



Implementación del Internet de las Cosas (Grupo 06)
Documentación de Reto
Profesor: Roberto Treviño y Rafael Dávalos

Jesús A. Trujillo	A00827538
Edith P. Benvenuto	A00828840
Diego A. Torres	A01283874
Luis Gutierrez	A01283825
Gerardo L. Arrambide	A01283504

© 2020 Derechos reservados: Ninguna parte de esta obra puede ser reproducida o transmitida, mediante ningún sistema o método, electrónico o mecánico, sin conocimiento por escrito de los autores.

Monterrey, Nuevo León, 4 de Diciembre, 2020.

Antecedentes IoT

El Internet de las cosas o mejor conocido como “Internet of Things” es algo que, en la actualidad, forma parte de nuestras vidas en todo momento y sinceramente no me imagino un mundo sin este concepto. El concepto de conectar cualquier objeto a Internet y con ello crear una infinidad de nuevas aplicaciones es algo que asombra pero al mismo tiempo no sorprende. Parece que es un concepto que nos ha acompañado desde hace muchísimo tiempo dada su omnipresencia pero en realidad, este es un concepto bastante reciente.

El término Internet of Things fue dado por primera vez en público por el profesor de MIT Kevin Ashton en una conferencia en 2009, y desde ese momento este concepto ha crecido exponencialmente.

Pero, ¿Qué es el Internet de las cosas?

Se refiere a la interconexión digital de cualquier objeto con Internet, esto los transforma en objetos inteligentes capaces de ejercer autonomía para la resolución de problemas. Esto no solo puede tener un impacto económico en la industria sino también un impacto positivo con el medio ambiente.

A continuación se presenta como implementamos todo lo aprendido sobre este concepto en nuestro proyecto.

Introducción

A lo largo del semestre, estuvimos aprendiendo sobre sistemas digitales y el análisis y diseño de bases de datos. El propósito de este reto será demostrar los conocimientos aprendidos en un sistema informático interconectado entre las casas de los diferentes integrantes del equipo para tener como resultado final, una aplicación domótica que reciba información de los múltiples sensores que existen en el equipo.

El primer paso para lograr esto fue modelar una base de datos que tenga los atributos necesarios para recibir la información que se planeó mandar con los sensores de cada integrante. Para hacer el reto más desafiante, no todos los integrantes contaban con los mismos sensores conectados a los NodeMcu, lo que hace que la base de datos pueda contar con información más variada y dejó un mayor reto para el equipo en términos del modelado y la implementación.

El segundo paso fue conectar cada uno de los componentes necesarios de cada integrante del equipo para poder enviar datos desde sus respectivos NodeMcu a través de las redes WiFi. Asimismo, se fue desarrollando el código que serviría para enviar los datos recibidos del hardware hacia la base de datos alojada en la nube. Desarrollar el código fue de los pasos más retadores del trabajo por las diferentes conexiones que se hicieron para poder enviar exitosamente los datos de los sensores.

El propósito de este documento es explicar más a fondo cada uno de los pasos anteriormente mencionados, desde la perspectiva de cada estudiante, para poder comprobar el trabajo que se realizó durante la clase.

Descripción de los Elementos del Reto

- Microcontrolador NodeMCU

El microcontrolador NodeMCU está basado en Arduino y es compatible con su lenguaje de programación. Este cuenta con un módulo ESP8266 que realiza la conexión a Internet vía WiFi. Esta conexión a Internet nos permite enviar los datos recibidos por los sensores a la base de datos.

Planeamos utilizar este Microcontrolador para que reciba los datos que sean enviados por los sensores y los envíe a la base de datos. Todos los integrantes del equipo deberán tener este Microcontrolador ya que el propósito del reto es que todos manden la información recibida por sus sensores y esta se integre con la información de los demás.

- Protoboard

Un Protoboard es un tablero el cual permite el paso de corriente para crear diferentes circuitos. Este tiene varios orificios en el cual se insertan los diferentes componentes electrónicos y cables para hacer la conexión entre estos. Es indispensable saber cómo funciona y como pasa la corriente a través de este para la creación de cualquier circuito. Debemos utilizar un Protoboard para poder hacer los circuitos que contienen los sensores y el NodeMCU. Como todos los integrantes del equipo decidimos trabajar con el circuito en físico, todos debemos de saber cómo funciona este tablero. En él conectaremos el NodeMCU, los diferentes sensores que cada quien implemente y diferentes leds que indiquen ciertas condiciones. Además utilizaremos cables y resistores para el paso de corriente del nodeMCU a los sensores y a la corriente positiva y negativa.

- Sensor de Humedad y Temperatura

El sensor de temperatura y humedad que se implementó fue el DHT11. Integra un sensor capacitivo de humedad y un termistor para medir el aire. Muestra los datos recibidos mediante una señal digital en el pin de datos.

- Sensor Ultrasónico

El modelo del sensor utilizado es el HC-SR04. Este es un sensor que mide la distancia mediante ultrasonido en un rango de 2 a 450cm. Posee dos transductores, un emisor y un receptor. El emisor emite 8 pulsos de ultrasonido(40 KHz) cuando recibe la orden desde el pin TRIG, las ondas de sonido viajan por el aire hasta rebotar al encontrarse con un objeto. Este impacto es detectado por el receptor y el pin ECHO manda la señal de regreso.

- Sensor de Sonido

El modelo del sensor de sonido utilizado es el KY-037. Este modelo permite detectar cualquier tipo de sonido superior al rango ajustado por el potenciómetro que lleva incluido. Está compuesto por dos salidas, Una salida analogica y otra señal digital que se activa cuando la intensidad del sonido ha alcanzado cierto límite previamente configurado.

- Sensor Magnético

Este sensor es un switch magnético que envía una señal, cuando la separación entre los imanes es de 2 cm o más, este es principalmente utilizado en los accesos tanto de puertas como de ventanas. El modelo utilizado es el ALA-005.

