

Proyecto Final Inel 4206-020: Temperatura y Capacidad

Edgar Suarez

Universidad de Puerto Rico Recinto de Mayagüez, Departamento de Ingeniería Eléctrica y Computación
Arroyo, Puerto Rico
edgar.suarez1@upr.edu

Carlos Curet

Universidad de Puerto Rico Recinto de Mayagüez, Departamento de Ingeniería Eléctrica y Computación
Loíza, Puerto Rico
carlos.curet2@upr.edu

Rubén E. Torres González

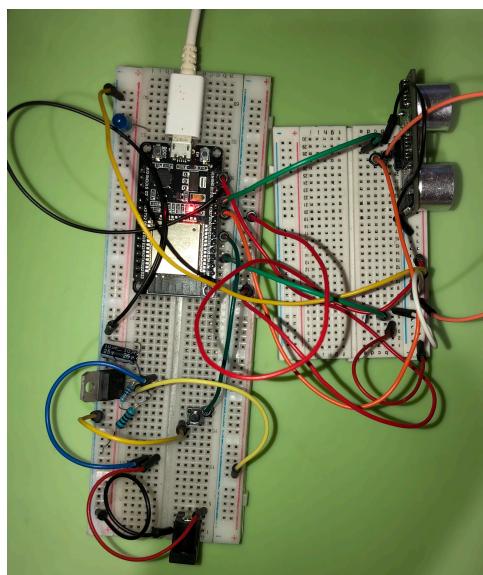
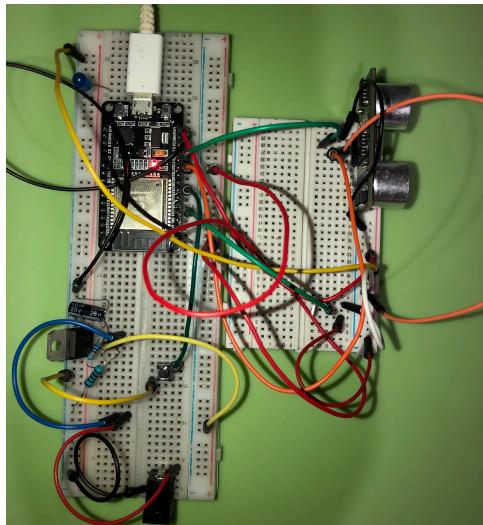
Universidad de Puerto Rico Recinto de Mayagüez, Departamento de Ingeniería Eléctrica y Computación
Adjuntas, Puerto Rico
ruben.torres9@upr.edu

Tabla de Contenido:

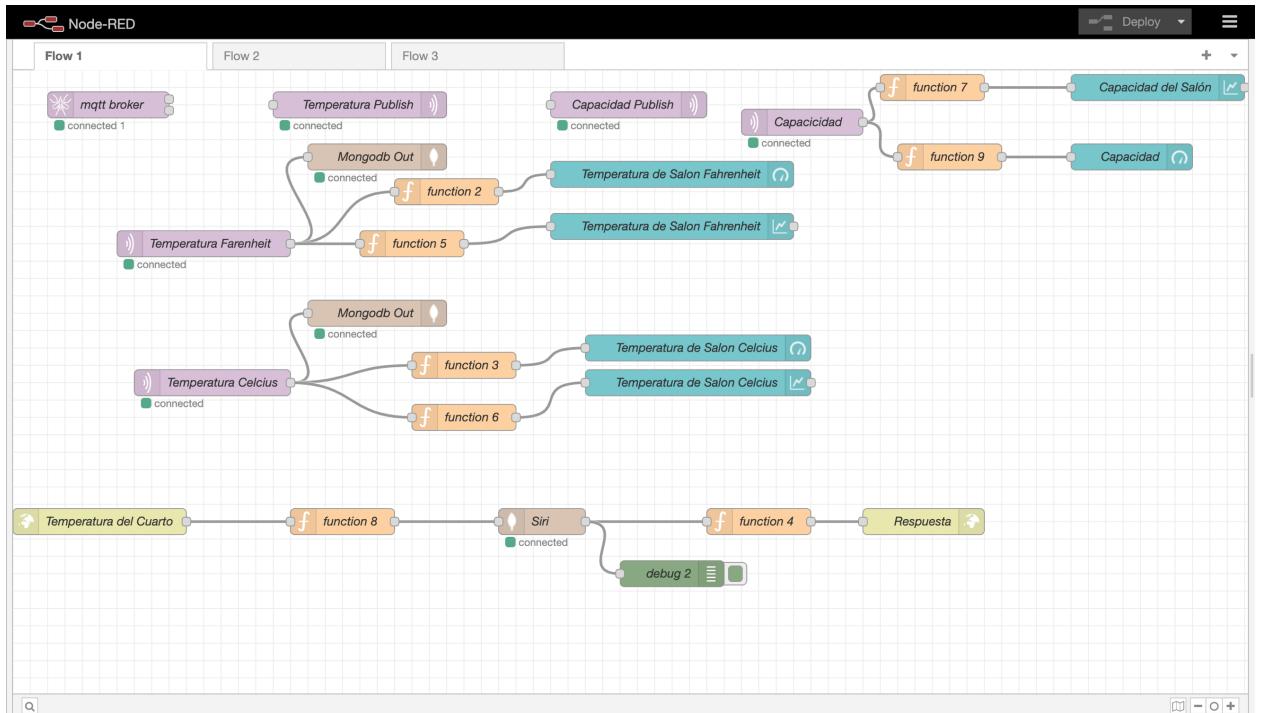
1. Architecture and Design	3 - 5
2. Bill of Materials	5 - 6
3. Project Plan	6
4. Tool Set List	6
5. Readme File	7 - 8
6. Aws	8 - 9
7. Project Code	9 - 12
8. Shortcut on Iphone	13

Architecture and Design:

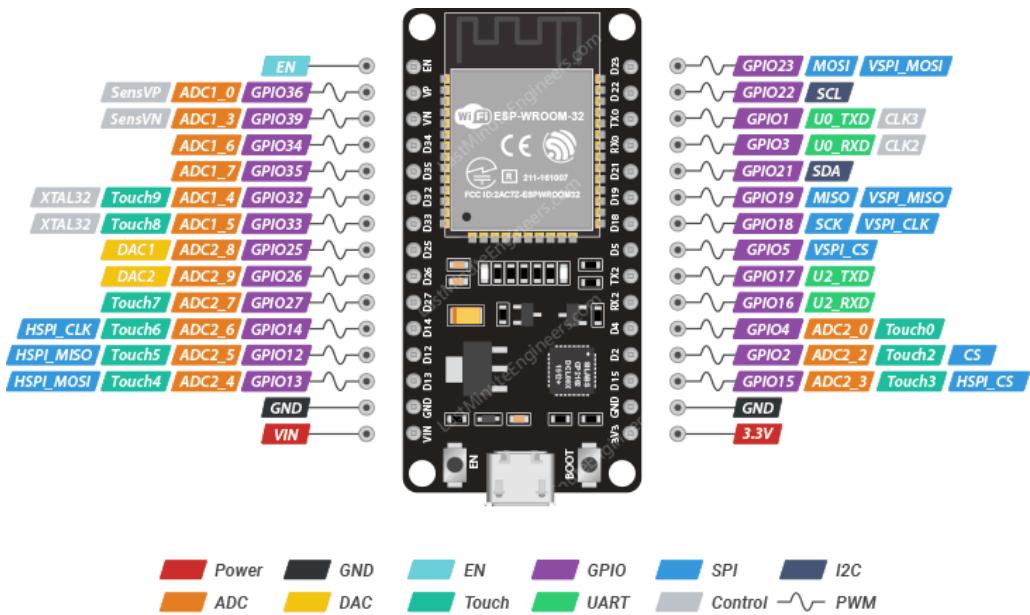
- ESP32 Connections



- Flow-Diagram utilizado en Node-Red:



- Esp32 Dev. Board Pinot:



ESP32 Dev. Board / Pinout

Last Minute
ENGINEERS.com

- Whois.com:

- Studio 3T (MongoDB):
 - 1)

2)

Bill of Materials:

ESP8266 – Proyecto Final Temperatura y Capacidad				
	Nombre de la Pieza	Descripción	Cantidad	Precio
1	ESP32	Microcontrolador	1	\$7.99
2	Breadboard	Utilizado para crear el circuito	1	\$4.50
3	Micro USB-Cable Data	Cable micro usb utilizado para transportar data y “power”	1	\$7.00
4	LMT84	Sensor de temperatura	1	\$1.04
5	HC-SR04	Sensor de movimiento	1	\$4.50
6	Jumper Cables	Cables utilizados para conectar el circuito	100	\$7.39

Project Plan:

Title	Assignees	Status
1 Conectarse a IOS ó Android #6	CarlosCuret1, ed...	Done
2 Terminar la Documentación del Proyecto #14	CarlosCuret1, ed...	Done
3 Terminar el código #15	CarlosCuret1, ed...	Done
4 Set Up AWS #2	CarlosCuret1, ed...	Done
5 Comenzar a Codificar #10	CarlosCuret1, ed...	Done
6 Set Up Node-Red #3	CarlosCuret1, ed...	Done
7 Bajar y actualizar todos los paquetes de Linux #8	CarlosCuret1, ed...	Done
8 Set Up Github #4	CarlosCuret1, ed...	Done
9 Draft 1 of AC On Architecture/Design Doc #1	CarlosCuret1, ed...	Done
10 Design Through Node-Red #5	CarlosCuret1, ed...	Done
11 Finalize Cloud Side #7	CarlosCuret1, ed...	Done
12 Buscar y comprar sensor de temperatura y movimiento #9	CarlosCuret1, ed...	Done
13 FSM Doc #11	CarlosCuret1, ed...	Done
14 Provisioning and Configuration Doc #12	CarlosCuret1, ed...	Done

Tool Set List:

- VS-Code
- Node-Red
- AWS
- Puttygen

- Github
- Studio 3T
- MongoDB
- MQTT Broker
- Arduino IDE
- Google Meet
- Google Docs
- Canva
- Microsoft Word
- Siri

Readme File:

Título del Proyecto: ¿Cuál es la Temperatura y Capacidad?

Colaboradores: Carlos Curet, Edgar Suarez, Rubén Torres

Descripción del Proyecto:

Durante este proyecto, nuestro objetivo es poder conocer la temperatura y cantidad de personas en el salón que sea elegido. Esto se logrará a través de un ESP-32 con sensores de temperatura y proximidad. Toda la data recopilada por estos sensores será enviada al ESP-32. El cual estará conectado a un servidor o interface (Ubuntu). De esta forma, como la información va a estar en el servidor los usuarios podrán preguntarle a Siri o a Google (dependiendo de su formato operativo) cual es la temperatura del salón y su capacidad.

Este proyecto se llevará a cabo utilizando herramientas como Github, Node-Red, VS-code, Powershell, Puttygen, Linux, MongoDB, Studio 3T, AWS. Muchas otras como google meets o google docs, para poder mantener una comunicación entre el equipo. Todas estas herramientas tienen su propósito durante el proyecto. Comenzando por, AWS, el cual nos permite crear un servidor. Ya que, necesitamos enviar la información a este servidor para que este se la envie a nuestros teléfonos en tiempo real. Este servidor creado en aws, se trabajará desdennuestras computadoras. Esto se logra conectando nuestras computadoras al servidor a través de puttygen y las claves abiertas o "open ssh keys". Luego, powershell nos permite descargar las herramientas que utilizaremos como Node-Red y MongoDB. Esta herramienta conocida como Node-Red nos permite configurar de una manera sencilla nuestra página web donde se verán los datos de la temperatura. Además, MongoDB nos permite guardar los datos recopilados y estos se pueden ver en Studio 3T.

