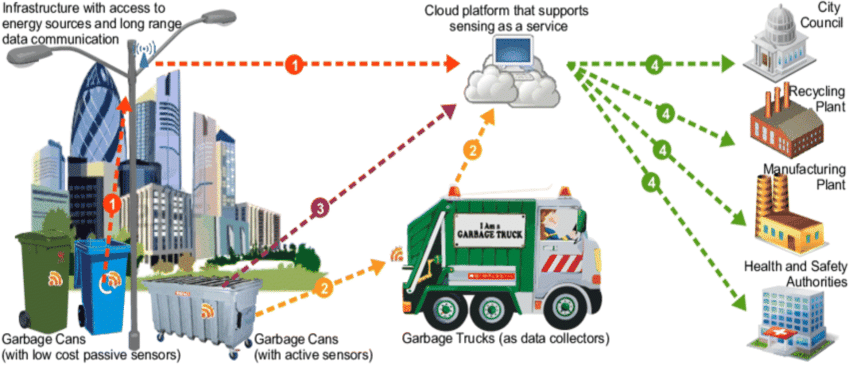
**DOCUMENTATION**

SMART WASTE MANAGEMENT



**TEAM DETAILS**

|  |  |
| --- | --- |
| **MENTOR** | **MRS.M.Maheshwari** |
| **LEADER** | **R.Abinaya(310121104002)** |
| **MEMBERS** | **K.Aruna(310121104011)**  **A.James Soosanna(310121104043)**  **J.Kaviya(310121104049)**  **K.Keerthana(310121104050)** |
| **PROBLEM DESCRIPTION** | **Describing the project objectives,IOT Sensor setup,ESP32 integration by including the diagrams,schematics and screenshots of the IOT sensor and web application.** |

**IOT(INTERNET OF THINGS)**

****

IoT technology can be described as the ‘Internet of Things,’ based on the concept that all the desired devices can be connected inside a specific network for sharing data and information without any manual intervention. We can segregate the devices used for this purpose based on their capability to send, receive, and gather data in the network. IoT’s salient properties include its secure nature, usability on smart devices, ability to connect to any network, and facilitation of faster connectivity.

**IOT CAN BE USED TO IMPROVE THE WASTE MANAGEMENT:**

* IoT-based monitoring provides real-time data for better decision-making.
* Smart waste management ca minimize the waste over flow by using the IOT.
* Improved Environment cleanliness by reducing the waste overflow.

**WASTE MANAGEMENT IN IOT**

A combination of IoT sensors and AI vision can automate processes from waste collection to disposal. For instance, many cities have deployed IoT-equipped smart bins that sent alerts when they're full. These systems ensure that bins are not overflowing and make the waste collection process more energy efficient.

**OVERVIEW OF WORKING**

Waste collection is an essential [city service](https://www.iotforall.com/what-makes-smart-city-2019/), yet existing waste management systems are resource-intensive, inefficient, and outdated. The Internet of Things (IoT) has the potential to greatly optimize collection services and reduce operational costs for cities.

All humans produce municipal solid waste, commonly known as trash or garbage, on a daily basis, yet essential waste collection systems in cities are often taken for granted by residents until a garbage bin overflows. Due to recent population growth and urbanization, waste production in cities has increased, and municipal waste collection operations need to adapt to be able [to ensure](https://www.iotforall.com/smart-city-approaches-real-world/) [clean cities.](https://www.iotforall.com/smart-city-approaches-real-world/)

**OBJECTIVES**

* Smart waste management solutions use sensors placed in waste receptacles to measure fill levels and to notify city collection services when bins are ready to be emptied.
* Over time, historical data collected by sensors can be used to identify fill patterns, optimize driver routes and schedules, and reduce operational costs.
* The cost of these sensors is steadily decreasing, making IoT waste bins more feasible to implement and more attractive to city leaders.

