universidad de león

Escuela de Ingenierías
Industrial, Informática y Aeroespacial

GRADO EN INGENIERÍA EN ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Trabajo de Fin de Grado

DESARROLLO DE UN SISTEMA DE SUPERVISIÓN Y MONITORIZACIÓN DEL FUNCIONAMIENTO DE PANELES SOLARES

DEVELOPMENT OF A SYSTEM FOR SUPERVISION AND MONITORING OF THE OPERATION OF SOLAR PANELS

Autor: Marcos Martín Calderón
Tutor: José Guillermo Rosas Mayoral

(Julio, 2023)

UNIVERSIDAD DE LEÓN
Escuela de Ingenierías Industrial, Informática y Aeroespacial

GRADO EN INGENIERÍA EN ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA
Trabajo de Fin de Grado

ALUMNO: Marcos Martín Calderón

TUTOR: Guillermo Rosas Mayoral

TÍTULO: Desarrollo de un sistema de Supervisión y Monitorización del funcionamiento de paneles solares

TITLE: Development of a system for Supervision and Monitoring of the operation of solar panels

CONVOCATORIA: Julio, 2023

RESUMEN: Durante el desarrollo de este Trabajo de Fin de Grado se expone tanto el planteamiento como el desarrollo de un prototipo de sistema de supervisión y monitorización del funcionamiento de sistemas fotovoltaicos. Este sistema permite recoger a tiempo real los parámetros de funcionamiento del sistema, analizarlos y almacenarlos en una base de datos MySQL. Gracias a la implementación del web service Node-RED y la comunicación MQTT, el prototipo se integra dentro del paradigma de la Industria 4.0. Con este prototipo se da una solución que cumple con las necesidades actuales del mercado y usando durante el desarrollo, herramientas innovadoras dentro del mercado.

ABSTRACT: This end-of-degree project explains both the approach and the development of a prototype of a system of supervision and monitoring of the operation of photovoltaic system. This system allows the collection in real time of the operating parameters of the system, analyze them and store them in a MySQL database. Thanks to the implementation of the web service Node-RED and the MQTT communication, this prototype is integrated into the 4.0 Industry. With this prototype it is guaranteed a solution that meets the current needs in the market and it is as well assured the use of innovating tools.

Palabras clave: Diseño, panel fotovoltaico, MQTT, Node-RED.

Firma del alumno:	**VºBº Tutor/es:**

Índice de contenidos

Índice de contenidos 3

Índice de figuras 7

Glosario de signos, símbolos, unidades, acrónimos y términos 9

1. Introducción 11

1.1. Objetivos generales 11

1.1.1. Toma de variables del sistema fotovoltaico 11

1.1.2. Comunicación del sistema a través del protocolo MQTT. 11

1.1.3. Implantación de servidor de almacenamiento de datos. 11

1.1.4. Monitorización desde la nube. 11

1.2. Objeticos particulares 12

1.2.1. Búsqueda de equipos para la toma de variables del sistema 12

1.2.2. Implementación del software de comunicación. 12

1.2.3. Implementación del software de toma de variables. 12

1.2.4. Implementación del software de captura de imágenes. 12

1.2.5. Implementación del software de control desde la nube. 12

1.2.6. Implementación de una interfaz gráfica. 12

1.2.7. Implementación del servidor de datos. 12

2. Contexto 13

2.1. Evolución energética en España 13

2.2. Energía renovable en España 14

2.3. Energía solar fotovoltaica en España 15

3. Estado del arte 18

3.1. Revoluciones industriales 18

3.1.1. La Primera Revolución Industrial 18

3.1.2. La Segunda Revolución Industrial 18

3.1.3. La Tercera Revolución Industrial 19

3.2. Industria 4.0 19

3.3. Energía solar y Componentes de la radiación solar 20

3.4. Efecto fotoeléctrico 21

3.4.1. Semiconductores-Uniones “P-N” 21

3.4.2. Carácter de la radiación solar 21

3.4.3. Principio físico 21

3.4.4. Dopaje de Semiconductores-Uniones P-N 22
3.5. Células fotovoltaicas 22
3.6. Módulos fotovoltaicos 23
3.7. Tipos de inversores fotovoltaicos 24
2.6.1. Inversor central 25
2.6.2. Micro inversor 25
3.8. Curvas características de una célula fotovoltaica 25
2.7.0. Curva Corriente-Tensión (I-V) 26
2.7.1. Influencia de la radicación solar 27
2.7.2. Influencia de la temperatura 28
3.9. Automatización industrial 4.0 29
3.9.1. Pirámide de automatización, Modelo CIM 29
3.9.2. Sistemas SCADA 30
3.10. Sistemas de monitorización solar 31
3.11. Protocolo MQTT 32
3.11.1. Cliente y Bróker 34
3.11.2. Topic 34
3.11.3. QoS (Quality of Service) 34
3.12. Base de datos MySQL 35
4. Diseño general del sistema solución 37
4.1. Toma de datos del sistema fotovoltaico 37
4.2. Captura de imágenes del sistema 38
4.3. Comunicación MQTT 39
4.4. Servidor de gestión y almacenamiento de datos 39
4.5. Interfaz de supervisión y visualización 40
5. Hardware utilizado en el estudio 42
4.1. Microcontrolador 42
5.1.1. Lectura de entradas analógicas 43
4.2. Módulo ESP32 CAM 43
4.2.1. Adaptador Serie USB a TTL 44
4.2.2. Configuración de pines para la carga 44
4.3. Sensor de temperatura 45
4.4. Sensor de voltaje 46
4.5. Sensor de corriente 47
4.6. PC servidor 48
4.7. PC de trabajo 48

5. Software utilizado en el estudio 49
5.1. Arduino IDE y C++ 49
5.2. Node-RED 50
5.2.1. Nodos MQTT 51
5.2.2. Nodo MySQL 52
5.2.3. Nodos Dashboard 52
5.3. Eclipse Mosquitto 52
5.4. Protocolo SSH 52
5.5. Apache Friends Xampp 53
6. Preparación de equipos 55
6.1. Instalación y Configuración Arduino IDE 55
6.1.1. Instalación del núcleo ESP32 55
6.1.2. Librerías de programación 56
6.2. Instalación Node-RED 57
6.3. Conexión SSH entre Windows y Linux Ubuntu 58
6.4. Instalación Broker Mosquitto 59
6.5. Instalación y configuración Xampp 60
6.6. Conexiones eléctricas entre dispositivos 61
7. Diseño de software del microcontrolador 62
7.1. Lectura de señales analógicas de los sensores 62
7.2. Tratamiento de datos y muestreo 62
7.3. Conexión a la red WLAN 63
7.4. Comunicación MQTT 63
8. Diseño de software del Módulo ESP32 CAM 65
8.1. . Conexión a la red WLAN 65
8.2. Comunicación MQTT 65
8.3. Captura de imagen 66
8.4. Configuración de imagen 66
9. Diseño base de datos MySQL en phpmyadmin 67
10. Diseño del Web Service Node-RED 68
10.1. Comunicación MQTT 68
10.2. Inserción de datos en MySQL 69
10.3. Interfaz de visualización Dashboard 70
10.4. Configuración interna de mensajes 71
11. Resultados obtenidos 73
11.1. Conexión de componentes electrónicos 73

11.2. Programación del microcontrolador y módulo ESP32 CAM 73
11.3. Programación del web service 74
11.4. Interfaz de supervisión y monitorización 75
11.5. Almacenamiento de datos en MySQL 77
12. Conclusiones 78
12.1. Hitos futuros 78
12.1.1. Algoritmo de predicción de producciones 78
12.1.2. Implementar acceso al servidor desde fuera de la red local 79
Bibliografía 80
Anexo 1. Código C++ Microcontrolador 85
Anexo 2. Código C++ ESP32 CAM 89
Anexo 3. Funcionalidades del sistema 92

Índice de figuras

Ilustración 1.1 Evolución de la generación eléctrica renovable (%) 2014-2018.....15

Ilustración 1.2 Insolación anual España 16

Ilustración 2.1 Generación de flujo de electrones durante el efecto fotoeléctrico.....22

Ilustración 2.2 Formas de interconexión células fotovoltaicas.....24

Ilustración 2.3 Comparativa rendimiento de inversores.....25

Ilustración 2.4 Curvas características células fotovoltaicas. Curva I-V y Curva P-V.....26

Ilustración 2.5 Curva I-V y sus parámetros fundamentales.....27

Ilustración 2.6 Efecto de la radiación solar sobre la potencia generada en célula fotovoltaica.....28

Ilustración 2.7 Efecto de la temperatura sobre la potencia generada en célula fotovoltaica.....29

Ilustración 2.8 Pirámide de automatización.....30

Ilustración 2.9 Esquema básico de un sistema SCADA.....31

Ilustración 2.10 Dinámica de transmisión de información protocolo MQTT.....33

Ilustración 3.1 Esquema visual sistema de captación de variables.....38

Ilustración 3.2 Esquema visual sistema de captura de imagen.....39

Ilustración 3.3 Estructura del sistema solución.....41

Ilustración 4.1 Pinout ESP32 WROOM32.....42

Ilustración 4.2 Módulo ESP32 CAM.....44

Ilustración 4.3 Conexión ESP32 CAM y adaptador USB a TTL.....45

Ilustración 4.4 Módulo termistor KY-028.....46

Ilustración 4.5 Módulo sensor de voltaje FZ0430.....47

Ilustración 4.6 Pinout Módulo sensor de corriente ACS712.....47

Ilustración 5.1 Plataforma Arduino IDE.....50

Ilustración 5.2 Editor de flujo Node-RED 51

Ilustración 5.3 Panel de control de herramienta Xampp.....54

Ilustración 6.1 Gestor de tarjetas de Arduino IDE.....56

Ilustración 6.2 Administrador de paleta Node-RED.....58

Ilustración 6.3 Terminal PC trabajo con conexión SSH hacia PC servidor.....59

Ilustración 6.4 Interfaz phpmyadmin.....61

Ilustración 9.1 Creación de DB relacional en phpmyadmin……..67

Ilustración 10.1 Configuración nodo MQTT y servidor……69

Ilustración 10.2 Función inserción de datos en DB MySQL……72

Ilustración 10.3 Sentencia if de comparación en nodo función……73

Ilustración 11.1 Cableado dispositivos electrónicos……..74

Ilustración 11.2 Diagramas de flujo del web service Node-RED……75

Ilustración 11.3 Pantalla sensores en Dashboard……76

Ilustración 11.4 Pantalla Inicio en Dashboard ……76

Ilustración 11.5 Pantalla curvas en Dashboard……76

Ilustración 11.6 Datos registrados en base de datos MySQL……77

Ilustración Anexo 3.1 Visión global del sistema……92

Ilustración Anexo 3.2 Panel Sensores en condiciones normales de trabajo……92

Ilustración Anexo 3.3 Panel Inicio durante condiciones normales de trabajo……93

Ilustración Anexo 3.4 Panel Sensores con valor de corriente bajo……93

Ilustración Anexo 3.5 Panel Inicio con mensaje de alerta y estado bajo de producción……94

Ilustración Anexo 3.6 Panel de Inicio durante producción paneles media……94

Ilustración Anexo 3.7 Panel de Inicio ……95

Glosario de signos, símbolos, unidades, acrónimos y términos

Base de datos (DB): colección de datos estructurados, los cuales guardan relación entre sí.

Buffer: Memoria intermedia que almacena una imagen antes de ser procesada

CEI: Computer Enterprise Integrated

CIM: Computer Integrates Manufacturing

Energía primaria: Fuentes de energía naturales en su forma original o inalteradas.

FA: Factory Automation

Hardware: Partes físicas y tangibles de un sistema informático, formado por componentes eléctricos, electrónicos y electromecánicos.

Insolación: Intervalo de tiempo durante el cual el sol incide sobre una superficie, en el transcurso de un periodo de tiempo.

Isohelia: Curva, dibujada sobre un mapa, que indica los lugares con la misma duración de insolación durante un intervalo de tiempo dado.

JIT: Just In Time

Latencia: retraso en la llegada de un paquete al destino, medido en unidades de tiempo.

LDR: Light Dependent Resistor

Librerías Arduino: colección de funciones que se incluyen de forma sencilla y explicita dentro de un sketch de programación y proporciona una funcionalidad especifica.

ONU: Organización de las Naciones Unidas

PAEE: Plan de Ahorro y Eficiencia Energética

PEN: Plan Energético Nacional

PER: Plan de Energías Renovables

PNIEC: Plan Nacional Integrado de Energía y Clima

Radiación solar: Conjunto de radiaciones electromagnéticas emitidas por el Sol (W/m^2).

Red WLAN (Wireless Local Area Network): red inalámbrica de comunicación para distancias cortas mediante ondas de radio o infrarrojas.

SCADA: Supervisory And Data Acquisition

Software: Conjunto de componentes lógicos que hacen posible el desarrollo de tareas específicas.

Termistor: Elemento de detección de temperatura compuesto por un material semiconductor sintetizado, que presenta un cambio de resistencia en proporción a un cambio de temperatura.

TLS: Transport Layer Security

Transceptor: Dispisitivo que cuenta con un transmisor y un receptor que comparten la misma circuitería.

Web Service: interfaz multiplataforma (cliente y servidor no tienen por qué contar con la misma comunicación) y distribuida (diferentes clientes acceden a un mismo servicio) mediante la que dos máquinas o aplicaciones se comunican entre sí.

MQTT (Message Queuing Telemetry Transport: protocolo de mensajería ligero para clientes conectados a redes con limitaciones de ancho de banda o poco fiables.

1. Introducción

La industria energética ha estado enfocada desde finales del siglo XX, en una transición energética hacia fuentes de generación eléctrica renovable. Durante los primeros años se centró en la expansión de la energía eólica, más rentable que los sistemas fotovoltaicos en aquellos años. A partir de 2018, tras implantarse nuevas leyes reguladoras de la energía solar y desarrollarse sistemas con mayores rendimientos, la energía solar ha incrementado su presencia en el mapa energético español.

Ante la necesidad de hacer uso de grandes áreas de terreno para la instalación de paneles fotovoltaicos, el auge de las nuevas tecnologías y las necesidades de la industria 4.0, aparece un nuevo concepto en el sector fotovoltaico: La supervisión y monitorización fotovoltaica. En esencia, este concepto surge para mejorar los rendimientos de los sistemas fotovoltaicos, aumentando la producción eléctrica por área ocupada. Además, intenta mejorar la detección de errores y fallos en las instalaciones.

Dada la reciente introducción de los sistemas de supervisión en la industria fotovoltaica, los sistemas desarrollados sobre la materia presentan elevados costes y ven sus funciones limitadas. Esta propuesta aporta un método de supervisión fotovoltaica que cubra las necesidades del sector, implementando tecnologías diferentes a las utilizadas hasta el momento en estas tareas y siendo ideal para sistemas con micro inversores.

1.1. Objetivos generales

Los objetivos generales del presente trabajo son los siguientes:

1.1.1. Toma de variables del sistema fotovoltaico.

Se pretende realizar la medición de 3 variables de los paneles fotovoltaicos. Una de ellas física (temperatura) y dos eléctricas (tensión e intensidad).

1.1.2. Comunicación del sistema a través del protocolo MQTT.

Se pretende implementar la comunicación bidireccional del sistema a través del protocolo MQTT para transmitir el valor de las variables recibidas.

1.1.3. Implantación de servidor de almacenamiento de datos.

Se pretende implantar un servidor donde gestionar y almacenar los datos extraídos del proceso.

1.1.4. Monitorización desde la nube.

Se pretende implementar un sistema de visualización de los datos desde la nube y captura de imagen del estado del sistema.

1.2. Objeticos particulares

Los objetivos particulares del presente trabajo son los siguientes:

1.2.1. *Búsqueda de equipos para la toma de variables del sistema*.

Se realizará una búsqueda tanto de los sensores necesarios para la captación de las variables, como de los elementos necesarios para la toma de imágenes y de los dispositivos embebidos necesarios para el control.

1.2.2. Implementación del software de comunicación.

Se diseñará un programa para la recepción e interpretación de datos que fluyen dentro del sistema.

1.2.3. Implementación del software de toma de variables.

Se diseñará un programa para la toma de variables del proceso fotovoltaico.

1.2.4. Implementación del software de captura de imágenes.

Se diseñará un programa para la captura de imágenes del sistema fotovoltaico.

1.2.5. Implementación del software de control desde la nube.

Se diseñará un software necesario para enviar y recibir mediante MQTT los datos del sistema y mensajes relacionados con el estado del proceso.

1.2.6. *Implementación de una interfaz gráfica*.

Se diseñará una interfaz humano-máquina para visualizar los datos obtenidos y dar órdenes de gestión de estos.

1.2.7. *Implementación del servidor de datos*.

Se implementará un servidor de recepción y almacenamiento de datos del proceso.

2. Contexto

Para situar el estudio dentro del marco social y económico en el que se desarrolla, es importante conocer la evolución de la energía solar fotovoltaica en España. Por ello, es necesario tratar tres puntos clave: “Evolución energética en España”, “Energía renovable en España” y “Energía solar fotovoltaica en España”.

2.1. Evolución energética en España

El consumo energético en España durante el último siglo ha estado ligado al desarrollo industrial, situaciones de conflicto bélico o crisis económicas, evolucionando en cuatro etapas claramente diferenciadas: [1]

1. Inicios de siglo hasta el Plan de Estabilización (1959)
2. Desde el Plan de Estabilización hasta 1973
3. Desde 1973 hasta 1998
4. Desde 1998 hasta la actualidad

Durante los inicios de la primera etapa se logra un gran desarrollo industrial con el comienzo de la Primera Guerra Mundial debido a la gran demanda exterior y provocando un fuerte incremento del consumo energético en forma de carbón [1], dado su bajo coste y alta densidad energética [2]. Tras la guerra civil, este desarrollo se vio frenado por la ausencia de materias primas [1] y las dificultades de satisfacer el aumento de consumo de energía, principalmente eléctrica de origen hidráulico. [3]

La segunda etapa comienza con la llegada del *“Plan de Estabilización de 1959”*, lo que supone un cambio en la estructura energética [1]. Se produce la apertura de España al exterior, situación que permite establecer acuerdos internacionales [4] y permite la entrada al país de nuevas fuentes de energía primaria como gas natural y petróleo. Se pone en funcionamiento la energía nuclear en 1968 y el consumo de carbón permanece estable. [1]

Durante la década de los 70 se producen dos crisis energéticas y económicas (1973 y 1979), conocidas como las crisis del petróleo y originadas por la guerra de Yom Kippur, dando lugar al comienzo de una nueva etapa energética en el país en la cual se elevan los precios de combustibles fósiles (principalmente petróleo). Estos hechos reflejan la dependencia del sistema energético y económico sobre los combustibles fósiles, provocando la búsqueda de fuentes nacionales de energía alternativa [5]y la aprobación del PEN en 1979, como plan de actuación. [1] Sin embargo, con la aparición de nuevos yacimientos de petróleos y el incremento de entrada de gas natural al país, generó un nuevo periodo de estabilidad para los combustibles fósiles. [5], [6]

El inicio de la última etapa se sitúa en 1998 tras la entrada oficial de España en el acuerdo internacional denominado *"Protocolo de Kioto"*, y se extiende hasta la actualidad. En el apartado 2.2 del presente Estudio se tratará en profundidad los antecedentes que originan la llegada de esta nueva etapa, así como los hechos que están marcando su desarrollo. [6]

2.2. Energía renovable en España

Las energías renovables han estado presentes durante el último siglo, tal y como hemos visto en el apartado anterior 2.1, principalmente como energía secundaria de origen hidroeléctrico, y siempre subordinadas al uso de combustibles fósiles.

En 1987 es publicado por la ONU el informe Brundtland, logrando una toma de conciencia sobre los efectos negativos para la conservación del planeta, debido al consumo desmesurado de energía. La creación de la Cumbre de Rio 1992, cuyo objetivo es tratar las emisiones de gas de efecto invernadero, dará por sentado la necesidad de un cambio en la relación entre energía y crecimiento económico. Ambos hechos, marcan los antecedentes de una nueva etapa energética, donde las energías renovables marcarán el camino del consumo energético. [6]

Esta nueva etapa se inicia con la entrada de España en el *"Protocolo de Kioto de 1997",* en 1998 y con el objetivo de tomar medidas energéticas de reducción de emisiones de CO_2 a partir de 2005. [7] Para ello se crean planes de acción para el desarrollo de las energías renovables y el ahorro energético:

- **Plan de Fomento de las Energías Renovables (2000-2010):** fija como objetivo generar en 2010 el 30% de la electricidad a partir de fuentes de energía renovable. [8]
- **PAEE (2008.2012):** establece un ahorro energético del 9% en 2016 y para lo que establece revisiones periódicas (2012-2014;2015-2016). [9]
- **PER (2011-2020):** fija como objetivo una cuota mínima del 20% y 10% de energía renovable en el consumo final bruto del país y del consumo de energía en el sector del transporte. [10]

Como resultado de los Planes establecidos, en 2018 el 40,1% de la electricidad consumida en España fue generada en instalaciones renovables y el 16% de la energía total consumida provenía de fuentes renovables. Dentro de este porcentaje, destaca la energía eólica con un 49,35% de energía eléctrica originada frente al 34,03% que presento la energía hidráulica y los 7,8% de la energía solar. [11]

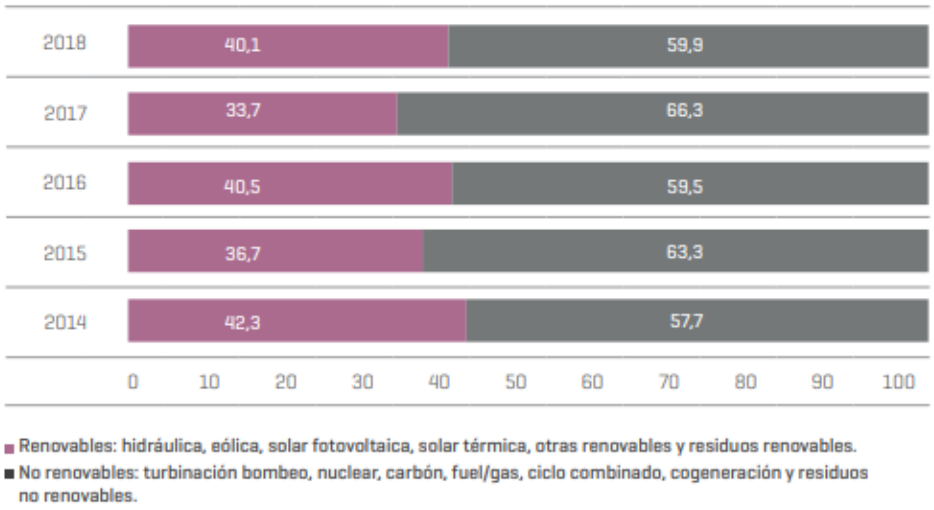

Ilustración 2.1 Evolución de la generación eléctrica renovable (%) 2014-2018 Fuente: Red Eléctrica de España

En 2019 según los datos aportados por Red Eléctrica de España, las energías renovables fueron las máximas productoras de electricidad, dejando atrás a los combustibles fósiles y a la energía nuclear.

En 2020, Europa y por consiguiente España, se adentraron en un nuevo proyecto denominado "*Triple 20"* con tres objetivos marcados: Reducción de emisiones de gases en un 20%; Aumento del 20% en el uso de energías renovables; Mejora de la eficiencia en un 20%. [10]

Actualmente, España también cuenta con el "*PNIEC 2021-2030"* con el objetivo establecido por la Comisión Europea de alcanzar el 74% de producción eléctrica procedente de energías renovables en 2030 y una economía descarbonizada para 2050. [12]

2.3. Energía solar fotovoltaica en España

La Península Ibérica y el resto de los territorios que componen España, cuentan con un gran potencial de aprovechamiento solar si se compara con el resto de Europa. Se debe a dos razones:

- Niveles de Insolación elevados. Según la ilustración 2.2, se diferencian tres zonas de insolación delimitadas por isohelias: Zona norte, con mayor nubosidad y obteniendo un nivel de insolación anual de 2000 horas

anuales; Zona central con, con niveles entre 2000 y 2600 horas; Zona sur, donde se superan las 2600 horas.

