ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Desarrollo de un Sistema Smart para la adquisición y monitoreo de parámetros de funcionamiento de línea extrusora de tubos PVC en industria de tuberías plásticas.

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingeniero en Electrónica y Automatización

Presentado por:

Gustavo José De Las Heras Castro

Eduardo Humberto Tumbaco Rocafuerte

GUAYAQUIL - ECUADOR

Año: 2024

DEDICATORIA

A mis padres, Marco De Las Heras y Lorena Castro, por su amor incondicional, por ser mi guía y mi fortaleza en cada paso que doy.

A mi abuela Blanca, mi hermana Lorena y mis tíos Jorge y Gustavo, por su apoyo, sus sacrificios y por enseñarme el valor del esfuerzo.

A Annie y su familia, por su compañía en este camino, por sus palabras de ánimo y por ser un pilar en los momentos más difíciles.

- Gustavo José De Las Heras Castro

A mis padres, Humberto Tumbaco e Inés Rocafuerte por todo su amor, paciencia, sacrificios durante todo este proceso y su fe inquebrantable en mí.

A mi hermana Nurys, amigos por ser mi soporte y mis cómplices en toda esta aventura.

A mis tías Elsa y Aleja, por su apoyo incondicional en todo momento.

- Eduardo Humberto Tumbaco Rocafuerte

AGRADECIMIENTOS

Agradecemos, al PhD. Efrén Herrera y al PhD. Ricardo Cajo, quienes con paciencia y dedicación nos brindaron su guía y conocimientos para desarrollar este proyecto.

A nuestras familias y amigos, por su amor, apoyo incondicional y por ser nuestra mayor fuente de motivación.

Al Ing. Jhony López, por brindar los recursos y el espacio necesarios para desarrollar este proyecto.

DECLARACIÓN EXPRESA

Nosotros, Gustavo José De Las Heras Castro y Eduardo Humberto Tumbaco Rocafuerte acordamos y reconocemos que:

La titularidad de los derechos patrimoniales de autor (derechos de autor) del proyecto de graduación corresponderá al autor o autores, sin perjuicio de lo cual la ESPOL recibe en este acto una licencia gratuita de plazo indefinido para el uso no comercial y comercial de la obra con facultad de sublicenciar, incluyendo la autorización para su divulgación, así como para la creación y uso de obras derivadas. En el caso de usos comerciales se respetará el porcentaje de participación en beneficios que corresponda a favor del autor o autores.

La titularidad total y exclusiva sobre los derechos patrimoniales de patente de invención, modelo de utilidad, diseño industrial, secreto industrial, software o información no divulgada que corresponda o pueda corresponder respecto de cualquier investigación, desarrollo tecnológico o invención realizada nosotros durante el desarrollo del proyecto de graduación, pertenecerán de forma total, exclusiva e indivisible a la ESPOL, sin perjuicio del porcentaje que nos corresponda de los beneficios económicos que la ESPOL reciba por la explotación de nuestra innovación, de ser el caso.

En los casos donde la Oficina de Transferencia de Resultados de Investigación (OTRI) de la ESPOL comunique los autores que existe una innovación potencialmente patentable sobre los resultados del proyecto de graduación, no se realizará publicación o divulgación alguna, sin la autorización expresa y previa de la ESPOL.

Guayaquil, 18 de febrero del 2025.

Gustavo José De Las Heras Castro	Eduardo Humberto Tumbaco Rocafuerte

EVALUADORES

EFREN VINICIO HERRERA MUENTES

Digitally signed by EFREN VINICIO HERRERA MUENTES
Date: 2025.02.19 00:08:09 -05'00'

PhD. Efrén Herrera

PROFESOR DE LA MATERIA

Firmado electrónicamente por:
RICARDO ALFREDO CAJO DIAZ

PhD. Ricardo Cajo

PROFESOR TUTOR

RESUMEN

Este proyecto presenta el diseño e implementación de un sistema inteligente para la adquisición y monitoreo de parámetros operativos en una línea extrusora de tuberías plásticas, con el objetivo de mejorar la producción y garantizar la calidad del producto final. El sistema busca solucionar la falta de herramientas fiables para el monitoreo en tiempo real de variables críticas, utilizando tecnologías accesibles y económicamente viables. La hipótesis planteada sostiene que la integración de protocolos de comunicación modernos y herramientas de análisis predictivo mejorará significativamente el rendimiento de la línea de producción. La justificación del proyecto radica en la necesidad de una solución adaptada a las condiciones y limitaciones del entorno industrial.

El desarrollo incluyó la instalación y configuración de un PLC LOGO! y una red de dispositivos conectados mediante Ethernet y MQTT. Se emplearon herramientas como Node-RED, Python y Grafana para la captura, procesamiento y visualización de datos. Asimismo, se implementó una base de datos MariaDB para el almacenamiento de la información y su posterior análisis. Los datos recopilados permitieron implementar un algoritmo de predicción para anticipar mantenimientos futuros y la vida útil del motor, utilizando información sobre temperatura, vibración y tiempos de producción.

Los resultados muestran un sistema capaz de monitorear y analizar en tiempo real las variables operativas, logrando identificar patrones relevantes para el mantenimiento predictivo. Esto permitió reducir tiempos de inactividad no planificados y mejorar la eficiencia energética de la línea extrusora. En conclusión, el sistema desarrollado proporciona una solución eficiente y escalable, con opción para ser replicada en otras líneas industriales similares.

Palabras clave: Monitoreo, Mantenimiento Predictivo, Extrusión, Automatización, Industria Plástica.

ABSTRACT

This project presents the design and implementation of an intelligent system for acquiring and monitoring operational parameters in a plastic pipe extrusion line, aiming to improve production and ensure the quality of the final product. The system addresses the lack of reliable tools for real-time monitoring of critical variables by utilizing accessible and cost-effective technologies. The proposed hypothesis suggests that the integration of modern communication protocols and predictive analysis tools will significantly enhance the production line's performance. The project's justification lies in the need for a solution tailored to the conditions and limitations of the industrial environment.

The development included the installation and configuration of a LOGO! PLC and a network of devices connected via Ethernet and MQTT. Tools such as Node-RED, Python, and Grafana were employed for data acquisition, processing, and visualization. Additionally, a MariaDB database was implemented to store information for subsequent analysis. The collected data enabled the implementation of a prediction algorithm to anticipate future maintenance needs, using information on temperature, vibration, and production times.

The results demonstrate a system capable of monitoring and analyzing operational variables in real time, identifying relevant patterns for predictive maintenance. This reduced unplanned downtime and improved the energy efficiency of the extrusion line. In conclusion, the developed system provides an efficient and scalable solution, with the potential for replication in similar industrial lines.

Keywords: Monitoring, Predictive Maintenance, Extrusion, Automation, Plastic Industry.

ÍNDICE GENERAL

ABREVIATURAS

ESPOL	Escuela Superior Politécnica del Litoral
PLC	Controlador Lógico Programable
MQTT	Transporte de telemetría de colas de mensajes
PVC	Policloruro de vinilo
WiFi	Fidelidad inalámbrica
IEEE	Instituto de Ingenieros Eléctricos y Electrónicos
SQL	Lenguaje de consulta estructurado
XT, EXT	Extrusora

SIMBOLOGÍA

mm/s	Milímetro por segundo
kg	Kilogramo
m	Metro
V	Voltio
A	Amperio
ºC	Grado Celsius
bar	Bar
RPM	Revoluciones por minuto

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE PLANOS

CAPÍTULO 1

1. INTRODUCCIÓN

En la empresa cliente, productora de tuberías plásticas, el proceso de extrusión es una de las etapas más críticas para garantizar la calidad del producto final. Este proceso involucra el manejo de variables fundamentales, tales como la temperatura, la velocidad de extrusión y la vibración de los equipos, que son determinantes para asegurar que la tubería moldeada cumpla con los estándares requeridos. Sin embargo, actualmente muchas líneas de extrusión carecen de sistemas de monitoreo y control fiables que permitan gestionar estas variables de manera precisa y continua. Esta situación provoca que, en muchos casos, los operadores dependan de observaciones manuales, lo cual puede llevar a errores operativos y a una gestión ineficiente de la energía y los materiales.

En este contexto, la variabilidad de los compuestos de materia prima empleados y la ausencia de un monitoreo constante representan un reto significativo para la empresa. La falta de visibilidad y control sobre parámetros críticos puede resultar en productos de baja calidad o tuberías mal moldeadas, con consecuencias directas tanto en la satisfacción del cliente como en los costos de producción. Por lo tanto, surge la necesidad de desarrollar una solución que permita la adquisición y monitoreo fiable de estos parámetros, mejorando el proceso de fabricación y la gestión de la información para la toma de decisiones.

El presente proyecto se centra en la creación de un prototipo de un sistema SMART para la línea de extrusión de tuberías plásticas, aprovechando herramientas de desarrollo modernas y protocolos de comunicación de media fiabilidad como Profinet, MQTT, comunicación S7, y WiFi. Esta solución se basa en el uso de un lenguaje de programación flexible como Python y plataformas de diseño como Node-Red y LOGO! SoftComfort, junto con un sistema de visualización en tiempo real mediante Grafana.

Además, se plantea la integración de un modelo de red neuronal de regresión para el análisis predictivo del tiempo de vida útil del motor principal, basado en datos de temperatura y vibración. Esta característica permitirá a la empresa implementar un plan de mantenimiento preventivo y reducir el riesgo de fallos inesperados.

1.1 Descripción del problema

En la actualidad, las líneas de extrusión de tuberías plásticas enfrentan desafíos significativos que afectan tanto la calidad del producto como la eficiencia del proceso. La extrusión es un proceso continuo en el que la materia prima es transformada en un producto final a través de un control de varios parámetros operativos. Sin embargo, muchas de estas líneas carecen de sistemas adecuados para monitorear y gestionar estos parámetros de forma efectiva, lo que conduce a diversas complicaciones.

Uno de los problemas más destacados es la falta de un monitoreo fiable de las variables críticas del proceso, tales como la temperatura del material, la presión del molde, la velocidad de extrusión y la vibración de componentes como el motor principal. Sin estos datos en tiempo real, los operadores deben basarse en su experiencia y en mediciones manuales esporádicas, lo que puede llevar a decisiones inadecuadas y, en consecuencia, a la producción de tuberías de baja calidad. Estas tuberías mal moldeadas no solo generan desperdicio de material, sino que también implican costos adicionales debido a retrabajos y la insatisfacción del cliente.

Además, la variabilidad de los compuestos de materia prima, que pueden incluir aditivos y diferentes tipos de plásticos, añade una capa adicional de complejidad. Cada tipo de material puede requerir condiciones específicas de operación para asegurar que el producto final cumpla con las especificaciones. Sin un sistema de monitoreo que permita observar estos parámetros en tiempo real, la producción puede ser inconsistente, aumentando el riesgo de errores en el proceso de extrusión.

Otro aspecto crítico es el consumo de energía. Las líneas extrusoras suelen operar bajo condiciones subóptimas, lo que no solo afecta la calidad del producto, sino que también incrementa el uso de energía. La falta de información precisa sobre el estado del proceso impide la identificación de ineficiencias y la implementación de estrategias de eficiencia energética.

Finalmente, la ausencia de un sistema integrado de recolección y análisis de datos dificulta la toma de decisiones. Sin acceso a información precisa y actualizada, la gestión operativa se dificulta. Esto impide que la empresa se adapte a cambios en la producción, en las características de los materiales o en la demanda del mercado, lo que puede poner en riesgo su competitividad.

1.2 Justificación del problema

La solución propuesta, que implica el desarrollo de un sistema SMART para la adquisición y monitoreo de parámetros en una línea de extrusión de tuberías plásticas, surge por su capacidad para abordar las necesidades críticas de la industria y las limitaciones actuales en los procesos de fabricación. Este sistema permitirá un control riguroso de las variables operativas clave, como la temperatura, la presión y la velocidad de extrusión, asegurando que las tuberías producidas cumplan con los estándares de calidad requeridos y minimizando el riesgo de productos defectuosos.

Además, la integración de protocolos de comunicación como Profinet, MQTT y S7 facilitará un intercambio de información fluido entre los dispositivos y sistemas involucrados en la línea extrusora, optimizando así las condiciones de operación en función de los datos recogidos. Esto no solo incrementará la eficiencia del proceso de producción, sino que también permitirá la identificación de oportunidades para reducir el consumo energético, generando beneficios económicos directos y contribuyendo a la sostenibilidad ambiental.

La visualización de datos a través de herramientas como Grafana proporcionará a los gerentes y operadores una interfaz intuitiva para interpretar la información de manera efectiva, facilitando la toma de decisiones informadas y proactivas. Asimismo, el desarrollo de un modelo de red neuronal para predecir el tiempo de vida útil del motor principal permitirá implementar estrategias de mantenimiento predictivo, reduciendo el riesgo de fallos inesperados y prolongando la vida útil del equipo.

Este enfoque garantiza que la empresa pueda adaptarse a los requerimientos de producción. La elección de tecnologías y herramientas accesibles y sostenibles asegura que la solución pueda ser implementada sin requerir inversiones desproporcionadas, brindando un retorno sobre la inversión favorable a mediano y largo plazo. En conjunto, esta solución no solo responde a las necesidades inmediatas de la empresa, sino que también se alinea con las tendencias hacia la industria 4.0 y la fabricación inteligente.

1.3 Objetivos

1.3.1 Objetivo General

Desarrollar un prototipo de un sistema SMART que permita la adquisición y monitoreo de parámetros de funcionamiento en una línea de extrusión de tuberías plásticas, utilizando herramientas de desarrollo y protocolos de comunicación de media fiabilidad, con el fin de optimizar el proceso de fabricación y mejorar la gestión de la información para toma de decisiones.

1.3.2 Objetivos Específicos

1. Implementar un sistema de comunicación basado en protocolos como Profinet, MQTT, comunicación S7, y WiFi que permita el correcto funcionamiento de los dispositivos para el envío y recepción de la información sobre la línea de extrusión.
2. Diseñar un sistema programable mediante herramientas de desarrollo como LOGO! SoftComfort, Node-red, y Python, que adquiera y procese la información de manera precisa, facilitando la correcta interpretación de los datos.
3. Integrar una interfaz gráfica utilizando Grafana, que reciba información de una base de datos en MariaDB y permita la visualización en tiempo real de los parámetros de la línea de extrusión, mejorando así la toma de decisiones operativas.
4. Desarrollar un modelo de prueba de una red neuronal de regresión utilizando datos de temperatura y vibración, que prediga el tiempo de vida útil del motor principal de la línea de extrusión, permitiendo una mejor planificación del mantenimiento.

1.4 Marco teórico

1.4.1 Línea de Extrusión

Una línea de extrusión es un proceso industrial utilizado para moldear materiales plásticos (y también metales o alimentos) en formas continuas y específicas, normalmente de sección transversal uniforme, mediante la aplicación de calor, presión y un molde (o matriz). En la extrusión, el material plástico es alimentado a través de una máquina que lo funde y lo empuja hacia una boquilla o matriz que le da la forma deseada. Luego, el material extruido se enfría y solidifica en la forma final. [1]

Ilustración 1: Línea de Extrusión [2]

1.4.2 Tuberías plásticas

Las tuberías plásticas son uno de los productos más comunes fabricados mediante extrusión. Estas tuberías se utilizan en una amplia variedad de aplicaciones, desde sistemas de agua potable y drenaje hasta aplicaciones industriales y de telecomunicaciones. Las tuberías plásticas tienen varias ventajas sobre las de metal, como la resistencia a la corrosión, el peso más liviano y la facilidad de instalación. Las materias primas más comunes para la fabricación de estas tuberías son el polipropileno y el policloruro de vinilo.

Ilustración 2: Tuberías plásticas [3]

1.4.3 Materia prima

1.4.3.1 Polipropileno

El polipropileno es un tipo de plástico derivado del propileno, un gas que se encuentra en el petróleo crudo y en los gases de refinación. Es un tipo de polímero termoplástico que se utiliza comúnmente en la fabricación de una amplia gama de productos debido a su bajo costo, alta resistencia y versatilidad. Se utiliza en aplicaciones que van desde envases y botellas hasta piezas de automóviles y dispositivos médicos.

El polipropileno es altamente resistente a la humedad, los productos químicos y los cambios de temperatura, lo que lo hace ideal para su uso en ambientes difíciles. Además, es un material reciclable y se puede reutilizar para crear nuevos productos.

1.4.3.2 Policloruro de vinilo

El PVC o policloruro de vinilo, es un plástico que surge a partir de la polimerización del monómero de cloroetileno (también conocido como cloruro de vinilo). Los componentes del PVC derivan del cloruro de sodio y del gas natural o del petróleo, e incluyen cloro, hidrógeno y carbono.

En su estado original, el PVC es un polvo amorfo y blanquecino. La resina resultante de la mencionada polimerización es un plástico que puede emplearse de múltiples maneras, ya que permite producir objetos flexibles o rígidos.

Una de las propiedades más interesantes del PVC es que resulta termoplástico: al ser sometido al calor, se vuelve blando y se puede moldear con facilidad. Al enfriarse, recupera la solidez anterior sin perder la nueva fisonomía.

1.4.4 Parámetros del proceso

1.4.4.1 Temperatura del material

La temperatura del material es la temperatura de la materia prima que se encuentra en la tolva inicial del proceso de extrusión.

1.4.4.2 Presión de la masa

La presión de la masa es la que se encuentra dentro del túnel del tornillo sin fin por donde la materia prima pasa para llegar al molde.

1.4.4.3 Temperatura del motor principal

Esta temperatura es la que se encuentra en la carcasa del motor principal.

1.4.4.4 Vibraciones del motor principal

Las vibraciones del motor se miden en velocidades de mm/s en los ejes x y z en la carcasa de este.

1.4.4.5 RPM del motor principal

Las revoluciones por minuto al que gira el rotor. Esta señal viene del variador de frecuencia principal de la línea extrusora.

1.4.4.6 Amperaje del motor principal

El amperaje que consume el motor durante su puesta en marcha. Esta señal viene del variador de frecuencia principal de la línea extrusora.

1.4.4.7 Tiempo de Producción Actual y Total

El tiempo de producción actual es el tiempo (en horas) en el cual la línea de extrusión produce material en una sola sesión. El tiempo total es el acumulado durante un año. Estos parámetros son calculados por el PLC LOGO.

1.4.4.8 Tiempo de Calentamiento

El tiempo de calentamiento es el tiempo (en horas) en el cual los cabezales de calentamiento mantienen una alta temperatura en el molde. Este tiempo es calculado por el PLC LOGO.

1.4.4.9 Virutas A y B

Son los excedentes que se producen en las cortadoras A y B de la línea extrusora, medidas en kg.

1.4.5 Controladores

1.4.5.1 PLC LOGO!

El PLC LOGO! es un controlador lógico programable (PLC, por sus siglas en inglés) desarrollado por Siemens, que está diseñado para aplicaciones de automatización pequeñas o de nivel básico. Es especialmente popular por su facilidad de uso, flexibilidad y pequeño tamaño, lo que lo hace ideal para tareas sencillas de control y automatización, como en sistemas de control de iluminación, sistemas de calefacción, automatización de puertas, control de bombas, entre otros.

Ilustración 3: PLC LOGO! [4]

1.4.5.2 Raspberry Pi

Raspberry Pi es una pequeña computadora de bajo costo, del tamaño de una tarjeta de crédito, diseñada para fomentar el aprendizaje de la programación, la informática y la electrónica. Aunque originalmente se creó como una herramienta educativa para estudiantes y aficionados, su versatilidad y accesibilidad lo han convertido en una plataforma muy popular para proyectos de automatización, robótica, internet de las cosas (IoT), multimedia, estudio de sistemas operativos, y prototipado rápido.

Ilustración 4: Raspberry Pi 4 [5]

1.4.6 Sistemas de comunicación

1.4.6.1 Red WIFI

Una red de comunicación Wi-Fi es un sistema que permite conectar dispositivos electrónicos a través de señales de radiofrecuencia, facilitando la transmisión de datos sin necesidad de cables. Basada en el estándar IEEE 802.11, esta red opera en frecuencias de 2.4 GHz, 5 GHz, y más recientemente en 6 GHz, lo que permite una comunicación rápida y eficiente. En aplicaciones de recolección de datos, esta red permite que los dispositivos terminales, como sensores y otros equipos móviles, envíen información en tiempo real a un servidor central para su almacenamiento y análisis. Los dispositivos se conectan mediante adaptadores inalámbricos que transmiten los datos a un enrutador, el cual actúa como un puente de comunicación entre los dispositivos y la red de datos más amplia, ya sea una red local o Internet. [6]

Ilustración 5: Red de comunicación WIFI [7]

1.4.6.2 Protocolo de comunicación MQTT

MQTT fue desarrollado originalmente por IBM en 1999, pensado para permitir la comunicación en dispositivos que funcionan en redes con limitaciones de ancho de banda y energía. Este protocolo sigue un modelo de publicación y suscripción, lo que significa que los dispositivos (o clientes) pueden publicar datos en un 'tema' o suscribirse a temas para recibir información. Todo esto se gestiona a través de un servidor central llamado broker, que actúa como intermediario, enrutando los mensajes entre clientes.

Una característica clave de MQTT es su simplicidad y su bajo consumo de ancho de banda, ya que utiliza un protocolo binario que reduce la carga de los datos transmitidos. A diferencia de otros protocolos, MQTT también permite establecer un nivel de calidad de servicio (QoS), que asegura distintos grados de fiabilidad en la entrega de los mensajes. Esto es particularmente útil en aplicaciones donde la transmisión de datos no puede fallar, como en sistemas de automatización, monitoreo remoto, y en el Internet de las Cosas (IoT). Los dispositivos se conectan de forma constante al broker, lo que permite que los datos se distribuyan en tiempo real, favoreciendo el funcionamiento de sistemas que requieren respuestas rápidas y precisas. [8]

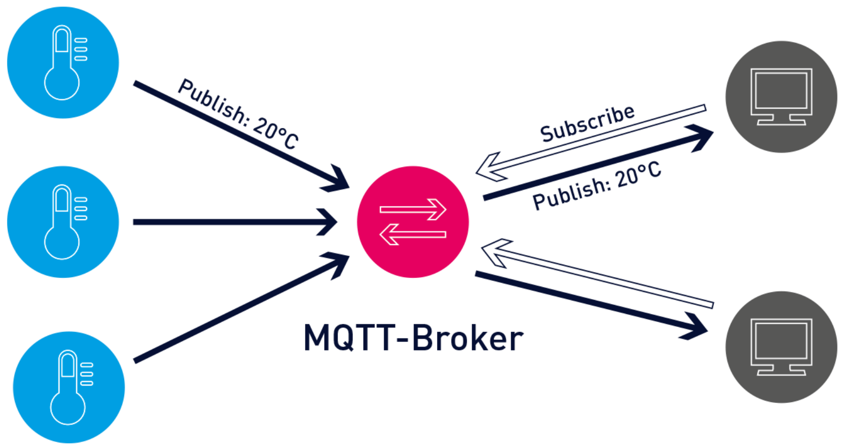


Ilustración 6: Protocolo de Comunicación MQTT [9]

1.4.6.3 Ethernet

Ethernet es una tecnología de red de área local (LAN) ampliamente utilizada que permite la interconexión de dispositivos en una misma red mediante un estándar de comunicación por cables o inalámbrico. Fue desarrollada inicialmente por Xerox en la década de 1970 y, desde entonces, ha evolucionado significativamente en cuanto a velocidad y rendimiento. Ethernet utiliza una arquitectura de comunicación basada en paquetes, donde los datos se dividen en pequeños fragmentos o frames que son transmitidos a través de cables de par trenzado, coaxial o fibra óptica, dependiendo de la velocidad y el alcance requeridos. Esta tecnología opera en la capa de enlace de datos del modelo OSI, usando identificadores únicos llamados direcciones MAC para dirigir y controlar el flujo de información entre dispositivos, [10]

La estandarización de Ethernet, gestionada por el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) bajo el estándar 802.3, permite una compatibilidad universal y ha facilitado su adopción global en entornos de trabajo y en redes domésticas. Con la introducción de tecnologías como Gigabit Ethernet y, más recientemente, 10 Gigabit Ethernet, esta tecnología ofrece velocidades de transmisión que soportan desde simples redes de oficina hasta aplicaciones de gran demanda de datos, como centros de datos y servicios en la nube.

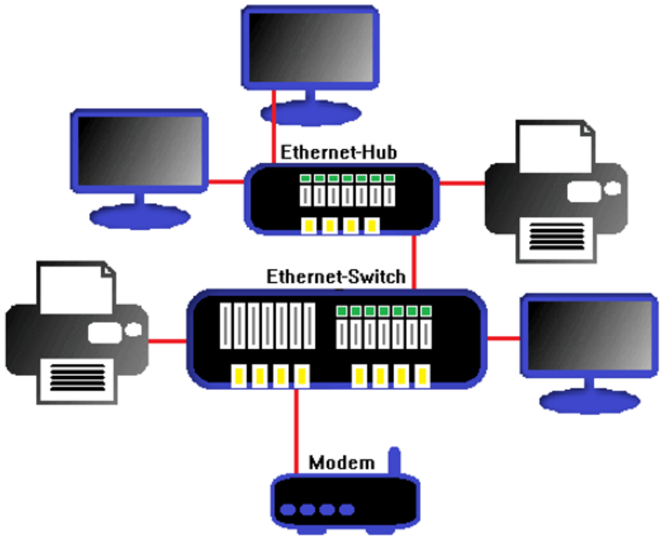


Ilustración 7: Sistema de conexión vía Ethernet [11]

1.4.7 Herramientas de Desarrollo

1.4.7.1 Node-RED

Es una herramienta de libre acceso que facilita el desarrollo de aplicaciones IoT y flujos de trabajo de automatización. Diseñado con una interfaz visual intuitiva, Node-RED brinda a los usuarios conectar equipos y plataformas de nube de manera eficiente. Sus nodos predefinidos, que representan funciones específicas, se pueden arrastrar y soltar para construir flujos de trabajo de forma fácil para la automatización y la interconexión de dispositivos. [12]

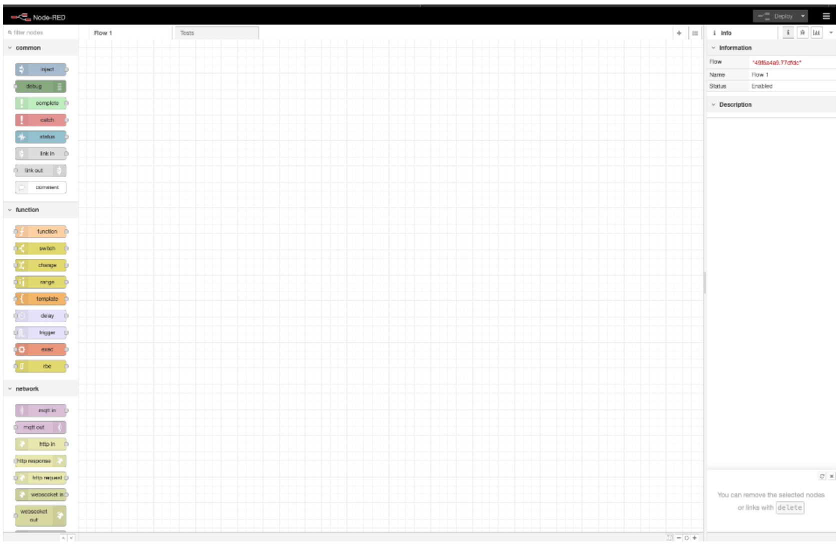


Ilustración 8: Entorno de trabajo en Node-RED

1.4.7.1.1 Nodo S7

En el ámbito de la automatización industrial, la integración de Node-RED con los controladores lógicos programables Siemens S7 representa un avance significativo para el monitoreo y control de procesos en tiempo real. Node-RED es una plataforma de desarrollo de flujo visual que permite construir aplicaciones de IoT mediante nodos específicos, como los de Siemens S7, los cuales facilitan la comunicación directa con estos controladores. Estos nodos permiten leer y escribir datos en el PLC, haciendo posible la visualización de variables de proceso como temperaturas, estados de dispositivos, y parámetros de maquinaria en aplicaciones de supervisión remota. Esta integración es especialmente valiosa en el contexto de la Internet Industrial de las Cosas (IIoT), donde la comunicación de datos de planta con aplicaciones en la nube se convierte en una herramienta clave para implementar estrategias de mantenimiento predictivo y análisis de datos en tiempo real. Según Hernández y Smith (2019), Node-RED simplifica el proceso de adquisición y análisis de datos en entornos industriales al ofrecer compatibilidad con protocolos como MQTT y HTTP, permitiendo una supervisión continua y el ajuste de parámetros sin necesidad de una programación avanzada. De acuerdo con Siemens AG (2020), los controladores S7-1200 y S7-1500, principales modelos de PLC de Siemens cuentan con características avanzadas de conectividad que, junto con Node-RED, posibilitan la creación de sistemas interconectados para la industria 4.0. [13]

