



A5.1 Learning activity

Sensor system and actuation of the color of an object, and visual interface

Instructions

- Perform a color identification system using a TCS34725 RGB sensor, a nodeMCU ESP32, an SG90 Servomotor actuator, any communication protocol and a visual interface that it can be developed by the team or supported by others such as Node-red for example.
- Any activity or challenge must be carried out using the **MarkDown style with .md extension** and the VSCode development environment, and must be prepared as a **single page** document, that is, if the document has images, links or Any external document must be accessed from tags and links, and must be named with the nomenclature **A5.1_TituloActivity_NombreAlumno.pdf**.
- It is a requirement that the .md contains a tag of the link to the repository of your document in GITHUB, for example **Link to my GitHub** and at the end of the challenge it should be uploaded to github.
- From the **.md** file, export a **.pdf** file that must be uploaded to classroom within its corresponding section, serving as evidence of its delivery, since being the **official** platform, the rating of your activity.
- Considering that the **.PDF** file, which was obtained from the **.MD** file, both must be identical.
- Your repository in addition to having a **readme.md** file in your root directory, with information such as student data, work team, subject, career, advisor data, and even logo or images, it must have a section of contents or index, which really are links or **links to your .md documents** , _ avoid using text_ to indicate internal or external links.
- A structure is proposed as indicated below, however any other that supports you can be used to organize your repository.

```
- readme.md
- blog
  - C5.1_TituloActividad.md
  - C5.2_TituloActividad.md
- img
- docs
  - A5.1_TituloActividad.md
  - A5.2_TituloActividad.md
```

Development

1. Use the following list of materials to prepare the activity

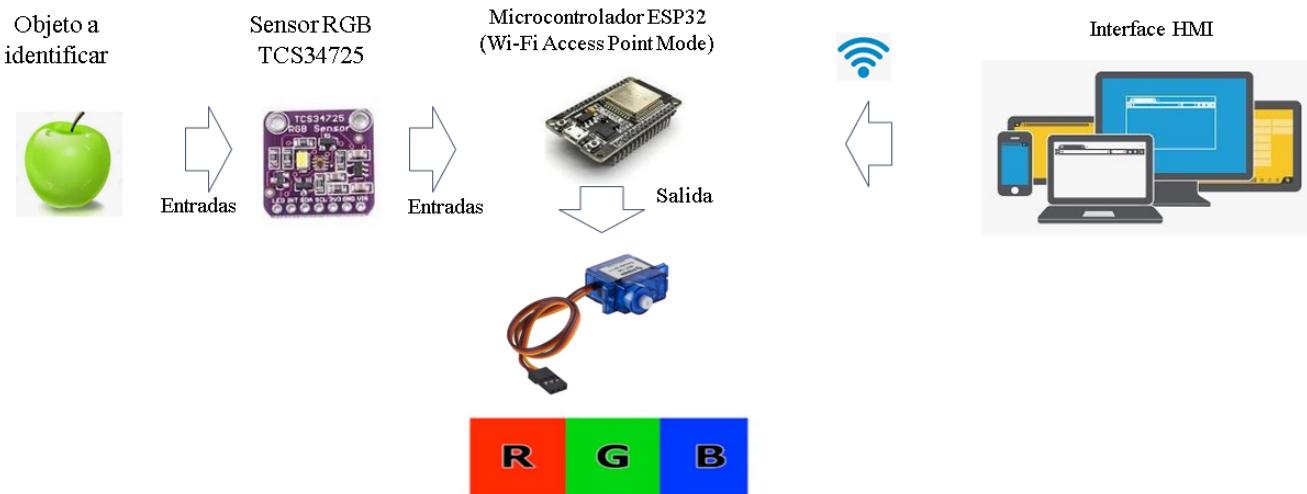
Quantity	Description
1	RGB Sensor TCS34725
1	5V voltage source

Quantity	Description
1	NodeMCU ESP32
1	BreadBoard
1	Jumpers M/M
1	SG90 servo motor

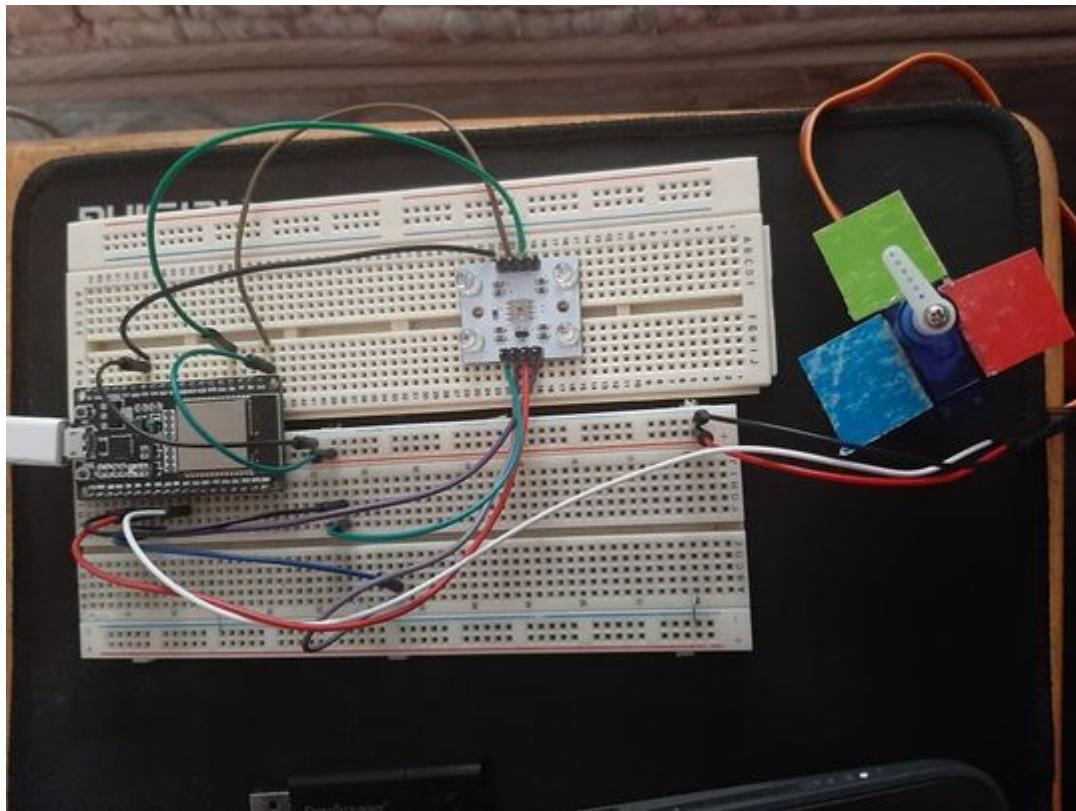
2. Based on the images shown in **Figure 1**, assemble an electronic circuit, in order to obtain a system capable of complying with the following instructions:

- The first phase of the activity will consist of, when placing an object in front of the RGB sensor, it should identify what color it is (it is recommended to use Red, Green, and Blue objects to greater precision), which should show in a visual interface which color was detected.
- The second phase will consist of adding an actuator that, based on the color identified, the actuator should point to the color being detected.

Figure 1 ESP32 Color Sensor and Servomotor Circuit



3. Place the picture of the assembled circuit here



4. Place the program created within the Arduino environment in this place

```
#define S0 15 //Pin de resolucion de datos
#define S1 2 //Pin de resolucion de datos
#define S2 0 //Pin para el filtro de color
#define S3 4 //Pin para el filtro de color
#define Color 34
#define ServoM 5
#include <WiFi.h>
//Servidor web en puerto 80
WiFiServer server(80);

int freq = 50, ServoChannel = 0, resolution = 8;
String C;

//Datos de la red wifi
const char* ssid = "INFINITUM537D";
const char* password = "1263743512";

//Variables globales
int contConexion = 0;
String header; //Variable para guardar el HTTP request

//Codigo HTML
String PAG = "<!DOCTYPE html><html>
<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">
<META HTTP-EQUIV='Refresh' CONTENT='1'>
<link rel=\"icon\" href=\"data:,\">
<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto;
```

```
text-align: center;"}
".button { background-color: #009E06; border: none; color: white; padding: 16px
40px;
"text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;"}
".button2 {background-color: #FF0000;}.button3 {background-color: #000CFF;}
```