- Coste de instalaciones de autoconsumo más bajo de Europa, a lo largo de la vida útil del sistema. [13]

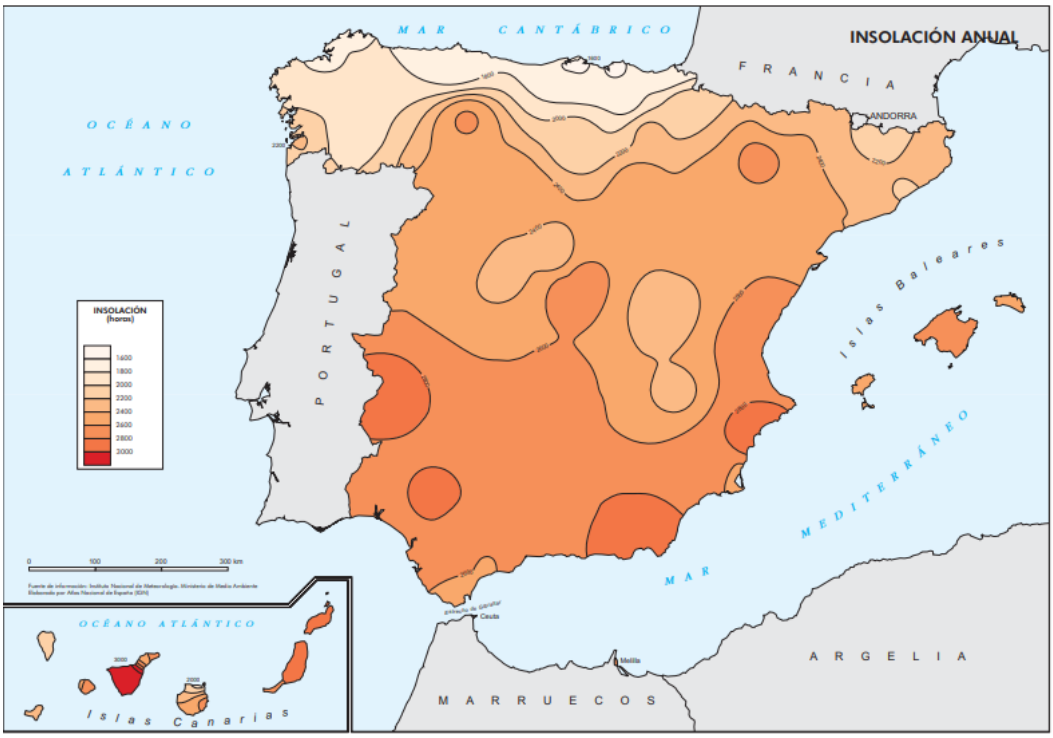

Ilustración 2.2 Insolación anual España *Fuente: Instituto Geográfico Nacional*

Pese a ser uno de los países que mejores condiciones reúne en el aprovechamiento de energía solar fotovoltaica, no se obtuvo un gran desarrollo de esta durante los primeros años de explotación renovable. En el año 2018, únicamente el 3% de la demanda de energía eléctrica es cubierta con este tipo de energía.

Por ello, en este mismo año se aprobó el Real Decreto para la regulación de condiciones administrativas, económicas y técnicas referentes al consumo de energía fotovoltaica, con vigencia actual. Esto permitió la creación de un marco estable y de libre mercado para el desarrollo, logrando que los ciudadanos tengan acceso a la producción y venta de energía en instalaciones de menor potencia, además de compensaciones económicas por verter energía limpia a la red y el derecho de autoconsumo compartido. [14]

En el último informe anual de Red Eléctrica de España, la energía solar fotovoltaica ha representado el 16,6% de la energía total nacional, alcanzando los

27.283 GWh. Esto supone un aumento del 32,9% respecto al año anterior tras aportar 4500MW más al parque de generación. Se demuestra así, el gran desarrollo que está experimentando la energía solar fotovoltaica tras los planes para su fomento e instalación. [15]

3. Estado del arte

Con el objetivo de situar el presente estudio en el marco tecnológico en el que se desarrolla y conocer su aspecto innovador, es necesario tratar los siguientes apartados y así comprender algunas nociones básicas. Estos apartados versan “Las revoluciones industriales”,

3.1. Revoluciones industriales

Durante los dos siglos de historia anteriores, se suceden tres revoluciones industriales que sientan las bases tecnológicas, sociales y económicas, permitiendo desarrollar la cuarta revolución industrial o industria 4.0, tratada en el apartado 3.2.

3.1.1. *La Primera Revolución Industrial*

Tiene lugar entre la segunda mitad del siglo XVIII y el inicio del siglo XIX. Supone un cambio importante en el sistema productivo al sustituir el trabajo manual y la tracción animal, por maquinas industriales accionadas por el motor de vapor. Esta evolución, provoca un cambio en la concepción de la industria, pasando de ser una industria distribuida (en talleres artesanales) a una industria concentrada en fábricas, generando, además, importantes cambios sociales y económicos. [16]

También genera cambios en el consumo energético, sustituyendo las fuentes de energía biológicas (madera) por fuentes de energía minerales (carbón). [17]

3.1.2. *La Segunda Revolución Industrial*

Habitualmente ubicada entre 1870 y 1914, comienza tras un cambio de paradigma en el desarrollo tecnológico, evolucionando del desarrollo con baja base científica, a un desarrollo fundamentado en la investigación y la ciencia. La década previa a la segunda revolución industrial, se conoce como el periodo de historia cuantitativamente más innovador. Se desarrollan principalmente las formas de transformación de energía eléctrica. Se inventa la dinamo, el motor eléctrico de CC y CA y se desarrollan otros usos de la electricidad, como es la transmisión de información, inventando el telégrafo. [18], [19]

Con estos precedentes, se consigue desarrollar durante la segunda revolución industrial el teléfono, la radio y la televisión. Esto supone un avance radical en el campo de las telecomunicaciones. Además, se logra desarrollar el motor de combustión interna, dando lugar a la aparición del automóvil. [18]

3.1.3. *La Tercera Revolución Industrial*

A falta de consenso, su inicio se estima a mediados del siglo XX con la aparición de tecnologías como la electrónica e internet, que fomentan la digitalización de los procesos.[20] Para lograr el desarrollo digital, se fundamenta en tres conceptos: Las energías renovables, el almacenamiento de energía y las redes de suministro inteligentes. Según varios autores, esta Tercera Revolución Industrial se presenta como una solución adecuada al cambio climático y la pobreza energética sufrida durante periodos anteriores. [21]

3.2. Industria 4.0

También conocida como Cuarta Revolución Industrial, se asienta sobre las tres revoluciones anteriormente tratadas. Parte de un alto grado de digitalización desarrollado durante la Tercera Revolución Industrial, tras lograrse tecnologías como internet y el estudio de objetos inteligentes, con el fin de conseguir un nuevo cambio en el paradigma dentro de la producción industrial. Se busca una producción basada en sistemas modulares y eficientes, describiendo escenarios donde los productos sean capaces de gestionar su propio proceso de fabricación. Con esto, se pretende lograr sistemas de producción en los que se consigan productos individuales de tamaño unitario manteniendo los costes de producción en cadena o a gran escala. [22], [23]

Dentro de la Industria 4.0, existen dos líneas de desarrollo diferenciadas:

- Una enfocada hacia la demanda individualizada, flexibilidad del desarrollo de productos, disminución de los tiempos de producción y mejora de la eficiencia en términos económicos y medioambientales.
- Otra guiada por el aumento del grado de automatización y sistematización de los procesos, miniaturización de los productos electrónicos y la conexión a red de todos los elementos de los sistemas industriales.

Lo que ambas corrientes pretenden lograr es cumplir con la creciente demanda de cambio, pasando de una industria con mercados y sistemas de producción enfocados al producto y su fabricación, a una industria enfocada como servicio. Este objetivo, lleva a la Industria 4.0 a realizar un cambio organizacional de los sistemas de producción, a través de conceptos como CIM, CEI, FA o JIT, así como conseguir, tanto la integración horizontal del tráfico de datos entre los socios, proveedores y los clientes, y la integración vertical desde el inicio de desarrollo del producto hasta el resultado final, dentro de las empresas. El modelo que surge como resultado, es un sistema donde

todos los procesos que intervienen están integrados en un sistema de información común y a tiempo real, fusionando el mundo real y el virtual. Con esto se logra satisfacer la demanda de inmediatez de la industria real y la problemática de individualización. [24]-[26]

3.3. Energía solar y Componentes de la radiación solar

La energía solar es la energía contenida en la radiación solar. Supone la mayor fuente de radiación electromagnética del planeta. Para su aprovechamiento el primer paso es su captación, existiendo actualmente dos sistemas muy diferentes según este aspecto:

- Sistemas Pasivos: se basan en la utilización directa de la energía solar sin hacer uso de elementos externos.
- Sistemas Activos: se basan en la captación de la radiación solar por medio de elementos denominados colectores.

Dentro de los sistemas activos, podemos diferenciar dos tipos de aprovechamiento de la energía en función de las características del colector: Se puede realizar un aprovechamiento del calor contenido en la radiación solar (Energía Térmica) o un aprovechamiento de la energía luminosa (fotones) de la radiación solar, con el fin de generar corriente eléctrica a través de células fotovoltaicas basadas en el efecto fotovoltaico (Energía Fotovoltaica). [1]

La radiación solar ha de ser objeto de estudio para una buena planificación, diseño y aplicación de un proyecto solar. Esta, está formada por la suma de tres componentes diferenciadas según el modo de incidencia de los rayos solares en la tierra: [27]

- Radiación directa: Procede directamente del Sol sin ser desviada por la atmósfera.
- Radiación difusa: Sufre cambios direccionales de reflexión y difusión en la atmósfera.
- Radiación Albedo: Reflejada por la reflexión del suelo.

Estas componentes de la radiación intervienen en los procesos de aprovechamiento de la energía solar, aunque dependiendo del sistema de aprovechamiento elegido, tendrán mayor o menor importancia. [2]

3.4. Efecto fotoeléctrico

El efecto fotoeléctrico o fotovoltaico es la creación de una fuerza electromotriz en un semiconductor tras producirse la absorción de radiación luminosa. La producción de energía eléctrica se basa en este efecto y se produce en ciertos materiales cargados eléctricamente. [1]

3.4.1. Semiconductores-Uniones "P-N"

Los materiales pueden clasificarse en tres grupos eléctricamente hablando: conductores, semiconductores y aislantes. Para esta aplicación, los materiales usados son semiconductores (principalmente Silicio y Germanio) los cuales a bajas temperaturas poseen propiedades aislantes y a altas temperaturas, propiedades conductoras que permiten la libertad de movimiento de los electrones.[1] Dentro de los semiconductores existen dos tipos de portadores de corriente: electrones libres (dotados de carga eléctrica negativa) y huecos (dotados de carga eléctrica positiva por ausencia de electrones). Ciertos semiconductores poseen conducción por huecos y se denominan semiconductores tipo p, y otros poseen conducción por electrones y se denominan semiconductores tipo n. [27], [28]

3.4.2. Carácter de la radiación solar

La radiación solar (luz) tiene carácter de onda y de partícula (naturaleza dual). El cuerpo es lo que se conoce como fotones y llevan asociados una cantidad de energía. Además, los átomos se componen de un núcleo que alberga la carga eléctrica positiva (protones y neutrones) y de electrones con carga eléctrica negativa girando en capas de energía entorno al núcleo, gracias al enlace electrón-núcleo. Los electrones con mayor energía que la última capa, se escapan. [27]

3.4.3. Principio físico

Al incidir la luz sobre un material semiconductor, los fotones que la componen transmiten su energía a los electrones de valencia del semiconductor y rompen el enlace electrón-núcleo, escapando. Al lugar que deja el electrón se le denomina hueco y adquiere la misma carga que el electrón, pero positiva. Cada enlace roto supone la liberación de un electrón y un hueco creado, de forma que ambos son desplazados por la estructura del material. Su movimiento es aleatorio y sin dirección definida pudiendo originar recombinaciones. Como solución, se crea un campo eléctrico en el semiconductor para conducir el movimiento de electrones y huecos, circulando cargas positivas y negativas en direcciones opuestas. [1], [27], [28]

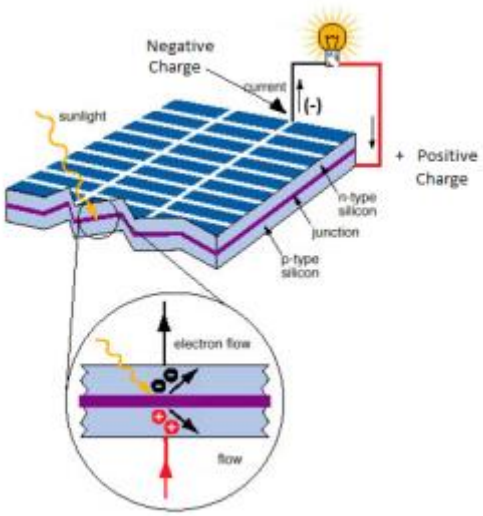

Ilustración 3.1 Generación de flujo de electrones durante el efecto fotoeléctrico

En la ilustración 3.1 se puede observar el principio de generación de flujos direccionados de cargas que tiene lugar dentro de un panel fotovoltaico.

3.4.4. *Dopaje de Semiconductores-Uniones P-N*

Consiste en introducir impurezas dentro de la red cristalina de los semiconductores con el fin de crear un campo eléctrico que mejore la conductividad del material y así conducir las cargas.

Para crear el campo eléctrico se unen dos regiones de material semiconductor, en el caso del silicio: unión p-n con cuatro electrones de valencia. Al introducir dentro del silicio átomos de Fósforo, con cinco electrones de valencia, se producen cargas negativas excedentes con conducción eléctrica tipo n. En cambio, al introducir átomos de Boro, con tres electrones de valencia, aparecen cargas positivas excedentes con conducción eléctrica tipo p. De este modo, los huecos originados por el material p, son ocupados por los electrones libre del material tipo n, creando un campo eléctrico y una conducción estable y con dirección definida. [27], [28]

3.5. Células fotovoltaicas

La célula fotovoltaica es el dispositivo electrónico donde tiene lugar el efecto fotoeléctrico y se transforma la energía contenida en la radiación solar en energía eléctrica. Para la mayoría de los casos, se componen de una placa de silicio dopado con “unión p” por un lado y “unión n” en el otro, colocada entre una capa superior de vidrio templado y una inferior de aluminio para su protección. Existen distintos tipos y los más usados son: [1], [2]

- Células de silicio monocristalino. Fabricadas a partir de un único cristal de silicio a través de un proceso de fabricación complejo y costoso, basado en varias fases de cristalización. Presentan una estructura perfectamente ordenada que logra rendimientos entre 15 y 20%. Visualmente presentan monocromía azulada y metálica.
- Células de silicio policristalino. Fabricadas de forma similar a las monocristalinas, pero con menos fases de cristalización. Esto reduce su coste de producción y provoca que dispongan de una estructura ordenada y separada en regiones, reduciendo el rendimiento entre el 12 y 15%. Visualmente presentan una estructura de cristales con tonos azules y grises metálicos.
- Células de capa fina. Formadas por silicio amorfo a través de un proceso de fabricación más simple y menos costoso a las anteriores. No presentan una estructura cristalina ordenada y definida, lo que provoca que su rendimiento sea del 10% o menor. Visualmente presentan un tono marrón homogéneo.

Actualmente también se están desarrollando células fotovoltaicas con Arsenio en combinación con Galio o Diseleniuro de Cobre, en sustitución del Silicio. Esto se debe a que el nuevo material mantiene sus propiedades a temperaturas elevadas y obtiene rendimientos del 27% con espesores finos, con el inconveniente de necesitar un proceso de fabricación complejo y costoso. Otro tipo de células en desarrollo, son las de película delgada. Estas se componen por la unión de Cadmio (CdS) y Sulfuro cuproso (Cu2S), gracias a los cuales se consigue la absorción de radiación solar con menos material activo y espesores de tan solo 0,001mm. Esto provoca que su coste sea inferior al resto de células y da la posibilidad de automatizar su proceso de fabricación, razones que permiten ser una buena opción para determinadas aplicaciones, pese a presentar rendimientos del 5%. [1], [13]

3.6. Módulos fotovoltaicos

Un módulo fotovoltaico, también denominado panel solar, está compuesto por la asociación de celdas fotovoltaicas conectadas entre sí. Su forma de interconexión varía en función del valor final de tensión y corriente necesarios, pudiendo ser en serie y/o en paralelo, y todas las células asociadas deben tener los mismos parámetros eléctricos para evitar descompensaciones en el sistema.

Al llevar a cabo la asociación en serie, se consigue aumentar el valor de tensión manteniendo el mismo valor de corriente y realizando la asociación en paralelo, se consigue mantener estable la tensión aumentando la intensidad (Ilustración 3.2). Por ello, el método más usado es asociar células en serie hasta el voltaje deseado y posteriormente asociar en paralelo el conjunto de células en serie hasta el valor de corriente deseado. Este proceso es igualmente utilizado a la hora de conectar varios módulos fotovoltaicos en una instalación. [13]

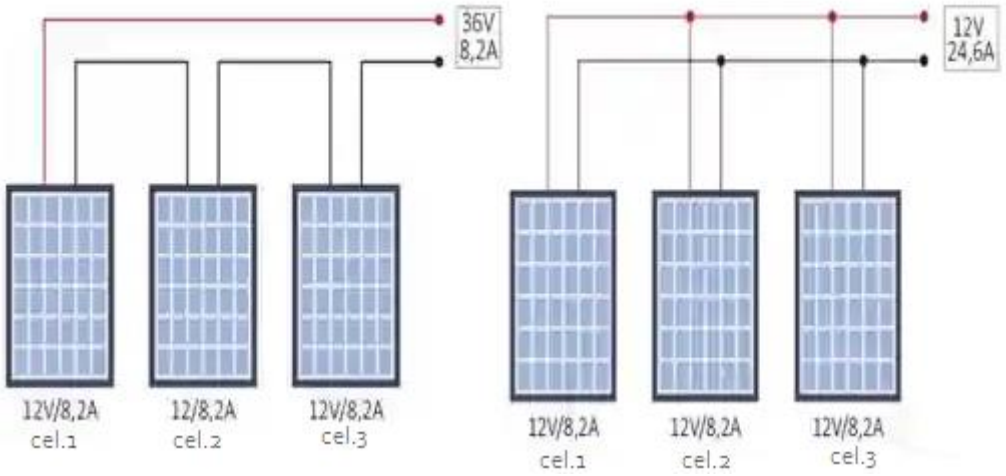

Ilustración 3.2 Formas de interconexión células fotovoltaicas *Fuente: Blog de Energías renovables V.Baeschlin*

Existe una gran variedad de módulos fotovoltaicos en función de sus parámetros eléctricos, tamaño y composición.

3.7. Tipos de inversores fotovoltaicos

Uno de los dispositivos más importantes dentro de las instalaciones fotovoltaicas, es el inversor. Este se encarga de transformar la corriente continua, que recibe de los paneles fotovoltaicos, en corriente alterna para consumo, almacenamiento o vertido a la red. [2]

En la actualidad hay diferentes tipos de inversores en función del tipo de instalación y el destino de generación, pero principalmente hay dos tipos de inversores: Inversores centrales y micro inversores. [29]

Ilustración 3.3 Comparativa de rendimiento de inversores Fuente: Cambio energético

3.7.1. Inversor central

Se conectan varias series de paneles solares a cada inversor. De este modo, se agrupan varios paneles en serie, para unirse todas estas series en paralelo y conectarse al inversor. Con esto se consigue reducir el coste del sistema, al necesitar un solo inversor, así como reducir la posibilidad de fallos e cableado. Sin embargo, en caso de que un panel de la serie reduzca su producción por fallo o mal funcionamiento, el resto de los paneles igualaran la producción de este (ilustración 3.3). Por tanto, la eficiencia del sistema será menor. Además, no podremos monitorear cada panel de forma independiente, únicamente se podrá visualizar el rendimiento de cada serie.

3.7.2. Micro inversor

Se conecta un inversor a cada uno de los paneles, siendo más elevado el coste de la instalación. L salida de cada inversor de conecta en paralelo con el resto. Por tanto, si uno de los paneles sufre una reducción en su producción, no afectará al resto y la eficiencia del sistema será mayor que en el caso de los inversores centrales (ilustración 3.3). Además, con este tipo de inversores se puede realizar el monitoreo de cada panel.

3.8. Curvas características de una célula fotovoltaica

El funcionamiento de las células fotovoltaicas se estudia a partir de dos curvas que representan la relación entre corriente o potencia generada entre los bornes del panel, frente al voltaje que aparece entre estos (Ilustración 3.4). Dada la relación entre la potencia y la intensidad generada en las células fotovoltaicas, el estudio se centra sobre la curva I-V. [1]

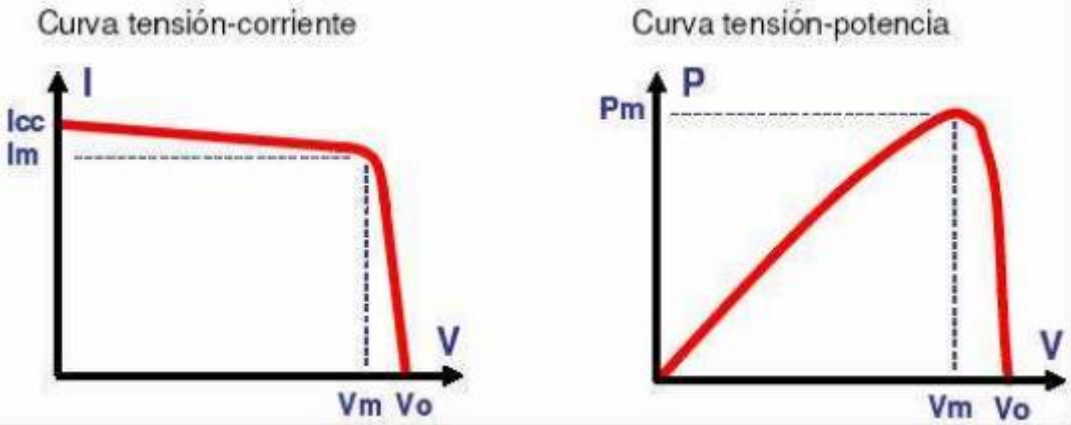

Ilustración 3.4 Curvas características células fotovoltaicas. Curva I-V y Curva P-V Fuente: Ingelible

3.8.1. Curva Corriente-Tensión (I-V)

La curva I-V, viene dada por la suma algebraica de la corriente fotogenerada (I_L), debida a la generación de portadores al incidir radiación, y la corriente de oscuridad o de diodo (I_D), debida a la recombinación de portadores en la unión p-n. Por tanto, la ecuación fundamental de la corriente de generación (I) en una célula fotovoltaica viene expresada por:

$$I = I_L - I_D$$

Tomando como referencia de signos una corriente de generación positiva entregada a una carga bajo una tensión positiva, se obtiene la curva característica I-V (Ilustración 3.5), juntos a sus parámetros fundamentales:

- I_{SC}, Corriente de cortocircuito: Máximo valor de corriente que entrega la célula. Se obtiene en condiciones de cortocircuito, a tensión nula.
- V_{OC}, Tensión en circuito abierto: Máxima tensión que entrega la célula. Se obtiene en condiciones de circuito abierto, a intensidad nula.
- MPP, Punto de máxima potencia o Potencia de pico: Máxima potencia que puede entregar la célula
- V_{mp}, Tensión a máxima potencia o nominal: Tensión entregada por la célula en condiciones de máxima potencia.
- I_{mp}, Corriente a máxima potencia o nominal: corriente entregada por la célula en condiciones de máxima potencia.

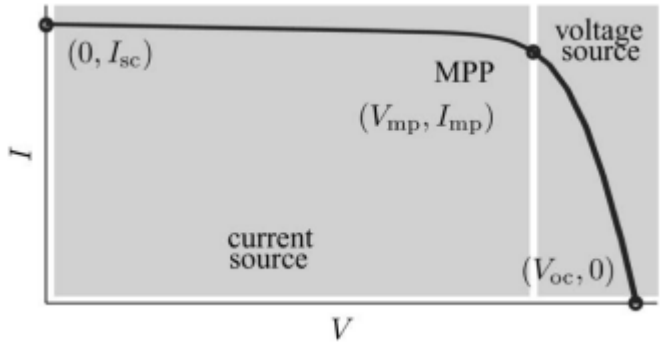

Ilustración 3.5 Curva I-V y sus parámetros fundamentales *Fuente: ResearchGate*

Con los parámetros obtenidos en la curva I-V se obtiene el factor de forma de la célula fotovoltaica (FF), indicador de la calidad de la célula en términos energéticos. Relaciona la potencia máxima capaz de ser entregada por la célula y la potencia máxima real que puede entregar. Es el resultado del cociente entre la potencia máxima y el producto de la tensión de circuito abierto y la intensidad de cortocircuito, siendo más eficiente la célula, cuanto más cercano sea su FF a la unidad. Generalmente su valor oscila entre 0.7 y 0.8, y su expresión es de la forma:

$$FF = \frac{I_M * V_M}{V_{OC} * I_{SC}}$$

Un condicionante del funcionamiento de las células fotovoltaicas, y por tanto de sus parámetros fundamentales, son las condiciones meteorológicas a las que se exponen, así como la radicación solar incidente. [30]

3.8.2. *Influencia de la radicación solar*

La corriente generada por la célula fotovoltaica es proporcional a la intensidad de radiación solar y a la superficie de la célula expuesta a esta. Cuanta menor luz sea recibida o menor sea la intensidad de la radiación, menor intensidad y potencia son generadas y, por tanto, menor rendimiento, mientras los valores de tensión descienden

en menor medida y cercanos a valores de circuito abierto, pudiendo considerarse despreciables. Estos efectos se muestran en la ilustración 3.6. [2], [30]

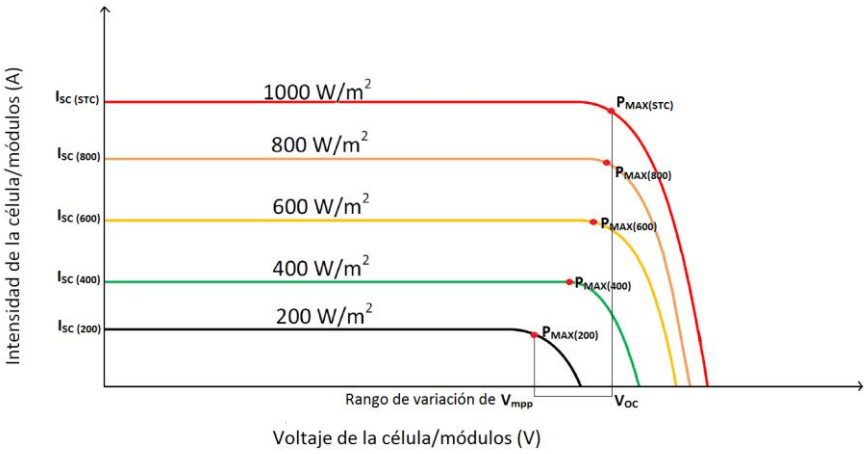

Ilustración 3.6 Efecto de la radiación solar sobre la potencia generada en célula fotovoltaica *Fuente: Ingelibre*

3.8.3. Influencia de la temperatura

La tensión en bornes de una célula fotovoltaica es función de la temperatura a la que se encuentra la célula (temperatura de trabajo), de forma que, al aumentar la temperatura de trabajo, se produce una caída de tensión, mientras la corriente generada permanece prácticamente constante. Al disminuir la tensión, desciende la potencia entregada y por tanto el rendimiento de la célula. El efecto es mostrado en la ilustración 3.7. [2], [30]

La temperatura de trabajo depende de la temperatura ambiente, así como de la irradiación, y su expresión matemática es la siguiente:

$$T_C = T_a + G * \frac{TONC - 20}{800}$$

Donde: T_C es la temperatura de trabajo (ºC), T_a es la temperatura ambiente(ºC), TONC es la temperatura de operación nominal (ºC) y G la irradiancia (W/m^2).

