GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

Sistema inteligente para habitaciones hoteleras

Autor: Luis Bueno Archaga

Director: Francisco Martín Martínez

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Sistema y espejo inteligente para habitaciones hoteleras

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2022/23 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Luis Bueno Archaga Fecha: ……/ ……/ ……

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Firmado por MARTIN MARTINEZ
FRANCISCO MARIA - ***7410**
el día 05/06/2023
con un certificado emitido
por AC FNMT Usuarios

Fdo.: Francisco Martín Martínez Fecha: ……/ ……/ ……

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

Sistema inteligente para habitaciones hoteleras

Autor: Luis Bueno Archaga

Director: Francisco Martín Martínez

Madrid

Agradecimientos

A todos los que me han apoyado, especialmente a mis padres y a mi hermana.

SISTEMA INTELIGENTE PARA HABITACIONES HOTELERAS

Autor: Bueno Archaga, Luis.
Director: Martín Martínez, Francisco.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Sistema para habitaciones hoteleras con un espejo inteligente, controlable con gestos, para mostrar distinta información en tiempo real de la red y de la estancia (temperatura y humedad), además de regular elementos de iluminación como la persiana y las bombillas del cuarto. Todo lo anterior está soportado por un sistema Raspberry Pi.
Palabras clave: Raspberry Pi, Domótica, Habitación, Espejo inteligente, Hostelería

1. Introducción

El trabajo se enfoca en satisfacer las nuevas necesidades de los clientes del mercado de la hostelería, con el fin de poder llevar a cabo un proyecto innovador para mejorar su calidad y eficiencia. Los tres problemas que se deben resolver son la domotización de las habitaciones de los huéspedes, siendo necesario conocer la humedad y la temperatura de la habitación, así como poder controlar la iluminación con las luces y las persianas, y por último, mostrar información a los clientes en una interfaz cómoda y sencilla como un espejo inteligente manejado con gestos. Estos tres objetivos deben estar incluidos en un mismo diseño para poder proporcionar una mejor visualización de la información.

2. Definición del proyecto

Para cumplir con los objetivos del diseño se tiene que controlar con gestos tanto las vistas del espejo inteligente como el movimiento del motor de la persiana, para esto se va a usar un solo programa de Python (ANEXO II) con el reconocimiento de gestos (Mediapipe Guía de soluciones, 2023) y el control de los pines de la Raspberry Pi (Python Software Foundation, 2022).

Lo principal es mostrar en el espejo la información distribuida en distintas vistas con datos económicos de la reserva del cliente, el estado de la persiana, de las luces, la humedad con la temperatura de su aposento y de internet: el clima, la calidad del aire, las noticias del día y un calendario con festivos. Se consigue mostrar esto gracias a la plataforma MagicMirror[2] basada en tecnologías web como HTML, CSS y JavaScript. Esta tecnología también consta de una comunidad muy activa que aporta múltiples módulos en los que se ha apoyado este proyecto.

3. Descripción del sistema

El proyecto por parte del hardware se basa en una RaspBerry Pi conectada a internet que se compone de una interfaz en un espejo inteligente, una cámara para controlar tanto las pestañas que se muestran en el espejo como el motor de la persiana del cuarto, un sensor de temperatura y humedad y por último de un sistema de luces. Esto se puede ver esquematizado en la Ilustración 1. El espejo inteligente consiste en una pantalla con un espejo acrílico delante que permite la visión de las partes más brillantes (con el brillo de una pantalla encendida es suficiente para que permita la visión de esa parte) y refleja la parte oscura de la pantalla.

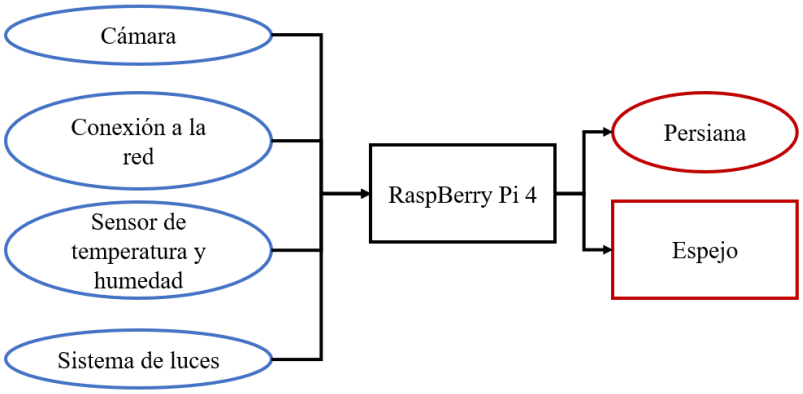

Ilustración 1. Esquema Hardware

Por parte del software hay dos partes fundamentales: el programa de Python (ANEXO II) que como ya hemos explicado es el responsable de dar soporte a las funcionalidades de control con gestos, y la plataforma de MagicMirror[2] que enseña la información que recibe de la red (principalmente gracias a la tecnología API REST), gestiona el movimiento entre pestañas de los gestos que recibe de Python, muestra el estado de las luces, la temperatura y humedad del sensor y los datos que considere el hotel relevante para el cliente (en este proyecto a modo de ejemplo se presenta el estado económico del huésped, el menú del día y el estado del aparcamiento) en una tabla en formato JSON. El diseño software se puede apreciar en la Ilustración 2.

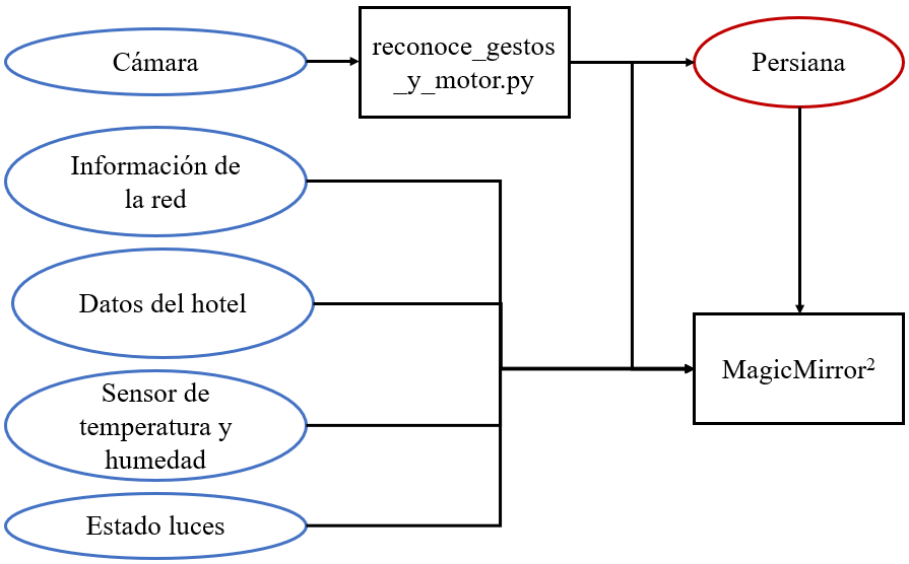

Ilustración 2. Esquema Software

4. Resultados

Se ha creado una versión del proyecto de forma satisfactoria y aunque se debería adaptar para un entorno comercial de hostelería, su desarrollo cumple con los objetivos con algunas problemáticas, pero ninguna siendo relevante.

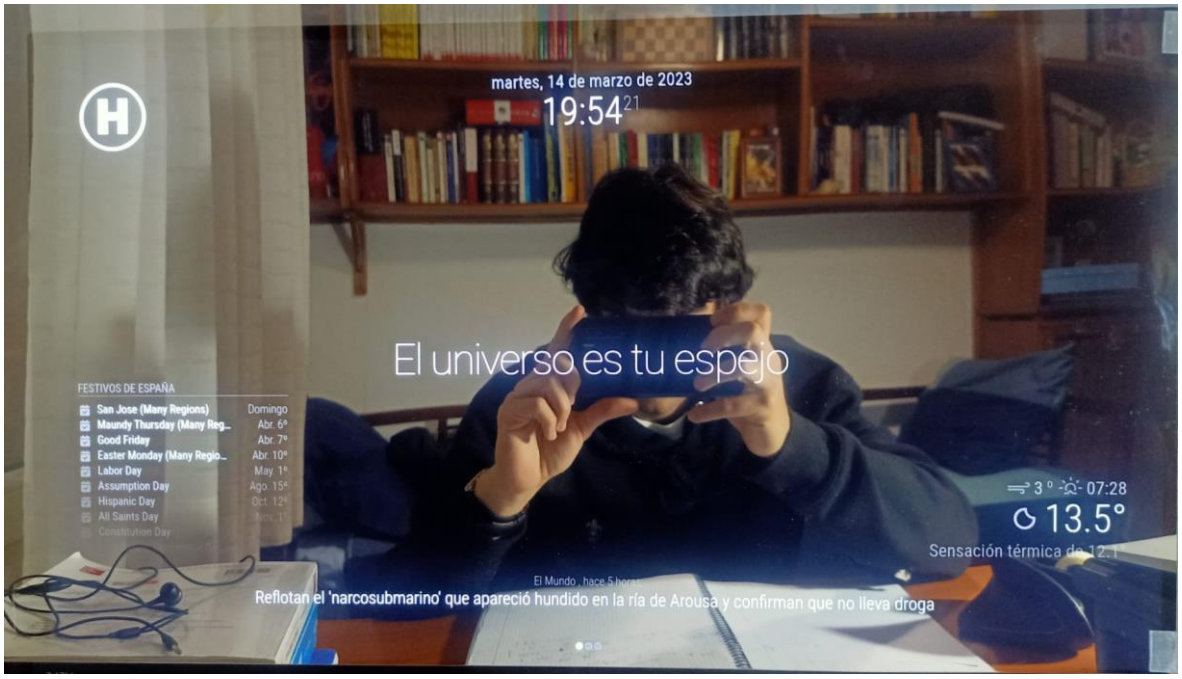

Ilustración 3 – Desarrollo del proyecto

5. Conclusiones

Se ha cumplido con los objetivos aportando la flexibilidad de diseño basada en MagicMirror², junto con las ventajas de interacción del control por gestos (frente al uso de botones o las pantallas táctiles), ya que no entra en contacto el usuario con el espejo.

Este nuevo enfoque centrado en juntar todas las partes en un sistema para poder mostrar toda la información da a la domotización de habitaciones de hostelería diferenciación en el servicio.

Para proyectos futuros se puede explorar el desarrollo de un sistema de control de las luces o añadir los sistemas para recibir información del hotel. Es decir, en vez de solo mostrar los datos (como está de ejemplo en este proyecto, con el estado económico del huésped, el menú del día y el estado del aparcamiento) también recibirlos de un procedimiento más automático y personalizable.

6. Referencias

[1] *Mediapipe Guía de soluciones*. (9 de Mayo de 2023). Obtenido de https://developers.google.com/mediapipe/solutions/guide

[2] Python Software Foundation. (6 de Febrero de 2022). *A module to control Raspberry Pi GPIO channels*. Obtenido de https://pypi.org/project/RPi.GPIO/

INTELLIGENT SYSTEM FOR HOTEL ROOMS

Author: Archaga, Luis Bueno.
Director: Martinez, Francisco Martin.
Collaborating Entity: ICAI - Comillas Pontifical University

PROJECT SUMMARY

This project focuses on meeting the new needs of clients in the hospitality market by developing an innovative system for hotel rooms. The system includes a smart mirror that can be controlled through gestures and displays real-time information from the network and the room, such as temperature and humidity. It also allows for the regulation of lighting elements like blinds and light bulbs. The entire system is supported by a Raspberry Pi.

Keywords: Raspberry Pi, Home automation, Room, Smart mirror, Hospitality

1. Introduction

The objective of this project is to improve the quality and efficiency of hotel services by addressing three main challenges: domotization of guest rooms, which involves monitoring humidity and temperature, as well as controlling lighting and blinds; and providing clients with information through a comfortable and user-friendly interface, such as a smart mirror operated with gestures. These objectives are integrated into a single design to enhance information visualization.

2. Definition of the project

To achieve the design objectives, both the smart mirror views and the blind motor movement are controlled using gestures. A Python program (ANNEX II) is used for gesture recognition (Mediapipe Solutions Guide, 2023), along with controlling the Raspberry Pi pins (Python Software Foundation, 2022).

The key functionality includes displaying different views of information on the mirror, such as economic data related to the client's reservation, blind status, lighting status, room temperature and humidity, and internet-based content like weather, air quality, daily news, and holiday calendar. This is made possible by utilizing the MagicMirror2 platform, which is built on web technologies such as HTML, CSS, and JavaScript. The project benefits from an active community that provides multiple modules supporting this platform.

3. Description of the system

The hardware aspect of the project involves a Raspberry Pi connected to the internet, comprising a smart mirror interface, a camera for controlling the displayed content and the blind motor, a temperature and humidity sensor, and a lighting system. This setup is illustrated in Diagram 1. The smart mirror consists of a screen covered with an acrylic mirror that allows clear visibility of the brighter areas, while reflecting the dark areas.

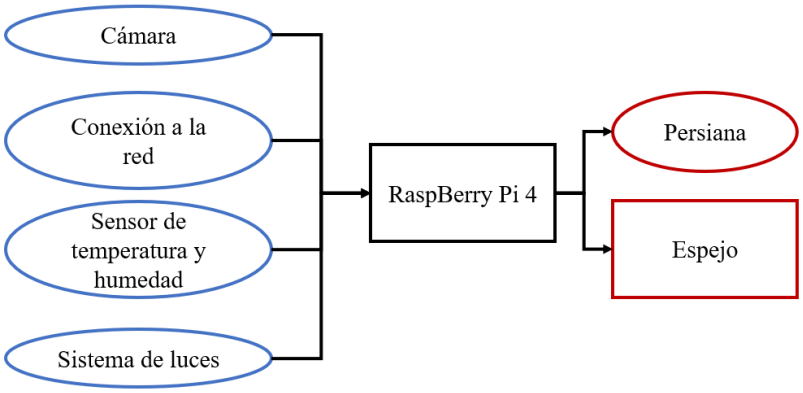

Diagram 1: Hardware Diagram

On the software side, there are two main components: the Python program (ANNEX II) responsible for gesture-based control functions, and the MagicMirror2 platform, which displays information received from the network, primarily using REST API technology. The platform manages the switching between gesture-based tabs received from Python, shows the status of lights, temperature and humidity readings from the sensor, and presents relevant data to the client in a table format, such as the economic status of the guest, daily menu, and parking availability, stored in JSON format. The software design is depicted in Diagram 2.

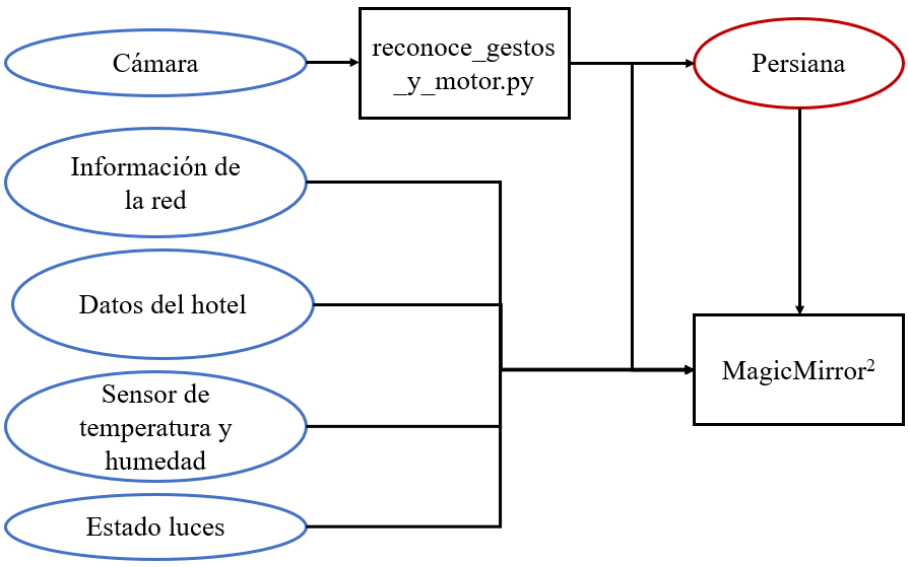

Diagram 2: Software Scheme

4. Results

A satisfactory version of the project has been developed, although it requires adaptation for commercial hospitality environments. The project has successfully achieved its objectives, with some minor issues that are not considered significant.