</style></head>

```
"<body><h1>Actividad 5.1 Sensor de colores </h1>";

String CR = "<p> Color: ROJO </p>"
"<p><a><button class=\"button button2\"> ROJO </button></a></p>";
String CV = "<p> Color: VERDE </p>"
"<p><a><button class=\"button\"> VERDE </button></a></p>";
String CA = "<p> Color: AZUL </p>"
"<p><a><button class=\"button button3\"> AZUL </button></a></p>";

String END = "</body></html>";

void setup() {
    // put your setup code here, to run once:
    pinMode(S0,OUTPUT);
    pinMode(S1,OUTPUT);
    pinMode(S2,OUTPUT);
    pinMode(S3,OUTPUT);

    pinMode(Color,INPUT);

    digitalWrite(S0,HIGH);
    digitalWrite(S1,HIGH);

    ledcSetup(ServoChannel, freq, resolution);
    ledcAttachPin(ServoM, ServoChannel);
    Serial.begin(115200);

    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)
    {
        Serial.println("");
        Serial.println("WiFi conectado");
        Serial.println(WiFi.localIP());
        server.begin(); // Iniciamos el servidor
    }
    else
    {
        Serial.println("");
        Serial.println("Error de conexion");
    }
}
```

```
}

void loop() {
    // put your main code here, to run repeatedly:
    ColorSensor();
    Pagina();

}

void ColorSensor(){
    digitalWrite(S2,LOW);      // establece fotodiodos
    digitalWrite(S3,LOW);      // con filtro rojo
    int rojo = pulseIn(Color, LOW); // obtiene duracion de pulso de salida del
sensor
    delay(200);      // demora de 200 mseg

    digitalWrite(S2,HIGH);     // establece fotodiodos
    digitalWrite(S3,HIGH);     // con filtro verde
    int verde = pulseIn(Color, LOW); // obtiene duracion de pulso de salida del
sensor
    delay(200);      // demora de 200 mseg

    digitalWrite(S2,LOW);      // establece fotodiodos
    digitalWrite(S3,HIGH);     // con filtro azul
    int azul = pulseIn(Color, LOW); // obtiene duracion de pulso de salida del
sensor
    delay(200);      // demora de 200 mseg

    Serial.print("R:");      // muestra texto
    Serial.print(rojo);      // muestra valor de variable rojo