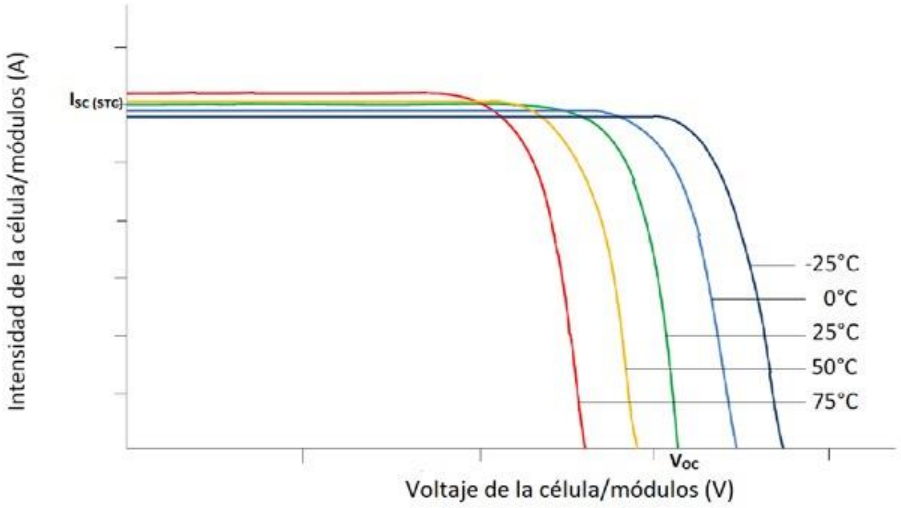

Ilustración 3.7 Efecto de la temperatura sobre la potencia generada en célula fotovoltaica *Fuente: Ingelibre*

3.9. Automatización industrial 4.0

Según lo expuesto en el apartado 3.2 del presente estudio, estamos inmersos en una nueva revolución industrial centrada no solo en aportar soluciones en la mecanización, sino en implantar sistemas de control e información capaces de gestionar procesos cada vez más complejos y flexibles, de forma más eficiente y rentable. [26] La automatización industrial 4.0 surge como solución a estas necesidades de la industria actual, ante la imposibilidad de solucionarse con procesos manuales.

3.9.1. *Pirámide de automatización, Modelo CIM*

Para que los procesos automatizados puedan desarrollar todo su potencial, se requieren flujos de información a través de todos los componentes que forman parte de los procesos industriales, integrando los procesos de producción con los procesos de gestión de empresa. Por esta razón, se establece un sistema de control jerarquizado: Pirámide de automatización, CIM (Ilustración 3.8). [31]

Ilustración 3.8 Pirámide de Automatización Fuente: Escuela de ingenierías de León

Se establecen 5 niveles, de forma que cada nivel lleve a cabo labores específicas y realice un tratamiento y filtrado de la información entre los niveles superiores e inferiores. Con esto se consigue limitar los flujos de información a los estrictamente necesarios en cada nivel. Dentro de este trabajo, únicamente se tratarán los tres niveles inferiores: [31]

- Nivel de campo: Adquisición de datos y ejecución de acciones en función de algoritmos de control y consignas del nivel superior. Formado por dispositivos de operación elementar (sensores, actuadores…).
- Nivel de control: Mando y control de elementos de campo. Formado por sistemas electrónicos donde se ejecutan algoritmos y cálculos de gestión de información. (PLC, PC industrial, Microcontroladores…).
- Nivel de supervisión: Tratamiento de datos, monitorización, gestión de alarmas y mantenimientos, control de calidad y seguimiento de órdenes de fabricación. Supervisa las unidades productivas funcionales.

3.9.2. *Sistemas SCADA*

SCADA (Supervisory Control And Data Acquisition, o en castellano, Adquisición de datos y supervisión de control) es una aplicación software para el control de producción, realizando la comunicación entre los dispositivos y controlando el proceso desde una estación remota [32]. Para ello utiliza una interfaz gráfica que muestra el comportamiento a tiempo real del proceso y desde donde se llevan a cabo funciones de diálogo con el operador para establecer ordenes sobre la unidad de control (Nivel de

control). En la ilustración 3.9 se puede apreciar el esquema básico de un sistema de adquisición, supervisión y control, con los componentes de los tres niveles que intervienen y sus flujos de información.

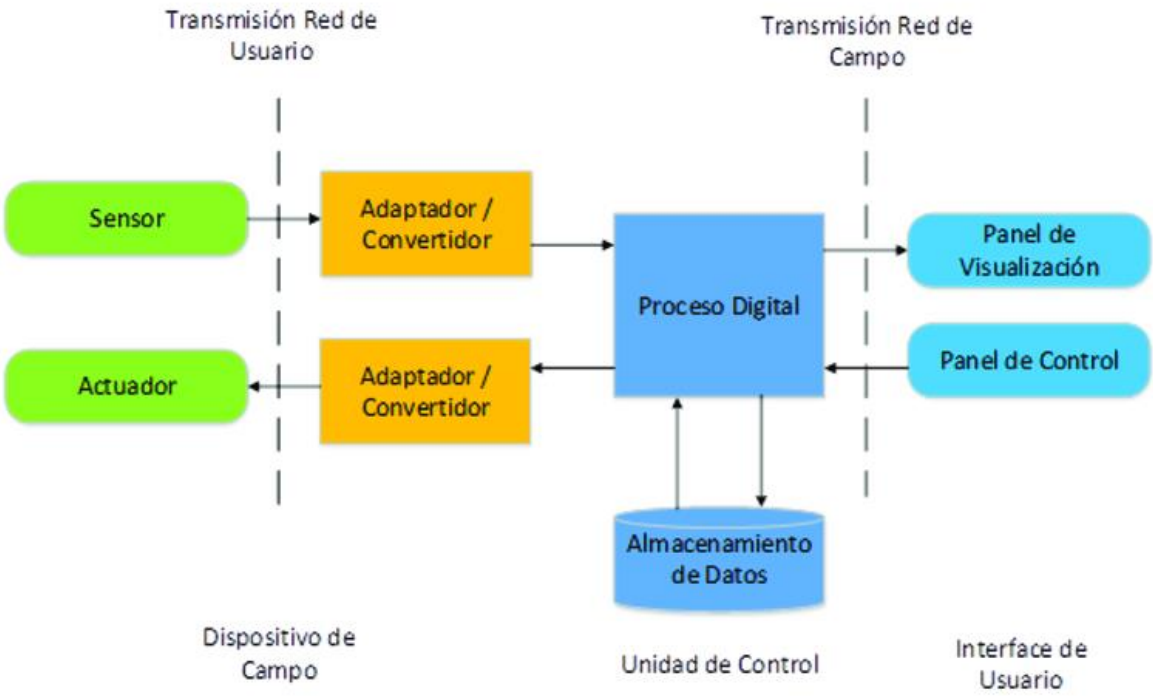

Ilustración 3.9 Esquema básico de un sistema SCADA *Fuente: Propia*

Las funciones principales de los SCADA [31], [32] son:

- Adquisición de datos: Recoger, procesar y almacenar la información recibida.
- Supervisión: Observar desde interfaz gráfica la evolución de las variables de control.
- Control: Modificar el comportamiento del proceso actuando en el nivel de control.
- Transmisión de información con dispositivos de campo y de control.
- Base de datos.
- Representación gráfica de los datos.

3.10. Sistemas de monitorización solar

En la industria fotovoltaica los sistemas de monitorización nos permiten verificar el correcto funcionamiento de la instalación de forma remota y a tiempo real. Con esto se consigue actuar rápido en caso de detectar daños y/o averías, además de mejorar el

rendimiento de la instalación, analizando las diferentes variables que influyen en la producción fotovoltaica.

Actualmente existen múltiples sistemas para monitorización de sistemas fotovoltaicos. En la mayoría de los casos, estos sistemas se integran en los inversores (dispositivo que convierte la corriente continua en alterna) recopilando información sobre el estado de la instalación para ser enviada al sistema de monitorización. Existen dos tipos de monitorización a través del inversor:

- Monitorización remota: El inversor tiene conexión a internet y guarda la información en la nube.
- Monitorización local: Para visualizar los datos es necesario conectarse a la red interna del inversor.

Estos sistemas pueden ser universales, pudiendo funcionar con diferentes marcas y tipos de inversores, o exclusivos, únicamente para ciertas marcas. Dentro de los universales, los más utilizados son: Solar-Log [33], Scada solar [34] o Abora Monitor [35]. Todos ellos cuentan con características similares; recopilan información de variables del sistema, elaboran informes de producciones e históricos, reflejan fallos en la instalación, permiten la conexión a la plataforma a través de diferentes equipos y permiten conexión de inversores de diferentes marcas. Su precio es variable ya que permiten adaptar los sistemas a cada cliente, estableciendo un mínimo de presupuesto de 570 euros, como es el ejemplo de Solar-Log 300 [33].

3.11. Protocolo MQTT

MQTT (Message Queue Telemetry Transport) es un protocolo de código abierto, inventado por Andy Stanford-Clark y Arlen Nipper en 1999 para funcionar con un ancho de banda mínimo y una pérdida de batería mínima, en la monitorización de oleoductos en un SCADA industrial [36].

Se considera un protocolo Machine to Machine (M2M) [37] diseñado con una dinámica publicación/suscripción muy ligera, lo que convierte a MQTT en el protocolo perfecto para IoT y aplicaciones donde se requiera minimizar el consumo energético o el ancho de banda disponible sea reducido [38]

Funciona sobre el protocolo de red TCP/IP [39] con una arquitectura definida Servidor-Cliente, donde el dispositivo que realiza la publicación o suscripción es el cliente y el bróker o gestor MQTT es el servidor. En cuanto a la seguridad en la

transmisión de información, proporciona integridad de datos, autenticación y confidencialidad a través de TLS. [40], [41]

La transmisión de información en el protocolo MQTT se realiza en forma de paquetes o mensajes [22]. Estos mensajes se publican bajo un tópico, en vez de enviarse a cada uno de los dispositivos que necesitan la información, y aquellos dispositivos suscritos al tópico, tendrán acceso a la información. De este modo, no es necesario conocer la dirección de los dispositivos y tampoco se necesita que haya un suscriptor en todo momento, lo que permite escalar el sistema a millones de dispositivos. [42]

Sin embargo, los protocolos TCP/IP requieren el direccionamiento para la transmisión de información, necesitando indicar la dirección IP destino para que los routers puedan encaminar el mensaje [43]. Por esto, se dispone de un bróker en la red MQTT encargado de hacer que los mensajes lleguen al destino. Todos los mensajes se envían desde los dispositivos al bróker y este se encarga de reenviar cada mensaje a todos los dispositivos que estén suscritos al tópico correspondiente al mensaje. [42]

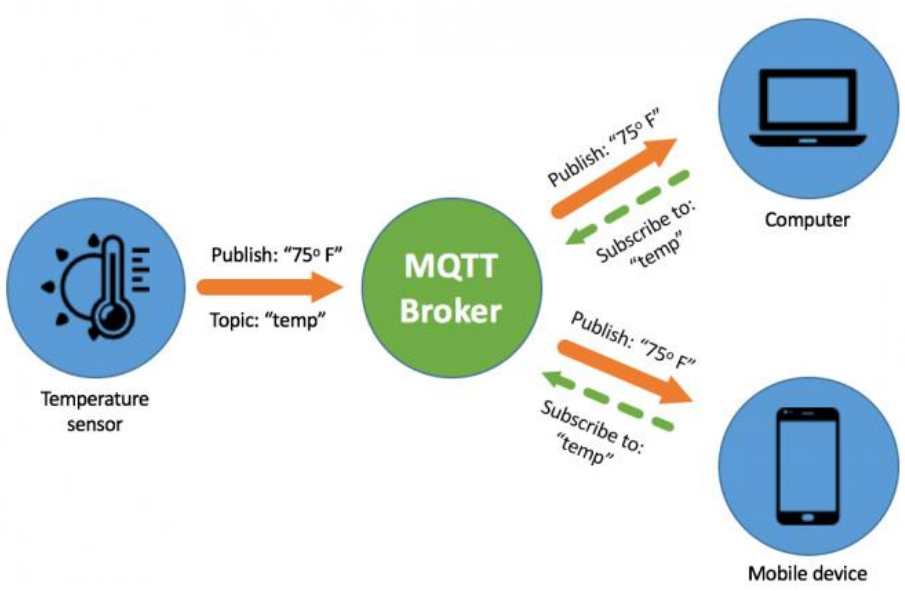

Ilustración 3.10 Dinámica de transmisión de información protocolo MQTT

En la ilustración 3.10 se observa como el bróker MQTT recibe la suscripción del ordenador y el móvil al tópico "temp" y, por tanto, al recibir un mensaje del dispositivo publicador, que es el sensor de temperatura, bajo el mismo tópico "temp", envía la información a ambos dispositivos suscritos.

3.11.1. *Cliente y Bróker*

Como ya se ha explicado, dentro del protocolo MQTT disponemos de dos elementos: el cliente y el bróker:

- El cliente MQTT puede ser cualquier dispositivo conectado a un broker, puede ser tanto un ordenador, móvil o un microcontrolador. Su función es suscribirse a un tópico, publicar mensajes o ambas funciones a la vez. A la hora de aplicar un cliente MQTT se encuentran librerías en diferentes lenguajes de programación, lo que da al protocolo sencillez y flexibilidad.
- En cuanto al bróker, es el dispositivo al que se conectan todos los clientes de la red para realizar intercambios de información. Su función es hacer llegar ellos mensajes a todos los clientes suscritos a cada tópico, verificando al cliente que se conecta.

El protocolo se basa en TCP/IP. Es decir, la conexión se realiza siempre desde el cliente al bróker. El cliente envía un mensaje al bróker solicitando la conexión (connect) y el bróker responde con un mensaje sobre el estado de la conexión (connaback). Durante el mensaje “connect” el cliente envía su usuario y contraseña, cifrados a través de TLS o SSL, anteriormente citadas en este apartado. Tras realizarse la conexión, el cliente está conectado al bróker hasta que no envíe un mensaje de desconexión al bróker.

3.11.2. *Topic*

Para que el bróker pueda diferenciar los mensajes que recibe y guiar estos hacia el cliente adecuado, cuenta con los topic o tópicos. Los clientes, envían sus mensajes al bróker realizando una publicación asociada a un tópico, además de suscribirse a los tópicos publicados por otros clientes. El bróker conoce a que tópico está suscrito cada uno de los clientes, pudiendo subscribirse a varios tópicos, y envía la información a los clientes suscritos al tópico asociado a cada mensaje. Este proceso de suscripción/publicación, se puede observar en la ilustración 3.10.

3.11.3. QoS (Quality of Service)

Para medir la garantía de entrega de los mensajes y establecer la forma en que se asegura la recepción de los mensajes, se usa QoS o Calidad del Servicio en Castellano. Se establecen tres niveles numerados, de modo que cuanto mayor sea el QoS, más fiable es el envío y recepción de mensajes.

- QoS 0: El mensaje se envía una vez y no se recibe confirmación de entrega. Es el que menos recursos requiere, pero no da fiabilidad.
- QoS 1: Se garantiza que el mensaje llega al receptor, al enviarse todas las veces necesarias hasta obtener un mensaje de confirmación de entrega (puback).
- QoS 2: Se realiza una confirmación de conexión preparada entre emisor y receptor, así como una confirmación de envío y recepción. Consume muchos más recursos, por lo que el rendimiento es menor y aumenta la latencia.

En el caso de publicar un emisor, un mensaje con un QoS 2, y hay un receptor suscrito con QoS 1 y otro con QoS 2, cada receptor recibirá el mensaje con el QoS con el que está suscrito. De este modo, el primero recibe mensaje con QoS 1 y el segundo recibe el mismo mensaje, pero con QoS 2.

3.12. Base de datos MySQL

En cuanto a la definición general, MySQL [44] es un sistema RDBMS (Relational Database Management System) de gestión de bases de datos relacionales de código abierto con un modelo cliente-servidor y multiplataforma. Es una base de datos desarrollada y gestionada por Sun Microsystems. Para conocer en profundidad MySQL hay que analizar cada uno de los términos de la definición:

- RDBMS: es un software o servicio para administrar y crear bases de datos basadas en un modelo relacional.
- Código abierto: es de libre uso, cualquiera puede usar el software personalizando el código fuente para adaptarse a diversas necesidades. Sin embargo, trabaja con licencia GPL (General Public License) que determina sus usos según las condiciones.
- Modelo cliente-servidor: los dispositivos donde se instala y ejecuta el software RDBMS son clientes. Estos se conectan a un servidor RDBMS para acceder los datos.
- Multiplataforma: puede instalarse en entornos con sistemas operativos Windows, Mac, Linux o ambientes Unix.
- Relacional: los datos almacenados se organizan en forma de tablas.

Para comunicar el cliente y el servidor en un entorno RDBMS, es decir, para interactuar el cliente con la base de datos, usa el lenguaje de consulta estructurado o SQL (Structured Query Language). Con esto, se pueden realizar diferentes acciones:

- Consultar datos existentes.
- Manipular los datos agregando, eliminando, cambiando u ordenando los datos y sus valores.
- Definir los tipos de datos.
- Controlar el acceso a los datos a través de técnicas de seguridad.

Dentro del mercado, MySQL no es el único RDBMS, pero si el más popular. Esto se debe a que es flexible y fácil de usar, es seguro y de alto rendimiento. Además, cuenta con varias plataformas del lado del cliente para su gestión y visualización, contando con plataformas como MySQL WorkBench [44], SequelPro o incluso administradores web como phpMyAdmin [45], que permiten trabajar en la nube o dentro de una red local.

4. Diseño general del sistema solución

Previo a la elección de componentes hardware y software implementados en el sistema, es necesario realizar un primer diseño general del sistema aportado como solución. Con este diseño se logra conocer las características requeridas de cada componente y enmarcar la función de cada uno de ellos dentro del sistema, explicando más a fondo en los apartados 5 y 6 las especificaciones de cada componente.

Partiendo de los objetivos generales descritos en la introducción del trabajo, podemos dividir el sistema en cinco bloques: Toma de datos del sistema fotovoltaico, Captura de imágenes del sistema, Comunicación MQTT, Servidor de gestión y almacenamiento de información, e Interfaz de supervisión y visualización.

4.1. Toma de datos del sistema fotovoltaico

Según lo expuesto en el apartado 3.4, la energía fotovoltaica se basa en la generación de una fuerza electromotriz que provoca la aparición de una tensión y corriente de carácter continuo entre los bornes de un panel fotovoltaico. Por tanto, para analizar el funcionamiento del sistema fotovoltaico, es necesario conocer el valor de estos parámetros eléctricos en cada momento, y construir las curvas características expuestas en el apartado 3.8. Dentro de este mismo apartado, se analizan los factores externos al sistema, que influyen sobre estos parámetros eléctricos y los efectos que provocan. Uno de estos factores es la temperatura, y es necesario conocer sus valores para realizar la supervisión del sistema.

La medida de estos parámetros y factores se realiza a través de sensores analógicos. En cuanto a los sensores de voltaje e intensidad, deben realizar la medida de cualquier valor que este dentro del rango de funcionamiento del panel. El sensor de temperatura debe medir valores de temperatura dentro de un rango de temperatura amplio, con temperaturas que puedan bajar de 0ºC, así como superar los 35ºC, al realizar la medida de temperatura alcanzada en los paneles.

Todos los sensores deben otorgar un valor de salida analógico que pueda ser leído por el elemento que capte, gestione y adecue estas señales a valores estandarizados. Este elemento será un microcontrolador, con comunicación Wifi-incorporada para realizar el envío de información a la nube a través del protocolo MQTT.

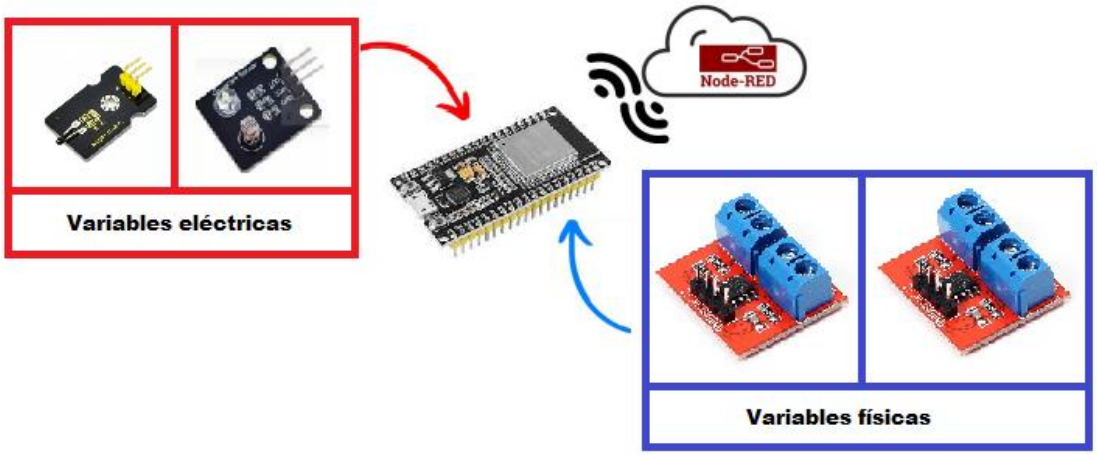

Ilustración 4.1. Esquema visual sistema de captación de variables Fuente: Propia

En la ilustración 4.1 se puede observar el esquema del bloque de captación, en el cual, aparecen los diferentes dispositivos elegidos. Como elemento central se encuentra el microcontrolador ESP32 al cual llegan las señales del sensor de voltaje FZ0430, del sensor de corriente ACS712 y del sensor de temperatura KY-028. Una vez se procesan estas señales, envía los datos al servidor a través de la nube.

4.2. Captura de imágenes del sistema

Uno de los trabajos realizados dentro de un sistema de producción fotovoltaica es la revisión visual del estado de los paneles, comprobando los niveles de suciedad que presentan o si alguno de estos presenta daños físicos o mecánicos. Con esto, se pretende establecer si es necesario realizar reparaciones o procesos de limpieza sobre los paneles, evitando pérdidas en la producción, al provocar estas situaciones, efectos negativos sobre el rendimiento. Este trabajo es costo en cuanto a recursos económicos y humanos, ya que muchas veces el acceso a los paneles es complejo al situarse normalmente en zonas elevadas.

Para solucionarlo, se instala dentro del sistema una cámara que permita tomar imágenes de los paneles cada vez que sea necesario. La cámara elegida para este proceso ha de ser capaz de conectarse a la red Wifi para realizar envío de imágenes y recibir órdenes para su control, a través del protocolo MQTT. Esta orden se entrega al sistema a través de una interfaz humano-máquina. Además, debe disponer de resoluciones de imagen aceptables.

Por tanto, en el presente trabajo se implantará un módulo ESP32 CAM. Este módulo es en esencia un microcontrolador ESP32, similar al usado en la captación de

variables, al que se agrega una cámara OV2640. Por tanto, incorpora comunicación Wifi y recursos para establecer el protocolo.

