UNIVERSIDAD INTERNACIONAL DE LA RIOJA

Universidad Internacional de La Rioja
Máster universitario en Ingeniería de Software y Sistemas Informáticos

Desarrollo de una aplicación web cloud para detectar CO en domicilios

Trabajo Fin de Máster

Tipo de trabajo: Desarrollo práctico

Presentado por: Romero Pazmiño, Santiago

Director/a: Escuin Finol, David

Ciudad: Quito, Ecuador.
Fecha: 18/09/2018

Resumen

El desarrollo de sistemas que involucran Tecnologías de la Información y Comunicación (TIC) está ligado cada vez más al uso de software y hardware libre que en conjunto permiten la integración y reutilización de componentes que se prueban amplia y continuamente en un afán de mejora.

En los últimos años el campo de Internet de las Cosas (IoT) ha experimentado una evolución tanto en el hardware como en software, siendo este último elemento el que sigue marcando la tendencia actual al permitir diseñar e implementar aplicaciones. También, el hardware libre no se ha estancado debido a que a más de reducir significativamente los costos, ofrece plataformas de desarrollo que hacen más sencilla la investigación en diferentes niveles, incluyendo prácticas de laboratorio y sistemas complejos en producción.

El objetivo del presente trabajo es implementar un sistema de detección y registro de niveles de CO de bajo costo en domicilios usando hardware y software libre, tecnologías web y servicios cloud que sea capaz de salvaguardar vidas humanas.

Para alcanzar este objetivo, se conectará el sensor de CO al dispositivo Arduino, el cual convertirá las señales analógicas en digitales y enviará estos datos al Raspberry Pi mediante una conexión serial. Este dispositivo utilizará un servicio web para enviar la lectura ADC, la medición de voltaje, el estado de detección del gas, la cantidad de partículas por millón del CO y la cantidad de mg/m3 del CO a una base de datos de la aplicación web desplegada en la nube.

A partir de los datos obtenidos desde el sensor, se realizará un análisis e interpretación de dicha información. Estos resultados permitirán realizar varias gráficas, obteniendo así una visión más clara del comportamiento en las mediciones registradas.

Palabras Clave: Tecnologías de la Información y Comunicación (TIC), Internet de las Cosas (IoT), Arduino, Raspberry Pi.

Abstract

The development of systems that involve Information and Communication Technologies (ICT) is increasingly linked to the use of free software and free hardware that together allow the integration and reuse of the components that are tested widely and continuously in an effort to improve.

In recent years the field of Internet of Things (IoT) has experienced an evolution in both hardware and software, with software being the element that continues to set the current trend by allowing design and implementation of applications. Also, free hardware has not stagnated because, apart from significantly reducing costs, it offers development platforms that make research at different levels easier, including laboratory practices and complex systems in production.

The objective of this work is to implement a low cost system of detection and recording CO levels in homes using free hardware and free software, web technologies and cloud services that is capable of safeguarding human lives.

To achieve this goal, the CO sensor will be connected to the Arduino device, which will convert the analog signals into digital and send this data to the Raspberry Pi through a serial connection. This device will use a web service to send the ADC reading, the voltage measurement, the gas detection status, the number of particles per million of the CO and the amount of mg / m3 of the CO to a database of the web application deployed in the cloud.

Based on the data obtained from the sensor, an analysis and interpretation of that information will be performed. These results will allow several graphs to be made, obtaining a clearer vision of the behavior in the recorded measurements.

Keywords: Information and Communication Technologies (ICT), Internet of Things (IoT), Arduino, Raspberry Pi.

Índice de contenidos

Índice de tablas

Índice de figuras

1. Introducción

La calidad del aire en las viviendas es importante para la salud de las personas. Existen gases tóxicos que no pueden ser percibidos fácilmente y de manera oportuna por el ser humano provocando daños severos e irreparables en el organismo.

Por este motivo el desarrollo e implementación de un sistema que permita la temprana detección y medición de CO en el entorno, tiene como finalidad precautelar la salud de los ocupantes de un domicilio, evitando que estén expuestos al CO y ejecutar acciones oportunas en cuanto se realice la detección del gas.

Este trabajo fin de máster trata del diseño y programación de una aplicación web orientada a la detección y medición de monóxido de carbono CO en domicilios priorizando el empleo de tecnologías de hardware y software libre en su gran mayoría, por lo que se utilizará un dispositivo de tamaño reducido como el Raspberry Pi con sistema operativo Raspbian (distribución basada en Debian), una placa de desarrollo denominada Arduino, un entorno open source de ejecución para JavaScript en el servidor como Node.js y el framework Express.js para el soporte de aplicaciones web, un framework de JavaScript de código abierto para trabajar en el cliente como Angular y un sistema gestor de base de datos NoSQL de código abierto como MongoDB.

Se desarrollará código en el IDE de Arduino para procesar la señal del sensor de CO y enviar estos datos al Raspberry Pi de forma serial. También se implementará código en Python para recibir los datos seriales y consumir un servicio web haciendo posible que esta información se almacene en un SGBD.

Este SGBD, el servicio desplegado en Node.js y la sección de código desarrollada usando Angular residirán en la nube permitiendo que se implemente una aplicación web robusta con tecnología JavaScript, que a la vez sea segura y cuya fuente de datos sean los datos del sensor de CO.

Dependiendo de la cantidad de CO que el cuerpo humano puede tolerar y que no afecta de forma permanente a la salud, se configurará un mensaje de alerta o emergencia que será enviado vía e-mail a la persona responsable del domicilio.

El diseño del sistema, la configuración de cada uno de sus elementos y la implementación del código necesario se describen en los siguientes capítulos.

1.1 Justificación

Debido al mayor interés en la construcción de edificios energéticamente eficientes, los hogares y oficinas ahora son más compactos, lo que ha aumentado el potencial para la acumulación de CO y ha convertido a este gas en una amenaza constante para todas estas

instalaciones enviando cada año a 20000 personas a urgencias con envenenamiento por CO.

Es por esta razón que es esencial precautelar el estado de salud de las personas y salvar vidas humanas mediante la detección y medición del gas tóxico monóxido de carbono de una forma económica, sin hacer uso de ningún software propietario de IoT y que permita la visualización de la información recogida en una aplicación web.

Existen en el mercado una gran variedad de tipos de sensores de CO como: térmicos, infrarrojos, por desplazamiento, distancia y posición, cuyos costos oscilan desde unidades hasta centenas de dólares, siendo los más económicos los del tipo térmico. Además, estos sensores ofrecen un amplio rango de detección, característica que es útil para aplicaciones como la planteada en este proyecto.

Según los expertos, se estima que la industria de IoT alcance los 1.2 a 14.4 trillones de dólares para 2025 opacando a Internet y a los dispositivos móviles, razón por la que es importante analizar esta tecnología.

Según [1], IoT promueve un mayor nivel de conciencia sobre el mundo y una plataforma desde la cual monitorear las reacciones a las condiciones cambiantes a las que está expuesta la humanidad. El enfoque de IoT permite cubrir aplicaciones que van desde lo micro a lo macro, y desde lo trivial a lo crítico como es el caso de los gases tóxicos.

Dentro de los casos críticos se puede mencionar que IoT contribuye a una gestión más inteligente y eficaz frente a desastres naturales (deslaves, terremotos, avalanchas, etc.), ya que hace posible predecir de forma más precisa la aparición de condiciones que pueden desencadenar incendios forestales o hacer que estos se vuelvan incontrolables a la vez que permite a los equipos de contención actuar de forma oportuna y realizar evacuaciones dirigidas.

Otro campo de aplicación es la automatización de la gestión del tráfico en función de las condiciones cambiantes; idea que se ha usado para el desarrollo de aplicaciones de estacionamiento que guían inteligentemente a los conductores a fin de encontrar espacios libres, reduciendo el tiempo empleado en la búsqueda a la vez que se disminuye drásticamente las emisiones de los vehículos.

Quizás el entorno más importante de IoT es el relacionado al cuidado de la salud de las personas, en el que dispositivos portátiles son capaces de detectar una serie de problemas, incluso antes de que ocurran quebrantos o afectaciones al cuerpo humano, e inmediatamente administren medicamentos que salven vidas o implementen servicios de emergencia-alertas con información detallada de la situación actual de peligro.

Este proyecto de diseño e implementación de un sistema de bajo costo para la detección y medición del gas tóxico monóxido de carbono haciendo uso de hardware y software libre en su gran mayoría está alineado con la visión de IoT a fin de conseguir un planeta mucho más

"inteligente", en el que se garantice la seguridad de las personas y se contribuya a la mejora de las posibilidades de la humanidad de dejar un legado sostenible para las generaciones futuras.

1.2 Planteamiento del trabajo

De acuerdo a [2], existe en el mercado una gran variedad de sistemas comerciales costosos de detección de humo y monóxido de carbono como Nest Project, que cuando detectan la presencia del gas realizan una serie de acciones como activar una alarma local o enviar notificaciones al celular, pero un control eficaz radica en poder medir la cantidad de partículas por millón (PPM) del gas en el medio ambiente a fin de poder realizar todas estas acciones reactivas de forma oportuna y en base a valores de CO que afectan notoriamente al organismo humano a la vez que se evitan falsas alarmas. Este tipo de sistemas de detección y medición de gases tóxicos también están disponibles en el mercado, pero su costo es muy elevado y la tecnología incluyendo el software presenta restricciones y se limita al usuario a la configuración de las características que ofrece la compañía desarrolladora sin ninguna posibilidad de inclusión de nuevas funcionalidades y comunicación con sistemas externos.

Para solucionar estas limitaciones en cuanto a la medición y detección de gases tóxicos en domicilios se plantea un sistema innovador de bajo costo en comparación a otros equipos especializados en la medición de monóxido de carbono, que utilice un sensor de gas MQ-7, dispositivos de hardware libre, software de código abierto y que además brinde a los usuarios movilidad en cuanto al acceso e interacción con la aplicación web que recogerá las mediciones, procesará estos datos y enviará notificaciones a los usuarios dependiendo del valor de CO detectado.

En cuanto a hardware libre se empleará la plataforma de desarrollo Arduino y un Raspberry Pi, respecto a software libre se desplegará la aplicación desarrollada en JavaScript en el entorno de ejecución open source Node.js utilizando los frameworks Express.js y Angular, junto al sistema gestor de base de datos NoSQL de código abierto MongoDB.

El trabajo propuesto a más de lo descrito anteriormente permite que los datos recibidos por parte del sensor sean analizados y visualizados en gráficas a fin de que se pueda determinar el comportamiento de las mediciones del gas.

Uno de los objetivos que se pretende alcanzar con este proyecto es diseñar e implementar un sistema de detección y registro de niveles de CO de bajo costo en domicilios usando en su mayoría hardware y software libre, tecnologías web y servicios cloud de forma que se pueda salvaguardar vidas humanas mediante el envío de notificaciones a los usuarios de la aplicación y del sistema.

1.3 Estructura de la memoria

Este trabajo consta de seis capítulos que detallan el desarrollo de una aplicación web cloud para detectar y registrar niveles de CO en domicilios.

En el primer capítulo se realiza una breve introducción, indicando cuál es la justificación del trabajo y la planificación a seguir para poder cumplir los objetivos y cubrir todas las necesidades.

El segundo capítulo consta de un marco teórico que hace referencia a conceptos como IoT, Node.js, Express.js, Angular, MongoDB y Python. También se describe el contexto y estado del arte relacionado a los sistemas de detección y medición de gases tóxicos, resaltando los siguientes trabajos:

- Monitoreo Ambiental Inteligente basado en IOT usando Arduino.
- Monitoreo del Campus de la Universidad de Taif utilizando técnicas de Internet de las Cosas.
- Diseño e implementación de un dispositivo IoT de bajo coste para entornos agrícolas.
- Sistema de monitoreo de la calidad del aire basado en IoT usando Raspberry Pi.
- Sistema de monitoreo de la calidad del aire (AQMS) usando Raspberry Pi e IoT.

Es importante mencionar que ninguno de los trabajos indicados anteriormente considera de manera integral los cuatro aspectos fundamentales en los que se basa el desarrollo del TFM: hardware libre, tecnologías web y cloud, software libre y medición de magnitudes ambientales.

En el tercer capítulo se detallan los objetivos concretos y la metodología de trabajo a seguir para diseñar e implementar un sistema de bajo costo para la detección y medición del gas tóxico monóxido de carbono en domicilios haciendo uso de hardware y software libre, tecnologías web y servicios cloud de forma que se pueda salvaguardar vidas humanas.

En el cuarto capítulo se describirá el proceso de desarrollo empleado para implementar correctamente la aplicación web, el cual consta de tres partes fundamentales:

- Identificación de requisitos funcionales y no funcionales.
- Descripción del sistema desarrollado.
- Evaluación.

En el quinto capítulo, se analizan los resultados obtenidos y se determinan si se han cumplido con todos los objetivos planteados. Además, se detallan posibles aplicaciones de un trabajo futuro que tome como base el presente proyecto.

Por último, en el sexto capítulo se detallan todas las referencias bibliografías empleadas haciendo énfasis en aquellas utilizadas en el capítulo del Contexto y estado del arte.

2. Contexto y estado del arte

2.1. Introducción al tema

Según [3], la fuga de gases y la detección de estos es un grave problema en la vida diaria ya que la mayoría de ellos son difíciles de percibir, altamente inflamables y pueden provocar daños a la salud de las personas y a la infraestructura. Para reducir el riesgo de que se produzcan tales situaciones, la ciencia y tecnología han realizado un gran esfuerzo desarrollando técnicas confiables, dispositivos y elementos que permiten la detección de fugas de gas. El conocer la existencia de una fuga no siempre es suficiente para iniciar una acción correctiva por lo que es imprescindible realizar correctamente la medición de la cantidad de gas en el ambiente.

De acuerdo a [4], la fuga de gases tóxicos es la razón principal de los accidentes industriales y muertes de los trabajadores. Los contaminantes emitidos por las industrias a la atmósfera también son una causa de la contaminación ambiental y afectan en gran medida la salud de las personas y animales al reducir los niveles de oxígeno y aumentar los niveles de gases nocivos como el amoníaco, monóxido de carbono, etc.

La población al depender más del uso de petróleo, gas y carbón para generar energía y satisfacer la demanda de un mercado en continuo crecimiento, provoca que la liberación de contaminantes nocivos aumente día a día por parte de la industria.

Alrededor de 1.1 billones de personas respiran aire contaminado y se han registrado 7 millones de muertes en todo el mundo, razón por la cual los gases tóxicos y sus efectos tienen consecuencias terribles obligando al desarrollo de sistemas innovadores de detección.

Las fugas de gases y su detección se pueden manejar de manera efectiva mediante el uso de sensores y la automatización con IoT, siendo un ejemplo claro de estos sistemas los proyectos que se describen en el siguiente apartado.

Es importante mencionar que Internet of Things (IoT) es un paradigma que considera la presencia de una variedad de sensores, equipos y objetos en cualquier entorno, en el cual mediante conexiones inalámbricas o cableadas permiten la interacción entre los elementos del sistema con el fin de crear aplicaciones o servicios. Las aplicaciones más comunes son las de monitoreo ambiental, cuyo objetivo es utilizar sensores para colaborar en la protección ambiental monitoreando parámetros como la temperatura, humedad y la calidad del aire. Este trabajo fin de máster implementa un sistema de bajo costo cuyo fin no solo es

monitorear la calidad del aire debido a la presencia del gas monóxido de carbono sino que mediante la detección y medición de este gas tóxico permite salvaguardar la vida de las personas enviando notificaciones a los usuarios cuando el nivel del gas puede provocar daños en el organismo. Además, el sistema ofrece una aplicación web cloud donde se observa mediante gráficos las mediciones registradas haciendo que el análisis de datos y comportamiento de las mediciones sea más fácil de realizar a la vez que el usuario puede acceder a la información en tiempo real y desde cualquier ubicación.

2.2. Marco Conceptual

2.2.1 IoT

Según [5], IoT (Internet of Things) es la conexión de objetos materiales a Internet permitiendo la transmisión de datos de forma remota a través de la configuración de capacidades de captura y procesamiento de datos ambientales. Una vez que se ha establecido la conexión, cada objeto adquiere una dirección de red que lo hace identificable de forma única. Estos objetos generalmente poseen la capacidad de detección en alguna de sus capas que les permite registrar dinámicamente los cambios en el entorno y transmitir esa información a través de Internet.

Para que un dispositivo sea considerado una “cosa u objeto” en IoT se deben cumplir las siguientes características esenciales:

- El dispositivo debe ser capaz de recopilar y transmitir datos a otros dispositivos o directamente a Internet.
- El dispositivo debe tener la capacidad de responder frente a acciones determinadas.
- El dispositivo debe tener la capacidad de recibir información de la red.

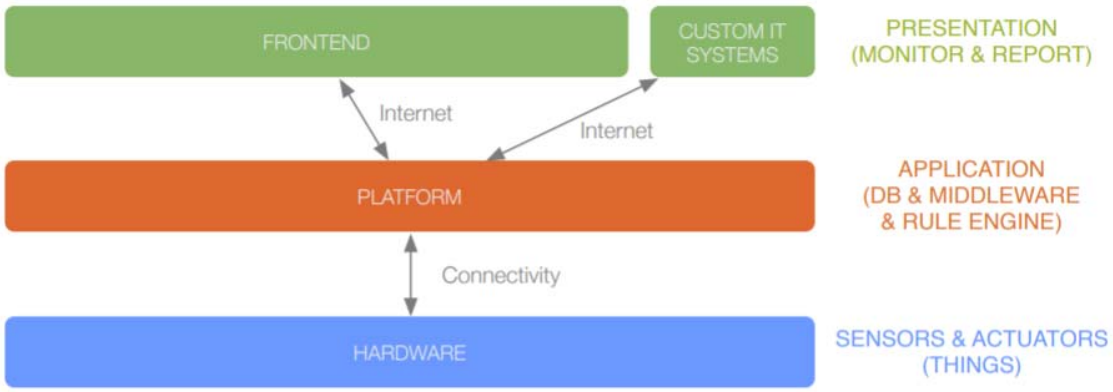


Figura 1. Capas de la Arquitectura de IoT [6].

De acuerdo a [5], [6] y como se puede observar en la Figura 1, existen básicamente tres capas que conforman la arquitectura de IoT. La primera capa se conoce como capa física, en la cual las etiquetas RFID (Radio Frequency Identification), sensores y actuadores se utilizan. Estos dispositivos operan en un entorno determinado para recopilar datos y actuar de acuerdo a las condiciones de este medio.

La segunda capa es la denominada capa de red, conformada por redes celulares pequeñas, redes de área local (LAN), etc. Para que se puedan explotar todos los beneficios de estos elementos, los sistemas de IoT necesitan utilizar un protocolo de comunicaciones común a través del middleware (software que permite la comunicación y gestión de datos) a fin de que la interpretación de los datos se pueda efectuar y desplegar dentro de las aplicaciones de IoT.

La última capa es la capa de aplicación que comprende las aplicaciones o la tecnología de comunicación digital que el usuario observa e interactúa, siendo el ejemplo más conocido Google Home.

IoT es una tecnología que se aplica en la mayoría de escenarios como: el control de la calidad del aire; monitoreo de bosques, ríos, lagos y océanos; conservación de la vida silvestre; implementación de redes eléctricas inteligentes; prevención y mitigación de los efectos de desastres naturales; gestión eficaz de residuos. Por estas razones y debido a que la información obtenida gracias al uso de IoT permite a las personas tomar decisiones en la vida diaria e incluso automatizar áreas particulares de la misma, se ha decidido incluir esta tecnología en el presente trabajo.

2.2.2 Node.js

De acuerdo a [7], se puede mencionar que Node.js es un entorno en tiempo de ejecución multiplataforma de código abierto para el desarrollo de aplicaciones en el lado del servidor construido sobre el motor de JavaScript de Google Chrome. Las aplicaciones de Node.js se escriben en JavaScript y se pueden ejecutar en Mac OS X, Windows y Linux. Cuenta con una amplia librería de módulos JavaScript que simplifica el desarrollo de aplicaciones web.

Toda solicitud de archivo en Node.js se maneja forma asincrónica, es decir:

- La tarea se envía al sistema de archivos del computador.
- Node.js está listo para las próximas solicitudes.

- Cuando el sistema de archivos se ha abierto y lee el archivo, el servidor devuelve el contenido al cliente.

Las principales áreas donde Node.js se aplica ampliamente son: streaming de datos, aplicaciones web, chats, proxies, tableros de mando. A su vez, las empresas, aplicaciones y proyectos más destacados que usan Node.js son: eBay, General Electric, GoDaddy, Microsoft, PayPal, Uber, Netflix, LinkedIn.

En este trabajo se ha incluido esta tecnología ya que según [8], Node.js ofrece las siguientes ventajas:

- JavaScript es simple de aprender, por lo que Node.js es adecuado para desarrolladores con poca experiencia.
- Debido a las tecnologías innovadoras de Google como el uso de mecanismos de eventos y la imposibilidad de almacenar datos en búfer, Node.js es extremadamente rápido.
- Es altamente escalable ya que el mecanismo de eventos y el modelo de subproceso único ayudan al servidor a responder de una manera no bloqueante y poder manejar más peticiones.
- Es extensible debido a que la comunidad de soporte produce varios módulos para agregar capacidades adicionales a las aplicaciones de Node.js.
- Permite ahorrar tiempo y dinero, ya que al usar JavaScript para desarrollar aplicaciones tanto del lado del servidor como del cliente, no es necesario contratar desarrolladores por separado para el Back-End y el Front-End.

2.2.3 Express.js

Según [9], Express es el framework del lado servidor para el desarrollo web más popular alojado en el entorno en tiempo de ejecución Node.js. Está escrito en JavaScript y es a su vez es la librería que se usa como base para otros frameworks web de Node.js. Proporciona mecanismos para:

- Construir manejadores de peticiones con diferentes métodos HTTP (GET, POST, DELETE, etc.) para diferentes rutas URL.
- Definir la configuración común de aplicaciones web, estableciendo el puerto que se utilizará para la conexión y la ubicación de las plantillas que se utilizan para representar la respuesta.
- Agregar el "middleware" adicional de procesamiento de peticiones en cualquier punto dentro del esquema de manejo de peticiones.
- Construir APIs (Interfaces de programación de aplicaciones).

Express.js es multiplataforma y compatible con el patrón de diseño Modelo-Vista-Controlador. Los sitios web más importante que fueron creados usando Express.js son: IBM, Wal-Mart, Target, Mozilla, eHow, Flickr, Fandango, The New York Times.

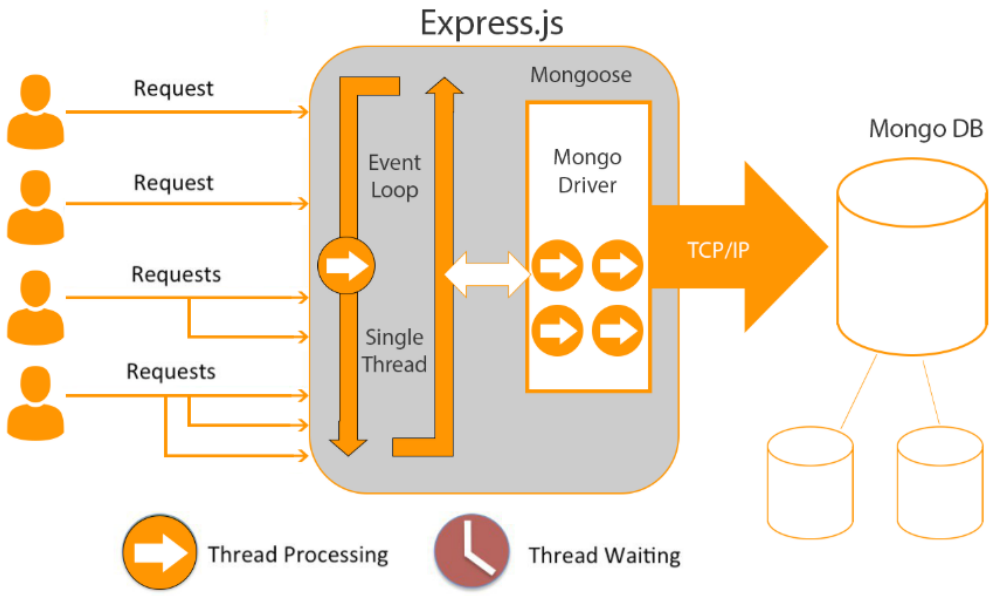


Figura 2. Funcionamiento de Express.js [10].

De acuerdo a [10] y analizando la forma de trabajar de Express.js representada en la Figura 2, se puede expresar:

- El servidor recibe la petición y registra el procesamiento de datos del evento. Luego, el evento envía la petición al servidor Mongo DB y es necesario que transcurra cierto tiempo para que los datos solicitados sean devueltos. De forma simultánea, el servidor está procesando otras solicitudes. Cuando arriban los datos al servidor, Mongoose invoca al manejador de pedidos del usuario (API), el cual entrega los datos al cliente.

Se ha decidido incluir este framework en el presente trabajo ya que según [10], algunas de las ventajas más destacadas que ofrece Express.js son:

- Manejo de una gran cantidad de peticiones de los usuarios ya que Node.js junto con Express.js son capaces de soportar miles de acciones simultáneas.
- Express.js cuenta con el soporte necesario por parte de una comunidad open source, la cual revisa y mejora el código continuamente.
- Fácil integración de servicios de terceros a través del desarrollo de paquetes de middleware útiles para resolver cualquier problema de desarrollo web relacionado a

cookies, sesiones, inicio de sesión de usuarios, parámetros URL, datos POST, encabezados de seguridad y más.

- Fácil de aprender ya que está bien documentado y emplea JavaScript.

2.2.4 Angular

Tomando en cuenta [11], Angular es un framework estructural de JavaScript para aplicaciones web dinámicas que permite usar HTML como lenguaje de plantilla y extender la sintaxis de HTML para expresar los componentes que requiera la aplicación de forma clara y concisa.

Una de las características fundamentales de Angular es el enlace de datos que permite la sincronización automática de los datos entre los componentes de la vista y el modelo. Otra característica importante es la inyección de dependencia que ayuda al programador con el hecho de que la aplicación sea más fácil de desarrollar, comprender y probar. Ambas características a más de reducir significativamente las líneas del código a escribir han sido razones a considerar para que se estudie esta tecnología y se utilice de forma práctica en este trabajo. Angular es el socio ideal en el Front-End para cualquier tipo de tecnología del lado servidor.

Angular viene incorporado con los siguientes elementos:

- Herramientas para la construcción de una aplicación CRUD en un paquete que incluye directivas básicas de plantillas, validación de formularios, enrutamiento, componentes reutilizables y más.
- Herramientas para realizar pruebas unitarias, pruebas extremo a extremo, etc.

También se ha incluido este framework en el presente proyecto ya que de acuerdo a [12], las principales ventajas de Angular son:

- Es un framework de código abierto, completamente gratuito y utilizado por miles de desarrolladores de todo el mundo bajo la licencia de Apache 2.0 que mantiene su compatibilidad con Google.
- Angular maneja automáticamente el código JavaScript adecuado para cada navegador web.
- Permite construir aplicaciones web de gran escala y alto rendimiento a la vez que son fáciles de mantener y se tiene que escribir menos código.
- Enlace de datos bidireccional que permite sincronizar el Modelo y la Vista, es decir que a medida que los datos cambian en un componente, también lo hace en el otro, permitiendo reducir el tiempo de desarrollo.

- Las directivas extienden la funcionalidad de HTML y dan la capacidad suficiente a los desarrolladores para asignar comportamientos especiales al DOM, creando de esta forma contenido dinámico con HTML.
- Angular cuenta con el soporte de una comunidad open source que proporciona materiales de capacitación, foros y herramientas de terceros que permiten encontrar una solución a casi todos los problemas que surjan al momento de desarrollar software.
- Arquitectura basada en componentes que proporciona una mayor calidad de código y favorece la reutilización, legibilidad, implementación de pruebas unitarias en un entorno amigable y la mantenibilidad.
- TypeScript que se compila totalmente en JavaScript, es el lenguaje base para Angular ya que permite obtener un código más limpio y mayor escalabilidad al detectar y eliminar errores comunes por parte del desarrollador.

2.2.6 MongoDB

De acuerdo a [13], MongoDB es una base de datos open source, multiplataforma, NoSQL con un modelamiento orientado a documentos que ofrece alto rendimiento, alta disponibilidad y escalamiento automático. Un registro en MongoDB es un documento, que es una estructura de datos compuesta por pares (clave: valor). Los documentos de MongoDB son similares a los objetos JSON. Los valores de las claves pueden incluir otros documentos, arreglos y arreglos de documentos.

La arquitectura de MongoDB se basa en los siguientes elementos:

- Base de datos: Es el contenedor físico de colecciones. Cada una de las bases de datos tiene su propio conjunto de archivos en el sistema de archivos. Un solo servidor MongoDB tiene múltiples bases de datos.
- Colección: Es un grupo de documentos de MongoDB, equivalente a una tabla RDBMS (Sistema de gestión de base de datos relacionales). Una colección existe dentro de una única base de datos. No hay esquemas cuando se trata de colecciones. Dentro de una colección, los diversos documentos pueden tener campos variados, pero la mayoría de los documentos dentro de una colección tienen un propósito similar o relacionado.
- Documento: Es un conjunto de pares clave-valor. Los documentos están asociados con esquemas dinámicos, es decir, los documentos en una misma colección no necesitan tener la misma estructura o campos, y los campos comunes en los documentos de una colección pueden tener varios tipos de datos.

Se ha considerado el empleo de este tipo de tecnología en el presente trabajo ya que según [14], las principales ventajas de MongoDB son:

- Proporciona persistencia de datos de alto rendimiento ya que al contar con un soporte para modelo de datos se reduce la actividad de entrada y salida en el sistema de base de datos.
- Compatibilidad con múltiples motores de almacenamiento.
- MongoDB al ser una base de datos sin esquema otorga flexibilidad y libertad para almacenar datos de diferentes tipos.
- Se puede almacenar una gran cantidad de datos mediante la distribución de la información en varios servidores conectados a la aplicación, logrando de esta forma que MongoDB sea escalable horizontalmente y pueda balancear la carga.
- Mediante el indexado es fácil acceder a los documentos y se proporciona una respuesta rápida. La velocidad de MongoDB es 100 veces más rápida que la de una base de datos relacional.
- MongoDB tiene características como replicación que ayudan a aumentar la disponibilidad de datos y por consiguiente el rendimiento es muy alto.
- Se pueden realizar consultas en base al campo, rango y también es posible usar expresiones regulares.
- Soporta el modelamiento de datos JSON con esquemas dinámicos.

2.2.7 Python

De acuerdo [15], Python es un lenguaje de alto nivel, interpretado, orientado a objetos y dinámico ya que realiza una verificación de tipo en tiempo de ejecución a fin de que el tipo de dato configurado coincida con la información esperada.

Respecto a los frameworks disponibles en Python se puede destacar a Django, que es el más común, de código libre y abierto, y que permite la creación de sitios web dirigidos por bases de datos. Sitios web como Instagram y Mozilla hacen uso de este framework.

Los componentes de Python son los siguientes:

- Una función es un conjunto de sentencias declaradas bajo un nombre en particular que pueden ser llamadas en cualquier parte del programa. Puede devolver un valor.
- Una clase es un tipo de datos abstractos. Python al ser un lenguaje orientado a objetos es compatible con el uso de clases y objetos.
- Un módulo Python es una colección de clases y funciones relacionadas. Los principales módulos están orientados a cálculos matemáticos, manipulación de cadenas, programación web, etc.
- Un paquete es un conjunto de módulos.

Se ha empleado este tipo de lenguaje en una parte de este trabajo ya que de acuerdo a [15] y [16], las características más sobresalientes de Python son:

- Facilidad para entender, aprender y codificar usando tutoriales permitiendo que cualquier principiante pueda entender los principios básicos de Python.
- Se interpreta línea por línea facilitando la realización de pruebas y la depuración.
- Al ser de código abierto y libre, el idioma y el código fuente están disponibles para el público de forma gratuita para que puedan realizar cambios e incluso distribuirlo.
- Es portátil ya que Python se puede ejecutar en varias plataformas como Windows, Mac, Linux y otras más.
- Permite desarrollar interfaces gráficas de usuario.
- Es extensible ya que permite escribir partes de código en otros lenguajes como C o C++.
- Hace posible que la productividad de los programadores mejore al disponer de un lenguaje simple y que cuenta con extensas bibliotecas a la vez que el hecho de escribir menos permite que se haga más.
- Python al formar parte plataformas de desarrollo como Raspberry Pi, materializa la oportunidad de conectar este lenguaje con el mundo de IoT.
- Es compatible tanto con los paradigmas de programación orientados a objetos como a procedimientos.

2.3. Desarrollo

A continuación, se describen los trabajos más importantes considerando como criterio de selección los siguientes aspectos: empleo de técnicas de detección y medición de gases, uso de hardware y software libre, conexión a plataformas de IoT y utilización de recursos de bajo costo.

2.3.1 Monitoreo Ambiental Inteligente basado en IOT usando Arduino [17].

Este proyecto describe el diseño e implementación de un dispositivo de monitoreo ambiental estandarizado utilizando la placa Arduino, el lenguaje de programación C y una plataforma de Internet of Things. De manera global, el sistema obtiene información del entorno mediante sensores y envía estos datos directamente a Internet, donde pueden ser accedidos en cualquier momento y en cualquier lugar. El sistema permite medir con

precisión variables como temperatura, humedad, nivel de luz, calidad del aire y hace posible la detección de terremotos mediante un sensor de vibraciones.

