**PRACTICA N°1 – 4to HITO**

**CONSIGNA:**

Realizar un sistema domótico para el control de luces desde 3 diferentes aplicaciones:

1. Control desde una pagina web con un boton de encendido y apagado
2. Control desde una aplicación móvil con botón de encendido y botón de apagado
3. Control desde la misma aplicación móvil con comandos de voz

**ORIENTACIÓN**:

Realizar una aplicación en APP INVENTOR para el control web de luces desde el celular y control mediante voz utilizando el gestor de reconocimiento de voz de Google, a su vez las luces que sean controladas desde la web utilizando su dirección IP del NodeMCU.

**INSTRUMENTO DE EVALUACIÓN:**

REALIZAR UNA APLICACIÓN Y SU PROGRAMA PARA EL NODEMCU PARA EL CONTROL DE:

CONTROL CON BOTONES ENCENDER APAGAR APP INVENTOR

CONTROL CON VOZ UTILIZANDO GOOGLE APP INVENTOR

CONTROL WEB NAVEGADOR

**PROYECTO A PRESENTAR:**

1.- Presentar la aplicación móvil para el control mediante botones de encendido y apagado (apk y aia)

2.- Presentar el programa completo para el control de las 3 aplicaciones, realizado en IDE Arduino

3.- Realizar un video explicando paso a paso el funcionamiento del proyecto.

PAGINA WEB <https://appinventor.mit.edu/>

**MODO ESTACION (UTILIZANDO ROUTER)**



//MODO ESTACION

#include <WiFi.h>

//------------------Servidor Web en puerto 80---------------------

WiFiServer server(80);

//---------------------Credenciales de WiFi-----------------------

const char\* ssid = "ZTE-e1c593";

const char\* password = "b805abe1";

//---------------------VARIABLES GLOBALES-------------------------

int contconexion = 0;

String header; // Variable para guardar el HTTP request

String estadoSalida = "off";

const int salida = 2;

//------------------------CODIGO HTML------------------------------

String pagina = "<!DOCTYPE html>"

"<html>"

"<head>"

"<meta charset='utf-8' />"

"<title>Servidor Web ESP32</title>"

"</head>"

"<body>"

"<center>"

"<h1>Servidor Web ESP32</h1>"

"<h1>MODO ESTACION</h1>"

"<p><a href='/on'><button style='margin:auto; background-color:green;color:#f5a9a9;height:50px;width:100px'>ON</button></a></p>"

"<p><a href='/off'><button style='margin:auto; background-color:red;color:#f5a9a9;height:50px;width:100px'>OFF</button></a></p>"

"</center>"

"</body>"

"</html>";

//---------------------------SETUP--------------------------------

void setup() {

Serial.begin(115200);

Serial.println("");

pinMode(salida, OUTPUT);

digitalWrite(salida, LOW);

// Conexión WIFI

WiFi.begin(ssid, password);

//Cuenta hasta 50 si no se puede conectar lo cancela

while (WiFi.status() != WL\_CONNECTED and contconexion <50) {

++contconexion;

delay(500);

Serial.print(".");

}

if (contconexion <50) {

//para usar con ip fija

//IPAddress ip(192,168,1,180);

//IPAddress gateway(192,168,1,1);

//IPAddress subnet(255,255,255,0);

//WiFi.config(ip, gateway, subnet);

Serial.println("");

Serial.println("WiFi conectado");

Serial.println(WiFi.localIP());

server.begin(); // iniciamos el servidor

}

else {

Serial.println("");

Serial.println("Error de conexion");

}

}

//----------------------------LOOP----------------------------------

void loop(){

WiFiClient client = server.available(); // Escucha a los clientes entrantes

if (client) { // Si se conecta un nuevo cliente

Serial.println("New Client."); //

String currentLine = ""; //

while (client.connected()) { // loop mientras el cliente está conectado

if (client.available()) { // si hay bytes para leer desde el cliente

char c = client.read(); // lee un byte

Serial.write(c); // imprime ese byte en el monitor serial

header += c;

if (c == '\n') { // si el byte es un caracter de salto de linea

// si la nueva linea está en blanco significa que es el fin del

// HTTP request del cliente, entonces respondemos:

if (currentLine.length() == 0) {

client.println("HTTP/1.1 200 OK");

client.println("Content-type:text/html");

client.println("Connection: close");

client.println();

// enciende y apaga el GPIO

if (header.indexOf("GET /on") >= 0) {

Serial.println("GPIO on");

estadoSalida = "on";

digitalWrite(salida, HIGH);

} else if (header.indexOf("GET /off") >= 0) {

Serial.println("GPIO off");

estadoSalida = "off";

digitalWrite(salida, LOW);

}

// Muestra la página web

client.println(pagina);

// la respuesta HTTP temina con una linea en blanco

client.println();

break;