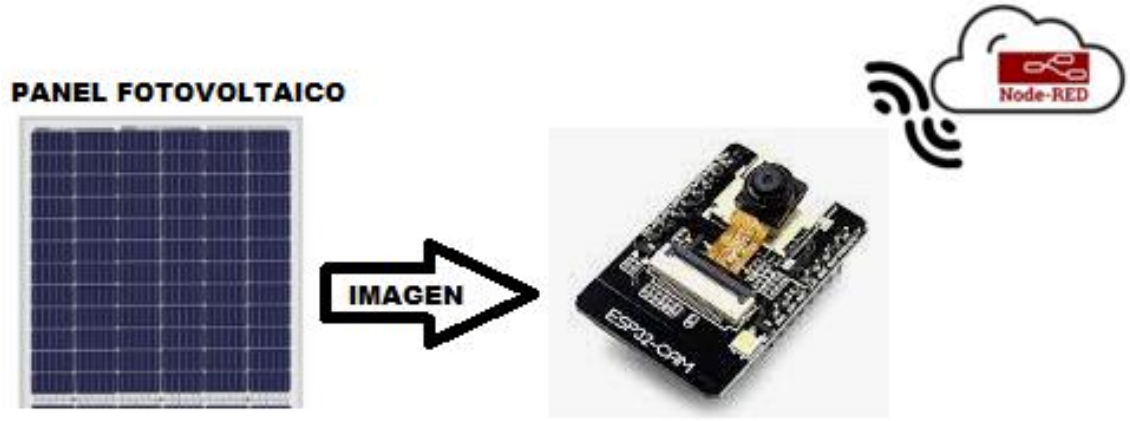

Ilustración 4.2 Esquema visual sistema de captura de imagen *Fuente: Propia*

4.3. Comunicación MQTT

Para poder realizar la transmisión de los datos extraídos en el sistema de captación, es necesario establecer una comunicación entre el sistema de captación y el servidor. Así mismo, es necesario establecer comunicación entre los dispositivos de la nube y el servidor, y así visualizar los datos. El modo en que el sistema diseñado realiza estas comunicaciones es por medio del protocolo MQTT sobre TCP/IP, explicado en el apartado 3.10. De este modo, se establece una red WLAN a la que se conecta nuestro servidor, el microcontrolador a través de su módulo Wifi integrado, y los dispositivos desde donde visualizar los datos.

Este protocolo requiere del uso de un bróker que gestione la transmisión de mensajes y el cual irá alojado en el servidor, al ser el nodo central del sistema, y al cual todos los dispositivos estarán conectados. El bróker elegido es “Eclipse Mosquito”.

4.4. Servidor de gestión y almacenamiento de datos

Para ejercer la función de servidor, se dispone de un PC con sistema operativo Linux Ubuntu. Dentro de este equipo se implanta una base de datos MySQL, para la cual es necesario disponer de un software que permita la creación y gestión de esta. El software elegido es “Apache Friends Xampp”. Además, los mensajes que se intercambian en el sistema han de ser escritos e interpretados por todos los dispositivos. Por tanto, se necesita un enlace para todos los dispositivos hardware, API y servicios que componen la solución. Para ello, se implementa un web service de procesamiento

de información, así como gestión de envío y recepción de mensajes. El web service elegido es Node-RED, y se instala sobre el PC servidor.

Como se explica en el apartado 6.5 junto a su funcionamiento, Xampp incluye la base de datos MySQL, el servidor web Apache, el intérprete de lenguajes script PHP y el administrador phpMyAdmin. La integración de estos programas forma un estándar para operar una base de datos MySQL a través de la nube, o como el caso del presente proyecto, operar dentro de una red de área local.

En definitiva, se implanta dentro de nuestro servidor una base de datos MySQL, que administraremos a través de phpMyAdmin y visualizaremos a través del servidor web HTTP Apache. De este modo, se recogen los datos extraídos por el sistema de captación y se direccionan hacia la base de datos, donde se crearán tablas con datos históricos del sistema y hacia una interfaz de visualización. Para realizar este direccionamiento se usa la herramienta Node-RED explicada en el apartado 6.2.

4.5. Interfaz de supervisión y visualización

La visualización de los datos históricos se realiza a través de la propia interfaz humano-máquina de phpMyAdmin, donde se visualizan los datos de forma relacional. Dentro del web service usado en la solución, se incluye una interfaz de visualización llamada “Dashboard” que será la usada en el trabajo. Está se configura a través de la misma plataforma de configuración para Node-RED y será accesible para cualquier dispositivo con acceso al servidor. Dentro de ella se visualizarán los datos medidos a tiempo real, así como curvas y gráficas donde se caracteriza el estado de los valores durante los intervalos de funcionamiento. Además, se añadirán alertas sobre errores producidos durante el funcionamiento del sistema o funcionamientos anómalos. También se podrá solicitar a través de la interfaz la captura de imágenes del sistema y visualizarlas.

Una vez analizados los cinco bloques que componen la solución se establece la estructura de la solución, ilustración 4.3.

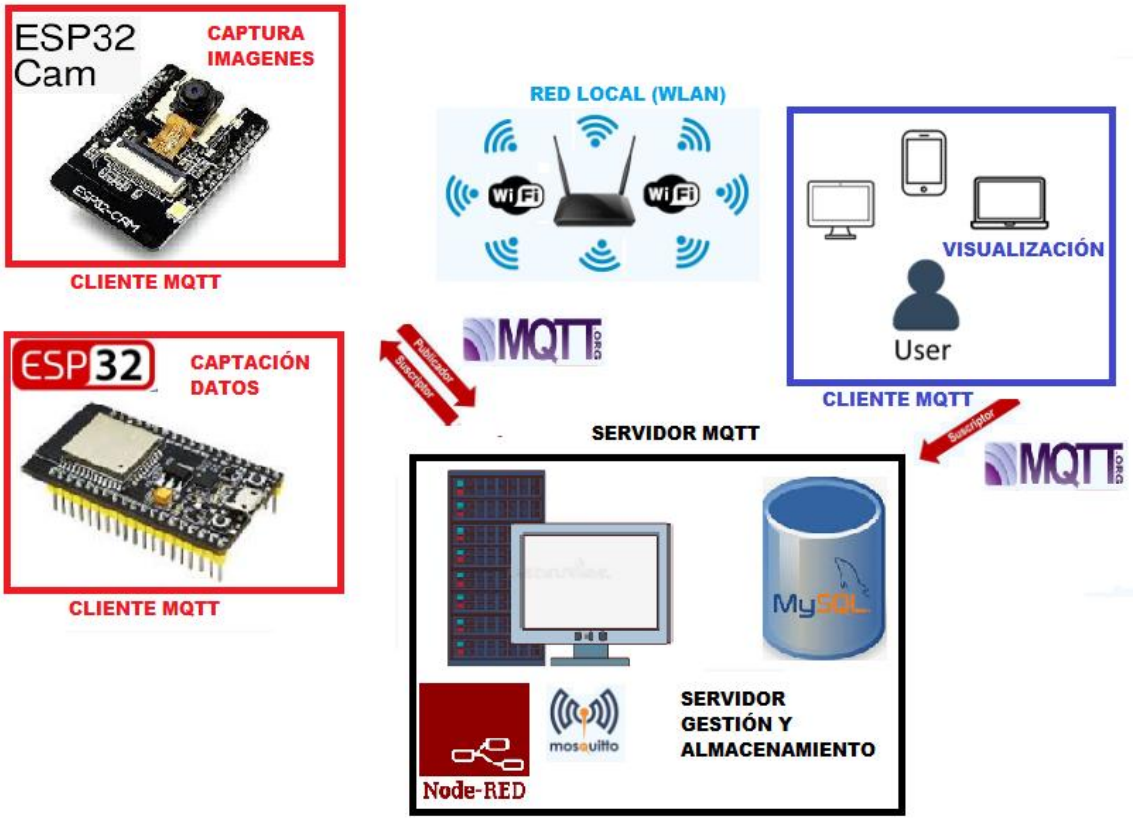

Ilustración 4.3 Estructura del sistema solución Fuente: Propia

5. Hardware utilizado en el estudio

Para llevar a cabo la captación y monitorización a tiempo real de los parámetros de funcionamiento de paneles fotovoltaicos, así como para gestionar y almacenar los datos extraídos, son necesarios diversos componentes electrónicos que compondrán el sistema de supervisión y monitorización.

5.1. Microcontrolador

El microcontrolador utilizado es el ESP32 en su versión modular ESP32-WROOM-32 [46], agregado sobre una placa de desarrollo para facilitar su uso. En la ilustración 5.1 se puede observar físicamente este módulo.

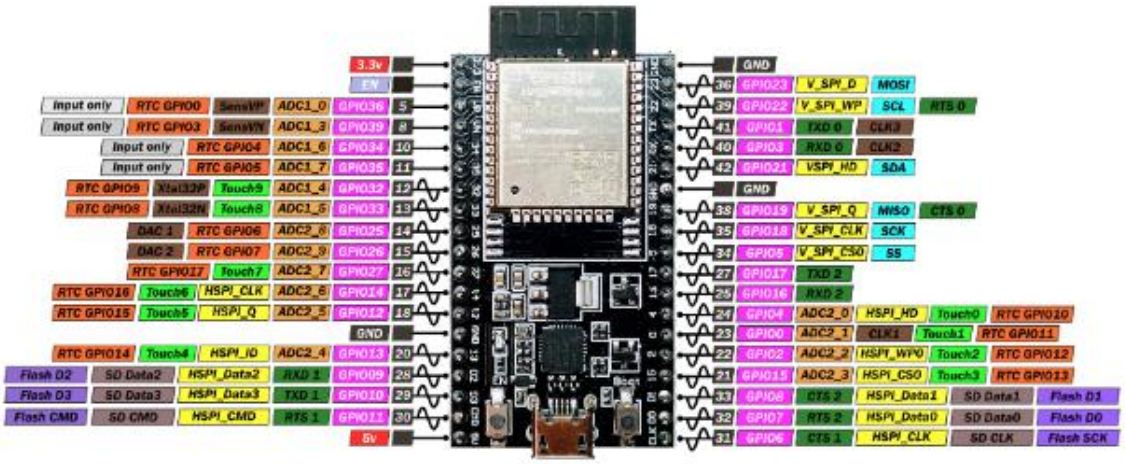

Ilustración 5.1 Pinout ESP32 WROOM32 Fuente: Espressif

Este microcontrolador posee conectividad Wifi al incorporar un Transceptor WiFI 802.11 BGN y conexión Bluetooth, permitiendo ser conectado en redes locales de forma sencilla, algo con lo que otros microcontroladores no cuentan y para lo que precisan añadir módulos externos de comunicación. Su diseño es robusto y preparado para ambientes industriales, contando con un rango de tolerancia de temperatura se sitúa entre los -40ºC y los 85ºC.

Se encuentra dentro de los microcontroladores de propósito general más potentes del mercado, al contar con una estructura de 32 bit con dos núcleos de procesamiento, un reloj de 2,4 GHz de frecuencia y una memoria Flash de 4MB. Además, el módulo cuenta con 38 pines de propósito general (GPIO) correspondientes a pines de alimentación, de control y pines de entrada/salida digitales y analógicos, permitiendo realizar la lectura de hasta 16 sensores analógicos.

Además, la placa de desarrollo incorpora un convertidor USB-UART CP2102 que permite la conexión y programación del módulo ESP32-WROOM-32 mediante USB. Esto permite no requerir un programador externo. Este circuito integrado tiene la finalidad de convertir las señales del puerto USB (nivel lógico 5 Voltios) y su protocolo de comunicación, a señales de nivel lógico 3.3 Voltios con protocolo de comunicación UART. Con esto se consigue evitar daños en el microcontrolador por sobretensiones en las entradas de comunicación, ya que posee una tensión máxima de funcionamiento de 3.6 Voltios. El protocolo UART, fácilmente interpretable por los microcontroladores, logra simplificar la comunicación a través del puerto serial, aunque es necesario instalar drivers, disponibles en la página del fabricante [47], para la comunicación entre el circuito integrado y el ordenador.

5.1.1. *Lectura de entradas analógicas*

Dentro de los aspectos técnicos del microcontrolador, es necesario analizar el modo en que se lleva a cabo la lectura de valores de entrada analógicos, ya que se usará durante el desarrollo del presente trabajo.

El ESP32 no tiene capacidad para realizar lecturas de señales analógicas, por lo que necesita convertir estas señales a señales digitales. Para ello emplea dos ADC (Conversor Analógico/Digital) con una resolución máxima de 12 bit, lo que hace que obtenga un valor entero entre 0-4095, para cada valor de tensión de entrada. Este valor entero, se convertirá a binario, para poder ser procesado. Como ejemplo, dado un sensor que otorga una señal de 1.65V obtenemos un entero de 2047. El valor de la entrada puede variar entre 0 V y 3.3 V.

Cabe destacar que una vez iniciada la conexión Wifi, únicamente puede usarse el ADC1 correspondiente a las 6 primeras entradas analógicas. Esto se debe a que el ADC2 se reserva para la comunicación.

5.2. Módulo ESP32 CAM

Según lo expuesto en el apartado 4.2, para la captura de imágenes de los paneles que componen el sistema se utiliza el módulo ESP32 CAM del fabricante AI-Thinker [48]. Se trata de un pequeño módulo basado en el chip ESP32 y que integra una cámara OV2640 y conexión para tarjeta microSD de hasta 4GB. Esto hace que las características del procesador y de la comunicación Wifi y bluetooth, sean idénticas a las expuestas para el módulo ESP32 WROOM32 del apartado 5.1. Únicamente varia la velocidad de reloj a 1.6 GHz.

Ilustración 5.2 Módulo ESP32 CAM Fuente: AI-Thinker

A diferencia del módulo ESP32 del apartado 5.1, únicamente cuenta con 9 GPIO, ya que muchos de los pines del chip los usa para la gestión de la cámara y el conector microSD. Además, no cuenta con convertidor USB-UART CP2102 para la conexión y programación a través de USB. Su voltaje de alimentación es de 5V.

La cámara OV2640 que incorpora posee por defecto una resolución UXGA 1622x1200 px que puede ser configurada a resoluciones inferiores, un sensor 2 Megapixel y una transferencia de imagen entre los 15 y 60 FPS en función de la resolución configurada.

5.2.1. Adaptador Serie USB a TTL

Al no disponer de puerto USB para la conexión y carga del programa desde dispositivos externos, se usa un adaptador Serie USB a TTL con chip FTDI FT232RL para realizar la comunicación entre el ESP32 CAM y el PC de programación.

5.2.2. Configuración de pines para la carga

Dentro del módulo, existen tres pines con funciones especiales. Uno de estos es el GPIO 0, el cual determina si el ESP32 CAM está listo para cargar el programa o si está ejecutándolo. Esta función al realiza gracias a su conexión con una resistencia pull-up de 10k Ohmios. Al conectar este pin a masa, el módulo pasa al modo carga hasta que se presiona el botón de reset que incorpora. Además de este pin, el GPIO 1 y 3 están preparados para realizar comunicaciones serie, estableciendo el 1 como pin TX y el 3 como pin RX. Estos pines serán conectados con los terminales del adaptador, según se indica en la ilustración 5.3.

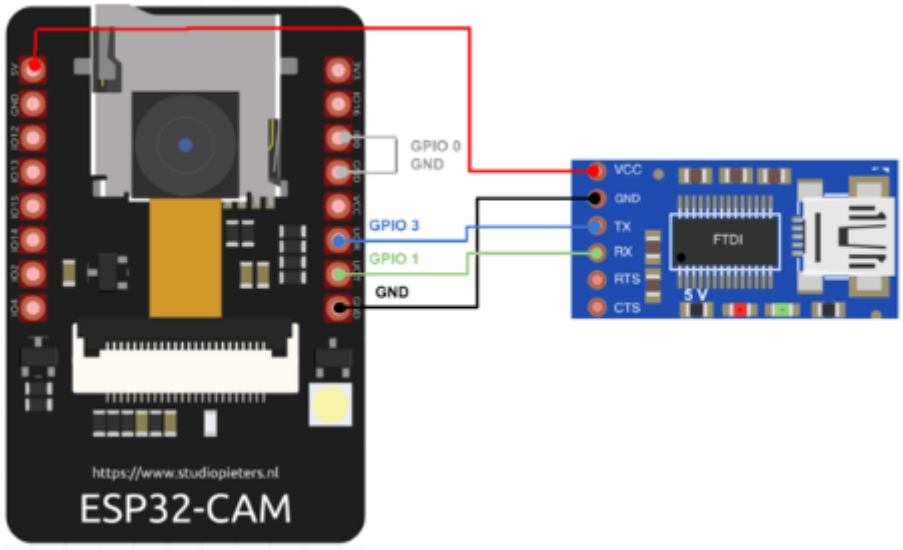

Ilustración 5.3 Conexión ESP32 CAM y Adaptador USB a TTL Fuente: Programafacil.com

5.3. Sensor de temperatura

Como anteriormente se explicó, el sensor de temperatura se implementa en el sistema para obtener la temperatura a la que se encuentra el panel fotovoltaico durante su funcionamiento. Según de su tecnología de funcionamiento, existen diferentes tipos de sensores de temperatura, capaces de generar señales analógicas o digitales según el valor de temperatura al que se encuentren.

Para este trabajo, se usará un sensor del tipo resistivo y más específicamente, un termistor NTC (Coeficiente de Temperatura Negativo). Es un tipo de sensor de temperatura pasivo, compuesto por un material semiconductor sintetizado que presenta un cambio de resistencia descendente en proporción a un cambio de temperatura positivo. Por tanto, el valor de tensión de la señal analógica del termistor disminuye con el aumento de la temperatura. [49]

Este termistor se agrega sobre una pequeña placa de circuito, dando origen al módulo KY-028 [50]. Cuenta con dos conexiones para señales de entrada, alimentación y masa, y otras dos para señales de salida, una analógica y otra digital. Además, cuenta con un potenciómetro donde ajustar el valor de consigna que produce el cambio de estado en la salida digital. La tensión de funcionamiento puede ser de 3.3V-5V, contando con un rango de medida de temperatura de -55ºC-+125ºC y una precisión de medida de ±0.5V.

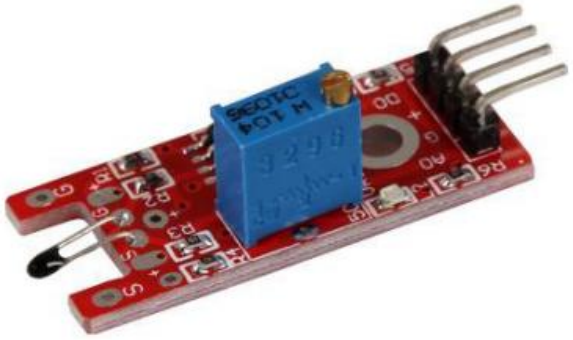

Ilustración 5.4 Módulo termistor KY-028 Fuente: AZ-Delivery

El módulo se compone de tres partes principales. Primero, la unidad de sensor en contacto con el área de medición genera una señal analógica y la envía a una segunda unidad, el amplificador. Este amplifica la señal a valores adecuados para microcontroladores Arduino y la envía a la salida analógica de la placa. El tercer componente es un comparador, cambia el estado de la señal digital si el valor de la señal del amplificador cae por debajo del valor de consigna establecida en el potenciómetro. La señal analógica la obtiene midiendo la caída de tensión en el termistor, tras enviar una pequeña corriente a través de él. [50]

5.4. Sensor de voltaje

El sensor implementado en el sistema para realizar la medida de tensión del panel fotovoltaico es el módulo sensor de voltaje FZ0430, el cual solo realiza medida de voltaje DC debido a su funcionamiento. Este módulo se puede ver en la ilustración 5.6.

El FZ0430, es un módulo sencillo, diseñado para trabajar con placas de desarrollo de Arduino y similares. Está compuesto por un divisor de tensión de dos resistencias, una de 7.5K y otra de 30K, de forma que la tensión a medir es dividida por un factor de 5, ajustándose la señal a valores adecuados para las entradas analógicas del microcontrolador. Por tanto, la tensión máxima que puede medir es de 25V para microcontroladores con tensión de alimentación de 5V y de 16.5 para aquellos con una alimentación de 3.3V, como es el caso del microcontrolador de este trabajo.

Ilustración 5.6 Módulo sensor de voltaje FZ0430 *Fuente: AZ-Delivery*

Como se puede apreciar en la ilustración 5.6, dispone de 3 pines, uno para la alimentación (5V o 3.3V), otro para la masa y un tercero donde obtenemos la señal de medida. Además, cuenta con dos conectores, a través de los cuales conectaremos el dispositivo a medir. La resolución de medida es 0.00489V, y el voltaje mínimo de medida es de 0.02441V.

5.5. Sensor de corriente

El sensor implementado en el sistema para realizar la medida de intensidad otorgada por el panel fotovoltaico es el módulo sensor de intensidad ACS712 [51]. Este módulo puede realizar medidas tanto de corriente alterna como continua y está diseñado específicamente para trabajar con microcontroladores Arduino o placas de desarrollo ESP32. En la ilustración 5.7 se observa el módulo citado y su pinout.

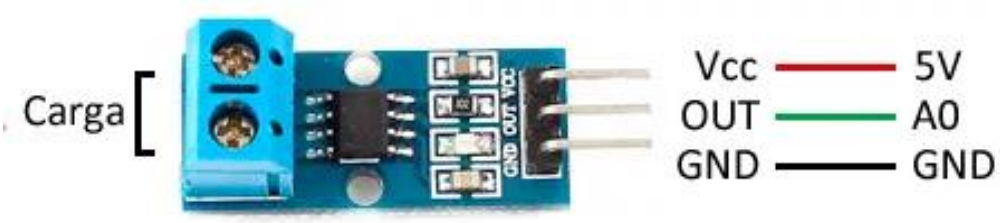

Ilustración 5.7 Pinout Módulo sensor de corriente ACS7112 *Fuente:Luis Llamas*

El módulo está compuesto por un sensor hall, de precisión y bajo la acción de offset, junto con un canal de cobre de resistencia igual a 1,2 mΩ y aislado galvánicamente en la superficie del integrado. Al fluir la corriente por este canal se genera un campo magnético detectado por el sensor hall y convertido a un valor de tensión proporcional. Este valor de tensión se conecta al pin de salida del módulo para poder ser detectado por el microcontrolador. La salida del sensor es altamente

independiente de la temperatura y gracias a su baja resistencia, las pérdidas son despreciables.

Como se puede apreciar en la ilustración 5.7, dispone de 3 pines, uno para la alimentación (5V), otro para la masa y un tercero donde obtenemos la señal de medida. Además, cuenta con dos conectores, a través de los cuales conectaremos el dispositivo a medir. La resolución de medida es 66 mV/A con un rango de medida entre 0 A y 5 A y soporta una corriente máxima cinco veces superior al rango de medida. Para poder usarse junto a placas de desarrollo ESP32, debe considerarse no sobrepasar salidas superiores a 3.3 V, por lo que el límite de intensidad medida será 3.3 A.

5.6. PC servidor

Tal y como se expone en el apartado 3.3 del trabajo, el servidor se implementa sobre un PC con sistema operativo Linux. La distribución instalada es Ubuntu 16.04 LTS de 32 específica para procesadores de 32 bits. El equipo es un HP Pavilion-dv5000-EZ16, con una RAM de 1GB y un disco de memoria de 100GB. Cuenta con un procesador Genuine Intel® CPU T2400 @ 1.83 GHz x 2.

5.7. PC de trabajo

Durante el desarrollo del sistema, se cuenta con el apoyo de un equipo “LENOVO ideapad320” con sistema operativo Windows 10 y versión 22H2 de 2022. Posee un procesador Intel® Core™ i5-8250U CPU@1.60GHz 1.80 G basado en 64 bits y con una RAM de 8GB A través de este equipo, se llevará a cabo la ejecución software de los diferentes procesos necesarios para la implementación del sistema. Por tanto, dentro de este equipo se dispone de: entorno de programación Arduino IDE, herramienta Node-RED, software Apache Friends Xamppn, conexión SSH y conexión a red WLAN.

6. Software utilizado en el estudio

Para poder realizar la comunicación de los diferentes dispositivos hardware expuestos en el apartado 5 del presente trabajo, así como programar y configurar estos dispositivos con el fin de realizar las acciones necesarias para que el sistema cumpla con los objetivos expuestos en los apartados 1.1 y 1.2, se requieren los componentes software expuestos en los siguientes subapartados.

6.1. Arduino IDE y C++

El sistema de supervisión y monitorización requiere programar el microcontrolador expuesto en el apartado 5.1 y el módulo expuesto en el apartado 5.2, con el fin de tomar la medida de los diferentes parámetros, así como realizar la captura de imágenes. Además, se debe realizar el envío de datos hacia unidades capaces de analizarlos y gestionarlos, además de recibir órdenes de control.

Uno de los lenguajes de programación más usados para microcontroladores, y el elegido para este trabajo, es C++. Es un lenguaje de alto nivel que permite programar tanto microcontroladores como aplicaciones, sin necesidad de usar lenguajes de bajo nivel, como lenguaje máquina o ensamblador. Es compilado, de forma que el programa se traduce entero a lenguaje máquina, para posteriormente ser ejecutado. Esta compilación, se ejecutará las veces que sea necesario, sin necesidad de volver a compilar. Además, C++ permite compilar un mismo programa para diferentes sistemas, pudiendo escribir un solo programa para varios sistemas [52].

Para escribir, compilar y cargar el código en el microcontrolador, hay diferentes alternativas que van desde editores de código (como Visual Code) [53], hasta la plataforma Arduino IDE (Entorno de Desarrollo Integrado) [54], que usaremos en este trabajo (Ilustración 6.1). Esta plataforma, es un entorno de programación empaquetado como un programa de aplicación, en el que se incluye un editor de código, un compilador y un constructor de interfaz gráfica formado por un monitor Serial y un Serial plotter. Cuenta con una herramienta para cargar el programa compilado en la memoria flash del hardware.