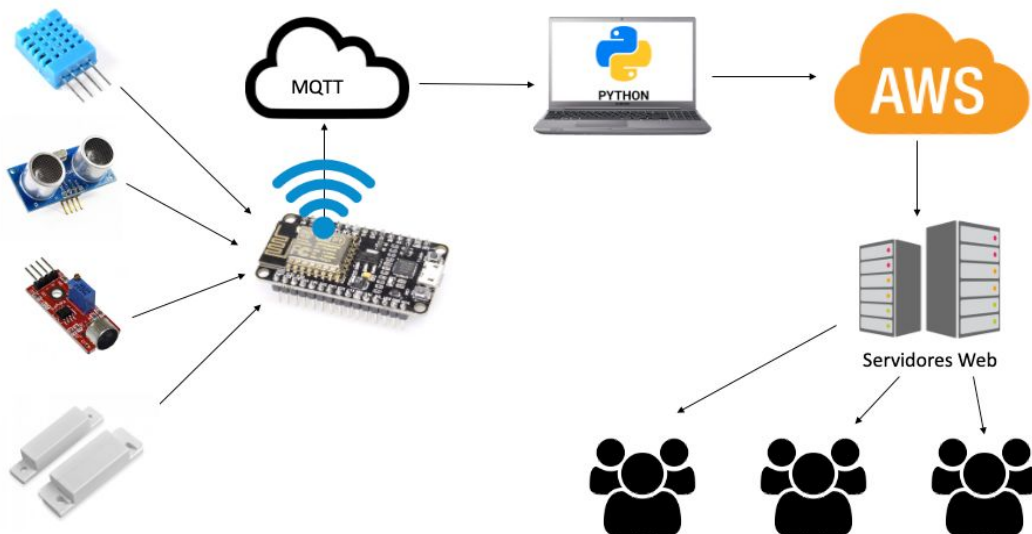
- Broker MQTT

MQTT es un protocolo de red usado para la comunicación machine-to-machine, generalmente ejecutado sobre TCP/IP como base. Es un servicio de mensajería push con patrón pub/sub. La arquitectura del MQTT sigue la topología de estrella donde tiene un nodo central, el servidor también conocido como broker del MQTT es dicho servidor central a donde los clientes se conectan, es el encargado de gestionar la red y enviar los mensajes manteniendo activo el canal. Es importante mencionar que el broker utilizado por el equipo será el HiveMQ, con host en la dirección <http://www.hivemq.com/demos/websocket-client/>

- Base de Datos

Es una colección organizada de datos e información estructurada, que son almacenados para su uso posterior, las bases de datos suelen estar modeladas en tablas (con filas y columnas) para tener una consulta de datos con mayor eficiencia. Se ha decidido incorporar en el diseño el uso de una base de datos relacional con host en AWS.

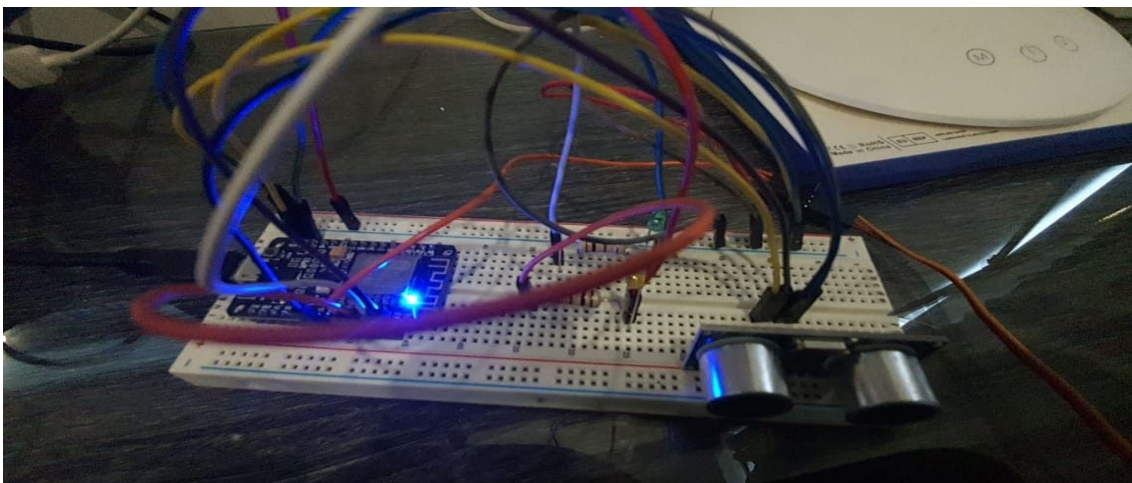
Diseño de los Elementos Interconectados



Desarrollo

Como nadie del equipo estaba familiarizado con la plataforma Proteus, todos optamos por comprar los componentes y hacer el circuito en físico. Por lo tanto el primer paso fue conseguir los componentes y armar el circuito. Esto no fue tan fácil para algunos ya que hubo varios problemas al conectar estos circuitos a la computadora y empezar a manipularlos con código.

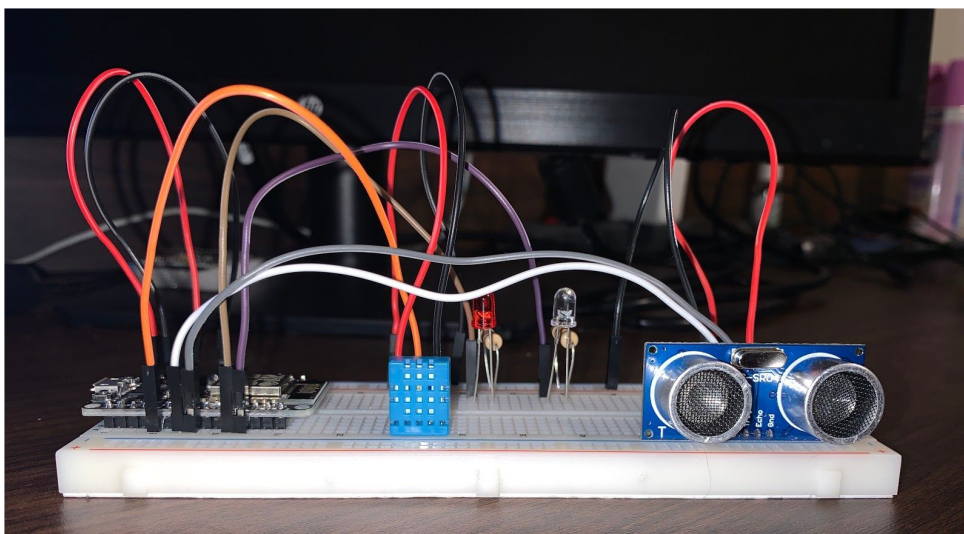
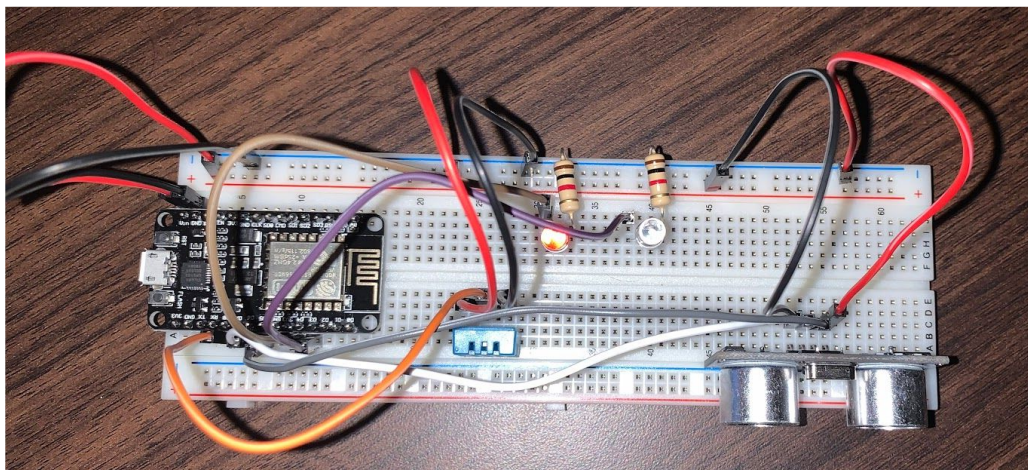
El primer circuito en tener éxito y conectarse correctamente fue el de Jesús Trujillo. No se tuvo ningún problema grave con este circuito, aunque con la finalidad de expandir la funcionalidad de los actuadores se agregaron un servomotor y un relevador de 5 volts, esto para lograr encender una lámpara de escritorio y un abanico, respectivamente.



