industriales
etsii

Escuela Técnica Superior de Ingeniería Industrial

# UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Escuela Técnica Superior de Ingeniería Industrial

# Desarrollo de una pasarela de datos USB-Ethernet con Raspberry Pi

**TRABAJO FIN DE GRADO**

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES


**Autor:** **Martínez Royo, Miguel**
Director: Torres Sánchez, Roque
Codirector: López Riquelme, Juan Antonio


Cartagena, Diciembre de 2017

Universidad Politécnica de Cartagena

## RESUMEN.

Elaboración de un sistema, mediante el uso de un SBC, que funcione como pasarela de datos entre un campo de nodos sensorizados y el usuario que examine dichos datos. Se procesarán las tramas recibidas por los nodos para posteriormente organizar los datos en una base de datos estructurada. Se desarrollará también una interfaz gráfica que establezca la comunicación con la base de datos y permita modificarla y acceder a los datos de acuerdo con las necesidades.



## SUMMARY.

Making a way to communicate a node field with the administrator of the network, using an SBC system. After processing the information received from nodes, the system will store the data in an organized DataBase. Moreover, a graphical user interface to modify the DataBase and access to data required from the user will be implemented.

# ÍNDICE DE CONTENIDO.

# ÍNDICE DE FIGURAS.

# Capítulo 1

# Introducción

## 1.1 Objetivo del proyecto.

Este proyecto se centra en diseñar y desarrollar una pasarela de datos USB-Ethernet para una red inalámbrica de sensores. Dicha red se está utilizando para monitorizar el estado del suelo y de la planta en una parcela de cerezos, localizada en el término municipal de Jumilla.

Un elemento esencial de la red de sensores en el nodo sumidero. Dicho nodo intercambia información con el resto de nodos de la red de forma inalámbrica utilizando la especificación ZigBee, que trabaja sobre el protocolo de comunicación IEEE 802.15.4.

Otro elemento esencial de una red de sensores es el denominado “servidor”. Dicho elemento, que puede ser local o remoto, es el encargado de almacenar la información recolectada por los sensores conectados a los nodos de forma robusta, así como de proporcionar mecanimos sencillos para visualizar los datos recogidos por la red de una manera sencilla y eficiente. Y es en esta línea en la que se centra este trabajo con los dos siguientes objetivos principales:

- Se pretende obtener un dispositivo compacto utilizando plataformas hardware de bajo coste, que permita realizar pasarelas de comunicaciones entre dispositivos con comunicaciones serie (RS232 y USB) y dispositivos con enlace Ethernet.

- Se pretende también, desarrollar una base de datos que almacene las mediciones recibidas de una serie de nodos. Siendo posible además el acceso a datos concretos y la edición de la red de nodos.

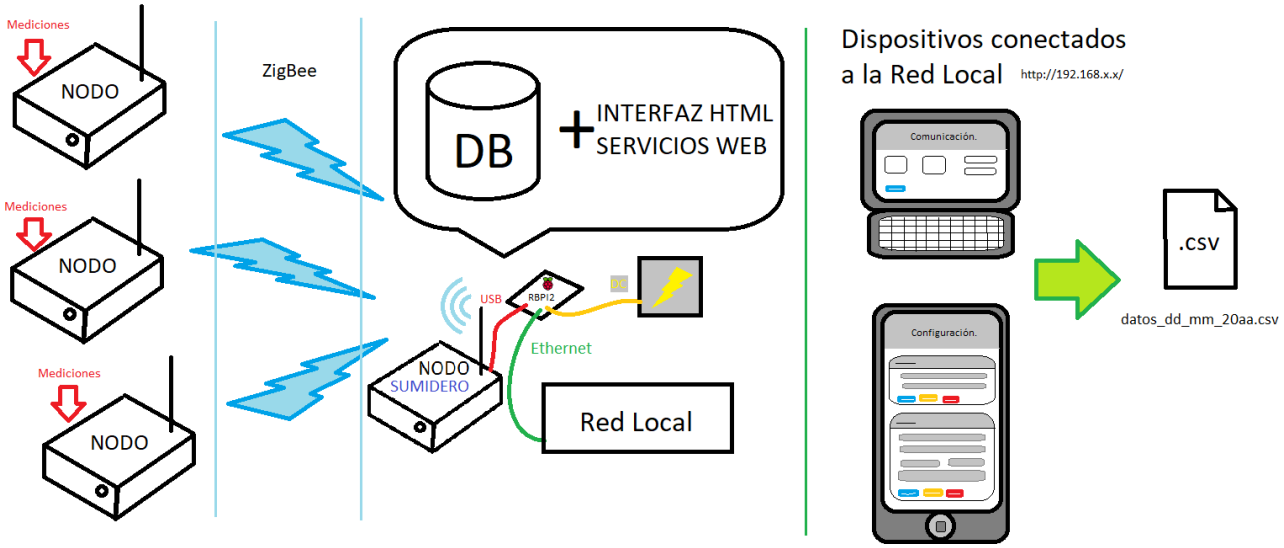


*Figura 1.1 Esquema del trabajo*

## 1.2 Plan de trabajo.

Para comenzar, se deberá elegir una plataforma hardware de bajo coste que se adecúe a las necesidades del proyecto. Una vez dado ese paso, a continuación, se deberán realizar los siguientes softwares:

- Lector del puerto serie del dispositivo hardware: Deberá identificar las “tramas de datos” enviadas por los nodos y ser capáz de seccionar los datos adecuadamente.
- Base de datos: Almacenará las mediciones de forma ordenada y jerárquica en la memoria local del dispositivo.
- Interfaz de comunicación usuario - base de datos: Permitirá la edición de la base de datos, si se produjeran modificaciones de la red de nodos, y la adquisición de datos concretos en una franja temporal establecida por el usuario.

El dispositivo hardware estará conectado vía ethernet a una red local, lo que posibilitará el acceso a la interfaz de comunicación a través de cualquier dispositivo conectado a dicha red. Además, para facilitar el trabajo con entornos como Excel, los datos se podrán descargar con formato CSV.

# Capítulo 2

# Estado del Arte

## 2.1 Introducción.

En este capítulo se va a realizar un estudio sobre las diferentes opciones que se plantean para la realización del proyecto, analizando ciertas características técnicas y potencial de desarrollo. Se explicará el por qué de las elecciones que se vayan tomando. En concreto, los elementos se pueden clasificar en: SBC, sistemas operativos para Raspberry Pi, bases de datos, servicios web y diseño de interfaces de usuario.

## 2.2 Single Board Computer o PC de placa única (SBC).

La plataforma hardware que más se adecúa a la realización de este proyecto, dada la complejidad y variedad de programación requerida, es un SBC.

Se trata de placas que contienen todos o la mayor parte de los componentes de un ordenador. La principal característica de estas placas frente a un PC usual es su reducido tamaño, aproximadamente de dimensiones similares a una tarjeta de visita. Otra de sus características fundamentales es su económico precio.

Por último, cabe destacar una tercera característica, y es que ofrecen poca potencia frente a los procesadores de ordenadores comunes. Su potencia es suficiente aun así para su uso en el mundo de la ofimática, del desarrollo e incluso del mundo multimedia.

Las placas que se van a someter a estudio para el proyecto son: Raspberry Pi 2 Model B, BeagleBoard-X15 y ODROID-C2.

### 2.2.1 Raspberry Pi 2 B

*Figura 2.1 Raspberry Pi 2 B, fig. 1*

*Figura 2.2 Raspberry Pi 2 B, fig. 2*

Especificaciones [1]:

- Procesador: ARM Cortex-A7 900 MHz de 4 núcleos
- GPU: VideoCore IV 3D
- 1 GB RAM
- 4 puertos USB
- 1 puerto Full HDMI
- 1 puerto Ethernet
- 40 puertos GPIO
- 1 puerto Micro SD
- Interfaz de cámara (CSI)
- Interfaz de pantalla (DSI)
- SO: Raspbian (Debian), Linux (Ubuntu) o Windows 10 IoT
- Precio aproximado: 30 €

Cabe destacar que posee una amplia comunidad que desarrolla software para este SBC y es altamente polivalente. Algunas empresas lo implementan en sus productos comerciales, como por ejemplo en el mundo de la impresión 3D (SolidRay 2) [2].

### 2.2.2 BeagleBoard-X15.

*Figura 2.3 BeagleBoard-X15*

Especificaciones [3]:

- Procesador: ARM Cortex-A15 1.5 GHz de 2 núcleos
- GPU: gráficos 2D / 3D
- 2 GB RAM DDR3
- 3 puertos USB 3.0 y 1 puerto 2.0
- 1 puerto Full HDMI
- 2 puertos Ethernet
- 1 puerto Micro SD
- SO: Debian, Android, Linux (Ubuntu) o Cloud9 IDE
- Precio aproximado: 230 €

### 2.2.3 ODROID-C2.

*Figura 2.4 ODROID-C2*

Especificaciones [4]:

- Procesador: ARM Cortex-A53 1,5 GHz de 4 núcleos
- GPU: Mali-450
- 2 GB RAM DDR3
- 4 puertos USB
- 1 puerto HDMI 4K / 60 Hz
- 1 puerto Ethernet
- 1 puerto mini-jack (3.5 mm) de Audio/Vídeo
- 40 puertos GPIO + 7 pines I2S
- 1 puerto Micro SD
- SO: Linux (Ubuntu) o Android
- Precio aproximado: 40 €

### 2.2.4 Conclusión.

Por criterio económico, se descartaría completamente la opción de BeagleBoard. Entre las dos opciones restantes hay una pequeña diferencia de precios, en torno a los 10 €, sin embargo, Raspberry posee una enorme comunidad de soporte y es más común el uso de esta plataforma para este tipo de proyectos.

Por tanto, se seleccionará **Raspberry Pi 2 Model B** para realizar el proyecto.

Una vez seleccionada la tarjeta, el siguiente paso será seleccionar el sistema operativo con el que operará dicha tarjeta, ya que existen diferentes posibilidades que se pueden utilizar.

## 2.3 Sistemas Operativos para Raspberry Pi.

Actualmente, existe una gran variedad de sistemas operativos desarrollados para este SBC, tanto oficiales como desarrollados por usuarios. Algunos de los SOs oficiales que ofrece Raspberry son:

- Raspbian: Sistema operativo oficial diseñado para el SBC, basado en Debian.
- Ubuntu Mate: Sistema operativo basado en Ubuntu.

- Ubuntu Core: Nuevo sistema operativo de Ubuntu para desarrolladores.
- Windows 10 IoT: Sistema operativo basado en Windows 10 para desarrolladores. Cabe destacar que no posee escritorio y que se configura de forma remota a través del software de desarrollo de Windows, Visual Studio.
- Open Source Media Center (OSMC): Sistema que convierte la Raspberry en un reproductor multimedia.
- LibreELEC: Sistema operativo creado para el uso de Kodi, un software multimedia.
- PiNet: Sistema operativo diseñado para el ámbito escolar, se utiliza para la gestión de una red de Raspberrys.
- RISC OS Open Limited (ROOL): Sistema operativo desarrollado para dispositivos que utilicen microprocesadores de ARM.
- Weather Station: Sistema operativo desarrollado por Oracle para establecer una estación meteorológica particular. [5]

Dadas las características de los diferentes sistemas operativos propuestos, cabe someter a estudio dos de ellos: Raspbian y Windows 10 IoT.

### 2.3.1 Raspbian.

Características [6]:

- Posee una versión con escritorio y una versión “lite” sin escritorio más optimizada, en cuanto a memoria que ocupa y recursos que consume.
- Viene con algunos softwares pre-instalados para educación, programación y usos generales. (Python, Scratch, Sonic Pi, Java, Mathematica, entre otros).
- Se trata de un sistema operativo particularmente desarrollado para la Raspberry Pi, basado en Debian, aprovechando todos los recursos de la misma con la máxima eficiencia.
- Posee una gran comunidad de usuarios que contribuyen a su continua mejora.

### 2.3.2 Windows 10 Internet of Things (IoT).

Características:

- Se pueden desarrollar y compilar aplicaciones de forma remota a través del IDE Visual Studio instalado en otro ordenador.
- El sistema operativo funciona como ejecutable de las aplicaciones que se le manden. No posee programas ni escritorio instalados por defecto.
- Es muy reciente, todavía no posee una comunidad muy extensa, aunque permite desarrollar aplicaciones muy prácticas usando lenguajes como C#.

### 2.3.3 Conclusión.

Raspbian es el sistema operativo desarrollado con el equipo de Raspberry para sus dispositivos y está altamente documentado por la comunidad Raspberry, posibilitando de manera muy sencilla la implementación de software y bases de datos. Por el contrario, no hay mucha información sobre software realcionado con bases de datos para Windows 10 IoT.

Por tanto, se seleccionará **Raspbian Lite** (sin escritorio) como sistema operativo para la tarjerta seleccionada en la sección anterior.

## 2.4 Bases de datos.

Las bases de datos [7] posibilitan el almacenamiento de los datos de forma sistemática en formato digital. Se utilizan determinados programas denominados DBMS (*DataBase Management System*), que permiten almacenar y posteriormente acceder a los datos de manera rápida y estructurada.

Se pueden clasificar siguiendo distintos criterios: según el contexto en el que se implementen, la ultilidad de las mismas o las necesidades que satisfagan. De entre todos los tipos, hay dos modelos que pueden ser interesantes para este proyecto, según una clasificación de acuerdo con el modelo de administración de los datos: Bases de datos orientadas a objetos y Bases de datos relacionales o Entidad-Relación. [8]

### 2.4.1 Bases de datos orientadas a objetos.

Se trata de almacenar datos como objetos completos (estado y comportamiento). Incorpora los conceptos más importantes del paradigma de objetos:

- Encapsulación: Permite ocultar información al resto de objetos, impidiendo accesos indebidos.
- Herencia: Permite que los objetos puedan heredar comportamientos dentro de una jerarquía de clases.
- Polimorfismo: Permite operar sobre distintos tipos de objetos a través de una operación.

En este modelo de base de datos los usuarios pueden definir operaciones sobre los datos como parte de definición de la base de datos. Una operación (función) se especifica en dos partes: la interfaz (signatura) de una operación, que incluye el nombre de la operación y los tipos de datos de sus argumentos (parámetros), y la implementación (método), que se especifica separadamente y puede modificarse sin afectar a la interfaz. [9]

### 2.4.2 Bases de datos relacionales o Entidad-Relación.

Es el modelo aplicado actualmente para representar problemas reales y administrar datos dinámicamente. Su fundamento es el uso de “relaciones”. Estas relaciones se podrían considerar en forma lógica como conjunto de tuplas. Se piensa en cada relación como si fuera una tabla compuesta por registros (filas de una tabla), que representarían las tuplas, y campos (las columnas).

El lugar y la forma en la que se almacenen los datos no tienen relevancia mediante este modelo. La información se puede recuperar o almacenar mediante el uso de “consultas”.

El lenguaje más habitual de estas bases de datos es SQL (*Structured Query Language*), un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales (DBMS). [10]

### 2.4.3 Conclusión.

Por la sencillez de aplicación, la versatilidad y la organización del **modelo relacional**, será el que se utilice para desarrollar el proyecto. Se utilizará como

DBMS el entorno de MySQL y phpMyAdmin. Habrá que realizar lo que se conoce como un diagrama entidad-relación, estableciendo la estructura de la base de datos que se desarrollará antes de comenzar a programarla.

## 2.5 Servicios Web.

Se trata de una tecnología que se utiliza para el intercambio de datos entre aplicaciones, para ello hace uso de un conjunto de protocolos y estándares ya definidos. La ventaja de estos servicios es que permiten la comunicación entre aplicaciones con lenguajes de programación diferentes y ejecutadas en cualquier plataforma. Esta interoperabilidad se debe a la adopción de estándares abiertos.

La principal razón para el uso de los servicios Web [11] es que se pueden utilizar con HTTP sobre TCP (*Transmission Control Protocol*) en el puerto de red 80. El puerto 80 es el puerto que queda libre de los firewalls para los navegadores web, por eso precisamente lo utlizan los servicios Web.

Otra cualidad que destacar de los servicios Web es que pueden aportar gran independencia entre la aplicación que usa el servicio y el propio servicio.

Entre los diversos estándares posibles, interesan particularmente dos: SOAP (*Simple Object Access Protocol*) y REST (*REpresentational State Transfer*). Estos estándares trabajan a través del protocolo HTTP (*HyperText Transfer Protocol*). [12]

### 2.5.1 SOAP.

Se trata de un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. Básicamente es un paradigma de mensajería de una dirección sin estado, que puede ser utilizado para formar protocolos más complejos y completos según las necesidades de las aplicaciones que lo implementan. Se basa en tres partes fundamentales:

- *Envelope* (“envoltura”): Define qué hay en el mensaje y cómo se debe procesar.
- Conjunto de reglas de codificación para expresar instancias de tipos de datos.

- Convención para representar llamadas y respuestas a procedimiento.

Se define por tres características principales:

- Extensibilidad: seguridad y WS-routing son extensiones aplicadas en el desarrollo.
- Neutralidad: bajo protocolos de transporte TCP puede ser utilizado sobre HTTP, SMTP o JMS.
- Independencia: permite cualquier modelo de programación

### 2.5.2 REST.

Se trata de una arquitectura software para sistemas hipermedia distribuidos como la *World Wide Web* (WWW). Fue desarrollado por uno de los principales autores de la especificación del protocolo HTTP, Roy Fielding[1], y actualmente es ampliamente utilizado por la comunidad de desarrollo.

El término REST se utiliza hoy en día para describir cualquier interfaz entre sistemas que utilice directamente HTTP para obtener datos o indicar la ejecución de operaciones sobre los datos, en cualquier formato, sin las abstracciones de los protocolos basados en patrones de intercambio de mensajes, como SOAP.

Fundamentalmente se caracteriza por lo siguiente:

- Un protocolo cliente/servidor sin estado: Cada mensaje HTTP contiene toda la información necesaria para comprender la petición. Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes.
- Un conjunto de operaciones bien definidas que se aplican a todos los recursos de información: HTTP en sí define un pequeño conjunto de operaciones, las más importantes son POST, GET, PUT y DELETE.
- Una sintaxis universal para identificar los recursos: cada recurso es direccionable únicamente a través de su URI (*Uniform Resource Identifier*).

[1] Roy Thomas Fielding (1965): Fue uno de los principales autores de la especificación del protocolo HTTP y fundador de la arquitectura REST. También estuvo implicado en el desarrollo de HTML, fue uno de los fundadores del servidor web Apache. [15]

- El uso de hipermedios, tanto para la información de la aplicación como para las transiciones de estado de la aplicación: la reresentación de este estado en este sistema son típicamente HTML ó XML, es decir, que se puede navegar de un recurso REST a otros a partir de enlaces, sin requerir del uso de registros u otra infraestructura adicional. [14]

### 2.5.3 Conclusión.

Por la versatilidad y potencialidad del protocolo tipo **REST**, será el que se utilice para desarrollar las comunicaciones entre la base de datos y la interfaz HTML que se desarrolle.

## 2.6 Diseño de interfaces de usuario.

Lo más cómodo para acceder a la información de forma sencilla y rápida es una interfaz gráfica, que permita acceder fácilmente a los datos que se necesiten y también modifcar la red de nodos, si fuera necesario.

Se pueden desarrollar interfaces de diferentes maneras, y de acuerdo con la forma de acceder a ella se pueden diferenciar entre: App nativa para Smartphone/Tablet, para PC o HTML5 responsiva.

Las aplicaciones responsivas permiten que se pueda acceder a la interfaz desde cualquier dispositivo, ya sea smartphone, tablet o PC. Todos los elementos que se incluyan se adaptarán automáticamente al dispositivo desde el que se acceda.

Para desarrollar una aplicación responsiva se pueden utilizar diversidad de alternativas, como por ejemplo Bootstrap ó HTML5 Boilerplate.

### 2.6.1 Bootstrap.

Es uno de los proyectos Open Source más populares en Github. Simplifica el proceso de creación de diseños Web combinando CSS y JavaScript. Fue desarrollado inicialmente por Twitter.

La última plantilla desarrollada se denomina Jumbotron. Incluye unos pocos elementos básicos: menú de navegación, bloque principal y el sistema de

12 columnas de Jumbotron, a través del cual se pueden contemplar 12 columnas en la página o combinaciones de ellas de forma libre. [16]

### 2.6.2 HTML5 Boilerplate.

La mayor diferencia respecto a Bootstrap es que se trata de una plantilla, no de un Framework. Es también bastante popular en la plataforma GitHub.

H5BP está pensado para desarrollar rápidamente una interfaz, sin entrar en funciones complejas o diseños detallados. [17]

### 2.6.3 Conclusión.

Para realizar aplicaciones sencillas y de forma rápida se recomienda el uso de HTML5 Boilerplate, sin embargo, si se pretende realizar algo más complejo de mediana embergadura se recomienda entonces utilizar Bootstrap. Por tanto, se utilizará **Bootstrap** para desarrollar la interfaz necesaria, ya que se neceritarán algunas funciones complejas para realizar ciertas acciones.

# Capítulo 3

# Software desarrollado

## 3.1 Configuración de Raspberry Pi.

Como paso inicial, una vez instalado el sistema operativo, que se puede descargar directamente de la página web de Raspberry, hay que instalar una serie de archivos en la plataforma y configurar ciertos parámetros.

Lo primero es configurar el acceso a través de SSH, la fecha y hora, y expandir la memoria SD para aprovechar todo el espacio. Todo esto se hace desde la propia configuración del sistema.

A continuación, hay que instalar ciertos archivos necesarios para la programación Python, como son las librerias *Serial* y *MySQLdb*. Para que el programa se ejecute al encender el dispositivo, habrá que incluirlo en el archivo "rc.local", que se ejecuta al arrancar.

Posteriormente, hay que instalar *apache*, *mysql* y *phpmyadmin* para configurar la base de datos y los servicios web. Cuando se instalen, habrá que introducir una clave para el usuario "root" de phpMyAdmin. Para que todo funcione correctamente dicah clave tiene que coincidir con la incluída en la programación de los servicios web, "raspberry". También habrá que modificar el archivo "apache2.conf" para habilitar los permisos de acceso remoto cuando se acceda desde cualquier otro terminal a las interfaces, e incluir también permisos para el acceso a la configuración de phpMyAdmin (será necesario para importar la base de datos que hayamos desarrollado).

Para finalizar sólo hay que importar los archivos que se desarrollen a través de comunicación SSH mediante algún programa. En este caso, se ha realizado mediante winSCP.

Habrá que asignarle una IP fija a la Raspberry e incluirla en los archivos que se expondrán a continuación. Para hacerlo hay que modificar el fichero "dhcpcd.conf" por la parte del final, donde se deberá incluir la subred a la que pertenece, una máscara de red y la dirección de IP estática.

## 3.2 Desarrollo de software.

Una vez configurada la plataforma hardware sobre la que se va a trabajar, el siguiente paso es desarrollar el software necesario, para ello se dividirá en las cuatro partes principales a desarrollar:

- Base de datos del tipo Entidad-Relación: Servirá para almacenar los datos recogidos teniendo perfectamente identificados todos los parámetros: el sensor que mide el dato ("tipo_sensor"), el nodo al que pertenece dicho sensor ("nodo"), la instalación en la que se encuentra ("instalación"), el valor medido ("dato_canal") y las unidades en las que se está midiendo ("unidad"). Además, se dispondrá de otras tablas que aportarán información importante sobre la red de nodos y ayudarán a que todas las tablas queden perfectamente relacionadas.
- Lector de tramas: Un programa que lea el puerto Serie (USB) por el que se recibirán las tramas enviadas por los nodos (vía ZigBee), las interpretará e insertará los datos en la tabla "dato_canal" (tabla en la que se almacenarán todos los datos recibidos) de la base de datos diseñada. Si detectara algún error, se almacenaría en un archivo "log.txt" que se creará mensualmente, para ser capaces de analizar si algo está fallando en la comunicación, el nodo o los sensores.
- Interfaz HTML 5 responsiva: A través de ella se podrá editar la red (instalaciones, nodos, sensores…) y obtener datos concretos que se necesiten, introduciendo los sensores y el período temporal que se pretenda monitorizar.

- Servicios web para conectar con la base de datos: A través de la interfaz, se hace uso de estos servicios para poder comunicarse con la base de datos de forma segura y precisa.