Ilustración 6.1 Plataforma Arduino IDE

Posee una gran variedad de herramientas, así como la posibilidad de insertar en sus ficheros de programa, librerías con funciones específicas. Las librerías pueden obtenerse desde la web de Arduino, así como compartirse entre terceros al tratarse de programas de código, donde solo se requiere modificar ciertas instrucciones. Cabe destacar que los programas creados, se componen de un único fichero con extensión "ino" que debe ser guardado en una carpeta con el mismo nombre que el fichero [54].

6.2. Node-RED

Tal y como se describe en la página web de Node-RED [55], "Node-RED es una herramienta de programación para conectar dispositivos hardware, API y servicios en línea de formas nuevas e interesantes". Usa un lenguaje de muy alto nivel, el cual se realiza de forma gráfica y por bloques, sin necesidad de usar lenguajes complejos como HTML o JavaScript. Está basada en Node.js, un entorno de ejecución JavaScript de código abierto, multiplataforma y ligero, que se usa para desarrollar aplicaciones ideales para ejecutar en la nube. Esto hace que existan numerosas librerías de bloques disponibles con funcionalidades que podemos desarrollar en nuestros proyectos.

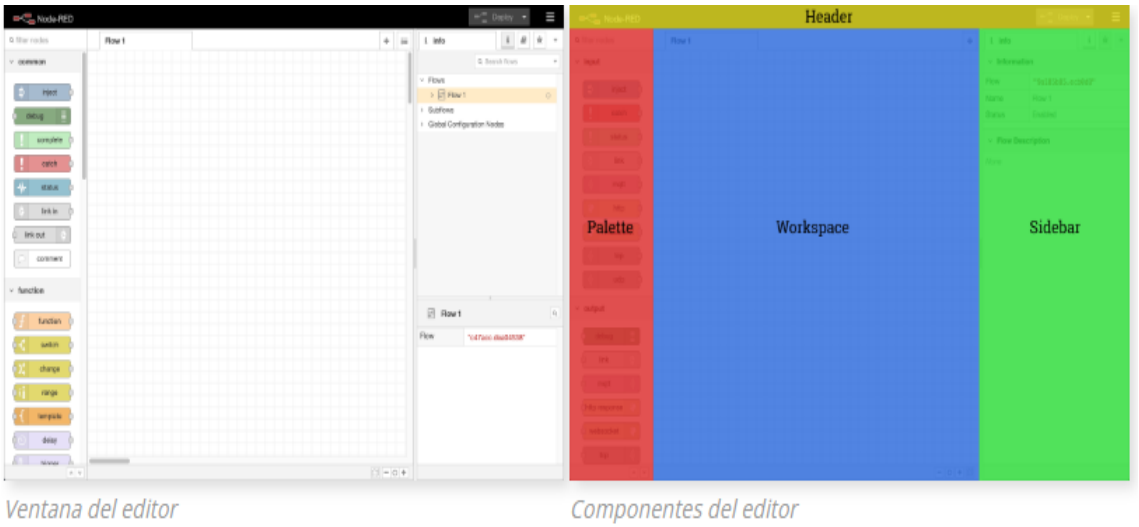

Ilustración 6.2 Editor de flujo Node-RED *Fuente: Nodered.org*

Su editor de flujo (Ilustración 6.2) está basado en el navegador, lo que permite acceder a él desde otros equipos con conexión al servidor donde se aloja la herramienta, únicamente hay que introducir la IP del servidor en un navegador. Está formado por cuatro componentes: Encabezado superior que contiene el botón de implementación y el menú principal, la paleta que contiene los nodos disponibles, el espacio de trabajo donde se agregan los nodos y se crean los flujos, y la barra derecha de configuraciones. Dentro de este editor de flujo, se eligen los nodos necesarios en el proyecto, se configuran según las especificaciones, y se lleva a cabo su conexión para que establezcan la comunicación [55]. Existen tres tipos de nodos básicos:

- Nodos Indicadores: encargados de iniciar la ejecución del flujo de proceso. Únicamente poseen un puerto de salida.
- Nodos Intermedios: reciben información y la procesan para pasarla al siguiente nodo. Poseen un puerto de entrada y varios de salida.
- Nodos Terminales: poseen solo puerto de entrada y se encargan de ejecutar otros flujos, realizar eventos de notificación o llamar a otros servicios.

6.2.1. Nodos MQTT

El editor de flujo de Node-RED cuenta con nodos específicos para realizar comunicaciones. Como se verá en apartados posteriores del trabajo, únicamente hay que arrastrar estos nodos a nuestro espacio de trabajo y realizar las configuraciones de Servidor y tópico. Se necesitará indicar el puerto de comunicación, la IP del servidor y los nombres de Topic a los que se suscribe o sobre los que publica cada cliente. [55]

6.2.2. Nodo MySQL

Dentro del editor de flujo de Node-RED se puede agregar un nodo específico para realizar lecturas e insertar datos en una base de datos MySQL. Como se verá en el apartado 7.2 del presente trabajo, requiere de una instalación dentro del editor de flujo para poder ser utilizados. Este nodo permite configurar el direccionamiento de los datos extraídos del sistema hacia una base de datos MySQL instalada sobre el PC servidor. [56]

6.2.3. Nodos Dashboard

Como se explica en el apartado 4.4 del presente trabajo, el web service Node-RED incluye su propia interfaz de visualización llamada “Dashboard” con funcionalidades similares a una pantalla HMI. Esta interfaz se configura desde el editor de flujo de Node-RED encontrándose dentro de este, diferentes nodos específicos para añadir objetos, curvas, gráficas o elementos de interacción sobre el panel de visualización. Su acceso es vía web y como se tratará en el apartado 7.2, para hacer uso estos nodos es necesario realizar una instalación dentro del editor de flujo. [57]

6.3. Eclipse Mosquitto

La definición de Eclipse Mosquitto dada en la página web de Eclipse Foundatión [58], organización a cargo de su desarrollo, es la siguiente: “Eclipse Mosquitto es un bróker de código abierto (con licencia EPL/EDL) que implementa las versiones 5.0, 3.1.1 y 3.1 del protocolo MQTT”. Es compatible con Windows, Mac y Linux, incluyéndose como herramienta en los repositorios de algunas de sus distribuciones.

Este bróker ha alcanzado mucha popularidad al ser un liviano, Open Source (código abierto) y multiplataforma, pudiendo usarse en dispositivos móviles, computadoras o microcontroladores. Proporciona un método ligero para enviar mensajes usando el modelo publicación/suscripción.

Además, cuenta con una biblioteca C para implementar clientes MQTT en diversas plataformas, así como clientes MQTT en línea de comandos con los conocidos comandos “*mosquitto_pub” y “mosquitto_sub”*, que pueden ejecutarse en los terminales de diferentes sistemas operativos.

6.4. Protocolo SSH

El protocolo SSH (Secure Shell) [59] es un protocolo de red para administración remota, destinado principalmente a la conexión de máquinas a las que se accede por la línea de comandos. Permite controlar servidores remotos a través de internet gracias al

uso de mecanismos de autenticación. Surge en 1995 como sustituto de Telnet (Red de telecomunicaciones), funcionando de forma similar, pero con protocolos de encriptación.

Una de las características más importantes de SSH es su seguridad al usar técnicas de criptografía en todas las comunicaciones hacia y desde el servidor remoto. Proporciona una conexión cifrada y una autenticación robusta compatible con la ejecución de comandos y transferencia de archivos.

Funciona a través del modelo cliente-servidor, operando por defecto en el puerto TCP 22. El servidor escucha a la espera de conexiones entrantes, siendo el cliente el que debe iniciar la conexión SSH asegurando una conexión simétrica segura. Una vez iniciada la conexión, el servidor lleva a cabo la autenticación del cliente y establece el entorno de conexión Shell.

6.5. Apache Friends Xampp

Se trata de una distribución de Apache que incluye diferentes softwares libres y servidores de datos, dando lugar a una herramienta de desarrollo web basada en PHP. Con ella se consigue sencillez a la hora de realizar la configuración y activación de servidores MySQL y servicios de gestión de estos. Los softwares incluidos son: [45]

- Servidor HTTP Apache: permite visualizar paginas alojadas en un servidor remoto, en nuestro propio servidor o en nuestro PC, a través del navegador.
- Lenguaje PHP: diseñado para realizar páginas web dinámicas. Ejecutado sobre el servidor permite crear páginas web a través de códigos PHP de entrada.
- MySQL: Es una base de datos explicada en el apartado 3.12 del presente trabajo.
- PhpMyAdmin: aplicación web para administrar bases de datos MySQL a través de una interfaz amistosa y no trabajar con datos en crudo a través de la consola de comandos.

Estas aplicaciones están pensadas para instalarse en un servidor y ser gestionadas desde un navegador, otorgando flexibilidad para ejecutar los programas en nuestra red local o en la nube.

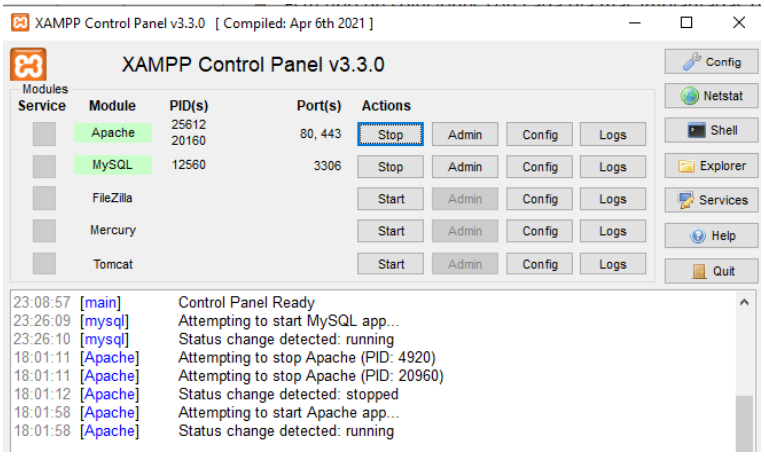

Ilustración 6.3 Panel de control de herramienta Xampp *Fuente: Propia*

En la ilustración 6.3 se observa la interfaz del control de Xampp, a través de la cual se activan y configuran los diferentes servicios incluidos dentro de la distribución.

7. Preparación de equipos

Los equipos hardware y software explicados en los apartados 5 y 6 del presente trabajo, requieren de acciones previas a su integración. En algunos casos, estas acciones son necesarias e indispensables para el correcto funcionamiento del sistema, mientras que, en otros casos, simplifican el trabajo durante la implementación del sistema.

7.1. Instalación y Configuración Arduino IDE

La instalación del Arduino IDE se realiza de forma sencilla y de forma automática desde la página oficial de Arduino [54], donde podemos elegir la versión que mejor se adecue a nuestro sistema. Esta instalación se lleva a cabo en el PC de trabajo.

7.1.1. Instalación del núcleo ESP32

Una vez instalado, para poder programar un ESP32 desde el entorno Arduino es necesario descargar e instalar el núcleo (o core) del ESP32 para Arduino. Con esto, Arduino es capaz de detectar el microcontrolador y establecer una conexión para cargar en el procesador el código instalado. Este proceso se realiza en dos pasos sencillos:

- Paso 1: Actualizar la base de datos del gestor de placas. Para ello se ejecuta Arduino IDE y se accede al apartado “Gestor de tarjetas adicionales” a través de “Archivo>Preferencias”. Dentro de este apartado, se agrega la URL específica para nuestra placa y fabricante. En este caso, la URL añadida es: *https://dl.espressif.com/dl/package_esp32_index.json*
- Paso 2: Instalar el paquete de recursos que contine la placa de desarrollo. Para ello se “accede al gestor de tarjetas” (Ilustración 7.1) a través de “Herramientas>Placas>Gestor de tarjetas”, se busca el paquete correspondiente a ESP32 y se instala.

Tras realizar estos pasos, únicamente hay que seleccionar la placa de desarrollo que se desea programar a través de “Herramientas>Placas”. En este caso hay que seleccionar “ESP32 Dev Module” para la programación del sistema de captación de variables de proceso, y “AI Thinker ESP32-CAM” para programar la captura de

imágenes.

Ilustración 7.1 Gestor de tarjetas de Arduino IDE *Fuente: Propia*

7.1.2. Librerías de programación

Previo a la programación del microcontrolador, es necesario agregar en los archivos del entorno de programación Arduino IDE tres “Librerías Arduino”. Estas librerías son indispensables para configurar y programar ciertas tareas en los microcontroladores, como la comunicación Wifi, el uso de protocolo MQTT o la configuración y captura de imagen. Además, permiten simplificar el programa a ejecutar y su desarrollo. Las librerías añadidas para el presente trabajo son:

- Wifi.h: Permite habilitar la conexión a la red (local o internet) mediante el uso del módulo Wifi del microcontrolador, pudiendo configurar el microcontrolador como servidor o cliente. Además, proporciona información sobre la conexión a través de ciertos comandos específicos. Los comandos usados en el presente estudio son: *WiFi.begin(ssid,password); WiFi.status(); WiFi.localIP().* Su función dentro del programa se detalla en el apartado 8.3 de este trabajo. [60]
- Pubsubclient.h: Permite establecer en microcontroladores y dispositivos IoT un cliente MQTT. Usa por defecto la versión MQTT 3.1.1, por lo que puede trabajar con QoS 0 y QoS 1 en la lectura de mensajes y únicamente publicar en QoS 0, con un tamaño máximo de mensaje es de 256 bytes. Los comandos usados durante el trabajo son*: PubSubClient client(espClient); client.setServer(IP,Port); client.setCallback(callback); client.connected(); client.publish(topic,payload).* Su función dentro del programa se detalla en los apartados 8.2 y 8.4 de este trabajo. [61]

- Esp_camera.h: Permite realizar la configuración del módulo ESP32 CAM, estableciendo la configuración de cada pin del módulo, inicializar la cámara, configurar la calidad y dimensiones de imagen, y realizar la captura. Los comandos usados en este trabajo son: *confi.xxx* y *camera_example_capture()* el cual se inserta a través de la biblioteca de ejemplos que contiene Arduino IDE. Su función dentro del programa se detalla en los apartados 8.3 y 8.4 del presente trabajo.

Para agregarlas en los archivos de programa, únicamente hay que realizar la descarga desde la web “Arduino Home” y añadirla en el archivo “libraries” creado durante la instalación del Arduino IDE. Dentro del programa, deben inicializarse en las primeras líneas de código y agregarse desde la pestaña Programa>Añadir Librería, donde podremos seleccionar entre todas las librerías instaladas en el IDE de Arduino.

7.2. Instalación Node-RED

La instalación se realiza dentro del PC servidor que contiene sistema operativo Linux Ubuntu y en dos procesos que se realizan a través del terminal de Símbolo del sistema. El primer proceso es instalar Node.js. Para ello se ejecuta los comandos: *sudo apt install nodejs* y *sudo apt install npm.* Una vez ejecutados, se procede al segundo proceso donde se instala Node-RED. Para ello se ejecutan los comandos: *sudo npm install -g -unsafe-perm node-red* y *sudo node-red* que activa el servicio.

Para acceder al editor de flujo, se debe escribir en el navegador la dirección sobre la que trabaja, siendo por defecto la dirección 127.0.0.1. Dentro de esta, se editarán los flujos de configuración y gestión de comunicación entre dispositivos. Para acceder a su interfaz de visualización, se debe escribir la dirección IP del equipo donde se alberga el servicio Node-RED junto con el puerto de comunicación y el comando” /ui”. En este caso la IP es 192.168.1.35, correspondiente al PC servidor, y el puerto de comunicación 1880. [55]

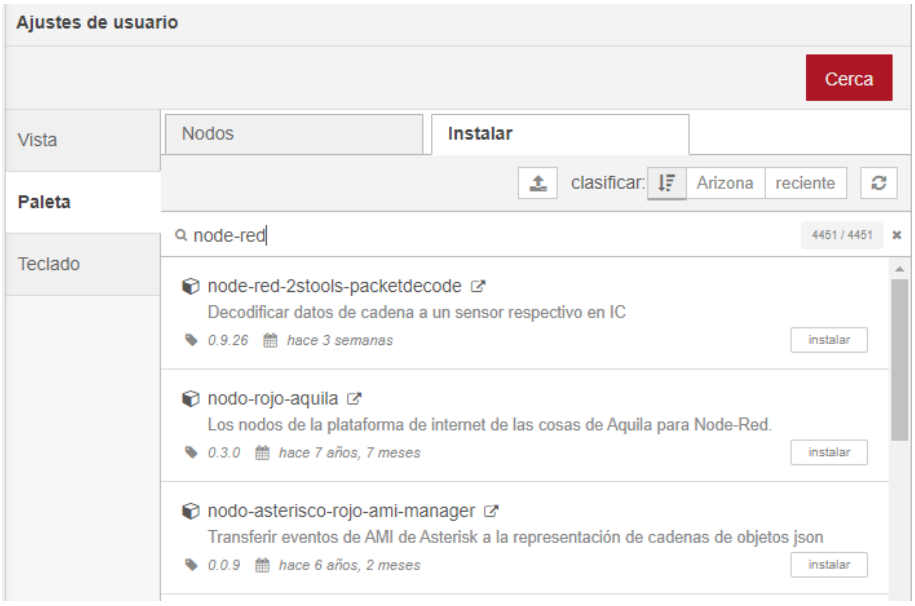

Ilustración 7.2 Administrador de paleta Node-RED *Fuente: Propia*

Por último, hay que realizar la instalación de los nodos correspondientes al protocolo MQTT, a la base de datos MySQL, a la interfaz Dashboard y al reconocimiento de imagenes. Para ello se accede al editor de "paleta" de la ilustración 7.2, se inserta en el buscador "node-red-node-mqtt", "node-red-node-mysql", "node-red-node-dashboard" y "node-red-contrib-ui-media" y se selecciona la opción a instalar. Tras esto, aparecen dentro de los nodos disponibles, los correspondientes a comunicación MQTT, servicio MySQL, visualización en Dashboard y nodos para gestión y visualización de imagenes.

7.3. Conexión SSH entre Windows y Linux Ubuntu

El objetivo de esta conexión es acceder al PC que ejerce la función de servidor desde el PC de trabajo. Con esto se consigue realizar todos los procesos de configuración y programación desde el mismo PC, facilitando el trabajo y permitiendo tener el servidor en puntos de la red WLAN a los que no tenemos fácil acceso. En los apartados 5.6 y 5.7 se explican las características de estos dispositivos, necesarias para conocer el método con el que podemos establecer esta conexión y el cual dividimos en tres pasos:

- Activar el servicio SSH en Ubuntu: Las distribuciones Linux tienen instalada por defecto la aplicación "OpenSSH" para poder trabajar con este protocolo. Sin embargo, se encuentra desactivada, por lo que debemos abrir el terminal (herramienta de línea de comandos) e introducir el comando "*sudo apt install*

openssh-server". Con esto, ya se habrá instalado y activado el servicio SSH en Ubuntu.

- Instalar el Cliente SSH en Windows 10: En versiones de Windows 10 posteriores a 2017, se encuentra como característica opcional del sistema operativo la herramienta OpenSSH. Para instalarla, basta con acceder al administrador de funciones opcionales del panel de configuración, buscar la opción "Cliente OpenSSH" y seleccionar la opción de instalar.
- Establecer conexión: para conectar el PC de trabajo al servidor, únicamente hay que abrir el terminal del ordenador desde donde controlar el servidor e introducir el comando con los datos del equipo servidor: "*ssh nombre_del_equipo@ip_del_equipo".*

Cabe destacar que, para realizar esta comunicación, ambos equipos han de estar dentro de la misma red WLAN, es decir, necesitan tener IPs del mismo tipo. Para conocer las IPs de los equipos, únicamente hay que abrir los terminales y escribir los comandos "ipconfig" para Windows y "ifconfig" para Ubuntu. Por defecto se realiza la conexión a través del puerto 22.

Ilustración 7.3 Terminal PC de trabajo con conexión SSH hacia PC servidor Fuente: Propia

En la ilustración 7.3 se visualiza el terminal de comandos del PC de trabajo durante la configuración del PC servidor a través de la conexión SSH.

7.4. Instalación Broker Mosquitto

La mayoría de las distribuciones Linux tienen integrado en sus repositorios el Broker Mosquitto. Una de estas distribuciones es Ubuntu, la cual opera en nuestro PC

servidor, por lo que para instalarlo únicamente debemos ejecutar en el terminal los comandos: "*sudo apt update", "sudo apt upgrade" y "sudo apt-get install mosquitto mosquitto-clients"*.

Además, para este trabajo necesitamos que mosquitto se ejecute por defecto al arrancar el sistema. De este modo, en caso de reinicio del servidor se establecer de forma automática la comunicación MQTT entre los dispositivos sin necesidad de acciones por parte del usuario. Para ello, se ejecuta en el terminal el comando "*sudo systemctl enable mosquitto.service".*

7.5. Instalación y configuración Xampp

La instalación del software se realiza sobre el PC servidor de forma sencilla a través de la página oficial de Apache Friends. Durante la instalación se añadirán automáticamente todos los softwares descritos en el apartado 6.5 del trabajo y se creara un usuario llamado "root" (por defecto).

Una vez instalado y dentro de la interfaz Xampp, debemos activar el servicio Apache y MySQL. Con esto se activa la base de datos MySQL que incorpora y, además, se establece la visualización web con el servidor HTTP. Tras la activación, se escribe en el navegador "localhost" o la dirección del equipo donde está instalado para acceder a la interfaz del administrador "phpmyadmin" (Ilustración 7.4). Por defecto trabaja sobre el puerto 3306. Para dar seguridad a la base de datos, accedemos "cuentas de usuario" dentro del administrador, y establecemos en el usuario "root" una contraseña segura. Esta misma contraseña debe ser cambiada en el archivo "config.inc.php" que se encuentra dentro de los archivos de programa de Xampp.

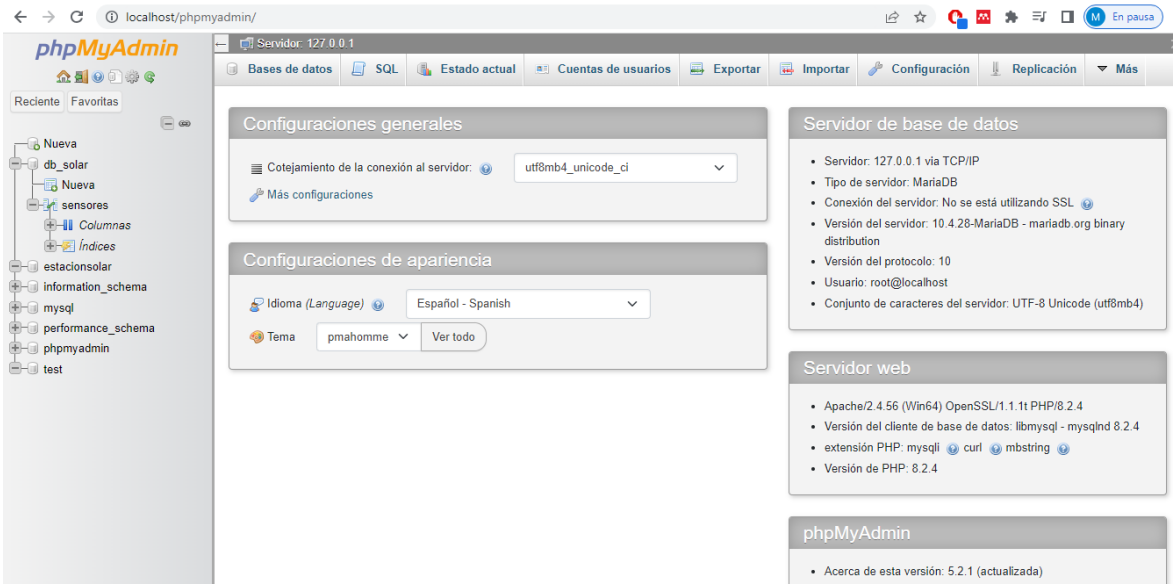

Ilustración 7.4. Interfaz phpmyadmin *Fuente: Propia*

Los archivos con la información que se almacenan en la base de datos MySQL se guardan dentro de los archivos de programa de Xampp. Para ello, durante la instalación se crea una carpeta llamada “data” dentro de la carpeta “mysql”, donde se almacenan dentro de la carpeta de la base de datos implementada en el sistema.

7.6. Conexiones eléctricas entre dispositivos

Para poder implementar el sistema solución, se deben llevar a cabo las conexiones eléctricas entre los dispositivos electrónicos que forman el sistema de captación de variables del sistema. Estas conexiones son las correspondientes a la alimentación de los dispositivos y a la entrada de señales analógicas en el microcontrolador. Además, se debe realizar, según lo descrito en el apartado 5.2.2, la conexión del adaptador USB a TTL y el módulo ESP32 CAM.

La alimentación del módulo ESP32 CAM se realiza a través de sus pines de VIN y GND, los cuales se conectan al adaptador anteriormente citado, para obtener el valor de tensión 5V necesario. El microcontrolador se alimenta también con un nivel de tensión 5V a través de su conector USB. Para la alimentación de los sensores, se usa la fuente de alimentación de 3.3V que se dispone como salida en el módulo del microcontrolador. Cabe destacar la limitación de corriente durante el uso de esta fuente, ya que no debe superar los 500mA. Las señales analógicas de cada sensor se conectan a los pines de lectura analógica del microcontrolador. Además, la entrada del sensor de voltaje se debe conectar en paralelo a las señales de salida del sistema fotovoltaico y la del sensor de intensidad en serie.