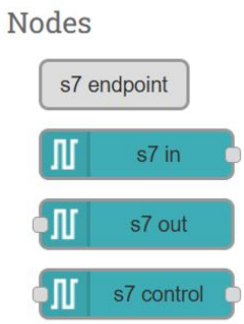


Ilustración 9: Nodos de Comunicación S7

1.4.7.1.2 Nodo SQL

La integración de bases de datos en tiempo rea9l en entornos de automatización se facilita con Node-RED y su compatibilidad con nodos específicos para MariaDB, una de las bases de datos relacionales de código abierto más utilizadas en la industria. Estos nodos de MariaDB permiten a los usuarios ejecutar consultas SQL directamente desde Node-RED, lo que posibilita almacenar, recuperar y gestionar grandes volúmenes de datos generados por sistemas de monitoreo y control en procesos industriales. Esta conexión es esencial en aplicaciones de IoT y en la creación de sistemas de toma de decisiones basados en datos históricos y en tiempo real. Al utilizar estos nodos, los datos adquiridos por sensores y dispositivos pueden registrarse en una base de datos MariaDB, habilitando la generación de informes históricos y la implementación de modelos predictivos para optimizar procesos y reducir tiempos de inactividad en plantas industriales. Además, Node-RED simplifica la manipulación de datos mediante su interfaz visual, permitiendo a los ingenieros sin experiencia en programación avanzada beneficiarse de la gestión y análisis de datos en sus sistemas de automatización. [14]

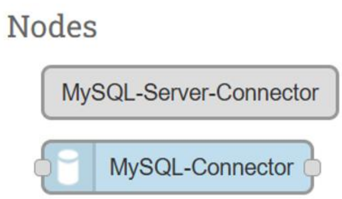


Ilustración 10: Nodo de Servidor SQL

1.4.7.1.3 Nodo MQTT

En el contexto de la Internet de las Cosas (IoT) y la automatización industrial, los nodos de comunicación MQTT en Node-RED se han consolidado como herramientas clave para la transmisión de datos de manera eficiente y en tiempo real. MQTT (Message Queuing Telemetry Transport) es un protocolo de mensajería ligero y de baja latencia, diseñado para redes con limitaciones de ancho de banda y aplicaciones que requieren confiabilidad en la entrega de datos (Banks & Gupta, 2021). Los nodos MQTT en Node-RED permiten publicar y suscribirse a mensajes en una red de dispositivos IoT, haciendo posible que sensores y actuadores intercambien información sin la necesidad de conexiones complejas. Esta capacidad es fundamental en aplicaciones de monitoreo y control donde los dispositivos deben enviar datos a un servidor central o a una plataforma en la nube para su análisis y visualización en tiempo real. Además, la interfaz visual de Node-RED facilita la configuración de estos nodos, lo cual permite a usuarios sin experiencia avanzada en programación crear flujos de datos robustos y escalables. [15]

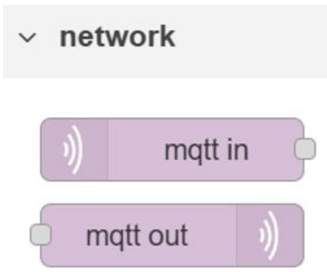


Ilustración 11: Nodo de Comunicación MQTT

1.4.7.2 Grafana

Grafana es una plataforma de acceso libre enfocada en la visualización de datos, clave para el análisis de información. Su función principal es conectarse con diversas fuentes de datos, desde bases de datos hasta servicios en la nube, y mostrar la información almacenada. La versatilidad de Grafana reside en su capacidad para personalizar y organizar las visualizaciones según las necesidades del usuario. Además, permite configurar alertas que notifican al usuario cuando se cumplen ciertos umbrales o condiciones específicas, facilitando respuestas rápidas ante cualquier cambio en los datos. [16]

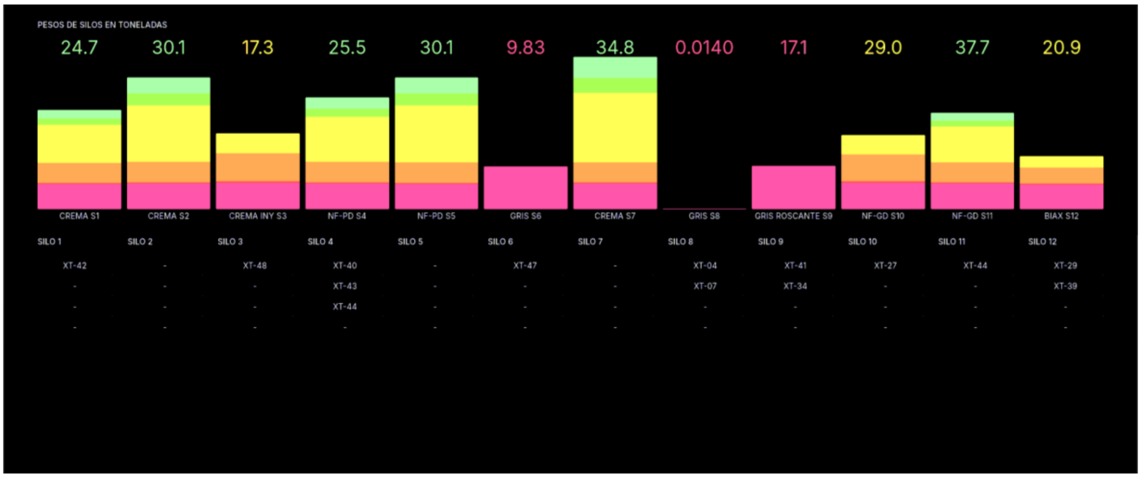


Ilustración 12: Entorno visual en Grafana

1.4.7.3 MariaDB

MariaDB es un sistema de gestión de bases de datos relacional de código abierto, que se deriva de MySQL y es compatible en gran medida con este último. Fue desarrollado para ofrecer una alternativa gratuita y de alto rendimiento a MySQL, con mejoras en la seguridad, escalabilidad y flexibilidad. MariaDB soporta una amplia variedad de tipos de datos y estructuras de índices, y es ampliamente utilizado en aplicaciones web, sistemas de gestión de contenido y análisis de datos. [17]

1.4.7.4 DBeaver

DBeaver es una herramienta de administración y desarrollo de bases de datos de código abierto. Es compatible con una gran variedad de sistemas de gestión de bases de datos (como MySQL, PostgreSQL, Oracle, SQLite, SQL Server, MariaDB, entre otros), y permite a los usuarios ejecutar consultas SQL, administrar estructuras de bases de datos, y visualizar datos de manera gráfica. DBeaver es popular tanto entre desarrolladores como administradores de bases de datos debido a su interfaz intuitiva y su capacidad para integrarse con diversas plataformas y herramientas.

Esta herramienta también facilita la importación y exportación de datos, la creación de diagramas de base de datos y el manejo de conexiones de forma segura, lo que la convierte en una opción muy versátil para la administración de datos y el análisis. [18]

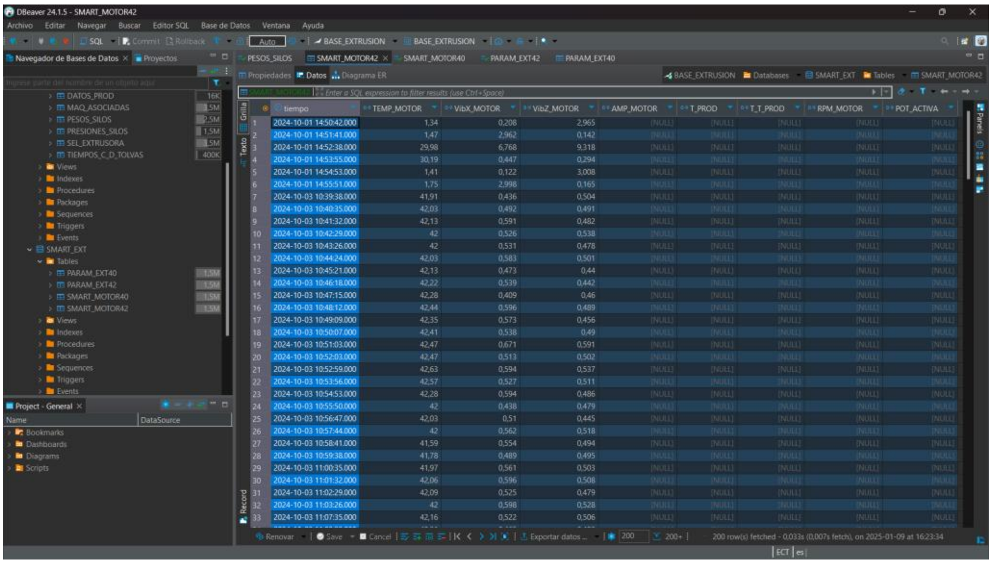


Ilustración 13: Entorno de trabajo de DBeaver

1.4.8 Sensores

1.4.8.1 Sensor para predicción QMT30VT2

El sensor QM30VT2 es un sensor de vibración y temperatura de la marca Banner Engineering diseñado para aplicaciones industriales. Este sensor permite monitorear la condición de máquinas al detectar cambios en la vibración y temperatura, lo que ayuda a identificar problemas mecánicos como desbalance, desalineación y desgastes. El sensor QM30VT2 es compacto y robusto, ideal para entornos industriales difíciles, y puede integrarse con sistemas de monitoreo y mantenimiento predictivo.

El sensor ofrece comunicación a través de protocolos como Modbus y se puede conectar a sistemas de control o redes industriales para recopilar datos en tiempo real, lo cual es útil para el mantenimiento preventivo de equipos y para reducir el tiempo de inactividad en la producción. [19]

Ilustración 14: Sensor de vibración y temperatura QM30VT2 [20]

1.4.9 Redes Neuronales

Las redes neuronales son modelos computacionales inspirados en la estructura y funcionamiento del cerebro humano, diseñadas para procesar información de manera similar a como lo harían las neuronas biológicas. En estos modelos, una red neuronal está compuesta por una serie de "neuronas" artificiales (también llamadas nodos o unidades) distribuidas en capas: una capa de entrada que recibe los datos, una o varias capas ocultas que procesan la información, y una capa de salida que entrega el resultado. Cada neurona de una capa está conectada con las neuronas de la siguiente capa mediante pesos sinápticos, que determinan la influencia que cada neurona tiene en las demás. La capacidad de aprendizaje de una red neuronal se basa en la modificación de estos pesos sinápticos a través de un proceso de entrenamiento, generalmente mediante algoritmos de optimización como el descenso de gradiente. A medida que una red neuronal aprende, ajusta sus pesos sinápticos para minimizar el error entre sus salidas y los valores esperados, lo que le permite identificar patrones y realizar predicciones o clasificaciones. Este aprendizaje puede ser supervisado, no supervisado o reforzado, dependiendo del tipo de tarea y de los datos disponibles. Las redes neuronales se aplican en campos como el reconocimiento de voz e imágenes, la predicción de series temporales y en sistemas de recomendación, entre otros, debido a su capacidad para manejar grandes volúmenes de datos y resolver problemas complejos. [21]

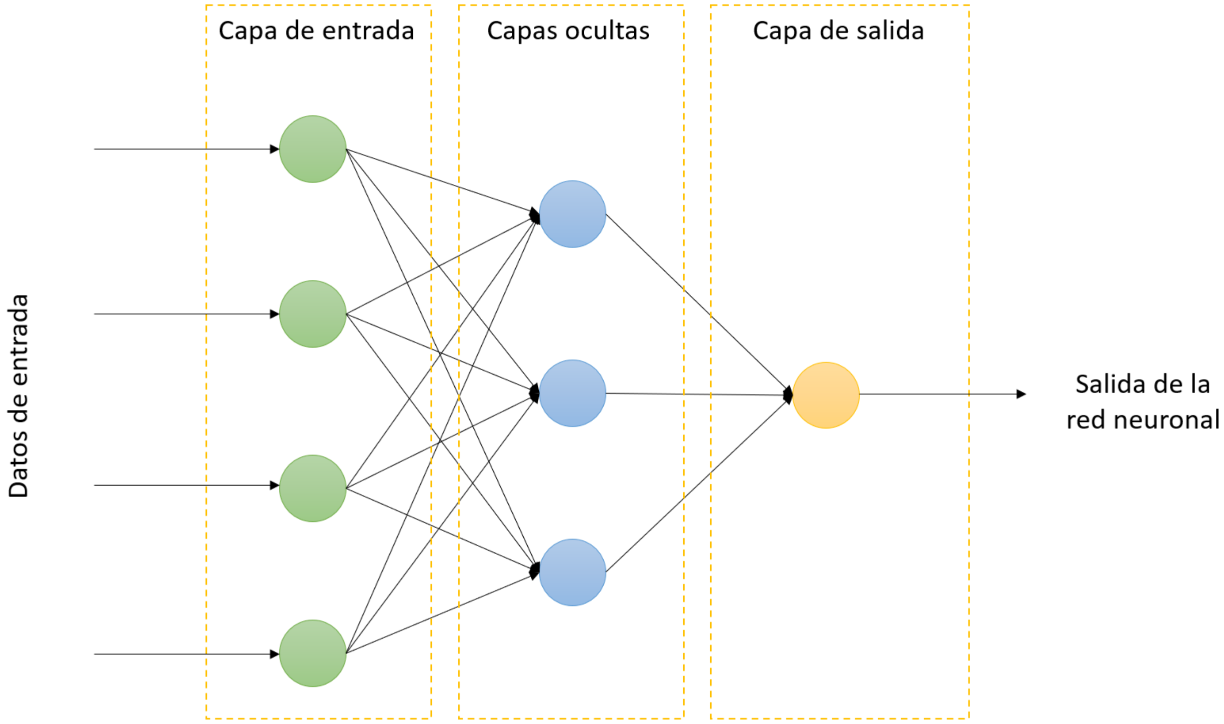


Ilustración 15: Estructura de red neuronal [22]

1.4.9.1 Red Neuronal de regresión

Las redes neuronales de regresión son un tipo específico de red neuronal diseñada para abordar problemas donde la salida es continua en lugar de categórica, lo que las hace ideales para tareas de predicción numérica. A diferencia de las redes neuronales para clasificación, donde la red asigna una entrada a una categoría específica, las redes de regresión buscan predecir un valor real, como la temperatura de un sistema, el precio de una vivienda o la vida útil restante de un componente. En términos de estructura, una red neuronal de regresión sigue el mismo principio básico de las redes neuronales convencionales, con capas de entrada, capas ocultas y una capa de salida. Sin embargo, en lugar de funciones de activación en la capa de salida que limitan la salida a un rango específico (como la función sigmoide o softmax), se emplean funciones de activación lineales que permiten a la red producir una gama continua de valores. [23]

1.4.9.2 Herramientas para Redes Neuronales

1.4.9.2.1 Tensor Flow

Tensor Flow es una plataforma de código abierto desarrollada por Google para el aprendizaje automático y la inteligencia artificial. Su arquitectura flexible y eficiente permite a los usuarios desarrollar y entrenar modelos de machine learning y deep learning de manera escalable, desde aplicaciones simples hasta complejos sistemas de IA. Utilizando grafos computacionales, Tensor Flow facilita la distribución de cálculos en múltiples dispositivos, lo que lo hace adecuado para aplicaciones tanto en entornos de servidores como en dispositivos móviles. La biblioteca es conocida por su capacidad para trabajar con grandes volúmenes de datos, optimizar el rendimiento y adaptarse a diferentes tipos de redes neuronales, incluidas las redes convolucionales y recurrentes. La facilidad de uso de Tensor Flow, junto con su extensa documentación y comunidad activa, lo ha consolidado como una de las herramientas más utilizadas en el campo de la IA. [24]

Ilustración 16: Características de Tensor Flow [25]

1.4.9.2.2 Keras

Keras es una biblioteca de alto nivel para el desarrollo de redes neuronales, diseñada para ser fácil de usar, modular y extensible. Originalmente fue creada como una interfaz para diversas plataformas de deep learning, y más tarde se integró oficialmente en TensorFlow como su API principal para la construcción y entrenamiento de modelos de aprendizaje profundo. Keras facilita la creación de redes neuronales mediante una sintaxis sencilla, lo que permite a los desarrolladores y científicos de datos experimentar rápidamente con diferentes arquitecturas de redes. A pesar de su simplicidad, Keras proporciona suficiente flexibilidad para definir modelos avanzados, como redes convolucionales, recurrentes y redes de autoencoders. [26]

CAPÍTULO 2

2. METODOLOGÍA

En esta sección, se presenta el enfoque metodológico planteado para el desarrollo de este proyecto. Partiendo de la necesidad de mejorar la fiabilidad y eficiencia en el monitoreo de variables críticas, se evaluaron distintas alternativas de solución para seleccionar aquella que mejor se adecuara a los requerimientos del proyecto.

Este capítulo busca proporcionar una visión estructurada del proceso de diseño y desarrollo, asegurando que el sistema final cumpla con los estándares de calidad y funcionalidad necesarios para su integración en el entorno industrial.

2.1 Soluciones alternativas

Para abordar la problemática de la falta de un sistema confiable de monitoreo en la línea extrusora, se analizaron varias alternativas que permitieran una solución eficaz y sostenible a largo plazo. La primera opción considerada fue la adquisición de un sistema comercial preconfigurado, capaz de monitorear parámetros clave de la producción. Los sistemas comerciales presentan la ventaja de su rápida implementación y de contar con soporte técnico del proveedor; sin embargo, esta opción se limitaba en términos de personalización. Debido a las especificaciones de nuestra línea de producción y a la necesidad de integrar múltiples protocolos de comunicación como Profinet y MQTT, resultaba evidente que un sistema así requeriría adaptaciones significativas para satisfacer plenamente las necesidades del proyecto. Además, el costo asociado a un sistema preconfigurado podría representar una inversión poco rentable dada la naturaleza específica de nuestra aplicación.

Otra alternativa contemplada fue el desarrollo de un sistema completamente propio. Este enfoque permitiría diseñar cada componente y configurar los módulos de monitoreo específicamente para los requisitos de la línea extrusora. Esta opción ofrecía máxima flexibilidad y control sobre cada aspecto del sistema, lo que permitiría un ajuste preciso para cada protocolo de comunicación, variable y criterio de monitoreo. No obstante, el desarrollo de un sistema desde cero conllevaría altos costos y tiempos de implementación más extensos, además de requerir un equipo especializado para el diseño y mantenimiento de los componentes. Dado el nivel de inversión y los recursos técnicos que esta alternativa demandaría, su viabilidad fue reconsiderada en favor de opciones más accesibles en términos de implementación y sostenibilidad.

Finalmente, se evaluó la posibilidad de una solución híbrida, basada en la integración de un sistema semi-comercial adaptado mediante herramientas de software de código abierto, como Node-Red, Python y Grafana. Esta alternativa permitía combinar hardware comercial con software flexible e intuitivo, lo cual permitiría configurar las funciones de monitoreo y comunicación necesarias para la extrusora sin comprometer la eficiencia ni elevar significativamente los costos. Al integrar protocolos propios y herramientas modulares de software, este enfoque ofrecía una solución escalable y con un costo menor en comparación con un sistema comercial completo. Asimismo, esta opción brindaba una mayor adaptabilidad que un sistema preconfigurado, permitiendo realizar ajustes específicos sin reemplazar los componentes centrales.

Tras analizar las ventajas y desventajas de cada alternativa, se determinó que la última solución mencionada era la más adecuada para satisfacer los requerimientos del proyecto. Este enfoque permitía aprovechar los recursos disponibles y asegurar la integración efectiva con las tecnologías existentes, además de facilitar la actualización futura del sistema con nuevos módulos y herramientas de análisis predictivo. Así, la solución seleccionada ofrece un balance entre costo, flexibilidad y eficiencia, logrando satisfacer los objetivos de monitoreo necesarios para la operación confiable de la línea extrusora.

2.2 Diseño y metodología de la solución

El sistema desarrollado tiene como objetivo central ofrecer una solución confiable y automatizada para la adquisición y monitoreo de parámetros de interés en la línea extrusora de tubos. Este sistema abarca desde la captura de datos en tiempo real hasta su análisis visual y predictivo, permitiendo una gestión más precisa y eficiente del proceso de producción.

Para lograrlo, se ha implementado una infraestructura que integra diferentes dispositivos y herramientas de software que, trabajando en conjunto, recogen, procesan y almacenan datos para su posterior análisis. En primer lugar, el control de señales se realiza a través de un PLC LOGO!, que actúa como el principal equipo encargado de recoger datos sobre las variables esenciales, como el estado de "marcha" y "calentamiento" de la extrusora. Estos datos se transmiten de forma continua a una Raspberry Pi 3 mediante el protocolo de comunicación S7, donde son organizados y enviados por Node-RED a una Raspberry Pi 4.

La Raspberry Pi 4 tiene un papel fundamental en el almacenamiento de los datos, pues utiliza sentencias SQL para guardar en una base de datos MariaDB que asegura su persistencia y permite que estos puedan ser consultados y analizados. Además, mediante el uso del protocolo MQTT, los datos fluyen de manera eficiente a través de la red, posibilitando un acceso rápido y confiable.

Para el análisis y la visualización, se emplea la plataforma Grafana, que, conectada a la base de datos, permite generar gráficos e informes en tiempo real. Esta interfaz visual facilita a los operadores y al equipo de mantenimiento la identificación rápida de tendencias y el monitoreo continuo de los parámetros de funcionamiento. De esta manera, cualquier desviación o alteración de los valores normales puede ser detectada a tiempo, evitando problemas en la calidad del producto y mejorando el uso de energía.

Finalmente, se ha implementado un sistema de mantenimiento predictivo mediante redes neuronales, que procesa los datos históricos para prever posibles fallos y reducir el tiempo de inactividad. Con esta solución, se busca no solo mejorar la eficiencia del proceso productivo, sino también proporcionar una herramienta inteligente que anticipe necesidades de mantenimiento preventivo y correctivo.

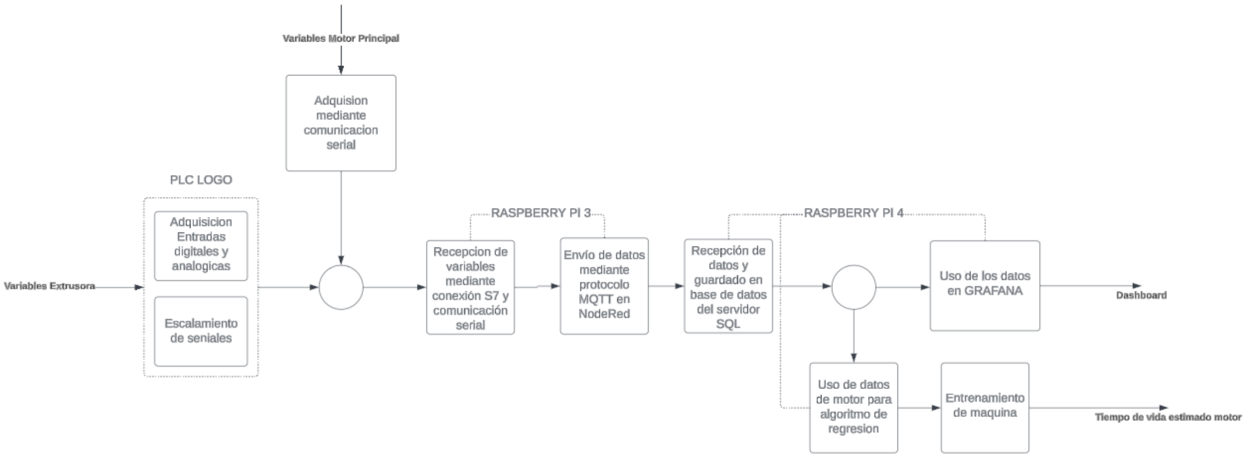


Ilustración 17: Diagrama de bloques del Sistema Smart

2.3 Selección de materiales y equipos

Para el desarrollo e implementación del sistema de monitoreo se vieron involucrados diferente equipos y materiales que se detallan a continuación.

Tabla de equipos:

EQUIPO	CANT	MODELO	FUNCION	DESCRIPCION GRAFICA
PLC LOGO!	1	8.2	Adquirir, gestionar y distribuir los datos o información recolectada de sensores	
Módulo de expansión analógica	2	AM2	Gestión de entradas analógicas de la programación	
Módulo de expansión RTD	1	AM2 RTD	Modulo para la gestión y adquisición de temperatura de proceso	
Fuente de alimentación	1	Logo Power	Encargada de la distribución y alimentación de energía necesaria.	

Switch de distribución de red	1	SCALANCE XC208	Switch que genera una red cerrada para comunicar al logo con diferentes dispositivos adicionales.	
Rasberry	5	Pi 4	Encargadas de ser los dispositivos que reciban la información que adquiera el PLC logo y procesarla.	
Sensor de temperatura	1	PT100	sensor analógico que permite tomar lectura entre -200°C - 850°C	
Variador de frecuencia	1	Danfoss NPX Vacon	Adquirir información de RPM y Corriente del motor de la extrusora	

Tabla 1: Componentes de equipos para la implementación

Componentes y materiales eléctricos.

EQUIPO	CANTIDAD	MODELO	DESCRIPCION GRAFICA
Tablero eléctrico	1	General Electric	
Cables de control	30m	18AWG B/A/V	
Borneras rápidas	30	Tipo tornillo	
Riel din	1		
Tornillos autoperforantes	1 caja	Tipo estrella - ½”	
Terminales eléctricos	1 caja	Tipo punta – 18awg	

Etiquetadora			

Tabla 2: Componentes eléctricos y herramientas

2.4 Asignación de Direcciones IP

A continuación, se detallan las direcciones IP asignadas a los diferentes equipos que componen el sistema SMART, el cual consta del PLC LOGO y las Raspberry:

Equipo	Dirección IP
PLC LOGO	Ethernet: 192.XXX.XX.203
Raspberry Pi 3 Tablero	Ethernet: 192.XXX.XX.205
	WLAN: 10.XX.XX.23
Raspberry Pi 4 Broker MQTT	WLAN: 10.XX.XX.33
Raspberry Pi 4 Base de datos	WLAN: 10.XX.XX.32
Raspberry Pi 4 Oficina de Producción	WLAN: 10.XX.XX.34
Raspberry Pi 3 Alarmas	WLAN: 10.XX.XX.12

Tabla 3: Direcciones IP de equipos del sistema SMART

2.5 Procedimiento para la implementación

2.5.1 Programación del PLC LOGO!

En la automatización de una línea extrusora de tubos, el PLC LOGO! juega un papel esencial al actuar como el controlador de señales y procesador de datos que son críticos para la operación. Con su capacidad para monitorear variables clave en tiempo real y enviar información a otros dispositivos en el sistema, el PLC LOGO! se convierte en un elemento crucial para garantizar una adquisición eficiente y confiable.

La programación del PLC LOGO! se centra en la gestión de dos variables esenciales: el estado de "marcha" y el estado de "calentamiento". Estos parámetros son fundamentales para controlar y supervisar la extrusora, ya que reflejan si el sistema está en funcionamiento y si las condiciones del proceso son óptimas para la producción. Para captar estas señales, se configuraron entradas digitales en el PLC LOGO!, donde cada entrada está asociada a interruptores específicos que indican el estado de estas variables en el tablero propio de la extrusora.

Además, el sistema cuenta con temporizadores programados que miden el tiempo de producción y el tiempo de calentamiento en intervalos de horas, ofreciendo así un control más detallado del ciclo de producción. Los temporizadores activan y desactivan contadores acumulativos cada vez que el sistema detecta que el estado de "marcha" o "calentamiento" está activo. Estos contadores permiten llevar un registro acumulativo de las horas de operación y de calentamiento, datos que posteriormente son mostrados para un análisis operacional de la máquina. Con esta funcionalidad, se asegura que el sistema cuente con información precisa sobre el tiempo total de funcionamiento, un aspecto esencial para prever y evitar fallas en el equipo.

Un aspecto esencial de la programación del PLC LOGO! es el manejo de alarmas y medidas de seguridad. La línea extrusora debe operar dentro de parámetros predefinidos para evitar daños en el equipo y asegurar la calidad del producto final. Con esto en mente, se programaron bloques de comparación que contrastan continuamente los valores de las variables clave con sus límites aceptables. Si una variable, como el tiempo de calentamiento, supera el límite seguro, el sistema activa automáticamente una señal de alarma. Esta alarma, conectada a una salida digital, proporciona una advertencia visual que alerta a los operadores sobre la necesidad de realizar ajustes o intervenir en el proceso. De este modo, el sistema no solo garantiza la precisión en la captura y transmisión de datos, sino que también incluye una capa de seguridad que protege tanto la máquina como el proceso de producción.

Finalmente, tras programar estos bloques y configurar el sistema de comunicación, se llevaron a cabo pruebas para verificar el correcto funcionamiento de la lógica y asegurar la precisión en los datos recolectados.

2.5.2 Configuración de Node-Red en Raspberry Pi 3

La comunicación entre el PLC LOGO! y la Raspberry Pi 3, encargada de centralizar los datos, se realiza mediante el protocolo S7. Este protocolo garantiza una transferencia de datos confiable y permite que la información fluya en intervalos regulares, manteniendo la base de datos siempre actualizada. La programación del PLC incluye bloques específicos de comunicación S7 que facilitan esta interacción. Cada dato capturado y procesado por el PLC se asigna a una dirección específica y se transmite en ciclos de un segundo. Esta frecuencia de actualización asegura que los datos estén siempre disponibles para el análisis y la visualización en tiempo real en la plataforma de monitoreo Grafana.

2.5.3 Creación de base de datos MariaDB

En el diseño del sistema de monitoreo de la línea extrusora, la Raspberry Pi 4 desempeña un papel clave como servidor de base de datos. Utilizando MariaDB, se ha creado una infraestructura que permite gestionar de manera eficiente los datos generados por el PLC LOGO! y otros componentes del sistema. Esta base de datos actúa como el núcleo donde se almacenan y organizan las variables esenciales del proceso de extrusión.

MariaDB fue seleccionada por su rendimiento, flexibilidad y compatibilidad con dispositivos de bajo consumo como la Raspberry Pi. Su arquitectura ligera asegura que, incluso en un entorno con recursos limitados, el servidor pueda manejar grandes volúmenes de datos y múltiples consultas simultáneas sin comprometer la velocidad ni la estabilidad.

