciones realizadas en lcd-image-converter

Fuente: Elaboración del Autor

Una vez realizado exportamos la imagen en formato C, esta nos generará un archivo que utilizaremos para mostrar en nuestro display.

Nos dirigimos a nuestro código y pegamos la matriz. Para una codificación más estructurada podemos tener en un archivo aparte todas las matrices pertenecientes a imágenes y en nuestro boceto podemos simplemente incluirlo.

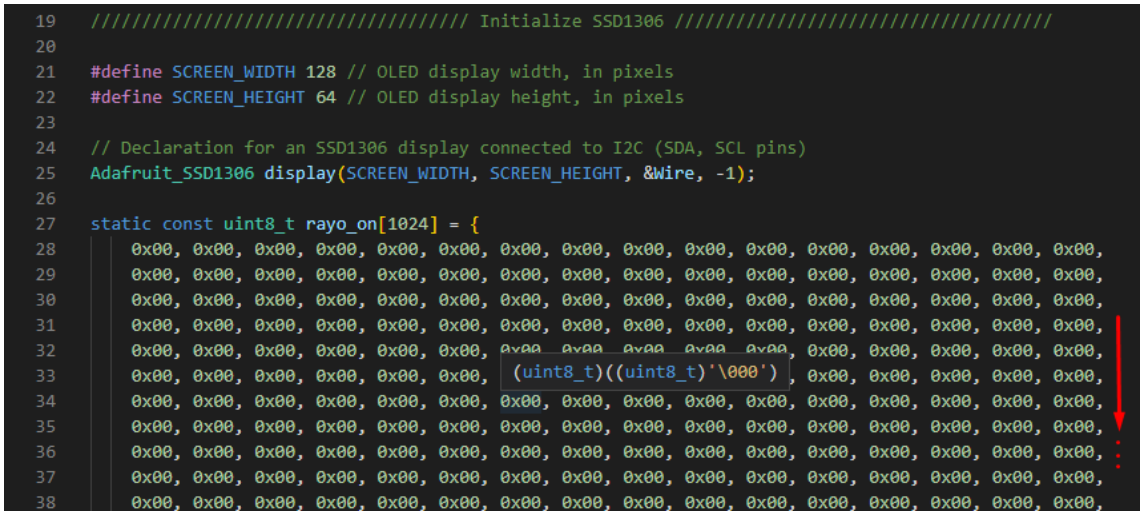


Figura 29: Matriz en C de la imagen

Fuente: Elaboración del Autor

En la sección que queramos, pegamos el siguiente código, el cual nos va a permitir mostrar la imagen en nuestro display. El drawBitmap será en encargado de mostrar la imagen, en este se debe primero especificar la ubicación (0,0), el nombre la matriz que generamos (rayo_on), el tamaño de la imagen (128,64) y el color (1) que por defecto es WHITE.

```
display.clearDisplay();
display.drawBitmap(0, 0, rayo_on, 128, 64, 1);
display.display();
```

Al flashear el código obtendremos el resultado de la Figura 30.

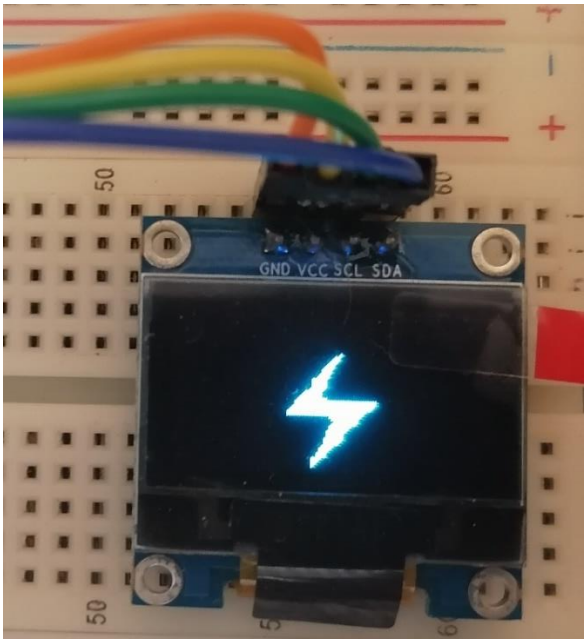


Figura 30: Imagen de mapa de bits en el display SSD1306

Fuente: Elaboración del Autor

2.5.2. Instalación y configuración del sensor MPU6050

El proceso para obtener las señales del sensor es muy sencillo, para esto necesitamos hacer uso de una librería, para ello existen varias alternativas, pero para este proyecto hacemos uso de la librería i2cdevlib de Jeff Rowberg. Para su instalación nos dirigimos a “Librerías” y buscamos “MPU6050”, la seleccionamos e instalamos.

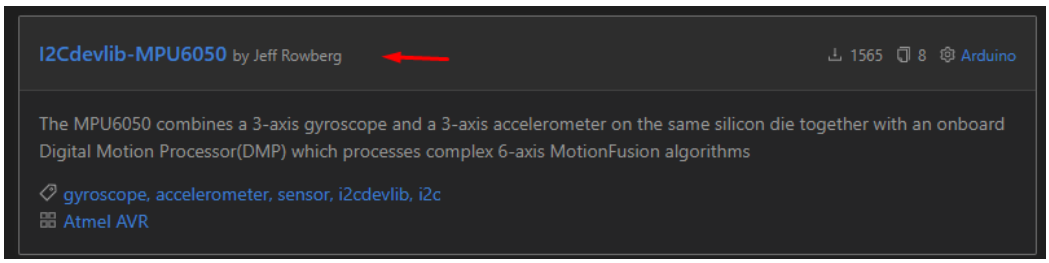


Figura 31: Instalación de la librería 12Cdevlib - MPU6050

Fuente: Elaboración del Autor

Una vez instaladas, aparecerán en nuestro Platformio.ini. El proceso para mostrar las señales tanto del giroscopio como del acelerómetro es muy sencillo, lo primero que debemos hacer es importar las librerías en nuestro código.

```
#include "I2Cdev.h"
#include "MPU6050.h"
```

Procedemos a inicializar el sensor y las variables que este arroja, las cuales son 6, 3 del giroscopio y 3 del acelerómetro, además del tiempo de muestreo.

```
MPU6050 sensor;
int16_t accelX = 0, accelY =0, accelZ;
int16_t gyroX = 0, gyroY =0, gyroZ;
```

En el setup () inicializamos el sensor.

```
// join I2C bus (I2Cdev library doesn't do this automatically)
Wire.begin();           //Iniciando I2C

// initialize device
Serial.println("Initializing I2C devices...");
sensor.initialize();    //Iniciando el sensor

// verify connection
if (sensor.testConnection()) Serial.println("Sensor iniciado");
else delay(500); Serial.println("Error al iniciar el sensor");
```

Por último, en el lazo procedemos a obtener los valores del sensor y a normalizarlos (escalar sus lecturas) para finalmente imprimirlos por consola.

```
// Leer las aceleraciones y velocidades angulares
sensor.getAcceleration(&accelX, &accelY, &accelZ);
sensor.getRotation(&gyroX, &gyroY, &gyroZ);

float ax_m_s2 = accelX * (9.81/16384.0);
float ay_m_s2 = accelY * (9.81/16384.0);
float az_m_s2 = accelZ * (9.81/16384.0);
float gx_deg_s = gyroX * (250.0/32768.0);
float gy_deg_s = gyroY * (250.0/32768.0);
float gz_deg_s = gyroZ * (250.0/32768.0);

Serial.println(ax_m_s2);
Serial.println(ay_m_s2);
Serial.println(az_m_s2);
Serial.println(gx_deg_s);
Serial.println(gy_deg_s);
Serial.println(gz_deg_s);
```

Al ejecutar obtendremos los datos del sensor de manera inmediata conforme al movimiento del mismo, también podemos hacer uso del display como se observa en la Figura 32 para imprimir los valores del sensor.

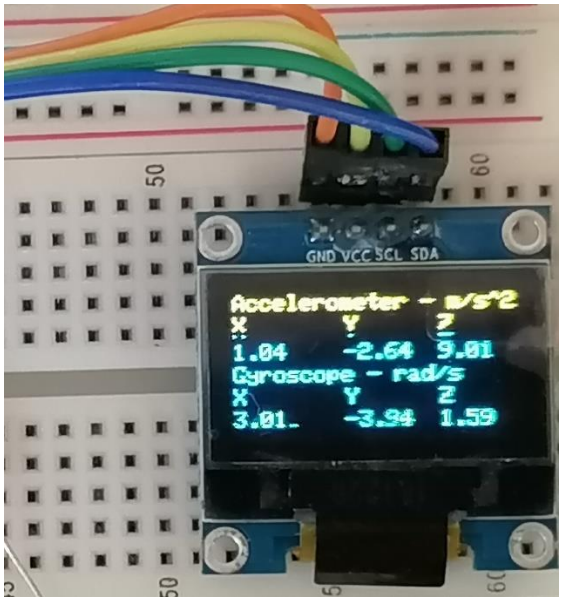


Figura 32: Señales del sensor MPU6050 en el display SSD1306

Fuente: Elaboración del Autor

2.5.3. Instalación y configuración del amplificador digital PAM8403

Obtener audio es muy sencillo, para hacer uso de esta salida se deben seguir una serie de procesos teniendo como principal el uso de una librería. Para este proceso hacemos uso del amplificador PAM8403, aunque otra opción es el amplificador LM386 cuyas conexiones son muy similares.

Utilizamos la librería XT DAC Audio 4.2.1, la cual permite la reproducción de archivos WAV. Esta librería es completamente compatible con el módulo ESP32. Su descarga se la realiza desde su página oficial y posteriormente se agrega los archivos principales en la sección Lib del proyecto, esto debido a que esta librería no se encuentra en el repositorio de PlatformIO.

Posterior a eso simplemente debemos declararla, esta librería trae consigo varios ejemplos de cómo utilizarla, podemos guiarnos de uno de ellos una vez tengamos implementado nuestro prototipo anteriormente mencionado.

Conversión de texto a voz.

La librería mencionada anteriormente permite reproducir archivos WAV, pero, debido al tamaño del archivo este audio debe ser previamente modificado y transformado en lenguaje C. Lo primero a realizar es la selección del audio, como nuestro convertidor de texto a voz utilizaremos Notevibes [40], el cual es una herramienta online muy sencilla de utilizar, solo debemos ingresar a notevibes.com, crear una cuenta y comenzar a escribir el texto a transformar.

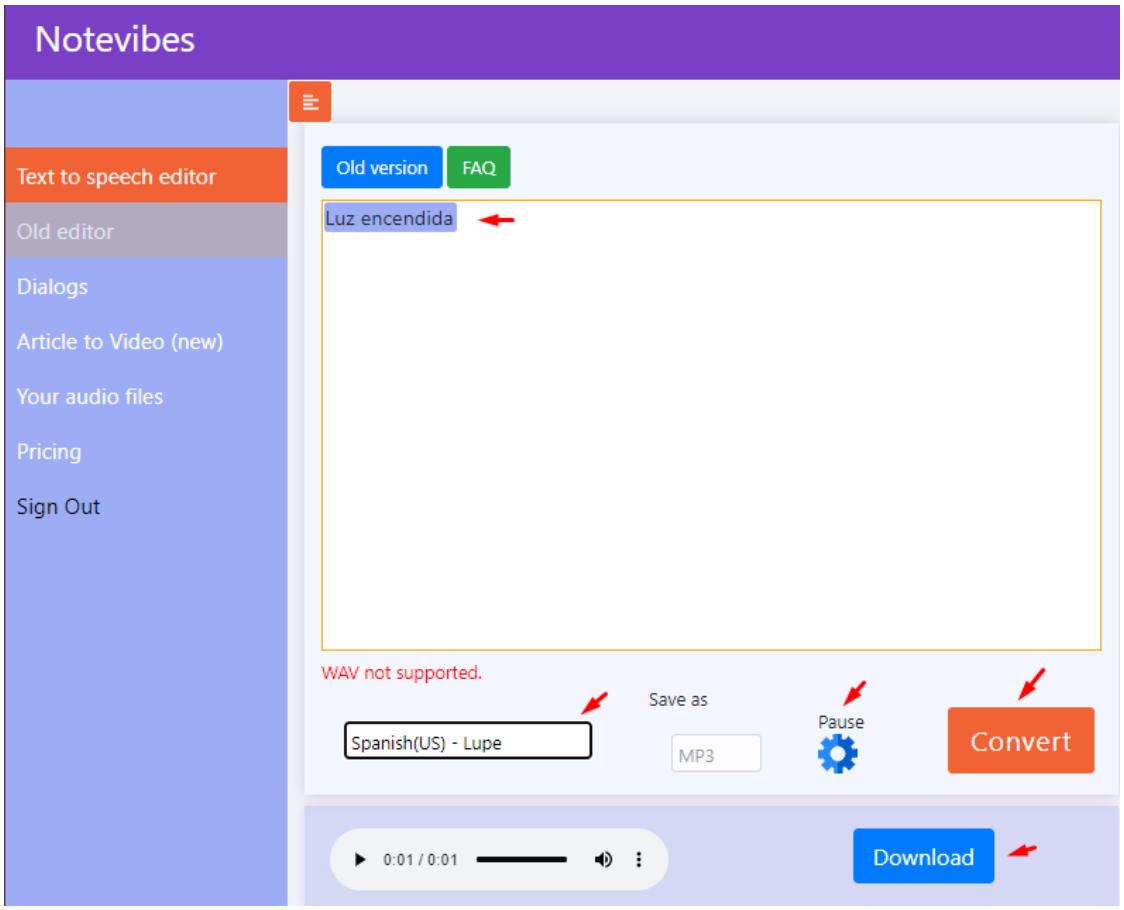


Figura 33: Notevibes para transformar de texto a voz

Fuente: Elaboración del Autor

Entre las frases a transformar a voz seleccionamos los gestos mencionados anteriormente.

Configuración de la frecuencia (Hz) del audio

Una vez seleccionado el audio utilizamos el software Audacity para su transformación. Para ellos seleccionamos el archivo y lo subimos a Audacity.

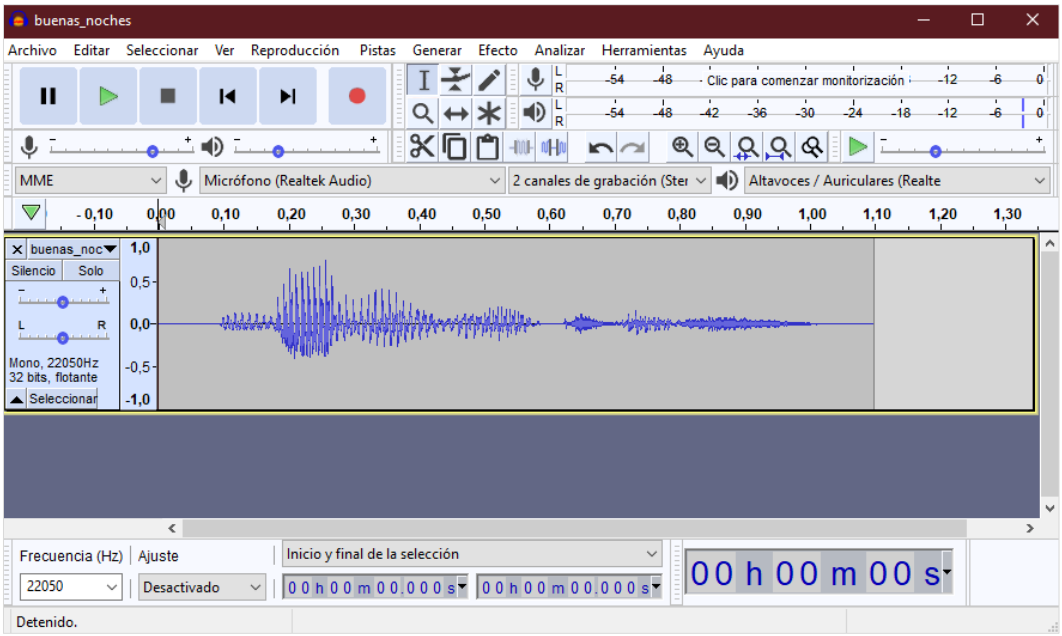


Figura 34: Audacity para bajar la frecuencia del audio y exportarlo a formato wav

Fuente: Elaboración del Autor

Disminuimos la frecuencia Hz, esta puede estar entre 8000 a 16000, puede ser mayor, pero este archivo requerirá más espacio, por esto lo recomendable sería dejarlo en 16000 u 8000, en nuestro caso le dejaremos en 16000.

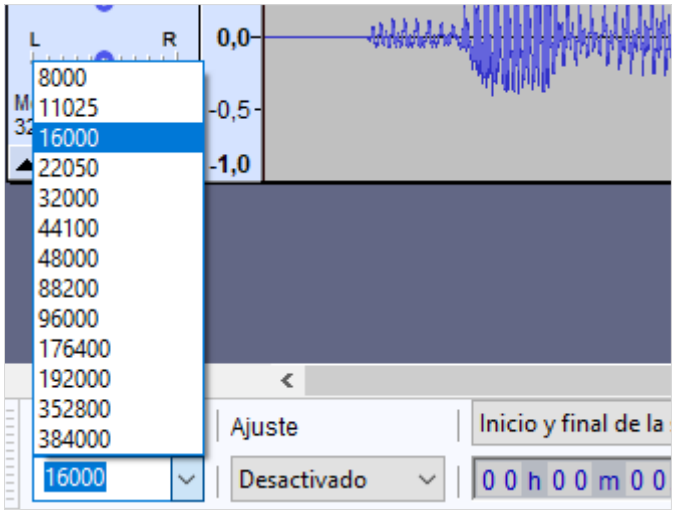


Figura 35: Disminuir la frecuencia del audio a 16000 Hz

Fuente: Elaboración del Autor

Luego nos dirigimos a la pestaña “Pista > Mezclar > Mezclar y generar” seleccionamos la opción y observaremos que el audio cambió.

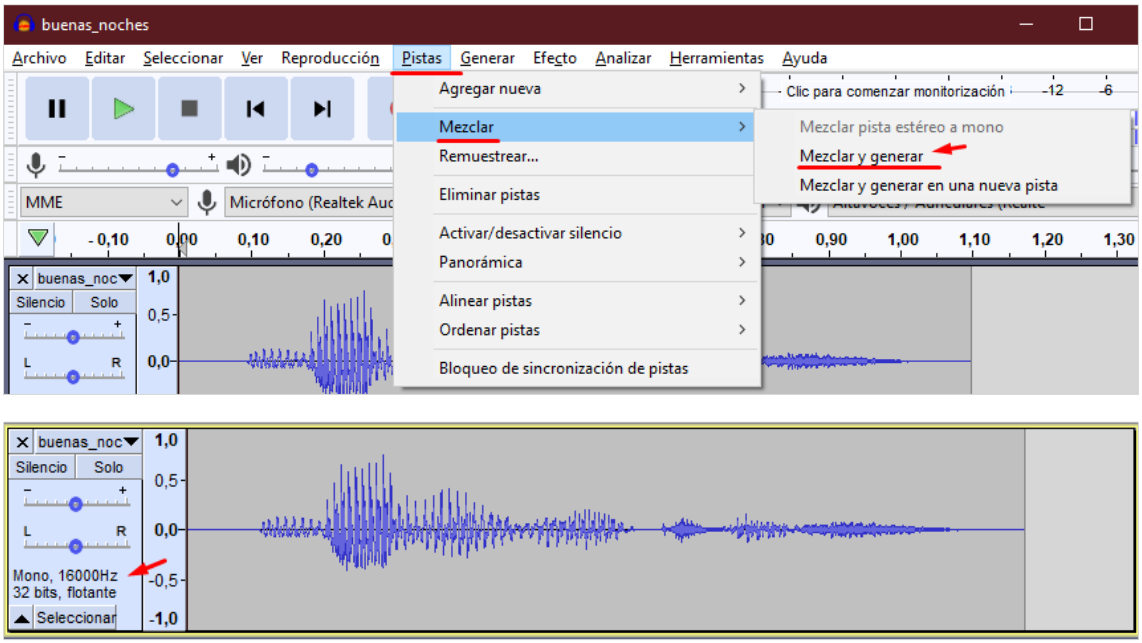


Figura 36: Mezclar y generar el audio a 16000 Hz

Fuente: Elaboración del Autor

Una vez realizado exportamos el audio a formato wav., luego al guardar el audio debemos tener en cuenta la extensión en la que se lo guarda, esta debe ser de la siguiente manera.