El sensor de calidad del aire (MQ-135) detecta las muestras que contienen moléculas de gas a través de un electrodo. Luego, estas moléculas reaccionan electroquímicamente a un potencial apropiado aplicado en el electrodo de detección. La corriente eléctrica generada por la reacción es directamente proporcional a la concentración del gas y se convierte en voltaje para que el gas pueda ser medido.

En cuanto al sensor de humedad y temperatura (DHT 22), la mayoría de sensores están construidos en base a una estructura porosa de cerámica y la detección de humedad se realiza mediante la adsorción de agua sobre las superficies cerámicas, haciendo que las propiedades eléctricas como la resistencia, la capacitancia o la conducción electrolítica cambien.

La detección de posibles terremotos se lleva a cabo mediante un sensor de vibraciones sísmicas de baja frecuencia y de alta sensibilidad, capaz de detectar estas ondas sísmicas mediante la conversión del movimiento del terreno en una señal analógica de voltaje.

El sensor de luz consiste de una LDR (Light dependent resistor) o una foto resistencia, la cual permite que la resistencia disminuya a medida que aumenta la intensidad de la luz incidente, y viceversa. Puede actuar como un sensor, ya que se puede obtener una caída de voltaje en función de la luz detectada.

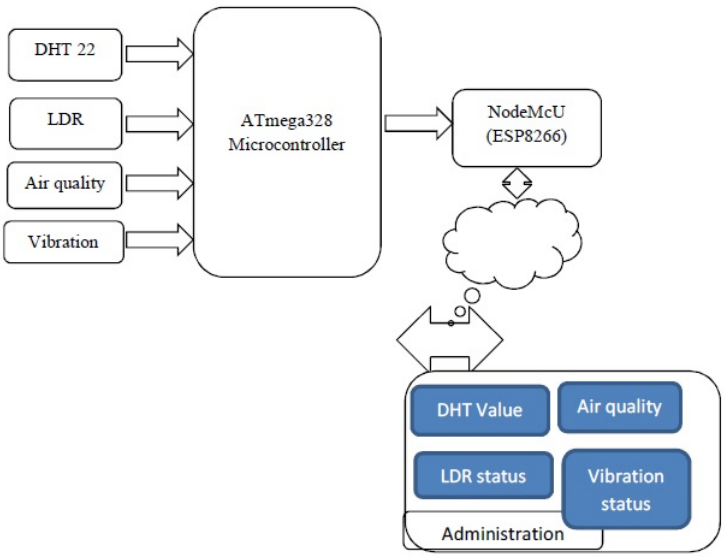


Figura 3. Sistema de monitoreo ambiental inteligente basado en IoT y usando Arduino.

Como se puede observar en la Figura 3, el sistema involucra el uso de los sensores descritos anteriormente que están conectados al microcontrolador del Arduino. Este controlador se puede programar con el IDE de Arduino, ya que viene pre programado con un gestor de arranque que permite al usuario cargar nuevo código para interactuar con una gran cantidad de sensores y dispositivos mediante los pines de entrada / salida analógicos.

El Arduino procesa los datos recibidos de los sensores y los calibra en unidades de medición estándar dependiendo del parámetro ambiental. Luego, mediante una comunicación serial esta información se envía a un módulo NodeMcU, que permite la conexión con el servidor de IoT Adafruit a través de Internet ya que funciona como un adaptador de Wi-Fi para cualquier micro controlador.

El servidor de IoT Adafruit está basado en Ruby on Rails y en Node.js e incluye APIs y librerías como REST y MQTT que están disponibles en el IDE Arduino. Una de las características de estas librerías es que están diseñadas para usar la menor cantidad de recursos posibles para que puedan funcionar adecuadamente en plataformas como Arduino. El servidor se crea en el domino www.adafruit.com, donde el usuario puede crear diferentes paneles e interfaces gráficas de usuario que pueden interactuar con los sistemas desarrollados mediante la dirección IP del servidor.

2.3.2 Monitoreo del Campus de la Universidad de Taif utilizando técnicas de Internet de las Cosas [18].

Las principales aplicaciones actuales de WSN (Wireless Sensor Network) incluyen el monitoreo del medio ambiente, la detección de inundaciones, la predicción del tiempo y de actividades sísmicas. También, las redes de sensores inalámbricos se usan cada vez más en aplicaciones de salud para controlar cambios en el organismo, el comportamiento y la frecuencia cardíaca del paciente permitiendo un monitoreo continuo de la enfermedad y aumentando las oportunidades para ayudar activamente a la persona.

El sistema propuesto en este proyecto consiste en el despliegue de una red de sensores utilizando la tecnología Zigbee, que recopila y monitorea diferentes tipos de mediciones que muestran la condición ambiental de los edificios, incluyendo temperatura, humedad, concentración de CO2 y calidad del aire.

El sistema consiste de los siguientes elementos:

- Micro controlador o Arduino (módulo de procesamiento central): permite el envío de datos a Internet (plataforma cloud de IoT - Cayenne) y hace posible que esta

información pueda ser accedida desde cualquier smartphone. Está presente únicamente en el nodo coordinador.

- Módulo XBee: Es un transceptor de radio frecuencia y una antena que sirven para enviar los datos recopilados y recibir datos e instrucciones de los demás nodos intermedios. Este módulo está presente en los nodos intermedios y en el nodo coordinador.
- Módulo de alimentación: Contiene dispositivos de almacenamiento de energía como baterías recargables.
- Módulos de detección: Se compone de varias sondas y sensores incluyendo amplificadores y conversores analógico-digital para detectar y monitorear los parámetros físico-químicos del entorno.

La red de sensores inalámbricos sigue una topología del tipo malla como se observa en la Figura 4, y consta de los siguientes nodos:

- Nodo coordinador: almacena la información crìtica del sistema.
- Nodos intermedios o enrutadores: si alguno de ellos falla permite mantener la comunicación de toda la red.
- Dispositivos finales: son los diferentes tipos de sensores distribuidos en la red, incluyen:
  - Sensores de infrarrojo pasivo (PIR) que permiten medir la luz infrarroja y detectar movimiento.
  - Sensores digitales de temperatura y humedad (DHT11).
  - Sensores de lluvia que permiten medir la humedad a través de los pines de salida analógica y proporcionan una salida digital cuando se excede el umbral de humedad.
  - Detectores de gases inflamables (MQ-2).
  - Fotoceldas o fotorresistencias (LDR).

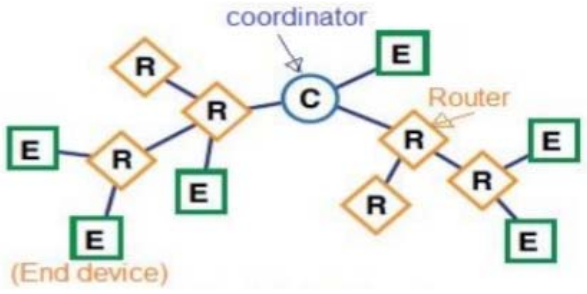


Figura 4. Red de sensores inalámbricos.

Cabe recalcar que los datos que proporciona el sistema son adquiridos en tiempo real a través de la aplicación web cloud de IoT denominada Cayenne, la cual también está disponible para plataformas móviles de forma gratuita en Google Play, permitiendo así el uso de la misma en cualquier smartphone.

Esta aplicación obligatoriamente exige crear una cuenta gratuita en el sitio web: www.cayenne.mydevices.com, para poder pasar los datos de los sensores del sistema a Internet y disponer de un tablero de monitoreo y control de los valores recolectados por dichos sensores. Al ser una aplicación cloud hace posible que esta información pueda ser obtenida remotamente en cualquier parte del mundo mediante el acceso a Internet.

2.3.3 Diseño e implementación de un dispositivo IoT de bajo coste para entornos agrícolas [19].

Este trabajo consiste de un sistema que permite registrar factores ambientales mediante el uso de varios sensores y generar alertas cuando estas mediciones sean perjudiciales para los cultivos, a la vez que se utilizan tecnologías como la plataforma de desarrollo Arduino junto a su IDE de programación, microcontroladores para la conexión Wi-Fi y el establecimiento de un enlace con la plataforma web de IoT llamada ThingSpeak para el envío de los datos recogidos.

Es importante mencionar que ThingSpeak es una plataforma de Internet of Things que hace posible la recolección y almacenamiento de datos al mismo tiempo que permite el desarrollo de aplicaciones IoT. De forma similar a otras plataformas de IoT del mercado como Cayenne, Adafruit, Xively, etc., requiere de la creación de una cuenta gratuita para recibir y almacenar datos enviados desde diferentes dispositivos como Arduino, Raspberry, Beaglebone, entre otros.

Los sensores implementados en este trabajo son:

- Temperatura: El sensor DS18B20 proporciona una salida que cuenta con una serie de pines que deben ser conectados a una entrada digital y a las entradas de alimentación respectivamente en un Arduino.
- Detectores de movimiento: El sensor HC-SR501 utiliza la tecnología infrarroja a fin de evitar intrusos en los cultivos.
- Humedad: El sensor YL-69 permite medir la humedad en el suelo de los cultivos mediante la aplicación de una diferencia de voltaje entre los terminales del sensor. Para medir la humedad ambiental se utiliza el sensor DHT11, el cual también permite obtener el valor de temperatura.

- Lluvia: El sensor MH-RD hace posible la detección de la presencia de lluvia mediante el uso de una superficie con un circuito impreso con forma de canales.
- Sensor fotoeléctrico: El fotorresistor GL5516 cambia el valor de la resistencia dependiendo de la intensidad de luz recibida.
- Sensor de gases: El sensor MQ-2 permite detectar la presencia de gases inflamables como GLP, propano, metano, alcohol, hidrógeno y humo.

El funcionamiento del sistema es el siguiente:

- El dispositivo se enlaza a la red Wi-Fi mediante el micro controlador ESP8266 integrado en la placada WebMos D1 Mini y previamente conectado al Arduino.
- Se leen los estados de las salidas de todos los sensores y se almacenan en una variable del tipo float.
- Se imprime por el puerto serial la información de cada uno de los parámetros ambientales recogidos.
- Se envían estos datos a los canales de ThingSpeak permitiendo observar la evolución de cada uno de los parámetros y las alertas generadas.

2.3.4 Sistema de monitoreo de la calidad del aire basado en IoT usando Raspberry Pi [20].

De acuerdo al diagrama simplificado del sistema que se muestra en la Figura 5, se puede mencionar lo siguiente:

- El dispositivo Raspberry Pi es el nodo principal que controla el sistema. Los sensores se utilizan para detectar diferentes parámetros ambientales, como partículas, monóxido de carbono, dióxido de carbono, temperatura, humedad y presión. Estos sensores están conectados a la placa Arduino y el Raspberry Pi está conectado mediante un cable USB con la placa Arduino. Los datos detectados por los sensores se transmiten continuamente a través del Raspberry Pi a la nube mediante Internet.
- Los sensores DSM501A se utilizan para medir las partículas de humo y polvo presentes en el entorno. DHT22 y BMP180 tienen salidas digitales para medir la temperatura, la humedad y la presión. Los sensores, MQ9 (sensor de gas) y MQ135 (sensor de calidad del aire) son sensores analógicos utilizados para medir el monóxido de carbono y el dióxido de carbono.
- El protocolo ligero MQTT (Message Queuing Telemetry Transport) hace posible el establecimiento de la comunicación entre los sensores y los clientes. El cliente puede acceder a los datos que se muestran en el tablero utilizando la identificación del dispositivo, pero el cliente no puede modificar los datos recibidos. MQTT está diseñado para dispositivos que se comunican a través de redes de alta latencia, bajo ancho de banda y no confiables. A diferencia del protocolo HTTP, no sigue la

arquitectura de petición / respuesta sino que sigue la arquitectura de publicar / suscribir.

- Node-RED es una herramienta de programación visual básica y de código abierto fácil de usar para aplicaciones IoT que permite integrar dispositivos de hardware, API y servicios en línea de una manera interesante y creativa. Node-RED tiene embebido una librería que dispone de miles de flujos y nodos que permiten al usuario conectar todo tipo de dispositivos y servicios. Los flujos se pueden ejecutar en el Raspberry Pi o en la nube, ya que Node-RED incluye el entorno de ejecución Node.js. Una vez que se realiza e implementa el flujo, los datos se pueden ver en el tablero de la plataforma de IoT.

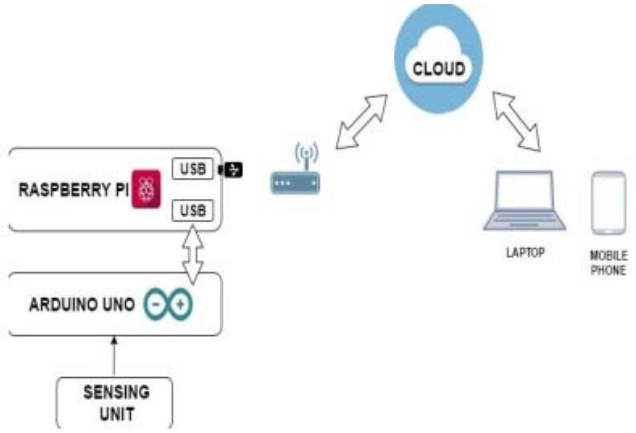


Figura 5. Diagrama simplificado del Sistema de monitoreo de la calidad del aire.

Es importante mencionar que para usar los servicios en la nube de IBM, en este proyecto se ha creado una cuenta en IBM Bluemix y, al mismo tiempo, se ha registrado el dispositivo. Una vez que el dispositivo se haya registrado, la plataforma Bluemix IoT reconocerá al usuario proporcionando el token de autenticación que se puede usar para la comunicación de datos desde el dispositivo hacia la plataforma de IoT.

En este sistema se implementan un flujo de Node-RED, que involucra la conexión de dos nodos. Un nodo Serial In para recibir los datos que provienen de la placa Arduino y un nodo Watson IoT para enviar los datos a la nube.

2.3.5 Sistema de monitoreo de la calidad del aire (AQMS) usando Raspberry Pi e IoT [21].

En este trabajo, los nodos web están diseñados usando un Raspberry Pi como unidad de procesamiento, el módulo electrónico GrovePi+ que se utiliza para conectarse a los sensores y un transceptor Wi-Fi para establecer la comunicación de datos entre el nodo web y el servidor. Un nodo web es un nodo independiente capaz de conectarse al servidor web y

a la base de datos de ThinkSpeak en la nube con la ayuda de un enrutador. La arquitectura general de un nodo web se observa en la Figura 6.

Cada nodo web tiene una base de datos MySQL local que es capaz de conectar y almacenar datos en el servidor de base de datos de la red. El modelo detallado se distingue del AQMS genérico debido a que la ejecución de ciertos servicios como la administración de datos, agregación centralizada de datos, toma de decisiones y análisis de datos para múltiples nodos web requiere más recursos e infraestructura por lo cual debe llevarse a cabo en el servidor de la nube.

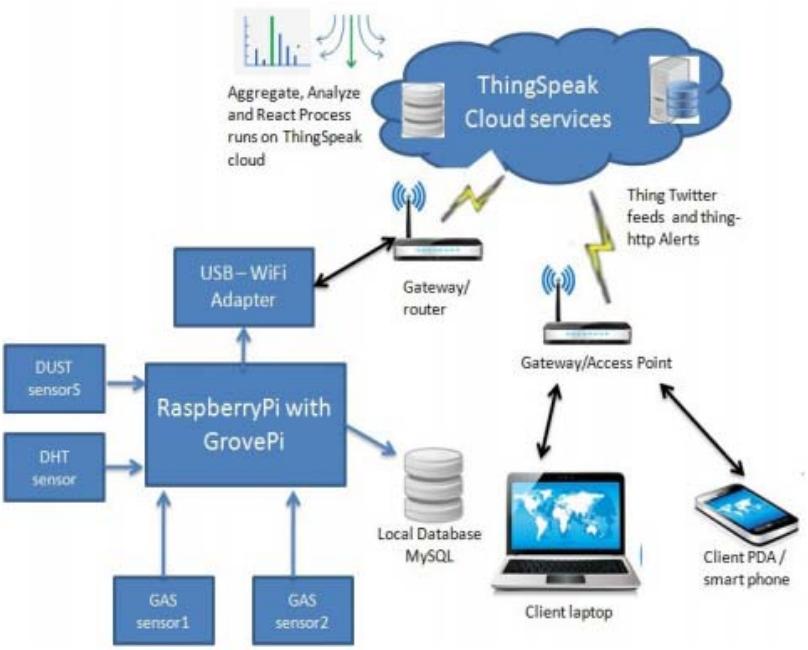


Figura 6. Arquitectura del nodo web.

Los usuarios pueden monitorear los flujos de datos en tiempo real al iniciar sesión en el canal público de los servicios en la nube de ThingSpeak y el administrador puede automatizar ciertos métodos que son oportunos para la toma de decisiones.

El módulo GrovePi+ consiste de un controlador con soporte ADC e i2c. El Raspberry Pi y este módulo se comunican utilizando la interfaz i2c. Se pueden conectar múltiples sensores analógicos y digitales al Raspberry Pi mediante el módulo GrovePi+. Se utiliza un sensor DHT (para temperatura y humedad), un sensor MQ-5 (para humo), MQ-7 (para CO) y MQ-135 (para CO2) a fin de supervisar las variables del entorno. Los controladores y los scripts de la aplicación se desarrollan utilizando PYTHON.

ThingSpeak permite la supervisión remota de datos y ofrece otros servicios como la visualización de datos con soporte para MATLAB, alertas vía Twitter y comunicación de

dispositivos. Al igual que en otros proyectos, es necesario crear una cuenta de usuario en https://thingspeak.com/ y generar los canales necesarios para la supervisión y análisis. Para realizar un análisis offline, los datos se pueden importar desde la base de datos ThingSpeak en formato CSV, JSON o PHP.

2.4. Conclusiones

Existen muchos sensores de gases de la familia MQ, de entre los cuales resaltan los siguientes:

- MQ-2: Este sensor es sensible a una gran variedad de gases, pero principalmente se usa para detectar GLP, metano y propano en interiores.
- MQ-135: Este sensor es sensible a varios gases como NH3, NOx, alcohol, CO2 y humos por lo que se utiliza fundamentalmente en el control de la calidad del aire en interiores.
- MQ-7: Este sensor de muy bajo costo se utilizará en este trabajo fin de máster ya que es un sensor específico para detectar concentraciones de monóxido de carbono en el aire que van desde las 20 ppm hasta las 2000 ppm. El MQ-7 tiene dentro un pequeño calentador junto con un sensor electro-químico, que son los encargados de generar corriente eléctrica y una señal analógica que puede ser leída por un pin de entra analógica del Arduino.

La tecnología WSN es una buena opción para comunicar sensores inalámbricos en exteriores de una forma confiable y que además ofrece redundancia debido al uso de nodos intermedios que se encargan de garantizar el enrutamiento hacia el nodo coordinado en las peores condiciones. Debido a que el proyecto planteado se realizará en interiores donde se cuenta con una red Wi-Fi y únicamente se hará uso de un dispositivo integrado de recepción, procesamiento y envío de datos a Internet, la aplicación de esta tecnología no tendría sentido alguno o si se implementara sería subutilizada.

Debido a que uno de los requisitos del sistema a diseñar e implementar es el bajo costo se deberá considerar el uso de dispositivos hardware que cumplan con este requisito, siendo los candidatos: Arduino, Raspberry Pi, o una combinación de ambos. A continuación, se realizará un análisis de cada una de estas plataformas de desarrollo.

La placa Arduino Mega al no haber sido diseñada para IoT, no tiene embebido el hardware necesario para contar con una conexión Wi-Fi, razón por la cual el dispositivo Raspberry Pi 3 Modelo B presenta una ventaja al contar con esta característica por defecto integrada en su placa.

Arduino presenta un tiempo de arranque menor que Raspberry Pi 3 pero a medida que se añaden módulos o funcionalidades a la placa base este parámetro es comparable al del sistema operativo Rapbian Stretch de la plataforma Raspberry Pi 3, el cual presenta tiempo de arranques mucho menores que los que se obtenían en anteriores versiones del sistema operativo y en plataformas como el Raspberry Pi 1 y 2.

Raspberry Pi en cualquiera de sus modelos no dispone de entradas analógicas. En cambio, Arduino sí cuenta con este tipo de entradas y todas las demás que presenta Raspberry Pi, razón por la que Arduino permite el tratamiento de señales analógicas emitidas por una gran variedad de dispositivos como sensores.

Un problema que presenta Arduino es el aumento de energía consumida cada vez que se le añaden componentes. Sin embargo, en el Raspberry Pi esta característica se encuentra optimizada a fin de reducir al máximo el consumo de energía.

Para establecer una conexión con una plataforma de IoT, en Arduino es necesario configurar tanto el dispositivo Wi-Fi como el código del micro controlador de Arduino. No obstante, en Raspberry Pi la configuración para comunicarse con la aplicación web es más sencilla ya que no se necesitan incluir una serie de librerías como en el IDE de Arduino y resulta mucho mejor si la aplicación web no es propietaria brindado así la suficiente libertad y flexibilidad al usuario.

Arduino al igual que Raspberry Pi son considerados dispositivos de hardware libre de bajo costo siendo Arduino el más accesible, popular y de menor valor en el mercado.

Considerando la facilidad de conexión Wi-Fi, tiempos de arranque del sistema operativo, capacidad para manejar entradas analógicas, optimización del consumo de energía, sencillez y flexibilidad en la configuración para establecer la comunicación con una aplicación web se ha considerado utilizar tanto Arduino como Raspberry Pi a fin de aprovechar al máximo cada una de las ventajas de estos dispositivos en el sistema propuesto.

Existen muchas plataformas de IoT capaces de recibir datos de sensores que miden parámetros ambientales como temperatura, humedad, vibraciones sísmicas, detección de gases, medición de la intensidad de luz; unas con más facilidades que otras pero que dependen totalmente de las capacidades que ofrece la plataforma. Es por esta razón que el proyecto a desarrollar plantea la creación de una aplicación web cloud no propietaria que permita recibir los datos y visualizarlos en tiempo real y desde cualquier ubicación si se dispone del acceso autorizado a la vez que se utilizan servicios sencillos REST de forma segura. De esta manera también se evita tener que configurar información sensible de

autenticación para la plataforma de IoT en Arduino o configurar el código necesario en el IDE para definir los canales a usar para enviar los datos a la aplicación.

Por último, las plataformas de IoT utilizan determinados protocolos para el envío de datos como MQTT y por consiguiente es obligatorio incluir estas librerías al desarrollar el código en Arduino. Debido a esto, lo mejor es realizar esta conexión en Raspberry Pi mediante Python donde el número de librerías a utilizar se reduce notablemente al desarrollar y emplear una aplicación web cloud propia e independiente.

3. Objetivos concretos y metodología de trabajo

3.1. Objetivo general

Implementar un sistema de detección y registro de niveles de CO de bajo costo en domicilios usando hardware y software libre, tecnologías web y servicios cloud de forma que se pueda salvaguardar vidas humanas.

3.2. Objetivos específicos

- Analizar los niveles de CO que sean perjudiciales para la salud de las personas en domicilios.
- Instalar y configurar dispositivos de hardware libre y un sensor de gases para poder detectar y medir el monóxido de carbono en domicilios.
- Utilizar dispositivos de hardware libre para la lectura, procesamiento y transmisión de los niveles de CO en el entorno.
- Crear un servicio web que permita el envió de las mediciones de CO desde el Raspberry Pi hacia la aplicación web.
- Diseñar e implementar una aplicación web que pueda recibir los datos transmitidos por el sensor, cumpla con el patrón MVC, satisfaga los requisitos de seguridad en cuanto a control de acceso, permita visualizar las mediciones y emita notificaciones en caso de ser necesario.
- Utilizar servicios en la nube con la finalidad de colocar la plataforma en línea y por ende que la herramienta desarrollada sea funcional.
- Desarrollar un esquema de generación de reportes en base a los datos medidos.
- Realizar pruebas de evaluación y verificación del sistema.

3.2. Metodología del trabajo

Se conectará el sensor de CO al dispositivo Arduino, el cual convertirá las señales analógicas en digitales y enviará estos datos al Raspberry Pi mediante una conexión serial. Este dispositivo utilizará un servicio web creado previamente para enviar mediciones de voltaje, lecturas ADC, estados de detección, cantidad de partículas por millón de CO y valores de mg/m3 del monóxido de carbono a una base de datos que forma parte de la aplicación web desplegada en la nube.

A partir de los datos obtenidos desde el sensor, se realizará una interpretación y análisis de dicha información que permita realizar varias gráficas, contribuyendo a la concepción de una visión más clara del comportamiento en las mediciones registradas.

La metodología de trabajo a utilizar en el presente trabajo fin de máster será la metodología en cascada. Esta metodología según [23], toma las actividades comunes para todos los procesos de desarrollo de software (especificación, desarrollo, validación y evolución), y las representa de acuerdo a la Figura 7 como fases individuales secuenciales: análisis de requisitos, diseño, implementación y pruebas. Una fase no puede comenzar hasta que no haya finalizado la anterior.

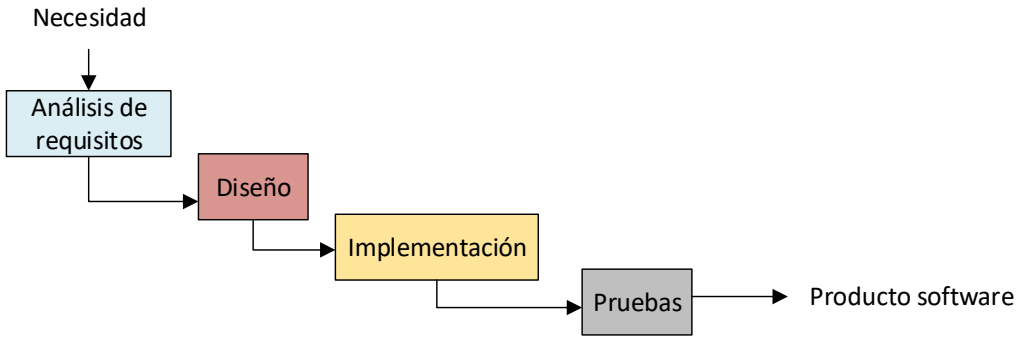


Figura 7. Ciclo de vida del desarrollo de software para la metodología en cascada.

Análisis de requisitos: En base al diagrama de casos de uso y las tablas correspondientes a la especificación de cada uno de los casos de uso se podrá determinar los requisitos funcionales. También es importante obtener los requisitos no funcionales.

Diseño: Se detallará la arquitectura de la aplicación web para detectar y medir los niveles de CO en domicilios. Además en base a las especificaciones de cada uno de los casos de uso se obtendrán los diagramas de secuencia, diagramas de clase y diagramas de entidad-relación.

Implementación: Se realizará la codificación del Back-End utilizando el entorno de ejecución Node.js, el framework Express.js, el lenguaje de programación JavaScript y el servidor de base de datos NoSQL MongoDB. También, se implementará la codificación del Front.End utilizando el framework Angular y el lenguaje de programación TypeScript.
En cuanto al equipo transmisor de las mediciones de los niveles de CO, se desarrollará código en el IDE Arduino y código Python en el Raspberry Pi. Además, se presentará el costo referencial del sistema de detección y registro de niveles de CO en domicilios.

Pruebas: Durante la fase de implementación ya se habrán realizado las pruebas de cada uno de los componentes de la aplicación web que reflejarán el cumplimiento de los requisitos funcionales. A fin de que los requisitos no funcionales también estén cubiertos se realizará una encuesta sobre diferentes aspectos de la aplicación web a funcionarios del área de TIC de la Unidad de Negocio CELEC EP Coca Codo Sinclair.

En la Figura 8 se muestra el diagrama de casos de uso para el sistema y la aplicación web a desarrollar. Desde la Tabla 1 hasta la Tabla 10 se observa el detalle para cada caso de uso.

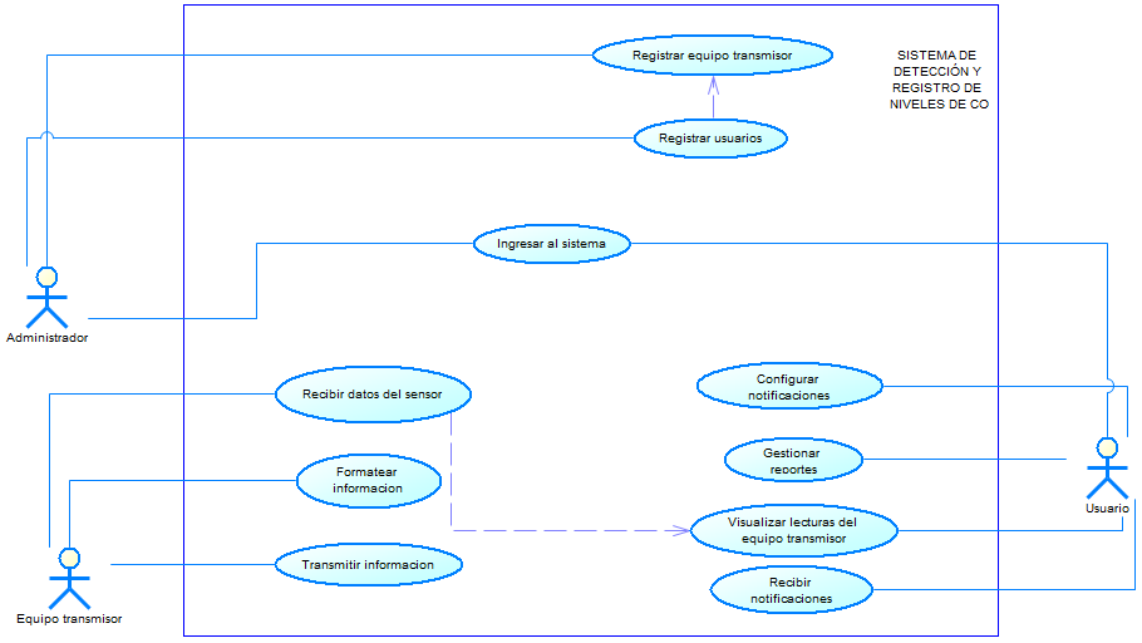


Figura 8. Diagrama de caso de uso para el sistema de detección y registro de niveles de CO de bajo costo en domicilios.

Tabla 1. Especificación del caso de uso: Registrar usuarios.

Identificador	CU-DES-APP-ADM-001	
Nombre	Registrar usuarios.	
Descripción	Se usa para poder registrar a todos los usuarios involucrados en el sistema.	
Precondición	El administrador ha accedido al sistema y cuenta con todos los permisos necesarios.	
Postcondición	Los cambios realizados por el administrador se ven reflejados en las cuentas de los usuarios afectadas.	
Actores	Administrador, usuario.	
Secuencia Normal	**Paso**	**Acción**
	1	El administrador consulta los datos de los usuarios. En caso de que no se encuentre un usuario se sigue la Secuencia alternativa.
	2	El sistema muestra la información solicitada consultando a la base de datos.
	3	El administrador puede crear, editar o eliminar usuarios específicos.
	4	En el caso de que se trate de guardar valores no permitidos o incompletos se sigue la Secuencia de error.
	5	Una vez que se han validado los valores, el sistema actualiza la base de datos.
	6	Para cualquier acción del administrador, el sistema muestra un mensaje de confirmación.
Secuencia alternativa	**Paso**	**Acción**
	1	El administrador está listo para registrar un nuevo usuario.
	2	Vuelve al paso 3 de la Secuencia Normal.
Secuencia de error	**Paso**	**Acción**
	1	Se informa donde se ha cometido el error para que el campo pueda ser llenado correctamente.
	2	Con los campos rellenados, vuelve al paso 5 de la Secuencia Normal.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	Una vez que el administrador ha modificado el registro de algún usuario, este cambio se reflejará cuando el usuario acceda al sistema.	

Tabla 2. Especificación del caso de uso: Registrar equipo transmisor.

Identificador	CU-DES-APP-ADM-002	
Nombre	Registrar equipo transmisor.	
Descripción	Se usa para poder registrar el equipo transmisor en el sistema y poder asignarlo a un usuario.	
Precondición	El usuario a asignar debe estar registrado en el sistema.	
Postcondición	Al registro del equipo transmisor se le podrán asignar las lecturas de las mediciones correspondientes.	
Actores	Equipo transmisor.	
Secuencia Normal	**Paso**	**Acción**
	1	El sistema valida los datos registrados del usuario.
	2	El sistema crea un nuevo equipo transmisor y lo asocia al usuario.

	3	El sistema actualiza la base de datos y muestra un mensaje de confirmación.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	Es indispensable que se asigne el equipo transmisor al usuario correcto.	

Tabla 3. Especificación del caso de uso: Ingresar al sistema.

Identificador	CU-DES-APP-ADM-003	
Nombre	Ingresar al sistema.	
Descripción	Se usa para que tanto el usuario como el administrador puedan ingresar al sistema.	
Precondición	Tanto el usuario como el administrador deberán conocer la dirección de la página principal de la aplicación web.	
Postcondición	El usuario o el administrador podrán hacer uso de las funcionalidades permitidas por el sistema.	
Actores	Administrador, usuario.	
Secuencia Normal	**Paso**	**Acción**
	1	El usuario/administrador ingresa las credenciales de acceso.
	2	El sistema realiza la consulta acerca del login realizado y accede a la base de datos.
	3	La base de datos retorna los registros del usuario/administrador.
	4	El sistema genera un token que junto a los datos de usario se envían hacia el Front End.
	5	El sistema muestra un mensaje indicando que el acceso ha sido exitoso caso contrario se sigue la Secuencia de error.
Secuencia de error	**Paso**	**Acción**
	1	El sistema mediante un mensaje indica que el usuario o password ingresados no son correctos.
	2	Se vuelve al paso 1 de la Secuencia normal.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	Tanto el usuario como el administrador deben estar logeados en el sistema para poder realizar cualquier acción.	