    Serial.print("\t");      // espacio de tabulacion

    Serial.print("V:");      // muestra texto
    Serial.print(verde);      // muestra valor de variable verde

    Serial.print("\t");      // espacio de tabulacion

    Serial.print("A:");      // muestra texto
    Serial.println(azul);      // muestra valor de variable azul
    // y salto de linea

    if ( rojo < 25 && verde > 58 && azul > 7){  // si valores dentro del rango
        Serial.println("ROJO"); // muestra texto
        C = "ROJO";
        ledcWrite(ServoChannel,8); // 0 grados
    }
    else if ( verde < 50 && rojo > 30 && rojo < 70 && azul > 9){ // si valores
dentro del rango
        Serial.println("VERDE"); // muestra texto
        C = "VERDE";
        ledcWrite(ServoChannel,24); // 90 grados
    }
}
```

```
    else if ( azul < 25 && rojo > 70 && verde > 40){ // si valores dentro del
rango
    Serial.println("AZUL");           // muestra texto
    C = "AZUL";
    ledcWrite(ServoChannel,34); //180 grados
}
}

void Pagina()
{
    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();

                        client.println(PAG);

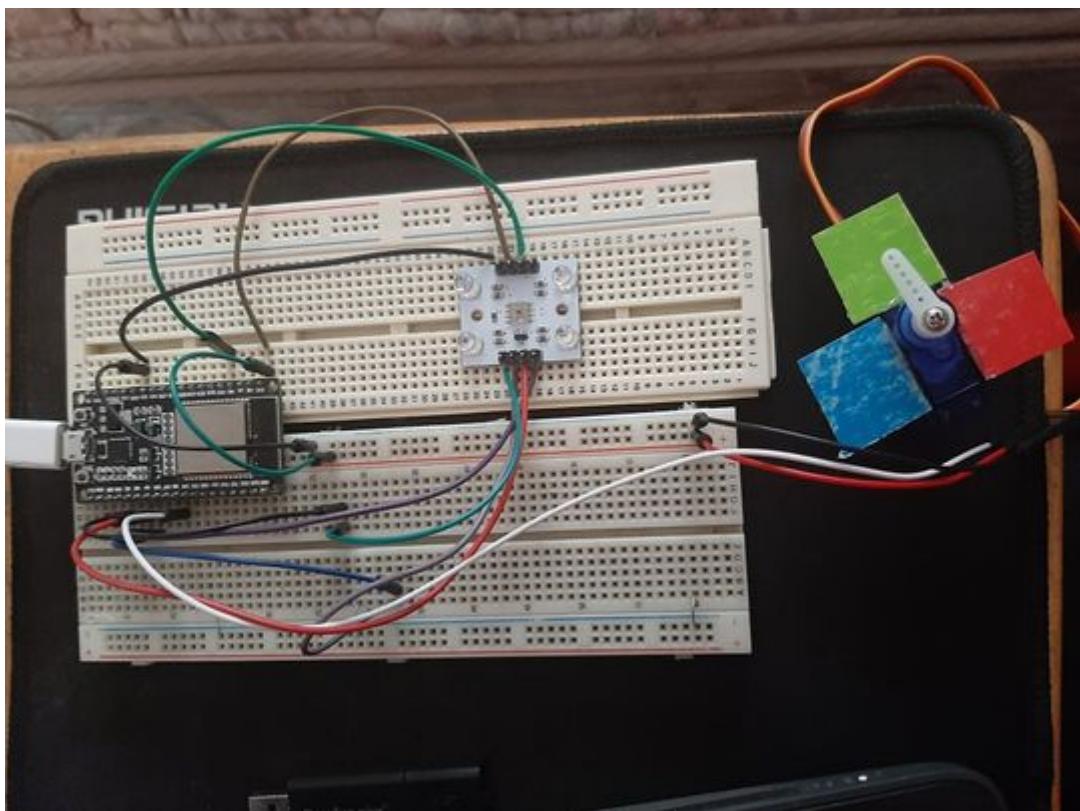
                        if(C == "ROJO")
                        {
                            client.println(CR);
                        }
                        else if(C == "VERDE")
                        {
                            client.println(CV);
                        }
                        else if(C == "AZUL")
                        {
                            client.println(CA);
                        }

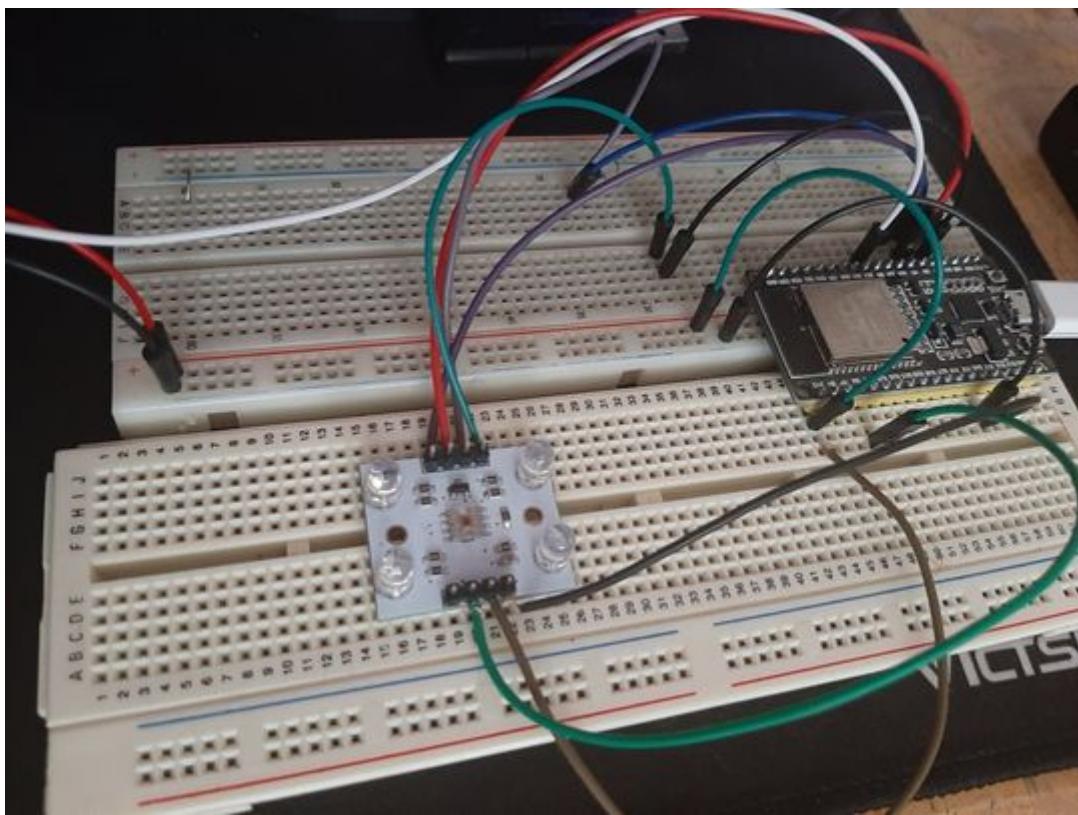
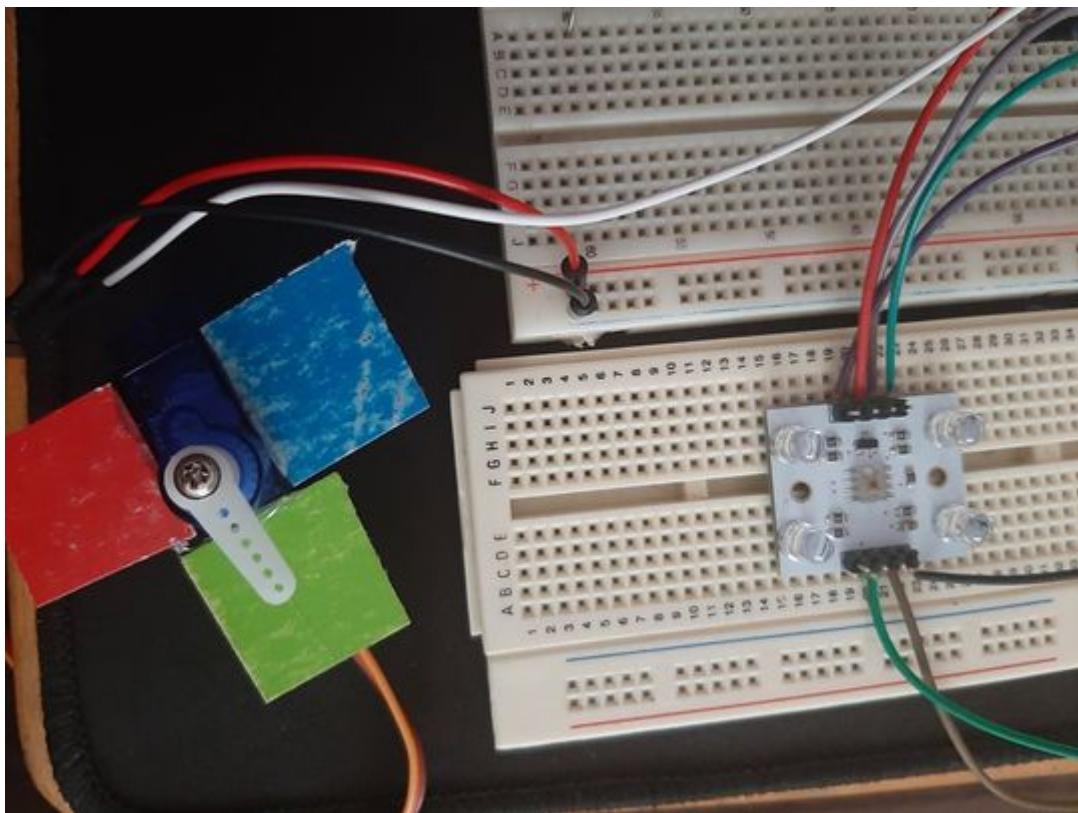
                        client.println(END);
                        // la respuesta HTTP termina con una linea en blanco
                        client.println();
                        break;
                    } else { // si tenemos una nueva linea limpiamos currentLine
                        currentLine = "";
                    }
                }
            }
        }
    }
}
```

