Escuela de Doctorado
y Estudios de Posgrado
Universidad de La Laguna

Máster Universitario en Ingeniería Industrial

# Trabajo Fin de Máster

---

# *Desarrollo de un Sistema de Monitorización y Algoritmo Inteligente de Predicción para Comunidades Energéticas*

---

**Autor**: Ángel Marcos Trujillo Trujillo

**Tutor**: Juan Albino Méndez Pérez

**Cotutor:** José Manuel González Cava

*Mayo de 2024*

*La publicación de este Trabajo Fin de Máster solo implica que el estudiante ha obtenido al menos la nota mínima exigida para superar la asignatura correspondiente, no presupone que su contenido sea correcto, aunque si aplicable. En este sentido, la ULL no posee ningún tipo de responsabilidad hacia terceros por la aplicación total o parcial de los resultados obtenidos en este trabajo. También pone en conocimiento del lector que, según la ley de protección intelectual, los resultados son propiedad intelectual del alumno, siempre y cuando se haya procedido a los registros de propiedad intelectual o solicitud de patentes correspondientes con fecha anterior a su publicación.*

## Agradecimientos

En primer lugar, me gustaría agradecer a mis tutores de este Trabajo Fin de Máster, D. Juan Albino Méndez Pérez y D. José Manuel González Cava, por su dedicación y compromiso. Gracias por la confianza depositada en mí durante la realización de este proyecto.

Quiero expresar mi agradecimiento también al proyecto "Sustainable Atlantic Communities (SAtComm)" EAPA_0019/2022, cofinanciado por la Unión Europea a través del programa Interreg Atlantic Area, que ha brindado apoyo al trabajo desarrollado en este TFM.

Del mismo modo, quiero destacar el apoyo del Vicerrectorado de Sostenibilidad e Infraestructuras de la ULL, que ha facilitado el acceso a las infraestructuras y datos necesarios para llevar a cabo este proyecto.

# Contenidos Generales

# Índice de Tablas

# Índice de Figuras

## Resumen

El objetivo del trabajo es doble, por un lado, se centrará en la monitorización de la energía fotovoltaica producida y la energía consumida de las diferentes instalaciones de la ULL. Por otra parte, se desarrollará un modelo basado en aprendizaje automático para la predicción de la energía generada en las instalaciones.

En la primera parte del trabajo se desarrollará un sistema de monitorización energética en instalaciones fotovoltaicas usando una plataforma basada en Internet de las Cosas (IoT). La aplicación desarrollada consistirá en la monitorización de las instalaciones de generación fotovoltaica de la Universidad de La Laguna (ULL) y usará la plataforma *Thingsboard* que permite recolectar, visualizar y analizar datos de dispositivos conectados a internet de una manera eficiente.

Para la implementación del sistema, se utilizarán herramientas avanzadas de extracción de datos, incluyendo técnicas de Web Scraping y Automatización Robótica de Procesos (RPA), que permitirán recoger datos en tiempo real de los inversores fotovoltaicos y otros dispositivos. Estos datos serán integrados y sincronizados en ThingsBoard, facilitando la visualización y el análisis a través de cuadros de mando intuitivos.

En la segunda parte del trabajo se desarrolló un modelo de predicción utilizando redes neuronales de tipo Long Short-Term Memory (LSTM) para anticipar la generación de energía fotovoltaica del Edificio SEGAI de la ULL. Este modelo no solo permitirá prever la generación de energía en función de variables históricas y condiciones ambientales, sino que también proporcionará una herramienta valiosa para la toma de decisiones y la gestión eficiente del consumo energético.

El proyecto desarrollado establece la base para la implementación de una comunidad energética que se está desarrollando en la Universidad de La Laguna en el contexto del proyecto Europeo “Sustainable Atlantic Communities” (SAtComm).

**Palabras clave:** Monitorización, Generación fotovoltaica, IoT, ThingsBoard, Web Scraping, RPA, APIs, Red Neuronal Recurrente, LSTM.

## Abstract

The objective of the work is twofold, on the one hand, it will focus on monitoring the photovoltaic energy produced and the energy consumed by the different facilities of the ULL. On the other hand, a model based on automatic learning will be developed for the prediction of the energy generated in the installations.

In the first part of the work, an energy monitoring system for photovoltaic installations using a platform based on the Internet of Things (IoT) will be developed. The application will consist of monitoring the photovoltaic generation facilities of the University of La Laguna (ULL) and will use the *Thingsboard* platform that allows the collection, visualization and analysis of data from devices connected to the Internet in an efficient way.

For the implementation of the system, advanced data extraction tools will be used, including Web Scraping and Robotic Process Automation (RPA) techniques, which will allow real-time data collection from photovoltaic inverters and other devices. This data will be integrated and synchronized in ThingsBoard, facilitating visualization and analysis through intuitive dashboards.

In the second part of the work, a prediction model was developed using Long Short-Term Memory (LSTM) neural networks to anticipate the photovoltaic energy generation of the SEGAI building of the ULL. This model will not only allow forecasting energy generation based on historical variables and environmental conditions, but will also provide a valuable tool for decision-making and efficient management of energy consumption.

The developed project establishes the basis for the implementation of an energy community that is being developed at the University of La Laguna in the context of the European project "Sustainable Atlantic Communities" (SAtComm).

**Keywords:** Monitoring, Photovoltaic generation, IoT, ThingsBoard, Web Scraping, RPA, APIs, Recurrent Neural Network, LSTM.

# 1. Introducción

En este capítulo se establece el marco contextual del trabajo, comenzando por una revisión de los antecedentes del mismo, así como la relevancia y las potenciales contribuciones que se derivan. Posteriormente, se definen los objetivos perseguidos diferenciando entre los propósitos generales y los específicos.

## 1.1. Antecedentes

La transición energética representa un movimiento global necesario y estratégico hacia la sustitución de fuentes de energía basadas en combustibles fósiles por alternativas renovables. Este cambio profundo en la producción y consumo de energía es esencial para abordar los desafíos medioambientales que presentan el cambio climático y la degradación ambiental. Esta urgencia es especialmente crítica en un territorio como Canarias caracterizado por un sistema eléctrico aislado, de mayor fragilidad y con una dependencia muy elevada de fuentes de energía fósil.

La relevancia de la transición energética se debe a su capacidad para mitigar el impacto negativo de los combustibles fósiles en el medio ambiente. A nivel global, en torno al 80% del consumo energético proviene de fuentes fósiles, cuya explotación masiva acarrea consecuencias ambientales adversas, tales como la contribución al calentamiento global a través de emisiones de CO2 y otros gases de efecto invernadero, y la acidificación de ciclos hídricos [1]. Este cambio climático antropogénico lleva consigo el riesgo de alteraciones en los patrones climáticos, la elevación del nivel del mar, la acidificación de los océanos y un incremento en eventos climáticos extremos.

Ante este panorama, una transición energética orientada hacia la erradicación de combustibles fósiles en favor de fuentes renovables se vuelve esencial. Esta transición está potenciada no solo por la imperativa ambiental sino también por la inminente escasez de fuentes fósiles y la inestabilidad geopolítica derivada de su desigual distribución [2]. Las energías renovables, más allá de su carácter sostenible, han demostrado ser económicamente viables, adelantando incluso a las fuentes fósiles en términos de costos.

Por tanto, la transición energética no es solo una respuesta a la crisis medioambiental, sino también una estrategia de desarrollo integral que promueve la sostenibilidad, la equidad y la prosperidad a largo plazo para la humanidad.

En este contexto, en Canarias se ha desarrollado un Plan de Transición Energética denominado Plan de Transición Energética de Canarias (PTECan) que tratará de impulsar el empleo de energías renovables, reduciéndose así las emisiones contaminantes [3]. Este ambicioso plan propone cubrir gran parte de las finalidades de la Ley Canaria de Cambio Climático mediante la coordinación con la Estrategia y el Plan de Acción Climáticas, en la línea de un futuro Plan Integral de Lucha contra el Cambio Climático.

El PTECan se organiza conforme al Reglamento (UE) 2018/1999, asegurando una alineación completa con las directrices del Gobierno de España y el Plan Nacional Integrado de Energía y Clima (PNIEC). Se distingue por integrar distintos niveles de gobernanza, incluyendo el Parlamento de Canarias, un Comité Internacional de Supervisión, el Observatorio Energético de Canarias y las distintas Agencias Insulares de Energía.

Para 2030, se plantea que el 60% de la demanda eléctrica del archipiélago sea cubierta por fuentes renovables, marcando un progreso significativo hacia la meta de descarbonización total de la economía de Canarias para 2040, tal como se establece en la

Declaración de Emergencia Climática del Gobierno de Canarias. El plan prevé diferentes trayectorias hacia la descarbonización, contemplando escenarios para los años actuales hasta el 2050.

En la vanguardia de las diferentes iniciativas, las comunidades energéticas emergen como un instrumento clave para democratizar la energía, promoviendo la participación ciudadana y local en la producción y gestión de la energía renovable. La comunidad energética no solo refuerza la autonomía y resiliencia local sino que también alinea a los consumidores con los objetivos más amplios de sostenibilidad y descarbonización del sistema energético.

Las comunidades energéticas representan un enfoque innovador y cooperativo que permite a los participantes compartir energía renovable localmente generada. Estas comunidades enfatizan la producción local de energías renovables a través de la instalación y gestión de unidades de producción propiedad de los ciudadanos, lo que varía desde paneles solares en hogares individuales hasta proyectos comunitarios de mayor escala. Ofrecen beneficios socioeconómicos y ambientales, incluyendo la reducción de costos energéticos, la mejora de la resiliencia local, y la promoción de una transición energética sostenible. Al implicar a la población para que participen activamente en la generación y gestión de la energía, las comunidades energéticas fomentan una gobernanza comunitaria inclusiva y contribuyen a la autonomía energética, la reducción de la dependencia de fuentes de energía no renovables y el apoyo a la sostenibilidad a largo plazo [4].

Además de los beneficios mencionados, las comunidades energéticas impulsan la participación ciudadana activa en el sector energético, crucial para el cambio hacia una economía más verde. Este enfoque ayuda a enfrentar la pobreza energética y promueve una mayor independencia de las compañías eléctricas tradicionales, incrementando la competitividad. El marco legal europeo y español ofrece un sólido soporte para el desarrollo de estas comunidades, fomentando la generación, consumo y almacenamiento de energía renovable, así como la eficiencia energética y la movilidad sostenible. A pesar de sus numerosos beneficios, existen retos como la necesidad de mayor claridad regulatoria y apoyo financiero, esenciales para facilitar su expansión y asegurar su sostenibilidad a largo plazo [5].

En esta línea, la Universidad de La Laguna, a través del grupo de investigación de Ingeniería de Control y Sistemas Inteligentes, participa en un proyecto Europeo centrado en la creación de un modelo de comunidad energética eficiente, que pueda ser replicado en el territorio europeo. El proyecto denominado Sustainable Atlantic Communities (SAtComm) contempla la implementación de un piloto de una comunidad energética en las instalaciones de la Universidad de La Laguna. Este Trabajo Fin de Máster se enmarca dentro de este proyecto y pretende aportar la base para construir la comunidad energética en la ULL.

Tanto a nivel de la mencionada comunidad, como a cualquier otro nivel del sistema eléctrico donde se quiera tener una gestión eficiente, destaca la importancia de la monitorización y la predicción de los datos de energía. Sólo partiendo de una monitorización

y predicción confiable se podrá avanzar hacia la optimización del uso de los recursos renovables para garantizar una transición energética eficiente. La monitorización continua de los datos de energía permite identificar patrones de consumo y detectar áreas de mejora en la generación y distribución de energía. Por otro lado, la capacidad de predecir la generación de energía fotovoltaica es crucial para la gestión eficiente del sistema energético, especialmente en un contexto donde las fuentes renovables, por su naturaleza variable, requieren una integración inteligente en la red eléctrica.

## 1.2. Interés de la propuesta y beneficios esperados

Disponer de un sistema centralizado de monitorización donde se visualicen los datos de las distintas plantas fotovoltaicas objeto de estudio permitirá una gestión más eficaz de los recursos energéticos, identificando oportunidades para optimizar el uso de energía y reducir los posibles desperdicios. Además, facilitará la toma de decisiones informada, permitiendo así, responder rápidamente a cambios de demanda de energía o en la producción de las plantas.

Al monitorizar continuamente el rendimiento de las plantas, el sistema puede identificar cualquier anomalía o falla, permitiendo así identificar funcionamientos fuera de los márgenes deseados, averías o necesidades de mantenimiento. La monitorización energética se llevará a cabo en una avanzada plataforma de Internet de las Cosas (IoT). La implementación de esta tecnología IoT representa un salto cualitativo en la forma en que se recopilan, analizan y gestionan los datos energéticos. Esto nos permitirá comprender mejor cómo y cuándo se consume la energía y desarrollar nuevas tecnologías y métodos para su gestión..

Las plataformas IoT se han convertido en elementos clave en la transformación digital de diversos sectores, incluido el energético radicando su importancia en la capacidad de conectar dispositivos y sistemas a la red, permitiendo la recopilación, el análisis y el intercambio de datos en tiempo real. La importancia de los datos en el escenario tecnológico actual abre todas las posibilidades para la implementación de estrategias basadas en inteligencia artificial para la gestión de los sistemas.

En relación con esto, el interés de esta propuesta también radica en integrar un modelo de predicción basado en aprendizaje automático que nos permita predecirla generación fotovoltaica. Esto representa un avance en la capacidad de anticipar la cantidad de energía que será generada por fuentes solares. Desde la perspectiva de la gestión de la red eléctrica, los beneficios de un modelo de predicción son muy elevados. La capacidad de prever fluctuaciones en la generación de energía solar permite una gestión más refinada de la red, asegurando la estabilidad y la eficiencia del suministro. También se abren posibilidades relacionadas con la gestión de la demanda, que representa una de las líneas de máximo interés en la gestión futura de las redes eléctricas.

En definitiva, esta propuesta no solo representa una mejora en la eficiencia y gestión de la energía solar a través de la monitorización y análisis centralizado, sino que también se alinea con los objetivos globales de sostenibilidad y transición energética. La implementación de un sistema de monitorización es un paso adelante hacia la innovación en el sector energético, ofreciendo una solución integral que aborda tanto las necesidades actuales como los desafíos futuros. A nivel local, la propuesta establece la base sobre la que se podrá desarrollar la implementación de la comunidad energética del proyecto SAtComm en la ULL.

## 1.3. Objetivos

El objetivo principal de este proyecto es diseñar un sistema centralizado para monitorizar las variables energéticas de mayor interés en los edificios de la Universidad de La Laguna que cuenten con instalaciones fotovoltaicas. Asimismo, en el proyecto se plantea el desarrollo de un modelo de predicción para la generación fotovoltaica en un horizonte temporal de 24 horas..

El proyecto se enfoca en siete  edificios de la ULL ya que fueron los pioneros en la implementación de energía fotovoltaica dentro del campus, estableciendo un precedente importante en la transición hacia fuentes de energía renovables en un futuro. Además, la elección de estos edificios se debe a que se cuenta con el acceso a los datos de los distintos inversores fotovoltaicos como al consumo de potencia de cada uno de estos. Los edificios que se monitorizarán son los siguientes:

1. Edificio SEGAI.
2. Edificio Estabulario.
3. Escuela Politécnica Superior de Ingeniería (EPSI).
4. Facultad de Biología.
5. Facultad de Derecho.
6. Facultad de Bellas Artes.
7. Facultad de Ciencias Políticas, Sociales y de la Comunicación.

De forma más detallada, dentro del marco del proyecto se han establecido una serie de objetivos específicos. Estos objetivos están orientados a facilitar la consecución de la finalidad principal del Trabajo Fin de Máster. Los objetivos específicos relacionados con la monitorización son:

- Estudio de las instalaciones fotovoltaicas existentes.
- Implementación y fusión de soluciones tecnológicas avanzadas para la recolección de datos de las diversas instalaciones. Este proceso se llevará a cabo mediante técnicas de WebScraping y Automatización Robótica de Procesos (RPA), con el objetivo de obtener la información necesaria de cada una de las plantas.

- Incorporación y sincronización de los datos recopilados de cada planta en la plataforma de Internet de las Cosas (IoT) *ThingsBoard*. Además, se procederá a registrar y almacenar los históricos de datos correspondientes a cada una de las plantas.
- Desarrollo de un cuadro de mando para la recogida y visualización de la información generada dentro de la plataforma *ThingsBoard*.

Los objetivos específicos relacionados con la implementación de un modelo de predicción basado en técnicas de inteligencia artificial son:

- Implementación de un modelo predictivo empleando técnicas de Machine Learning para anticipar la generación fotovoltaica en uno de los edificios.
- Análisis de diferentes variables y su influencia en los resultados de predicción del modelo.

## 1.4. Estructura de la memoria

El presente documento se organiza en cuatro capítulos principales que tratarán de cubrir todos los aspectos relacionados con la monitorización de variables energéticas y la creación de un modelo que nos ayude a predecir la generación de energía. Se describirá el planteamiento seguido en cada uno de los puntos y los resultados obtenidos.

El primer capítulo es introductorio, donde se muestra el estado actual y los antecedentes relacionados con la energía solar fotovoltaica y comunidades energéticas. También se describe el interés de la propuesta de estudio planteada en el documento y cuáles serán los objetivos que se persiguen en dicho proyecto.

El segundo capítulo establece el contexto teórico y conceptual del proyecto, abordando como se suele llevar a cabo la monitorización de las instalaciones, las tecnologías de recolección de datos y el papel del Internet de las Cosas (IoT) haciendo especial énfasis en el software ThingsBoard. Finalmente, se profundizará en la aplicación de redes neuronales, específicamente en las redes neuronales recurrentes (RNN), como las redes LSTM (Long Short-Term Memory), para la predicción de series temporales.

En el tercer capítulo se describirán las distintas metodologías empleadas durante el desarrollo del proyecto. Se incluye una descripción general de la estructura del sistema de monitorización en la que nos basaremos. Acto seguido, se exponen las técnicas y el desarrollo empleado para extraer los datos y su proceso de incorporación y sincronización con la plataforma ThingsBoard. Además, se incluye el desarrollo de cómo se realizaron los diferentes cuadros de mando para visualizar la información. Por último, se expone la metodología llevada a cabo para desarrollar un modelo de predicción basado en técnicas de Machine Learning.

El cuarto capítulo se centra en presentar los distintos resultados obtenidos. Se expondrán los cuadros de mando implementados para cada uno de los edificios, así como los

resultados del modelo de predicción desarrollado y su correspondencia con la generación real.

# 2. Contextos y fundamentos teóricos

En este capítulo se establece el contexto teórico y conceptual del proyecto. Se abordará una introducción a las comunidades energéticas y se describirán de forma genérica las instalaciones fotovoltaicas objeto de estudio que se sitúan en la Universidad de la Laguna. Además, se explicarán las tecnologías comúnmente empleadas para monitorizar las instalaciones, las tecnologías de recolección de datos y se introducirá el papel del Internet de las Cosas (IoT) haciendo énfasis en el software ThingsBoard. Finalmente, se profundizará en la aplicación de redes neuronales, específicamente en las redes neuronales recurrentes (RNN), como las redes LSTM (Long Short-Term Memory), para la predicción de series temporales.

## 2.1. Energía Fotovoltaica y Comunidades energéticas

La transición hacia sistemas energéticos sostenibles y renovables es una prioridad global frente al cambio climático y la creciente demanda energética. El presente proyecto se sitúa frente a una iniciativa innovadora en la Universidad de La Laguna que tiene como objetivo implementar una comunidad energética, no solo para aprovechar la energía fotovoltaica sino también para fomentar la participación activas de los usuarios de la universidad en la gestión y producción de energía renovable.

La adopción de este modelo no sólo simboliza un paso hacia adelante en el compromiso con el medio ambiente y la innovación tecnológica, sino que también ofrece múltiples beneficios. Entre ellos, se encuentran la reducción de costes energéticos, la promoción de la conciencia ambiental entre la comunidad universitaria, y la contribución al logro de objetivos de desarrollo sostenible. Además, al integrar a los usuarios en la gestión y decisión sobre la energía que consumen, se fomenta una cultura de responsabilidad y participación activa en la transición energética.

A continuación, se detallarán los fundamentos teóricos que subyacen a las tecnologías fotovoltaicas y a las comunidades energéticas.

### 2.1.1. Principios básicos de la Energía Fotovoltaica

La energía fotovoltaica es la transformación directa de la radiación solar en electricidad. Esta transformación se produce en unos dispositivos denominados paneles fotovoltaicos. En los paneles fotovoltaicos, la radiación solar excita los electrones de un dispositivo semiconductor generando una pequeña diferencia de potencial. La conexión en serie de estos dispositivos permite obtener diferencias de potencial mayores [7].

Por tanto, cuándo hablamos de una instalación solar fotovoltaica, hablamos de una instalación que produce energía eléctrica a partir de la luz solar. Las instalaciones pueden ser fundamentalmente de dos tipos:

***1. Instalaciones aisladas.*** Se prescinde completamente de la red de suministro eléctrico, siendo adecuadas para lugares aislados a los que no llegan las compañías eléctricas. En estas instalaciones para tener electricidad en ausencia de luz solar se deben incorporar baterías.

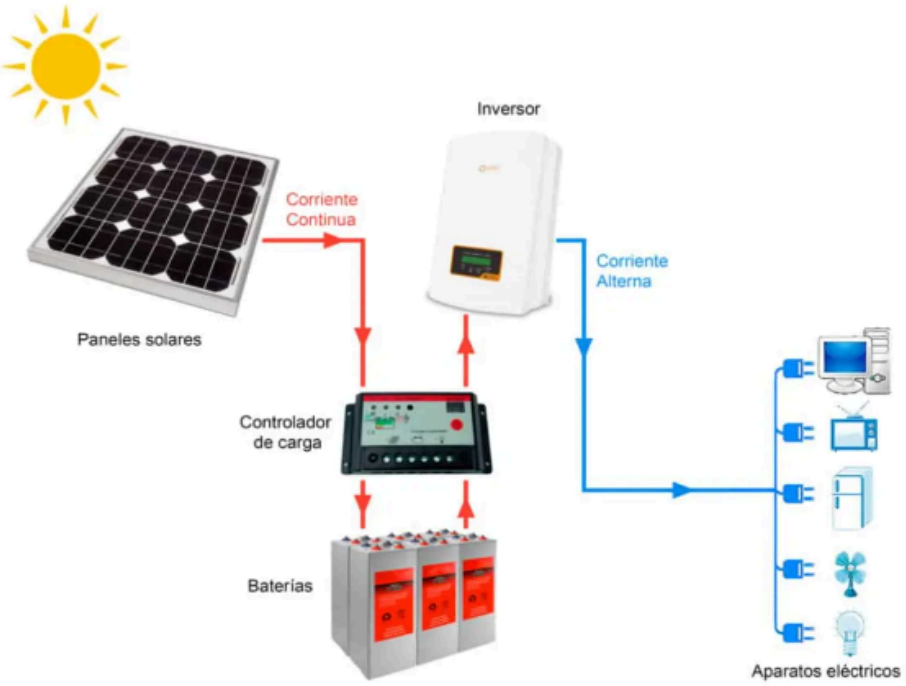


*Figura 2.1: Esquema de instalación de un sistema aislado de la red eléctrica "off grid" [8].*

***2. Instalaciones conectadas a red.*** Estos sistemas fotovoltaicos son los más usados, consisten en utilizar conjuntamente la instalación fotovoltaica con la red eléctrica general. De esta forma se utiliza la red eléctrica como apoyo durante los periodos en los que no hay producción solar. Es el tipo de instalación de autoconsumo más económico y con menos coste de mantenimiento, ya que no se necesitan baterías para tenerlo en funcionamiento.

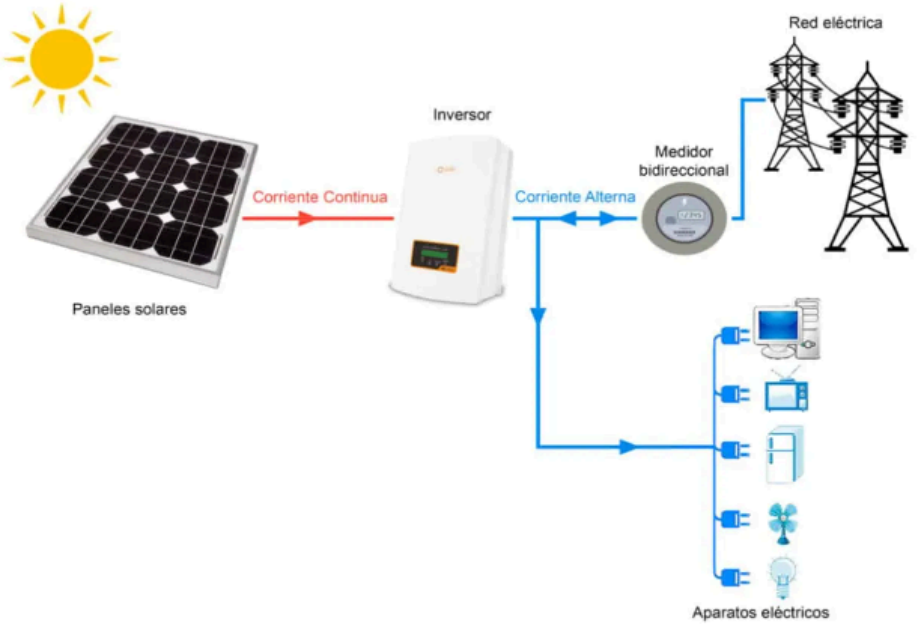


*Figura 2.2: Esquema de instalación de un sistema conectado a la red eléctrica "on grid" [8].*

### 2.1.2. Descripción y componentes de la instalación fotovoltaica en los edificios de la Universidad de La Laguna

Los siete edificios objeto de estudio de este proyecto pertenecientes a la Universidad de La Laguna, disponen de una instalación fotovoltaica conectada a la red en la modalidad de *suministro con autoconsumo sin excedentes*, por lo que, disponen de un *mecanismo antivertido* que impide la inyección de energía excedentaria a la red de transporte o de distribución. Estas instalaciones están en funcionamiento desde el año 2023 y todas ellas disponen de los componentes que se detallan a continuación.

#### 2.1.2.1. Módulos fotovoltaicos

El módulo fotovoltaico adoptado en cada una de las instalaciones es de 420 Wp cada uno, encontrándose conectados en serie formando agrupaciones (strings).

*Figura 2.3: Módulo fotovoltaico.*

#### 2.1.2.2. Inversor fotovoltaico

El inversor es el encargado de transformar la corriente continua en corriente alterna con la que pueden funcionar los diferentes aparatos eléctricos y electrodomésticos que albergan los edificios.

Para la conversión de la corriente continua procedente de los módulos fotovoltaicos a corriente alterna se disponen de inversores de conexión a red en los diferentes edificios del modelo que se muestra en la Figura 2.4.


*Figura 2.4: Inversor fotovoltaico Goodwe.*

### 2.1.2.3. Protecciones eléctricas

En las instalaciones solares fotovoltaicas hay que proteger el cableado entre los paneles y el regulador. Para ello, las instalaciones disponen de fusibles o interruptores automáticos tal como se muestra en la Figura 2.5.


*Figura 2.5: Portafusible.*

La instalación solar también está protegida con dispositivos contra sobretensiones transitorias, que pueden producirse en caso de impacto de rayos y es necesario enviar a tierra.


*Figura 2.6: Dispositivo de protección contra sobretensiones transitorias.*

Además, en las instalaciones se debe proteger el tramo baterías-inversor y los circuitos de consumo en corriente alterna y en corriente continua.

A continuación, a modo de resumen se puede observar el siguiente esquema de protecciones de una instalación solar fotovoltaica conectada a red. En el lado de corriente continua del inversor se aprecian los fusibles y el protector contra sobretensiones. En la parte de corriente alterna del inversor tenemos otro protector contra sobretensiones, un interruptor diferencial que protege ante posibles derivaciones a tierra y un interruptor automático magnetotérmico que protege contra sobreintensidades.

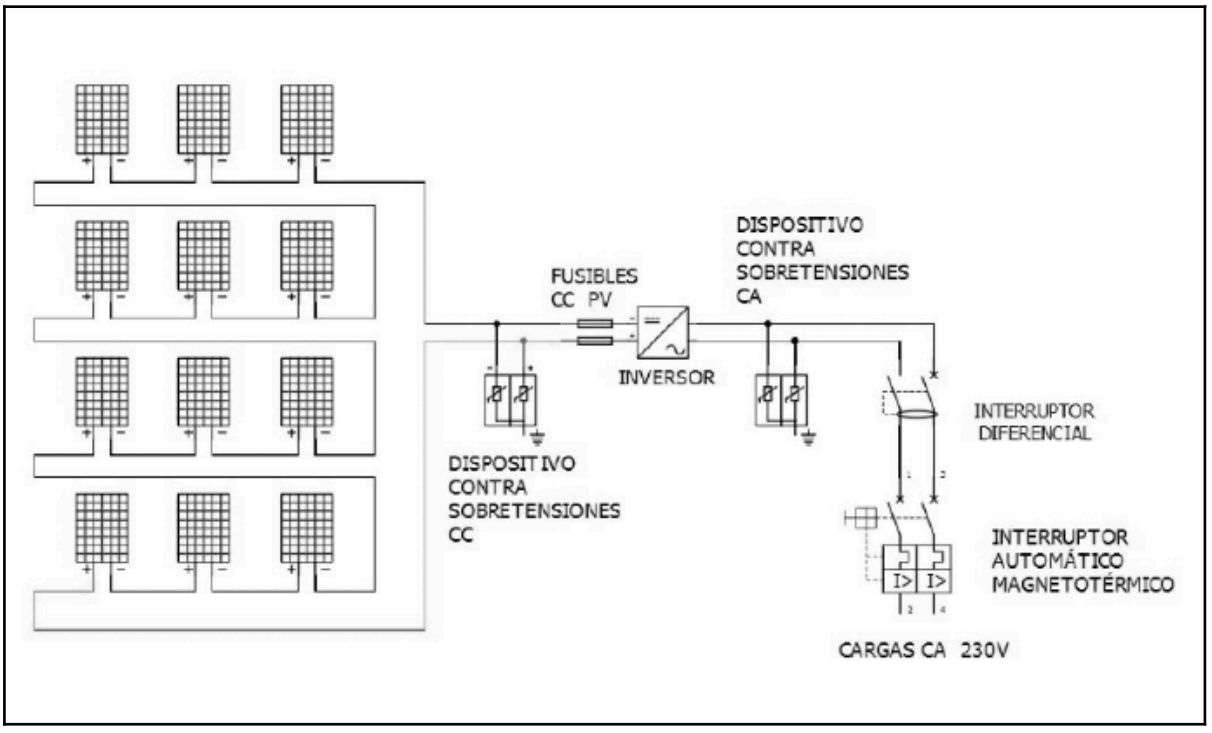


*Figura 2.7: Esquema de protección tipo de una instalación solar fotovoltaica conectada a red [9].*

### 2.1.3. Introducción a las Comunidades Energéticas

Las comunidades energéticas son agrupaciones de pymes y entidades legales del mismo entorno local que producen, distribuyen y consumen su propia energía renovable, consiguiendo así, mejores acuerdos con las compañías eléctricas. De este modo, tanto el ciudadano como el consumidor final tienen una implicación mayor en su comunidad y a su vez se benefician de las ventajas sociales y económicas que esto conlleva. Teniendo así, que las comunidades energéticas promueven:

- El impulso de un cambio en el modelo energético.
- Fomento de la energía limpia y la participación ciudadana.
- La comercialización de energía 100% renovable.

Cada comunidad energética local tiene una estructura diferente. Algunas están formadas por vecindarios y pequeñas comunidades y en otras ocasiones, intervienen entidades externas, como compañías eléctricas para favorecer el ahorro económico en su actividad. No obstante, todas tienen unas características comunes, que son:

- Deben ser una persona jurídica para la participación abierta y voluntaria.
- Ser controlada por socios y/o miembros.
- Realizar su actividad en exclusiva en ámbito local. Es decir, dentro de un o varios municipios limitados.
- Que los beneficios obtenidos se destinen al ahorro económico de los miembros de la comunidad energética.
- Que el nivel de producción no supere al consumo.
- Impedir la compra de producción en mercados especulativos.

La finalidad de las comunidades energéticas es crear un entorno sostenible, la participación ciudadana, promoviendo el empleo local y los beneficios económicos y sociales que tiene la inversión en sistemas renovables [6].

## 2.2. Monitorización de las instalaciones fotovoltaicas

La monitorización es crucial para detectar ineficiencias, predecir el mantenimiento, y tomar decisiones informadas en la gestión energética, siendo esencial para garantizar una gestión eficiente y optimizar la producción de energía.

Las variables de interés en este contexto abarcan medidas eléctricas de demanda, generación y almacenamiento, tal como se ilustra en la Figura 2.8.

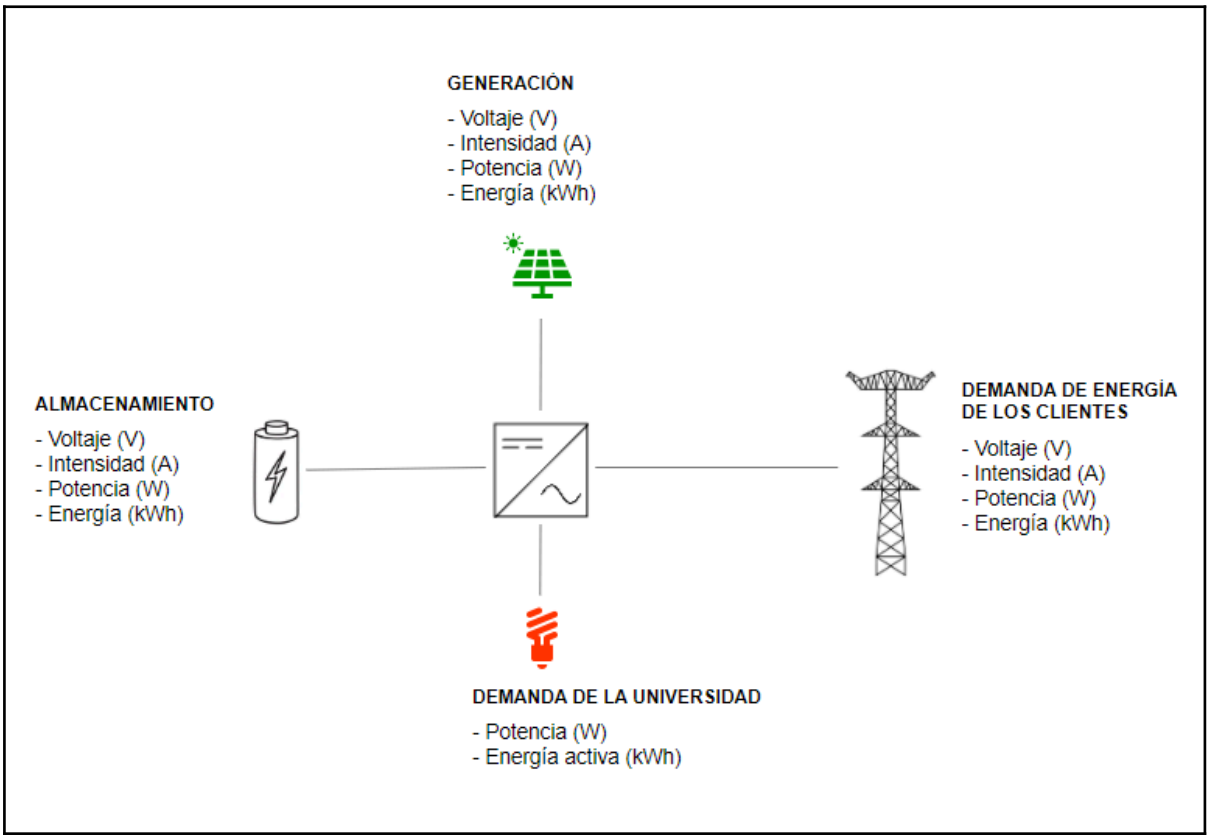


*Figura 2.8: Esquema de las áreas fundamentales de monitorización.*

Además, la monitorización efectiva de una instalación fotovoltaica no solo incluye el análisis de medidas eléctricas, sino que también abarca variables meteorológicas, tales como la temperatura y la irradiancia, así como la dirección y velocidad del viento, entre otras.