Todo se podrá programar y hacer posible utilizando las herramientas de organización que github nos ofrece para proyectos en grupos como este. Donde lo más útil es poder diluir el trabajo entre los miembros del grupo. Para de esta forma poder trabajar en diferentes áreas de manera simultánea.

Copyright (c) [2023] [Carlos Curet, Edgar Suarez, Rubén Torres]

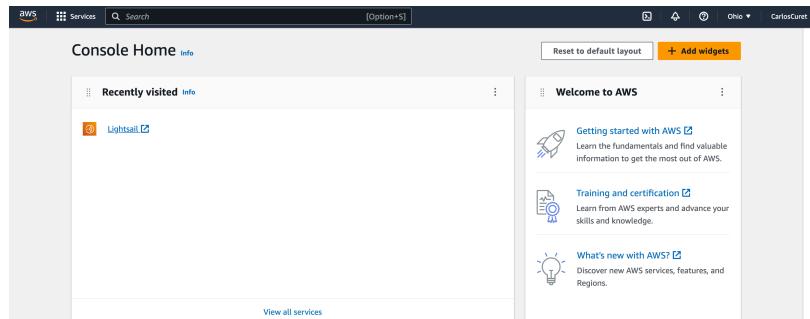
Por la presente se concede permiso, sin cargo, a cualquier persona que obtenga una copia de este software y los archivos de documentación asociados, para tratar en el software sin restricciones, incluidos sin limitarse a, los derechos de usar, copiar, modificar, fusionar, publicar, distribuir, sublicenciar y/o vender copias del software y permitir a las personas a quienes se les a provisto para hacerlo, sujeto a las siguientes condiciones:

El aviso de derechos de autor anterior y este aviso de permiso se incluirán en todos copias o partes sustanciales del Software.

EL SOFTWARE SE PROPORCIONA "TAL CUAL", SIN GARANTÍA DE NINGÚN TIPO, EXPRESA O IMPLÍCITO, INCLUYENDO, PERO NO LIMITADO A LAS GARANTÍAS DE COMERCIABILIDAD PARA UN PROPÓSITO PARTICULAR Y DE NO VIOLACIÓN. EN NINGÚN CASO LOS AUTORES O TITULARES DE LOS DERECHOS DE AUTOR SERÁN RESPONSABLES DE CUALQUIER RECLAMACIÓN, DAÑOS U OTRAS RESPONSABILIDADES, YA SEA EN UNA ACCIÓN DE CONTRATO, AGRAVIO O DE OTRA FORMA DERIVADA DE, FUERA DE O EN CONEXIÓN CON EL SOFTWARE O EL USO U OTROS TRATOS EN EL SOFTWARE.

AWS:

1)



2)

The screenshot shows the Amazon Lightsail home page. At the top, there's a banner for "Introducing Amazon Lightsail for Research". Below the banner, there are two instance cards:

- Ubuntu-2**: Running, 1 GB RAM, 1 vCPU, 40 GB SSD. IP: 44.196.27.157. Zone: Virginia.
- Ubuntu-1**: Running, 1 GB RAM, 1 vCPU, 40 GB SSD. IP: 54.167.93.83. Zone: Virginia.

Below the instances, there's a "Good evening!" message and a search bar.

3)

The screenshot shows the "Connect to your instance" page for instance Ubuntu-1. It provides instructions for connecting using a browser or an SSH client.

Use your browser: Connect using our browser-based SSH client. There's a "Connect using SSH" button.

Use your own SSH client: Connect using an SSH client. There's a "Download default key" link.

Details for the instance:

- CONNECT TO**: **44.196.27.137**
- IPv4: 2600:1f18:774d:4b00:b492:6a46:371af88b
- USER NAME: **ubuntu**
- SSH KEY: Download default key

4)

The screenshot shows the "IPv4 networking" section of the Amazon Lightsail interface. It displays the public and private IP addresses of the instance.