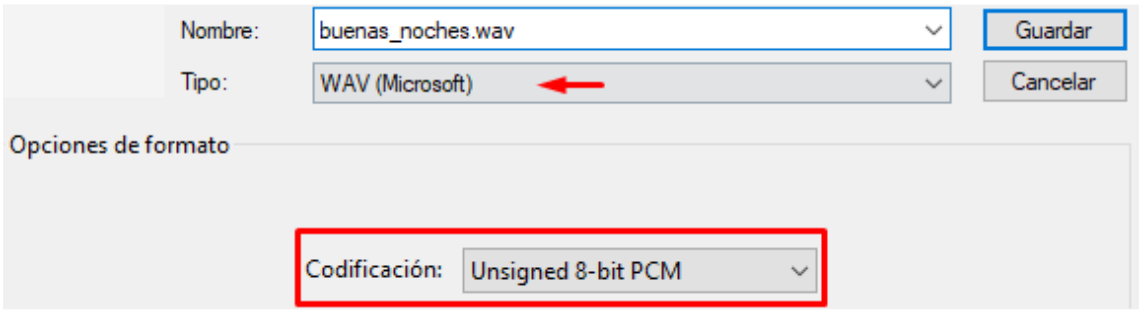


Figura 37: Extensión y codificación del audio a exportar

Fuente: Elaboración del Autor

Transformar audio en formato C.

Una vez con el audio en formato WAV utilizamos la herramienta HxD [41] para transformarlo a lenguaje C; para esto abrimos el archivo .wav con el software y observaremos que este se ha transformado.

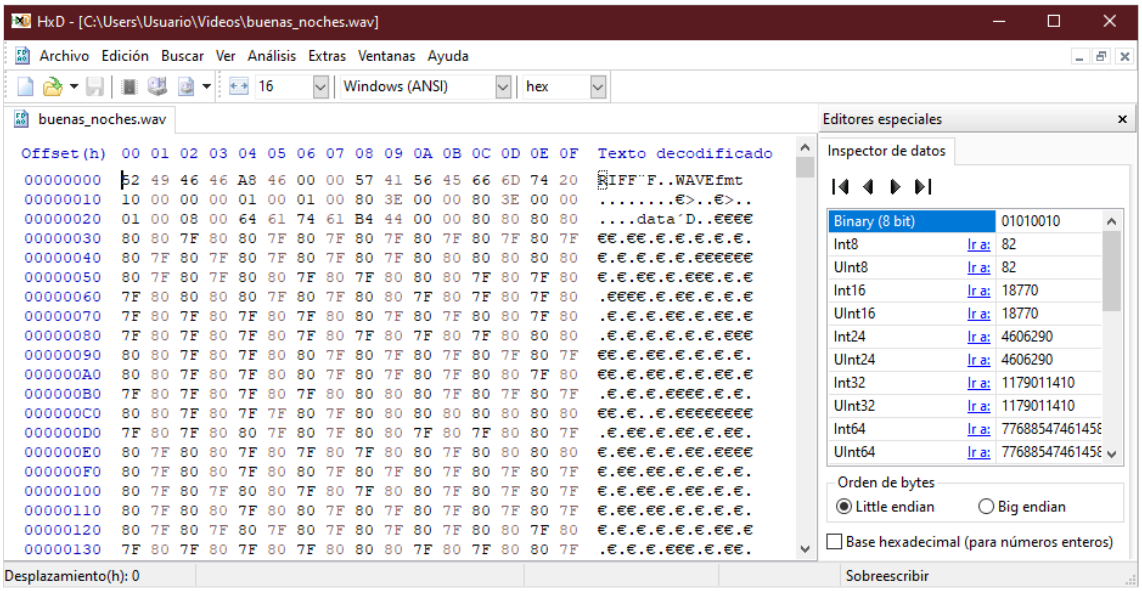


Figura 38: Herramienta HxD para transformar a matriz en C

Fuente: Elaboración del Autor

Ahora solo tendríamos que exportarlo a lenguaje C, para eso nos dirigimos a Archivo > Exportar > C.

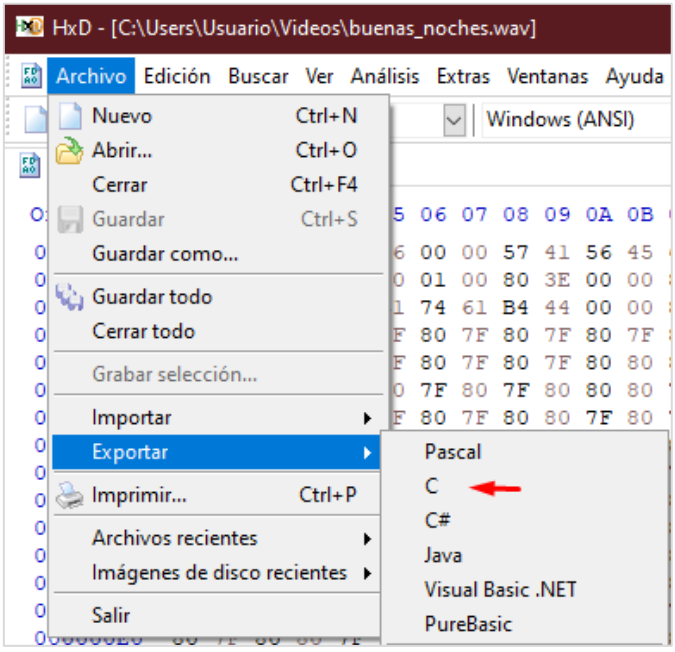


Figura 39: Exportar la matriz a lenguaje C

Fuente: Elaboración del Autor

Codificación.

Para una codificación mejor estructurada se crea un archivo aparte en el que colocaríamos nuestra matriz en C del audio y la integramos a nuestro boceto.

Como podemos observar en la Figura 42, en la que tenemos un archivo con el nombre SoundData.h con la matriz generada anteriormente.

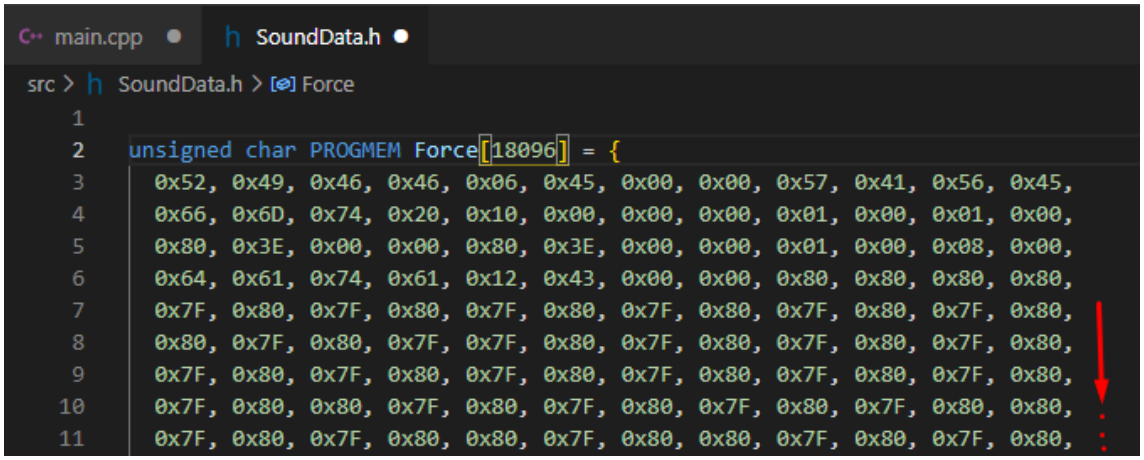


Figura 40: Integración de la matriz al código

Fuente: Elaboración del Autor

La librería nos facilita la codificación, para que ésta reproduzca el audio en solo un par de líneas de código, como es el siguiente. Primero incluimos la librería y el archivo de audio

```
#include "SoundData.h"
#include "XT_DAC_Audio.h"
```

Luego inicializamos en un objeto el archivo de audio.

```
XT_Wav_Class ForceWithYou(Force);
```

Seleccionamos el pin de salida DAC.

```
XT_DAC_Audio_Class DacAudio(25,0);
```

En el lazo llenamos el búfer de sonido con datos y lo reproducimos.

```
DacAudio.FillBuffer();
  if(ForceWithYou.Playing==true)
    DacAudio.Play(&ForceWithYou);
```

2.5.4. Instalación y configuración de Home Assistant

La plataforma domótica Home Assistant se utilizará para la automatización de dispositivos comerciales como lo son un Bombillo y un enchufe, los cuales por medio de MQTT cambiará el estado de los mismos a través de publicaciones y suscripciones. Es decir, Hassio nos servirá como servidor en el cual tendremos alojados nuestros dispositivos a manipular, cuyo proceso se puede observar en la Figura 41.

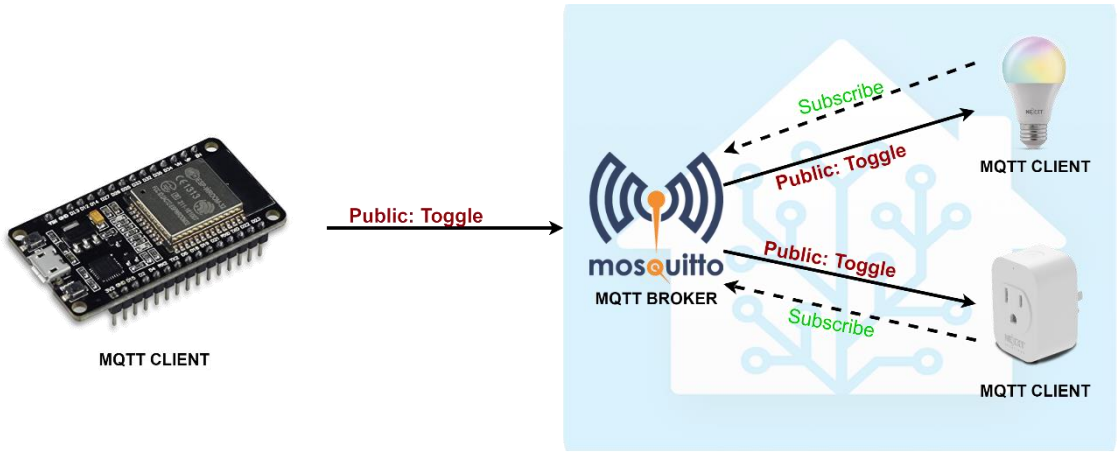


Figura 41: Comunicación entre ESP32 y los dispositivos integrados en Hassio

Fuente: Elaboración del Autor

2.5.4.1. Instalación de Home Assistant en una máquina virtual

La instalación de la plataforma Home Assistant se puede realizar de varias maneras, para este proyecto haremos uso de una máquina virtual utilizando Oracle VirtualBox como herramienta de virtualización. Para ello nos dirigimos al sitio oficial de Home Assistant y la descargamos como dispositivo virtual (x86_64/UEFI) seleccionando VDI (Virtual Desktop Infrastructure).

Una vez descargado ingresamos a VirtualBox y creamos una máquina virtual con las siguientes configuraciones.

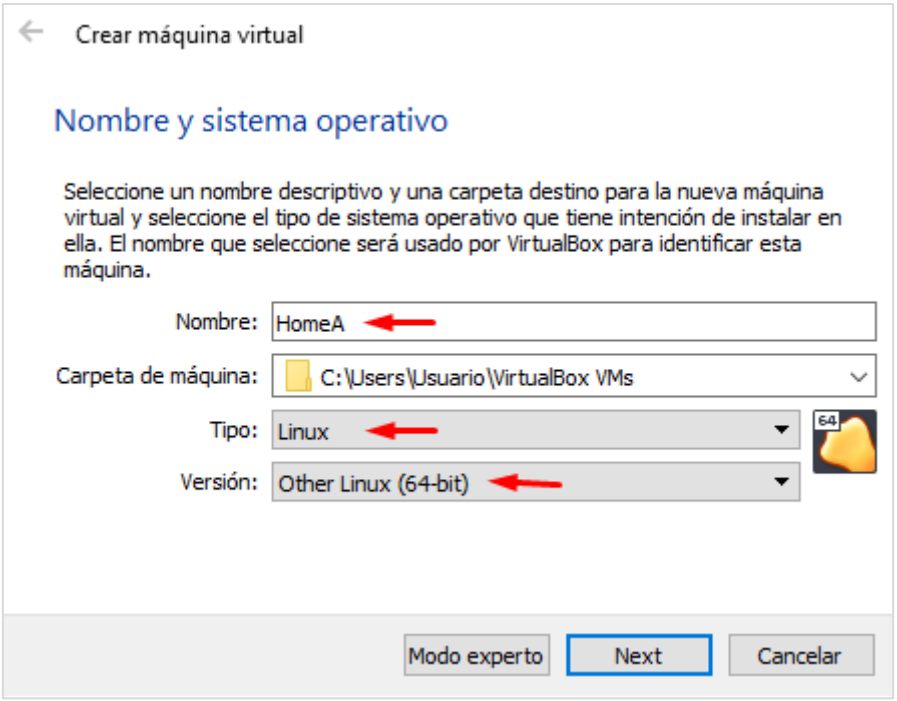


Figura 42: Nombre y sistema operativo de la máquina virtual

Fuente: Elaboración del Autor

Asignamos 2048 de memoria y damos clic en siguiente.

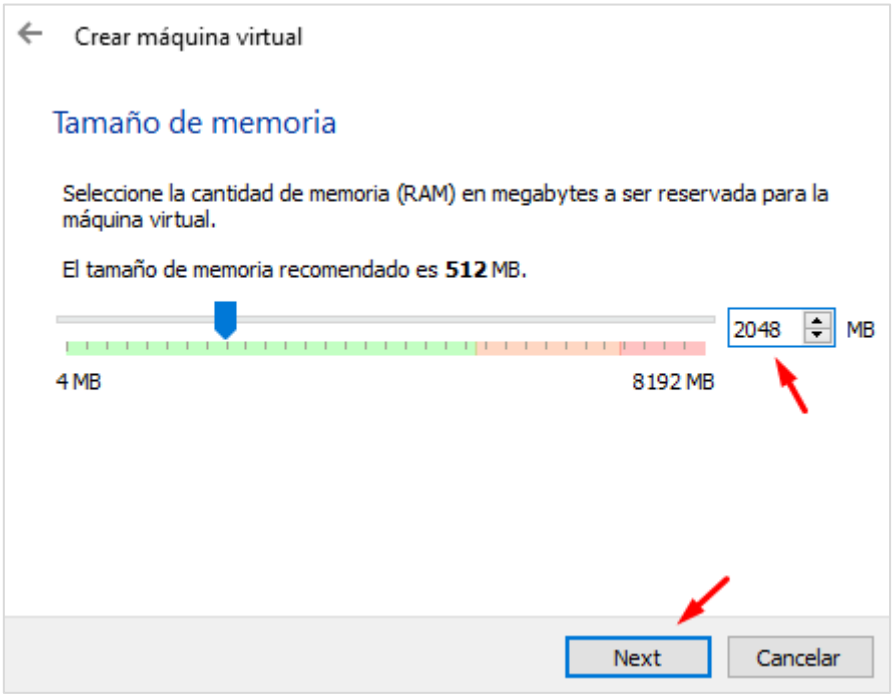


Figura 43: Asignación de memoria para la máquina virtual

Fuente: Elaboración del Autor

Seleccionamos el disco que descargamos y creamos.

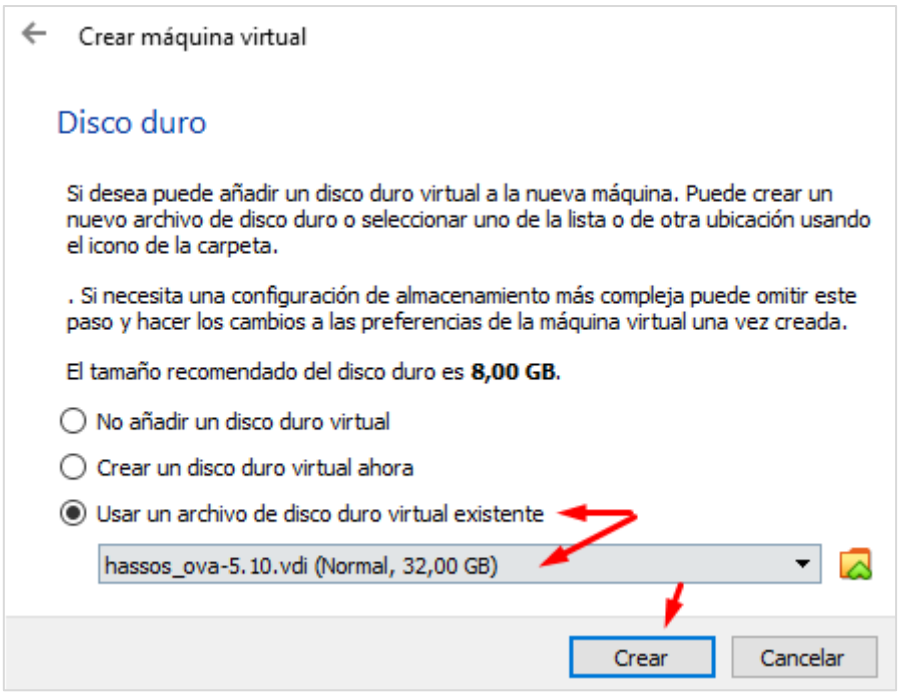


Figura 44: Seleccionamos el disco duro, archivo VDI

Fuente: Elaboración del Autor

Una vez creada nuestra máquina virtual nos dirigimos a configuraciones, “Sistemas” y habilitamos EFI.

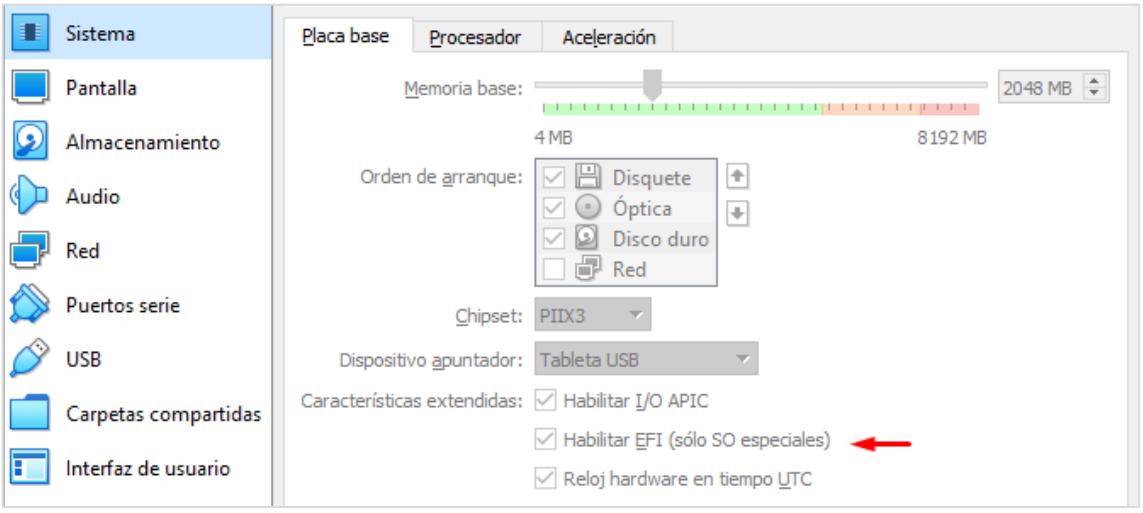


Figura 45: Habilitar EFI para la máquina virtual

Fuente: Elaboración del Autor

Luego nos dirigimos a “Red” y seleccionamos Adaptador Puente y aceptar.

