Escuela Politécnica Superior de Linares

Grado en Ingeniería de Tecnologías de Telecomunicación

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Linares

CONTROL AUTOMÁTICO DE UNA VIVIENDA BASADO EN RASPBERRY PI

Alumno: Inmaculada Moreno Roldán

Tutor 1: Prof. D. José Fernando Rivas Peña

Tutor 2: Prof. D. Pedro J. Reche López

Dpto: Ingeniería de Telecomunicación

Junio, 2019

RESUMEN

Se llama domótica al conjunto de tecnologías aplicadas al control y la automatización inteligente de la vivienda, que permite una gestión eficiente del uso de la energía, que aporta seguridad y confort, además de comunicación entre el usuario y el sistema.

Con la llegada de los teléfonos inteligentes, el avance hacía el internet de las cosas está creciendo a gran velocidad. Cada vez más hay más objetos conectados a Internet que nos permiten realizar acciones cotidianas de forma remota y ahorrando tiempo.

Este proyecto tiene como objetivo presentar una plataforma web, que permite el control sobre los diferentes elementos electrónicos de bajo coste, junto a un prototipo de vivienda inteligente. Se desarrollaran temas como ahorro energético y confort, debido a los diferentes sensores implementados en el sistema para dar respuesta al usuario o situación determinada. También se abordará el tema de la seguridad en la vivienda, con sistemas de alertas y video vigilancia.

El conjunto de funciones que proporciona la combinación de hardware y software se le ha adjudicado un nombre comercial para referirse a este, Indovy (combinación de las palabras interior, domótica y vigilancia).

ABSTRACT

Home automation is the set of technologies applied to the control and smart home automation system, which allows efficient management of the use of energy, which provides security and comfort, as well as communication between the user and the system.

With the arrival of smartphones, the advance towards the internet of things is growing at great speed. More and more objects are connected to the Internet that allow us to perform daily actions remotely and saving time.

The objective of this project is to present a web platform, which allows the control over the different low cost electronic elements, together with a smart housing prototype. Topics such as energy saving and comfort will be developed, due to the different sensors implemented in the system to respond to the user or specific situations. It will also treat the issue of security in housing, with alert systems and video surveillance.

The set of functions provided by the combination of hardware and software has been assigned a trade name to refer to this, Indovy (combination of the words interior, home automation and surveillance).

ÍNDICE

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

Hace 30 años con el uso de la información a través de buscadores de internet empezó una nueva "revolución industrial", dando lugar a tener a nuestra disposición millones de datos a través de contenido multimedia. El paso del tiempo y el avance de nuevas tecnologías permitieron transportar grandes cantidades de datos, creándose el concepto de Big Data y evolucionando hacía el Internet de las cosas (IoT), término con el cual podemos afirmar que se llegará a un momento en el que se conectarán a Internet más "cosas u objetos" que personas.

Dicho esto, se diferencian tres etapas en el Internet de las cosas [1]:

1. **Tecnología embebida:** computadoras en dispositivos que, mediante controladores, se podía acceder a sus datos.
2. **Masificación de la conectividad y el cloud computing:** dispositivos que toman y envían información desde la nube.
3. **Big Data y herramientas de analítica:** agrega inteligencia a los datos.

Diferentes estudios aseguran que el IoT es la próxima transformación tecnológica. En el 2015 alrededor de 10 mil millones de dispositivos se conectaron a Internet, mientras que para el 2020, Gartner, una de las principales empresas de tecnología de la información en el mundo, prevé que habrá 34 mil millones de dispositivos, de los cuales, se estima que 26 mil millones representarán al ecosistema IoT [2]. Por lo que, el protocolo IPv6 aporta grandes beneficios a la hora de desarrollar IoT, entre ellos la cantidad de dispositivos que se pueden conectar a la red.

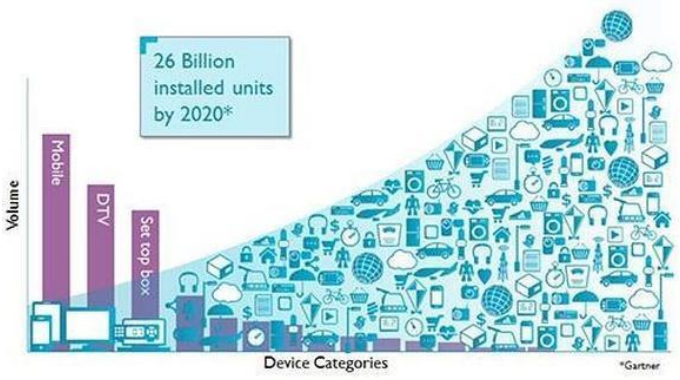


Figura 1. Gráfica de la evolución del IoT en un futuro.

1.1. Introducción al IoT (Internet de las cosas)

Según la RAE, se define como “Conjunto de sistemas que automatizan las diferentes instalaciones de una vivienda”, es decir, una vivienda inteligente que responde ante las necesidades de los inquilinos que viven en ella. Para ello el sistema se ayuda de unos sensores que recogen la información necesaria para dar una respuesta.

El concepto de Internet de las cosas fue propuesto por Kevin Ashton en el Auto-ID Center del MIT en 1999, donde se realizaban investigaciones en el campo de la identificación por radiofrecuencia en red (RFID) y tecnologías de sensores.

Por ejemplo, si los libros, termostatos, refrigeradores, la paquetería, lámparas, botiquines, partes automotrices, entre otros estuvieran conectados a Internet y equipados con dispositivos de identificación, no existirían, en teoría, artículos fuera de stock o medicinas caducadas; se sabría exactamente la ubicación, cómo se consumen en el mundo; el extravío sería cosa del pasado y se sabría qué está encendido o apagado en todo momento [3].

Gracias al sistema RFID (siglas de radio frequency identification) bastará con integrar un chip de pocos milímetros en cualquier objeto del hogar, del trabajo o de la ciudad para poder procesar y transmitir información a partir de él constantemente. Cada uno de los objetos conectados a Internet tiene una IP específica y mediante esa IP puede ser accedido pare recibir instrucciones. Así mismo, puede contactar con un servidor externo y enviar los datos que recoja.

Abi Research, asegura que para el mismo año existirán 30 mil millones de dispositivos inalámbricos conectados a Internet. Con la próxima generación de aplicaciones de Internet (protocolo IPv6) se podrían identificar todos los objetos, algo que no se podía hacer con IPv4.

Según los expertos existirán al menos 4 niveles de inteligencia de los dispositivos conectados a Internet:

- **Nivel 1: Identidad**. El objeto será capaz de identificarse de manera única
- **Nivel 2: Ubicación**. Se podrá saber dónde está dicho objeto o dónde ha estado.
- **Nivel 3: Estado**. Será capaz de comunicar el estado en que se encuentra así como sus características.
- **Nivel 4: Contexto**. El objeto será capaz de percibir el entorno en que se encuentra.
- **Nivel 5: Criterio**. El objeto se comunica, se identifica, se ubica, analiza su entorno, decide y ejecuta en función de su criterio. [4]

A continuación, se enumeran una serie de ventajas que ofrecen esta tecnología:

- **Seguridad:** se refiere a sistemas de alarma y cámaras de video vigilancia, que te avisa cuando detecta un intruso en la vivienda, mediante sensores de presencia, por ejemplo. O avisos de incidencias que ocurren en la vivienda, como escapes de gas.
- **Comodidad:** proporcionar una vida más confortable haciendo ciertas tareas más fáciles al usuario y útil para personas con discapacidad. Destacar también que la automatización de ciertas tareas conlleva al ahorro en tiempo del inquilino.
- **Consumo eficiente:** se ahorra energía gracias al apagado automático de las luces, gestión de temperatura de la vivienda, apagado de electrodomésticos a distancia, etc.
- **Nuevo modelo de negocio:** se crean nuevas empresas que ofrecen la implementación de nuevos servicios. Empresas como Cisco System y Ericson por nombrar algunas, están trabajando e impulsando fuertemente en el concepto IoT y muchas ya diseñan productos específicos para gestionar la interconexión de los objetos.

Por otra parte, nos lleva a mencionar algunas de las desventajas:

- **Privacidad:** Para que el sistema sea eficiente, se debe de observar los hábitos del consumidor en todos sus aspectos y niveles.
- **Infraestructura:** los inmuebles se encarecen debido a que aumenta la complejidad de construcción.
- **Economía:** la parte económica es una los interrogantes más importantes debido a que se requiere de inversión tecnológica.
- **Protocolo:** la transición de un protocolo que unifique las comunicaciones IoT se está desarrollando lentamente. [5][6]

Por último, hacer mención a la seguridad de esta tecnología. La empresa Hewlett Packard realizó un estudio en 2015, reportando que entre otros hallazgos respecto a los dispositivos IoT, el 70% de ellos tiene vulnerabilidades de seguridad en sus contraseñas, hay problema con el cifrado de los datos o los permisos de acceso, y el 50% de las aplicaciones de dispositivos móviles no encriptan las comunicaciones. La firma de seguridad Kaspersky Lab realizó pruebas en objetos conectados al IoT y encontró que una cámara monitor para bebé podía hackearse para interceptar el vídeo y en una cafetera que transmitía información sin encriptar, se podía conocer la contraseña de la red WiFi en donde estuviera conectada. Dado esto, la seguridad se ha convertido en una situación de mucha importancia.

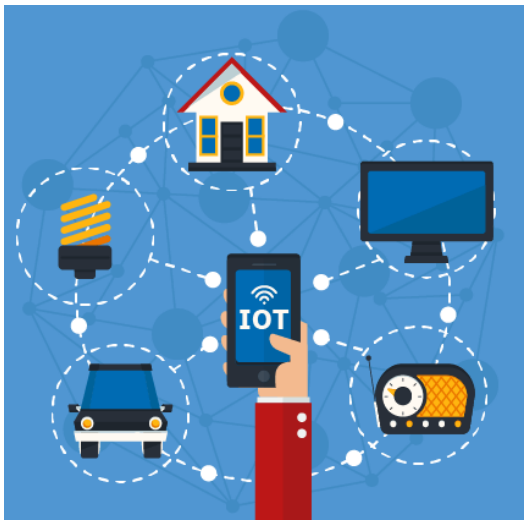


Figura 2. Ilustración que representa la tecnología IoT.

1.2. Objetivos

El objetivo de este proyecto es realizar una maqueta de una vivienda inteligente donde poder demostrar el funcionamiento de los distintos sensores implementados y poder controlar los distintos dispositivos electrónicos del hogar a través de una plataforma web.

Los puntos a tratar más importantes son:

- **Control**: poder manejar diferentes aparatos electrónicos y eventos de la vivienda tanto por reconocimiento de voz como por la plataforma web.
- **Seguridad**: observar en tiempo real lo que ocurre en la vivienda y mediante Telegram mandar avisos de eventos que puedan ser de carácter peligroso.
- **Automatización**: sistemas que por sí solos están programados para dar una respuesta al inquilino.

Toda la información que recoge los sensores será enviada un servidor, implementado en una raspberry pi, para demostrar que es posible que en cualquier dispositivo se puede controlar la gestión de la vivienda en remoto, de esto saldrá una demo de un producto, el cual se denomina, "Indovy". Finalmente, se muestra la figura 3 el resultado final que se obtiene y el cual se explicará a lo largo de los capítulos de esta memoria.

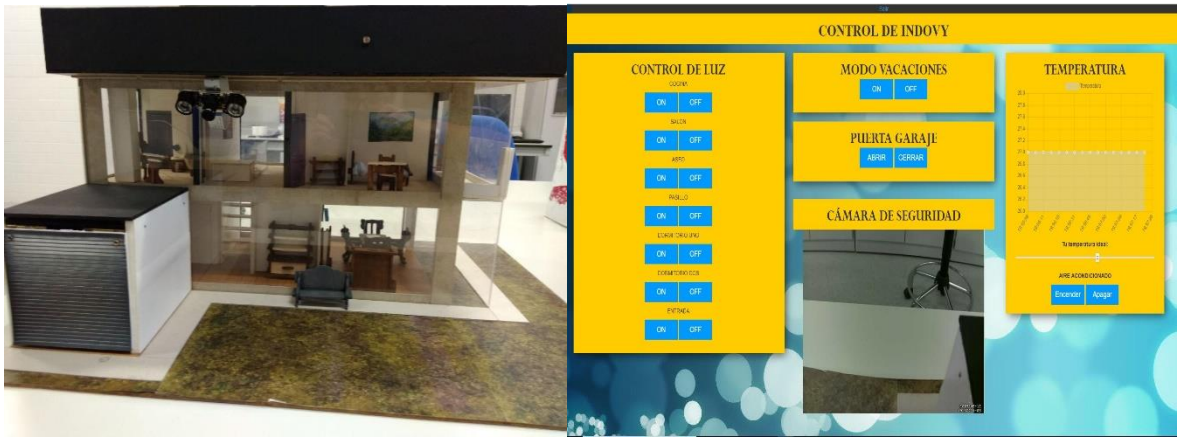


Figura 3. Prototipo final de una vivienda inteligente basado en raspberry pi 3.

2. MARCO TEÓRICO. HARDWARE.

En este capítulo se va a analizar cada uno de los aspectos teóricos del hardware utilizado en este proyecto para facilitar así una mejor comprensión del mismo.

Se expondrá las características y el funcionamiento de los distintos dispositivos utilizados en este prototipo, sensores, micrófono, cámara y por supuesto, el núcleo de este proyecto, Raspberry pi.

2.1. Raspberry pi

Raspberry Pi es un computador de placa reducida, aproximadamente del tamaño de una tarjeta de crédito, de bajo costo desarrollado en Reino Unido por la Fundación Raspberry Pi, con el objetivo de estimular la enseñanza de ciencias de la computación en las escuelas. En la figura 4 se muestra la raspberry pi 3 modelo B, con la cual se desarrollará el proyecto.

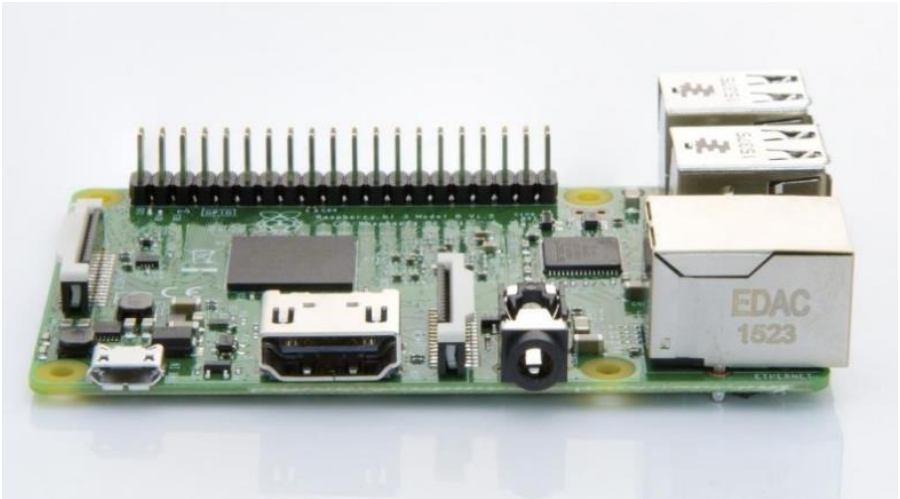


Figura 4. Raspberry Pi 3 modelo B

Actualmente en el mercado se encuentra varios modelos. A continuación, se muestra en la figura 5 las características de las distintas versiones y en donde se aprecia la evolución de dicha computadora.[7]

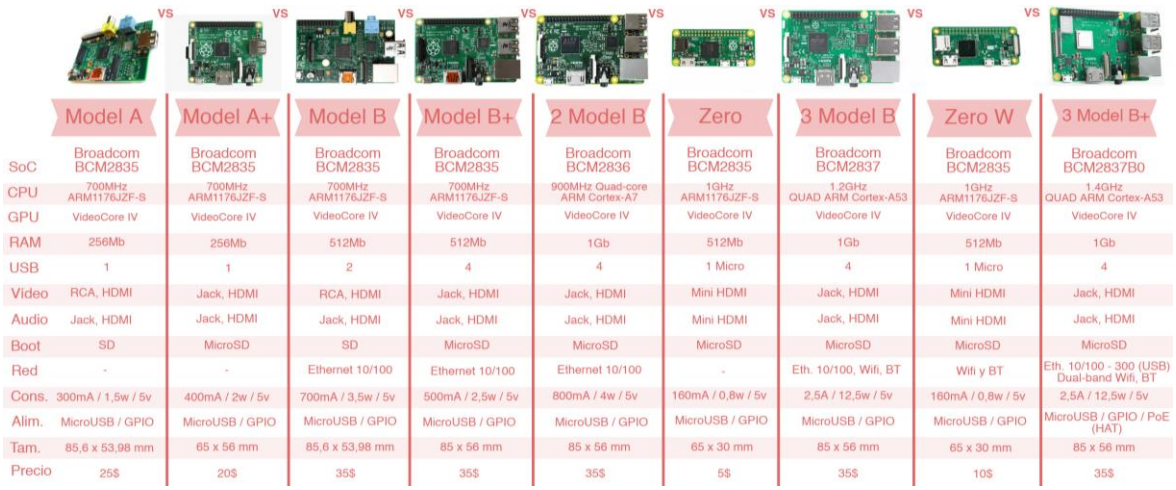


Figura 5. Comparativa entre los distintos modelos de Raspberry Pi

2.1.1. *Partes de la Raspberry Pi 3*

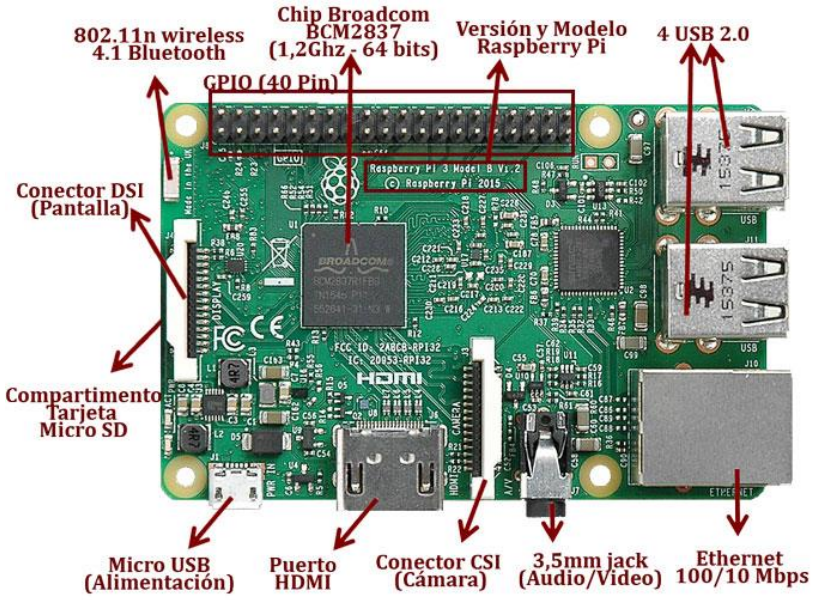


Figura 6. Partes de la Raspberry Pi 3 modelo B

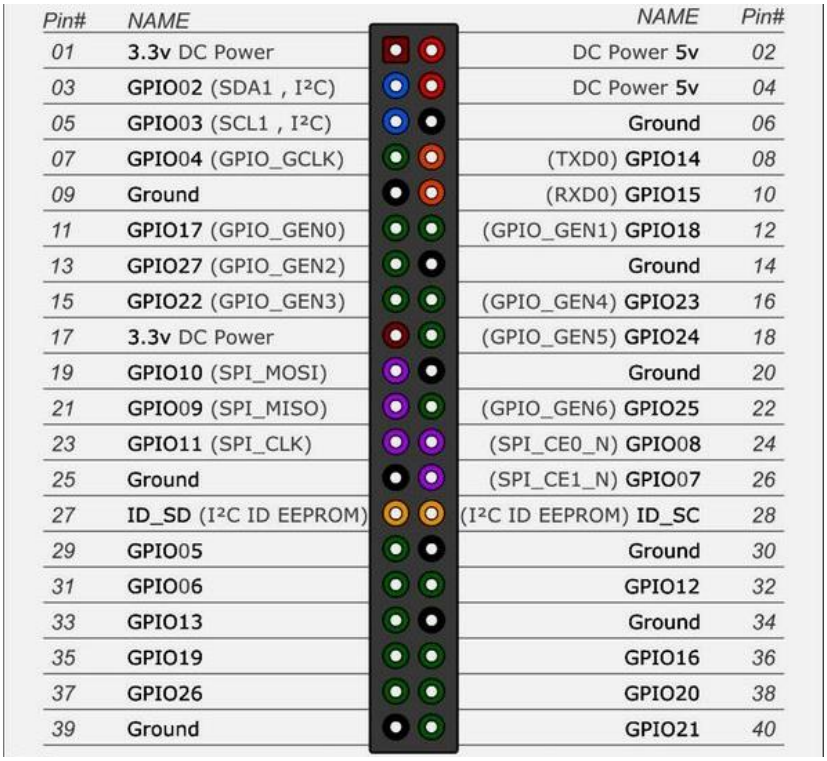
Para conocer más el núcleo de este proyecto se va a hacer un recorrido por la placa, describiendo los distintos elementos que se indican la figura 6.

En el centro de todas las placas Raspberry Pi se encuentra un semiconductor cuadrado, más comúnmente conocido como circuito integrado o chip. Este es el módulo system-on-chip (SoC) Broadcom BCM2837, es el encargado de proporcionarle a la Raspberry Pi sus capacidades de procesamiento de propósito general, de renderización de

gráficos y de entrada/salida. Apilado sobre este chip se encuentra otro semiconductor, este es el encargado de proveer la memoria a la Raspberry Pi para el almacenamiento temporal de datos mientras sus programas se encuentran en ejecución. Este tipo de memoria es conocida como memoria RAM (memoria de acceso aleatorio), ya que la computadora puede leer o escribir en cualquier parte de la memoria y en cualquier momento. La RAM es volátil, lo que significa que cualquier cosa almacenada en esta memoria se eliminará cuando la Raspberry Pi sea desenchufada.

Por la parte de abajo del SoC se encuentran las salidas de video de la Raspberry Pi. El conector plata es un puerto HDMI (High Definition Multimedia Interface), es del mismo tipo de conector que se puede encontrar en los reproductores de video y muchos decodificadores de cable y satélite. Cuando es conectado a un televisor o monitor moderno, el puerto HDMI proporciona video en alta resolución y audio digital. El conector Jack de 3.5 mm es un puerto de video compuesto y está diseñado para conectarse a televisores antiguos que no cuentan con conexión HDMI. La calidad del video es más baja que la que está disponible a través del HDMI y también transmite audio; como una señal analógica.

Los pines que se observan en la parte superior izquierda son el header o conector con entradas-salidas de propósito general (GPIO) que puede utilizarse para conectar la Raspberry Pi a otros hardwares. El uso más frecuente de este puerto es para conectar una placa de expansión (Add-On). El puerto GPIO es extremadamente poderoso, pero también frágil; hay que evitar conectar nada en ellos mientras la Raspberry Pi se encuentra encendida.



Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I²C)	DC Power 5v	04
05	GPIO03 (SCL1 , I²C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I²C ID EEPROM)	(I²C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Figura 7. Header de la Raspberry pi 3 modelo B

Los pines no disponen de protección, por lo que hay que tener cuidado con los voltajes e intensidades a la hora de conectar sensores a ellos, siempre se debe colocar una resistencia para no dañar la placa. Como se observa en la figura 7 no todos los pines tienen la misma función:

- **Pines de alimentación:** hay tanto de 5V como de 3.3 V, limitados a 50 mA. Y por tanto, también hay toma de tierra (GND).
- **GPIO normales:** son conexiones configurables, las cuáles se pueden programar para las distintas funciones que se necesitan en el proyecto.
- **GPIO especiales:** se encuentran algunos pines destinados a una interfaz UART, con conexiones TXD y RXD que sirven para comunicaciones en serie, como por ejemplo, conectar con una placa Arduino. También se puede ver en la figura 7 otros como SDA, SCL, MOSI, MISO, SCLK, CE0, CE1, etc.

El conector de plástico y metal debajo del puerto GPIO es el puerto DSI (Display Serial Interface), que sirve para conectar sistemas de pantalla plana controlada digitalmente. Estas pantallas rara vez son utilizadas excepto por los desarrolladores profesionales en sistemas embebidos, ya que el puerto HDMI es más flexible.

Un segundo conector de plástico y metal, se encuentra a la derecha del puerto HDMI, es el puerto CSI (Camera Serial Interface), que proporciona una conexión de alta velocidad para el módulo de cámara de la Raspberry Pi u otros sistemas de cámaras con conexión CSI compatibles con la Raspberry Pi.

En la esquina inferior de la placa está la toma de alimentación. Esta es una toma micro-USB. La conexión de un cable micro-USB a un adaptador de energía adecuado encenderá la Raspberry Pi; a diferencia de un PC, la Raspberry Pi no cuenta con un botón de encendido y arranca inmediatamente cuando el cable de la alimentación es conectado.

En la parte inferior de la placa, a mano izquierda, se encuentra una ranura para tarjetas micro-SD. Una tarjeta de memoria micro-SD (Secure Digital) proporciona el almacenamiento para el sistema operativo, los programas, datos y otros archivos, y no es volátil; a diferencia de la RAM, la memoria SD conservará su información incluso cuando la energía se pierda.

Al borde, a mano derecha hay diferentes conectores dependiendo de qué modelo de Raspberry Pi, el Modelo A o el Modelo B. En el modelo de la figura 6 se muestra el puerto Ethernet de 100/10 Mbps y 4 conectores USB 2.0.

Arriba de éstos se encuentran una serie de LEDs (Light Emitting Diodes). Los dos primeros LEDs que están etiquetados con ACT y PWR proporcionan una notificación de actividad y de energía respectivamente, y están presentes en todas las placas.

Por último, indicar el módulo wifi 802.11n y el módulo bluetooth 4.1, los cuales hacen ventajoso usar este modelo para el desarrollo de este proyecto, ya que reducimos cableado. La voy a [8]

2.1. Sensores

En ese apartado se explicará más detenidamente las características y funcionalidades de los distintos sensores utilizados en la maqueta.

2.1.1. Sensor infrarrojo de movimiento PIR HC-SR501

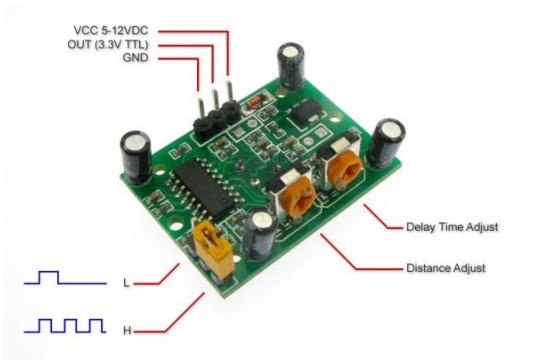


Figura 8. Partes de PIR HC-SR501. Sensor infrarrojo de movimiento.

El módulo PIR modelo HC-SR501 es de bajo costo, pequeño, e incorpora la tecnología más reciente en sensores de movimiento. El sensor utiliza 2 potenciómetros y un jumper que permiten modificar sus parámetros y adaptarlo a las necesidades de la aplicación: sensibilidad de detección, tiempo de activación y respuesta ante detecciones repetitivas.

Sus especificaciones técnicas más importantes son [9]:

- Usa el PIR LHI778 y el controlador BISS0001.
- Voltaje de alimentación: de 5 a 12 VDC.
- Consumo promedio: <1 miliamperio.
- Rango de distancia de 3 a 7 metros ajustable.
- Angulo de detección: cono de 110°.
- Ajustes: 2 potenciómetros para ajuste de rango de detección y tiempo de alarma activa.

- Jumper para configurar la salida de alarma en modo mono-disparo ó disparo repetitivo.
- Salida de alarma de movimiento con ajuste de tiempo entre 3 segundos a 5 minutos.
- Salida de alarma activa Vo con nivel alto de 3.3 volts y 5 mA, lista para conexión de un led, ó un transistor, etc.
- Tiempo de inicialización: después de alimentar el módulo HC-SR05, debe transcurrir 1 minuto antes de que inicie su operación normal. Durante ese tiempo, es posible que el módulo active 2 ó 3 veces su salida.
- Tiempo de salida inactiva: cada vez que la salida pase de activa a inactiva, permanecerá en ese estado los siguientes 3 segundos. Cualquier evento que ocurra durante ese lapso es ignorado.
- Temperatura de operación: -15° a +70° C.
- Dimensiones: 3.2 x 2.4 x 1.8 cm.

Este sensor tiene varias aplicaciones, las más comunes son para productos de seguridad, iluminación automática, automatización y control industrial, puertas y timbres automáticos, juguetes, etc.

A continuación, se describe los principios de funcionamiento de este sensor.

La radiación infrarroja: Todos los seres vivos e incluso los objetos, emiten radiación electromagnética infrarroja, debido a la temperatura a la que se encuentran. A mayor temperatura, la radiación aumenta. Esta característica ha dado lugar al diseño de sensores de infrarrojo pasivos, en una longitud de onda alrededor de los 9.4 micrones, los cuales permiten la detección de movimiento, típicamente de seres humanos o animales. Estos sensores son conocidos como PIR, y toman su nombre de 'Pyroelectric Infrared' o 'Passive Infrared'.

El lente de Fresnel: El lente de Fresnel es un encapsulado semiesférico hecho de polietileno de alta densidad cuyo objetivo es permitir el paso de la radiación infrarroja en el rango de los 8 y 14 micrones. El lente detecta radiación en un ángulo con apertura de 110° y, adicionalmente, concentra la energía en la superficie de detección del sensor PIR, permitiendo una mayor sensibilidad del dispositivo.

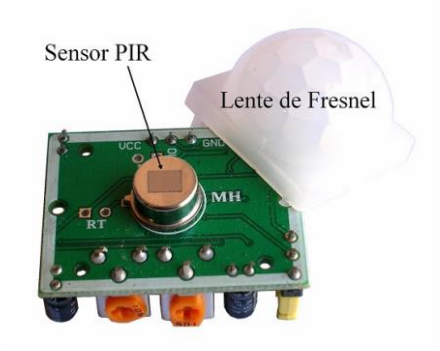


Figura 9. Lente de Fresnel y Sensor PIR.

En los sensores de movimiento, el sensor PIR consta en realidad de 2 elementos detectores separados, siendo la señal diferencial entre ambos la que permite activar la alarma de movimiento. En el caso del HC-SR501, la señal generada por el sensor ingresa al circuito integrado BISS0001, el cual contiene amplificadores operacionales e interfaces electrónicas adicionales. Las funciones y ajustes complementarios del sensor de movimiento son:

- **Ajuste de parámetros:** mediante 2 potenciómetros, el usuario puede modificar tanto la sensibilidad como la distancia de detección del PIR.
- **Detección automática de luz** (esta función no está disponible al adquirir el sensor de fábrica)**:** por medio de una foto resistencia CdS (Sulfuro de Cadmio), se deshabilita la operación del sensor en caso que exista suficiente luz visible en el área. Esta función es utilizada en caso de sensores que enciendan lámparas en lugares poco iluminados durante la noche, y especialmente en corredores o escaleras.

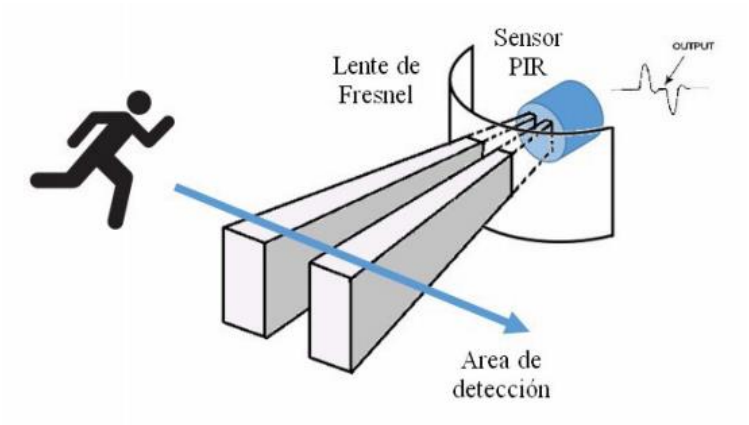


Figura 10. Ilustración del área de detección del sensor PIR

Rango de detección de los sensores PIR: Como se indicó anteriormente, el rango de detección de movimiento de los PIR es ajustable y generalmente funcionan con alcances de hasta 7 metros, y con aperturas de 90° a 110°, como se muestra en la figura 11. El montaje del PIR puede realizarse tanto en piso, muro ó techo, según convenga a la aplicación.

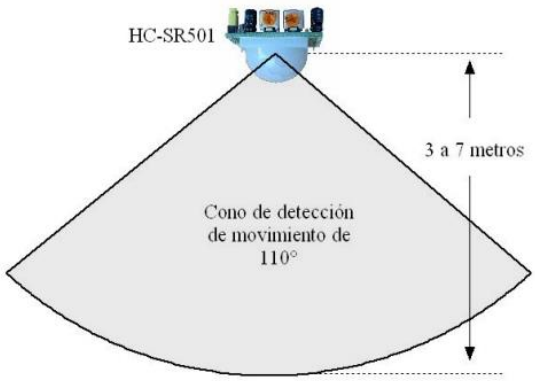


Figura 11. Rango de detección de los sensores PIR

Por último, se describe los ajustes y configuraciones que puede tener el sensor de este apartado.

Potenciómetros: el usuario puede ajustar tanto el tiempo de disparo de la señal de alarma de movimiento, como la distancia de detección. Los potenciómetros correspondientes deben girarse en la dirección mostrada en la figura 12 para realizar los ajustes.

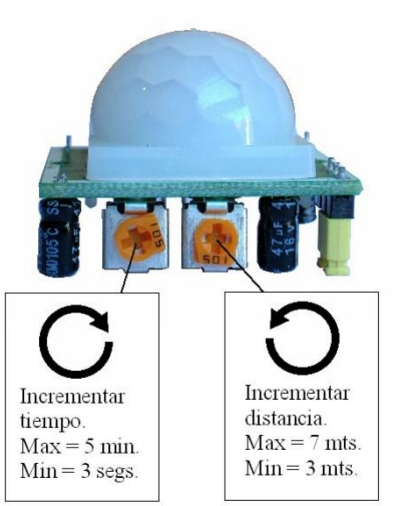


Figura 12. Función de los distintos potenciómetros que lo componen

Posición del jumper: De acuerdo a la figura 8, el usuario puede trabajar en 2 modos de operación:

- **1 solo disparo:** en este modo, cuando ocurre una detección de movimiento (el cual se identificará como 'evento'), la salida del sensor se activa durante el tiempo que se haya ajustado a través del potenciómetro correspondiente. Para efectos de ejemplo, supongamos que el tiempo de activación es de 60 segundos. Si durante esos 60 segundos ocurre un segundo evento, éste no será considerado.
- **Disparos repetitivos:** en este modo, cada evento detectado genera un nuevo tiempo de activación. Volviendo al ejemplo de tiempo de 60 segundos. Cuando ocurre el primer evento, la salida se activa. Si transcurridos 30 segundos ocurre un segundo evento, entonces se sumarán 60 segundos al tiempo transcurrido, dando un total de 90 segundos continuos con la salida activa. Y así, cada evento adicional, sumará un tiempo de 60 segundos de activación al tiempo ya transcurrido.

En cualquier caso, si la salida regresa a su estado inactivo, habrá un lapso de 3 segundos durante los cuales los nuevos eventos no serán considerados. Pasados esos 3 segundos, el dispositivo regresa a su funcionamiento normal [10].

2.1.2. Sensor de ultrasonido HC-SR04

Figura 13. Sensor de ultrasonido HC-SR04

Este sensor HC-SR04 es muy útil para medir distancias y detectar obstáculos, basándose en el principio de funcionamiento del sonar.

Se enumera alguna de las características que se pueden encontrar en su correspondiente datasheet [11]:

- Voltaje de alimentación: +5VDC
- Corriente en espera: <2 mA
- Corriente de trabajo: 15 mA
- Ángulo eficaz: <15°
- Rangos de distancia: 2 cm a 400 cm
- Resolución: 0.3 cm
- Ángulo de medida: 30°
- Ancho de pulso de disparo (Trigger Input Pulse Width): 10 µs
- Eco (Echo): salida del sensor.
- Frecuencia de ultrasonido: 40 KHz

Tal y como se muestra en la figura 13, hay cuatro pines de conexión.

- **VCC:** para alimentación de +5VDC
- **Trig:** entrada del pulso de disparo para iniciar la medición.
- **Echo:** retorna un pulso proporcional a la distancia que rebota el sonido.
- **GND:** toma de tierra

Como se observa en la figura 13, está formado por dos cilindros, uno de ellos es el emisor y el otro es el receptor. Su operación no es afectada por la luz del sol o materiales oscuros, aunque materiales acústicamente blandos son difíciles de detectar, por ejemplo las telas.

Su funcionamiento se basa en enviar un pulso de alta frecuencia inaudible por el ser humano de al menos 10 microsegundos por el pin Trig. El sensor enviará 8 pulsos de 40 KHz y coloca su salida Echo a alto, se debe detectar este evento e iniciar un conteo de tiempo. La salida Echo se mantendrá en alto hasta recibir el echo reflejado por el obstáculo a lo cual el sensor pondrá su pin de Echo a nivel bajo, es decir, termina de contar el tiempo, lo comentado se representa en la figura 14, para su mejor entendimiento. Se obtiene así el tiempo que ha tardado el pulso en llegar al sensor de nuevo. Por último, se recomienda dar un tiempo de aproximadamente 50 ms de espera después de terminar la cuenta. [12]

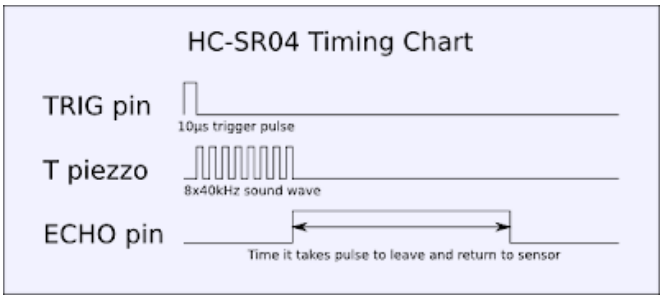


Figura 14. Tabla de tiempos del sensor HC-SR04

Conociendo la velocidad del sonido en el aire, mediante la ecuación (1), donde γ es la constante adiabática, R es la constante de gas, M es la masa molecular del gas y T la temperatura en ºC obtenida del sensor de temperatura DTH11 en ese momento.

$$\upsilon \frac{m}{s} = \sqrt{\frac{\gamma RT}{M}} = \sqrt{\frac{1.4 * 8.34 \frac{J}{kg\ mol} * T}{29 \frac{kg}{mol}}} \quad (1)$$

Se puede obtener la distancia, mediante las ecuaciones (2) y (3):

$$\upsilon\ \frac{m}{s} * 100\ \frac{cm}{m} * \frac{1}{1000000}\frac{s}{\mu s} = \frac{1}{29.2}\frac{cm}{\mu s}\ \ (3)$$

$$\text{Distancia (cm)} = \frac{Tiempo\ (\mu s)}{29.2*2}\ (3)$$

Hay que dividir el tiempo por dos porque hemos medido el tiempo que tarda en ir y volver la señal sonora, por lo que la distancia recorrida es el doble, como se muestra en la figura 15.