Tabla 4. Especificación del caso de uso: Gestionar reportes.

Identificador	CU-DES-APP-USR-001	
Nombre	Gestionar reportes.	
Descripción	Se usa para poder obtener reportes de acuerdo a los datos obtenidos del equipo transmisor.	
Precondición	Deben existir registros de las lecturas de los equipos transmisores.	
Postcondición	El usuario podrá visualizar el reporte generado.	
Actores	Usuario.	
Secuencia Normal	**Paso**	**Acción**
	1	El usuario solicita la generación de un reporte ingresando una fecha inicial y final.

	2	El sistema verifica que existan registros para las fechas requeridas accediendo a la base de datos. Caso contrario sigue la Secuencia de error.
	3	El sistema obtiene los valores máximos, mínimos y promedios para todas las variables medidas durante el período de tiempo consultado.
	4	El sistema muestra los valores en una tabla y los grafica de forma adecuada.
	5	El usuario puede analizar los valores y el comportamiento de las variables medidas.
Secuencia de error	**Paso**	**Acción**
	1	Se indica el error producido y el usuario modifica el rango de fechas ingresado para que sea posible la generación del reporte.
	2	Se vuelve al paso 3 de la Secuencia normal.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	Únicamente el usuario puede solicitar la generación de reportes.	

Tabla 5. Especificación del caso de uso: Visualizar lecturas del equipo transmisor.

Identificador	CU-DES-APP-USR-002	
Nombre	Visualizar lecturas del equipo transmisor.	
Descripción	Se usa para que el usuario pueda visualizar las lecturas del equipo o equipos transmisores asignados previamente.	
Precondición	Todos los datos relacionados a las mediciones realizadas por el equipo transmisor deben estar correctamente almacenados en la BD.	
Postcondición	El usuario podrá cambiar el rango de fechas y horas en la visualización de los datos tantas veces como desee.	
Actores	Usuario.	
Secuencia Normal	**Paso**	**Acción**
	1	El usuario consulta las lecturas del equipo transmisor que desea visualizar. En caso de que no se encuentren las lecturas se sigue la Secuencia alternativa.
	2	El sistema realiza la consulta acerca de las lecturas solicitadas y accede a la base de datos.
	3	La base de datos retorna los registros de las lecturas.
	4	El sistema muestra al usuario las lecturas del equipo transmisor solicitadas de forma gráfica. En caso de que no se pueda visualizar de forma correcta se sigue la Secuencia de error.
Secuencia alternativa	**Paso**	**Acción**
	1	El sistema muestra un mensaje indicando que la lectura no existe.
	2	Vuelve al paso 1 de la Secuencia Normal.
Secuencia de error	**Paso**	**Acción**
	1	Se informa al usuario acerca del error.
	2	Vuelve al paso 1 de la Secuencia Normal.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	El usuario es el único que podrá visualizar las lecturas del equipo transmisor.	

Tabla 6. Especificación del caso de uso: Configurar notificaciones.

Identificador	CU-DES-APP-USR-003	
Nombre	Configurar notificaciones.	
Descripción	Se usa para poder configurar los parámetros de las notificaciones.	
Precondición	El usuario debe estar registrado en el sistema y debe tener asignado un equipo transmisor.	
Postcondición	Una vez que el usuario configure los parámetros de las notificaciones, el sistema generará las notificaciones y el usuario podrá recibir las mismas.	
Actores	Usuario.	
Secuencia Normal	**Paso**	**Acción**
	1	El usuario configura los niveles de alarma y emergencia.
	2	En el caso de que los valores ingresados no sean correctos se sigue la Secuencia de error.
	3	El sistema muestra un mensaje de confirmación.
Secuencia de error	**Paso**	**Acción**
	1	Se indica en que campo existe el error para que se verifique el valor.
	2	Superado el error, se vuelve al paso 3 de la Secuencia Normal.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	El sistema envía las notificaciones una vez que compara las mediciones registradas de CO con los parámetros configurados y únicamente si superan los límites establecidos.	

Tabla 7. Especificación del caso de uso: Recibir notificaciones.

Identificador	CU-DES-APP-USR-004	
Nombre	Recibir notificaciones.	
Descripción	Se usa para que el usuario pueda recibir notificaciones que han sido generadas automáticamente por el sistema.	
Precondición	El usuario ha configurado los valores máximos permitidos para cada tipo de notificación.	
Postcondición	El usuario recibirá las notificaciones correspondientes dependiendo del nivel de concentración del monóxido de carbono y podrá tomar medidas inmediatas en su domicilio.	
Actores	Usuario.	
Secuencia Normal	**Paso**	**Acción**
	1	El equipo transmisor está continuamente enviando las mediciones al sistema.
	2	El sistema almacena los registros de las mediciones en la base de datos.
	3	El sistema compara las mediciones con los valores máximos configurados por el usuario para cada tipo de notificación. Si la comparación falla se sigue la Secuencia de error.
	4	En el caso que se superen los valores máximos, el sistema genera una notificación.
	5	Si los valores no superan los límites configurados se sigue la Secuencia alternativa.
	6	El usuario recibe las notificaciones emitidas por el sistema.

Secuencia alternativa	**Paso**	**Acción**
	1	El sistema sigue recibiendo los datos del equipo transmisor.
	2	Vuelve al paso 2 de la Secuencia Normal.
Secuencia de error	**Paso**	**Acción**
	1	Se indica la condición de error para que se verifique el valor.
	2	Se vuelve al paso 3 de la Secuencia Normal.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	El sistema envía las notificaciones una vez que compara las mediciones registradas de CO con los parámetros configurados y únicamente si superan los límites establecidos.	

Tabla 8. Especificación del caso de uso: Recibir datos del sensor.

Identificador	CU-DES-APP-ETX-001	
Nombre	Recibir datos del sensor.	
Descripción	Se usa para que el Raspberry Pi pueda recibir los datos obtenidos por el sensor y la placa Arduino.	
Precondición	El administrador desarrolló el código en el IDE Arduino para poder obtener las mediciones correctas en base a las señales enviadas por el sensor y que puedan ser impresas vía serial.	
Postcondición	El administrador podrá formatear la información en base a los datos recibidos para su posterior envío a la aplicación web.	
Actores	Administrador.	
Secuencia Normal	**Paso**	**Acción**
	1	El administrador establece el puerto serial a utilizar.
	2	El administrador define el reseteo inicial.
	3	El administrador establece que el buffer se encuentre vacío.
	4	El sistema comienza a leer línea a línea los datos enviados por la placa Arduino caso contrario se sigue la Secuencia de error.
Secuencia de error	**Paso**	**Acción**
	1	El sistema muestra al administrador el código del error producido.
	2	El administrador resuelve el error y vuelve al paso 4 de la Secuencia normal.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	El administrador necesariamente debe realizar un reseteo inicial y vaciar el buffer a fin de que posteriormente se pueda formatear la información.	

Tabla 9. Especificación del caso de uso: Formatear información.

Identificador	CU-DES-APP-ETX-002
Nombre	Formatear información.
Descripción	Se usa para que los datos recibidos puedan ser transmitidos en el formato correcto.
Precondición	El sistema debe leer continuamente línea a línea los datos enviados por la placa Arduino.

Postcondición	El administrador podrá enviar la información formateada a la aplicación web.	
Actores	Administrador.	
Secuencia Normal	**Paso**	**Acción**
	1	El administrador elimina los caracteres de final de línea \n.
	2	El administrador almacena el resultado en una lista.
	3	El administrador separa los campos de las medidas y elimina los caracteres 'b.
	4	El sistema imprime la salida y en caso que el administrador detecte que la salida no es la adecuada sigue la Secuencia de error.
Secuencia de error	**Paso**	**Acción**
	1	El administrador corrige el formato del campo de salida erróneo.
	2	Vuelve al paso 4 de la Secuencia normal.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	El administrador debe asegurarse de que cada campo de las medidas únicamente contenga los valores correctos sin ningún carácter adicional.	

Tabla 10. Especificación del caso de uso: Transmitir información.

Identificador	CU-DES-APP-ETX-003	
Nombre	Transmitir información.	
Descripción	Se usa para que los datos con el formato correcto puedan ser enviados a la aplicación web.	
Precondición	Los datos deben estar disponibles en el formato adecuado.	
Postcondición	La aplicación web podrá almacenar los datos enviados por el Raspberry Pi.	
Actores	Administrador.	
Secuencia Normal	**Paso**	**Acción**
	1	El administrador define una función para el envío de la información.
	2	El administrador emplea la función con las mediciones previamente formateadas.
	3	El sistema envía la información a la aplicación web mediante el consumo de un servicio REST. Si se produce algún error en el envío se sigue la Secuencia de error.
Secuencia de error	**Paso**	**Acción**
	1	El sistema muestra un mensaje al administrador indicando que se ha producido un error.
	2	El administrador identificada el error y lo resuelve. Vuelve al paso 3 de la Secuencia normal.
Importancia	Muy importante	
Urgencia	Inmediatamente	
Observaciones	El administrador debe asegurarse de que la aplicación web esté disponible al momento de iniciar la transmisión de datos.	

4. Desarrollo práctico

4.1. Identificación de requisitos

Los requisitos funcionales indispensables para el diseño e implementación de un sistema de bajo costo para la detección y medición del gas tóxico monóxido de carbono en domicilios haciendo uso de hardware y software libre, tecnologías web y servicios cloud de forma que se pueda salvaguardar vidas humanas son:

- El proyecto permitirá la medición constante de los niveles de CO en el medio ambiente, mediante la utilización de un sensor para este gas, un Arduino y un Raspberry Pi.
- Las mediciones de CO serán transmitidas continuamente a un servidor por medio del Raspberry Pi, para su posterior tratamiento y visualización.
- El servidor web, alojado en la nube deberá utilizar software libre en su gran mayoría.
- El servidor web expondrá un servicio que permita la transmisión de información entre el dispositivo de medición y la aplicación.
- La aplicación web permitirá registrar los datos de un usuario para que posteriormente este pueda ingresar al sistema.
- La aplicación web exigirá al usuario que ingrese las credenciales de acceso para poder hacer uso de las funcionalidades del sistema.
- La aplicación web permitirá registrar el equipo transmisor en el sistema y asociarlo a un usuario en específico.
- La aplicación web posibilitará la configuración de los parámetros de las notificaciones a la vez que generará notificaciones automáticamente cuando se registren mediciones fuera del valor umbral configurado.
- La aplicación web permitirá visualizar las notificaciones generadas.
- La aplicación web implementará una correcta visualización de los datos recibidos desde el equipo transmisor.
- Se deberá generar reportes en base a la información obtenida del equipo transmisor.
- La aplicación web permitirá al usuario administrar el equipo transmisor con el fin de realizar actividades de mantenimiento.

Los requisitos no funcionales a cumplir son:

- La aplicación web deberá ser segura y permitir que el usuario pueda autenticarse mediante un usuario y contraseña.
- La aplicación web deberá estar disponible para los usuarios cuando estos requieran obtener datos.
- La aplicación web podrá ser transferida de un entorno a otro con facilidad.
- La aplicación web permitirá que nuevos equipos de transmisión se integren sin afectar el diseño.
- La aplicación web permitirá que nuevos cambios puedan ser realizados de forma rápida y sencilla.

- La aplicación web deberá ser fácil de comprender, utilizar y atractiva al cliente.
- La aplicación web deberá funcionar de acuerdo a los requisitos definidos anteriormente.

4.2. Descripción del sistema software desarrollado

De acuerdo a los requisitos identificados en el apartado anterior se diseñó la siguiente arquitectura para el sistema de bajo costo para la detección y medición del gas tóxico monóxido de carbono en domicilios.

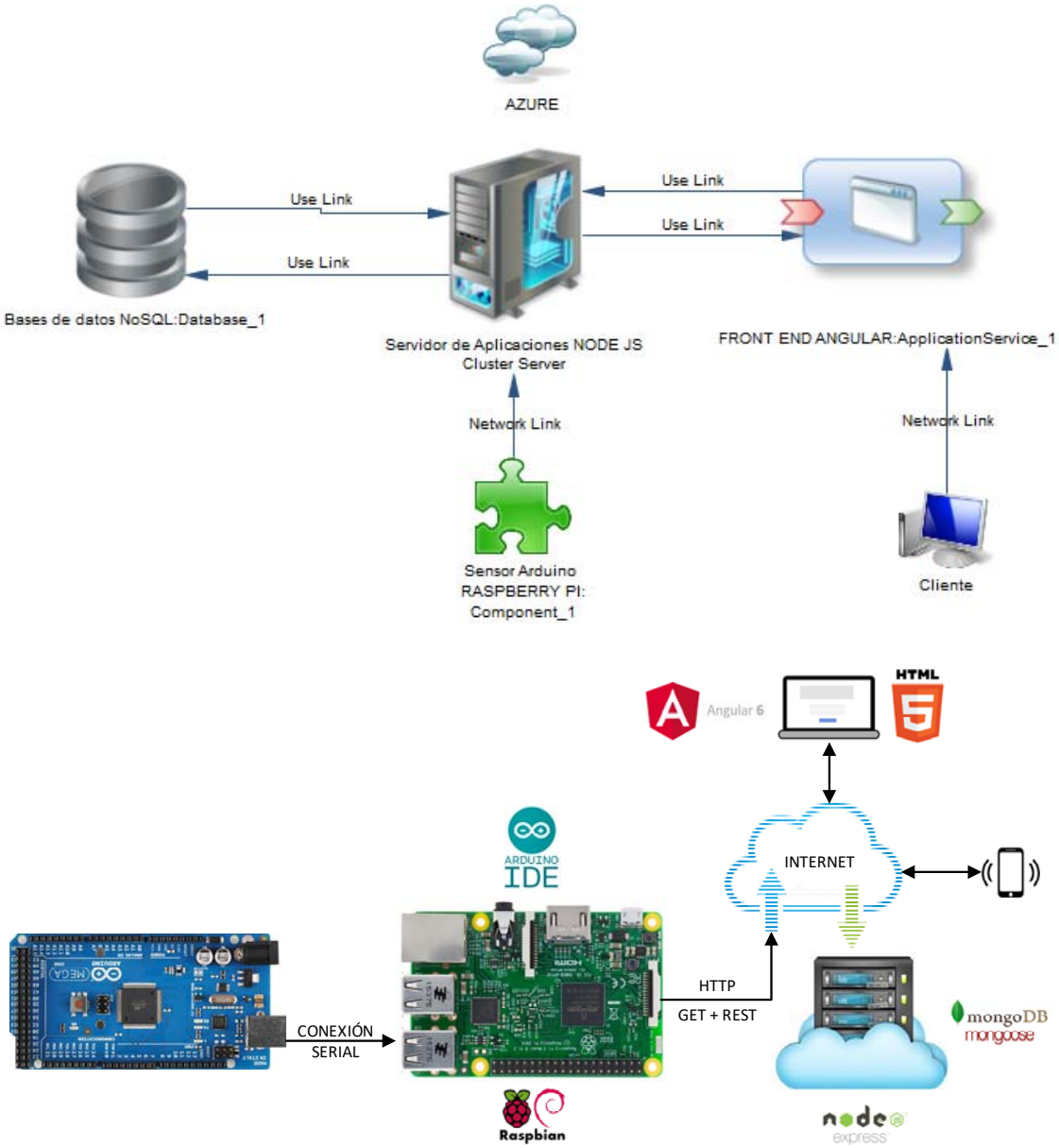


Figura 9. Arquitectura del sistema de detección y medición de CO en domicilios.

4.2.1 Instalación y configuración del sistema operativo Raspbian.

Como punto de partida se carga el sistema operativo Raspbian Stretch versión 2018-04-18 en una tarjeta micro SD mediante el programa Win32DiskImager. Este programa permite seleccionar el archivo de la imagen del sistema operativo así como el dispositivo de almacenamiento donde se alojará este archivo. Esto se observa en la Figura 10.

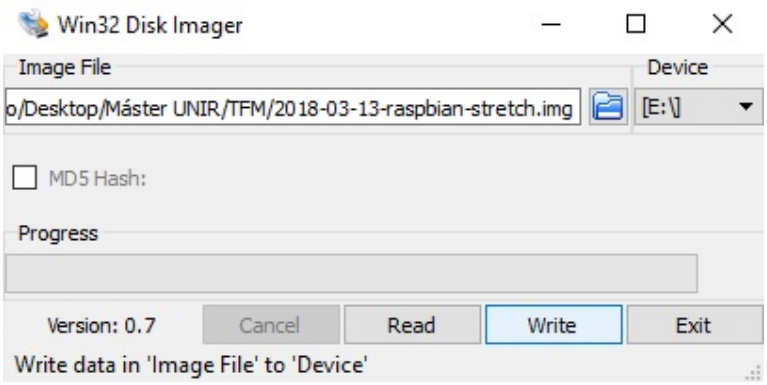


Figura 10. Empleo de Win32DiskImager para grabar Raspbian Stretch.

Luego se inserta la tarjeta micro SD en la ranura posterior de la plataforma de desarrollo Raspberry Pi y se espera a que el proceso de arranque culmine. Inmediatamente se solicita el ingreso del usuario y clave por defecto que son pi y raspberry correspondientemente.

Después mediante la herramienta raspi-config se puede configurar el software del Raspberry Pi como se observa en la Figura 11.

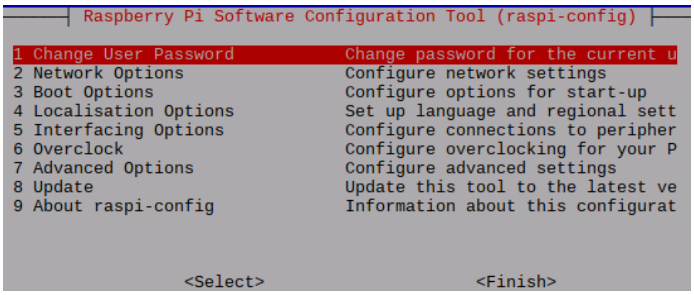


Figura 11. Herramienta raspi-config.

Una de las opciones importantes en el menú de la figura anterior es “Advanced Options”, ya que dentro de este submenú se puede expandir el sistema de archivos para que toda la capacidad de almacenamiento de la tarjeta micro SD pueda ser ocupada por el sistema operativo como se observa en la Figura 12.

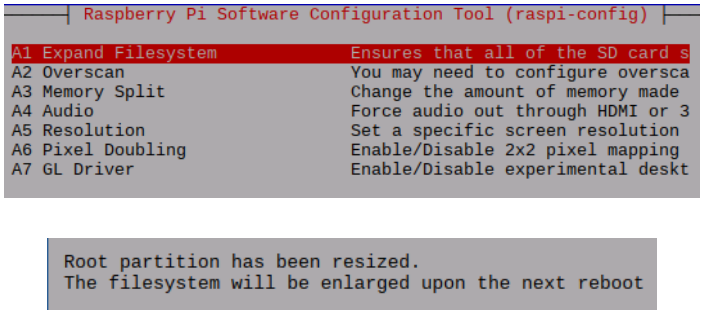


Figura 12. Expansión del sistema de archivos a la capacidad máxima de la tarjeta micro SD.

Otra opción a considerar es “Localisation Options”, dentro de la cual es posible cambiar la zona horaria y ajustar este parámetro a la localidad geográfica más próxima. En este caso se selecciona Guayaquil como se muestra en la Figura 13.

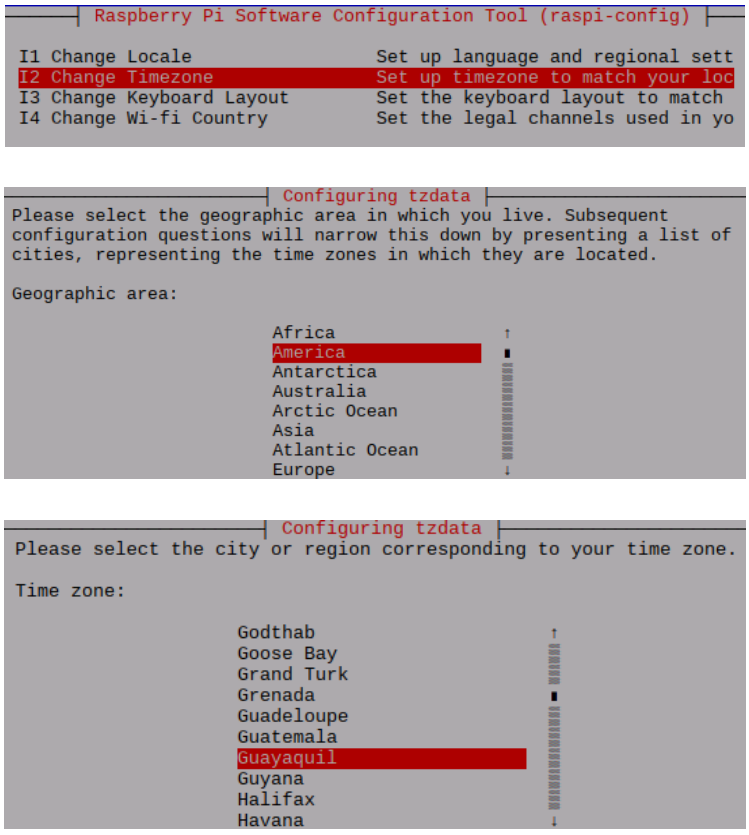


Figura 13. Configuración de la zona horaria.

Dentro de “Interfacing Options” se pueden activar servicios importantes del software del Raspberry Pi como SSH y VNC, los cuales hacen posible que se establezca una comunicación remota con el software, se active un monitor externo y se visualice el escritorio de Raspbian Stretch como el de la Figura 14.

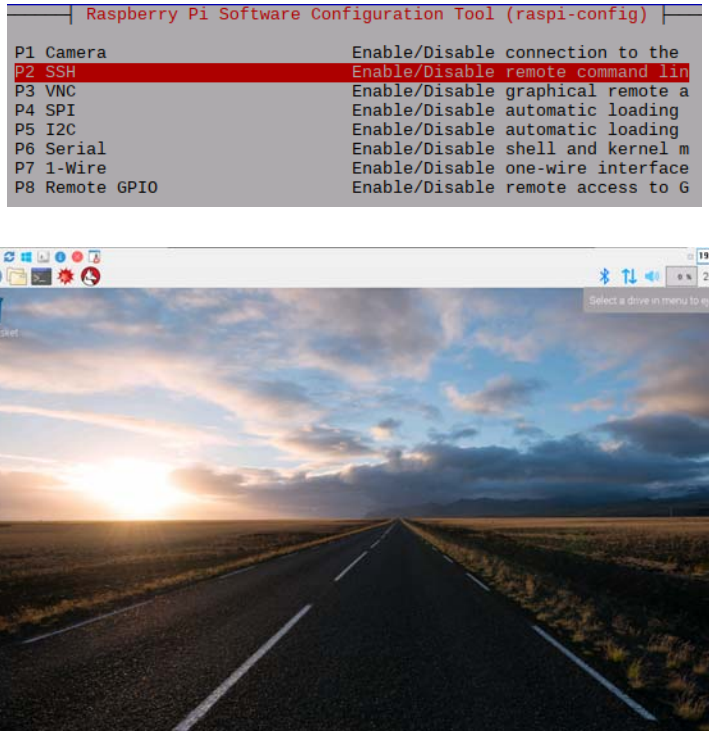


Figura 14. Activación de SSH y VNC que permite la visualización de un escritorio remoto.

En el Anexo I se detalla el Manual de puesta en marcha de la herramienta, en el que se incluye la instalación del IDE Arduino, las consideraciones para la conexión del sensor de CO, la creación de una máquina virtual con Ubuntu Server 18.04 LTS y la instalación de MongoDB, Node.js, npm y Angular CLI.

4.2.2 Diseño e implementación de la aplicación web.

En base al diagrama de caso de uso y especificaciones para cada uno de los casos de uso del Capítulo 3, se definieron los diagramas de secuencia que inician en la Figura 15 y terminan en la Figura 24. De igual forma se estableció el diagrama de clases de la Figura 25, los diagramas de Entidad-Relación lógico y conceptual de la Figura 26 y 27.

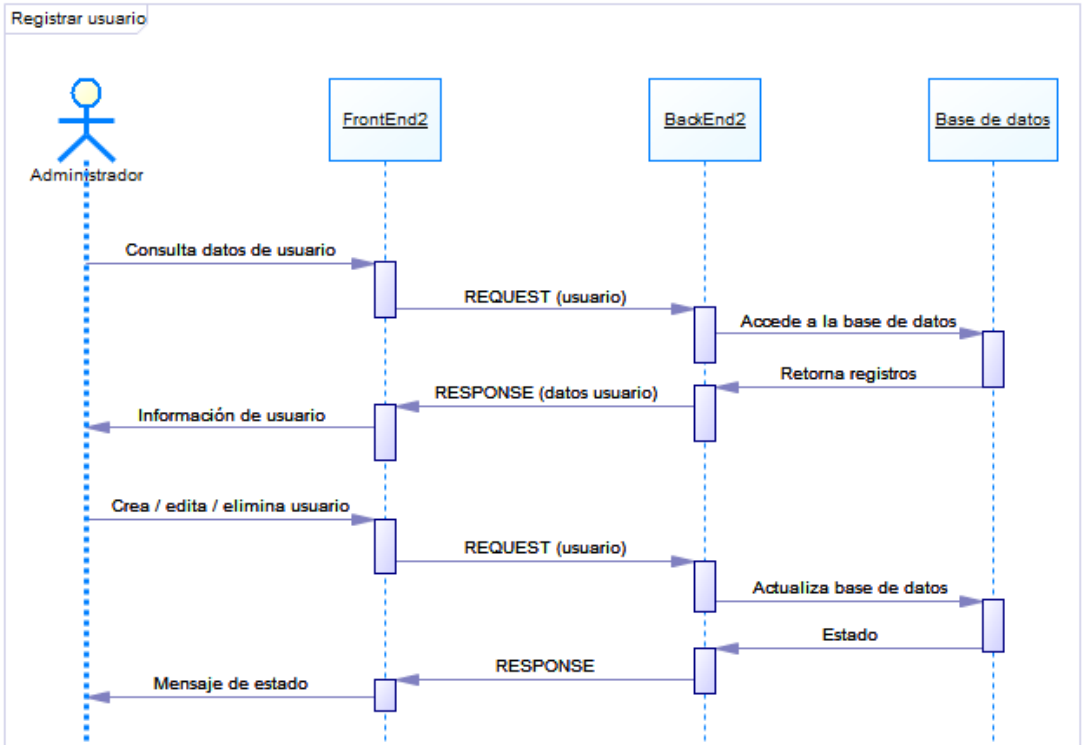


Figura 15. Diagrama de secuencia para registrar usuarios.

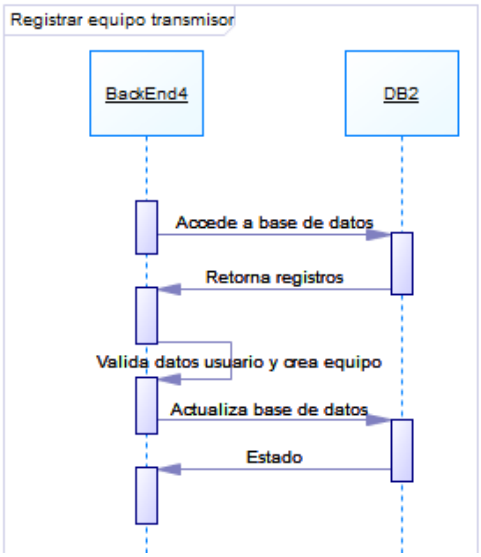


Figura 16. Diagrama de secuencia para registrar equipo transmisor.

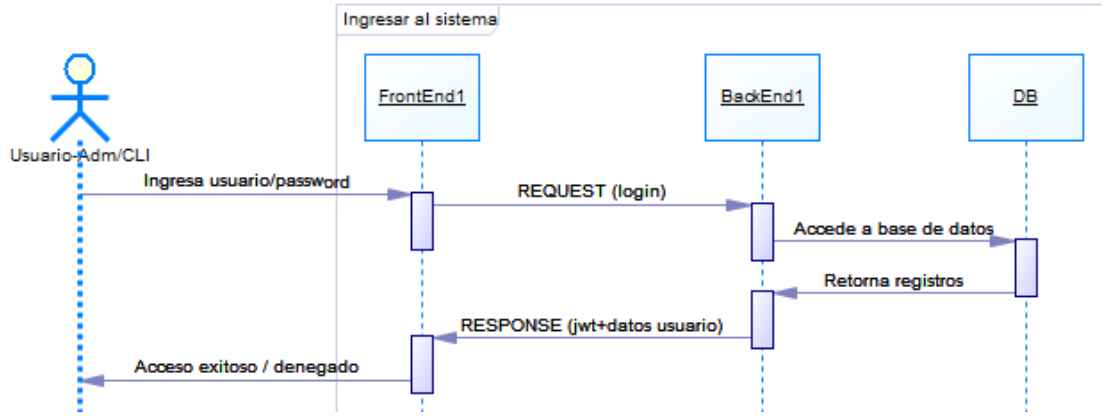


Figura 17. Diagrama de secuencia para ingresar al sistema.

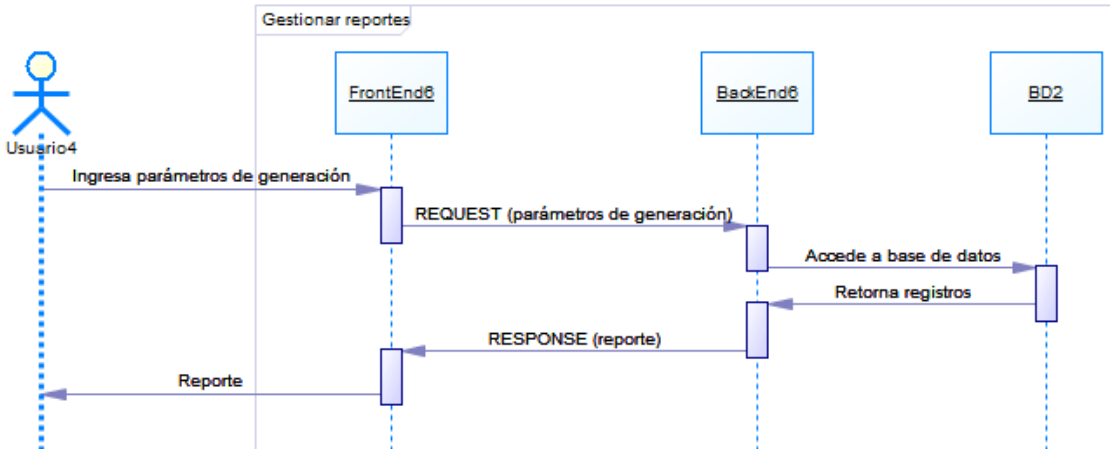


Figura 18. Diagrama de secuencia para gestionar reportes.

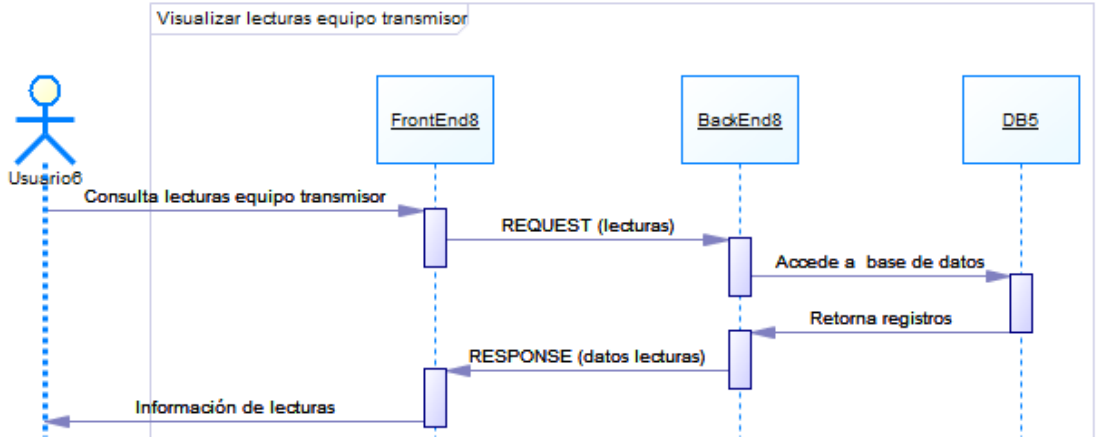


Figura 19. Diagrama de secuencia para visualizar lecturas del equipo transmisor.

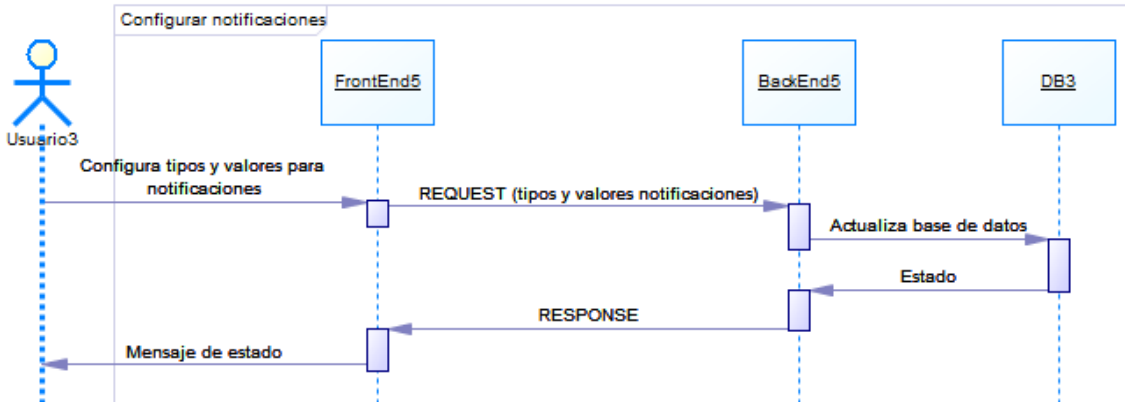


Figura 20. Diagrama de secuencia para configurar notificaciones.