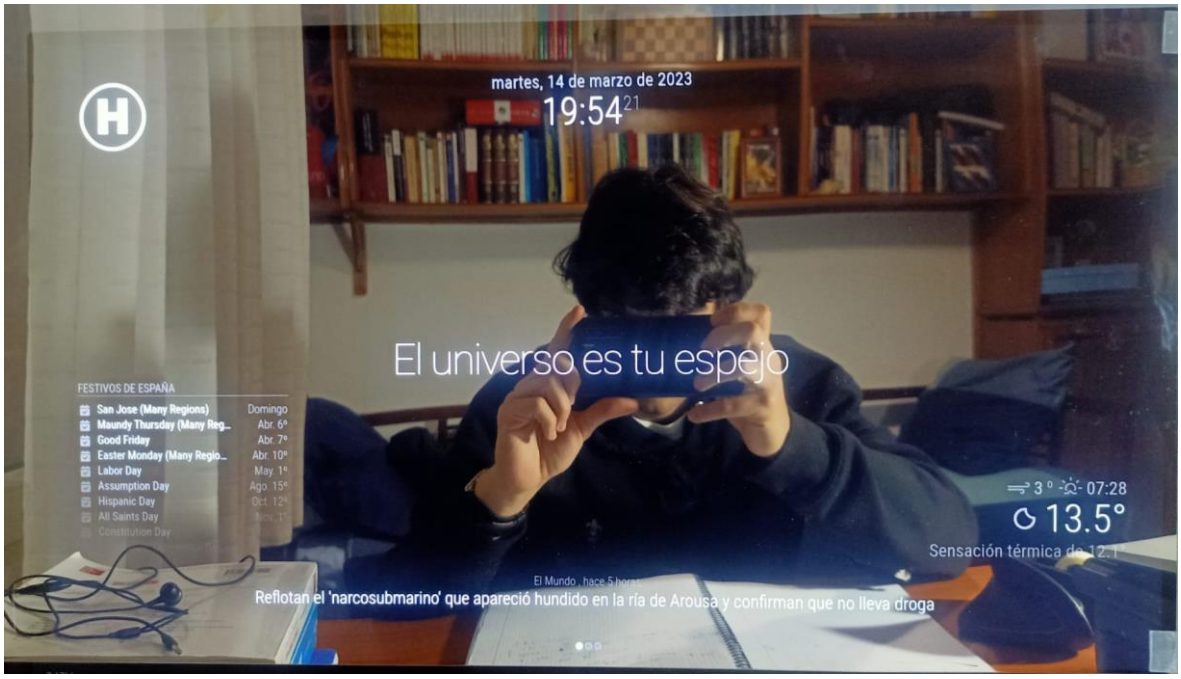

Diagram 3: Project development

5. Conclusions

The project has successfully met its objectives by offering design flexibility through the MagicMirror2 platform and leveraging the benefits of gesture-based control, which eliminates the need for physical contact with the mirror compared to buttons or touch screens. This novel approach, integrating all components into a comprehensive system for displaying information, provides a differentiated service in the automation of hotel rooms.

For future projects, possibilities include exploring the development of a lighting control system or incorporating systems to receive information from the hotel. This would allow for not only displaying data (as demonstrated in this project, with the economic status of the guest, the menu of the day and the status of the car park) also receiving them from a more automatic and customizable procedure.

6. References

[1] Mediapipe Solutions Guide. (May 9, 2023). Retrieved from https://developers.google.com/mediapipe/solutions/guide

[2] Python Software Foundation. (February 6, 2022). A module to control Raspberry Pi GPIO channels. Retrieved from https://pypi.org/project/RPi.GPIO/

Índice de la memoria

1. Introducción 6

1.1 Motivación del proyecto 6

2. Estado de la Cuestión 8

2.1.1 Comerciales 8

2.1.2 No comerciales 12

Capítulo 3. Definición del Trabajo 14

3.1 Justificación 14

3.2 Objetivos 14

3.3 Metodología 15

3.4 Planificación y Estimación Económica 16

4. Descripción de las Tecnologías 18

4.1 MagicMirror^2 18

4.2 MediaPipe reconocimiento de gestos 19

4.3 API REST 20

5. Sistema desarrollado 21

5.1 Diseño Hardware 21

5.2 Diseño de interfaces 24

5.2.1 Reposo 25

5.2.2 Estado de la habitación 26

5.2.3 Estado del hotel 27

5.3 Diseño Software 28

5.3.1 Entradas 32

5.3.2 Control 39

5.3.3 Salidas 42

6. Análisis de resultados 49

6.1 Análisis del sistema y diseño 49

6.2 Implementación 50

7. Conclusiones y Trabajos Futuros 51

8. Bibliografía 52

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS 57

ANEXO II 59

ANEXO III 63

Índice de figuras

Figura 1. Conexión del motor con el controlador L298N y el sensor DHT11 22
Figura 2. Asignación de pines GPIO Raspberry Pi 4 23
Figura 3. Asignación de pines del sensor DHT11 23
Figura 4. Interfaz del controlador de motores L298N 24
Figura 5. Espejo acrílico 25
Figura 6. Ventana de reposo 26
Figura 7. Ventana con información del estado de la habitación 27
Figura 8. Ventana con información del estado del hotel 28
Figura 9. Diagrama de flujo del espejo inteligente 29
Figura 10. Módulos y su funcionalidad en MagicMirror 31
Figura 11. Captura de la cámara con mapeo de la mano 33
Figura 12. Captura con referencias del lado izquierdo y derecho 34
Figura 13. API keys de mi cuenta en OpenWeahter 35
Figura 14. Página con la portada de El País en XML 36
Figura 15. Plataforma de datos de calidad del aire Confirmación de token de API 37
Figura 16. mDNS Discovery aplicación para obtener el bridge id 38
Figura 17. Aplicación test para la API de Philips 39
Figura 18. Diagrama de flujo funciones de control 41
Figura 19. Diagrama de flujos Funciones de la Ventana 1 45
Figura 20. Diagrama de flujos Funciones de la Ventana 2 46
Figura 21. Diagrama de flujos Funciones de la Ventana 3 47
Figura 22. Montaje del espejo inteligente 49
Figura 23. ODS 11 Ciudades y comunidades sostenibles 57
Figura 24. ODS 3 Salud y bienestar 57
Figura 25. ODS 13 Acción climática 58

Índice de tablas

Tabla 1. Comparativa de atributos en soluciones comerciales 11
Tabla 2. Comparativa de atributos en soluciones no comerciales 13

1. INTRODUCCIÓN

El proyecto se centra en satisfacer las nuevas necesidades de los clientes del mercado de la hostelería, más concretamente implementar el proyecto en las habitaciones que contratan los huéspedes. En estos cuartos, se busca aplicar sistemas inteligentes para mostrar información de temperatura y humedad, controlar elementos como la iluminación con persianas y mostrarlo junto con el estado de las bombillas y datos que se obtienen de la red en un espejo inteligente. Todo lo anterior soportado por una Raspberry Pi y manejado por gestos.

1.1 MOTIVACIÓN DEL PROYECTO

Satisfacer las nuevas necesidades de clientes de hostelería, teniendo un gran impacto en la experiencia del huésped de hoteles, creando un proyecto que reciba datos de temperatura y de humedad, que controle luces y persianas, y que muestre esta información. Gracias a tener la información del mismo sistema se presentará en un espejo inteligente con el que se podrá interactuar con gestos para ver distintas ventanas con diferentes datos. Reconocemos el impacto significativo que las soluciones tecnológicas pueden tener en este ámbito, mejorando su comodidad y conveniencia durante su estancia.

2. ESTADO DE LA CUESTIÓN

El estudio del estado de la cuestión se divide en soluciones comerciales y no comerciales, a su vez se separa en un desarrollo de cada solución y a continuación en una tabla que sintetiza las características relevantes de cada propuesta. Los atributos que son más relevantes para mejorar la experiencia del huésped, que son los problemas esenciales que pretende resolver este proyecto, son, mostrar la información, controlar las persianas y las luces de la habitación, medir tanto temperatura como humedad y tener integrada toda la solución en el mismo sistema para enseñar los datos del cuarto. Esto lo desarrollamos en *Tabla 1*.

2.1.1 COMERCIALES

2.1.1.1 GFP Lab hospitality automation

Según su página web, su objetivo es implementar múltiples sistemas eléctricos, de fontanería, de termorregulación e informáticos, la empresa ofrece una cerradura electrónica para abrir la habitación con una tarjeta RFID que a su vez activaría el resto de los sistemas automáticamente al entrar en el cuarto (GFPLab).

Se activaría un sistema termorregulador conectado a radiadores, sistemas de climatización centralizados y sistemas de iluminación con puntos de luz regulables o luces RGB. Todos los sistemas anteriores integrados en internet de las cosas y ofrecen una interfaz para gestionar automáticamente las llaves para los clientes con tarjetas RFID o virtuales. Cabe resaltar la importancia de las tarjetas RFID en sus diseños.

2.1.1.2 ArabaDomotic

Ofrecen supervisar la climatización, iluminación y energía, mejorar la seguridad general de los accesos y supervisar los sistemas de seguridad, facilitar el control remoto de los sistemas y la transmisión de datos para la intercomunicación (ArabaDomotic).

No concretan los dispositivos que conforman estos sistemas y cabe resaltar que en hostelería solo hay un ejemplo de un proyecto que realizaron en el “Hotel Jardines de Uleta” pero

implementaron la iluminación y climatización del parking, salones polivalentes y pasillos. Es decir, no implementaron sus sistemas en habitaciones.

2.1.1.3 Rimini System Integrator

Ofrecen varios servicios como el control del acceso a la habitación, de la temperatura, del audio y video, de la seguridad (con alarmas de humo, gas, intrusión, agua o alarmas técnicas) y de conexiones a Internet (RiminiSystemIntegrator). Estos servicios están enfocados para gestionar del hotel en vez de los huéspedes de este. No especifican ni desarrollan la aplicación, diseño o instalación de estos sistemas.

2.1.1.4 Vikey

Ofrecen para la hostelería distintos dispositivos que se instalan para poder disponer de un sensor de humedad y temperatura, un sensor de ruido, un sensor de filtraciones de agua, un termostato, una válvula para el control de radiadores, un sensor de movimiento, un detector de humo, un sensor de apertura de puertas y ventanas, unos sistemas inteligentes de ahorro de energía para enchufes y cables, un control remoto por infrarrojos y un switch inteligente de pared capaz de monitorear la fuente de alimentación de otros dispositivos (Vikey). Estos dispositivos no muestran la información excepto por la temperatura en el termostato.

2.1.1.5 Microdevice

Se enfoca en poder gestionar la seguridad, el ahorro de energía, las alarmas y los accesos de control (Microdevice). Esto facilita la gestión del hostelero en vez de mejorar la experiencia del cliente. Específicamente ajustan con sistemas inteligentes la temperatura de los dormitorios y las zonas comunes, la iluminación y las alarmas de exclusivamente zonas comunes se controlan también.

2.1.1.6 MG Electricidad

Dicen tener capacidad de aplicar servicios enfocados a domicilios para la hostelería. Estos son el control de acceso, luz, temperatura, electricidad, música, persianas y alarmas de seguridad (MGelectricidad). Esta información y esta gestión se da desde una aplicación, ya que está enfocado al hogar. La aplicación para el cliente de un hotel requeriría de un móvil

u otro dispositivo capaz de conectarse y de funcionar correctamente o no mostraría información al huésped. En la recepción se configuran y monitorean parámetros generales del hotel, en concreto con datos relacionados con los cuartos de los clientes.

2.1.1.7 Hotel Suppliers

Se trata de una empresa distribuidora de soluciones para hoteles inteligentes, que a diferencia de los anteriores incluye espejos inteligentes (Hotel-suppliers). También hay que destacar que ofrecen un control de acceso al hotel, de las persianas, del manejo de la energía del hotel y sistemas de señalización de hoteles. Su proveedor es Livmark d.o.o. y no muestra ninguna información de la habitación.

2.1.1.8 Livmark d.o.o.

No está enfocado en buscar soluciones domóticas, sino que diseñan exclusivamente pantallas que forma parte de este espejo, tiene calidad full HD, es táctil y tiene un diseño enfocado a la industria hostelera basado en Android (Livmark). El espejo cumple con la IP(“Ingress Protection”) 65, certificando que es hermético al polvo y está protegido frente al agua proyectada. No muestra información de la habitación.

Nombre	Sensor de entrada en la habitación	Sistemas en partes independientes o en un sistema integrado	Medidas (H humedad T temperatura)	Control Domótico (L luz P persianas)	Muestra información de la habitación con un espejo inteligente
GFP Lab hospitality automation	RFID	Integrado en un sistema	H T	L P	×
Araba Domotic	Control digital de accesos	Integrado en un sistema	H T	L P	×*
Rimini System Integrator	Control digital de accesos	Integrado en un sistema	H T	L	×
Vikey®	×	Partes independientes	H T	P	×
Microdevice	Control de accesos	Integrado en un sistema	H T	L	×
MG Electricidad	Bloqueo de la puerta	Integrado en un sistema	H T	L P	×**
Hotel Suppliers	Control de accesos	Partes independientes	H T	L P	×***
Livmark d.o.o.	×	Partes independientes	×	×	×***
Nuestra solución	×	Integrado en un sistema	H T	L P	✓****

* Se gestiona a través de dispositivos como un ordenador, teléfono móvil o Tablet conectado a internet.

**Con una aplicación de un teléfono móvil o Tablet.

***No muestra información de la habitación, es un espejo inteligente táctil basado en Android con conexión a internet.

**** Un espejo inteligente que muestra información de la habitación

Tabla 1. Comparativa de atributos en soluciones comerciales

2.1.2 NO COMERCIALES

No hay soluciones no comerciales enfocadas a hostelería, pero hay múltiples proyectos con espejos inteligentes. Los atributos diferenciadores de los espejos inteligentes los estudiaremos en la Tabla 2.

2.1.2.1 Hackster.io "Gesture Controlled Smart Mirror" de Team CodersCafe: Shebin Jose Jacob, Nekhil

Espejo inteligente basado en Raspberry Pi, no está integrado con otros aparatos fuera del espejo inteligente excepto por internet y se puede interactuar con él (Hackster). Con gestos en una cámara puedes ver distintas ventanas de la interfaz. Muestra dos ventanas, en una aparecen la previsión meteorológica del día, la hora, una frase y una gráfica de casos de covid, y en la otra se ven los titulares de las noticias del día.

2.1.2.2 Hackster.io "Smart Mirror Touchscreen (with Face Recognition)" de Eben Kouao

Pantalla táctil gracias a un marco infrarrojo alrededor del espejo inteligente, también cuenta con una cámara para el reconocimiento facial (Hackster). Basado en Raspberry Pi y no tiene otros aparatos fuera del espejo inteligente, excepto la conexión a internet. Se enseña una ventana con la hora, una agenda con cosas que hacer, un mensaje con una imagen, un menú de reproducción de música y la previsión meteorológica de la semana.

2.1.2.3 Instructables.com "How to Make a DIY Smart Mirror" de WickedMakers

Sin conexión con otros sistemas es un espejo inteligente, excepto por internet, basado en Raspberry Pi (instructables). Se interactúa gracias a un ratón y teclado con el espejo inteligente. Aparece una ventana con la previsión meteorológica de la semana, una agenda con cosas que hacer, un reproductor de música, una imagen en el centro y la hora con la fecha.