## 3.3 Base de datos.

Hay que desarrollar una base de datos del tipo entidad-relación de manera que se puedan tener identificados todos los elementos que intervienen en el proceso, además de poder almacenar información adicional que pueda ser de interés. Todos estos elementos debén de estar relacionados entre sí de forma coherente para que se puedan realizar consultas complejas a través de los elementos de conexión. Así pues, la estructura que se ha definido para este proyecto es la siguiente:

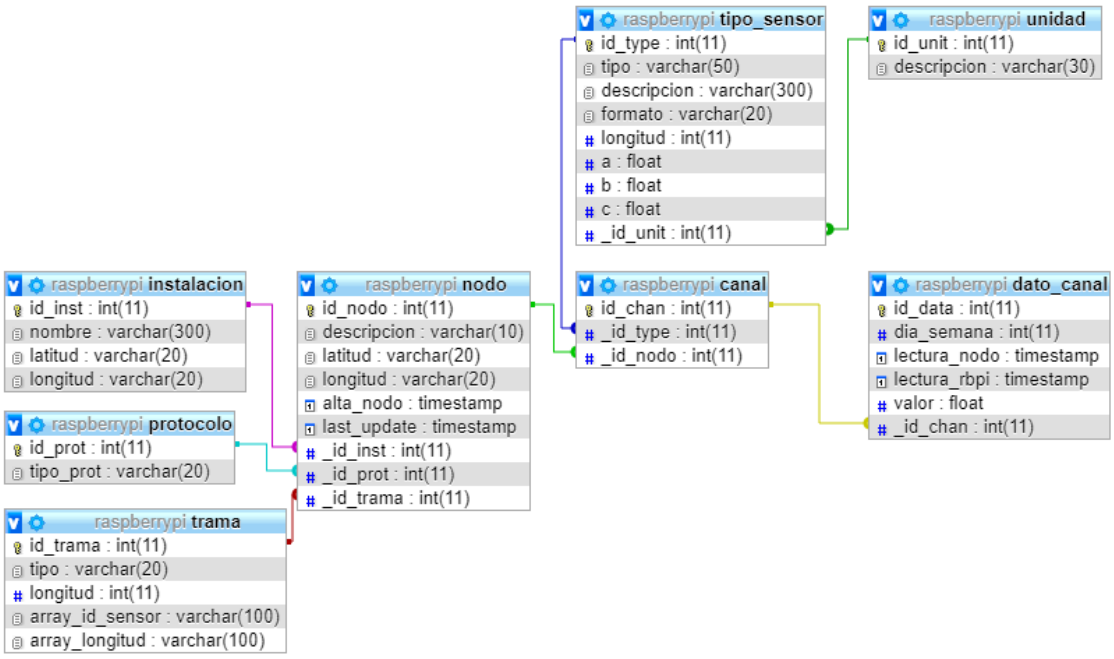


*Figura 3.1 Diseño de la base de datos*

Tabla “instalacion”:

En esta tabla se almacenará la información de las instalaciones operativas que se controlen. El significado de cada columna es el siguiente:

- *id_inst*: Corresponde a un índice que se establece de manera automática para identificar las distintas instalaciones. Se utiliza como clave primaria y está relacionada (1:N) con la clave foránea *_id_inst* de la tabla “nodos”.

- *nombre*: Almacenaría el nombre de la instalación, por ejemplo: “Finca de Jumilla”.
- *Latitud y longitud*: Coordenadas geográficas que almacenan la ubicación de la instalación.

Tabla “protocolo”:

En esta tabla se almacenarán los diferentes protocolos de comunicación que pueden utilizar los nodos, en principio los establecemos fijos: Wi-Fi, IEEE 802.15.4 y ZigBee. El significado de las columnas es el siguiente:

- *id_prot*: Corresponde a un índice que se establece de manera automática para identificar los diferentes protocolos. Se utiliza como clave primaria y está relacionada (1:N) con la clave foránea *_id_prot* de la tabla “nodos”.
- *tipo_prot*: Almacena el nombre del protocolo, por ejemplo: “Wi-Fi 802.11”.

Tabla “trama”:

En esta tabla se almacenarán los diferentes tipos de trama que se pueden encontrar y la información que contiene cada una de ellas. Las columnas indican lo siguiente:

- *id_trama*: Corresponde a un índice que se establece de manera automática para identificar las distintas tramas. Se utiliza como clave primaria y está relacionada (1:N) con la clave foránea *_id_trama* de la tabla “nodos”.
- *tipo*: Almacena el nombre de la trama, por ejemplo: “C30001”. Puede haber tramas con el mismo nombre pero que tengan diferente configuración de sensores, con lo que se almacenarían como dos tipos de trama distintos.
- *longitud*: Almacena el tamaño de la trama, en cuanto a la cantidad de caracteres que contiene.
- *array_id_sensor*: Almacena una cadena de caracteres compuesta por los índices correspondientes a los sensores que mandan

información a través de la trama, separados un índice de otro por una coma (“,”). Por ejemplo: “1,2,3,5,6,7”.

- *array_longitud*: Como el anterior, almacena una cadena compuesta por las longitudes de las medidas que envían los sensores, ordenados de forma respectiva con la cadena de índices anterior. Por ejemplo: “9,5,9,5,4,4”.

Tabla “nodo”:

En esta tabla se almacenarán todos los nodos que se registren. Las columnas indican lo siguiente:

- *id_nodo*: Corresponde a un índice que se establece de manera automática para identificar los nodos de nuestra red. Se utiliza como clave primaria y está relacionada (1:N) con la clave foránea *_id_nodo* de la tabla “canal”.
- *descripcion*: Almacena el nombre del nodo, por ejemplo: “Nodo 1”.
- *latitud* || *longitud*: Coordenadas geográficas que almacenan la ubicación del nodo.
- *alta_nodo*: En esta columna se almacena la fecha y la hora en la que se dio de alta el nodo.
- *last_update*: Aquí se almacena la fecha y hora del último dato recibido por el nodo.
- *_id_inst* || *_id_prot* || *_id_trama*: Son las claves foráneas que relacionan las tablas anteriores con ésta. Indican la instalación a la que pertenece el nodo, el protocolo de comunicación que utiliza y el tipo de tramas que envía, respectivamente.

Tabla “tipo_sensor”:

En esta tabla se almacenarán los diferentes sensores que tengan instalados los nodos. Se considera distinto también cuando un mismo sensor envía dos medidas, habrá que registrar este sensor como si fuera dos distintos, “Sensor Medida 1” y “Sensor Medida 2”. Las columnas indican lo siguiente:

- *id_type*: Corresponde a un índice que se establece de manera automática para identificar los diferentes sensores. Se utiliza como

clave primaria y está relacionada (1:N) con la clave foránea *_id_type* de la tabla “canal”.

- *tipo*: Aquí se guardará una abreviatura para referirse al sensor, por ejemplo: “PM25cm”. Esto servirá para simplificar espacio en la interfaz y en los títulos cuando se obtienen los datos.
- *descripcion*: Almacena la descripción sobre el sensor, por ejemplo: “Potencial Matricial a 25 cm”.
- *formato*: Esta columna almacena el formato de la medida que se va a enviar luego en la trama, por ejemplo: “+00.00000”.
- *longitud*: Esta columna está relacionada diréctamente con la anterior, pues almacenará la longitud de caracteres de la cadena *formato*, en el caso ejemplo: “9”.
- *a* || *b* || *c*: Estas tres columnas almacenan parámetros de calibración de los sensores de acuerdo con una ecuación del tipo: “$a \cdot x^2 + b \cdot x + c$”, donde “x” corresponde a la medida enviada por el sensor y el resultado de la ecuación corresponderá al valor real.
- *_id_unit:* Clave foránea que relaciona la tabla “unidad” con ésta. Indicará la unidad en la que mide el sensor en cuestión.

Tabla “unidad”:

En esta tabla se almacenarán las diferentes unidades utilizadas por los sensores para medir. Las columnas indican lo siguiente:

- *id_unit*: Corresponde a un índice que se establece de manera automática para identificar las distintas unidades. Se utiliza como clave primaria y está relacionada (1:N) con la clave foránea *_id_unit* de la tabla “tipo_sensor”.
- *descripcion*: Almacena el símbolo o abreviatura de la unidad, por ejemplo: “kPa”.

Tabla “canal”:

En esta tabla se almacenarán los canales utilizados por los sensores de cada nodo, es decir, cada nodo dispondrá de un canal por cada sensor que posea. Las columnas indican lo siguiente:

- *id_chan*: Corresponde a un índice que se establece de acuerdo con el formato: "*_id_nodo* · 100 + *_id_type"*. Por ejemplo, para el sensor "PM25cm", que tiene índice (*id_type*): "1", del "Nodo 1" su *id_chan* = "101". Se utiliza como clave primaria y está relacionada (1:N) con la clave foránea *_id_chan* de la tabla "dato_canal".
- *_id_type* || *_id_nodo*: Son las claves foráneas que relacionan las tablas "tipo_sensor" y "nodo" con ésta. Indican el sensor y el nodo, respectivamente, a los que corresponde ese canal.

Tabla "dato_canal":

En esta tabla se almacenarán todos los datos recibidos por la Raspberry Pi. Las columnas indican lo siguiente:

- *id_data*: Corresponde a un índice que se establece de manera automática para identificar los datos recibidos. Se utiliza como clave primaria.
- *dia_semana*: Almacena el día de la semana en el que el nodo envío la trama. Se expresa mediante un número entero entre 1 y 7, estableciendo el inicio de semana en Domingo (1).
- *lectura_nodo*: Se almacena la fecha y hora en la que el nodo envío la trama.
- *lectura_rbpi*: Se almacena la fecha y hora en la que la Raspberry Pi recibió la trama.
- *valor*: Almacena el valor medido por el sensor en cuestión, ya calibrado con la ecuación correspondiente para dicho sensor.
- *_id_chan*: Clave foránea que relaciona la tabla "canal" con ésta. Indicará el canal al que corresponde el valor asociado.

Por tanto, cuando la Raspberry Pi reciba una trama, ésta la seccionará y almacenará uno a uno los valores asociándoles el canal que le corresponda a cada uno. Por ejemplo, si se captara una trama con 7 medidas, se crearían 7 filas en la tabla "dato_canal".

La base de datos está desarrollada a través de phpMyAdmin desde un PC con cotejamiento "utf8_spanish_ci". Para instalarla en la Raspberry Pi hay que

exportarla del *localhost* del PC e importarla en el *localhost* de la Raspberry, a través del software de phpMyAdmin instalado en la SBC. Para importarla hay que crear primero una base de datos nueva vacía y sobrescribirla con la base de datos en cuestión.

## 3.4 Lector de tramas.

Para realizar el programa de lectura de puertos se empleará el lenguaje Python, que es un lenguaje muy sencillo de programar y tiene librerías (MySQLdb) para trabajar con bases de datos del tipo SQL. La librería que se empleará para la lectura del puerto serie será “Serial”, y se creará una simulación de tramas a través de Arduino UNO para comprobar la correcta lectura y recepción de datos a través del puerto serie. *(ver Figura 3.2)*

El programa que ejecutará Arduino será similar al mostrado a continuación, pero añadiendo un contador y dos o tres tramas más:

```
int cont=0;
void setup() {
  // Se inicia la comunicación serie.
  Serial. begin(9600);
}
void loop() {
  // Añadimos una espera entre el envío de una trama y otra.
  delay(15000);
  Serial.println(":010AD30001-000732.6+25.4-000663.0+23.73+23.76368+17.34899+16.426358AEFFF3.974700004200717");
  cont++;
}
```

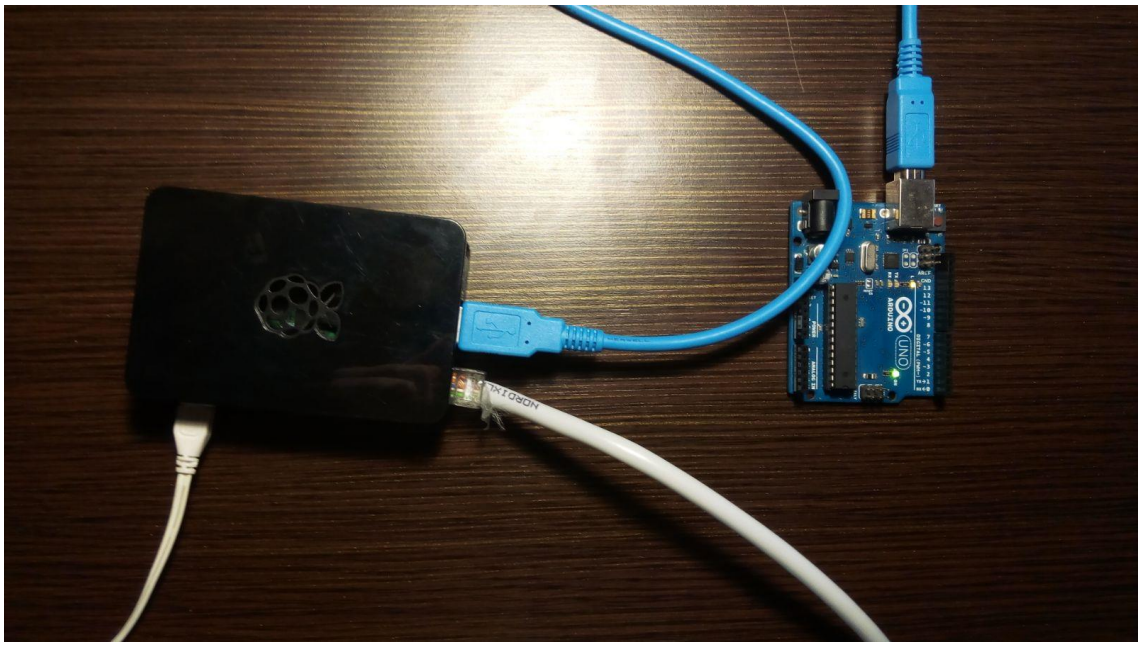

*Figura 3.2 Raspberry Pi conectada a Arduino UNO*

Mientras, el código del programa en Python que la Raspberry ejecutará para monitorizar el puerto serie será el siguiente:

- Cabecera: Se importan las librerías y se establece el puerto serie y las variables de conexión con la base de datos.

```
# coding=utf-8
import MySQLdb
import serial
import time

# Establecemos el puerto serie que recibirá los datos.
puerto_serie = serial.Serial('/dev/ttyACM0', baudrate = 9600, timeout = 0.5)
lectura_puerto = ""

# Introducimos los datos de nuestra base de datos.
DB_HOST = 'localhost'
DB_USER = 'root'
DB_PASS = 'raspberry'
DB_NAME = 'raspberrypi'
```

- Función para ejecutar las consultas: A través de esta función se ejecutan consultas de forma sencilla y genérica. Si la consulta es del tipo *SELECT* enviará el resultado de la misma.

```
def run_query(query=""):

    datos = [DB_HOST, DB_USER, DB_PASS, DB_NAME]
    conn = MySQLdb.connect(*datos)  # Conecta con la base de datos.
    cursor = conn.cursor()  # Crea un cursor.
    cursor.execute(query)  # Ejecuta una consulta.
    if query.upper().startswith('SELECT'):
        data = cursor.fetchall()  # Trae los resultados de un SELECT.
    else:
        conn.commit()  # Hace efectiva la escritura de datos.
        data = None

    cursor.close()  # Cierra el cursor.
    conn.close()  # Cierra la conexión.

    return data
```

- Función para obtener la fecha: Esta función sirve para obtener la fecha en el formato *TIMESTAMP* requerido para insertarla en la base de datos. También se redondearán los valores para que sólo haya valores en los minutos 15, 30, 45 y 00, por lo que los segundos siempre serán 00.

```
def process_date(fecha=""):
    seg_int = int(fecha[0:2])
    minu_int = int(fecha[2:4])
    hor_int = int(fecha[4:6])
    dia_sem_int = int(fecha[6])
    dia_int = int(fecha[7:9])
```

```
    mes_int = int(fecha[9:11])
    year_int = int(fecha[11:13])

    if seg_int >= 30:  # Redondeamos los segundos.
        minu_int += 1
    seg = "00"  # Redondeamos a los minutos, por tanto segundos = 00.

    if minu_int < 8:  # Redondeamos los minutos a 00, 15, 30 y 45.
        minu = "00"
    elif minu_int >= 8 and minu_int < 23:
        minu = "15"
    elif minu_int >= 23 and minu_int < 38:
        minu = "30"
    elif minu_int >= 38 and minu_int < 53:
        minu = "45"
    elif minu_int >= 53:
        minu = "00"
        hor_int += 1

    if hor_int > 23:  # Redondeamos la hora.
        hor = "00"
        dia_sem_int += 1
        dia_int +=1
    else:
        hor = str(hor_int).zfill(2)  # Esta función nos rellena los huecos con el carácter 0.

    if dia_sem_int > 7:
        dia_sem_int = 1

    if dia_int > 28 and mes_int == 2 and year_int % 4 != 0:  # Corregimos la fecha
        dia_int = 1
        mes_int = 3
    elif dia_int > 31 and mes_int == 12:
        dia_int = 1
        mes_int = 1
        year_int += 1
    elif dia_int > 30 and (mes_int == 4 or mes_int == 6 or mes_int == 9 or mes_int == 11):
        dia_int = 1
        mes_int += 1
    elif dia_int > 31 and (mes_int==1 or mes_int==3 or mes_int==5 or mes_int==7 or mes_int==8 or\
mes_int==10):
        dia_int = 1
        mes_int += 1

    dia = str(dia_int).zfill(2)
    mes = str(mes_int).zfill(2)

    if year_int > 99:  # Redondeamos el año, solo podemos tener en cuenta un siglo por el formato.
        year = "00"
    else:
        year = str(year_int).zfill(2)
    date = "20" + year + "-" + mes + "-" + dia + " " + hor + ":" + minu + ":" + seg
    return (date, dia_sem_int)  # Aplicamos el formato TIMESTAMP y añadimos el día de la semana.
```

- Programa principal: Se ejecuta dentro de un bucle infinito (while True). Se le añade un pequeño retardo de una décima de segundo para que no se sature el programa, y la condición interna de ".inWaiting()", para que sólo ejecute el resto del código cuando el puerto detecte alguna trama. Se comienza creando un archivo de texto *"log"*[2] dónde se almacenarán los errores que puedan surgir durante la

[2] Sólo se crea el fichero la primera vez, el resto de veces se edita añadiendo contenido.

lectura y se establece la longitud máxima de lectura en función del mayor tamaño de trama registrado.

```
while True:
   time.sleep(0.1)
   while puerto_serie.inWaiting() > 0:

      # Creamos un archivo log para almacenar posibles errores que puedan surgir.
      log = open("log_" + time.strftime("%y%m") + ".txt", "a")

      query = "SELECT MAX(longitud) FROM trama"  # Seleccionamos el mayor tamaño de trama.
      result = run_query(query)

      lectura_puerto += puerto_serie.read(result[0][0])  # Lectura de la trama.
      # lectura_puerto +=puerto_serie.readline()
      lectura_puerto = lectura_puerto.strip()
```

A continuación, se comprueba que la trama comienza correctamente con ":01". Si es así, se obtiene el nodo (en la trama estará en formato hexadecimal) y, a partir de él, el tipo y longitud de trama, y, los ID y longitudes de los sensores que contiene. Si no es así, devuelve un error al archivo *log*.

```
      if lectura_puerto.startswith(":01"):  # Comprobamos si el comienzo de trama es correcto
         nodo = int(lectura_puerto[3:5], 16)  # Seleccionamos el número de nodo

         # Obtenemos Longitud y Tipo de trama e ID y Longitudes de sensores, a partir del nodo.
         query = "SELECT trama.tipo, trama.longitud, trama.array_id_sensor, \
trama.array_longitud FROM trama JOIN nodo ON trama.id_trama=nodo._id_trama WHERE nodo.id_nodo=%i"\
% nodo
         result = run_query(query)

         tipo = result[0][0]
         longitud = result[0][1]
         array_id = result[0][2]
         array_long = result[0][3]
         fin_lect_trama = len(tipo) + 5  # Lo utilizaremos para comenzar a incrementar
```

Se utilizará como marca de inicio el tamaño del tipo de trama, incrementándole el comienzo de trama y el nodo, por ejemplo: para el "Nodo 5" con tipo de trama "C30001" sería: ":0105C30001", luego marca de inicio = 11, (6 de la trama + 5 de ":0105"). Lo que hará el programa después será ir incrementando las longitudes de medida de los diferentes sensores y utilizar el ID para insertar los datos del sensor correspondiente en la base de datos. Antes de eso, se comprobará si la longitud de la trama recibida coincide con la longitud de la trama del nodo en cuestión, además también se comprobará si coincide el tipo de trama. Si algo no coincidiera, se almacenaría un error en el archivo *log*.

```
        if len(lectura_puerto) == longitud:
            if lectura_puerto[5:fin_lect_trama] == tipo:
                # Lectura de datos.
                fecha = lectura_puerto[-13:]
                date = process_date(fecha)
                fecha_sql = date[0]
                dia_sem = date[1]
                ides = array_id.split(",")
                longs = array_long.split(",")

                # Insertamos todos los datos de una vez, elaborando la consulta por partes.
                query = "INSERT INTO dato_canal (id_data, dia_semana, lectura_nodo, \
lectura_rbpi, valor, _id_chan) VALUES "
                cont = 0
                flag = 0
                ini = fin_lect_trama
                fin = fin_lect_trama
                while cont < len(ides):
                    # El EnviroSCAN tiene una marca a mitad de trama que hay que considerer.
                    if (ides[cont]=='5' or ides[cont]=='6' or ides[cont]=='7') and flag==0:
                        ini += 1
                        flag = 1
                    fin = ini + int(longs[cont])  # Mara final de lectura para el sensor.

                    # Los dendrometros dan el valor en hexadecimal.
                    if (ides[cont] == '8' or ides[cont] == '9'):
                        valor = int(lectura_puerto[ini:fin], 16)
                    else:
                        valor = float(lectura_puerto[ini:fin])

                    # Corregimos el valor medido con los parámetros de calibracion.
                    calibracion = "SELECT %f * %f * a + %f * b + c FROM tipo_sensor WHERE \
id_type = %i" % (valor, valor, valor, int(ides[cont]))
                    valor_sql = run_query(calibracion)

                    # Establecemos el id del canal correspondiente.
                    idchan = nodo * 100 + int(ides[cont])
                    query += "(NULL, %i, '%s', CURRENT_TIMESTAMP, %f, %i)" \
% (dia_sem, fecha_sql, valor_sql[0][0], idchan)
                    cont += 1
                    if cont < len(ides):
                        query += ", "
                    ini = fin

                run_query(query)

                # Se actualiza la fecha a la que el nodo recibio el ultimo dato.
                actualizar = "UPDATE nodo SET last_update = '%s' WHERE id_nodo = %i" \
% (fecha_sql, nodo)
                run_query(actualizar)

            else:
                # Error, Tipo de trama mal.
                log.write(time.strftime("%X %A, %d %B %Y") +"\nError. El tipo de trama debería\
ser " + tipo + ", que es el registrado para el nodo " + str(nodo) + "\n" + lectura_puerto + \
"\n\n\n")
        else:
            # Error, trama tam. distinto a lo previsto, posiblemente falle un sensor.
            log.write(time.strftime("%X %A, %d %B %Y") +\
            "\nError. Trama de tam. distinto a lo esperado, posiblemente por el fallo de un sensor.\
            Habria que modificar el Nodo " + str(nodo) + "\n" + lectura_puerto + "\n\n\n")
    else:
        # Error, la trama no comienza como debería.
        log.write(time.strftime("%X %A, %d %B %Y") +\
        "\nLa trama no comienza por :01\n" + lectura_puerto + "\n\n\n")
    lectura_puerto = ""
    log.close()
```

Y así concluiría el programa de lectura de tramas. Se ha verificado la robustez de dicho módulo mediante el envío tanto de tramas correctas, como de tramas con errores desde una placa Arduino UNO. Para ello, ha sido necesario diseñar y desarrollar el Sketch .ino correspondiente.

Se ha creado también un programa idéntido al lector de tramas, pero, en lugar de acceder a las tramas recopiladas del puerto serie, accede a las tramas almacenadas en un fichero TXT. Estas tramas son recogidas de un campo real, para comprobar el correcto funcionamiento del programa.

## 3.5 Interfaz en HTML 5 responsiva.

Se ha desarrollado una interfaz de comunicación responsiva con el framework Bootstrap, para que, al acceder desde cualquier terminal, no varíe la estructura y se puedan controlar y visualizar todos los elementos correctamente. Se dividirá en dos páginas principalmente: Una desde la que se accede a la información que se pretenda obtener y la otra para configurar la base de datos.

En cuanto a la página para la "**Comunicación"**, tiene un aspecto visual como el representado en la siguiente figura (*ver Figura 3.3*):


*Figura 3.3 Página de Comunicación*

Para poder obtener los datos que queramos primero se deberán seleccionar los nodos que se quieren observar, después los sensores de los que se quiera obtener datos y posteriormente un intervalo de fechas. Después, se pulsa el botón *"*Obtener .csv*"* para obtener un archivo CSV (separado por ";" y utilizando "." para los decimales) con la siguiente distribución:

| Día semana (Número) | Fecha de lectura | Sensor X del Nodo i (Unid.) | Sensor X del Nodo j (Unid.) | … | Sensor Y del Nodo i (Unid.) | Sensor Y del Nodo j (Unid.) | … |
|---|---|---|---|---|---|---|---|

*Tabla 3-1 Cabecera del archivo .csv*

Aquellos valores que no dispongan de datos en alguna fecha concreta aparecerán como "0" o vacíos. Los datos estarán ordenados por fecha, de más antiguo a más actual.

Los títulos de las columnas de valores vendrán dados con el siguiente formato: "Nodo X *Abreviatura del sensor* (Unidades en las que mide)", donde *X* corresponde al número del nodo.

El código completo en HTML 5 se adjunta en el **Anexo I** del documento, ya que principalmente corresponde a un diseño se puede comprobar mejor con las imágenes mostradas. Para realizar el diseño se han tenido que programar ciertas funciones en JavaScript y jQuery, pero esta parte del proyecto no corresponde al objetivo principal y por eso se adjunta el código fuente.

Para que el código funcione adecuadamente habrá que editar en el comienzo de los ficheros JS una variable llamada "ip". En esta variable se define la dirección IP de la Raspberry para que se pueda acceder a las interfaces de manera remota mediante dicha ip.

