



**UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE CUENCA
CARRERA DE INGENIERÍA ELECTRÓNICA**

**ANÁLISIS DE HERRAMIENTAS HTML Y JAVASCRIPT PARA EL DESARROLLO DE
PÁGINAS WEB APLICADAS A LA GESTIÓN DE DATOS Y VARIABLES DENTRO DE
SISTEMAS DE AUTOMATIZACIÓN BASADOS EN PLCS**

Trabajo de titulación previo a la obtención del
título de Ingeniero Electrónico

AUTOR: LUIS GONZALO VIÑAMAGUA PAREDES
TUTOR: ING. JULIO CÉSAR ZAMBRANO ABAD, MSc.

Cuenca - Ecuador
2022

**CERTIFICADO DE RESPONSABILIDAD Y AUDITORÍA DEL TRABAJO DE
TITULACIÓN**

Yo, Luis Gonzalo Viñamagua Paredes con documento de identificación N° 0107085441, manifiesto que:

Soy el autor y responsable del presente trabajo; y, autorizo a que, sin fines de lucro, la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Cuenca, 13 de septiembre del 2022

Atentamente,



Luis Gonzalo Viñamagua Paredes

0107085441

**CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE
TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA**

Yo, Luis Gonzalo Viñamagua Paredes con documento de identificación N° 0107085441, expreso mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del Proyecto técnico: “Análisis de herramientas HTML y JavaScript para el desarrollo de páginas web aplicadas a la gestión de datos y variables dentro de sistemas de automatización basados en PLCs”, el cual ha sido desarrollado para optar por el título de: Ingeniero Electrónico, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que hago la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, 13 de septiembre del 2022

Atentamente,



Luis Gonzalo Viñamagua Paredes

0107085441

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Julio César Zambrano Abad con documento de identificación N° 0301489696, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: ANALISIS DE HERRAMIENTAS HTML Y JAVASCRIPT PARA EL DESARROLLO DE PÁGINAS WEB APLICADAS A LA GESTIÓN DE DATOS Y VARIABLES DENTRO DE SISTEMAS DE AUTOMATIZACIÓN BASADOS EN PLCS, realizado por Luis Gonzalo Viñamagua Paredes con documento de identificación N° 0107085441, obteniendo como resultado final el trabajo de titulación bajo la opción Proyecto técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Cuenca, 13 de septiembre del 2022

Atentamente,

A handwritten signature in blue ink, enclosed in a blue oval. The signature reads "Julio César Zambrano Abad".

Ing. Julio César Zambrano Abad, MSc.

0301489696

ÍNDICE GENERAL

ÍNDICE DE FIGURAS.....	iv
ÍNDICE DE TABLAS	ix
AGRADECIMIENTOS	x
DEDICATORIA	xi
RESUMEN	xiii
INTRODUCCIÓN	xiv
ANTECEDENTES DEL PROBLEMA DE ESTUDIO.....	xvi
JUSTIFICACIÓN (IMPORTANCIA Y ALCANCES)	xvii
OBJETIVOS	xviii
Introducción al servidor web integrado en autómatas de la marca Siemens	1
1.1 Características básicas del servidor web	1
1.1.1 Prestaciones del servidor web	1
1.1.2 Ventajas respecto al uso de un servidor web.....	2
1.2 Proceso de implementación del servidor web	2
1.2.1 Configuración del servidor WEB	2
1.2.2 Acceso al servidor web a la CPU SIMATIC desde una PC o dispositivos HMI	3
1.3 Páginas del servidor web.....	3
1.3.1 Página de Inicio	4
1.3.2 Diagnóstico.....	5
1.3.3 Búfer de diagnóstico.....	6
1.3.4 Información del módulo	7
1.3.5 Avisos.....	9
1.3.6 Comunicación.....	9
1.3.7 Topología	11
1.3.8 Estado de variables.....	13
1.3.9 Tablas de observación	13
1.3.10 Páginas de usuario.....	13
1.3.11 Explorador de archivos.....	14
1.3.12 Datalogs.....	15
1.4 Procedimiento para crear y acceder a páginas web definidas por el usuario desde el servidor web	
15	
1.5 Comandos AWP	16
Configuración e instalación del proceso industrial a monitorear	19
2.1 Arquitectura del sistema de velocidad.....	19
2.1.1 Convertidores estándar SINAMICS G120	19
2.1.2 Módulo SIMATIC S7 1516-3 PN-DP	21
2.2 Configuración del servidor WEB mediante el programa TIA PORTAL V16	22

2.2.1	Activación del servidor web.....	22
2.2.2	Configuración del nivel de administración de los usuarios.....	23
2.2.3	Configuración de interfaces Ethernet para comunicación con el servidor	24
2.2.4	Asignación de dirección IP.....	25
2.2.5	Asignación de páginas web definidas por el usuario:	25
2.2.6	Instrucción WWW.....	26
2.3	Configuraciones previas en el variador SINAMICS G120 para el control de un motor asíncrono	
	27	
2.3.1	Configuración de puesta en marcha desde el panel IOP-2	27
2.4	Configuración del variador en TIA PORTAL V16 para la comunicación con el PLC S7-1500.31	
2.5	Configuración y uso de la librería "Library SINAMICS Extended" (LSINAExt).....	36
2.5.1	Librería (LSINAExt)	36
2.5.2	Bloque de función SINA_SPEED_TLG20	37
2.5.3	Telegrama estándar 20	37
2.6	Bloque de función SINA_SPEED_TLG20 en TIA PORTAL V16:	38
2.6.1	Parámetros de entrada	38
2.6.2	Parámetros de salida.....	40
2.7	Configuración del bloque SINA_SPEED_TLG20.....	41
Creación de la aplicación web mediante el uso de herramientas de lenguajes de programación basados en HTML y JavaScript.....		45
3.1	Normativas a considerar.....	46
3.2	Normas y estándares de diseño para HMI.....	46
3.3	Conceptos Generales de diseño web	53
3.4	Creación de páginas web:.....	54
3.4.1	Estructura y elementos de una página web	54
3.4.2	Elementos HTML (etiquetas).....	54
3.5	Programa requerido:.....	56
3.6	Librerías a utilizarse:.....	57
3.6.1	Bootstrap	57
3.6.2	Jquery	57
3.6.3	Smoothie charts	57
3.7	Diseño de la aplicación web.....	57
3.8	Desarrollo de la aplicación web en Visual Studio Code	59
3.9	Implementación del documento HTML principal.....	61
3.9.1	Sección de cabecera (head) del documento HTML	62
3.9.2	Sección de cuerpo (Body) del documento HTML	63
3.9.3	Implementación de la sección “Encabezado”	65
3.9.4	Implementación de la sección “Barra de Navegación”	65
3.9.5	Implementación de la sección “Puesta en marcha del motor”	68
3.9.6	Implementación de la sección “Lectura de Valores”	80
3.9.7	Implementación de la sección “Notificaciones”.....	84

3.9.8	Gestión de Alarmas y Notificaciones.....	86
3.9.9	Implementación de la sección “Gráficos”	89
4	Creación de la aplicación web usando Laravel para la gestión del almacenamiento y descarga de los datos del proceso	93
4.1	Conceptos generales:.....	93
4.1.1	Definición de una base de datos	93
4.1.2	Tipos de bases de datos	94
4.1.3	Servidor de base de datos	94
4.2	Programas necesarios	95
4.2.1	Servidor de base de datos: MYSQL	95
4.2.2	XAMPP	96
4.3	Desarrollo de la base de datos	96
4.4	Creación de la aplicación para la gestión de la información de la base de datos usando LARAVEL	
	101	
4.4.1	Framework LARAVEL	102
4.4.2	Instalación del framework Laravel	102
4.4.3	Creación del proyecto LARAVEL	103
4.4.4	Acceso al proyecto creado.....	104
4.4.5	Creación de un virtual host para el acceso al proyecto de Laravel.....	105
4.5	Implementación de la aplicación web desarrollada en LARAVEL	107
4.5.1	Implementación del “Modelo” de la aplicación.....	107
4.5.2	Migraciones.....	108
4.5.3	Creación del modelo “Registros”	109
4.5.4	Implementación del “controlador” de la aplicación.....	111
4.5.5	Implementación de los elementos “Vista”	119
Pruebas y resultados.....		123
5.1	Conexión del motor asíncrono	123
5.2	Configuración de la interfaz de red del dispositivo cliente	124
5.3	Configuraciones finales en TIA PORTAL.....	127
5.4	Carga del programa en el PLC	128
5.5	Pruebas de la aplicación WEB	130
5.5.1	Pruebas de la sección “Puesta en marcha del motor”.....	133
5.5.2	Gestión de alarmas y advertencias	139
5.5.3	Pruebas de los botones de la sección “barra de navegación” y la aplicación diseñada en Laravel	142
Conclusiones y recomendaciones		150
6.1	Conclusiones	150
6.2	Recomendaciones y trabajos futuros	151
Referencias Bibliográficas		152

ÍNDICE DE FIGURAS

Figura 1.1 Esquema de un servidor WEB vía comunicación PROFINET.....	1
Figura 1.2 Ajustes iniciales del servidor web en TIA PORTAL [3].....	2
Figura 1.3 Configuración de la lista de usuarios del servidor.....	3
Figura 1.4 Pagina "Intro" inicial del servidor web de un PLC S7-1500.....	4
Figura 1.5 Página inicial posterior al iniciar sesión en el servidor web.....	5
Figura 1.6 Página de diagnóstico en la sección de "Identificación".....	6
Figura 1.7 Ficha "Memoria" de la página de Diagnóstico.	6
Figura 1.8 Búfer de diagnóstico.....	7
Figura 1.9 Significado de cada símbolo de los estados posibles de los equipos de la estación del PLC [3].	7
Figura 1.10 Pagina estándar "Información del Módulo" [3].....	8
Figura 1.11 Ficha "Estadística" de la página estándar "Información del módulo" [3].....	8
Figura 1.12 Página estándar "Avisos" [3]	9
Figura 1.13 Página estándar "Comunicación".....	9
Figura 1.14 Ficha "Recursos" de la página estándar "Comunicación".....	10
Figura 1.15 Ficha "Conexiones" de la página estándar "Comunicación"	10
Figura 1.16 Vista grafica (Topología prevista y real).	11
Figura 1.17 Significado de los enlaces de acuerdo a su color en las respectivas topologías [3].....	11
Figura 1.18 Página “Topología” (Vista de tabla).....	12
Figura 1.19 Significado de los símbolos de estado de los dispositivos PROFINET [3]	12
Figura 1.20 Página estándar "Topología" en la ficha "Vista general de estado".....	12
Figura 1.21 Página estándar "Estado de variables" [3]	13
Figura 1.22 Página estándar "Tablas de observación"[3].....	13
Figura 1.23 Sección "Paginas de usuario".....	14
Figura 1.24 Página estándar "Navegador de archivos"	14
Figura 1.25 Página estándar "DataLogs" [3].....	15
Figura 1.26 Pasos para acceder a páginas propias del usuario desde el servidor WEB.	16
Figura 1.27 Comandos AWP con su respectiva sintaxis según su función [3]	17
Figura 2.1 Arquitectura del sistema de velocidad a ser controlado y monitorizado a través de la aplicación WEB.....	19
Figura 2.2 Unidad de potencia PM240-2 [6].....	20
Figura 2.3 SINAMICS G120, tamaños FSA, FSB y FSC con Control Unit CU240E 2 F y Basic Operator Panel BOP 2 [6].	21
Figura 2.4 SINAMICS G120 con la unidad CU250S 2 PN con control mediante el panel IOP-2	21
Figura 2.5 PLC 1516-3 PN/DP [7].....	22
Figura 2.6 Activación del servidor web.	23
Figura 2.7 Configuración de administración de usuarios.....	23
Figura 2.8 Configuración del nivel de acceso del servidor web.....	24
Figura 2.9 PLC Siemens 1516-3 PN/DP con sus respectivas interfaces PROFINET y PROFIBUS.....	24
Figura 2.10 Asignación de la dirección IP de la interfaz X1 para acceder al servidor WEB.....	25
Figura 2.11 Configuración y asignación de páginas definidas por el usuario en el servidor.....	26
Figura 2.12 Instrucción WWW en TIA PORTAL V16 (STEP 7) [13].....	26
Figura 2.13 Configuración del bloque WWW en TIA PORTAL	27
Figura 2.14 Puesta en marcha rápida y restablecimiento de los parámetros de fábrica en el variador SINAMICS G120.....	27
Figura 2.15 Configuración de parámetros del motor.	28
Figura 2.16 Configuración de la macro de conexión de bus de campo.....	28
Figura 2.17 Ajustes guardados correctamente del variador SINAMICS.....	29
Figura 2.18 Configuración del parámetro 922 correspondiente al telegrama de comunicación.	29
Figura 2.19 Telegrama 20 responsable de la comunicación PROFINET.	29
Figura 2.20 Modo control del variador SINAMICS.	30
Figura 2.21 Identificación del motor en el variador.....	30
Figura 2.22 Variador SINAMICS en modo Automático estableciendo comunicación vía PROFINET....	31

Figura 2.23 Detección del variador SINAMICS y del PLC 1516-3 PN/DP	31
Figura 2.24 Asignación de dirección IP al variador SINAMICS.....	32
Figura 2.25 Selección del archivo GSD del variador SINAMICS G120 C250S- 2PN.....	32
Figura 2.26 Representación del archivo GSD del variador SINAMICS G120 C250S- 2PN versión 4.7....	32
Figura 2.27 Configuración del telegrama 20 en el variador SINAMICS.....	33
Figura 2.28 Asignación de dirección IP al variador SINAMICS.....	33
Figura 2.29 Conexión PROFINET entre el PLC SIEMENS y el variador SINAMICS.	34
Figura 2.30 Asignación del nombre al variador SINAMICS.....	34
Figura 2.31 Búsqueda del variador vía PROFINET.....	35
Figura 2.32 Detección del variador antes configurado.....	35
Figura 2.33 Variador SINAMICS G120 encontrado satisfactoriamente.....	36
Figura 2.34 Bloque función SINA_SPEED_TLG20 [12].....	37
Figura 2.35 Bloque función SINA_SPEED_TLG20 en TIA PORTAL V16.....	38
Figura 2.36 Programa principal en TIA PORTAL V16.....	41
Figura 2.37 Ubicación del bloque SINA_SPEED_TLG20 en TIA PORTAL	41
Figura 2.38 Configuración de parámetros de entrada del bloque SINA_SPEED_TLG20.....	42
Figura 2.39 Placa del motor SIEMENS asíncrono a controlar.....	42
Figura 2.40 Configuración de los parámetros HWIDSTW y HWIDZSW del bloque función.....	43
Figura 2.41 Configuración de parámetros de salida del bloque SINA_SPEED_TLG20.....	43
Figura 3.1 Acceso a la aplicación web desde un dispositivo de un cliente	45
Figura 3.2 Cuadro de agrupación en pantalla HMI [14].	48
Figura 3.3 Objetos dinámicos que representan diferentes estados de válvulas [14].	49
Figura 3.4 Alineación de unidades y datos de ingeniería [14].	50
Figura 3.5 Tendencia en una pantalla HMI.[15]	52
Figura 3.6 Iconos estándar para HMI [14].	52
Figura 3.7 Entorno de desarrollo Visual Studio Code [23].....	57
Figura 3.8 Boceto principal de la interfaz de la aplicación WEB.....	58
Figura 3.9 Archivos htm creados.	60
Figura 3.10 Variables de lectura y escritura relacionados con su respectivo archivo htm.....	60
Figura 3.11 Estructura básica del documento HTML creado.....	62
Figura 3.12 Sección HEAD del documento HTML creado.	62
Figura 3.13 Estructura de la aplicación WEB en filas y columnas.	64
Figura 3.14 Código implementado para la estructura de la aplicación WEB.	64
Figura 3.15 Código implementado para la sección “Encabezado” de la aplicación.	65
Figura 3.16 Sección 2 de la aplicación WEB.....	65
Figura 3.17 Código implementado para la sección 2 de la aplicación.	66
Figura 3.18 Clase btnnavegacion con sus atributos.	66
Figura 3.19 Sección 3 de la aplicación WEB.....	68
Figura 3.20 Elementos de control y su función en las variables de escritura.....	69
Figura 3.21 Código implementado para el botón ARRANQUE.....	70
Figura 3.22 Clase btncontrol con sus atributos.	70
Figura 3.23 Colocación del icono en el botón “ARRANQUE”.	70
Figura 3.24 Script generado en Visual Studio Code.	71
Figura 3.25 Función para “encender” el motor desde el botón ARRANQUE.	71
Figura 3.26 Función implementada para el encendido del motor desde botón ARRANQUE.	72
Figura 3.27 Código implementado para el botón APAGAR.....	72
Figura 3.28 Función para “apagar” el motor desde el botón PARO.	72
Figura 3.29 Código implementado para el elemento Multiselect.....	73
Figura 3.30 Elemento Multiselect para el control de sentido de giro.....	73
Figura 3.31 Función para definir el sentido de giro del motor.	74
Figura 3.32 Código implementado para el botón de control “Ajuste de velocidad”.....	74
Figura 3.33 Código implementado para el elemento “Modal”.....	74
Figura 3.34 Código implementado para configurar los RPM del motor desde un valor ingresado por teclado.	75
Figura 3.35 Código implementado para enviar el valor en RPM desde una barra deslizante.....	75

Figura 3.36 Configuración en TIA PORTAL para recibir valores de tipo flotante (Real), desde la aplicación.	76
Figura 3.37 Script “enviar” para él envío de valores desde los controles de velocidad.	77
Figura 3.38 Código para implementación de imágenes en la aplicación WEB.	78
Figura 3.39 Imágenes; "images/flujoON.png" y "images/flujoOFF.png"	78
Figura 3.40 Clase de CSS “posicion4”	78
Figura 3.41 Código para implementación de imágenes que indican el sentido de giro en la aplicación WEB.	79
Figura 3.42 Sección “Lectura de Valores” de la aplicación web.	80
Figura 3.43 Código implementado para visualizar los datos de la variable “Velocidad”.	81
Figura 3.44 Elementos para mostrar los datos de la variable “Velocidad” y su código de implementación.	81
Figura 3.45 Clase de CSS “parametrosalida”.	81
Figura 3.46 Código implementado para visualizar los datos de la sección “Lectura de valores”	82
Figura 3.47 Función setInterval.	82
Figura 3.48 Función para mostrar valores en el campo “Estado del motor”	83
Figura 3.49 Función para mostrar los valores de los campos; “Velocidad”, “Corriente”, “Torque”, y “Potencia”.	83
Figura 3.50 Función para mostrar los valores de los campos; “Sentido de giro” y “Velocidad Configurada”.	84
Figura 3.51 Sección “Notificaciones”	84
Figura 3.52 Código implementado para implementar la sección “Notificaciones”	85
Figura 3.53 Función para mostrar los valores de los campos de la sección “Notificaciones”	85
Figura 3.54 Código implementado para insertar la imagen de alerta.	86
Figura 3.55 Código implementado para definir las condiciones para mostrar la imagen de la notificación impuesta.	86
Figura 3.56 Código implementado para insertar imágenes que indican el estado del motor.	86
Figura 3.57 Código implementado para insertar imágenes que indican el sentido del motor.	87
Figura 3.58 Código implementado para mostrar la imagen según el estado del motor.	87
Figura 3.59 Código implementado para mostrar las imágenes que indican el sentido de giro del motor	88
Figura 3.60 Código implementado para insertar la imagen y el sonido ante una alarma.	88
Figura 3.61 Código implementado para mostrar la imagen y reproducir el sonido ante una alarma.	89
Figura 3.62 Archivo “grafico3.html”.	90
Figura 3.63 Configuración de la sección head del archivo “grafico3”	90
Figura 3.64 Código implementado para insertar el elemento canvas y los controles de los ejes de la gráfica “Tendencia”	91
Figura 3.65 Scripts necesarios para el funcionamiento de los gráficos.	91
Figura 3.66 Creación del grafico semicircular en función de la velocidad.	92
Figura 3.67 Código implementado para las funciones que crean el grafico “Tendencia”.	92
Figura 4.1 Proceso de almacenamiento de datos mediante la aplicación desarrollada en Laravel.	93
Figura 4.2 Instalación del software XAMPP	97
Figura 4.3 Encendido de los módulos APACHE y MYSQL en el software XAMPP.	97
Figura 4.4 Interfaz phpMyAdmin de MYSQL	98
Figura 4.5 Creación de la base de datos en MYSQL.	98
Figura 4.6 Base de datos creada en MYSQL.	98
Figura 4.7 Creación de la tabla de datos “registros”.	100
Figura 4.8 Configuración de las propiedades de los campos de la tabla registros.	100
Figura 4.9 Configuraciones finales en la tabla de la base de datos.	101
Figura 4.10 Propiedades de la tabla “registros”.	101
Figura 4.11 Directorio de una aplicación creada en Laravel [37].	102
Figura 4.12 Selección de la versión de PHP para el software Composer.	103
Figura 4.13 Ubicación en la carpeta “htdocs”.	103
Figura 4.14 Instalación del programa Laravel.	103
Figura 4.15 Creación de un proyecto nuevo en Laravel.	103
Figura 4.16 Proyecto de Laravel “monitoreo”.	104

Figura 4.17 Directorios del proyecto de Laravel “monitoreo”.....	104
Figura 4.18 Activación del servicio del proyecto monitoreo.....	104
Figura 4.19 Acceso al proyecto de Laravel.....	105
Figura 4.20 Archivo “hosts” de Windows.....	105
Figura 4.21 Dominio establecido para acceso a la aplicación de Laravel.....	105
Figura 4.22 Configuración del host virtual para el acceso a la aplicación web.....	106
Figura 4.23 Acceso a la aplicación web usando un host virtual.....	106
Figura 4.24 Conexión entre la base de datos de MYSQL y Laravel.....	107
Figura 4.25 Migraciones realizadas en el proyecto de Laravel.....	108
Figura 4.26 Tablas generadas por defecto al ejecutar las migraciones.....	108
Figura 4.27 Tabla de migración “registros”.....	109
Figura 4.28 Funciones up y down de la clase “registros”.....	109
Figura 4.29 Creación del modelo Registro en Laravel.....	110
Figura 4.30 Modelo “Registro”.....	110
Figura 4.31 Clase Registro.....	111
Figura 4.32 Patrón de funcionamiento de una aplicación en Laravel [45].....	111
Figura 4.33 Creación del controlador ApiRegistroController en Laravel.....	112
Figura 4.34 Controlador ApiRegistroController.....	113
Figura 4.35 Ruta “post” y “get” del controlador ApiRegistroController.....	113
Figura 4.36 Proceso para el almacenamiento de datos del proceso usando la aplicación de Laravel.....	114
Figura 4.37 Función implementada para él envío de valores a la aplicación diseñada en Laravel.....	115
Figura 4.38 Relación de cada variable de proceso con su respectivo campo en la base de datos.....	115
Figura 4.39 Controlador RegistroController.....	116
Figura 4.40 Funciones grafico () y datos ().....	117
Figura 4.41 Rutas get del controlador RegistroController.....	117
Figura 4.42 Funciones del Controlador ReporteController.....	118
Figura 4.43 Función create del controlador ReporteController.....	119
Figura 4.44 Configuraciones de las rutas del controlador ReporteController.....	119
Figura 4.45 Función de retorno del elemento vista “index” desde el controlador RegistroController.....	120
Figura 4.46 Carpeta “registro” en la sección “vistas”.....	120
Figura 4.47 Elemento vista “index.blade.php”.....	121
Figura 4.48 Elemento vista “reporte.blade.php”.....	122
Figura 4.49 Elemento vista “visual.blade.php”.....	122
Figura 5.1 Banco de trabajo para prácticas de automatización.....	123
Figura 5.2 Conexiones del proceso de control.....	124
Figura 5.3 Dirección IP del dispositivo cliente para el acceso al servidor WEB.....	124
Figura 5.4 Sección “Mostrar redes disponibles” en Windows 10.....	125
Figura 5.5 Sección “Centro de redes y recursos compartidos”.....	125
Figura 5.6 Propiedades de Ethernet de la interfaz del dispositivo “Cliente”.....	126
Figura 5.7 Configuración de la dirección IP de la interfaz de red del equipo cliente.....	126
Figura 5.8 Configuración de la página de usuario del servidor WEB del PLC utilizado.....	127
Figura 5.9 Bloques generados en TIA PORTAL para la comunicación del PLC mediante el servidor WEB.....	128
Figura 5.10 Carga del programa en el PLC utilizado.....	128
Figura 5.11 Carga del programa en el PLC utilizado.....	129
Figura 5.12 Ventana “Cargar en dispositivo”.....	129
Figura 5.13 Etapa final del proceso de carga.....	130
Figura 5.14 Panel de control XAMPP.....	130
Figura 5.15 Acceso al servidor web desde el buscador Google Chrome.....	131
Figura 5.16 Portada de inicio del servidor web.....	131
Figura 5.17 Página inicial del servidor WEB.....	132
Figura 5.18 Página de usuario implementada en el servidor WEB.....	132
Figura 5.19 Aplicación WEB implementada en el servidor web del PLC SIEMENS.....	133
Figura 5.20 Estado inicial de la aplicación WEB.....	133
Figura 5.21 Pasos para la muestra en marcha del motor desde la aplicación WEB.....	134

Figura 5.22 Selección del sentido del motor.....	135
Figura 5.23 Proceso de control sin la presencia de errores.....	135
Figura 5.24 Uso de imágenes interactivas en la aplicación web.....	136
Figura 5.25 Controles de ajuste de velocidad del motor.....	136
Figura 5.26 Ajuste de velocidad mediante un valor ingresado.....	137
Figura 5.27 Caracteres no permitidos en la configuración de ajuste de velocidad.....	137
Figura 5.28 Valores no permitidos en la configuración de ajuste de velocidad.....	137
Figura 5.29 Estado de variables de entrada en la aplicación WEB.....	138
Figura 5.30 Encendido del motor.....	139
Figura 5.31 Parámetros de salida visualizados en el panel operador IOP.....	139
Figura 5.32 Control de velocidad mediante la barra deslizante.....	140
Figura 5.33 Advertencia gestionada por la aplicación WEB.....	140
Figura 5.34 Desconexión del PLC y el variador de frecuencia.....	141
Figura 5.35 Alarmas gestionadas por la aplicación WEB.....	141
Figura 5.36 Corrección de las fallas de conexión entre el PLC SIEMENS y el variador de frecuencia.....	142
Figura 5.37 Botón “Gráficos” de la aplicación WEB.....	142
Figura 5.38 “Gráficas” en función de la variable “velocidad actual” en la aplicación WEB.....	143
Figura 5.39 Control de la velocidad del motor en 475 RPM desde la barra deslizante.....	143
Figura 5.40 Grafica tendencia cuando el motor alcanza los 475 RPM.....	144
Figura 5.41 Visualización de las variables de entrada y salida del bloque función desde TIA PORTAL V16.....	144
Figura 5.42 Grafica “Tendencia” cuando el motor ha alcanzado 875 RPM.....	145
Figura 5.43 Interfaz de acceso a los datos almacenados en el servidor local del equipo.....	145
Figura 5.44 Datos almacenados desde la aplicación WEB.....	146
Figura 5.45 Interfaz “Reportes”.....	146
Figura 5.46 Configuración de fecha y hora de inicio y fin para descarga y/o visualización de datos del proceso.....	147
Figura 5.47 Selección de tipo de “Reporte”.....	147
Figura 5.48 Selección de Reporte “Visual”.....	147
Figura 5.49 Reporte Visual con cada una de las variables del proceso.....	148
Figura 5.50 Datos totales adquiridos en el “Reporte Visual”.....	148
Figura 5.51 Reporte “Registros.xlsx” descargado desde la interfaz de Laravel.....	149
Figura 5.52 Archivo “Registros.xlsx” abierto en el entorno Microsoft Excel.....	149

ÍNDICE DE TABLAS

Tabla 2.1 Compatibilidad entre unidades de control y módulos de potencia (Tecnología SIEMENS)	20
Tabla 2.2 Estructura del telegrama estándar 20 [12].....	37
Tabla 2.3 Significado de palabras y valores del telegrama estándar 20.....	38
Tabla 2.4 Parámetros de entrada del Bloque SINA_SPEED_TLG20 [12].....	38
Tabla 2.5 Bits de la palabra de mando STW1.....	39
Tabla 2.6 Parámetros de salida del Bloque SINA_SPEED_TLG20 [12]	40
Tabla 2.7 Parámetros de entrada y salida a monitorearse con el uso del Servidor WEB del PLC.....	44
Tabla 3.1 Colores designados para diferentes elementos de un HMI [14].....	47
Tabla 3.2 Etiquetas comunes en HTML.....	55
Tabla 3.3 Propiedades comunes de CSS para elementos HTML.....	55
Tabla 3.4 Elementos de cada una de las secciones de la aplicación WEB.....	58
Tabla 3.5 Comandos AWP insertados en cada archivo htm.	61
Tabla 3.6 Función de cada sentencia implementada en la sección HEAD.	62
Tabla 3.7 Elementos de la sección 1 (Encabezado) y su respectivo código de implementación.	65
Tabla 3.8 Elementos de la sección 2 y su respectivo código de implementación.	67
Tabla 3.9 Elementos de la sección 3 y su respectiva función en la aplicación WEB.....	68
Tabla 4.1 Campos de la tabla de estructuración de la base de datos.	99
Tabla 4.2 Funciones del Controlador RegistroController.	116

AGRADECIMIENTOS

Agradezco, en primer lugar, a mis padres quienes me proporcionaron todo su apoyo y me acompañaron en el desarrollo de esta meta personal. A mis hermanas quienes me brindaron su ayuda en toda mi etapa universitaria. Por último, agradezco a mi tutor, Ing. Julio Zambrano, quien me brindo su confianza, paciencia, conocimiento y apoyo para el desarrollo de esta etapa universitaria.

Luis Gonzalo Viñamagua Paredes

DEDICATORIA

Dedico este proyecto técnico con gran fervor a todos los miembros de mi familia Gonzalo, Diana, Tania quienes son parte de esta meta y me brindaron su acompañamiento en los malos y buenos momentos. Dedico de manera especial este trabajo, a mi madre, Luz María, quien ha sido la pieza fundamental de mi vida y que me proporciono todas sus fuerzas a acompañarme en mis estudios. Dedico este triunfo alcanzado a todos los profesores de la universidad que me brindaron su conocimiento, confianza y sus palabras de apoyo para poder culminar esta meta.

Luis Gonzalo Viñamagua Paredes

GLOSARIO

HTML: Lenguaje de Marcas de Hipertexto- *Hypertext Markup Language*

PLC: Controlador lógico programable.

HTTP: ‘protocolo de transferencia de hipertextos’-*Hypertext Transfer Protocol*,

LAN: Red de Área Local - Local Area Network.

RESUMEN

El presente proyecto técnico hace un análisis sobre la utilización de las diferentes herramientas proporcionadas por los lenguajes de HTML y JavaScript para la gestión de variables de procesos usando autómatas. En base a los lenguajes antes mencionados, se creó una aplicación web con la que se podrá controlar las variables que ponen en funcionamiento un sistema de velocidad. Mediante el entorno de desarrollo Visual Studio Code se programó la aplicación web mientras que con el programa TIA Portal V16, se configuro el servidor web que permite el control del sistema de velocidad conformado por un autómata y un variador de frecuencia. Por medio de la creación de la aplicación web se pudo controlar las diferentes variables del proceso controlado, permitiendo al usuario visualizar cada uno de los datos proporcionado por un variador de frecuencia. Mientras que el lenguaje de JavaScript permitió crear diferentes graficas de apoyo para informar al operador de la situación actual del proceso. Además, mediante herramientas como MYSQL y Laravel, se gestionó la recepción, almacenamiento y descarga de datos recopilados para un análisis posterior del operario.

INTRODUCCIÓN

El monitoreo de datos permite verificar el funcionamiento correcto de un proceso o sistema industrial, permitiendo identificar de manera oportuna fallas presentes o que puedan ocurrir. Además, dada las nuevas tecnologías, se requiere de nuevas formas o métodos más sofisticadas que permitan recopilar, manipular y gestionar información de procesos a gran escala con la finalidad de incrementar la productividad de la maquinaria. Por ello, equipos destinados a la automatización de procesos, incorporan nuevas alternativas de supervisión y administración de equipos industriales.

Autómatas o PLCs de la firma Siemens incorporan servicios denominado “Servidor web”, el cual permite el fácil acceso y control de procesos industriales mediante la implementación de páginas web. Con esto se proporciona una nueva alternativa que proporciona ventajas como la de prescindir de hardware y software para supervisión, facilidad de implementación para supervisar un proceso de control y la rapidez de acceso a información del proceso industrial por medio de tecnologías como ethernet.

Por otro lado, al usar páginas web o documentos web para supervisar procesos industriales, se puede aprovechar todas las ventajas que proporciona lenguajes como HTML y JavaScript, para creación de aplicaciones web. Lenguaje como html permiten insertar diferentes controles o formularios para gestionar el control de variables del PLC. Además, por medio del lenguaje de complemento CSS, se puede crear interfaces muy ilustrativas que se pueden adaptar a cualquier dispositivo desde el que se acceda al servidor web. Por último, con JavaScript se pueden crear una gran variedad de gráficos que pueden representar los datos provenientes de variables de procesos tales como magnitudes de presión, velocidad, corriente, torque, etc. Con lo antes mencionado, el proyecto técnico propone el control de un proceso industrial simple por medio de la creación de una aplicación web, que será diseñada por medio de lenguajes de HTML y JavaScript. Para ello se abordará varios pasos que serán detallados en los seis capítulos de esta monografía.

El primer capítulo presenta cada una de las características que posee el servidor web integrado en los autómatas SIEMENS. Se describe las configuraciones previas para poner en marcha el servidor web, así como las páginas web estándar que posibilitan acceder a información del proceso controlado por el PLC.

El segundo capítulo presenta el proceso a ser controlado, en el cual se propone una arquitectura de un sistema de velocidad. Luego se configurará el control del proceso por medio del programa TIA PORTAL V16, ya que es necesario configurar el bloque de función que controla el sistema de velocidad.

Para el tercer capítulo, se empieza con el análisis de los elementos o formularios que posee HTML para el control de las variables del proceso mediante el servidor web del PLC SIEMENS. Aquí, se definirá las secciones de la aplicación web que se creará con la finalidad de gestionar los datos del proceso de control implementado. Además, con los lenguajes de CSS y JavaScript, se programan las funciones necesarias para recolección y visualización de datos en la aplicación web. Por último, se muestra como mediante librerías elaboradas en JavaScript permiten crear diferentes tipos de gráficos que pueden representar los datos de un proceso de control.

En el cuarto capítulo, por medio del entorno de trabajo de Laravel, se crea una aplicación web que se complementa con la aplicación antes creada en html. Esta aplicación permite gestionar el almacenamiento de los datos proveniente de las variables del proceso. Además, por medio de esta aplicación se gestionará el acceso y visualización a cada uno de los datos almacenados en una base de MYSQL. Finalmente, como una de las implementaciones finales de la aplicación creada de Laravel, se implementó la función de descarga y visualización de datos donde el usuario puede gestionar un rango de datos definido por fecha y hora.

En el capítulo quinto se muestra cada uno de las pruebas elaboradas para corroborar el funcionamiento de la aplicación creada en HTML y JavaScript, donde se pudo obtener con gran rapidez los datos del proceso, así como el control de las diferentes variables.

Finalmente, en el capítulo sexto, se encontrarán las conclusiones y recomendaciones que se obtuvieron con respecto al uso de las diferentes herramientas de HTML y JavaScript, para la gestión de datos y variables mediante el uso de autómatas o PLCs. Se muestran algunas recomendaciones con respecto al uso del servidor web, y aspectos a considerar en la programación de la aplicación web así como las limitaciones del servidor web y trabajos futuros a realizarse.

ANTECEDENTES DEL PROBLEMA DE ESTUDIO

Los nuevos avances de la tecnología traen consigo nuevos cambios en la instrumentación y control de procesos, donde arquitecturas que poseen interfaces Ethernet han pasado a ser las predominantes debido a su alta velocidad y grandes prestaciones [1].

Equipos como PLC (Controladores Lógicos Programables) son ampliamente usados en el control industrial, en función de aspectos positivos como: presupuestos económicos, facilidad de instalación, robustez y flexibilidad en aplicaciones. Por lo tanto, si un PLC se integra con un servidor web garantiza un flujo oportuno de información, dado que posee un acceso directo al autómata. Además, dado a los interfaces Ethernet que poseen los autómatas se puede monitorear y compartir información para una toma de decisiones rápida y oportuna.

El monitoreo de datos de procesos industriales que se basan en Ethernet/Internet ofrece al usuario grandes facilidades para acceder a los datos de un proceso desde localizaciones remotas que pueden estar ubicadas en cualquier lugar del planeta, siempre y cuando se tenga acceso a internet. Otro aspecto importante es que el desarrollador no debe invertir dinero en la adquisición de licencias y software propietario dado que existe un gran lote de herramientas para diseñar en la web y muchas de estas son de uso libre.

Con esto el usuario puede acceder a estos datos para su análisis correspondiente en cualquier momento y lugar los 365 días, las 24 horas del día con la ayuda de una conexión a Internet. Dichos datos pueden ser registrados en tarjetas de tipo SD, o se pueden descargar a través de una red propia Ethernet o a través de Internet.

Para formar un centro de control y monitoreo, este estará formado por varios equipos como un PLC de control, PC del operador, equipos y sensores de procesos industriales, entre otros. La interconexión del PC del operador hacia el PLC de control, se lo realiza a través de un bus industrial o mediante protocolos tipo LAN y TCP/IP.

En este contexto se plantea este proyecto de investigación, mediante el cual se analizar las herramientas HTML y JavaScript para el desarrollo de páginas web aplicadas a la gestión de datos y variables dentro de sistemas de automatización basados en PLCs. Específicamente este proyecto se basará en el uso de autómatas S7-1200/1500 de la firma SIEMENS.

JUSTIFICACIÓN (IMPORTANCIA Y ALCANCES)

El monitoreo de datos a través de un servidor puede resultar de gran utilidad dentro de los procesos industriales. Una de las principales ventajas es el acceso remoto no solo a los datos del proceso sino también a los recursos del CPU, por otra parte, este tipo de sistema de monitoreo permite abaratlar costos dado que no es necesario adquirir programas informáticos especializados [2].

Entre otras ventajas se destaca las funciones de seguridad que se pueden incorporar, por lo que se podrá acceder al servidor web por medio de un protocolo de transmisión seguro conocido como "HTTPS". También se podrá acceder a la configuración del sistema, por medio de los distintos usuarios agregados a una lista determinada.

Con las propiedades mostradas anteriormente, es posible realizar aplicaciones que se visualicen en un monitor, el cual se comunica directamente con un PLC o CPU para observar el proceso industrial que este último tiene como encargo. En complemento, bajo este concepto también es posible utilizar cualquier dispositivo móvil como un teléfono o una tableta para poder acceder a los datos de un proceso.

De acuerdo al diseño de la página web la interacción entre el usuario y el proceso podrá ser de lectura y escritura. Por lo tanto, el usuario podrá crear dicha página web combinando los distintos lenguajes de programación como HTML, JavaScript y CSS. Como una ventaja adicional, las páginas web permiten la posibilidad de acceder a información independientemente del sistema operativo navegador que se esté usando.

La propuesta del proyecto de titulación es crear una página web dinámica, la cual será guardada en el PLC. Desde una PC remota se podrá acceder al servidor web, que contiene dicha página a la que accederemos desde un buscador y desde ahí gestionar al monitoreo de variables y datos del proceso. Además, mediante de entornos de trabajo como Laravel, se puede implementar una aplicación que pueda gestionar el almacenamiento y descarga de datos que se alojaran en una base de datos de MYSQL. Por consiguiente, se plantea una aplicación que proporciona los datos para un análisis posterior de datos, con la finalidad de prevenir el mal funcionamiento de los equipos de un proceso, y la posibilidad que estos datos puedan ser compartidos en un servicio de la nube mediante implementaciones futuras.

Se busca dar una nueva alternativa de HMI (human-machine interface), mediante la creación de una página web, implementada en un servidor de un PLC S7-1200/1500 de SIEMENS. Dependiendo de la disponibilidad en laboratorio se utilizarán procesos didácticos para desarrollar aplicaciones.

OBJETIVOS

OBJETIVO GENERAL

Analizar el conjunto de herramientas HTML y JavaScript para el desarrollo de páginas web aplicadas a la gestión de datos y variables dentro de sistemas de automatización basados en PLCs SIMATIC de la firma SIEMENS.

OBJETIVOS ESPECÍFICOS

- Analizar las características y modo de operación de los servidores web incrustados dentro los PLCs SIMATIC S7-1500 de Siemens.
- Analizar la flexibilidad, versatilidad y facilidad de uso de herramientas HTML y JavaScript en áreas de desarrollo de aplicaciones de monitoreo y control basadas en el servidor web de un PLC SIMATIC S7-1500 de Siemens.
- Desarrollar una base de datos para almacenar y desplegar variables de procesos industriales visualizadas a través de la aplicación web.
- Desarrollar una aplicación de automatización utilizando herramientas didácticas de laboratorio para controlar y monitorizar sus variables mediante el uso del servidor web del PLC.

CAPÍTULO 1

Introducción al servidor web integrado en autómatas de la marca Siemens

En este capítulo se aborda las características y prestaciones que posee el servidor web de la marca SIEMENS. Se hace una breve descripción de cada una de las páginas web estándar que vienen incorporadas en el servidor y sus diferentes utilidades. Además, se presenta el proceso de configuración e implementación para establecer el servicio del servidor web en los CPU o autómatas de la firma SIEMENS.

1.1 Características básicas del servidor web

1.1.1 Prestaciones del servidor web

El servidor web integrado en los PLCs de la firma SIEMENS, permite a los usuarios autorizados monitorear y administrar la CPU o el autómata a través de una red creada. A su vez, esto permite interactuar con un proceso industrial sin la necesidad de recurrir a programas propietarios. En adición, se debe destacar que, gracias al servidor web es posible acceder al PLC y monitorizar un proceso desde un lugar remoto [3]. Por lo tanto, un servidor web en el campo de la automatización, permite que datos e información del autómata sean presentados a través de páginas web creadas bajo formatos HTML o XML, accediendo a estas con ayuda de un navegador de internet [4].

Hoy en día, gracias a los avances de las comunicaciones industriales, a través de Ethernet, es posible acceder directamente a un proceso por medio de una intranet. Por consiguiente, es factible y razonable usar estándares ya existentes, como tecnología http y lenguajes relacionados como HTML y JavaScript para acceder a la información del proceso, mediante páginas o aplicaciones crudas en dichas plataformas. En la Figura 1.1, se muestra una arquitectura típica de un servidor web el cual se incorpora a un proceso a través de una comunicación Profinet. Además, se visualiza de una manera rápida el proceso de implementación de la página web en el servidor web.

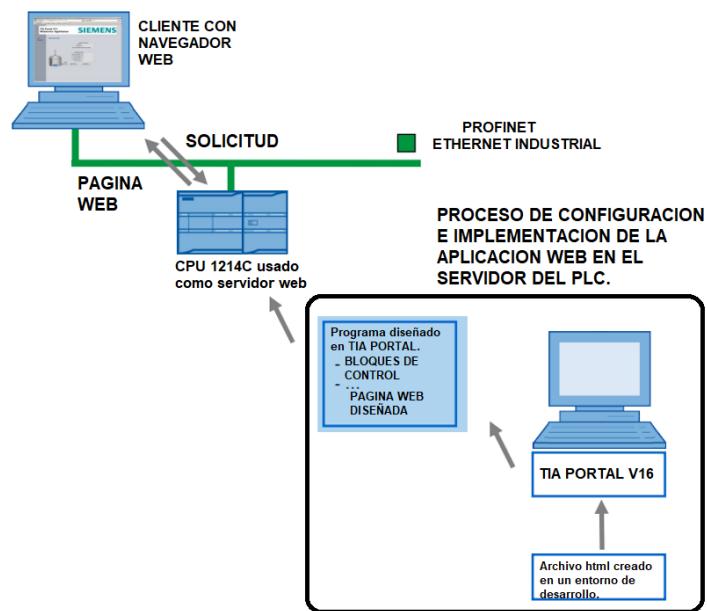


Figura 1.1 Esquema de un servidor WEB vía comunicación PROFINET.

Con la familia de controladores SIMATIC s7-1200 y s7-1500, es posible acceder de manera remota a las variables de automatización a través de una página web definida por el usuario. Además, es posible acceder a información adicional por medio de páginas web estándar ya integradas, algunas de ellas son: página de inicio, contenido de búfer de diagnóstico, información del módulo, Información sobre la comunicación,

estado de variables, etc [3].

A continuación, se hace una descripción de las funcionalidades del servidor web de autómatas SIEMENS, configuraciones y recursos como las diferentes páginas estándar incorporadas en el servidor web. Esta información es obtenida del manual “SIMATIC S7-1500, Servidor web Manual de funciones” [3].

1.1.2 Ventajas respecto al uso de un servidor web

Al tener acceso a través de un navegador web, se pueden mostrar los datos del proceso y hasta en cierta medida se puede controlar el proceso desde cualquier ordenador sin necesidad de recurrir softwares adicionales.

Como ventaja también se destaca el hecho de poder aprovechar la infraestructura de red de una planta sin la necesidad de usar o emplear un hardware adicional.

En cuanto a las funciones de seguridad, el servidor web permite el acceso a un usuario por medio del protocolo seguro de transferencia "https", además de permitir el acceso solamente a usuarios previamente autorizados de una lista configurada.

1.2 Proceso de implementación del servidor web

1.2.1 Configuración del servidor WEB

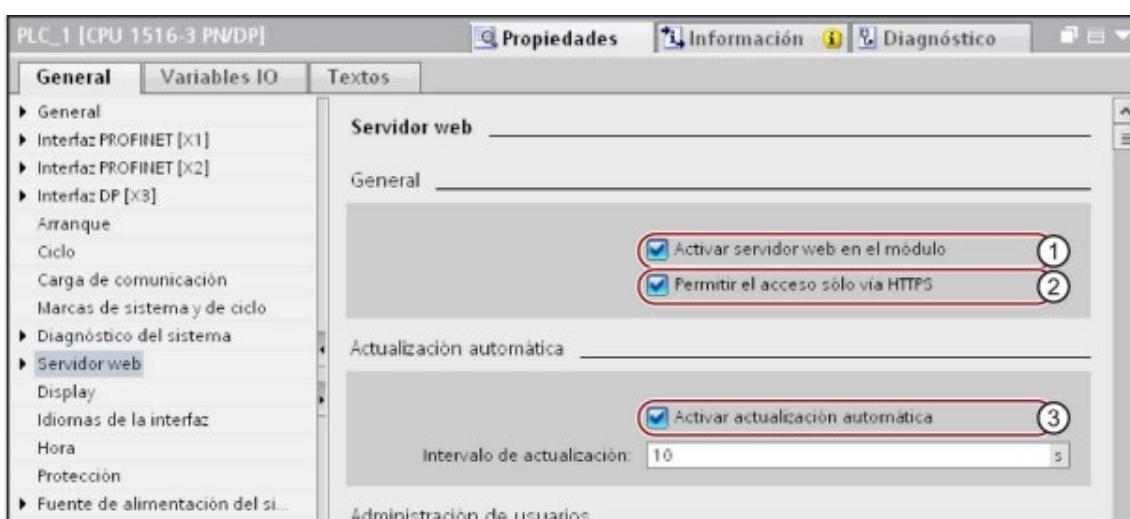


Figura 1.2 Ajustes iniciales del servidor web en TIA PORTAL [3].

Como se muestra en la figura 1.2, una vez que un autómata ha sido incorporado a un proyecto, como primer paso se debe activar el servidor web. Para ello, en TIA PORTAL, en la sección de “Dispositivos y redes” y en la venta de inspección de la ficha “General”, se debe dar clic en la casilla de “Activar servidor web en el módulo”.

Como siguiente paso y si se desea acceder a las páginas web mediante una transferencia “segura”, se debe activar la casilla “Permitir el acceso solo vía HTTPS”. Por último, se debe activar la casilla de Actualización automática, de esta manera se podrán actualizar las páginas estándar del servidor.

A continuación, se deberá configurar la administración de usuarios. En la figura 1.3, se muestran tres apartados para configurar los administradores del servidor web. En la sección “Nombre” se coloca el usuario con su correspondiente nombre. En la sección de “Nivel de Acceso”, se otorgan los derechos de acceso teniendo como opciones: Mínimo (no se otorgan permisos para acceder a las funciones del servidor) o Administrativo (se otorgan los permisos para el control total del PLC). Finalmente, en el apartado “Contraseña”, se deberá establecer una clave de seguridad [3].

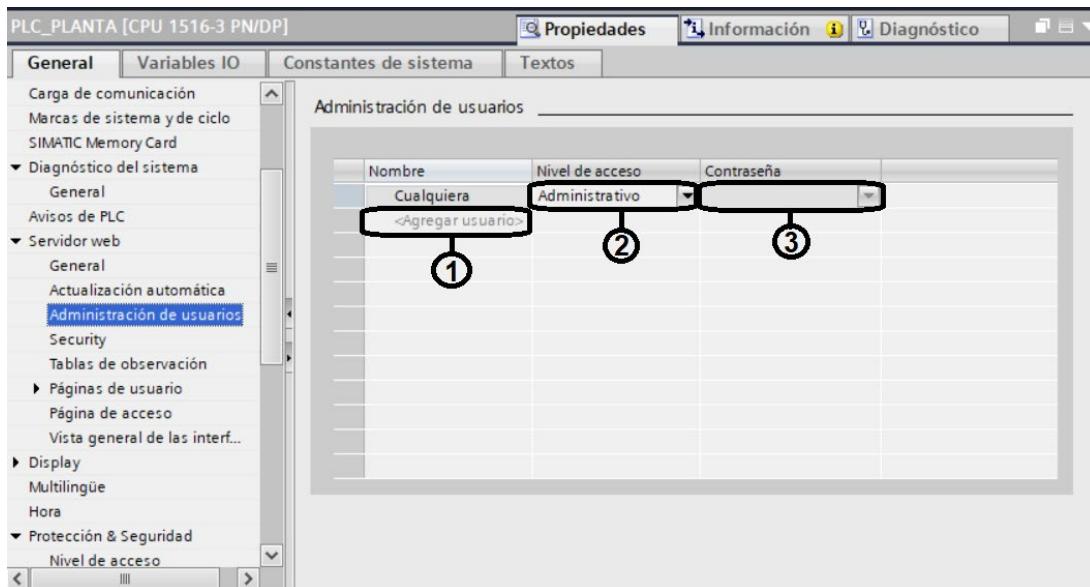


Figura 1.3 Configuración de la lista de usuarios del servidor.

En caso de que no se agregue un usuario predeterminado, y un usuario acceda al servidor web, este será denominado usuario “Cualquiera”, el cual tendrán permisos de acceso mínimo. Con ello, el usuario podrá tener lectura de las páginas “Intro” y de inicio.

1.2.2 Acceso al servidor web a la CPU SIMATIC desde una PC o dispositivos HMI

Para acceder al servidor web, se procede con los siguientes pasos:

1. En STEP 7 o TIA PORTAL, cargue el proyecto con las instrucciones del proceso a ejecutarse en el PLC o CPU, en la cual se encuentra activado el servidor web.
2. Establezca la conexión entre el PLC y la PC de usuario, por medio de enlaces inalámbricos como WLAN bajo estándares como SCALANCE W788-1RR o SCALANCE W784-1, etc. En caso de no optar por la primera alternativa, con el uso de un módulo de comunicaciones se podría comunicar con el dispositivo de visualización a través de una interfaz PROFINET.
3. Abra el navegador web, de su preferencia, de los cuales se sugiere programas como: Firefox, Google Chrome o Safari.
4. En la sección de dirección del navegador, introducir la dirección IP del PLC, con la que se ha configurado al servidor WEB, introduciendo la IP como, por ejemplo; <http://192.168.0.12>. Luego de insertar la dirección, se desplegará la página de introducción o “INTRO”, desde ahí el usuario podrá acceder a las diferentes páginas web estándar y por último a la página definida por el usuario que es de nuestro interés.

1.3 Páginas del servidor web

Una vez que se ha podido acceder al servidor web, aparecerá una página denominada “INTRO”, la cual se muestra en la figura 1.4:



Figura 1.4 Pagina "Intro" inicial del servidor web de un PLC S7-1500.

Una vez mostrada la página de introducción, al presionar ENTER se desplegarán una serie de páginas las cuales son las siguientes:

- Página de Inicio
- Login
- Diagnóstico
- Búfer de diagnóstico
- Información del Modulo
- Avisos
- Comunicación
- Topología
- Estado de variables
- Tablas de usuario
- *Páginas de usuario*

A continuación, se describen cada una de las páginas web estándar ya integradas en el servidor web por defecto hasta llegar a la sección de “páginas de usuario” que será la sección de interés del proyecto técnico.

1.3.1 Página de Inicio

Para acceder a las funciones de las distintas páginas web del servidor es necesario iniciar sesión mediante un usuario y su contraseña que han sido definidos durante la configuración del servidor en STEP 7. Por lo tanto, si se inicia sesión, con los permisos autorizados y configurados, se desplegará la interfaz visualizada en la figura 1.5:

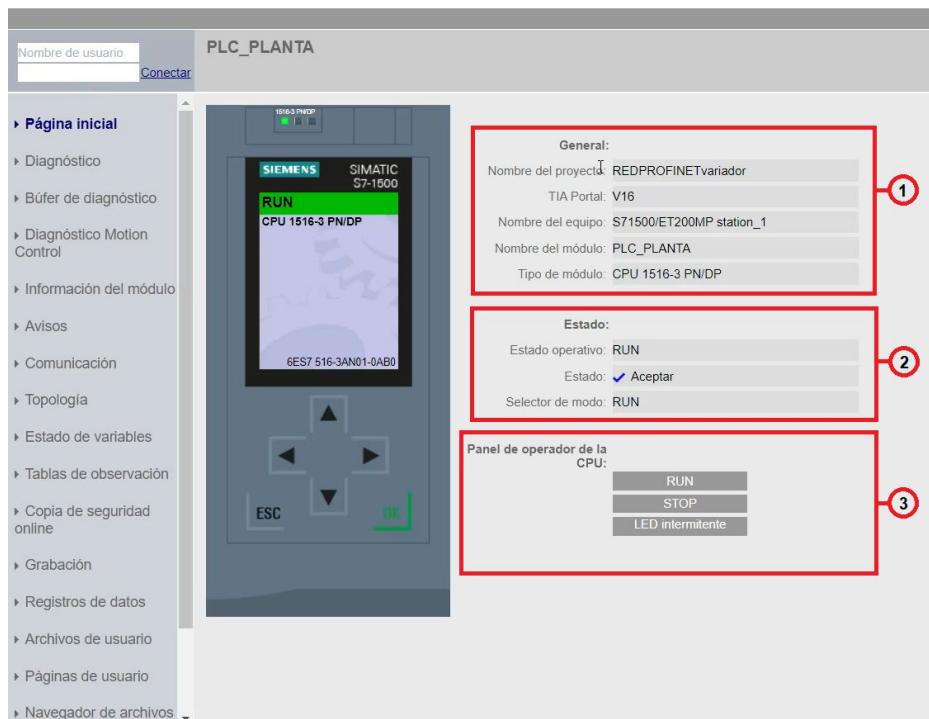


Figura 1.5 Página inicial posterior al iniciar sesión en el servidor web.

En la figura 1.5, se muestra la CPU o el PLC, indicando su estado actual con ayuda de un led, el cual estará encendido o apagado. En la sección 1), aparece información sobre el PLC que contiene dicho servidor web. En 2) muestra el estado de la CPU, en el momento en el que se está accediendo desde el navegador. En 3), panel de mando, podremos cambiar el estado del PLC (arranque o parada) o simplemente hacer parpadear los leds del PLC.

1.3.2 Diagnóstico

En esta página, se encontrará información de las fichas de identificación y memoria, como lo indica la figura 1.6. En la ficha “identificación”, se mostrará información del identificador de instalación, situación y número de serie. En “referencia”, se mostrará un código de ayuda para tener información del hardware que se maneja. Finalmente, en el campo “versión”, se mostrará las versiones de hardware, firmware, y bootloader.

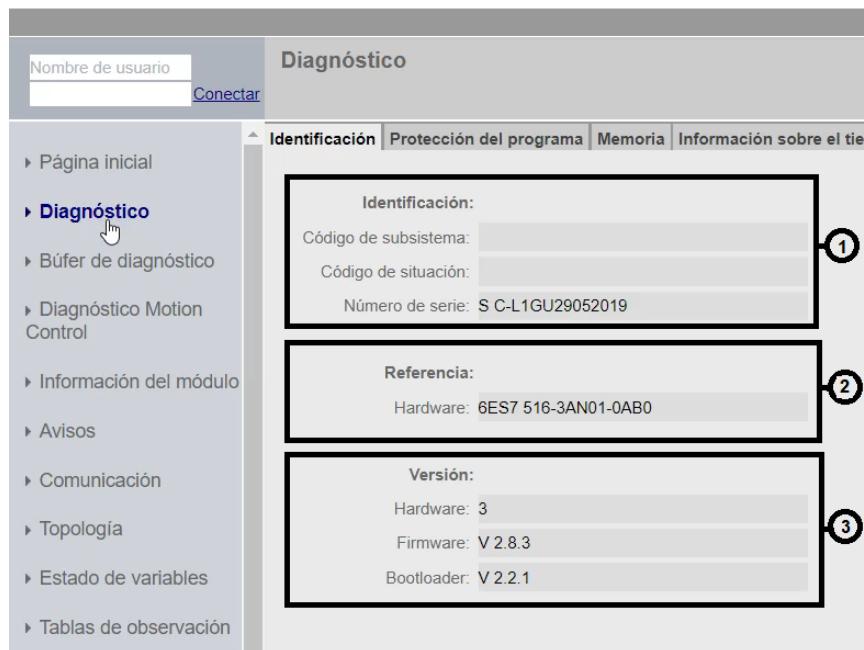


Figura 1.6 Página de diagnóstico en la sección de "Identificación".

En la ficha de “Memoria”, se muestran valores actuales de la memoria ocupada del total de memoria disponible (véase la figura 1.7).

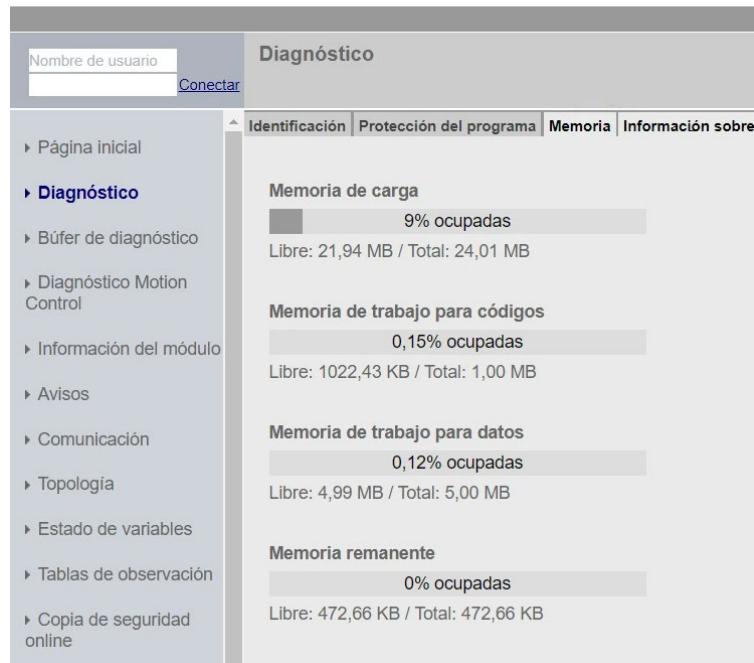


Figura 1.7 Ficha "Memoria" de la página de Diagnóstico.

1.3.3 Búfer de diagnóstico

En esta sección, se muestra 3 parámetros de importancia como “Búfer de diagnóstico entradas 1-50”, “Evento” y “detalles”. La figura 1.8 muestra la sección antes descrita.

En 1) se muestran un número determinado de avisos, esto en base al número de entradas que la CPU admite, teniendo un intervalo de 50 entradas como máximo. En 2) contiene los eventos de diagnóstico con fecha y hora. En 3) muestra información más detallada dependiendo del evento escogido en 2).

SIEMENS S71500/ET200MP station_1/PLC_PLANTA

07:20:59 21.03.2012 Hora local del F

Nombre de usuario	Búfer de diagnóstico				
	Entradas 1 a 50 de búfer de diagnóstico				
	Número	Hora	Fecha	Estado	Evento
» Página inicial	1	07:03:25 634	21.03.2012	Evento saliente	Fallo de un dispositivo IO
» Diagnóstico	2	07:02:40 604	21.03.2012	Evento entrante	Fallo de un dispositivo IO - Dispositivo IO no encontrado
» Búfer de diagnóstico	3	07:02:40 604	21.03.2012	Evento saliente	Fallo de un dispositivo IO - Tiempo de vigilancia excedido
» Diagnóstico Motion Control	4	07:02:38 100	21.03.2012	Evento entrante	Fallo de un dispositivo IO - Tiempo de vigilancia excedido
» Información del módulo	5	07:02:38 098	21.03.2012	Evento saliente	Fallo de datos E/S en el componente de hardware
» Avisos	6	07:02:38 091	21.03.2012	Evento entrante	Fallo de datos E/S en el componente de hardware
» Comunicación	7	06:55:25 667	21.03.2012	Evento entrante	Estado operativo siguiente tras transición - La CPU cambia del estado ARRANQUE a RUN
» Topología	8	06:55:02 823	21.03.2012	Evento entrante	Estado operativo siguiente tras transición - La CPU cambia del estado STOP a ARRANQUE
» Estado de variables	9	06:55:02 723	21.03.2012	Evento entrante	Estado operativo siguiente tras transición - La CPU cambia del estado STOP (initialización) a STOP
» Tablas de observación	10	06:54:58 811	21.03.2012	Evento entrante	Conexión (POWER ON) - La CPU cambia del estado DESCONEXIÓN a STOP (initialización)
» Copia de seguridad online	11	06:53:42 990	21.03.2012	Evento entrante	Desconexión (POWER OFF) - La CPU cambia del estado RUN a DESCONEXIÓN
» Grabación	12	06:19:26 228	21.03.2012	Evento saliente	Fallo de un dispositivo IO
» Registros de datos	13	06:19:01 247	21.03.2012	Evento entrante	Fallo de un dispositivo IO - Dispositivo IO no encontrado
» Archivos de usuario	14	06:19:01 247	21.03.2012	Evento saliente	Fallo de un dispositivo IO - Tiempo de vigilancia excedido
» Páginas de usuario	15	06:18:58 743	21.03.2012	Evento entrante	Fallo de un dispositivo IO - Tiempo de vigilancia excedido
	16	05:48:58 592	21.03.2012	Evento entrante	Estado operativo siguiente tras transición - La CPU cambia del estado ARRANQUE a RUN
	17	05:48:33 254	21.03.2012	Evento entrante	Estado operativo siguiente tras transición - La CPU cambia del estado STOP a ARRANQUE
	18	05:48:33 154	21.03.2012	Evento entrante	Estado operativo siguiente tras transición - La CPU cambia del estado STOP (initialización) a STOP
	19	05:48:29 247	21.03.2012	Evento entrante	Conexión (POWER ON) - La CPU cambia del estado DESCONEXIÓN a STOP (initialización)
	20	09:32:20 605	20.03.2012	Evento entrante	Desconexión (POWER OFF) - La CPU cambia del estado RUN a DESCONEXIÓN
	21	09:10:28 304	20.03.2012	Evento saliente	Fallo de un dispositivo IO
	22	09:10:28 305	20.03.2012	Evento saliente	Fallo de un dispositivo IO - Dispositivo IO no encontrado
	Detalles:				
	Error: Fallo de un dispositivo IO - SINAMICS-G120SV-PN				
	Evento saliente				

Figura 1.8 Búfer de diagnóstico.

1.3.4 Información del modulo

En esta sección, el estado de la estación (CPU), se visualizará a través de símbolos y comentarios. Por ello es necesario entender cada uno de los símbolos y su respectivo significado mediante el uso de la figura 1.9.

Símbolo	Color del símbolo	Significado
✓ verde	verde	El componente funciona correctamente
✗ gris	gris	Esclavos PROFIBUS o dispositivos PROFINET desactivados.
?	gris	Estado no determinable <ul style="list-style-type: none"> El "Estado no determinable" se indica, por ejemplo, durante el diagnóstico de sistema para todos los módulos de periferia y sistemas de periferia configurados después de rearrancar la CPU. Este estado también puede indicarse temporalmente durante el funcionamiento, en caso de que se produzca un alud de alarmas de diagnóstico en todos los módulos. No se puede emitir el estado de los módulos de un subsistema que está conectado a un CP.
!	rojo	Componente "no accesible" Se indica en los módulos desenchufados o módulos configurados pero no existentes.
!	negro	No hay datos de entrada o salida disponibles. Los canales de entrada o salida del (sub)módulo están bloqueados.
!	verde	Mantenimiento necesario (Maintenance Required)
!	amarillo	Mantenimiento solicitado (Maintenance Demanded)
!	rojo	Error: componente inutilizado o no disponible por tratarse del tipo incorrecto
!	rojo	El estado de un módulo en un nivel más profundo no se corresponde con el estado "El componente funciona correctamente"

Figura 1.9 Significado de cada símbolo de los estados posibles de los equipos de la estación del PLC [3].

En la figura 1.9, se muestra cada símbolo que indicara el estado de los módulos de la estación de la CPU para que el operador tome las respectivas acciones dependiendo del caso. Por otro lado, se puede encontrar ms información en otras fichas como “Identificación” y “Estadística”. Se muestra a continuación la figura 1.10 la cual indica la página “Información del módulo” y su respectiva descripción de las fichas y

parámetros.

Estado	Nombre	Referencia	Dirección IP	Comentario
<input checked="" type="checkbox"/>	MvSCALANCE	6GK5204-0AB00-2BA3	192.168.3.217	Topología
<input checked="" type="checkbox"/>	IM155-5PNST	6ES7155-5AA00-0AB0	192.168.3.122	Topología
<input checked="" type="checkbox"/>	IM155-6PNST	6ES7155-6AA00-0BN0	192.168.3.123	Topología

Paquetes de datos enviados:

- Enviados correctamente: 6159
- Colisiones en el intento de envío: 0
- Interrumpido por otros motivos: 0

Paquetes de datos recibidos:

- Recibidos sin errores: 1435
- Rechazado debido a errores: 0
- Denegados por escasez de recursos: 0

Estadística Puerto 1

Paquetes de datos enviados:

- Enviados correctamente: 869

Figura 1.10 Pagina estándar "Información del Módulo" [3].

En “Información de modulo” (1), se presenta una tabla con información del sistema maestro DP, sistema maestro PROFINET IO, demás módulos y submódulos. EN 2) se puede acceder a información adicional de módulos superiores. En “topología”, al presionar en dicho título, accederá a la vista gráfica de la topología prevista. En 4) si existe un enlace disponible, se podrá acceder a un servidor web del dispositivo seleccionado. En la ficha “Estado”, muestra información del estado del módulo ante un fallo o un aviso repentino. En la ficha “Identificación”, se muestran datos del módulo escogido o seleccionado. Finalmente, en la ficha de “Estadísticas”, se muestran valores de paquetes enviados y recibidos, correspondientes a estadísticas totales y estadísticas de determinados puertos (véase la figura 1.11).

Paquetes de datos enviados:

- Enviados correctamente: 6159
- Colisiones en el intento de envío: 0
- Interrumpido por otros motivos: 0

Paquetes de datos recibidos:

- Recibidos sin errores: 1435
- Rechazado debido a errores: 0
- Denegados por escasez de recursos: 0

Estadística Puerto 1

Paquetes de datos enviados:

- Enviados correctamente: 869

Paquetes de datos recibidos:

- Recibidos sin errores: 317
- Rechazado debido a errores: 0
- Denegados por escasez de recursos: 0

Figura 1.11 Ficha "Estadística" de la página estándar "Información del módulo" [3].

1.3.5 Avisos

Como se muestra en la figura 1.12, se tiene dos apartados, como el apartado de búfer de “Avisos”, en el cual se presenta de manera resumida un análisis de los errores en el equipo. Los avisos del PLC, se pueden mostrar en orden cronológico si se presiona la ficha de “Fecha”. Además, hay demás ítems que detallan información de los avisos como; número de aviso, fecha hora, texto de aviso, etc. En el segundo apartado “Detalles del número de aviso”, se muestra información detallada de algún aviso previamente seleccionado.

Nº aviso	Fecha	Hora	Texto de aviso	Estado	Acuse
1	13.11.2014	08:23:24.644	Dispositivo PN 5 a sistema PN... aparecido		
2	13.11.2014	08:23:24.798	Dispositivo PN 4 a sistema PN... aparecido		
3	13.11.2014	08:23:24.948	PB-Slave 3 a sistema PB... aparecido		
4	13.11.2014	08:23:25.099	PB-Slave 1 a sistema PB... aparecido		
5	13.11.2014	08:23:25.251	Dispositivo PN 3 a sistema PN... aparecido		
6	13.11.2014	08:23:25.402	Dispositivo PN 2 a sistema PN... aparecido		
7	13.11.2014	08:23:25.553	Dispositivo PN 1 a sistema PN... aparecido		

Detalles del número de aviso: 93
Nombre abreviado: SCALANCE-X204IRT Referencia: 6GK5204-0BA00-2BA3

Figura 1.12 Página estándar "Avisos" [3].

1.3.6 Comunicación

En esta página se encuentra información de cuatro fichas como; parámetros, estadísticas, recursos y conexiones, cada una de las fichas posee parámetros de interés e información.

La figura 1.13, muestra la ficha “Parámetros”. En “Conexión de red”, se visualiza información de las interfaces Ethernet y Profinet del CPU que se está manejando además de mostrar su dirección MAC. En 3), se muestra a dirección IP y la subred configurada el PLC. En 4), aparecerá información de número de puertos, estado de link, ajustes, modo y medio de conexión.

Parámetros

Interfaz PROFINET [X1]:

Conexión de red:

- Dirección MAC: AC-64-17-48-D7-38
- Nombre: plc_planta.interfaz_profinet_1

Parámetro IP:

- Dirección IP: 192.168.0.4
- Máscara de subred: 255.255.255.0
- Router estándar: 192.168.0.1
- Ajustes IP: Dirección IP configurada en el proyecto

Propiedades físicas:

Nº de puerto	Estado de link	Ajustes	Modo	Medio de conexión
X1 P1	Aceptar	Automático	TP 100 Mbit/s dúplex	Cable de cobre
X1 P2	Aceptar	Automático	TP 100 Mbit/s dúplex	Cable de cobre

Figura 1.13 Página estándar "Comunicación".

En la ficha de “Estadísticas” se mostrarán información sobre la trasferencia de datos del PLC.

Continuando con la ficha de “Recursos de conexión”, la figura 1.14 muestra 3 apartados que son: “Recursos”, se indica el consumo de los recursos de las conexiones; “Número de Conexiones”, se muestra el número máximo de conexiones y el número de conexiones no ocupadas; y “Conexiones”, que indica

información respecto a la conexiones ocupadas y reservadas para la comunicación con protocolos de ES, S7, Open User, HMI, entre otras (véase la figura 1.14).

Nombre de usuario: [] **Conectar:**

Comunicación

Parámetros | Estadística | **Recursos de conexión** | Estado de la conexión

N.º de conexiones:

- N.º máx. conexiones: 256
- Conexiones no ocupadas: 250

Conexiones:	reservadas	ocupadas
Comunicación ES:	4	0
Comunicación HMI:	4	0
comunicaciones S7:	0	0
Comunicación OpenUser:	0	0
comunicaciones Web:	2	6
Otros tipos de comunicación:	--	0

Estado	ID local (Hex)	Slot de pasarela	Tipo de dirección remoto	Dirección remota	Tipo
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc

Estado de la conexión:

Detalles:

- Detalles de la dirección
 - Dirección local: 192.168.0.4
 - Puerto local: 80
 - Dirección remota: 192.168.0.5
 - Puerto remoto: 62778
- Diagnóstico
 - Causa del error:
- Estadística
 - Intentos actuales de establecer la conexión: 0
 - Intentos de conexión fallidos: 0

Figura 1.14 Ficha "Recursos" de la página estándar "Comunicación".

Finalmente, en la ficha de “Estado de Conexión”, se muestra tres apartados que se pueden visualizar en la figura 1.15, visualizándose la ficha “Conexiones”, donde se indica el estado de cada conexión del PLC. En 2), se indica información resumida de las conexiones establecidas y que se están estableciendo. La tabla muestra además información como; estado de la conexión, slot, la ID local, dirección remota (dirección IP), tipo de conexión, etc. En el apartado “Detalles”, se encontrará información detallada de la conexión antes seleccionada.

Nombre de usuario: [] **Conectar:**

Comunicación

Parámetros | Estadística | Recursos de conexión | **Estado de la conexión**

Estado	ID local (Hex)	Slot de pasarela	Tipo de dirección remoto	Dirección remota	Tipo
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc
Conexión establecida	0	---	IPv4	192.168.0.5	Adhoc

Estado de la conexión:

Detalles:

- Detalles de la dirección
 - Dirección local: 192.168.0.4
 - Puerto local: 80
 - Dirección remota: 192.168.0.5
 - Puerto remoto: 62778
- Diagnóstico
 - Causa del error:
- Estadística
 - Intentos actuales de establecer la conexión: 0
 - Intentos de conexión fallidos: 0

Figura 1.15 Ficha "Conexiones" de la página estándar "Comunicación".

1.3.7 Topología

En dicha página se mostrará la estructura de la topología del sistema PROFINET IO implementado, por tanto, se muestran 3 fichas que contienen sus respectivas vistas de la topología. Estas son:

- Vista gráfica (topología prevista y real)
- Vista de tabla (sólo topología real)
- Vista general de estado (sin representación de las relaciones topológicas)

➤ Vista gráfica

En esta sección se mostrará dos tipos de topologías. Topología prevista, que indica la asignación topológica de dispositivos PROFINET que fallan, y sus respectivas diferencias entre un modelo previsto y real. Mientras que la topología real, muestra la configuración topológica de dispositivos PROFINET de un sistema PROFINET IO y demás dispositivos PROFINET no configurados que muestran una relación de cercanía directa.

La figura 1.16 muestra las diferentes conexiones entre los diferentes dispositivos PROFINET.

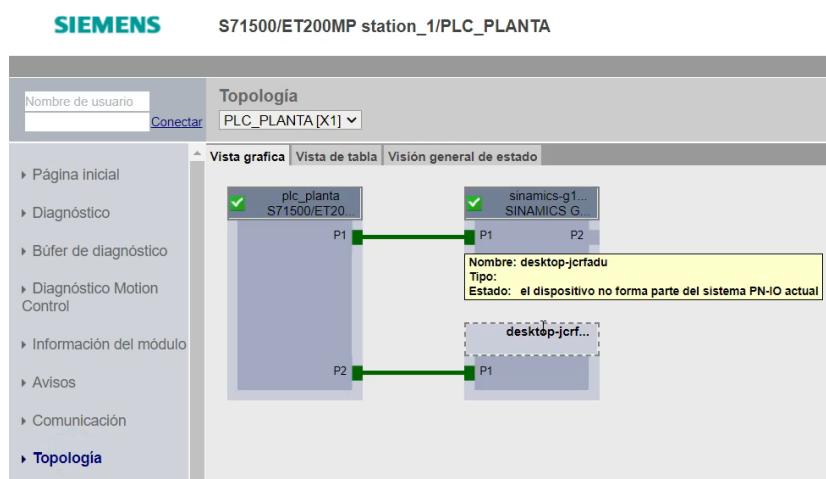


Figura 1.16 Vista grafica (Topología prevista y real).

Cada conexión posee un color específico. Para entender su significativo, se muestra la figura 1.17 donde se indica el significado de los colores que puede tener un enlace o una conexión.

Enlace	Significado	
	Topología prevista	Topología real
verde	La conexión real actual se corresponde con la conexión teórica configurada.	Conexiones detectadas
rojo	La conexión real actual no se corresponde con la conexión teórica configurada (p. ej. puerto intercambiado).	-
amarillo	No se puede diagnosticar la conexión. Causas: <ul style="list-style-type: none"> • La comunicación con un dispositivo no funciona (p. ej. se ha desenchufado el cable) • Conexión con un componente pasivo (p. ej. switches o cables) • Conexión con dispositivos/dispositivos PROFINET de otro controlador IO o subsistema IO 	-

Figura 1.17 Significado de los enlaces de acuerdo a su color en las respectivas topologías [3].

➤ Vista de tabla

Se muestra la respectiva tabla de los dispositivos conectados bajo el esquema de “topología real” (véase la figura 1.18). Además, se visualizan iconos o símbolos (1) cuyo significado se explica en la figura 1.19.

Puerto	Estado	Nombre	Tipo de módulo	Puerto	Nombre	Puerto
	✓	plc_planta	S71500/ET200MP station	port-001	sinamics-g120sv-pn	port-001
	✓	sinamics-g120sv-pn	SINAMICS G120 CU250S-2 PN Vector V4.7	port-002	désktop-jcrfadu	port-001
		desktop-jcrfadu		port-001	plc_planta	port-001
				port-002		
				port-001	plc_planta	port-002

Figura 1.18 Página “Topología” (Vista de tabla).

Símbolo	Significado
	Dispositivos PROFINET configurados y accesibles
	Dispositivos PROFINET no configurados y accesibles
	Dispositivos PROFINET configurados pero no accesibles
	Los interlocutores cuya relación de vecindad no se puede determinar o cuya relación de vecindad se ha detectado de forma incompleta o incorrecta

Figura 1.19 Significado de los símbolos de estado de los dispositivos PROFINET [3].

➤ Vista general de estado

Se muestra de manera sinóptica todos los dispositivos PROFINET IO/ PROFINET, en una sola página. Mediante los símbolos mostrados, se puede dar un diagnóstico rápido cuando se produce un error (véase la figura 1.20).

Vista grafica	Vista de tabla	Visión general de estado	
	plc_planta S71500/ET200...		sinamics-g120... SINAMICS G1...

Figura 1.20 Página estándar "Topología" en la ficha "Vista general de estado".

1.3.8 Estado de variables

En la figura 1.21, se muestra el estado de las variables que se procesan en el PLC, por medio de 3 apartados. En “Dirección”, se indica una “dirección simbólica”, cuyo comportamiento se desea observar. En “Formato”, se escogerá un formato para visualizar la variable escogida. En “Valor”, se muestra el valor de la variable es su estado de operación, en relación al formato escogido anteriormente.

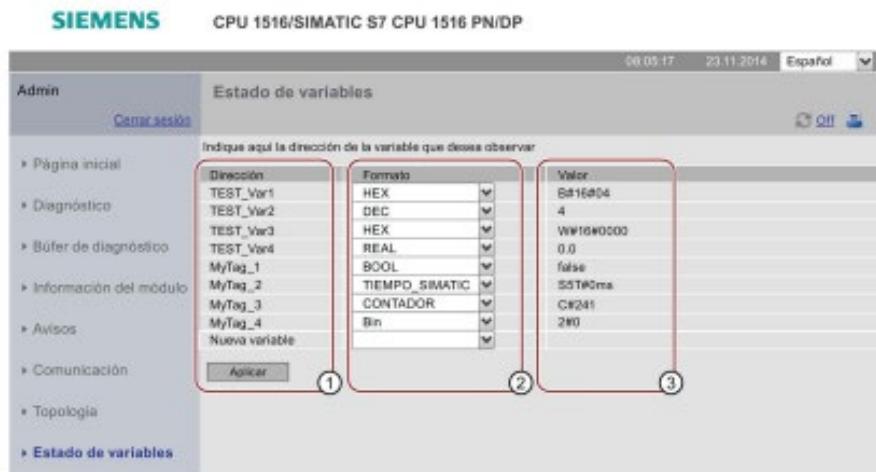


Figura 1.21 Página estándar "Estado de variables" [3].

1.3.9 Tablas de observación

En esta sección se pueden observar tablas de observación, que han sido previamente configuradas y aptas para la página web (véase figura 1.22). En esta página se podrá seleccionar la tabla y mediante el apartado de “Nombre”, se indica el nombre simbólico de la variable. Se muestra en el apartado de “Dirección”, la dirección de la variable, además del “Formato” y “Valor”, de la variable escogida.

Nombre	Dirección	Formato	Valor	Comentario
"Tag_1"	%I0.0	BIN	2#0	
"Tag_2"	%I0.1	BOOL	FALSE	
"Tag_3"	%Q0.0	BIN	2#1	TRUE
"Tag_4"	%Q0.0	BOOL	0	
"Tag_5"	%I3.3	DEZ		

Figura 1.22 Página estándar "Tablas de observación" [3].

1.3.10 Páginas de usuario

En esta sección será de suma importación pues en esta parte se podrán cargar páginas HTML creadas por el usuario para leer datos del sistema o del proceso industrial que maneja el PLC. Con la ayuda de STEP 7 (TIA PORTAL), se generan los bloques DB Web Control y DB de fragmento los cuales cuando se carguen en el PLC, se podrá sincronizar el programa del proceso con el servidor WEB (véase la figura 1.23).

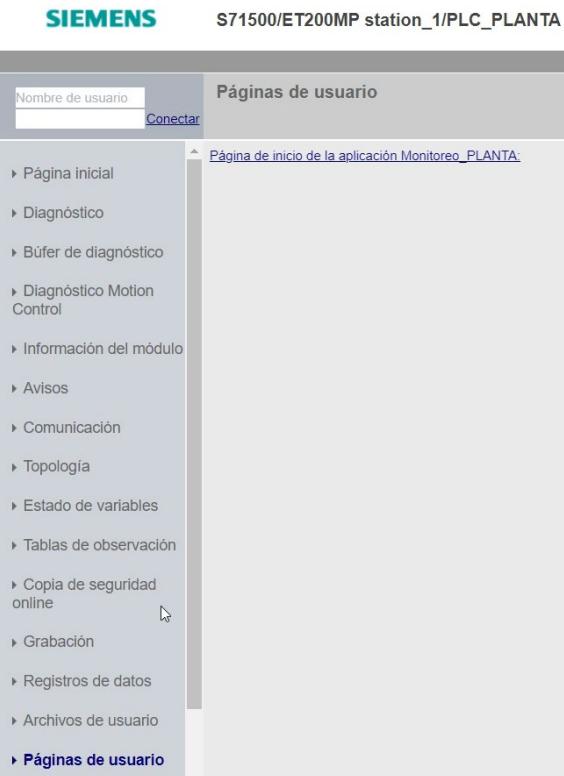


Figura 1.23 Sección "Páginas de usuario".

1.3.11 Explorador de archivos

En el navegador se tiene la posibilidad de observar el contenido de una SIMATIC Memory Card, una vez que nos ubiquemos en la página “Explorador de archivos”, esto sin la necesidad de usar el programa STEP 7 o TIA Portal. Aquí se visualizarán una lista de archivos y directorio disponibles. Estos se podrán descargar, borrar renombrar o cargar (véase la figura 1.24).

The screenshot shows the Siemens TIA Portal interface. At the top, it displays 'SIEMENS' and 'S71500/ET200MP station_1/PLC_PLANTA'. On the left, there's a sidebar with a 'Nombre de usuario' input field and a 'Conectar' button. Below these are links: 'Página inicial', 'Diagnóstico', 'Búfer de diagnóstico', 'Diagnóstico Motion Control', and 'Información del módulo'. The main content area is titled 'Navegador de archivos' and shows a table of files in the directory 'S71500/ET200MP station_1 /'. The table has columns for Nombre (Name), Tamaño (Size), Modificado el (Modified on), Borrar (Delete), and Renombrar (Rename). Two files are listed: '_LOG_' (32768 bytes, modified 15:28:06 11.12.2018) and 'crdinfo.bin' (512 bytes, modified 15:28:06 11.12.2018). Below the table, there's a section titled 'Operaciones de directorio.' with a file upload input field and a 'Cargar archivo' (Load file) button.

Figura 1.24 Página estándar "Navegador de archivos".

1.3.12 Datalogs

Siendo la última página estándar del servidor, permite mostrar los datalogs que se han creado, mostrándonos parámetros de interés como Nombre, Tamaño, Última modificación, además de permitirnos consultar el archivo datalog escogido (véase la figura 1.25).

Nombre	Tamaño	Modificado el	Recuperar y borrar
MyDataLog1.csv	43	12:05:18 22.07.2014	[Icono]
MyDataLog2.csv	17	09:32:07 22.07.2014	[Icono]
MyDataLog3.csv	8	17:01:41 22.07.2014	[Icono]

Figura 1.25 Página estándar "DataLogs" [3].

1.4 Procedimiento para crear y acceder a páginas web definidas por el usuario desde el servidor web

Dado que la parte fundamental del proyecto técnico es un análisis de las herramientas que nos proporciona lenguajes como HTML y JavaScript para el control y monitoreo de datos y variables con el uso de PLCs, es oportuno enumerar cada uno de los pasos para crear páginas web definidas por el usuario, pues dicha aplicación o página es la que monitorea y muestra el estado de las variables del proceso. Por lo tanto, se muestra los pasos a seguir para crear y guardar la página web en el servidor del autómata S7-1500 de la firma SIEMENS por medio de la figura 1.26.

Pasos:

1. Para la creación de la página de usuario es necesario el uso de un editor de HTML, tomando como consideración que el IDE, cumpla con los estándares de W3C. Se crea el archivo HTML, donde el usuario o desarrollador puede incluir librerías de JavaScript según sea su conveniencia.
2. En un directorio de la PC de usuario, guarde el archivo HTML creado conjuntamente con archivos fuentes como *gif, *js o *jpg.

3. En el programa STEP 7(TIA PORTAL), configure la sincronización de la página del usuario con el programa del PLC por medio de la instrucción “WWW” y “genere los bloques” de control que regulan la visualización de las páginas web.
4. Terminado las configuraciones previas, cargue el programa en el PLC o CPU.
5. Desde el navegador web, abra la página de usuario creada y guardada en el servidor WEB del PLC.

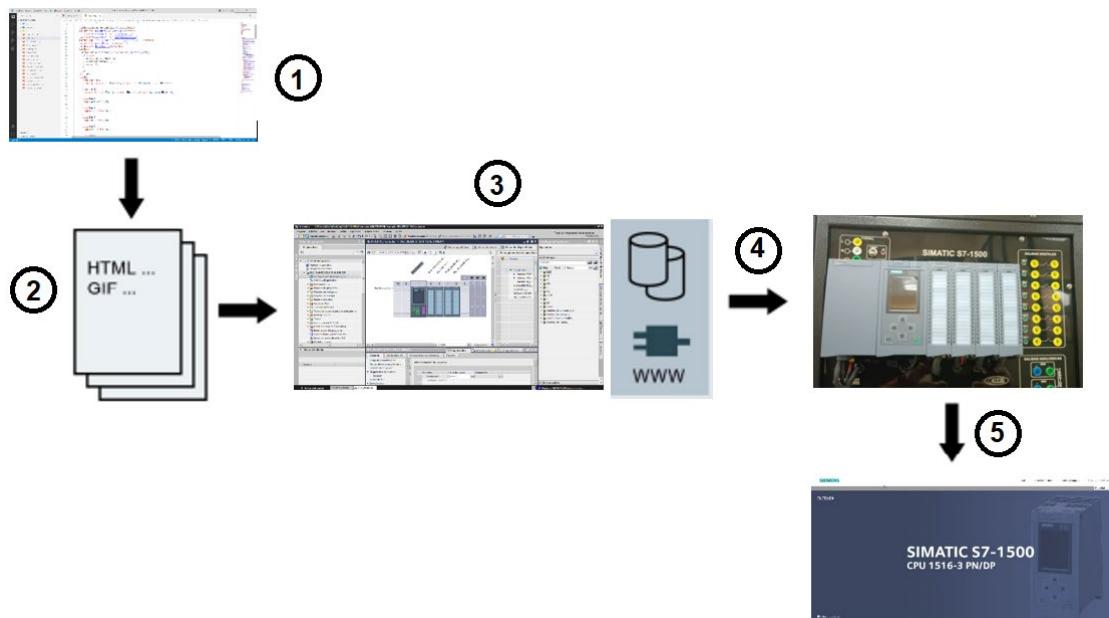


Figura 1.26 Pasos para acceder a páginas propias del usuario desde el servidor WEB.

En un capítulo posterior, se explicará con más detalles la creación de páginas web utilizando las herramientas de HTML y librerías de JavaScript. Así mismo, se explicará con mayor detalle la programación y configuración del servidor web en TIA PORTAL. Se describirá el proceso para la configuración de páginas definidas por el usuario en el servidor web y demás configuraciones para poder vincular la transferencia de información entre la página del usuario y los datos del proceso del PLC o CPU.

1.5 Comandos AWP

Como una de las partes más importantes para la comunicación del PLC con la página de usuario, son los comandos AWP (Automation Web Programming), que son comandos con una sintaxis especial introducidos en forma de comentario en el documento de HTML, estos permiten las siguientes funciones [1]:

- Leer variables PLC
- Escribir variables PLC
- Definir tipos Enum
- Asignar tipos Enum a variables
- Definir fragmentos de bloques de datos
- Importar fragmentos de bloques de datos
- Leer variables especiales

- Escribir variables especiales

A continuación, se muestra la representación de la sintaxis en HTML, para ejecutar las funciones antes mencionadas por medio de la figura 1.27:

Función	Representación
Leer variables PLC	<code>:=<Varname>:</code>
Escribir variables PLC	<code><!-- AWP_In_Variable Name='<Varname1>' --></code>
Leer variables especiales	<code><!-- AWP_Out_Variable Name='<Typ>:<Name>' --></code>
Escribir variables especiales	<code><!-- AWP_In_Variable Name='<Typ>:<Name>' --></code>
Definir tipos Enum	<code><!-- AWP_Enum_Def Name='<Name Enum-Typ>' Values='0:<Text_1>',1:'<Text_2>',...,x:'<Text_y>' --></code>
Asignar tipos Enum a variables	<code><!-- AWP_Enum_Ref Name='<Varname>' Enum='<Name Enum-Typ>' --></code>
Definir fragmentos de bloques de datos	<code><!-- AWP_Start_Fragment Name='<Name>' [Type=<Typ>] [ID=<Id>] --></code>
Importar fragmentos de bloques de datos	<code><!-- AWP_Import_Fragment Name='<Name>' --></code>

Figura 1.27 Comandos AWP con su respectiva sintaxis según su función [3].

➤ Lectura de variables PLC

En este caso se pueden leer variables “Out” o de salida, al denominarse de salida se toma en cuenta la dirección de salida tomando como referencia el punto de vista del controlador. Para ello se emplea la siguiente sintaxis:

`:=<Varname>:`

En el apartado de `<Varname>`, se colocará las variables a leer del proyecto en TIA PORTAL.

Lectura de variables del tipo String y Character

Se puede leer variables de tipo String mediante el empleo de la siguiente función:

`:="Varname".MyString:`

➤ Escritura de variables PLC

Para leer variables “In” o de entrada, donde las variables a leerse, se las colocara en la parte de Header del documento WEB, y después con el uso de la sintaxis siguiente, se podrá leer la variable de entrada:

`<!-- AWP_In_Variable Name='<Varname1>' Name='<Varname2>' Name='<Varname3>' -->`

➤ Leer variables especiales

Se pueden leer variables especiales conocidas como variables HTTP, las cuales se pueden leer desde el PLC y poder transferirlas a variables especiales en el encabezado de respuesta HTTP. Para ello se empleará la siguiente sintaxis:

`<!-- AWP_Out_Variable Name='<Type>:<Name>' Use='<Varname>' -->`

➤ Escribir variables especiales

El servidor web permite escribir en el PLC valores de variables especiales escritas en un encabezado HTTP, como por ejemplo, se puede guardar información de una cookie de una página de usuario que accede a una página. Para ejecutar lo antes mencionado, se emplea la sintaxis:

```
<!-- AWP_In_Variable Name='<Type>:<Name>' Use='Varname' -->
```

➤ Definir tipos Enum

Por medio del siguiente comando, se puede definir tipos “Enum”, los cuales permiten cambiar valores numéricos del programa del PLC en textos y de manera viceversa. La sintaxis a emplearse:

```
<!-- AWP_Enum_Def_Name='<Name Enum-Typ>' Values='0:<Text_1>',  
1:<Text_2>, ..., x:<Text_y>' -->
```

➤ Asignación de tipos Enum a variables

La asignación de variables del programa del usuario a variables de tipo Enum, se las utiliza para operaciones de lectura y escritura. En operaciones de lectura, el servidor cambia un valor leído del PLC por un valor de texto Enum, mientras que en la operación de escritura, el servidor web, reemplaza el valor de texto Enum por un valor entero correspondiente de la enumeración. Para ello se usa el comando:

```
<!-- AWP_Enum_Ref_Name='<Varname>' Enum="<Enum-Type>" -->
```

➤ Definición de fragmentos

Se puede trabajar con “fragmentos” mediante el servidor web, con el fin de poder trabajar con páginas completas o con elementos individuales como imágenes o documentos. Para ello se emplea el comando:

```
<!-- AWP_Start_Fragment_Name='<Name>' [Type="<Typ>"] [ID="<Id>"] -->
```

➤ Importación de fragmentos

Con el servidor web, se pueden definir fragmentos en una página HTML e importarlos a otra página web, este recurso puede emplearse en caso de que se requiera importar archivos como logotipos o imágenes. La sintaxis a usarse:

```
<!-- AWP_Import_Fragment_Name='<Name>' -->
```

CAPÍTULO 2

Configuración e instalación del proceso industrial a monitorear

En este capítulo se describe la arquitectura y la configuración del sistema de velocidad que va a ser controlado y monitorizado a través de la aplicación WEB. Este sistema está conformado por el accionamiento SINAMICS G120, un motor de inducción de 0.75 HP y un controlador lógico programable S7-1500. Los componentes del sistema se comunican mediante Profinet, por consiguiente, el accionamiento cuenta con una unidad de control (CU) que soporta este estándar de comunicación abierto basado en ethernet. La CU utilizada para este proyecto es la 250S-2.

2.1 Arquitectura del sistema de velocidad

Para el presente proyecto se propone el control de un motor trifásico mediante el uso de un variador de frecuencia y un PLC. La arquitectura del sistema propuesto se muestra en la Figura 2.1. Se asume esta arquitectura como un ejemplo práctico de una de las aplicaciones en control de movimiento que son demandadas en la mayoría de los entornos industriales. El uso de un variador de frecuencia es recomendable para aplicaciones de velocidad, puesto que permite un óptimo comportamiento de un motor, además de protegerlo ante eventualidades anormales. Por lo tanto, gracias al variador de frecuencia es posible regular la velocidad del motor, desarrollar un sistema de alarmas ante condiciones anormales y obtener parámetros de monitoreo como, por ejemplo: velocidad, potencia, corriente, torque, etc.

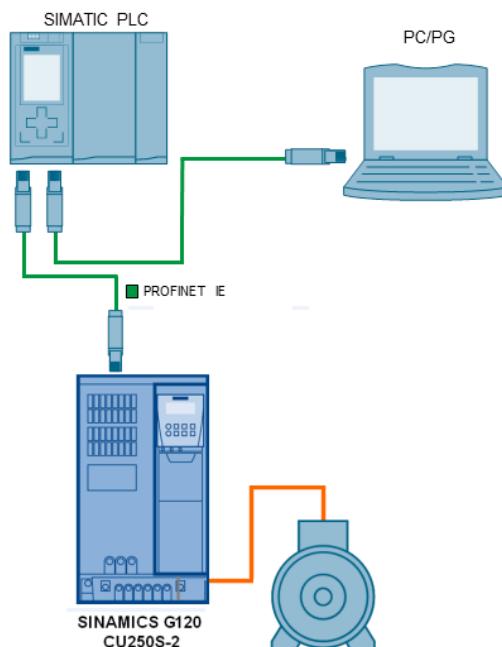


Figura 2.1 Arquitectura del sistema de velocidad a ser controlado y monitorizado a través de la aplicación WEB.

A continuación, se describe de manera detallada cada uno de los componentes de la arquitectura.

2.1.1 Convertidores estándar SINAMICS G120

Se trata de una unidad de la marca SIEMENS destinada a la regulación precisa del par o la velocidad de motores trifásicos. Cuenta con diferentes versiones con gamas de potencia desde 0.37 kW a 250 kW [6]. Este sistema de convertidor modular posee dos unidades funcionales:

- Power Module (PM), también conocida como unidad de potencia
- Control Unit (CU), también conocida como unidad de control

➤ Unidad de Potencia.

Este módulo se encarga de suministrar la potencia necesaria al motor. Para su funcionamiento emplea una tecnología basada en transistores bipolares de puerta aislada (IGBT), juntamente con técnicas de modulación PWM. Posee un gran nivel de seguridad y precisión sin la necesidad de acoplar un encoder en el motor. Por otra parte, gracias a la tecnología “Efficient Infeed”, no requiere de refrigeración adicional, proporcionando un ahorro energético considerable [6]. En la figura 2.2 se muestra una vista panorámica del PM que será utilizado en este proyecto.



Figura 2.2 Unidad de potencia PM240-2 [6].

➤ Unidad de control.

Esta unidad se encarga de controlar y vigilar la etapa de potencia (PM) y al motor trifásico. A su vez, permite la comunicación con un controlador central o local y con dispositivos destinados a labores de vigilancia [6]. Dentro de la marca SIEMENS, existen diferentes modelos de unidades de control que se pueden combinar con los módulos de potencia. La tabla 2.1 muestra un informe de compatibilidad entre estos componentes.

Tabla 2.1 Compatibilidad entre unidades de control y módulos de potencia (Tecnología SIEMENS).

Unidades de control	Unidades de potencia con protección IP20	
	PM 240-2	PM250
CU230P-2	X	X
CU240E-2	X	X
CU250S-2	X	X

Los accionamientos disponen de unidades de control que se pueden comunicar mediante protocolos de comunicaciones como PROFINET o PROFIBUS. Además, como una opción para interactuar con la unidad de control, existen paneles operadores, disponibles en dos versiones: panel operador inteligente “IOP” y el básico de segunda generación “BOP-2”, que se acoplan a los convertidores con el fin de realizar ajustes en los parámetros del variador de frecuencia. Con estos paneles también es posible realizar un control manual del motor, visualizar alarmas y diagnosticar fallos que se presentan en el equipo, en la red o en el motor trifásico [5].

En la figura 2.3 se puede apreciar los diferentes convertidores SINAMICS G120 disponibles, una vez que se han acoplado la unidad de control, el módulo de potencia y el panel operador, respectivamente.



Figura 2.3 SINAMICS G120, tamaños FSA, FSB y FSC con Control Unit CU240E 2 F y Basic Operator Panel BOP 2 [6].

La unidad de control CU250S-2 posee diferentes versiones. Para este proyecto se utilizará la CU 250S-2 PN, que cuenta con una interfaz PROFINET y está destinada a aplicaciones complejas como por ejemplo extrusoras y centrifugadoras, además de aplicaciones con o sin encoder (Función Posicionador simple EPos) [5]. La figura 2.4, muestra una vista frontal de la CU 250S-2 PN con un panel de operador IOP-2.



Figura 2.4 SINAMICS G120 con la unidad CU250S 2 PN con control mediante el panel IOP-2

2.1.2 Módulo SIMATIC S7 1516-3 PN-DP

Se considera uno de los autómatas con más altas prestaciones ofrecidas por la marca SIEMENS, perteneciente a la familia S7 (véase la figura 2.5). Pertenece a la serie 1500, por lo que el número 1516 es su modelo específico. Después del número antes mencionado y del guion, se indica el número de interfaces de comunicación, con lo cual, las siglas PN indicarán una interfaz PROFINET y las siglas DP, una interfaz PROFIBUS, bajo el estándar RS485. Además, este PLC posee un puerto Ethernet de alta velocidad (alrededor de 1 Gbps) y una memoria de trabajo de 6,5 MB [5].



Figura 2.5 PLC 1516-3 PN/DP [7].

Debido a su manejo sencillo, precio económico, es destinado para aplicaciones en; máquinas textiles, máquinas envasadoras, industria eléctrica, automovilística, procesos de tratamiento de agua residuales y procesos de alimentación y bebidas. Debido a su aplicación en entornos industriales ofrece protección ante perturbaciones electromagnéticas, choques y vibraciones [7].

2.2 Configuración del servidor WEB mediante el programa TIA PORTAL V16

2.2.1 Activación del servidor web

El PLC S7 1500 seleccionado para este proyecto posee un “servidor WEB” incorporado, el cual debe ser configurado en la sección de Propiedades de la CPU. Para esto, es necesario dirigirse a la opción “Servidor Web”, dentro de la pestaña General y marcar la casilla “Activar servidor web en el módulo”. Este proceso se ilustra de mejor manera en la figura 2.6. Cabe indicar que la segunda casilla que permite al usuario acceder al servidor WEB mediante el protocolo seguro HTTPS debe quedar desactivada.

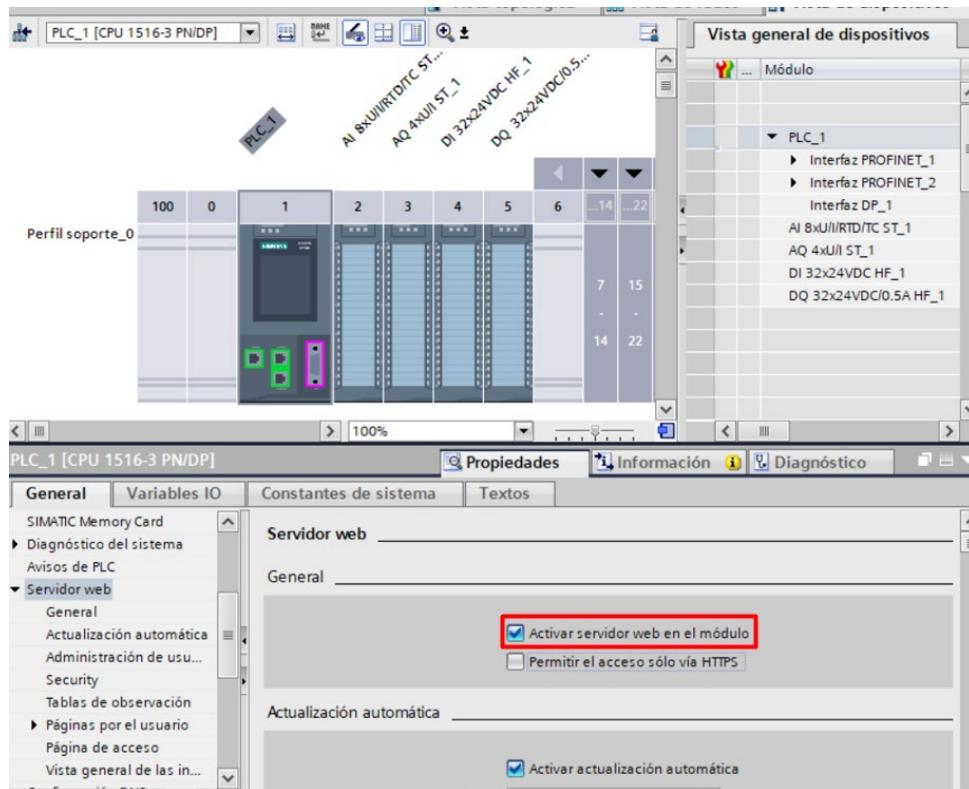


Figura 2.6 Activación del servidor web.

2.2.2 Configuración del nivel de administración de los usuarios.

En esta sección, se configura todas las funciones permitidas para el usuario, por tanto, si se marcan todas las funciones como lo indica la figura 2.7, se otorgará un nivel de acceso “Administrativo”.

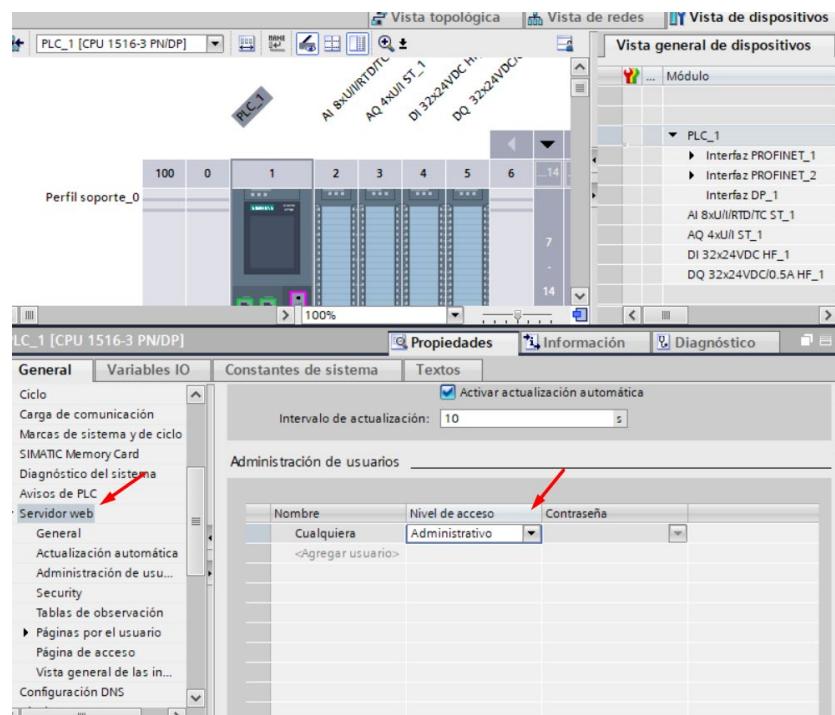


Figura 2.7 Configuración de administración de usuarios.

Luego, se otorga los permisos de “Protección & Seguridad”. En esta sección podemos configurar una contraseña con el fin de que solo una persona autorizada pueda acceder a funciones de lectura y escritura del proceso que se estará monitoreando con el servidor web. Caso contrario, si no se configura una contraseña, se otorgará un acceso completo al servidor para el monitoreo por parte de cualquier persona. La configuración de Protección y Seguridad se ilustra de mejor manera en la figura 2.8.

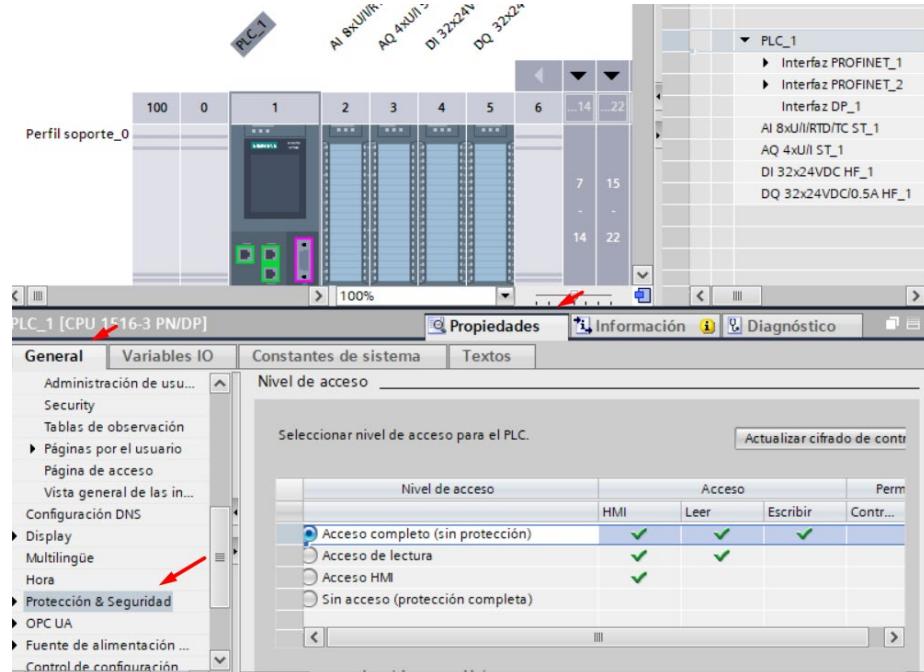


Figura 2.8 Configuración del nivel de acceso del servidor web.

2.2.3 Configuración de interfaces Ethernet para comunicación con el servidor

Dado que se emplea una comunicación por medio del bus de campo PROFINET, es necesario configurar las interfaces Ethernet, por tanto, es necesario asignar una dirección IP tomando en cuenta la red en la que se encuentra el PLC. Para este fin se ha escogido la interfaz X1 (véase la Figura 2.9), la cual posee dos puertos. Para este caso, el primer puerto se conectará a un computador, mientras que el segundo puerto, se conectará al variador de frecuencia para el control del motor asincrónico. Por consiguiente, una vez configurado el servidor, con solo ingresar la dirección IP desde el computador conectado al primer puerto, se podrá acceder al servidor web.

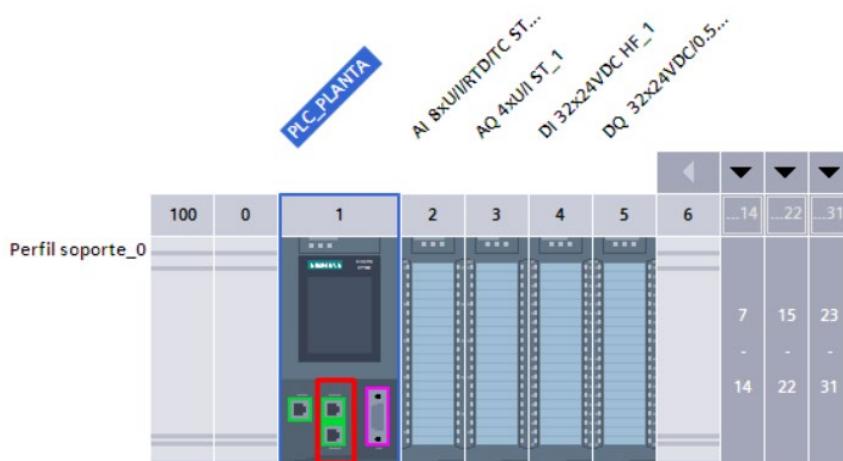


Figura 2.9 PLC Siemens 1516-3 PN/DP con sus respectivas interfaces PROFINET y PROFIBUS.

2.2.4 Asignación de dirección IP

En la figura 2.10 se muestra la interfaz PROFINET que debe ser configurada. Situándonos en la pestaña de “Vista de dispositivos”, luego señalando la interfaz X1 y haciendo clic en esta, aparecerá un menú nuevo indicando que se ha seleccionado la interfaz PROFINET. En la parte final del menú que se ha desplegado y situándonos en la pestaña “General (2)” y en “direcciones Ethernet”, debemos agregar la dirección IP que será parte de la red: 192.168.0.0/24. La dirección asignada se utilizará en un navegador web con el fin de ingresar al servidor que contiene la aplicación de monitoreo.

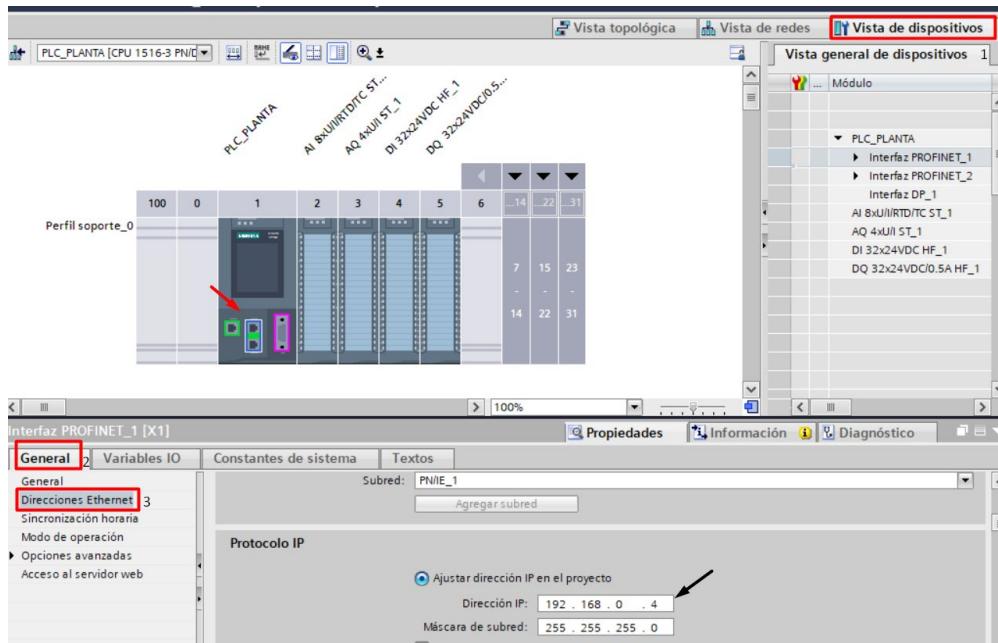


Figura 2.10 Asignación de la dirección IP de la interfaz X1 para acceder al servidor WEB.

2.2.5 Asignación de páginas web definidas por el usuario:

En esta parte, se debe configurar varios campos. Para una mejor ilustración estos campos han sido numerados en la figura 2.11. En “1”, se debe seleccionar la carpeta que contiene el documento HTML (aplicación de monitoreo diseñada), en “2” se debe seleccionar el archivo HTML principal, mientras que en “3”, se asignará un nombre a la aplicación que se está monitoreando a través del servidor web. El campo “4” es el botón que permite generar los bloques, los cuales son los responsables del control del proceso y generar los fragmentos necesarios para establecer la comunicación con el servidor web del PLC. Este botón debe ser presionado una vez se hayan cargado los datos de los campos anteriores.

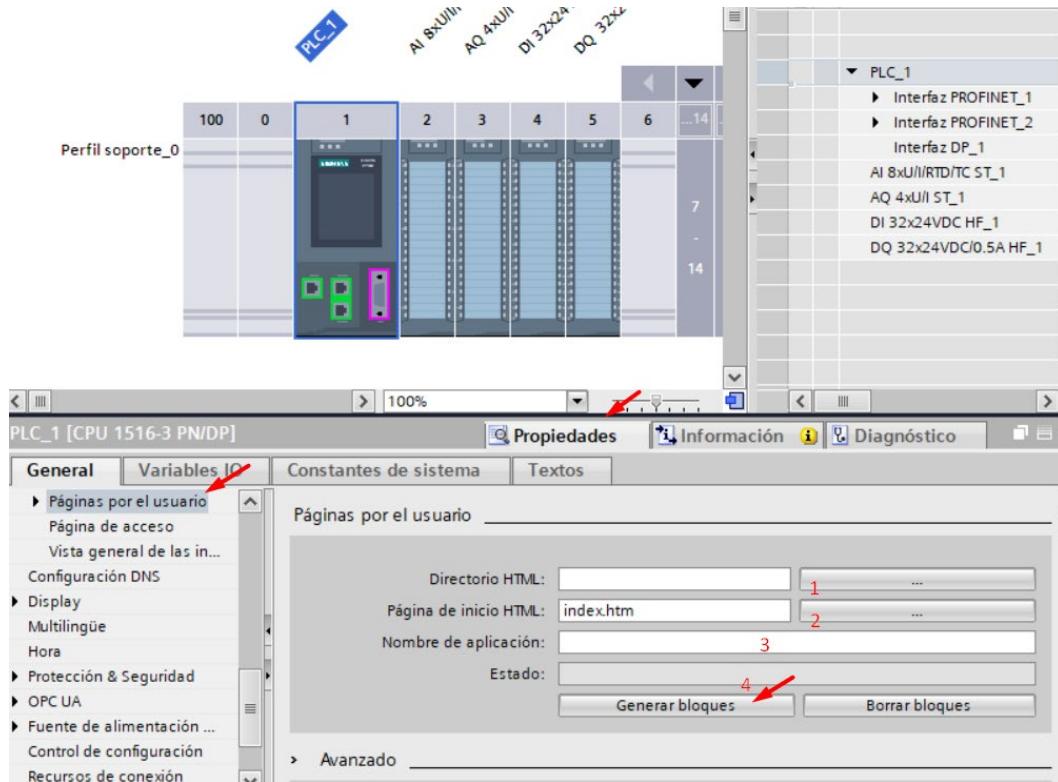


Figura 2.11 Configuración y asignación de páginas definidas por el usuario en el servidor.

Cabe mencionar que esta configuración se la completará una vez que se haya definido y programado la aplicación WEB con los editores de HTML y JavaScript. Todos los detalles sobre la aplicación WEB se abordarán en los Capítulos 3.

2.2.6 Instrucción WWW

Esta instrucción se encarga de sincronizar las páginas definidas por el usuario con el programa del autómata definido en TIA PORTAL. En otras palabras, este bloque se encarga de inicializar el servidor web del PLC (véase la Figura 2.12). El programa TIA PORTAL (STEP 7) se encarga de generar el DB Web Control, el cual es un parámetro de entrada de la instrucción WWW, además de indicar el contenido de las páginas web, información de estado y control [13].



Figura 2.12 Instrucción WWW en TIA PORTAL V16 (STEP 7) [13].

La instrucción WWW tiene una entrada (CTRL_DB) y una salida (RET_VAL). El parámetro CTRL_DB debe ser asignado con el valor 333, el cual no puede ser modificado, pues no se podría acceder a las páginas del usuario ubicadas en el servidor WEB del PLC. Por otro lado, el parámetro RET_VAL será un parámetro de tipo INT, el cual devuelve un dato o código de error que pueda generarse en el bloque WWW [13].

La figura 2.13 muestra como se ha insertado el bloque WWW en la sección de “Bloques de Programa”, específicamente en el programa de usuario MAIN. En el bloque se ha colocado el valor 333 como parámetro de entrada en CTRL_DB, mientras que la variable “ValorRetorno” (MW4) se ha asignado como variable para visualizar los códigos de error si es que ocurre un problema en el bloque WWW.

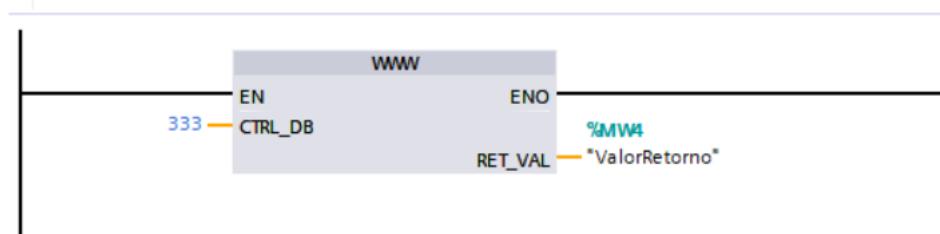


Figura 2.13 Configuración del bloque WWW en TIA PORTAL.

2.3Configuraciones previas en el variador SINAMICS G120 para el control de un motor asíncrono

El variador de frecuencia se conectará al PLC S7 1500 vía el bus de campo PROFINET. Previo a la configuración de red es necesario configurar una serie de parámetros en el variador, ya sea usando el panel operador o usando la interfaz de TIA Portal. En este caso se ha optado por la opción mediante el panel operador IOP-2. El procedimiento para esta configuración se detalla a continuación:

2.3.1 Configuración de puesta en marcha desde el panel IOP-2

Una vez encendido el variador, en la sección de “Instalación” se debe configurar la puesta en marcha rápida. Si se desconoce la configuración previa que puede tener el variador lo más recomendable es realizar un restablecimiento de los parámetros de fábrica. Estos dos procedimientos se ilustran en la figura 2.14.

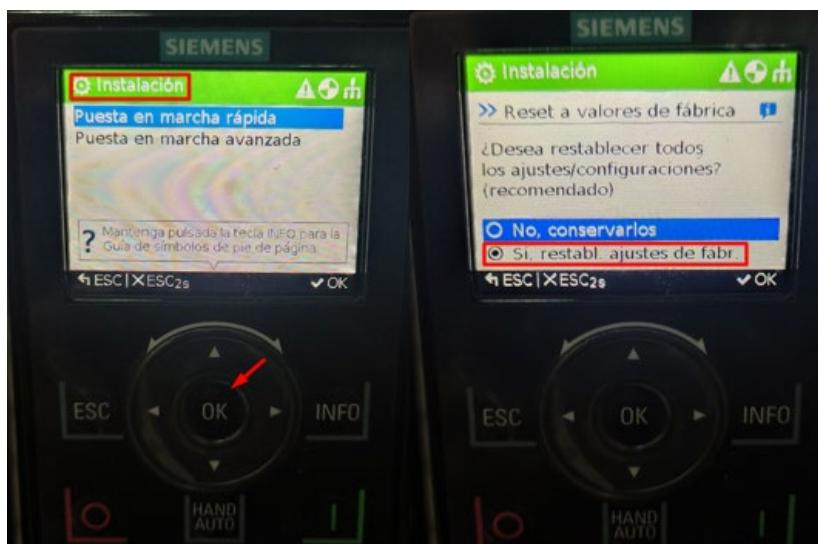


Figura 2.14 Puesta en marcha rápida y restablecimiento de los parámetros de fábrica en el variador SINAMICS G120.

Como se puede observar en la figura 2.14, el panel IOP-2, integra un botón central y una zona de navegación a su alrededor, la cual permite de manera táctil navegar entre las diversas opciones de los menús de configuración.

Luego de esperar el reinicio de parámetros de fábrica del variador SINAMICS, será necesario ingresar algunos parámetros de configuración, como, por ejemplo: norma del motor, tensión de alimentación, tipo de motor a controlar, corriente, potencia, velocidad máxima, etc. Para ingresar la mayoría de estos datos es necesario revisar la placa del motor. La figura 2.15 ilustra de mejor manera el menú de parámetros de configuración del motor.

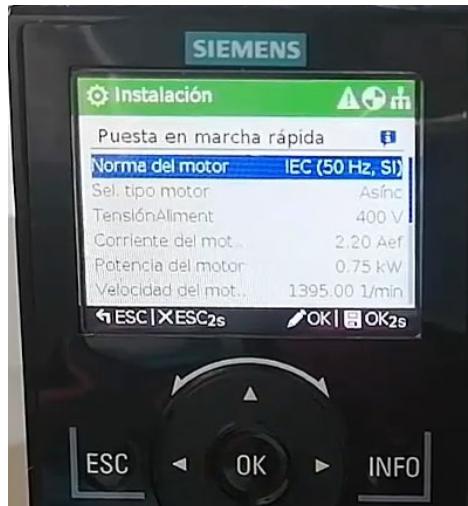


Figura 2.15 Configuración de parámetros del motor.

Por otra parte, es necesario configurar un parámetro adicional ubicado al final de la lista de parámetros. Este parámetro corresponde a la “Configuración de E/S. En esta configuración se debe escoger la opción “Seleccionar macro” y finalmente se elegirá la opción de “Bus de campo (7)”. Esto con el fin de establecer una comunicación con el variador por medio del bus del campo PROFINET. Este proceso de configuración se ilustra de mejor manera en la Figura 2.16.

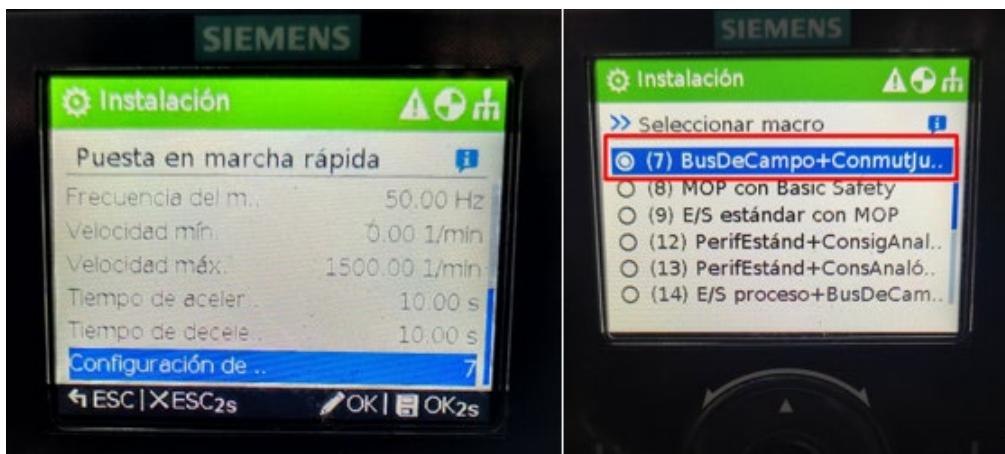


Figura 2.16 Configuración de la macro de conexión de bus de campo.

Luego del paso anterior, al presionar “OK”, transcurridos dos segundos, se guardarán los ajustes antes mencionados (véase la Figura 2.17). De esta manera se finalizará el proceso de puesta en marcha rápida del variador.

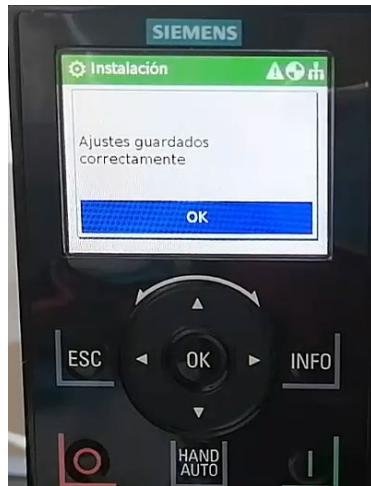


Figura 2.17 Ajustes guardados correctamente del variador SINAMICS.

A continuación, será necesario seleccionar el telegrama de comunicación, el cual será el responsable de gestionar la comunicación PROFINET entre el variador y el autómata. Para esta configuración, en la sección de “Parámetros”, se debe buscar el parámetro “922” mediante la opción “Búsqueda por número”. Esta configuración se ilustra de mejor manera en la Figura 2.18.

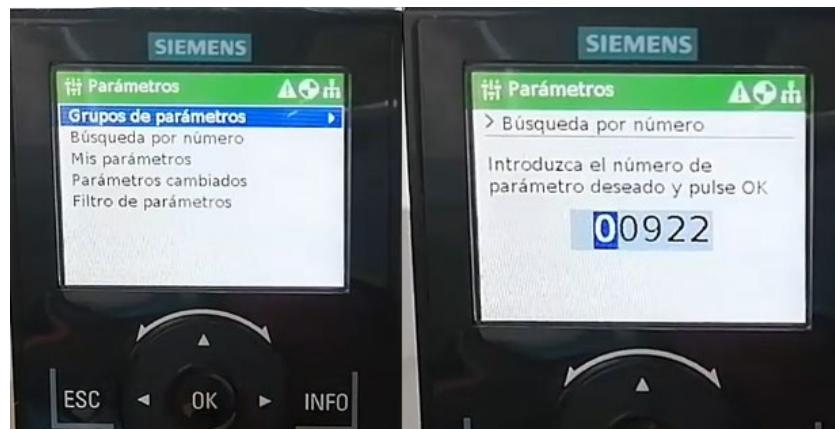


Figura 2.18 Configuración del parámetro 922 correspondiente al telegrama de comunicación.

Una vez que se ha buscado dicho parámetro, al presionar “OK”, se escogerá el Telegrama estándar 20 (véase la Figura 2.19). Finalmente, al presionar “OK” se guardarán los cambios realizados.



Figura 2.19 Telegrama 20 responsable de la comunicación PROFINET.

Como paso final, se procede a la identificación de datos del motor. Para esto, es necesario establecer el modo “Control” presionando el botón “HAND AUTO” (1). Posteriormente, se debe presionar el botón de encendido. Este proceso se ilustra de mejor manera en la figura 2.20.



Figura 2.20 Modo control del variador SINAMICS.

Al ejecutar este proceso aparecerá un mensaje indicando que la identificación del motor está en curso (véase la Figura 2.21). Al llegar al 100% se deberá presionar el botón OK.



Figura 2.21 Identificación del motor en el variador.

Finalmente, presionando nuevamente “HAND AUTO”, se establece un control automático para la conexión con el variador mediante el bus de campo PROFINET. Esto se puede verificar con la aparición de un ícono en la parte superior derecha del módulo IOP. Esto se ilustra de mejor manera en la figura 2.22.

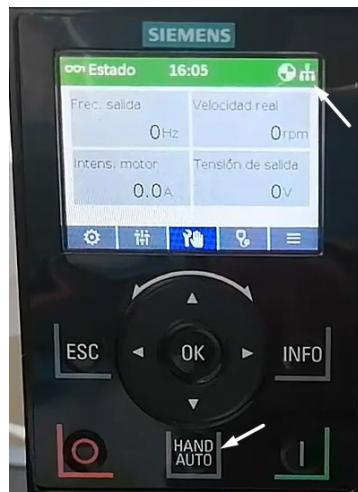


Figura 2.22 Variador SINAMICS en modo Automático estableciendo comunicación vía PROFINET.

2.4 Configuración del variador en TIA PORTAL V16 para la comunicación con el PLC S7-1500.

Configuración de la red PROFINET

Una vez conectados y energizados los equipos, en la opción “Accesos online” de TIA Portal, se debe verificar si el programa informático ha detectado al PLC y al variador (véase la Figura 2.23). Al situarnos sobre el variador “sinamics-g120sv-pn”, en la pestaña “Online y diagnóstico” asignamos una dirección IP al variador que estará en la red 192.168.0.0/24. Para este caso, se asignará al variador la dirección IP “192.168.0.7” con máscara de subred “255.255.255.0” (véase la Figura 2.24).

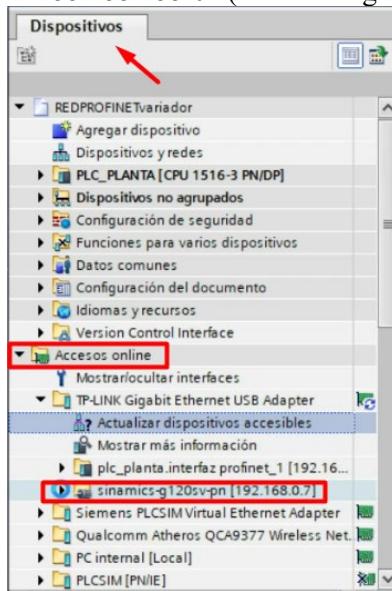


Figura 2.23 Detección del variador SINAMICS y del PLC 1516-3 PN/DP.

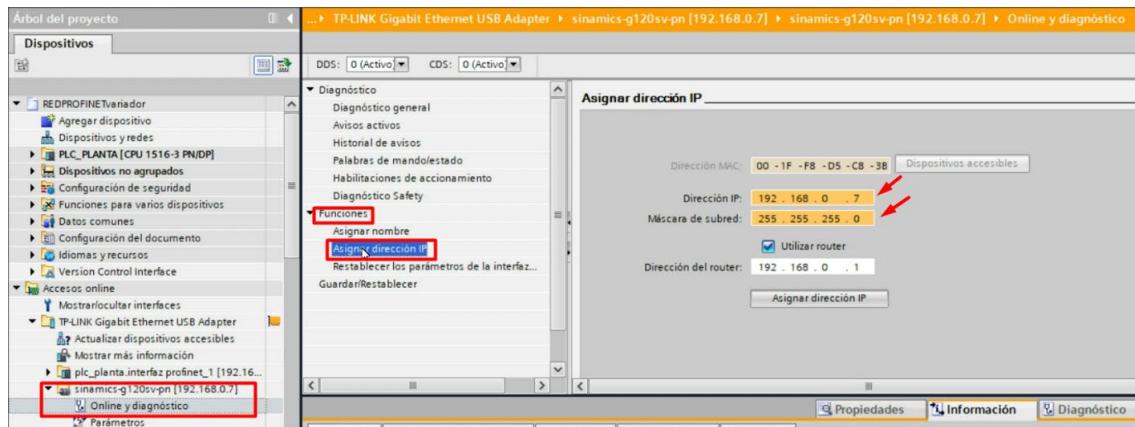


Figura 2.24 Asignación de dirección IP al variador SINAMICS.