A continuación, se pasa a describir cada una de estas variables:

***1. Generación:***

- Voltaje (V): medición de la diferencia de potencial eléctrico producida por los paneles fotovoltaicos.
- Intensidad (A): corriente eléctrica generada por la instalación fotovoltaica.
- Potencia (W): tasa a la que la energía es generada por los paneles solares.
- Energía (kWh): total de energía generada durante un periodo determinado.

***2. Almacenamiento:***

- Voltaje (V): refleja la tensión en el sistema de almacenamiento, como las baterías.
- Intensidad (A): corriente que circula hacia o desde el sistema de almacenamiento.
- Potencia (W): capacidad del sistema de almacenamiento para liberar o absorber potencia.
- Energía (kWh): energía acumulada o entregada por el sistema de almacenamiento.

***3. Demanda de la Universidad:***

- Potencia (W): potencia instantánea consumida por el edificio.
- Energía activa (kWh): total de energía consumida por el edificio en un tiempo determinado.

***4. Demanda de Energía de los Clientes:***

- Se refiere a la energía consumida por las entidades o personas que utilizan la energía producida por la instalación fotovoltaica.

***5. Variables meteorológicas:***

- Temperatura: afecta la eficiencia operativa de los paneles fotovoltaicos. Los paneles tienen una temperatura óptima de funcionamiento y la eficiencia puede disminuir si las temperaturas son demasiado altas o bajas.
- Irradiancia: cantidad de energía solar que llega a una superficie por unidad de área, comúnmente expresada en kilovatios por metro cuadrado ($kW/m^2$). Es un indicador clave de cuánta energía podrían potencialmente generar los paneles solares.
- Dirección del Viento y Velocidad del Viento: estos factores influyen en el enfriamiento de los paneles solares y pueden afectar su rendimiento. Un enfriamiento adecuado puede mejorar la eficiencia de los paneles, mientras que vientos muy fuertes podrían potencialmente dañar la infraestructura sin las medidas de protección adecuadas.

Otros términos que deben ser comprendidos para plantear el sistema de monitorización de plantas fotovoltaicas son los siguientes:

- Consumo Eléctrico: cantidad de energía eléctrica que es consumida por los dispositivos y sistemas en un período de tiempo.
- Demanda: potencia eléctrica máxima requerida por los usuarios en un momento dado o durante un período específico.
- Generación: proceso de conversión de energía solar a energía eléctrica por los paneles fotovoltaicos.
- Almacenamiento: capacidad del sistema para almacenar energía generada y no utilizada inmediatamente, típicamente en forma de baterías u otros medios de almacenamiento energético.

### 2.2.1. Funcionamiento del sistema de monitorización

Los sistemas de monitorización de producción fotovoltaica operan a través del inversor donde la mayoría de compañías ofrecen un software propio para visualizar las principales variables de interés. También existen compañías que diseñan programas de

monitorización de energía solar a petición del cliente compatibles con la mayoría de inversores [10].

Para poder acceder a las demás variables eléctricas se suelen emplear *analizadores de red* que miden y analizan la calidad de la energía en la red eléctrica de la instalación. Estos dispositivos son capaces de detectar y registrar una amplia gama de parámetros eléctricos como la frecuencia de la red, los armónicos, el factor de potencia, y otros índices de calidad de energía que son esenciales para el diagnóstico de problemas y para la optimización del rendimiento de la planta.


*Figura 2.9: Analizador de red.*

La integración de un analizador de red suele requerir el uso de un *transformador de corriente toroidal,* también conocido como transformador de núcleo dividido. Se instala alrededor del conductor del circuito de alimentación y opera bajo el principio del electromagnetismo, donde la corriente que fluye a través del conductor induce una corriente en el transformador que puede ser medida de manera segura.

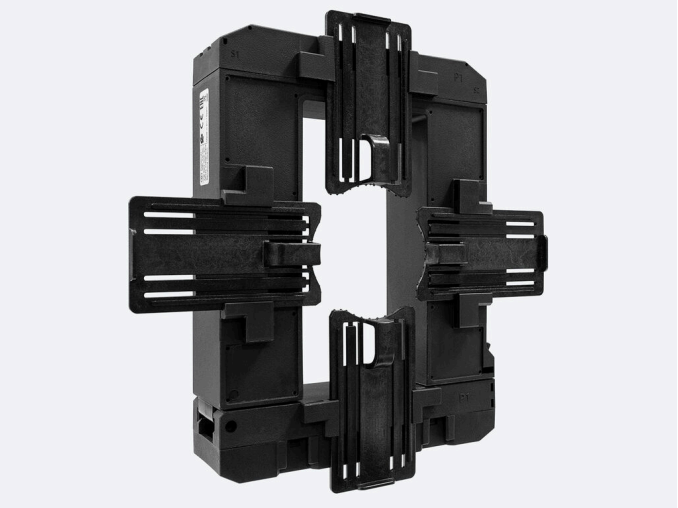

*Figura 2.10: Transformador de núcleo partido.*

En cuánto a las variables meteorológicas, existen diferentes *sensores de irradiación solar y de temperatura* que nos permiten conocer las condiciones ambientales que afectan directamente la eficiencia de los paneles fotovoltaicos.

El acceso a las variables de una instalación se realiza a través de protocolos de comunicación industrial estandarizados como ModBus, BACnet o SNMP. Estos facilitan la integración y el intercambio de datos entre los analizadores de red, los sensores ambientales y los sistemas de gestión de edificios (BMS) o plataformas de monitorización dedicadas.

Una vez que los datos son recogidos, se centralizan en una base de datos del BMS o de la plataforma de monitorización, donde los algoritmos de software procesan y correlaciona la información, permitiéndonos acceder a los datos visualizados a través de paneles de mandos interactivos, permitiendo ofrecer vistas en tiempo real, históricos de datos, gráficos de tendencias, y alertas automatizadas [11].

Además, la incorporación de tecnologías como el *Internet de las Cosas (IoT)* ha revolucionado los sistemas de monitorización al permitir que los sensores y dispositivos estén conectados a la red. Esto posibilita una comunicación en tiempo real y remota, mejorando así la capacidad de respuesta ante eventos inesperados y optimizando la gestión de las instalaciones.

Dicho esto, tenemos que un esquema de monitorización de una planta fotovoltaica de manera general se asemeja al que se muestra a continuación:

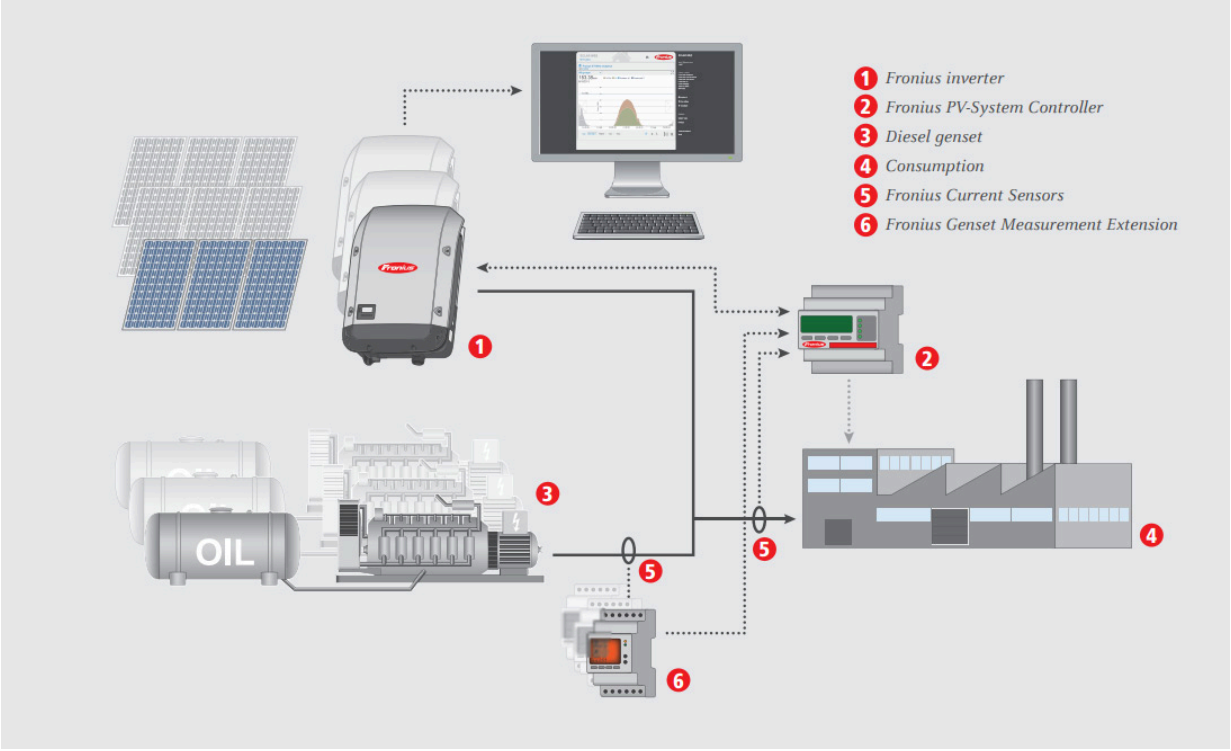


*Figura 2.11: Esquema de monitorización de una planta fotovoltaica [12].*

## 2.3. Tecnologías de recolección de datos: Web Scraping, RPA y APIs

La recolección de datos es un proceso fundamental que mejora la compresión del rendimiento de los sistemas. Esta fase es vital ya que proporciona la base de datos necesaria para el análisis y la toma de decisiones. Existen varias herramientas y librerías en diferentes entornos de programación diseñadas para recolectar datos de manera eficiente y automática, adecuándose a la complejidad y la escala de la infraestructura energética.

Este apartado explora las herramientas y técnicas que automatizan la extracción de datos, una práctica conocida como *Web Scraping*, y cómo se complementa con la *Automatización Robótica de Procesos (RPA)*. Ambas técnicas son fundamentales en la recopilación de datos sin la intervención manual, permitiendo una supervisión y análisis más ágiles y profundos. Además, se abordará el rol de las *Interfaces de Programación de Aplicaciones (APIs)* en la automatización de la recolección y envío de datos.

### 2.3.1. Web Scraping

El *Web Scraping* es una metodología de extracción de datos empleada para recabar información de sitios web. Esta técnica, que opera mediante el uso de scripts automatizados o bots, permite la obtención de datos directamente desde el código HTML de las páginas web, especialmente cuando la información deseada no está disponible a través de bases de datos estructuradas o APIs de fácil acceso.

En esencia, el web scraping es la práctica de visitar una URL para extraer datos útiles del HTML presentado. A menudo, estos datos son recogidos en un formato crudo y no estructurado, es decir, a diferencia de los datos que se presentan en formatos convencionales como CSV, TXT o Excel, los datos adquiridos mediante web scraping requieren un procesamiento posterior para transformarlos en un formato manejable y útil para análisis o aplicaciones específicas.

Aunque en algunas ocasiones se puede tener la ventaja de descargar archivos en formatos estándar como .csv directamente desde un sitio web, hay situaciones en las que es necesario identificar y seleccionar elementos específicos dentro de la estructura de una página web para recopilar la información requerida [13].

Para realizar web scraping, se suelen utilizar varias librerías especializadas. Aquí algunas de las más empleadas:

- *Requests:* permite enviar peticiones HTTP de manera sencilla a los servidores web y obtener el contenido de las páginas.
- *BeautifulSoup:* permite analizar el contenido HTML obtenido y extraer los datos necesarios.

- *lxml:* librería de parsing que puede manejar HTML y XML y es conocida por su velocidad y facilidad de uso.
- *Scrapy:* framework de web scraping de código abierto y colaborativo que proporciona herramientas para extraer, procesar y almacenar datos de sitios web.

### 2.3.2. Automatización Robótica de Procesos (RPA)

La *Automatización Robótica de Procesos* (RPA, por sus siglas en inglés) es una tecnología que permite automatizar tareas repetitivas y basadas en reglas en sistemas informáticos. Consiste en utilizar software o bots para imitar y ejecutar las acciones que normalmente realizaría un ser humano dentro de un proceso empresarial. Esto implica:

- Identificación de tareas repetitivas y basadas en reglas.
- Desarrollo de robots.
- Implementación y prueba.
- Monitoreo y mantenimiento.

El RPA busca aumentar la productividad y reducir los errores en tareas administrativas y de oficina, así como liberar a los empleados de actividades repetitivas para que puedan centrarse en tareas más estratégicas y de mayor valor. Siendo los beneficios los siguientes entre otros:

- Reducción de errores humanos.
- Aumento de la eficiencia operativa.
- Mejora de la precisión en el procesamiento de datos.
- Liberación de tiempo para que los empleados se enfoquen en tareas de mayor valor agregado.

Para realizar la Automatización Robótica de Procesos (RPA), existen varias herramientas y librerías. Las opciones más utilizadas son las siguientes:

- *UiPath:* plataforma de RPA líder en la industria que permite automatizar las tareas de escritorio y web. Proporciona una interfaz visual para arrastrar y soltar, lo que facilita la creación de flujos de trabajo de automatización sin necesidad de escribir código.
- *Power Automate:* herramienta de automatización de servicios ofrecida por Microsoft.
- *PyAutoGUI:* librería de Python que permite controlar el teclado y el ratón, facilitando la automatización de tareas en la interfaz de usuario.
- *Selenium:* originalmente diseñada para pruebas de aplicaciones web, Selenium también se suele utilizar para automatizar tareas en navegadores web. A través de Selenium WebDriver, se pueden simular interacciones del usuario con páginas web para la extracción de datos o automatización de tareas.

La información presentada se ha extraído de la referencia bibliográfica número [14], tal como se detalla en el apartado de bibliografía de este documento.

### 2.3.3. Interfaces de programación de aplicaciones (APIs)

Son conjuntos de protocolos, rutinas y herramientas para construir software y aplicaciones. Actúan como intermediarios que permiten la interacción entre diferentes programas de software de manera eficiente y segura. Las APIs nos proporcionan un método estandarizado para solicitar y recibir datos de una aplicación en particular.

Las APIs son especialmente valiosas porque permiten acceder a funcionalidades o datos de aplicaciones sin necesidad de entender el código fuente o la estructura interna del software en cuestión. Esto es particularmente útil para recopilar datos de servicios web, ya que las APIs ofrecen una forma de obtener información actualizada directamente desde la fuente, a menudo en formatos fáciles de manejar, como JSON o XML.

En términos de extracción de datos, las APIs pueden proporcionar acceso a datos en tiempo real y son fundamentales para aplicaciones que necesitan información actualizada continuamente, como las aplicaciones meteorológicas, de mercados financieros, o redes sociales. A diferencia del web scraping, que puede estar limitado por la necesidad de parsear HTML y que puede verse afectado por cambios en la estructura del sitio web, las APIs ofrecen una interfaz más estable y robusta para la obtención de datos [15].

## 2.4. Infraestructura como servicio (IaaS)

La *infraestructura como servicio (IaaS)* es un modelo de servicio en la nube que ofrece recursos de infraestructura bajo demanda, como computación, almacenamiento, redes y virtualización, a empresas y particulares a través de la nube [16].

La infraestructura nos ofrece un servidor en la nube donde se ejecutarán códigos de manera continua, sin la necesidad de mantener operativo un servidor físico local o un ordenador personal. Las ventajas de este servidor son las siguientes:

- *Disponibilidad Continua:* los servidores en la nube pueden operar de manera ininterrumpida, lo que significa que los códigos o bots de automatización pueden funcionar 24/7 sin la intervención humana.
- *Acceso Remoto:* los servidores en la nube se pueden gestionar y monitorear remotamente.
- *Rendimiento y Seguridad:* se invierte significativamente en seguridad y mantenimiento, asegurando que la infraestructura esté protegida contra amenazas y funcione óptimamente.

## 2.5. Internet de las cosas (IoT)

El *Internet de las cosas (IoT)* juega un papel fundamental en la transformación del sector energético, impulsando la eficiencia, la sostenibilidad y la automatización. Su capacidad para conectar dispositivos y sensores a internet permite la recopilación y análisis de datos en tiempo real, posibilitando una gestión energética más inteligente y eficiente.

El IoT, o Internet de las Cosas, no tiene una definición única y universalmente aceptada. Sin embargo, podemos entenderlo mejor al analizar sus dos componentes: Internet y Cosas.

- *Internet:* una red global de redes interconectadas que permite el intercambio de información entre miles de millones de usuarios en todo el mundo.
- *Cosas:* cualquier objeto físico del mundo real, desde dispositivos electrónicos y productos tecnológicos hasta muebles, ropa, cubos de basura, edificios, plantas, etc. No limitándose únicamente a objetos digitales.

Combinando estas definiciones, podemos definir el IoT como que es "Una red abierta y completa de objetos inteligentes que tienen la capacidad de relacionarse entre sí, comunicarse las informaciones, datos y recursos capturados de las situaciones y cambios en el entorno". [17]

En otras palabras, el IoT da vida a los objetos cotidianos al conectarlos a Internet. Esto permite que los objetos recopilen datos, se comuniquen entre sí y respondan a su entorno de forma inteligente, es decir, gracias a esta tecnología podemos conectar cualquier objeto que usamos día a día a Internet permitiéndonos, por ejemplo, automatizar procesos, tener al alcance de nuestra mano el control de dichos objetos y monitorizar los entornos a través de los continuos datos obtenidos de los mismos.

El IoT está en constante evolución y su potencial es ilimitado. A medida que la tecnología avanza y los costes se reducen, el IoT se integrará cada vez más en nuestra vida diaria, transformando la forma en que vivimos, trabajamos y nos relacionamos con el mundo que nos rodea. Para hacernos una idea, en 2019 había 10.000 millones de dispositivos conectados al IoT. Para 2025, las predicciones estiman que habrá cerca de 40.000 millones.

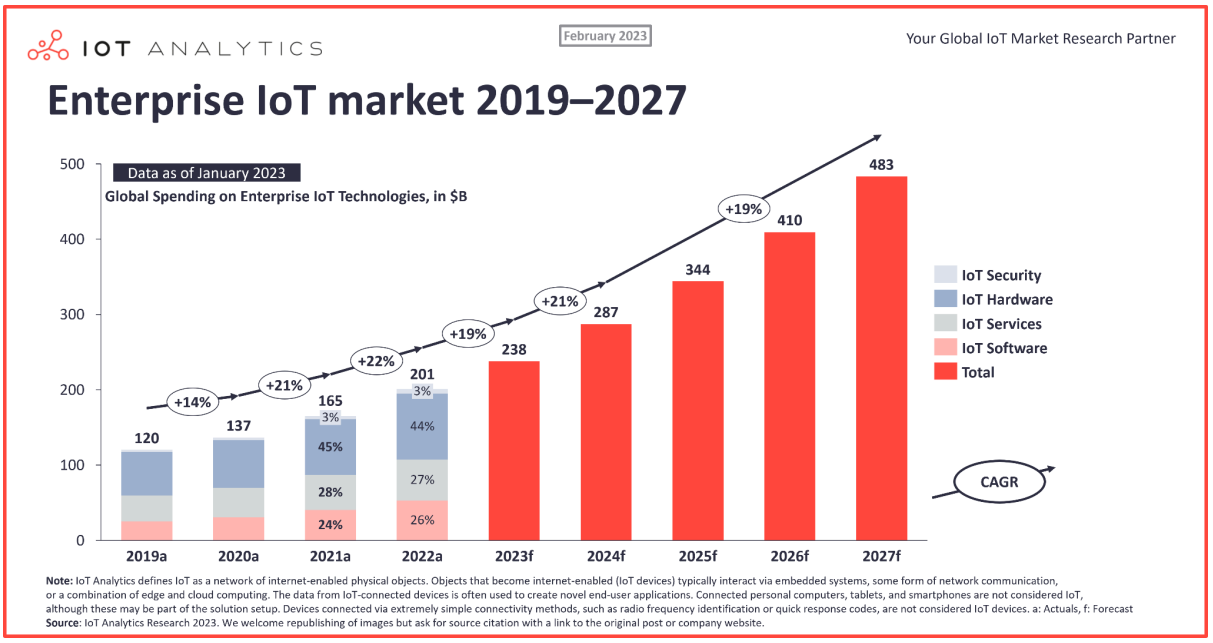


*Figura 2.12: Tendencia de crecimiento de dispositivos IoT [18].*

### 2.5.1. Funcionamiento del internet de las cosas

El Internet de las Cosas (IoT) funciona gracias a la interacción de cuatro pilares fundamentales: tecnología de sensores, conectividad, procesamiento de datos e interfaz de usuario.

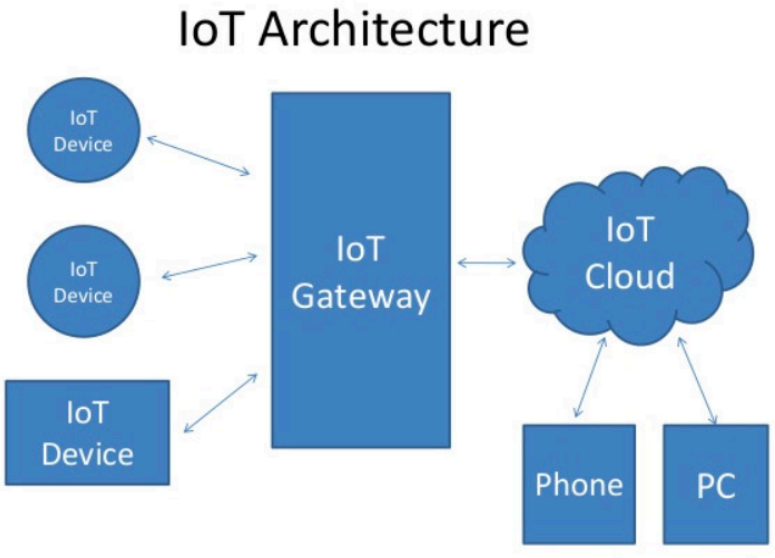


*Figura 2.13: Modelo de arquitectura del IoT [19].*

#### 2.5.1.1. Tecnología de sensores

Los sensores permiten implementar la interfaz entre el entorno y los dispositivos de gestión digital del IoT. Permiten recopilar datos del entorno, desde información simple como la temperatura o el movimiento hasta datos más complejos como imágenes o audio.

#### 2.5.1.2. Conectividad de los dispositivos

Los datos recopilados por los sensores necesitan ser transmitidos a un centro de procesamiento. Las opciones de conectividad más comunes incluyen WiFi, Bluetooth, satélite, ethernet, etc.

#### 2.5.1.3. Procesamiento de datos

En la fase de procesamiento de datos, la información recopilada por los sensores se convierte en algo útil. Las tareas de procesamiento de datos pueden incluir:

- *Recogida y almacenamiento*: los datos se recopilan de los sensores y se almacenan en una base de datos o en la nube.
- *Limpieza y análisis:* los datos se limpian para eliminar errores y luego se analizan para extraer información útil.
- *Visualización:* los datos se presentan de una manera fácil de entender para los usuarios, a través de gráficos, tablas o paneles de mando.
- *Toma de decisiones*: los datos se pueden utilizar para tomar decisiones automáticas o para proporcionar información a los usuarios para que puedan tomar decisiones informadas.

#### 2.5.1.4. Interfaz de usuario

La interfaz de usuario es el puente entre los datos y el usuario. Permite a los usuarios interactuar con el sistema IoT, ver los datos recopilados y controlar los dispositivos conectados.

La información presentada se ha extraído de la referencia bibliográfica número [20], tal como se detalla en el apartado de bibliografía de este documento.

### 2.5.2. Plataformas IoT

Una plataforma del internet de las cosas es la infraestructura y las aplicaciones necesarias para reunir, almacenar, analizar y ofrecer inteligencia a partir de los datos generados por los dispositivos conectados a Internet. Esta plataforma unificada, a menudo alojada en una nube pública, centraliza la recopilación de los datos y permite un intercambio sencillo de información entre aplicaciones [21]. Las plataformas IoT permiten a los usuarios

finales tomar decisiones informadas basadas en datos en tiempo real y facilitan la automatización de tareas y procesos.

Existen numerosas plataformas IoT, cada una diseñada para satisfacer diferentes necesidades. Entre estas, podemos encontrar soluciones bien establecidas y reconocidas como *ThingsBoard*, ThingSpeak, Grafana, así como opciones ofrecidas por gigantes tecnológicos como AWS IoT Core, Microsoft Azure IoT Hub, Google Cloud IoT Core e IBM Watson IoT.

Para llevar a cabo este proyecto, se ha seleccionado *ThingsBoard* como nuestra plataforma principal. Esta elección se basa en la compatibilidad que posee *ThingsBoard* con los objetivos y requisitos del proyecto. Otro factor decisivo en nuestra selección ha sido la disponibilidad de la versión de código abierto que *ThingsBoard* dispone.

## 2.6. Plataforma IoT ThingsBoard

*ThingsBoard* es una plataforma de código abierto para la Internet de las Cosas (IoT), que proporciona una amplia gama de funcionalidades para la gestión, monitorización y análisis de datos recogidos de dispositivos IoT. Entre sus funcionalidades destacan las siguientes:

- *Recopilación y Gestión de Datos:* ThingsBoard permite la recopilación de datos de una variedad de sensores y dispositivos.
- *Integración de Dispositivos:* la plataforma es compatible con una amplia gama de protocolos de comunicación de IoT, como MQTT, CoAP y HTTP, lo cual es esencial para conectar con diferentes tipos de sensores y dispositivos.
- *Procesamiento y Almacenamiento de Datos:* ThingsBoard proporciona herramientas para el procesamiento de datos en tiempo real, permitiendo filtrar, agregar y transformar los datos según sea necesario. Además, ofrece opciones de almacenamiento de datos que facilitan la consulta y el análisis de grandes volúmenes de información.
- *Visualización de Datos:* cuenta con un potente sistema de dashboards personalizables, donde puedes visualizar los datos recopilados en formatos comprensibles como gráficos, tablas y mapas.
- *Análisis y Reportes:* permite realizar análisis de datos, lo cual es crucial para optimizar el rendimiento de una planta fotovoltaica.
- *Alertas y Automatización:* la plataforma permite configurar alarmas y alertas basadas en ciertos umbrales o eventos, lo cual es vital para la detección temprana de problemas o fallos en el sistema. Ofrece también opciones de automatización para controlar dispositivos o ejecutar ciertas acciones basadas en los datos o eventos.
- *Seguridad y Escalabilidad:* ofrece características de seguridad robustas para proteger los datos.

La información presentada se ha extraído de la referencia bibliográfica número [22], tal como se detalla en el apartado de bibliografía de este documento.

### 2.6.1. Arquitectura de ThingsBoard

La arquitectura de ThingsBoard se basa en una serie de componentes que trabajan juntos para proporcionar una plataforma escalable y segura para la gestión de dispositivos IoT. Los componentes principales de la arquitectura son:

- *ThingsBoard Core*: es el núcleo de la plataforma y es responsable de la gestión de dispositivos, el procesamiento de datos y la configuración de reglas.
- *ThingsBoard Transports:* son responsables de la comunicación con los dispositivos IoT. Admiten una amplia gama de protocolos de comunicación, como MQTT, CoAP y HTTP entre otros.
- *Rule Engine (Motor de reglas):* permite procesar en tiempo real los datos recogidos de los dispositivos IoT.
- *Interfaz de Usuario (UI) de ThingsBoard:* actúa como el punto de interacción principal para los usuarios con la plataforma, permitiéndonos interactuar con la plataforma e implementar distintas funcionalidades de gestión de dispositivos como:
    - Visualización de datos: permite visualizar los datos recogidos de los dispositivos IoT en diferentes formatos, como gráficos, tablas y mapas.
    - Gestión de dispositivos: permite gestionar los dispositivos IoT conectados a la plataforma, incluyendo su configuración, agrupación y visualización de su estado.
    - Creación de dashboards: se pueden crear tableros personalizados (dashboards) que muestran la información más relevante para el usuario.
    - Configuración de alertas: permite configurar alertas para notificar a los usuarios sobre eventos específicos, como el fallo de un dispositivo o la superación de un umbral predefinido.
    - Ejecución de acciones: se pueden ejecutar acciones sobre los dispositivos IoT, como encender o apagar un dispositivo.
- *External Systems:* permiten la integración de ThingsBoard con otros sistemas, como sistemas de gestión de edificios.
- *Message Queue (Cola de mensajes):* se utiliza para desacoplar el procesamiento de datos de la recepción de datos de los dispositivos IoT. Esto mejora la escalabilidad y el rendimiento de la plataforma. ThingsBoard soporta múltiples implementaciones de colas de mensajes como Kafka, RabbitMQ, etc.
- *Base de Datos:* almacena los datos recogidos de las entidades (dispositivos, activos clientes, paneles, etc.) y datos de telemetría. La plataforma admite tres opciones de

bases de datos SQL, NoSQL e Híbrido, pudiendo ser PostgreSQL con Cassandra o PostgreSQL con TimescaleDB. A continuación, se describe que almacena cada una de estas bases de datos:

- SQL: almacena todas las entidades y las telemetrías de datos estructurados.
- NoSQL: almacena todas las entidades y la telemetría de datos no estructurados o semiestructurados. Actualmente esta opción está obsoleta en favor de emplear el enfoque híbrido debido a las limitaciones de NoSQL que dispone con entidades de IoT.
- Híbrido (PostgreSQL + Cassandra): almacena todas las entidades en la base de datos PostgreSQL y datos de series temporales en la base de datos Cassandra.
- Híbrido (PostgreSQL + TimescaleDB): almacena todas las entidades en la base de datos PostgreSQL y los datos de series temporales en la base de datos Timescale.

La Figura 2.14 presenta los componentes esenciales del sistema y las interfaces que ofrecen, donde los distintos elementos de la arquitectura se ilustran mediante bloques y las interconexiones entre ellos se señalan con flechas.

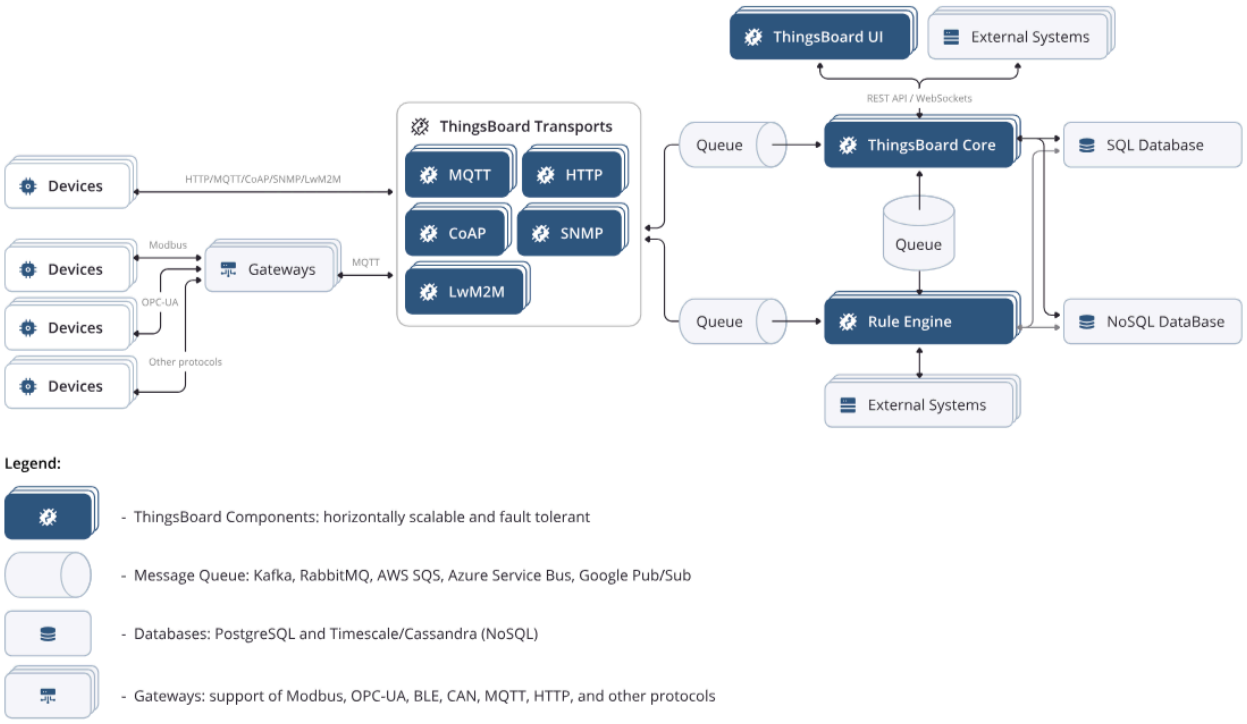


*Figura 2.14: Diagrama de la arquitectura que emplea ThingsBoard [23].*

La información presentada se ha extraído de la referencia bibliográfica número [23], tal como se detalla en el apartado de bibliografía de este documento.

## 2.7. Introducción a las Redes Neuronales Artificiales

Las redes neuronales artificiales (RNA) son un tipo de modelo computacional dentro del ámbito de la inteligencia artificial (IA) y el aprendizaje automático. Su objetivo principal es procesar información de manera similar a como lo hace el cerebro humano, permitiendo a las máquinas aprender y adaptarse a partir de datos. Para ello, las RNA se basan en la estructura y el funcionamiento de las neuronas biológicas, las unidades básicas del sistema nervioso. Cada neurona artificial, al igual que las biológicas, recibe entradas de otras neuronas, las procesa y genera una salida. Estas conexiones entre neuronas, denominadas sinapsis, se representan en las RNA como pesos, los cuales determinan la fuerza de la influencia de una neurona sobre otra.

Las RNA se componen de tres elementos fundamentales:

- *Neuronas:* unidades básicas de procesamiento de información. Cada neurona recibe entradas, las procesa mediante una función de activación y genera una salida.
- *Capas:* las neuronas se organizan en capas, donde cada capa recibe información de la capa anterior y la procesa antes de enviarla a la siguiente capa. Existen diferentes tipos de capas, como las de entrada, ocultas y salida.
- *Conexiones:* las neuronas se conectan entre sí mediante sinapsis artificiales, representadas por pesos. Estos pesos determinan la fuerza de la influencia de una neurona sobre otra y se ajustan durante el proceso de aprendizaje [25].