Por otro lado, la página para la "**Configuración"** de la base de datos, tiene un aspecto como el representado en la siguiente figura:

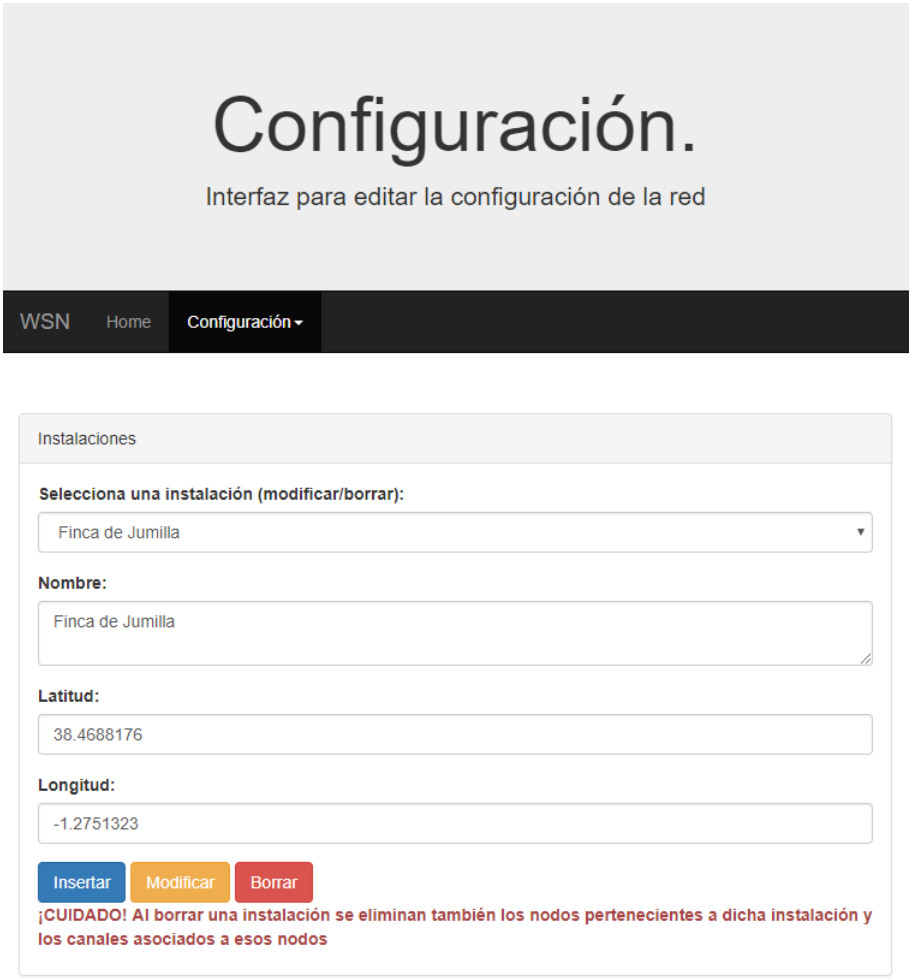


*Figura 3.4 Página para la configuración de la Base de Datos - 1 (instalaciones)*

La primera diferencia apreciable con respecto a la anterior página es la disponibilidad de un desplegable que permite acceder rápidamente al módulo o bloque concreto que se quiera: Instalaciones, Nodos, Sensores o Unidades.

Se ha comentado previamente que el código se adjuntará en el anexo correspondiente, pero para comprender mejor el funcionamiento de la página a continuación se da una breve explicación sobre ello.

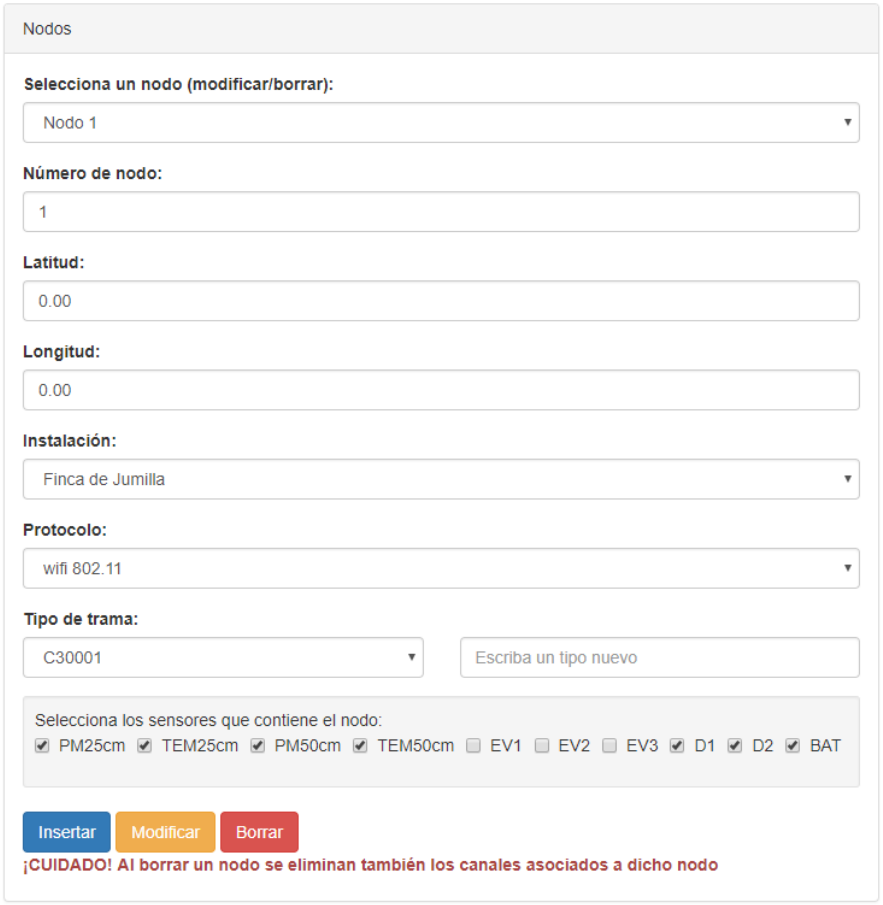


*Figura 3.5 Página para la configuración de la Base de Datos - 2 (nodos)*

Lo primero que hará la página será cargar las listas de instalaciones, nodos, sensores, unidades, tramas y protocolos con los datos almacenados en la base de datos. Si la base de datos está vacía puede que aparezca “Undefined” en todos los campos o que aparezcan vacíos. Para solventar este problema se pueden insertar nuevos datos nuevos desde la interfaz.

Todos los módulos funcionan de manera similar. Se pueden insertar datos rellenando los campos necesarios y pulsando el botón “Insertar” (en azul) colocado en la parte inferior del módulo. Para modificar o borrar datos, bastará con seleccionar el elemento se pretenda editar y pulsar el botón correspondiente, si lo que se quiere es modificar dicho elemento, se tendrá que haber cambiado sus valores previamente. Hay que tener en cuenta que al eliminar un elemento se eliminarán también todos los elementos foráneos relacionados con él.

Cuando se realice alguna de las acciones (Insertar, Modificar o Borrar), aparecerá un mensaje de alerta (*ver Figura 3.6*) debajo del módulo que se haya utilizado, informando sobre si se ha realizado correctamente la operación o ha

surgido algún problema durante la misma, indicando la posible causa de ello. Los mensajes se irán acumulando si no se eliminan (pulsando el icono “X” situado a la derecha del mensaje) o se refresca la página.

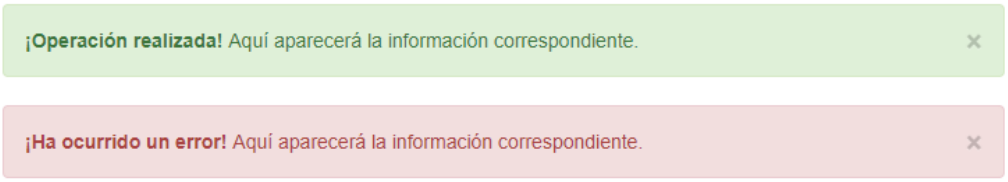


*Figura 3.6 Mensajes de alerta*

Como particularidades de cada módulo, cabría destacar y explicar algunas de ellas:

- *Instalaciones*: Es el más básico, ya que sólo tiene en cuenta el nombre y las coordenadas de la instalación.
- *Nodos*: Las tramas de cada nodo son “independientes”. Cuando se define un nodo hay que definir un tipo de trama, o nombre de trama, que será con lo que comiencen las tramas que envíe este nodo. Este tipo puede ser uno ya existente o uno nuevo a definir. Posteriormente habrá que seleccionar los sensores que contiene dicho nodo y de esta manera se definirá la trama asociada al nodo.
- *Sensores*: La abreviatura del sensor será lo que aparezca cuando se obtengan sus datos y, también, en el módulo de nodos cuando haya que definir los sensores del nodo. El formato corresponde a la forma que tendrá la medida en la trama. La unidad en la que mide el sensor puede ser una existente o una nueva a definir. En cuanto a los parámetros de calibración, si se desconocen habrá que poner “a = c = 0” y “b = 1”, para que:

  $$val.cal. = a \cdot x^2 + b \cdot x + c = x \equiv val.medido$$

  Lo normal es que los sensores tengan asociados algún tipo de recta de calibración para precisar la medida en función de las condiciones de funcionamiento del sensor.
- *Unidades*: Se utilizará para modificar o borrar unidades existentes.

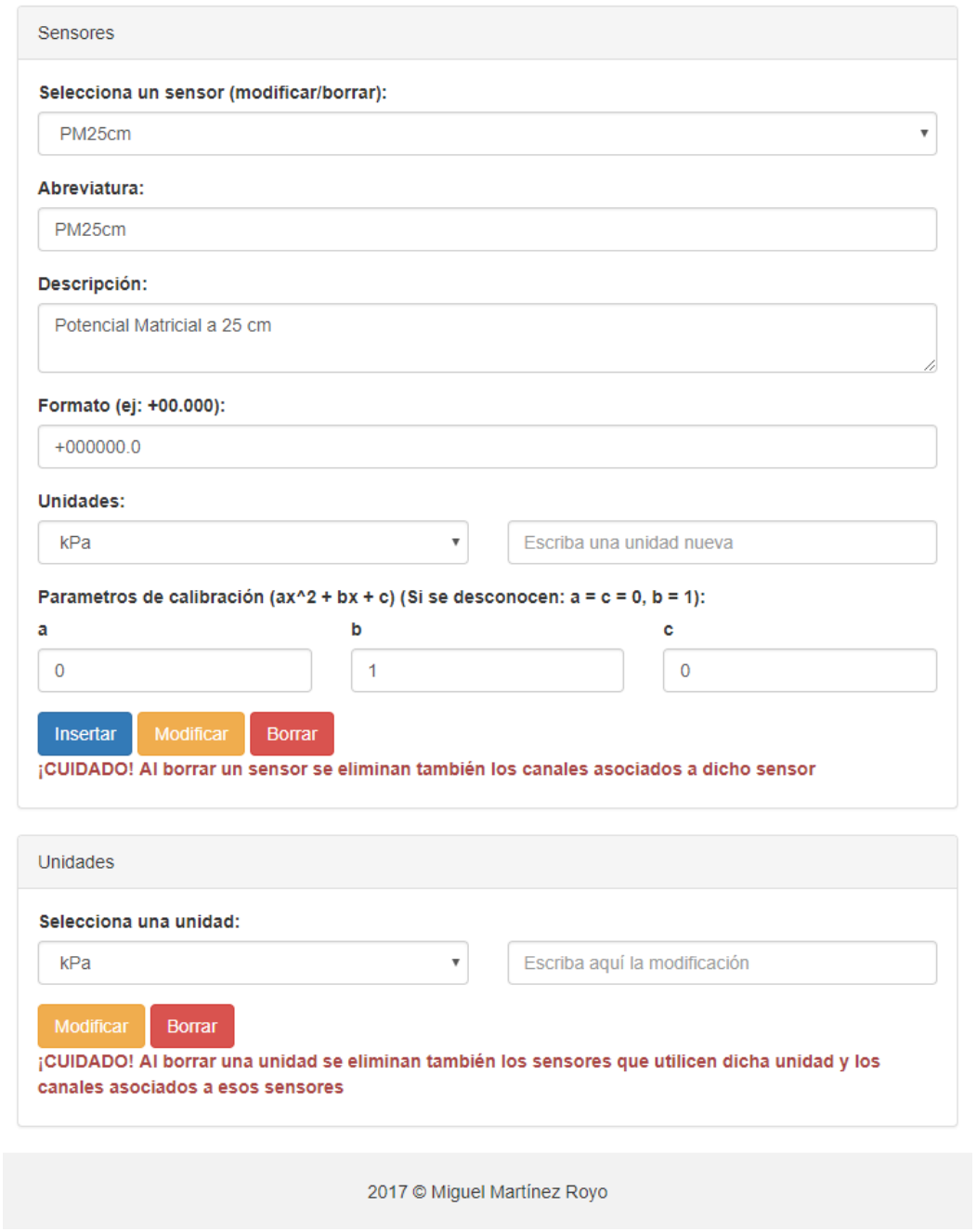


*Figura 3.7 Página para la configuración de la Base de Datos - 3 (sensores y unidades)*

Todas las funciones que hacen posible el funcionamiento de la interfaz están definidas en el Anexo I en los archivos JS (*funciones.js* y *obtenciondatos.js*), donde se aplican funciones en JavaScript y jQuery que permiten la visualización y actualización de los datos.

## 3.6 Servicios Web.

Se establecerá la comunicación entre la base de datos y la interfaz HTML a través de lo que se conoce como Servicios Web. Entre los diversos estándares que se pueden utilizar para estos servicios, se trabajará con el tipo REST (*Representational State Transfer*), que sirve para interactuar con la base de datos a través de peticiones realizadas desde la interfaz.

Para entender un poco más el funcionamiento de estos servicios se explicará de manera breve cómo se trabaja con ellos y después se explicará algún ejemplo conreto del proyecto.

### 3.6.1 Funcionamiento básico.

Un servicio web se basa en un sistema de petición-respuesta entre un cliente y un servidor, en el caso de los de tipo REST se implementan directamente los métodos de petición definidos en el protocolo HTTP, de los cuales se hará uso de GET y POST.

*http://programaenlinea.net/wp-content/uploads/2016/04/rest1.jpg*

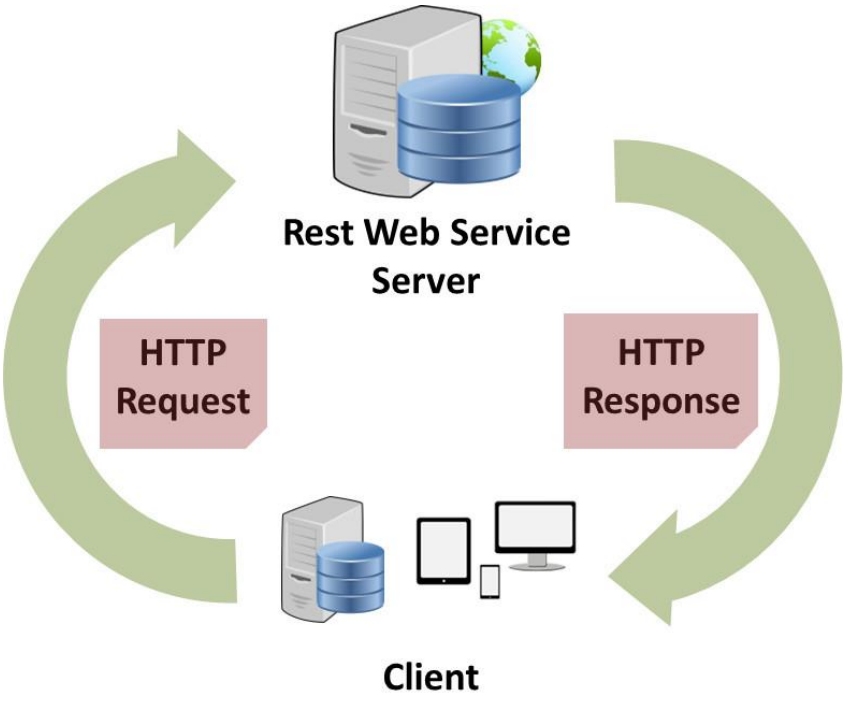


*Figura 3.8 Servicios Web*

A efectos prácticos se podría de alguna manera jugar un poco con la programación y utilizar indistintamente ambos, pero por criterio de coherencia en la programación el método GET, como su nombre indica, se utiliza para peticiones en las que se solicite información y el servidor responda con la información solicitada, mientras que, el método POST se utiliza cuando se envía una petición junto a un mensaje JSON con información que se pretenda procesar en el servidor sin ánimo de recibir una respuesta más que si se ha realizado o no correctamente la petición.

Estos métodos se asocián a direcciones almacenadas en el servidor y registradas en el fichero PHP. Se incluye como primera función en ese fichero, una función básica para comprobar el correcto funcionamiento del servicio, donde se solicita mediante GET que se envíe un saludo al parámetro nombre que se envía implícito en la petición: “GET localhost/wsn/v1/hola/{name}”, si por ejemplo ejecutamos: “GET localhost/wsn/v1/hola/Miguel”, la respuesta será: “Hola, Miguel”. El método GET no se incluye en la dirección, sino que va asociado a ella, porque previamente se ha definido así.

### 3.6.2 Servicios web aplicados al proyecto.

Por la extensión del código, se adjunta el código completo en el **Anexo II** del trabajo y se procederá a explicar la esencia de cómo se establece la comunicación a través de un ejemplo, como es la inserción de datos.

Hay un servicio web de inserción por cada módulo (instalaciones, nodos y sensores). Dichos servicios web se invocan mediante el botón “Insertar” de la interfaz HTML.

Primero se ejecuta la función JavaScript que llama al servicio web. Se utilizará un servicio del tipo POST, ya que se va a mandar la información que se quiere insertar en la base de datos a través de un mensaje JSON.

```
// Inserta una instalación en la base de datos
function insertarinstalacion(){
        xmlhttp = new XMLHttpRequest();
        var url = "http://" + ip + "/wsn/v1/insertarinstalacion";
        xmlhttp.open("POST", url, true);
        xmlhttp.setRequestHeader("Content-type", "application/json");
```

Se establece una petición HTTP con el método POST, tomando como referencia de *localhost* la IP de la Raspberry que estará definida previamente al principio del código.

Con la función ".onreadystatechange" se establece la acción que realiza el programa cuando el servicio web se ejecuta y el cliente recibe la respuesta. En este caso se mostrará un mensaje con el resultado de la petición mediante la función "alerts()" y se actualizarán las listas de las instalaciones mediante la función "obtenerinstalaciones()".

```
xmlhttp.onreadystatechange = function () {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                var jsonresponse = JSON.parse(xmlhttp.responseText);
                if (jsonresponse.message != null){
                        alerts(jsonresponse.error,jsonresponse.message,"#instalacion");
                        obtenerinstalaciones();
                }
        }
}
```

A continuación, se define el cuerpo del mensaje JSON que se envía con la petición del método POST. Este cuerpo funciona como un "diccionario" (en programación), que es básicamente un array en el que los elementos llevan valores asociados a ellos. En este mensaje se incluyen los datos que se vayan insertar del elemento "instalación".

```
        var nombre = document.getElementById("idinstalaciontitle");
        var latitud = document.getElementById("idinstalacionlatitud");
        var longitud = document.getElementById("idinstalacionlongitud");
        var data = JSON.stringify({
                nombre: nombre.value,
                latitud: latitud.value,
                longitud: longitud.value
        });
        xmlhttp.send(data);
}
```

Para nodos y sensores se porocederá de forma similar, cambiando el cuerpo del mensaje JSON con los valores que correspondan.

En cuanto a la programación que ejecutará el servicio, una vez se reciba la petición, será la siguiente:

```
$app->post('/insertarinstalacion', function(Request $request, Response $response) {

    // Obtenemos los parámetros del body
    // Que vienen codificados en JSON
    $bodyPost = $request->getParsedBody();
    $nombre = $bodyPost['nombre'];
    $latitud = $bodyPost['latitud'];
```

```
    $longitud = $bodyPost['longitud'];
    // Parámetros de la base de datos
    $servername = "localhost";
    $username = "root";
    $password = "raspberry";
```

Se define el servicio en la dirección: *http:// [ip]/wsn/v1/instertarinstalacion*. Se obtienen, lo primero, los parámetros del cuerpo del mensaje JSON y se definen los parámetros de conexión con la base de datos. Después se establece la conexión con la misma:

```
    try
    {
            $dbname = "raspberrypi";
            $conn = mysqli_connect($servername, $username, $password, $dbname);
            // Comprobación de la conexión
            if (!$conn) {  // Se devuelve lo siguiente si falla la conexión
                    $data["error"] = true;
                    $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                    $newResponse = $response->withJson($data);
                    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                    return $newResponse;
            }

            [...]

    }
    catch(PDOException $e)
    {
            $data["error"] = true;
            $data["message"] = "Fallo de conexión: " . $e->getMessage();
            $newResponse = $response->withJson($data);
            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
            return $newResponse;
    }
    mysqli_close($conn);
});
```

Si se establece correctamente la conexión, lo primero que se hace es comprobar si la instalación que se pretende insertar existe, si no es así entonces se inserta:

```
            // Comprobar si ya existe la instalación en la tabla
            // SELECT `nombre` FROM `instalacion` WHERE `nombre` = "Finca de jumilla 1"
            // Sólo compruebo que no se repita el nombre de la instalación
            $sql = "SELECT `nombre` FROM `instalacion` WHERE `nombre` = '";
            $sql = $sql . $nombre;
            $sql = $sql . "'";
            $result = mysqli_query($conn, $sql);
            if (mysqli_num_rows($result) > 0) { // True: ya existe y devuelvo un error
                    $msg["error"] = true;
                    $msg["message"] = "La instalación: " . $nombre . " ya existe";
                    $newResponse = $response->withJson($msg);
                    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                    return $newResponse;
            } else {  // Si la instalación no existe la insertamos
                    //Insertar la instalacion
                    //INSERT INTO `instalacion`(`id_inst`, `nombre`, `latitud`, `longitud`) VALUES
(NULL,'nombre','latitud','longitud')
```

```
                $sql = "INSERT INTO `instalacion`(`id_inst`, `nombre`, `latitud`, `longitud`) VALUES (NULL, '";
                $sql = $sql . $nombre;
                $sql = $sql . "', '";
                $sql = $sql . $latitud;
                $sql = $sql . "', '";
                $sql = $sql . $longitud;
                $sql = $sql . "')";
                $result = mysqli_query($conn, $sql);
                if($result == TRUE) {
                        $msg["error"] = false;
                        $msg["message"] = "Instalación: " . $nombre . " insertada correctamente";
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                } else {
                        $msg["error"] = true;
                        $msg["message"] = "Error insertando la instalación: " . $nombre;
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
        }
```

Los servicios tipo GET se estructuran de forma algo diferente, dado que no se envía ningún mensaje JSON:

- En JavaScript:

```
function obtenerinstalaciones(){
        var xmlhttp = new XMLHttpRequest();
        xmlhttp.onreadystatechange = function(){
                if ((this.readyState == 4)&&(this.status == 200)){
                        if (this.responseText != null){
                                jsonresponse = JSON.parse(this.responseText);
                                $("#listaInstalacion").empty();
                                $("#listaInstalacionNodo").empty();
                                if (jsonresponse.message != null){
                                        var miselect = document.getElementById("listaInstalacion");
                                        cargarlistas(jsonresponse.message, miselect);
                                        miselect = document.getElementById("listaInstalacionNodo");
                                        cargarlistas(jsonresponse.message, miselect);
                                        seleccionainstalacion();
                                        seleccionanodo();
                                }
                        }
                }
        }
        xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerlistainstaciones", true);
        xmlhttp.send();
}
```

- En PHP:

```
$app->get('/obtenerlistainstaciones', function(Request $request, Response $response) {

   // Nos conectamos a la base de datos
   $servername = "localhost";
   $username = "root";
   $password = “raspberry”;

   try
   {
        $dbname = "raspberrypi";
```

```
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {
                $data["error"] = true;
                $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                $newResponse = $response->withJson($data);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
        // SELECT `id_inst`, `nombre` FROM `instalacion` WHERE 1
        $sql = "SELECT `id_inst`, `nombre` FROM `instalacion` WHERE 1";
        $result = mysqli_query($conn, $sql);
    $data = array();
        if (mysqli_num_rows($result) > 0) {
                while($row = mysqli_fetch_assoc($result)) {
                        $objeto = array('id' => $row["id_inst"], 'title' => $row["nombre"]);
                        array_push($data, $objeto);
                }
                $msg["error"] = false;
                $msg["message"] = $data;
                $newResponse = $response->withJson($msg);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        } else {
                $msg["error"] = true;
                $msg["message"] = "No hay instalaciones definidas";
                $newResponse = $response->withJson($msg);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
    }
    catch(PDOException $e)
    {
        $data["error"] = true;
      $data["message"] = "Fallo de conexión: " . $e->getMessage();
      $newResponse = $response->withJson($data);
      $newResponse = $newResponse->withHeader('Content-type', 'application/json');
      return $newResponse;
    }
    mysqli_close($conn);
});
```

Se han definido servicios web para todas las interacciones con la base de datos: insertar, modificar y borrar datos, cargar las listas de las interfaces HTML, mostrar los valores del elemento seleccionado y obtener los valores solicitados por el usuario para un archivo CSV.

# Capítulo 4

# Resultados y conclusiones

## 4.1 Pruebas realizadas.

En esta sección se describen los pasos necesarios para configurar una instalación completa en el sistema. En concreto, se definirá una instalación en el "Campo de Cartagena" compuesta por 3 nodos, con la siguiente configuración:

- Nodo 20 [3] : un sensor de humedad del aire, un sensor de temperatura ambiente y un sensor de luminosidad.
- Nodo 21: un sensor de humedad del aire y un sensor de temperatura.
- Nodo 22: un sensor de humedad del aire y un sensor de luminosidad.

El primer paso para configurar la instalación anterior consiste en abrir la interfaz de usuario "Confuguración", en la dirección: http://192.168.0.122/wsn/v1/configuracion.html (en este caso se ha definido como IP estática la mostrada, se podría modificar), e introducir la información mostrada en la *Figura 4.1*:

[3] El sistema ya tiene 10 nodos registrados para la instalación de la Finca de Jumilla, por lo que los índices de los nodos que se pueden insertar serán a partir del Nodo 10.