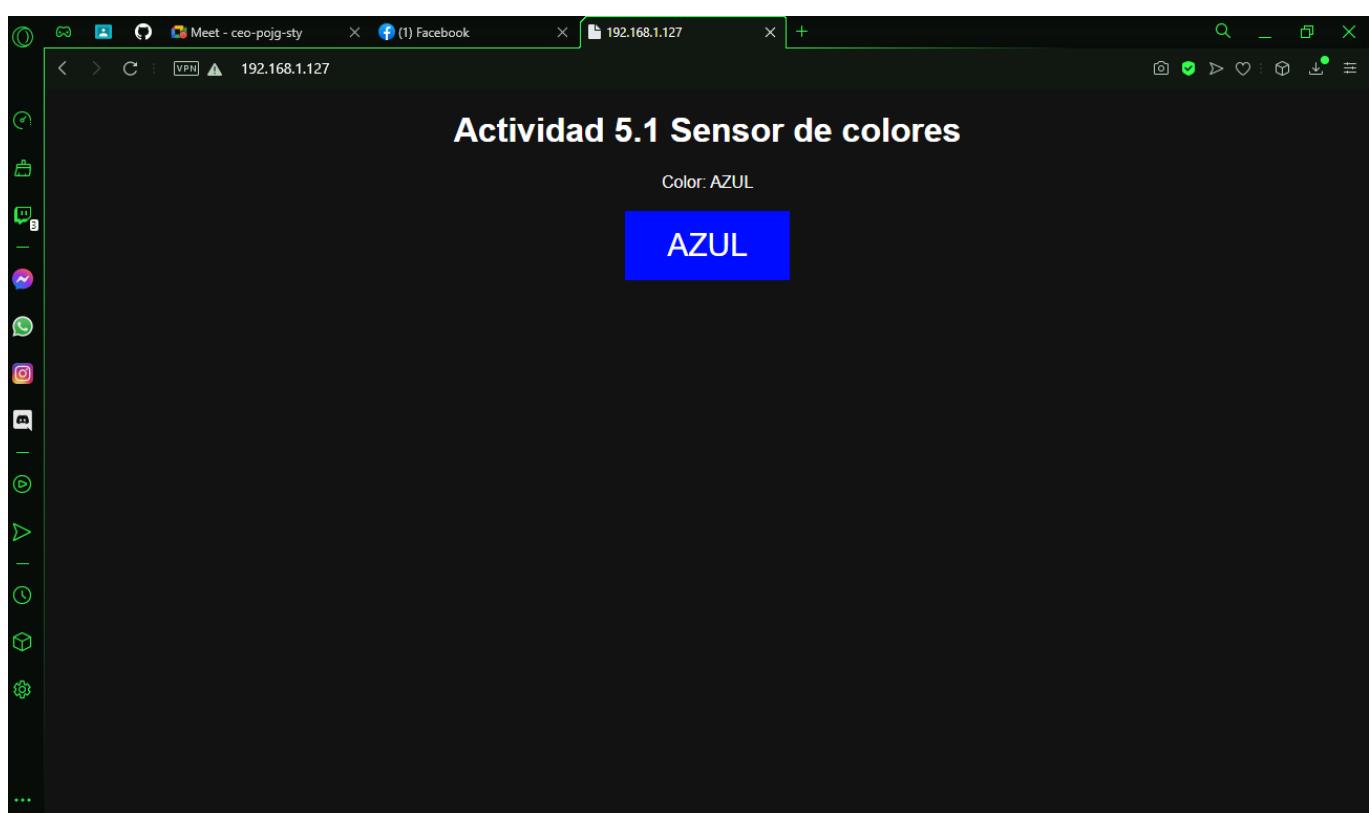
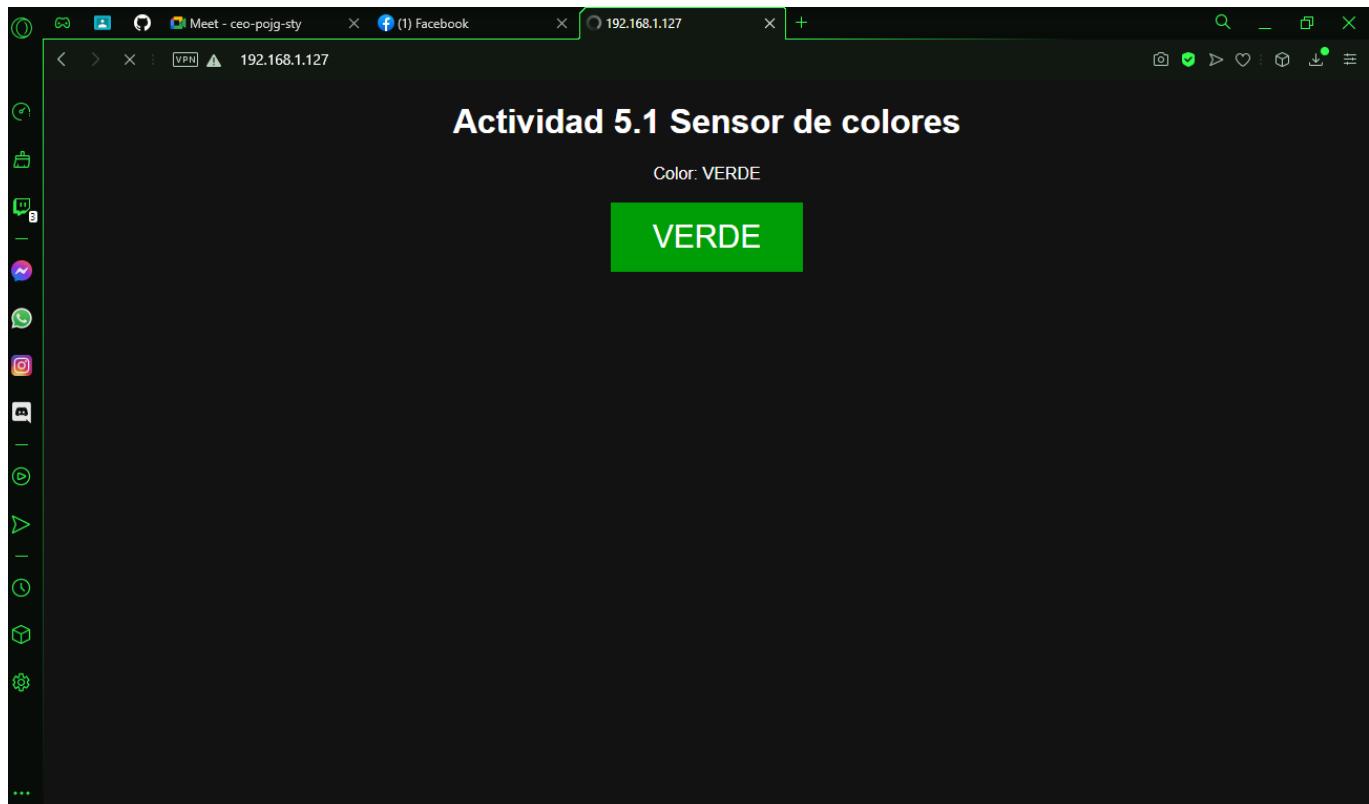
```
        }
    } else if (c != '\r') { // si C es distinto al caracter de retorno de
carro
        currentLine += c;      // lo agrega al final de currentLine
    }
}
}

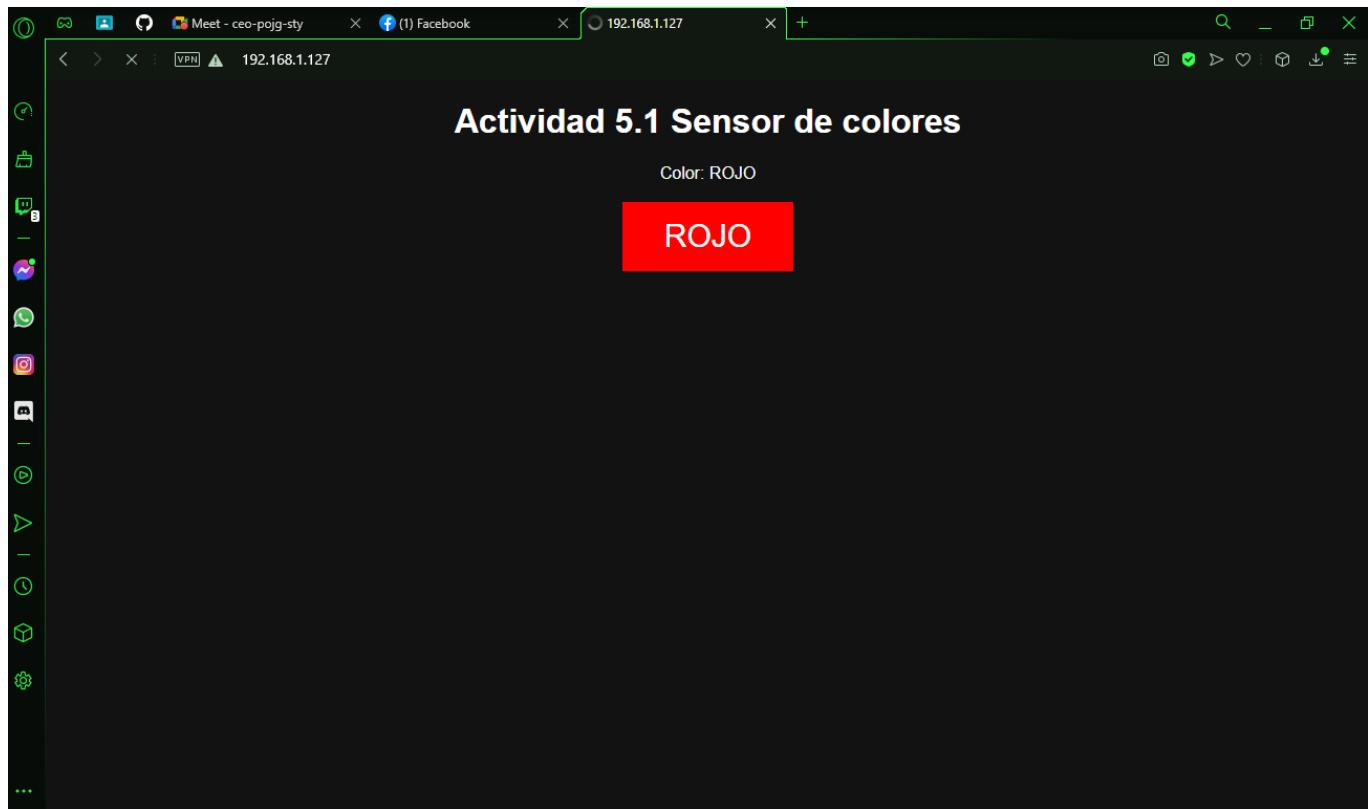
// Limpiamos la variable header
header = "";
// Cerramos la conexión
client.stop();
Serial.println("Client disconnected.");
Serial.println("");
}
```