La base de datos está diseñada para registrar de forma estructurada los datos clave, como el tiempo de marcha y de calentamiento, junto con marcas de tiempo que permiten rastrear y analizar el comportamiento de la línea extrusora a lo largo del tiempo. Esto resulta fundamental para habilitar funcionalidades avanzadas como el mantenimiento predictivo y la optimización del rendimiento de la línea.

La Raspberry Pi 4, al actuar como servidor SQL (lenguaje de dominio de MariaDB), permite que otros dispositivos del sistema, como las Raspberry y las plataformas de visualización como Grafana, accedan a los datos almacenados de manera segura y eficiente. A través de protocolos estándar, como TCP/IP, los datos fluyen desde los puntos de adquisición (PLC LOGO! y sensores) hasta MariaDB, donde son procesados y puestos a disposición para su análisis.

Este enfoque no solo centraliza el almacenamiento de datos, sino que también permite una escalabilidad significativa. A medida que se identifiquen nuevas variables o se requiera incorporar nuevas funcionalidades, el diseño de la base de datos puede adaptarse sin necesidad de cambios estructurales significativos. Esto asegura que el sistema siga siendo funcional y relevante a largo plazo, respondiendo a las necesidades de monitoreo y análisis de la línea extrusora.

Dentro de la base, se crean tablas para poder guardar la información; una llamada SMART_MOTOR42, donde se guardan los datos para el entrenamiento del algoritmo de regresión, y otra llamada PARAM_EXT42, donde se almacenarán el resto de los datos:

[redacted]	[redacted]
tiempo	Tiempo
TEMP_MOTOR	ESTADO
VibX_MOTOR	PRES_MASA
VibZ_MOTOR	TEMP_MATERIAL
AMP_MOTOR	VIRUTA_A
T_PROD	VIRUTA_B
T_T_PROD	T_CALENTAMIENTO
RPM_MOTOR	
POT_ACTIVA	

Tabla 4: Tablas de la base de datos

2.5.4 Desarrollo del Script Python para el sensor predictivo QM30VT2

La Raspberry Pi 3 también se encarga de la adquisición y procesamiento de información proveniente del sensor QM30VT2. Este dispositivo, diseñado para medir vibraciones y temperatura, proporciona datos esenciales para evaluar el estado del motor de la extrusora, contribuyendo así al mantenimiento predictivo y al análisis de condiciones operativas críticas.

El sensor QM30VT2 se conecta a la Raspberry Pi 3 a través de un convertidor Modbus RS485 a USB, utilizando comunicación serial para transmitir datos en tiempo real. Esta conexión permite a la Raspberry Pi 3 recibir información bruta directamente desde el sensor, que incluye valores relacionados con la temperatura del motor y vibraciones en los ejes X y Z. Sin embargo, los datos en su estado original son crudos y no entendibles por el ser humano; por ende, requieren procesamiento para ser útiles en el sistema.

Mediante un script desarrollado en Python, la Raspberry Pi 3 procesa estos datos antes de transmitirlos al sistema general. Este script establece la comunicación con el sensor utilizando librerías especializadas, interpreta los valores obtenidos, y los convierte a unidades comprensibles aplicando fórmulas de transformación específicas proporcionadas por el fabricante del sensor. Por ejemplo, los registros hexadecimales leídos se transforman en valores de temperatura en grados Celsius o en mediciones de vibración expresadas en términos de velocidad en mm/s.

Una vez que los datos han sido procesados, son transmitidos a la Raspberry Pi 4 ubicada en la oficina de producción mediante el protocolo MQTT. Cada conjunto de datos, como la temperatura del motor o las vibraciones en los ejes X y Z, se publica en tópicos específicos, lo que permite que los componentes posteriores del sistema identifiquen y procesen cada variable de manera independiente.

En la Raspberry Pi 4, los datos recibidos se integran al flujo general del sistema a través de Node-RED, donde se procesan y almacenan en la base de datos MariaDB. Además, estas variables se visualizan en tiempo real en dashboards configurados en Grafana, proporcionando a los operadores una visión detallada del estado del motor de la extrusora. Estos datos al ser guardados en la base de datos, específicamente en la tabla de SMART_MOTOR42, pueden ser utilizados para el entrenamiento del algoritmo de regresión para obtener los datos predictivos necesarios para su posterior análisis de parte de los supervisores de la línea de extrusión y que estos puedan ejecutar mantenimientos de prevención de ser el caso.

2.5.5 Configuración de Node-Red y Grafana en Raspberry Pi 4

En el sistema de monitoreo y adquisición de datos de la línea extrusora, la Raspberry Pi 4 ubicada en la oficina de producción desempeña un papel fundamental como intermediaria entre las etapas de adquisición de datos y su almacenamiento y visualización. Esta unidad es responsable de recibir los datos transmitidos desde la Raspberry Pi 3 mediante el protocolo MQTT, procesarlos, y finalmente almacenarlos en la base de datos MariaDB ubicada en el servidor central.

La Raspberry Pi 4 está configurada como un cliente MQTT que escucha continuamente los datos publicados por la Raspberry Pi 3. Estos datos contienen información procesada y transmitida desde el PLC LOGO!, como el tiempo de marcha, el tiempo de calentamiento y otras variables relevantes. Al utilizar MQTT, se garantiza un método de transmisión eficiente, ligero y confiable, especialmente adecuado para redes industriales.

Una vez recibidos, los datos son decodificados y procesados en la Raspberry Pi 4. Esto incluye tareas como verificar la integridad de los datos, convertirlos a formatos compatibles con la base de datos, y asignarlos a las variables correspondientes antes de su almacenamiento.
La Raspberry Pi 4 se comunica con la base de datos MariaDB alojada en otra Raspberry Pi para almacenar la información de manera estructurada. Este proceso se realiza mediante Node-RED, que actúa como una herramienta intermedia para gestionar los flujos de datos. Un nodo específico de MySQL en Node-RED inserta los datos en las tablas correspondientes, siguiendo el diseño previamente establecido en la base de datos.Además de procesar y almacenar datos, la Raspberry Pi 4 en la oficina de producción tiene instalado Grafana, una plataforma para la creación de dashboards interactivos.

Grafana está configurado para conectarse directamente a la base de datos MariaDB y consultar los datos almacenados en tiempo real. Esto permite generar gráficos e informes que ofrecen una visión clara y detallada del estado de la línea extrusora. Los dashboards en Grafana incluyen indicadores clave, como gráficos del tiempo acumulado de marcha y calentamiento, alertas sobre posibles desviaciones en los parámetros de operación, y tendencias históricas de las variables críticas. Esta funcionalidad no solo facilita el monitoreo, sino que también permite a los operadores y supervisores tomar decisiones informadas basadas en datos precisos y actualizados.

Este diseño modular y distribuido garantiza que el sistema sea flexible, escalable y resiliente, permitiendo que se adapte a las necesidades cambiantes de la producción. Además, su capacidad para presentar los datos de manera visual mejora significativamente la capacidad de los operadores para monitorear y gestionar la línea extrusora de manera proactiva.

2.5.6 Desarrollo del Script Python para las alarmas auditivas

En el sistema de monitoreo de la línea extrusora, una segunda Raspberry Pi 3 ubicada en la oficina de producción está dedicada a generar alarmas auditivas cuando el tiempo de calentamiento de la línea alcanza o supera las 4 horas. Este mecanismo de alerta busca notificar de manera inmediata a los operadores sobre condiciones anómalas que podrían impactar la eficiencia energética o la calidad del proceso.

Esta Raspberry Pi 3 recibe desde la Raspberry Pi 4 un mensaje MQTT que indica una condición de alarma. Este mensaje, enviado desde Node-RED, se activa cuando el tiempo de calentamiento supera el umbral establecido de 4 horas.

Cuando la Raspberry Pi 3 recibe un mensaje true a través de MQTT, su script en Python ejecuta un archivo de audio predefinido utilizando VLC, un reproductor multimedia robusto y compatible con múltiples formatos. VLC está configurado para reproducir el sonido a través de la salida analógica de la Raspberry Pi, garantizando que la señal pueda ser amplificada para su distribución en el entorno industrial.

La salida analógica de la Raspberry Pi está conectada a la entrada AV de un amplificador, el cual aumenta la intensidad de la señal antes de enviarla a una bocina. Este diseño asegura que las alarmas sean claramente audibles en el área de producción, incluso en condiciones de ruido elevado.

Este enfoque distribuye la lógica del sistema de manera eficiente: la Raspberry Pi 4 concentra el procesamiento y la toma de decisiones sobre el tiempo de calentamiento, mientras que la Raspberry Pi 3, la cual es de menor capacidad de procesamiento, se especializa simplemente en la reproducción de las alarmas. Esto no solo simplifica el diseño general, sino que también facilita el mantenimiento y la escalabilidad del sistema.

La implementación de este sistema refuerza el monitoreo integral de la línea extrusora, asegurando que cualquier desvío significativo en los tiempos de operación sea detectado y comunicado de inmediato.

2.5.7 Entrenamiento del Algoritmo de Regresión

El sistema de monitoreo contara con un subsistema de predicción de la vida restante del motor eléctrico de la línea de extrusión, este sistema está basado en una de las ramas de la inteligencia artificial tal como son las redes neuronales, la red implementada se ejecutara en un dispositivos adicional y no en los dispositivos anteriormente mencionado en este documento, esto debido a que la capacidad de procesamiento de los dispositivos terminales (raspberry) no cuentas con el poder computacional que se necesita para realizar una red que cumpla con estándares de eficacia y confiabilidad; por lo cual todo el proceso de entrenamiento e implementación de la red se lo llevara a cabo en un nuevo dispositivo (laptop anidada a red) con el fin de que el sistema se encuentre completamente comunicado con todos los dispositivos terminales y reciba la información necesaria para ejecutar el proceso de la red neuronal. Profundizando en la red, se da a conocer que la implementación se llevará a cabo con la ayuda de software de programación “Visual Studio Code”, donde se procederá a la instalación de las plataformas necesarias: Python, Tensor Flow, Keras, Pandas, numpy, sklearn, matplotlib, etc.; estas será las herramientas que darán vida al entrenamiento y posteriormente al modelo de la red.

2.5.7.1 Proceso de entrenamiento

En primera instancia lo primero que sea realizara es la recolección de información mediante una base de datos que se generara a partir de los datos recolectados de este proyecto; se tiene estimado un número aproximado de 8000 datos necesarios para que la red pueda considerar eficaz en su labor, ya que modelos o redes que cuentas con bases menos densa son consideradas redes sencillas y de poca credibilidad debido a la falta de información. A continuación, se muestra el modelo de base que obtendrá, cabe recalcar que la tabla de la base mostrada a continuación solo es un ejemplo de los datos que se obtendrá y se procesaran posteriormente.

Con la base obtenida el paso siguiente es realizar una purificación de la base, esto se refiere a que posiblemente la base sufra cambios para adaptase al modelo de red aumentando o disminuyendo. Posteriormente se comenzará con el proceso de identificación de entradas y datos objetivos (salidas) las cuales entran que atravesar una escalamiento y normalización de datos con el fin de simplificar el procesamiento de información y el aprendizaje de la red. Despuesta de procesar la información la red pasa a proceso de definición del tipo de red, donde se implementa el diseño y características, tales como numero de neuronas, número de entradas y numero de salidas entre otros. Para finalizar el proceso, se procederá al entrenamiento y visualización resultados para brindar información relevante sobre la red y verificar su eficacia y veracidad.

CAPÍTULO 3

3. RESULTADOS Y ANÁLISIS

En este capítulo se presentan y analizan los resultados obtenidos tras la implementación del sistema desarrollado para la adquisición y monitoreo de parámetros en la línea de extrusión de tuberías plásticas. Se describen de forma clara y concisa los datos generados por el sistema, destacando aquellos más representativos y relevantes para evaluar su desempeño. Además, se detallan las evidencias que respaldan el cumplimiento de los objetivos propuestos, relacionándolos con la solución planteada.

Asimismo, se incluye un análisis detallado de los costos asociados al desarrollo e implementación de la solución, evaluando su viabilidad económica y tecnológica. Este análisis permite determinar la sostenibilidad del sistema en el contexto industrial y valorar su impacto en términos de mejora de la producción y optimización de recursos.

3.1 Análisis de Costos

ITEMS	DENOMINACIÓN	CANTIDAD	UNIDAD DE MEDIDA	PRECIO UNITARIO	TOTAL
1	**PLC Logo 8.2**	1		$400.00	$400.00
2	**LOGO! Power**	1		$100.00	$100.00
3	**LOGO! DM16 24R**	1		$200.00	$200.00
4	**LOGO! AM2**	2		$150.00	$300.00
5	**LOGO! AM2 RTD**	1		$185.00	$185.00
6	**LOGO! CSM (ethernet switch)**	1		$155.00	$155.00
7	**Borneras (caja 100 unidades)**	1		$10.00	$10.00
8	**Breaker 1 polo 15amp**	2		$20.00	$40.00
9	**Canaletas**	1	m	$15.00	$15.00
10	**Tornillos autoperforantes**	100		$0.50	$50.00
11	**Riel din**	3	m	$10.00	$30.00
12	**Caja electrica**	1		$100.00	$100.00
13	**Ap delta - DVW-W02W2-E2-EU**	1		$300.00	$300.00
14	**Raspberry Pi 4**	5		$200.00	$1,000.00
15	**RTD PT100**	1		$100.00	$100.00
16	**VDF VAGON**	1		$1,500.00	$1,500.00
17	**Sensor Banner qm30vt2**	1		$400.00	$400.00
18	**Adaptador RS485 a USB**	1		$15.00	$15.00
19	**Switch Siemens Scalance XC208**	1		$800.00	$80 0.00
20	**Terminales electricos 18Awg**	100		$0.15	$15.00

Subtotal	$5,715.00
I.V.A (15%)	$857.25
TOTAL	$6,572.25

Tabla 5: Costos de los Materiales, Equipos, y su Total

En la tabla 5 se contempla todos los dispositivos y materiales que ayudan a conformar el proyecto, si bien algunos de estos ya se encontraban instalados al momento de implementar este trabajo, también se los consideran para corroborar que tan factible es implementarlo desde cero. SI bien es una cantidad considerable de presupuesto, es mucho más accesible que otros métodos más robustos como pueden ser sistemas compuestos puramente por PLC en vez de Raspberry, lo cual se traduce en un ahorro económico significativo para la industria.

3.2 Análisis de la Implementación

3.2.1 Diseño e Instalación del Tablero Eléctrico

La distribución de los elementos en el tablero eléctrico se realizó como muestra la siguiente figura:

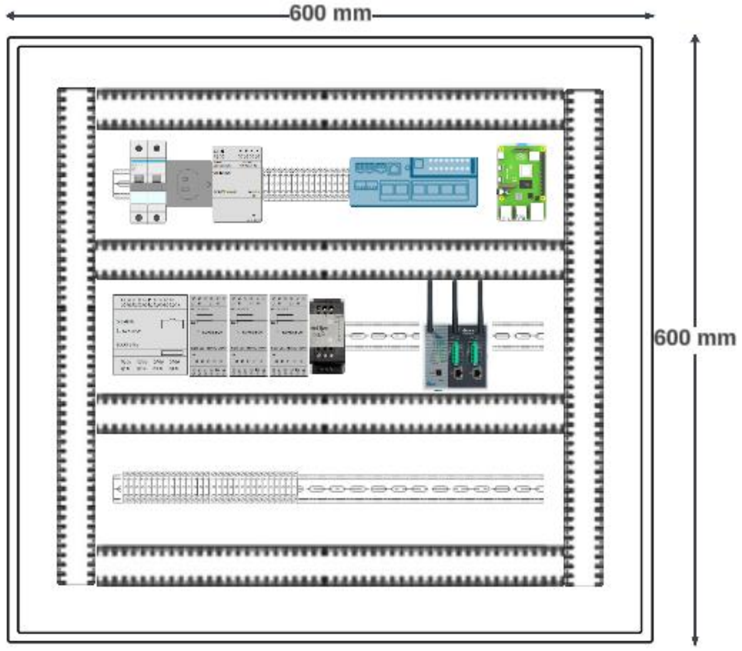


Ilustración 18: Representación 2D del Tablero Eléctrico

Se puede observar la presencia del PLC LOGO! con sus respectivos módulos los cuales facilitaron la adquisición de las diferentes señales analógicas provenientes de la línea de extrusión, además de la Raspberry Pi 3 que se encargó de recibir los datos del PLC y del sensor QM30VT2 para transmitirlos por MQTT. Se tiene la fuente de 24 VDC LOGO! POWER para alimentar los dispositivos en el tablero. Además, se cuenta con un switch SCALANCE XC2208 para conectar la Raspberry Pi, el PLC LOGO! y el AP DELTA mediante cable Ethernet a la misma subred.
La instalación del tablero quedó como se muestra a continuación:

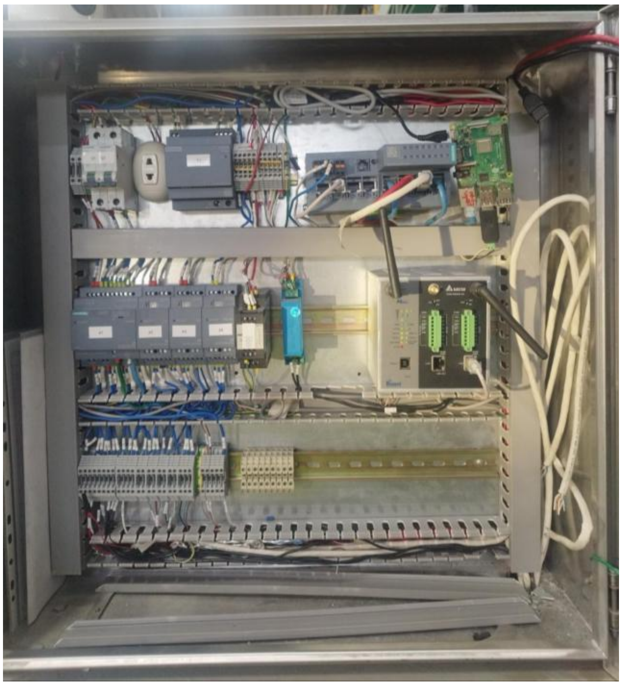


Ilustración 19: Tablero Eléctrico instalado

3.2.2 Programación del PLC LOGO

Para la programación del PLC, se utilizó el software LOGO!Soft Comfort V8.3, en el cual se utilizaron diferentes bloques para el diseño lógico que permitió una gestión eficiente de las variables clave del proceso, tanto las señales digitales como las analógicas, asegurando una captura y procesamiento precisos de los datos operativos.
Los bloques de entrada digital fueron configurados para recibir las señales de marcha y calentamiento. Estas señales determinaron el estado operativo de la línea y permitieron activar los contadores asociados. Por ejemplo, cuando la señal de marcha estaba activa,

se habilitaba el contador del tiempo de producción, mientras que la señal de calentamiento activaba el contador correspondiente a este parámetro. La lógica programada se encuentra en las figuras 20, 21 y 22:

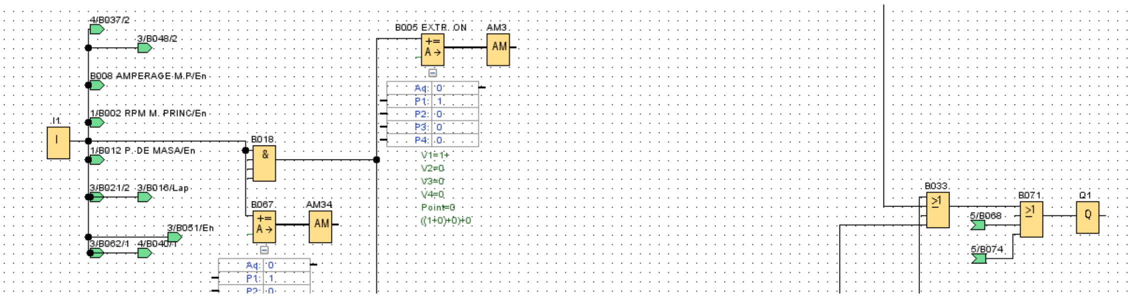


Ilustración 20: Entrada digital 1 y señal de Marcha

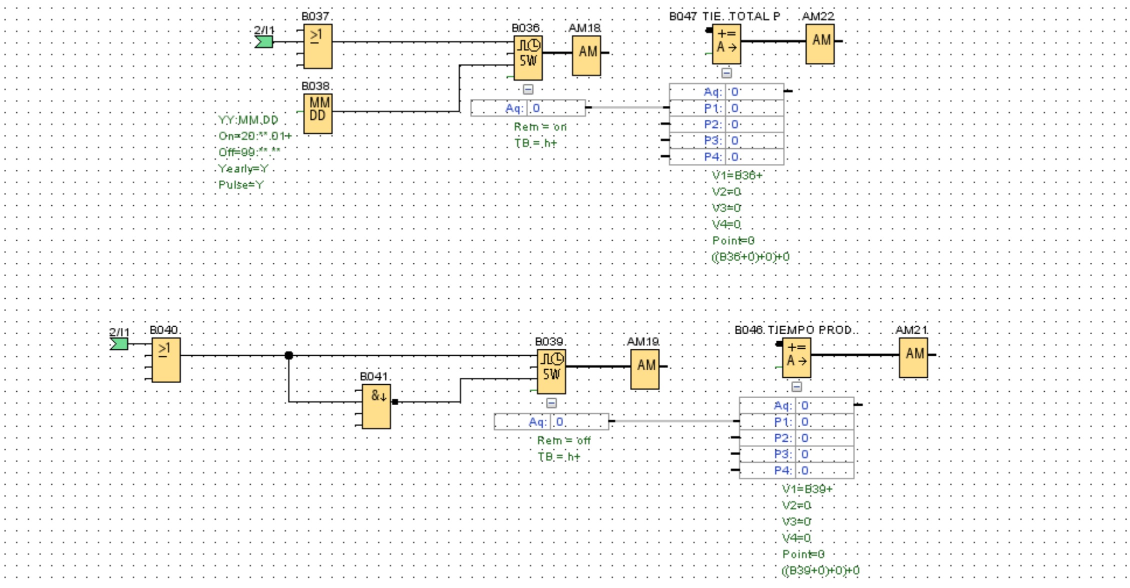


Ilustración 21: Variables de Tiempo de Producción Actual y Total (Anual)

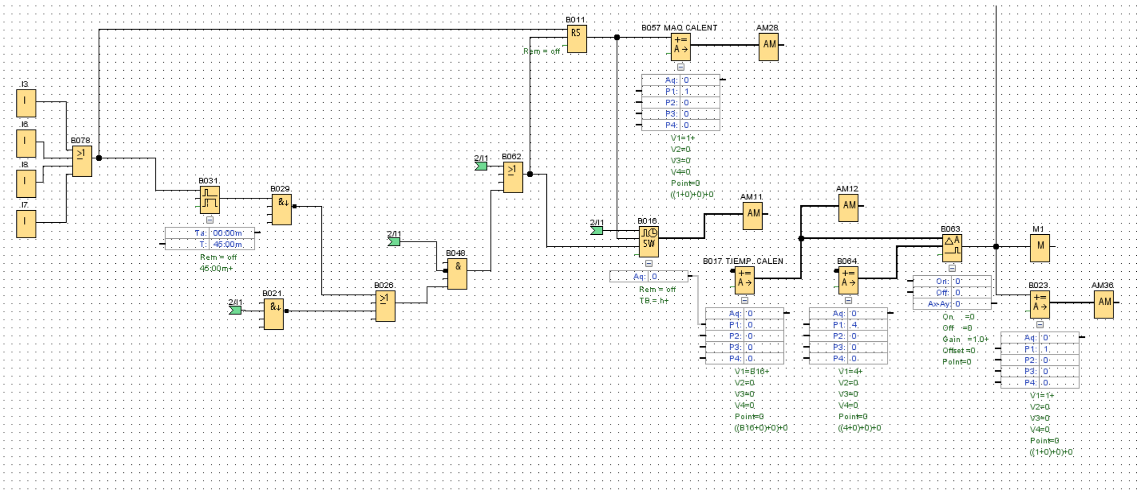


Ilustración 22: Entradas digitales de Calentadores y señal de Calentamiento, Tiempo de Calentamiento

En cuanto a los contadores, se implementaron bloques que registraron tres variables principales:

- Tiempo de producción: Este contador acumuló las horas durante las cuales la línea estuvo operativa en una sola sesión, proporcionando una medida del tiempo efectivo de trabajo.
- Producción total: Igual que el anterior, pero este se reinicia cada año de producción.
- Tiempo de calentamiento: Este contador acumuló las horas de calentamiento en una sola sesión, permitiendo identificar posibles excesos que pudieran afectar la eficiencia energética o la calidad del producto.

En el ámbito de las entradas analógicas, se conectaron sensores que proporcionaron datos en tiempo real de variables críticas del proceso. Estas señales incluyeron:

- RPM del motor: Monitorizadas para asegurar que la velocidad de operación se mantuviera dentro de los rangos establecidos.
- Amperaje: Indicador de la carga eléctrica del motor, útil para detectar sobrecargas o ineficiencias.
- Temperatura del material: Medida fundamental para garantizar que el material extrusado alcanzara la viscosidad adecuada para su moldeado.
- Presión de masa: Variable crucial para controlar la calidad de las tuberías producidas, asegurando que no hubiera obstrucciones o irregularidades en el flujo del material.

- Lecturas de virutas A y B: Utilizadas para detectar los residuos de las cortadoras A y B de la línea extrusora, las cuales deben ser retiradas después de cierto límite de masa.

