



Reto: Entrega Final Implementación del Internet de las Cosas

Prof. Sergio Ruiz, Rolando Vallejo, Luis Montesinos

01ero de diciembre de 2020.

Tecnológico de Monterrey Campus Ciudad de México

A01657396 Brian Isaac Serranía Mendoza

A01023684 Iñigo Enrique Zepeda Ceballos

A01652617 Diego Enrique Jiménez Urgell

A01657645 Diego Eduardo Rodríguez Guzmán

A01658904 Bruno Passarette Santos

Contenido:

Introducción	03
Resumen Ejecutivo	03
Análisis Descriptivo	04
Implementación del diseño conceptual	06
Requerimientos funcionales y no funcionales	06
Administración de alcance, tiempo y presupuesto	07
Flujo de los datos en la solución	08
Integración de componentes	10
Sistemas Digitales	14
Validación de correcto funcionamiento	14
Conteo de ocurrencias	16
Maquina de estados de riesgos	17
Evidencia de correcta conexión con la nube	22
Bases de datos	23
DB en MySQL con datos	23
DB completa hasta 3NF	25
Diagrama ER	28
Modelo Relacional	29
Recursos de un sistema computacional	30
Análisis del uso de distintas arquitecturas	30
Análisis de la ejecución de procesos asíncronos del sistema de IoT	31
Aplicación Productor – Consumidor	31
Temas del Módulo 3	33
Justificación de la importancia del aprendizaje	34
Diseño Interactivo	38
Actualización final de la página web	38
Prueba Heurística	41
Administración de Proyectos	43
Seguimiento Puntual de las actividades	43

Internet de las cosas (IoT)	45
Documentación de la topología y el proceso	45
Diagrama de direcciones y protocolos	54
Explicación de interacción entre los dispositivos	54
Inclusión de sensores e interpretación de datos	55
Conclusiones	56
Importancia de los temas de cada módulo	56
Aprendizajes por integrante	58
Referencias	61

Resumen Ejecutivo:

Problema: La Ciudad de México es considerada una de las ciudades con más habitantes, esto va de la mano con una gran contaminación en el ambiente, problema que día con día va en aumento y afecta a cada uno de los residentes de esta. Para entender un poco más de este problema es necesario entender que es la contaminación y esta se denomina el conjunto de sustancias químicas que son dañinas tanto para el medio ambiente como para los creadores de todo esto, los seres humanos; principalmente es causada por la quema de combustibles, residuos de fábricas, emisiones resultantes de la combustión de automóviles, entre otros. Aunque es importante mencionar que este problema se encuentra en la mayoría de las ciudades, la capital del país es la 3era con el aire más contaminado del mundo.

Solución: Dentro de las distintas soluciones que podemos encontrar para tratar el tema de la contaminación proponemos el análisis en tiempo real en las distintas instalaciones, de esta manera se empezarán con estudios que nos permitan determinar los horarios y zonas de la Ciudad de México que más contaminan y en consecuencia determinar los agentes que los causan, si es por tráfico mejorar las alternativas viales en las horas especificadas por el estudio o si es por una fábrica ubicar si cuenta con los permisos correspondientes y los procesos del tratado de contaminantes pertinentes.

Propuesta: Aprovechando el crecimiento exponencial en cuanto a temas de tecnología se crea un sistema de red con sensores interconectados que pueden ser consultados en cualquier lugar y a cualquier hora del día para tener un control y estudio específico, esto no solo amortizaría costos y gastos de renta, personal de oficinas y oficinas en sí, sino que también se optimizaría el tiempo pues desde solo un lugar se puede monitorizar toda la Ciudad de México en conjunto con los datos sobre la calidad del aire proporcionado por el Sistema de Monitoreo Atmosférico.

Beneficios: Los principales beneficiados con esta nueva tecnología y software son los habitantes de la Ciudad, recordamos que la OMS reporta poco más de 4 millones de muertes prematuras a nivel internacional dentro de las cuales 18% de estas son causadas por enfermedades pulmonares y cáncer de pulmón consecuencia de la contaminación ambiental, seguido del fortalecimiento que se genera en los objetivos de desarrollo sustentable de la ONU para el 2030 en México.

3.1 Análisis descriptivo:

Objetivos y Alcances: El proyecto se centra en la instalación de sensores interconectados y un software completamente nuevo que permita el análisis y estudio de los contaminantes que hay en el ambiente, se trabajará en conjunto con el Sistema de Monitoreo Atmosférico con datos proporcionados por esa institución y con los datos y mediciones obtenidas vía IoT de nuestros sensores, de esta manera se tiene un control total de los horarios y ubicaciones que generan más partículas contaminantes, en consecuencia se pueden hacer planes y campañas estratégicas para determinar causantes y buscar la manera de reducirlos.

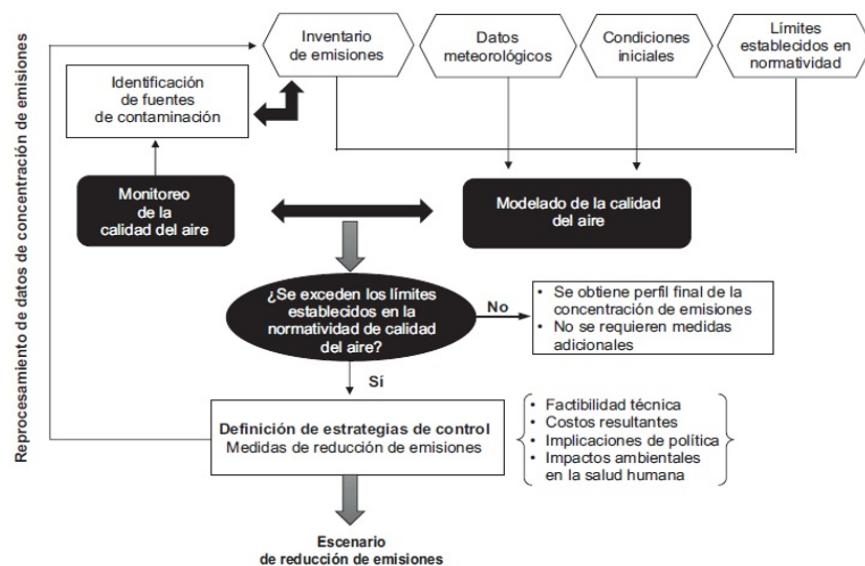
Antecedentes: Según estudios de la Organización Mundial de la Salud, la contaminación es causante de poco más de 4 millones de muertes prematuras anuales a nivel mundial, sin embargo, es importante mencionar que el hecho de ser una de las ciudades más habitadas del planeta afecta proporcionalmente pues se disparan los contaminantes que se encuentran en el aire, otro punto es lo complicado que se ha vuelto la movilidad por el mismo tema, generando tráfico en ciertas horas en específico y aumentando las emisiones de gases por los automóviles que tratan de llegar a su siguiente destino.

Las últimas acciones que ha tenido el gobierno mexicano han derivado en la disminución de contaminantes y aunque definitivamente no es suficiente, se han encontrado varias áreas de oportunidad dentro de las cuales están el seguir incentivando por elegir compartir los automóviles, usar el transporte público, cambiar las flotillas de camiones por aquellos que sean a base de electricidad, mejorar la planeación vial que hay en ciertas horas del día. Dentro de las áreas de oportunidad que exploramos, encontramos un punto importante que no se ha mejorado en mucho tiempo y es el hecho de cómo se han manejado, estudiado e investigado estos datos y es que los planes que hasta el día de hoy encontramos son a base de dos análisis del medio ambiente al día y esto genera mucha fluctuación con los datos y no podemos determinar con precisión en qué zonas y horarios debemos enfocarnos para reducirlos.

Para todas las situaciones anteriormente mencionadas se adapta nuestro proyecto junto con el software pues ayudará de manera directa los estudios ambientales que se vayan generando, demostrando en dónde se necesita poner más atención y esfuerzo para combatir este tema de manera tajante.

Descripción del problema: Los gases contaminantes y partículas químicas que podemos encontrar en ocasiones en el ambiente son sumamente tóxicas, la mala planeación y la falta de estrategias para atacar estos problemas han incentivado el aumento de los mismos, esto ha generado una serie de complicaciones y problemas en los habitantes de la zona en la que estamos trabajando actualmente (en este caso la Ciudad de México) aunque en realidad este problema es a nivel mundial donde derivan desde afectaciones en la salud, económicos y ambientales.

Modelo Actual: Conforme el análisis que realizamos sobre el modelo actual de la toma de datos y muestreos podemos observar que se lleva un inventario de emisiones en cada toma de cantidades, conformado por los datos meteorológicos, las condiciones iniciales y los límites establecidos en la normatividad y con eso se genera el modelado de la calidad del aire, sin embargo si en algún punto esos datos exceden la normatividad apenas se hace el estudio para identificar las fuentes de contaminación y después los costos, impactos en la salud humana, entre otras cosas y este proceso se repite en cada toma de datos cuando se excede de la normatividad, un punto a favor de este proyecto es empezar a crear programas directos para aminorar el tiempo con el que se toman decisiones, además de poder estar en constante monitoreo en caso que algo se salga de los estándares comunes, así no se tiene que hacer de cero y dependiendo de cada muestreo.



Necesidades del proyecto: Para implementar esta propuesta, se requiere el uso de distintos recursos computacionales, puesto que es un sistema con muchos componentes.

En primer lugar, se necesitan los sensores físicos, aunque en este caso no se tiene acceso a hardware. Por lo tanto, los sensores se van a simular utilizando Packet Tracer y Symulink. La información recopilada por estos sensores “virtuales” será transferida a una base de datos. Por lo tanto, se necesita un DBMS, en este caso MySQL y un protocolo de transferencia de información. Además de esto, para la página web se requiere conocimiento de PHP, para recopilar la información de la base de datos, y un servidor que funcione como host de la página.

Dejando de lado los tecnicismos, se considera hacer un software que reciba todos los datos percibidos por los sensores y los almacenará en una base de datos virtual para poder acceder a ellos de una forma sencilla y amigable, de manera automática se plantea poder hacer con estos datos unas gráficas que faciliten el estudio de un cierto periodo de tiempo y ver los avances a los que se han llegado gracias a la implementación de este proyecto; no sin olvidar que las estrategias y planes que se implementen pueden irse mejorando y actualizando conforme la zona en la que se esté aplicando dependiendo directamente de la toma de datos.

3.2 Realizar la implementación del diseño conceptual, para lograr la integración de los componentes en un prototipo del sistema de IoT.

3.2.1 Mencionar los requerimientos funcionales y no funcionales.

Funcionales:

- ∅ Medir las concentraciones de NO₂, CO, SO₃, O₃, PM2.5, PM10 del ambiente. Como es imposible realizarlo de manera física, se deben modelar los sensores en Packet Tracer.
- ∅ Utilizar los datos acerca de la calidad del aire proporcionados por el Sistema de Monitoreo atmosférico, y calcular el promedio utilizando un modelo de Simulink.
- ∅ Diseñar un sistema de información utilizando la stack LAMP, para que los datos sean almacenados de acuerdo con una estampa de tiempo y sea posible acceder a ellos.
- ∅ Transferir las lecturas de los sensores a la base de datos utilizando un paquete de manejo de archivos en Matlab, implementando técnicas de verificación de integridad de los datos, como paridad y checksum.

- ∅ Diseñar una página web con HTML, CSS y PHP que permita acceder a los datos según las necesidades del usuario.

No Funcionales:

- ∅ Diseñar una página web que sea amigable con el usuario, con un diseño estético y que tenga en cuenta las necesidades de los usuarios potenciales.
- ∅ Lograr una transferencia rápida de la información desde los sensores a la base de datos.
- ∅ Garantizar la disponibilidad del sitio web, mediante el uso de un servidor que almacene la base de datos y los scripts de la página web.

3.2.2 Describir la manera en que se administraron las dimensiones: Alcance, Tiempo y Presupuesto para cumplir con los requerimientos.

Según el PMI (Project Management Institute) la gestión de proyectos incluye tres aspectos fundamentales que son: el alcance, el tiempo y la línea de costos.

- ∅ **Alcances:** En este apartado se encontrarán detalles del proyecto, estructura de este y el desglose de cómo se llevaron a cabo con las actividades, la estimación y gestión del cronograma.

Dentro de los alcances que se entregan al final de este proyecto se considera una simulación funcional de registro de datos en sensores, almacenamiento en una base de datos, diseño conceptual para la futura implementación por parte de la Comisión Ambiental de la Megalópolis (CAME) en caso de ser aprobado el proyecto, todo esto a través de plataformas tipo IoT y sistemas digitales que pueden ser consultados a cualquier hora y momento por personal previamente capacitado.

El orden del proyecto fue establecido a través de diagramas de Gantt, juntas administrativas y el apoyo de asesores externos a nuestra empresa.

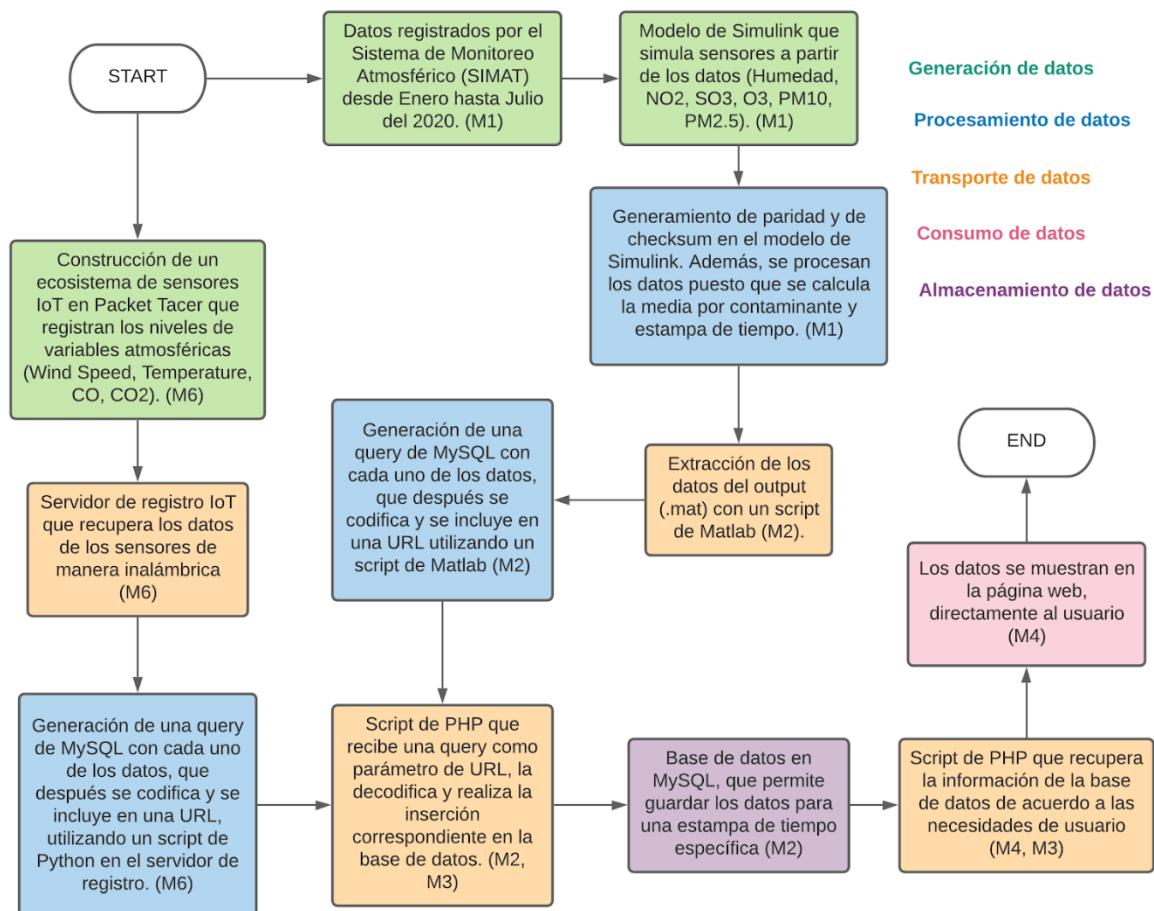
- ∅ **Tiempo:** Aquí podremos observar las fechas de inicio y de finalización del proyecto en cuestión. El tiempo que se menciona en este punto tiene que ver únicamente por la creación e implementación de los recursos necesarios para la creación de la simulación en cuestión, dentro de las fechas establecidas se menciona el inicio del proyecto cuya fecha fue el 20 de septiembre de 2020 al 27 de noviembre de 2020.

∅ **Presupuesto:** En este caso es la versión aprobada del presupuesto que se entregó por cada una de las fases del proyecto.

Al ser una simulación del correcto funcionamiento de los sensores y la base de datos que será implementada no se requirió de un presupuesto inicial, es decir nuestro presupuesto es de \$0 MXN.

3.2.3 Describir el flujo de los datos en la solución del reto con énfasis en la manera en que los temas vistos en clase (cada módulo) se usan para permitir que los datos sean utilizados en implementación IoT.

Para lograr cumplir con estos requerimientos y proporcionar una experiencia satisfactoria al usuario, se debe considerar la integración de varios componentes distintos, desde Packet Tracer y Matlab, hasta scripts en PHP. El siguiente diagrama muestra el flujo de los datos en nuestra propuesta de solución al reto:



La generación de datos consiste en la modelación de los sensores IoT, y se compone de dos sistemas diferentes. Por un lado, está Cisco Packet Tracer, que permite modelar redes y sistemas de hardware. El software incluye un simulador del entorno, que modifica los valores de ciertas variables ambientales dependiendo de la hora del día y de la presencia de otros dispositivos (como vehículos que generan CO₂). Utilizando los conocimientos del módulo de IoT (6) se modelaron cuatro sensores (CO, CO₂, temperatura, velocidad del viento) y se conectaron con un servidor de registro mediante un protocolo TCP/IP. Además, los datos de las lecturas de cada sensor se envían con protocolo HTTP. Posteriormente, mediante un script de Python se crea una query para enviar los datos a la base de datos.

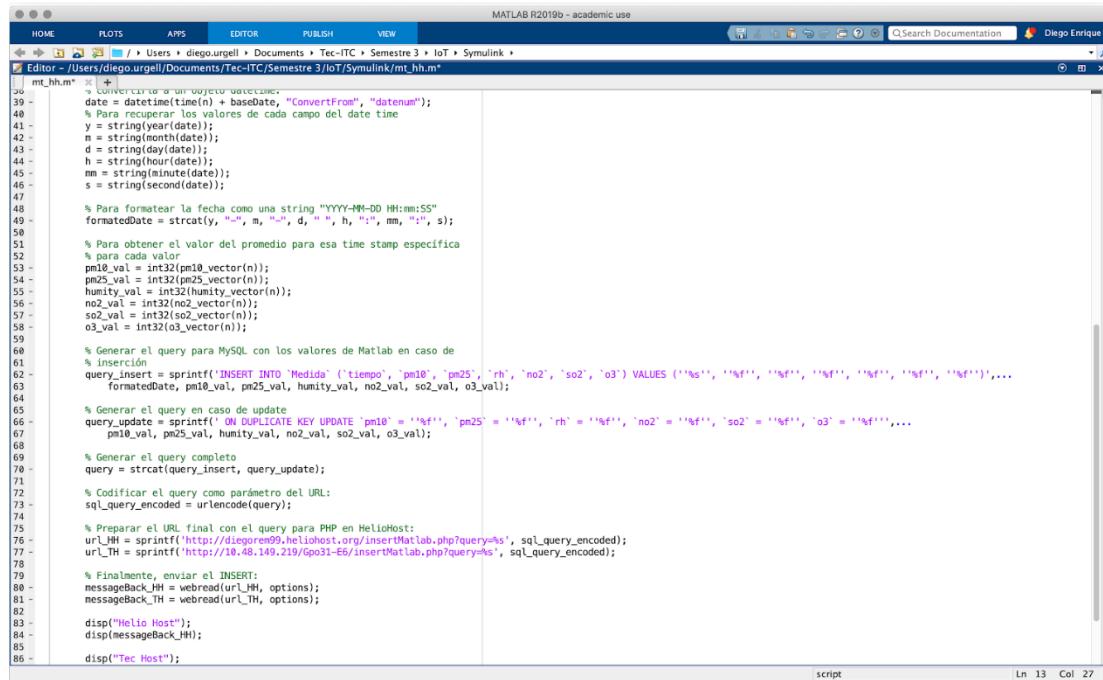
De manera similar, con los datos del Sistema de Monitoreo Atmosférico se simularon sensores IoT, mediante un modelo de Simulink. Utilizando el conocimiento obtenido en el módulo de Sistemas Digitales, se modelaron sensores que leen la información del SIMAT, y posteriormente el modelo obtiene el promedio de un conjunto de lecturas, y además calcula la paridad y el checksum como medidas de integridad. Los datos se registran como una serie de tiempo en un archivo externo.