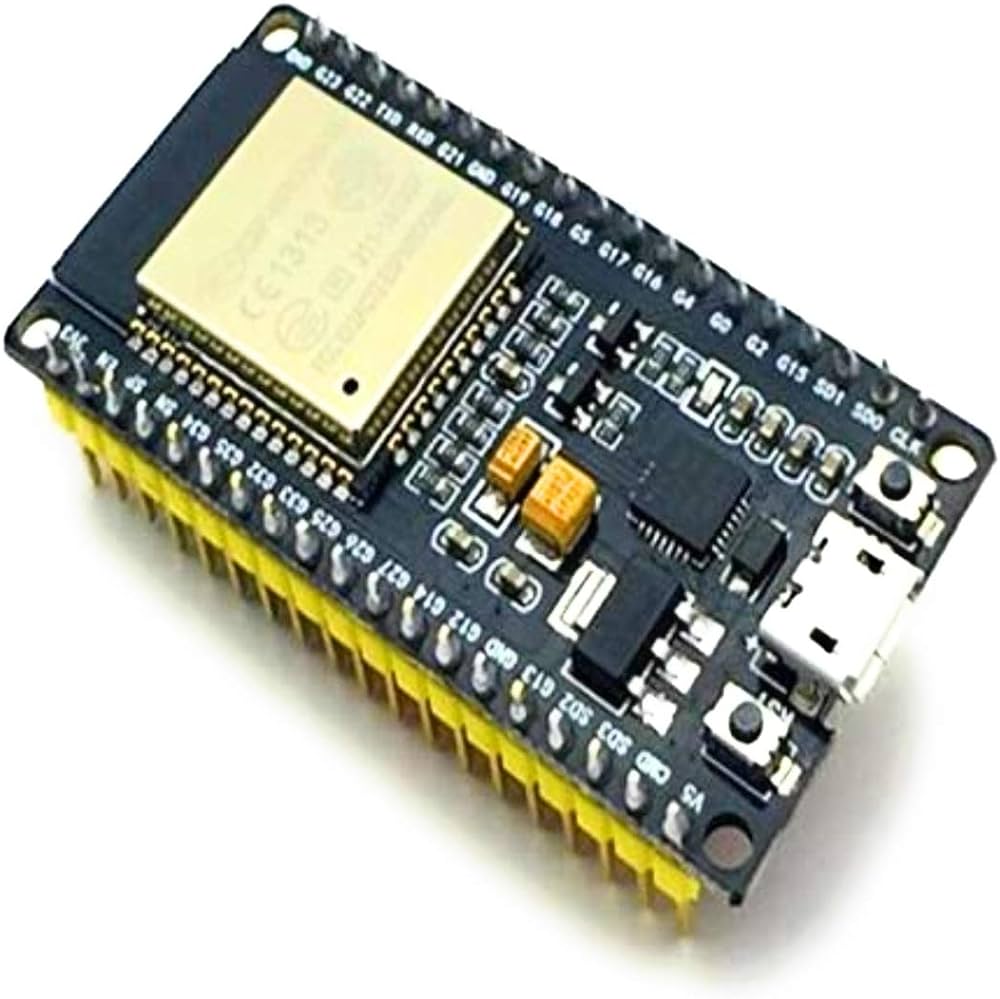
**ABSTRACT**

Every person in this world throws waste in the form of plastics, wet waste, dry waste and etc. Also, every person looks for a place or a plastic container to dispose that waste, that plastic container is the Dustbin which they look for.Dustbin is a plastic container where everyone can dispose their waste. Dustbin is used as a storage place to dispose waste, but we cannot estimate the exact amount of waste disposed by a society, and the dustbin cannot take more waste as the space should be available in it to take more. We need to know the level of waste in the dustbin and based on that we can intimate people to use the dustbin or not. In this Smart Dustbin project, we have designed a prototype where the lid of the dustbin is opened, on detection of human hand and waste, and the level of waste available inside the dustbin is sent as notification in the form of LED. The main components we used in making this prototype are Arduino, NODEMCU, Servo Motor and Ultrasonic Sensors. The software component is the application named as Blynk which is used to get notification. This dustbin can be a start to Smart Waste Management System where the officials can clean or empty the dustbin which depends on the notification received by them and not waiting for a call from a person of a society who informs the garbage trucks to come and take the waste from them.

**COMPONENTS USED IN WASTE MANAGEMENT**

* ESP32
* ULTRASONIC SENSOR
* SERVO MOTOR
* SOIL MOISTURE BREAKOUT

**ESP32 :**

****

ESP32 can perform as a complete standalone system or as a slave device to a host MCU, reducing communication stack overhead on the main application processor. ESP32 can interface with other systems to provide Wi-Fi and Bluetooth functionality through its SPI / SDIO or I2C / UART interfaces.

