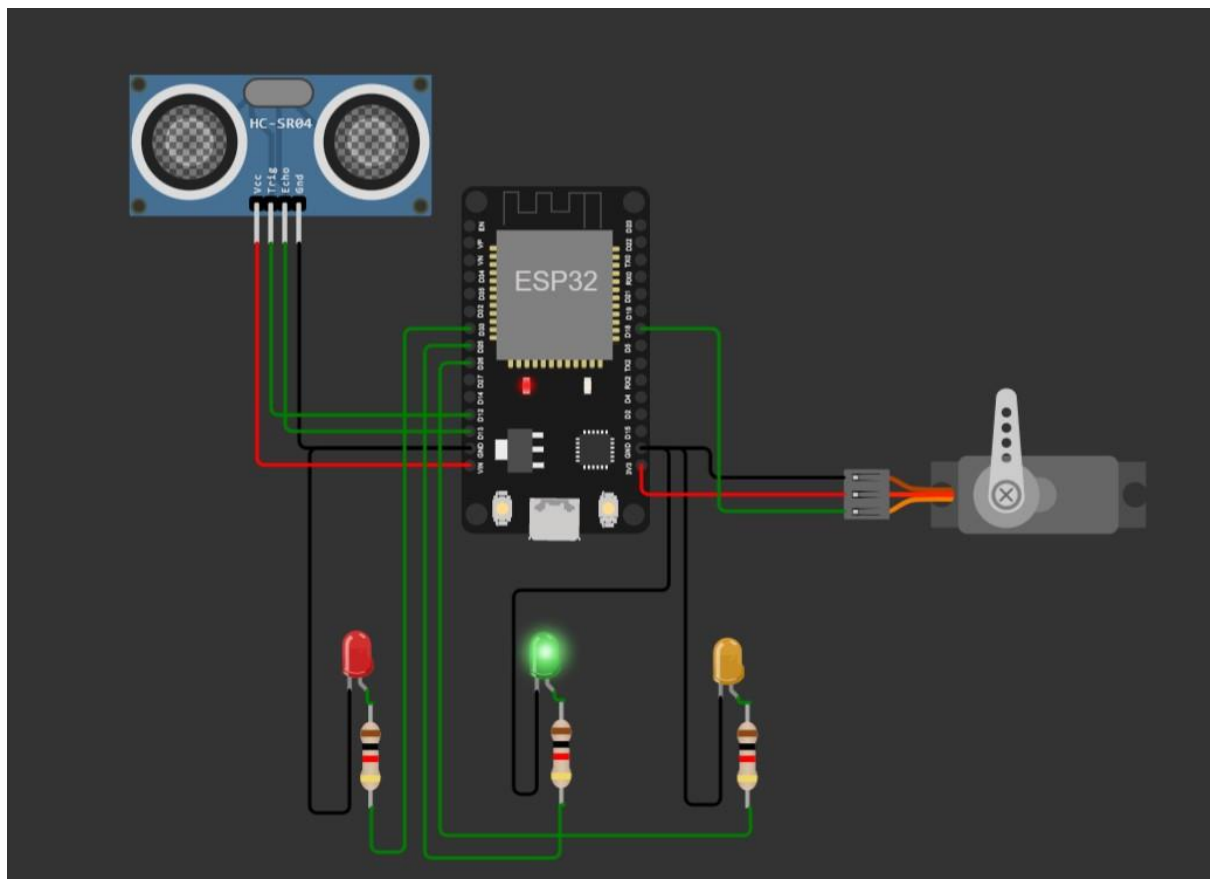# Develop a wokwi simulation

## As we have developed the project in wokwi, we have simulated using wokwi.

```
Sending payload: {"weight":14,"level":236}
Publish ok
 Weight of the Garbage in the bin(kg):10
 The Garbage Bin is NOT FULL
 Garbage_level_in_bin(cm):236
Sending payload: {"weight":10,"level":236}
Publish ok
```
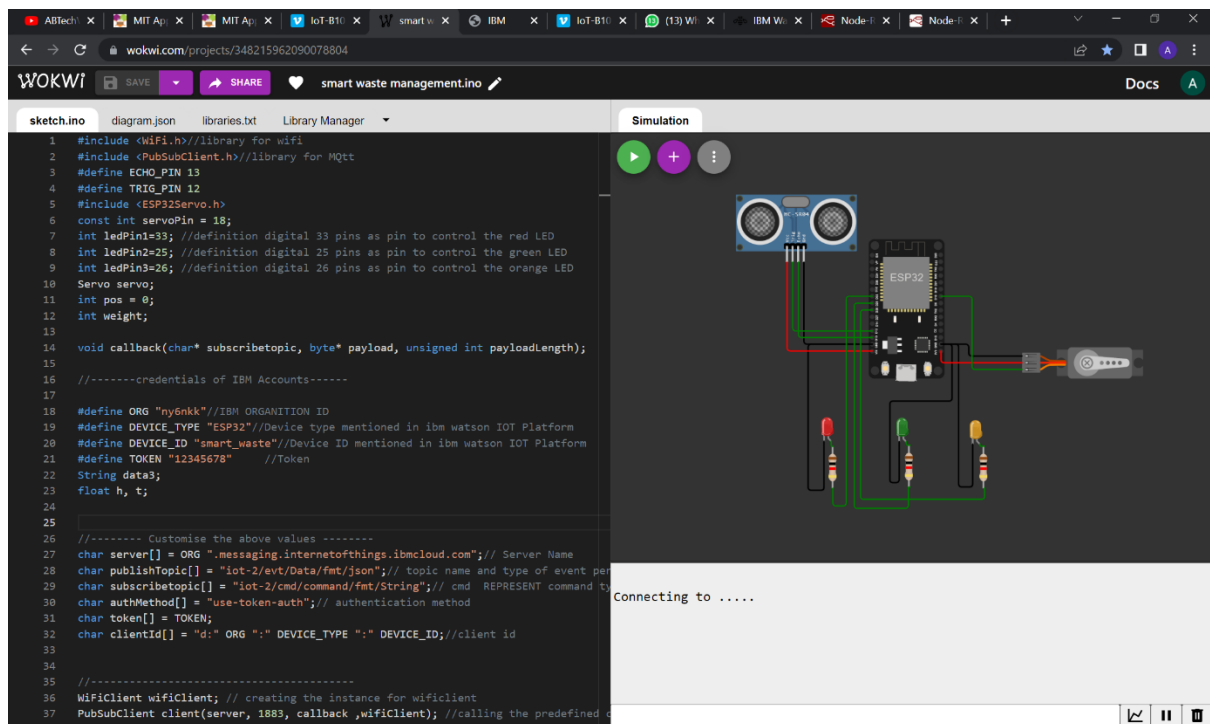


```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#define ECHO_PIN 13
#define TRIG_PIN 12
#include <ESP32Servo.h>
const int servoPin = 18;
int ledPin1=33; //definition digital 33 pins as pin to control the red LED
int ledPin2=25; //definition digital 25 pins as pin to control the green LED
int ledPin3=26; //definition digital 26 pins as pin to control the orange LED
Servo servo;
int pos = 0;
int weight;

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);

//--------credentials of IBM Accounts------

#define ORG "ny6nkk"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP32"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "smart_waste"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
float h, t;


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event per
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command ty
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id


//----------------------------------------
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined c
```

PNT2022TMID35789