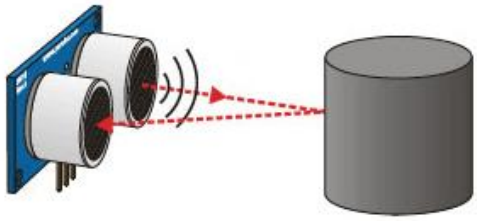


Figura 15. Recorrido que hace la señal sonora cuando se refleja en un obstáculo.

El rango de medición teórico es de 2cm a 400 cm, con una resolución de 0.3 cm. En la práctica, sin embargo, el rango de medición real es mucho más limitado, en torno a 20 cm a 2 metros.

Es un sensor de baja precisión, ya que en una habitación con un gran número de objetos, el sonido rebota en las superficies generando ecos y falsas mediciones. Tampoco es adecuado para funcionar en exteriores.

Si se necesita mayor precisión, se debe usar medidores de distancia por infrarrojos y sensores ópticos. [13]

2.1.3. Servo motor SG90

Figura 16. Servo motor SG90

El servo SG90 es uno de los servomotores más versátiles y usado en todo tipo de proyectos de robótica. Es muy pequeño pero aun así ofrece una fuerza de 1.8 Kg/cm, así que es válido para todo tipo de robots bípedos o para desplazar diversas piezas motrices.

Es un motor eléctrico que nos permite mantener la posición que indiquemos, siempre que esté dentro del rango de operación del dispositivo. Por otro lado nos permite también controlar la velocidad de giro.

A continuación, se enumeran algunas de sus características principales [14]:

- Velocidad: 0.10 sec /60°
- Torque: 1.8 Kg-cm
- Voltaje de funcionamiento: 3.0-7.2V
- Temperatura de funcionamiento: -30 ℃ ~ 60 ℃
- Ángulo de rotación: 0º - 180°
- Ancho de pulso: 500-2400 µs
- Longitud de cable de conector: 24.5cm

Es utilizado para implementarlo como actuadores en algunos robots, como sistema de dirección en juguetes RC, cuando se requiere control de posición sin retroalimentación, se usa también en múltiples robots DOF como robots humanoides, etc. [15]

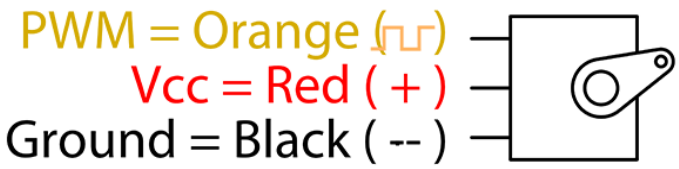


Figura 17.Conexiones del sensor SG90 y función de cada una

Consta de tres cables, tal y como muestra la figura 17, el cable rojo y marrón se usa para alimentar el servomotor a +5 V y el cable naranja sirve para enviar señales PWM, por esta razón hay que conectarlo al pin de la GPIO de la raspberry que genera este tipo de señal, la cual debe tener una frecuencia de 50 Hz, ya que el periodo de PWM debe ser de 20 ms. El tiempo de encendido puede variar entre 1 ms y 2 ms. Entonces cuando el tiempo de encendido es de 1ms, el motor estará en 0 ° y cuando 1.5 ms el motor será 90 °, de manera similar cuando sea 2ms será 180 °. Por lo tanto, al variar el tiempo de encendido de 1 ms a 2 ms, el motor se puede controlar de 0 ° a 180 °. La figura 18 muestra de forma gráfica lo explicado anteriormente.

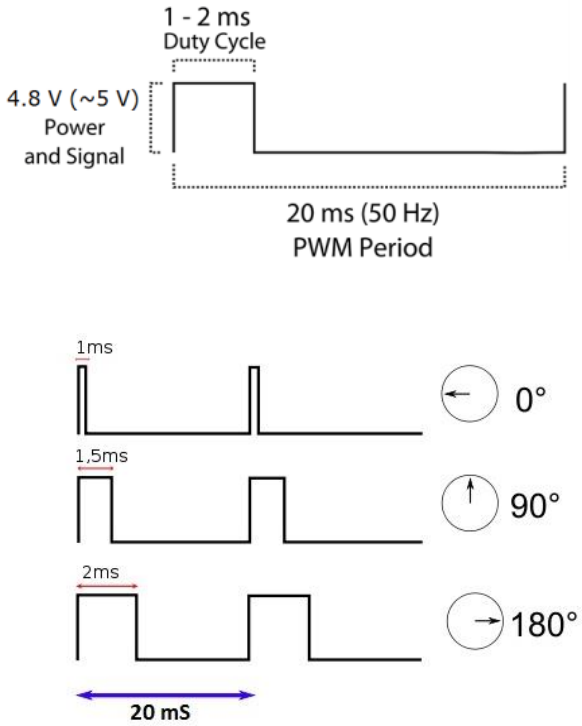


Figura 18. Posición del sensor SG90 en grados según el tiempo de encendido con un ciclo de 20 ms

2.1.4. Módulo MQ-135. Detector de contaminación y polución.

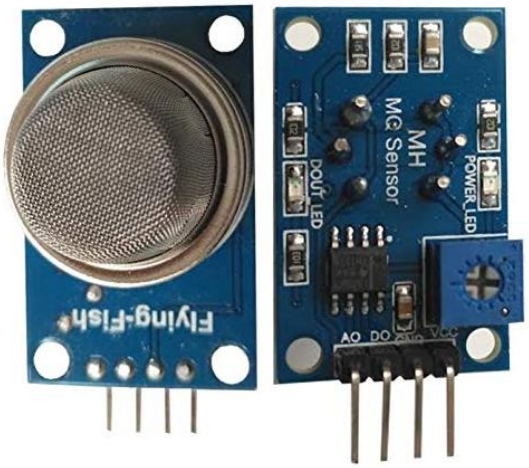


Figura 19. Módulo MQ-135. Detector de contaminación en el ambiente.

El módulo MQ-135 se utiliza para el control de calidad del aire para edificios, son adecuados para la detección de NH3, NOx, alcohol, benceno, humo, CO2, etc.

Los sensores de gas de la serie MQ son sensores analógicos por lo que son fáciles de implementar con cualquier microcontrolador, ya que los encontramos en módulos que nos facilitan las conexiones, basta con alimentarlo.

Se enumeran algunas de sus características más importantes [16]:

- Voltaje de operación: 5V DC
- Corriente de operación: 150mA
- Potencia de consumo: 800mW
- Tiempo de pre-calentamiento: 20 segundos
- Resistencia de carga: Potenciómetro (Ajustable)
- Detección de partes por millón: 10ppm~1000ppm
- Concentración detectable: Amoniaco, sulfuro, benceno, humo
- Concentración de oxígeno: 2%~21%
- Humedad de operación: <95%RH
- Temperatura de operación: -20°C~70°C

Estos sensores son electroquímicos y varían su resistencia cuando se exponen a determinados gases, internamente posee un calentador encargado de aumentar la temperatura interna y con esto pueda reaccionar con los gases provocando un cambio en el valor de la resistencia. Por lo tanto, se comporta como una resistencia y necesita una resistencia de carga (RL) para cerrar el circuito y con este hacer un divisor de tención y poder leerlo desde un microcontrolador. En la figura 20 se expone el circuito interno que grafica el funcionamiento explicado anteriormente.

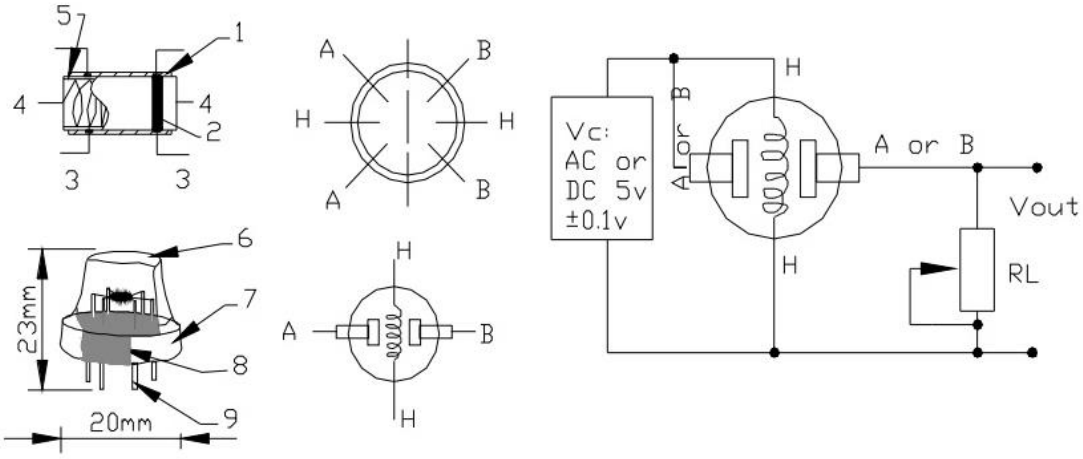


Figura 20. Esquema del circuito integrado en el MQ-135

Tiene seis pines el pin A, H, B se puede observar que se conectan a 5 V.

Notar también que tienen una salida digital la cual internamente trabaja con un comparador y con la ayuda de un potenciómetro podemos calibrar el umbral y así poder interpretar la salida digital como presencia o ausencia del gas. En la figura 21 se observa el LM393 el cual funciona como comparador, el MQ como calentador con resistencias a tierra para evitar que se estropeé y el potenciómetro para detectar ciertos gases calibrándolo.[17]

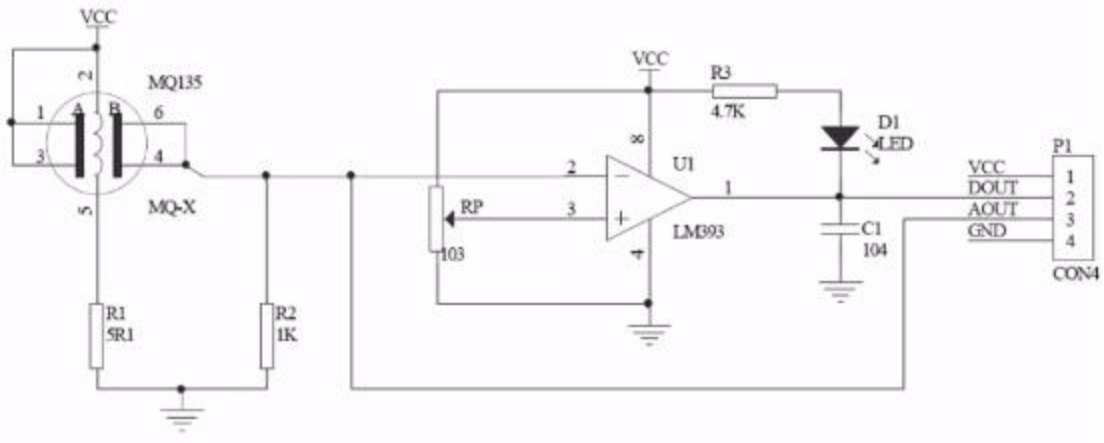


Figura 21. Circuito integrado en el módulo MQ-135

La diferencia entre los distintos tipos de sensores MQ es la sensibilidad a cierta gama de gases, pero siempre detectan a más de un gas, por lo que es necesario revisar los datasheet para escoger el sensor adecuado para nuestra aplicación, por ello se escoge el MQ-135.

2.1.5. Módulo de sonido pasivo de bajo nivel

Figura 22. Buzzer. Módulo de sonido pasivo de bajo nivel

Este módulo permite crear tonos cuando se envía un pulso alto al pin I/O, pin situado en el centro, tal y como muestra la figura 22.

Es un zumbador pasivo por lo que no tiene una fuente oscilante incorporada que emita el sonido si se utilizan señales de CC, en su lugar se debe usar ondas cuadradas cuya frecuencia esté entre 2kHz y 5kHz.

Técnicamente los buzzers son transductores electroacústicos piezoeléctricos, es decir, dispositivos que convierten señales eléctricas en sonido.

Los materiales piezoeléctricos tiene la propiedad especial de variar su volumen al ser atravesados por corrientes eléctricas.

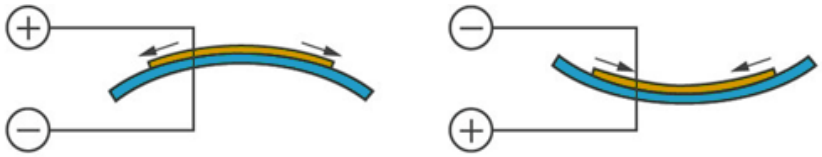


Figura 23. Movimiento de membrana del buzzer cuando aparecen corrientes eléctricas.

Un buzzer aprovecha este fenómeno para hacer vibrar una membrana al atravesar el material piezoeléctrico con una señal eléctrica, tal y como se muestra en la figura 23. Son dispositivos pequeños y compactos, con alta durabilidad, y bajo consumo eléctrico. Por contra, la calidad de sonido es reducida.[18]

Se enumeran algunas de sus características principales:

- Voltaje de funcionamiento: 3.3 V – 5 V
- Intensidad máxima: 30 mA / 5 VDC
- Frecuencia de resonancia: 1.5 Hz – 2.5 KHz
- Dimensiones: 3 cm*1.5cm

2.1.6. Sensor de temperatura DTH11

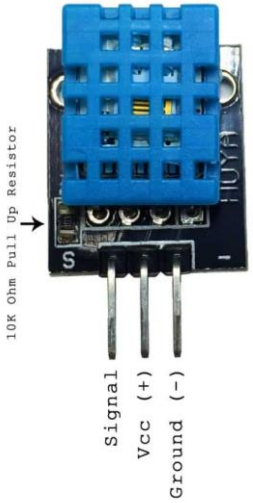


Figura 24. Sensor de temperatura DTH11

El sensor DTH11 tiene como objetivo medir la temperatura ambiente y humedad relativa. Como se observa en la figura 24 hay tres pines, dos para alimentación, positivo y negativo, y una de salida, de la que se obtiene una señal digital bien calibrada, los coeficientes de calibración se almacenan como programas en la memoria OTP, que son utilizados por el proceso de detección de la señal interna del sensor.

El sensor consta de un microcontrolador de 8 bits y 2 sensores resistivos encapsulados en la caja de plástico azul, cuyos materiales tienen las características adecuadas para una respuesta rápida y precisa.

Hay varios modelos en el mercado, en este proyecto se usa el que viene soldado a una placa, ya que nos ahorra montar la resistencia que actúa como pull-up[1]. [19]

[1] Cuando el cable de conexión es más corto que 20 metros, se recomienda una resistencia de tracción de 5K;cuando el cable de conexión tiene más de 20 metros, se elige una resistencia pull-up apropiada.

Como puede ver, el pin de datos está conectado a un pin de E / S de la MCU y se utiliza una resistencia de pull-up de 5 K. Este pin de datos genera el valor de la temperatura y la humedad como datos en serie. Si se conecta la interfaz DHT11 con raspberry, la cual se comunica a través de un único hilo (Single – Bus), hay librerías listas para usar que proporcionan una implementación rápida.

La salida dada por el pin de datos será del orden de 8 bits de datos enteros de humedad + 8 bits de datos decimales de humedad + datos enteros de temperatura de 8 bits + datos de temperatura fraccional de 8 bits + bit de paridad de 8 bits. Para solicitar que el módulo DHT11 envíe estos datos, el pin de E / S debe dejarse momentáneamente bajo y luego mantenerse alto como se muestra en la figura 25 siguiente [20]:

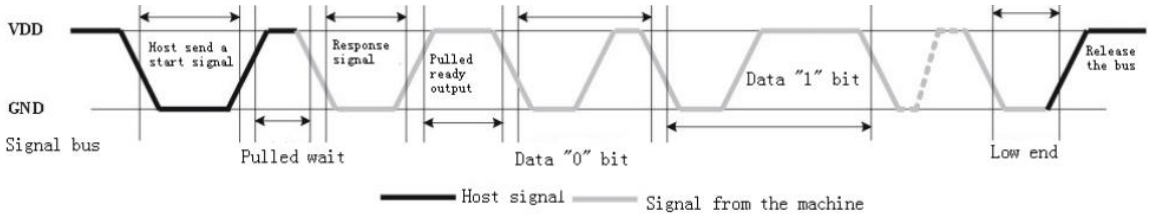


Figura 25. Representación de la salida del pin de datos del DTH11

Se enumera algunas de sus características más importantes [21]:

- Alimentación: 3Vdc ≤ Vcc ≤ 5Vdc
- Rango de medición de temperatura: 0 a 50 °C
- Precisión de medición de temperatura: ±2.0 °C.
- Resolución Temperatura: 0.1°C
- Rango de medición de humedad: 20% a 90% RH.
- Precisión de medición de humedad: 4% RH.
- Resolución Humedad: 1% RH
- Tiempo de sensado: 1 seg.

Por último, este sensor se utiliza sobre todo para aplicaciones como sistemas HVAC (Calefacción, ventilación y aire acondicionado), reguladores de humedad, deshumidificadores o estaciones climáticas.

2.1.7. Mini ventilador de refrigeración

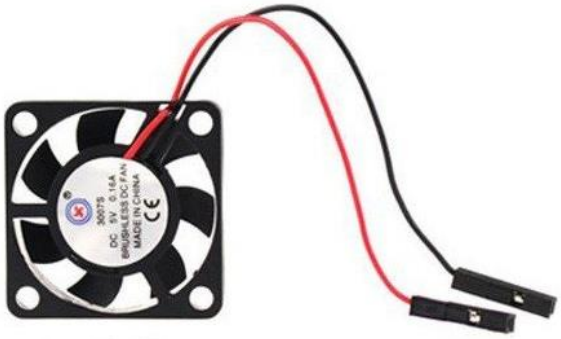


Figura 26. Mini ventilador de refrigeración

Este mini ventilador sirve para reducir la temperatura de microcontroladores cuando se conecta a 5 V, si se considera que hace mucho ruido se puede alimentar a 3.3 V. Puede girar a una velocidad de 6000 rpm y la temperatura disminuye entre unos 5 y 15 Celsius. Es recomendable que trabaje combinado con un disipador.

Aunque se utiliza para disminuir la temperatura del hardware, en este proyecto se usa para simular el aire acondicionado de una vivienda.

2.1.8. Módulo de cámara Kuman con visión nocturna

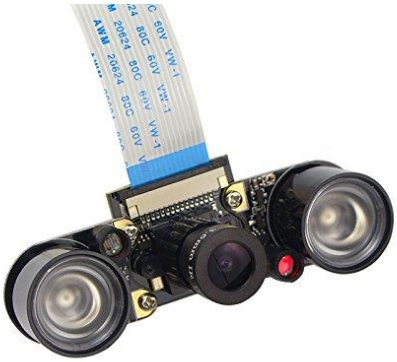


Figura 27. Cámara Kuman con visión nocturna de 5Mpx y 1080p.

Este módulo de cámara se conecta al puerto CSI de la Rapsberry Pi anteriormente citado en apartado 2.1.1. Como se ve en la figura 27 tiene dos led de infrarrojos para visión nocturna. Estos led hay que acoplarlos, tienen dos terminales uno es de 3.3 V y otro es negativo. Están controlados por una célula fotoeléctrica, a medida que esta la luz disminuye la potencia del led aumenta y la cámara tienen mayor alcance de visión cuando hay poca iluminación. Si se va a tener conectados los leds infrarrojos todo el tiempo, será necesario conectar detrás de ellos un disipador de calor ya que tienden a calentarse bastante.

A continuación, se enumeran algunas de las características:

- Foco ajustable de visión nocturna infrarroja 500W de Raspberry PI + luz de fotográfica
- Soporta alta potencia entre 2 y 3W , 850 LED infrarrojos y / o flash de relleno
- Lente: 1/4 5 m
- Apertura (F): 2,9
- Longitud focal: 3.29 mm
- Diagonal: 72.4 grados
- Sensor resolución mejor : 1080p (2592 × 1944 píxeles)
- Dimensión: 25mm x 24mm x 6mm

2.1.9. Diodo LED

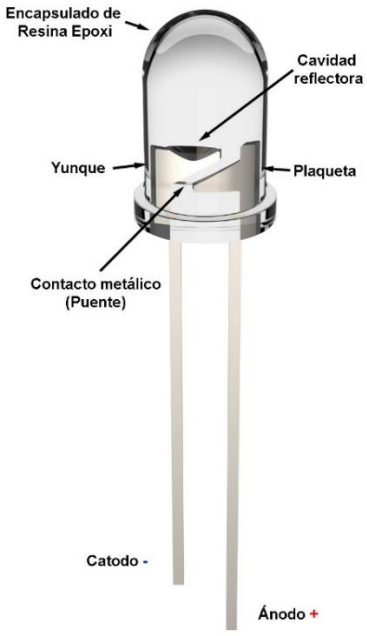


Figura 28. Partes de un diodo LED

En la actualidad existen muchos tipo de leds diferentes, dependiendo de sus características como la potencia, espectro de luz en cual emiten, encapsulado, por lo que hay revisar el datasheet del diodo que se va a usar para no llegar a dañarlo.

En la figura 28 se puede observar las distintas partes que lo componen [22]:

- **Ánodo**: esta patilla siempre es más larga para conectar la tensión positiva.
- **Cátodo**: esta patilla siempre es más corta para conectar la tensión negativa.
- **Cavidad reflectora**: está compuesta por un material semiconductor, que determina el color de la luz.
- **Contacto metálico:** une los dos polos, es decir, une el yunque y la plaqueta.

Compuesto	Color	Longitud de Onda	Caida de Tensión
Arseniuro de Gálio	Infrarojo	940 nm	1.8 a 2.2v
Arseniuro fosfuro de galio	Rojo	630 nm	1.8 a 2.2v
Fosfuro de galio	Verde	555 nm	2 a 3.5v
Nitruro de galio e indio	Azul	450 nm	3.1 a 3.2v
Carbono	Ultravioleta	400 nm	3.3 a 3.8v
	Blanco	-	2.8 a 3.6v

Figura 29. Tabla que relaciona el material semiconductor con el color de la luz

Observando la figura 30 se procede a explicar el funcionamiento de un diodo LED. Lo primero explicar que es un diodo que funciona en polarización directa, es decir, permite el paso de la corriente en un solo sentido, mientras que en sentido opuesto es imposible la circulación.

Cuando a un diodo LED le aplicamos corriente eléctrica procedente de una batería o de cualquier otra fuente de corriente directa, con el fin de polarizarlo directamente, los electrones comienzan a fluir desde la región "N" (cátodo) hacia la región "P" (ánodo). Cada vez que un electrón atraviesa la barrera de potencial que se forma en el punto de unión o juntura entre ambas regiones del diodo para unirse a un hueco emite, simultáneamente, un fotón de luz. De esa forma el electrón libera el exceso de energía que adquirió previamente para poder ingresar en la órbita de un átomo que posea un hueco libre. La luz y color correspondiente a la energía que libera el electrón cuando eso ocurre, puede que sea visible o no al ojo humano, cuestión ésta que depende de la composición química de los materiales semiconductores que se han utilizado para fabricar el diodo. [23]

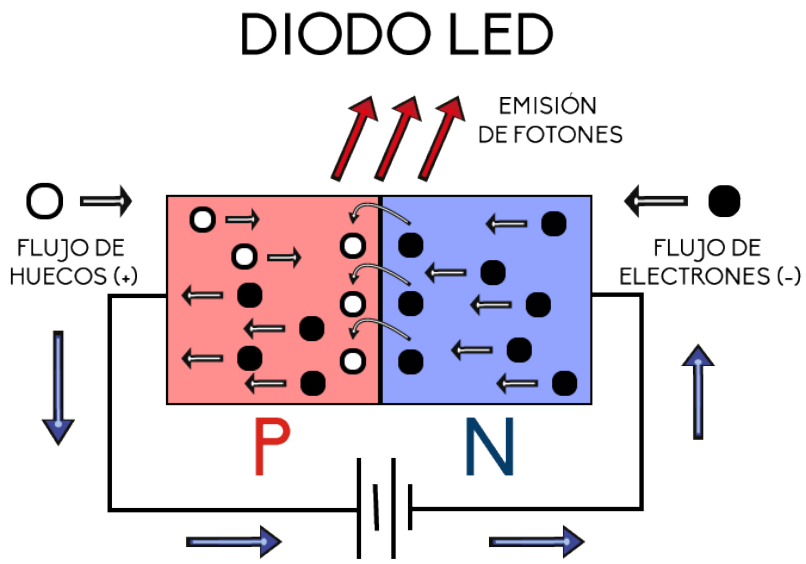


Figura 30. Esquema del funcionamiento de un diodo LED.

2.1.10. Micrófono USB

Figura 31. Mini micrófono USB para PC

Se usa un mini micrófono de USB, tal y como se muestra la figura 31, para la implementación del reconocimiento de voz del sistema, con cuál se podrá encender y apagar las luces de la vivienda ordenándolo a través de la voz.

Las principales características del dispositivo son:

- Distancia efectiva: 2 metros o más
- Sensibilidad: -67 dBV / pBar, -47dBV / Pascal 4dB
- Respuesta de frecuencia: 100-16kHz
- Micrófono omnidireccional
- Tamaño: 22.5 mm x 18.3 mm x 7.1 mm
- Bueno para software de dictado de voz o Skype
- Funciona sin controladores

3. MARCO TEÓRICO. SOFTWARE

3.1. Comunicación web

En este apartado se expondrá los diferentes protocolos y tecnologías utilizadas para desarrollo de la plataforma web y la interacción con los diferentes sensores.

3.1.1. Arquitectura cliente – servidor

La arquitectura cliente – servidor es un diseño de software en que participan dos conceptos diferenciados, el cliente y el servidor, cada uno se encarga de sus respectivas tareas.

El cliente es la parte que solicita ciertos recursos al servidor. Por ejemplo, un usuario que solicita visitar una página web, su propio dispositivo es el cliente y el que le envía la información que necesita es el servidor.

El servidor es el rol que desempeña un equipo ofreciendo un conjunto de servicios a los clientes. Siguiendo el ejemplo anterior, es donde están almacenadas las páginas webs que visita el usuario.

Cuando un usuario se conecta a Internet y solicita una página web a través de su barra de direcciones, lo que hace es establecer contacto con el ordenador que tiene almacenada esa página web. En este ordenador está funcionando un programa que se conoce, genéricamente, como servidor web. Este programa busca la página solicitada en el servidor y se la envía al cliente.

Cuando el usuario abre su navegador y solicita una página web, su ordenador no se conecta directamente al servidor de dicha página. Se conecta a uno de los servidores DNS establecidos. El servidor DNS contiene un directorio con todas las páginas web del mundo y las IPs de los correspondientes servidores. Así pues, el DNS busca en la lista la página que el usuario ha solicitado y el número de IP correspondiente. Esto es lo que se llama **resolver el nombre**. En resumen, la conexión a una página web se establece así, tal y como se representa en la figura 32:

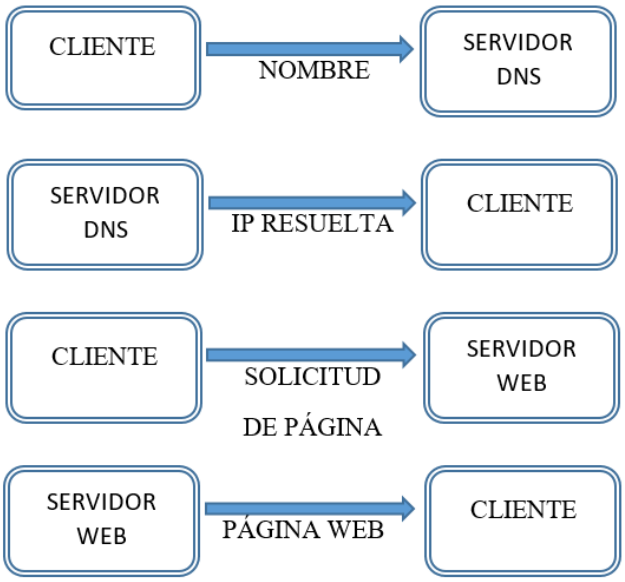


Figura 32. Peticiones y respuestas del cliente - servidor para establecer conexión a una página web

Conectar entre sí dos o más ordenadores para que puedan compartir información y recursos no es tarea simple. Es necesario que la forma en que transmiten y reciben información los distintos equipos esté normalizada. Para este fin existen unos protocolos de comunicación. El conjunto de normas establecidas para la transferencia de información es un protocolo. Todos los ordenadores actuales se comunican entre sí siguiendo lo que se conoce como pila de protocolos TCP/IP. Esta pila abarca diversos protocolos que los equipos informáticos emplean según el tipo de información que transfieran entre ellos.

A fin de que todos los ordenadores del mundo "conozcan" e implementen correctamente dichos protocolos se creó lo que hoy día se conoce como el modelo OSI de la ISO (International Standard Organization). El modelo OSI (Open Standard Interconnectivity) reúne todos los protocolos de los distintos servicios de que se puede disponer en una red de ordenadores.

Debemos saber que la información que se transmite entre un cliente y un servidor no viaja "de golpe", sino que antes de ser enviada es dividida en pequeños fragmentos que se conoce con el nombre de paquetes. Cuando estos paquetes llegan al equipo de destino, son reagrupados para construir la información original.

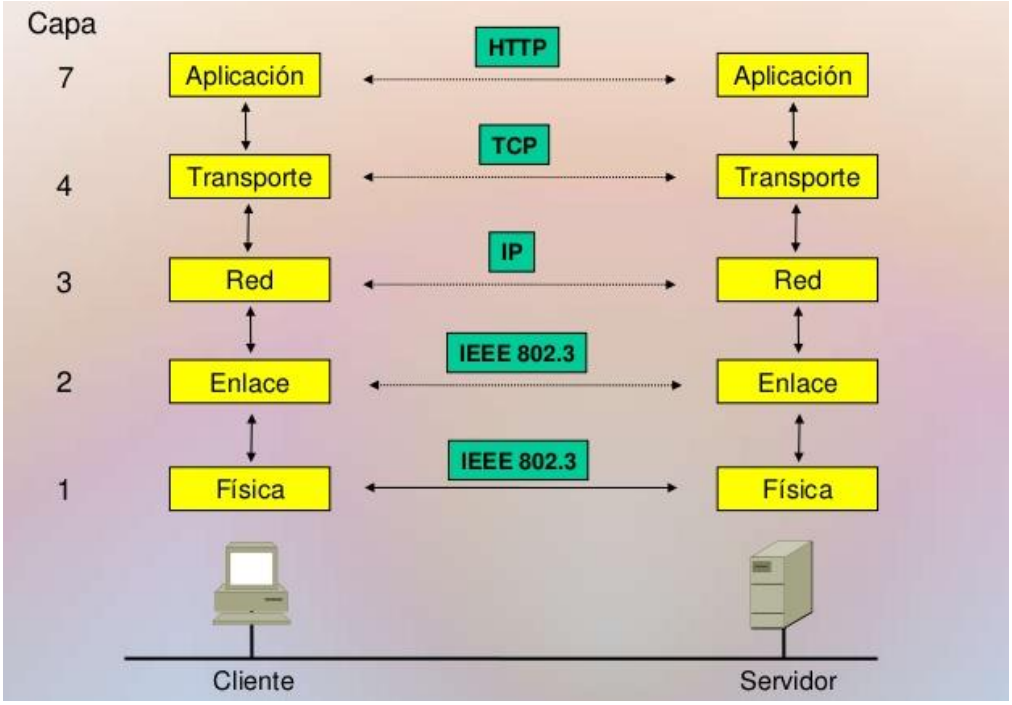


Figura 33. Acceso a un servidor Web desde un cliente en una LAN Ethernet

Hay dos razones por lo que la información se transfiere de la manera descrita anteriormente; por una parte, dada la compleja estructura de nodos de la red, cada paquete puede viajar por un camino diferente, con lo cual puede haber varios paquetes viajando del servidor al cliente. Y por otro lado, si algún paquete se pierde o deteriora durante el trayecto, no es necesario reenviar toda la información, sino que basta con volver a mandar ese paquete.

Los paquetes están formados por dos partes: cuerpo y cabecera. El cuerpo es donde se encuentra el fragmento de información que se transmite y en la cabecera se encuentra otra información de interés como el protocolo empleado, el tamaño del paquete, el checksum, la direcciones IP de origen y destino, puerto de comunicación empleado e información adicional. [24]

3.1.2. HTTP

El protocolo HTTP (Hypertext Transfer Protocol) es el protocolo de comunicación más extendido en Internet. La versión más extendida es la 1.1 (RFC 2616 de 1999).

Cuando se navega por Internet en primer lugar, está la solicitud, cursada por el cliente al servidor, este escucha su petición y, si es posible, le remite los contenidos solicitados, a esta comunicación se le denomina respuesta. Y a continuación, se cierra la comunicación, hasta que el cliente haga otra solicitud. Por esta razón HTTP es un protocolo sin estados. El desarrollo de aplicaciones web necesita frecuentemente mantener estado. Para esto se usan las cookies, que es información que un servidor puede almacenar en el sistema cliente.

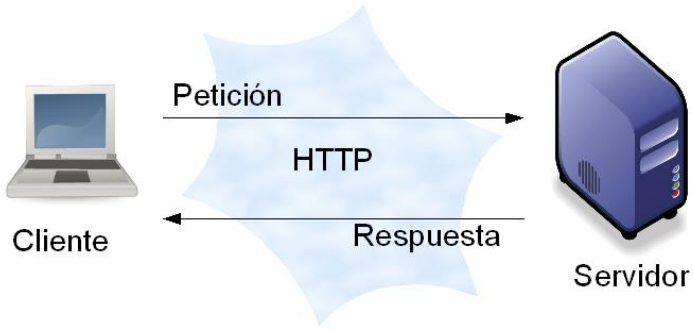


Figura 34. Esquema de comunicación entre un cliente y servidor del protocolo HTTP

Los mensajes HTTP son texto plano y se componen de tres partes, como se muestra en la figura 35:

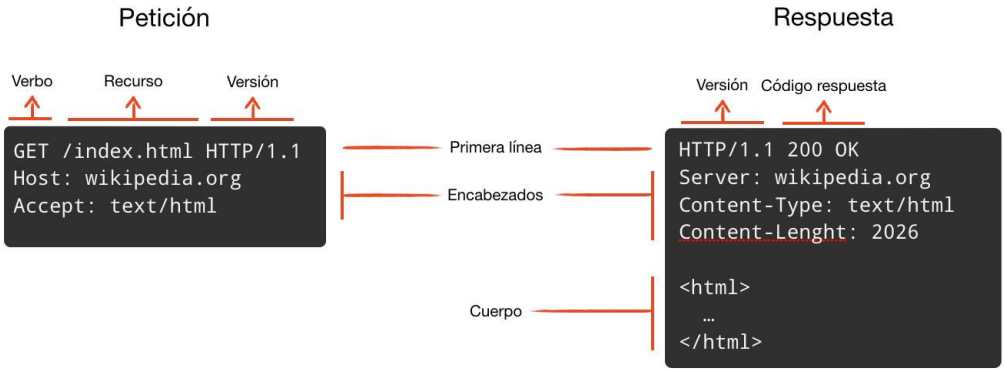


Figura 35. Estructura de una petición y una respuesta de los mensajes HTTP

- **La primera línea** (que es diferente para la petición y la respuesta):

  Para las peticiones: contiene el método de petición, es decir, la acción solicitada al servidor, seguido de URL y la versión HTTP.

  Para respuestas: muestra la versión seguido del código de respuesta.
- **Los encabezados:** están formados por las cabeceras cuyo final se indica con una línea en blanco. Son metadatos y dan flexibilidad al protocolo.
- **El cuerpo (opcional):** su presencia depende del tipo de recurso al que hace referencia la URL. Normalmente son los datos que se intercambian cliente y servidor.

Hay diferentes métodos de petición para hablar con el servidor, se enumeran los más usados:

- **GET**: se utiliza para obtener información de una web
- **POST**: se utiliza para enviar o crear información
- **PUT**: se utiliza para actualizar información
- **DELETE**: se utiliza para eliminar información

Hay métodos de respuesta que son enviados por el servidor al navegador e interpretadas por este. Es un código numérico que indica que ha pasado con la petición.

- **Códigos con formato 1xx**: Respuestas informativas. Indica que la petición ha sido recibida y se está procesando.
- **Códigos con formato 2xx**: Respuestas correctas. Indica que la petición ha sido procesada correctamente.
- **Códigos con formato 3xx**: Respuestas de redirección. Indica que el cliente necesita realizar más acciones para finalizar la petición.
- **Códigos con formato 4xx**: Errores causados por el cliente. Indica que ha habido un error en el procesado de la petición a causa de que el cliente ha hecho algo mal.
- **Códigos con formato 5xx**: Errores causados por el servidor. Indica que ha habido un error en el procesado de la petición a causa de un fallo en el servidor. Quijao, J. L. (2010). Domine PHP y MySQL. Alfaomega.
- [*25*]

3.2. Framework de desarrollo web.

En el desarrollo de Software, un framework es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, en base a la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, librerías y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Sirve para poder desarrollar o escribir código de manera más fácil, permite tener todo mejor organizado y lo más importante permite poder reutilizar el código. Nos permite tener mayor productividad, minimizar los cotos en cuanto al desarrollo, es más sencillo encontrar herramientas adaptadas al framework concreto y además nos ayuda a minimizar errores.

El patrón de diseño más conocido para aplicaciones web, es la arquitectura MVC (Modelo – Vista – Controlador) que ayudan a separar los datos y la lógica de negocio de la interfaz con el usuario.