Para insertar la información de los sensores en la base de datos, se escribió un script de PHP que recibe una query con sintaxis de MySQL en un formato especial como parámetro de su URL, una vez que recupera y decodifica la query, la ejecuta y realiza la inserción. Para realizar esto se combinó el conocimiento del módulo de Bases de Datos y de Administración de Recursos de un Sistema Computacional. En el mismo servidor que almacena el script de PHP para insertar datos, se colocarán scripts de HTML y PHP que serán una página web, esta será la interfaz por medio de la cual el usuario podrá interactuar con el sistema de información. La página web tendrá mecanismos para que el usuario pueda generar reportes personalizados considerando fechas y variables específicas, todo esto será manejado por medio de PHP.

3.3 Realizar la implementación del diseño conceptual, para lograr la integración de los componentes en un prototipo del sistema de IoT. Así como la implementación de los:

∅ Programas que los controlen:

→ Script de Matlab:

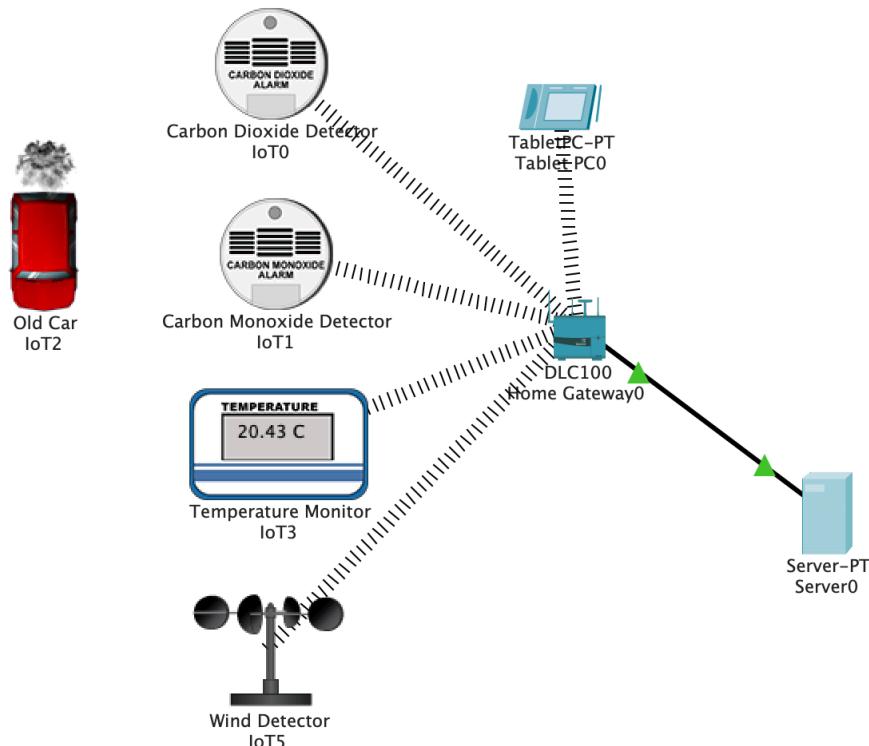


```

% Convertir a un objeto vectorial
date = datetime(n) + baseDate, "ConvertFrom", "datenum");
% Para recuperar los valores de cada campo del date time
y = string(year(date));
n = string(month(date));
d = string(day(date));
h = string(hour(date));
m = string(minute(date));
s = string(second(date));
%
% Para formatear la fecha como una string "YYYY-MM-DD HH:mm:ss"
formattedDate = strcat(y, "-", n, "-", d, " ", h, ":", m, ":", s);
%
% Para obtener el valor del promedio para esa time stamp específica
% para cada valor
pm10_val = int32(pm10_vector(n));
pm25_val = int32(pm25_vector(n));
humidity_val = int32(humid_vector(n));
no2_val = int32(no2_vector(n));
so2_val = int32(so2_vector(n));
o3_val = int32(o3_vector(n));
%
% Generar el query para MySQL con los valores de Matlab en caso de
% inserción
query_insert = sprintf('INSERT INTO `Medida` (`tiempo`, `pm10`, `pm25`, `rh`, `no2`, `so2`, `o3`) VALUES (''%f'', ''%f'', ''%f'', ''%f'', ''%f'', ''%f'', ''%f'')...', ...
    formattedDate, pm10_val, pm25_val, humidity_val, no2_val, so2_val, o3_val);
%
% Generar el query en caso de update
query_update = sprintf(' ON DUPLICATE KEY UPDATE `pm10` = ''%f'', `pm25` = ''%f'', `rh` = ''%f'', `no2` = ''%f'', `so2` = ''%f'', `o3` = ''%f''...', ...
    pm10_val, pm25_val, humidity_val, no2_val, so2_val, o3_val);
%
% Generar el query completo
query = strcat(query_insert, query_update);
%
% Codificar el query como parámetro del URL:
sql_query_encoded = urldecode(query);
%
% Preparar el URL final con el query para PHP en HelioHost:
url_HH = sprintf('http://diepreno99.heliohost.org/insertMatlab.php?query=%s', sql_query_encoded);
url_TH = sprintf('http://10.48.149.219/Gp031-E9/InsertMatlab.php?query=%s', sql_query_encoded);
%
% Finalmente, enviar el INSERT:
messageBack_HH = webread(url_HH, options);
messageBack_TH = webread(url_TH, options);
%
disp("Helio Host");
disp(messageBack_HH);
%
disp("Tec Host");

```

→ Sistema de sensores IoT:



→ Script de python en Packet Tracer:

```

11~ def main():
12     http = RealHTTPClient()
13     http.onDone(onHTTPDone)
14
15     day = 1
16     hour = 0
17
18~ while day <= 30:
19
20     print(" ")
21     co2_level = Environment.get("CO2")
22     co_level = Environment.get("CO")
23     ws_level = Environment.get("Wind Speed")
24     temp_level = Environment.get("Ambient Temperature")
25     dt = ("2020-01-{0} {1}:59:00")
26     dt = dt.format(day, hour)
27
28     query_insert = "INSERT INTO `Medida` (`tiempo`, `co`, `co2`, `wsp`, `tmp`) VALUES ('{0}', {1}, {2}, {3}, {4}) "
29     query_insert = query_insert.format(dt, co_level, co2_level, ws_level, temp_level)
30
31     query_update = "ON DUPLICATE KEY UPDATE `co` = {0}, `co2` = {1}, `wsp` = {2}, `tmp` = {3}"
32     query_update = query_update.format(round(co_level, 2), round(co2_level, 2), round(ws_level, 2), round(temp_level, 2))
33
34     query = query_insert + query_update
35     print(query)
36
37     url_hh = 'http://diegorem99.heliohost.org/insertMatlab.php?query='
38     url_hh = url_hh + query
39
40     url_th = 'http://10.48.149.219/Gpo31-E6/insertMatlab.php?query='
41     url_th = url_th + query
42
43     print(dt)
44
45     print("Helio Host")
46     http.get(url_hh)
47
48     print("Tec Host")
49     http.get(url_th)
50
51     hour = (hour + 1)%24
52
53~ if hour == 23:
54     day += 1

```

→ Script de PHP:

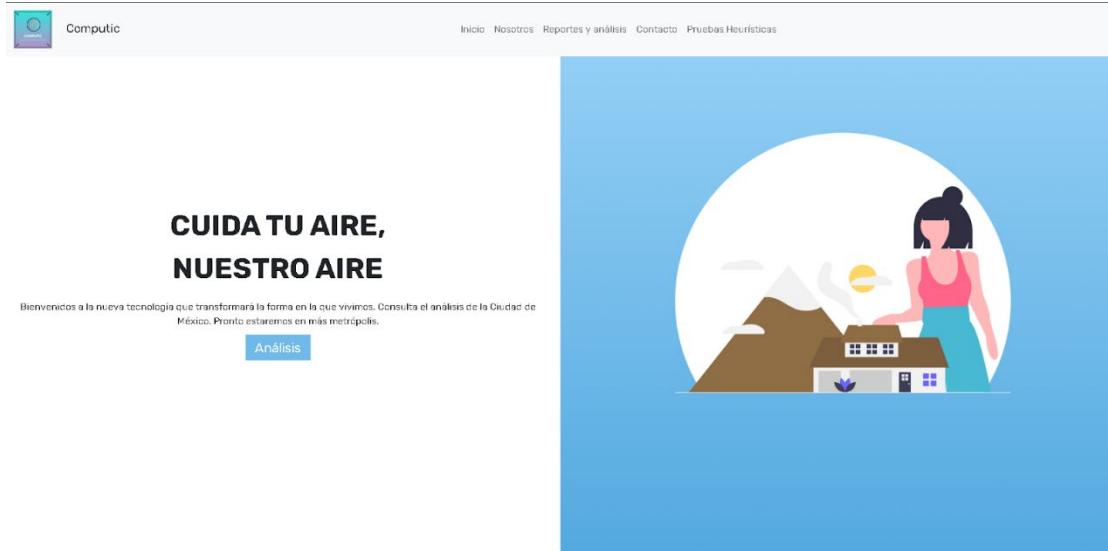
```

1  <?php
2
3  $servername = "10.48.149.219";
4  $username = "gpo31_equipo6";
5  $password = "gpo31_equipo61234";
6  $dbname = "gpo31-e6";
7
8  $param_query = $_GET['query'];
9
10 if($param_query)
11 {
12     $sql = urldecode($param_query);
13     echo $sql;
14     // Create connection
15     $conn = new mysqli($servername, $username, $password, $dbname);
16
17     // Check connection
18     if ($conn->connect_error)
19     {
20         die("Connection failed: " . $conn->connect_error . "<br/>");
21     }
22     echo "Connected successfully";
23
24     if($conn->query($sql) === TRUE)
25     {
26         echo "Inserted successfully";
27     }
28     else
29     {
30         echo "Error: " . $sql . $conn->error;
31     }
32
33     $conn->close();
34 }
35 else
36 {
37     echo "Query Error. Aborting.";
38 }
?>

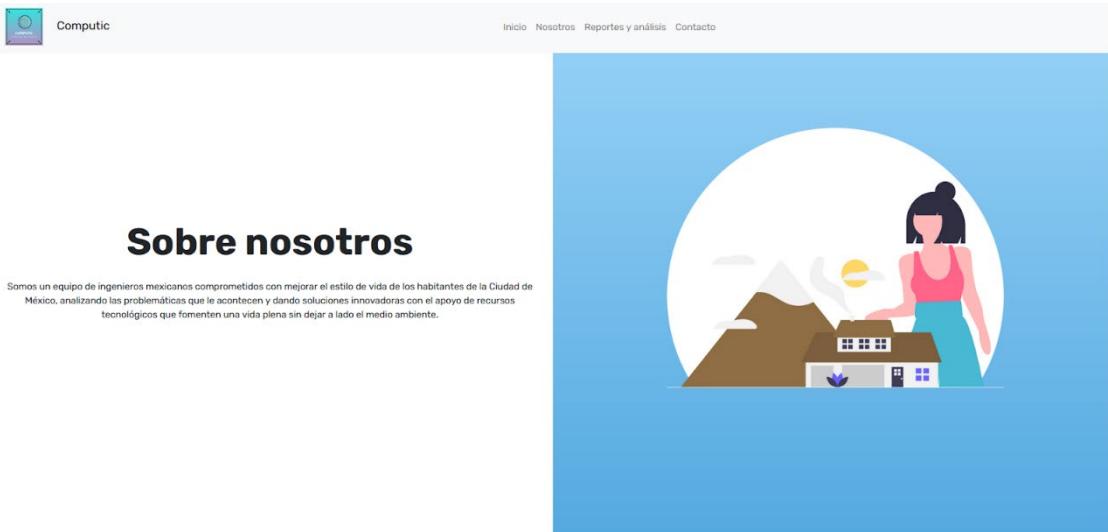
```

∅ El diseño centrado del usuario

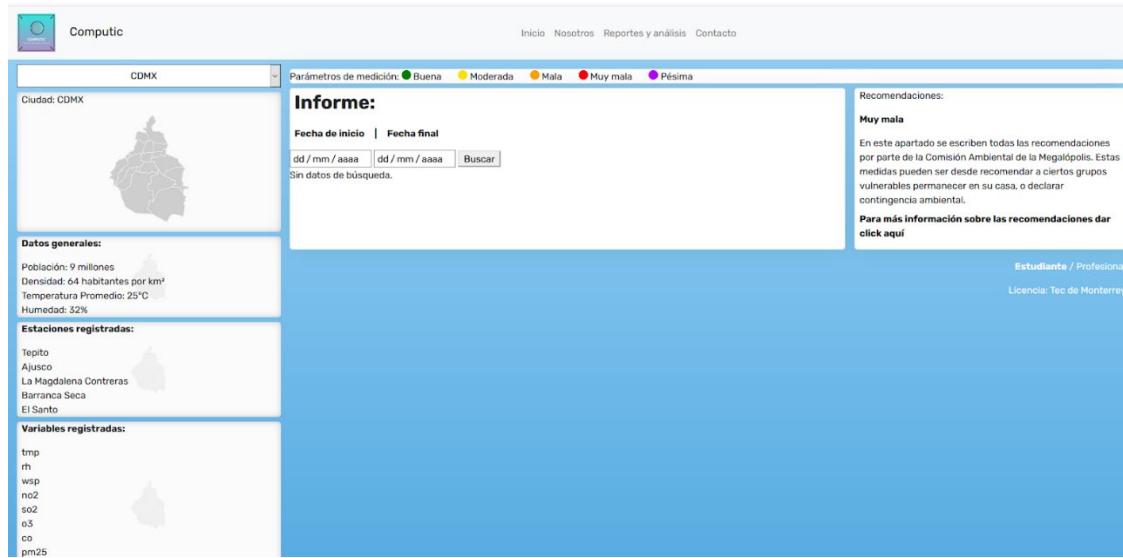
- Página de inicio: Aquí el usuario podrá navegar a las diferentes pestañas que se encuentran en la página, el diseño de esta se centra en el fácil acceso e interfaz que puede ser ocupado por el cliente final. (El diseño, fuentes y transiciones podrán ser modificadas conforme el cliente lo requiera).



- Página sobre nosotros: El usuario puede leer una pequeña descripción sobre la empresa, de esta manera se sentirá con más confianza hacia la empresa desarrolladora de software, además que podrá conocer un poco más de nosotros y de las aptitudes que podemos llegar a desarrollar en este u otro proyecto que tengan en mente.



→ Página del informe y análisis: Dicha pestaña será descrita más adelante. Se visualizan los datos obtenidos desde la base de datos, directamente en la página web.



Computic

Parámetros de medición: ● Buena ○ Moderada ○ Mala ● Muy mala ● Pésima

Informe:

Fecha de inicio | Fecha final
 dd / mm / aaaa dd / mm / aaaa
 Sin datos de búsqueda.

Datos generales:
 Población: 9 millones
 Densidad: 64 habitantes por km²
 Temperatura Promedio: 25°C
 Humedad: 52%

Estaciones registradas:
 Tepito
 Ajusco
 La Magdalena Contreras
 Barranca Seca
 El Santo

Variables registradas:
 tmp
 rh
 wsp
 no2
 so2
 o3
 co
 pm25

Recomendaciones:
Muy mala
 En este apartado se escriben todas las recomendaciones por parte de la Comisión Ambiental de la Megalópolis. Estas medidas pueden ser desde recomendar a ciertos grupos vulnerables permanecer en su casa, o declarar contingencia ambiental.
[Para más información sobre las recomendaciones dar click aquí](#)

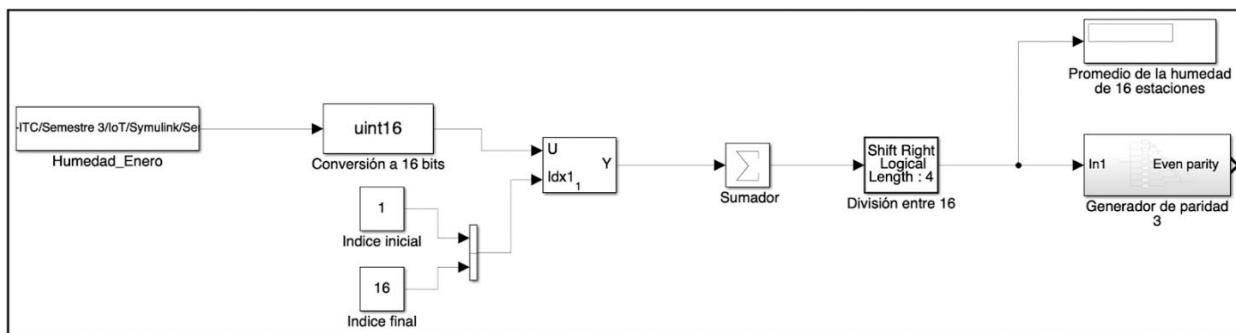
Estudiante / Profesional
 Licencia: Tec de Monterrey

1. Sistemas Digitales

4.1.1 Validar el correcto funcionamiento de Humedad Relativa y Temperatura.

Para enviar la información a la base de datos en primer lugar debe haber un pre procesamiento de las lecturas. En lugar de enviar los dieciséis datos de cada cantidad para cada estampa de tiempo (cada una proviene de una de las estaciones), se calcula el promedio de las lecturas para cada cantidad, y esa es la información para enviar. Por lo tanto, se debe implementar un modelo para calcular el promedio de cierto número de datos.

El siguiente diagrama es un modelo de Simulink que cumple con ese objetivo:



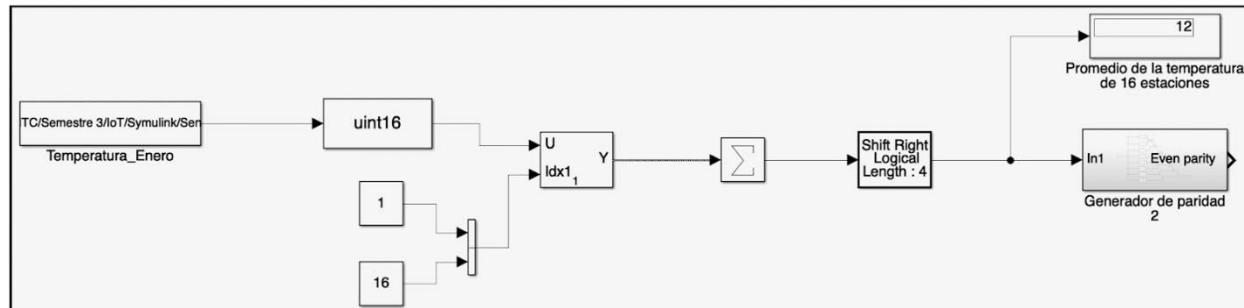
Para cada cantidad (NO₂, CO, SO₃, O₃, PM2.5, PM10, temperatura, humedad, WSP) hay un archivo por mes que contiene las mediciones de varias estaciones (alrededor de 20) por cada estampa de tiempo. Entonces, el primer paso es recuperar un vector que contiene las lecturas de todas las estaciones para un momento específico. En el modelo mostrado, el bloque "Humedad Enero" recibe un vector con 26 lecturas de humedad, ya que hay veintiséis estaciones que reportan la humedad. Posteriormente, el bloque "Conversión a 16 bits" se encarga de generar enteros sin signo de 16 bits, para evitar que se presente un desborde al momento de sumar los datos.

El siguiente es un bloque selector, que recibe los datos del vector y los selecciona uno por uno a partir del Índice inicial y el Índice final proporcionados. Solo se incluyen dieciséis datos en la suma dado que el mecanismo que se usará para la división únicamente permite dividir entre potencias de dos. El bloque sumador es el que se encarga de acumular la suma de los datos. Después, el bloque de "División entre 16" realiza un shift a la derecha de longitud 4 en los bits de la suma de los datos . Esto implica una división entre dieciséis, dado que un solo shift a la derecha es una división entre dos (Fisher et al, 2003). Finalmente, ese dato se muestra en el display.