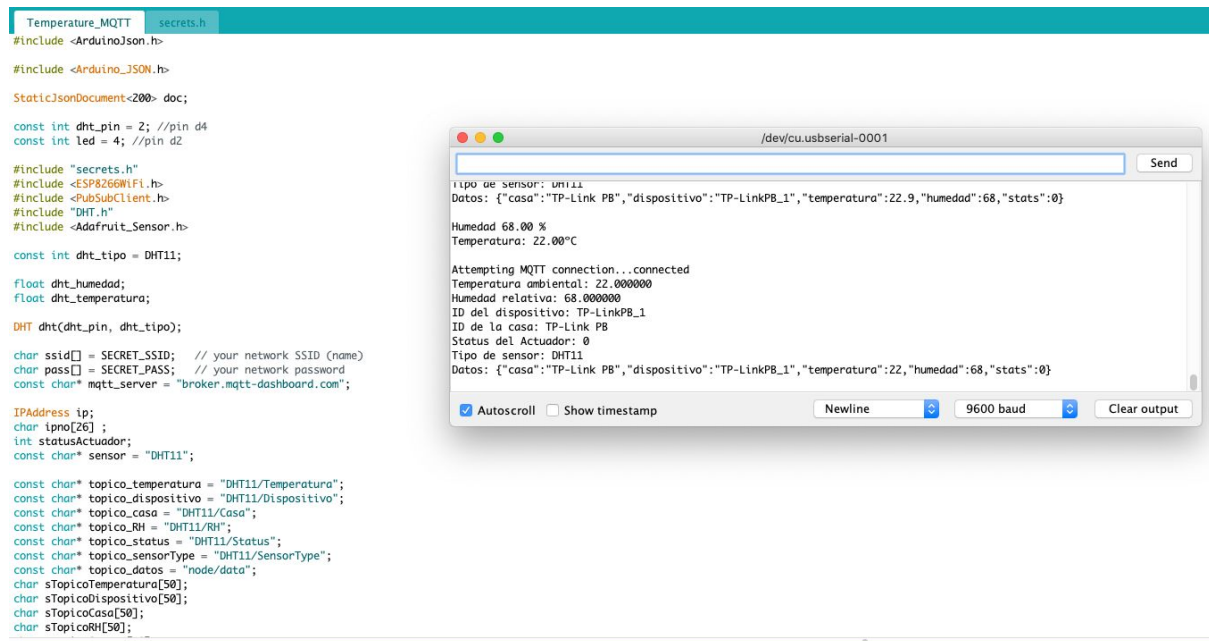
Hubo varios problemas técnicos con el circuito de Gerardo Arrambide. En primera parte se cree que uno de los sensores utilizados en el primer circuito que realizó estaba defectuoso lo que ocasionó un corto. Además de esto, no tenía nada de práctica con este tipo de circuitos eléctricos ya que esta era la primera vez que trabajaba con uno. A consecuencia de esto se quemó el sensor de temperatura y el NodeMCU dejó de funcionar.



Después de conseguir nuevos componentes para reemplazar los defectuosos se volvió a armar el circuito desde 0 y finalmente se llegó a un circuito funcional .



y con este nuevo circuito se logró mandar los datos de temperatura y humedad a la base de datos mediante MQTT.



```
Temperature_MQTT secrets.h
#include <ArduinoJson.h>

#include <Arduino_JSON.h>

StaticJsonDocument<200> doc;

const int dht_pin = 2; //pin d4
const int led = 4; //pin d2

#include "secrets.h"
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <DHT.h>
#include <Adafruit_Sensor.h>

const int dht_tipo = DHT11;

float dht_humedad;
float dht_temperatura;

DHT dht(dht_pin, dht_tipo);

char ssid[] = SECRET_SSID; // your network SSID (name)
char pass[] = SECRET_PASS; // your network password
const char* mqtt_server = "broker.mqtt-dashboard.com";

IPAddress ip;
char ipno[26];
int statusActuador;
const char* sensor = "DHT11";

const char* topico_temperatura = "DHT11/Temperatura";
const char* topico_dispositivo = "DHT11/Dispositivo";
const char* topico_casa = "DHT11/casa";
const char* topico_RH = "DHT11/RH";
const char* topico_status = "DHT11/Status";
const char* topico_sensorType = "DHT11/SensorType";
const char* topico_datos = "node/data";
char sTopicoTemperatura[50];
char sTopicoDispositivo[50];
char sTopicoCasa[50];
char sTopicoRH[50];
```

Serial Monitor (/dev/cu.usbserial-0001) Send

tipo de sensor: DHT11
Datos: {"casa":"TP-Link PB","dispositivo":"TP-LinkPB_1","temperatura":22.9,"humedad":68,"stats":0}
Humedad 68.00 %
Temperatura: 22.00°C
Attempting MQTT connection...connected
Temperatura ambiental: 22.000000
Humedad relativa: 68.000000
ID del dispositivo: TP-LinkPB_1
ID de la casa: TP-Link PB
Status del Actuador: 0
Tipo de sensor: DHT11
Datos: {"casa":"TP-Link PB","dispositivo":"TP-LinkPB_1","temperatura":22,"humedad":68,"stats":0}

☒ Autoscroll ☐ Show timestamp Newline 9600 baud Clear output

Todos los integrantes del equipo utilizaron la placa de desarrollo NodeMCU ESP8266, con excepción de Edith Benvenuto, ya que el modulo de WiFi del NodeMCU era el ESP32, la diferencia más notable entre ellos es que el ESP 32 es un procesador de doble núcleo y el ESP8266 es de uno. Sin embargo, el ESP32 suele tener una falla muy común, que al subir un programa a la placa, esta demora demasiado tiempo y muestra el error “Failed to connect to ESP32: Time out waiting for packet header”, esto sucede cuando el tiempo de espera es excedido, por lo que se soldó un capacitor electrolítico radial, de 10 micro Faradios a 63 Volts, este sirvió para incrementar la potencia eléctrica y para modular la señal en la fuente de alimentación. Después de soldar el capacitor electrolítico (ver en imagen 1), el error desapareció y el NodeMCU ESP32 no presentó más problemas. Sin embargo, en caso que el ESP32 presente otro inconveniente, se consiguió la placa de desarrollo NodeMCU con ESP8266 (imagen 2).