8. Diseño de software del microcontrolador

El sistema de supervisión y monitorización del funcionamiento de paneles fotovoltaicos, en el modo en que se ha planteado en apartados anteriores, requiere programar el microcontrolador expuesto en el apartado 5.1 a través del IDE Arduino y usando el lenguaje C++., explicados en los apartados 6.1.

Para que el sistema cumpla con los objetivos propuestos, el programa debe contener funcionalidades básicas que se exponen en los siguientes apartados.

8.1. Lectura de señales analógicas de los sensores

Según lo expuesto en las características de los sensores instalados en el sistema (apartados 5.3, 5.4 y 5.5), obtenemos como entrada al microcontrolador cuatro señales analógicas con un rango de variación entre 0 V y 3.3 V en función del valor medido. Estas señales son adaptadas a un valor digital para poder ser procesadas por el microcontrolador (apartado 5.1.1) entregando un valor entero entre 0 y 4095. Por tanto, es necesario crear cuatro variables tipo “int” que almacenen la lectura de cada sensor. Para realizar la lectura se usa la función *analogRead(“pin”),* donde únicamente hay que especificar en ella el pin que se desea leer.

Para adaptar el valor obtenido en la lectura a valores reales de cada variable, se usa la función *map()* la cual, igualada a variables tipo float, almacena el valor real de las variables con decimales. Dentro de esta función se especifica la variable donde se almacena la lectura, el rango de la variable int y el rango de la variable real. Cada variable tiene unos rangos de variable real al corresponderse con los rangos de medida del sensor.

8.2. Tratamiento de datos y muestreo

Para poder enviar los datos a través de una red WiFi y con protocolo de comunicación MQTT, es necesario que los datos enviados sean del tipo char (carácter). Por ello, dentro del programa se convierten las variables tipo float, donde almacenamos el valor real de las lecturas, en arrays tipo char. La función que lleva a cabo la conversión es *dtostrf(),* en la cual se especifica la variable float a convertir y el char array donde se almacena la conversión.

Dentro del microcontrolador se realiza el cálculo de parámetros de funcionamiento de los paneles, como es el caso de la potencia. Para su cálculo, se multiplica el valor de tensión y corriente de cada lectura y se guardan los resultados un array tipo float. Una vez el array alcanza una extensión de 10 valores, se calcula la

media de estos y así conocer cada cierto número de muestras, el valor promedio de esta. Además, se implementan sentencias “if” en el programa con el fin de realizar un primer análisis de los datos extraídos, estableciendo alertas sobre las condiciones de trabajo o errores de funcionamiento del sistema fotovoltaico, e indicaciones sobre el estado de funcionamiento del sistema. Dentro de estas sentencias se establece el valor de cuatro variables denominadas: *altatemperatura, bajatemperatura* y *circuitoabierto* y *estado.* Las tres primeras son tipo bool y obtienen un valor alto cuando se superan los límites de temperatura de trabajo del sistema o cuando los valores de generación presentan anomalías, como es el caso de la tercera variable. La variable denominada *estado* es de tipo int y adquiere un valor 2 cuando los valores de producción son bajos, un 1 cuando el rendimiento es intermedio y un 1 cuando la generación es óptima.

8.3. Conexión a la red WLAN

La conexión del equipo con la red LAN se realiza de forma inalámbrica mediante WiFi, gracias al módulo de comunicación WiFi integrado en el microcontrolador. Para realizar la conexión y configuración de este dispositivo, se utiliza la librería Wifi.h [60] específica para placas ESP32. Gracias a esta librería se realiza la conexión a la red a través de la función *WiFi.begin(),* en la que se indica el nombre y la contraseña de la red. Además, nos permite conocer el estado de la conexión a través del uso de la función *WiFi.status()*, así como la IP que se adjudica al equipo, a través de la función *WiFi.localIP().*

8.4. Comunicación MQTT

Una vez conectado a red, el dispositivo debe conectarse al servidor MQTT. Para esta conexión se usa la librería PubSubClient.h [61], mediante la cual es necesario especificar la dirección IP del servidor, el puerto de comunicación y las credenciales necesarias para establecer la conexión a través de la función *client.setServer().* El microcontrolador se establece como cliente en esta conexión, pudiendo realizar suscripciones para recibir datos o publicaciones para enviarlos. Dado el diseño del sistema, únicamente se requiere el envío de datos desde este, hacia el servidor MQTT. Por tanto, únicamente usaremos la función *client.publish()* en la cual hay que insertar el topic bajo el que se publica la información y la variable char que contiene los datos. Para este sistema, se crean siete topics: “esp32/voltaje” bajo el que se publican datos de tensión, “esp32/temperatura” bajo el que se publican datos de temperatura, “esp32/potencia” bajo los que se publican datos de potencia, “esp32/corriente” bajo el que se publican datos de corriente, “esp32/altatemperatura” bajo el que se publican

alertas por exceso de temperatura, “esp32/bajatemperatura” bajo el que se publican alertas por descenso brusco de temperatura, “esp32/circuitoabierto” bajo el que se publican alertas sobre errores de conexión del circuito y “esp32/estado” bajo el que se publican datos sobre el estado de la instalación.

En caso de pérdida de conexión con el servidor durante el funcionamiento del sistema, se introduce una sentencia que llame a la función *reconnect()*. Para saber el estado de la conexión, se usa la función *client.connected().*

9. Diseño de software del Módulo ESP32 CAM

Además de requerir la programación del microcontrolador expuesto en el apartado anterior, el sistema requiere programar el módulo ESP32 CAM expuesto en el apartado 5.3 y haciendo uso del lenguaje C++ en el entorno IDE Arduino explicado en el apartado 6.1. Para cumplir con los objetivos propuestos, el programa debe contener determinadas funcionalidades que se exponen a continuación.

9.1. . Conexión a la red WLAN

La conexión del equipo con la red LAN se realiza de forma inalámbrica a través del módulo de comunicación Wifi que incorpora. Dadas las semejanzas entre el microcontrolador usado en el apartado anterior y el módulo tratado en este apartado, la configuración y conexión de este dispositivo se implementa a través del uso de las mismas funciones e implementando el mismo bloque de programa que se trata en el apartado 8.3. Únicamente se diferencian en la dirección IP otorgada, siendo diferente para cada dispositivo.

9.2. Comunicación MQTT

Una vez conectado a red, el dispositivo se conecta al servidor MQTT. Al igual que en el caso del microcontrolador, se hace uso de la librería PubSubClient.h [61] y la conexión al servidor se realiza configurando el comando *client.setServer()* de la misma manera. De este modo, el módulo se establece en la red como cliente MQTT pudiendo realizar suscripciones para recibir o publicar datos.

Para la publicación de datos desde el dispositivo hacia el servidor MQTT, se establece una “ventana” de envío. Esto se debe al elevado número de datos necesarios para enviar una imagen. Para iniciar el envío usa la función *client.beginPublish()* en la que se configura en topico “esp32/cam/foto” y una variable que indica el tamaño del mensaje. Para enviar los datos se usa la función *client.write()* en la que se indica en tamaño y el array que contiene la imagen y sirve a modo de buffer. Para finalizar el envío, se inserta la función *client.endPublish().*

Como se ha descrito en apartados anteriores, la toma de imagen se realiza tras la recepción de una orden procedente de la interfaz de supervisión. Esta orden se recibe en un mensaje MQTT desde el servidor. Por tanto, se establece la función *client.subscribe()* con la que se suscribe el dispositivo al topico “esp32/cam/fotografiar”. Cada vez que reciba un mensaje asociado a ese tópico, se inicia la función *callback()*

que guarda el mensaje en una variable string y ejecuta una sentencia if para comprobar si el valor de mensaje.

En caso de pérdida de conexión con el servidor durante el funcionamiento del sistema, se introduce una sentencia que llame a la función *reconnect()*. Para saber el estado de la conexión, se usa la función *client.connected().*

9.3. Captura de imagen

Para poder implantar esta funcionalidad dentro del código de ejecución del módulo, se utiliza la librería esp_camera.h. Tras agregarla a nuestro programa, nos permitirá introducir funciones específicas para la captura, almacenamiento y procesador de imágenes. Tras analizarse el valor del mensaje en la sentencia if del apartado anterior, en caso de recibir un valor correcto se ejecuta la función *camera_capture()*. Esta función realiza la inicialización de los comandos ejecutados dentro de las funciones *camera_fb_t*fb=esp_camera_fbget(),* así como la medida de errores con la función *esp_err_t camera_capture().* Esta imagen se almacena en un array que se envía del modo expuesto en el apartado 9.2.

9.4. Configuración de imagen

Para realizar la configuración de la imagen se dispone de comandos específicos dentro de la librería esp_camera.h. Además, se agrega al programa el ejemplo ESP32-camera, contenido en los archivos de programa del IDE Arduino. Dentro de este ejemplo, se añade la definición de cada uno de los pines del módulo, necesarios para trabajar la captura de imagen, y las funciones: *camera_config_t config y esp_camera_init(&config).* Estas funciones sirven para inicializar y encender la cámara agregada al módulo.

Dentro de esta configuración, también es necesario establecer la configuración de las dimensiones y calidad de la imagen capturada. Para ello se usan los comando: *config.frame_size, config.jpeg_quality* y *config.fb_count.* Estos comandos también se añaden a través del ejemplo anteriormente citado.

10. Diseño base de datos MySQL en phpmyadmin

Para establecer una base de datos MySQL dentro de phpmyadmin, se deben realizar una serie de configuraciones para poder recibir y almacenar los datos.

El primer paso es crear una nueva base de datos, a la que se deberá otorgar un nombre específico. Al tratarse de un tipo relacional de base de datos, se debe crear una tabla, dentro de la base creada, donde insertar los datos. Esto se realiza de forma sencilla a través de la interfaz web del administrador, donde únicamente hay que seleccionar el tipo de dato que se almacena en cada columna y el nombre de la variable que se almacena en cada columna.

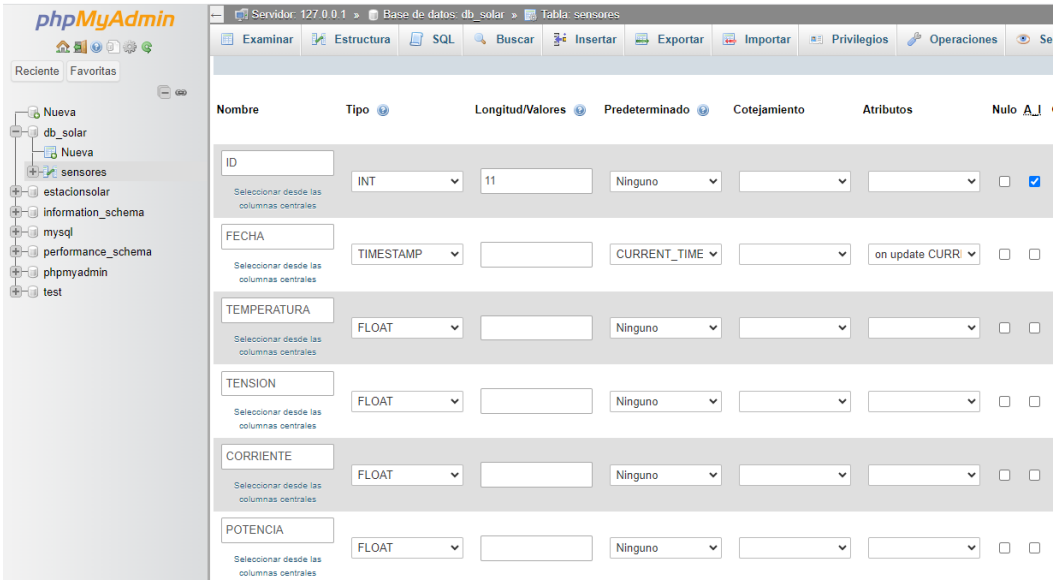

Ilustración 10.1. Creación de DB relacional en phpmyadmin *Fuente: Propia*

Con el fin de otorgar funcionalidad y orden a los valores almacenados, se permite configurar columnas de indexación y columnas con registros de tiempo, las cuales se agregan a la tabla de forma automática.

En la ilustración 10.1 se observa el panel de configuración de tabla agregada dentro de la base de datos implantada en el presente trabajo. Dentro de esta tabla, se registran los valores de tensión, corriente, temperatura y potencia del sistema fotovoltaico. Además, se añade indexación y registros de tiempo. El direccionamiento de cada valor a su celda correspondiente se realiza a través de los flujos Node-RED contenidos en el PC servidor.

11. Diseño del Web Service Node-RED

La capacidad de supervisar el sistema fotovoltaico desde la nube, así como realizar el direccionamiento de datos extraídos del sistema hacia la base de datos MySQL, es la característica principal que enmarca este trabajo en el área de la Industria 4.0. Como se explica en apartados anteriores, esto es posible gracias a la implementación en el sistema del web service Node-RED encargado de procesar los flujos de información y de ofrecer la interfaz de visualización a través del navegador web.

Para que el sistema cumpla con los requisitos establecidos durante el diseño general del sistema, se deben implementar en su editor de flujo determinadas funciones y procesos que se exponen en los siguientes apartados.

11.1. Comunicación MQTT

Durante la instalación descrita en el apartado 7.2, se agregan en la paleta del editor de flujo, los nodos “mqtt in” y “mqtt out” para realizar comunicaciones MQTT bajo protocolo TCP/IP. Para recibir los parámetros de funcionamiento del sistema fotovoltaico, procedentes del microcontrolador, y los datos de imagen enviados a través del módulo de cámara, se usa el primer tipo de nodo MQTT.

Sobre el espacio de trabajo se agregan tantos nodos “mqtt in” como tópicos se implementan en el software del microcontrolador y en el módulo. Para configurar estos nodos, hay que determinar el puerto de comunicación del bróker MQTT y la dirección IP del equipo que lo contiene (Ilustración 11.1). Además, se indica el tópico al que se suscribe cada nodo. De este modo, una vez conectados al servidor, reciben como entrada los datos asociados a su tópico y los direccionan de salida en el flujo sobre el que se han implementado.

El nodo “mqtt out” se agrega sobre el espacio de trabajo para enviar la orden de captura de imagen. Esta orden se envía bajo un tópico establecido dentro de este módulo, y con destino el bróker MQTT. Por tanto, se debe añadir el puerto de comunicación del bóker MQTT y la dirección IP del PC servidor que lo contiene. El dato enviado, será el que reciba en su entrada y procedente del flujo al que se conecta.

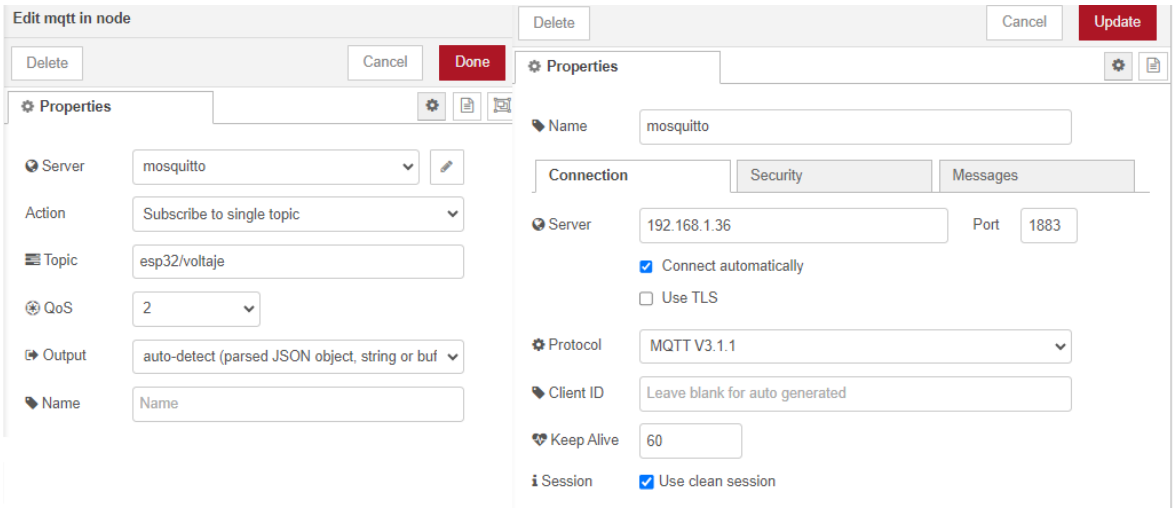

Ilustración 11.1 Configuración nodo MQTT y Servidor *Fuente: Propia*

11.2. Inserción de datos en MySQL

Esta funcionalidad se consigue gracias a la instalación, explicada en el apartado 7.2, del nodo “mysql” [62] en la paleta del editor de flujo. Una vez insertado en el espacio de trabajo, permite realizar inserciones o eliminaciones en la base de datos MySQL. Para configurar este nodo, hay que determinar la dirección IP del equipo donde se implanta MySQL, el puerto de comunicación que usa la base de datos, el nombre de la base de datos creada en el apartado 9, y el usuario y contraseña configurados durante la configuración de phpmyadmin del apartado 7.5.

Previo a la entrada de los datos al nodo “mysql”, debe especificarse a través de un nodo “función”, la celda sobre la que se escribe el valor. Para ello se escribe dentro de dicho nodo el código JavaScript de la ilustración 11.2, donde se especifica dentro del msg.topic la acción a realizar, en este caso una inserción, y el direccionamiento de cada variable a la columna deseada. Además, para realizar la inserción de más de una variable dentro de la base de datos, deben agruparse los valores de estas en un mismo mensaje. Para ello se usa un nodo “join” [63] que inserta todos los datos en un mismo mensaje tipo string. Al estar comprimidos todos los valores en un solo mensaje, dentro del nodo “función” debe segregarse cada dato de forma independiente. Para ello dentro del código anteriormente citado e ilustrado en 11.2 se añade el comando “msg.payload.split(“,”)” [64]que convierte el mensaje string en un array y así poder almacenar cada valor en una variable independiente.

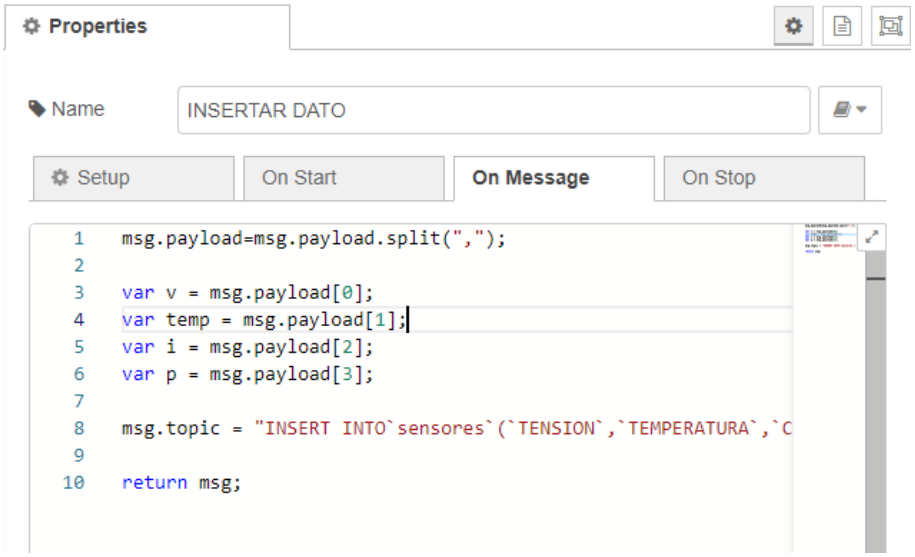

Ilustración 11.2 Función inserción de datos en DB MySQL *Fuente: Propia*

11.3. Interfaz de visualización Dashboard

La Interfaz de visualización se diseña e implementa a través de los nodos agregados a la paleta del editor de flujo durante el proceso de instalación de la librería para Dashboard [57] del apartado 7.2. Estos nodos permiten insertar botones, gráficas y curvas, cuadros de texto o establecer formularios de entrada. También se usan para la implementación de la interfaz de visualización, nodos correspondientes a la librería de reconocimiento de imágenes, como es el caso del nodo “template” [65] cuya función es mostrar imágenes en Dashboard.

Haciendo uso de la librería Dashboard [57] se modifica la distribución y forma de los elementos que componen la interfaz gráfica. De esta manera se establecen 3 tableros o pantallas independientes, entre los que se dividen por funcionalidades los objetos añadidos. Para seleccionar el tablero que se desea visualizar, se dispone de un menú desplegable donde realizar la selección.

El primer tablero, se corresponde con un menú de inicio donde se visualiza el nombre del sistema que se supervisa y el estado de este. Para ello se agregan en el espacio de trabajo Node-RED nodos “text”. Estos muestran en la Dashboard el mensaje que reciben en su entrada. Además, este tablero sirve de interfaz humano-máquina al disponer de un botón que, tras ser activado, agrega una foto actual de los paneles fotovoltaicos que componen el sistema de producción. Para esta funcionalidad se agrega un nodo “button” que envía el mensaje “tomar foto” hacia el flujo al que se conecta. Para mostrar la imagen se agrega en la Dashboard mediante el nodo “template” que recibe está en formato base64.

El segundo tablero se dedica a la visualización de las variables de temperatura, voltaje, corriente y potencia a tiempo real. Para ello se dispone del nodo “guage” en cuya configuración se indica el tablero sobre el que se visualiza, el rango de medida y la unidad de medida. Los nodos gauge, se conectan a la salida de los nodos “mqtt in” correspondientes para obtener los valores recogidos en el microcontrolador.

El tercer tablero se dedica a la visualización de las curvas y gráficas que muestran el avance temporal sobre el estado de cada variable. Para ello se implementa el nodo “chart”. Este nodo visualiza en el eje X el tiempo transcurrido y en sobre el eje Y los valores recibidos a su entrada. Por tanto, a su entrada se conecta la salida de los nodos “mqtt in” correspondientes a las variables de temperatura, voltaje, corriente y potencia. Además, se puede seleccionar entre varios tipos de gráficos y modificar el tamaño y rangos de visualización de cada eje.

Para la visualización de alarmas se usa el nodo “notify”. Este nodo muestra una pantalla emergente que contiene el texto que recibe como mensaje en su entrada. Esta ventana emergente se podrá visualizar dentro de cualquiera de los tres tableros que componen el sistema de supervisión y monitorización.

11.4. Configuración interna de mensajes

Dentro del espacio de trabajo es necesario establecer nodos que adapten o modifiquen los mensajes transmitidos en los flujos. Para adaptar los mensajes de alerta y estado, se agregan cuatro nodos “function” que realizan la comparación de los valores que reciben a su entrada a través de sentencias “if” contenidas en su configuración (ilustración 11.3). En función del resultado de la comparación, otorgaran mensajes de salida diferentes.

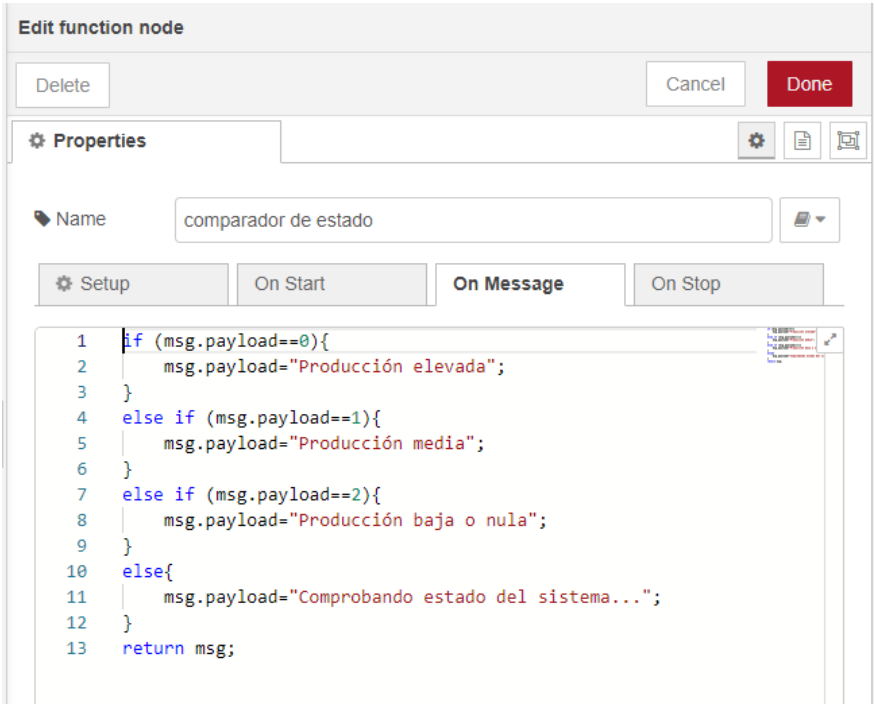

Ilustración 11.3 Sentencia if de comparación en nodo función *Fuente: Propia*

Además, para poder realizar la visualización de imagen es necesario convertir los datos recibidos del módulo ESP32 CAM contenidos en un array, en datos en formato base64. Para ello se utiliza el nodo “base64” [66].

12. Resultados obtenidos

Se han implementado los diseños expuestos según las características de los componentes descritos y cumpliendo con los objetivos propuestos. Dentro de los apartados siguientes, se da visibilidad a los resultados obtenidos en cada uno de los diseños y al sistema propuesto como solución.