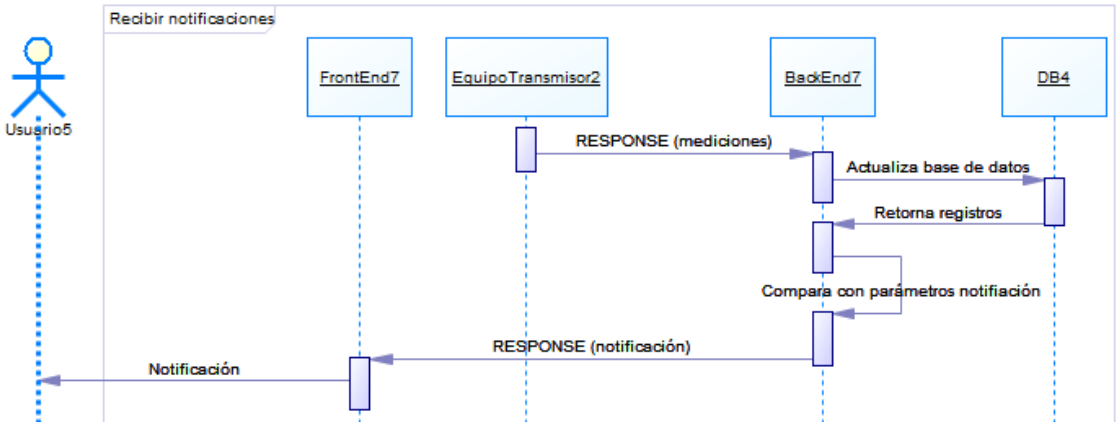


Figura 21. Diagrama de secuencia para recibir notificaciones.

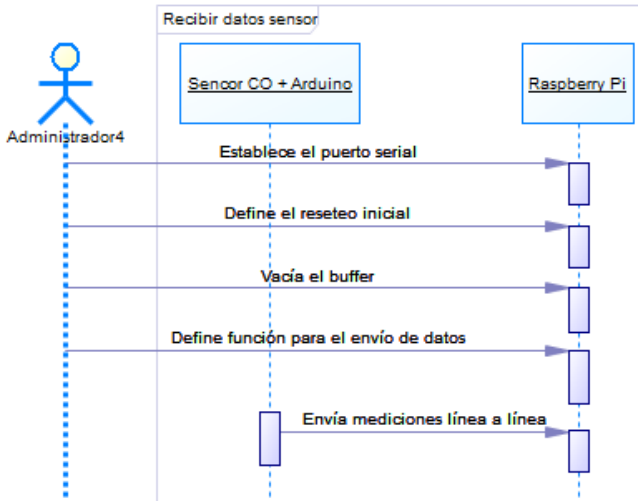


Figura 22. Diagrama de secuencia para recibir datos del sensor.

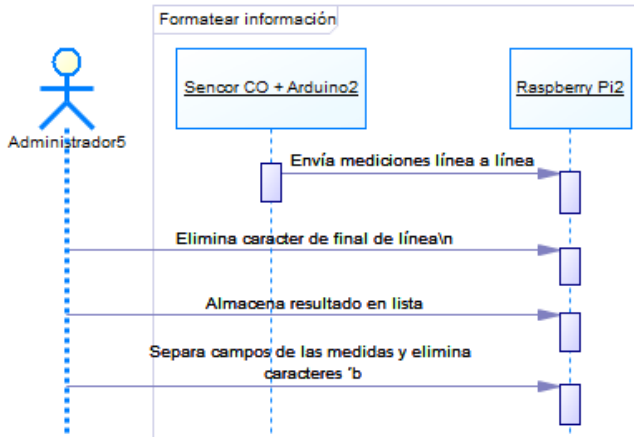


Figura 23. Diagramas de secuencia para formatear información.

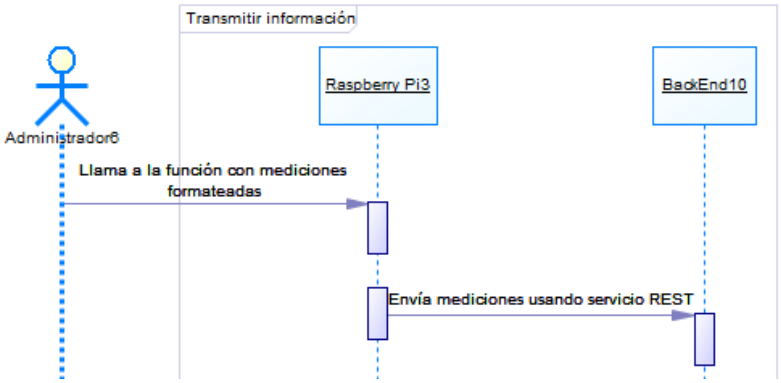


Figura 24. Diagrama de secuencia para transmitir información.

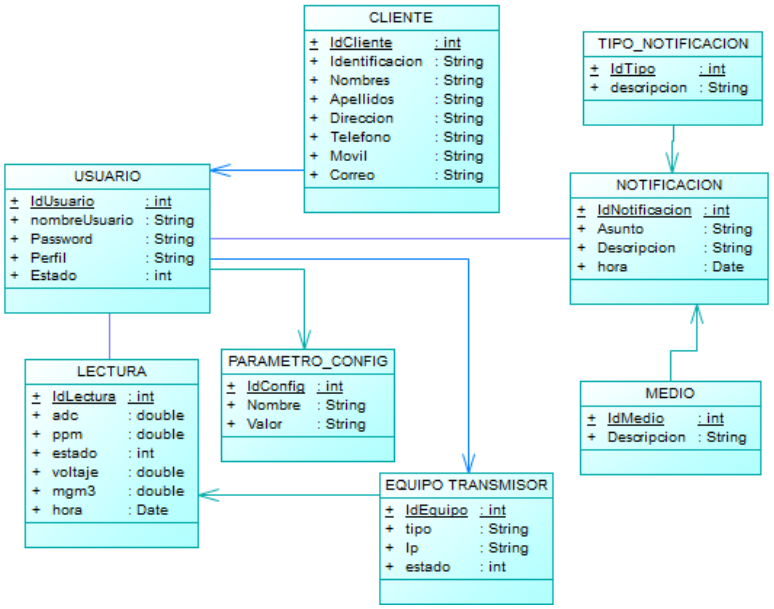


Figura 25. Diagrama de clases para el sistema de detección y registro de niveles de CO de bajo costo en domicilios.

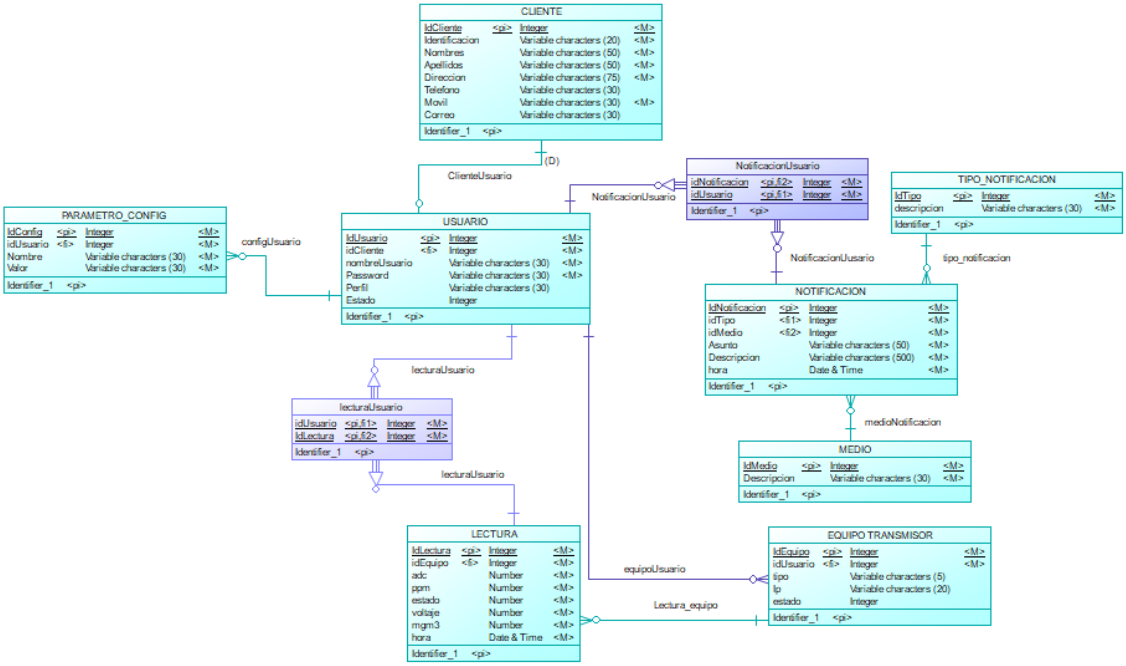


Figura 26. Diagrama de entidad-relación lógico para el sistema de detección y registro de niveles de CO de bajo costo en domicilios.

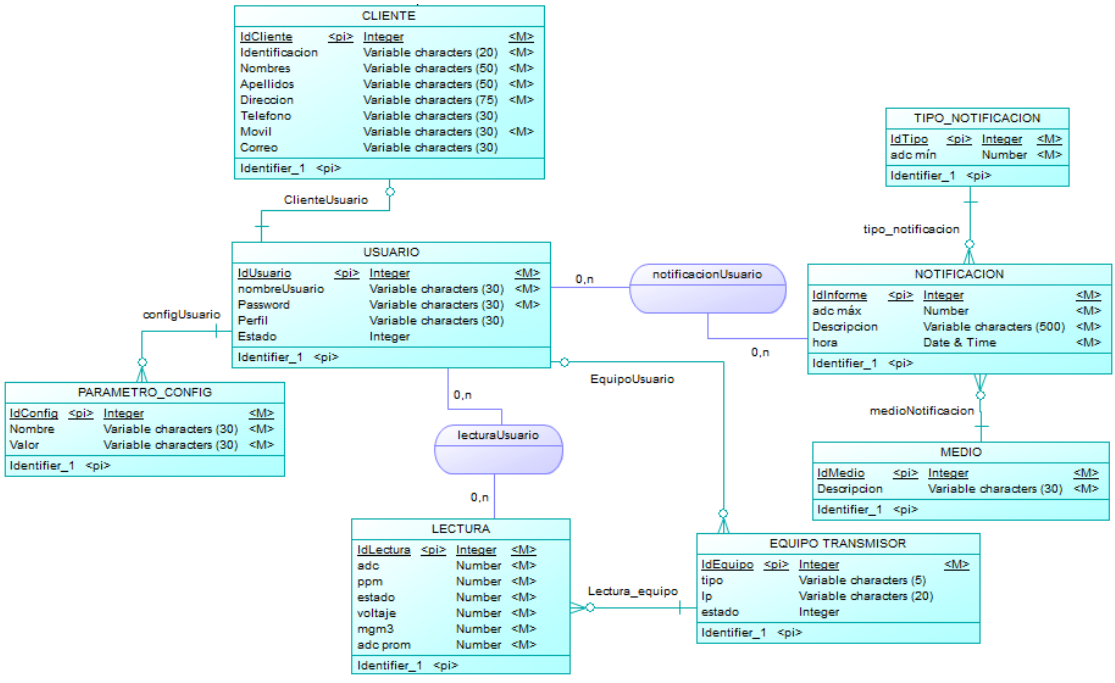


Figura 27. Diagrama de entidad-relación conceptual para el sistema de detección y registro de niveles de CO de bajo costo en domicilios.

Para el diseño e implementación de la aplicación web se consideró el patrón Modelo-Vista-Controlador y el despliegue tanto de un Back-End como de un Front-End.

En la Figura 28, se puede observar la pantalla de Login y en la Figura 29, se muestra la pantalla de Registro con la opción del botón de Ayuda. Esta pantalla está disponible únicamente para un usuario con perfil de administrador. También en la Figura 31, se observa la pantalla para crear un nuevo usuario mediante la opción del botón de Ayuda.

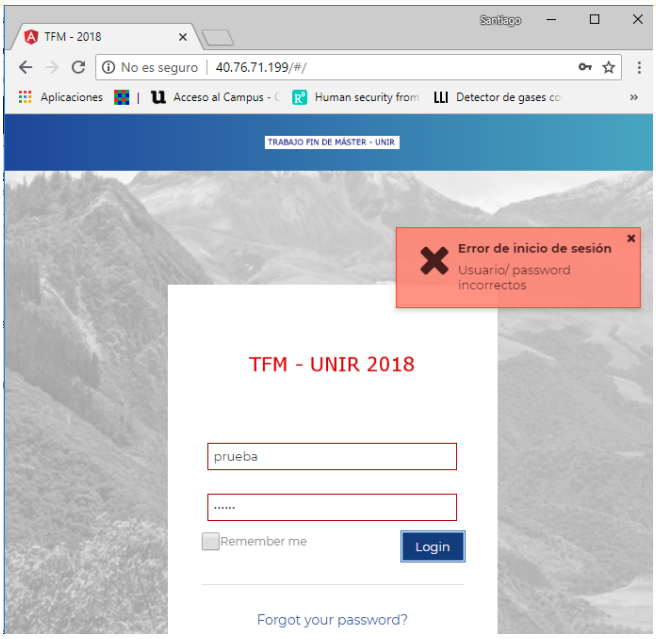
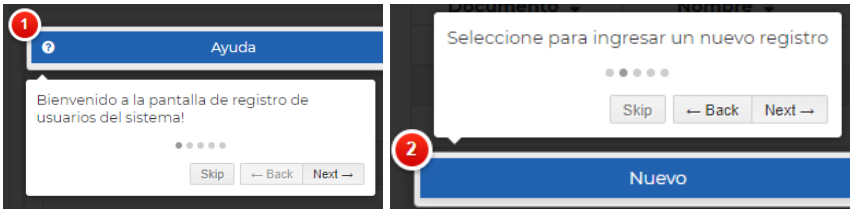
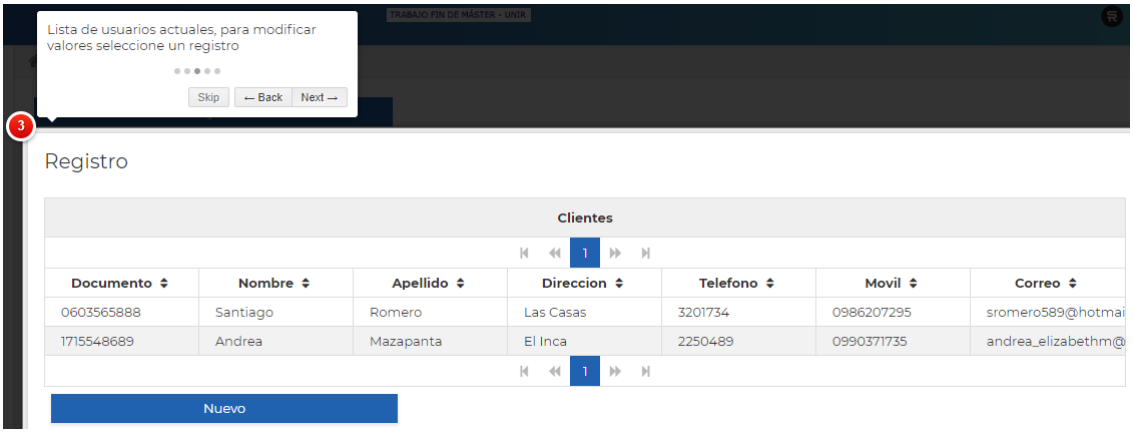


Figura 28. Pantalla de Login.





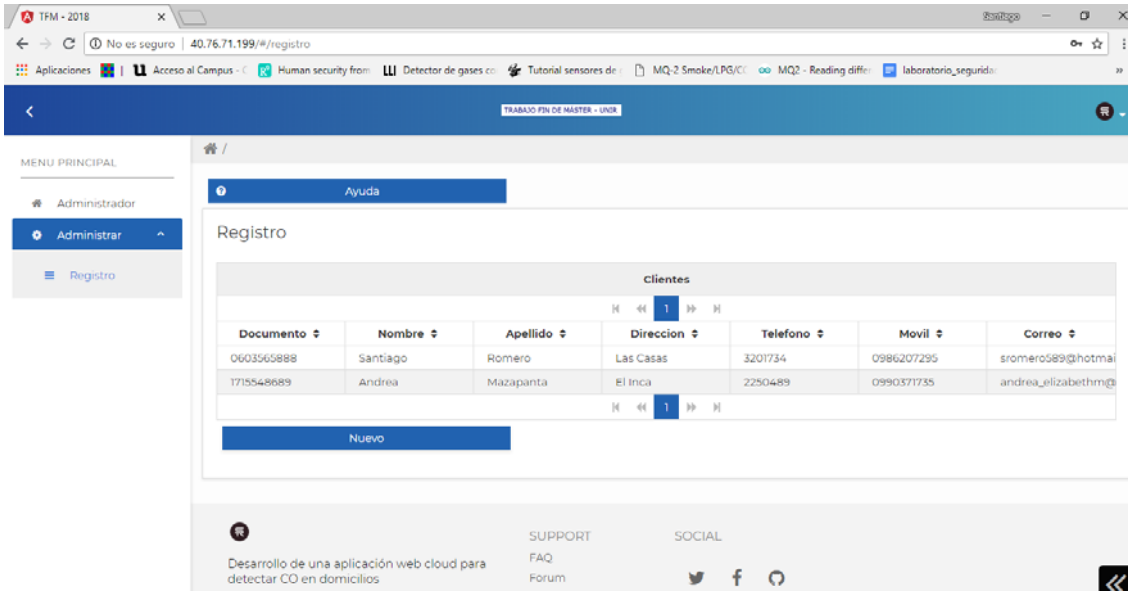


Figura 29. Pantalla de Registro.

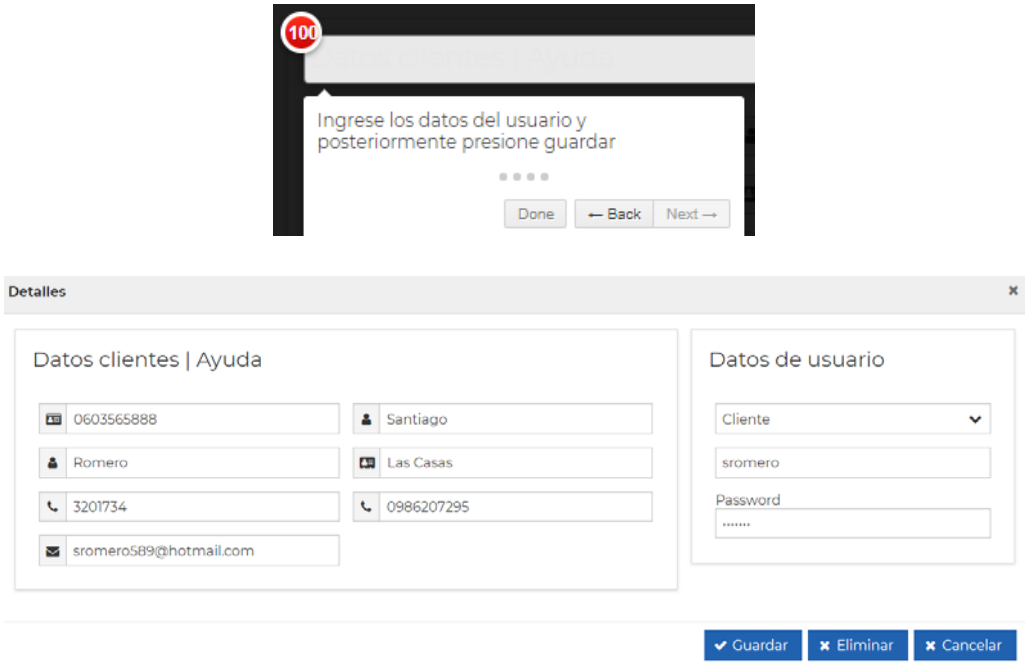


Figura 30. Pantalla de creación de usuarios.

4.2.3 Programación en Python, envío de datos y pruebas.

De acuerdo a los diagramas de secuencia de las Figuras 22, 23 y 24 se ha desarrollado el diagrama de flujo que se observa en la Figura 31.

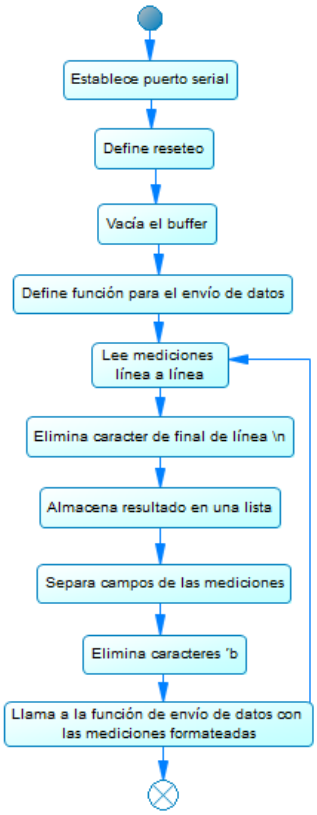
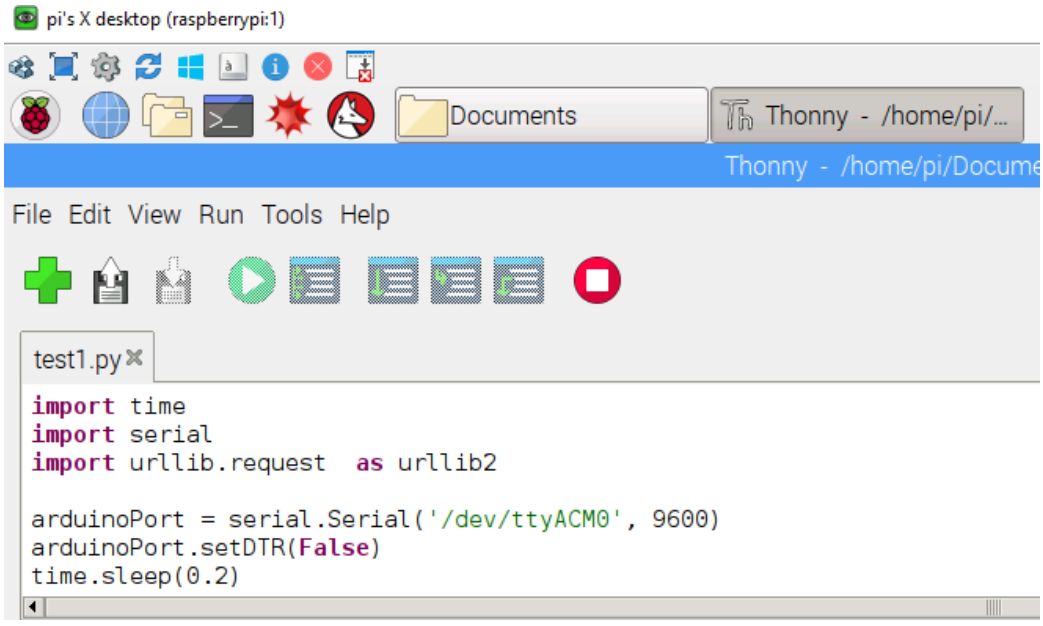


Figura 31. Diagrama de flujo de la transmisión de datos del Raspberry Pi a la aplicación web.

De acuerdo a la Figura 31, en el Raspberry Pi, se codificó en Python un archivo que se encarga de enviar los datos recibidos desde el Arduino a la aplicación web utilizando el método GET de HTTP.

En la Figura 32, se puede apreciar la ejecución del script desarrollado en Python.



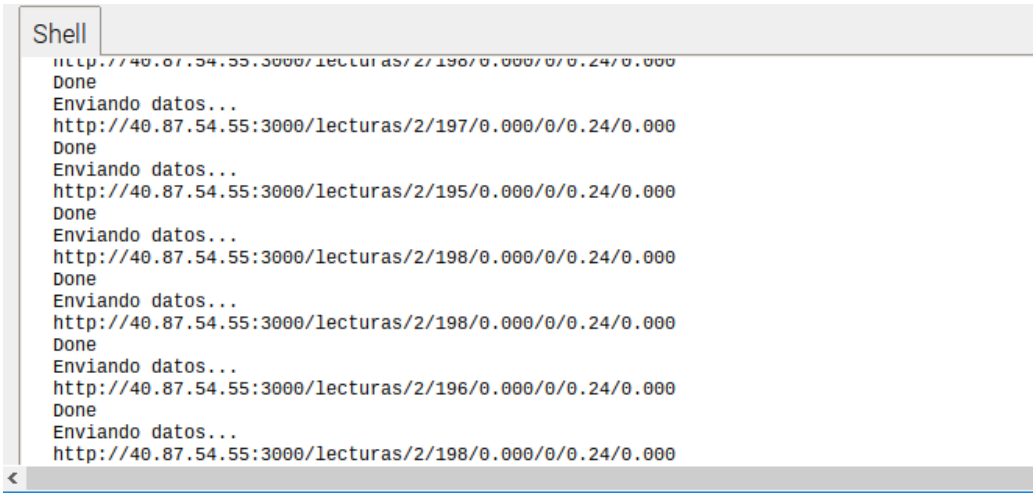


Figura 32. Ejecución del código Pyhton para el envío de datos desde el Raspberry Pi hacia el Back-End de la aplicación web.

En la Figura 33, se puede observar la gráfica de las mediciones realizadas durante un período de tiempo específico utilizando la opción del botón de Ayuda y en la Figura 34, se muestra la tabla de valores para estas mediciones.

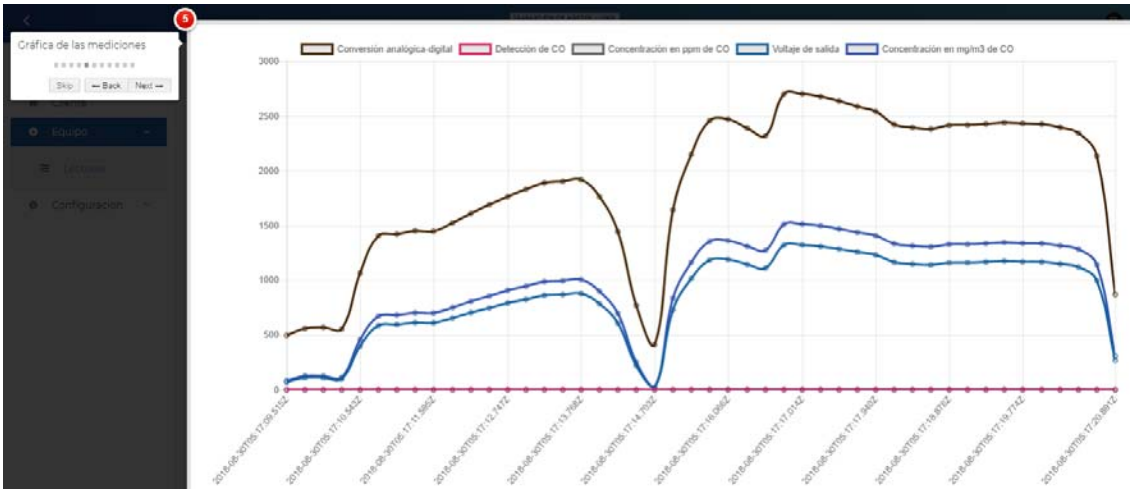
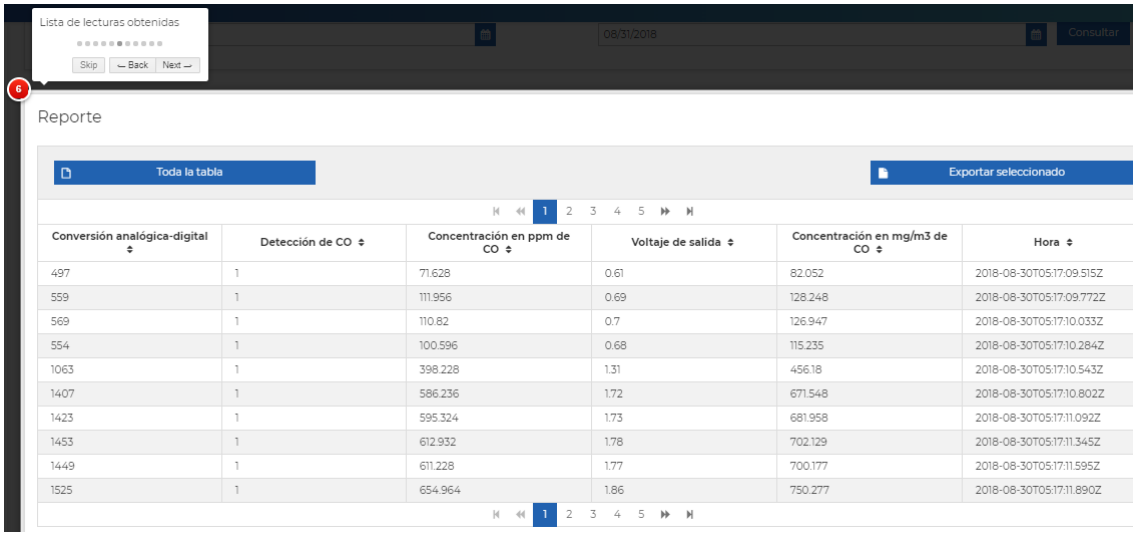


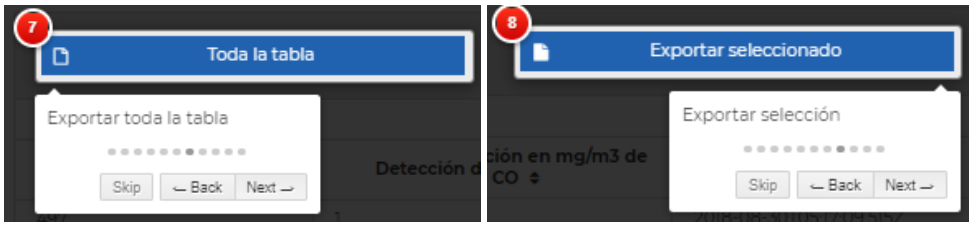
Figura 33. Gráfica de las mediciones realizadas entre el 27 de Agosto y 31 de Agosto del 2018.



Conversión analógica-digital	Detección de CO	Concentración en ppm de CO	Voltaje de salida	Concentración en mg/m3 de CO	Hora
497	1	71.628	0.61	82.052	2018-08-30T05:17:09.515Z
559	1	111.956	0.69	128.248	2018-08-30T05:17:09.772Z
569	1	110.82	0.7	126.947	2018-08-30T05:17:10.033Z
554	1	100.596	0.68	115.235	2018-08-30T05:17:10.284Z
1063	1	398.228	1.31	456.18	2018-08-30T05:17:10.543Z
1407	1	586.236	1.72	671.548	2018-08-30T05:17:10.802Z
1423	1	595.324	1.73	681.958	2018-08-30T05:17:11.092Z
1453	1	612.932	1.78	702.129	2018-08-30T05:17:11.345Z
1449	1	611.228	1.77	700.177	2018-08-30T05:17:11.595Z
1525	1	654.964	1.86	750.277	2018-08-30T05:17:11.890Z

Figura 34. Tabla de valores para las mediciones realizadas.

Es importante mencionar que existe la posibilidad de que todas las mediciones registradas o aquellas seleccionadas por el usuario puedan ser exportadas en formato .CSV para realizar un análisis riguroso de los datos como se observa en la Figura 35.



	A	B	C	D	E	F
1	Conversión analógica-digital	Detección de CO	Concentración en ppm de CO	Voltaje de salida	Concentración en mg/m3 de CO	Hora
2	1352	1	557.268	1.65	638.364	2018-08-26T21:52:35.433Z
3	1457	1	614.636	1.78	704.081	2018-08-26T21:52:35.705Z
4	1496	1	636.788	1.83	729.456	2018-08-26T21:52:35.925Z
5	1548	1	665.756	1.89	762.64	2018-08-26T21:52:36.141Z
6	1595	1	694.156	1.95	795.173	2018-08-26T21:52:36.358Z
7	1671	1	737.324	2.04	844.623	2018-08-26T21:52:36.573Z
8	1733	1	771.404	2.12	883.662	2018-08-26T21:52:36.789Z
9	1788	1	802.644	2.18	919.448	2018-08-26T21:52:37.011Z
10	1829	1	827.068	2.23	947.427	2018-08-26T21:52:37.229Z
11	1870	1	849.788	2.28	973.453	2018-08-26T21:52:37.449Z
12	1908	1	870.804	2.33	997.527	2018-08-26T21:52:37.662Z
13	1940	1	888.412	2.37	1017.698	2018-08-26T21:52:37.878Z
14	1969	1	906.02	2.4	1037.868	2018-08-26T21:52:38.092Z
15	1994	1	919.652	2.43	1053.484	2018-08-26T21:52:38.316Z
16	2015	1	932.716	2.46	1068.449	2018-08-26T21:52:38.546Z
17	2038	1	944.076	2.49	1081.462	2018-08-26T21:52:38.764Z
18	2053	1	953.164	2.51	1091.873	2018-08-26T21:52:38.986Z
19	2069	1	962.82	2.53	1102.934	2018-08-26T21:52:39.209Z
20	2078	1	967.932	2.54	1108.79	2018-08-26T21:52:39.423Z
21	2089	1	973.612	2.55	1115.296	2018-08-26T21:52:39.642Z
22	2095	1	977.588	2.56	1119.851	2018-08-26T21:52:39.860Z
23	2096	1	977.02	2.56	1119.2	2018-08-26T21:52:40.075Z

download (1)

Figura 35. Exportación en formato .CSV de las mediciones realizadas.

4.2.4 Configuración de notificaciones y envío de mensajes.

De acuerdo a [31], se puede mencionar que el monóxido de carbono CO es un gas incoloro, inodoro e insípido, resultado de la oxidación incompleta del carbono durante los procesos de combustión. Los principales efectos sobre la salud son:

- La hipoxia producida por la inhalación de CO, puede afectar al corazón, cerebro, plaquetas y endotelio de los vasos sanguíneos.
- Disminución de la percepción visual, capacidad de trabajo, destreza manual y habilidad de aprendizaje.

En la Tabla 11 se puede apreciar los valores máximos permitidos para el monóxido de carbono a 25 °C de temperatura y a 760 mmHg de presión.

Tabla 11. Resumen de la NECA (Norma Ecuatoriana de la Calidad del Aire)

Contaminante	Valor	Unidad	Período de medición	Excedencia permitida
CO	10	mg/m3	Concentración en 8 horas consecutivas	1 vez por año
	30	mg/m3	Concentración máxima en 1 hora	1 vez por año

[31]

En la Tabla 12 se observan las concentraciones del contaminante monóxido de carbono que definen los niveles de alerta, alarma y emergencia en la calidad de aire de acuerdo al literal 4.1.3.2 de la NECA.

Tabla 12. Niveles de alerta, alarma y emergencia para CO.

Contaminante y período de tiempo	Alerta	Alarma	Emergencia
Concentración promedio de monóxido de carbono en ocho horas (ug/m3)	15000	30000	40000

[32]

Según [31], un indicador ambiental importante es el Índice Quiteño de la Calidad del Aire (IQCA), el cual es una escala numérica entre 0 y 500 que presenta rangos intermedios expresados en diferentes colores. En la Tabla 13 se detallan estos rangos, las categorías del IQCA y los valores límites considerando:

- El nivel deseable u óptimo corresponde al 50% del valor máximo para un período de medición de 8 horas consecutivas de la Tabla 11.
- El nivel aceptable o bueno equivale al 100% del valor máximo para un período de medición de 8 horas consecutivas de la Tabla 11.
- El nivel de precaución no indica la ocurrencia de una condición crítica de contaminación pero muestra una excedencia a reportar.
- Los niveles de alerta, alarma y emergencia están acorde a los valores de la Tabla 12.

Tabla 13. Categorías IQCA.

Rango	Categoría	CO (ug/m3)	Condición desde el punto de vista de la salud
0-50	Nivel deseable u óptimo	0-5000	Óptima
51-100	Nivel aceptable o bueno	5001-10000	Buena
101-200	Nivel de precaución	10001-15000	No saludable para individuos extremadamente sensibles (enfermos crónicos y convalecientes)
201-300	Nivel de alerta	15001-30000	No saludable para individuos sensibles (enfermos)
301-400	Nivel de alarma	30001-40000	No saludable para la mayoría de la población y peligrosa para individuos sensibles
401-500	Nivel de emergencia	>40000	Peligrosa para toda la población

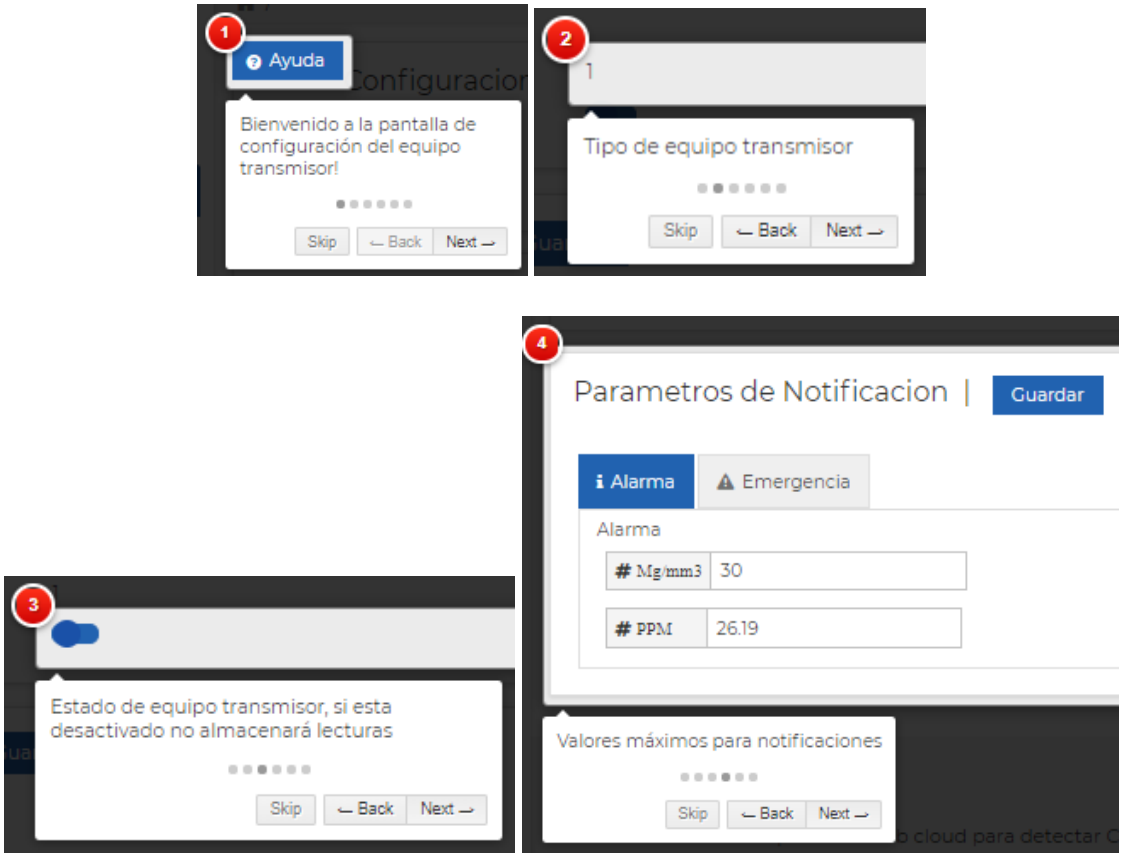
[31]

En base a las categorías de nivel de alarma y emergencia de la Tabla 13 y considerando las lecturas recibidas por parte del equipo transmisor, es posible configurar los parámetros de generación de las notificaciones y enviar estas notificaciones al usuario.

En la Figura 36, se puede observar la pantalla de Configuración del Equipo y de los Parámetros de las Notificaciones que cada usuario puede definir de acuerdo a su necesidad utilizando la opción de Ayuda.

En cambio, en la Figura 37 se muestran las notificaciones recibidas en la propia aplicación web una vez que se han sobrepasado los límites máximos permitidos y se ha generado un mensaje de alarma o emergencia.

En la Figura 38 se observa el e-mail enviado por parte de la aplicación web a la cuenta de correo del cliente una vez que se han cumplido las condiciones para la generación de notificaciones.



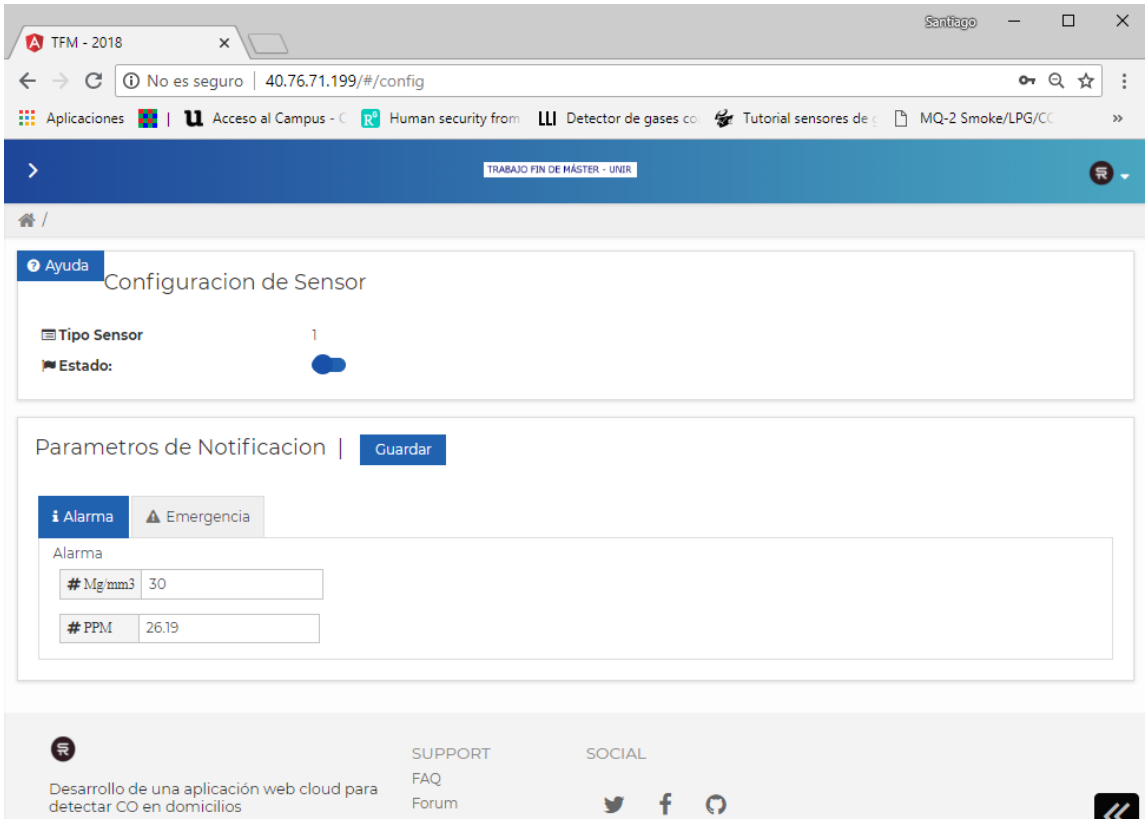


Figura 36. Pantalla de configuración del equipo y de los parámetros de notificación.

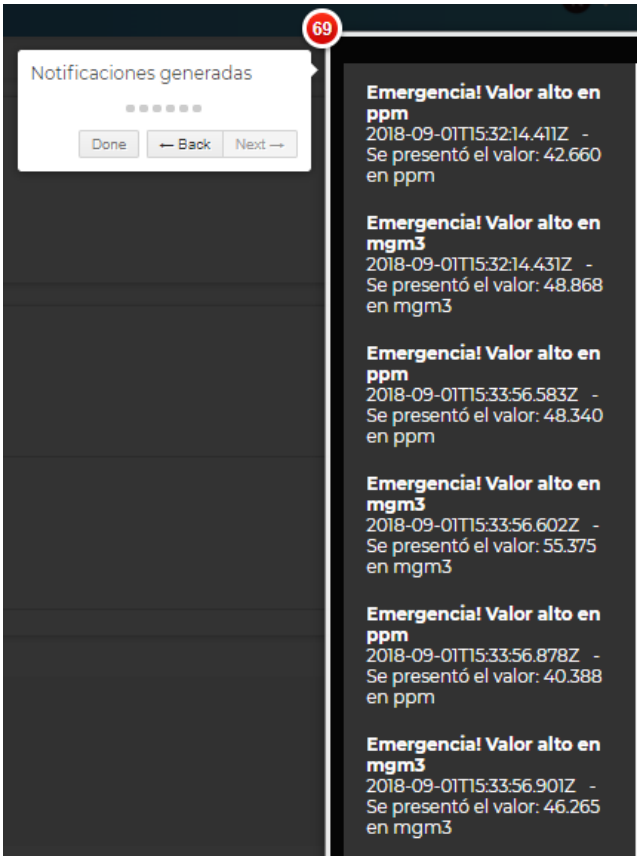


Figura 37. Notificaciones recibidas en el perfil del cliente dentro de la aplicación web.

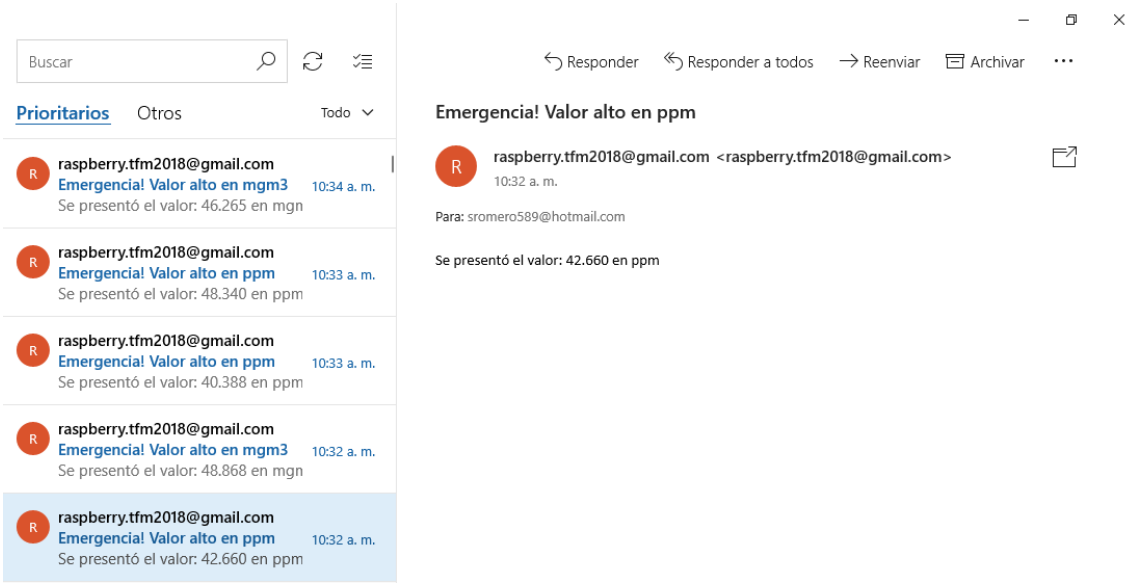


Figura 38. Notificaciones enviadas por la aplicación web vía e-mail a la cuenta de correo del cliente.

4.2.5 Generación de reportes.

Es posible obtener los valores máximos, mínimos y promedios para las mediciones realizadas en un determinado período de consulta. Estos datos son muy importantes ya que mediante una gráfica se puede analizar el comportamiento de cada una de las magnitudes en el transcurso del tiempo.

La tabla de valores se muestra en la Figura 39 y la gráfica correspondiente a estos valores se observa en la Figura 40 mediante el uso de la opción de Ayuda.

Figura 39. Tabla de valores máximos, mínimos y promedios.

Figura 40. Gráfica de valores máximos, mínimos y promedios.

En la Tabla 14 se detalla el costo referencial del sistema de detección y registro de niveles de CO en domicilios.

Tabla 14. Costo referencial del sistema de detección y registro de niveles de CO en domicilios.

SISTEMA DE DETECCIÓN Y REGISTRO DE NIVELES DE CO EN DOMICILIOS			
EQUIPO TRANSMISOR			
ÍTEM	**CANTIDAD**	**COSTO UNITARIO ($)**	**COSTO TOTAL ($)**
Arduino Mega 2560 R3 + Cable USB	1	20	20
Raspberry Pi 3 + Cargador 5V 3A + Case + Ventilador	1	85	85
Sensor de CO MQ-7	1	5	5
Cables para conexión del sensor de CO a la placa Arduino	4	0.25	1
Caja de madera para el equipo transmisor	1	10	10
Instalación del equipo	1	10	10
SUBTOTAL 1			**131**

TRABAJO DE INGENIERÍA Y DESPLIGUE EN LA NUBE			
ÍTEM	**MESES**	**COSTO UNITARIO ($)**	**COSTO TOTAL ($)**
Trabajo de Ingeniería	6 (Abril-Septiembre)	385	2310
Pago en Microsoft Azure por el despliegue de la aplicación web en la nube	6 (Abril-Septiembre)	25	150
SUBTOTAL 2			**2460**
TOTAL (SUBTOTAL 1 + SUBTOTAL 2)			**2591**

4.3. Evaluación

Es importante mencionar que durante la realización de las pruebas de cada uno de los componentes de la aplicación web en la fase de implementación, el equipo transmisor fue instalado y continúa funcionando como se puede observar en la Figura 41.

Figura 41. Equipo Transmisor instalado y operativo.

Cabe recalcar que la aplicación web es capaz de trabajar con muchos más dispositivos instalados en cualquier localidad que cuente con cobertura WiFi e incluso diferentes prototipos.

Para evidenciar la satisfacción de los requisitos no funcionales se realizó una encuesta sobre diferentes aspectos de la aplicación web a funcionarios del área de TIC de la Unidad de Negocio CELEC EP Coca Codo Sinclair.

Con el objetivo de que en cada perfil de usuario se disponga de registros de lecturas a consultar, se realizó el envío de datos desde el equipo transmisor cambiando el parámetro

equipo para que los usuarios puedan probar todas las funcionalidades del sistema. También se otorgó un perfil de administrador común para que esta opción de la aplicación web también pueda ser verificada.

En el Anexo II se detallan las preguntas que conforman la Encuesta. Los resultados obtenidos de la realización de la Encuesta se muestran a partir de la Figura 42 hasta la Figura 50.

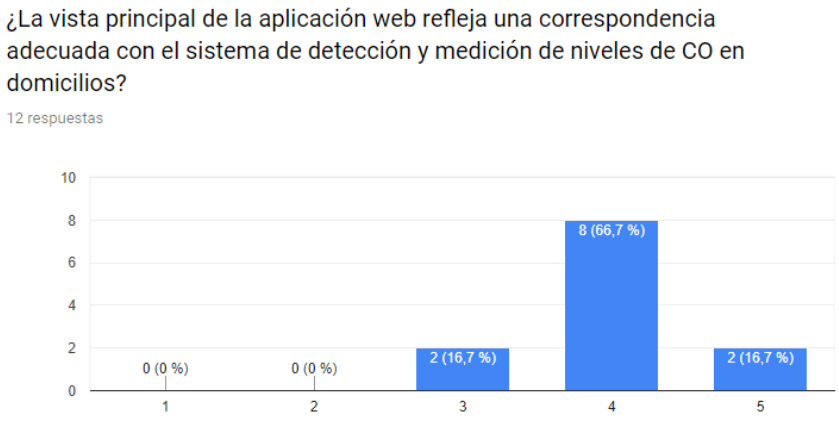


Figura 42. Identidad de la aplicación web.

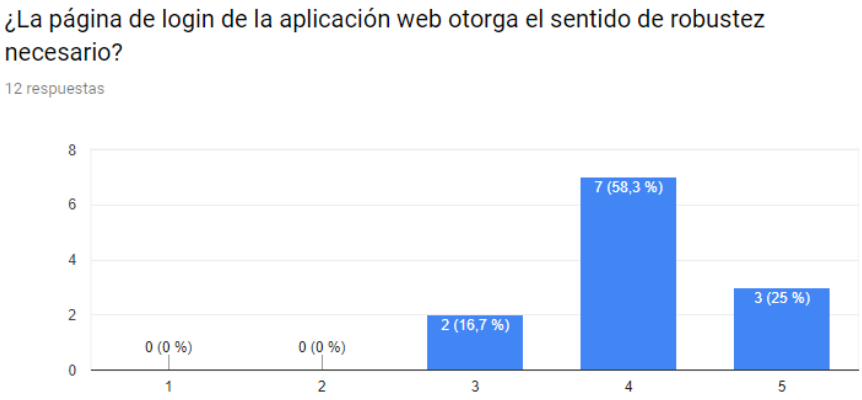


Figura 43. Seguridad de la aplicación web.

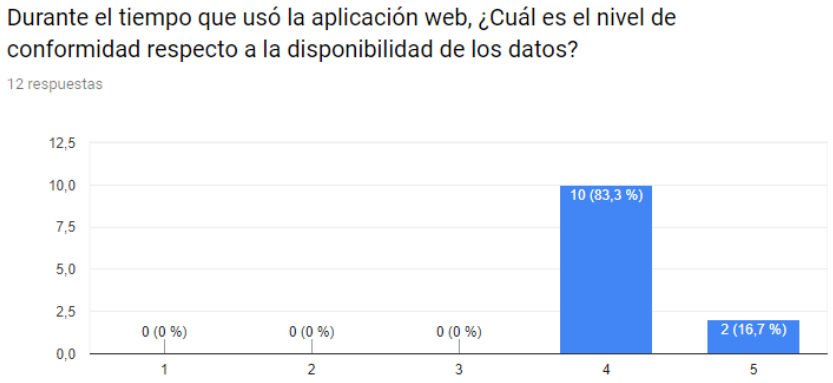


Figura 44. Disponibilidad de la aplicación web.

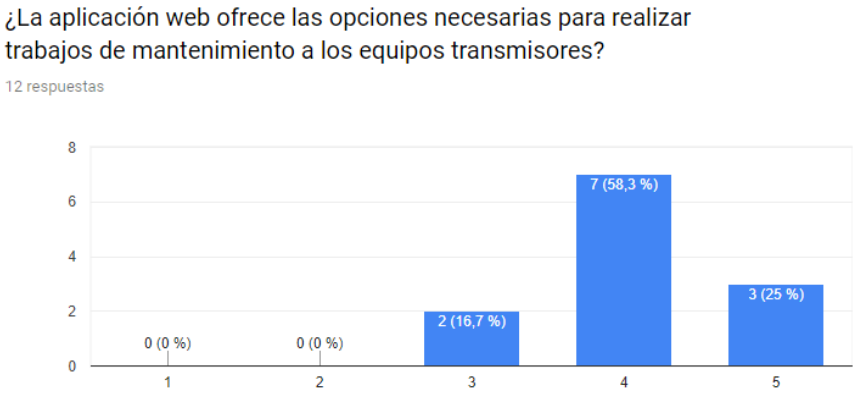


Figura 45. Mantenimiento de la aplicación web.

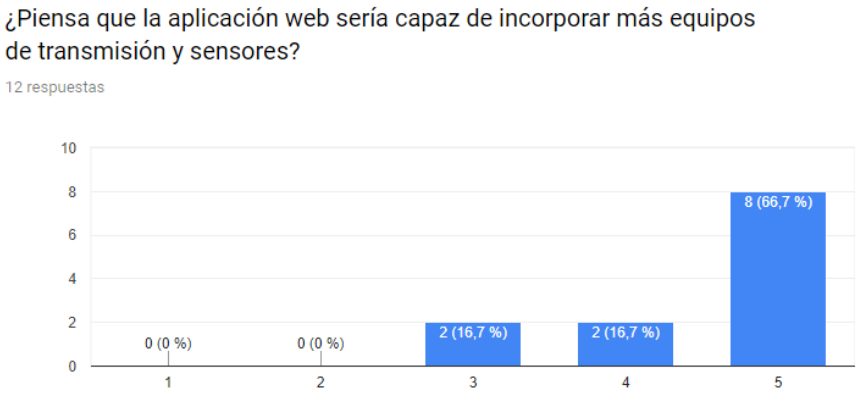


Figura 46. Escalabilidad de la aplicación web.

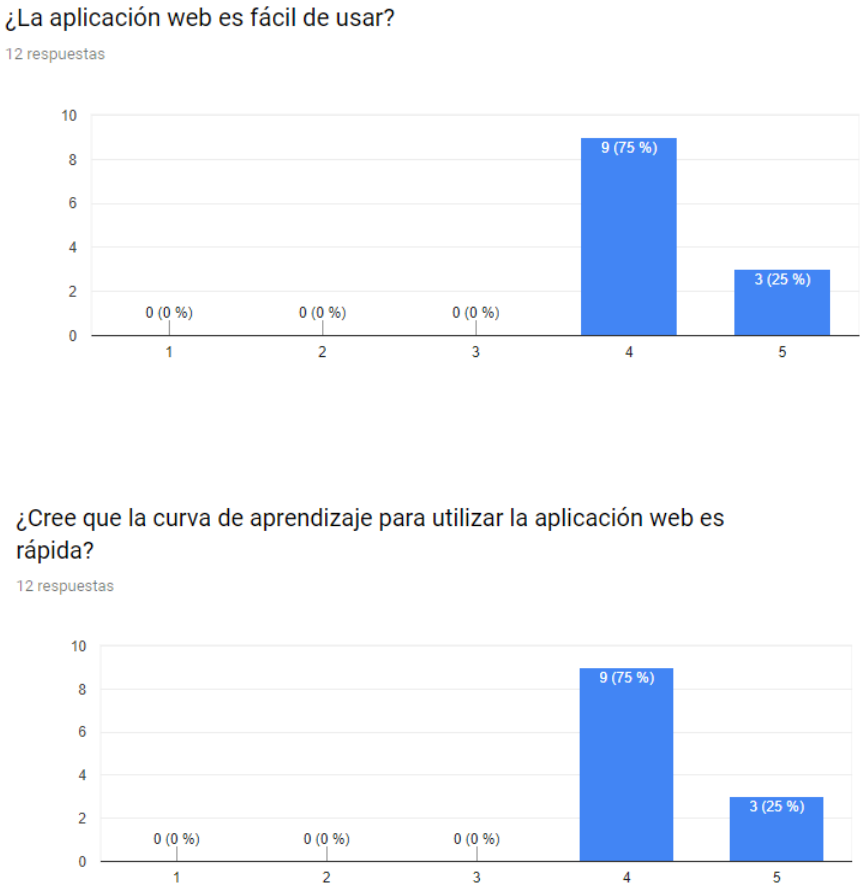
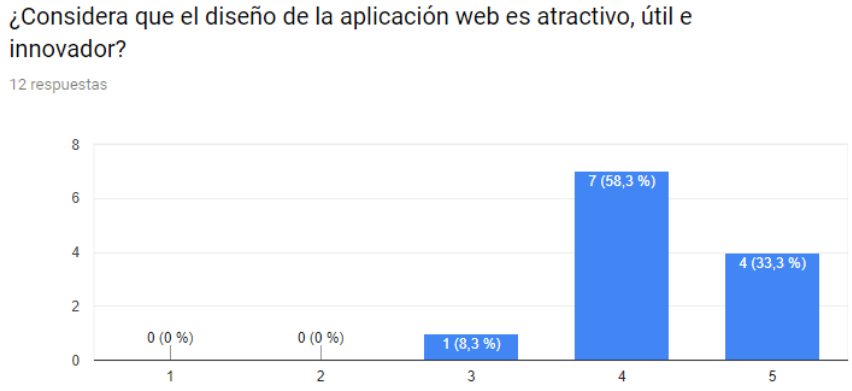


Figura 47. Usabilidad de la aplicación web.



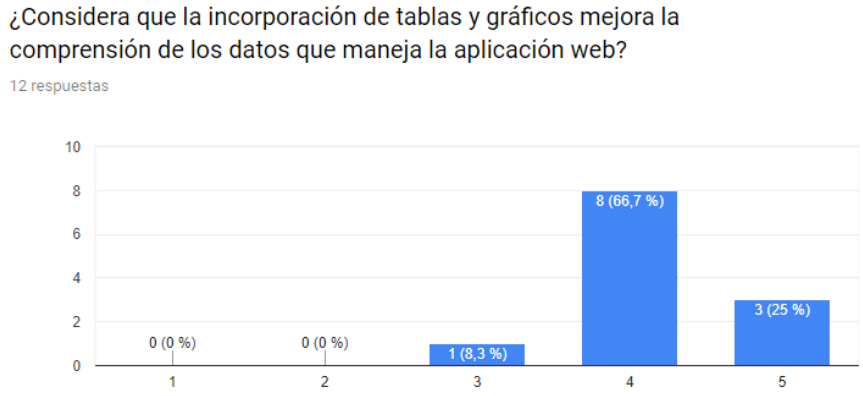
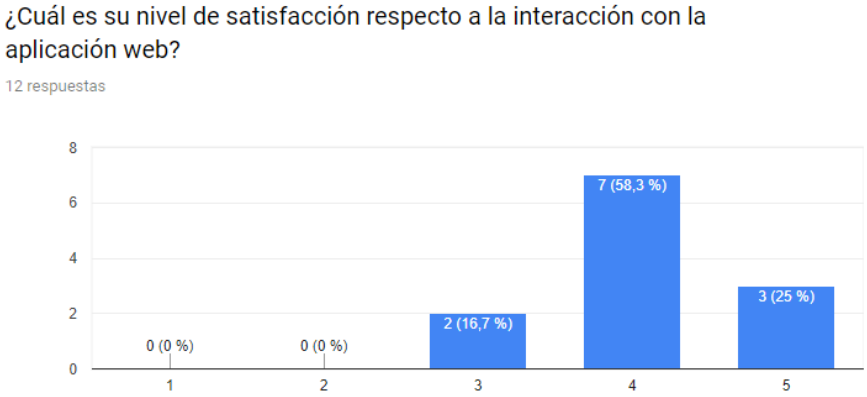
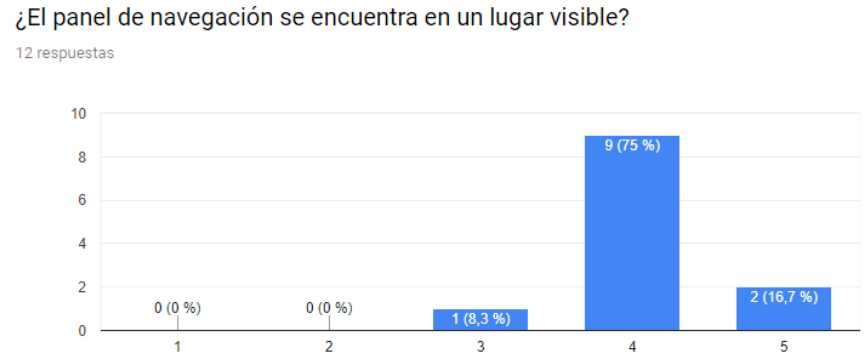


Figura 48. Diseño de la aplicación web.





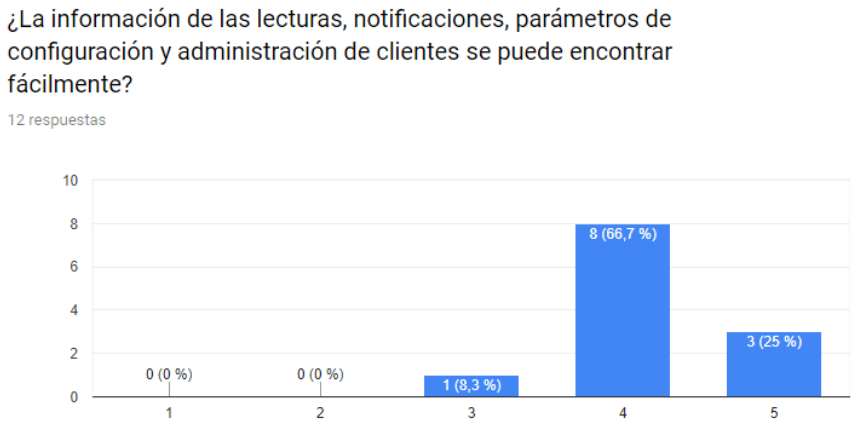


Figura 49. Navegación de la aplicación web.

Figura 50. Recomendación acerca del uso de la aplicación web.

5. Conclusiones y trabajo futuro

5.1. Conclusiones

- Se pudo implementar un sistema de detección y registro de niveles de CO de bajo costo en domicilios usando hardware y software libre, tecnologías web y servicios cloud precautelando la salud de las personas y evitando la pérdida de vidas humanas debido a la inhalación del gas tóxico monóxido de carbono.
- En base al análisis de los niveles de CO que son perjudiciales para la salud de las personas en domicilios, se logró configurar el envío de notificaciones de alerta y emergencia a los usuarios mediante la conexión con el servicio de correo electrónico de Gmail a fin de que puedan tomar las medidas de prevención necesarias.

- Se logró en su gran mayoría seguir la tendencia “free” ya que se utilizó dispositivos de hardware libre para la lectura, procesamiento y transmisión de los niveles de CO en el entorno a la vez que se empleó software libre tanto en la infraestructura necesaria para el desarrollo de la app web como en la propia codificación.

- Se diseñó e implementó una aplicación web que es capaz de recibir los datos transmitidos por el sensor de CO, cumple con el patrón MVC, satisface los requisitos de seguridad en cuanto a control de acceso, permite la visualización de las mediciones y emite notificaciones de alerta y emergencia en caso de que se superen los valores umbrales definidos salvaguardando la vida de las personas en domicilios.

- Se generaron reportes personalizados considerando la identificación de los equipos transmisores, fecha de registro de los datos y variables medidas.

- Luego de haber realizado las pruebas de evaluación tanto a nivel de sistema como en la medición de las variables eléctricas y ambientales, envío de datos a la aplicación web y validación del funcionamiento de la aplicación web se puede manifestar que se ha cumplido con los objetivos fijados y cubriendo los requisitos funcionales y no funcionales.

- Al analizar los resultados obtenidos por la Encuesta realizada se puede decir que los requisitos no funcionales como la seguridad, disponibilidad, mantenimiento, usabilidad y diseño de la aplicación web alcanzaron un nivel 4, es decir, de satisfacción. Respecto a la escalabilidad de la aplicación web se puede manifestar que alcanzó un nivel de 5, es decir, muy satisfecho. También es importante señalar que el 92 % de los encuestados expresó que recomendaría el uso de la aplicación web.

5.2. Líneas de trabajo futuro

Una posible área de trabajo de acuerdo al proyecto realizado sería implementar todo el sistema en el dispositivo de hardware libre Raspberry Pi a excepción de la conversión analógica a digital realizada por la placa de desarrollo de hardware Arduino, la cual se mantendría conectada al sensor de CO. Es decir, en el Raspberry Pi se montaría el servidor web, un entorno de ejecución con todos los frameworks requeridos, un sistema gestor de base de datos asociado y se configuraría de tal forma para que pueda servir contenido a través de Internet. La aplicación web podría ser accesible mediante una IP pública, el uso de direccionamiento NAT y el empleo de un firewall a nivel de red, transporte y aplicación. Además, se podría añadir más sensores para el control del entorno en interiores como temperatura, vibraciones, etc.

Otra posible área de trabajo que se base en el presente proyecto sería de implementar todo el sistema usando únicamente el sensor de CO y la placa Arduino dejando a un lado el uso de la plataforma Raspberry Pi. Toda la aplicación web e infraestructura tendría que ser desplegada en la placa Arduino.

6. Bibliografía

[1] Brendan O'Brien, "Why the 'Internet of Things' is Important", ARIA [En línea]. Disponible en: https://www.ariasystems.com/blog/internet-things-important/. [Accedido: 09-may-2018]

[2] ARC Advisory Group, "IoT and Analytics Can Add Smartness to Fire Detection", 2017 [En línea]. Disponible en: https://industrial-iot.com/2017/08/iot-analytics-smart-fire-detection/. [Accedido: 10-may-2018]

[3] Anandhakrishnan S, Deepesh Nair, Rakesh K, Sampath K,Gayathri S Nair, "IOT Based Smart Gas Monitoring System", National Conference on "Emerging Research Trends in Electrical, Electronics & Instrumentation", pp. 82-87, 2017. [En línea]. Disponible en: http://www.iosrjournals.org/iosr-jeee/Papers/Conf.17017/Volume-3/13.%2082-87.pdf. [Accedido: 11-may-2018]

[4] Dr. Chalasani Srinivas, Mohan Kumar.Ch, "Toxic gas detection and monitoring utilizing Internet of Things", International Journal of Civil Engineering and Technology (IJCIET), Volume 8, Issue 12, December 2017, pp. 614–622. [En línea]. Disponible en: http://www.iaeme.com/MasterAdmin/UploadFolder/IJCIET_08_12_067/IJCIET_08_12_067.pdf. [Accedido: 12-may-2018]

[5] Leanne McRae, Katie Ellis and Mike Kent, "The Internet of Things (IoT): Education and Technology", Curtin University, 2018. [En línea]. Disponible en: https://www.ncsehe.edu.au/wp-content/uploads/2018/02/IoTEducation_Formatted_Accessible.pdf. [Accedido: 13-may-2018]

[6] Alvaro Everlet, Javier Pastor, Introducción al Internet de las Cosas, Construyendo un Proyecto de IOT, Carriots, Universidad Rey Juan Carlos, Noviembre 2013. [En línea]. Disponible en: https://www.carriots.com/newFrontend/img-carriots/press_room/Construyendo_un_proyecto_de_IOT.pdf. [Accedido: 14-may-2018]

[7] W3schools, "Node.js Introduction". [En línea]. Disponible en: https://www.w3schools.com/nodejs/nodejs_intro.asp. [Accedido: 15-may-2018]

[8] Kiran Malvi, "The Positive and Negative Aspects of Node.js Web App Development", 18 de Junio 2018. [En línea]. Disponible en: https://www.mindinventory.com/blog/pros-and-cons-of-node-js-web-app-development/. [Accedido: 16-may-2018]

[9] MDN web docs, "Express/Node introduction". [En línea]. Disponible en: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction. [Accedido: 16-may-2018]

[10] Tymets Volodymyr, "Express.js Mobile App Development: Pros and Cons for Developers", 08 de Junio 2017. [En línea]. Disponible en: https://apiko.com/blog/express-mobile-app-development/. [Accedido: 16-may-2018]

[11] AngularJS, "What Is AngularJS?". [En línea]. Disponible en: https://docs.angularjs.org/guide/introduction. [Accedido: 17-may-2018]

[12] AltexSoft, "The Good and the Bad of Angular Development", 29 de Septiembre 2017. [En línea]. Disponible en: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/. [Accedido: 17-may-2018]

[13] MongoDB, "Introduction to MongoDB". [En línea]. Disponible en: https://docs.mongodb.com/manual/introduction/. [Accedido: 18-may-2018]

[14] Data Flair, "Advantages of MongoDB | Disadvantages of MongoDB", 17 de Abril 2018. [En línea]. Disponible en: https://data-flair.training/blogs/advantages-of-mongodb/. [Accedido: 18-may-2018]

[15] Python, “General Python FAQ”. [En línea]. Disponible en: https://docs.python.org/3/faq/general.html#id2. [Accedido: 19-may-2018]
[16] DataFlair, “Advantages and Disadvantages of Python Programming Language”, Enero 2018. [En línea]. Disponible en: https://data-flair.training/blogs/advantages-and-disadvantages-of-python/. [Accedido: 19-may-2018]
[17] Ajay, Dr. Baswaraj Gadgay, Veeresh Pujari, Pallavi B.V, “IOT Based Smart Environmental Monitoring Using Arduino”, International Journal for Research in Applied Science & Engineering Technology (IJRASET), Vol. 5, Junio 2017. [En línea]. Disponible en: https://www.ijraset.com/fileserve.php?FID=8144. [Accedido: 20-may-2018]
[18] Dr. Salah Talha Babiker Edris, “Monitoring of Taif University Campus by using The Internet of Things Techniques”, International Journal of Applied Engineering Research, Vol. 12, 2017, pp. 14377-14381. [En línea]. Disponible en: https://www.ripublication.com/ijaer17/ijaerv12n24_53.pdf. [Accedido: 20-may-2018]
[19] Jesús Castillo Izquierdo, Diseño e implementación de un dispositivo IoT de bajo coste para entornos agrícolas, Escuela Tecnica Superior de Ingenierías Informática y de Telecomunicación, Granada, Junio 2017. [En línea]. Disponible en: http://wpd.ugr.es/~jorgenavarro/thesis/2017_TFG_JesusCastilloIzquierdo.pdf. [Accedido: 21-may-2018]
[20] Somansh Kumar, Ashish Jasuja, “Air Quality Monitoring System Based on IoT using Raspberry Pi”, International Conference on Computing, Communication and Automation, 2017. [En línea]. Disponible en: https://zapdf.com/air-quality-monitoring-system-based-on-iot-using-raspberry-p.html. [Accedido: 21-may-2018]
[21] C. Balasubramaniyan, D. Manivannan, “IoT Enabled Air Quality Monitoring System (AQMS) using Raspberry Pi”, Indian Journal of Science and Technology, Vol 9(39), Octubre 2016. Disponible en: http://www.indjst.org/index.php/indjst/article/view/90414/74439. [Accedido: 21-may-2018]
[22] Jesús Castillo Izquierdo, Diseño e implementación de un dispositivo IoT de bajo coste para entornos agrícolas, Escuela Tecnica Superior de Ingenierías Informática y de Telecomunicación, Granada, Junio 2017. [En línea]. Disponible en: http://wpd.ugr.es/~jorgenavarro/thesis/2017_TFG_JesusCastilloIzquierdo.pdf. [Accedido: 21-may-2018]
[23] Roger S. Pressman, Ingeniería del software, Un enfoque práctico, Séptima Edición, Mc Graw Hill, México, D. F., 2010.
[24] Microsoft Azure, Uso de claves SSH con Windows en Azure, Abril 2018. [En línea]. Disponible en: https://docs.microsoft.com/es-es/azure/virtual-machines/linux/ssh-from-windows. [Accedido: 22-may-2018]
[25] Yhorby Matias Heredia, Microsoft Azure, Creando una máquina virtual con Linux, Junio 2016. [En línea]. Disponible en: http://ymatias.com/2016/06/07/microsoft-azure-creando-una-maquina-virtual-con-linux/. [Accedido: 22-may-2018]
[26] DigitalOcean, “How to Install MongoDB on Ubuntu 18.04”, 7 de Junio 2018. [En línea]. Disponible en: https://www.digitalocean.com/community/tutorials/how-to-install-mongodb-on-ubuntu-18-04. [Accedido: 23-may-2018]
[27] DigitalOcean, “How to Install Node.js on Ubuntu 18.04”, 27 de Abril 2018. [En línea]. Disponible en: https://www.digitalocean.com/community/tutorials/how-to-install-node-js-on-ubuntu-18-04. [Accedido: 23-may-2018]
[28] HANWEI ELECTRONICS CO., LTD, “Technical Data MQ-7 Gas Sensor”. Disponible en: https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf. [Accedido: 24-may-2018]
[29] Jorge Anibal Suntaxi Pichuasamin, Diseño y construcción de un prototipo portátil de monitoreo ambiental, mediante un Sistema autónomo de adquisición de datos portátil con comunicación USB hacia un PC, Facultad de Ingeniería Eléctrica y Electrónica, Escuela Politécnica Nacional, Enero 2015.

[30] Mónica Tejerina Fernández, UF 1909 – Toma de muestras de contaminantes atmosféricos, Editorial Eleraning S.L., Edición: 5.1, pp. 267-268, 2015.

[31] Valeria Díaz Suárez, Informe de la calidad del aire del Distrito Metropolitano de Quito, Secretaría de Ambiente, Alcaldía Metropolitana de Quito, Abril 2015, pp. 4-9. [En línea]. Disponible en: http://www.quitoambiente.gob.ec/ambiente/images/Secretaria_Ambiente/red_monitoreo/informacion/iqca_2014.pdf. [Accedido: 26-may-2018]

[32] Ministerio del Ambiente, Norma Ecuatoriana de Calidad del Aire, Libro VI, Anexo 4, Junio 2011, pp. 6. [En línea]. Disponible en: http://www.quitoambiente.gob.ec/ambiente/images/Secretaria_Ambiente/red_monitoreo/informacion/norma_ecuato_calidad.pdf. [Accedido: 26-may-2018]

Anexos

Anexo I. Manual de puesta en marcha de la herramienta

Instalación del IDE ARDUINO

El primer paso es descargar la lista de los paquetes que contienen los repositorios y la información actualizada de las versiones más recientes de estos paquetes mediante el comando que se visualiza en la Figura 51.

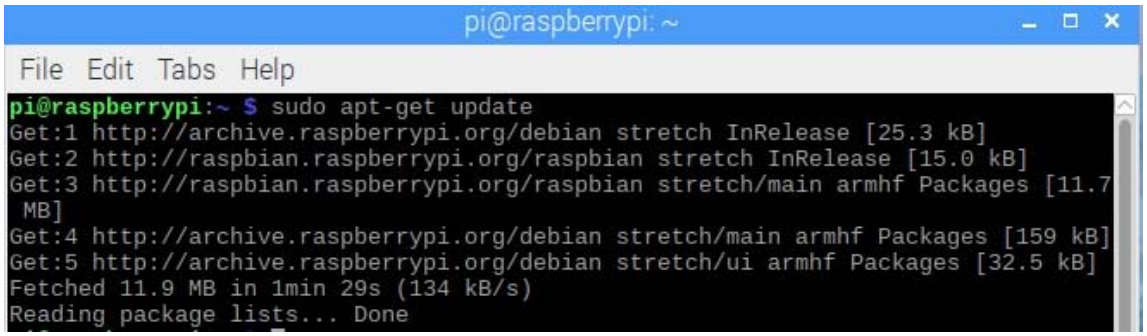


Figura 51. Actualización de la lista de paquetes.

Una vez que se tienen las listas actualizadas de los paquetes, se procede a instalar el IDE de Arduino como se observa en la Figura 52.

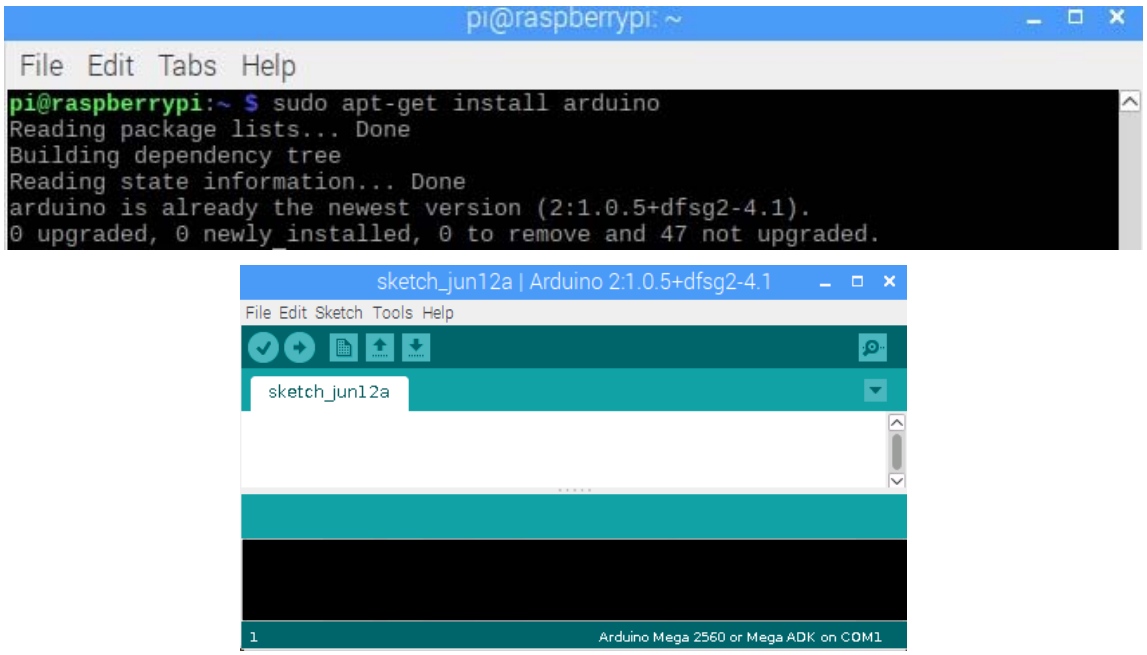


Figura 52. Instalación del IDE Arduino.

Con el propósito de que el Raspberry Pi se pueda comunicar mediante una conexión serial con la plataforma de desarrollo Arduino, es necesario que se agregue el usuario “pi” al terminal serial tty del grupo dialout y que se verifique el puerto de comunicación serial mediante los comandos de la Figura 53.

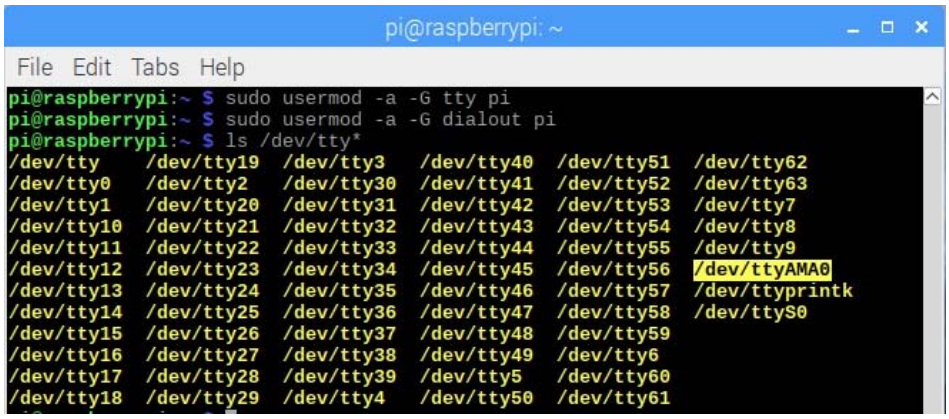


Figura 53. Habilitación y verificación del acceso al puerto serial.

Consideraciones para la conexión del sensor de CO

Una vez que ya se ha instalado el IDE Arduino, se debe analizar en detalle el sensor de monóxido de carbono a utilizar, definir el esquema de conexión con la placa Arduino y programar en este entorno de desarrollo a fin de poder convertir las señales analógicas de entrada en datos digitales concordantes con las unidades de medida de la variables eléctricas y ambientales involucradas.

De acuerdo a [28], se puede mencionar que el sensor MQ-7 presenta una alta sensibilidad al monóxido de carbono, es estable y de larga vida útil, es de tamaño reducido y de muy bajo costo a la vez que se usa en equipamiento para la detección de CO en hogares, en la industria y en automóviles. Un parámetro fundamental es el rango de detección del gas el cual va desde las 20 ppm hasta las 2000 ppm de CO. En cuanto a la conexión del sensor, el fabricante menciona que se debe colocar una resistencia de carga, preferentemente de 10 KΩ, de forma que se implemente un divisor de voltaje entre la resistencia del sensor y esta resistencia de carga logrando así una adecuada medición del monóxido de carbono.

El diagrama de conexión del sensor de CO con la placa Arduino es el que se observa en la Figura 54.

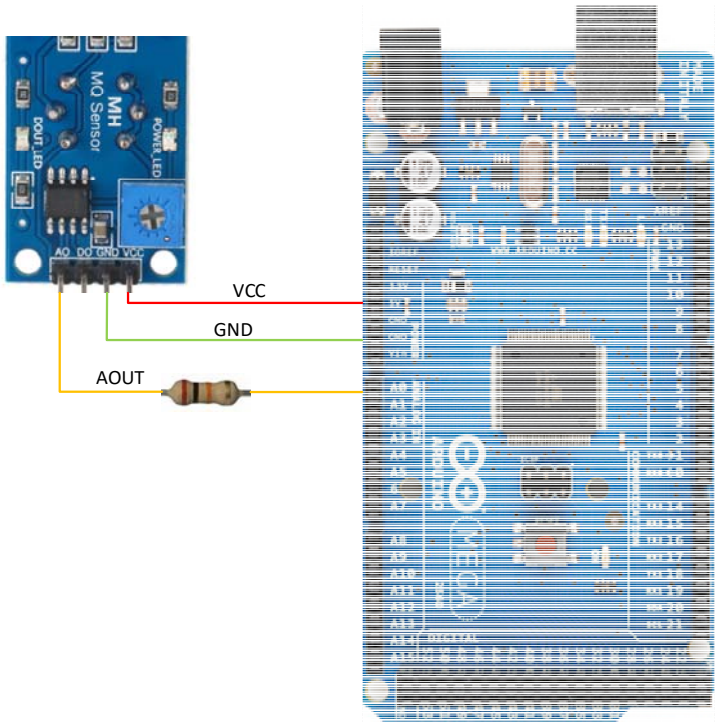


Figura 54. Esquema de conexión del sensor MQ-7 y la placa Arduino Mega.

Considerando [29], es importante indicar que para un voltaje de alimentación de 5 V y una concentración de 20 ppm, el voltaje de salida del sensor es de 0.5 V, y de igual forma para una concentración de 2000 ppm, el voltaje de salida del sensor es 4.75 V; consiguiendo así el acondicionamiento total de la señal proporcionada por el sensor.

En base a [29], se ha utilizado en el código de Arduino la ecuación lineal de calibración para la conversión entre el voltaje y la concentración del gas CO, la cual fue obtenida en base a la realización de una serie de pruebas de generación de diferentes puntos de concentración.

La ecuación utilizada es la siguiente:

$$y = 0.568x - 212.94$$

Donde:

x: Es el valor resultante de la conversión analógica – digital con una resolución de 12 bits y por tanto varía de 0 a 4095.

y: Concentración en ppm que depende del valor de la conversión analógica - digital.

Para obtener el valor de voltaje es necesario emplear la expresión:

$$y = 5x/4095$$

Donde:

x: Es el valor resultante de la conversión analógica – digital con una resolución de 12 bits y por tanto varía de 0 a 4095.

y: Voltaje en V que depende del valor de la conversión analógica – digital.

Según [30], para representar la concentración del gas en mg/m3 se utiliza la siguiente fórmula:

$$y = \frac{p * m * x}{62.36 * 1 * (273 + t)} = \frac{760 * 28.01 * x}{62.36 * (273 + 25)}$$

Donde:

p: Es la presión atmosférica en mmHg (760)

m: Es el peso molecular del CO en g/mol (28.01)

x: Es la concentración del gas en ppm

62.36: Es la constante para los gases ideales en mmHg L / mol °K

t: Es la temperatura en °C (25)

Es indispensable disponer de la concentración del monóxido de carbono tanto en ppm como en mg/m3 a fin de poder definir en secciones posteriores notificaciones de acuerdo a las normas establecidas por organismos nacionales e internacionales en cuanto a los niveles de alarma y emergencia de gases nocivos para el ser humano.

El diagrama de flujo para la medición de parámetros se observa en la Figura 55.

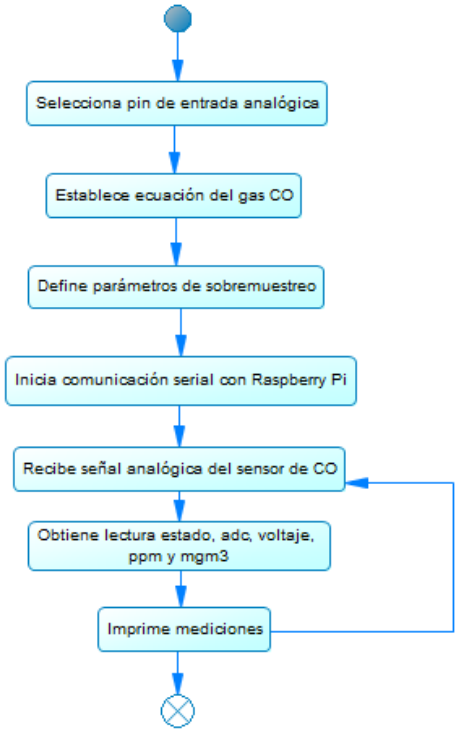


Figura 55. Diagrama de flujo para la medición de parámetros.

Considerando el diagrama de flujo de la Figura 55 es posible desarrollar el código necesario en el IDE Arduino que permita determinar el estado de detección del gas, el voltaje de salida, la concentración del CO tanto en ppm como en mg/m3 y el valor de la conversión A/D.

Se realizó una comparación entre las mediciones obtenidas mediante el código desarrollado y el valor obtenido en ppm por un equipo especializado a fin de verificar que los resultados sean confiables y muy próximos a los de un equipo usado en la industria. En las Figuras 56, 57 y 58 se muestra este proceso.

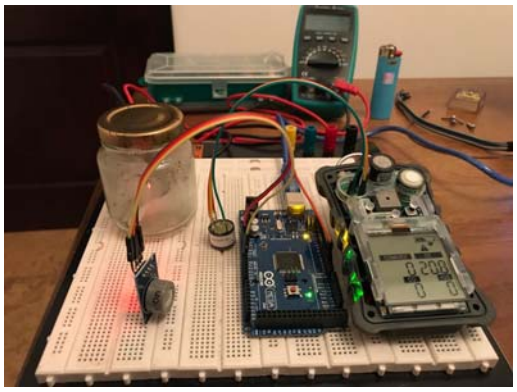


Figura 56. Preparación de Arduino + sensor de CO y equipo especializado + sensor de CO

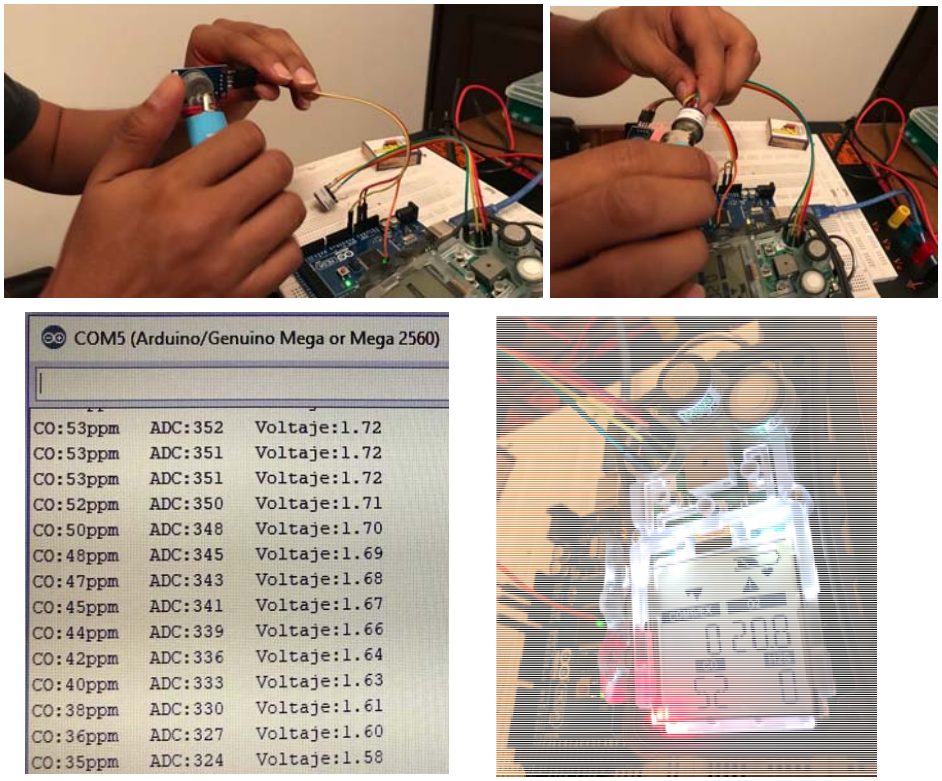


Figura 57. Realización del primer test y comparación de resultados (ppm=52).

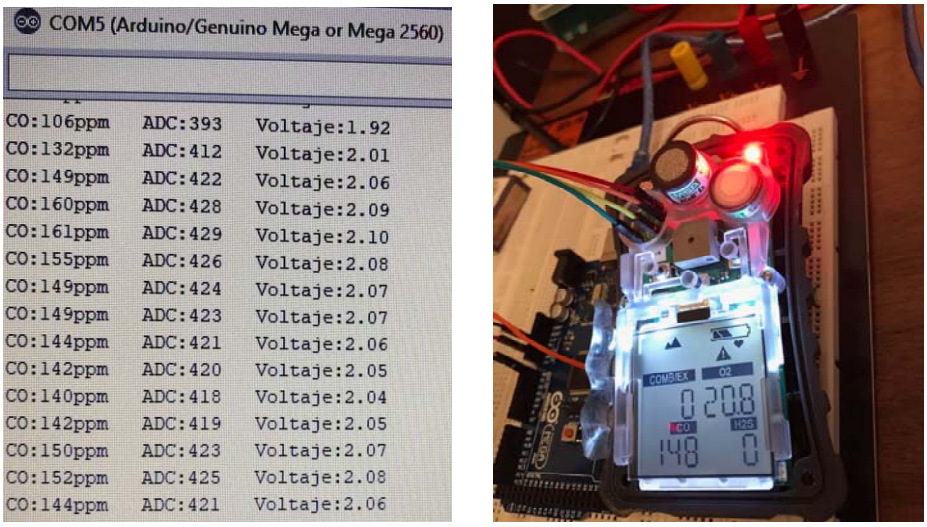
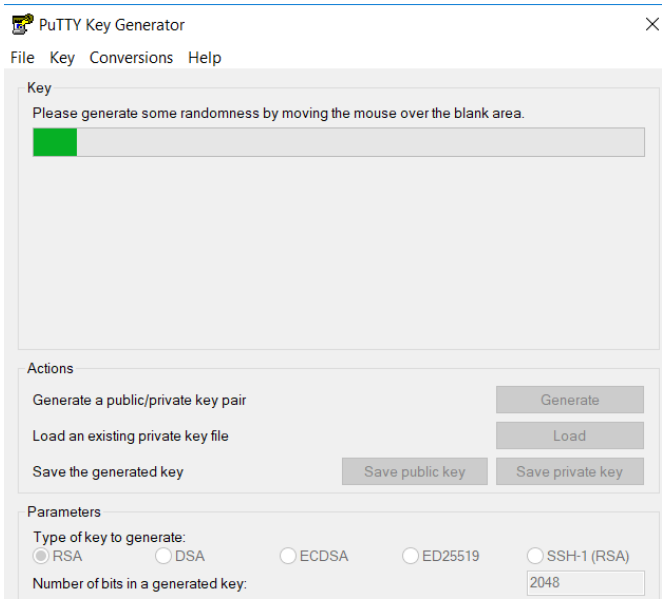


Figura 58. Realización del segundo test y comparación de resultados (ppm=149 (valor más cercano) y ppm=148).

Creación de una máquina virtual con Ubuntu Server 18.04 LTS

De acuerdo a [24], una buena opción para establecer una conexión SSH segura con una máquina virtual de Linux hospedad en Azure es usar un par de claves (pública y privada). Para poder generar estas claves se utilizó la herramienta PuTTY Key Generator, especificando RSA como algoritmo de cifrado y definiendo una longitud de clave de 2048 bits. Esto se observa en la Figura 59.



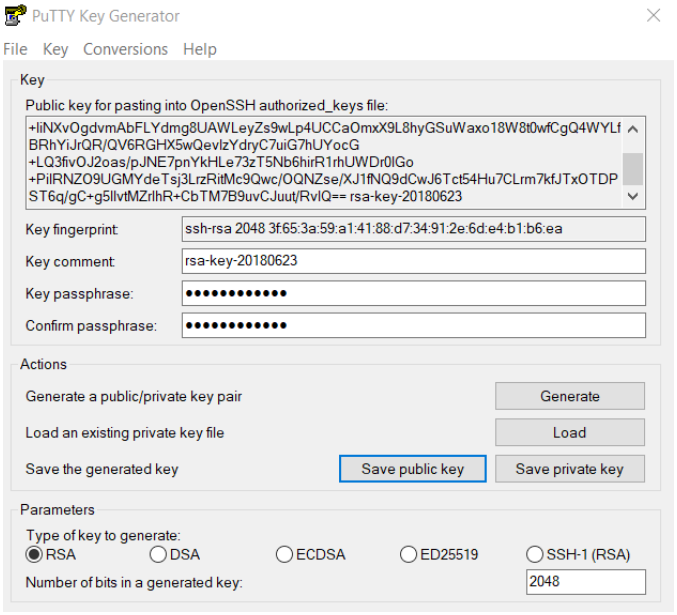


Figura 59. Generación de una clave pública y privada con RSA y 2048 bits de longitud de clave.

Una vez que se han generado este par de claves se procedió a guardar los archivos correspondientes a fin de poder utilizarlos en la creación de la máquina virtual y en la conexión SSH posterior. En base a [25], al ingresar al Portal de Azure y seleccionar la opción de Máquina virtuales es posible seleccionar una gran variedad de opciones, de entre las cuales se selecciona Ubuntu Server 18.04 LTS como se observa en la Figura 60.

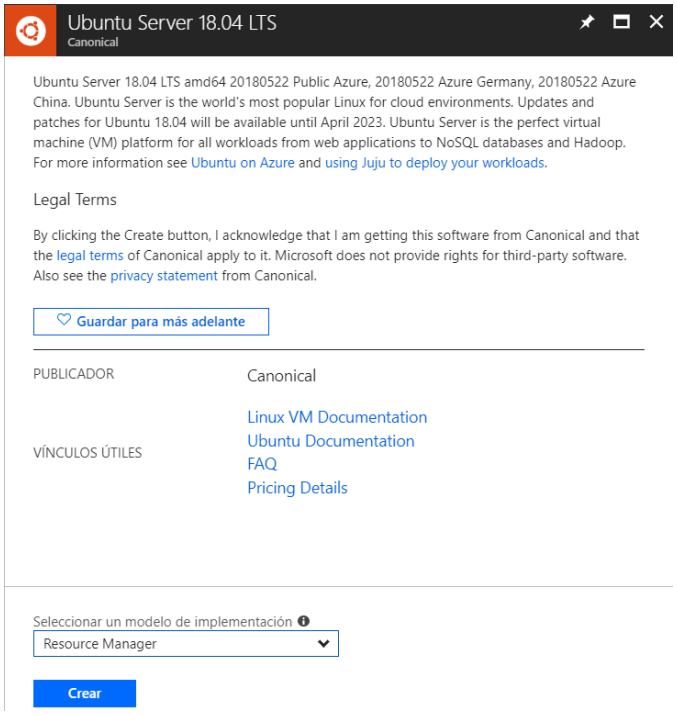


Figura 60. Creación de una máquina virtual Ubuntu Server 18.04 LTS.

Una vez que se ha creado la máquina virtual deseada es necesario configurar algunos parámetros básicos como los que se visualizan en la Figura 61. Un campo importante es el de la clave pública SSH generada anteriormente y que se visualiza en la Figura 59.

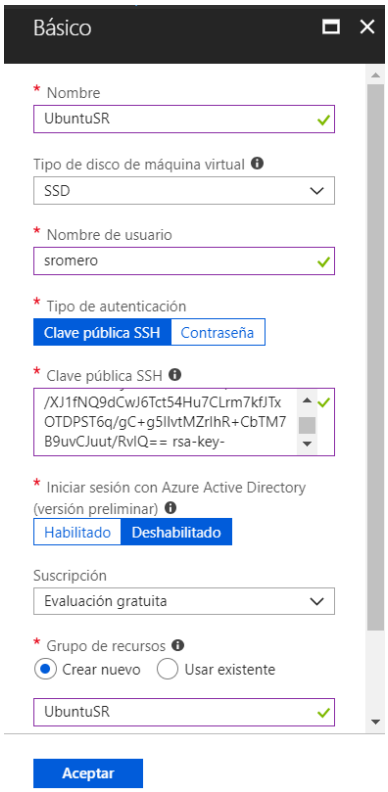


Figura 61. Configuración básica de una máquina virtual Ubuntu Server 18.04 LTS.

El siguiente paso es elegir el tamaño de la máquina virtual, en este caso se seleccionó un equipo de 2 núcleos, 8 GB de RAM y 4 discos duros SSD de 16 GB como se refleja en la Figura 62.

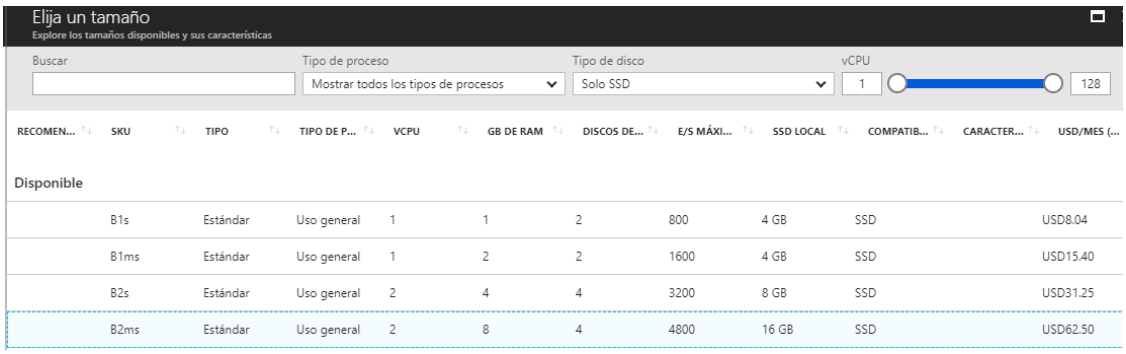


Figura 62. Elección del tamaño de la máquina virtual Ubuntu Server 18.04 LTS.

Por último, es necesario configurar algunas características más como las relacionadas al almacenamiento y red. Un punto clave es el campo de la Dirección IP pública, la cual debe definirse como estática a fin de que se pueda acceder a la aplicación web que alojará este servidor utilizando los mismos 4 octetos generados al momento de crear la máquina virtual. Esto se observa en la Figura 63.

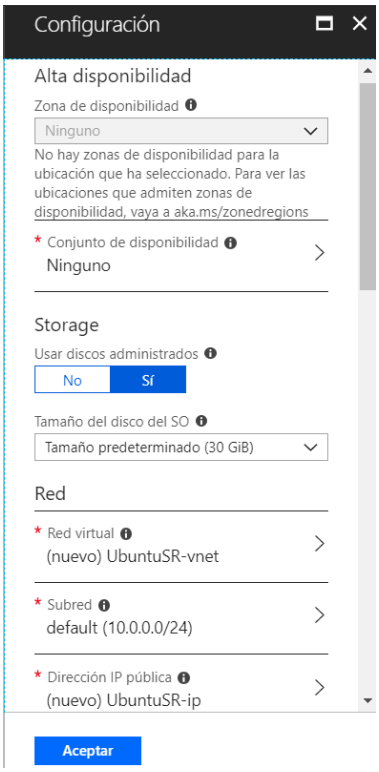


Figura 63. Configuración del almacenamiento y red de la máquina virtual Ubuntu Server 18.04 LTS.

También, de acuerdo a [25] y como se muestra en la Figura 64, para poder acceder vía SSH a la máquina virtual creada, es necesario especificar la dirección IP pública (40.87.54.55) y definir la autenticación mediante la selección de la clave privada generada con la herramienta PuTTY Key Generator en pasos anteriores.

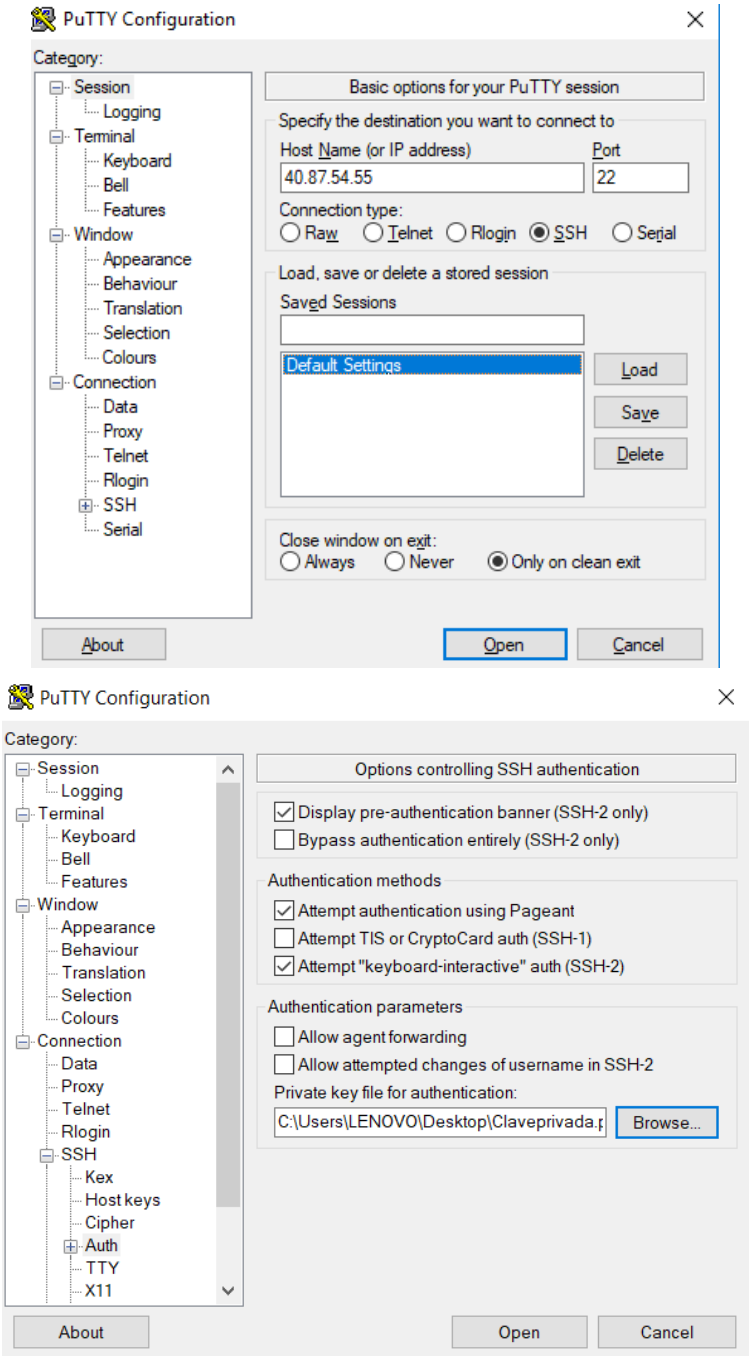


Figura 64. Configuración de la conexión SSH en PuTTY para acceder a la máquina virtual Ubuntu Server 18.04 LTS.

Una vez que se establece la conexión SSH, inmediatamente se solicita el nombre del usuario definido previamente en la configuración básica de la máquina virtual y el password de la clave pública a fin de poder acceder a la consola del servidor como se visualiza en la Figura 65.

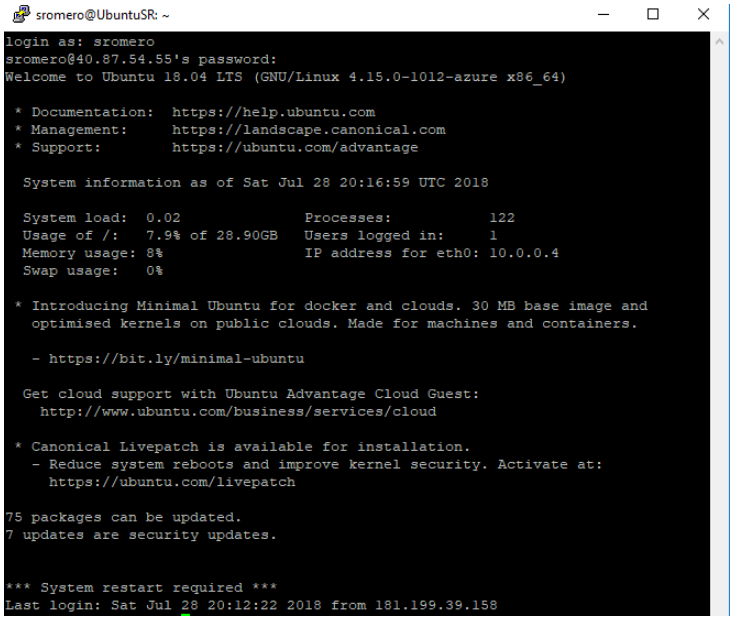


Figura 65. Acceso al terminal de la máquina virtual Ubuntu Server 18.04 LTS.

Instalación de MongoDB, Node.js, npm y Angular CLI.

Según [26] [26], para instalar MongoDB primero se debe descargar la lista de los paquetes que contienen los repositorios y la información actualizada de la versión más reciente de estos paquetes mediante el comando que se visualiza en la Figura 66.

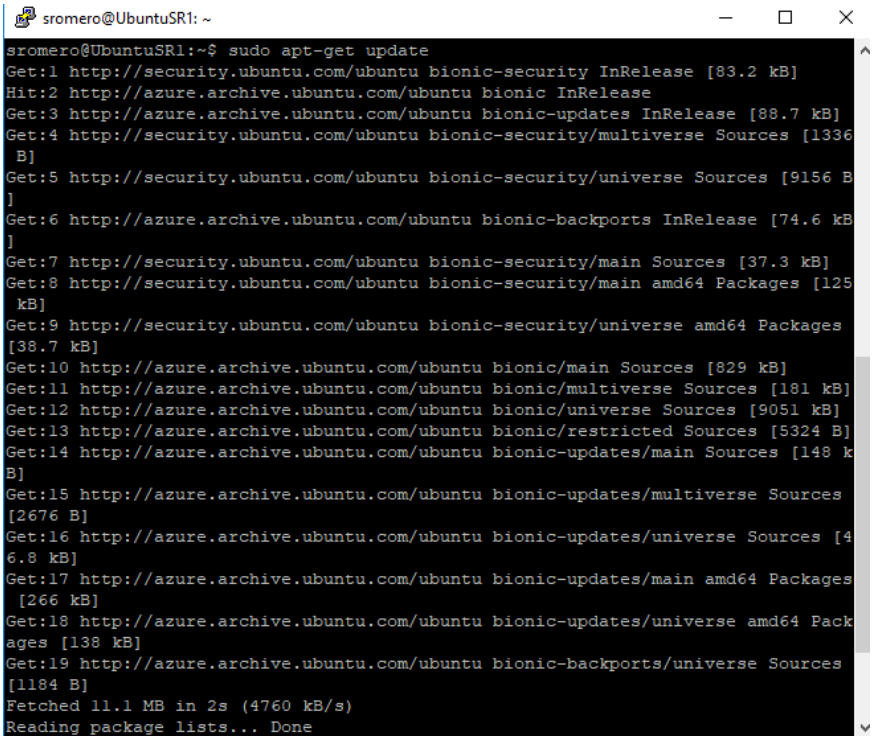


Figura 66. Actualización de la lista de paquetes en el terminal de Ubuntu Server 18.04 LTS.

Luego es necesario instalar el paquete MongoDB como se observa en la Figura 67.

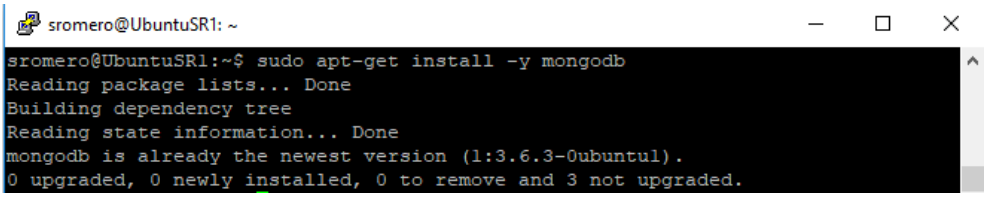


Figura 67. Instalación de MongoDB.

Con el fin de verificar el estado del servicio, la versión de la base de datos, la dirección del servidor, el puerto y el estado de la salida se pueden ejecutar los comandos de la Figura 68.

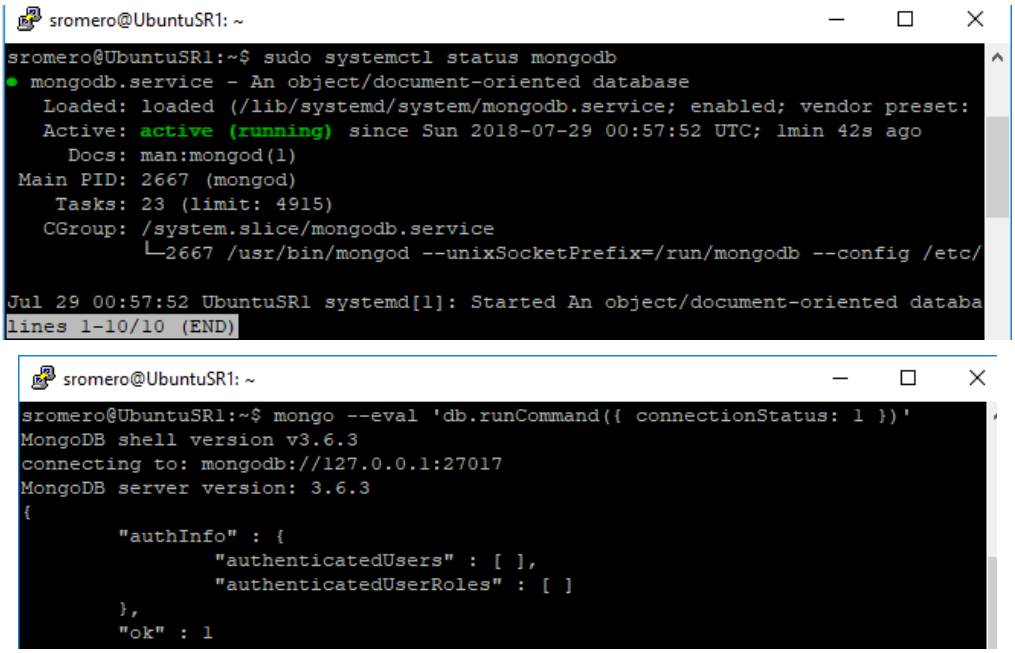


Figura 68. Verificación de la instalación de MongoDB.

De acuerdo a [27], para instalar Node.js es necesario ejecutar el comando que se visualiza en la Figura 69.

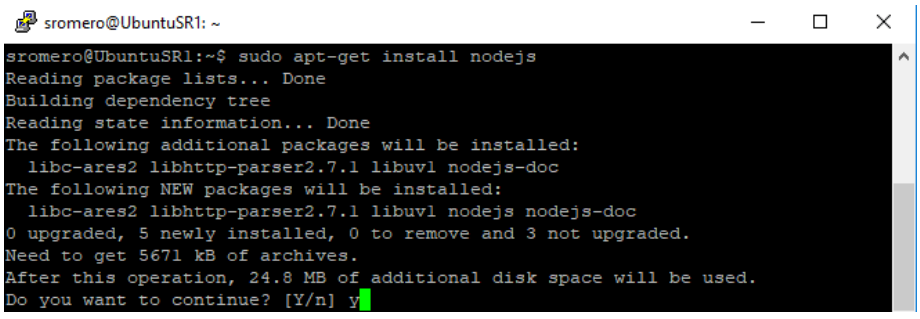


Figura 69. Instalación de Node.js.

En la Figura 70 se observa la versión de Node.js instalada y la verificación en un explorador web del cliente.

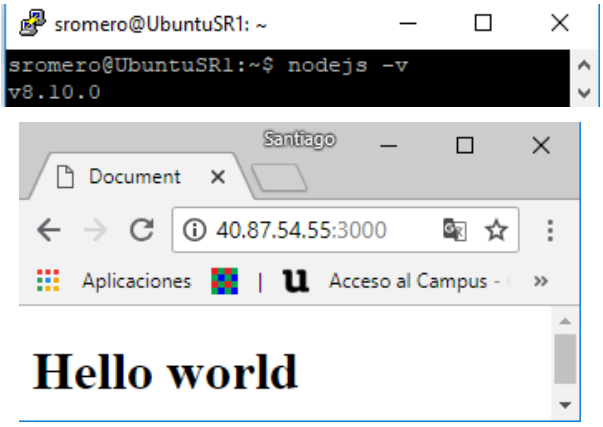


Figura 70. Verificación de la instalación de Node.js.

También en base a [27], es importante instalar el administrador de paquetes de Node.js conocido como npm de la forma que muestra la Figura 71.

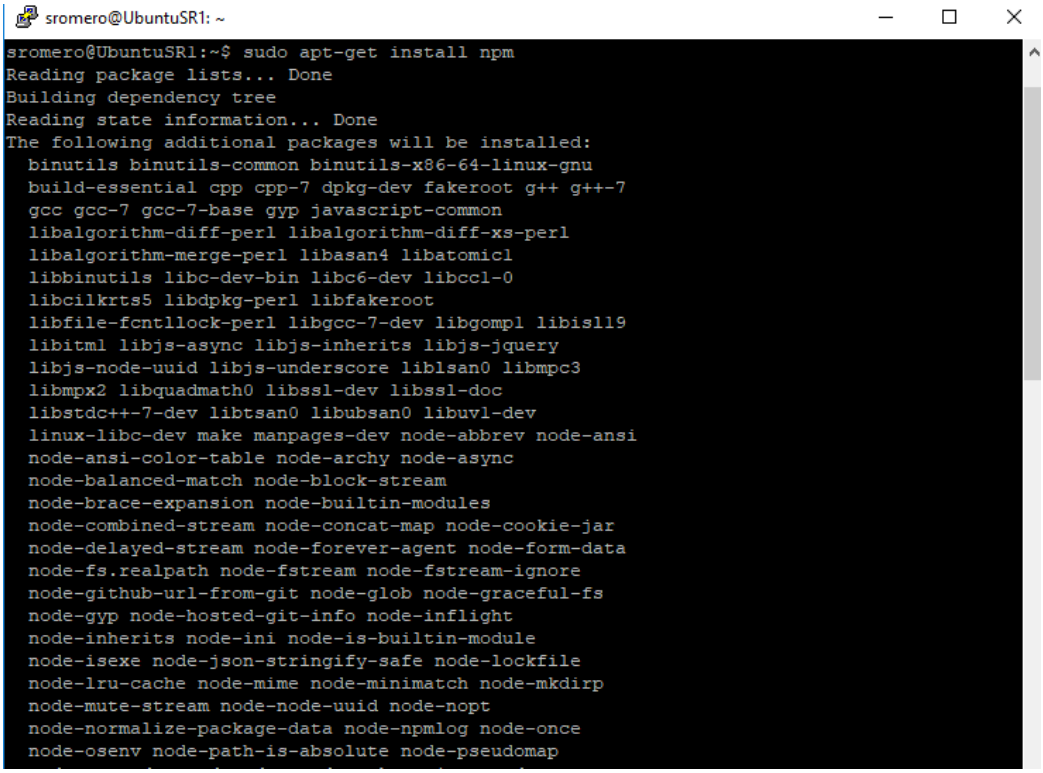


Figura 71. Instalación de npm.

En la Figura 72, se puede observar la versión de npm instalada.

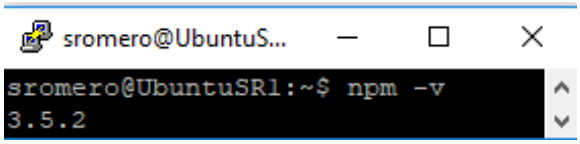


Figura 72. Verificación de la versión de npm.

La instalación de Angular CLI consiste en el comando que se visualiza en la Figura 73.

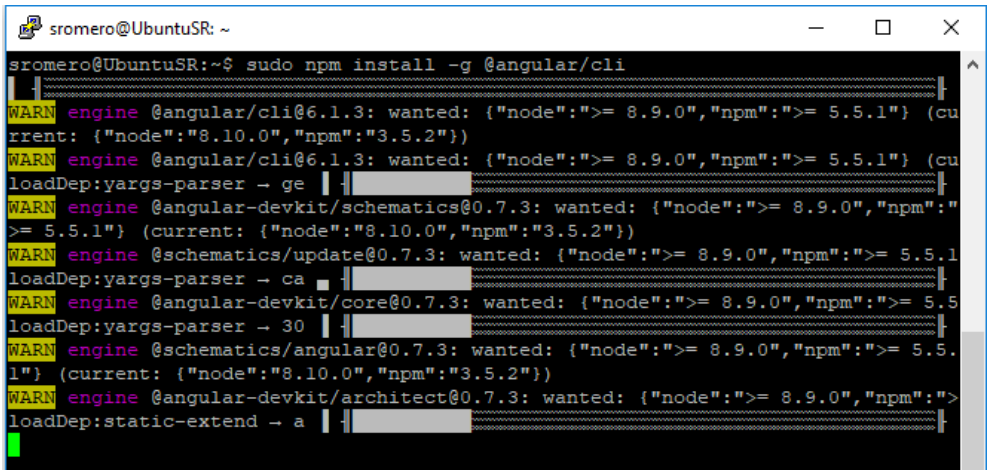


Figura 73. Instalación de Angular CLI.

En la Figura 74, se puede observar la versión de Angular CLI instalada.

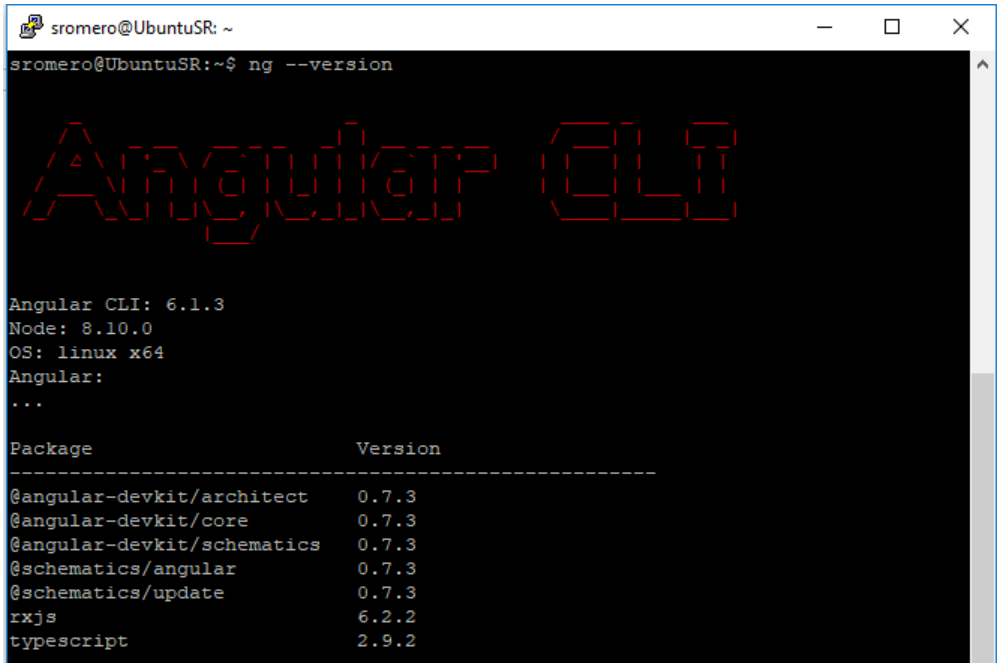


Figura 74. Verificación de la versión de Angular CLI.

Anexo II. Encuesta de la Aplicación Web para detección y medición de CO en domicilios

Por favor responda a las siguientes preguntas considerando que 1 es el nivel más bajo y 5 el más alto.

Identidad

1. ¿La vista principal de la aplicación web refleja una correspondencia adecuada con el sistema de detección y medición de niveles de CO en domicilios?

Seguridad

2. ¿La página de login de la aplicación web otorga el sentido de robustez necesario?

Disponibilidad

3. Durante el tiempo que usó la aplicación web, ¿Cuál es el nivel de conformidad respecto a la disponibilidad de los datos?

Mantenimiento

4. ¿La aplicación web ofrece las opciones necesarias para realizar trabajos de mantenimiento a los equipos transmisores?

Escalabilidad

5. ¿Piensa que la aplicación web sería capaz de incorporar más equipos de transmisión y sensores?

Usabilidad

6. ¿La aplicación web es fácil de usar?
7. ¿Cree que la curva de aprendizaje para utilizar la aplicación web es rápida?

Diseño

8. ¿Considera que el diseño de la aplicación web es atractivo, útil e innovador?
9. ¿Considera que la incorporación de tablas y graficos mejora la comprensión de los datos que maneja la aplicación web?

Navegación

10. ¿Cuál es su nivel de satisfacción respecto a la interacción con la aplicación web?
11. ¿El panel de navegación se encuentra en un lugar visible?

12. ¿La información de las lecturas, notificaciones, parámetros de configuración y administración de clientes se puede encontrar fácilmente?

Recomendación de la aplicación web

13. ¿Recomendaría el uso de la aplicación web?

Anexo III. Artículo

Desarrollo de una aplicación web cloud para detectar CO en domicilios.

Santiago Romero.

Resumen.

El desarrollo de sistemas que involucran Tecnologías de la Información y Comunicación (TIC) está ligado cada vez más al uso de software y hardware libre que en conjunto permiten la integración y reutilización de componentes que se prueban amplia y continuamente en un afán de mejora.
En los últimos años el campo de Internet de las Cosas (IoT) ha experimentado una evolución tanto en el hardware como en software, siendo este último elemento el que sigue marcando la tendencia actual al permitir diseñar e implementar aplicaciones. También, el hardware libre no se ha estancado debido a que a más de reducir significativamente los costos, ofrece plataformas de desarrollo que hacen más sencilla la investigación en diferentes niveles, incluyendo prácticas de laboratorio y sistemas complejos en producción.
El objetivo del presente trabajo es implementar un sistema de detección y registro de niveles de CO de bajo costo en domicilios usando hardware y software libre, tecnologías web y servicios cloud que sea capaz de salvaguardar vidas humanas.
Para alcanzar este objetivo, se conectará el sensor de CO al dispositivo Arduino, el cual convertirá las señales analógicas en digitales y enviará estos datos al Raspberry Pi mediante una conexión serial. Este dispositivo utilizará un servicio web para enviar la lectura ADC, la medición de voltaje, el estado de detección del gas, la cantidad de partículas por millón del CO y la cantidad de mg/m3 del CO a una base de datos de la aplicación web desplegada en la nube.
A partir de los datos obtenidos desde el sensor, se realizará un análisis e interpretación de dicha información. Estos resultados permitirán realizar varias gráficas, obteniendo así una visión más clara del comportamiento en las mediciones registradas.

Palabras Clave: Tecnologías de la Información y Comunicación (TIC), Internet de las Cosas (IoT), Arduino, Raspberry Pi.

Introducción.

La calidad del aire en las viviendas es importante para la salud de las personas. Existen gases tóxicos que no pueden ser percibidos fácilmente y de manera oportuna por el ser humano provocando daños severos e irreparables en el organismo.
Por este motivo el desarrollo e implementación de un sistema que permita la temprana detección y medición de CO en el entorno, tiene como finalidad precautelar la salud de los ocupantes de un domicilio, evitando que estén expuestos al CO y ejecutar acciones oportunas en cuanto se realice la detección del gas.
Este trabajo fin de máster trata del diseño y programación de una aplicación web orientada a la detección y medición de monóxido de carbono CO en domicilios priorizando el empleo de tecnologías de hardware y software libre en su gran mayoría, por lo que se utilizará un dispositivo de tamaño reducido como el Raspberry Pi con sistema operativo Raspbian (distribución basada en Debian), una placa denominada Arduino, un entorno open source de ejecución para JavaScript en el servidor como Node.js y el framework Express.js para el soporte de aplicaciones web, un framework de JavaScript de código abierto para trabajar en el cliente como Angular y un sistema gestor de base de datos NoSQL de código abierto como MongoDB.
En cuanto a software se desarrollará código en el IDE de Arduino para procesar la señal del sensor de CO y enviar estos datos al Raspberry Pi de forma serial. También se

implementará código en Python para recibir los datos seriales y consumir un servicio web haciendo posible que esta información se almacene en un SGBD.
Este SGBD, el servicio desplegado en Node.js y la sección de código desarrollada usando Angular residirán en la nube permitiendo que se implemente una aplicación web robusta con tecnología JavaScript, que a la vez sea segura y cuya fuente de datos sean los datos del sensor de CO.
Dependiendo de la cantidad de CO que el cuerpo humano puede tolerar y que no afecta de forma permanente a la salud, se configurará un mensaje de alerta o emergencia que será enviado vía e-mail a la persona responsable del domicilio.
Primero se realiza una revisión al contexto y estado del arte, luego se determinan los objetivos generales y específicos, se explica la metodología de trabajo a emplear y se inicia con el desarrollo práctico. En esta sección se detallan los requisitos funcionales y no funcionales, el diseño e implementación de la aplicación web, la evaluación de cada uno de los componentes y del sistema, finalizando con una encuesta que evidencia el cumplimiento de los requisitos no funcionales.

Contexto y estado del arte.

IoT

Según [5], IoT es la conexión de objetos materiales a Internet permitiendo la transmisión de datos de forma remota a través de la configuración de capacidades de captura y procesamiento de datos ambientales. Una vez que se ha establecido la conexión, cada objeto adquiere una dirección de red que lo hace identificable de forma única. Estos objetos generalmente poseen la capacidad de detección en alguna de sus capas que les permite registrar dinámicamente los cambios en el entorno y transmitir esa información a través de Internet.

Node.js

De acuerdo a [7], se puede mencionar que Node.js es un entorno en tiempo de ejecución multiplataforma de código abierto para el desarrollo de aplicaciones en el lado del servidor construido sobre el motor de JavaScript de Google Chrome. Las aplicaciones de Node.js se escriben en JavaScript y se pueden ejecutar en Mac OS X, Windows y Linux. Cuenta con una amplia librería de módulos JavaScript que simplifica el desarrollo de aplicaciones web.

Express.js

Según [9], Express es el framework del lado servidor para el desarrollo web más popular alojado en el entorno en tiempo de ejecución Node.js. Está escrito en JavaScript y es a su vez es la librería que se usa como base para otros frameworks web de Node.js. Proporciona mecanismos para construir manejadores de peticiones con diferentes métodos HTTP (GET, POST, DELETE, etc.) para varias rutas URL, mecanismo para agregar el "middleware" adicional de procesamiento de peticiones en cualquier punto dentro del esquema de manejo de peticiones, etc.

Angular

Tomando en cuenta [11], Angular es un framework estructural de JavaScript para aplicaciones web dinámicas que permite usar HTML como lenguaje de plantilla y extender la sintaxis de HTML para expresar los componentes que requiera la aplicación de forma clara y concisa. Una de las características fundamentales de Angular.js es el enlace de datos que permite la sincronización automática de los datos entre los componentes de la vista y el modelo.

MongoDB

De acuerdo a [13], MongoDB es una base de datos open source, multiplataforma, NoSQL con un modelamiento orientado a documentos que ofrece alto rendimiento, alta disponibilidad y escalamiento automático. Un registro en MongoDB es un documento, que es una estructura de datos compuesta por pares (clave: valor). Los documentos de MongoDB son similares a los objetos JSON. Los valores de las claves pueden incluir otros documentos, arreglos y arreglos de documentos.

Pyhton

De acuerdo [15], Python es un lenguaje de alto nivel, interpretado, orientado a objetos y dinámico ya que realiza una verificación de tipo en tiempo de ejecución a fin de que el tipo de dato configurado coincida con la información esperada. Existen muchas implementaciones que ejecutan la versión 2.x de Python, pero la versión 3.x la reemplazará totalmente a futuro. La implementación más común es CPython, escrita en C que compila un programa de Python en bytecode intermedio.

Conclusiones del estado del arte.

Considerando la facilidad de conexión Wi-Fi, tiempos de arranque del sistema operativo, capacidad para manejar entradas analógicas, optimización del consumo de energía, sencillez y flexibilidad en la configuración para establecer la comunicación con una aplicación web se ha considerado utilizar tanto Arduino, el sensor de CO conocido como MQ-7 y el Raspberry Pi a fin de aprovechar al máximo cada una de las ventajas de estos dispositivos en el sistema propuesto.
Existen muchas plataformas de IoT capaces de recibir datos de sensores que miden parámetros ambientales como temperatura, humedad, vibraciones sísmicas, detección de gases, medición de la intensidad de luz; unas con más facilidades que otras pero que dependen totalmente de las capacidades que ofrece la plataforma. Es por esta razón que el proyecto a desarrollar plantea la creación de una aplicación web cloud no propietaria que permita recibir los datos y visualizarlos en tiempo real y desde cualquier ubicación si se dispone del acceso autorizado a la vez que se utilizan servicios sencillos REST de forma segura. De esta forma también se evita tener que configurar información sensible de autenticación para la plataforma de IoT en Arduino o configurar el código necesario en el IDE para definir los canales a usar para enviar los datos a la aplicación.
Por último, las plataformas de IoT utilizan determinados protocolos para el envío de datos como MQTT y por consiguiente es obligatorio incluir estas librerías al desarrollar el código en Arduino. Debido a esto, lo mejor es realizar esta conexión en Raspberry Pi mediante Python donde el número de librerías a utilizar se reduce notablemente al desarrollar y emplear una aplicación web cloud propia e independiente.

Diseño e implementación.

De acuerdo a los requisitos identificados se diseñó la siguiente arquitectura para el sistema de bajo costo para la detección y medición del gas tóxico monóxido de carbono en domicilios.