5. Place here evidence that you consider important during the development of the activity.









6. For the demonstration of the activity, more than one object must be used to cover at least three colors.

7. Insert images of **evidence** such as meetings of the team members held for the development of the activity

The screenshot shows a Google Meet session. On the left, an Arduino IDE window titled "SensorColor Arduino 1.8.15" displays the following code:

```

#define S0 15 //Pin de resolucion de datos
#define S1 2 //Pin de resolucion de datos
#define S2 0 //Pin para el filtro de color
#define S3 4 //Pin para el filtro de color
#define Color 34
#define ServoM 5
#include <WiFi.h>
//Servidor web en puerto 80
WiFiServer server(80);

int freq = 50, ServoChannel = 0, resolution = 8;
String C;

//Datos de la red wifi
const char* ssid = "INFINITUM537D";
const char* password = "1263743512";

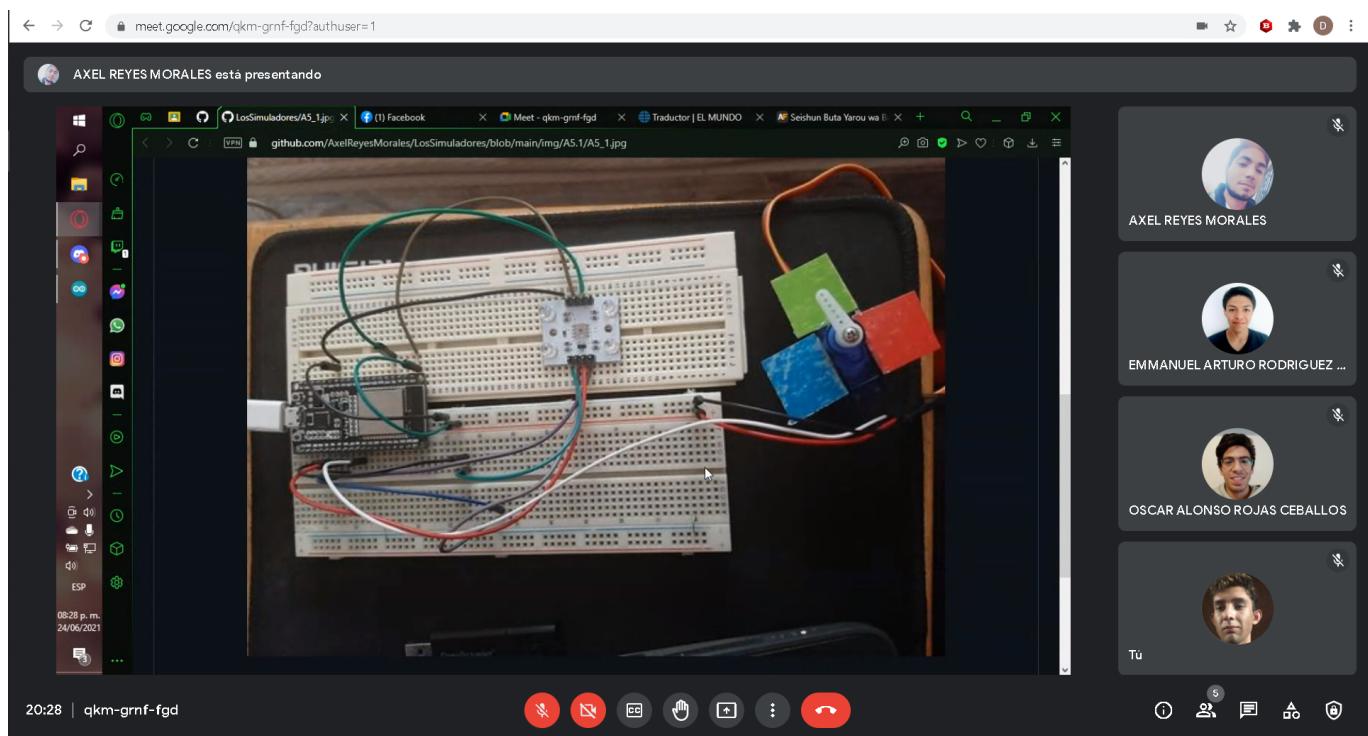
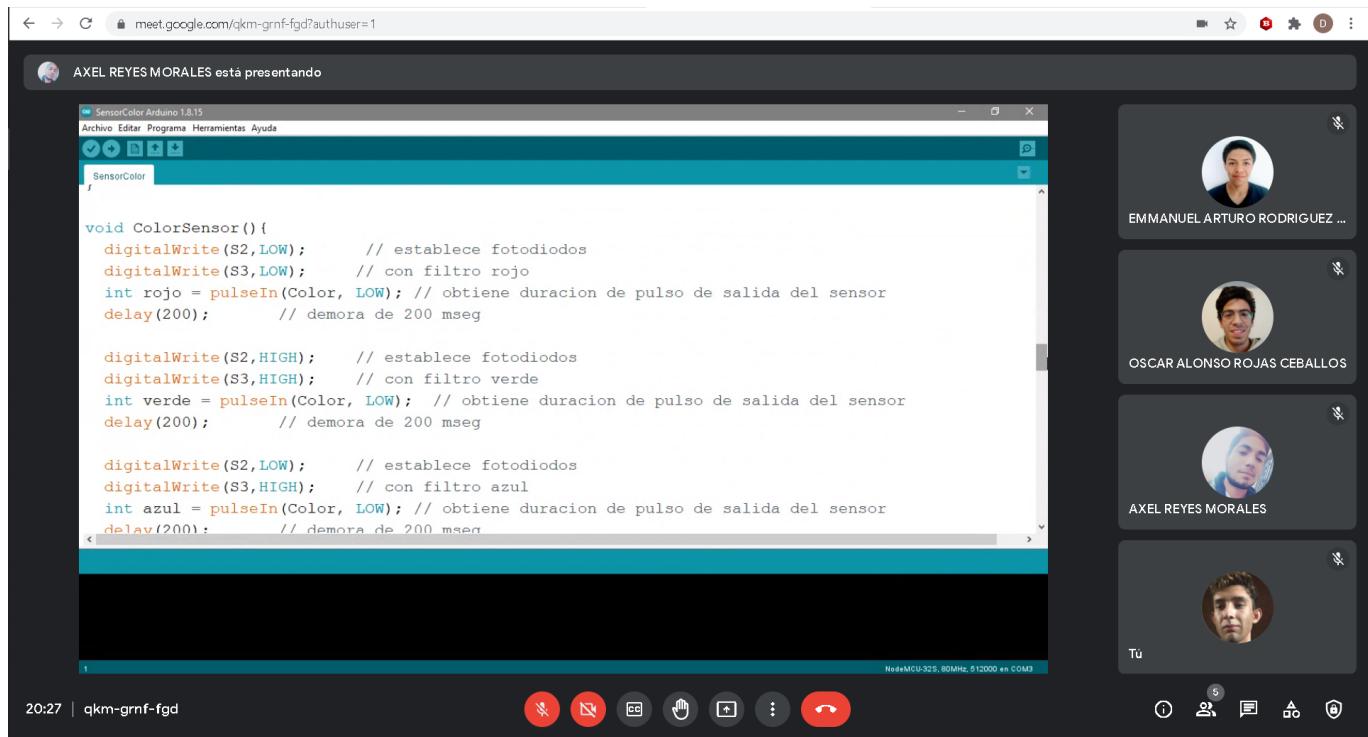
```

