

1. INFORMACIÓN GENERAL DE LA INVESTIGACIÓN

- 1.1. Semillero de Investigación: ONTARE
- 1.2. Tutor del semillero: Ph.D Luis Armando Cobo
- 1.3. Nombre del estudiante: Oscar David Jimenez Bonilla
- 1.4. Título Proyecto: PreventLink
- 1.5. Periodo: 2020-2
- 1.6. Fecha de presentación: 12 – Dic – 2020

2. DESARROLLO DEL INFORME DE INVESTIGACIÓN

- 2.1. Objetivo de la actividad (utilice entre 60 y 120 palabras):

Desarrollar un prototipo de un EPP (Elemento de Protección Personal) basado en tecnología Arduino que permita escrutar la información que proviene de este elemento y controlar los dispositivos que hacen parte del dispositivo.

- 2.2. Descripción detallada de las actividades realizadas (se recomienda describir las actividades realizadas por cada objetivo específico que se planteó en el proyecto):

Para este proyecto se parte de la idea de los elementos de protección personal por lo cual se requiere desarrollar un prototipo de un elemento de protección personal, en este caso se escogió un tapabocas. Este dispositivo tendrá un conjunto de sensores (temperatura y humedad), tres leds y un motor conectados y se podrá controlar y vigilar desde cualquier computador utilizando una aplicación. Esta aplicación recibe y transfiere datos a través de una conexión wifi gracias un módulo basado en Arduino montado en el tapabocas, y recibimos la información captada por los sensores y controlamos los elementos conectados al tapabocas, en este caso los leds y el motor, de forma remota.

Actividades

- 1. Investigación:

Para comenzar el proyecto, se inició con una investigación acerca de los mecanismos de comunicación en red a través de Sockets/ip, siguiendo las instrucciones del tutor. A partir de esta investigación se logró hacer un programa en una raspberry pi que controlara unos leds que éste poseía, pero el inconveniente que se tuvo fue que el dispositivo era muy caro y no se estaba controlando estos elementos de una forma remota. Luego de la investigación, se llegue a la parte donde se decidió trabajar con Arduino y el módulo de wifi ESP 8266. A partir de allí se empezó una investigación sobre como programar en python y C, aplicaciones de comunicación y de esta forma lograr conectar estos un PC con el Arduino a través de una red wifi.

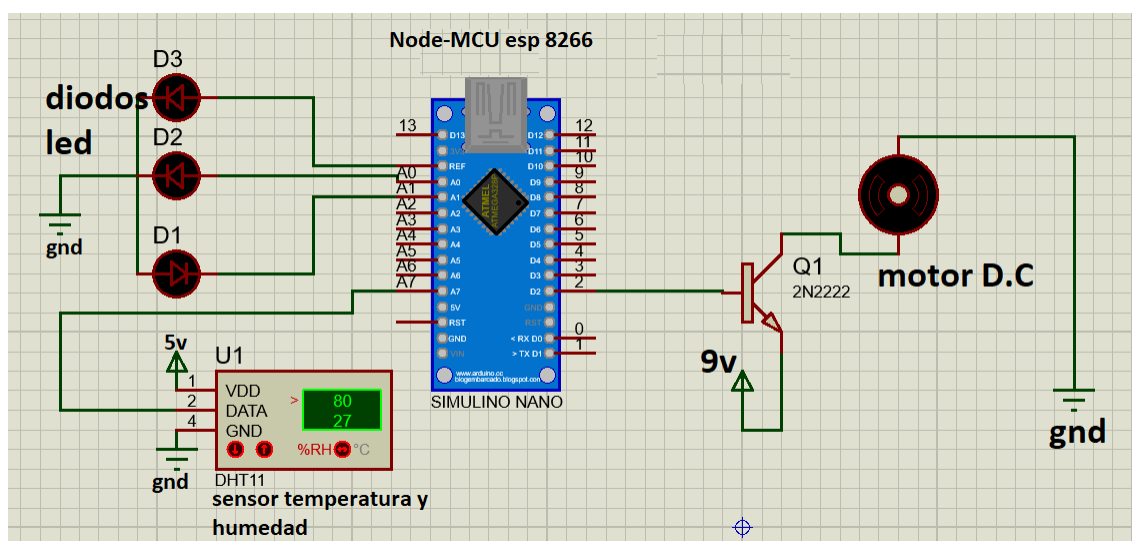
Se utilizaron proyectos de nuestro país, consultando fuentes como Google Scholar, y se encontraron proyectos de grados en los cuales utilizaban la tecnología que se quería implantar, además encontramos en YouTube tutoriales interesante, y logramos aprender las nociones básicas sobre lo que es la ip y los sockets web que son las nuevas tecnologías usadas actualmente para lograr la comunicación entre equipos en una red.