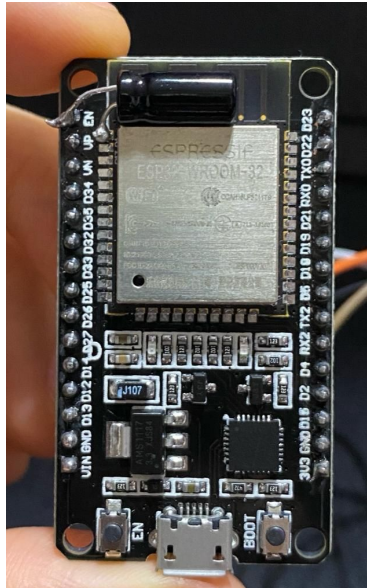


Imagen 1 NodeMCU ESP32 con capacitor electrolítico

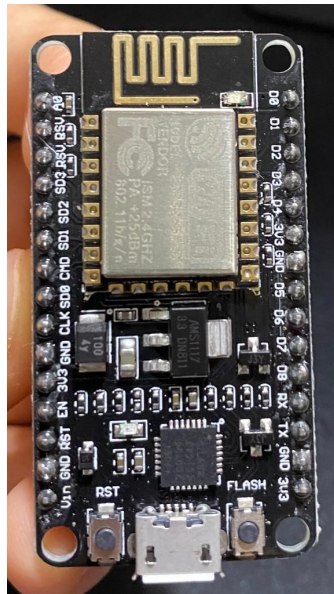
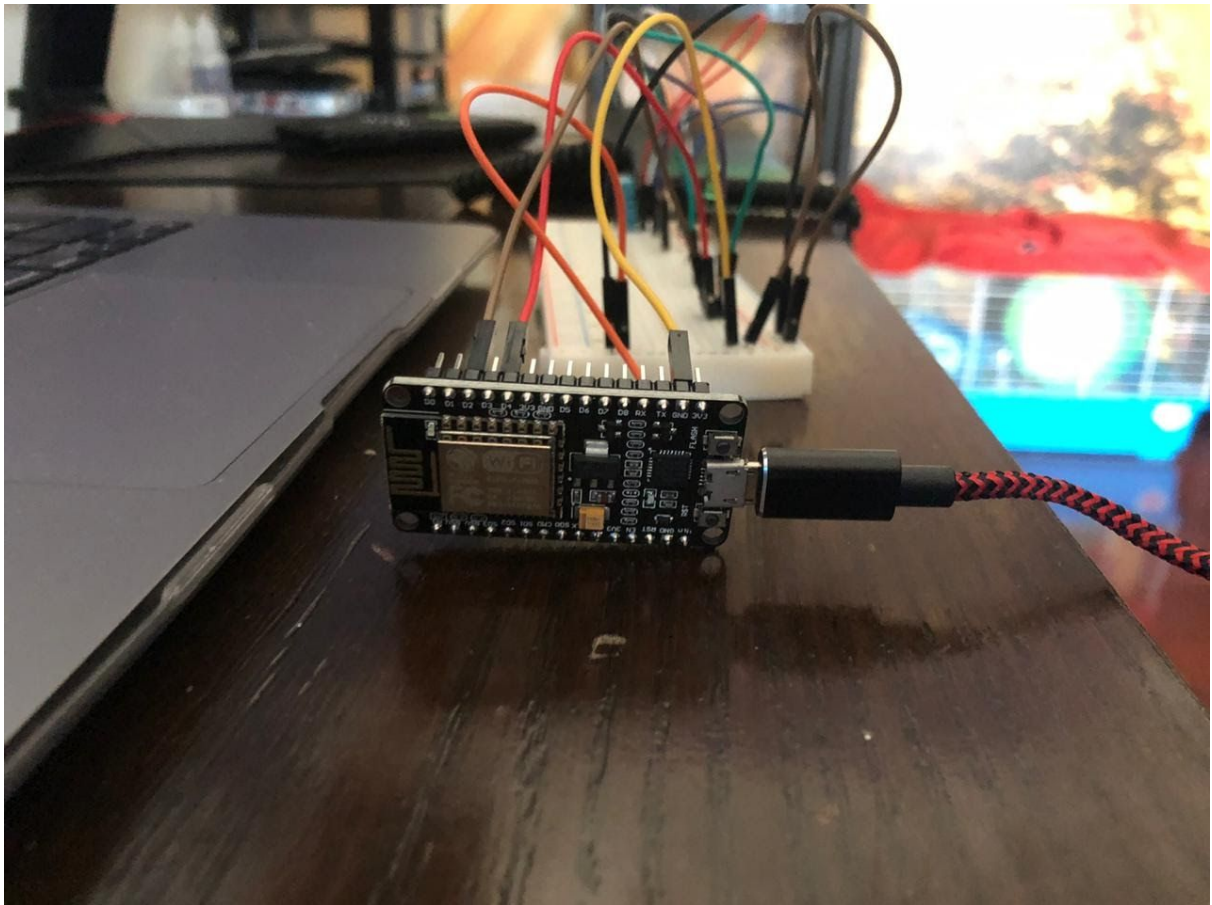
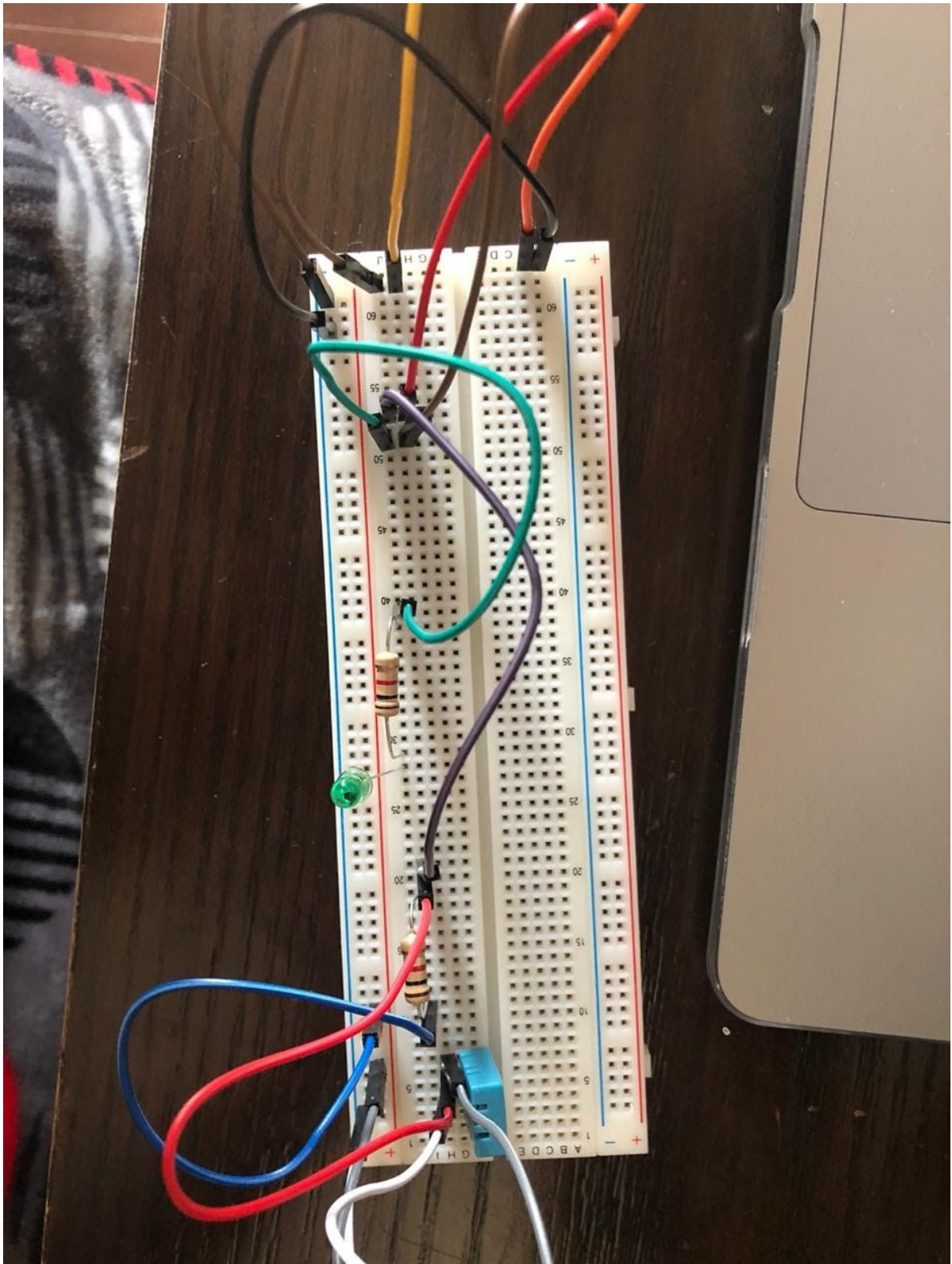