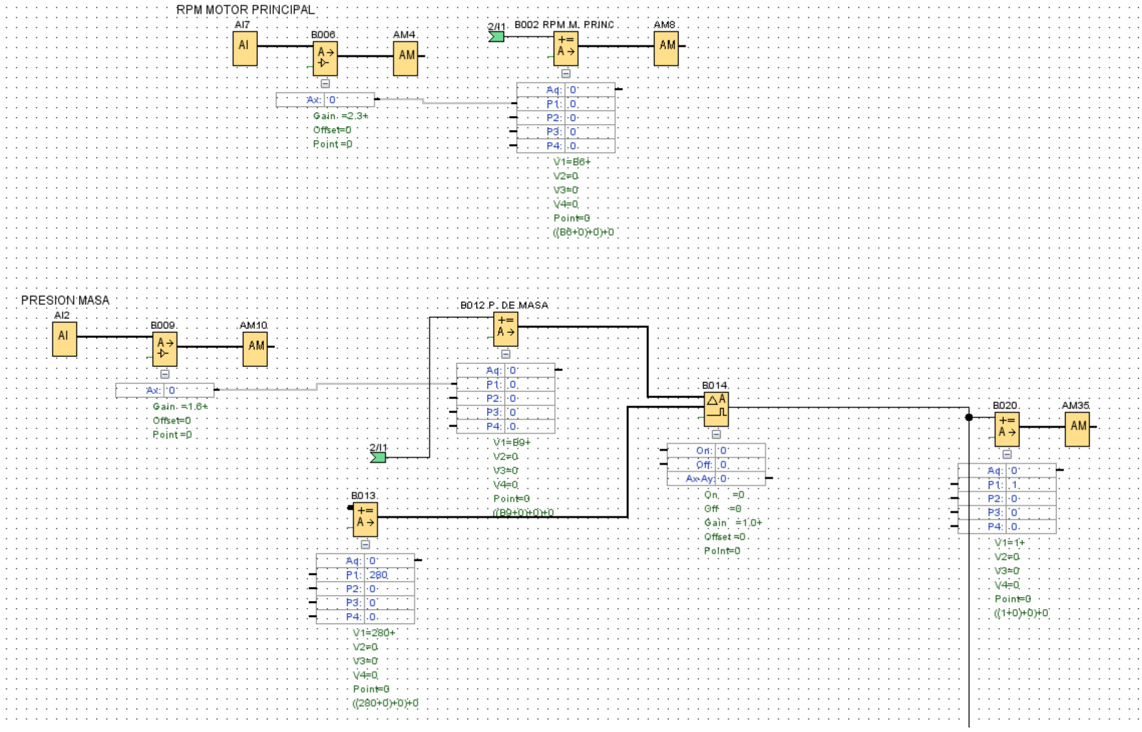


Ilustración 23: Señales de RPM de Motor Principal y Presión de Masa

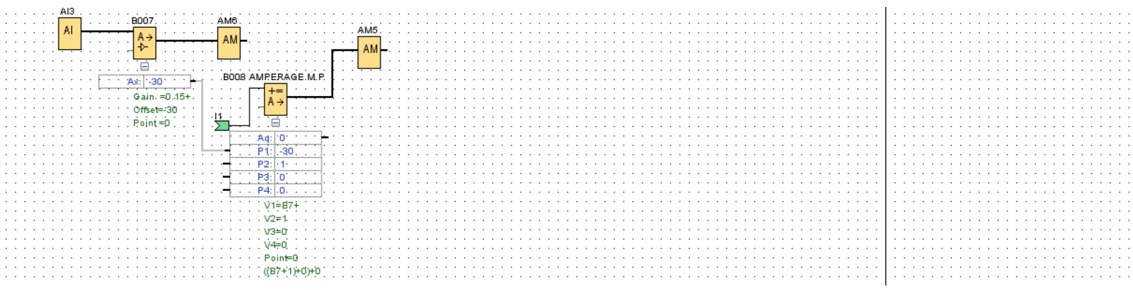


Ilustración 24: Señal de Amperaje del Motor Principal

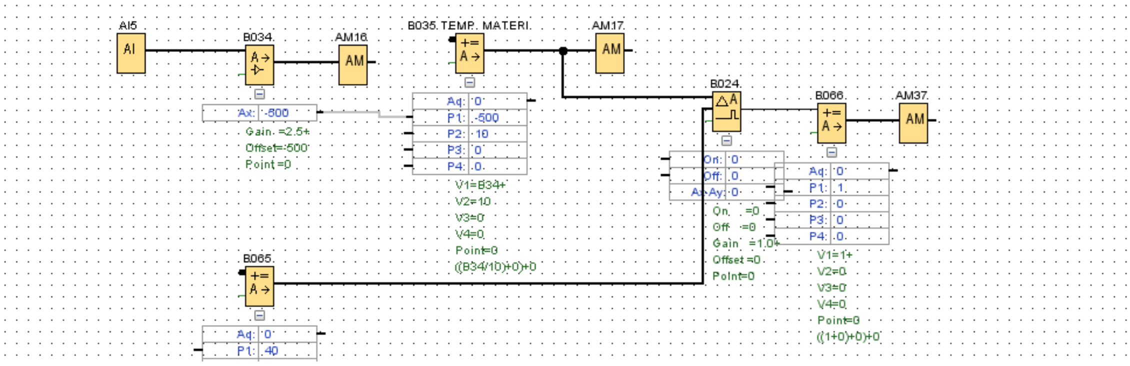


Ilustración 25: Señal de Temperatura de Material

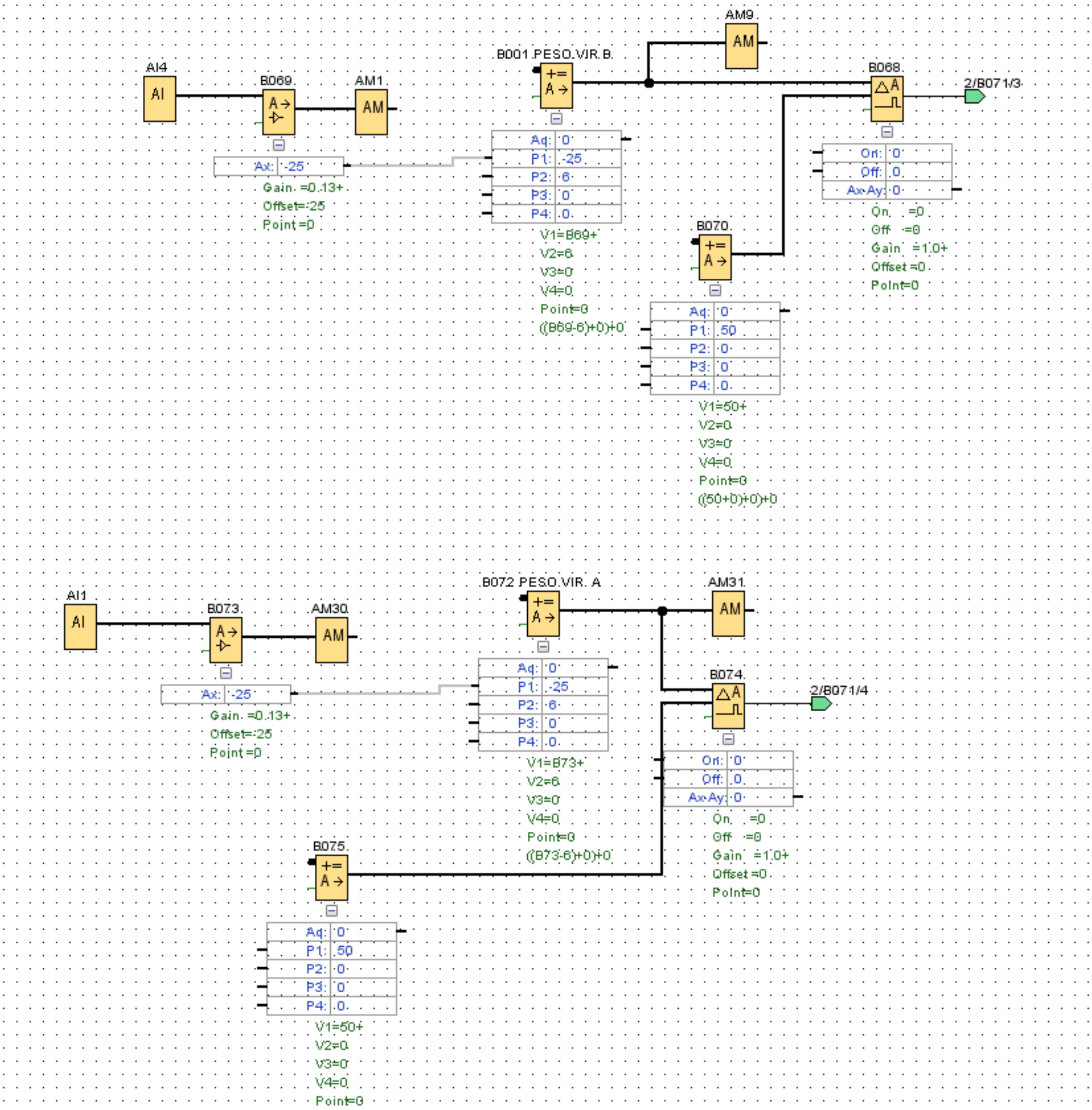
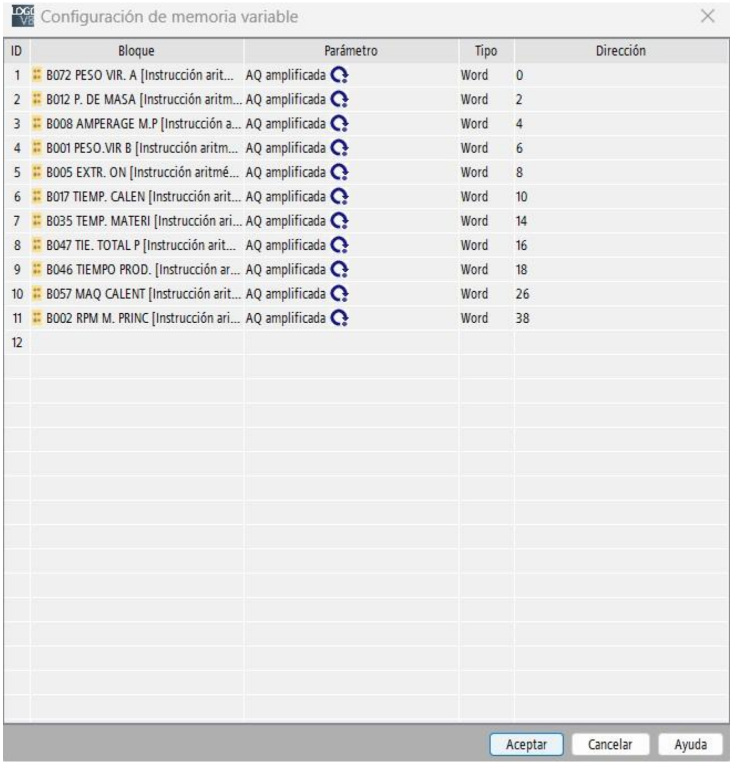


Ilustración 26: Ilustración 26: Señales de Virutas A y B

La programación del PLC LOGO! integró estas señales y contadores de forma modular, permitiendo su transmisión mediante el protocolo S7 hacia la Raspberry Pi 3 del tablero con ayuda del cable Ethernet. En la ilustración 27 se tienen las direcciones de esas variables:


Configuración de memoria variable

ID	Bloque	Parámetro	Tipo	Dirección
1	B072 PESO VIR. A [Instrucción arit...	AQ amplificada	Word	0
2	B012 P. DE MASA [Instrucción aritm...	AQ amplificada	Word	2
3	B008 AMPERAGE M.P [Instrucción a...	AQ amplificada	Word	4
4	B001 PESO.VIR B [Instrucción aritm...	AQ amplificada	Word	6
5	B005 EXTR. ON [Instrucción aritmé...	AQ amplificada	Word	8
6	B017 TIEMP. CALEN [Instrucción arit...	AQ amplificada	Word	10
7	B035 TEMP. MATERI [Instrucción ari...	AQ amplificada	Word	14
8	B047 TIE. TOTAL P [Instrucción arit...	AQ amplificada	Word	16
9	B046 TIEMPO PROD. [Instrucción ar...	AQ amplificada	Word	18
10	B057 MAQ CALENT [Instrucción arit...	AQ amplificada	Word	26
11	B002 RPM M. PRINC [Instrucción ari...	AQ amplificada	Word	38
12				

Ilustración 27: Direcciones de las variables en software LOGO! SoftComfort

Los resultados evidenciaron que el PLC LOGO! gestionó de manera confiable las variables críticas del proceso, proporcionando una base sólida para la adquisición de datos y contribuyendo significativamente a la mejora del control y la supervisión de la línea extrusora. Su diseño simplificó la integración con los sistemas posteriores, asegurando una operación fluida y eficiente en todo momento.

3.2.3 Programación en Node-Red en Raspberry Pi 3 del Tablero

La Raspberry Pi 3 fue configurada para establecer una comunicación directa mediante un cable Ethernet utilizando el protocolo S7, lo que permitió recibir las variables del proceso de manera eficiente y confiable.

En el entorno de Node-RED, se implementaron nodos específicos para la comunicación con el PLC. El nodo S7 fue configurado con la dirección IP del PLC LOGO! y las direcciones de memoria correspondientes a cada variable a monitorear, como se muestran en las imágenes 28 y 29. Este enfoque permitió que la Raspberry Pi 3 capturara las señales digitales y analógicas previamente procesadas en el PLC.

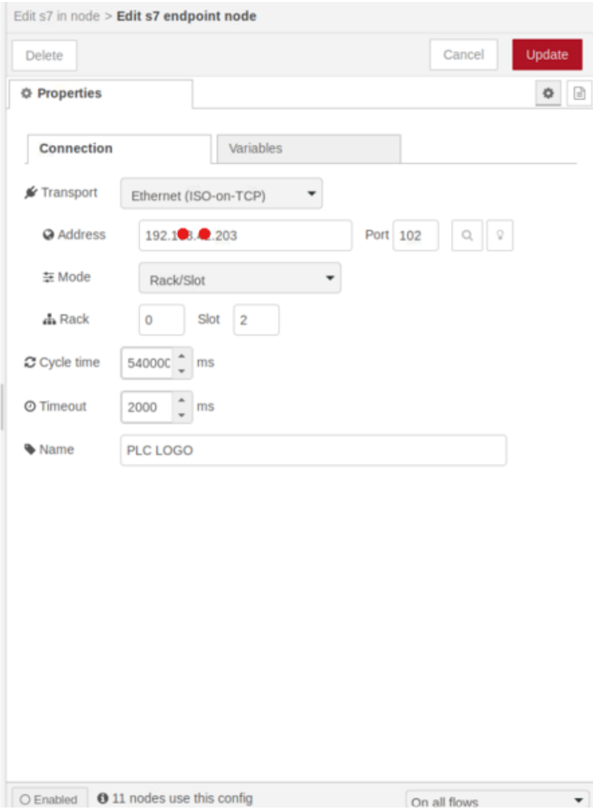


Ilustración 28: Configuración de la conexión con el PLC en Nodo S7

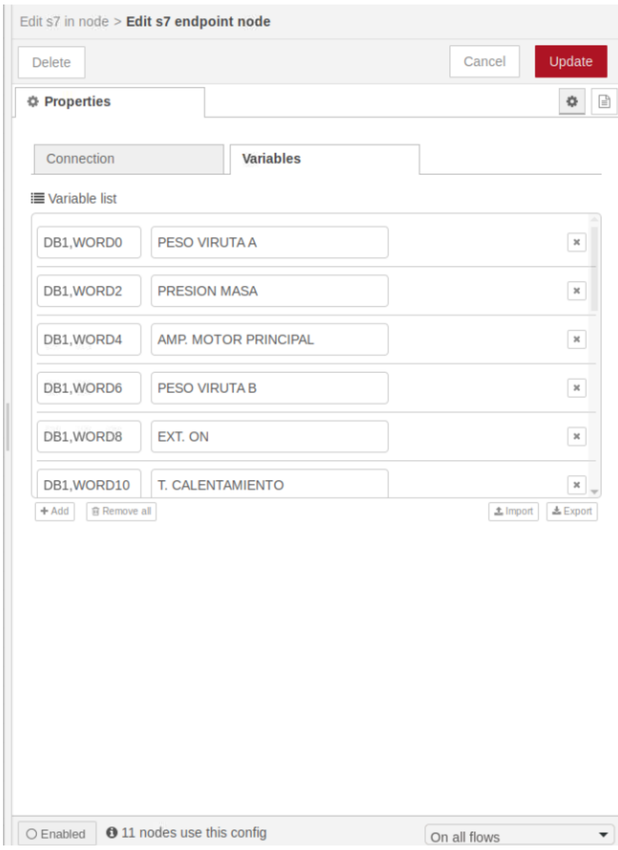


Ilustración 29: Configuración de direcciones de las variables en el Nodo S7

Las direcciones colocadas en el nodo S7 de las variables que provienen del PLC se listan a continuación:

Nombre de la variable	Dirección en Nodo S7
PESO VIRUTA A	DB1,WORD0
PRESION MASA	DB1,WORD2
AMP. MOTOR PRINCIPAL	DB1,WORD4
PESO VIRUTA B	DB1,WORD6
EXT. ON	DB1,WORD8
T. CALENTAMIENTO	DB1,WORD10
TEMP. MATERIAL	DB1,WORD14
T. TOT. PRODUCCION	DB1,WORD16
T. ACTUAL PRODUCCION	DB1,WORD18
SEÑAL CALENTAMIENTO	DB1,WORD26
RPM MOTOR	DB1,WORD38

Tabla 6: Direcciones de las variables en Nodo S7 de NodeRed

Una vez que las variables fueron recibidas, utilizando nodos MQTT, estas se publicaron en tópicos específicos, asegurando el envío de la información hacia los demás componentes del sistema. El bróker MQTT se configuró como muestra la figura 30. Cada variable se asignó a un tópico único, siguiendo una estructura jerárquica que facilitó su identificación y procesamiento en las etapas posteriores, como se ilustra en la figura 31.

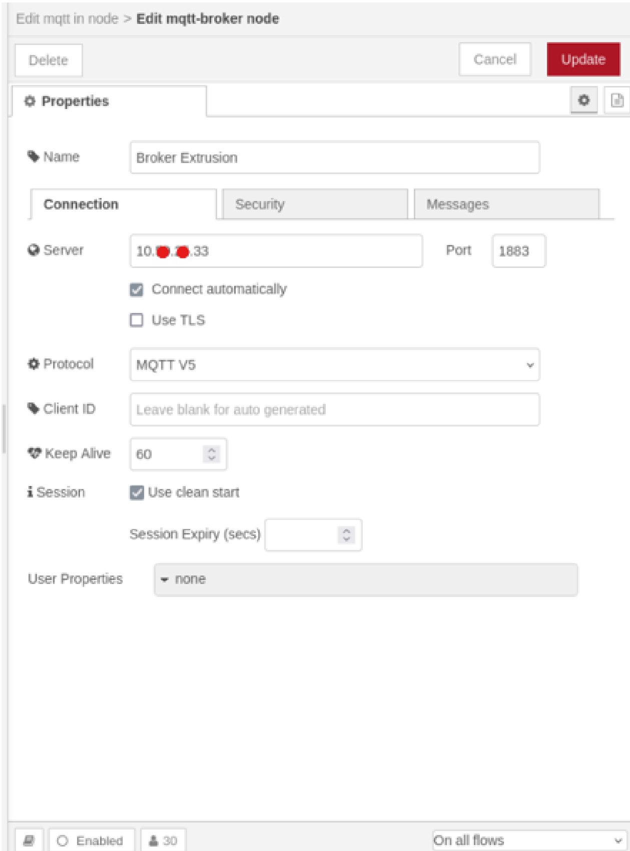


Ilustración 30: Configuración de conexión al Broker MQTT en Nodo MQTT Broker

Los tópicos MQTT que se utilizaron para el envío de los datos los demás clientes MQTT se listan a continuación:

Nombre de la variable	Tópico MQTT correspondiente
PESO VIRUTA A	Extrusion/XT42/VirutaA
PRESION MASA	Extrusion/XT42/presionMasa
AMP. MOTOR PRINCIPAL	Extrusion/XT42/AmpMotor
PESO VIRUTA B	Extrusion/XT42/VirutaB

EXT. ON	Extrusion/XT42/ExtOn
T. CALENTAMIENTO	Extrusion/XT42/Tcal
TEMP. MATERIAL	Extrusion/XT42/TempMaterial
T. TOT. PRODUCCION	Extrusion/XT42/TTprod
T. ACTUAL PRODUCCION	Extrusion/XT42/Tprod
SEÑAL CALENTAMIENTO	Extrusion/XT42/SignalCal
RPM MOTOR	Extrusion/XT42/rpmMP

Tabla 7: Tópicos MQTT para las Variables del Proceso

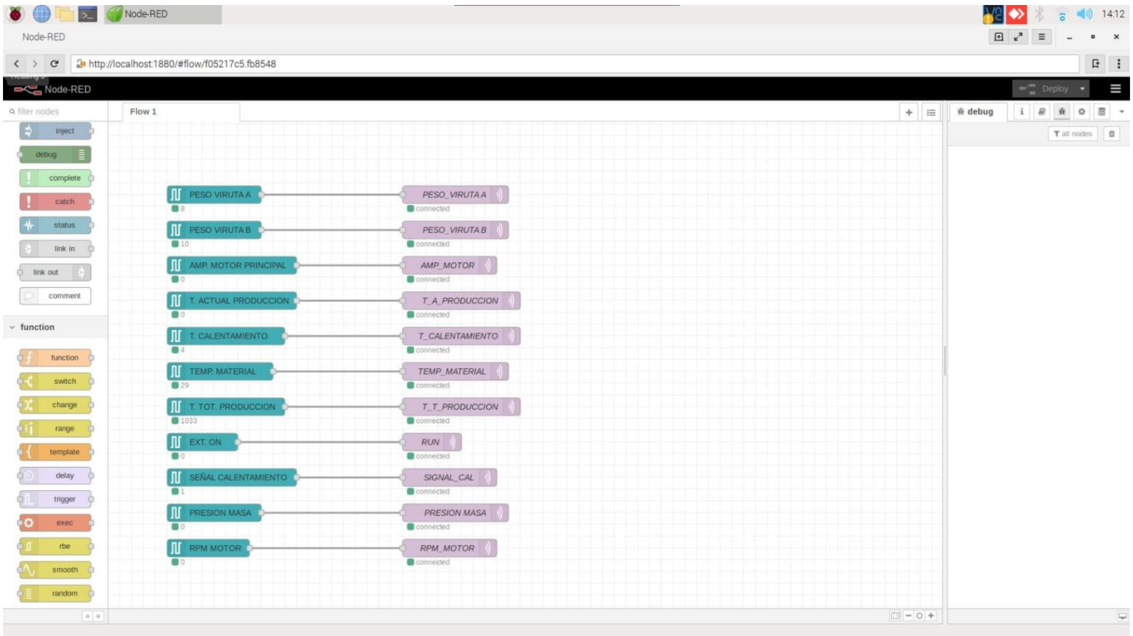


Ilustración 31: Diagrama de Flujo en NodeRed de Raspberry de Tablero Eléctrico

El uso de MQTT como protocolo de transmisión ofreció ventajas significativas en términos de rapidez, confiabilidad y flexibilidad, permitiendo que los datos fueran recibidos de manera eficiente por la Raspberry Pi 4 ubicada en la oficina de producción. Los resultados obtenidos durante las pruebas demostraron que la Raspberry Pi 3 cumplió de manera efectiva con su función de intermediario entre el PLC y el sistema central. La configuración del nodo S7 y la transmisión por MQTT garantizaron que las variables fueran capturadas y transmitidas en tiempo real, minimizando la latencia y maximizando la confiabilidad del sistema.

Además, la modularidad y escalabilidad del diseño facilitaron la integración de nuevas variables o sensores en caso de ser necesario, lo que posiciona al sistema como una solución robusta y preparada para futuras expansiones.

3.2.4 Configuración mediante Python del sensor QM30VT2

En la Raspberry Pi 3 se implementó un script en Python que corre en segundo plano para procesar los datos provenientes del sensor QM30VT2, encargado de medir vibración y temperatura en el motor. Este sensor está conectado a la Raspberry mediante un convertidor Modbus RS485 a USB, lo que permite establecer una comunicación serial. El script se diseñó para realizar lecturas periódicas de las variables ofrecidas por el sensor, transformando los datos brutos en valores legibles y útiles para el monitoreo de la línea extrusora.

Utilizando librerías específicas como pymodbus, el script configuró los parámetros necesarios para la comunicación, como la velocidad de transmisión, las direcciones del sensor y las frecuencias de lectura. En cada ciclo, el script obtenía los valores de la temperatura del motor, así como las vibraciones en los ejes X y Z, y realizaba las conversiones requeridas para expresar los datos en unidades comprensibles, siguiendo las especificaciones técnicas del sensor.

Una vez procesados, estos datos se publicaban mediante MQTT, utilizando tópicos predefinidos para cada variable, como se muestra en la figura 32. Esto permitió que la información del sensor fuera transmitida de manera eficiente a la Raspberry Pi 4, donde se integraron con las demás variables del sistema para su análisis y visualización.

La integración del script Python con el sensor QM30VT2 amplió las capacidades del sistema, proporcionando datos complementarios esenciales para el mantenimiento predictivo.

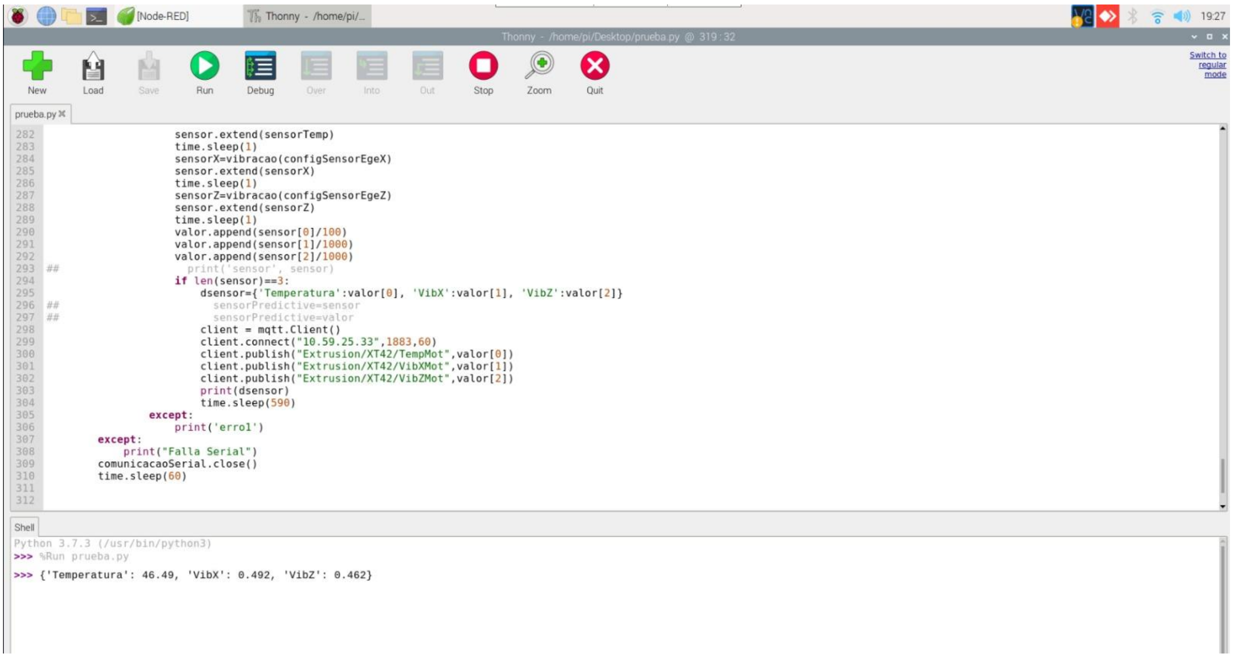


Ilustración 32: Resultados del Código Python para la adquisión de datos del sensor QM30VT2

3.2.5 Programación en Node-Red de Raspberry Pi 4 de Oficina de Producción

En la Raspberry Pi 4, se diseñó un flujo en Node-RED (Ilustración 33) para gestionar los datos recibidos mediante MQTT, provenientes de las diferentes fuentes del sistema. Este flujo no solo organiza los datos, sino que también realiza el procesamiento necesario para almacenarlos de manera estructurada en la base de datos SMART_EXT, utilizando las tablas SMART_MOTOR42 y PARAM_EXT42 según la naturaleza de las variables.

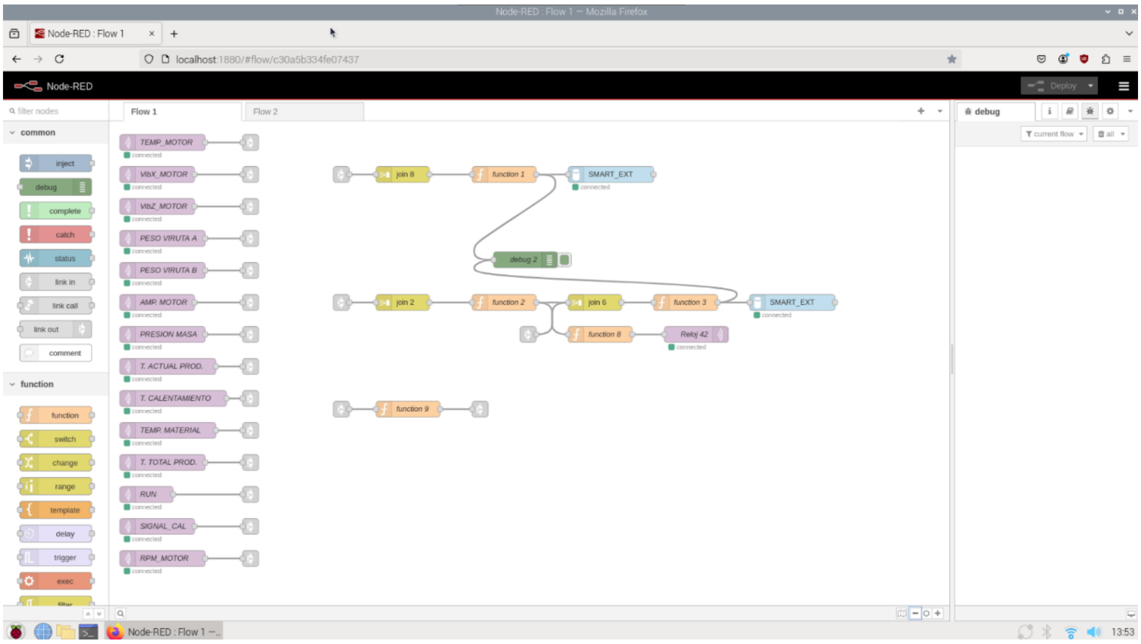


Ilustración 33: Diagrama de Flujo en NodeRed de Raspberry de Oficina de Producción

El flujo comienza con nodos MQTT In, que capturan las variables transmitidas desde la Raspberry Pi 3 y otros dispositivos del sistema. Estas variables son procesadas en diferentes nodos Function para clasificarlas y transformarlas antes de enviarlas a la base de datos. El procesamiento de las variables relacionadas con el motor se realiza en un nodo Function 1, donde se encapsulan datos como temperatura del motor, vibraciones en los ejes X y Z, amperaje del motor, tiempos de producción (actual y acumulado anual), RPM del motor y potencia eléctrica. La potencia eléctrica se calcula dinámicamente utilizando la fórmula:

$$Potencia\ =\ 220\,V\ \times\ Amperaje\ del\ Motor\ \times\ Factor\ de\ potencia$$

Estos datos se integran en un comando SQL INSERT (Ilustración 34), que es enviado a un nodo MySQL Server conectado al servidor MariaDB de la otra Raspberry Pi. Este procedimiento asegura un almacenamiento preciso en la tabla SMART_MOTOR42, diseñada para contener variables críticas utilizadas en el algoritmo de predicción del tiempo de vida útil del motor.