*Figura 4.1 Inserción de instalación.*

Una vez definida la instalación, el siguiente paso es dar de alta los nodos. Para ello previamente hay que definir los sensores nuevos que se van a utilizar, ya que son nuevos y no están definidos todavía. A modo de ejemplo se muestra en las *Figuras 4.2, 4.3* y *4.4* la información que se tiene que introducir para configurar uno de los sensores y, posteriormente, uno de los nodos:

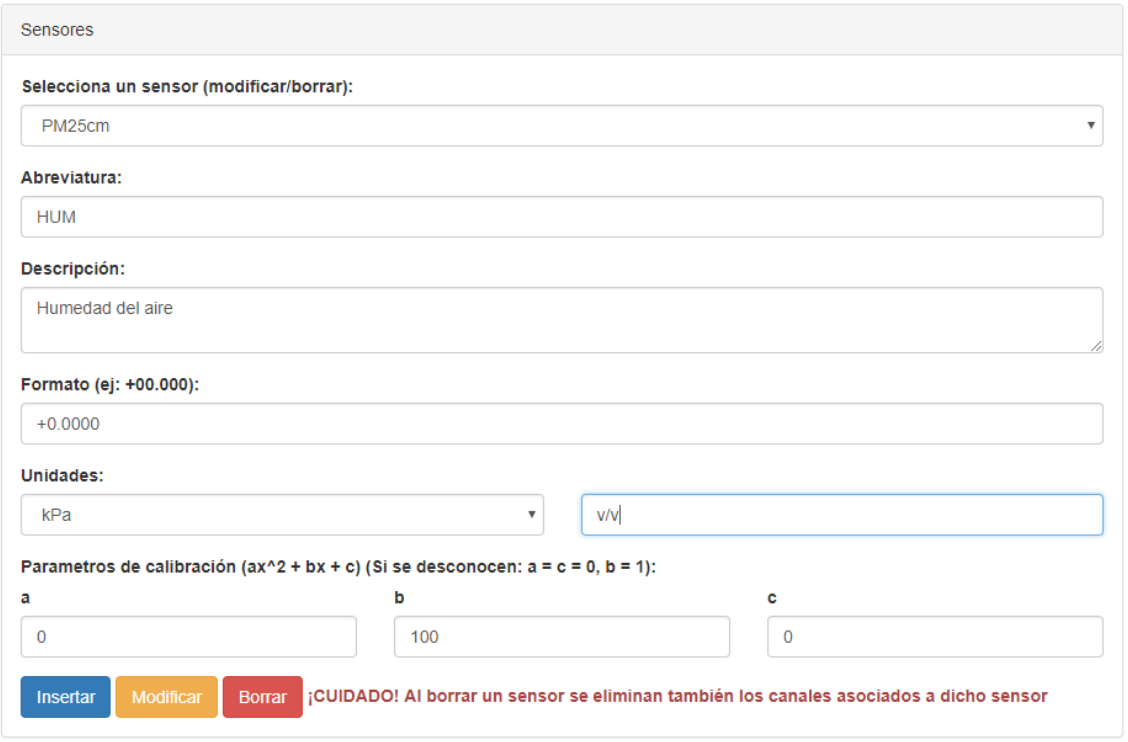


*Figura 4.2 Inserción de sensor.*

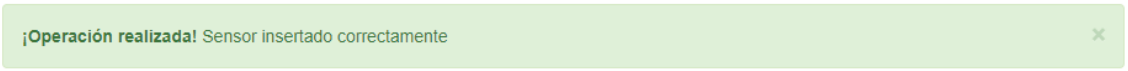


*Figura 4.3 Mensaje de sensor insertado correctamente.*

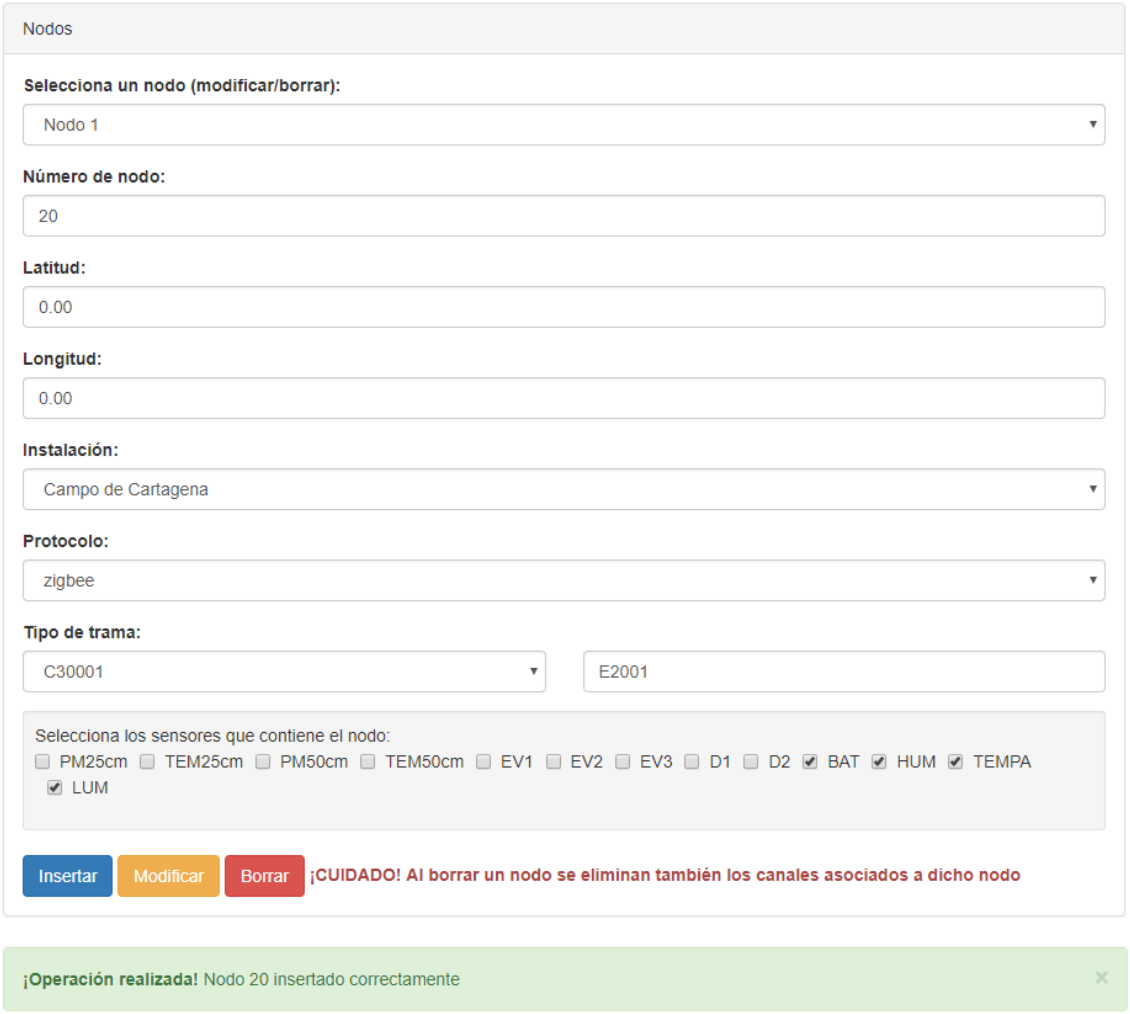


*Figura 4.4 Inserción de nodo.*

Una vez configurada la nueva instalación, se pueden realizar algunas pruebas mediante el envío de tramas desde la placa Arduino UNO. En concreto, se enviarán las tramas mostradas en la siguiente tabla:

| | |
|---|---|
| 1 | :0114E20013.99+0.9825+24.6+256341711186011217 |
| 2 | :0115E20013.99+0.9825+24.63512186011217 |
| 3 | :0116E20013.99+0.9825+256345613186011217 |
| 4 | :0114E20013.99+0.9825+24.6+256341728186011217 |
| 5 | :0115E20013.99+0.9825+24.63529186011217 |

| | |
|---|---|
| 6 | :0116E20013.99+0.9825+256345630186011217 |
| 7 | :0115E20013.99+0.9825+24.63541186011217 |
| 8 | :0116E20013.99+0.9825+256345643186011217 |

*Tabla 4-1 Tramas para la simulación.*

Tras ejecutar el Sketch sobre la placa Arduino UNO, se habrán insertado nuevos registros en la tabla *dato_canal* de la base de datos. En concreto, se pueden comprobar los datos introducidos en la *Figura 4.5*, que muestra cómo se han dividido las tramas e insertado correctamente los datos:

| id_data ▼ 1 | dia_semana | lectura_nodo | valor | _id_chan |
|---|---|---|---|---|
| 5274 | 6 | 2017-12-01 18:45:00 | 25634 | 2213 |
| 5273 | 6 | 2017-12-01 18:45:00 | 98.25 | 2211 |
| 5272 | 6 | 2017-12-01 18:45:00 | 3.99 | 2210 |
| 5271 | 6 | 2017-12-01 18:45:00 | 24.6 | 2112 |
| 5270 | 6 | 2017-12-01 18:45:00 | 98.25 | 2111 |
| 5269 | 6 | 2017-12-01 18:45:00 | 3.99 | 2110 |
| 5268 | 6 | 2017-12-01 18:30:00 | 25634 | 2213 |
| 5267 | 6 | 2017-12-01 18:30:00 | 98.25 | 2211 |
| 5266 | 6 | 2017-12-01 18:30:00 | 3.99 | 2210 |
| 5265 | 6 | 2017-12-01 18:30:00 | 24.6 | 2112 |
| 5264 | 6 | 2017-12-01 18:30:00 | 98.25 | 2111 |
| 5263 | 6 | 2017-12-01 18:30:00 | 3.99 | 2110 |
| 5262 | 6 | 2017-12-01 18:30:00 | 25634 | 2013 |
| 5261 | 6 | 2017-12-01 18:30:00 | 24.6 | 2012 |
| 5260 | 6 | 2017-12-01 18:30:00 | 98.25 | 2011 |
| 5259 | 6 | 2017-12-01 18:30:00 | 3.99 | 2010 |
| 5258 | 6 | 2017-12-01 18:15:00 | 25634 | 2213 |
| 5257 | 6 | 2017-12-01 18:15:00 | 98.25 | 2211 |
| 5256 | 6 | 2017-12-01 18:15:00 | 3.99 | 2210 |
| 5255 | 6 | 2017-12-01 18:15:00 | 24.6 | 2112 |
| 5254 | 6 | 2017-12-01 18:15:00 | 98.25 | 2111 |
| 5253 | 6 | 2017-12-01 18:15:00 | 3.99 | 2110 |
| 5252 | 6 | 2017-12-01 18:15:00 | 25634 | 2013 |
| 5251 | 6 | 2017-12-01 18:15:00 | 24.6 | 2012 |
| 5250 | 6 | 2017-12-01 18:15:00 | 98.25 | 2011 |
| 5249 | 6 | 2017-12-01 18:15:00 | 3.99 | 2010 |

*Figura 4.5 Datos almacenados en la tabla dato_canal.*

Para finalizar, ahora que los datos están registrados en la base de datos, se pueden extraer a través de la interfaz de usuario “Comunicación” en un fichero CSV. Para ello, se rellenan los campos como se muestra en la *Figura 4.6*:

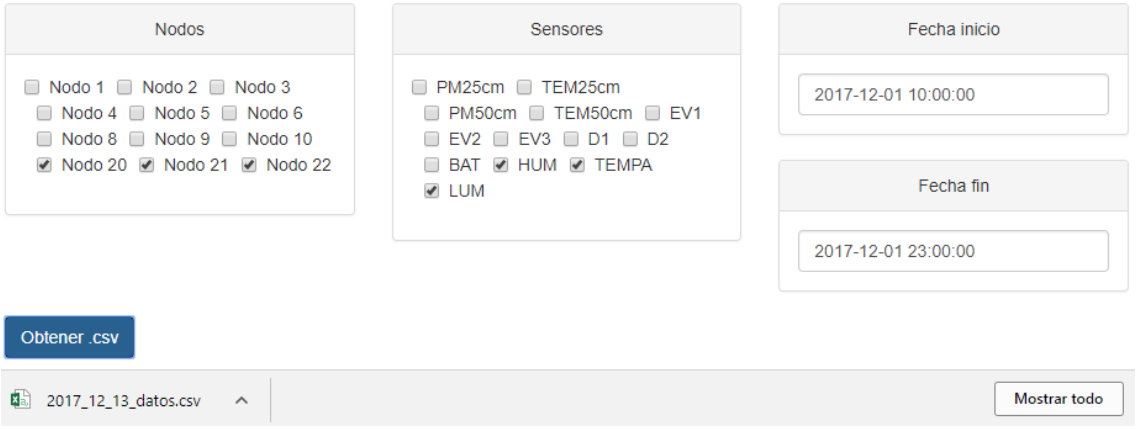


*Figura 4.6 Obtención de fichero CSV.*

Se puede abrir el fichero con un editor de texto para ver que el formato es el correcto o abrirlo con Excel en forma de tabla como se muestra en las *Figuras 4.7* y *4.8*:

```
Dia;Fecha;Nodo 20 HUM (v/v);Nodo 21 HUM (v/v);Nodo 22 HUM (v/v);Nodo 20 TEMPA (grad. C);Nodo 21 TEM
6;2017-12-01 18:15:00;98.25;98.25;98.25;24.6;24.6;;25634;25634
6;2017-12-01 18:30:00;98.25;98.25;98.25;24.6;24.6;;25634;25634
6;2017-12-01 18:45:00;0;98.25;98.25;;24.6;;;25634
```

*Figura 4.7 fichero CSV – 1.*

| Dia | Fecha | Nodo 20 HUM (v/v) | Nodo 21 HUM (v/v) | Nodo 22 HUM (v/v) | Nodo 20 TEMPA (grad. C) | Nodo 21 TEMPA (grad. C) | () | Nodo 20 LUM (lumis) | Nodo 22 LUM (lumis) |
|---|---|---|---|---|---|---|---|---|---|
| 6 | 01/12/2017 18:15 | 98.25 | 98.25 | 98.25 | 24.6 | 24.6 | | 25634 | 25634 |
| 6 | 01/12/2017 18:30 | 98.25 | 98.25 | 98.25 | 24.6 | 24.6 | | 25634 | 25634 |
| 6 | 01/12/2017 18:45 | 0 | 98.25 | 98.25 | | 24.6 | | | 25634 |

*Figura 4.8 Fichero CSV – 2.*

## 4.2 Conclusiones.

La parte más compleja de este proyecto quizás sea la cantidad y variedad de diferentes módulos software que conlleva. Partiendo del propio sistema operativo, ya que Raspbian Lite es un sistema operativo sin escritorio, se trabaja diréctamente con la consola, mediante comandos de Linux. Las configuraciones que se han presentado del dispositivo, aparentemente sencillas, llevan detrás también mucha investigación y pruebas de ensayo-error.

Desde el planteamiento inicial de la pasarela de datos, se ha ido desarrollando poco a poco el resto del proyecto.

Se ha desarrollado una base de datos robusta y bien estructurada en la que almacenar toda la información y moldeable a futuras modificaciones de la red. Seguido de ello, también se ha creado una interfaz muy visual y cómoda para poder acceder a los datos y configurar la red de forma sencilla.

Mediante esta plataforma de bajo coste económico se puede gestionar toda una red de nodos desde cualquier dispositivo conectado a su red local. Podría ser posible incluso acceder conectándose, remotamente, a un dispositivo dentro de la red y desde él a la Raspberry. Invirtiendo una pequeña cantidad de dinero también se podría comprar un dominio y un servidor para acceder directamente desde internet. En definitiva, el proyecto tiene rango de mejora si se quisiera, en ese aspecto.

Con lo que se puede concluir que se han cumplido correctamente, incluso ampliado, los objetivos del proyecto.

## 4.3 Futuras implementaciones.

El proyecto tiene bastante rango de mejora invirtiendo más tiempo y recursos económicos. Algunas de esas mejoras podrían ser:

- Comprar un dominio y contratar un servidor web para poder acceder a la interfaz HTML desde cualquier dispositivo con conexión a Internet. Para ello, sería recomendable también añadir cierto nivel de seguridad a través de una identificación con usuario y contraseña, por ejemplo.
- Limitar la modificación de la red, para que sólo determinado/s usuario/s puedan realizarla, o por lo menos las acciones más críticas para el sistema, como eliminar elementos de la red, ya que al hacerlo se eliminan todos los elementos asociados mediante claves foráneas.
- Implementar contadores en las tablas de la base de datos que indiquen los elementos que estén relacionados con cada elemento de la tabla, por ejemplo, para un nodo, podría ser útil saber la

cantidad de sensores que contiene o la cantidad de datos que ha registrado ese nodo.

- Implementar más condiciones a la hora de obtener los datos, como poder elegir un mes, unos días o un intervalo de horas en concreto. Incluso, que procese los datos con algún tipo de operación, como una media artimética o desviación típica, localizar datos atípicos o fuera de un rango de seguridad previamente definido.
- Podría añadirse una función que cada cierto tiempo (semana, mes, entre otros) almacene los datos en algún sitio, como un servidor FTP, y los elimine de la memoria interna del dispositivo, para liberar espacio.
- Sería buena idea también realizar algún tipo de “backup” de la base de datos cuando se eliminen elementos de la misma, por si se quisiera volver atrás y recuperarlos.

# Capítulo 5

# Bibliografía

## 5.1 Bibliografía citada.

[1] Raspberry Pi 2 B: *Especificaciones*. Sitios web:
https://www.raspberrypi.org/products/raspberry-pi-2-model-b/#buy-now-modal
https://es.rs-online.com/web/p/processor-microcontroller-development-kits/1259525/?src=raspberrypi

[2] SolidRay 2: *Motaje de Raspberry Pi*. Sitio web:
https://kitprinter3d.com/es/blog/-manual-de-montaje-solidray-2-el-software-conexion-wifi-n56

[3] BeagleBoard-X15: *Especificaciones*. Sitios web:
https://beagleboard.org/x15
https://www.digikey.com/product-detail/en/precision-technology-inc/BEAGLEBOARD-X15/1777-1001-ND

[4] ODROID-C2: *Especificaciones*. Sitio web:
http://www.hardkernel.com/main/products/prdt_info.php?g_code=G145457216438

[5] Sistemas Operativos para Raspberry Pi. Sitio web:
https://www.raspberrypi.org/downloads/

[6] Raspbian: *Características*. Sitio web:
https://www.raspberrypi.org/downloads/raspbian/

[7] (Libro UPCT) Sánchez, P. *Programación avanzada ingeniería del software y bases de datos*.

[8] Bases de datos: *Información*. Sitio web: https://es.wikipedia.org/wiki/Base_de_datos

[9] Bases de datos orientadas a objetos: *Información*. Sitio web: https://es.wikipedia.org/wiki/Base_de_datos_orientada_a_objetos

[10] Bases de datos relacionales: *Información*. Sitio web: https://es.wikipedia.org/wiki/Base_de_datos_relacional

[11] (PFC-UPCT) Jerez, A. *Implementación de un cliente REST para un servicio de almacenamiento de ficheros basado en metadatos* (2015): Capítulos 1 y 2. Obtenido de: http://repositorio.upct.es/bitstream/handle/10317/4561/pfc6085.pdf?sequence=1&isAllowed=y

[12] Servicios web: *Información*. Sitio web: https://es.wikipedia.org/wiki/Servicio_web

[13] SOAP: *Información*. Sitio web: https://es.wikipedia.org/wiki/Simple_Object_Access_Protocol

[14] REST: *Información*. Sitio web: https://es.wikipedia.org/wiki/Transferencia_de_Estado_Representacional

[15] Roy Thomas Fielding: *Biografía*. Sitio web: https://es.wikipedia.org/wiki/Roy_Fielding

[16] Bootstrap: *Información*. Sitio web: https://es.wikipedia.org/wiki/Bootstrap_(framework)

[17] HTML5 Boilerplate: *Información y comparación*. Sitio web: https://www.sitepoint.com/boilerplate-bootstrap/

Código de obtención del archivo CSV: https://gist.github.com/lmfresneda/64101b73efe10ef8b6fd

## 5.2 Otra Bibliografía utilizada.

*Geeky Theory*. Sitio web: https://geekytheory.com/

*Arduino*. Sitio web: https://www.arduino.cc/

(Libro) Bahit, E. *Python para principiantes*. Obtenido de: http://librosweb.es/libro/python/

(Libro) Wachenchauzer, R; Manterola, M.; Curia, M.; Medrano, M.; Paez, N. *Algorítmos de Programación con Python*. Obtenido de: http://librosweb.es/libro/algoritmos_python/

*Tutorial Python*. Sitio web: https://www.tutorialpython.com/

*MySQL*. Sitio web: https://dev.mysql.com/doc/

*Slim*. Sitio web: https://www.slimframework.com/

*w3schools.com*. Sitio web: https://www.w3schools.com/

# ANEXO I – CÓDIGO DE LA INTERFAZ HTML5.

Para el diseño de la interfaz html5 se han utilizado los estilos responsivos de Bootstrap, el código es el siguiente:

- Página de “Home” (*home.html*) o interfaz para comprobar los datos:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>WSN</title>
  <link rel="icon" type="image/png" href="./favicon.png">
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <script type="application/javascript" src="obtenciondatos.js"></script>
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
  <style>
    /* Remove the navbar's default rounded borders and increase the bottom margin */
    .navbar {
      margin-bottom: 50px;
      border-radius: 0;
    }

    /* Remove the jumbotron's default bottom margin */
    .jumbotron {
      margin-bottom: 0;
    }

    /* Add a gray background color and some padding to the footer */
    footer {
      background-color: #f2f2f2;
      padding: 20px;
    }
  </style>
</head>

<body onload="inicializar()"> <!-- Para cargar los datos -->

<!-- Encabezado -->
<div class="jumbotron">
  <div class="container text-center">
    <h1>Comunicación.</h1>
    <p>Interfaz de comunicación con la red</p>
  </div>
</div>

<nav class="navbar navbar-inverse">
  <div class="container-fluid">
    <div class="navbar-header">
      <a class="navbar-brand" href="#">WSN</a>
    </div>
    <ul class="nav navbar-nav">
      <li class="active"><a href="#">Home</a></li>
      <li class="dropdown"><a class="dropdown-toggle" data-toggle="dropdown" href="#">Configuración<span
class="caret"></span></a>
```

```
      <ul class="dropdown-menu">
        <li><a href="./configuracion.html#instalacion">Instalaciones</a></li>
        <li><a href="./configuracion.html#nodo">Nodos</a></li>
        <li><a href="./configuracion.html#sensor">Sensores</a></li>
              <li><a href="./configuracion.html#unidad">Unidades</a></li>
      </ul>
     </li>
    </ul>
  </div>
</nav>

<div class="container-fluid">
  <div class="row">
    <div class="col-sm-4">
      <div class="panel panel-default">
        <div class="panel-heading text-center">Nodos</div>
              <div class="panel-body">
         <div class="form-group" id="catalogonodos">
              </div>
             </div>
      </div>
    </div>
    <div class="col-sm-4">
      <div class="panel panel-default">
        <div class="panel-heading text-center">Sensores</div>
             <div class="panel-body">
          <div class="form-group" id="catalogosensores">
              </div>
             </div>
      </div>
    </div>
    <div class="col-sm-4">
      <div class="panel panel-default">
        <div class="panel-heading text-center">Fecha inicio</div>
             <div class="panel-body">
              <input class="form-control" id="fecha_ini" type="text" placeholder="20AA-MM-DD HH:MM:SS" maxlength="19">
             </div>
      </div>
      <div class="panel panel-default">
        <div class="panel-heading text-center">Fecha fin</div>
        <div class="panel-body">
              <input class="form-control" id="fecha_fin" type="text" placeholder="20AA-MM-DD HH:MM:SS" maxlength="19">
             </div>
      </div>
    </div>
  </div>
  <div class="form-group" id="catalogosensores">
    <button type="button" class="btn btn-primary" id = "btnCsv" onclick = "csv()">Obtener .csv</button>
  </div>
</div>

<footer class="container-fluid text-center">2017 &copy; Miguel Martínez Royo</footer>

</body>
</html>
```

- Archivo *obtenciondatos.js* asociado a la página “Home”:

```
// Poner la ip del dispositivo
var ip="192.168.1.104";

function inicializar(){
        obtenersensores();
        obtenernodos();
}

function obtenernodos(){
        var xmlhttp = new XMLHttpRequest();
```

```
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				$("#catalogonodos").empty();
				if (jsonresponse.message != null){
					cargarNodos(jsonresponse.message);
				}
			}
		}
	}
	xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerlistanodos", true);
	xmlhttp.send();
}

function obtenersensores(){
	var xmlhttp = new XMLHttpRequest();
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				$("#catalogosensores").empty();
				if (jsonresponse.message != null){
					cargarSensores(jsonresponse.message);
				}
			}
		}
	}
	xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerlistasensores", true);
	xmlhttp.send();
}

function cargarNodos(messg){
	for (var i = 0; i < messg.length; i++){
		checksNodos(messg[i].id, messg[i].title);
	}
}

function cargarSensores(messg){
	for (var i = 0; i < messg.length; i++){
		checksSensores(messg[i].id, messg[i].title);
	}
}

function checksNodos(id, tipo){
	var checkin = '<label class="checkbox-inline"><input type="checkbox" value="nodo" id="' + id + '">' + tipo + '</label>';
	$("#catalogonodos").append(checkin);
}

function checksSensores(id, tipo){
	var checkin = '<label class="checkbox-inline"><input type="checkbox" value="sensor" id="' + id + '">' + tipo + '</label>';
	$("#catalogosensores").append(checkin);
}

function csv() {
	xmlhttp = new XMLHttpRequest();
	var url = "http://" + ip + "/wsn/v1/obtenercsv";
	xmlhttp.open("POST", url, true);
	xmlhttp.setRequestHeader("Content-type", "application/json");
	xmlhttp.onreadystatechange = function () {
		if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
			var jsonresponse = JSON.parse(xmlhttp.responseText);
			if (jsonresponse.message != null){
				if (jsonresponse.error == false) {
					arrayObjToCsv(jsonresponse.message);
				} else {
					alert("Error: " + jsonresponse.error + "\n" + jsonresponse.message);
				}
			}
		}
	}
```