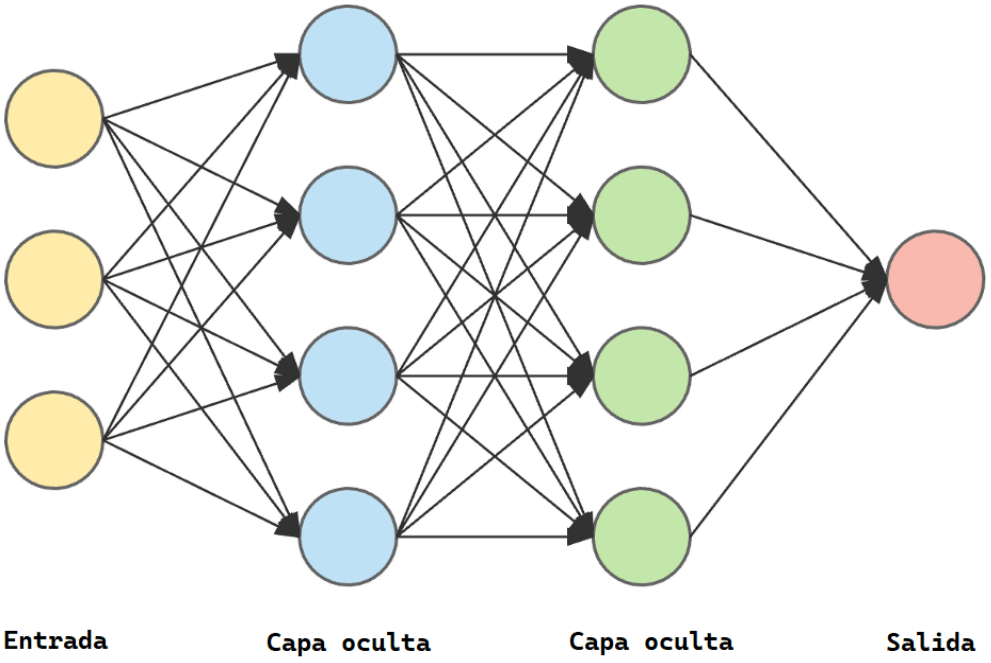


*Figura 2.15: Red neuronal artificial.*

## 2.8. Redes Neuronales Recurrentes (RNN)

Las redes neuronales recurrentes (RNNs) son un tipo de red neuronal artificial con una arquitectura única que les permite procesar y analizar secuencias de datos. A diferencia de las redes neuronales feedforward estándar, que sólo consideran la entrada actual, las RNNs pueden almacenar información de entradas pasadas y utilizarla para realizar predicciones o tomar decisiones [26]. Esta característica las convierte en herramientas muy potentes para una amplia gama de tareas, incluyendo el procesamiento del lenguaje natural, el reconocimiento de voz y la predicción de series temporales.

Las RNNs se basan en la idea de que la información actual no solo depende de la entrada actual, sino también de las entradas y salidas pasadas (Figura 2.16). Para ello, las RNNs incorporan un bucle en su arquitectura que les permite "recordar" información de pasos de tiempo previos (Figura 2.17). Este bucle se implementa mediante una unidad de memoria interna, también conocida como célula de memoria. La célula de memoria almacena información relevante del pasado y la utiliza para procesar la entrada actual y generar la salida.

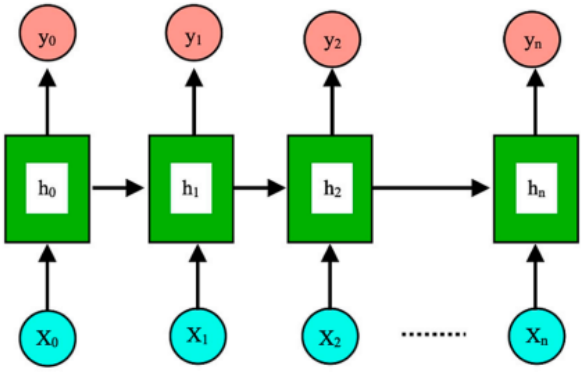


*Figura 2.16: Arquitectura de una RNN en estado desplegado [26].*

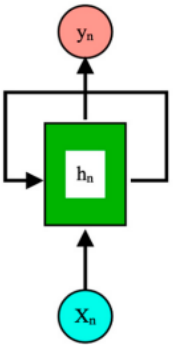


*Figura 2.17: Arquitectura de una RNN en estado plegado [26].*

## 2.9. Redes Neuronales Long - Short Term Memory (LSTM)

Las redes neuronales recurrentes (RNNs) básicas poseen una arquitectura que les permite procesar secuencias de datos, tomando en cuenta información de pasos de tiempo previos. Sin embargo, estas redes presentan una limitación importante: la longitud de la secuencia procesada debe ser relativamente corta para que las activaciones anteriores, es decir, la memoria de la red, tengan un efecto relevante en la predicción actual. Esto es debido al problema de desvanecimiento de gradiente (*vanishing*), lo que dificulta que las redes aprendan dependencias a largo plazo en los datos.

El desvanecimiento de gradiente es un problema que surge en el entrenamiento de redes neuronales, donde las actualizaciones de los pesos de la red se vuelven cada vez más pequeñas a medida que se retropropaga el error a través del tiempo. En el caso de las RNNs básicas, este problema se intensifica debido a la naturaleza recurrente de la red, lo que limita su capacidad para aprender dependencias a largo plazo [27]. Para superar esta limitación, se han desarrollado variantes de las RNNs que incorporan mecanismos para controlar el flujo de información a través de la memoria interna de la red. Una de las variantes más populares son las redes neuronales Long - Short Term Memory (LSTM).

Las LSTM permiten a las RNNs recordar sus entradas durante un largo período de tiempo. Esto se debe a que LSTM contiene su información en la memoria. Una neurona de una LSTM puede leer, escribir y borrar información de su memoria [28]. Esta capacidad se logra mediante el uso de celdas de memoria y tres compuertas específicas: la compuerta de entrada, la de olvido y la de salida. Estas compuertas regulan el flujo de información que entra y sale de la celda de memoria, permitiendo que la red decida qué datos son importantes para retener o descartar a lo largo del tiempo. En una red LSTM, la compuerta de olvido decide qué información de la celda de memoria se debe descartar. Simultáneamente, la compuerta de entrada actual actualiza el estado de la celda con información nueva. Por último, la compuerta de salida determina qué parte del estado de la celda se transfiere a la salida de la red [29].

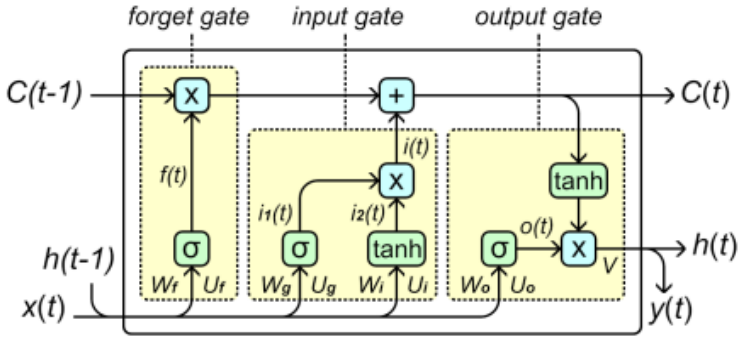


*Figura 2.18: Arquitectura de una red LSTM [30].*

El proceso de actualización de la memoria en las LSTM implica la integración de la información pasada (retenida por la compuerta de olvido) y la nueva información (a través de

la compuerta de entrada), lo cual se ajusta en cada paso del tiempo según la relevancia detectada por la red para la tarea específica que realiza.

# 3. Metodología

En este capítulo se describirán las distintas metodologías empleadas durante el desarrollo del proyecto. Se incluye una descripción general de la estructura del sistema de monitorización en la que nos basaremos. Acto seguido, se exponen las técnicas y el desarrollo empleado para extraer los datos y su proceso de incorporación y sincronización con la plataforma ThingsBoard. Además, se incluye el desarrollo de cómo se realizaron los diferentes cuadros de mando para visualizar la información. Por último, se expone la metodología llevada a cabo para desarrollar un modelo de predicción basado en técnicas de aprendizaje automático.

## 3.1. Estructura del sistema de monitorización

En este proyecto, se plantea un sistema de monitorización donde los edificios no disponen aún de analizadores de red o no se tiene acceso a ellos, por tanto, la información para la monitorización se obtendrá a través de la plataforma *GoodWe*, fabricante de inversores fotovoltaicos, la cual proporciona datos de la energía fotovoltaica generada por cada una de las plantas y el consumo de energía de cada edificio..

Además, cabe destacar que tal y como se mostró en apartados anteriores los edificios disponen de una instalación fotovoltaica conectada a la red en la modalidad de suministro con autoconsumo sin excedentes, por lo que, están provistos de un mecanismo antivertido que impide la inyección de energía excedentaria a la red de transporte o de distribución. Esto se debe a que la normativa actual no permite verter excedentes a la red sin la correspondiente autorización y contrato de suministro, siendo el proceso de obtención de esta autorización bastante elevado, lo que hizo que inicialmente la Universidad de La Laguna prefiriera acogerse a esta modalidad.

Debido al mecanismo antivertido, la energía fotovoltaica generada se verá limitada por el consumo del edificio en cada momento. Esto significa que no se podrá conocer la generación real de la instalación, es decir, la plataforma GoodWe solo mostrará la energía generada que ha sido consumida por el edificio, no la energía total producida por los paneles.

Dicho esto, la estructura de monitorización de cada una de las plantas se basará en el esquema que se muestra en la Figura 3.1. Inicialmente se extraen los datos de cada una de las plantas de la plataforma Sems Portal de Goodwe mediante tecnologías de recolección de datos que se expondrán en detalles en apartados posteriores. Estos datos se enviarán a dispositivos virtuales creados dentro del Software ThingsBoard. Internamente la plataforma IoT tratará estos datos gracias a cada uno de los componentes principales que dispone, almacenando las telemetrías de cada uno de los dispositivos en bases de datos NoSQL (Cassandra), mientras que las entidades de cada uno de estos se guardan en bases de datos SQL (PostgreSQL).

Posteriormente, se procesan y visualizan en cuadros de mandos que muestran la información más relevante de cada planta, como la generación de energía, el consumo y las alarmas.

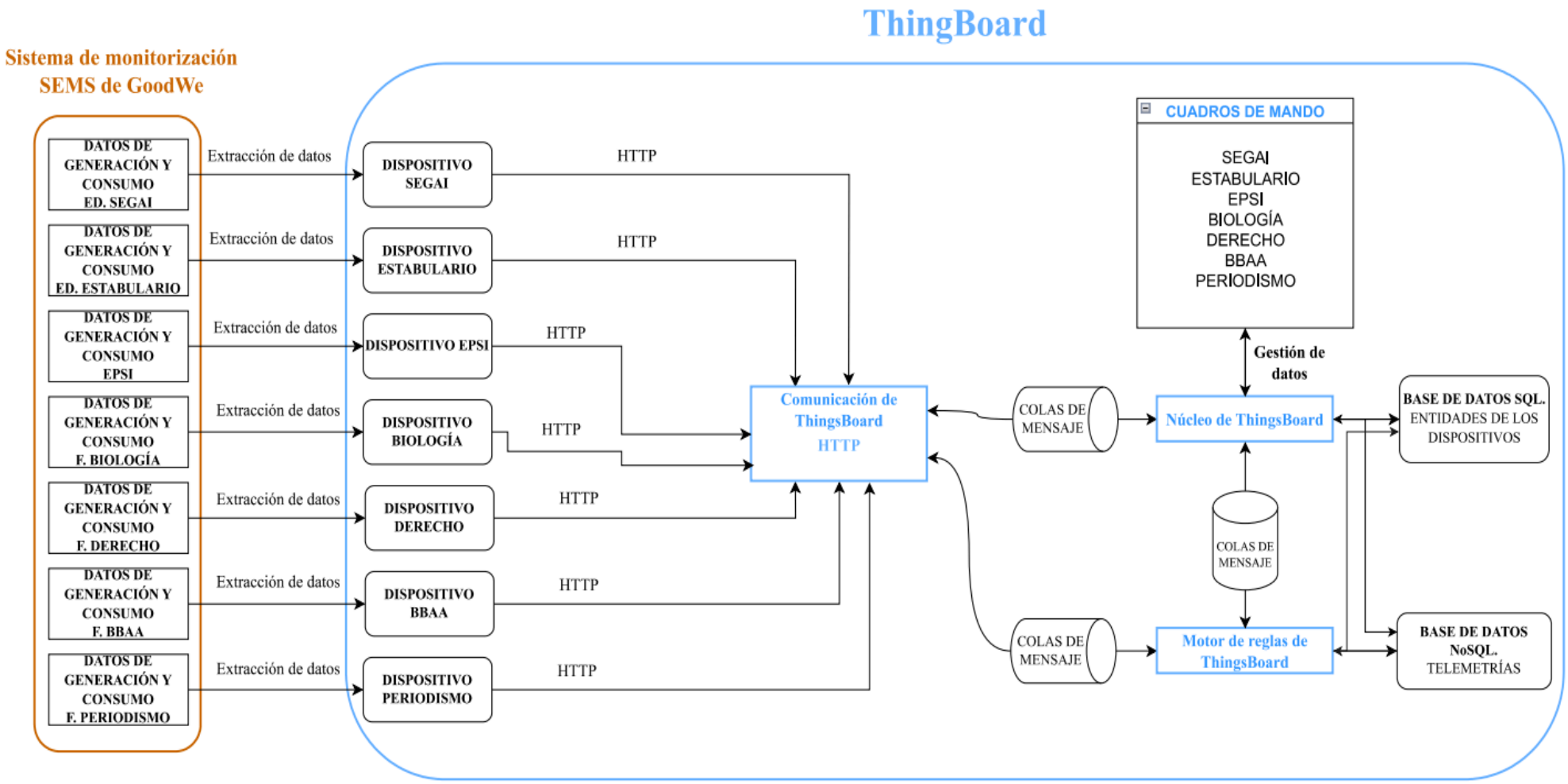


*Figura 3.1: Estructura del sistema de monitorización.*

## 3.2. Detalle del proceso de recolección y análisis de datos

La recolección de datos se realizará mediante las herramientas expuestas en el marco teórico. Para ello, se utilizará el lenguaje de programación *Python*. La finalidad principal es comprender y obtener los datos de la aplicación que ofrece Goodwe para monitorizar cada planta. El proceso se dividirá en tres fases principales:

1. *Extracción de datos del día actual para un edificio:* se seleccionará un edificio como y se procederá a la extracción de datos del día actual de la aplicación Goodwe.
2. *Extracción de datos históricos de cada edificio:* se obtendrán los datos históricos de cada uno de los edificios.
3. *Extracción de datos del día actual para todos los edificios:* una vez desarrollado y validado el script para la extracción de datos del día actual en un edificio, se desarrollará un único código que automatice la extracción de datos para todos los edificios, obteniendo así, la información completa de las siete plantas.

### 3.2.1. Herramienta de extracción de datos: Selenium

Antes de comenzar con la extracción de datos, se realizó un análisis de las diferentes herramientas disponibles para este fin en base a criterios como la facilidad de uso, la eficiencia y la compatibilidad que esta dispone con la aplicación de la cual vamos a extraer los datos.

Dado que se debe emplear herramientas que simulan interacciones humanas con la interfaz web como iniciar sesión en un portal con credenciales, navegar por la página, y finalmente descargar un archivo de datos, hablaremos de *RPA*. Por tanto, para realizar estas acciones emplearemos *Selenium*, herramienta de código abierto que nos permitirá simular interacciones con la página web para extraer los datos.

Para utilizar Selenium, es fundamental comprender el código *HTML* de la página web. La herramienta “Inspeccionar” del navegador permite visualizar y analizar dicho código. Esta funcionalidad nos permite ver el código HTML de la página, así como los estilos CSS aplicados y otros elementos JavaScript que puedan estar influyendo en la interacción con la web.

El código HTML es la base de cualquier página web. Al comprenderlo, se pueden identificar los elementos específicos con los que interactuamos con Selenium como:

- *Etiquetas:* definen la estructura y el contenido de la página (párrafos, imágenes, formularios, etc.).
- *Atributos:* proporcionan información adicional sobre las etiquetas (id, clase, nombre, etc.).
- *Localizadores:* permiten seleccionar de forma precisa elementos específicos en el código HTML (XPath, CSS Selectors, etc.) [24].

A continuación, se describen los comandos principales que emplearemos en las diferentes fases del proyecto:

1. *driver.get(url):*

   Este comando le indica a Selenium que abra una nueva ventana del navegador y navegue a la URL especificada. Es el primer paso para interactuar con la página web, permitiendo al script acceder a la misma para comenzar el proceso de automatización.

2. *WebDriverWait y EC.element_to_be_clickable:*

   WebDriverWait junto con Expected Conditions (como element_to_be_clickable) se utilizan para esperar de manera inteligente a que ciertos elementos de la página estén en un estado específico antes de realizar acciones sobre ellos. Esto se emplea para asegurar que las interacciones, como hacer clic o enviar texto, se realicen sólo cuando los elementos están completamente cargados y accesibles, previniendo errores por intentar interactuar con elementos aún no disponibles.

3. *find_element(By.locator, "value"):*

   Este método se emplea para localizar elementos dentro de la página web utilizando diferentes selectores. Dependiendo de la necesidad específica y la estructura de la página web, se pueden emplear selectores como ID, XPATH, CSS Selector, Name, y otros. Es fundamental para identificar precisamente los elementos con los que queremos interactuar, tales como campos de texto para introducir las credenciales de inicio de sesión o botones para enviar el formulario.

4. *send_keys(text):*

   send_keys() se utiliza para simular la entrada de texto en un campo de formulario. En el contexto de automatización con Selenium, permite ingresar datos como nombres de usuario, contraseñas, o cualquier otro texto requerido por el formulario web.

5. *click():*

   El método click() simula el clic sobre un elemento, como puede ser un enlace, un botón de formulario o cualquier otro elemento interactivo. Este comando es esencial para navegar a través de las páginas o enviar formularios después de completar los campos requeridos.

6. *ActionChains(driver).move_to_element(element).click().perform():*

   ActionChains permite automatizar secuencias complejas de acciones de teclado y ratón. En este caso, se utiliza para mover el cursor sobre un elemento específico y luego hacer clic en él. Es especialmente útil para interactuar con elementos que requieren acciones más sofisticadas que un simple clic.

7. *driver.quit():*

   Este comando cierra la ventana del navegador y termina la sesión del navegador.

Tras haber detallado los diversos comandos y herramientas esenciales para la automatización de nuestro proceso de recopilación de datos, se deberán seguir los siguientes pasos generales dentro de la aplicación Goodwe:

1. Iniciar sesión en la aplicación Goodwe con la cuenta correspondiente.
2. Seleccionar el edificio correspondiente en la aplicación.
3. Identificar los datos específicos que se desean extraer.
4. Extraer los datos seleccionados.
5. Almacenar los datos extraídos en un formato adecuado para su posterior análisis.

La información presentada se ha extraído de la referencia bibliográfica número [24], tal como se detalla en el apartado de bibliografía de este documento.

### 3.2.2. Extracción de datos para un único edificio

Para la prueba piloto se seleccionó el edificio SEGAI, ya que es un buen representante de los demás edificios que se analizarán en el proyecto. El código correspondiente a la extracción de datos de este edificio se encuentra en el archivo *SEGAI_IAAS.ipynb* (para más detalles ver Apéndice A.1).

Inicialmente, se importan las librerías necesarias para la extracción de datos. Acto seguido, se configura el navegador web utilizando Chrome y se establecen las preferencias de descargas, pudiendo iniciar así, con el proceso de automatización tal y como se muestra en la Figura 3.2.

```python
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.action_chains import ActionChains
from webdriver_manager.chrome import ChromeDriverManager
import time # Para trabajar con fechas y tiempo
import os # Librería que nos permite interactuar con el sistema de archivos
import glob # para buscar archivos en un directorio
import pandas as pd
import requests # Para realizar solicitud HTTP con ThingsBoard
import json # Formato al que hay que enviar los datos a la platafor IoT ThinsBoard

# Carpeta de donde se guardará el archivo con los datos
download_folder = "C:\\Users\\angel\\Desktop\\MASTER\\TFM"

options = webdriver.ChromeOptions()  # Opciones de navegación con selenium

# Configuraciones de descarga en modo headless
options.add_argument("--headless")
options.add_argument("--disable-gpu")
options.add_argument("--window-size=1920x1080")

prefs = {
    "download.default_directory": download_folder,
    "download.prompt_for_download": False,
    "download.directory_upgrade": True,
    "safebrowsing.enabled": True,
    "safebrowsing.disable_download_protection": True,
    "profile.default_content_setting_values.automatic_downloads": 1, # Permitir múltiples descargas
}

options.add_experimental_option("prefs", prefs)

# Para asegurarse de que no se bloqueen las descargas en modo headless
options.add_argument("--no-sandbox")
options.add_argument("--disable-dev-shm-usage")

# Inicializar el driver con las opciones configuradas
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=options)
```

*Figura 3.2: Fragmento de código inicial.*

**Paso 1: Inicio de sesión**

Primero, se debe iniciar sesión en la aplicación Goodwe utilizando las credenciales del usuario. Además, otro aspecto importante es que tal y como se observa en la Figura 3.3 existe una casilla de condiciones de uso y política de privacidad que hay que rellenar si queremos poder acceder a la web.

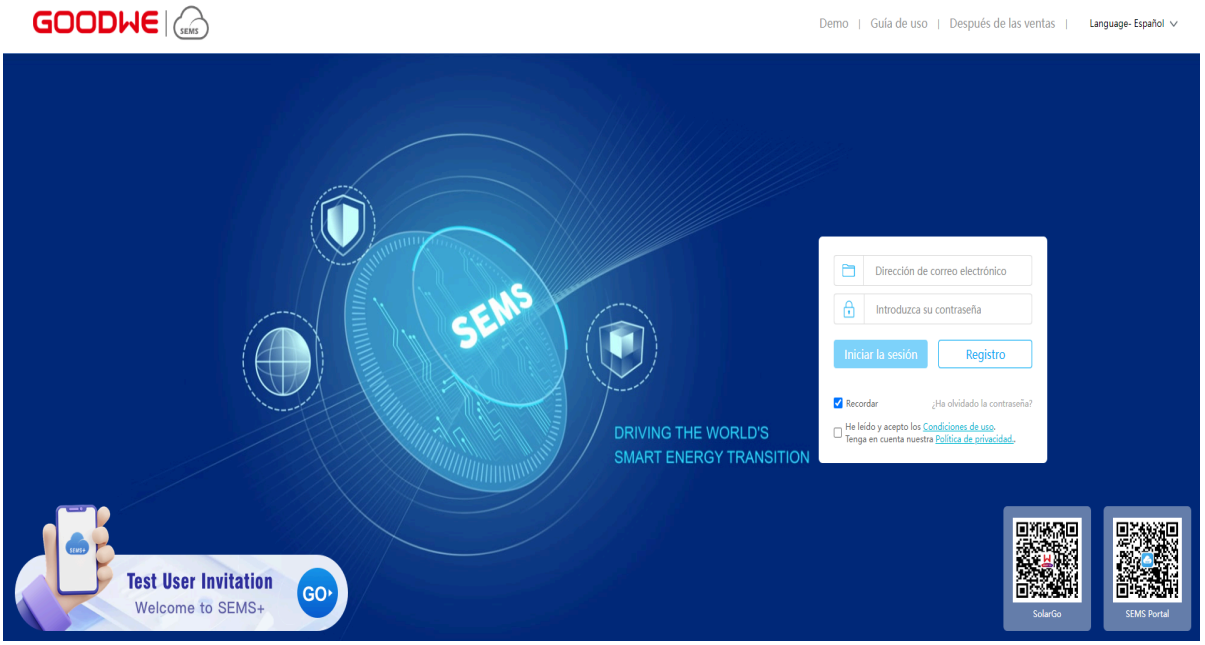


*Figura 3.3: Navegador mostrando la página de inicio de sesión.*

Para realizar esta acción, primero se define la URL de la página de inicio de sesión y se utiliza el método get() del driver de Selenium para abrirla en el navegador.

Acto seguido, se utiliza la clase WebDriverWait para esperar hasta que la casilla de verificación de las condiciones de uso y política de privacidad sea visible y clickable. Si la casilla no está seleccionada, se hace clic en ella.

Posteriormente, se localizan los campos de entrada de correo electrónico y contraseña, así como el botón de inicio de sesión, mediante sus IDs. Se definen las variables correspondientes con las credenciales del usuario y se introducen en los campos adecuados. Finalmente, se hace clic en el botón de inicio de sesión y se espera unos segundos para que la página principal de Sems Portal cargue completamente.

```python
# Inicio de sesión y descarga de datos de Sems Portal
login_url = "https://www.semsportal.com/home/login"
driver.get(login_url)

# Completar el formulario de inicio de sesión
time.sleep(2)  # Esperar a que la página cargue

# Esperar a que la página cargue y localizar la casilla de condiciones de uso y política de privacidad
checkbox = WebDriverWait(driver, 10).until(
    EC.element_to_be_clickable((By.ID, "readStatement"))
)

# Hacer clic en la casilla de verificación si no está ya seleccionada
if not checkbox.is_selected():
    checkbox.click()

email_input = driver.find_element(By.ID, "username")
password_input = driver.find_element(By.ID, "password")
login_button = driver.find_element(By.ID, "btnLogin")

# Credenciales
your_email = "*****************"
your_password = "********"

email_input.send_keys(your_email)
password_input.send_keys(your_password)
login_button.click()

time.sleep(5)  # Esperar a que la página cargue
```

*Figura 3.4: Fragmento de código para el inicio de sesión en Sems Portal.*

**Paso 2: Selección del edificio correspondiente**

Una vez iniciamos sesión, la interfaz con la que nos encontramos es la que se muestra en la Figura 3.5. En ella se observan los diferentes edificios que podemos seleccionar con información general como:

- *Planta*: identifica cada edificio.
- *Ubicación:* indica la ubicación donde se encuentra el edificio.
- *Capacidad:* muestra la capacidad total de generación de energía del edificio.
- *Rendimiento:* muestra el porcentaje de producción actual con respecto a la capacidad total del edificio.

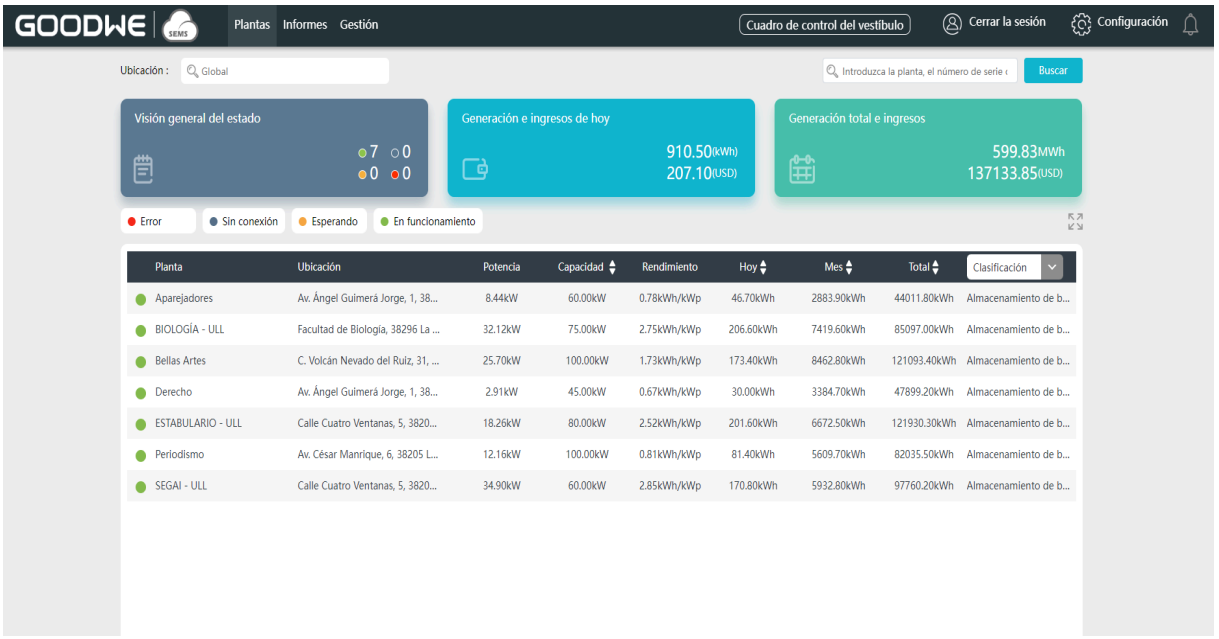


*Figura 3.5: Interfaz de Sems Portal con la lista de edificios disponibles.*

El siguiente código que se ilustra en la Figura 3.6 describe el proceso de selección para localizar el elemento que contiene el texto “SEGAI-ULL”:

```
# Localizar el elemento por el texto que contiene
plant_link = WebDriverWait(driver, 10).until(
    EC.element_to_be_clickable((By.XPATH, "//span[contains(text(), 'SEGAI - ULL')]"))
)
plant_link.click()

# Esperar a que la página cargue
time.sleep(5)  # Espera para que todos los elementos se carguen completamente
```

*Figura 3.6: Fragmento de código que permite localizar el texto “SEGAI-ULL”.*

Para identificar el edificio deseado, se utiliza un selector XPath específico. Este selector busca un elemento span dentro del código HTLM que contenga el texto "SEGAI - ULL" en cualquier lugar dentro de su contenido. De esta manera, se asegura una búsqueda precisa y robusta. Por otro lado, se emplea la clase WebDriverWait nuevamente para esperar hasta que el elemento que contiene el nombre del edificio sea visible y clickable. Por último, tras seleccionar el edificio, se espera un tiempo para que la página del edificio seleccionado cargue completamente.

**Paso 3: Extracción de datos**

La Figura 3.7 muestra la interfaz de Sems Portal una vez que se ha seleccionado el edificio SEGAI - ULL. Dentro de esta interfaz existe una sección de gráficos y tendencias que muestran la información detallada sobre la producción y el consumo del edificio.

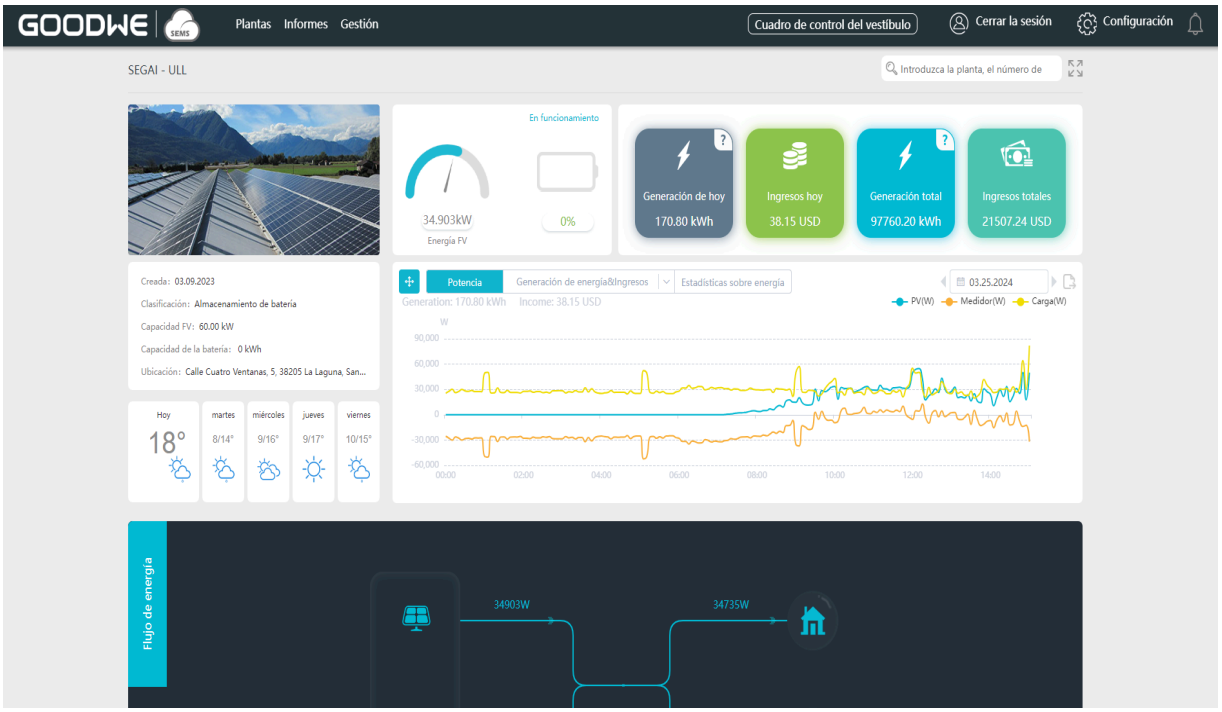


*Figura 3.7: Interfaz de Sems Portal del edificio SEGAI.*

El objetivo principal es poder extraer los datos que ofrecen estos gráficos, para ello, debemos hacer clic en el icono de extracción de datos que se encuentra en la esquina superior derecha del gráfico señalado con un círculo en rojo en la Figura 3.8.

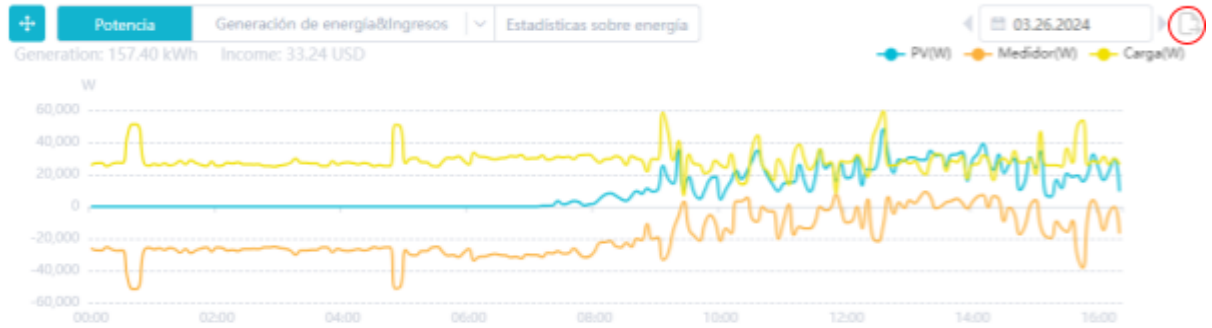


*Figura 3.8: Gráfico de tendencia de producción y consumo.*

Para poder extraer los datos, primero se establece un tiempo de espera para que desaparezcan los dos elementos específicos que no permiten descargar los datos de forma instantánea:

- *'.el-loading-mask':* este elemento indica que la página está cargando datos.
- *'.el-loading-spinner'*: este elemento es un spinner que se muestra mientras se cargan los datos.

Una vez estos elementos desaparezcan y la página esté lista para la siguiente acción, se utiliza la biblioteca "ActionsChains" para realizar acciones como mover el cursor y poder hacer clic. Por tanto, se crea una instancia de "ActionChains" y se mueve el cursor sobre el icono de exportación (.goodwe-station-charts__export.fr).

```
# Espera de overload y spinner de carga no visible
WebDriverWait(driver, 20).until(EC.invisibility_of_element((By.CSS_SELECTOR, '.el-loading-mask')))
WebDriverWait(driver, 20).until(EC.invisibility_of_element((By.CSS_SELECTOR, '.el-loading-spinner')))

# Realizamos la acción del clic usando ActionChains
action = ActionChains(driver)
export_icon = WebDriverWait(driver, 20).until(
    EC.element_to_be_clickable((By.CSS_SELECTOR, ".goodwe-station-charts__export.fr"))
)
action.move_to_element(export_icon).click().perform()
```

*Figura 3.9: Fragmento de código para la descarga del archivo.*

Además, para evitar errores y garantizar la descarga, se implementa un sistema de reintentos donde la acción de descargar el archivo se realiza en tres ocasiones.