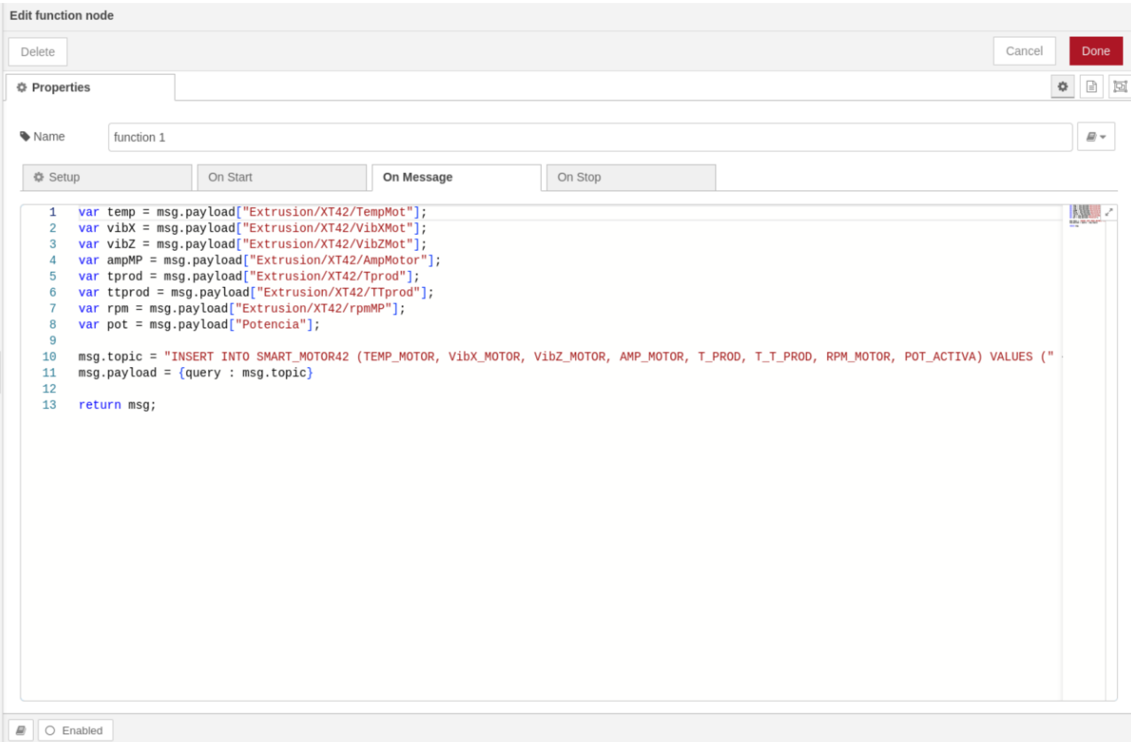


Ilustración 34: Código de la función para guardar los datos en Tabla SMART_MOTOR42

Por otro lado, las variables relacionadas con el proceso general se procesan en un nodo Function 3, que prepara los datos para ser almacenados en la tabla PARAM_EXT42. Estas variables incluyen los pesos de las virutas A y B, la presión de masa, la temperatura del material, el tiempo de calentamiento y el estado de la máquina. Este último se calcula previamente en un nodo Function 2, que combina las señales de marcha y calentamiento para determinar si la máquina está en producción, en calentamiento, o apagada. Dependiendo de la combinación de estas señales, se asigna un estado, como se muestra en la imagen 35:

1: Produciendo (marcha activa y calentamiento inactivo).
2: Calentando (calentamiento activo y marcha inactiva).
3: Apagada (ambas señales inactivas).

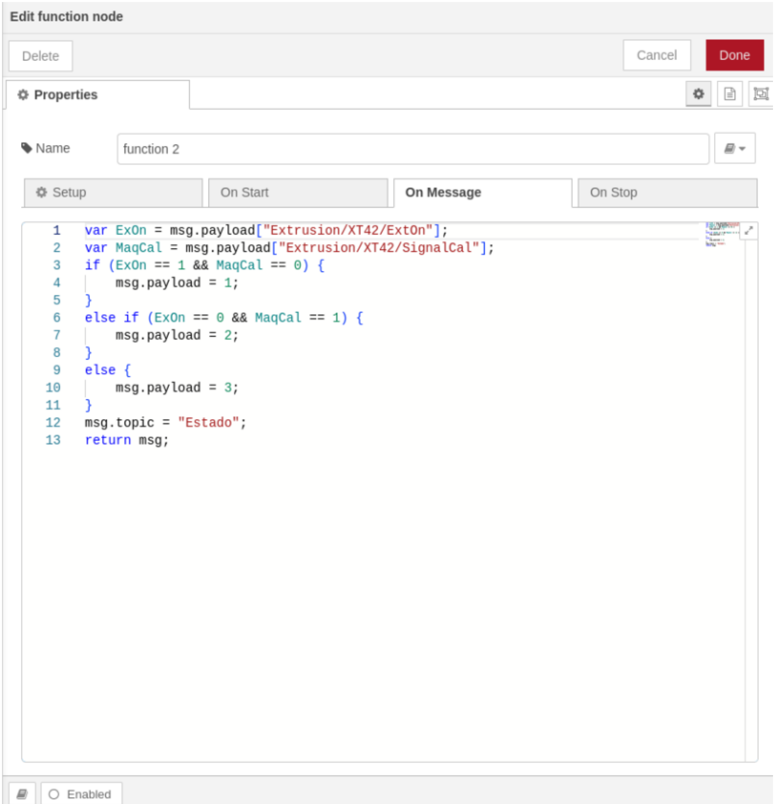


Ilustración 35: Código de la función para obtener el valor de Estado de la Máquina

El nodo Function 3 genera también un comando SQL INSERT para almacenar estos datos en la tabla correspondiente, utilizando un nodo MySQL Server para conectarse al servidor de la base de datos. Se muestra esto en las siguientes figuras:

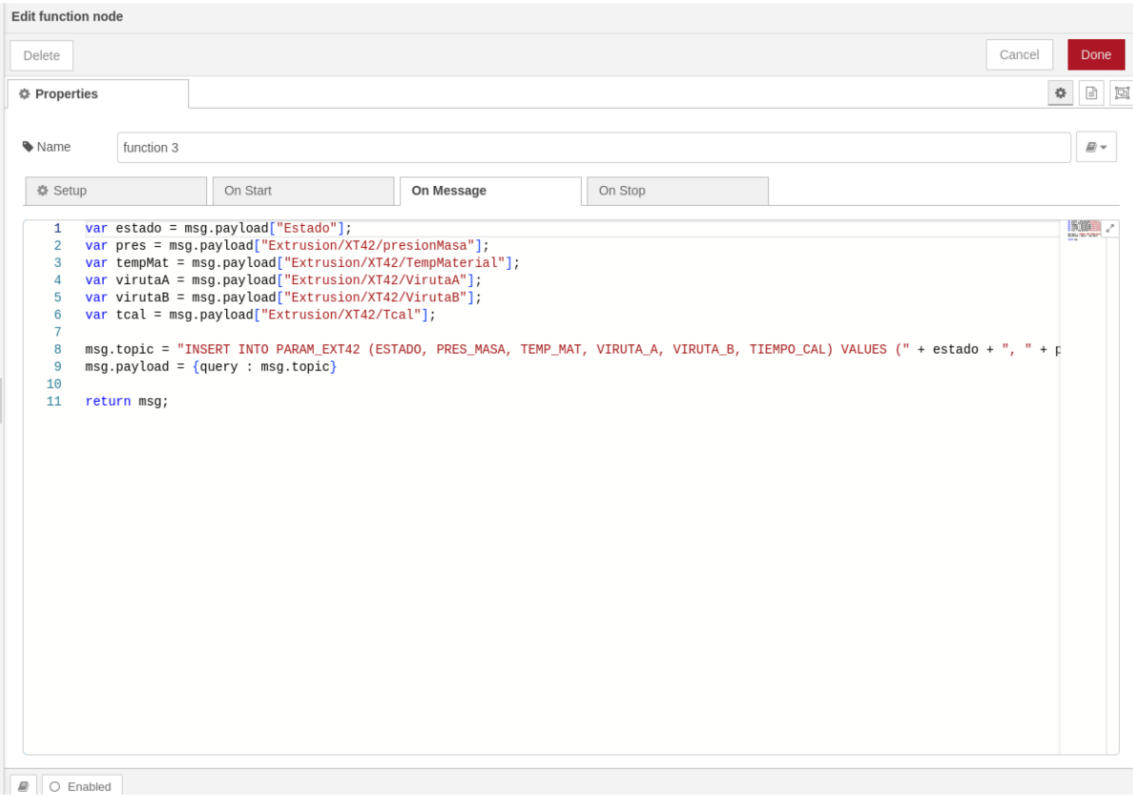


Ilustración 36: Código de la función para guardar los datos en Tabla PARAM_EXT42

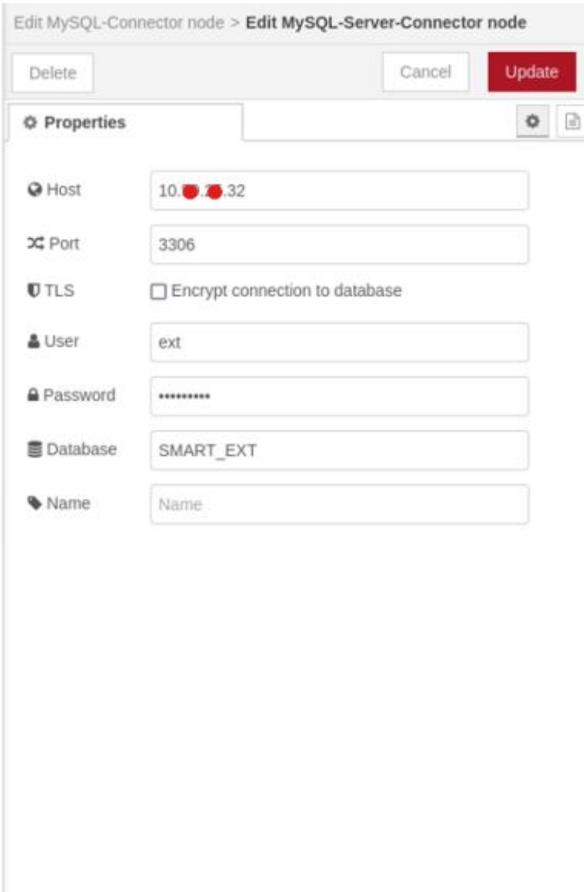


Ilustración 37: Configuración de conexión a Base de Datos en Nodo MySQL Server

Este flujo integral permitió no solo un manejo eficiente de las variables en tiempo real, sino también una clasificación clara y ordenada para su análisis posterior. La capacidad de Node-RED para realizar transformaciones dinámicas mediante nodos Function aseguró la flexibilidad del sistema, permitiendo adaptaciones rápidas ante cambios o necesidades futuras. La integración con MariaDB, a través de comandos SQL precisos, fortaleció la confiabilidad del almacenamiento de datos, garantizando la disponibilidad y consistencia de la información para el monitoreo y análisis de la línea extrusora.

3.2.6 Creación de tablas en Raspberry de Base de Datos

En la Raspberry Pi designada como servidor de base de datos, se instaló MariaDB para gestionar de manera eficiente la información recopilada del sistema de monitoreo. Durante el proceso, se configuró un usuario específico con una contraseña segura y se creó la base de datos denominada SMART_EXT, que almacena las variables provenientes de la línea extrusora.

Para habilitar el acceso remoto a la base de datos, se realizaron modificaciones en el archivo de configuración principal de MariaDB ubicado en /etc/mysql/mariadb.cnf. Específicamente, se ajustaron los parámetros relacionados con el enlace de direcciones IP, permitiendo que el servidor aceptara conexiones externas, como se observa en la ilustración 38. Este paso fue esencial para que otros dispositivos del sistema, como la Raspberry Pi 4, pudieran interactuar con la base de datos sin restricciones.

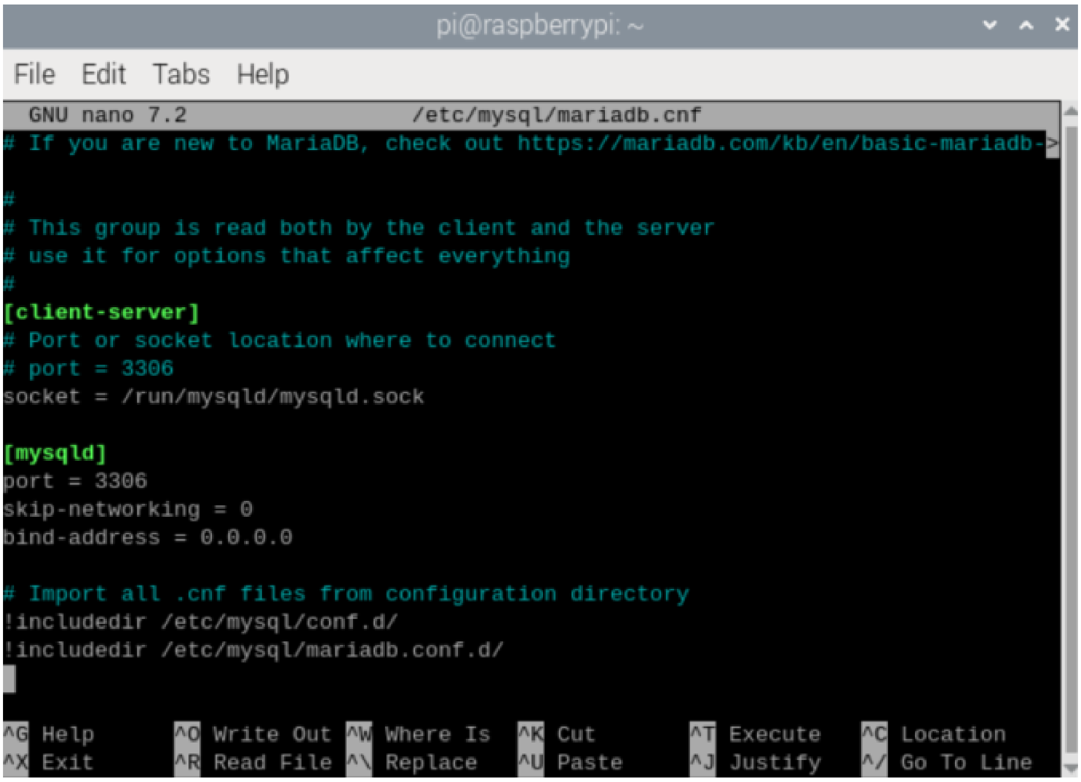


Ilustración 38: Modificaciones en el archivo de configuración de MariaDB en Raspberry de Base de Datos

Adicionalmente, se configuró el firewall UFW (Uncomplicated Firewall) para permitir el tráfico entrante en el puerto 3306, que es el puerto predeterminado utilizado por MariaDB. Este ajuste garantizó que las conexiones remotas pudieran establecerse de manera segura y sin interferencias. Por último, al usuario creado en MariaDB se le otorgaron todos los permisos necesarios sobre la base de datos SMART_EXT, asegurando que pudiera realizar operaciones como consultas, inserciones y modificaciones, fundamentales para el manejo dinámico de la información.

Finalmente, se estableció una conexión remota a la base de datos utilizando la herramienta DBeaver desde una PC portátil, detallándose en la figura 39. Esto permitió gestionar y estructurar las tablas de manera más sencilla e intuitiva, aprovechando la interfaz gráfica de DBeaver para agilizar la creación y modificación de estructuras de datos.

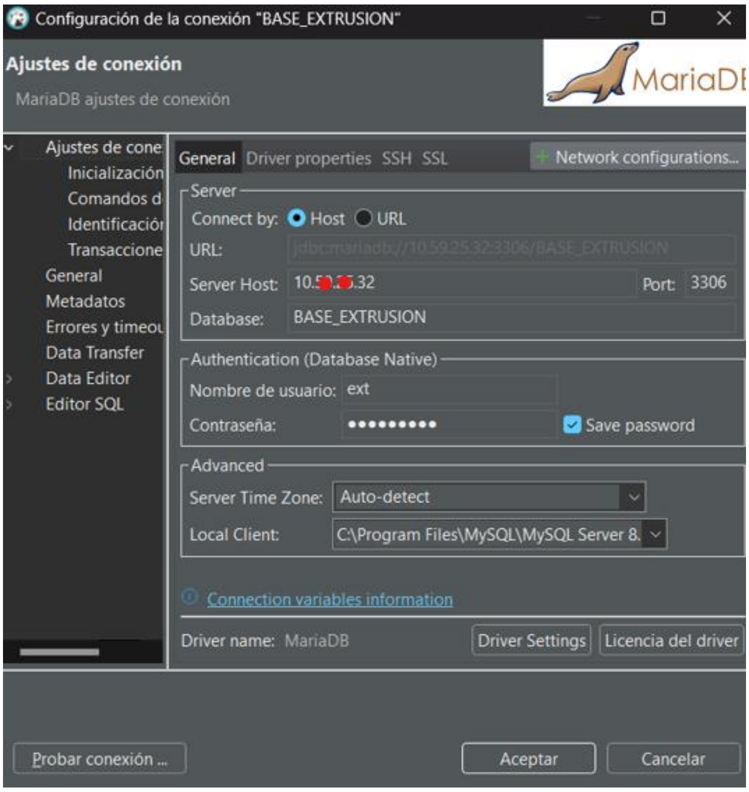


Ilustración 39: Configuración de la conexión a Base de Datos en DBeaver

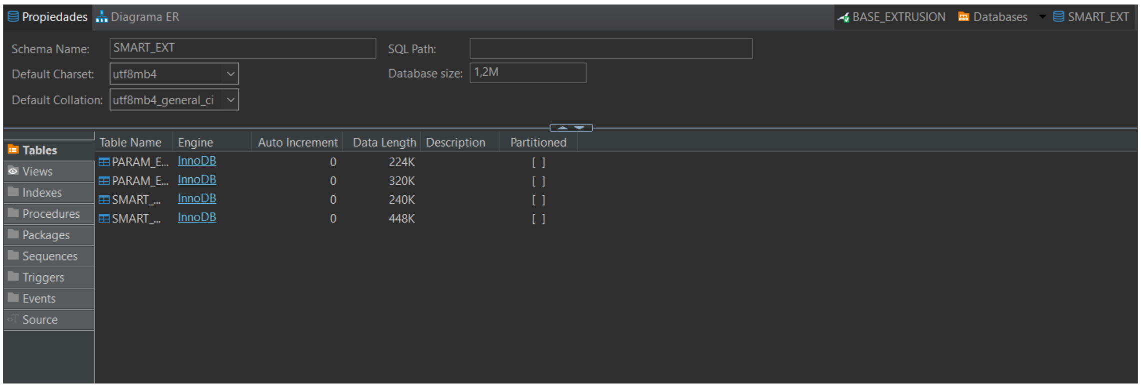


Ilustración 40: Tablas de la Base de Datos SMART_EXT

Como se corrobora en la imagen anterior, se crearon dos tablas principales: SMART_MOTOR42 y PARAM_EXT42, cada una diseñada para almacenar tipos específicos de datos relacionados con el monitoreo de la línea extrusora. La tabla SMART_MOTOR42 fue diseñada para registrar las variables críticas que alimentan el algoritmo de regresión destinado a estimar el tiempo de vida útil del motor (Ilustración 41). Estas variables incluyen mediciones clave como la temperatura del motor, vibración en los ejes X y Z, y cualquier otro parámetro relevante para el análisis predictivo.

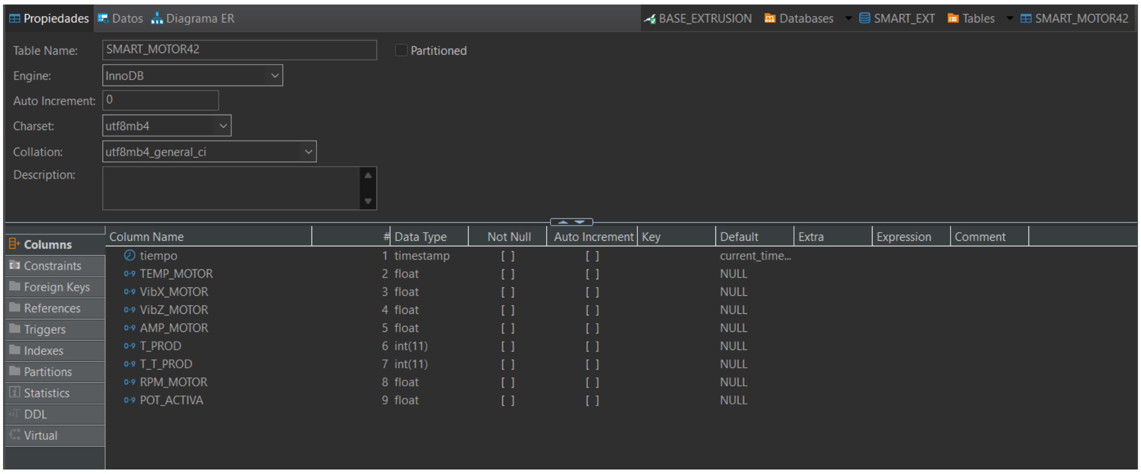


Ilustración 41: Columnas de la Tabla SMART_MOTOR42

Por otro lado, la tabla PARAM_EXT42 se estructuró para almacenar las variables restantes capturadas del sistema, como el estado de la máquina, tiempo de calentamiento, y las lecturas analógicas, temperatura del material, presión de masa, y datos sobre las virutas A y B (Ilustración 42).

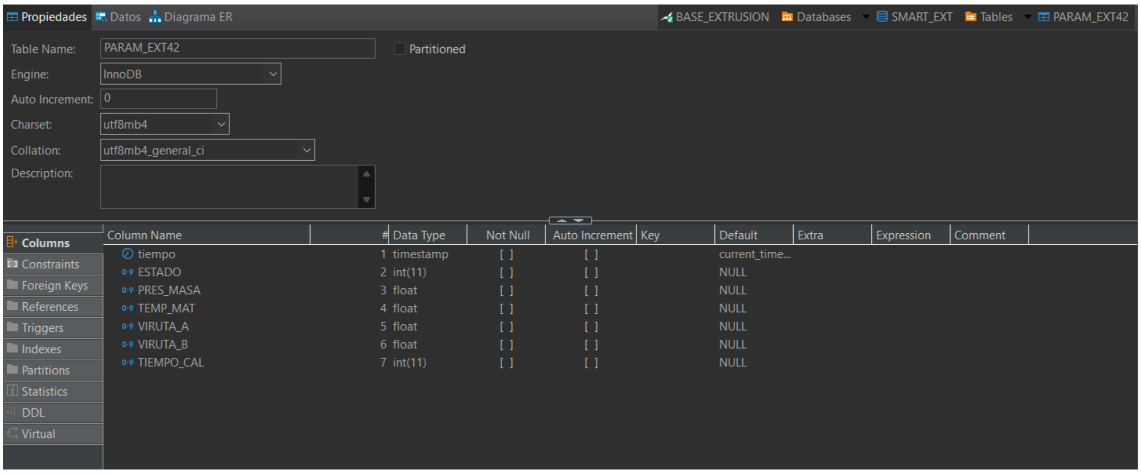


Ilustración 42: Columnas de la Tabla PARAM_EXT42

La conexión remota a través de DBeaver no solo facilitó la creación inicial de estas tablas y sus columnas correspondientes, sino que también permitió realizar ajustes y pruebas rápidas en la base de datos durante las etapas de desarrollo y prueba del sistema. Esta flexibilidad resultó especialmente útil para garantizar que las estructuras de datos fueran adecuadas para las necesidades del sistema y que pudieran ser fácilmente consultadas y actualizadas por las diferentes Raspberry Pi.

3.2.7 Programación de alarmas mediante Python en Raspberry Pi 3 de Oficina de Producción

En la Raspberry Pi 3 destinada al sistema de alertas auditivas, se desarrolló un script en Python que reproduce un archivo de audio cuando el tiempo de calentamiento de la línea extrusora supera las 4 horas. Este sistema, integrado mediante MQTT, permite recibir señales de activación desde la Raspberry Pi. El script utiliza la librería paho.mqtt.client para establecer una conexión con el broker MQTT configurado en la dirección IP 10.XX.XX.33 y el puerto 1883. Una vez conectado, el cliente se suscribe al tópico audio/Tcal/XT42/opex, donde se publica la señal que indica la activación de la alerta. Esto se muestra en la ilustración 43.

Cuando el script recibe un mensaje en el tópico, se ejecuta una función que verifica si el payload corresponde a "true" o "1". Si es así, activa la función de reproducción de audio, que utiliza el reproductor VLC configurado en modo silencioso para emitir el archivo de alerta. El archivo de audio está localizado en un directorio específico de la Raspberry Pi y se reproduce mediante la salida analógica del sistema, configurada para direccionarse a un dispositivo de audio conectado a un amplificador. Este amplificador aumenta la señal para que sea emitida a través de una bocina, asegurando que el personal de la planta pueda escuchar la alerta incluso en entornos ruidosos.

El script opera en segundo plano, manteniéndose en un bucle continuo que escucha nuevas señales en tiempo real. Durante las pruebas, el sistema respondió de manera eficiente, activando la alerta inmediatamente al recibir el mensaje de activación desde el sistema central. Además, la flexibilidad del diseño permite modificar fácilmente los parámetros del broker MQTT, los tópicos de suscripción o incluso el archivo de audio, adaptándose a posibles necesidades futuras.

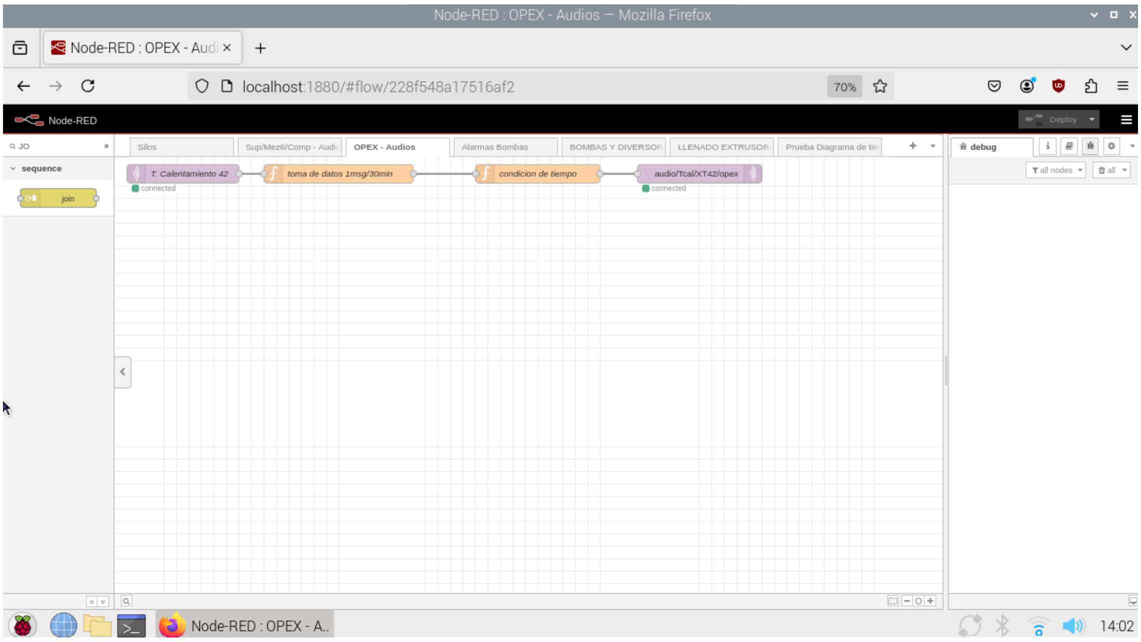


Ilustración 43: Diagrama de Flujo para activación de Alarma Sonora

Este sistema de alertas auditivas complementa de manera efectiva el monitoreo de la línea extrusora, añadiendo una capa adicional de seguridad y proactividad. Al alertar al personal sobre tiempos excesivos de calentamiento, previene posibles daños al equipo y a la materia prima, contribuyendo a optimizar la operación, y asegurando un manejo más eficiente y sostenible del proceso productivo.

3.2.8 Configuración de Algoritmo de Regresión

En el dispositivo terminal (laptop), se procedió a realizar la programación de la red neuronal, ante de comenzar a desglosar todo el proceso de programación y entrenamiento, es importante destacar los cambios que sufrió la base de información antes de realizar cualquier proceso de aprendizaje.

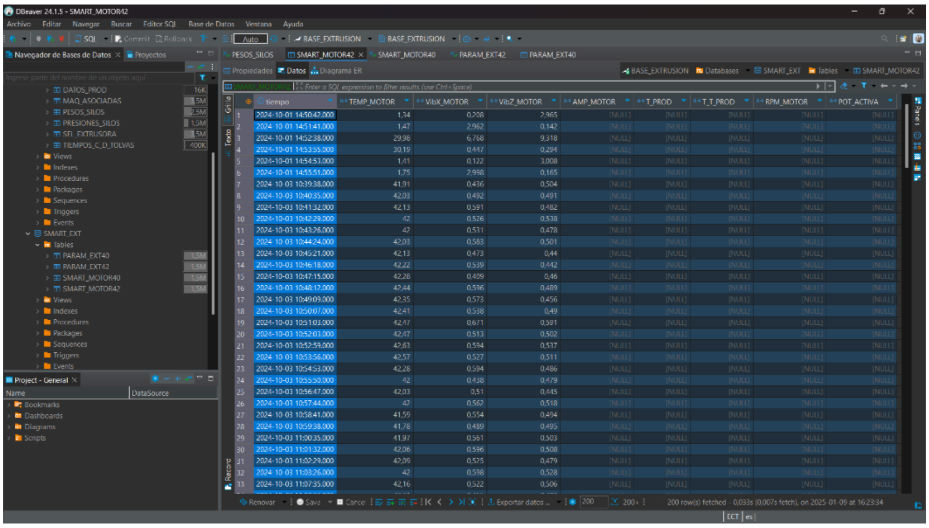


Ilustración 44: Data de información desde Dbeaver.

La Ilustración 45 detalla los primero 33 datos registrados por la base de datos puesta en funcionamiento, dentro de esta base se pueden observar los mismos caracteres de columnas de que definieron en el ejemplo del capítulo 2 de la sección de entrenamiento del algoritmo de regresión.