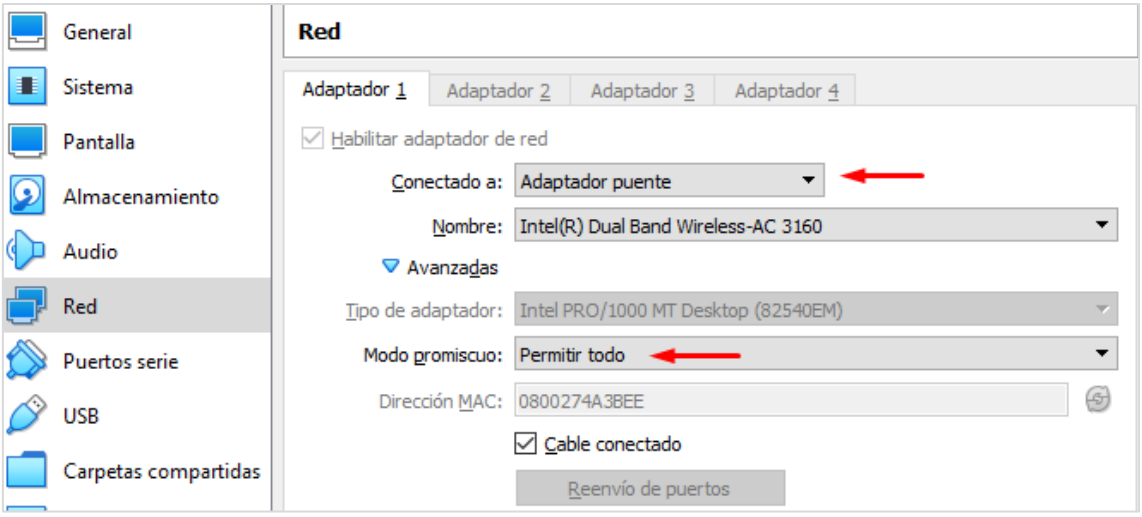


Figura 46: Configuraciones de red de la máquina virtual

Fuente: Elaboración del Autor

Por último, iniciamos la máquina virtual y comenzará el proceso de instalación.

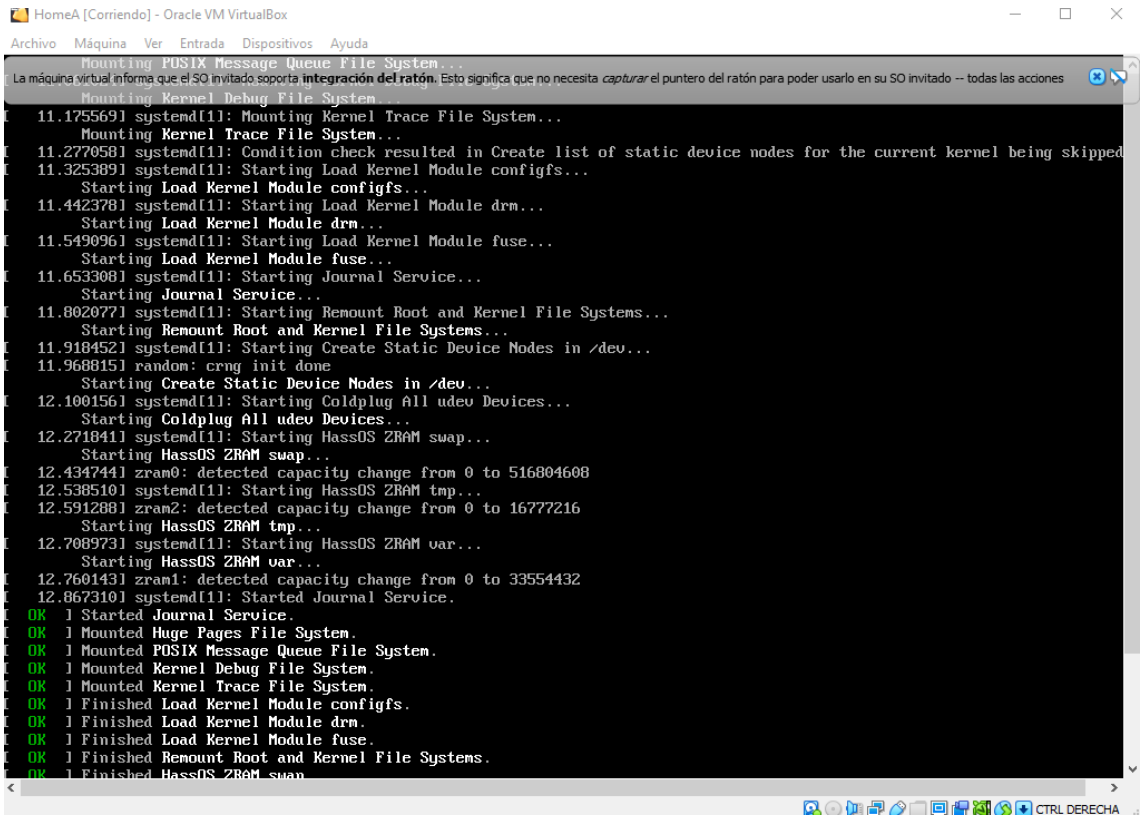


Figura 47: Instalación e inicio de Hassio en la máquina virtual

Fuente: Elaboración del Autor

Ingresamos “root” para iniciar, luego en “ha>” ingresamos “login” (ha > login) y listo. Por último, buscamos la IP utilizando el comando “nmcli”.

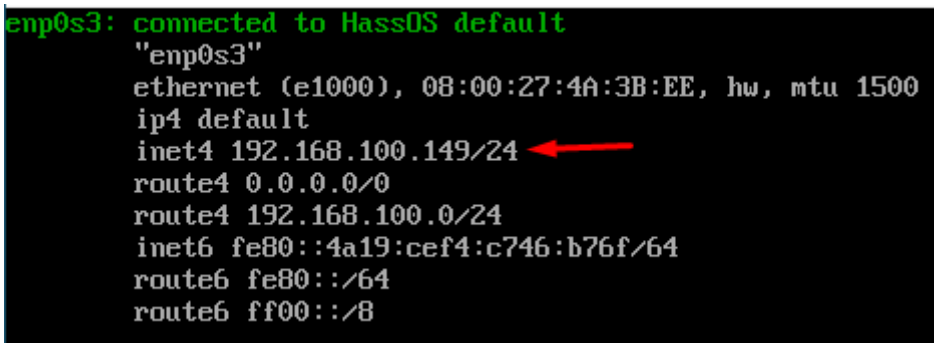


Figura 48: IP asignada para ingresar a Hassio

Fuente: Elaboración del Autor

La ingresamos en nuestro navegador añadiendo el puerto 8123. Una vez dentro esperamos a que inicie, luego creamos un usuario y contraseña.

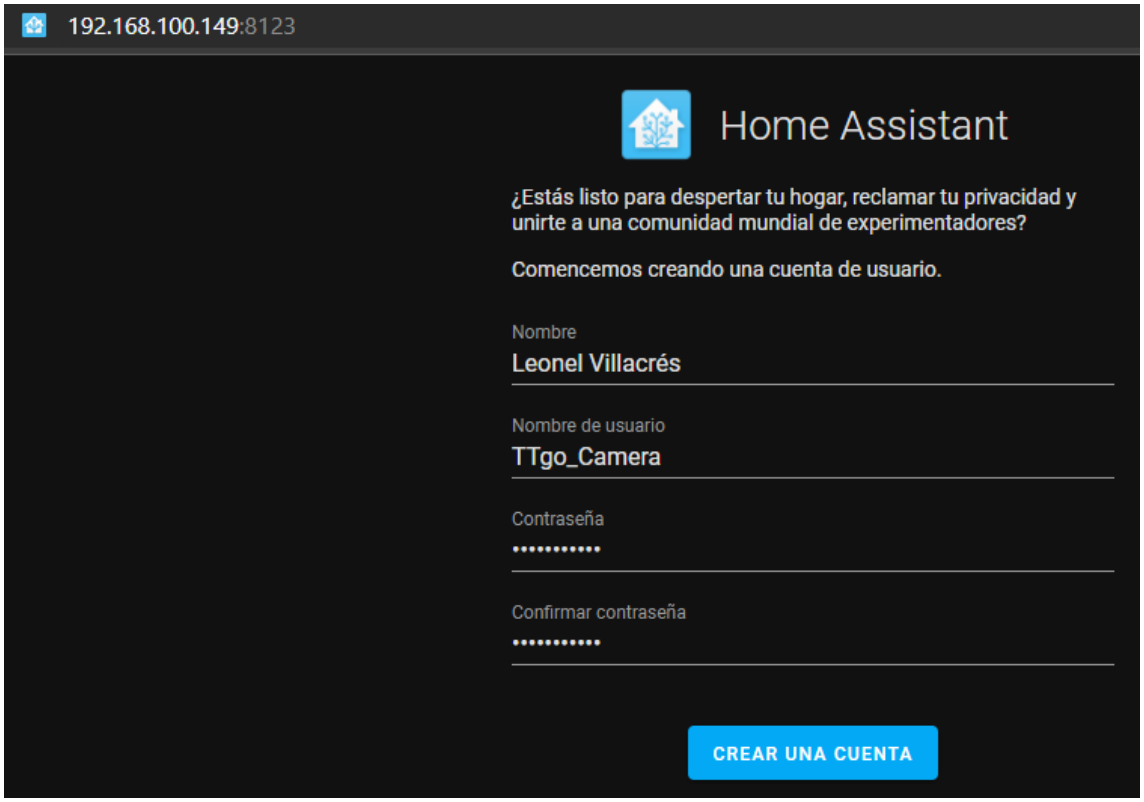


Figura 49: Creación del usuario y contraseña para Hassio

Fuente: Elaboración del Autor

Agregamos información de nuestra ubicación y damos clic en siguiente para por último terminar con el proceso.

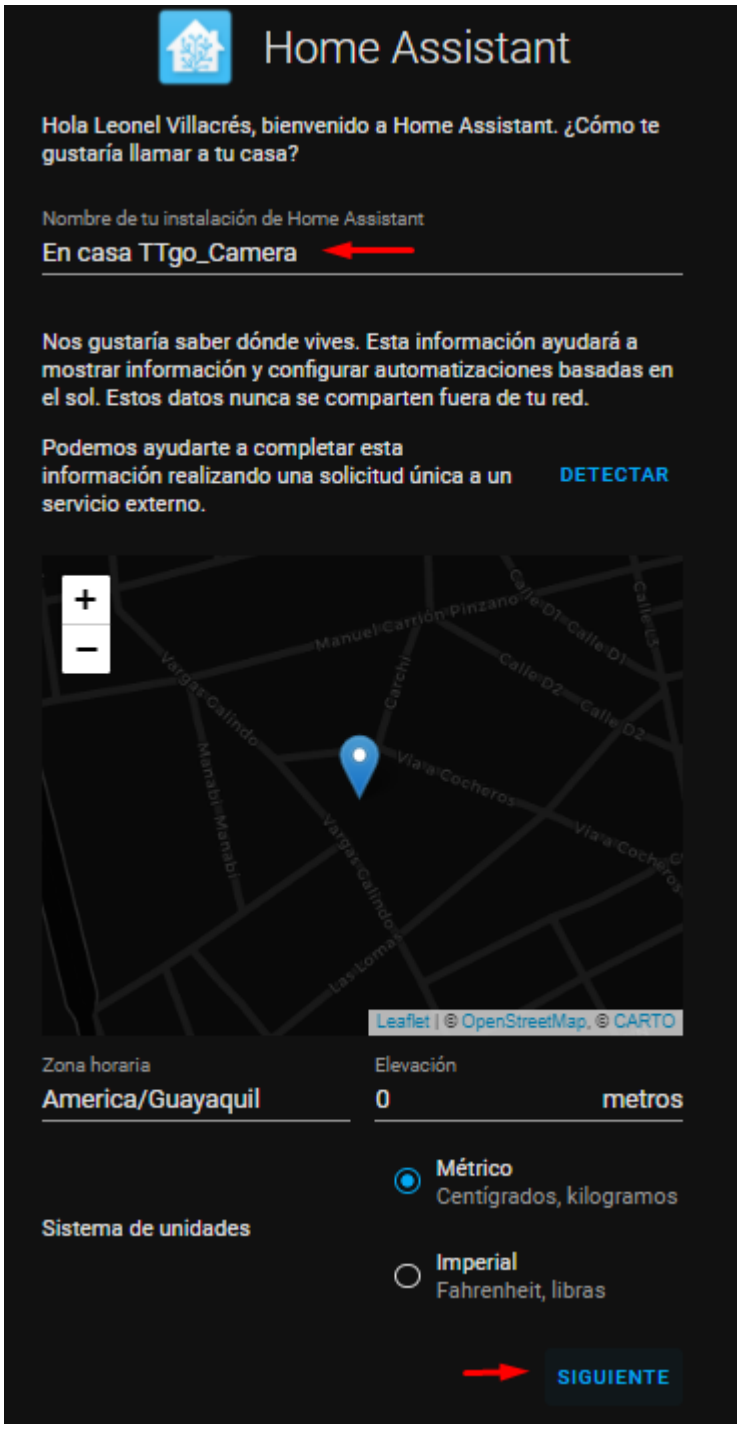


Figura 50: Agregar nombre de hogar y ubicación en Hassio

Fuente: Elaboración del Autor

Y listo, hemos terminado la instalación, como inicio nos aparecerá la siguiente pantalla.

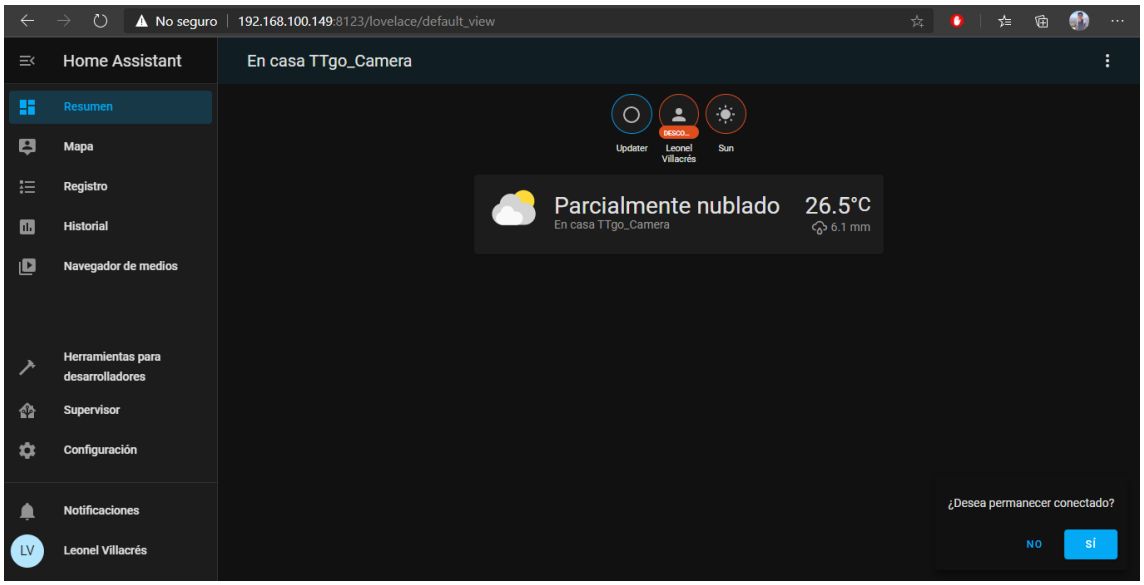


Figura 51: Pantalla de inicio de Hassio

Fuente: Elaboración del Autor

Ahora, si necesitamos opciones más avanzadas, sea para agregar Add-on avanzados o más configuraciones, nos dirigimos a “Usuario” y habilitamos la opción Modo avanzado.

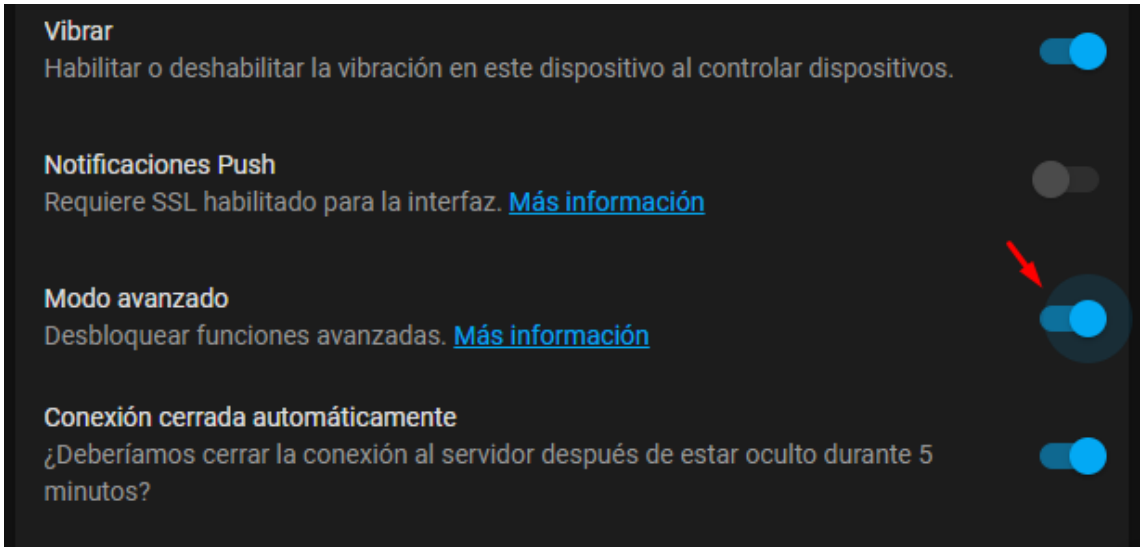


Figura 52: Habilitar modo avanzado en Hassio

Fuente: Elaboración del Autor

Para volver a iniciar el servicio de Home Assistant en la máquina virtual, se te pedirá un login, a este le ingresas “root”.

Establecer IP fija.

Cada vez que iniciemos Home Assistant a este se le asignará una IP para su arranque, de esto se encarga el DHCP. Pero, para nuestro proyecto no podemos permitir la asignación de manera aleatoria, esto debido a que el firmware del ESP32 debe tener conocimiento de la IP de este para su conexión es por esto que nuestro Home Assistant debe correr con una IP fija. Para realizar este proceso hacemos lo siguiente:

Anteriormente Home Assistant no permitía realizar estos cambios de manera gráfica y se tenía que recurrir a otros métodos, desde la creación de archivos en la raíz, hasta la manipulación del router. Pero, a partir de la última actualización del 2020 esto ya es posible, para ello nos dirigimos a Supervisor > System > Host, una vez dentro podremos hacer los cambios.

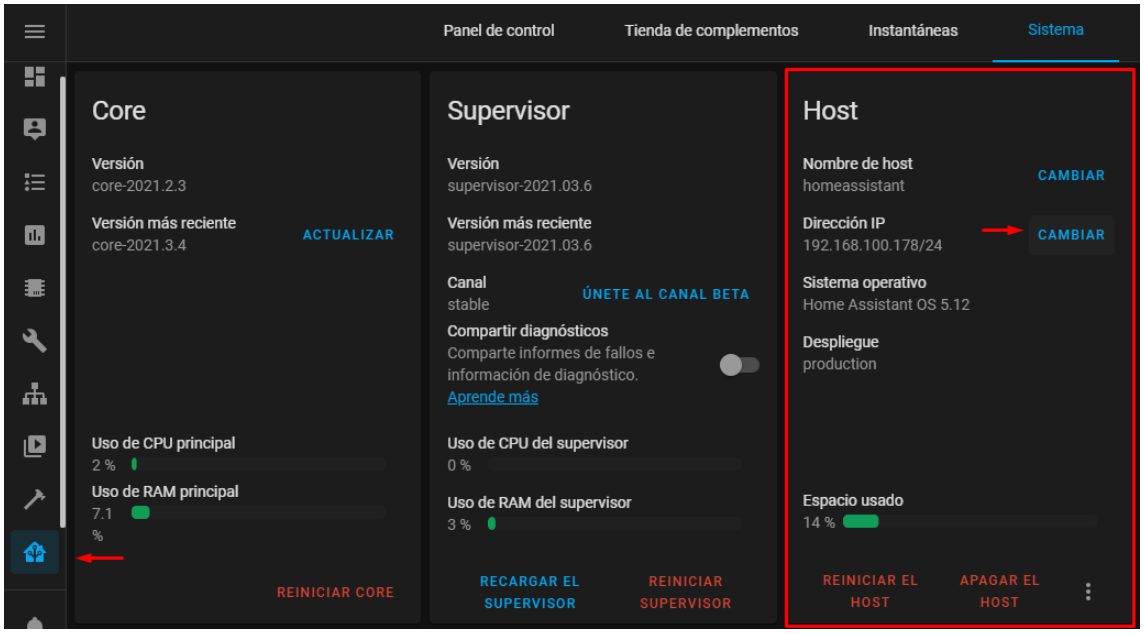


Figura 53: Cambio de IP de Hassio

Fuente: Elaboración del Autor

Seleccionamos Estática y escribimos la IP. Para este cambio asignaremos como IP la 192.168.100.178/24. Por último, damos clic en aceptar y reiniciamos el servicio.