```
	var fecha_ini = document.getElementById("fecha_ini");
	var fecha_fin = document.getElementById("fecha_fin");
	var array_nodos = "";
	var array_sensores = "";
	var flag_nod = 0;
	var flag_sen = 0;
	var grupo = document.getElementsByTagName('input');
	for (i=0;i<grupo.length;i++){
		if ((grupo[i].type == "checkbox")&&(grupo[i].checked == 1)){
			if (grupo[i].value == "nodo") {
				if(flag_nod == 1){
					array_nodos += ",";
				}
				array_nodos += grupo[i].id;
				flag_nod = 1;
			}
			if (grupo[i].value == "sensor") {
				if(flag_sen == 1){
					array_sensores += ",";
				}
				array_sensores += grupo[i].id;
				flag_sen = 1;
			}
		}
	}
	var data = JSON.stringify({
		fecha_ini: fecha_ini.value,
		fecha_fin: fecha_fin.value,
		array_nodos: array_nodos,
		array_sensores: array_sensores
	});
	xmlhttp.send(data);
}

function arrayObjToCsv(ar) {
	//comprobamos compatibilidad
	if(window.Blob && (window.URL || window.webkitURL)){
		var contenido = "",
			d = new Date(),
			blob,
			reader,
			save,
			clicEvent;
		//creamos contenido del archivo
		for (var i = 0; i < ar.length; i++) {
			//construimos cabecera del csv
			if (i == 0)
				contenido += Object.keys(ar[i]).join(";") + "\n";
			//resto del contenido
			contenido += Object.keys(ar[i]).map(function(key) {
				return ar[i][key];
			}).join(";") + "\n";
		}
		//creamos el blob
		blob =  new Blob(["\ufeff", contenido], {type: 'text/csv'});
		//creamos el reader
		var reader = new FileReader();
		reader.onload = function (event) {
			//escuchamos su evento load y creamos un enlace en dom
			save = document.createElement('a');
			save.href = event.target.result;
			save.target = '_blank';
			//aquí le damos nombre al archivo
			save.download = d.getFullYear() + "_" + (d.getMonth()+1) + "_" + d.getDate() + "_datos.csv";
			try {
				//creamos un evento click
				clicEvent = new MouseEvent('click', {
					'view': window,
					'bubbles': true,
					'cancelable': true
```

```
                                        });
                                } catch (e) {
                                        //si llega aquí es que probablemente implemente la forma antigua de crear un enlace
                                        clicEvent = document.createEvent("MouseEvent");
                                        clicEvent.initEvent('click', true, true);
                                }
                                //disparamos el evento
                                save.dispatchEvent(clicEvent);
                                //liberamos el objeto window.URL
                                (window.URL || window.webkitURL).revokeObjectURL(save.href);
                        }
                        //leemos como url
                        reader.readAsDataURL(blob);
                } else {
                        //el navegador no admite esta opción
                        alert("Su navegador no permite esta acción");
                }
}
```

El código correspondiente a la obtención del archivo CSV, la función *arrayObjToCsv(ar)*, se obtuvo del siguiente sitio web:

https://gist.github.com/lmfresneda/64101b73efe10ef8b6fd

- Página de "Configuración" (*configuracion.html*) o interfaz para la modificación de la base de datos:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>WSN</title>
  <link rel="icon" type="image/png" href="./favicon.png">
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <script type="application/javascript" src="funciones.js"></script>
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
  <style>
    /* Remove the navbar's default rounded borders and increase the bottom margin */
    .navbar {
      margin-bottom: 50px;
      border-radius: 0;
    }

    /* Remove the jumbotron's default bottom margin */
     .jumbotron {
      margin-bottom: 0;
    }

    /* Add a gray background color and some padding to the footer */
    footer {
      background-color: #f2f2f2;
      padding: 20px;
    }
  </style>
</head>

<body onload="inicializar()"> <!-- Para cargar las listas -->

<!-- Encabezado -->
<div class="jumbotron">
  <div class="container text-center">
```

```
   <h1>Configuración.</h1>
   <p>Interfaz para editar la configuración de la red</p>
  </div>
</div>

<nav class="navbar navbar-inverse">
  <div class="container-fluid">
   <div class="navbar-header">
    <a class="navbar-brand" href="#">WSN</a>
   </div>
   <ul class="nav navbar-nav">
    <li><a href="./home.html">Home</a></li>
    <li class="dropdown active"><a class="dropdown-toggle" data-toggle="dropdown" href="#">Configuración<span
class="caret"></span></a>
      <ul class="dropdown-menu">
       <li><a href="#instalacion">Instalaciones</a></li>
       <li><a href="#nodo">Nodos</a></li>
       <li><a href="#sensor">Sensores</a></li>
            <li><a href="#unidad">Unidades</a></li>
      </ul>
    </li>
   </ul>
  </div>
</nav>

<div class="container-fluid">
  <!-- Panel para manejar instalaciones -->
  <div class="panel panel-default" id="instalacion">
   <div class="panel-heading">Instalaciones</div>
   <div class="panel-body">
    <!-- Formulario Instalaciones -->
    <form id="idforminstalacion">
          <!-- Lista instalaciones -->
          <div class="form-group">
           <label for="listaInstalacion">Selecciona una instalación (modificar/borrar):</label>
           <select class="form-control" id="listaInstalacion" onchange="seleccionainstalacion()"></select>
          </div>
          <!-- Título -->
          <div class="form-group">
           <label for="idinstalaciontitle">Nombre:</label>
           <textarea class="form-control" rows="2" id="idinstalaciontitle" maxlength="300"></textarea>
          </div>
      <!-- Latitud -->
          <div class="form-group">
           <label for="idinstalacionlatitud">Latitud:</label>
           <input type="text" class="form-control" id="idinstalacionlatitud" maxlength="20">
          </div>
          <!-- Longitud -->
          <div class="form-group">
           <label for="idinstalacionlongitud">Longitud:</label>
           <input type="text" class="form-control" id="idinstalacionlongitud" maxlength="20">
          </div>
          <!-- Botones para editar -->
          <button type="button" class="btn btn-primary" id = "btninsertarinstalacion" onclick =
"insertarinstalacion()">Insertar</button>
          <button type="button" class="btn btn-warning" id = "btnmodificarinstalacion" onclick =
"modificarinstalacion()">Modificar</button>
          <button type="button" class="btn btn-danger" id = "btnborrarinstalacion" onclick =
"borrarinslacion()">Borrar</button>
          <label class="text-danger">¡CUIDADO! Al borrar una instalación se eliminan también los nodos pertenecientes a dicha
instalación y los canales asociados a esos nodos</label>
    </form>
   </div>
  </div>

  <!-- Panel para manejar nodos -->
  <div class="panel panel-default" id="nodo">
   <div class="panel-heading">Nodos</div>
   <div class="panel-body">
    <!-- Formulario Nodos -->
    <form id="idformnodo">
```

```
    <!-- Lista nodos -->
          <div class="form-group">
            <label for="listaNodo">Selecciona un nodo (modificar/borrar):</label>
            <select class="form-control" id="listaNodo" onchange="seleccionanodo()"></select>
          </div>
          <!-- Descripción -->
          <div class="form-group">
            <label for="idnododes">Número de nodo:</label>
            <input type="text" class="form-control" id="idnododes" maxlength="3">
          </div>
          <!-- Latitud -->
    <div class="form-group">
            <label for="idnodolatitud">Latitud:</label>
            <input type="text" class="form-control" id="idnodolatitud" maxlength="20">
          </div>
          <!-- Longitud -->
          <div class="form-group">
            <label for="idnodolongitud">Longitud:</label>
            <input type="text" class="form-control" id="idnodolongitud" maxlength="20">
          </div>
          <!-- Instalación a la que pertenece -->
          <div class="form-group">
            <label for="listaInstalacionNodo">Instalación:</label>
            <select class="form-control" id="listaInstalacionNodo"></select>
          </div>
          <!-- Protocolo -->
          <div class="form-group">
            <label for="listaProtocolo">Protocolo:</label>
            <select class="form-control" id="listaProtocolo"></select>
          </div>
          <!-- Iipo de trama -->
          <div class="form-group">
            <label for="listaTrama">Tipo de trama:</label>
            <div class="row">
              <div class="col-xs-6">
                <select class="form-control" id="listaTrama"></select>
              </div>
              <div class="col-xs-6">
                <input class="form-control" id="tipotrama" type="text" placeholder="Escriba un tipo nuevo" maxlength="20">
              </div>
            </div>
          </div>
          <!-- Selección de sensores -->
          <div class="well well-sm"> Selecciona los sensores que contiene el nodo:
            <div class="form-group" id="catalogosensores">
            </div>
          </div>
    <div>
      <button type="button" class="btn btn-primary" id = "btninsertarnodo" onclick = "insertarnodo()">Insertar</button>
            <button type="button" class="btn btn-warning" id = "btnmodificarnodo" onclick =
"modificarnodo()">Modificar</button>
            <button type="button" class="btn btn-danger" id = "btnborrarnodo" onclick = "borrarnodo()">Borrar</button>
            <label class="text-danger">¡CUIDADO! Al borrar un nodo se eliminan también los canales asociados a dicho
nodo</label>
      </div>
    </form>
  </div>
 </div>

 <!-- Panel para manejar sensores -->
 <div class="panel panel-default" id="sensor">
  <div class="panel-heading">Sensores</div>
  <div class="panel-body">
   <!-- Formulario Sensores -->
   <form id="idformsensor">
          <!-- Lista sensores -->
          <div class="form-group">
            <label for="listaSensor">Selecciona un sensor (modificar/borrar):</label>
            <select class="form-control" id="listaSensor" onchange="seleccionasensor()"></select>
          </div>
          <!-- Tipo -->
```

```
            <div class="form-group">
              <label for="idsensortipo">Abreviatura:</label>
              <input type="text" class="form-control" id="idsensortipo" maxlength="50">
            </div>
            <!-- Descripción -->
            <div class="form-group">
              <label for="idsensordes">Descripción:</label>
              <textarea class="form-control" rows="2" id="idsensordes" maxlength="300"></textarea>
            </div>
            <!-- Formato -->
            <div class="form-group">
              <label for="idsensorformato">Formato (ej: +00.000):</label>
              <input type="text" class="form-control" id="idsensorformato" maxlength="20">
            </div>
            <!-- Unidades -->
            <div class="form-group">
              <label>Unidades:</label>
              <div class="row">
                <div class="col-xs-6">
                  <select class="form-control" id="listaUnidad"></select>
                </div>
                <div class="col-xs-6">
                  <input class="form-control" id="desunidad" type="text" placeholder="Escriba una unidad nueva" maxlength="30">
                </div>
              </div>
            </div>
            <!-- Parámetros de calibración -->
            <div class="form-group">
              <label>Parametros de calibración (ax^2 + bx + c) (Si se desconocen: a = c = 0, b = 1):</label>
              <div class="row">
                <div class="col-xs-4">
                  <label for="acalib">a</label>
                  <input class="form-control" id="acalib" type="text">
                </div>
                <div class="col-xs-4">
                  <label for="bcalib">b</label>
                  <input class="form-control" id="bcalib" type="text">
                </div>
                <div class="col-xs-4">
                  <label for="ccalib">c</label>
                  <input class="form-control" id="ccalib" type="text">
                </div>
              </div>
            </div>
            <!-- Botones para editar -->
      <div>
            <button type="button" class="btn btn-primary" id = "btninsertarsensor" onclick = "insertarsensor()">Insertar</button>
        <button type="button" class="btn btn-warning" id = "btnmodificarsensor" onclick = "modificarsensor()">Modificar</button>
            <button type="button" class="btn btn-danger" id = "btnborrarsensor" onclick = "borrarsensor()">Borrar</button>
            <label class="text-danger">¡CUIDADO! Al borrar un sensor se eliminan también los canales asociados a dicho sensor</label>
      </div>
    </form>
   </div>
  </div>

  <!-- Panel para manejar unidades -->
  <div class="panel panel-default" id="unidad">
    <div class="panel-heading">Unidades</div>
    <div class="panel-body">
      <!-- Formulario Unidades -->
      <form id="idformunidad">
        <!-- Unidades -->
        <div class="form-group">
          <label for="listaUnidadMod">Selecciona una unidad:</label>
          <div class="row">
            <div class="col-xs-6">
                  <select class="form-control" id="listaUnidadMod"></select>
            </div>
```

```
            <div class="col-xs-6">
              <input class="form-control" id="iddesunidad" type="text" placeholder="Escriba aquí la modificación" maxlength="30">
        </div>
            </div>
    </div>
    <!-- Botones para editar -->
    <div>
            <button type="button" class="btn btn-warning" id = "btnmodificarunidad" onclick = "modificarunidad()">Modificar</button>
       <button type="button" class="btn btn-danger" id = "btnborrarunidad" onclick = "borrarunidad()">Borrar</button>
            <label class="text-danger">¡CUIDADO! Al borrar una unidad se eliminan también los sensores que utilicen dicha unidad y los canales asociados a esos sensores</label>
    </div>
   </form>
  </div>
 </div>
</div>

<footer class="container-fluid text-center">2017 &copy; Miguel Martínez Royo</footer>

</body>
</html>
```

- Archivo *funciones.js* asociado a la página “Configuración”:

```
// Poner la ip del dispositivo
var ip="192.168.1.104";

function inicializar(){
        obtenerinstalaciones();
        obtenerprotocolos();
        obtenertramas();
        obtenerunidades();
        obtenersensores();
        obtenernodos();
}

// **********************
// *** INSTALACIONES ***
// **********************

function obtenerinstalaciones(){
        var xmlhttp = new XMLHttpRequest();
        xmlhttp.onreadystatechange = function(){
                if ((this.readyState == 4)&&(this.status == 200)){
                        if (this.responseText != null){
                                jsonresponse = JSON.parse(this.responseText);
                                $("#listaInstalacion").empty();
                                $("#listaInstalacionNodo").empty();
                                if (jsonresponse.message != null){
                                        var miselect = document.getElementById("listaInstalacion");
                                        cargarlistas(jsonresponse.message, miselect);
                                        miselect = document.getElementById("listaInstalacionNodo");
                                        cargarlistas(jsonresponse.message, miselect);
                                        seleccionainstalacion();
                                        seleccionanodo();
                                }
                        }
                }
        }
        xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerlistainstaciones", true);
        xmlhttp.send();
}

function cargarlistas(messg, miselect){
        for (var i = 0; i < messg.length; i++){
                var opcion = document.createElement('option');
```

```
                        opcion.innerHTML = messg[i].title;
                        opcion.value = messg[i].id;
                        miselect.append(opcion);
                }
}

function seleccionainstalacion(){
        var seleccion = document.getElementById("listaInstalacion");
        var xmlhttp = new XMLHttpRequest();
        xmlhttp.onreadystatechange = function(){
                if ((this.readyState == 4)&&(this.status == 200)){
                        if (this.responseText != null){
                                jsonresponse = JSON.parse(this.responseText);
                                if (jsonresponse.message != null){
                                        document.getElementById("idinstalaciontitle").value =
jsonresponse.message.nombre;
                                        document.getElementById("idinstalacionlatitud").value =
jsonresponse.message.latitud;
                                        document.getElementById("idinstalacionlongitud").value =
jsonresponse.message.longitud;
                                }
                        }
                }
        }
        xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerdatosinstalaciones/" + seleccion.value, true);
        xmlhttp.send();
}

// Inserta una instalación en la base de datos
function insertarinstalacion(){
        xmlhttp = new XMLHttpRequest();
        var url = "http://" + ip + "/wsn/v1/insertarinstalacion";
        xmlhttp.open("POST", url, true);
        xmlhttp.setRequestHeader("Content-type", "application/json");
        xmlhttp.onreadystatechange = function () {
                if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                        var jsonresponse = JSON.parse(xmlhttp.responseText);
                        if (jsonresponse.message != null){
                                alerts(jsonresponse.error,jsonresponse.message,"#instalacion");
                                obtenerinstalaciones();
                        }
                }
        }

        var nombre = document.getElementById("idinstalaciontitle");
        var latitud = document.getElementById("idinstalacionlatitud");
        var longitud = document.getElementById("idinstalacionlongitud");
        var data = JSON.stringify({
                nombre: nombre.value,
                latitud: latitud.value,
                longitud: longitud.value
        });
        xmlhttp.send(data);
}

// Modifica una instalación en la base de datos
function modificarinstalacion() {
        var seleccion = document.getElementById("listaInstalacion");
        xmlhttp = new XMLHttpRequest();
        var url = "http://" + ip + "/wsn/v1/modificarinstalacion/" + seleccion.value;
        xmlhttp.open("POST", url, true);
        xmlhttp.setRequestHeader("Content-type", "application/json");
        xmlhttp.onreadystatechange = function () {
                if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                        var jsonresponse = JSON.parse(xmlhttp.responseText);
                        if (jsonresponse.message != null){
                                alerts(jsonresponse.error,jsonresponse.message,"#instalacion");
                                obtenerinstalaciones();
                        }
                }
        }
```

```
	var nombre = document.getElementById("idinstalaciontitle");
	var latitud = document.getElementById("idinstalacionlatitud");
	var longitud = document.getElementById("idinstalacionlongitud");
	var data = JSON.stringify({
		nombre: nombre.value,
		latitud: latitud.value,
		longitud: longitud.value
	});
	xmlhttp.send(data);
}

// Borra una instalación en la base de datos
function borrarinslacion() {
	var seleccion = document.getElementById("listaInstalacion");
	var xmlhttp = new XMLHttpRequest();
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				if (jsonresponse.message != null){
				 alerts(jsonresponse.error,jsonresponse.message,"#instalacion");
				 obtenerinstalaciones();
				 obtenernodos();
				}
			}
		}
	}
	xmlhttp.open("GET", "http://" + ip + "/wsn/v1/borrarinstalacion/" + seleccion.value, true);
	xmlhttp.send();
}

// **********************
// ***    NODOS    ***
// **********************

function obtenernodos(){
	var xmlhttp = new XMLHttpRequest();
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				$("#listaNodo").empty();
				if (jsonresponse.message != null){
					var miselect = document.getElementById("listaNodo");
					cargarlistas(jsonresponse.message, miselect);
					seleccionanodo();
				}
			}
		}
	}
	xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerlistanodos", true);
	xmlhttp.send();
}

function obtenerprotocolos(){
	var xmlhttp = new XMLHttpRequest();
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				$("#listaProtocolo").empty();
				if (jsonresponse.message != null){
					var miselect = document.getElementById("listaProtocolo");
					cargarlistas(jsonresponse.message, miselect);
				}
			}
		}
	}
	xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerlistaprotocolos", true);
	xmlhttp.send();
```

```
}

function obtenertramas(){
	var xmlhttp = new XMLHttpRequest();
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				$("#listaTrama").empty();
				if (jsonresponse.message != null){
					var miselect = document.getElementById("listaTrama");
					cargarlistas(jsonresponse.message, miselect);
				}
			}
		}
	}
	xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerlistatramas", true);
	xmlhttp.send();
}

function seleccionanodo(){
	var seleccion = document.getElementById("listaNodo");
	var xmlhttp = new XMLHttpRequest();
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				if (jsonresponse.message != null){
					document.getElementById("idnododes").value = jsonresponse.message.descripcion;
					document.getElementById("idnodolatitud").value = jsonresponse.message.latitud;
					document.getElementById("idnodolongitud").value = jsonresponse.message.longitud;
					document.getElementById("listaInstalacionNodo").value = jsonresponse.message.instalacion;
					document.getElementById("listaProtocolo").value = jsonresponse.message.protocolo;
					document.getElementById("listaTrama").value = jsonresponse.message.trama;
					var arraysensores = jsonresponse.message.idsensor;
					var arr = arraysensores.split(",");
					var grupo = document.getElementsByTagName('input');
					for (i=0;i<grupo.length;i++){
						if (grupo[i].type == "checkbox") {
							grupo[i].checked = 0;
						}
					}
					arr = arraysensores.split(",");
					for (i = 0; i < arr.length; i++) {
						document.getElementById(arr[i]).checked = 1;
					}
					for (i = 0; i < arr.length; i++) {
						document.getElementById(arr[i]).checked = 1;
					}
				}
			}
		}
	}
	xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerdatosnodos/" + seleccion.value, true);
	xmlhttp.send();
}

// Inserta un nodo en la base de datos
function insertarnodo(){
	xmlhttp = new XMLHttpRequest();
	var url = "http://" + ip + "/wsn/v1/insertarnodo";
	xmlhttp.open("POST", url, true);
	xmlhttp.setRequestHeader("Content-type", "application/json");
	xmlhttp.onreadystatechange = function () {
		if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
```

```
                                var jsonresponse = JSON.parse(xmlhttp.responseText);
                                if (jsonresponse.message != null){
                                        alerts(jsonresponse.error,jsonresponse.message,"#nodo");
                                        obtenertramas();
                                        resetTrama();
                                        obtenernodos();
                                }
                        }
                }

                var descripcion = document.getElementById("idnododes");
                var latitud = document.getElementById("idnodolatitud");
                var longitud = document.getElementById("idnodolongitud");
                var instalacion = document.getElementById("listaInstalacionNodo");
                var protocolo = document.getElementById("listaProtocolo");
                var trama;
                if (document.getElementById("tipotrama").value == ""){
                        trama = document.getElementById("listaTrama");
                } else {
                        trama = document.getElementById("tipotrama");
                }
                var array_id = "";
                var array_long = "";
                var flag = 0;
                var env = 0;
                var longitud_total = 0;
                var grupo = document.getElementsByTagName('input');
                for (i=0;i<grupo.length;i++){
                        if ((grupo[i].type == "checkbox")&&(grupo[i].checked == 1)){
                                if(flag == 1){
                                        array_id += ",";
                                        array_long += ",";
                                }
                                if((grupo[i].id == "5")||(grupo[i].id == "6")||(grupo[i].id == "7")){
                                        env = 1;
                                }
                                array_id += grupo[i].id;
                                array_long += grupo[i].value;
                                longitud_total += parseInt(grupo[i].value);
                                flag = 1;
                        }
                }
                // 5 correspone al inicio de trama :01XX y 13 a la fecha
                longitud_total += trama.value.length + 5 + env + 13;
                var data = JSON.stringify({
                        descripcion: descripcion.value,
                        latitud: latitud.value,
                        longitud: longitud.value,
                        instalacion: instalacion.value,
                        protocolo: protocolo.value,
                        trama: trama.value,
                        array_id: array_id,
                        array_long: array_long,
                        longitud_total: longitud_total
                });
                xmlhttp.send(data);
}

// Modifica un nodo en la base de datos
function modificarnodo() {
                var seleccion = document.getElementById("listaNodo");
                xmlhttp = new XMLHttpRequest();
                var url = "http://" + ip + "/wsn/v1/modificarnodo/" + seleccion.value;
                xmlhttp.open("POST", url, true);
                xmlhttp.setRequestHeader("Content-type", "application/json");
                xmlhttp.onreadystatechange = function () {
                        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                                var jsonresponse = JSON.parse(xmlhttp.responseText);
                                if (jsonresponse.message != null){
                                        alerts(jsonresponse.error,jsonresponse.message,"#nodo");
                                        obtenertramas();
```

```
                                        resetTrama();
                                        obtenernodos();
                                }
                        }
                }

                var latitud = document.getElementById("idnodolatitud");
                var longitud = document.getElementById("idnodolongitud");
                var instalacion = document.getElementById("listaInstalacionNodo");
                var protocolo = document.getElementById("listaProtocolo");
                var trama;
                if (document.getElementById("tipotrama").value == ""){
                        trama = document.getElementById("listaTrama");
                } else {
                        trama = document.getElementById("tipotrama");
                }
                var array_id = "";
                var array_long = "";
                var flag = 0;
                var env = 0;
                var longitud_total = 0;
                var grupo = document.getElementsByTagName('input');
                for (i=0;i<grupo.length;i++){
                        if ((grupo[i].type == "checkbox")&&(grupo[i].checked == 1)){
                                if(flag == 1){
                                        array_id += ",";
                                        array_long += ",";
                                }
                                if((grupo[i].id == "5")||(grupo[i].id == "6")||(grupo[i].id == "7")){
                                        env = 1;
                                }
                                array_id += grupo[i].id;
                                array_long += grupo[i].value;
                                longitud_total += parseInt(grupo[i].value);
                                flag = 1;
                        }
                }
                // 5 correspone al inicio de trama :01XX y 13 a la fecha
                longitud_total += document.getElementById("listaTrama").value.length + 5 + env + 13;
                var data = JSON.stringify({
                        latitud: latitud.value,
                        longitud: longitud.value,
                        instalacion: instalacion.value,
                        protocolo: protocolo.value,
                        trama: trama.value,
                        array_id: array_id,
                        array_long: array_long,
                        longitud_total: longitud_total
                });
                xmlhttp.send(data);
}

// Borra una instalación en la base de datos
function borrarnodo() {
                var seleccion = document.getElementById("listaNodo");
                var xmlhttp = new XMLHttpRequest();
                xmlhttp.onreadystatechange = function(){
                        if ((this.readyState == 4)&&(this.status == 200)){
                                if (this.responseText != null){
                                        jsonresponse = JSON.parse(this.responseText);
                                        if (jsonresponse.message != null){
                                                alerts(jsonresponse.error,jsonresponse.message,"#nodo");
                                                obtenernodos();
                                        }
                                }
                        }
                }
                xmlhttp.open("GET", "http://" + ip + "/wsn/v1/borrarnodo/" + seleccion.value, true);
                xmlhttp.send();
}
```

```
// *********************
// ***  SENSORES  ****
// *********************

function cargarlistasSensor(messg, miselect){
	for (var i = 0; i < messg.length; i++){
		var opcion = document.createElement('option');
		opcion.innerHTML = messg[i].title;
		opcion.value = messg[i].id;
		miselect.append(opcion);
		checks(messg[i].id, messg[i].title, messg[i].longitud);
	}
}

function obtenersensores(){
	var xmlhttp = new XMLHttpRequest();
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				$("#listaSensor").empty();
				$("#catalogosensores").empty();
				if (jsonresponse.message != null){
					var miselect = document.getElementById("listaSensor");
					cargarlistasSensor(jsonresponse.message, miselect);
					seleccionasensor();
					seleccionanodo();
				}
			}
		}
	}
	xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerlistasensores", true);
	xmlhttp.send();
}

function obtenerunidades(){
	var xmlhttp = new XMLHttpRequest();
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				$("#listaUnidad").empty();
				$("#listaUnidadMod").empty();
				if (jsonresponse.message != null){
					var miselect = document.getElementById("listaUnidad");
					cargarlistas(jsonresponse.message, miselect);
					miselect = document.getElementById("listaUnidadMod");
					cargarlistas(jsonresponse.message, miselect);
					seleccionasensor();
				}
			}
		}
	}
	xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerlistaunidades", true);
	xmlhttp.send();
}

function seleccionasensor(){
	var seleccion = document.getElementById("listaSensor");
	var xmlhttp = new XMLHttpRequest();
	xmlhttp.onreadystatechange = function(){
		if ((this.readyState == 4)&&(this.status == 200)){
			if (this.responseText != null){
				jsonresponse = JSON.parse(this.responseText);
				if (jsonresponse.message != null){
					document.getElementById("idsensortipo").value =
jsonresponse.message.tipo;
					document.getElementById("idsensordes").value =
jsonresponse.message.descripcion;
					document.getElementById("idsensorformato").value =
jsonresponse.message.formato;
```

```
                                                document.getElementById("listaUnidad").value =
jsonresponse.message.unidad;
                                                document.getElementById("acalib").value = jsonresponse.message.a;
                                                document.getElementById("bcalib").value = jsonresponse.message.b;
                                                document.getElementById("ccalib").value = jsonresponse.message.c;
                                        }
                                }
                        }
                }
                xmlhttp.open("GET", "http://" + ip + "/wsn/v1/obtenerdatossensores/" + seleccion.value, true);
                xmlhttp.send();
}

// Inserta un sensor en la base de datos
function insertarsensor(){
                xmlhttp = new XMLHttpRequest();
                var url = "http://" + ip + "/wsn/v1/insertarsensor";
                xmlhttp.open("POST", url, true);
                xmlhttp.setRequestHeader("Content-type", "application/json");
                xmlhttp.onreadystatechange = function () {
                        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                                var jsonresponse = JSON.parse(xmlhttp.responseText);
                                if (jsonresponse.message != null){
                                        alerts(jsonresponse.error,jsonresponse.message,"#sensor");
                                        obtenerunidades();
                                        resetUnit();
                                        obtenersensores();
                                }
                        }
                }

                var tipo = document.getElementById("idsensortipo");
                var descripcion = document.getElementById("idsensordes");
                var formato = document.getElementById("idsensorformato");
                var longitud = document.getElementById("idsensorformato");
                var unidad;
                var insertUnit;
                if (document.getElementById("desunidad").value == ""){
                        unidad = document.getElementById("listaUnidad");
                        insertUnit = 0;
                } else {
                        unidad = document.getElementById("desunidad");
                        insertUnit = 1;
                }
                var a = document.getElementById("acalib");
                var b = document.getElementById("bcalib");
                var c = document.getElementById("ccalib");
                var data = JSON.stringify({
                        tipo: tipo.value,
                        descripcion: descripcion.value,
                        formato: formato.value,
                        longitud: longitud.value.length,
                        unidad: unidad.value,
                        a: a.value,
                        b: b.value,
                        c: c.value,
                        insertUnit: insertUnit
                });
                xmlhttp.send(data);
}

// Modifica un sensor de la base de datos
function modificarsensor(){
                var seleccion = document.getElementById("listaSensor");
                xmlhttp = new XMLHttpRequest();
                var url = "http://" + ip + "/wsn/v1/modificarsensor/" + seleccion.value;
                xmlhttp.open("POST", url, true);
                xmlhttp.setRequestHeader("Content-type", "application/json");
                xmlhttp.onreadystatechange = function () {
                        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                                var jsonresponse = JSON.parse(xmlhttp.responseText);
```

```
                        if (jsonresponse.message != null){
                                alerts(jsonresponse.error,jsonresponse.message,"#sensor");
                                obtenerunidades();
                                resetUnit();
                                obtenersensores();
                        }
                }
        }

        var tipo = document.getElementById("idsensortipo");
        var descripcion = document.getElementById("idsensordes");
        var formato = document.getElementById("idsensorformato");
        var longitud = document.getElementById("idsensorformato");
        var unidad;
        var insertUnit;
        if (document.getElementById("desunidad").value == ""){
                unidad = document.getElementById("listaUnidad");
                insertUnit = 0;
        } else {
                unidad = document.getElementById("desunidad");
                insertUnit = 1;
        }
        var a = document.getElementById("acalib");
        var b = document.getElementById("bcalib");
        var c = document.getElementById("ccalib");
        var data = JSON.stringify({
                tipo: tipo.value,
                descripcion: descripcion.value,
                formato: formato.value,
                longitud: longitud.value.length,
                unidad: unidad.value,
                a: a.value,
                b: b.value,
                c: c.value,
                insertUnit: insertUnit
        });
        xmlhttp.send(data);
}

// Borra un sensor de la base de datos
function borrarsensor() {
        var seleccion = document.getElementById("listaSensor");
        var xmlhttp = new XMLHttpRequest();
        xmlhttp.onreadystatechange = function(){
                if ((this.readyState == 4)&&(this.status == 200)){
                        if (this.responseText != null){
                                jsonresponse = JSON.parse(this.responseText);
                                if (jsonresponse.message != null){
                                        alerts(jsonresponse.error,jsonresponse.message,"#sensor");
                                        obtenersensores();
                                }
                        }
                }
        }
        xmlhttp.open("GET", "http://" + ip + "/wsn/v1/borrarsensor/" + seleccion.value, true);
        xmlhttp.send();
}

// **********************
// ***  UNIDADES  ****
// **********************

// Modifica una unidad de la base de datos
function modificarunidad() {
        var seleccion = document.getElementById("listaUnidadMod");
        var otraselec = document.getElementById("iddesunidad");
        var xmlhttp = new XMLHttpRequest();
        xmlhttp.onreadystatechange = function(){
                if ((this.readyState == 4)&&(this.status == 200)){
                        if (this.responseText != null){
                                jsonresponse = JSON.parse(this.responseText);
```

```
                                        if (jsonresponse.message != null){
                                                alerts(jsonresponse.error,jsonresponse.message,"#unidad");
                                                obtenerunidades();
                                                resetUnitMod();
                                        }
                                }
                        }
                }
                xmlhttp.open("GET", "http://" + ip + "/wsn/v1/modificarunidad/" + seleccion.value + "/" + otraselec.value, true);
                xmlhttp.send();
}

// Borra una unidad de la base de datos
function borrarunidad() {
                var seleccion = document.getElementById("listaUnidadMod");
                var xmlhttp = new XMLHttpRequest();
                xmlhttp.onreadystatechange = function(){
                        if ((this.readyState == 4)&&(this.status == 200)){
                                if (this.responseText != null){
                                        jsonresponse = JSON.parse(this.responseText);
                                        if (jsonresponse.message != null){
                                                alerts(jsonresponse.error,jsonresponse.message,"#unidad");
                                                obtenerunidades();
                                                obtenersensores();
                                        }
                                }
                        }
                }
                xmlhttp.open("GET", "http://" + ip + "/wsn/v1/borrarunidad/" + seleccion.value, true);
                xmlhttp.send();
}

function resetTrama() {
                document.getElementById("tipotrama").value = "";
}

function resetUnit() {
                document.getElementById("desunidad").value = "";
}

function resetUnitMod() {
                document.getElementById("iddesunidad").value = "";
}

function alerts(error, msg, direccion){
                var tipo;
                var cabeza;
                if (error == true) {
                        tipo = "danger";
                        cabeza = "¡Ha ocurrido un error!"
                } else {
                        tipo = "success"
                        cabeza = "¡Operación realizada!"
                }
                var insertAlerts = '<div class="alert alert-' + tipo + ' alert-dismissable fade in""><a href="#" class="close" data-
dismiss="alert" aria-label="close">&times;</a><strong>' + cabeza + '</strong> ' + msg + '</div>';
                $(direccion).after(insertAlerts);
}

function checks(id, tipo, longitud){
                var checkin = '<label class="checkbox-inline"><input type="checkbox" value="' + longitud + '" id="' + id + '">' + tipo +
'</label>';
                $("#catalogosensores").append(checkin);
}
```