Por otro lado, se ha añadido una función llamada "esperar_descarga_completa" como se muestra en la Figura 3.9, que se encarga de esperar hasta que la descarga del archivo se haya completado. La función comienza almacenando el tiempo actual en la variable start_time. Esta variable se utiliza para controlar el tiempo que ha transcurrido desde que se inició la descarga del archivo.

La función entra en un bucle que se ejecuta hasta que se cumpla una de las dos condiciones:

- Se encuentre un archivo con la extensión especificada en la carpeta de descarga.
- Se supere el tiempo de espera.

Dentro del bucle, se emplea una función que nos permite obtener todos los archivos que coinciden con la extensión especificada en la carpeta de descarga, devolviéndonos así una lista con las rutas de los archivos que coinciden con la búsqueda.

Si se encuentra algún archivo en la lista, se utiliza la función "max" para seleccionar el archivo más reciente. Si se ha encontrado un archivo se imprime un mensaje que indica que el archivo se ha descargado correctamente y se muestra la ruta del archivo. Si no se encuentra ningún archivo en la lista y el tiempo de espera timeout ha sido superado, se imprime un mensaje que indica que se ha excedido el tiempo de espera para la descarga.

```python
# Función para verificar la descarga completa del archivo y guardarlo como el archivo más reciente
def esperar_descarga_completa(extension="xls", timeout=60):
    start_time = time.time()
    while True:
        files = glob.glob(download_folder + f"/*.{extension}")
        if files:
            latest_file = max(files, key=os.path.getctime)  # Obtener el archivo más reciente
            print("Archivo descargado con éxito: " + latest_file)
            print()
            return latest_file
        elif time.time() - start_time > timeout:
            print("Tiempo de espera para la descarga excedido.")
            print()
            return None
        time.sleep(1)
```

*Figura 3.10: Función para verificar la descarga completa del archivo.*

### 3.2.3. Extracción de datos históricos de cada edificio

Los pasos iniciales para extraer todos los datos históricos de cada una de las plantas son los mismos que en la fase anterior. Sin embargo, una vez que nos encontremos en la interfaz específica del edificio, se debe seguir un proceso distinto:

1. Localizar el botón que se observa señalado con un círculo rojo en la Figura 3.11 para cambiar de día. Con cada clic se procede a retroceder un día.
2. Hacer clic sobre el icono de exportación para iniciar la descarga de los datos de la planta actual.
3. Repetir los pasos 1 y 2 en base a los días de los que la planta dispone de información.

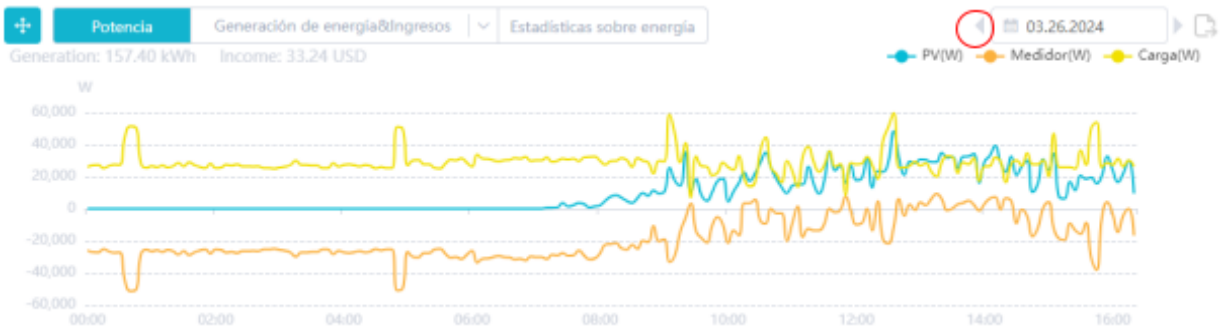


*Figura 3.11: Gráfico de tendencia de producción y consumo indicando el icono para cambiar de día.*

Para lograr esto, se implementó un bucle que incluye el paso 3 mencionado en la sección anterior, el cual se ejecutará según los días de los que deseamos obtener datos. Al final del bucle, se añadió un fragmento de código tal y como se muestra en la Figura 3.12 que comprueba si la descarga del archivo se ha realizado correctamente o si es el primer día. Si se cumple alguna de estas condiciones, se clica en el icono mostrado en la Figura 3.11 y se retrocede un día.

```
# Si el archivo se descargó correctamente o después del primer día, procede a retroceder un día
# Se añade la comprobación de archivo_descargado para asegurar que solo retrocedemos si el día actual fue exitoso
if day != 0 and archivo_descargado:
    WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.CSS_SELECTOR, ".station-date-picker_left"))).click()
    time.sleep(2)  # Espera a que la página se actualice
```

*Figura 3.12: Fragmento de código para retroceder un día en la extracción de los datos.*

El código correspondiente a la extracción de datos históricos del edificio SEGAI se encuentra en el archivo *SEGAI_IAAS_AnalisisHistorico.ipynb* (para más detalle ver Apéndice A.1).

### 3.2.4. Extracción de datos para todos los edificios

Para automatizar la extracción de datos de todos los edificios en un único código, se creó una lista llamada *"edificios_tokens"*. Esta lista contiene la información que permite identificar y acceder a cada edificio.

Cada elemento de la lista es un diccionario que contiene tres claves:

1. *"nombre":* permite identificar y localizar el edificio dentro de la interfaz de Goodwe.
2. *"token":* token de acceso único que nos permite enviar datos a ThingsBoard.
3. *"url":* la URL que te lleva a la interfaz del edificio, permitiendo verificar que los datos descargados corresponden al edificio correcto.

Para la extracción de datos, ahora mismo solo nos interesa conocer el texto por el cual podemos localizar el edificio dentro de la interfaz de Sems Portal y su url específica, por tanto, ignoraremos por ahora el token de cada uno de los edificios.

| LISTA DE EDIFICIOS | |
|---|---|
| ***Nombre*** | ***Url*** |
| Aparejadores | https://www.semsportal.com/powerstation/PowerStatusSnMin/cf4818e8-f972-440d-919d-2eefd176fdf9 |
| BIOLOGÍA - ULL | https://www.semsportal.com/powerstation/PowerStatusSnMin/6012db63-bcc1-4c85-84d1-373c87c16838 |
| Bellas Artes | https://www.semsportal.com/powerstation/PowerStatusSnMin/641e57d7-2762-4d6a-9f02-4fa9049539f3 |
| Derecho | https://www.semsportal.com/powerstation/PowerStatusSnMin/b2c6d945-2a31-4a26-861c-f0c83890d691 |
| ESTABULARIO - ULL | https://www.semsportal.com/powerstation/PowerStatusSnMin/a55500ef-bf7e-4847-99a9-3e3f5d336e95 |
| Periodismo | https://www.semsportal.com/powerstation/PowerStatusSnMin/3bf6f983-4566-4018-8c5f-25aad14464eb |
| SEGAI - ULL | https://www.semsportal.com/powerstation/PowerStatusSnMin/2c6eb65e-dca9-4c6f-810f-769064cc6ca8 |

*Tabla 3.1: Contenido de la lista "edificios_tokens".*

En este caso, el proceso de inicialización del navegador se encapsuló dentro de una función denominada "inicializar_navegador()". Adicionalmente, se implementó otra función, "iniciar_sesion_y_descargar()", encargada de iniciar sesión y descargar los datos correspondientes a cada edificio. Esta función recibe como parámetros el controlador del navegador, la ruta de destino para el archivo descargado y el identificador del edificio cuyos datos se desean obtener.

Un cambio notable en esta última función en comparación con la extracción de datos de un solo edificio, es que para acceder a la información del edificio especificado, se aprovechan los datos almacenados previamente en la lista. Este procedimiento se ilustra en la Figura 3.13.

```
# Navegar al enlace específico del edificio
WebDriverWait(driver, 10).until(
    EC.element_to_be_clickable((By.XPATH, f"//span[@title='{edificio['nombre']}']"))
).click()

time.sleep(5)  # Esperar a que todos los elementos se carguen completamente


# Navegar a la URL específica del Edificio
driver.get(edificio['url'])
time.sleep(3)  # Espera para asegurar que la página ha cargado completamente
```

*Figura 3.13: Fragmento de código que permite acceder a la interfaz de cada edificio.*

Para descargar los datos de cada uno de los edificios, fuera de estas funciones, se itera a través de la lista previamente definida. Al comenzar cada iteración del bucle, se llama a la función "inicializar_navegador()". La instancia del navegador se almacena en la variable driver, la cual se utilizará para controlar y automatizar las interacciones con las páginas web.

Después de inicializar el navegador, se lleva a cabo la operación de inicio de sesión y descarga de datos para el edificio actual, haciendo uso de la función "iniciar_sesion_y_descargar()".

Por otro lado, la estructura de control try-finally asegura que, independientemente de si la operación de descarga se completa con éxito o si ocurre un error durante el proceso, la sesión del navegador se cierre correctamente. Esto se debe a que existe un límite de tiempo para que la sesión en el portal de SEMS permanezca activa. Si el navegador no se cierra, la sesión podría expirar y el proceso de descarga de datos fallaría.

```
for edificio in edificios_tokens:
    driver = inicializar_navegador()
    try:
        iniciar_sesion_y_descargar(driver, download_folder, edificio)
    finally:
        driver.quit()  # Cerramos el navegador
```

*Figura 3.14: Fragmento de código para descargar los datos de todos los edificios.*

El código correspondiente a la extracción de datos de cada uno de los edificios se encuentra en el archivo *Web2ThingsBoard.ipynb* (para más detalle ver Apéndice A.1).

## 3.3. Incorporación y sincronización de datos en ThingsBoard

Para integrar los datos a la plataforma ThingsBoard, primero debemos crear dispositivos virtuales dentro de la misma. Estos dispositivos virtuales actúan como entidades que representan los sensores o dispositivos físicos de los que se obtienen los datos. Al crear un dispositivo virtual, se genera un token de acceso único que permite enviar datos de telemetría a la plataforma en nombre del dispositivo.

Este token de acceso es un elemento fundamental para la comunicación entre los datos extraídos de cada edificio y la plataforma ThingsBoard.

### 3.3.1. Creación de dispositivos

Para agregar un nuevo dispositivo debemos acceder a la plataforma ThingsBoard. Una vez dentro de la plataforma, nos dirigimos a la sección "Entidades" del menú lateral. En esta sección, se selecciona la pestaña "Dispositivos".

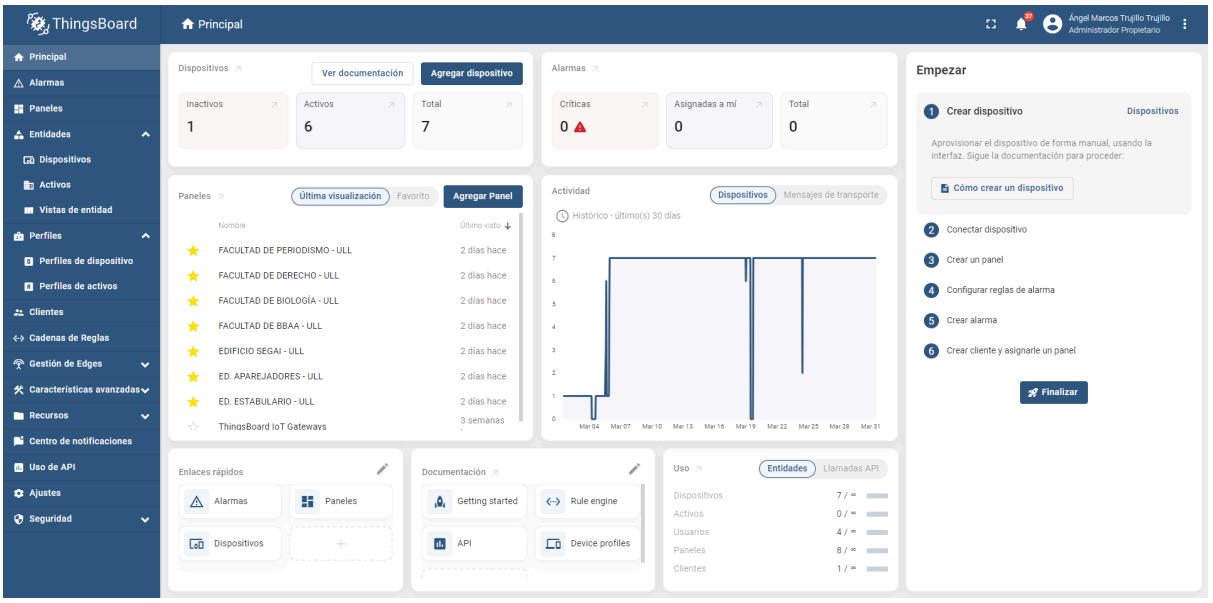


*Figura 3.15: Interfaz principal de ThingsBoard.*

En la esquina superior derecha, seleccionamos el icono "+" y luego se selecciona "Agregar nuevo dispositivo" en el menú desplegable.

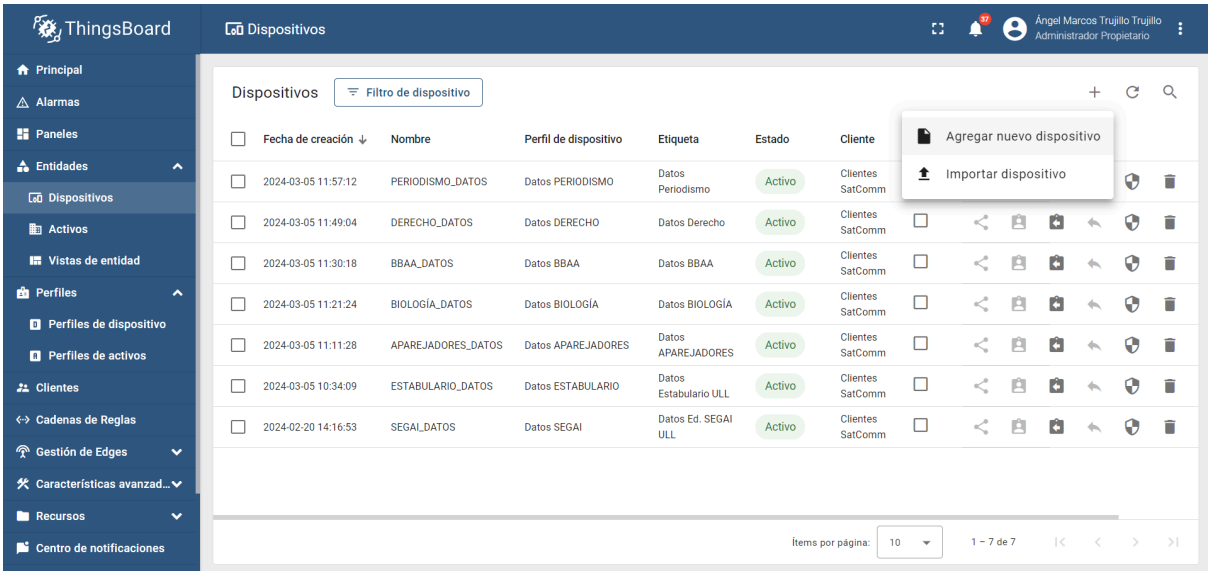


*Figura 3.16: Interfaz de ThingsBoard antes de agregar nuevo dispositivo.*

En el formulario para agregar un nuevo dispositivo, uno de los campos obligatorios es el nombre del dispositivo. Este nombre debe ser único para el dispositivo que se está creando. En nuestro caso, se ha elegido el nombre del edificio como nombre del dispositivo para facilitar la identificación del dispositivo en la plataforma ThingsBoard.

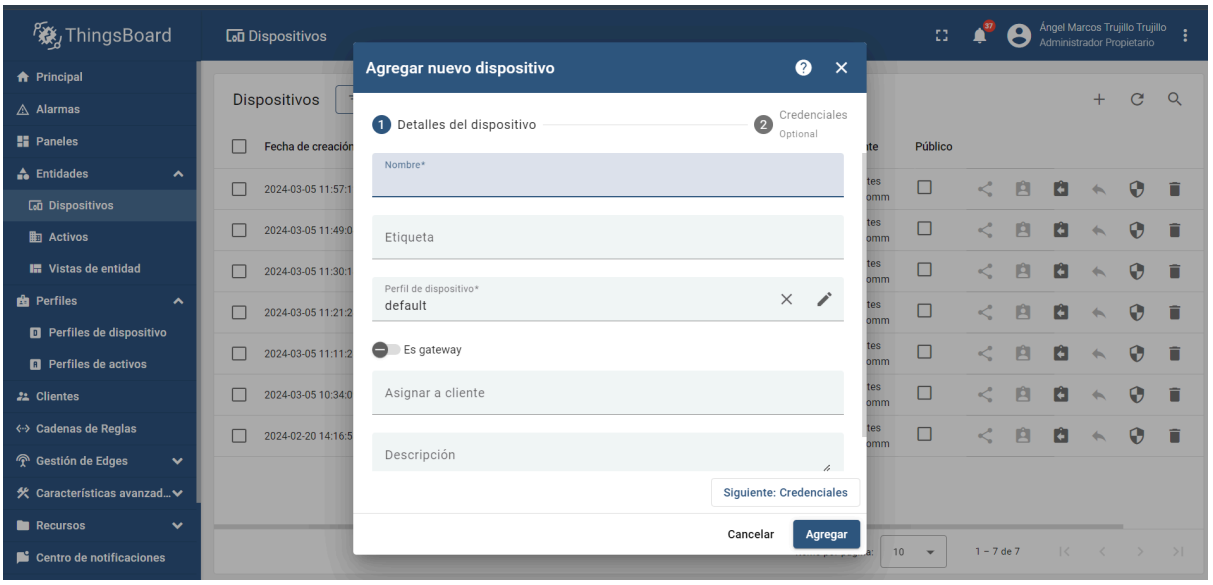


*Figura 3.17: Interfaz de ThingsBoard al clicar en agregar nuevo dispositivo.*

Una vez creado el dispositivo virtual, se mostrará la página de resumen del dispositivo. En esta página, se encuentra información importante sobre el dispositivo, como el token de acceso.

El token de acceso es una cadena de caracteres alfanuméricos que se utiliza para autenticar al dispositivo y poder enviar datos de telemetría a la plataforma.

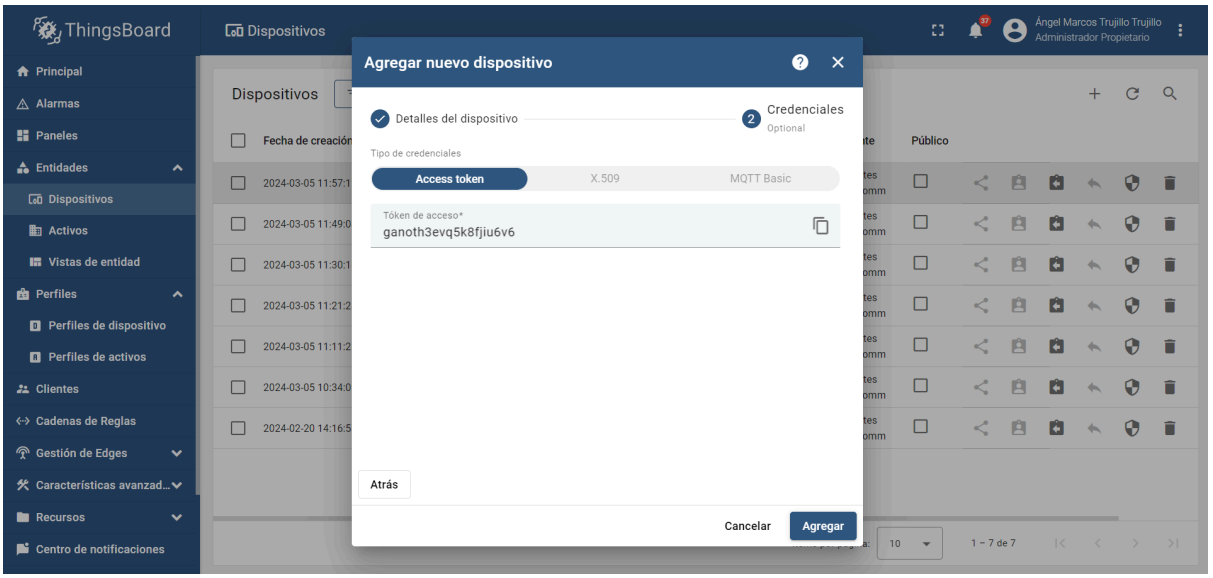


*Figura 3.18: Credenciales del dispositivo.*

Este proceso se repetirá para cada uno de los siete dispositivos que manejamos, resultando en un total de siete dispositivos virtuales y siete tokens de acceso.

### 3.3.2. Integración de datos

La integración de datos en ThingsBoard se efectúa a través de un proceso automatizado en el cual los datos de telemetría extraídos previamente de cada edificio se transmiten a la plataforma haciendo uso de las interfaces de programación de aplicaciones (APIs).

El primer paso es preparar el archivo de datos que contiene la información que se desea enviar a ThingsBoard. En este caso, se trata de un archivo Excel con las siguientes columnas:

- *Time:* fecha y hora de la medición.
- *PV(W):* potencia generada por los paneles fotovoltaicos.
- *Load(W):* potencia consumida por el edificio.
- *Meter(W):* potencia que falta o sobra de energía fotovoltaica para cubrir el consumo del edificio.

Para poder enviar los datos, se crea una función que se añade a los script nombrados en apartados anteriores. La función recibe como argumentos el nombre del archivo y el token de acceso del dispositivo virtual creado en ThingsBoard.

Dentro de ella, se itera sobre cada fila del archivo Excel y se construye un objeto JSON con los datos de la medición. Este objeto JSON se envía a la API de ThingsBoard utilizando el token de acceso del dispositivo del edificio correspondiente.

Posteriormente, se verifica el código de respuesta de la API de ThingsBoard para confirmar que los datos se enviaron correctamente.

Por último, si los datos se enviaron correctamente, el archivo Excel utilizado se elimina automáticamente de la carpeta de origen. Esta acción se realiza para evitar redundancias y mantener la eficiencia del sistema de gestión de datos.

En situaciones donde el envío de datos no se completa satisfactoriamente debido a un error, se procede igualmente a eliminar el archivo Excel. Esta medida se adopta para mitigar el riesgo de que los datos de telemetría, en intentos posteriores de envío, no se correspondan adecuadamente con el edificio real al que pretenden representar. La eliminación del archivo en caso de fallo garantiza que cada intento de transmisión se base en datos precisos y actualizados, evitando la acumulación de archivos obsoletos o incorrectos que podrían llevar a inexactitudes en el procesamiento de datos.

El código correspondiente a la función diseñada para la transmisión de datos hacia ThingsBoard se muestra detalladamente en la Figura 3.19:

```python
def enviar_a_thingsboard(archivo_reciente, token):
    # URL del Sevirdor IAAS y Token del Dispositivo en ThinsBoard
    url = f"https://greenenergy.iaas.ull.es:8080/api/v1/{token}/telemetry"

    # Leer el archivo excel
    df = pd.read_excel(archivo_reciente, header=2)
    # Inicializar el contador de datos enviados exitosamente
    envios_exitosos = 0
    # Iterar sobre cada fila del DataFrame y enviar los datos a ThingsBoard
    for index, row in df.iterrows():
        # Construir el objeto de datos con las filas del DataFrame y el tiempo
        data = {
            "ts": pd.to_datetime(row['Time']).timestamp() * 1000, # Formato UNIX
            "values": {
                "PV(W)": row['PV(W)'],
                "Meter(W)": row['Meter(W)'],
                "Load(W)": row['Load(W)']
            }
        }
        headers = {'Content-Type': 'application/json', 'X-Authorization': f"Bearer {token}"}
        response = requests.post(url, data=json.dumps(data), headers=headers)
        # Verificación de datos enviados correctamente
        if response.status_code == 200:
            print(f"Dato enviado correctamente: {data}")
            envios_exitosos += 1
        else:
            print(f"Error al enviar dato: {response.text}")

    # Si los datos fueron enviados correctamente, borramos archivo de carpeta de origen
    if envios_exitosos == len(df):
        print()
        print("Todos los datos han sido enviados a ThingsBoard exitosamente. Eliminando archivo...")
        os.remove(archivo_reciente)
        print("Archivo eliminado con éxito.")
    else:
        print("Algunos datos no se pudieron enviar correctamente. El archivo se eliminará igualmente.")
        os.remove(archivo_reciente)
        print()
```

*Figura 3.19: Código de la función para integrar los datos.*

### 3.3.3. Automatización y programación de la transmisión de datos

Después de establecer los dispositivos virtuales y confirmar la eficacia del script *Web2ThingsBoard* para el envío de datos de telemetría, se introduce un temporizador en el *Servidor IAAS* para automatizar la ejecución del script. Este temporizador está configurado para activar el script automáticamente cada 8 minutos, garantizando así la obtención y el envío periódicos de datos actualizados para cada edificio a sus respectivos dispositivos en ThingsBoard.

La programación de esta ejecución trae consigo varias ventajas significativas:

- *Automatización:* el proceso de recolección y transmisión de datos se lleva a cabo de manera autónoma, eliminando la necesidad de intervención manual.
- *Actualización Continua:* la periodicidad de 8 minutos para la ejecución del código asegura que la información en ThingsBoard se mantenga constantemente actualizada, reflejando los cambios y condiciones más recientes de cada edificio.

A pesar de estos beneficios, es crucial llevar a cabo una supervisión regular del funcionamiento del script. Esta vigilancia constante es esencial para asegurar que el flujo de datos hacia *ThingsBoard* se mantenga sin interrupciones..

## 3.4. Cuadros de mando para la visualización de la información

Los paneles de ThingsBoard son herramientas esenciales para la visualización y monitorización de datos en tiempo real de dispositivos IoT. En este apartado, se describe el proceso de creación de diferentes paneles y la configuración de widgets específicos para mostrar:

- *Lista de entidades:* muestra una lista actualizada de entidades junto a sus valores más recientes (consumo, generación, etc.).
- *Gráficas temporales:* visualiza la evolución temporal de dos variables clave:
    - Consumo: permite observar el comportamiento del consumo energético del edificio en un periodo determinado.
    - Generación fotovoltaica: muestra la producción de energía por parte de los paneles solares en un intervalo de tiempo específico.
- *Señales de alarma:* informa sobre las alarmas activas o históricas relacionadas con una entidad específica (edificio), facilitando la identificación de posibles problemas o anomalías.

### 3.4.1. Creación del panel

Para crear un nuevo panel hay que navegar a la página “Paneles” en el menú principal. Una vez en esta sección, se debe hacer clic en el icono representado por un signo "+", situado en la esquina superior derecha de la pantalla. Al interactuar con este ícono, se desplegará un menú que incluye la opción "Crear nuevo panel", la cual debe ser seleccionada para proceder.

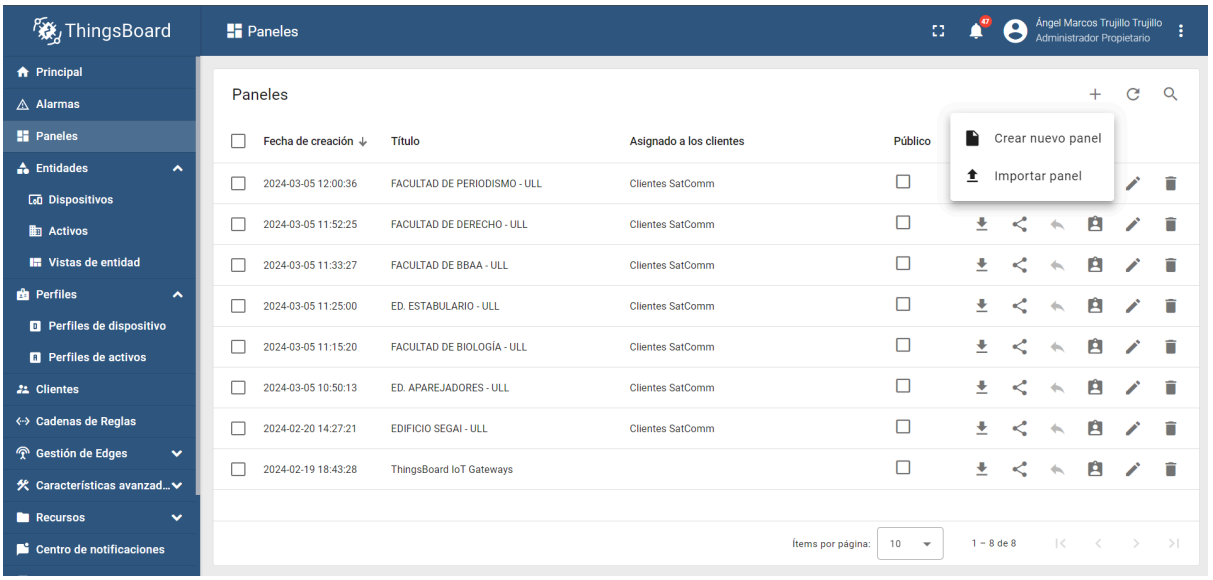


*Figura 3.20: Interfaz de gestión de paneles.*

Al acceder a la sección, se presenta un cuadro de diálogo donde se debe introducir un título para el nuevo panel. En nuestro caso, el título del panel corresponderá con el nombre del Edificio correspondiente.

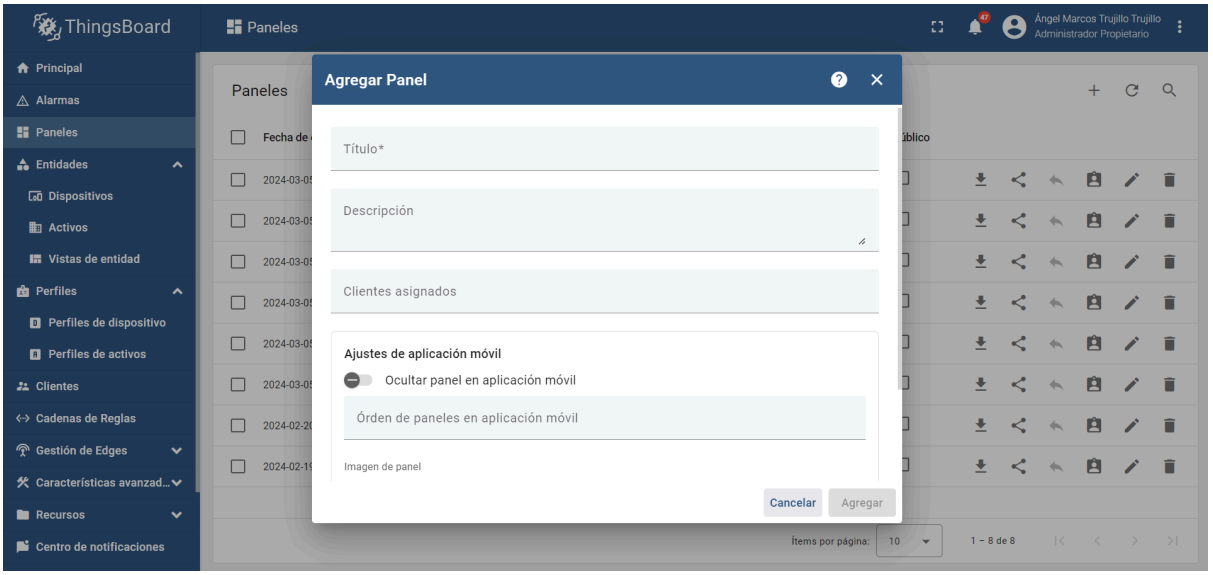


*Figura 3.21: Cuadro de diálogo para agregar un nuevo panel.*

Con la información debidamente introducida, se concreta la creación del panel, que se abrirá automáticamente para su edición. El siguiente paso implica la personalización del panel a través de la adición de widgets.


*Figura 3.22: Panel de mando vacío.*

## 3.4.2. Creación de widgets

Los widgets son componentes esenciales en cualquier sistema de gestión y visualización de datos. Al emplear widgets, los usuarios pueden configurar paneles para monitorizar y controlar dispositivos IoT, así como para visualizar datos en tiempo real y tendencias históricas de manera eficiente.

### 3.4.2.1. Widget de tabla entidades

El primer paso consiste en ingresar al modo de edición del panel seleccionado, lo cual se logra abriendo el panel y haciendo clic en el botón "Modo de edición" ubicado en la esquina superior derecha. Una vez en modo de edición, se tiene la opción de añadir un widget a través del botón "Agregar widget".

En el siguiente paso, se explora la biblioteca de widgets para encontrar el paquete denominado "Tablas". Dentro de este paquete, se selecciona el widget "Tabla de entidades", lo que desencadena la apertura de una ventana para agregar el widget.

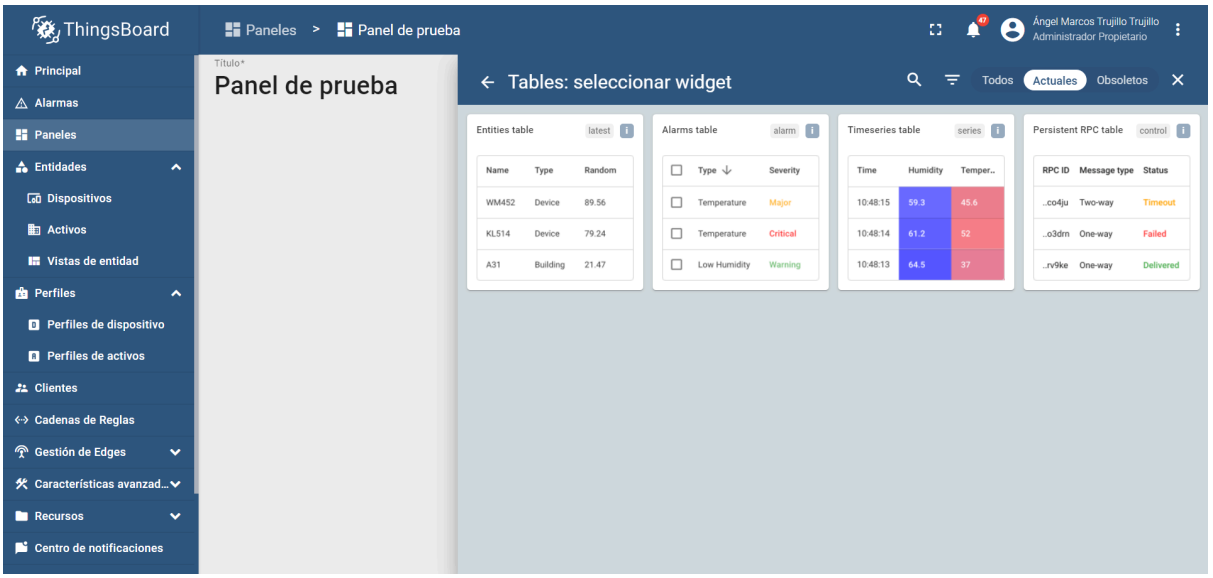


*Figura 3.23: Selección del widget de la tabla de entidades.*

Posteriormente, se especifica el dispositivo previamente creado para cada uno de los edificios como fuente de datos, permitiendo que el widget identifique qué información representa.