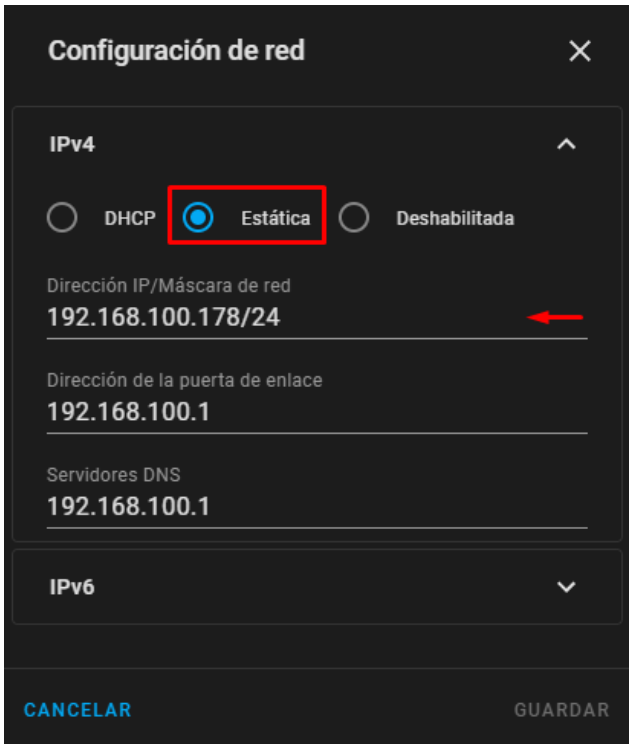


Figura 54: Asignación de la nueva IP para Hassio

Fuente: Elaboración del Autor

Cada vez que iniciamos la plataforma observaremos la IP asignada.

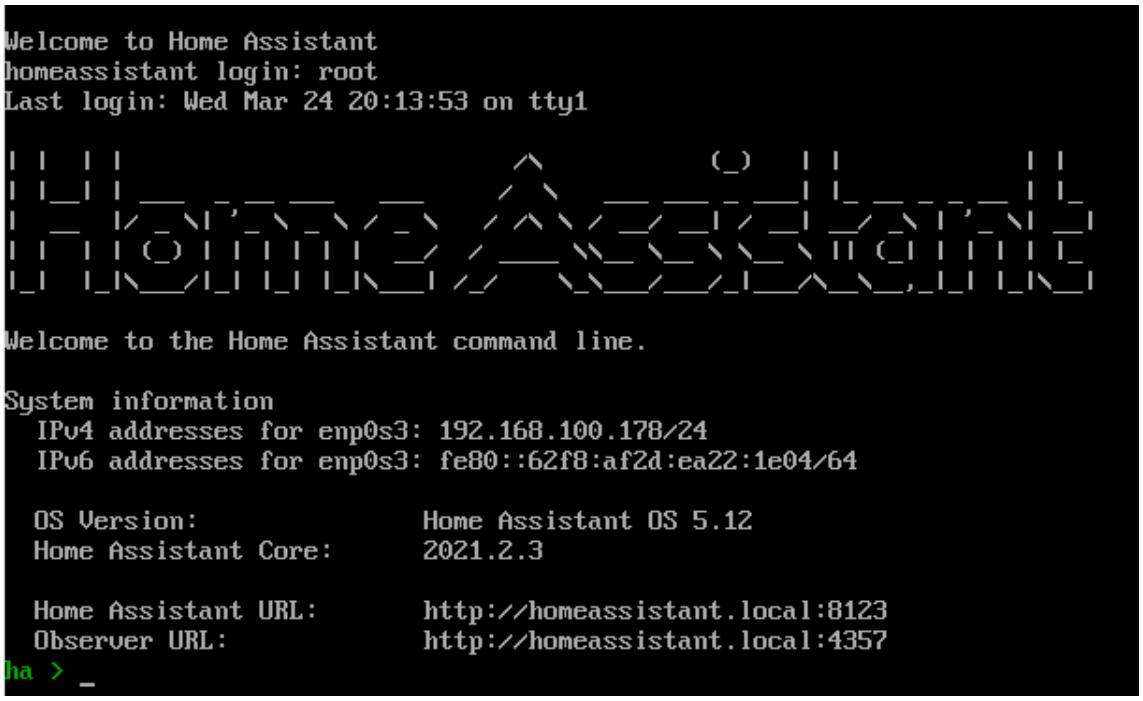


Figura 55: Inicio de Hassio utilizando la nueva IP

Fuente: Elaboración del Autor

2.5.4.1.1. Instalación del ADD-on File editor

Esta herramienta será imprescindible para la edición de ficheros de Home Assistant desde una interfaz web. Para su instalación nos dirigimos a Add-on Store de Home Assistant y buscamos File editor y lo seleccionamos.

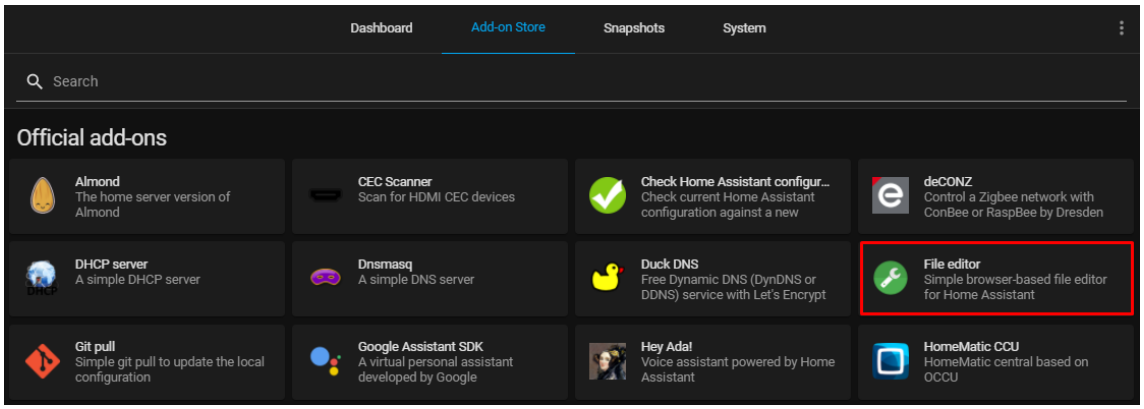


Figura 56: Búsqueda del Add-on File editor

Fuente: Elaboración del Autor

Una vez dentro damos clic en instalar.

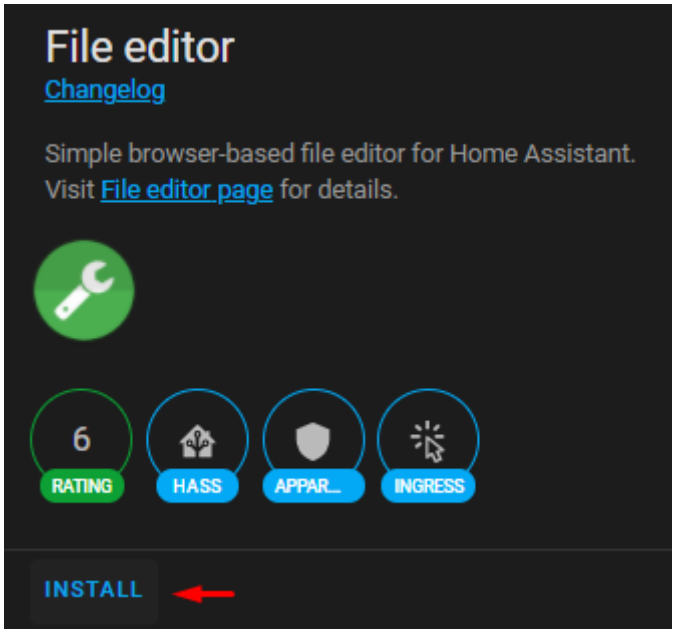


Figura 57: Instalación de File editor en Hassio

Fuente: Elaboración del Autor

Una vez instalado dejamos marcada la opción Start on boot para tenerlo de acceso rápido en las opciones de la parte izquierda del Home Assistant.

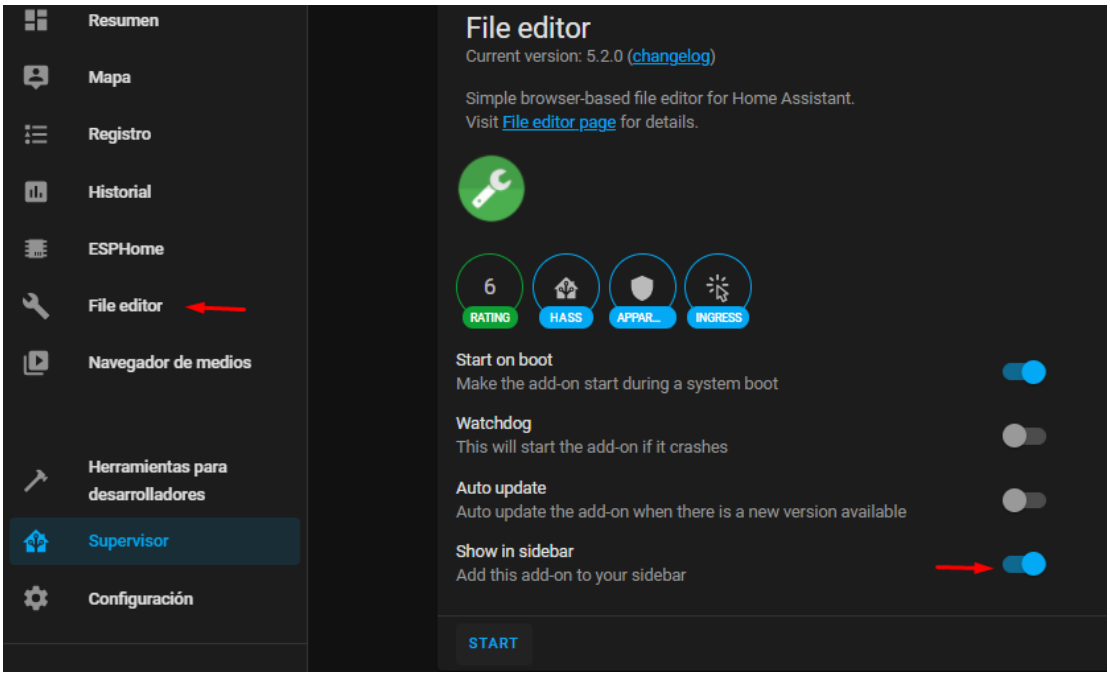


Figura 58: Inicio de la herramienta File editor

Fuente: Elaboración del Autor

Una vez culminado damos clic en iniciar (start) y lo abrimos, en el podemos editar cualquier fichero de Home Assistant, además de otras funcionalidades.

2.5.4.1.2. Instalación del Add-on Mosquitto Broker (MQTT)

Para la instalación primero debemos crear un usuario, para esto nos dirigimos a “Configuración > Usuarios > Agregar usuarios” y creamos uno.

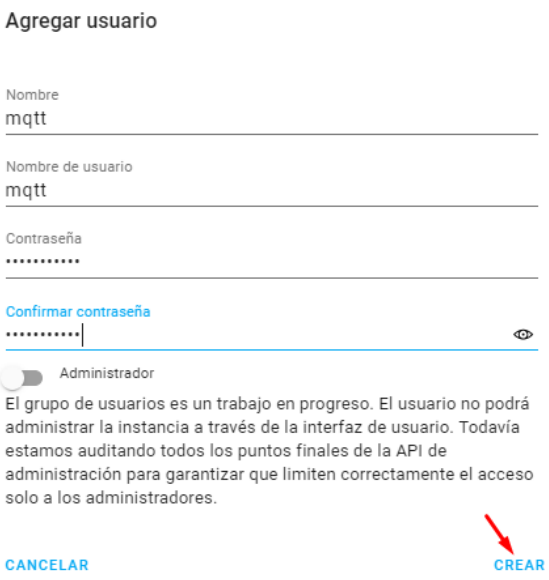


Figura 59: Creación de un usuario para MQTT

Fuente: Elaboración del Autor

Una vez creado el usuario nos dirigimos a la pestaña “Supervisor” y en la pestaña Store buscamos Mosquitto bróker, lo seleccionamos e instalamos.

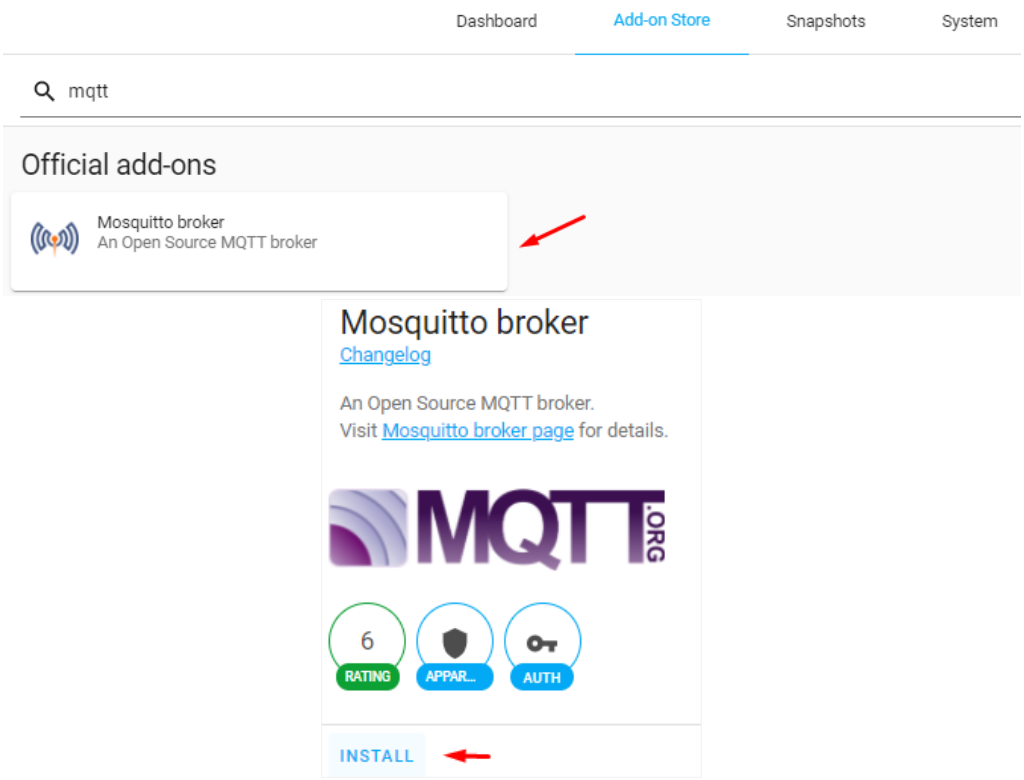


Figura 60: Búsqueda e instalación de Mosquitto broker

Fuente: Elaboración del Autor

Una vez instalado lo iniciamos y nos dirigimos a “Configuración > Integraciones”, ahí dentro encontraremos a MQTT.

Figura 61: Configuración de MQTT

Fuente: Elaboración del Autor

Damos clic en configurar, habilitamos y damos clic en enviar, nos aparecerá un mensaje de confirmación. Para finalizar reiniciamos el sistema.

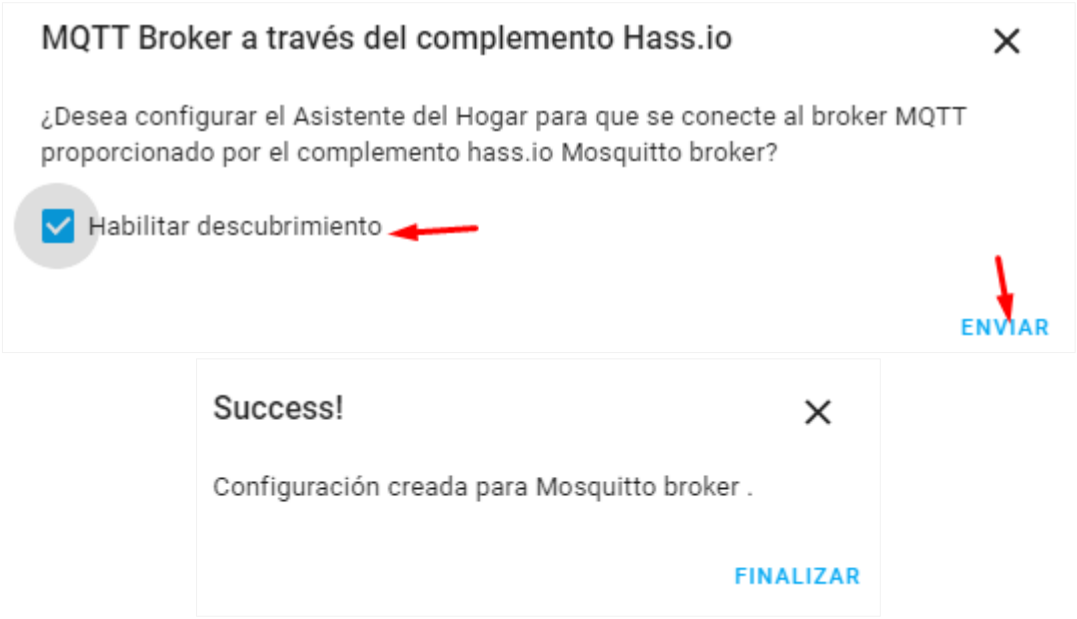


Figura 62: Habilitar descubrimiento de MQTT

Fuente: Elaboración del Autor

2.5.4.2. Integración de dispositivos domóticos comerciales

Los dispositivos a utilizar para la automatización del hogar (en conjunto con el sistema orientado a la domótica, Home Assistant) serán un bombillo y un enchufe, los cuales pertenecen a la empresa Nexxt Solutions. Un punto importante es que dicha empresa se asoció con Tuya Smart para expandir la cartelera de productos y soluciones para el hogar inteligente al mercado. Esta asociación brindó la oportunidad de integrar esta marca de dispositivos a Home Assistant.

La asociación permite que dispositivos de Nexxt puedan ser manipulados utilizando la app Tuya Smart, además de su propia aplicación. Esto nos va a servir para poder utilizar estos dispositivos en Hassio (debido a su compatibilidad con Home Assistant).

2.5.4.2.1. Descarga de la App Tuya Smart e integración de dispositivos

Para descargar la aplicación debemos ubicarla en la tienda de apps, dependiendo del dispositivo que se esté utilizando (Android, iOS, entre otros), en nuestro caso utilizaremos un dispositivo con sistema operativo Android.

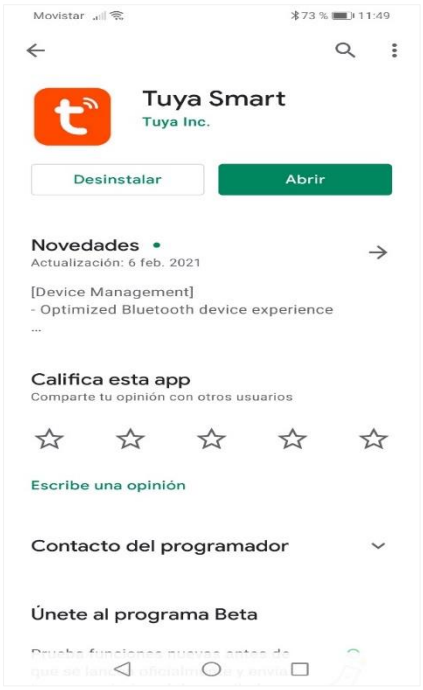


Figura 63: Tuya Smart en la Play Store

Fuente: Elaboración del Autor

Una vez descargado, lo abrimos y procedemos a crear una nueva cuenta o puede agregar su cuenta de Google, en nuestro caso crearemos una nueva.