**ULTRASONIC SENSOR:**

****

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible sound (i.e. the sound that humans can hear). Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has travelled to and from the target).

**SERVO MOTOR:**

****

A servomotor (or servo motor or simply servo(to be differentiated from [servomechanism](https://en.wikipedia.org/wiki/Servomechanism), which may also be called a servo)) is a [rotary actuator](https://en.wikipedia.org/wiki/Rotary_actuator) or [linear actuator](https://en.wikipedia.org/wiki/Linear_actuator) that allows for precise control of angular or linear position, velocity, and acceleration in a [mechanical system](https://en.wikipedia.org/wiki/Mechanical_system). It consists of a suitable [motor](https://en.wikipedia.org/wiki/Motor) coupled to a [sensor](https://en.wikipedia.org/wiki/Sensor) for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.Servomotors are not a specific class of motor, although the term *servomotor* is often used to refer to a motor suitable for use in a [closed-loop control](https://en.wikipedia.org/wiki/Closed-loop_control) system.Servomotors are used in applications such as [robotics](https://en.wikipedia.org/wiki/Robotics), [CNC machinery](https://en.wikipedia.org/wiki/CNC_machine), and [automated manufacturing](https://en.wikipedia.org/wiki/Automated_manufacturing).

**SOIL MOISTURE BREAKOUT:**

****

The Moisture & Humidity breakout measures soil moisture, temperature and humidity. The breakout features the DHT11 humidity sensor for accurate readings of ambient temperature and moisture. The Moisture/Humidity breakout also features two probes used for measuring the moisture levels in soil. Add smarts to your next botanical Arduino project.

**SIMULATION STEPS:**

STEP 1:Access Wowki

* Go to the website(<https://wowki.com>)

