



Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA

**GRADO EN INGENIERÍA INFORMÁTICA
MENCIÓN EN TECNOLOGÍAS DE LA INFORMACIÓN**

**Sistema de Medición y Control de Consumo
y Producción de una Instalación Eléctrica
Fotovoltaica Doméstica**

Alumno: Sergio Muñumer Blázquez

**Tutores: Dr. César Llamas Bello
D.^a Elena Castrillo Romón (IES La Merced)**



Para el que lo intentó.

Agradecimientos

La realización de este trabajo de fin de grado no habría sido posible sin la colaboración del equipo técnico de *Energética Coop* y de Elena Castrillo, profesora de electrónica en el *IES La Merced*, a todos ellos les doy las gracias por el conocimiento y el tiempo dedicado al proyecto.

A mayores me gustaría mencionar a mi tutor, Don César Llamas Bello, por su disponibilidad y comprensión para con el alumno, pues, la realización del proyecto ha sido paralela al estado de alarma nacional, lo que ha supuesto una dificultad añadida para todos.

Resumen

El presente trabajo de fin de grado ha sido realizado con el objetivo de gestionar el autoconsumo de una vivienda autosostenible. Dicho proyecto a su vez, se puede dividir en dos subproyectos.

Por un lado, se ha diseñado y prototipado un dispositivo para la toma de mediciones, capturando intensidad y tensión de la corriente eléctrica, cuya utilidad será doblemente aplicada: para medir el consumo de una carga eléctrica, (Véase un electrodoméstico o cualquier aparato eléctrico) y para medir la producción de una instalación solar fotovoltaica.

Y por otro lado, se ha desarrollado un sistema informático para la gestión y monitorización de las medidas tomadas por el dispositivo anteriormente mencionado. Dicho sistema cuenta con una API REST como back-end para la gestión de la lógica de negocio y control de acceso a la base de datos, y un servidor de front-end para la visualización de cara al usuario.

Abstract

This final degree project has been carried out with the aim of managing the self-consumption of a self-sustaining home. This project, in turn, can be divided into two subprojects.

On the one hand, a device has been designed and prototyped for taking measurements, capturing intensity and voltage of the electric current, the utility of which will be doubly applied: to measure the consumption of an electric charge, (see an electrical appliance or any electrical appliance) and to measure the production of a photovoltaic solar installation.

On the other hand, a computer system has been developed for the management and monitoring of the measures taken by the aforementioned device. Said system has a REST API as a back-end for business logic management and access control to the database, and a front-end server for user-facing visualization.

Índice general

Agradecimientos	III
Resumen	V
Abstract	VII
Lista de figuras	XVII
Lista de tablas	XXI
1. Introducción	1
1.1. Introducción	1
1.2. Motivación	2
1.3. Objetivos y etapas	3
1.4. Recursos utilizados	4
1.5. Marco socio-económico	5
1.5.1. Presupuesto	5
1.5.2. Impacto social	7
1.6. Estructura del documento	7
	IX

2. Metodología	9
2.1. Proceso de desarrollo	9
2.2. Lenguajes de programación	10
2.2.1. C	10
2.2.2. JavaScript	10
2.3. Conceptos y Herramientas	10
2.3.1. Visual Studio Code	10
2.3.2. Arduino IDE	11
2.3.3. Node.js	11
2.3.4. React	12
2.3.5. MongoDB	12
2.3.6. PuTTY	12
2.3.7. Postman	12
2.3.8. API REST	13
2.3.9. Daemon	13
2.3.10. Autoconsumo	13
2.4. Planificación	15
2.4.1. Asignación de recursos	16
3. Análisis	19
3.1. Identificación de usuarios	19
3.1.1. Administradores	20
3.1.2. Usuario estándar	20
3.2. Requisitos funcionales	20
3.3. Requisitos no funcionales	20

3.3.1. Accesibilidad	20
3.3.2. Seguridad	21
3.3.3. Usabilidad	21
3.3.4. Disponibilidad	21
3.3.5. Requisitos de información	21
3.3.6. Requisitos de restricción de información	21
3.4. Casos de uso	22
3.4.1. Diagramas de casos de uso	22
3.4.2. Especificación de casos de uso	24
3.5. Modelo de dominio	47
3.6. Diagramas de secuencia	47
3.6.1. Acceso al sistema	47
3.6.2. Crear Entidad	48
3.6.3. Eliminar Entidad	48
3.6.4. Modificar Entidad	48
3.6.5. Obtener Entidad	49
3.6.6. Registrar medición de producción	49
3.6.7. Registrar medición de consumo	49
3.7. Diccionario de datos	53
3.7.1. Usuario	53
3.7.2. Administrador	54
3.7.3. Inversor	54
3.7.4. Panel solar	54
3.7.5. Instalación	55
3.7.6. Medición	55

3.7.7. Enumeraciones	55
4. Diseño	57
4.1. Arquitectura	57
4.1.1. Arquitectura física	57
4.1.2. Arquitectura lógica	58
4.2. Interfaz gráfica	60
5. Implementación	65
5.1. API REST	65
5.1.1. Estructura	65
5.1.2. Controllers	68
5.1.3. Mongo	68
5.1.4. Routes	69
5.2. Front-End	70
5.2.1. Estructura	70
5.2.2. Componentes	71
5.2.3. Rutas	72
5.2.4. Funciones	73
5.3. Raspberry Pi Zero W: Daemon	73
5.4. Protocolo de comunicación: Arduino Nano - Raspberry Pi Zero W	74
5.5. Arduino Nano	75
6. Diseño Hardware	77
6.1. Introducción	77
6.2. Arduino	77

6.2.1. ¿Qué es?	77
6.2.2. Comparando dispositivos	78
6.2.3. Arduino Nano	78
6.3. Raspberry Pi	79
6.3.1. ¿Qué es?	79
6.3.2. Comparando dispositivos	79
6.3.3. Raspberry Pi Zero W	79
6.4. Otros componentes	80
6.4.1. NRF24L01	80
6.4.2. ADS1115	81
6.4.3. ACS712	81
6.4.4. FZ0430	81
6.4.5. Transformador	81
6.5. Circuitos	82
6.6. Consideraciones	82
7. Pruebas	83
7.1. Pruebas de caja blanca	83
7.2. Pruebas de caja negra	83
7.3. Consideraciones	92
8. Conclusiones	93
8.1. Conclusiones	93
8.2. Líneas futuras	94

A. Guía de instalación	95
A.1. Arduinos	95
A.2. Raspberry Pi	96
A.2.1. Instalación del sistema operativo	96
A.2.2. Configuración de la red Wifi	96
A.2.3. Instalación del servicio: RegistroDatos	97
A.3. API REST: Node.JS	98
A.4. Front-end: React	98
B. Manual de usuario	99
B.1. Administrador	99
B.1.1. Vinculación de instalaciones	102
B.1.2. Añadir entidades	105
B.1.3. Cerrar sesión	105
B.2. Usuario	107
C. Guía de montaje	113
C.1. Esquemas	113
C.1.1. Arduino Nano	113
C.1.2. Raspberry Pi Zero W	114
C.1.3. NRF24L01	115
C.1.4. ACS712	115
C.1.5. ADS1115	116
C.1.6. FZ0430	116
C.2. Módulo Producción/Consumo	117
C.2.1. Medida tensión	117

ÍNDICE GENERAL

C.2.2. Medida intensidad	118
C.3. Circuitos	119
C.3.1. Módulo producción	119
C.3.2. Módulo consumo	120
Bibliografía	121

Índice de figuras

2.1. Diagrama de Gantt simplificado	16
3.1. Diagrama casos de uso: Administrador	23
3.2. Diagrama casos de uso: Usuario estándar	24
3.3. Modelo de dominio	47
3.4. Diagrama de secuencia: Acceso al sistema	48
3.5. Diagrama de secuencia: Crear entidades	49
3.6. Diagrama de secuencia: Eliminar entidades	50
3.7. Diagrama de secuencia: Modificar entidades	51
3.8. Diagrama de secuencia: Obtener entidades	52
3.9. Diagrama de secuencia: Obtener medida producción	52
3.10. Diagrama de secuencia: Obtener medida consumo	52
4.1. Diagrama de despliegue	59
4.2. Interfaz usuario estándar: Tiempo real	60
4.3. Interfaz usuario estándar: Históricos	61
4.4. Interfaz usuario estándar: Perfil	62
4.5. Interfaz usuario administrador: Ver datos de entidad	63
4.6. Interfaz usuario administrador: Añadir entidad	64

5.1. Estructura de la API.	67
5.2. Controlllers.	68
5.3. Mongo.	68
5.4. Routes.	69
5.5. Routes: index.js	69
5.6. Estructura aplicación React	70
5.7. React: Componentes utilizados	72
5.8. React: Rutas	73
5.9. React: Funciones	74
B.1. Vista: Administradores.	99
B.2. Vista: Barra de búsqueda.	100
B.3. Vista: Selección de elementos.	100
B.4. Vista: Diálogo de confirmación.	101
B.5. Vista: Botón modificar.	101
B.6. Vista: Formulario de modificación.	102
B.7. Vista: Usuarios.	103
B.8. Vista: Instalación de usuario.	103
B.9. Vista: Asignación de instalaciones.	104
B.10. Vista: Nuevo inversor.	105
B.11. Vista: Añadir entidad.	106
B.12. Vista: Cerrar sesión.	106
B.13. Vista: Usuario con conexión al dispositivo.	107
B.14. Vista: Usuario sin conexión al dispositivo.	108
B.15. Vista: Histórico diario.	109

B.16. Vista: Histórico mensual.	110
B.17. Vista: Histórico anual.	111
B.18. Vista: Menú desplegable superior.	112
B.19. Vista: Mi perfil.	112
C.1. Imagen recuperada de: https://bigdanzblog.wordpress.com/2015/01/30/cant-get-i2c-to-work-on-an-arduino-nano-pinout-diagrams	113
C.2. Imagen recuperada de: https://cdn.sparkfun.com/assets/learn_tutorials/6/7/6/PiZero_1.pdf	114
C.3. Imagen recuperada de: https://saber.patagoniatec.com/2014/07/nrf24l01-rfx2401-nrf24l01-pa-lna-arduino-argentina-ptec	115
C.4. Imagen recuperada de: https://www.luisllamas.es/arduino-intensidad-consumo-electrico-ac712	115
C.5. Imagen recuperada de: https://www.industrialshields.com/es_ES/blog/nuestro-blog-1/post/ads1115-16-bits-adc-board-143	116
C.6. Imagen recuperada de: https://www.luisllamas.es/arduino-sensor-corriente-sct-013	116
C.7. Circuito medidor de tensión	117
C.8. Circuito medidor de intensidad	118
C.9. Circuito módulo producción	119
C.10. Circuito módulo consumo	120

Índice de cuadros

1.1. Costes software	5
1.2. Costes hardware	6
1.3. Costes indirectos	6
1.4. Coste total	6
3.1. Requisitos funcionales: Parte 1	26
3.2. Requisitos funcionales: Parte 2	27
3.3. Requisitos no funcionales	28
3.4. Requisitos no funcionales: Accesibilidad	28
3.5. Requisitos no funcionales: Seguridad	28
3.6. Requisitos no funcionales: Usabilidad	29
3.7. Requisitos no funcionales: Disponibilidad	29
3.8. Requisitos de información	29
3.9. Requisitos de restricción de información	30
3.10. Administrador: CU-01: Iniciar sesión	30
3.11. Administrador: CU-02: Crear administrador	31
3.12. Administrador: CU-03: Eliminar administrador	31
3.13. Administrador: CU-04: Modificar administrador	32

3.14. Administrador: CU-05: Ver administradores	32
3.15. Administrador: CU-06: Crear usuario	33
3.16. Administrador: CU-07: Eliminar usuario	33
3.17. Administrador: CU-08: Modificar usuario	34
3.18. Administrador: CU-09: Ver usuarios	34
3.19. Administrador: CU-10: Crear inversor	35
3.20. Administrador: CU-11: Eliminar inversor	35
3.21. Administrador: CU-12: Modificar inversor	36
3.22. Administrador: CU-13: Ver inversores	36
3.23. Administrador: CU-14: Crear panel solar	37
3.24. Administrador: CU-15: Eliminar panel solar	37
3.25. Administrador: CU-16: Modificar panel solar	38
3.26. Administrador: CU-17: Ver paneles solares	38
3.27. Administrador: CU-18: Crear instalación	39
3.28. Administrador: CU-19: Eliminar instalación	39
3.29. Administrador: CU-20: Modificar instalación	40
3.30. Administrador: CU-21: Ver instalaciones	40
3.31. Administrador: CU-22: Cerrar sesión	41
3.32. Usuario: CU-23: Iniciar sesión	41
3.33. Usuario: CU-24: Ver perfil	42
3.34. Usuario: CU-25: Modificar contraseña	43
3.35. Usuario: CU-26: Ver instalación	43
3.36. Usuario: CU-27: Ver gráfica en tiempo real	44
3.37. Usuario: CU-28: Ver gráfica por días	44
3.38. Usuario: CU-29: Ver gráfica por mes	45

3.39. Usuario: CU-30: Ver gráfica por año	46
3.40. Usuario: CU-31: Cerrar sesión	46
3.41. Diccionario de datos: Usuario	53
3.42. Diccionario de datos: Administrador	54
3.43. Diccionario de datos: Inversor	54
3.44. Diccionario de datos: Panel solar	54
3.45. Diccionario de datos: Instalación	55
3.46. Diccionario de datos: Medición	55
5.1. Protocolo de comunicación	74
5.2. Códigos	75
6.1. Comparativa Arduinos	78
6.2. Comparativa Raspberries	79
7.1. Prueba de caja negra: Iniciar sesión	84
7.2. Prueba de caja negra: Crear administrador	84
7.3. Prueba de caja negra: Crear usuario	84
7.4. Prueba de caja negra: Crear inversor	85
7.5. Prueba de caja negra: Crear panel solar	85
7.6. Prueba de caja negra: Crear instalación	85
7.7. Prueba de caja negra: Modificar administrador	86
7.8. Prueba de caja negra: Modificar usuario	86
7.9. Prueba de caja negra: Modificar inversor	86
7.10. Prueba de caja negra: Modificar panel solar	87
7.11. Prueba de caja negra: Modificar instalación	87

7.12. Prueba de caja negra: Eliminar administrador	87
7.13. Prueba de caja negra: Eliminar usuario	88
7.14. Prueba de caja negra: Eliminar inversor	88
7.15. Prueba de caja negra: Eliminar panel	88
7.16. Prueba de caja negra: Eliminar instalación	89
7.17. Prueba de caja negra: Cerrar sesión	89
7.18. Prueba de caja negra: Ver perfil	89
7.19. Prueba de caja negra: Modificar contraseña	90
7.20. Prueba de caja negra: Ver histórico diario	90
7.21. Prueba de caja negra: Ver histórico mensual	90
7.22. Prueba de caja negra: Ver histórico anual	91

Capítulo 1

Introducción

1.1. Introducción

En la actualidad, España es uno de los países de la Union Europea que más paga en su factura de la luz. A su vez, somos una potencia en energía solar a tener en cuenta. Tras la abolición del impuesto al sol, tenemos más accesible que nunca la posibilidad de producir nuestra propia energía y de manera gratuita. Mediante la instalación de paneles solares en nuestra vivienda o negocio podremos reducir nuestra factura de la luz entorno al 70 %. Este tipo de instalaciones tiene una vida útil de más de 30 años, y entorno a 7-10 años de tiempo de amortización, por lo que sino tienes una ya, estas perdiendo dinero.

En el mercado actual existen alternativas de pago similares al ofrecido por este proyecto. Dichas alternativas solo capturan datos del consumo de una red, ofreciendo información estadística de dicho consumo, dejando de lado u olvidando la posibilidad de producir tu propia energía.

El presente proyecto pretende servir de guía en el diseño y elaboración de un prototipo para medir el consumo de un hogar, así como la producción de una instalación fotovoltaica doméstica; a mayores de un sistema informático para monitorizar y gestionar los datos obtenidos. En este documento se describirán todas las fases necesarias para la elaboración del sistema, desde la selección de componentes hardware, hasta el desarrollo de un front-end para la visualización y gestión de los datos obtenidos a través de los dispositivos.

1.2. Motivación

La realización del presente trabajo de fin de grado supone la adquisición de unos conocimientos variados, cuyo ámbito supera el ámbito informático.

En el presente proyecto se han aplicado conocimientos electrónicos, físicos y por supuesto informáticos, para realizar tareas como la construcción y montaje de un circuito capaz de medir el consumo/producción de cargas eléctricas, o como el desarrollo de un sistema informático para monitorizar dichas mediciones.

La consecución de ciertas tareas, como el desarrollo de un back-end basado en una *API REST*, a supuesto al alumno una necesidad de ampliación de los conocimientos adquiridos durante el *Grado en Ingeniería Informática*. A mayores se ha tenido que trabajar con placas *Arduino* y sensores, lo que supone una gran novedad e incentivo a nivel personal.

Podemos concluir, que el presente proyecto ha servido al alumno como un gran desafío, en el que se han puesto a prueba sus capacidades y ha servido como refuerzo tanto a nivel personal como profesional.

1.3. Objetivos y etapas

El objetivo principal de este trabajo de fin de grado es el diseño y desarrollo de un sistema de monitorización del consumo de aparatos eléctricos domésticos y de la producción de una instalación solar fotovoltaica.

Para la consecución de dicho objetivo, podemos establecer diferentes etapas en el desarrollo del sistema.

Primera etapa

Se debe realizar el diseño y montaje del circuito para la toma de mediciones. Dicho circuito esta formado por sensores y dos placas arduino. Es el encargado de emitir datos hacia la Raspberry.

Taras principales:

- Estudio y análisis del problema a resolver.
- Estudio de mercado: tecnologías.
- Estudio de mercado: Arduino, sensores...
- Diseño arquitectura hardware.
- Montaje del circuito.
- Programación arduinos.

Segunda etapa

Se debe desarrollar un demonio en Linux para la recepción de datos emitidos por el circuito de la etapa anterior. A su vez, dicho programa enviará los datos a la BBDD a través de la API.

- Configuración: Raspberry.
- Diseño del demonio en Linux.
- Pruebas de recepcón de datos desde el arduino.

Tercera etapa

Se debe desarrollar e implementar una API REST como servidor que reciba las peticiones enviadas por el demonio y almacene la información en la base de datos.

Tareas principales:

- Estudio: API REST.
- Realización de cursos online: Node.JS y Express.
- Desarrollo de una aplicación servidor.
- Pruebas de recepción de datos desde Raspberry.

Cuarta etapa

Se debe desarrollar un front-end para la visualización de datos a través de un navegador.

Tareas principales:

- Estudio: React.
- Realización de cursos online: React.
- Desarrollo del servidor de front-end con React.
- Vinculación: React-Node-JS, Módulo Cors.
- Pruebas.

1.4. Recursos utilizados

Para el desarrollo del presente trabajo de fin de grado se ha dispuesto con los siguientes recursos:

- Ordenador de sobremesa.
- Raspberry Pi Zero W.
- Arduino Nano: procesador ATmega328 (2 unidades).

- Transceptor inalámbrico NRF24L01 (2 unidades).
- Conversor analógico-digital ADS1115 (2 unidades).
- Divisor de tensión FZ0430 (2 unidades).
- Sensor de corriente ACS712 (2 unidades).
- Transformador (2 unidades).
- Pack cables dupont.
- Cable USB-TTL.
- Cables USB (2 unidades).
- Cargador USB (2 unidades).
- Regleta (2 unidades).

1.5. Marco socio-económico

1.5.1. Presupuesto

A continuación se muestra el presupuesto de desarrollo del proyecto clasificado por tipo de gasto.

Costes software

Descripción	Precio (€)
Visual Studio Code	0€
Arduino IDE	0€
PuTTY	0€
MongoDB	0€
Postman	0€
Total:	0€

Cuadro 1.1: Costes software

Costes hardware

Descripción	Precio (€)	Cantidad
Raspberry Pi Zero W.	10€	1
Arduino Nano	4€	2
ADS1115	6€	1
FZ0430	2€	1
ACS712	3€	1
STC-013	6€	1
Pack cables dupont	5€	1
Cables USB	4€	2
Cargador USB	16€	2
Transformador€	4	1
NRF24L01	2€	2
Total:	68€	

Cuadro 1.2: Costes hardware

Costes indirectos

Descripción	Precio (€)	Cantidad	Precio total
Ordenador personal	1000	1	1000€
Gastos de luz	10€/mes	4	40€
Gastos de internet	30€/mes	4	120€
Total:			1160€

Cuadro 1.3: Costes indirectos

Coste total

Descripción	Precio (€)
Costes hardware	48€
Costes software	0€
Costes indirectos	1160€
Total:	1208€

Cuadro 1.4: Coste total

1.5.2. Impacto social

Gracias al desarrollo de las tecnologías a día de hoy cualquier hogar puede transformarse en una vivienda inteligente, capaz de gestionar el ahorro energético, mejorando el confort de los inquilinos.

En esta dirección, podemos afirmar que el presente proyecto, da un paso adelante mejorando la calidad de vida del usuario, ofreciéndole una serie de beneficios.

Sociales

El proyecto ofrece la posibilidad de mejorar la gestión energética de los equipos y dispositivos electrónicos, permitiendo al usuario localizar las franjas horarias y los dispositivos de mayor gasto. A mayores, el usuario podrá visualizar su instalación en tiempo real en cualquier momento.

Económicos

A mayores el usuario obtendrá información detallada sobre la producción de sus paneles solares, como consecuencia, podrá gestionar sus equipos para ahorrar cierta cantidad de dinero en la factura de la luz.

Medioambientales

La vinculación del proyecto hacia las energías renovables es un echo. Si el usuario es capaz de gestionar su consumo en las horas de producción, estará disminuyendo significativamente el gasto de energía procedente de energías no renovables.

1.6. Estructura del documento

Esta memoria ha sido estructurada en capítulos en función de su carácter, siguiendo las fases establecidas según la *Ingeniería del software*.

En el primer capítulo del documento se ha presentado el proyecto, sus objetivos, motivaciones y los materiales que se requieren para su consecución.

1.6. ESTRUCTURA DEL DOCUMENTO

En el *Capítulo II* ahondaremos en la metodología de trabajo seguida, las herramientas y conceptos necesarios, los lenguajes de programación utilizados y la arquitectura del proyecto.

El *Capítulo III* ha sido dedicado a toda la parte referente al *Análisis software*. Incluyendo análisis de requisitos, diferentes tipos de diagramas, diccionarios de datos...

En el *Capítulo IV* abordaremos el diseño del sistema, su arquitectura, física y lógica y el diseño de interfaces.

En el *Capítulo V* se detalla como se ha implementado el sistema tras el análisis y diseño.

En el *Capítulo VI* se expondrán los componentes seleccionados para el diseño hardware del dispositivo.

En el *Capítulo VII* podremos analizar las pruebas realizadas sobre el sistema.

Para finalizar, en el *Capítulo VIII* se encuentran las conclusiones y líneas futuras del presente proyecto.

Al final del documento se incluyen varios apéndices con guías para la instalación, montaje y uso del sistema.

Capítulo 2

Metodología

2.1. Proceso de desarrollo

Para el desarrollo de este proyecto de fin de grado se ha echo uso de uno de las metodologías ágiles mas conocidas y practicadas: *Extreme programming* [20]. Teniendo en cuenta la duración del plazo de realización del proyecto, es la metodología que mejor se ajusta a nuestra situación.

El objetivo de este método de trabajo es el desarrollo y gestión del proyecto de manera eficaz, flexible y controlada, teniendo como puntos claves la reutilización del código desarrollado y la realimentación de las distintas partes implicadas.

El inicio del proyecto se ve marcado por el desarrollo de una planificación temporal a plazos, en base a las exigencias del cliente y a la dificultad de los casos de uso. Esta planificación no se ha seguido de forma precisa, pero sí de manera orientativa.

Una vez entregada la hoja de planificación al cliente, se comenzó desarrollando iteraciones. Cada dos semanas se presentó el desarrollo al cliente, de esta manera los requisitos del sistema se ajustan de mejor manera a los requerimientos del cliente.

Antes de cada entrega, se ha realizado una pequeña fase de pruebas de *caja blanca* para comprobar que nos ceñimos a los requisitos.

2.2. Lenguajes de programación

2.2.1. C

El lenguaje *C* es un lenguaje de programación de propósito general originalmente desarrollado por *Dennis Ritchie* entre 1969 y 1972 en los *Laboratorios Bell*, como evolución de su anterior lenguaje *B*. Débilmente tipificado, tipos de datos estáticos, ofrece un control a muy bajo nivel.

Es un lenguaje orientado a la implementación de sistemas operativos, concretamente *Unix*. Apreciado por su eficiencia, es el lenguaje mas popular para crear software de sistema, aunque también se utiliza en otros ámbitos.

La utilización de este lenguaje ha sido clave para el desarrollo de un *Daemon* de *Linux*, encargado de recibir y transmitir al servidor, las medidas realizadas por los dispositivos de medición.

A su vez, ha sido necesario para el desarrollo de los *Sketches* de Arduino

2.2.2. JavaScript

JavaScript es un lenguaje de programación interpretado, cuyo verdadero nombre es ECMAScript. Se define como orientado a objetos, basado en prototipos, bajo el paradigma imperativo, débilmente tipado y dinámico.

Típicamente es un lenguaje nativo del lado del cliente, pero con la aparición de nuevos *frameworks* su uso se ha extendido al lado del servidor.

Utilizaremos *JavaScript* para llevar a cabo el desarrollo del front-end y de la API del sistema.

2.3. Conceptos y Herramientas

2.3.1. Visual Studio Code

El editor de código es la principal herramienta para un desarrollador. Para este proyecto se ha elegido Visual Studio Code. Creado por *Microsoft* ha medidos del 2015, se ha consolidado como una de las mejores herramientas para programar *JavaScript*.

Esta herramienta ha sido utilizada para el desarrollo de los códigos del *Backend/FrontEnd* en *JavaScript*. El desarrollo de scripts de *Linux* en *C* también ha sido llevado a cabo con VSC.

2.3.2. Arduino IDE

El entorno de desarrollo integrado de Arduino es una aplicación multiplataforma escrita en *Java*. El código fuente para el IDE se publica bajo la *Licencia Pública General de GNU*.

Ha sido utilizado para el desarrollo de los *Sketches* de los dispositivos Arduino, su compilación, y carga.

2.3.3. Node.js

Node.js es un entorno que trabaja en tiempo de ejecución, de código abierto, multiplataforma, nos permite a los desarrolladores, crear herramientas en el lado del servidor con *JavaScript*.

La piedra angular de Node es su gestor de paquetes NPM (Node Packet Manager) permitiendo añadir librerías/funciones sin preocuparnos por las dependencias.

Express es uno de los módulos mas populares de Node. Es la librería subyacente para un gran número de otros frameworks populares de Node. Es el encargado de convertir una aplicación *JavaScript* en una aplicación del lado del servidor. Proporciona mecanismos para:

- Escritura de manejadores de peticiones con diferentes verbos HTTP en diferentes caminos URL (rutas).
- Integración con motores de renderización de "vistas" para generar respuestas mediante la introducción de datos en plantillas.
- Establecer ajustes de aplicaciones web como qué puerto usar para conectar, y la localización de las plantillas que se utilizan para renderizar la respuesta.
- Añadir procesamiento de peticiones "middleware" adicional en cualquier punto dentro de la tubería de manejo de la petición.

Utilizaremos estas herramientas para el desarrollo de una API en *backend* que maneje la persistencia y la lógica de la aplicación.

2.3.4. React

React es una librería de *JavaScript* focalizada en el desarrollo de interfaces de usuario. Será nuestro principal aliado para realizar un *frontend* SPA (Single Page Application), válida para dispositivos móviles.

Esta librería fue desarrollada por *Facebook*, es de software libre y cada día suma más adeptos a su comunidad. Sus ventajas radican en el paradigma de la programación reactiva, con ello podremos desarrollar en *JavaScript* aplicaciones de manera más sencilla y ordenada que utilizando otras herramientas populares como *jQuery*.