2. Diseño del circuito:

Para el diseño del circuito, se indagó en muchos módulos y tarjetas que existen en el mercado. Al final nos decidimos por el módulo ESP8266 para este proyecto, por sus características y precio. Después de varios ensayos, se descubrió que trabajar con una Arduino se hacía muy complejo, así que se realizó una búsqueda de opciones a este dispositivo y encontramos un módulo llamado NODEMCU, el cual está basado en Arduino, se programa también desde el IDE de Arduino y logra hacer las tareas que se necesitaban de una forma más sencilla. Luego se agregaron un sensor de temperatura y otro de humedad, los leds y finalmente se utilizó un transistor 2N222 para dar paso a la corriente que alimenta el motor que hace parte del EPP.

El diagrama final del circuito realizado se presenta en la Figura 1. A continuación se presenta una descripción de los elementos utilizados en él.

Figura 1. Diagrama del Circuito



Fuente: Elaboración por parte del autor.

Materiales utilizados:

- Node MCU:

El NodeMcu es un kit de desarrollo de código abierto basado en el popular chip ESP8266 (ESP-12E), que utiliza el lenguaje de programación Lua para crear un ambiente de desarrollo propicio para aplicaciones que requiera conectividad Wifi de manera rápida. Es un chip altamente integrado diseñado para las necesidades de un nuevo mundo conectado. Ofrece una solución completa y autónoma de redes Wi-Fi, lo que le permite alojar la aplicación o servir como puente entre Internet y un microcontrolador.



l placa node mcu tomado de <https://programarfacil.com/podcast/nodemcu-tutorial-paso-a-paso/>

- Leds:

Un diodo emisor de luz o led5n 1 (también conocido por la sigla LED, del inglés light-emitting diode) es una fuente de luz constituida por un material semiconductor dotado de dos terminales. Se trata de un diodo de unión p-n, que emite luz cuando está activado.



II diodos leds tomado de: <https://sites.google.com/site/electronicacompleta29/diodo-led-light-emitting-diode>

- DHT 11:

El DHT11 es un sensor de humedad relativa y temperatura de bajo costo y de media precisión a un bajo precio. La salida suministrada es de tipo digital utilizando solamente 1 pin de datos.



III sensor humedad y temperatura dht11 tomado de <https://naylampmechatronics.com/sensores-temperatura-y-humedad/57-sensor-de-temperatura-y-humedad-relativa-dht11.html>

- Motor D.C:

El motor de corriente continua, denominado también motor de corriente directa, motor CC o motor DC (por las iniciales en inglés direct current), es una máquina que convierte energía eléctrica en mecánica, provocando un movimiento rotatorio, gracias a la acción de un campo magnético.



IV motor dc tomado de: https://es.wikipedia.org/wiki/Motor_de_corriente_continua

- Transistor 2N2222:

Es un transistor de silicio de mediana potencia con una polaridad npn. Transistor 2N2222. Es un transistor de silicio y baja potencia, diseñado para aplicaciones de amplificación lineal y conmutación. En este caso lo utilizamos para dejar fluir la corriente que va al motor



V transistor tomado de <https://www.bigtronica.com/centro/semiconductores/transistores/1855-transistor-npn-2n2222-5053212018559.html>

Finalmente, para el circuito se conectaron los anteriores elemento en los siguientes puertos del módulo NODEMCU

D0= motor

D1=led

D2=led

D4= sensor

- Presupuesto para la construcción del circuito:

ELEMENTO	PRECIO
Node mcu	\$25.000 COP
Leds x3	\$900 COP
Motor d.c	\$5.000 COP
DHT 11	\$15.000 COP
2N2222	\$200 COP
total	\$46.100

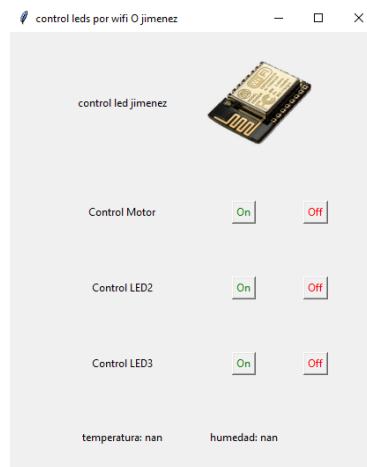
3. Desarrollo del código:

Para el desarrollo del código que soporta el circuito anterior, se realizó una investigación exhaustiva, ya que se encontró que la mayoría de los tutoriales que se encontraban en Internet, conectan el módulo NODEMCU a través de PHP a una página desarrollada en java o HTML pero nunca utilizan Python como lenguaje de comunicación con la tarjeta, el cual era uno de los requerimientos de la investigación a la cual pertenece este proyecto.

Para comenzar se utilizó el Arduino IDE para programar la tarjeta. El código correspondiente se anexa al final de este documento. Para llevar a cabo la comunicación del módulo o tarjeta con otra aplicación ejecutándose en un PC, se conectó el módulo a wifi al módulo, se estableció un puerto por el cual se va a enviar y recibir datos. Debido a que cuando se ejecuta el código, la ip

del dispositivo cambia, entonces utilizando el puerto serial del módulo, se imprime la dirección ip del módulo, así como el puerto por el que se va a transmitir y recibir los datos. Estos datos ingresan al programa escrito en Python y que se ejecuta en la PC conectada a la misma red del módulo. También se definieron funciones para leer los datos que envía el PC. Estos datos representan una tarea que se necesita sea realizada por el módulo, como captar la temperatura o encender un led.

En el programa en Python que se ejecuta en la PC se utilizó una librería llamada Thread, con la cual podemos crear un objeto socket ip identificado con un número de puerto y a través del cual se logra enviar y recibir datos, en este caso información numérica que representan órdenes para el dispositivo remoto, que luego son interpretados por el código escrito en C que se ejecuta en el módulo remoto. Finalmente, se utilizó TKinter para la interfaz gráfica de esta aplicación. Se adjunta código al final del documento.



VI interfaz Python

Para la construcción del código se dificultó mucho la parte de C dado que no sabía programar en este lenguaje, por lo que se utilizó varios tutoriales guía para realizarlos, cabe destacar que mi código de Python usa la versión 3 de este lenguaje.

4. Prototipo.

Una vez funcionando el circuito y el código de Python se procedió a diseñar y desarrollar el prototipo final. Para esto se ensambló en un tapabocas el sensor, el módulo y los leds en la parte interna del tapabocas para lograr un prototipo parecido al dado en el inicio (PreventLink)



VII ensamble de prototipo

Dado que todos estos dispositivos de protección personal deben seguir una norma y una reglamentación, se establece que este es solo un prototipo y no se conocen consecuencias de trabajar con módulos tan cerca de la cabeza, así como de su regulación a nivel nacional.

Finalmente, el prototipo ensamblado, el cable que sale es la salida al motor y a la batería pues es de alto riesgo utilizar baterías en la cara.