Nombre	Control con gestos	Pantalla táctil	Basado en Raspberry Pi	Ratón y teclado	Reconocimiento facial	Sistemas en partes independientes o en un sistema integrado
Gesture Controlled Smart Mirror	✓	×	✓	×	×	Parte independiente
Smart Mirror Touchscreen (with Face Recognition)	×	✓	✓	×	✓	Parte independiente
How to Make a DIY Smart Mirror	×	×	✓	✓	×	Parte independiente
Nuestra solución	✓	×	✓	×	×	Integrado en un sistema

Tabla 2. Comparativa de atributos en soluciones no comerciales

Hemos optado por el control de gestos, porque es la más cómoda para moverse de una ventana a otra ventana y puedes usarla sin tocar nada, por lo tanto, evitas tener que limpiar el espejo o el ratón y teclado que sería trabajo del personal del hotel innecesario. En lo referente al software, preferimos que esté basado en Raspberry Pi como las soluciones No comerciales, en vez de basarlo en Android como los espejos inteligentes de las soluciones Comerciales. Esto se debe a que Raspberry Pi tiene una plataforma modular de espejos inteligentes "MagicMirror²".

Capítulo 3. DEFINICIÓN DEL TRABAJO

3.1 JUSTIFICACIÓN

La justificación para este proyecto es la falta en el mercado de un sistema con todas las características que proponemos. Al desarrollar esta solución, ofrecemos a los clientes del mercado de la hostelería una opción única y atractiva. Nuestro proyecto combina tecnología, comodidad para los huéspedes y un enfoque intuitivo basado en gestos, lo cual lo convierte en una propuesta altamente competitiva.

En resumen, este proyecto radica en su capacidad para satisfacer las necesidades emergentes de los clientes de la hostelería, mejorando su experiencia a través de la implementación de sistemas inteligentes en las habitaciones. Además, la carencia de soluciones similares en el mercado da una ventaja competitiva con un fuerte potencial de comercialización.

3.2 OBJETIVOS

Los objetivos de este proyecto son:

1. Desarrollar un espejo inteligente controlable con gestos.
2. Implementar la visualización de información en tiempo real de la red, así como datos de la luz, temperatura y humedad del aposento.
3. Regular los elementos de iluminación, como las persianas, mediante el uso del sistema.
4. Lograr una integración fluida entre el programa de control de gestos en Python y la plataforma MagicMirror2 para una interacción intuitiva y cómoda del usuario.
5. Proporcionar una mayor comprensión de los elementos, con una mejor visualización de la información del cuarto.

6. Contribuir a la experiencia de los clientes en el sector de la hostelería mediante soluciones tecnológicas innovadoras.

Esto nos permitirá desarrollar un trabajo completo y funcional que se ajuste a las necesidades y expectativas de clientes del mercado hotelero, ofreciendo una opción destacable por su comodidad, versatilidad y características innovadoras.

3.3 METODOLOGÍA

Para lograr los objetivos establecidos, la metodología será:

1. Diseño de la interfaz de cada ventana: se busca detallar antes de empezar a programar el diseño de cada ventana del espejo inteligente.
2. Programación de cada ventana del espejo inteligente: basándonos en el trabajo anterior y utilizando de software el "Raspberry Pi Raspbian" y la plataforma modular de espejos inteligentes "MagicMirror²".
3. Programación de la interacción con el espejo inteligente: teniendo el trabajo anteriormente dicho se verá que forma de interactuar con el espejo inteligente es la más conveniente y se implementará usando "Raspberry Pi Raspbian".
4. La integración sensores humedad y temperatura, y actuadores de luces y de persianas.
5. Documentación y redacción del TFG durante que se llevará a cabo durante todo el desarrollo del proyecto.

3.4 PLANIFICACIÓN Y ESTIMACIÓN ECONÓMICA

La planificación se muestra en el siguiente diagrama de Gantt:

Tarea	Propuesta Fecha final	11 2022	12 2022	1 2023	2 2023	3 2023	4 2023	5 2023	6 2023
Diseño de interfaz de cada ventana	30/11/2022								
Programación de cada ventana del espejo inteligente	29/1/2023								
Programación de la interacción con el espejo inteligente	5/2/2023								
Integración de sensores de humedad y temperatura	28/4/2023								
Integración de actuadores para luces y persianas	19/5/2023								

La estimación del coste de desarrollo:

- Raspberry Pi y componentes relacionados: 86,39 € (raspipc.es, 2009)
- Espejo acrílico: 42,76 € (Supreme Tech, 2016)
- Cámara Raspberry Pi: 8,49 € (AZDelivery, 2016)
- Sensor DHT11 de temperatura y humedad: 5,25 € (BricoGeek, s.f.)
- Cables HDMI, micro HDMI y demás elementos de montaje: 5 €
- Controlador motores DC: 3,99 € (OcioDual, 2019)
- Monitor o pantalla: en este caso se ha usado el monitor Philips 243V7QDSB/00 de 100€ pero puede utilizarse otro.

El coste estimado para el desarrollo del proyecto se estima en: 251,88 € contando con la pantalla.
Es importante que estos costes son estimaciones y varían los precios del mercado y las necesidades específicas.

4. DESCRIPCIÓN DE LAS TECNOLOGÍAS

Las tecnologías que vamos a explicar se aclaran para tratarse en diferentes apartados, siendo estas la plataforma MagicMirror^2, la librería MediaPipe y la arquitectura API REST.

4.1 MAGICMIRROR^2

Es una plataforma modular de código abierto para espejos inteligentes (MagicMirror² Documentación, 2023). MagicMirror² utiliza Electron como "application wrapper" pudiendo transformar su sistema en una aplicación de escritorio independiente en Raspberry Pi (Electron Documentación, 2023). Electron combina Chromium y Node.js, creando aplicaciones con tecnologías web como HTML, CSS y JavaScript.

La característica principal de esta plataforma es la modularidad, que se ve beneficiada por todas los atributos de MagicMirror^2, como la personalización o la amplia comunidad activa que colabora en el desarrollo de módulos. Gracias a esto se amplían mucho las posibilidades de añadir funcionalidades o información al espejo. Como veremos en el desarrollo del proyecto, aunque no haya un módulo que cumpla específicamente con el cometido que queremos, podemos apoyarnos en otros y configurarlos con ciertas adaptaciones, lo cual aumenta todavía más la versatilidad de esta plataforma para nuestras necesidades.

Otra característica de MagicMirror² es la personalización completa del aspecto y la funcionalidad del espejo inteligente. Pudiendo elegir entre múltiples módulos existentes o crear sus propios módulos según las preferencias.

Cabe mencionar como la interfaz de usuario es flexible y adaptable. Permitiendo organizar los módulos en la pantalla del espejo en la disposición preferida, lo que permite mostrar información relevante para los huéspedes, como las noticias, el clima, el calendario, el menú del día, etc.

Cómo veremos en el apartado de API REST, MagicMirror[2] puede integrarse con diversas API, lo que da información en tiempo real para los módulos. Esta integración de API da pie a aumentar todavía más las personalizaciones realizables con esta herramienta.

Contar con una comunidad activa de usuarios y desarrolladores que colaboran en nuevos módulos, brindando soporte y compartiendo ideas, caracteriza enormemente la plataforma (Teeuw, 2023). El crecimiento constante de miembros, con sus contribuciones, mantiene actualizadas las últimas mejoras y novedades en cada funcionalidad.

4.2 MEDIAPIPE RECONOCIMIENTO DE GESTOS

MediaPipe es una tecnología de librerías y herramientas desarrollada por Google enfocada en procesar datos multimedia (Mediapipe Guía de soluciones, 2023). Es especialmente popular en el ámbito del reconocimiento de gestos o posturas, ya que realiza un análisis de movimientos en tiempo real.

Su principal fortaleza radica en la capacidad de realizar inferencias de modelos de aprendizaje automático en dispositivos locales con recursos limitados, como una Raspberry Pi. Utiliza redes neuronales profundas y técnicas de visión, haciéndolo ideal para aplicaciones basadas en gestos.

Una de las características es el enfoque modular y escalable. Proporcionando varios módulos predefinidos, como detección de manos, seguimiento facial y detección de posición corporal, que se pueden combinar y adaptar para cada proyecto. Hay también herramientas y API para desarrollar nuevas soluciones personalizadas.

El flujo de trabajo en MediaPipe generalmente implica capturar datos de entrada, y pasarlos a módulos de procesamiento encargados de tareas específicas. Estos módulos pueden personalizarse y ajustarse a las necesidades.

Es una tecnología versátil y potente de reconocimiento de gestos en tiempo real. Teniendo la disposición modular, la capacidad de funcionar con recursos limitados y la amplia gama de módulos predefinidos.

4.3 API REST

La tecnología de API REST (Representational State Transfer) es utilizada en el desarrollo de aplicaciones web y servicios en línea (Fielding, 2000). Basada en principios y convenciones de la comunicación de datos eficiente entre diferentes sistemas. Una API REST expone endpoints (puntos de acceso) que permiten a los clientes realizar operaciones específicas, los recursos disponibles en el servidor. Los recursos son entidades o conjuntos de datos que pueden ser accedidos o manipulados a través de la API.

La principal característica es su estado sin sesión, donde cada solicitud realizada tiene toda la información necesaria para completarla, dando interoperabilidad y separación entre el cliente y el servidor.

Hay tres conceptos clave para esta tecnología:

1. Las URL (Uniform Resource Locators) identifica el acceso a un recurso.
2. Los verbos HTTP que son una indicación de la acción en un proceso.
3. Los formatos de datos normalmente son JSON o XML.

Además, es importante la utilización correcta de los códigos de estado HTTP y la uniformidad de las interfaces.

Todo lo dicho proporciona una forma estándar y flexible de intercambiar datos entre sistemas. Su diseño se basa en principios simples pero potentes, convirtiendo la arquitectura en una opción común para el desarrollo de aplicaciones y servicios web.

5. SISTEMA DESARROLLADO

En este capítulo, se presenta en detalle el sistema desarrollado para el espejo inteligente, que busca mejorar la experiencia de los usuarios de habitaciones hoteleras, con aspectos clave para hacerlo único y eficiente. Además, se abordarán las decisiones de diseño tomadas durante su desarrollo en el hardware, la interfaz, el software, las comunicaciones con el exterior, los sensores y los actuadores, y la forma de obtener información.

5.1 DISEÑO HARDWARE

La configuración del hardware en este proyecto se compone de:

1. Raspberry Pi: Es el núcleo del sistema y se encarga de controlar todas las funcionalidades. Se conecta a través de los puertos GPIO que se utilizan para la comunicación con otros dispositivos permitiendo la interacción con sensores y actuadores externos.

2. Espejo inteligente: La pantalla se conecta a la Raspberry Pi mediante una interfaz de video, como micro HDMI o DisplayPort. Colocándose un espejo acrílico frente a la pantalla para reflejar la parte oscura y permitir la visión de las partes más brillantes.

3. Sensor de temperatura y humedad: El sensor se conecta a través de los pines GPIO. Utiliza un protocolo de comunicación específico, como I2C, para transmitir los datos al microcontrolador.

4. Sistema de iluminación: El control de persianas se realiza mediante motores de corriente continua y se conoce el estado de las luces por conexión con la red.

5. Cámara: Se utiliza para capturar los gestos realizados por el usuario conectándose el conector CSI (Camera Serial Interface), para capturar imágenes y enviarlas al programa de control de gestos.

Una vez analizado cada elemento elegido sin especificarlos concretamente, a partir de ahora nos basaremos en la implementación para las capacidades concretas y sus enlaces. En la Figura 1 vemos las conexiones de la Raspberry pi con el sensor de humedad y temperatura, el motor de corriente continua y su controlador de motores. Esas conexiones serían todas aunque falta la cámara que se conecta al CSI (AZ-Delivery, 2023), el HDMI y la alimentación de la Raspberry Pi USB-C.

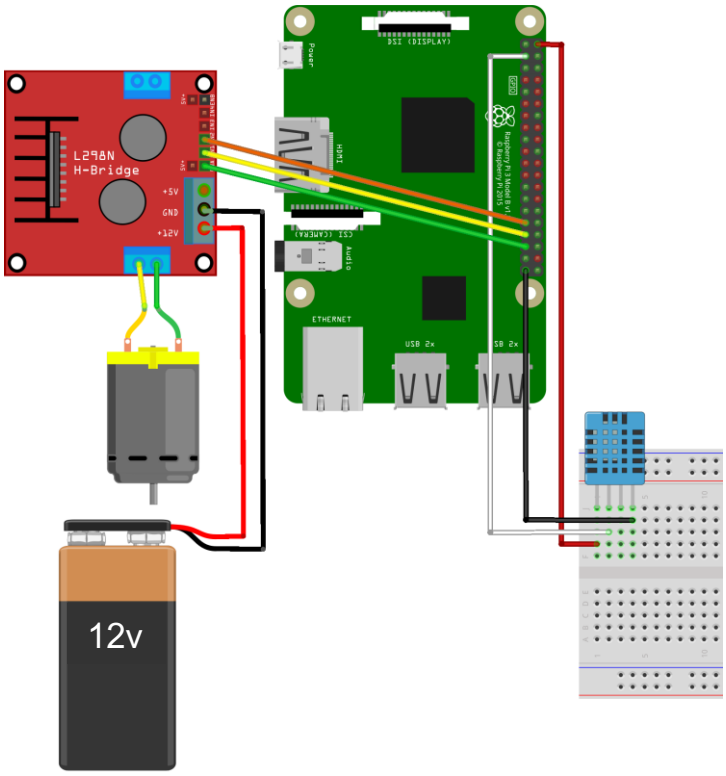

Figura 1. Conexión del motor con el controlador L298N y el sensor DHT11

Todas los enlaces se realizan en la Raspberry Pi 4 model B (no es exactamente el modelo que aparece en la Figura 1 aunque es muy parecido) y la distribución de los pines es la que aparece en la Figura 2.

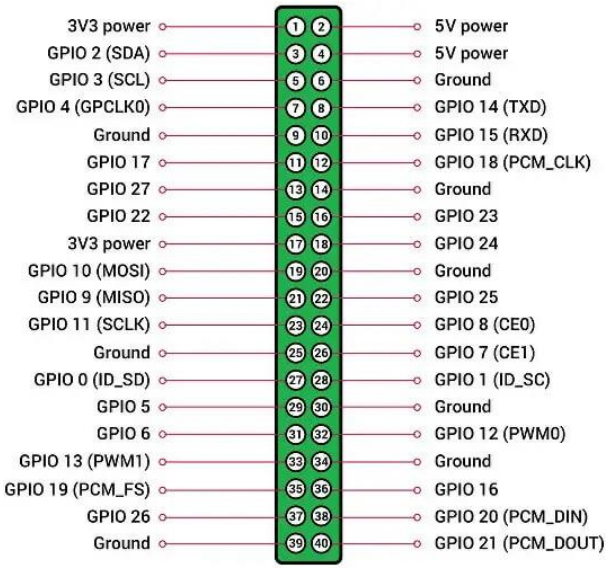

Figura 2. Asignación de pines GPIO Raspberry Pi 4

El sensor DHT11 (Temperature and Humidity Module, 2021) que como vemos en la Figura 3 se tiene que alimentar entre 3.5 V y 5.5 V, en nuestro caso con 5 V proveniente de uno de los pines (el 2 por ejemplo) de la placa. Los datos los da en serie bidireccional en un solo cable. La conexión con el sensor de temperatura y humedad DHT11 la recibe Raspberry Pi por el GPIO02 (SDA1, I^2C) siendo 2 bytes la humedad (1 byte con la parte entera del número y 1 byte con los decimales del número) y 2 bytes la temperatura (también 1 byte la parte entera y 1 byte la decimal).

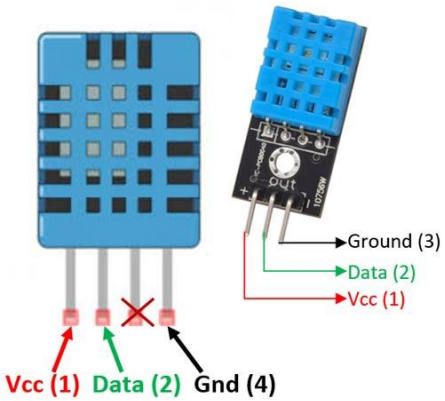

Figura 3. Asignación de pines del sensor DHT11

El motor de corriente continua necesita de un controlador de motores, en este caso un L298N (STMicroelectronics, 2000). La distribución como aparece en la Figura 4 necesita de

alimentación externa y tiene que habilitarse (Enable A / Enable B) para funcionar el control de los motores que recibe en los "Control inputs".