PUBLIC IP: **44.196.27.137** (Static IP)

PRIVATE IP: **172.26.6.13**

A note states: "Your instance is using a static IP as its public IPv4 address. A static IP doesn't change when you stop and start your instance."

IPv4 Firewall: Create rules to open ports to the internet, or to a specific IPv4 address or range. There's a "Learn more about firewall rules" link.

+ Add rule			
Application	Protocol	Port or range / Code	Restricted to
SSH	TCP	22	Any IPv4 address Lightsail browser SSH/RDP
HTTP	TCP	80	Any IPv4 address
HTTPS	TCP	443	Any IPv4 address
Custom	TCP	1880 → 1885	Any IPv4 address

Project Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
#include <string.h>
#include <Arduino.h>
#include <stdio.h>
#include "driver/adc.h"

#define LMT84_PIN ADC1_CHANNEL_4 // GPIO 32
#define trigPin GPIO_NUM_27
#define echoPin GPIO_NUM_14
#define interruptButton GPIO_NUM_4
#define wifiLed GPIO_NUM_13
#define buttonLed GPIO_NUM_12

const char *ssid = "Eduardo";
const char *password = "1234512345";

// const char *ssid = "IPhone";
// const char *password = "12345678";

const char *mqtt_server = "masbarato.space";

WiFiClient espClient;

PubSubClient client(espClient);

long duration;
int distance;
int counter = 0;
unsigned long lastDebounceTime = 0;
unsigned long debounceDelay = 250;

void setup()
{
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(interruptButton, INPUT);
    pinMode(wifiLed, OUTPUT);
    pinMode(buttonLed, OUTPUT);

    digitalWrite(interruptButton, LOW);

    Serial.begin(115200);
    // Connect to Wi-Fi network
    WiFi.begin("LIB-0762397", "j8kcXyvWcb2x");
```

```

while (WiFi.status() != WL_CONNECTED)
{
    delay(1000);
    Serial.println("Connecting to WiFi...");
    gpio_set_level(wifiLed, 0);
}

Serial.println("Connected to WiFi");

// Connect to MQTT broker
client.setServer(mqtt_server, 1883);

while (!client.connected())
{
    Serial.println("Connecting to MQTT broker...");
    if (client.connect("ESP32Client"))
    {
        Serial.println("Connected to MQTT broker");
    }
    else
    {
        Serial.print("Failed to connect to MQTT broker, rc=");
        Serial.print(client.state());
        Serial.println(" retrying in 5 seconds");
        delay(5000);
    }
}
}

void loop()
{

    if (WiFi.status() == WL_CONNECTED)
    {

        gpio_set_level(wifiLed, 1);
    }
    else
    {
        gpio_set_level(wifiLed, 0);
    }

    uint16_t rawValue = adc1_get_raw(LMT84_PIN);

    float voltage = rawValue * (3.3 / 4095.0);

    float temperature = (voltage - 1.25) / 0.05;
}

```

```

Serial.print("Temperature: ");
Serial.print(temperature);
Serial.println(" degrees Celsius");

char temperatureStr[10];
snprintf(temperatureStr, 10, "%.2f", temperature);
client.publish("Temperatura", temperatureStr);

digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance = duration * 0.034 / 2;
Serial.print("Distance: ");
Serial.println(distance);

if (distance <= 150)
{ // 183
  Serial.println("Movement detected!");
  counter++;
}
int old_counter = counter;

int buttonState = digitalRead(interruptButton);
if (buttonState == HIGH && (millis() - lastDebounceTime) > debounceDelay)
{
  counter--;
  Serial.println("Button pressed! Count = " + String(counter));
  lastDebounceTime = millis();
}
Serial.print("Counter: ");
Serial.println(counter);

char distanceStr[10];
snprintf(distanceStr, 10, "%d", old_counter);
client.publish("Capacidad", distanceStr);

snprintf(distanceStr, 10, "%d", counter);
client.publish("Capacidad", distanceStr);

delay(1000);
digitalWrite(interruptButton, LOW);
}

```

Shortcut on Iphone:

