UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Sistema de automatización y control de una vivienda y puerta de garaje

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Amores Dolz, Pedro

Tutor/a: Hervás Jorge, Antonio

Cotutor/a: Capilla Lladró, Roberto

CURSO ACADÉMICO: 2021/2022

Agradecimientos

A mi hermano por haber sido el propulsor de este proyecto conjunto, así como el encargado de dirigir la parte común de ambos trabajos. Por supuesto a mis padres por su ayuda y motivación que me ha permitido llegar hasta aquí. Por último, también agradecer a los tutores del trabajo por su tiempo y dedicación.

Resumen

En este trabajo de final de grado se va a implementar con Home Assistant un sistema integral de automatización y control de una vivienda y una puerta de garaje, permitiendo el control desde un smartphone o pc de la iluminación, climatización, sistema de seguridad, video vigilancia y apertura de puerta de garaje mediante lectura de matrícula. Todo ello se llevará a cabo utilizando elementos de bajo coste y fácilmente incorporables en instalaciones ya existentes.

Además del control, también se programarán diferentes automatizaciones de los elementos del sistema, lo que permitirá minimizar la interacción por parte del usuario, anticipándose el sistema a las necesidades habituales y consiguiendo a su vez una mejora notable en el consumo de energía eléctrica de la vivienda.

Para dotar al usuario de una mayor comodidad e independencia, se integrará todo el control del sistema con Google Assistant, pudiendo llevar a cabo el control total de la vivienda a través de órdenes de voz, sin necesidad de hacer un uso intensivo del smartphone.

Este trabajo se completa junto con un segundo trabajo de final de grado realizado por David Amores Dolz y titulado “Sistema de automatización y control de una zona de ocio con piscina y cerramiento abatible”, que en conjunto forman la automatización integral de todas las estancias de una vivienda aislada con parcela, piscina y zona de ocio.

Palabras clave: home assistant, control, automatización, google assistant, smartphone, climatización, iluminación, sistema de seguridad, video vigilancia.

Abstract

In this final degree project, Home Assistant will implement a comprehensive automation and control system for a home and a garage door, allowing control from a smartphone or PC of lighting, air conditioning, security system, video surveillance. and opening of the garage door by reading the license plate. All this will be carried out using low-cost elements that can be easily incorporated into existing installations.

This work is completed together with a second final degree project carried out by David Amores Dolz and entitled "Automation and control system of a leisure area with swimming pool and folding enclosure", which together form the integral automation of all the rooms of an isolated house with a plot, swimming pool and leisure area.

Keywords: home assistant, control, automation, google assistant, smartphone, air conditioning, lighting, security system, video surveillance.

Resum

En aquest treball de final de grau s'implementarà amb Home Assistant un sistema integral d'automatització i control d'un habitatge i una porta de garatge, permetent el control des d'un telèfon intel·ligent de la il·luminació, climatització, sistema de seguretat, vídeo vigilància i obertura de porta de garatge mitjançant lectura de matrícula. Tot això es durà a terme utilitzant elements de baix cost i fàcilment incorporables en instal·lacions ja existents.

A més del control, també es programaran diferents automatitzacions dels elements del sistema, la qual cosa permetrà minimitzar la interacció per part de l'usuari, anticipant-se el sistema a les necessitats habituals i aconseguint al seu torn una millora notable en el consum d'energia elèctrica de l'habitatge.

Per a dotar a l'usuari d'una major comoditat i independència, s'integrarà tot el control del sistema amb Google Assistant, podent dur a terme el control total de l'habitatge a través d'ordres de veu, sense necessitat de fer un ús intensiu del telèfon intel·ligent.

Aquest treball es completa juntament amb un segon treball de final de grau realitzat per David Amores Dolz i titulat “Sistema d'automatització i control d'una zona d'oci amb piscina i tancament abatible”, que en conjunt formen l'automatització integral de totes les estades d'un habitatge aïllat amb parcel·la, piscina i zona d'oci.

Paraules clau: home assistant, control, automatització, google assistant, telèfon intel·ligent, climatització, il·luminació, sistema de seguretat, vídeo vigilància.

Tabla de contenidos

Índice de figuras

1. Introducción

Este trabajo surge de la necesidad de encontrar un sistema de automatización de bajo coste para una vivienda y una puerta de garaje que permita integrar cualquier dispositivo existente en el sistema sin necesidad de que este sea de una marca determinada ni depender de una lista de compatibilidad reducida de elementos. En la actualidad, tenemos un amplio abanico de productos comerciales dirigidos al control de elementos de una vivienda, pero en todos los casos nos encontramos con dos restricciones muy frecuentes, la primera es utilizar todos los elementos de una misma marca, quedando bloqueado el control de dispositivos que dicha marca no ofrezca en su catálogo. La segunda restricción es el coste de los elementos, dado que debemos elegir la misma marca para cualquier tipo de elemento, perdiendo flexibilidad en funcionalidad y coste.

En caso de tener interés en la utilización de elementos de control de diferentes marcas, debido al coste de los mismos o a sus características, nos encontramos con la problemática de tener que hacer uso de diferentes aplicaciones para el control de cada elemento, además de no poder realizar automatizaciones interesantes debido a la incomunicación entre los elementos de diferentes marcas.

Para dar solución a estos problemas, nos animamos a crear un sistema de bajo coste basado en Home Assistant, en el que podamos integrar el mayor número posible de elementos de diferentes fabricantes pudiendo realizar un sistema integral automatizado que permita conectar y controlar todos estos elementos entre sí. Esto permitirá poder realizar la selección idónea de elementos de control desde el punto de vista de nuestra instalación existente, sin tener que realizar costosas y complejas modificaciones en las instalaciones y equipamientos actuales de la vivienda.

1.1 Motivación

La motivación de este trabajo, surge como consecuencia de haber terminado el grado en ingeniería informática, compartir con mi hermano la misma profesión y afición por la informática y disponer hasta el momento de un sistema de control de la vivienda con muy poca flexibilidad. Por todo lo anterior y tras realizar un pequeño estudio de la plataforma Home Assistant, decidimos llevar a cabo la implementación de un sistema de control y automatización no solo que sustituya el que teníamos hasta la fecha, sino que amplíe el control a todas las estancias de nuestra propia vivienda, sobre todo buscando la facilidad y minimizar el tiempo a la hora de integrar un nuevo elemento en el futuro.

A raíz de lo anterior, surgió y se llevó a cabo el proyecto, con un extra de exigencia por ser el trabajo de final de grado.

Debido a la envergadura del proyecto y las diferentes disciplinas que se tratan en el mismo, pudimos dividir todos los objetivos en dos trabajos de fin de grado, tratando en el actual la implementación del sistema Home Assistant, el control de la vivienda y la puerta del garaje,

dejando así para el alcance del trabajo de David Amores Dolz el control de la piscina, de una zona de ocio y de un cerramiento abatible.

1.2 Objetivos

El objetivo principal del proyecto es controlar todos aquellos elementos de uso habitual que requieran desplazarse al usuario, teniendo así todo el control al alcance de la mano, además de automatizar todo aquello que sea posible para evitar depender de las acciones del usuario.

A continuación, resumimos una serie de objetivos que queremos alcanzar y que vamos a trabajar en el proyecto, como son:

- Apertura de la puerta de garaje mediante lectura de matrícula para evitar el uso de dispositivos en el instante anterior a la llegada a casa.
- Tener la posibilidad de saber que la puerta del garaje ha sido cerrada correctamente tras salir de la vivienda.
- Iluminar la parcela cuando se acceda por la noche, antes de llegar a los interruptores interiores de la vivienda.
- Encender luces de cortesía cuando anochece de manera automática.
- Control de iluminación de diferentes estancias.
- Permitir el control de la alarma Risco desde el mismo sistema de control de la vivienda, sin tener que hacer uso de la aplicación de la propia marca.
- Evitar salir de casa sin dejar la alarma armada por un descuido.
- Minimizar el consumo de electricidad del botellero, ya que solo es utilizado durante el fin de semana.
- Poder controlar la climatización de forma remota para acondicionar la estancia antes de llegar a la vivienda.
- Realizar la supervisión del sistema de cámaras Hikvision desde el mismo sistema sin tener que hacer uso de la aplicación de la propia marca.
- Permitir encender y lanzar trabajos en la impresora 3d en remoto para evitar esperas al llegar a la vivienda.

1.3 Estructura de la memoria

En este apartado, resumiremos como se estructuran todos los contenidos de la memoria que da vida a este proyecto.

En primer lugar, analizaremos la oferta de productos existentes en el mercado, que podrían cubrir la funcionalidad del proyecto de manera total o parcial, realizando una crítica de cada uno de ellos que nos llevará a presentar nuestra propuesta de sistema de automatización integral.

En base a nuestra propuesta, analizaremos las diferentes partes del problema junto a sus posibles soluciones, descartando las menos óptimas y argumentando el porqué de la solución elegida, mostrando una lista de componentes del sistema y su valoración económica.

Una vez hemos mostrado la solución elegida, daremos paso a describir el diseño de la solución, en el que explicaremos la arquitectura del sistema, así como la tecnología y software utilizados. En este punto también profundizaremos en la descripción de cada uno de los componentes hardware que componen el sistema.

Tras la descripción de todos los elementos y tecnologías del sistema, se desarrollará el proyecto paso a paso, mostrando todos los detalles de implementación del mismo para su puesta en marcha, como las líneas de código, configuraciones o procedimientos de instalación.

Por último, se mostrarán fotografías reales de la implantación de todo el sistema en funcionamiento, así como las conclusiones a las que hemos llegado tras el desarrollo y puesta en marcha del mismo.

2. Estado del arte

En la actualidad, disponemos de una gran variedad de posibilidades a la hora de elegir los elementos que conforman el sistema de control, sobre todo, dependiendo de las particularidades funcionales y el objetivo a alcanzar. En nuestro caso, hemos intentado elegir un dispositivo de bajo coste lo más generalista posible como un Sonoff[1] Mini R2 para todo aquello que sea controlable mediante el corte y suministro de electricidad. Por otro lado, hemos elegido Raspberry Pi[2] para todos aquellos sistemas que tengan una complejidad superior, como puede ser la lectura de placas de matrícula, o la ejecución de Home Assistant[3].

2.1 Crítica al estado del arte

Realizando una búsqueda de productos existentes en el mercado actualmente, podemos encontrar diferentes opciones, teniendo en común la falta de operatividad con elementos de otras marcas.

Estos dispositivos están diseñados para cubrir las necesidades objetivas de cada elemento, sin ofrecer una perspectiva del control total de la vivienda.

A continuación, presentamos algunos de los productos evaluados:

- **Cámara para lectura de matrículas Hikvision[4] DS-2CD3646G2/P-IZS**

  En este dispositivo encontramos una cámara con unas características excepcionales para la captura y lectura de placas de matrícula, incluso en condiciones de luz desfavorables.

  Hikvision es una marca reconocida y de gran prestigio en el sector de la seguridad y video vigilancia, pero no tiene una línea de productos centrada en el control de una vivienda, por lo que no podemos encontrar dentro de la misma marca otros elementos para automatizar por ejemplo la climatización.

  A pesar de no existir esos otros dispositivos que podamos necesitar para realizar diferentes controles, este equipo posee una API que permite realizar la integración de este equipo en cualquier otro sistema.

1 https://sonoff.tech/products/
2 https://www.raspberrypi.com/
3 https://www.home-assistant.io/
4 https://www.hikvision.com/es/

Además, por tratarse de un dispositivo de uso profesional, presenta un coste demasiado elevado, en torno a 1200€, sin que necesitemos el gran potencial que presenta este producto ya que nuestro objetivo es abrir una puerta en una entrada con muy bajo nivel de tráfico.

Figura 1 Cámara Hikvision DS-2CD3646G2/P-IZS

- ## Relé Smart Wifi TUYA[5]

Este dispositivo de uso común nos permite controlar el encendido y apagado de cualquier elemento alimentado por la red eléctrica, desde una bombilla hasta un frigorífico.

Mediante una conexión Wifi, el dispositivo se conecta al cloud de TUYA, pudiendo controlar dicho dispositivo desde la app TUYA en cualquier smartphone.

El coste de este equipo es de tan solo 8,95€, lo que lo haría un perfecto candidato si no fuera porque en la aplicación de esta marca no podríamos integrar una cámara para lectura de matrículas o un sistema de seguridad.

Figura 2 Relé Smart Wifi TUYA

5 https://www.tuya.com/

- **Xiaomi Mi Home**[6]

Se trata de un sistema completo de control doméstico, ofrece tanto los dispositivos hardware como la app para el control de los mismos.

Este sistema puede ser la solución que más se acerca a nuestras necesidades, pero no llega a dar cobertura a la integración de sistemas externos a la marca que podemos necesitar para el caso de la lectura de matrícula o si deseamos contar con un sistema de seguridad robusto.

Dentro de esta solución, se ofrece un sistema de seguridad compuesto de varios sensores que dista mucho de la robustez que debe ofrecer un sistema de seguridad profesional, como puede ser contar con un sistema de comunicaciones de backup mediante GPRS/4G o contar con la certificación de grado 2 que nos asegure que el sistema es difícil de sabotear.

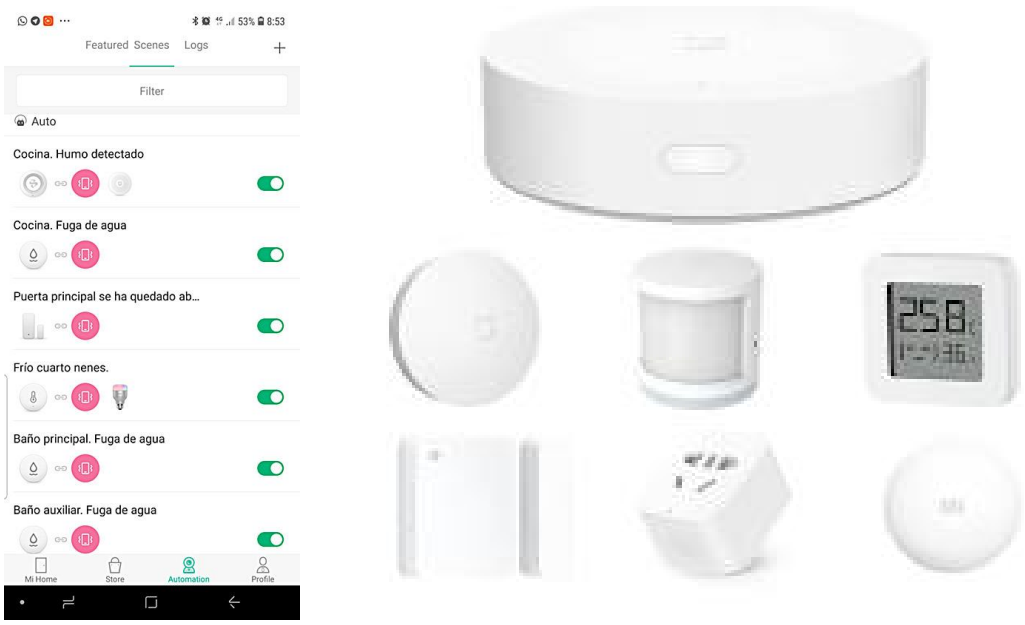

Figura 3 Elementos de Mi Home

2.2 Propuesta

Tras la evaluación anterior y debido a la falta de cumplimiento de nuestros objetivos por parte de los sistemas comerciales actuales, hemos decidido hacer uso de la plataforma Home Assistant como base para nuestro sistema de control y automatización.

6 https://www.mi.com/es/smart-home

Elegimos esta plataforma por ser de código abierto, debido a su madurez y al gran número de integraciones de las que ya dispone tanto con protocolos existentes como con dispositivos de marcas comerciales. El hecho de contar con el código fuente del sistema, nos capacita para poder combatir cualquier deficiencia que podamos encontrar durante el desarrollo del sistema.

En la parte física, optamos por utilizar Raspberry Pi como equipo que ejecute la plataforma Home Assistant, obteniendo así un equipo de reducidas dimensiones y bajo consumo para el que además existe un sistema operativo específico de Home Assistant, por lo que ya está comprobada la compatibilidad y estabilidad del equipo.

Para el control de dispositivos, contaremos con dos elementos diferentes, por un lado, el relé wifi Sonoff Mini R2, al que le cargaremos el firmware tasmota[7] para independizarlo de la marca y poder comunicar con él a través del protocolo MQTT. Por otro lado, utilizaremos Raspberry Pi para los controles complejos como es el de la cámara con detección de matrículas haciendo uso de la librería OpenALPR[8].

7 https://tasmota.github.io/docs/

8 https://github.com/openalpr/openalpr

3. Análisis del problema

Como venimos indicando en puntos anteriores, una de las dificultades a la que nos enfrentamos es que no existe un sistema comercial capaz de controlar todos los objetivos previstos en este proyecto desde una misma aplicación y de manera conjunta.

El proyecto que pretendemos desarrollar trata de construir un sistema que se pueda implementar de manera sencilla en cualquier instalación existente, posibilitando integrar cualquier dispositivo analógico o digital sin presentar demasiadas restricciones.

La mayor dificultad es la necesidad de tener que integrar dispositivos de diferentes fabricantes contra el mismo sistema, para que así puedan operar entre ellos y poder cumplir todos los objetivos de control y automatización previstos.

3.1 Identificación y análisis de soluciones posibles

Previamente al desarrollo del sistema planteado en este proyecto, hemos valorado diferentes soluciones que dan cobertura a la funcionalidad requerida, pero todas ellas han sido descartadas por diferentes motivos que se exponen a continuación.

- **Desarrollo del sistema automatizado utilizando soluciones comerciales.**

Esta solución es una de las más ágiles y sencillas de implementar, pues en el mercado existe una gran variedad de dispositivos que de forma separada pueden llegar a cubrir todos los requerimientos de automatización que tengamos. Estos dispositivos están diseñados para que cualquier usuario sea capaz de instalarlos y configurarlos.

Buscando una marca con la que poder adquirir todos los dispositivos, nos damos cuenta que no existe ningún fabricante que pueda cubrir el total de las necesidades de automatización, pues el propio término automatización cubre un amplio espectro de posibilidades. Algunos de ellos son especialistas en hogar, otros en jardín, piscinas, energía, iluminación, etc.

Siendo consciente de que debemos contar con dispositivos de varios fabricantes, empezamos a buscar una solución que nos permitiera controlar dispositivos de diferentes marcas desde un sistema centralizado, pues queremos huir de tener que utilizar varias aplicaciones para el control del sistema, además en algunos casos requerimos la interacción entre dispositivos, como puede ser un sensor de movimiento y una luz.

Encontramos la plataforma TUYA, un sistema cloud diseñado para el control del hogar, el cual cuenta con la integración de multitud de fabricantes de dispositivos de automatización. A pesar de ser un sistema con muchas posibilidades, lo descartamos debido a la imposibilidad de poder realizar nuestras propias integraciones que podamos necesitar fuera del abanico de marcas compatibles. Por otro lado, tampoco nos ofrece un sistema de programación de

automatizaciones y control de eventos potente, más allá de programar acciones periódicas o lanzar ciertas acciones cuando suceda un evento concreto.

- **Desarrollo de aplicación personalizada para smartphone.**

Puesto que uno de nuestros requisitos es tener la posibilidad de integrar cualquier dispositivo en el sistema, plantemos el desarrollo de una aplicación personalizada para android, pues de esta forma no tendríamos límites en cuanto a las posibilidades del software de control.

Con la idea de realizar un prototipo para evaluar la prueba, desarrollamos una aplicación para android mediante el framework Ionic. Esto nos permite mostrar cualquier control en pantalla que envíe una orden mediante HTTP a cualquier dispositivo de la red.

Para todos aquellos dispositivos que no tengan acceso a la red, como un aire acondicionado convencional, o para aquellos que no podamos controlar mediante peticiones HTTP, planteamos programar un microcontrolador ESP32 que reciba nuestra orden desde la aplicación y la convierta en una señal eléctrica para controlar un relé, un emisor de infrarrojos o cualquier otro elemento.

Una vez desarrollado el prototipo, nos damos cuenta que la tarea de integrar nuevos dispositivos en el sistema se vuelve muy costosa y lenta, pues cada vez tenemos que haces cambios mediante código en la interfaz de la aplicación, compilar y redistribuir la misma a todos los usuarios del sistema.

Además, al contener el microcontrolador la lógica de parte del sistema, cualquier cambio provoca tener que modificar el firmware y volver a grabarlo físicamente en el dispositivo.

3.2 Solución propuesta

- **Diseño y desarrollo de unidades de control mediante Raspberry Pi y Sonoff Mini R2 con integración en Home Assistant.**

Elegimos el software de código abierto Home Assistant, especializado en centralizar el control y automatización de todos los sistemas que se pueden encontrar en una vivienda y con capacidad para poder realizar cualquier desarrollo personalizado mediante lo que en el software se conoce como “integraciones” o “complementos”.

Puesto que la centralización software del sistema recae en el software Home Assistant, en él podremos contar con toda la información y datos existente en cualquier elemento del sistema, facilitando así las automatizaciones, control de estados, reacciones a eventos, integraciones con los usuarios, etc.

La comunicación entre los diferentes elementos del sistema se realizará mediante TCP/IP, en algunos casos de manera cableada por ethernet y en otros casos vía radio mediante WiFi. Por

esto, es posible ejecutar Home Assistant sobre una gran variedad de dispositivos como servidores, ordenadores, Raspberry, Intel NUC[9] o cualquier máquina virtual.

Debido a que el software centralizado se va a ejecutar sobre un equipo sin conexión directa con los elementos de control, necesitaremos desplegar pequeños dispositivos autónomos que se comuniquen con Home Assistant mediante la red y para ello utilizarán el protocolo de mensajes MQTT, especialmente diseñado para este tipo de cometidos.

Entre los dispositivos que desarrollaremos e integraremos en el alcance de este proyecto, encontraremos:

- Relés WiFi para controlar el encendido y apagado de cargas alimentadas a 230VAC

- Dispositivo emisor de infrarrojos para control de equipos con mando a distancia.

- Cámaras de video vigilancia.

- Sistema de seguridad.

- Controlador de puerta con lectura de matrículas.

9 https://www.intel.es/content/www/es/es/products/details/nuc.html

3.3 Presupuesto sistema automatización para puerta de garaje

A continuación, se detalla un desglose de todos los materiales y dispositivos empleados en el sistema de automatización de la puerta de garaje con lectura de matrículas:

Producto	Coste Unidad	Cantidad	Total
Raspberry Pi 3 B+	35,00 €	1	35,00 €
Tarjeta MicroSD 32GB	7,50 €	1	7,50 €
Fuente alimentación Raspberry Micro USB	6,90 €	1	6,90 €
Cámara IP Hikvision WDR IP67	76,80 €	1	76,80 €
Fuente alimentación 12VDC	5,50 €	1	5,50 €
Placa de relés de 4 canales	9,00 €	1	9,00 €
Switch 5 puertos	10,00 €	1	10,00 €
Sensor de ruido	4,90 €	1	4,90 €
Sensor inductivo	12,30 €	1	12,30 €
Horas de desarrollo	45,00 €	20	900,00 €
		TOTAL:	1067,90 €

3.4 Presupuesto sistema automatización vivienda

A continuación, se detalla un desglose de todos los materiales y dispositivos empleados en el sistema de automatización de la vivienda, teniendo en cuenta el alcance de los elementos automatizados:

Producto	Coste Unidad	Cantidad	Total
Sonoff Mini R2	6,50 €	1	6,50 €
Sonoff Basic R2	25,00 €	1	25,00 €
Emisor IR TSAL6400	1,40 €	1	1,40 €
Transistor 2N3904	1,20 €	1	1,20 €
Cámara IP Hikvision	58,50 €	3	175,50 €
Central alarma Risco	450,00 €	1	450,00 €
Horas de desarrollo	45,00 €	32	1440,00 €
		TOTAL:	2099,60 €

4. Diseño de la solución

En este apartado, detallaremos el diseño conceptual de la solución final elegida, así como todos los requisitos de software y hardware necesarios para llevar a cabo el proyecto.

4.1 Arquitectura del sistema de la puerta de garaje

A continuación, se muestra un diagrama de bloques parcial, que representa cómo se estructura todo el sistema de automatización de la puerta de garaje, identificando de forma clara los componentes del sistema y la relación que existe entre ellos.

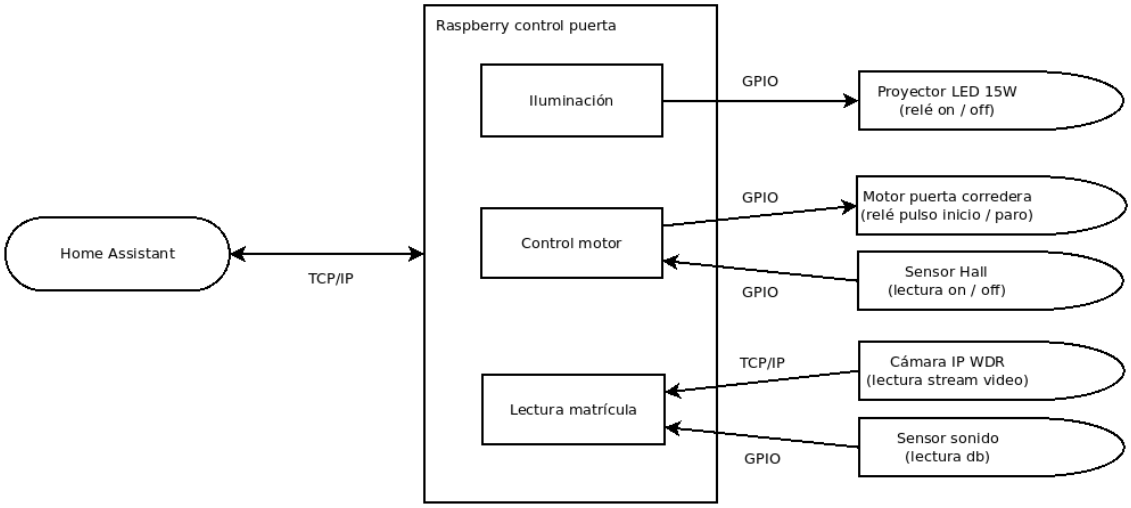

Figura 4 Diagrama de bloques sistema automatización puerta de garaje.

4.2 Arquitectura del sistema de automatización de vivienda

A continuación, se muestra un segundo diagrama de bloques parcial, que representa cómo se estructura todo el sistema de automatización de la vivienda, identificando de forma clara los componentes del sistema y la relación que existe entre ellos.

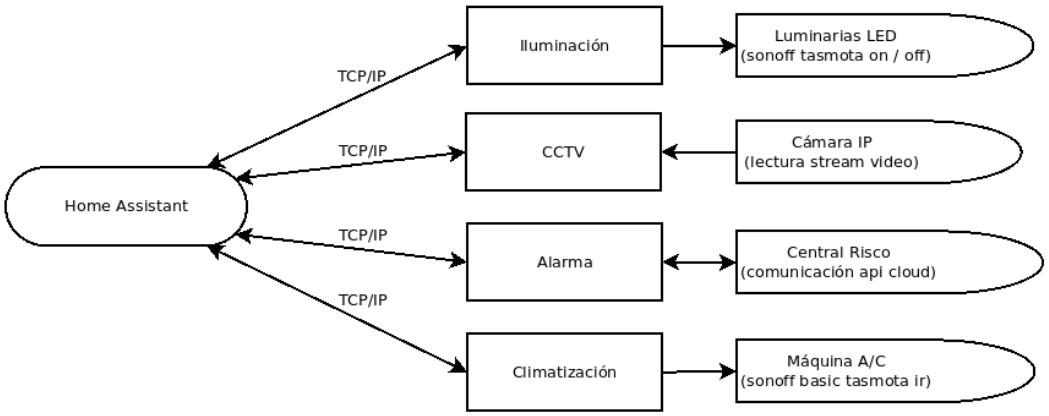

Figura 5 Diagrama de bloques sistema automatización vivienda.

4.3 Tecnología utilizada

En este apartado detallaremos en profundidad todas las herramientas software y tecnologías que se han empleado en el desarrollo del proyecto.

4.3.1 Sistema operativo Raspberry Pi OS

Raspberry Pi OS[10], anteriormente conocido como Raspbian, es la distribución de Linux, basada en Debian, oficial del mini ordenador Raspberry Pi.

Podemos encontrar diferentes versiones del sistema, con escritorio instalado, con escritorio y programas recomendados o la versión lite que viene sin ningún escritorio y con un peso de tan solo 297MB. Además, podemos encontrar estas versiones tanto para arquitectura de 32 bits como para 64 bits.

Puesto que el uso que vamos a hacer del sistema operativo no es como usuario, si no como servidor o sistema autónomo de automatización, hemos elegido la versión lite del sistema que

10 https://www.raspberrypi.com/software/

nos brinda un mayor rendimiento, menor tamaño de sistema y minimiza los problemas que pueden surgir ante actualizaciones o reinicios inesperados debido al menor número de paquetes instalados en el sistema.

4.3.2 Firmware Tasmota

Tasmota es un firmware de código abierto creado para ejecutarse sobre los microcontroladores ESP8266, ESP32, ESP32-S o ESP32-C3, todos ellos de la marca Espressif[11]. Su función principal es facilitar el control de los dispositivos mediante comunicación MQTT a través de redes WiFi, ofreciendo una interfaz web mediante la cual permite parametrizar y configurar el comportamiento del dispositivo.

Dentro del firmware Tasmota encontramos diferentes compilaciones del propio firmware, destinadas a las diferentes tipologías de dispositivos, como, por ejemplo:

- tasmota: compilación general, destinada a actuadores y sensores de uso común.
- tasmota-sensors: destinada a dispositivos que actúan como sensores.
- tasmota-ir: destinada a dispositivos que actúan como emisores y receptores de infrarrojos.
- tasmota-zbbridge: destinada a dispositivos que actúan como pasarelas o coordinadores zigbee.

4.3.3 Lenguajes de programación

- **Python:**

Dentro del ecosistema Raspberry, el lenguaje más utilizado es python, es por ello que es en el lenguaje en el que encontramos la mayoría de librerías de uso común como puede ser la que permite el acceso a sus puertos GPIO (entradas y salidas) o la que facilita el uso del protocolo de comunicaciones MQTT. Al disponer del gestor de paquetes PIP, se facilita la tarea de instalación y mantenimiento de dependencias y librerías.

Figura 6 Logotipo lenguaje programación Python.

11 https://www.espressif.com/en/products/socs

- **YAML:**

YAML es un formato de serialización de datos muy utilizado para definir ficheros de configuración, entre otros usos.

Home Assistant utiliza este formato por defecto para la configuración del sistema, por lo que debemos hacer uso del mismo para poder trabajar con los archivos de configuración en Home Assistant.

```
sensor:
  - platform: template
    sensors:
      entidades_perdidas:
        friendly_name: Dispositivos desconectados
        unit_of_measurement: entities
        icon_template: "{{ 'mdi:check-circle' if is_state('sensor.entidades_perdidas','0') else 'mdi:alert-circle' }}"
        value_template: >
          {{ states|selectattr('state','in',['unavailable','unknown','none'])
            |rejectattr('domain','in',['group', 'person', 'device_tracker', 'automation', 'script'])
            |rejectattr('entity_id','search','smartphone')
            |rejectattr('entity_id','search','octoprint')
            |map(attribute='entity_id')
            |list|count }}
        attribute_templates:
          entities: >
            {{ states|selectattr('state','in',['unavailable','unknown','none'])
              |rejectattr('domain','in',['group', 'person', 'device_tracker', 'automation', 'script'])
              |rejectattr('entity_id','search','smartphone')
              |rejectattr('entity_id','search','octoprint')
              |map(attribute='entity_id')
              |list }}
```

Figura 7 Ejemplo de un archivo YAML.

4.3.4 Entorno de desarrollo

Tanto Home Assistant como Raspberry Pi OS son sistemas operativos basados en Linux. En el caso de Home Assistant contamos con una interfaz web para poder interactuar con el sistema, la cual nos permitirá instalar un editor de ficheros propio.

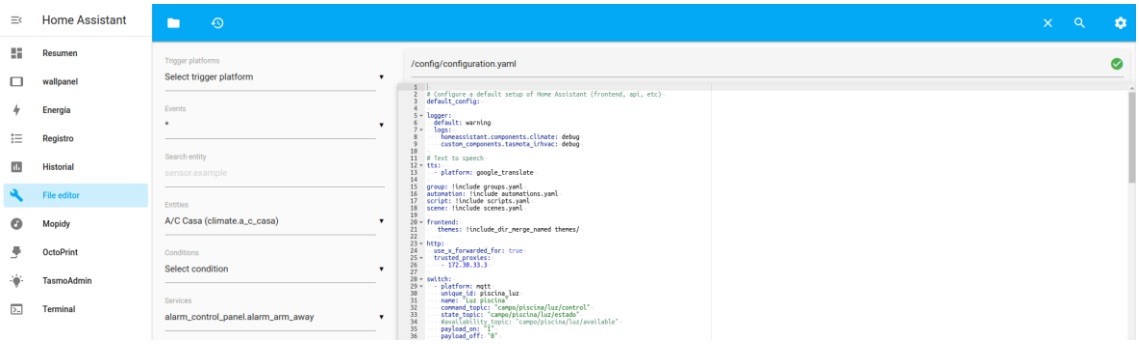

Figura 8 Editor de texto Home Assistant

En el sistema Raspberry Pi OS no contamos con ningún entorno gráfico, ya que haremos uso de la versión lite. En su defecto, todo el control del sistema lo haremos a través de la shell del sistema, a la que conectaremos mediante SSH. Dentro de la shell haremos uso del editor de texto nano[12], el cual nos permitirá desarrollar y editar los scripts necesarios.

Figura 9 Editor de texto nano

4.3.6 Librerías o paquetes

Para el desarrollo de la parte software del proyecto ha sido necesario hacer uso de algunas librerías y paquetes existentes, lo cual ha facilitado y agilizado el trabajo a realizar.

A continuación, detallamos todas las dependencias de las que hemos hecho uso, así como el método de instalación de las mismas.

- **Paquete Pip:**

Es el gestor de paquetes oficial de Python, el cual nos permite instalar cualquier paquete de manera ágil y sencilla, permitiendo además obtener la última versión existente. Para instalarlo en su versión 3 en un sistema basado en debian debemos ejecutar el comando siguiente.

```
sudo apt-get install python3-pip
```

Una vez instalado, podremos obtener e instalar cualquier paquete o dependencia en Python 3 utilizando el siguiente comando:

```
pip3 install NombreDelPaquete
```

12 https://www.nano-editor.org/

- **Paquete paho-mqtt:**

Mediante esta librería, podremos establecer una comunicación mediante MQTT con el broker principal de manera muy sencilla, con tan solo unas líneas. Además de las funciones para publicar y suscribirse a un tópico, esta librería nos brinda la funcionalidad completa disponible en las comunicaciones MQTT como la calidad de servicio QOS o la función de retención de estados retain.

Para instalar el paquete debemos ejecutar el siguiente comando.

```
pip3 install paho-mqtt
```

- **Paquete RPi.GPIO:**

Para el control de los puertos GPIO de Raspberry Pi desde Python, debemos instalar la librería Rpi.GPIO, la cual ofrece una interfaz entre Python y el sistema operativo que nos permite hacer uso de las funciones necesarias para el control de puertos, pudiendo definir el comportamiento de los mismos, controlar su estado y demás funciones.

Para instalar el paquete debemos ejecutar el siguiente comando.

```
pip3 install RPi.GPIO
```

- **Paquete OpenALPR:**

El análisis de imágenes para la detección de objetos o identificación de texto es un proceso muy complejo que daría para el desarrollo de un proyecto por sí mismo, por eso nos ha sido imprescindible hacer uso de la librería OpenALPR, la cual nos brinda toda esta funcionalidad, con tan solo pasarle una imagen es capaz de devolvernos el texto de una placa de matrícula.

Para instalar el paquete debemos ejecutar el siguiente comando.

```
pip3 install openalpr
```

- **Paquete HikvisionAPI:**

Las cámaras de CCTV son dispositivos que tienen un alto nivel de seguridad que impide el acceso a su imagen, pues están diseñadas como parte de un sistema de control de seguridad, es por esto que el acceso a la imagen de la misma requiere de un procedimiento de autorización en el equipo y posterior captura de fotogramas del stream de video.

Para realizar esta tarea, debemos conocer bien la API del fabricante de la cámara con la que estamos trabajando, es por ello que hemos hecho uso de la librería HikvisionAPI, la cual ya está preparada para realizar todo este proceso con tan solo llamar a una función, a la que pasaremos los datos de autenticación.

Para instalar el paquete debemos ejecutar el siguiente comando.

```
pip3 install hikvisionapi
```

4.4 Componentes del sistema automatización puerta de garaje

En este apartado, detallaremos todos los componentes que se utilizan y componen el proyecto de automatización de la piscina, tanto software como hardware.

4.4.1 Placa base Raspberry Pi 3B+

Se trata de un mini ordenador de bajo coste que cuenta con un amplio abanico de conexiones y puertos de entrada y salida, lo que lo convierte en un equipo muy versátil que podemos utilizar como ordenador personal, equipo multimedia, servidor o como core de cualquier sistema.

Todo lo necesario para hacer funcionar el ordenador va incluido en la propia placa, procesador, memoria RAM, tarjeta gráfica, sonido, red, WiFi, USB, HDMI, etc. El único componente que podemos actualizar es el disco duro ya que se trata de una tarjeta micro sd que va introducida en un slot en la placa.

En nuestro caso, hemos elegido este dispositivo ya que es uno de los equipos de menor coste que pueden correr una distribución Linux, lo que nos permite realizar programas complejos y potentes de manera ágil haciendo uso de la extensa biblioteca de librerías python existentes.

A pesar de ser un equipo de bajo coste, su madurez en el mercado la hace destacar como uno de los equipos más robustos y estables de su gama.

Por último, a través de su cabecera de pines encontramos todas las conexiones que necesitaremos para poder conectar cualquier tipo de sensor, así como alimentaciones de 3.3V y 5V.

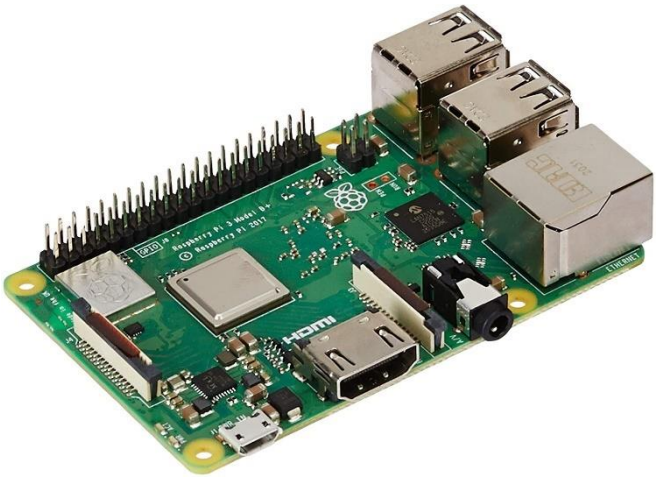

Figura 10 Raspberry Pi 3 B+

Especificaciones técnicas:

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
- 1GB RAM
- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board
- 100 Base Ethernet
- 40-pin extended GPIO
- 4 USB 2 ports
- Micro SD port for loading your operating system and storing data
- Upgraded switched Micro USB power source up to 2.5A

4.4.2 Switch Ethernet

Mediante un switch ethernet de uso común, se realizará la conexión de la Raspberry Pi y la cámara IP WDR al resto de la red existente, pudiendo comunicar cada uno de los dispositivos del sistema entre sí de forma simultánea.

Las principales características por las que hemos elegido este switch han sido el reducido tamaño, la posibilidad de poder fijarlo en vertical y la presencia de leds en el frontal que facilitan el mantenimiento, pudiendo saber si hay enlace en todos los dispositivos conectados con solo un vistazo.

Figura 11 Edimax Switch 5 x 10/100

4.4.3 Placa de relés (4 canales)

Esta placa de relés permite a la Raspberry Pi, la cual solo tiene entradas y salidas de señal (0-5v), controlar dispositivos que trabajen a tensiones diferentes. Con una señal de 5v procedente de la Raspberry Pi, se activará el relé que dará 230V a la luz LED o 12V a la electrónica del motor de la puerta.

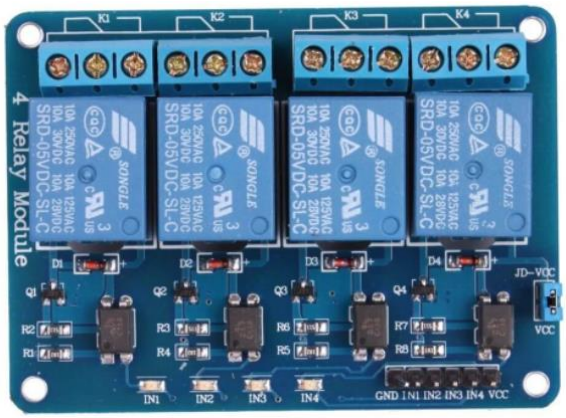

Figura 12 Placa de relés de 4 canales 5v

4.4.4 Cámara IP WDR IP67

La cámara que debemos utilizar debe cumplir una serie de características para permitir el correcto funcionamiento del sistema en todo momento.

Debido a que la cámara irá instalada en el exterior de la puerta del garaje, es necesario que esta cumpla con alguna certificación que nos garantice que puede funcionar durante tiempo prolongado a la intemperie, con temperaturas elevadas, lluvia, etc. En este caso cubrimos esta necesidad con la certificación IP67.

El otro aspecto importante, es conseguir una imagen nítida y clara de los vehículos que se sitúan delante de la cámara, de otra forma no sería posible interpretar el texto identificador de la placa de matrícula. Para esto, necesitamos una buena resolución que permita captar las letras a una distancia de 2 o 3 metros, en este caso 1920x1080.

Además, puesto que la cámara está en el exterior, habrá ocasiones en las que la luz directa del sol quemará la imagen capturada impidiendo ver el contenido de la placa de matrícula, por ello necesitaremos que la cámara cuente con la tecnología WDR, la que se encargará de ajustar la intensidad de luz recibida por la cámara de forma selectiva en cada zona de la imagen, impidiendo que aparezcan zonas quemadas en la imagen capturada.

Figura 13 Cámara Hikvision WDR IP67

4.4.5 Sensor de ruido

Este sensor se compone de un micrófono, un amplificador que potencia la señal y un regulador que nos permite ajustar a que nivel de ruido queremos que se active la salida digital D0.

Además de la salida digital, también cuenta con una salida analógica que entrega continuamente un voltaje de 0 a 5v proporcional al ruido captado por el micrófono. En este caso solo utilizaremos la salida digital ya que solo necesitamos detectar cuando el nivel de ruido sube por encima de un umbral determinado.

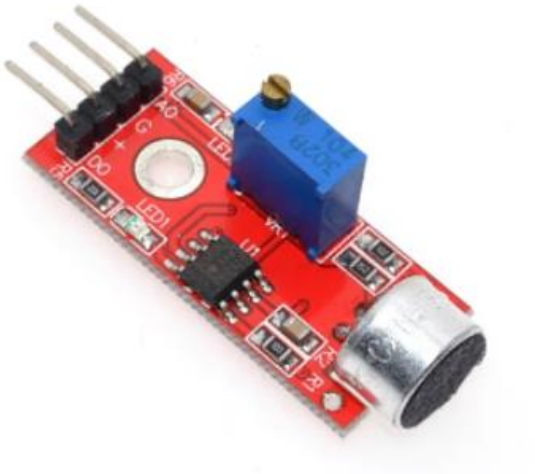

Figura 14 Sensor de ruido

4.4.6 Sensor inductivo

Los sensores inductivos están diseñados para detectar materiales ferrosos cercanos, sin necesidad de contacto. Para este proyecto utilizaremos un sensor de 12mm de diámetro con una capacidad de detectar un metal hasta 4mm de distancia.

Figura 15 Sensor inductivo

4.5 Componentes del sistema automatización vivienda

En este apartado, detallaremos todos los componentes software y hardware que se utilizan para el desarrollo del proyecto de automatización de la vivienda.

4.5.1 Sonoff mini R2

Este dispositivo es un interruptor remoto con posibilidad de conectar una maniobra manual, por defecto viene preparado para conectar y usar a través de la app de la propia marca, pero mediante una actualización de firmware podremos integrarlo en nuestro sistema.

Figura 16 Sonoff Mini R2

4.5.2 Sonoff basic R2 con emisor IR

El dispositivo Sonoff basic R2 está diseñado para controlar el encendido y apagado de una luminaria en remoto, mediante conexión WiFi, pero mediante la actualización de su firmware y añadiendo un emisor infrarrojo (TSAL6400 [13]) controlado por un transistor (2N3904 [14]), podremos convertirlo en un mando a distancia remoto integrable en el sistema mediante WiFi.

Figura 17 Sonoff basic R2, transistor, emisor IR

4.5.3 Cámara IP

Hemos elegido tres cámaras CCTV domo con conexión TCP/IP de la marca Hikvision por la relación calidad precio, pero para la integración se podría usar cualquier cámara del mercado que permita activar el protocolo ONVIF, algo que se encuentra de manera habitual en la mayoría de modelos de cámaras CCTV.

Figura 18 Cámara domo IP Hikvision

13 https://www.vishay.com/docs/81011/tsal6400.pdf

14 https://www.nteinc.com/specs/original/2N3904.pdf

4.5.4 Central alarma Risco[15]

Como encargado de la seguridad de la vivienda, hemos elegido una central Risco que ya vienen certificadas con grado 2, necesario para asegurar un funcionamiento correcto y robusto, además de ser imprescindible para permitir conectar la misma a una central receptora de alarmas.

Esta central viene con sus propios sensores de detección y permite ser controlada desde la propia app de la marca a través de la conexión cloud. El cloud de Risco permite realizar integraciones mediante una API, lo que aprovecharemos para poder conectar nuestro sistema Home Assistant con el cloud de Risco para poder controlar la central por completo.

Figura 19 Central Risco con sensor de presencia, contacto y mando

15 https://www.riscogroup.com/spain

5. Desarrollo de la solución propuesta

En este apartado explicaremos paso a paso, todos los puntos del desarrollo llevado a cabo desde el inicio hasta la puesta en marcha de cada uno de los dos sistemas de automatización objeto de este proyecto, empezando por la automatización de la puerta de garaje y terminando por la automatización de la vivienda.

5.1 Instalación Home Assistant en Raspberry Pi

Existen múltiples formas de instalación de Home Assistant, que en algunos casos dependen del equipo donde pretendemos utilizarlo. Podemos instalarlo en Windows, Linux, Mac, en una máquina virtual, en una placa Raspberry Pi, directamente en un equipo x86, etc.

Dependiendo del tipo de equipo, podemos elegir el tipo de instalación de Home Assistant, pudiendo instalarlo incluso con su propio sistema operativo. En la siguiente tabla podemos ver las diferencias funcionales que existen en los cuatro tipos de instalación.

	OS	Container	Core	Supervised
Automations	✅	✅	✅	✅
Dashboards	✅	✅	✅	✅
Integrations	✅	✅	✅	✅
Blueprints	✅	✅	✅	✅
Uses container	✅	✅	❌	✅
Supervisor	✅	❌	❌	✅
Add-ons	✅	❌	❌	✅
Backups	✅	✅[1]	✅[1]	✅
Managed OS	✅	❌	❌	❌

Figura 20 Comparativa tipos instalación Home Assistant

Nosotros hemos optado por la instalación de Home Assistant con su propio sistema operativo, que además de ser la más completa, es la recomendada para placas Raspberry Pi. Como desventaja, no podremos hacer uso de la Raspberry Pi para otro fin que no sea Home Assistant, mientras que en una instalación de tipo contenedor podríamos ejecutar varios contenedores a la vez aprovechando el mismo hardware.

La instalación del modo OS en Raspberry Pi es muy sencilla con el programa Balena Etcher[16]. Debemos seleccionar la opción flash from url e indicar la siguiente URL.

https://github.com/home-assistant/operating-system/releases/download/8.2/haos_rpi4-64-8.2.img.xz

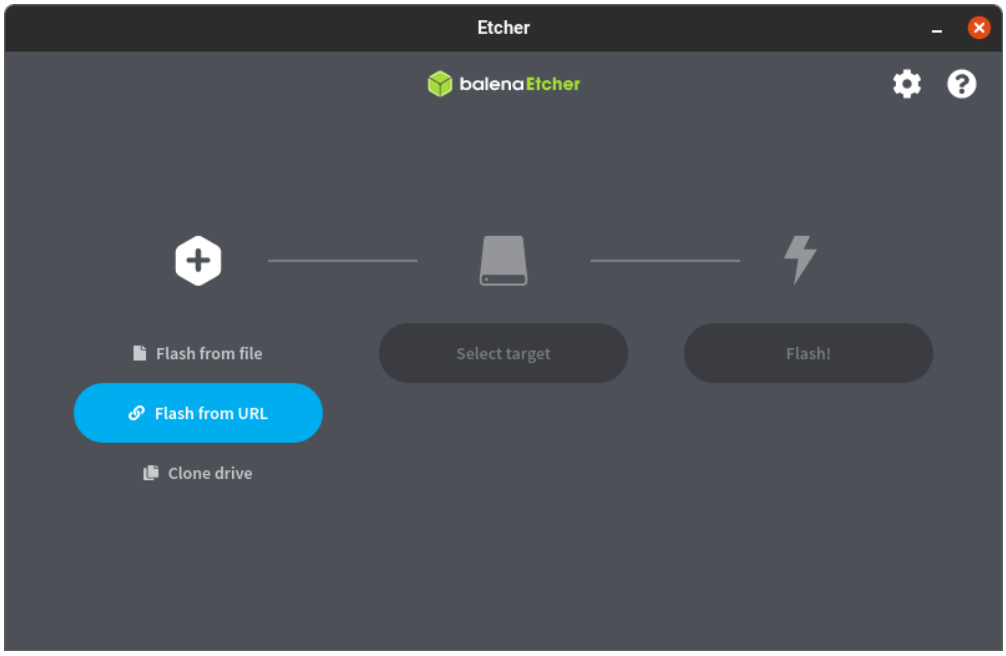

Figura 21 Cargar imagen desde URL en balena Etcher

A continuación, seleccionamos la tarjeta micro sd y pinchamos en el botón flash, quedando a la espera de que el proceso termine.

16 https://www.balena.io/etcher/

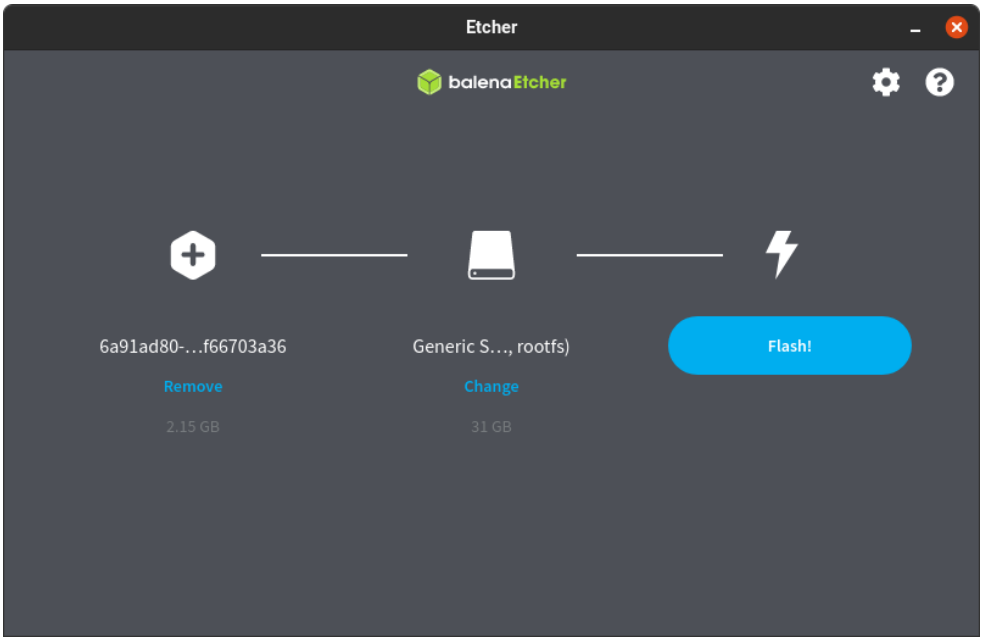

Figura 22 Grabar imagen en micro sd en balena Etcher

Una vez haya terminado el proceso, podemos retirar la tarjeta micro sd del lector, introducirla en la ranura de la Raspberry Pi y conectar los cables de alimentación y red.

En dos o tres minutos, podremos acceder a Home Assistant desde cualquier navegador web de la red, introduciendo la dirección http://homeassistant.local:8123

Durante el primer acceso a la interfaz de Home Assistant, nos pedirá los datos para poder crear una nueva cuenta de usuario, que será la que nos de acceso al sistema como administrador del mismo.

En la pantalla principal, encontraremos un menú a la izquierda que nos da acceso a todo el sistema, mostrando las siguientes opciones principales.

- Resumen: Muestra la pantalla principal de control, conocida como Lovelace, que podremos personalizar mediante componentes conocidos como tarjetas.
- Energía: Panel diseñado para llevar un seguimiento y control del consumo eléctrico de la instalación, así como de la producción de sistemas fotovoltaicos.
- Registro: Veremos un listado, ordenado cronológicamente, con todas las acciones que van ocurriendo en el sistema.
- Historial: Podemos ver un histórico de todos los estados por los que pasa una entidad en una línea de tiempo.
- Configuración: Nos da acceso a un nuevo menú que permite configurar parámetros de Home Assistant, instalar integraciones, crear usuarios, etc.

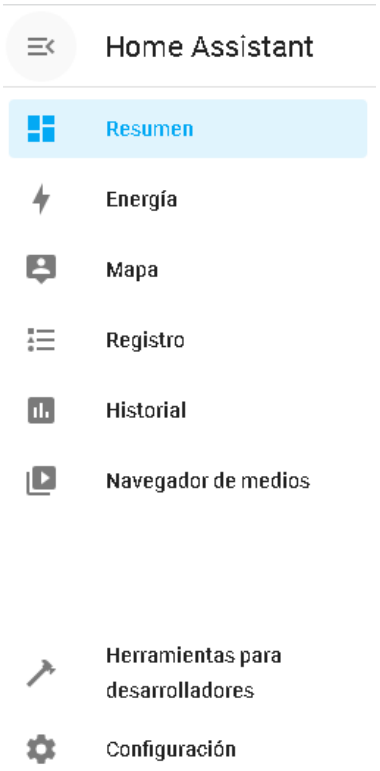

Figura 23 Menú principal Home Assistant

5.2 Desarrollo sistema automatización puerta de garaje

Durante el desarrollo del sistema de control de puerta de garaje, utilizaremos los siguientes dispositivos:

- Raspberry Pi modelo 3 B+.
- Placa de relés de 4 canales.
- Sensor de ruido.
- Cámara IP WDR.
- Switch.

El objetivo principal de este sistema, será permitir la apertura de la puerta mediante Home Assistant o bien mediante la lectura de la placa de matrícula del vehículo, confirmando en este último caso la acción de apertura con un toque de claxon. Aprovechando la inversión realizada en el sistema, controlaremos el encendido y apagado de la iluminación y monitorizaremos el estado de la puerta.

5.2.1 Instalación y configuración de Raspberry Pi OS

5.2.1.1 Instalación

Para la instalación del sistema operativo, necesitaremos una tarjeta micro sd de al menos 16GB y con una velocidad de lectura escritura equivalente al menos a clase 10 para asegurar un correcto funcionamiento del sistema.

En la propia tarjeta es donde se grabará el sistema operativo y desde donde será directamente ejecutado, haciendo las veces de disco duro.

Existen multitud de programas para grabar el sistema operativo en la tarjeta micro sd, todos ellos basados en la descarga de la imagen del sistema operativo y posterior grabado a la tarjeta. Podremos bajar la imagen de la página oficial y grabarla con nuestro programa favorito (dd, rufus[17], balena[18]…) o directamente utilizar la utilidad oficial Raspberry Pi Imager[19], la cual se encarga de hacer la descarga y posterior grabado en la tarjeta, tan solo elegiremos la versión del sistema operativo, la tarjeta de destino y pincharemos en el botón "WRITE".

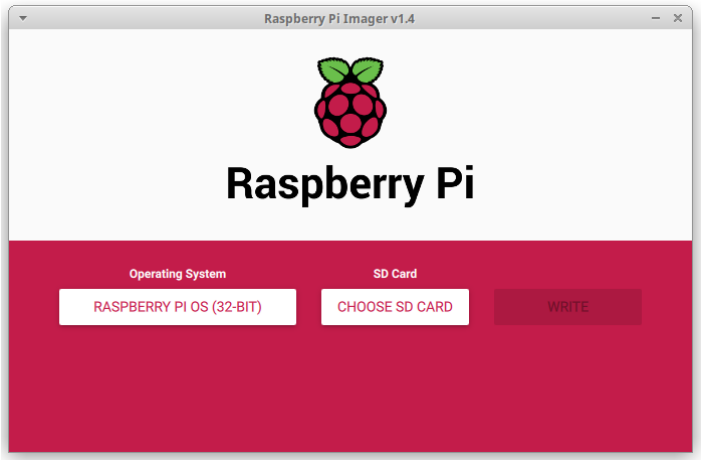

Figura 24 Captura de Raspberry Pi Imager

5.2.1.2 Configuración

Una vez completado el grabado del sistema operativo en la tarjeta, estaría listo para ser arrancado en la Raspberry Pi, pero necesitaríamos una pantalla y un teclado para poder interactuar con el sistema ya que por defecto Raspberry Pi OS no permite conexiones mediante ssh.

17 https://rufus.ie/es/

18 https://www.balena.io/etcher/

19 https://www.raspberrypi.com/software/

Para evitar el uso de periféricos, antes de extraer la tarjeta del pc, crearemos un archivo con nombre ssh en la raíz de la partición boot de la tarjeta. Con este archivo, Raspberry Pi OS activará automáticamente el servidor SSH en el primer arranque, permitiéndonos conectarnos al terminal sin la necesidad de conectar una pantalla a la Raspberry Pi.

Por último, solo queda extraer la tarjeta e introducirla en la ranura de la Raspberry Pi, conectar el cable de red y conectar la fuente de alimentación. Pasados un par de minutos ya debemos poder encontrar el equipo en la red, para ello podemos hacer uso del software Nmap ejecutando el siguiente comando, en el que deberemos especificar el rango de red correcto.

```
sudo nmap -PU 10.0.0.1/24
```

En la salida que nos dará Nmap[20], podemos observar los hosts encontrados en la red con su dirección MAC y al lado de esta el fabricante, con lo que podemos encontrar fácilmente la IP asignada a la Raspberry Pi.

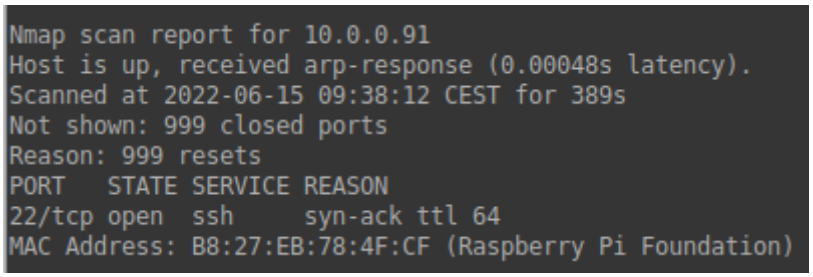

Figura 25 Captura salida comando Nmap

Una vez conocida la dirección IP del equipo, podemos abrir un terminal y conectarnos a la Raspberry Pi mediante el protocolo ssh. Para ellos utilizaremos el comando ssh, al que le indicaremos el usuario, por defecto pi, y la dirección IP de la Raspberry Pi, en este caso:

```
ssh pi@10.0.0.91
```

Al ejecutar el comando por primera vez, nos pedirá aceptar el fingerprint, que es un identificador único que valida y asegura la autenticidad de cada equipo. A continuación, nos pedirá la contraseña de acceso para el usuario pi, que por defecto es raspberry.

Ahora ya nos encontramos dentro del terminal de la Raspberry Pi por lo que cualquier comando que escribamos, será ejecutado directamente sobre ella.

En este punto, podemos empezar con la configuración del sistema Raspberry Pi OS. Para ellos ejecutaremos en primer lugar el comando:

```
raspi-config
```

20 https://nmap.org/

Nos aparecerá un menú interactivo en consola con todas las opciones de configuración del sistema, de todas ellas debemos entrar en Localisation Options y configurar tanto Locale como Time Zone, indicando en ambas nuestro idioma y zona horaria.

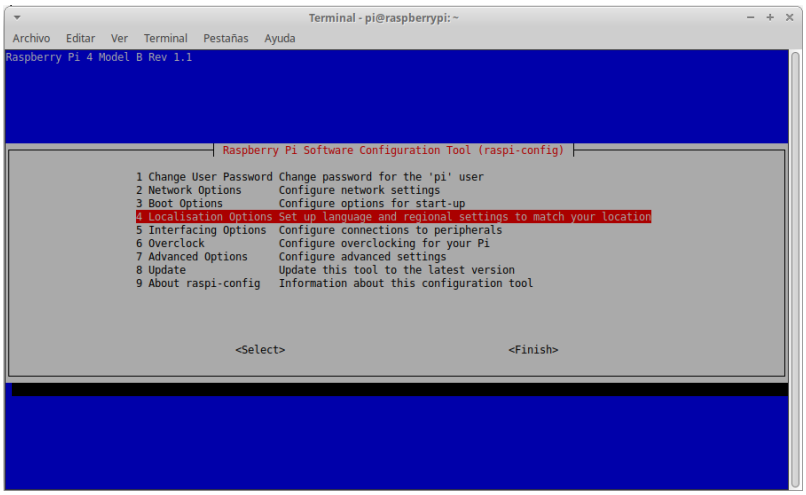

Figura 26 Menú principal raspi-config

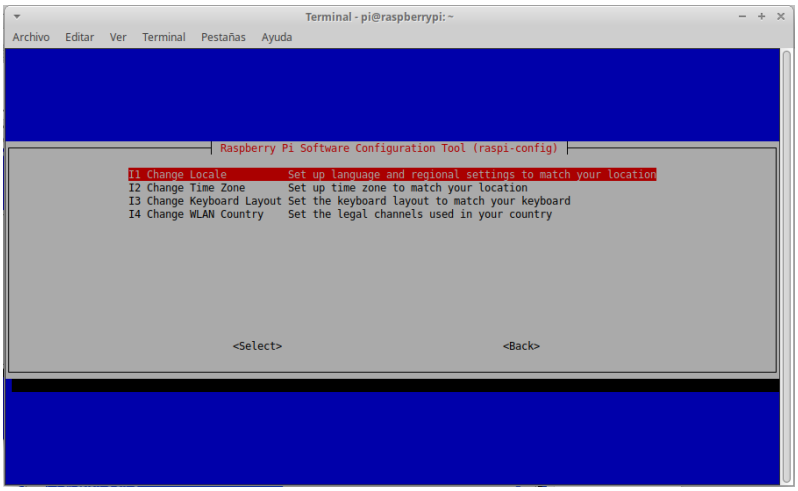

Figura 27 Menú localización raspi-config

Tras haber completado la configuración del sistema, procedemos a actualizar los paquetes del sistema a su última versión, para ello ejecutaremos un update con el gestor de paquetes, el cual actualizará la lista de paquetes y posteriormente un upgrade para actualizar dichos paquetes del sistema.

```
sudo apt-get update

sudo apt-get upgrade
```

Cuando finalice el proceso de actualización de los paquetes del sistema, tendremos la Raspberry Pi lista para funcionar y para poder instalar el software que necesitemos.

5.2.2 Instalación de dependencias y librerías en Raspberry Pi OS

Para poder ejecutar el script que se encargará de la lectura de matrículas y el script que conectará la Raspberry Pi con Home Assistant, el sistema debe contar con todas las librerías y dependencias necesarias, las cuales se listan a continuación.

Ambos scripts tienen en común el lenguaje utilizado para su desarrollo, por lo que en primer deberemos instalar el entorno python3 y su gestor de paquetes con el siguiente comando.

```
sudo apt-get install python3 python3-pip
```

Otra funcionalidad común que compartirán ambos scripts, es el manejo de los puertos de entrada y salida conocidos como GPIO. Para poder manejarlos desde python3, instalaremos la librería correspondiente con el siguiente comando.

```
sudo pip3 install RPi.GPIO
```

En el script encargado de realizar la lectura de matrículas utilizaremos la suite openalpr, para lo que necesitaremos instalar las librerías openalpr en el sistema y la interfaz para poder hacer uso de ellas desde el entorno python3, además también utilizaremos la librería hikvisionapi para facilitar la captura del stream de la cámara. Instalaremos todo con los comandos siguientes.

```
sudo apt-get install libopenalpr2

sudo pip3 install openalpr

sudo pip3 install hikvisionapi
```

Puesto que existirá un segundo script de python3 encargado de realizar la integración con Home Assistant y controlar las salidas de la Raspberry Pi, debemos instalar también la librería que nos facilita la comunicación MQTT con el siguiente comando.

```
sudo pip3 install paho-mqtt
```

5.2.3 Conexión de los dispositivos

La instalación física de los componentes que conforman el sistema requiere de la interconexión de todos los elementos entre sí, incluso con los dispositivos a automatizar ya existentes. La conexión entre Raspberry Pi, la cámara IP y el router, se realizará con cableado de red UTP y conectores RJ45. El resto de conexiones se deben hacer con cable eléctrico de 0.5mm, excepto la alimentación del proyector LED que se realizará con la misma sección de cable con la que se encontrará instalado.

El cableado y conexión de los dispositivos se realizará según el esquema siguiente.

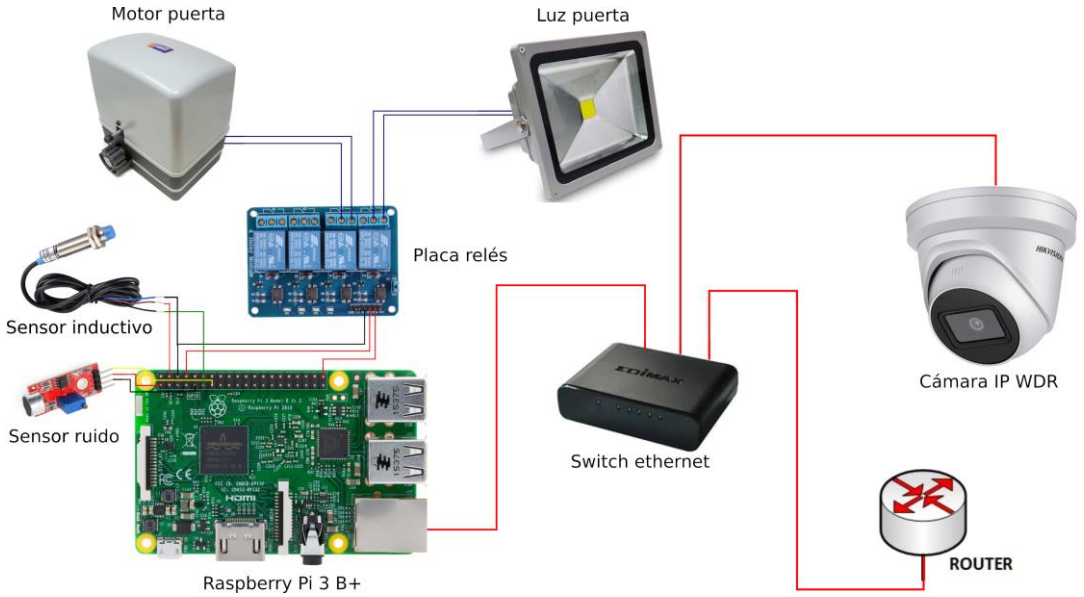

Figura 28 Esquema de conexión de dispositivos

El sensor de ruido deberá estar ubicado en un lugar protegido del viento y la lluvia, pero cercano a la parte exterior de la puerta y en ningún caso debe encontrarse encerrado en un recinto que evite la entrada del ruido exterior.

El sensor inductivo se instalará en algún lugar de la puerta que permita la detección del metal a menos de 4mm cuando esté cerrada y pierda el contacto cuando la puerta empiece a abrir.

La cámara deberá situarse en uno de los laterales de la puerta, enfocando al centro del paso de los vehículos.

El resto de componentes deberán ser instalados en un lugar protegido y aislado, como puede ser un armario de fibra, una caja estanca o cualquier otro registro.

5.2.4 Desarrollo del software de automatización

Empezaremos por el desarrollo del script que automatizará la apertura de la puerta del garaje mediante la lectura de placas de matrícula, al que llamaremos lectura-placas.py y será el encargado de realizar capturas desde la cámara, pasarlas a openalpr, filtrar las matrículas autorizadas, monitorizar el sensor de ruido y dar la orden de apertura al motor.

En primer lugar, abriremos el archivo lectura-placas.py con el editor de textos nano con el siguiente comando.

```
nano lectura-placas.py
```

Cada vez que queramos guardar el archivo utilizaremos el atajo ctrl+o y para salir de la edición del archivo ctrl+x.

Importaremos todas las librerías de las que vamos a hacer uso durante el desarrollo del script.

```
from openalpr import Alpr
from hikvisionapi import Client
import time
import logging
import io
import RPi.GPIO as GPIO
```

Inicializamos el log para que guarde por defecto en el fichero door.log y escribimos en el log el inicio del script.

```
logging.basicConfig(filename='/home/pi/door.log', format='%(asctime)s -
%(levelname)s - %(message)s', datefmt='%d-%b-%y %H:%M:%S',
level=logging.INFO)
logging.info('inicio script')
```

A continuación, vamos a declarar las variables en las que indicaremos el número de GPIO de la Rasberry Pi al que está conectado cada dispositivo.

```
pin_light = 21
pin_door = 14
pin_noise = 27
```

Una vez identificados los pines mediante variables, deberemos indicar que uso vamos a hacer de dichos pines, entrada o salida y fijar su estado inicial para los de salida.

```
#definición de pins
GPIO.setmode(GPIO.BCM)
GPIO.setup(pin_light, GPIO.OUT) #luz
GPIO.setup(pin_door, GPIO.OUT) #relé motor puerta
GPIO.setup(pin_noise, GPIO.IN) #sensor de ruido

#iniciliazación de pins
GPIO.output(pin_light, GPIO.HIGH) #luz apagada
GPIO.output(pin_door, GPIO.HIGH) #relé motor puerta apagado
```

En este punto nos queda definir una variable para guardar el objeto Alpr, el cual nos da acceso a la llamada de funciones de la librería openalpr, otra variable que utilizaremos a continuación para crea el objeto que conecta con la cámara y un array con las placas de matrícula autorizadas.

```
alpr = Alpr("eu", "/etc/openalpr/openalpr.conf", "/usr/share/openalpr/runtime_data")
plates = ['1234BCD','8888DDD','0000HNR']
cam = None
```

Tenemos todo lo necesario para iniciar la conexión con el stream de la cámara utilizando la clase Client de la librería hikvisionapi. Puesto que no podremos continuar ejecutando el programa sin acceso a la cámara, se ha creado un bucle infinito que intentará conectar con la cámara cada 5 segundos, en caso de que esta no esté disponible en algún momento concreto. Cuando la conexión a la cámara se realice correctamente, la ejecución del programa continuará saliendo del bucle.

```
while True: #bucle infinito hasta que conecte la cámara
    try:
        cam = Client('http://10.0.0.100', 'usuario', 'password')
        if cam is not None:
            logging.info('conexión a la cámara establecida')
            break
    except:
        logging.error('error conectando a la cámara')
        time.sleep(5)
```

Con la conexión a la cámara establecida, ya podemos empezar nuestra secuencia de programa que se repetirá de manera indefinida y la cual dará vida a la automatización de la puerta mediante la lectura de la placa de matrícula.