} else { // si tenemos una nueva linea limpiamos currentLine

currentLine = "";

}

} else if (c != '\r') { // si C es distinto al caracter de retorno de carro

currentLine += c; // lo agrega al final de currentLine

}

}

}

// Limpiamos la variable header

header = "";

// Cerramos la conexión

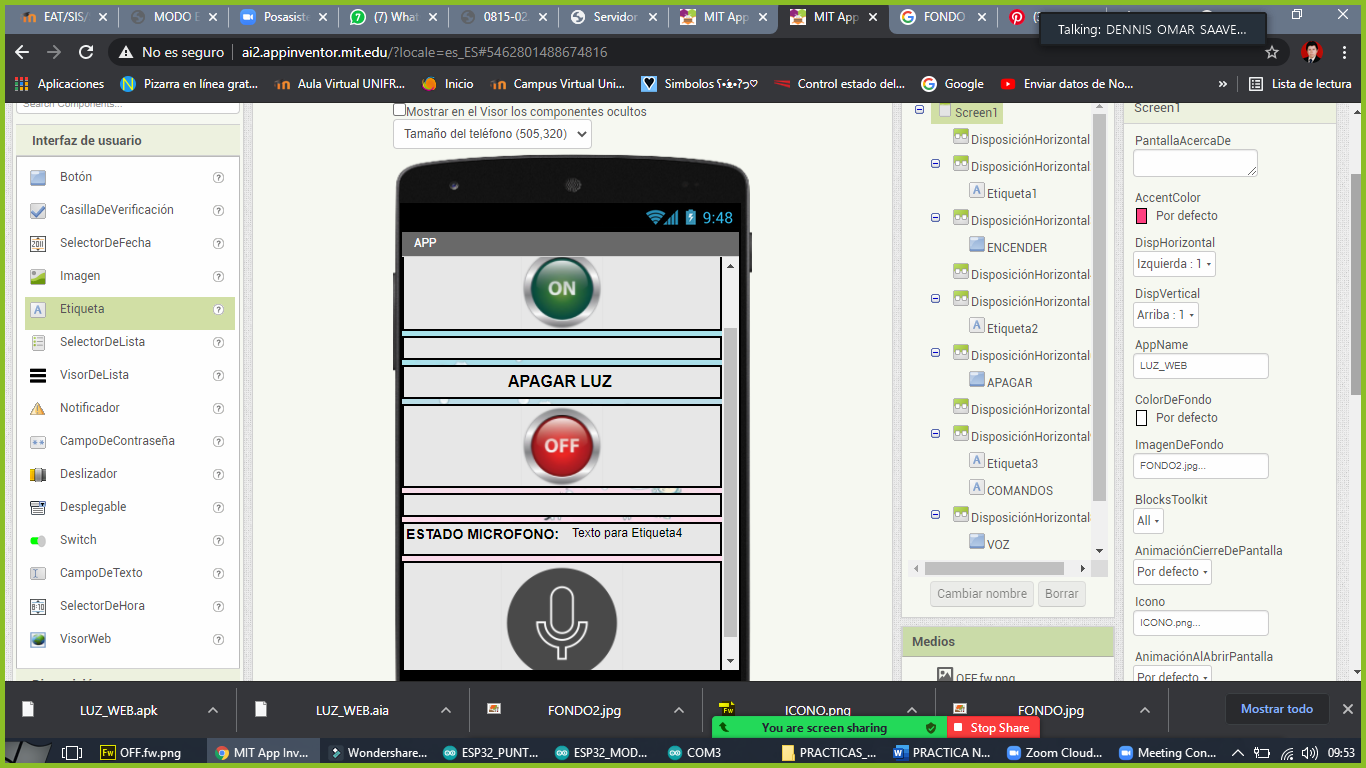
client.stop();

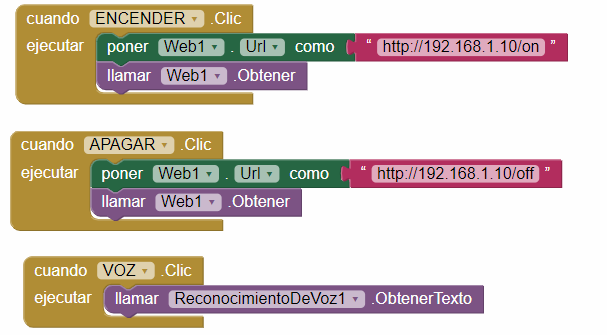
Serial.println("Client disconnected.");