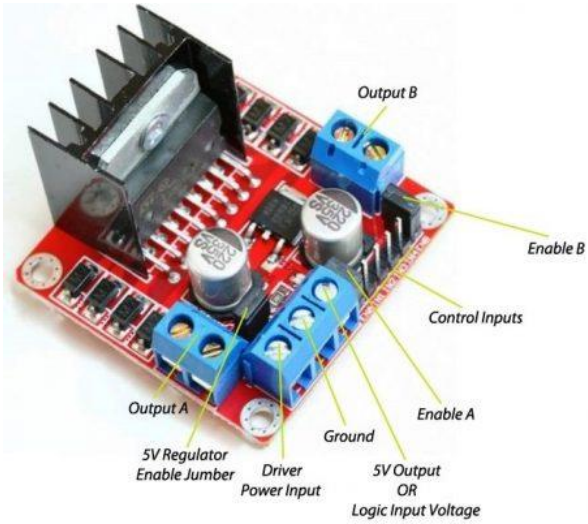

Figura 4. Interfaz del controlador de motores L298N

5.2 DISEÑO DE INTERFACES

Para no sobrecargar cada ventana con información, el espejo inteligente consta de tres pestañas donde la primera sirve de reposo mostrándose al inicio del programa, la siguiente es información interna de la habitación y la última externa del hotel. Esto se consigue con un módulo de terceros que permite rotar entre distintas pantallas (shbatm, 2022).

Un concepto clave en el diseño de la interfaz son los colores claros y oscuros por el funcionamiento del espejo acrílico que se coloca delante de la pantalla. Este tipo de espejo refleja luz de la zona con más iluminación a la de menor iluminación, en una dirección. Por lo tanto, si ves desde el lado con más luminosidad se reflejará la imagen, mientras que en el otro se podrá ver a través de manera translúcida como en la Figura 5. Esta propiedad óptica del espejo acrílico se logra con materiales y recubrimientos especiales que permiten el paso selectivo de la luz en una sola dirección.

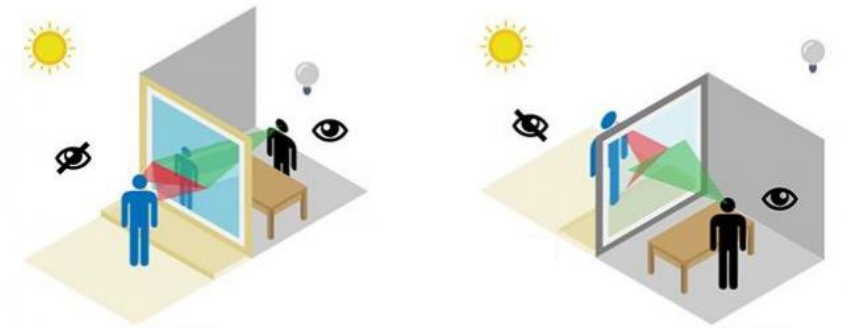

Figura 5. Espejo acrílico

Otra característica de todas las ventanas es el logo del hotel arriba a la izquierda que ha sido añadido en el propio CSS (lenguaje para presentación visual en HTML) general del espejo entero y el reloj en el centro superior que es un módulo por defecto de MagicMirror[2].

5.2.1 REPOSO

Hay cuatro elementos que caracterizan a esta vista que vemos en la Figura 6:

A. Los mensajes centrales que varían según la hora del día.

B. Día y hora.

C. Logo del Hotel.

D. El calendario de festivos de España.

E. Noticias: van rotando titulares del día de periódicos españoles.

F. Información del clima que a su vez contiene: arriba la hora del anochecer, a su izquierda el viento con su dirección, abajo del todo la sensación térmica y en el centro un icono basado en el tiempo con la temperatura.

G. Número de pestaña para facilitar el desplazamiento (shbatm, 2022).

Estos módulos han sido módulos por defecto de MagicMirror2 editados para tener información personalizada.

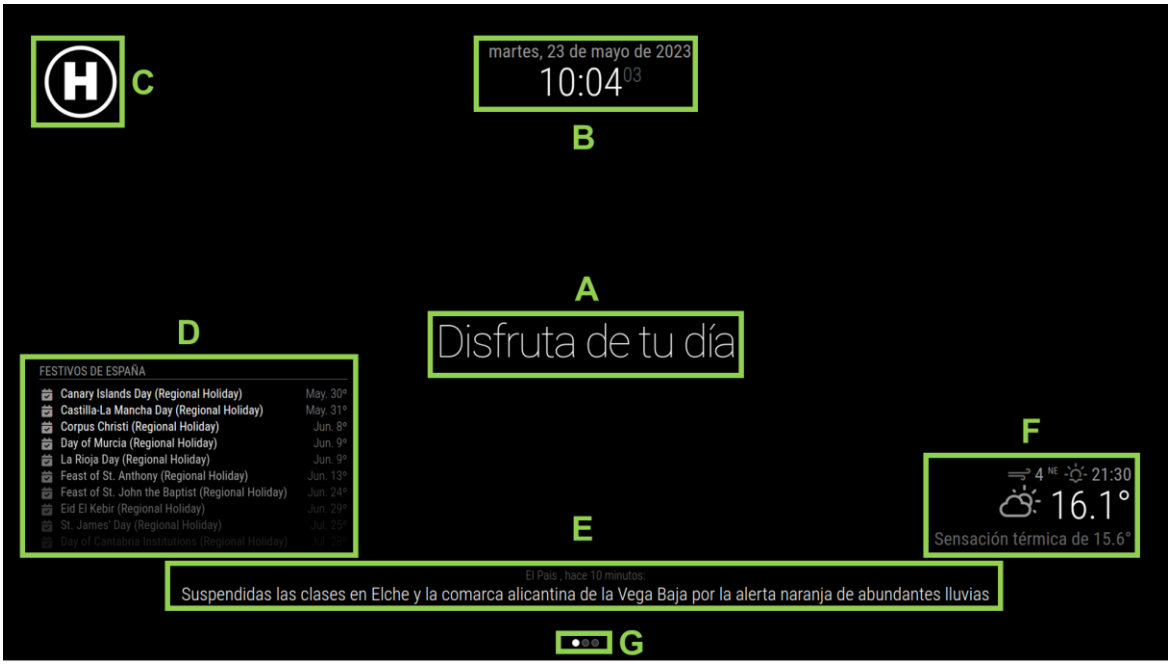

Figura 6. Ventana de reposo

5.2.2 ESTADO DE LA HABITACIÓN

En esta pestaña que se presenta en la Figura 7:

A. El sensor de humedad y temperatura DHT11 (grenagit, 2022).

B. Día y hora.

C. Logo del Hotel.

D. Información del Hue Bridge: muestra el estado de las bombillas (Schmidt, 2022) veremos de donde viene la entrada en el apartado del Estado de la iluminación..

E. El estado de la persiana: muestra información de un JSON (timdows, 2023) proveniente del programa Python (ANEXO II) que controla el motor.

F. Número de la ventana para facilitar el desplazamiento (shbatm, 2022).

Figura 7. Ventana con información del estado de la habitación

5.2.3 ESTADO DEL HOTEL

Esta vista como aparece en la Figura 8 :

A. Tabla en formato JSON: información económica de gasto de la habitación.

B. Día y hora.

C. Logo del Hotel.

D. Información del índice de calidad del aire (Gonzalez, 2022) que veremos en la parte de Índice de calidad del aire de donde se obtiene.

E. Tabla en formato JSON: el estado del aparcamiento

F. Tabla en formato JSON: la información económica de gasto de la habitación.

G. Número de pestaña para el desplazamiento.

Figura 8. Ventana con información del estado del hotel

5.3 *DISEÑO SOFTWARE*

Se ha realizado un diseño de software que abarca diferentes aspectos para asegurar un funcionamiento eficiente y una experiencia de usuario satisfactoria. El software funciona gracias a múltiples Entradas (como la calidad del aire en API REST, los gestos de la cámara o la temperatura del cuarto) esta información de entrada tiene tres posibles repercusiones: afectar a la interfaz de MagicMirror[2] desplazando las pestañas, editar la información de alguno de los módulos o dar una salida para controlar el motor de la persiana.

El funcionamiento para el usuario se basa en los gestos como se ve en el diagrama de flujos de la Figura 9. El archivo principal para esto es “reconoce_gestos_y_motor.py” ANEXO II que recibe la entrada por cámara de los gestos reconociendo el número de dedos (uno para desplazarse en MagicMirror[2] y dos dedos para controlar el motor) y la dirección del gesto (izquierda o derecha). Se consigue el movimiento del motor en el mismo archivo Python ANEXO II con el control de tres pines de la Raspberry Pi (lo veremos con más en detalle en) y el estado en el que se encuentra el motor se escribe en “Persiana.json” dentro de “\modules” lo cual lee el módulo “MMM-JsonTable” (timdows, 2023) y muestra en el espejo

los datos del JSON en este caso el estado de la persiana. El desplazamiento en cambio lo lleva a cabo el módulo "MMM-Carousel" (shbatm, 2022) con la simulación de la pulsación de la tecla de flecha derecha o izquierda que lleva a cabo el archivo Python ANEXO II utilizando la función "press()" de la librería pyautogui (Al Sweigart Revision, 2019).

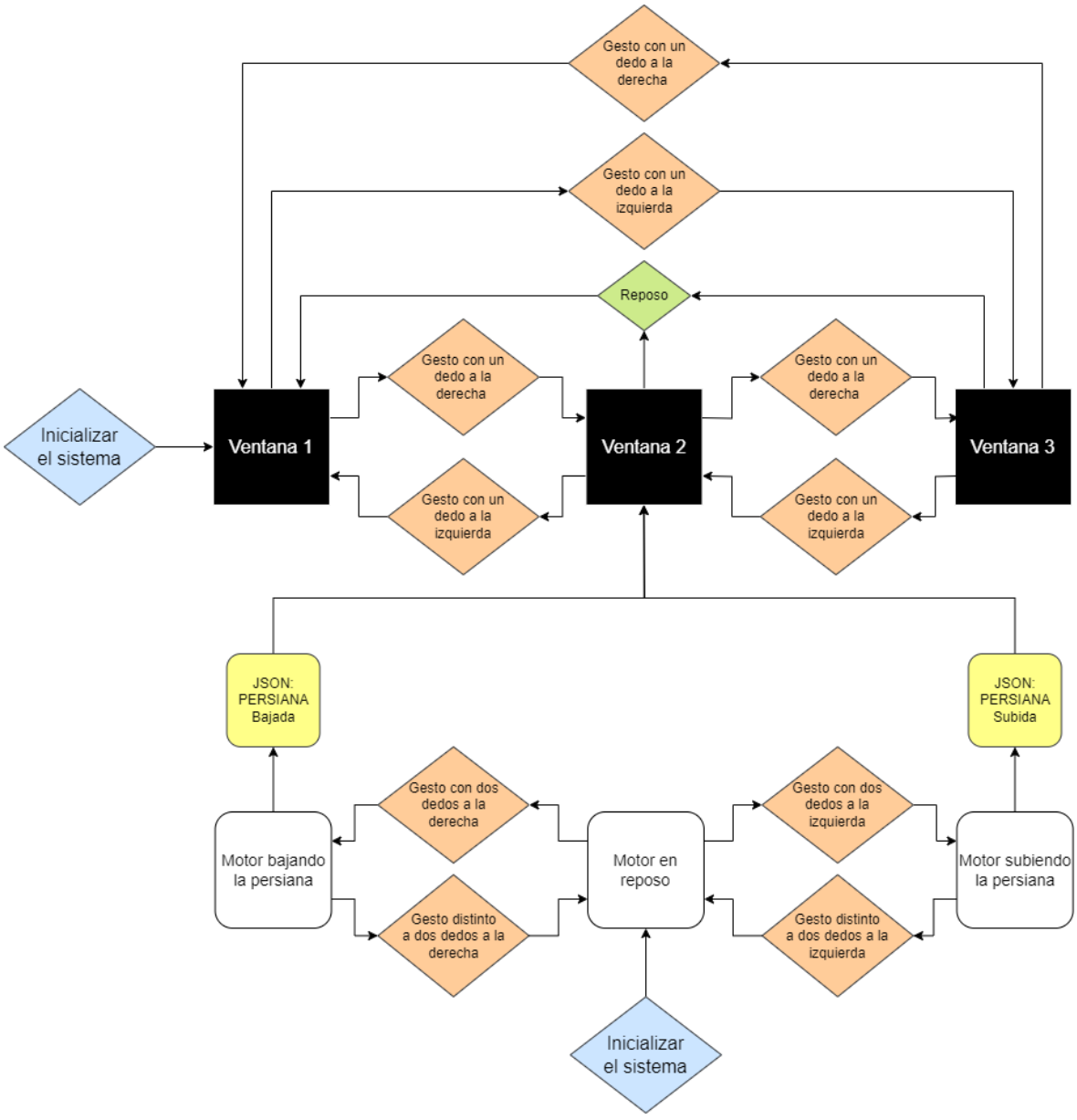

Figura 9. Diagrama de flujo del espejo inteligente

Además de los gestos, hay otras entradas que afectan al sistema, pero no las controla el usuario de la habitación, sino otras fuentes externas como los JSON con información del

hotel, los datos de la red o del sensor de temperatura y humedad. "MMM-JsonTable" (timdows, 2023) muestra esos archivos JSON, al igual que el del estado de la persiana, "MMM-DHT-Sensor" (grenagit, 2022) recibe y muestra la temperatura y humedad, y los encargados de la información de la red son: "newsfeed" (MagicMirror, 2023), "calendar" (MagicMirror, 2023), "weather" (MagicMirror, 2023) y "MMM-AQI" (Gonzalez, 2022).

Los módulos que componen este diseño de la plataforma MagicMirror^2 con la función que corresponde a cada uno de ellos lo vemos en la Figura 10. Los módulos que comienzan con "MMM-…" son desarrollados por terceros y los que no tienen ese prefijo son incluidos por defecto en MagicMirror^2.

Todos los módulos se llaman desde "config.js" ANEXO III un JavaScript que está en la carpeta "\config" de la plataforma MagicMirror^2 pudiendo configurarlos parcialmente y si es necesario una mayor personalización, cada módulo tiene su propia carpeta dentro del directorio "\modules" con su nombre y su propio JavaScript y CSS para poder editarlo con más profundidad. Los módulos que no empiezan por "MMM-…", es decir, los que soporta MagicMirror^2 por defecto, están con su nombre dentro de "\modules\default". Alguno de los módulos es más complejo y necesita múltiples archivos, pero se configuran igualmente desde el JavaScript con su nombre porque es el que llama "config.js". En general, con configurar correctamente "config.js" de "\config" es suficiente en el aspecto de JavaScript y no es necesario editar el de cada módulo, pero los tamaños de letra y otros detalles estéticos se tienen que realizar desde el CSS de cada módulo.