5. Bibliografía:

<https://programarfacil.com/podcast/nodemcu-tutorial-paso-a-paso/>

<https://agelectronica.blog/category/iot/>

<https://www.youtube.com/watch?v=lsPnZzrrCtg&t=113s>

<https://incyt.upse.edu.ec/ciencia/revistas/index.php/rctu/article/view/490/458>

<https://www.youtube.com/channel/UCqxBxJnwt2JTWPm2c0kx0Yw>

<https://www.youtube.com/watch?v=zQY6Pfpw8>

<https://www.hubor-proteus.com/proteus-pcb/dise%C3%B1ador-gr%C3%A1fico-de-programaci%C3%B3n-de-equipos-arduino/330-el-taller-iot-de-proteus-para-raspberry-pi.html>

<https://programarfacil.com/podcast/nodemcu-tutorial-paso-a-paso/>

<http://www.skulltrap.co/2019/09/lot-con-arduino-nodemcu-esp8266.html>

https://www.youtube.com/results?search_query=controlar+nodemcu+desde+pagina+php

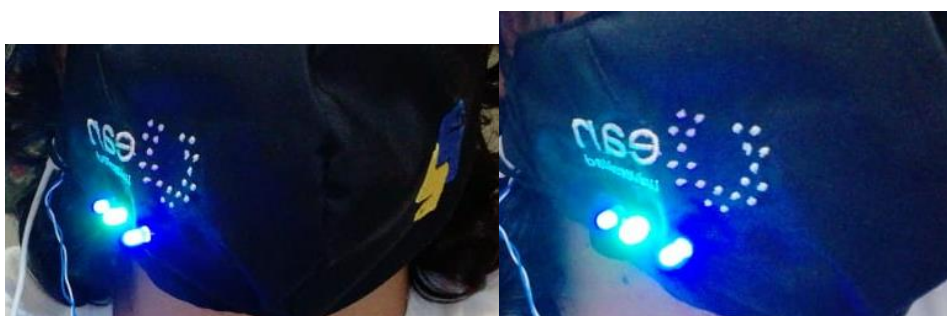
<https://parzibyte.me/blog/2020/02/11/encender-apagar-led-wifi-nodemcu-esp8266/>

<https://www.esploradores.com/practica-4-station-conexion-desde-cualquier-sitio-del-mundo-con-access-point-encendido-y-apagado-de-un-led/>

<https://electronilab.co/tienda/nodemcu-board-de-desarrollo-con-esp8266-wifi-y-lua/>

<https://github.com/freedomwebtech/Raspberry-pi-socket4>

	FORMATO	Versión: 2
		Código: INV-103-f5
INFORME FINAL SEMILLEROS DE INVESTIGACIÓN		Fecha: 30/Jun/2020



	FORMATO	Versión: 2
		Código: INV-103-f5
INFORME FINAL SEMILLEROS DE INVESTIGACIÓN		Fecha: 30/Jun/2020

- 2.3. Productos derivados del proyecto de investigación. Debe anexar las evidencias de acuerdo con la siguiente información:

Durante este proyecto se desarrollaron diversos prototipos: uno en raspberry PI, otro en Arduino y finalmente se tomó el desarrollado en el NODEMCU, por su facilidad, tamaño, costo y prestaciones para así presentar nuestro prototipo v0.1 de PreventLink. Este consta de un procesador de 244hz, un modulo wifi esp8266, 3 leds, sensor de humedad y temperatura y salida para un motor dc entre 5-12v


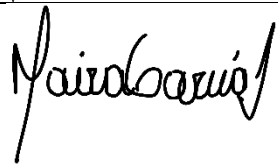


VIII producto final

	FORMATO	Versión: 2
		Código: INV-103-f5
INFORME FINAL SEMILLEROS DE INVESTIGACIÓN		Fecha: 30/Jun/2020

2.3. Realice la descripción de lo que significó para usted el desarrollo de la experiencia en investigación y lo que le aportó (utilice entre 60 y 120 palabras):

Colombia al ser un país tan poco desarrollado posee pocas posibilidades de investigación de innovación y los estudiantes pocas veces nos vemos motivados por la investigación y el aprendizaje, esto que hice considero son los pasos que debemos seguir para encontrar amor a la ciencia, a su aplicación y a lograr un país desarrollado e industrial, me sentí demasiado contento pues durante la coyuntura pude dedicar tiempo y construir un prototipo real de lo que buscábamos con el proyecto, me siento muy feliz al ver que funciona como lo esperábamos luego de muchas horas de trabajo, aprendí demasiado y me siento muy motivado por el aprendizaje autodidacta y la investigación porque nos facilita la vida, nos muestra tecnologías aplicables a industrias colombianas y desarrolla mi capacidad de aprender y aplicar conocimientos