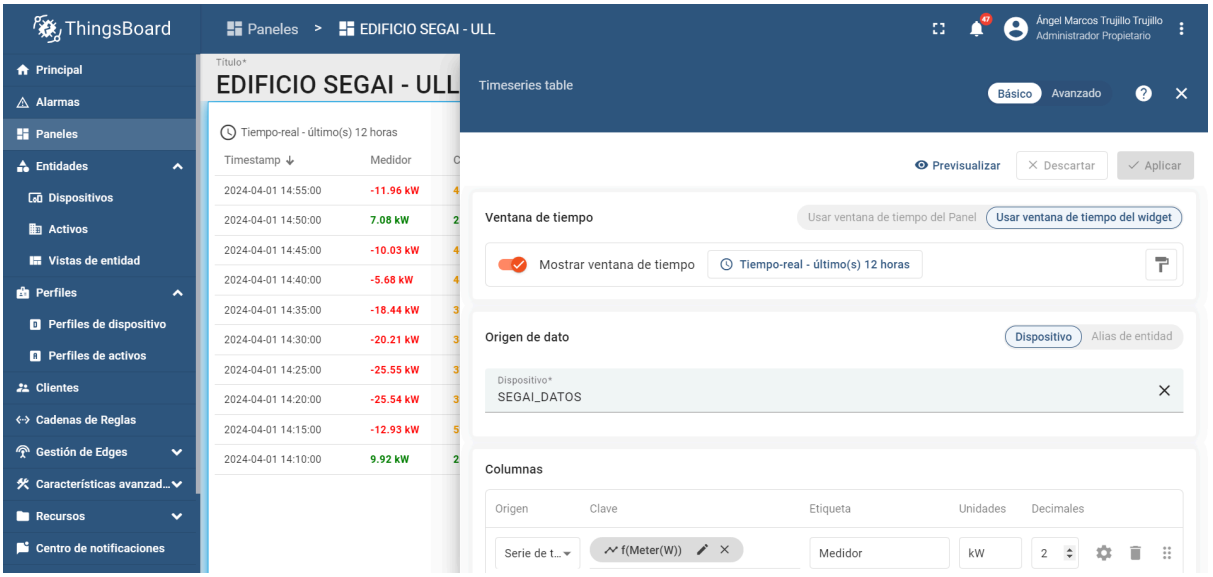


*Figura 3.24: Configuración del widget de tabla de series temporales.*

Para visualizar la información de los datos extraídos de cada una de las plantas, se debe ir añadiendo columnas. Esto se hace a través de la opción "Agregar columna", donde se introducirá una nueva clave de datos. Al hacer clic en el campo, se despliega una lista con las claves de datos disponibles, de la cual se debe seleccionar "Meter(W), Load(W) y PV(W)".

Además, se realizó una conversión de cada una de las medidas, originalmente en vatios, a kilovatios. Esta transformación se logró mediante la inclusión de una función de

post-procesamiento en los ajustes generales de las claves de datos. Como se muestra en la Figura 3.25, la función aplicada divide el valor de la entrada en vatios por 1000.

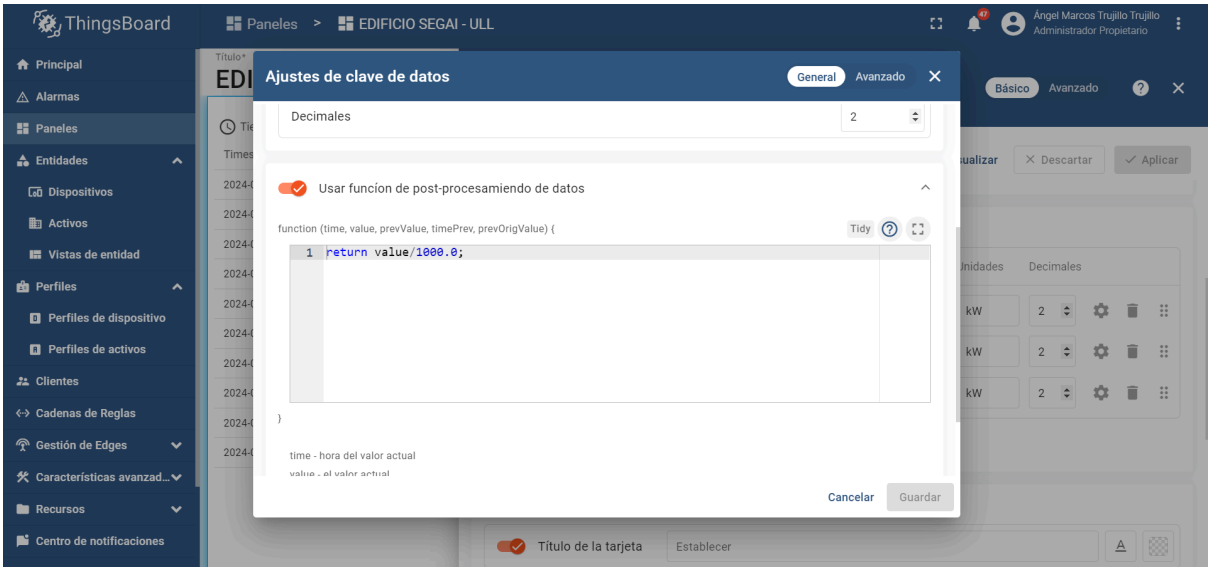


*Figura 3.25: Configuración de función de post-procesamiento de unidades.*

En la configuración avanzada de las claves de datos, se ha implementado una función de estilo de celda que aporta una capa adicional de interactividad y claridad a la visualización de datos. Esta función está diseñada para cambiar el color de la celda en función del valor numérico que contiene cada variable. De tal manera, los valores se representan visualmente a través de un código de colores que facilita la rápida identificación de su estado. Por ejemplo, valores críticos pueden aparecer en rojo, mientras que valores normales pueden mostrarse en verde, y así sucesivamente.

```
var medidor = value;
var color = 'black';
if (medidor) {
    if (medidor > 0) {
        color = 'green';
    } else {
        color = 'red';
    }
}
return {
  fontWeight: 'bold',
  color: color
};
```

*Figura 3.26: Estilo de celda de la variable Meter (W).*

La Figura 3.26 muestra un fragmento de código utilizado en ThingsBoard para aplicar estilos condicionales a las celdas de un widget de tabla en función de los datos de energía fotovoltaica y consumo de un edificio. En este caso, la variable medidor representa la diferencia entre la energía generada por la instalación fotovoltaica y el consumo del edificio.

Si el valor de medidor es mayor que cero, lo que indica que la energía generada es superior a la consumida, la celda se colorea de verde. Por el contrario, si la energía generada es igual o menor que la consumida, la celda se colorea de rojo.

Sin embargo, debido a la implementación de un sistema antivertido en cada edificio, esta característica visual adquiere un papel menos destacado, ya que la generación fotovoltaica se regula de tal manera que no sobrepasa el consumo de energía del edificio. Aunque es posible que en ciertas circunstancias, y en menor medida, la generación de energía fotovoltaica supere temporalmente el consumo ya que ningún sistema es infalible.

Tanto para la carga energética (Load) y la producción de energía fotovoltaica (PV), aplicó una función que asigna colores a las celdas basándose en los valores de estas variables. Para ello, se estudiaron los valores típicos y anormales de cada uno de los edificios.

Para las variables de carga energética (Load) y producción de energía fotovoltaica (PV), se implementó una función de estilo que asigna colores a las celdas en función de los valores que estas variables adquieren.

```
var carga = value;
var color = 'black';
if (carga > 80) {
    color = 'red';
} else if (carga >= 30) {
    color = 'orange';
} else {
        color = 'green';
}

return {
  fontWeight: 'bold',
  color: color
};
```

```
var pv = value;
var color = 'black';
if (pv > 50) {
    color = 'green';
} else if (pv >= 10) {
    color = 'orange';
} else {
        color = 'red';
}

return {
  fontWeight: 'bold',
  color: color
};
```

*Figura 3.27: Estilo de celda de las variables Load (W) y PV (W).*

Este enfoque se desarrolló tras analizar los patrones de consumo y generación fotovoltaica de los edificios, identificando rangos de valores considerados normales y aquellos que se desvían de la media. Mediante esta metodología, se estableció un sistema de codificación visual que facilita la monitorización en tiempo real y permite una rápida identificación de cualquier anomalía o eficiencia en la gestión energética del edificio. A continuación, se muestran las tablas de tendencias de consumo y generación fotovoltaica de cada uno de los edificios.

| ***Rango de valores*** | ***Condición*** |
|---|---|
| *Consumo del Edificio Estabulario* | |
| > 70 kW | Consumo elevado |
| 25 kW ≤ Consumo ≤ 70 kW | Consumo normal |
| < 25 kW | Consumo bajo |
| *Consumo de la Facultad de Periodismo* | |
| > 65 kW | Consumo elevado |
| 20 kW ≤ Consumo ≤ 65 kW | Consumo normal |
| < 20 kW | Consumo bajo |
| *Consumo de la Facultad de Aparejadores* | |
| > 50 kW | Consumo elevado |
| 15 kW ≤ Consumo ≤ 50 kW | Consumo normal |
| < 15 kW | Consumo bajo |
| *Consumo de la Facultad de Biología* | |
| > 100 kW | Consumo elevado |
| 35 kW ≤ Consumo ≤ 100 kW | Consumo normal |
| < 35 kW | Consumo bajo |
| *Consumo de la Facultad de Bellas Artes* | |
| > 95 kW | Consumo elevado |
| 20 kW ≤ Consumo ≤ 95 kW | Consumo normal |
| < 20 kW | Consumo bajo |
| *Consumo de la Facultad de Derecho* | |
| > 90 kW | Consumo elevado |
| 20 kW ≤ Consumo ≤ 90 kW | Consumo normal |
| < 20 kW | Consumo bajo |
| *Consumo del Edificio SEGAI* | |
| > 80 kW | Consumo elevado |
| 30 kW ≤ Consumo ≤ 80 kW | Consumo normal |
| < 30 kW | Consumo bajo |

*Tabla 3.2: Tendencias de consumo.*

| ***Rango de valores*** | ***Condición*** |
|---|---|
| *Generación fotovoltaica del Edificio Estabulario* | |
| > 50 kW | Generación óptima |
| 10 kW ≤ Generación ≤ 50 kW | Generación normal |
| < 10 kW | Poca generación |
| *Generación fotovoltaica de la Facultad de Periodismo* | |
| > 50 kW | Generación óptima |
| 10 kW ≤ Generación ≤ 50 kW | Generación normal |
| < 10 kW | Poca generación |
| *Generación fotovoltaica de la Facultad de Aparejadores* | |
| > 35 kW | Generación óptima |
| 10 kW ≤ Generación ≤ 35 kW | Generación normal |
| < 10 kW | Poca generación |
| *Generación fotovoltaica de la Facultad de Biología* | |
| > 70 kW | Generación óptima |
| 30 kW ≤ Generación ≤ 70 kW | Generación normal |
| < 30 kW | Poca generación |
| *Generación fotovoltaica de la Facultad de Bellas Artes* | |
| > 60 kW | Generación óptima |
| 20 kW ≤ Generación ≤ 60 kW | Generación normal |
| < 20 kW | Poca generación |
| *Generación fotovoltaica de la Facultad de Derecho* | |
| > 50 kW | Generación óptima |
| 10 kW ≤ Generación ≤ 50 kW | Generación normal |
| < 10 kW | Poca generación |
| *Generación fotovoltaica del Edificio SEGAI* | |
| > 50 kW | Generación óptima |
| 10 kW ≤ Generación ≤ 50 kW | Generación normal |
| < 10 kW | Poca generación |

*Tabla 3.3: Tendencia de generación fotovoltaica.*

### 3.4.2.2. Widgets de gráficos

Se incorporaron gráficos al panel de control para poder visualizar los datos a lo largo del tiempo y poder estudiar así el comportamiento de las variables monitorizadas. Para llevar a cabo esta implementación, fue necesario navegar por la biblioteca de widgets. Dentro de esta colección, se localizó el paquete denominado "Gráficos", y se seleccionó el widget específico "Gráfico de series temporales".

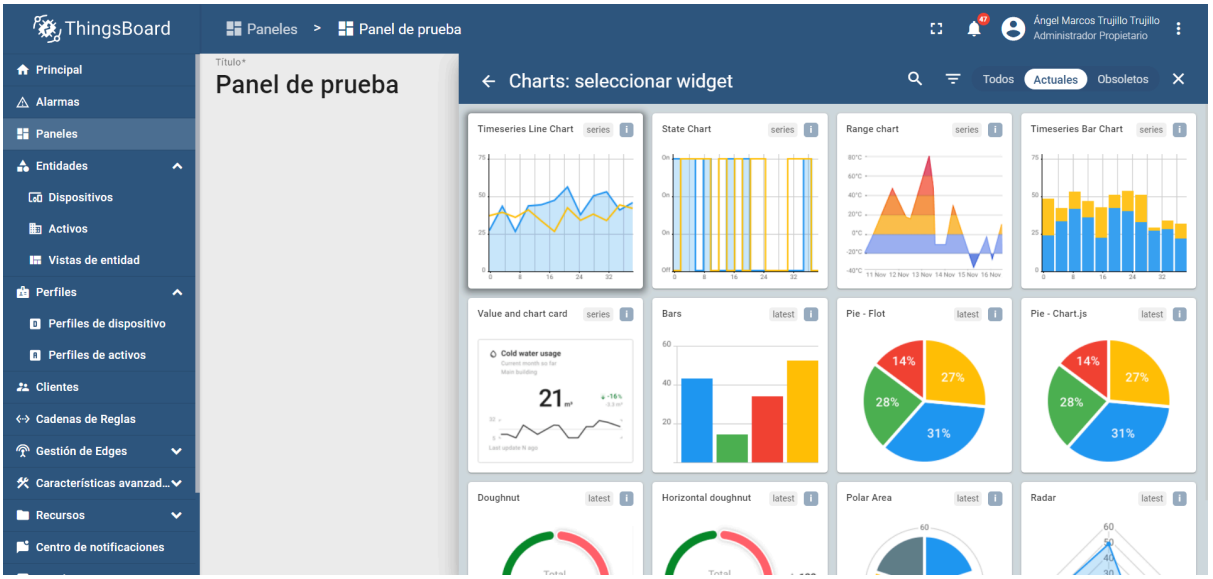


*Figura 3.28: Widgets de gráficos en ThingsBoard.*

Se implementaron dos gráficas distintas con el propósito de monitorizar la generación fotovoltaica y el consumo energético de los edificios a lo largo del tiempo. Para cada una de estas gráficas, se procedió a especificar los dispositivos correspondientes como fuentes de datos, asegurando que la información presentada refleja las métricas precisas de interés. Posteriormente, en la sección "Series" de la configuración del widget, se definieron las claves de datos relevantes para cada gráfica: una para el seguimiento del consumo energético del edificio (Load) y otra para la generación fotovoltaica (PV).

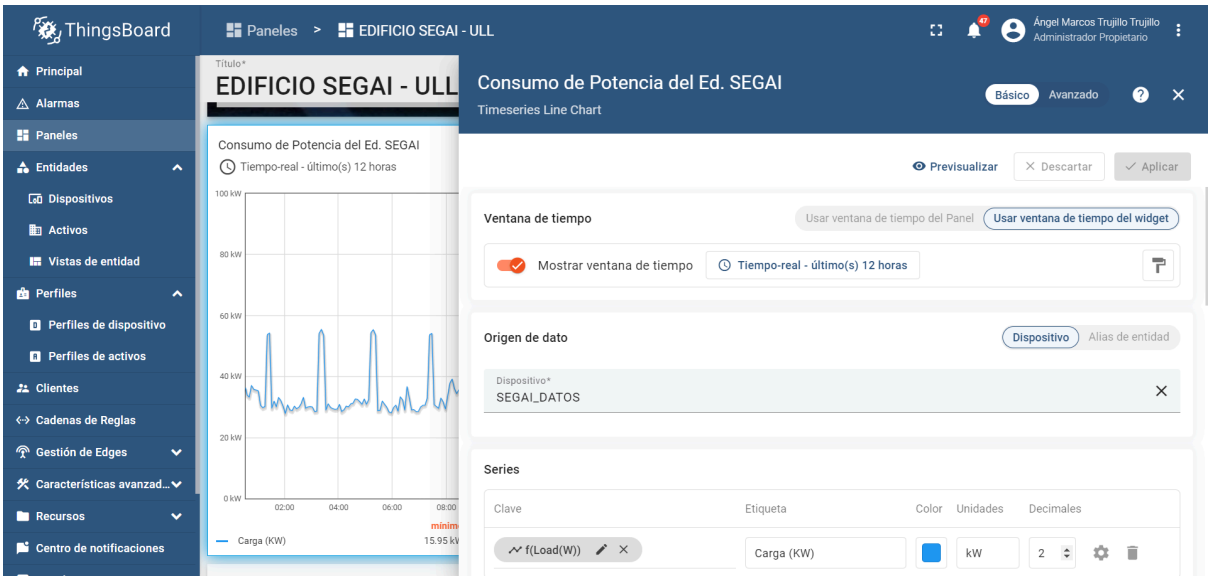


*Figura 3.29: Configuración del widget de gráfico de series temporales.*

### 3.4.2.3. Widgets de tarjetas

Para facilitar la visualización en tiempo real del consumo y la generación de energía de cada uno de los edificios, se implementaron widgets de gráfico con fondo. Se diseñaron con un fondo distintivo que representa visualmente el edificio y su instalación fotovoltaica, permitiendo una rápida identificación del mismo.


*Figura 3.30: Widgets de consumo y generación fotovoltaica actual.*

En cuanto a la configuración de colores de los widgets, se establecieron parámetros basados en los rangos de valores mostrados en las Tabla 3.2 y 3.3 para una interpretación cromática intuitiva de los datos. Los colores se asignan automáticamente según el valor actual que muestra el widget: por ejemplo, si el valor de consumo está dentro de un rango bajo, se muestra en verde, indicando condiciones normales o eficiencia energética. Un rango medio se

representa en naranja, señalando un estado de alerta moderada, y un rango alto en rojo, indicando un consumo que podría requerir atención.

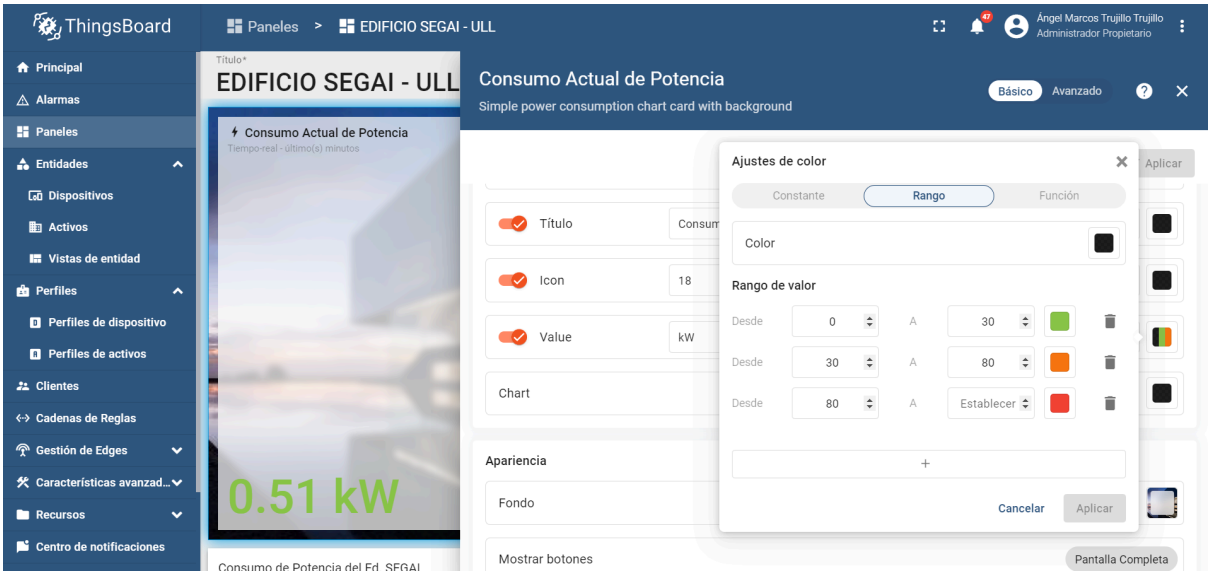


*Figura 3.31: Configuración de widgets de consumo y generación.*

### 3.4.2.4. Widgets de alarmas

El widget "Tabla de Alarmas" se utiliza para visualizar las alertas asociadas a una entidad en un intervalo de tiempo específico, proporcionando una herramienta crucial para el seguimiento en tiempo real de cualquier incidencia o anomalía.

Para su configuración, se inicia accediendo al modo de edición del panel de control. Dentro de la biblioteca de widgets disponibles, se navega hasta localizar la categoría "Widgets de alarma", donde se elige el widget "Tabla de alarmas".

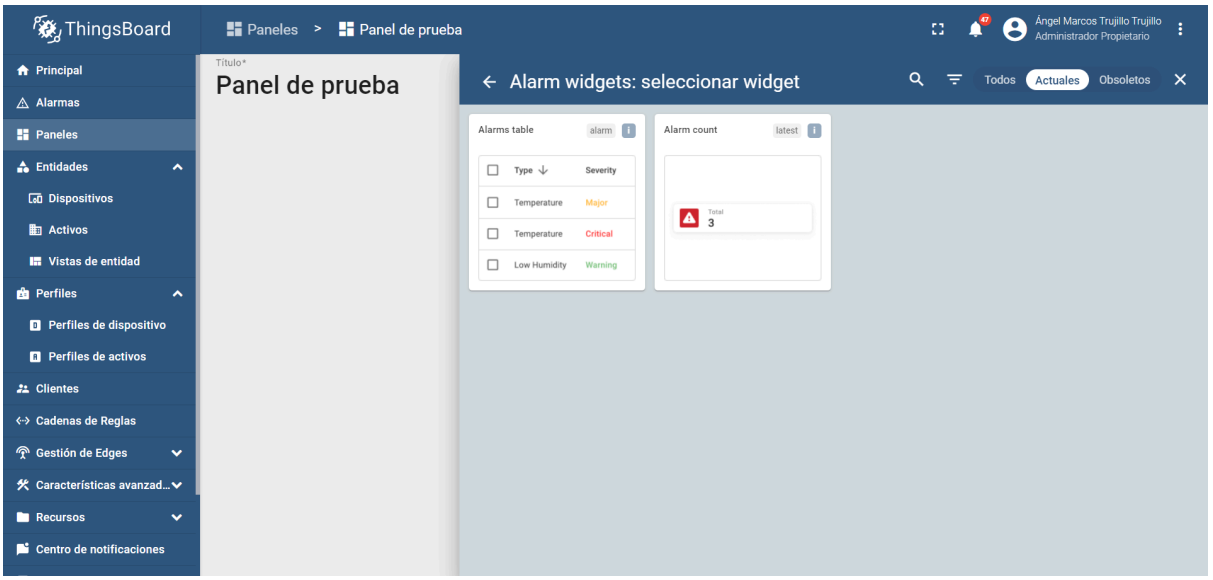


*Figura 3.32: Selección del widget de alarmas.*

La entidad que servirá como fuente de las alarmas se especifica mediante la selección del dispositivo previamente creado. Además, se configuran los parámetros detallados de las alarmas, tales como los estados y la severidad con la que serán representados en el widget.

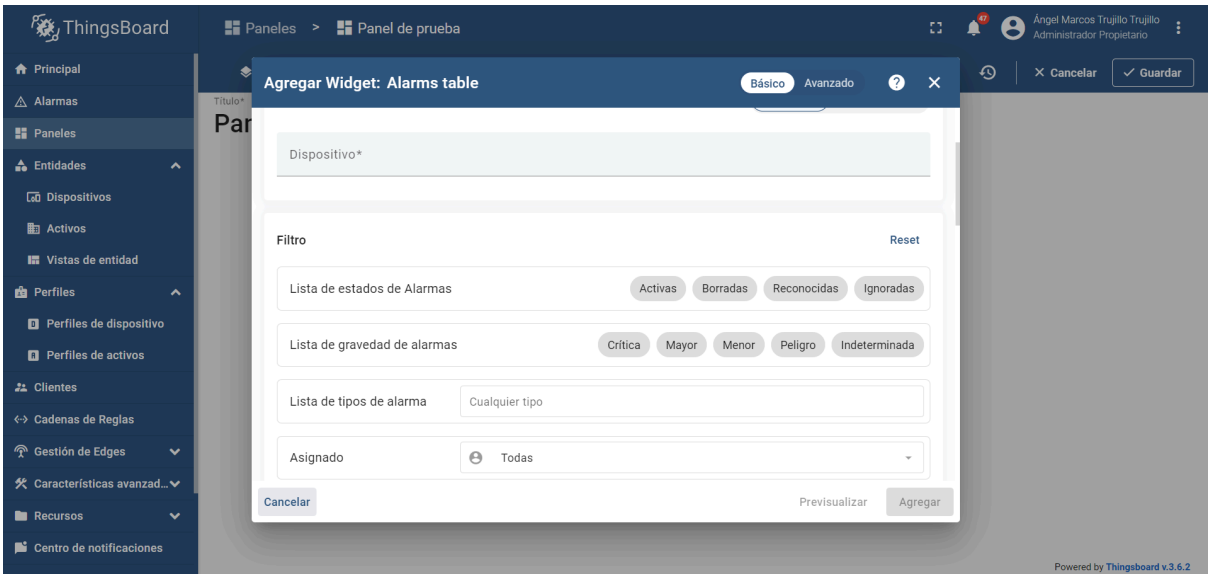


*Figura 3.33: Configuración del widget de la tabla de alarmas.*

Una vez añadido el widget de tabla de alarmas, el siguiente paso será detallar el proceso de configuración de las reglas de alarma y la forma en que serán generadas. Este proceso implica definir los criterios y condiciones bajo los cuales se activarán las alertas.

### 3.4.3. Configuración de reglas de alarma

Para establecer un sistema de alertas eficiente, se ha optado por utilizar la funcionalidad de reglas de alarma que ofrece *ThingsBoard*. Esta herramienta nos permite configurar alarmas que se activan bajo condiciones específicas, previamente definidas por el usuario.

Para proceder con esta configuración, es necesario establecer un perfil de dispositivo único para cada edificio dentro del sistema. Esta creación de perfiles se refleja en la Figura 3.34.

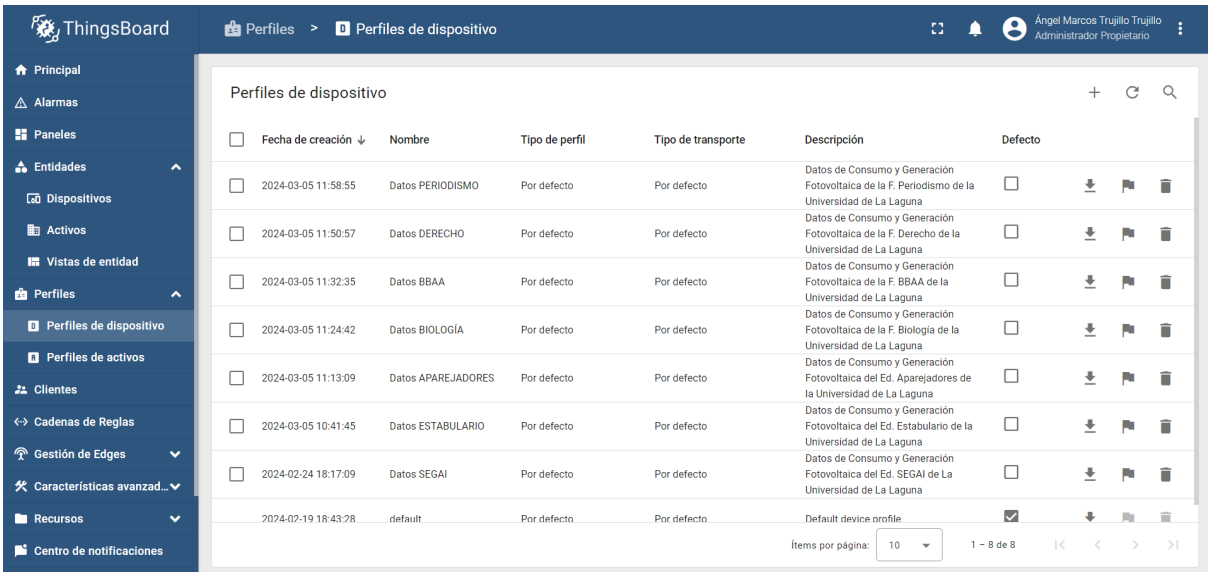


*Figura 3.34: Vista de perfiles de dispositivos.*

Dentro del perfil, nos dirigimos a la sección de "Reglas de alarma". Aquí es donde definimos el tipo de alarma y establecemos las condiciones que desencadenan la alerta. Para configurar estas condiciones, seleccionamos un tipo de clave y asignamos un nombre distintivo que identifique la naturaleza de la alarma. Por ejemplo, para las alarmas relacionadas con el consumo, se estableció el nombre "Notificación de Consumo Excesivo", mientras que para las de rendimiento fotovoltaico se usó "Notificación de Bajo Rendimiento PV". Posteriormente, elegimos un tipo de valor adecuado que corresponda a la métrica que estamos monitoreando. Esta configuración se ilustra en las Figuras 3.35 y 3.36.


*Figura 3.35: Alarma de consumo excesivo.*


*Figura 3.36: Alarma de bajo rendimiento PV.*

### 3.4.3.1. Creación de alarma

Con la regla de alarma ya activa, procedemos a la configuración del Widget de Alarma que se estableció previamente. En la sección correspondiente a la lista de tipos de alarma del widget, incorporamos las dos alarmas que habíamos definido: "Notificación de Bajo Rendimiento PV" y "Notificación de Consumo Excesivo". A estas alertas les asignamos el dispositivo específico al cual se aplican, asegurando así una correlación directa y efectiva entre la alarma y la entidad monitorizada. Esta configuración queda detallada en la Figura 3.37.

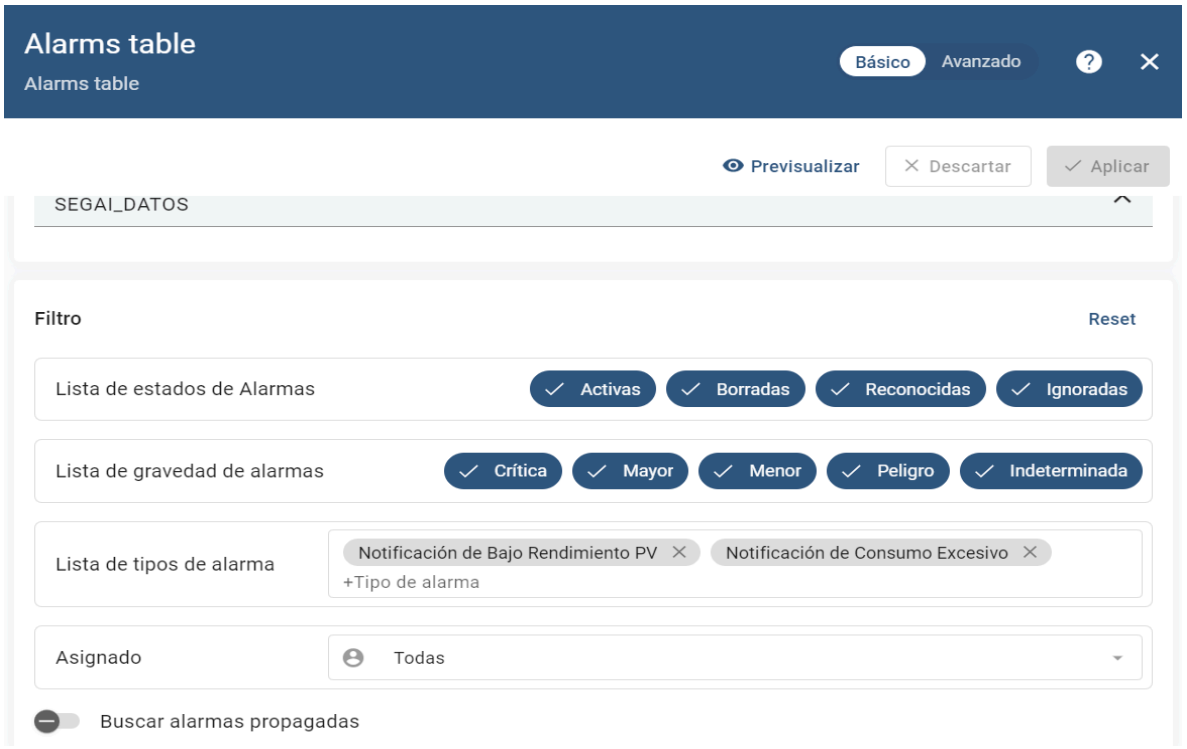


*Figura 3.37: Configuración del widget de alarma.*

### 3.4.4. Asignación de dispositivos y paneles al cliente

Una funcionalidad destacada dentro *ThingsBoard* es la flexibilidad para asignar paneles específicos a clientes individuales. Esta característica permite asignar distintos dispositivos a clientes distintos, facilitando la personalización del acceso y la visualización de datos. Por tanto, es posible crear un panel y asignarlo a una pluralidad de clientes; sin embargo, cada cliente únicamente tendrá acceso a la información de los dispositivos que le han sido asignados garantizando así, que los usuarios solo puedan visualizar sus respectivos dispositivos y los datos asociados.

Para crear un cliente dentro de la plataforma se accede inicialmente a la sección "Clientes" en el panel de control. En el campo correspondiente, se ingresa el nombre asignado al nuevo cliente; para nuestro proyecto este será denominado "Clientes SatComm".

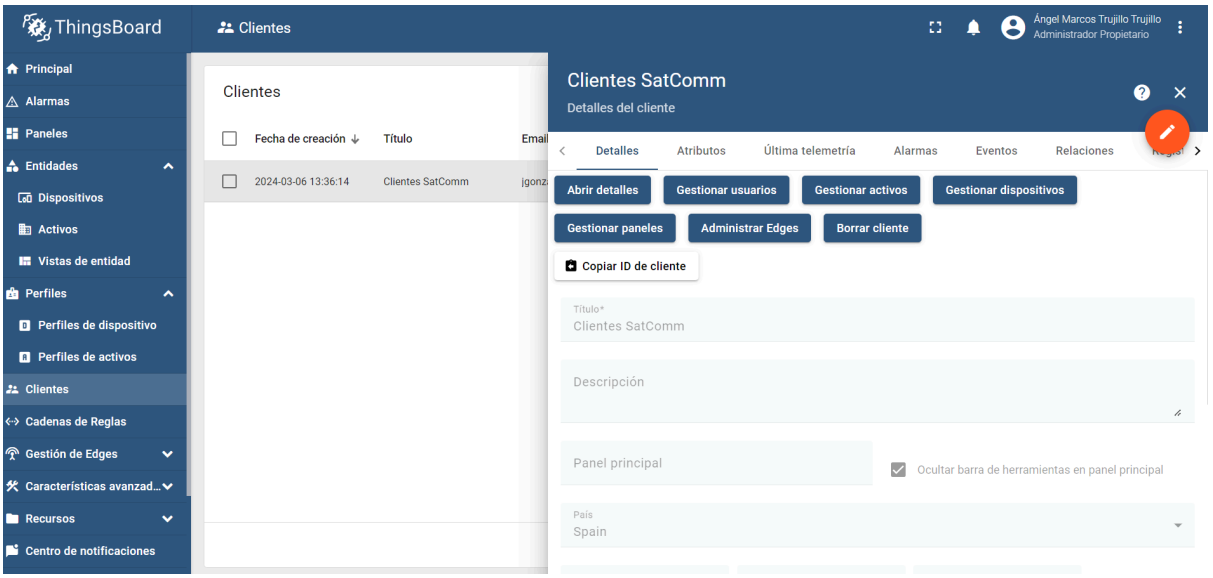


*Figura 3.38: Pantalla de gestión de clientes.*

Además, dentro del cliente "Clientes SatComm", se ha procedido con la creación de usuarios que estarán vinculados exclusivamente a este cliente garantizando que los usuarios asignados puedan acceder y administrar solamente los dispositivos y paneles correspondientes a "Clientes SatComm".