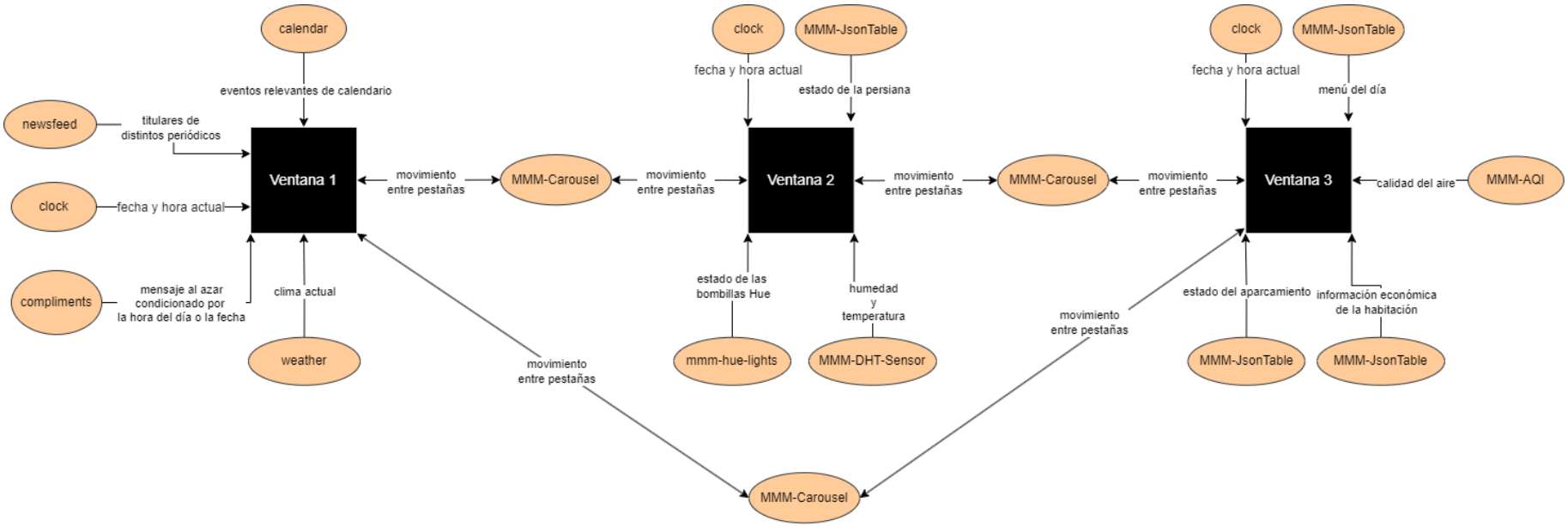

Figura 10. Módulos y su funcionalidad en MagicMirror

5.3.1 ENTRADAS

Se dividen en tres tipos, en la primera categoría hay dos entradas una de temperatura y otra de humedad que provienen del mismo sensor, luego la cámara para reconocer gestos y por último la distinta información que se obtiene por API REST u otros medios de la red.

5.3.1.1 Temperatura y humedad

Ya hemos visto en el inicio de Diseño Software el archivo "config.js" ANEXO III donde se llama a los distintos módulos de MagicMirror[2] y en este caso nos centramos en "MMM-DHT-Sensor". Los datos de temperatura y red los recibe el módulo de MagicMirror[2] MMM-DHT-Sensor desarrollado por terceros (grenagit, 2022) que utiliza la librería de C bcm2835 para leer la entrada digital en I^2C (McCauley, 2021) y se muestra en el Diseño de interfaces gracias a las funciones de JavaScript.

En concreto, se genera una estructura HTML representando el contenido visual del módulo en el espejo inteligente. En este caso, crea una tabla que muestra la temperatura y humedad, utilizando iconos y estilos diferentes dependiendo de los valores. Estos datos se obtienen con la siguiente comunicación con el socket:

1. El módulo envía una notificación con un GET al socket dando al servidor los parámetros 'sensorPin' y 'sensorType' (el 11 de DHT11).
2. El servidor procesa la notificación y envía una respuesta a través de con los datos.
3. El módulo recibe la respuesta en 'socketNotificationReceived' y llama a 'processSensorData' para procesar y mostrar los datos del sensor.

Ahora que sabemos cómo funciona el módulo, a la hora de configurarlo se puede editar su aspecto en el CSS del directorio del módulo y desde el "config.js" general de MagicMirror[2] se configura el aspecto, las unidades de medida que usa o los tiempos de carga y animación. En este caso no ha sido necesario editar el archivo "MMM-DHT-Sensor.js" para personalizar el modulo en mayor profundidad.

5.3.1.2 Cámara

Toda la configuración y el uso de la información de la cámara se desarrolla en el archivo "reconoce_gestos_y_motor.py" que vemos en el ANEXO II. La cámara es la Raspberry Pi V1.3 (AZ-Delivery, 2023) y recibe su imagen la librería OpenCV (doxygen, 2022) por la clase VideoCapture que, como indica su nombre en inglés, captura el video de la cámara. Sobre esta captura del video se mapean con MediaPipe reconocimiento de gestos los puntos necesarios para saber la posición de cada uno de los dedos. Un punto tiene su ubicación dentro de la imagen y distintas conexiones para formar una mano de forma esquemática, como vemos en la Figura 11. El número de puntos de cada parte de la mano está definido y se compone de: las coordenadas de posición de 3 puntos para cada uno de los dedos de la mano, excepto el pulgar que necesita de 4 puntos y la palma de la mano que se define adecuadamente con 5 puntos de donde se desglosa cada uno de los dedos.

Figura 11. Captura de la cámara con mapeo de la mano

Como hemos explicado en el inicio del Diseño Software el número de dedos condiciona el control: con un dedo se desplazan las pantallas de MagicMirror[2] y con dos dedos se dirige el motor de la persiana. El número de dedos extendidos que reconoce MediaPipe reconocimiento de gestosy la posición del dedo, permiten que el condicional "if" (ANEXO II) diferencie entre los lados a los que se deslizan las ventanas del espejo inteligente o la dirección de rotación del motor. Esto se consigue gracias a la información de la posición del dedo en el eje X que se puede comparar con un parámetro arbitrario en ese mismo eje X que limite el lado de la izquierda y de la derecha, para así saber la posición relativa a la cámara del punto del dedo extendido que estamos intentado acotar en esos tres puntos. En la Figura

12 las líneas marcan los límites donde se pasa a considerar gesto a la derecha o a la izquierda. La comunicación con el motor la veremos en detalle en el apartado del Motor de persiana.

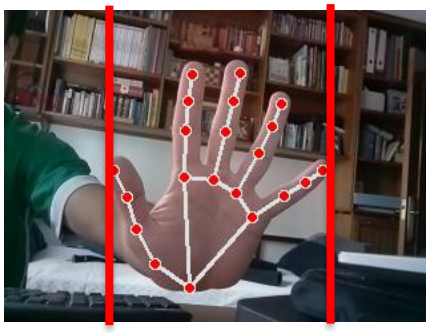

Figura 12. Captura con referencias del lado izquierdo y derecho

5.3.1.3 Información de la red

Hay cinco módulos que obtienen información de la red.

5.3.1.3.1 Estado del clima

La configuración del módulo "weather" (MagicMirror, 2023) se lleva a cabo desde "config.js" ANEXO III y además se hizo un pequeño ajuste en el código de "weather.js" del tamaño de los iconos que acompañan la temperatura. En "config.js" se establece la posición en la pestaña del módulo, el proveedor de la información del tiempo (entre las opciones que da el módulo se ha elegido "openweathermap"), el tipo de información del clima si es actual, de cada hora, cada día o la predicción del día, la localización y el token del API key.

El estado del clima actual se obtiene de OpenWeather donde al crear una cuenta conseguimos una API key como en la Figura 13. Esto permite dar las condiciones climáticas actuales al módulo del espejo inteligente. Para mostrar el tiempo en la localidad deseada se puede ver la identificación (por ejemplo en Madrid es 3117735) en el documento que aparece se descarga en el siguiente enlace: http://bulk.openweathermap.org/sample/city.list.json.gz. Este servicio tiene limitaciones al número de llamadas por minuto, variando según el servicio que se haya contratado en la web (OpenWeather, 2023).

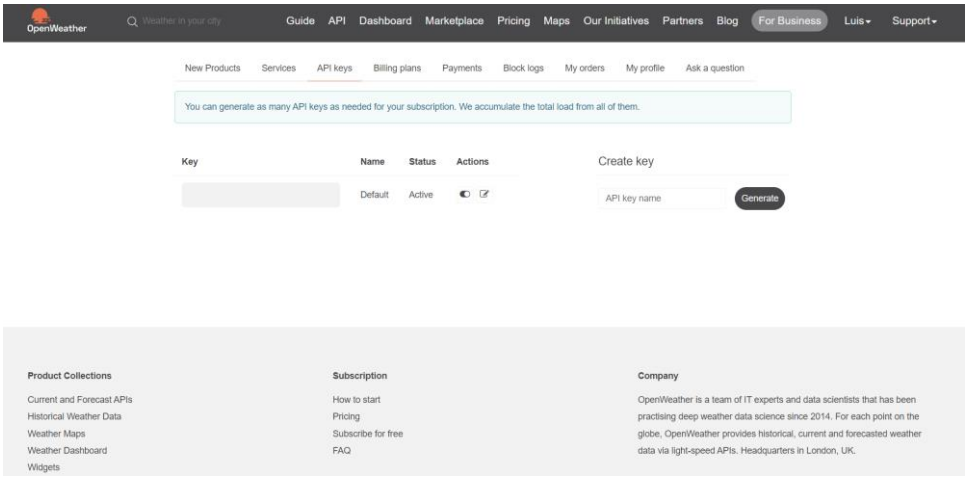

Figura 13. API keys de mi cuenta en OpenWeahter

5.3.1.3.2 Periódicos

El módulo "newsfeed" (MagicMirror, 2023) se configura en "config.js" ANEXO III con la posición en la ventana del módulo, las fuentes de información de las noticias que en este caso son múltiples y opciones de aspecto como enseñar el título, la fecha o las actualizaciones de los titulares.

Las noticias se reciben aprovechando la arquitectura API REST para cada uno de los periódicos recibiendo en formato XML todo lo relacionado con cada artículo de prensa. Los cuatro utilizados son El País (El País, 2023), ABC (ABC, 2023), El Mundo (El Mundo, 2023)y ElDiario.es (ElDiario.es, 2023). Un ejemplo del aspecto de estas páginas es la Figura 14.

Figura 14. Página con la portada de El País en XML

5.3.1.3.3 Índice de calidad del aire

El módulo "MMM-AQI" (Gonzalez, 2022) desarrollado por terceros, se configura en "config.js" ANEXO III: el título que aparece en la interfaz, la posición del módulo y la configuración de la fuente de la información con el token de la API que obtiene los datos, la ciudad de los datos que se quieren mostrar, habilitar los índices individuales de todos los distintos contaminantes y los tiempos de espera, de actualización de los datos y de duración de la animación de actualización.

El índice de calidad del aire proviene del Air Quality Open Data Platform donde solicitas una API key (The World Air Quality Index Project Team, 2023), te envían un correo de confirmación que te lleva a la página que podemos ver en la Figura 15 donde se encuentra el token y un ejemplo del formato del JSON recibido al hacer un GET. El módulo también necesita saber la ciudad de donde mostrar los datos con el nombre es suficiente, aunque a veces puede dar problemas y hay que añadir la id de la ciudad.

Figura 15. Plataforma de datos de calidad del aire Confirmación de token de API

5.3.1.3.4 Calendario

"Calendar" (MagicMirror, 2023) módulo por defecto de MagiMirror[2] configurado en "config.js" ANEXO III con el título que aparece del módulo en este caso "Festivos de España", la posición en el diseño y opciones de la fuente de información con su URL para llegar a los datos y el símbolo que acompaña a cada evento del calendario que se puede elegir entre distintas opciones.

Los datos del calendario de festivos, en este caso el de España, tiene que seguir el estándar iCalendar (B. Desruisseaux, 2009) en formato ics para funcionar adecuadamente y se obtiene de una url, por ejemplo: https://www.officeholidays.com/ics-all/spain. Se podría editar por otro calendario cualquiera ya que el formato ics es utilizado por múltiples plataformas.

5.3.1.3.5 Estado de la iluminación

La información de la bombilla hue en verdad lo recibe del Hue bridge, que es el sistema que permite controlar y conectar con hasta 50 luces. Para tener acceso a esta información primero se necesita una cuenta de desarrollador de Hue (Philips, 2022).

Luego se requiere la dirección IP del Hue bridge que la hemos obtenido desde una aplicación móvil encontrando en la red su mDNS como se ve en la Figura 16.

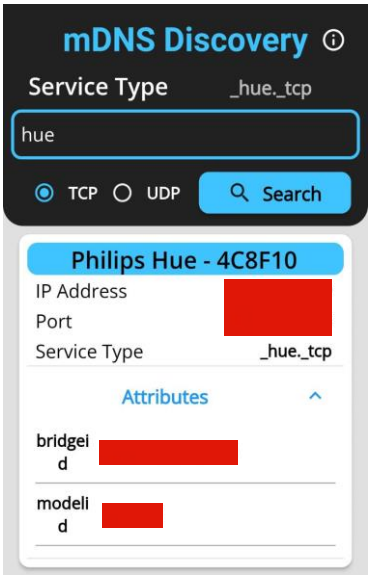

Figura 16. mDNS Discovery aplicación para obtener el bridge id

Por último, el usuario autorizado para el uso del Hue Bridge esto se obtiene con un POST de API REST en https://<bridge ip address>/debug/clip.html dándonos el username, por ejemplo la Figura 17.

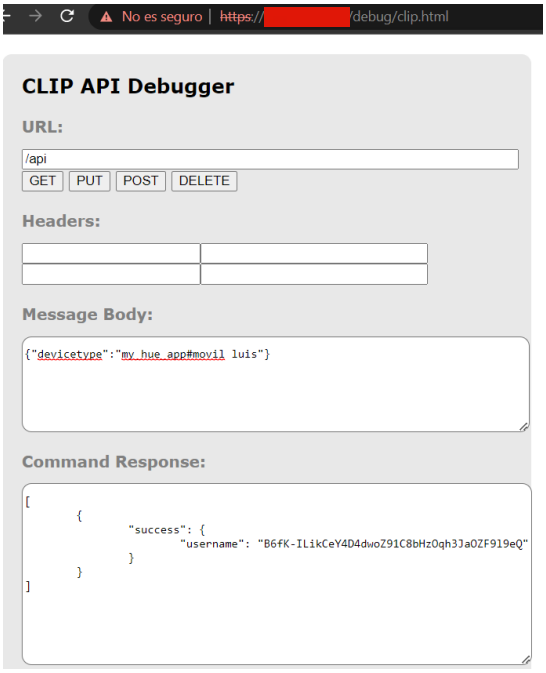

Figura 17. Aplicación test para la API de Philips

Obtenemos esta información para que el módulo puede mostrar en tiempo real el estado de las luces con el color y la intensidad de brillo de cada una. El módulo desarrollado por terceros que soporta esto es "mmm-hue-lights" (Schmidt, 2022) que se llama y configura desde "config.js" ANEXO III la posición del módulo en la ventana y ip del Hue Bridge con el usuario autorizado que obtuvimos.

5.3.2 CONTROL

Hay dos partes esenciales que se controlan por parte del usuario con gestos para usarlo sin tener que pulsar botones o tocar la pantalla del espejo. Ambas funciones de control se llevan a cabo desde reconoce_gestos_y_motor.py en Python (ANEXO II), como hemos visto en las entradas de Cámara.

Primero vamos a explicar el desplazamiento entre pestañas, para esto se utiliza la cámara extendiendo un dedo de tu mano y moviéndolo hasta la zona izquierda o la zona derecha de la imagen que recibe el programa. Esto es equivalente a pulsar la tecla de flecha izquierda o derecha, por eso mismo el módulo de MagicMirror[2] no tiene problema de compatibilidad

para recibir esta información, ya que está diseñado para funcionar con teclado (shbatm, 2022).

El control del movimiento del motor de la persiana es parecido para el usuario, siendo la única diferencia tener que extender dos dedos en vez de uno y luego, de forma equivalente al deslizamiento entre ventanas, desplazar los dedos a derecha o a izquierda. Cabe señalar que el movimiento del motor lo explicaremos en el apartado de . El programa Python tras el movimiento del motor edita un archivo JSON gracias a una librería que explicaremos en Motor de persiana actualizando el JSON que muestra el MagicMirror[2] con la información del estado de la persiana.

Estas funciones se ven en la Figura 18 de forma gráfica.