Luego, se ubicará en la sección de “Dispositivos y redes” y presionando en la pestaña “Vista y redes”, se deberá observar al PLC escogido con todas sus interfaces de comunicación (dos interfaces PROFINET y una interfaz PROFIBUS). En esta sección se debe configurar el telegrama del variador “SINAMICS G120 C250S-2 PN”, esto con el fin de que se configure el telegrama de comunicación a través del archivo GSD del variador, este último contenido funciones básicas del dispositivo. Por tanto, se debe buscar en la sección de “Catálogo”, el GSD correspondiente al variador SINAMICS utilizado (véase la Figura 2.25).

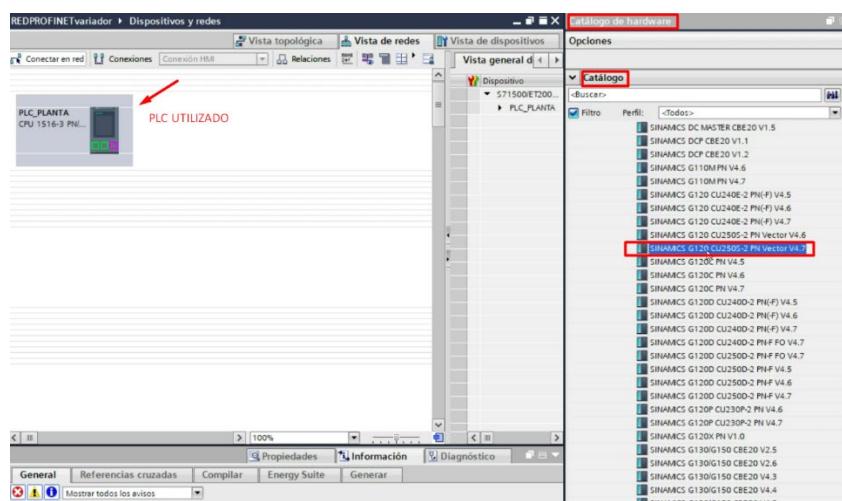


Figura 2.25 Selección del archivo GSD del variador SINAMICS G120 C250S-2PN.

Con la ayuda del ratón se debe arrastrar el archivo GSD a la zona de “Vista de redes”, donde automáticamente aparecerá una imagen representativa del variador SINAMICS G120. La Figura 2.26 describe el proceso antes mencionado.

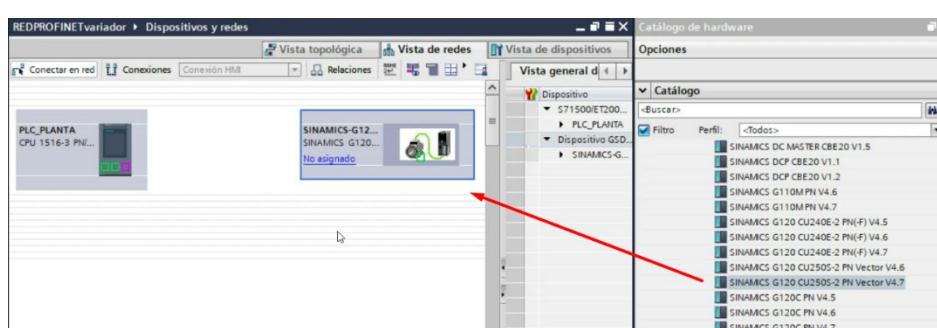


Figura 2.26 Representación del archivo GSD del variador SINAMICS G120 C250S-2PN versión 4.7.

Luego al hacer doble clic en la imagen del variador, se abrirá una pantalla tal como se ilustra en la figura 2.27. En esta instancia se debe asignar el telegrama a utilizar. Para este caso se utilizará el telegrama estándar 20, el cual fue configurado anteriormente en el variador. Para ejecutar este proceso es necesario ubicarse en la sección “Catalogó” y en la opción de “Submódulos” se debe arrastrar el telegrama 20 a la sección de “Vista general de dispositivos” del módulo.

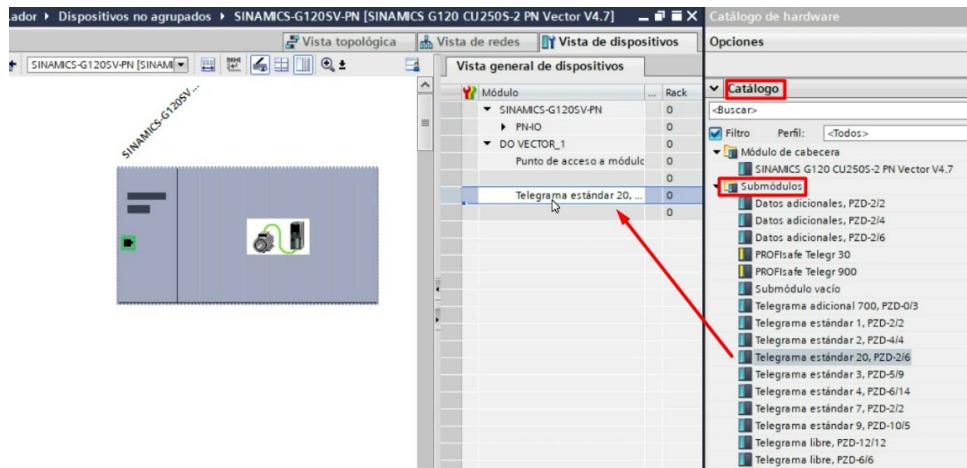


Figura 2.27 Configuración del telegrama 20 en el variador SINAMICS.

Regresando a la sección de “Vista de redes”, se tendrá que comprobar que las direcciones IP del PLC estén debidamente configuradas para tener acceso al servidor WEB y para interactuar con el variador SINAMICS. Haciendo clic en el recuadro verde del archivo GSD del variador de frecuencia, se podrá ingresar a la sección de “Propiedades” para ajustar la dirección IP del variador. Para este caso se debe ingresar la dirección “192.168.0.7” con máscara de subred “255.255.255.0” (véase la Figura 2.28).

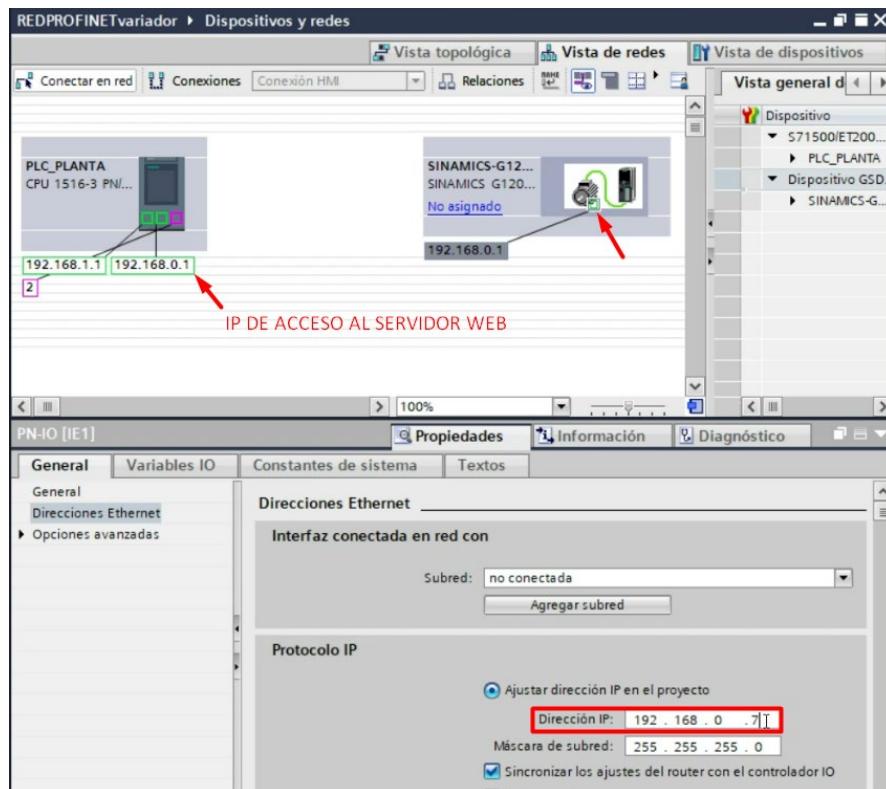


Figura 2.28 Asignación de dirección IP al variador SINAMICS.

Como un siguiente paso, se realizará la conexión del PLC con el variador. El resultado de este proceso se ilustra en la Figura 2.29.

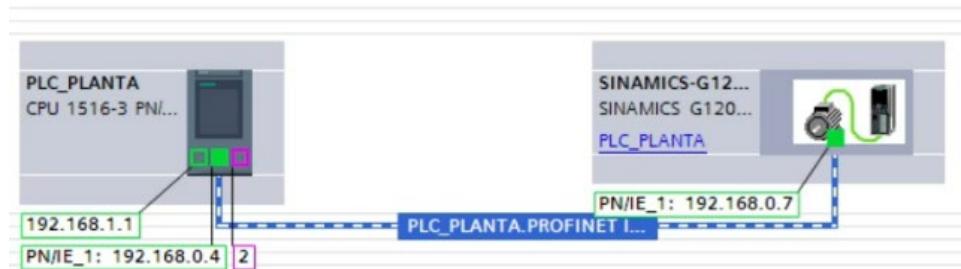


Figura 2.29 Conexión PROFINET entre el PLC SIEMENS y el variador SINAMICS.

También se deberá verificar que el variador SINAMICS, esté asignado con el nombre correcto, el cual fue mostrado en la sección “Acceso online” (véase la Figura 2.30). Para establecer conexión con el PLC se debe seguir los siguientes pasos. Como primer paso, haciendo clic en la conexión del PLC y el variador, seleccionamos la opción “Asignar nombre de dispositivo”.

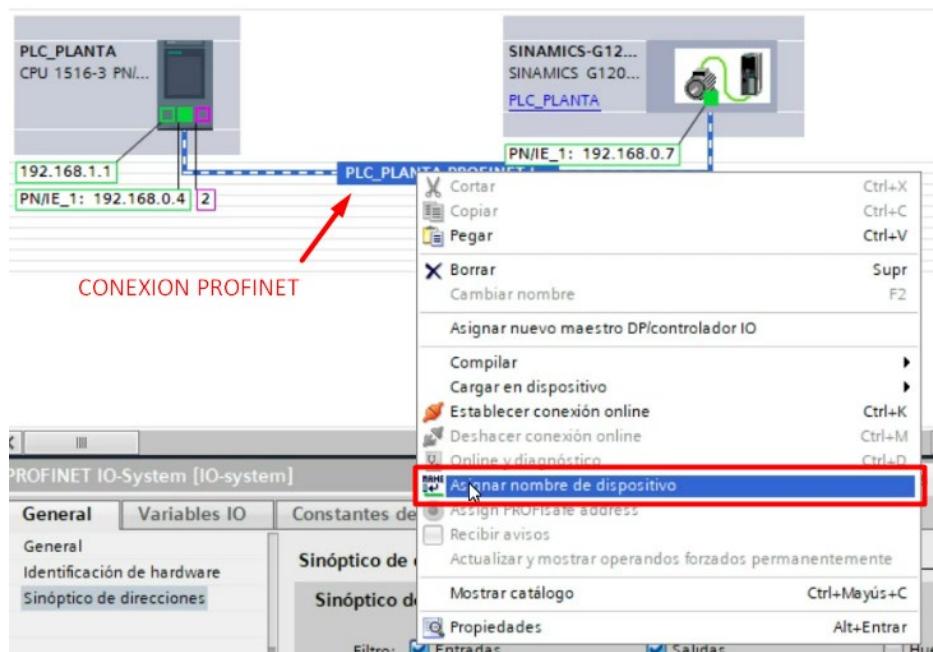


Figura 2.30 Asignación del nombre al variador SINAMICS.

Al presionar la opción antes mencionada, se abrirá una nueva ventana, donde se buscará el nombre del dispositivo PROFINET, que en este caso es “sinamics-g120sv-pn”, cuyo nombre fue detectado en la sección de “Accesos online”. Este proceso se ilustra de mejor manera en la Figura 2.31.

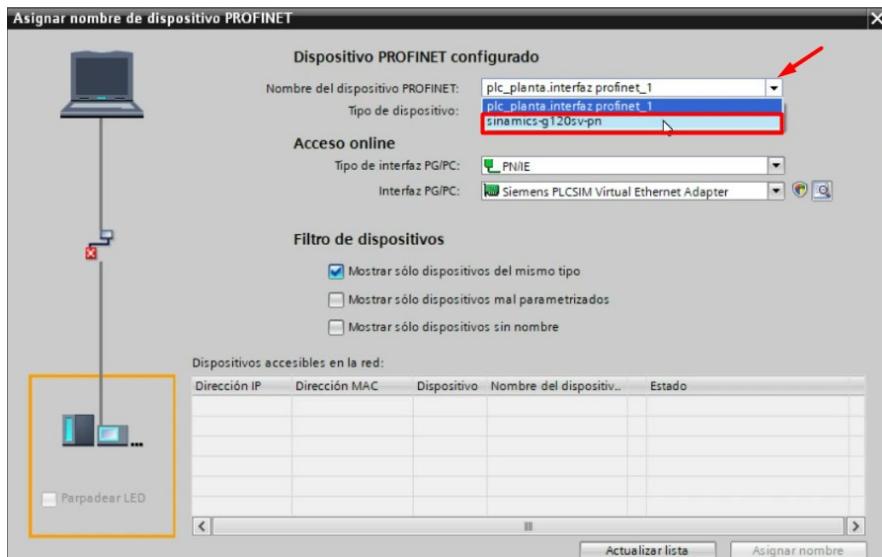


Figura 2.31 Búsqueda del variador vía PROFINET.

En la sección de “Dispositivos accesibles en la red”, se debe buscar el dispositivo utilizando la Interfaz PG/PC del computador. Para este efecto, se debe dar clic en la opción “Actualizar lista”. Como se observa en la Figura 2.32, al ejecutar este proceso, se ha detectado el variador con el nombre “sinamics-g120sv-pn”. Además, se visualiza la dirección IP asignada anteriormente.

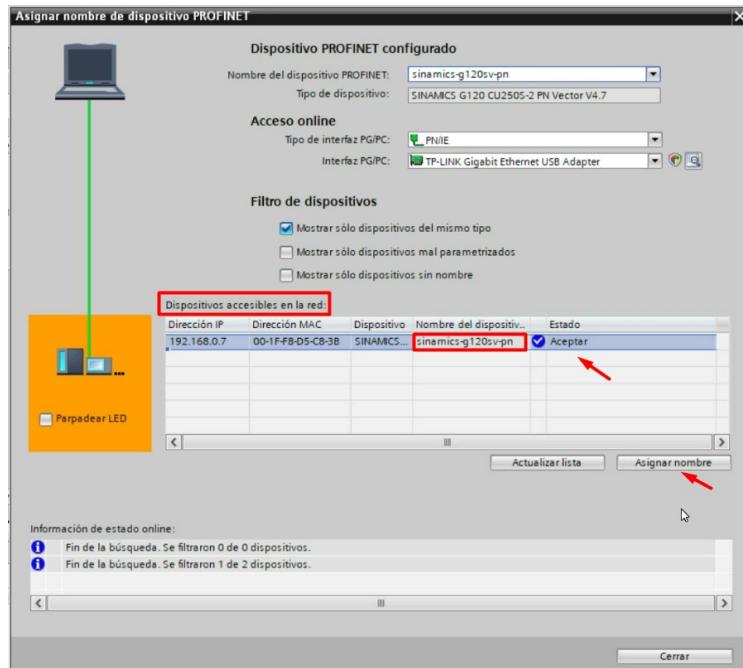


Figura 2.32 Detección del variador antes configurado.

Al seleccionar el variador encontrado, se activará la opción “Asignar nombre”. Seguidamente se debe dar clic sobre esta opción. La Figura 2.33 muestra que, al ejecutar el proceso antes mencionado, se ha detectado satisfactoriamente el variador con el que se trabajará. Como paso final, haciendo clic en “Cerrar”, finalizaremos la detección y conexión del PLC con el variador SINAMICS G120.

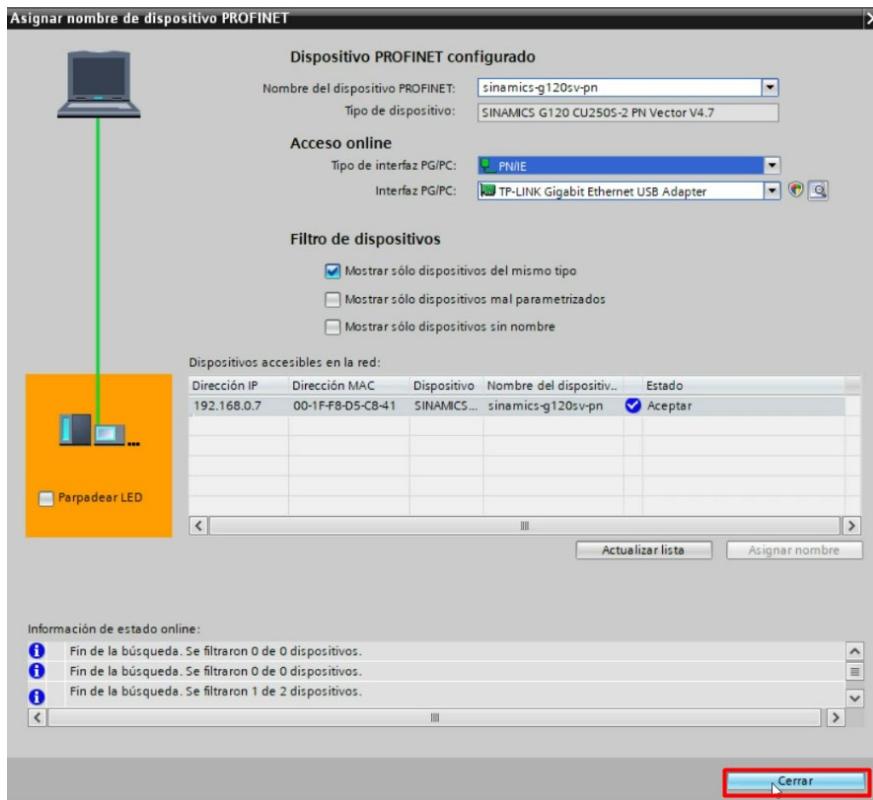


Figura 2.33 Variador SINAMICS G120 encontrado satisfactoriamente.

2.5 Configuración y uso de la librería "Library SINAMICS Extended" (LSINAExt)

Como se mencionó anteriormente, la comunicación entre el PLC y el variador de frecuencia se da por medio de telegramas. Existen diferentes telegramas que permitirán el intercambio de datos de manera cíclica entre estos dispositivos. Entre estos telegramas se puede mencionar; el telegrama estándar PROFIdrive (telegrama 1), el telegrama PROFIsafe (telegrama 30) y los telegramas libres “free” utilizados para comunicaciones específicas del usuario [12].

En este apartado se brindarán detalles sobre la configuración del bloque SINA_SPEED utilizando el telegrama 20, el cual ha sido seleccionado para este proyecto.

El telegrama de comunicación este compuesto por palabras (16 bits). Estas palabras son de dos tipos: palabras de “mando” y palabras de “estado”. Las palabras de mando envían instrucciones y consignas desde el PLC y las palabras de estado reflejan el “estado” del accionamiento. Dependiendo del tipo de telegrama, se envía un número distinto de valores reales, consignas, palabras de control o de estado. Existe un telegrama común, “telegrama 1”, que contiene las palabras de mando y estado, este telegrama es básico para el arranque del funcionamiento y comprobación del estado del variador de frecuencia [8][11].

2.5.1 Librería (LSINAExt)

Esta librería contiene bloques de función para el control cíclico sencillo de un variador SINAMICS. El intercambio de datos entre los PLCs SIMATIC y los accionamientos SINAMICS se lo realizará a través de los buses de campo PROFINET o PROFIBUS DP. Por consiguiente, esta librería es de gran ayuda para la aplicación a desarrollar.

Este bloque permite de una manera sencilla controlar el variador de frecuencia sin necesidad de conocer la estructura del telegrama. El bloque entrega parámetros de salida relacionados a los datos de proceso a manejar, además de estados y números de error respectivamente [12].

2.5.2 Bloque de función SINA_SPEED_TLG20

Para la presente aplicación se utilizará el bloque de función SINA_SPEED_TLG20 (FB38003). Al usar este bloque no es necesario conocer la estructura del telegrama, pues los parámetros de entrada/salida se los puede configurar de una manera sencilla. Los parámetros necesarios se transmiten de acuerdo con la especificación del telegrama y la respuesta se envía de forma estandarizada en los parámetros de salida [12]. La estructura de este bloque se muestra en la figura 2.34.

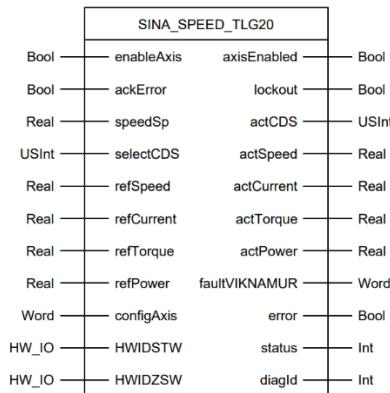


Figura 2.34 Bloque función SINA_SPEED_TLG20 [12].

Con la ayuda de este bloque de funciones, se puede ajustar la velocidad real del variador SINAMICS mediante el telegrama estándar 20, así como recibir parámetros como la corriente real, par real y potencia activa real, siempre y cuando se haya especificado de manera correcta el valor de referencia respectivo.

2.5.3 Telegrama estándar 20

El telegrama estándar 20 contiene de dos palabras de recepción y seis palabras de transmisión. Si se hace un análisis desde el punto de vista del accionamiento (variador de frecuencia), las palabras de “transmisión” (Status words) serán los datos de proceso a enviar, mientras que las palabras de “recepción” (Control words) son los datos de proceso recibidos [12]. La estructura de este telegrama se ilustra de mejor manera en la Tabla 2.2.

Tabla 2.2 Estructura del telegrama estándar 20 [12].

	PZD1	PZD2	PZD3	PZD4	PZD5	PZD6
(Receive telegram)	STW1	NSOLL_A				
(Transmit telegram)	ZSW1	NIST_A_GLATT	IAIST_GL ATT	MIST_GL ATT	P_IST_G_LATT	MELD_N_AMUR

En la Tabla 2.2, se muestran las distintas palabras de control, palabras de estado, valores reales, y consignas que el telegrama estándar 20 posee. La Tabla 2.3, explica el funcionamiento de cada PZD (Iniciales alemanas para “Datos de proceso”).

Tabla 2.3 Significado de palabras y valores del telegrama estándar 20.

Telegrama estándar 20			
Datos recibidos por el variador SINAMICS		Datos emitidos por el variador SINAMICS	
STW1	Palabra de control	ZSW1	Palabra de estado
NSOLL_A	Consigna de velocidad	NIST_A_GLATT	Valor real suavizado de velocidad
		IAIST_GLATT	Valor real suavizado de corriente
		MIST_GLATT	Valor real suavizado de par
		P_IST_GLATT	Valor real suavizado de potencia activa
		MELD_NAMUR	Palabra de fallo según la definición VIK-NAMUR

2.6 Bloque de función SINA_SPEED_TLG20 en TIA PORTAL V16:

Como se observa en la figura 2.35, el bloque de función tiene parámetros de entrada y salida. Por lo tanto, es necesario conocer cada uno de estos parámetros para pasar a la configuración de las variables y valores a leer y escribir por medio del uso del WEB SERVER del controlador SIMATIC 1516.

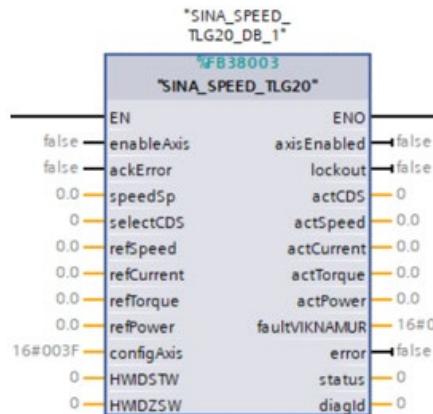


Figura 2.35 Bloque función SINA_SPEED_TLG20 en TIA PORTAL V16.

2.6.1 Parámetros de entrada

En la Tabla 2.4, se muestra el tipo de dato, su unidad de medida y una breve descripción de los parámetros del bloque, con el fin de que se pueda configurar de manera correcta los datos a enviar al variador de frecuencia SINAMICS.

Tabla 2.4 Parámetros de entrada del Bloque SINA_SPEED_TLG20 [12].

#	Nombre	Tipo de dato	Descripción
1	enableAxis	Bool	0 → 1 = Encendido del variador
2	ackError	Bool	0 → 1 = Detección de fallas
3	speedSp	Real	Consigna de velocidad [1/min]
4	selectCDS	USint	Selección de juego de datos de comando [p810] (0: CDS0, 1: CDS1)
5	refSpeed	Real	Velocidad de referencia [p2000] en [1/min]
6	refCurrent	Real	Corriente de referencia [p2002] en [Aeff]
7	refTorque	Real	Par de referencia [p2003] en [Nm]
8	refPower	Real	Potencia de referencia [p2005] en [kW]
9	configAxis	Word	Parámetro de entrada (Número Binario) para el control de bits de la palabra de control cuando no se dispone como un parámetro de entrada.
10	HWIDSTW	HW IO	ID de hardware de la ranura del punto de ajuste (setpoint)
11	HWIDZSW	HW IO	ID de hardware de la ranura de valor real

Los parámetros HWIDSTW y HWIDZSW hacen referencia a la identificación (nombre) de hardware del telegrama utilizado. El parámetro "configAxis" corresponde a los bits de la palabra de "control" 1 (STW1), donde dicho valor viene establecido con un valor estándar "16#003F". Por otra parte, al configurar el parámetro de entrada "enableAxis", será posible poner en funcionamiento al motor.

Para entender el valor hexadecimal de "16#003F" es necesario conocer la palabra de "mando" STW1, esto con el fin también de conocer las diferentes funciones de cada bit y su acción en el variador. La Tabla 2.5 muestra el significado de la palabra de mando antes mencionada.

Tabla 2.5 Bits de la palabra de mando STW1.

Palabra de mando STW1			
Bits (Word de 16 bits)	Valor/Significado	(Motor gira en sentido horario)	(Motor gira en sentido antihorario)
0	0= El variador se desconecta del motor	1	1
	0 → 1 = el convertidor este "Listo para el servicio"		
1	0= Se desconecta inmediatamente el motor	1	1
	1= Se conecta el motor		
2	0= Parada rápida del motor	1	1
	1=Se puede conectar el motor. Sin parada rápida		
3	0 =Bloquear servicio (Suprimir impulsos)	1	1
	1 = Sin bloquear servicio (habilitación de impulsos)		
4	0= La salida del generador de rampa se pone a cero/ Bloquear GdR	1	1
	1= Habilitación del generador de rampa/No bloquear GdR		
5	0= Detener GdR	1	1
	1= Habilitar GdR		
6	0= El convertidor frena el motor con el tiempo de deceleración del generador de rampa. /Bloquear consigna (setpoint).	0	1
	1= El motor acelera con el tiempo de aceleración hasta alcanzar la consigna. /Habilitar consigna (setpoint).		
7	0 → 1 = Se confirma un fallo	0	0
8	Siempre 0/ Valor reservado	0	0
9	Siempre 0/ Valor reservado	0	0
10	0= Ningún control por PLC	0	0
	1= Mando por PLC		
11	0= No existe inversión de consigna.	0	0
	1= Inversión de sentido/Se invierte la consigna en el convertidor.		
12	Siempre 0/ Valor reservado	0	0

13	0= No aumenta la consigna almacenada en el potenciómetro motorizado	0	0
	1= Aumenta la consigna		
14	0= No reduce la consigna almacenada en el potenciómetro motorizado	0	0
	1= reduce la consigna almacenada		
15	Siempre 0/ Valor reservado	0	0
	Valor del parámetro “configAxis”	16#003F	16#007F

Como un resumen final del parámetro "configAxis", éste servirá como un parámetro de entrada para poder establecer el sentido de giro del motor asíncrono a manejar, siempre y cuando se envié el valor binario en hexadecimal de manera correcta. Como se nota en la Tabla 2.5, en la columna 4, al final, el parámetro configAxis, toma un valor de "16#007F", esto ocurre cuando se ha asignado al bit 6 el valor de uno. Al asignar el valor de uno a este bit, se "invierte" el setpoint del variador, logrando así revertir el sentido de giro del motor, tal y como nos indica el manual [10]. Con ello se podrá lograr el sentido de giro del motor en un sentido antihorario. Por lo que finalmente, con estos dos valores "16#003F" y "16#007F", se podrá controlar el sentido de giro del motor.

2.6.2 Parámetros de salida

En la Tabla 2.6 se muestra el tipo de dato, su unidad de medida y una breve descripción de los parámetros de salida del bloque.

Tabla 2.6 Parámetros de salida del Bloque SINA_SPEED_TLG20 [12].

#	Nombre	Tipo de dato	Descripción
1	axisEnabled	Bool	1 = Convertidor en funcionamiento
2	lockout	Bool	1 = Encendido inhibido activo
3	actCDS	USInt	Conjunto de datos de comando activo (0 = CDS0, 1 = CDS1)
4	actSpeed	Real	Velocidad actual [1/min]
5	actCurrent	Real	Valor actual real en [Aeff]
6	actTorque	Real	Valor real del par en [Nm]
7	actPower	Real	Valor real de potencia activa alisado en [kW]
8	faultVIKNAMUR	Word	Palabra de error según la definición de VIK-NAMUR
9	error	Bool	1 = Error en el bloque de funciones/accionamiento
10	status	Int	La variable “Status” tendrá diferentes tipos de valores de estado del variador, si el valor es 7002, nos indicara que el bloque de función se está ejecutando. Si existe un valor de tipo 7xxx nos indicará una advertencia, finalmente si el valor es de tipo 8xxx, mostrará una descripción del error. Véase la guía principal para más detalles de cada valor de estado [8].
11	diagId	Int	Código de error de las funciones del sistema DPWR_DAT/DPRD_DAT

2.7 Configuración del bloque SINA_SPEED_TLG20

Luego de haber conocido el funcionamiento del bloque de función SINA_SPEED_TLG20, así como los respectivos parámetros de entrada y salida, se procede a insertar el bloque en la sección de “Bloques de Programa”. Específicamente, para este caso, se insertará este bloque en el programa principal (véase la Figura 2.36), donde también se configurará los bloques del SERVIDOR WEB del PLC Siemens.

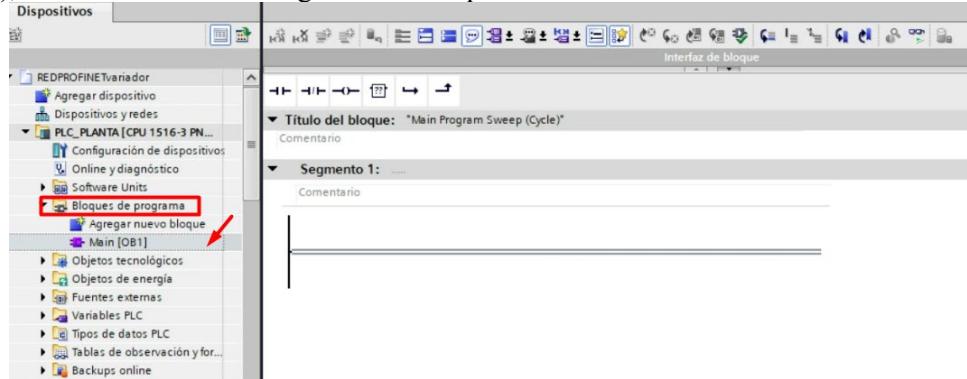


Figura 2.36 Programa principal en TIA PORTAL V16.

Como siguiente paso, se debe buscar en la sección de “Librerías”, la opción de “Librerías globales”. En esta parte se podrá encontrar el bloque SINA_SPEED_TLG20, dentro de “Plantillas maestras”. Dicho bloque debe ser arrastrado hacia el programa principal para configurar sus parámetros. El resultado de la ejecución de este proceso se ilustra en la Figura 2.37.

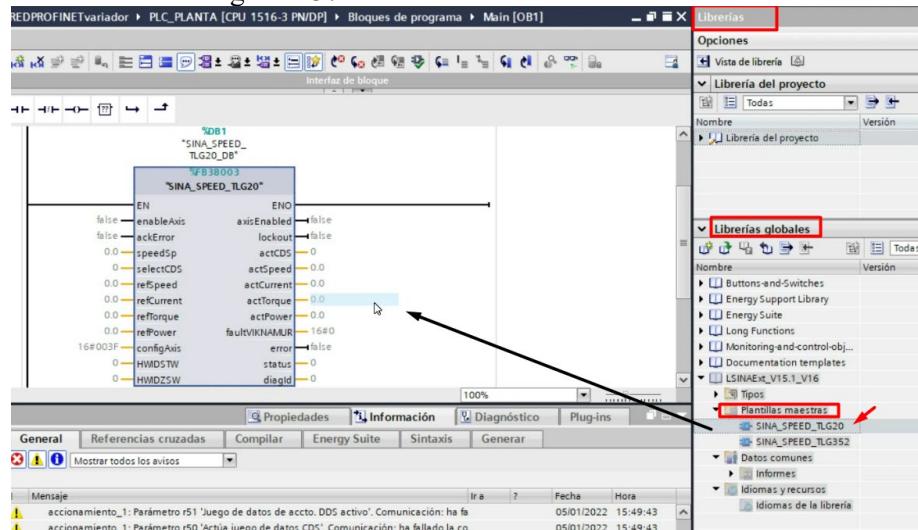


Figura 2.37 Ubicación del bloque SINA_SPEED_TLG20 en TIA PORTAL.

A continuación, se debe configurar los parámetros de entrada y salida con su respectivo tipo de dato. En esta sección se asignará los nombres a cada uno de los parámetros, los cuales serán usados para operaciones de lectura y escritura desde la aplicación web que se implementara en un capítulo posterior.

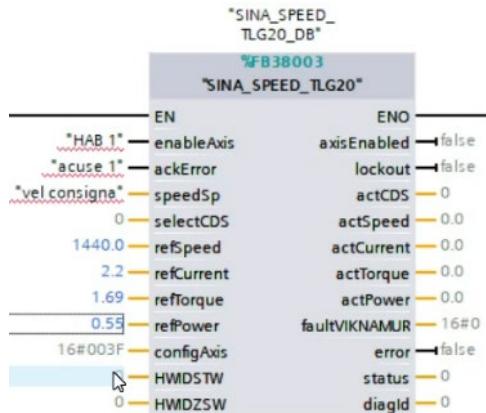


Figura 2.38 Configuración de parámetros de entrada del bloque SINA_SPEED_TLG20.

Como se observa en la figura 2.38, se han configurado los parámetros de entrada, además se ha ingresado los valores referenciales de velocidad, corriente, torque y potencia. Dichos valores están relacionados a la placa del motor a controlar. La figura 2.39 muestra los datos de la placa del motor a utilizar, el cual se conectará en triangulo.

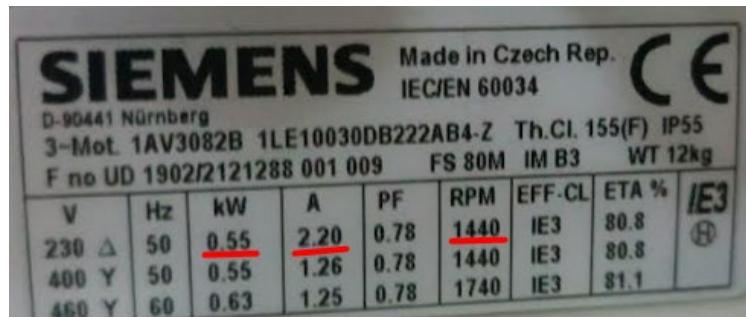


Figura 2.39 Placa del motor SIEMENS asíncrono a controlar.

A continuación, se debe configurar los parámetros HWIDSTW y HWIDZSW. Como se mencionó anteriormente, estos parámetros hacen referencia a la identificación de hardware del telegrama utilizado, por lo tanto, se seleccionará este parámetro dependiendo del variador que se esté utilizando. Para este caso, se utilizará el variador SINAMICS G120S PN-DO. La figura 2.40 muestra en detalle la configuración antes mencionada.

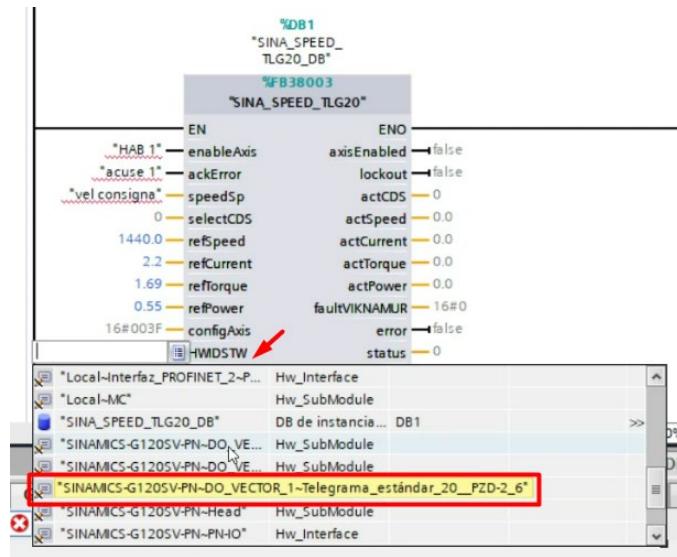


Figura 2.40 Configuración de los parámetros HWIDSTW y HWIDZSW del bloque función.

Finalmente, se configuran los parámetros de salida del bloque, para luego validar las variables con los nombres correspondientes, esto con el fin de asignar el tipo de dato al parámetro y una dirección específica del PLC (véase la Figura 2.41).

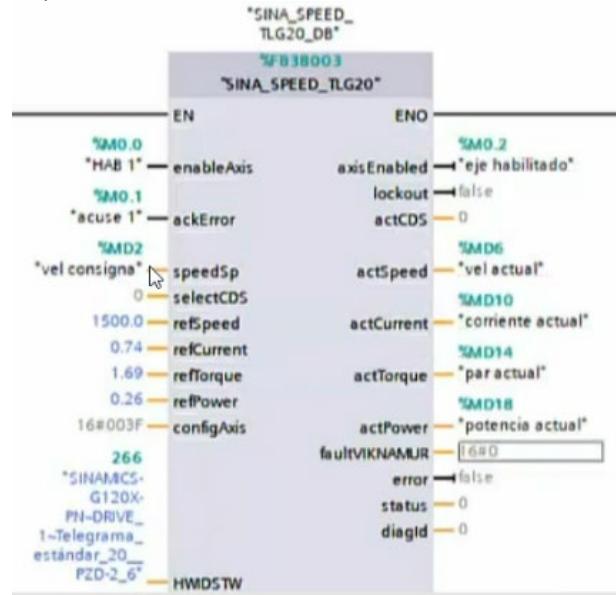


Figura 2.41 Configuración de parámetros de salida del bloque SINA_SPEED_TLG20.

Para finalizar este capítulo, en la Tabla 2.7 se muestra un resumen para conocer las variables y parámetros de lectura y escritura del accionamiento, esto con el propósito de construir el sistema de control y visualización del motor mediante el uso del servidor WEB.

Tabla 2.7 Parámetros de entrada y salida a monitorearse con el uso del Servidor WEB del PLC.

Parámetros de interés del bloque función SINA SPEED TLG20			
Parámetros de escritura en el variador desde el SERVIDOR WEB		Parámetros de lectura en el variador desde el SERVIDOR WEB	
Nombre de parámetro	Función	Nombre de parámetro	Función
“HAB 1”	Enciende el variador posibilitando el giro del eje del motor	“vel actual”	Parámetro que devuelve la velocidad actual del motor
“vel consigna”	Parámetro variable el cual controla la velocidad del motor	“corriente actual”	Devuelve la corriente actual del motor
“configAxis”	Establece el sentido de giro del motor.	“par actual”	Devuelve el par actual del motor
		“potencia actual”	Devuelve la potencia activa actual del motor

CAPÍTULO 3

Creación de la aplicación web mediante el uso de herramientas de lenguajes de programación basados en HTML y JavaScript.

En este capítulo, se hace un análisis y descripción de como mediante el uso de herramientas proporcionadas por lenguajes como HTML y JavaScript, se puede monitorear variables de un proceso industrial desde una aplicación WEB. Esta aplicación contiene elementos de control que permiten controlar el encendido y apagado del motor y con la incrustación de imágenes en el documento web, se podrá visualizar de una manera intuitiva el proceso que se está controlando. Cada uno de los controles o “formularios” son elementos proporcionados por HTML y mediante funciones de JavaScript, se podrá realizar operaciones de lectura y escritura en las variables del proceso del sistema de velocidad implementado.

Entre otras de las funciones de la aplicación WEB que se implementará, se podrá visualizar los datos proporcionados por el variador en cuadros de textos, identificando cada variable por medio de etiquetas que indiquen el tipo de magnitud y unidad leída.

Con lo antes mencionado, se pretende mostrar una nueva alternativa al uso de dispositivos HMI (Human Machine Interface) para laborares de monitoreo, mediante la creación de una aplicación web, desde la cual podremos operar el proceso industrial. La aplicación presenta una interfaz gráfica con la que se podrá obtener datos de un proceso de los equipos e instalaciones de una planta industrial, para su respectivo análisis. Además, otra de las funciones de la aplicación web es permitir direccionar a la aplicación web de Laravel que se encarga de la gestión y descarga de datos que son almacenados en una base de datos. Esta aplicación será abordada en el capítulo 4.

Diagrama general del funcionamiento de la aplicación

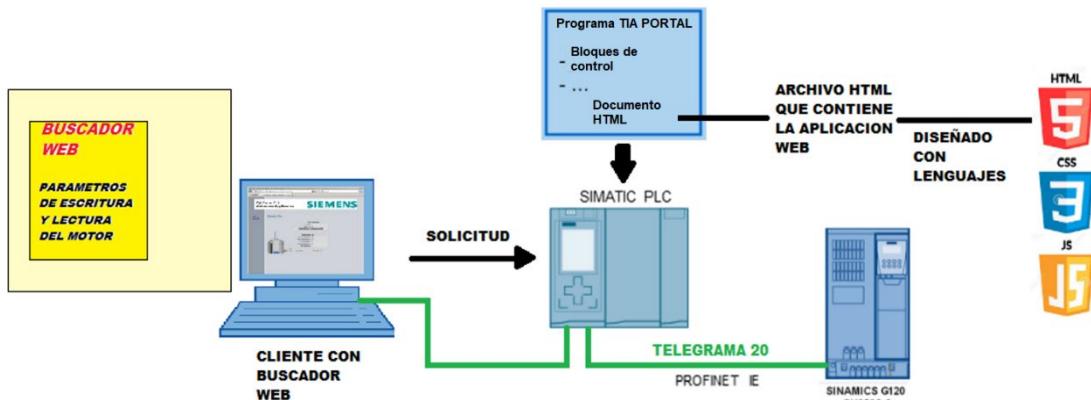


Figura 3.1 Acceso a la aplicación web desde un dispositivo de un cliente.

En la figura 3.1, se muestra una idea global del funcionamiento del proceso, donde un usuario o cliente accede al documento html que se implementara en el servidor web del PLC. Este archivo, contiene en sí mismo, la aplicación WEB que se encarga de recibir y escribir los datos de las diferentes variables configuradas en el bloque función.

Si se observa en la parte derecha de la figura 3.1, esta aplicación web estará diseñada con las herramientas de los lenguajes HTML, CSS y JavaScript y una vez terminada su implementación, se procederá a configurarla en el PLC Siemens.

Finalmente, en una etapa de pruebas, esta aplicación podrá ser ejecutada, cuando un usuario desde un buscador web realice una “solicitud” vía http, acceda a los parámetros de lectura del motor, como; velocidad actual, corriente actual, potencia activa, así como otros parámetros de salida del bloque SINAMICS. Además, el usuario podrá manipular los diferentes parámetros de entrada como el encendido del motor, configuración del sentido de giro y variar la velocidad del motor.

3.1 Normativas a considerar

El propósito de la aplicación es presentar de manera adecuada a un operador la información que necesita para mantener el control del proceso de forma eficiente. Si se presenta de una manera correcta la información de las variables y datos del variador de frecuencia SINAMICS, ofrece la posibilidad de tener una respuesta para reconocer y actuar ante una situación anormal o de peligro para el funcionamiento del proceso industrial.

Para crear la aplicación web bajo un diseño óptimo para el monitoreo del proceso, se tomará en cuenta consideraciones establecidas por estándares ANSI/ISA-101.01 (“*Human Machine Interfaces for Process Automation*”) y ANSI/ISA-18.2 (“*Management of Alarm Systems for the Process Industries*”) para el diseño de interfaces HMI.

A continuación, se hace una breve descripción de los elementos que componen un HMI y posteriormente, se presenta algunas normas de diseño recopiladas del manual “*Process HMI Style Guide*” que recoge información de los estándares antes mencionados [14]:

- **Jerarquía de pantalla:** estructuración de la información en pantallas y definición de la jerarquía de las pantallas.
- **Navegación de pantalla:** método para navegar entre las diferentes pantallas.
- **Disposición de la pantalla:** distribución de imágenes e información en la pantalla.
- **Contenido de la pantalla:** incrustación de imágenes estáticas y dinámicas colocadas en el HMI, tales como datos numéricos, bomba, válvulas, etc.
- **Representación y gestión de alarmas:** presentación de alarmas y como el usuario puede manipularlas.
- **Seguridad:** acceso a personas autorizadas al contenido y control de funciones.
- **Rendimiento de pantalla:** la interacción de los usuarios, el tiempo de respuesta a la llamada inicial, cambio y recepción de datos.

3.2 Normas y estándares de diseño para HMI

Convenciones de color

El uso del color en un HMI no debe ser un atributo que debe ser usado en exceso. El color debe ser una herramienta limitada. El color tiene un valor y será destinado a situaciones poco usuales o anormales, además para diferenciar textos estáticos, campos de entrada o datos en vivo o tiempo real. Por ello se presentan algunas pautas de color para elementos de un HMI:

- Alarms: se usará colores intensos y brillantes, además no destinarlos a representar otros tipos de elementos.
- Datos en vivo: se recomienda usar colores con poca intensidad y fríos, ya que distraen menos diferenciando de la información estática. Se puede usar colores como el azul o verde oscuro.
- Fondo de pantalla: usar colores de fondo no saturados, como el gris claro, el cual no dificulta la visibilidad de los demás elementos que presentan color.
- No usar colores degradados.
- Interior de un elemento estático: el color de relleno será el mismo del color de fondo de pantalla.
- La representación de un estado de un equipo o elemento no debe quedarse o limitarse a usar un color, se puede añadir características como relleno, forma o texto para indicar el estado actual.
- Cuando se use un color, este deberá “contrastar” con el fondo de la aplicación, con la finalidad de no causar fatiga visual. Considerar factores externos como la iluminación del entorno, que puede causar poca distinción del color de los elementos.

A continuación, la tabla 3.1 contiene algunas recomendaciones en cuanto a colores que deben usarse para diferentes tipos de elementos de un HMI:

Tabla 3.1 Colores designados para diferentes elementos de un HMI [14].

Elemento	Color	Código (RGB/Hex)
Color para la pantalla		
Fondo de pantalla	light Gray 224	R224 G224 B224 #E0E0E0
Fondo de un panel de pestanas	light Gray 224	R224 G224 B224 #E0E0E0
Colores para objetos estáticos		
Fondo de un título	Dark Gray 63	R063 G063 B063 #3F3F3F
fondo de un grupo de encabezados	Dark Gray 63	R063 G063 B063 #3F3F3F
Líneas de proceso y conectores	Gray 160	R160 G160 B164 #D8D8D8
Cuadro de agrupación	Light Gray 232	R232 G232 B232 #E8E8E8
Color para notificaciones		
Alarma de prioridad baja	Magenta	R145 G106 B173 #916AAD
Alarma de prioridad media	Yellow	R245 G225 B027 #F5E11B
Alarma de alta prioridad	Orange	R236 G134 B041 #EC8629
Alarma de prioridad urgente	Red	R226 G032 B040 #E22028
Error de programa/error de configuración	Black	R000 G000 B000 #000000
Colores para el estado de un elemento		
Apagado/des energizado/inactivo/detenido/cerrado	Gray	R128 G128 B128 #808080
Encendido / Energizado / Activo / Cerrado	Off White	R240 G240 B240 #F0F0F0
Deshabilitado/Fuera de servicio	Gray	R128 G128 B128 #808080
Transición (arranque, parada, aceleración, Deceleración, Apertura, Cierre)	Light Blue	R147 G194 B228 #93C2E4
Color para elementos de entrada de datos		
Etiquetas	Dark Gray 63	R063 G063 B063 #3F3F3F
Unidades de ingeniería (etiqueta)	Light Gray 91	R145 G145 B145 #919191
Fondo: Campo de entrada, casilla de verificación, botón circular (se permiten ediciones)	White	R255 G255 B255 #FFFFFF
Fondo: Campo de entrada, casilla de	Light Gray	R224 G224 B224 #E0E0E0

verificación, botón circular (no se permiten ediciones)		
Color de visualización de datos dinámicos		
Etiquetas	Dark Gray 63	█ R063 G063 B063 #3F3F3F
Unidades de ingeniería (etiqueta)	Light Gray 91	█ R145 G145 B145 #919191
Datos (Fuente)	Blue	█ R071 G092 B167 #475CA7
Borde de diagrama de datos	Light Gray	█ R192 G192 B192 # C0C0C0
Botones de navegación		
Relleno	Light Gray 198	█ R198 G198 B198 #C6C6C6
Contorno	Gray 170	█ R170 G170 B170 #AAAAAA
Etiqueta	Dark Gray 63	█ R063 G063 B063 #3F3F3F

Cuadros de agrupación

Se debe usar un cuadro de agrupación para encapsular elementos que tengan relación alguna con el fin de no causar un desorden visual en las pantallas. Como una ayuda adicional, se puede insertar un encabezado que indique el tema que representa el grupo. Se puede usar un color gris claro 232 para el cuadro. En la figura 3.2, se puede observar un ejemplo de cuadro de agrupación.



Figura 3.2 Cuadro de agrupación en pantalla HMI [14].

Representación del equipo de proceso

Los diferentes equipos del proceso pueden aparecer de forma estática y dinámica para mostrar el contexto de su función o el estado del elemento controlado. Para ello, se empleará figuras basadas en estándares ANSI/ISA-5.1 e ISA-5.5.

- *Objetos de proceso dinámico*

Estos elementos muestran el estado de un objeto y su estado estará en función de las capacidades del objeto, por ejemplo, el estado abierto-cerrado de una válvula o el estado de un motor; Arranque-Detenido. Un objeto dinámico también mostrara notificaciones con respecto a alguna situación anormal mostrando como por ejemplo una alarma de alerta. Estos podrán aparecer generalmente a un lado izquierdo o derecho de un objeto de proceso.

La figura 3.3 ilustra algunos objetos de procesos, en este caso, estados de una válvula.

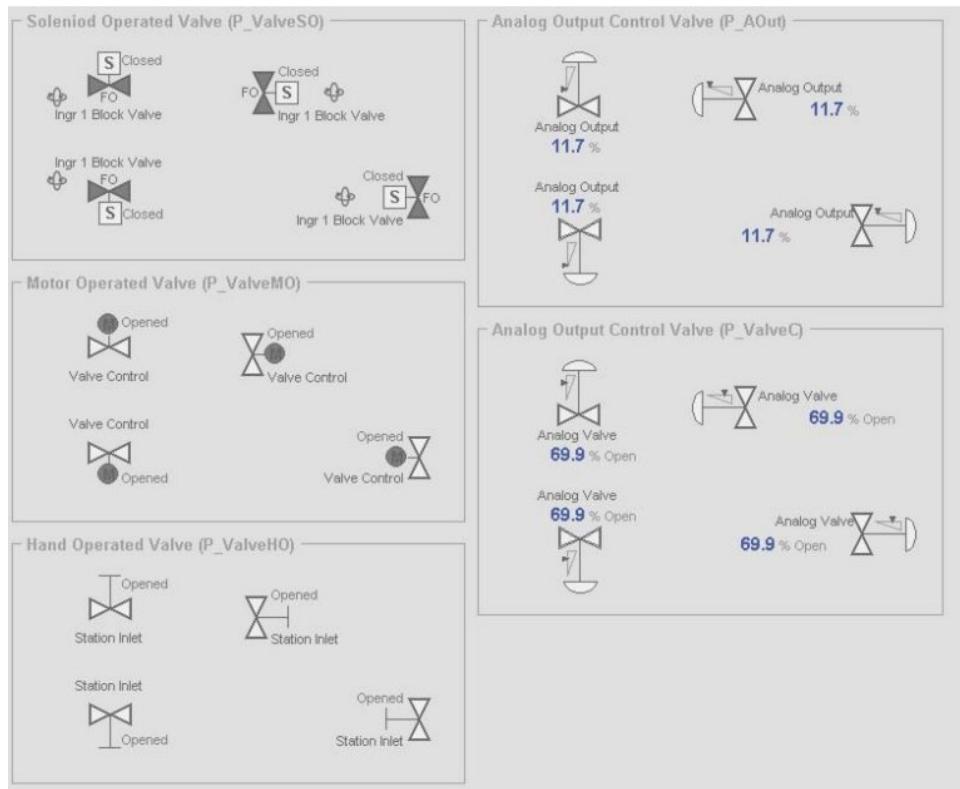


Figura 3.3 Objetos dinámicos que representan diferentes estados de válvulas [14].

Como otro aspecto a considerar, los colores de estado estarán también bajos las normativas de los estándares ISA.

- *Objetos de proceso estáticos*

Estos objetos no contienen datos dentro de ellos, pero si entregan una comprensión al usuario del dato presentado, como, por ejemplo; líneas de proceso o tanques de lleno. Estos objetos es recomendable darles un color gris oscuro para su delineado y relleno del interior.

Algunos consejos a tomar en cuenta con respecto a objetos estáticos:

- Evitar usar imágenes tridimensionales, usar demasiados colores o degradados.
- Desarrollar formas y tamaños estándar para los instrumentos, bombas o recipientes.
- El tamaño del objeto deberá relacionarse con la importancia de este en el proceso.
- Revisar las normativas ISA-5.5 para el uso de gráficos simbólicos estándar.

Tipos y tamaños de fuente

A continuación, se muestran algunos consejos respecto a los textos en pantalla, los cuales deberán ser claros para la compresión durante la navegación en la pantalla:

- Utilizar el tipo de fuente Sans Serif por su legibilidad.
- El tamaño de fuente, generalmente usado es de 10 puntos, y usado para datos recibidos, encabezados, leyendas o títulos. Etiquetas o anotaciones pueden ser de un menor tamaño.
- Un aspecto a tomar en cuenta, es la relación del tamaño de fuente y el tamaño de pantalla, pues citando como un ejemplo concreto a un monitor montado en una pared y alejado del personal, el tamaño de fuente deberá ser adecuado para que los operarios logren una visibilidad optima de la pantalla.

Alineación de datos

Datos de tipo numérico deberán estar alineados y justificados a la derecha como una norma general. Además, unidades de ingeniería estarán justificados a la izquierda. Solo se justificarán a la izquierda, datos que no tienen unidad de ingeniería y no están relacionados con otros datos. Se utilizará una justificación “centrada”, si los datos aparecen agrupados en un diagrama, botón o tabla.

A continuación, la figura 3.4 muestra un ejemplo del proceso antes mencionado:

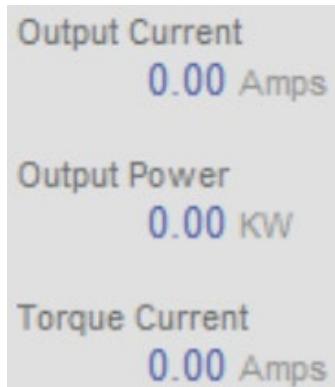


Figura 3.4 Alineación de unidades y datos de ingeniería [14].

Texto estático

Este tipo de texto no cambiara durante el proceso de operación. Con lo antes mencionado, se tomarán consideraciones como:

- Elegir un tamaño de fuente según jerarquía y considerando su visualización.
- Minimizar abreviaturas para no presentar dificultad a la hora de su traducción.
- Considerar un espacio adicional para la traducción del texto a otros idiomas.

Texto de la barra de título

En caso de pantallas emergentes tendrá un título en una barra de texto, el cual describa el contenido mostrado.

Títulos

Este describe el contenido mostrado en pantalla. Se recomienda utilizar fuentes en negrita, tamaño de fuente grande, además de procurar ubicarlo en la misma posición en cada una de las pantallas que se utilicen para lograr identificarlo.

Encabezados de grupo

Describen un grupo de controles o conjunto de datos en pantalla. Como una característica propia, estos se justifican a la izquierda y estar en negrita.

Etiquetas

Describen cualquier texto estático como títulos de pantalla, navegación, datos, etc. Estas se justificarán a la izquierda y los datos de proceso serán justificados a la derecha debajo de su respectiva etiqueta. Además, su uso será medido evitando la distracción de los usuarios.

Para un correcto formato de etiquetas, se muestran una serie de parámetros a seguir:

- No usar un texto negro, usar un color gris oscuro.
- Utilizar un tipo de fuente Sans Serif
- El uso de letras mayúsculas puede ser destinado para la descripción de títulos, palabras aisladas, equipos, etc.
- Asegurarse y considerar el entorno para definir un tamaño, fuente y color a utilizarse en las etiquetas. El uso de etiquetas puede ser probado con el fin de que los operarios se adapten y verifiquen el uso de etiquetas para el control y monitoreo del proceso.

Datos dinámicos

Estos gráficos deberán proporcionar información para recibir los medios adecuados para una toma de decisiones, es por ello que un “vistazo rápido”, ayudara al operador a captar los datos del proceso y tomar las respectivas acciones del caso.

A continuación, se muestran de manera general directrices a tomar en cuenta para datos dinámicos:

- Se definirá una tasa de muestreo de los datos en base a la “velocidad” en la que el operario deberá reaccionar dependiendo del caso específico.
- Proporcionar un “contexto” para los datos colocándolos cerca de una etiqueta u objeto estático.
- Dimensionar el tamaño del objeto que representa los datos dinámicos en base al tamaño de pantalla y jerarquía de este.
- Utilizar una fuente de menor tamaño a la de los datos dinámicos para unidades de ingeniería y etiquetas.
- El color de fondo de los datos dinámicos será el mismo del color de fondo de pantalla, salvo excepciones como: datos en un diagrama que utilizan un borde de acuerdo a un estándar del diagrama, notificaciones o enumeraciones de un estado.

Datos de texto dinámico

Estos se usarán cuando sean necesarios dado que son difíciles de determinar a partir de una presentación visual y necesita de un enfoque elevado.

Algunas consideraciones a tomar en cuenta:

- Ubicar las etiquetas en la parte de arriba o a la izquierda de los datos dinámicos.
- casos especiales; las unidades de ingeniería cuando se muestran con datos numéricos dinámicos seguirán el mismo estándar de unidades de ingeniería estáticas.

Datos numéricos dinámicos

Se utilizarán cuando sean necesarios. Algunos aspectos a tomar en cuenta al presentar datos numéricos dinámicos:

- Las unidades de ingeniería deberán aparecer debajo o a la derecha de los datos dinámicos. A si mismo dichos datos deberán tener su respectiva unidad de ingeniería.
- Valores enteros no llevarán punto decimal y su número de cifras significativas y lugares decimales estarán de acuerdo a la exactitud, precisión y rango de los datos.
- Se determinará una dimensión del dato numérico en base a la precisión de la medición.

Tendencias

Este grafico permite mostrar valores a lo largo del tiempo, por lo que se puede usar habitualmente para comparar valores relacionados o similares. Es de gran ayuda, permitiendo predecir estados futuros para proceder de mejor manera en una toma de decisiones. Además, con la información pasada que proporciona la gráfica permite establecer una mejor metodología en la resolución de un problema. Como una desventaja de este recurso es su espacio que puede llegar a ser algo grande. La figura 3.5 muestra un ejemplo de un gráfico “Tendencia”.

Un gráfico de tendencias tiene componentes como:

- Eje Y
- Eje X
- Subtitulo con su unidad de ingeniería
- Límite máximo y mínimo del eje Y
- Leyenda
- Valor Numérico
- Zona/punto de alarma ALTA
- Zona/punto de alarma BAJA
- Setpoint
- Unidades de ingeniería

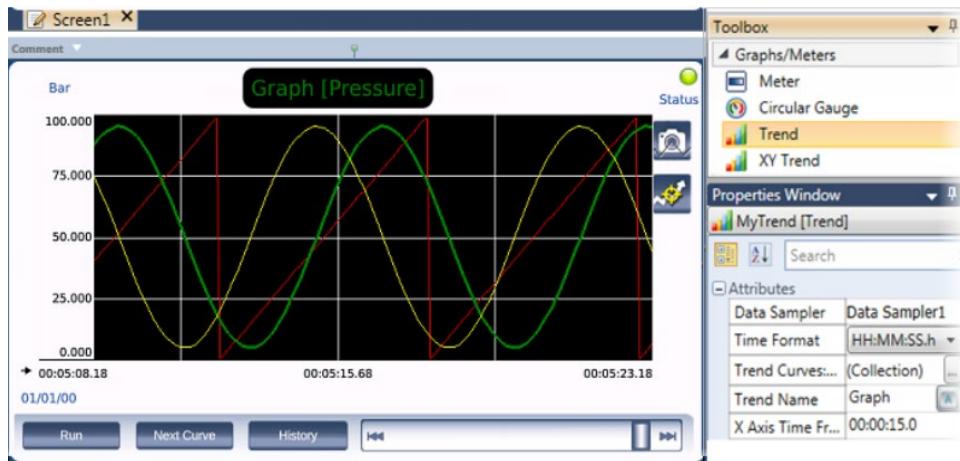


Figura 3.5 Tendencia en una pantalla HMI.[15]

Iconos

La figura 3.6, muestra algunos iconos estándar para pantallas HMI:

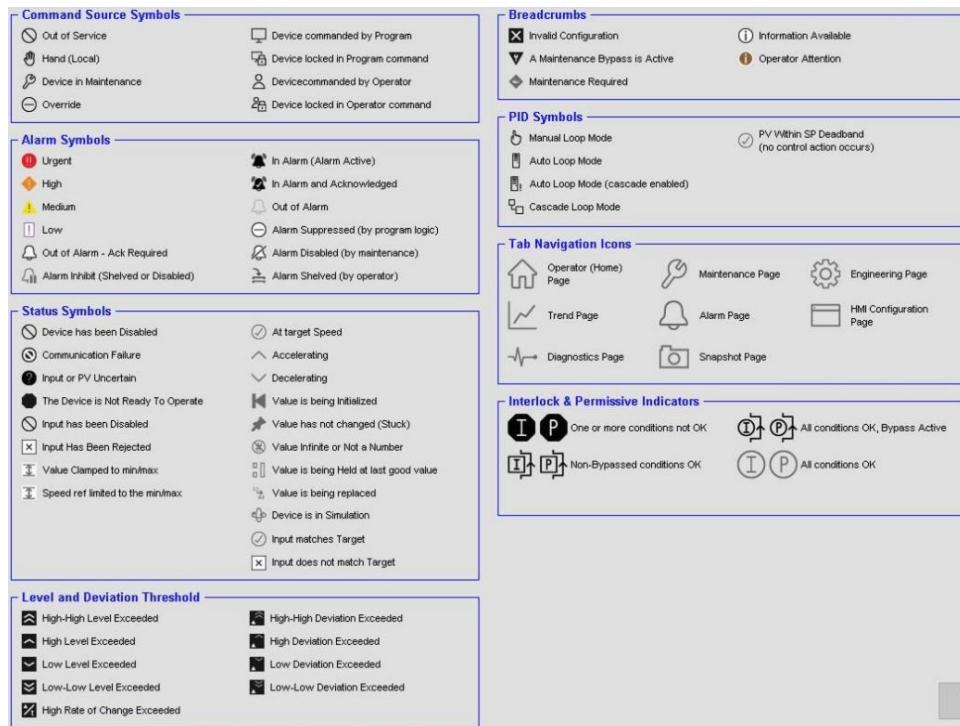


Figura 3.6 Iconos estándar para HMI [14].

Con respecto al tamaño de los iconos dependerá del caso, siendo recomendable un tamaño de 32 x 32 píxeles para iconos en botones y 16 x 16 píxeles para iconos de información. Además, el color estará en función de los colores estándar establecidas según cada caso lo indica las normas ANSI.

Controles de entrada

Los controles de entrada deben estar en función de los objetivos y tareas de los operarios, por lo que se deben determinar solamente controles necesarios. Una característica de los controles es la retroalimentación, por ello, la interfaz debe proporcionar una manera en la que el usuario sabe de la respuesta que provocó su control en el sistema.

Tamaño de controles de entrada

Cuando se habla de controles es importante definir el tamaño del control pues este ayudará a la interacción del usuario con la pantalla, y la zona de impacto como en casos de controles táctiles, el tamaño será un

aspecto a considerar. Por tanto, a continuación, se muestran algunas recomendaciones para algunos tipos controles, indicando un tamaño adecuado para los mismos:

- Botones de comando: este operador da como resultado un cambio de estado booleano. Se recomienda un tamaño de 40 x40 px. Entre estos botones se debe mantener una distancia para evitar acciones no deseadas. El texto o el ícono incrustado en el botón, deberá indicar el comando que se dará inicio, no el estado actual, además de estar justificado al centro.
- Botones de navegación: (Tamaño- 35 x 35 px). Estos ayudaran al operario a dirigirse a otras páginas. Evitar un espacio reducido entre estos tipos de botones para evitar una navegación no deseada. El texto del botón hará una aclaración del destino de la navegación. Estos poseerán una apariencia diferente a los botones de comando.
- Campos de entrada: (20 píxeles de alto y el ancho definido por el parámetro de entrada). El objeto deberá ser capaz de reconocer si el parámetro de entrada esta fuera de rango. Este campo deberá poseer un contexto, por tanto, se lo puede ubicar cerca de un objeto o etiqueta estática. Si el valor es de tipo numérico entero, no se incluirá el punto decimal.

Alermas

Permiten al usuario alertar o notificar una situación anormal que ocurre en el sistema. Según el estándar ANSI/ISA 18.2, define una alarma como un medio visible o audible que indica al usuario u operario, un mal funcionamiento de un proceso, equipo o condición anormal que requiere de una solución o respuesta. Una característica importante de una alarma, es su prioridad, donde cada alarma, se la diferenciaría por parámetros como; color, forma e ícono, este último siendo un recurso visual para llamar la atención del usuario. Su ubicación es importante, ya que se deben ubicar en una zona donde proporcionen un contexto útil y llamen la atención del operario. Por tanto, se las pueden incluir en las siguientes áreas: En componentes de la pantalla, sección de resumen de alarmas, barra de navegación, etc.

3.3 Conceptos Generales de diseño web

Principios básicos de páginas web

Para la creación de páginas web usando las herramientas de HTML y JavaScript, es necesario realizar un análisis previo acerca del lenguaje HTML, y demás lenguajes de codificación de aplicaciones o páginas web.

Lenguaje de marcado

Conocido como el lenguaje que permite codificar un documento web, donde junto con el texto, se incorporan etiquetas o marcas. Dichas marcas muestran el texto del documento, una vez que son interpretadas por el buscador web. El lenguaje de marcado o marcas más conocido es el HTML [16].

○ Lenguaje HTML

HTML (HyperText Markup Language), denominado como un lenguaje de marcado, el cual describe la página para el navegador web, es decir interpreta el código fuente de la página y muestra esa interpretación en el dispositivo del usuario [17].

Este lenguaje se encarga de estructurar la página, organizar su contenido y compartir la información. Sin embargo, debido a sus limitadas funciones, surgieron otros lenguajes que implementaban características y lograban sofisticadas aplicaciones, dando una nueva experiencia en la web. Algunos de estos lenguajes fueron Java y Flash [18].

○ Lenguaje CSS

Este lenguaje no se vincula de manera directa con HTML, pues nace como un complemento para las exigencias de diseño y funcionalidad que exigía la web debido a las limitaciones y complejidad que presentaba el lenguaje HTML. Particularmente, este lenguaje se centraría en el diseño y presentación del documento web, mientras que HTML se encargaría de la parte de la estructura. Por ende, las páginas web, trabajarían conjuntamente con HTML y CSS, donde este último agregaba estilos visuales como color, fondo, bordes a los distintos elementos del documento [18].

El formato de CSS, sigue la siguiente estructura; Selector {Propiedad: valor}, donde el selector puede poseer varios parámetros.

- *Lenguaje JavaScript*

Se considera como un lenguaje pilar y de complemento en la creación de páginas web. Debido a su rendimiento en procesamiento de código máquina, es la mejor opción para la web y los navegadores. Este lenguaje es ampliamente usado, en servidores, aplicaciones de escritorio y dispositivos móviles. Por ello, JavaScript agrega interactividad a un sitio web, y la incorporación de las API (Interfaces de programación de aplicaciones), permite incorporar librerías para crear gráficas o elementos gráficos, motores 3D para video juegos o una interfaz que acceda a una base de datos, etc [18].

3.4 Creación de páginas web:

El principal componente para acceder al servidor web de monitoreo, es la creación de la página o aplicación web, la cual estará diseñada bajo las distintas herramientas que ofrece HTML y JavaScript. Para la elaboración de una página web es necesario una interfaz que asocia los elementos de la página con “etiquetas”, dado que la página web está escrita en HTML y este es conocido como lenguaje de etiquetas o marcado. Estas herramientas se llaman “*Web authoring tools*” que ayudan generando de manera automática el código de cada elemento de la página. Algunos ejemplos de programas son Microsoft Front Page y Dreamweaver, de Adobe o Visual Code Studio, etc. [19].

3.4.1 Estructura y elementos de una página web

Un documento o página web, posee:

- **Encabezado:** El encabezado o “head section”, posee información general como título, información acerca de la temática del documento, datos del formato de página y scripts para crear interacción con el usuario.
- **Cuerpo:** Cuerpo o “body section”, en dicha parte se ubica el texto con la información y demás elementos que el autor muestra al lector [19].

Comúnmente en páginas web se utilizan “Web page headers”, que son encabezamientos que dividen el documento en distintas secciones. Entre otros elementos utilizados, se encuentran los “style sheets”, conocidos como cascading style sheets (CSS), los cuales son archivos encargados del formato de texto del documento web. También se pueden agregar elementos de audio, video o gráficos estáticos o dinámicos, pero que requiere de un lenguaje dinámico conocido DHTML (Dynamic HTML), que permite incorporar animaciones dando más interacción a la página y que no se torne aburrida [19].

3.4.2 Elementos HTML (etiquetas)

Las denominadas etiquetas, permiten identificar y estructurar las diferentes partes o secciones de un documento web, por lo tanto, un archivo HTML, contiene “elementos HTML”, situados entre etiquetas (pares de etiquetas). Como una característica general, los elementos HTML, están enmarcados al inicio con el símbolo “<” y de la misma manera al final con “</” [20].

Etiquetas típicas de HTML

Existen un conjunto general de etiquetas que permiten estructurar información, la tabla 3.2 detalla el funcionamiento de algunas etiquetas, las cuales han sido utilizadas en el desarrollo de la aplicación WEB [21]:

Tabla 3.2 Etiquetas comunes en HTML.

Representación	Función
<code><head> </head></code>	Proporciona información general acerca del archivo HTML, estos incluyen título, enlaces a hojas de estilos (archivos CSS) y scripts.
<code><title> </title></code>	Define un título que aparecerá en la pestaña de la página del buscador web.
<code><link rel= href= ></code>	Define la relación entre un archivo externo y el documento html. Generalmente se usa para enlazar archivos css.
<code><script> </script></code>	Inserta o hace una referencia a un código JavaScript.
<code><style> </style></code>	Contiene información respecto al “estilo” de un elemento o del documento html.
<code><body></body></code>	Esta sección muestra el cuerpo del contenido del documento HTML, indicando donde inicia la parte visible del contenido a mostrar.
<code><form></form></code>	Contiene controles interactivos para el usuario, con los que puede enviar información de regreso a un servidor.
<code><div>...</div></code>	Agrupa elementos HTML, estructurando el documento en distintas secciones. No tiene ningún efecto en el contenido, pues esto estará determinado por atributos como “class” o “id”.
<code><h1> </h1>.....<h6> </h6></code>	Representan los seis niveles de encabezados disponibles en HTML, siendo <code><h1></code> el nivel más alto.
<code><label> </label></code>	Otorga un título a un elemento de una interfaz de usuario.
<code></code>	Inserta una imagen en el documento.
<code><input></code>	Crea controles interactivos para formularios con la finalidad de aceptar datos de usuario. Tiene muchas variaciones y esta estará definida por el atributo “type”.

Propiedades típicas de CSS

Con la ayuda de la tabla 3.3, se muestra de manera general propiedades que presenta CSS a los diferentes elementos de HTML, las cuales han sido utilizadas en la aplicación WEB desarrollada [22]:

Tabla 3.3 Propiedades comunes de CSS para elementos HTML.

Propiedad CSS	Función	Valores de ejemplo
<code>position</code>	Especifica como se posicionarán los elementos en el documento HTML.	<code>relative, absolute, fixed, o sticky.</code>
<ul style="list-style-type: none"> ○ <code>Top</code> ○ <code>left</code> ○ <code>bottom</code> ○ <code>right</code> 	<ul style="list-style-type: none"> ○ Posición inicial desde arriba ○ Posición inicial desde la izquierda ○ Posición inicial desde abajo ○ Posición inicial desde la derecha 	<code>20 px , 20%</code>
<code>width</code>	Define el ancho del área de contenido de un elemento	<code>120 px</code>
<code>height</code>	Define la altura del contenido de un elemento	<code>20%</code>

font-family	Define a una familia de fuentes para aplicarlo a un elemento determinado.	Arial, Calibri
<ul style="list-style-type: none"> ○ margin ○ margin-top ○ margin-right ○ margin-bottom ○ margin-left 	<ul style="list-style-type: none"> ○ Margen/Distancia general ○ Margen/distancia superior ○ Margen/distancia derecha ○ Margen/distancia inferior ○ Margen/distancia izquierda 	15px, 15%
<ul style="list-style-type: none"> ○ background-color ○ background-image 	<ul style="list-style-type: none"> ○ Se especifica un color de fondo a un elemento. ○ Define a un conjunto de imágenes o una imagen de fondo a un elemento. 	<ul style="list-style-type: none"> ○ background-color: #777799; ○ background-image: url("https://mdn.mozilla.org/files/11991/start transparent.gif"),
text-align	Alinea de manera horizontal el contenido dentro de un elemento de bloque.	justify ,left, right, center,
border-color	Establece el color de los cuatro bordes de un elemento.	border-color: black;

3.5 Programa requerido:

Para la codificación de la aplicación WEB mediante el uso de lenguajes HTML y JavaScript, se hará uso de un editor de código fuente como lo es Visual Studio Code (VS Code).

Visual Studio Code

Es un entorno de edición de código fuente desarrollado por Microsoft, de tipo libre y para múltiples plataformas, con disponibilidad para Mac Os, Linux y Windows. Entre sus características están; su software multiplataforma, depuración, la cual es capaz de detectar errores en el código, y evitar revisar un error línea por línea por el desarrollador. Otra de sus características más funcionales, esta “IntelliSense”, permitiendo autocompletar la sintaxis de un código, dando como resultado una forma más ágil de escribir un código [23].

Debido a su soporte con cientos de extensiones o plugins, lo convierte en una gran opción para escribir códigos en diferentes tipos de lenguaje como JavaScript, HTML/CSS, Java, etc. Con todo lo antes mencionado, lo convierte en uno de los entornos de desarrollo más aceptado y usados. La figura 3.7 muestra la interfaz del entorno de desarrollo.

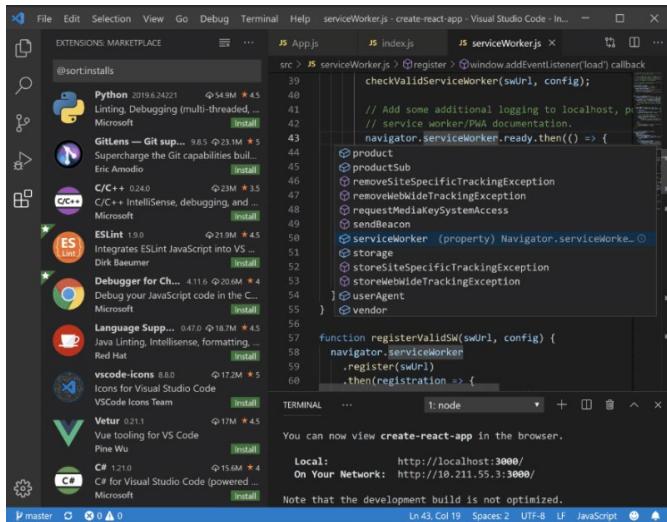


Figura 3.7 Entorno de desarrollo Visual Studio Code [23].

3.6 Librerías a utilizarse:

3.6.1 Bootstrap

Es un “framework” (una estructura base o plantilla que se aprovecha para la elaboración de un proyecto) de lenguaje CSS para la creación de aplicaciones y sitios web con interfaz “front-end”, la cual permite a la aplicación adaptarse a la pantalla de cualquier dispositivo (Propiedad Responsive) [24][25].

Es una herramienta que otorga un estilo a elementos de una página HTML, siendo su característica más importante, proporcionar interactividad en el documento web, por medio de componentes que se comunican con el usuario. Algunos de estos componentes son: barras de progreso, controles de página, menús de navegación.

Se compone de dos directorios:

- Css: contienen los archivos que proporcionan estilo a los elementos HTML.
- Js: aquí se sitúa el archivo “bootstrap.js”, responsable de proporcionar interactividad para la ejecución de las aplicaciones.

3.6.2 Jquery

Es una librería de JavaScript, cuya función es otorgar “interactividad” a una página o sitio web, destinada a resolver situaciones concretas como: un menú que se adapte a la pantalla del usuario, transición entre páginas, un header que cambia su tamaño, etc. Existen una gran cantidad de “plugins” para el desarrollo de distintas aplicaciones, los cuales requieren de una librería general insertada con la siguiente línea de código; “<script src="js/jquery-3.2.1.min.js"></script>” [26].

3.6.3 Smoothie charts

Es una librería diseñada para la creación de graficas que contienen datos en tiempo real. Está diseñada bajo el lenguaje de JavaScript y su empleo de espacio de memoria es pequeño [27]. Esta librería permite la implementación de la gráfica “Tendencia”, la cual está relacionada a los datos de velocidad del variador de frecuencia. La librería permite generar una curva en tiempo real en base a los datos leídos de velocidad en RPM del motor.

3.7 Diseño de la aplicación web

Como una guía para el diseño de la aplicación web que se implementara, se ha creado un boceto, con la finalidad de tener una estructura previa y una determinada organización de los elementos que funcionaran para el monitoreo de las variables y datos de la aplicación. Además, se determinará la ubicación y controles que permitirán el acceso a la aplicación que gestiona la base de datos que almacenara los diversos datos que

proporciona el variador de frecuencia.

La figura 3.8 muestra el boceto creado de la interfaz principal de la aplicación web y mediante los números marcados se indica cada una de las secciones creadas. Posteriormente, la tabla 3.4 muestra cada una de las secciones de la interfaz, y las partes que componen dicha sección.

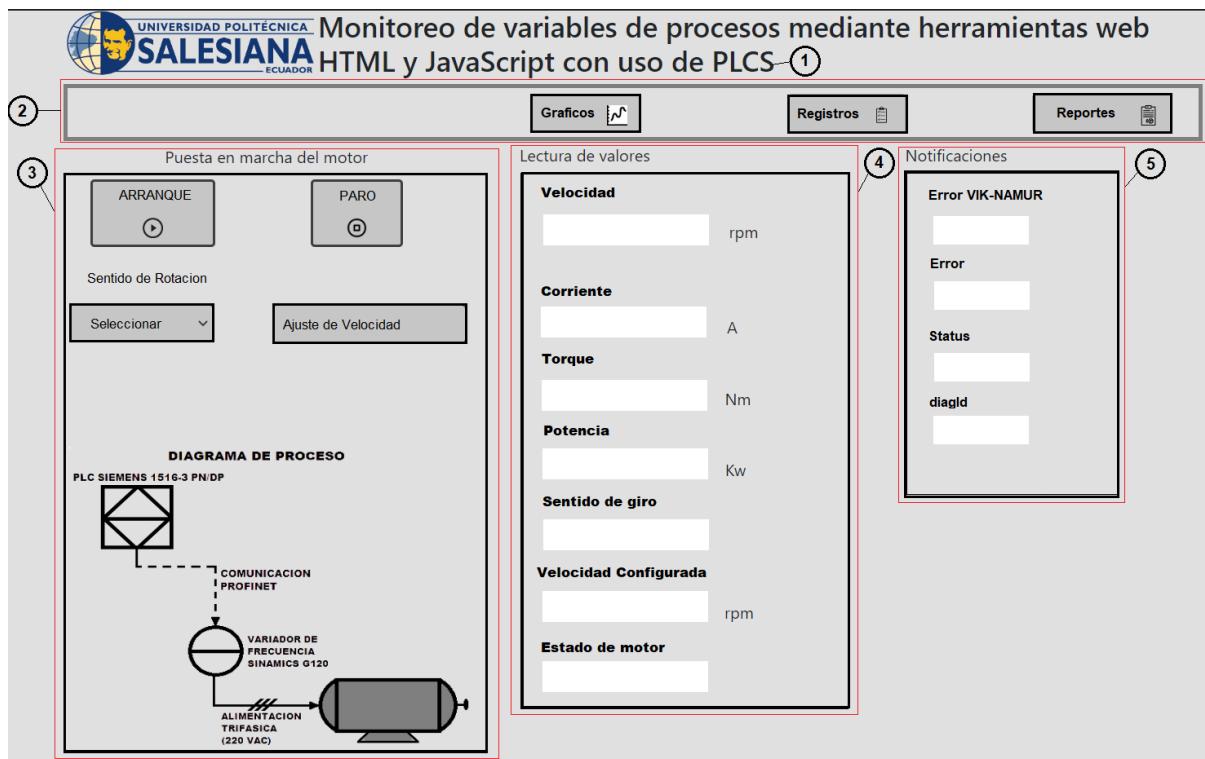


Figura 3.8 Boceto principal de la interfaz de la aplicación WEB.

Tabla 3.4 Elementos de cada una de las secciones de la aplicación WEB.

	Sección	Elementos
1	Encabezado	Posee el título de la aplicación creada.
2	Barra de navegación	Botones que dirigen a las secciones: <ul style="list-style-type: none"> ○ Gráficos ○ Registros ○ Reportes
3	Puesta en marcha del motor	<p>Se presentan botones de control del motor:</p> <ul style="list-style-type: none"> ○ Botón de ARRANQUE o encendido del motor ○ Botón de PARO o apagado del motor ○ Botón de selección de giro, el cual posee tres opciones; <ul style="list-style-type: none"> ➢ Seleccionar ➢ Antihorario ➢ Horario ○ Botón de ajuste de velocidad, el cual despliega la ventana para el control de los RPM del motor por medio de dos opciones: <ul style="list-style-type: none"> ➢ Barra deslizante comprendida entre valores de 0 y 1440. ➢ Cuadro de texto para el ingreso de valores mediante teclado. <p>Además, se presentan una imagen que proporciona una idea general del proceso que se está controlando, mostrando el</p>

		diagrama P&ID de la planta, a la vez, esta cambiara dependiendo del estado del motor y sentido de giro.
4	Lectura de valores: se muestran los valores de entrada y salida provenientes del bloque de función del controlador SINAMICS (Variador de Frecuencia)	<p>Se muestran los valores actuales de salida:</p> <ul style="list-style-type: none"> ○ Velocidad (rpm) ○ Corriente(A) ○ Torque (Nm) ○ Potencia (Kw) <p>También, se muestran los valores de entrada del bloque función con la finalidad de indicar al usuario que valores han sido configurados y enviados al variador, estos parámetros son:</p> <ul style="list-style-type: none"> ○ Sentido de giro ○ Velocidad configurada (rpm) ○ Estado de motor
5	Notificaciones: Se muestran los valores de las variables que devuelven códigos de error y estado del funcionamiento del variador de frecuencia.	<p>Se visualizan los valores de las variables de salida:</p> <ul style="list-style-type: none"> ○ Error VIK-NAMUR ○ Error ○ Status ○ diagld

3.8 Desarrollo de la aplicación web en Visual Studio Code

Lectura y escritura de variables desde la aplicación WEB usando servidor WEB

Debido a que es necesario leer y escribir las variables del bloque función para su control través de la aplicación web, se ha definido una estrategia para acceder a los valores de cada variable y así manipularlas.

Una vez que se ha creado el proyecto de HTML en el entorno de desarrollo (VS CODE), se creara un archivo .htm para cada una de las variables de lectura y escritura, esto con el objetivo de que cada variable se gestione de manera independiente. Mediante la creación de un archivo htm para cada variable, se pretende leer el estado de la variable sea de entrada o salida. En caso, de que la variable sea de escritura o que se puede modificar, por medio de los controles de html, se cambiara su valor o estado una vez que se dirija al archivo .htm que alberga la variable.

Además, por medio de este método, se logra que cada página htm, se actualice de manera independiente y por medio de un archivo principal “index.html”, se acceda a cada una de las variables de escritura y lectura. Así, se evita que el documento html principal tenga que actualizarse por completo cada cierto tiempo para acceder a los valores de las variables del variador.

Siguiendo con el proceso de implementación de la aplicación, se creará tres carpetas que almacenaran los archivos; CSS, JavaScript e Imágenes que se usaran para dar un contexto del proceso que se está controlando. La figura 3.9 muestra el proceso antes mencionado y cada uno de los archivos htm creados.

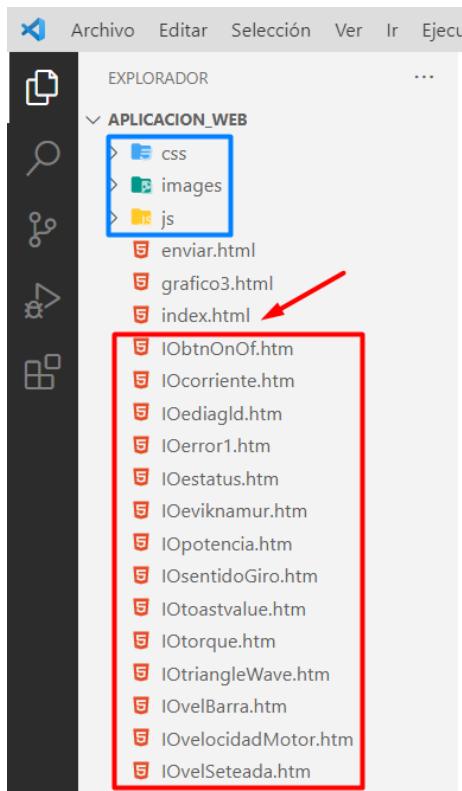


Figura 3.9 Archivos htm creados.

Mediante la figura 3.10, se muestra la relación de cada archivo htm con su respectiva variable de lectura(entrada) o escritura(salida).

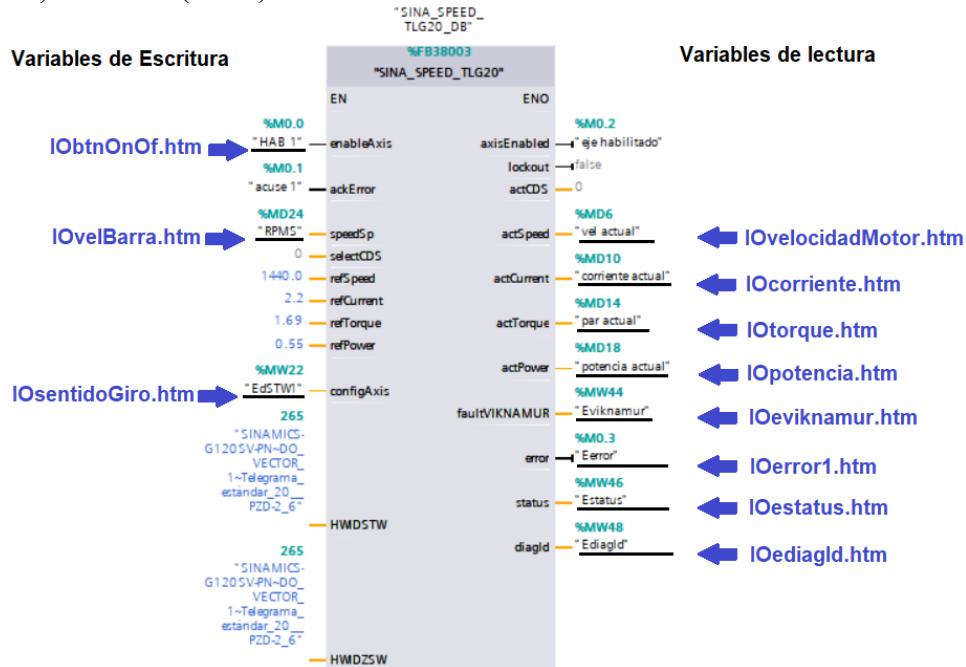


Figura 3.10 Variables de lectura y escritura relacionados con su respectivo archivo htm.

Comandos AWP

Ahora que se conoce la relación de cada una de las variables con sus respectivos archivos htm, es necesario insertar los comandos “AWP”, los cuales posibilitan la escritura y lectura de las variables desde el documento htm. En la tabla 3.5, se muestra los comandos AWP insertados en cada archivo htm y su función en el mismo.

Tabla 3.5 Comandos AWP insertados en cada archivo htm.

	Comando AWP	Función
Variables de escritura		
HAB1	<!-- AWP_In_Variable Name=' "HAB 1" '-->	Se define la variable “HAB1” para operaciones de escritura.
	:= "HAB 1":	Se lee el valor de la variable “HAB1” con la finalidad de conocer el valor enviado por el usuario al variador de frecuencia, esto como un método de retroalimentación. Este proceso se seguirá con las demás variables de escritura.
RPMS	<!-- AWP_In_Variable Name=' "RPMS" '-->	Se define la variable “RPMS” para operaciones de escritura.
	:= "RPMS":	Se lee el valor de la variable “RPMS”.
EdSTW1	<!-- AWP_In_Variable Name=' "EdSTW1" '-->	Se define la variable “EdSTW1” para operaciones de escritura. Esta variable es responsable del control del sentido de giro del motor.
	:= "EdSTW1":	Se lee el valor de la variable “EdsTW1”.
Variables de lectura		
vel actual	:= "vel actual":	Se lee el valor de la variable “vel actual”.
corriente actual	:= "corriente actual":	Se lee el valor de la variable “corriente actual”.
par actual	:= "par actual":	Se lee el valor de la variable “par actual”.
potencia actual	:= "potencia actual":	Se lee el valor de la variable “potencia actual”.
Eviknamur	:= "Eviknamur":	Se lee el valor de la variable “Eviknamur”.
Error	:= "Error":	Se lee el valor de la variable “Error”.
Estatus	:= "Estatus":	Se lee el valor de la variable “Estatus”.
Ediagld	:= "Ediagld":	Se lee el valor de la variable “Ediagld”.

3.9 Implementación del documento HTML principal

Como primer paso, se creará el archivo principal del documento WEB, este archivo se denominará “index”. En este archivo se implementará la interfaz principal, la cual muestra cada una de las secciones que poseerá la aplicación.

La figura 3.11 muestra la estructura principal del documento html. Se muestra como mediante etiquetas, se definen sus dos secciones principales; “head” y “body”. En la sección de cabecera (Head) se incluirán las

diferentes librerías de CSS y scripts implementados que se encargan de las “acciones” de los distintos elementos HTML. Mientras que en la sección del cuerpo (body), se definirá los distintivos elementos o “formularios” que controlan las diferentes funciones de la aplicación WEB.

```
<!DOCTYPE html>
<html lang="en">
  <head> ...
  </head>
  <body style="background-color: #e0e0e0;">
    </body>
```

Figura 3.11 Estructura básica del documento HTML creado.

3.9.1 Sección de cabecera (head) del documento HTML

En la sección “head”, como se mencionó anteriormente, se incluirá las diferentes librerías utilizadas, tal como lo indica la figura 3.12. Mientras que mediante tabla 3.6, se mostrará la función de cada línea implementada en la sección head.

```
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Monitoreo de variables y datos</title>
    <script src="js/jquery-2.0.2.min.js"></script>
    <link rel="stylesheet" href="css/master.css">
    <link rel="stylesheet" href="css/bootstrap.css">
    <script type="text/javascript" src="js/script.js"> </script>
    <script src="js/bootstrap.js"></script>
    <script src="js/enviar.js"></script>
    <script> ...
    </script>
    <style> ...
    </style>

  </head>
```

Figura 3.12 Sección HEAD del documento HTML creado.

Tabla 3.6 Función de cada sentencia implementada en la sección HEAD.

Sentencia	Función
<meta charset="utf-8">	Posibilita que el documento sea visualizado por cualquier idioma. Incluye todos los caracteres universales.
<meta http-equiv="X-UA-Compatible" content="IE=edge">	Define la compatibilidad del documento con el navegador WEB.
<meta name="viewport" content="width=device-width, initial-scale=1.0">	Optimiza el contenido del documento para que sea visualizado en dispositivos móviles.
<title>Monitoreo de variables y datos</title>	Define el título con el que aparecerá el documento en la pestaña del buscador WEB. En este caso es: Monitoreo de variables y datos.
<ul style="list-style-type: none"> ○ <script src="js/jquery-2.0.2.min.js"></script> ○ <script type="text/javascript"> 	Se especifica la dirección URL de archivos de JavaScript externos que será ejecutados en el documento. Estos documentos son: <ul style="list-style-type: none"> ○ jquery-2.0.2.min.js: se importa la librería general que contiene las funciones de Jquery.

<pre> <script> </script> ○ <script src="js/bootstrap.js"></scr ipt> ○ <script src="js/enviar.js"></script> > </pre>	<ul style="list-style-type: none"> ○ script.js: Contiene las funciones que posibilitan obtener los valores de las distintas variables de lectura y escritura; y realizar distintas acciones en base a los valores recibidos. También se configuran las funciones de escritura que posibilitan controlar los parámetros de entrada del bloque función. ○ bootstrap.js: contiene las funciones de JavaScript para el funcionamiento de la librería de CSS Bootstrap. ○ enviar.js: Contienen las funciones que posibilitan controlar el valor de los RPM del motor desde una barra deslizante y desde un cuadro de texto al ingresar un valor entre 0 y 1440 RPM.
<pre> ○ <link rel="stylesheet" href="css/master.css"> ○ <link rel="stylesheet" href="css/bootstrap.css" </pre>	<p>Se vinculan las librerías que se usarán en el proyecto, en este caso las librerías de lenguaje CSS; master.css y bootstrap.css.</p>

Cabe mencionar que la función de cada script implementado, será más detallada en secciones posteriores, cuando se implemente su función en cada elemento HTML.

3.9.2 Sección de cuerpo (Body) del documento HTML

Esta sección tiene como función principal, implementar los distintos elementos que controlaran el proceso del PLC como; controles, botones, gráficos, ventanas emergentes,

Estructura de la aplicación WEB:

Para definir las diferentes secciones de la aplicación WEB, mostradas en la figura 3.8, se hará uso de sistema de cuadriculas de la librería CSS de Bootstrap v4.5. Esta librería está formada por contenedores (class="container"), filas (class="row") y columnas (class="col"), elementos que permiten alinear y diseñar el contenido.

La figura 3.13 muestra como se ha estructurado las diferentes secciones en filas y columnas, indicándose por medio de color rojo, las filas y de color azul las columnas del documento html. La sección 1 (Encabezado) se la colocara en la fila 1 y ubicándose en una sola columna. La sección 2 (Barra de Navegación) se ubicará en la fila 2 y en una sola columna. Finalmente, la sección 3 (Puesta en marcha del motor), sección 4(Lectura de valores) y sección 5 (Notificaciones), se ubicarán en la fila 3 y colocándose cada sección en una columna diferente.

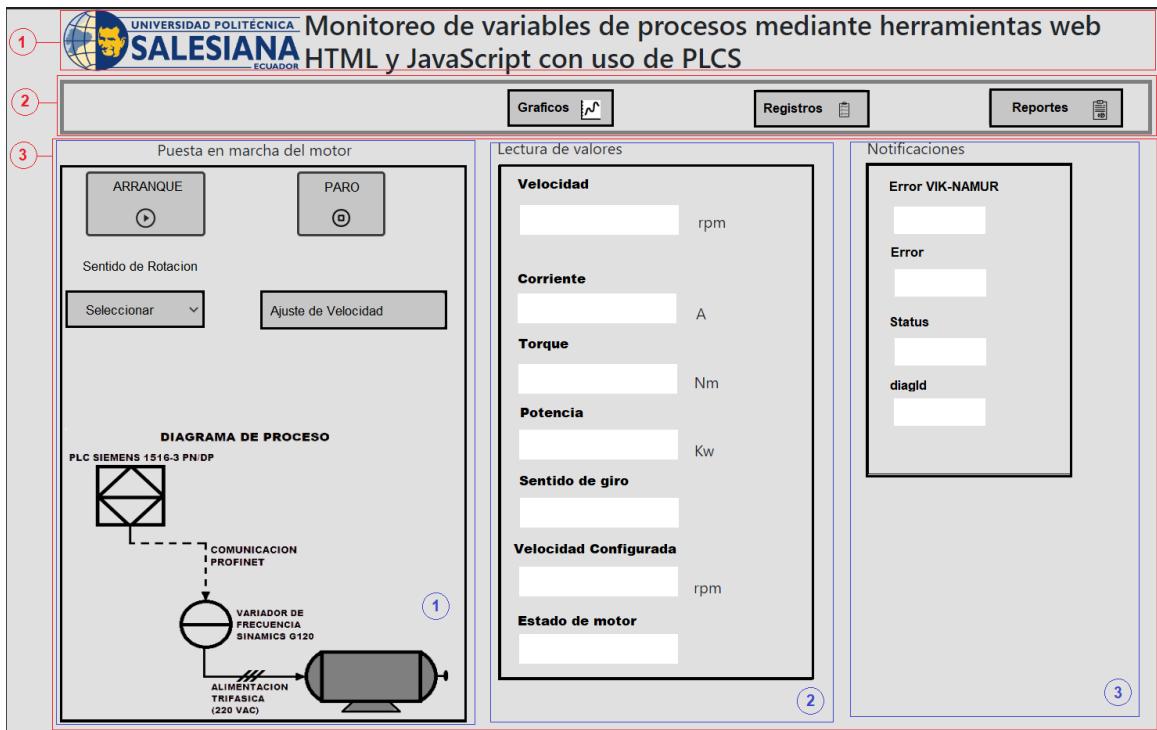


Figura 3.13 Estructura de la aplicación WEB en filas y columnas.

Estructuración de la aplicación web en el documento HTML

Una vez que se definió la ubicación de cada una de las secciones de la aplicación WEB, se implementara dicha alineación en el documento HTML. A continuación, en la figura 3.14, se muestra el código respectivo para la implementación de las filas y columnas de la interfaz. Se puede notar que al inicio de la sección body, se ha usado la propiedad “style” que permite de manera rápida aplicar una propiedad de estilo a un elemento HTML.

En este caso se ha definido un color de fondo “Light gray”, un color recomendado para el fondo de la aplicación ya que es un color que permite obtener una buena visibilidad de los controles del proceso. Luego, se usa la clase "class=container", que permite llenar y centrar todo contenido de la aplicación en el documento html. En este contenedor, se irán agregando cada una de las filas que se definieron para nuestra aplicación, tomando en cuenta que dentro de la clase “row”(filas) ,se colocaran las columnas que contienen el contenido de las secciones de la aplicación diseñada.

```
<body style="background-color: #e0e0e0;">
    <div class="container">
        <div class="row" > <!-- Fila 1 -->
            <div class="col-lg">...
            </div>
        </div>
        <div class="row" > <!-- Fila 2 -->
            <div class="col-lg">...
            </div>
        </div>
        <div class="row" > <!-- Fila 3 -->
            <div class="col-lg" >...
            </div>
            <div class="col-lg" >...
            </div>
            <div class="col-lg" >...
            </div>
        </div>
    </div>
</body>
```

Figura 3.14 Código implementado para la estructura de la aplicación WEB.

3.9.3 Implementación de la sección “Encabezado”

Luego de implementar las filas y columnas para estructurar las secciones de la aplicación, se procede con la implementación de cada sección y sus elementos en el documento html.

Con la figura 3.15, se muestra el código insertado en el documento para definir los elementos antes mostrados y posteriormente con la tabla 3.7, se indica la función de cada elemento y su código de implementación.

```
<div class="row" > <!-- Fila 1 -->
| <div class="col-lg">
|   <div>
|     
|     <h3>Monitoreo de variables de procesos mediante herramientas web HTML y JavaScript con uso de PLCS</h3>
|   </div>
| </div>
</div>
```

Figura 3.15 Código implementado para la sección “Encabezado” de la aplicación.

Tabla 3.7 Elementos de la sección 1 (Encabezado) y su respectivo código de implementación.

	Elemento	Código Implementado	Función
1			Se inserta una imagen con el elemento “img” que contiene un logotipo y se configura sus propiedades con los siguientes atributos: <ul style="list-style-type: none"> ○ src: se indica la ubicación de acceso a la imagen ○ width: se especifica el ancho de la imagen en pixeles. ○ height: se especifica la altura de la imagen en pixeles. ○ style: ubica a la imagen al margen izquierdo de la fila con el propósito de que un texto posterior aparezca al costado de este.
2	Monitoreo de variables de procesos mediante herramientas web HTML y JavaScript con uso de PLCS	<h3>Monitoreo de variables de procesos mediante herramientas web HTML y JavaScript con uso de PLCS</h3>	Se inserta un encabezado de tamaño nivel 3 (h3) que indica el título de la aplicación diseñada.

3.9.4 Implementación de la sección “Barra de Navegación”

La figura 3.16 muestra la sección “barra de Navegación”, y cada uno de sus elementos, los cuales han sido enumerados. Luego por medio de la figura 3.17 se muestra el código para la implementación de esta sección y mediante la tabla 3.8, se detalla el código implementado y función de cada elemento.

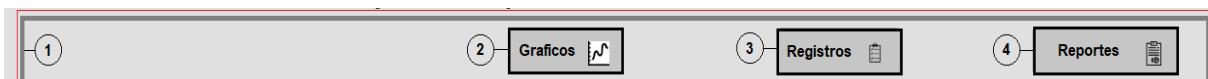


Figura 3.16 Sección 2 de la aplicación WEB.

```

<div class="row" > <!-- Fila 2 -->
  <div class="col-lg">
    | | | <!-- AQUI SE DEFINE LOS BOTONES DE NAVEGACION-->
    | | | <!-- Se crea caja que contiene a los botones de navegacion-->
    | | | <div class="caja" id="dato3" style="width: 29.5cm;height: 1.4cm; border: 3px solid #C6C6C6;">
    | | | | <!-- BOTON GRAFICOS-->
    | | | | <button style="margin-left:450px;" type="button" onclick="window.open('grafico3.html')" class="btn btn-default bttnavegacion">Graficos
    | | | | 
    | | | | </button>
    | | | | <!-- BOTON REGISTROS-->
    | | | | <button style="margin-left:140px;" type="button" onclick="window.open('http://monitoreo.com/registros')" class="btn btn-default bttnavegacion">Registros
    | | | | 
    | | | | </button>
    | | | | <!-- BOTON REPORTES-->
    | | | | <button style="margin-left:120px;" type="button" onclick="window.open('http://monitoreo.com/reporte')" class="btn btn-default bttnavegacion">Ver reportes
    | | | | 
    | | | | </button>
    | | | </div>
  </div>

```

Figura 3.17 Código implementado para la sección 2 de la aplicación.

Código implementado:

Se puede notar en la figura 3.17 que adentro de la fila y columna creadas, se ha insertado la clase de Bootstrap, "class=caja", con la finalidad de agrupar a los botones de navegación. Estos botones se denominan de “navegación”, ya que posibilitan dirigirse a otras secciones de la aplicación web u otras páginas.

Cada botón implementado hace uso de la clase `type="button"` y la clase `class="btn btn-default"`, siendo esta última, la clase que permite insertar un botón con determinadas características de color, forma, letra, color de texto y borde.

Sin embargo, como estos botones deben tener propiedades que los diferencien de botones de control, se ha creado la clase “bttnavegacion”. Además, cada botón posee un icono(imagen) con el fin de proporcionar una idea rápida o un contexto al usuario acerca de la función de este elemento. Esta clase edita las propiedades de la clase `class="btn btn-default"`, es decir modifica las propiedades de diseño del botón.

La figura 3.18 muestra la clase implementada “bttnavegacion”, donde adentro de las llaves, se indican las propiedades a modificar del botón. Con esta clase se definen propiedades de los botones de navegación como color de fondo, color del borde y color de texto. Esta clase se ubicará en la sección “head” del documento de html y adentro del atributo style, con la finalidad de indicar al documento que se ha creado una clase por parte del usuario.

Por otro lado, cuando se implemente el botón de navegación con la clase `class="btn btn-default"`, para agregar las características de la clase “bttnavegacion”, al botón, esta clase estará conjuntamente con la clase que implementa el botón. Es decir, se implementará un botón con las características de una clase “bttnavegacion”, cuando se implemente la línea; `class="btn btn-default bttnavegacion"`.

```

<style>
  .bttnavegacion{
    | | background-color: #c6c6c6; border-color: #AAAAAA; color: #3F3F3F;
  }

```

Figura 3.18 Clase bttnavegacion con sus atributos.

Tabla 3.8 Elementos de la sección 2 y su respectivo código de implementación.

	Elemento	Código Implementado/Función
1		<pre><div class="caja" id="dato3" style="width: 29.5cm; height: 1.4cm; border: 3px solid #C6C6C6;"></pre> <p>Se define la caja que agrupa los botones, la cual posee las siguientes características:</p> <ul style="list-style-type: none"> ○ Ancho ○ Altura ○ Borde
2		<pre><button style="margin-left:450px;" type="button" onclick="window.open('grafico3.html')" class="btn btn-default btnnavegacion">Graficos </button></pre> <p>Se crea el botón “Gráficos” cuyo nombre descrito se encuentra antecedido por el carácter “>”. Después del nombre del botón, se inserta una imagen que sirve como ícono para proporcionar una idea de la función del botón mediante la ayuda del atributo “img”. La función de este botón será abrir una nueva pestaña al hacer “click” en el botón usando el atributo “onclick” y mediante “window.open” indicar el archivo html a abrirse. Este archivo html tendrá los gráficos relacionados a la velocidad del motor, cuyo funcionamiento se abordará en una sección posterior.</p>
3		<pre><button style="margin-left:140px;" type="button" onclick="window.open('http://monitoreo.com/registros')" class="btn btn-default btnnavegacion">Registros </button></pre> <p>Se implementa el botón “Registros” el cual se encuentra ubicado a 140 pixeles a la izquierda del margen de la pantalla de la aplicación, se logra esto mediante el uso y la modificación del atributo “style”. Su función es direccionar al usuario a la aplicación web mediante el ingreso a la url “monitoreo.com/registros”. Esta aplicación web gestiona el almacenamiento, visualización y descarga de los datos de las variables del proceso. Este tema se abordará con más detalle en un capítulo posterior.</p>
4		<pre><button style="margin-left:120px;" type="button" onclick="window.open('http://monitoreo.com/reporte')" class="btn btn-default btnnavegacion">Ver reportes </button></pre> <p>Se implementa el botón “reportes”, cuya función es direccionar a la sección “Reportes” de la aplicación web de Laravel. En esta sección, se puede acceder a la descarga de los datos de las variables del proceso dentro de un rango de fecha y horas determinadas</p>

3.9.5 Implementación de la sección “Puesta en marcha del motor”

La figura 3.19 muestra la sección a implementar en el documento HTML y sus elementos que la componen, los cuales han sido enumerados.

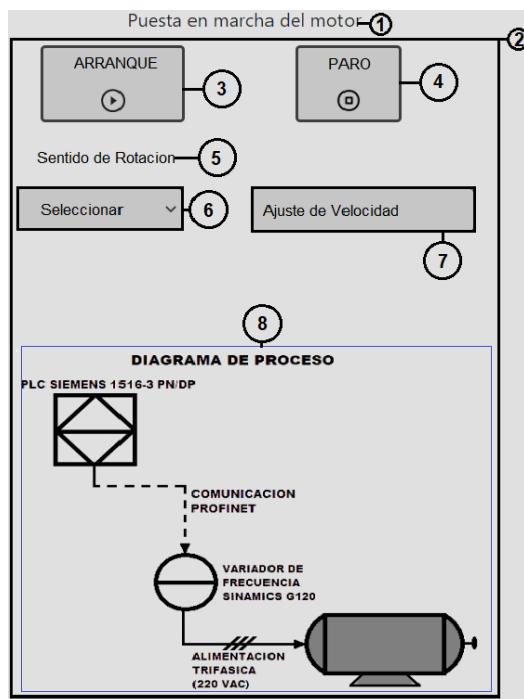


Figura 3.19 Sección 3 de la aplicación WEB.

Con la tabla 3.9, se muestra la función de los elementos enumerados para posteriormente ir describiendo el código de implementación de cada uno de ellos.

Tabla 3.9 Elementos de la sección 3 y su respectiva función en la aplicación WEB.

	Elemento	Función
1	Puesta en marcha del motor	Etiqueta que describe el propósito de la sección. Para ello, se hace uso del elemento “label”.
2		Elemento caja, el cual agrupa a los botones de control del proceso, además de contener a la imagen del diagrama del proceso. Por lo tanto, dentro de la clase “class=”caja””, se irá implementado el código de cada uno de los elementos que esta contiene.
3	ARRANQUE ▶	El botón “ARRANQUE” permite el control del encendido del motor. Su función está relacionada con la variable “HAB 1” del bloque SINA_SPEED, el cual controla el variador de frecuencia.
4	PARO □	El botón “PARO” permite el control del apagado del motor. Su función también está vinculada variable “HAB 1” del bloque función del variador de frecuencia.
5	Sentido de rotación	Etiqueta que describe la función del elemento ubicado en su zona inferior, elemento “multiselect”. Se

		implementa por medio del elemento de HTML denominado “label”.
6		Elemento de control, “multiselect” o “dropdowns”, permite escoger el sentido de giro del motor de una entre tres opciones; Seleccionar (sin sentido), horario o antihorario. Su función está relacionada a la variable “EdSTW1” del bloque función.
7		El botón tiene como función desplegar una ventana de aviso, donde se podrá controlar la velocidad del motor. Esta ventana esta implementada mediante la clase “modal”.
8		Esta imagen tiene como función, proporcionar una idea rápida del proceso que se está controlando. Además, dentro de dicha imagen se reflejará el estado del motor, es decir, la figura del motor cambiara de color dependiendo si este está encendido o apagado. También aparecerá una flecha una flecha encima del motor indicando el sentido de giro actual. Todas estas acciones tienen la finalidad de proporcionar una idea rápida del proceso que se ejecutando por el PLC y tomar alguna decisión o acción adecuada cuando se presente una falla en el proceso.

Implementación del código de los botones y elementos de control de la sección 3

Antes de proceder con la implementación de los controles, es necesario comprender la relación o función que ejecuta cada elemento en las variables de escritura, al presionar en cada uno de ellos. Para ello, con la ayuda de la figura 3.20, se muestra la acción que tiene cada elemento de control en las variables de escritura. Luego, se procederá con la implementación de cada elemento y botón de control, según su acción específica.

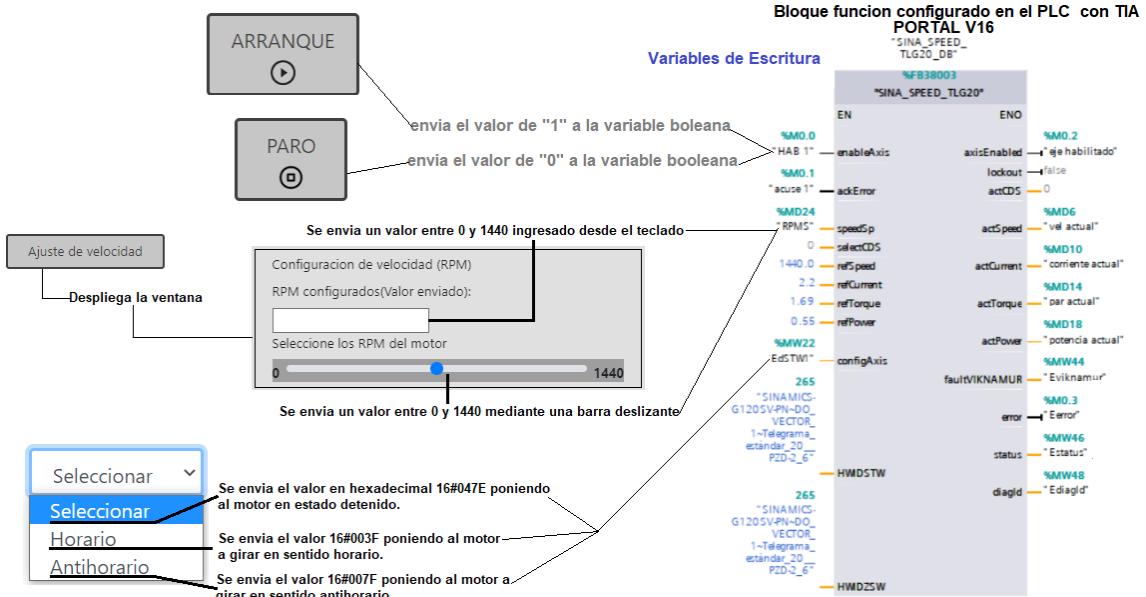


Figura 3.20 Elementos de control y su función en las variables de escritura.

➤ Implementación del botón de control “ARRANQUE”

Una vez que se ha creado el elemento “caja”, con la clase `class="caja"`, y definido su tamaño mediante el atributo `style`, se implementara el botón “ARRANQUE”, adentro de esta clase. Esto se debe a que cada botón este contenido en el elemento caja, por lo que su código de implementación también será colocado dentro de la misma clase. La figura 3.21 muestra el código implementado para insertar el elemento caja y el botón ARRANQUE.

```

<div class="row" > <!-- Fila 3 -->
  <div class="col-lg" >
    <label for="velocidad" style="margin-left: 100px;">Puesta en marcha del motor </label>
    <div class="caja" id="datosON" style="width:11cm; height:15cm;">
      <!-- BOTON ARRANQUE -->
      <button id="btnEncender" onclick="" class="btn btn-default btncontrol" style="width: 122px; margin-left: 20px ;"> ARRANQUE
        
        <input id="setEstadoMotorOn" type="hidden" name='HAB 1' value="1">
      </button>
    </div>
  </div>
</div>

```

Figura 3.21 Código implementado para el botón ARRANQUE.

Descripción del código implementado

Como se observa en la figura 3.21, el elemento “button” posee el atributo “id” denominado “btnEncender” cuyo nombre será único en el documento html. Este atributo permitirá a un script de JavaScript acceder dicho elemento para la ejecución de una acción determinada.

Al igual que en el caso de los botones de navegación, se ha definido la clase “btncontrol”, la cual asigna las propiedades de estilo a los botones de control. Las propiedades de esta clase, se aplicarán a todos los elementos de control de la sección 3. La figura 3.22 muestra la clase “btncontrol”, y sus propiedades de estilo.

```

.btncontrol{
  background-color: #c6c6c6; color: #3F3F3F; border: 3px solid #3F3F3F;
}

```

Figura 3.22 Clase btncontrol con sus atributos.

Luego mediante uso del elemento de html, “img”, se inserta una imagen como ícono para el botón. Esto como un medio de contexto para el operario posea una idea rápida de la función de dicho elemento. La figura 3.23 ilustra el proceso antes descrito.



Figura 3.23 Colocación del ícono en el botón “ARRANQUE”.

Finalmente, se tiene la línea de código “`<input id="setEstadoMotorOn" type="hidden" name='HAB 1' value="1">`”, la cual es de vital importancia para el funcionamiento del botón. Por medio del elemento “input”, se recibe el valor de 1 (`value="1"`), una vez que usuario haya presionado el botón ARRANQUE. Este valor almacenado en el atributo “value” será enviado a la variable “HAB 1”.

Sin embargo, el botón ARRANQUE, aun no es capaz de enviar todavía, el valor almacenado de “1”. Para ello, es necesario implementar una función bajo el lenguaje de JavaScript y Jquery para enviar el valor de entrada de “1” a la variable “HAB 1”, encargada de encender el motor.

Script.js

Este archivo externo de JavaScript tiene la finalidad de contener las funciones que permitirán enviar los valores de los botones de control a las variables de escritura del bloque función del motor. Como primer paso para implementar este script, es crear un archivo denominado “script.js” desde la interfaz de Visual Studio Code, y luego implementar la función `$(document).ready()`. Esta función tiene como objetivo ejecutar las funciones del script una vez que se haya cargado totalmente la página o documento WEB. La figura 3.24, muestra el proceso antes mencionado.

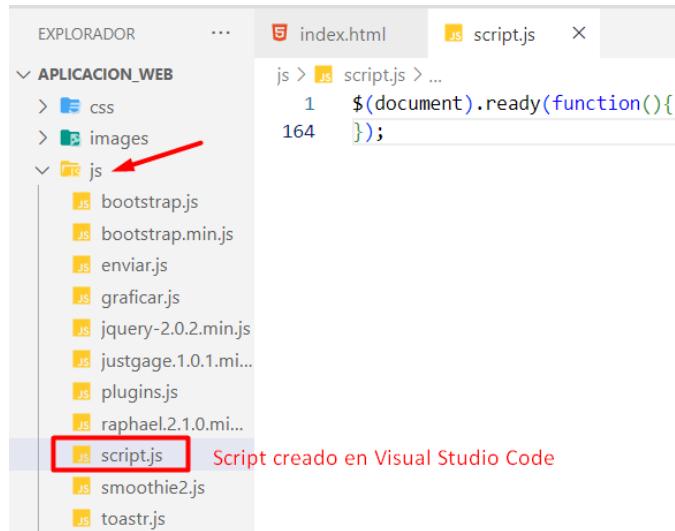


Figura 3.24 Script generado en Visual Studio Code.

Adentro de la sección de la función `$(document).ready()`, se implementará la función para encender el motor una vez que se produzca el “evento” o acción denominada “clic”. La figura 3.25, muestra el código implementado de la función antes mencionada.

```
$('#btnEncender').click(function(){
    url="IObtnOnOff.htm";
    name='HAB 1';
    val=$('#input[id=setEstadoMotorOn]').val();
    sdata=escape(name)+ '=' + val;
    $.post(url,sdata,function(result){});
    console.log('btn encender presionado');
});
```

Figura 3.25 Función para “encender” el motor desde el botón ARRANQUE.

Como se observa en la figura 3.25, la función `$('#btnEncender').click(function(){})`; hace referencia al botón ARRANQUE por medio de su id, #btnEncender. Con ello, se logra que el método “click” ejecute la función que posee, una vez que se haya presionado el botón. La función implementada adentro del método “click”, envía la información del botón ARRANQUE por medio de una solicitud emitida por el método POST al PLC Siemens por medio del servidor web. En este caso, enviar el valor de “1” a la variable “HAB 1”, la cual se encarga de encender el motor.

La figura 3.26 indica el proceso antes mencionado, con la finalidad, de comprender de mejor manera el proceso para escribir un “valor” en una variable de control del bloque de programa del PLC SIEMENS.

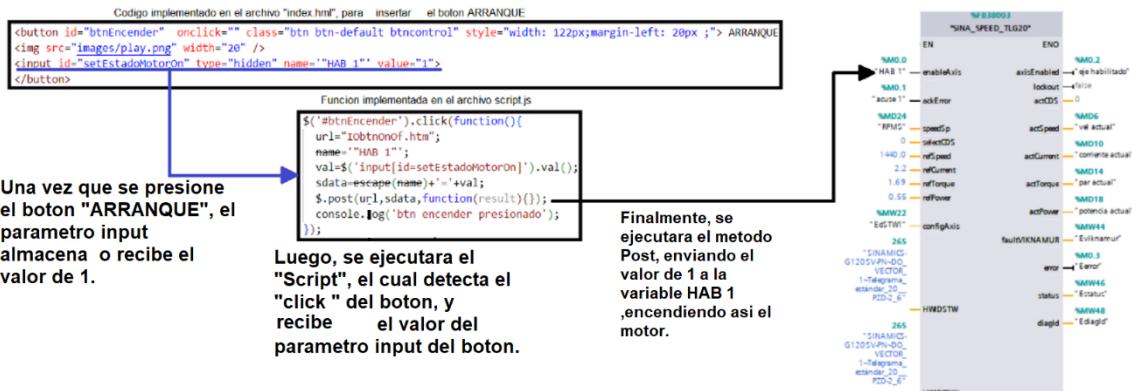


Figura 3.26 Función implementada para el encendido del motor desde botón ARRANQUE.

➤ Implementación del botón de control “PARO”

Para la implementación del botón de control “PARO”, se sigue el mismo procedimiento ejecutado para la implementación del botón de control “ARRANQUE”, tomando en cuenta que el elemento “input”, almacena un valor de cero una vez que se presione dicho botón. La figura 3.27 muestra el código implementado para el botón “PARO”.

```
<!-- BOTON APAGAR -->
<button id="btnApagar" onclick="" class="btn btn-default btncontrol" style="width: 90px; margin-left: 90px;" value="apaga()"> PARO

<input id="setEstadoMotorOf" type="hidden" name="HAB 1" value="0">
```

Figura 3.27 Código implementado para el botón APAGAR.

Ahora, es necesario que se implemente la función en el archivo script.js, para que se envié el valor de “0” a la variable “HAB 1” del bloque función, cuando se presione el botón “PARO”. La figura 3.28 muestra el código implementado de la función en el script.

```
$('#btnApagar').click(function(){
    url="IObtnOnOff.htm";
    name="HAB 1";
    val=$('#input[id=setEstadoMotorOf]').val();
    sdata=escape(name)+ '=' +val;
    $.post(url,sdata,function(result){});
    console.log('btn apagar presionado');
});
```

Figura 3.28 Función para “apagar” el motor desde el botón PARO.

Se puede visualizar nuevamente, que se hace uso del método “POST”, el cual permite el envío de datos a un servidor, en este caso, al servidor WEB integrado en el PLC Siemens. El método POST enviará el valor obtenido (“0”) del botón “PARO”, a la variable “HAB 1”, modificando su valor para apagar el motor.

➤ Implementación del elemento de control “multiselect”

En la figura 3.29, se muestra el código implementado para implementar el elemento de control “Multiselect”. Este elemento permite escoger al usuario una opción de entre tres opciones. Estas opciones definirán el sentido de giro del motor. A continuación, se presentan las tres opciones disponibles y su función:

- Seleccionar: el motor no tiene un sentido de giro, ya que se encuentra en estado de “PARADA”.
- Horario: el motor gira en sentido horario.

- Antihorario: el motor gira en sentido antihorario.

```
<!-- SELECTOR DE GIRO -->
<div class="form-group">
    <label for="exampleFormControlSelect1">Sentido de rotación</label>
    <select id="setEstadoA" name="EdSTW1" class="form-control btncontrol" style="width: 140px; " >
        <option value="16#047E">Seleccionar</option>
        <option value="16#003F">Horario</option>
        <option value="16#007F">Antihorario </option>
    </select>
</div>
```

Figura 3.29 Código implementado para el elemento Multiselect.

Para implementar este elemento de control, se hace uso de la clase class="form-group" y dentro de esta clase, se insertará el elemento “select”, el cual permite crear un menú de opciones. Este control está vinculado a la variable “EdSTW1” del bloque de control, cuya función es la de controlar el sentido de giro. Con lo antes mencionado, cada opción enviará un valor distinto a la variable “EdSTW1”. En el archivo html, cada opción está implementada por el elemento “option”, y el valor a enviar de cada opción será definido en el atributo “value”. Cuando se escoge una opción, la variable EdSTW1 recibirá el valor de la opción elegida.

La figura 3.30, muestra el resultado en el buscador WEB luego de la implementación el elemento “multiselect”. Se puede notar que por medio del elemento “label”, se ha insertado una etiqueta denominada “Sentido de rotación”, como un medio para indicar el propósito del elemento multiselect.

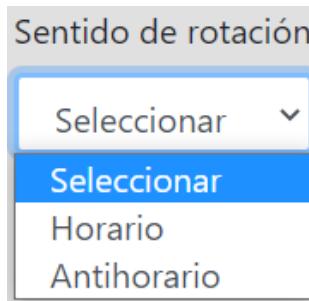


Figura 3.30 Elemento Multiselect para el control de sentido de giro.

Como se indicó en la figura 3.20, cada opción del elemento multiselect envía un valor determinado a la variable EdSTW1 para definir el sentido de giro del motor. Por lo tanto, es necesario, implementar la función para enviar el valor del elemento multiselect, ya que hasta ahora solo se ha insertado solo el elemento de control.

Mediante el lenguaje de JavaScript, se implementará la función que podrá detectar el “cambio” o la acción del usuario, cuando este elija la opción del sentido de giro y dependiendo de la opción escogida se enviará el valor de dicha opción.

Al igual que las anteriores funciones de JavaScript, esta función se encontrará en el archivo “Script”. Para acceder al estado del elemento multiselect desde el “Script”, se utiliza el atributo id “setEstadoA”.

En la figura 3.31, por medio de la función `$('#setEstadoA').on('change', function(){})`, se puede detectar un cambio de estado en el elemento multiselect, es decir cuando el usuario escoja una opción en dicho elemento, la función detectara el evento y enviará el valor de la opción escogida a la variable de control “EdSTW1”. Al igual que en casos anteriores, también se hace el método POST para enviar un valor correspondiente a una variable en el servidor del PLC.

```

$( '#setEstadoA' ).on('change',function(){
    url="I0sentidoGiro.htm";
    name='EdSTW1';
    val=$( 'select[id=setEstadoA]' ).val();
    sdata=escape(name)+ '=' +val;
    $.post(url,sdata,function(result){});
    console.log('btn giro presionado'+val);
}

```

Figura 3.31 Función para definir el sentido de giro del motor.

➤ Implementación del botón de control “Ajuste de velocidad”

La función de este botón será desplegar una ventana emergente que contiene los controles para enviar un valor comprendido entre 0 y 1440, para definir la velocidad en RPM del motor. La figura 3.32 muestra el código implementado para definir este botón.

```

<button type="button" class="btn btn-default position-absolute posicion3 btncontrol" data-toggle="modal" data-target="#exampleModal"
| style="width: 190px; margin-top: 60px;">
Ajuste de velocidad
</button>

```

Figura 3.32 Código implementado para el botón de control “Ajuste de velocidad”.

El elemento “button”, en este caso, posee el atributo `data-toggle="modal"`, el cual una vez que se presione el botón desplegará un modal. Este elemento es una ventana o cuadro de diálogo que contiene los controles para configurar la velocidad del motor.

Se ha propuesto dos formas para configurar el valor de RPM a la variable encargada de controlar la velocidad del motor (variable “RPMS”).

La primera opción será por medio del ingreso de un valor en un recuadro de texto haciendo uso del teclado. Mientras que la segunda opción, será por medio del envío de un valor obtenido de una barra deslizante comprendida en un rango de 0 y 1440 (RPM).

A continuación, con la figura 3.33, se muestra el código implementado para definir el elemento Modal. En la figura, se puede observar la estructura del elemento “Modal”, y el resultado que mostrara en el buscador web. Cabe mencionar, que todo el código de implementación del elemento modal se lo colocara después de la sección body del archivo “index.html”.

```

<!-- Modal -->
<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog" role="document">
    <div class="modal-content" style="background-color: #e0e0e0;">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModalLabel">Ajuste de velocidad</h5>
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">&times;</button>
      </div>
      <div class="modal-body">
        <div class="caja">
          <div class="col">
            <label for="rpms">Configuración de velocidad (RPM)</label>
            <br>
            <label for="rpmsset">RPM configurados (valor enviado):</label>
            <label for="vel" id="velocidad"></label>
            <input type="text" name="SET" id="getBarra">
          </div>
          <div class="col">
            <label for="customRange2">Seleccione los RPM del motor</label>
            <div class="d-flex" style="background-color: #e0e0e0;">
              <span class="font-weight-bold indigo-text mr-2 mt-1"></span>
              <input type="range" name="setRPMS" class="custom-range" min="0" max="1440" step="5" id="rangoBarra">
              <span class="font-weight-bold indigo-text ml-2 mt-1">1440</span>
            </div>
          </div>
        </div>
        <div class="modal-footer">
          <button type="button" class="btn btn-secondary" data-dismiss="modal">Cerrar</button>
        </div>
      </div>
    </div>
  </div>
</div>

```

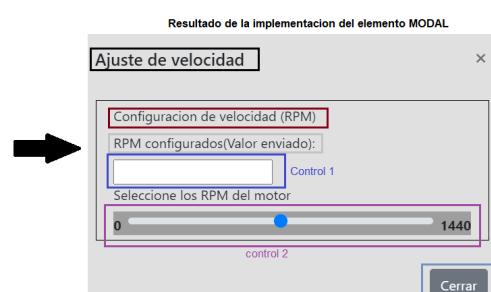


Figura 3.33 Código implementado para el elemento “Modal”.

En la figura 3.33, se puede observar cómo cada elemento mostrado en el buscador web tiene un color determinado, el cual coincide con su respectiva línea de código para su implementación (parte izquierda de

la figura). Además, por medio de los recuadros azul y morado, se visualizan los controles para visualizar los RPM del motor.

Elementos o controles para configurar la velocidad del motor

Por otro lado, para ingresar el valor en RPM por teclado en un recuadro de texto, se emplea el código de la figura 3.34.

```
<input type="text" name="SET" id="getBarra">
```

Figura 3.34 Código implementado para configurar los RPM del motor desde un valor ingresado por teclado.

El código antes implementado posee el atributo id= “getBarra”, el cual posibilita que este elemento sea llamado por un script, el cual servirá para él envío del valor ingresado en el recuadro de texto. Este valor se enviará a la variable “SET” del bloque de control.

Ahora, si se desea configurar los RPM de un motor desde una barra deslizante, se tiene el siguiente código de la figura 3.35.

```
<input type="range" name="setRPMS" class="custom-range" min="0" max="1440" step="5" id="rangoBarra">
```

Figura 3.35 Código implementado para enviar el valor en RPM desde una barra deslizante.

Por medio del elemento “range”, se enviarán valores entre 0 y 1440 a la variable “SET” haciendo uso de una barra deslizante.

Del mismo modo que en el control de velocidad desde el teclado, se usara un script para enviar el valor que se determine mediante la barra. Para ello es importante, saber el “id”, de este elemento para que pueda ser vinculado al script.

Consideración importante:

Debido a que los dos controles de la variable de velocidad (“RPMS”) envían valores de tipo entero y dicha variable acepta valores de tipo flotante, se propone la siguiente solución. Desde el programa de TIA PORTAL, se crea una nueva variable denominada “SET” de tipo entero, esta variable recibirá el valor de los dos controles de velocidad.

Luego haciendo uso del bloque CONV, se transforma el valor de tipo entero a tipo flotante. La variable que recibirá este valor flotante es la variable “RPMS”, la cual fue configurada para controlar la velocidad del motor. Es por ello, que los controles de velocidad están vinculadas a la variable “SET”. La figura 3.36 describe el proceso antes mencionado.

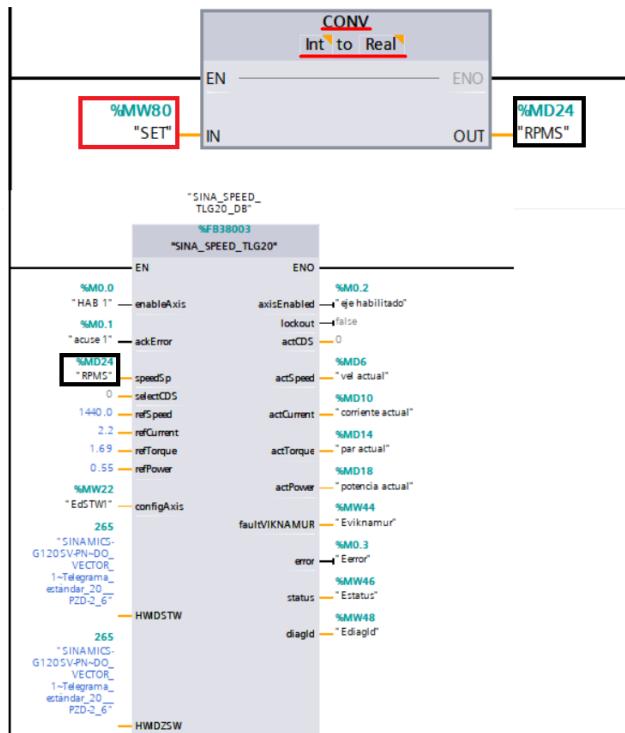


Figura 3.36 Configuración en TIA PORTAL para recibir valores de tipo flotante (Real), desde la aplicación.

Script “enviar.js”

Este script contiene las funciones para enviar los valores a la variable SET, haciendo uso del método POST. Este valor proviene de los dos tipos de controles por lo que es importante tener en cuenta el id de los elementos de control para vincularlos al script.

La figura 3.37 muestra el código implementado en el script creado, el cual será vinculado al archivo index.html por medio del elemento <script></script>.

La sección de color azul, define a la función encargada de enviar valores desde la barra deslizante mientras que la sección marcada de color rosado, contiene la función que envía los valores de velocidad ingresados por teclado. Ambas funciones contienen el método POST para él envió de valores al servidor.

```

js > enviar.js > ready() callback > setInterval() callback
1   $(document).ready(function(){
2
3     $.ajaxSetup({ cache: false });
4     setInterval(function() [
5       $.get("IOvelSeteada.htm", function(result){
6         |   $('#velocidad').text(result);
7         });
8       ],1000);
9   // acciones pra enviar
10  $('#rangoBarra').change(function(){
11    url="IOvelBarra.htm";
12    name='SET';
13    val=$('#input[id=rangoBarra]').val();
14    sdata=escape(name) +'='+val;
15    $.post(url,sdata,function(result){});
16    $('#getBarra').val(val);
17    console.log('barra cambiada');
18  });
19  $('#getBarra').change(function(){
20    result=$('#input[id=getBarra]').val();
21    if (result>0 && result<1441) {
22      url="IOvelBarra.htm";
23      name='SET';
24      val=$('#input[id=getBarra]').val();
25      sdata=escape(name) +'='+val;
26      $.post(url,sdata,function(result){});
27      $('#rangoBarra').val(result);
28      console.log('caja cambiada');
29    }
30    else{
31      |   alert('VALOR INCORRECTO,ingrese un valor entre 0 y 1440');
32    }
33    // $('#rangoBarra').val(result);
34  });
35 });
36 });

```

Figura 3.37 Script “enviar” para el envío de valores desde los controles de velocidad.

➤ Implementación del diagrama P&ID

El propósito de este elemento es otorgar un contexto del proceso que se está controlando mediante el uso de imágenes. Esta imagen es insertada por medio del elemento de html “img” y mediante sus atributos como width(ancho) y height(alto) se define su tamaño. La figura 3.38 muestra el código implementado para colocar la imagen, la cual representa el diagrama P&ID del proceso controlado.

En la figura también se puede notar que se insertan dos imágenes; "images/flujoON.png" y "images/flujoOFF.png", pero en pantalla solo se observara una de ellas. Esto es posible con ayuda del atributo `style="display: block;"`, el cual muestra una imagen en pantalla mientras que el atributo `style="display: none;"`; oculta una imagen. La finalidad de colocar dos imágenes, será la de mostrar una imagen cuando el motor este encendido("images/flujoON.png") y otra, cuando el motor este apagado.

La imagen de encendido y apagado del motor, se muestra en la figura 3.39. Sin embargo, por defecto la imagen a mostrar en pantalla será la imagen del motor apagado. En caso, de que presione el botón “ARRANQUE”, que enciende el motor, por medio de un script se cambiara a mostrar la imagen del motor encendido. Es decir, la imagen a mostrar estará en función del estado de la variable que controla el encendido del motor, “HAB 1”.

```

<!-- MUESTRO LA IMAGEN DEL DIAGRAMA P&ID SEGUN EL ESTADO DEL MOTOR -->
<div class="btn">
    <div class="position-absolute posicion4" id="encendido" style="display: none; ">
        
    </div>
    <div class="position-absolute posicion4 " id="apagado" style="display: block;">
        
    </div>
</div>

```

Figura 3.38 Código para implementación de imágenes en la aplicación WEB.

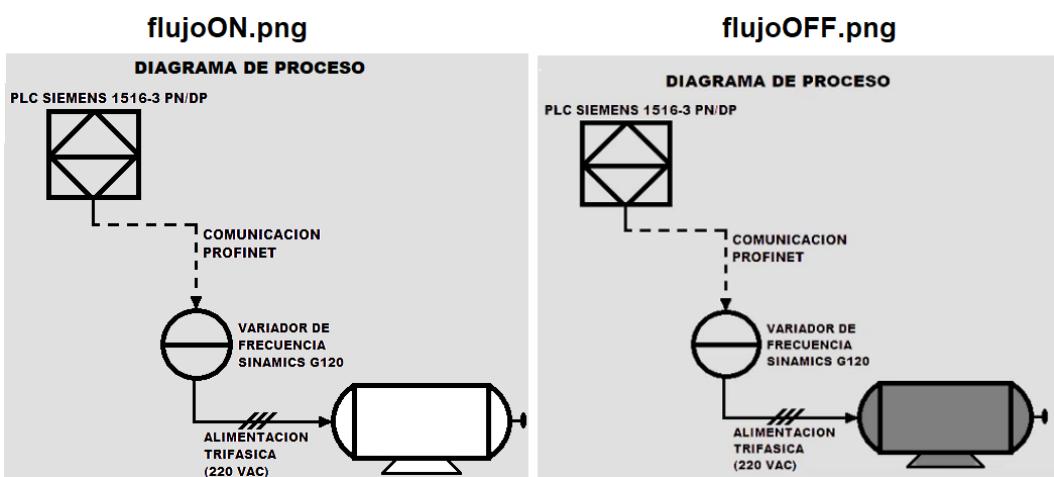


Figura 3.39 Imágenes; "images/flujoON.png" y "images/flujoOFF.png".

Otra de las propiedades configurada en el elemento img, es la clase `class="position-absolute posicion4"`, donde “posicion4”, es una clase de CSS, que define las propiedades de posición, es decir se define la posición en X y Y de la imagen. La figura 3.40, muestra la clase “posición 4” (recuadro azul), además de visualizar otras clases que definen la posición y demás propiedades de otros elementos. Cabe mencionar, que cada una de estas clases son definidas en la sección head del archivo principal “index”.

```

<style>
    .bttnnavegacion{
        background-color: #c6c6c6; border-color: #AAAAAA; color: #3F3F3F;
    }
    .btnccontrol{
        background-color: #c6c6c6; color: #3F3F3F; border: 3px solid #3F3F3F;
    }
    .posicion2{
        top:25px; left:255px;
    }
    .posicion3{
        top:101px; left:220px;
    }
    .posicion4{
        top:300px; left:20px;
    }
    .posicion5{
        top:485px; left:320px;
    }
    .posicion6{
        top:22px; left:145px;
    }
    .posicion7{
        top:37px; left:145px;
    }
    .parametros salida{
        color: #475CA7; border: 1px solid #e0e0e0; background-color: #e0e0e0; text-align: right; width: 4.5cm;
    }
    .parametros error{
        color: #475CA7; border: 1px solid #e0e0e0; background-color: #e0e0e0; width: 3cm;
    }
</style>

```

Figura 3.40 Clase de CSS “posicion4”.

Por otro lado, como un recurso para indicar el sentido de giro del motor, se insertará imágenes que contienen flechas que indican dos sentidos; horario y antihorario. Estas imágenes serán colocadas encima de la imagen del motor del proceso. Por lo que, dependiendo del sentido de giro configurado, aparecerá una imagen con la fecha correspondiente. La figura 3.41 muestra el código implementado para colocar cada una de las flechas que indican el sentido de giro del motor.

Para lograr ubicar la imagen de la flecha en la imagen del motor (flujoON.png), se hace uso de clase creada “posición 5”, que ubica la imagen en una posición X y Y específica. Esta clase creada se la puede observar en la figura 3.40.

Para mostrar cada imagen, se deberá implementar una función en JavaScript que muestra la imagen dependiendo del estado de la variable “EdSTW1”, responsable de controlar el giro del motor. Esta función será descrita en la sección “Gestión de alarmas y notificaciones”.

```
<!-- MUESTRO UNA IMAGEN DE REFERENCIA DE ACUERDO AL SENTIDO DE GIRO DEL MOTOR -->
<div class="btn">
  <div class="position-absolute posicion5" id="giroizq" style="display: none; ">
    | | 
  </div>
  <div class="position-absolute posicion5 " id="giroder" style="display: none;">
    | | 
  </div>
</div>
```

Figura 3.41 Código para implementación de imágenes que indican el sentido de giro en la aplicación WEB.

3.9.6 Implementación de la sección “Lectura de Valores”

En esta sección se visualizará el valor de cada variable de entrada y salida. La figura 3.42, muestra cada campo y su relación con cada una de las variables del proceso, que se visualizaran por medio de un recuadro de texto (elemento `input type="text"`).

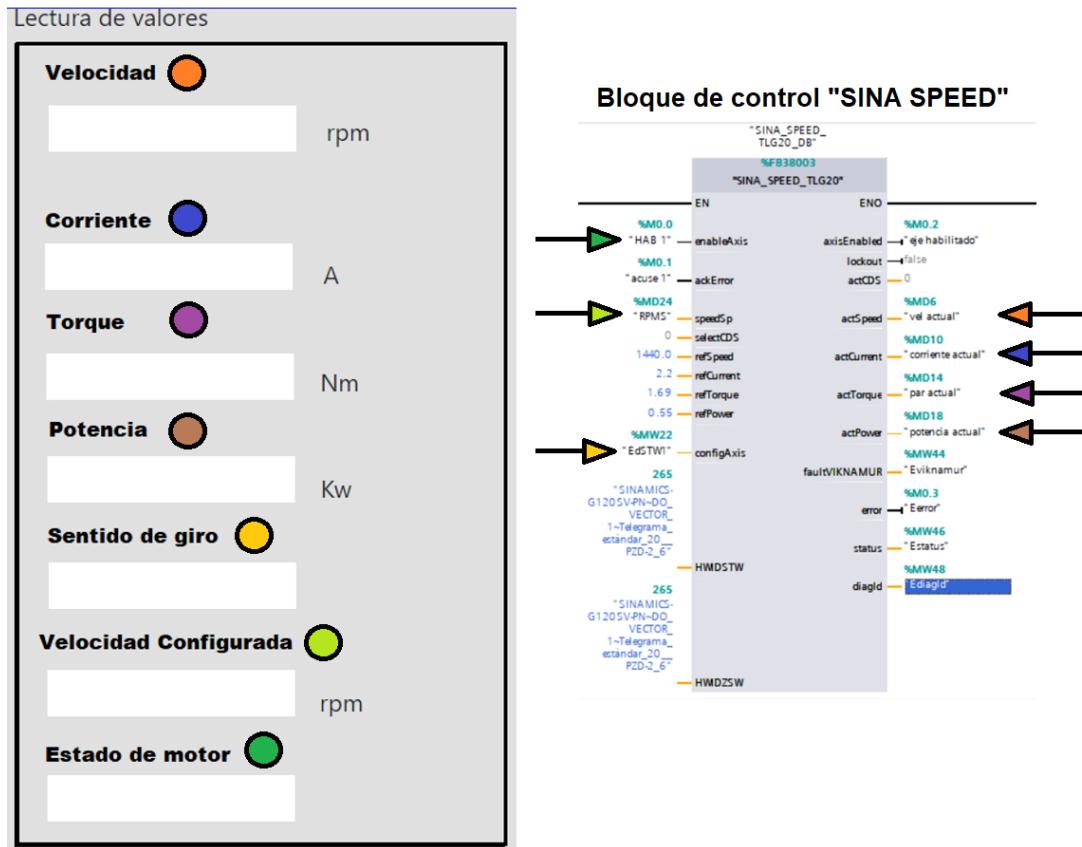


Figura 3.42 Sección “Lectura de Valores” de la aplicación web.

En la figura 3.42, se observa que cada campo (variable) a mostrar en un recuadro de texto, hace uso de dos etiquetas que indican la magnitud y la unidad de medida de la variable.

Por medio de la figura 3.43, se muestra el código de implementación para mostrar el valor del campo “Velocidad”, y las etiquetas que muestran su magnitud y unidad de medida. En la figura, se observa que se hace uso del elemento “caja” (línea 149), ya que este elemento alberga cada uno de los recuadros de texto que reciben los valores de las variables de proceso.

Otro de los elementos necesarios, es el elemento `<div class="col">`. Este elemento permite crear una columna para contener el cuadro de texto que muestra el dato de una variable, y las etiquetas. De esta manera los elementos de la columna se encontrarán alineados.

Luego mediante los elementos “label” (línea 152-153), se insertan las etiquetas que señalan la magnitud y unidad de medida(rpm). Finalmente, por medio del elemento `input type="text"`, se inserta un cuadro de texto que recibe el valor del campo “Velocidad”, el cual está relacionada con la variable “vel actual” del bloque función.

```

145 <div class="col-lg" >
146   <label for="label-password"> Lectura de valores
147   </label>
148
149   <div class="caja" id="datosM" style="margin-top: 0px; width: 8.5cm;">
150     <form action="">
151       <div class="col">
152         <label for="velocidad"><strong>Velocidad</strong> </label>
153         <label for="label-password" style="position: absolute; margin-top: 39px; margin-left: 110px;">rpm</label>
154         <input type="text" class="form-control parametrossalida" name="" id="VelMotor" value="">
155
156       <div class="btn">
157         <div class="position-absolute posicion2" id="alertaamarilla" style="display: none; ">
158           
159         </div>
160       </div>
161     </div>

```

Figura 3.43 Código implementado para visualizar los datos de la variable “Velocidad”.

Para un mejor entendimiento de la función de cada línea de código implementada, la figura 3.44 muestra la relación de cada línea de código de la figura 3.43 para mostrar los elementos en pantalla que permite visualizar los datos del campo (Velocidad).

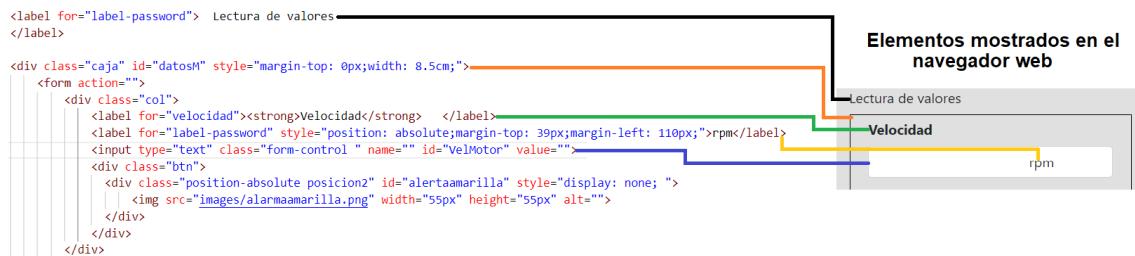


Figura 3.44 Elementos para mostrar los datos de la variable “Velocidad” y su código de implementación.

En la figura 3.44, se usa las propiedades `style="position: absolute; margin-top: 39px; margin-left: 110px;"`, para situar la etiqueta “rpm”, en una determinada posición, logrando que este elemento aparezca alineado a la derecha del cuadro de texto. Además, otra característica que se puede notar, es el fondo color blanco del recuadro de texto.

Con la finalidad de no causar una mala visualización del valor de la variable mediante el fondo blanco, se modifica las propiedades de este recuadro por medio de una clase de css. Por esta razón, se usa la clase “parametrossalida” (línea 154), visualizada en la figura 3.43, la cual permite modificar algunas propiedades del recuadro de texto.

La figura 3.45 muestra la clase “parametrossalida” y las propiedades de estilo aplicadas al recuadro de texto. Por medio de esta clase, se configuran parámetros como; color del texto ingresado (`color: #475CA7`), color de borde (`border: 1px solid #e0e0e0`), color de fondo (`background-color: #e0e0e0`), alineación del texto (`text-align: right`) y ancho del recuadro (`width: 4.5cm`).

```

.parametrossalida{
  color: #475CA7; border: 1px solid #e0e0e0 ; background-color: #e0e0e0; text-align: right; width: 4.5cm;
}

```

Figura 3.45 Clase de CSS “parametrossalida”.

Por último, uno de los atributos más importantes, es el atributo “id” (línea 154) del elemento “input type=text”, que permite crear un cuadro de texto, visualizado en la figura 3.43. Este atributo permite identificar el cuadro de texto, para asignarle el valor a mostrar. Por medio de este atributo, una función de jQuery implementada en un “script”, permitirá asignar el valor de cada variable a visualizar. Esta función se implementa, ya que el cuadro de texto por sí solo no es capaz de recibir valores y actualizarlos.

Una vez que se ha descrito los elementos de html para visualizar un dato en el campo “Velocidad”, se seguirá este mismo esquema para visualizar los demás datos de cada campo de la sección “Lectura de Valores”.

La figura 3.46 muestra el código implementado, para visualizar las demás variables de proceso y su presentación en el buscador web (sección derecha). En la figura, se puede notar ahora que cada recuadro de texto no es visible. Esto se debe a que ahora cada cuadro de texto tiene el mismo color usado para el fondo del cuerpo (body) del documento html.

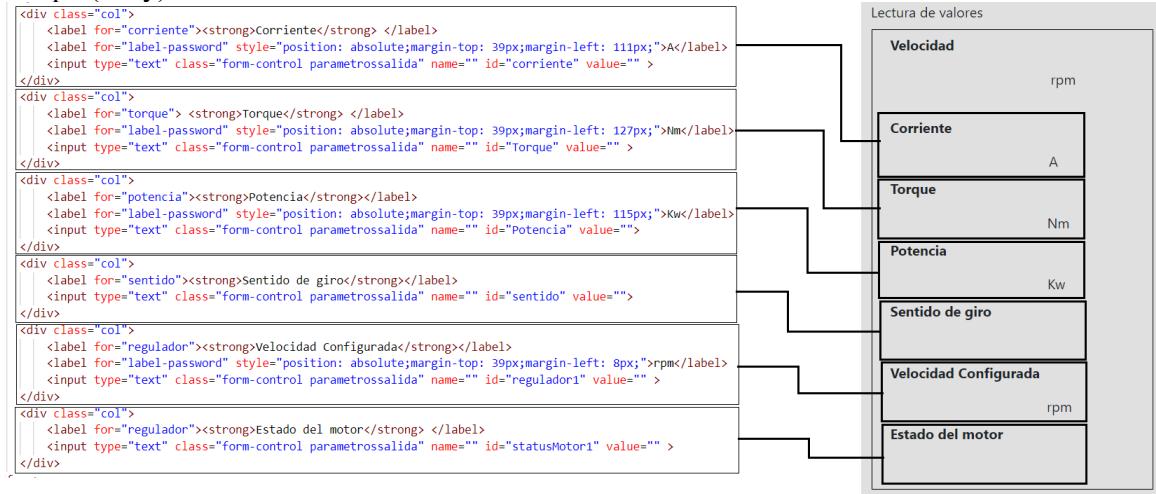


Figura 3.46 Código implementado para visualizar los datos de la sección “Lectura de valores”.

Función setInterval

El uso de esta función tiene como objetivo ejecutar cada cierto tiempo las funciones contenidas en su estructura. Esta función se la implementara en el api denominado “Script”, creado con JavaScript en la sección 3.9.5.

La figura 3.47, muestra la función implementada en el api “Script”. En la figura se puede notar que esta función tiene un parámetro de “1000”, este parámetro indica a la función que se actualice cada 1000 ms o un segundo, provocando que las funciones contenidas en esta, se actualicen cada segundo.

Figura 3.47 Función setInterval.

Función “get”

Esta función estará contenida en la función “setInterval”, con la finalidad de obtener los valores de las variables de proceso. Por lo que es necesario que se implemente esta función, para obtener los valores de cada una de las variables que se mostraran en sus respectivos campos de la sección “Lectura de valores”.

La figura 3.48, muestra la función (línea 8) para obtener el valor de la variable “HAB 1” del bloque función, contenida en el archivo .htm “I0btnOnOf.htm”, para mostrar en el recuadro del campo “Estado del motor”. Luego por medio de la función `$(()).val(result){};` (línea 9), se asigna el valor obtenido del archivo htm a la variable auxiliar “`#statusMotor1`”, esto como ayuda para establecer la condición de que si se obtiene un valor de “1” (línea 10), se muestre en el recuadro del campo “Estado del motor”, la palabra “Encendido” (línea 11). Caso contrario (línea 15), se mostrará en el recuadro de texto, la palabra “Apagado” (línea 16).

Se hace uso del condicional “if”, para mostrar un texto “Encendido” o “Apagado”, debido a que la variable del campo Estado del motor, “HAB 1”, devuelve un valor booleano; 0 o 1, Por ello, dependiendo del valor de la variable “HAB 1”, se mostrara la palabra correspondiente, indicando el estado actual del motor.

```
8     $.get("IObtnOnOff.htm", function(result){  
9         $('#statusMotor').val(result);  
10        if (result==1) {  
11            $('#statusMotor1').val("Encendido");  
12            document.getElementById('encendido').style.display="block";  
13            document.getElementById('apagado').style.display="none";  
14        }  
15        else{  
16            $('#statusMotor1').val("Apagado");  
17            document.getElementById('apagado').style.display="block";  
18            document.getElementById('encendido').style.display="none";  
19        }  
20    });
```

Figura 3.48 Función para mostrar valores en el campo “Estado del motor”.

Usando los mismos métodos o funciones empleadas para mostrar los valores en el campo “Velocidad”, se los emplearan para mostrar los valores de los demás campos. La figura 3.49 muestra el código implementado para mostrar los valores de los campos; “Velocidad”, “Corriente”, “Torque”, y “Potencia”.

```
1 ///////////////Se lee el valor actual de velocidad del motor/////////
2 $.get("IOvelocidadMotor.htm", function(result){
3     var res1=result.replace('&#x2d;', '');
4     $('#VelMotor').val(res1); -----> Se muestra el valor de la variable "vel actual" en el campo "Velocidad"
5     if (res1>=1400) {
6         $("#alertaamarilla").css('display','block');
7     } else {
8         $("#alertaamarilla").css('display','none');
9     }
10 });
11 ///////////////Se lee el valor actual de corriente del motor/////////
12 $.get("IOcorriente.htm", function(result){
13     $('#corriente').val(result); -----> Se muestra el valor de la variable "corriente actual" en el campo "Corriente"
14 });
15 ///////////////Se lee el valor actual de torque del motor/////////
16 $.get("IOTorque.htm", function(result){
17     var res2=result.replace('&#x2d;', '');
18     $('#Torque').val(res2); -----> Se muestra el valor de la variable "par actual" en el campo "Torque"
19 });
20 ///////////////Se lee el valor actual de potencia del motor/////////
21 $.get("IOPotencia.htm", function(result){
22     $('#Potencia').val(result); -----> Se muestra el valor de la variable "potencia actual" en el campo "Potencia"
23 });
```

Figura 3.49 Función para mostrar los valores de los campos; “Velocidad”, “Corriente”, “Torque”, y “Potencia”.

Por último, mediante la figura 3.50, se muestra las funciones para mostrar los valores de los campos; “Sentido de giro” y “Velocidad Configurada”.

```

/////////Se lee el parametro de entrada encargado del sentido de giro del motor/////////
$.get("IOsentidoGiro.htm", function(result){
    if (result==63) {
        $('#sentido').val("Horario"); → Se muestra la palabra "Horario", cuando se haya seleccionado este
        $('#giroizq').css('display','none'); sentido de giro(campo"Sentido de giro")
        $('#giroder').css('display','block');

    }
    else if(result==127) {
        $('#sentido').val("Antihorario"); → Se muestra la palabra "Antihorario", cuando se haya seleccionado
        $('#giroder').css('display','none'); este sentido de giro(campo"Sentido de giro")
        $('#giroizq').css('display','block');

    }
    else{
        $('#sentido').val("Seleccione giro"); → Se muestra las palabras "Seleccione giro", cuando no se haya
        $('#giroder').css('display','none'); seleccionado un sentido de giro(campo"Sentido de giro")
        $('#giroizq').css('display','none');
    }
});
/////////Se lee el parametro de entrada encargado de establecer los RPM del motor/////////
$.get("IOvelSeteada.htm", function(result){
    $('#regulador1').val(result); → Se muestra el valor de la variable "RPMS" en el campo "Velocidad configurada".
});

```

Figura 3.50 Función para mostrar los valores de los campos; “Sentido de giro” y “Velocidad Configurada”.

3.9.7 Implementación de la sección “Notificaciones”

En esta sección se mostrará los valores de variables que dependiendo del número específico que entregan indican un error y la existencia de fallas en el funcionamiento del variador de frecuencia. Estas variables son; Error VIK-NAMUR, Error, Status y diagId. La figura 3.51 muestra la sección a implementarse. El propósito de esta sección es identificar fallas que pueden ocurrir en el variador de frecuencia durante el proceso, mediante las cuatro variables que entrega el bloque de control y para luego tomar las debidas medidas correctivas por parte del operario.

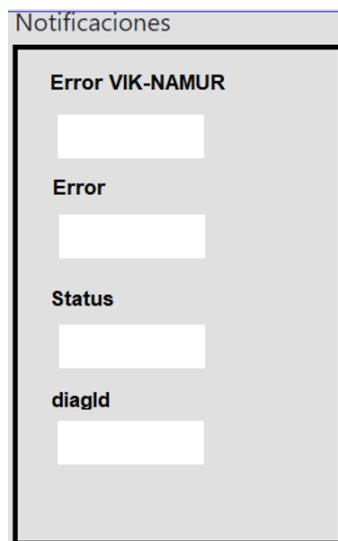


Figura 3.51 Sección “Notificaciones”.

Para la implementación de esta sección, se usa el mismo esquema planteado para la sección “Lectura de valores”. Cada valor de cada una de las variables, se mostrará mediante el uso de un recuadro de texto. La figura 3.52, muestra el código implementado para implementar la sección “Notificaciones”, que se inserta en el archivo principal “index.html”.

```

<div class="col-lg" >
    <label for="label-password">Notificaciones</label>
    <div class="caja" id="datos2" style="margin-top: 0px; width: 5.5cm;">
        <form action="" >
            <div class="col">
                <label for="Eviknamur">Error VIK-NAMUR </label>
                <input type="text" class="form-control parametroerror" name="" id="eviknamur" value="">
            </div>
            <div class="col">
                <label for="Error1">Error </label>
                <input type="text" class="form-control parametroerror" name="" id="error1" value="">
            </div>
            <div class="col">
                <label for="Estatus">Status </label>
                <input type="text" class="form-control parametroerror" name="" id="estatus" value="">
                <!-- ICONOS ALERTAS -->
            <div class="btn">
                <div class="position-absolute posicion7" id="alertaok" style="display: none; ">
                    
                </div>
                <div class="position-absolute posicion6" id="alertaRoja" style="display: none; ">
                    
                </div>
                <div class="audio">
                    <audio id="alerta" class="audio">
                        <source src="images/red-alert.mp3" type="audio/mp3">
                    </audio>
                </div>
            </div>
        </div>
        <div class="col">
            <label for="Ediagld">diagld </label>
            <input type="text" class="form-control parametroerror" name="" id="ediagld" value="">
        </div>
    </form>
</div>
</div>
</div>

```

Figura 3.52 Código implementado para implementar la sección “Notificaciones”.

Luego, nuevamente en el archivo “Script”, se implementará las funciones para obtener y mostrar los datos de las variables de la sección “Notificaciones”. La figura 3.53 muestra el código implementado en el archivo “Script”.

```

//Se lee el parametro error VIK NAMUR del variador/////
$.get("Ioeviknamur.htm", function(result){
    $('#eviknamur').val(result); ————— Se muestra el valor de la variable "Eviknamur" en el campo "Error VIK-NAMUR".
});
//Se lee el parametro error del variador/////
$.get("Ierror1.htm", function(result){
    $('#error1').val(result); ————— Se muestra el valor de la variable "Error" en el campo "Error".
});
//Se lee el parametro status del variador////
$.get("Iestatus.htm", function(result){
    var res12=result.replace('&#xd;','');
    $('#estatus').val(res12); ————— Se muestra el valor de la variable "Estatus" en el campo "Status".
// console.log(result);
// console.log('valorpositivo '+res12);
    if (res12==31231) { ...
    }
    else if(res12==31742) { ...
    }
    else if(res12==28674) { ...
    }
    else{...
    }
});————— Se muestra el valor de la variable "Estatus" en el campo "Status".

//Se lee el parametro diagld del variador////
$.get("Iediagld.htm", function(result){
    $('#ediagld').val(result);
    var res13=result.replace('&#xd;','');
    $('#ediagld').val(res13); ————— Se muestra el valor de la variable "Ediagld" en el campo "diagld".
    if (res13==32607) {
        $('#ediagld').val("80A1");
    }
});————— Se muestra el valor de la variable "Ediagld" en el campo "diagld".

```

Figura 3.53 Función para mostrar los valores de los campos de la sección “Notificaciones”.

3.9.8 Gestión de Alarmas y Notificaciones

Alertas

Una de las funciones implementadas en la aplicación web, es la notificación ante una alerta establecida. Se ha impuesto una alerta, la cual se activa cuando el valor de la variable “vel actual”, sobrepase las 1400 RPM. Para notificar al usuario de esta alerta, se hace uso de una imagen que aparecerá a lado del recuadro del campo “Velocidad” de la sección “Lectura de valores”. Como primer paso, se deberá insertar la imagen posicionándola cerca del recuadro que muestra el valor de la variable “vel actual”. La figura 3.54 muestra el código implementado para insertar la imagen en la aplicación web.

En la figura 3.54, muestra por medio del recuadro negro, que se ha insertado la imagen “alarmaamarilla”, usando el elemento de html “img”. Luego por medio de la clase de css, “posicion2”, se ubica la imagen en el recuadro que recibe el valor del campo “Velocidad”.

```
149 |     <div class="col">
150 |       <label for="velocidad"><strong>Velocidad</strong> </label>
151 |       <label for="label-password" style="position: absolute; margin-top: 39px; margin-left: 110px;">rpm</label>
152 |       <input type="text" class="form-control parametrossalida" name="" id="VelMotor" value="">
153 |       <div class="btn">
154 |         <div class="position-absolute posicion2" id="alertaamarilla" style="display: none; ">
155 |           
156 |         </div>
157 |       </div>
158 |     </div>
```

Figura 3.54 Código implementado para insertar la imagen de alerta.

Por medio de la figura 3.55, se muestra el código a implementar para mostrar la imagen cuando se dé la condición impuesta. En el api denominado “Script”, donde se configuro para obtener y mostrar el valor de la variable “vel actual”, se establece la condición de que si el valor de la variable es mayor a 1400 (línea 25), se muestre la imagen “alamaramarilla”, que se insertó en el documento de html (línea 26). Caso contrario (línea 27), la imagen aparecerá oculta (línea 28).

```
22 |   $.get("IOvelocidadMotor.htm", function(result){
23 |     var res1=result.replace('&#x2d;', '');
24 |     $('#VelMotor').val(res1);
25 |     if (res1>=1400) {
26 |       $("#alertaamarilla").css('display','block');
27 |     } else {
28 |       $("#alertaamarilla").css('display','none');
29 |     }
30 |   });

```

Figura 3.55 Código implementado para definir las condiciones para mostrar la imagen de la notificación impuesta.

Notificación de sentido de giro y estado del motor

La aplicación diseñada tiene la funcionalidad de mostrar por medio de imágenes el sentido de giro y estado del motor. Esto como un método rápido para notificar al operador las configuraciones actuales del motor, como lo es su sentido de giro y su estado de encendido o apagado.

Como primer paso, será insertar las imágenes en la posición donde se mostrarán. La figura 3.56 indica el código para implementar las imágenes que muestran el estado del motor, mientras que la figura 3.57 indica el código para implementar las imágenes que indican el sentido de giro.

```
124 |   <!-- MUESTRO LA IMAGEN DEL DIAGRAMA P&ID SEGUN EL ESTADO DEL MOTOR -->
125 |   <div class="btn">
126 |     <div class="position-absolute posicion4" id="encendido" style="display: none; ">
127 |       
128 |     </div>
129 |     <div class="position-absolute posicion4" id="apagado" style="display: block;">
130 |       
131 |     </div>
132 |   </div>
```

Figura 3.56 Código implementado para insertar imágenes que indican el estado del motor.

```
133 <!-- MUESTRO UNA IMAGEN DE REFERENCIA DE ACUERDO AL SENTIDO DE GIRO DEL MOTOR -->
134 <div class="btn">
135     <div class="position-absolute posiciones5" id="giroizq" style="display: none; ">
136         
137     </div>
138     <div class="position-absolute posicion5 " id="giroder" style="display: none;">
139         
140     </div>
```

Figura 3.57 Código implementado para insertar imágenes que indican el sentido del motor.

Luego mediante las funciones implementadas en el archivo “Script”, se programará la función para mostrar las imágenes según sea el caso. Primero, se implementará la función que muestra la imagen del estado del motor, es decir si el motor esta apagado se mostrara una imagen que tiene la figura del motor de un color oscuro, mientras que si el motor este encendido se mostrara una imagen que posee un motor con relleno de color gris claro. Esto como un método para diferenciar el estado del motor.

La figura 3.58, muestra la función para mostrar la imagen del motor correspondiente, según sea el caso. Si la variable “HAB 1”, responsable de encender el motor tiene un valor de uno (línea 10), se muestra la imagen del motor encendido (relleno color gris) (línea 12). Caso contrario, se mostrará la imagen del motor apagado (relleno color oscuro) (línea 17).

```
8 $.get("IObtOnOff.htm", function(result){  
9     $('#statusMotor').val(result);  
10    if (result==1) {  
11        $('#statusMotor1').val("Encendido");  
12        document.getElementById('encendido').style.display="block"; → Se muestra la imagen del motor encendido  
13        document.getElementById('apagado').style.display="none";  
14    }  
15    else{  
16        $('#statusMotor1').val("Apagado");  
17        document.getElementById('apagado').style.display="block"; → Se muestra la imagen del motor apagado  
18        document.getElementById('encendido').style.display="none";  
19    }  
20});
```

Figura 3.58 Código implementado para mostrar la imagen según el estado del motor.

Por otro lado, la figura 3.59 muestra la función para mostrar una imagen dependiendo del sentido de giro configurado. Como se mostró en la tabla 3.5, la variable que controla el sentido de giro es la variable “EdSTW1”, esta variable al leerla retorna dos valores según el caso en el archivo “IOsentidoGiro.htm”. Cuando se seleccione un sentido horario de giro del motor, se recibirá un valor de “63” en el archivo .htm. Mientras que cuando se seleccione un sentido antihorario en el archivo .htm se recibe un valor de “127”.

Bajo lo antes mencionado, se programará las condiciones para mostrar la imagen correspondiente. En la figura 3.59 se muestra mediante la instrucción “if”, si el valor recibido es 63(línea 46), se muestra una imagen con una flecha que gira en sentido horario (línea 49). Si el valor recibido es 127 (línea 52), se muestra una imagen con una flecha que gira en sentido antihorario (línea 55). Si no se presentan las dos condiciones anteriores (línea 58), no se mostrará ninguna de las dos imágenes.

```

45     $.get("IOsentidoGiro.htm", function(result){
46         if (result==63) {
47             $('#sentido').val("Horario");
48             $('#giroizq").css('display','none');
49             $('#giroder").css('display','block'); → Se muestra la imagen que indica el sentido horario
50                                         del motor.
51         }
52         else if(result==127) {
53             $('#sentido').val("Antihorario");
54             $('#giroder").css('display','none');
55             $('#giroizq").css('display','block'); → Se muestra la imagen que indica el sentido antihorario
56                                         del motor.
57         }
58         else{
59             $('#sentido').val("Seleccione giro"); → En caso de que no se haya seleccionado un sentido de giro, en el
60                                         recuadro de texto del campo "Sentido de giro", se mostrara el mensaje
61                                         "Seleccione giro"
62         }
63     });

```

Figura 3.59 Código implementado para mostrar las imágenes que indican el sentido de giro del motor.

Alertas

El bloque función que controla el motor posee 4 parámetros o variables de salida que notifican al usuario la existencia de errores y fallas en el variador de frecuencia. En base a lo mencionado anteriormente, se ha implementado la función de alarmas en la aplicación web. Se ha implementado mediante HTML y JavaScript, que se active una alarma por medio de uso de imágenes y sonidos. La alarma se activará cuando se presente una de las fallas posibles que pueden ocurrir, denominada “Error de acceso al dispositivo periférico” o “Falla de conexión”.

Esta falla puede ocurrir cuando el variador de frecuencia se ha desconectado del PLC de control. Cuando esta falla ocurre, la variable “Estatus” y la variable “EdiagId” emitirán un valor único indicando el error de periferia. La variable “Estatus” enviará el valor de -31231 al archivo “IOestatus.htm”. Usando el valor de la variable “Estatus”, se programará la función en el api “Script”, con la finalidad de mostrar una imagen que indica un símbolo de alerta y reproduce un sonido el cual no dejara de reproducirse sino se ha arreglado la falla existente.

El primer paso a desarrollar, será insertar la imagen de alerta cuando se presente la falla en el variador de frecuencia. Esta imagen se la ubicara en la sección “Notificaciones”, en el campo “Status”, ya que aquí se recibe el valor de la variable “Estatus”.

La figura 3.60 muestra el código implementado (recuadro negro) para insertar la imagen en el campo “Status” de la sección “Notificaciones”. Mientras que usando la clase “class=“audio””, se colocara el audio a reproducirse.

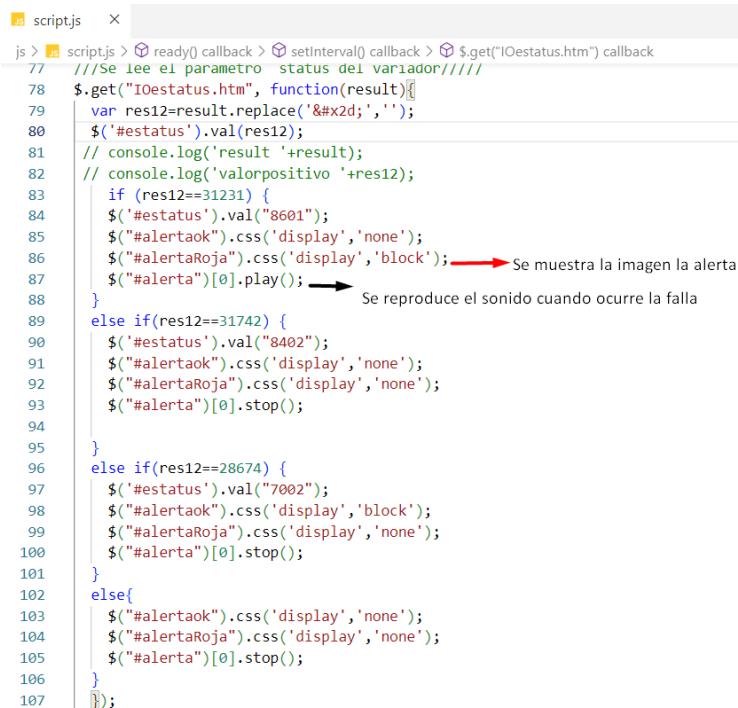
```

<div class="col">
    <label for="Estatus">Status </label>
    <input type="text" class="form-control parametroserror" name="" id="estatus" value="">
    <!-- ICONOS ALERTAS -->
    <div class="btn">
        <div class="position-absolute posicion7" id="alertaok" style="display: none; ">
            
        </div>
        <div class="position-absolute posicion6" id="alertaRoja" style="display: none; ">
            
        </div>
        <div class="audio">
            <audio id="alerta" class="audio">
                <source src="images/red-alert.mp3" type="audio/mp3">
            </audio>
        </div>
    </div>
</div>

```

Figura 3.60 Código implementado para insertar la imagen y el sonido ante una alarma.

Para finalizar la implementación de la alarma. En el archivo “script.js”, se programa la función que muestra la imagen y reproduce el sonido. La figura 3.61 muestra mediante la instrucción “if”, si la variable “Estatus” recibe el valor de 31231 (línea 83), se mostrará la imagen que indica la alarma (línea 86) y se reproducirá el sonido de alarma (línea 87). Caso contrario (línea 102), no se mostrará la imagen de alerta (línea 104) y no sonará ningún sonido (línea 105). En este caso para no trabajar con números negativos usando la instrucción `result.replace('-', '')`, se convierte un numero negativo a uno positivo.



```

js > script.js > ready() callback > setInterval() callback > $.get("IOestatus.htm") callback
77 //Se lee el parametro status del variador////
78 $.get("IOestatus.htm", function(result){
79     var res12=result.replace('-', '');
80     $('#estatus').val(res12);
81     // console.log('result '+result);
82     // console.log('valorpositivo '+res12);
83     if (res12==31231) {
84         $('#estatus').val("8601");
85         $('#alertaok').css('display','none');
86         $('#alertaRoja').css('display','block'); → Se muestra la imagen la alerta
87         $('#alerta')[0].play(); → Se reproduce el sonido cuando ocurre la falla
88     }
89     else if(res12==31742) {
90         $('#estatus').val("8402");
91         $('#alertaok').css('display','none');
92         $('#alertaRoja').css('display','none');
93         $('#alerta')[0].stop();
94     }
95     else if(res12==28674) {
96         $('#estatus').val("7002");
97         $('#alertaok').css('display','block');
98         $('#alertaRoja').css('display','none');
99         $('#alerta')[0].stop();
100    }
101  }
102 else{
103     $('#alertaok').css('display','none');
104     $('#alertaRoja').css('display','none');
105     $('#alerta')[0].stop();
106 }
107 });

```

Figura 3.61 Código implementado para mostrar la imagen y reproducir el sonido ante una alarma.

3.9.9 Implementación de la sección “Gráficos”

En esta sección se implementará dos gráficos que estarán en función de la variable de velocidad (“vel actual”). Esta sección se visualizará cuando se presione el botón “Gráficos” como se indicó en la tabla 3.8.

Como primer paso se creará el archivo html que contiene los gráficos y luego mediante el uso de las librerías “smoothie.js” y “JustGage”. Este archivo se denominará “grafico3.html”. La figura 3.62 muestra el archivo creado y su estructura principal.

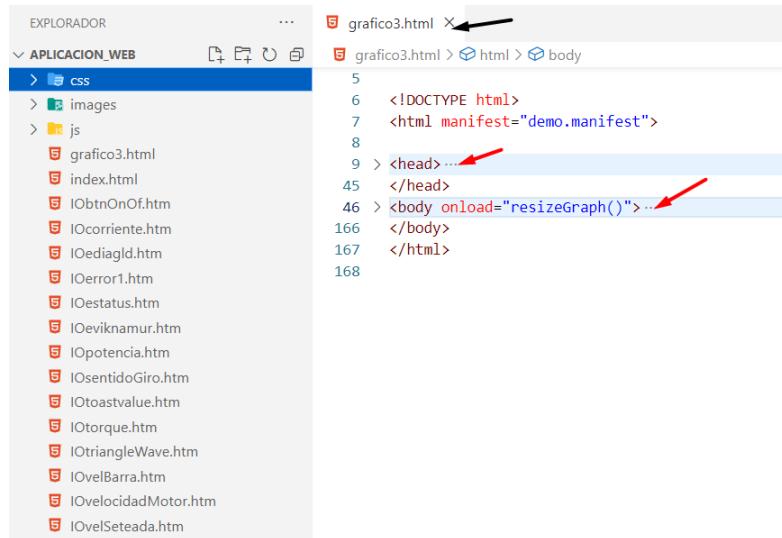


Figura 3.62 Archivo “grafico3.html”.

En la sección head, se vinculará cada una de las librerías necesarias para el funcionamiento de los gráficos. La figura 3.63, muestra cada una de las librerías de css empleadas por medio del elemento “link”.

```

grafico3.html ...
grafico3.html > html > body
8
9  <head>
10 <meta charset="utf-8">
11 <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
12
13 <title>Grafico Dinamico Tendencia</title>
14 <meta name="description" content="">
15 <meta name="viewport" content="user-scalable=no, width=device-width, initial-scale=1.0, maximum-scale=1.0">
16 <link rel="stylesheet" href="css/master.css">
17 <link rel="stylesheet" href="css/main.css">
18 <link rel="stylesheet" href="css/style.css" type="text/css" media="screen" />
19 <link rel="stylesheet" href="css/switch.css" type="text/css" media="screen" />
20 <link rel="stylesheet" href="css/toastr.css"/>
21 <link rel="stylesheet" href="css/toastr-responsive.css" />
22
23 <style>
24   body {
25     text-align: center;
26   }
27   #g1,#g2, #g3, #g4 {
28     width:200px; height:160px;
29     display: inline-block;
30     margin: 1em;
31   }
32   #logo {
33     float:right;
34     margin: 1em;
35   }
36   p {
37     display: block;
38     width: 450px;
39     margin: 2em auto;
40     text-align: left;
41   }
42 </style>

```

Figura 3.63 Configuración de la sección head del archivo “grafico3”.

Luego se configurará la sección body que contiene principalmente los elementos que posibilitan el funcionamiento de los dos tipos de gráficos.

El primer grafico está diseñado mediante la librería “smoothie.js”, el cual es un gráfico que está en función de la velocidad y del tiempo. Es decir, la gráfica recibe los valores de velocidad del motor y se desplaza en el tiempo. A este grafico se lo conoce como grafico “Tendencia”.

Mientras que el segundo gráfico, se crea mediante la librería “JustGage.js”. Este grafico posee una forma semicircular que se encuentra definida en un rango de entre 0 y un máximo 1440 RPM. Esto se debe a que está en función de los valores máximos y mínimos de la velocidad del motor. Este grafico poseerá un color

distinto según el valor que reciba dando un efecto de “llenado” en el gráfico.

Para crear el grafico denominado “Tendencia”, se insertará el elemento de html denominado “Canvas”, en el cual se pueden dibujar gráficos mediante el lenguaje de JavaScript. Luego mediante el elemento “select”, se definirán los controles para configurar las escalas del eje X (tiempo) y el eje Y (datos de velocidad). La figura 3.64 muestra cómo se creó el elemento “canvas” con un tamaño de 300 pixeles (height) de alto por 680 pixeles de ancho (width). Luego se han insertado los controles que modifican las escalas del X y Y de la gráfica tendencia.

```

<body onload="resizeGraph()">
<div id="main">
<h1>Tendencias</h1>
<div class="content" id="smoothie-div">
<div class="content-div">
<canvas class="smoothie" id="smoothie" height="300" width="680" ></canvas>
</div>
<div class="content-write">
    Seleccione el valor maximo(EjeY):
    <select onchange="smoothie.options maxValue=this.value;">
        <option value="1500">1500</option>
        <option value="1000" selected="true">1000</option>
        <option value="500">500</option>
        <option value="250">250</option>
    </select>
    Seleccione el valor minimo(EjeY):
    <select onchange="smoothie.options minValue=this.value;">
        <option value="40">40</option>
        <option value="20">20</option>
        <option value="10">10</option>
        <option value="0" selected="true">0</option>
    </select>
    Seleccione la escala de tiempo:
    <select onchange="changeTimeScale(this.value,smoothie,'smoothie');>
        <option value="30">30</option>
        <option value="20">20</option>
        <option value="10">10</option>
        <option value="5">5</option>
        <option value="2">2</option>
        <option value="1" selected="true">1</option>
    </select>
    <h4>Grafico Velocidad vs Tiempo</h4>
</div>

```

Figura 3.64 Código implementado para insertar el elemento canvas y los controles de los ejes de la gráfica “Tendencia”.

Luego se insertará el grafico “g1” con el elemento “div”, el cual es el segundo grafico con forma semicircular. Como paso siguiente, se colocarán los scripts necesarios para el funcionamiento de las gráficas, los cuales contienen las funciones principales para gestionar el funcionamiento de cada gráfico. La figura 3.65 muestra el proceso antes descrito.

```

<div id="g1"></div>
</div>
<script src="js/jquery-2.0.2.min.js"></script>
<script type="text/javascript" src="js/smoothie2.js"></script>
<script src="js/plugins.js"></script>
<script src="js/main.js"></script>
<script src="js/raphael.2.1.0.min.js"></script>
<script src="js/justgage.1.0.1.min.js"></script>
<script src="js/toastr.js"></script>

```

Figura 3.65 Scripts necesarios para el funcionamiento de los gráficos.

Continuando con el proceso de implementación, se programará un script el cual crea el segundo grafico por medio de la función “new JustGage ({});”. Seguido, con la función “refresh” se asignarán los valores de la velocidad al segundo gráfico. La figura 3.66, muestra la creación del segundo grafico (línea 98-104) y la función que asigna los valores correspondientes de velocidad a dicho gráfico (línea 111). Además, en la línea de código 110, se asigna a la variable “line 1”, los datos de velocidad que el grafico “Tendencia”, usa para el eje Y.

```

93 <script type="text/javascript">
94 $(document).ready(function(){
95 changeTimeScale(1,smoothie,'smoothie');
96 var stringValue;
97 var g1;
98 var g1 = new JustGage({
99   id: "g1",
100   min: 0,
101   max: 1440,
102   title: "Velocidad del motor",
103   label: "RPM"
104 });
105
106 $.ajaxSetup({ cache: false });
107 setInterval(function() {
108 $.get("IOvelocidadMotor.htm", function(result){
109   var valor=result.replace(' ','');
110   line1.append(new Date().getTime(),valor);
111   g1.refresh(parseInt(valor));
112 });
113 if (stringValue != '' && stringValue != result && result.trim() != ''){
114   toastr.options.timeout = 20000;
115   toastr.options.extendedTimeout = 0;
116   toastr.error(result.trim());
117 }
118 stringValue = result;
119 },1500);
120 });
121 </script>

```

Figura 3.66 Creación del grafico semicircular en función de la velocidad.

Mediante la creación de otro script, se implementará el primer grafico denominado tendencia mediante la función “new SmoothieChart”. Luego, se insertara la función `changeTimeScale()`, de la librería de “smoothie” para cambiar la escala de los ejes. La figura 3.67 muestra el código implementado para crear el grafico tendencia (línea 129), seguido se asigna los datos de velocidad al grafico(línea130).

```

123 <script type="text/javascript">
124
125   var line1 = new TimeSeries();
126   var line2 = new TimeSeries();
127   var x=0.0;
128
129 var smoothie = new SmoothieChart({minValue: 0, maxValue: 1500, labels: "#000000", millisPerPixel: 441, timeStamps: true, maxStorageTime
130 smoothie.addTimeSeries(line1, ({strokeStyle:"rgb(0, 0, 255)", lineWidth:2}));
131 smoothie.streamTo(document.getElementById("smoothie"), 1000);
132
133 function changeTimeScale(val, smooth, canvas) {
134   var c = document.getElementById(canvas);
135   var w = c.width;
136   var mspl = smooth.options.grid.millisPerLine;
137   var mspp = smooth.options.millisPerPixel;
138   var lpc = w * mspp / mspl;
139   var x = val * 60;
140   x = Math.ceil(x);
141   smooth.options.millisPerPixel = x / w * 1000;
142   smooth.options.grid.millisPerLine = w * smooth.options.millisPerPixel / lpc;
143 }
144 </script>

```

Figura 3.67 Código implementado para las funciones que crean el grafico “Tendencia”.

CAPÍTULO 4

4 Creación de la aplicación web usando Laravel para la gestión del almacenamiento y descarga de los datos del proceso

El presente capítulo se enfoca en la creación de la base de datos y la aplicación web que permite la gestión de datos adquiridos desde la aplicación creada en html del capítulo 3. La aplicación diseñada con el framework Laravel, permite funciones como; acceder a la sección Registros y sección Reportes, además de guardar en la base de datos de MYSQL, la información que se envía desde la aplicación de html.

La sección Registros tiene como función mostrar cada uno de los datos recopilados en la base de datos creada previamente en MYSQL. Por otro lado, en la sección Reportes, se generará dos tipos de reportes según la opción escogida por el usuario. El primer tipo de reporte denominado “visual”, mostrara un conjunto de datos definido por un rango de fecha y hora de inicio y fin que el usuario ha configurado previamente. Mientras que el segundo reporte, descarga un archivo de tipo csv compatible con Microsoft Excel con los datos definidos en el rango de fechas y horas del usuario.

Para la implementación de la aplicación web, es necesario un conjunto de programas como Visual Studio Code (editor de Laravel), Composer, XAMPP (MYSQL -Apache), siendo este último, el más importante. La figura 4.1 muestra una breve descripción del proceso de almacenamiento de los datos obtenidos por la aplicación desarrollada en html del capítulo 3. En la figura, se puede notar que la aplicación web desarrollada en Laravel, almacena los datos en la tabla de la base de datos de MYSQL. Cabe mencionar que la aplicación desarrollada en Laravel, solo recibirá datos de las variables de proceso, siempre y cuando el proceso esté activo y se haya accedido a la aplicación desarrollada en html desde el navegador web.

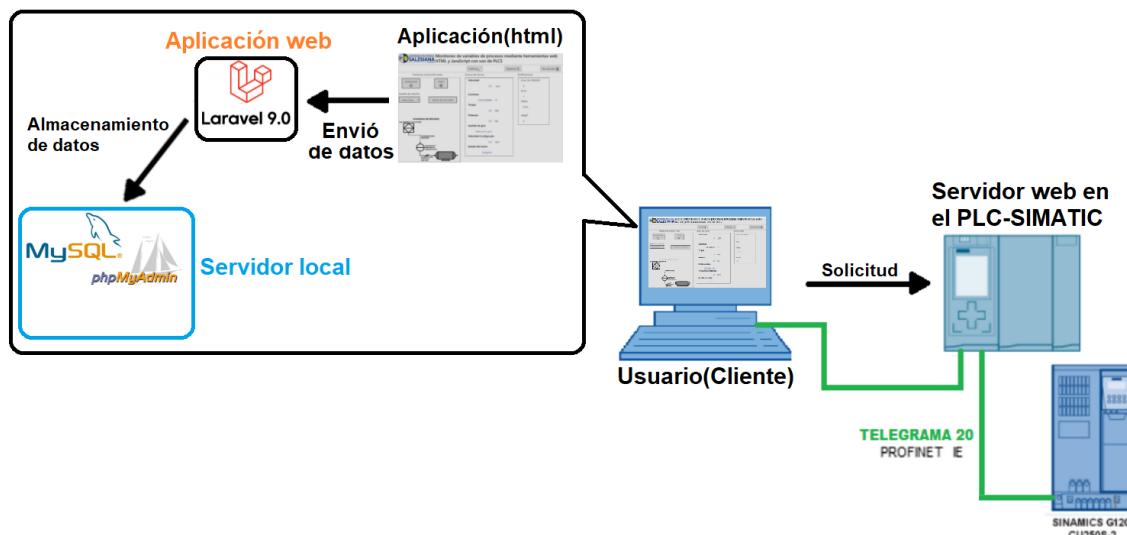


Figura 4.1 Proceso de almacenamiento de datos mediante la aplicación desarrollada en Laravel.

4.1 Conceptos generales:

4.1.1 Definición de una base de datos

Es una herramienta de recopilación y organización de información o datos estructurados. Esta información es proveniente de personas, productos, cosas, procesos etc. Generalmente, las bases de datos son controladas por sistemas de gestión de datos “DBMS”. Debido a que se necesita escribir y consultar información de una base de datos, se utiliza un lenguaje de consulta estructurada denominado SQL. El lenguaje SQL es utilizado en bases de datos con el objetivo de consultar, manipular, definir y controlar el acceso a los datos [28].

4.1.2 Tipos de bases de datos

Existen un gran número de tipos de bases de datos, sin embargo, un parámetro muy usado para organizaciones que utilizan bases de datos, es el modo de organización de datos. A continuación, se muestran algunos tipos de bases de datos [28]:

- Bases de datos relacionales

El contenido de la base de datos se organiza como un grupo de tablas estructuradas en filas y columnas. Es uno de los tipos de bases de datos más utilizado. Para la gestión de su contenido utilizan el lenguaje SQL y debido a su tecnología proporcionan una forma flexible y eficiente para acceder a la información estructurada.

- Bases de datos orientadas a objetos.

La información se presenta en forma de objetos, tal como lo hace el lenguaje de programación orientada a objetos.

- Bases de datos distribuidas

Esta base de datos está formada de dos o varias bases de datos, las cuales se ubican en sitios físicos o lógicos [29].

- Bases de datos NoSQL

Este tipo de bases de datos almacena y manipula datos no estructurados. Los datos no tienen que estar almacenados en tablas. La información guardada estará almacenada en diferentes máquinas del sistema, lo que proporciona una mayor escalabilidad y tolerancia a fallos [30].

- Bases de datos orientadas a grafos

Se almacena datos enfocados a entidades y las relaciones existentes entre entidades.

- Bases de datos OLTP

La característica de esta base de datos es su rapidez y su analítica que posibilita a un gran número de usuarios realizar numerosas transacciones.

Existen bases de datos de un uso poco común destinadas a funciones financieras, científicas o de algún tipo específico. Este tipo de bases de datos se enfocan en desarrollos tecnológicos y nuevos avances. Se pueden citar algunos tipos [28]:

- Bases de datos en la nube: es una recopilación de datos que reside en una plataforma de cloud, siendo esta de tipo tradicional o tipo DBaaS.
- Bases de datos de autogestión: este tipo de base de datos están basadas en la nube y por medio del machine learning, pueden gestionar por si solas funciones de seguridad, actualizaciones, copias de seguridad, entre otras tareas.
- Bases de datos multimodelo: combinan diversos tipos de modelos de bases de datos formando un único servidor integrado, permitiendo incorporar diferentes tipos de datos.
- Bases de código abierto: posee un código fuente de código abierto, tales bases de datos pueden ser de tipo NoSQL o SQL.

4.1.3 Servidor de base de datos

El servidor de base de datos sirve para gestionar las actualizaciones, almacenamiento, recuperación y administración de los datos de la base de datos. El software de un servidor permite exportar datos, configurar el acceso de los usuarios y respaldar la información de la base de datos [31].

Cómo elegir el tamaño de un servidor para una base de datos

Para elegir el tamaño de una base de datos se deben considerar aspectos como la naturaleza de las consultas

a la base de datos, frecuencia y tamaño de la base de datos. Sin embargo, no existen normas o reglas rígidas para determinar el tamaño del servidor para los datos, por lo que se puede tomar en cuenta los siguientes parámetros:

- Analizar y evaluar el rendimiento del servidor de la base de datos escogido para tener en cuenta aspecto en canto a término de almacenamiento e informática.
- Asegurar y considerar el suficiente espacio de almacenamiento para la base de datos, tanto para su operación actual y futura.
- Mantener los índices en la memoria RAM para evitar problemas de paginación.
- Considerar la CPU a emplear, la cual deberá soportar la memoria necesaria. En caso de tener un alto consumo de la CPU, emplear más CPU o mejorar las características de estas.
- Por temas de rendimiento y confiabilidad, usar la tecnología RAID para el servidor de la base de datos.

Servidores de base de datos de empresas más conocidos:

De los servidores más conocidos de base de datos se tienen:

- Oracle
- Sybase
- SQL Server
- DB2
- MySQL

4.2 Programas necesarios

4.2.1 Servidor de base de datos: MYSQL

MYSQL es un servidor de bases de datos cuyo software es de código abierto, permitiendo al usuario usarlo de manera libre y modificarlo a sus necesidades. Tiene la característica de poseer un modelo cliente-servidor, donde los clientes para acceder a los datos, se conectan a un base de datos relacional [32]. Este modelo de base de datos relacional utiliza tablas para organizar y almacenar datos estructurados. Por lo tanto, MYSQL se encargará de crear y administrar datos de la base de datos.

➤ Funcionamiento de MYSQL

Como se mencionó anteriormente, MYSQL se basa en un modelo cliente-servidor, donde una vez que se haya implementado la base de datos en MYSQL, un cliente realizará una consulta bajo el lenguaje SQL. El cliente se conectará al servidor por medio de una “red específica” con la finalidad de hacer una consulta donde luego el servidor proporciona la información solicitada mostrándola en la pantalla del usuario [32].

➤ Ventajas de MYSQL

Dado que MYSQL es uno de los servicios más populares posee algunas ventajas entre las cuales se puede mencionar [32]:

- Compatibilidad

MYSQL es compatible con plataformas como Ubuntu, Windows, Linux, proporcionando un gran rendimiento para el almacenamiento de grandes números de datos.

- Soporte comunitario

Con una comunidad muy activa y con gran apoyo, MYSQL proporciona un soporte para mejorar los recursos existentes y mejorar el sistema de base de datos que posee.

- Facilidad de uso

Este software posee una facilidad y requerimientos mínimos para lograr implementar una base de datos con gran rendimiento. Además, mediante interfaces gráficas como WorkBench o dbForge Studio, otorgan una simplicidad a MYSQL de empezar a usar bases de datos de mano de principiantes.

- o Seguridad

MYSQL ofrece varias funciones de seguridad, algunas de ellas de gran avance como la función Access Privilege System y de User Account Management proporcionando al usuario seguridad de los datos recopilados.

➤ *Tamaño máximo de una tabla de datos de MYSQL*

Generalmente se cree que el servidor de MYSQL tiene restricciones en cuanto a capacidad de espacio de memoria proporcionada a una tabla de una base de datos de MYSQL. Sin embargo, el espacio de una tabla está regido por las restricciones que tiene el sistema operativo sobre el tamaño de los archivos, mas no por limites internos de MYSQL [48]. Por ejemplo, en sistemas de 64 bits, el sistema operativo asigna un tamaño de varios terabytes para el funcionamiento de los archivos NTFS de MYSQL. Sin embargo, otro aspecto a considerar para el funcionamiento correcto de una tabla de MYSQL, será el tamaño de memoria RAM del equipo [49].

4.2.2 XAMPP

Es un servidor web multiplataforma diseñado para desarrolladores con la finalidad de crear y probar sus programas en un servidor web local. En este servidor se puede probar proyectos basados en diferentes tecnologías por medio de un servidor personal. Con esta plataforma, se puede corroborar el funcionamiento de proyectos basados en Apache (Servidor web), MYSQL (base de datos), programas con lenguajes de scripting/programación como lo son PHP y Perl [33][34].

Como se mencionaba anteriormente, XAMPP permite realizar diferentes pruebas de un sitio web, en un servidor o host local propio. El uso de un servidor local, ayuda a la ejecución de lenguajes de programación, dicho software puede ser instalado en un computador, de los cuales existen algunos tipos y su elección dependerá del lenguaje de programación a usar.

Otro de los usos fundamentales de un servidor local, se encuentra la función de almacenamiento de datos de sitios web dinámicos, este servidor podrá ser creado en un ordenador virtual o físico [33].

Entre las ventajas de XAMPP, es su sistema multiplataforma dada su compatibilidad con varias plataformas, para paquetes de MacOs, Linux y Windows. Su configuración es muy fácil y también su uso. Además, posee módulos esenciales como WordPress, MediaWiki, OpenSSL, siendo un programa que cuenta con una versión completa y una estándar.

4.3 Desarrollo de la base de datos

Luego de que se ha realizado una breve revisión acerca de los programas necesarios para la creación de la base de datos, como primer paso será la instalación del software multiplataforma XAMPP. Cuando se termine de instalar el programa XAMPP, como lo indica la figura 4.2, se procederá con el encendido de las plataformas APACHE y MYSQL.

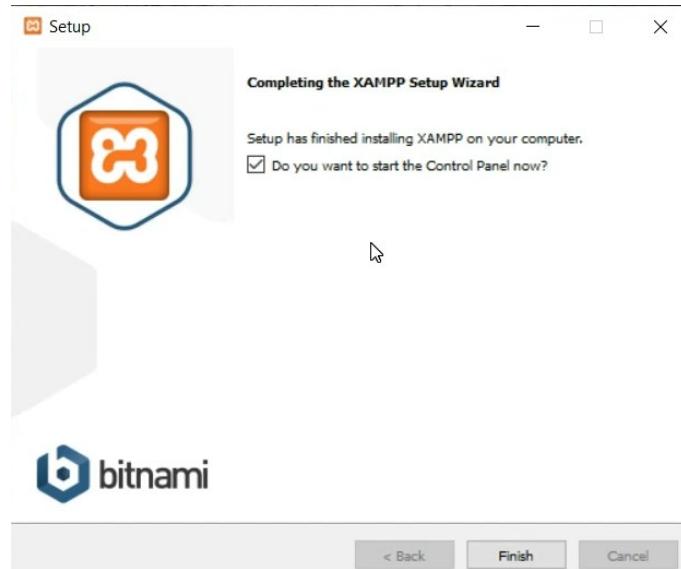


Figura 4.2 Instalación del software XAMPP.

Como se indicó anteriormente, se deberán encender los módulos Apache y MYSQL, para poder ejecutar la aplicación de Laravel, puesto que Apache permite ejecutar el lenguaje php usado por Laravel. Mientras que el programa MYSQL, gestionara y creara la base de datos para información recopilada desde la aplicación html.

La figura 4.3, muestra los módulos a encenderse por medio de los recuadros rojos, cuando se ha abierto el programa XAMPP Control Panel.

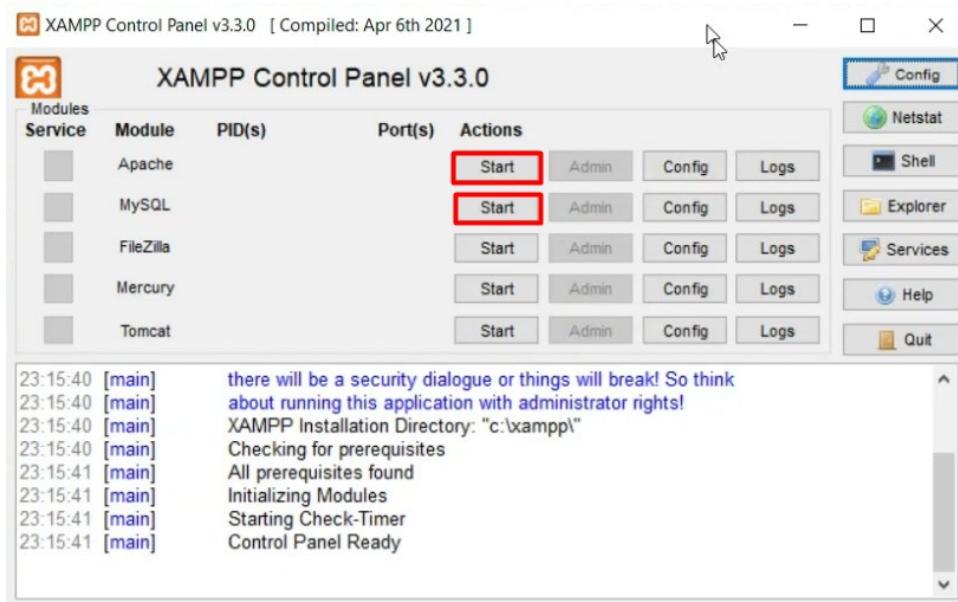


Figura 4.3 Encendido de los módulos APACHE y MYSQL en el software XAMPP.

• Creación de la base de datos

Como primer paso de gran importancia, será la creación de la base de datos por medio del programa o servidor MYSQL. Esta base de datos almacenara los datos provenientes de las variables del proceso. Para ello, se ha definido crear una base de datos de tipo relacional (SQL), ya que los datos estarán organizados mediante tablas. Con ayuda de un buscador web, se accederá al dominio; localhost/phpmyadmin/.

Luego del paso anterior, se visualizará la interfaz de la figura 4.4, donde para crear una nueva base de datos, se pulsará la opción “Nueva.”



Figura 4.4 Interfaz phpMyAdmin de MYSQL

Luego, se desplegará la ventana de la figura 4.5, donde se situará en la casilla señalada por la flecha con la finalidad de agregar un nombre a la nueva base de datos. La base de datos tendrá el nombre “_monitoreov1” y se creará una vez que se presione la opción crear.

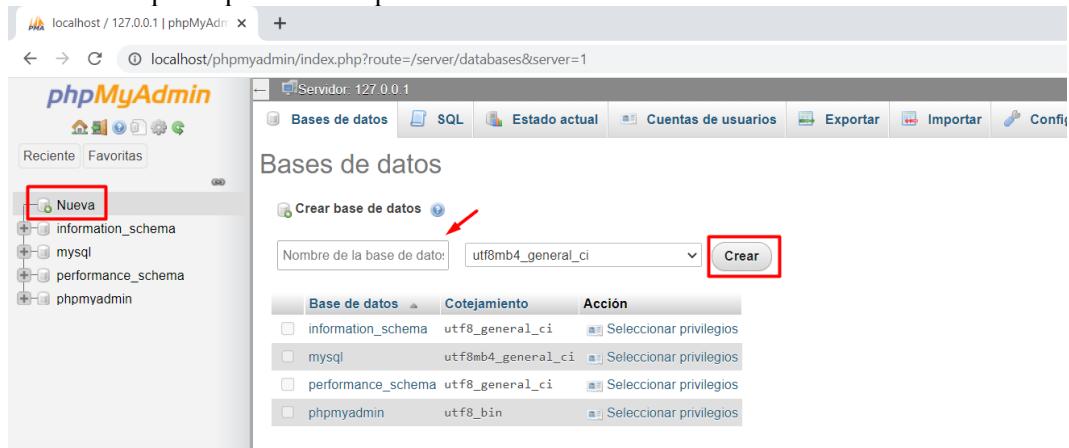


Figura 4.5 Creación de la base de datos en MYSQL.

Cuando se ha creado y definido la base de datos, se procederá a crear la tabla que contiene de manera organizada los datos de las variables del proceso. La figura 4.6, muestra la base de datos creada con el nombre “_monitoreov1” y muestra la opción “Crear tabla” para insertar una nueva tabla.



Figura 4.6 Base de datos creada en MYSQL.

Para crear la tabla en la base de datos, se posicionará en la casilla nombre, para definir el nombre de la tabla como “registros”. Luego se deberá definir el número de campos a guardar. El número de campos estarán

relacionado con las columnas de la tabla. Dado que se tienen 7 datos provenientes de las variables de proceso, se deberán definir 7 columnas.

Sin embargo, para saber la fecha en la que se ha creado y cargado cada uno de los 7 datos, se designara dos columnas más. Además, una columna adicional será agregada con la finalidad de indicar el número al que corresponde el conjunto de datos dentro de la tabla, generalmente conocido como “id”. Por lo tanto, se deberán designar 10 columnas a la tabla de la base de datos. La tabla 4.1 indica cada uno de los campos y su función en la tabla de datos denominada registros.

Tabla 4.1 Campos de la tabla de estructuración de la base de datos.

	Campo/Columnas	Función
1	id	Asigna un numero único al conjunto de datos almacenado dentro de la tabla de datos.
2	velocidad	Almacena el valor de la variable de velocidad actual del variador de frecuencia (variable: “vel actual”).
3	corriente	Almacena el valor de la variable de la corriente actual del variador (variable: “corriente actual”).
4	Torque	Almacena el valor de la variable del torque actual del variador (variable: “par actual”).
5	Potencia	Almacena el valor de la variable de potencia activa del variador (variable: “potencia actual”).
6	sentido	Almacena en forma de una cadena de caracteres, el sentido de giro actual en el que se encuentra el motor.
7	regulador1	Almacena el valor de la variable de velocidad previamente configurada del variador de frecuencia (variable: “RPMS”).
8	estadoMotor	Almacena en forma de una cadena de caracteres, el estado actual en el que se encuentra el motor.
9	created_at	Fecha en la que se ha creado el conjunto de datos.
10	updated_at	Fecha en la que se ha actualizado el conjunto de datos.

La figura 4.7, muestra el nombre asignado a la tabla de datos y el número de columnas que posee tabla. La tabla se creará al presionar la opción “Continuar”.



Figura 4.7 Creación de la tabla de datos “registros”.

Una vez que se ha creado las distintas columnas, se procederá a configurar los campos de cada columna y asignar el nombre de cada columna según el nombre asignado en la tabla 4.1. La figura 4.8, muestra cada columna de la tabla con su respectivo nombre.

Luego de asignar su nombre correspondiente a cada columna de la tabla, se procederá a configurar los siguientes campos:

- Tipo: se define el tipo de dato dependiendo de la información que se almacenara en la columna.
- Longitud/valores: se define la longitud máxima de valores a recibir en una columna.
- Cotejamiento: se define un determinado tipo de caracteres que se utilizaran en la base de datos.
- Atributos: se definen propiedades más precisas como la recepción de valores; solo positivos (UNSIGNED), solo números enteros o de punto flotante.
- A_I (AUTO_INCREMENT): El gestor de la base de datos, incrementa un valor de manera automática cuando se crea una secuencia de datos. Se usa para indexar un conjunto de datos. En este caso se configurará para el campo “id”, permitiendo asignar un numero único al conjunto de datos cuando se reciban cada uno de los valores de las variables.

La figura 4.8, muestra la configuración de cada campo de la tabla según la naturaleza de la información de la variable a recibir.

Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice	A_J	Comentarios
id	BIGINT	20	Ninguno		UNSIGNED	✓	—	✓	
velocidad	VARCHAR	255	Ninguno	utf8mb4_unicode_ci		✓	—	✓	
corriente	VARCHAR	255	Ninguno	utf8mb4_unicode_ci		✓	—	✓	
Torque	VARCHAR	255	Ninguno	utf8mb4_unicode_ci		✓	—	✓	
Potencia	VARCHAR	255	Ninguno	utf8mb4_unicode_ci		✓	—	✓	
sentidoGiro	VARCHAR	255	Ninguno	utf8mb4_unicode_ci		✓	—	✓	
regulador1	VARCHAR	255	Ninguno	utf8mb4_unicode_ci		✓	—	✓	
estadioMotor	VARCHAR	255	Ninguno	utf8mb4_unicode_ci		✓	—	✓	
created_at	TIMESTAMP		NULL			✓	—	✓	
updated_at	TIMESTAMP		NULL			✓	—	✓	

Figura 4.8 Configuración de las propiedades de los campos de la tabla registros.

Además, como se observa en la figura 4.9, se deberá definir el tipo de cotejamiento de la tabla (recuadro rojo), por lo cual, se escogerá la codificación de caracteres basado en la colación “utf8mb4_unicode_ci”. Por último, se deberá presionar la opción “Guardar”, para finalizar con la creación y configuración de la tabla de la base de datos.

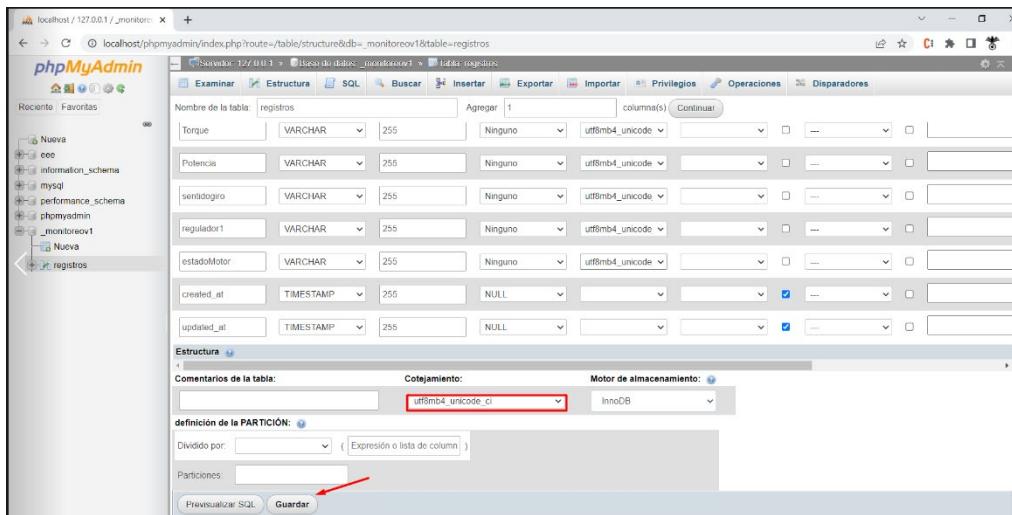


Figura 4.9 Configuraciones finales en la tabla de la base de datos.

Por medio de la figura 4.10, se puede observar las configuraciones antes realizadas a la tabla “registros”. Para ello, se ha seleccionado la tabla “registros” y se ha ubicado en la sección “Estructura”.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	bigint(20)		UNSIGNED	No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Más
2	velocidad	varchar(255)	utf8mb4_unicode_ci		No	Ninguna			Cambiar Eliminar Más
3	corriente	varchar(255)	utf8mb4_unicode_ci		No	Ninguna			Cambiar Eliminar Más
4	Torque	varchar(255)	utf8mb4_unicode_ci		No	Ninguna			Cambiar Eliminar Más
5	Potencia	varchar(255)	utf8mb4_unicode_ci		No	Ninguna			Cambiar Eliminar Más
6	sentido	varchar(255)	utf8mb4_unicode_ci		No	Ninguna			Cambiar Eliminar Más
7	regulador1	varchar(255)	utf8mb4_unicode_ci		No	Ninguna			Cambiar Eliminar Más
8	estadoMotor	varchar(255)	utf8mb4_unicode_ci		No	Ninguna			Cambiar Eliminar Más
9	created_at	timestamp			Sí	NULL			Cambiar Eliminar Más
10	updated_at	timestamp			Sí	NULL			Cambiar Eliminar Más

Figura 4.10 Propiedades de la tabla “registros”.

4.4 Creación de la aplicación para la gestión de la información de la base de datos usando LARAVEL

En esta sección, se implementará la aplicación con la cual se puede acceder y guardar los datos que se recopilaron en la base de datos antes creada. Cabe mencionar que los datos de la base de datos se obtienen desde la aplicación web desarrollada en el capítulo 3. Por ello, se ha designado el uso del framework o del “entorno de trabajo” denominado LARAVEL.

Por medio de este entorno, se implementará una aplicación web con la que se podrán acceder, guardar y visualizar los datos que se almacenan en la tabla de la base de datos “registros”. Además, con este entorno, se podrá gestionar la descarga o visualización de un reporte. Este reporte tiene la función de mostrar los datos en pantalla y/o descargar un archivo de extensión .xlsx, el cual contiene los datos dentro de un determinado rango de fecha y hora de inicio y fin.

4.4.1 Framework LARAVEL

Laravel es un framework basado en lenguaje PHP que permite el desarrollo de aplicaciones web de gran calidad. Laravel posee una estructura ordenada conformada por varios directorios y una arquitectura de clases, que posibilita separar su código según sus responsabilidades. Su característica principal es su modelo MVC (Modelo-Vista-Controlador), permitiendo eliminar la tarea de instanciar métodos y clases al usuario [35][36].

Para la ejecución de sus funciones emplea el uso del programa Artisan, el cual es una interfaz de comandos. Por medio de esta interfaz, se puede poner en funcionamiento la aplicación o detenerla, además de visualizar las rutas disponibles para la aplicación [36].

Entre algunas de sus ventajas, se encuentran; la facilidad y rapidez para su implementación, alta seguridad ya que no permite ingreso de amenazas o malware a la aplicación, uso de “migraciones” para la manipulación de bases de datos [50].

Como se mencionó anteriormente, Laravel posee una estructura ordenada de directorios, permitiendo trabajar de una manera fácil y flexible. Por medio del programa Composer, se puede crear un nuevo proyecto de Laravel que contiene los directorios para el funcionamiento de la aplicación [37]. La figura 4.11, muestra un ejemplo de un directorio de un proyecto implementado por Laravel

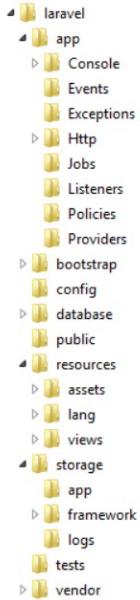


Figura 4.11 Directorio de una aplicación creada en Laravel [37].

4.4.2 Instalación del framework Laravel

Para instalar el framework Laravel se usará el programa Composer, ya que este programa gestiona las dependencias de PHP necesarias para el funcionamiento de la aplicación. Una vez que se descargue el programa Composer, durante la instalación es necesario especificar una versión de PHP. Para ello, usaremos la versión de PHP de la interfaz de XAMPP que se usó previamente para la instalación de MYSQL y gestionar la base de datos. La figura 4.12, muestra el proceso de definir la versión de PHP de XAMPP para el programa Composer.

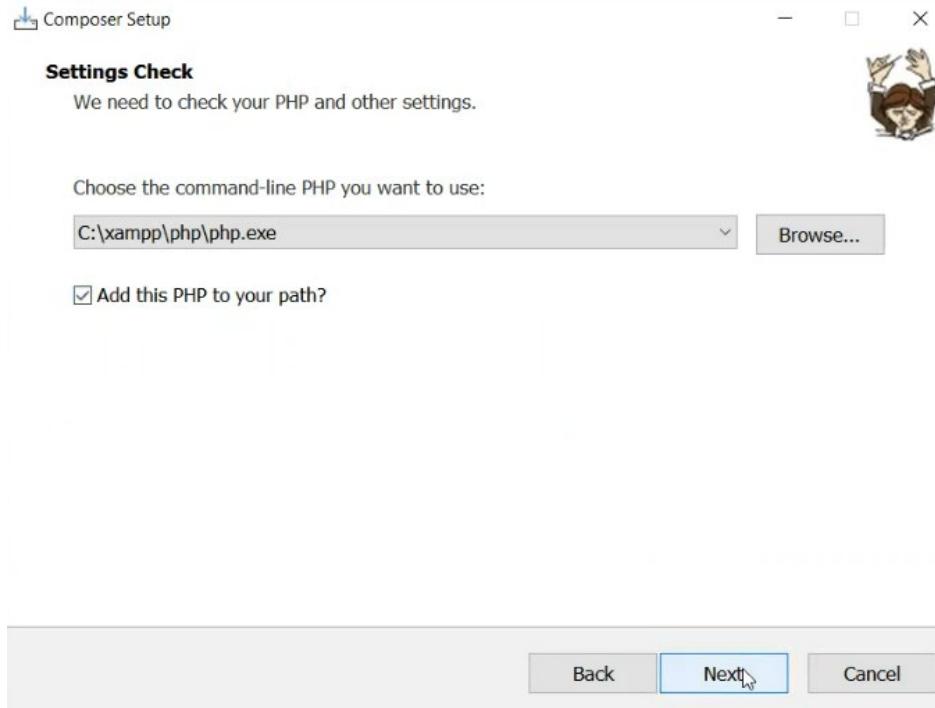


Figura 4.12 Selección de la versión de PHP para el software Composer.

Una vez que termine la instalación de Composer, se procederá con la instalación de LARAVEL por medio de la interfaz de comandos del “Símbolo del Sistema” del equipo. Como primer paso, es necesario ubicarse en la carpeta donde se ha instalado el programa XAMPP y dirigirse a la carpeta “htdocs”. La figura 4.13, muestra los comandos necesarios para ubicarse en la carpeta htdocs.

```
C:\Users\USER>cd C:\xampp\htdocs
C:\xampp\htdocs>
```

Figura 4.13 Ubicación en la carpeta “htdocs”.

4.4.3 Creación del proyecto LARAVEL

Para crear un nuevo archivo de LARAVEL, se deberá instalar una versión del software LARAVEL como indica la figura 4.14.

```
C:\xampp\htdocs>composer global require laravel/installer
```

Figura 4.14 Instalación del programa Laravel.

Cuando se halla ejecutado y finalizado el comando anterior, se ejecutará el comando de la figura 4.15, para crear un nuevo proyecto de Laravel denominado “monitoreo”.

```
C:\xampp\htdocs>laravel monitoreo
```

Figura 4.15 Creación de un proyecto nuevo en Laravel.

Ejecutado el comando anterior, se podrá verificar la creación del proyecto cuando se dirija a la carpeta htdocs, situándose primero en la carpeta principal de XAMPP. La figura 4.16, muestra el proyecto creado y por medio de la figura 4.17 cada una de los directorios del proyecto.

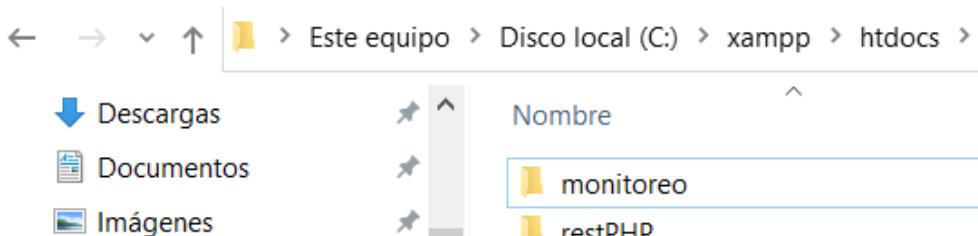


Figura 4.16 Proyecto de Laravel “monitoreo”.

	Nombre	
app	17/07/2022 22:28	Carpeta de archivos
bootstrap	14/03/2022 22:27	Carpeta de archivos
config	17/07/2022 22:26	Carpeta de archivos
database	14/03/2022 22:27	Carpeta de archivos
lang	14/03/2022 22:27	Carpeta de archivos
node_modules	14/03/2022 22:39	Carpeta de archivos
public	14/03/2022 22:27	Carpeta de archivos
resources	14/03/2022 22:27	Carpeta de archivos
routes	24/03/2022 23:59	Carpeta de archivos
storage	14/03/2022 22:27	Carpeta de archivos
tests	14/03/2022 22:27	Carpeta de archivos
vendor	17/07/2022 22:24	Carpeta de archivos
.editorconfig	14/03/2022 22:27	Archivo EDITORCO... 1 KB
.env	14/03/2022 22:33	Archivo ENV 1 KB
.gitattributes	14/03/2022 22:27	Documento de tex... 1 KB
.gitignore	14/03/2022 22:27	Documento de tex... 1 KB
.styleci	14/03/2022 22:27	Archivo YML 1 KB
artisan	14/03/2022 22:27	Archivo 2 KB
composer	17/07/2022 22:25	Archivo JSON 2 KB
composer.lock	17/07/2022 22:24	Archivo LOCK 303 KB
package	14/03/2022 22:27	Archivo JSON 1 KB
package-lock	14/03/2022 22:39	Archivo JSON 736 KB
phpunit	14/03/2022 22:27	Documento XML 2 KB
Procfile	14/03/2022 22:27	Archivo 1 KB
README	14/03/2022 22:27	Archivo MD 4 KB
webpack.mix	14/03/2022 22:27	Archivo JavaScript 1 KB

Figura 4.17 Directories del proyecto de Laravel “monitoreo”.

Como paso final, se pondrá en marcha o en servicio el proyecto “monitoreo”, para ello se situará en la carpeta del proyecto y se ejecutarán los comandos de la figura 4.18.

```
C:\xampp\htdocs>cd monitoreo
C:\xampp\htdocs\monitoreo>php artisan serve
```

Figura 4.18 Activación del servicio del proyecto monitoreo.

4.4.4 Acceso al proyecto creado

Cuando se crea un proyecto de Laravel, para acceder a la aplicación, se emplea una ruta que es proporcionada cuando se ejecute el comando de la figura 4.18. Esta ruta será: <http://127.0.0.1:80>, siendo el número 80, el puerto de comunicación. Cuando se ingrese la ruta antes mencionada en el navegador web, se visualizará la aplicación web de origen como lo indica la figura 4.19.

Esta aplicación será modificada usando el entorno de Visual Studio Code para las funcionalidades determinadas. Además, mediante el recuadro rojo de la figura 4.19, se puede observar la versión de Laravel y php empleadas.

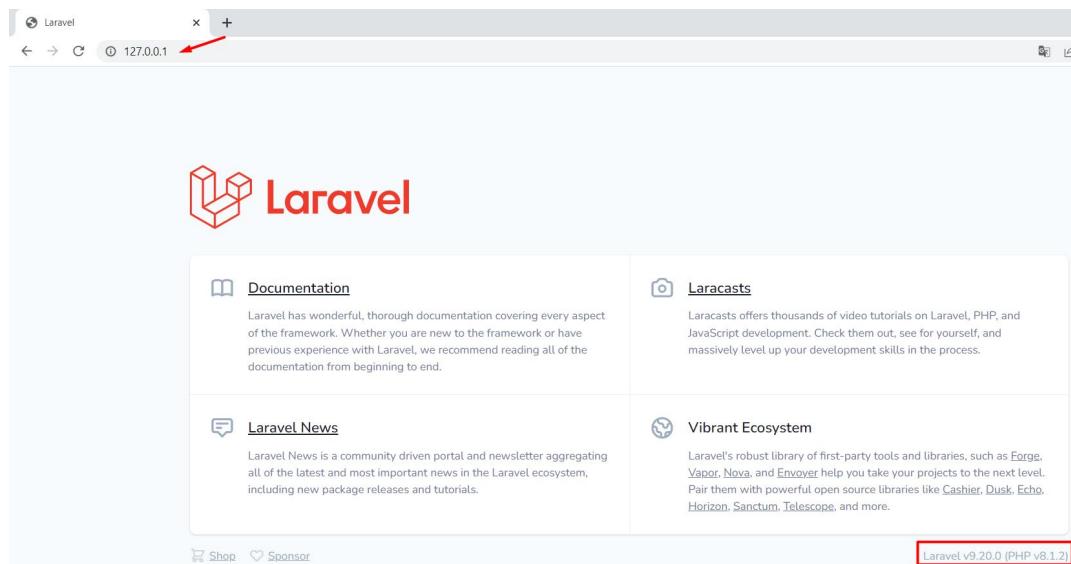


Figura 4.19 Acceso al proyecto de Laravel.

4.4.5 Creación de un virtual host para el acceso al proyecto de Laravel

Como se observó en la sección anterior, se accede a la aplicación web de Laravel por medio de una ruta específica. Sin embargo, es más fácil crear un host virtual por medio de XAMPP, para acceder a la aplicación de Laravel.

Para ello, como primero paso a seguir; se establecerá un dominio local el cual será ingresado en el navegador web. Este dominio tendrá como nombre “monitoreo.com”. Para establecer este dominio se modificará el archivo hosts de Windows. La figura 4.20, muestra la ruta a seguir para modificar el archivo hosts.

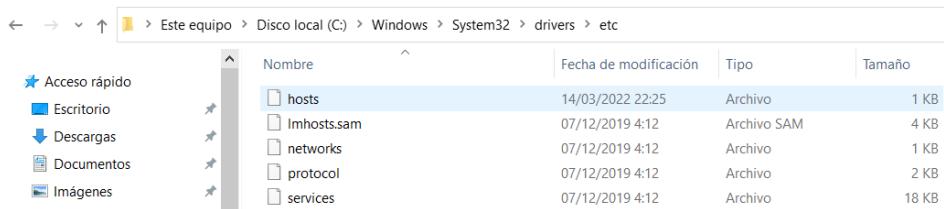


Figura 4.20 Archivo “hosts” de Windows.

La figura 4.21, muestra la configuración del dominio una vez que se asignado la url “monitoreo.com”.

```

hosts

C: > Windows > System32 > drivers > etc > hosts
1 # Copyright (c) 1993-2009 Microsoft Corp.
2 #
3 # This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
4 #
5 # This file contains the mappings of IP addresses to host names. Each
6 # entry should be kept on an individual line. The IP address should
7 # be placed in the first column followed by the corresponding host name.
8 # The IP address and the host name should be separated by at least one
9 # space.
10 #
11 # Additionally, comments (such as these) may be inserted on individual
12 # lines or following the machine name denoted by a '#' symbol.
13 #
14 # For example:
15 #
16 #      102.54.94.97      rhino.acme.com      # source server
17 #              38.25.63.10      x.acme.com        # x client host
18 #
19 # localhost name resolution is handled within DNS itself.
20 |   127.0.0.1          monitoreo.com
21 #   ::1                 localhost
22

```

Figura 4.21 Dominio establecido para acceso a la aplicación de Laravel.

Luego, como segundo paso; se configurará el host virtual en el archivo “httpd-vhosts.conf”, de XAMPP, ya que este programa permite establecer el host virtual para acceder al dominio antes configurado. La figura 4.22 muestra el código implementado para establecer el host virtual una vez que se ha accedido al directorio “\xampp\apache\conf\extra”. Cabe mencionar que este host debe apuntar al mismo puerto asignado por XAMPP para abrir el proyecto de Laravel, es decir el puerto 80.

```
httpd-vhosts.conf
C: > xampp > apache > conf > extra > httpd-vhosts.conf

21  #
22  # VirtualHost example:
23  # Almost any Apache directive may go into a VirtualHost container.
24  # The first VirtualHost section is used for all requests that do not
25  # match a ##ServerName or ##ServerAlias in any <VirtualHost> block.
26  #
27  ##<VirtualHost *:80>
28      ##ServerAdmin webmaster@dummy-host.example.com
29      ##DocumentRoot "C:/xampp/htdocs/dummy-host.example.com"
30      ##ServerName dummy-host.example.com
31      ##ServerAlias www.dummy-host.example.com
32      ##ErrorLog "logs/dummy-host.example.com-error.log"
33      ##CustomLog "logs/dummy-host.example.com-access.log" common
34  ##</VirtualHost>
35
36  <VirtualHost *:80>
37      ServerAdmin admin@monitoreo.com
38      DocumentRoot "C:/xampp/htdocs/monitoreo/public"
39      ServerName monitoreo.com
40      ErrorLog "logs/dummy-host2.example.com-error.log"
41      CustomLog "logs/dummy-host2.example.com-access.log" common
42  </VirtualHost>
```

Figura 4.22 Configuración del host virtual para el acceso a la aplicación web.

Finalmente, para corroborar el acceso a la interfaz principal del proyecto de Laravel, se ha ingresado a la url “monitoreo.com”, mostrándose la portada de la figura 4.23.

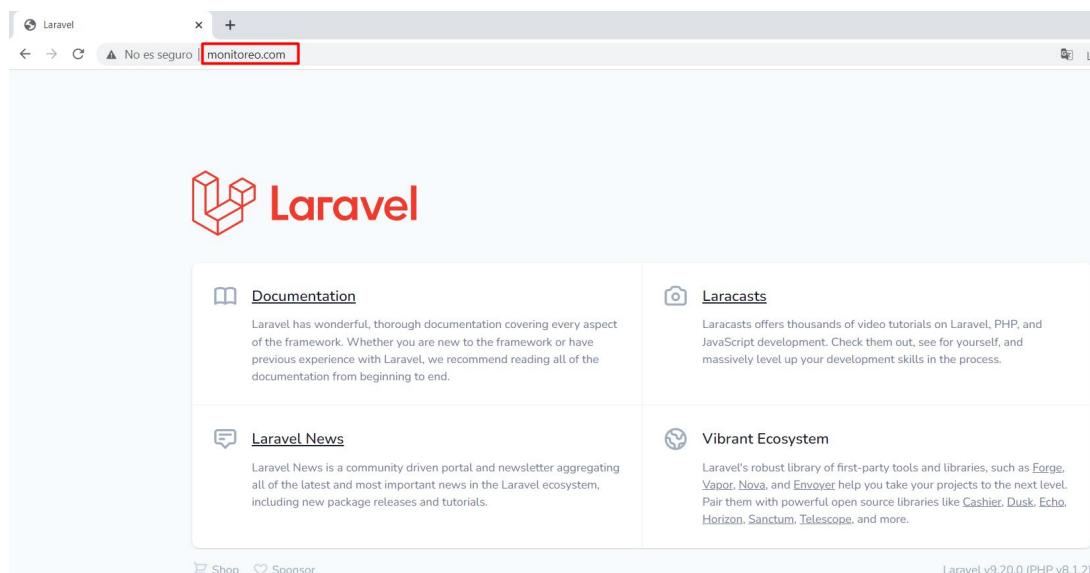


Figura 4.23 Acceso a la aplicación web usando un host virtual.

4.5 Implementación de la aplicación web desarrollada en LARAVEL

Para implementar la aplicación web que permitirá guardar, acceder y visualizar los datos que se recopilan en la base de datos de MYSQL, es necesario seguir un proceso determinado. Este proceso tiene un patrón específico y por medio del software Visual Studio Code, se editará el proyecto de Laravel para conseguir implementar la aplicación web que cumpla con los requerimientos deseados. El uso de Visual Studio Code permitirá editar el proyecto monitoreo implementado a través de Laravel, dado que esta interfaz permite codificar proyectos basados en lenguaje PHP.

Durante el desarrollo de la aplicación, se editarán algunos campos de los directorios del proyecto “monitoreo”, siguiendo un orden definido. Como primer paso se editarán los “Modelos”, que gestionan los datos de la aplicación (información recolectada en la tabla de MYSQL). Seguido, se editarán los “controladores” del proyecto, que definen la lógica de acceso a los datos. Por último, se configurará las “Vistas” del proyecto, donde se determina la presentación final de la aplicación al usuario [38].

4.5.1 Implementación del “Modelo” de la aplicación.

Los modelos es uno de los componentes principales de la aplicación, donde se gestiona el acceso o modificación de los datos de las variables de proceso [39]. Otras de las funciones que tiene el “Modelo” del proyecto, es la obtención, inserción y validación de los datos, esta última acción, ayuda a definir la forma correcta en la que los datos se guardaran en la tabla configurada previamente en MYSQL.

La función principal de este modelo, será la de interactuar con la tabla de la base de datos creada previamente. En el modelo, se implementará la función para agregar cada dato proveniente de la aplicación web de html a la tabla de datos de MYSQL.

Conexión de la base de datos de MYSQL y Laravel

Para poder guardar y acceder a los datos de la tabla registros a través de "modelo" creado en Laravel, es necesario vincular la base de datos “_monitoreov1” de MYSQL por medio de la configuración del archivo .env. Este archivo se encontrará en el directorio “config” del proyecto. La figura 4.24, muestra el archivo .env por medio la flecha roja. Mientras que, mediante el recuadro rojo, se indican los campos configurados para conexión entre el proyecto de Laravel y la base de datos “_monitoreov1”. Cabe mencionar que el “DBPORT=3306”, es el puerto asignado por XAMPP para acceder a la base de datos de MYSQL.

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:pQ9sDAXD4fLjdjpPa2ktMfvkmk9smEKdAKT15t1xJTM=
APP_DEBUG=true
APP_URL=http://localhost

LOG_CHANNEL=stack
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=_monitoreov1
DB_USERNAME=root
DB_PASSWORD=

BROADCAST_DRIVER=log
CACHE_DRIVER=file
FILESYSTEM_DISK=local
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
SESSION_LIFETIME=120
```

Figura 4.24 Conexión entre la base de datos de MYSQL y Laravel.

4.5.2 Migraciones

Dado que la base de datos “_monitoreo”, la cual almacena la información de las variables del proceso, será gestionada por Laravel, es necesario ejecutar el proceso denominado “Migraciones”. Por medio de las migraciones, Laravel permite gestionar los campos de la base de datos, como insertar información en las columnas de la tabla o compartir el esquema de la base de datos [40]. Al ejecutar las migraciones, Laravel crea a partir de la tabla de la base de datos diseñada en MYSQL (“registros”), en una tabla basada en lenguaje php [41].

Para ejecutar las migraciones se hará uso del comando de la figura 4.25 una vez que se situé en el en la carpeta del proyecto creado por Laravel (C:\xampp\htdocs\monitoreo).

```
C:\Users\USER>cd C:\xampp\htdocs\monitoreo  
C:\xampp\htdocs\monitoreo>php artisan migrate
```

Figura 4.25 Migraciones realizadas en el proyecto de Laravel.

Luego de ejecutar el proceso de Migraciones, en la base de datos de MYSQL, se crearán por defecto las tablas; failed_jobs, migrations, password_resets, personal_access_tokens, users. La figura 4.26, muestra las tablas creadas, que verifican que el proceso de migraciones se ha realizado correctamente.

Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Residuo a depurar
failed_jobs	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
migrations	Examinar Estructura Buscar Insertar Vaciar Eliminar	5	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
password_resets	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
personal_access_tokens	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	utf8mb4_unicode_ci	48.0 KB	-
registros	Examinar Estructura Buscar Insertar Vaciar Eliminar	27,120	InnoDB	utf8mb4_unicode_ci	2.5 MB	-
users	Examinar Estructura Buscar Insertar Vaciar Eliminar	1	InnoDB	utf8mb4_general_ci	32.0 KB	0 B

Figura 4.26 Tablas generadas por defecto al ejecutar las migraciones.

Por otro lado, las tablas de migración generadas en lenguaje php, se guardarán en la carpeta database en la sección “migrations” del proyecto de Laravel. Usando Visual Studio Code, se puede acceder a la tabla de migración denominada “registros”, la cual es la tabla de datos de MYSQL estructurada en lenguaje php.

En la figura 4.27, se puede observar la tabla de la base de datos “Registros” en lenguaje php, donde se visualiza una de las funciones principales. Por medio de la función Up, se podrá asignar y actualizar los campos de la tabla de MYSQL. La función creará una tabla por medio del método “Schema::create” y luego con la función (Blueprint \$table) se llenará cada instancia o columna de la tabla [41][42]. Por otro lado, la función down, revierte las acciones de la función Up. En la figura 4.28, se puede observar con detalle las dos funciones antes descritas.

```

2022_03_11_191418_create_registros_table.php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('registros', function (Blueprint $table) {
            $table->id();
            $table->string('velocidad');
            $table->string('corriente');
            $table->string('Torque');
            $table->string('Potencia');
            $table->string('sentido');
            $table->string('regulador1');
            $table->string('estadoMotor');

            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('registros');
    }
}

```

Figura 4.27 Tabla de migración “registros”.

```

2022_03_11_191418_create_registros_table.php

public function up()
{
    Schema::create('registros', function (Blueprint $table) {
        $table->id();
        $table->string('velocidad');
        $table->string('corriente');
        $table->string('Torque');
        $table->string('Potencia');
        $table->string('sentido');
        $table->string('regulador1');
        $table->string('estadoMotor');

        $table->timestamps();
    });
}

/**
 * Reverse the migrations.
 *
 * @return void
 */
public function down()
{
    Schema::dropIfExists('registros');
}

```

Figura 4.28 Funciones up y down de la clase “registros”.

4.5.3 Creación del modelo “Registros”

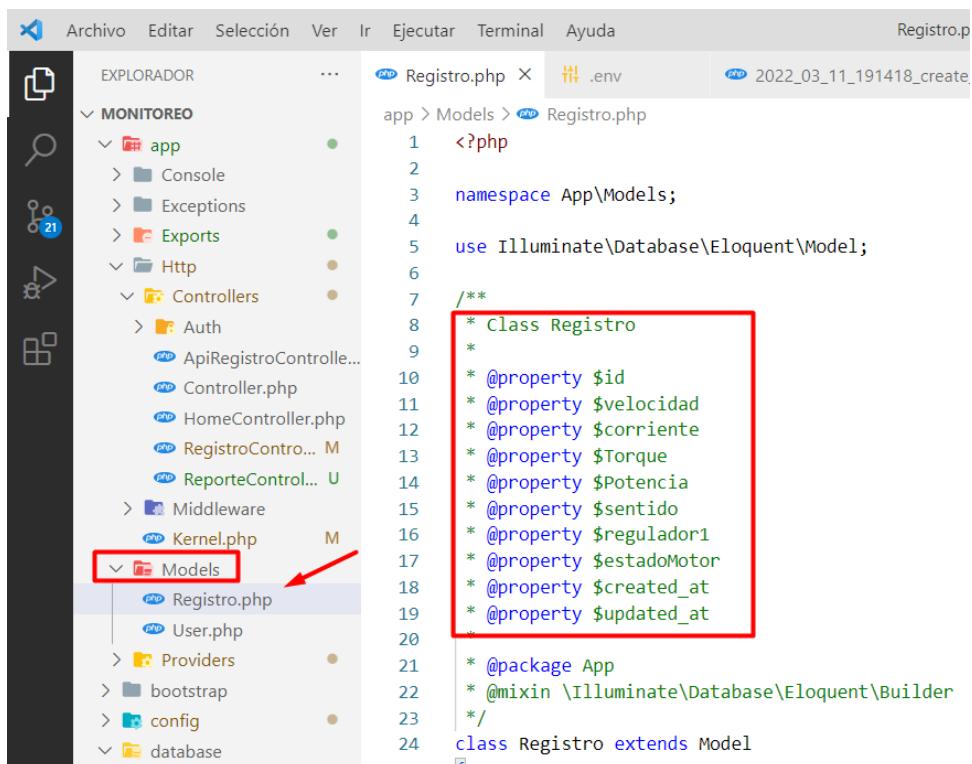
En esta parte, se creará el modelo Registros el cual podrá interactuar con la base de datos configurada previamente en MYSQL. Por medio de este modelo se podrá insertar, consultar y mostrar la información que se almacena en la base de datos. Para ello, es necesario, abordar el término “Eloquent ORM” (Mapeo Objeto Relacional), esta función es utilizada por Laravel, para manejar la base de datos, donde la tabla está definida por una clase modelo y cada campo de dicha tabla, será una instancia de la clase [43].

Como primer paso, se creará el modelo “Registro” por medio del comando de la figura 4.29.

```
C:\Users\USER>cd C:\xampp\htdocs\monitoreo
C:\xampp\htdocs\monitoreo>php artisan make:model Registro
```

Figura 4.29 Creación del modelo Registro en Laravel.

Por medio de este modelo “Registro”, se pretende enviar y recibir la información a la base de datos “_monitoreov1”. Usando el nombre Registro, el modelo podrá identificar la tabla “registros” de la base de datos “_monitoreov1”. Para mostrar los datos de cada una de las variables en la aplicación WEB, es necesario definir cada uno de los parámetros o campos de la tabla “registros” en el modelo “Registro”. La figura 4.30, muestra el modelo “Registro” creado por medio de Visual Studio Code. En la sección de inicio del modelo, se ha definido cada uno de los parámetros o campos de la tabla por medio de la sentencia `@property`.



```
Registro.php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;

/**
 * Class Registro
 *
 * @property $id
 * @property $velocidad
 * @property $corriente
 * @property $Torque
 * @property $Potencia
 * @property $sentido
 * @property $regulador1
 * @property $estadomotor
 * @property $created_at
 * @property $updated_at
 */
class Registro extends Model
```

Figura 4.30 Modelo “Registro”.

Luego, en la figura 4.31, se observa la clase Registro, donde por medio del array, se verifica o se valida que cada campo devuelva un parámetro o dato permitido. En caso, de presentarse un error, la sentencia “required”, emitirá un error anunciando que se enviado un campo vacío.

Luego, usando la sentencia “protected \$perPage = 20;”, se define la cantidad de resultados a visualizar por página en la aplicación WEB, es decir el número de conjunto de datos a mostrar provenientes de la tabla de la base de datos. Estos resultados serán mostrados en pantalla una vez que se implemente la sección “Registros” de la aplicación web de Laravel.

Por último, usando la propiedad “fillable” (recuadro azul), se configura para asignar el valor o información a cada uno de los campos de la tabla de migración “registros”. Esta información será recibida desde la base de datos de MYSQL, cuando se realice una consulta desde Laravel.

```

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
Registro.php - monitoro - Visual Studio Code
Registro.php M X .env 2022_03_11_191418_create_registros_table.php ReporteController.php index.blade.php

EXPLORADOR
MONITOREO
app
Console
Exceptions
Exports
Http
Controllers
Auth
ApiRegistroController.php
Controller.php
HomeController.php
RegistroController.php
ReporteController.php
Middleware
Kernel.php
Models
Registro.php
User.php
Providers
bootstrap
config
database

app > Models > Registro.php
24 class Registro extends Model
25
26
27 static $rules = [
28     'velocidad' => 'required',
29     'corriente' => 'required',
30     'Torque' => 'required',
31     'Potencia' => 'required',
32     'sentido' => 'required',
33     'regulador1' => 'required',
34     'estadoMotor' => 'required',
35 ];
36
37 protected $perPage = 20;
38
39 /**
40 * Attributes that should be mass-assignable.
41 *
42 * @var array
43 */
44 protected $fillable = ['velocidad', 'corriente', 'Torque', 'Potencia', 'sentido', 'regulador1', 'estadoMotor'];
45
46

```

Figura 4.31 Clase Registro.

4.5.4 Implementación del “controlador” de la aplicación.

Continuando con otro de los elementos a configurar en el proyecto de Laravel, será la configuración del o los controladores. El controlador es un elemento que gestiona la lógica de las peticiones http que llegan por parte del usuario [44]. Para entender de mejor manera, el funcionamiento del “Controlador”, es necesario entender de antemano, el concepto de “rutas”.

Como se observa en la figura 4.32, se muestra el patrón “Modelado-Controlador-Vista” que usa Laravel para el funcionamiento de las aplicaciones WEB. Cuando un usuario emite una solicitud para acceder a la aplicación web, se emite una petición “http” de tipo GET, la cual es recibida en el archivo routes.php del proyecto de Laravel. Por ello, las rutas con necesarias para manejar el flujo de las solicitudes http hacia y desde el cliente [45].

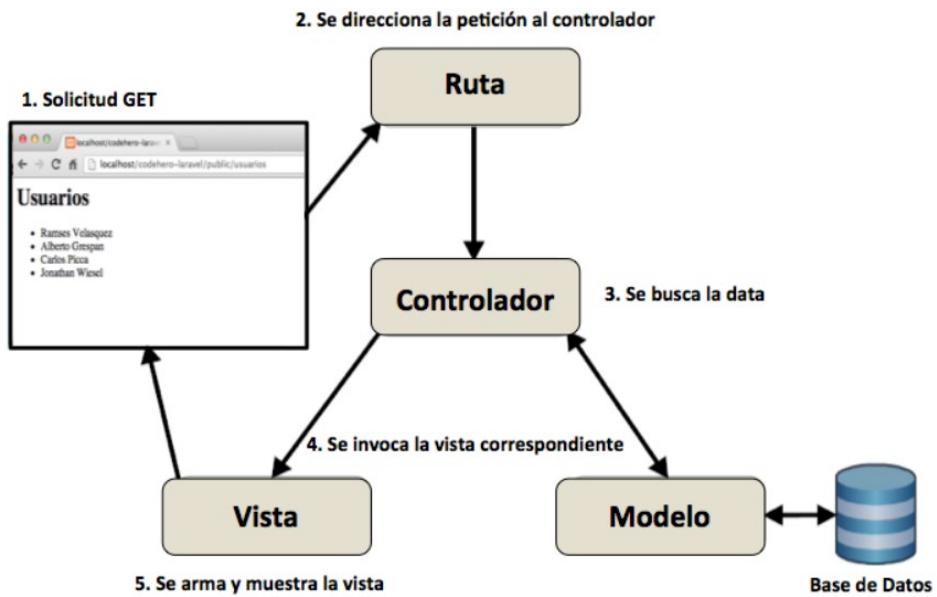


Figura 4.32 Patrón de funcionamiento de una aplicación en Laravel [45].

En Laravel, al crear un proyecto se implementará por defecto cuatro archivos que albergaran las rutas de la aplicación, siendo de gran importancia los archivos: routes/api.php y routes/web.php. En el archivo “routes/api.php”, se configurarán las rutas que la aplicación tendrá hacia las “API”. En este caso, por medio de una API incrustada en la aplicación html del capítulo 3, se enviarán los datos del proceso a la aplicación web de Laravel, con el fin de guardar los datos en la tabla de MYSQL. Por ello, es necesario definir una ruta en routes/api.php, para poder guardar los datos del proceso provenientes desde una API desarrollada en JavaScript.

Mientras que el archivo “routes/web.php”, se configuraran las rutas de la aplicación web que son ingresadas desde un navegador web para poder acceder a una sección determinada o un recurso particular [46]. En este proyecto se establecerán dos rutas en el archivo “routes/web.php”, para acceder a la sección “Registros” y a la sección “Reportes” de la aplicación web.

Adentro de los archivos routes.php, se definirían las rutas de tipo GET y POST con la finalidad de adquirir los datos de la base de datos desde la aplicación implementada desde Laravel. Es aquí, donde es necesario el uso de los controladores. Al establecer una ruta específica, se ejecutarán los diversos métodos de un controlador.

En este caso, para el funcionamiento de la aplicación web, se definirá dos rutas en el archivo “routes/api.php”. La principal función de estas rutas, será la de obtener los datos provenientes de la aplicación WEB diseñada en HTML, del capítulo 3 desde la aplicación de Laravel. Luego por medio de otra ruta, la aplicación de Laravel enviara la información recopilada de las variables de proceso a la tabla de la base de datos que se implementó en MYSQL.

Sin embargo, para determinar una ruta específica en Laravel, será necesario como primera instancia, la creación del constructor que este a su vez, interactúa con la tabla de la base de datos por medio del modelo “Registro”.

- Implementación del controlador “ApiRegistroController”

Por medio de este controlador, se definirá la lógica de respuesta ante las peticiones que se reciben de los archivos route.php, es decir dependiendo de la petición recibida se efectuará una determinada acción por medio del controlador ApiRegistroController.

El primer paso a desarrollar será la creación del controlador por medio de la interfaz de comandos. La figura 4.33, muestra el comando a ejecutar para crear el controlador.

```
C:\Users\USER>cd C:\xampp\htdocs\monitoreo  
C:\xampp\htdocs\monitoreo>php artisan make:controller ApiRegistroController
```

Figura 4.33 Creación del controlador ApiRegistroController en Laravel.

Luego para visualizar el archivo creado se dirigirá a la sección de HTTP y al directorio “Controllers” del proyecto. La figura 4.34, muestra el archivo creado “ApiRegistroController”. Además, se han creado dos controladores más; “RegistroController”, “ReporteController”, cuyo uso se emplearán más adelante.

En la figura 4.34, se pude observar dos funciones implementadas; guarda e index. La función guarda dentro de la clase ApiRegistroController, podrá guardar los datos provenientes de la aplicación web diseñada en HTML en la tabla de MYSQL, “registros” mediante el uso del modelo “Registro”.

Para finalizar el proceso de guardar los datos mediante el controlador ApiRegistroController, se deberá definir las rutas en el archivo api.php, dado que las solicitudes para guardar datos provienen de una API desarrollada en JavaScript y estas, serán recibidas en este archivo.

The screenshot shows the Visual Studio Code interface. The left sidebar displays a tree view of the project structure under 'EXPLORADOR'. A red arrow points to the 'Controllers' folder, which contains several files: ApiRegistroController.php, Controller.php, HomeController.php, RegistroController.php (highlighted with a blue border), and ReporteController.php. The main editor window shows the code for ApiRegistroController.php. A red box highlights the file tab 'ApiRegistroController.php X'. The code is as follows:

```

1 <?php
2
3 namespace App\Http\Controllers;
4
5 use App\Models\Registro;
6 use Illuminate\Http\Request;
7
8 class ApiRegistroController extends Controller
9 {
10     /**
11      * public function guarda(Request $request){
12
13         request()->validate(Registro::$rules);
14
15         $registro = Registro::create($request->all());
16
17         return response()->json($registro, 200);
18     }
19
20     public function index()
21     {
22         $registros = Registro::all();
23         return response()->json($registros, 200);
24     }
}

```

Figura 4.34 Controlador ApiRegistroController.

Para gestionar las solicitudes que permiten guardar el conjunto de datos desde la aplicación en HTML, se deberá implementar dos rutas; Route::post y Route::get. La ruta post, permite guardar valores mediante una solicitud “http post” proveniente de la aplicación web en HTML. La implementación de las rutas post y get, se las puede visualizar en la figura 4.35 por medio del recuadro negro.

The screenshot shows the Visual Studio Code interface. The left sidebar displays a tree view of the project structure under 'EXPLORADOR'. A red arrow points to the 'routes' folder, which contains api.php. The main editor window shows the code for api.php. A red box highlights the route definitions at the bottom of the file:

```

Route::post('guarda',[ApiRegistroController::class,'guarda']);
Route::get('todo',[ApiRegistroController::class,'index']);

```

Figura 4.35 Ruta “post” y “get” del controlador ApiRegistroController.

Al implementar la ruta “guarda” con el método Route::post, se define una ruta http, la cual es: 'http://monitoreo.com/api/guarda'. Cabe mencionar que la parte “monitoreo.com”, es el dominio que permite el acceso a la aplicación web de Laravel, mientras que la sección “api”, es la sección api.php que contiene la ruta “guarda”.

Al definirse esta ruta, la “API” de la aplicación de html del capítulo 3, encontrara el “camino” para enviar los datos obtenidos de las variables del proceso por medio de la emisión de una solicitud POST. A su vez

en esta ruta, al recibir la solicitud post, se llamará al controlador ApiRegistroController, para obtener los datos del proceso y enviárselos a la tabla de MYSQL empleando en modelo “Registro”. La figura 4.36, muestra de manera breve del proceso antes descrito.

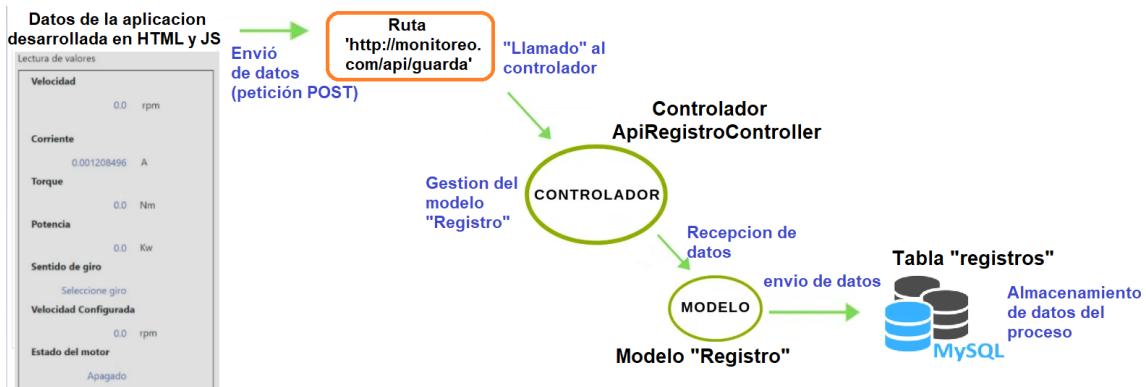


Figura 4.36 Proceso para el almacenamiento de datos del proceso usando la aplicación de Laravel.

Implementación de la API para el envío de valores a la aplicación de Laravel

Para el desarrollo de la aplicación desarrollada en HTML y JavaScript del capítulo 3, se implementó el uso de varios scripts o APIS. Una de las APIS usadas para el funcionamiento de la aplicación es la api “Script”, en la cual se configurará la función para el envío de datos a Laravel por medio de la petición post.

La figura 4.37, muestra desde la línea 124 a la línea 131, como se obtienen los datos de cada una de las variables del proceso, asignando su respectivo valor a las variables auxiliares (velocidad, corriente, Torque, Potencia, sentido, venviada, estadoMotor). Los datos son obtenidos de los cuadros de texto que muestran el valor de cada variable, en la sección “Lectura de valores”, de la aplicación diseñada en el capítulo 3.

La figura 4.38 muestra la relación de cada variable del proceso con su respectivo campo de la tabla de MYSQL.

Luego en la figura 4.37, en la línea de código 133, por medio del método POST, y usando el array “data”, se envía los valores de cada variable del proceso. Dentro del array, se asigna el valor obtenido de cada variable auxiliar a su respectivo campo de la tabla de MYSQL. Los 7 datos del array “data”, a través por el método POST, enviará cada segundo a la tabla registros, por lo que la base de datos de MYSQL, almacena información cada segundo.

```

js script.js    X
js > js script.js > ⚡ ready() callback > ⚡ setInterval() callback > ⚡ data
124     //valores para guardar
125         velocidad=$('#input[id=VelMotor]').val();
126         corriente=$('#input[id=corriente]').val();//document.getElementById('corriente').val;
127         Torque=$('#input[id=Torque]').val();//document.getElementById('Torque').val;
128         Potencia=$('#input[id=Potencia]').val();//document.getElementById('Potencia').val;
129         sentido=$('#input[id=sentido]').val();//document.getElementById('sentido').val;
130         venienda=$('#input[id=regulador1]').val();//document.getElementById('regulador1').val;
131         estadoMotor=$('#input[id=statusMotor1]').val();//document.getElementById('estadoMotor').val;
132
133         $.ajax({url:'http://monitoreo.com/api/guarda',method:'POST',data:[
134             "velocidad": velocidad,
135             "corriente": corriente,
136             "Torque": Torque,
137             "Potencia": Potencia,
138             "sentido": sentido,
139             "regulador1": venienda,
140             "estadoMotor": estadoMotor,
141         });
142     });
143     //fin salva valores
144 },1000);

```

Figura 4.37 Función implementada para el envío de valores a la aplicación diseñada en Laravel.



Figura 4.38 Relación de cada variable de proceso con su respectivo campo en la base de datos.

- Implementación del controlador “RegistroController”

Este controlador tiene como función principal mostrar al usuario cada uno de los datos que se recopilaron en la tabla de la base de datos de MYSQL. Al crear el controlador RegistroController, se implementan funciones por defecto, las cuales algunas de ellas serán modificadas para el funcionamiento de la aplicación. La figura 4.39 muestra el controlador implementado, y mediante la tabla 4.2, se muestra la utilidad de cada función.

```

11  /**
12  * Display a listing of the resource.
13  *
14  * @return \Illuminate\Http\Response
15  */
16
17  public function index()
18  {
19      $registros = Registro::latest()
20          ->take(20)->paginate();
21
22      //return response()->json($registros);
23      return view('registro.index', compact('registros'))
24          ->with('i', (request()->input('page', 1) - 1) * $registros->perPage());
25
26
27  /**
28  * Show the form for creating a new resource.
29  *
30  * @return \Illuminate\Http\Response
31  */
32  public function create()
33  {
34      $registro = new Registro();
35      return view('registro.create', compact('registro'));
36  }
37
38

```

Figura 4.39 Controlador RegistroController.

Tabla 4.2 Funciones del Controlador RegistroController.

Controlador RegistroController	
Función	Utilidad
public function index()	Visualiza una lista de recursos.
public function create()	Muestra el formulario para implementar un nuevo recurso.
public function store(Request \$request)	Almacena un recurso recién creado en una unidad de almacenamiento.
public function show(\$id)	Muestra un recurso específico
public function edit(\$id)	Visualiza el formulario para editar un recurso específico
public function update(Request \$request, Registro \$registro)	Actualiza un recurso específico en la unidad de almacenamiento.
public function destroy(\$id)	Elimina un recurso específico

Además de las funciones de la tabla 4., se implementará las funciones grafico () y datos (), las cuales se visualizan en la figura 4.40. La función grafico (), se empleará para recopilar los datos que el modelo Registro obtiene de la tabla “registro” de la base de datos de MYSQL. Luego, se “llamará” al elemento “Vista:index”, el cual se encargará de mostrar los datos mediante el orden en el que se ha creado cada dato guardado('created_at').

Mientras que la función datos (), permite transferir la información de la base de datos que se encuentra en lenguaje de MYSQL, a un lenguaje JSON. Usando el lenguaje JSON, se podrá visualizar el contenido de la tabla de la base datos en la aplicación WEB, que será presentado al usuario cuando solicite la información.

```

    app > Http > Controllers > RegistroController.php
    110 }
    111 }
    112 }
    113 }
    114 public function grafico()
    115 {
    116     $velocidad=Registro::select('velocidad')->orderBy('created_at','desc')->first();
    117     // return json_decode($velocidad);
    118     // return response()->json($velocidad,200);
    119     // return response()->json($velocidad,200);
    120     return view('graph.index',compact('velocidad'));
    121 }
    122 public function datos()
    123 {
    124     $velocidad=Registro::select('velocidad')->orderBy('created_at','desc')->first();
    125     // return json_decode($velocidad);
    126     // return response()->json($velocidad,200);
    127     // return response()->json($velocidad,200);
    128 }

```

Figura 4.40 Funciones grafico () y datos ().

Configuración de las rutas del controlador RegistroController.

Cuando el usuario requiere acceder a la sección “Registros” de la aplicación web, la cual muestra la información recopilada de la base de datos, se emitirá una petición “GET” que es gestionada en el archivo web.php.

Por lo tanto, se deberá editar el archivo “web.php” de la carpeta routes(rutas), con la finalidad de establecer una url, a la cual el usuario ingresa desde el navegador web para acceder a la sección “Registros”. Esta url, también sirve a la aplicación web de Laravel para llamar al elemento controlador RegistroController.

La figura 4.41, muestra el método “Route::resource” el cual establece la ruta url “monitoreo.com/registros” (línea 27). Cuando se ingrese a esta url desde el navegador web, se emitirá una petición GET, la cual llama al controlador RegistroController y a su vez este, ejecuta cada una sus funciones.

Por otro lado, usando la instrucción 'middleware', se permite acceder a los datos almacenados en MYSQL, cuando se acceda a la ruta “registros”.

```

    routes > web.php
    18 Route::get('/', function () {
    19     return view('welcome');
    20 });
    21
    22 Auth::routes();
    23
    24 Route::get('/home', [App\Http\Controllers\HomeController::class, 'index'])->name('home');
    25
    26
    27 Route::resource('registros', RegistroController::class);
    28
    29
    30 Route::group(['middleware' => ['cors']], function () {
    31     Route::get('datos',[RegistroController::class,'datos'])->name('datos');
    32     Route::get('grafico',[RegistroController::class,'grafico'])->name('grafico');
    33 });
    34

```

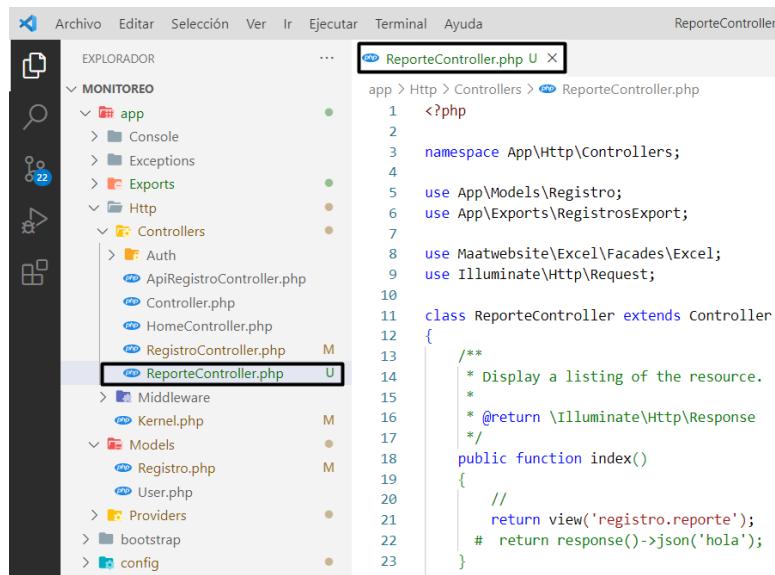
Figura 4.41 Rutas get del controlador RegistroController.

- Implementación del controlador “ReporteController”

Otra de las funciones de la aplicación web, es la descarga y/o visualización de datos, dentro de rango establecido por fecha y hora. Para implementar la función antes descrita, se ha implementado el controlador ReporteController, el cual poseerá los métodos para llamar al elemento “vista” que muestra los datos de la base de MYSQL y generar el archivo csv que contiene los datos determinados en el rango del usuario.

La figura 4.42, muestra el controlador que se ha implementado y a su vez la función index, la cual se implementa por defecto y ha sido modificada. La utilidad de la función index, será la de “retornar” el elemento vista “reporte”, permitiendo visualizar en pantalla, la interfaz de la sección “Reportes ” de la aplicación web. En esta vista, se presentarán los controles o formularios implementados bajo el lenguaje html para poder configurar el rango de datos que el usuario desea. Además de definir el tipo de reporte que el usuario desea.

Esta vista aparece como primera instancia cuando se acceda a través del navegador a la url “monitoreo.com/reportes”, pero para ello se deberá configurar la ruta correspondiente, siguiendo el mismo proceso para implementar las rutas del controlador registro. Esta implementación se la realizara con más detalle en la sección.



```

ReporteController.php U
app > Http > Controllers > ReporteController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Models\Registro;
6  use App\Exports\RegistrosExport;
7
8  use Maatwebsite\Excel\Facades\Excel;
9  use Illuminate\Http\Request;
10
11 class ReporteController extends Controller
12 {
13
14     /**
15      * Display a listing of the resource.
16      *
17      * @return \Illuminate\Http\Response
18      */
19     public function index()
20     {
21         //
22         //return view('registro.reporte');
23         # return response()->json('hola');
24     }
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
287
288
289
289
290
291
292
293
294
295
296
297
297
298
299
299
300
301
302
303
304
305
306
307
307
308
309
309
310
311
312
313
314
315
315
316
317
317
318
319
319
320
321
322
323
323
324
325
325
326
327
327
328
329
329
330
331
331
332
333
333
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404

```

```

    Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
    ReporteController.php - monitoreo - Visual Studio Code
    EXPLORADOR
    MONITOREO
    app
    > Console
    > Exceptions
    > Exports
    > Http
    > Controllers
    > Auth
    > ApiRegistroController.php
    > Controller.php
    > HomeController.php
    > RegistroController.php
    > ReporteController.php M
    > Middleware
    > Kernel.php M
    > Models
    > Registro.php M
    > User.php M
    > Providers
    > bootstrap
    ReporteController.php U X
    app > Http > Controllers > ReporteController.php
    29
    30
    31
    32
    33
    34
    35
    36
    37
    38
    39
    40
    41
    42
    43
    44
    45
    46
    47
    48
    49
    */
    public function create(Request $request)
    {
        //
        $fechaInicio=$request['fechaInicio'];
        $fechaFin=$request['fechaFin'];

        $registros=Registro::where('created_at','>',$fechaInicio)->where('created_at','<',$fechaFin)->get();
        // $registro=Registro::find(20);
        #2022-07-07T15:37:26.000002Z,"updated_at":"2022-07-07T15:37:26.000002Z
        if ($request['tipo']=='visual') {
            return response()->json($registros);
        // }
        elseif ($request['tipo']=='csv') {
            return Excel::download(new RegistroExport($fechaInicio,$fechaFin),'Registros.xlsx');
        }
    }

```

Figura 4.43 Función create del controlador ReporteController.

Configuración de las rutas del controlador ReporteController.

Cuando se ha definido las funciones del controlador ReporteController, es necesario configurar el archivo que recibe las peticiones get, una vez que el usuario requiere acceder a la sección “reportes” desde el navegador web. La finalidad de esta ruta, será la de indicar a la aplicación web que efectué el llamado al controlador ReporteController.

Se puede observar en la línea 35 (figura 4.44), mediante el uso del método “Route::get”, se gestiona que al recibir una petición get al acceder a la url “monitoreo.com/reporte”, se llame al controlador ReporteController. A su vez el controlador ejecuta la función “index” contenida en este.

```

    Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
    routes > web.php M X
    routes > web.php
    18 Route::get('/', function () {
    19     return view('welcome');
    20 });
    21
    22 Auth::routes();
    23
    24 Route::get('/home', [App\Http\Controllers\HomeController::class, 'index'])->name('home');
    25
    26
    27 Route::resource('registros', RegistroController::class);
    28
    29
    30 Route::group(['middleware' => ['cors']], function () {
    31     Route::get('datos',[RegistroController::class,'datos'])->name('datos');
    32     Route::get('grafico',[RegistroController::class,'grafico'])->name('grafico');
    33 });
    34
    35 Route::get('reporte',[ReporteController::class,'index'])->name('reporte.create');
    36
    37 Route::get('reporte/ver',[ReporteController::class,'create'])->name('reporte.ver');
    38
    39

```

Figura 4.44 Configuraciones de las rutas del controlador ReporteController.

4.5.5 Implementación de los elementos “Vista”

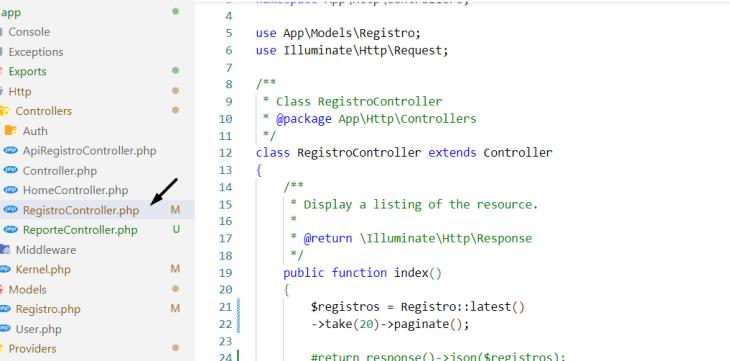
Como parte final de la configuración de la aplicación web, será la implementación de los elementos “Vista”. El elemento Vista, permite mostrar o presentar el contenido al usuario por medio de un archivo o documento HTML. Para la creación de un elemento vista, será necesario que este posea una extensión “.blade.php”, el cual será almacenado en el directorio por defecto “resources/views” de la aplicación [47].

Un proyecto implementado por Laravel, usa templates(plantillas) que son archivos que contienen partes de un código, el cual se repite en más de una vista. Dentro de este “template”, se albergarán archivos como el head de un documento HTML, elementos CSS del sistema y una sección para archivos implementados en JavaScript.

Como se mencionó anteriormente, en los dos controladores implementados; RegistroController y ReporteController, se llaman a elementos “Vista”. Por ello, se deberá implementar el elemento vista para cada controlador, ya que cada uno de estos posee una función específica.

- Implementación del elemento vista “index.blade.php”

La figura 4.45, muestra la función del controlador RegistroController, el cual retorna el elemento vista “index” usando la instrucción return. El elemento vista, “index”, permitirá mostrar cada uno de los elementos almacenados en la base de datos de MYSQL.



```
        RegistroController.php M X
app > Http > Controllers > RegistroController.php
4
5     use App\Models\Registro;
6     use Illuminate\Http\Request;
7
8 /**
9  * Class RegistroController
10 * @package App\Http\Controllers
11 */
12 class RegistroController extends Controller
{
13
14 /**
15  * Display a listing of the resource.
16  *
17  * @return \Illuminate\Http\Response
18  */
19 public function index()
20 {
21     $registros = Registro::latest()
22         ->take(20)
23         ->paginate();
24
25     #return response()->json($registros);
26     return view('registro.index', compact('registros'))
27         ->with('i', (request()->input('page', 1) - 1) * $registros->perPage());
28 }
```

Figura 4.45 Función de retorno del elemento vista “index” desde el controlador RegistroController.

Cuando el usuario requiera acceder a la ruta “monitoreo.com/registros” desde el navegador web, se mostrará el archivo html incrustado en el elemento vista “index.blade.php”. La figura 4.46, muestra la carpeta “registro” señalado por el recuadro negro que contiene cada una de los archivos templates que se usan para el controlador RegistroController y ReporteController.

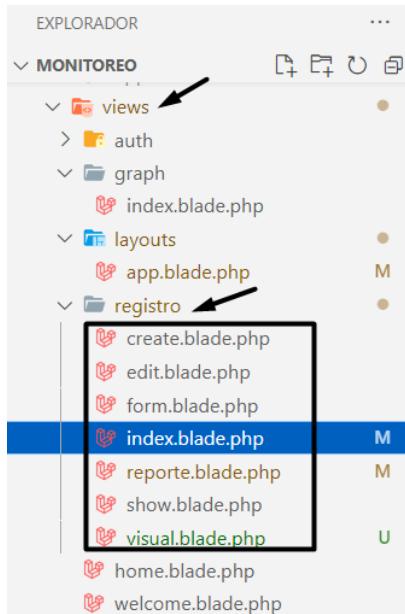
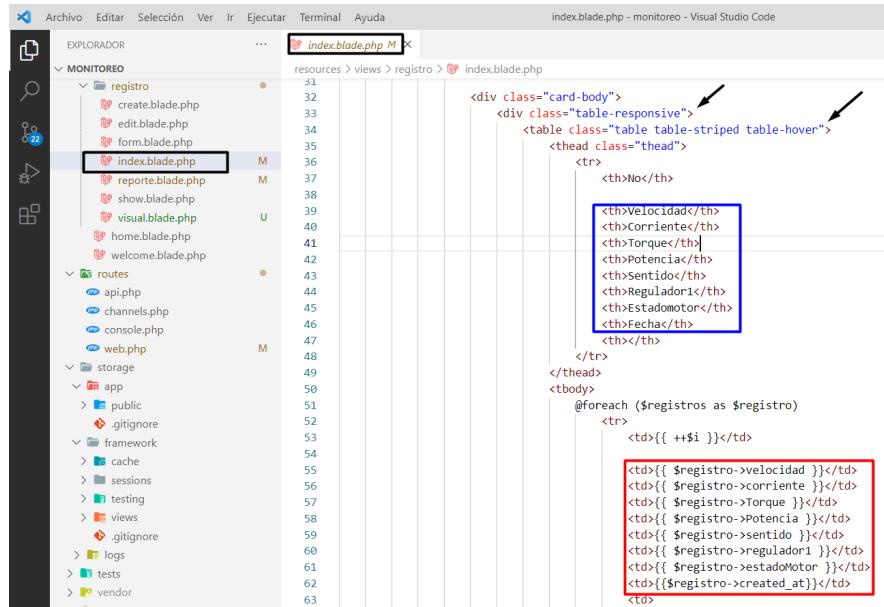


Figura 4.46 Carpeta “registro” en la sección “vistas”.

Luego de crear cada archivo blade.php, se implementará la sección en lenguaje html encargada de mostrar cada uno de los datos almacenados en la tabla de MYSQL. Cuando se acceda a la ruta registros, en el navegador, se ira mostrando cada dato almacenado mediante una tabla estructurada con las etiquetas de html “tr”, “td” y “th” y demás clases de la carpeta Bootstrap.

La figura 4.47, muestra la implementación de la tabla de datos a mostrar mediante las clases; class="table-responsive" y class="table table-striped table-hover". En el recuadro azul, se puede notar que por medio del tag de html "th", se definen las celdas de encabezado de la tabla. Mientras que cada dato de los campos de la tabla, será colocado en una celda definida por el tag "td".



```

<div class="card-body">
<div class="table-responsive">
<table class="table table-striped table-hover">
<thead class="thead">
<tr>
<th>No</th>
<th>Velocidad</th>
<th>Corriente</th>
<th>Torque</th>
<th>Potencia</th>
<th>Sentido</th>
<th>Regulador1</th>
<th>EstadoMotor</th>
<th>Fecha</th>
</tr>
</thead>
<tbody>
@foreach ($registros as $registro)
<tr>
<td>{{ $registro->velocidad }}</td>
<td>{{ $registro->corriente }}</td>
<td>{{ $registro->torque }}</td>
<td>{{ $registro->potencia }}</td>
<td>{{ $registro->sentido }}</td>
<td>{{ $registro->regulador1 }}</td>
<td>{{ $registro->estadomotor }}</td>
<td>{{ $registro->created_at }}</td>
</tr>
@endforeach
</tbody>
</table>
</div>
</div>

```

Figura 4.47 Elemento vista “index.blade.php”.

- Implementación de los elementos vista “reporte.blade.php” y “visual.blade.php”

Como parte final de las implementaciones de la aplicación web, es la configuración de los elementos vista previamente creados; “reporte.blade.php” y “visual.blade.php”.

Como se mencionó en la sección del controlador “ReporteController”, cuando se acceda a la sección “reportes”, se visualizará como primera instancia el elemento vista “reporte.blade.php”. Este elemento posee los controles para definir el rango de datos que el usuario desea establecer definiendo una fecha y hora de inicio y fin. Cada control o elemento que contiene el elemento vista, será implementado en lenguaje html igual que en la sección anterior, ya que estos elementos se visualizan en el navegador web cuando este interprete el documento html del elemento vista.

En la figura 4.47, se muestra los controles para definir un rango de datos en el archivo “reporte.blade.php”. Por medio del elemento de html, input required type="datetime-local" (recuadros negros), se solicita la fecha y hora de inicio y fin para establecer un rango de datos por parte del usuario. Mientras que por medio del elemento select required class="form-select" (recuadro azul), se seleccionara el tipo de reporte a obtener. Por último, usando el elemento button (señalado por la flecha azul), se confirmará y solicitará el reporte escogido.

```

resources > views > registro > reporte.blade.php
8  <div class="container">
9
10 <form action="{{route('reporte.ver')}}" method="get" enctype="multipart/form-data">
11 <div class="mb-3">
12   <label for="fechaInicio" class="form-label">Fecha de inicio</label>
13   <input required type="datetime-local" class="form-control" id="fechaInicio" name="fechaInicio" aria-describedby="fechaInicio">
14   <div id="fechaInicio" class="form-text">Ingresa la fecha desde la cual obtendras el reporte.</div>
15 </div>
16
17 <div class="mb-3">
18   <label for="fechaFin" class="form-label">Fecha de fin</label>
19   <input required type="datetime-local" class="form-control" id="fechaFin" name="fechaFin" aria-describedby="fechaFin">
20   <div id="fechaFin" class="form-text">Ingresa la fecha hasta la cual obtendras el reporte.</div>
21 </div>
22 <div class="mb-3">
23   <select required class="form-select" name="tipo" aria-label="Default select example">
24     <option value="" selected>Selecciona el tipo de reporte</option>
25     <option value="visual">visual</option>
26     <option value="csv">Excel .csv</option>
27   </select>
28 </div>
29
30 <button type="submit" class="btn btn-primary">Submit</button>
31
32 </form>
33
34 </div>
35
36
37
38
39 </div>

```

Figura 4.48 Elemento vista “reporte.blade.php”.

Por otro lado, cuando el usuario requiera un reporte visual, el controlador ReporteController llama al elemento vista “visual.blade.php”, ya que, por medio de este elemento, se muestran el conjunto de datos en pantalla. Para la configuración de este elemento, se seguirá el mismo esquema seguido para la implementación de la vista “index.blade.php”, ya que los datos solicitados serán presentados por medio de una tabla.

La figura 4.49, muestra la configuración de la vista “visual.blade.php”, donde el recuadro negro indica el encabezado de cada columna o campo de cada dato almacenado. Luego en el recuadro azul, se muestra como cada celda obtiene cada dato recopilado por medio de la instrucción \$registro. Finalmente, esta sección de html será mostrada en pantalla cuando por medio de los controles del elemento vista ““reporte.blade.php””, se seleccione un reporte de tipo “visual”.