FECHA-HORA	TEMPERATURA _MOTOR [°c]	VIBRACION_X_MOTOR [mm/s]	VIBRACION_X_MOTOR [mm/s]	AMPERAJE_MOTOR [Amps]	TIEMPO_PRODRODUCCION [horas]	TIEMPO_TOTAL_PRODUCCION	RPM_MOTOR	Status_temp	status_vib_x	status_vib_z	tiempo de vida restante	Status_General
2024-10-22 15:00:45.000	44.96	0.491	0.491	55	0	0	12	Operativo	Satisfactorio	Satisfactorio	30000	estable
2024-10-22 15:18:00.000	45.19	0.583	0.583	55	0	0	12	Sobrecalentamiento	Satisfactorio	Satisfactorio	30000	irregular
2024-10-22 15:27:57.000	45.32	0.524	0.524	55	0	0	12	Sobrecalentamiento	Satisfactorio	Satisfactorio	30000	irregular
2024-10-22 15:37:54.000	45.35	0.568	0.568	55	0	0	12	Sobrecalentamiento	Satisfactorio	Satisfactorio	30000	irregular
2024-10-22 15:47:51.000	45.19	0.645	0.645	55	0	0	12	Sobrecalentamiento	Satisfactorio	Satisfactorio	30000	irregular
2024-10-22 15:57:48.000	44.92	28.697	28.771	55	1	1	12	Operativo	Insatisfactorio	Insatisfactorio	29999	irregular
2024-10-22 16:07:46.000	45.05	0.537	0.537	55	1	1	12	Sobrecalentamiento	Satisfactorio	Satisfactorio	29999	irregular
2024-10-22 16:17:43.000	45.29	0.644	0.644	55	1	1	12	Sobrecalentamiento	Satisfactorio	Satisfactorio	29999	irregular
2024-10-22 16:27:40.000	45.09	0.547	0.547	55	1	1	12	Sobrecalentamiento	Satisfactorio	Satisfactorio	29999	irregular
2024-10-22 16:37:37.000	44.76	0.551	0.551	55	1	1	12	Operativo	Satisfactorio	Satisfactorio	29999	estable
2024-10-22 16:47:34.000	44.43	0.475	0.475	55	2	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
2024-10-22 16:57:31.000	44.53	0.509	0.509	55	2	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
2024-10-22 17:07:30.000	44.33	0.444	0.444	55	2	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
2024-10-22 17:17:27.000	43.98	0.446	0.446	53	2	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
2024-10-22 17:27:24.000	43.62	0.49	0.49	55	2	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
2024-10-22 17:37:21.000	52.12	0.454	0.454	55	2	2	12	Sobrecalentamiento	Satisfactorio	Satisfactorio	29998	irregular
2024-10-22 17:47:18.000	42.54	0.385	0.385	55	2	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
2024-10-22 17:57:15.000	42.28	0.349	0.349	55	3	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable
2024-10-22 18:07:12.000	41.75	0.513	0.513	56	3	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable
2024-10-22 18:17:09.000	41.84	0.414	0.414	56	3	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable
2024-10-22 18:27:07.000	41.78	0.522	0.522	57	3	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable
2024-10-22 18:37:04.000	41.59	0.463	0.463	55	3	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable

Ilustración 45: Data de información obtenida.

Por medio de la ilustración 46 se puede observar que la tabla ha sufrido ciertas modificaciones en los títulos de cada una de sus respectivas columnas, este se lo realizo con el fin de que la red pueda identificar de mejor manera cada valor. Por otro lado, se han generado nuevas características del sistema donde se puede apreciar que los nuevos datos hacen referencia a estado que se encuentra el motor, dando nacimiento a las columnas: "status_temp", "status_vib_x", "status_vib_z", "tiempo de vida restante" y por último "status_general".

Al mismo tiempo que se agregaron estas nuevas columnas también si opto por eliminar 3 de las columnas de datos proporcionados por la base principal, estas columnas fueron eliminadas ya que no tenían relevancia al momento del entrenamiento de la red y se podía obviar este tipo de información, quedando de la siguiente manera:

TEMPERATURA _MOTOR [°c]	VIBRACION_X_MOTOR [mm/s]	VIBRACION_X_MOTOR [mm/s]	AMPERAJE_MOTOR [Amps]	TIEMPO_TOTAL_PRODUCCION	RPM_MOTOR	Status_temp	status_vib_x	status_vib_z	tiempo de vida restante	Status_General
44.96	0.491	0.491	55	0	12	Operativo	Satisfactorio	Satisfactorio	30000	estable
45.19	0.583	0.583	55	0	12	Sobrecalenta	Satisfactorio	Satisfactorio	30000	irregular
45.32	0.524	0.524	55	0	12	Sobrecalenta	Satisfactorio	Satisfactorio	30000	irregular
45.35	0.568	0.568	55	0	12	Sobrecalenta	Satisfactorio	Satisfactorio	30000	irregular
45.19	0.645	0.645	55	0	12	Sobrecalenta	Satisfactorio	Satisfactorio	30000	irregular
44.92	28.697	28.771	55	1	12	Operativo	Insatisfactori	Insatisfactori	29999	irregular
45.05	0.537	0.537	55	1	12	Sobrecalenta	Satisfactorio	Satisfactorio	29999	irregular
45.29	0.644	0.644	55	1	12	Sobrecalenta	Satisfactorio	Satisfactorio	29999	irregular
45.09	0.547	0.547	55	1	12	Sobrecalenta	Satisfactorio	Satisfactorio	29999	irregular
44.76	0.551	0.551	55	1	12	Operativo	Satisfactorio	Satisfactorio	29999	estable
44.43	0.475	0.475	55	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
44.53	0.509	0.509	55	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
44.33	0.444	0.444	55	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
43.98	0.446	0.446	53	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
43.62	0.49	0.49	55	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
52.12	0.454	0.454	55	2	12	Sobrecalenta	Satisfactorio	Satisfactorio	29998	irregular
42.54	0.385	0.385	55	2	12	Operativo	Satisfactorio	Satisfactorio	29998	estable
42.28	0.349	0.349	55	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable
41.75	0.513	0.513	56	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable
41.84	0.414	0.414	56	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable
41.78	0.522	0.522	57	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable
41.59	0.463	0.463	55	3	12	Operativo	Satisfactorio	Satisfactorio	29997	estable

Ilustración 46: Ajuste de data a valores objetivo de entrenamiento

3.2.8.1 Tipo de red neuronal implementada.

Con la base de datos correctamente depurada y lista para el entrenamiento, no se puede dejar de lado las características que posee esta base y como está van a afectar a la red, definiendo así el tipo de red a utilizar. En un inicio, la idea de implementar una red de regresión se observaba como la opción más eficaz de la red; a medida que la base tomo forma y se analizó profundamente las necesidades y los resultados esperados; se optó por el diseño de una red neuronal multicapa o también llamada (MLP – Multilayer Perceptron). Debido a que este tipo de red cumple con las características que requiere la red.

3.2.8.2 *Entradas y valores objetivos.*

Para esta etapa de la implementación se es necesario identificar cuáles son las entradas y salidas requeridas o también llamadas valores objetivos, como su nombre lo indica son los valores que se desea llegar al finalizar el entrenamiento; con la base lista y modificada, este modelo de red neuronal tomara como entradas las primeras 9 columnas de la base y como valores objetivos a las dos últimas columnas.

Para el desarrollo y entrenamiento de la red, es de vital importancia que la base de entrenamiento contenga los resultados a los que se quiere llegar al terminal el entrenamiento ya que de esta forma es como se podrá analizar los resultados de brinde la red con los resultados reales de verían ser.

3.2.8.3 *Preparación de la data de entrenamiento*

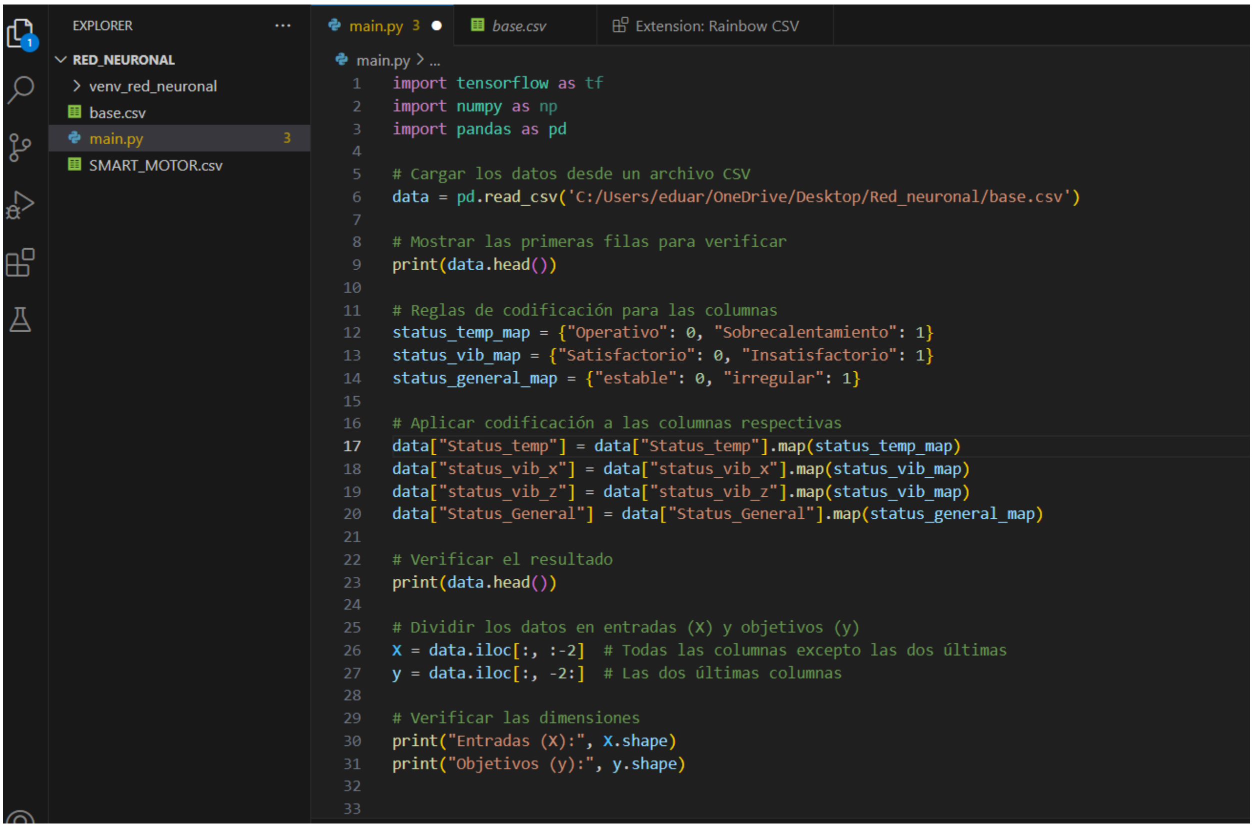


Ilustración 47: Ajuste de data real Pre-entrenamiento

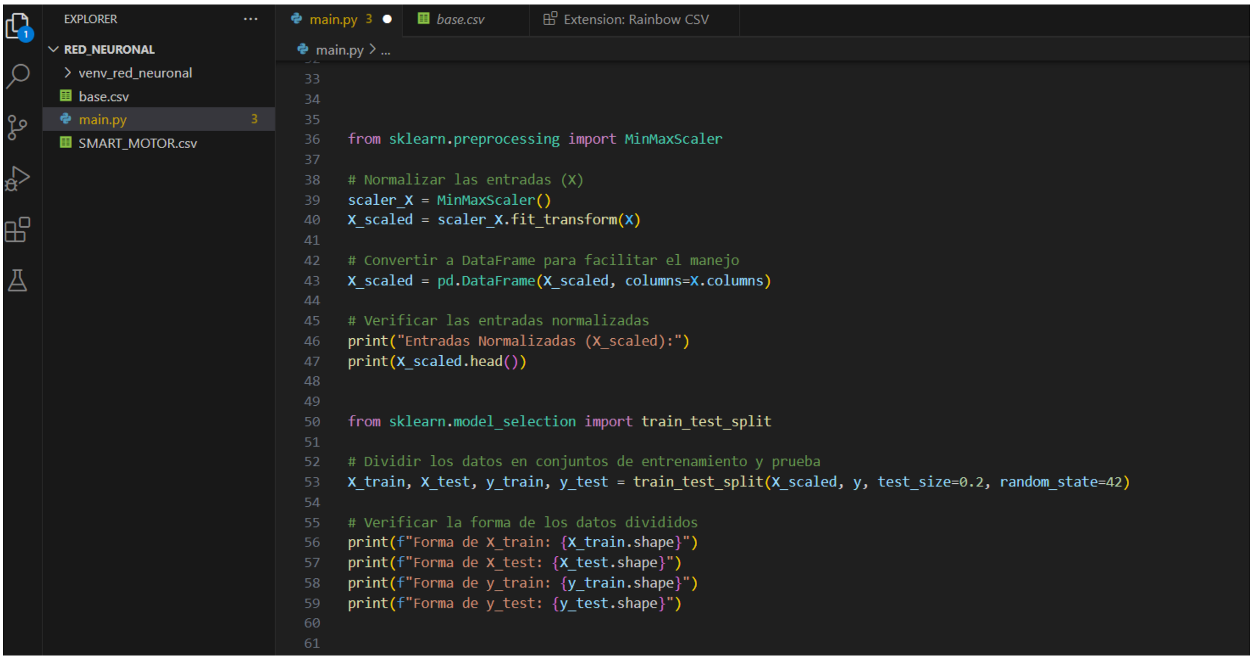
La imagen que se presenta previo a este texto detalla la preparación de la data en donde se puede observar que lo primero que se realizo es la importación de las librerías necesarias para esta preparación inicial, posteriormente en la línea de código 6 se observa que se procede a cargar el documento denominado “base.cvs” que contiene toda la data de la línea de extrusión.

Como se pudo notar, en la tabla final depurada por el usuario se detallaron algunas nuevas columnas donde la mayoría es de tipo alfabético y no numérico, por lo que representa un pequeño obstáculo para la red analizar este tipo de datos, por ende, en de la línea 12 a 14 del código previo lo que se realiza en un “Label Encoding”, donde lo que se procura es asignarle un valor numérico a cada una de las opciones alfabéticas posibles.

Para el caso específico de este proyecto, se cuenta con 3 pares de opciones de este tipo; em primer lugar se tiene a la columna “status_temp” la cual oscila entre el estado operativo y sobrecara las cuales serán remplazadas por el número 0 en el caso de ser operativo y el 1 en caso de ser sobrecarga.

Las columnas de “status_vib_x” y “status_vib_z” comparte características por lo tanto para usando la misma analogía, cuando el estatus sea satisfactorio el valor asignado será 0 y si el status llega ser satisfactorio el valor asignado será 0; y para finalizar la columna objetivo “status general”, donde para el estado “estables” será 0 y para el irregular será un 1. Y se clasificaran entre entradas y valores objetivos.

3.2.8.4 *Normalización y división de la data*



```python
from sklearn.preprocessing import MinMaxScaler

# Normalizar las entradas (X)
scaler_X = MinMaxScaler()
X_scaled = scaler_X.fit_transform(X)

# Convertir a DataFrame para facilitar el manejo
X_scaled = pd.DataFrame(X_scaled, columns=X.columns)

# Verificar las entradas normalizadas
print("Entradas Normalizadas (X_scaled):")
print(X_scaled.head())


from sklearn.model_selection import train_test_split

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Verificar la forma de los datos divididos
print(f"Forma de X_train: {X_train.shape}")
print(f"Forma de X_test: {X_test.shape}")
print(f"Forma de y_train: {y_train.shape}")
print(f"Forma de y_test: {y_test.shape}")
```

Ilustración 48: División de data de entrenamiento

La normalización es el proceso de llevar a toda la data de entrenamiento a valor o rangos establecidos en este caso de implementación se optó por una normalización entre 0 y 1, es los valores que originalmente tenía serán cambiados por calores entre 0 y 1 según se considere, tal y como el código de la imagen 48 realiza.

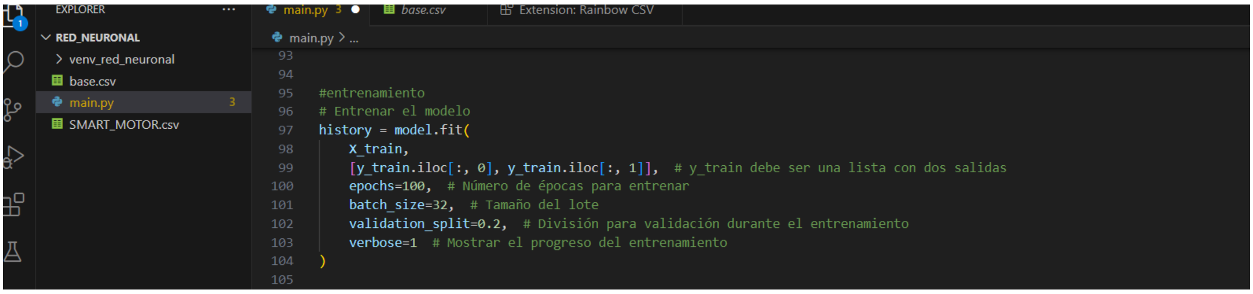
Otro punto importante dentro del entrenamiento es saber cómo gestionar la data, por esto se decidió usar el 80% de la data para el entrenamiento y el 20% restante para la evaluación de modelo y comparación con los datos reales.

3.2.8.5 Definición de entrada – salidas – características del modelo.


```python
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dense

# Definir la capa de entrada
input_layer = Input(shape=(X_train.shape[1],))

# Crear las capas ocultas
hidden_layer1 = Dense(64, activation='relu')(input_layer)
hidden_layer2 = Dense(32, activation='relu')(hidden_layer1)

# Salida para la predicción de "tiempo de vida restante" (regresión)
output_regression = Dense(1, name="regression_output")(hidden_layer2)

# Salida para la predicción de "Status_General" (clasificación)
output_classification = Dense(1, activation='sigmoid', name="classification_output")(hidden_layer2)

# Crear el modelo con dos salidas
model = Model(inputs=input_layer, outputs=[output_regression, output_classification])

# Compilar el modelo
model.compile(
    optimizer='adam',
    loss={'regression_output': 'mse', 'classification_output': 'binary_crossentropy'},
    metrics={'regression_output': 'mae', 'classification_output': 'accuracy'}
)

# Resumen del modelo
model.summary()
```

Ilustración 49: Características del Modelo de Red Neuronal

3.2.8.6 Entrenamiento y guardado de modelo de red neuronal




```python
#entrenamiento
# Entrenar el modelo
history = model.fit(
    X_train,
    [y_train.iloc[:, 0], y_train.iloc[:, 1]],  # y_train debe ser una lista con dos salidas
    epochs=100,  # Número de épocas para entrenar
    batch_size=32,  # Tamaño del lote
    validation_split=0.2,  # División para validación durante el entrenamiento
    verbose=1  # Mostrar el progreso del entrenamiento
)
```

Ilustración 50: Características Principales del Entrenamiento

La ilustración 50 muestra el modelo de red utilizado en este proyecto, y el desglose de cada una de sus características, para este modelo en particular, se puede notar en la ilustración de antecede esta información que este modelo se entrena por 100 epocas a un tamaño de entrenamiento por lote de 32 datos, es decir cada dato o conjunto de entrenamiento se lo hará de forma progresiva en conjunto de 32 datos.

Ilustración 51: Modelo de Red Neuronal con Keras.

Para esta instancia, la ilustración 51 da a conocer información de cómo se guarda el modelo de red entrenado y descrito en secciones anteriores, con lo cual se observa que se utiliza a Keras como principal gestor del modelo de red, esto debido a su confiabilidad y avances desarrollados específicamente mente para el entrenamiento de datos, esta subdivisión de Tensor Flow permite autoajustares a la red y corregir detalles al modelo entrenado y guardado.

3.2.9 Diseño de dashboard en Grafana de Raspberry Pi 4 de Oficina de Producción

En la Raspberry Pi 4, además de gestionar el flujo de datos en Node-RED para recibir las variables enviadas mediante MQTT y almacenarlas en la base de datos MariaDB, se configuró Grafana como herramienta de visualización para el dashboard del sistema de monitoreo. Grafana se conectó directamente a la base de datos SMART_EXT, permitiendo representar las variables de la línea extrusora de manera clara y en tiempo real.

El diseño del dashboard se centró en presentar la información de forma intuitiva y funcional. Los datos numéricos, como el tiempo de producción acumulado, el tiempo de calentamiento y las lecturas de las virutas A y B, se visualizaron mediante indicadores tipo gauge, como se ilustra en la figura 52. Estos gráficos permiten al usuario obtener una visión rápida de los valores actuales, destacando si están dentro de los rangos aceptables o si requieren atención inmediata.

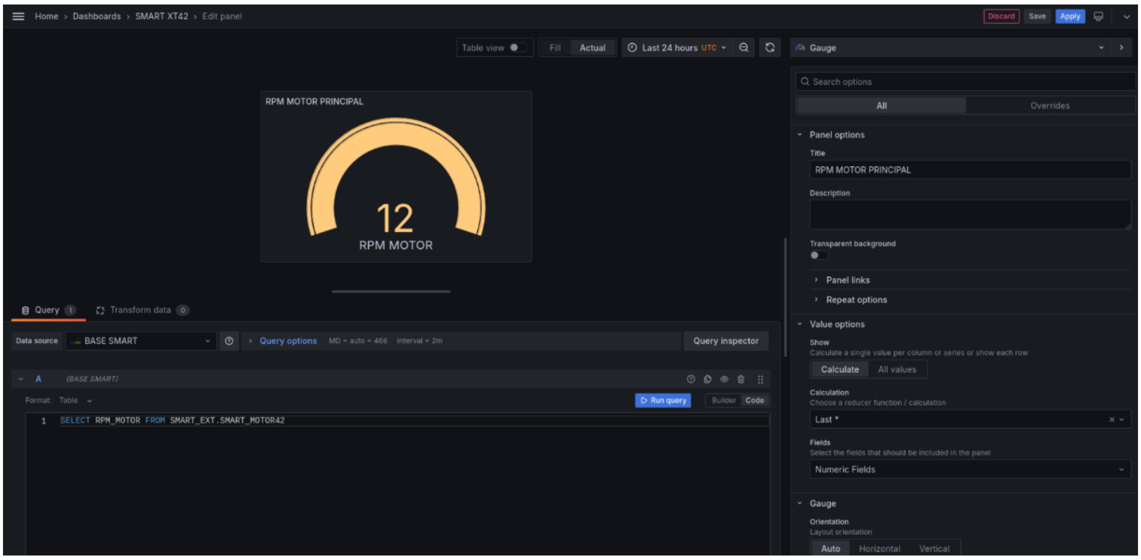


Ilustración 52: Configuración de un Gauge en Grafana

Por otro lado, las variables más críticas, como la temperatura del motor, el amperaje del motor y la temperatura del material, se mostraron utilizando gráficos de tipo time series, (Ilustración 53). Estos gráficos representan la evolución de cada variable a lo largo del tiempo, proporcionando a los operadores una herramienta para identificar tendencias, detectar posibles anomalías y analizar el comportamiento del sistema en diferentes períodos.

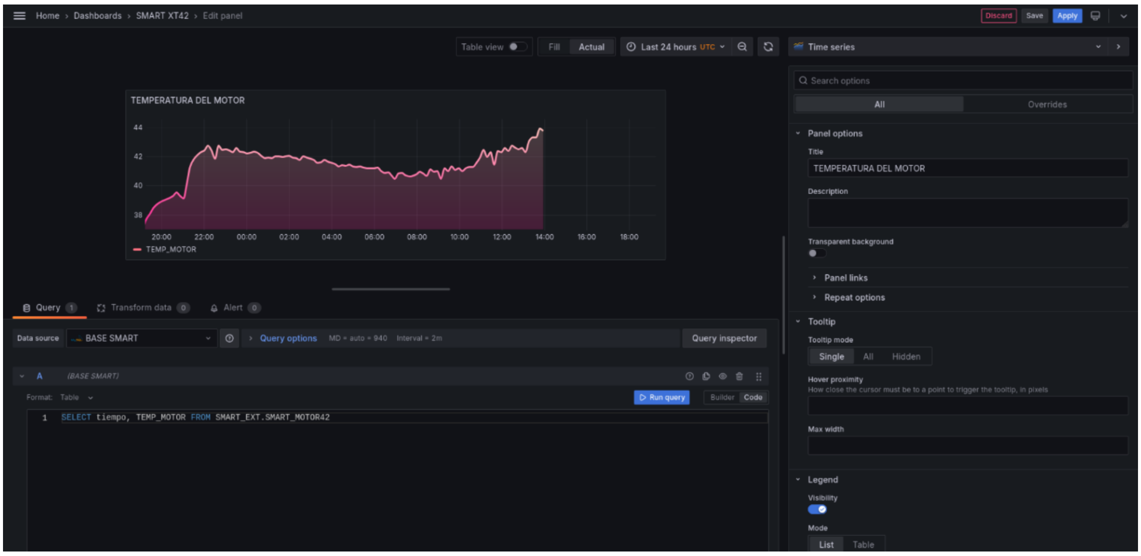


Ilustración 53: Configuración de un Time series en Grafana

La configuración de Grafana permitió personalizar alertas visuales en los gráficos, indicando valores fuera de los límites predefinidos mediante colores o marcadores específicos. Esto aseguró que los operadores pudieran identificar rápidamente cualquier desviación en las variables importantes, facilitando una respuesta proactiva a condiciones adversas.

Este dashboard integrado en Grafana proporcionó una plataforma potente y flexible para el monitoreo continuo de la línea extrusora. Su capacidad para representar tanto datos numéricos instantáneos como series temporales críticas mejoró significativamente la supervisión del sistema, permitiendo una toma de decisiones informada y en tiempo real. La elección de Grafana, combinada con la infraestructura de Node-RED y MariaDB, consolidó un sistema de monitoreo robusto, accesible y escalable.

3.3 Análisis de Resultados

3.3.1 Dashboard completo en Grafana

El dashboard implementado en Grafana proporciona una visualización integral de las variables clave relacionadas con la operación de la línea extrusora. La interfaz está organizada de forma clara y eficiente, permitiendo una comprensión rápida del estado actual y el desempeño del sistema.

Ilustración 54: Dashboard de Variables del Proceso en Grafana

Ilustración 55: Dashboard de Variables del Proceso en Grafana

Como se muestran en las anteriores imágenes, en la parte superior izquierda se destaca el indicador del estado de la máquina, que muestra de forma gráfica si está en producción, calentamiento o apagada. En la ilustración 54, se observa que la máquina está en modo de "Calentando", lo que refleja la etapa de ese momento del proceso.

Los gauges ofrecen una representación numérica de datos esenciales como el tiempo de producción actual, tiempo total de producción, RPM del motor principal, presión de masa y tiempo de calentamiento actual. Es importante resaltar que el tiempo total de producción acumula 1319 horas, un dato relevante para evaluar la durabilidad y el uso continuo del equipo. Además, el tiempo de calentamiento está en 4 horas, alcanzando un umbral crítico que activa alarmas de notificación. Cabe recalcar que el tiempo acumulable de producción total (anual), aparte de reiniciarse cada año, se encera cuando se realizan cargas de programa al PLC LOGO (ya sea para realizar modificaciones, agregar señales, etc). Es por esto por lo que en la imagen 55 se observa que el tiempo se ha reiniciado.

En la parte derecha del dashboard, los gráficos de series temporales presentan datos históricos relacionados con las variables más importantes: temperatura del motor, amperaje del motor y temperatura del material. Estos gráficos permiten analizar tendencias, identificar anomalías y correlacionar variables. Por ejemplo, para la figura 54, la temperatura del motor muestra una tendencia descendente, lo cual puede indicar un enfriamiento progresivo tras un periodo de inactividad. De manera similar, el amperaje del motor es cero en ese periodo de tiempo, sugiriendo que la máquina no está produciendo, mientras que la temperatura del material se mantiene casi invariable.

3.3.2 Llenado de las tablas en la Base de Datos

La base de datos SMART_EXT, estructurada para almacenar los datos relevantes de la línea extrusora, ha registrado valores precisos y continuos en sus tablas SMART_MOTOR42 y PARAM_EXT42, reflejando el correcto funcionamiento del sistema de adquisición y almacenamiento.