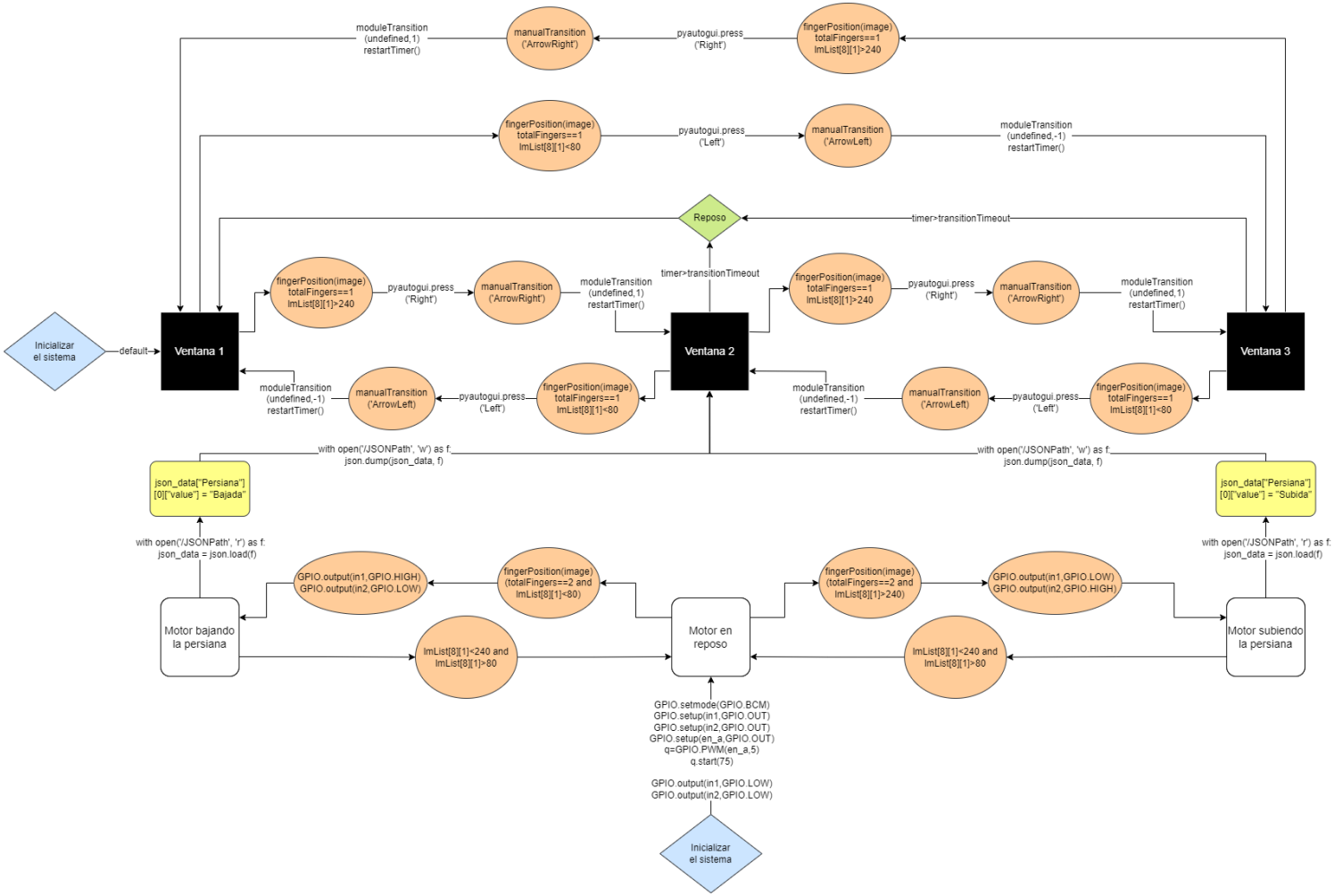

Figura 18. Diagrama de flujo funciones de control

5.3.3 SALIDAS

La salida del producto diseñado es el envío de información a las interfaces y el control de la iluminación mediante dos medios, las bombillas de Philips Hue, que se pueden controlar desde su aplicación con el Hue Bridge, y el motor de la persiana que como hemos visto funciona por corriente continua.

5.3.3.1 Interfaces

La salida del espejo inteligente se basa en múltiples funciones que se resumirían en la Figura 19, la Figura 20 y la Figura 21 que componen el flujo de cada ventana sin tener en cuenta el desplazamiento entre ventanas. Tener en cuenta que todos los módulos son llamados desde "config.js" ANEXO III y las funciones que se representan en las figura forman parte de los archivos de configuración de JavaScript que forman parte de cada uno de estos módulos desarrollados por terceros, que resultan de interés pero no se han desarrollado para este proyecto.

En todas las ventanas se tiene el logo del hotel ya que forma parte del fondo de MagicMirror[2] que está definido en el archivo CSS. El otro elemento común a todas las ventanas es el módulo "clock" (MagicMirror, 2023) que se muestra con la versión por defecto y situado en el centro superior como se define en la configuración general del ANEXO III.

En total se tiene el módulo "MMM-JsonTable" (timdows, 2023) cuatro veces en la Ventana 2 una vez y tres en la Ventana 3. Este módulo comienza con "start" inicializando el módulo y llamando a "scheduleUpdate" que estará de forma periódica llamando a "getJson". "getJson" envía la solicitud "MMM-JsonTable_GET_JSON" y en "socketNotificationReceived" se recibe si hace falta actualizar la información con "updateDom" o se finaliza el proceso.

En la Figura 19 vemos las distintas funciones de los módulos de MagicMirror[2] que componen la Ventana 1:

1. "compliments" (MagicMirror, 2022) aparece en la parte inferior de la figura y antes de hablar de sus funciones cabe señalar que la configuración es la predeterminada en

el ANEXO III, pero en la configuración de "compliments.js" las frases y las horas son personalizadas pensando en el público de un hotel en España. Tras aclarar la configuración, la función principal del módulo es componer un conjunto de complementos que aparecen aleatoriamente. Ese conjunto depende de la hora del día, del día en concreto y otras frases que pueden aparecer de forma indiferente a cualquier condición.

2. El módulo "weather" (MagicMirror, 2023) se inicia configurando variables y parámetros en "start", tras esto "getParams" construye la URL de solicitud al servidor de OpenWeatherMap y la solicitud se realiza en "updateWeather". Los datos recibidos de la solicitud se procesan para guardar los valores relevantes en "processWeather" y con esto en "getDom" se genera la estructura HTML que se mostrará, en formato DOM .
3. La primera función del módulo "calendar" (MagicMirror, 2023) es "start" donde se configura el calendario y se añaden los eventos al objeto "calendarData" gracias a "addCalendar". La información de "calendarData" se actualiza con lo recibido por "socketNotificationReceived" que viene al recibir una notificación del backend del módulo. Con "calendarData" "getDom" muestra en HTML la información.
4. El módulo "newsfeed" (MagicMirror, 2023) a su vez se inicializa con "start" y si "socketNotificationRecieved" recibe la notificación adecuada del backend, llega a "notificationRecieved" toda la información necesaria para seleccionar el artículo adecuado que luego "generateFeed" termina de formar en una lista adecuada de noticias para mostrar.

La Figura 20 representa las funciones de los módulos de MagicMirror[2] en la Ventana 2:

1. El módulo "mmm-hue-lights" (Schmidt, 2022) comienza al inicializarse con "start" lo que lleva a "getData" con la información de las luces de Hue que se procesa en "processHueData". Dependiendo del formato si es una lista o una cuadrícula se lleva a "renderList" o a "renderGrid", en ambos casos acaba el flujo

con "getDom" que genera en formato DOM lo necesario para la visualización en HTML

2. El módulo "MMM-DHT-Sensor" (grenagit, 2022) al igual que muchos de los otros módulos se inicializa con la función "start" y llama a "updateSensorData" que será luego llamada de forma periódica por "scheduleUpdate". "updateSensorData" realiza una llamada a "socketNotificationReceived" que recibe la notificación del backend, dando los datos a "processSensorData" que procesa la información anterior y termina en "getDom" que muestra en el espejo inteligente.

La Figura 21 contiene el diagrama de flujos con las funciones de los módulos en la Ventana 3:

1. Solo "MMM-AQI" (Gonzalez, 2022) no ha sido explicado antes ya que el resto son módulos que muestran tablas en formato JSON. La primera función es "start" llamando a "updateAQI" que será luego llamada por "scheduleUpdate" cada vez que pase cierto delay. "updateAQI" llama a "socketNotificationReceived" recibe la notificación del backend. Los datos van a "processSensorData" procesando la información de los contaminantes e indicadores y termina en "getDom" que se verá en la interfaz.