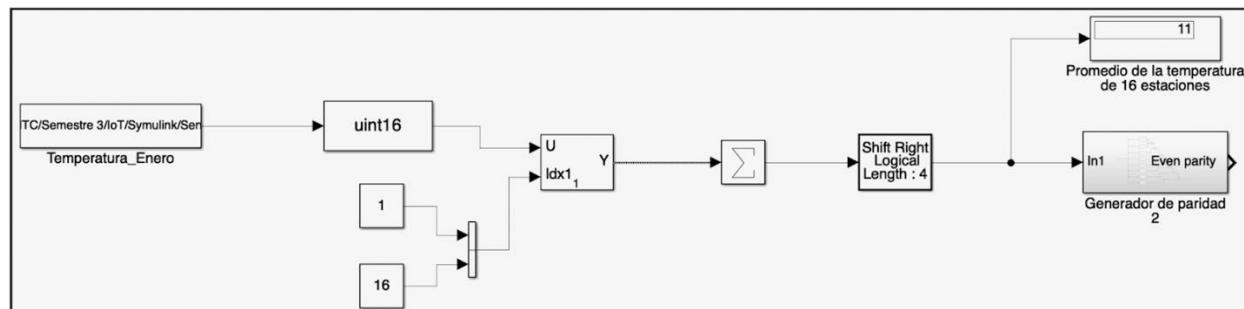
Para probar el modelo y verificar su correcto funcionamiento, se calculó el promedio de los datos de la temperatura para el mes de enero, para las primeras tres estampas de tiempo. La siguiente tabla muestra los datos reportados por los sensores, así como el promedio preciso de las lecturas de las dieciséis estaciones:

ACO	AJM	AJU	BJU	MON	CHO	CUA	CUT	FAC	GAM	HGM	INN	MER	MGH	MPA	NEZ	PROM
12	13	14	14	15	14	12	8	10	16	15	5	16	15	13	14	12.875
11	11	14	14	15	13	12	8	9	15	5	5	15	14	13	14	11.75
11	11	14	14	15	13	12	7	9	14	5	5	14	14	13	13	11.5

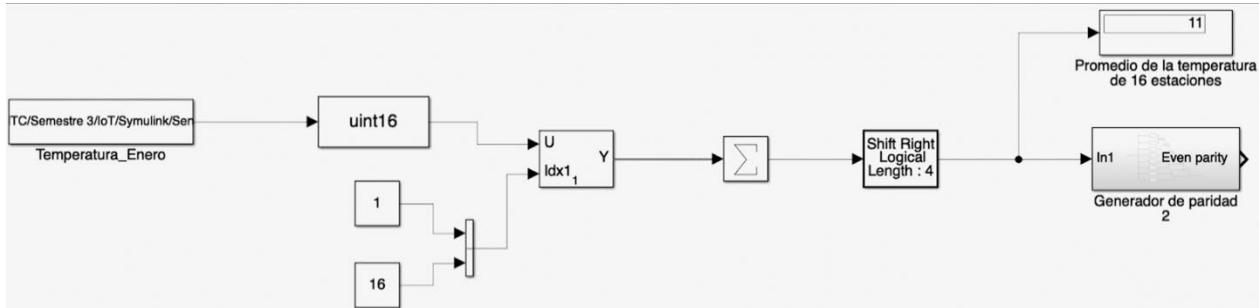
El modelo implementado debería de calcular la parte entera para cada uno de los tres conjuntos de lecturas. El primer promedio generado es de 12, como se puede ver en la siguiente captura. Esto coincide con el dato preciso mostrado en la tabla:



Para la segunda estampa de tiempo, el promedio es de 11, lo que también coincide con la parte entera del promedio preciso:



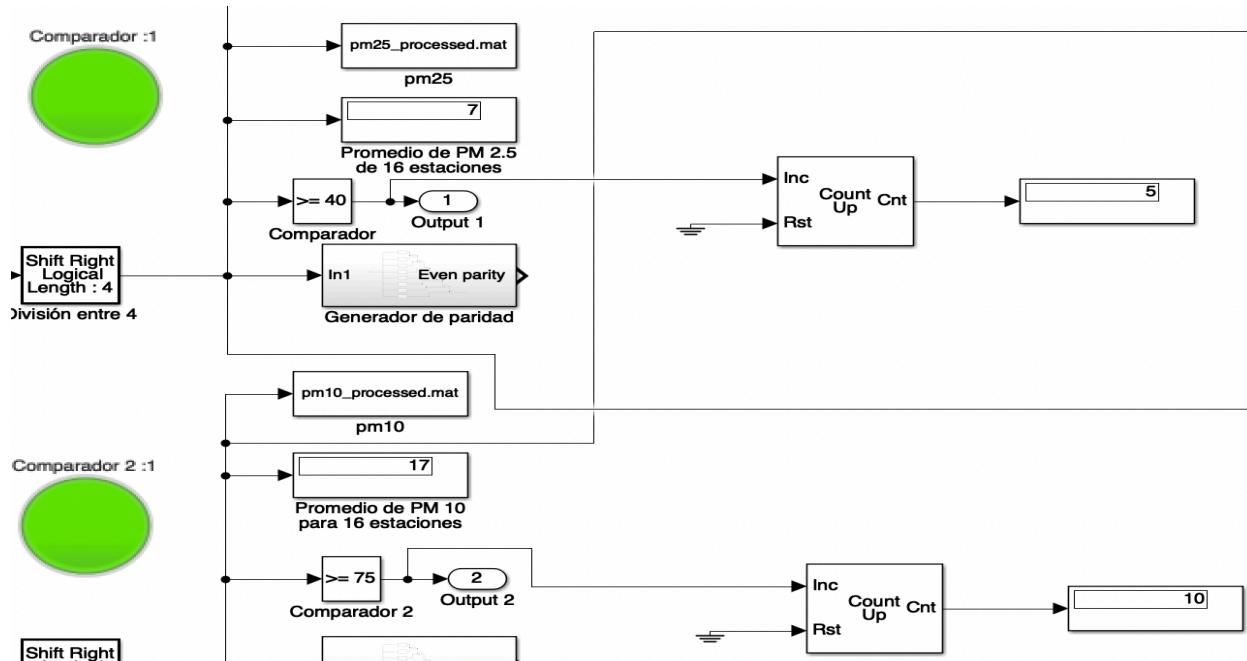
El tercer promedio calculado con el modelo de Matlab también es congruente con lo presentado en la tabla, como se muestra en la siguiente captura. Esto significa que el modelo de Matlab es correcto y está bien implementado.



De igual manera se realizó una prueba con la humedad, para determinar si el promedio obtenido por Simulink en el modelo es preciso.

4.1.2 Contar ocurrencias de PM2.5 y PM10.

Para lograr este objetivo, en la parte de PM2.5 y PM10 pusimos un contador en ambos casos. Lo modificamos para que solo tuviera una salida la cual era la cuenta. Cada una de ellas tenía 2 entradas. Una de ellas, la de increase va conectada al comparador que determina si el nivel es elevado, que es un booleano. La otra, el reset, va conectado a tierra ya que no hay ninguna señal que deba reiniciar el valor del contador. La salida, que representa la cuenta, se conectó a un display para mostrar el número de ocurrencias.



4.1.3 Máquina de estados de riesgos.

Como parte de la funcionalidad del modelo, se espera obtener el nivel de contingencia dependiendo de las lecturas de PM10 y PM25. Hay varias fases de contingencia, partiendo de "Prevención" (estado basal), hasta "Fase III", que indica que los niveles son muy elevados. La tabla que relaciona los niveles de los contaminantes con los de contingencia fue proporcionada en los requerimientos y se muestra a continuación:

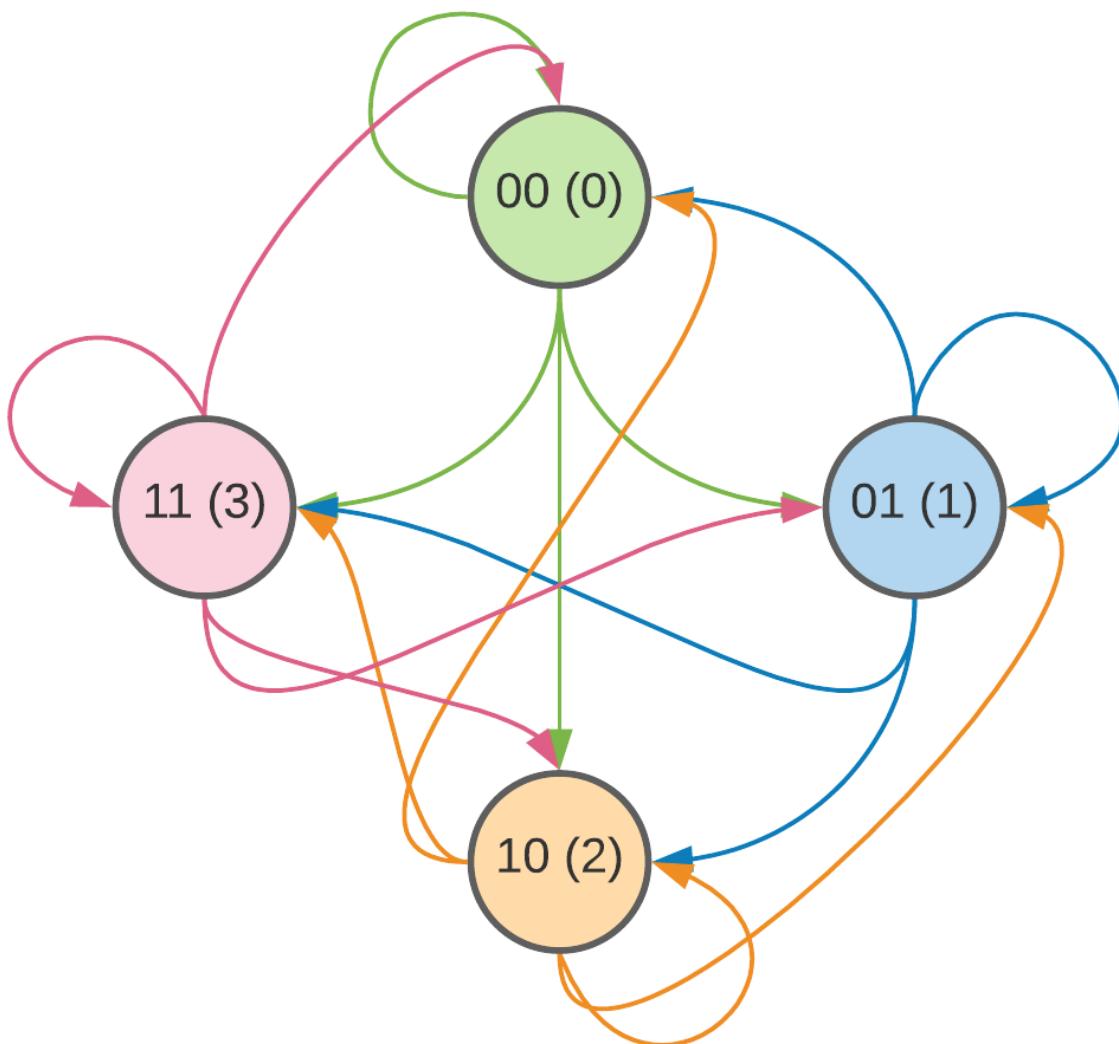
FASE	PM2.5	PM10
Preventiva	<25	<15
Fase I	<35	<25
Fase II	<45	<35
Fase III	<55	<45

Para implementar la funcionalidad, es necesario plantear una máquina de estados que indique cómo se determina el nivel de contingencia. En primer lugar, se definió que la equivalencia entre estados binarios y nivel de contingencia sería la siguiente:

Binario	Fase contingencia
00	Preventiva
01	Fase I
10	Fase II
11	Fase III

Además, se decidió que desde cualquier estado sería posible llegar a otro; es decir, aunque el nivel de contaminantes se encuentra en Fase III, si a la siguiente medición resulta que han disminuido lo suficiente, es posible pasar directamente a Fase I o Preventiva y saltarse la Fase II.

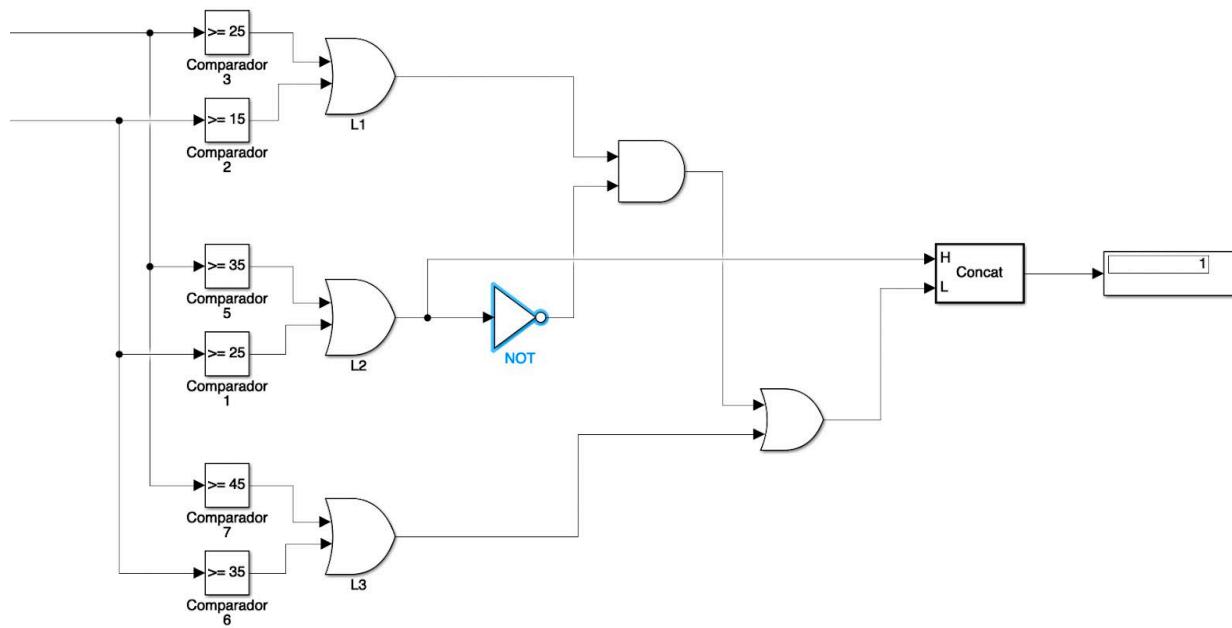
Lo mismo sucedería en el caso inverso, cuando el nivel de los contaminantes aumenta drásticamente, es posible saltarse fases para activar la tercera, entonces, el diagrama de estados se planteó de la siguiente manera:



En cuanto a la implementación del circuito, el primer paso fue agregar comparadores para determinar en qué nivel se encuentra cada variable. Por ejemplo, si el nivel de PM2.5 es mayor a 25, lo que indica que se encuentra dentro del rango de la Fase I.

Los comparadores de cada variable que pertenecen al mismo nivel de contingencia se usaron como entradas para tres compuertas OR, una para cada nivel. Esto significa que solo es necesario que una de las variables se encuentre por encima del límite para que se genere una señal verdadera.

Cada una de las señales se nombró como L1, L2, L3, donde L1 indica si los niveles sobrepasaron la fase preventiva, y así posteriormente. Esta parte del circuito se implementó de la siguiente manera (la señal inicial superior es PM2.5):



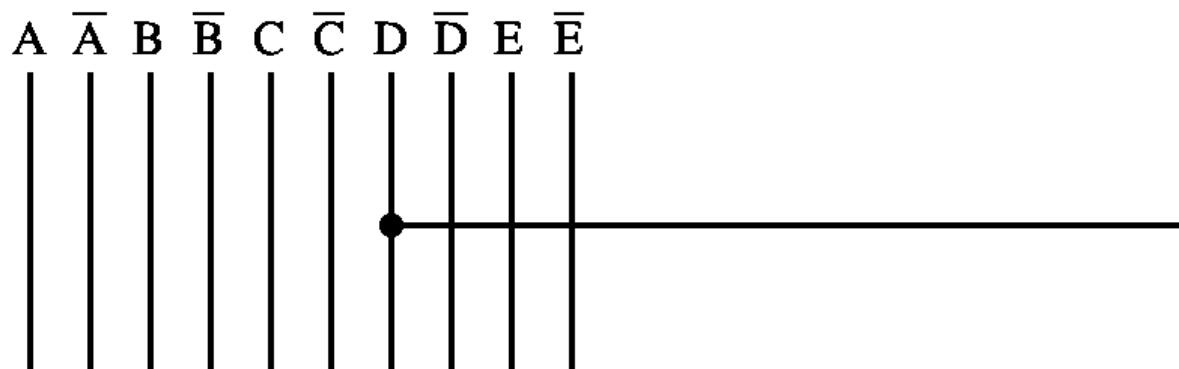
Posteriormente, es necesario implementar una lógica para realizar los estados. Se decidió utilizar flip-flops de tipo D, puesto que el diseño del circuito es más accesible. Puesto que los estados pueden ser representados con únicamente 2 bits, entonces solo se necesitan 2 flip-flops en el circuito (A y B). De acuerdo con nuestro diagrama de estados inicial, se construyó una tabla de verdad que indica el output esperado en el siguiente estado de acuerdo con las condiciones presentes. Se utilizaron condiciones no importa para combinaciones que no pueden existir en el sistema, como cuando L2 está en alto, pero L1 en bajo (porque si los valores son tan altos como para activar L2, entonces L1 también se activó).

Estado presente		Lineas de comparadores			Estado siguiente	
A	B	L_1	L_2	L_3	D_A	D_B
0	0	0	0	0	0	0
0	0	0	0	1	0	1
0	0	0	1	0	X	X
0	0	0	1	1	1	0
0	0	1	0	0	X	X
0	0	1	0	1	X	X
0	0	1	1	0	X	X

0	0	1	1	0	X	X
0	0	1	1	1	1	1
0	1	0	0	0	0	0
0	1	0	0	1	0	1
0	1	0	1	0	X	X
0	1	0	1	1	1	0
0	1	1	0	0	X	X
0	1	1	0	1	X	X
0	1	1	1	0	X	X
0	1	1	1	1	1	1
1	0	0	0	0	0	0
1	0	0	0	1	0	1
1	0	0	1	0	X	X
1	0	0	1	1	1	0
1	0	1	0	0	X	X
1	0	1	0	1	X	X
1	0	1	1	0	X	X
1	0	1	1	1	1	1
1	1	0	0	0	0	0
1	1	0	0	1	0	1
1	1	0	1	0	X	X
1	1	0	1	1	1	0
1	1	1	0	0	X	X
1	1	1	0	1	X	X
1	1	1	1	0	X	X
1	1	1	1	1	1	1

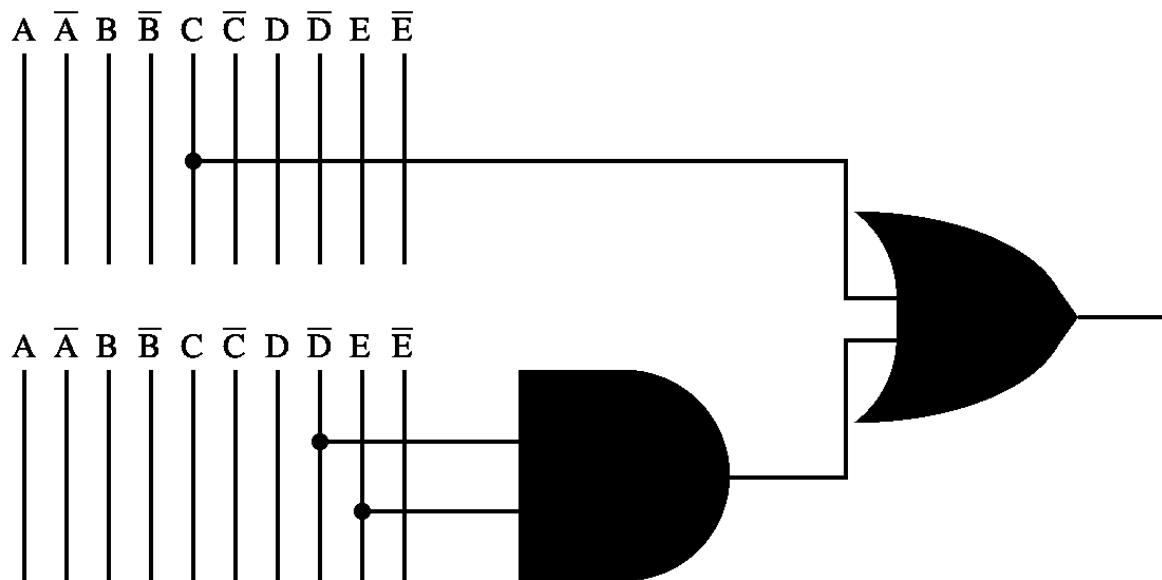
Para obtener expresiones lógicas a partir de la tabla de verdad, se pueden utilizar mapas de Karnaugh. Sin embargo, como hay cinco variables, el procedimiento es complejo y es muy común cometer errores. Para facilitar esta labor, se utilizó una herramienta web que simplifica a partir de una tabla de verdad (32X8, 2020); para la salida A generó la siguiente expresión:

$$y = D$$



Mientras que a la salida B:

$$y = C + D'E$$



Esto indica que las salidas de los flip-flops A y B no tienen ningún impacto en el estado siguiente del sistema, y que solo se define de acuerdo con las señales que provienen de los OR's de los comparadores. El motivo es que, al decidir qué se puede pasar desde cualquier fase a cualquier otra, el estado presente se vuelve innecesario, puesto que no tiene ningún impacto en el siguiente estado. Por este motivo, se decidió eliminar los flip-flops del diseño y construir los dos circuitos únicamente con las señales L1, L2 y L3. Para mostrar el nivel en un display, se concatenaron los dos bits y se conectaron a una display. Un cero indica fase preventiva, mientras que un 3 es Fase III de contingencia.

4.1.4 Evidencia de correcta conexión con la nube.

Los datos de Simulink se enviaron a la base de datos por medio de un script externo de Matlab. En primer lugar, se ejecuta el modelo de Simulink que procesa todas las lecturas de 16 estaciones para una misma estampa de tiempo y obtiene el promedio. Esto se realiza para seis variables atmosféricas. Una vez que se tiene el promedio, se guarda en un documento tipo .mat, con un formato de serie de tiempo. Un script externo de Matlab carga la información de los archivos, la extrae por estampa de tiempo, y genera una query. Posteriormente, esa query se envía tanto a Tec Host como a Helio Host por medio de un script de PHP que la recibe como parámetro de URL y realiza una conexión con la base de datos, seguida de una inserción. A continuación, se muestra una captura de pantalla del Script de Matlab y de la base de datos con la información:

```

 36 mt_hh.m" + 
 37 % Convertir a un vector dateime:
 38 dateime = (date1) + basetime; ["ConvertFrom", "datenum"];
 39 %
 40 % Para recuperar los valores de cada campo del date time
 41 y = string(year(date));
 42 m = string(month(date));
 43 d = string(day(date));
 44 h = string(hour(date));
 45 mm = string(minute(date));
 46 s = string(second(date));
 47 %
 48 % Para formatear la fecha como una string "YYYY-MM-DD HH:mm:ss"
 49 formattedDate = strcat(y, "-", m, "-", d, " ", h, ":", mm, ":", s);
 50 %
 51 % Para obtener el valor del promedio para esa time stamp específica
 52 % para cada variable
 53 pm10_val = int32(pm10_vector(n));
 54 pm25_val = int32(pm25_vector(n));
 55 humidity_val = int32(humidity_vector(n));
 56 no2_val = int32(no2_vector(n));
 57 so2_val = int32(so2_vector(n));
 58 o3_val = int32(o3_vector(n));
 59 %
 60 % Generar el query para MySQL con los valores de Matlab en caso de
 61 % inserción
 62 query_insert = sprintf('INSERT INTO `Medida` (`tiempo`, `pm10`, `pm25`, `rh`, `no2`, `so2`, `o3`) VALUES (''%s'', ''%f'', ''%f'', ''%f'', ''%f'', ''%f'', ''%f'')',...
 63 formattedDate, pm10_val, pm25_val, humidity_val, no2_val, so2_val, o3_val);
 64 %
 65 % Generar el query en caso de update
 66 query_update = sprintf('`ON DUPLICATE KEY UPDATE `pm10` = ''%f'', `pm25` = ''%f'', `rh` = ''%f'', `no2` = ''%f'', `so2` = ''%f'', `o3` = ''%f''',...
 67 pm10_val, pm25_val, humidity_val, no2_val, so2_val, o3_val);
 68 %
 69 % Generar el query completo
 70 query = strcat(query_insert, query_update);
 71 %
 72 % Codificar el query como parámetro del URL:
 73 sql_query_encoded = urlencode(query);
 74 %
 75 % Preparar el URL final con el query para PHP en HelioHost:
 76 url_HH = sprintf('http://diegoprod9.heliohost.org/insertMatlab.php?query=%s', sql_query_encoded);
 77 url_TH = sprintf('http://10.48.149.219/Gpo31-E5/insertMatlab.php?query=%s', sql_query_encoded);
 78 %
 79 % Finalmente, enviar el INSERT:
 80 messageBack_HH = webread(url_HH, options);
 81 messageBack_TH = webread(url_TH, options);
 82 %
 83 disp("Helio Host");
 84 disp(messageBack_HH);
 85 %
 86 disp("Tec Host");
 87

```

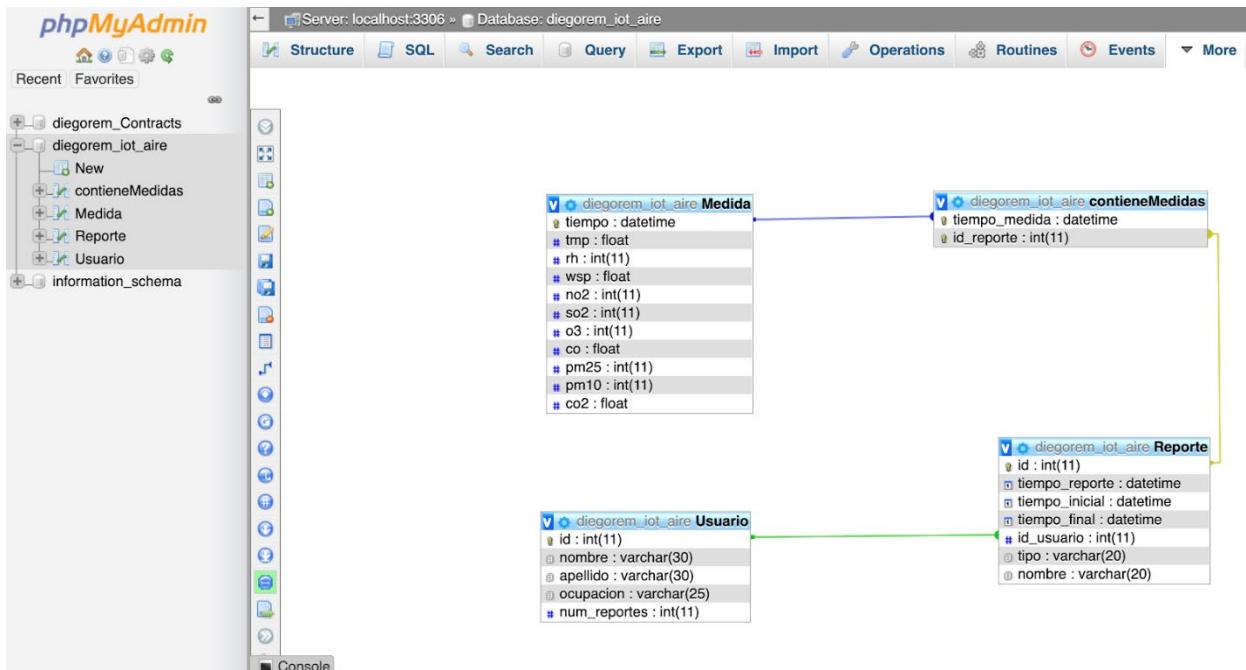

Los datos de esta relación provienen de dos sistemas distintos. El primero de ellos es Matlab; mediante un modelo de Simulink se lee la información proporcionada por el SIMAT (humedad relativa, NO₂, SO₂, O₃, PM10, PM2.5), se procesa, y se guarda en un archivo externo, posteriormente, un script de Matlab extrae los datos de cada variable atmosférica y los acomoda en una query, que incluye como llave primaria la estampa de tiempo. Por otro lado, en Packet Tracer se implementaron sensores de temperatura, velocidad de viento, CO y CO₂, y se creó la query con ayuda de un script de Python; ambas queries se enviaron a un script PHP mediante una HTTP request, y ese script las insertó en la base de datos. Es importante mencionar que en las queries se utilizó el statement "INSERT INTO ... ON DUPLICATE KEY UPDATE ...", para evitar errores en la inserción en casos en que ya existieran datos para una estampa de tiempo específica, como cuando ya se insertaron los datos de Simulink y posteriormente se insertan los de Packet Tracer o viceversa.

Es necesario mencionar que las otras relaciones del diseño conceptual de la base de datos no se utilizaron, debido a que no se cuenta con el conocimiento necesario de desarrollo web. La relación Usuario tiene el objetivo de guardar la información de un usuario del sistema; sin embargo, no se implementó un mecanismo de log in o de sign up para el registro, por lo que no se insertaron datos.

Por otro lado, la relación Reporte se utilizaría para almacenar ciertos detalles de reportes personalizados creados por los usuarios del sistema. Como no se tienen usuarios, tampoco se utilizó la relación "Reporte", y por lo tanto tampoco la relación "contieneMedida", que asociaba cada reporte con una o muchas fechas, es por estos motivos que únicamente se muestran datos en la relación "Medida".

4.2.2 DB completa y normalizada hasta 3NF.

A continuación, se muestra una captura de pantalla del modelo relacional desde PHP.



Para comprobar que la base de datos está en tercera forma normal, se analizará NF1, NF2, y finalmente NF3.

∅ **NF1: Solo atributos atómicos en las relaciones:** La relación medida cumple con esta característica, puesto que el atributo tiempo es una única estampa de tiempo, y el resto son enteros o floats, en la relación reporte también se incluyen solo atributos atómicos asociados a un reporte, como su ID, tiempo de creación, fecha final e inicial de datos, tipo de reporte, nombre de reporte, y el usuario que lo creó; en cuanto a "Usuario", también son atributos atómicos; en el caso del nombre, que es un atributo compuesto, se separó en nombre y apellido, por lo que resultó atómico y finalmente, "contieneMedidas" únicamente tiene dos atributos y ambos son llave primaria y atómicos, por lo tanto, la base de datos está en primera forma normal.

∅ **NF2: Dependencia funcional con todos los atributos no prime:**

→ Para la relación "Medida":

- Superllaves: No hay, puesto que ningún conjunto de valores de variables atmosféricas es único para una estampa de tiempo.
- DF con ANP:

- tiempo -> tmp: Verdadero, si se encuentra el mismo datetime, el valor de tmp debe ser el mismo.
- tiempo -> rh: Verdadero, si se encuentra el mismo datetime, el valor de rh debe ser el mismo.
- tiempo -> wsp: Verdadero, si se encuentra el mismo datetime, el valor de wsp debe ser el mismo.
- tiempo -> no2: Verdadero, si se encuentra el mismo datetime, el valor de no2 debe ser el mismo.
- tiempo -> so2: Verdadero, si se encuentra el mismo datetime, el valor de so2 debe ser el mismo.
- tiempo -> o3: Verdadero, si se encuentra el mismo datetime, el valor de o3 debe ser el mismo.
- tiempo -> co: Verdadero, si se encuentra el mismo datetime, el valor de co debe ser el mismo.
- tiempo -> pm10: Verdadero, si se encuentra el mismo datetime, el valor de pm10 debe ser el mismo.
- tiempo -> pm25: Verdadero, si se encuentra el mismo datetime, el valor de pm25 debe ser el mismo.
- tiempo -> co2: Verdadero, si se encuentra el mismo datetime, el valor de co2 debe ser el mismo.

Por lo tanto, la relación "Medida" está en segunda forma normal.

→ Para la relación "Reporte":

- Super llaves: {tiempo_reporte, id_usuario, tipo, nombre, tiempo_inicial, tiempo_final} Es super llave porque la combinación es única y sirve para identificar distintos reportes.
- ANP: No hay.

→ Para la relación "Usuario":

- Super llaves: No hay.
- ANP:

- id -> nombre: Verdadero, si se encuentra el mismo id, es la misma persona y el mismo nombre.
- id -> apellido: Verdadero
- id -> ocupacion: Verdadero
- id -> num_reportes: Verdadero

→ Para la relación “contieneMedida”:

- Ambos atributos son llaves primarias.

Por lo tanto, la base de datos está en segunda forma normal.

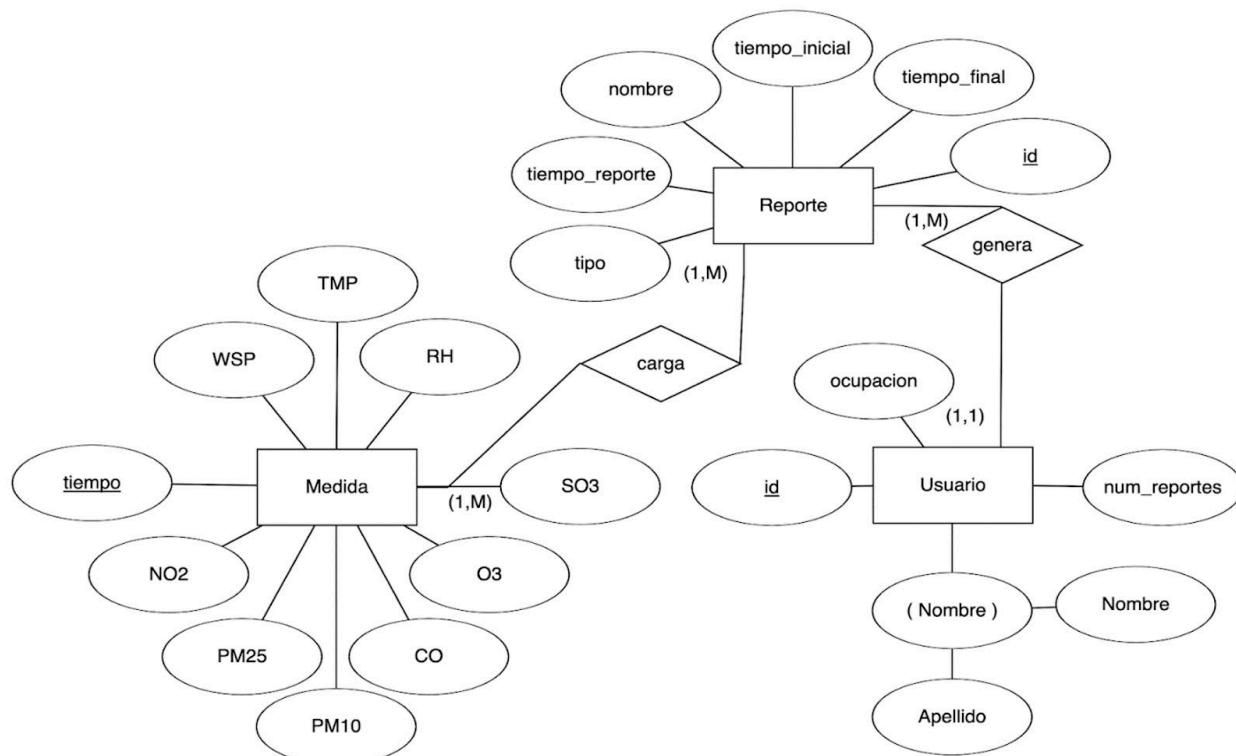
- ∅ **NF3: Sin Dependencia Funcional Transitiva:** No se encuentran dependencias funcionales transitivas en ninguna relación. Desde el diagrama de Entidad Relación se separaron las entidades de tal manera que se evitó la repetición de información. Cada relación tiene los atributos necesarios, sin almacenar los de otras relaciones a excepción de llaves foráneas. Esto significa que se encuentra en tercera forma normal.

4.2.3 Diagrama ER.

En este diagrama se puede apreciar a detalle el diseño conceptual propuesto para la base de datos, con las entidades “Reporte”, “Medida” y “Usuario”.

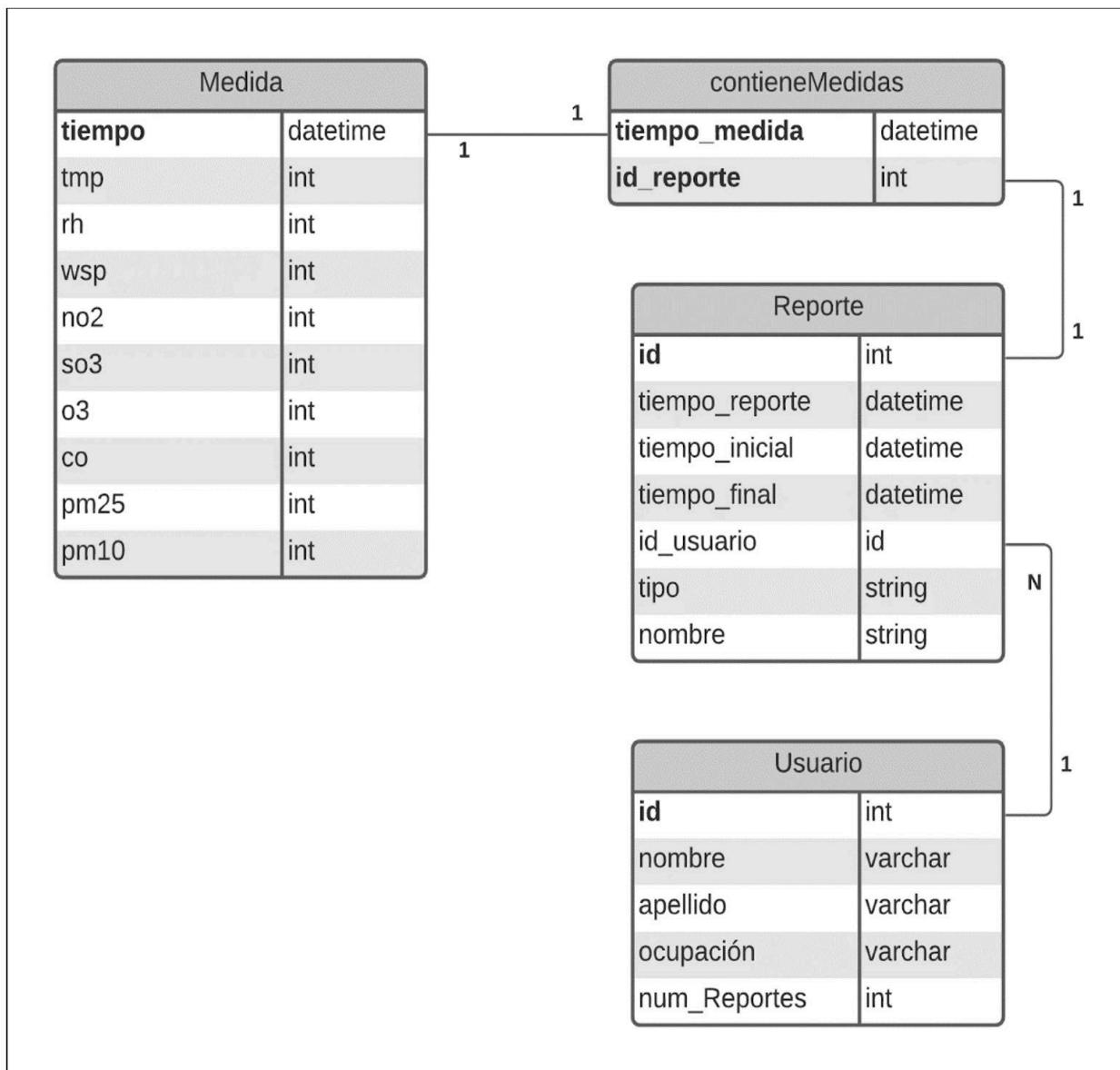
“Medida” almacena los promedios de cada una de las cantidades para un tiempo específico. “Usuario” almacena información importante de las personas que usarán el sistema de información.

“Reporte” funciona como un historial, que guarda las características de los reportes que los usuarios han generado, de manera que se puede recuperar fácilmente.



4.2.4 Modelo relacional.

El siguiente diagrama es el modelo relacional propuesto para la base de datos. En este caso, las entidades del modelo ER se convirtieron en relaciones (Medida, Reporte y Usuario). Para modelar las relaciones de ER, en el caso del Reporte se almacena la llave primaria del Usuario como una llave foránea, dado que es una relación de uno a muchos; para la relación (en ER) entre Reporte y Medida, se creó una relación nueva llamada contieneMedidas, en la que se almacena la timeStamp de cada Medida incluida en el reporte, junto con el id del Reporte.



3. Recursos de un sistema computacional.

3.1 Analizar la manera en que el uso de diferentes arquitecturas computacionales que ensamblan un sistema de IoT:

Normalmente, cuando se piensa en una computadora inmediatamente se visualiza una laptop, o una computadora de escritorio. Este tipo de dispositivos se caracterizan por tener una gran cantidad de funcionalidades, dependiendo de las aplicaciones y software que el usuario utilice (GeeksForGeeks, 2020). Sin embargo, también existen las computadoras embebidas, que están integradas a otros sistemas con el propósito de monitorear y controlar ciertos procesos. Tienen una funcionalidad mucho más limitada, pues solo pueden llevar a cabo una tarea específica, para la cual son diseñadas (Catsoulis, 2005). La mayor parte de los dispositivos utilizados en la vida cotidiana cuentan con computadoras embebidas, desde un refrigerador o una lavadora, hasta cualquier vehículo que cuente con sistemas electrónicos de control (Day, 2014). Los sistemas de información IoT se conforman por una gran cantidad de dispositivos interconectados, que cumplen funciones distintas. Algunos de ellos son sistemas embebidos, mientras que otros son computadoras con una amplia gama de funcionalidades.

La capa más baja en cualquier arquitectura de IoT consiste en sensores y/o actuadores, cuya función es monitorear el medio y recuperar información de este (McKendrick, 2016). En el caso del sistema propuesto para el monitoreo de la calidad del aire, existen sensores de temperatura, humedad, velocidad de viento, así como los que miden la concentración de ciertos contaminantes. Todos ellos se encontrarán repartidos en distintos puntos estratégicos de la ciudad, de manera que permitirán conocer el estado general del aire. Estos sensores únicamente llevan a cabo una única tarea, que es registrar los valores de las variables que monitorean, y enviarlos a una computadora central por cada estación. Ni siquiera deben tener un sistema operativo, pues su aplicación es fija. Por lo tanto, son un ejemplo de computadoras embebidas, cuya funcionalidad es muy específica y limitada.

Una vez que los datos fueron recibidos por la computadora central, se tienen que ejecutar ciertos scripts, en nuestro caso escritos en Matlab, para enviarlos a la base de datos. Esta computadora debe tener otro tipo de funcionalidades, sistema operativo, y conectividad a la red. Por lo tanto, es una computadora de escritorio. La base de datos MySQL se almacena

en un servidor, al igual que el script de PHP que regula la inserción de datos. Esta computadora es un servidor, que debe contar con un sistema operativo, así como con software diverso que le permita establecer la comunicación entre varios dispositivos, almacenar archivos, ejecutar scripts de PHP, etc. Por este motivo, se considera una computadora "de escritorio", en lugar de una embebida. Finalmente, las computadoras de escritorio también están presentes del lado del cliente, que debe tener acceso a un navegador de internet para acceder al sistema de información, cuya interfaz es una página web.

3.2 Analizar la ejecución de procesos asíncronos en el sistema de IoT:

Un proceso asíncrono es aquel que depende de otro. En el caso de la transferencia de información, se puede decir que es síncrona cuando el emisor y el receptor están en un proceso conjunto. Es decir, el dispositivo receptor sabe que va a recibir datos y espera al dispositivo emisor (GeeksForGeeks, 2019). En cambio, los procesos asíncronos ocurren de manera independiente a otros procesos o dispositivos. No importa si el dispositivo receptor sabe que va a recibir un mensaje, sino que se envía y se espera que esté listo para procesarlo.

Específicamente en IoT existen dos patrones de comunicación muy utilizados. El primero de ellos es el de Petición-Respuesta (Request- Response), en el que el cliente pide datos al emisor (XMPP, 2020). Esta situación se daría si existiera una unidad de procesamiento centralizada que pidiera datos a los sensores de manera periódica, lo cual no está incluido en la propuesta. También está el patrón de mensajería asíncrona (asynchronous messaging), en el que el emisor envía datos al receptor cuando él decida, sin importar si el receptor los está esperando (XMPP, 2020). En nuestro sistema IoT, la información de los sensores será enviada a la base de datos en intervalos determinados por nosotros; sin embargo, el sistema receptor no tiene forma de saber cuándo va a suceder. El script de PHP insertMatlab cumple la función de realizar las inserciones en la base de datos cuando se ejecute desde una URL, sin importar el momento. Es por esto por lo que la comunicación entre los sensores y el servidor en este sistema de IoT es asíncrona.

En cuanto al usuario del sistema de información, podría considerarse de tipo Request Response, puesto que el servidor envía la información desde la base de datos siempre que el usuario así lo solicite en la página web, sin embargo, el servidor no tiene definidos momentos específicos para enviar la información, pues esto depende de los clientes; desde este punto de vista, se puede considerar una relación asincrónica.

4.3.1 Aplicación Productor - Consumidor (Base de Datos y Sensores Simulados):

4.3.1.1 Funcionamiento de la aplicación de acuerdo con los requerimientos.

Se programó una aplicación en Python que permite simular un proceso de producción y consumo concurrente de datos. En primer lugar, para la producción de los datos se diseñó una superclase “Sensor” que representa un sensor genérico, y tres subclases para representar sensores de temperatura, presión atmosférica y humedad. Cada uno de estos sensores genera datos aleatorios dentro de un rango de valores apropiado dada la variable atmosférica que miden. Posteriormente, para el consumo y almacenamiento de datos se creó una clase “FakeDatabase”, que representa un base de datos. Mediante un método recibe un mensaje que consiste en una lista de seis elementos, de acuerdo con las especificaciones. Con los elementos del mensaje coloca el valor enviado en una lista correspondiente a la variable atmosférica correspondiente. Además, incluye funciones para obtener la media, moda y mediana una vez que se terminaron de almacenar los datos. Adicionalmente, se incluyó una clase “PipelineQueue”, que permite almacenar temporalmente los mensajes producidos por los sensores, para que la base de datos pueda consumirlos uno a uno independientemente del ritmo de producción. Se crearon seis sensores, dos de cada clase, así como una FakeDatabase. Además, se diseñó un proceso para producir datos y uno para consumirlos. Dentro de un Thread Pool Executor se crearon siete hilos que se ejecutan de manera concurrente, seis que producen datos con el uso de un sensor y uno que los consume mediante una base de datos. El correcto consumo de los datos fue posible gracias a la PipelineQueue. Una vez que los datos terminaron de producirse y almacenarse, se calcula la media, moda y mediana por variable, y se imprimen en la consola. A continuación, se muestra una captura de pantalla:

```
2020-11-24 11:32:28.180 INFO: Consumer storing message with id: 1
2020-11-24 11:32:28.180 INFO: Consumer received EXIT
Total number of updates: 240

PRESSURE
Number of readings: 80
Received values [501, 502, 503, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504, 504]
Length: 80
Mean: 539.6875
Mode: 504
Median: 531.5

HUMIDITY
Number of readings: 80
Received values [1, 2, 3, 3, 4, 5, 10, 12, 13, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80]
Length: 80
Mean: 51.85
Mode: 37
Median: 54.5

TEMPERATURE
Number of readings: 80
Received values [-15, -15, -14, -14, -13, -13, -11, -10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80]
Length: 80
Mean: 12.4625
Mode: 9
Median: 12.0
```

4.3.1.2 Respuesta en equipo : ¿Qué temas del Módulo 3 se utilizan en su aplicación y de qué manera?

Los temas del Módulo 3, Administración de recursos de un sistema computacional, fueron de gran importancia durante el desarrollo de nuestra aplicación. En cuanto a la arquitectura de un sistema computacional, se emplea de manera indirecta puesto que implementamos sensores y bases de datos simulados. Si bien no se interactuó con componentes físicos, es necesario reconocer que los sensores serían computadoras embebidas, mientras que la base de datos funciona sobre una computadora de escritorio.

Un tema que tiene aún más importancia en nuestra aplicación es "Administración de procesos". Durante el módulo aprendimos que un proceso se compone de una serie de hilos que pueden ejecutarse de manera independiente entre sí. Además, conocimos el concepto de concurrencia, que consiste en ejecutar un conjunto de procesos de forma

aparentemente simultánea. Sin embargo, lo que ocurre en realidad es que el sistema operativo guarda información de cada proceso en un Process Control Block (PCB), lo que le permite cambiar de contexto y regresar al proceso anterior sin complicaciones. Entonces, cuando varios procesos se ejecutan de manera concurrente en realidad cada uno de ellos se ejecuta por un lapso de tiempo muy pequeño, y después hay un cambio de contexto a otro proceso distinto, lo que genera la ilusión de que la ejecución es simultánea. Esto fue la base de nuestra aplicación, puesto que la producción de datos y su consumo es concurrente. Mediante el uso de un “Thread Pool Executor” se agregan seis procesos de producción, cada uno asociado a un sensor, y uno de consumo, asociado a una base de datos. A pesar de que pareciera que todos se ejecutan al mismo tiempo, los logs en la consola demuestran que es de manera secuencial.

Finalmente, en el tema de “Programación de entrada y salida de datos” aprendimos que existen mecanismos para que ciertos procesos produzcan datos y que después sean utilizados por un consumidor. Esto es de particular importancia cuando existen más de un generador de datos y solo uno que los consume, como en la aplicación, puesto que existen seis sensores y solo una base de datos. Para asegurar que todos los mensajes serán procesados y que el consumidor pueda atenderlos uno a uno, se utiliza una Pipeline. Esta estructura consiste en una file (Queue) que recibe los mensajes de distintos productores (sensores) y los almacena temporalmente mientras que el consumidor (base de datos) termina de procesar el dato anterior y está listo para el siguiente. De esta manera, los tres grandes temas aprendidos en el módulo se ponen en práctica en nuestra aplicación.

4.3.1.3 De manera individual, responder :Menciona un elemento técnico relevante que justifica la importancia de aprender cada tema del módulo (lista en Sesión1), en relación con su aplicación en el Reto.

- ∅ **Diego Jiménez:** Es importante aprender de arquitectura computacional porque en un sistema de IoT se utiliza una gran cantidad de dispositivos diferentes entre sí. Por un lado, están los sensores que son computadoras embebidas y tienen función específica, leer niveles de una variable atmosférica. Por otro lado, se

encuentran computadoras de escritorio que se utilizan para almacenar la base de datos, scripts, y para acceder a la página web.

En cuanto a Administración de procesos, si bien nosotros nos encargamos completamente de esto en el reto, la base de datos debe tener forma de realizar procesos concurrentes. Es decir, un usuario puede pedir un reporte (lo que ejecuta queries) al mismo tiempo que los datos de algún sensor se guardan (que también necesita una query). El DBMS incluye mecanismos de programación concurrente que permiten que la base de datos pueda realizar ambas labores de manera aparentemente "simultánea".

Finalmente, es importante conocer métodos para entrada y salida de datos puesto que en el reto tenemos varios sensores que producen información y solo una base de datos que los almacena. Mediante scripts de Matlab y Python se consiguió un mecanismo para poder enviar queries a la base de datos que ya incluyen las lecturas de un grupo de sensores para una estampa de tiempo específica. En lugar de ejecutar muchas queries, una para cada variable, se agrupan los valores y se envían solo dos a la base de datos. Posteriormente se reciben las queries y se envían a el único consumidor, la base de datos. Por los motivos anteriormente expuestos es importante aprender cada tema del Módulo.

- ∅ **Bruno Santos:** El bloque asignado a nosotros de IoT, fue un módulo de muchos aprendizajes. Donde aprendimos no solo la parte teórica de la materia, si no también nos enseñó cómo trabajar en un grupo de trabajo. Donde cada uno tiene sus roles asignados, además que haya un líder, todos trabajan juntos.

En el Módulo 3, administración de recursos de un sistema computacional, tuvimos la oportunidad de ver 3 grandes tópicos que nos ayudaron en la resolución de nuestro reto: Arquitectura de un sistema computacional, Concepto de proceso y Entrada y salida de datos.

La arquitectura de un sistema computacional, lo usamos en la parte de saber cómo podíamos conectar los diferentes dispositivos que utilizamos en nuestra

solución. Cómo conectar los sensores y las computadoras de manera inalámbrica.

Durante nuestras clases, pudimos ver que, en las bases de datos, y en los sensores, están corriendo una gran cantidad de procesos. En el punto de Concepto de Proceso pudimos aprender acerca de los mismos, donde aprendimos acerca de la prevención de errores que se pueden generar al correr varios procesos al mismo tiempo. Y con el software de DBSM (Database Management System), en nuestro caso pudimos utilizarlo como "forma de prevención" de esos errores, por permitirnos de correr procesos de manera simultánea.

Ya por último viene el punto de entrada y salida de datos que en mi opinión es uno de los más importantes, por el hecho de que estamos trabajando con diversas plataformas, como Packet Tracer, Matlab, una base datos, y entre otros. En este tópico pudimos ver la programación que se debe de hacer para enviar los datos, y la manera que los vamos a recibir del otro lado. La manera correcta de almacenar esos datos en nuestra base de datos y de igual forma manejarlo para finalmente implementar en nuestra página web.

- ∅ **Diego Rodríguez:** Es de suma importancia para este reto saber utilizar y tener en cuenta el funcionamiento de la administración de procesos ya que con este en los modelos creados en Simulink, podemos saber el procesamiento de muchos de los datos durante este y el cómo se procesan de manera simultánea todos estos procesos para completar este sistema y así correr de manera ordenada y completamente funcional. En cuanto a input y output es importante en mi forma de verlo para mientras recibimos datos, poder seguir realizando búsquedas y finalmente y no menos importante, la arquitectura computacional y utilizando la arquitectura de Von Neumann, estamos hablando de la base todo nuestro proyecto, es la forma con la que podemos comprender el almacenado de los datos y las instrucciones que recibiremos mediante sensores en este reto.
- ∅ **Iñigo Zepeda:** Como conclusión, es importante percatarnos acerca de los temas que pusimos en práctica, los cuales fueron aprendidos durante estas semanas de

curso. Los conocimientos adquiridos en los temas de Administración de procesos, Administración de recursos de un sistema computacional, Programación de entrada y salida de datos, se reflejan en la arquitectura y el diseño que se realizó en este sistema en Python.

El tema de administración de procesos fue muy importante aprenderlo puesto que aprendimos de qué manera podemos realizar procesos concurrentes, y que parezca que se están realizando todos simultáneamente.

Sobre la entrada y salida de datos, nos sirve mucho porque este mismo concepto lo estamos viendo en nuestro proyecto final. Al utilizar Packet Tracer, Matlab y mandar datos a una base de datos en PHPMyAdmin.

La arquitectura computacional es un tema muy importante de aprender puesto que tenemos conexiones entre dispositivos, y se tiene que eficientizar la manera en la que están interconectados.

De esta manera concluimos, que tuvimos un curso exitoso puesto que podemos aplicar a situaciones reales estos aprendizajes de diferentes enfoques computacionales.

- ∅ **Brian Serranía:** Los temas que vimos en el módulo 3 fueron de suma importancia, en pocas palabras nos ayudó a entender desde la base como es que los programas funcionan, como se llevan a cabo los procesos o las acciones síncronas y asíncronas, entre otras cosas; dentro de los puntos más importantes fue ver cómo funcionan los sensores y como se irán guardando los datos en las bases de datos pero con un requisito indispensable y es el hecho que se puedan realizar varios procesos “simultáneamente”, esto quiere decir que si yo pedía algo a nuestro programa de igual manera seguía recibiendo y guardando datos.

La importancia de este tema en particular reside en la variedad de cosas que le puedo pedir a mi programa y que sea capaz de realizarlas, es decir, sin problemas de pérdida de información o de no poder acceder a los datos almacenados porque sigue recibiendo información, lo hace un programa con muchos beneficios y funcionamientos.

Situaciones como las mencionadas anteriormente son aquellas que puedes aplicar cuando comprendes el funcionamiento desde las bases de los programas en particular, en lo personal me voy muy satisfecho con lo aprendido y con las entregas de los retos que como equipo pudimos desarrollar, a pesar de ser un bloque pesado en cuanto a la cantidad de información que recibimos y debimos aplicar, se pudo implementar de manera correcta conforme los requerimientos de cada una de las entregas.

4. Diseño interactivo:

4.4.1 Actualización final de la página web del equipo en línea.

- ∅ **Inicio:** <http://diegorem99.heliohost.org/iot/inicio.php>
- ∅ **Análisis:** <http://diegorem99.heliohost.org/iot/display-data.php>
- ∅ **Vista final del dashboard con vista de datos desde la DB:** Se puede observar la página con un filtro de datos, en este caso se obtuvieron los valores registrados de las estaciones, del 1 de enero de 2020 al 3 de enero de 2020. El usuario puede filtrar las fechas deseadas para lograr hacer un análisis a fondo.

Se hizo uso del lenguaje de programación PHP y MySQL. A través de PHP pudimos capturar los datos registrados en la base de datos, ubicada en PHPMyAdmin.

La página fue diseñada en HTML, CSS, utilizando librerías como Bootstrap y Jquery para lograr un diseño responsive, es decir, adaptable a cualquier dispositivo.


Computic

[Inicio](#) [Nosotros](#) [Reportes y análisis](#) [Contacto](#) [Reglas Heurísticas](#)

CDMX

Ciudad: CDMX



Datos generales:

Población: 9 millones
Densidad: 64 habitantes por km²
Temperatura Promedio: 25°C
Humedad: 32%

Estaciones registradas:

Tepito
Ajusco
La Magdalena Contreras
Barranca Seca
El Santo

Variables registradas:

tmp
rh
wsp
no2
so2
o3
co
pm25

Informe:

		Parámetros de medición: ● Buena ● Moderada ● Mala ● Muy mala ● Pésima										
		Fecha	tmp	rh	wsp	no2	so2	o3	co	pm25	pm10	co2
Fecha de inicio	Fecha final	dd / mm / aaaa	dd / mm / aaaa	dd / mm / aaaa	dd / mm / aaaa	dd / mm / aaaa	dd / mm / aaaa	dd / mm / aaaa	dd / mm / aaaa	dd / mm / aaaa	dd / mm / aaaa	
2020-01-01	2020-01-03	00:59:00	9.7	62	0.55	27	3	6	2.45	81	65	4.89
2020-01-01	2020-01-01	01:59:00	10.8	66	0.49	27	3	5	2.43	40	81	4.89
2020-01-01	2020-01-01	02:59:00	11.76	68	0.47	27	3	5	2.43	54	103	4.89
2020-01-01	2020-01-03	03:59:00	12.52	68	0.47	25	5	5	2.43	58	100	4.89
2020-01-01	2020-01-01	04:59:00	13.07	68	0.5	24	2	5	2.43	56	103	4.89
2020-01-01	2020-01-01	05:59:00	13.39	70	0.54	22	2	5	2.43	58	86	4.89
2020-01-01	2020-01-01	06:59:00	13.48	71	0.6	21	1	5	2.43	57	70	4.89
2020-01-01	2020-01-01	07:59:00	13.35	68	0.68	20	1	5	2.43	54	60	4.89
2020-01-01	2020-01-01	08:59:00	13	60	0.76	23	5	10	2.43	46	67	4.89
2020-01-01	2020-01-01	09:59:00	12.49	49	0.9	18	5	23	2.43	38	69	4.89
2020-01-01	2020-01-01	10:59:00	11.88	40	10.5	9	2	31	2.43	26	47	4.89
2020-01-01	2020-01-01	11:59:00	11.23	35	117	4	1	32	2.43	1	29	4.89
2020-01-01	2020-01-01	12:59:00	10.57	35	1.82	5	1	29	2.43	9	24	4.89
2020-01-01	2020-01-01	13:59:00	9.6	33	1.56	3	0	27	2.43	9	36	4.89
2020-01-01	2020-01-01	14:59:00	8.98	34	1.76	2	0	28	2.43	12	46	4.89
2020-01-01	2020-01-01	15:59:00	8.41	36	1.94	3	1	30	2.43	11	43	4.89
2020-01-01	2020-01-01	16:59:00	7.87	38	2.16	5	0	28	2.43	8	39	4.89
2020-01-01	2020-01-01	17:59:00	7.37	40	2.26	4	0	28	2.43	9	39	4.89
2020-01-01	2020-01-01	18:59:00	6.91	47	2.26	7	1	20	2.43	9	31	4.89
2020-01-01	2020-01-01	19:59:00	6.49	51	2.2	8	1	18	2.43	10	23	4.89
2020-01-01	2020-01-01	20:59:00	6.11	52	2.31	8	1	17	2.43	11	20	4.89
2020-01-01	2020-01-01	21:59:00	5.76	55	199	9	1	16	2.43	10	17	4.89

Recomendaciones:

Muy mala

En este apartado se escriben todas las recomendaciones por parte de la Comisión Ambiental de la Megalópolis. Estas medidas pueden ser desde recomendar a ciertos grupos vulnerables permanecer en su casa, o declarar contingencia ambiental.

Para más información sobre las recomendaciones dar click aquí

- Ø **Conexión con la base de datos:** Con este script se logró establecer la conexión con la base de datos.

```
index.php - Documento - Notepad - Final dropueda - index-display-data.php

<?php

$servername = "65.19.141.67";
$username = "diegorem_e6Website";
$password = "Gpo31-E61234";
$dbname = "diegorem_iot_aire";

$conn = new mysqli ($servername, $username, $password, $dbname);

// if ($conn->connect_error) {
//     die("Connection failed. Error:" . $conn->connect_error);
// }
// else {
//     echo "Connection OK!";
// }

?>
```

- Ø **Obtención de datos mediante Form Tag de HTML:** Con esta etiqueta pudimos obtener el valor deseado sobre los parámetros de búsqueda. Una vez obtenidos, utilizamos PHP para buscarlos en la base de datos.

```
<form action="" method="POST">
    <input name="fechaInicial" type="date">
    <input name="fechaFinal" type="date">
    <button type="submit">Buscar</button>
</form>
<!-- PHP EXAMPLE -->
<?php

if (!empty($_POST)) {
    $fechaInicial = $_POST['fechaInicial'];
    $fechaFinal = $_POST['fechaFinal'];

    // echo $fechaInicial;
    echo "<br>";
    // echo $fechaFinal;
}
```

- Ø **Script PHP:** Este script muestra los datos obtenidos en nuestra página web, mediante la generación de una query. Si hay una conexión exitosa, se lee el código y se muestran los valores deseados, tomando en cuenta los parámetros de fecha ingresados.

```
'$fechaInicial 00:00:00' AND '$fechaFinal
23:59:59' ";

// echo $sql;

$result = $conn->query($sql);

if ($result->num_rows > 0){
    echo "<table class='mx-auto' border=\"1\""
    style="border-collapse: collapse\>";
    echo "<th>Fecha</th>";
    echo "<th>tmp</th>";
    echo "<th>rh</th>";
    echo "<th>wsp</th>";
    echo "<th>no2</th>";
    echo "<th>so2</th>";
    echo "<th>o3</th>";
    echo "<th>co</th>";
    echo "<th>pm25</th>";
    echo "<th>pm10</th>";
    echo "<th>co2</th>";
    while ($row = $result->fetch_assoc()) {
        echo "<tr>";
        echo "<td class='px-1'>".$row["tiempo"]."</
        td>";
        echo "<td class='px-1'>".$row["tmp"]."</
        td>";
        echo "<td class='px-1'>".$row["rh"]."</
        td>";
        echo "<td class='px-1'>".$row["wsp"]."</
        td>";
        echo "<td class='px-1'>".$row["no2"]."</
        td>";
        echo "<td class='px-1'>".$row["so2"]."</
        td>";
        echo "<td class='px-1'>".$row["o3"]."</
        td>";
        echo "<td class='px-1'>".$row["co"]."</
        td>";
        echo "<td class='px-1'>".$row["pm25"]."</
        td>";
```

Ln 167 Col 53 Spaces:4 UFT-8 GBK PWD

Ø **Prueba Heurística:**

- Visibilidad del estado del sistema: Nuestra página web cumple con la primera heurística en el momento en que el usuario recibe "feedback" al navegar por las diferentes pestañas y al darle clic en diferentes botones se actualiza inmediatamente.
- Relación entre el sistema y el mundo real: Cumple con la segunda heurística porque tiene textos de fácil comprensión, con un lenguaje básico y una navegación sencilla entre pestañas, y donde también se encuentran colores para la calidad del aire, que hacen su identificación sencilla.
- Control y libertad del usuario: Cumple con la tercera heurística cuando observamos que en la página el usuario puede hacer y rehacer acciones que haya tomado. Por ejemplo, si cambia a otra pestaña, fácilmente puede regresar a la anterior. O también cuando escoge una fecha, o ciudad. Si desea cambiarla tiene una lista.
- Consistencia y estándares: Cumple con la cuarta heurística. Hicimos algo que fuera fácil para el usuario comprender qué secciones son botones que se pueden presionar, y que texto los va a mandar a otra pestaña. El diseño se mantiene en toda la página sin causar confusión. Quizá falte una descripción para los contaminantes del aire que vienen abreviados, pero en todo lo demás cumple.
- Prevención de errores: Cumple con la heurística por el hecho de que en el momento no hay errores, ya que en este caso puesto que muestra los datos que se almacenan en la base de datos. De todas maneras, si se selecciona (por ejemplo) alguna fecha errónea, se puede cambiar fácilmente a la correcta.
- Reconocer en lugar de recordar: De manera que entendemos que el usuario no debe de memorizar toda la página y sus funciones, realizamos la página de manera muy intuitiva, donde todas las opciones y pestañas a

las que se le puede dar clic están fácilmente accesibles al usuario. Entonces si cumple con la respectiva heurística.