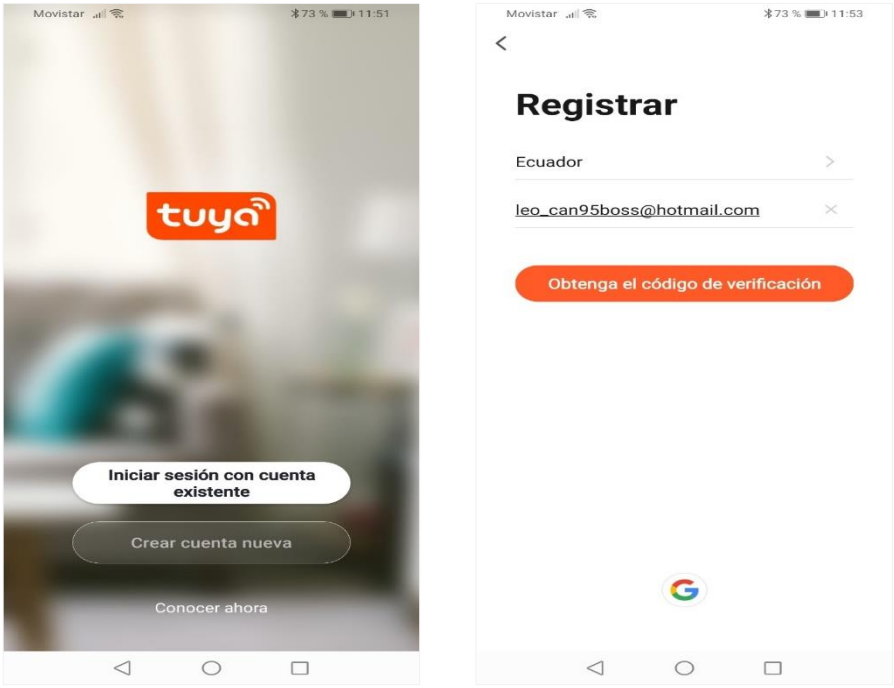


Figura 64: Registro en Tuya Smart

Fuente: Elaboración del Autor

Ya creada, procedemos a agregar un nuevo dispositivo, primero integraremos el enchufe, para ello nos dirigimos a la sección de “eléctricos” y seleccionamos el modelo, posteriormente nos pedirá las credenciales de nuestra red inalámbrica, en la cual conectaremos nuestros dispositivos y seguimos la guía de pasos.

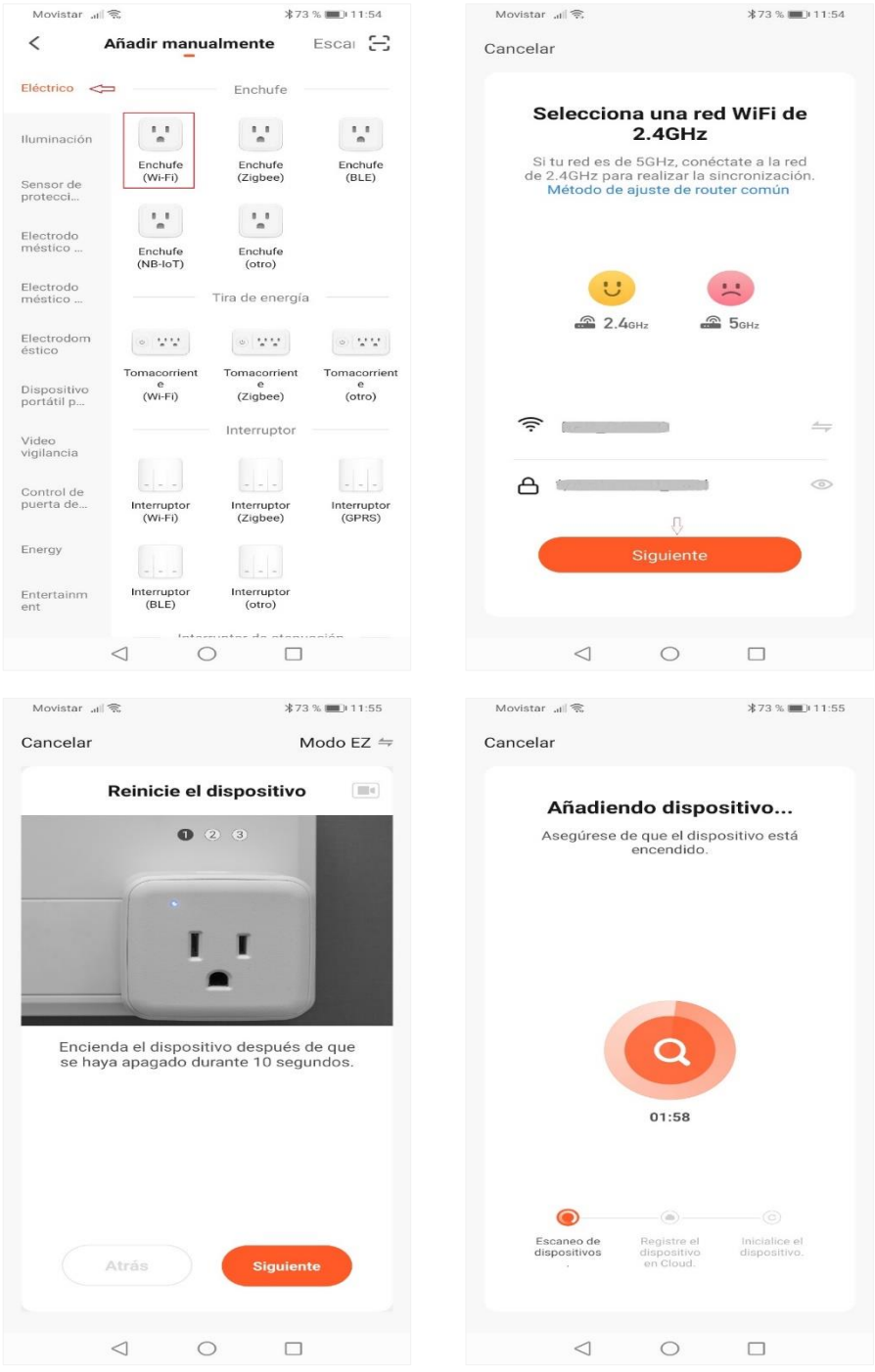


Figura 65: Integración del enchufe a Tuya Smart

Fuente: Elaboración del Autor

Una vez finalizado el proceso de integración ya podremos manipular el enchufe con nuestro dispositivo móvil. De igual manera seguimos los mismos pasos con el bombillo. Nos dirigimos a la sección de “Iluminación” y seleccionamos el modelo, posteriormente nos pedirá las credenciales de nuestra red inalámbrica, la agregamos y seguimos la guía de pasos.

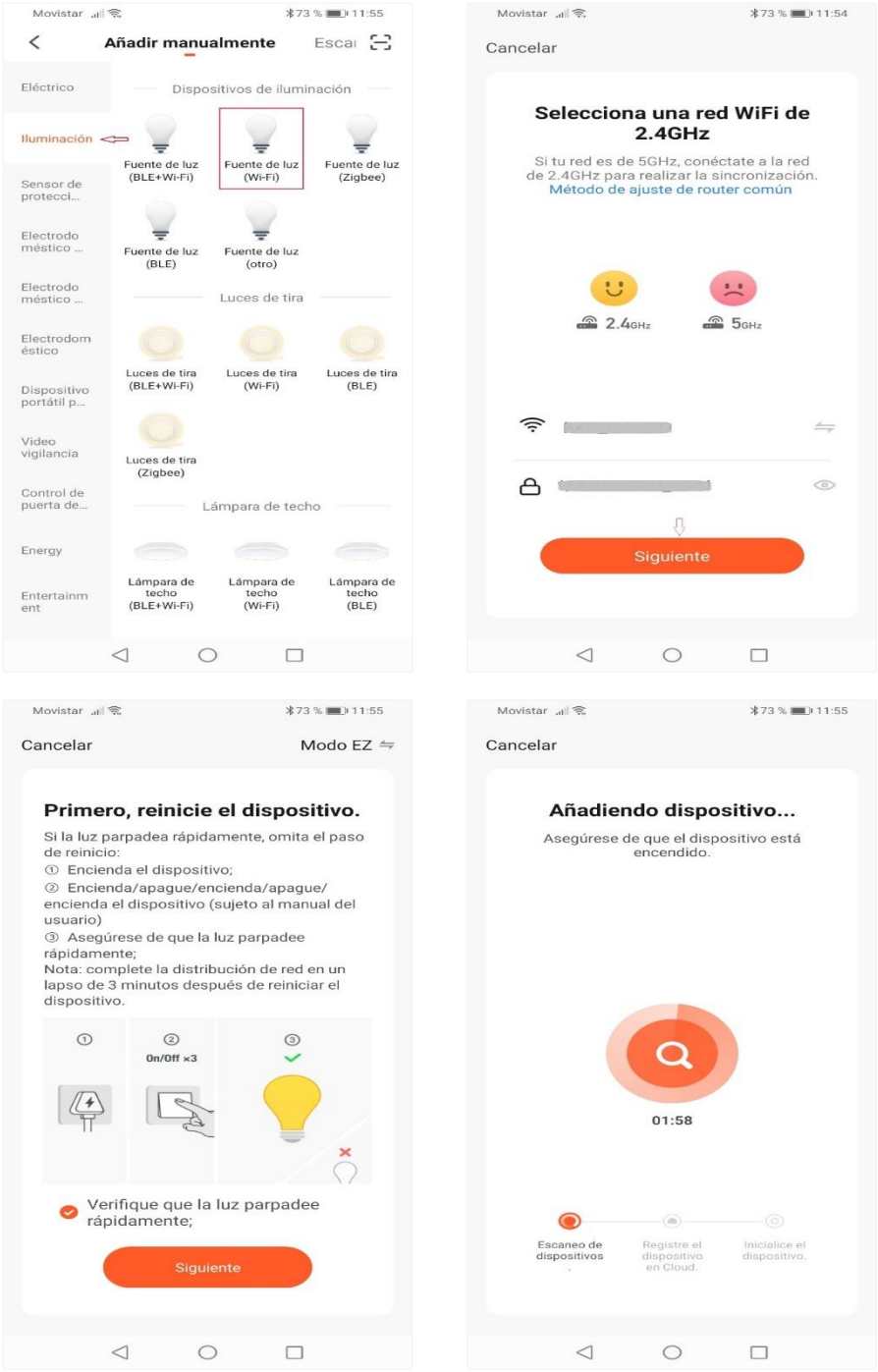


Figura 66: Integración del bombillo a Tuya Smart

Fuente: Elaboración del Autor

Finalizado el proceso ya podremos visualizar ambos dispositivos en la sección “Mi hogar”.

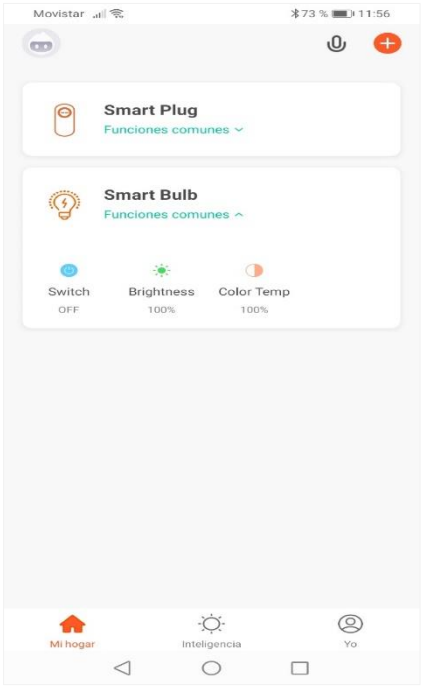


Figura 67: Pantalla de inicio de Tuya Smart

Fuente: Elaboración del Autor

Finalmente podemos hacer pruebas utilizando el dispositivo móvil.

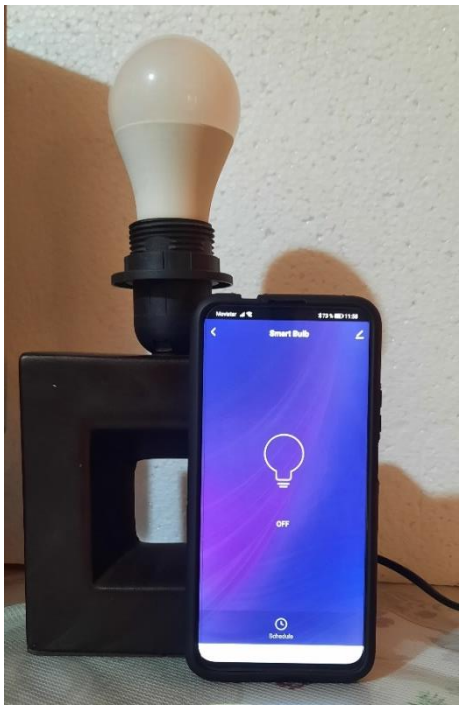


Figura 68: Encendido y apagado del bombillo utilizando la App

Fuente: Elaboración del Autor

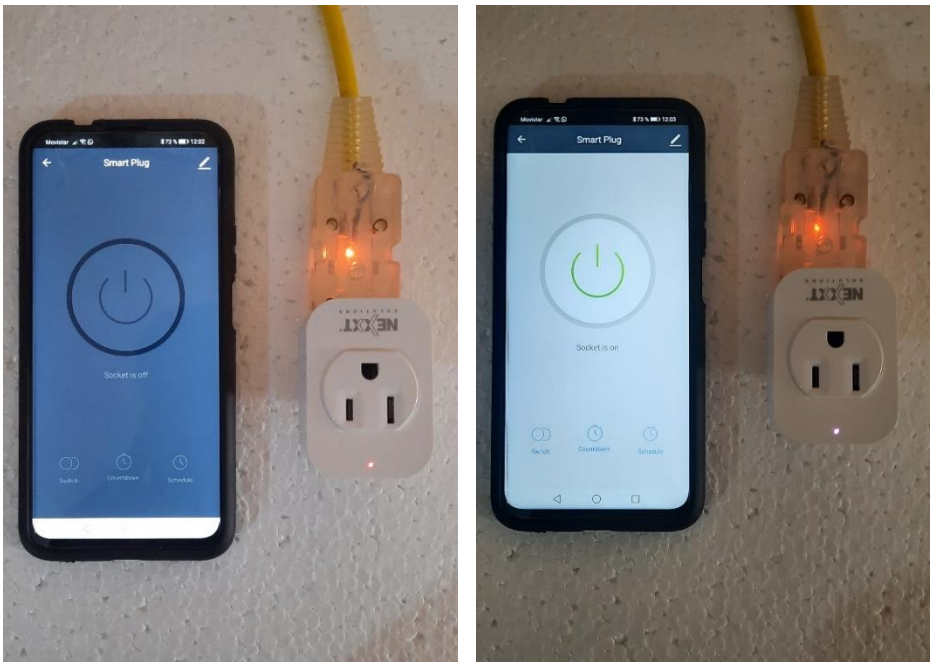


Figura 69: Encendido y apagado del enchufe utilizando la App

Fuente: Elaboración del Autor

2.5.4.2.2. Integración y automatización de dispositivos en Hassio

La comunidad de home Assistant proporciona soporte para integrar diferentes dispositivos comerciales de manera muy fácil, si nos dirigimos a su sitio oficial,

en la sección integraciones, podemos observar la cantidad de marcas soportadas para su incorporación en Hassio. Una de ellas y la que utilizaremos para integrar es Tuya Smart. La cual, para integrar hacemos lo siguiente: Nos dirigimos a “Configuración > Integraciones > Añadir integración”.

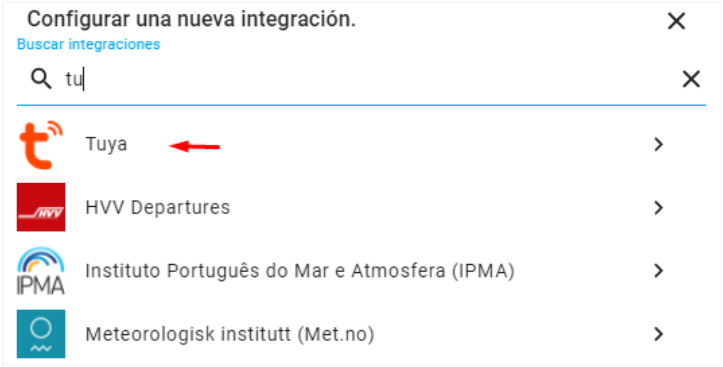


Figura 70: Búsqueda de la marca Tuya para su integración

Fuente: Elaboración del Autor

Nos pedirá el usuario, la contraseña, el código de nuestro país y seleccionar la aplicación en la cual registramos la cuenta. Recordar que el usuario y contraseña hace referencia a la que agregamos para crear nuestra cuenta en la app de Tuya Smart, mientras que el código del país para nuestro caso, Ecuador, es el 593; por último, seleccionamos Tuya, la cual, fue la app utilizada para crear la cuenta.

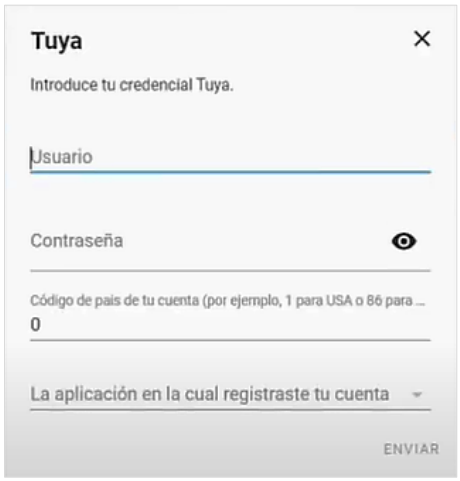


Figura 71: Registro del dispositivo tuya en la plataforma Hassio

Fuente: Elaboración del Autor

Si observamos en el panel “Integraciones” caeremos en cuenta que este se agregó correctamente y que estos dispositivos fueron detectados, pero, necesitamos que estos se encuentren en el archivo de configuración de Hassio.

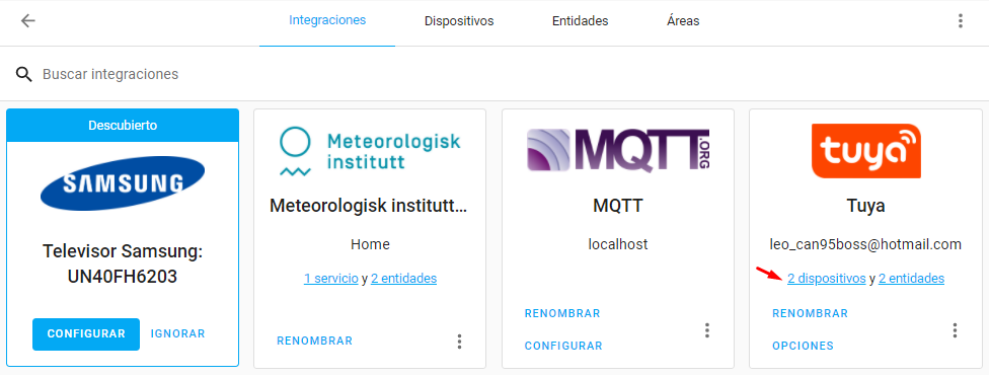


Figura 72: Lista de integraciones de Hassio

Fuente: Elaboración del Autor

En el archivo de configuración agregamos las siguientes líneas de código.

```
# Tuya configuration
tuya:
    username: YOUR_TUYA_USERNAME
    password: YOUR_TUYA_PASSWORD
    country_code: YOUR_ACCOUNT_COUNTRYCODE
```

Una vez realizado, reiniciamos el sistema para que se apliquen los cambios, esto se lo realiza dirigiéndose a “Configuración > Configuración general > Controles de servicio > Reiniciar”. Ahora, para agregar los dispositivos al panel principal seleccionamos en la parte superior derecha “Editar panel de control > Añadir tarjeta” y seleccionamos los dispositivos que queramos mostrar. Finalmente podemos realizar pruebas para comprobar su funcionamiento.

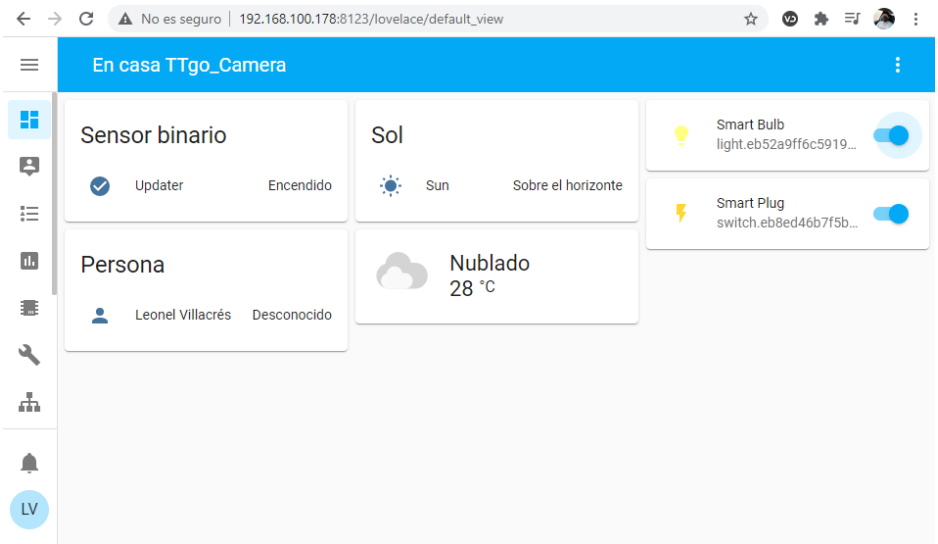


Figura 73: Integración del bombillo y del enchufe en el panel principal

Fuente: Elaboración del Autor

Controlar entidades de Hassio desde el dispositivo ESP32 utilizando MQTT.