2.3.5. MongoDB

MongoDB es un sistema de base de datos NoSQL, orientado a documentos y de código abierto.

El funcionamiento difiere del típico de las bases de datos relacionales. *MongoDB* guarda estructuras de datos BSON (Similar a JSON) con un esquema dinámico, haciendo que la integración de los datos en ciertas aplicaciones sea más sencilla y eficiente. El código fuente está disponible para los sistemas operativos *Windows*, *GNU/Linux* y *Solaris*.

Para este proyecto se ha utilizado un clúster en *MongoDB Atlas*. *MongoDB Atlas* es un gestor de bases de datos en la nube, que nos ofrece clústers de almacenamiento gratuitos hasta 512 MB.

2.3.6. PuTTY

PuTTY es un cliente SSH, Telnet, rlogin y TCP Raw con licencia libre. Utilizaremos esta herramienta para el control remoto de la *Raspberry Pi Zero W*.

2.3.7. Postman

Postman es una herramienta que se utiliza, sobre todo, para el testing de API REST, aunque también admite otras funcionalidades que se salen de lo que engloba el testing de este tipo de sistemas.

Gracias a esta herramienta, hemos podido testear y depurar la API del sistema.

2.3.8. API REST

Denominamos API a un conjunto de reglas y especificaciones que las aplicaciones pueden seguir para comunicarse entre ellas. De manera algo más coloquial, una API es un "puente" entre dos sistemas software distintos.

Una API se ve definida por las siguientes características:

- Protocolo cliente-servidor sin estado.
- Cuatro operaciones: POST, GET, PUT y DELETE.
- Objetos manipulados a través de URI's.
- Interfaz uniforme.
- Basado en capas.
- Uso de hipermedios.

2.3.9. Daemon

Utilizamos el término *daemon* haciendo referencia a un tipo especial de proceso informático no interactivo, es decir, se ejecuta en segundo plano en vez de ser controlado por la acción de un usuario [11].

Esté es el término usado en sistemas *POSIX*, en *Windows* reciben el nombre de "Servicios", mientras que en *MS-DOS* se les denominaba "Programa residente".

2.3.10. Autoconsumo

El objetivo principal del proyecto es desarrollar un sistema para gestionar el autoconsumo de una vivienda, pero: ¿Qué entendemos por *autoconsumo*?

Si buscamos la definición encontraremos algo similar a lo siguiente: *Consumo por parte de los productores de bienes o servicios de los productos que ellos mismos producen.*

Dicha definición traducida al ámbito que nos concierne, se entiende como la cantidad de energía consumida procedente de nuestros propios paneles solares.

Para calcular el ratio de autoconsumo de un hogar, necesitamos dos datos iniciales: cantidad de energía demandada por la vivienda y cantidad de energía producida por la instalación

2.3. CONCEPTOS Y HERRAMIENTAS

solar de la vivienda. Nuestro dispositivo de medición será el encargado de proporcionarnos estos datos, para ello cuenta con dos módulos: Consumo y Producción.

El módulo consumo se hallará tomando medidas tras el cuadro de luz de la vivienda. Si dichas medidas son positivas entenderemos que estamos importando energía de la red eléctrica. Si las medidas son negativas estaremos exportando energía hacia la red.

El módulo producción se hallará tras el inversor de la instalación solar, tomando medidas solamente positivas sobre la potencia generada.

Por lo tanto, para calcular el ratio de autoconsumo (2.1):

$$\text{RatioAutoconsumo} = \frac{\text{Producción} - \text{Exportada}}{\text{Producción} + \text{Importada} - \text{Exportada}} \quad (2.1)$$

2.4. Planificación

Para llevar a cabo la planificación del proyecto se han tenido en cuenta los objetivos y requisitos del sistema, así como los diferentes nodos físicos que intervienen en la solución al problema.

La planificación se puede dividir en 4 etapas.

Etapa 1: Dispositivo de medición

En la primera etapa del proyecto se desarrolló la arquitectura del dispositivo físico de medición. Se analizó el problema a resolver, y se han realizado diversos estudios de mercado para concluir que elementos son necesarios en nuestra arquitectura. Una vez hallada la solución, se procedió a la construcción del dispositivo para su posterior puesta a prueba.

Con el dispositivo funcionando, tomando medidas y enviándolas por el puerto serie, damos por finalizada la primera etapa del proyecto.

Duración de la etapa: 80 horas aprox.

Etapa 2: Daemon de Linux

Tenemos un dispositivo activo tomando medidas y enviándolas por el puerto serie, ahora nos falta un receptor para dicha información.

La segunda etapa consiste en el desarrollo de un *daemon* en *Linux* para recepcionar dichos datos y posteriormente enviarlos a un tercer nodo. Dicha etapa, contiene una sub-etapa final de pruebas; En ella se analizaron los diferentes casos para ambos tipos de medición (Producción y consumo) y se comprobó el correcto funcionamiento con la herramienta de *Linux*: GDB (*Gnu Project Debugger*).

Duración de la etapa: 80 horas aprox.

Etapa 3: API REST y Persistencia de los datos

Una vez que tenemos desarrollado el *daemon*, necesitamos un nodo servidor donde recibir todos los datos de mediciones para su posterior almacenamiento.

En esta etapa se llevó a cabo el desarrollo de una *API REST*. Dicha API proporciona diferentes URL's, para diferentes datos, que servirán como punto de recogida de información.

2.4. PLANIFICACIÓN

El script de *Linux* enviará la información que recibe del dispositivo de medición a la API a través de estas URL's. Una vez recibida la información en la API, está se almacena en un *cluster* de *MongoDB Atlas*.

Duración de la etapa: 90 horas aprox.

Etapa 4: Front-End

Una vez que tenemos el sistema de recolección y mantenimiento de la persistencia de los datos, a nuestro proyecto solo le falta un modo de visualizar dicho contenido.

La última etapa y también la más dura, consistió en el desarrollo de un front-end con *React*. Dicho *front* debe ser flexible en función del tipo de usuario que se identifique en la aplicación.

Duración de la etapa: 120 horas aprox.

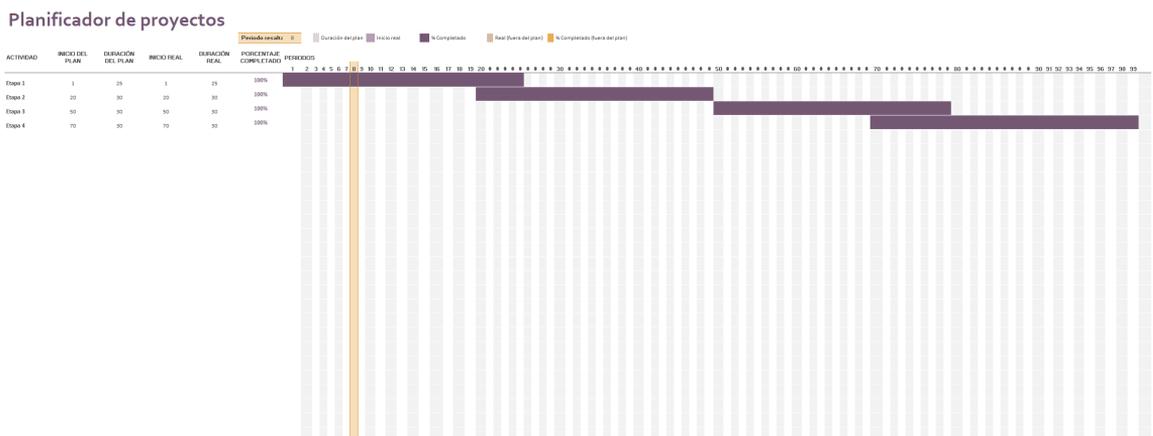


Figura 2.1: Diagrama de Gantt simplificado

2.4.1. Asignación de recursos

A continuación se muestran las asignaciones de recursos necesarios por etapas.

Etapa 1: Dispositivo de medición

- Recursos materiales:

- Ordenador de sobremesa.
 - Raspberry Pi Zero W.
 - Arduino Nano: ATmega328 (2 uds).
 - Transceptor NRF24L01 (2 uds).
 - Conversor ADS1115 (2 uds).
 - Divisor de tensión FZ0430 (2 uds).
 - Sensor de corriente ACS712 (2 uds).
 - Transformador (2 uds).
 - Cables dupont.
 - Cables USB.
 - Cargador USB (2 uds).
 - Regleta (2 uds).
 - Cable USB-TTL.
- Recursos software:
 - Arduino IDE.
 - Visual Studio Code.
 - PuTTY.
 - Raspbian S.O.
 - Programa para quemar imagen en tarjeta SD, cualquiera es válido.
 - Recursos personales:
 - Programador.

Etapas 2: Daemon de Linux

- Recursos materiales:
 - Ordenador de sobremesa.
 - Raspberry Pi Zero W.
 - Arduino Nano: ATmega328.
 - Cables USB.
 - Cargador USB.
 - Cable USB-TTL.

- Recursos software:
 - Visual Studio Code.
 - PuTTY.
- Recursos personales:
 - Programador.

Etapa 3: API REST y Persistencia de los datos

- Recursos materiales:
 - Ordenador de sobremesa.
- Recursos software:
 - Visual Studio Code.
 - Postman.
 - MongoDB Atlas.
- Recursos personales:
 - Programador.

Etapa 4: Front-end

- Recursos materiales:
 - Ordenador de sobremesa.
- Recursos software:
 - Visual Studio Code.
 - Postman.
 - MongoDB Atlas.
- Recursos personales:
 - Programador.

Capítulo 3

Análisis

En el *Capítulo 3* vamos a proceder al análisis de los requisitos necesarios para alcanzar los objetivos marcados.

La confección de los requisitos se realizó de una manera gradual. En el inicio se marcaron unas líneas globales, y durante el desarrollo se han ido modificando y añadiendo en función de los plazos y/o problemas encontrados en cada cambio de etapa.

Al principio de cada etapa, se han celebrado reuniones con el equipo técnico de *Energética Coop* donde establecimos los objetivos para dicha etapa, siempre teniendo en cuenta las líneas globales del proyecto marcadas al inicio. Tras finalizar una etapa, se celebraba otra reunión para analizar el trabajo realizado, y si este satisface las necesidades del sistema.

Ha sido necesario recortar funcionalidad del sistema que el equipo de *Energética Coop* deseaba, debido a la gran ambición del mismo en un escaso plazo de tiempo.

3.1. Identificación de usuarios

La aplicación va a ser utilizada por dos tipos de usuarios:

- Administradores
- Usuario estándar

Vamos a detallar un poco las diferentes posibilidades que ofrece la aplicación en función del tipo de usuario.

3.1.1. Administradores

El papel del usuario *Administrador* en el sistema, es el de administrar los recursos del mismo. Este usuario podrá crear nuevos usuarios, tanto estándar como administrador, eliminarlos, modificarlos y acceder a su información. Podrá crear, eliminar, modificar y acceder información sobre paneles solares, inversores e instalaciones, así como, será el encargado de vincular usuarios con su instalación.

3.1.2. Usuario estándar

El usuario estándar obtendrá una vista de cliente en la aplicación. Podrá visualizar en tiempo real las mediciones captadas por el dispositivo de medición, tanto de consumo como de producción. También podrá visualizar toda la información pertinente a su instalación y a su vinculación con la cooperativa energética. A mayores podrá visualizar un histórico de las mediciones realizadas, clasificado por días, meses y/o años

3.2. Requisitos funcionales

Los requisitos funcionales definen la funcionalidad del mismo. A continuación, podemos ver listados los requisitos funcionales del sistema (Cuadro 3.1 y Cuadro 3.2).

3.3. Requisitos no funcionales

Entendemos como requisito no funcional, en el ámbito de la ingeniería del software, a un requisito que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos. Por lo tanto, son las restricciones o condiciones impuestas por el equipo de *Energética Coop* sobre el sistema.

En el cuadro 3.3 podemos ver los requisitos no funcionales.

3.3.1. Accesibilidad

Los requisitos no funcionales de usabilidad han sido validados utilizando los diferentes navegadores disponibles en el mercado. A mayores para el diseño responsive, se ha utilizado la herramienta de *Google Chrome* para redimensionar pantallas.

Para ver los requisitos no funcionales de accesibilidad diríjase al cuadro 3.4.

3.3.2. Seguridad

Para llevar a cabo los requisitos no funcionales de seguridad se han utilizado los módulos de Node.js: *bcrypt* y *jsonwebtoken*.

Para ver los requisitos no funcionales de seguridad diríjase al cuadro 3.5.

3.3.3. Usabilidad

Para validar los requisitos no funcionales de usabilidad se han realizado pruebas de caja blanca y pruebas de caja negra asegurando la respuesta del sistema en cada operación. A mayores, al final de este documento se incluye un “*Manual de usuario*” para la comprensión del sistema.

Para ver los requisitos no funcionales de usabilidad diríjase al cuadro 3.6.

3.3.4. Disponibilidad

Para validar los requisitos no funcionales de disponibilidad se ha expuesto el sistema a una semana de uso continuado.

Para ver los requisitos no funcionales de disponibilidad diríjase al cuadro 3.7.

3.3.5. Requisitos de información

Entendemos requisitos de información como aquellos que especifican las entidades, cuya información, ha de ser gestionada por el sistema.

Para ver los requisitos no funcionales de información diríjase al cuadro 3.8.

3.3.6. Requisitos de restricción de información

Los requisitos de restricción de información, como su nombre indica, nos van a especificar condiciones para el almacenamiento de información.

Para ver los requisitos no funcionales de restricción de información diríjase al cuadro 3.9.

3.4. Casos de uso

Entendemos como caso de uso la descripción de una acción o actividad. Así, el diagrama de casos de uso es entendido como la descripción de actividades a realizar para llevar a cabo algún proceso/acción.

En el contexto de ingeniería del software, un diagrama de caso de uso representa a un sistema o subsistema como un conjunto de interacciones que se desarrollarán entre casos de uso y entre estos y sus actores en respuesta a un evento que inicia un actor principal. A continuación se muestran los diagramas para los diferentes actores que contempla el sistema.

3.4.1. Diagramas de casos de uso

Administrador

El actor administrador tiene como rol en el sistema, ejecutar las 4 operaciones principales (CRUD) sobre las diferentes entidades:

- C. Create, creación de entidades.
- R. Read, lectura de datos de entidades.
- U. Update, modificación de datos de entidades.
- D. Delete, eliminación de entidades.

En la figura 3.1 se muestra el diagrama de casos de uso para el actor *Administrador*.

Usuario

El actor usuario estándar tiene como rol en el sistema, la posibilidad de monitorizar su dispositivo de medición en tiempo real. A mayores puede ver históricos clasificados por:

- Días.
- Meses.
- Años.

En la figura 3.3 se muestra el diagrama de casos de uso para el actor *usuario estándar*.

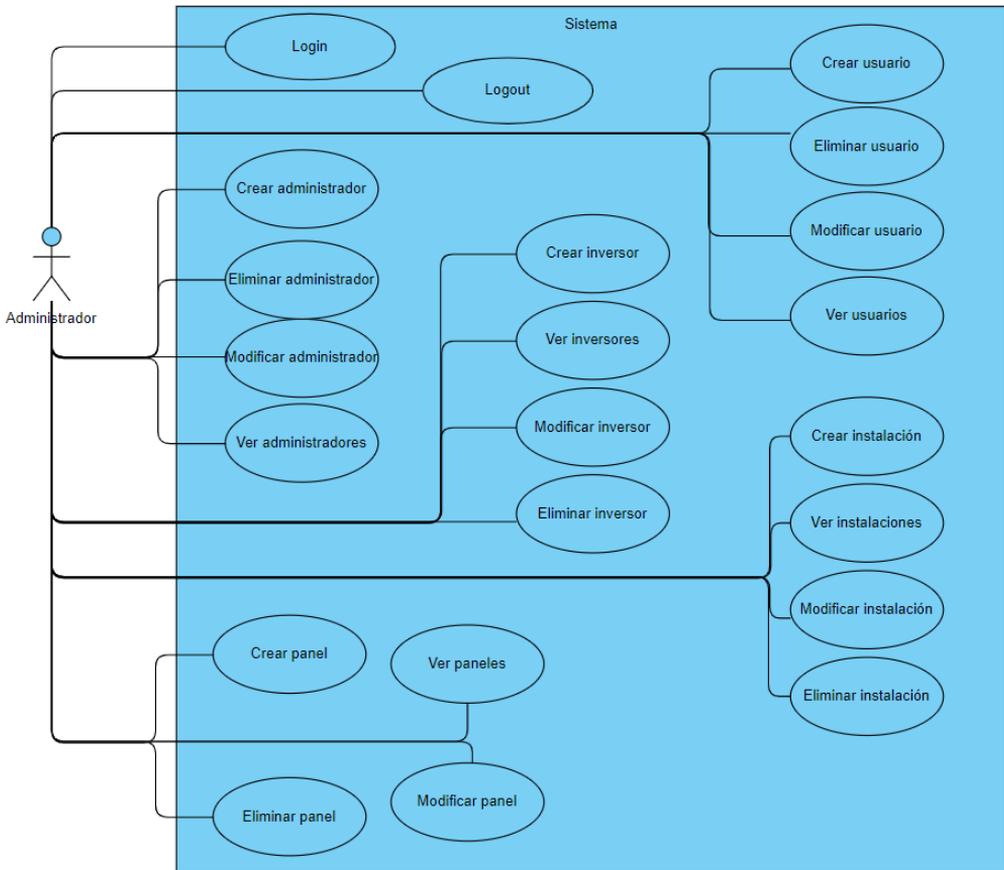


Figura 3.1: Diagrama casos de uso: Administrador

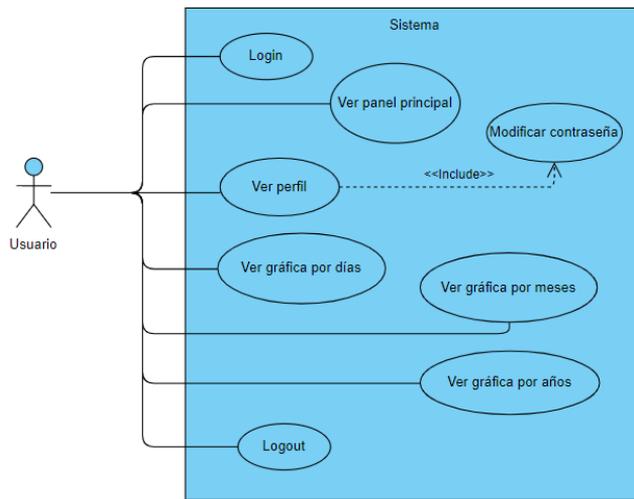


Figura 3.2: Diagrama casos de uso: Usuario estándar

3.4.2. Especificación de casos de uso

Administrador

CU-01	Iniciar sesión	Véase cuadro 3.10
CU-02	Crear administrador	Véase cuadro 3.11
CU-03	Eliminar administrador	Véase cuadro 3.12
CU-04	Modificar administrador	Véase cuadro 3.13
CU-05	Ver administradores	Véase cuadro 3.14
CU-06	Crear usuario	Véase cuadro 3.15
CU-07	Eliminar usuario	Véase cuadro 3.16
CU-08	Modificar usuario	Véase cuadro 3.17
CU-09	Ver usuarios	Véase cuadro 3.18
CU-10	Crear inversor	Véase cuadro 3.19
CU-11	Eliminar inversor	Véase cuadro 3.20
CU-12	Modificar inversor	Véase cuadro 3.21
CU-13	Ver inversores	Véase cuadro 3.22
CU-14	Crear panel solar	Véase cuadro 3.23
CU-15	Modificar panel solar	Véase cuadro 3.24
CU-16	Eliminar panel solar	Véase cuadro 3.25
CU-17	Ver paneles solares	Véase cuadro 3.26
CU-18	Crear instalación	Véase cuadro 3.27
CU-19	Eliminar instalación	Véase cuadro 3.28
CU-20	Modificar instalación	Véase cuadro 3.29
CU-21	Ver instalaciones	Véase cuadro 3.30
CU-22	Cerrar sesión	Véase cuadro 3.31

Usuario

CU-23	Iniciar sesión	Véase cuadro 3.32
CU-24	Ver perfil	Véase cuadro 3.33
CU-25	Modificar contraseña	Véase cuadro 3.34
CU-26	Ver instalación	Véase cuadro 3.35
CU-27	Ver gráfica en tiempo real	Véase cuadro 3.36
CU-28	Ver gráfica por días	Véase cuadro 3.37
CU-29	Ver gráfica por meses	Véase cuadro 3.38
CU-30	Ver gráfica por años	Véase cuadro 3.39
CU-31	Cerrar sesión	Véase cuadro 3.40

3.4. CASOS DE USO

- RF-01 El sistema deberá implementar un sistema de control de acceso.
- RF-02 El sistema deberá responder en función del tipo de usuario.
- RF-03 El sistema deberá solo servir información a usuarios autorizados.
- RF-04 El sistema deberá implementar un control de sesión de usuario.
- RF-05 El sistema deberá validar todos los formularios, según el tipo de campo, atributo o entidad
- RF-06 El sistema deberá responder en función del tipo de usuario (rol).
- RF-07 El sistema deberá tener un panel de control para los usuarios administrador donde gestionar la información de las entidades almacenadas en el sistema: Usuarios, Administradores, Paneles solares, Inversores e Instalaciones.
- RF-08 El sistema deberá mostrar la información de cada tipo de objeto gestionado en forma de tabla al administrador.
- RF-09 El sistema deberá poder hacer búsquedas filtradas en las tablas por campo de búsqueda.
- RF-10 El sistema deberá poder permitir gestionar dos tipos de usuarios: Administrador, usuario normal.
- RF-11 El sistema deberá poder permitir al administrador relacionar un usuario normal con su instalación.
- RF-12 El sistema deberá poder permitir al administrador relacionar un inversor con una instalación.
- RF-13 El sistema deberá poder permitir al administrador relacionar un panel solar con una instalación
- RF-14 El sistema deberá poder permitir al administrador gestionar información sobre inversores (CRUD).
- RF-15 El sistema deberá poder permitir al administrador gestionar información sobre paneles solares (CRUD).
- RF-16 El sistema deberá poder permitir al administrador gestionar información sobre instalaciones domésticas (CRUD).
- RF-17 El sistema deberá poder permitir al usuario normal acceder a la visualización de mediciones de consumo y mediciones de producción.

Cuadro 3.1: Requisitos funcionales: Parte 1

- RF-18 El sistema deberá implementar un dispositivo hardware, compuesto por dos módulos, el de producción y el de consumo, para la toma de medidas.
- RF-19 El sistema deberá tomar medidas de producción muestreando intensidad y voltaje y deberá calcular potencia instantánea, potencia activa, potencia aparente y factor de potencia a raíz de ellas
- RF-20 El sistema deberá tomar medidas de consumo muestreando intensidad y voltaje y deberá calcular potencia instantánea, potencia activa, potencia aparente, potencia máxima instantánea y factor de potencia a raíz de ellas.
- RF-21 El sistema deberá tomar medidas cada minuto para los históricos.
- RF-22 El sistema deberá tomar medidas cada 5 segundos para la muestra en tiempo real.
- RF-23 El sistema deberá comunicarse con una API de terceros para obtener información sobre provincias y localidades.
- RF-24 El sistema deberá mostrar a los usuarios normales las mediciones de consumo y producción en tiempo real
- RF-25 El sistema deberá mostrar al usuario un histórico clasificado por día de las mediciones de su dispositivo
- RF-26 El sistema deberá mostrar al usuario un histórico clasificado por meses de las mediciones de su dispositivo.
- RF-27 El sistema deberá mostrar al usuario un histórico clasificado por años de las mediciones de su dispositivo.
- RF-28 El sistema deberá mostrar al usuario normal toda la información personal y datos sensibles que le conciernen
- RF-29 El sistema deberá permitir la modificación de la contraseña de usuarios normales.
- RF-30 El sistema deberá mostrar al usuario normal toda la información referente a su instalación doméstica.
- RF-31 El sistema deberá mostrar al usuario normal un potenciómetro en tiempo real con las medidas de potencia instantánea de su instalación.

Cuadro 3.2: Requisitos funcionales: Parte 2

3.4. CASOS DE USO

- RNF-01 El sistema deberá implementar un frontend diseñado con la tecnología REACT
- RNF-02 El sistema deberá implementar una API backend diseñada con la tecnología NodeJS y Express.
- RNF-03 El sistema debe ser multiplataforma.
- RNF-04 El sistema deberá utilizar la tecnología MongoDB Atlas para manejar la persistencia.
- RNF-05 El sistema deberá utilizar el middleware JsonWebToken para el control de acceso a recursos.
- RNF-06 El sistema deberá implementar un dispositivo de medición diseñado con arduino nano y raspberry pi zero w.
- RNF-07 El sistema deberá utilizar Raspbian como SO de la raspberry pi zero w de medición.

Cuadro 3.3: Requisitos no funcionales

- RNFA-01 El sistema deberá ser accesible desde cualquier navegador web.
- RNFA-02 El sistema deberá ser accesible desde plataformas móviles.

Cuadro 3.4: Requisitos no funcionales: Accesibilidad

- RNFS-01 El sistema deberá disponer de un sistema de autenticación.
- RNFS-02 El sistema debe encriptar la información sensible, evitando retransmisiones en texto plano.
- RNFS-03 El sistema deberá implementar un mecanismo de emparejamiento instalación-dispositivo de medición.
- RNFS-04 El sistema deberá implementar middlewares para controlar el acceso a recursos.

Cuadro 3.5: Requisitos no funcionales: Seguridad

- RNFU-01 El sistema debe ser confiable, asegurando la respuesta esperada.
- RNFU-02 El sistema debe mostrar una vista en el navegador en función del tipo de usuario.
- RNFU-03 El sistema deberá ser fácilmente usable una vez leído el manual de funcionamiento.

Cuadro 3.6: Requisitos no funcionales: Usabilidad

- RNFD-01 El sistema deberá estar disponible las 24 horas del día, los 7 días de la semana.
- RNFD-02 El sistema deberá tomar medidas de consumo y producción las 24 horas del día, los 7 días de la semana.

Cuadro 3.7: Requisitos no funcionales: Disponibilidad

- RNFI-01 El sistema deberá almacenar la información de los usuarios clientes.
- RNFI-02 El sistema deberá almacenar la información de las instalaciones de los usuarios.
- RNFI-03 El sistema deberá poder vincular un usuario con su instalación.
- RNFI-04 El sistema deberá poder almacenar información sobre inversores.
- RNFI-05 El sistema deberá poder almacenar información sobre paneles solares.
- RNFI-06 El sistema deberá poder almacenar información sobre usuarios administradores.
- RNFI-07 El sistema deberá poder almacenar información sobre mediciones de consumo.
- RNFI-08 El sistema deberá poder almacenar información sobre mediciones de producción.

Cuadro 3.8: Requisitos de información

3.4. CASOS DE USO

- RNFRI-01 La información almacenada sobre usuarios normales debe satisfacer que no pueda haber 2 usuarios con el mismo NIE/NIF
- RNFRI-02 La información almacenada sobre instalaciones debe satisfacer que una instalación solo este vinculada con 1 cliente.
- RNFRI-03 La información almacenada sobre mediciones, ya sean consumo o producción, debe satisfacer que dichas mediciones esten vinculadas solo a 1 instalación
- RNFRI-04 La información almacenada sobre instalaciones debe satisfacer que solo haya 1 tipo de inversor vinculado.
- RNFRI-05 La información almacenada sobre instalaciones debe satisfacer que solo haya 1 tipo de panel solar vinculado.