Una vez establecido el cliente, se procedió a su asociación con los distintos dispositivos y paneles. Este proceso de vinculación está visualmente documentado en las Figuras 3.39 y 3.40.

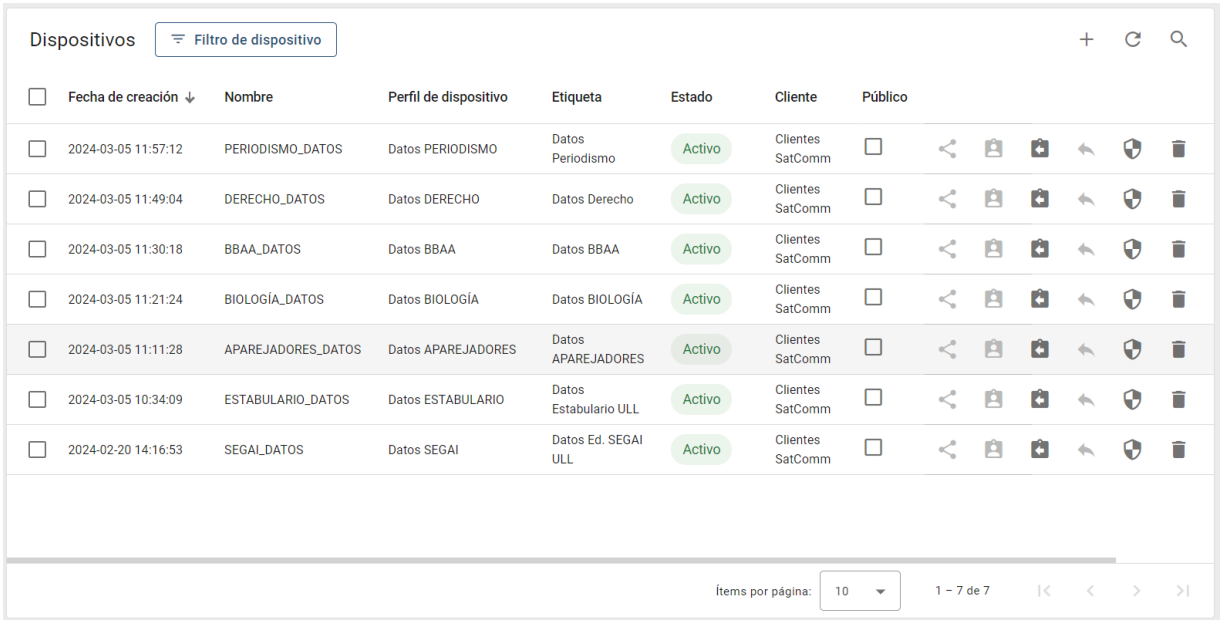


*Figura 3.39: Asignación de dispositivos al cliente SatComm.*

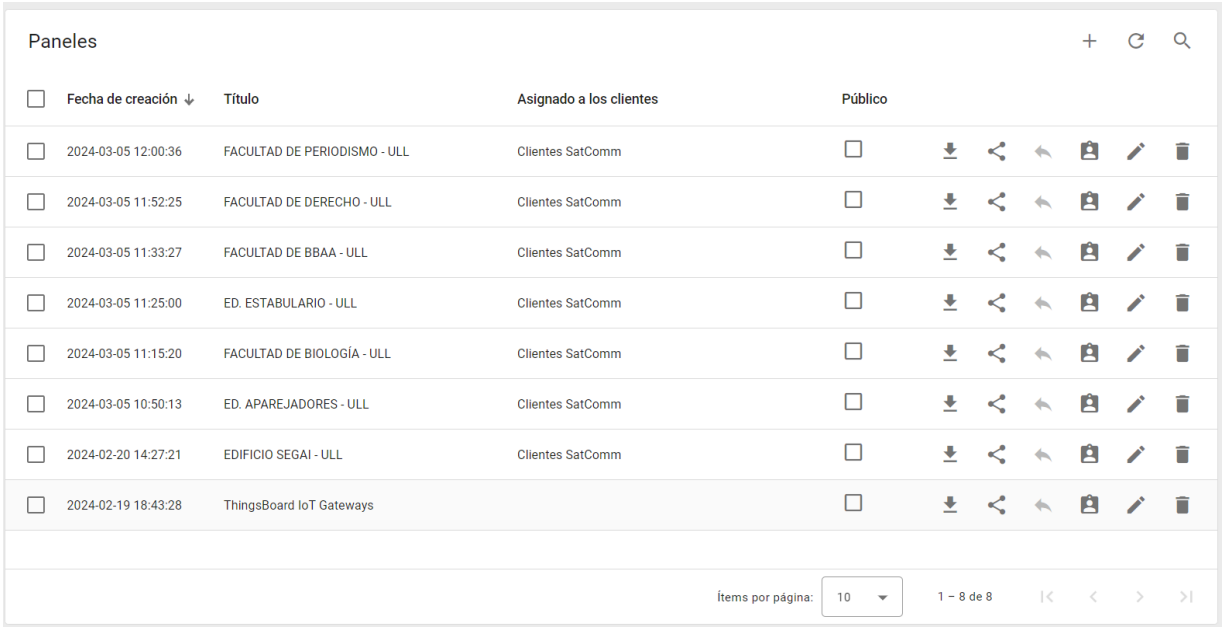


*Figura 3.40: Asociación de paneles al cliente SatComm.*

## 3.5. Predicción de la generación fotovoltaica utilizando redes neuronales LSTM

Atendiendo a las características y ventajas que introducen las redes LSTM (Long Short-Term Memory) en la predicción de series temporales, en este trabajo se propone el entrenamiento de una Red LSTM empleando un enfoque multivariado y multi-step. Este modelo permitirá manejar múltiples variables de entrada para realizar predicciones en varios pasos temporales hacia el futuro. Dentro de la implementación de nuestro modelo LSTM, podemos distinguir dos tipos de variables:

- *Variables a predecir* obtenidas como *salidas* del modelo.
- *Variables predictores (o covariables)* que son las variables de *entrada* al modelo a partir de las cuales se realizan las predicciones.

Para la construcción de este modelo que tiene como objetivo predecir la generación fotovoltaica diaria del Edificio SEGAI, se emplearán los datos que figuran en la Tabla 3.4:

| ***Variable*** | ***Descripción*** | ***Unidad*** | ***Tiempo de muestreo*** |
|---|---|---|---|
| Generación fotovoltaica | Histórico de datos de generación fotovoltaica de un año desde el 04/04/2023 del Edificio SEGAI. | W | 5 minutos |
| Consumo de energía | Histórico de datos de consumo energético de un año desde el 04/04/2023 del Edificio SEGAI. | W | 5 minutos |
| Temperatura ambiente | Histórico de temperatura ambiente desde el 04/04/2023 ofrecidos por la estación meteorológica de La Laguna. | ºC | 30 minutos |
| Irradiancia | Histórico de valores de irradiancia desde el 04/04/2023 ofrecidos por la estación meteorológica de La Laguna. | W/m² | 30 minutos |
| Hora del día | Hora correspondiente del día. | - | - |

*Tabla 3.4: Datos empleados para entrenar el modelo.*


*Figura 3.41: Variables de entrada y salida del modelo planteado.*

### 3.5.1. Recopilación de datos

Para llevar a cabo la predicción de la generación fotovoltaica del Edificio SEGAI, se recopilaron los siguientes datos:

1. *Datos de generación fotovoltaica y consumo de energía.* Estos datos se extrajeron de la web de Sems Portal. Para ello, se desarrolló un código en Python llamado Datos_SEMS_PORTAL.ipynb (para más detalles ver, Apéndice A.2).
2. *Datos meteorológicos.* Se utilizaron datos de la estación meteorológica de La Laguna, ubicada cerca del Edificio SEGAI [31]. Estos datos incluían variables como la temperatura ambiente, la irradiancia, la velocidad del viento y la humedad relativa. Los datos meteorológicos se extrajeron de la página web del Ministerio de Agricultura, Pesca y Alimentación mediante un código en Python llamado Datos_meteorologicos.ipynb (para más detalles ver, Apéndice A.2)

### 3.5.2. Generación del dataset supervisado

#### 3.5.2.1. Procesamiento de los datos

El proceso comenzó etiquetando y uniendo la información de los datos meteorológicos y energéticos en un único archivo a partir del registro horario correspondiente. Como el tiempo de muestreo de los datos energéticos era menor al de los datos meteorológicos (disponibles únicamente cada 30 minutos), se calculó el promedio de las variables energéticas durante periodos de 30 minutos. En la fase de preprocesamiento, además, se eliminaron datos duplicados y se emplearon técnicas de interpolación y promediado adaptados a cada variable para completar valores faltantes entre muestras. La evolución temporal de las variables empleadas se muestran en las Figuras 3.42 a 3.45.

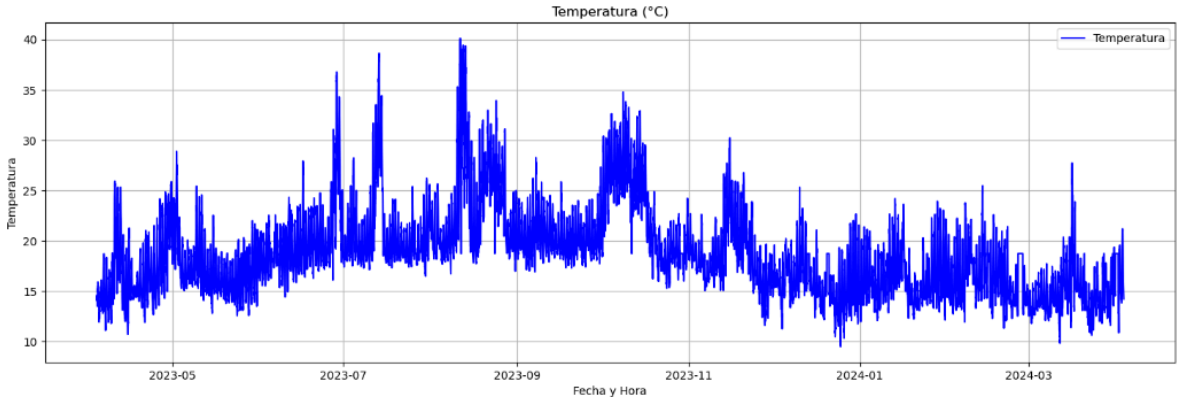


*Figura 3.42: Evolución temporal de la temperatura.*

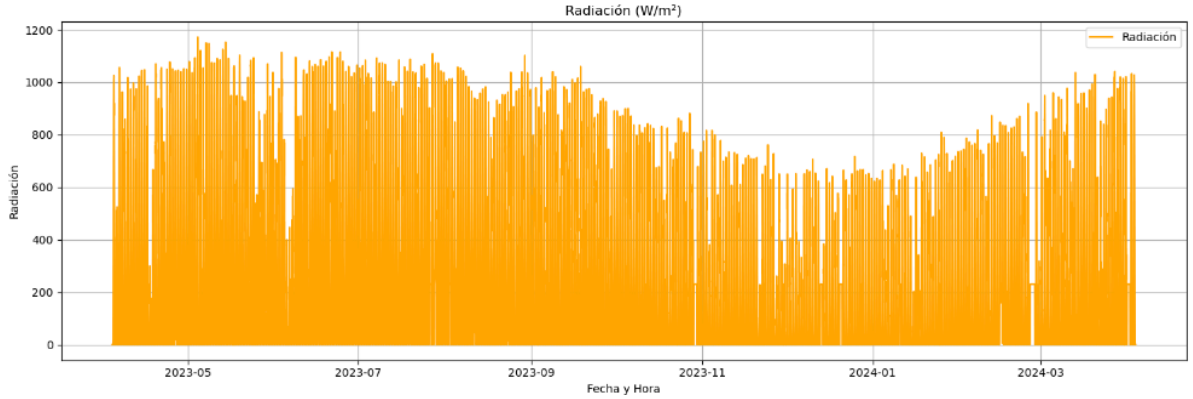


*Figura 3.43: Evolución temporal de la irradiancia.*

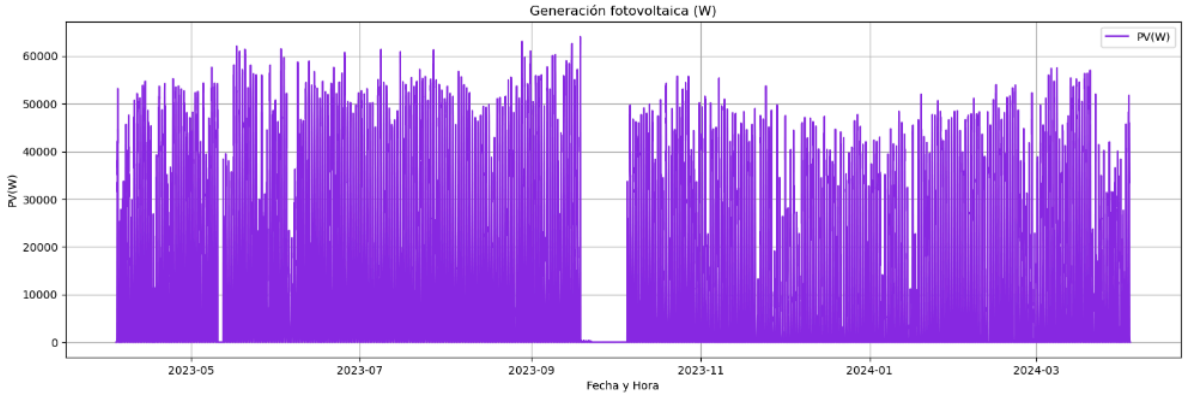


*Figura 3.44: Evolución temporal de la generación fotovoltaica.*

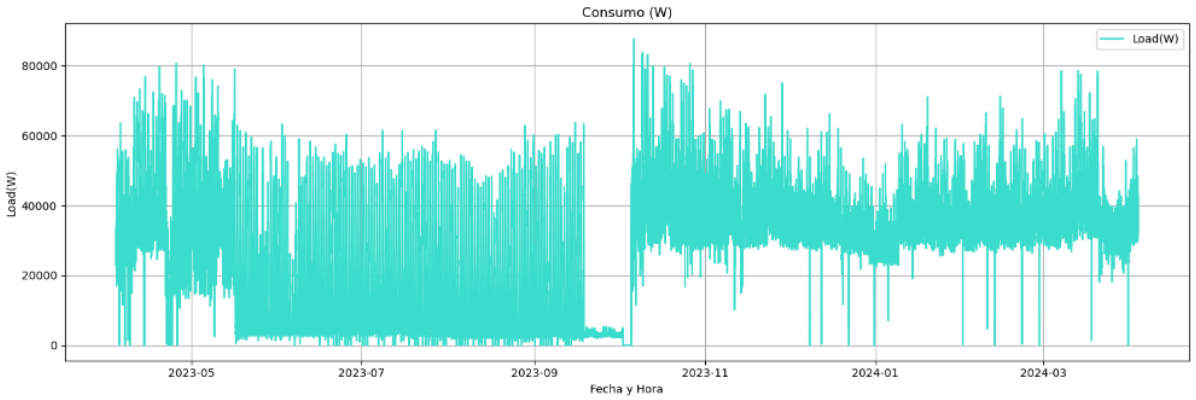


*Figura 3.45: Evolución temporal del consumo de potencia.*

Al analizar las Figuras 3.44 y 3.45, se observa que los datos comprendidos entre el 23 de septiembre de 2023 y el 5 de octubre de 2023 tienen un comportamiento anómalo. Para evitar que esto afecte al entrenamiento del modelo, se ha decidido no incluir estos datos en el dataframe.

Posteriormente, se generó el dataset supervisado adecuado para una Red LSTM en TensorFlow/Keras siguiendo las especificaciones detalladas en la documentación oficial [32]. Cada muestra estaba conformada por la evolución de las cinco variables de entrada en las últimas 72 horas, muestreada cada 30 minutos (5 x 144 datos). Además, cada muestra incluía la evolución de la variable a predecir (potencia generada) en las próximas 24 horas, en intervalos de 30 minutos (48 datos).

Por último, se normalizaron los datos empleando el método StandardScaler de la biblioteca sklearn. Este método ajusta cada variable para que tenga una media de 0 y una desviación estándar de 1 [32]. La estandarización es crucial para el proceso de aprendizaje automático, especialmente en modelos que son sensibles a la magnitud de las variables, como las redes neuronales. Al asegurar que todas las características tengan la misma escala, eliminamos sesgos introducidos por variables con diferentes rangos y magnitudes, lo cual facilita la convergencia del modelo durante el entrenamiento.

### 3.5.2.2. División del conjunto de datos de entrenamiento y validación

Para garantizar una evaluación efectiva de la generalización de nuestro modelo, dividiremos nuestros datos en conjuntos de entrenamiento y validación. A diferencia de otros modelos de Machine Learning, en el caso de series temporales se debe garantizar que se generan las divisiones sin mezclar aleatoriamente los datos.

El conjunto de entrenamiento (*train*) se utilizará para afinar los parámetros del modelo, permitiéndonos aprender las dinámicas subyacentes de los datos. Por otro lado, el conjunto de validación (*val*) servirá para verificar la ausencia de sobreajuste o subajuste (overfitting/underfitting) en el modelo y para el ajuste de sus hiperparámetros.

Para organizar los datos, se han considerado los primeros 22 días de cada mes para el conjunto de entrenamiento. Para asegurar que los datos empleados en validación no han sido incluidos en el entrenamiento, se descartarán los 3 días posteriores a partir del día 22 de cada mes atendiendo a los horizontes de predicción propuestos y a los históricos temporales contemplados en las variables de entrada, Los días restantes del mes serán incluidos en el conjunto de validación.

## 3.5.6. Entrenamiento del Modelo

### 3.5.6.1. Parametrización del modelo

El modelo de predicción se implementó utilizando la biblioteca TensorFlow y la arquitectura Sequential. La arquitectura del modelo se compone de dos capas principales:

1. *Capa LSTM:* esta capa es la responsable de procesar las secuencias de datos de entrada y extraer características relevantes. La cantidad de unidades en la capa LSTM se ajustó durante el entrenamiento para optimizar el rendimiento del modelo.

2. *Capa densa:* esta capa se utiliza para transformar la salida de la capa LSTM en una predicción final. La capa densa tendrá una única neurona con activación lineal, lo que significa que produce una salida escalar que representa la predicción del modelo.

Para optimizar los parámetros del modelo, se evaluaron dos algoritmos de optimización: Adam [33] y RMSprop [34]. Para evitar el sobreentrenamiento, se incorporó un mecanismo de Early Stopping. Este mecanismo finaliza el entrenamiento automáticamente si la pérdida de validación (*val_loss*) no mejora durante un número determinado de épocas [35]. Esto ayuda a prevenir un sobreajuste del modelo a los datos de entrenamiento, asegurando su capacidad de generalizar ante nuevos datos.

Durante el proceso de entrenamiento, se ajustaron diversas variables para optimizar el rendimiento del modelo. La variable principal que se ajustó fue el número de unidades en la capa LSTM. Se experimentó con diferentes configuraciones, tanto con una sola capa LSTM como con configuraciones de doble capa LSTM. A lo largo de las pruebas, se mantuvo constante el valor de la tasa de aprendizaje, el número de épocas y el tamaño del lote. Esto permitió aislar el efecto de los cambios en la arquitectura del modelo sobre su rendimiento. Así, la selección final de la arquitectura y los parámetros del modelo se basó en los resultados obtenidos en el conjunto de validación. Las diferentes configuraciones con las que se realizaron las pruebas se muestran en la Tabla 3.5.

| *Configuración* | *Nº de unidades por capa* |
|---|---|
| Configuración 1 | Capa LSTM de 128 unidades |
| Configuración 2 | Capa LSTM de 64 unidades |
| Configuración 3 | 1ª Capa LSTM de 128 unidades<br>2ª Capa LSTM de 64 unidades |

*Tabla 3.5: Configuraciones empleadas.*

#### 3.5.6.2. Selección de las variables de entrada

Además del conjunto de variables entrada-salida propuesto inicialmente, se llevaron a cabo diversas pruebas para evaluar el rendimiento del modelo en la predicción de generación fotovoltaica al introducir cambios tanto las variables de entrada consideradas como en el histórico temporal evaluado. En concreto, se consideraron los siguientes escenarios alternativos:

Escenario 1

Además de las variables de entrada incluidas en el dataset original, se incorporó información sobre las horas del día con el objetivo de comprender si la inclusión de esta característica contribuía significativamente al rendimiento de las predicciones.

Para incorporar la hora del día en el modelo, se extrae esta información de la columna fecha en nuestro dataframe. Posteriormente, se aplica una codificación cíclica transformando la hora del día en dos dimensiones mediante funciones seno y coseno. Este método se utiliza para preservar la naturaleza cíclica de las horas, permitiendo al modelo reconocer y manejar de manera efectiva las pautas que se repiten diariamente.

Escenario 2

La instalación actual hace que el inversor regule la energía generada por el sistema fotovoltaico en función de la demanda real. Por ello, el dataset original incluía como variable de entrada la información de la demanda del edificio. En este escenario se excluirá el consumo del edificio como variable predictora para analizar el impacto real en la predicción.

Escenario 3

Para analizar el efecto del histórico de la información de las variables de entrada sobre la predicción, en este escenario se propone evaluar el rendimiento del modelo cuando se considera información de las últimas 24 horas y 96 horas.

### 3.5.7. Evaluación de los resultados

Para evaluar el rendimiento del modelo durante el entrenamiento, se utilizó la función de pérdida RMSE (Root Mean Squared Error). Esta función mide el error cuadrático medio entre las predicciones del modelo y los valores reales observados [36].

$$RMSE = \sqrt{\frac{\Sigma i\ (Yi - Yi,pred)^2}{N}} \quad (1)$$

Donde:

- Y: generación fotovoltaica real,
- Y, pred: generación fotovoltaica predicha,
- N: cantidad de predicciones.

El rendimiento de la red se evaluó utilizando tres métodos. El primer método analiza el valor de pérdida utilizando los conjuntos de entrenamiento y evaluación. Para ello, se ha empleado la función *evaluate* de Keras. También, se evaluó el modelo empleando gráficas de pérdidas que muestran el error durante cada época de entrenamiento, permitiendo detectar si el modelo ha aprendido adecuadamente los patrones y si generaliza bien a nuevos datos. Por otro lado, se emplearon histogramas para el análisis de la distribución de los errores absolutos.

# 4. Resultados

En este capítulo, se presentan los distintos resultados obtenidos. Se expondrán los cuadros de mando implementados para cada uno de los edificios, así como los resultados del modelo de predicción desarrollado y su correspondencia con la generación real.

## 4.1. Monitorización energética en los edificios de La Universidad de La Laguna

Se ha implementado un sistema de monitorización para las principales variables eléctricas en siete edificios de la Universidad de La Laguna. Este sistema cubre un total de 520 kW instalados y se caracteriza por incluir los componentes descritos en el apartado 2.1.2. La lista completa de variables monitorizadas en cada uno de los edificios se detalla en la Tabla 4.1.

| ***Variable*** | ***Unidad*** | ***Tiempo de muestreo*** |
|---|---|---|
| Consumo energético | kW | 5 minutos |
| Generación fotovoltaica | kW | 5 minutos |

*Tabla 4.1: Variables monitorizadas.*

En las Figuras 4.3 a 4.9 se presentan los cuadros de mando creados para los diferentes edificios. Estos cuadros disponen de los seis widgets mencionados en secciones anteriores. A la izquierda de la pantalla se puede acceder a la información de los datos extraídos de cada planta. En el centro, se muestran dos gráficos distintos que monitorizan la generación fotovoltaica y el consumo energético del edificio a lo largo del tiempo. A la derecha, se sitúan dos gráficos adicionales que muestran la información del consumo y la generación fotovoltaica actual del edificio seleccionado.

Además, aunque no se observa en las figuras, en la parte inferior de cada uno de los paneles se encuentra el widget de alarma, cuya forma se ilustra en la Figura 4.1.

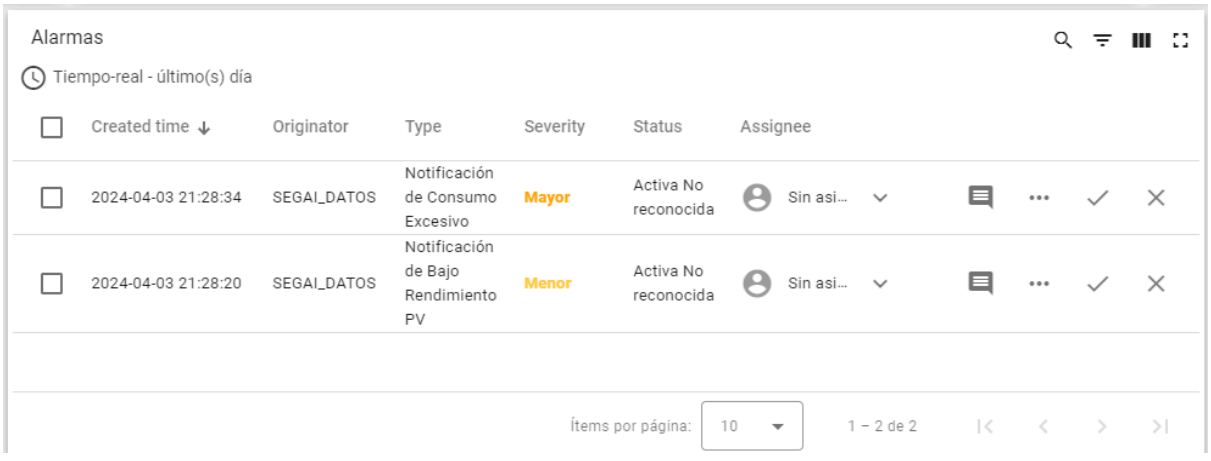


*Figura 4.1: Widget de alarmas que dispone cada panel de mando.*

Por otro lado, debido al reciente cambio horario en España, fue necesario ajustar el código para adaptar los envíos de datos a ThingsBoard. Esta modificación se debió a la diferencia de configuración de la zona horaria entre el servidor que aloja *ThingsBoard* y la

zona horaria de Tenerife. Para solucionar esto, se realizó un ajuste en el código restando una hora:

```
# Ajuste de hora debido al cambio de horario, en segundos
ajuste_horario_segundos = -3600  # Esto restará una hora

# Y luego, al calcular el timestamp ajustado:
timestamp_ajustado = (pd.to_datetime(row['Time']) + timedelta(seconds=ajuste_horario_segundos)).timestamp() * 1000
```

*Figura 4.2: Ajuste Horario.*

Por último, es importante destacar que la plataforma *ThingsBoard* realiza un promedio cada cinco minutos de los datos de telemetría proporcionados por cada dispositivo. Esto significa que las cifras reportadas en cada uno de los paneles de mando no coincidirán exactamente con las mediciones ofrecidas por el sitio web de Sems Portal variando por unos pocos vatios.


*Figura 4.3: Cuadro de mando de la Facultad de Periodismo.*

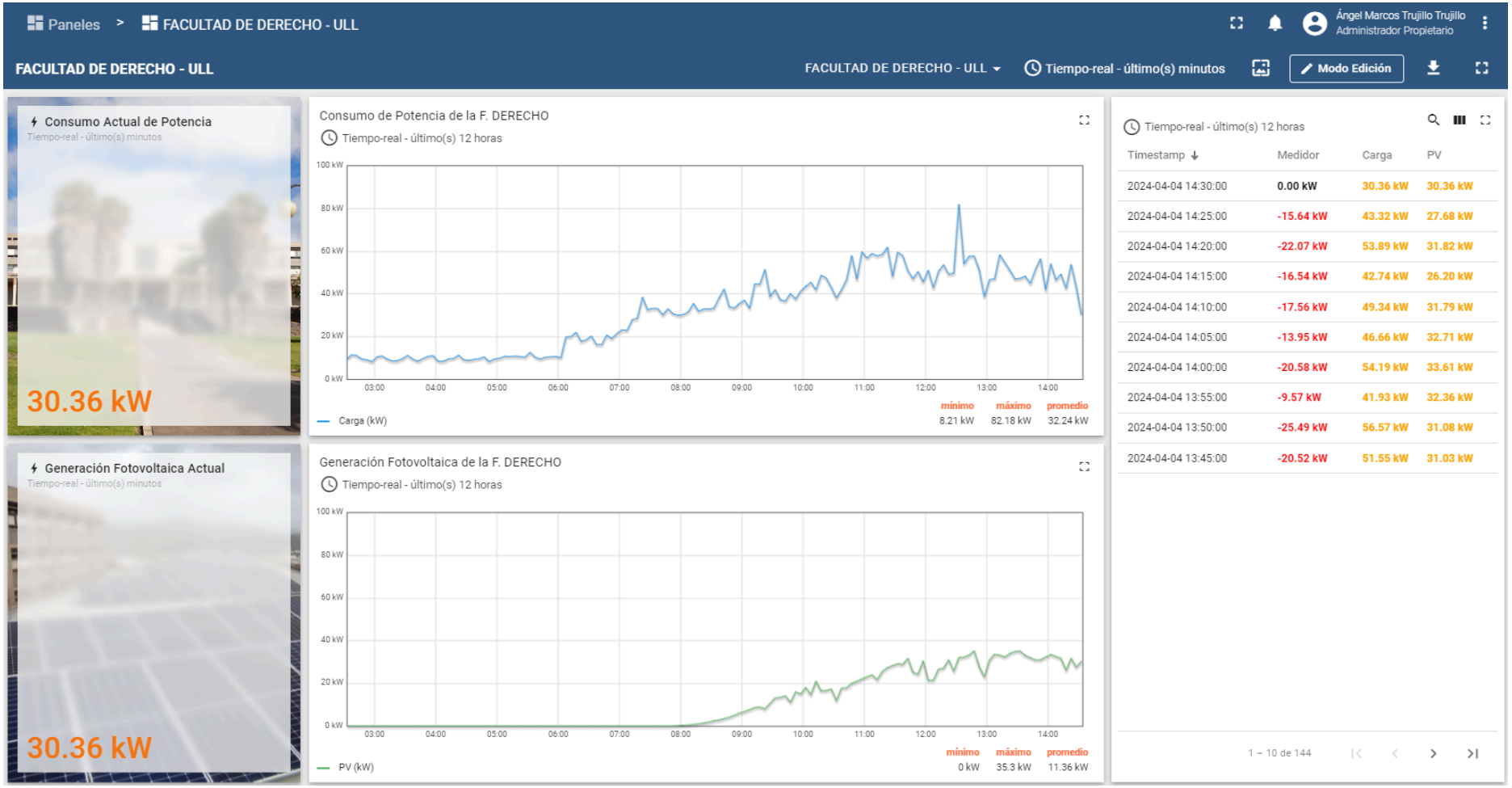


*Figura 4.4: Cuadro de mando de la Facultad de Derecho.*

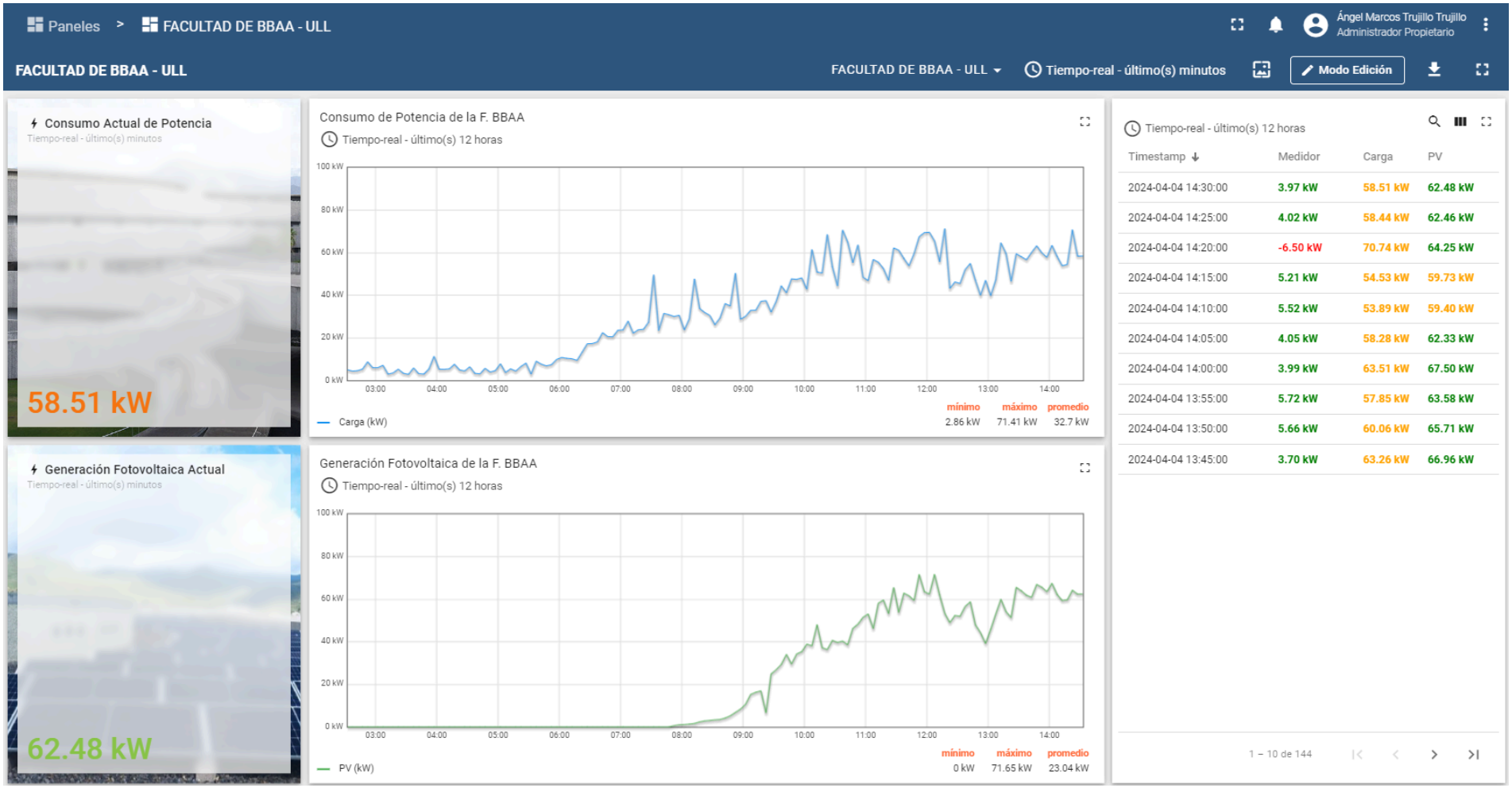


*Figura 4.5: Cuadro de mando de la Facultad de Bellas Artes.*

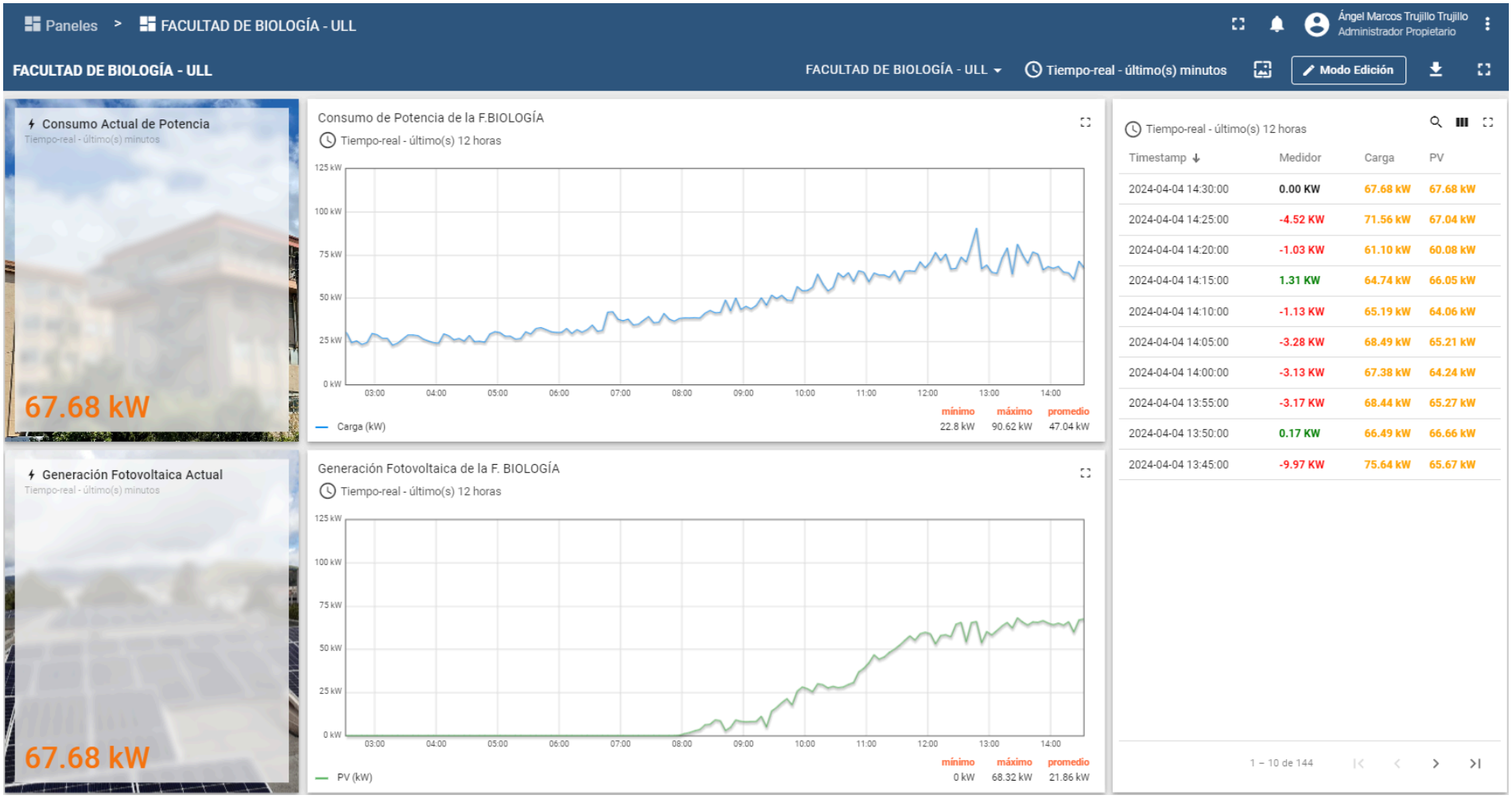


*Figura 4.6: Cuadro de mando de la Facultad de Biología.*

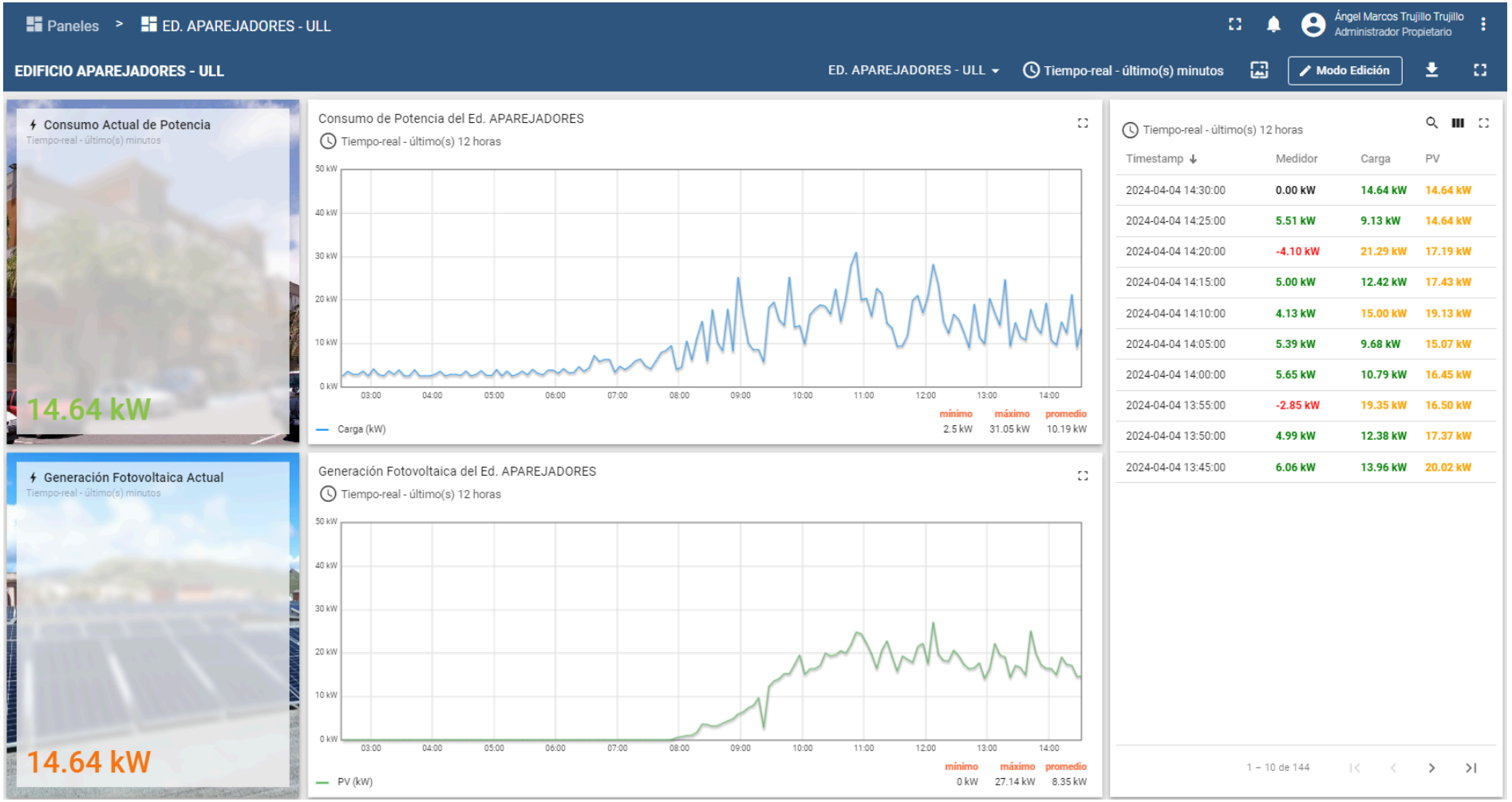


*Figura 4.7: Cuadro de mando de la Escuela Politécnica Superior de Ingeniería.*

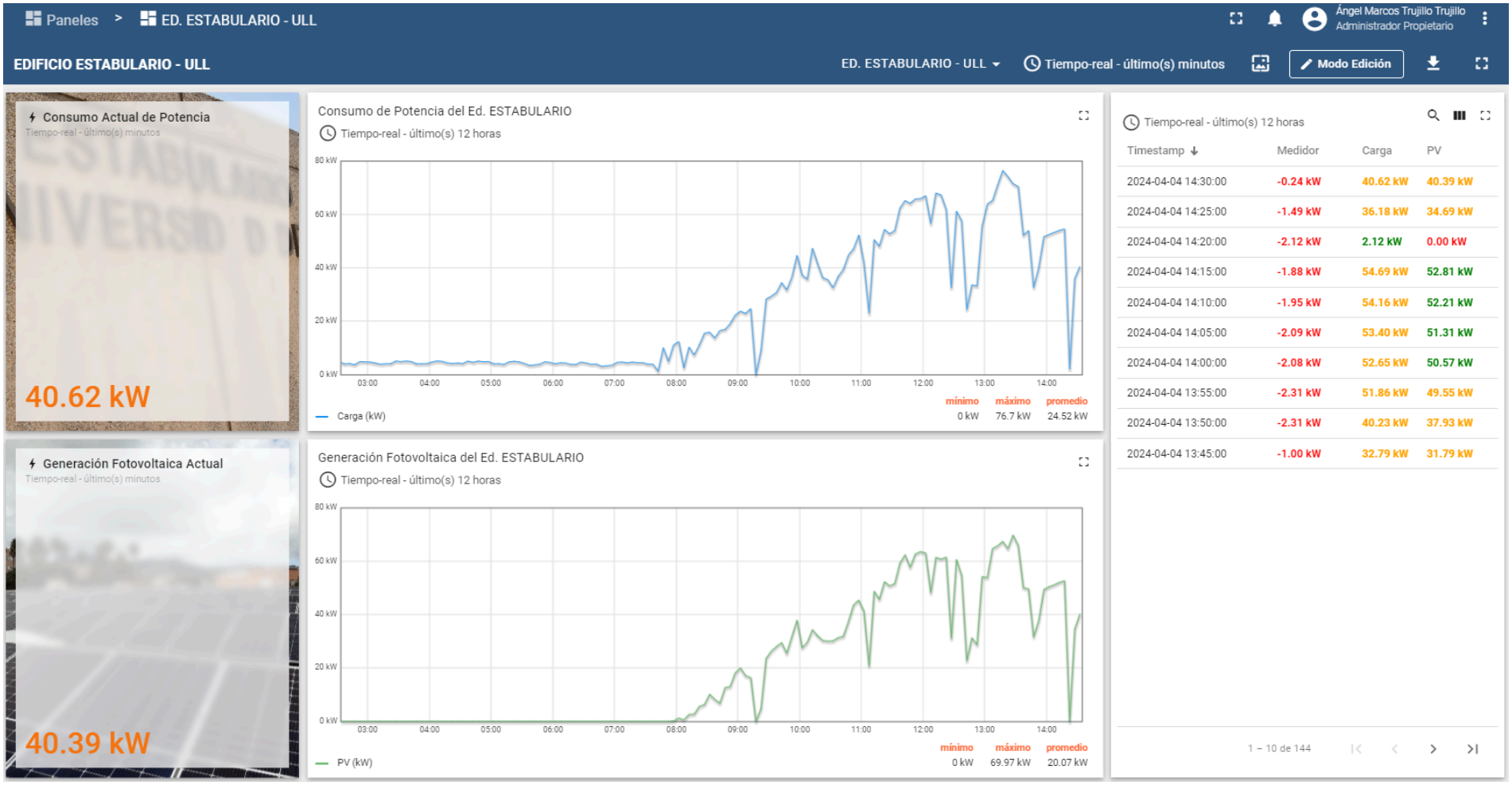


*Figura 4.8: Cuadro de mando del Edificio Estabulario.*

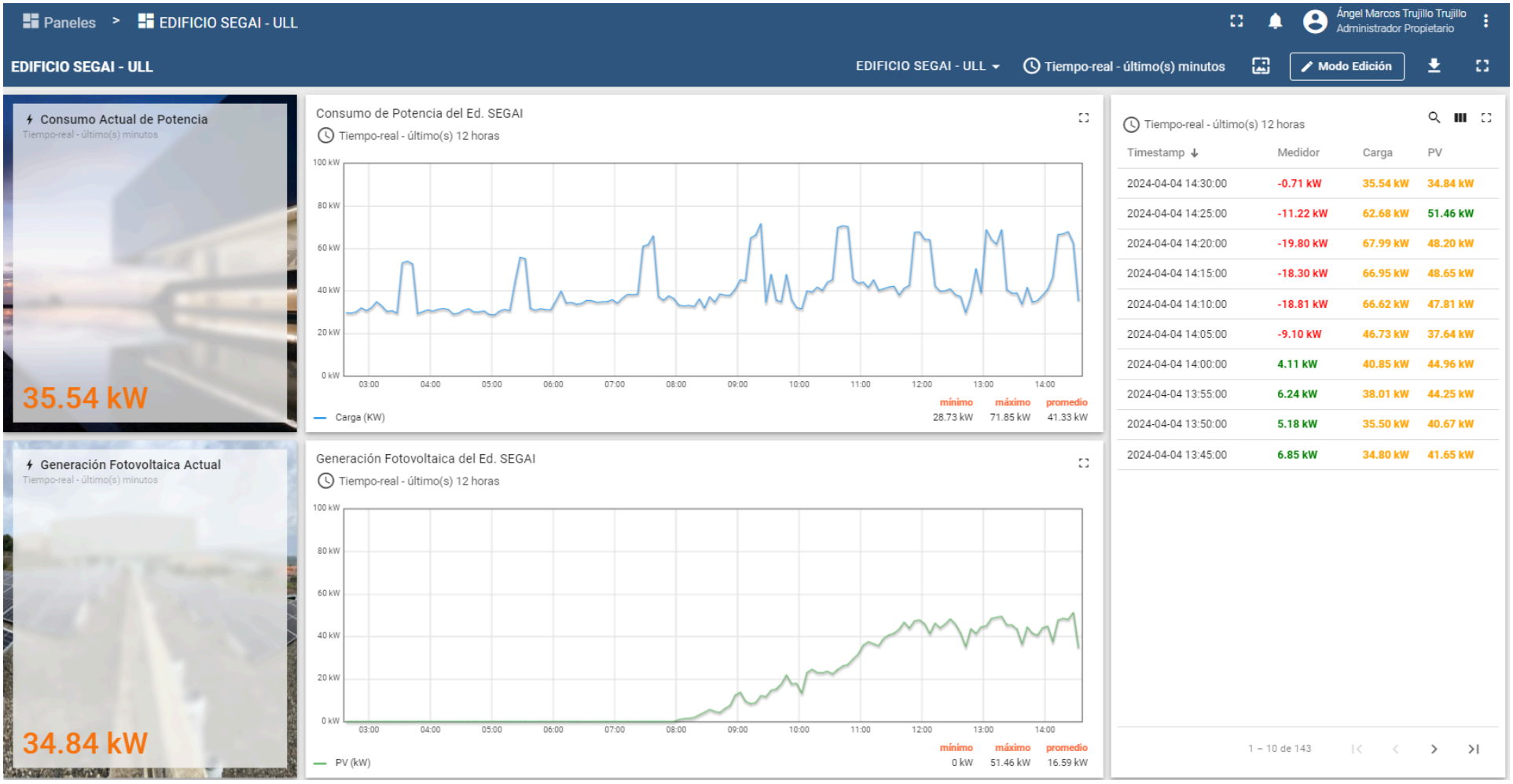


*Figura 4.9: Cuadro de mando del Edificio SEGAI.*

## 4.2. Evaluación del modelo LSTM en la predicción de generación fotovoltaica

En este apartado se presentarán y compararán los resultados de las diversas pruebas realizadas. En primer lugar, se analizará el efecto del optimizador empleado y el ajuste de los hiperparámetros en la predicción del modelo. Por último, se analizará el rendimiento del modelo en función de los distintos escenarios propuestos en la sección 3.5.6.2.

### 4.2.1. Búsqueda del mejor optimizador

Primero, se evaluó el rendimiento de los optimizadores Adam y RMSprop bajo los mismos parámetros para determinar cuál proporciona mejores resultados de predicción. Para esto, entrenamos un modelo caracterizado con los parámetros descritos en la Tabla 4.2.

| ***Parámetro del modelo*** | ***Valor*** |
|---|---|
| Ratio de aprendizaje | 0.0005 |
| Epochs | 60 |
| Batch size | 256 |
| Nº de capas LSTM | 1 |
| Nº de unidades capa LSTM | 128 |
| Patience | 15 |

*Tabla 4.2: Parámetros del modelo.*

Se analizó la evolución del entrenamiento del modelo a través de las pérdidas en los conjuntos de entrenamiento (*train loss*) y validación (*validation loss*) a lo largo de las diferentes épocas, tal y como se muestra en la Figura 4.10 y 4.11.

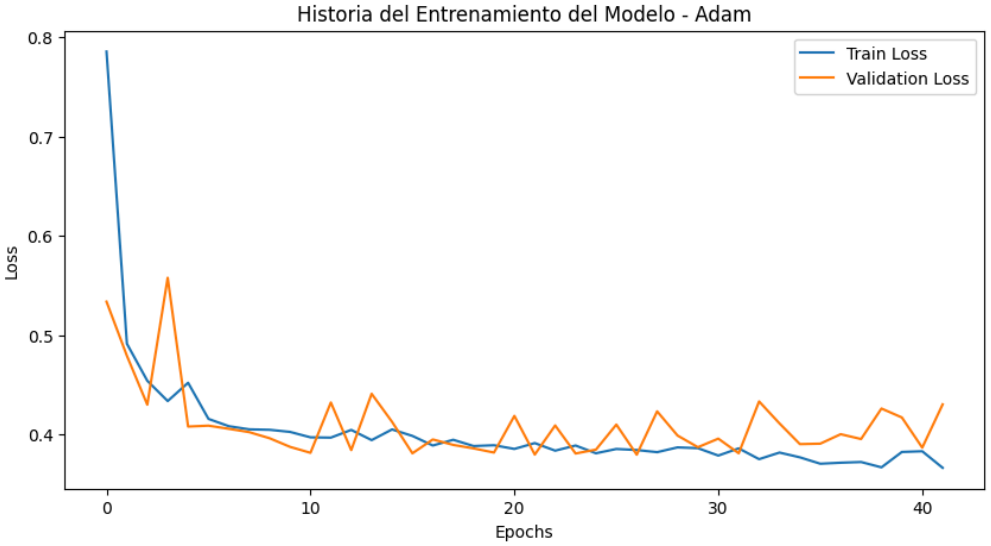


*Figura 4.10: Evolución del entrenamiento con el optimizador Adam.*

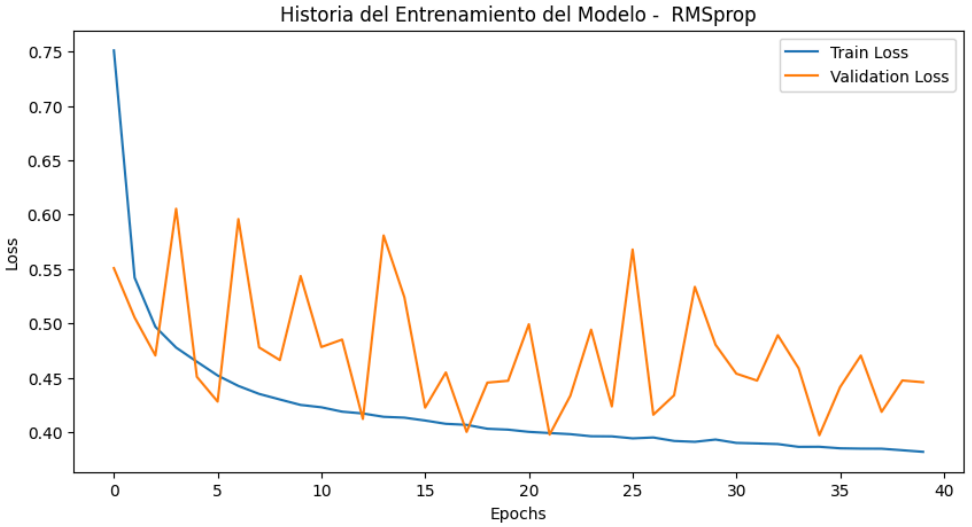


*Figura 4.11: Evolución del entrenamiento con el optimizador RMSprop.*

En ambas figuras, el descenso inicial de la pérdida de entrenamiento es similar, aunque la pérdida de validación con Adam muestra menos variabilidad y una tendencia más estable a lo largo de las épocas en comparación con RMSprop. La Tabla 4.3 muestra el error RMSE obtenido tanto en validación como en entrenamiento para cada optimizador.

| | | ***RMSE*** |
|---|---|---|
| *Adam* | Set de entrenamiento | 0.341 |
| | Set de validación | 0.361 |
| *RMSprop* | Set de entrenamiento | 0.410 |
| | Set de validación | 0.424 |

*Tabla 4.3: Comparación de RMSE para los optimizadores Adam y RMSprop.*

Tras evaluar el desempeño de los dos optimizadores, se ha decidido seleccionar Adam como el optimizador principal, ya que el Error Cuadrático Medio (RMSE) es menor. Los resultados obtenidos muestran que Adam alcanza un RMSE de 0.341 en el conjunto de entrenamiento y de 0.361 en el conjunto de validación, siendo estos valores inferiores a los obtenidos con RMSprop, donde el RMSE fue de 0.410 en entrenamiento y de 0.424 en validación.

### 4.2.2. Ajuste de hiperparámetros

Se exploraron diversas configuraciones de la arquitectura de la red para evaluar su impacto en el rendimiento del modelo. Inicialmente, utilizamos una configuración con una sola capa que contiene 128 unidades. Posteriormente, se procedió a ajustar este parámetro disminuyendo el número de unidades para observar cómo afecta al rendimiento del modelo. Además, se experimentó con una arquitectura más compleja, añadiendo una segunda capa LSTM de 64 unidades con el objetivo de determinar si una mayor profundidad mejoraría la capacidad predictiva del modelo.

Los resultados de cada uno de los entrenamientos se reflejan en la Tabla 4.4.

| | | ***RMSE*** |
|---|---|---|
| *Configuración 1* | Set de entrenamiento | 0.341 |
| | Set de validación | 0.361 |
| *Configuración 2* | Set de entrenamiento | 0.347 |
| | Set de validación | 0.365 |
| *Configuración 3* | Set de entrenamiento | 0.341 |
| | Set de validación | 0.366 |

*Tabla 4.4: Comparación de RMSE para las diferentes configuraciones.*

Como se observa en la Tabla 4.4, con una capa LSTM de 128 unidades se obtuvo un RMSE para el conjunto de entrenamiento de 0.341 y de 0.361 para el conjunto de validación.

Al disminuir la capa a 68 unidades, el modelo presenta un desempeño ligeramente peor tanto en el conjunto de entrenamiento como en el de validación.

Al añadir una segunda capa LSTM, se observa que el RMSE de entrenamiento no varía con respecto a la primera configuración. Esto indica que la capacidad del modelo para ajustarse a los datos de entrenamiento no se ha visto afectada por el aumento en la complejidad del modelo. Sin embargo, el RMSE de validación ha aumentado ligeramente, de 0.361 a 0.366 con la nueva configuración de dos capas.

Por tanto, tras evaluar las diferentes configuraciones del modelo, se ha decidido optar por la configuración inicial con una sola capa LSTM de 128 unidades. Esta decisión se basa en que esta configuración ofreció un equilibrio óptimo entre complejidad y rendimiento, demostrando ser suficiente para alcanzar los resultados deseados sin la necesidad de incrementar la complejidad con capas adicionales que no proporcionaron mejoras significativas en el RMSE de validación.

### 4.2.3. Evaluación de las variables de entrada

En esta fase se llevaron a cabo diversas pruebas para evaluar el rendimiento del modelo en función de los escenarios planteados en la sección 3.5.6.2.

#### 4.2.3.1. Escenario 1

Los errores RMSE de estas pruebas durante el entrenamiento del modelo se reflejan en la Tabla 4.5.

| | | ***RMSE*** |
|---|---|---|
| *Sin incluir la hora* | Set de entrenamiento | 0.341 |
| | Set de validación | 0.361 |
| *Incluyendo la hora* | Set de entrenamiento | 0.332 |
| | Set de validación | 0.360 |

*Tabla 4.5: Comparación de RMSE con la introducción de la hora como variable de entrada.*

A la vista de los resultados, la incorporación de la hora del día como variable predictiva ha tenido un efecto positivo en el modelo, reflejando una mejora en el RMSE del conjunto de entrenamiento, que disminuyó de 0.341 a 0.332. En el conjunto de validación, no se observaron cambios en el RMSE, lo que indica que, aunque la adición de la hora aporta cierta mejora en fase de entrenamiento, no altera significativamente el rendimiento en nuevos datos.

#### 4.2.3.2. Escenario 2

Para evaluar la relevancia del consumo como variable de entrada y su impacto en la adaptación del sistema fotovoltaico frente a un sistema antivertidos que impide que la generación exceda el consumo, se entrenó el modelo excluyendo esta característica. Esta aproximación permite determinar la importancia del consumo en la predicción de la generación fotovoltaica. Los resultados obtenidos se muestran en la Tabla 4.6.

| | | ***RMSE*** |
|---|---|---|
| *Con consumo* | Set de entrenamiento | 0.341 |
| | Set de validación | 0.361 |
| *Sin consumo* | Set de entrenamiento | 0.342 |
| | Set de validación | 0.362 |

*Tabla 4.6: Comparación de RMSE sin la introducción del consumo como variable de entrada.*

Pese a que el sistema presenta un sistema antivertido que afecta a la generación de energía fotovoltaica de la instalación en función del consumo, los resultados muestran que al excluir el consumo de energía como variable de entrada en nuestro modelo el RMSE de entrenamiento aumentó ligeramente de 0.341 a 0.342 y el de validación de 0.361 a 0.362. Estos cambios sugieren que, bajo las condiciones actuales, el consumo no desempeña un papel crucial en el rendimiento del modelo para predecir la generación fotovoltaica.

#### 4.2.3.3. Escenario 3

La Tabla 4.7 muestra los resultados obtenidos al entrenar y validar el modelo considerando información temporal correspondiente a las 24, 72 y 96 horas previas a la predicción.

| *Historial de datos de entrada* | | *RMSE* |
|---|---|---|
| *3 días* | Set de entrenamiento | 0.341 |
| | Set de validación | 0.361 |
| *1 día* | Set de entrenamiento | 0.348 |
| | Set de validación | 0.361 |
| *4 días* | Set de entrenamiento | 0.355 |
| | Set de validación | 0.362 |

*Tabla 4.7: Comparación de RMSE para diferentes longitudes de entrada.*

La variación en los resultados muestra que el modelo mantiene un desempeño consistente en términos de RMSE a través de diferentes configuraciones de historiales de datos de entrada, con pequeñas diferencias en los valores obtenidos. Específicamente, el aplicar una longitud de entrada de 3 días proporciona los mejores resultados tanto en entrenamiento como en validación.

### 4.2.4. Propuesta final del modelo

Basado en las pruebas realizadas anteriormente, se ha entrenado un modelo que incluye la información presentada en la Figura 4.12.

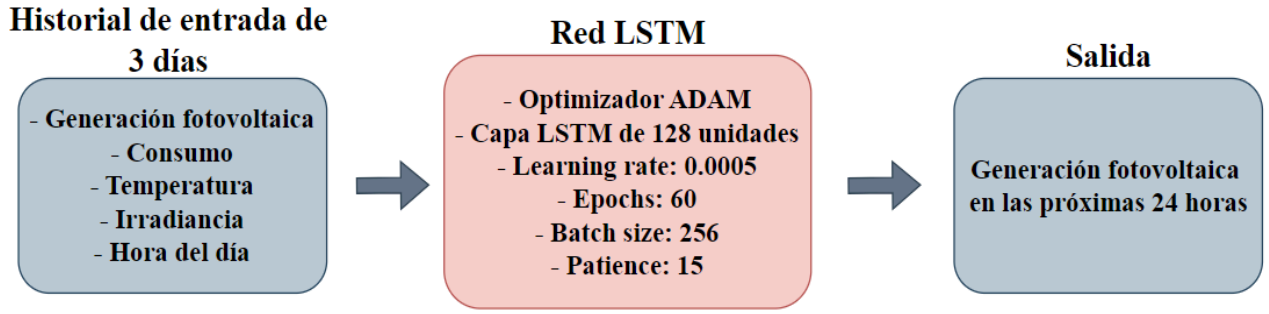


*Figura 4.12: Parametrización del mejor modelo.*

En esta sección, se ha evaluado el rendimiento de dicho modelo al realizar predicciones sobre el conjunto de datos de validación. En primer lugar, la Figura 4.13 muestra el histograma de errores de predicción . Este histograma ilustra la distribución de los errores en términos absolutos entre las predicciones realizadas y los valores reales del conjunto de validación. En éste se refleja una distribución centrada en un valor próximo a cero. La presencia de picos simétricos alrededor del centro indica que, en general, los errores son pequeños, con algunos valores atípicos en ambos extremos de la distribución.

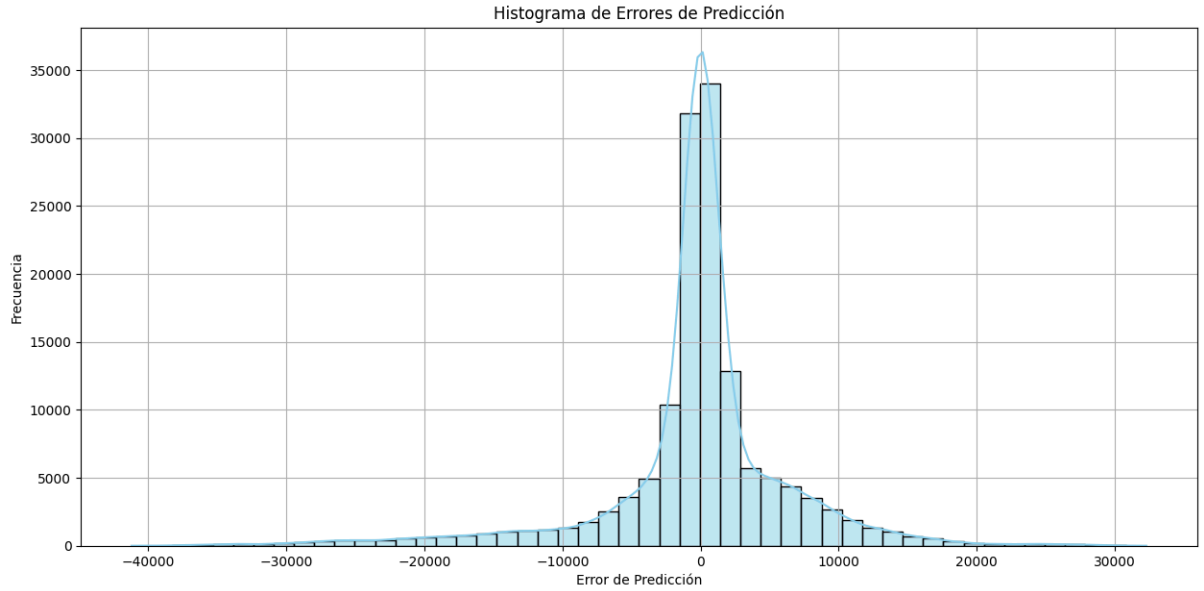


*Figura 4.13: Histograma de errores de predicción.*

Las Figuras 4.14 y 4.15 muestran un ejemplo de las predicciones realizadas por el modelo para dos días diferentes. Las gráficas indican que las predicciones son bastante similares a los valores reales medidos, demostrando así la capacidad del modelo para replicar la generación fotovoltaica en distintas condiciones.

Por otro lado, en ambas figuras se observa que en las predicciones hay valores menores que 0. Estos podrían corregirse posteriormente mediante post-procesamiento de los datos, haciendo que las predicciones con valores negativos se establezcan en 0. Este ajuste garantizaría que los resultados sean más coherentes y realistas.

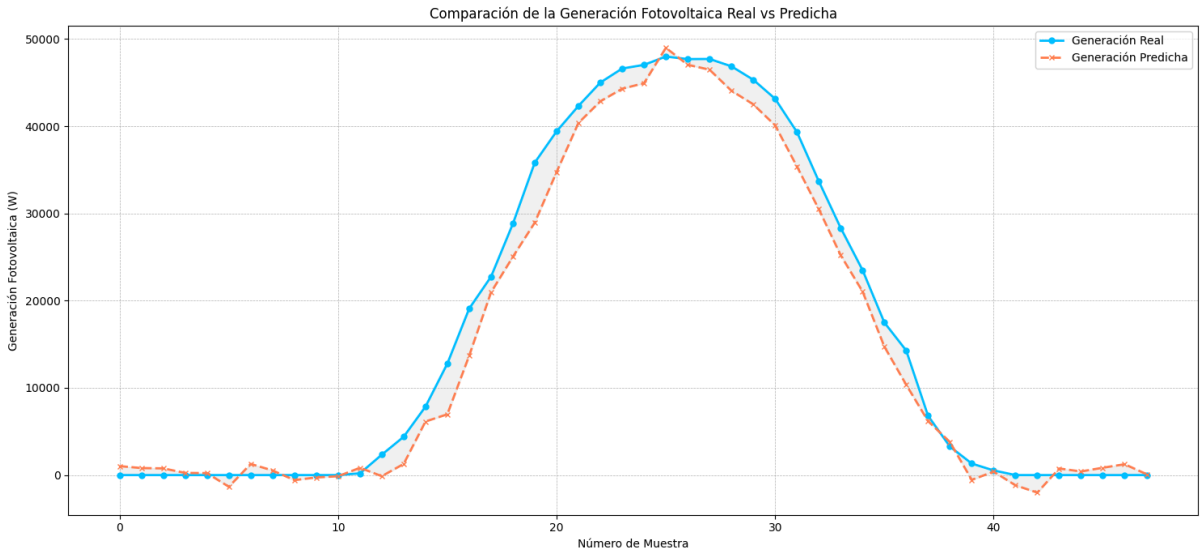


*Figura 4.14: Comparación de la generación real frente a la predicha para el 27 de junio de 2023.*

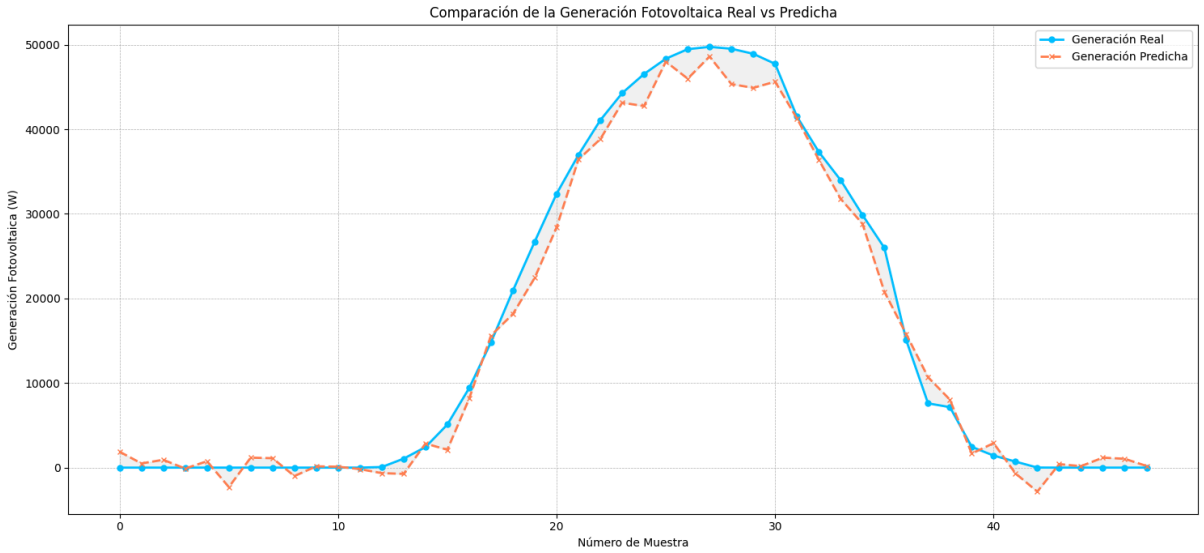


*Figura 4.15: Comparación de la generación real frente a la predicha para el 28 de junio de 2023.*

Además, se realizó una comparación entre la generación fotovoltaica real y la predicha, ejecutando el modelo tres veces para obtener información de tres días consecutivos. Este análisis permitió observar de manera detallada cómo se comporta el modelo durante periodos extendidos, proporcionando una visión de la precisión en la predicción de la generación fotovoltaica a lo largo del tiempo.

