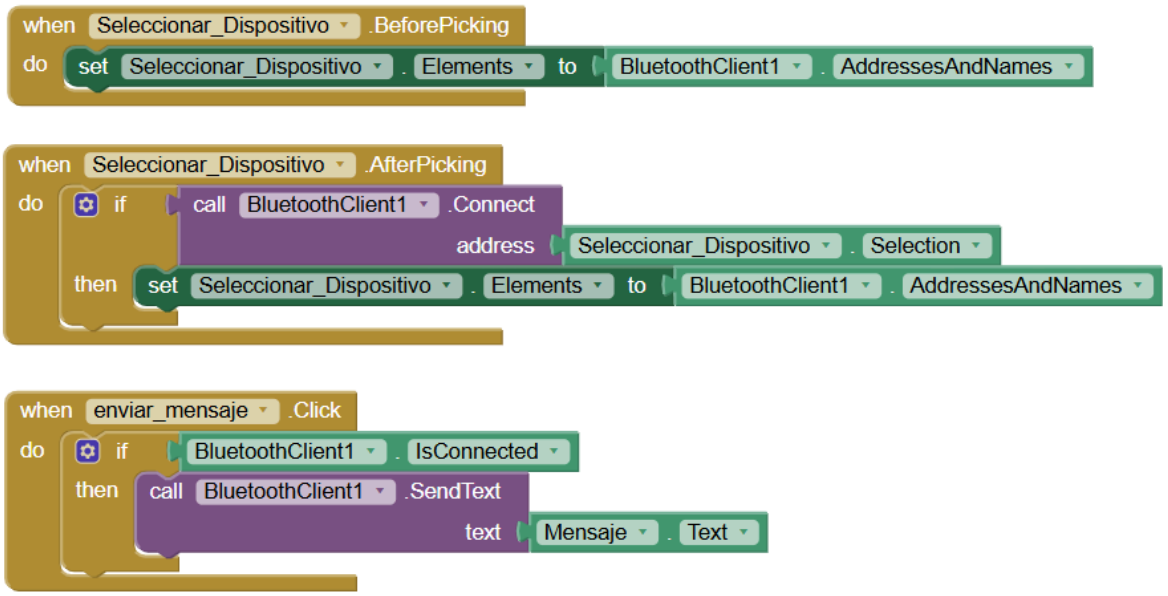
ACTIVIDAD 1

Código en bloques

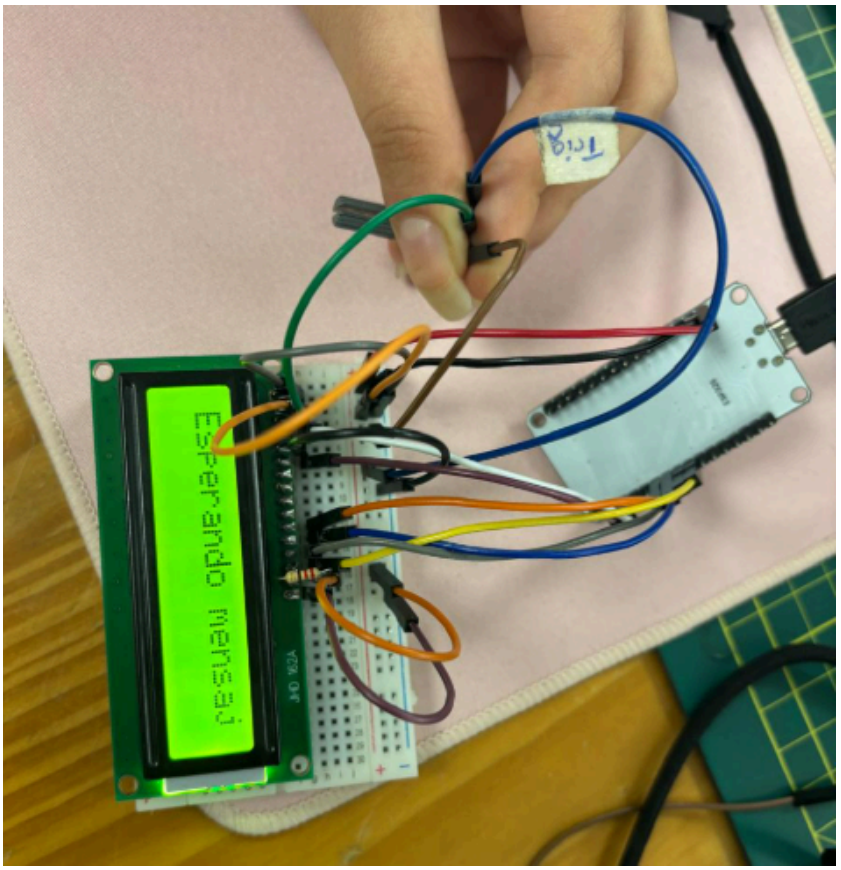


Código

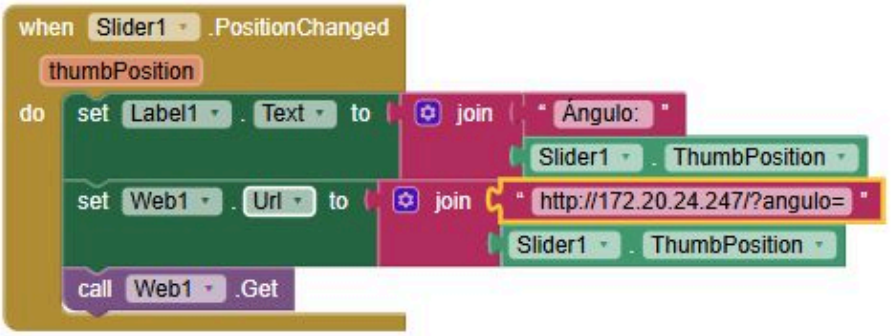
```
#include <Wire.h>
#include <LiquidCrystal.h>  // Librería para LCD paralelo
#include <BluetoothSerial.h>

// Definir el puerto Bluetooth
BluetoothSerial SerialBT;

// Configurar los pines del LCD paralelo
LiquidCrystal lcd(22, 23, 18, 19, 21, 5);  // RS, EN, D4, D5, D6, D7

void setup() {
  // Inicializar el puerto serie
  Serial.begin(115200);

  // Inicializar Bluetooth
  SerialBT.begin("ESP32_LCD");  // Nombre del dispositivo Bluetooth

  // Inicializar el LCD
  lcd.begin(16, 2);  // LCD 16x2
  lcd.print("Esperando mensaje...");

  delay(1000); // Esperar un segundo

  // Mensaje para verificar la inicialización
  Serial.println("ESP32 listo y esperando mensaje...");
}

void loop() {
```

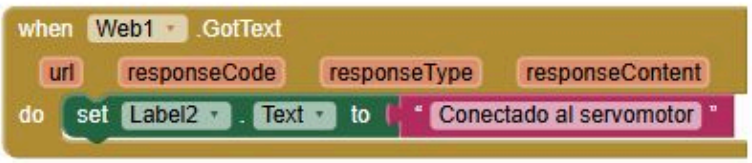
```
  if (SerialBT.available()) {
    String mensaje = SerialBT.readString();  // Leer todo el mensaje hasta que haya una pausa en la transmisión
    Serial.println(mensaje);  // Imprimir el mensaje recibido en el Monitor Serial

    lcd.clear();  // Limpiar el LCD
    lcd.setCursor(0, 0);  // Colocar el cursor en la primera fila
    lcd.print("Mensaje recibido:");
    lcd.setCursor(0, 1);  // Colocar el cursor en la segunda fila
    lcd.print(mensaje);   // Mostrar el mensaje en el LCD
  }
  delay(100); // Pequeño delay para evitar sobrecargar la comunicación
}
```

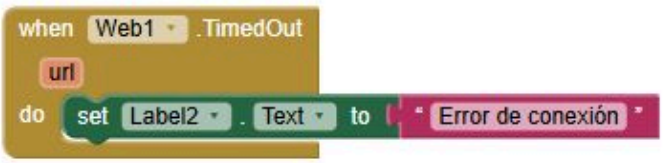
Circuito



<u>ACTIVIDAD 2</u>
Código en bloques







Código

```
#include <WiFi.h>
#include <Servo.h>

const char* ssid = "UPCH_CENTRAL";          // Cambia por tu SSID
const char* password = "CAYETANO2022";  // Cambia por tu contraseña