Cuadro 3.9: Requisitos de restricción de información

CU-01	Iniciar sesión
Actor	Administrador
Descripción	El administrador se identifica para acceder a sus funciones en el sistema.
Precondición	El usuario ha de ser administrador
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la URL de la aplicación. 2. El sistema muestra el login de acceso. 3. El usuario rellena el login de acceso con sus datos. 4. El sistema comprueba en BBDD la existencia del usuario y la vericidad de su contraseña. 5. El sistema redirige al usuario a la página principal de administrador
Postcondicion	-
Excepciones	<ol style="list-style-type: none"> 1. El usuario deja campos del login vacios. 2. El usuario introduce datos erróneos. 3. El usuario no existe en BBDD.
Frecuencia	Media

Cuadro 3.10: Administrador: CU-01: Iniciar sesión

CU-02	Crear administrador
Actor	Administrador
Descripción	El administrador creará un nuevo usuario administrador
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Nuevo administrador</i>. 2. El sistema muestra el formulario para la creación de administradores. 3. El usuario rellena el formularios. 4. El sistema comprueba que no hay campos vacios y la válidez de los datos. 5. El sistema crea el nuevo usuario administrador en BBDD. 6. El sistema muestra un mensaje de creación correcta y limpia el formulario.
Postcondicion	Se ha creado un nuevo administrador
Excepciones	<ol style="list-style-type: none"> 1. El usuario introduce campos vacios/inválidos. 2. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.11: Administrador: CU-02: Crear administrador

CU-03	Eliminar administrador
Actor	Administrador
Descripción	El administrador eliminará un usuario administrador
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Administradores</i>. 2. El sistema muestra todos los administradores en forma de tabla. 3. El usuario selecciona uno. 4. El sistema muestra el boton de borrar. 5. El usuario pica en el botón de borrar. 6. El sistema elimina el usuario seleccionado de BBDD.
Postcondicion	Se ha eliminado un administrador
Excepciones	<ol style="list-style-type: none"> 1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.12: Administrador: CU-03: Eliminar administrador

3.4. CASOS DE USO

CU-04	Modificar administrador
Actor	Administrador
Descripción	El administrador modificará un usuario administrador
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Administradores</i>. 2. El sistema muestra todos los administradores en forma de tabla. 3. El usuario selecciona uno. 4. El sistema muestra el boton de modificar. 5. El usuario pica en el botón de modificar. 6. El sistema muestra el formulario para modificar administrador. 6. El usuario rellena el formulario con los nuevos datos. 7. El sistema guarda la información en BBDD. 8. El sistema oculta el formulario y actualiza la tabla
Postcondicion	Se ha modificado un administrador
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Baja

Cuadro 3.13: Administrador: CU-04: Modificar administrador

CU-05	Ver administradores
Actor	Administrador
Descripción	El administrador visualizará todos los administradores
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Administradores</i>. 2. El sistema muestra todos los administradores en forma de tabla.
Postcondicion	Se muestra información sobre adminsitradores
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.14: Administrador: CU-05: Ver administradores

CU-06	Crear usuario
Actor	Administrador
Descripción	El administrador creará un nuevo usuario
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Nuevo usuario</i>. 2. El sistema muestra el formulario para la creación de usuarios. 3. El usuario rellena el formularios. 4. El sistema comprueba que no hay campos vacios y la válidez de los datos. 5. El sistema crea el nuevo usuario en BBDD. 6. El sistema muestra un mensaje de creación correcta y limpia el formulario.
Postcondicion	Se ha creado un nuevo usuario
Excepciones	<ol style="list-style-type: none"> 1. El usuario introduce campos vacios/inválidos. 2. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.15: Administrador: CU-06: Crear usuario

CU-07	Eliminar usuario
Actor	Administrador
Descripción	El administrador eliminará un usuario
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Usuarios</i>. 2. El sistema muestra todos los usuarios en forma de tabla. 3. El usuario selecciona uno. 4. El sistema muestra el boton de borrar. 5. El usuario pica en el botón de borrar. 6. El sistema elimina el usuario seleccionado de BBDD.
Postcondicion	Se ha eliminado un usuario
Excepciones	<ol style="list-style-type: none"> 1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.16: Administrador: CU-07: Eliminar usuario

3.4. CASOS DE USO

CU-08	Modificar usuario
Actor	Administrador
Descripción	El administrador modificará un usuario
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Usuarios</i>. 2. El sistema muestra todos los usuarios en forma de tabla. 3. El usuario selecciona uno. 4. El sistema muestra el boton de modificar. 5. El usuario pica en el botón de modificar. 6. El sistema muestra el formulario para modificar usuario. 6. El usuario rellena el formulario con los nuevos datos. 7. El sistema guarda la información en BBDD. 8. El sistema oculta el formulario y actualiza la tabla
Postcondicion	Se ha modificado un usuario
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Baja

Cuadro 3.17: Administrador: CU-08: Modificar usuario

CU-09	Ver usuarios
Actor	Administrador
Descripción	El administrador visualizará todos los usuarios
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Usuarios</i>. 2. El sistema muestra todos los usuarios en forma de tabla.
Postcondicion	Se muestra información sobre usuarios
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.18: Administrador: CU-09: Ver usuarios

CU-10	Crear inversor
Actor	Administrador
Descripción	El administrador creará un nuevo inversor
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Nuevo inversor</i>. 2. El sistema muestra el formulario para la creación de inversores. 3. El usuario rellena el formularios. 4. El sistema comprueba que no hay campos vacios y la válidez de los datos. 5. El sistema crea el nuevo inversor en BBDD. 6. El sistema muestra un mensaje de creación correcta y limpia el formulario.
Postcondicion	Se ha creado un nuevo inversor
Excepciones	<ol style="list-style-type: none"> 1. El usuario introduce campos vacios/inválidos. 2. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.19: Administrador: CU-10: Crear inversor

CU-11	Eliminar inversor
Actor	Administrador
Descripción	El administrador eliminará un inversor
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Inversores</i>. 2. El sistema muestra todos los inversores en forma de tabla. 3. El usuario selecciona uno. 4. El sistema muestra el boton de borrar. 5. El usuario pica en el botón de borrar. 6. El sistema elimina el inversor seleccionado de BBDD.
Postcondicion	Se ha eliminado un inversor
Excepciones	<ol style="list-style-type: none"> 1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.20: Administrador: CU-11: Eliminar inversor

3.4. CASOS DE USO

CU-12	Modificar inversor
Actor	Administrador
Descripción	El administrador modificará un inversor
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Inversores</i>. 2. El sistema muestra todos los inversores en forma de tabla. 3. El usuario selecciona uno. 4. El sistema muestra el boton de modificar. 5. El usuario pica en el botón de modificar. 6. El sistema muestra el formulario para modificar inversor. 6. El usuario rellena el formulario con los nuevos datos. 7. El sistema guarda la información en BBDD. 8. El sistema oculta el formulario y actualiza la tabla
Postcondicion	Se ha modificado un inversor
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Baja

Cuadro 3.21: Administrador: CU-12: Modificar inversor

CU-13	Ver inversores
Actor	Administrador
Descripción	El administrador visualizará todos los inversores
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Inversores</i>. 2. El sistema muestra todos los inversores en forma de tabla.
Postcondicion	Se muestra información sobre inversores
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.22: Administrador: CU-13: Ver inversores

CU-14	Crear panel solar
Actor	Administrador
Descripción	El administrador creará un nuevo panel solar
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Nuevo panel solar</i>. 2. El sistema muestra el formulario para la creación de paneles solares. 3. El usuario rellena el formularios. 4. El sistema comprueba que no hay campos vacios y la válidez de los datos. 5. El sistema crea el nuevo panel solar en BBDD. 6. El sistema muestra un mensaje de creación correcta y limpia el formulario.
Postcondicion	Se ha creado un nuevo panel solar
Excepciones	<ol style="list-style-type: none"> 1. El usuario introduce campos vacios/inválidos. 2. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.23: Administrador: CU-14: Crear panel solar

CU-15	Eliminar panel solar
Actor	Administrador
Descripción	El administrador eliminará un panel solar
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Paneles solares</i>. 2. El sistema muestra todos los paneles solares en forma de tabla. 3. El usuario selecciona uno. 4. El sistema muestra el boton de borrar. 5. El usuario pica en el botón de borrar. 6. El sistema elimina el panel solar seleccionado de BBDD.
Postcondicion	Se ha eliminado un panel solar
Excepciones	<ol style="list-style-type: none"> 1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.24: Administrador: CU-15: Eliminar panel solar

3.4. CASOS DE USO

CU-16	Modificar panel solar
Actor	Administrador
Descripción	El administrador modificará un panel solar
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Paneles solares</i>. 2. El sistema muestra todos los paneles solares en forma de tabla. 3. El usuario selecciona uno. 4. El sistema muestra el boton de modificar. 5. El usuario pica en el botón de modificar. 6. El sistema muestra el formulario para modificar panel solar. 6. El usuario rellena el formulario con los nuevos datos. 7. El sistema guarda la información en BBDD. 8. El sistema oculta el formulario y actualiza la tabla
Postcondicion	Se ha modificado un panel solar
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Baja

Cuadro 3.25: Administrador: CU-16: Modificar panel solar

CU-17	Ver paneles solares
Actor	Administrador
Descripción	El administrador visualizará todos los paneles solares
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Paneles solares</i>. 2. El sistema muestra todos los paneles solares en forma de tabla.
Postcondicion	Se muestra información sobre paneles solares
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.26: Administrador: CU-17: Ver paneles solares

CU-18	Crear instalación
Actor	Administrador
Descripción	El administrador creará una nueva instalación
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Nueva instalación</i>. 2. El sistema muestra el formulario para la creación de instalaciones. 3. El usuario rellena el formularios, seleccionando el usuario propietario de la misma. 4. El sistema comprueba que no hay campos vacios y la válidez de los datos. 5. El sistema crea la nueva instalación en BBDD. 6. El sistema muestra un mensaje de creación correcta y limpia el formulario.
Postcondicion	Se ha creado una nueva instalación
Excepciones	<ol style="list-style-type: none"> 1. El usuario introduce campos vacios/inválidos. 2. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.27: Administrador: CU-18: Crear instalación

CU-19	Eliminar instalación
Actor	Administrador
Descripción	El administrador eliminará una instalación
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Instalaciones</i>. 2. El sistema muestra todas las instalaciones en forma de tabla. 3. El usuario selecciona uno. 4. El sistema muestra el boton de borrar. 5. El usuario pica en el botón de borrar. 6. El sistema elimina la instalación seleccionada de BBDD.
Postcondicion	Se ha eliminado una instalación
Excepciones	<ol style="list-style-type: none"> 1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.28: Administrador: CU-19: Eliminar instalación

3.4. CASOS DE USO

CU-20	Modificar instalación
Actor	Administrador
Descripción	El administrador modificará una instalación
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Instalaciones</i>. 2. El sistema muestra todas las instalaciones en forma de tabla. 3. El usuario selecciona una. 4. El sistema muestra el boton de modificar. 5. El usuario pica en el botón de modificar. 6. El sistema muestra el formulario para modificar instalación. 6. El usuario rellena el formulario con los nuevos datos. 7. El sistema guarda la información en BBDD. 8. El sistema oculta el formulario y actualiza la tabla
Postcondicion	Se ha modificado una instalación
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Baja

Cuadro 3.29: Administrador: CU-20: Modificar instalación

CU-21	Ver instalaciones
Actor	Administrador
Descripción	El administrador visualizará todas las instalaciones
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario acceder a la sección <i>Instalaciones</i>. 2. El sistema muestra todas las instalaciones en forma de tabla.
Postcondicion	Se muestra información sobre instalaciones
Excepciones	1. No hay conexión con la BBDD.
Frecuencia	Media

Cuadro 3.30: Administrador: CU-21: Ver instalaciones

CU-22	Cerrar sesión
Actor	Administrador
Descripción	El usuario se cierra sesión y abandona el sistema
Precondición	El usuario ha de ser administrador y estar logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la sección <i>Cerrar sesión</i>. 2. El sistema elimina los datos de sesión creados para el usuario. 3. El sistema redirige al usuario a la página de <i>Login</i>
Postcondicion	El usuario abandona el sistema
Excepciones	-
Frecuencia	Media

Cuadro 3.31: Administrador: CU-22: Cerrar sesión

CU-23	Iniciar sesión
Actor	Usuario
Descripción	El usuario se identifica para acceder a sus funciones en el sistema.
Precondición	El usuario ha de estar registrado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la URL de la aplicación. 2. El sistema muestra el login de acceso. 3. El usuario rellena el login de acceso con sus datos. 4. El sistema comprueba en BBDD la existencia del usuario y la vericidad de su contraseña. 5. El sistema redirige al usuario a la página principal de administrador
Postcondicion	-
Excepciones	<ol style="list-style-type: none"> 1. El usuario deja campos del login vacios. 2. El usuario introduce datos erróneos. 3. El usuario no existe en BBDD.
Frecuencia	Media

Cuadro 3.32: Usuario: CU-23: Iniciar sesión

3.4. CASOS DE USO

CU-24	Ver perfil
Actor	Usuario
Descripción	El usuario accederá a la vista con sus datos personales
Precondición	El usuario ha de estar registrado y logueado
Sencuencia normal	<ol style="list-style-type: none">1. El usuario accede a la zona "Perfil".2. El sistema muestra la información con sus datos personales.3. El usuario rellena el login de acceso con sus datos.4. El sistema comprueba en BBDD la existencia del usuario y la vericidad de su contraseña.5. El sistema redirige al usuario a la página principal de administrador
Postcondicion	El sistema mostrará los datos
Excepciones	-
Frecuencia	Media

Cuadro 3.33: Usuario: CU-24: Ver perfil

CU-25	Modificar contraseña
Actor	Usuario
Descripción	El usuario modificará su contraseña desde la zona "Perfil"
Precondición	El usuario ha de estar registrado y logueado.
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la zona "Perfil". 2. El sistema muestra la información con sus datos personales. 3. El usuario pica al botón <i>Cambiar contraseña</i>. 4. El sistema muestra el formulario para el cambio de contraseña 5. El usuario rellena el formulario. 6. El sistema comprueba la validez de la contraseña nueva
Postcondicion	El sistema mostrará los datos
Excepciones	<ol style="list-style-type: none"> 1. El usuario deja campos del formulario vacios. 2. El usuario introduce una contraseña débil, que no cumple los requisitos
Frecuencia	Baja

Cuadro 3.34: Usuario: CU-25: Modificar contraseña

CU-26	Ver instalación
Actor	Usuario
Descripción	El usuario tendrá acceso a la información referente a su instalación
Precondición	El usuario ha de estar registrado y logueado.
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la pantalla principal. 2. El sistema muestra la información con sus datos referentes a la instalación.
Postcondicion	El sistema mostrará los datos
Excepciones	-
Frecuencia	Alta

Cuadro 3.35: Usuario: CU-26: Ver instalación

3.4. CASOS DE USO

CU-27	Ver gráfica en tiempo real
Actor	Usuario
Descripción	El usuario tendrá acceso a la gráfica de mediciones en tiempo real
Precondición	El usuario ha de estar registrado y logueado. Ha de tener una instalación asociada
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la pantalla principal. 2. El sistema muestra la gráfica con datos en tiempo real de medición.
Postcondicion	El sistema mostrará los datos
Excepciones	<ol style="list-style-type: none"> 1. El usuario no tiene una instalación asociada. 2. El dispositivo de medición del usuario esta desconectado.
Frecuencia	Alta

Cuadro 3.36: Usuario: CU-27: Ver gráfica en tiempo real

CU-28	Ver gráfica por días
Actor	Usuario
Descripción	El usuario tendrá acceso a la gráfica de mediciones clasificado por días
Precondición	El usuario ha de estar registrado y logueado. Ha de tener una instalación asociada
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la sección <i>Diario</i>. 2. El sistema muestra la gráfica para las mediciones del día actual. 3. El usuario selecciona un día pasado. 4. El sistema muestra la gráfica para las mediciones del dia seleccionado
Postcondicion	El sistema mostrará los datos
Excepciones	<ol style="list-style-type: none"> 1. El usuario no tiene una instalación asociada. 2. El dispositivo de medición no tiene medidas para el día indicado.
Frecuencia	Alta

Cuadro 3.37: Usuario: CU-28: Ver gráfica por días

CU-29	Ver gráfica por mes
Actor	Usuario
Descripción	El usuario tendrá acceso a la gráfica de mediciones clasificado por días
Precondición	El usuario ha de estar registrado y logueado. Ha de tener una instalación asociada
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la sección <i>Mensual</i>. 2. El sistema muestra la gráfica para las mediciones del mes actual. 3. El usuario selecciona un mes pasado. 4. El sistema muestra la gráfica para las mediciones del mes seleccionado.
Postcondicion	El sistema mostrará los datos
Excepciones	<ol style="list-style-type: none"> 1. El usuario no tiene una instalación asociada. 2. El dispositivo de medición no tiene medidas para el mes indicado.
Frecuencia	Alta

Cuadro 3.38: Usuario: CU-29: Ver gráfica por mes

3.4. CASOS DE USO

CU-30	Ver gráfica por año
Actor	Usuario
Descripción	El usuario tendrá acceso a la gráfica de mediciones clasificado por días
Precondición	El usuario ha de estar registrado y logueado. Ha de tener una instalación asociada
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la sección <i>Anual</i>. 2. El sistema muestra la gráfica para las mediciones del año actual. 3. El usuario selecciona un año pasado. 4. El sistema muestra la gráfica para las mediciones del año seleccionado
Postcondicion	El sistema mostrará los datos
Excepciones	<ol style="list-style-type: none"> 1. El usuario no tiene una instalación asociada. 2. El dispositivo de medición no tiene medidas para el año indicado.
Frecuencia	Media

Cuadro 3.39: Usuario: CU-30: Ver gráfica por año

CU-31	Cerrar sesión
Actor	Usuario
Descripción	El usuario se cierra sesión y abandona el sistema
Precondición	El usuario ha de estar registrado y logueado
Sencuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la sección <i>Cerrar sesión</i>. 2. El sistema elimina los datos de sesión creados para el usuario. 3. El sistema redirige al usuario a la página de <i>Login</i>
Postcondicion	El usuario abandona el sistema
Excepciones	-
Frecuencia	Media

Cuadro 3.40: Usuario: CU-31: Cerrar sesión

3.5. Modelo de dominio

A continuación se presenta el modelo de dominio del problema a resolver, en él se muestran las tablas del sistema y sus relaciones.

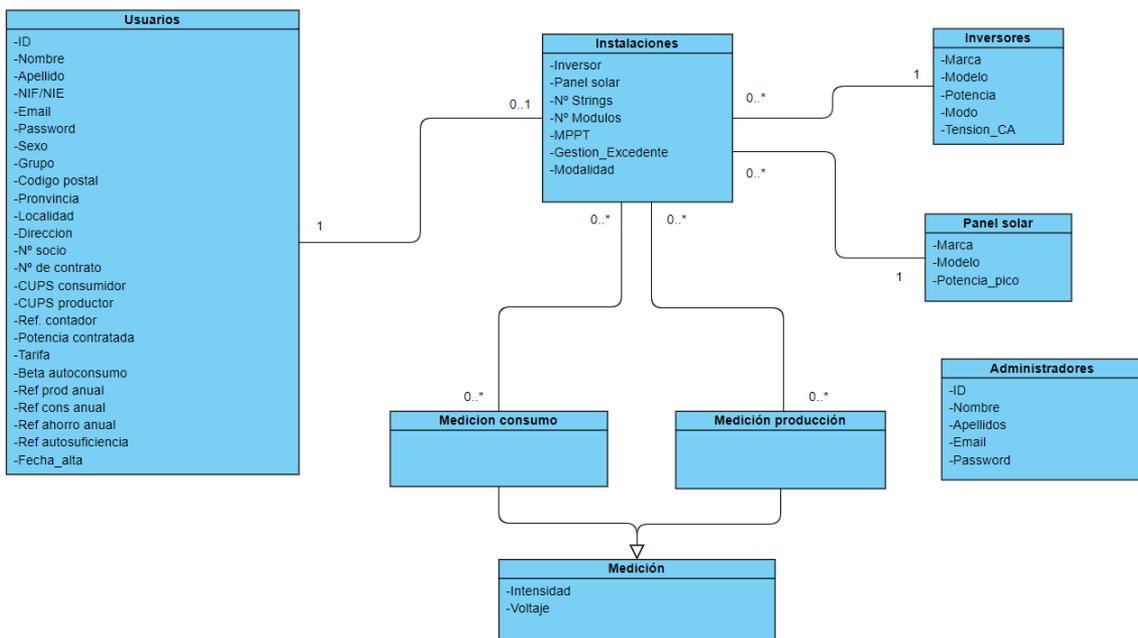


Figura 3.3: Modelo de dominio

3.6. Diagramas de secuencia

A continuación vamos a detallar los diagramas de secuencia de los casos de uso mas significativos. Podríamos explicar todos uno a uno, pero quedaría inutilmente extenso, ya que muchos casos son equivalentes en su flujo de ejecución.

3.6.1. Acceso al sistema

Véase la figura 3.4 para obtener una vista del flujo de actividad seguido en el inicio de sesión en el sistema.

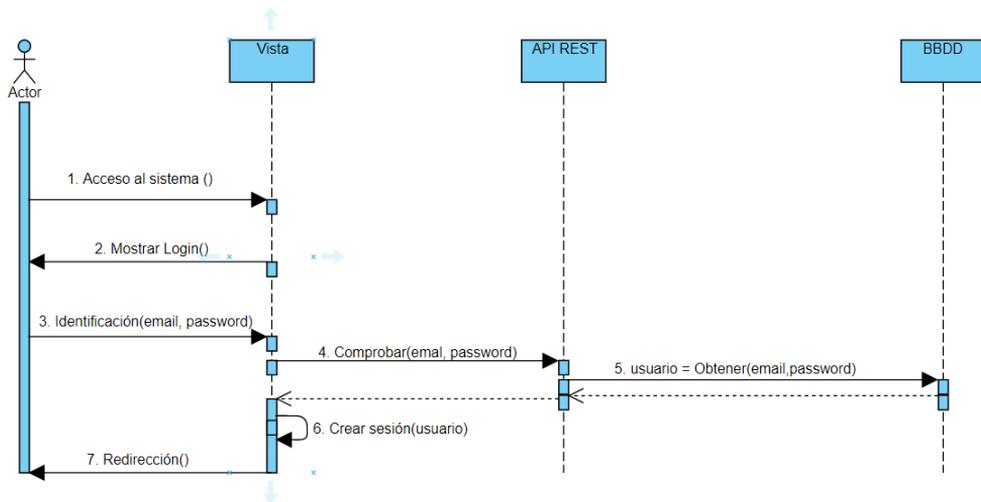


Figura 3.4: Diagrama de secuencia: Acceso al sistema

3.6.2. Crear Entidad

Véase la figura 3.5 para obtener una vista del flujo de actividad del caso de uso correspondiente a la creación de objetos del sistema, válido para usuarios, administradores, inversores, instalaciones y paneles solares.

3.6.3. Eliminar Entidad

Véase la figura 3.6 para obtener una vista del flujo de actividad del caso de uso correspondiente a la eliminación de objetos del sistema, válido para usuarios, administradores, inversores, instalaciones y paneles solares.

3.6.4. Modificar Entidad

Véase la figura 3.7 para obtener una vista del flujo de actividad del caso de uso correspondiente a la modificación de objetos del sistema.

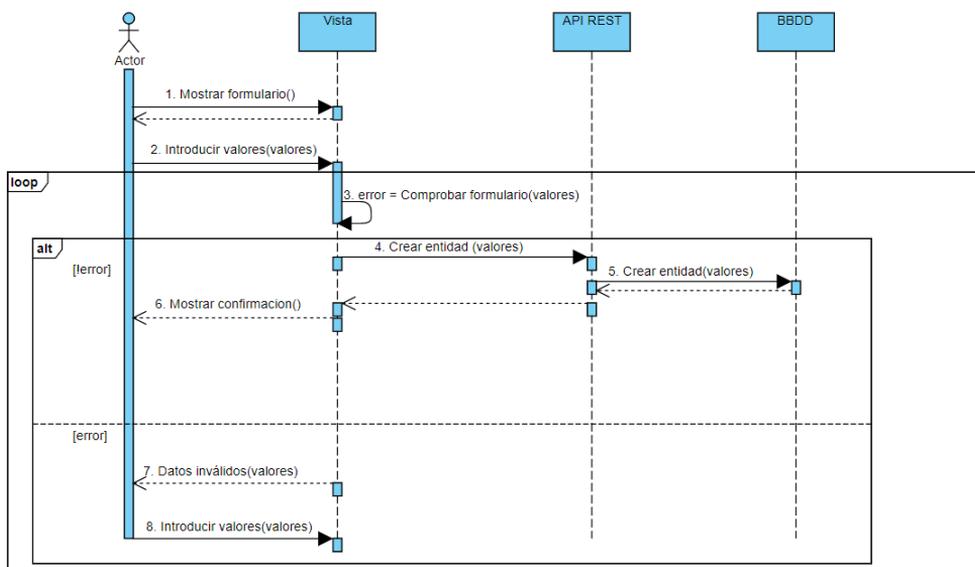


Figura 3.5: Diagrama de secuencia: Crear entidades

3.6.5. Obtener Entidad

Véase la figura 3.8 para obtener una vista del flujo de actividad del caso de uso correspondiente a la obtención de objetos del sistema.

3.6.6. Registrar medición de producción

Véase la figura 3.9 para obtener una vista del flujo de actividad del caso de uso correspondiente a la obtención de medidas de producción.

3.6.7. Registrar medición de consumo

Véase la figura 3.10 para obtener una vista del flujo de actividad del caso de uso correspondiente a la obtención de medidas de consumo.