Firma estudiante	Oscar David Jimenez cc1015478830
Vo.Bo. TUTOR	
Vo.Bo. DIRECTOR	

	FORMATO	Versión: 2
		Código: INV-103-f5
INFORME FINAL SEMILLEROS DE INVESTIGACIÓN		Fecha: 30/Jun/2020

Anexos: códigos

- ARDUINO:

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include "DHT.h"
#include <Thread.h>
#include <ThreadController.h>
Thread hiloDHT=Thread();
const char* ssid = "familiabonilla";
const char* password = "Lina2014Queen2015Radio2016";
WiFiServer servidorTCP(8266);
WiFiClient clienteTCP;
#define LED D0
#define LED1 D1
#define LED2 D2
#define DHTPIN D4 // aca conecto el sensor
#define DHTTYPE DHT11
#define TIEMPO_SAMPLEO 5000
DHT dht (DHTPIN, DHTTYPE);
ThreadController controladorHilos = ThreadController();
void leeDHT() {
  float hum=dht.readHumidity();
  float temp=dht.readTemperature();
  String datos="temperatura: ";
  datos += temp;
  datos += "\nhumedad: ";
  datos += hum;
  datos += "\n";
  Serial.print(datos);
  clienteTCP.print(datos);
}

void setup() {
  pinMode(LED,OUTPUT);
  pinMode(LED1,OUTPUT);
  pinMode(LED2,OUTPUT);
```

	FORMATO	Versión: 2
		Código: INV-103-f5
INFORME FINAL SEMILLEROS DE INVESTIGACIÓN		Fecha: 30/Jun/2020

```

Serial.begin(115200);
delay(100);
dht.begin();

Serial.print("Conectandose a: ");
Serial.println(ssid);

WiFi.begin(ssid, password); //Intentamos conectarnos a la red Wifi

while (WiFi.status() != WL_CONNECTED) { //Esperamos hasta que se conecte.
  delay(200);
}

Serial.print ("Conectado, IP: ");
Serial.println (WiFi.localIP());

servidorTCP.begin();
hiloDHT.onRun(leeDHT);
hiloDHT.setInterval(TIEMPO_SAMPLEO);

}

void loop() {

  if (!clienteTCP.connected()) {
    // try to connect to a new client
    clienteTCP = servidorTCP.available();
    if (clienteTCP.connected())
      Serial.println("cliente conectado");
    controladorHilos.add(&hiloDHT);
  }
  else {
    // read data from the connected client
    if (clienteTCP.available() > 0) {
      char dato = clienteTCP.read();
      Serial.write(dato);
      if (dato == '1'){ // led que esta en D0
        digitalWrite(LED, HIGH);
        Serial.write(dato);
      }
      else if (dato == '0'){

        digitalWrite(LED, LOW);
        Serial.write(dato);
      }
    }
  }
}

```

 ean universidad	FORMATO	Versión: 2
		Código: INV-103-f5
INFORME FINAL SEMILLEROS DE INVESTIGACIÓN		Fecha: 30/Jun/2020

```

    }
    if (dato == '2'){ // aca para el led 1 que esta en d1
      digitalWrite(LED1, HIGH);
      Serial.write(dato);
    }
    else if (dato == '3'){

      digitalWrite(LED1, LOW);
      Serial.write(dato);

    }
    if (dato == '4'){ // aca va el del led2 que esta en D1
      digitalWrite(LED2, HIGH);
      Serial.write(dato);
    }
    else if (dato == '5'){

      digitalWrite(LED2, LOW);
      Serial.write(dato);

    }

  }

}
controladorHilos.run();
}
}

```

- CODIGO PYTHON:

```

from tkinter import * #Importamos tkinter para nuestra interfáz
gráfica
import socket #Importamos la librería socket para poder comunicarnos
con nuestro ESP8266
import threading # lib multihilo
import tkinter as tk
# espacios
padX=20
padY=30