Como se puede observar en las Figuras 4.16 y 4.17, el modelo predice la tendencia general de generación fotovoltaica, ajustándose a la curva de generación real. Sin embargo, la Figura 4.17 muestra que el modelo incurre en mayores errores en situaciones donde hay cambios abruptos en la generación. Esto podría deberse a cambios repentinos en las condiciones meteorológicas difícilmente predecibles, como el paso de nubes. Este efecto podría intentar minimizarse en versiones futuras del modelo si se incorporara, además, información de la predicción de las variables meteorológicas durante el periodo objetivo.

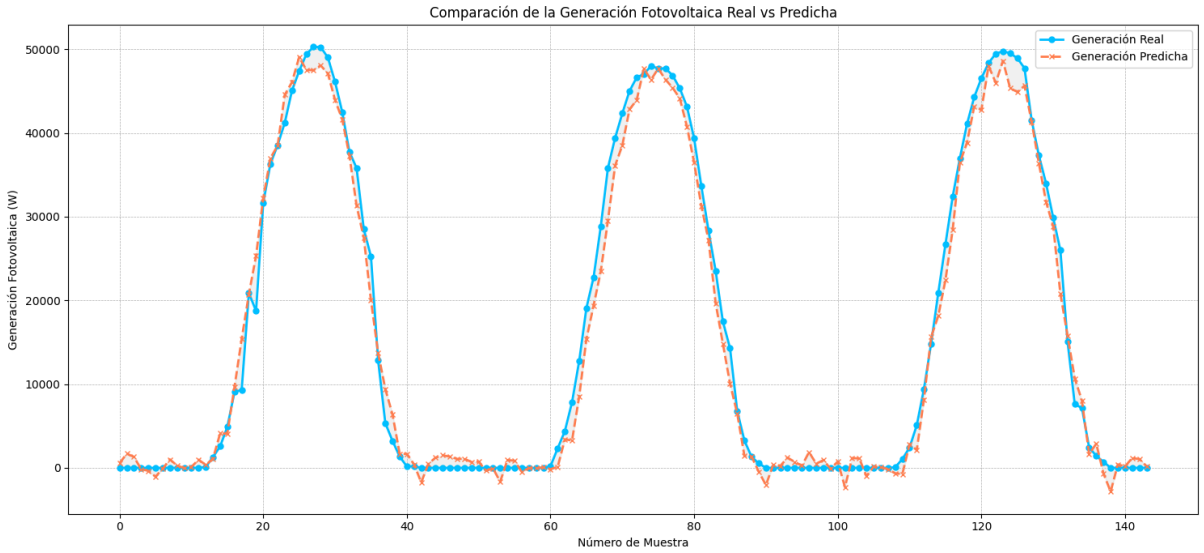


*Figura 4.16: Comparación de la generación real frente a la predicha para los días 26, 27 y 28 de junio de 2023.*

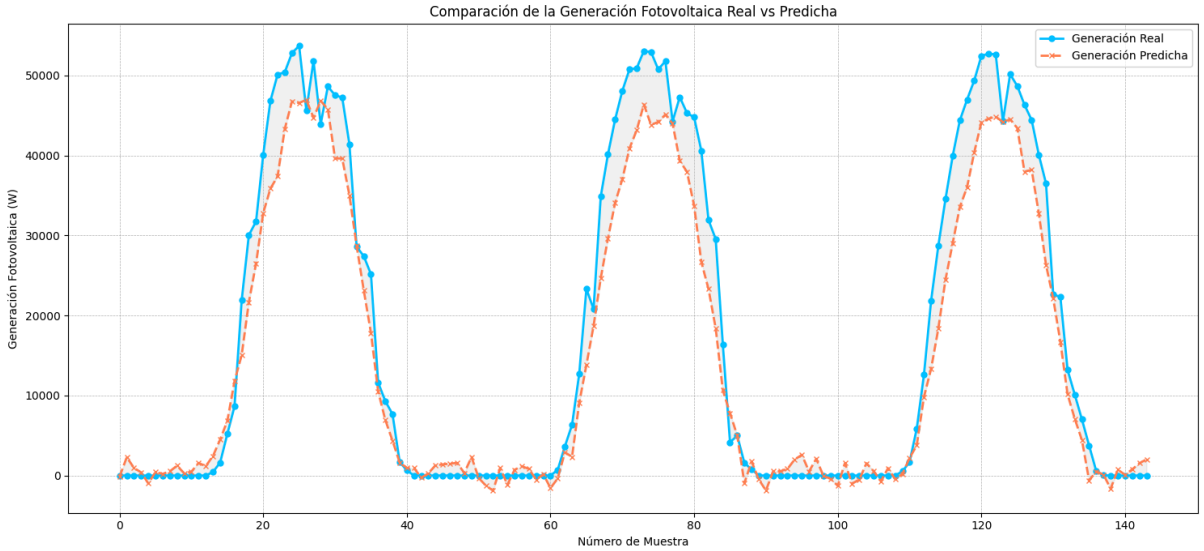


*Figura 4.17: Comparación de la generación real frente a la predicha para los días 26, 27 y 28 de abril de 2023.*

# 5. Conclusiones y líneas futuras

En este capítulo se presentan las conclusiones derivadas del proyecto, así como las líneas futuras de investigación y desarrollo que se podrían explorar.

## 5.1. Conclusiones

La monitorización de variables energéticas ha ganado importancia en la gestión eficiente de recursos energéticos, especialmente en contextos donde la demanda energética es significativa. Integrar esta monitorización en plataformas IoT no solo optimiza el consumo energético sino que también facilita la toma de decisiones basada en datos. Dentro de este campo, este trabajo aporta contribuciones significativas en dos áreas principales: el diseño de un sistema de monitorización centralizado para ocho edificios de la Universidad de La Laguna y el desarrollo de un modelo predictivo para anticipar la generación fotovoltaica, destacando la relevancia de estas iniciativas para promover la sostenibilidad y la eficiencia energética, aspectos cruciales en un ámbito donde abordar estos desafíos continúa siendo un reto considerable. Las principales conclusiones de este trabajo se presentan a continuación:

- Se implementaron con éxito técnicas de Automatización Robótica de Procesos (RPA) utilizando scripts de Python para automatizar la recogida de datos. Esta automatización sustituye las funciones que tradicionalmente realizaría un humano, optimizando el proceso y reduciendo la posibilidad de error, lo que permite una gestión de datos más eficiente y efectiva.
- Los datos recolectados de cada planta fotovoltaica en los edificios se integraron eficazmente en la plataforma IoT ThingsBoard. Esta incorporación nos ha permitido profundizar en el entendimiento del funcionamiento de estas plataformas, que se están utilizando cada vez más debido a las numerosas ventajas que ofrecen en diversas aplicaciones.
- Se desarrollaron los diferentes cuadros de mando que facilitaron la visualización de la información sobre la generación fotovoltaica y el consumo energético de cada edificio. Estas herramientas permitieron un acceso y análisis más eficiente de los datos, mejorando la gestión y el seguimiento del rendimiento energético.
- Se desarrolló un modelo predictivo utilizando redes LSTM para la estimación de la generación fotovoltaica en las próximas 24 horas. Para ello, se han analizado diferentes configuraciones para la optimización de su rendimiento. Pese a que el modelo ha mostrado algunas dificultades ante la predicción de cambios repentinos en la generación, en este trabajo se ha evidenciado la capacidad del modelo para la realización de predicciones de generación de forma adecuada. Estos avances establecen una base sólida para futuras investigaciones y ajustes en el modelo.

## 5.2. Líneas futuras

Los resultados y conclusiones obtenidos en este proyecto sientan las bases para futuras investigaciones dirigidas a desarrollar un sistema de monitorización integral para todos los edificios de la Universidad de La Laguna equipados con instalaciones fotovoltaicas. Asimismo, motivan la búsqueda de un modelo que prediga con mayor precisión la generación

fotovoltaica, maximizando así la eficiencia y efectividad de la gestión energética. A continuación, se presentan algunas posibles líneas de mejora para el futuro:

- Extensión de la implementación de sistemas de monitorización a más edificios dentro de la Universidad de La Laguna.
- Implementar analizadores de redes, como el Circutor CVM-E3-MINI-ITF-WiEth, que permitiría obtener información directa y precisa de la planta fotovoltaica sin depender de los datos de inversores y contadores, que a menudo pueden entregar valores erróneos o alterados. Esta tecnología aprovecha la conectividad Wi-Fi, eliminando la necesidad de cableado de comunicaciones, y permitiría actualizar los datos continuamente en la plataforma IoT con mayor frecuencia y exactitud, mejorando así, significativamente el sistema de monitorización y permitiéndonos estudiar mejor cómo gestionar la energía.
- Otra mejora significativa sería integrar soluciones de almacenamiento de energía para optimizar el consumo energético. Esta integración permitiría una gestión más eficiente de los picos de demanda y suministro, almacenando energía durante periodos de baja demanda y liberándola cuando sea necesario, maximizando así el uso de la energía generada, aunque para ello, primero sería necesario legalizar cada uno de los edificios que disponen de un sistema antivertido.
- Una mejora adicional para el modelo de predicción sería la búsqueda de una mayor cantidad de datos para utilizar en la fase de entrenamiento. Dado que la instalación fotovoltaica del edificio SEGAI es reciente y sólo se disponía de datos desde 2023, la incorporación de nuevos datos históricos mejoraría significativamente la precisión y la robustez del modelo predictivo.
- Explorar el uso de otras redes neuronales, como las GRU (Gated Recurrent Units), para evaluar su impacto comparado con las LSTM. Además, sería beneficioso experimentar con otras configuraciones de hiperparámetros para optimizar más el rendimiento del modelo predictivo.
- Añadir como entrada las predicciones de la irradiancia, la principal variable para predecir la generación fotovoltaica. También sería beneficioso instalar una estación meteorológica en el edificio SEGAI, permitiendo acceder a datos meteorológicos propios de la instalación. Esto no solo facilita la inclusión de la irradiancia, sino también otras variables como la velocidad del viento, enriqueciendo así el aprendizaje del modelo.
- Desarrollar el modelo para predecir la generación fotovoltaica en situaciones donde no se exista sistema antivertidos. Esto permitiría estimar los valores potenciales de generación fotovoltaica sin las limitaciones impuestas por dicho sistema, ofreciendo una visión más completa de la capacidad productiva real del sistema fotovoltaico instalado en el edificio.

En general, el desarrollo de estas futuras líneas de trabajo representará un significativo avance para el presente proyecto, mejorando la eficacia y la amplitud del sistema de monitorización. Además, permitirá contar con un modelo más avanzado que genere predicciones más precisas sobre la generación fotovoltaica.

# 6. Conclusions and future research

This chapter presents the conclusions drawn from the project, as well as future lines of research and development that could be explored.

## 6.1. Conclusions

The monitoring of energy variables has gained importance in the efficient management of energy resources, especially in contexts where energy demand is significant. Integrating this monitoring into IoT platforms not only optimizes energy consumption but also facilitates data-driven decision-making. Within this field, this work have done significant contributions in two main areas: the design of a centralized monitoring system for eight buildings of the University of La Laguna and the development of a predictive model to anticipate photovoltaic generation, highlighting the relevance of these initiatives to promote sustainability and energy efficiency, crucial aspects in a field where addressing these challenges remains a considerable challenge. The main conclusions of this work are presented below:

- Robotic Process Automation (RPA) techniques were successfully implemented using Python scripting to automate data collection. This automation replaces functions that would traditionally be performed by a human, optimizing the process and reducing the possibility of error, allowing for more efficient and effective data management.
- The data collected from each PV plant in the buildings was efficiently integrated into the IoT ThingsBoard platform. This incorporation has allowed us to deepen our understanding of the operation of these platforms, which are increasingly being used due to the many advantages they offer in various applications.
- The different dashboards were developed to facilitate the visualization of information on the photovoltaic generation and energy consumption of each building. These tools enabled more efficient access and analysis of data, improving the management and monitoring of energy performance.
- A predictive model was developed using LSTM networks for the estimation of PV generation in the next 24 hours. For this purpose, different configurations have been analyzed for performance optimization. Although the model has shown some difficulties in predicting sudden changes in generation, this work has demonstrated the model's ability to adequately predict generation. This progress establishes a solid basis for future research and adjustments to the model.

## 6.2. Future research

The results and conclusions obtained in this project lay the foundations for future research aimed at developing a comprehensive monitoring system for all the buildings of the University of La Laguna equipped with photovoltaic installations. They also motivate the search for a model that more accurately predicts photovoltaic generation, thus maximizing the efficiency and effectiveness of energy management. Some possible lines of improvement for the future are presented below:

- Extending the implementation of monitoring systems to more buildings within the University of La Laguna.
- Implementing network analysers, such as the Circutor CVM-E3-MINI-ITF-WiEth, which would allow direct and accurate information to be obtained from the photovoltaic plant without relying on data from inverters and meters, which can often deliver erroneous or altered values. This technology leverages Wi-Fi connectivity, eliminating the need for communications cabling, and would allow data to be continuously updated on the IoT platform with greater frequency and accuracy, thus significantly improving the monitoring system and allowing us to better study how to manage energy.
- Another significant improvement would be to integrate energy storage solutions to optimize energy consumption. This integration would allow more efficient management of peak demand and supply, storing energy during periods of low demand and releasing it when needed, thus maximizing the use of the energy generated, although this would first require legalization of each of the buildings that have an anti-discharge system.
- A further improvement to the prediction model would be to find more data to use in the training phase. Since the photovoltaic installation of the SEGAI building is recent and data was only available from 2023, the incorporation of new historical data would significantly improve the accuracy and robustness of the predictive model.
- Explore the use of other neural networks, such as GRU (Gated Recurrent Units), to assess their impact compared to LSTMs. In addition, it would be beneficial to experiment with other hyperparameter configurations to further optimize the performance of the predictive model.
- Add as input the irradiance predictions, the main variable for predicting PV generation. It would also be beneficial to install a weather station in the SEGAI building, allowing access to the facility's own weather data. This not only facilitates the inclusion of irradiance, but also other variables such as wind speed, thus enriching the learning of the model.
- Develop the model to predict photovoltaic generation in situations where there is no anti-spill system. This would allow estimating the potential PV generation values without the limitations imposed by such a system, offering a more complete view of the real productive capacity of the PV system installed on the building.

In general terms, the development of these future lines of work will represent a significant advance for the present project, improving the efficiency and comprehensiveness of the monitoring system. It will also provide a more advanced model to generate more accurate predictions of PV generation.

# 7. Presupuesto

En este capítulo se detalla el presupuesto necesario para la implementación del sistema de monitorización energética y para el desarrollo de un modelo predictivo utilizando técnicas de aprendizaje automático. Se presentarán estimaciones detalladas de los costos asociados con cada componente del proyecto, incluyendo tanto los recursos materiales como humanos requeridos para su ejecución.

# Cuadro de Precios Descompuestos

| Nº | Código | Ud | Descripción | | | Total |
|---|---|---|---|---|---|---|
| | | | ***1 DESARROLLO DE SISTEMA DE MONITORIZACIÓN*** | | | |
| 1.1 | 1.1. | **Ud** | **Implementación y fusión de soluciones tecnológicas avanzadas para la recolección de datos de las diversas instalaciones. Este proceso se llevará a cabo mediante técnicas de WebScraping y Automatización Robótica de Procesos (RPA), con el objetivo de obtener la información necesaria de cada una de las plantas.** | | | |
| | M01B0070 | 17,500 h | Ingeniero Industrial | 25,490 | | 446,08 |
| | | 3,000 % | Costes indirectos | 446,080 | | 13,38 |
| | | | **Precio total por Ud .....** | | | **459,46** |
| | | | **Son cuatrocientos cincuenta y nueve Euros con cuarenta y seis céntimos** | | | |
| 1.2 | 1.2. | **Ud** | **Incorporación y sincronización de los datos recopilados de cada planta en la plataforma de Internet de las Cosas (IoT) ThingsBoard. Además, se registrará y almacenará los históricos de datos correspondientes a cada una de las plantas.** | | | |
| | M01B0070 | 7,500 h | Ingeniero Industrial | 25,490 | | 191,18 |
| | 40001194_030 | 1,000 Ud | Servidor IaaS (Infrastructure as a Servi… | 125,000 | | 125,00 |
| | | 3,000 % | Costes indirectos | 316,180 | | 9,49 |
| | | | **Precio total por Ud .....** | | | **325,67** |
| | | | **Son trescientos veinticinco Euros con sesenta y siete céntimos** | | | |
| 1.3 | 1.3. | **Ud** | **Desarrollo de cuadros de mando para la recogida y visualización de la información generada dentro de la plataforma IoT ThingsBoard.** | | | |
| | M01B0070 | 5,000 h | Ingeniero Industrial | 25,490 | | 127,45 |
| | | 3,000 % | Costes indirectos | 127,450 | | 3,82 |
| | | | **Precio total por Ud .....** | | | **131,27** |
| | | | **Son ciento treinta y un Euros con veintisiete céntimos** | | | |
| 1.4 | 1.4. | **Ud** | **Elaboración de la documentación completa y detallada que cubra todas las fases del desarrollo del sistema de monitorización, desde la concepción inicial hasta la implementación final** | | | |
| | M01B0070 | 75,000 h | Ingeniero Industrial | 25,490 | | 1.911,75 |
| | | 3,000 % | Costes indirectos | 1.911,750 | | 57,35 |
| | | | **Precio total por Ud .....** | | | **1.969,10** |
| | | | **Son mil novecientos sesenta y nueve Euros con diez céntimos** | | | |

| Nº | Código | Ud | Descripción | | | Total |
|---|---|---|---|---|---|---|
| | | | ***2 MODELO DE PREDICCIÓN DE LA GENERACIÓN FOTO…*** | | | |
| 2.1 | 2.1. | **Ud** | **Proceso de recopilar datos meteorológicos y energéticos en el Edificio Segai para entrenar posteriormente a la red neuronal** | | | |
| | M01B0070 | 50,000 | h | Ingeniero Industrial | 25,490 | 1.274,50 |
| | | 3,000 | % | Costes indirectos | 1.274,500 | 38,24 |
| | | | | **Precio total por Ud .....** | | **1.312,74** |
| | | | | **Son mil trescientos doce Euros con setenta y cuatro céntimos** | | |
| 2.2 | 2.2. | **Ud** | **La creación del modelo involucra el diseño de la arquitectura del modelo predictivo, seleccionando técnicas y algoritmos, como las redes neuronales LSTM, y configurando capas, neuronas, funciones de activación y parámetros iniciales.**<br><br>**La evaluación del modelo se realizará usando un conjunto de datos de prueba no visto durante el entrenamiento, para medir su precisión y capacidad predictiva, validando así su efectividad y realizando ajustes necesarios para optimizar su rendimiento.** | | | |
| | BG6Z2111 | 1,000 | u | Ordenador | 650,000 | 650,00 |
| | M01B0070 | 50,000 | h | Ingeniero Industrial | 25,490 | 1.274,50 |
| | | 3,000 | % | Costes indirectos | 1.924,500 | 57,74 |
| | | | | **Precio total por Ud .....** | | **1.982,24** |
| | | | | **Son mil novecientos ochenta y dos Euros con veinticuatro céntimos** | | |
| 2.3 | 2.3. | **Ud** | **Elaboración de la documentación completa y detallada que cubra todas las fases del desarrollo del modelo predictivo, desde la concepción inicial hasta la implementación final** | | | |
| | M01B0070 | 25,000 | h | Ingeniero Industrial | 25,490 | 637,25 |
| | | 3,000 | % | Costes indirectos | 637,250 | 19,12 |
| | | | | **Precio total por Ud .....** | | **656,37** |
| | | | | **Son seiscientos cincuenta y seis Euros con treinta y siete céntimos** | | |

PRESUPUESTO Y MEDICION

PRESUPUESTO PARCIAL Nº 1 DESARROLLO DE SISTEMA DE MONITORIZACIÓN

| Nº | DESCRIPCION | UDS. | LARGO | ANCHO | ALTO | CANTIDAD | PRECIO | IMPORTE |
|---|---|---|---|---|---|---|---|---|
| 1.1 | **Ud. Implementación y fusión de soluciones tecnológicas avanzadas para la recolección de datos de las diversas instalaciones. Este proceso se llevará a cabo mediante técnicas de WebScraping y Automatización Robótica de Procesos (RPA), con el objetivo de obtener la información necesaria de cada una de las plantas.** | | | | | | | |
| | Edificio Segai | 1 | | | | 1,000 | | |
| | Edificio Estabulario | 1 | | | | 1,000 | | |
| | Escuela Politécnica Superior de Ingeniería (EPSI) | 1 | | | | 1,000 | | |
| | Facultad de Biología | 1 | | | | 1,000 | | |
| | Facultad de Derecho | 1 | | | | 1,000 | | |
| | Facultad de Bellas Artes | 1 | | | | 1,000 | | |
| | Facultad de Ciencias Políticas, Sociales y de la Comunicación | 1 | | | | 1,000 | | |
| | | | | | | 7,000 | 459,46 | 3.216,22 |
| 1.2 | **Ud. Incorporación y sincronización de los datos recopilados de cada planta en la plataforma de Internet de las Cosas (IoT) ThingsBoard. Además, se registrará y almacenará los históricos de datos correspondientes a cada una de las plantas.** | | | | | | | |
| | Edificio Segai | 1 | | | | 1,000 | | |
| | Edificio Estabulario | 1 | | | | 1,000 | | |
| | Escuela Politécnica Superior de Ingeniería (EPSI) | 1 | | | | 1,000 | | |
| | Facultad de Biología | 1 | | | | 1,000 | | |
| | Facultad de Derecho | 1 | | | | 1,000 | | |
| | Facultad de Bellas Artes | 1 | | | | 1,000 | | |
| | Facultad de Ciencias Políticas, Sociales y de la Comunicación | 1 | | | | 1,000 | | |
| | | | | | | 7,000 | 325,67 | 2.279,69 |
| 1.3 | **Ud. Desarrollo de cuadros de mando para la recogida y visualización de la información generada dentro de la plataforma IoT ThingsBoard.** | | | | | | | |
| | Edificio Segai | 1 | | | | 1,000 | | |
| | Edificio Estabulario | 1 | | | | 1,000 | | |
| | Escuela Politécnica Superior de Ingeniería (EPSI) | 1 | | | | 1,000 | | |
| | Facultad de Biología | 1 | | | | 1,000 | | |
| | Facultad de Derecho | 1 | | | | 1,000 | | |
| | Facultad de Bellas Artes | 1 | | | | 1,000 | | |
| | Facultad de Ciencias Políticas, Sociales y de la Comunicación | 1 | | | | 1,000 | | |
| | | | | | | 7,000 | 131,27 | 918,89 |
| 1.4 | **Ud. Elaboración de la documentación completa y detallada que cubra todas las fases del desarrollo del sistema de monitorización, desde la concepción inicial hasta la implementación final** | | | | | | | |
| | | | | | | 1,000 | 1.969,10 | 1.969,10 |

Total presupuesto parcial nº 1 ... 8.383,90

PRESUPUESTO PARCIAL Nº 2 MODELO DE PREDICCIÓN DE LA GENERACIÓN FOTOVOLTAICA

| Nº | DESCRIPCION | UDS. | LARGO | ANCHO | ALTO | CANTIDAD | PRECIO | IMPORTE |
|---|---|---|---|---|---|---|---|---|
| 2.1 | **Ud. Proceso de recopilar datos meteorológicos y energéticos en el Edificio Segai para entrenar posteriormente a la red neuronal** | | | | | | | |
| | Edificio Segai | 1 | | | | 1,000 | | |
| | | | | | | 1,000 | 1.312,74 | 1.312,74 |
| 2.2 | **Ud. La creación del modelo involucra el diseño de la arquitectura del modelo predictivo, seleccionando técnicas y algoritmos, como las redes neuronales LSTM, y configurando capas, neuronas, funciones de activación y parámetros iniciales.**<br><br>**La evaluación del modelo se realizará usando un conjunto de datos de prueba no visto durante el entrenamiento, para medir su precisión y capacidad predictiva, validando así su efectividad y realizando ajustes necesarios para optimizar su rendimiento.** | | | | | | | |
| | Edificio Segai | 1 | | | | 1,000 | | |
| | | | | | | 1,000 | 1.982,24 | 1.982,24 |
| 2.3 | **Ud. Elaboración de la documentación completa y detallada que cubra todas las fases del desarrollo del modelo predictivo, desde la concepción inicial hasta la implementación final** | | | | | | | |
| | | | | | | 1,000 | 656,37 | 656,37 |

Total presupuesto parcial nº 2 ... 3.951,35

## RESUMEN POR CAPITULOS

| | |
|---|---|
| CAPITULO DESARROLLO DE SISTEMA DE MONITORIZACIÓN | 8.383,90 |
| CAPITULO MODELO DE PREDICCIÓN DE LA GENERACIÓN FOTOVOLT… | 3.951,35 |
| REDONDEO..... | |
| PRESUPUESTO DE EJECUCION MATERIAL..... | 12.335,25 |

EL PRESUPUESTO DE EJECUCION MATERIAL ASCIENDE A LAS EXPRESADAS DOCE MIL TRESCIENTOS TREINTA Y CINCO EUROS CON VEINTICINCO CÉNTIMOS.

Proyecto: DESARROLLO DE SISTEMA DE MONITORIZACIÓN Y MODELO PREDICTIVO

| Capítulo | Importe |
|---|---|
| Capítulo 1 DESARROLLO DE SISTEMA DE MONITORIZACIÓN | 8.383,90 |
| Capítulo 2 MODELO DE PREDICCIÓN DE LA GENERACIÓN FOTOVOLTAICA | 3.951,35 |
| Presupuesto de ejecución material | 12.335,25 |
| 16% de gastos generales | 1.973,64 |
| 6% de beneficio industrial | 740,12 |
| Suma | 15.049,01 |
| 7% IGIC | 1.053,43 |
| Presupuesto de ejecución por contrata | 16.102,44 |

Asciende el presupuesto de ejecución por contrata a la expresada cantidad de DIECISEIS MIL CIENTO DOS EUROS CON CUARENTA Y CUATRO CÉNTIMOS.

# Bibliografía

[1] International Energy Agency. (2023). IEA – International Energy Agency. Recuperado de https://www.iea.org/

[2] REN21. (2023). Building the sustainable energy future with renewable energy. Recuperado de https://www.ren21.net/

[3] Gobierno de Canarias. (2023). Plan de Transición Energética de Canarias PTECan-2030. Recuperado de https://www.gobiernodecanarias.org/energia/materias/planificacion/

[4] Ahmed, S., Ali, A., & D'Angola, A. (2023). A Review of Renewable Energy Communities: Concepts, Scope, Progress, Challenges, and Recommendations. Sustainability.

[5] Instituto para la Diversificación y Ahorro de la Energía (IDAE). (2023). Comunidades Energéticas. Recuperado de https://www.idae.es/ayudas-y-financiacion/comunidades-energeticas

[6] Seguro Renovables. (2023). Qué son las comunidades energéticas y qué ventajas tienen. Recuperado de https://www.segurorenovables.com/energia-renovable/que-son-las-comunidades-energeticas-y-que-ventajas-tienen/

[7] APPA Renovables. (2023). Qué es la energía fotovoltaica. APPA - Asociación de Empresas de Energía Renovables. Recuperado de https://www.appa.es/appa-fotovoltaica/que-es-la-energia-fotovoltaica/

[8] Direnergy. (2023). Tipos de instalaciones solares. Recuperado de https://www.direnergy.net/index.php/blog/tipos-instalaciones-solares/

[9] García Martín, P. F. (2022). Energía solar fotovoltaica para todos (2ª ed.). Editorial Marcombo.

[10] SotySolar. (2023). Monitorización de instalaciones fotovoltaicas. Recuperado de https://sotysolar.es/placas-solares/monitorizacion

[11] Modbus Organization, Inc. (2006). MODBUS Application Protocol Specification V1.1b3. Recuperado de http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf

[12] DSC Solar. (2023). Conectados a Red. Recuperado de https://dscsolar.es/fotovoltaica/inversores-cc-ca/conectados-a-red/

[13] Díaz González, S. (2023). Web Scraping. Sistemas de Percepción (Máster de Ingeniería Industrial). Universidad de La Laguna.

[14] Torres Rodríguez, I. J. (2023). RPA. Informática Industrial Avanzada (Máster en Ingeniería Industrial). Universidad de La Laguna.

[15] MuleSoft. (2023). What is an API?. MuleSoft, Inc. Recuperado de https://www.mulesoft.com/resources/api/what-is-an-api

[16] Google Cloud. (2023). ¿Qué es IaaS?. Google Cloud. Recuperado de https://cloud.google.com/learn/what-is-iaas?hl=es

[17] Madakam, S., Lake, V., Lake, V., & Lake, V. et al. (2015). Internet of Things (IoT): A literature review. Journal of Computer and Communications, 3(05), 164.

[18] IoT Analytics. (2023). Global IoT market size to grow 19% in 2023. IoT Analytics. Recuperado de https://iot-analytics.com/iot-market-size/

[19] Lvivity. (2023). Peculiarities of Developing IoT Solutions for Business. Recuperado de https://lvivity.com/main-peculiarities-of-developing-iot-solutions-for-business

[20] Kinsta. (2023). ¿Qué es el IdC (Internet de las Cosas)?. Recuperado de https://kinsta.com/es/base-de-conocimiento/que-es-iot/

[21] Cognizant. (2023). ¿Qué es una plataforma del Internet de las Cosas (IoT)?. Cognizant España. Recuperado de https://www.cognizant.com/es/es/glossary/internet-of-things-platform

[22] ThingsBoard. (2023). ThingsBoard: Getting Started Guide. Recuperado de https://thingsboard.io/docs/getting-started-guides/what-is-thingsboard/

[23] ThingsBoard. (2023). ThingsBoard architecture. Recuperado de https://thingsboard.io/docs/

[24] Selenium. (2023). The Selenium Browser Automation Project. Selenium. Recuperado de https://www.selenium.dev/documentation/

[25] Geekflare. (2023). Introducción a las redes neuronales. Recuperado de https://geekflare.com/neural-networks/

[26] Mohamad Radzi, P.N.L.; Akhter, M.N.; Mekhilef, S.; Mohamed Shah, N. Review on the Application of Photovoltaic Forecasting Using Machine Learning for Very Short- to Long-Term Forecasting. Sustainability 2023, 15, 2942.

[27] Sotaquirá, M. (2019). ¿Qué son las Redes LSTM? Codificando Bits. Recuperado de https://www.codificandobits.com/blog/redes-lstm/

[28] Torres, J. (2019). Redes Neuronales Recurrentes. Jordi TORRES.AI. Recuperado de https://torres.ai/redes-neuronales-recurrentes/

[29] Sotaquirá, M. (2021). Las redes LSTM (parte 1): las compuertas y la celda de memoria candidata. Codificando Bits. Recuperado de https://www.codificandobits.com/curso/fundamentos-deep-learning-python/redes-recurrentes-9-lstm-compuertas-celda-memoria/

[30] Arnay, R., Hernández-Aceituno, J., Gomez-Gonzalez, J.-F., Mendez-Perez, J. A. (2024). Energy forecasting using intelligent models.

[31] Gobierno de Canarias. (2024). Visualizador agroclimático. Recuperado de https://www3.gobiernodecanarias.org/agricultura/agroclimatica/#/visualizador

[32] Scikit-learn developers. (2023). sklearn.preprocessing.StandardScaler. Scikit-learn 1.4.2 documentation. Recuperado de https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[33] Keras. (n.d.). Adam. Keras Documentation. Recuperado de https://keras.io/api/optimizers/adam/

[34] Keras. (n.d.). RMSprop. Keras Documentation. Recuperado de https://keras.io/api/optimizers/rmsprop/

[35] Keras. (n.d.). EarlyStopping. Recuperado de https://keras.io/api/callbacks/early_stopping/

[36] Stack Overflow. (2017). RMSE loss function in Keras. Recuperado de: https://stackoverflow.com/questions/43855162/rmse-rmsle-loss-function-in-keras

# Apéndice

# Apéndice A. Códigos implementados

## Apéndice A.1. Repositorio de códigos implementados para desarrollar el sistema de monitorización en ThingsBoard

Los códigos empleados para el desarrollo del Sistema de Monitorización en ThingsBoard están disponibles en el siguiente enlace: [Repositorio en GitHub](https://github.com/AngelMarcos5/Sistema-Monitorizacion-ThingsBoard).

***Descripción de los Archivos:***

- *Web2ThingsBoard.ipynb:* Recopila los datos de todos los edificios que se encuentran en la página de Goodwe para el día actual y los envía a cada uno de los dispositivos correspondientes en ThingsBoard.
- *SEGAI_IAAS_AnalisisHistorico.ipynb:* Recopila los datos históricos del Edificio SEGAI desde la página de Goodwe y los envía al dispositivo correspondiente en ThingsBoard.
- *SEGAI_IAAS.ipynb:* Extrae los datos del Edificio SEGAI desde la página de Goodwe para el día actual y los envía al dispositivo correspondiente en ThingsBoard.

## Apéndice A.2. Repositorio de códigos implementados para desarrollar el modelo de predicción basado en LSTM

Los códigos empleados para el desarrollo del modelo de predicción de la generación fotovoltaica del Edificio SEGAI de la ULL están disponibles en el siguiente enlace: [Repositorio en GitHub]( https://github.com/AngelMarcos5/Modelo-Prediccion-LSTM).

***Descripción de los Archivos:***

- *Datos_SEMS_PORTAL.ipynb:* Extrae los datos de generación y consumo del Edificio SEGAI desde la página de Goodwe para cada día y los guarda en una ruta predefinida.
- *Datos_meteorologicos.ipynb:* Automatiza la extracción de los datos meteorológicos de la estación de La Laguna desde la página web del Ministerio de Agricultura, Pesca y Alimentación.
- *LSTM_MODEL_PART1.ipynb:* Prepara y analiza los datos que posteriormente se utilizarán para crear y entrenar el modelo.
- *LSTM_MODEL_PART2.ipynb:* Crea y entrena la Red LSTM, se realizan ajustes de hiperparámetros y pruebas, y se realizan predicciones para diferentes días con el mejor modelo y configuración.