Crearemos un bucle infinito, pues vamos a estar realizando la misma comprobación de manera indefinida, la cual consiste en realizar una captura de la cámara mediante la función picture y buscar una placa de matrícula en la misma mediante la función recognize_array, que nos devolverá un array de strings con las matrículas leídas.

```
while True: #ejecución permanente
    try: #control de excepción por si perdemos conexión con la cámara
        response = cam.Streaming.channels[101].picture(method='get',
type='opaque_data') #nos devuelve una imagen de la cámara
    except:
        logging.error('camera error')
        time.sleep(5)
        continue

    img = io.BytesIO(response.content)
```

```
analysis = alpr.recognize_array(img.getvalue()) #array con matrículas
```

En caso de no detectar ninguna matrícula, el programa no ejecutará ningún código adicional y volverá de nuevo a iniciar el bucle de captura y búsqueda. Si la función recognize_array nos ha devuelto un array que no esté vacío, quiere decir que ha encontrado algo en la imagen, por lo que montaremos una lista con las matrículas detectadas para trabajar más cómodamente.

```
if len(analysis['results']) > 0: #placa de matrícula detectada
        detected_plate = list()
        for n in range(len(analysis['results'])):
            detected_plate.append(analysis['results'][n]['plate'].replace(" ",""))
#llenamos una lista con las matrículas encontradas
        logging.info('placa de matrícula detectada '+str(detected_plate))
```

A pesar de detectar matrículas en la imagen procesada, no tiene por qué continuar el flujo de apertura de puerta ya que es posible que sean matrículas no autorizadas, de tal forma que solo quedarán registradas en el log y el proceso de captura volverá a empezar. Solo en el momento que alguna de las placas detectadas coincida con alguna de las placas autorizadas, el script mandará la orden de encender la luz para indicar al usuario la correcta identificación de su placa autorizada, posteriormente quedará esperando 10 segundos a detectar una señal positiva que provenga del sensor de ruido, para ello usamos la función wait_for_edge, la cual espera un cambio de estado en el pin indicado. Si es recibida esa señal del sensor de ruido, activaremos la salida del relé del motor de puerta durante un segundo para que entregue un impulso de apertura a la electrónica del motor, entonces esperaremos 75 segundos y apagaremos la luz, pues el cierre de la puerta es automático mediante la propia electrónica del motor.

```
if(len(set(plates).intersection(detected_plate)) == 1): #encontrada matrícula autorizada
            logging.info('allowed plate')
            GPIO.output(pin_light, GPIO.LOW)
            logging.info('light power on')
            if(GPIO.wait_for_edge(pin_noise, GPIO.FALLING, timeout=10000)):
                logging.info('noise detected')
                time.sleep(0.1)
                if(GPIO.wait_for_edge(pin_noise, GPIO.FALLING, timeout=100)):
                    logging.info('start door open cycle')
                    GPIO.output(pin_door, GPIO.LOW)
                    time.sleep(1)
                    GPIO.output(pin_door, GPIO.HIGH)
                    time.sleep(75)
                    logging.info('end door open cycle')
            GPIO.output(pin_light, GPIO.HIGH)
            time.sleep(1)
            logging.info('light power off')
```

El sensor de ruido devolverá una señal cuando detecte un nivel de ruido superior al calibrado mediante su potenciómetro integrado, girando a derecha e izquierda este potenciómetro podemos subir o bajar la sensibilidad con la cual debe activar la salida. Teniendo en cuenta que queremos detectar el sonido del claxon, deberíamos bajar al máximo la sensibilidad e ir incrementando en pasos de ¼ de vuelta hasta que veamos que se activa la señal al sonar el claxon.

Para poder ejecutar el script a la vez que vamos haciendo cambios sobre él, podemos utilizar el comando:

python3 lectura-placas.py

Quedará bloqueada la consola con la ejecución del programa, pero podremos terminar la ejecución pulsando las teclas ctrl+c.

Llegado el final del desarrollo del primer script y habiendo realizado las pruebas de funcionamiento oportunas, podemos añadir la ejecución automática del script en el archivo del sistema */etc/rc.local* antes de la línea exit 0.

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi

/usr/bin/python3 /home/pi/openalpr.py

exit 0
```

Figura 29 Archivo /etc/rc.local editado

El segundo script que desarrollaremos, se encargará de realizar la comunicación con Home Assistant mediante el protocolo MQTT. La comunicación será usada para recibir órdenes que alterarán el estado de los pines y devolver el estado de los mismos a Home Assistant, de esta forma podremos controlar el mismo dispositivo desde diferentes fuentes y todas estarán sincronizadas entre sí. La notificación de estados es necesaria para avisar a Home Assistant de que la puerta ha sido abierta desde el programa de lectura de matrículas, ya que este es ajeno y totalmente autónomo.

Como hemos hecho anteriormente, lo primero que realizaremos es la importación de las librerías de las que vamos a hacer uso.

```
import paho.mqtt.client as mqtt
import RPi.GPIO as GPIO
import json
import time
```

Para minimizar errores de escritura y optimizar el código, inicializaremos un objeto tipo diccionario para establecer la relación entre el topic de control, con el que recibimos datos, el topic de estado, con el que notificamos cambio de estado y el número de pin de la Raspberry Pi. Aprovechamos para definir una variable con el número de pin del sensor inductivo.

```
devices = {
    'campo/puerta/luz/control' : {'publish':'campo/puerta/luz/estado', 'gpio':21},
    'campo/puerta/motor/control': {'publish':'campo/puerta/motor/estado', 'gpio':14},
}
pin_inductive = 17
```

El siguiente paso es definir e inicializar los GPIO, para ello recorremos el diccionario de topics definiendo cada GPIO como salida y por otro lado definimos el GPIO pin_inductive como entrada ya que lo tenemos fuera de los topics debido a que no hay un control directo desde Home Assistant y por tanto desde un topic.

```
GPIO.setmode(GPIO.BCM)
for topic in devices:
    GPIO.setup(devices[topic]['gpio'], GPIO.OUT)
GPIO.output(14, 1) #output motor off, not recovery from retain
GPIO.setup(pin_inductive, GPIO.IN)
```

A pesar de no haber un control sobre el pin del inductivo por parte de Home Assistant, sí que queremos que este sepa el estado en el que se encuentra el sensor, por ello crearemos una interrupción para que cada vez que el sensor inductivo cambie de estado, llame a la función inductive_event.

```
GPIO.add_event_detect(pin_inductive, GPIO.BOTH, callback=inductive_event,
bouncetime=100)
```

En la función inductive_event, que será llamada cada vez que haya un cambio de estado en el inductivo, lo que haremos será publicar el estado del correspondiente topic al broker MQTT.

```
def inductive_event(pin):
    if(GPIO.input(pin_inductive)):
        client.publish('campo/puerta/motor/estado', 0)
    else:
        client.publish('campo/puerta/motor/estado', 1)
```

Mediante la clase Client de la librería paho.mqtt realizaremos la conexión al broker MQTT ejecutado en Home Assistant. La conexión requiere la dirección IP del servidor, usuario y contraseña, además debemos pasar dos funciones callback a la clase Client que se ejecutarán cuando se establezca la conexión con el broker MQTT y cuando se reciba un mensaje. Como requisito de la librería, llamaremos a la función loop_forever que será la que se encargue de gestionar el envío y recepción de mensajes en segundo plano, llamando a las funciones callback.

```
client = mqtt.Client()
client.on_connect = on_connect #función callback llamada al conectar
client.on_message = on_message #función callback llamada al recibir mensaje
client.username_pw_set(username="usuario", password="password")
client.connect("10.0.0.89", 1883, 60) #ip, puerto y timeout
client.loop_forever()
```

La función on_connect realizará la suscripción a todos los topics MQTT que vayamos a utilizar mediante la función client.subscribe, tras esto recibiremos los mensajes de estos topics cuando sean emitidos por el broker MQTT.

```
def on_connect(client, userdata, rc, properties=None):
    print('Connected with result code ' + str(rc))
    for topic in devices:
        client.subscribe(topic)
        print('subscribe: '+str(topic))
```

La función on_message recibirá el topic junto con el payload de cada mensaje enviado por el broker. El payload será un valor 0 o 1 correspondiente al estado apagado o encendido respectivamente. Mediante el array devices encontraremos el GPIO vinculado a cada topic y cambiaremos el estado de ese GPIO al estado que venga en el payload del mensaje, consiguiendo un intérprete entre los mensajes MQTT y los pines de la Raspberry Pi.

Hay una excepción en el comportamiento para el topic correspondiente al motor de la puerta, esto es porque en lugar de un comportamiento encendido y apagado tipo interruptor, necesitamos un comportamiento de tipo pulso o pulsador, cambiando el estado durante tan solo 2 segundos para dar al orden de apertura al motor, volviendo al estado inicial de nuevo.

Tras el tratamiento del mensaje y la ejecución de las órdenes correspondientes, volveremos a publicar el estado del topic al broker para que siempre se encuentre sincronizado, de lo contrario Home Assistant daría por hecho que la luz ha sido encendida sin tener comprobación de que haya sido así.

```
def on_message(client, userdata, msg):
    if msg.topic == 'campo/puerta/motor/control' and int(msg.payload) == 0: #caso especial para un topic
        if msg.retain == 0: #comportamiento tipo pulso durante 2 segundos
            GPIO.output(devices[msg.topic]['gpio'], 0)
            time.sleep(2)
            GPIO.output(devices[msg.topic]['gpio'], 1)
    else: #comportamiento general de los topics
        GPIO.output(devices[msg.topic]['gpio'], int(msg.payload)) #cambio estado GPIO
        client.publish(devices[msg.topic]['publish'], msg.payload) #publicación del nuevo estado al broker MQTT
```

Una vez hemos terminado el desarrollo del script y tras realizar las oportunas pruebas mediante la ejecución manual del programa, con el siguiente comando.

python3 mqtt.py

Debemos programar la ejecución automática del mismo, para ello crearemos un servicio en el sistema con systemd creando el archivo **/etc/systemd/system/mqtt.service** con el siguiente código.

```
[Unit]
Description=MQTT Service
Wants=network.target
After=network.target

[Service]
Type=simple
User=root
WorkingDirectory=/home/pi
ExecStart=/usr/bin/python3 /home/pi/mqtt.py
Restart=on-failure
RestartSec=5s

[Install]
WantedBy=multi-user.target
```

Figura 30 Código del servicio MQTT

5.3 Desarrollo sistema de automatización vivienda

Durante el desarrollo del sistema de automatización de la vivienda, utilizaremos los siguientes elementos:

- Sonoff Mini R2
- Sonoff Basic R2
- Emisor IR TSAL6400
- Transistor 2N3904
- Cámara IP Hikvision
- Central alarma Risco

El objetivo principal de este sistema, será permitir el control de la iluminación de la vivienda, un aparato de aire acondicionado convencional que no dispone de conexión a la red y el control del sistema de seguridad, el cual abarca desde las cámaras CCTV hasta la central de alarma Risco.

5.3.1 Personalización de Sonoff basic R2 con emisor IR

El Sonoff basic R2 es un relé con conexión WiFi diseñado para ser controlado en remoto mediante la app oficial de Sonoff, sin embargo, existe la opción de poder instalarle el firmware de código abierto Tasmota lo que abre las puertas a hacer un uso amplio y diferente aprovechando la electrónica del dispositivo.

En este caso, queremos utilizar el dispositivo como un mando de infrarrojos controlado mediante MQTT a través de su conexión WiFi. Esta es la única manera viable de poder controlar un aire acondicionado o una televisión que todavía no tiene conexión de red, reproduciendo los códigos que son enviados por su control remoto o mando a distancia.

Para convertir el relé remoto en un emisor infrarrojo con control remoto tendremos que modificar tanto el software, ya que queremos controlarlo por MQTT, como la electrónica ya que necesitamos un emisor IR.

Por facilidad, actualizaremos primero el firmware y a continuación conectamos el emisor IR a través de un transistor.

Antes de empezar debemos descargar la última versión del firmware tasmota-ir, que es la compilación que más funcionalidades tiene para trabajar con emisión y recepción de infrarrojos. También descargaremos la última versión del programa ESP Easy Flasher, con el que grabaremos el firmware.

La actualización del firmware de este dispositivo solo es posible a través de su puerto UART físico, por lo que necesitaremos un soldador y un conversor USB a UART TTL que soldaremos a la placa del Sonoff respetando el siguiente pinout.

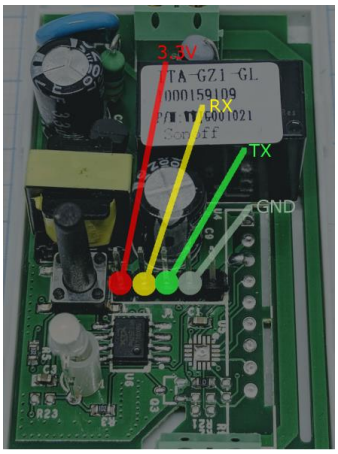

Figura 31 pinout puerto UART Sonoff basic R2

Una vez soldado el conversor USB UART TTL a la placa del Sonoff, podemos conectar el USB al ordenador mientras mantenemos presionado el pulsador del Sonoff, lo que forzará que entre en modo programación.

Abrimos el software de programación ESP Easy Flasher[21] y seleccionamos el puerto COM que haya sido asignado en el sistema al conversor USB UART TTL, ahora cargamos el fichero del firmware tasmota-ir.bin y pincharemos en el botón Flash ESP easy FW para que comience el proceso de grabado en el microcontrolador.

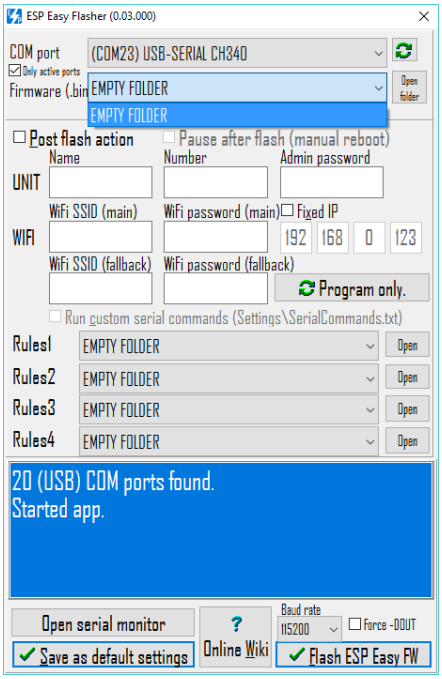

Figura 32 captura interfaz ESP Easy Flasher

21 https://github.com/Grovkillen/ESP_Easy_Flasher

Cuando termine el proceso podemos retirar las soldaduras y ya tendremos el firmware tasmota-ir cargado en el Sonoff.

Aprovechando que tenemos a mano el soldador y la placa desmontada, soldaremos el transistor y el emisor IR como se indica en el siguiente esquema.

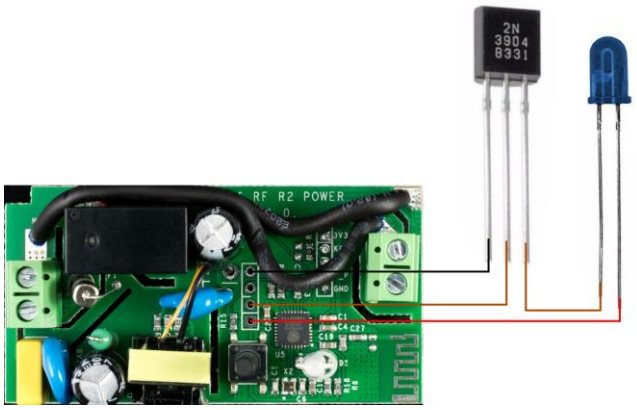

Figura 33 Esquema conexión IR a Sonoff

En este punto ya podemos volver a montar la carcasa del Sonoff, teniendo en cuenta que el emisor IR debe quedar visible, para ello podemos hacer un orificio de 5mm en la propia carcasa o extender el emisor con un cable que salga hasta la distancia deseada de la carcasa.

Con el equipo personalizado a nivel de software y hardware, procederemos a la configuración que le permitirá actuar como mando a distancia controlado por MQTT, todas las configuraciones las realizaremos en la interfaz de tasmota.

En el primer arranque del firmware tasmota, este configura el radio WiFi en modo punto de acceso para permitirnos conectarnos a él, esto es debido a todavía carece de configuración que le permita conectar a nuestra red WiFi existente.

Tendremos que buscar en nuestro ordenador una red WiFi cuyo nombre empieza por tasmota_ seguido de un código alfanumérico. Conectaremos a ella sin necesidad de contraseña y accederemos a la IP de tasmota 192.168.4.1 a través del navegador.

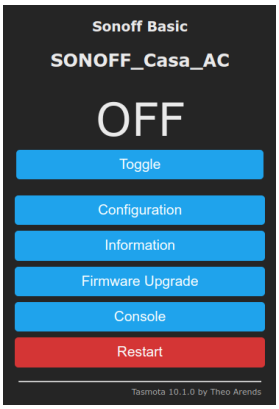

Figura 34 Captura interfaz principal de tasmota.

Entraremos en el menú configuración y en el menú WiFi, donde configuraremos el nombre y contraseña de acceso a nuestra red WiFi, teniendo también la opción de realizar un escaneado de redes que nos permitirá pinchar en cualquiera de las redes detectadas.

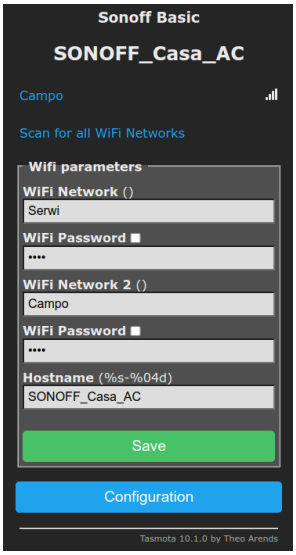

Figura 35 Captura interfaz configuración WiFi de tasmota.

Habiendo conectado el Sonoff a nuestra red WiFi, ya podemos conectarnos a su interfaz web con la IP de nuestra red local, la cual podemos encontrar con el comando Nmap como hemos visto en puntos anteriores.

Teniendo el dispositivo conectado en la red local junto con el resto de dispositivos, vamos a configurar la conexión del Sonoff a Home Assistant mediante el broker MQTT. Tendremos que especificar la dirección IP del servidor MQTT, el usuario, la contraseña y el topic con el que se identificará este dispositivo.

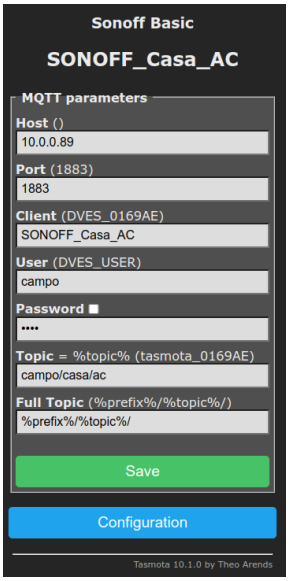

Figura 36 Captura interfaz configuración MQTT en tasmota.

En este momento, podemos controlar el Sonoff desde Home Assistant, pero lo que haríamos sería activar y desactivar el relé que trae por defecto ya que no le hemos dicho que ahora también tiene un emisor infrarrojo, para activarlo iremos al menú module dentro de la configuración de tasmota y le indicaremos que en su pin GPIO3 tiene un emisor IR.

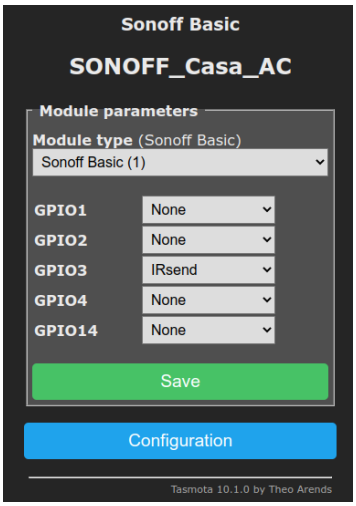

Figura 37 Captura interfaz configuración module en tasmota.

Ha quedado totalmente configurado el firmware tasmota para poder recibir mensajes MQTT desde Home Assistant a través de la red WiFi y automáticamente los reenviará a través del emisor IR, el cual estará apuntando a la máquina de aire acondicionado.

5.3.2 Configuración Sonoff mini R2 con tasmota

Contamos con el dispositivo Sonoff mini R2, el cual ya viene diseñado para poder encender y apagar una luz o un electrodoméstico de forma sencilla desde la app o desde un interruptor manual.

El problema que tenemos es que este dispositivo se maneja a través del cloud de Sonoff, lo que nos hace dependientes de la marca además de una conexión a internet constante. Para solucionar estos inconvenientes, vamos a grabar el firmware tasmota en el dispositivo, lo que permitirá conectarlo a nuestra red WiFi y trabajar de manera directa con Home Assistant, sin dependencia de servidores de terceros o de conexión a internet.

Para poder grabar el firmware tasmota en el microcontrolador del Sonoff mini R2, debemos soldar el conversor USB UART TTL a los pines VCC, GND, TX y RX según la siguiente imagen.

Figura 38 Pinout Sonoff mini R2

Con el conversor soldado, procederemos a grabar el firmware siguiendo el mismo método que el empleado para el Sonoff basic R2 en el punto anterior, pero en este caso grabaremos la compilación general de tamota, el fichero tasmota.bin, esto es debido a que el uso que le vamos a dar a este dispositivo es el de control de luces mediante la conmutación del relé, por lo que no requerimos ninguna compilación con funcionalidades especiales.

Tras terminar el grabado del firmware, retirar las soldaduras y volver a montar el Sonoff, lo alimentaremos y realizaremos la configuración de tasmota desde su interfaz web, para ello configuraremos la conexión a nuestra red WiFi y al servidor MQTT igual que lo hemos hecho en el punto anterior.

Por último, conectaremos el Sonoff al circuito de iluminación que vayamos a controlar, así como al interruptor que lo controle actualmente, siguiendo este esquema.

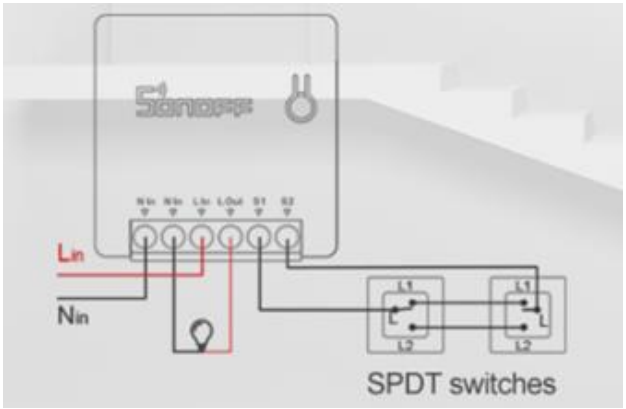

Figura 39 Esquema de conexión Sonoff mini R2.

5.3.3 Integración de dispositivos en Home Assistant

Contamos con dispositivos instalados en la vivienda que han sido configurados, pero todavía no son controlables desde Home Assistant, por lo que vamos a realizar las configuraciones necesarias para que Home Assistant reconozca todos los dispositivos y así podremos hacer uso de ellos de forma manual o bien en cualquier automatización programada.

Dentro de la vivienda contamos con dispositivos Sonoff mini R2 utilizados para encender y apagar luces o electrodomésticos, los cuales funcionan con el firmware tasmota, también tenemos el mando IR que controla el aire acondicionado que también funciona con tasmota.

Por otro lado, contamos con el sistema de seguridad, compuesto por una central de alarma Risco y tres cámaras Hikvision.

De lo descrito anteriormente asumimos que debemos de hacer tres integraciones, una para tasmota, una para la central Risco y otra para las cámaras Hikvision, independientemente de cuantos dispositivos haya de cada.

Empezaremos instalando la integración de tasmota que viene incluida en Home Assistant por defecto. Para ello vamos a configuración, integraciones, añadir integración y pinchamos sobre tasmota.

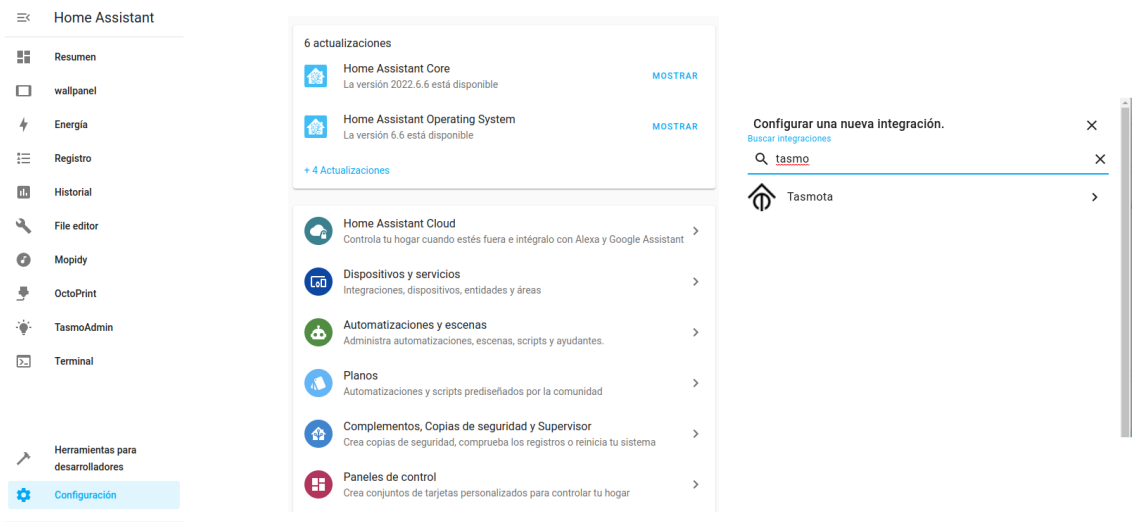

Figura 40 Capturas instalación integración tasmota en Home Assistant.

Tras instalar la integración, detectará automáticamente los dispositivos tasmota que hemos conectado anteriormente al broker MQTT de Home Assistant, por cada dispositivo nos creará diferentes entidades con las que podremos controlar el encendido o apagado, así como otros valores del propio Sonoff, nivel de señal wifi, tiempo desde último reinicio, etc.

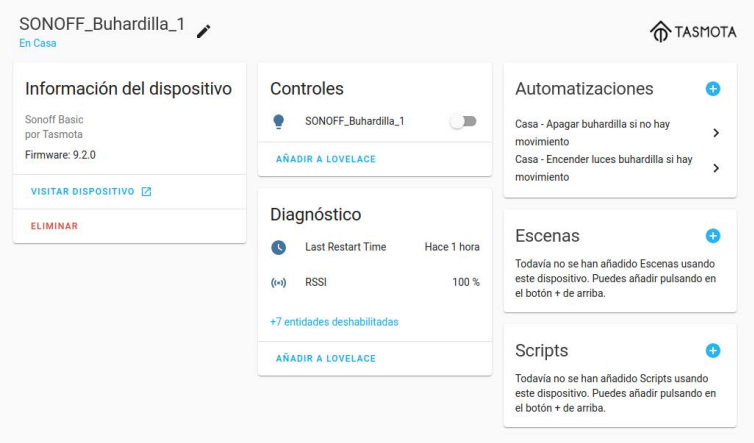

Figura 41 Captura dispositivo auto detectado por la integración tasmota.

Puesto que a través de IR podemos controlar una gran variedad de dispositivos, debemos decirle a Home Assistant que en este caso el aparato a controlar es un aire acondicionado, de tal manera que nos permita mostrar un control en pantalla acorde con las funcionalidades de un equipo de climatización.

En primer lugar, tendremos que instalar el componente tasmota_irhvac, el cual posibilita crear una entidad en Home Assistant de tipo climatización que comunica con el controlador a través de MQTT con el firmware tasmota-ir. Para instalar este componente solo debemos descargarlo, copiarlo en el directorio custom_components y después reiniciar Home Assistant.

Para añadir la entidad de tipo climate editaremos el fichero configuration.yaml añadiendo el siguiente código.

```
climate:
  - platform: tasmota_irhvac
    name: "A/C Casa"
    command_topic: "cmnd/campo/casa/ac/irhvac"
    state_topic: "stat/campo/casa/ac/RESULT"
    availability_topic: "tele/campo/casa/ac/LWT"
    temperature_sensor: sensor.testigo_ac_y_temperatura_hdc1080_temperature
    vendor: "HAIER_AC"
    hvac_model: "-1"
    min_temp: 16
    max_temp: 30
    target_temp: 24
    initial_operation_mode: "off"
    away_temp: 24
    precision: 1
    supported_modes:
```

```
  - "auto"
  - "cool"
  - "dry"
  - "fan_only"
  - "heat"
  - "off"
supported_fan_speeds:
  - "min"
  - "medium"
  - "max"
  - "auto"
supported_swing_list:
  - "off"
  - "vertical"
default_quiet_mode: "Off"
default_turbo_mode: "Off"
default_econo_mode: "Off"
celsius_mode: "On"
default_light_mode: "Off"
default_filter_mode: "Off"
default_clean_mode: "Off"
default_beep_mode: "Off"
default_sleep_mode: "-1"
```

Los parámetros que vinculan la entidad con el dispositivo tasmota son command_topic, state_topic y availability_topic que debemos establecerlos según el nombre de topic que hayamos configurado en el firmware tasmota del Sonoff.

Otro parámetro importante es el vendor, que establece el protocolo de control remoto IR que utiliza nuestro aparato de aire acondicionado.

Por último, podemos añadir una tarjeta de control al panel Lovelace que nos permita controlar desde la pantalla el aire acondicionado. Para ello iremos al panel donde queramos mostrar el control y pincharemos en el botón añadir tarjeta, seleccionaremos la tarjeta termostato e indicaremos el id de la entidad climate creada anteriormente.

Figura 42 Añadir tarjeta termostato en Home Assistant.

Para realizar la integración con las cámaras, haremos uso de otra integración de Home Assistant llamada ONVIF, nombre del protocolo estándar utilizado por las cámaras en el sector de CCTV. Elegimos esta opción porque no existe una integración directamente con la marca Hikvision y porque al ser más generalista, con un protocolo estándar, nos ofrecerá mayor versatilidad en un futuro si queremos integrar cámaras de diferentes fabricantes.

A diferencia de la integración de tasmota, la de ONVIF no permite la detección de dispositivos en tiempo de ejecución, por lo que tenemos que realizar el proceso de añadir integración por cada dispositivo que tengamos, aunque sí que nos permite realizar una búsqueda de las cámaras ONVIF detectadas en la red.

Tanto si añadimos la cámara mediante la búsqueda automática, como si indicamos la IP y puerto de forma manual, tendremos que indicar el usuario y contraseña de acceso a la cámara. Además, debemos asegurarnos que hemos activado el protocolo ONVIF en todas las cámaras ya que no siempre viene activado por defecto.

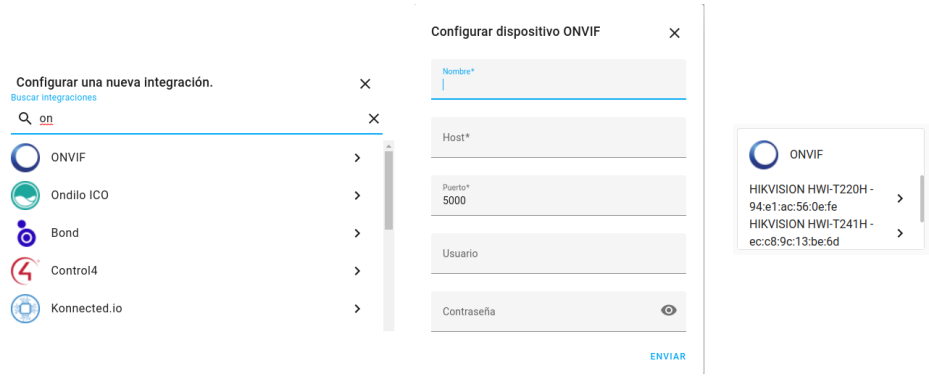

Figura 43 Captura del proceso de integración ONVIF.

Una vez añadidas tantas integraciones como cámaras queramos conectar, veremos los dispositivos creados en Home Assistant con todas sus entidades, entre ellas el stream de video o la hora el último reinicio.

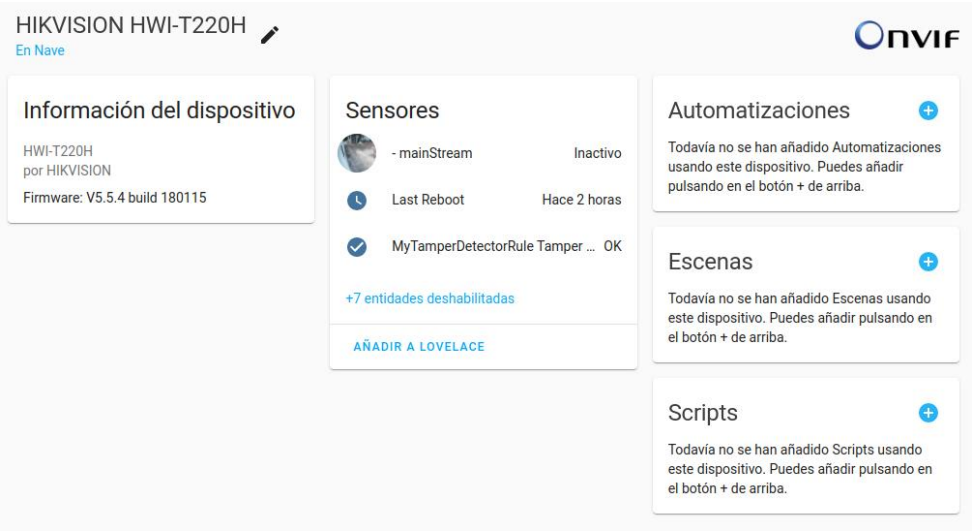

Figura 44 Captura del dispositivo creado por la integración ONVIF.

Por último, nos quedará por integrar el panel de alarma Risco. En este caso, aunque Home Assistant cuenta con una integración también para la marca Risco, esta solo es compatible con las centrales de gama alta que permiten la gestión de particiones, mientras que nuestra central es una Risco iConnect que no tiene gestión de particiones y por tanto no es compatible.

Analizando el código de la integración, disponible en github, observamos que la incompatibilidad no proviene de la propia integración, si no de la librería de python pyrisco de la que hace uso la integración, es ella la que no prevé el uso de paneles sin particiones.

Llegados a este punto, en el que no existe una integración válida disponible para el equipo que deseamos conectar, tenemos la opción de desarrollar una integración y subirla al sistema de manera manual.

Realizando una búsqueda en github hemos encontrado un pull request del paquete pyrisco que introduce la compatibilidad con centrales sin particiones, sin embargo, las dependencias de las integraciones que instalamos en Home Assistant son descargadas mediante el gestor de paquetes pip, por lo que no podemos forzar a la integración a que utilice la versión de pyrisco del pull request.

Para solucionar este problema sin tener que hacer un desarrollo completo de una integración, vamos a descargar la integración y modificar sus requerimientos para que en este caso fuerce el uso de la librería descarga del pull request de github y no desde el gestor de paquetes pip.

En primer lugar, debemos descargar la carpeta con la integración desde el repositorio oficial de Home Assistant en github, la encontraremos dentro de components con el nombre de risco.

Tras la descarga, editaremos el fichero manifest.json y cambiaremos la dependencia pyrisco==0.3.1 por el pull request de github, quedando de la siguiente forma.


```
{
  "domain": "risco",
  "name": "Risco",
  "config_flow": true,
  "documentation": "https://www.home-assistant.io/integrations/risco",
  "requirements": ["git+https://github.com/SimonThoustrup/pyrisco.git@partitionless#pyrisco==0.3.2"],
  "codeowners": ["@OnFreund"],
  "quality_scale": "platinum",
  "iot_class": "cloud_polling",
  "loggers": ["pyrisco"]
}
```


Figura 45 Archivo manifest.json de la integración risco modificado.

Con la integración preparada, solo resta subirla al directorio custom_components de Home Assistant mediante el editor de archivos del mismo y proceder a la instalación de la integración Risco de la misma forma que hemos instalado el resto. De forma automática instalará desde custom_components y no desde la oficial, por una cuestión de prioridades.

Al añadir la integración, nos solicitará el nombre de usuario y contraseña del cloud de Risco, además del pin de armado y desarmado del panel. Tras aceptar, nos creará un dispositivo para la central y uno por cada sensor inalámbrico con el que contemos. En las entidades de la central podremos armar y desarmar la misma, en las entidades de los sensores podremos ver el estado en el que se encuentra el sensor; abierto, cerrado, presencia, etc.

Figura 46 Captura tarjeta resumen de la integración Risco.

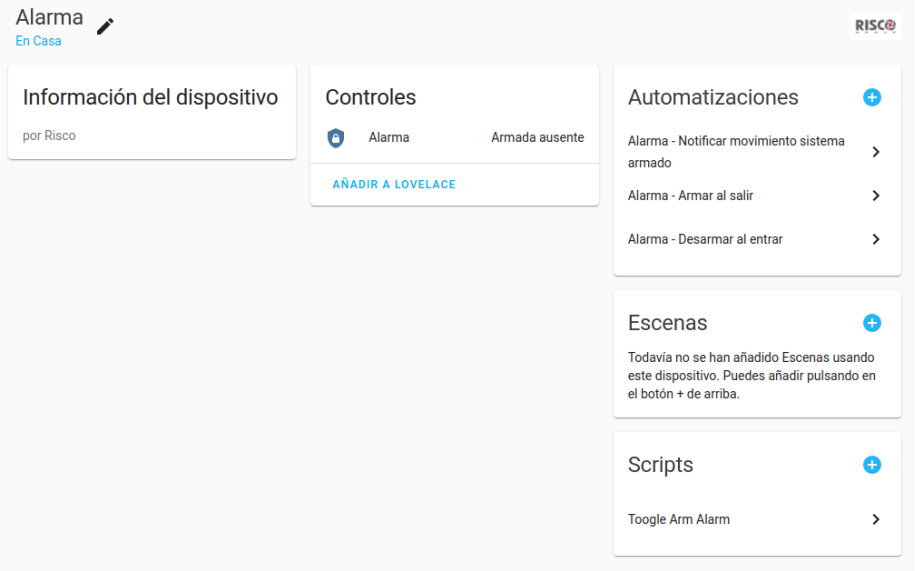

Figura 47 Captura dispositivo central creado por la integración Risco.

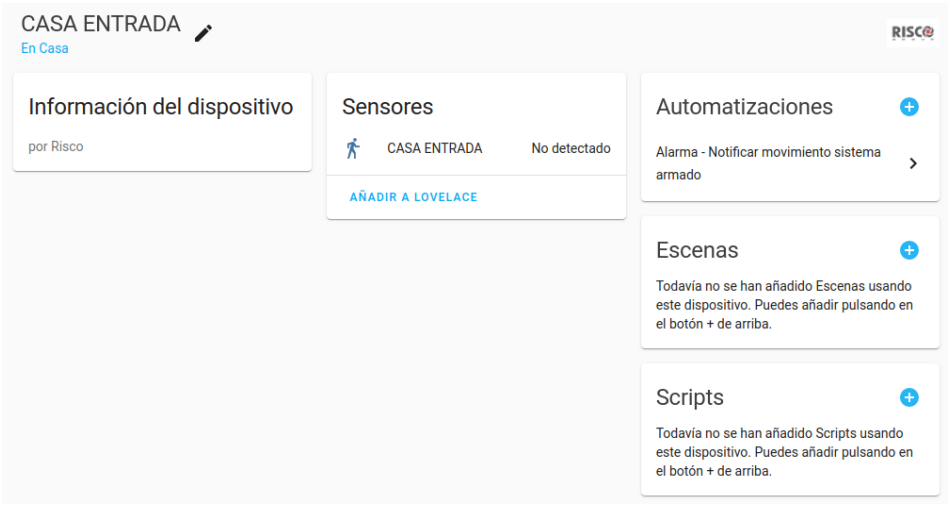

Figura 48 Captura dispositivo sensor creado por la integración Risco.

5.4 Integración sistema automatización piscina

En este punto vamos a integrar, dentro de nuestro sistema Home Assistant, la automatización de la piscina realizada por el compañero David Amores Dolz en su trabajo de fin de grado conjunto.

Para poder añadir todos los dispositivos que permiten el control de la piscina como entidades en Home Assistant, deberemos añadir las entidades de tipo interruptor (encendido y apagado) y las de tipo luz RGB.

Añadiremos el siguiente código en el archivo configuration.yaml por cada entidad de tipo interruptor, cambiando el unique_id, el name y los dos topic según los establecidos en el software del dispositivo de control, lo que puede consultarse en el proyecto de David Amores Dolz.

```
switch:
  - platform: mqtt
    unique_id: piscina_agua
    name: "Bomba de agua"
    command_topic: "campo/piscina/agua/control"
    state_topic: "campo/piscina/agua/estado"
    payload_on: "1"
    payload_off: "0"
    retain: true
```

Para añadir la luz RGB, se añadirán las siguientes líneas en el mismo archivo de configuración, modificando el unique_id, name y los topic según los establecidos en el software de la Raspberry Pi que controla la piscina.

```
light:
  - platform: mqtt
    unique_id: piscina_rgb
    name: "RGB Piscina"
    state_topic: "campo/piscina/rgb/estado"
    command_topic: "campo/piscina/rgb/control"
    brightness_state_topic: "campo/piscina/rgb/brillo/estado"
    brightness_command_topic: "campo/piscina/rgb/brillo/control"
    rgb_state_topic: "campo/piscina/rgb/color/estado"
    rgb_command_topic: "campo/piscina/rgb/color/control"
    brightness_scale: 100
    payload_on: "1"
    payload_off: "0"
    retain: true
```

Faltará por añadir las siguientes líneas para la entidad correspondiente al sensor de nivel de agua de la piscina, que en este caso será de tipo binary_sensor, el cual tiene dos estados y es de solo lectura. Será necesario modificar el state_topic al establecido en el software de la Raspberry Pi.

```
binary_sensor:
  - platform: mqtt
    name: "Nivel Piscina"
    state_topic: "campo/piscina/nivel/estado"
    payload_on: "1"
    payload_off: "0"
```

Por último, solo quedará reiniciar Home Assistant para disponer de todas las entidades en el sistema, pudiendo añadir el control de las mismas al panel Lovelace mediante las correspondientes tarjetas.

Hemos elegido una tarjeta de entidades que nos ofrece la funcionalidad de encendido y apagado de cada entidad y una tarjeta de tipo rgb-light-card para los colores de la luz RGB. La instalación de la tarjeta personalizada rgb-light-card la hemos realizado mediante el gestor de componentes HACS siguiendo las instrucciones del desarrollador que podemos encontrar en la bibliografía.

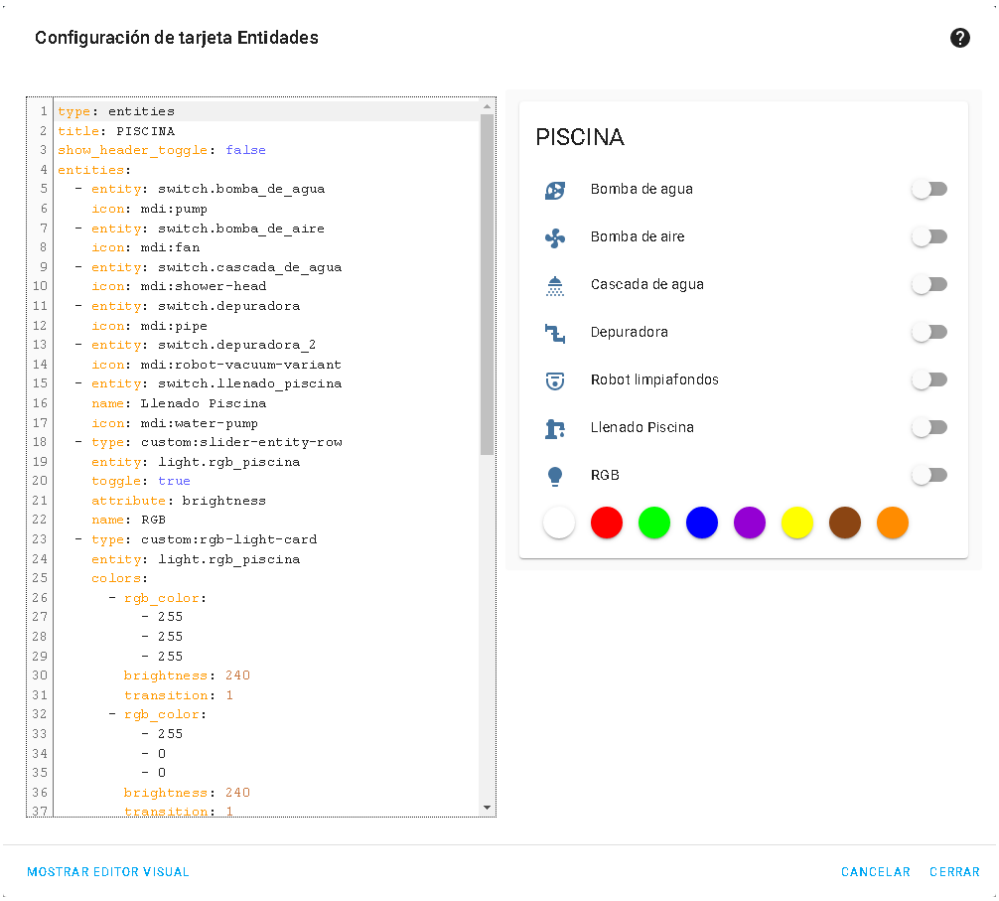

Figura 49 Configuración tarjeta control piscina.

5.5 Integración sistema automatización techo abatible

En este punto vamos a integrar, dentro de nuestro sistema Home Assistant, la automatización del techo abatible y sistema multimedia realizada por el compañero David Amores Dolz en su trabajo de fin de grado conjunto.

De la misma forma que hemos añadido las entidades en el punto anterior, deberemos añadir cada entidad para la integración de la Raspberry Pi que controla el techo abatible y el sistema multimedia.

Definiremos una entidad de tipo switch para la apertura y cierre del techo añadiendo las siguientes líneas, en las que cambiaremos los topic según los establecidos en el software de la Raspberry Pi.

```
switch:
  - platform: mqtt
    unique_id: chillout_techo
    name: "Techo chillout"
    command_topic: "campo/chillout/techo/control"
    state_topic: "campo/chillout/techo/estado"
    payload_on: "1"
    payload_off: "0"
    optimistic: true
    retain: true
```

Para el control de la luz RGB, definiremos una entidad de tipo light con la parametrización para el control de los colores, añadiendo las siguientes líneas, en las que editaremos los topic MQTT según los establecidos en la Raspberry Pi.

```
light:
  - platform: mqtt
    unique_id: chillout_rgb
    name: "RGB Chillout"
    state_topic: "campo/chillout/rgb/estado"
    command_topic: "campo/chillout/rgb/control"
    brightness_state_topic: "campo/chillout/rgb/brillo/estado"
    brightness_command_topic: "campo/chillout/rgb/brillo/control"
    rgb_state_topic: "campo/chillout/rgb/color/estado"
    rgb_command_topic: "campo/chillout/rgb/color/control"
    brightness_scale: 100
    payload_on: "1"
    payload_off: "0"
    retain: true
```

En esta integración, tenemos un nuevo tipo de entidad que es el reproductor multimedia, el cual no podría parametrizarse como switch o como luz ya que debe tener sus funciones de reproducir, pausar, control de volumen, cambio de canción, etc.

Para poder tener un control total del reproductor, añadiremos una entidad de tipo media_player mediante el protocolo mpd. De esta forma, Home Assistant se podrá comunicar con Mopidy sabiendo que es un reproductor y por ende facilitando todas las funciones que permiten su control. Tan solo debemos añadir las siguientes líneas, indicando la ip y puerto en los que está configurado Mopidy.

```
media_player:
  - platform: mpd
    host: 10.0.0.91
    port: 6600
```

Tras un reinicio de Home Assistant, dispondremos de todas las entidades en el sistema para poder crear la tarjeta de control en Lovelace. Como en puntos anteriores, iremos al panel deseado y pincharemos sobre el botón añadir tarjeta, configurando la misma a nuestro gusto.

Como se puede ver en el siguiente ejemplo hemos utilizado una tarjeta de tipo button para la apertura y cierre del techo, una de tipo media-control para el reproductor de música y una de tipo rgb-light-card, como la utilizada en el control de la piscina, para la luz RGB.

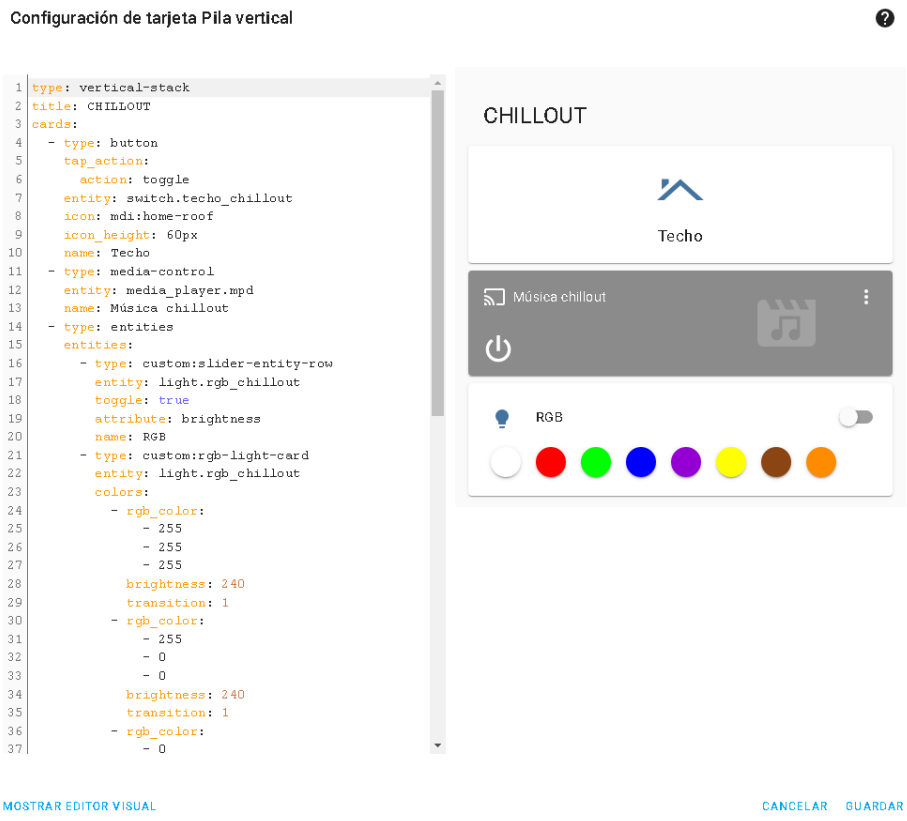

Figura 50 Configuración tarjeta control techo abatible.

6. Pruebas

A continuación, presentaremos una serie de imágenes reales tomadas de todo el sistema de automatización, tanto de los elementos del sistema como de capturas de la propia aplicación de Home Assistant en funcionamiento.

6.1 Pruebas del sistema automatizado puerta de garaje

6.1.1 Motor puerta corredera con sensor inductivo

Figura 51 Sensor inductivo detectando puerta cerrada

Figura 52 Sensor inductivo detectando puerta abierta

6.1.2 Cámara lectura de matrículas con sensor de ruido

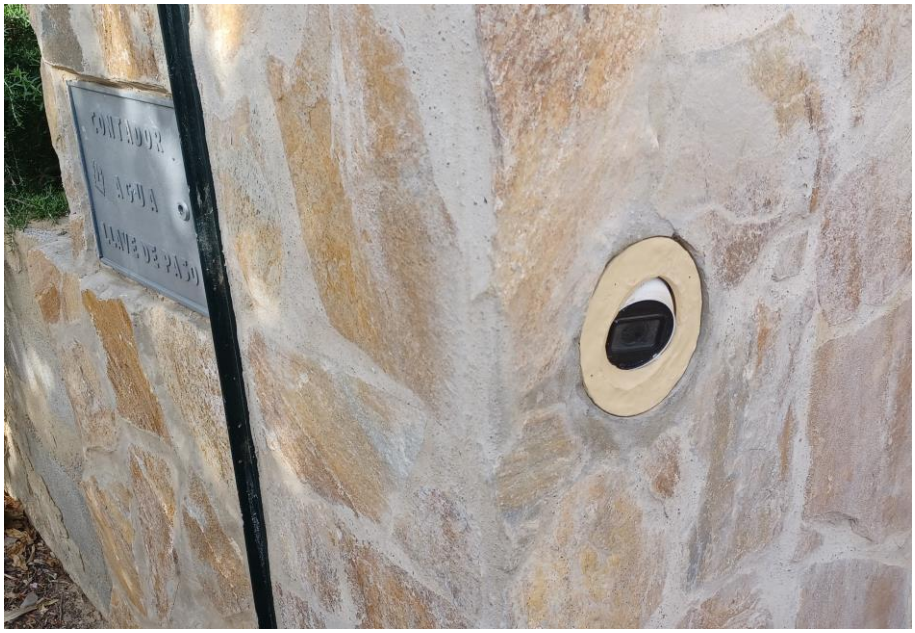

Figura 53 Cámara captura matrículas con sensor de ruido integrado en marco

6.1.3 Centralización Raspberry Pi y conexiones

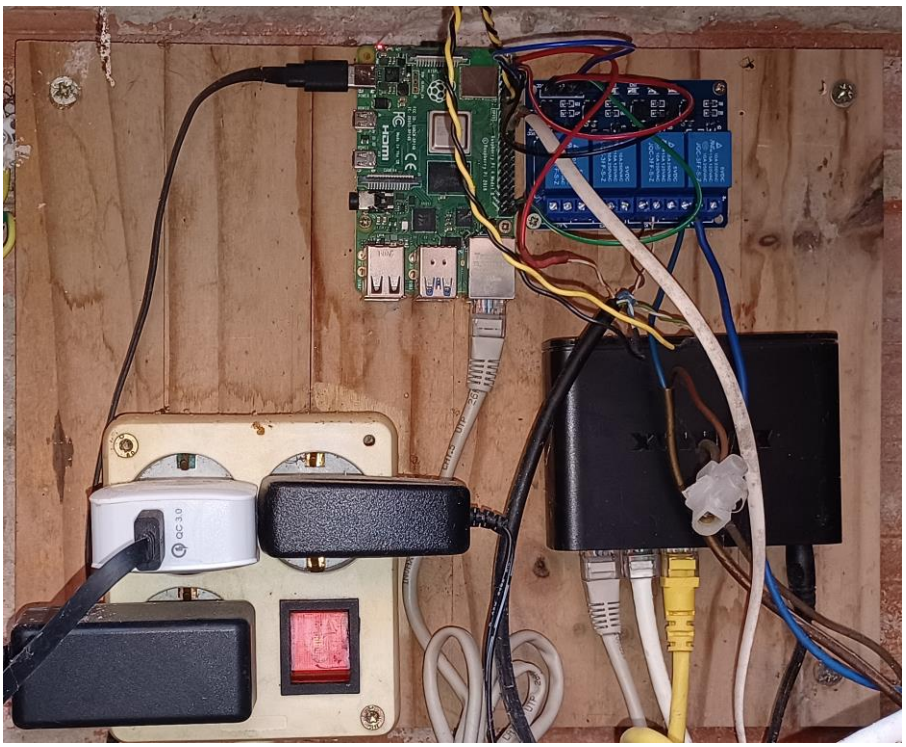

Figura 54 Instalación Raspberry Pi y conexiones en registro estanco

6.1.5 Luz de paso e indicador matrícula autorizada

Figura 55 Proyector LED para luz de paso e indicador matrícula autorizada

6.1.6 Cuadro de mandos Home Assistant

Para el control de la puerta hemos utilizado una tarjeta en la que se muestra la imagen de la cámara de fondo, en blanco y negro cuando la puerta está cerrada y en color cuando la puerta está abriendo o cerrando.

En la parte inferior de la tarjeta, encontramos dos iconos para la apertura de puerta y el encendido de luz respectivamente. Además, es posible abrir la puerta haciendo una pulsación sobre cualquier parte del fondo de la tarjeta, o bien manteniendo una pulsación prolongada, lo que provocará la apertura peatonal de la puerta, con tan solo 80 cm de apertura.

Figura 56 Captura tarjeta de control de puerta en Home Assistant

6.2 Pruebas del sistema automatizado de vivienda

6.2.1 Control de iluminación

Figura 57 Luminaria automatizada

Figura 58 Panel de control de luminarias en Home Assistant

6.2.2 Control de aire acondicionado

Figura 59 Emisor IR enfocado al aparato A/C

Figura 60 Panel de control del A/C en Home Assistant

6.2.3 Supervisión y control sistema de seguridad

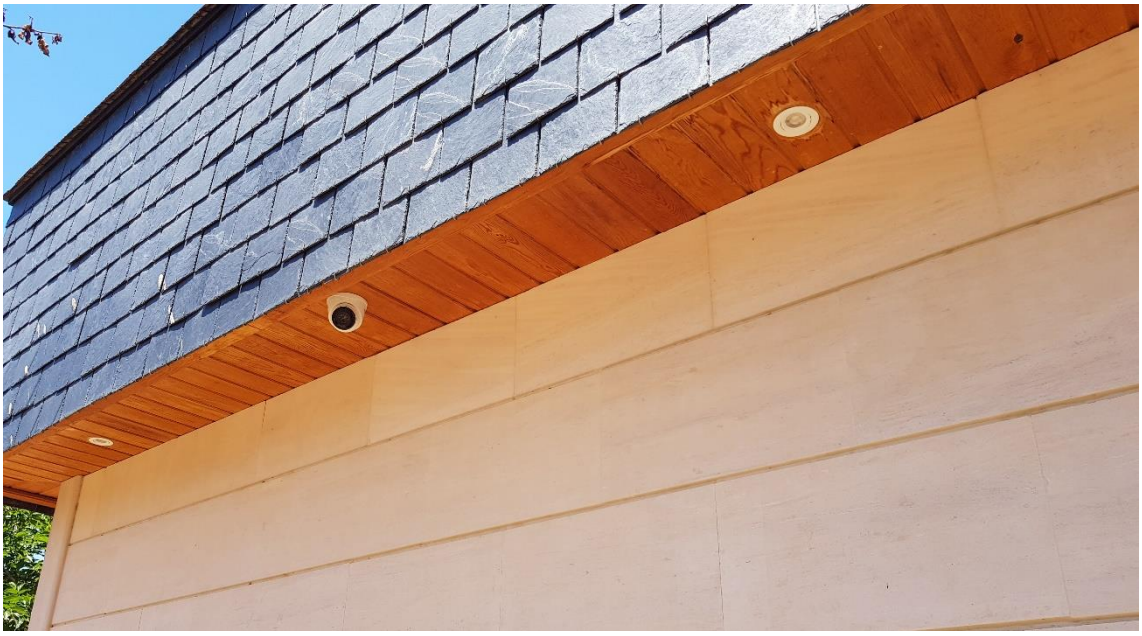

Figura 61 Cámara domo instalada en techo

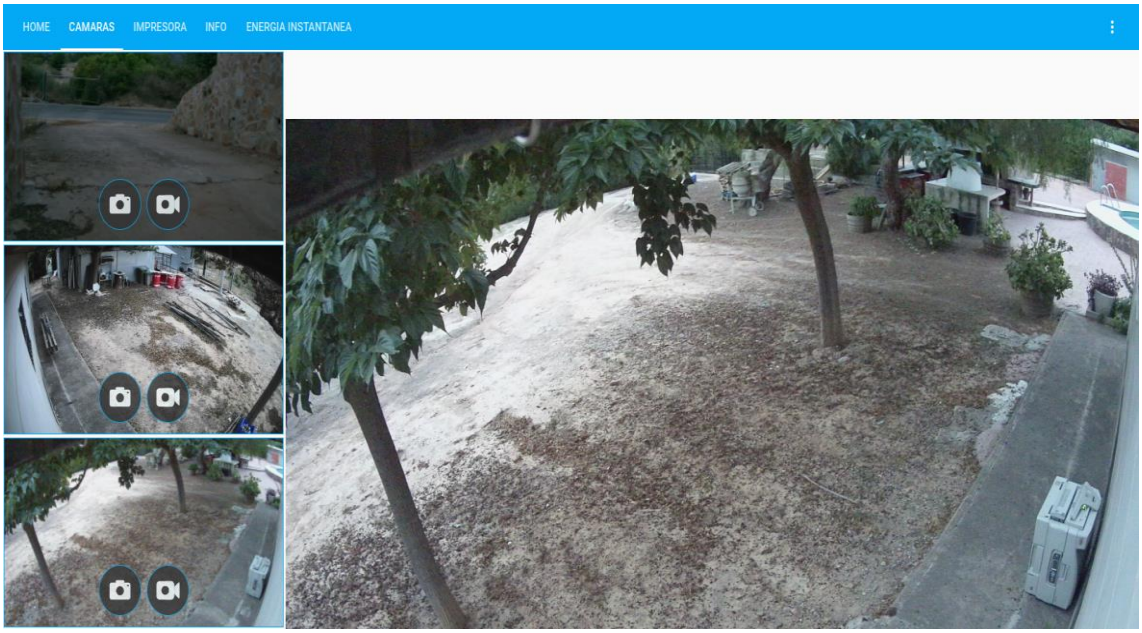

Figura 62 Panel de visualización de cámaras

Figura 63 Central risco instalada en cuadro general de telecomunicaciones.

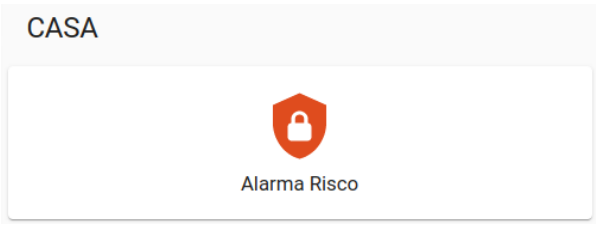

Figura 64 Tarjeta de control de alarma Risco armada

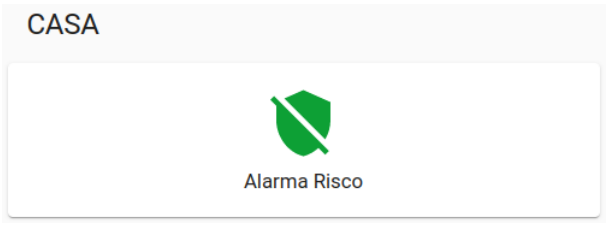

Figura 65 Tarjeta de control de alarma Risco desarmada

7. Conclusiones

Durante los meses de desarrollo del proyecto nos hemos encontrado con muchas dificultades surgidas por la falta de conocimiento en el campo de la domótica y la automatización, descubriendo nuevos protocolos que son usados de forma específica en este campo y ampliando conocimientos en el campo de la electrónica.

Para llegar al resultado final obtenido, hemos tenido que pasar por pequeños desarrollos de prueba que han sido descartados pero que no podríamos haber evaluado correctamente sin haberlos ejecutado.

Quedamos satisfechos con la elección de Home Assistant como núcleo del sistema, pues hemos podido cumplir el principal objetivo que era poder integrar cualquier dispositivo en el sistema de automatización, sin haber tenido que descartar ninguno de los que nos hemos propuesto integrar.

Algunas integraciones han resultado muy sencillas, limitándose a la instalación de un componente en pocos minutos. Otras de ellas han sido complejas, ya que han requerido de un profundo estudio del elemento a integrar, así como del desarrollo o adaptación de componentes software. Lo importante es que para todos los casos hemos encontrado una solución viable, lo que indica que estamos lejos de conocer los límites del sistema y que disponemos una gran versatilidad para futuras integraciones.

Finalmente, ha sido una satisfacción ver como el sistema sigue funcionando de forma estable y sin interrupciones pasados tres meses de la puesta en marcha del mismo.

7.1 Relación del trabajo desarrollado con los estudios cursados

El desarrollo del proyecto ha conllevado la aplicación de diferentes disciplinas en el campo de la informática. Hemos aprovechado muchos de los conocimientos aprendidos en las diferentes asignaturas del grado, como lo son los referentes a la parte electrónica y de programación de microcontroladores de las asignaturas Instrumentación industrial y Mecatrónica, los correspondientes a la parte de diseño y configuración de la red local de la asignatura Redes de computadores y los relacionados con el desarrollo de software y gestión de proyectos software de las asignaturas Diseño de software y Proyecto de ingeniería de software.

8. Trabajos futuros

En la fase de desarrollo del proyecto, hemos leído grandes cantidades de información y documentación, así como casos de éxito de diferentes integraciones, todo ello nos ha ido generando nuevas ideas de automatización que no hemos realizado debido a la limitación de tiempo del que hemos dispuesto para la realización de este proyecto.

Además, conforme hemos ido avanzando en la integración de nuevos sensores y actuadores, han ido surgiendo ideas de automatizaciones que hacen más cómodo el uso del sistema y facilitan el control de la vivienda.

A continuación, resumimos las posibles mejoras o ampliaciones que han surgido:

- Integración de TV: Nos permitiría mostrar la imagen de una cámara cuando detecta movimiento o se abre la puerta.
- Control del sistema mediante asistente de voz.
- Implementación de cuadro de mandos mediante WallPanel en una tablet android: Nos permitiría el control total del sistema y la monitorización a tiempo real del mismo sin tener que hacer uso de nuestro smartphone.

9. Referencias y bibliografía

Instalación Home Assistant en Raspberry Pi - [En línea] [Último acceso: 14 de Septiembre de 2021.]

https://www.youtube.com/watch?v=4zhP3Jh6lg0

Documentación oficial instalación Home Assistant en Raspberry Pi - [En línea] [Último acceso: 14 de Septiembre de 2021.]

https://www.home-assistant.io/installation/raspberrypi

Tutorial instalación Home Assistant - [En línea] [Último acceso: 14 de Septiembre de 2021.]

https://programarfacil.com/domotica/home-assistant/

Instalación Home Assistant Community Store - [En línea] [Último acceso: 23 de Septiembre de 2021.]

https://hacs.xyz/docs/installation/installation

Repositorio de tarjetas Lovelace Home Assistant - [En línea] [Último acceso: 3 de Octubre de 2021.]

https://home-assistant-cards.bessarabov.com/

Repositorio de tarjetas Lovelace - [En línea] [Último acceso: 3 de Octubre de 2021.]

https://github.com/custom-cards

Documentación desarrollo integraciones Home Assistant - [En línea] [Último acceso: 6 de Octubre de 2021.]

https://developers.home-assistant.io/docs/architecture/devices-and-services

Tarjeta Lovelace para control de luz RGB - [En línea] [Último acceso: 23 de Marzo de 2022.]

https://github.com/bokub/rgb-light-card

Componente Home Assistant para control tasmota-ir - [En línea] [Último acceso: 26 de Octubre de 2021.]

https://community.home-assistant.io/t/tasmota-mqtt-irhvac-controler/162915

Agrupar entidades por estados en Home Assistant - [En línea] [Último acceso: 9 de Abril de 2022.]

https://smarthomepursuits.com/display-offline-unavailable-or-missing-sensors-in-home-assistant/

Fork librería pyrisco para centrales sin particiones - [En línea] [Último acceso: 18 de Septiembre de 2021.]

https://github.com/SimonThoustrup/pyrisco/tree/partitionless

Características compilaciones Tasmota - [En línea] [Último acceso: 16 de Septiembre de 2021.]

https://github.com/arendst/Tasmota/blob/development/BUILDS.md

Configurar MQTT con Tasmota en Home Assistant - [En línea] [Último acceso: 16 de Septiembre de 2021.]

https://www.youtube.com/watch?v=g3xuXU5lZ1U

Actualización Sonoff con IR - [En línea] [Último acceso: 15 de Septiembre de 2021.]

https://domotuto.com/flasheo-del-sonoff-basic-para-anadirle-ir/

Protocolo Haier utilizado en tasmota-ir - [En línea] [Último acceso: 23 de Septiembre de 2021.]

https://github.com/crankyoldgit/IRremoteESP8266/blob/master/src/ir_Haier.cpp

10. Anexo

Objetivos de desarrollo sostenible

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	**Alto**	**Medio**	**Bajo**	**No Procede**
ODS 1. **Fin de la pobreza.**				**X**
ODS 2. **Hambre cero.**				**X**
ODS 3. **Salud y bienestar.**				**X**
ODS 4. **Educación de calidad.**				**X**
ODS 5. **Igualdad de género.**				**X**
ODS 6. **Agua limpia y saneamiento.**				**X**
ODS 7. **Energía asequible y no contaminante.**		**X**		
ODS 8. **Trabajo decente y crecimiento económico.**				**X**
ODS 9. **Industria, innovación e infraestructuras.**	**X**			
ODS 10. **Reducción de las desigualdades**.				**X**
ODS 11. **Ciudades y comunidades sostenibles**.				**X**
ODS 12. **Producción y consumo responsables.**		**X**		
ODS 13. **Acción por el clima.**				**X**
ODS 14. **Vida submarina.**				**X**
ODS 15. **Vida de ecosistemas terrestres.**				**X**
ODS 16. **Paz, justicia e instituciones sólidas.**				**X**
ODS 17. **Alianzas para lograr objetivos.**				**X**

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

ODS 7. Energía asequible y no contaminante.

Este ODS, lo hemos seleccionado, ya que nuestro sistema se compone de dispositivos de bajo consumo y se lleva un control del funcionamiento de los dispositivos en el momento adecuado, es decir, no se hace una sobre utilización de los dispositivos, solo se utilizan en el momento que se requieren, ya sean el motor de apertura de la puerta, luminarias o el propio sistema de aire acondicionado.

ODS 9. Industria, innovación e infraestructuras.

Este otro ODS, es uno de los que más destaca en nuestro trabajo final de grado, ya que hemos creado un sistema con recursos reutilizados que se les ha dado otra funcionalidad, creando así una innovación en nuestro trabajo.

Por otra parte, las infraestructuras que hemos llevado a cabo ya tienen una configuración de bajo uso de recursos y de una alta eficiencia energética.

ODS 12. Producción y consumo responsables.

Por último, este ODS también podemos relacionarlo con nuestro trabajo final de grado, ya que como se especifica en el trabajo, hemos realizado y convertido viejos dispositivos y los hemos adaptado a nuestro sistema, como es el caso del motor de la puerta, un motor antiguo sin funcionar, que finalmente se ha conseguido reparar por nosotros mismos, y los hemos aprovechado para ser reutilizados y adaptados para la apertura y cierre de la puerta de entrada y adaptándolo al nuevo sistema de apertura con detección de matrícula.