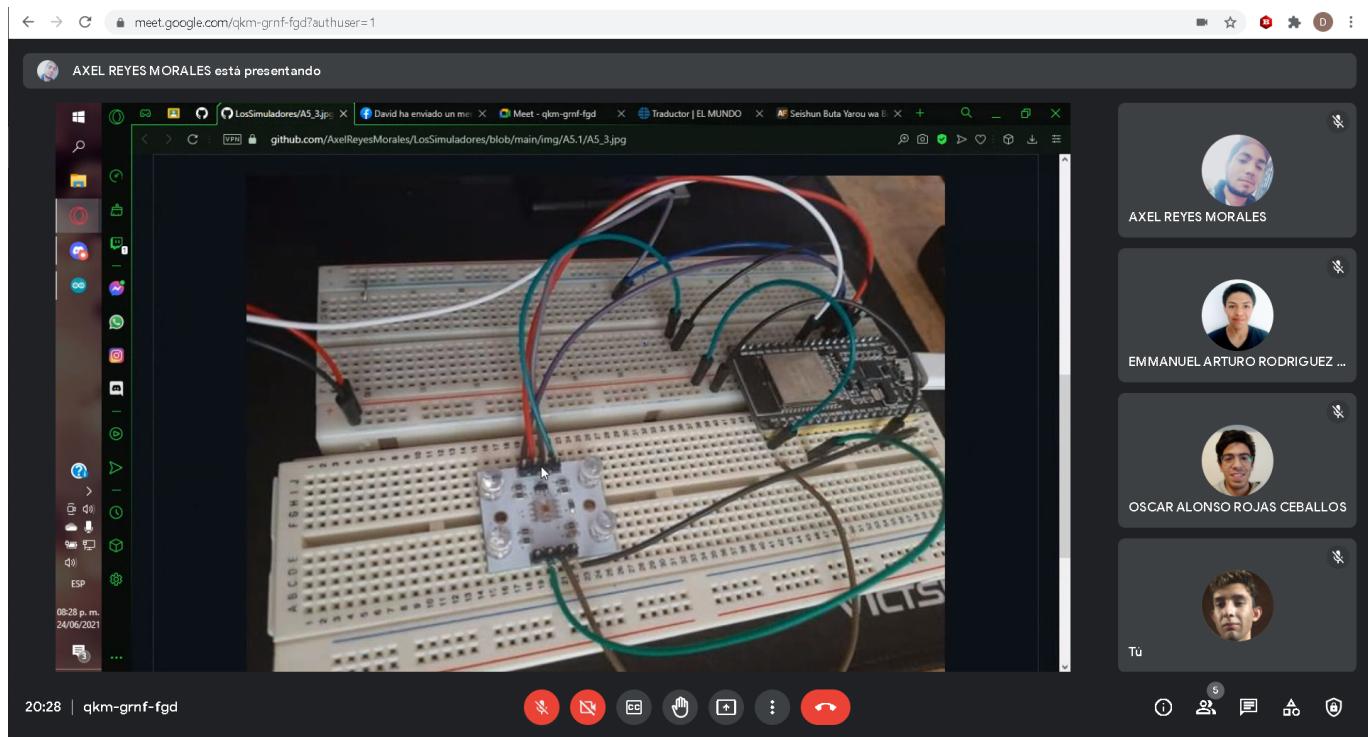
On the right, a participant list is visible with four entries:

- EMMANUEL ARTURO RODRIGUEZ ...
- OSCAR ALONSO ROJAS CEBALLOS
- AXEL REYES MORALES
- Tú

A screenshot of a Google Meet video call interface. On the left, a presentation slide titled "SensorColor Arduino 1.8.15" is displayed. The slide content includes code snippets for generating an HTML page and setting up an Arduino sketch. On the right, a participant list shows four participants: EMMANUEL ARTURO RODRIGUEZ, OSCAR ALONSO ROJAS CEBALLOS, AXEL REYES MORALES, and Tú. The participant list has a dark theme with circular profile pictures. At the bottom, a toolbar with various video and audio controls is visible.

A screenshot of a Google Meet video call interface. On the left, a presentation slide titled "SensorColor Arduino 1.8.15" is displayed. The slide content includes an Arduino sketch for a color sensor project. On the right, a participant list shows four participants: EMMANUEL ARTURO RODRIGUEZ, OSCAR ALONSO ROJAS CEBALLOS, AXEL REYES MORALES, and Tú. The participant list has a dark theme with circular profile pictures. At the bottom, a toolbar with various video and audio controls is visible.





Conclusions

Axel: To conclude I can say that this practice was somewhat laborious to understand the behavior of the color sensor readings to be able to identify the colors that were needed to identify, in addition to trying to create the interface in Node-Network but for lack of time not concrete and had to use a website created from the ESP32, since it was possible to identify colors, work with a servo motor to move it and point to the color that had been put on the sensor and that could also be seen on the website that was created, it was somewhat complicated but the expected result was reached.

David: Among the things we could observe during the development of the activity was the functionality of the RGB sensor and the servomotor, which we were able to use to make this activity work, during this activity we made a UI with a web page, we were able to make it quite easy because we had already worked with one during the previous activity, but the implementation of the RGB sensor was quite tricky and we had to investigate to be able to use it and make it work, fortunately we were able to finish the activity and see the results before the time limit.

Emmanuel: In conclusion, a circuit was made where the color of an object would be identified, between the colors red, green and blue, using an RGB sensor TSC3200 which will distinguish the difference of colors and this will be sent to the ESP32 which depending on the color that detected, it will move an SG90 servomotor, which will point to a certain degree depending on where the color identifier is placed and finally it will also be received by a web server, which we used in a previous practice, and this page will show the color that the sensor detects giving as a result of the server operation, together with the servo motor and the RGB sensor.

Oscar Rojas: In this practice we learned how to make use of a rgb sensor to detect which is the color of the object in front of the sensor, in this case we make use of a TSC3200 sensor, we implemented a servomotor to indicate which color is the one the sensor is detecting, in this case we only use 3 colors, red, blue and green, like in the previous practice we make use of a web server inside the ESP32 to indicate the color that the sensor was detecting.

Repositories

Axel [Go to my repository](#)

David [Go to my repository](#)

Emmanuel [Go to my repository](#)

Oscar Rojas [Got to my repository](#)