Imagen 2 NodeMCU ESP8266

El siguiente circuito en probarse fue el de Alex Torres, al principio se tuvieron problemas ya que se creía que no iba a funcionar el proyecto en la computadora de Alex por la manera en la que estaba tomando la señal mediante el USB. Esto resultó ser un problema de cómo estaba cableado el protoboard en vez de un problema de la computadora, lo que trajo optimismo al equipo.



Después de esto, Alex se encontró con un problema similar al de Edith, en donde no se recibe el packet header. Este problema se pudo resolver ya que Alex se dio cuenta de que estaba conectándose al puerto incorrecto. Al poder realizar esto, se hicieron unas cuantas pruebas para asegurarnos de que los sensores y otros componentes estuvieran funcionando de manera correcta, no hubo problema por esta parte.



Ya solo faltaba el paso final, conectarse a Wi-Fi para poder enviar la información a la base de datos. Se realizaron múltiples pruebas para intentar hacerlo funcionar pero por alguna razón, el NodeMcu no se conectaba a la red de la casa de Alex, ni a la red de su celular. No logramos comprender por qué sucedía esto entonces

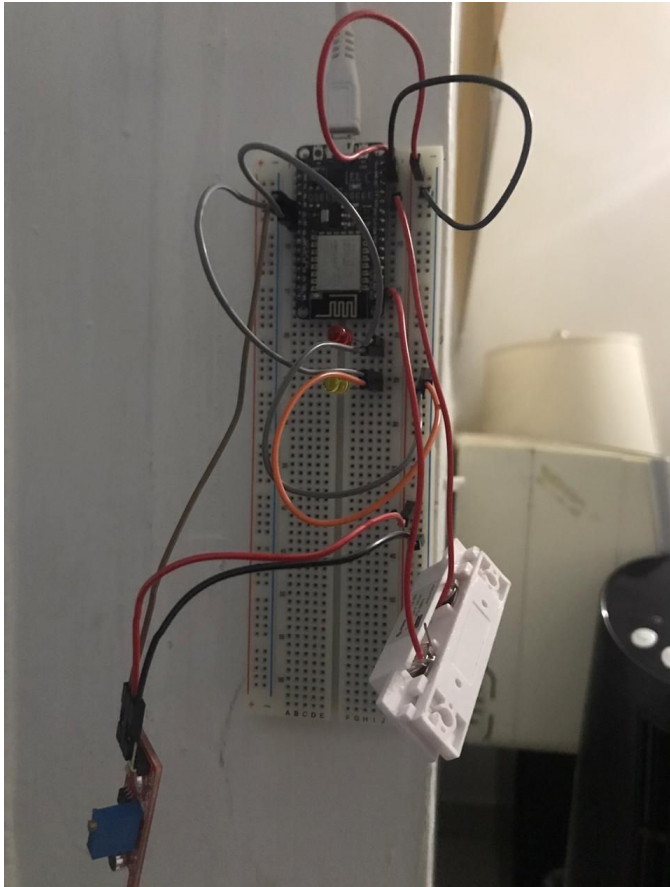
decidimos continuar con el proyecto, sin embargo se aprendió mucho durante esta etapa y no la consideramos como un fracaso ya que algunos integrantes del equipo no conocían sobre hardware antes de este bloque y aún así lograron conectar la placa y recibir información.

Por otra parte, Luis se encargó de la conexión de dos sensores extras al proyecto, uno acústico y otro magnético para puertas. A continuación se describe el proceso llevado a cabo.