```

resources > views > registro > visual.blade.php
31 <div class="card-body">
32   <div class="table-responsive">
33     <table class="table table-striped table-hover">
34       <thead class="thead">
35         <tr>
36           <th>No</th>
37           <th>Velocidad</th>
38           <th>Corriente</th>
39           <th>Torque</th>
40           <th>Potencia</th>
41           <th>Sentido</th>
42           <th>Regulador1</th>
43           <th>Estadomotor</th>
44           <th>Fecha</th>
45       </tr>
46     </thead>
47     <tbody>
48       @foreach ($registros as $registro)
49         <tr>
50           <td>{{ $loop->index }}</td>
51           <td>{{ $registro->velocidad }}</td>
52           <td>{{ $registro->corriente }}</td>
53           <td>{{ $registro->Torque }}</td>
54           <td>{{ $registro->Potencia }}</td>
55           <td>{{ $registro->sentido }}</td>
56           <td>{{ $registro->regulador1 }}</td>
57           <td>{{ $registro->estadomotor }}</td>
58           <td>{{ $registro->created_at }}</td>
59         </tr>
60       @endforeach
61     </tbody>
62   </table>
63 </div>

```

Figura 4.49 Elemento vista “visual.blade.php”.

CAPÍTULO 5

Pruebas y resultados

El propósito de este capítulo, es mostrar cada una de las funciones de la aplicación WEB creada usando los lenguajes y herramientas de HTML y JavaScript. Cada sección posee funciones implementadas que serán corroboradas y dichos resultados, analizados para verificar el funcionamiento correcto del proceso controlado.

Como primer paso, se deberá configurar la interfaz de red del dispositivo cliente, el cual es la laptop con la que ha sido desarrollado el presente proyecto. Cuando se configure esta la interfaz de red del equipo, se podrá acceder al PLC Siemens para cargar el programa implementado en TIA PORTAL (Versión 16) y, por consiguiente, acceder al servidor WEB.

5.1 Conexión del motor asíncrono

Para las pruebas de la aplicación WEB que controlará el encendido y apagado de un motor asíncrono, se hará uso de los módulos del laboratorio de REDES INDUSTRIALES de la Universidad Politécnica Salesiana, sede CUENCA. La figura 5.1, muestra el banco de trabajo con los distintos módulos para el desarrollo de prácticas en temas de automatización. Para las pruebas del control de motor, se hará uso del PLC SIEMENS 1516-3 PN/DP, el convertidor SINAMICS, módulo de alimentación trifásica, y el motor asíncrono SIEMENS.



Figura 5.1 Banco de trabajo para prácticas de automatización.

Para controlar el motor asíncrono desde el variador de frecuencia, se conectará el motor en una conexión triángulo, ya que se trabajará en un voltaje de 220v. La figura 5.2, muestra el motor conectado al variador de frecuencia del banco de trabajo. Además, se muestra la conexión PROFINET del PLC SIEMENS con el variador, y de la misma manera, la conexión del equipo “cliente” con el PLC.

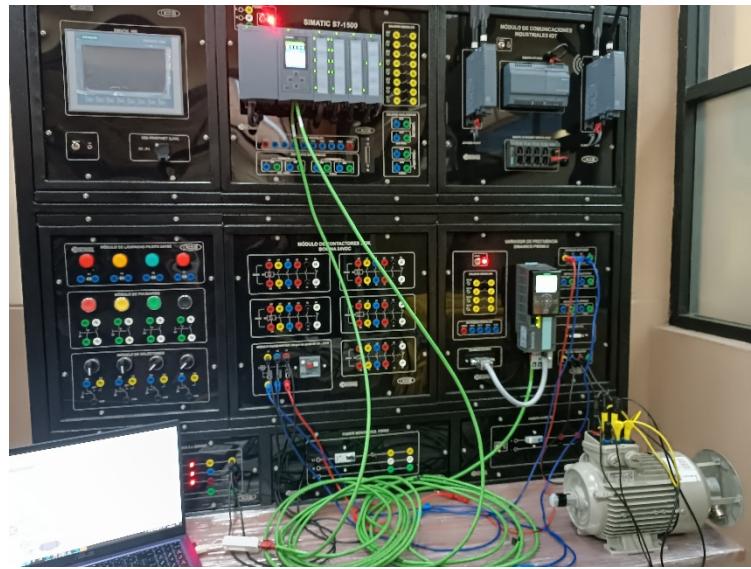


Figura 5.2 Conexiones del proceso de control.

5.2 Configuración de la interfaz de red del dispositivo cliente

Dado que el dispositivo escogido para las pruebas de la aplicación web es una laptop que se conecta por medio de un adaptador de red al PLC, es necesario configurar la interfaz para estar en conexión con la misma red del PLC. Como se observa en la figura 5.3, la dirección IP con la que se accede al servidor WEB es la dirección 192.168.0.4 que se encuentra en la red 192.168.0.0/24.

Tomando en consideración la red del PLC, se deberá escoger una dirección IP que se encuentre en la misma red del servidor WEB con la que la interfaz de la laptop se conectará a dicha red. Finalmente, la IP escogida es la 192.168.0.5/24, la cual será configurada en la interfaz de la laptop.

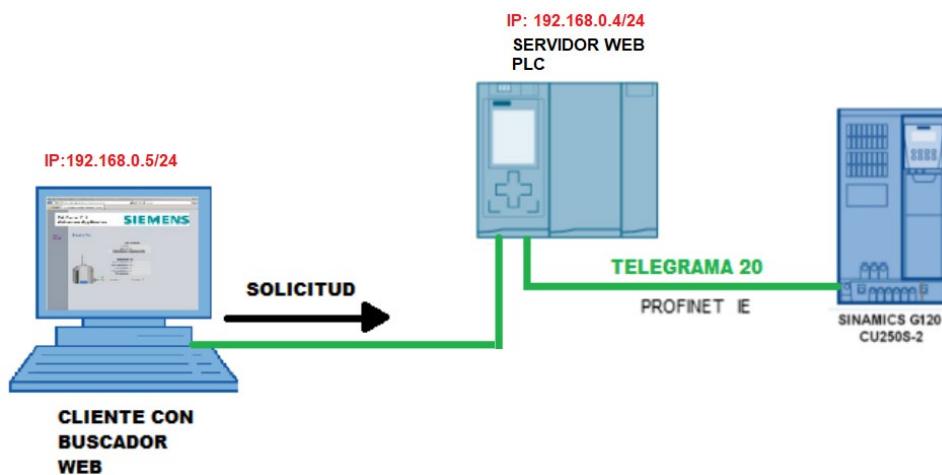


Figura 5.3 Dirección IP del dispositivo cliente para el acceso al servidor WEB.

Configuración de la dirección IP de la interfaz

Para asignar la dirección IP de la interfaz de la laptop, se ingresa desde el buscador de Windows a la sección “Mostrar redes disponibles”. Luego aparecerá la ventana que se muestra en la figura 5.4, donde seleccionaremos la opción “Centro de redes y recursos compartidos”.

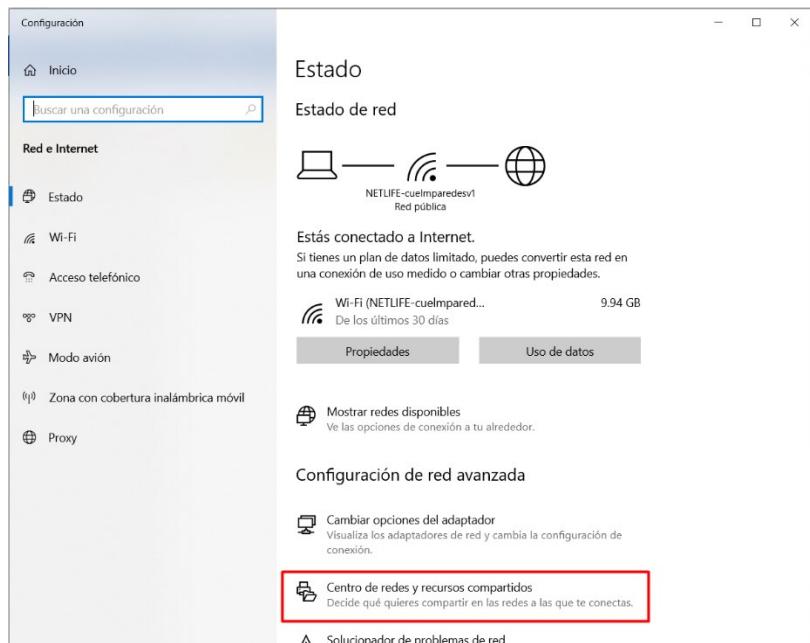


Figura 5.4 Sección “Mostrar redes disponibles” en Windows 10.

Al seleccionar la opción “Centro de redes y recursos compartidos”, se desplegará la ventana de la figura 5.5, donde se selecciona la interfaz “Ethernet”, la cual es la interfaz que se conecta a la interfaz X1 del PLC Siemens para acceder al servidor WEB.

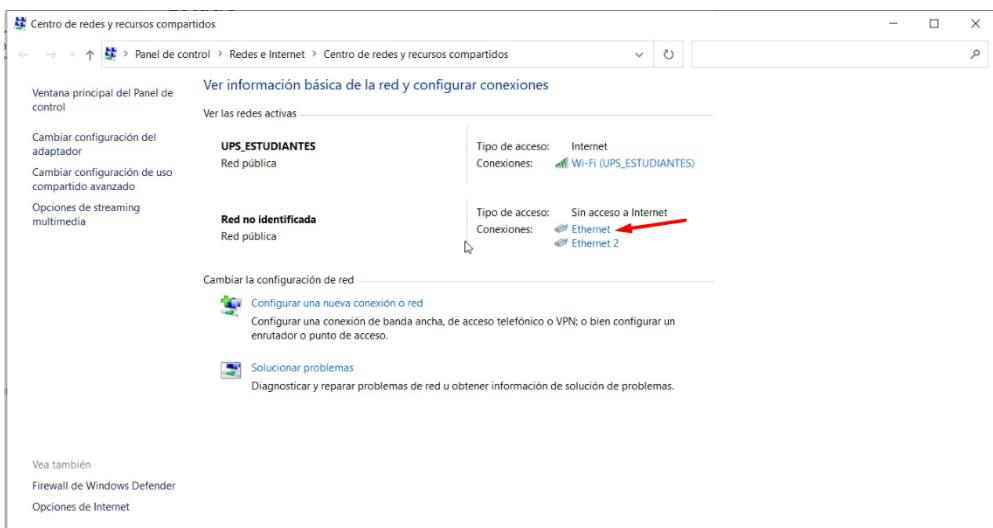


Figura 5.5 Sección “Centro de redes y recursos compartidos”.

Cuando se selecciona la interfaz “Ethernet”, se desplegará la ventana de la figura 5.6, donde primero, se asegurará que la interfaz escogida sea la correcta. En este caso, la interfaz tiene como nombre es “TP-LINK Gigabit Ethernet USB Adapter”. Esta interfaz necesita una dirección IP para conectarse a la interfaz del PLC Siemens, una vez que se configure el parámetro “Protocolo de Internet versión 4 (TCP/IPv4)”.

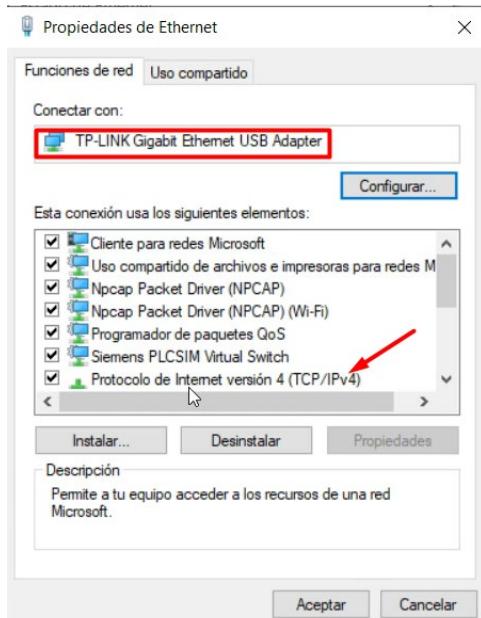


Figura 5.6 Propiedades de Ethernet de la interfaz del dispositivo “Cliente”.

En la sección de propiedades “Protocolo de Internet versión 4 (TCP/IPv4)”, se configurará los parámetros dirección IP y máscara de subred, como lo indica la figura 5.7. Luego de configurar los parámetros antes mencionados, se procederá a guardar las configuraciones una vez que se presione el botón “Aceptar”, dando por finalizado las configuraciones necesarias.

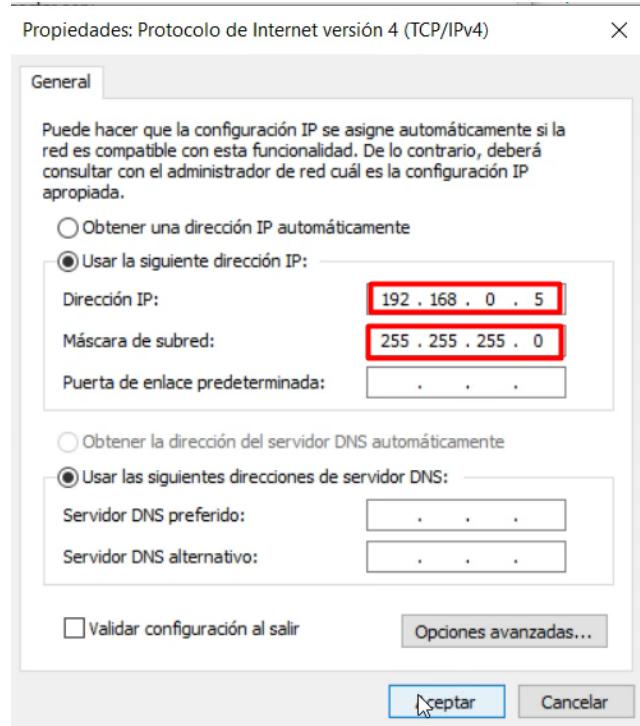


Figura 5.7 Configuración de la dirección IP de la interfaz de red del equipo cliente.

5.3 Configuraciones finales en TIA PORTAL

Dado que el servidor WEB configurado en TIA PORTAL, requiere del documento HTML que fue creado para el control del proceso mediante el servidor WEB, es necesario especificar la “página de usuario” en las configuraciones del servidor. La figura 5.8, muestra la sección “General”, cuando se ha situado en la sección de “Propiedades” del PLC escogido (“PLC_PLANTA”).

En la pestaña de “Páginas de usuario”, se muestra cuatro campos que han sido enumerados, los cuales deberán ser configurados. En “1”, se escoge la carpeta en la que se ubica el documento html creado por el usuario. En “2”, se escogerá el archivo html que contiene la aplicación web creada. En “3”, se asigna un nombre a la aplicación WEB. En 4, se mostrará el estado de los bloques que controlan el servidor web. Luego, se procede a presionar la opción “Generar bloques”. Al presionar la opción anterior, en el parámetro “4”, aparecerá “DB generado”, indicando que los bloques de control han sido generados.

Una vez que se complete el paso anterior, el programa estará listo para ser cargado en el PLC.

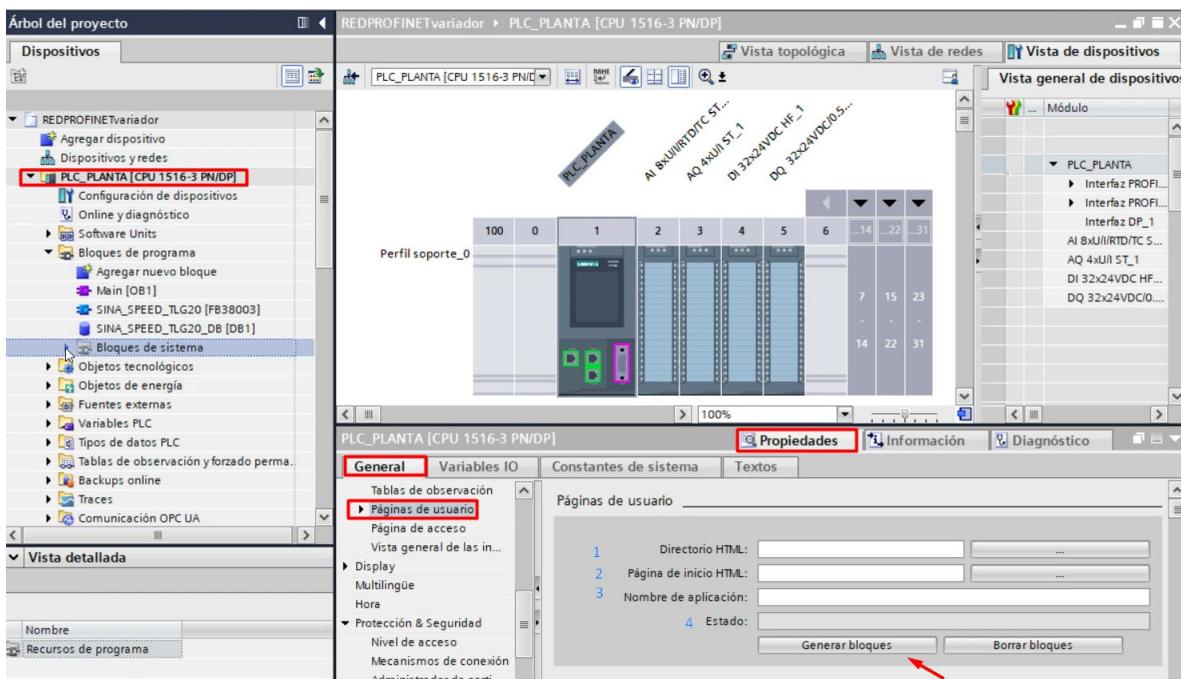


Figura 5.8 Configuración de la página de usuario del servidor WEB del PLC utilizado.

La figura 5.9, muestra el proceso antes descrito, mediante el recuadro rojo que indica que el programa ha generado los bloques para controlar el proceso mediante el servidor WEB.

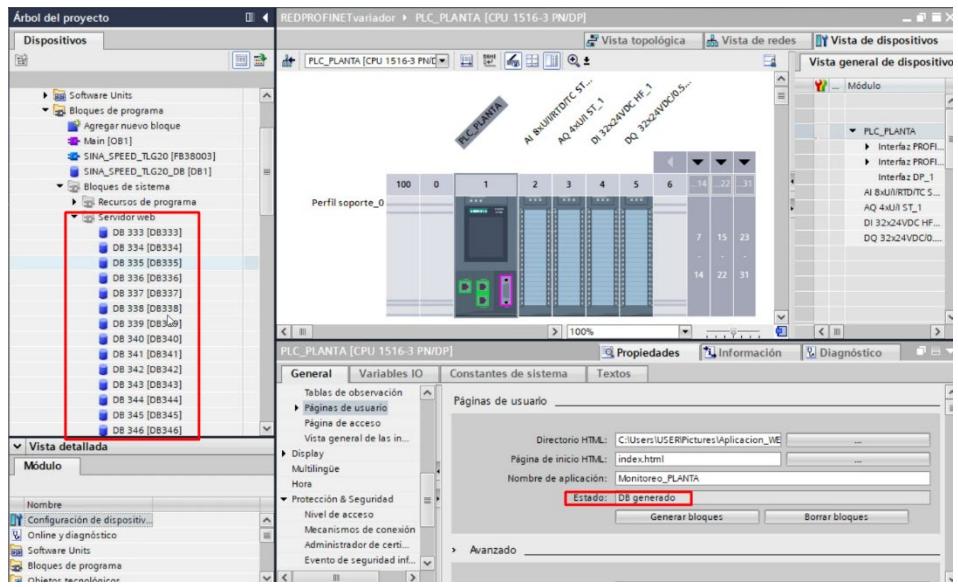


Figura 5.9 Bloques generados en TIA PORTAL para la comunicación del PLC mediante el servidor WEB.

5.4 Carga del programa en el PLC

Como paso final, se cargará el programa implementado en TIA PORTAL V16 en el PLC utilizado. La figura 5.10, muestra los pasos enumerados a seguir. En “1”, se selecciona el PLC en el que se cargara el programa luego de presionar en el nombre del dispositivo. En “2”, al presionar la opción “Online”, aparecerá un menú desplegable para seleccionar la opción del tipo de carga en el dispositivo. Como paso final, en “3”, se presionará la opción “Cargar y resetear programa PLC en el dispositivo”, para que aparezca el cuadro de dialogo mostrado en la figura 5.11.

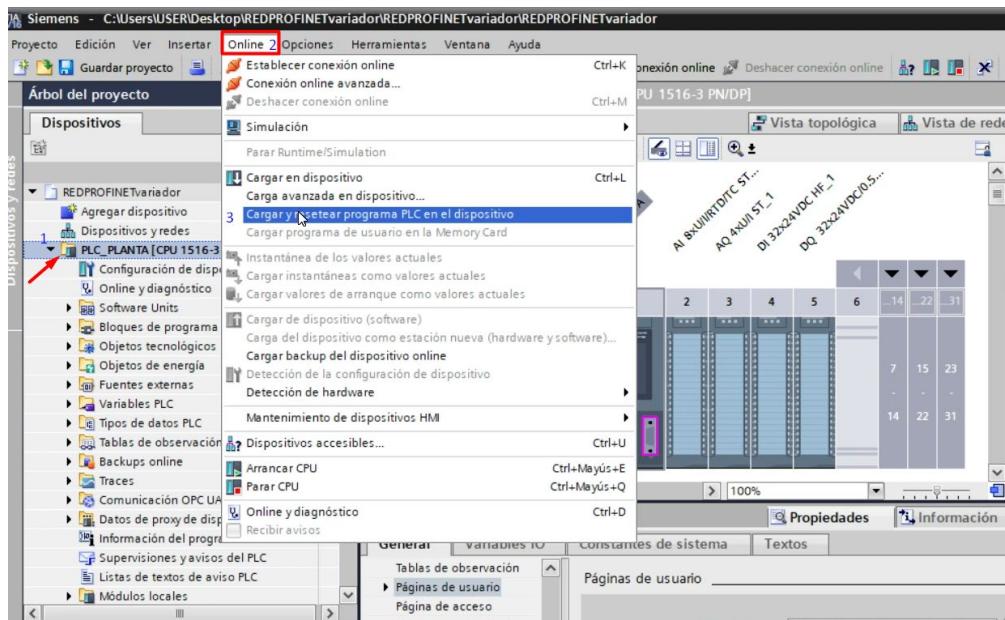


Figura 5.10 Carga del programa en el PLC utilizado.

Luego que aparezca el cuadro “Vista preliminar Carga”, se escogerá la opción “Parar todos”, ya que se necesita que el dispositivo o el módulo (PLC), se encuentre en estado STOP para cargar el programa.

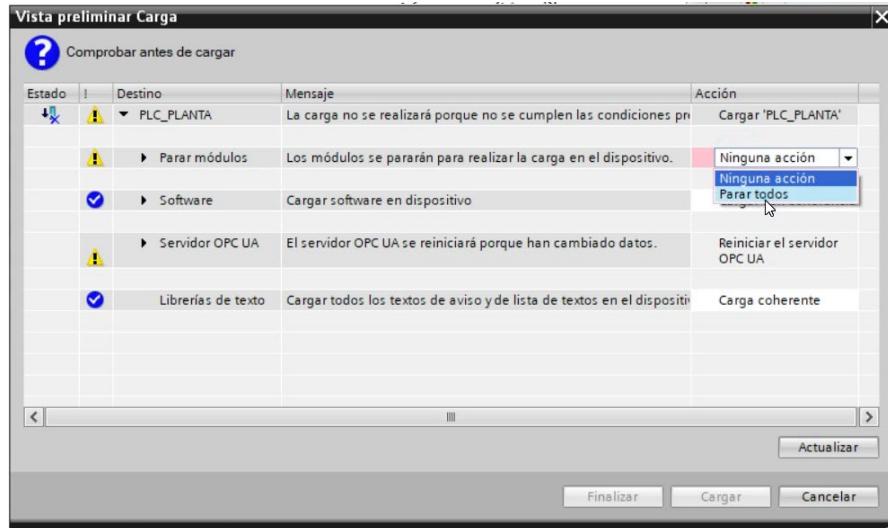


Figura 5.11 Carga del programa en el PLC utilizado.

Cuando se haya seleccionado la opción “Parar todos”, se activará la opción “Cargar”, y se presionara esta opción. Luego aparecerá la ventana que muestra la figura 5.12, indicando que se va a “Cargar en dispositivo”, el programa determinado.

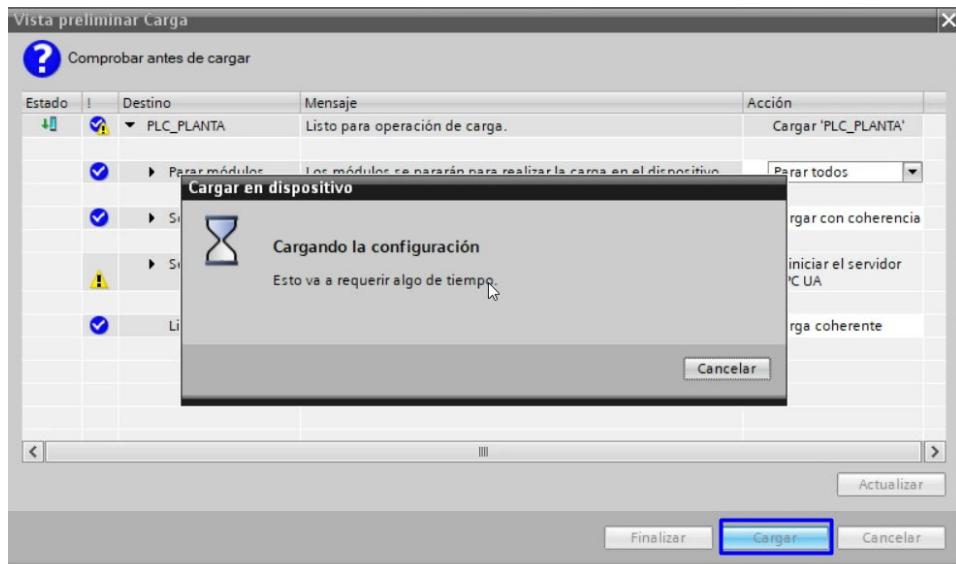


Figura 5.12 Ventana “Cargar en dispositivo”.

Como paso final, se mostrará la ventana “Resultados de la operación de carga” como lo muestra la figura 5.13. Se escogerá la opción “Arrancar modulo”, para colocar el PLC en modo RUN y el proceso de carga finalizará al presionar la opción “Finalizar”.

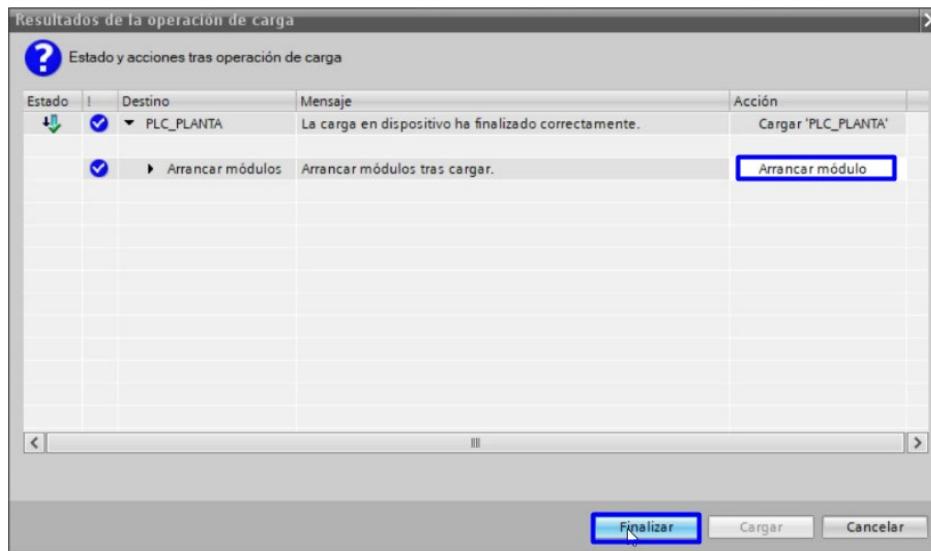


Figura 5.13 Etapa final del proceso de carga.

5.5 Pruebas de la aplicación WEB

Como primer paso, se requiere que desde la interfaz del programa XAMPP, se enciendan los módulos APACHE y MYSQL, este último, encargado de gestionar las tablas de la base de datos del proceso controlado. Para ello en el equipo, se procederá a abrir el programa XAMPP. La figura 5.14, muestra mediante los cuadros en rojo, los módulos que deberán encenderse para poner en servicio y acceder a la base de datos.

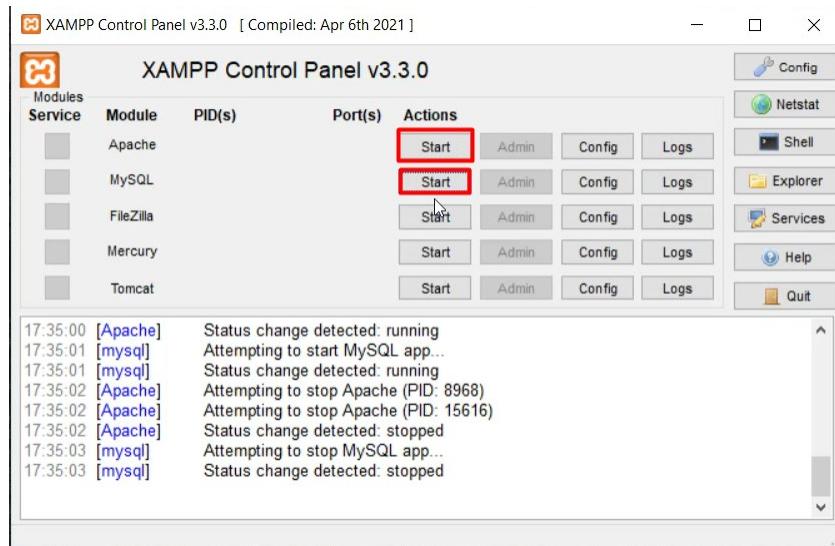


Figura 5.14 Panel de control XAMPP.

Para acceder al servidor web del PLC Siemens, es necesario el uso de un buscador WEB. El navegador web escogido es el programa “Google Chrome”, dado que es compatible con la sintaxis de la aplicación WEB. Al abrir el buscador, se deberá ingresar la IP de la interfaz configurada para acceder al servidor web. En este caso, el dispositivo cliente, se conecta a la interfaz X1 del PLC, es por ello que se deberá ingresar la IP configurada en el PLC. En el navegador web, se ingresará la ip “192.168.0.4”, como lo muestra la figura 5.15 y al presionar la tecla “Enter”, se accederá al servidor.

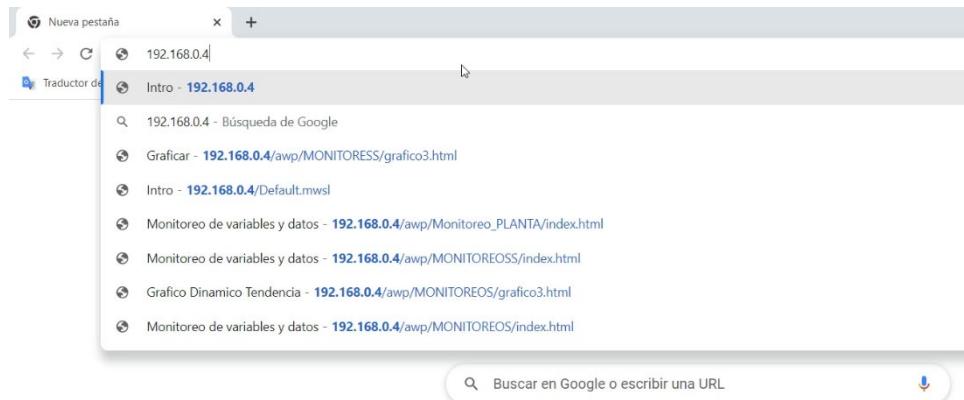


Figura 5.15 Acceso al servidor web desde el buscador Google Chrome.

La figura 5.16, muestra la página de portada del servidor WEB, una vez que se ha accedido a la dirección ip: 192.168.0.4.



Figura 5.16 Portada de inicio del servidor web.

Al presionar la opción “Entrar”, se podrá acceder a las páginas “estándar” que posee el servidor WEB del PLC, las cuales se describieron en el capítulo 1 de esta monografía. Por medio de la figura 5.17, se muestra la “Pagina inicial”, que permite visualizar el estado del PLC encargado de controlar el proceso. El “PLC_PLANTA”, muestra un estado “RUN” e información acerca del proyecto que se ha cargado en el PLC, el cual es “REDPROFINETvariador”.

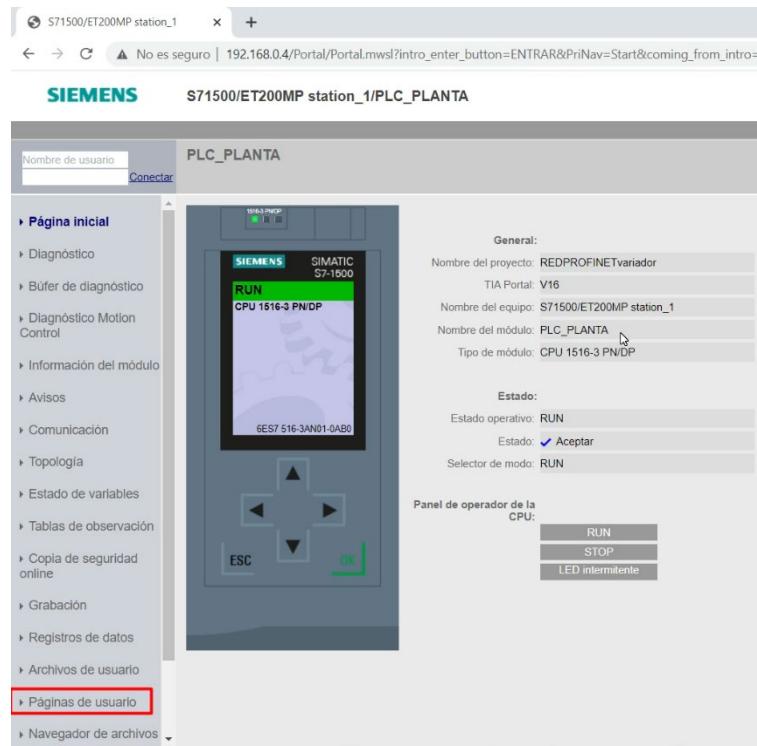


Figura 5.17 Página inicial del servidor WEB.

Luego, si se selecciona la sección o página “Páginas de usuario”, se podrá acceder a la aplicación WEB que se ha implementado. En la sección “Páginas de usuario”, aparecerá la opción que posibilita dirigir a la aplicación WEB cargada en el PLC como lo muestra la figura 5.18. Si se presiona la opción “Página de inicio de la aplicación Monitoreo_Planta” , se desplegará una nueva pestaña abriendo finalmente la aplicación diseñada.

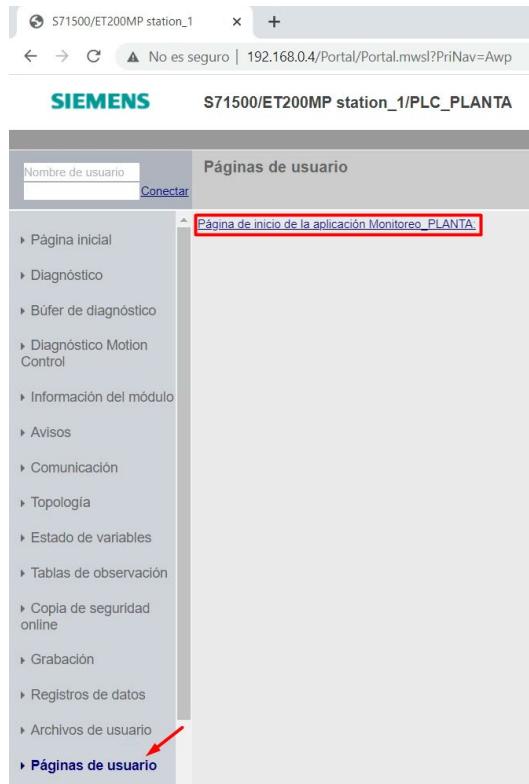


Figura 5.18 Página de usuario implementada en el servidor WEB.

La figura 5.19, muestra la aplicación WEB implementada, la cual se abrió en una pestaña nueva y aparece con el nombre “Monitoreo de variables y datos”. Como primera característica a resaltar, es la sección de lectura de valores, donde al abrir la aplicación, se inicia la lectura inmediata de los valores de velocidad actual, corriente actual, torque actual y potencia activa, valores entregados por el variador de frecuencia.

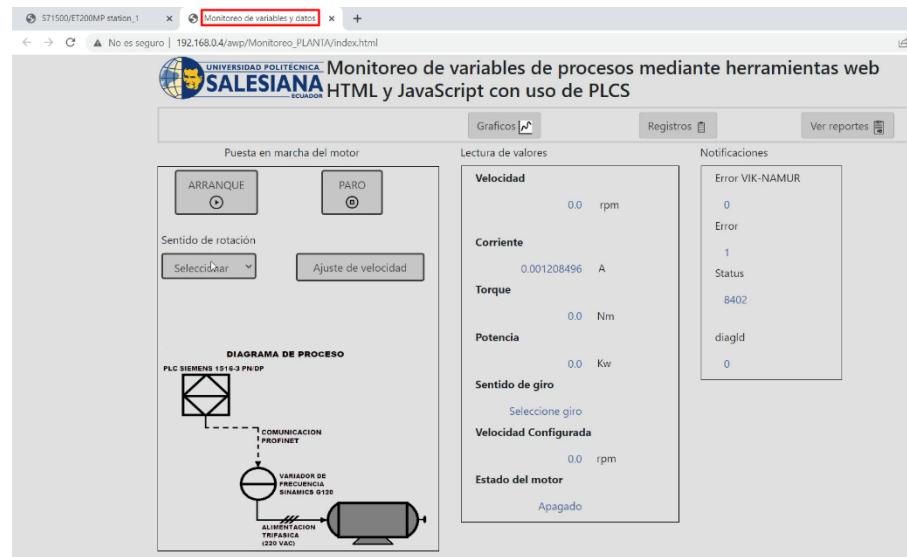


Figura 5.19 Aplicación WEB implementada en el servidor web del PLC SIEMENS.

5.5.1 Pruebas de la sección “Puesta en marcha del motor”

Al iniciar la aplicación WEB, la sección de Notificaciones por medio de la variable “Error” y “Status” notifican dos errores, como lo muestra la figura 5.20 mediante los recuadros rojos. El error definido de “1” por la variable “Error” indica la existencia de errores en el bloque de control del variador SINAMICS. Mientras que el error asignado por Status de “8402”, indica un bloqueo de conexión en el variador de frecuencia.

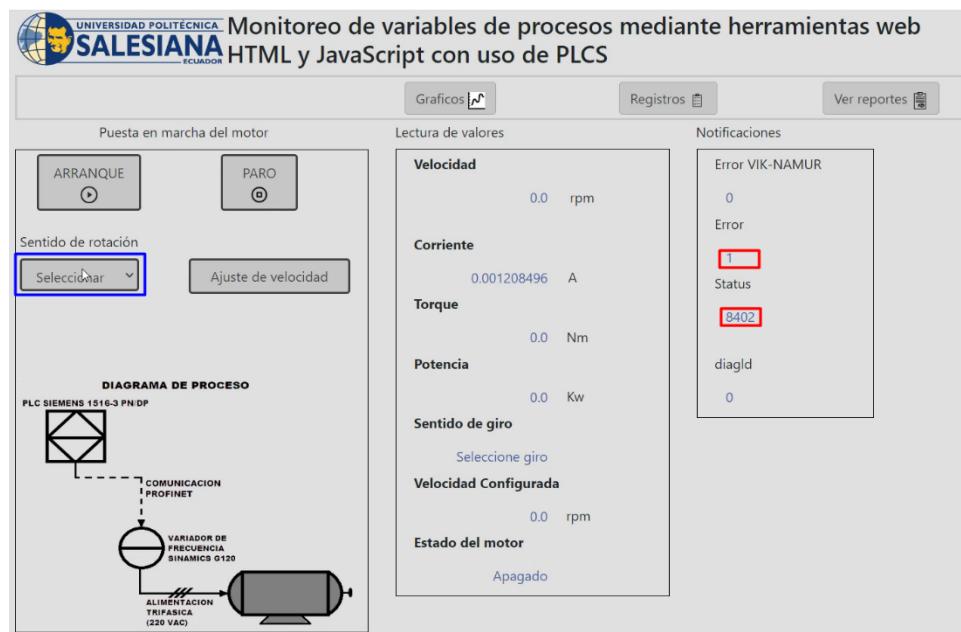


Figura 5.20 Estado inicial de la aplicación WEB.

Los errores antes mencionados, se producen debido a que la variable “configAxis”, controlada por el elemento “Multiselect,” al iniciar la aplicación, el elemento se mantiene en la opción “Seleccionar”. La variable “configAxis” del bloque función es responsable de controlar el sentido de giro del motor, y dado que el elemento Multiselect está en la opción Seleccionar, el elemento envía el valor en hexadecimal de “**16#047E**”. El valor antes mencionado, produce los dos errores en “Error” y “Status”, ya que dicho valor desconecta el motor del variador de frecuencia y deja al motor en estado de “Parada natural”. A continuación, se indica el proceso para poner en marcha el motor asíncrono conectado al variador de frecuencia SINAMICS.

Encendido del motor

Para encender el motor, se procederá con los siguientes pasos que se enumeran en la sección “Puesta en marcha del motor” por medio de la figura 5.21:

1. Se seleccionará el sentido de giro por medio del elemento Multiselect escogiendo un sentido Horario o Antihorario.
2. Luego, se definirá la velocidad en RPM mediante la interfaz que aparece cuando se pulsa el botón “Ajuste de velocidad”. Se tendrá en consideración que la velocidad máxima permitida por el motor es de 1440 RPM y su velocidad mínima de 0.
3. Finalmente, por medio del botón de control “ARRANQUE, se pondrá en marcha el motor”.

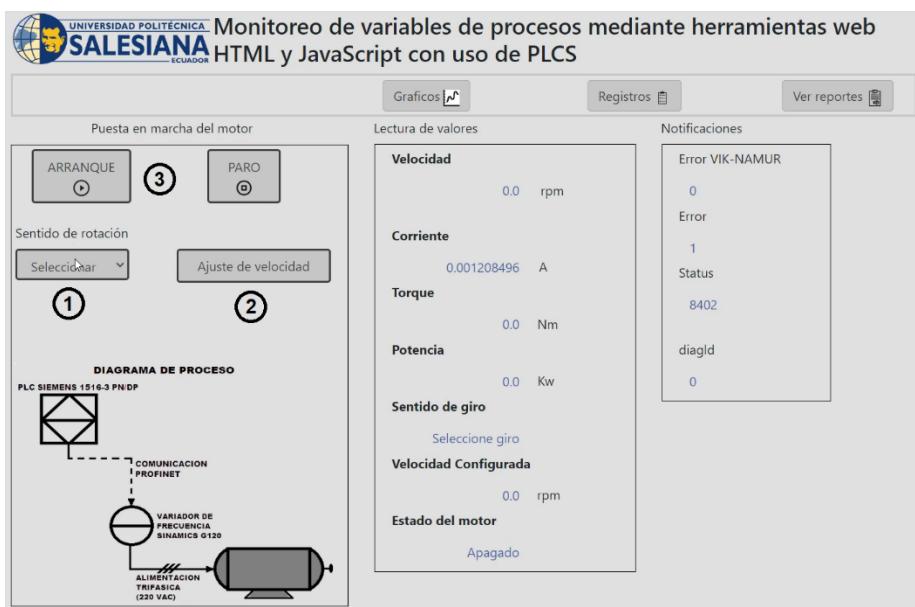


Figura 5.21 Pasos para la muestra en marcha del motor desde la aplicación WEB.

A continuación, se seguirán los pasos enumerados anteriormente para poner en marcha al motor: Como primer paso como se indica la figura 5.22, se selecciona el sentido de giro del motor mediante el control Multiselect. En este caso, se ha escogido un sentido de giro “horario”.

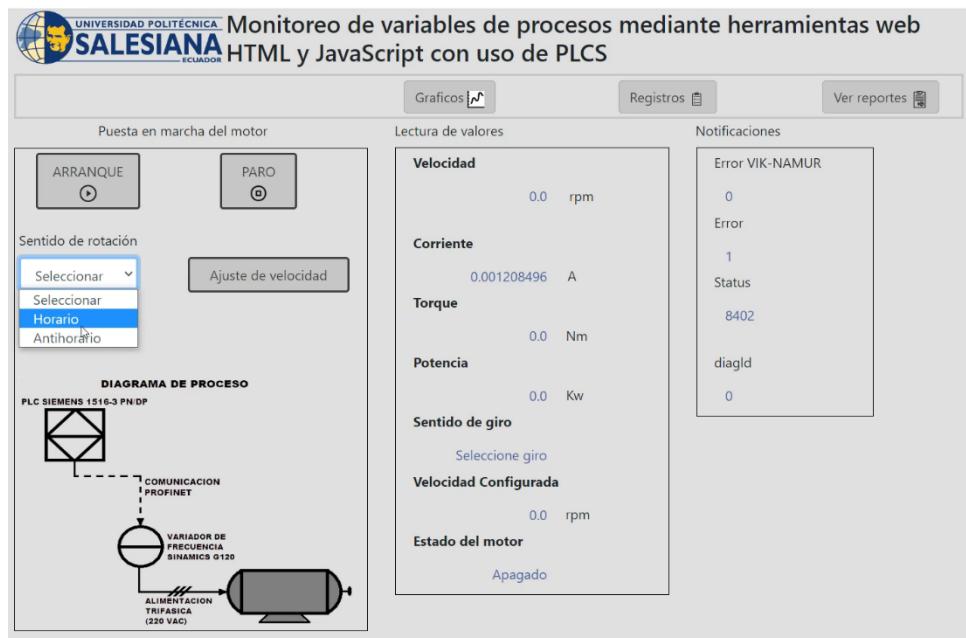


Figura 5.22 Selección del sentido del motor.

Luego de escoger el sentido de giro del motor, se puede notar un cambio automático de las variables “Error” y “Status” como se muestra en la figura 5.23. El valor en la variable “Error” tiene un valor de “0”, indicando que el bloque de control del variador no presenta errores. Mientras que la variable “Status”, por medio del valor “7002”, indica que no existen fallas activas en el bloque de control.

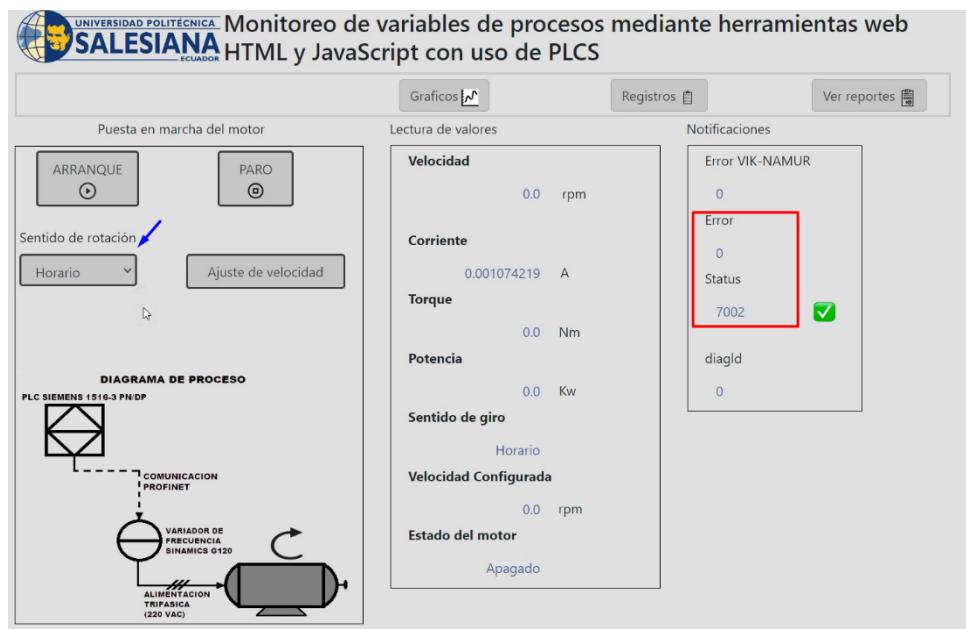


Figura 5.23 Proceso de control sin la presencia de errores.

Además, en la figura 5.24, se puede observar en la imagen “DIAGRAMA DE PROCESO”, que el motor tiene un color gris oscuro indicando su estado de apagado. Por otro lado, mediante el uso de imágenes y JavaScript, se puede observar una flecha que indica al usuario el sentido de giro, la cual está en sentido horario. Con lo antes descrito, se puede evidenciar el uso de imágenes interactivas para el usuario, para tener una vista rápida de lo que está sucediendo en el proceso.

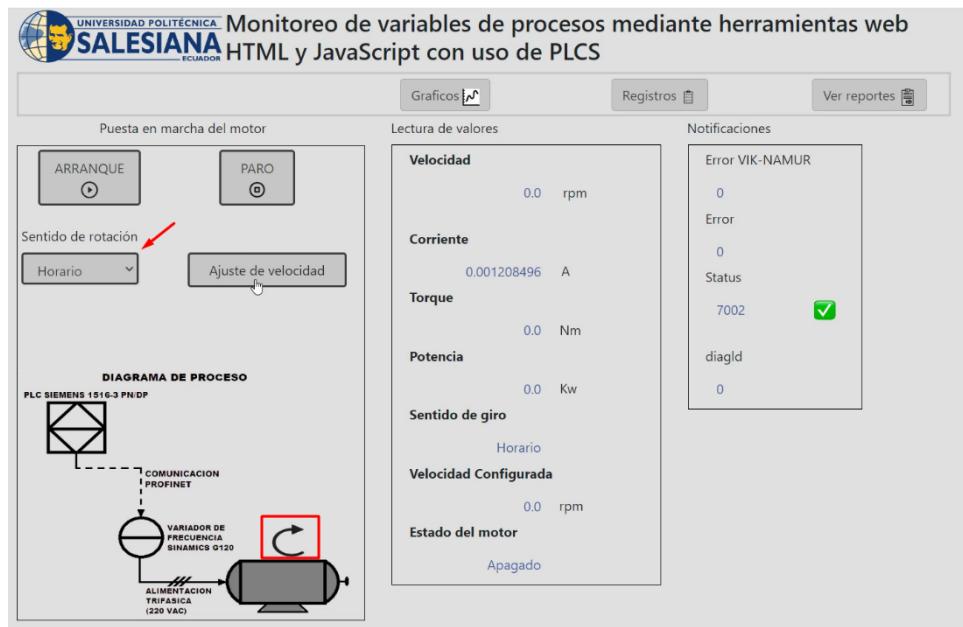


Figura 5.24 Uso de imágenes interactivas en la aplicación web.

Continuando con el segundo paso, se determinará el número de RPM con las que el motor funcionará cuando se presione el botón de control "Ajuste de velocidad". Cuando se ha pulsado el botón anterior, se desplegará el cuadro de dialogo mostrado en la figura 5.25.

En dicha figura, se observa dos formas de configurar la velocidad del motor en RPM, una mediante un recuadro señalado por el cuadro negro donde se ingresa un valor por teclado. Mientras que la segunda forma será mediante el envío de valores usando una barra deslizante que se encuentra en un rango entre 0 y 1440, la cual esta señalada por el recuadro azul.

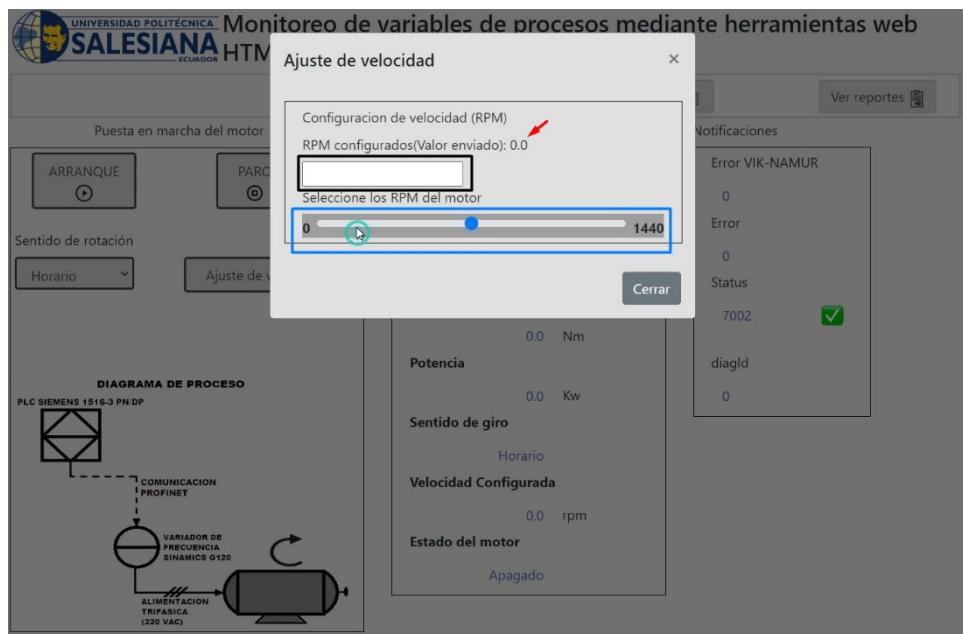


Figura 5.25 Controles de ajuste de velocidad del motor.

Como una primera prueba, se ingresará un valor desde el cuadro de texto como se observa en la figura 5.26. Se ha ingresado el valor de 120 RPM y al presionar la tecla "Enter", el valor ha sido enviado al variador de frecuencia. Además, como un método de retroalimentación, en la ventana emergente, se observa el valor de 120 RPM (señalado por la línea roja), antecedido por la etiqueta "RPM configurados". Esto ayudara como un método para el usuario conozca el valor que se ha enviado al bloque control por medio del servidor

WEB.

Completado el paso anterior, se cerrará el cuadro de dialogo por medio del botón “Cerrar”.

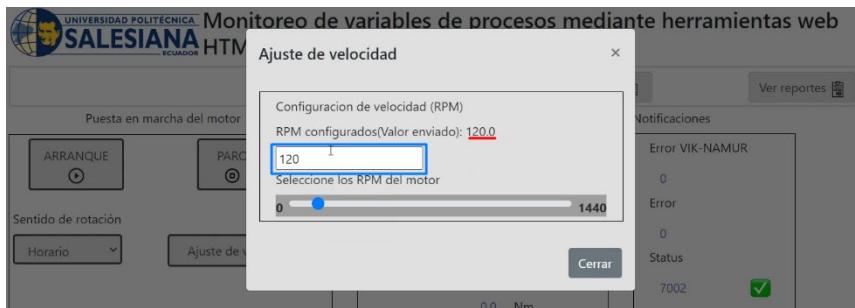


Figura 5.26 Ajuste de velocidad mediante un valor ingresado.

En caso, de que se ingresen valores o caracteres no permitidos como; letras (figura 5.27) y valores no permitidos (figura 5.28), la página desplegará una ventana emergente que indicara un mensaje de alerta de valores o caracteres ingresados incorrectamente.

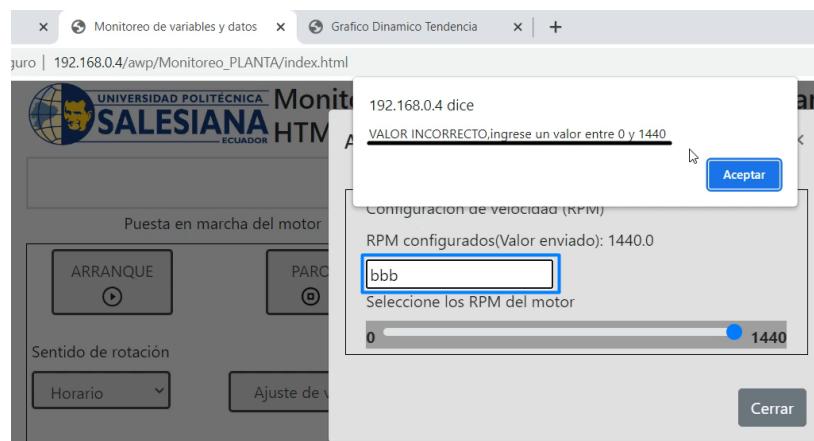


Figura 5.27 Caracteres no permitidos en la configuración de ajuste de velocidad.

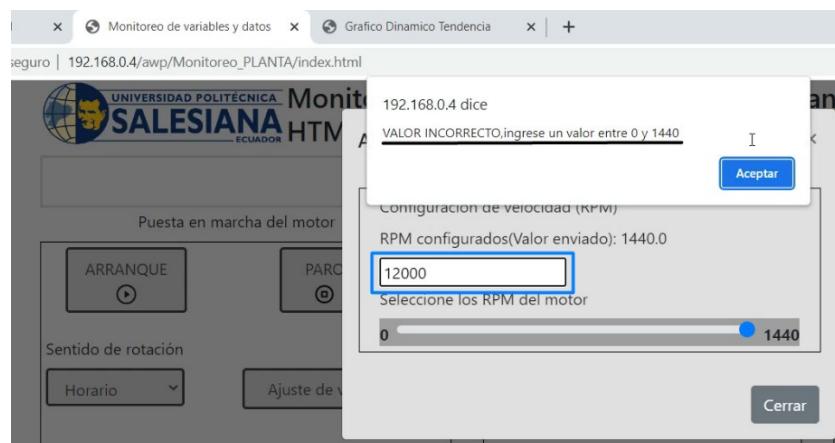


Figura 5.28 Valores no permitidos en la configuración de ajuste de velocidad.

Continuando con el proceso de arranque, se tiene el paso final, en el cual se presionará el botón “ARRANQUE” para encender el motor. La figura 5.29, muestra mediante el recuadro rojo, el estado de las variables de entrada que han sido modificadas, es decir, las configuraciones previas que se han modificado como el sentido de giro, velocidad configurada o ajustada, y el estado actual en el que se encuentra el motor.

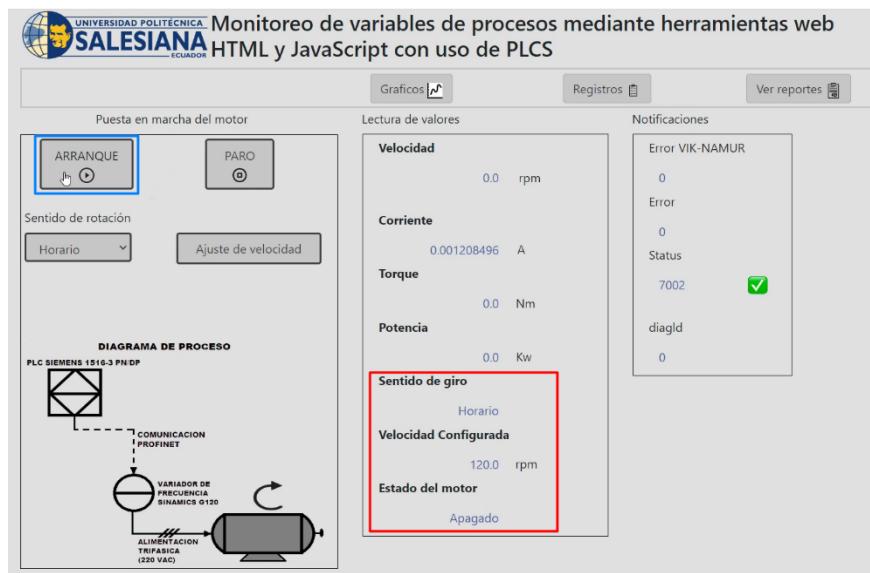


Figura 5.29 Estado de variables de entrada en la aplicación WEB.

Cuando se presione el botón ARRANQUE, el variador en un tiempo aproximado de tres segundos, establecerá la velocidad del motor en 120 RPM. En la figura 5.30, se puede observar la velocidad configurada y la velocidad actual enviada por el variador de frecuencia. De acuerdo a los valores de velocidad configurada y velocidad actual, existe una ligera variación, esto se debe a propiedades propias en el funcionamiento de un motor asincrónico.

En la sección de “Lectura de valores”, también se puede observar los valores actuales de corriente, torque y potencia activa cuando el motor está girando a 120 RPM. Mediante la sección de “Notificaciones”, en el proceso no existen errores ni fallas en el variador de frecuencia. Esto se puede notar en el valor de las variables “Error VIK-NAMUR”, “Error”, “diagId”, que muestran un valor de “0” indicando la presencia nula de errores. Por otro lado, la variable “Status” indica un valor de “7002”, cuyo valor refleja que no existen fallas en el variador y mediante el ícono en verde, indica al usuario u operario que el proceso marcha correctamente.

Otra característica a resaltar, es el cambio en la imagen del motor. Mediante el color blanco, el motor indica que se ha encendido esto como una descripción rápida para el operario del estado actual en el que se encuentra el motor.

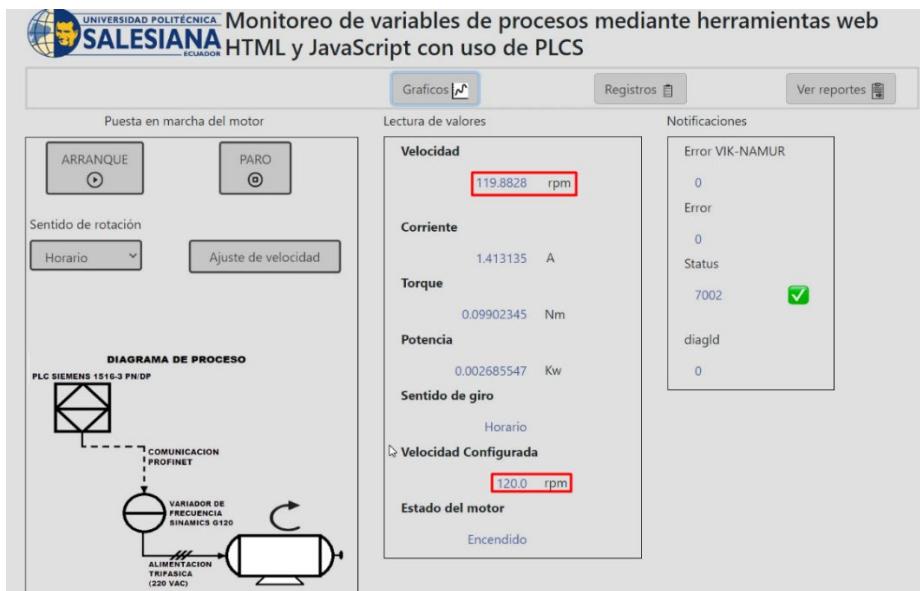


Figura 5.30 Encendido del motor.

Mientras tanto en el variador, se podrá observar la velocidad de 120 RPM regulada desde la aplicación WEB por medio del panel operador IOP.

La figura 5.31, muestra dos parámetros; la velocidad real, velocidad que posee el motor durante su funcionamiento, y la tensión de salida, proporcionada por el variador de frecuencia al motor.



Figura 5.31 Parámetros de salida visualizados en el panel operador IOP.

5.5.2 Gestión de alarmas y advertencias

Otra de las funciones implementadas en la aplicación WEB, es la gestión de advertencias ante determinados sucesos. Como una advertencia de prueba, se ha impuesto la condición; si el motor sobrepasa los 1400 RPM, se usará una imagen como alerta para el usuario.

Por medio del botón de “Ajuste de velocidad”, se puede configurar la velocidad hasta el valor máximo de 1440 RPM usando la barra deslizante, como se muestra en la figura 5.32.

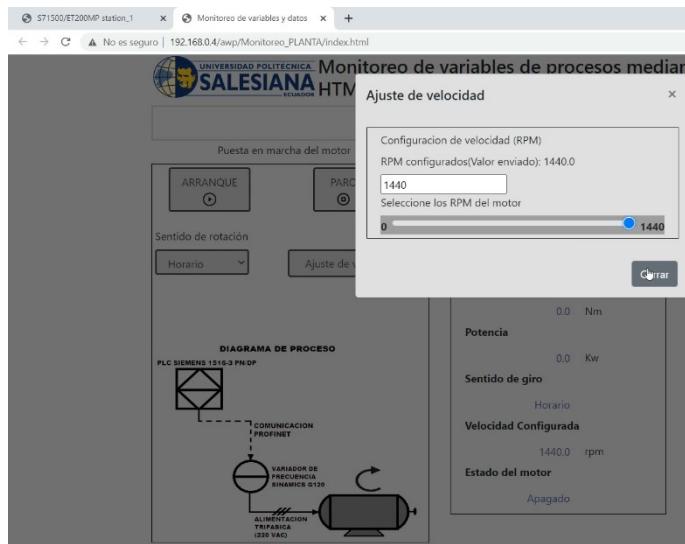


Figura 5.32 Control de velocidad mediante la barra deslizante.

Luego de configurar una velocidad de prueba, se puede evidenciar los resultados en la figura 5.33. Por medio de las dos imágenes, se puede notar que el ícono de alerta de la parte derecha de la figura solo se muestra al sobrepasar los 1400 RPM.

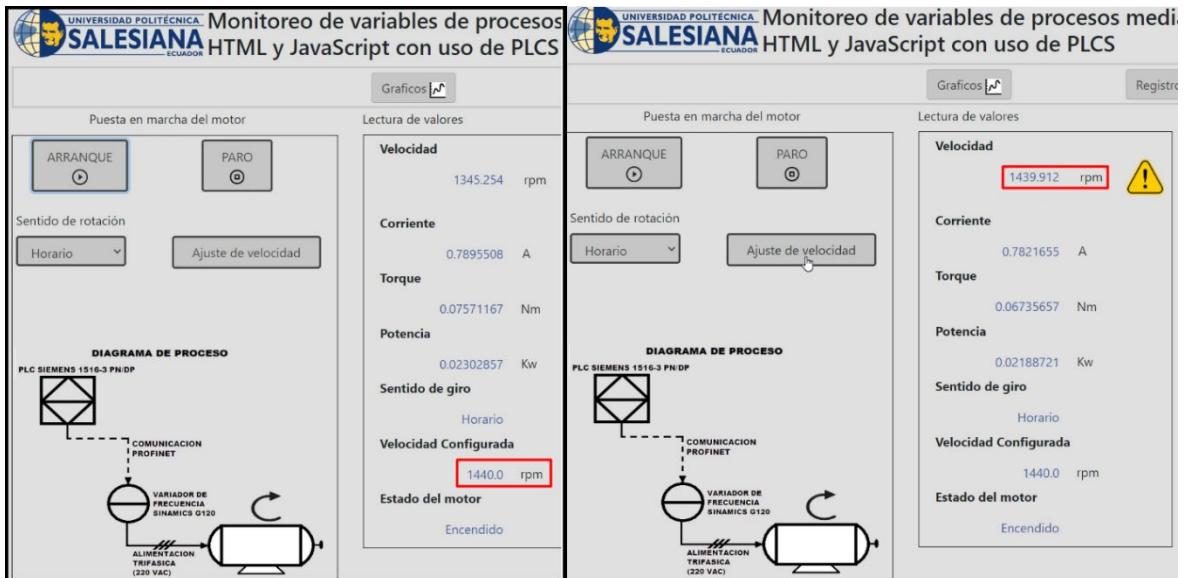


Figura 5.33 Advertencia gestionada por la aplicación WEB.

Por otro lado, la aplicación presenta otra función para la gestión de alarmas ante fallas denominadas “Prioritarias”. Para probar esta función, se ha implementado el uso de imágenes y sonidos cuando se presente el error “8601” en la variable “Status” del bloque función.}

El error “8601” indica que ha existido un error al “escribir los datos del telegrama” y una de las causas es la desconexión del PLC y del variador. Por lo tanto, la falla causante puede ocurrir cuando el cable o medio (Ethernet) que conecta el PLC y el variador usando el protocolo PROFINET, se desconecte en uno de los dos dispositivos.

En la figura 5.34, se puede observar que se ha retirado el cable Ethernet que comunica la PLC y el variador de frecuencia, ocasionando que en el panel operador IOP, aparezcan las fallas y alarmas activas.

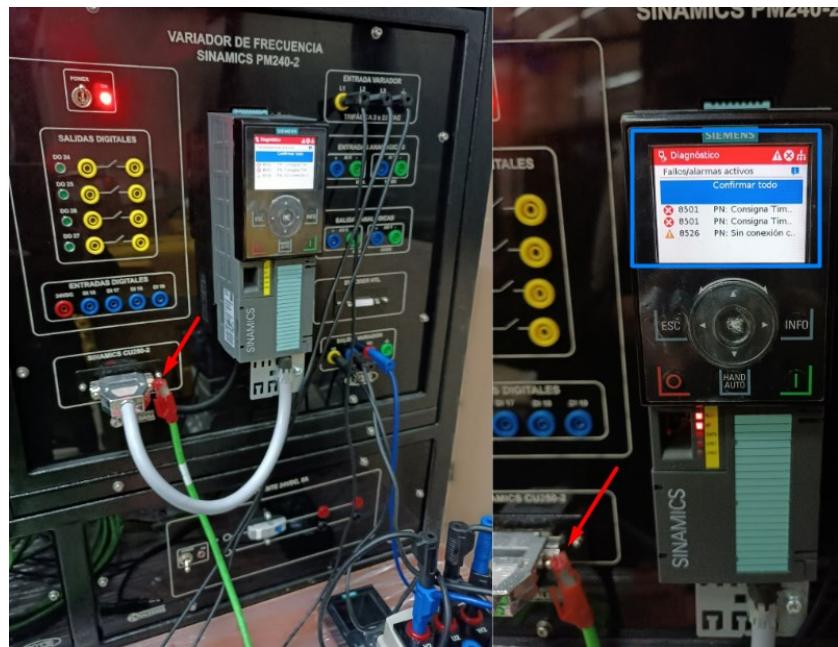


Figura 5.34 Desconexión del PLC y el variador de frecuencia.

Una vez que se produzca una “Falla de conexión” entre el PLC y el variador, se verá reflejado la siguiente advertencia en la aplicación WEB, como lo indica la figura 5.35. Por medio de un ícono (imagen) de color rojo, se informa de manera rápida al usuario que existe una falla prioritaria que deberá ser atendida. Al mismo tiempo que se detecta la falla, la aplicación reproduce un sonido como alarma, la cual solo será silenciada hasta que se corrija la falla ocasionada. Si se observa el recuadro rojo, se muestra que se ha dejado de recibir datos del proceso. Además, la variable de Notificaciones, “diagId”, muestra un valor de “80A1”, indicando que existe un error al acceder a un dispositivo periférico.

Monitoreo de variables de procesos mediante herramientas web			
Graficos	Registros	Ver reportes	
Puesta en marcha del motor	Lectura de valores	Notificaciones	
ARRANQUE PARO	Velocidad Corriente Torque Potencia	Error VIK-NAMUR diagId 80A1	
Sentido de rotación Antihorario Ajuste de velocidad	Sentido de giro Antihorario Velocidad Configurada 120.0 rpm Estado del motor Apagado		
DIAGRAMA DE PROCESO PLC SIEMENS 1516-3 PNDP			
COMUNICACION PROFINET VARIADOR DE FRECUENCIA SINAMICS G120 ALIMENTACION TRIFASICA (220 VAC)			

Figura 5.35 Alarmas gestionadas por la aplicación WEB.

Para corregir la falla presentada en el variador, se volverá a conectar el cable Ethernet que conecta el PLC y el variador de frecuencia. Luego en el variador, se posicionará en la opción “Confirmar todo”, y se presionará el botón “OK”, hasta que aparezca el mensaje “No hay fallos/alarmas presentes”. La figura 5.36, muestra el proceso antes mencionado. Al presionar el botón “ESC”, se accederá nuevamente a la interfaz de inicio.

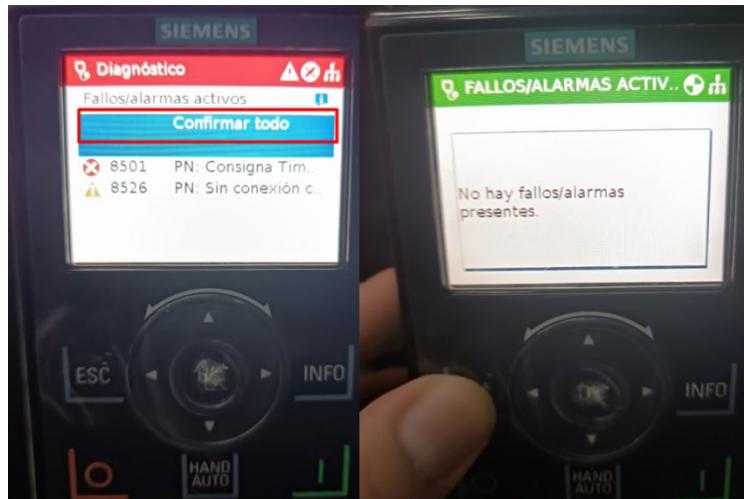


Figura 5.36 Corrección de las fallas de conexión entre el PLC SIEMENS y el variador de frecuencia.

5.5.3 Pruebas de los botones de la sección “barra de navegación” y la aplicación diseñada en Laravel

A continuación, se verificará el funcionamiento de cada uno de los botones de navegación, los cuales permiten acceder a las otras secciones de la aplicación WEB.

- *Botón "Gráficos"*

En la figura 5.37, se observa el botón Gráficos, el cual despliega en una nueva pestaña del navegador WEB, la gráfica denominada “Tendencia”. La gráfica estará en función de la velocidad actual (velocidad leída) y desplazándose en el eje X en función del tiempo.

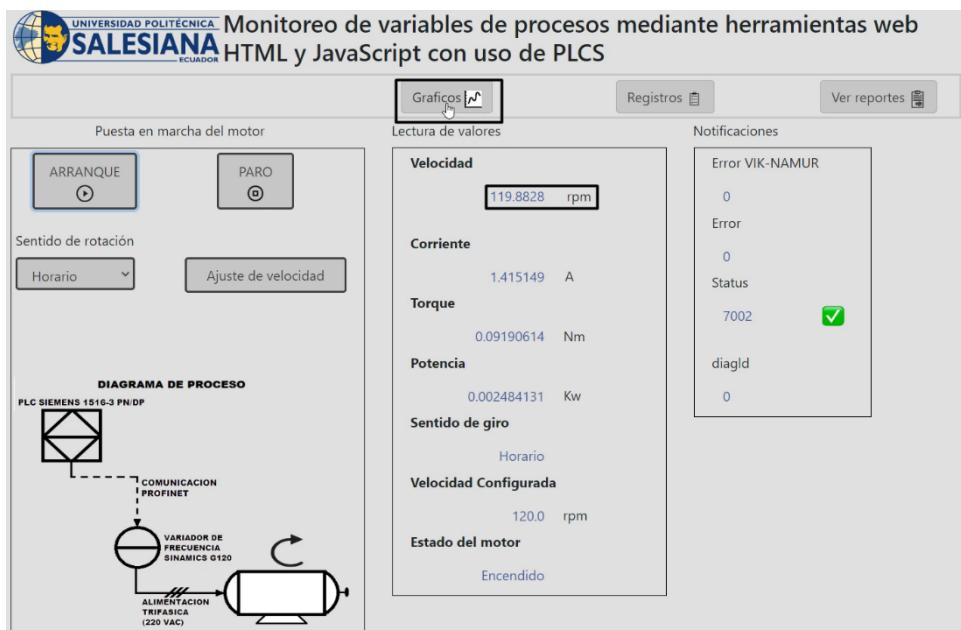


Figura 5.37 Botón “Gráficos” de la aplicación WEB.

Cuando se presione el botón “Gráficos” se abrirá una nueva pestaña que despliega una página WEB que contiene los dos gráficos relacionados o en función de la velocidad del motor. La figura 5.38, muestra los dos gráficos en función de la velocidad entregada por el motor.

Mediante la primera gráfica, se puede evidenciar que esta ha iniciado en el tiempo en el cual se ha presionado el “Botón Gráficos”. La gráfica inicia en el valor de 120 RPM, dado que es la velocidad configurada actualmente y también dicho valor se ve reflejado en el segundo grafico de tipo “Pastel”.

El segundo grafico igual que el primero, estará en función de la velocidad actual del motor, y posee una forma semicircular. El grafico posee un color que se irá incrementando en un rango de 0 a 1440 RPM. En la figura 5.38, el grafico posee un color verde claro, el cual ira cambiando según se incremente el valor de la variable de la velocidad del motor.

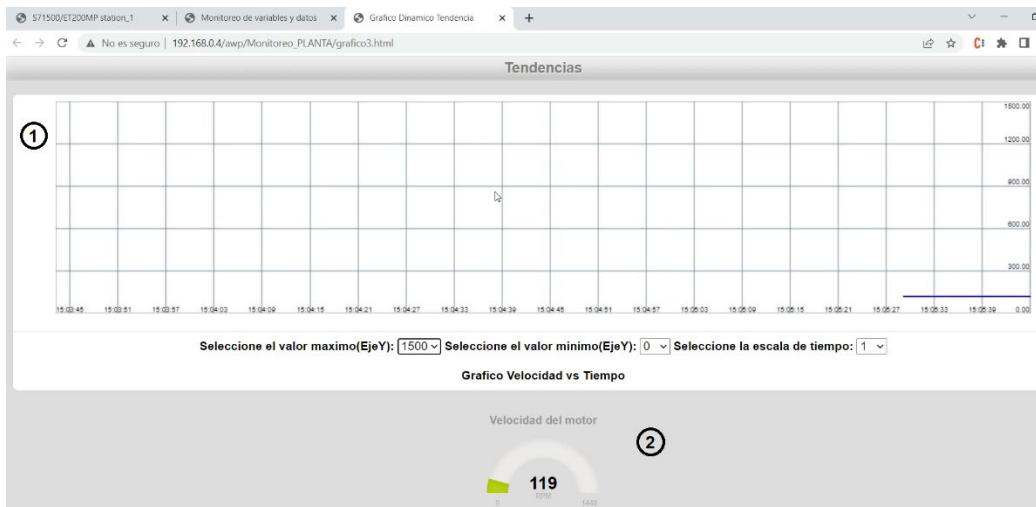


Figura 5.38 “Gráficas” en función de la variable “velocidad actual” en la aplicación WEB.

Para observar el comportamiento que tiene la gráfica “Tendencia”, se puede variar la velocidad mediante la barra deslizante y luego apreciar los diferentes cambios que tiene la gráfica. Mediante la figura 5.39, se ha establecido en 475 RPM la velocidad del motor, dando como resultado el comportamiento de la gráfica “Tendencias” de la figura 5.40.

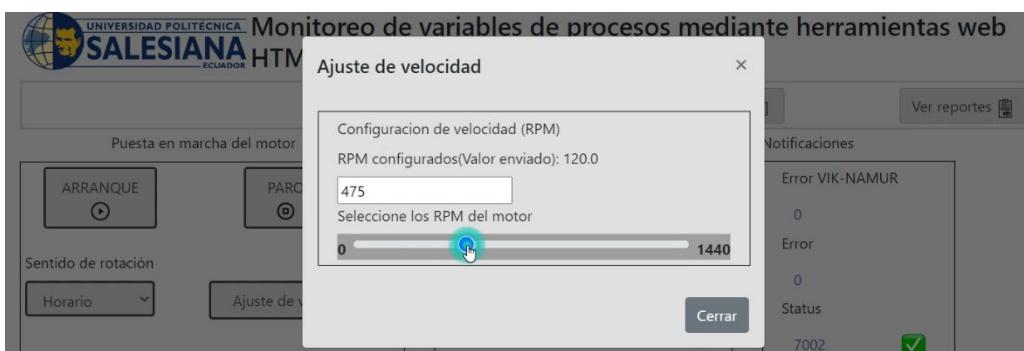


Figura 5.39 Control de la velocidad del motor en 475 RPM desde la barra deslizante.

En la figura 5.40, se observa por medio de la flecha roja, el punto en que se inició el cambio de velocidad de 120 a 475 RPM. Mientras que, con la flecha negra, se observa cuando el motor ha alcanzado la velocidad de 475 RPM, manteniéndose estable.

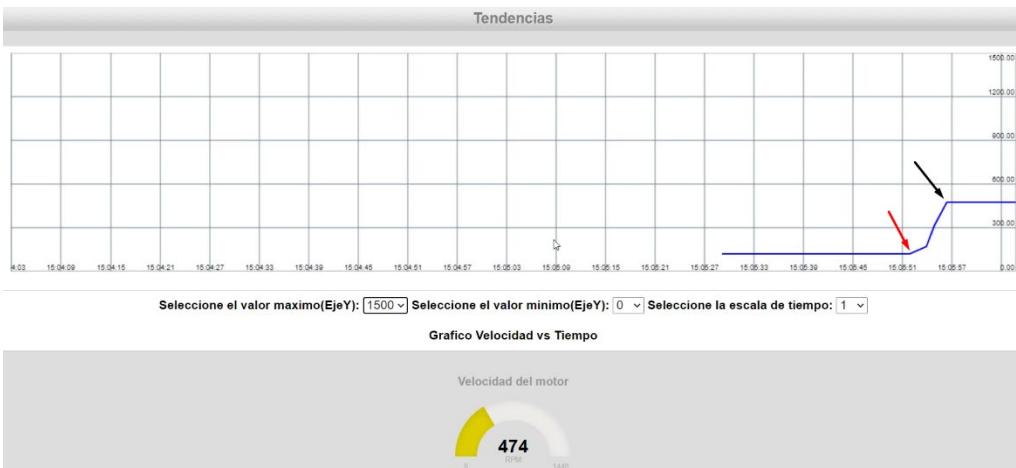


Figura 5.40 Grafica tendencia cuando el motor alcanza los 475 RPM.

Como otro método para verificar los valores que se adquieren en la aplicación WEB son correctos, es visualizarlos desde el programa TIA PORTAL (STEP 7). En la figura 5.41, se ha ubicado en la sección “Bloques de Programa” del proyecto cargado en el PLC, para visualizar las configuraciones establecidas en el bloque función que controla el variador de frecuencia.

Para visualizar los valores actuales de las variables de entrada y salida del bloque función, es necesario establecer una conexión online, cuando se presione la opción enmarcada en rojo “Establecer conexión online”. Luego, si se presiona el icono “Activar Observación”, se podrá acceder a los datos actuales de las variables de entrada y salida.

Mediante la fecha color negro en la figura 5.41, se puede observar que la variable “vel actual”, refleja un valor de 874.9512 RPM, cuyo valor se puede contrastar con el valor que proporciona el grafico “Pastel” de la figura 5.42. Por lo tanto, los valores adquiridos mediante la aplicación WEB, son correctos y coincidentes con los valores que se envían desde el variador de frecuencia al PLC SIEMENS.

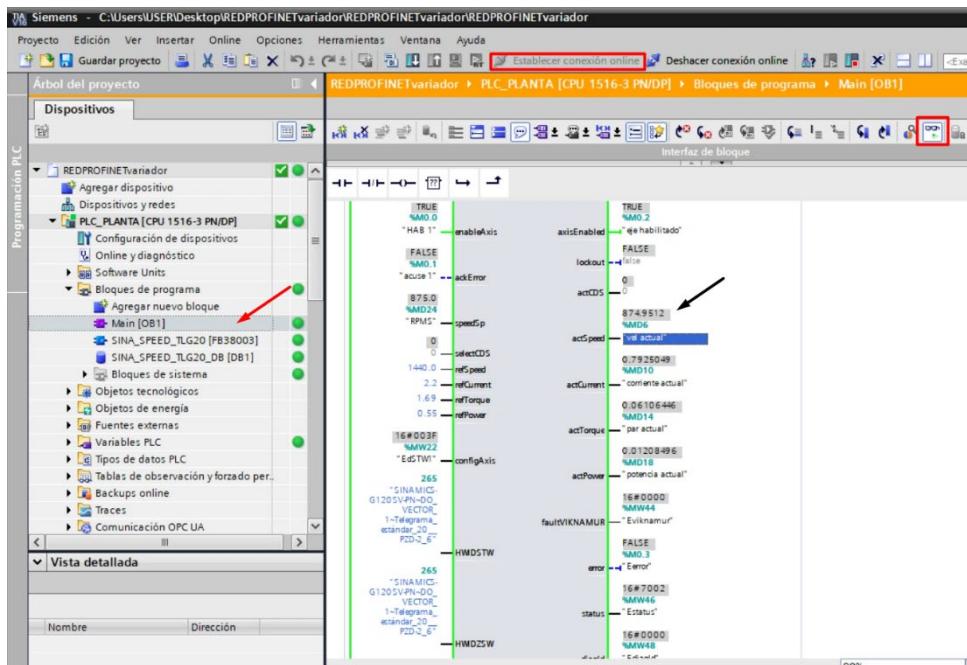


Figura 5.41 Visualización de las variables de entrada y salida del bloque función desde TIA PORTAL V16.

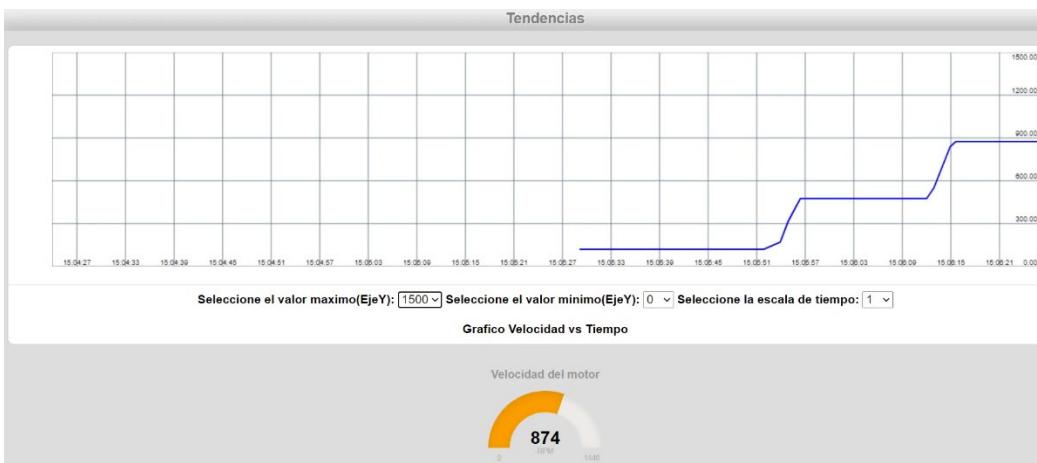


Figura 5.42 Grafica “Tendencia” cuando el motor ha alcanzado 875 RPM.

- *Botón “Registros”*

Este botón permite acceder a la interfaz encargada de la gestión de la base de datos, es decir visualizar cada uno de los datos adquiridos desde la aplicación WEB mediante la ayuda del framework “LARAVEL”. La interfaz de Laravel permite la visualización de cada uno de los datos que se almacenan en un servidor local de la PC, siempre y cuando el proceso de control este activo y se halla abierto o accedido a la interfaz de la aplicación creada.

Si se presiona el botón “Registros”, se abrirá una nueva pestaña en el buscador WEB, accediendo a la interfaz de Laravel y al dominio local “monitoreo.com”. La figura 5.43, muestra la ventana desplegada al presionar el botón Registros.

En la interfaz, se puede notar cada uno de los valores de las variables de salida y variables de entrada que se almacenan en la base de datos local. Cada una de las 7 variables son almacenadas cada segundo y por medio del parámetro “Fecha”, se podrá conocer la hora y fecha en la que se adquirió el conjunto de datos en la base de datos.

No	Velocidad	Corriente	Torque	Potencia	Sentido	Regulador1	Estadomotor	Fecha	Show	Edit	Delete
1	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:16	Show	Edit	Delete
2	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:15	Show	Edit	Delete
3	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:14	Show	Edit	Delete
4	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:13	Show	Edit	Delete
5	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:12	Show	Edit	Delete
6	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:12	Show	Edit	Delete
7	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:11	Show	Edit	Delete
8	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:10	Show	Edit	Delete
9	0.0	0.001208496	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:09	Show	Edit	Delete
10	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:08	Show	Edit	Delete
11	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:07	Show	Edit	Delete
12	0.0	0.001208496	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:06	Show	Edit	Delete

Figura 5.43 Interfaz de acceso a los datos almacenados en el servidor local del equipo.

En la figura 5.44, se muestra en la sección final de la interfaz, el acceso a las páginas previas que contienen datos obtenidos en fechas anteriores.

The screenshot shows a web-based monitoring application. At the top, there are three tabs: 'S71500/ET200MP station_1' (active), 'Monitoreo de variables y datos' (inactive), and 'Laravel' (inactive). Below the tabs is a table with 20 rows of data. The columns include: 'U.U', 'U.U', 'U.U', 'Horario', '0/5.0', 'Apagado', and '2022-08-03 17:45:11'. Each row has a set of 'Show', 'Edit', and 'Delete' buttons. At the bottom of the table is a navigation bar with page numbers from 1 to 1300, with the current page highlighted in blue.

U.U	U.U	U.U	Horario	0/5.0	Apagado	2022-08-03 17:45:11	Show	Edit	Delete		
8	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:10	Show	Edit	Delete
9	0.0	0.001208496	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:09	Show	Edit	Delete
10	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:08	Show	Edit	Delete
11	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:07	Show	Edit	Delete
12	0.0	0.001208496	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:06	Show	Edit	Delete
13	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:05	Show	Edit	Delete
14	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:04	Show	Edit	Delete
15	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:03	Show	Edit	Delete
16	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:02	Show	Edit	Delete
17	0.0	0.001208496	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:01	Show	Edit	Delete
18	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:45:00	Show	Edit	Delete
19	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:44:59	Show	Edit	Delete
20	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:44:58	Show	Edit	Delete

Figura 5.44 Datos almacenados desde la aplicación WEB.

- *Botón “Reportes”*

Este botón tiene la función de “direccional” a la interfaz de “Reportes” de la base de datos, implementada igualmente con el framework LARAVEL. En esta sección, se podrá descargar y acceder a los datos almacenados desde los controles implementados. Para visualizar o acceder y descargar los valores, se ingresará una fecha y hora de inicio y fin, con la finalidad de establecer un rango de datos establecidos. En la figura 5.45, se muestra la interfaz Reportes (recuadro verde) donde se podrán acceder a un conjunto de datos, cuando se establezca un rango de datos mediante el control enmarcado con el recuadro azul.

The screenshot shows a web-based monitoring application. At the top, there is a header with the logo of the Universidad Politécnica Salesiana and the text "Monitoreo de variables de procesos mediante herramientas web HTML y JavaScript con uso de PLCs". Below the header, there are three tabs: 'Graficos' (selected), 'Registros', and 'Reportes' (highlighted with a green box). The main content area contains a form for selecting a date range. The form fields are: 'Fecha de inicio' (dd/MM/yyyy) and 'Fecha de fin' (dd/MM/yyyy). Below these fields is a dropdown menu labeled 'Selecciona el tipo de reporte' and a 'Submit' button. A red box highlights the date range selection area.

Figura 5.45 Interfaz “Reportes”.

En la figura 5.46, se muestra como se ha establecido una fecha y hora de inicio y fin cuando se ha pulsado el icono señalado por el cuadro rojo. Luego del paso anterior, se procederá con el siguiente paso; definir el tipo de “Reporte” a escoger.

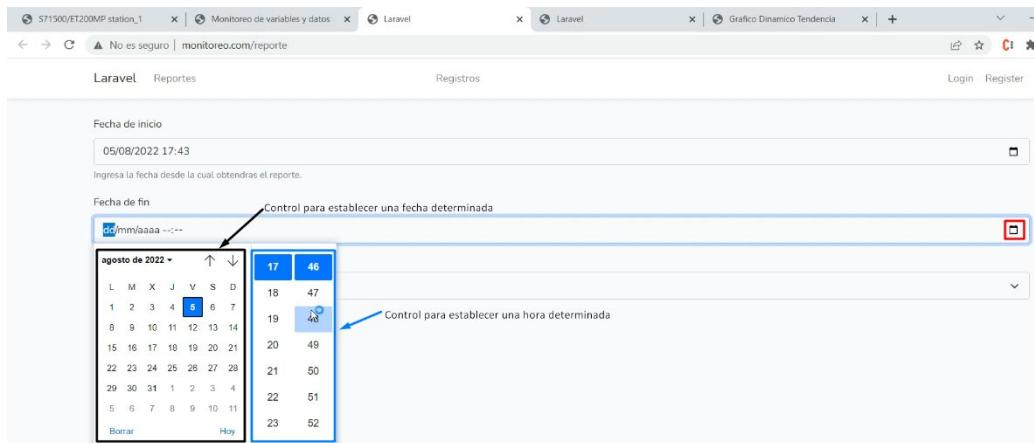


Figura 5.46 Configuración de fecha y hora de inicio y fin para descarga y/o visualización de datos del proceso.

Como una prueba rápida, se ha seleccionado un rango de datos que se han recopilado durante dos minutos (17:43-17:45), para luego proceder a escoger el tipo de reporte como lo indica la figura 5.47, mediante el recuadro rojo.

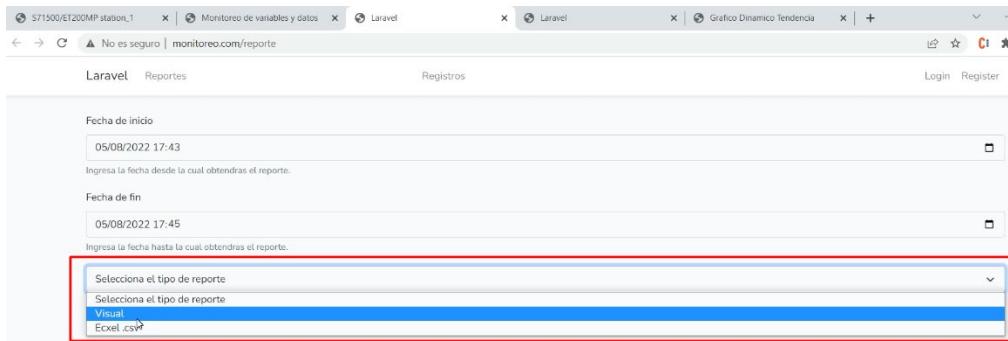


Figura 5.47 Selección de tipo de “Reporte”.

Si se escoge la opción reporte “Visual” como se indica en la figura 5.48, se mostrará mediante una nueva pestaña el conjunto de datos obtenido en el rango de fecha y hora definido anteriormente. Los datos obtenidos se muestran en la figura 5.49 y figura 5.50, los cuales están ordenados de forma ascendente según su fecha de adquisición.

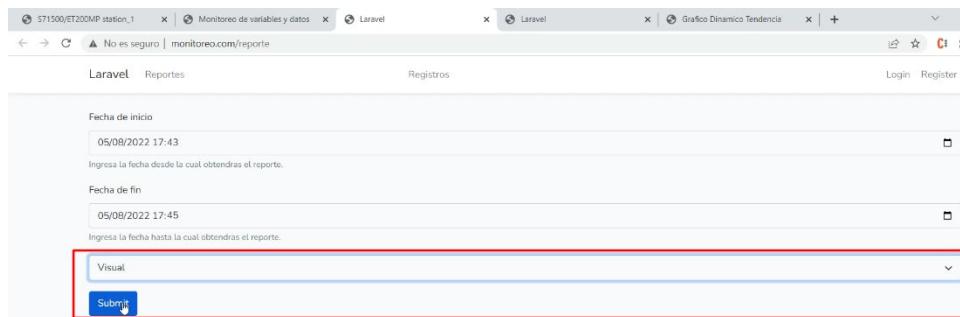


Figura 5.48 Selección de Reporte “Visual”.

No	Velocidad	Corriente	Torque	Potencia	Sentido	Regulador1	Estadomotor	Fecha
0	914.4141	0.8056641	0.0705542	0.0146698	Horario	1440.0	Encendido	2022-08-05 17:43:01
1	1055.479	0.7982788	0.07210144	0.01728821	Horario	1440.0	Encendido	2022-08-05 17:43:02
2	1200.059	0.7922363	0.07406128	0.02000733	Horario	1440.0	Encendido	2022-08-05 17:43:03
3	1345.254	0.7895508	0.07571167	0.02302957	Horario	1440.0	Encendido	2022-08-05 17:43:04
4	1439.824	0.784314	0.06900696	0.02245789	Horario	1440.0	Encendido	2022-08-05 17:43:05
5	1439.912	0.7836426	0.06869751	0.02232361	Horario	1440.0	Encendido	2022-08-05 17:43:06
6	1439.912	0.7831055	0.06880067	0.02235718	Horario	1440.0	Encendido	2022-08-05 17:43:07
7	1439.912	0.7814941	0.06828491	0.02229004	Horario	1440.0	Encendido	2022-08-05 17:43:08
8	1439.912	0.7806885	0.06828491	0.02225647	Horario	1440.0	Encendido	2022-08-05 17:43:09
9	1439.912	0.7810913	0.06828491	0.02215576	Horario	1440.0	Encendido	2022-08-05 17:43:11
10	1439.912	0.7829712	0.06797547	0.02222229	Horario	1440.0	Encendido	2022-08-05 17:43:11
11	1439.912	0.7837769	0.06787232	0.02205505	Horario	1440.0	Encendido	2022-08-05 17:43:12
12	1439.912	0.7837769	0.06776917	0.02202149	Horario	1440.0	Encendido	2022-08-05 17:43:13
13	1439.912	0.7829712	0.06766602	0.02205505	Horario	1440.0	Encendido	2022-08-05 17:43:14
14	1439.912	0.780957	0.06766602	0.02208862	Horario	1440.0	Encendido	2022-08-05 17:43:15
15	1439.912	0.780957	0.06745972	0.02192078	Horario	1440.0	Encendido	2022-08-05 17:43:16

Figura 5.49 Reporte Visual con cada una de las variables del proceso.

No	Velocidad	Corriente	Torque	Potencia	Sentido	Regulador1	Estadomotor	Fecha
101	674.9121	0.8024414	0.05755738	0.008761597	Horario	675.0	Encendido	2022-08-05 17:44:43
102	674.9121	0.8020386	0.05796998	0.008761597	Horario	675.0	Encendido	2022-08-05 17:44:44
103	674.9121	0.8031128	0.05714478	0.008795166	Horario	675.0	Encendido	2022-08-05 17:44:44
104	674.9121	0.8041871	0.05745423	0.008728027	Horario	675.0	Encendido	2022-08-05 17:44:45
105	674.9121	0.8049927	0.05766053	0.008728027	Horario	675.0	Encendido	2022-08-05 17:44:46
106	674.9121	0.8045899	0.05755738	0.008761597	Horario	675.0	Encendido	2022-08-05 17:44:47
107	674.9121	0.8041871	0.05745423	0.008728027	Horario	675.0	Encendido	2022-08-05 17:44:48
108	674.9121	0.8036499	0.05755738	0.008694459	Horario	675.0	Encendido	2022-08-05 17:44:49
109	674.9121	0.8024414	0.05755738	0.008761597	Horario	675.0	Encendido	2022-08-05 17:44:50
110	674.9121	0.8016357	0.05683533	0.008728027	Horario	675.0	Encendido	2022-08-05 17:44:51
111	604.8633	0.8106323	0.04930542	0.006713868	Horario	675.0	Apagado	2022-08-05 17:44:53
112	410.2734	0.854004	0.04961487	0.00453186	Horario	675.0	Apagado	2022-08-05 17:44:54
113	262.793	0.9614258	0.05456604	0.003356934	Horario	675.0	Apagado	2022-08-05 17:44:55
114	118.916	1.380505	0.09799195	0.002618408	Horario	675.0	Apagado	2022-08-05 17:44:56
115	0.703125	0.02618408	0.0002062988	0.0	Horario	675.0	Apagado	2022-08-05 17:44:57
116	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:44:58
117	0.0	0.001074219	0.0	0.0	Horario	675.0	Apagado	2022-08-05 17:44:59

Figura 5.50 Datos totales adquiridos en el “Reporte Visual”.

Por otro lado, si se selecciona un reporte de tipo “Excel.csv”, se podrá descargar un archivo de tipo csv compatible con el programa Microsoft Excel. Los datos descargados estarán dentro del conjunto o del rango de datos definidos mediante los controles de la interfaz de LARAVEL. Por medio de la figura 5.51, se ha seleccionado el tipo de reporte “Excel.csv” dando como resultado el archivo enmarcado por el recuadro azul cuando se ha presionado el botón “Submit”.

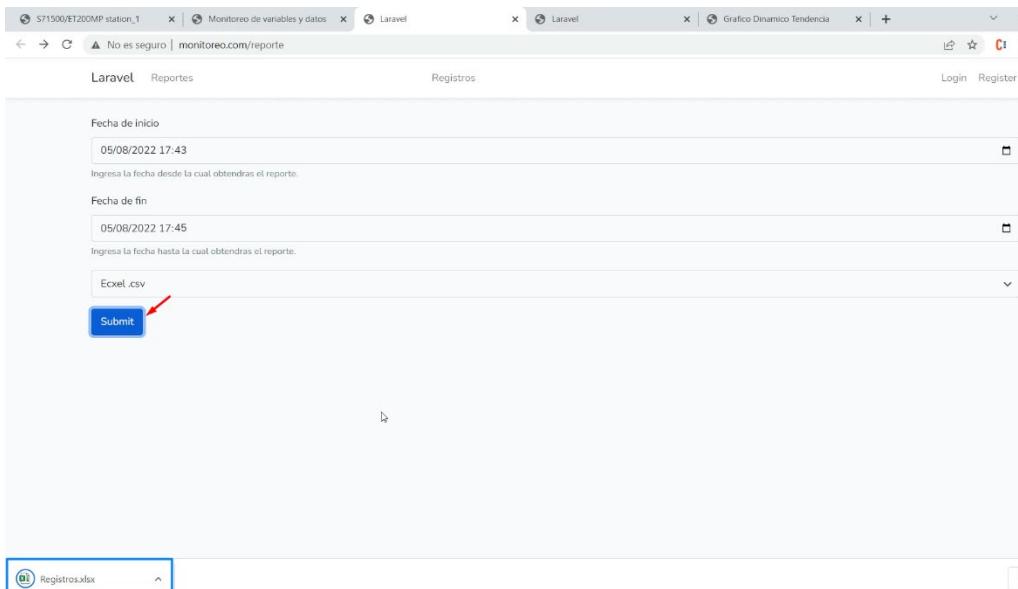


Figura 5.51 Reporte “Registros.xlsx” descargado desde la interfaz de Laravel.

Como una prueba final, se puede corroborar que el archivo descargado, es compatible con el programa “Microsoft Excel” y se puede trabajar con el conjunto de datos del archivo csv, como lo muestra la figura 5.52. En la figura, se visualiza el conjunto de datos provenientes de las variables leídas por la aplicación WEB y ordenados mediante filas y columnas para un próximo análisis.

	A	B	C	D	E	F	G	H	I	J	K	L
1	#	velocidad	corriente	Torque	Potencia	sentido	regulador1estadoMo	created	updated			
2	25884	914,4141	0,805664	0,070554	0,01467	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:01.000000Z		
3	25885	1055,479	0,798279	0,072101	0,017288	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:02.000000Z		
4	25886	1200,059	0,792236	0,074061	0,020007	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:03.000000Z		
5	25887	1345,254	0,789551	0,075712	0,023029	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:04.000000Z		
6	25888	1439,824	0,784314	0,069007	0,022458	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:05.000000Z		
7	25889	1439,912	0,783643	0,068698	0,022324	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:06.000000Z		
8	25890	1439,912	0,783106	0,068801	0,022357	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:07.000000Z		
9	25891	1439,912	0,781494	0,068285	0,022299	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:08.000000Z		
10	25892	1439,912	0,780689	0,068285	0,022256	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:09.000000Z		
11	25893	1439,912	0,781091	0,068285	0,022156	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:11.000000Z		
12	25894	1439,912	0,782971	0,067975	0,022223	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:11.000000Z		
13	25895	1439,912	0,783777	0,067872	0,022055	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:12.000000Z		
14	25896	1439,912	0,783777	0,067769	0,022021	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:13.000000Z		
15	25897	1439,912	0,782971	0,067666	0,022055	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:14.000000Z		
16	25898	1439,912	0,780957	0,067666	0,022089	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:15.000000Z		
17	25899	1439,912	0,780957	0,06746	0,021921	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:16.000000Z		
18	25900	1439,912	0,782166	0,067357	0,021887	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:18.000000Z		
19	25901	1439,912	0,78324	0,067253	0,021887	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:18.000000Z		
20	25902	1439,912	0,78324	0,06715	0,021887	Horario	1440	Encendido	2022-08-0	2022-08-05T22:43:19.000000Z		
21	25903	1397,285	0,782166	0,059311	0,018698	Horario	775	Encendido	2022-08-0	2022-08-05T22:43:21.000000Z		
22	25904	1250,156	0,78324	0,057248	0,016248	Horario	775	Encendido	2022-08-0	2022-08-05T22:43:21.000000Z		
23	25905	1109,619	0,784448	0,055288	0,013864	Horario	775	Encendido	2022-08-0	2022-08-05T22:43:22.000000Z		
24	25906	965,5664	0,787402	0,053741	0,011749	Horario	775	Encendido	2022-08-0	2022-08-05T22:43:23.000000Z		
25	25907	820,9863	0,793579	0,051781	0,009567	Horario	775	Encendido	2022-08-0	2022-08-05T22:43:24.000000Z		
26	25908	775,0195	0,797205	0,058589	0,010272	Horario	775	Encendido	2022-08-0	2022-08-05T22:43:25.000000Z		
27	25909	775,0195	0,797876	0,058486	0,010272	Horario	775	Encendido	2022-08-0	2022-08-05T22:43:27.000000Z		
28	25910	775,0195	0,798547	0,058073	0,010172	Horario	775	Encendido	2022-08-0	2022-08-05T22:43:28.000000Z		
29	25911	775,0195	0,797876	0,058383	0,010272	Horario	775	Encendido	2022-08-0	2022-08-05T22:43:28.000000Z		
30	25912	664,1016	0,806201	0,050646	0,00762	Horario	775	Apagado	2022-08-0	2022-08-05T22:43:29.000000Z		

Figura 5.52 Archivo “Registros.xlsx” abierto en el entorno Microsoft Excel.

Capítulo 6

Conclusiones y recomendaciones

6.1 Conclusiones

El objetivo principal del proyecto técnico tuvo el objetivo principal de analizar las herramientas proporcionadas por HTML y JavaScript para la creación de páginas web que gestionan datos y variables usando autómatas de la firma SIEMENS; dicha página web pudo ser implementada con éxito mediante el uso de entornos como Visual Studio Code y TIA PORTAL V16.

Al usar los distintos elementos de control que proporciona el lenguaje de html, se pueden controlar las diferentes variables de un proceso industrial, mediante el uso de botones, barras deslizantes, elementos multiopcion, etc. Además, dada la flexibilidad y los diferentes elementos que existen de html, se puede crear de manera rápida una aplicación web para controlar un proceso industrial. Una de las ventajas que se pudo corroborar de html, es su facilidad de implementación de los elementos html, ya que, por medio de etiquetas, se puede ir estructurando de manera rápida un documento o página web compatible con diferentes navegadores web.

Otra de los recursos que proporciona html, es su complementación con el lenguaje de CSS, con el que se pudo crear botones, barras de control, recuadros de texto, con un estilo definido para lograr visualizar los datos y controles de manera óptima. El lenguaje CSS, permitió configurar las propiedades de los elementos de control con la finalidad de estructurar los controles de una manera ordena en la interfaz que se presenta al usuario o cliente.

Por otro lado, el lenguaje de JavaScript ofrece una amplia gama de librerías y scripts de código abierto que permiten implementar gráficos que estén en función de datos obtenidos de una variable de un proceso. Uno de estos gráficos implementado, fue la gráfica tendencia la cual proporciona en tiempo real, el valor de velocidad del motor alrededor del tiempo. De la misma manera, mediante JavaScript, se pudo implementar las funciones necesarias para enviar datos al servidor web y así poder controlar las variables de entrada. Por último, mediante JavaScript, se pudo establecer la gestión de alarmas y notificaciones antes determinadas condiciones establecidas, ofreciendo una gran herramienta para la toma oportuna de decisiones ante fallas en el proceso.

Mediante el framework Laravel y el servidor XAMPP, se implementó una base de datos y una aplicación que permitió el acceso a cada uno de los datos almacenados del sistema de velocidad implementado. Siendo esta una alternativa rápida y sencilla debido a las características de código abierto de MYSQL y a la estructura dividida de Laravel. Además de la facilidad de implementación de un proyecto con Laravel, mediante la función implementada de exportación o descarga de datos, se puede usar el archivo csv generado, para un análisis posterior de los datos, ayudando así a predecir futuras fallas o un mal comportamiento de los equipos.

Luego de que se implementó y configuro la página web en el PLC 1516-3 PN/DP, se controló cada una de las variables sin retardo alguno, y sin problemas, ya que por medios como ethernet industrial para comunicaciones PROFINET, permiten comunicaciones de hasta 100 Mbps. Con ello, se corrobora la ventaja en cuanto a velocidad cuando se usa el servidor web mediante el medio Ethernet, y el uso casi nulo de cableado para controlar periféricos como el variador de frecuencia utilizado.

6.2 Recomendaciones y trabajos futuros

Durante el desarrollo del proyecto técnico, se requirió actualizar los autómatas a una versión compatible con el programa TIA PORTAL V16. Por lo que se sugiere actualizar el firmware del PLC a una versión actual o compatible con el programa de TIA PORTAL que se desee usar.

Por otro lado, se recomienda siempre revisar las conexiones PROFINET, ya que el PLC y el variador de frecuencia, se conectan por medio de este bus de campo, con la finalidad de evitar pérdidas de tiempo durante la implementación del servidor web.

Otra de las consideraciones que se deberán tomar en cuenta, será los permisos de acceso del equipo cliente al servidor web, por lo que se recomienda desactivar el firewall, para así evitar fallas de conexiones con el servidor.

Como un trabajo futuro a implementarse, se podrían gestionar funciones de seguridad tanto para la aplicación de html y la implementada con Laravel, con el propósito de que las aplicaciones no permitan el acceso a personal no autorizado, y se prevenga de ataques no deseados a los equipos del proceso.

Referencias Bibliográficas

- [1] Joby Antony, Basanta Mahato, Sachin Sharma, Gaurav Chitranshi. A web PLC using distributed web servers for data acquisition and control Web based PLC. India. 2011.
- [2] Siemens. S7-1500, SIMATIC Drive Controller, ET 200SP, ET 200pro Web server Function Manual. Properties of the Web server. Pag.14. 2019. [Online]. Available: <https://support.industry.siemens.com/cs/document/59193560/simatic-s7-1500-simatic-drive-controller-et-200sp-et-200pro-web-server?dti=0&lc=en-EC>
- [3] Siemens. SIMATIC (S7-1500 SERVIDOR WEB/Manual de Funciones). 2014. Alemania. [Online]. Available: https://cache.industry.siemens.com/dl/files/560/59193560/att_109205/v1/s71500_webserver_function_manual_es-ES_es-ES.pdf
- [4] Verónica Almache. Diseño de un HMI en web servers del PLC s7- 1200/1500 para el control de un proceso multivariable de un módulo didáctico para el laboratorio de hidráulica y neutrónica de la universidad de las fuerzas armadas ESPE extensión Latacunga. 2017. Latacunga.
- [5] D. Pedro Centeno Pomareta. INTRODUCCIÓN A TIA PORTAL CON S7-1500. Madrid. 2017.
- [6] SIEMENS. (7 de enero de 2022). Convertidores estándar SINAMICS G120. [Online]. Available: <https://mall.industry.siemens.com/mall/es/WW/Catalog/Products/10215579>
- [7] SIEMENS. (2016). S7-1500-CPU 1516-3 PN/DP (6ES7516-3AN01-0AB0) Manual de producto. [Online]. Available: https://cache.industry.siemens.com/dl/files/914/59191914/att_915308/v1/s71500_cpu1516_3_pn_dp_manual_es-ES_es-ES.pdf
- [8] Roberto AS. (11 de febrero de 2017). Siemens Sinamics G120 por Profinet IO. [Online]. Available: <https://automatizacioncavanilles.blogspot.com/2017/02/siemens-sinamics-g120.html>.
- [9] Technology team. (21 de noviembre de 2018). What is telegram in drive. [Online]. Available: <https://support.industry.siemens.com/forum/ww/en/posts/what-is-telegram-in-drive/203191/?page=0&pageSize=10>.
- [10] SIEMENS. (9/2015). SINAMICS G120C low voltage inverter (Operating Instructions). Setpoint Calculation-Overview of setpoint processing. (pág. 189). [Online]. Available: https://media.automation24.com/manual/se/109478830_G120C_BA7_0715_PI_eng.pdf
- [11] SIEMENS. (7/2019). Function blocks to control the SINAMICS with SIMATIC S7 in TIA Portal (SINAMICS S, G, V / communication / function block). Fundamentals-Cyclic Communication. (pág.16). [Online]. Available: <https://publikacje.siemensinfo.com/pdf/675/SIMATIC>
- [12] SIEMENS. (11/2019). Library LSINAExt Control of a SINAMICS drive via function blocks (SINAMICS / V1.0 / Control via function blocks). (pág.8-13). [Online]. Available: https://cache.industry.siemens.com/dl/files/655/109747655/att_1002477/v1/109747655_LSI_NAExt_DOC_v101_en.pdf?fbclid=IwAR3CxWyzJaioUrrFIPoajCzO60ilpnNJ3ubDFy6hIm_do1DlFKBi-5T2b8
- [13] SIEMENS. (05/2021) SIMATIC S7-1500, SIMATIC Drive Controller, ET 200SP, ET 200pro Servidor web. Programación de la instrucción WWW. (pág.148-149). [Online]. Available: <https://support.industry.siemens.com/cs/document/59193560/simatic-s7-1500-simatic-drive-controller-et-200sp-et-200pro-servidor-web?dti=0&dl=es&lc=en-WW>
- [14] Rockwell Automation. Process HMI Style Guide. May 2019. [Online]. Available: [proces-wp023_-en-p.pdf \(rockwellautomation.com\)](http://process-wp023_-en-p.pdf).
- [15] UNITRONICS. HE: Trends. [Online]. Available: https://www.unitronicsplc.com/Download/SoftwareHelp/UniLogic_Knowledgebase/HMI/HMI_Elements/HE_Trends.htm
- [16] Juan Ferrer Martínez. Aplicaciones web. Esquema de funcionamiento de un servidor web. Pág. 16-19.RA-MA.S.A. Editorial y Publicaciones. España. 2014.

- [17] William Harrel. HTML, CSS & JavaScript® Mobile Development FOR Dummies. Pag. 15. John Wiley & Sons, Inc. Hoboken, NJ.2011.
- [18] J. D. Gauchat. El gran libro de HTML5, CSS3 y Javascript. Pag. Primera edición. MARCOMBO, S.A. 2012. Barcelona
- [19] Pablo C. López José, Miguel A. Giráldez. La tercera revolución. Comunicación, tecnología y su nomenclatura en inglés. Pag.104-107. Netbiblo , S. L. España. Primera Edición. 2007.
- [20] SIEMENS. Creating and Using Own Web Pages for SIMATIC S7-1200 SIMATIC STEP 7 V11. (Pag.17-22). 05/2014. [Online]. Available: <https://support.industry.siemens.com/cs/document/58862931/creaci%C3%B3n-y-utilizaci%C3%B3n-de-p%C3%A1ginas-web-propias-en-el-s7-1200-?dti=0&dl=es&lc=en-US>
- [21] Mozilla. Referencia de Elementos HTML. (febrero 2021). [Online]. Available: <https://developer.mozilla.org/es/docs/Web/HTML/Element>
- [22] Mozilla. CSS. (julio 2021). [Online]. Available: <https://developer.mozilla.org/es/docs/Web/CSS>
- [23] Rock Content. Bootstrap: guía para principiantes de qué es, por qué y cómo usarlo. (abril 2020). [Online]. Available: <https://rockcontent.com/es/blog/bootstrap/>
- [24] Rock Content. Guía completa del Framework: qué es, cuáles tipos existen y por qué es importante en Internet. (enero 2020). [Online]. Available: <https://rockcontent.com/es/blog/framework/>
- [25] Rock Content. Bootstrap: guía para principiantes de qué es, por qué y cómo usarlo. (enero 2020). [Online]. Available: <https://rockcontent.com/es/blog/bootstrap/>
- [26] Laura Chuburu. Qué es JQuery y cómo implementarlo. (2020). [Online]. Available: <https://www.laurachuburu.com.ar/tutoriales/que-es-jquery-y-como-implementarlo.php>
- [27] Joe Walnes,Drew Noakes.SMOOTHIE Charts.[Online]. Available: <http://smoothiecharts.org/>
- [28] Oracle. ¿Qué es una base de datos?.2018. [Online]. Available: <https://www.oracle.com/mx/database/what-is-database/#relational>
- [29] Ayudaley. Base de datos distribuida. ¿Qué es? Características. [Online]. Available: <https://ayudaleyprotecciondatos.es/bases-de-datos/distribuida/>
- [30] UNIR. Bases de Datos NoSQL: qué son y cuáles son sus ventajas.2015. [Online]. Available: <https://www.unir.net/ingenieria/revista/bases-de-datos-nosql/>
- [31] IWEB. Una introducción a servidores de bases de datos.2014. [Online]. Available: <https://iweb.com/es/blog/una-introduccion-a-servidores-de-bases-de-datos>
- [32] HostGator. MySQL: qué es y cuáles son sus beneficios.2021. [Online]. Available: <https://www.hostgator.mx/blog/mysql-conoce-que-es-y-que-ventajas-tiene/>
- [33] Javatpoint. XAMPP TUTORIAL.2021. [Online]. Available: <https://www.javatpoint.com/xampp>
- [34] Mundobytes. Qué Es Xampp Usos, Características, Opiniones, Precios. [Online]. Available: <https://mundobytes.com/xampp/>
- [35] Desarrolloweb.Laravel.[Online].Available: <https://desarrolloweb.com/home/laravel#:~:text=Laravel%20es%20un%20popular%20framework,work,en%20el%20mundo%20de%20Internet>
- [36] Rafael Altuve. Qué es Laravel: Características y ventajas. [Online]. Available: <https://openwebinars.net/blog/que-es-laravel-caracteristicas-y-ventajas/>
- [37] Cesar Anton. Arquitectura de una aplicación en Laravel. [Online]. Available: <https://platzi.com/blog/arquitectura-laravel#:~:text=La%20arquitectura%20de%20Laravel%20es,con%20la%20base%20de%20datos.>
- [38] Javier Archeni. Importa las tablas de tu base de datos a un nuevo proyecto Laravel.2020. [Online]. Available: <https://javierarcheni.com/blog/importa-las-tablas-de-tu-base-de-datos-a-un-nuevo-proyecto-laravel/>
- [39] DesarrolloWeb. Introducción a modelos en Laravel.2015. [Online]. Available: <https://desarrolloweb.com/articulos/introduccion-modelos-laravel.html>
- [40] Duilio Palacios. Base de datos: Migraciones – Documentación de Laravel 6.2020.[Online]. Available: <https://styde.net/laravel-6-doc-base-de-datos->

migraciones/#:~:text=Las%20migraciones%20son%20como%20el,de%20datos%20de%20tu%20aplicaci%C3%B3n.

- [41] Codea. Migraciones en Laravel. [Online]. Available: <https://codea.app/blog/migraciones-en-laravel>
- [42] Oulub. Laravel 8.x / Base de datos: migraciones. [Online]. Available: <https://www.oulub.com/es-ES/Laravel/migrations#creating-columns>
- [43] Víctor Peña. Modelos y Eloquent ORM. [Online]. Available: <https://norvicsoftware.com/modelos-y-eloquent-orm-en-laravel-8/#:~:text=En%20Laravel%20los%20modelos%20son,%2C%20insertar%2C%20editar%20y%20borrar>.
- [44] Víctor Peña. Controladores en Laravel. [Online]. Available: <https://norvicsoftware.com/controladores-en-laravel-8/>
- [45] Norvic. Desarrollo web en Laravel. [Online]. Available: <https://norvicsoftware.com/desarrollo-web-con-laravel/>
- [46] Laravel Tip. ¿Qué son los Routes en Laravel? .[Online]. Available: [https://www.laraveltip.com/que-son-los-routes-en-laravel/#:~:text=Tipos%20de%20Rutas%20en%20Laravel,-El%20sistema%20de&text=Solo%20basta%20con%20utilizar%20la,options\(%24uri%2C%20%24callback\)%3B](https://www.laraveltip.com/que-son-los-routes-en-laravel/#:~:text=Tipos%20de%20Rutas%20en%20Laravel,-El%20sistema%20de&text=Solo%20basta%20con%20utilizar%20la,options(%24uri%2C%20%24callback)%3B)
- [47] Laravel.Views. [Online]. Available: <https://laravel.com/docs/9.x/views#creating-the-first-available-view>
- [48] Mysql. Limits on Table Size.2022. [Online]. Available: <https://dev.mysql.com/doc/mysql-reslimits-excerpt/8.0/en/table-size-limit.html>
- [49] Ubuntu. How large can a MySQL database become?.[Online]. Available: <http://www.iasptk.com/large-can-mysql-database-become/>
- [50] Pooja Sharma. Las 10 principales ventajas de los servicios de desarrollo de Laravel para empresas. [Online]. Available: <https://cynoteck.com/es/blog-post/top-10-advantages-of-laravel-development-services-for-enterprises/#:~:text=La%20ventaja%20de%20usar%20Laravel,web%20est%C3%A1%20seguro%20y%20protegido.>