# ANEXO II – CÓDIGO COMPLETO DE LOS SERVICIOS WEB.

```
<?php
header("Access-Control-Allow-Origin: *");
header('Access-Control-Allow-Credentials: true');
header('Access-Control-Allow-Methods: PUT, GET, POST, DELETE, OPTIONS');
//header("Access-Control-Allow-Headers: X-Requested-With");
header('Access-Control-Allow-Headers: Origin, Content-Type, X-Auth-Token');
header('Content-Type: text/html; charset=utf-8');
header('P3P: CP="IDC DSP COR CURa ADMa OUR IND PHY ONL COM STA"');

use \Psr\Http\Message\ServerRequestInterface as Request;
use \Psr\Http\Message\ResponseInterface as Response;

require '../libs/Slim/vendor/autoload.php';

$app = new \Slim\App;

/* Usando GET para crear un servicio simple */
// GET localhost/wsn/v1/hola/miguel
// Lo dejo para probar que funciona rápidamente
$app->get('/hola/{name}', function (Request $request, Response $response) {
   $name = $request->getAttribute('name');
   $response->getBody()->write("Hola, $name");
   return $response;
});



        //***************
        //-INSTALACIONES-
        //***************



//*****************************************************
// POST para insertar una instalación
// se puede probar fácilmente con postman
// POST localhost/wsn/v1/insertarinstalacion
// Body tipo raw (application/json)
/* {
        "nombre": "nombre de la finca",
        "latitud": "valor asociado como string",
        "longitud": "valor asociado como string"
}*/
//*****************************************************
$app->post('/insertarinstalacion', function(Request $request, Response $response) {

   // Obtenemos los parámetros del body
   // Que vienen codificados en JSON
   $bodyPost = $request->getParsedBody();
   $nombre = $bodyPost['nombre'];
```

```
    $latitud = $bodyPost['latitud'];
    $longitud = $bodyPost['longitud'];

    // Parámetros de la base de datos
    $servername = "localhost";
    $username = "root";
    $password = “raspberry”;

    try
    {
            $dbname = "raspberrypi";
            $conn = mysqli_connect($servername, $username, $password, $dbname);
            // Comprobación de la conexión
            if (!$conn) {  // Se devuelve lo siguiente si falla la conexión
                    $data["error"] = true;
                    $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                    $newResponse = $response->withJson($data);
                    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                    return $newResponse;
            }
            // Comprobar si ya existe la instalación en la tabla
            // SELECT `nombre` FROM `instalacion` WHERE `nombre` = "Finca de jumilla 1"
            // Sólo compruebo que no se repita el nombre de la instalación
            $sql = "SELECT `nombre` FROM `instalacion` WHERE `nombre` = '";
            $sql = $sql . $nombre;
            $sql = $sql . "'";
            $result = mysqli_query($conn, $sql);
            if (mysqli_num_rows($result) > 0) { // True: ya existe y devuelvo un error
                    $msg["error"] = true;
                    $msg["message"] = "La instalación: " . $nombre . " ya existe";
                    $newResponse = $response->withJson($msg);
                    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                    return $newResponse;
            } else {  // Si la instalación no existe la insertamos
                    //Insertar la instalacion
                    //INSERT INTO `instalacion`(`id_inst`, `nombre`, `latitud`, `longitud`) VALUES
(NULL,'nombre','latitud','longitud')
                    $sql = "INSERT INTO `instalacion`(`id_inst`, `nombre`, `latitud`, `longitud`) VALUES (NULL, '";
                    $sql = $sql . $nombre;
                    $sql = $sql . "', '";
                    $sql = $sql . $latitud;
                    $sql = $sql . "', '";
                    $sql = $sql . $longitud;
                    $sql = $sql . "')";
                    $result = mysqli_query($conn, $sql);
                    if($result == TRUE) {
                            $msg["error"] = false;
                            $msg["message"] = "Instalación: " . $nombre . " insertada correctamente";
                            $newResponse = $response->withJson($msg);
                            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                            return $newResponse;
                    } else {
                            $msg["error"] = true;
                            $msg["message"] = "Error insertando la instalación: " . $nombre;
                            $newResponse = $response->withJson($msg);
                            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                            return $newResponse;
                    }
            }
    }
    catch(PDOException $e)
    {
            $data["error"] = true;
            $data["message"] = "Fallo de conexión: " . $e->getMessage();
            $newResponse = $response->withJson($data);
            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
            return $newResponse;
    }
    mysqli_close($conn);
});
```

```
/* Usando GET para consultar la lista de instalaciones */
// GET localhost/wsn/v1/obtenerlistainstaciones
$app->get('/obtenerlistainstaciones', function(Request $request, Response $response) {

  // Nos conectamos a la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;

  try
  {
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {
              $data["error"] = true;
              $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
              $newResponse = $response->withJson($data);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        }
        // SELECT `id_inst`, `nombre` FROM `instalacion` WHERE 1
        $sql = "SELECT `id_inst`, `nombre` FROM `instalacion` WHERE 1";
        $result = mysqli_query($conn, $sql);
    $data = array();
        if (mysqli_num_rows($result) > 0) {
              while($row = mysqli_fetch_assoc($result)) {
                    $objeto = array('id' => $row["id_inst"], 'title' => $row["nombre"]);
                    array_push($data, $objeto);
              }
              $msg["error"] = false;
              $msg["message"] = $data;
              $newResponse = $response->withJson($msg);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        } else {
              $msg["error"] = true;
              $msg["message"] = "No hay instalaciones definidas";
              $newResponse = $response->withJson($msg);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        }
  }
  catch(PDOException $e)
  {
        $data["error"] = true;
    $data["message"] = "Fallo de conexión: " . $e->getMessage();
    $newResponse = $response->withJson($data);
    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
    return $newResponse;
  }
  mysqli_close($conn);
});

/* Usando GET para obtener los parámetros de una instalación seleccionada */
// GET localhost/wsn/v1/obtenerdatosinstalaciones/3
$app->get('/obtenerdatosinstalaciones/{idinstalacion}', function(Request $request, Response $response) {

  // Obtenemos el id de la instalación de la que queremos obtener datos
  $idinstalacion = $request->getAttribute('idinstalacion');

  // Nos conectamos a la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;

  try
  {
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
```

```
        if (!$conn) {
                $data["error"] = true;
                $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                $newResponse = $response->withJson($data);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }

        // Procesar la query
        // SELECT `nombre`, `latitud`, `longitud` FROM `instalacion` WHERE `id_inst`=1
        $sql = "SELECT `nombre`, `latitud`, `longitud` FROM `instalacion` WHERE `id_inst` = '";
        $sql = $sql . $idinstalacion;
        $sql = $sql . "'";
        $result = mysqli_query($conn, $sql);
        if (mysqli_num_rows($result) > 0) {
                if($row = mysqli_fetch_assoc($result)) {
                        $objeto = array('nombre' => $row["nombre"], 'latitud' => $row["latitud"], 'longitud' =>
$row["longitud"]);
                }
                $msg["error"] = false;
                $msg["message"] = $objeto;
                $newResponse = $response->withJson($msg);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        } else {
                $msg["error"] = true;
                $msg["message"] = "La instalación seleccionada no dispone de datos";
                $newResponse = $response->withJson($msg);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
  }
  catch(PDOException $e)
  {
        $data["error"] = true;
    $data["message"] = "Fallo de conexión: " . $e->getMessage();
    $newResponse = $response->withJson($data);
    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
    return $newResponse;
  }
  mysqli_close($conn);
});

//*****************************************************
// POST localhost/wsn/v1/modificarinstalacion/{idinstalacion}
// Body tipo raw (application/json)
/* {
        "nombre": "nombre de la instalación",
        "latitud": "vslor asociado como string",
        "longitud": "valor asociado como string"
} */
//*****************************************************
$app->post('/modificarinstalacion/{idinstalacion}', function(Request $request, Response $response) {

  // Obtenemos el id de la instalación que queremos modificar
  $idinstalacion = $request->getAttribute('idinstalacion');

  // Obtenemos los parámetros del body
  // Que vienen codificados en JSON
  $bodyPost = $request->getParsedBody();
  $nombre = $bodyPost['nombre'];
  $latitud = $bodyPost['latitud'];
  $longitud = $bodyPost['longitud'];

  // Parámetros de la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;

  try
  {
```

```
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {  // Se devuelve lo siguiente si falla la conexión
                $data["error"] = true;
                $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                $newResponse = $response->withJson($data);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
        // Mediante un UPDATE actualizamos los datos de la instalación
        // UPDATE `instalacion` SET `nombre` = 'd', `latitud` = 'e', `longitud` = 'f' WHERE `instalacion`.`id_inst` = 2;
        $sql = "UPDATE `instalacion` SET `nombre` = '";
        $sql = $sql . $nombre;
        $sql = $sql . "', `latitud` = '";
        $sql = $sql . $latitud;
        $sql = $sql . "', `longitud` = '";
        $sql = $sql . $longitud;
        $sql = $sql . "' WHERE `instalacion`.`id_inst` = '";
        $sql = $sql . $idinstalacion;
        $sql = $sql . "'";
        $result = mysqli_query($conn, $sql);
        if ($result == TRUE) {
                $msg["error"] = false;
                $msg["message"] = "Instalación: " . $nombre . " modificada";
                $newResponse = $response->withJson($msg);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        } else {
                $msg["error"] = true;
                $msg["message"] = "Error al modificar la instalación: " . $nombre;
                $newResponse = $response->withJson($msg);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
    }
    catch(PDOException $e)
    {
        $data["error"] = true;
      $data["message"] = "Fallo de conexión: " . $e->getMessage();
      $newResponse = $response->withJson($data);
      $newResponse = $newResponse->withHeader('Content-type', 'application/json');
      return $newResponse;
    }
    mysqli_close($conn);
});

/* Usando GET para borrar una instalación seleccionada */
// GET localhost/wsn/v1/borrarinstalacion/3
$app->get('/borrarinstalacion/{idinstalacion}', function(Request $request, Response $response) {

    // Obtenemos el id de la instalación que queremos borrar
    $idinstalacion = $request->getAttribute('idinstalacion');

    // Nos conectamos a la base de datos
    $servername = "localhost";
    $username = "root";
    $password = “raspberry”;

    try
    {
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {
                $data["error"] = true;
                $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                $newResponse = $response->withJson($data);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
```

```
            // DELETE FROM `instalacion` WHERE `id_inst` = '3';
            $sql = "DELETE FROM `instalacion` WHERE `id_inst` = '";
            $sql = $sql . $idinstalacion;
            $sql = $sql . "'";
            $result = mysqli_query($conn, $sql);
            if ($result == TRUE) {
                        $msg["error"] = false;
                        $msg["message"] = "Instalación borrada";
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
            } else {
                        $msg["error"] = true;
                        $msg["message"] = "No existen registros de la instalación seleccionada";
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
            }
   }
   catch(PDOException $e)
   {
            $data["error"] = true;
      $data["message"] = "Fallo de conexión: " . $e->getMessage();
      $newResponse = $response->withJson($data);
      $newResponse = $newResponse->withHeader('Content-type', 'application/json');
      return $newResponse;
   }
   mysqli_close($conn);
});



            //***************
            //-   NODOS   -
            //***************



//******************************************************
// POST localhost/wsn/v1/insertarnodo
// Body tipo raw (application/json)
/* {
            "descripcion": "número del nodo",
            "latitud": "valor asociado como string",
            "longitud": "valor asociado como string",
            "instalacion": "indice de la instalación",
            "protocolo": "indice del protocolo"
            "trama": "tipo de trama",
            "array_id": "array de id de los sensores",
            "array_long": "array de longitudes los datos recogidos por los sensores",
            "longitud_total": "longitud total de la trama"
} */
//******************************************************
$app->post('/insertarnodo', function(Request $request, Response $response) {

   // Obtenemos los parámetros del body
   // Que vienen codificados en JSON
   $bodyPost = $request->getParsedBody();
   $descripcion = $bodyPost['descripcion'];
   $latitud = $bodyPost['latitud'];
   $longitud = $bodyPost['longitud'];
   $instalacion = $bodyPost['instalacion'];
   $protocolo = $bodyPost['protocolo'];
   $trama = $bodyPost['trama'];
   $array_id = $bodyPost['array_id'];
   $array_long = $bodyPost['array_long'];
   $longitud_total = $bodyPost['longitud_total'];

   // Parámetros de la base de datos
   $servername = "localhost";
```

```
$username = "root";
$password = “raspberry”;

try
{
	$dbname = "raspberrypi";
	$conn = mysqli_connect($servername, $username, $password, $dbname);
	// Comprobación de la conexión
	if (!$conn) {		// Se devuelve lo siguiente si falla la conexión
		$data["error"] = true;
		$data["message"] = "Fallo de conexión: " . mysqli_connect_error();
		$newResponse = $response->withJson($data);
		$newResponse = $newResponse->withHeader('Content-type', 'application/json');
		return $newResponse;
	}

	// Comprobar si ya existe el nodo en la tabla
	// SELECT * FROM `nodo` WHERE `id_nodo` = '1'
	// Sólo compruebo que no se repita el número del nodo
	$sql = "SELECT * FROM `nodo` WHERE `id_nodo` = '";
	$sql = $sql . $descripcion;
	$sql = $sql . "'";
	$result = mysqli_query($conn, $sql);
	if (mysqli_num_rows($result) > 0) { // True: ya existe y devuelvo un error
		$msg["error"] = true;
		$msg["message"] = "El nodo " . $descripcion . " ya existe";
		$newResponse = $response->withJson($msg);
		$newResponse = $newResponse->withHeader('Content-type', 'application/json');
		return $newResponse;
	} else { // Si el nodo no existe lo insertamos
		// Comprobamos si la trama existe
		// SELECT `id_trama` FROM `trama` WHERE ((`tipo` = "D30001")AND(`array_id_sensor` = "1,2,3,4,8,9,10"))
		$sql = "SELECT `id_trama` FROM `trama` WHERE ((`tipo` = '";
		$sql = $sql . $trama;
		$sql = $sql . "')AND(`array_id_sensor` = '";
		$sql = $sql . $array_id;
		$sql = $sql . "'))";
		$result = mysqli_query($conn, $sql);
		if (mysqli_num_rows($result) == 0) {
			// No existe, la creo
			// INSERT INTO `trama`(`id_trama`, `tipo`, `longitud`, `array_id_sensor`, `array_longitud`) VALUES
(NULL, 'D3001', '7', '1,2,3', '1,2,3')
			$sql = "INSERT INTO `trama`(`id_trama`, `tipo`, `longitud`, `array_id_sensor`, `array_longitud`)
VALUES (NULL, '";
			$sql = $sql . $trama;
			$sql = $sql . "', '";
			$sql = $sql . $longitud_total;
			$sql = $sql . "', '";
			$sql = $sql . $array_id;
			$sql = $sql . "', '";
			$sql = $sql . $array_long;
			$sql = $sql . "')";
			$result = mysqli_query($conn, $sql);
			if ($result == FALSE) {
				$msg["error"] = true;
				$msg["message"] = "Error al crear la trama nueva";
				$newResponse = $response->withJson($msg);
				$newResponse = $newResponse->withHeader('Content-type', 'application/json');
				return $newResponse;
			}
			// Obtenemos el id de la trama que acabamos de crear
			$sql = "SELECT `id_trama` FROM `trama` WHERE ((`tipo` = '";
			$sql = $sql . $trama;
			$sql = $sql . "')AND(`array_id_sensor` = '";
			$sql = $sql . $array_id;
			$sql = $sql . "'))";
			$result = mysqli_query($conn, $sql);
		}
		// Metemos el id de la trama seleccionada en una nueva variable
		if ($row = mysqli_fetch_assoc($result)) {
			$idtrama = $row["id_trama"];
```

```
                        }

                        //Insertar el nodo
                        //INSERT INTO `nodo` (`id_nodo`, `descripcion`, `latitud`, `longitud`, `alta_nodo`, `last_update`, `_id_inst`,
`_id_prot`, `_id_trama`) VALUES (NULL, 'nodo 20', '32', '32', CURRENT_TIMESTAMP, '0000-00-00 00:00:00.000000', '1', '1', '1');
                        $sql = "INSERT INTO `nodo` (`id_nodo`, `descripcion`, `latitud`, `longitud`, `alta_nodo`, `last_update`,
`_id_inst`, `_id_prot`, `_id_trama`) VALUES ('";
                        $sql = $sql . $descripcion;
                        $sql = $sql . "', 'Nodo ";
                        $sql = $sql . $descripcion;
                        $sql = $sql . "', '";
                        $sql = $sql . $latitud;
                        $sql = $sql . "', '";
                        $sql = $sql . $longitud;
                        $sql = $sql . "', CURRENT_TIMESTAMP, '0000-00-00 00:00:00', '";
                        $sql = $sql . $instalacion;
                        $sql = $sql . "', '";
                        $sql = $sql . $protocolo;
                        $sql = $sql . "', '";
                        $sql = $sql . $idtrama;
                        $sql = $sql . "')";
                        $result = mysqli_query($conn, $sql);
                        if ($result == TRUE) {
                                // Creamos los canales del nuevo nodo, de acuerdo a los sensores seleccionados
                                // INSERT INTO `canal` (`id_chan`, `_id_type`, `_id_nodo`) VALUES (NULL, '1', '10'), (NULL, '2', '10');
                                $arr = explode(",", $array_id);
                                $flag = 0;  // Inicio de valores
                                $sql = "INSERT INTO `canal` (`id_chan`, `_id_type`, `_id_nodo`) VALUES ";
                                for ($i = 0; $i < count($arr); $i++){
                                        if ($flag == 0) {
                                                $sql = $sql . "('";
                                                $flag = 1;
                                        } else {
                                                $sql = $sql . "), ('";
                                        }
                                        $idcanal = $descripcion * 100 + $arr[$i];
                                        $sql = $sql . $idcanal;
                                        $sql = $sql . "', '";
                                        $sql = $sql . $arr[$i];
                                        $sql = $sql . "', '";
                                        $sql = $sql . $descripcion;
                                        $sql = $sql . "')";
                                }
                                $result = mysqli_query($conn, $sql);
                                if($result == TRUE){
                                        $msg["error"] = false;
                                        $msg["message"] = "Nodo " . $descripcion . " insertado correctamente";
                                        $newResponse = $response->withJson($msg);
                                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                                        return $newResponse;
                                }else{
                                        $msg["error"] = true;
                                        $msg["message"] = "Error al insertar los canales del Nodo " . $descripcion;
                                        $newResponse = $response->withJson($msg);
                                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                                        return $newResponse;
                                }
                        }else{
                                $msg["error"] = true;
                                $msg["message"] = "Error al insertar el Nodo " . $descripcion;
                                $newResponse = $response->withJson($msg);
                                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                                return $newResponse;
                        }
                }
        }
        catch(PDOException $e)
        {
                $data["error"] = true;
            $data["message"] = "Fallo de conexión: " . $e->getMessage();
            $newResponse = $response->withJson($data);
```

```
    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
    return $newResponse;
  }
  mysqli_close($conn);
});

/* Usando GET para consultar la lista de nodos */
// GET localhost/wsn/v1/obtenerlistanodos
$app->get('/obtenerlistanodos', function(Request $request, Response $response) {

  // Nos conectamos a la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;

  try
  {
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {
              $data["error"] = true;
              $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
              $newResponse = $response->withJson($data);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        }
        // SELECT `id_nodo`, `descripcion` FROM `nodo` WHERE 1
        $sql = "SELECT `id_nodo`, `descripcion` FROM `nodo` WHERE 1";
        $result = mysqli_query($conn, $sql);
    $data = array();
        if (mysqli_num_rows($result) > 0) {
              while($row = mysqli_fetch_assoc($result)) {
                    $objeto = array('id' => $row["id_nodo"], 'title' => $row["descripcion"]);
                    array_push($data, $objeto);
              }
              $msg["error"] = false;
              $msg["message"] = $data;
              $newResponse = $response->withJson($msg);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        } else {
              $msg["error"] = true;
              $msg["message"] = "No hay ningún nodo definido";
              $newResponse = $response->withJson($msg);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        }
  }
  catch(PDOException $e)
  {
        $data["error"] = true;
    $data["message"] = "Fallo de conexión: " . $e->getMessage();
    $newResponse = $response->withJson($data);
    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
    return $newResponse;
  }
  mysqli_close($conn);
});

/* Usando GET para consultar la lista de protocolos */
// GET localhost/wsn/v1/obtenerlistaprotocolos
$app->get('/obtenerlistaprotocolos', function(Request $request, Response $response) {

  // Nos conectamos a la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;

  try
  {
```

```
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {
                $data["error"] = true;
                $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                $newResponse = $response->withJson($data);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
        // SELECT `id_prot`, `tipo_prot` FROM `protocolo` WHERE 1
        $sql = "SELECT `id_prot`, `tipo_prot` FROM `protocolo` WHERE 1";
        $result = mysqli_query($conn, $sql);
    $data = array();
        if (mysqli_num_rows($result) > 0) {
                // output data of each row
                while($row = mysqli_fetch_assoc($result)) {
                        $objeto = array('id' => $row["id_prot"], 'title' => $row["tipo_prot"]);
                        array_push($data, $objeto);
                }
                $msg["error"] = false;
                $msg["message"] = $data;
                $newResponse = $response->withJson($msg);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        } else {
                $msg["error"] = true;
                $msg["message"] = "No hay protocolos definidos";
                $newResponse = $response->withJson($msg);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
  }
  catch(PDOException $e)
  {
        $data["error"] = true;
    $data["message"] = "Fallo de conexión: " . $e->getMessage();
    $newResponse = $response->withJson($data);
    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
    return $newResponse;
  }
  mysqli_close($conn);
});

/* Usando GET para consultar la lista de tramas */
// GET localhost/wsn/v1/obtenerlistatramas
$app->get('/obtenerlistatramas', function(Request $request, Response $response) {

  // Nos conectamos a la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;

  try
  {
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {
                $data["error"] = true;
                $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                $newResponse = $response->withJson($data);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
        // SELECT `tipo` FROM `trama` GROUP BY `tipo
        $sql = "SELECT `tipo` FROM `trama` GROUP BY `tipo`";
        $result = mysqli_query($conn, $sql);
    $data = array();
        if (mysqli_num_rows($result) > 0) {
                // output data of each row
```

```
                        while($row = mysqli_fetch_assoc($result)) {
                                $objeto = array('id' => $row["tipo"], 'title' => $row["tipo"]);
                                array_push($data, $objeto);
                        }
                        $msg["error"] = false;
                        $msg["message"] = $data;