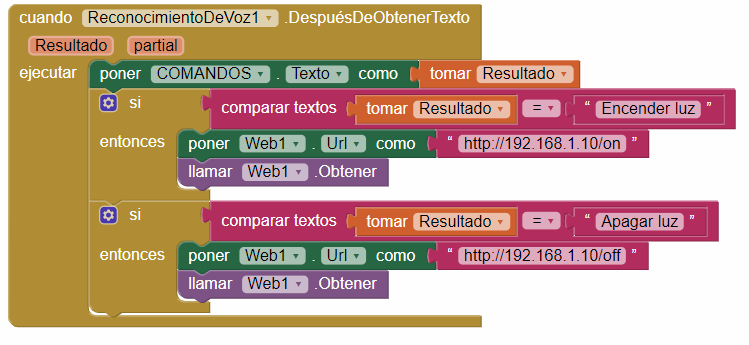
Serial.println("");

}

}







MODO ACCESS POINT (SIN ROUTER)

CODIGO

//#include <ESP8266WiFi.h> // ESP8266

#include <WiFi.h> //Habilitar para el ESP32

//------------------Servidor Web en puerto 80---------------------

WiFiServer server(80);

//---------------------Credenciales de WiFi-----------------------

const char\* ssid = "SAAVEDRA";

const char\* password = "12345678";

//---------------------VARIABLES GLOBALES-------------------------

int contconexion = 0;

String header; // Variable para guardar el HTTP request

String estadoSalida1 = "off1";

const int LED\_1 = 22;

const int LED\_2 = 3;

//------------------------CODIGO HTML------------------------------

String pagina = "<!DOCTYPE html>"

"<html>"

"<head>"

"<meta charset='utf-8' />"

"<title>Familia Saavedra</title>"

"</head>"

"<body>"

"<center>"

"<h1>CONTROL LUCES</h1>"

"<h4>INTERRUPTOR</h4>"

"<p><a href='/on1'><button style='margin:auto; background-color:green;color:white;font-size:1.2em;padding:12px;'>ON SALA</button></a>"

" <a href='/off1'><button style='margin:auto; background-color:red;color:white;font-size:1.2em;padding:12px;'>OFF SALA</button></a></p>"

"<h4>PULSADOR</h4>"

"<p><a href='/on2'><button style='margin:auto; background-color:green;color:white;font-size:1.2em;padding:12px;'>ON DORMITORIO</button></a>"

" <a href='/off2'><button style='margin:auto; background-color:red;color:white;font-size:1.2em;padding:12px;'>OFF DORMITORIO</button></a></p>"

"</center>"

"</body>"

"</html>";

//---------------------------SETUP--------------------------------

void setup() {

Serial.begin(115200);

Serial.println("");

pinMode(LED\_1, OUTPUT);

digitalWrite(LED\_1, LOW);

pinMode(LED\_2, OUTPUT);

digitalWrite(LED\_2, LOW);

// Conexión WIFI

WiFi.softAP(ssid, password);

Serial.println( "" );

Serial.println( "WiFi AP is now running" );

Serial.println( "IP address: " );

Serial.println( WiFi.softAPIP() );

server.begin();

}

//----------------------------LOOP----------------------------------

void loop(){

WiFiClient client = server.available(); // Escucha a los clientes entrantes

if (client) { // Si se conecta un nuevo cliente

Serial.println("New Client."); //

String currentLine = ""; //

while (client.connected()) { // loop mientras el cliente está conectado

if (client.available()) { // si hay bytes para leer desde el cliente

char c = client.read(); // lee un byte

Serial.write(c); // imprime ese byte en el monitor serial

header += c;

if (c == '\n') { // si el byte es un caracter de salto de linea

// si la nueva linea está en blanco significa que es el fin del

// HTTP request del cliente, entonces respondemos:

if (currentLine.length() == 0) {

client.println("HTTP/1.1 200 OK");

client.println("Content-type:text/html");

client.println("Connection: close");

client.println();

// enciende y apaga el GPIO

if (header.indexOf("GET /on1") >= 0) {

Serial.println("Led1 on");

estadoSalida1 = "on1";

digitalWrite(LED\_1, HIGH);

} else if (header.indexOf("GET /off1") >= 0) {

Serial.println("Led1 off");

estadoSalida1 = "off1";

digitalWrite(LED\_1, LOW);

}

if (header.indexOf("GET /on2") >= 0) {

digitalWrite(LED\_2, HIGH);

} else if (header.indexOf("GET /off2") >= 0) {

digitalWrite(LED\_2, LOW);

}

// Muestra la página web

client.println(pagina);

// la respuesta HTTP temina con una linea en blanco

client.println();

break;

} else { // si tenemos una nueva linea limpiamos currentLine

currentLine = "";

}

} else if (c != '\r') { // si C es distinto al caracter de retorno de carro

currentLine += c; // lo agrega al final de currentLine

}

}

}

// Limpiamos la variable header

header = "";

// Cerramos la conexión

client.stop();

Serial.println("Client disconnected.");

Serial.println("");

}

}