Para el sensor acústico, se utilizó su salida digital de información, la cual se activa cuando los datos detectados por el sensor superan la tensión del potenciómetro incluido en el mismo. Seguido a esto, solo se conectó la parte de alimentación de energía a la línea positiva conectada al pin de 5V de la placa y la parte de tierra a la línea negativa conectada a tierra.

Por otra parte, el sensor magnético de puerta implementa una salida analógica la cual recibe los datos de transmisión de corriente a través del circuito. Uno de los extremos se conecta al pin de lectura analógica del NodeMCU y el otro a la alimentación de 3.3V de la placa. Ahora cuando la parte del imán entra en contacto con esta parte conectada, se cierra el circuito y deja pasar la corriente hacia el pin analógico. De este modo, cuando la señal recibida en la lectura analógica es mayor a 0, significa que ambas partes del sensor están unidas, indicando que la puerta está cerrada.

Finalmente, se agregaron dos diodos de emisión de luz, los cuales funcionan como actuadores y son activados cuando se reciben las señales de los sensores.



Desarrollo del script de Arduino

Para poder desarrollar un sistema embebido como el que se solicita por los requerimientos del reto sería necesario escribir un código de Arduino que logre controlar la placa genérica NodeMCU ESP8266 y ESP32. En este caso, el NodeMCU deberá de lograr conectarse con un bróker de protocolo de mensajería MQTT y realizar publicaciones de datos.

El equipo se dió a la tarea de investigar el formato usado para la escritura de tópicos en MQTT, resultando con que lo mejor sería crear un tópico por dato, o crear un tópico de tópicos, el cual sería algo así: “node/#”. Después de realizar pruebas conjuntas con el código de Python se decidió cambiar el formato de los tópicos a uno más sencillo (“node/data”), donde el código de Python después se encargaría de diferenciar a qué tabla se insertarán los datos en AWS.

Usualmente, un programa de arduino usa como recurso dos funciones, la función setup que solo se ejecuta una vez, y la función loop que se ejecuta siempre y cuando la placa esté energizada. En este caso el programa sería mucho más complicado puesto que incorpora el uso de MQTT y la conexión a la red para publicar el tópico.

En el programa de arduino se incluirán diversas funciones, una de ellas se encargará de conectar el NodeMCU a la red, por lo también se debería de descargar la librería ESP8266WiFi; es importante mencionar que cuando el NodeMCU logre conectar se a la red mostrará en el monitor serial un mensaje con la red y la dirección IP asignada por el router. Posteriormente, se llamaría a la función para conectar el dispositivo al bróker de MQTT, en donde este ciclo se ejecutará hasta que el dispositivo logre conectarse al bróker.

Habiendo ya recorrido esas dos funciones el arduino comenzaría a ejecutar infinitamente la función loop, que en su interior llama a otras funciones. La primera función en ejecutarse sería la de medir, ahí se establecerán los datos numéricos a enviar, y además se crearía y formaría el string en formato JSON que el tópico node/data enviará a MQTT. Al final solo se cargan los datos con sprintf a los buffers de los tópicos.

Posteriormente, se llamará a la función conectar mqtt, la cual lo primero que se verifica es que el dispositivo se encuentre conectado al bróker, en caso de que esto sea falso, se llamaría a la función conectar. Habiendo comprobado esto se verifica que no haya empalme de datos con un algoritmo de conteo de tiempo, luego se imprimen los valores que el documento JSON incluirá y se manda llamar el método publish de la librería de MQTT para arduino. Ya que se publica el tópico en MQTT el programa tendrá una condición en que esperará 10 minutos para volver a ejecutar la función loop.

A continuación se anexa una dirección al repositorio en GitHub de los programas desarrollados para el reto:

- https://github.com/EITrujo01/IoT_TC1004B

Conexión y Aplicación de IoT

Dentro de los requerimientos principales solicitados por el profesorado para el desarrollo de este proyecto estaría el conectar cada placa utilizada por el equipo a un servicio de broker de protocolo MQTT, que a su vez deberá ser conectado a un servidor en AWS, aunque si es necesario mencionar que primero se deberán de procesar estos datos mediante la implementación de un algoritmo en python, y el uso debido de un par de librerías relacionadas con la publicación y suscripción de los datos.

Con apoyo de los archivos temporales de tipo JSON generados por el NodeMCU y la programación designada en Python, se podrían procesar los datos encontrados en el interior de este documento temporal. Es importante mencionar que esto también se puede realizar debido a la estructura con la que se define el tópico utilizado, puesto que anteriormente, cuando se utilizaban múltiples tópicos, el programa en Python, y la inserción de los datos fallaban.

El programa de Python en sí es sencillo, y resulta bastante conveniente que se divida y se maneje con llamadas a funciones. Cabe mencionar que parte del código se encontró en el interior de un repositorio de GitHub, sin embargo, este no contaba con algún tipo de documentación, por lo que el equipo se dio a la tarea de investigar el funcionamiento de cada uno de los métodos utilizados.

Para que Python pueda conectarse de alguna manera al broker de MQTT será necesario usar alguna librería externa, en este caso se usó paho-mqtt, la cual se tuvo que añadir a la biblioteca mediante la ejecución del siguiente comando en el Símbolo de Sistema de Windows: `pip install paho-mqtt`. Para poder usar esta librería y sus métodos sería necesario crear un objeto del tipo paho-mqtt, el cual fue designado a llamarse "mqtt". Posteriormente, en el

“main” del programa se llamaría a un método llamado client, al cual variables declaradas de forma global para conectarse al bróker de HiveMQ. Luego se llamaría a la función on_connect que lograría conectar Python con la base de datos, habiendo logrado esto se espera a que llegue algún tópico a MQTT.

Para poder realizar una búsqueda o insertar datos en la base de datos en AWS se tuvo que incorporar en el diseño la librería pymysql, la cual nuevamente se instaló al correr el comando pip install PyMySQL en el Símbolo de Sistema de Windows. Al igual que anteriormente, se creó un objeto del tipo pymysql, sin definición, el cual principalmente se usará para realizar las inserciones de los datos procesados en la base de datos.

Cada vez que la función on_connect detecte la llegada de una serie de datos de parte del bróker se deberá de detectar sobre cuál de las 4 tablas se intentará insertar el set de datos, esto se realiza con apoyo de algunas variables específicas usadas justamente para diferenciar cada set. Un gran ejemplo de esto es que dentro del set de datos mandados por el NodeMCU al bróker se encuentra la variable “temperatura”, mientras que para el sensor de ultrasonido la variable específica sería “distancia”. Cuando se logra diferenciar qué set de datos es enviado, mediante una serie de if, se llama a la función específica que realizará la inserción de los datos en la tabla correcta, con apoyo del método execute de la librería de pymysql. Al recibirse la señal de que la inserción se realiza correctamente, se imprime una línea que confirma que los datos fueron guardados y se termina el proceso.