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                } else {
                        $msg["error"] = true;
                        $msg["message"] = "No hay tramas definidas";
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
        }
        catch(PDOException $e)
        {
                $data["error"] = true;
            $data["message"] = "Fallo de conexión: " . $e->getMessage();
            $newResponse = $response->withJson($data);
            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
            return $newResponse;
        }
        mysqli_close($conn);
});

/* Usando GET para obtener los parámetros del nodo seleccionado */
// GET localhost/wsn/v1/obtenerdatosnodos/3
$app->get('/obtenerdatosnodos/{idnodo}', function(Request $request, Response $response) {

        // Obtenemos el id del nodo del que queremos obtener datos
        $idnodo = $request->getAttribute('idnodo');

        // Nos conectamos a la base de datos
        $servername = "localhost";
        $username = "root";
        $password = “raspberry”;

        try
        {
                $dbname = "raspberrypi";
                $conn = mysqli_connect($servername, $username, $password, $dbname);
                // Comprobación de la conexión
                if (!$conn) {
                        $data["error"] = true;
                        $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                        $newResponse = $response->withJson($data);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
                // Procesar la query
                // SELECT nodo.id_nodo, nodo.latitud, nodo.longitud, nodo._id_inst, nodo._id_inst, nodo._id_prot, trama.tipo,
trama.array_id_sensor FROM nodo JOIN trama ON nodo._id_trama = trama.id_trama WHERE nodo.id_nodo = 3
                $sql = "SELECT nodo.id_nodo, nodo.latitud, nodo.longitud, nodo._id_inst, nodo._id_inst, nodo._id_prot, trama.tipo,
trama.array_id_sensor FROM nodo JOIN trama ON nodo._id_trama = trama.id_trama WHERE nodo.id_nodo = '";
                $sql = $sql . $idnodo;
                $sql = $sql . "'";
                $result = mysqli_query($conn, $sql);
                if (mysqli_num_rows($result) > 0) {
                        // output data of each row
                        if($row = mysqli_fetch_assoc($result)) {
                                $objeto = array('descripcion' => $row["id_nodo"], 'latitud' => $row["latitud"], 'longitud' =>
$row["longitud"], 'instalacion' => $row["_id_inst"], 'protocolo' => $row["_id_prot"], 'trama' => $row["tipo"], 'idsensor' =>
$row["array_id_sensor"]);
                        }
                        $msg["error"] = false;
                        $msg["message"] = $objeto;
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
```

```
                return $newResponse;
        } else {
                $msg["error"] = true;
                $msg["message"] = "No hay registrados datos del Nodo " . $idnodo;
                $newResponse = $response->withJson($msg);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
  }
  catch(PDOException $e)
  {
        $data["error"] = true;
    $data["message"] = "Fallo de conexión: " . $e->getMessage();
    $newResponse = $response->withJson($data);
    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
    return $newResponse;
  }
  mysqli_close($conn);
});

//*****************************************************
// POST localhost/wsn/v1/modificarnodo/{idnodo}
// Body tipo raw (application/json)
/* {
        "latitud": "valor asociado como string",
        "longitud": "valor asociado como string",
        "instalacion": "indice de la instalación",
        "protocolo": "indice del protocolo"
        "trama": "tipo de trama",
        "array_id": "array de id de los sensores",
        "array_long": "array de longitudes los datos recogidos por los sensores",
        "longitud_total": "longitud total de la trama"
        } */
//*****************************************************
$app->post('/modificarnodo/{idnodo}', function(Request $request, Response $response) {

  // Obtenemos el id del nodo que queremos modificar
  $idnodo = $request->getAttribute('idnodo');

  // Obtenemos los parámetros del body
  // Que vienen codificados en JSON
  $bodyPost = $request->getParsedBody();
  $latitud = $bodyPost['latitud'];
  $longitud = $bodyPost['longitud'];
  $instalacion = $bodyPost['instalacion'];
  $protocolo = $bodyPost['protocolo'];
  $trama = $bodyPost['trama'];
  $array_id = $bodyPost['array_id'];
  $array_long = $bodyPost['array_long'];
  $longitud_total = $bodyPost['longitud_total'];

  // Parámetros de la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;

  try
  {
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {  // Se devuelve lo siguiente si falla la conexión
                $data["error"] = true;
                $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                $newResponse = $response->withJson($data);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
        // Comprobamos si la trama existe
        // SELECT `id_trama` FROM `trama` WHERE ((`tipo` = "D30001")AND(`array_id_sensor` = "1,2,3,4,8,9,10"))
        $sql = "SELECT `id_trama` FROM `trama` WHERE ((`tipo` = '";
```

```
		$sql = $sql . $trama;
		$sql = $sql . "')AND(`array_id_sensor` = '";
		$sql = $sql . $array_id;
		$sql = $sql . "'))";
		$result = mysqli_query($conn, $sql);
		if (mysqli_num_rows($result) == 0) {
			// No existe, la creo
			// INSERT INTO `trama`(`id_trama`, `tipo`, `longitud`, `array_id_sensor`, `array_longitud`) VALUES (NULL,
'D3001', '7', '1,2,3', '1,2,3')
			$sql = "INSERT INTO `trama`(`id_trama`, `tipo`, `longitud`, `array_id_sensor`, `array_longitud`) VALUES
(NULL,'";
			$sql = $sql . $trama;
			$sql = $sql . "', '";
			$sql = $sql . $longitud_total;
			$sql = $sql . "', '";
			$sql = $sql . $array_id;
			$sql = $sql . "', '";
			$sql = $sql . $array_long;
			$sql = $sql . "')";
			$result = mysqli_query($conn, $sql);
			if($result == FALSE){
				$msg["error"] = true;
				$msg["message"] = "Error al insertar la trama";
				$newResponse = $response->withJson($msg);
				$newResponse = $newResponse->withHeader('Content-type', 'application/json');
				return $newResponse;
			}
			// Obtenemos el id de la trama que acabamos de crear
			$sql = "SELECT `id_trama` FROM `trama` WHERE ((`tipo` = '";
			$sql = $sql . $trama;
			$sql = $sql . "')AND(`array_id_sensor` = '";
			$sql = $sql . $array_id;
			$sql = $sql . "'))";
			$result = mysqli_query($conn, $sql);
		}
		if ($row = mysqli_fetch_assoc($result)){
			$idtrama = $row["id_trama"];
		}
		// Mediante un UPDATE actualizamos los datos del nodo
		// UPDATE `nodo` SET `latitud` = 'e', `longitud` = 'f', `_id_inst` = '1', `_id_prot` = '3', `_id_trama` = '3' WHERE `id_nodo` =
2;
		$sql = "UPDATE `nodo` SET `latitud` = '";
		$sql = $sql . $latitud;
		$sql = $sql . "', `longitud` = '";
		$sql = $sql . $longitud;
		$sql = $sql . "', `_id_inst` = '";
		$sql = $sql . $instalacion;
		$sql = $sql . "', `_id_prot` = '";
		$sql = $sql . $protocolo;
		$sql = $sql . "', `_id_trama` = '";
		$sql = $sql . $idtrama;
		$sql = $sql . "' WHERE `id_nodo` = '";
		$sql = $sql . $idnodo;
		$sql = $sql . "'";
		$result = mysqli_query($conn, $sql);
		if ($result == TRUE) {
			// Borramos los canales antiguos e insertamos los nuevos
			// DELETE FROM `canal` WHERE `_id_nodo` = 3;
			$sql = "DELETE FROM `canal` WHERE `_id_nodo` = '";
			$sql = $sql . $idnodo;
			$sql = $sql . "'";
			$result = mysqli_query($conn, $sql);
			// Creamos los nuevos canales del nodo modificado
			// INSERT INTO `canal` (`id_chan`, `_id_type`, `_id_nodo`) VALUES (NULL, '1', '10'), (NULL, '2', '10');
			$arr = explode(",", $array_id);
			$flag = 0;
			$sql = "INSERT INTO `canal` (`id_chan`, `_id_type`, `_id_nodo`) VALUES ";
			for ($i = 0; $i < count($arr); $i++) {
				if ($flag == 0) {
					$sql = $sql . "('";
					$flag = 1;
```

```
                                } else {
                                        $sql = $sql . ", ('";
                                }
                                $idcanal = $idnodo * 100 + $arr[$i];
                                $sql = $sql . $idcanal;
                                $sql = $sql . "', '";
                                $sql = $sql . $arr[$i];
                                $sql = $sql . "', '";
                                $sql = $sql . $idnodo;
                                $sql = $sql . "')";
                        }
                        $result = mysqli_query($conn, $sql);
                        if ($result == TRUE) {
                                // Eliminamos las tramas que estén en desuso
                                // DELETE FROM `trama` WHERE `id_trama` NOT IN (SELECT `_id_trama` FROM `nodo` GROUP BY
`_id_trama`);
                                $sql = "DELETE FROM `trama` WHERE `id_trama` NOT IN (SELECT `_id_trama` FROM `nodo`
GROUP BY `_id_trama`)";
                                $result = mysqli_query($conn, $sql);
                                $msg["error"] = false;
                                $msg["message"] = "Nodo " . $idnodo . " modificado correctamente";
                                $newResponse = $response->withJson($msg);
                                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                                return $newResponse;
                        } else {
                                $msg["error"] = true;
                                $msg["message"] = "Error insertando los canales del Nodo " . $idnodo;
                                $newResponse = $response->withJson($msg);
                                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                                return $newResponse;
                        }
                } else {
                        $msg["error"] = true;
                        $msg["message"] = "Error modificando el Nodo " . $idnodo;
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
        }
        catch(PDOException $e)
        {
                $data["error"] = true;
            $data["message"] = "Fallo de conexión: " . $e->getMessage();
            $newResponse = $response->withJson($data);
            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
            return $newResponse;
        }
        mysqli_close($conn);
});

/* Usando GET para borrar un nodo seleccionado */
// GET localhost/wsn/v1/borrarnodo/3
$app->get('/borrarnodo/{idnodo}', function(Request $request, Response $response) {

        // Obtenemos el id del nodo que queremos borrar
        $idnodo = $request->getAttribute('idnodo');

        // Nos conectamos a la base de datos
        $servername = "localhost";
        $username = "root";
        $password = “raspberry”;

        try
        {
                $dbname = "raspberrypi";
                $conn = mysqli_connect($servername, $username, $password, $dbname);
                // Comprobación de la conexión
                if (!$conn) {
                        $data["error"] = true;
                        $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                        $newResponse = $response->withJson($data);
```

```
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }

                // Procesar la query
                // DELETE FROM `nodo` WHERE `id_nodo` = 3;
                $sql = " DELETE FROM `nodo` WHERE `id_nodo` = '";
                $sql = $sql . $idnodo;
                $sql = $sql . "'";
                $result = mysqli_query($conn, $sql);
                if ($result == TRUE) {
                        $msg["error"] = false;
                        $msg["message"] = "Nodo " . $idnodo . " borrado";
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                } else {
                        $msg["error"] = true;
                        $msg["message"] = "No se encuentra el Nodo " . $idnodo;
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
        }
        catch(PDOException $e)
        {
                $data["error"] = true;
            $data["message"] = "Fallo de conexión: " . $e->getMessage();
            $newResponse = $response->withJson($data);
            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
            return $newResponse;
        }
        mysqli_close($conn);
});



                //**************
                //-  SENSORES  -
                //**************



//******************************************************
// POST localhost/wsn/v1/insertarsensor
// Body tipo raw (application/json)
/* {
                "tipo": "abreviatura del sensor",
                "descripcion": "descripción del sensor",
                "formato": "valor asociado como string que nos da el formato de los valores del sensor",
                "longitud": "valor entero de que mide la longitud del formato",
                "unidad": "unidades en las que mide el sensor",
                "a": "parámetro de calibración",
                "b": "parámetro de calibración",
                "c": "parámetro de calibración"
                "insertUnit": bandera para saber si hay que insertar o no una unidad nueva
                }*/
//******************************************************
$app->post('/insertarsensor', function(Request $request, Response $response) {

        // Obtenemos los parámetros del body
        // Que vienen codificados en JSON
        $bodyPost = $request->getParsedBody();
        $tipo = $bodyPost['tipo'];
        $descripcion = $bodyPost['descripcion'];
        $formato = $bodyPost['formato'];
        $longitud = $bodyPost['longitud'];
        $unidad = $bodyPost['unidad'];
        $a = $bodyPost['a'];
        $b = $bodyPost['b'];
        $c = $bodyPost['c'];
```

```
    $insertUnit = $bodyPost['insertUnit'];

    // Parámetros de la base de datos
    $servername = "localhost";
    $username = "root";
    $password = “raspberry”;

    try
    {
            $dbname = "raspberrypi";
            $conn = mysqli_connect($servername, $username, $password, $dbname);
            // Comprobación de la conexión
            if (!$conn) {  // Se devuelve lo siguiente si falla la conexión
                    $data["error"] = true;
                    $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                    $newResponse = $response->withJson($data);
                    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                    return $newResponse;
            }

            // Comprobar si ya existe el sensor en la tabla
            // SELECT `descripcion` FROM `tipo_sensor` WHERE `descripcion` = 'Potencial Matricial'
            // Sólo compruebo que no se repita el nombre del sensor
            // Se pueden comprobar más cosas y poner un AND
            $sql = "SELECT `descripcion` FROM `tipo_sensor` WHERE `descripcion` = '";
            $sql = $sql . $descripcion;
            $sql = $sql . "'";
            $result = mysqli_query($conn, $sql);
            if (mysqli_num_rows($result) > 0) { // True: ya existe y devuelvo un error
                    $msg["error"] = true;
                    $msg["message"] = "El sensor ya existe";
                    $newResponse = $response->withJson($msg);
                    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                    return $newResponse;
            }else{ // El sensor no existe... lo insertamos
                    // Comprobamos si se mide con una nueva unidad, si es así la insertamos previamente
                    if($insertUnit == "1"){
                            // Comprobamos si existe la unidad "nueva"
                            $sql = "SELECT `descripcion` FROM `unidad` WHERE `descripcion` = '";
                            $sql = $sql . $unidad;
                            $sql = $sql . "'";
                            $result = mysqli_query($conn, $sql);
                            if (mysqli_num_rows($result) == 0) {
                                    // INSERT INTO `unidad` (`id_unit`, `descripcion`) VALUES (NULL, 'Grados Kelvin');
                                    $sql = "INSERT INTO `unidad` (`id_unit`, `descripcion`) VALUES (NULL, '";
                                    $sql = $sql . $unidad;
                                    $sql = $sql . "')";
                                    $result = mysqli_query($conn, $sql);
                                    if($result == FALSE){
                                            $msg["error"] = true;
                                            $msg["message"] = "Error insertando la unidad " + $unidad;
                                            $newResponse = $response->withJson($msg);
                                            $newResponse = $newResponse->withHeader('Content-type',
'application/json');
                                            return $newResponse;
                                    }
                            }
                            // Obtenemos el id de la unidad que acabamos de crear
                            $sql = "SELECT `id_unit` FROM `unidad` WHERE `descripcion` = '";
                            $sql = $sql . $unidad;
                            $sql = $sql . "'";
                            $result = mysqli_query($conn, $sql);
                            if ($row = mysqli_fetch_assoc($result)){
                                    $unidad = $row["id_unit"];
                            }
                    }
                    // Insertar el sensor
                    // INSERT INTO `tipo_sensor` (`id_type`, `tipo`, `descripcion`, `formato`, `longitud`, `a`, `b`, `c`, `_id_unit`)
VALUES (NULL, 'Pot. Mat. 25cm', 'Potencial Matricail', '+0.0000', '7', '3.5641', '4.489951', '-2.1458', '4');
                    $sql = "INSERT INTO `tipo_sensor` (`id_type`, `tipo`, `descripcion`, `formato`, `longitud`, `a`, `b`, `c`,
`_id_unit`) VALUES (NULL, '";
```

```
                        $sql = $sql . $tipo;
                        $sql = $sql . "', '";
                        $sql = $sql . $descripcion;
                        $sql = $sql . "', '";
                        $sql = $sql . $formato;
                        $sql = $sql . "', '";
                        $sql = $sql . $longitud;
                        $sql = $sql . "', '";
                        $sql = $sql . $a;
                        $sql = $sql . "', '";
                        $sql = $sql . $b;
                        $sql = $sql . "', '";
                        $sql = $sql . $c;
                        $sql = $sql . "', '";
                        $sql = $sql . $unidad;
                        $sql = $sql . "')";
                        $result = mysqli_query($conn, $sql);
                        if($result == TRUE){
                                $msg["error"] = false;
                                $msg["message"] = "Sensor insertado correctamente";
                                $newResponse = $response->withJson($msg);
                                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                                return $newResponse;
                        }else{
                                $msg["error"] = true;
                                $msg["message"] = "Error insertando el sensor";
                                $newResponse = $response->withJson($msg);
                                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                                return $newResponse;
                        }
                }
        }
        catch(PDOException $e)
        {
                $data["error"] = true;
                $data["message"] = "Fallo de conexión: " . $e->getMessage();
                $newResponse = $response->withJson($data);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
        mysqli_close($conn);
});

/* Usando GET para consultar la lista de sensores */
// GET localhost/wsn/v1/obtenerlistasensores
$app->get('/obtenerlistasensores', function(Request $request, Response $response) {

        // Nos conectamos a la base de datos
        $servername = "localhost";
        $username = "root";
        $password = “raspberry”;

        try
        {
                $dbname = "raspberrypi";
                $conn = mysqli_connect($servername, $username, $password, $dbname);
                // Comprobación de la conexión
                if (!$conn) {
                        $data["error"] = true;
                        $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                        $newResponse = $response->withJson($data);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
                // SELECT `id_type`, `tipo` FROM `tipo_sensor` WHERE 1
                $sql = "SELECT `id_type`, `tipo`, `longitud` FROM `tipo_sensor` WHERE 1";
                $result = mysqli_query($conn, $sql);
                $data = array();
                if (mysqli_num_rows($result) > 0) {
                        // output data of each row
                        while($row = mysqli_fetch_assoc($result)) {
```

```
                                $objeto = array('id' => $row["id_type"], 'title' => $row["tipo"], 'longitud' => $row["longitud"]);
                                array_push($data, $objeto);
                        }
                        $msg["error"] = false;
                        $msg["message"] = $data;
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                } else {
                        $msg["error"] = true;
                        $msg["message"] = "No se encuentra la lista de sensores";
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
        }
        catch(PDOException $e)
        {
                $data["error"] = true;
            $data["message"] = "Fallo de conexión: " . $e->getMessage();
            $newResponse = $response->withJson($data);
            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
            return $newResponse;
        }
        mysqli_close($conn);
});

/* Usando GET para consultar la lista de unidades */
// GET localhost/wsn/v1/obtenerlistaunidades
$app->get('/obtenerlistaunidades', function(Request $request, Response $response) {

        // Nos conectamos a la base de datos
        $servername = "localhost";
        $username = "root";
        $password = “raspberry”;

        try
        {
                $dbname = "raspberrypi";
                        $conn = mysqli_connect($servername, $username, $password, $dbname);
                // Comprobación de la conexión
                if (!$conn) {
                        $data["error"] = true;
                        $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                        $newResponse = $response->withJson($data);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
                // SELECT `id_unit`, `descripcion` FROM `unidad` WHERE 1
                $sql = "SELECT `id_unit`, `descripcion` FROM `unidad` WHERE 1";
                $result = mysqli_query($conn, $sql);
            $data = array();
                if (mysqli_num_rows($result) > 0) {
                        // output data of each row
                        while($row = mysqli_fetch_assoc($result)) {
                                $objeto = array('id' => $row["id_unit"], 'title' => $row["descripcion"]);
                                array_push($data, $objeto);
                        }
                        $msg["error"] = false;
                        $msg["message"] = $data;
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                } else {
                        $msg["error"] = true;
                        $msg["message"] = "No se encuentra la lista de unidades";
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
        }
```

```
    catch(PDOException $e)
    {
            $data["error"] = true;
        $data["message"] = "Fallo de conexión: " . $e->getMessage();
        $newResponse = $response->withJson($data);
        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
        return $newResponse;
    }
    mysqli_close($conn);
});

/* Usando GET para obtener los parámetros del nodo seleccionado */
// GET localhost/wsn/v1/obtenerdatosnodos/3
$app->get('/obtenerdatossensores/{idsensor}', function(Request $request, Response $response) {

    // Obtenemos el id del nodo del que queremos obtener datos
    $idsensor = $request->getAttribute('idsensor');

    // Nos conectamos a la base de datos
    $servername = "localhost";
    $username = "root";
    $password = “raspberry”;

    try
    {
            $dbname = "raspberrypi";
            $conn = mysqli_connect($servername, $username, $password, $dbname);
            // Comprobación de la conexión
            if (!$conn) {
                    $data["error"] = true;
                    $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                    $newResponse = $response->withJson($data);
                    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                    return $newResponse;
            }
            // Procesar la query
            // SELECT `tipo`, `descripcion`, `formato`, `_id_unit`, `a`, `b`, `c` FROM `tipo_sensor` WHERE `id_type` = 1
            $sql = "SELECT `tipo`, `descripcion`, `formato`, `_id_unit`, `a`, `b`, `c` FROM `tipo_sensor` WHERE `id_type` = '";
            $sql = $sql . $idsensor;
            $sql = $sql . "'";
            $result = mysqli_query($conn, $sql);
            if (mysqli_num_rows($result) > 0) {
                    if($row = mysqli_fetch_assoc($result)) {
                            $objeto = array('tipo' => $row["tipo"], 'descripcion' => $row["descripcion"], 'formato' =>
$row["formato"], 'unidad' => $row["_id_unit"], 'a' => $row["a"], 'b' => $row["b"], 'c' => $row["c"]);
                    }
                    $msg["error"] = false;
                    $msg["message"] = $objeto;
                    $newResponse = $response->withJson($msg);
                    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                    return $newResponse;
            } else {
                    $msg["error"] = true;
                    $msg["message"] = "No se encuentran los datos del sensor seleccionado";
                    $newResponse = $response->withJson($msg);
                    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                    return $newResponse;
            }
    }
    catch(PDOException $e)
    {
            $data["error"] = true;
        $data["message"] = "Fallo de conexión: " . $e->getMessage();
        $newResponse = $response->withJson($data);
        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
        return $newResponse;
    }
    mysqli_close($conn);
});

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