ESP_IP = '192.168.1.7' #IP de nuestro modulo

```

 ean universidad	FORMATO	Versión: 2
		Código: INV-103-f5
INFORME FINAL SEMILLEROS DE INVESTIGACIÓN		Fecha: 30/Jun/2020

```

ESP_PORT = 8266#Puerto que hemos configurado para que abra el ESP

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #objeto socket
s.connect((ESP_IP , ESP_PORT)) #Nconectarnos
root=Tk() # ventana
root.title("control leds por wifi O jimenez ") #Ctitutlo
tempVar=StringVar()
humVar=StringVar()
datos= {'temperatura:': tempVar,'humedad:': humVar}# diccionario
frame = Frame(root) # ventanita

lbl_titulo = Label(frame, text="control led jimenez") # titulito
imagenESP = PhotoImage(file="esp8266.png") #subir imagen
lbl_imagen = Label(frame, image=imagenESP) #poner img

lbl_titulo.grid(row=0, column=0, pady=padY,padx=padX) #
lbl_imagen.grid (row=0, column=1,columnspan=2,pady=padY,padx=padX) #

lbl_LEDControl = Label (frame, text="Control Motor")
lbl_LEDControl.grid (row=1, column=0)
lbl_LEDControl1 = Label (frame, text="Control LED2")
lbl_LEDControl1.grid (row=3, column=0)
lbl_LEDControl2 = Label (frame, text="Control LED3")
lbl_LEDControl2.grid (row=5, column=0)

# funciones de encender
def enciendeLED(): #Función para encender el LED
    print("Encendiendo LED")
    dato = '1'
    s.send(dato.encode(encoding='utf_8')) #enviamos 1 al modulo

def apagaLED():
    print("Apagando LED")
    dato = '0'
    s.send(dato.encode(encoding='utf_8'))

def enciendeLED2():
    print("Encendiendo LED")
    dato = '2'
    s.send(dato.encode(encoding='utf_8'))

def apagaLED2():
    print("Apagando LED")
    dato = '3'
    s.send(dato.encode(encoding='utf_8'))

def enciendeLED3():
    print("Encendiendo LED")
    dato = '4'
    s.send(dato.encode(encoding='utf_8'))

```

	FORMATO	Versión: 2
		Código: INV-103-f5
INFORME FINAL SEMILLEROS DE INVESTIGACIÓN		Fecha: 30/Jun/2020

```

def apagaLED3():
    print("Apagando LED")
    dato = '5'
    s.send(dato.encode(encoding='utf_8'))

btn_LEDOn = Button(frame, text="On", fg="green", command=enciendeLED)
btn_LEDOff = Button(frame, text="Off", fg="red", command=apagaLED)

btn_LEDOn2 = Button(frame, text="On", fg="green",
    command=enciendeLED2)
btn_LEDOff2 = Button(frame, text="Off", fg="red", command=apagaLED2)

btn_LEDOn3 = Button(frame, text="On", fg="green", command=enciendeLED3)
btn_LEDOff3 = Button(frame, text="Off", fg="red", command=apagaLED3)

btn_LEDOn.grid (row=1, column=1,pady=padY)
btn_LEDOff.grid (row=1, column=2,pady=padY)

btn_LEDOn2.grid (row=3, column=1,pady=padY)
btn_LEDOff2.grid (row=3, column=2,pady=padY)

btn_LEDOn3.grid (row=5, column=1,pady=padY)
btn_LEDOff3.grid (row=5, column=2,pady=padY)

lbl_temp=tk.Label(frame, textvariable=tempVar) ## etiqueta temp
lbl_hum=tk.Label(frame, textvariable=humVar)
lbl_temp.grid(row=11,column=0, padx=padX,pady=padY)
lbl_hum.grid(row=11,column=1, padx=padX,pady=padY)

def recibedatos():
    while True:
        cadena=s.recv(1023)
        print(cadena)
        lineas=cadena.splitlines()
        for linea in lineas:
            dato=linea.split()
            datos[dato[0].decode()].set(dato[0].decode()+"
"+dato[1].decode())
hiloRecepcion=threading.Thread(target=recibedatos)
hiloRecepcion.start()
frame.pack()
root.mainloop()

```