Es sumamente importante mencionar que este programa deberá estarse corriendo las 24 horas en algún equipo con conexión a internet, es por ello que se montó un dispositivo de cómputo adicional al personal, al cual se accede cada vez que sea necesario realizar modificaciones en el programa de Python mediante el uso de las funciones del Escritorio Remoto de Windows.

```
PS C:\Users\jtrujo\Documents\Arduino> python -u "c:\Users\jtrujo\Documents\Arduino\MQTT_MySQL_Publisher.py"
MQTT Client Connected
MySQL Client Connected

Transmission received from DHT11 Alondra2_2 at 2020-12-04 20:08:31.324009
data logged

Transmission received from ultrasound Alondra2_1 at 2020-12-04 20:09:08.117932
data logged

Transmission received from DHT11 TP-LinkPB_1 at 2020-12-04 20:10:01.667500
data logged

Transmission received from DHT11 Alondra2_2 at 2020-12-04 20:18:31.822941
data logged

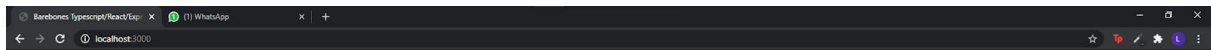
Transmission received from ultrasound Alondra2_1 at 2020-12-04 20:19:08.460277
data logged
```

En la siguiente dirección se podrá encontrar el programa de Python desarrollado, el cual se guarda en el repositorio de GitHub del reto:

- https://github.com/EITrujo01/IoT_TC1004B/blob/main/MQTT_MySQL_Publisher.py

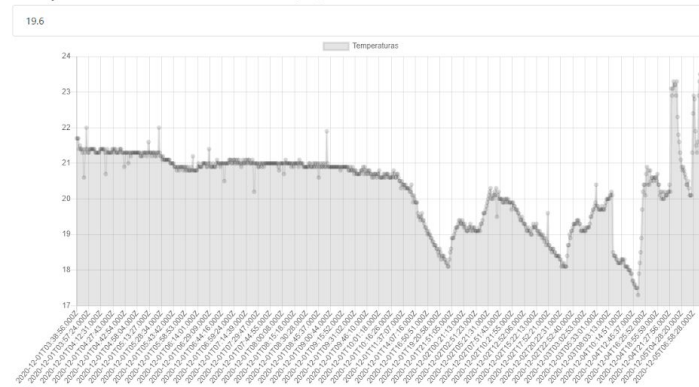
Finalmente, se desarrolló un sistema de visualización de datos a través de una aplicación Web desarrollada con el framework de React Native. Para esto, se realizó una conexión a la base de datos hosteada en Amazon Web Services para obtener los datos necesarios mediante queries ejecutados desde la aplicación.

Después de esto, usando componentes de TypeScript se creó un componente de visualización en donde se mostraban los datos de la temperatura actual, el estado de la lámpara y una gráfica de las últimas 25 temperaturas registradas. A continuación se muestra el prototipo de esta aplicación, adjuntando los ejemplos de código de los queries realizados.



Dashboard de datos:

Temperatura actual del cuarto (°C):



Lámpara:



```
export const last = async() =>{
  return new Promise((resolve,reject)=>{
    Connection.query('SELECT * FROM TempData ORDER BY DateTempStatus DESC LIMIT 1;', (err,results)=>{
      if(err){
        return reject(err);
      }
      resolve(results);
    });
  });
}

export const all = async() =>{
  return new Promise((resolve,reject)=>{
    Connection.query('SELECT * FROM TempData WHERE ID_HOUSE = "Alondra2" LIMIT 25;', (err,results)=>{
      if(err){
        return reject(err);
      }
      resolve(results);
    });
  });
}
```

```
export const last = async()=>{
  return new Promise((resolve,reject)=>{
    Connection.query('SELECT * FROM UltrasoundData ORDER BY DateDistStatus DESC LIMIT 1;', (err,results)=>{
      if(err){
        return reject(err);
      }
      resolve(results);
    });
  });
}
```

Conclusión General

Sin duda la realización de este proyecto no fue del todo sencillo. Por desgracia no todos los integrantes del equipo estaban familiarizados con los Arduinos y nunca habían trabajado con ellos. Afortunadamente aprendimos a reconocer las fortalezas de cada integrante para encontrar la funcionalidad que cada uno podía tener. Como resultado trabajamos de manera eficiente como equipo y pudimos llevar el proyecto adelante.

Este tema de IoT en verdad fue muy interesante de explorar y estamos seguros que esta no será la última vez que apliquemos conceptos sobre este tema. Además aprender a mandar información de diferentes dispositivos a una base de datos es fundamental para nuestra carrera. Consideramos que este proyecto funcionó como una buena introducción a este tema que va ser muy importante en nuestra carrera.

Como equipo, nos gustaría que esta clase de proyectos se sigan aplicando en los siguientes semestres ya que ayudan mucho a comprender cómo se da la integración de todos los temas, algo que es más difícil de ver cuando solo trabajamos con teoría.

Conclusiones Individuales

Gerardo Arrambide

En general este reto me gusto ya que logre implementar todo lo que aprendí en los diferentes módulos del bloque. Sin duda lo más complicado del reto fue la parte circuitos y la conexión de este con la base de datos. Yo nunca había trabajado con Arduino o con algo parecido por lo que esto fue una experiencia completamente nueva para mi. Afortunadamente varios integrantes de mi equipo ya estaban familiarizados con este tema y me ayudaron con todas las dificultades que se me presentaron. Al Inicio batalle bastante con la conexión del circuito por esta misma razón y aun con la ayuda de mi equipo, termine quemando todo el circuito. Por esta razón considero que este proyecto

hubiera sido mucho más efectivo y práctico si se hubiera hecho de manera presencial en el aula.

La parte de bases de datos no me pareció muy complicada ya que es un tema con el que ya había trabajado antes y entendía bastante bien. Por otro lado la conexión entre en NodeMCU y la base de datos si fue un reto. Me tuve que apoyar bastante de mis compañeros e investigaciones externas para llevar esta parte adelante y afortunadamente al final todo salió bien.

Finalmente me gustaría agradecer a los dos profesores de este bloque ya que sin su ayuda este proyecto no hubiera podido salir adelante.

Edith Benvenuto

En lo personal, este proyecto final realmente me retó a dar lo mejor de mi, ya que pude ver la aplicación de la mayoría de los conceptos vistos en clase en nuestra vida cotidiana y como realmente podrían facilitar muchas tareas realmente sencillas y reducir gastos en ciertas áreas como la energía eléctrica, en el caso de nuestro proyecto.

Lo más difícil para mi fue trabajar en equipo de manera remota, ya que cuando alguno de nosotros tenía algún problema, ya fuera con las conexiones digitales, como con las conexiones a la base de datos, era realmente complicado encontrar dicho problema entre todos.

Como mencioné en la sección de desarrollo, se me presentaron ciertos problemas en las conexiones con el NodeMCU y fue un poco complicado para mí conocer las diferencias entre los dos NodeMCU con diferente módulo de wifi que utilicé, ESP32 y el ESP8266, encontrar como conectarlo y traducir la entrada de los pines a la IDE de Arduino. También el entender el funcionamiento del MQTT y como este se conectaba a la base de datos fue algo complicado para mi.

El reto fue realmente interesante y aplicable para nuestro día a día, a pesar de que tuve algunas complicaciones y dificultades, siempre hubo mucho apoyo, tanto de parte de los profesores de ambos módulos, como de mis compañeros de equipo, todos siempre estuvieron dispuestos a resolver cualquier duda que surgiera.

Diego Torres

En lo personal, este proyecto me pareció muy divertido y me dejó mucho aprendizaje, me hubiera gustado mucho realizarlo de manera presencial pero aún así me llevo una buena experiencia.

El proyecto planteado me retó a entender y aplicar nuevos conocimientos que fui aprendiendo conforme iba trabajando. Nunca antes había manejado algún tipo de hardware o Arduino pero gracias a la ayuda de mis compañeros y profesores pude entenderle.

Tuve algunas dificultades al momento de realizar este proyecto, al principio eran fallas que se podían solucionar de manera muy sencilla pero al no tener conocimientos sobre el tema, me tardé un poco en averiguar qué era lo que estaba sucediendo. De hecho, no pude completar todo el proyecto ya que mi NodeMcu nunca se pudo conectar a Internet, sin embargo la experiencia que adquirí me ayudará mucho en futuros proyectos y creo que fue un buen comienzo.

En general me pareció sumamente interesante este reto y me gustaría que de aquí en adelante todos fueran parecidos, donde se enseñan diferentes conceptos y los podamos aplicar en un proyecto ya que en lo personal pienso que es la manera en la que mejor se aprende.

Jesús A. Trujillo

Personalmente debo decir que ya estaba algo relacionado con el mundo del internet de las cosas, puesto que cada vez mi papá integra más dispositivos

de este tipo en casa. Es muy común ver cómo se van agregando dispositivos diferentes, algunos van desde cámaras wifi hasta tomacorrientes controlados por alguna aplicación móvil, incluso he visto como los aires acondicionados de las áreas sociales se encienden solos según la temperatura de la habitación.

Cuando ví en mi plan de estudios esta materia sinceramente creí que sería un poco más diferente, de hecho creí que estaría un poco más enfocada a la parte de software que hardware y sistemas embebidos puesto que personalmente deseo conocer la parte de kubernetes. La verdad es que al principio me desanimé un poco cuando comenzamos a ver módulos de digitales, aunque poco tiempo pasó para cuando la clase comenzó a formar en mí interés.

Personalmente puedo decir que no tuve complicaciones mayores al momento de crear el circuito de Arduino en la protoboard, lo más grave que pasó fue que conecté con polaridad invertida el sensor de ultrasonido, por lo tanto este no envió correctamente los datos. En verdad lo que considero un poco más complicado, o más bien lo que tardó más tiempo fue lograr que el tópico de mqtt del arduino se enviara en formato de JSON, esto con el fin de usar parse ya sea en python o con node-red.

Gracias al desarrollo del reto pude aplicar numerables conocimientos e ideas formuladas en clase, de las cuales muchas de ellas fueron bastante abstractas, por lo que realmente terminé de comprender el concepto al ejecutar alguna acción relacionada con el reto mismo. Agradezco en gran manera la forma en la que se fue desarrollando el reto, puesto que el equipo en general de esta manera pudo esforzarse e intentar sacar lo mejor de cada integrante.

Luis Eduardo Gutiérrez

Me pareció que esta materia fue muy interesante, creo que ha sido una de las materias en la que más he aprendido a lo largo de mi carrera y creo que pude aplicar bastantes conocimientos. Creo que la solución a la que llegamos como equipo fue muy completa y cumplió con los estándares.

Personalmente estuve encargado principalmente de la conexión de los sensores acústicos y de la puerta, al igual que del desarrollo de la aplicación web para visualizar los datos. Creo que la aportación en conjunto de cada uno de los integrantes del equipo fue lo que nos hizo poder cumplir con los requisitos de este trabajo.

Por otra parte, siento que este es un primer paso hacia encontrar nuevos proyectos dentro del área del IoT, que personalmente a mi me llama mucho la atención y siento que a través de los diferentes conceptos aprendidos en clase podemos aplicar nuevas soluciones a esta área en un futuro.

En general este reto me pareció muy interesante y creo que pude aprender mucho, y uno de los aspectos que más valor agregó fue el hecho de poder emplear de manera práctica los conocimientos adquiridos para implementarlos en una solución real.

Referencias

Oracle. (2018). ¿Qué es una base de datos? Recuperado en Diciembre 01, 2020, de <https://www.oracle.com/mx/database/what-is-database/>

Naylamp Mechatronics. (2020). Sensor de temperatura y humedad relativa DHT11. Recuperado en Diciembre 01, 2020, de <https://www.naylampmechatronics.com/sensores-temperatura-y-humedad/57-sensor-de-temperatura-y-humedad-relativa-dht11.html>

Código IoT. (2019). ¿QUÉ ES EL MICROCONTROLADOR NODEMCU? Recuperado en Diciembre 01, 2020, de <https://www.codigoiot.com/que-es-el-microcontrolador-nodemcu/#:~:text=Un%20microcontrolador%20NodeMCU%20es%20un,conectarse%20a%20Internet%20vía%20WiFi.>

Naylamp Mechatronics. (2020). Sensor Ultrasonido HC-SR04. Recuperado en Diciembre 01, 2020, de https://www.naylampmechatronics.com/sensores-proximidad/10-sensor-ultrasonido-hc-sr04.html?search_query=sensor+ultrasonico

Naylamp Mechatronics. (2020, December 01). Módulo KY-037 Sensor de Sonido. Recuperado en Diciembre 01, 2020, de <https://cdmxelectronica.com/producto/modulo-ky-037-sensor-de-sonido/>

Telliottosceola. (2020, Enero 28). Mosquitto Subscriber MySQL Publisher. GitHub. Recuperado el 15 de noviembre de 2020 de https://github.com/ncd-io/Mosquitto_Subscriber_MySQL_Publisher