3.6. DIAGRAMAS DE SECUENCIA

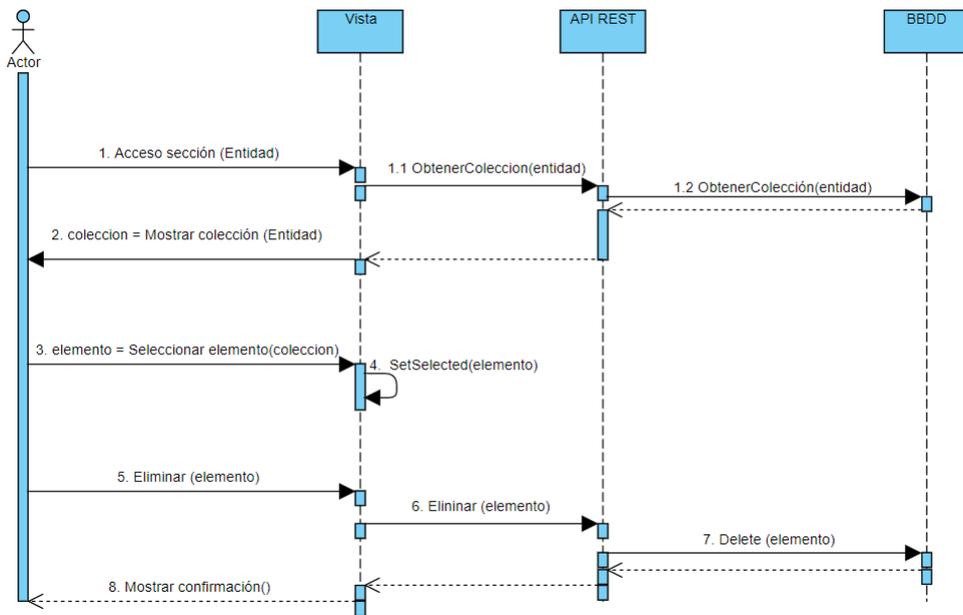


Figura 3.6: Diagrama de secuencia: Eliminar entidades

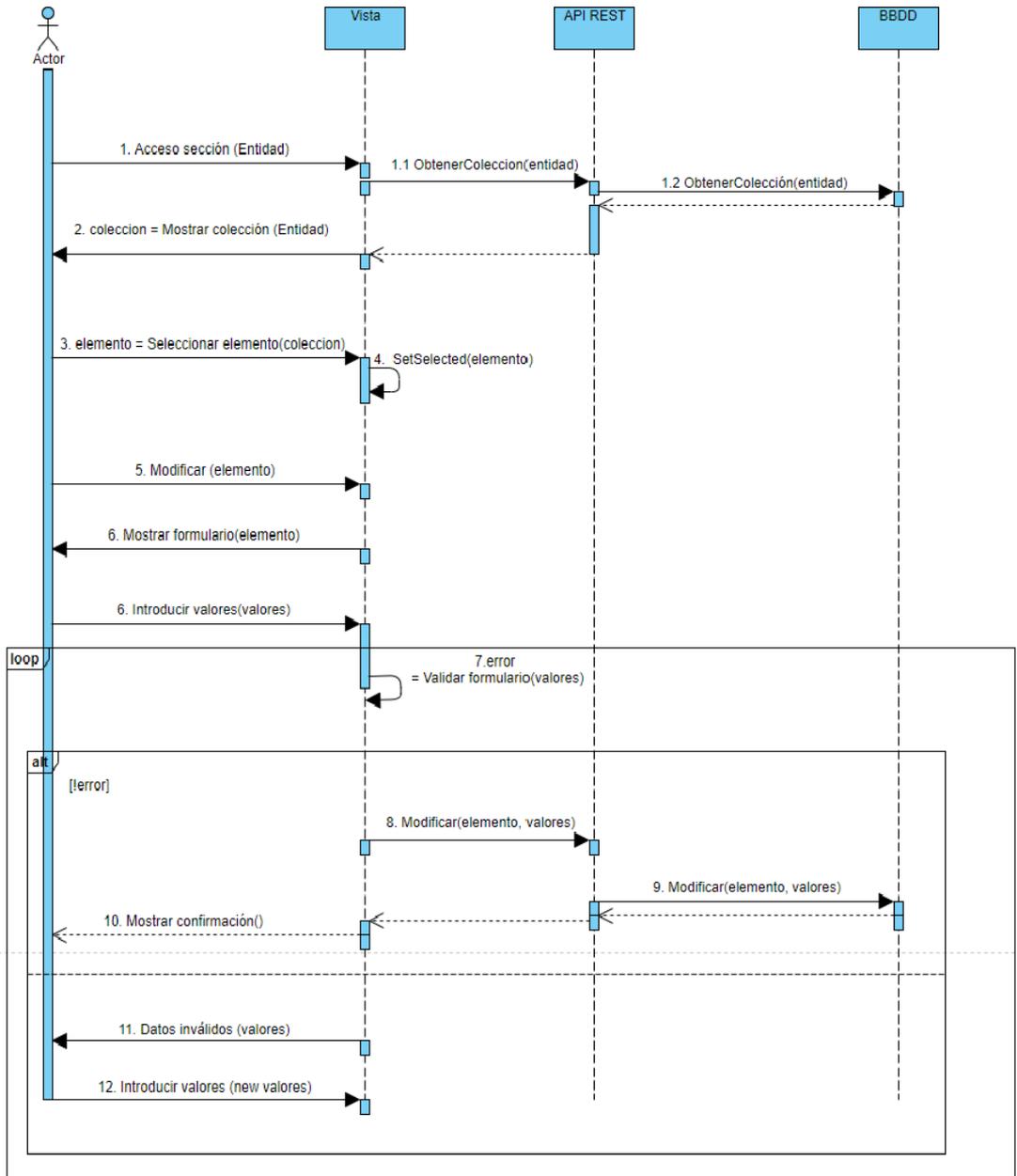


Figura 3.7: Diagrama de secuencia: Modificar entidades

3.6. DIAGRAMAS DE SECUENCIA

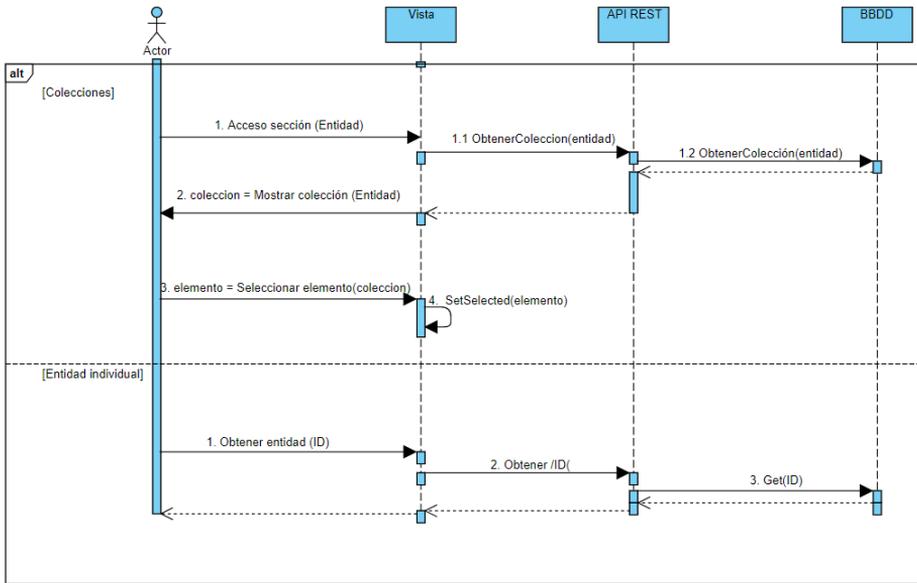


Figura 3.8: Diagrama de secuencia: Obtener entidades

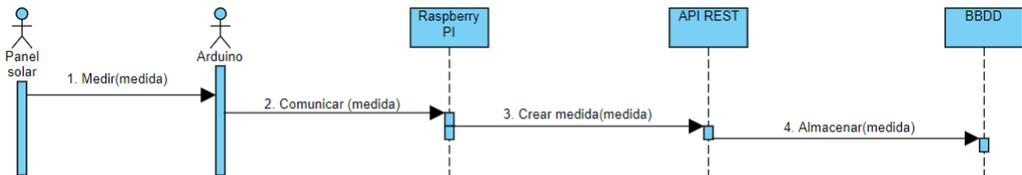


Figura 3.9: Diagrama de secuencia: Obtener medida producción

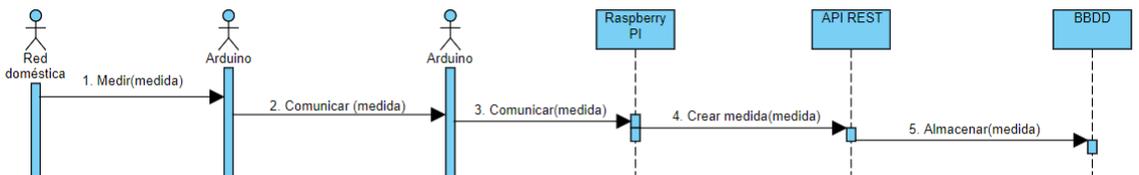


Figura 3.10: Diagrama de secuencia: Obtener medida consumo

3.7. Diccionario de datos

En este apartado vamos enumerar detalladamente las características de los datos que el sistema debe tratar, así como las *Collections* que la aplicación almacena en *MongoDB*.

Todas las colecciones tienen activada la opción de *MongoDB timestamps*, cuya utilidad es almacenar la fecha de creación y modificación de cada elemento.

3.7.1. Usuario

Véase Cuadro 3.41.

Campo	Tipo	Requerido	Único
Email	String	Sí	Sí
Nombre	String	Sí	No
Apellido	String	Sí	No
Contraseña	String	Sí	No
IDFiscal	String	Sí	Sí
Teléfono	Number	Sí	No
Sexo	Enum	Sí	No
Grupo	Enum	Sí	No
IDProvincia	Number	Sí	No
NombreProvincia	String	Sí	No
IDLocalidad	Number	Sí	No
NombreLocalidad	String	Sí	No
CP	Number	Sí	No
Dirección	Number	Sí	No
Tarifa	Enum	Sí	No
Libre	Boolean	Sí	No
BetaAutoconsumo	Number	Sí	No
AnualProdRef	Number	Sí	No
AnualConsRef	Number	Sí	No
AutosuficienciaRef	Number	Sí	No
AnualAhorroRef	Number	Sí	No
NumSocio	Number	Sí	Sí
NumContrato	Number	Sí	No
CUPSConsumidor	String	Sí	No
CUPSProductor	String	Sí	No
ContadorRef	String	Sí	No
Potencia	Number	Sí	No

Cuadro 3.41: Diccionario de datos: Usuario

3.7.2. Administrador

Véase Cuadro 3.42.

Campo	Tipo	Requerido	Único
Email	String	Sí	Sí
Nombre	String	Sí	No
Apellido	String	Sí	No
Contraseña	String	Sí	No

Cuadro 3.42: Diccionario de datos: Administrador

3.7.3. Inversor

Véase Cuadro 3.43.

Campo	Tipo	Requerido	Único
Marca	String	Sí	No
Modelo	String	Sí	No
Potencia	Number	Sí	No
Tension CA	Number	Sí	No
Tipo	Enum	Sí	No

Cuadro 3.43: Diccionario de datos: Inversor

3.7.4. Panel solar

Véase Cuadro 3.44.

Campo	Tipo	Requerido	Único
Marca	String	Sí	No
Modelo	String	Sí	No
Potencia	Number	Sí	No

Cuadro 3.44: Diccionario de datos: Panel solar

3.7.5. Instalación

Véase Cuadro 3.45.

Campo	Tipo	Requerido	Único	Ref
Usuario	ObjectId	Sí	Sí	Usuario
Inversor	ObjectId	Sí	No	Inversor
Panel	ObjectId	Sí	No	Panel solar
NumStrings	Number	Sí	No	No
NumModulos	Number	Sí	No	No
MPPT	Number	Sí	No	No
GestionExcedente	Enum	Sí	No	No
Modalidad	Enum	Sí	No	No

Cuadro 3.45: Diccionario de datos: Instalación

3.7.6. Medición

Véase Cuadro 3.46.

Campo	Tipo	Requerido	Único	Ref
Intensidad	Number	Sí	Sí	No
Voltaje	Number	Sí	No	No
Potencia	Number	Sí	No	No
Instalación	ObjectId	Sí	No	Instalación

Cuadro 3.46: Diccionario de datos: Medición

3.7.7. Enumeraciones

En la colección *Usuario*:

- Grupo:
 - Persona física
 - Socio Nivel 1
 - Socio Nivel 2

- Energética
 - Ayuntamientos y Administraciones Públicas
 - Comunidad de Vecinos
 - Cooperativa
 - Empresa pequeña
 - Empresa mediana
 - Empresa grande
 - Gestor de fincas
 - Negocio/Autónomo
 - ONG
 - Organizaciones políticas/sociales
- Sexo:
 - Hombre
 - Mujer
 - Otros
 - Tarifa:
 - 2.0TD
 - 3.0TD

En la colección *Inversor*:

- Tipo:
 - Monofásico
 - Trifásico

En la colección *Instalación*:

- Gestión de excedente:
 - Con vertido a red.
 - Sin vertido a red.
- Modalidad:
 - Compartido.
 - No compartido.

Capítulo 4

Diseño

El diseño del sistema ha supuesto un desafío para el alumno, tanto por la complejidad del sistema, como por el proceso de diseño del mismo.

En dicho proceso se requieren conocimientos que se escapan del ámbito informático, incluyendo cálculos sobre magnitudes físicas o el diseño de circuitos para la medición de tensión e intensidad de la red eléctrica.

Debido a la complejidad adherida al diseño hardware del dispositivo, no vamos a detallar en esta sección dicho proceso. Para ello, se ha dedicado el *Capítulo VI* de manera íntegra, mejorando la claridad y comprensión del lector.

4.1. Arquitectura

En esta sección del documento vamos a analizar la arquitectura elegida para el desarrollo del proyecto. Dividiremos la sección en dos partes: Arquitectura física y Arquitectura lógica.

4.1.1. Arquitectura física

Uno de los requisitos elementales del sistema es la localización del servidor del front-end y del servidor de la API en el edificio de *Energética Coop*. A partir de esa decisión, nos condicionamos a que el dispositivo de medición debe contar con la posibilidad de transmitir datos mediante WIFI.

Otro condicionante para el diseño de la arquitectura física es la posible distancia, en el

4.1. ARQUITECTURA

hogar del cliente, entre el módulo de producción y el módulo de consumo. Dicha distancia no puede superar los 100 metros.

A continuación se describen los diferentes elementos que conforman la arquitectura física.

Nodos

- Cliente.
- Central energética.
- Dispositivo de medición.
- Front-end: Navegador del cliente.
- MongoDB Atlas.

Diagrama de despliegue

Véase la figura 4.1 para ver el diagrama de despliegue.

4.1.2. Arquitectura lógica

Se ha diseñado la arquitectura lógica en base a la arquitectura física anteriormente planteada.

Para el desarrollo del proyecto se ha seguido un modelo arquitectónico basado en capas, donde las capas de presentación y negocio se localizan en la sede de *Energética* y la capa de lógica de datos en *MongoDB Atlas*.

Capa de presentación

La capa de presentación es la encargada de gestionar las vistas de usuario. Controla las entradas, valida campos, gestiona sesión... Es la capa de interacción de cara al usuario.

Su correspondencia a nivel de implementación sería el front-end realizado con *React*.

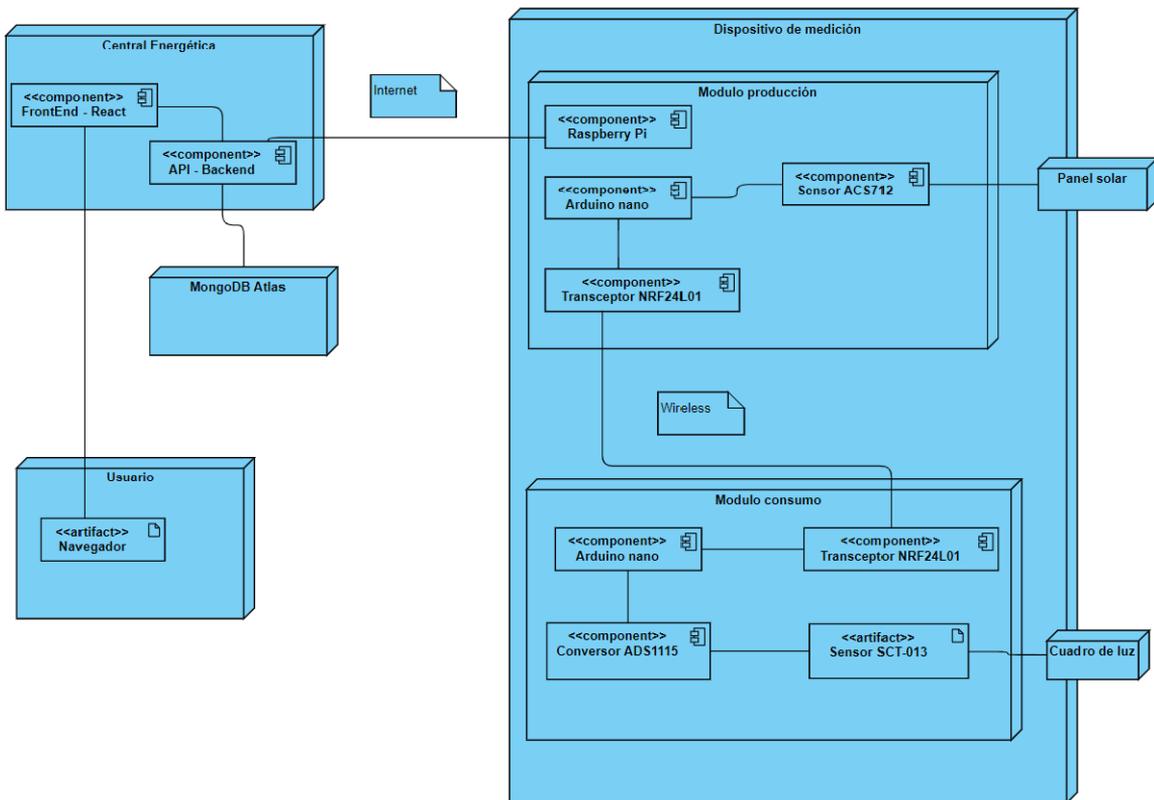


Figura 4.1: Diagrama de despliegue

Capa de lógica de negocio

La siguiente capa es la que controla la lógica del negocio. Esta capa gestiona las operaciones necesarias con los datos del usuario, implementa seguridad, control de excepciones... Esta capa es la *gateway* para el acceso a la base de datos.

Su correspondencia a nivel de diseño sería la API REST realizada con *Node.js* y *Express*.

Capa de lógica de datos

Esta es la capa que maneja la persistencia de la aplicación. Únicamente recibe y sirve datos requeridos por la capa de lógica de negocio, separando así responsabilidades.

Su correspondencia a nivel de implementación sería el nodo *MongoDB Atlas*.

4.2. Interfaz gráfica

El sistema cuenta con la existencia de dos tipos de usuario, como consecuencia vamos a tener 2 interfaces principales, una para administradores y otra para usuarios.

IU: Usuario estándar

En esta sección vamos a plantear el diseño de la interfaz para el usuario estándar.

El objetivo es mantener una interfaz uniforme modificando unicamente los componentes centrales de la pantalla. Para ello todas las interfaces de usuario estándar deben implementar un componente en común, que denominaremos, *UserHeader.js*, donde se definirá el menu superior con botón para menu desplegable y el *drawer* lateral con menu de navegación.

Comenzaremos por la página principal a la que el usuario accede cuando se identifica correctamente en el sistema: “Tiempo real” (Véase figura 4.2).

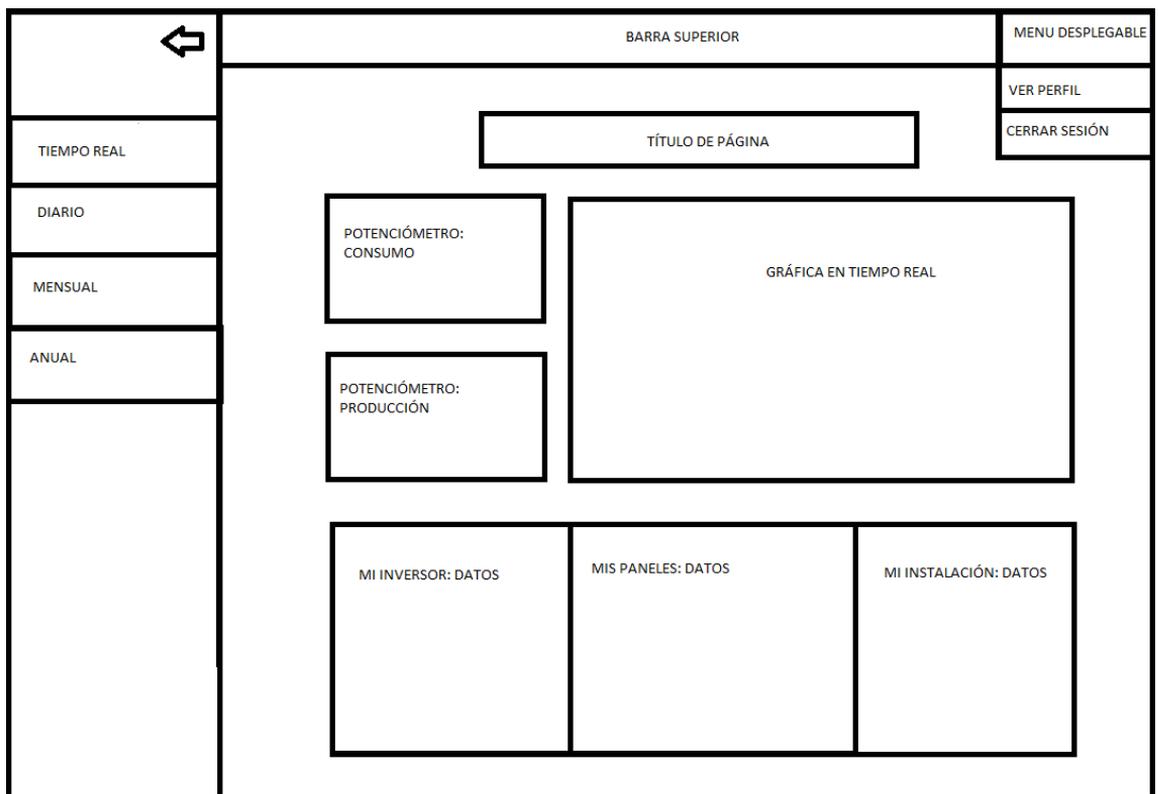


Figura 4.2: Interfaz usuario estándar: Tiempo real

Posteriormente tenemos 3 páginas cuya interfaz no varía, solo varía la funcionalidad. Serían las páginas que muestran los históricos por días, meses y años. La única diferencia a nivel visual, es el selector de fecha. En el caso del histórico diario dicho selector permite seleccionar año, mes y día. En el caso del histórico mensual el selector permite seleccionar año y mes, pero no día, y en el caso del histórico anual solo permite seleccionar año. (Véase figura 4.3).

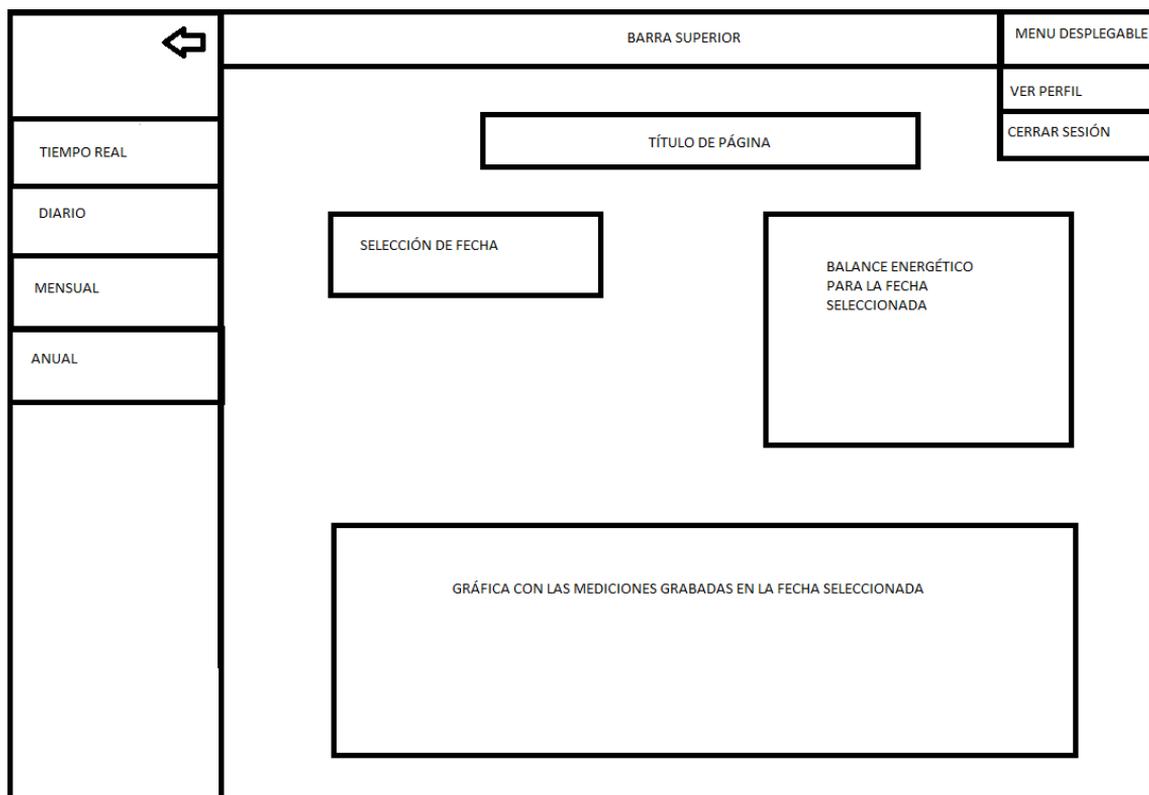


Figura 4.3: Interfaz usuario estándar: Históricos

Y por último, en la figura 4.4 tendríamos la página de los datos de usuario: Perfil.

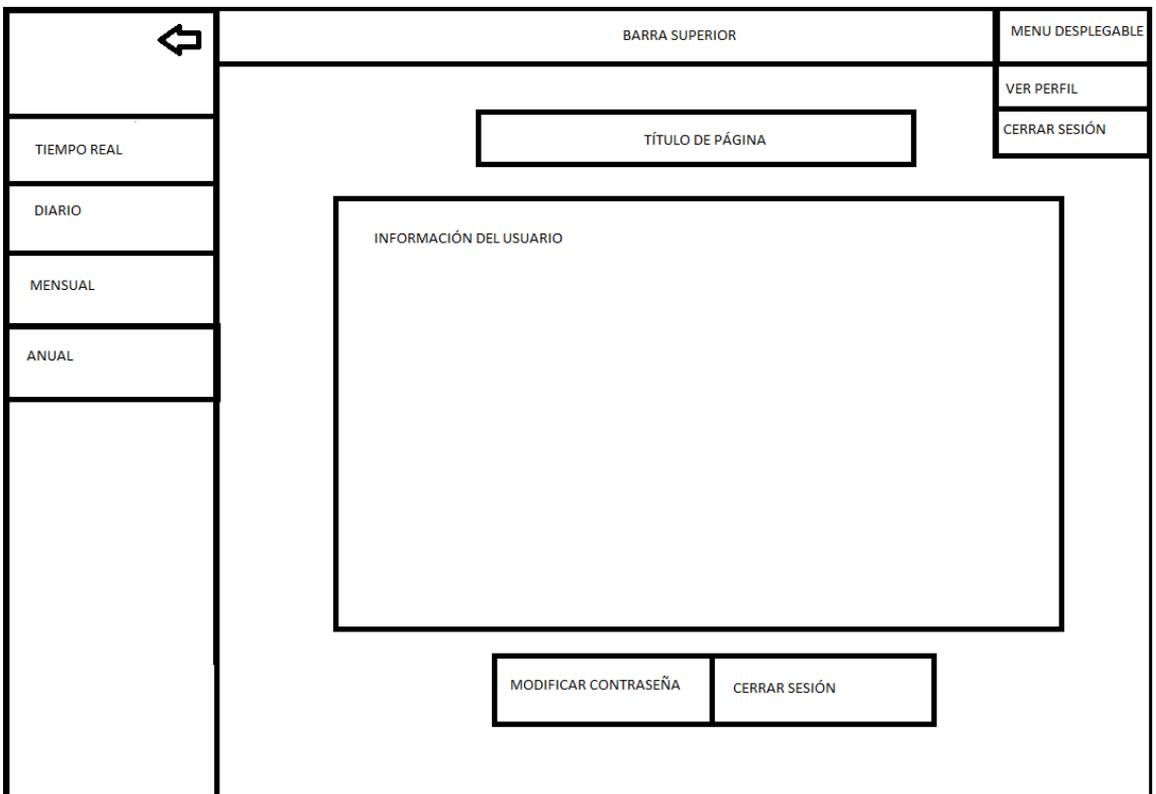


Figura 4.4: Interfaz usuario estándar: Perfil

IU: Administrador

El caso del usuario administrador es aún mas sencillo a la hora de diseñar interfaces, ya que se repite el diseño por cada entidad que el sistema maneja (Administradores, Usuarios, Inversores, Paneles e Instalaciones). Básicamente hay 2 tipos de interfaz para el administrador:

- Interfaz que muestra una tabla con los datos en BBDD.
- Interfaz que muestra un formulario para insertar nuevos datos

Comenzaremos con el caso de una interfaz que muestra una tabla de datos. (Véase la figura 4.5).

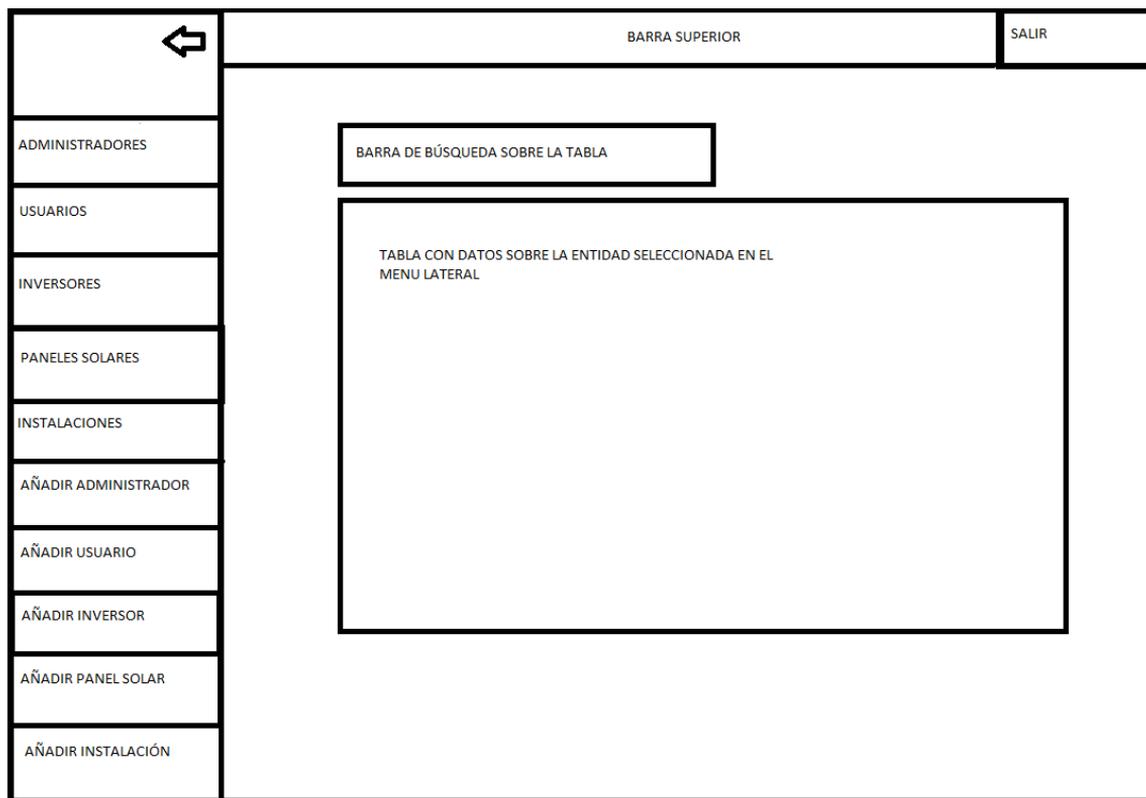


Figura 4.5: Interfaz usuario administrador: Ver datos de entidad

4.2. INTERFAZ GRÁFICA

En estas tablas, por cada tupla de datos se añaden dos botones: “Eliminar” y “Modificar”. En caso de seleccionar la opción “Modificar” se despliega un formulario con los datos de la tupla seleccionada en un diálogo *pop-up*, con dos botones: “Modificar” y “Cancelar”. En caso de seleccionar la opción “Eliminar”, se muestra un dialogo de confirmación *pop-up* con dos botones: “Confirmar” y “Cancelar”.

A continuación se muestra el diseño de la interfaz para páginas de creación de datos. (Véase la figura 4.6).

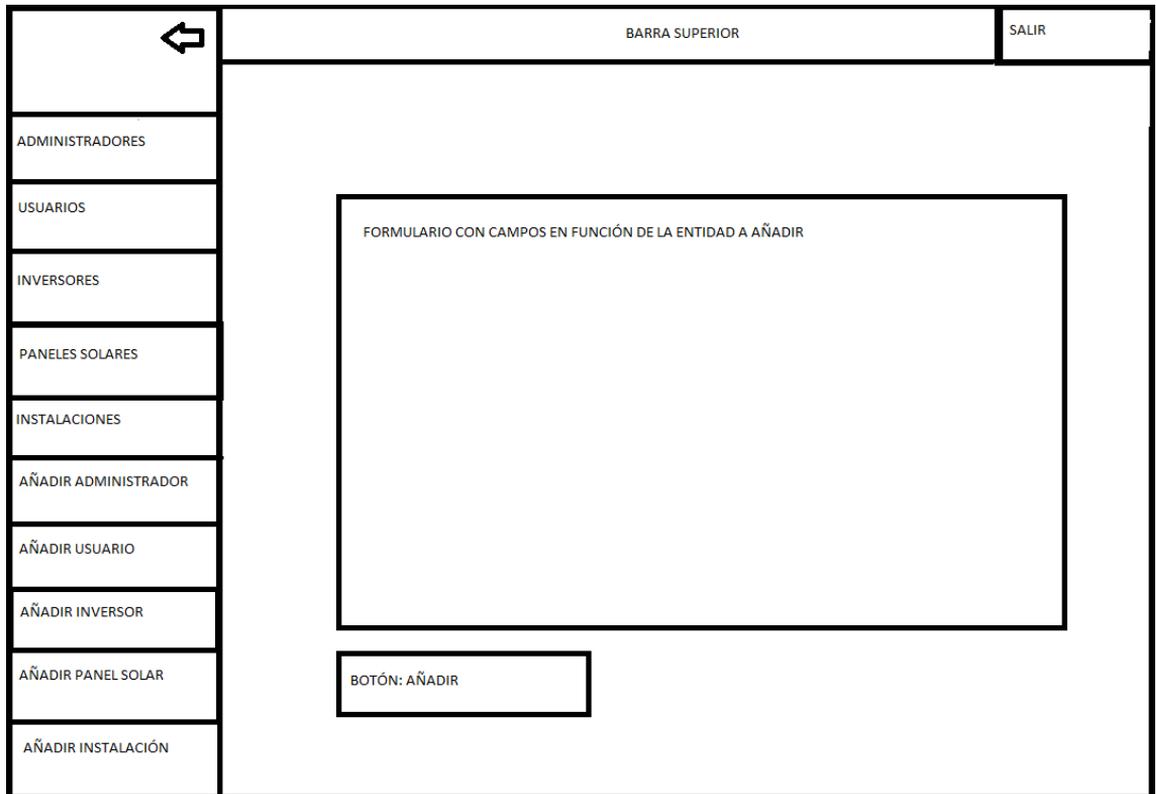


Figura 4.6: Interfaz usuario administrador: Añadir entidad

Capítulo 5

Implementación

5.1. API REST

En esta sección vamos a ver a la estructura de la API, y los diferentes ficheros que la componen.

5.1.1. Estructura

Podemos ver la estructura en la figura 5.1. Vamos a detallar cada subdirectorio y la función que realiza.

- **Node_Modules:** En este directorio se almacenan las librerías utilizadas por la aplicación. El gestor NPM es el encargado de controlar las dependencias.
- **Sessions:** Directorio creado automáticamente por node.
- **Src:** Directorio principal donde se localizan los recursos.
 1. **Controllers:** En este directorio se localizan los controladores de cada objeto que maneja la aplicación. Dichos controladores realizan las operaciones necesarias, es decir, controlan la lógica de negocio. Entendemos por objeto las entidades con las que vamos a trabajar: usuario, instalación, inversor...
 2. **Lib:** En este directorio alojamos aquellas librerías propias, que son reutilizadas en diferentes partes de la aplicación.
 3. **Middlewares:** En este directorio se localizan los métodos de autenticación para obtener acceso a recursos.

4. Mongo: En este directorio se definen los *Schemas* o modelos de las entidades que vamos a almacenar en MongoDB.
 5. Routes: En este directorio se localizan las URI's para acceder a los diferentes recursos
- Index.js: Es el archivo principal de la aplicación donde se define el servidor y se configuran los parámetros.
 - Env: En este archivo se almacenan las variables de entorno del servidor.
 - Env.Example: Este archivo muestra un ejemplo del formato de almacenamiento de las variables de entorno.
 - GitIgnore: En este archivo se declaran los archivos de los que no se quiere llevar cuenta en el control de versiones.
 - Prettierc: Este es un archivo de configuración para el embellecedor de código “Prettier”, donde se definen parámetros para ajustar el formato.
 - Package.json: En este archivo se enumeran las dependencias para que el gestor pueda descargarlas cuando instalamos el servidor.
 - Package-lock.json: En este archivo se enumeran dependencias que no deben quedar tras el período de desarrollo del servidor.

A continuación vamos a detallar los directorios mas significativos.

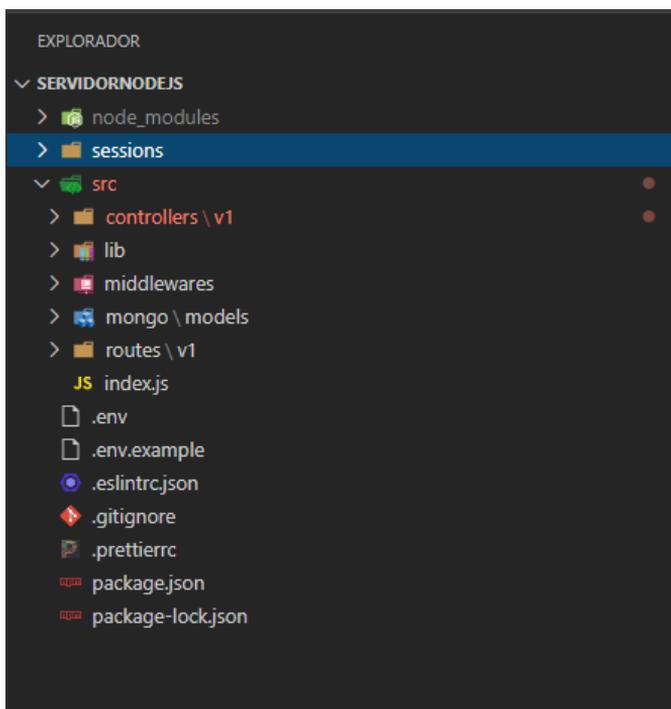


Figura 5.1: Estructura de la API.

5.1.2. Controllers

En el directorio *Controllers*, podemos encontrar el subdirectorio *v1*, y dentro los controladores pertenecientes a dicha versión inicial. De esta manera podemos mantener mejor el control de versiones.

Como se aprecia en la figura 5.2 dicho directorio contiene un controlador por cada entidad que se almacena en MongoDB.



Figura 5.2: Controllers.

5.1.3. Mongo

Dentro del directorio *Mongo* nos encontramos con el directorio *Models*. En cada uno de los ficheros incluidos en dicho subdirectorio podemos encontrar los esquemas que definen las estructuras de datos que queremos almacenar. En dichos esquemas, posteriormente se expondrá el código fuente, se definen los tipos de datos que componen una estructura, sus atributos, y otras opciones como si el atributo es requerido y/o único.

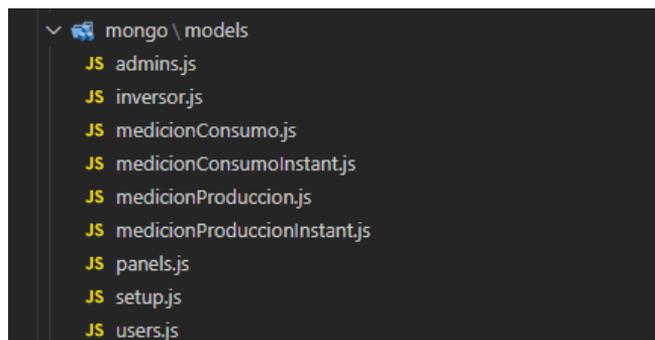


Figura 5.3: Mongo.

5.1.4. Routes

Dentro del directorio *Routes* nos encontramos con el directorio *v1* y dentro con las rutas para dicha versión inicial. Cada entidad manejada por el sistema contiene diferentes URI's para realizar diferentes acciones. Por ejemplo: Para la entidad "Users" tenemos *users-routes.js*. En este documento se definen las URI's para el login, la creación de usuarios, su eliminación...

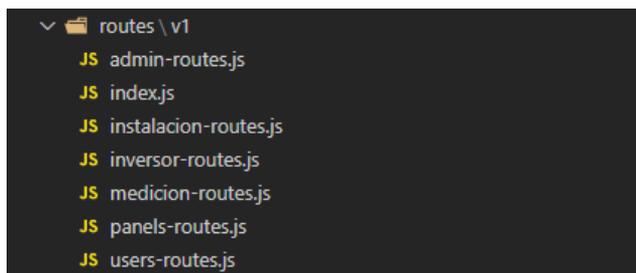


Figura 5.4: Routes.

Este directorio contiene un archivo llamado *index.js*, este sirve como punto de centralización de la información, en él se define el inicio de la URI y en cada archivo de ruta se complementa la URI. Véase la figura 5.5.

```
src > routes > v1 > JS index.js > ...
You, 24 days ago | 1 author (You)
1  const usersRoutes = require('./users-routes');
2  const adminRoutes = require('./admin-routes');
3  const instalacionesRoutes = require('./instalacion-routes');
4  const medicionRoutes = require('./medicion-routes');
5  const inversorRoutes = require('./inversor-routes');
6  const panelsRoutes = require('./panels-routes');
7
8
9  module.exports = (app) => {
10   app.use('/api/v1/users', usersRoutes);
11   app.use('/api/v1/admins', adminRoutes);
12   app.use('/api/v1/instalaciones', instalacionesRoutes);
13   app.use('/api/v1/mediciones', medicionRoutes);
14   app.use('/api/v1/inversor', inversorRoutes);
15   app.use('/api/v1/panels', panelsRoutes);
16 };
17
```

Figura 5.5: Routes: index.js

5.2. Front-End

En este apartado vamos a detallar la estructura de archivos que componen el front-end para la visualización de datos.

5.2.1. Estructura

Véase la figura 5.6 para ver la estructura de una aplicación desarrollada con React. En los siguientes apartados vamos a detallar los directorios mas significativos, pues varios de ellos son creados automáticamente por React y no he tenido que intervenir en ellos.

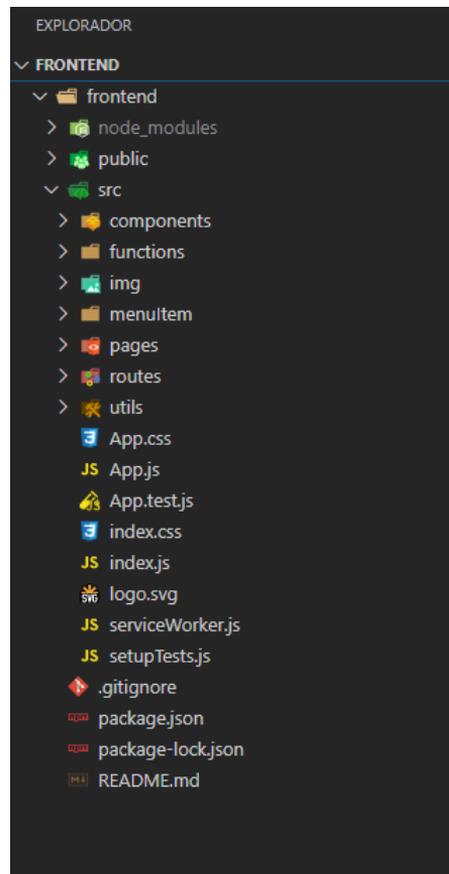


Figura 5.6: Estructura aplicación React

- **Node_Modules:** En este directorio se almacenan las librerías utilizadas por la aplicación. El gestor NPM es el encargado de controlar las dependencias.
- **Publics:** Definición del archivo *robots.txt*.
- **Src:** Directorio principal donde se localizan los recursos.
 1. **Components:** En el paradigma del desarrollo con React, todo funciona con componentes reutilizables. En este directorio definidos dichos componentes, como pueden ser los selectores de fecha, las cabeceras, los menús...
 2. **Functions:** En este directorio alojamos las funciones que se repiten a lo largo del sistema. Sería el caso de las llamadas a la API, tareas como borrar una entidad, modificarla, crearla...
 3. **Img:** Directorio que almacena las imagenes utilizadas.
 4. **MenuItem:** En este directorio se aloja la lista de elementos que componen los menús. Es una forma de mejorar la eficiencia en el código.
 5. **Pages:** En este directorio se localizan las vistas al usuario, es decir las páginas que va a visualizar y que requieren de componentes y funciones.
 6. **Routes:** Directorio que aloja las cadenas de rutas.
 7. **Utils:** Directorio para almacenar los colores utilizados.
- **App.js:** Es el archivo principal de la aplicación. En el se establecen las rutas y las páginas que componen la web así como las propiedades que se transmiten entre páginas.
- **Index.js:** Es el archivo de despliegue para *App.js*.
- **GitIgnore:** En este archivo se declaran los archivos de los que no se quiere llevar cuenta en el control de versiones.
- **Package.json:** En este archivo se enumeran las dependencias para que el gestor pueda descargarlas cuando instalamos el servidor.
- **Package-lock.json:** En este archivo se enumeran dependencias que no deben quedar tras el período de desarrollo del servidor.

5.2.2. Componentes

A continuación vamos a detallar los componentes utilizados para el desarrollo del proyecto, todos ellos alojados en el directorio *Components*. (Véase al figura 5.7).

- **AdminHeader.js:** Componente para la cabecera de la interfaz del usuario administrador.

- ApexChartBar.js: Componente para crear gráficas de barras.
- ApexChartDiary.js Componente para crear gráficas multi-línea.
- ApexChartLine.js Componente para crear gráficas multi-línea de meses y años.
- ApexChartLive.js: Componente para crear gráficas dinámicas.
- ApexRadial.js: Componente para crear los potenciómetros.
- DatePicker.js: Componente versionado de “react-datepicker” para la obtención de datos de la API tras seleccionar fecha.
- RealTimeChart.js: Componente para interfaz de usuario estándar: Tiempo real.
- Title.js: Componente versionado de “react-typography” para títulos.
- UserHeader.js: Componente para la cabecera de la interfaz del usuario estándar.

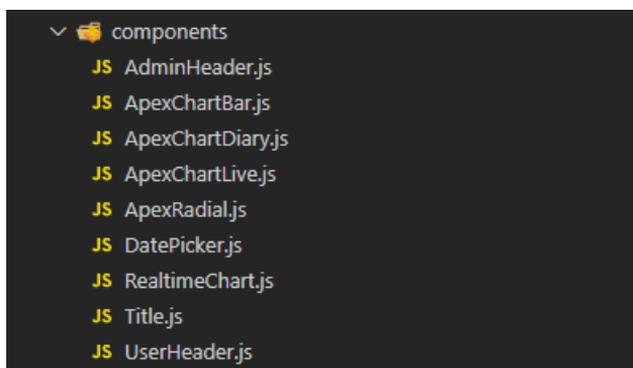


Figura 5.7: React: Componentes utilizados

5.2.3. Rutas

En este directorio se definen todas las rutas que utiliza la aplicación.

Haremos una distinción entre rutas internas, externas y rutas contra la API. (Véase la figura 5.8).

- Ruta interna: Direccionamiento entre páginas dentro de la aplicación.
- Ruta externa: Puntos de enlace para la obtención de datos de API's de terceros.
- Ruta API: Puntos de enlace para la obtención de datos de nuestra API.

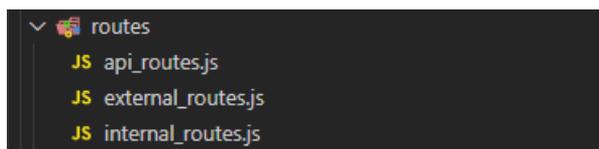


Figura 5.8: React: Rutas

5.2.4. Funciones

En este directorio se encuentra todas las funciones que se utilizan en la aplicación. (Véase figura 5.9). La mayoría se explican por su propio nombre, por ello procederemos a explicar las menos intuitivas:

- `ExpiredSession.js`: Función llamada cuando el tiempo de sesión de usuario, es decir expira el JSON Web Token. Elimina los datos de sesión y redirecciona a la página de identificación.
- `Validations.js`: Función llamada para validar campos de formularios, véase NIF/NIE, emails, contraseñas, etc.

5.3. Raspberry Pi Zero W: Daemon

En esta sección del documento vamos a explicar ligeramente por encima la función de dicho programa. En los *Anexas* podrá visualizar el código fuente íntegro con comentarios.

Como ya hemos comentado anteriormente, la función de este programa es la de recolectar medidas y enviarlas a la API.

Básicamente es un bucle infinito, donde se comprueba la conexión con el puerto serie; si dicha conexión es exitosa, se prepara un buffer para la recepción de un array de bytes, contenedor de las medidas enviadas por el *Arduino* y se envía dicha información a la API. Si dicha conexión no es dada, repite el intento de conexión cada 5 segundos.

Cuando el programa recibe un dato correctamente, realiza 2 cometidos:

- Envía una medida instantánea a la API, para poder realizar una gráfica en tiempo real.
- Envía una medida cada minuto, promediando las medidas instantáneas para realizar los históricos por días, meses y años.

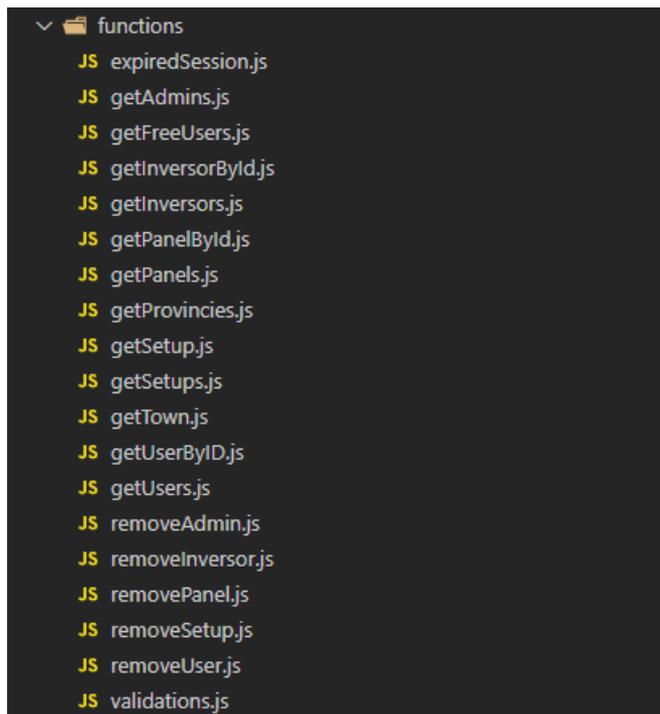


Figura 5.9: React: Funciones

Las medidas originalmente tomadas son de tipo *float*, pero para su transmisión por el puerto serie han de ser transformadas a tipo *Byte*. Esto nos obliga a añadir una suma de comprobación en cada transmisión de datos, posteriormente se detallará el protocolo de comunicación entre dispositivos.

5.4. Protocolo de comunicación: Arduino Nano - Raspberry Pi Zero W

Se ha diseñado un sencillo protocolo para la comunicación de mensajes entre dispositivos. Véase el cuadro 5.1.

Dato 1	Dato 2	Dato 3	Código	Checksum
--------	--------	--------	--------	----------

Cuadro 5.1: Protocolo de comunicación

El tamaño de cada trama es de 14 Bytes, cada dato ocupa 4 Bytes, el código 1 Byte y la suma de comprobación 1 Byte. En el cuadro 5.2 podemos ver los diferentes códigos.

Tipo	Valor
Medición de consumo	Q
Medición de producción	P

Cuadro 5.2: Códigos

Para realizar la suma de comprobación, se suman el valor de todos los bytes de la trama y se añade dicha suma al final de la trama.

5.5. Arduino Nano

En esta sección vamos a explicar brevemente la funcionalidad del código de los arduinos, posteriormente se expondrá el código fuente con comentarios sobre el mismo.

El cometido de estos dispositivos es capturar dos datos de la red eléctrica: Intensidad y Tensión. Con ellos calcularemos la potencia instantánea, potencia aparente y coseno de phi, este último necesario para calcular la potencia activa en un futuro.

El programa es un bucle infinito, que toma medidas de manera continua, y cada minuto calcula el promedio y lo envía al siguiente dispositivo en la cadena de comunicación para llegar hasta la API.

Llegados a este punto podemos diferenciar cada tipo de arduino según su función.

Emisor

Dicho arduino es el componente del módulo consumo, toma medidas tras el cuadro de luz del hogar, las promedia, y las envía al otro arduino llamado “Receptor” a través del chip *nRF24L01*.

Receptor

Dicho arduino es el componente del módulo producción, toma medidas tras el inversor de la instalación solar, las promedia, y las envía a la API. A mayores, es el encargado de recibir las tramas del “Emisor” y enviarlas a la *Raspberry Pi Zero W* a través del puerto serie.

Capítulo 6

Diseño Hardware

6.1. Introducción

Tras analizar el problema a resolver, ¿Cómo poder medir la intensidad y/o corriente que pasa a través de un cable? Se comenzó a realizar un estudio de mercado abordando distintos tipos de microcontroladores, sensores compatibles, invasivos, no invasivos... Todo ello para elaborar el diseño hardware del dispositivo de medición.

Aunque haya sido mencionado con anterioridad, merece la pena resaltar que el proyecto se ha desarrollado durante el período de alarma nacional por la pandemia del COVID-19 (2020), por lo que, el equipo de desarrollo se vió en el compromiso de utilizar los materiales que tenía a su rápida disposición, y no aquellos materiales que hubieran sido deseables.

A continuación vamos a realizar un pequeño análisis sobre los componentes necesarios para nuestro proyecto. Para ver la guía de montaje de los circuitos diríjase al *Apendice C*.

6.2. Arduino

6.2.1. ¿Qué es?

Arduino es una plataforma de creación electrónica de código abierto.

El hardware del dispositivo consiste en una placa con un microcontrolador (Atmel AVR) y puertos de entrada y salida. Existen diferentes modelos con diferentes especificaciones donde varían: el tipo de microcontrolador, frecuencia de reloj, memoria flash, SRAM o EEPROM.

6.2.2. Comparando dispositivos

A continuación se realiza una pequeña comparativa, en el cuadro 6.1, entre diferentes modelos de Arduino. El objetivo es comprobar cual se adaptar mejor a nuestra situación.

	UNO	Leonardo	Mega	Mini	Nano
I/O Digital	16	20	54	14	14
In Analógica	6	12	16	6	8
Out PWR	6	7	14	8	6
UART	1	1	4	1	1
Memoria	32kb	32kb	256kb	32kb	32kb
Precio	6 €	10 €	12 €	2 €	2 €

Cuadro 6.1: Comparativa Arduinos

En este proyecto utilizaremos el modelo: Arduino Nano.

6.2.3. Arduino Nano

En este proyecto vamos a utilizar la versión mas simplificada de Arduino, cuyas especificaciones son suficientes para la labor:

- Microcontrolador: ATmega328 con cargador de inicio preprogramado.
- Tension de entrada (límite): +6V a +20V
- Tensión de entrada (Recomendado): +7 a +12V
- Pines Digitales I/O: 14 (6 están provistos de salida PWM)
- Pines analógicos de entrada: 6
- DC para Pines I/O: 40mA
- Memoria Flash: 32KB (2 KB para cargador de inicio (bootloader))
- SRAM: 2KB
- EEPROM: 1KB
- Frecuencia de reloj: 16MHz
- Dimensiones: 18.5 mm x 43.2 mm

En la bibliografía se incluye el *Datasheet* del dispositivo.

6.3. Raspberry Pi

6.3.1. ¿Qué es?

Raspberry Pi es básicamente un ordenador de placa reducida de bajo costo. El proyecto es llevado a cabo por Raspberry Pi Foundation y tiene como objetivo estimular la enseñanza de informática.

No podemos entender Raspberry Pi como un hardware privado, en su web oficial explican que disponen de contratos de distribución y venta con varias empresas, pero al mismo tiempo cualquier empresa y/o entidad podría hacerse redistribuidor de tarjetas Raspberry, por lo que se entiende, que estamos ante un producto de propiedad registrada pero de uso libre tanto a nivel educativo como particular.

Sin embargo, el software si es de código abierto. El sistema operativo básico es una adaptación de *Debian*, llamada Raspbian, aunque puede funcionar igualmente con *Windows 10*.

6.3.2. Comparando dispositivos

A continuación, en el cuadro 6.2, se realiza una pequeña comparativa entre diferentes modelos de Raspberry. El objetivo es comprobar cual se adaptar mejor a nuestra situación.

	rPi 3B	rPi 2B	rPi 1B+	rPi Zero W
CPU	ARMv8	ARM Cortex A7	ARM1176JZF-S	ARM1 BCM2835
GPU	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV
RAM	1GB	1GB	512MB	512MB
USB	USB2.0 x4	USB2.0 x4	USB2.0 x4	Micro USB
Ethernet	SI	SI	SI	NO
WiFi	NO	NO	NO	SI
Bluetooth	NO	NO	NO	BLE 4.1
Consumo	4 W	4 W	3 W	1 W
Precio	40 €	32 €	23 €	15 €

Cuadro 6.2: Comparativa Raspberries

6.3.3. Raspberry Pi Zero W

Para el desarrollo del proyecto vamos a contar con la versión *Zero W* de Raspberry Pi. Este modelo fue sacado a la venta en 2015 como celerabración por los 5 años del proyecto.

El dispositivo cuenta con la siguientes especificaciones:

- CPU: Broadcom BCM2835 1Ghz: Un procesador mononúcleo que incorpora una potencia de un 40 % superior a la Raspberry Pi 1.
- Memoria RAM de 512MB LPDDR2.
- Lector para tarjetas Micro-SD.
- Dos conexiones micro-USB para alimentación e Intercambio de Datos.
- 40 Pines de conexión GPIO externos iguales que los modelos A+,B+ Y 2B.
- Salida Mini-HDMI para audio y vídeo hasta 1080P.
- GPU Dual Core VideoCore IV con 512MB de RAM.
- Dimensiones: 65MM x 30mm x .
- Bluetooth 4.
- Modulo WiFi: Cypress CYW43438.

6.4. Otros componentes

6.4.1. NRF24L01

El NRF24L01 es un chip de comunicación inalámbrica fabricado por *Nordic Semiconductor* que podemos utilizar con nuestro Arduino Nano.

El NRF24L01 integra un transceptor RF (transmisor + receptor) a una frecuencia entre 2.4GHz a 2.5GHz, una banda libre para uso gratuito. La velocidad de transmisión es configurable entre 250 Kbps, 1Mbps, y 2 Mbps y permite la conexión simultánea con hasta 6 dispositivos [1].

El NRF24L01 también incorpora la lógica necesaria para que la comunicación sea robusta, como corrección de errores y reenvío de datos si es necesario, liberando de esta tarea al procesador. El control del módulo se realiza a través de bus SPI, por lo que es sencillo controlarlo desde un procesador como Arduino.

La banda de frecuencia es de 2400 a 2525 MHz, pudiendo elegir entre 125 canales espaciados a razón de 1MHz. Se recomienda usar las frecuencias de 2501 a 2525 MHz para evitar interferencias con las redes Wifi.

6.4.2. ADS1115

El ADS1115 es un convertor analógico-digital. Algunos modelos de arduino cuentan con un convertor analógico-digital interno, ¿Por qué añadir uno externamente? Los convertidores internos manejan hasta 10 bits de precisión, mientras que el ADS1115 tiene una resolución de 16 bits [2]. Obtenemos una mayor precisión en la medida y liberamos la CPU de carga de trabajo.

6.4.3. ACS712

El ACS712 es un sensor de corriente tanto alterna como continua, que permite medir la intensidad eléctrica que atraviesa un conductor [38]. Será el componente esencial, conjunto al arduino, para medir la intensidad y/o potencia consumida por una carga.

El dispositivo esta formado por un sensor de efecto Hall de precisión y bajo offset junto con un canal de conducción localizado cerca de la superficie del chip. Cuando la corriente fluye por el canal genera un campo magnético, detectable por el sensor Hall, y es convertido en una tensión.

6.4.4. FZ0430

El FZ0430 es un módulo comercial capaz de medir tensiones de hasta 25V para un procesador con alimentación a 5V, y de hasta 16V para uno de 3.3V.

Básicamente es un divisor de tensión con resistencias de 30k y 7.5k, por lo tanto este componente podría ser sustituido por el circuito análogo formado por las resistencias descritas.

Dicho componente en combinación con un transformador (220V/15V) nos permite tomar mediciones sobre corriente alterna a 220V. Es necesaria la calibración de dichos sensores, pues los valores de fábrica pueden estar ligeramente modificados [5].

6.4.5. Transformador

Denominamos transformador a un elemento eléctrico que permite aumentar o disminuir la tensión en un circuito eléctrico de corriente alterna, manteniendo la potencia. Como comentamos anteriormente, en combinación con el divisor de tensión, nos permitirá tomar medidas de tensión [26].

6.5. Circuitos

Las conexiones necesarias entre dispositivos se pueden ver en el *Apéndice C*. Dicho apéndice incluye imágenes de los circuitos montados por el alumno (Véase sección C.3).

6.6. Consideraciones

El diseño detallado en las secciones anteriores se corresponde con la versión inicial del dispositivo de medición. En posteriores versiones se implementará un cambio básico para llevar este dispositivo a una futura “fase de producción”.

Dicho cambio se corresponde con la sustitución de los sensores de toma de intensidad de efecto hall (invasivos) por sensores de pinza amperimétrica (no invasivos) evitando tener que “conectarnos” a la red doméstica. La decisión de usar el sensor efecto hall fue obligada debido al estado de alarma, pues dicho sensor se encontraba en disposición del equipo técnico de *Energética Coop*. A nivel de software dicho cambio es irrelevante, por lo que el alumno no se ha visto influido.

Capítulo 7

Pruebas

7.1. Pruebas de caja blanca

En programación, denominamos pruebas *de caja blanca* a un tipo específico de pruebas que se realiza sobre las funciones internas del sistema, es decir, sirven para comprobar el correcto funcionamiento de mecanismos y métodos que llevan a cabo acciones en el sistema [41].

En esta sección no vamos a detallar este tipo de pruebas, ya que han sido realizadas paralelamente al desarrollo del sistema y no resultan significativas para el entendimiento del sistema.

7.2. Pruebas de caja negra

Según la *teoría de sistemas y física*, entendemos *caja negra* como un elemento que se estudia desde el punto de vista de las entradas que recibe y los resultados que produce, sin darle importancia a los mecanismos internos que dan lugar a dicho resultado [3]. Este tipo de pruebas son esenciales para llevar a cabo la comprobación de los requisitos funcionales del sistema.

A continuación se detalla la batería de pruebas llevada a cabo para comprobar la funcionalidad del sistema.

7.2. PRUEBAS DE CAJA NEGRA

PCN-01	Iniciar sesión
Objetivo	Identificarse
Actor	Administrador, Usuario
Precondición	El usuario ha de estar registrado.
Acción esperada	Una vez completado el formulario de login, el sistema debe redirigir a la página principal
Resultado	Positivo

Cuadro 7.1: Prueba de caja negra: Iniciar sesión

PCN-02	Crear administrador
Objetivo	Crear un nuevo usuario tipo administrador
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Administradores</i> .
Acción esperada	Tras completar el formulario de creación, dar al boton “Crear administrador”, se debe ver un mensaje que dice ”Usuario creado correctamente”, y el sistema limpiará el formulario.
Resultado	Positivo

Cuadro 7.2: Prueba de caja negra: Crear administrador

PCN-03	Crear usuario
Objetivo	Crear un nuevo usuario tipo estándar
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Usuarios</i> .
Acción esperada	Tras completar el formulario de creación, dar al boton “Crear cuenta”, se debe ver un mensaje que dice ”Usuario creado correctamente”, y el sistema limpiará el formulario.
Resultado	Positivo

Cuadro 7.3: Prueba de caja negra: Crear usuario

PCN-04	Crear inversor
Objetivo	Crear un nuevo inversor
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Inversores</i> .
Acción esperada	Tras completar el formulario de creación, dar al boton “Añadir inversor”, se debe ver un mensaje que dice ”Inversor añadido correctamente”, y el sistema limpiará el formulario.
Resultado	Positivo

Cuadro 7.4: Prueba de caja negra: Crear inversor

PCN-05	Crear panel solar
Objetivo	Crear un nuevo panel solar
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Paneles</i> .
Acción esperada	Tras completar el formulario de creación, dar al boton “Añadir panel”, se debe ver un mensaje que dice ”Panel añadido correctamente”, y el sistema limpiará el formulario.
Resultado	Positivo

Cuadro 7.5: Prueba de caja negra: Crear panel solar

PCN-06	Crear instalación
Objetivo	Crear una nueva instalación
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Instalaciones</i> .
Acción esperada	Tras completar el formulario de creación, dar al boton “Crear instalación”, se debe ver un mensaje que dice ”Instalación creada correctamente”, y el sistema limpiará el formulario.
Resultado	Positivo

Cuadro 7.6: Prueba de caja negra: Crear instalación

7.2. PRUEBAS DE CAJA NEGRA

PCN-07	Modificar administrador
Objetivo	Modificar los datos de un usuario tipo administrador
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Administradores</i> .
Acción esperada	El usuario hará click sobre el usuario que desee, de los mostrados en el listado. Se le mostrará un formulario con los datos para actualizar, tras completar, se hará click en "Modificar". El usuario mostrará un mensaje "Usuario actualizado correctamente" y refrescará el listado de usuarios.
Resultado	Positivo

Cuadro 7.7: Prueba de caja negra: Modificar administrador

PCN-08	Modificar usuario
Objetivo	Modificar los datos de un usuario tipo estándar
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Usuarios</i> .
Acción esperada	El usuario hará click sobre el usuario que desee, de los mostrados en el listado. Se le mostrará un formulario con los datos para actualizar, tras completar, se hará click en "Modificar". El usuario mostrará un mensaje "Usuario actualizado correctamente" y refrescará el listado de usuarios.
Resultado	Positivo

Cuadro 7.8: Prueba de caja negra: Modificar usuario

PCN-09	Modificar inversor
Objetivo	Modificar los datos de un inversor
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Inversores</i> .
Acción esperada	El usuario hará click sobre el inversor que desee, de los mostrados en el listado. Se le mostrará un formulario con los datos para actualizar, tras completar, se hará click en "Modificar". El usuario mostrará un mensaje "Inversor actualizado correctamente" y refrescará el listado de inversores.
Resultado	Positivo

Cuadro 7.9: Prueba de caja negra: Modificar inversor

CAPÍTULO 7. PRUEBAS

PCN-10	Modificar panel solar
Objetivo	Modificar los datos de un panel solar
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Paneles</i> .
Acción esperada	El usuario hará click sobre el panel que desee, de los mostrados en el listado. Se le mostrará un formulario con los datos para actualizar, tras completar, se hará click en "Modificar". El usuario mostrará un mensaje "Panel actualizado correctamente" y refrescará el listado de paneles.
Resultado	Positivo

Cuadro 7.10: Prueba de caja negra: Modificar panel solar

PCN-11	Modificar instalación
Objetivo	Modificar los datos de una instalación
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Instalaciones</i> .
Acción esperada	El usuario hará click sobre la instalación que desee, de las mostradas en el listado. Se le mostrará un formulario con los datos para actualizar, tras completar, se hará click en "Modificar". El usuario mostrará un mensaje "Instalación actualizada correctamente" y refrescará el listado de instalaciones.
Resultado	Positivo

Cuadro 7.11: Prueba de caja negra: Modificar instalación

PCN-12	Eliminar administrador
Objetivo	Elimina un usuario administrador.
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Administradores</i> .
Acción esperada	El usuario hará click sobre el usuario que desee, de los mostrados en el listado. Posteriormente hará click en el icono de la papelera. El sistema pedirá confirmación para eliminar la entidad. Tras confirmar, se desvanece el dialogo de confirmación y se refresca el listado.
Resultado	Positivo

Cuadro 7.12: Prueba de caja negra: Eliminar administrador

7.2. PRUEBAS DE CAJA NEGRA

PCN-13	Eliminar usuario
Objetivo	Elimina un usuario estándar.
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Usuarios</i> .
Acción esperada	El usuario hará click sobre el usuario que desee, de los mostrados en el listado. Posteriormente hará click en el icono de la papelera. El sistema pedirá confirmación para eliminar la entidad. Tras confirmar, se desvanece el dialogo de confirmación y se refresca el listado.
Resultado	Positivo

Cuadro 7.13: Prueba de caja negra: Eliminar usuario

PCN-14	Eliminar inversor
Objetivo	Elimina un inversor.
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Inversores</i> .
Acción esperada	El usuario hará click sobre el inversor que desee, de los mostrados en el listado. Posteriormente hará click en el icono de la papelera. El sistema pedirá confirmación para eliminar la entidad. Tras confirmar, se desvanece el dialogo de confirmación y se refresca el listado.
Resultado	Positivo

Cuadro 7.14: Prueba de caja negra: Eliminar inversor

PCN-15	Eliminar panel solar
Objetivo	Elimina un panel solar.
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Paneles</i> .
Acción esperada	El usuario hará click sobre el panel que desee, de los mostrados en el listado. Posteriormente hará click en el icono de la papelera. El sistema pedirá confirmación para eliminar la entidad. Tras confirmar, se desvanece el dialogo de confirmación y se refresca el listado.
Resultado	Positivo

Cuadro 7.15: Prueba de caja negra: Eliminar panel

CAPÍTULO 7. PRUEBAS

PCN-16	Eliminar instalación
Objetivo	Elimina una instalación.
Actor	Administrador
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Instalaciones</i> .
Acción esperada	El usuario hará click sobre la instalación que desee, de las mostradas en el listado. Posteriormente hará click en el icono de la papelera. El sistema pedirá confirmación para eliminar la entidad. Tras confirmar, se desvanece el dialogo de confirmación y se refresca el listado.
Resultado	Positivo

Cuadro 7.16: Prueba de caja negra: Eliminar instalación

PCN-17	Cerrar sesión
Objetivo	Cerrar sesión de usuario. Abandonar aplicación.
Actor	Administrador, Usuario
Precondición	El usuario ha de estar registrado e identificado.
Acción esperada	El usuario hará click sobre el boton "Cerrar sesión". El sistema borrara los datos de sesión y redirigirá al usuario a la página de inicio de sesión.
Resultado	Positivo

Cuadro 7.17: Prueba de caja negra: Cerrar sesión

PCN-18	Ver perfil
Objetivo	Ver los datos propios de un usuario.
Actor	Usuario
Precondición	El usuario ha de estar registrado e identificado.
Acción esperada	El usuario hará click sobre el boton "Perfil". El sistema mostrará los datos pertenecientes a ese usuario.
Resultado	Positivo

Cuadro 7.18: Prueba de caja negra: Ver perfil

7.2. PRUEBAS DE CAJA NEGRA

PCN-19	Modificar contraseña
Objetivo	Cambiar la contraseña propia de un usuario.
Actor	Usuario
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Perfil</i>
Acción esperada	El usuario hará click sobre el boton "Cambiar contraseña". El sistema mostrará el formulario de cambio de contraseña. Tras completarlo, el sistema válida los datos ofrecidos, si todo es correcto, limpia el formulario y ofrece un mensaje de "Contraseña actualizada".
Resultado	Positivo

Cuadro 7.19: Prueba de caja negra: Modificar contraseña

PCN-20	Ver histórico diario
Objetivo	Ver el registro de medidas de un día concreto.
Actor	Usuario
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Diario</i>
Acción esperada	El usuario seleccionará un día en un selector modo calendario. El sistema ofrecerá una gráfica con las medidas para el día seleccionado.
Resultado	Positivo

Cuadro 7.20: Prueba de caja negra: Ver histórico diario

PCN-21	Ver histórico mensual
Objetivo	Ver el registro de medidas de un mes concreto
Actor	Usuario
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Mensual</i>
Acción esperada	El usuario seleccionará un mes de un año en un selector modo calendario. El sistema ofrecerá una gráfica con los valores promedios para el mes seleccionado.
Resultado	Positivo

Cuadro 7.21: Prueba de caja negra: Ver histórico mensual

PCN-22	Ver histórico anual
Objetivo	Ver el registro de medidas de un año concreto
Actor	Usuario
Precondición	El usuario ha de estar registrado e identificado. Sección <i>Anual</i>
Acción esperada	El usuario seleccionará un año en un selector modo calendario. El sistema ofrecerá una gráfica con los valores promedios para el año seleccionado.
Resultado	Positivo

Cuadro 7.22: Prueba de caja negra: Ver histórico anual

7.3. Consideraciones

Debido al estado de alarma, ha sido imposible acceder a una instalación fotovoltaica durante el período de desarrollo del proyecto. Por lo que, en ningún momento ha sido posible probar la medición directa de un inversor. Sin embargo, el fundamento teórico para tomar medidas de intensidad tras el inversor de una instalación solar, es el mismo que para tomar medidas en cualquier punto de una red doméstica.

A su vez, debido a que no se ha podido adquirir, durante el período de desarrollo del proyecto, sensores no invasivos, no se ha podido conectar el dispositivo al cuadro de luz del hogar. En su lugar, para realizar pruebas sobre las mediciones, se han empleado cargas variables. (Véase cualquier aparato electrónico).

Capítulo 8

Conclusiones

8.1. Conclusiones

Poniendo punto y final a este Trabajo de Fin de Grado, podemos concluir que los objetivos principales han sido cumplidos:

- Se ha diseñado y montado un dispositivo capaz de tomar medidas de tensión y corriente de la red eléctrica, tanto en su módulo de producción como en su módulo consumo.
- Se ha creado un sistema informático para la gestión y monitorización de las medidas capturadas por el dispositivo de medición.

La realización del Trabajo de Fin de Grado ha supuesto un verdadero desafío para el alumno, donde se han puesto a prueba los conocimientos adquiridos durante el *Grado en Ingeniería Informática*.

Ha sido una gran oportunidad para demostrar la capacidad del alumno para enfrentarse a nuevos desafíos desde cero, donde se ha tenido que ampliar mucho conocimiento procedente de cursos online y fuentes externas.

Sin duda, la experiencia mas reseñable ha sido trabajar en colaboración de profesionales en el ámbito energético, de los cuales he aprendido todo lo posible. Las continuas reuniones con el equipo técnico han servido al alumno para hacerse una idea de como es el mundo laboral cuando trabajas como desarrollador independiente.

Cabe resaltar la dificultad que el estado de alarma ha añadido al desarrollo del proyecto, impidiendo el desarrollo previamente planeado y dando lugar a un escenario más dinámico

y reactivo. Ha sido un lamento para el alumno no poder probar el sistema con un panel solar y un cuadro de luz como puntos de toma de medidas, como se ideó en un inicio.

8.2. Líneas futuras

La larga extensión del trabajo ha obligado a recortar funciones, que serán implementadas en un futuro. Vamos a enumerar dichos recortes:

- Mecanismo de emparejamiento: La API está provista de una URI para realizar el emparejamiento instalación-dispositivo. Falta rediseñar el Daemon de Linux y el “Esquema” de la entidad “Instalación”.
- Servicio de autenticación central: Una de las propuestas mas ambiciosas por parte de *Energética Coop*, era la elaboración de un CAS (Sistema de autenticación central).
- Calculo de la potencia máxima por intervalos en segundos.
- Comparativas entre gráficas.

Las posibilidades que este sistema puede ofrecer son inmensas, desde mejorar muchas funcionalidades ya implementadas hasta implementar algoritmos de *big-data* para analizar los datos y poder crear recomendaciones inteligentes al usuario cliente. A continuación se listan las mejoras necesarias en un futuro cercano:

- Gráfica interanual: Comparativa entre diferentes años.
- Ampliar el diseño del sistema para instalaciones trifásicas.
- Añadir nuevos tipos de gráfica, tipo “Streamgraph”.
- Sustituir los sensores invásivos por no invásivos.
- Mejorar la interfaz del sistema.
- Añadir la posibilidad de parametrizar el intervalo de medida.

Apéndice A

Guía de instalación

Este pequeño apéndice pretende servir de guía para la instalación del sistema una vez que la infraestructura este montada y los dispositivos conectados. Enumeraremos la lista de tareas a seguir:

1. Arduinos: Cargar códigos.
2. Raspberry Pi:
 - a) Instalación del S.O.
 - b) Configuración de la red Wifi.
 - c) Instalación del daemon: RegistroDatos.
3. Instalación del servidor de back-end: API.
4. Instalación del servidor de front-end: React.

A.1. Arduinos

Para la configuración de los arduinos basta con cargar los scripts correspondientes a cada uno, a través de *Arduino IDE*.

A.2. Raspberry Pi

A.2.1. Instalación del sistema operativo

Podemos descargar la imagen del S.O de manera gratuita desde la página oficial de la plataforma:

- <https://www.raspberrypi.org/downloads/raspberry-pi-os/>

No ahondaremos mucho en el tema ya que desde la plataforma podremos obtener guías gratuitas [12] para la instalación de *Raspberry Pi OS*.

A.2.2. Configuración de la red Wifi

Para comenzar, listaremos las redes al alcance del dispositivo [6]:

- `sudo iwlist wlan0 scan`

Una vez que tengamos localizada la red a la que queremos acceder, debemos editar el archivo *wpa_supplicant.conf* [8], para ello:

- `sudo nano /etc/wpa_supplicant/wpa_supplicant.conf`

Añadimos las siguientes líneas al final del fichero:

- ```
network={
 ssid="nombre-red-wifi"
 psk="password-red-wifi"
 proto=WPA2
 key_mgmt=WPA-PSK
}
```

Tras reiniciar el dispositivo con `sudo reboot` podremos comprobar si la conexión se ha realizado correctamente con el comando:

- `ifconfig wlan0`

El siguiente paso será asignar una IP fija al dispositivo [7]. Para ello debemos conocer los siguientes datos:

- La puerta de enlace la obtenemos con:
  - `ip route show`
- El servidor DNS lo obtenemos con:
  - `cat /etc/resolv.conf`

Para finalizar, debemos modificar el archivo `dhcpcd.conf`:

- Abrimos para modificar:
  - `sudo nano /etc/dhcpcd.conf`
- Copiamos el siguiente fragmento al final del fichero, sustituyendo por los valores de tu red:
  - ```
interface InterfazRed
static ip_address=IPFija/máscara
static routers=PuertaEnlace
static domain_name_servers=PuertaEnlace DNS
```

A.2.3. Instalación del servicio: RegistroDatos

El último paso necesario para tener listo nuestro dispositivo *Raspberry Pi Zero W* sería la configuración del servicio responsable de leer las medidas por el puerto serie, emitidas por el arduino.

Previamente, debemos tener localizados en el dispositivo los archivos `scriptRegistroDatos.service`, `registroDatos.cpp` y `instalar.sh`. Para su instalación basta con lanzar el ejecutable: `instalar.sh`.

Dicho ejecutable compilará el fichero escrito en C++, y lo moverá a la ruta: `/usr/bin`. A mayores, copiará el archivo `scriptRegistroDatos.service` en la ruta: `/etc/systemd/system/`.

Por último activará en el servicio en el inicio y lo arrancará:

- ```
systemctl start registroDatos
systemctl enable registroDatos
```

## A.3. API REST: Node.JS

Comenzaremos descargando Node.JS desde la web oficial: <https://nodejs.org/es/>.

La instalación es realmente sencilla por lo que no profundizaremos en ello, basta con ejecutar el instalador, seleccionar la unidad de almacenamiento y aceptar la licencia de uso. Tras instalar el paquete podemos ejecutar en un terminal `node -v` para comprobar que se instaló correctamente. Con la instalación de *Node* viene incluida la instalación de su gestor de paquetes *NPM*.

Una vez instalado, debemos localizarnos en el directorio *ServidorNodeJs* y abrir una terminal.

1. Instalar las dependencias del proyecto:

- `npm install`

2. Ejecutar el servidor:

- `npm start`

Cabe resaltar la necesidad de tener *Python* (Version 2.7) instalado en el sistema para resolver algunas dependencias.

## A.4. Front-end: React

Para lanzar el servidor de front-end basta con que nos localicemos en el directorio *FrontEnd-React* y abramos una terminal:

1. Instalar las dependencias del proyecto:

- `npm install`

2. Ejecutar el servidor:

- `npm start`

Si hemos realizado todos los pasos correctamente ya tendremos nuestro sistema listo para su uso.

## Apéndice B

# Manual de usuario

En este apéndice vamos a detallar un pequeño manual para futuros usuarios. Echaremos un vistazo a las diferentes opciones de interacción que tiene el usuario contra el sistema, haciendo una distinción según el tipo de usuario.

### B.1. Administrador

Comenzaremos por la página principal de un administrador, tras haber iniciado sesión. Dicha página principal se corresponde con la gestión de usuarios administrador. En el menú lateral se corresponde con la primera opción.

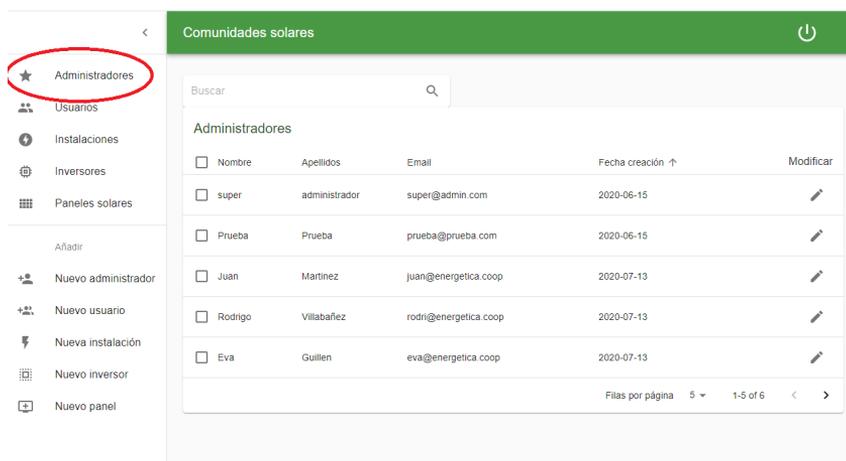


Figura B.1: Vista: Administradores.

## B.1. ADMINISTRADOR

Los administradores en cada página de entidad tendrán una barra de búsqueda para filtrar los campos de la tabla. (Véase figura B.2).

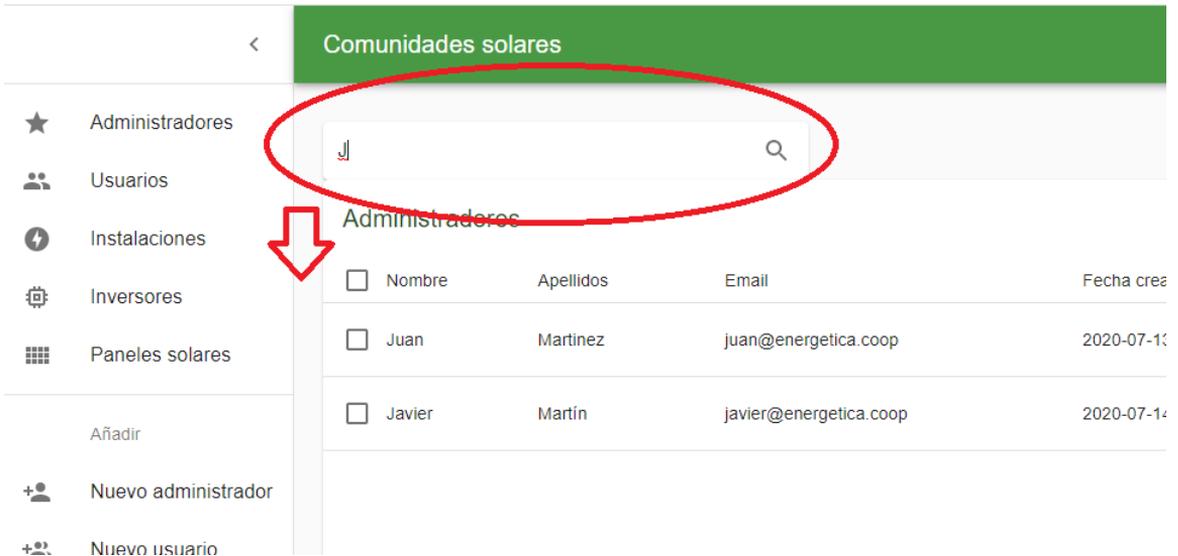


Figura B.2: Vista: Barra de búsqueda.

Por cada fila que nos aparece en la tabla, tendremos dos posibilidades de acción sobre el elemento: Modificar y/o Eliminar. Comenzaremos por la opción eliminar. Para ello debemos seleccionar algún elemento que deseemos eliminar. En la figura B.3 vemos lo que ocurre al seleccionar un elemento de la tabla. Se muestra un botón para eliminar los elementos seleccionados. Podemos seleccionar tantos elementos como deseemos.

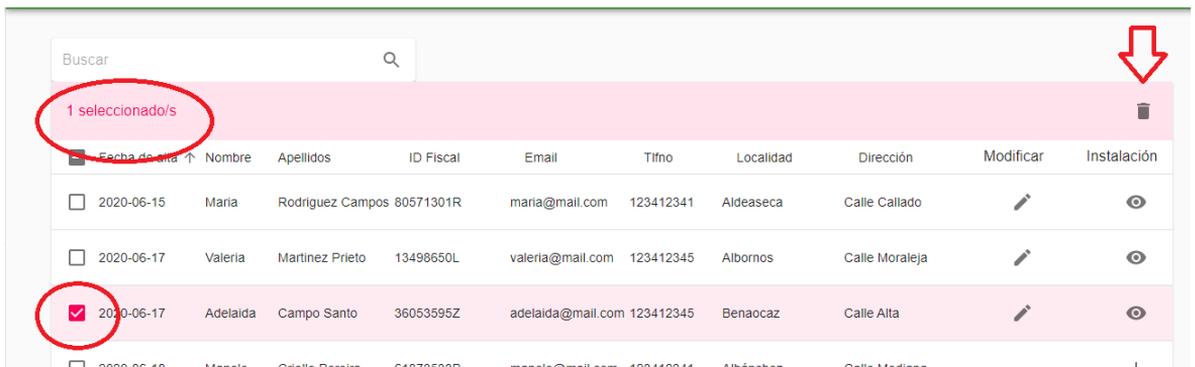


Figura B.3: Vista: Selección de elementos.

Si decidimos eliminar algún elemento, se mostrará el siguiente diálogo de confirmación. (Figura B.4). Si elegimos “Cancelar” volveremos a la pantalla anterior. Si elegimos “Sí, estoy seguro” se borrarán los elementos seleccionados.

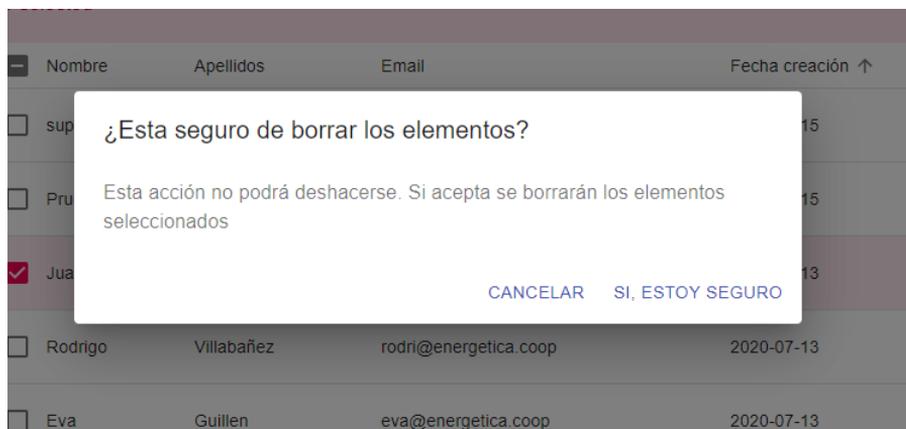


Figura B.4: Vista: Diálogo de confirmación.

Ahora pasemos a la opción “Modificar”. Si sobre la tabla seleccionamos la opción *Modificar*, es decir, clickamos sobre el lápiz que sale en la fila de un elemento (Véase figura B.5, obtendremos un diálogo pop-up con un formulario para la modificación de datos. (Véase figura B.6).

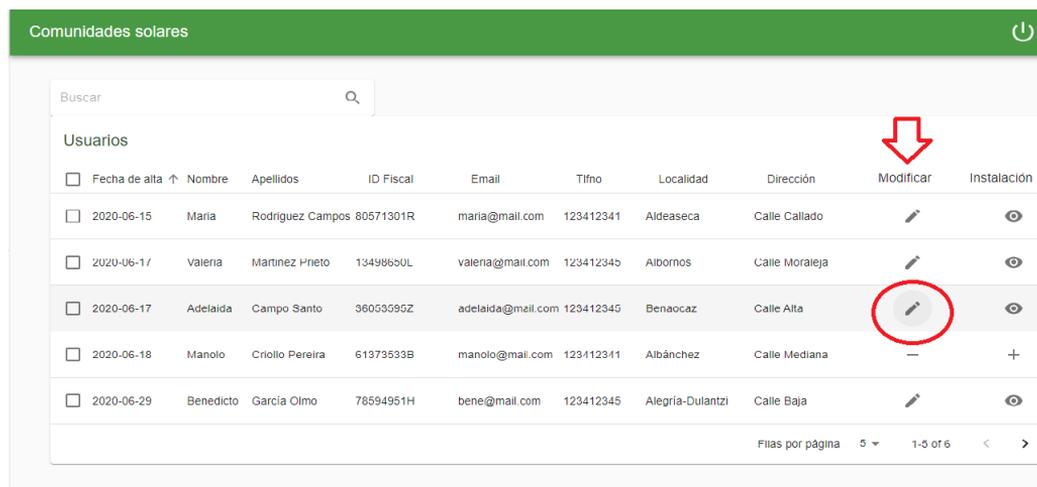


Figura B.5: Vista: Botón modificar.



1 selected

Modificar usuario

Datos personales

Nombre  
super

Apellidos  
administrador

Email  
super@admin.com

CANCELAR MODIFICAR

Guillen eva@energetica.coop 2020-07-13

Figura B.6: Vista: Formulario de modificación.

### B.1.1. Vinculación de instalaciones

En la aplicación web diseñada todas las pantallas para el administrador comparten vista, solo varían los campos, sus tipos y validaciones. Todas menos la pantalla de *Usuarios*. En esta pantalla se tendremos diferentes posibilidades de acción sobre las filas de la tabla (Véase figura B.7).

Tenemos dos tipos de usuarios: Con instalación asignada, y sin instalación asignada.

Si el usuario tiene una instalación asignada: Aparecerán 2 iconos un lápiz y ojo. El lápiz es para modificar la entidad, y el ojo para visualizar los datos de la instalación asignada. Es el caso de “Maria Rodriguez Campos”.

Si el usuario no tiene una instalación asignada: Aparecerá el icono de asignar instalación (+). Es el caso de “Manolo Criollo Pereira”.

Comenzaremos por los usuarios con instalación asignada.

Opción “Modificar”: Si seleccionamos el lápiz, obtendremos un formulario para la modificación del usuario. Dicho formulario es análogo al de la figura B.7 pero con los campos correspondientes a un usuario normal.

Opción “Ver instalación”: Si seleccionamos el ojo, cambiaremos a la siguiente pantalla (Véase figura B.8), donde se muestran los datos de la instalación. En esta pantalla tendremos a nuestra disposición un botón “Modificar” para actualizar los datos de la instalación si

| <input type="checkbox"/> | Fecha de alta ↑ | Nombre    | Apellidos        | ID Fiscal | Email             | Tífono    | Localidad        | Dirección      | Modificar | Instalación |
|--------------------------|-----------------|-----------|------------------|-----------|-------------------|-----------|------------------|----------------|-----------|-------------|
| <input type="checkbox"/> | 2020-06-15      | Maria     | Rodríguez Campos | 80571301R | maria@mail.com    | 123412341 | Aldeaseca        | Calle Callado  |           |             |
| <input type="checkbox"/> | 2020-06-17      | Valeria   | Martínez Prieto  | 13498650L | valeria@mail.com  | 123412345 | Albornos         | Calle Moraleja |           |             |
| <input type="checkbox"/> | 2020-06-17      | Adelaida  | Campo Santo      | 36053595Z | adelaida@mail.com | 123412345 | Benaocaz         | Calle Alta     |           |             |
| <input type="checkbox"/> | 2020-06-18      | Manolo    | Criollo Pereira  | 61373533B | manolo@mail.com   | 123412341 | Albánchez        | Calle Mediana  | -         | +           |
| <input type="checkbox"/> | 2020-06-29      | Benedicto | García Olmo      | 78594951H | bene@mail.com     | 123412345 | Alegría-Dulantzi | Calle Baja     |           |             |

Figura B.7: Vista: Usuarios.

fuera necesario. Si clickamos se desplegará un formulario con los campos necesarios para modificar una instalación.

Pasemos al caso del usuario que no tiene una instalación asignada. Por lo tanto, sobre la tabla picamos en el icono (+) que aparece en la fila de “Manolo Criollo Pereira”. Se nos mostrará la siguiente pantalla (Véase figura B.9), con el nombre del usuario seleccionado en la tabla, y un formulario para cumplimentar con los datos de la instalación.

Figura B.8: Vista: Instalación de usuario.

Comunidades solares ⏻

Usuario: 71182016y, Sergio Rodriguez ▾

|                                         |                                                                          |                                                                                                                                                                                                          |
|-----------------------------------------|--------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Inversor</b><br>Sunny Boy 2.0 VL40 ▾ | <b>Paneles</b><br>JAP60S01 275-SC ▾<br>Potencia pico instalada (Wp)<br>0 | <b>Instalación</b><br>Número de módulos<br>Número de strings<br>Número de paneles instalados<br>0<br>MPPT<br>Modalidad de la instalación<br>Compartido ▾<br>Gestión del excedente<br>Con vertido a red ▾ |
|-----------------------------------------|--------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Figura B.9: Vista: Asignación de instalaciones.

### B.1.2. Añadir entidades

En el menú lateral, contamos con 5 opciones para visualizar datos (5 primeras) y 5 opciones de creación de datos (5 últimas). A continuación vamos a mostrar un ejemplo de página de creación de entidades, crear un objeto inversor. Para ello en el menú lateral seleccionamos “Nuevo inversor”. (Véase figura B.10). Se nos mostrará una pantalla con un

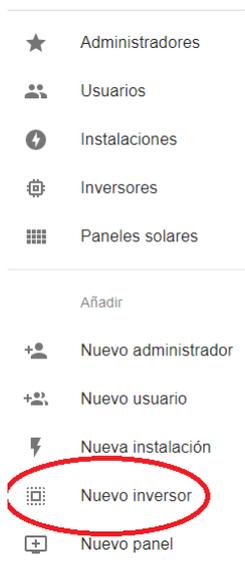


Figura B.10: Vista: Nuevo inversor.

formulario para cumplimentar con los datos del inversor (Véase figura B.11). Si en vez de “Nuevo inversor”, picasemos a “Nuevo usuario” obtendríamos una pantalla similar pero con un formulario con los campos pertenecientes a la entidad usuario.

### B.1.3. Cerrar sesión

Para finalizar este pequeño manual, cerraremos con el caso *Cerrar sesión*, para el cuál el administrador, dispone de un botón en la barra superior. (Véase figura B.12).

Comunidades solares ⌵

Inversor

Marca  Modelo  Potencia

Tensión CA  Selección de tipo   
 Monofásico ▼

**AÑADIR INVERSOR**

Figura B.11: Vista: Añadir entidad.

⌵

⬆

| Fecha creación ↑ | Modificar |
|------------------|-----------|
| 2020-06-15       |           |
| 2020-06-15       |           |
| 2020-07-13       |           |

Figura B.12: Vista: Cerrar sesión.

## B.2. Usuario

En esta segunda parte, vamos a describir las diferentes posibilidades de interacción de un usuario sobre el sistema.

Comenzaremos por la página principal que el usuario ve tras iniciar sesión. Esta página se corresponde con la primera opción en el menú lateral, llamada “Panel de control” y en ella se muestran las medidas instantáneas (Véase figura B.13).

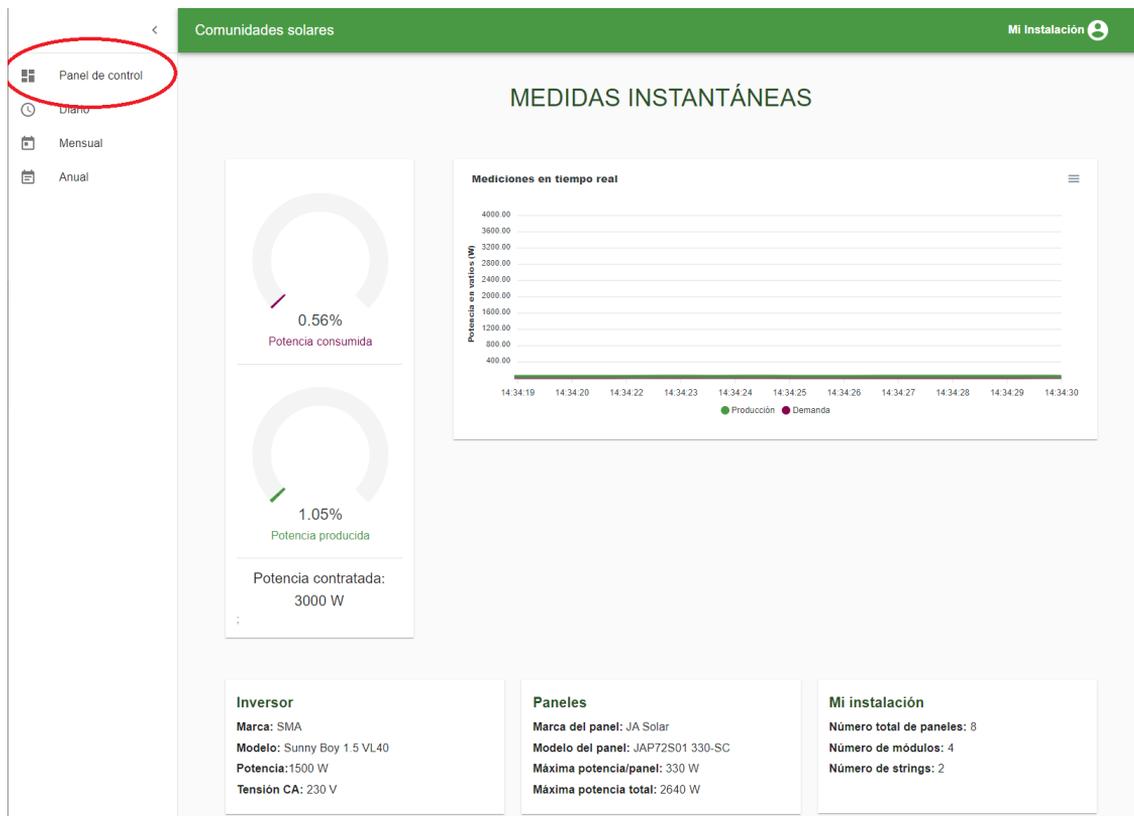


Figura B.13: Vista: Usuario con conexión al dispositivo.

En esta página podremos ver varios elementos:

- Gráfica con las potencias instantáneas medidas por el dispositivo.
- Potenciómetro de 0 a la potencia contratada para las mediciones de consumo.
- Potenciómetro para la potencia generada tomada por las mediciones de producción.
- Datos de la instalación perteneciente al usuario.

## B.2. USUARIO

---

En caso de que no dispongamos de conexión al dispositivo nos aparecerá un mensaje indicandoló en lugar de los elementos gráficos (Véase figura B.14).



Figura B.14: Vista: Usuario sin conexión al dispositivo.

Ahora pasemos a analizar el menú lateral del usuario. En el menú tenemos varias opciones:

- Ver panel de control (Véase figura B.13).
- Ver histórico por días (Véase figura B.15).
- Ver histórico por meses (Véase figura B.16).
- Ver histórico por años (Véase figura B.17).

En cada histórico podremos seleccionar la fecha que queremos visualizar, se nos mostrará la gráfica con los datos y un balance energético para la fecha seleccionada. Dicho balance muestra:

- Producción.
- Demanda.
- Exportada.
- Importada.
- Ratio de autoconsumo.

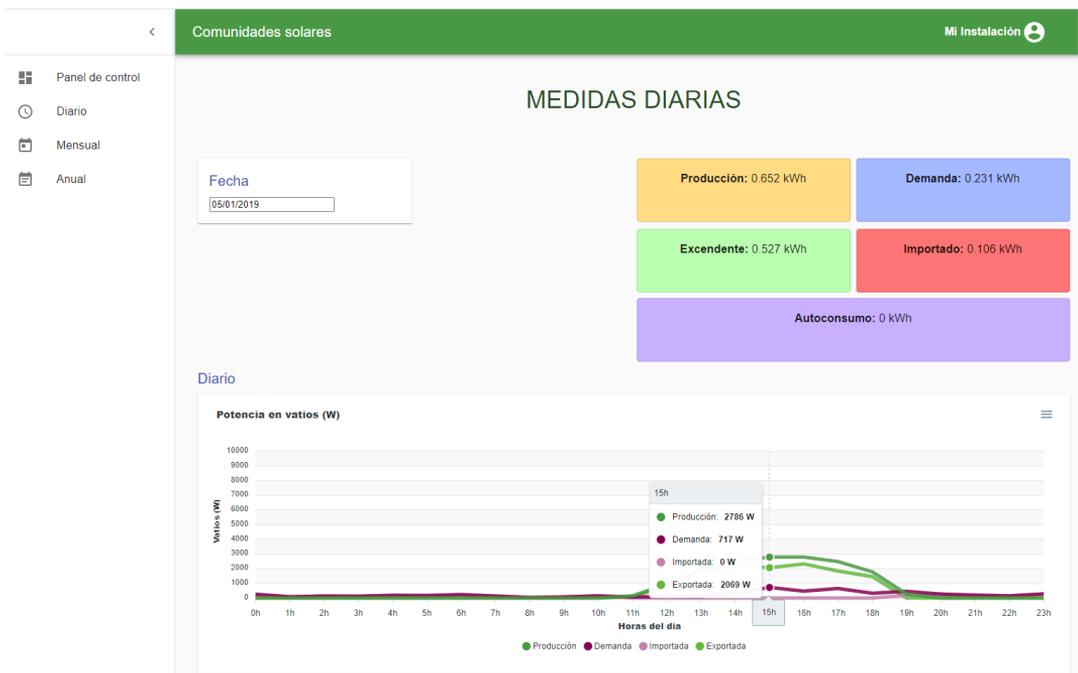


Figura B.15: Vista: Histórico diario.

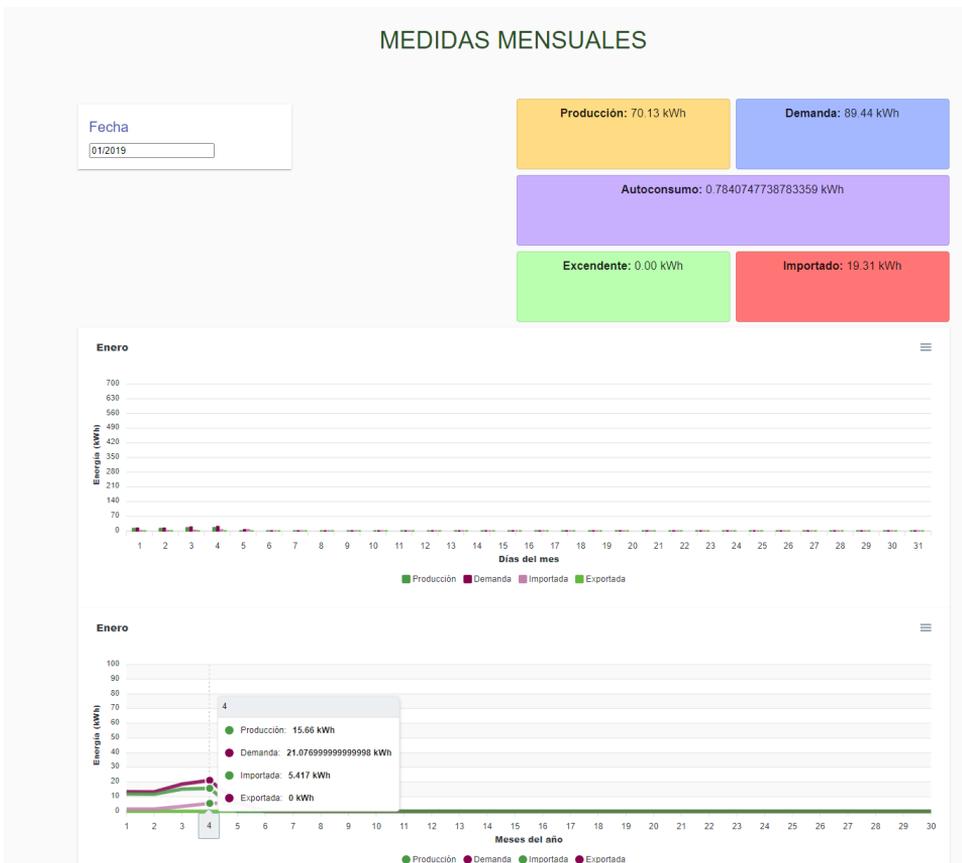


Figura B.16: Vista: Histórico mensual.

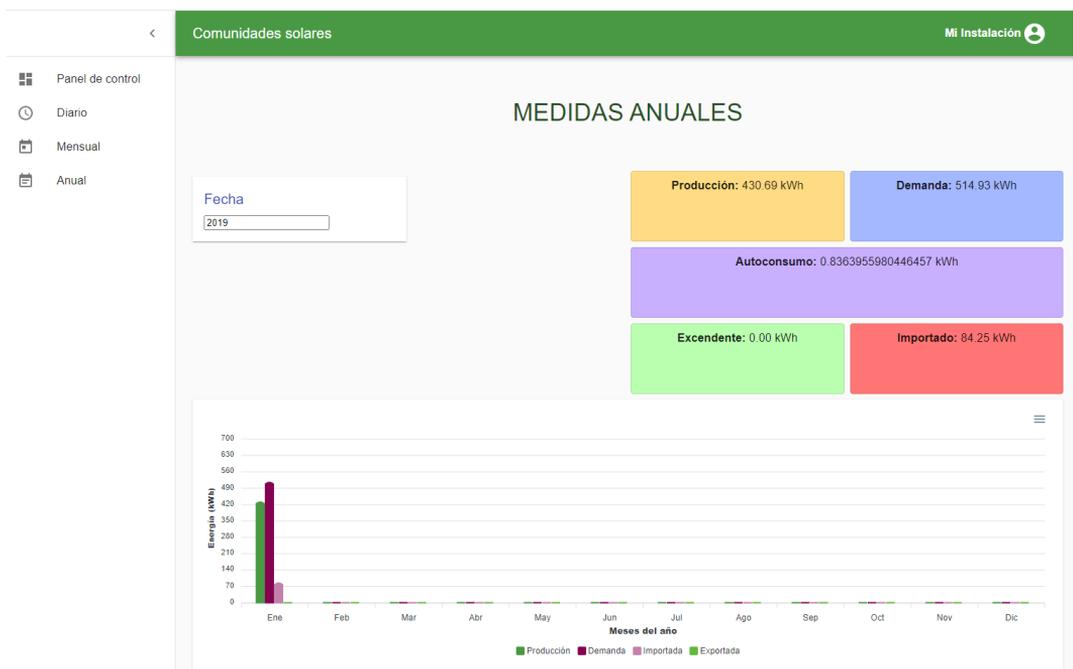


Figura B.17: Vista: Histórico anual.

## B.2. USUARIO

---

Por último solo nos queda echar un vistazo al menú desplegable superior de la aplicación. (Véase figura B.18). Contamos con 2 opciones: “Mi perfil” y “Cerrar sesión”.



Figura B.18: Vista: Menú desplegable superior.

Opción “Mi perfil”: Se nos mostrará una pantalla con nuestros datos (Véase figura B.19). En esta pantalla tendremos 2 botones: “Cambiar contraseña” y “Cerrar sesión”.

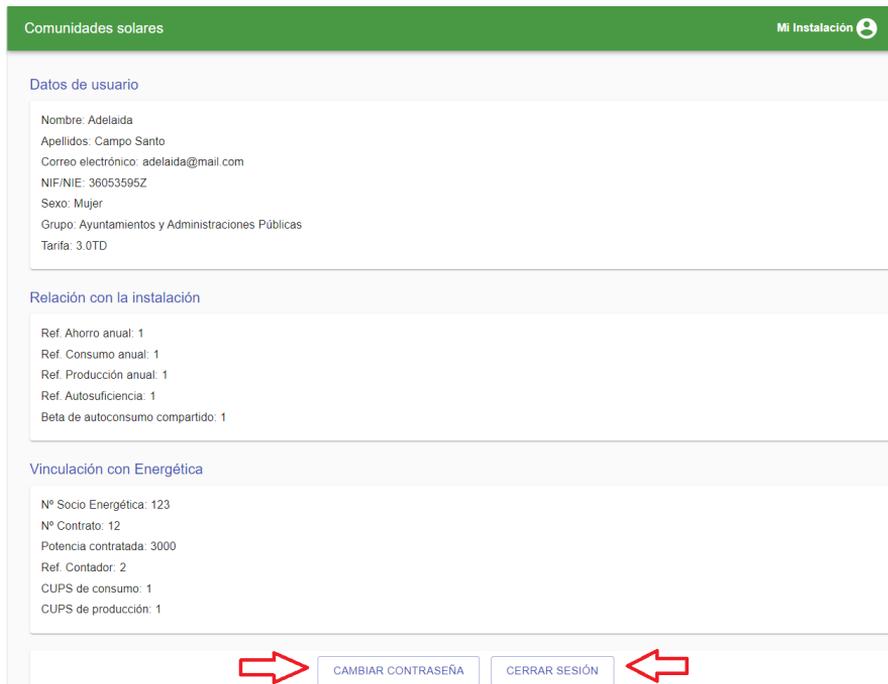


Figura B.19: Vista: Mi perfil.

Si decidimos cambiar la contraseña, se nos desplegará debajo un formulario para el cambio de la contraseña.

Si decidimos cerrar sesión saldremos de la aplicación.

# Apéndice C

## Guía de montaje

En esta pequeña guía expondremos los esquemas de los dispositivos utilizados en el diseño de los circuitos de medición. A mayores se incluyen esquemas del diseño a alto nivel del montaje, y capturas de los circuitos montados por el alumno.

### C.1. Esquemas

#### C.1.1. Arduino Nano

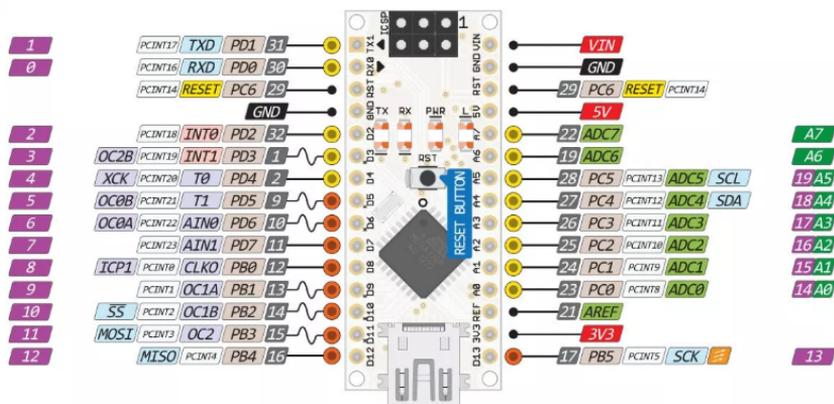


Figura C.1: Imagen recuperada de: <https://bigdanzblog.wordpress.com/2015/01/30/cant-get-i2c-to-work-on-an-arduino-nano-pinout-diagrams>

### C.1.2. Raspberry Pi Zero W

#### Raspberry Pi Zero v1.3




| Position | Power | Ground | Control | GPIO |
|----------|-------|--------|---------|------|
| Wiring   | BCM   | Serial | PWM     | Misc |

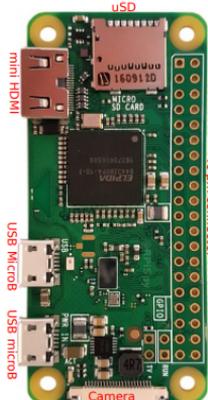
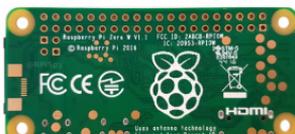
Different places use different pin numbers  
GPIO, Wiring, and BCM have been included.

|          |      |    |     |    |     |
|----------|------|----|-----|----|-----|
|          | 3.3V | 1  | 2   | 5V |     |
| SDA      | 8    | 2  | 3   | 4  | 5V  |
| SCL      | 9    | 3  | 5   | 6  | GND |
| GPCLK0   | 4    | 7  | 4   | 7  |     |
|          | GND  | 9  | 10  | 14 | 15  |
| spi1 CS1 | 17   | 0  | 17  | 11 | 12  |
|          | 27   | 2  | 27  | 13 | 14  |
|          | 22   | 3  | 22  | 15 | 16  |
|          | 3.3V | 10 | 19  | 20 | GND |
| MOSI     | 12   | 10 | 19  | 20 | GND |
| MISO     | 13   | 9  | 21  | 22 | 25  |
| SCLK     | 14   | 11 | 23  | 24 | 8   |
|          | GND  | 25 | 26  | 7  | 11  |
| ID_SD    | 30   | 0  | DNC | 27 | 28  |
| GPCLK1   | 5    | 21 | 5   | 29 | 30  |
| GPCLK2   | 6    | 22 | 6   | 31 | 32  |
| PWM1     | 13   | 23 | 13  | 33 | 34  |
|          | 19   | 24 | 19  | 35 | 36  |
|          | 26   | 25 | 26  | 37 | 38  |
|          | GND  | 39 | 40  | 21 | 29  |
|          |      |    |     | 21 | 21  |

|      |         |      |    |     |     |
|------|---------|------|----|-----|-----|
| PP1  | USB     | TV + | TV | Run | Run |
| PP6  | GND     | TV - | TV | Run | Run |
| PP8  | 3.3V    |      |    |     |     |
| PP14 | SD CLK  |      |    |     |     |
| PP15 | SD CMD  |      |    |     |     |
| PP16 | SD DAT0 |      |    |     |     |
| PP17 | SD DAT1 |      |    |     |     |
| PP18 | SD DAT2 |      |    |     |     |
| PP19 | SD CD   |      |    |     |     |
| PP22 | USB D+  |      |    |     |     |
| PP23 | USB D-  |      |    |     |     |

GPIO 0 and 1 are reserved - Do Not Connect  
PAL or NTSC via composite video on TV pads  
Run - temporarily connect pins to reset chip (or start chip after a shutdown)  
Camera Connector (not on Zero 1.1 or 1.2) - 22pin, 0.5mm  
Board Dimensions - 65mm x 30mm x 0.2mm  
Mounting holes M2.5

#### Raspberry Pi Zero W v1.1

**Processor - BCM2835**  
ARM v7  
Single Core  
1GHz  
(same as B/B+ and A/A+)

**Memory**  
512MB RAM  
uSD slot to run OS

**Video**  
mini HDMI  
PAL or NTSC via pads  
HDMI capable of 1080p

**USB**  
microB for power  
microB for OTG

**Audio**  
from HDMI port only

**Wireless**  
2.4GHz  
802.11n  
Bluetooth 4.1/BLE



Figura C.2: Imagen recuperada de: [https://cdn.sparkfun.com/assets/learn\\_tutorials/6/7/6/PiZero\\_1.pdf](https://cdn.sparkfun.com/assets/learn_tutorials/6/7/6/PiZero_1.pdf)

### C.1.3. NRF24L01

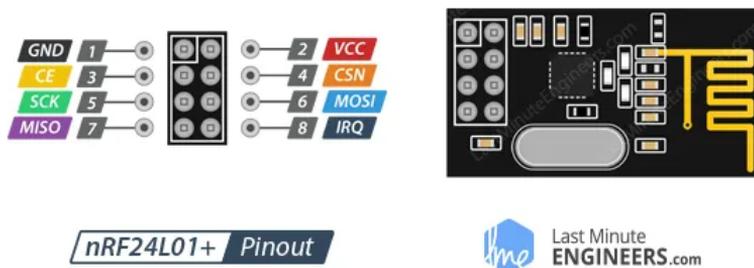


Figura C.3: Imagen recuperada de: <https://saber.patagoniatec.com/2014/07/nrf24l01-rfx2401-nrf24l01-pa-1na-arduino-argentina-ptec>

### C.1.4. ACS712



Figura C.4: Imagen recuperada de: <https://www.luisllamas.es/arduino-intensidad-consumo-electrico-ac712>

### C.1.5. ADS1115

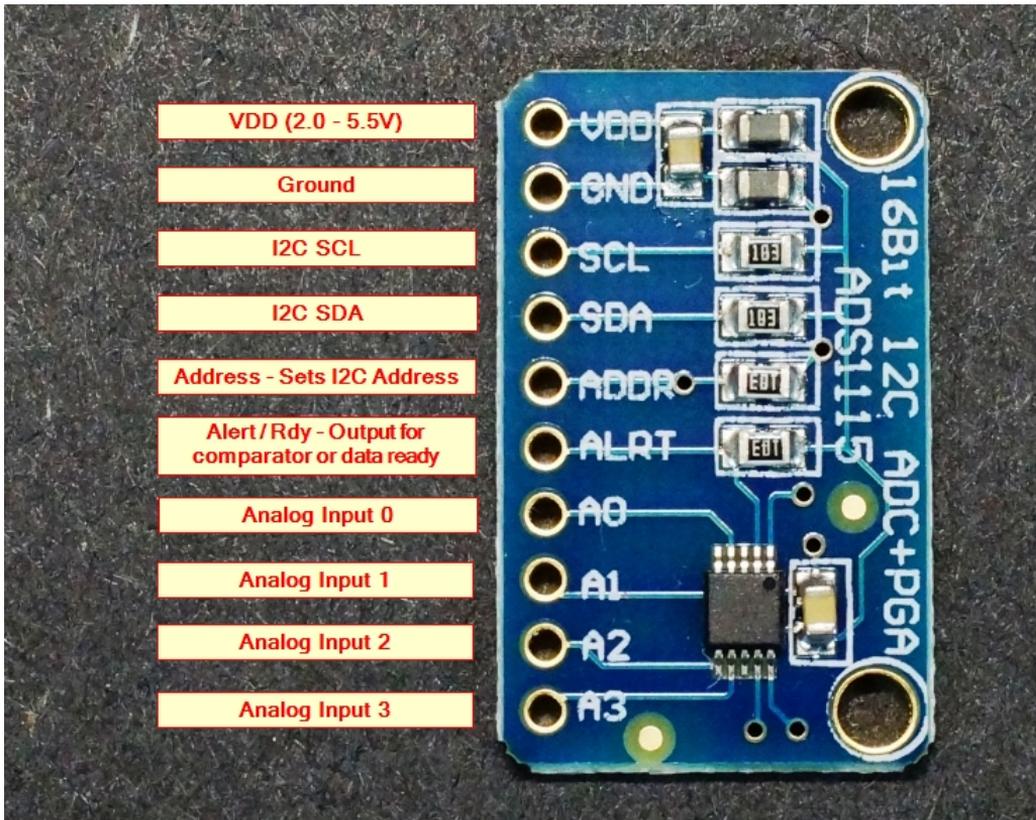


Figura C.5: Imagen recuperada de: [https://www.industrialshields.com/es\\_ES/blog/nuestro-blog-1/post/ads1115-16-bits-adc-board-143](https://www.industrialshields.com/es_ES/blog/nuestro-blog-1/post/ads1115-16-bits-adc-board-143)

### C.1.6. FZ0430



Figura C.6: Imagen recuperada de: <https://www.luisllamas.es/arduino-sensor-corriente-sct-013>

## C.2. Módulo Producción/Consumo

### C.2.1. Medida tensión

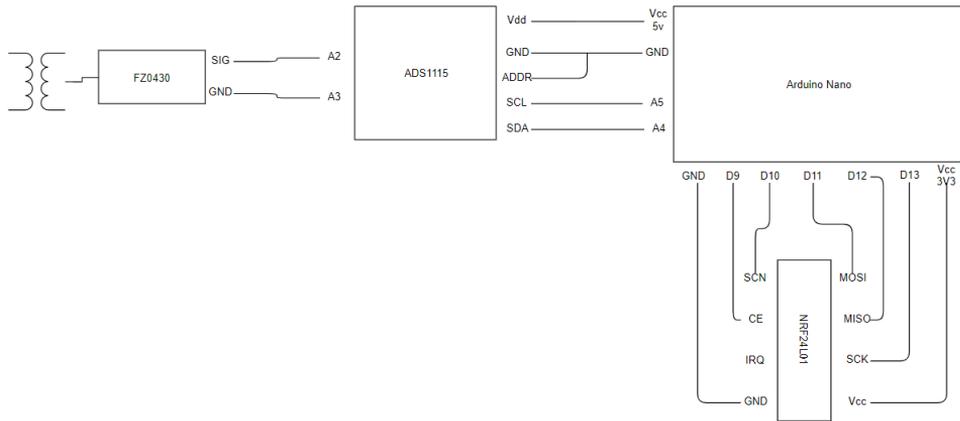


Figura C.7: Circuito medidor de tensión

### C.2.2. Medida intensidad

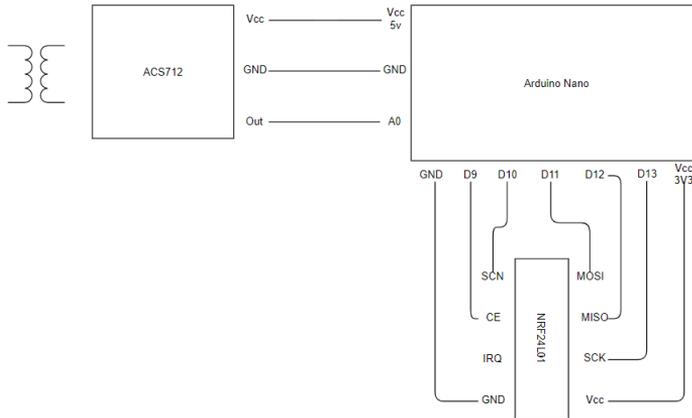


Figura C.8: Circuito medidor de intensidad

### C.3. Circuitos

#### C.3.1. Módulo producción

A continuación se muestra el circuito montado para el módulo de producción. hay algunas consideraciones a tomar en cuenta:

- Debido al estado de alarma se montó el circuito sin el divisor de tension FZ0430, en su lugar hay 2 potenciómetros (30k Ohm y 7.5k Ohm) simulando su función.
- Como no se ha podido acceder a un panel solar, solo se ha podido probar el circuito de medición de producción con cargas variables, enchufadas a los “Conectores” de la imagen C.9. El fundamento teórico es el mismo, y se a codificado los arduinos para que puedan transmitir potencias negativas.

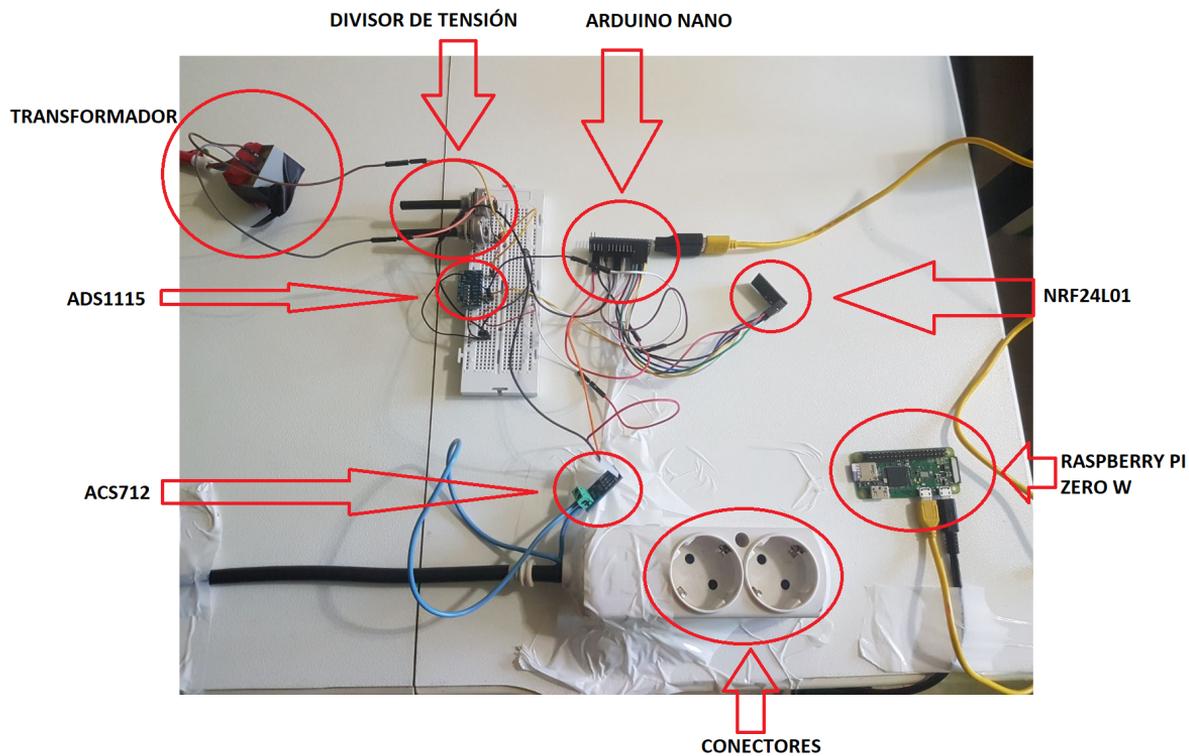


Figura C.9: Circuito módulo producción

### C.3.2. Módulo consumo

A continuación se muestra el circuito montado para el módulo de consumo. Debemos tener en cuenta algunas consideraciones:

- En este caso si contamos con el divisor de tension FZ0430.
- Debido a que no disponemos de sensores no invasivos la forma adaptada para medir el consumo, es enchufar aparatos electrónicos al conector de la imagen C.10. Sin embargo, desde el punto de vista software no cambia nada utilizemos un sensor invasivo o no invasivo. Es decir es igual tomar medidas a raíz del conector, que a raíz del cuadro de luz del hogar.

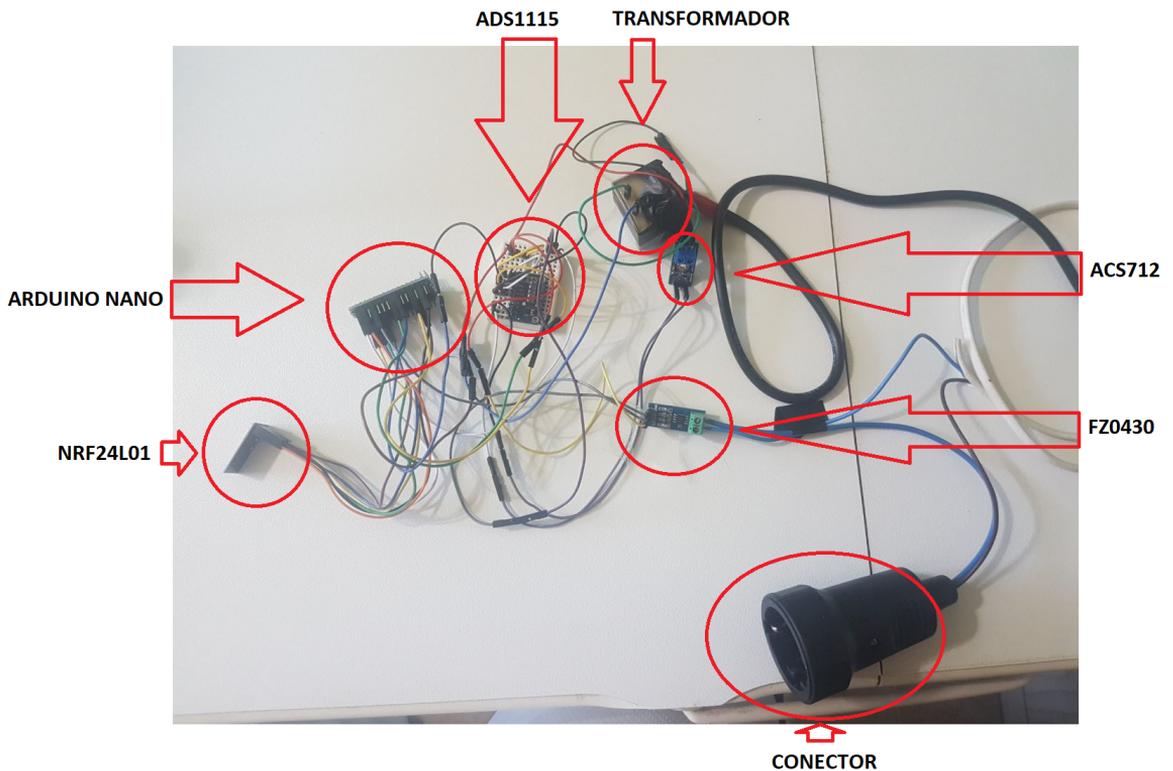


Figura C.10: Circuito módulo consumo

## Bibliografía

- [1] *ACS712 Datasheet(PDF) - Allegro MicroSystems*. URL: <https://www.alldatasheet.es/datasheet-pdf/pdf/168326/ALLEGRO/ACS712.html> (visitado 15-07-2020).
- [2] *ADS1115 16 bits ADC board*. Boot & Work Corp. S.L. Library: [www.industrialshields.com](http://www.industrialshields.com). URL: <https://www.industrialshields.com/blog/nuestro-blog-1/post/ads1115-16-bits-adc-board-143> (visitado 15-07-2020).
- [3] *Black box*. En: *Wikipedia*. Page Version ID: 964572940. 26 de jun. de 2020. URL: [https://en.wikipedia.org/w/index.php?title=Black\\_box&oldid=964572940](https://en.wikipedia.org/w/index.php?title=Black_box&oldid=964572940) (visitado 15-07-2020).
- [4] Mike Bostock. *D3.js - Data-Driven Documents*. Library Catalog: [d3js.org](http://d3js.org). URL: <https://d3js.org/> (visitado 15-07-2020).
- [5] *Calibración*. ardubasic. Library Catalog: [ardubasic.wordpress.com](http://ardubasic.wordpress.com). URL: <https://ardubasic.wordpress.com/tag/calibracion/> (visitado 15-07-2020).
- [6] *Comandos de Red fundamentales que todo administrador debería conocer*. Pandora FMS - The Monitoring Blog. Library Catalog: [pandorafms.com](http://pandorafms.com) Section: Comunidad. 13 de sep. de 2017. URL: <https://pandorafms.com/blog/es/comandos-de-red/> (visitado 15-07-2020).
- [7] *Como asignar IP fija a una Raspberry Pi » Raspberry para novatos*. Raspberry para novatos. Library Catalog: [raspberryparanovatos.com](http://raspberryparanovatos.com) Section: Tutoriales. 31 de jul. de 2019. URL: <https://raspberryparanovatos.com/tutoriales/asignar-ip-fija-raspberry-pi/> (visitado 15-07-2020).
- [8] *Configurar Wifi en Raspberry Pi por GUI o Terminal*. Luis Llamas. Library Catalog: [www.luisllamas.es](http://www.luisllamas.es). URL: <https://www.luisllamas.es/raspberry-pi-wifi/> (visitado 15-07-2020).
- [9] *Connecting an nRF24L01+ to Raspberry Pi*. Hackster.io. Library Catalog: [www.hackster.io](http://www.hackster.io). URL: <https://www.hackster.io/wirekraken/connecting-an-nrf24l01-to-raspberry-pi-9c0a57> (visitado 15-07-2020).

- [10] *Convert float to 4 Bytes and then back again — Mbed*. URL: <https://os.mbed.com/forum/helloworld/topic/2053/?page=1#comment-54126> (visitado 15-07-2020).
- [11] *Daemon (computing)*. En: *Wikipedia*. Page Version ID: 964557692. 26 de jun. de 2020. URL: [https://en.wikipedia.org/w/index.php?title=Daemon\\_\(computing\)&oldid=964557692](https://en.wikipedia.org/w/index.php?title=Daemon_(computing)&oldid=964557692) (visitado 15-07-2020).
- [12] *Download Raspberry Pi OS for Raspberry Pi*. Raspberry Pi. Library Catalog: [www.raspberrypi.org](http://www.raspberrypi.org) URL: <https://www.raspberrypi.org/downloads/raspberry-pi-os/> (visitado 15-07-2020).
- [13] *El Club del Autodidacta*. El Club del Autodidacta. Library Catalog: [elclubdelautodidacta.es](http://elclubdelautodidacta.es). URL: <http://elclubdelautodidacta.es/wp/> (visitado 15-07-2020).
- [14] *Empezando – React*. Library Catalog: [es.reactjs.org](http://es.reactjs.org). URL: <https://es.reactjs.org/docs/getting-started.html> (visitado 15-07-2020).
- [15] *Enviar o recibir un array por puerto serie en Arduino*. Luis Llamas. Library Catalog: [www.luisllamas.es](http://www.luisllamas.es). URL: <https://www.luisllamas.es/array-puerto-serie-arduino/> (visitado 15-07-2020).
- [16] *How to create a React frontend and a Node/Express backend and connect them*. freeCodeCamp.org. Library Catalog: [www.freecodecamp.org](http://www.freecodecamp.org). 7 de sep. de 2018. URL: <https://www.freecodecamp.org/news/create-a-react-frontend-a-node-express-backend-and-connect-them-together-c5798926047c/> (visitado 15-07-2020).
- [17] *How to send float over serial*. URL: <https://forum.arduino.cc/index.php?topic=323586.0> (visitado 15-07-2020).
- [18] *Installation & Getting Started*. ApexCharts.js. Library Catalog: [apexcharts.com](http://apexcharts.com). URL: <https://apexcharts.com/docs/installation/> (visitado 15-07-2020).
- [19] *Introducción al desarrollo backend con Node.js y Express*. Udemy. Library Catalog: [www.udemy.com](http://www.udemy.com). URL: <https://www.udemy.com/course/introduccion-al-desarrollo-backend-con-nodejs-y-express/> (visitado 15-07-2020).
- [20] Judit Izquierdo. “¿Qué es el XP Programming?” En: *Thinking for Innovation* (4 de sep. de 2014). Section: Agile y Scrum. URL: <https://www.iebschool.com/blog/que-es-el-xp-programming-agile-scrum/> (visitado 15-07-2020).
- [21] *JavaScript*. Documentación web de MDN. Library Catalog: [developer.mozilla.org](http://developer.mozilla.org). URL: <https://developer.mozilla.org/es/docs/Web/JavaScript> (visitado 15-07-2020).
- [22] *javascript - Formatting ISODate from MongoDB*. Stack Overflow. Library Catalog: [stackoverflow.com](http://stackoverflow.com). URL: <https://stackoverflow.com/questions/11486779/formatting-isodate-from-mongodb> (visitado 15-07-2020).
- [23] *Linux man pages*. URL: <https://linux.die.net/man/> (visitado 15-07-2020).

- [24] *Material Icons - Material-UI*. Library Catalog: material-ui.com. URL: <https://material-ui.com/es/components/material-icons/> (visitado 15-07-2020).
- [25] *Material-UI: A popular React UI framework*. Library Catalog: material-ui.com. URL: <https://material-ui.com/> (visitado 15-07-2020).
- [26] *Medir tensiones de 220V-230V con Arduino y transformador*. Luis Llamas. Library Catalog: www.luisllamas.es. URL: <https://www.luisllamas.es/medir-tensiones-de-220v-230v-con-arduino-y-transformador/> (visitado 15-07-2020).
- [27] *Mongoose v5.9.24: Getting Started*. URL: <https://mongoosejs.com/docs/> (visitado 15-07-2020).
- [28] *npm — build amazing things*. Library Catalog: www.npmjs.com. URL: <https://www.npmjs.com/> (visitado 15-07-2020).
- [29] *Optimized high speed nRF24L01+ driver class documentation: RF24 Class Reference*. URL: <http://tmrh20.github.io/RF24/classRF24.html#a8e2eacacfb96426c192> (visitado 15-07-2020).
- [30] *Program an Arduino UNO with your Raspberry Pi*. The MagPi magazine. Library Catalog: magpi.raspberrypi.org. URL: <https://magpi.raspberrypi.org/articles/program-arduino-uno-raspberry-pi> (visitado 15-07-2020).
- [31] *React Hooks, useEffect. Añadiendo funcionalidad en el ciclo de vida de nuestro componente - III*. midu.dev. Library Catalog: midu.dev. URL: <https://midu.dev/react-hooks-use-effect-funcionalidad-en-el-ciclo-vida-componentes/> (visitado 15-07-2020).
- [32] *React. 5 Consejos para evitar “wasted renders” — by Gerardo Fernández — Medium*. URL: <https://medium.com/@ger86/react-5-consejos-para-evitar-wasted-renders-5d5195146458> (visitado 15-07-2020).
- [33] *reactjs - How to pass params with history.push/Link/Redirect in react-router v4?* Stack Overflow. Library Catalog: stackoverflow.com. URL: <https://stackoverflow.com/questions/44121069/how-to-pass-params-with-history-push-link-redirect-in-react-router-v4> (visitado 15-07-2020).
- [34] *Sensor de corriente eléctrica no invasivo con Arduino y SCT-013*. Luis Llamas. Library Catalog: www.luisllamas.es. URL: <https://www.luisllamas.es/arduino-sensor-corriente-sct-013/> (visitado 15-07-2020).
- [35] *Stack Overflow - Where Developers Learn, Share, & Build Careers*. Stack Overflow. Library Catalog: stackoverflow.com. URL: <https://stackoverflow.com/> (visitado 15-07-2020).
- [36] *The MongoDB 4.2 Manual — MongoDB Manual*. <https://github.com/mongodb/docs/blob/v4.2/> Library Catalog: docs.mongodb.com. URL: <https://docs.mongodb.com/manual/> (visitado 15-07-2020).

- [37] *Tutorial básico NRF24L01 con Arduino*. URL: [https://naylampmechatronics.com/blog/16\\_Tutorial-b%C3%A1sico-NRF24L01-con-Arduino.html](https://naylampmechatronics.com/blog/16_Tutorial-b%C3%A1sico-NRF24L01-con-Arduino.html) (visitado 15-07-2020).
- [38] *Tutorial sensor de corriente ACS712*. URL: [https://naylampmechatronics.com/blog/48\\_tutorial-sensor-de-corriente-ac712.html](https://naylampmechatronics.com/blog/48_tutorial-sensor-de-corriente-ac712.html) (visitado 15-07-2020).
- [39] *Usar map, filter y reduce para olvidarnos de los bucles for* — *Misael Taveras*. URL: <https://taverasmisael.com/blog/usar-map-filter-y-reduce-para-olvidarnos-de-los-bucles-for> (visitado 15-07-2020).
- [40] *What is the difference between /dev/ttyUSB and /dev/ttyACM?* URL: <https://rfc1149.net/blog/2013/03/05/what-is-the-difference-between-devttyusbx-and-devttyacmx/> (visitado 15-07-2020).
- [41] *White box (software engineering)*. En: *Wikipedia*. Page Version ID: 951643368. 18 de abr. de 2020. URL: [https://en.wikipedia.org/w/index.php?title=White\\_box\\_\(software\\_engineering\)&oldid=951643368](https://en.wikipedia.org/w/index.php?title=White_box_(software_engineering)&oldid=951643368) (visitado 15-07-2020).
- [42] *Working with refs in React*. CSS-Tricks. Library Catalog: [css-tricks.com](https://css-tricks.com/working-with-refs-in-react/). 16 de ago. de 2018. URL: <https://css-tricks.com/working-with-refs-in-react/> (visitado 15-07-2020).