Para este proceso se necesita tener instalado en nuestro Home Assistant el Broker Mosquito, el cual es un servidor intermediario MQTT que recibe y distribuye mensajes a través del servidor MQTT. Una vez instalado necesitamos agregar los dispositivos en el archivo de configuración “configuration.yaml”, a estos se los agrega como switch debido a que funcionará como el mismo. No olvidar que en el mismo archivo también debe estar registrado MQTT.

```
mqtt:
  broker: localhost
  port: 1883
  username: 'mqtt'
  password: 'Leonel/*123'

# Discover some devices automatically
discovery:

# Interruptor ESP32 bombillo
switch:
  - platform: mqtt
    name: "Example_Switch"
    command_topic: "room/light"
    payload_on: "on"
    payload_off: "off"

# Interruptor ESP32 enchufe
switch:
  - platform: mqtt
    name: "Example_Switch_Plug"
    command_topic: "room/plug"
    payload_on: "on"
    payload_off: "off"
```

Al observar el switch notamos que tiene como topic “room/light” y como acciones “on y off”, este lo utilizará para encender y apagar el bombillo. Lo mismo sucede con el enchufe, pero con un nombre y topic diferente.

Ahora necesitamos crear una nueva automatización, primero para el bombillo, para esto nos dirigimos a “Configuración > Automatizaciones > Añadir automatización” y seleccionamos “Empezar con una automatización vacía”.

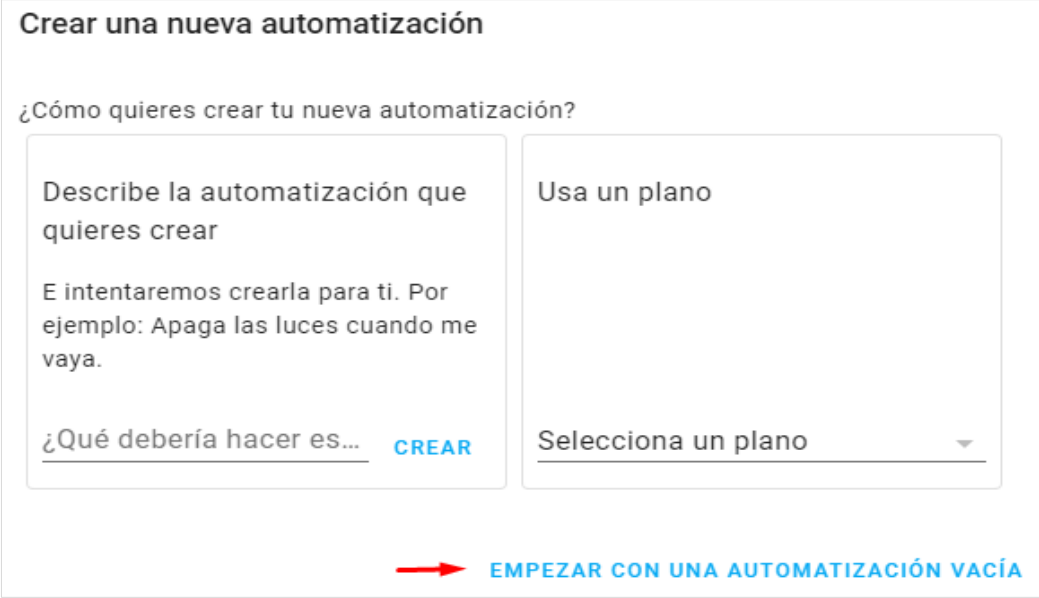


Figura 74: Crear una nueva automatización

Fuente: Elaboración del Autor

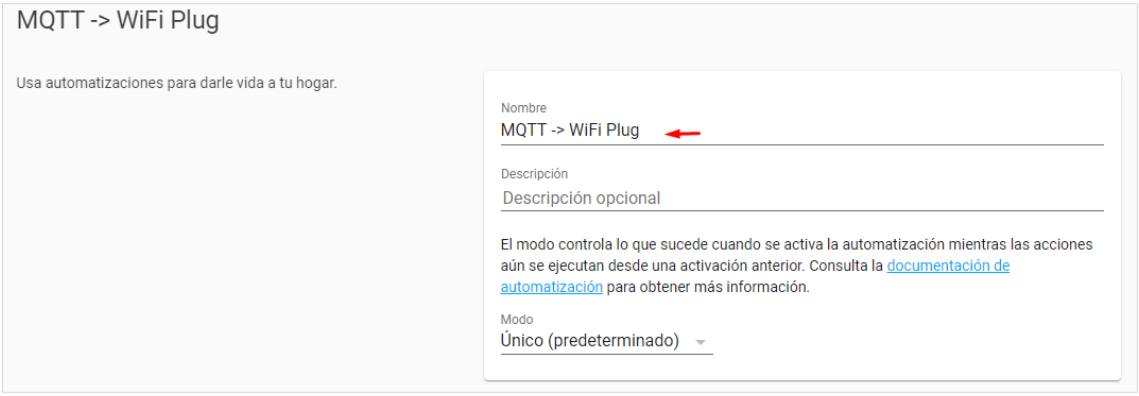


Figura 75: Agregar un nombre a la automatización

Fuente: Elaboración del Autor

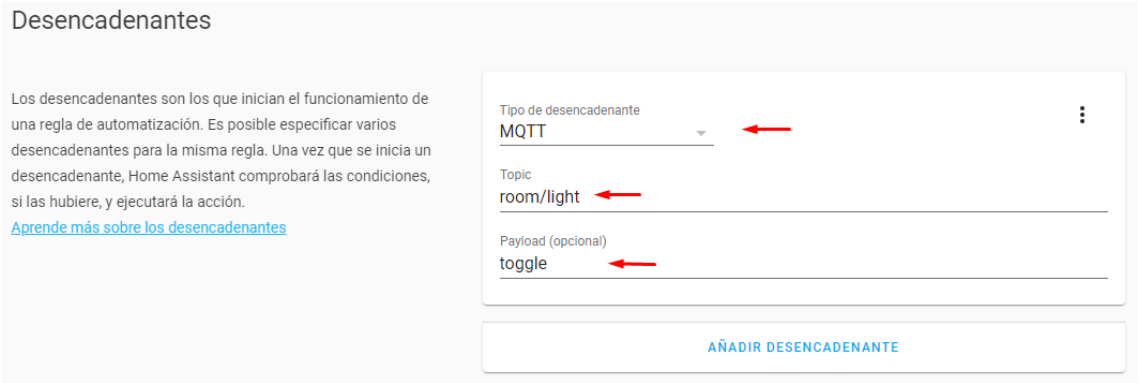


Figura 76: Agregamos un desencadenante y el topic a la automatización

Fuente: Elaboración del Autor

En acciones agregamos lo siguiente, no olvidar que al tratarse de una bombilla debemos poner en servicio “light.toggle”, en nombre ponemos el dispositivo con su id (este id se lo encuentra al momento de agregar el dispositivo en el panel principal o buscándolo directamente en integraciones > tuya o en servicios).

Figura 77: Agregamos el servicio y el dispositivo

Fuente: Elaboración del Autor

Guardamos los cambios, reiniciamos el sistema y listo. Ahora, para automatizar el enchufe hacemos exactamente lo mismo, pero en el servicio seleccionamos “switch.toggle”. Guardamos y reiniciamos el sistema. Con esto, son dos las automatizaciones que tenemos en nuestro Hassio, estas nos servirán para manipular estos dos dispositivos por medio de nuestro ESP32 a través de MQTT.

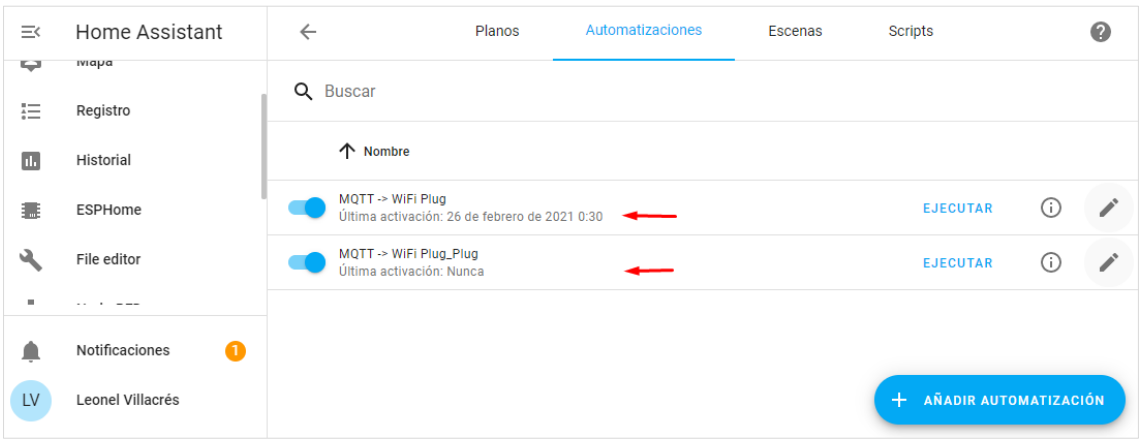


Figura 78: Automatizaciones realizadas

Fuente: Elaboración del Autor

Con esto tenemos agregado nuestros dispositivos a Home Assistant como switch y pertenecen como cliente a MQTT, solo nos faltaría el firmware necesario para ejecutar estas acciones. Para ello haremos uso de la librería “PubSubClient”, la

cual nos va a proporcionar un cliente para realizar mensajes de publicación/suscripción simple con un servidor que admita MQTT.

Figura 79: Descarga de la librería PubSubClient en PlatformIO

Fuente: Elaboración del Autor

Una vez instalada aparecerá en nuestro Platformio.ini. El proceso de comunicación por MQTT es el siguiente: primero importamos las librerías.

```
#include <WiFi.h>
#include <PubSubClient.h>
```

Ahora iniciamos las variables necesarias como lo son: El SSID y la contraseña de nuestra red wifi en la que se encuentran nuestros dispositivos, las credenciales de Hassio y la información MQTT de nuestros dispositivos. La IP del Broker es la misma en la que se ejecuta Hassio.

```
// WiFi Network Credentials
const char *ssid = "NAME_RED";  // name of your WiFi network
const char *password = "PASSWORD"; // password

// Home Assistant Credentials
const char *HA_USER = "TTgo_Camera";
const char *HA_PASS = "Leonel/*123";

// MQTT Network
IPAddress broker(192,168,100,178); // IP address of your MQTT broker

//Bombillo
const char *ID_LIGHT = "Example_Switch"; //Name of our device
const char *TOPIC_LIGHT = "room/light";  // Topic to subcribe to
const char *MESSAGE = "toggle";   // Message to publish to our topic

//Enchufe
```

```
const char *ID_PLUG = "Example_Switch_Plug"; // Name of our device
const char *TOPIC_PLUG = "room/plug";  // Topic to subcribe to

WiFiClient wclient;

PubSubClient client(wclient); // Setup MQTT client
```

Creamos un método llamado reconnect() que se encargará de la conexión con MQTT y a la vez de las publicaciones.

```
// Reconnect to client
void reconnect() {
  // Loop until we're reconnected

  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect(ID_LIGHT,HA_USER,HA_PASS)) {
      Serial.println("connected");
      Serial.print("Publishing to: ");
      Serial.println(TOPIC_LIGHT);
      Serial.println('\n');

    } else {
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }

    if (client.connect(ID_PLUG,HA_USER,HA_PASS)) {
      Serial.println("connected");
      Serial.print("Publishing to: ");
      Serial.println(TOPIC_PLUG);
      Serial.println('\n');

    } else {
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
```

En setup() iniciamos la red y el broker.

```
WiFi.begin(ssid, password); // Connect to network

while (WiFi.status() != WL_CONNECTED) { // Wait for connection
  delay(500);
  Serial.print(".");
}
client.setServer(broker, 1883);
```

Para finalmente en el lazo realizar la conexión y las acciones necesarias para ejecutar la publicación.

```
if (!client.connected()){ // Reconnect if connection is lost
    reconnect();}
client.loop();
client.publish(TOPIC, MESSAGE);
```

2.5.5. Implementación de las Redes Neuronales Artificiales

Para el desarrollo e implementación de redes neuronales es necesario una lista de módulos, teniendo como principal Python (64 bits), en el cual se desarrollará el proceso de redes neuronales para su posterior implementación en el firmware.

Tabla 5: Módulos necesarios para el desarrollo de redes neuronales

Software (paquete, librería, etc.)	Instalación
Tensorflow	pip install tensorflow
Numpy	pip install numpy
Pyserial	pip install pyserial
Matplotlib	pip install matplotlib
StandardScaler de Scikit-learn	pip install -u scikit-learn

Fuente: Elaboración del Autor

Los módulos mencionados en la Tabla 5 son fundamentales para el proceso de redes neuronales, en cuanto a la versión, es necesario poseer la última de cada una. Su instalación es muy sencilla usando únicamente la ventana de comandos de su sistema operativo o haciendo uso de un entorno virtual (recomendable).

2.5.5.1. Observar el comportamiento de las señales obtenidas del sensor.

La idea en este paso es observar el movimiento de cada señal y determinar cuáles de estas señales son las más relevantes a utilizar para el entrenamiento de nuestra red neuronal.

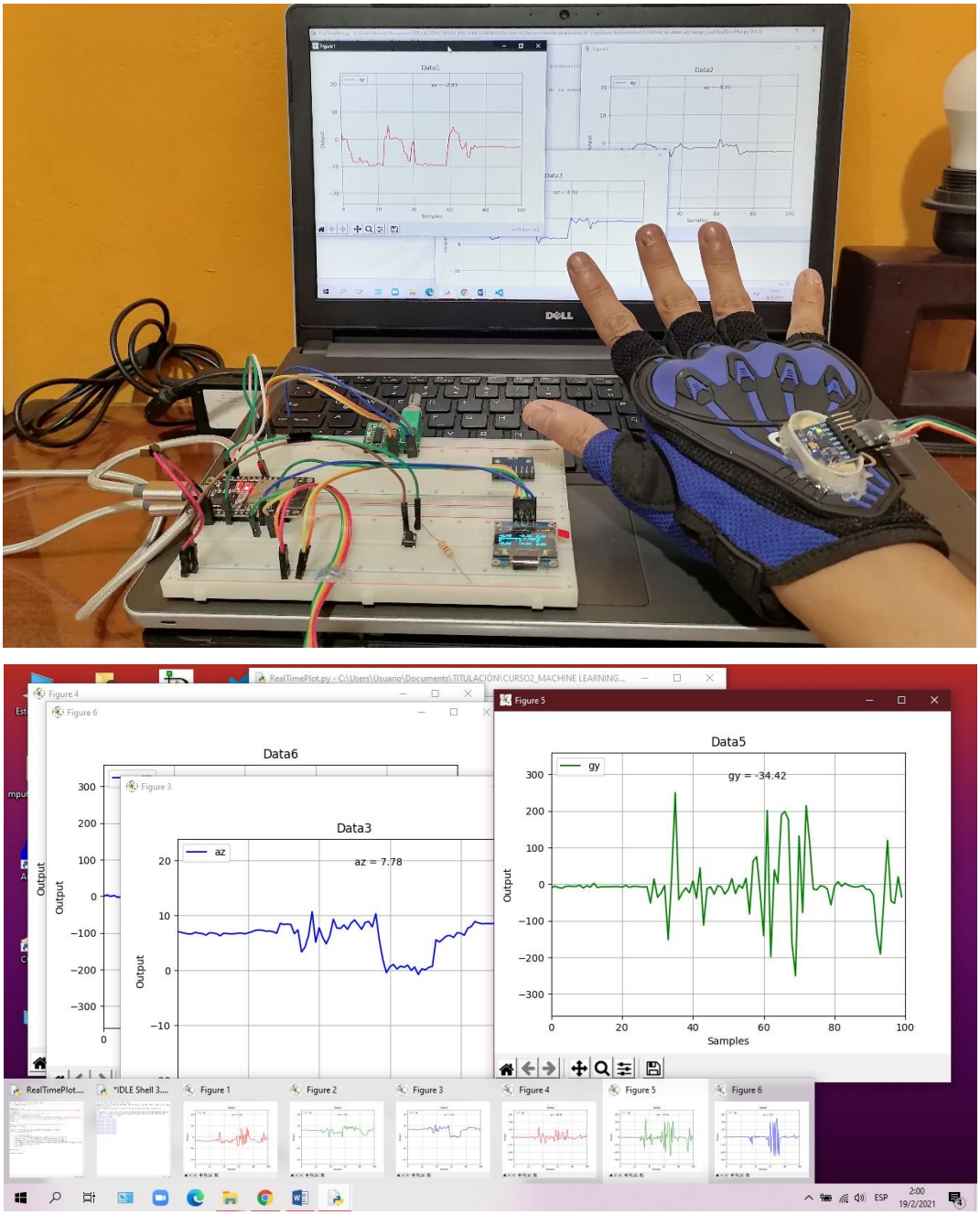


Figura 80: Señales obtenidas del sensor

Fuente: Elaboración del Autor

Al observar las señales pudimos notar que las más apropiadas para este proceso son las obtenidas del acelerómetro, debido a que las mismas nos permiten detectar fuerzas dinámicas como vibraciones o movimientos.

2.5.5.2. Generar un conjunto de datos

Al recolectar un conjunto de datos, cada uno de ellos debe tener un tiempo (para todos será el mismo, en este caso 3 segundos), el cual se utilizará para detectar el movimiento, generando así series temporales. Además de que son 9 los valores obtenidos en cada movimiento, 3 pertenecientes al valor eficaz, 3 de longitud de onda y 3 del valor medio absoluto.

Son 10 los gestos que enviaremos a entrenamiento y para este paso generaremos un conjunto de datos de 500 muestras para cada gesto, Tabla 6.

Tabla 6: Conjunto de datos para entrenamiento

Conjunto de datos		
ID	**Gesto**	**Datos**
G1	Buenos días	500
G2	Buenas tardes	500
G3	Buenas noches	500
G4	¿Cómo te sientes?	500
G5	Luz encendida, Luz apagada	500
G6	Tomacorriente activado, Tomacorriente desactivado	500
G7	¿Qué hora es?	500
G8	¿Cómo llego a esta dirección?	500
G9	¿Cómo te llamas?	500
G10	¿Cuántos años tienes?	500
	Total:	**5000**

Fuente: Elaboración del Autor

Como podemos observar en la Figura 81 en la que muestra el resultado final una vez se concluye de recolectar las 500 muestras de un gesto. Este arroja 9 gráficas de barra, cada una perteneciente a las características del movimiento obtenido por el acelerómetro. Estos valores se almacenarán en un archivo de formato npy.