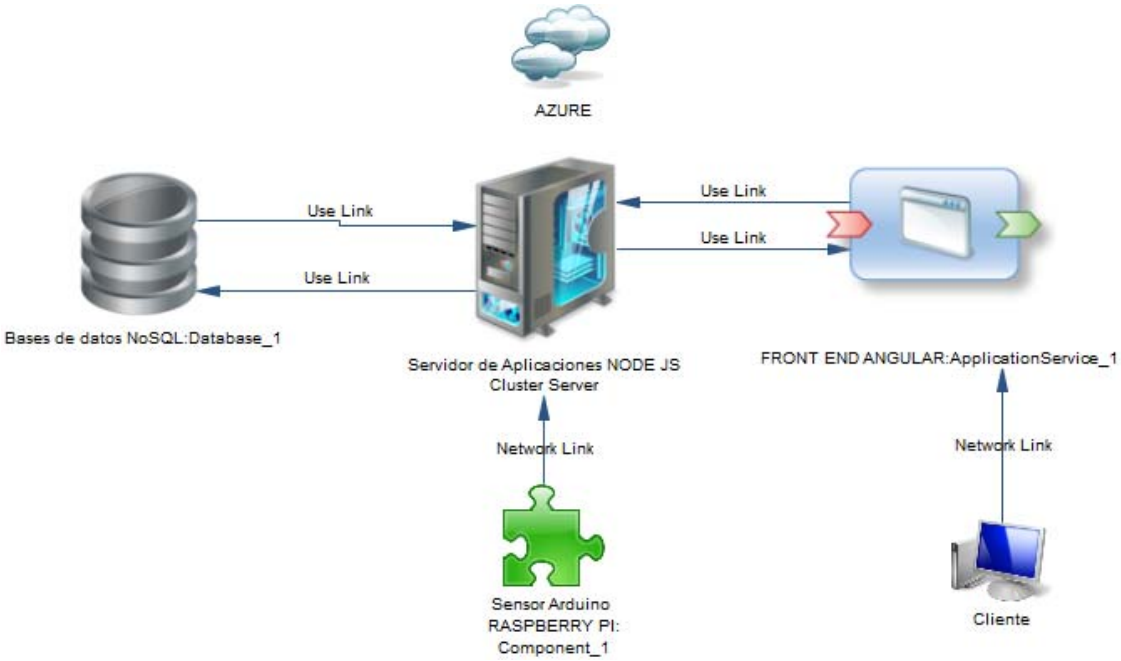


Figura 1. Arquitectura del sistema de detección y medición de CO en domicilios.

Las principales funcionalidades de la aplicación web son las siguientes:

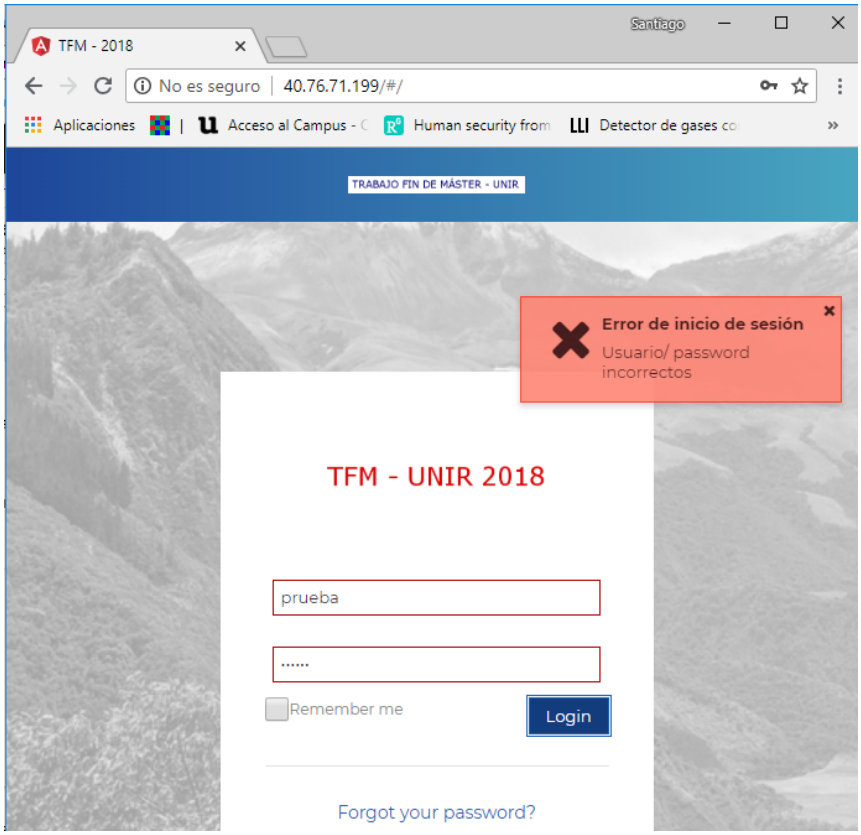


Figura 2. Pantalla de Login.

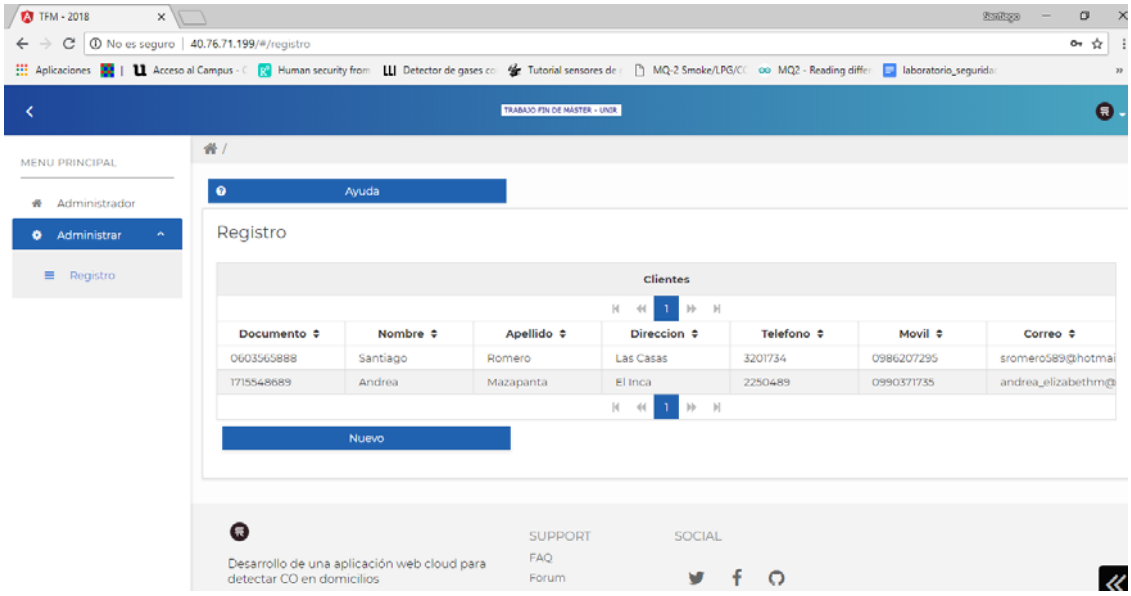


Figura 3. Pantalla de Registro.

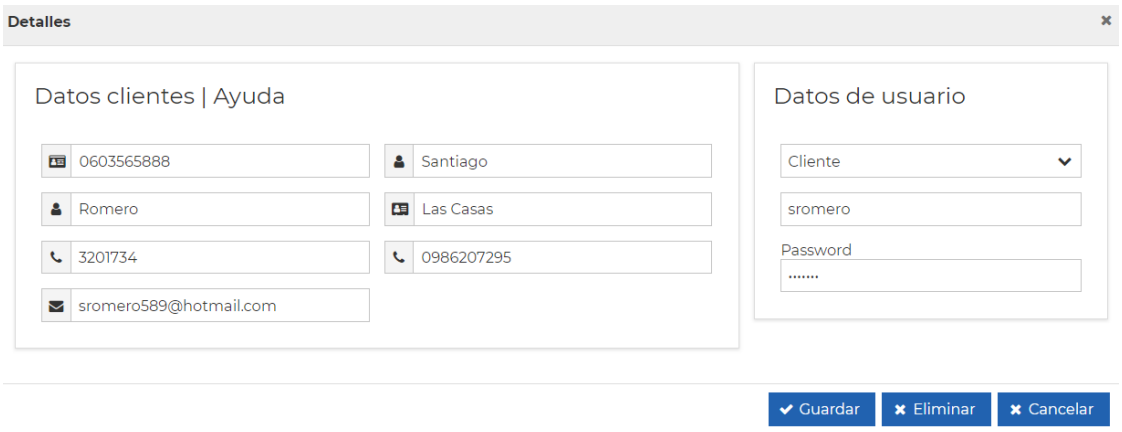


Figura 4. Pantalla de creación de usuarios.

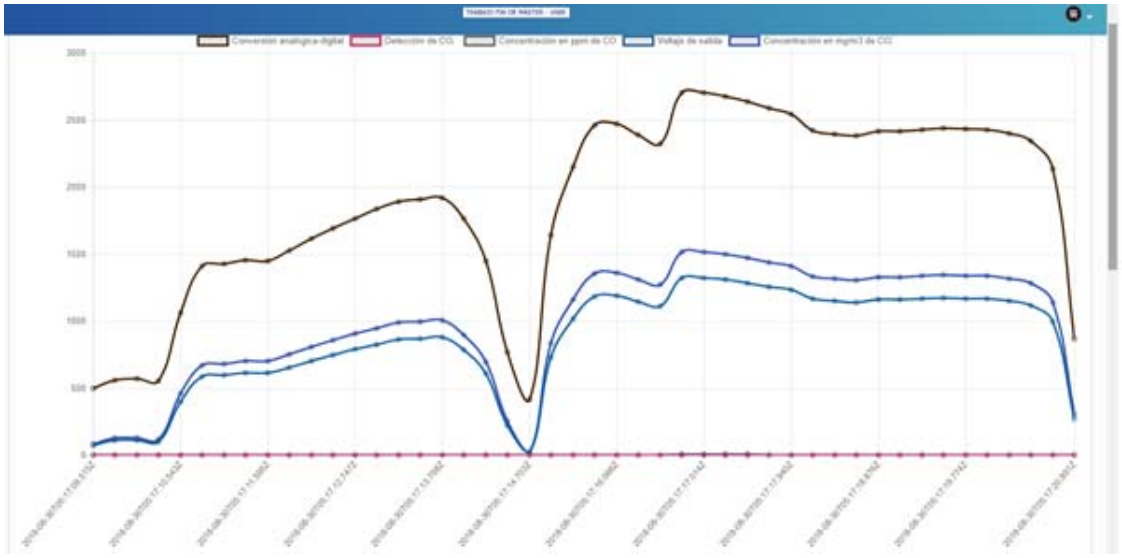


Figura 5. Gráfica de las mediciones realizadas entre el 27 de Agosto y 31 de Agosto del 2018.

Toda la tabla | Exportar seleccionado

1 2 3 4 5

Conversión analógica-digital	Detección de CO	Concentración en ppm de CO	Voltaje de salida	Concentración en mg/m3 de CO	Hora
497	1	71.628	0.61	82.052	2018-08-30T05:17:09.515Z
559	1	111.956	0.69	128.248	2018-08-30T05:17:09.772Z
569	1	110.82	0.7	126.947	2018-08-30T05:17:10.033Z
554	1	100.596	0.68	115.235	2018-08-30T05:17:10.284Z
1063	1	398.228	1.31	456.18	2018-08-30T05:17:10.543Z
1407	1	586.236	1.72	671.548	2018-08-30T05:17:10.802Z
1423	1	595.324	1.73	681.958	2018-08-30T05:17:11.092Z
1453	1	612.932	1.78	702.129	2018-08-30T05:17:11.345Z
1449	1	611.228	1.77	700.177	2018-08-30T05:17:11.595Z
1525	1	654.964	1.86	750.277	2018-08-30T05:17:11.890Z

1 2 3 4 5

Figura 6. Tabla de valores para las mediciones realizadas.

	A	B	C	D	E	F
1	Conversión analógica-digital	Detección de CO	Concentración en ppm de CO	Voltaje de salida	Concentración en mg/m3 de CO	Hora
2	1352	1	557.268	1.65	638.364	2018-08-26T21:52:35.433Z
3	1457	1	614.636	1.78	704.081	2018-08-26T21:52:35.705Z
4	1496	1	636.788	1.83	729.456	2018-08-26T21:52:35.925Z
5	1548	1	665.756	1.89	762.64	2018-08-26T21:52:36.141Z
6	1595	1	694.156	1.95	795.173	2018-08-26T21:52:36.358Z
7	1671	1	737.324	2.04	844.623	2018-08-26T21:52:36.573Z
8	1733	1	771.404	2.12	883.662	2018-08-26T21:52:36.789Z
9	1788	1	802.644	2.18	919.448	2018-08-26T21:52:37.011Z
10	1829	1	827.068	2.23	947.427	2018-08-26T21:52:37.229Z
11	1870	1	849.788	2.28	973.453	2018-08-26T21:52:37.449Z
12	1908	1	870.804	2.33	997.527	2018-08-26T21:52:37.662Z
13	1940	1	888.412	2.37	1017.698	2018-08-26T21:52:37.878Z
14	1969	1	906.02	2.4	1037.868	2018-08-26T21:52:38.092Z
15	1994	1	919.652	2.43	1053.484	2018-08-26T21:52:38.316Z
16	2015	1	932.716	2.46	1068.449	2018-08-26T21:52:38.546Z
17	2038	1	944.076	2.49	1081.462	2018-08-26T21:52:38.764Z
18	2053	1	953.164	2.51	1091.873	2018-08-26T21:52:38.986Z
19	2069	1	962.82	2.53	1102.934	2018-08-26T21:52:39.209Z
20	2078	1	967.932	2.54	1108.79	2018-08-26T21:52:39.423Z
21	2089	1	973.612	2.55	1115.296	2018-08-26T21:52:39.642Z
22	2095	1	977.588	2.56	1119.851	2018-08-26T21:52:39.860Z
23	2096	1	977.02	2.56	1119.2	2018-08-26T21:52:40.075Z

download (1)

Figura 7. Exportación en formato .CSV de las mediciones realizadas.

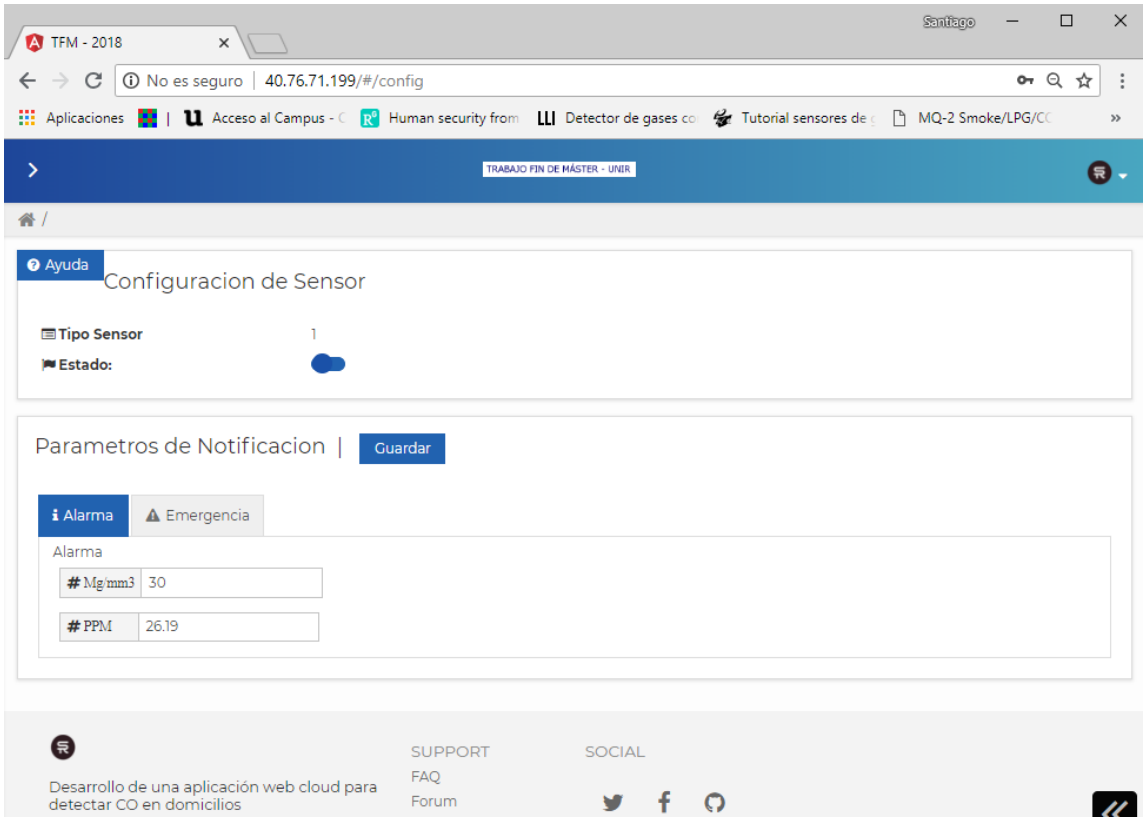


Figura 8. Pantalla de configuración del equipo y de los parámetros de notificación.

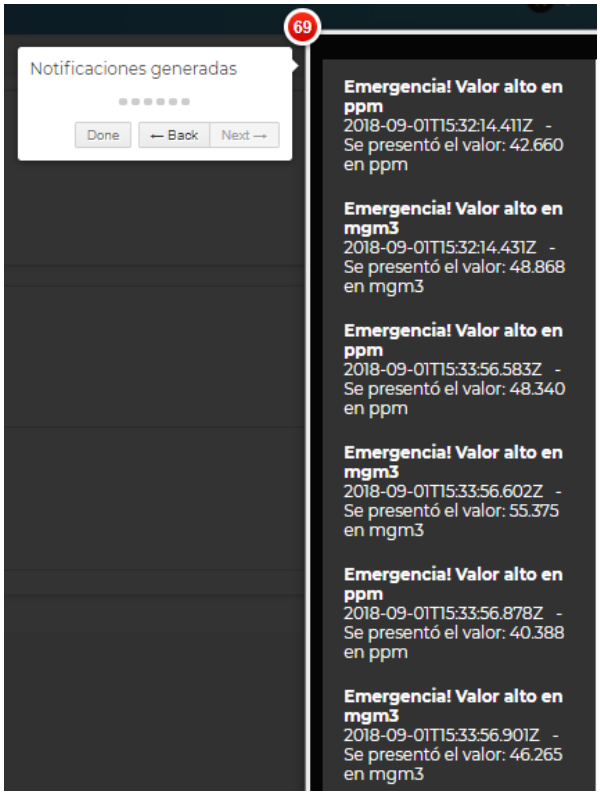


Figura 9. Notificaciones recibidas en el perfil del cliente dentro de la aplicación web.

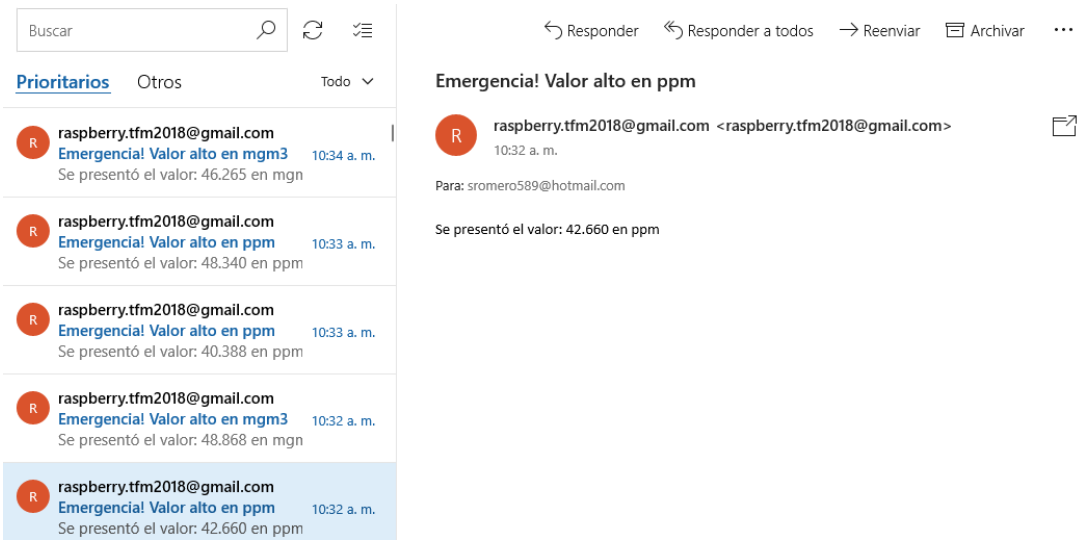


Figura 10. Notificaciones enviadas por la aplicación web vía e-mail a la cuenta de correo del cliente.

1

Magnitud	Promedio	Maximo	Minimo
PPM	282	510.124	40.388
ADC	870	1274	444
VOLTAJE	1.06	1.56	0.54
MGM3	323	584.359	46.265
ESTADO	1.00	1	1

1

Figura 11. Tabla de valores máximos, mínimos y promedios.

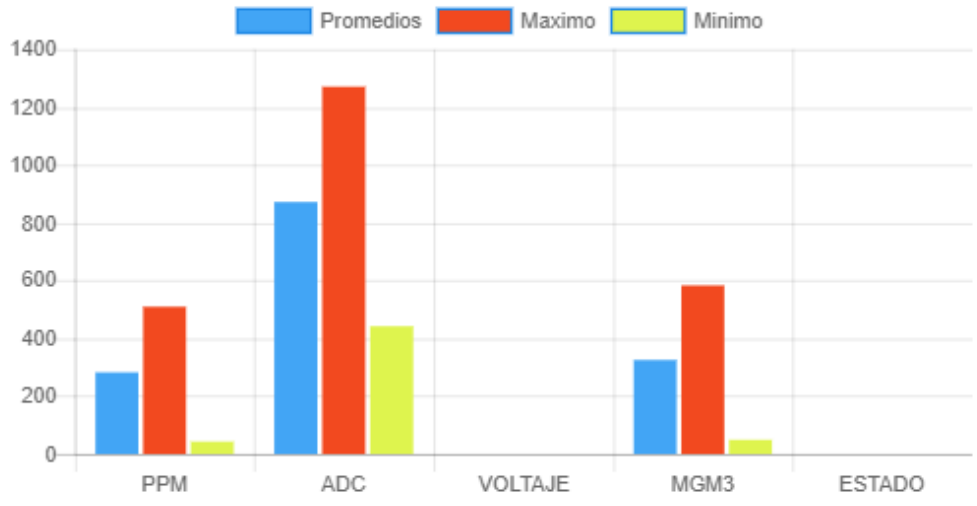


Figura 12. Gráfica de valores máximos, mínimos y promedios.

Conclusiones

Se pudo implementar un sistema de detección y registro de niveles de CO de bajo costo en domicilios usando hardware y software libre, tecnologías web y servicios cloud precautelando la salud de las personas y evitando la pérdida de vidas humanas debido a la inhalación del gas tóxico monóxido de carbono.
En base al análisis de los niveles de CO que son perjudiciales para la salud de las personas en domicilios, se logró configurar el envío de notificaciones de alerta y emergencia a los usuarios mediante la conexión con el servicio de correo electrónico de Gmail a fin de que puedan tomar las medidas de prevención necesarias.
Se logró en su gran mayoría seguir la tendencia “free” ya que se utilizó dispositivos de hardware libre para la lectura, procesamiento y transmisión de los niveles de CO en el entorno a la vez que se empleó software libre tanto en la infraestructura necesaria para el desarrollo de la app web como en la propia codificación.
Se diseñó e implementó una aplicación web que es capaz de recibir los datos transmitidos por el sensor de CO, cumple con el patrón MVC, satisface los requisitos de seguridad en cuanto a control de acceso, permite la visualización de las mediciones y emite notificaciones de alerta y emergencia en caso de que se superen los valores umbrales definidos salvaguardando la vida de las personas en domicilios.
Se generaron reportes personalizados considerando la identificación de los equipos transmisores, fecha de registro de los datos y variables medidas.
Luego de haber realizado las pruebas de evaluación tanto a nivel de sistema como en la medición de las variables eléctricas y ambientales, envío de datos a la aplicación web y validación del funcionamiento de la aplicación web se puede manifestar que se ha cumplido con los objetivos fijados y cubriendo los requisitos funcionales y no funcionales.
Al analizar los resultados obtenidos por la Encuesta realizada se puede decir que los requisitos no funcionales como la seguridad, disponibilidad, mantenimiento, usabilidad y diseño de la aplicación web alcanzaron un nivel 4, es decir, de satisfacción. Respecto a la escalabilidad de la aplicación web se puede manifestar que alcanzó un nivel de 5, es decir, muy satisfecho. También es importante señalar que el 92 % de los encuestados expresó que recomendaría el uso de la aplicación web.

Bibliografía

[1] Brendan O’Brien, “Why the ‘Internet of Things’ is Important”, ARIA [En línea]. Disponible en: https://www.ariasystems.com/blog/internet-things-important/. [Accedido: 09-may-2018]

[2] ARC Advisory Group, “IoT and Analytics Can Add Smartness to Fire Detection”, 2017 [En línea]. Disponible en: https://industrial-iot.com/2017/08/iot-analytics-smart-fire-detection/. [Accedido: 10-may-2018]

[3] Anandhakrishnan S, Deepesh Nair, Rakesh K, Sampath K,Gayathri S Nair, “IOT Based Smart Gas Monitoring System”, National Conference on "Emerging Research Trends in Electrical, Electronics & Instrumentation", pp. 82-87, 2017. [En línea]. Disponible en: http://www.iosrjournals.org/iosr-jeee/Papers/Conf.17017/Volume-3/13.%2082-87.pdf. [Accedido: 11-may-2018]

[4] Dr. Chalasani Srinivas, Mohan Kumar.Ch, “Toxic gas detection and monitoring utilizing Internet of Things”, International Journal of Civil Engineering and Technology (IJCIET), Volume 8, Issue 12, December 2017, pp. 614–622. [En línea]. Disponible en: http://www.iaeme.com/MasterAdmin/UploadFolder/IJCIET_08_12_067/IJCIET_08_12_067.pdf. [Accedido: 12-may-2018]

[5] Leanne McRae, Katie Ellis and Mike Kent, “The Internet of Things (IoT): Education and Technology”, Curtin University, 2018. [En línea]. Disponible en:

https://www.ncsehe.edu.au/wp-content/uploads/2018/02/IoTEducation_Formatted_Accessible.pdf. [Accedido: 13-may-2018]

[6] Alvaro Everlet, Javier Pastor, Introducción al Internet de las Cosas, Construyendo un Proyecto de IOT, Carriots, Universidad Rey Juan Carlos, Noviembre 2013. [En línea]. Disponible en: https://www.carriots.com/newFrontend/img-carriots/press_room/Construyendo_un_proyecto_de_IOT.pdf. [Accedido: 14-may-2018]

[7] W3schools, "Node.js Introduction". [En línea]. Disponible en: https://www.w3schools.com/nodejs/nodejs_intro.asp. [Accedido: 15-may-2018]

[8] Kiran Malvi, "The Positive and Negative Aspects of Node.js Web App Development", 18 de Junio 2018. [En línea]. Disponible en: https://www.mindinventory.com/blog/pros-and-cons-of-node-js-web-app-development/. [Accedido: 16-may-2018]

[9] MDN web docs, "Express/Node introduction". [En línea]. Disponible en: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction. [Accedido: 16-may-2018]

[10] Tymets Volodymyr, "Express.js Mobile App Development: Pros and Cons for Developers", 08 de Junio 2017. [En línea]. Disponible en: https://apiko.com/blog/express-mobile-app-development/. [Accedido: 16-may-2018]

[11] AngularJS, "What Is AngularJS?". [En línea]. Disponible en: https://docs.angularjs.org/guide/introduction. [Accedido: 17-may-2018]

[12] AltexSoft, "The Good and the Bad of Angular Development", 29 de Septiembre 2017. [En línea]. Disponible en: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/. [Accedido: 17-may-2018]

[13] MongoDB, "Introduction to MongoDB". [En línea]. Disponible en: https://docs.mongodb.com/manual/introduction/. [Accedido: 18-may-2018]

[14] Data Flair, "Advantages of MongoDB | Disadvantages of MongoDB", 17 de Abril 2018. [En línea]. Disponible en: https://data-flair.training/blogs/advantages-of-mongodb/. [Accedido: 18-may-2018]

[15] Python, "General Python FAQ". [En línea]. Disponible en: https://docs.python.org/3/faq/general.html#id2. [Accedido: 19-may-2018]

[16] DataFlair, "Advantages and Disadvantages of Python Programming Language", Enero 2018. [En línea]. Disponible en: https://data-flair.training/blogs/advantages-and-disadvantages-of-python/. [Accedido: 19-may-2018]

[17] Ajay, Dr. Baswaraj Gadgay, Veeresh Pujari, Pallavi B.V, "IOT Based Smart Environmental Monitoring Using Arduino", International Journal for Research in Applied Science & Engineering Technology (IJRASET), Vol. 5, Junio 2017. [En línea]. Disponible en: https://www.ijraset.com/fileserve.php?FID=8144. [Accedido: 20-may-2018]

[18] Dr. Salah Talha Babiker Edris, "Monitoring of Taif University Campus by using The Internet of Things Techniques", International Journal of Applied Engineering Research, Vol. 12, 2017, pp. 14377-14381. [En línea]. Disponible en: https://www.ripublication.com/ijaer17/ijaerv12n24_53.pdf. [Accedido: 20-may-2018]

[19] Jesús Castillo Izquierdo, Diseño e implementación de un dispositivo IoT de bajo coste para entornos agrícolas, Escuela Tecnica Superior de Ingenierías Informática y de Telecomunicación, Granada, Junio 2017. [En línea]. Disponible en:

http://wpd.ugr.es/~jorgenavarro/thesis/2017_TFG_JesusCastilloIzquierdo.pdf. [Accedido: 21-may-2018]

[20] Somansh Kumar, Ashish Jasuja, “Air Quality Monitoring System Based on IoT using Raspberry Pi”, International Conference on Computing, Communication and Automation, 2017. [En línea]. Disponible en: https://zapdf.com/air-quality-monitoring-system-based-on-iot-using-raspberry-p.html. [Accedido: 21-may-2018]

[21] C. Balasubramaniyan, D. Manivannan, “IoT Enabled Air Quality Monitoring System (AQMS) using Raspberry Pi”, Indian Journal of Science and Technology, Vol 9(39), Octubre 2016. Disponible en: http://www.indjst.org/index.php/indjst/article/view/90414/74439. [Accedido: 21-may-2018]

[22] Jesús Castillo Izquierdo, Diseño e implementación de un dispositivo IoT de bajo coste para entornos agrícolas, Escuela Tecnica Superior de Ingenierías Informática y de Telecomunicación, Granada, Junio 2017. [En línea]. Disponible en: http://wpd.ugr.es/~jorgenavarro/thesis/2017_TFG_JesusCastilloIzquierdo.pdf. [Accedido: 21-may-2018]

[23] Roger S. Pressman, Ingeniería del software, Un enfoque práctico, Séptima Edición, Mc Graw Hill, México, D. F., 2010.

[24] Microsoft Azure, Uso de claves SSH con Windows en Azure, Abril 2018. [En línea]. Disponible en: https://docs.microsoft.com/es-es/azure/virtual-machines/linux/ssh-from-windows. [Accedido: 22-may-2018]

[25] Yhorby Matias Heredia, Microsoft Azure, Creando una máquina virtual con Linux, Junio 2016. [En línea]. Disponible en: http://ymatias.com/2016/06/07/microsoft-azure-creando-una-maquina-virtual-con-linux/. [Accedido: 22-may-2018]

[26] DigitalOcean, “How to Install MongoDB on Ubuntu 18.04”, 7 de Junio 2018. [En línea]. Disponible en: https://www.digitalocean.com/community/tutorials/how-to-install-mongodb-on-ubuntu-18-04. [Accedido: 23-may-2018]

[27] DigitalOcean, “How to Install Node.js on Ubuntu 18.04”, 27 de Abril 2018. [En línea]. Disponible en: https://www.digitalocean.com/community/tutorials/how-to-install-node-js-on-ubuntu-18-04. [Accedido: 23-may-2018]

[28] HANWEI ELECTRONICS CO., LTD, “Technical Data MQ-7 Gas Sensor”. Disponible en: https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf. [Accedido: 24-may-2018]

[29] Jorge Anibal Suntaxi Pichuasamin, Diseño y construcción de un prototipo portátil de monitoreo ambiental, mediante un Sistema autónomo de adquisición de datos portátil con comunicación USB hacia un PC, Facultad de Ingeniería Eléctrica y Electrónica, Escuela Politécnica Nacional, Enero 2015.

[30] Mónica Tejerina Fernández, UF 1909 – Toma de muestras de contaminantes atmosféricos, Editorial Eleraning S.L., Edición: 5.1, pp. 267-268, 2015.

[31] Valeria Díaz Suárez, Informe de la calidad del aire del Distrito Metropolitano de Quito, Secretaría de Ambiente, Alcaldía Metropolitana de Quito, Abril 2015, pp. 4-9. [En línea]. Disponible en: http://www.quitoambiente.gob.ec/ambiente/images/Secretaria_Ambiente/red_monitoreo/informacion/iqca_2014.pdf. [Accedido: 26-may-2018]

[32] Ministerio del Ambiente, Norma Ecuatoriana de Calidad del Aire, Libro VI, Anexo 4, Junio 2011, pp. 6. [En línea]. Disponible en:

http://www.quitoambiente.gob.ec/ambiente/images/Secretaria_Ambiente/red_monitoreo/informacion/norma_ecuato_calidad.pdf. [Accedido: 26-may-2018]