- Flexibilidad y eficiencia de uso: Por el momento no cumple. Como tal nuestra página aún no tiene diferentes settings, pero en un futuro se podrá cambiar entre versión de *Estudiante* y de *Profesional*. La versión de profesional tendrá más opciones y será aún más personalizable.
- Estética y diseño minimalista: Nuestra página cumple con la heurística de diseño minimalista, ya que es una página sencilla para el usuario, no contiene gran carga de recursos visuales, y solamente los necesarios. La tipografía y el tamaño de letra son de fácil lectura. No hay manera de que el usuario se pierda en la página, sea iniciante o profesional.
- Ayudar a los usuarios a reconocer, diagnosticar y recuperarse de los errores: Como mencionado en la quinta heurística, nuestro sistema no tiene por qué tener errores, es un sistema simple. En caso de que haya alguno con hacer un refresh de la página es suficiente. Teniendo eso en cuenta, nuestra página cumple con la heurística número 9.
- Ayuda y documentación: Teniendo en mente que el gobierno también actúa en el área que estamos trabajando, nosotros implementamos en nuestra página vínculos directos con páginas del gobierno, donde el usuario puede acceder para conocer más acerca del problema de la calidad del aire, por lo tanto, si se cumple.

5. Administración de proyecto:

4.5.1 Seguimiento puntual de las actividades registradas en Trello de la 3ta etapa y la entrega final con las fechas y responsables de cada equipo.

Checklist 3er Avance

in list [TODO \(READY\)](#)

- Description**
Add a more detailed description...
- 2 - Resumen Ejecutivo (Brian Serranía)** Hide completed items Delete
100% 
- Resumen Ejecutivo Actualizado**
- Add an item

- SUGGESTED**
 - Join
- ADD TO CARD**
 - Members
 - Labels
 - Checklist
 - Due Date
 - Attachment
 - Cover
- POWER-UPS**
 - + Add Power-Ups
- ACTIONS**
 - Move
 - Copy
 - ▣ Make Template
 - ⌚ Watch
 - ▣ Archive
 - 🔗 Share

- 3 - Introducción (Diego Urgell y Bruno Santos)** Hide completed items Delete
100% 
- 3.1 Análisis descriptivo.
- 3.2 Realizará la implementación del diseño conceptual, para lograr la integración de los componentes en un prototipo del sistema de IoT.
- 3.3 La implementación implica la construcción de los sistemas electrónicos para la conexión de al menos 8 sensores del sistema (4 en PT y 4 en Matlab).
- 3.4 Implementación final de los programas.
- Add an item

- 1 - Sistemas Digitales (Brian Serranía y Diego Rodríguez)** Hide completed items Delete
100% 
- 4.1.1 Validar el correcto funcionamiento de Humedad Relativa y Temperatura.
- 4.1.2 Contar ocurrencias de PM2.5 y PM10.
- 4.1.3 Maquina de estados de riesgos.

- 4.1.3 Maquina de estados de riesgos.

- 4.1.4 Evidencia de correcta conexión con la nube.

Add an item

2 - Bases de Datos (Diego Urgell e Iñigo Zepeda)

[Hide completed items](#) [Delete](#)

100%

- 4.2.1 DB en MySQL con datos

- 4.2.2 DB completa y normalizada hasta 3NF (imagen del modelo relacional desde PHPMyAdmin / tab Designer).

- 4.2.3 Diagrama ER (ímagen)

Add an item

3 - Recursos de un Sistema Computacional (Diego Urgell y Bruno Santos)

[Hide completed items](#) [Delete](#)

100%

- 4.3.1 Aplicación Productor-Consumidor (Base de Datos y Sensores Simulados).

Add an item

5 - Administración de Proyectos (Brian Serranía)

[Hide completed items](#) [Delete](#)

100%

- 4.5.1 Seguimiento puntual de las actividades registradas en Trello de la 3ta etapa y la entrega final con las fechas y responsables de cada equipo.

Add an item

6 - Internet de las Cosas (IoT) (Iñigo Zepeda y Diego Rodríguez)

[Hide completed items](#) [Delete](#)

100%

- 4.6.1 Documentación de la topología y el proceso.

- 4.6.2 Debe incluirse un diagrama con las direcciones y los protocolos involucrados

- 4.6.1 Documentación de la topología y el proceso.
- 4.6.2 Debe incluirse un diagrama con las direcciones y los protocolos involucrados
- 4.6.3 así como la explicación de la interacción entre los diferentes dispositivos.
- 4.6.4 Se deben incluir 4 sensores uno de CO, otro CO2 y otros 2 a decidir por el equipo justificando su elección y explicando cómo estos sensores podrían ayudar en la interpretación de los datos.

Add an item

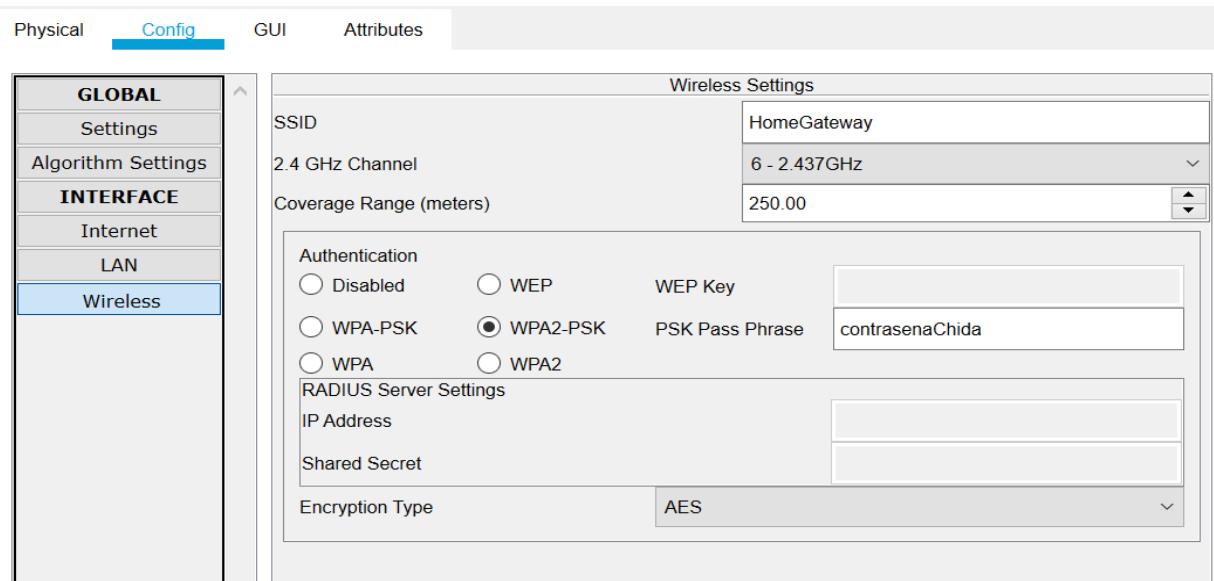
6. Internet de las cosas (IoT)

4.6.1 Documentación de la topología y el proceso.

Lo primero que haremos será agregar un Home Gateway con el cual conectaremos los 4 sensores de los cuales recibiremos en el server y posteriormente enviaremos hacia nuestra base de datos en línea.

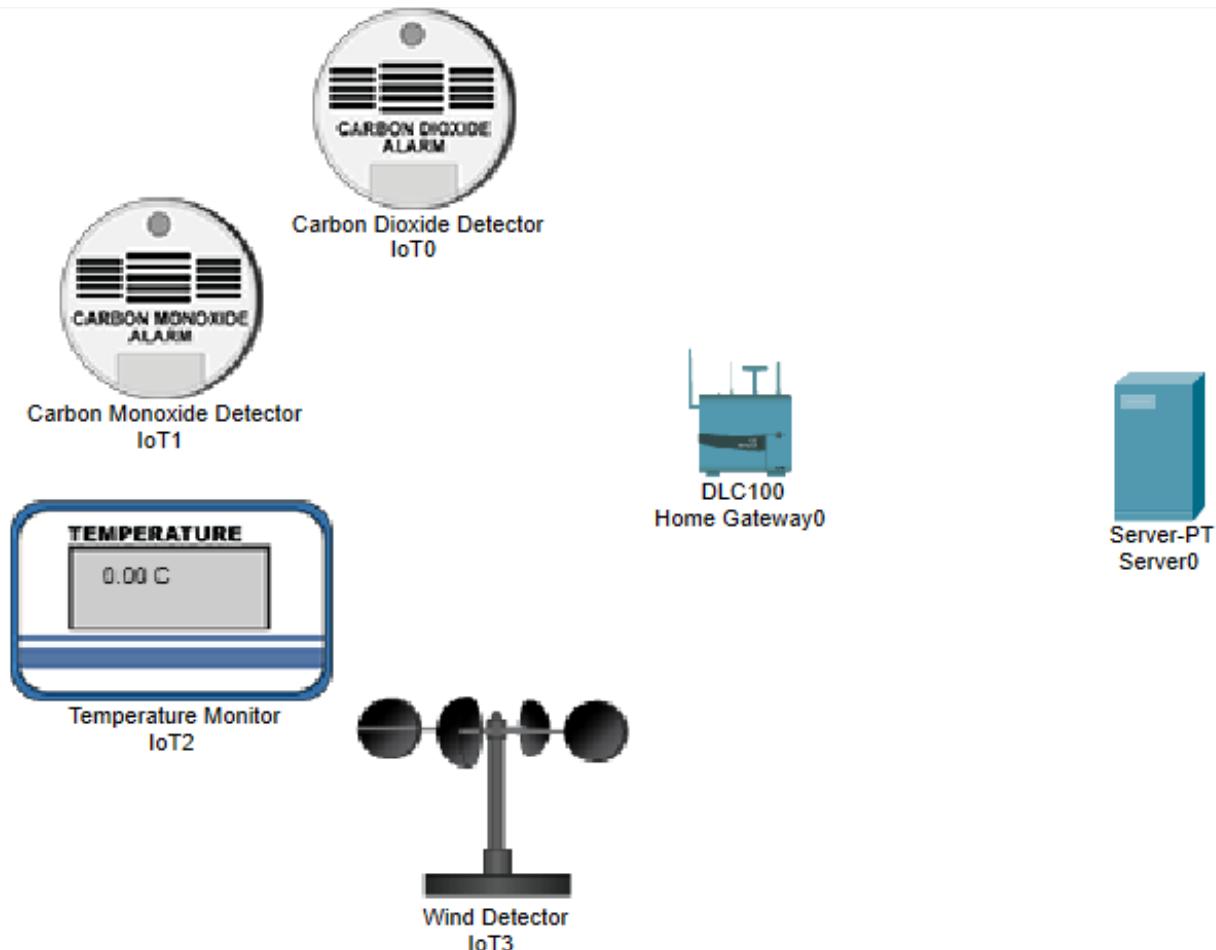


Ahora que agregamos este dispositivo modificaremos sus atributos wireless haciendo que este esté protegido por una contraseña que nosotros mismo creamos. Esta contraseña será utilizada por los 2 sensores que estarán conectados a este para poder recibir los datos de estos.

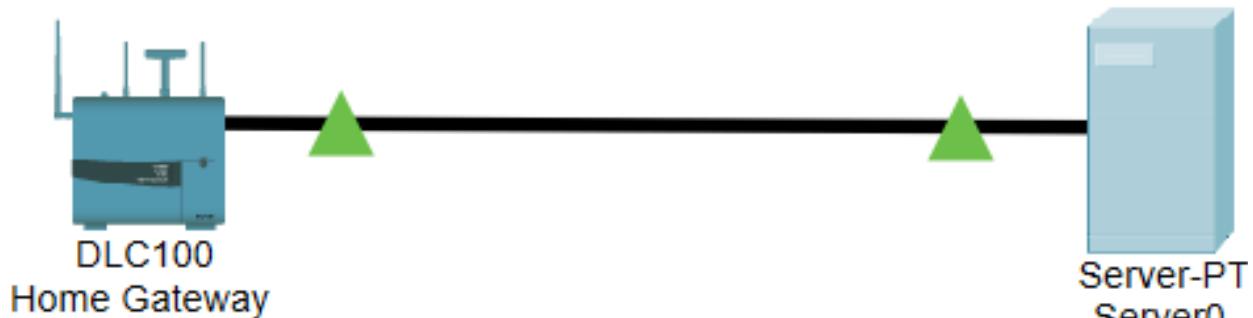


Wireless Settings			
SSID	HomeGateway		
2.4 GHz Channel	6 - 2.437GHz		
Coverage Range (meters)	250.00		
Authentication <input type="radio"/> Disabled <input type="radio"/> WEP WEP Key <input type="radio"/> WPA-PSK <input checked="" type="radio"/> WPA2-PSK PSK Pass Phrase: contraseñaChida <input type="radio"/> WPA <input type="radio"/> WPA2			
RADIUS Server Settings IP Address Shared Secret			
Encryption Type	AES		

El siguiente paso será agregar los 4 sensores los cuales conectaremos al Gateway recién creado y un servidor al cual se le transmitirán los datos. Utilizaremos un detector de viento, monitor de temperatura, detector de monóxido de carbono y un detector de dióxido de carbono.

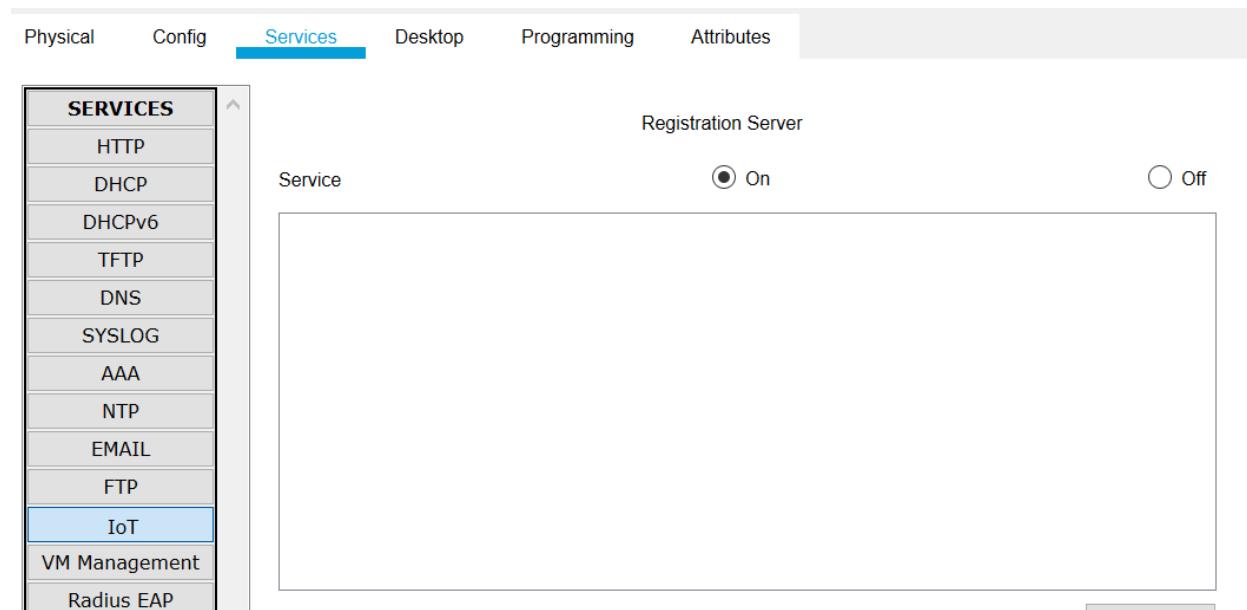


Primero conectaremos el servidor hacia el Gateway con un cable de cobre desde el puerto Ethernet 1 del Gateway hacia el puerto FastEthernet0 del servidor para una conexión rápida y segura.



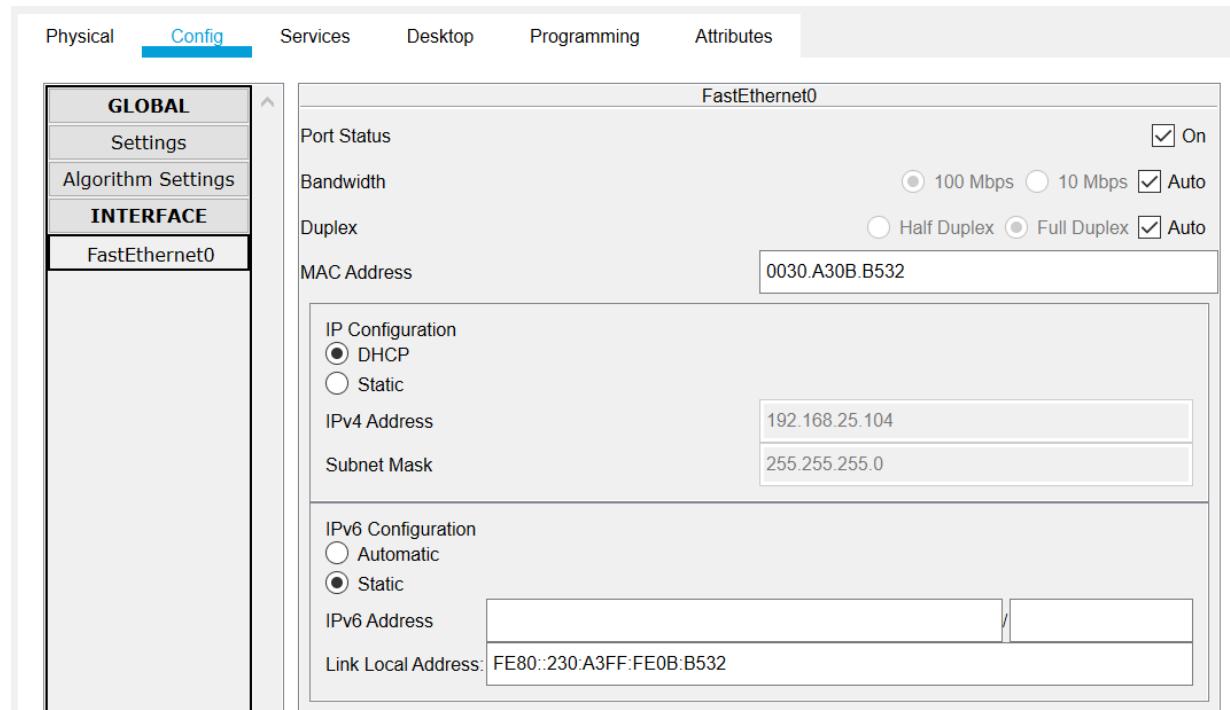
Ahora para dejar listo el servidor y que podamos conectar ambos sensores a este, crearemos una cuenta desde la interfaz de internet de una tablet.

Primero cambiaremos el ajuste de IoT y habilitaremos el registro en el servidor.



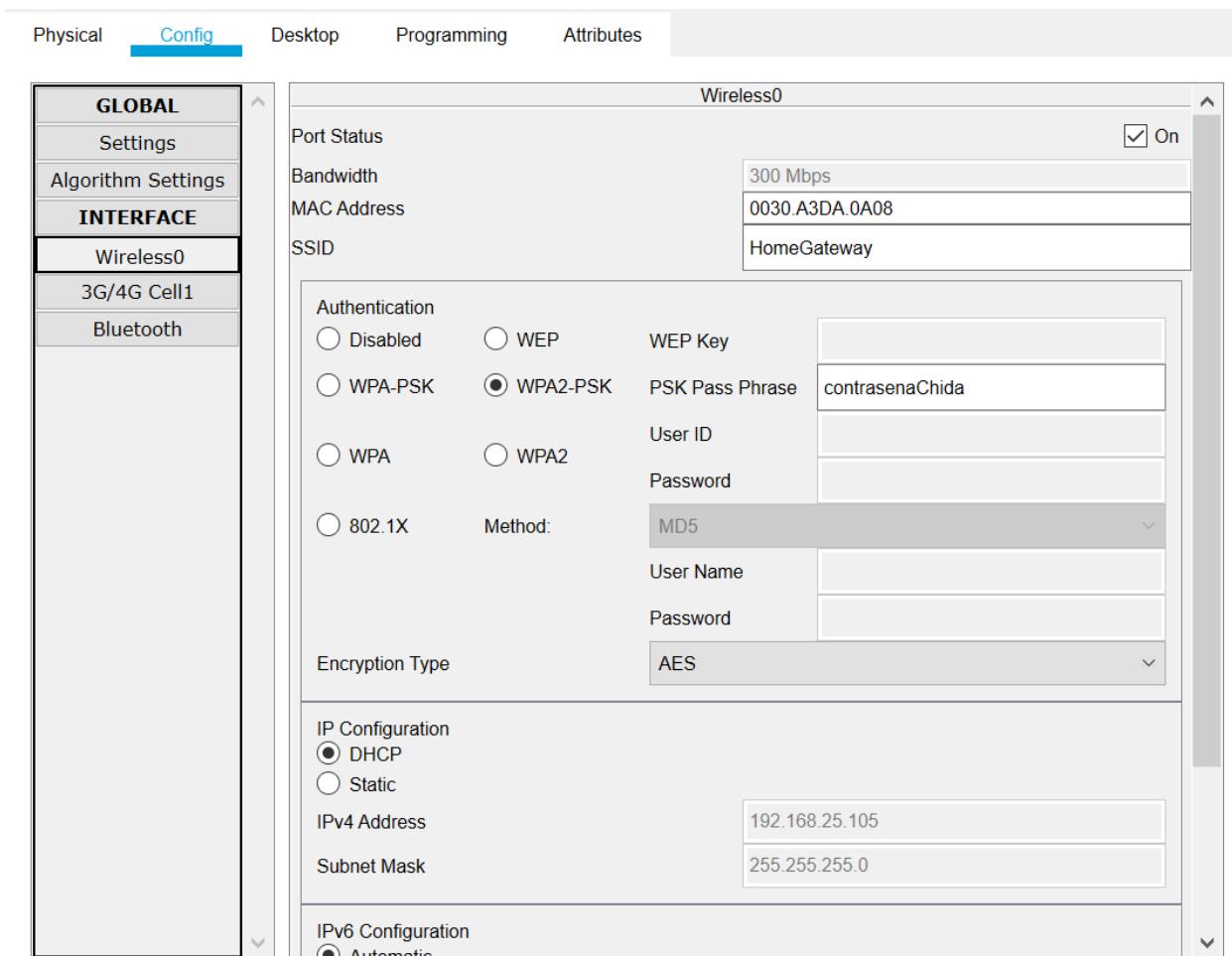
The screenshot shows a software interface for managing services. The top navigation bar includes tabs for Physical, Config, Services (which is highlighted in blue), Desktop, Programming, and Attributes. On the left, a sidebar lists various services: HTTP, DHCP, DHCPv6, TFTP, DNS, SYSLOG, AAA, NTP, EMAIL, FTP, IoT (which is selected and highlighted in blue), VM Management, and Radius EAP. The main panel displays the 'Registration Server' configuration for the IoT service. It shows a status section with 'Service' and two radio buttons: 'On' (selected) and 'Off'. Below this is a large empty text area for configuration details.

Después igualmente dentro del servidor nos iremos a la configuración de la conexión y cambiaremos la configuración de IP a DHCP para que cree una IP propia para después conectarnos con la tablet.



The screenshot shows a software interface for network configuration. The top navigation bar includes tabs for Physical, Config (which is highlighted in blue), Services, Desktop, Programming, and Attributes. On the left, a sidebar lists GLOBAL, Settings, Algorithm Settings, and INTERFACE. Under INTERFACE, 'FastEthernet0' is selected. The main panel displays configuration for the 'FastEthernet0' interface. It includes sections for Port Status (checked), Bandwidth (100 Mbps selected), Duplex (Full Duplex selected), MAC Address (0030.A30B.B532), IP Configuration (DHCP selected), IPv4 Address (192.168.25.104), Subnet Mask (255.255.255.0), IPv6 Configuration (Automatic selected), IPv6 Address (FE80::230:A3FF:FE0B:B532), and Link Local Address (also FE80::230:A3FF:FE0B:B532).

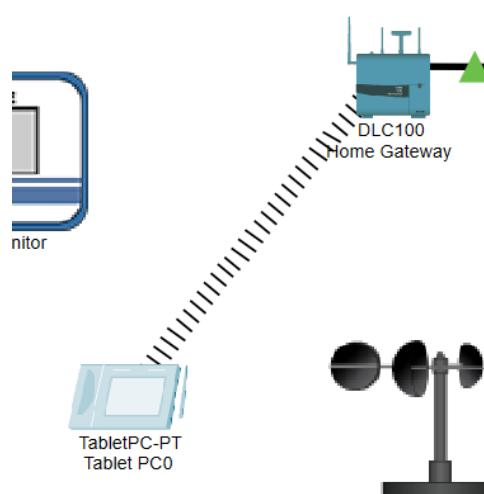
Ahora crearemos la tablet y después la conectaremos a nuestro Gateway para tener acceso hacia el servidor.



The screenshot shows a network configuration interface with the following details:

- Wireless0** tab selected.
- Port Status**: On (checked).
- Bandwidth**: 300 Mbps.
- MAC Address**: 0030.A3DA.0A08.
- SSID**: HomeGateway.
- Authentication**:
 - WPA2-PSK is selected.
 - WEP, WPA, and 802.1X options are available but not selected.
 - WEP Key field is empty.
 - PSK Pass Phrase: contrasenaChida.
 - User ID and Password fields are empty.
 - Method: MD5.
 - User Name and Password fields are empty.
- Encryption Type**: AES.
- IP Configuration**:
 - DHCP is selected.
 - Static option is available but not selected.
 - IPv4 Address: 192.168.25.105.
 - Subnet Mask: 255.255.255.0.
- IPv6 Configuration**: Automatic.

Después de haber cambiado la SSID y haber puesto la contraseña de nuestro Gateway nuestra tablet aparecerá conectada de forma inalámbrica.



Ahora entraremos desde el internet de la tablet hacia el servidor y veremos un inicio de sesión normal.

Sign up now'." data-bbox="112 145 876 401"/>

Web Browser

< > URL http://192.168.25.100 Go Stop X

Registration Server Login

Username:

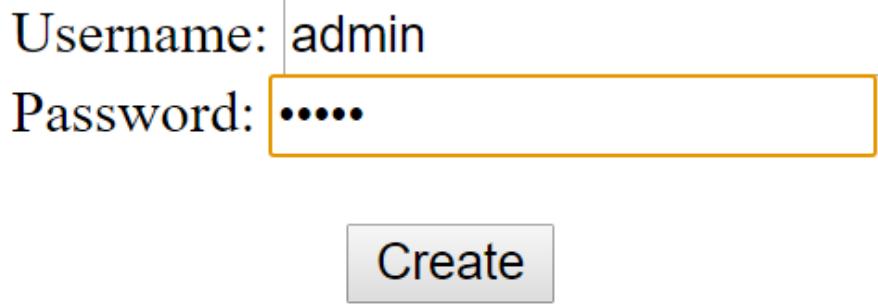
Password:

Sign In

Don't have an IoE account? [Sign up now](#)

Como no tenemos ninguna cuenta creada ahora, crearemos una nueva la cual hará que posteriormente los sensores puedan conectarse al servidor.

Registration Server Account Creation



Username: admin

Password:

Create

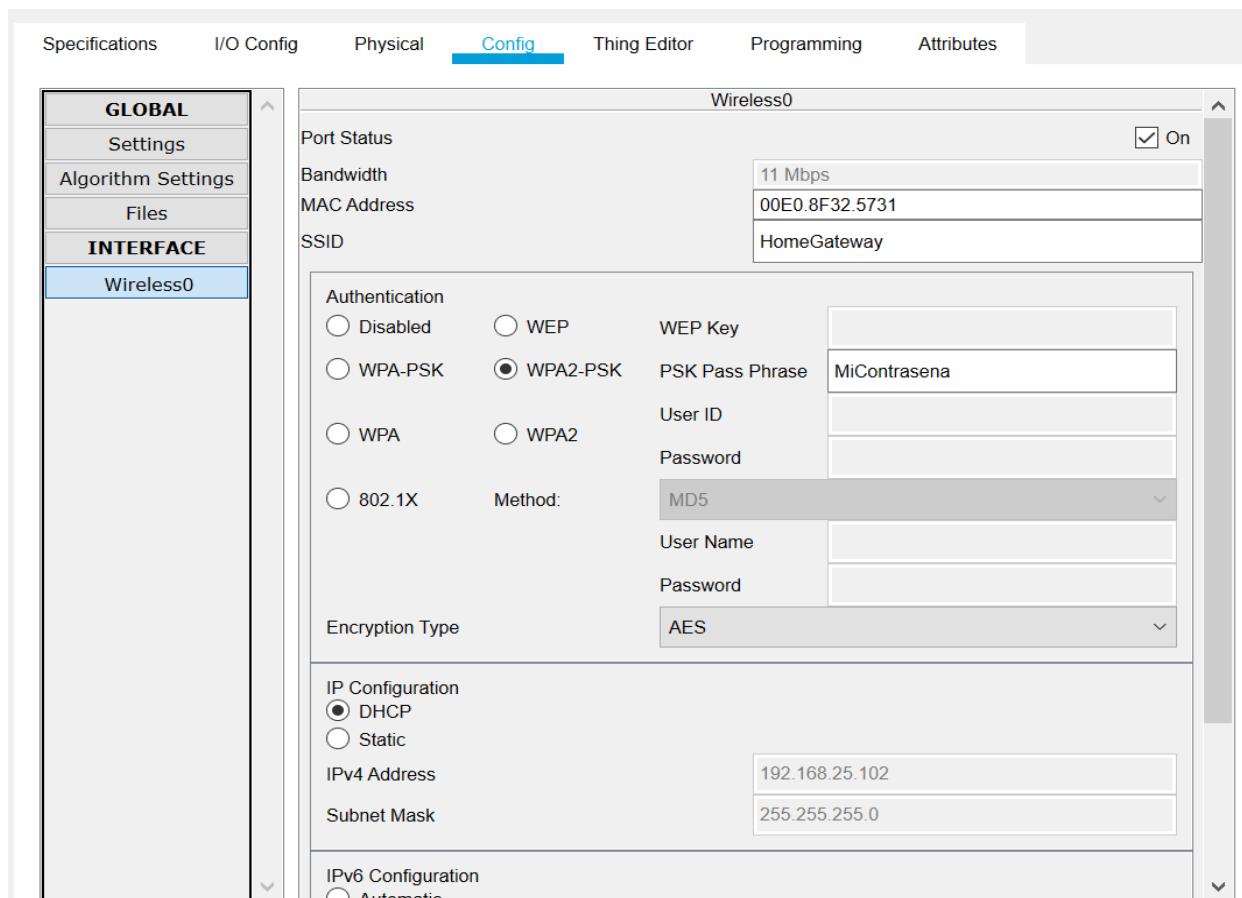
Utilizaremos admin como usuario y contraseña para facilitar el ingreso de información posteriormente. Ya con esto está completamente listo el servidor para la conexión de los sensores.

El primer paso para conectar estos sensores será cambiar adaptador de red el cual es un PT-IOT-NM-1CE a uno inalámbrico el cual será PT-IOT-NM-1W. Este paso aplica para los 4 sensores ya que con el otro adaptador solo acepta conexiones alámbricas y en este caso queremos que sean inalámbricas.

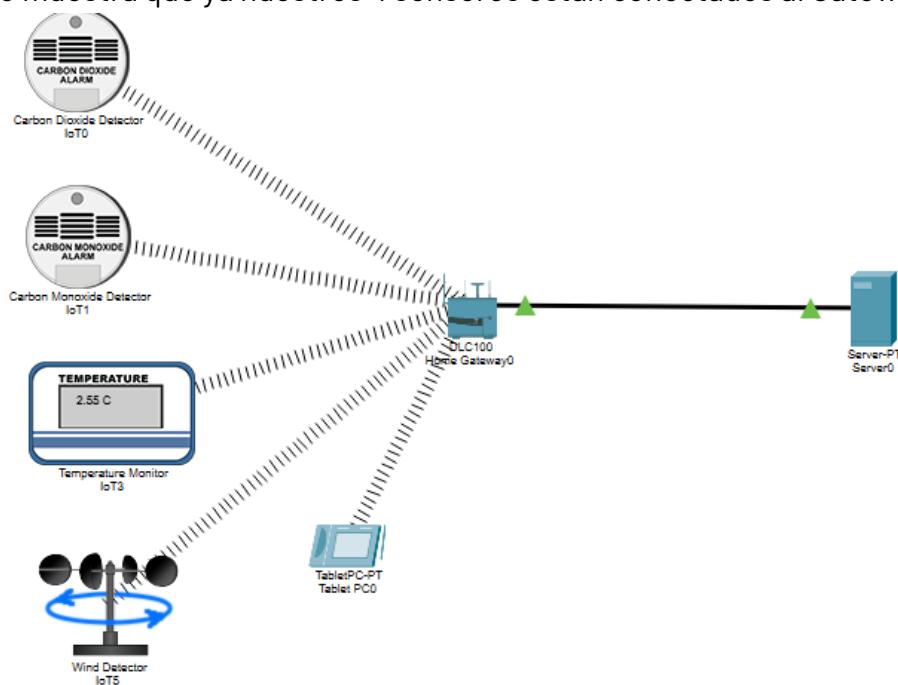
Network Adapter

PT-IOT-NM-1W

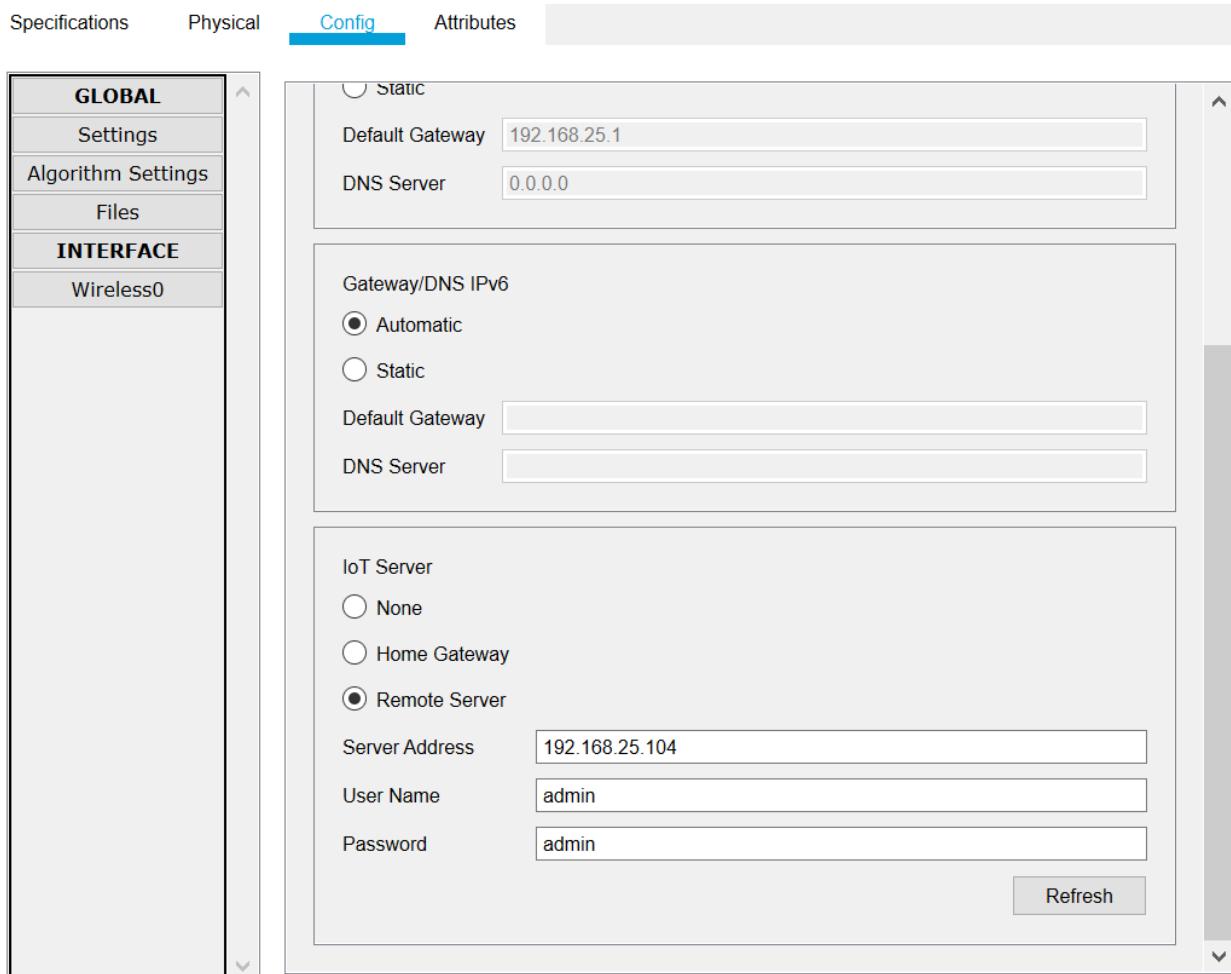
Después conectaremos ambos sensores a nuestro Gateway para así tener acceso al servidor.



Así se muestra que ya nuestros 4 sensores están conectados al Gateway:



Ahora cambiaremos los ajustes globales de los sensores a como se muestra en la imagen ya ingresando con la cuenta y contraseña que creamos en el servidor y le daremos a conectar para que desde el servidor se tenga acceso a los sensores.



Ahora si comprobamos desde la tablet la IP del servidor para comprobar que ya están conectados todos los sensores al server.

Physical Config **Desktop** Programming Attributes

Web Browser X

< > URL http://192.168.25.104/home.html Go Stop

IoT Server - Devices Home | Conditions | Editor | Log Out

▼ ● IoT0 (PTT0810AEP1-)	Carbon Dixoide Detector
No remote control API.	
▼ ● IoT1 (PTT08106Q52-)	Carbon Monoxide Detector
Alarm	●
Level	0.00000572205
▼ ● IoT3 (PTT0810UL0N-)	Temperature Monitor
Temperature	18.5 °C
▼ ● IoT5 (PTT08107FH0-)	Wind Detector
Wind	●

Ahora se empezará con la programación para que se reciba los datos de los sensores, los interprete y de ahí mismo los mande hacia nuestro hosting.