```
// POST localhost/wsn/v1/modificarsensor/{idtipo}
// Body tipo raw (application/json)
/* {
        "tipo": "abreviatura del sensor",
        "descripcion": "descripción del sensor",
        "formato": "valor asociado como string que nos da el formato de los valores del sensor",
        "longitud": "valor entero de que mide la longitud del formato",
        "unidad": "unidades en las que mide el sensor",
        "a": "parámetro de calibración",
        "b": "parámetro de calibración",
        "c": "parámetro de calibración",
        "insertUnit": bandera para saber si hay que insertar o no una unidad nueva
}*/
//*****************************************************
$app->post('/modificarsensor/{idtipo}', function(Request $request, Response $response) {

  // Obtenemos el id del nodo que queremos modificar
  $idtipo = $request->getAttribute('idtipo');

  // Obtenemos los parámetros del body
  // Que vienen codificados en JSON
  $bodyPost = $request->getParsedBody();
  $tipo = $bodyPost['tipo'];
  $descripcion = $bodyPost['descripcion'];
  $formato = $bodyPost['formato'];
  $longitud = $bodyPost['longitud'];
  $unidad = $bodyPost['unidad'];
  $a = $bodyPost['a'];
  $b = $bodyPost['b'];
  $c = $bodyPost['c'];
  $insertUnit = $bodyPost['insertUnit'];

  // Parámetros de la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;

  try
  {
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {           // Se devuelve lo siguiente si falla la conexión
                $data["error"] = true;
                $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
                $newResponse = $response->withJson($data);
                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                return $newResponse;
        }
        // Comprobamos si se mide con una nueva unidad, si es así la insertamos previamente
        if($insertUnit == "1"){
                // Comprobamos si existe la unidad "nueva"
                $sql = "SELECT `descripcion` FROM `unidad` WHERE `descripcion` = '";
                $sql = $sql . $unidad;
                $sql = $sql . "'";
                $result = mysqli_query($conn, $sql);
                if (mysqli_num_rows($result) == 0) {
                        // INSERT INTO `unidad` (`id_unit`, `descripcion`) VALUES (NULL, 'Grados Kelvin');
                        $sql = "INSERT INTO `unidad` (`id_unit`, `descripcion`) VALUES (NULL, '";
                        $sql = $sql . $unidad;
                        $sql = $sql . "')";
                        $result = mysqli_query($conn, $sql);
                        if($result == FALSE){
                                $msg["error"] = true;
                                $msg["message"] = "Error insertando la unidad";
                                $newResponse = $response->withJson($msg);
                                $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                                return $newResponse;
                        }
                }
                // Obtenemos el id de la unidad que acabamos de crear
```

```
                $sql = "SELECT `id_unit` FROM `unidad` WHERE `descripcion` = '";
                $sql = $sql . $unidad;
                $sql = $sql . "'";
                $result = mysqli_query($conn, $sql);
                if ($row = mysqli_fetch_assoc($result)){
                        $unidad = $row["id_unit"];
                }
        }

        // Actualizamos el array de longitudes de las tramas afectadas
        // SELECT `id_trama`, `array_id_sensor`, `array_longitud` FROM `trama` WHERE `id_trama` IN (SELECT `nodo`.`_id_trama` FROM `nodo` JOIN `canal` ON `nodo`.`id_nodo` = `canal`.`_id_nodo` WHERE `canal`.`_id_type` = '3' GROUP BY `_id_trama`)
        $sql = "SELECT `id_trama`, `longitud`, `array_id_sensor`, `array_longitud` FROM `trama` WHERE `id_trama` IN (SELECT `nodo`.`_id_trama` FROM `nodo` JOIN `canal` ON `nodo`.`id_nodo` = `canal`.`_id_nodo` WHERE `canal`.`_id_type` = '";
        $sql = $sql . $idtipo;
        $sql = $sql . "' GROUP BY `_id_trama`)";
        $result = mysqli_query($conn, $sql);

        $data = array();

        if (mysqli_num_rows($result) > 0) {
                while($row = mysqli_fetch_assoc($result)) {
                        $objeto = array('id' => $row["id_trama"], 'longitud_max' => $row["longitud"], 'id_sensor' => $row["array_id_sensor"], 'longitud' => $row["array_longitud"]);
                        array_push($data, $objeto);
                }
                for ($i = 0; $i < count($data); $i++){
                        $arr_id = explode(",", $data[$i]['id_sensor']);
                        $arr_long = explode(",", $data[$i]['longitud']);
                        $longitud_suma = $data[$i]['longitud_max'];
                        for ($j = 0; $j < count($arr_id); $j++){
                                if ($arr_id[$j] == $idtipo){
                                        $longitud_suma -= (int) $arr_long[$j];
                                        $longitud_suma += $longitud;
                                        $arr_long[$j] = $longitud;
                                }
                        }
                        $longitudes_sql = implode(",", $arr_long);
                        // UPDATE `trama` SET `longitud`='3', `array_longitud`='3' WHERE `id_trama`='3'
                        $sql = "UPDATE `trama` SET `longitud` = '";
                        $sql = $sql . $longitud_suma;
                        $sql = $sql . "', `array_longitud` = '";
                        $sql = $sql . $longitudes_sql;
                        $sql = $sql . "' WHERE `id_trama` = '";
                        $sql = $sql . $data[$i]['id'];
                        $sql = $sql . "'";
                        $result = mysqli_query($conn, $sql);
                }
        }

        // Mediante un UPDATE actualizamos los datos del sensor
        // UPDATE `tipo_sensor` SET `tipo` = 'x', `descripcion` = 'xxx', `formato` = '0.000', `longitud` = '3', `a` = '1', `b` = '2', `c` = '3', `_id_unit` = '2' WHERE `id_type` = '2';
        $sql = "UPDATE `tipo_sensor` SET `tipo` = '";
        $sql = $sql . $tipo;
        $sql = $sql . "', `descripcion` = '";
        $sql = $sql . $descripcion;
        $sql = $sql . "', `formato` = '";
        $sql = $sql . $formato;
        $sql = $sql . "', `longitud` = '";
        $sql = $sql . $longitud;
        $sql = $sql . "', `a` = '";
        $sql = $sql . $a;
        $sql = $sql . "', `b` = '";
        $sql = $sql . $b;
        $sql = $sql . "', `c` = '";
        $sql = $sql . $c;
        $sql = $sql . "', `_id_unit` = '";
        $sql = $sql . $unidad;
        $sql = $sql . "' WHERE `id_type` = '";
```

```
		$sql = $sql . $idtipo;
		$sql = $sql . "'";
		$result = mysqli_query($conn, $sql);
		if($result == TRUE) {
			$msg["error"] = false;
			$msg["message"] = "Sensor modificado correctamente";
			$newResponse = $response->withJson($msg);
			$newResponse = $newResponse->withHeader('Content-type', 'application/json');
			return $newResponse;
		} else {
			$msg["error"] = true;
			$msg["message"] = "Error modificando el sensor";
			$newResponse = $response->withJson($msg);
			$newResponse = $newResponse->withHeader('Content-type', 'application/json');
			return $newResponse;
		}
	}
	catch(PDOException $e)
	{
		$data["error"] = true;
	  $data["message"] = "Fallo de conexión: " . $e->getMessage();
	  $newResponse = $response->withJson($data);
	  $newResponse = $newResponse->withHeader('Content-type', 'application/json');
	  return $newResponse;
	}
	mysqli_close($conn);
});

/* Usando GET para borrar un sensor seleccionado */
// GET localhost/wsn/v1/borrarsensor/3
$app->get('/borrarsensor/{idtipo}', function(Request $request, Response $response) {

	// Obtenemos el id del sensor que queremos borrar
	$idtipo = $request->getAttribute('idtipo');

	// Nos conectamos a la base de datos
	$servername = "localhost";
	$username = "root";
	$password = “raspberry”;

	try
	{
		$dbname = "raspberrypi";
		$conn = mysqli_connect($servername, $username, $password, $dbname);
		// Comprobación de la conexión
		if (!$conn) {
			$data["error"] = true;
			$data["message"] = "Fallo de conexión: " . mysqli_connect_error();
			$newResponse = $response->withJson($data);
			$newResponse = $newResponse->withHeader('Content-type', 'application/json');
			return $newResponse;
		}
		// Actualizamos el array de longitudes e id de las tramas afectadas
		// SELECT `id_trama`, `array_id_sensor`, `array_longitud` FROM `trama` WHERE `id_trama` IN (SELECT
`nodo`.`_id_trama` FROM `nodo` JOIN `canal` ON `nodo`.`id_nodo` = `canal`.`_id_nodo` WHERE `canal`.`_id_type` = '3' GROUP BY
`_id_trama`)
		$sql = "SELECT `id_trama`, `longitud`, `array_id_sensor`, `array_longitud` FROM `trama` WHERE `id_trama` IN (SELECT
`nodo`.`_id_trama` FROM `nodo` JOIN `canal` ON `nodo`.`id_nodo` = `canal`.`_id_nodo` WHERE `canal`.`_id_type` = '";
		$sql = $sql . $idtipo;
		$sql = $sql . "' GROUP BY `_id_trama`)";
		$result = mysqli_query($conn, $sql);
		$data = array();
		if (mysqli_num_rows($result) > 0) {
			while($row = mysqli_fetch_assoc($result)) {
				$objeto = array('id' => $row["id_trama"], 'longitud_max' => $row["longitud"], 'id_sensor' =>
$row["array_id_sensor"], 'longitud' => $row["array_longitud"]);
				array_push($data, $objeto);
			}
			for ($i = 0; $i < count($data); $i++){
				$arr_id = explode(",", $data[$i]['id_sensor']);
				$arr_long = explode(",", $data[$i]['longitud']);
```

```
                                $longitud_suma = (int)  $data[$i]['longitud_max'];
                                $new_array_id = "";
                                $new_array_long = "";
                                $flag = 0;
                                for ($j = 0; $j < count($arr_id); $j++){
                                        if ($arr_id[$j] != $idtipo){
                                                if ($flag == 1) {
                                                        $new_array_id = $new_array_id . ",";
                                                        $new_array_long = $new_array_long . ",";
                                                } else {
                                                        $flag = 1;
                                                }
                                                $new_array_id = $new_array_id . $arr_id[$j];
                                                $new_array_long = $new_array_long . $arr_long[$j];
                                        } else {
                                                $longitud_suma -= (int) $arr_long[$j];
                                        }
                                }
                                $longitudes_sql = implode(",", $arr_long);
                                //    UPDATE  `trama`  SET  `longitud`='3',  `array_id_sensor`='3',  `array_longitud`='3'  WHERE
`id_trama`='3'
                                $sql = "UPDATE `trama` SET `longitud` = '";
                                $sql = $sql . $longitud_suma;
                                $sql = $sql . "', `array_id_sensor` = '";
                                $sql = $sql . $new_array_id;
                                $sql = $sql . "', `array_longitud` = '";
                                $sql = $sql . $new_array_long;
                                $sql = $sql . "' WHERE `id_trama` = '";
                                $sql = $sql . $data[$i]['id'];
                                $sql = $sql . "'";
                                $result = mysqli_query($conn, $sql);
                        }
                }

                // Procesar la query
                // DELETE FROM `tipo_sensor` WHERE `id_type` = '3';
                $sql = "DELETE FROM `tipo_sensor` WHERE `id_type` = '";
                $sql = $sql . $idtipo;
                $sql = $sql . "'";
                $result = mysqli_query($conn, $sql);
                if ($result == TRUE) {
                        $msg["error"] = false;
                        $msg["message"] = "Sensor borrado";
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                } else {
                        $msg["error"] = true;
                        $msg["message"] = "No se encuentra el sensor que quiere borrar";
                        $newResponse = $response->withJson($msg);
                        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
                        return $newResponse;
                }
        }
        catch(PDOException $e)
        {
                $data["error"] = true;
            $data["message"] = "Fallo de conexión: " . $e->getMessage();
            $newResponse = $response->withJson($data);
            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
            return $newResponse;
        }
        mysqli_close($conn);
});



                //**************
                //-  UNIDADES  -
                //**************
```

```
/* Usando GET para modificar una unidad seleccionada */
// GET localhost/wsn/v1/modificarunidad/3/kilos
$app->get('/modificarunidad/{idunit}/{descripcion}', function(Request $request, Response $response) {

  // Obtenemos el id de la unidad que queremos modificar y la nueva descripción
  $idunit = $request->getAttribute('idunit');
  $descripcion = $request->getAttribute('descripcion');

  // Nos conectamos a la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;

  try
  {
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {
              $data["error"] = true;
              $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
              $newResponse = $response->withJson($data);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        }
        // Procesar la query
        // UPDATE `unidad` SET `descripcion` = 'kilos' WHERE `id_unit` = '2';
        $sql = "UPDATE `unidad` SET `descripcion` = '";
        $sql = $sql . $descripcion;
        $sql = $sql . "' WHERE `id_unit` = '";
        $sql = $sql . $idunit;
        $sql = $sql . "'";
        $result = mysqli_query($conn, $sql);
        if ($result == TRUE) {
              $msg["error"] = false;
              $msg["message"] = "Unidad modificada";
              $newResponse = $response->withJson($msg);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        } else {
              $msg["error"] = true;
              $msg["message"] = "No se encuentra la unidad seleccionada";
              $newResponse = $response->withJson($msg);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        }
  }
  catch(PDOException $e)
  {
        $data["error"] = true;
    $data["message"] = "Fallo de conexión: " . $e->getMessage();
    $newResponse = $response->withJson($data);
    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
    return $newResponse;
  }
  mysqli_close($conn);
});

/* Usando GET para borrar una unidad seleccionada */
// GET localhost/wsn/v1/borrarunidad/3
$app->get('/borrarunidad/{idunit}', function(Request $request, Response $response) {

  // Obtenemos el id del sensor que queremos borrar
  $idunit = $request->getAttribute('idunit');

  // Nos conectamos a la base de datos
  $servername = "localhost";
  $username = "root";
  $password = “raspberry”;
```

```
  try
  {
        $dbname = "raspberrypi";
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Comprobación de la conexión
        if (!$conn) {
              $data["error"] = true;
              $data["message"] = "Fallo de conexión: " . mysqli_connect_error();
              $newResponse = $response->withJson($data);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        }
        // Procesamos la query para eliminar la unidad
        // DELETE FROM `unidad` WHERE `id_unit` = '3';
        $sql = "DELETE FROM `unidad` WHERE `id_unit` = '";
        $sql = $sql . $idunit;
        $sql = $sql . "'";
        $result = mysqli_query($conn, $sql);
        if ($result == TRUE) {
              $msg["error"] = false;
              $msg["message"] = "Unidad borrada";
              $newResponse = $response->withJson($msg);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        } else {
              $msg["error"] = true;
              $msg["message"] = "No se encuentran datos de la unidad a eliminar";
              $newResponse = $response->withJson($msg);
              $newResponse = $newResponse->withHeader('Content-type', 'application/json');
              return $newResponse;
        }
  }
  catch(PDOException $e)
  {
        $data["error"] = true;
    $data["message"] = "Fallo de conexión: " . $e->getMessage();
    $newResponse = $response->withJson($data);
    $newResponse = $newResponse->withHeader('Content-type', 'application/json');
    return $newResponse;
  }
  mysqli_close($conn);
});



        //**************
        //-   CSV    -
        //**************



//******************************************************
// POST localhost/wsn/v1/obtenercsv
// Body tipo raw (application/json)
//2017-09-24 08:50:19
//2017-09-25 07:27:08
/* {
        "fecha_ini": "Fecha inicio de búsqueda",
        "fecha_fin": "Fecha final de búsqueda",
        "array_nodos": "Nodos de los que se quiere obtener datos",
        "array_sensores": "Sensores que se quieren comprobar"
} */
//******************************************************
$app->post('/obtenercsv', function(Request $request, Response $response) {

  // Obtenemos los parámetros del body
  // Que vienen codificados en JSON
  $bodyPost = $request->getParsedBody();
  $fecha_ini = $bodyPost['fecha_ini'];
  $fecha_fin = $bodyPost['fecha_fin'];
```

```
$array_nodos = $bodyPost['array_nodos'];
$array_sensores = $bodyPost['array_sensores'];

// Parámetros de la base de datos
$servername = "localhost";
$username = "root";
$password = “raspberry”;

try
{
	$dbname = "raspberrypi";
	$conn = mysqli_connect($servername, $username, $password, $dbname);
	// Comprobación de la conexión
	if (!$conn) {  // Se devuelve lo siguiente si falla la conexión
		$data["error"] = true;
		$data["message"] = "Fallo de conexión: " . mysqli_connect_error();
		$newResponse = $response->withJson($data);
		$newResponse = $newResponse->withHeader('Content-type', 'application/json');
		return $newResponse;
	}

	$arr_nod = explode(",", $array_nodos);
	$arr_sen = explode(",", $array_sensores);
	$csv = array();
	$arr_cabez = array();
	for ($i = 0; $i < count($arr_sen); $i++) {
		for($j = 0; $j < count($arr_nod); $j++) {
			$idcanal = ((int) $arr_nod[$j]) * 100 + (int) $arr_sen[$i];
			// SELECT nodo.descripcion AS nodo, tipo_sensor.tipo, unidad.descripcion FROM (((nodo JOIN canal ON nodo.id_nodo=canal._id_nodo) JOIN tipo_sensor ON canal._id_type=tipo_sensor.id_type) JOIN unidad ON tipo_sensor._id_unit=unidad.id_unit) WHERE canal.id_chan = 101
			$sql = "SELECT nodo.descripcion AS nodo, tipo_sensor.tipo, unidad.descripcion FROM (((nodo JOIN canal ON nodo.id_nodo=canal._id_nodo) JOIN tipo_sensor ON canal._id_type=tipo_sensor.id_type) JOIN unidad ON tipo_sensor._id_unit=unidad.id_unit) WHERE canal.id_chan = '";
			$sql = $sql . $idcanal;
			$sql = $sql . "'";
			$result = mysqli_query($conn, $sql);
			$cabecera = array();
			if (mysqli_num_rows($result) > 0) {
				while($row = mysqli_fetch_assoc($result)) {
					$objeto = array('nodo' => $row["nodo"], 'sensor' => $row["tipo"], 'unidad' => $row["descripcion"]);
					array_push($cabecera, $objeto);
				}
			}
			$titulo = $cabecera[0]['nodo'] . " " . $cabecera[0]['sensor'] . " (" . $cabecera[0]['unidad'] . ")";
			array_push($arr_cabez, $titulo);
			if (($i == 0)AND($j == 0)) {
				// CREATE TABLE datos AS SELECT dia_semana AS Dia, lectura_nodo AS Fecha, valor AS 'PM25 Nddo 1 (kPa)' FROM dato_canal WHERE (_id_chan = 101 AND lectura_nodo BETWEEN '2017-10-05 17:15:00' AND '2017-10-05 17:30:00') ORDER BY lectura_nodo
				$sql = "CREATE TABLE datos AS SELECT dia_semana AS Dia, lectura_nodo AS Fecha, valor AS '";
				$sql = $sql . $titulo;
				$sql = $sql . "' FROM dato_canal WHERE (_id_chan = '";
				$sql = $sql . $idcanal;
				$sql = $sql . "' AND lectura_nodo BETWEEN '";
				$sql = $sql . $fecha_ini;
				$sql = $sql . "' AND '";
				$sql = $sql . $fecha_fin;
				$sql = $sql . "') ORDER BY lectura_nodo";
				mysqli_query($conn, $sql);
			} else {
				$sql = "ALTER TABLE datos ADD `";
				$sql = $sql . $titulo;
				$sql = $sql . "` float";
				mysqli_query($conn, $sql);
				$sql = "SELECT dia_semana, lectura_nodo, valor FROM dato_canal WHERE (_id_chan = '";
				$sql = $sql . $idcanal;
				$sql = $sql . "' AND lectura_nodo BETWEEN '";
```

```
                                        $sql = $sql . $fecha_ini;
                                        $sql = $sql . "' AND '";
                                        $sql = $sql . $fecha_fin;
                                        $sql = $sql . "') ORDER BY lectura_nodo";
                                        $result = mysqli_query($conn, $sql);
                                        if (mysqli_num_rows($result) > 0) {
                                                while($row = mysqli_fetch_assoc($result)) {
                                                        $sql = "UPDATE datos SET `";
                                                        $sql = $sql . $titulo;
                                                        $sql = $sql . "` = '";
                                                        $sql = $sql . $row["valor"];
                                                        $sql = $sql . "' WHERE Fecha = '";
                                                        $sql = $sql . $row["lectura_nodo"];
                                                        $sql = $sql . "'";
                                                        mysqli_query($conn, $sql);
                                                        if (mysqli_affected_rows($conn) == 0) {
                                                                // INSERT INTO `datos`(`id_data`, `dia_semana`,
`lectura_nodo`, `lectura_rbpi`, `valor`, `_id_chan`)
                                                                $sql = "INSERT INTO `datos` (`Dia`, `Fecha`";
                                                                for ($x = 0; $x < count($arr_cabez); $x++){
                                                                        $sql = $sql . ", `";
                                                                        $sql = $sql . $arr_cabez[$x];
                                                                        $sql = $sql . "`";
                                                                }
                                                                $sql = $sql . ") VALUES ('";
                                                                $sql = $sql . $row["dia_semana"];
                                                                $sql = $sql . "', '";
                                                                $sql = $sql . $row["lectura_nodo"];
                                                                $sql = $sql . "', ";
                                                                for ($y=0;$y<(count($arr_cabez)-1);$y++) {
                                                                        $sql = $sql . "'', ";
                                                                }
                                                                $sql = $sql . "'";
                                                                $sql = $sql . $row["valor"];
                                                                $sql = $sql . "')";
                                                                $actualiza = mysqli_query($conn, $sql);
                                                        }
                                                }
                                        }
                                }
                        }
                }

                // SELECT * FROM `datos` WHERE 1 ORDER BY `Fecha`
                $sql = "SELECT * FROM `datos` WHERE 1 ORDER BY `Fecha`";
                $result = mysqli_query($conn, $sql);
                if (mysqli_num_rows($result) > 0) {
                        while($row = mysqli_fetch_assoc($result)) {
                                // $valores = array('Dia' => $row["Dia"], 'Fecha' => $row["Fecha"]);
                                $columnas = array('Dia', 'Fecha');
                                $valores = array($row["Dia"], $row["Fecha"]);
                                for ($k = 0; $k < count($arr_cabez); $k++){
                                        array_push($columnas, $arr_cabez[$k]);
                                        array_push($valores, $row[$arr_cabez[$k]]);
                                }
                                $objeto = array_combine($columnas, $valores);
                                array_push($csv, $objeto);
                        }
                }
                // DROP TABLE `datos`
                $sql = "DROP TABLE `datos`";
                mysqli_query($conn, $sql);

                $data["error"] = false;
            $data["message"] = $csv;
                $newResponse = $response->withJson($data);
            $newResponse = $newResponse->withHeader('Content-type', 'application/json');
            return $newResponse;
        }
        catch(PDOException $e)
        {
```

```
            $data["error"] = true;
        $data["message"] = "Fallo de conexión: " . $e->getMessage();
        $newResponse = $response->withJson($data);
        $newResponse = $newResponse->withHeader('Content-type', 'application/json');
        return $newResponse;
    }
    mysqli_close($conn);
});

/* Lanzamos la aplicación */
$app->run();

?>
```