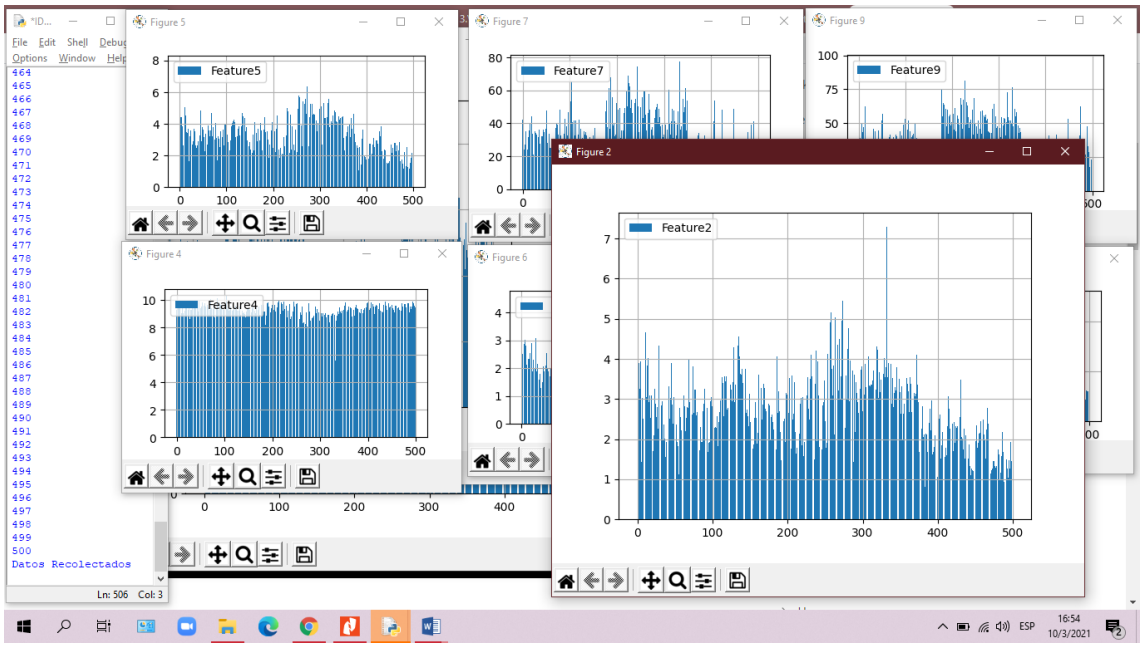


Figura 81: Resultados de la recolección del conjunto de datos del gesto G1

Fuente: Elaboración del Autor

2.5.5.3. Entrenamiento

Para la etapa de entrenamiento se dividió el conjunto de datos, en datos de entrenamiento y datos de prueba. Es decir, del conjunto de datos se obtendrá un 20% para usarlos para validación y 80% para entrenamiento.

Son varios los Hiper-Parámetros que se establecieron para lograr obtener un modelo aceptable y capaz de dar solución al problema, entre ellos tenemos:

- Número de iteraciones
- Tasa de aprendizaje.
- Cantidad de neuronas ocultas.
- Optimizador
- Número de capas

Además de las pruebas realizadas al modelo generado, como las pruebas de validación, necesarias para observar qué tan bien funciona el modelo con nuevos datos. Otra prueba realizada al modelo es la de overfitting con un subconjunto del 20% de los datos de entrenamiento para validación. Un punto importante a tener en cuenta es que mientras más complejo sea el modelo se va a consumir más recursos en el dispositivo. Es recomendable trabajar con un modelo más sencillo (menos parámetros).

2.5.5.4. Inferencia

Una vez completado el proceso de entrenamiento se sustraen los pesos, los valores de normalización y la estructura de la red neuronal y se integra esa información en nuestro firmware. El orden en el que se ingresaron los datos al entrenamiento será el mismo en el que el sistema los etiquetará, es decir, la etiqueta 0 pertenece al G1, la etiqueta 1 al G2 y así sucesivamente.

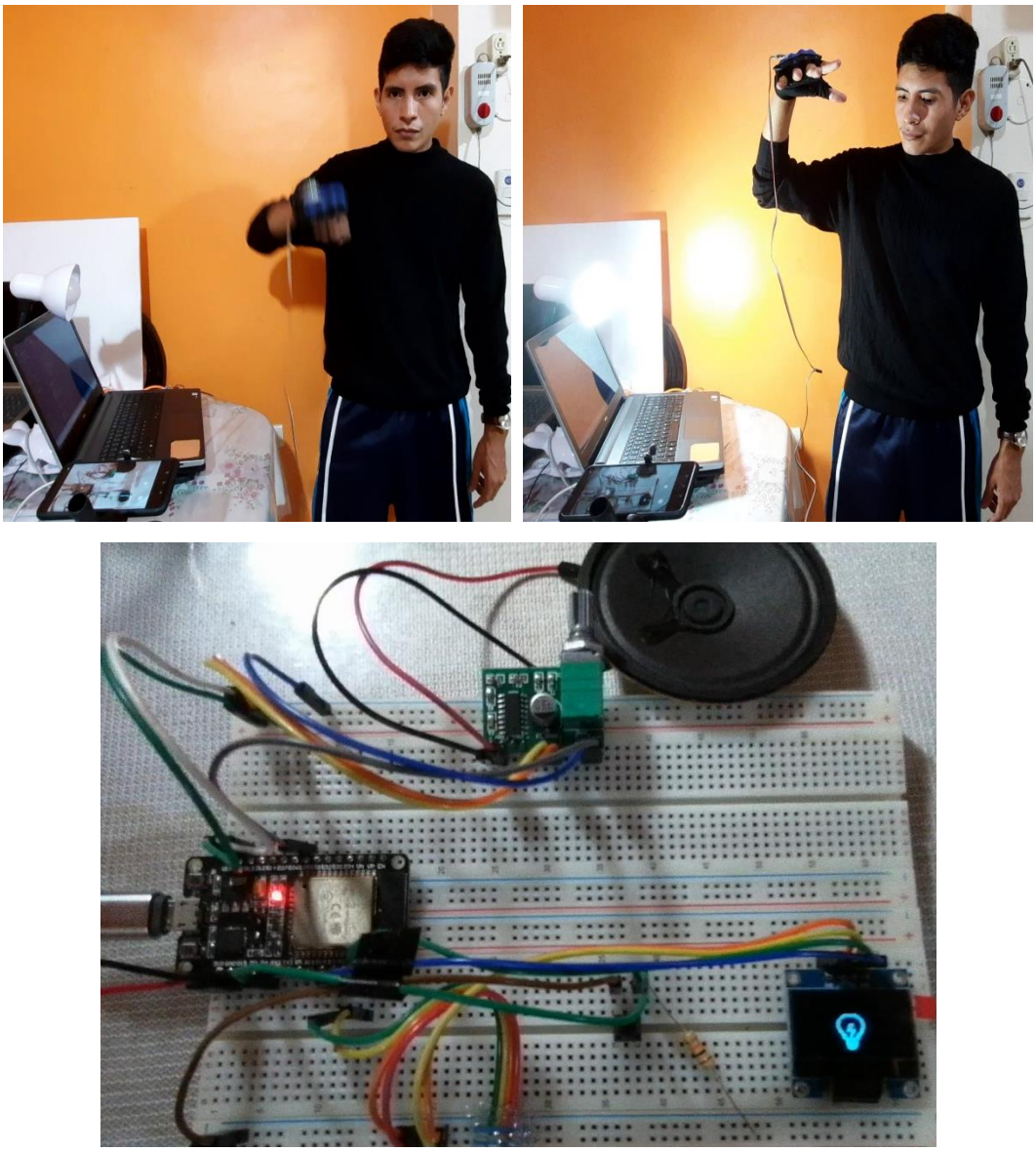


Figura 82: Realizando el Gesto Encender, iluminar G5

Fuente: Elaboración del Autor

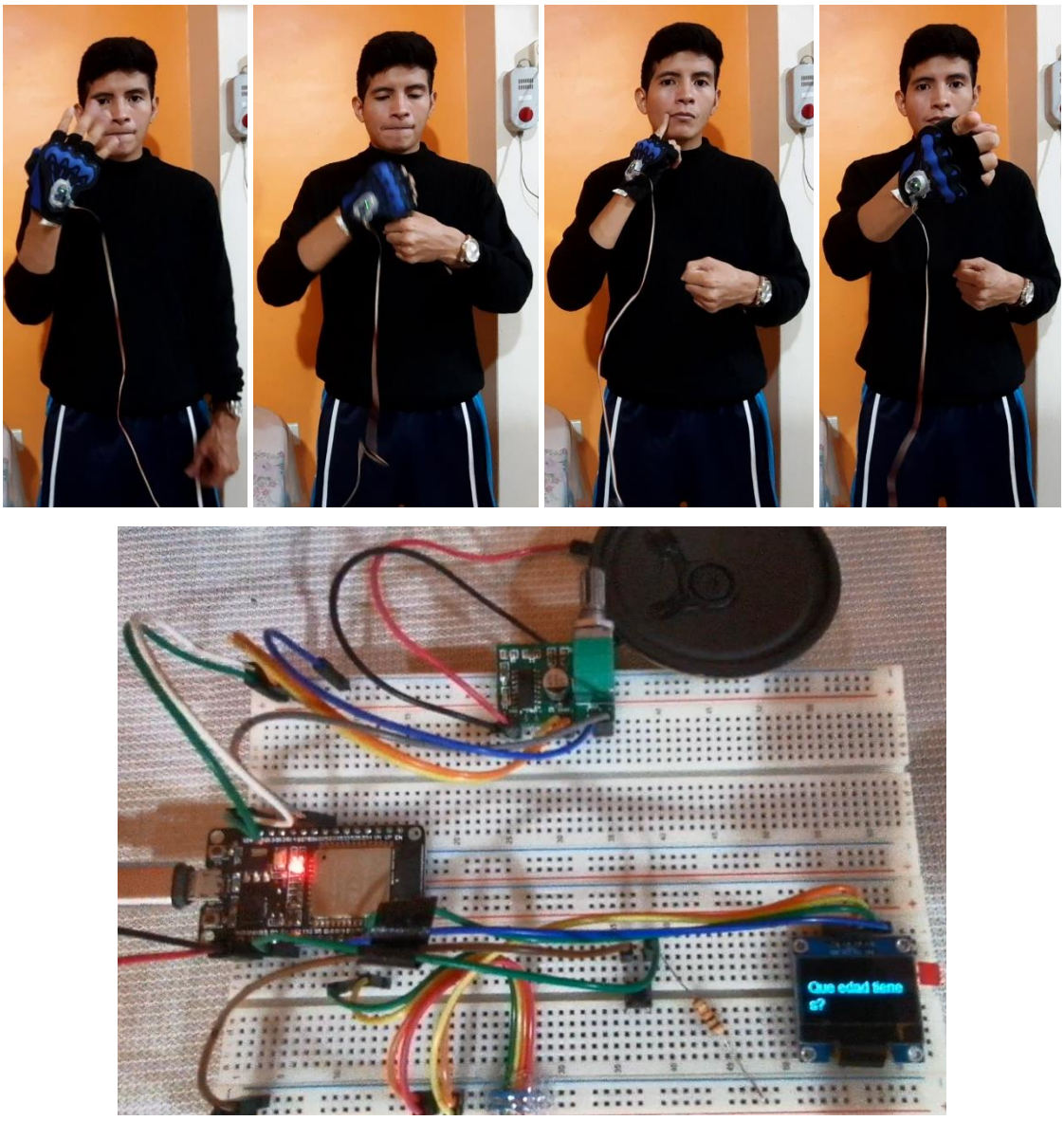


Figura 83: Realizando el gesto ¿Cuántos años tienes? G10

Fuente: Elaboración del Autor

3. CAPÍTULO III: EVALUACIÓN DEL PROTOTIPO

3.1. Plan de evaluación

Para evaluar los resultados obtenidos del proyecto se realizará un análisis comparativo de los modelos generados en el proceso de redes neuronales; también se realizarán pruebas de rendimiento del mejor modelo, que además incluye la precisión de la predicción y los resultados obtenidos al realizar los gestos.

3.1.1. Métricas de evaluación

Una de las métricas más importantes para evaluar el modelo son las métricas de clasificación, estas se inician a partir de la salida o clasificación final (resultado). Según [42], a partir del resultado podemos calcular varias métricas para determinar el rendimiento de nuestro modelo, considerando que el valor obtenido de cada clasificación permite crear una matriz de confusión (Tabla 7).

Una matriz de confusión nos permitirá evaluar los resultados obtenidos de nuestro modelo y determinar su rendimiento aplicando una serie de métricas de evaluación, entre las cuales utilizaremos las mencionadas por [43], las cuales son: las métricas de precisión, F-score, exactitud, sensibilidad y especificidad.

Tabla 7: Matriz de confusión

		Predicción de la Red Neuronal	
		Positivo	**Negativo**
Situación real	**Positivo**	Verdadero Positivo (VP)	Falso negativo (FN)
	Negativo	Falso Positivo (FP)	Verdadero Negativo (VN)

Fuente: Elaboración del Autor

Según [44], en una matriz de confusión o matriz de contingencia se ilustra el significado de VP, FN, FP y VN a los cuales se los describe como:

- Verdadero positivo (VP): Es el número de clases detectadas correctamente por el modelo. Este resultado en su totalidad nos indicaría que el modelo funciona correctamente.

- Falso negativo (FN): Es la cantidad de objetos que la red no ha detectado, es decir, son los objetos positivos erróneamente detectados como negativos.
- Falso positivo (FP): Son los objetos negativos detectados erróneamente como positivos, es decir, los objetos que no son de cierta clase, el modelo los clasifica como que si lo son (resultados clasificados con etiquetas diferentes).
- Verdadero Negativo (VN): Son los objetos negativos que el modelo clasificó correctamente.

Exactitud (Accuracy). Es una de las métricas más utilizadas para resumir el rendimiento del modelo de clasificación, esta nos permite determinar por medio de un porcentaje que tan bien clasifica el modelo.

$$Accuracy = \frac{VP + VN}{VP + FP + VN + FN} \quad (4)$$

Precisión. Con esta métrica podemos determinar el porcentaje de exactitud en los resultados obtenidos en la clasificación del modelo.

$$Precision = \frac{VP}{VP + FP} \quad (5)$$

Sensibilidad (Recall). Con esta métrica podemos determinar la efectividad del modelo en su clasificación.

$$Recall = \frac{VP}{VP + FN} \quad (6)$$

Valor de referencia (F-Score). Con esta métrica podemos tener una idea general del rendimiento del sistema.

$$F - score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (7)$$

Especificidad. Con esta métrica podemos determinar qué tan efectivo es el modelo identificando el porcentaje de valores identificados correctamente como negativos.

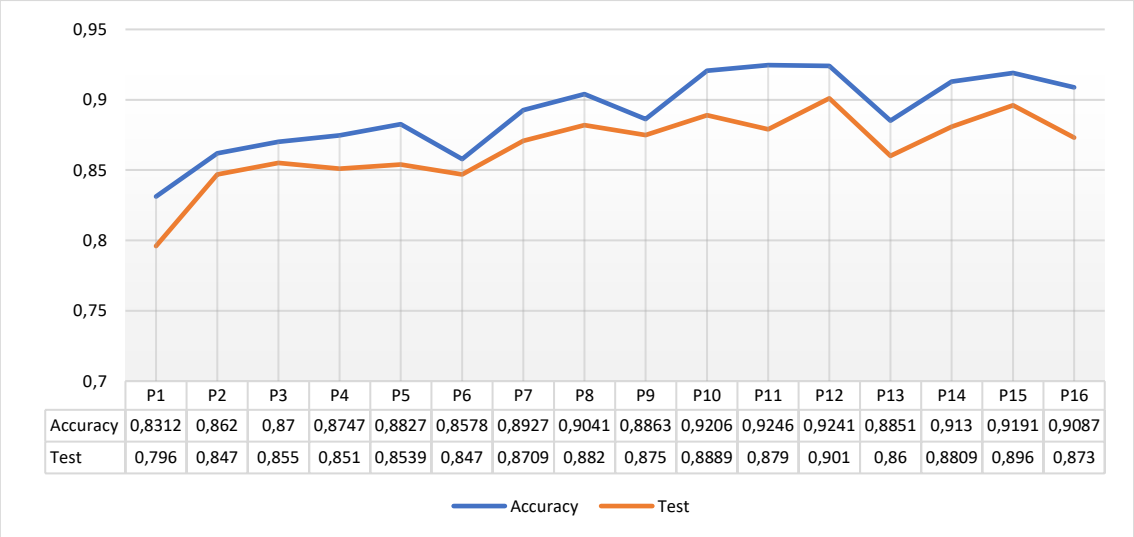
$$Specificity = \frac{VN}{VN + FP} \quad (8)$$

Otra de las pruebas a realizar es de Overfitting, las cuales se aplicarán a los modelos generados en el proceso de entrenamiento. Estas pruebas permitirán determinar si el modelo se encuentra con sobreajuste, es decir, si las muestra ingresadas en el entrenamiento carecen de datos, el modelo se ajustará a aprender los casos particulares que se enviaron en la data y será incapaz de reconocer nuevos datos, este es uno de los principales problemas en el aprendizaje automático.

3.2. Resultados de la evaluación

3.2.1. Pruebas de validación del modelo.

En el proceso de entrenamiento se realizaron 16 pruebas con el fin de generar un modelo con un porcentaje mayor en exactitud y en validación (prueba). Igualmente verificando que el mismo no contenga problemas de Overfitting. De los cuales, los mejores 3 modelos fueron comparados entre sí, para así seleccionar el mejor. En la Figura 84 se resumen las pruebas realizadas.



	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16
Accuracy	0,8312	0,862	0,87	0,8747	0,8827	0,8578	0,8927	0,9041	0,8863	0,9206	0,9246	0,9241	0,8851	0,913	0,9191	0,9087
Test	0,796	0,847	0,855	0,851	0,8539	0,847	0,8709	0,882	0,875	0,8889	0,879	0,901	0,86	0,8809	0,896	0,873

Figura 84: Resumen de los modelos generados en el proceso de Redes Neuronales

Fuente: Elaboración del Autor

Un punto importante a recalcar es que en todo el proceso de entrenamiento solo se tomó en consideración probar con 3 optimizadores, como lo son SGD, RMSprop y Adam; mientras que los otros híper-parámetros se cambiaron

constantemente hasta generar los siguientes modelos de la Tabla 8, los cuales obtuvieron mejores resultados.

Tabla 8: Mejores modelos generados en el proceso de Redes Neuronales

Híper-Parámetros	Prueba 1	Prueba 2	Prueba 3
Entorno de ejecución	GPU	GPU	GPU
Optimizador	Adam	RMSprop	SGD
Modelo	Secuencial	Secuencial	Secuencial
Número de iteraciones (Épocas)	1200	1000	1000
Tasa de aprendizaje	-	-	0,01
Neuronas ocultas	12	12	6
Número de capas ocultas	1	1	1
Resultados de entrenamiento			
Valor de pérdida (Loss)	0,2078	0,2322	0,3369
Valor de exactitud (Accuracy)	0,9241	0,9191	0,8700
Evaluar el modelo (Test accuracy)	0,9010	0,8960	0,8550
Overfitting	Figura 85		
Tiempo de ejecución	312,33 s	269,011 s	257,83 s

Fuente: Elaboración del Autor

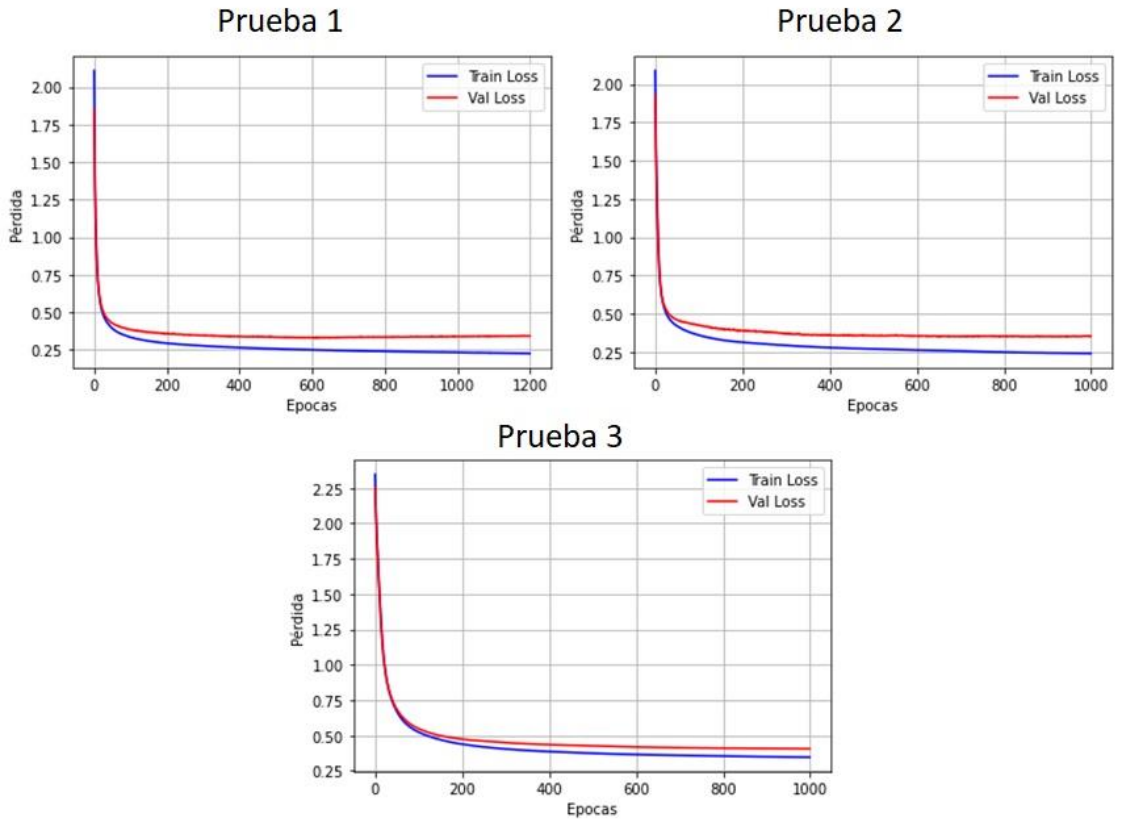


Figura 85: Pruebas para verificar problemas de Overfitting

Fuente: Elaboración del Autor

En la Figura 85 podemos observar que no se encuentran problemas considerables de Overfitting comparando las gráficas de perdida de entrenamiento y pérdida de validación, siendo óptimos para posteriormente extraer sus pesos e insertarlos en el sistema. Pero existe una ligera mejoría en la prueba 1 con el optimizador Adam, siendo este el modelo seleccionado.