12.1. Conexión de componentes electrónicos

Se han conectado los diferentes dispositivos electrónicos siguiendo las indicaciones descritas en el apartado 7.6 del estudio. Para facilitar la distribución de las conexiones se usa una PCB. Las conexiones realizadas se pueden observar en la ilustración 12.1.

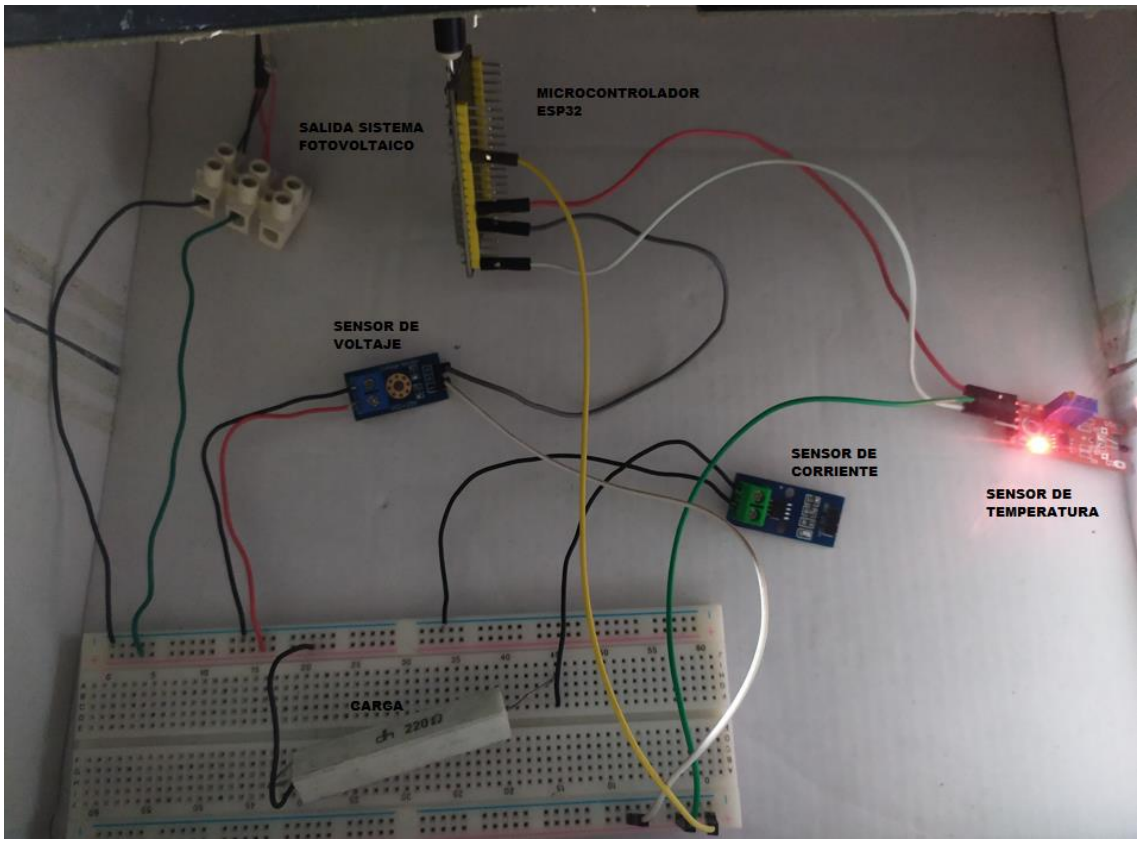

Ilustración 12.1 Cableado dispositivos electrónicos *Fuente: Propia*

Durante las pruebas realizadas sobre el sistema, se sustituye la resistencia del circuito a modo de carga, por la batería de un teléfono móvil.

12.2. Programación del microcontrolador y módulo ESP32 CAM

Se ha programado el microcontrolador y el módulo ESP32 CAM de acuerdo con lo expuesto en los apartados “Diseño de software del microcontrolador” y “Diseño de

software del módulo ESP32 CAM", obteniendo como resultado los códigos C++ contenidos de forma correspondiente en los Anexos 1 y 2.

12.3. Programación del web service

Siguiendo con lo establecido en el apartado "Diseño del web service Node-RED", se ha programado en el editor de Node-RED los flujos representados en la ilustración 12.2.

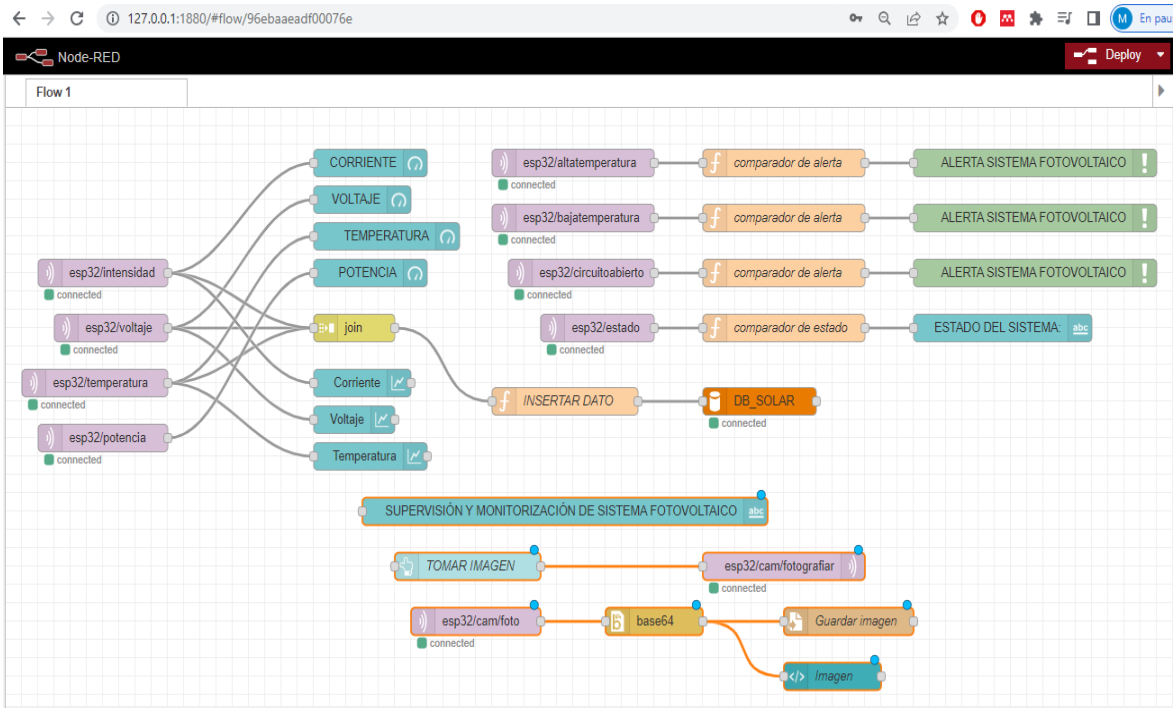

Ilustración 12.2 Diagramas de flujo del web service Node-RED *Fuente: Propia*

El flujo representado en la parte superior derecha se corresponde con la recepción vía MQTT y direccionamiento de las variables de funcionamiento del sistema, hacia los nodos de visualización de los tableros "sensores" y "curvas" de la Dashboard. Además, se direccionan conjuntamente al nodo "join" explicado en el apartado 11.3, a la vez que la salida de este se conecta al nodo "mysql" creado. Con este flujo se consigue la visualización en tiempo real del estado de las variables, así como la representación gráfica de su evolución.

El flujo representado en la parte superior derecha se corresponde con la recepción vía MQTT y direccionamiento de las variables de alerta y estado del sistema, hacia los nodos "function", que realizan los procesos de comparación, explicados en el apartado 11.4, a la vez que sus salidas se conectan a los nodos "notify" y "text". Con este flujo se consigue la creación de alertas y visualización del estado de producción del sistema en el tablero "inicio".

El flujo inferior, se corresponde con el envío de la orden de captura de imagen y recepción y visualización de esta. Para ello se conectan los nodos “button” y “mqtt out” que envía la orden al presionar el botón en la Dashboard del tablero “inicio”. Además, se reciben los datos de imagen a través del nodo “mqtt in” y se direccionan al nodo “base64”, explicado en el apartado 11.4, para enviar los datos hacia el nodo “template” explicado en el apartado 11.3 y al nodo “write filw” para guardar las imágenes en un archivo de memoria del PC servidor. Además, se dispone de un nodo “text” libre para agregar el título del sistema en el tablero de inicio. Con este flujo se configura el tablero “inicio” y las notificaciones emergentes del sistema.

12.4. Interfaz de supervisión y monitorización

La interfaz de supervisión se divide en tres tableros o pantallas con distintas funcionalidades explicadas en el apartado 11.3. Además, se dispone un menú para seleccionar la pantalla a visualizar.

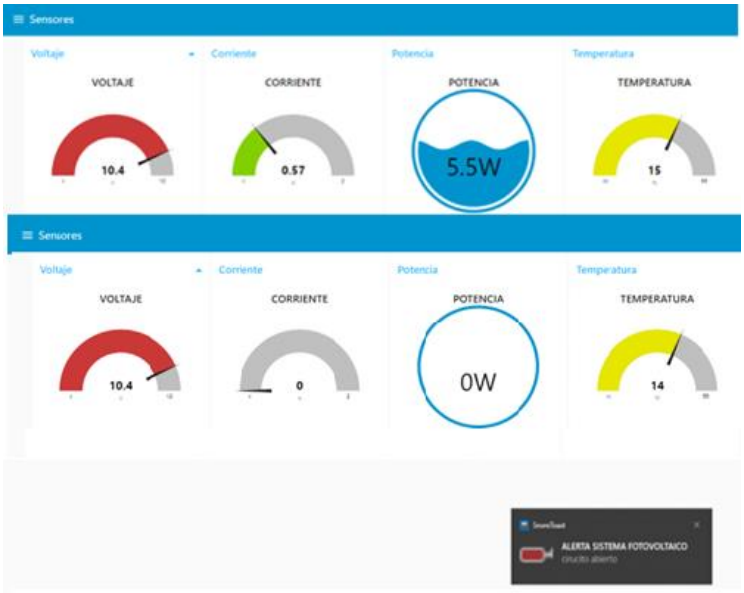

Ilustración11.3 Pantalla sensores en Dashboard *Fuente: Propia*

En la ilustración 11.3 se puede apreciar la pantalla de “sensores” donde se visualiza en tiempo real el valor de cada variable del sistema. Además, se observa una anomalía en el sistema, dando valores óptimos de tensión y una corriente nula. Por tanto, surge una ventana emergente con la alerta correspondiente.

Ilustración 11.4 Pantalla inicio en Dashboard *Fuente: Propia*

En la ilustración 11.4 se observa la pantalla de incio que contiene la información sobre el estado de producción del sistema, un boton para realizar la orden de captura de imagen y la imagen tomada. Entre la selección del botón y la recepción de la imagen en la pantalla, hay un retraso de unos pocos segundos. Esto se debe al peso de datos enviado.

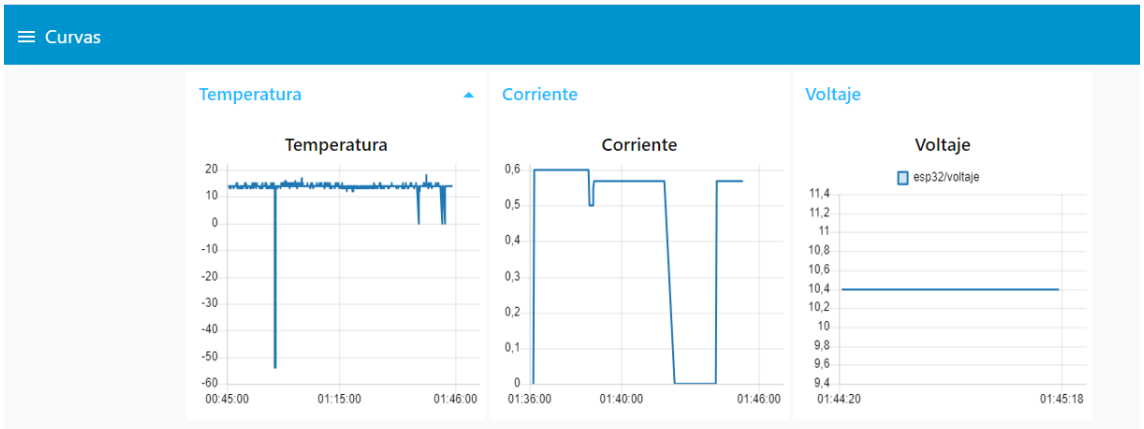

Ilustración 11.5 Pantalla curvas en Dashboard *Fuente: Propia*

En la ilustración 11.5 se observa la pantalla de curvas. En esta, se muestran el avance temporal de los valores de cada parámetro medido del sistema. Cabe destacar, que los datos mostrados en las gráficas de la ilustración han sido escogidos manualmente, para poder observar los diferentes tipos de gráfica y resoluciones que se puede mostrar.

12.5. Almacenamiento de datos en MySQL

Siguiendo lo establecido en el apartado 10 del presente trabajo, se ha creado una base de datos para el sistema de supervisión y monitorización, llamado “DB_solar”, a través del administrador phpmyadmin. Además, se ha creado una tabla con indexación y registro de tiempo automático. Dentro de esta tabla se almacenan los datos de temperatura, voltaje, corriente y potencia, tal y como se observa en la ilustración 11.6.

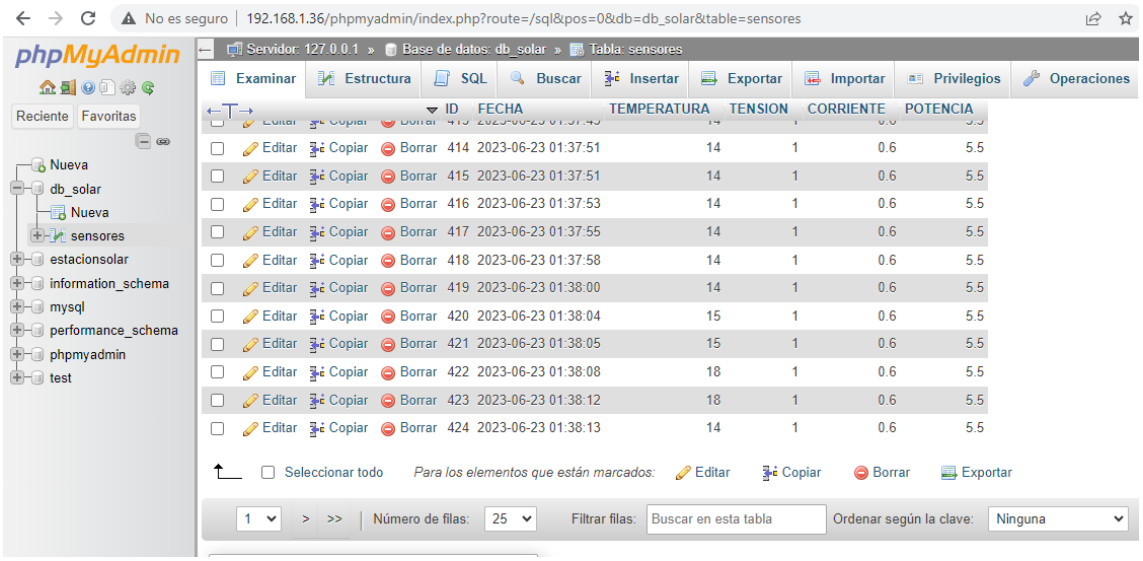

Ilustración 11.6 Datos registrados en Base de datos MySQL *Fuente: Propia*

A través del administrador también se realiza la monitorización de los datos extraídos, pudiendo acceder a él a través de cualquier equipo de la red WLAN establecida. Además, permite realizar operaciones de gestión, como eliminar datos innecesarios o erróneos.

13. Conclusiones

La industria 4.0 aboga por la implantación de sistemas altamente automatizados con los que poder lograr el máximo rendimiento de las instalaciones y optimizar los recursos. Además, una de las bases fundamentales de la Industria 4.0 es la integración de los sistemas, de manera que se consigue una industria conectada gracias al uso de tecnologías de comunicación.

Las técnicas de supervisión y monitorización de sistemas de producción han sido objeto de estudio y desarrollo desde el comienzo de la Industria 4.0, buscando realizar el seguimiento y control de los parámetros más representativos de los sistemas. La aplicación de estas técnicas dentro de la industria fotovoltaica, permiten conocer el estado y rendimiento de los equipos con el fin de tomar decisiones o estrategias que determinen una mayor eficiencia. Además, gracias a la monitorización se pueden detectar de forma rápida cualquier problemática en los dispositivos, pudiendo evitar deficiencias energéticas de forma prolongada o daños sobre otros equipos.

En este Trabajo de Fin de Grado se implementa un sistema de supervisión y monitorización sobre el funcionamiento de paneles fotovoltaico, cumpliendo con los objetivos plantados. Se desarrolló un sistema de captación de parámetros del proceso. Se logró la comunicación MQTT con el fin de conectar el sistema, a través de la nube, con el resto de los dispositivos. Se desarrollo una aplicación en la nube que permite la monitorización y pequeñas acciones de control sobre la toma de imágenes. Se implantó una base de datos donde recoger y guardar los datos extraídos.

13.1. Hitos futuros

Pese a que la solución desarrollada es completamente funcional, existen diferentes acciones que amplíen su funcionalidad y logren ventajas competitivas del sistema dentro de la industria actual.

Algunas de estas mejoras son:

13.1.1. Algoritmo de predicción de producciones

Conocer a medio y corto plazo los niveles de producción del sistema fotovoltaico a través del análisis de partes meteorológicos. Esto supone ventajas competitivas y económicas, además de establecer mejor organización en los trabajos a realizar sobre el sistema fotovoltaico.

13.1.2. Implementar acceso al servidor desde fuera de la red local

Una mejora potencial del sistema es habilitar la comunicación del sistema de supervisión con dispositivos que no se encuentren conectados a la red WLAN establecida.

Bibliografía

[1] Francisco JARABO FRIEDRICH Celestino PEREZ DOMINGUEZ Nicolás ELORTEGUI ESCARTIN José FERNANDEZ GONZALEZ José Juan MACIAS HERNANDEZ, *El libro de las energías renovables*.

[2] R. C. G. E. I. D. T. Y. F. S. A. U. Javier María Méndez Muñiz, "ENERGÍA SOLAR FOTOVOLTAICA".

[3] Roque Moreno Fonseret, "LA ESCASEZ DE ENERGÍA ELÉCTRICA EN LA POSTGUERRA (1943-50)," *Universidad de Alicante*. Accessed: May 08, 2023. [Online]. Available: https://rua.ua.es/dspace/bitstream/10045/54614/1/Anales-Historia-Contemporanea_06_07.pdf

[4] Rocío Sánchez Lissen y María Teresa Sanz Díaz, "El Plan de Estabilización español de 1959: Juan Sardá Dexeus y la economía social de mercado," *EconomicHistoryResearch*, pp. 1–10, Dec. 2013.

[5] Dra. Dña. M. S. G. de las H. H. D. José María Lorca Alcalá, "El impacto económico de la crisis del petróleo en los últimos años del franquismo (1973-1979).," Universidad de Zaragoza, Zaragoza, 2015.

[6] M. G. L. E. D. O. M. SEVILLA JIMÉNEZ, "Las energías renovables en España," Valladolid, Jan. 2013.

[7] Jefatura del Estado, "BOE-A-2005-1967," Feb. 2005. Accessed: May 09, 2023. [Online]. Available: https://www.boe.es/eli/es/ai/1997/12/11/(1)/con

[8] I. para la D. y A. de E. IDAE, *PLAN DE FOMENTO DE LAS ENERGÍAS RENOVABLES EN ESPAÑA*. 1999, pp. 1–302. Accessed: May 10, 2023. [Online]. Available: https://www.idae.es/uploads/documentos/documentos_4044_PFER2000-10_1999_1cd4b316.pdf

[9] Pedro A. Prieto González Jefe Departamento Doméstico y Edificios, *PLAN DE ACCIÓN DE AHORRO Y EFICIENCIA ENERGÉTICA EN ESPAÑA 2008-2012: AYUDAS ECONÓMICAS PARA EL SECTOR DE LA EDIFICACIÓN Y EQUIPAMIENTO*. ESPAÑA, 2008, pp. 1–6. Accessed: May 10, 2023. [Online]. Available: https://www.ceisp.com/fileadmin/pdf/Medidas_PAEE_2008_II.pdf

[10] I. de D. y A. de E. IDAE, *Resumen del Plan de Energías Renovables 2011-2020*. 2011, pp. 1–64. Accessed: May 10, 2023. [Online]. Available: https://www.idae.es/sites/default/files/documentos/publicaciones_idae/documentos_resumen_per_2011-2020_15f3dad6.pdf

[11] Red Eléctrica de España, "AVANCE DEL INFORME DEL SISTEMA ELÉCTRICO ESPAÑOL 2018," Jan. 2019. Accessed: May 11, 2023. [Online]. Available: https://www.ree.es/sites/default/files/11_PUBLICACIONES/Documentos/InformesSistemaElectrico/2019/Avance_ISE_2018.pdf

[12] Ministerio para la Transición Ecológica y Reto Demográfico, *PLAN NACIONAL INTEGRADO DE ENERGÍA Y CLIMA 2021-2030*. 2020, pp. 1–427. Accessed: May 11,

2023. [Online]. Available: https://www.miteco.gob.es/images/es/pnieccompleto_tcm30-508410.pdf

[13] Cayetano Espejo Marín, “ENERGÍA SOLAR FOTOVOLTAICA EN ESPAÑA,” MURCIA, 2004. Accessed: May 12, 2023. [Online]. Available: file:///C:/Users/marco/Downloads/Dialnet-LaEnergiaSolarFotovoltaicaEnEspana-1173549.pdf

[14] Jefatura del Estado, *Real Decreto-ley 15/2018, de 5 de octubre, de medidas urgentes para la transición energética y la protección de los consumidores.* 2018. Accessed: May 12, 2023. [Online]. Available: https://www.boe.es/eli/es/rdl/2018/10/05/15/con

[15] Red Eléctrica, “Red Eléctrica - Informes del Sistema Eléctrico,” *Informes del sistema eléctrico*, 2023. https://www.sistemaelectrico-ree.es/informe-del-sistema-electrico/generacion/generacion-de-energia-electrica/generacion-renovable-de-energia-electrica (accessed May 12, 2023).

[16] P. Temin, *Steam and waterpower in the early nineteenth century*, 2nd ed., vol. 26. J Econ Hits, 1966.

[17] R.L.Hills, *Power in the industrial revolution*. Manchester University Press, 1970.

[18] J. Mokyr, *The Second Industrial Revolution, 1870-1914*. Northwesterm University , 1998.

[19] D. Hochfelder, *The telegraph in America, 1832-1920*. JHU Press, 2012.

[20] J. Taalbi, *Origins and pathways of innovation in the third industrial revolution1*, 5th ed., vol. 28. Industrial and Corporate Change, 2019.

[21] J. Rifkin, “Leading the Way to the Third Industrial Revolution: A New Energy Agenda for the European Union in the 21 st Century-The Next Phase of European Integration-Executive Summary.”

[22] G.L. Ribeiro, *¿ Cuánto más grande mejor? Proyectos de gran escala: una forma de producción vinculada a la expansión de sistemas económicos*. 1987.

[23] T. Ohno, *El sistema de producción Toyota: más allá de la producción a gran escala*. 2018.

[24] Ë. Alsene, “The computer integrated of enterprise,” in *The computer integrated of enterprise*, 1999.

[25] J.A. Rehg, *Computer-integrated manufacturing*. 1995.

[26] S. P. T.C. Cheng, *Just-in-time manufacturing: an introduction*. 1996.

[27] Marcelo Martínez, “RADIACIÓN SOLAR - CONCEPTOS Y APLICACIONES,” May 2016. Accessed: May 15, 2023. [Online]. Available: https://puntoganadero.cl/imagenes/upload/_5cc085baa668a.pdf#:~:text=Ahora%20bien%2C%20la%20radiaci%C3%B3n%20solar%20que%20llega%20a,%28radiaci%C3%B3n%20reflejada%20por%20la%20superficie%20terrestre%29%20%28Figura%202%29.

[28] K. , P. L. , R. A. A. evna, A. K. , L. V. , & S. E. Ranabhat, “An introduction to solar cell technology,” in *An introduction to solar cell technology*, Journal of Applied Engineering Science, Ed., 2016, pp. 481–491.

[29] "¿Qué tipos de inversor fotovoltaico existen?" https://www.mpvsolarreference.com/post/qu%C3%A9-tipos-de-inversor-fotovoltaico-existen (accessed Jun. 21, 2023).

[30] Roberto Herrera Salcedo, "MODELADO Y CARACTERIZACIÓN DE PANELES SOLARES".

[31] Departamento de ingeniería de sistemas y automática, "SUPERVISIÓN INDUSTRIAL HMI."

[32] Departamento de ingeniería de sistemas y automática, "SISTEMAS SCADA."

[33] "Solarmart." https://solarmat.es/es/accesorios-analizador-de-red/software-power-studio-scada.html (accessed May 22, 2023).