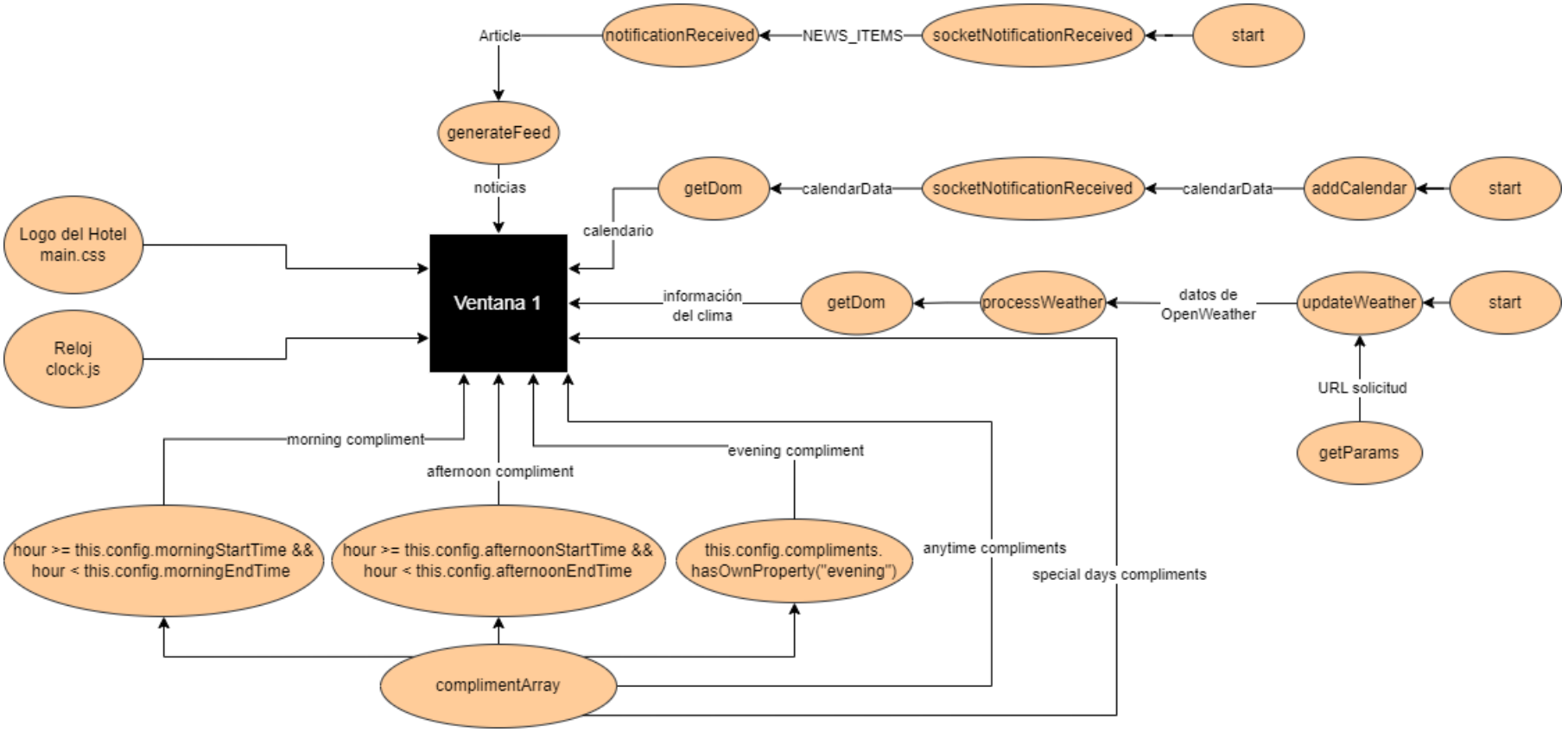

Figura 19. Diagrama de flujos Funciones de la Ventana 1

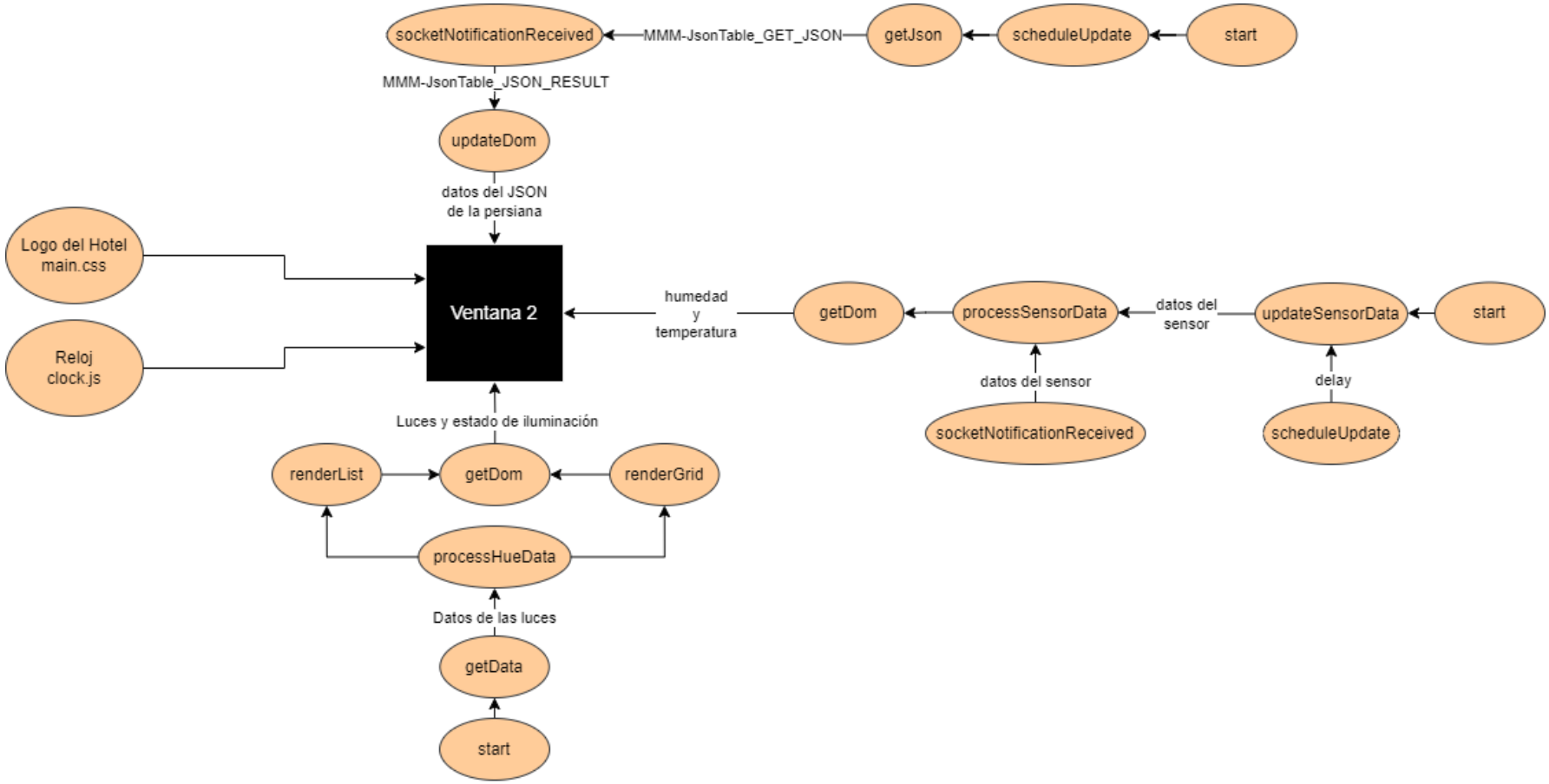

Figura 20. Diagrama de flujos Funciones de la Ventana 2

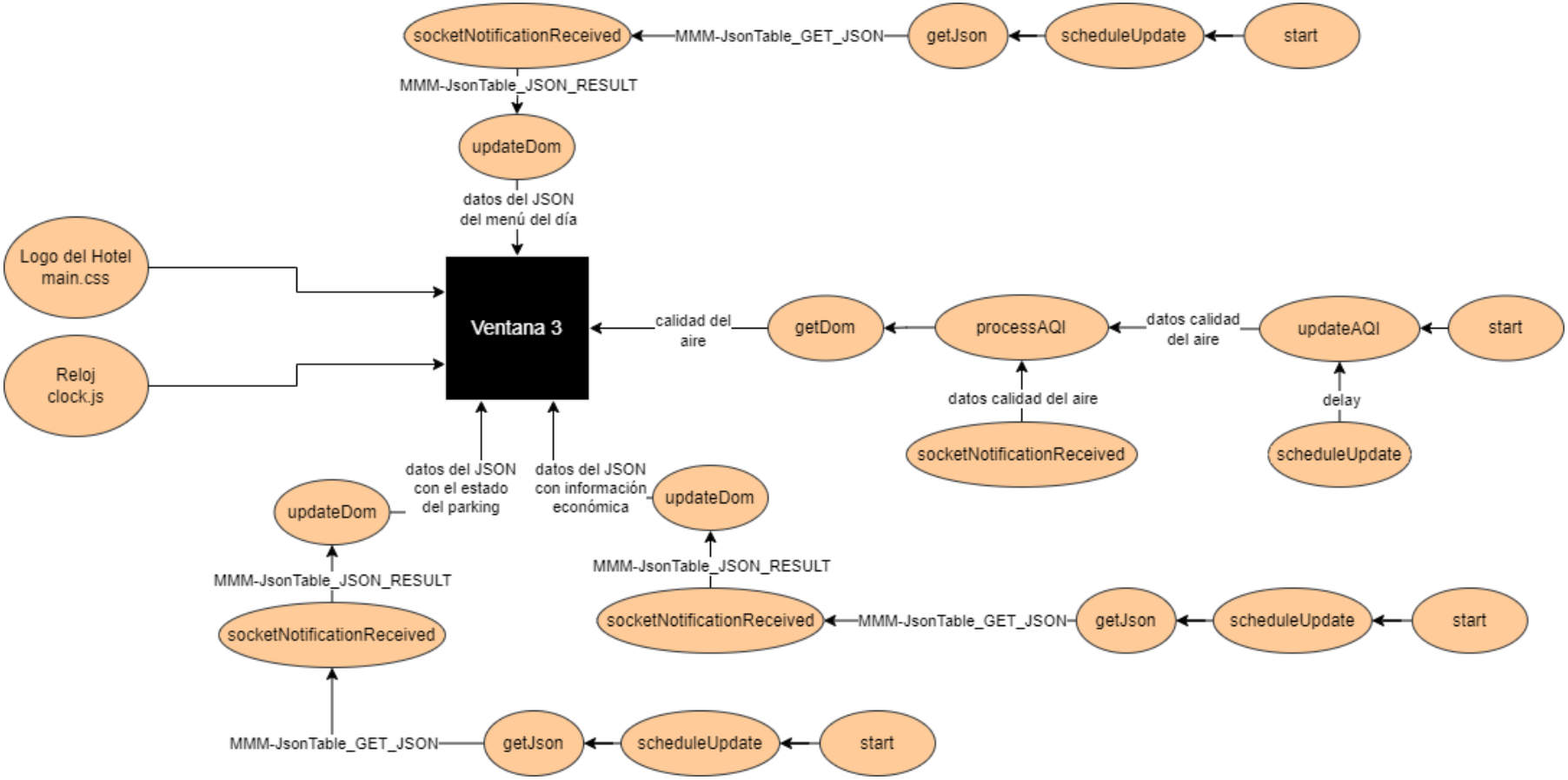

Figura 21. Diagrama de flujos Funciones de la Ventana 3

5.3.3.2 Motor de persiana

En el contexto de este espejo inteligente, el movimiento del motor se lleva a cabo gracias a la librería RPi.GPIO de Python que permite controlar los pines GPIO estableciendo el modo de funcionamiento de los pines (entrada o salida) (Python Software Foundation, 2022). Esto se lleva a cabo leyendo el estado de un pin de entrada o controlando el estado de un pin de salida. En este caso solo queremos controlar los pines de salida para comunicar desde la Raspberry Pi al controlador del motor cuando queremos mover la persiana y la dirección del movimiento.

Hay tres pines que en este caso se usaran como pines de salida. Uno para habilitar (enable) el cual lo requiere el controlador del motor y solo tiene que inicializarse con una señal pwm, que se genera gracias a la función “PWM” que tiene la librería RPi.GPIO (Python Software Foundation, 2022), para poder luego usar los otros dos pines. Estos otros dos pines tienen tres combinaciones posibles in1 “LOW” in2 “LOW” que indica el reposo del motor, in1 “HIGH” in2 “LOW” que en este caso sería la subida de la persiana y por último in1”LOW” in2 “HIGH” que sería la bajada. Esto se ve en el código de Python del ANEXO II donde los tres últimos condicionales “if” contienen el control de estos pines haciendo esto mismo.

6. ANÁLISIS DE RESULTADOS

6.1 ANÁLISIS DEL SISTEMA Y DISEÑO

El diseño general se ha estudiado en apartados anteriores, pero para llegar a resultados se ha tenido que hacer un análisis más concreto de las partes. El cableado para la conexión de todas las partes se ha hecho con material del laboratorio que no es el ideal para un negocio de hostelería donde las distancias entre partes como la cámara y el motor de la persiana tendrán que ser mayores, pero es suficiente para simularlo. La cámara tiene su puerto dedicado como ya hemos visto y aparece en el montaje de la Figura 22.

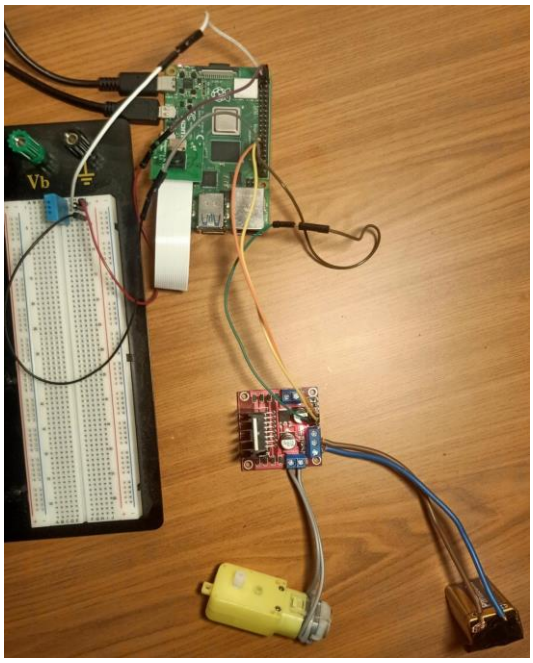

Figura 22. Montaje del espejo inteligente

La comunicación entre el programa de Python y el MagicMirror[2] se realizó satisfactoriamente gracias a simular el pulsar la tecla de la flecha izquierda o derecha que se hace en Python con la librería Pyautogui (Al Sweigart Revision, 2019). La implementación del proyecto se muestra en el siguiente video:

https://docs.google.com/document/d/1EnG2S1bKuv0_Hw9tKUzqve7NU_dNXZ_stD9Tc2m3xMs/edit?usp=sharing

6.2 IMPLEMENTACIÓN

Se ha llevado a cabo la implementación del sistema en un monitor Philips 243V7QDSB/00 de cómo se ve en funcionamiento en la Ilustración 3 que no sería recomendable para un hotel al tener un soporte que se tendría que quitar para poder instalarlo, por ejemplo en el espejo del baño o de la entrada de un cuarto, igualmente para comprobar el funcionamiento del sistema solo se necesita por parte de la pantalla una conexión de video con la Raspberry Pi 4 Model B.

El sensor DHT11 funciona perfectamente, al igual que el controlador del motor L298N y el motor de corriente continua. En cambio, la cámara tiene una resolución limitada (AZ-Delivery, 2023) y con los distintos programas en Raspberry Pi al mismo tiempo se congela la imagen de la cámara, ocasionando un mayor tiempo para el reconocimiento de gestos. Esto se podría mejorar con una mejor cámara que requiera menor consumo de RAM para la Raspberry Pi.

7. CONCLUSIONES Y TRABAJOS FUTUROS

Los objetivos propuestos al principio han sido llevados a cabo, ya que el proyecto busca adaptarse a los intereses comerciales del sector de la hostelería. Se cumple con el propósito de mostrar la información relevante para el huésped en un espejo inteligente y el control del motor de la persiana. La implementación del sistema se puede ajustar a distintas necesidades, dando mayor flexibilidad para los hoteles y las distintas habitaciones donde se instale. Necesitar un sistema de alimentación externa para el controlador del motor complica la instalación y el mantenimiento, esto se podría suplir pero haría falta un controlador diferente u otro sistema de alimentación.

Hay múltiples posibles mejoras para este prisma de la domotización de cuartos de hotel, pero principalmente una mejor monitorización del sistema de luces y poder recibir de sensores situados en el hotel o de sistemas inteligentes la información del hotel que en este proyecto se muestra en tablas JSON en el espejo inteligente. Esta última mejora sería interesante para dar un servicio más personalizado para cada hotel dependiendo de los distintos servicios que oferten y de su infraestructura.

Se podrían mandar mensajes al usuario de la habitación con múltiples detalles relevantes como por ejemplo de un nivel dañino en el índice de calidad del air, o con alta temperatura en el cuarto. Existe también la opción de poder realizar más acciones, como poder llamar al servicio del hotel.

8. BIBLIOGRAFÍA

[1] AZ-Delivery. (2023). *Raspberry Pi V1.3 Cámara Hoja de datos*. Obtenido de https://cdn.shopify.com/s/files/1/1509/1638/files/Kamera_fur_Raspberry_Pi_V1.3_Datenblatt.pdf?5797214750144745094

[2] ABC. (2023). Obtenido de https://www.abc.es/rss/feeds/abcPortada.xml

[3] Al Sweigart Revision. (2019). *PyAutoGUI's documentación*. Obtenido de https://pyautogui.readthedocs.io/en/latest/

[4] *ArabaDomotic*. (s.f.). Recuperado el 13 de 11 de 2022, de Domótica Hoteles: https://arabadomotic.com/domotica-hoteles/

[5] AZDelivery. (23 de Octubre de 2016). *AZDelivery Camara Raspberry Pi, 5 Megapixeles, Sensor Ov5647 1080p, Modulo Cámara Video Soporte Vision con Cable Flexible 15 cm Compatible con Raspberry Pi con E-Book Incluido!* Obtenido de https://www.amazon.es/AZDelivery-c%C3%A1mara-para-Raspberry-Pi/dp/B01M6UCEM5/ref=d_pd_sbs_sccl_2_1/257-1044547-6103539?pd_rd_w=UiB6c&content-id=amzn1.sym.71bd5875-7b22-4aa5-b52a-39d0a79d59c6&pf_rd_p=71bd5875-7b22-4aa5-b52a-39d0a79d59c6&pf_rd_r=QTN2E09Z0ZP

[6] B. Desruisseaux, E. O. (Septiembre de 2009). *iCalendar (RFC 5545)*. Obtenido de https://icalendar.org/RFC-Specifications/iCalendar-RFC-5545/

[7] BricoGeek. (s.f.). Obtenido de https://tienda.bricogeek.com/sensores-temperatura/986-sensor-de-humedad-y-temperatura-dht11.html?srsltid=AR57-fBvRabde_dofroIbhhLRvuoHLwVk2zFKUtYkZk7xn5AL6wSi5trUTU

[8] doxygen. (28 de Diciembre de 2022). *OpenCV 4.7.0*. Obtenido de https://docs.opencv.org/4.7.0/

[9] El Mundo. (2023). Obtenido de https://e00-elmundo.uecdn.es/elmundo/rss/portada.xml

[10] El País. (2023). Obtenido de https://feeds.elpais.com/mrss-s/pages/ep/site/elpais.com/portada

[11] ElDiario.es. (2023). Obtenido de https://www.eldiario.es/rss

[12] *Electron Documentación*. (2023). Obtenido de https://www.electronjs.org/docs/latest/

[13] Fielding, R. T. (2000). Architectural Styles and the Design of Network-Based Software Architectures. *Association for Computing Machinery*, 162. Obtenido de https://dl.acm.org/doi/book/10.5555/932295

[14] *GFPLab*. (2018). Recuperado el 13 de 11 de 2022, de Hospitality automation: https://www.gfplab.com/domotics/hotel-domotics/?lang=en

[15] Gonzalez, R. (25 de Abril de 2022). *MMM-AQI*. Obtenido de https://github.com/ryck/MMM-AQI

[16] grenagit. (23 de Junio de 2022). *MMM-DHT-Sensor*. Obtenido de https://github.com/grenagit/MMM-DHT-Sensor

[17] *Hotel-suppliers*. (2022). Recuperado el 13 de 11 de 2022, de Smart automation solutions: https://www.hotel-suppliers.com/supplier/livmark-hotel-smart-mirrors/

[18] Kouao, E. (21 de 5 de 2020). *Hackster*. Recuperado el 13 de 11 de 2022, de Smart Mirror Touchscreen (with Face Recognition): https://www.hackster.io/eben-kouao/smart-mirror-touchscreen-with-face-recognition-6c84cc

[19] *Livmark*. (2021). Recuperado el 13 de 11 de 2022, de Smart Mirrors Solution for Hotels: https://smartmirror.livmark.hr/hotel-smart-mirror/

[20] MagicMirror. (22 de Septiembre de 2022). *compliments*. Obtenido de https://docs.magicmirror.builders/modules/compliments.html

[21] MagicMirror. (8 de Abril de 2023). *Calendar*. Obtenido de https://docs.magicmirror.builders/modules/calendar.html

[22] MagicMirror. (4 de Abril de 2023). *Clock*. Obtenido de https://docs.magicmirror.builders/modules/clock.html

[23] MagicMirror. (14 de Enero de 2023). *News Feed*. Obtenido de https://docs.magicmirror.builders/modules/newsfeed.html

[24] MagicMirror. (4 de Abril de 2023). *Weather Module*. Obtenido de https://docs.magicmirror.builders/modules/weather.html

[25] *MagicMirror² Documentación*. (14 de Enero de 2023). Obtenido de https://docs.magicmirror.builders/

[26] McCauley, M. (30 de Marzo de 2021). *C library for Broadcom BCM 2835 as used in Raspberry Pi*. Obtenido de http://www.airspayce.com/mikem/bcm2835/

[27] *Mediapipe Guía de soluciones*. (9 de Mayo de 2023). Obtenido de https://developers.google.com/mediapipe/solutions/guide

[28] *MGelectricidad*. (2022). Recuperado el 13 de 11 de 2022, de Domotic systems in home, hotels and companies: https://www.mgelectricidad.com/services/domotic-systems/

[29] *Microdevice*. (2022). Recuperado el 13 de 11 de 2022, de Hotel Supervisor: https://microdevice.com/en/system/hotel-supervisor

[30] OcioDual. (3 de Octubre de 2019). *OcioDual Controlador L298N Motores DC PAP Driver Stepper Doble puente H para Electrónica Robótica Proyectos Raspberry PIC AVR*. Obtenido de https://www.amazon.es/OcioDual-Controlador-Motores-Driver-Stepper/dp/B07YNR5KWP/ref=asc_df_B07YNR5KWP/?tag=googshopes-21&linkCode=df0&hvadid=628580889748&hvpos=&hvnetw=g&hvrand=16591132038595559796&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocp

[31] OpenWeather. (2023). *OpenWeather*. Obtenido de https://openweathermap.org/

[32] Philips. (2022). *Philips Hue Developer*. Obtenido de https://developers.meethue.com/

[33] Python Software Foundation. (6 de Febrero de 2022). *A module to control Raspberry Pi GPIO channels*. Obtenido de https://pypi.org/project/RPi.GPIO/

[34] Python Software Foundation. (25 de Mayo de 2023). *JSON encoder and decoder*. Obtenido de https://docs.python.org/3/library/json.html