3.2.2. Pruebas de rendimiento de la Red Neuronal.

Las pruebas de rendimiento del modelo se las realizó seleccionando un grupo de 11 individuos (previamente capacitados), los cuales utilizarán el sistema ejecutando los 10 gestos para comprobar su funcionamiento, finalizando con el análisis de los resultados obtenidos. La matriz de confusión mostró resultados muy óptimos para cada gesto a excepción de G1 y G2, los cuales, al guardar similitud entre ambos y además del G3 que de igual manera es similar al G2, este genera confusión en el modelo generando resultados poco favorables.

Tabla 9: Matriz de confusión del modelo

		Predicción del modelo										
		G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	**Total**
Test	G1	**5**	6	0	0	0	0	0	0	0	0	11
	G2	0	**6**	5	0	0	0	0	0	0	0	11
	G3	0	1	**10**	0	0	0	0	0	0	0	11
	G4	0	0	0	**9**	0	0	0	0	0	2	11
	G5	0	0	0	0	**11**	0	0	0	0	0	11
	G6	0	0	0	0	0	**10**	0	1	0	0	11
	G7	0	0	0	0	0	0	**11**	0	0	0	11
	G8	1	0	0	2	0	0	0	**8**	0	0	11
	G9	0	0	0	0	0	0	0	0	**11**	0	11
	G10	0	0	0	0	0	0	0	0	1	**10**	11
	Total	6	13	15	11	11	10	11	9	12	12	110

Fuente: Elaboración del Autor

Una vez generada la matriz de confusión con los resultados de las pruebas, se aplicaron cada una de las métricas de evaluación a cada uno de los gestos generando los siguientes resultados:

Tabla 10: Resultados aplicando las Métricas de evaluación a los gestos

	Total gestures	Correct	Precision	Accuracy	Specificity	Recall	f-score	acounting accuracy
G1	11	5	0,83	0,94	0,99	0,45	0,59	0,45
G2	11	6	0,46	0,89	0,93	0,55	0,50	0,55
G3	11	10	0,67	0,95	0,95	0,91	0,77	0,91
G4	11	9	0,82	0,96	0,98	0,82	0,82	0,82
G5	11	11	1,00	1,00	1,00	1,00	1,00	1,00
G6	11	10	1,00	0,99	1,00	0,91	0,95	0,91
G7	11	11	1,00	1,00	1,00	1,00	1,00	1,00
G8	11	8	0,89	0,96	0,99	0,73	0,80	0,73
G9	11	11	0,92	0,99	0,99	1,00	0,96	1,00
G10	11	10	0,83	0,97	0,98	0,91	0,87	0,91
Total	**110**	**91**	**0,84**	**0,96**	**0,98**	**0,82**	**0,82**	**0,82**

Fuente: Elaboración del Autor

Una vez concluida las pruebas de rendimiento del modelo podemos determinar resultados como la exactitud, especificidad, sensibilidad, valor de referencia y la precisión del modelo en el reconocimiento de cada gesto y del modelo en general, obtenido resultados superiores al 80% en cada una de las pruebas realizadas. Las pruebas en la que menor porcentaje se obtuvieron fueron las de Sensibilidad (recall), valor de referencia (f-score) y la exactitud contable (acounting accuracy) cuyo valor se obtuvo de la división entre la cantidad de aciertos sobre la cantidad de gestos (Correct / Total gestures).

Dicho anteriormente, los Gestos G1, G2 y G3 fueron de gran confusión para el modelo debido a su similitud, pero el G3 guarda una mayor precisión debido a que existe un patrón que lo distingue considerablemente del G1 y G2. También podemos rescatar que los gestos considerados simples fueron los que mejores resultados obtuvieron. Concluyendo que el modelo funciona perfectamente cuando los gestos contienen patrones simples y con mayor diferencia entre ellos.

3.3. Conclusiones

- Para el proyecto se logró el aprendizaje y la aplicación de técnicas de Inteligencia Artificial utilizando Machine Learning, aplicando algoritmos de Redes Neuronales Artificiales consiguiendo 91 aciertos de 110 gestos realizados y como resultado general se obtuvo una precisión del 84.2%, una exactitud de 96.5%, especificidad de 98.1%, una sensibilidad (recall) de 82.7% y un valor de referencia (f-score) de 82.5%.
- El proyecto contó con la implementación de la plataforma domótica Home Assistant y de dispositivos domóticos comerciales que por medio de herramientas de comunicación facilitaron la conexión entre ambos, demostrando ser una herramienta muy versátil.
- La integración de los dispositivos fue muy rápida y sencilla de realizar debido a que el mismo se encuentra compatible con una gama de dispositivos comerciales de diferentes fabricantes sin mencionar la capacidad de integrar dispositivos creados por cualquier usuario; también cuenta con una comunidad muy activa y de diversos Add-on que permiten una fácil interacción entre la plataforma, el dispositivo y el usuario.
- El proceso de redes neuronales artificiales concluyó con un modelo capaz de identificar un gesto previamente entrenado y de brindar un resultado de voz, texto y la automatización de dispositivos a través de MQTT en Hassio o en cualquier plataforma escogida por el usuario.
- Para generar estos resultados se logró con éxito la integración de varios componentes electrónicos, tales como: un ESP32 (cerebro del sistema), un amplificador digital, un display SSD1306 y principalmente un sensor MPU6050 capaz de obtener las señales de movimiento generadas por la mano utilizando únicamente señales de un acelerómetro de 3 ejes, además de otros componentes fundamentales para su comunicación.
- La integración de todos estos componentes y las técnicas de Inteligencia Artificial aplicadas, generó un prototipo capaz de traducir gestos para diferentes fines.
- Las pruebas realizadas a los modelos generados en el proceso de entrenamiento permitieron una rápida selección para su implementación en el sistema; mientras que las pruebas de rendimiento aplicando las

métricas de evaluación, obtuvieron resultados muy favorables para cada gesto a excepción de G1 y G2, los cuales, por similitud existieron problemas, pero en los gestos con patrones altamente diferentes no existieron inconvenientes. Finalizando con un modelo capaz de brindar una solución aceptable y de bajo costo (uso de un sensor y de componentes altamente accesibles) en el reconocimiento de gestos sujetos a un lenguaje de señas.

- Este proyecto utilizó gestos pertenecientes a la lengua de señas ecuatorianas para su manipulación, además de que el mismo puede ser consumido por personas de esta comunidad. Finalizando con un sistema capaz de traducir (voz y texto) saludos, preguntas y palabras capaces de automatizar un dispositivo.

3.4. Recomendaciones

- En este proyecto la instalación de Home Assistant fue realizada en una máquina Virtual por motivos de pruebas, pero a nivel profesional este método limita las funciones de esta potente herramienta recomendando una placa Raspberry Pi para su instalación.
- Para una mejor precisión en el reconocimiento de gestos es recomendable utilizar otros sensores además del MPU6050 y otros tipos de señales, además del acelerómetro. Entre estos pueden ser sensores Flex que facilitarían el reconocimiento e incluiría a la lista gestos estáticos, señales de giroscopio o EMG que aumentarían la cantidad de señales a procesar pudiendo generar mejores resultados en el reconocimiento.

REFERENCIAS BIBLIOGRÁFICAS

[1] J. Guerra, D. Vallejo-Huanga, N. Jaramillo, R. Macas, and D. Díaz, “Nondeterministic Finite Automata for Modeling an Ecuadorian Sign Language Interpreter,” *Adv. Intell. Syst. Comput.*, vol. 1213 AISC, pp. 369–376, 2021, doi: 10.1007/978-3-030-51328-3_51.

[2] P. Chinju, A. Ganesh, and C. Sunitha, “An overview of IoT based smart homes,” *Proc. 2nd Int. Conf. Inven. Syst. Control. ICISC 2018*, no. Icisc, pp. 43–46, 2018, doi: 10.1109/ICISC.2018.8398858.

[3] R. C. Parocha and E. Q. B. MacAbebe, “Implementation of home automation system using OpenHAB framework for heterogeneous IoT devices,” *Proc. - 2019 IEEE Int. Conf. Internet Things Intell. Syst. IoTaIS 2019*, pp. 67–73, 2019, doi: 10.1109/IoTaIS47347.2019.8980370.

[4] U. Singh and M. A. Ansari, “Smart Home Automation System Using Internet of Things,” in *2019 2nd International Conference on Power Energy Environment and Intelligent Control, PEEIC 2019*, 2019, pp. 144–149, doi: 10.1109/PEEIC47157.2019.8976842.

[5] B. Mazon-Olivo, D. Hernández-Rojas, J. Maza-Salinas, and A. Pan, “Rules engine and complex event processor in the context of internet of things for precision agriculture,” *Comput. Electron. Agric.*, vol. 154, no. September, pp. 347–360, 2018, doi: 10.1016/j.compag.2018.09.013.

[6] J. Berrú-ayala, D. Hernandez-rojas, and P. Morocho-díaz, “SCADA System Based on IoT for Intelligent Control of Banana Crop Irrigation,” *Int. Conf. Appl. Technol.*, vol. 1193, pp. 243–256, 2020, doi: 10.1007/978-3-030-42517-3.

[7] R. K. Kodali and S. Yerroju, “Energy Efficient Home Automation Using IoT,” in *2018 International Conference on Communication, Computing and Internet of Things (IC3IoT)*, 2018, pp. 151–154, doi: 10.1109 / IC3IoT.2018.8668155.

[8] J. J. Cartuche Calva, D. L. Hernández Rojas, R. F. Morocho Román, and C. D. Radicelli García, “Seguridad IoT: Principales amenazas en una

taxonomía de activos," *HAMUT'AY*, vol. 7, no. 3, pp. 51–59, 2021, doi: 10.21503/hamu.v7i3.2192.

[9] D. L. Hernández-Rojas, T. M. Fernández-Caramés, P. Fraga-Lamas, and C. J. Escudero, *Design and practical evaluation of a family of lightweight protocols for heterogeneous sensing through BLE beacons in IoT telemetry applications*, vol. 18, no. 1. Sensors, 2018.

[10] A. G. Ismaeel and M. Q. Kamal, "Worldwide auto-mobi: Arduino IoT home automation system for IR devices," *Int. Conf. Curr. Res. Comput. Sci. Inf. Technol. ICCIT 2017*, pp. 52–57, 2017, doi: 10.1109/CRCSIT.2017.7965533.

[11] S. Mahmud, S. Ahmed, and K. Shikder, "A smart home automation and metering system using internet of things (IoT)," *1st Int. Conf. Robot. Electr. Signal Process. Tech. ICREST 2019*, pp. 451–454, 2019, doi: 10.1109/ICREST.2019.8644232.

[12] W. A. Jabbar *et al.*, "Design and Fabrication of Smart Home with Internet of Things Enabled Automation System," *IEEE Access*, vol. 7, pp. 144059–144074, 2019, doi: 10.1109/ACCESS.2019.2942846.

[13] N. Vikram, K. S. Harish, M. S. Nihaal, R. Umesh, A. Shetty, and A. Kumar, "A low cost home automation system using wi-fi based wireless sensor network incorporating internet of things (IoT)," *Proc. - 7th IEEE Int. Adv. Comput. Conf. IACC 2017*, vol. 100, pp. 174–178, 2017, doi: 10.1109/IACC.2017.0048.

[14] H. Singh, V. Pallagani, V. Khandelwal, and U. Venkanna, "IoT based smart home automation system using sensor node," *Proc. 4th IEEE Int. Conf. Recent Adv. Inf. Technol. RAIT 2018*, pp. 1–5, 2018, doi: 10.1109/RAIT.2018.8389037.

[15] M. Al-Kuwari, A. Ramadan, Y. Ismael, L. Al-Sughair, A. Gastli, and M. Benammar, "Smart-home automation using IoT-based sensing and monitoring platform," *Proc. - 2018 IEEE 12th Int. Conf. Compat. Power Electron. Power Eng. CPE-POWERENG 2018*, pp. 1–6, 2018, doi: 10.1109/CPE.2018.8372548.

[16] Bharati Wukkadada ; Kirti Wankhede ; Ramith Nambiar ; Amala Nair, “Comparison with HTTP and MQTT In Internet of Things (IoT),” *2018 Int. Conf. Inven. Res. Comput. Appl. (ICIRCA 2018)*, pp. 249–253, 2018.

[17] S. Shapsough, F. Aloul, and I. A. Zualkernan, “Securing Low-Resource Edge Devices for IoT Systems,” *2018 Int. Symp. Sens. Instrum. IoT Era, ISSI 2018*, pp. 1–4, 2018, doi: 10.1109/ISSI.2018.8538135.

[18] A. M. Campoverde, D. L Hernández, and B. E. Mazón, “Cloud computing con herramientas open-source para Internet de las cosas,” vol. 6, pp. 173–182, 2015.

[19] D. L. Hernández-Rojas, T. M. Fernández-Caramés, P. Fraga-Lamas, and C. J. Escudero, *A plug-and-play human-centered virtual TEDS architecture for the web of things*, vol. 18, no. 7. Sensors, 2018.

[20] J. Novillo-Vicuña, D. Hernández-Rojas, B. Mazón-Olivo, J. Molina Rios, and O. Cárdenas Villavicencio, *Arduino y el Internet de las Cosas*, 1ra ed. 3Ciencias, 2018.

[21] K. Dokic, *Microcontrollers on the edge – is esp32 with camera ready for machine learning?*, vol. 12119 LNCS. Springer International Publishing, 2020.

[22] A. Jaramillo-Yánez, M. E. Benalcázar, and E. Mena-Maldonado, “Real-time hand gesture recognition using surface electromyography and machine learning: A systematic literature review,” *Sensors (Switzerland)*, vol. 20, no. 9, pp. 1–36, 2020, doi: 10.3390/s20092467.

[23] O. Barybin, E. Zaitseva, and V. Brazhnyi, “Testing the security ESP32 internet of things devices,” *2019 IEEE Int. Sci. Conf. Probl. Infocommunications Sci. Technol. PIC S T 2019 - Proc.*, pp. 143–146, 2019, doi: 10.1109/PICST47496.2019.9061269.

[24] A. Maier, A. Sharp, and Y. Vagapov, “Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things,” *2017 Internet Technol. Appl. ITA 2017 - Proc. 7th Int. Conf.*, pp. 143–148, 2017, doi: 10.1109/ITECHA.2017.8101926.

[25] P. Rai and M. Rehman, “ESP32 based smart surveillance system,” *2019 2nd Int. Conf. Comput. Math. Eng. Technol. iCoMET 2019*, pp. 1–3, 2019, doi: 10.1109/ICOMET.2019.8673463.

[26] A. Nelson *et al.*, “Wearable multi-sensor gesture recognition for paralysis patients,” *Proc. IEEE Sensors*, no. 1, pp. 1–4, 2013, doi: 10.1109/ICSENS.2013.6688531.

[27] P. G. Asteris and V. G. Mokos, “Concrete compressive strength using artificial neural networks,” *Neural Comput. Appl.*, vol. 32, no. 15, pp. 11807–11826, 2020, doi: 10.1007/s00521-019-04663-2.

[28] K. Dokic, M. Martinovic, and B. Radisic, “Neural Networks with ESP32-Are Two Heads Faster than One?,” *Proc. - 2020 6th Conf. Data Sci. Mach. Learn. Appl. CDMA 2020*, pp. 141–145, 2020, doi: 10.1109/CDMA47397.2020.00030.

[29] C. M. Correa Torres and J. F. Valencia Gómez, “Configuración de un Control de Temperatura en un Sistema Embebido de Bajo Costo, usando Herramientas de Inteligencia Artificial y el Internet de las Cosas,” *RISTI - Rev. Ibérica Sist. e Tecnol. Informação*, no. 34, pp. 68–84, 2019, doi: 10.17013/risti.34.68-84.

[30] C. E. Carbajal Peñaloza, A. J. Jiménez Alfaro, É. Organiche Corona, and N. E. Rodríguez Maya, “Nariz electrónica : Herramienta para detección de gases empleando redes neuronales artificiales .,” *Rev. Tecnol. Digit.*, vol. 8, no. 1, pp. 39–47, 2018.

[31] N. Shahid, T. Rappon, and W. Berta, “Applications of artificial neural networks in health care organizational decision-making: A scoping review,” *PLoS One*, vol. 14, no. 2, pp. 1–22, 2019, doi: 10.1371/journal.pone.0212356.

[32] P. H. Kuo and C. J. Huang, “A high precision artificial neural networks model for short-Term energy load forecasting,” *Energies*, vol. 11, no. 1, pp. 1–13, 2018, doi: 10.3390/en11010213.

[33] E. H. Galvis-Serrano, I. Sánchez-Galvis, N. Flórez, and S. Zabala-Vargas, “Clasificación de Gestos de la Lengua de Señas Colombiana a partir del

Análisis de Señales Electromiográficas utilizando Redes Neuronales Artificiales," *Inf. tecnológica*, vol. 30, no. 2, pp. 171–180, 2019, doi: 10.4067/s0718-07642019000200171.

[34] L. Wang, Z. Wang, H. Qu, and S. Liu, "Optimal Forecast Combination Based on Neural Networks for Time Series Forecasting," *Appl. Soft Comput. J.*, vol. 66, pp. 1–17, 2018, doi: 10.1016/j.asoc.2018.02.004.

[35] M. J. Cheok, Z. Omar, and M. H. Jaward, "A review of hand gesture and sign language recognition techniques," *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 1, pp. 131–153, 2019, doi: 10.1007/s13042-017-0705-5.

[36] FENASEC, *Diccionario oficial de lengua de señas Ecuatoriana*, 1ra ed. Quito: Federación Nacional de Personas Sordas del Ecuador (FENASEC), 2012.

[37] Vicepresidencia de la República del Ecuador, *Glosario Básico de Lengua de Señas Ecuatoriana*. Quito: Federación Nacional de Personas Sordas del Ecuador (FENASEC).

[38] D. Rivas *et al.*, "LeSigLa_EC: Learning sign language of Ecuador," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10676 LNCS, pp. 170–179, 2017, doi: 10.1007/978-3-319-71084-6_19.

[39] Riuson, "LCD Image Converter." https://lcd-image-converter.riuson.com/en/about/ (accessed Apr. 15, 2021).

[40] "Notevibes." https://notevibes.com/es/ (accessed Apr. 15, 2021).

[41] M. Hörz, "HxD - Freeware Hex Editor and Disk Editor | mh-nexus." https://mh-nexus.de/en/hxd/ (accessed Apr. 15, 2021).

[42] A. S. i Serrano and A. Lopez Peña, "YOLO Object Detector for Onboard Driving Images," *Univ. Auton. Barcelona*, 2017.

[43] S. Nofallah *et al.*, "Machine learning techniques for mitoses classification," *Comput. Med. Imaging Graph.*, vol. 87, no. November 2020, p. 101832, 2021, doi: 10.1016/j.compmedimag.2020.101832.

[44] M. Massiris, C. Delrieux, and J. Á. Fernández, “Detección de equipos de protección personal mediante red neuronal convolucional YOLO,” *Actas las XXXIX Jornadas Automática, Badajoz, 5-7 Septiembre 2018*, pp. 1022–1029, 2018, doi: 10.17979/spudc.9788497497565.1022.