- **Controlador:** se puede controlar el acceso (incluso todo) a nuestra aplicación, esto pueden ser: archivos, scripts o programas; cualquier tipo de información que permita la interfaz. Así, se puede diversificar el contenido de forma dinámica, y estática (a la vez); pues, sólo se podrá controlar ciertos aspectos.
- **Modelo:** Este miembro del controlador maneja las operaciones lógicas, y de manejo de información para resultar de una forma explicable, y sin titubeos. Cada miembro debe ser meticulosamente llamado, en su correcto nombre y en principio, con su verdadera naturaleza: el manejo de información, su complementación directa.
- **Vista:** la interfaz gráfica que interactúa con el usuario final del programa (GUI). Después de todo, a este miembro le toca evidenciar la información obtenida hasta hacerla llegar con el controlador. [26]

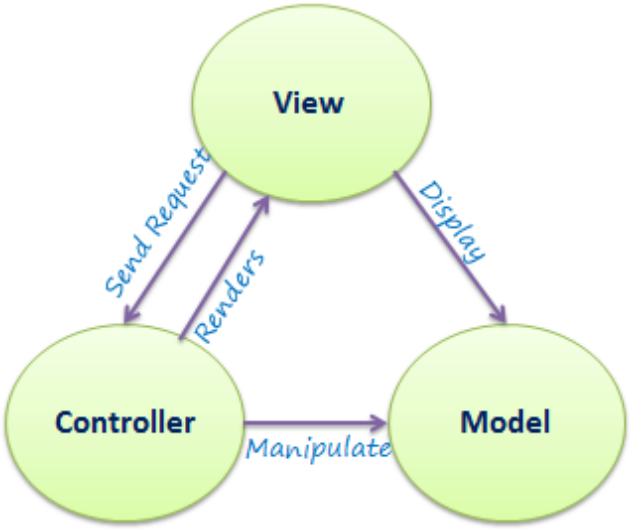


Figura 36. Esquema de patrón MVC

3.2.1. Django

Django es un framework para aplicaciones web gratuito y de código abierto, escrito en Python, aunque también sirve para el desarrollo de procesos de servidor. Su filosofía más importante es la de DRY (Don't Repeat Yourself), es decir, haz el trabajo una vez y reutilízalo tanto veces como puedas. Está soportado por la Django Software Foundation y es utilizado por empresas como Instagram, Pinterest, NASA, Reddit, etc.

A continuación, se enumeran características de este framework que lo hacen muy útil:

- Utiliza el patrón MVC, con ciertos matices.
- Dispone de ORM (Object Relational Mapping).
- Mecanismos del patrón Publisher-Subscriber, es decir, como un manejador de eventos.
- Tiene herramientas de Auto-generación de código.
- Interfaz web de administración automatizada.
- Proporciona un Workflow: proporciona una manera de trabajar
- Muy buena documentación.
- Un gran soporte por parte de la comunidad.

- Sistema de middleware que permite desarrollar una capa entre framework y aplicaciones (ideal para cachés, compresores, etc.).
- Infinidad de plugins/módulos compatibles que nos solucionan muchas tareas.
- Protección CSRF.
- Soporte de sesiones.
- Internacionalización.
- Sistema de diseño de URLs amigables.
- Incluye un servidor web para desarrollar.

PATRÓN MVC / MTV

Hay que aclarar las diferencias sobre el patrón MVC que hay en este framework para entenderlo mejor. En Django, llaman vistas a lo que serían los controladores (de hecho se suele crear en un archivo views.py). La presentación (vista en MVC) la delegan en plantillas. Por tanto, es un MVC, pero llamando vista a los controladores MVC y templates a las vistas MVC.

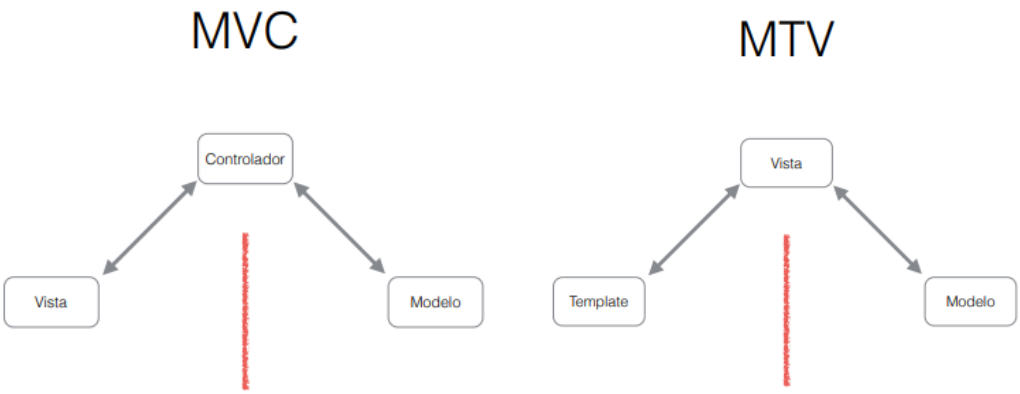


Figura 37. Esquema del patrón MVC vs esquema del patrón MTV (django)

Teniendo como referencia la figura 37, se describe la función de cada una de las partes de dicho patrón:

- **Modelo:**

Representa las entidades de nuestra aplicación, es una capa de abstracción para estructurar y manipular los datos de la aplicación web. Por ejemplo, usuarios, mensajes entre usuarios, posts de un blog, etc.

- **Vista / Template (django)**

La vista se encarga de presentar los datos que el controlador le indica en el formato que le indica. Por ejemplo: una página HTML, datos en JSON, datos en XML.

- **Controlador / Vista (django)**

Se encarga de gestionar los modelos y cómo serán éstos presentados (las vistas). Es el nexo de unión del patrón, ya que un módelo y una vista nunca "se tocan". Los controladores manejan las URLs que solicita el navegador (o cliente) y se encarga de las gestiones necesarias con los modelos para presentarlos como se solicita (con la vista).

PROCESO DE UNA PETICIÓN HTTP

Teniendo la arquitectura en cuenta, veremos a grandes rasgos como se procesa una petición.

Cuando Django recibe una petición HTTP, lo primero que se hace es crear una instancia de la clase HttpRequest que contiene todas las propiedades de la petición y diferentes métodos útiles. Luego se realiza la resolución de la URL. Esto consiste en seleccionar la función de la vista (a partir de la URL especificada en la petición HTTP) que participará en la creación de la respuesta de la aplicación. Una vez que hemos resuelto que función resolverá la URL especificada, se invoca a la función del view con la instancia request de la petición HTTP como primer parámetro, el método de la vista contiene generalmente todo el trabajo lógico, operaciones como obtener objetos de la Base de Datos a través de los modelos, cálculos, transformaciones y finalmente la construcción de la representación de la respuesta final al usuario. [27]

3.3. Python

¿Qué es python?

Es un lenguaje de programación interpretado cuya sintaxis es limpia y esto lleva a un código legible. Fue creado por Guido van Rossum a principios de los años 90 y su nombre proviene del grupo de cómicos ingleses "Monty Python". [28]

A continuación, se enumeran algunas de las características de este lenguaje más importantes:

Lenguaje interpretado: es aquel que se ejecuta utilizando un programa intermedio llamado intérprete, en lugar utilizar un compilador. Los lenguajes compilados a la hora de ejecutarlos son más rápidos pero los lenguajes interpretados son más flexibles y más portables.

Tipado dinámico: no es necesario declarar el tipo de dato que va a contener una determinada variable, sino que su tipo se determinará en tiempo de ejecución según el tipo del valor al que se asigne, y el tipo de esta variable puede cambiar si se le asigna un valor de otro tipo.

Fuertemente tipado: no permite tratar a una variable como si fuera de un tipo distinto al que tiene, se requiere convertir dicha variable concretamente el nuevo tipo. Por ejemplo, si la variable es tipo string no se puede tratar como entero.

Multiplataforma: El intérprete de Python está disponible en multitud de plataformas (UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS, etc.) por lo que si no utilizamos librerías específicas de cada plataforma nuestro programa podrá correr en todos estos sistemas sin grandes cambios.

Orientado a objetos: es un paradigma de programación que pretende modelar todo en función a clases y a objetos, el cual permite un uso de conceptos de cohesión, polimorfismo, herencia, abstracción, etc.

Frameworks de gran utilidad: cuenta con frameworks de gran calibre, los cuales auxilian desde el desarrollo web, hasta el desarrollo de juegos o algoritmos científicos de cálculos avanzados. En este proyecto se hará uso de Django, mencionado en el apartado anterior.

Sintaxis simple: con el objetivo de simplificar el código y hacerlo más legible, este lenguaje hace uso de indentaciones por ejemplo cuando se usan condicionales y también se ha reducido el uso de caracteres como llaves ({) y punto y coma (;). [29]

Estos son los motivos por los cuales en este proyecto se han programado scripts en este lenguaje para la hora de interaccionar con los distintos sensores y la utilización del framework Django para el desarrollo de la web.

3.4. HTML y CSS

HTML es un lenguaje de marcas de hipertexto, tal y como sus siglas indican “Hyper Text Markup Language”, que se utiliza para el desarrollo de páginas webs. Se encarga de distribuir los distintos elementos que contienen una web, dando así una estructura y formato.

Este lenguaje fue desarrollado por la Organización Europea de Investigación Nuclear (CERN) en el año 1945 con la finalidad de desarrollar un sistema de almacenamiento donde las cosas no se perdieran, que pudieran ser conectadas a través de hipervínculos.

Nació públicamente en un documento llamado HTML Tags (Etiquetas HTML), publicado por primera vez en Internet por Tim Berners-Lee en 1991. En esta publicación se describen 22 etiquetas que mostraban un diseño inicial y relativamente simple de HTML. Varios de estos elementos se conservan en la actualidad. Otros se han dejado de usar, y muchos otros se han ido añadiendo con el paso de los años. De esta manera, se puede hablar de que han existido distintas versiones a lo largo de la historia de internet. En este proyecto se va a trabajar con el HTML estándar actual, que es el utilizado por los navegadores y páginas web de hoy en día.

Para la escritura de este lenguaje, se crean etiquetas que aparecen especificadas a través de paréntesis angulares: < y >. Entre sus componentes, los elementos dan forma a la estructura esencial del lenguaje, ya que tienen dos propiedades: el contenido en sí mismo y sus atributos. Permite ciertos códigos que se conocen como scripts, los cuales brindan instrucciones específicas a los navegadores que se encargan de procesar el lenguaje. Entre los scripts que pueden agregarse, los más conocidos son JavaScript y PHP. [30]

```
<!DOCTYPE html>
<html lang="es-ES">
  <head>
    <meta charset="utf-8">
    <title>Ejemplo de 2 párrafos</title>
  </head>
  <body>
    <p>Esto es un párrafo.</p>
    <p>Esto es otro párrafo.</p>
  </body>
</html>
```

Figura 38. Ejemplo de documento HTML

CSS es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado, como HTML. Sus siglas en ingles son Cascading Style Sheets. Describe como se tienen que renderizar los elementos que aparecen en una web.

El gran impulso de los lenguajes de hojas de estilos se produjo con el boom de Internet y el crecimiento exponencial del lenguaje HTML para la creación de documentos electrónicos. La guerra de navegadores y la falta de un estándar para la definición de los estilos dificultaban la creación de documentos con la misma apariencia en diferentes navegadores. [31]

El organismo W3C (World Wide Web Consortium), encargado de crear todos los estándares relacionados con la web, propuso la creación de un lenguaje de hojas de estilos específico para el lenguaje HTML y se presentaron nueve propuestas. Las dos propuestas que se tuvieron en cuenta fueron la CHSS (Cascading HTML Style Sheets) y la SSP (Stream-based Style Sheet Proposal). La propuesta CHSS fue realizada por Håkon Wium Lie y SSP fue propuesto por Bert Bos. Entre finales de 1994 y 1995 Lie y Bos se unieron para definir un nuevo lenguaje que tomaba lo mejor de cada propuesta y lo llamaron CSS. En 1995, el W3C decidió apostar por el desarrollo y estandarización de CSS y lo añadió a su grupo de trabajo de HTML. A finales de 1996, el W3C publicó la primera recomendación oficial, conocida como "CSS nivel 1". El 12 de Mayo de 1998, se publica su segunda recomendación oficial, conocida como "CSS nivel 2". La versión de CSS que utilizan todos los navegadores de hoy en día es CSS 2.1. Al mismo tiempo, la siguiente recomendación de CSS, conocida como "CSS nivel 3", continúa en desarrollo desde 1998 y hasta el momento sólo se han publicado borradores. [32]

La principal característica de CSS es que posibilita establecer una separación entre el contenido y la forma de presentación del documento (dada por las fuentes, los colores y las capas empleadas). Así se puede lograr que muchos documentos HTML compartan la apariencia, utilizando una única hoja de estilo para todos (que se especifica en un archivo .css). Gracias a esta particularidad, se evita tener que repetir el código en la estructura.

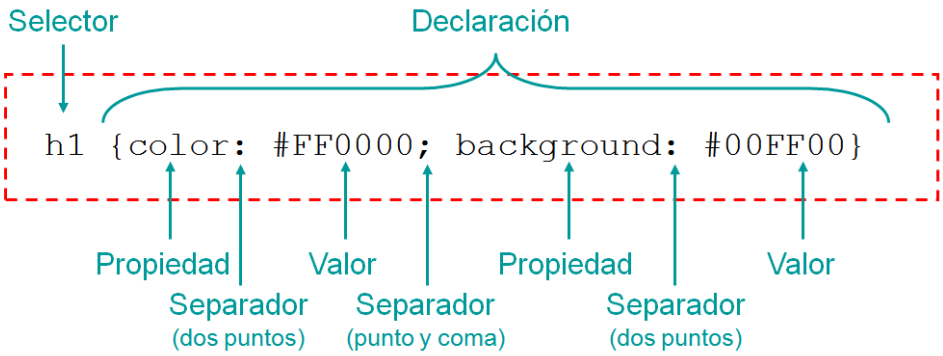


Figura 39. Ejemplo de sintaxis CSS para dar formato a un encabezado

La figura 39, muestra la sintaxis CSS para dar formato a todos los encabezados etiquetados como h1. Todas las reglas CSS están formadas por distintas declaraciones compuestas por una propiedad y su correspondiente valor.

Los frameworks CSS son bibliotecas preparadas para permitir la simplificación, y el mayor cumplimiento de los estándares en el diseño de páginas web usando el lenguaje CSS. Algunos de los frameworks CSS más comunes son Foundation, Blueprint, Bootstrap, Cascade Framework y Materialize. Como en la programación de bibliotecas en los lenguajes de script, los frameworks CSS son usualmente incorporados como hojas de estilo .css externas referenciadas con la etiqueta <*link*>. [33]

Estos dos lenguajes son lo que se van a utilizar para hacer un desarrollo front-end en este proyecto, junto a bootstrap.

3.4.1. Bootstrap

Es un framework para CSS y Javascript, nacido gracias a la compañía Twitter, que permite realizar un diseño responsive de las páginas web, haciendo más sencillo el desarrollo de una interfaz limpia y organizada.

Su funcionamiento se basa en un sistema de rejilla. Es una estructura dividida en 12 columnas y dos tipos de contenedores, las cuales el desarrollador puede gestionar en función de sus necesidades y teniendo en cuenta los cuatro tamaños de dispositivo. A continuación se procede a explicar los elementos más importantes del modelo de rejilla de bootstrap.

Hay dos tipos de contenedores:

- **.container**: usa una anchura fija limitada para mostrar el contenido.
- **.container-fluid**: usa el 100 % de la anchura de la pantalla para mostrar el contenido.

Con el prejio **.row** se puede dividir el contenido en filas y estas a su vez en 12 columnas que se pueden apilar según las necesidades.

En la tabla 1 se muestra los prefijos de clase que se usan para las diferentes resoluciones de pantalla. [34]

	Dispositivos extra pequeños para teléfonos (<768px)	**Tabletas dedispositivos pequeños(≥768px)**	**Dispositivos de sobremesa medianos (≥992px)**	**Dispositivos de escritoriograndes(≥1200px)**
Comportamiento de cuadrícula	Horizontal en todo momento	Colapsado para comenzar, horizontal por encima de los puntos de ruptura.		
Ancho del contenedor	Ninguno (auto)	750 px	970 px	1170 px
Prefijo de clase	.col-xs-	.col-sm-	.col-md-	.col-lg-
# de columnas	12			
Ancho de columna	Auto	~ 62 px	~ 81 px	~ 97 px
Ancho del canal	30 px (15 px en cada lado de una columna)			
Anidables	Sí			
Compensaciones	Sí			
Ordenación de columnas	Sí			

Tabla 1. Aspectos del sistema de rejilla de Bootstrap que funcionan en varios dispositivos

3.5. JavaScript

En este proyecto también se hace uso del lenguaje JavaScript para añadir comportamiento e interactividad al sitio web.

Se le conoce como el lenguaje de la web, es útil para del desarrollo front end y back end, reacciona antes los eventos del usuario, sirve para validar formularios, procesar información, mostrar mapas o ubicaciones, etc.

Al estar formado variables, funciones, estructuras de control y métodos se clasifica cómo un lenguajes de programación interpretado, maduro y estable a diferencia de CSS y HTML que no lo son. Está orientado a objetos basado en prototipos, no existen las clases por lo que el concepto de herencia es distinto. Notar también que es muy complejo y para obtener grandes resultados, hay que implementar muchísimas líneas de código, por lo tanto, para simplificar esta tarea se hace uso de las librerías como jquery.

El lenguaje JavaScript está pensado para ser ejecutado por los navegadores web después de que se hayan cargado los elementos HTML y CSS, este corre sobre el DOM (modelo de objeto de documento), el cuál son los elementos que integran nuestra página web.

Para introducir lenguaje JavaScript en una página HTML se usa el elemento <script>, para indicar el comienzo del código y </script>, que indica el final del mismo. También se puede trabajar con archivos externos, cuya extensión es ".js" y son llamados desde el archivo HTML, como se muestra en la figura 40.

- Documento HTML:

```
<html>
  <head>
    <title> Ejemplo… </title>
    <script  type="text/javascript"  src="/js/codigo.js"  />
  </head>
  <body>
  </body>
</html>
```

- Archivo *codigo.js*:

```
alert("Un mensaje de prueba");
```

Figura 40. Ejemplo de implementación de Javascript en un documento externo

3.6. Telegram Bot API

La API de Bot es una interfaz basada en HTTP creada para desarrolladores interesados en crear bots para Telegram.

Se hace uso de esta herramienta para enviar al usuario alertas de eventos que ocurren en la vivienda a través de mensajes, por ejemplo, cuando hay un humo o escape de gas en la vivienda.

Lo primero es crear el Bot para la aplicación mediante BotFather, cuya función es la de crearlo y recibe un TOKEN de autenticación único. Este identificador único es el que se usa para realizar peticiones a la API.

Todos los métodos diseñados en la API de Bot son insensibles a mayúsculas y minúsculas. Soporta GET y la POST en métodos HTTP. Se usa la cadena de consulta de URL o application, json o application, x-www-form-urlencoded, multipart / form-data para pasar parámetros en solicitudes de API de Bot. En una llamada exitosa, se devolverá un objeto JSON que contiene el resultado.

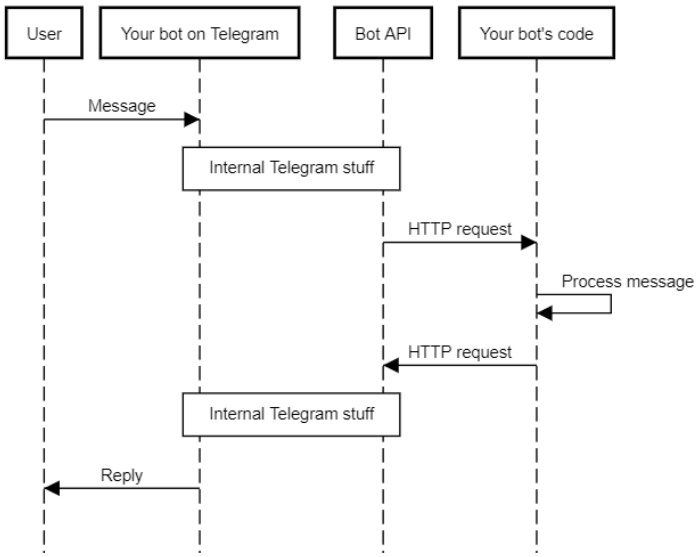


Figura 41. Diagrama de flujo del funcionamiento de la API de Telegram

Cada método cumple una función de enviar un mensaje, foto, audio, animación, documentos, etc. Se expone solo las características de los métodos sendMessage y sendPhoto, ya que son los que se usan en este proyecto, para más información acceder a la referencia. [35]

- **sendMessage:**

Se utiliza este método para enviar mensajes de texto. En caso de éxito, se devuelve el mensaje enviado.

Parámetros	**Tipo**	**Necesario**	**Descripción**
chat_id	Integer or String	Si	Identificador único para el chat de destino o el nombre de usuario del canal de destino (en el formato @channelusername)
Text	String	Si	Texto del mensaje a enviar.
parse_mode	String	Opcional	Envíe *Markdown* o *HTML* , si desea que las aplicaciones de Telegram muestren negrita, cursiva, texto de ancho fijo o URL en línea en el mensaje de su bot.
disable_web_page_preview	Boolean	Opcional	Deshabilita vistas previas de enlaces para enlaces en este mensaje
disable_notification	Boolean	Opcional	Envía el mensaje en silencio . Los usuarios recibirán una notificación sin sonido.
reply_to_message_id	Integer	Opcional	Si el mensaje es una respuesta, ID del mensaje original.

Tabla 2. Distintos parámetros del método SendMessage de la API de Telegram

- **sendPhoto:**

Se utiliza este método para enviar fotos. En caso de éxito, se devuelve el mensaje enviado.

Parámetros	**Tipo**	**Necesario**	**Descripción**
chat_id	Integer or String	Si	Identificador único para el chat de destino o el nombre de usuario del canal de destino (en el formato @channelusername)
Photo	InputFile or String	Si	Foto para enviar. Pase un file_id como String para enviar una foto que existe en los servidores de Telegram, pase una URL HTTP como String para que Telegram obtenga una foto de Internet, o cargue una nueva foto utilizando multipart / form-data.
Caption	String	Opcional	Leyenda de la foto, entre 0-200 caracteres
parse_mode	String	Opcional	Envíe *Markdown* o *HTML* , si desea que las aplicaciones de Telegram muestren negrita, cursiva, texto de ancho fijo o URL en línea en el título de los medios.
disable_notification	Boolean	Opcional	Envía el mensaje en silencio . Los usuarios recibirán una notificación sin sonido.
reply_to_message_id	Integer	Opcional	Si el mensaje es una respuesta, ID del mensaje original.

Tabla 3. Distintos parámetros del método SendMessage de la API de Telegram

3.7. Reconocimiento de voz. Wit.ai.

Wit.ai es una plataforma gratuita para realizar proyectos de reconocimiento de voz, la cual se creó con la idea de diseñar un único sistema de reconocimiento de voz y que sea de uso estandarizado en el mundo de la tecnología. Esta orientado para ser usado por empresas que no tienen tanto recursos para crear su propio sistema como apple o google.

Esta tecnología es de código abierto que reconoce palabras y discursos y los pasa al formato JSON para que los desarrolladores puedan disponer de ellos en sus aplicaciones.

Además del reconocimiento de voz, Wit.ai tiene muchas APIs para crear bots, automatizar respuestas, proyectos de domótica, robots, etc. [36]

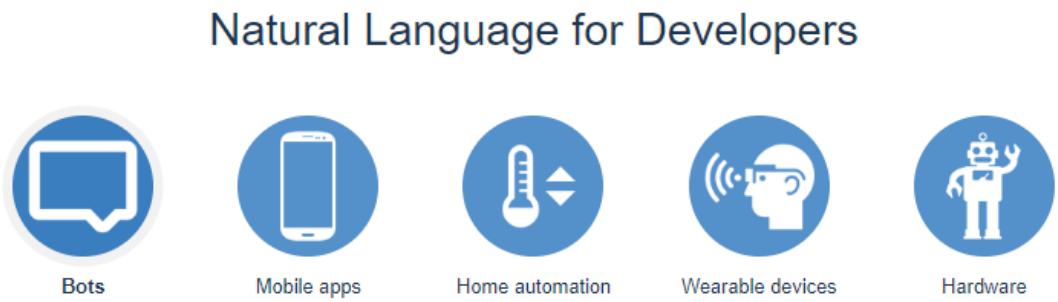


Figura 42. Proyectos desarrollados con la API wit.ai

A la plataforma hay que indicarle lo que el usuario va a decir, ya que no todo el mundo se expresa de la misma forma, para ello hay que distinguir entre tres conceptos: expresiones, entidades y valores.

- **Expresiones**: se puede testear la expresión que dirá el usuario
- **Entidades**: se trabaja por grupos, por ejemplo si se quiere encender luces de distintas habitaciones, se configura una entidad denominada luces, en la cual se relaciona con diferentes valores.
- **Valores**: los valores se agrupan en la entidad configurada para una determinada función.

En la figura 43 se muestra los diferentes conceptos que ofrece la plataforma a la hora de desarrollar una aplicación para un mejor entendimiento de lo definido anteriormente.

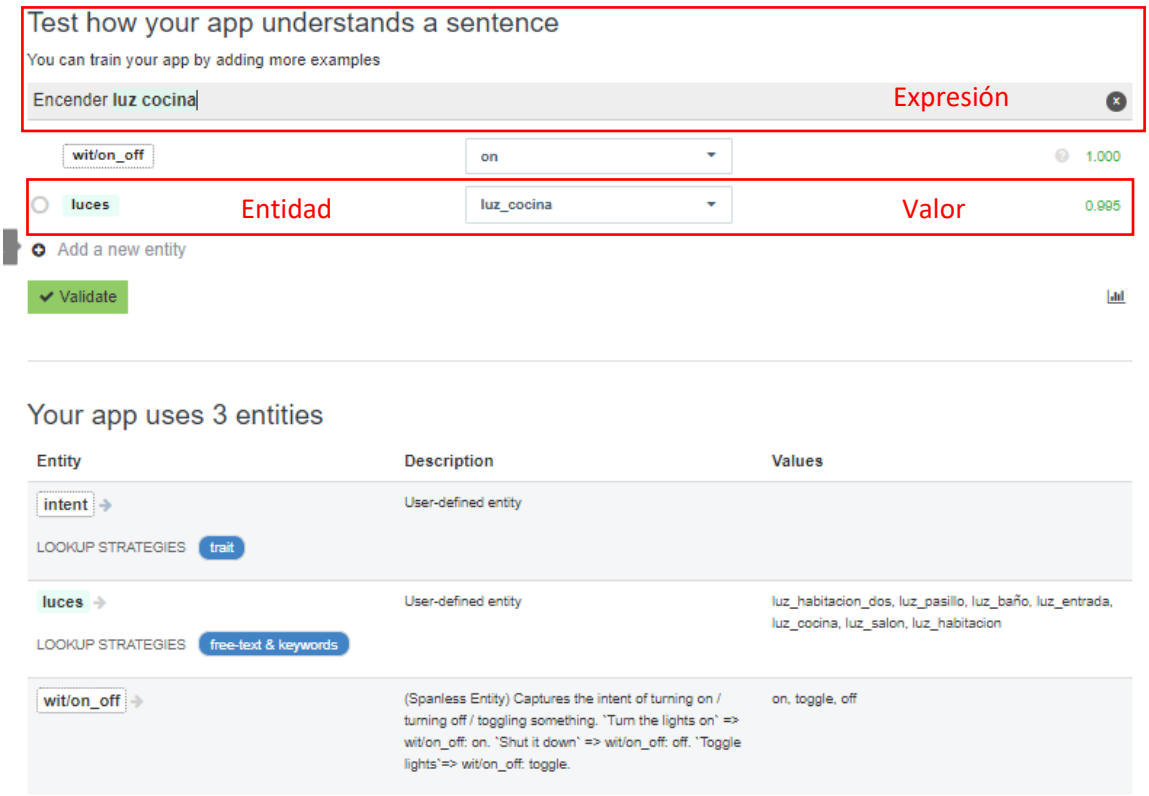


Figura 43. Plataforma Wit.ai. Ejemplo de una entidad configurada.

En este proyecto se va a usar esta plataforma para interactuar con la raspberry pi y poder encender las diferentes luces de la habitación. Los pasos realizados para esta implementación se detalla en el capítulo siguiente.

4. IMPLEMENTACIÓN

Una vez que se ha expuesto el marco teórico de lo que consta este proyecto, se procede a explicar el desarrollo y funcionamiento. Se puede dividir en dos partes:

- **Hardware**: lo forman el conjunto de circuitos que se han tenido que diseñar para el funcionamiento de cada uno de los sistemas implementados.

- **Software**: lo forma tanto los scripts que trabajan en conjunto con los sensores, para dotarlos de "inteligencia", como la programación que hay detrás de la aplicación web, con la cual, el usuario tiene el control de los eventos que sucedan en la vivienda.

4.1. Preparación de la raspberry pi

Lo primero que hay que hacer, antes de trabajar con la raspberry pi y desarrollar cualquier proyecto en ella, es preparar su sistema operativo.

Para ello se necesita un teclado, un ratón, una pantalla, cable HDMI y una tarjeta SD de clase 10, ya que al ser aquí donde se instala el sistema operativo raspbian, se requiere rapidez para que vaya fluido.

Los pasos a seguir para poner a punto la raspberry pi son los siguientes:

1. Formatear la tarjeta SD
2. Descargar la imagen del sistema operativo raspbian accediendo al siguiente enlace : https://www.raspberrypi.org/downloads/raspbian/
3. Montar la imagen de en la tarjeta SD. En este caso, se ha usado el programa Win32DiskImager.
4. Se introduce la tarjeta SD en el lector de la raspberry, y esta se conecta a una pantalla a través de una conexión HDMI junto a un teclado y un ratón.

5. Se le proporciona alimentación y en ese momento se podrá apreciar que instala el sistema operativo mostrando una expansión de archivos y en cuestión de segundos se mostrará en la pantalla el escritorio del SO.
6. En la sección “Raspberry pi configuración” que se encuentra en preferencias cuando se expande el botón de inicio, se configura el idioma, zona horaria, una nueva contraseña (es importante no dejar la contraseña por defecto por temas de seguridad) y en interfaces se habilita ssh (se recomienda cambiar el puerto por defecto de acceso) para luego acceder a ella en remoto.
7. Por último, se conecta a una red wifi para dotarla de conectividad y se actualiza los paquetes y dependencias lanzando este comando mediante la terminal: *sudo apt-get update && sudo apt-get upgrade*. Una vez finalizado este paso, se reinicia lanzando el comando *sudo reboot*.

4.2. Conexiones en la GPIO de la Raspberry pi

Antes de proceder a explicar la implementación de cada uno de los sistemas por separado se muestra un esquema en la figura 44 de lo que hay conectado en cada uno de los pines de la GPIO de la computadora.

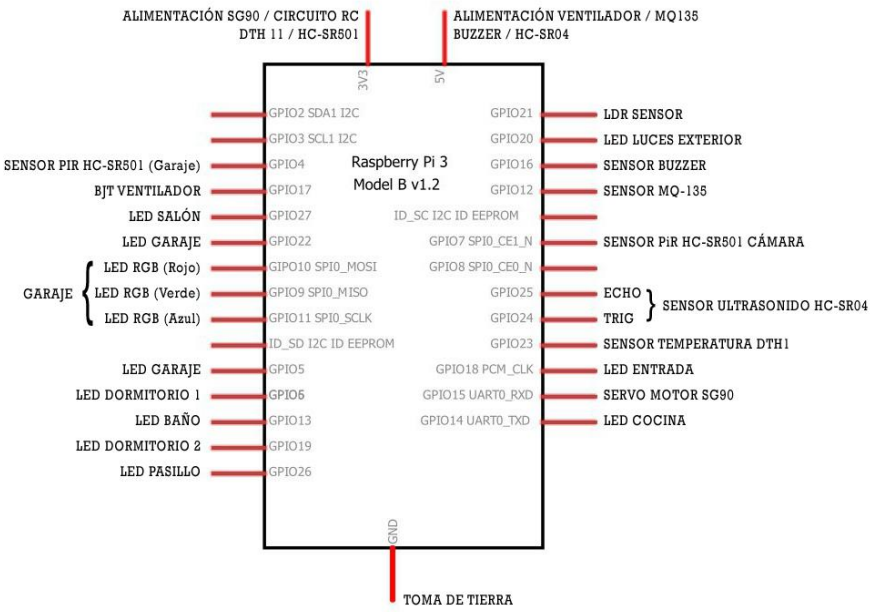


Figura 44. Dispositivos conectados a cada uno de los pines de la GPIO

4.3. Sistemas implementados y seguridad de la vivienda

A continuación, se van a explicar los diferentes sistemas desarrollados para cada una de las funcionalidades concretas.

4.3.1. Sistema de aire acondicionado

Se diseña un sistema que controla la temperatura de la vivienda, la cual se ajusta al parámetro establecido por el usuario mediante la plataforma web. Cuando la temperatura está por encima del valor de referencia, el aire acondicionado se activa automáticamente.

La temperatura que devuelve el sensor será monitorizada en tiempo real, dicha implementación se explicará más adelante en el apartado 4.5.2.

El sensor que se necesita para medir la temperatura es el DTH11, anteriormente explicado en el apartado 2.1.6 y se usa un mini ventilador para simular un equipo de aire acondicionado, expuesto en el apartado 2.1.7.

El circuito de la figura 45, muestra el hardware diseñado para el funcionamiento de este sistema. A través del pin 23 se recoge los datos de temperatura y humedad que obtiene del sensor alimentado a 3,3 V y se conecta la base del BD137 al pin 17, en cuanto se active este, el transistor entra en estado de saturación y activa el ventilador, mientras tanto estará en corte.

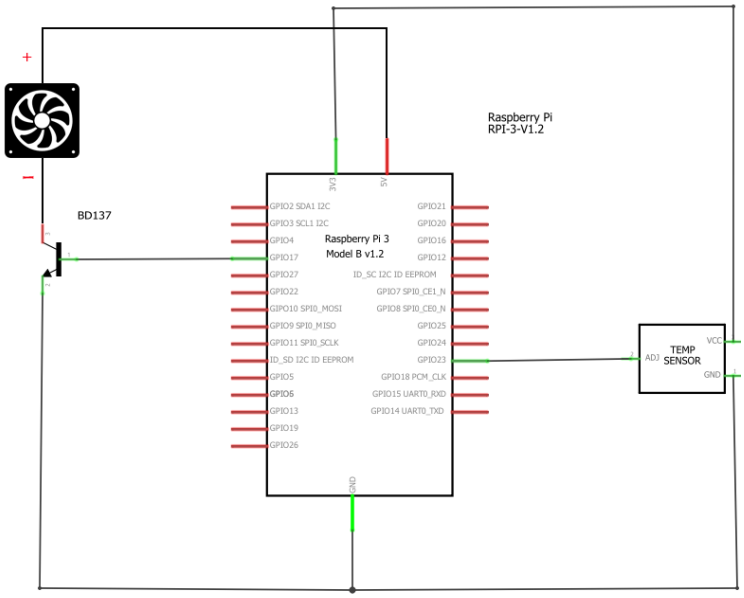


Figura 45. Circuito para sistema de temperatura automatizado

Para obtener las lecturas del sensor, en la Raspberry Pi se instala y desarrolla el software siguiente, siguiendo los siguientes pasos:

1. Instalar git: *sudo apt-get install git-core*
2. Crear una carpeta donde guardar las librerías que se necesita: *mkdir sys_temp*.
3. Se clona el repositorio git de la librería de Adafruit en la carpeta sys_temp: *git clone* *https://github.com/adafruit/Adafruit_Python_DHT.git*
4. Se instala el siguiente paquete para añadir librerías Python: *sudo apt-get install build-essential python-dev*
5. Al clonar el repositorio se crea una carpeta, entrar en ella: *cd Adafruit_Python_DHT*
6. Instalar la librería DTH de Adafruit: *sudo python setup.py install*
7. Dentro de la carpeta /scripts_sensores, se crea un nuevo directorio: *mkdir temperatura*.
8. En el directorio /scripts_sensores/temperatura se clona el repositorio: *git clone* *https://github.com/internetdelascosas/RaspberryPi-DHT11.git*
9. En este directorio /scripts_sensores/temperatura/RaspberryPi-DHT11, aparecen tres archivos: *dht_consola.py, dht_log.py y README.md*

- **README.md** es un archivo de texto que contiene información sobre el proyecto.

- **dht_consola.py** es el programa escrito en Python que al ejecutarse usando el comando python dht_consola.py, se obtiene la temperatura y humedad leído por el sensor DTH11.

- **dht_log.py** es un programa un poco más avanzado, básicamente hace exactamente lo mismo que el programa dht_consola.py pero en vez de mostrar las variables en la consola, las escribe a un archivo de log en /var/log/iot/, al igual que el programa anterior, se puede ver el código fuente en el repositorio GitHub.

10. Hay que modificar el script dht_consola.py para adecuarlo a la funcionalidad deseada. Para que cuando la temperatura suba cierto valor, el ventilador se active.

    A continuación en la figura 46, se muestra un grafo del funcionamiento de dicho código.

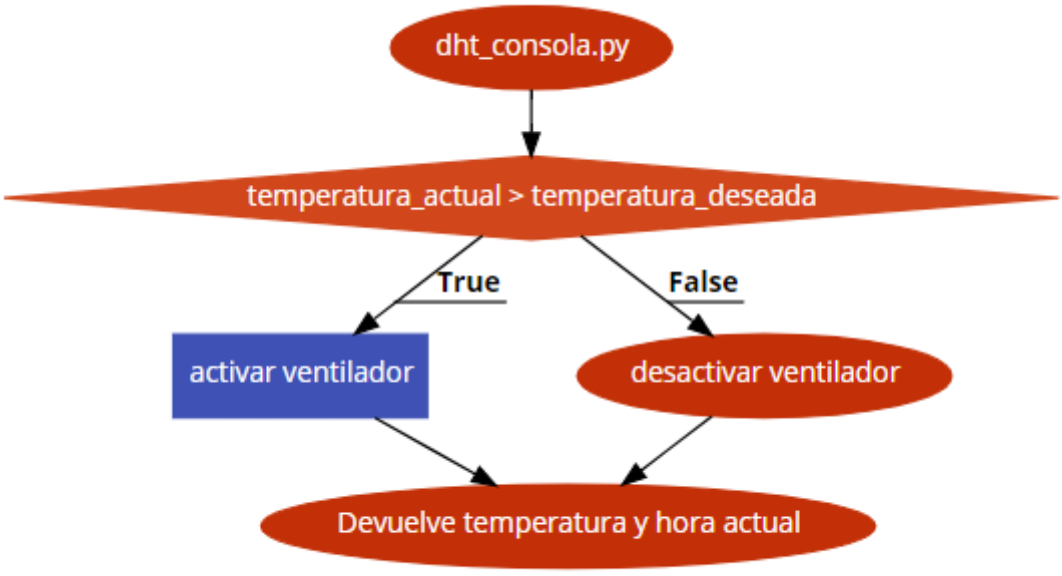


Figura 46. Diagrama de flujo del script dht_console.py para el sistema de aire acondicionado.

4.3.2. Sistema de detección de humos y otros gases tóxicos

Se diseña un sistema para que avise al usuario de la presencia de humo u otros gases en la cocina mediante una alarma sonora y aviso por sms al bot de telegram.

Se necesita un sensor MQ-135 y un módulo buzzer, descrito anteriormente en el apartado 2.1.4 y apartado 2.1.5, respectivamente.

Cuando el sensor MQ-135 detecta humo u otros gases, se activa el pin 16 de la Raspberry Pi, lo que produce un tono que avisa al usuario de que el aire está contaminado. Las conexiónes de los diferentes módulos se pueden observar en la figura 47.

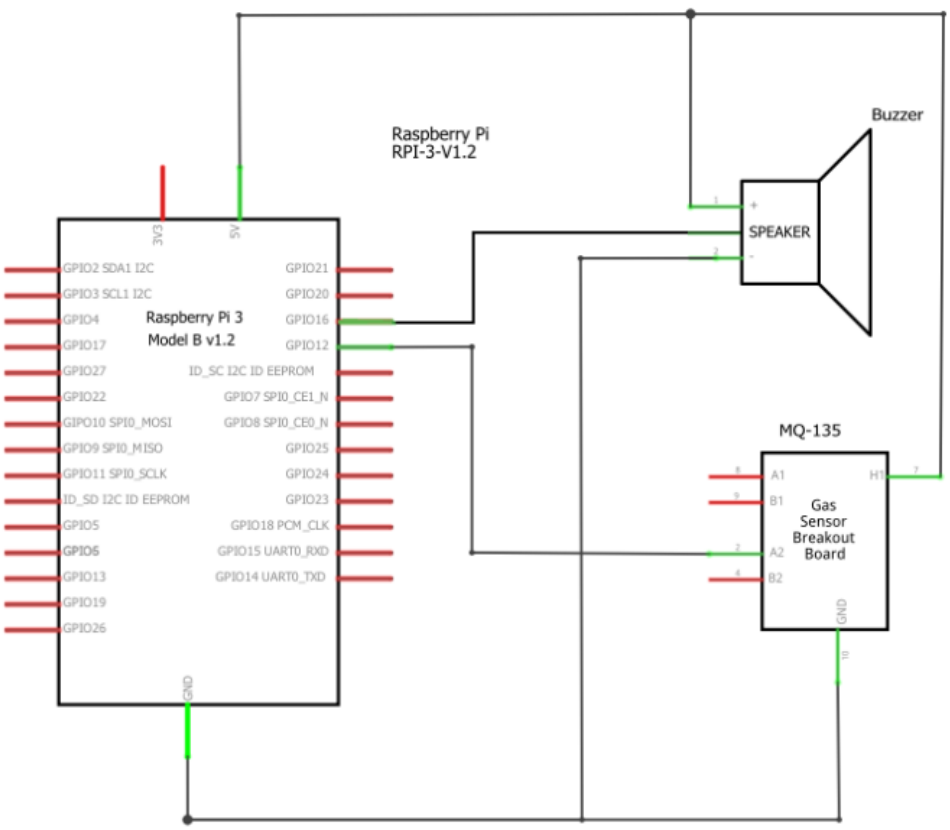


Figura 47. Circuito implementado para el sistema de detección de humos

Para que el sistema sea inteligente, se programa un script en Python llamado sensor_gas_conaviso.py, el cual también ejecuta el script alarma_humo.py para mandar un aviso al telegram. En la figura 48 se muestra un diagrama de flujo para entender mejor el funcionamiento.

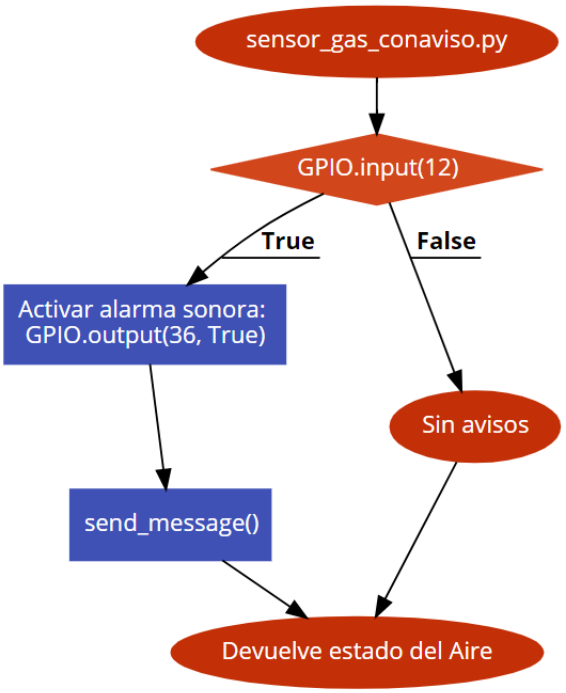


Figura 48. Diagrama de flujo del script sensor_gas_conaviso.py para el sistema de detección de humos

4.3.3. Sistema de sensor de aparcamiento

Se diseña un sistema para que cuando el usuario entre con su vehículo en el garaje, a través de un código de luces, sea avisado de que se acerca a un obstáculo y evitar así un choque.

Para la implementación, es necesario un sensor ultrasonido HC-SR04, mencionado anteriormente en el apartado 2.1.2, un diodo RGB y resistencias.

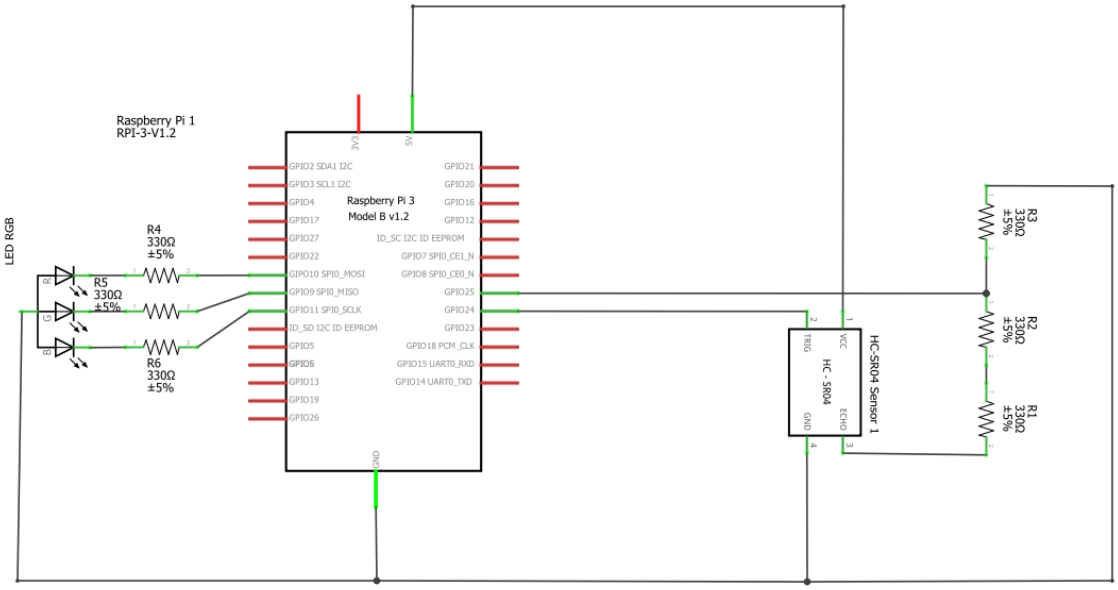


Figura 49. Circuito implementado para sistema de aparcamiento

Se procede a explicar la figura 49 para un mejor entendimiento del montaje diseñado. En el pin 24, que estará configurado como salida, se conecta al pin trig del sensor HC-SR04, el terminal echo se conecta a un divisor de tensión formado por tres resistencias y se alimenta a 5V. Se usa los pines 10, 9 y 11 como salida para conecta el diodo RGB.

Cálculo de resistencias:

- **Divisor de tensión**

  Echo es 5V y se quiere en V_d una tensión 3,3 V ya que es el valor de referencia de los pines de la raspberry pi.

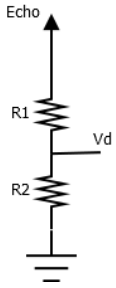


Figura 50. Esquema de un divisor de tensión

Analizando el circuito se obtiene las siguientes ecuaciones:

$$V_d = I * R_2 \quad (4)$$

$$Echo = I * R_1 + I * R_2 \quad (5)$$

Dividiendo la ecuación (4) por (5), se obtiene la siguiente expresión:

$$\frac{V_d}{Echo} = \frac{I * R_2}{I * (R_1 + R_2)} \quad (6)$$

Simplificando y despejando **Vd** de la ecuación (6), se llega a la siguiente ecuación:

$$V_d = \frac{Echo * R_2}{R_1 + R_2} \quad (7)$$

Por último, sustituimos los valores de V_d y Echo en la ecuación (7) y simplificando se obtiene:

$$3{,}3 = \frac{5 * R_2}{R_1 + R_2}$$

$$3{,}3 * R_1 + 3{,}3 * R_2 = 5 * R_2$$

$$R_2 \cong 2 * R_1 \ (8)$$

Se fija el valor de R_1 a 330 **Ω** y R_2, por lo tanto, es 660 **Ω**, tal y como se muestra en la figura 51.

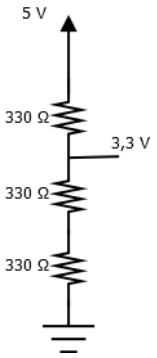


Figura 51. Esquema de las resistencias y tensión deseado en el divisor de tensión

V_d se conecta al pin 25 configurado como entrada. Notar que los GPIO configurados como entrada se comportan como un circuito abierto, por lo que no drenan corriente del circuito.

- **Resistencias Led RGB**

El voltaje min para que encienda el led según el datasheet es de 1.9 V para el rojo y de 2.9 V para el verde y el azul para una corriente de 20 mA.

Sabiendo esto, se calcula las resistencias para cada patilla del led:

$$R_{rojo} = \frac{1{,}9}{20 * 10^{-3}} = 95\ \Omega \cong\ 100\ \Omega\ \ (9)$$

$$R_{azul} = \frac{2{,}9}{20 * 10^{-3}} = 145\ \Omega\ \cong\ 150\ \Omega\ (10)$$

$$R_{verde} = \frac{2{,}9}{20 * 10^{-3}} =\ 145\ \Omega\ \cong 150\ \Omega\ (11)$$

Para terminar, se programa un script llamado sensor_aparcamiento.py para que cuando el coche se vaya acercando a la pared el sensor cambie de verde a azul y de azul a rojo conforme se vaya acercando, siendo el color rojo el peor caso. Dicho código se explica en la figura 52 con un diagrama de flujo.

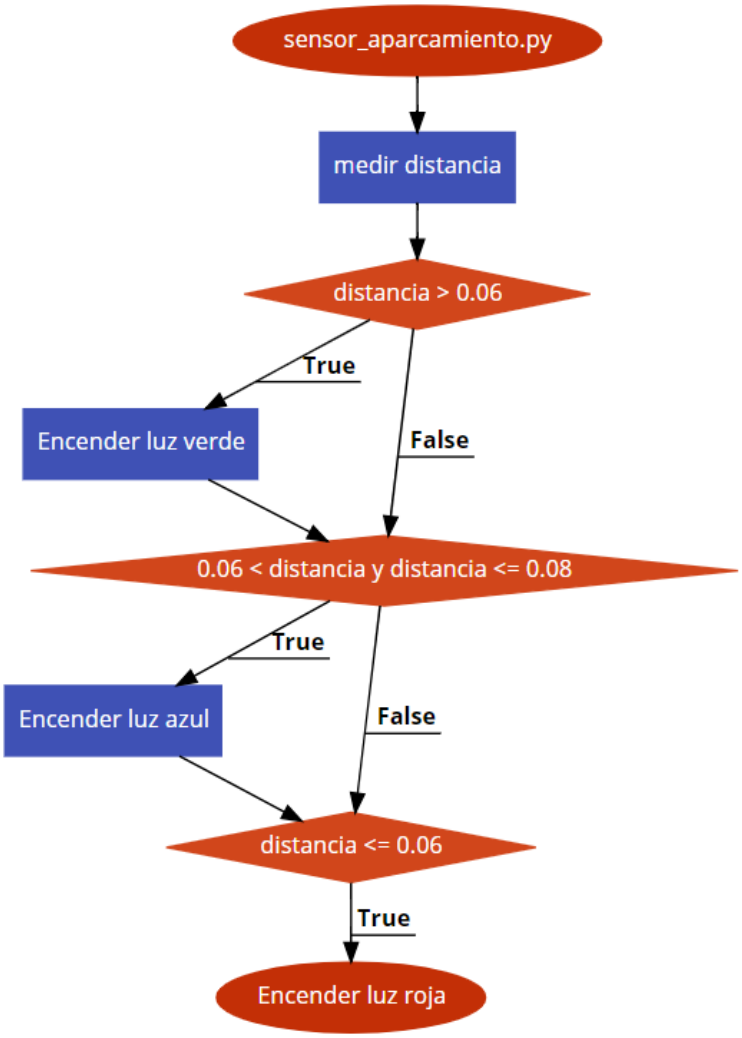


Figura 52. Diagrama de flujo del script sensor_aparcamiento.py para el sistema de detección de obstáculos en el aparcamiento

4.3.4. Sistema de encendido de luz sensor de presencia

Se diseña un sistema para que cuando el usuario entre a una habitación, en este caso se pone en el garaje, se encienda la luz al detectar movimiento. Para ello ha sido necesario usar un sensor HC-SR501 expuesto en el apartado 2.1, un led y una resistencia.

Se realiza el siguiente circuito, mostrado en la figura 53, para implementar el sistema deseado:

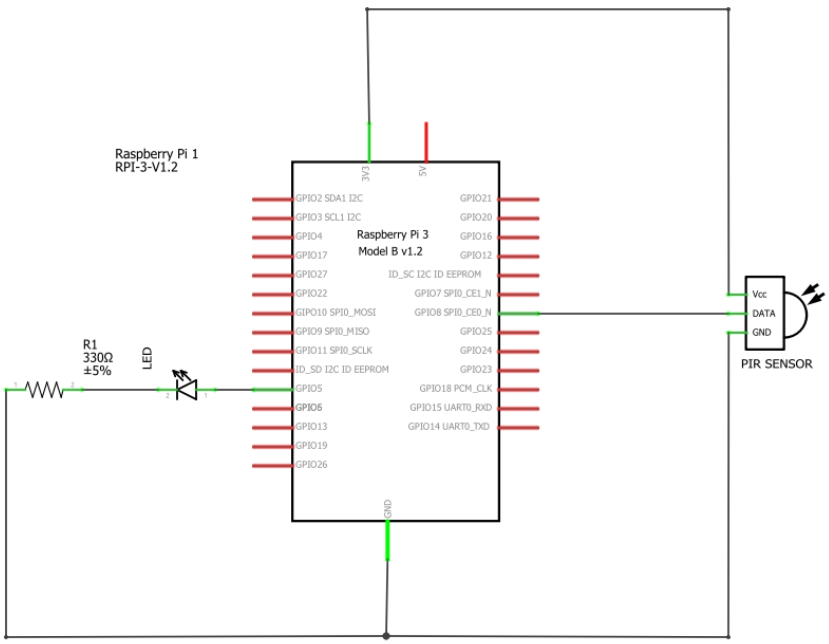


Figura 53. Circuito diseñado para un sistema de encendido de luz al detectar movimiento

Se procede a realizar el cálculo de la resistencia R1, el voltaje es de 3.3 V, que es el proporcionado por el pin y el led necesita unos 10 mA para encenderse, por lo que:

$$R_1 = \frac{V}{I} = \frac{3.3}{0.010} = 330\ \Omega \quad (12)$$

Como se ha mencionado antes en el apartado 2.3.1, el sensor HC-SR501 tiene dos modos de disparo, continuo y de repetición. Se escoge el modo continuo para esta aplicación ya que se desea que mientras haya movimiento del usuario en la habitación la luz permanezca encendida. Para ello se coloca el jumper en la posición H.

Se desarrolla un script en Python llamado luz_sensor_garaje.py. Su funcionamiento se basa en que cuando se recibe información de que el sensor PIR ha cambiado de estado de 0 a 1 (momento en el que ha detectado movimiento) a través del pin 4, se active el pin 22 que es donde está conectado el led, tal y como muestra el grafo en la figura 54.

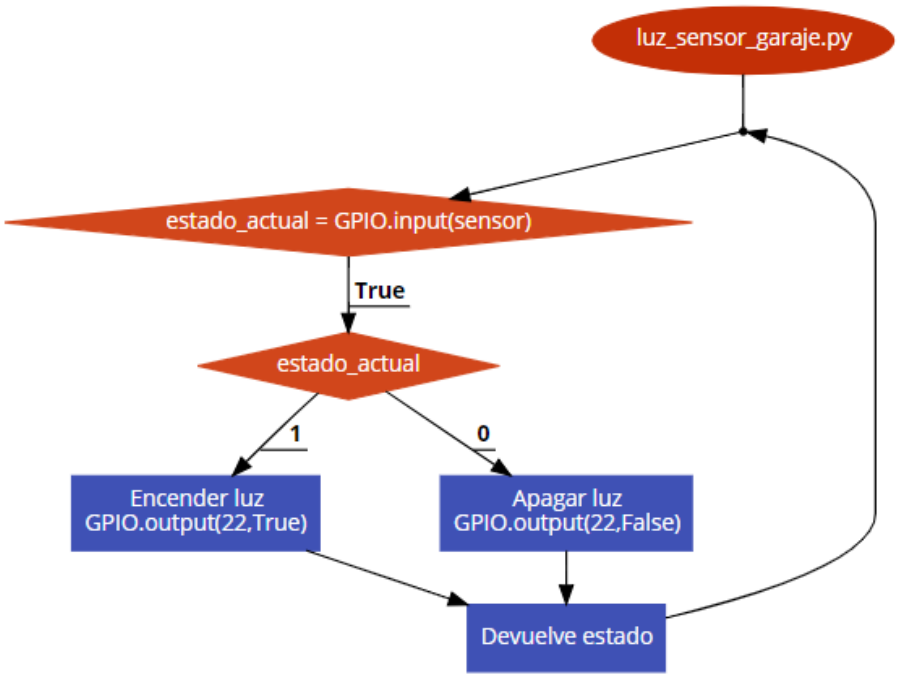


Figura 54. Diagrama de flujo del script luz_sensor_garaje.py para activar la luz en presencia de movimiento

4.3.5. Sistema de encendido de luces exteriores mediante LDR

Se diseña un sistema para que se enciendan las luces exteriores automáticamente cuando empiece a anochecer.

Para que este funcione correctamente, el circuito diseñado consta de un condensador, un sensor LDR, resistencias y diodos LED.

En la figura 55, se representa el circuito elaborado para este sistema, y a partir del cual se explicará el funcionamiento del mismo:

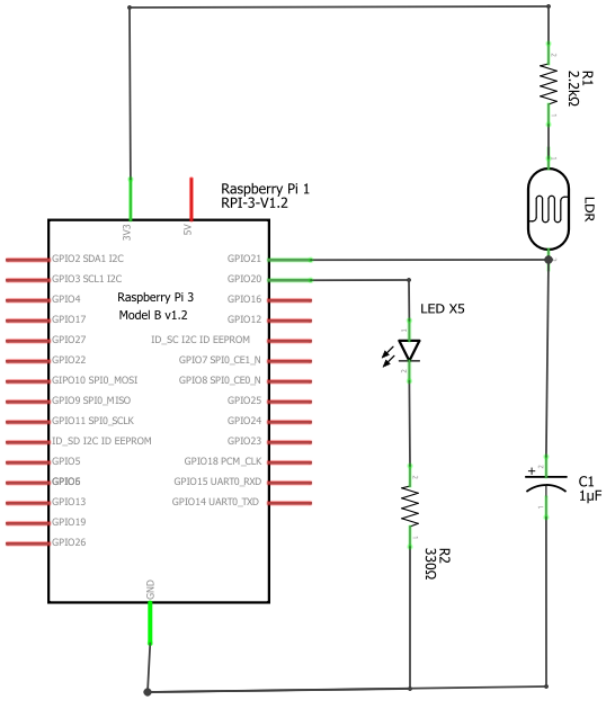


Figura 55.Circuito implementado para un sistema de encendido de luces exteriores cuando empieza a anochecer

Para trabajar con sensores analógicos se utiliza un circuito del tipo RC, como bien se sabe, consta de una resistencia y un condensador en serie. El tiempo de carga de un capacitor depende del circuito resistivo externo, por ejemplo, si se coloca una resistencia muy alta va a tardar más cargar que con una más pequeña, tal y como se demuestra en la ecuación (13):

$t = R * C$ (13), donde t es el tiempo en segundos, R es resistencia en ohmios y C es capacitancia en faradios.

El valor de la resistencia R va a ser variable ya que en condiciones de oscuridad el valor resistivo del LDR aumenta y en condiciones de luminosidad el valor de la resistencia disminuye, de este modo se juega con el tiempo de carga del condensador.

La cantidad de luz que hay en el entorno es equivalente al tiempo de carga del condensador, por lo tanto, cuando la carga es suficientemente alta para que el pin donde se conecta lo detecte como un nivel alto, se encenderá el diodo LED.

Se realiza un script en Python llamado luz_patio.py. El código se basa en que cuando el tiempo de carga del condensador suba, se active el pin 20, como se muestra en la figura 56.

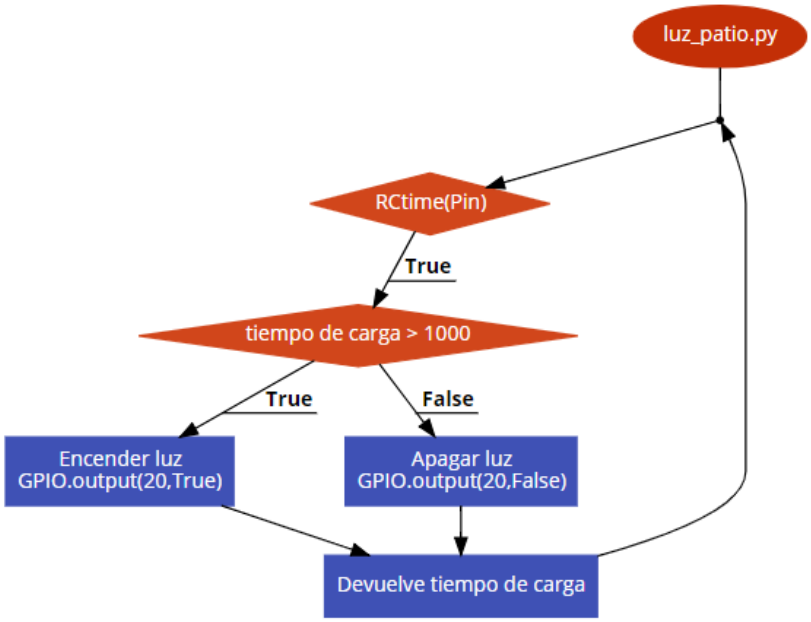


Figura 56. Diagrama de flujo del script luz_patio.py para el sistema de iluminación automático de exteriores

Se implementa una función llamada RCtime, que devuelve el tiempo de carga del condensador dependiendo las condiciones de iluminación del exterior.

4.3.6. Sistema de puerta automatizada por control remoto

Se diseña un sistema para abrir la puerta de la cochera de forma remota. Para ello solo es necesario un servomotor SG90 conectado directamente al GPIO de la raspberry pi 3, tal y como se muestra en la figura 57.

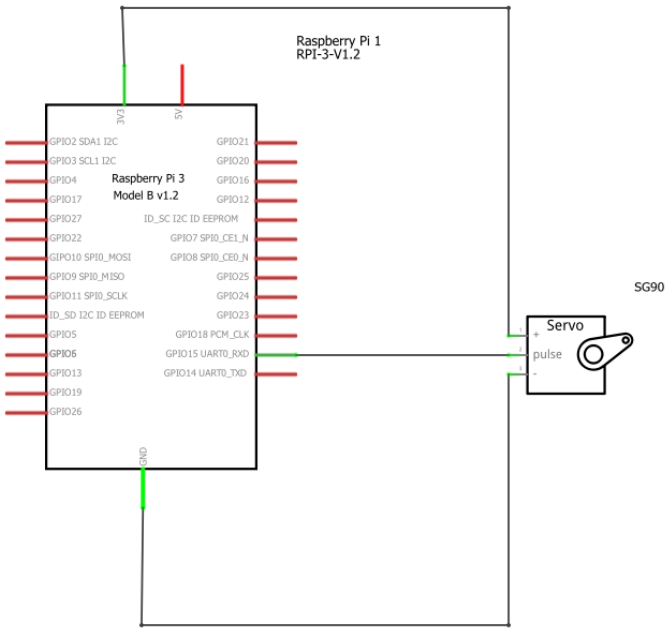


Figura 57. Circuito implementado para un sistema de apertura de puerta automática en el garaje.

Se conecta al pin 15 el cable naranja del servo motor que es el que recibe la señal.

Para que haga el movimiento que se necesita se realiza un script, que pase progresivamente de los 90º a los 180º para bajar la persiana y al contrario para subirla, haciendo uso de la ecuación de la recta:

$$y = mx + n \quad (14).$$

En este caso, se realizan dos scripts llamados subir_persiana.py y bajar_persiana.py.

Mediante la plataforma web el usuario podrá abrir y cerrar la puerta del garaje. Dicha implementación se expondrá con más detalle en el apartado 4.5.1.

A continuación, se muestra en las figuras 58 y 57 el grafo que explica el código implementado.

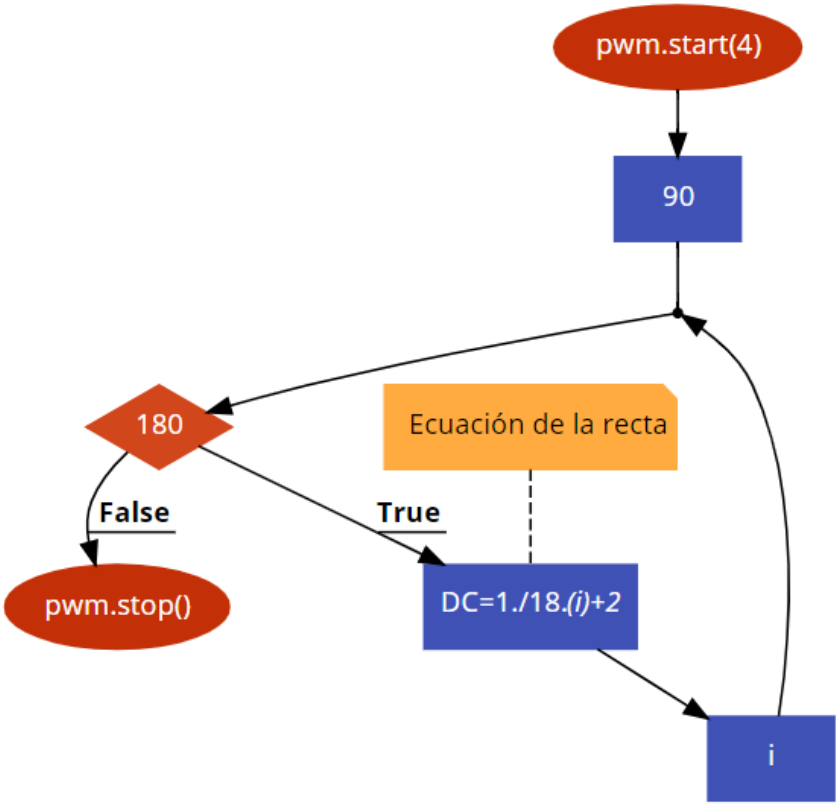


Figura 58. Diagrama de flujo del script subir_persiana.py para subir la puerta de la cochera

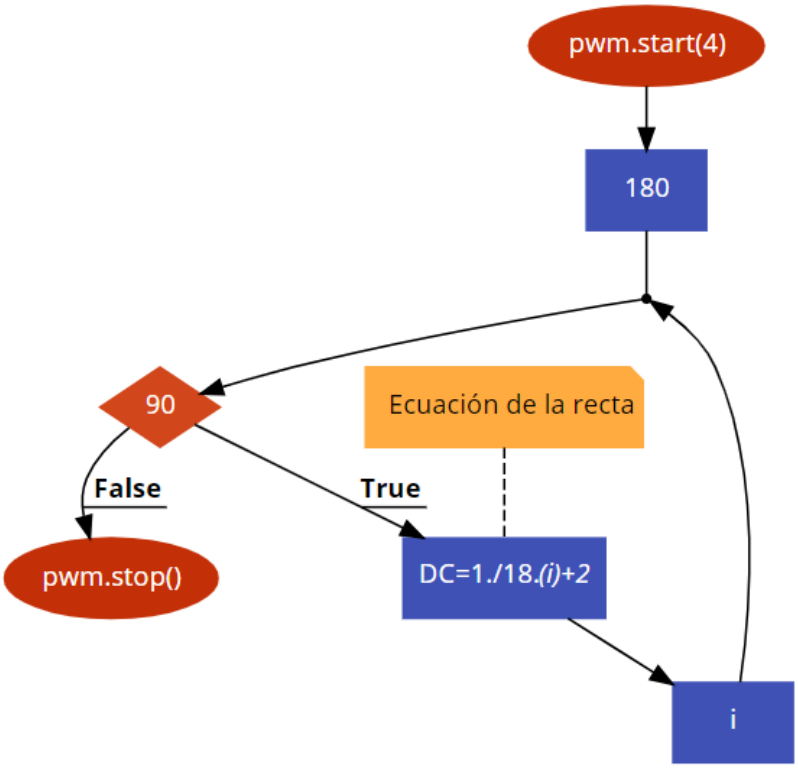


Figura 59. Diagrama de flujo del script bajar_persiana.py para bajar la puerta de la cochera

4.3.7. Sistema de encendido de luz por voz

Se diseña un sistema para poder encender y apagar las luces mediante la voz. Para ello se hace uso de una plataforma de reconocimiento de voz, llamada Wit.ai, la cual es explicada en el apartado 3.7 y de un mini micrófono USB para pc.

Antes de explicar esta implementación, se expone en la figura 60 el circuito realizado para los led de cada habitación.

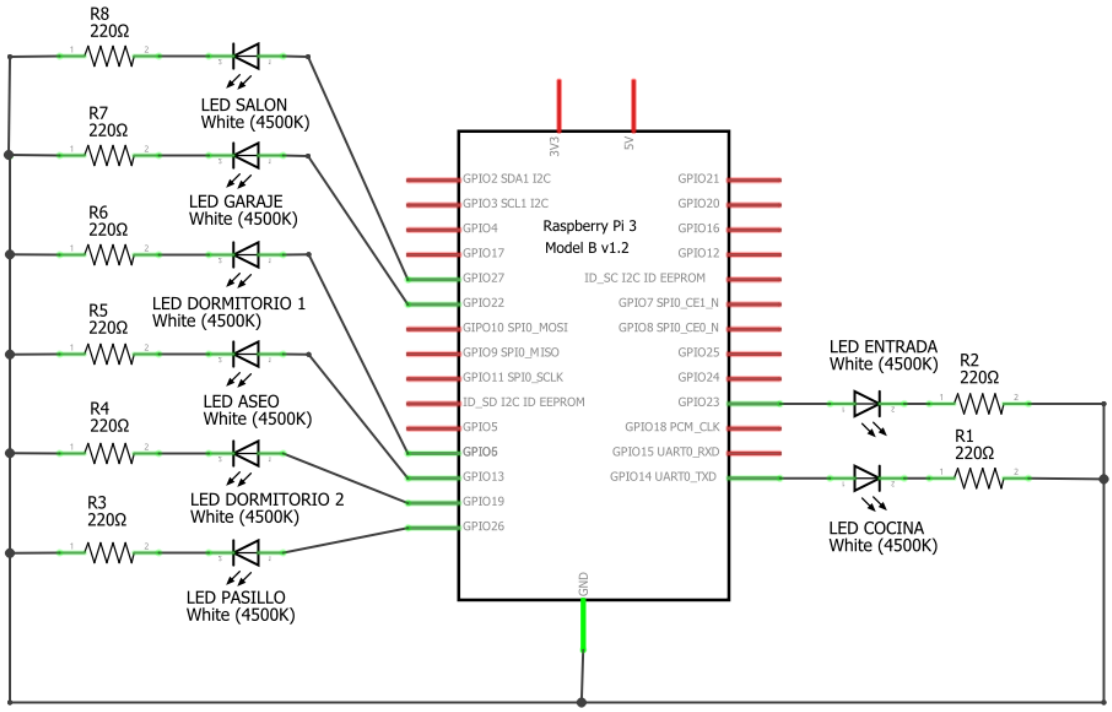


Figura 60. Circuito implementado para encender un led en cada habitación

A continuación, se realizan los cálculos para las resistencias, teniendo en cuenta que la salida de cada pin es de 3.3 V y se quiere una corriente de 15 mA para no dañar a la raspberry pi. Se hace uso de la ley de Ohm para este cálculo:

$$R_1 = R_2 = R_3 = R_4 = R_5 = R_6 = R_7 = R_8 = \frac{3.3\ V}{15 * 10^{-3}\ A} = 220\ \Omega \quad (15)$$

Se crea un script de encendido y apagado de los led para cada habitación.

Se expone los pasos que hay que realizar para encender y apagar cada uno de los leds [37], mediante voz, que se observan en la figura 61:

1. Darse de alta en la plataforma wit.ai: esto se realiza como un registro normal en una página.
2. Se pincha en la pestaña “Console” situada a la arriba a la derecha y después en el símbolo “+”, para añadir una nueva aplicación. Se rellenan los datos que piden.

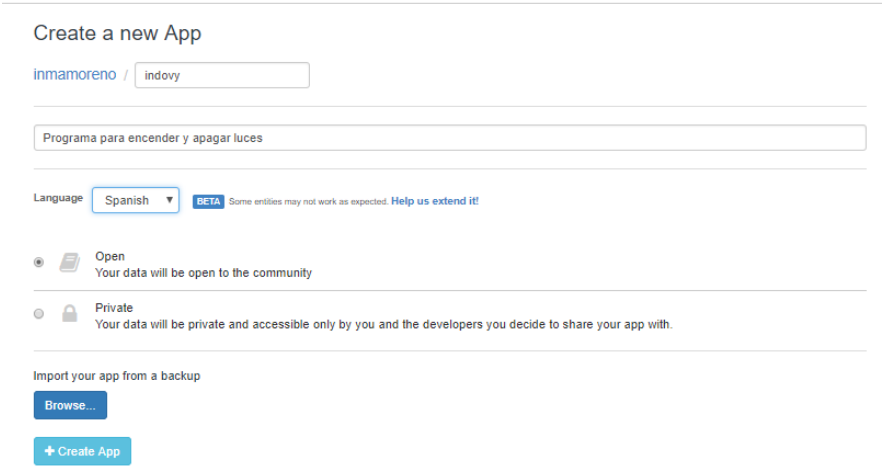


Figura 61. Formulario completado de Wit.ai para crear una nueva aplicación

3. Dentro de la aplicación, se pincha en la pestaña “Settings” para ajustar parámetros de idioma y zona horaria. Lo importante de esta sección es copiar el código de *Server Access Token,* ya que este hace referencia a la aplicación creada y será la vía para comunicarse con la API.

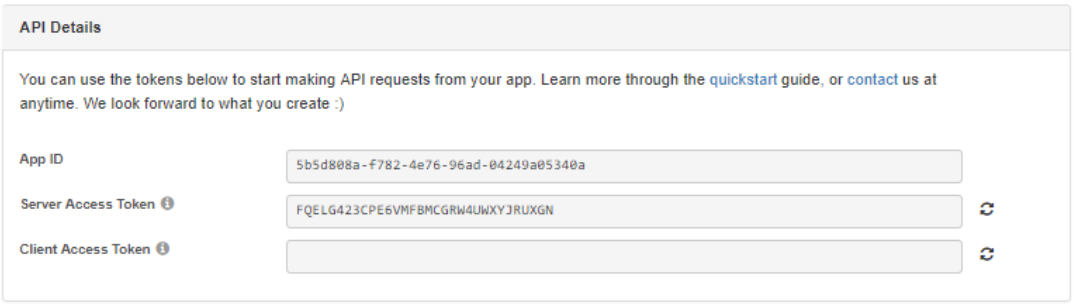


Figura 62. Información a utilizar de la aplicación creada en wit.ai

4. Se instala en la raspberry pi, los paquetes necesarios con los siguientes comandos:

   sudo apt-get update
   sudo apt-get upgrade
   sudo apt-get install arecord
   sudo apt-get install python3-pip
   sudo apt-get install portaudio19-dev
   sudo apt-get install python-setuptools
   sudo pip install pyaudio
   sudo pip install configparser
   sudo apt-get install git
   sudo pip install wit

5. Se copia el proyecto de Github, lanzando el siguiente comando: *git clone https://github.com/cucopc/witrecopi.git*
6. En el directorio /home/pi/witrecopi hay varios archivos para desarrollar el proyecto, en este caso se edita el script llamado wit.py y se busca la línea donde pone wit_access_token = “ ”, donde se escribe el código de Server Access Token anteriormente obtenido en el punto 3.
7. Al archivo llamado grabar.sh, se le da permiso de ejecución con este comando: *sudo chmod +x grabar.sh*
8. Cambiar las acciones del script llamado voz.py para que responda la aplicación a las órdenes de usuario y se ejecute el script que corresponda en cada caso.
9. Dentro de la plataforma de wit.ai, hay que dar de alta las entidades necesarias. Tal y como se muestra en la figura 63.