[35] raspipc.es. (2009). *Kit de inicio Raspberry Pi4 modelo 1GB con todo lo necesario.* Recuperado el 25 de Mayo de 2023, de

https://www.raspipc.es/index.php?ver=tienda&accion=verArticulo&idProducto=2009

[36] *RiminiSystemIntegrator*. (s.f.). Recuperado el 13 de 11 de 2022, de https://www.riminisystemintegrator.it/en/domotic-for-hotels-and-accommodation-facilities/

[37] Schmidt, M. (15 de Marzo de 2022). *mmm-hue-lights*. Obtenido de https://github.com/michael5r/mmm-hue-lights

[38] shbatm. (1 de Febrero de 2022). *MMM-Carousel*. Obtenido de https://github.com/shbatm/MMM-Carousel

[39] STMicroelectronics. (Enero de 2000). *L298N hoja de datos*. Obtenido de https://www.sparkfun.com/datasheets/Robotics/L298_H_Bridge.pdf

[40] Supreme Tech. (19 de Agosto de 2016). *Supreme Tech 456mm x 608mm x 1mm Espejo Transparente acrílico bidireccional*. Obtenido de https://www.amazon.es/Supreme-Tech-101-1-1-acr%C3%ADlico-transparente/dp/B01CZ35XWY/ref=sr_1_1?__mk_es_ES=%C3%85M%C3%85%C5%BD%C3%95%C3%91&keywords=espejo%2Bbidireccional&qid=1665002335&qu=eyJxc2MiOiIzLjU5IiwicXNhIjoiMi4xMSIsInFzcCI6IjAuMDAifQ%3D%3D&s=kitc

[41] Team CodersCafe: Shebin Jose Jacob, N. (25 de 1 de 2022). *Hackster*. Recuperado el 13 de 11 de 2022, de Gesture Controlled Smart Mirror: https://www.hackster.io/coderscafe/gesture-controlled-smart-mirror-bced8f

[42] Teeuw, M. (18 de Mayo de 2023). *3rd Party Modules*. Obtenido de https://github.com/MichMich/MagicMirror/wiki/3rd-Party-Modules

[43] *Temperature and Humidity Module.* (2021). Obtenido de DHT11 Product Manual: https://components101.com/sites/default/files/component_datasheet/DHT11-Temperature-Sensor.pdf

[44] The World Air Quality Index Project Team. (23 de Mayo de 2023). *Air Quality Open Data Platform*. Obtenido de https://aqicn.org/data-platform/token/

[45] timdows. (3 de Abril de 2023). *MMM-JsonTable*. Obtenido de https://github.com/timdows/MMM-JsonTable

[46] *Vikey*. (s.f.). Recuperado el 13 de 11 de 2022, de Hotel Automation: https://vikey.it/en/domotics/

[47] WickedMakers. (19 de 6 de 2020). *instructables*. Recuperado el 13 de 11 de 2022, de How to Make a DIY Smart Mirror: https://www.instructables.com/DIY-Smart-Mirror-1/

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS

Figura 23. ODS 11 Ciudades y comunidades sostenibles

Este trabajo enfocado en la industria de la hostelería busca un diseño seguro, resiliente, inclusivo y sostenible para las comunidades y ciudades. Gracias a facilitar el ahorro de energía del diseño y mostrar de forma accesible la información.

Figura 24. ODS 3 Salud y bienestar

Con este trabajo facilitaremos el bienestar de los clientes de hostelería. Los huéspedes podrán tener una climatización e iluminación favoreciendo su confort.

Figura 25. ODS 13 Acción climática

Se luchará contra el cambio climático con este trabajo al ayudar con la eficiencia energética de la iluminación y al informar de la calidad del aire desde las habitaciones de hoteles.

ANEXO II

CÓDIGO PYTHON DEL RECONOCE_GESTOS_Y_MOTOR.PY

```
#import
import cv2
import mediapipe as mp
import pyautogui
import time
import RPi.GPIO as GPIO
from time import sleep
import json

# Inicialización del motor
GPIO.setwarnings(False)

in1 = 6
in2 = 5
en_a = 13
GPIO.setmode(GPIO.BCM)
GPIO.setup(in1,GPIO.OUT)
GPIO.setup(in2,GPIO.OUT)
GPIO.setup(en_a,GPIO.OUT)

q=GPIO.PWM(en_a,5)
q.start(75)

GPIO.output(in1,GPIO.LOW)
GPIO.output(in2,GPIO.LOW)
motorMove=0
##################################

# Inicialización del reconocimiento de gestos
printHand=0
pyautogui.PAUSE = 0.01
mp_drawing = mp.solutions.drawing_utils
mp_hands = mp.solutions.hands
##################################

tipIds = [4, 8, 12, 16, 20]
state = None
Gesture = None
wCam, hCam = 320, 240
############################

#Mapeo de la posición de los dedos
def fingerPosition(image, handNo=0):
    lmList = []
```

```
    if results.multi_hand_landmarks:
        myHand = results.multi_hand_landmarks[handNo]
        for id, lm in enumerate(myHand.landmark):

            h, w, c = image.shape
            cx, cy = int(lm.x * w), int(lm.y *  h)
            lmList.append([id, cx, cy])
    return lmList

# Para la entrada de la webcam:
cap=cv2.VideoCapture(0)
cap.set(3, wCam)
cap.set(4, hCam)
with mp_hands.Hands(
    min_detection_confidence=0.8,
    min_tracking_confidence=0.5) as hands:
  while cap.isOpened():
    success, image = cap.read()
    if not success:
        print("El marco de la cámara está vacío.")
        break
    image = cv2.cvtColor(cv2.flip(image, 1), cv2.COLOR_BGR2RGB)
    image.flags.writeable = False
    results = hands.process(image)

    # Mapeo de la mano en la imagen
    image.flags.writeable = True
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
    if results.multi_hand_landmarks:
      for hand_landmarks in results.multi_hand_landmarks:
        mp_drawing.draw_landmarks(
            image, hand_landmarks, mp_hands.HAND_CONNECTIONS)
    lmList = fingerPosition(image)
    if len(lmList) != 0:
        fingers = []
        for id in range(1, 5):
            if lmList[tipIds[id]][2] < lmList[tipIds[id] - 2][2]:

                fingers.append(1)
            if (lmList[tipIds[id]][2] > lmList[tipIds[id] - 2][2] ):

                fingers.append(0)
        totalFingers = fingers.count(1)

        if totalFingers == 4:
            state = "Play"
        if totalFingers == 0 and state == "Play":
            state = "Pause"

        if totalFingers == 1:
            if lmList[8][1]<80 and printHand!=1:
                printHand=1
                print("left")
```

```
                pyautogui.press('left')
            if lmList[8][1]>240 and printHand!=2:
                printHand=2
                print("Right")
                pyautogui.press('Right')
            if lmList[8][1]<240 and lmList[8][1]>80:
                printHand=0
        if totalFingers == 2:
            if lmList[8][1]<80 and motorMove!=1:
                motorMove=1
                GPIO.output(in1,GPIO.HIGH)
                GPIO.output(in2,GPIO.LOW)

                print("Subiendo")

                # Abrimos el JSON
                with
open('/home/pi/EspejoInteligente/MagicMirror/modules/Persiana.json', 'r') as f:
                    # Cargamos el JSON
                    json_data = json.load(f)

                # Cambiamos el valor a "subida"
                json_data["Persiana"][0]["value"] = "Subida"

                # Escribimos en el JSON
                with
open('/home/pi/EspejoInteligente/MagicMirror/modules/Persiana.json', 'w') as f:
                     json.dump(json_data, f)

                print('Json subida')

            if lmList[8][1]>240 and motorMove!=2:
                motorMove=2
                GPIO.output(in1,GPIO.LOW)
                GPIO.output(in2,GPIO.HIGH)

                print('Bajando')

                # Abrimos el JSON
                with
open('/home/pi/EspejoInteligente/MagicMirror/modules/Persiana.json', 'r') as
Persiana:
                   # Cargamos el JSON
                   json_data = json.load(Persiana)

                # Cambiamos el valor a "Bajada"
                json_data["Persiana"][0]["value"] = "Bajada"

                # Escribimos en el JSON
                with
open('/home/pi/EspejoInteligente/MagicMirror/modules/Persiana.json', 'w') as
Persiana:
                    json.dump(json_data, Persiana)
```

```
                print('Json bajada')
            if lmList[8][1]<240 and lmList[8][1]>80 and motorMove!=0:
                motorMove=0
                GPIO.output(in1,GPIO.LOW)
                GPIO.output(in2,GPIO.LOW)

                print('Stop')

    cv2.imshow("Media Controller", image)
    key = cv2.waitKey(1) & 0xFF

    # si se presiona `q` se rompe el bucle
    if key == ord("q"):
        break
cv2.destroyAllWindows()
```

ANEXO III

CÓDIGO JAVASCRIPT DE CONFIG.JS

```
/* MagicMirror² Config Sample
 *
 * By Michael Teeuw https://michaelteeuw.nl
 * MIT Licensed.
 *
 * For more information on how you can configure this file
 * see https://docs.magicmirror.builders/configuration/introduction.html
 * and https://docs.magicmirror.builders/modules/configuration.html
 */
let config = {
	address: "0.0.0.0",		// Address to listen on, can be:
			// - "localhost", "127.0.0.1", "::1" to listen on loopback interface
			// - another specific IPv4/6 to listen on a specific interface
			// - "0.0.0.0", "::" to listen on any interface
			// Default, when address config is left out or empty, is "localhost"
	port: puerto,
	basePath: "/",
// The URL path where MagicMirror² is hosted. If you are using a Reverse proxy
// you must set the sub path here. basePath must end with a /
	ipWhitelist: ["127.0.0.1", "::ffff:127.0.0.1", "::1", "ip"],
// Set [] to allow all IP addresses

// or add a specific IPv4 of 192.168.1.5 :

// ["127.0.0.1", "::ffff:127.0.0.1", "::1", "::ffff:192.168.1.5"],

// or IPv4 range of 192.168.3.0 --> 192.168.3.15 use CIDR format :

// ["127.0.0.1", "::ffff:127.0.0.1", "::1", "::ffff:192.168.3.0/28"],

	useHttps: false,    // Support HTTPS or not, default "false" will use HTTP
	httpsPrivateKey: "",
// HTTPS private key path, only require when useHttps is true
	httpsCertificate: "",
// HTTPS Certificate path, only require when useHttps is true

	language: "es",
	locale: "es-SP",
	logLevel: ["INFO", "LOG", "WARN", "ERROR"],
 // Add "DEBUG" for even more logging
	timeFormat: 24,
	units: "metric",
	// serverOnly:  true/false/"local" ,
	// local for armv6l processors, default
```

```
	//   starts serveronly and then starts chrome browser
	// false, default for all NON-armv6l devices
	// true, force serveronly mode, because you want to.. no UI on this device

	modules: [
		{
			module: "alert",
		},
		{
			module: "updatenotification",
			position: "top_bar",
		},
		{
			module: "clock",
			position: "top_center",
		},
		{
			module: "calendar",
			header: "Festivos de España",
			position: "bottom_left",
			config: {
				calendars: [
					{
						symbol: "calendar-check",
						url: "webcal://www.calendarlabs.com/ical-
calendar/ics/70/Spain_Holidays.ics"
					}
				]
			}
		},
		{
			module: "compliments",
			position: "middle_center",
		},
		{
			module: "weather",
			position: "bottom_right",
			config: {
				weatherProvider: "openweathermap",
				type: "current",
				location: "Madrid",
				locationID: "3117735", // ID de
http://bulk.openweathermap.org/sample/city.list.json.gz; descomprime el archivo
gz y encuentra tu ciudad
				apiKey: "apikey obtenida del Usuario en openweathermap"
			}
		},{
			module: "newsfeed",
			position: "bottom_bar",
			config: {
				feeds: [
					{
						title: "El Pais",
```

```
                                        url: "https://feeds.elpais.com/mrss-
s/pages/ep/site/elpais.com/portada",
                                    },
                                    {
                                        title: "ABC",
                                        url:
"https://www.abc.es/rss/feeds/abcPortada.xml",
                                    },
                                    {
                                        title: "El Mundo",
                                        url: "https://e00-
elmundo.uecdn.es/elmundo/rss/portada.xml",
                                    },
                                    {
                                        title: "ElDiario.es",
                                        url: "https://www.eldiario.es/rss",
                                    }
                                ],
                                showSourceTitle: true,
                                showPublishDate: true,
                                broadcastNewsFeeds: true,
                                broadcastNewsUpdates: true
                        }},
                                {
                        module: "MMM-DHT-Sensor",
                        position: "middle_center",
                        header: "Temperatura y Humedad del cuarto",
                        config: {
                                sensorPin: 2,
                                sensorType: 11,
                                updateInterval: 60* 1000,
                                initialLoadDelay: 0,
                                animationSpeed: 100,
                                units: "metric",
                                relativeScale: 35,
                                debug: false
                        }
                },
                                {
                        module: 'MMM-AQI',
                        position: 'bottom_left',
                        header: 'Air Quality Index (AQI)',
                        config: {
                                token: "token obtenido de la web",
                                city: "Madrid",
                                iaqi: true,
                                updateInterval: 30 * 60 * 1000, // Cada media hora
                                initialLoadDelay: 0,
                                animationSpeed: 1000,
                                debug: false
                        }
                        },{
                        module: "mmm-hue-lights",
```

```
                    position: "bottom_right",
                    config: {
                          bridgeIp: 'ip de Hue Bridge',
                          user: 'ususario asignado'
                    }
              },{
                module: 'MMM-JsonTable',
                carouselId:"MMM-JsonTable1",
                position: 'bottom_right',
                header: 'Estado del parking',
                config: {
                    url: 'http:// ip:puerto /modules/Estado_del_parking.json',
// Required
                    arrayName: 'Estado_del_parking'
                }
              },
              {
                module: 'MMM-JsonTable',
                carouselId:"MMM-JsonTable2",
                header:'Información económica',
                position: 'bottom_center',
                config: {
                    size: 3,
                    url:
'http://ip:puerto/modules/A_abonar_check_out_ofertas.json', // Required
                    arrayName: 'A_abonar_check_out_ofertas'
                }
              },
              {
                module: 'MMM-JsonTable',
                carouselId:"MMM-JsonTable3",
                header:'Menu del dia',
                position: 'top_center',
                config: {
                      size: 1,
                    url: 'http:// ip:puerto /modules/Menu_del_dia.json',
// Required
                    arrayName: 'Menu_del_dia'
                }
              },
              {
                module: 'MMM-JsonTable4',
                header:'Persiana',
                position: 'bottom_left',
                config: {
                    size: 1,
                    url:'http:// ip:puerto /modules/Persiana.json',// Required
                    arrayName: 'Persiana'
                }
              },
              {
                    module: 'MMM-KeyBindings',
              config: {
```

```
                        evdev: { enabled: false },
                  enableKeyboard: true
                  },
            },
            {
            module: 'MMM-Carousel',
            position: 'bottom_bar', // Required only for navigation controls
            config: {
                  transitionInterval: 1000000000,
                  mode: 'slides',
                  showPageIndicators: true,
                  showPageControls: false,
                  slides: {
                        main: ['clock',
'weather','calendar','compliments','newsfeed'],
                        "Slide 1": ['clock', 'MMM-DHT-Sensor','mmm-hue-
lights',"MMM-JsonTable4"],
                        "Slide 2": ['MMM-AQI',
                        {name:'MMM-JsonTable',carouselId:"MMM-JsonTable1"},
                        {name:'MMM-JsonTable',carouselId:"MMM-JsonTable2"},
                        {name:'MMM-JsonTable',carouselId:"MMM-JsonTable3"},
                        'clock'],
                  },
                  keyBindings: {
                        enabled: true,
                        enableKeyboard: true,
                        mode: "DEFAULT",
                        map: {
                              NextSlide: "ArrowRight",
                              PrevSlide: "ArrowLeft",
                              Slide0:    "Home"
                        },

                  }},
            },

      ]


};

/*************** DO NOT EDIT THE LINE BELOW ***************/
if (typeof module !== "undefined") {module.exports = config;}
```