[34] "Monsol Scada solar." https://www.solarweb.net/directorio/empresa/4504/Monsol (accessed May 22, 2023).

[35] "Abora Solar Technology." https://abora-solar.com/panel-solar-hibrido/monitorizacion/ (accessed May 22, 2023).

[36] K. A. H. M. S. H. S.R. Akbar, "Message queue telemetry transport protocols implementation for wireless sensor networks communication—A performance review," 2017.

[37] V. Gazis, *A Survey of Standards for Machine-to-Machine and the Internet of Things*, vol. 19. " IEEE Communications Surveys & Tutorials, 2016.

[38] S. E. L. C. K. Rose, *The internet of things: An overview*, vol. 80. The internet society (ISOC), 2015.

[39] D. E. Comer and H. A. A. Soto, *Redes globales de información con Internet y TCP/IP*, vol. 1. Prentice hall, 1996.

[40] F. K. S. S. and J. S. T. Jager, "On the security of TLS-DHE in the standard model." Annual Cryptology Conference, 2012.

[41] B. Harris and R. Hunt, *TCP/IP security threats and attack methods*, vol. 22. Comput Commun, 1999.

[42] K. O. K. H. P. P. and P. K. N. Tantitharanukul, "MQTT-topics management system for sharing of open data," in *MQTT-topics management system for sharing of open data*, International Conference on Digital Arts, Media and Technology (ICDAMT), 2017, pp. 62–65.

[43] U. D. Black, *IP routing protocols: RIP, OSPF, BGP, PNNI, and Cisco routing protocols*. Prentice Hall Professional, 2000.

[44] "MySQL." https://www.mysql.com/ (accessed Jun. 21, 2023).

[45] Apache Friends, "Apache Friends Xampp." https://www.apachefriends.org/es/download.html (accessed Jun. 15, 2023).

[46] "ESP32WROOM32 Datasheet," 2023. [Online]. Available: https://www.espressif.com/en/support/download/documents.

[47] Silicon Labs, “CP210x USB to UART Bridge VCP Drivers,” *Silicon Labs*. https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads (accessed May 25, 2023).

[48] “ESP32-CAM Development Board.”

[49] Ramón Pallàs Areny, *Sensores y acondicionadores de señal,* 2nd ed. Barcelona: Marcombo.

[50] “KY-028 Temperature Sensor module (Thermistor) KY-028 Temperature Sensor module (Thermistor) Contents.”

[51] P. By ALLDATASHEETCOM, “ACS712_06 ALLEGRO | Alldatasheet.” [Online]. Available: www.allegromicro.com

[52] A. B. G. E. Al Stevens, *Thinking in C++*, 2nd ed., vol. 1. 2000. Accessed: Jun. 06, 2023. [Online]. Available: http://vergil.chemistry.gatech.edu/resources/programming/pdf/TIC2Vone.pdf

[53] “Visual Studio.” https://code.visualstudio.com/ (accessed Jun. 06, 2023).

[54] “Arduino IDE 2.1.0.” https://www.arduino.cc/en/software (accessed Jun. 06, 2023).

[55] “Node RED. Low-code programming for event-driven applications.” https://nodered.org/ (accessed Jun. 06, 2023).

[56] nodered.org, “Node-RED-Node-MySQL,” *nodered.org*. https://flows.nodered.org/node/node-red-node-mysql (accessed Jun. 15, 2023).

[57] “node-red-dashboard (node) - Node-RED.” https://flows.nodered.org/node/node-red-dashboard (accessed Jun. 20, 2023).

[58] “Eclipse Mosquitto™ Un corredor MQTT de código abierto.” https://mosquitto.org/ (accessed Jun. 06, 2023).

[59] A. De, S. Operativos, J. Raúl, and L. Medina, “Protocolo SSH,” 2004.

[60] “Libreria WiFi.h Arduino IDE,” *ARDUINO HOME*. https://reference.arduino.cc/reference/en/libraries/wifi/ (accessed Jun. 12, 2023).

[61] “PubSubClient - Arduino Libraries.” https://www.arduinolibraries.info/libraries/pub-sub-client (accessed Jun. 21, 2023).

[62] “node-red-node-mysql (node) - Node-RED.” https://flows.nodered.org/node/node-red-node-mysql (accessed Jun. 21, 2023).

[63] “Create a single message from separate streams of messages : Node-RED.” https://cookbook.nodered.org/basic/join-streams (accessed Jun. 21, 2023).

[64] “Splitting data from msg.payload - General - Node-RED Forum.” https://discourse.nodered.org/t/splitting-data-from-msg-payload/14473 (accessed Jun. 21, 2023).

[65] “node-red-contrib-ui-media (node) - Node-RED.” https://flows.nodered.org/node/node-red-contrib-ui-media (accessed Jun. 21, 2023).

[66] “node-red-node-base64 (node) - Node-RED.” https://flows.nodered.org/node/node-red-node-base64 (accessed Jun. 27, 2023).

Anexo 1. Código C++ Microcontrolador

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

//*****************************************
//********CONFIGURACIÓN WIFI***************
//*****************************************
#define ssid_Secret "MOVISTAR_CCE0"
#define password_Secret "jSfWsrzeAkGU28GPD5Qp"
const char* ssid = ssid_Secret;     //contiene el nombre de la red WIFi
const char* password = password_Secret;  //contiene la contraseña de
la red WiFi

//*****************************************
//*********CONFIGURACIÓN MQTT**************
//*****************************************
const char* mqtt_server = "168.192.1.36";
const int port_server= 1883;

//*****************************************
//**********VARIABLES TEMPERATURA**********
//*****************************************
int Tpin=33;
int lecturaTemp=0;
float Temp;

//*****************************************
//**********VARIABLES VOLTAJE**************
//*****************************************
const int voltajeMax=25000; //16.5V-->16500mv
int lecturaVoltaje=0;  //valor entrada analogica 0-4095
float voltaje;
int pinVoltaje=35;

//*****************************************
//************VARIABLES CORRIENTE**********
//*****************************************
int lecturaCorriente=0;
float corriente;
int pinCorriente=32;
float i;

//*****************************************
//*************VARIABLES POTENCIA**********
//*****************************************
float potencia[10];
float suma=0;
float media;
float p;

//*****************************************
//*************VARIABLES MENSAJE***********
//*****************************************
long lastMsg = 0;
char* msg[50];
int value = 0;
```

```
//*****************************************
//************VARIABLES DE ALERTA************
//*****************************************
int error=0;
bool altatemperatura;
bool bajatemperatura;
bool circuitoabierto;
int estado;


//damos nombre al cliente ESP, para el caso de tener más de un cliente
WiFiClient espClient;
PubSubClient client(espClient);



void setup() {

  Serial.begin(115200); //inicia el puerto serie en 115200 baulios
  conectar_wifi();      //llama a la función para conectar al wifi
  client.setServer("192.168.1.36",1883);  //inicia conexión con
servidor
}

void loop() {

  //si no esta conectado, llama a reconectar
  if(!client.connected()){
    reconnect();
  }
  client.loop();

  //*************************************
  //*********LECTURA SENSORES**************
  //*************************************

  //LECTURA VOLTAJE
  lecturaVoltaje = analogRead(pinVoltaje);
  voltaje = (map(lecturaVoltaje, 0, 4095, 0, voltajeMax)/1000.0)+1;

  //LECTURA TEMPERATURA
  lecturaTemp = analogRead(Tpin);
  Temp = map(lecturaTemp, 573, 4095, 180, 0)-55;  //573 es el valor
para 3.3V
   //mapeo de 573 a 4095 ya que lo alimento a 3.3V desde ESP32

  //LECTURA DE CORRIENTE
  lecturaCorriente = analogRead(pinCorriente);
  corriente = map(lecturaCorriente, 0, 4095, 0, 2);

  //CÁLCULO DE POTENCIA

  p=voltaje*corriente;

  //ALERTAS DE TEMPERATURA
  if (Temp>28){
    altatemperatura=1;
  }
    else if (Temp<3){
      bajatemperatura=1;
```

```
    }
    else{
      altatemperatura=0;
      bajatemperatura=0;
    }

  //ALERTAS ELECTRICAS
  if (voltaje>6 && corriente<0.1){
    circuitoabierto=1;
  }
  else{
    circuitoabierto=0;
  }
  if (voltaje>8 && corriente>=0.3){
    estado=0;
   }
  else if (voltaje>8 && corriente<0.3){
    estado=1;
   }
  else if (voltaje<=8){
    estado=2;
  }


  //LLAMADA A FUNCIONES DE IMPRIMIR LOS VALORES EN PUERTO SERIE
  imprimirVoltaje(voltaje);
  imprimirTemperatura(Temp);
  imprimirIntensidad(corriente);
  Serial.println(media);
  delay(1000);

  //CONVERTIR FLOAT A CHAR ARRAY
  char volt[8];
  dtostrf(voltaje, 1, 2, volt);
  char tempe[8];
  dtostrf(Temp, 1, 2, tempe);
  char corriente[8];
  dtostrf(i, 1, 2, corriente);
  char altaT[8];
  dtostrf(altatemperatura, 1, 2, altaT);
  char bajaT[8];
  dtostrf(bajatemperatura, 1, 2, bajaT);
  char coff[8];
  dtostrf(circuitoabierto, 1, 2, coff);
  char pot[8];
  dtostrf(p, 1, 2, pot);
  char est[8];
  dtostrf(estado, 1, 2, est);

  client.publish("esp32/voltaje",volt);
  client.publish("esp32/temperatura",tempe);
  client.publish("esp32/intensidad",corriente);
  client.publish("esp32/altatemperatura",altaT);
  client.publish("esp32/bajatemperatura",bajaT);
  client.publish("esp32/circuitoabierto",coff);
  client.publish("esp32/potencia",pot);
  client.publish("esp32/estado", est);
 }


//*****************************
```

```
//*******FUNCIONES PUERTO*******
//**********SERIE***************
//******************************
void imprimirVoltaje(float v){
  Serial.print("Voltaje =");
  Serial.print(v);
  Serial.println("V");
  Serial.println(lecturaVoltaje);
}

void imprimirTemperatura(float T ){
  Serial.print(T);
  Serial.println("°C");
}

void imprimirIntensidad(float I){
  Serial.print(I);
  Serial.println("A");
}

//******************************
//*******FUNCION WIFI***********
//******************************
void conectar_wifi(){

  WiFi.begin(ssid,password); //inicia la configuración WiFi

  while (WiFi.status()!= WL_CONNECTED){ //wifi.status devuelve el
estado de la conexión
    delay(500);                         //mientras no este conectado
devuelve mensaje
    Serial.print("Conectando a red WiFi..."); //conectando a red
    Serial.println(ssid);
  }
  Serial.print("Conectado a red WiFi:"); //una vez conectado, devuelve
mensaje de conectado
  Serial.println(ssid);
  Serial.print("Dirección IP ESP32:");
  Serial.println(WiFi.localIP()); //imprime IP de la ESP32
}

//*********************************
//*****FUNCION ESTADO MQTT*********
//*********************************
void reconnect(){

  if(client.connect("ESP32")){

    Serial.println("Conectado a mosquitto");
  }
  else{

    Serial.println("Conectando a mosquitto...");
  }
}
```

Anexo 2. Código C++ ESP32 CAM

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

//****************************************
//********CONFIGURACIÓN WIFI***************
//****************************************
const char* ssid = "***";
const char* password = "***";
//****************************************
//*********CONFIGURACIÓN MQTT**************
//****************************************
const char* mqttServer = "***";
const int mqttPort = 1883;
//SE DEFINEN PINES DEL MÓDULO ESP32CAM
#define CAMERA_MODEL_AI_THINKER
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27

#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22
int sacarFoto;
//damos nombre al cliente ESPCAM, para el caso de tener más de un
cliente
WiFiClient espcamClient;
PubSubClient client(espcamClient);

//*****************************
//*******FUNCION WIFI***********
//*****************************
void conectar_wifi(){

  WiFi.begin(ssid,password); //inicia la configuración WiFi

  while (WiFi.status()!= WL_CONNECTED){ //wifi.status devuelve el
estado de la conexión
    delay(500);                         //mientras no este conectado
devuelve mensaje
    Serial.print("Conectando a red WiFi..."); //conectando a red
    Serial.println(ssid);
  }
  Serial.print("Conectado a red WiFi:"); //una vez conectado, devuelve
mensaje de conectado
```

```
  Serial.println(ssid);
  Serial.print("Dirección IP ESP32:");
  Serial.println(WiFi.localIP()); //imprime IP de la ESP32
}
void callback(char* topic, byte* payload, unsigned int length){

  payload[length]='\0';
  String mensaje = String((char*)payload);
  if (mensaje="tomar foto"){
    sacarFoto=1;
  }
  else{
    sacarFoto=0;
  }
}
//*********************************
//*****FUNCION ESTADO MQTT*********
//*********************************
void reconnect(){

  if(client.connect("ESP32")){

    Serial.println("Conectado a mosquitto");
  }
  else{

    Serial.println("Conectando a mosquitto...");
  }
}

void setup() {

  Serial.begin(115200);//inicia el puerto serie en 115200 baulios
  conectar_wifi();     //llama a la función para conectar al wifi
  client.setServer(mqttServer, mqttPort);//inicia conexión con
servidor
  client.subscribe("esp32/cam/fotografiar");

 //FUNCIÓN DE LECTURA QUE SE EJECUTA AL RECIBIT MENSAJE POR EL TOPIC
  client.setCallback(callback);

  //INICIALIZA Y CONFIGURA LA CÁMARA
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
```

```
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;

  //CONFIGURA DIMENSION Y CALIDAD DE IMAGEN
  if(psramFound()){
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }
  //ARRANCA LA CÁMARA
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
  if(sacarFoto=1){
camera_capture();
  }
}

  esp_err_t camera_capture(){

      //capture a frame
      camera_fb_t * fb = esp_camera_fb_get();
      if (!fb) {
          ESP_LOGE(TAG, "Frame buffer could not be acquired");
          return ESP_FAIL;
      }

         client.beginPublish("esp32/cam/foto",  fb->len, false);
      client.write(fb->buf, fb->len);
      client.endPublish();

      esp_camera_fb_return(fb);
      return ESP_OK;
  }

void loop() {
  //si no esta conectado, llama a reconectar
  if(!client.connected()){
    reconnect();
  }
  client.loop();
}
```

Anexo 3. Funcionalidades del sistema

En la ilustración Anexo 3.1 se ofrece una visión general del sistema sobre el conexionado y situación de dispositivos de medida. Cabe destacar, la inserción del sensor de temperatura entre ambos paneles, para medir la temperatura a la que se encuentran.

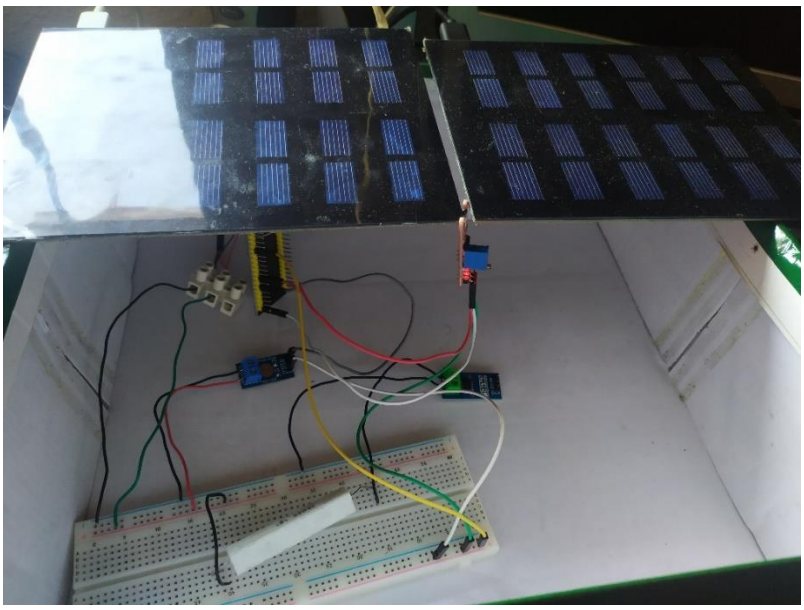

Ilustración Anexo 3.1 Visión global del sistema Fuente: Propia

Durante el funcionamiento del sistema fotovoltaico bajo condiciones normales de trabajo, se obtienen las ilustraciones Anexo 3.2 y 3.3 donde se aprecian valores de temperatura, voltaje e intensidad normales. Además, se visualizan físicamente el estado los paneles fotovoltaicos y un mensaje de funcionamiento corrector del sistema.

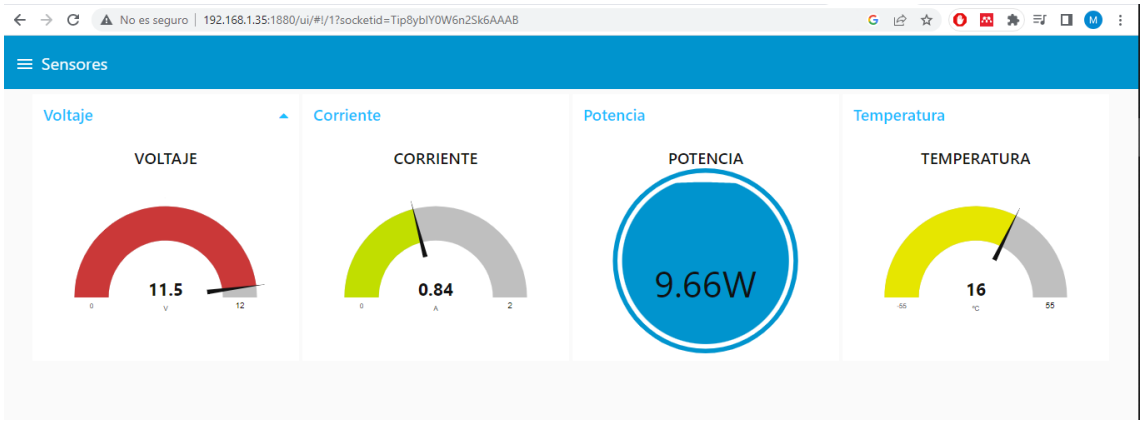

Ilustración Anexo 3.2 Panel Sensores en condiciones normales de trabajo Fuente: Propia

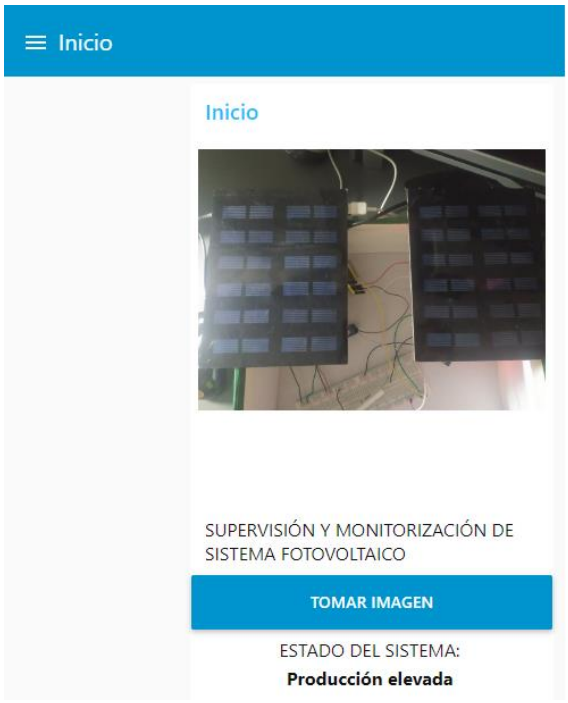

Ilustración Anexo 3.3 Panel de Inicio durante condiciones normales de trabajo Fuente: Propia

En las ilustraciones 3.4 y 3.5, se observa un valor de corriente bajo en el funcionamiento del sistema, viéndose reflejado a tiempo real en la pantalla de sensores. Además, muestra una alerta de circuito abierto como indicativo de posible problema, así como en la pantalla de inicio, el mensaje de producción baja o nula del sistema.

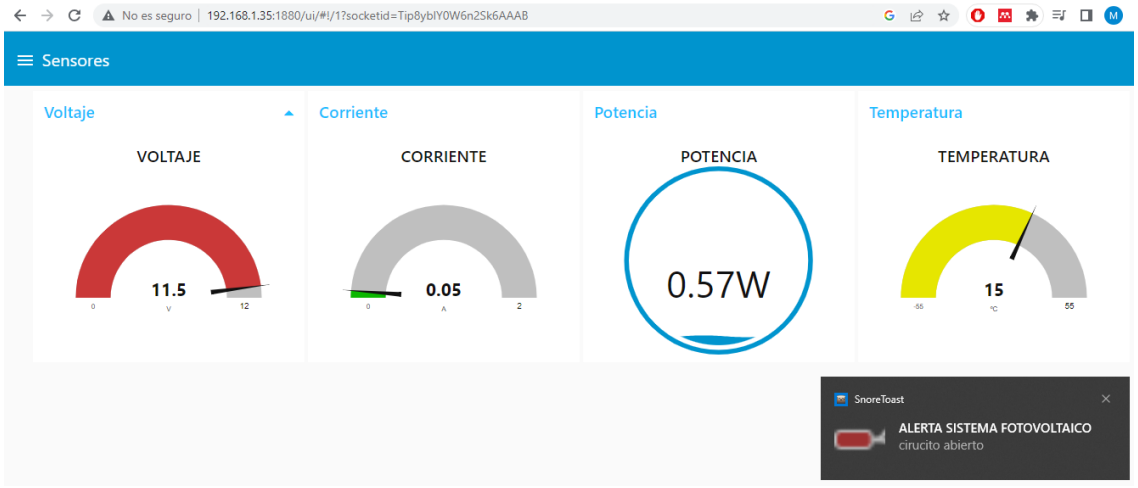

Ilustración Anexo 3.4 Pantalla Sensores con valor de corriente bajo Fuente: Propia

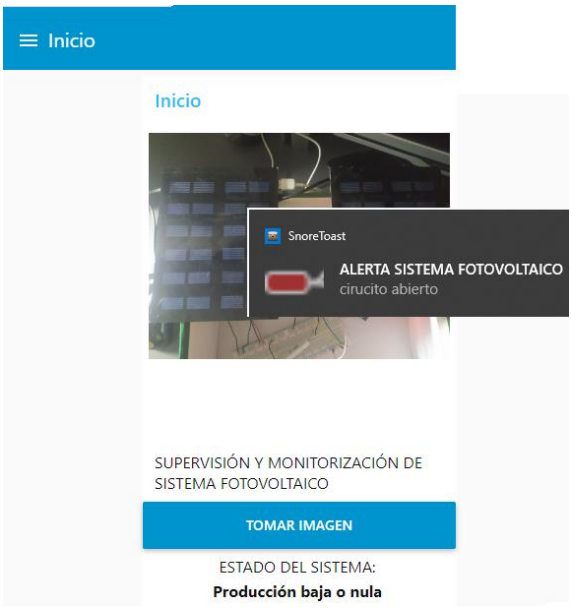

Ilustración Anexo 3.5 Pantalla Inicio con mensaje de alerta y estado bajo de producción Fuente: Propia

Además de alertas por producciones bajas y circuito abierto, se pueden recibir alertas por temperatura en paneles elevadas o inferiores a 0ºC. Estos paneles se visualizan de la misma forma que en la alerta de la ilustración 3.5.

Ilustración Anexo 3.6 Panel de Inicio durante producción de paneles media Fuente: Propia

En la ilustración 3.7 se observa la pantalla de inicio tras haber realizado la captura de imagen desde otra perspectiva, y así poder comprobar el correcto funcionamiento del sistema de captura de imagen.

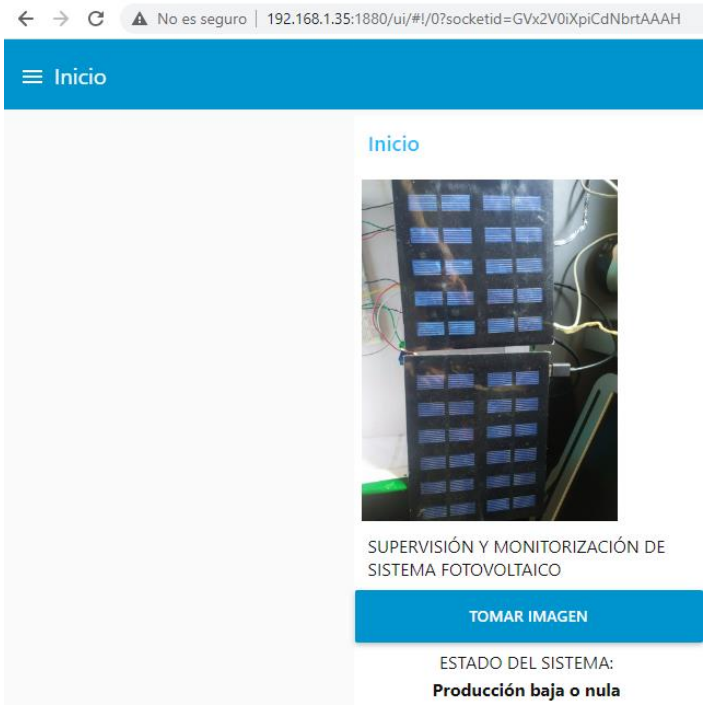

Ilustración Anexo 3.7 Panel de Inicio Fuente: Propia