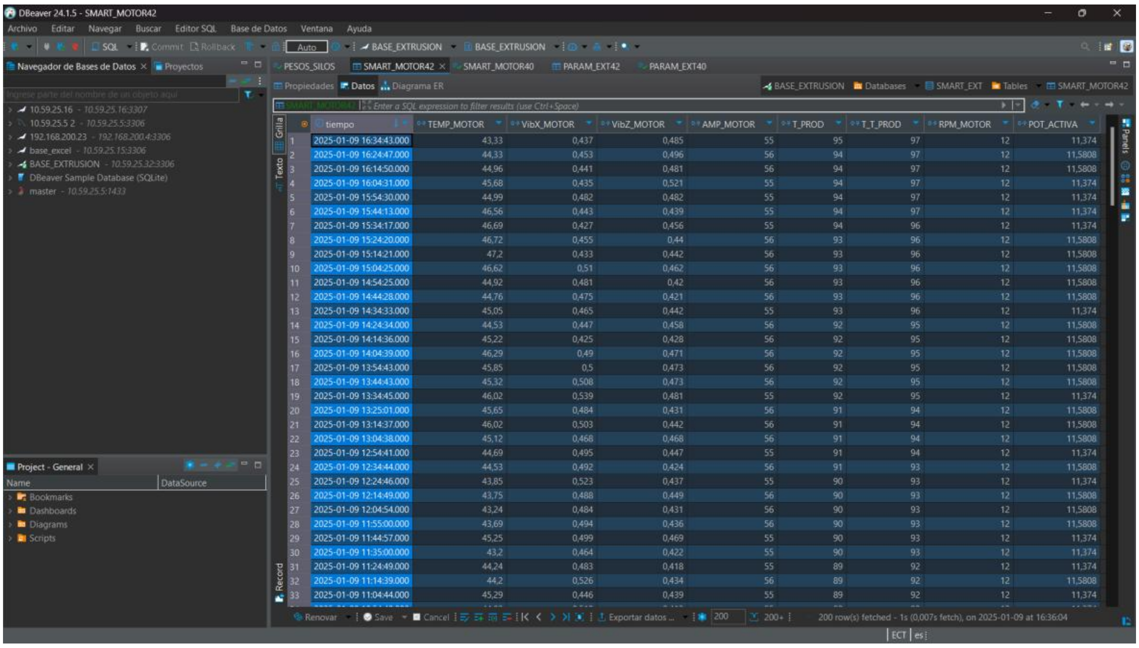
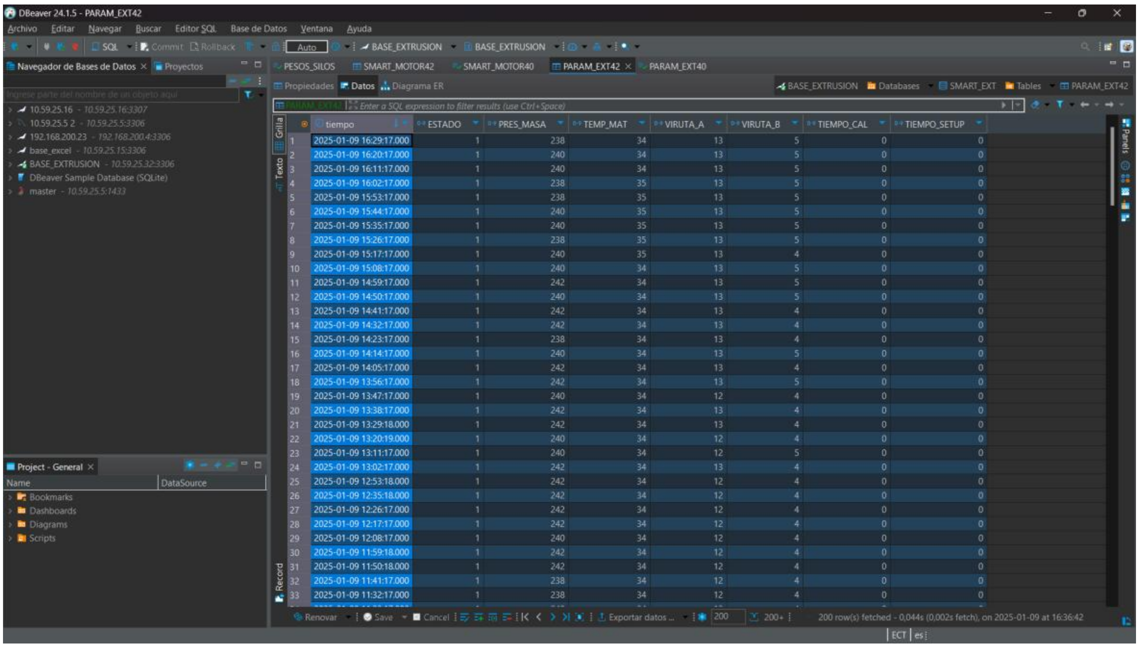


Ilustración 56: Tabla SMART_MOTOR42 con datos recientes

En la tabla de la figura 56 se almacenan parámetros clave relacionados con el motor principal de la línea extrusora, como temperatura del motor, vibración en los ejes X y Z, amperaje, tiempo actual y total de producción, RPM del motor y potencia. Estos datos permiten monitorear en tiempo real el estado del motor y realizar análisis predictivos.

La temperatura del motor muestra valores en promedio de 45.63 °C, lo que indica un comportamiento térmico estable en las condiciones actuales de operación. Las vibraciones en los ejes X y Z tienen valores bajos, con un rango promedio de 0.4 a 0.55 mm/s, lo que sugiere que el motor opera en condiciones normales, sin indicios de desequilibrios significativos. El amperaje del motor oscila entre 55 y 56 A, manteniendo una estabilidad que confirma que el motor no está sometido a cargas anormales. El tiempo total de producción registra un acumulado de 97 horas, mientras que el tiempo actual de producción es 95 horas, lo cual refleja que a inicios del año 2025 el tiempo de producción anual se ha reiniciado, confirmando su correcto funcionamiento. La potencia del motor se mantiene en 11,3 kW, calculada con base en el voltaje constante de 220 V, el amperaje registrado y el factor de potencia estimado.

Estos datos confirman que el sistema es capaz de capturar y almacenar información crucial para un análisis eficiente del estado operativo del motor.



#	tiempo	ESTADO	PRES_MASA	TEMP_MAT	VIRUTA_A	VIRUTA_B	TIEMPO_CAL	TIEMPO_SETUP
1	2025-01-09 16:29:17.000	1	238	34	13	5	0	0
2	2025-01-09 16:20:17.000	1	240	34	13	5	0	0
3	2025-01-09 16:11:17.000	1	240	34	13	5	0	0
4	2025-01-09 16:02:17.000	1	238	35	13	5	0	0
5	2025-01-09 15:53:17.000	1	238	35	13	5	0	0
6	2025-01-09 15:44:17.000	1	240	35	13	5	0	0
7	2025-01-09 15:35:17.000	1	240	35	13	5	0	0
8	2025-01-09 15:26:17.000	1	238	35	13	5	0	0
9	2025-01-09 15:17:17.000	1	240	35	13	4	0	0
10	2025-01-09 15:08:17.000	1	240	34	13	5	0	0
11	2025-01-09 14:59:17.000	1	242	34	13	5	0	0
12	2025-01-09 14:50:17.000	1	240	34	13	5	0	0
13	2025-01-09 14:41:17.000	1	242	34	13	4	0	0
14	2025-01-09 14:32:17.000	1	242	34	13	4	0	0
15	2025-01-09 14:23:17.000	1	238	34	13	4	0	0
16	2025-01-09 14:14:17.000	1	240	34	13	5	0	0
17	2025-01-09 14:05:17.000	1	242	34	13	4	0	0
18	2025-01-09 13:56:17.000	1	242	34	13	5	0	0
19	2025-01-09 13:47:17.000	1	240	34	12	4	0	0
20	2025-01-09 13:38:17.000	1	242	34	13	4	0	0
21	2025-01-09 13:29:18.000	1	242	34	13	4	0	0
22	2025-01-09 13:20:19.000	1	240	34	12	4	0	0
23	2025-01-09 13:11:17.000	1	240	34	12	5	0	0
24	2025-01-09 13:02:17.000	1	242	34	13	4	0	0
25	2025-01-09 12:53:18.000	1	242	34	12	4	0	0
26	2025-01-09 12:35:18.000	1	242	34	12	4	0	0
27	2025-01-09 12:26:17.000	1	242	34	12	4	0	0
28	2025-01-09 12:17:17.000	1	242	34	12	4	0	0
29	2025-01-09 12:08:17.000	1	240	34	12	4	0	0
30	2025-01-09 11:59:18.000	1	242	34	12	4	0	0
31	2025-01-09 11:50:18.000	1	242	34	12	4	0	0
32	2025-01-09 11:41:17.000	1	238	34	12	4	0	0
33	2025-01-09 11:32:17.000	1	238	34	12	4	0	0

Ilustración 57: Tabla PARAM_EXT42 con datos recientes

En esta tabla de la figura 57 se concentran variables del proceso de extrusión, como la presión de la masa, la temperatura del material, los pesos de las virutas A y B, el estado de la máquina y el tiempo de calentamiento.

La presión de masa tiene un valor promedio de 240 bar, dentro de los límites establecidos para un proceso de extrusión eficiente. La temperatura del material se mantiene en 34 °C, una temperatura adecuada para el manejo de los compuestos de materia prima específicos de esta línea. Los pesos de las virutas A y B son 34 kg y 13 kg, respectivamente, lo que indica un exceso de viruta en la cortadora de la línea A de la máquina de extrusión. El estado de la máquina está definido como "Produciendo" (valor de 1). Las columnas de tiempo de calentamiento y tiempo de setup registran valores de 0 horas, confirmando que el sistema no ha requerido ajustes o mantenimientos en los últimos intervalos registrados.

Estos datos corroboran que el sistema es capaz de diferenciar claramente los estados operativos de la máquina y proporciona información detallada sobre los parámetros del proceso.

3.3.3 Resultados del algoritmo de regresión

A continuación, se detalla información relevante del proceso de entrenamiento del modelo, como primera sección se encuentra lo observado por la ilustración 58, allí se puede observar parte de proceso de entrenamiento por época, la ilustración antes mencionada muestra el progreso del entrenamiento de un modelo de aprendizaje profundo durante varias épocas, específicamente entre las épocas 97, 98 y 99 de un total de 100. El modelo tiene dos salidas: una para clasificación y otra para regresión. Los indicadores principales incluyen precisión (classification_output_accuracy), pérdida de clasificación (classification_output_loss), y error medio absoluto para la regresión (regression_output_mae). Además, se reportan valores de validación (val_) para ambas tareas.

En la salida de clasificación, la precisión (classification_output_accuracy) se mantiene constante alrededor de 0.88, indicando un desempeño sólido en la predicción de las clases. Las pérdidas de clasificación (classification_output_loss) fluctúan entre 0.37 y 0.45, lo que refleja una reducción constante en el error. Por otro lado, en la salida de regresión, el modelo alcanza un error medio absoluto (regression_output_mae) de entre 10.7 y 12.6, mientras que las pérdidas de regresión (regression_output_loss) se encuentran alrededor de 175, mostrando también un comportamiento estable. Los valores de validación (val_) confirman el buen desempeño del modelo, con una precisión de clasificación similar a la del entrenamiento (0.88) y pérdidas controladas. El tiempo de procesamiento por paso es bajo, variando entre 2 ms y 53 ms, lo que indica un modelo eficiente en términos computacionales. Estos resultados sugieren que el modelo está bien entrenado, mostrando un ajuste adecuado para ambas tareas en las etapas finales.

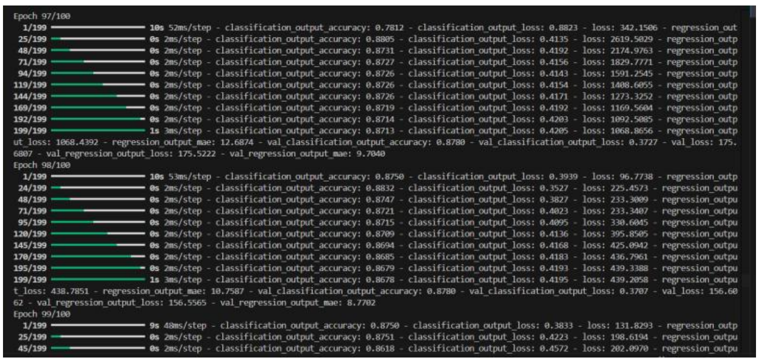

```
Epoch 97/100
  1/199 10s 52ms/step - classification_output_accuracy: 0.7812 - classification_output_loss: 0.8823 - loss: 342.1506 - regression_out
 25/199 0s 2ms/step - classification_output_accuracy: 0.8805 - classification_output_loss: 0.4135 - loss: 2619.5029 - regression_outp
 48/199 0s 2ms/step - classification_output_accuracy: 0.8731 - classification_output_loss: 0.4192 - loss: 2174.9763 - regression_outp
 71/199 0s 2ms/step - classification_output_accuracy: 0.8727 - classification_output_loss: 0.4156 - loss: 1829.7771 - regression_outp
 94/199 0s 2ms/step - classification_output_accuracy: 0.8726 - classification_output_loss: 0.4143 - loss: 1591.2545 - regression_outp
119/199 0s 2ms/step - classification_output_accuracy: 0.8726 - classification_output_loss: 0.4154 - loss: 1408.6055 - regression_outp
144/199 0s 2ms/step - classification_output_accuracy: 0.8726 - classification_output_loss: 0.4171 - loss: 1273.3252 - regression_outp
169/199 0s 2ms/step - classification_output_accuracy: 0.8719 - classification_output_loss: 0.4192 - loss: 1169.5604 - regression_outp
192/199 0s 2ms/step - classification_output_accuracy: 0.8714 - classification_output_loss: 0.4203 - loss: 1092.5085 - regression_outp
199/199 1s 3ms/step - classification_output_accuracy: 0.8713 - classification_output_loss: 0.4205 - loss: 1068.8656 - regression_outp
ut_loss: 1068.4392 - regression_output_mae: 12.6874 - val_classification_output_accuracy: 0.8780 - val_classification_output_loss: 0.3727 - val_loss: 175.
6807 - val_regression_output_loss: 175.5222 - val_regression_output_mae: 9.7040
Epoch 98/100
  1/199 10s 53ms/step - classification_output_accuracy: 0.8750 - classification_output_loss: 0.3939 - loss: 96.7738 - regression_outp
 24/199 0s 2ms/step - classification_output_accuracy: 0.8832 - classification_output_loss: 0.3527 - loss: 225.4573 - regression_outpu
 48/199 0s 2ms/step - classification_output_accuracy: 0.8747 - classification_output_loss: 0.3827 - loss: 233.3009 - regression_outpu
 71/199 0s 2ms/step - classification_output_accuracy: 0.8721 - classification_output_loss: 0.4023 - loss: 233.3407 - regression_outpu
 95/199 0s 2ms/step - classification_output_accuracy: 0.8715 - classification_output_loss: 0.4095 - loss: 330.6045 - regression_outpu
120/199 0s 2ms/step - classification_output_accuracy: 0.8709 - classification_output_loss: 0.4136 - loss: 395.8505 - regression_outpu
145/199 0s 2ms/step - classification_output_accuracy: 0.8694 - classification_output_loss: 0.4168 - loss: 425.0942 - regression_outpu
170/199 0s 2ms/step - classification_output_accuracy: 0.8685 - classification_output_loss: 0.4183 - loss: 436.7961 - regression_outpu
195/199 0s 2ms/step - classification_output_accuracy: 0.8679 - classification_output_loss: 0.4193 - loss: 439.3388 - regression_outpu
199/199 1s 3ms/step - classification_output_accuracy: 0.8678 - classification_output_loss: 0.4195 - loss: 439.2058 - regression_outpu
t_loss: 438.7851 - regression_output_mae: 10.7587 - val_classification_output_accuracy: 0.8780 - val_classification_output_loss: 0.3707 - val_loss: 156.60
62 - val_regression_output_loss: 156.5565 - val_regression_output_mae: 8.7702
Epoch 99/100
  1/199 9s 48ms/step - classification_output_accuracy: 0.8750 - classification_output_loss: 0.3833 - loss: 131.8293 - regression_outp
 25/199 0s 2ms/step - classification_output_accuracy: 0.8751 - classification_output_loss: 0.4223 - loss: 198.6194 - regression_outpu
 45/199 0s 2ms/step - classification_output_accuracy: 0.8618 - classification_output_loss: 0.4572 - loss: 202.0970 - regression_outpu
```

Ilustración 58: Resultados Preliminares del Entrenamiento

3.3.3.1 *Modelos entrenados*

3.3.3.1.1 Primer Modelo

Dentro de todo el proceso de entrenamiento la red paso por varias fases de ajuste y calibración de características con el fin de que la red mejore su proceso de predicción. En inicio se comenzó con características de 50 épocas de entrenamiento con un conjunto de datos de 32, tal como se puede observar en la ilustración 59; esto hizo que se tengas resultados bastante dispersos y lejanos del valor de predicción de la vida útil motor.

```
# Entrenamiento
# Entrenar el modelo
history = model.fit(
    X_train,
    [y_train.iloc[:, 0], y_train.iloc[:, 1]],  # y_train debe ser una lista con dos salidas
    epochs=50,  # Número de épocas para entrenar
    batch_size=32,  # Tamaño del lote
    validation_split=0.2,  # División para validación durante el entrenamiento
    verbose=1  # Mostrar el progreso del entrenamiento
)
```

Ilustración 59: Características de primeros entrenamientos – Modelo 1

Lo que se logró obtener como resultados de este primer entrenamiento los describe la ilustración 60: el modelo muestra buenos resultados en la tarea de regresión, con una pérdida de regresión de 65.12 y un MAE de 0.8637, lo que indica que las predicciones son bastante erróneas. Por otro lado, la pérdida de clasificación de 0.5706 (57%) sugiere que el modelo tiene dificultades para clasificar correctamente las muestras, lo que podría deberse a un desbalance de clases o a una arquitectura no óptima., como se puede observar en la ilustración 60.

```
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - classification_output_accuracy: 0.8599 - classification_output_loss: 0.5842 - loss: 69.5653 - regression_output
_loss: 68.9589 - regression_output_mae: 4.3530
Pérdida total: 66.40790557861328
Pérdida regresión: 65.12422943115234
Pérdida clasificación: 0.5748600363731384
MAE regresión: 0.8636820912361145
```

Ilustración 60: Resultados de precisión de clasificación y predicción – Modelo 1

3.3.3.1.2 Segundo Modelo

Para el segundo modelo se usaron las características descritas por la ilustración 61, donde se encuentran modificaciones de épocas ahora cambias a 100 y disminuyendo su grupo de muestra a 20.

```
# Entrenar el modelo
history = model.fit(
    X_train,
    [y_train.iloc[:, 0], y_train.iloc[:, 1]],  # y_train debe ser una lista con dos salidas
    epochs=100,
    batch_size=20,  # Tamaño de lote ajustado
    validation_split=0.2,
    verbose=1,
    callbacks=[early_stopping]  # Añadir EarlyStopping
)
```

Ilustración 61: Características de entrenamiento – Modelo 2

Como resultados de este segundo entrenamiento se logró obtener que el modelo muestra buenas mejoras al momento de realizar la regresión, con una pérdida de regresión de 3.7506 y un MAE de 0.8637, lo que indica que las predicciones son bastante cercanas a valor real. Sin embargo, la pérdida de clasificación de 0.4034 (40%) empeorando su sistema de clasificación desde el primer modelo, por lo que se observa una mejora significativa en parte de la red pero que aun necesita ajuste y modificaciones de parámetros, tal como lo indica en la ilustración 62

```
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - classification_output_accuracy: 0.8599 - classifica
loss: 2.6701 - regression_output_mae: 0.4610
Pérdida total: 3.750659704208374
Pérdida regresión: 3.304534673690796
Pérdida clasificación: 0.40349081158638
MAE regresión: 0.8636820912361145
Precisión clasificación: 0.4481223225593567
```

Ilustración 62: Resultados de predicción y clasificación – Modelo 2

3.3.3.1.3 Modelo Final

Después de varios entrenamientos fallidos, se logró entrenar un modelo que cumplía con los resultados esperados, para ellos se usaron las siguientes modificaciones en los parámetros de la red, con 120 épocas de entrenamiento y un grupo de análisis de 30. Toda esta información es visible en la ilustración 63 siguiente.

```python
# Entrenar el modelo
history = model.fit(
    X_train,
    [y_train.iloc[:, 0], y_train.iloc[:, 1]],  # y_train debe ser una lista con dos salidas
    epochs=120,
    batch_size=30,  # Tamaño de lote ajustado
    validation_split=0.2,
    verbose=1,
    callbacks=[early_stopping]  # Añadir EarlyStopping
)
```

Ilustración 63: Características de entrenamiento – Modelo Final

En los resultados plasmados en la ilustración 64, las mejoras que se observó en la perdida de regresión (0.35) y el MAE(Mean Absolute Error o error absoluto medio) de regresión (0.86) indican que la red se encuentra en puntos óptimos para realizar predicción de la vida útil de motor, ya lo que indica el perdida de precisión es que tan lejos va a estar el valor que proporcione la red del valor real; el estándar para esta medida en el área de redes neuronales es de 0.5 para que la red se considere eficaz para realizar una tarea de predicción al igual que el MAE que siempre se requiere que tienda a cero para minorizar el error. Por último, se puede observar que la precisión de clasificación también sufrió un cambio, pero esta vez positivo al llegar a 0.816 (aprox. 81.6%) dotando a la red de confiabilidad al momento de predecir estados del motor.

```
 1/63 ━━━━━━━━━━━━━━━━━━━━ 19s 313ms/step - classification_output_accuracy: 0.8125 - classification_output_loss: 0.4906 - lo
30/63 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - classification_output_accuracy: 0.8575 - classification_output_loss: 0.4671 - loss:
61/63 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - classification_output_accuracy: 0.8597 - classification_output_loss: 0.4648 - loss:
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - classification_output_accuracy: 0.8599 - classification_output_loss: 0.4646 - loss:
loss: 0.1730 - regression_output_mae: 0.2054
Pérdida total: 0.8131304383277893
Pérdida regresión: 0.3500424027442932
Pérdida clasificación: 0.46063998341560364
MAE regresión: 0.8636820912361145
Precisión clasificación: 0.81620149910449982
```

Ilustración 64: Resultados de precisión de predicción y clasificación–Modelo Final

3.3.3.1.4 Resultados Gráficos de Modelo Final

La ilustración 65 muestra que la pérdida del modelo disminuye drásticamente durante las primeras 20 épocas, pasando de valores iniciales extremadamente altos a cifras cercanas a cero. Esto indica que el modelo aprende rápidamente en las etapas iniciales y, después de estabilizarse, alcanza un punto en el que tanto la pérdida de entrenamiento como la de validación convergen, manteniéndose casi idénticas. Este comportamiento refleja un equilibrio adecuado entre el ajuste al conjunto de entrenamiento y la capacidad del modelo para generalizar en datos no vistos.

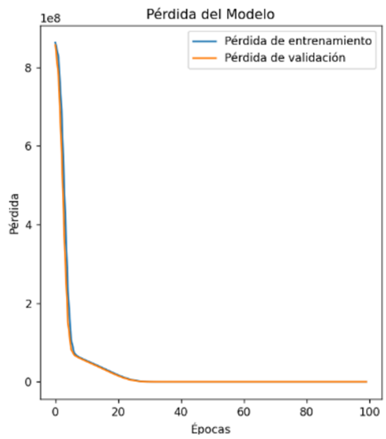


Ilustración 65: Pérdida del Modelo Final

Por otro lado, la ilustración 66 detalla la precisión de la tarea de clasificación, muestra un desempeño robusto con valores iniciales cercanos a 0.8 que mejoran rápidamente hasta estabilizarse en torno a 0.87 a partir de la época 10. Sin embargo, se observan picos negativos en la precisión de validación alrededor de las épocas 40 y 60, lo que sugiere dificultades momentáneas del modelo para generalizar correctamente en esos momentos. A pesar de ello, el modelo logra recuperarse de manera consistente, mostrando que, aunque existieron variaciones temporales, el proceso de entrenamiento fue efectivo y las predicciones se mantuvieron estables.

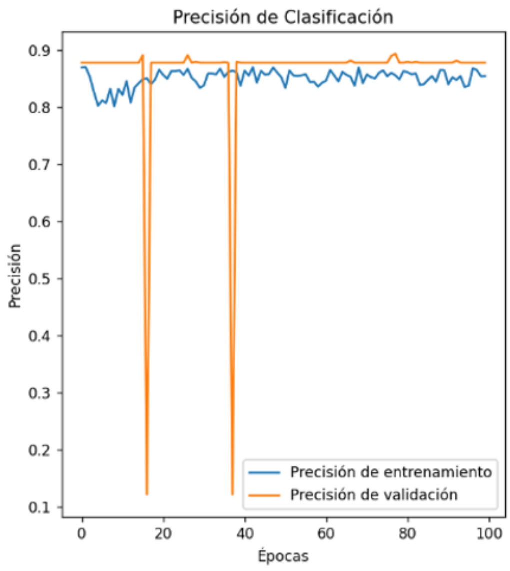


Ilustración 66: Precisión por clasificación del Modelo Final

En complemento, los resultados reflejan un modelo bien ajustado que ha aprendido tanto los patrones presentes en los datos de entrenamiento como en los de validación. La cercanía entre las métricas de entrenamiento y validación en ambos gráficos refuerza esta idea, sugiriendo que el modelo tiene un buen equilibrio entre precisión y generalización; Los picos en la precisión de validación podrían ser analizados más a fondo para garantizar una mayor estabilidad en futuros ajustes o configuraciones del entrenamiento.

3.3.3.2 Pruebas

Para las pruebas de la Red Neuronal se desarrolló un script, donde se carga el modelo de red y el modelo escalador (modelo que se encarga de convertir todos los datos naturales que se tienen de proyecto, a datos que pueda entender la red). La ilustración 67 muestra el código y estructura utilizada para realizar consulta a la red.

```python
from tensorflow.keras.models import load_model
import joblib

# Cargar el modelo entrenado
model = load_model('smart_motor_modelonuevo.keras')

# Cargar el escalador guardado
scaler_X = joblib.load('scaler_X1.pkl')

# Crear los datos de prueba con las mismas columnas usadas para entrenar el modelo
# Asegúrate de que las columnas sean las mismas y en el mismo orden
new_data = pd.DataFrame({
    'TEMPERATURA _MOTOR [°c]': [45],
    'VIBRACION_X_MOTOR [mm/s]': [0.31],  # Ejemplo de vibración en el eje X
    'VIBRACION_Z_MOTOR [mm/s]': [0.42], # Ejemplo de vibración en el eje Z
    'AMPERAJE_MOTOR [Amps]': [54],  # Ejemplo de amperaje del motor
    'TIEMPO_TOTAL_PRODUCCION': [1500],  # Ejemplo de tiempo total de producción
    'RPM_MOTOR': [12],  # Ejemplo de RPM del motor
    "Status_temp": [0],
    "status_vib_x": [0],
    "status_vib_z": [0]
})

# Normalizar los nuevos datos usando el escalador
new_data_scaled = scaler_X.transform(new_data)

# Realizar las predicciones
predictions = model.predict(new_data_scaled)

# Obtener las predicciones de regresión y clasificación
predicted_regression = predictions[0]  # Tiempo de vida restante
predicted_classification = predictions[1]  # Status General

# Mostrar las predicciones
print(f"Predicción de Tiempo de Vida Restante (Regresión): {predicted_regression}")
print(f"Predicción de Status General (Clasificación): {'Estable' if predicted_classification < 0.5 else 'Irregular'}")
```

Ilustración 67: Programación de Pruebas de la Red

Usando de modelo los datos presentados por la ilustración 68, se obtuvo resultados muy buenos, y que se puede observar que los datos de consulta envían son datos de condiciones normales en el uso de motor de la extrusora, con temperaturas de 45 °C, 54

Amperios de corriente, y vibraciones estándar en X y Z menores a 0.5 y un tiempo de producción de 1500 horas; la red debe dar como resultados la vida útil de un motor con un estándar de horas de 30000 horas de vida útil.

Según la ilustración 69, la Red proporciona un valor de 28508.086H de trabajo, mientras que por datos de la data y resultados esperado debería ser 28500 horas, se puede observar que la red hace un muy buen trabajo al momento de predecir la vida útil teniendo un margen de error (0.03%) resultados obtenidos gracias a la configuración de la red y a la extensa data manejada de aproximadamente 10.000 datos recolectados.

```
new_data = pd.DataFrame({
    'TEMPERATURA _MOTOR [°c]': [45],
    'VIBRACION_X_MOTOR [mm/s]': [0.31],  # Ejemplo de vibración en el eje X
    'VIBRACION_Z_MOTOR [mm/s]': [0.42], # Ejemplo de vibración en el eje Z
    'AMPERAJE_MOTOR [Amps]': [54],  # Ejemplo de amperaje del motor
    'TIEMPO_TOTAL_PRODUCCION': [1500],  # Ejemplo de tiempo total de producción
    'RPM_MOTOR': [12],  # Ejemplo de RPM del motor
    "Status_temp": [0],
    "status_vib_x": [0],
    "status_vib_z": [0]
})
```

Ilustración 68: Datos de prueba de la Red Neuronal

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 122ms/step
Predicción de Tiempo de Vida Restante (Regresión): [[28508.086]]
Predicción de Status General (Clasificación): Estable
PS C:\Users\eduar\OneDrive\Desktop\Red_neuronal>
```

Ilustración 69: Resultados obtenido de la red neuronal

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

El desarrollo del sistema de monitoreo para la línea extrusora representa un avance significativo en la implementación de tecnologías inteligentes dentro de la industria de tuberías plásticas. Este proyecto demostró ser capaz de abordar una importante problemática: la falta de monitoreo confiable de las variables operativas esenciales, que históricamente impactan tanto la calidad del producto como el consumo energético.

Entre las fortalezas más destacadas del sistema desarrollado se encuentra su modularidad y escalabilidad. La integración de herramientas como Node-RED, MQTT, MariaDB y Grafana permitió crear un flujo de datos eficiente y un sistema visualmente intuitivo que puede ampliarse con facilidad para incluir nuevas variables o sensores según las necesidades futuras. Además, la incorporación de un algoritmo de regresión para estimar la vida útil del motor, basado en datos históricos, marcó un paso hacia el mantenimiento predictivo, reduciendo tiempos de inactividad y mejorando la planificación operativa.

Sin embargo, se identificaron algunas debilidades durante la ejecución. Por ejemplo, la dependencia de la infraestructura de red para la transmisión de datos introduce vulnerabilidades que podrían afectar la estabilidad del sistema en caso de interrupciones en la conexión (que es una situación algo frecuente). Además, aunque el sistema ofrece un diseño flexible, la configuración inicial de algunos componentes, como los nodos de comunicación y la base de datos, requiere conocimientos técnicos específicos que podrían dificultar su implementación por parte de operadores sin formación técnica previa.

En comparación con otros trabajos similares, este proyecto destaca por su enfoque en tecnologías accesibles y de bajo costo, lo que lo hace viable para empresas pequeñas y medianas, pero no una opción muy recomendable para las más grandes que requieren sistemas más robustos. La elección de herramientas de código abierto y dispositivos como las Raspberry Pi refuerza su sostenibilidad económica, posicionándolo como una solución innovadora dentro del contexto de la industria local.

4.1 Conclusiones

- El sistema desarrollado cumplió con los objetivos planteados al inicio del proyecto, proporcionando una herramienta confiable para la adquisición, monitoreo y análisis de variables operativas en la línea extrusora.
- La integración de tecnologías como MQTT, Node-RED y Grafana permitió crear un flujo de datos continuo y eficiente, que mejoró el control observacional del proceso productivo.
- El algoritmo de regresión implementado con los datos del motor contribuyó al mantenimiento predictivo, el cual es capaz de reducir los riesgos de fallos inesperados y optimizar los tiempos de operación.
- La solución es económicamente viable respecto a otros sistemas, lo que la hace replicable en empresas con recursos limitados.
- El sistema no solo mejoró la supervisión en tiempo real, sino que también ofreció la capacidad de análisis histórico, permitiendo decisiones más informadas y estratégicas.

4.2 Recomendaciones

- Reducir el número de dispositivos redundantes para garantizar la estabilidad del sistema y prevenir interrupciones en la transmisión de datos.
- Realizar capacitaciones al personal para que utilicen el sistema de la manera correcta, de tal manera que contribuyan a la disminución de tiempos de inactividad.
- Ampliar el sistema con sensores adicionales para monitorear variables como el consumo de energía, número de tubos fabricados en cierto período de tiempo, etc.
- Incorporar análisis avanzados basados en machine learning para mejorar la precisión del mantenimiento predictivo y anticipar fallos con mayor antelación.
- Realizar estudios a largo plazo sobre el impacto económico del sistema, evaluando ahorros energéticos, reducción de fallos y mejoras en la calidad del producto.

- Considerar la implementación de un sistema de almacenamiento en la nube para respaldar los datos en caso de borrados accidentales de la base de datos y sus tablas.

BIBLIOGRAFÍA

5. BIBLIOGRAFÍA

[1] S. Toro, «Caracterización energética de ladrillera y,» 2022.

[2] «Blog Danzé Trade | Maquinaria y materiales para los sectores del plástico, espumas, papel y cartón,» 03 08 2021. [En línea]. Available: https://blog.fdtecsl.com/tecno-system-extrusoras-y-lineas-completas-para-extrusion-de-materiales-termoplasticos. [Último acceso: 12 2024].

[3] A. Maquinaria, «5 puntos que debes saber sobre las tuberías de plástico,» 20 01 2016. [En línea]. Available: https://www.aristegui.info/5-puntos-que-debes-saber-sobre-las-tuberias-de-plastico/.

[4] SIEMENS, 2014.

[5] R. P. Foundation, 2019.

[6] J. A. C. Falcon, WI-Fi, Lo que necesita conocer, Espana: Grupo Ramirez Cogollor, 2028.

[7] I. SA, «Diseño de redes Wifi en Monterrey,» 21 04 2024. [En línea]. Available: https://intertres.com/diseno-de-redes-wifi-en-monterrey/.

[8] J. &. B. A. Oasis, MQTT Essentials: A Beginner's Guide to the Protocol for the Internet of Things, Packt Publishing, 2020.

[9] Paessler, «MQTT,» 15 01 2025. [En línea]. Available: https://www.paessler.com/es/it-explained/mqtt.

[10] R. &. G. P. Olson, Ethernet and Networking Fundamentals., McGraw-Hill Education., 2021.

[11] I. D. Guide, «¿Qué es Ethernet (IEEE 802.3)?,» 08 12 2022. [En línea]. Available: https://www.ionos.es/digitalguide/servidores/know-how/ethernet-ieee-8023/.

[12] B. G. Peñaranda, DESARROLLO DE UN SISTEMA IOT PARA EL MONITOREO REMOTO DE VARIABLES EN UN HUERTO AGRÍCOLA, Cuenca, 2024.

[13] Node-Red, «node-red-contrib-s7,» 2020.

[14] Node-Red, «node-red-contrib-mysql-config,» 2023.

[15] S. Chanthakit, «MQTT Based Air Quality Monitoring System,» Bangkok, 2018.

[16] A. P. K. Mainak Chakraborty, «Monitoring Cloud-NAtive Appications,» 2021.

[17] M. Foundation, «MariaDB,» 2023. [En línea]. Available: https://mariadb.org/. [Último acceso: 05 11 2024].

[18] D. Community, «DBeaver,» 2024. [En línea]. Available: https://dbeaver.io/. [Último acceso: 05 11 2024].

[19] Banner, «QM30VT siries».

[20] BANNER, 2022.

[21] F. Chollet, Deep Learning with Python., Manning Publications., 2018.

[22] «Interactive Chaos,» 2021. [En línea]. Available: https://interactivechaos.com/es/manual/tutorial-de-machine-learning/estructura-de-una-red-neuronal.

[23] V. S. Cerda, «Disenio, implementaicon y evaluacion de un ared neuronal de regresion para clasifiacion de frutas,» Valencia.

[24] W. Collobert R., «A unified architecture for natural language processing,» New York, 2019.

[25] Itop.es, «Tensorflow,» [En línea]. Available: https://www.itop.es/soluciones-tecnologicas/business-analytics-business-intelligence/tensorflow.html.

[26] J. Torres, Deep learnign -Introduccion a practicas con Keras, Barcelona, 2019.

[27] J. &. R. K. Kurose, Computer Networking: A Top-Down Approach, Pearson, 2017.

[28] F. d. M. Industrial, «¿Cómo se compone y funciona una línea de extrusión?,» [En línea]. Available: https://fabricantes-maquinaria-industrial.es/se-compone-funciona-una-linea-extrusion/. [Último acceso: 05 11 2024].

[29] P. &. M. J. Wright, The Industrial IoT: Realizing the Potential of Connected Manufacturing, Springer, 2021.

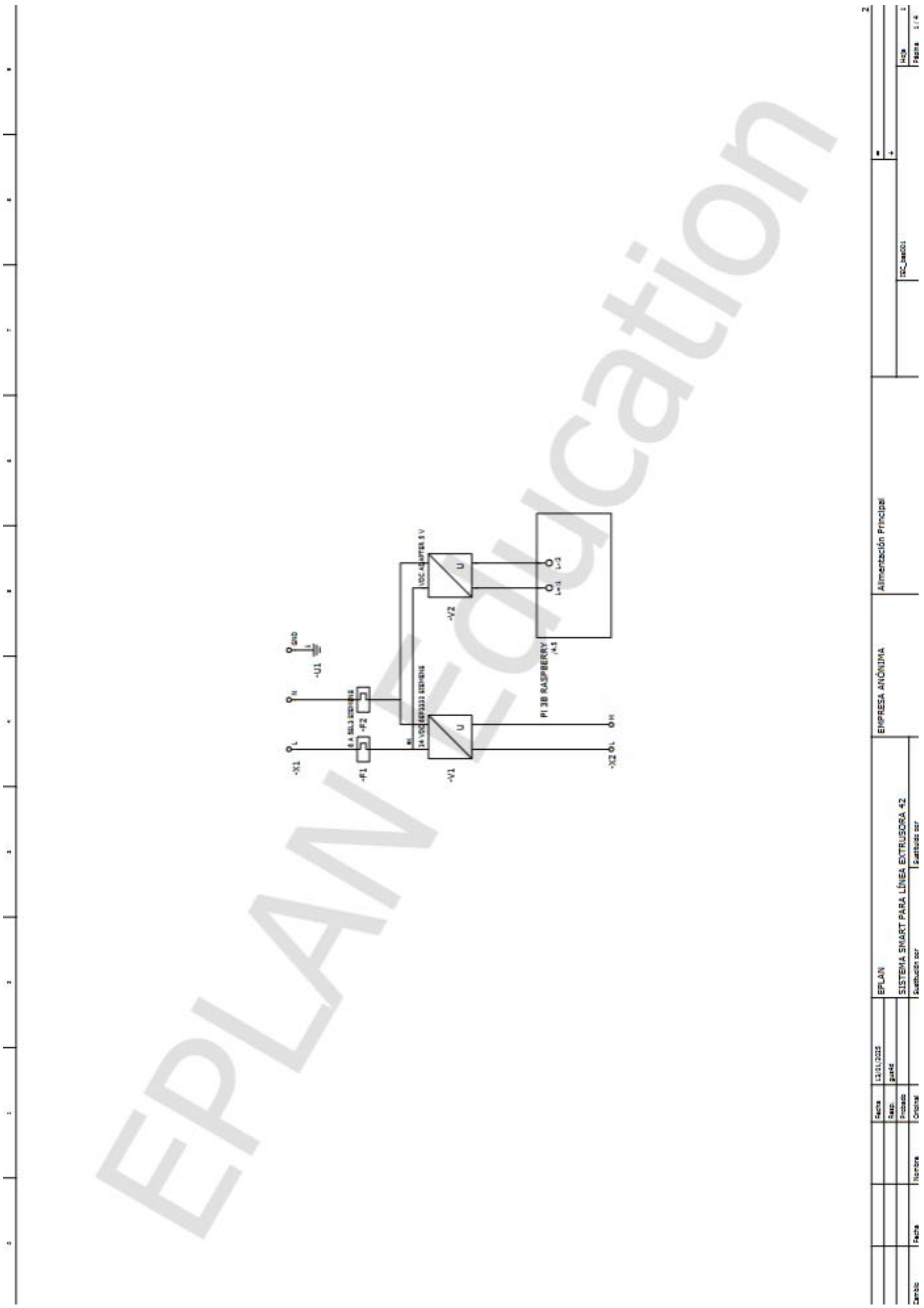
[30] J. B. &. A. J. Predd, Bridging Databases and IIoT: A Node-RED-based System for Data Acquisition and Analysis, Journal of Automation & Control Engineering, 2018.

[31] A. &. S. J. Hernández, IoT-Enhanced Industrial Systems: Application of Node-RED in Industrial Control, International Journal of Advanced Manufacturing Technology., 2019.

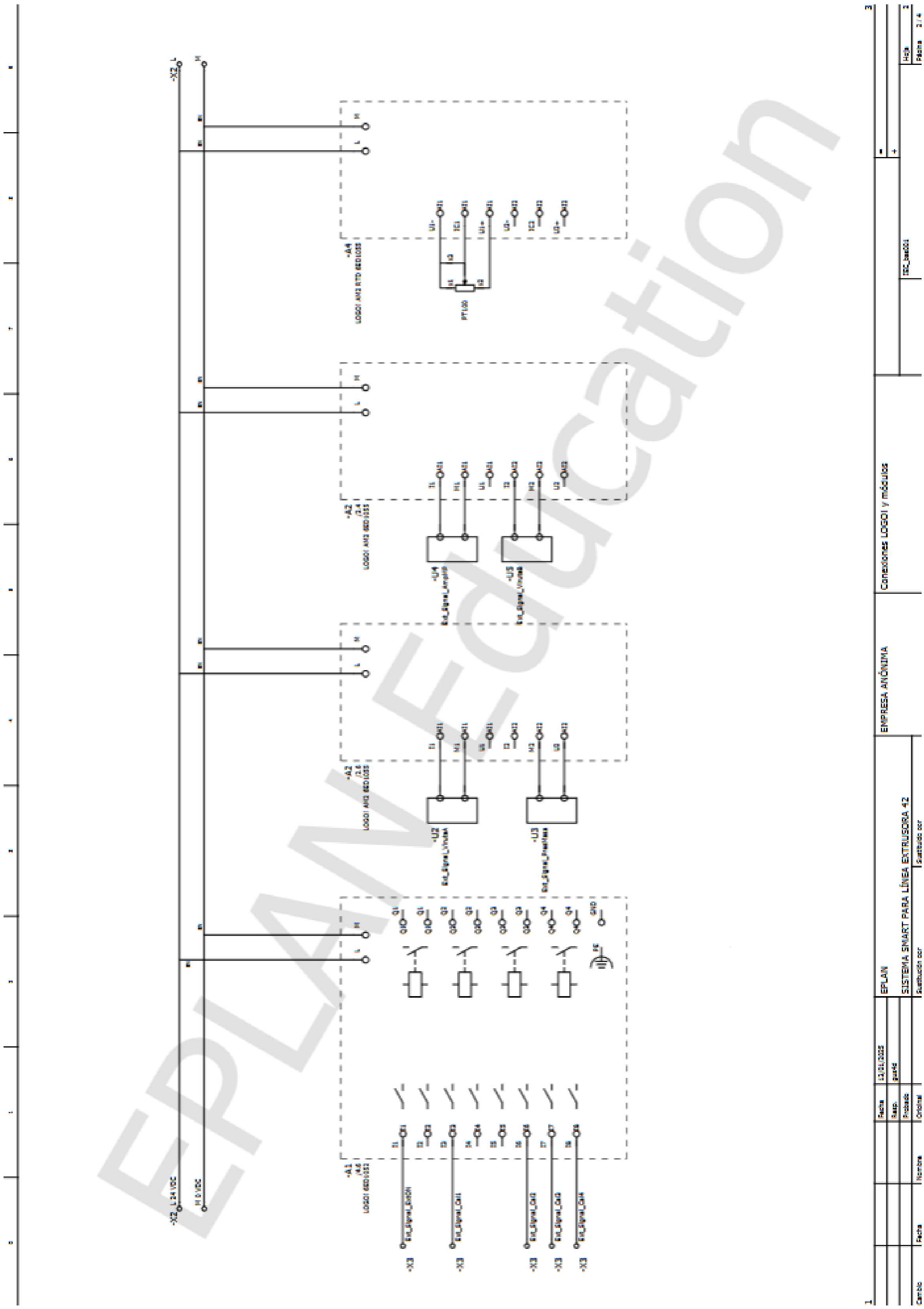
[32] I. B. Y. Goodfellow, Deep Learning, MIT Press, 2017.

APÉNDICES

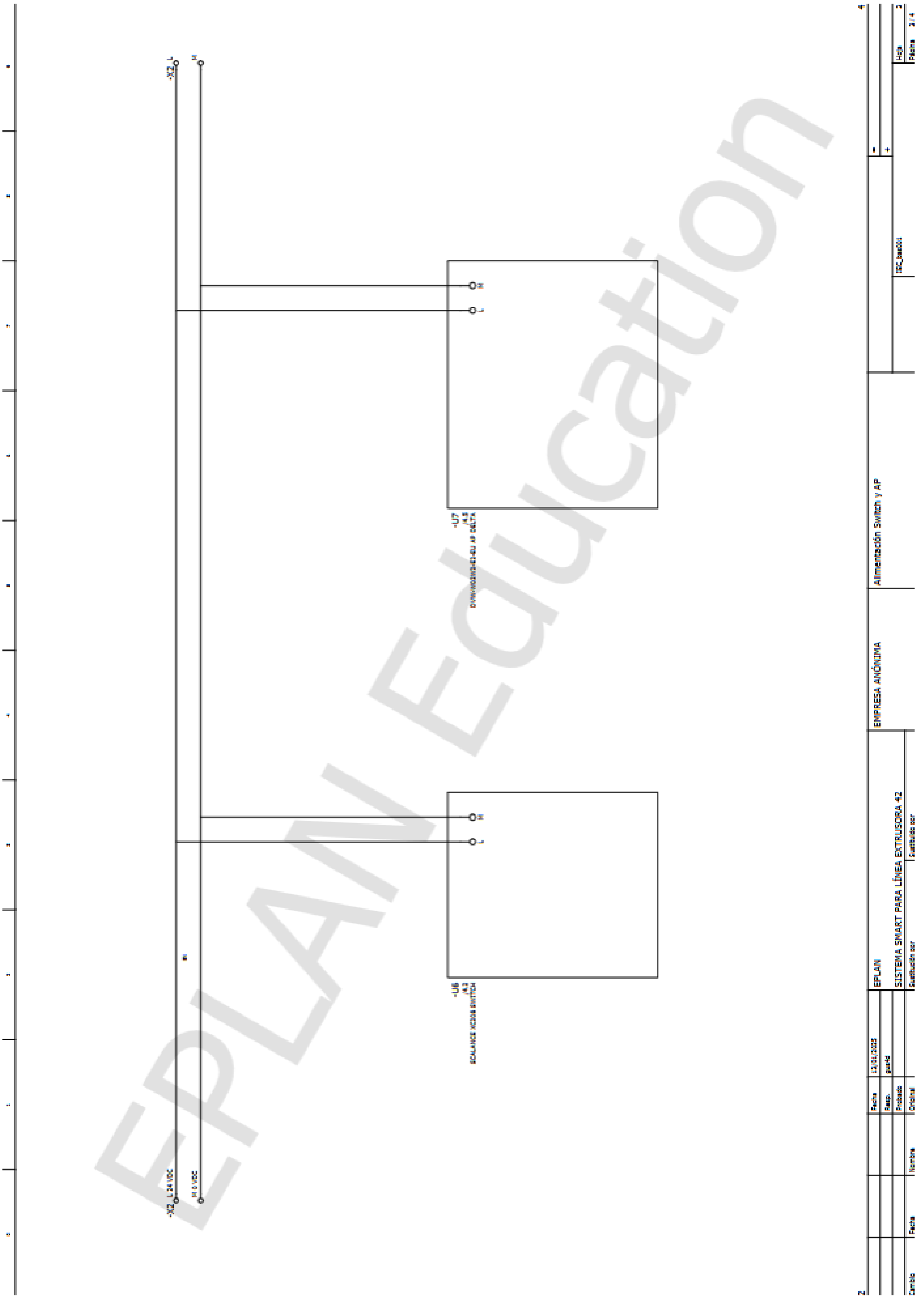
APÉNDICE A



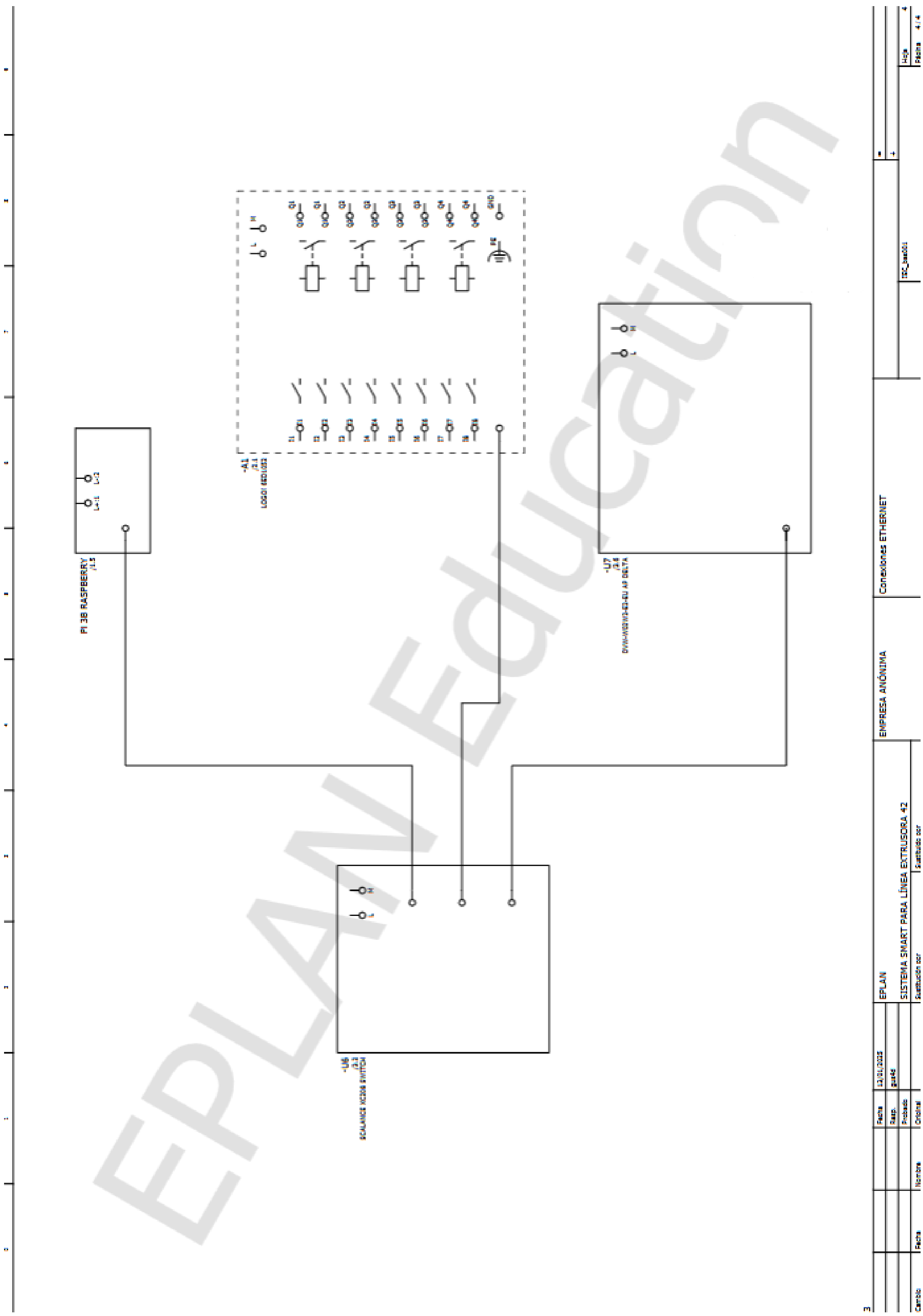
PLANO 1: Alimentación Principal



PLANO 2: Conexiones del PLC LOGO! y sus módulos



PLANO 3: Alimentación de Switch y AP



PLANO 4: Conexiones ETHERNET

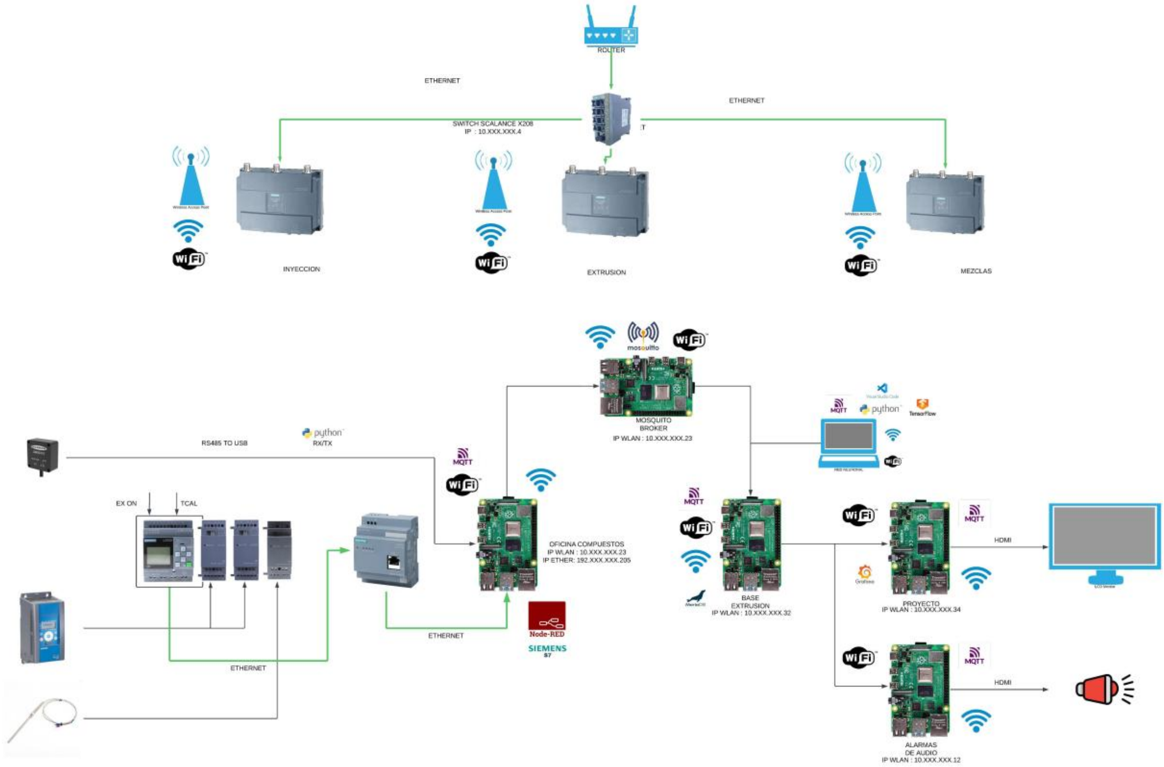


Ilustración 70: Diagrama de Conexiones Simplificado del Sistema

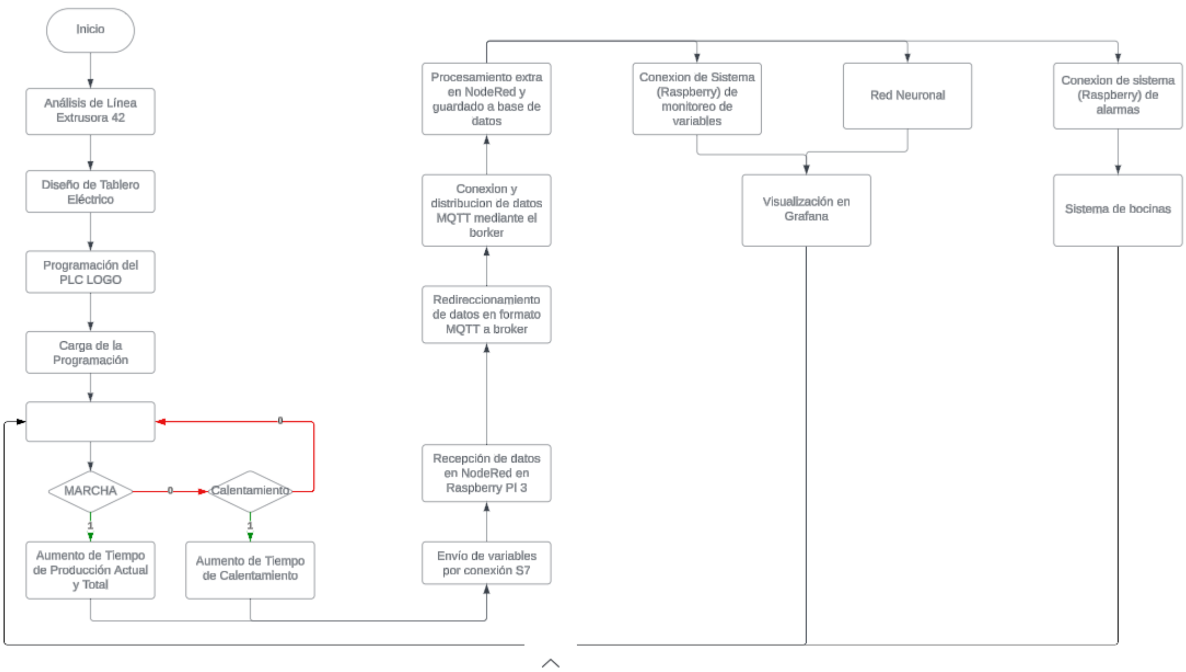


Ilustración 71: Diagrama de Flujo del Proyecto Integrador

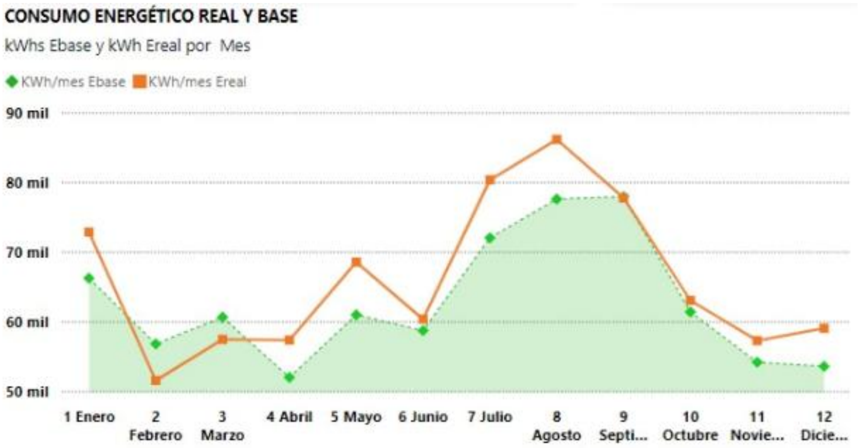


Ilustración 72: Evidencias de Disminución de Consumo Energético de XT-42

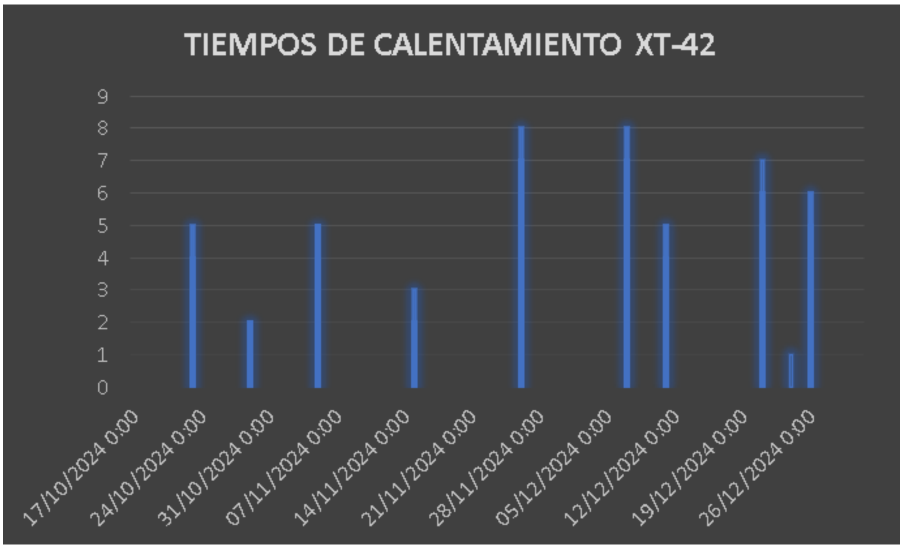


Ilustración 73: Evidencias de Disminución de Tiempos de Calentamiento de XT-42