WiFiServer server(80);
Servo miServo;

void setup() {
  Serial.begin(115200);
  miServo.attach(13);  // Cambia el pin si usas otro

  WiFi.begin(ssid, password);
  Serial.print("Conectando a WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nConectado a la red WiFi");
  Serial.print("IP del ESP32: ");
```

```
  Serial.println(WiFi.localIP());

  server.begin();
}

void loop() {
  WiFiClient client = server.available();
  if (client) {
    Serial.println("\nCliente conectado");
    String request = client.readStringUntil('\r');
    Serial.print("Petición: ");
    Serial.println(request);
    client.readStringUntil('\n');  // Limpia buffer

    int anguloIndex = request.indexOf("angulo=");
    if (anguloIndex != -1) {
      String anguloStr = request.substring(anguloIndex + 7);
      int espacioIndex = anguloStr.indexOf(' ');
      if (espacioIndex != -1) {
        anguloStr = anguloStr.substring(0, espacioIndex);
      }

      int angulo = anguloStr.toInt();
      angulo = constrain(angulo, 0, 180);
      miServo.write(angulo);
      Serial.print("Ángulo recibido: ");
      Serial.println(angulo);

      // Respuesta correcta con mensaje esperado
      client.println("HTTP/1.1 200 OK");
      client.println("Content-Type: text/plain");
      client.println("Connection: close");
      client.println();
      client.println("Conectado al servomotor");
    } else {
      // Respuesta si no se recibe parámetro 'angulo'
      client.println("HTTP/1.1 400 Bad Request");
      client.println("Content-Type: text/plain");
      client.println("Connection: close");
      client.println();
      client.println("Error: no se recibió parámetro angulo");
    }

    delay(10);
    client.stop();
    Serial.println("Cliente desconectado");
  }
}
```

Circuito