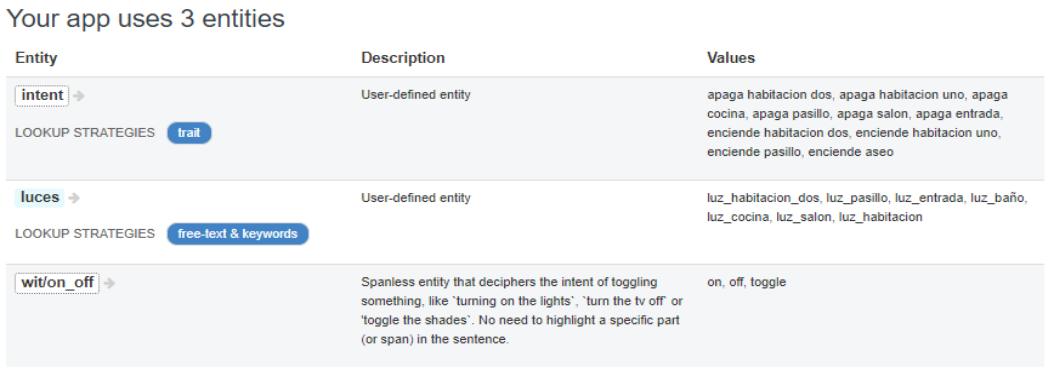


Figura 63. Entidades y expresiones que se crean en wit.ai para encender las luces mediante la voz

Para finalizar, se ejecuta el script voz.py, cuyo pseudocódigo se puede consultar el grafo de la figura 64, la cual escuchara lo que dice el usuario devolviéndolo en forma de texto.

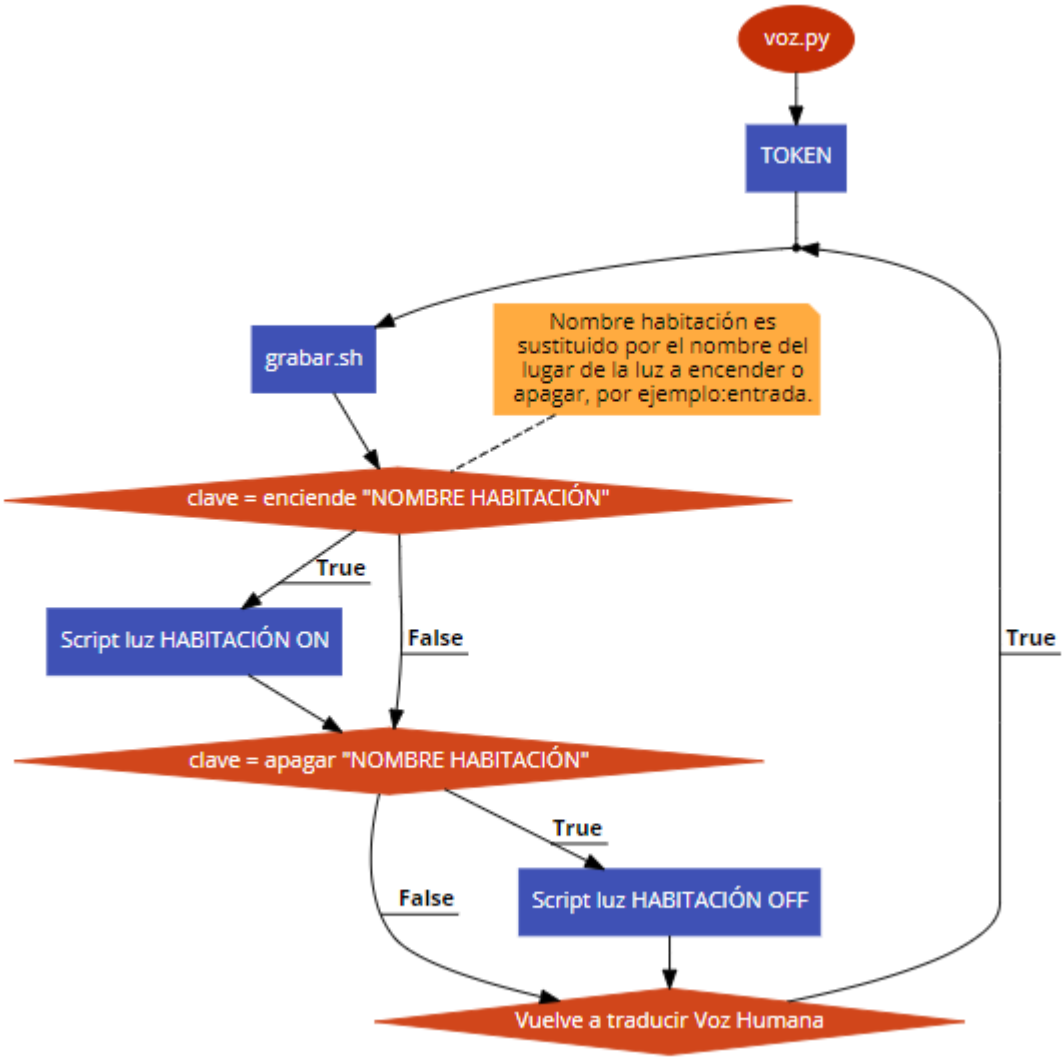


Figura 64. Grafo de la implementación del script voz.py para encender y apagar luces mediante voz

4.3.8. Sistema de alertas por mensajes a través del bot de telegram

Se crea un sistema de alertas para avisar al usuario de eventos que ocurren en su vivienda a través de mensajes al bot de Telegram creado específicamente para ello. Los bots son simplemente cuentas de Telegram operadas por software, no personas, y a menudo tienen características de inteligencia artificial. Pueden hacer cualquier cosa: enseñar, jugar, buscar, transmitir, recordar, conectarse, integrarse con otros servicios o incluso pasar comandos a Internet of Things [38]. El código de Telegram está abierto para todos los desarrolladores, lo mismo pasa con su API, la cual es usada en este proyecto y explicada anteriormente en el apartado 3.6.

Los eventos de los que usuario será avisado por Telegram son:

- Alerta cuando entre un intruso en la vivienda
- Alerta cuando haya un escape de gas

A continuación, se expone los pasos a seguir para crear el bot para la aplicación deseada:

1. En el buscador de Telegram se busca @BotFather, este es el padre de todos los bots. Iniciando una conversación con él, es posible crear el bot. Si mandas */start*, muestra los comandos posibles para configurar el bot, tal y se observa en el figura 65.

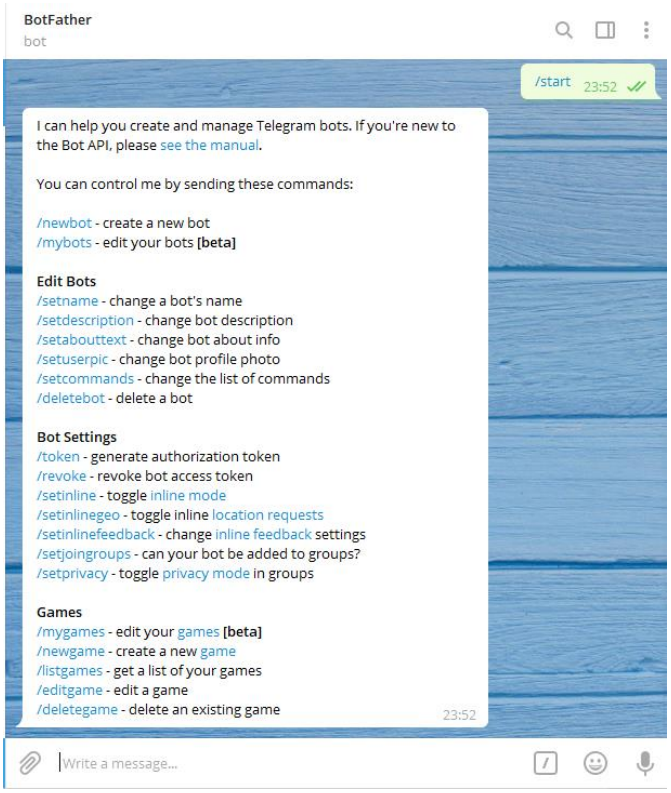


Figura 65. Opciones que ofrece BotFather para configurar un bot

2. Con el comando /newbot, se crea el bot llamado “Indovybot”, tal y como muestra en la figura 66, cuando se termina de configurar el botFather te devuelve un código único llamado TOKEN, mediante el cual se podrá establecer comunicación con la API.

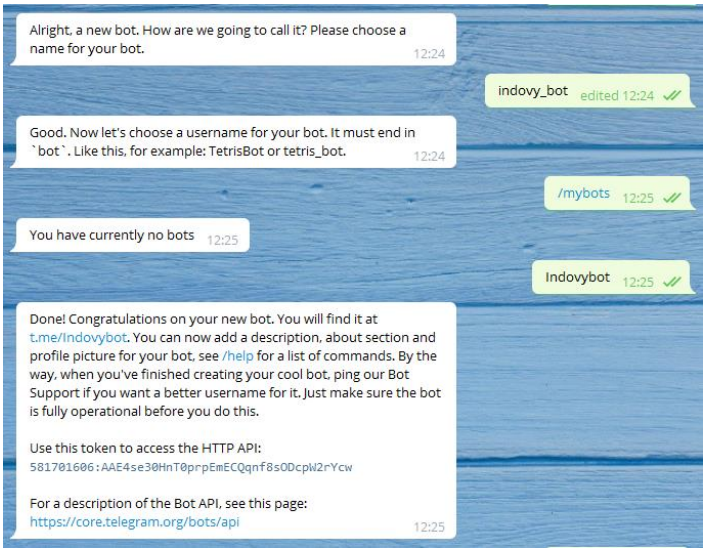


Figura 66. Ejemplo de cómo se crea un bot

3. Una vez que ya se ha creado el bot para las alertas, es necesario además del TOKEN conseguir el valor del chat_id. Para ello se accede al bot creado en el punto anterior y se inicia una conversión. Acto seguido, se accede a esta url a través de un navegador: https://api.telegram.org/bot<TOKEN>/getupdates

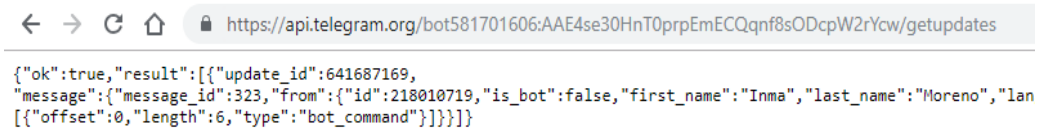


Figura 67. Cadena de valores que devuelve la Api Telegram al hacer una petición con el TOKEN de un bot creado

Como se puede apreciar en la figura 67, se devuelve una cadena de valores en donde se encuentre el campo "id", este número es el chat_id.

4. Por último, para probar que el bot funciona se ejecuta esta url en el navegador: https://api.legram.org/bot$token/sendMessage?chat_id=$chat_id&text=Hello+World en el navegador. (Reemplazando $token y $chat_id con las ID correspondientes). Tal y como se muestra en la figura 68.

Figura 68. Comprobar que el bot creado funciona

Una vez creado el bot, se instala la biblioteca en la raspberry pi lanzando este comando: *pip install pyTelegramBotAPI.*

Se crea dos scripts llamados alarma_humo.py y alarma_presencia.py. El primero manda un texto avisando de que hay humo en la vivienda con el método send_message() y el segundo manda un texto y una foto avisando de que se ha registrado actividad sospechosa en la vivienda haciendo uso del método anterior y send_photo().

4.3.9. Cámara de seguridad

En este apartado se expone la instalación del módulo de cámara de seguridad y el servicio motion, el cual nos permitirá emitir en streaming lo que sucede en tiempo real en la vivienda.

Primero se conecta el cable plano de la cámara a la raspberry pi en el conector CSI, levantando las dos pestañas laterales para poder introducirlo bien y con los cables metálicos mirando al puerto HDMI, tal y como se muestra en la figura 69, una vez bien alineados se aprieta las pestañas para bloquearlo e impedir que se mueva.

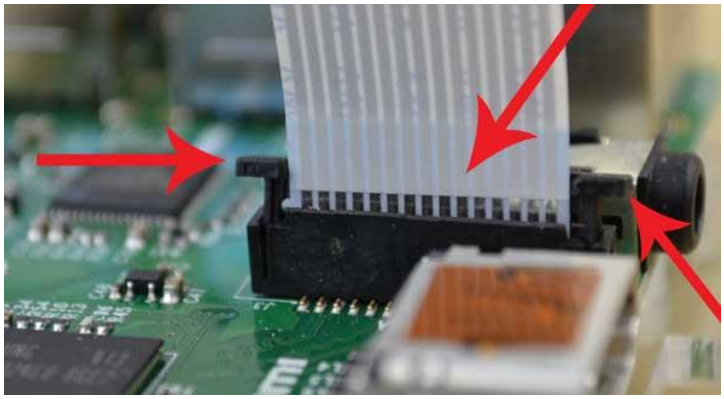


Figura 69. Ilustración del módulo de la cámara bien conectado al conector CSI

Para asegurar de que la cámara esté encendida, se accede a la configuración de la raspberry pi con el comando sudo raspi-config y se habilita la cámara. Es necesario un reinicio.

Para terminar la instalación se abre el archivo de módulos con el comando *sudo nano /etc/modules* y se ingresa la siguiente línea si no existe "*bcm2835-v412*" y por último, se reinicia la raspberry pi.

Se procede a instalar y configurar motion. Se realizan los siguientes pasos descritos a continuación:

1. Para instalarlo se hace uso del siguiente comando: *sudo apt-get install motion*
2. Después se edita el archivo de configuración motion.conf lanzado esta línea: *sudo nano /etc/motion/motion.conf*
3. Se modifican estas líneas del archivo y se configuran estos parámetros:
    - daemon on
    - stream_localhost off
    - output_pictures off
    - ffmpeg_output_movies off
    - stream_maxrate 100 (Esto permitirá la transmisión en tiempo real, pero requiere más ancho de banda y recursos)
    - framerate 100 (Esto permitirá capturar 100 fotogramas por segundo)
    - ancho 640 (Esto cambia el ancho de la imagen mostrada)
    - altura 480 (Esto cambia la altura de la imagen mostrada)
4. Se configura el daemon y para ello se modifica el archivo de movimiento con el siguiente comando: *sudo nano /etc/default/motion*
5. Se busca la siguiente línea y se activa: *start_motion_daemon=yes*

Para activar el servicio de motion y empezar a grabar se usa el siguiente comando:

sudo service motion start

Para desactivar el servicio:

sudo service motion stop

Para acceder al contenido que se está grabando se usa la dirección ip de la raspberry y el puerto 8081. En este caso es 192.168.0.16:8081. [39]

En la plataforma web del usuario se integra un espacio para visualizar el video en streaming, tal y como se muestra en el figura 70.

Figura 70. Modo de visualizar el contenido grabado en tiempo real de la cámara.

4.3.10. Sistema de detección de intrusos

Se diseña un sistema para detectar movimientos sospechosos en la vivienda cuando no hay nadie. Se encarga de enviar un aviso y una foto del entorno al usuario mediante Telegram. Para ello es necesario el sensor PIR HC-SR501 expuesto en el apartado 2.1 y el servicio de motion y el módulo de la cámara implementado en el apartado 4.3.9. A continuación, se muestra en la figura 71 el circuito diseñado para implementar esta funcionalidad:

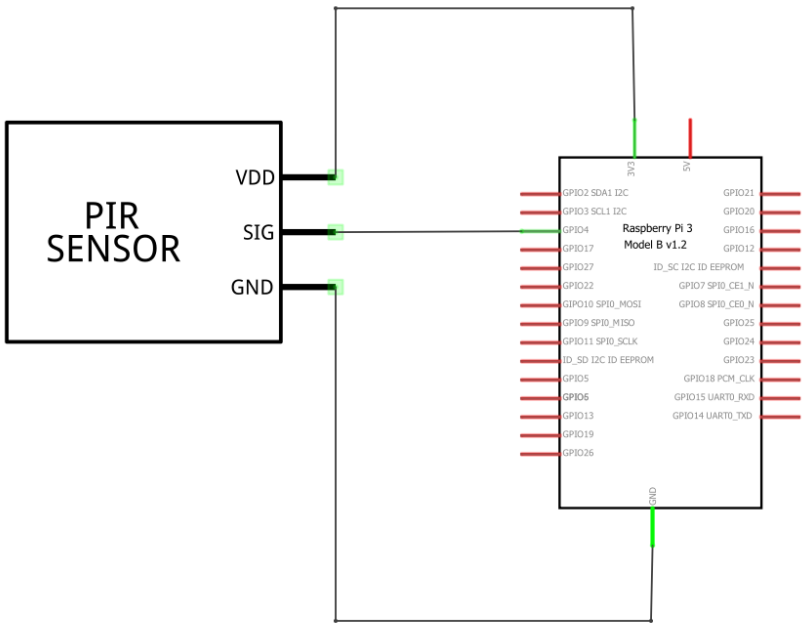


Figura 71. Circuito diseñado para un sistema de detección de intrusos

Por último, se realizan los scripts en python para dotar de inteligencia al sistema. El script principal se llama sensor_intrusos.py y dentro de él al entrar en las condiciones correspondientes ejecuta los scripts desarrollados en bash activar_motion.sh y parar_motion.sh, ya que para capturar una foto hay que parar el servicio de motion que está transmitiendo en streaming continuamente. Y el script denominado tomar_foto.py, toma la foto en el momento que se detecta movimiento. En el grafo de la figura 72 se muestra los procesos de esta implementación.

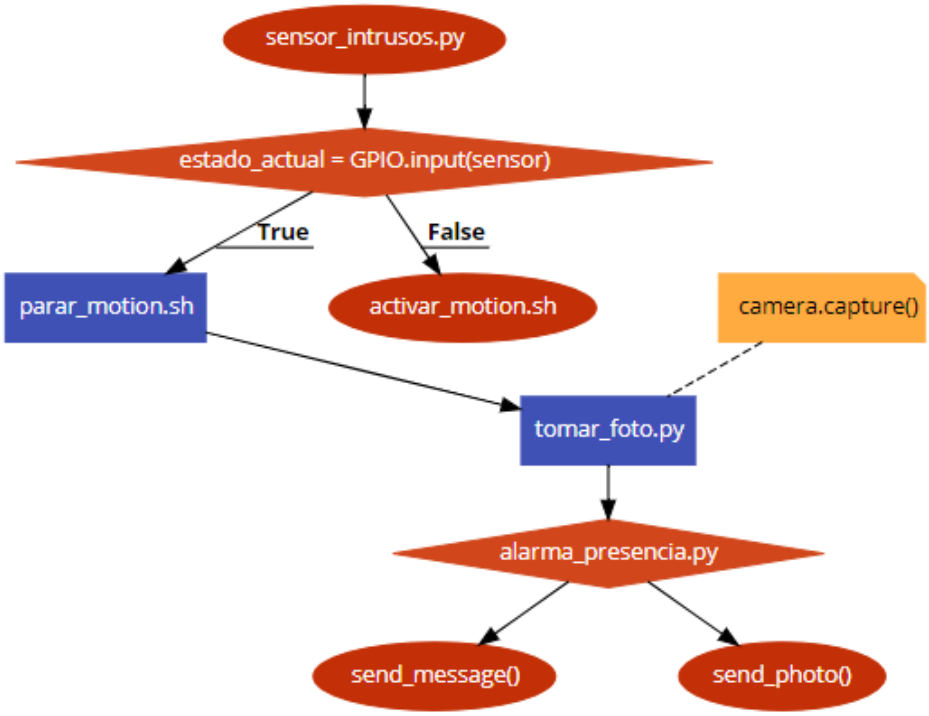


Figura 72. Diagrama de flujo del script sensor_intrusos.py para la detección de intrusos en la vivienda.

4.4. Instalación del Servidor Web

Django es un framework web que se va a utilizar para construir la aplicación web de usuario, donde se controlará todo el sistema domótico, en un servidor local, el cual, por supuesto, estará en la raspberry pi.

Proporciona un simple servidor web de desarrollo para probar de forma local aplicaciones web Django.

A continuación, se va a proceder a indicar los comandos necesarios a ingresar en el terminal de la rapsberry pi para instalar el servidor apache2 y Django:

1. Actualización del software del sistema:
   sudo apt-get update
   sudo apt-get upgrade

2. Instalar apache2:
   sudo apt-get install apache2 –y

3. Instalar Pip y Django:
   sudo apt-get install python-setuptools python-dev build-essential
   sudo easy_install pip
   sudo pip install django==1.10.3

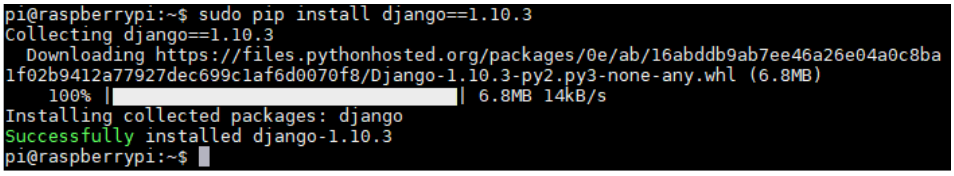


Figura 73.Resultado de instalar correctamente el paquete de django

Para comprobar que apache está corriendo, ingresamos la dirección ip de la raspberry pi en el navegador y tiene que salir este resultado, tal y como se muestra en la figura 74:

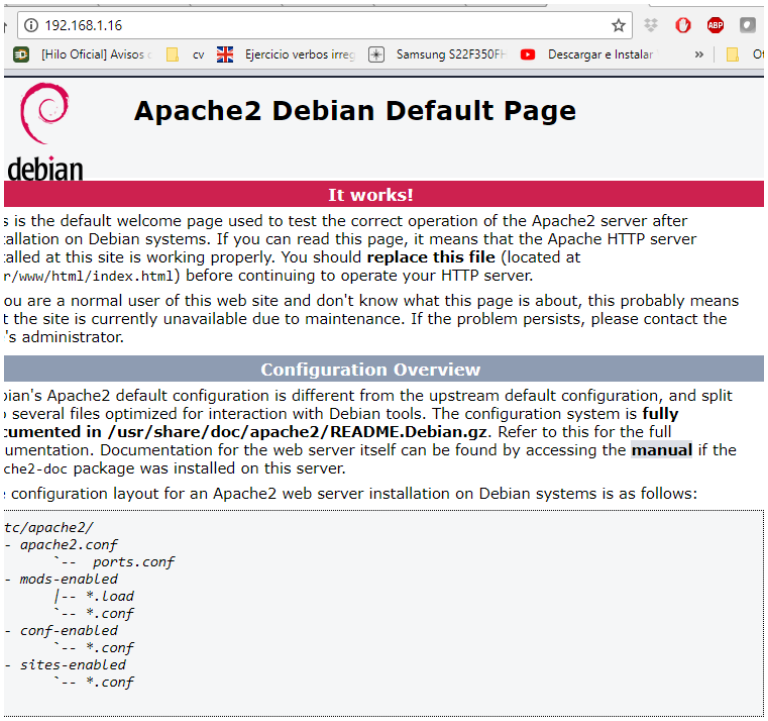


Figura 74. Captura de pantalla donde se comprueba el correcto funcionamiento del servidor apache

4. Para comenzar un proyecto en Django, primero se crea un directorio: *mkdir Dev && cd Dev* y nombramos el proyecto: *mkdir indovy && cd indovy*

5. Se crea un entorno virtual lanzando el siguiente comando para instalar los paquetes necesarios: *sudo pip install virtualenv.*

```
pi@raspberrypi:~/Dev/indovy$ sudo pip install virtualenv
Collecting virtualenv
  Downloading https://files.pythonhosted.org/packages/ed/ea/e20b5cbebf45d3096e8138ab74e
da139595d827677f38e9dd543e6015bdf/virtualenv-15.2.0-py2.py3-none-any.whl (2.6MB)
    100% |████████████████████████████████| 2.6MB 16kB/s
Installing collected packages: virtualenv
Successfully installed virtualenv-15.2.0
pi@raspberrypi:~/Dev/indovy$
```

Figura 75. Instalación de un entorno virtual correcto

6. Se instala el intérprete de Python 3: *virtualenv -p python3*

```
pi@indovy:~/Dev/indovy $ virtualenv . -p python3
Running virtualenv with interpreter /usr/bin/python3
Using base prefix '/usr'
New python executable in /home/pi/Dev/indovy/bin/python3
Also creating executable in /home/pi/Dev/indovy/bin/python
Installing setuptools, pip, wheel...done.
```

Figura 76. Instalación correcta de intérprete de python 3

7. Se activa el entorno virtual, el cual indica que se ha activado poniendo el nombre del proyecto en paréntesis: *source bin/activate*

```
pi@indovy:~/Dev/indovy $ source bin/activate
(indovy) pi@indovy:~/Dev/indovy $ python
Python 3.5.3 (default, Jan 19 2017, 14:11:04)
[GCC 6.3.0 20170124] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
```

Figura 77. Activar el entorno de python y comprobar la versión de python utilizada

Lanzando el comando python en la terminal, se activa el Shell del mismo y se puede comprobar la versión que está usando, en este caso python 3.5.3.

8. Dentro del entorno virtual, instalamos Django y se empieza un proyecto mediante estos comandos:
   pip install django==1.10.3
   django-admin.py startproject indovy

9. Se mueve el directorio de esta manera:
   mv /home/pi/Dev/indovy/indovy /home/pi/Dev/indovy/src

10. Para configurar el servidor se accede a la configuración de apache ingresando el siguiente comando: *sudo nano /etc/apache2/sites-available/000-default.conf* y se configura el python-path con el directorio del proyecto y el paquete de python con la versión con la que se está trabajando.

```
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com

    ServerName www.example.com

    ServerAdmin webmaster@localhost

    Alias /static /home/pi/Dev/indovy/static
        <Directory /home/pi/Dev/indovy/static>
           Require all granted
         </Directory>

    <Directory /home/pi/Dev/indovy/src/indovy>
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>

    WSGIDaemonProcess indovy python-path=/home/pi/Dev/indovy/src:/home/pi/Dev/indovy/lib/python3.5/site-packages
    WSGIProcessGroup indovy
    WSGIScriptAlias / /home/pi/Dev/indovy/src/indovy/wsgi.py


    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Figura 78. Configuración realizada en el archivo de configuración del servidor apache

11. Dentro del directorio /home/pi/Dev/indovy/src, se realiza la migraciones necesarias, ejecutando el archivo “manage.py”: *python manage.py migrate*

```
(indovy) pi@raspberrypi:~/Dev/indovy/src$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying sessions.0001_initial... OK
(indovy) pi@raspberrypi:~/Dev/indovy/src$
```

Figura 79. Primeras migraciones de Django

12. Se reinicia apache y se comprueba si el servidor local está funcionando con el siguiente comando: *sudo service apache2 restart*

Y aparece el siguiente error:

Figura 80. Error DisallowedHost al instalar un servidor Django por primera vez.

Este error aparece porque hay que dar permiso a la ip de la raspberry en el archivo de setting.py de Django. Por lo tanto, en el directorio /Dev/indovy/src/indovy y en el archivo anteriormente mencionado, se busca el campo “ALLOWED_HOST”, se añade la ip y se guardan cambios. La configuración es tal y como se muestra en la figura 81:

```
# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/1.10/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'yx-)5m25a%r!6$j2o9v8=(zi56o*58u&^pj1^c&wjav3@8kmf4'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = ['192.168.1.16']


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
```

Figura 81. Configuración del archivo settings.py en el campo ALLOWED_HOSTS

Por último, se vuelve a reiniciar apache y se comprueba que el entorno de desarrollo de Django funciona correctamente, el resultado que se obtiene se ilustra en la figura 82:

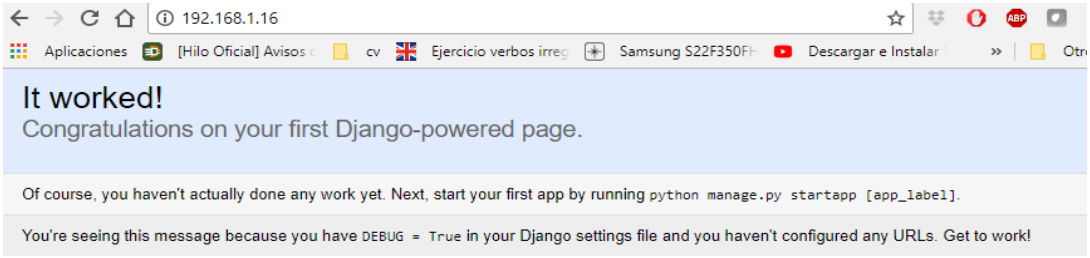


Figura 82. Entorno de Django instalado y configurado correctamente

4.5. Aplicación web de usuario

En este apartado se va a exponer como se han desarrollado las distintas partes de la aplicación web de usuario. Se puede diferenciar en varias tareas, la implementación de un login para los usuarios, un panel de control para las luces, cochera y el modo vacaciones, monitorización de la temperatura de la vivienda y video en streaming de la cámara de seguridad.

Para un mejor entendimiento, se muestra en la figura 83 un esquema de cómo se estructura el proyecto de Django realizado:

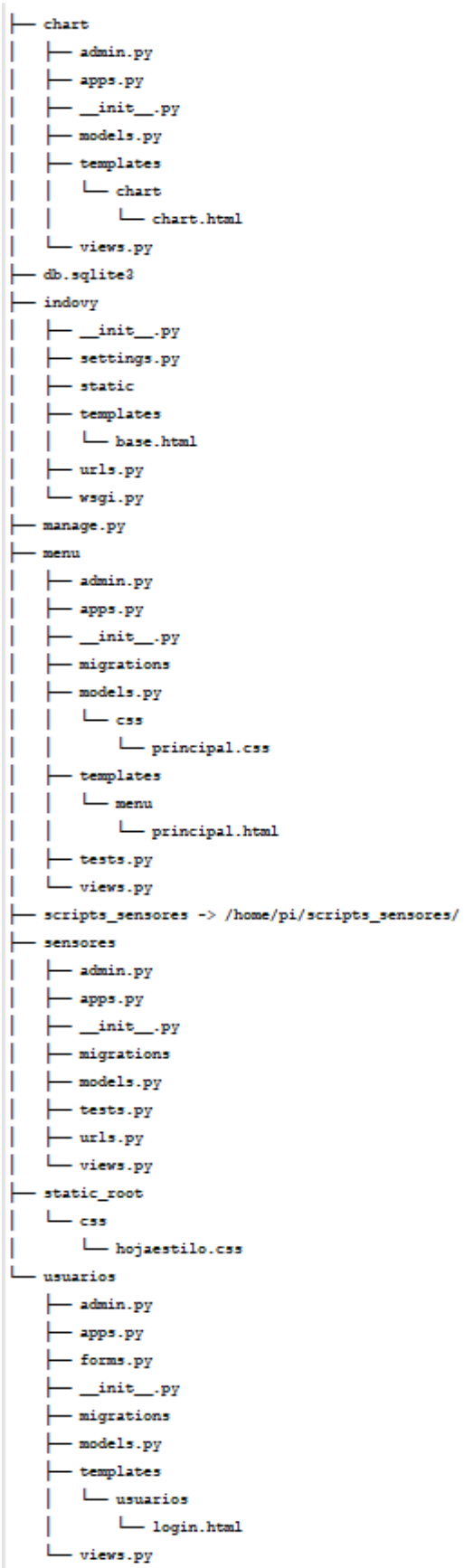


Figura 83. Estructura final del proyecto Django realizado

Se procede a explicar cómo está formado un proyecto Django, poniendo de ejemplo este proyecto en particular. Principalmente, se tiene el paquete principal del proyecto llamado indovy, dentro del cual se encuentran varios archivos:

- **__init__.py :** esto indica a python que indovy es un paquete de este.
- **settings.py :** es el archivo de configuración del proyecto.
- **urls.py :** se define las urls disposibles que va a tener el sitio web.
- **wsgi.py :** para puesta en marcha en producción con servidores.

Fuera del paquete principal del proyecto se encuentra un archivo denominado manage.py, es un archivo de utilidades de Django muy útil.

Como se puede observar en la figura anterior según se han necesitado se han creado varias aplicaciones, ya que Django organiza las distintas funcionalidades que necesitas en aplicaciones, las cuales se han denominado menú, usuario, sensores y chart.

Para crear una aplicación en Django se realizan los siguientes pasos:

1. Mediante la terminal lanzamos el siguiente comando: *python manage.py startapp NOMBRE DE APP*, por ejemplo en uno de los casos se ha ejecutado python manage.py startapp menu. Se crea un directorio que contiene los scripts para programar las vistas, modelos y templates necesarios, tal y como se muestra en el figura 84.

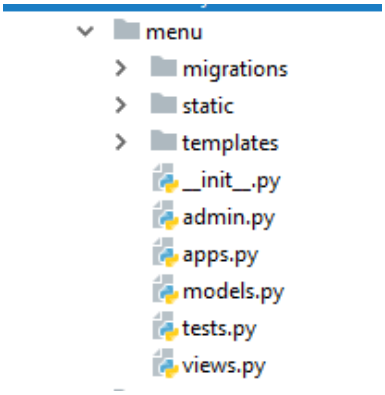


Figura 84. Estructura de una app de Django

2. Hay que añadir la aplicación creada al archivo settings.py del proyecto. Se busca la línea INSTALLED_APPS y se añade el nombre a dicha tupla, tal y como se muestra en la figura 85.

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'menu',
    'usuarios',
    'indovy',
    'sensores',
    'chart',
    'rest_framework',
    'scripts_sensores'
]
```

Figura 85. Tupla INSTALLED_APPS del archivo settings.py

Estos dos pasos descritos anteriormente hay que repetirlos cada vez que se quiera comenzar una nueva aplicación en un proyecto django.

Las apps de Django también tienen su estructura, ya que cuando este se da de "alta", se crea un paquete python llamado del mismo modo que la aplicación, dentro del cual se encuentra estos archivos:

- **__init__.py :** indica que es un paquete Python.
- **admin.py :** configuración para el admin de Django.
- **models.py :** descripción de los modelos.
- **test.py :** archivo para escribir los tests de la aplicación
- **views.py :** aquí se escribe los controladores

Por último, se expone la funcionalidad de las distintas aplicaciones que se ha creado:

- **Menú**: contiene la implementación de los distintos botones de la interfaz de usuario y el html del menú principal.
- **Usuarios**: se desarrolla la interfaz para el login y logout del usuario.
- **Chart**: se hace uso de una librería chart.js para monitorizar la temperatura de la vivienda.
- **Sensores**: encargado de que cada botón obtenga una funcionalidad concreta y se comunique con cada uno de los sensores.

4.5.1. Panel de control

Se realiza un panel general donde el usuario registrado puede apagar y encender las luces de cada habitación, abrir la puerta del garaje, un registro de la temperatura y configurarla, ver la cámara de seguridad en tiempo real y activar el modo vacaciones cuando el inquilino sale de la vivienda.

Se procede a explicar cómo se ha implementado la página principal de la plataforma web. En el archivo models.py se crean dos modelos llamados luz y sensores, con los cuales se dan de alta en el administrador de Django los datos de los diferentes sensores que se han implementado en la vivienda. Notar que cuando se crean modelos, hay que realizar una migración, lanzando el siguiente comando: *python manage.py migrate*. En la figura 86 se puede observar el archivo models.py desarrollado.

```python
from django.db import models
#from __future__ import unicode_literals

# Create your models here.

class luz(models.Model):

    habitacion = models.CharField(max_length=150)
    description = models.TextField(blank=True, null=True, default="")
    created_at = models.DateTimeField(auto_now_add=True)
    modified_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.habitacion

class sensores(models.Model):

    elemento = models.CharField(max_length=150)
    sensor = models.CharField(max_length=150)
    description = models.TextField(blank=True, null=True, default="")
    created_at = models.DateTimeField(auto_now_add=True)
    modified_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.elemento
```

Figura 86. Archivo models.py de los modelos creados en la app de menu

Ahora en el archivo views.py se obtienen ciertos datos necesarios de los modelos creados y se renderizan en una plantilla HTML llamada principal.html.

No hay que olvidar crear la url en el archivo urls.py del proyecto, añadiendo esta línea: *url(**r'^$'**, Principal.as_view(), name='**Principal**')*

Con esto se hace un despliegue de botones para activar y desactivar sensores, pero falta dotarlos de funcionalidad. Para ello se crea una nueva aplicación llamada sensores en la cual se programan distintas funciones en el archivo views.py las cuales se encargan de dotar interacción, junto a javascript en la plantilla principal.html, entre la plataforma y los elementos en la vivienda. A continuación, se enumera las distintas funciones implementadas:

- encender(pin): activa el pin del led que se desea encender
- apagar(pin): desactiva el pin del led que se desea apagar
- persiana_garaje_subir(servopin): activa el pin para subir la persiana del garaje, dicho código está explicado en el apartado 4.3.6.
- persiana_garaje_bajar(servopin): desactiva el pin para bajar la persiana del garaje, dicho código está explicado en el apartado 4.3.6.

- sensor_intrusos_activo(sensor): activa el sensor de presencia que avisa al usuario de movimiento sospechosos en la vivienda mediante telegram, se puede consultar el código con más detalle en el apartado 4.3.10
- sensor_intrusos_desactivo(sensor): desactiva el sensor de presencia y activa el servicio de motion de la cámara de seguridad.
- medir_temperatura(): recoge la temperatura del sensor DTH-11 y la devuelve.
- luz_salon_on(request): llama a la función encender(pin) e indica el pin que se debe activar.
- luz_cocina_on(request): llama a la función encender(pin) e indica el pin que se debe activar.
- luz_aseo_on(request): llama a la función encender(pin) e indica el pin que se debe activar.
- luz_pasillo_on(request): llama a la función encender(pin) e indica el pin que se debe activar.
- luz_dorm1_on(request): llama a la función encender(pin) e indica el pin que se debe activar.
- luz_dorm2_on(request): llama a la función encender(pin) e indica el pin que se debe activar.
- luz_entrada_on(request): llama a la función encender(pin) e indica el pin que se debe activar.
- luz_salon_off(request): llama a la función encender(pin) e indica el pin que se debe activar.
- luz_cocina_off(request): llama a la función apagar(pin) e indica el pin que se debe apagar.
- luz_aseo_off(request): llama a la función apagar(pin) e indica el pin que se debe apagar.
- luz_pasillo_off(request): llama a la función apagar(pin) e indica el pin que se debe apagar.
- luz_dorm1_off(request): llama a la función apagar(pin) e indica el pin que se debe apagar.
- luz_dorm2_off(request): llama a la función apagar(pin) e indica el pin que se debe apagar.

- luz_entrada_off(request): llama a la función apagar(pin) e indica el pin que se debe apagar.
- vacaciones_on(request): con esta función activa el modo vacaciones el cuál se encargar de la seguridad de la vivienda cuando el usuario está ausente simulando encendido y apagado de luces cómo si hubiera actividad dentro y llama a la función sensor_intrusos_activo(sensor).
- vacaciones_off(request): desactiva la funcionalidad de modo vacaciones cuando el usuario lo desee. Llama a la función sensor_intrusos_desactivo(sensor).
- garage_on(request): llama a la función persiana_garaje_subir(servopin) para activar el pin que sube la persiana.
- garage_off(request): llama a la función persiana_garaje_bajar(servopin) para activar el pin que baja la persiana.
- set_temperatura_esperada(request): llama a la función medir_temperatura(), la cual le devuelve la temperatura actual que luego es comparada con la temperatura deseada que el usuario introduce en la web y dependiendo de ello se activa o desactiva el ventilador.
- aire_off(request): se activa el ventilador cuando el usuario lo desea.
- aire_on(request): se desactiva el ventilador cuando el usuario lo desea.

Cada vista se relaciona con su url correspondiente por ello se crea un archivo llamado urls.py donde se añaden, tal y como se muestra en la figura 87.

```
urlpatterns = [
    url(r'^vacaciones_on/$', views.vacaciones_on, name='vacaciones_on'),
    url(r'^vacaciones_off/$', views.vacaciones_off, name='vacaciones_off'),
    url(r'^garage_off/$', views.garage_off, name='garage_off'),
    url(r'^garage_on/$',views.garage_on, name='garage_on'),
    url(r'^luz_cocina_on/$',views.luz_cocina_on, name='luz_cocina_on'),
    url(r'^luz_cocina_off/$',views.luz_cocina_off, name='luz_cocina_off'),
    url(r'^luz_salon_on/$',views.luz_salon_on, name='luz_salon_on'),
    url(r'^luz_salon_off/$',views.luz_salon_off, name='luz_salon_off'),
    url(r'^luz_dorm1_on/$',views.luz_dorm1_on, name='luz_dorm1_on'),
    url(r'^luz_dorm1_off/$',views.luz_dorm1_off, name='luz_dorm1_off'),
    url(r'^luz_dorm2_on/$',views.luz_dorm2_on, name='luz_dorm2_on'),
    url(r'^luz_dorm2_off/$',views.luz_dorm2_off, name='luz_dorm2_off'),
    url(r'^luz_aseo_on/$',views.luz_aseo_on, name='luz_aseo_on'),
    url(r'^luz_aseo_off/$',views.luz_aseo_off, name='luz_aseo_off'),
    url(r'^luz_pasillo_on/$',views.luz_pasillo_on, name='luz_pasillo_on'),
    url(r'^luz_pasillo_off/$',views.luz_pasillo_off, name='luz_pasillo_off'),
    url(r'^luz_entrada_on/$',views.luz_entrada_on, name='luz_entrada_on'),
    url(r'^luz_entrada_off/$',views.luz_entrada_off, name='luz_entrada_off'),
    url(r'^grafica_temperatura/$',views.set_temperatura_esperada, name='grafica_temperatura'),
    url(r'^aire_off/$',views.aire_off, name='aire_off'),
    url(r'^aire_on/$',views.aire_on, name='aire_on'),

]
```

Figura 87. URLs de las vistas creadas en la aplicación sensores

Haciendo uso de javascript se le indica a cada botón a la función correspondiente que tiene que llamar mediante el nombre configurado en las urls, para que cada botón se realice la tarea que debe en el sensor correspondiente. Se muestra un ejemplo en la figura 88:

```
<p> {{ luz }} </p>

<button class="button" id="salonON" data-url="{% url "luz_salon_on" %}">ON</button>
<button class="button" id="salonOFF" data-url="{% url "luz_salon_off" %}">OFF</button>
<script type="text/javascript">
        $('#salonON').click(function(){
            url = $(this).attr('data-url');
            $.get(url, function(data) {})
        })
</script>
<script type="text/javascript">
        $('#salonOFF').click(function(){
            url = $(this).attr('data-url');
            $.get(url, function(data) {})
        })
</script>

<p>   </p>
```

Figura 88. Javascript para dotar de interactividad el botón de ON y OFF de la luz del salón

De esta manera, ya se tiene un panel dotado con las distintas funcionalidades que se necesitan para controlar la vivienda en remoto.

4.5.2. Monitorización temperatura

Para monitorizar la temperatura que recoge el sensor DTH11, se hace uso de una librería de javascript llamada Chart.js, la cual permite crear todo tipo de gráficos sobre canvas HTML5.

Se procede a explicar cómo implementar esta librería en Django. Lo primero es instalar django-chartjs a través del terminal lanzando el comando *pip install django-chartjs*, crear la aplicación llamada chart y añadirla al settings del proyecto.

En la vista se crea una función la cual recoge los datos de temperatura y hora actual de los archivos de texto que crea el script dht_console.py y los devuelve para luego hacer referencia a ellos en el código javascript que se añade en principal.html, tal y cómo se muestra en la figura 89:

```
<h1 class="titulos"> {{ sensores }} </h1>
<canvas id="myChart" width="500" height="400"></canvas>
<script type="text/javascript">
    function refresca_grafica() {
        $.get('{% url "line_chart_json" %}', function(data) {
          console.log("data");
          console.log(data);
            var ctx = $("#myChart").get(0).getContext("2d");
            new Chart(ctx, {
                type: 'line', data: data
            });
        });
    }
    setInterval(function(){refresca_grafica()}, 15000)
</script>
```

Figura 89. Javascript implementado para realizar la gráfica de temperatura.

Por último, se añade a la interfaz una funcionalidad para que el usuario configure la temperatura ideal de la vivienda. Para ello se setea el valor del campo con javascript y se recoge el valor con el método get en la función set_temperatura_esperada() descrita anteriormente.

La figura 90 muestra el resultado final de esta implementación:

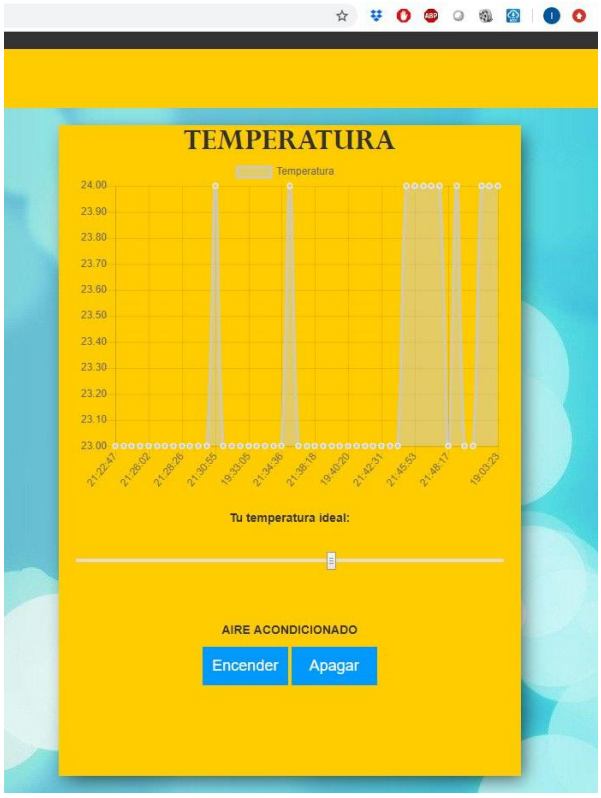


Figura 90. Monitorización y configuración de temperatura en la interfaz web

4.5.3. Implementación del login y logout

En la plataforma web es necesario implementar un login para que pueda loguear con su usuario cada inquilino y por seguridad.

Lo primero es crear la aplicación llamada usuarios y añadirla al settings del proyecto. Se creará un directorio e implementamos un formulario en un archivo de python llamado forms.py donde se importa el modelo forms y se crea una clase llamada LoginForm(forms.Form) , ya que un formulario de django es un clase que tiene que heredar de la clase Form, en ella se crean los campos de usuario y contraseña de tipo CharField y con un parámetro denominado label, además en el campo de contraseña se le configura un widget para que la pinte con asteriscos y no en texto plano.

En el archivo views.py, el controlador, hay que instanciar el formulario y pasarlo a la plantilla dentro del contexto. Se implementa dos clases llamadas Login y Logout.

Dentro de clase Login, se realizan dos funciones get y post, en ambas se ha instanciado el formulario y mensajes de error, los cuales se han pasado al contexto y redenrizado en una plantilla html. Notar que en la función post que se han desarrollado condiciones para validar el formulario.

La clase Logout es mas sencilla de desarrollar, solo había que tener en cuenta si el usuario está registrado o no, en caso de que no estuviera se redirege al formulario de login.

Hay que crear la plantilla HTML llamada login.html, para ello se crea una carpeta nombrada como templates, dentro de la cual se crea otra denominado usuarios, para evitar conflictos de nombres, y en ella se crea el archivo HTML y se desarrolla tal y como se muestra en la figura 91:

```
{% extends 'base.html' %}
{% load staticfiles %}
{% block title %} Login {% endblock %}
{% block section %}
    {% for error in errors %}
        <p>{{ error }}</p>
    {% endfor %}
    <div class="login_pos">
        <div id="general_login">
            <form class="formulario" method="post">
                {% csrf_token %}
                {{ login_form.as_p }}
            <div>
                <button class="boton" type="submit">Entrar</button>
            </div>

            </form>
        </div>
    </div>
{% endblock %}
```

Figura 91. Plantilla del formulario de login

Hay que dotar de seguridad el formulario para ello en la etiqueta form se pone un atributo method, para que envíe el formulario a través del método post para que a la hora del registro no se coloque el usuario y contraseña en la url del navegador.

Como se puede apreciar hay que utilizar el csrf_token, con esto se evita por ejemplo, ataques de fuerza bruta tipo post a nuestra web, ya que Django tiene implementado las típicas configuraciones básicas de seguridad de la web.

Por último, se configura las urls en el archivo urls.py dentro del directorio de nuestro proyecto.

Se observa en la siguiente figura 92, el resultado final de la implementación:

Figura 92. Interfaz web donde el usuario final introduce sus datos para loguearse.

4.5.4. Cámara de seguridad en la web

Para mostrar en la interfaz web el contenido en tiempo real de la cámara que se ha implementado en el apartado 4.3.9, sólo es necesario añadir en la plantilla principal.html la etiqueta img y añadir de atributo la dirección de la cámara, tal y como se muestra en el figura 93:

```
<div class="camara">
    <hl style="text-align:center; font-family: 'Lobster';">Cámara de seguridad</hl>
</div>
<div class=" ">
    <img style="-webkit-user-select: none;" src="http://192.168.0.16:8081/">
</div>
```

Figura 93. Código HTML para mostrar el contenido de la cámara

4.5.5. Front-end del panel web

En este capítulo se expone las herramientas utilizadas para hacer el diseño web, se ha hecho uso del framework bootstrap y los archivos estáticos de django para almacenar los archivos css.

Bootstrap facilita la maquetación de sitios web y ofrece herramientas para que se adapte en cualquier resolución de pantalla, ahorrando así mucho trabajo. Para usar este framework se ha añadido el CDN en el archivo html llamado "base.html", plantilla general creada en el proyecto principal, para usarlo de forma remota. La línea que hay que añadir es:

<link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap-theme.min.css">.

Se procede ahora a explicar cómo configurar los archivos estáticos de Django.

Se crea una carpeta llamada "static_root" en la raíz del proyecto para guardar allí los archivos estáticos en otra carpeta llamada "css" y se crea el archivo css denominado "hojaestilo.css".

A continuación, se configura la dirección de los archivos estáticos en el archivo llamado settings.py, tal y como se muestra en la figura 94:

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/1.10/howto/static-files/

STATIC_URL = '/static/'

#STATICFILS_DIRS = (os.path.join(BASE_DIR,'static'),)

STATIC_ROOT = os.path.join(BASE_DIR,'static_root')
```

Figura 94. Configuración de la dirección de los archivos estáticos en settings.py

Por último, se carga los archivos estáticos en la plantilla html, introduciendo la siguiente línea: *{% load staticfiles %}.*

Está listo para añadir en hojaestilo.css el formato que se desea a cada elemento de la web. Por ejemplo, en la figura 95 se muestra el formato que se le ha asignado a los botones con la class=button.

```css
.button {
    position: relative;
    background-color: #0099FF;
    border: none;
    font-size: 18px;
    color: #FFFFFF;
    padding: 10px;
    width: 100px;
    text-align: center;
    -webkit-transition-duration: 0.4s; /* Safari */
    transition-duration: 0.4s;
    text-decoration: none;
    overflow: hidden;
    cursor: pointer;
}

.button:after {
    content: "";
    background: #004d80;
    display: block;
    position: absolute;
    padding-top: 300%;
    padding-left: 350%;
    margin-left: -20px!important;
    margin-top: -120%;
    opacity: 0;
    transition: all 0.8s
}

.button:active:after {
    padding: 0;
    margin: 0;
    opacity: 1;
    transition: 0s
}
```

Figura 95. Formato css asignado a los botones de la interfaz web

Una vez implementado el diseño web, el resultado final de la interfaz web se muestra en la siguiente figura:

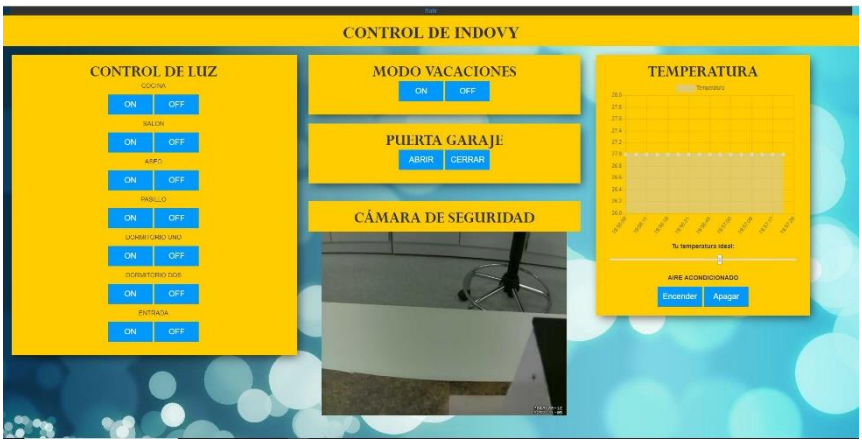


Figura 96. Plataforma Web final donde se controlan todos los sensores

5. RESULTADOS DEL PROTOTIPO

5.1. Resultado final de cada sistema

En este apartado sólo se mostrará la salida de los diferentes scripts programados para cada sistema y así verificar su correcto funcionamiento. Por otro lado, mediante la maqueta de una vivienda inteligente se expondrá como sería el funcionamiento de los sistemas trabajando en tiempo real.

5.1.1. Sistema de aire acondicionado automatizado

Se comprueba que ejecutando el script dht_console.py y configurando que se quiere una temperatura ideal de 22 º que el ventilador se activa cuando la temperatura supera el umbral establecido. Los resultados se muestran la figura 97:

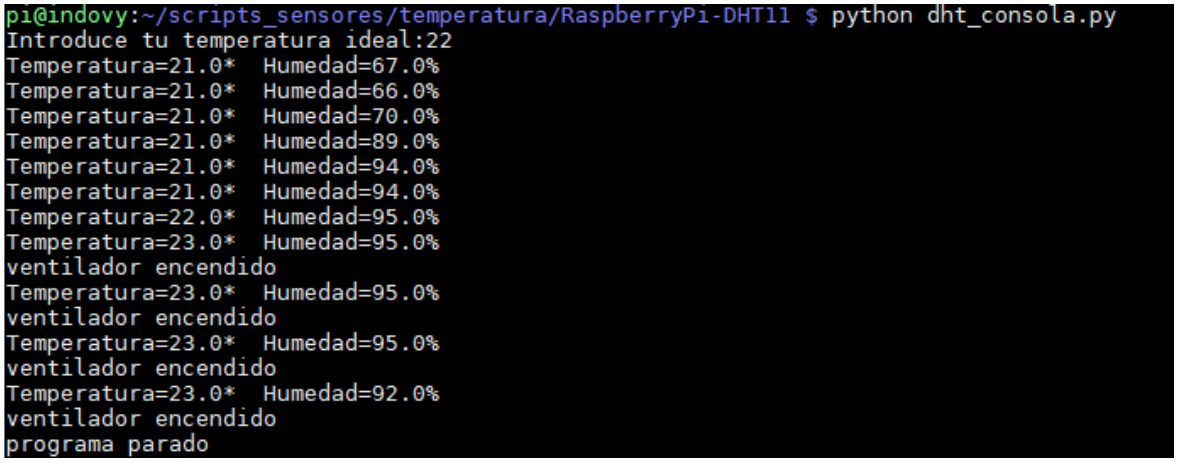


Figura 97. Salida del script dht_console.py donde se comprueba el funcionamiento del sensor de temperatura dht11

Dicha temperatura ideal la puede configurar el usuario mediante la interfaz web, a la vez, que monitoriza la temperatura actual de su vivienda expuesta en una gráfica. Haciendo uso de la herramienta para desarrolladores del navegador, se observa en la figura 98 cómo se llama correctamente a las urls, grafica_temperatura y json_grafica, que están relacionadas con las vistas de django.

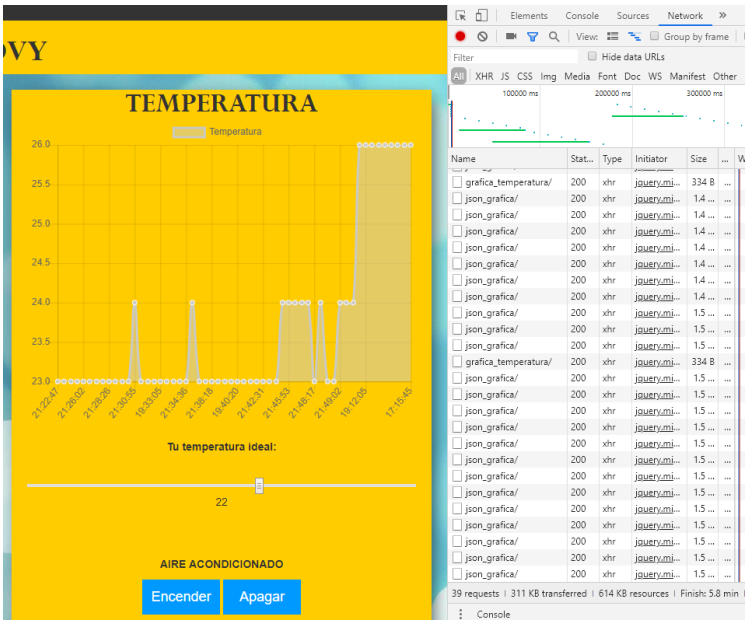


Figura 98. Comprobar mediante el navegador que se setea correctamente la temperatura y la monitorización

5.1.2. Sistema de detección de humos y otros gases tóxicos

Para comprobar que el resultado que se obtiene es correcto, se ejecuta el script llamado sensor_gas_conaviso.py, cuando el sensor MQ-135 detecte el gas del mechero debe enviar un aviso al telegram de que hay contaminación en el ambiente y además debe sonar una alarma. Los resultados se muestran en la figura 99:

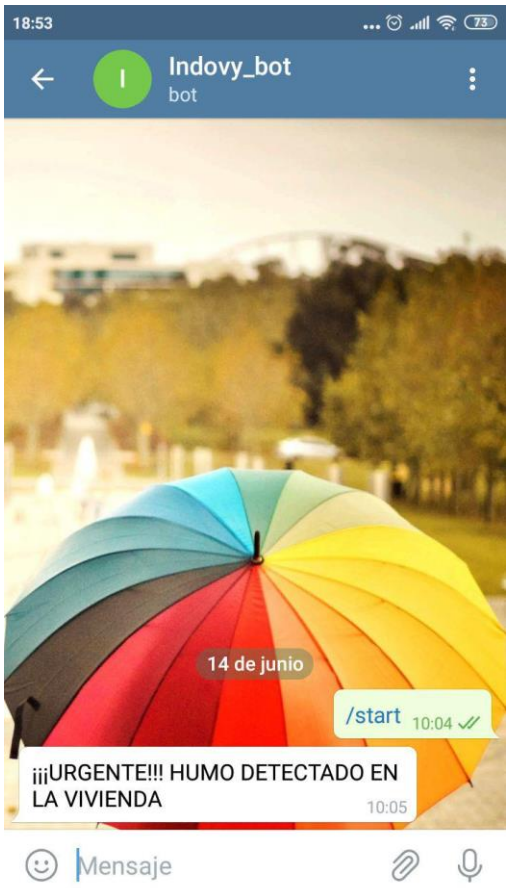


Figura 99. Resultado final que comprueba que avisa correctamente al usuario cuando detecta humo en la vivienda

5.1.3. Sistema de sensor de aparcamiento

Para comprobar que el sistema tiene un funcionamiento correcto, se ejecuta el script llamado sensor_aparcamiento.py, se comprueba la figura 100 que conforme el objeto (en un caso real sería un vehículo) se acerca a la pared, se cambia el color del led.

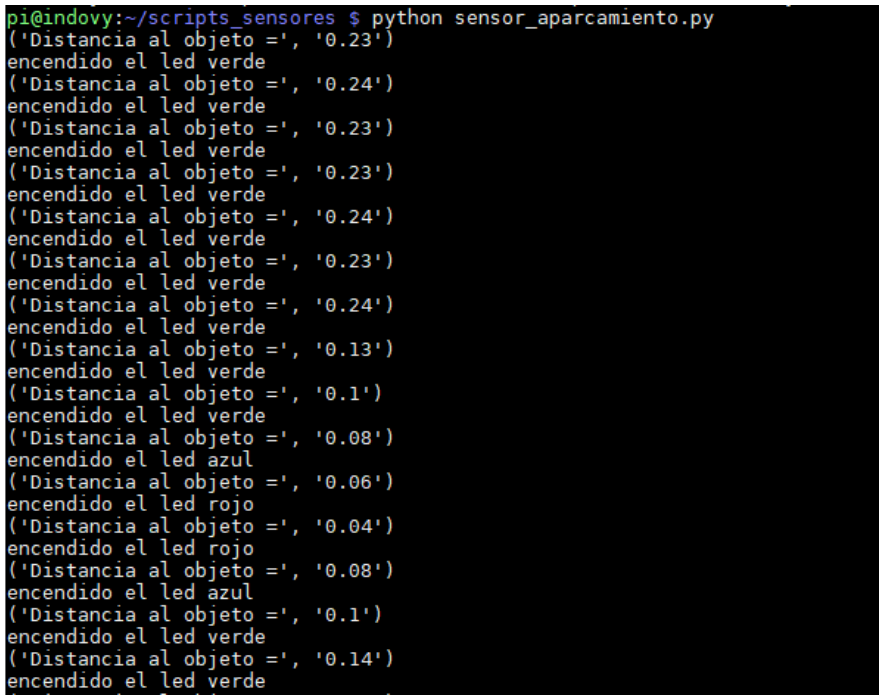


Figura 100. Salida del script sensor_aparcamiento.py donde se comprueba el correcto funcionamiento del sistema

5.1.4. Sistema de encendido de luz sensor de presencia

Se comprueba ejecutando el script encender_garajeconsensor.py que el sistema funciona correctamente. En la figura 101 se muestra que cuando el sensor detecta movimiento, cambia su estado de 0 a 1, se activa el led situado en el garaje.

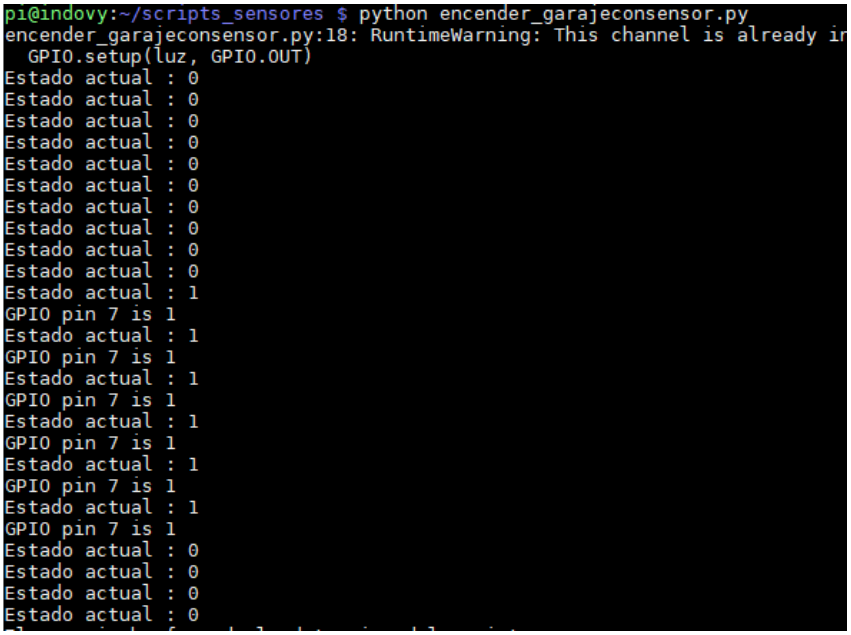


Figura 101. Salida del script encender_garajeconsensor.py donde se comprueba el correcto funcionamiento del sistema

5.1.5. Sistema de encendido de luces exteriores mediante LDR

Se ejecuta el script luz_patio.py para comprobar que cuando la zona donde el sensor LDR se oscurece las luces del patio se encienden. En la figura 102 se muestra el resultado:

Figura 102. Salida del script luz_patio.py donde se comprueba el correcto funcionamiento del sistema

5.1.6. Sistema de puerta automatizada por control remoto

Se comprueba que cuando el usuario pulsa el botón de Abrir en el apartado “puerta garaje”, se abre la puerta del garaje y lo mismo con el botón de cerrar. En la figura 103 se muestra cómo se ha llamado a la vista correctamente, mediante la opción herramienta para desarrolladores del navegador, y se ha abierto y cerrado la puerta.

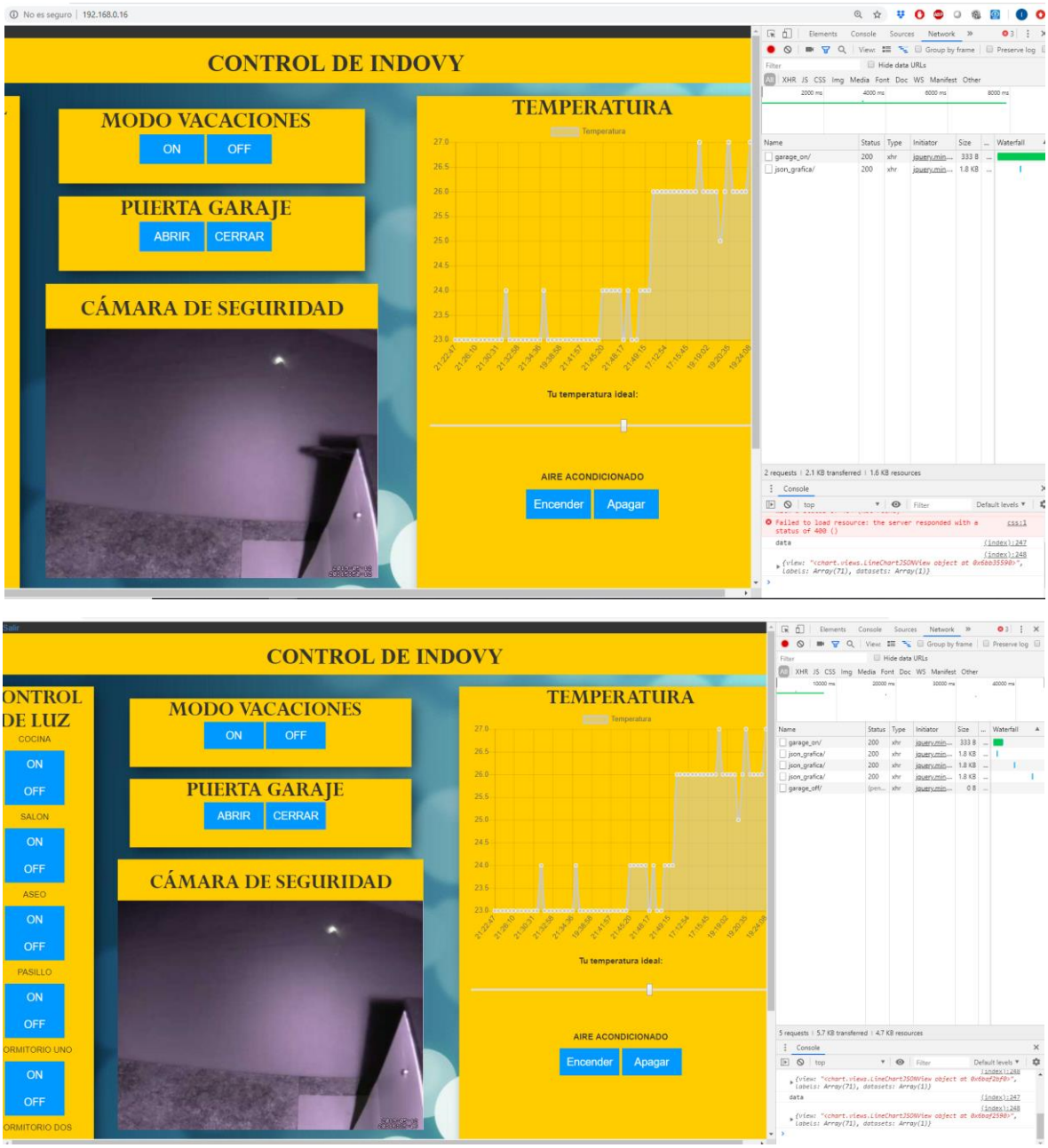


Figura 103. Se comprueba que al pulsar abrir y cerrar la puerta del garaje, se llama a la URL que ejecutará la vista correspondiente

5.1.7. Sistema de encendido de luz por voz

Se muestra la salida del script voz.py en la figura 104, para comprobar que interpreta bien lo que dice el usuario y se ejecuta la acción.

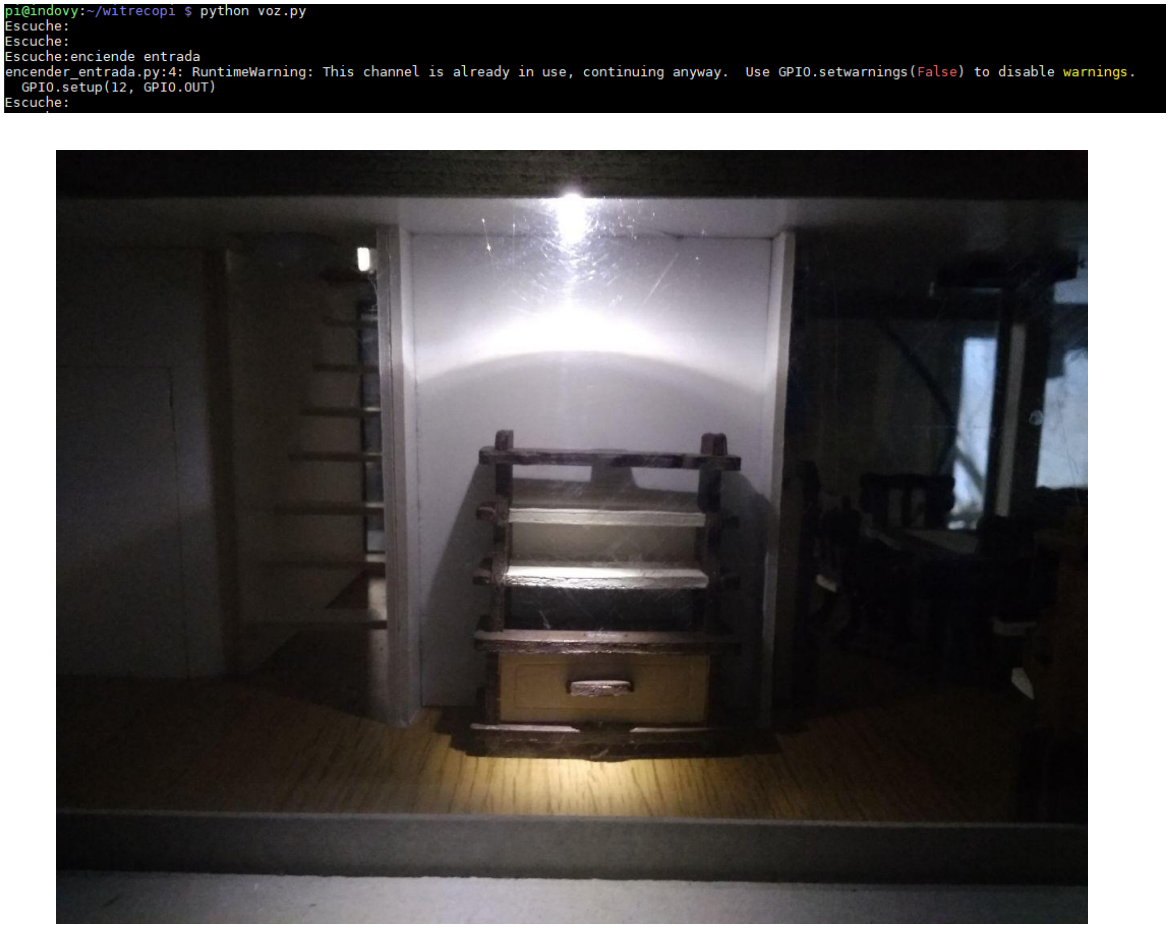


Figura 104. Salida del script voz.py y demostración de su correcto funcionamiento.

5.1.8. Sistema de luces mediante la interfaz web

Se hace uso de la herramienta para desarrolladores que proporciona el navegador para comprobar que se llama correctamente a la vista y se enciende y apagar la luz de cada habitación, tal y cómo se muestra en el figura 105.

Figura 105. Comprobación de que el pulsar el botón de ON y OFF de las luces de las habitaciones se llama correctamente a la URL relacionada con su correspondiente vista.

5.1.9. Cámara de seguridad

En la interfaz web del usuario se muestra la grabación en tiempo real que realiza la cámara, tal y como se muestra en la figura 106:

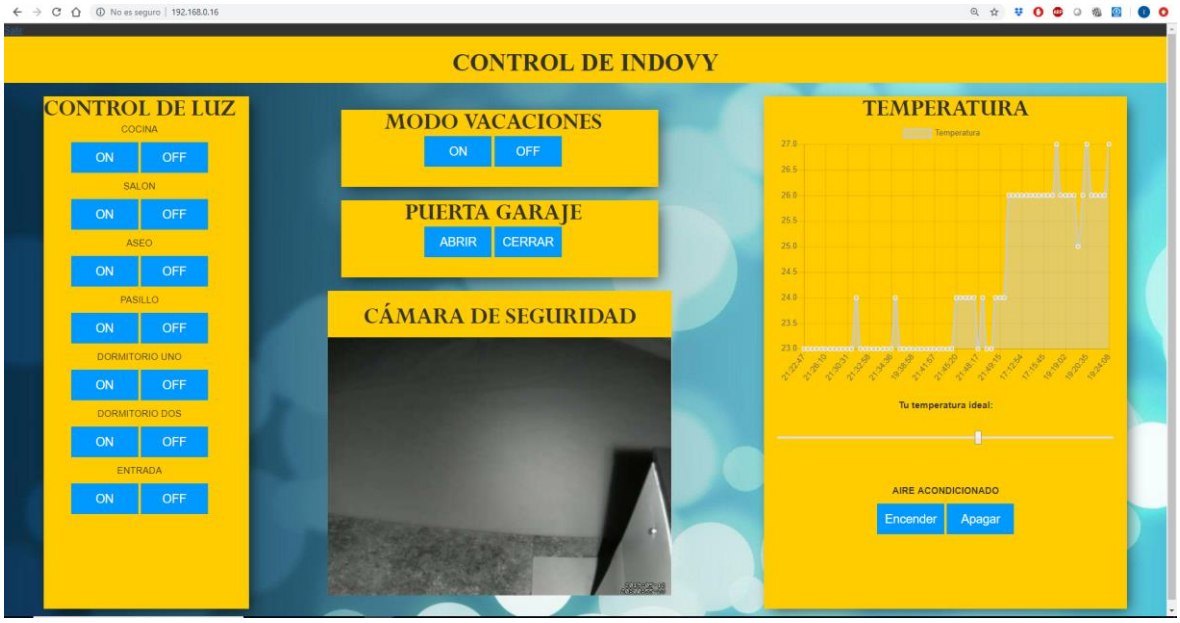


Figura 106. Muestra en la plataforma web de la cámara de seguridad

5.1.10. Sistema de detección de intrusos

Para activar la detección de intrusos, se activa el modo vacaciones mediante la plataforma web, tal y cómo se muestra en la figura 107, se puede observar que llama a la vista correctamente y se obtiene como resultado, una simulación de actividad en la vivienda encendiendo y apagando luces, a la vez que el sensor de presencia si registra un movimiento, manda un aviso al telegram de lo que está ocurriendo, como se observa en la siguiente figura 108.

Figura 107. Al pulsar el botón ON del apartado MODO VACACIONES, se llama correctamente a la URL, la cual llamará a la vista que cumple dichas funciones

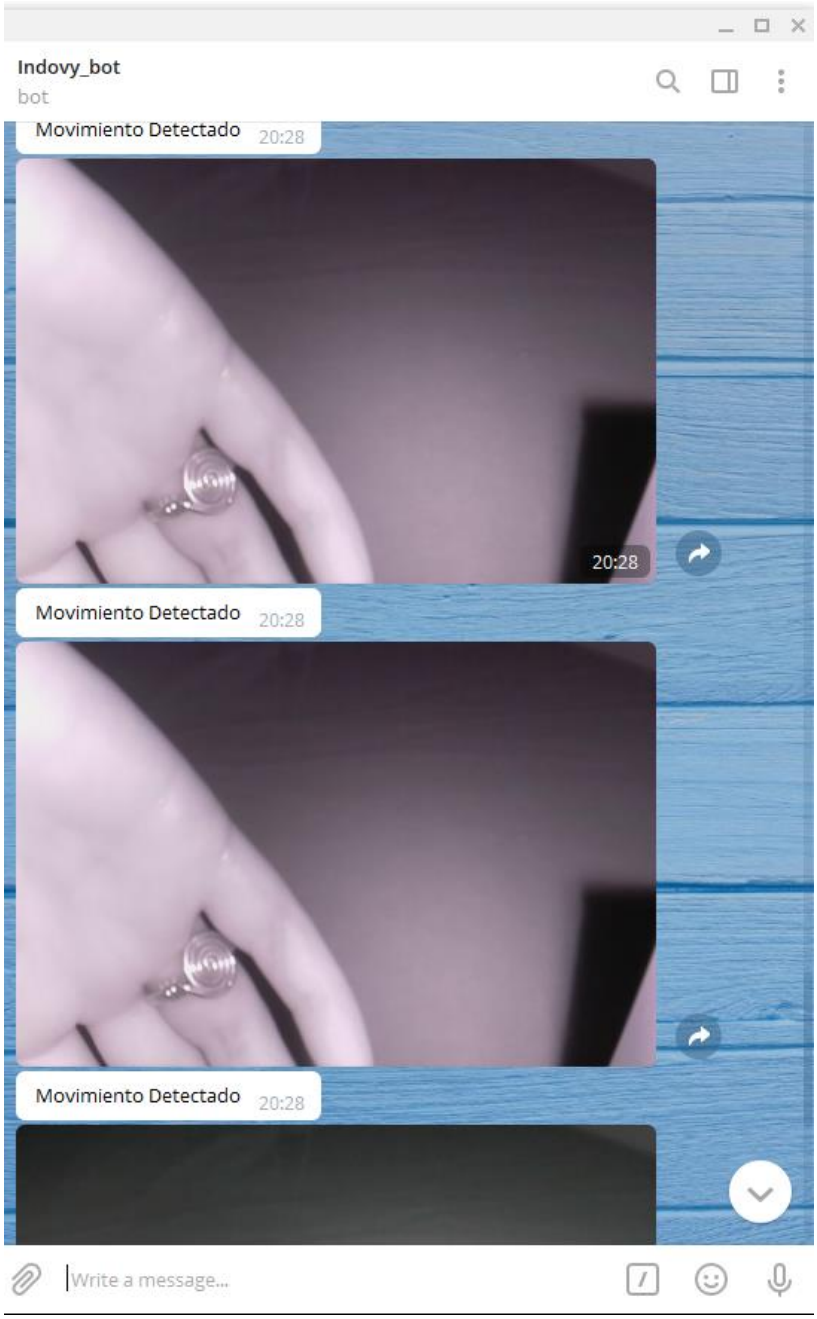


Figura 108. Aviso por telegram cuando se detecta un movimiento sospechoso al activar el MODO VACACIONES

5.2. Maqueta

Se ha realizado una maqueta de una vivienda inteligente para comprobar el funcionamiento del sistema completo.

Para la fabricación del prototipo se han realizado los planos de los que iba a ser el edificio, los cuales se pueden consultar en el anexo 8.1. Los materiales empleados han sido madera contrachapada y cartón pluma. En la figura 109 se muestra el prototipo terminado:

Figura 109. Prototipo de la vivienda terminado con todos los elementos necesarios

Cada uno de los sistemas expuestos anteriormente con sus respectivos componentes se encuentra en el tejado junto a la raspberry pi. Se pueden distinguir cuatro placaboards, la primera recoge todos los led de interior que iluminan a la vivienda, la segunda el sistema de encendido automático de las luces exteriores, la tercera es el sistema de aire acondicionado, el detector de gases tóxicos y el sistema de detección de intrusos, la cuarta recoge todos los sistemas implementados en la cochera, sensor de aparcamiento y encendido automático de luces mediante el sensor de presencia y por último, asignado como número 5 en la figura 110, el micrófono USB para dar órdenes mediante la voz.

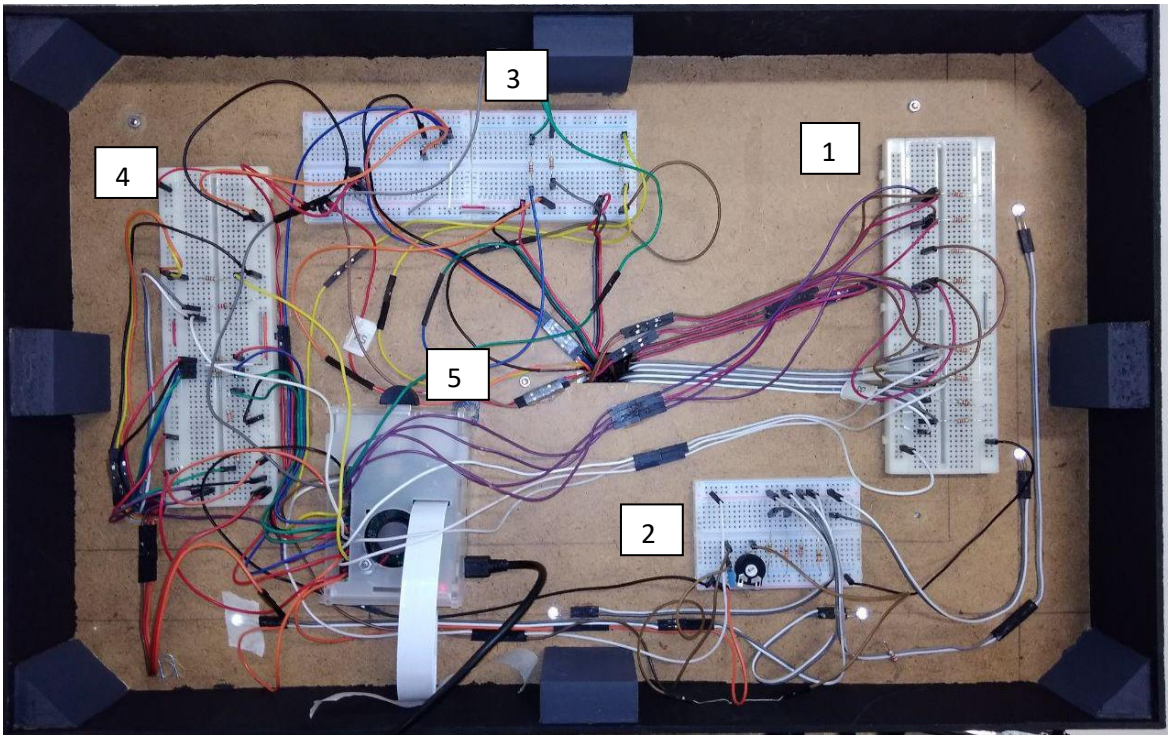


Figura 110. Circuitos implementados de los diferentes sistemas explicados

Se ha usado un criterio de colores a la hora de realizar el cableado para facilitar la identificación de cada uno de los elementos. Se expone en la tabla 4 los colores empleados en cada uno de los elementos:

Color	Patilla Sensor
Negro	GND
Rojo	Alimentación de 5 V
Naranja	Alimentación de 3 V
Verde	Pin OA del MQ-135
Azul	Pin Data del DTH-11
Morado	Patilla ánodo de los led planta 2
Marrón	Patilla cátodo de los led planta 2
Amarillo	Patilla OUT (TTL) sensor HC-SR501
Blanco	Patilla ánodo de los led planta 1
Gris	Patilla cátodo de los led planta 1

Tabla 4. Código de colores del cableado empleado en cada sensor

6. PRESUPUESTO DEL PROTOTIPO

En este capítulo se desarrolla un presupuesto del prototipo realizado para desarrollar este proyecto. En la tabla 5 se calcula los gastos de los materiales utilizados y en la tabla 6 la mano de obra realizada dividida en tres secciones, la parte de realizar la maqueta, la del desarrollo del código de programación y el diseño de los circuitos electrónicos.

Materiales	**Cantidad**	**Precio/Unidad (€)**	**Total (€)**
Raspberry pi 3 Model B	1	37,50	37,50
Resistencias	18	0,05	0,90
Condensador	1	1,25	1,25
Transistor	2	1,56	3,12
Cámara	1	28,99	28,99
Led	15	0,20	3,00
Led RGB	1	0,30	0,30
Micrófono USB	1	3,95	3,95
HC-SR501	1	2,00	2,00
MQ-135	1	3,18	3,18
Módulo de Buzzer	1	1,69	1,69
DTH-11	1	1,00	1,00
Mini ventilador	1	2,36	2,36
HC-SR04	1	1,95	1,95
Sensor LDR	1	0,25	0,25
Protoboard	5	4,95	24,75
Cajas de Cables	3	2,50	7,5
Decoración maqueta	1	18,00	18,00
Cartón pluma para estructura de maqueta	8	6,00	48
Total (IVA incluido)			189,69

Tabla 5. Presupuesto de materiales utilizados en el prototipo

Mano de obra	Horas	Coste (€/hora)	Total (€)
Desarrollo de hardware	360	60	21.600
Desarrollo de software	480	60	28.800
Desarrollo de maqueta	180	60	10.800
Total			**61.200**
Total + IVA (16%)			**70.992**

Tabla 6. Presupuesto de mano de obra de un ingeniero de titulado

7. CONCLUSIÓN

Con este proyecto se ha pretendido adentrase en el mundo del internet de la cosas, conocer sus características principales y demostrar que se puede implementar un sistema de domótica en una vivienda para mejorar la calidad de vida de un particular tanto en seguridad y confort, como en ahorro energético.

Es destacable el estudio del desarrollo de plataformas web y distintas tecnologías que aportan al usuario interacción y control sobre la vivienda.

Por último mencionar que es un proyecto escalable, se pueden añadir muchas más funcionalidades como por ejemplo: un sistema inteligente de regadío para el césped, con sensores que midan la humedad de la tierra y la activación de los aspersores cuando el nivel de humedad esté bajo; electrodomésticos que automatizan tareas, como un frigorífico que te avisa de lo que falta por comprar, una lavadora programable en remoto, persianas que se abren y cierran según la intensidad lumínica o puertas que se abren y cierran con lectores de huellas.

Hay un sinfín de posibilidades con el desarrollo del IoT y esto es lo que demuestra este proyecto, poder conocer la exponencial evolución que proviene del mundo del Internet y cómo se irán desarrollando diferentes protocolos de comunicación para adaptarse a las nuevas necesidades.

8. ANEXOS

8.1.Planos de la estructura del prototipo de la vivienda

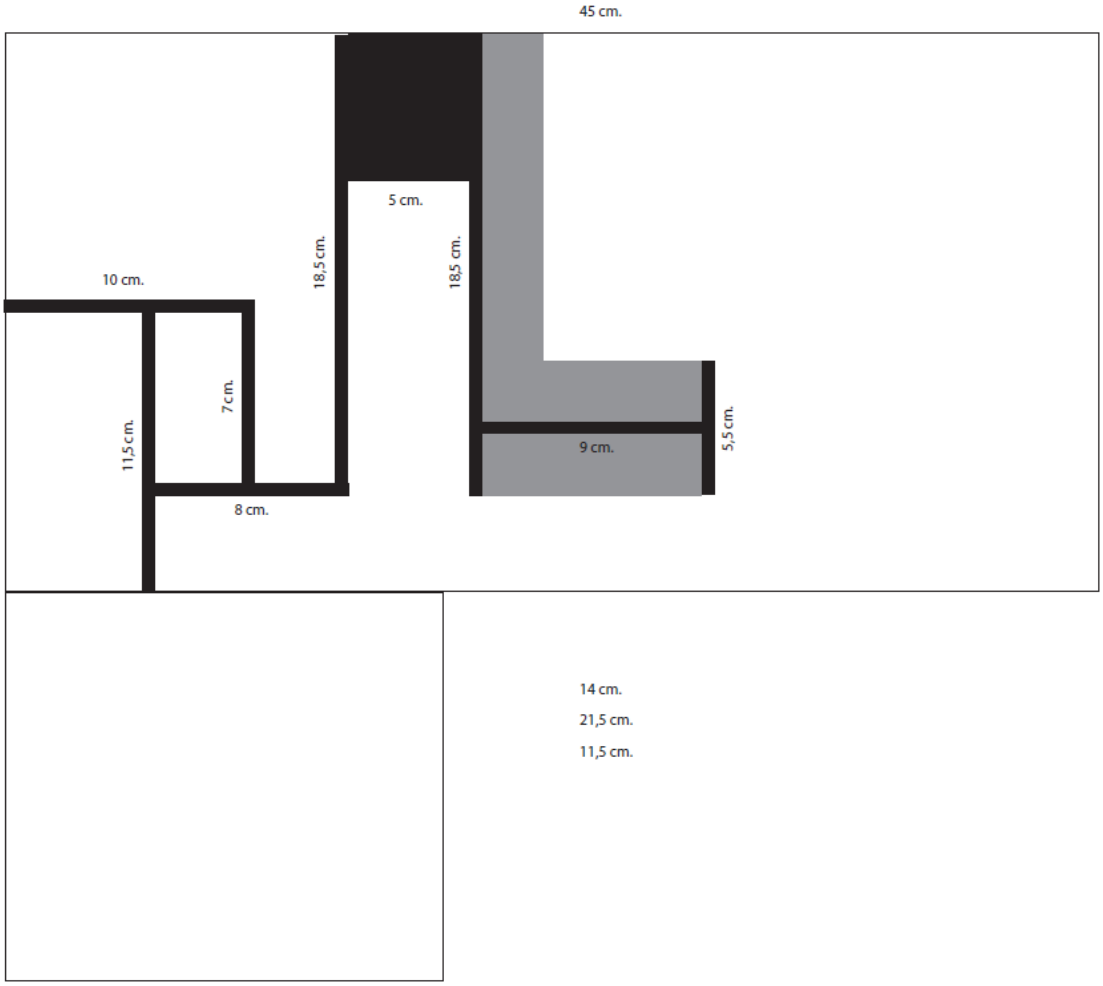


Figura 111. Plano de la estructura de la planta primera del prototipo de la vivienda

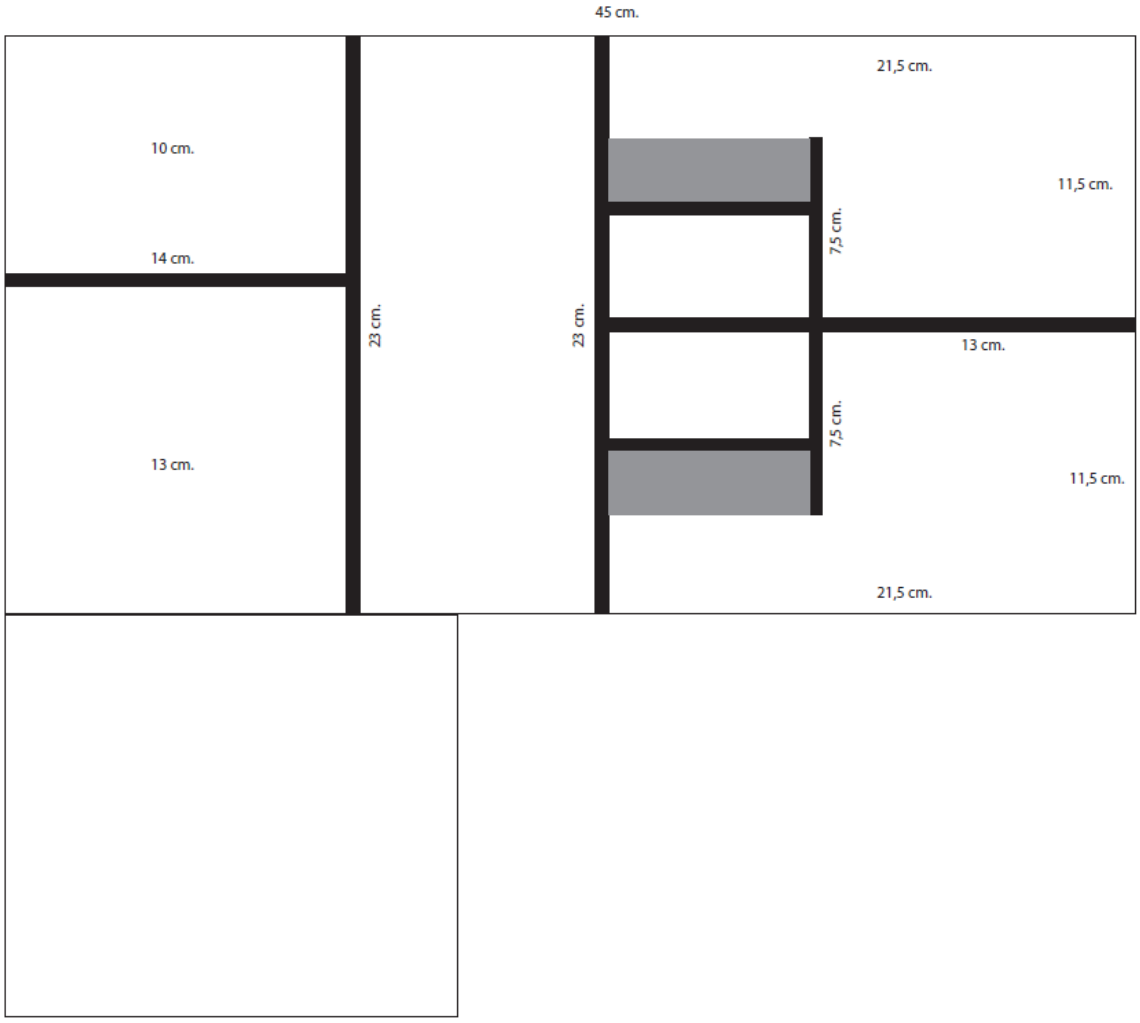


Figura 112. Plano de la estructura de la planta segunda del prototipo de la vivienda

8.2.Manual de usuario

En este anexo se explica que funciones tiene cada elemento de la interfaz web dirigida al usuario final.

En primer lugar, se muestra la página donde cada usuario hace login con su cuenta. Basta con escribir el usuario y contraseña en el apartado correspondiente y pulsar el botón de entrar.

Figura 113. Interfaz login donde el usuario entra en su cuenta privada

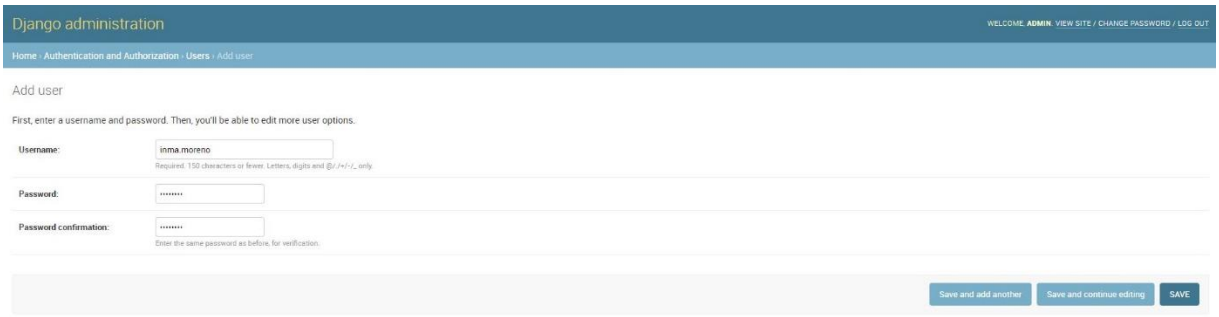


Figura 114. Panel de administración de Django para añadir usuarios nuevos

En el administrador de django, el inquilino puede configurar el usuario y contraseña de cada uno de los miembros de la familia. Para entrar a la plataforma de administración, basta con añadir /admin, a la dirección del navegador. Luego, el usuario administrador se loguea con sus crecenciales y en el apartado “User”, pinchar en “add” para añadir el usuario nuevo. En la figura 114, se muestra la ubicación final donde se añaden los datos.

En la figura 115 se muestra cada uno de los apartados del menú principal enumerados para una posterior explicación.

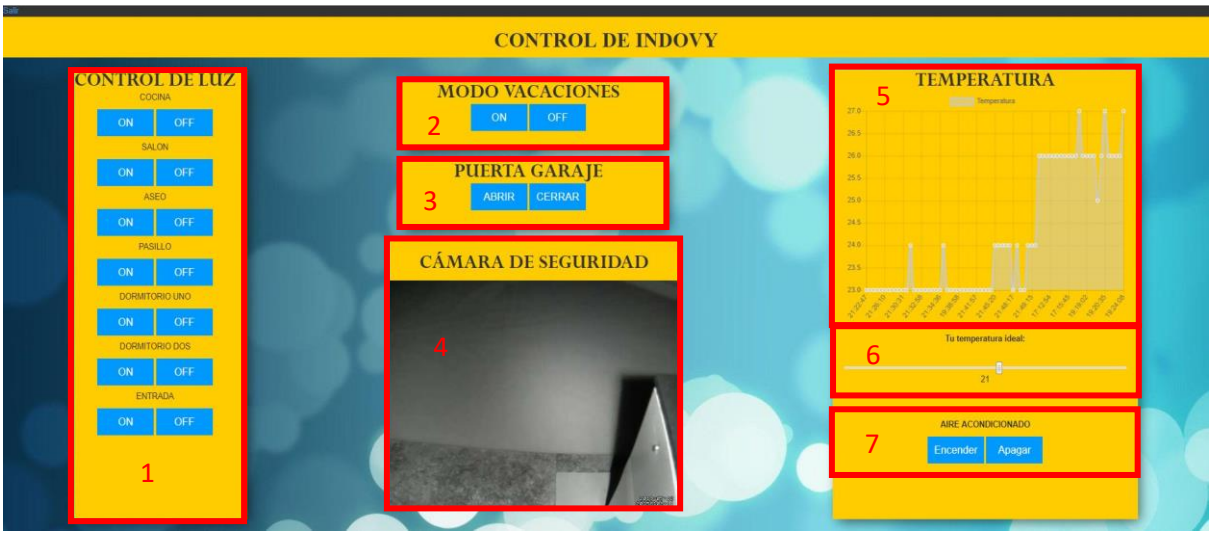


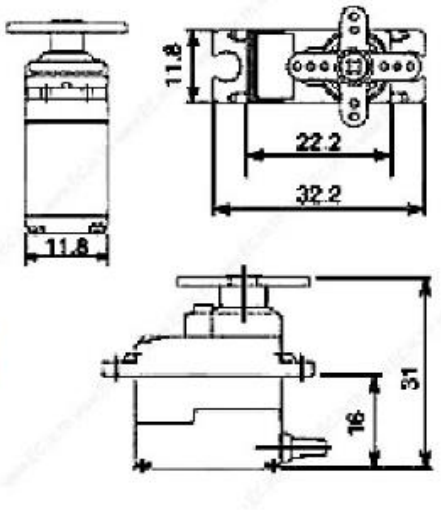
Figura 115. Panel de control de Indovy del usuario final numerado por secciones

1. **Control de luz:** en esta sección el usuario podrá apagar y encender las luces de las distintas habitaciones de la vivienda.
2. **Modo vacaciones:** cuando el usuario se marcha fuera por una larga temporada, tiene la opción de activar el modo vacaciones para que se simule, mediante apagado y encendido de luces, que hay actividad dentro de ella y, a la vez, se activa el sensor de presencia para detectar movimientos sospechosos, los cuales son registrados mediante mensajería a través de Telegram por medio de fotografías. En este modo la cámara de seguridad se desactivará para echar fotos.
3. **Puerta garaje:** el usuario puede abrir y cerrar la puerta del garaje de forma remota.
4. **Cámara de seguridad:** se puede observar en tiempo real lo que sucede en los alrededores.
5. **Temperatura:** gráfica que monitoriza en tiempo real la temperatura de la vivienda.
6. **Temperatura ideal:** el usuario puede configurar la temperatura a la que desea mantener la habitación, si esta sube por encima de ese valor, se activa el aire acondicionado automáticamente.
7. **Aire acondicionado:** el usuario puede apagar y encender el aire acondicionado a su antojo.

8.3. Datasheet

En este anexo se recogen todos los datasheet de los sensores utilizados para diseñar cada uno de los circuitos.

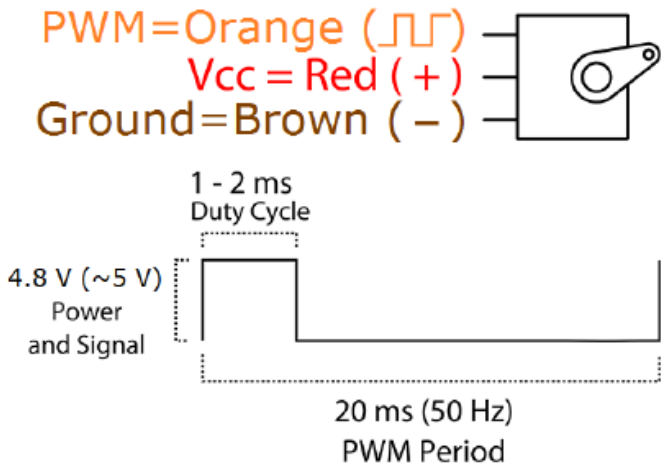
SG90 9 g Micro Servo



Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but *smaller*. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.

Specifications

- Weight: 9 g
- Dimension: 22.2 x 11.8 x 31 mm approx.
- Stall torque: 1.8 kgf·cm
- Operating speed: 0.1 s/60 degree
- Operating voltage: 4.8 V (~5V)
- Dead band width: 10 μs
- Temperature range: 0 ºC – 55 ºC

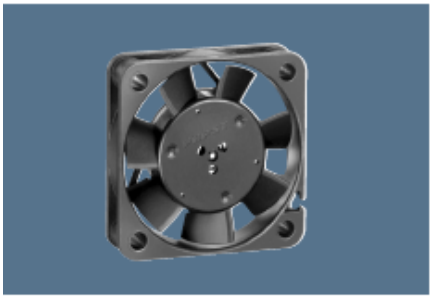


Position "0" (1.5 ms pulse) is middle, "90" (~2 ms pulse) is all the way to the right, "-90" (~1 ms pulse) is all the way to the left.

max. 9 m^3/h

DC axial fans

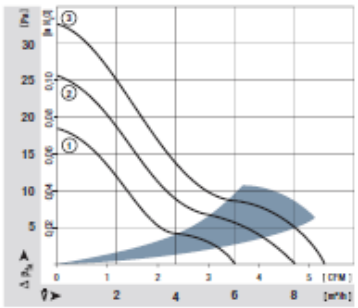
Series 400 F 40 x 40 x 10 mm

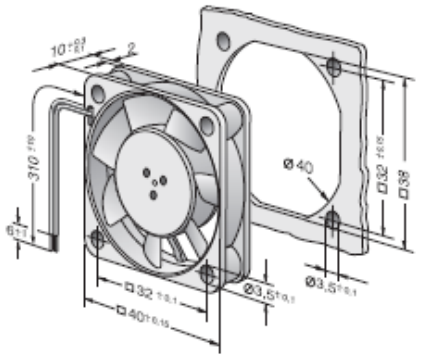


Highlights:
- Compact fan with low power consumption.
- Some models suitable for use at high ambient temperatures.

General characteristics:
- Material: fibreglass-reinforced plastic. Impeller PA, housing PBT.
- Fully integrated electronic commutation.
- Protected against reverse polarity and locking.
- Connection via single strands AWG 28, TR 64. Bared and tin-plated.
- Air exhaust over struts. Direction of rotation counter-clockwise, seen on rotor.
- Mass: 17 g.

Nominal data	Air flow	Air flow	Nominal voltage	Voltage range	Sound pressure level	Sound power level	Sintec sleeve bearings ▫ / Ball bearings ■	Power input	Nominal speed	Temperature range	Service life L_{10} (20 °C) ebm-papst Standard / Service life 60 °C ebm-papst Standard	Life expectancy L_{10} (40 °C) see P. 15	Curve	Specials
Type	**m^3/h**	**CFM**	**VDC**	**VDC**	**dB(A)**	**Bel(A)**	▫ / ■	**Watts**	**RPM**	**°C**	**Hours**	**Hours**	**P. 110**	
405 F	8	4,7	**5**	4,5...5,5	22,1	4,4	▫	0,7	5 400	-20...+70	45 000 / 15 000	47 500	2	/2
405 FH	9	5,3	**5**	4,5...5,5	26,0	4,6	▫	0,9	6 000	-20...+70	45 000 / 15 000	47 500	3	/2
412 FM	6	3,5	**12**	10...14	16,5	3,8	▫	0,6	4 300	-20...+70	45 000 / 15 000	47 500	1	
412 F	8	4,7	**12**	10...14	22,1	4,4	▫	0,7	5 400	-20...+70	45 000 / 15 000	47 500	2	
412 FH	9	5,3	**12**	10...14	26,0	4,6	▫	0,8	6 000	-20...+70	45 000 / 15 000	47 500	3	/2
414 F	8	4,7	**24**	20...28	22,1	4,4	▫	0,8	5 400	-20...+70	45 000 / 15 000	47 500	2	/2
414 FH	9	5,3	**24**	21,6...26,4	26,0	4,4	▫	0,9	6 000	-20...+70	45 000 / 15 000	47 500	3	
Models with temperature range up to +85 °C.														
412 FM-074	6	3,5	**12**	10...14	16,5	3,8	▫	0,4	4 300	-20...+85	45 000 / 15 000	47 500	1	/2
412 F-130	8	4,7	**12**	10...14	22,1	4,4	▫	0,6	5 400	-20...+85	45 000 / 15 000	47 500	2	
412 FH-132	9	5,3	**12**	10...14	26,0	4,6	▫	0,7	6 000	-20...+85	45 000 / 15 000	47 500	3	/2





ebmpapst

TECHNICAL DATA MQ-135 GAS SENSOR

FEATURES

Wide detecting scope Fast response and High sensitivity
Stable and long life Simple drive circuit

APPLICATION

They are used in air quality control equipments for buildings/offices, are suitable for detecting of NH3,NOx, alcohol, Benzene, smoke,CO_2 ,etc.

SPECIFICATIONS

A. Standard work condition

Symbol	Parameter name	Technical condition	Remarks
Vc	Circuit voltage	5V±0.1	AC OR DC
V_H	Heating voltage	5V±0.1	ACOR DC
R_L	Load resistance	can adjust	
R_H	Heater resistance	33Ω±5%	Room Tem
P_H	Heating consumption	less than 800mw	

B. Environment condition

Symbol	Parameter name	Technical condition	Remarks
Tao	Using Tem	-10□-45□	
Tas	Storage Tem	-20□-70□	
R_H	Related humidity	less than 95%Rh	
O_2	Oxygen concentration	21%(standard condition)Oxygen concentration can affect sensitivity	minimum value is over 2%

C. Sensitivity characteristic

Symbol	Parameter name	Technical parameter	Ramark 2
Rs	Sensing Resistance	30KΩ-200KΩ (100ppm NH_3)	Detecting concentration scopeⅡ 10ppm-300ppm NH_3 10ppm-1000ppm Benzene 10ppm-300ppm Alcohol
α (200/50) NH_3	Concentration Slope rate	≤0.65	
Standard Detecting Condition	Temp: 20□±2□ Vc:5V±0.1 Humidity: 65%±5% Vh: 5V±0.1		
Preheat time	Over 24 hour		

D. Structure and configuration, basic measuring circuit

	Parts	Materials
1	Gas sensing layer	SnO_2
2	Electrode	Au
3	Electrode line	Pt
4	Heater coil	Ni-Cr alloy
5	Tubular ceramic	Al_2O_3
6	Anti-explosion network	Stainless steel gauze (SUS316 100-mesh)
7	Clamp ring	Copper plating Ni
8	Resin base	Bakelite
9	Tube Pin	Copper plating Ni

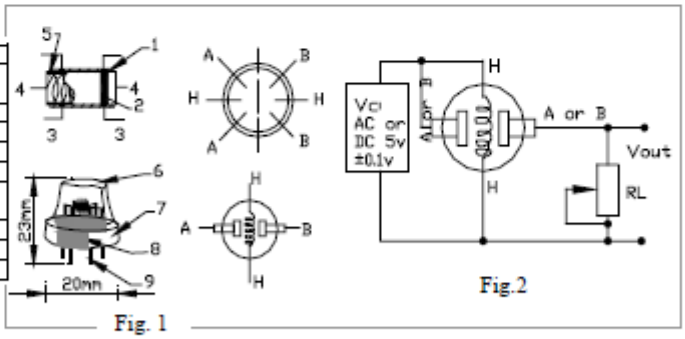
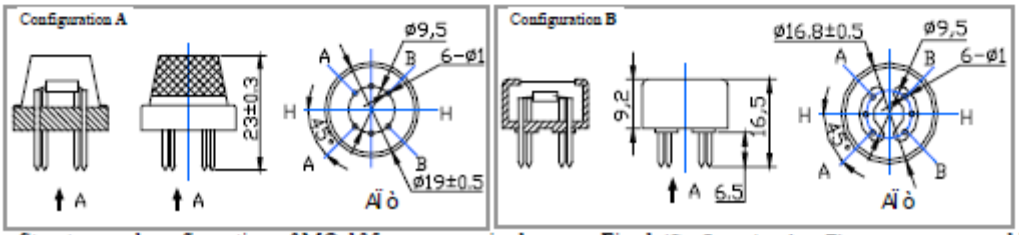


Fig. 1 Fig.2



Structure and configuration of MQ-135 gas sensor is shown as Fig. 1 (Configuration **A or B**), sensor composed by micro AL_2O_3 ceramic tube, Tin Dioxide (SnO_2) sensitive layer, measuring electrode and heater are fixed into a crust made by plastic and stainless steel net. The heater provides necessary work conditions for work of sensitive

components. The enveloped MQ-135 have 6 pin ,4 of them are used to fetch signals, and other 2 are used for providing heating current.

Electric parameter measurement circuit is shown as Fig.2

E. Sensitivity characteristic curve

Fig.2 sensitivity characteristics of the MQ-135

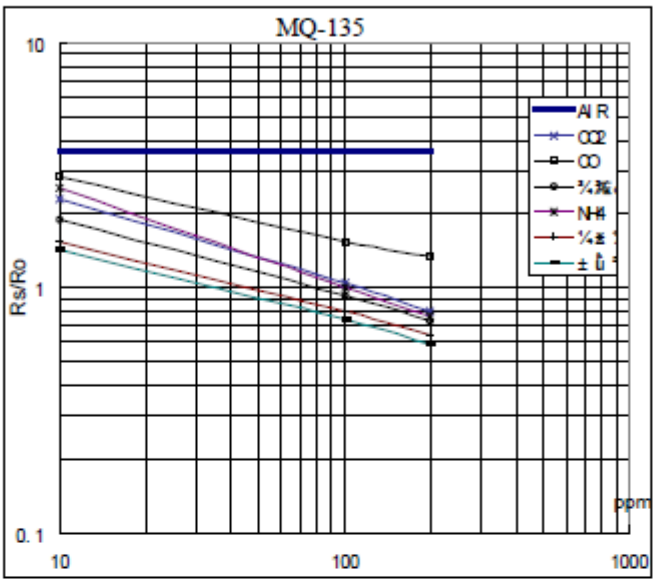


Fig.3 is shows the typical sensitivity characteristics of the MQ-135 for several gases.
in their: Temp: 20℃
Humidity: 65%
O_2 concentration 21%
RL=20kΩ
Ro: sensor resistance at 100ppm of NH_3 in the clean air.
Rs:sensor resistance at various concentrations of gases.

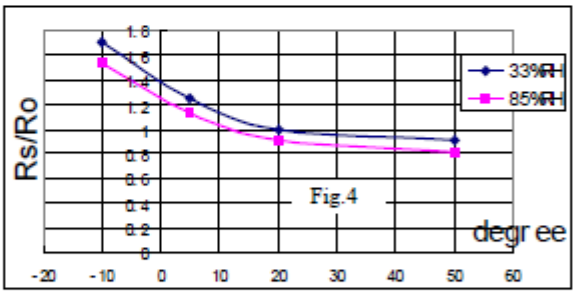
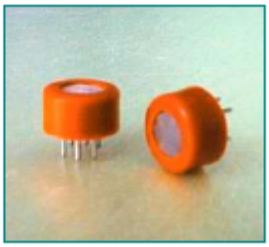


Fig.4 is shows the typical dependence of the MQ-135 on temperature and humidity.
Ro: sensor resistance at 100ppm of NH_3 in air at 33%RH and 20 degree.
Rs: sensor resistance at 100ppm of NH_3 at different temperatures and humidities.

SENSITVITY ADJUSTMENT

Resistance value of MQ-135 is difference to various kinds and various concentration gases. So,When using this components, sensitivity adjustment is very necessary. we recommend that you calibrate the detector for 100ppm NH_3 or 50ppm Alcohol concentration in air and use value of Load resistancethat(R_L) about 20 KΩ(10KΩ to 47 KΩ).

When accurately measuring, the proper alarm point for the gas detector should be determined after considering the temperature and humidity influence.



Specification

1 Electrical parameters

Product Type	HC--SR501 Body Sensor Module
Operating voltage range	DC 4.5-20V
Quiescent Current	<50uA
Level output	High 3.3 V /Low 0V
Trigger	L can not be repeated trigger/H can be repeated trigger(Default repeated trigger)
Delay time	5-200S(adjustable) the range is (0.xx second to tens of second)
Block time	2.5S(default)Can be made a range(0.xx to tens of seconds
Board Dimensions	32mm*24mm
Angle Sensor	<100 ° cone angle
Operation Temp.	-15-+70 degrees
Lens size sensor	Diameter:23mm(Default)

2 Features:

1, the automatic sensor: to enter the sensor output range is high, people leave the sensor range of the automatic delay off high, output low.
2, the photosensitive control (optional, factory is not set) may set the photosensitive control during the day or light intensity without induction.

3, the temperature compensation (optional, factory is not set): In the summer when the ambient temperature rises to 30 ~ 32 ℃, slightly shorter detection range, temperature compensation can be used as a performance compensation.

4, two trigger mode: (can be selected by jumpers)

a. can not repeat the trigger: the sensor output high, the delay time is over, the output will automatically become low from high;

b. repeatable trigger: the sensor output high after the delay period, if the human body in its sensing range

Activities, its output will remain high until after the delay will be left high to low (sensor module review

Measured activities of each body will be automatically extended after a delay time, and the final event of the delay time

Starting point of time).

5, with induction blocking time (the default setting: 2.5S block time): sensor module, after each sensor output (high change

Into a low level), you can set up a blockade followed by time period, in this time period the sensor does not accept any sensor signal.

This feature can have a "sensor output time" and "blocking time" the interval between the work produced can be applied to detect the interval Products; also inhibit this function during load switching for a variety of interference. (This time can be set at zero seconds

- Tens of seconds).

6, the working voltage range: the default voltage DC4.5V-20V.

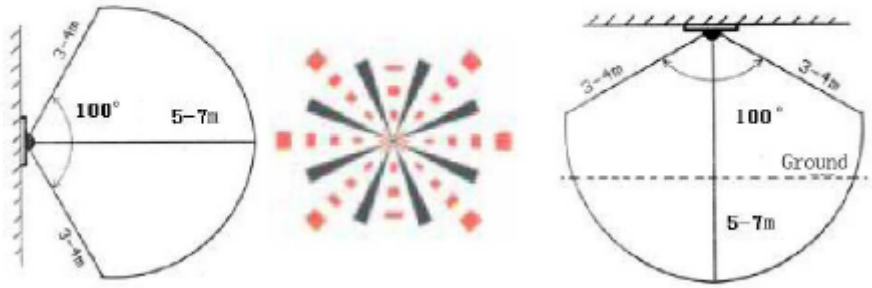
7, micro-power consumption: static current "50 microamps, especially for battery-powered automatic control products.

8, the output high level signals: types of circuits can be easily and docking.

3 Instructions:

1. Sensing module for about a minute after power initialization time, during the interval to the output module
0-3 times a minute in standby mode.
2. Should avoid direct lighting such as interference sources close the surface of the lens module so as to avoid the introduction of interference signal generator malfunction; use of the environment to avoid the flow of the wind, the wind sensor will also cause interference.
3. Sensor module using a dual probe, the probe's window is rectangular, dual (A per B million) in the direction of the ends of long, when the body passed from left to right or right to left when the reach the dual IR time, distance difference, the greater the difference, more sensitive sensors, when the body from the front to the probe or from top to bottom or from bottom to top direction passing, dual IR not detected changes in the distance, no difference value, the sensor insensitive or does not work; so the sensors should be installed dual direction of the probe with human activities as much as possible parallel to the direction of maximum to ensure that the body has been passed by the dual sensor probe. To increase the sensing range of angles, the module using a circular lens, the probe also makes sense on all four sides, but still higher than the upper and lower left and right direction of sensing range, sensitivity and strong, still as far as possible by the above installation requirements. VCC, trig (control side), echo (receiving end), GND

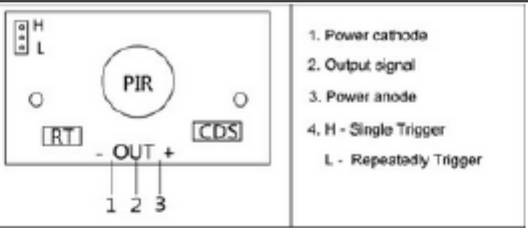
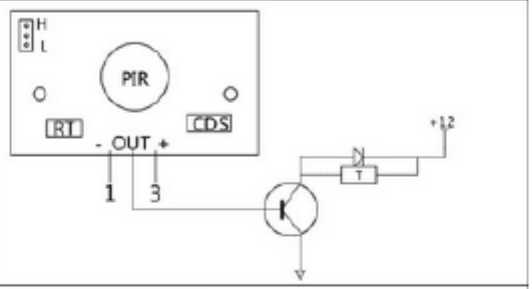
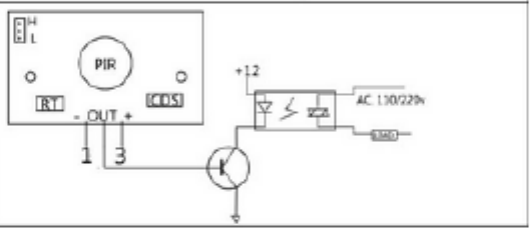
Induction Range:



Dimensions and Adjustment:

Note: The potentiometer clockwise to adjust the distance, sensing range increases (about 7 meters), on the contrary, sensing range decreases (about 3 meters).

Delay adjustment potentiometer clockwise rotation, sensor delay longer (about 300S), the other hand, induction by the short delay (about 5S).

Pin Define	
DC Load	
AC Load	

Applications:

1, Security Products

2, the human body sensors toys

3, the human body sensor lighting

4, industrial automation and control, etc.

It can automatically and quickly open various types of incandescent, fluorescent lamps, buzzer, automatic doors, electric fans, automatic washing machine and dryer

Machines and other devices, is a high-tech products. Especially suitable for enterprises, hotels, shopping malls, warehouses and family aisles, corridors and other sensitive.

Ultrasonic Ranging Module HC - SR04

Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
(3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.
Test distance = (high level time×velocity of sound (340M/S) / 2,

Wire connecting direct as following:

- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

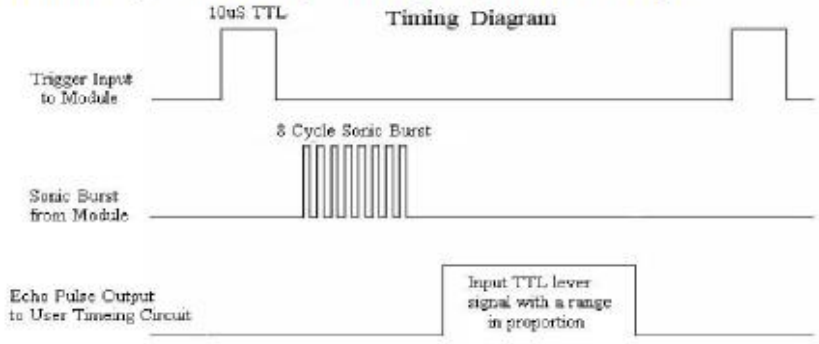
Electric Parameter

Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
MeasuringAngle	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm

Vcc Trig Echo GND

Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.



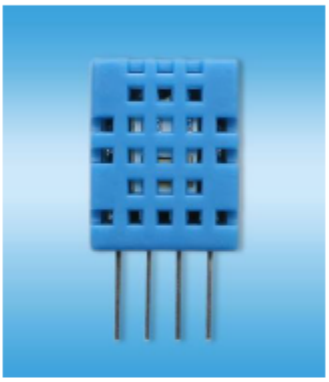
Attention:

- The module is not suggested to connect directly to electric, if connected electric, the GND terminal should be connected the module first, otherwise, it will affect the normal work of the module.
- When tested objects, the range of area is not less than 0.5 square meters and the plane requests as smooth as possible, otherwise ,it will affect the results of measuring.

www.Elecfreaks.com

1、Product Overview

DHT11 digital temperature and humidity sensor is a composite Sensor contains a calibrated digital signal output of the temperature and humidity. Application of a dedicated digital modules collection technology and the temperature and humidity sensing technology, to ensure that the product has high reliability and excellent long-term stability. The sensor includes a resistive sense of wet components and an NTC temperature measurement devices, and connected with a high-performance 8-bit microcontroller.



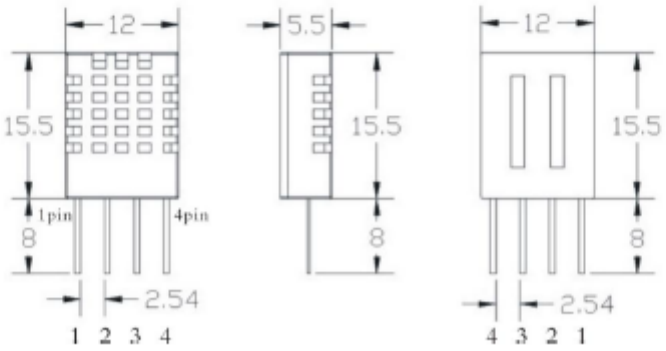
2、Applications

HVAC, dehumidifier, testing and inspection equipment, consumer goods, automotive, automatic control, data loggers, weather stations, home appliances, humidity regulator, medical and other humidity measurement and control.

3、Features

Low cost, long-term stability, relative humidity and temperature measurement, excellent quality, fast response, strong anti-interference ability, long distance signal transmission, digital signal output, and precise calibration.

4、Dimensions (unit: mm)



5、Product parameters

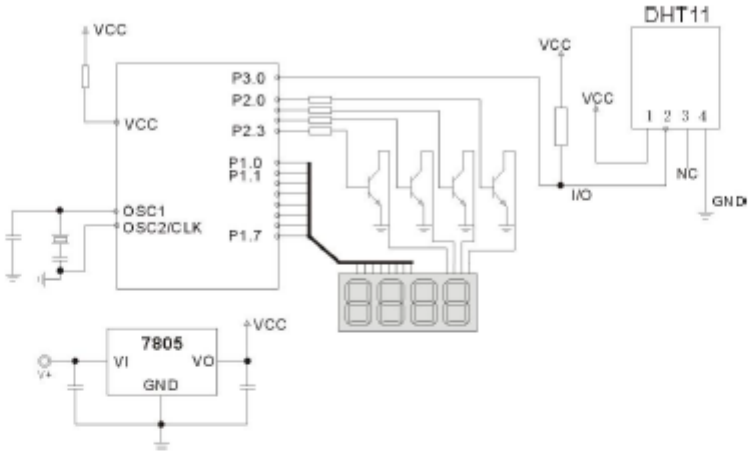
Relative humidity
Resolution: 16Bit
Repeatability: ±1% RH
Accuracy: At 25℃ ±5% RH
Interchangeability: fully interchangeable
Response time: 1 / e (63%) of 25℃ 6s
1m / s air 6s
Hysteresis: <± 0.3% RH
Long-term stability: <± 0.5% RH / yr in

Temperature
Resolution: 16Bit
Repeatability: ±0.2℃
Range: At 25℃ ±2℃
Response time: 1 / e (63%) 10S

Electrical Characteristics
Power supply: DC 3.5 ~ 5.5V
Supply Current: measurement 0.3mA standby 60μ A
Sampling period: more than 2 seconds

Pin Description
1, the VDD power supply 3.5 ~ 5.5V DC
2 DATA serial data, a single bus
3, NC, empty pin
4, GND ground, the negative power

6、Typical circuit



Microprocessor and DHT11 of connection typical application circuit as shown above, DATA pull the microprocessor I / O ports are connected.

1. Typical application circuit recommended in the short cable length of 20 meters on the 5.1K pull-up resistor, the resistance of greater than 20 meters under the pull-up resistor on the lower of the actual situation.

2. When using a 3.5V voltage supply cable length shall not be greater than 20cm. Otherwise, the line voltage drop will cause the sensor power supply shortage, caused by measurement error.

3. Each read out the temperature and humidity values are the results of the last measurement For real-time data, sequential read twice, but is not recommended to repeatedly read the sensors, each read sensor interval is greater than 5 seconds can be obtainedaccurate data.

7、Serial communication instructions (single-wire bi-directional)

◎Single bus Description

DHT11 uses a simplified single-bus communication. Single bus that only one data line, the system of data exchange, control by a single bus to complete. Device (master or slave) through an open-drain or tri-state port connected to the data line to allow the device does not send data to release the bus, while other devices use the bus; single bus usually require an external one about 5.1kΩ pull-up resistor, so that when the bus is idle, its status is high. Because they are the master-slave structure, and only when the host calls the slave, the slave can answer, the host access devices must strictly follow the single-bus sequence, if the chaotic sequence, the device will not respond to the host.

◎Single bus to transfer data defined

DATA For communication and synchronization between the microprocessor and DHT11, single-bus data format, a transmission of 40 data, the high first-out.

Data format:

The 8bit humidity integer data + 8bit the Humidity decimal data +8 bit temperature integer data + 8bit fractional temperature data +8 bit parity bit.

◎Parity bit data definition

"8bit humidity integer data + 8bit humidity decimal data +8 bit temperature integer data + 8bit temperature fractional data" 8bit checksum is equal to the results of the last eight.

Example 1: 40 data is received:

0011 0101	0000 0000	0001 1000	0000 0000	0100 1101
High humidity 8	Low humidity 8	High temp. 8	Low temp. 8	Parity bit

Calculate:

0011 0101+0000 0000+0001 1000+0000 0000= 0100 1101

Received data is correct:

Humidity: 0011 0101=35H=53%RH

Temperature: 0001 1000=18H=24℃

Example 2: 40 data is received:

0011 0101	0000 0000	0001 1000	0000 0000	0100 1001
High humidity 8	Low humidity 8	High temp. 8	Low temp. 8	Parity bit

Calculate:

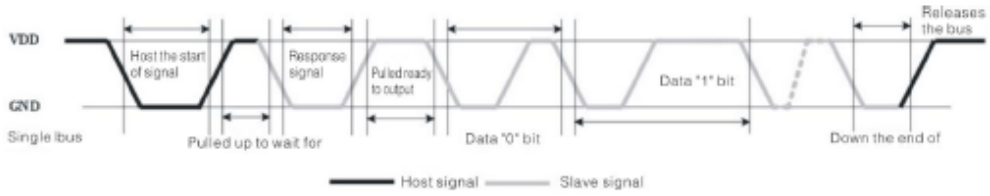
0011 0101+0000 0000+0001 1000+0000 0000= 0100 1101

01001101≠0100 1001

The received data is not correct, give up, to re-receive data.

◎Data Timing Diagram

User host (MCU) to send a signal, DHT11 converted from low-power mode to high-speed mode, until the host began to signal the end of the DHT11 send a response signal to send 40bit data, and trigger a letter collection. The signal is sent as shown.



Data Timing Diagram

Note: The host reads the temperature and humidity data from DHT11 always the last measured value, such as twice the measured interval of time is very long, continuous read twice to the second value of real-time temperature and humidity values.

◎Peripherals read steps

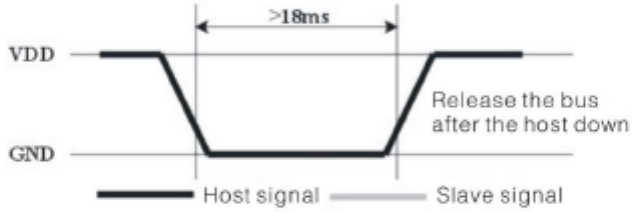
Communication between the master and slave can be done through the following steps (peripherals (such as microprocessors) read DHT11 the data of steps).

Step 1:

After power on DHT11 (DHT11 on after power to wait 1S across the unstable state during this period can not send any instruction), the test environment temperature and humidity data, and record the data, while DHT11 the DATA data lines pulled by pull-up resistor has been to maintainhigh; the DHT11 the DATA pin is in input state, the moment of detection of external signals.

Step 2:

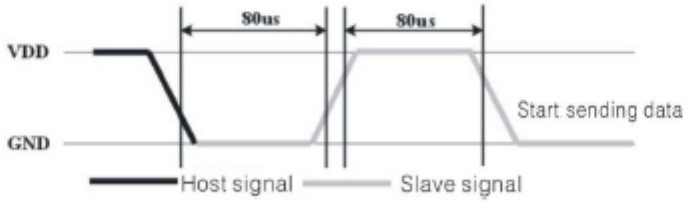
Microprocessor I / O set to output at the same time output low, and low hold time can not be less than 18ms, then the microprocessor I / O is set to input state, due to the pull-up resistor, a microprocessor/ O DHT11 the dATA data lines also will be high, waiting DHT11 to answer signal, send the signal as shown:



Host sends a start signal

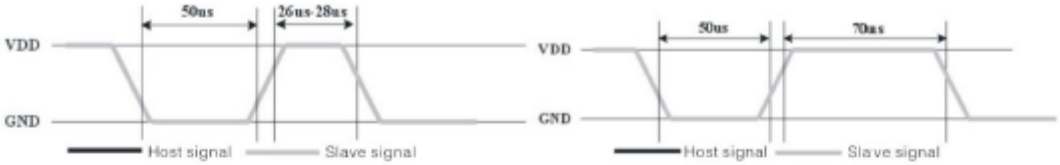
Step 3:

DATA pin is detected to an external signal of DHT11 low, waiting for external signal low end the delay DHT11 DATA pin in the output state, the output low of 80 microseconds as the response signal, followed by the output of 80 micro-seconds of high notification peripheral is ready to receive data, the microprocessor I / O at this time in the input state is detected the I / O low (DHT11 response signal), wait 80 microseconds highdata receiving and sending signals as shown:



Step 4:

Output by DHT11 the DATA pin 40, the microprocessor receives 40 data bits of data "0" format: the low level of 50 microseconds and 26-28 microseconds according to the changes in the I / O levellevel, bit data "1" format: the high level of low plus, 50 microseconds to 70 microseconds. Bit data "0", "1" signal format as shown:



Bit data "0" bit format　　Bit data "1" bit format

End signal:

Continue to output the low 50 microseconds after DHT11 the DATA pin output 40 data, and changed the input state, along with pull-up resistor goes high. But DHT11 internal re-test environmental temperature and humidity data, and record the data, waiting for the arrival of the external signal.

8、Application of information

1. Work and storage conditions

Outside the sensor the proposed scope of work may lead to temporary drift of the signal up to 300%RH. Return to normal working conditions, sensor calibration status will slowly toward recovery. To speed up the recovery process may refer to "resume processing". Prolonged use of non-normal operating conditions, will accelerate the aging of the product.

Avoid placing the components on the long-term condensation and dry environment, as well as the following environment.

A, salt spray

B, acidic or oxidizing gases such as sulfur dioxide, hydrochloric acid

Recommended storage environment

Temperature: 10 ~ 40 ℃ Humidity: 60% RH or less

2. The impact of exposure to chemicals

The capacitive humidity sensor has a layer by chemical vapor interference, the proliferation of chemicals in the sensing layer may lead to drift and decreased sensitivity of the measured values. In a pure environment, contaminants will slowly be released. Resume processing as described below will accelerate this process. The high concentration of chemical pollution (such as ethanol) will lead to the complete damage of the sensitive layer of the sensor.

3. The temperature influence

Relative humidity of the gas to a large extent dependent on temperature. Therefore, in the measurement of humidity, should be to ensure that the work of the humidity sensor at the same temperature. With the release of heat of electronic components share a printed circuit board, the installation should be as far as possible the sensor away from the electronic components and mounted below the heat source, while maintaining good ventilation of the enclosure. To reduce the thermal conductivity sensor and printed circuit board copper plating should be the smallest possible, and leaving a gap between the two.

4. Light impact

Prolonged exposure to sunlight or strong ultraviolet radiation, and degrade performance.

5. Resume processing

Placed under extreme working conditions or chemical vapor sensor, which allows it to return to the status of calibration by the following handler. Maintain two hours in the humidity conditions of 45℃ and <10% RH (dry); followed by 20-30℃ and> 70% RH humidity conditions to maintain more than five hours.

6. Wiring precautions

The quality of the signal wire will affect the quality of the voltage output, it is recommended to use high quality shielded cable.

7. Welding information

Manual welding, in the maximum temperature of 300℃ under the conditions of contact time shall be less than 3 seconds.

8. Product upgrades

Details, please the consultation Aosong electronics department.

9、The license agreement

Without the prior written permission of the copyright holder, shall not in any form or by any means, electronic or mechanical (including photocopying), copy any part of this manual, nor shall its contents be communicated to a third party. The contents are subject to change without notice.

The Company and third parties have ownership of the software, the user may use only signed a contract or software license.

10、Warnings and personal injury

This product is not applied to the safety or emergency stop devices, as well as the failure of the product may result in injury to any other application, unless a particular purpose or use authorized. Installation, handling, use or maintenance of the product refer to product data sheets and application notes. Failure to comply with this recommendation may result in death and serious personal injury. The Company will bear all damages resulting personal injury or death, and waive any claims that the resulting subsidiary company managers and employees and agents, distributors, etc. that may arise, including: a variety of costs, compensation costs, attorneys' fees, and so on.

11、Quality Assurance

The company and its direct purchaser of the product quality guarantee period of three months (from the date of delivery). Publishes the technical specifications of the product data sheet shall prevail. Within the warranty period, the product was confirmed that the quality is really defective, the company will provide free repair or replacement. The user must satisfy the following conditions:

① The product is found defective within 14 days written notice to the Company;

② The product shall be paid by mail back to the company;

③ The product should be within the warranty period.

The Company is only responsible for those used in the occasion of the technical condition of the product defective product. Without any guarantee, warranty or written statement of its products used in special applications. Company for its products applied to the reliability of the product or circuit does not make any commitment.

Datasheet	5mm RGB LED common cathode http://www.kitronik.co.uk

Kitronik Ltd – 5mm RGB LED Common Cathode

TECHNOLOGY DATA SHEET & SPECIFICATIONS

Device Selection Guide

Chip		Lens Colour
Material	Emitted Color	
AlGaInP	Red	Water clear
InGaN	Green	
InGa1N	Blue	

Features

- Uniform light output.
- Low power consumption.
- I.C. compatible.
- Long life solderability.
- Common Cathode.

Descriptions

- The Red source colour devices are made with AlGaInP on GaAs substrate.
- The Green source colour devices are made with InGaN on sic.
- The Blue source colour devices are made with InGaA1N on sic.

Usage Notes:

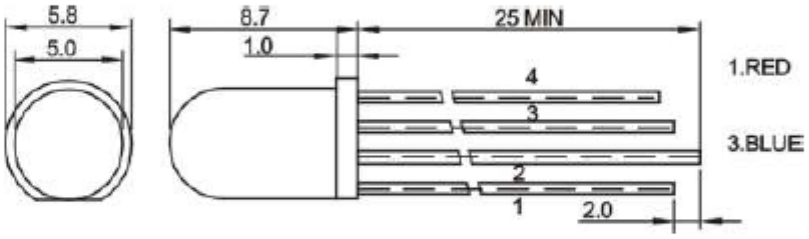
- The ultra bright LED is an electrostatic sensitive device, so static electricity and surge will damage the LED.
- It is required to wear a wrist-band when handling the LED. All device, equipment, machinery, desk and ground must be properly grounded.
- When using LED, it must use a protective resistor in series with DC current about 20mA.

Applications

- Status indicators.
- Commercial use.
- Advertising signs.
- Back lighting.

Datasheet	5mm RGB LED common cathode http://www.kitronik.co.uk

Package Dimensions



UNIT:mm

1. RED
2. COMMON CATHODE
3. BLUE
4. GREEN

Notes:

- Other dimensions are in millimetres, tolerance is 0.25mm except being specified.
- Protruded resin under flange is 1.5mm Max LED.
- Bare copper alloy is exposed at tie-bar portion after cutting.

Datasheet	5mm RGB LED common cathode http://www.kitronik.co.uk

Absolute Maximum Rating (Ta=25°C)

Parameter	Symbol	Absolute Maximum Rating	Unit
Forward Pulse Current	IFPM	70	mA
Forward Current	IFM	30	mA
Reverse Voltage	VR	5	V
Power Dissipation	PD	140	mW
Operating Temperature	Topr	-40~+80	°C
Storage Temperature	Tstg	-40~+100	°C
Soldering Temperature	Tsol	Reflow Soldering : 260 °C for 10 sec. Hand Soldering : 350 °C for 3 sec.	°C

Electro-Optical Characteristics (Ta=25°C)

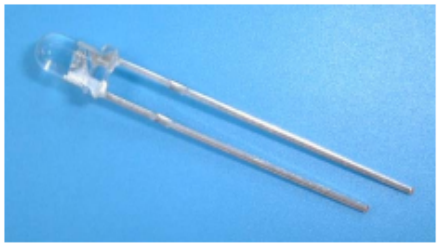
Parameter	Symbol	Device	Min.	Typ.	Max.	Unit	Test Condition
Luminous Intensity	Iv	Red Green Blue	1000 1200 1000	1500 2000 1500	2300 2700 2200	mcd	IF=20mA
Viewing Angle	2θ1/2	Red Green Blue	40	---	50	Deg	
Peak Emission Wavelength	λp	Red Green Blue	635 520 460	640 525 465	650 530 470	nm	IF=20mA
Spectral Line Half-Width	Δλ	Red Green Blue	15 15 25	20 20 30	25 25 35	nm	IF=20mA
Forward Voltage	VF	Red Green Blue	1.9 2.9 2.9	--- --- ---	2.5 3.5 3.5	V	IF=20mA
Reverse Current	IR	Red Green Blue	---	---	10	μA	VR=5V

EVERLIGHT

Technical Data Sheet
3 mm Round LED (T-1)

204-15UTC/S400-X9

Features



- Popular T-1 colorless 3mm package.
- High luminous power.
- Typical chromaticity coordinates x=0.29, y=0.28 according to CIE1931.
- Bulk, available taped on reel.
- Pb free.
- ESD-withstand voltage: up to 4KV
- The product itself will remain within RoHS compliant version.

Descriptions

- The series is designed for application required high luminous intensity.
- The phosphor filled in the reflector converts the blue emission of InGaN chip to ideal white.

Applications

- Outdoor Displays
- Optical Indicators
- Backlighting
- Marker Lights

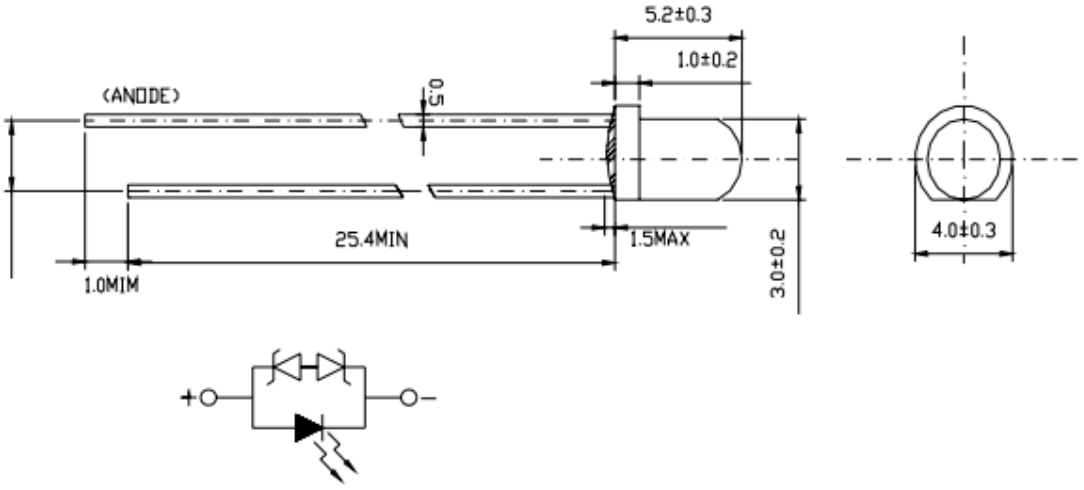
Device Selection Guide

PART NO.	Chip		Lens Color
	Material	Emitted Color	
204-15UTC/S400-X9	InGaN	White	Water Clear

Technical Data Sheet
3 mm Round LED (T-1)

204-15UTC/S400-X9

Package Dimensions



Notes:

1.All dimensions are in millimeters, and tolerance is 0.25mm except being specified.

2.Lead spacing is measured where the lead emerges from the package.

3.Protruded resin under flange is 1.5mm Max. LED.

Absolute Maximum Ratings (Ta=25℃)

Parameter	Symbol	Rating	Unit
Continuous Forward Current	I_F	25	mA
Peak Forward Current(Duty /10 @ 1KHZ)	I_{FP}	100	mA
Reverse Voltage	V_R	5	V
Operating Temperature	T_{opr}	-40 ~ +85	℃
Storage Temperature	T_{stg}	-40 ~ +100	℃
Soldering Temperature (T=5 sec)	T_{sol}	260 ± 5	℃
Power Dissipation	P_d	100	mW
Zener Reverse Current	Iz	100	mA
Electrostatic Discharge	ESD	4K	V

Technical Data Sheet
3 mm Round LED (T-1)

204-15UTC/S400-X9

Electro-Optical Characteristics (Ta=25℃)

Parameter	Symbol	Condition	Min.	Typ.	Max.	Units
Forward Voltage	V_F	I_F=20mA	--	3.5	4.0	V
Zener Reverse Voltage	Vz	Iz=5mA	5.8	----	----	V
Reverse Current	I_R	V_R=5V	--	--	50	uA
Luminous Intensity	I_V	I_F=20mA	2850	----	7150	mcd
Viewing Angle	2 θ 1/2	I_F=20mA	--	25°	--	deg
Chromaticity Coordinates	x	I_F=20mA	--	0.29	--	
	y	------	--	0.28	--	

Luminous Intensity Combination (mcd at 20mA)

I_V Ranks	Y	Z0
Min.	2850	4500
Max.	4500	7150

*Measurement Uncertainty of Luminous Intensity: ±15%

Forward Voltage Combination (V at 20mA)

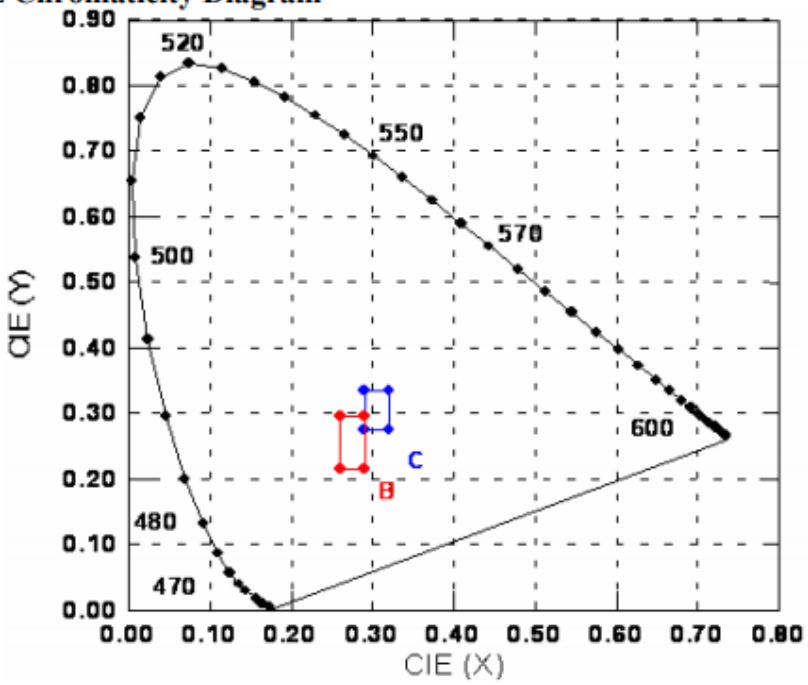
Group	Rank	
Min.	3.00	3.50
Max.	3.50	4.00

*Measurement Uncertainty of Forward Voltage：±0.1V

Technical Data Sheet
3 mm Round LED (T-1)

204-15UTC/S400-X9

CIE Chromaticity Diagram



Color Ranks (IF=20mA，Ta=25℃)

色度等級 Color grading	CIE X		CIE Y		Test condition
	Min.	Max.	Min.	Max.	
B1	0.260	0.275	0.215	0.275	IF=20mA
B2	0.275	0.290	0.245	0.295	
C1	0.290	0.305	0.275	0.315	
C2	0.305	0.320	0.295	0.335	

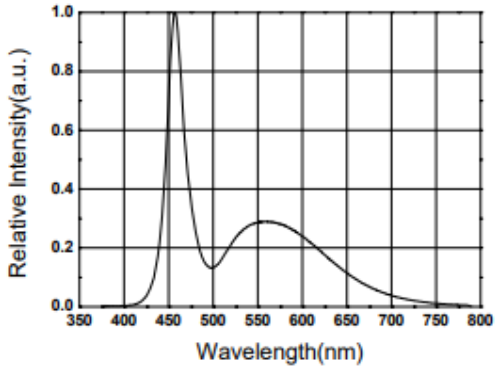
*Measurement uncertainty of the color coordinates：±0.01

Technical Data Sheet
3 mm Round LED (T-1)

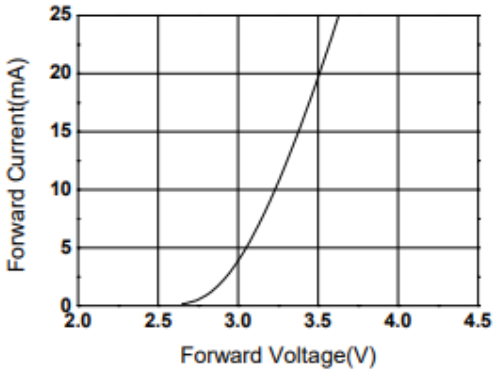
204-15UTC/S400-X9

Typical Electro-Optical Characteristics Curves

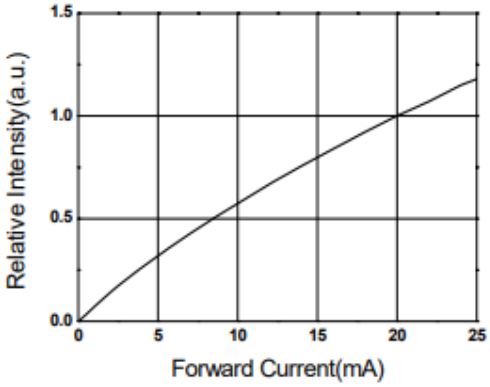
Relative Intensity vs. Wavelength



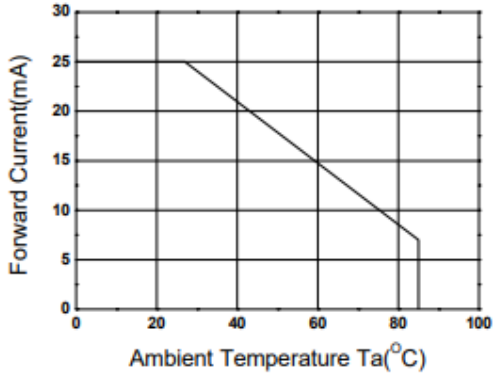
Forward Current vs. Forward Voltage



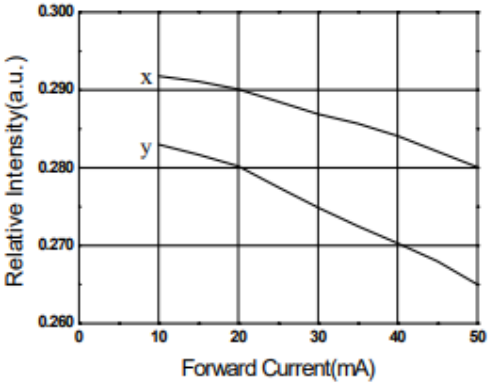
Relative Intensity vs. Forward Current



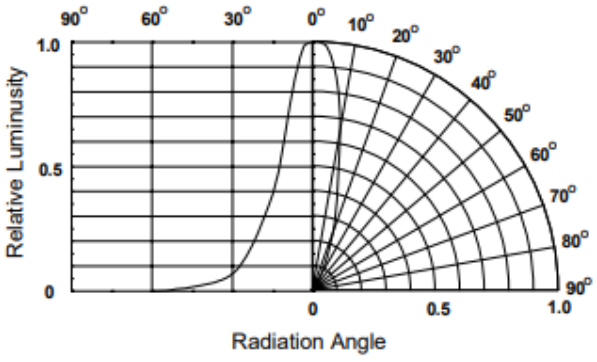
Forward Current vs. Ambient Temp.



Chromaticity Coordinate vs. Forward Current



Relative Intensity vs. Angle Displacement



9. REFERENCIAS BIBLIOGRÁFICAS

[1] Blog consulta. [En línea] [Citado el: 15 Junio 2018] Disponible en: http://www.tynmagazine.com/las-tres-fases-de-la-evolucion-de-internet-de-las-cosas-2/

[2] Blog consulta. [En línea] [Citado el: 15 Junio 2018] Disponible en: http://vintagecomunicacion.com/la-evolucion-del-internet-de-las-cosas/

[3] Blog consulta. [En línea] [Citado el: 15 Junio 2018] Disponible en: https://es.wikipedia.org/wiki/Internet_de_las_cosas#Control_de_objetos

[4] Blog consulta. [En línea] [Citado el: 15 Junio 2018] Disponible en: https://www.domodesk.com/221-a-fondo-que-es-iot-el-internet-de-las-cosas.html

[5] Blog consulta. [En línea] [Citado el: 15 Junio 2018] Disponible en: https://www.ucema.edu.ar/revista-ucema/nro28/la-internet-de-las-cosas

[6] Blog consulta. [En línea] [Citado el: 15 Junio 2018] Disponible en: https://ovacen.com/internet-de-las-cosas/

[7] Blog consulta. [En línea] [Citado el: 15 Junio 2018] Disponible en: https://comohacer.eu/comparativa-y-analisis-raspberry-pi-vs-competencia/

La voy a [8] Upton, Eben. Raspberry Pi. Guía de usuario. Edición 2016. ISBN-13: 978-8441538719

[9] Datasheet. [En línea] [Citado el: 15 Junio 2018] Disponible en: https://electronilab.co/wp-content/uploads/2013/12/HC-SR501.pdf

[10] Blog consulta. [En línea] [Citado el: 15 Junio 2018] Disponible en: http://a-electronics.com.mx/index.php?controller=attachment&id_attachment=486

[11] Datasheet. [En línea] [Citado el: 15 Junio 2018] Disponible en: https://www.mouser.com/ds/2/813/HCSR04-1022824.pdf

[12] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: http://bkargado.blogspot.com/2013/09/todosobrehc-sr04.html

[13] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: http://pitando.australiando.es/2016/02/18/controla-un-sensor-de-proximidad-desde-tu-raspberry-pi/

[14] Datasheet. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: http://datasheetcafe.databank.netdna-cdn.com/wp-content/uploads/2015/12/SG90.pdf

[15] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: https://components101.com/servo-motor-basics-pinout-datasheet

[16] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: https://www.olimex.com/Products/Components/Sensors/SNS-MQ135/resources/SNS-MQ135.pdf

[17] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: https://naylampmechatronics.com/blog/42_Tutorial-sensores-de-gas-MQ2-MQ3-MQ7-y-MQ13.html

[18] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: https://www.luisllamas.es/reproducir-sonidos-arduino-buzzer-pasivo-altavoz/

[19] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: http://fpaez.com/sensor-dht11-de-temperatura-y-humedad/

[20] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: https://components101.com/dht11-temperature-sensor

[21] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en: https://akizukidenshi.com/download/ds/aosong/DHT11.pdf

[22] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en:

https://www.zonamaker.com/electronica/intro-electronica/componentes/el-diodo-led

[23] Blog consulta. [En línea] [Fecha de consulta: 23 junio 2018] Disponible en:

http://www.asifunciona.com/fisica/af_leds/af_leds_3.htm

[24] Quijao, J. L. (2010). Domine PHP y MySQL. Alfaomega.

[25] Blog consulta. [En línea] [Fecha de consulta: 24 junio 2018] Disponible en: https://es.wikipedia.org/wiki/Protocolo_de_transferencia_de_hipertexto

[26] Blog consulta. [En línea] [Fecha de consulta: 24 junio 2018] Disponible en: https://www.ecured.cu/Framework

[27] Blog consulta. [En línea] [Fecha de consulta: 24 junio 2018] Disponible en: https://es.wikipedia.org/wiki/Django_(framework)

[28] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: http://mundogeek.net/archivos/2008/01/10/%C2%BFque-es-python/

[29] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: https://www.bejob.com/7-razones-para-programar-en-python/

[30] Blog consulta. [En línea] [Fecha de consulta: 24 junio 2018] Disponible en: https://definicion.de/html/

[31] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: https://definicion.de/css/

[32] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: http://librosweb.es/libro/css/capitulo_1/breve_historia_de_css.html

[33] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: https://es.wikipedia.org/wiki/Hoja_de_estilos_en_cascada

[34] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: https://getbootstrap.com/docs/3.4/css/

[35] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: https://core.telegram.org/bots/api

[36] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: https://wit.ai/

[37] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: https://cucopc.es/2018/03/08/rasbperry-pi-tutorial-reconocimiento-voz-gratis-wit-2018/

[38] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: https://telegram.org/blog/bot-revolution

[39] Blog consulta. [En línea] [Fecha de consulta: 26 junio 2018] Disponible en: https://pimylifeup.com/raspberry-pi-port-forwarding/