Toda la programación se realizará dentro de los programas del servidor utilizando una plantilla base la cual es Real HTTP Client con un lenguaje de programación python. A continuación, está el código utilizado para la recepción de datos y la operación para mandar los datos al hosting real:

```
from realhttp import*
from time import*
from environment import Environment
from ioeclient import IoEClient

def onHTTPDone(status, data):
    print("status:"+str(status))
    print("data:"+data)
```

```

def main():
    http=RealHTTPClient()
    http.onDone(onHTTPDone)

    day = 1
    hour = 0

    while day <=30:

        print(" ")
        co2_level=Environment.get("CO2")
        co_level=Environment.get("CO")
        ws_level=Environment.get("Wind Speed")
        temp_level=Environment.get("Ambient Temperature")
        dt=("2020-01-{0}{1}:59:00")
        dt=dt.format(day, hour)

        query_insert="INSERT INTO `Medida`(`tiempo`, `co`, `co2`, `wsp`, `tmp`)
VALUES ('{0}', {1}, {2}, {3}, {4}) "
        query_insert = query_insert.format(dt,co_level,co2_level,ws_level, .
temp_level)

        query_update="ON DUPLICATE KEY UPDATE `co`={0}, `co2`={1}, `wsp`={2},
`tmp`={3}"
        query_update = query_update.format(round(co_level,2),round(co2_level,
2), round(ws_level, 2),
round(temp_level, 2))

        query = query_insert + query_update
        print(query)

        url_hh = 'http://diegorem99.heliohost.org/insertMatlab.php?query='
        url_hh = url_hh + query

        url_th = 'http://10.48.149.219/Gpo31-E6/insertMatlab.php?query='
        url_th = url_th + query

        print(dt)

        print("Helio Host")
        http.get(url_hh)

        print("Tec Host")
        http.get(url_th)

        hour = (hour + 1) % 24

        if hour == 23:
            day += 1

        sleep(2)

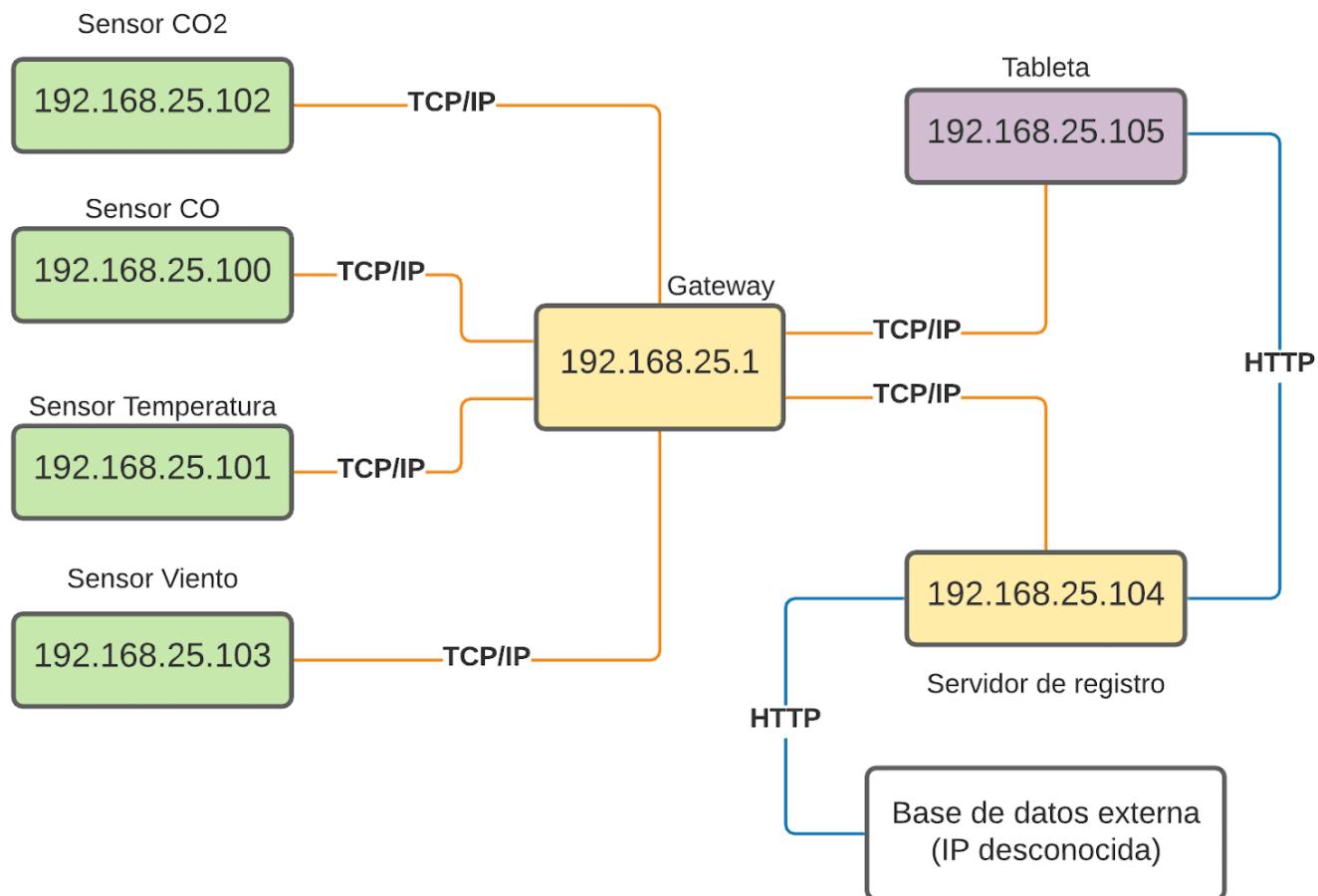
if __name__ == "__main__":
    main()

```

Y al correr este programa arroja los datos que se envían hacia el hosting en la consola, cabe recalcar que mientras siga corriendo el programa, cada segundo seguirá mandando los datos actuales a la base de datos.

```
Starting New Project (Python)...
()
INSERT INTO `Medida` (`tiempo`, `co`, `CO2`, `WSP`, `tmp`) VALUES ('2020-11-22 21:59:00', 0.000005722049991163658,
0.036010801792144775, 1.78643000125885, 2.865339994430542)
New Project (Python) stopped.
```

4.6.2 Debe incluirse un diagrama con las direcciones y los protocolos involucrados.



4.6.3 Así como la explicación de la interacción entre los diferentes dispositivos.

Los dispositivos interactúan de manera integrada para lograr la obtención de los datos y su envío a la base de datos en un servidor externo (Helio Host y Tec Host). En primer lugar, es importante mencionar que todos los dispositivos, los cuatro sensores, la gateway, la tableta y el servidor de registro, están conectados por medio del protocolo TCP/IP, dentro de una

sola red. Esto significa que entre ellos es posible intercambiar información fácilmente. Los sensores producen los datos, y el servidor tiene acceso a ellos porque funciona como un servidor de registro IoT y obtiene la información mediante el protocolo TCP/IP. Por otro lado, la tableta permite ver la información de cada sensor de manera interactiva. En este caso es necesario realizar una request utilizando el protocolo HTTP, dirigida al servidor de registro, que es el que tiene acceso directo a los datos.

Finalmente, a partir de Packet Tracer es posible hacer HTTP requests a servidores externos (fuera de la aplicación) y reales. Se aprovechó esta funcionalidad para insertar la información producida por los sensores en las bases de datos almacenadas en Tec Host y en Helio Host. Un script de PHP almacenado en los servidores recibe la query como un parámetro de URL y realiza la inserción correspondiente. De esta manera la información puede llegar desde Packet Tracer a la base de datos.

[4.6.4. Se deben incluir 4 sensores uno de CO, otro CO2 y otros 2 a decidir por el equipo justificando su elección y explicando como estos sensores podrían ayudar en la interpretación de los datos.](#)

Además de incluir los sensores de CO y CO₂, decidimos implementar otros dos tipos de sensores. Optamos por el uso de un sensor de temperatura, y también de un sensor de velocidad del viento. De tal manera que contemplamos como primordiales los sensores de monóxido de carbono y dióxido de carbono, tomamos las variables más cercanas a las que corresponden a estos. Sabemos que cuando hay una mayor concentración de monóxido y dióxido de carbono, esto genera un cambio en la temperatura, la cual asciende. Además de eso, la temperatura puede llegar a ser utilizada como una medida para la calidad del aire. Tomando esto en cuenta, es donde entraría el sensor de temperatura, que a nuestro ver, es esencial para el desarrollo de nuestro reto. En cuanto a la velocidad del viento, sabemos que este es un factor que puede generar una variación en los niveles de los gases, ya que en el mismo viento se concentra el dióxido y monóxido de carbono, y de eso podemos concluir que entre menos viento más concentradas las sustancias y viceversa.

6. Conclusiones.

5.1 Mencionar la importancia de cada uno de los temas de cada módulo en la implementación de la solución al reto.

Los conocimientos adquiridos durante los dos bloques fueron de gran utilidad para resolver la situación problema y diseñar una solución que integrara todos los componentes necesarios de un sistema de información IoT. En primer lugar, a pesar de que no tuvimos acceso a hardware de forma física, el módulo de Sistemas Digitales fue útil para implementar un modelo de Simulink que simula seis sensores, que leen información sobre ciertas variables ambientales obtenida del Sistema de Monitoreo Ambiental (SIMAT) y obtiene el promedio mediante un bit shift. Además, se incluyen comparadores que determinan si ciertas lecturas están por encima del nivel recomendado, así como un contador de ocurrencias para estos casos. De igual manera se implementó una máquina de estados que indica en qué nivel de contingencia en cada momento de acuerdo con ciertos límites establecidos. Para todo el modelo se puso en práctica el conocimiento de sistemas numéricos, álgebra booleana, contadores, flip-flops, lógica combinacional, etc.

Además del hardware, una parte importante del sistema es el software. En cuanto al módulo de bases de datos, fue fundamental ya que es lo que permite almacenar la información generada por los sensores y acceder a ella posteriormente. En primer lugar, se diseñó la base de datos conceptualmente, utilizando el modelo entidad relación y posteriormente el modelo relacional. Esto permitió definir bien cuáles fueron los atributos y relaciones necesarias. Además, se definió la relación entre ellas. Posteriormente se optimizó el diseño de la base de datos mediante el proceso de normalización, hasta que se encontró en tercera forma normal. Esto permitiría guardar la información de manera segura y garantizando la integridad de los datos. Además, se utiliza la menor cantidad de almacenamiento posible. De igual forma, el conocimiento de SQL nos permitió realizar la inserción de los datos por medio de un script PHP que crea una conexión con el servidor. Los temas aprendidos durante este módulo fueron cruciales para el desarrollo del sistema.

Igualmente, dentro de la categoría de software, el módulo de administración de recursos de un sistema computacional fue muy útil, aunque no se aplicó de manera directa en el reto. En primer lugar, nos permitió comprender cómo las diferentes arquitecturas

computacionales tienen un rol importante en un sistema de información IoT. Posteriormente, conocimos cómo los procesos concurrentes pueden llevar a pérdida de información en una base de datos si no se manejan apropiadamente, y qué técnicas se pueden utilizar para administrarlos, como un lock. Estos conocimientos se pusieron en práctica mediante una aplicación de tipo productor consumidor, que nos enseñó cómo se pueden administrar updates concurrentes en una base de datos.

La parte más tangible del proyecto es que en la página web, fue utilizado un diseño óptimo para que el usuario puede moverse fácilmente entre las opciones proporcionadas a él. De este modo también puede escoger los rangos de tiempo en los cuales quiere que le aparezca la información.

En cuanto a la administración de proyectos nos dimos cuenta la importancia que tiene esto para poder trabajar de una manera fluida y que todos estemos en el mismo canal avanzando en cierto proyecto, es importante mencionar que de alguna manera no le veíamos tanta importancia a este proceso en específico, conforme fuimos avanzando nos dimos cuenta de que en realidad si no teníamos esta organización.

Como equipo nos vamos muy contentos de que a pesar de ser ingenieros tengamos este tipo de acercamiento con temas no tan relativos a nuestra carrera pues en muchas ocasiones no estaremos trabajando como administradores o mercadólogos que nos apoyen a darle seguimiento, entonces fue realmente bueno poder hacer un proyecto de esta índole. Finalmente, un módulo que permitió integrar el conocimiento de los demás y entender cómo se aplican en un sistema de información, fue el de IoT. Aprendimos diversas arquitecturas de Internet de las Cosas, y maneras de conectar los datos con el usuario final. Algo que fue de mucha utilidad fue el concepto de "Fog layer", pues fue lo que se implementó en Simulink al obtener el promedio de los datos. También aprendimos a utilizar Packet Tracer, un software que permite modelar sistemas de redes e incluso conectarlas con el mundo exterior. Implementamos un sistema que tenía cuatro sensores, un gateway, una tableta y un servidor de registro. Mediante un script de Python, fuimos capaces de enviar una query con los datos a nuestra base de datos real. Los conocimientos de este módulo complementaron muy bien lo aprendido en Sistemas Digitales en cuanto a los sensores, y nos permitieron entender la interacción entre dispositivos físicos.

5.2 Mencionar por cada miembro (con nombre incluido) del equipo la experiencia y aprendizajes del reto

∅ **Bruno Santos:** Desde un principio me había llamado la atención la materia de IoT, aunque no tenía un conocimiento profundo acerca de los temas.

Durante la realización de este reto vimos diversos temas, temas que aunque fueran diferentes, teníamos que encontrar una manera de juntarlos todos para concluir con éxito nuestro reto. En lo personal, no tuve grandes dificultades en la materia, con excepción de los módulos de administración de recursos de un sistema computacional, y de base de datos, que fueron dos que me costaron más trabajo en comprender los temas, y ver la manera de aplicar todo lo aprendido en la resolución de nuestro reto.

Un punto positivo que me gustaría agregar es acerca de la forma de trabajo, donde desde un principio nos fue dada la instrucción de definir roles. Y aunque fueran definidos, tuvimos la oportunidad de cooperar entre todos, para un mejor desempeño en equipo.

El área de IoT, aunque se me hizo muy interesante, no creo que sea algo con lo que quiera trabajar. De todos modos, considero muy importante, y de mucho valor el hecho de que tengamos la oportunidad de ver diversos temas, y que seguramente en un futuro podrán abrirnos nuevas puertas.

En lo general, salgo muy satisfecho con la materia y con la solución del reto propuesta por mis compañeros y yo. Esperaba tal vez un contacto más directo con los sensores y cosas del tipo, pero es comprensible que por la situación pandémica que estamos no fuera posible hacerlo de tal manera.

∅ **Diego Rodríguez:** En mi opinión aprecio ver este reto ya que la verdad la mayoría de estos temas no los esperaba ver en mi vida o hasta que estuviera trabajando así que todo lo aprendido lo valoro mucho y espero aplicarlo de una buena manera en el futuro, todo lo que vimos se me hizo de sumamente interesante y en ningún momento creí que lo que enseñaban no me fuera a servir para algo, aunque fuera de conocimiento general. Me gustó la forma en la que se combinaron todos los temas de los 6 módulos, en el principio la verdad no llegaba a ver la forma en la que los

íbamos a unir, pero poco a poco con el tiempo me di cuenta cuan relacionados estaban todos los temas para llegar a lograr hacer el reto al 100%. De los temas que entre todos que más me gustaron fue la parte del módulo de IoT, aunque no aprendí tanto y de la forma como hubiera querido, me gustaría seguir trabajando con todo esto en la herramienta y tal vez llegar después a profundizar más en este. Después y Bases de datos también sentí que para la carrera en donde estoy era sumamente importante, ya que mi hermano se dedica a vender páginas web por fin siento que puedo llegar a servir un poco en lo que hace.

- ∅ **Iñigo Zepeda:** Este reto de IoT ha sido uno de los más interesantes y complejos que he tenido a lo largo de mi carrera. El primer punto que abordaré es que abarca muchos temas vistos durante clase y semestres anteriores. Esto es de gran importancia puesto que nos empezamos a dar cuenta de que es necesario saber conocimientos pasados.

Fue muy interesante puesto que cada quien en el equipo tenía diferentes tareas que debía de cumplir, para hacer del reto el mejor posible. Algunos se encargaban de Packet Tracer, otros de Matlab, otros de organizar el equipo. Como podemos darnos cuenta, así es como funciona en la vida real un equipo de trabajo, y es fantástico que tuve una probada de esa experiencia laboral.

En mi caso, yo estaba a cargo del front-end de la página web. Me encargué del diseño UI, el cual fue hecho en Adobe XD y Photoshop, y después fue pasado a código en HTML y CSS utilizando Bootstrap y JQuery. De igual manera implementé la funcionalidad para poder leer los datos de la base de datos, en la página web, a través de PHP y MySQL. Realmente fue muy interesante puesto que esto es lo que a mi me gusta, y me agrado que lo pude aplicar a un proyecto bastante elaborado.

- ∅ **Diego Urgell:** Esta unidad de formación fue de mucho aprendizaje muy variado. Me gustó mucho que al final logré entender el funcionamiento del sistema IoT en todos los niveles: producción de los datos, conexión con otros dispositivos, almacenamiento en bases de datos y diseño web para mostrarlos a los usuarios. Creo que este enfoque integral me permitió comprender que los sistemas de software pueden llegar a ser muy complejos, pues algo "sencillo" como un sistema de

monitoreo requirió la implementación de muchos componentes que realizan labores separadas.

En particular, yo implementé el modelo de Simulink que simulaba sensores, y me encargué de que llegaran a la base de datos por medio de un script de Matlab. De igual manera, por el lado de Packet Tracer estuve a cargo de enviar los datos a la base de datos por medio de un script de Python. Una de las mayores dificultades que tuve fue integrar los datos, ya que provenían de fuentes distintas. Tuve que investigar e implementar una solución que permitiera ajustar las variaciones de horas y evitar errores de llave duplicada. La parte que más disfruté fue la administración de recursos computacionales, puesto que estoy muy interesado en el cómputo concurrente, paralelo y distribuido. La aplicación productor consumidor fue un buen ejercicio para poner práctica la administración de datos producidos concurrentemente, y en un futuro me gustaría explorar más de eso.

Algo que me gustó de este bloque fue que en el equipo hubo apoyo y trabajo colaborativo, y creo que es el equipo de mejor rendimiento con el que he trabajado. Creo que repartir las tareas fue muy útil porque así ciertas personas nos concentráramos en áreas específicas y nos esmeramos en conseguir el mejor resultado en lo que nos correspondía, y creo que esto se puede apreciar en nuestros trabajos.

- ∅ **Brian Serranía:** No sé por dónde empezar realmente, ha sido un bloque muy pesado pero satisfactorio, la pandemia fue algo que nos pegó a todos y que nos cambió la vida de la noche a la mañana, agradezco de antemano por todo el apoyo que nos brindaron todos los profesores involucrados en el bloque por dar los mejor de sí mismos, para que todos nosotros pudiéramos aprender. Es interesante todo lo que podemos aportar como equipos y la manera en la que debemos de aterrizar las diferentes ideas para plasmarlas en una idea y plan en común, personalmente me llevo todo lo de administración de proyectos, es un área que me gusta mucho y me doy cuenta de que probablemente mi futuro va por ahí. Gracias nuevamente a los profesores por tanto empeño desarrollado a través del bloque, siento que lo que más me llevo es eso, la pasión y profesionalismo por lo que haces y te gusta hacer.

Referencias.

23X8.(2020). Logic circuit simplification(SOP and POS). Retrieved November 28, 2020, from <http://www.32x8.com/index.html>

Comisión Federal para la Protección contra Riesgos Sanitarios.(2017). Efectos a la salud por la contaminación del aire ambiente. Retrieved October 20, 2020, from <https://www.gob.mx/cofepris/acciones-y-programas/3-efectos-a-la-salud-por-la-contaminacion-del-aire-ambiente>

Day, A. (2014). Electronic Braking Systems. *Braking of Road Vehicles*, 385-428. <https://doi.org/10.1016/b978-0-12-397314-6.00011-5>

Dirección de la Calidad del Aire. (2019). Normatividad. Retrieved November 04, 2020, from <http://www.aire.cdmx.gob.mx/default.php?opc=%27ZaBhnml>

Dirección de la Calidad del Aire. (2020). Programa para Prevenir y Responder a Contingencias Ambientales Atmosféricas en la Ciudad de México. Retrieved November 04, 2020, from <http://www.aire.cdmx.gob.mx/default.php?opc=%27YqBhnmu>

Fisher, Perkins, Walker, & Wolfart.(2003). Bitshift Operators. Retrieved November 04, 2020, from <https://homepages.inf.ed.ac.uk/rbf/HIPR2/bitshift.htm>

GeeksForGeeks. (2020, June 16). *Difference between Computer and Embedded System*. GeeksforGeeks. <https://www.geeksforgeeks.org/difference-between-computer-and-embedded-system/>

Hoffman, C. (2019, September 30). What Is a Checksum (and Why Should You Care)? Retrieved November 06, 2020, from <https://www.howtogeek.com/363735/what-is-a-checksum-and-why-should-you-care>

Mathworks. (2020). Handling and Converting Dates. Retrieved November 06, 2020, from <https://www.mathworks.com/help/finance/handling-and-converting-dates.html>

McKendrick, J. (2016, November 30). Fog Computing: A New IoT Architecture? Retrieved November 04, 2020, from <https://www.rtinsights.com/what-is-fog-computing-open-consortium/>

OMS. (2018). Calidad del aire y salud. Retrieved October 20, 2020, from [https://www.who.int/es/news-room/fact-sheets/detail/ambient-\(outdoor\)-air-quality-and-health](https://www.who.int/es/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health)

Rouse, M. (2019, April 04). What is a Checksum? - Definition from WhatIs.com. Retrieved November 06, 2020, from <https://searchsecurity.techtarget.com/definition/checksum>

SEMARNAT. (2018). Informe del Medio Ambiente: Atomósfera. Retrieved October 20, 2020, from <https://apps1.semarnat.gob.mx:8443/dgeia/informe18/tema/cap5.html>

SCIELO. (2020) Control de la contaminación atmosférica en la Zona Metropolitana del Valle de México. Retrieved November 06, 2020, from http://www.scielo.org.mx/scielo.php?pid=S0186-72102019000300631&script=sci_arttext

Tutoriaspoint. (2019). Fletcher's Checksum. Retrieved November 06, 2020, from <https://www.tutorialspoint.com/fletcher-s-checksum>

XMPP. (2020). Internet of Things (IoT) - Communication Patterns. XMPP. <https://xmpp.org/uses/iot/patterns.html>