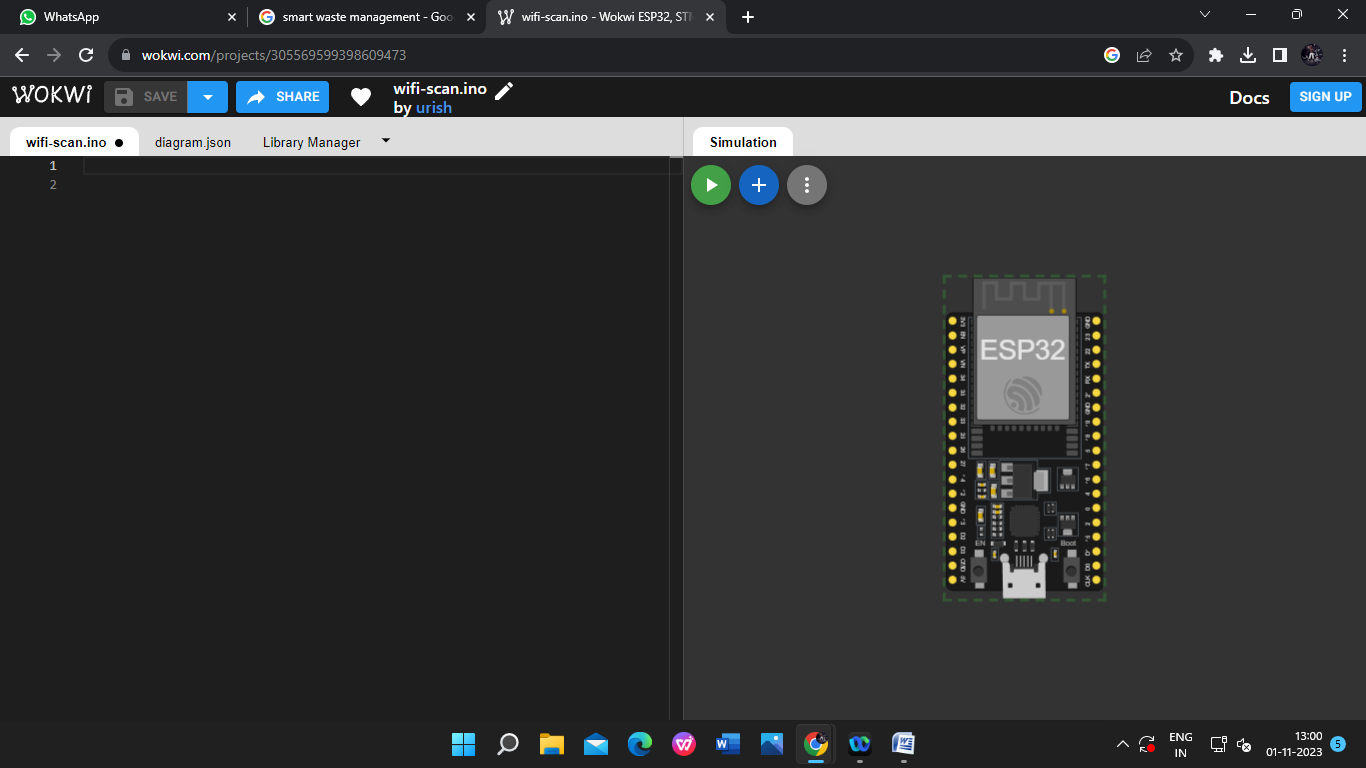
STEP 2: Create a Project

* Click on the new project

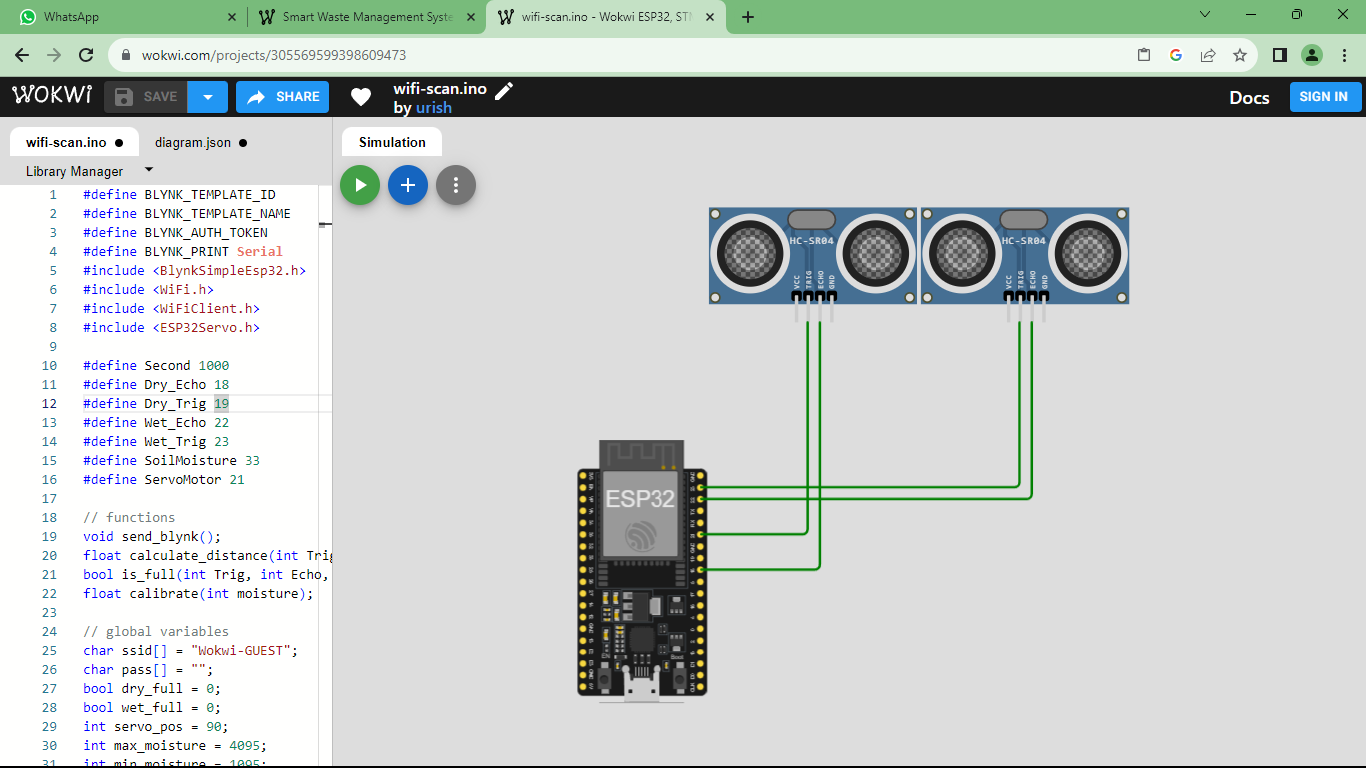
STEP 3: Add component

* In the component panel search for a “ESP32”

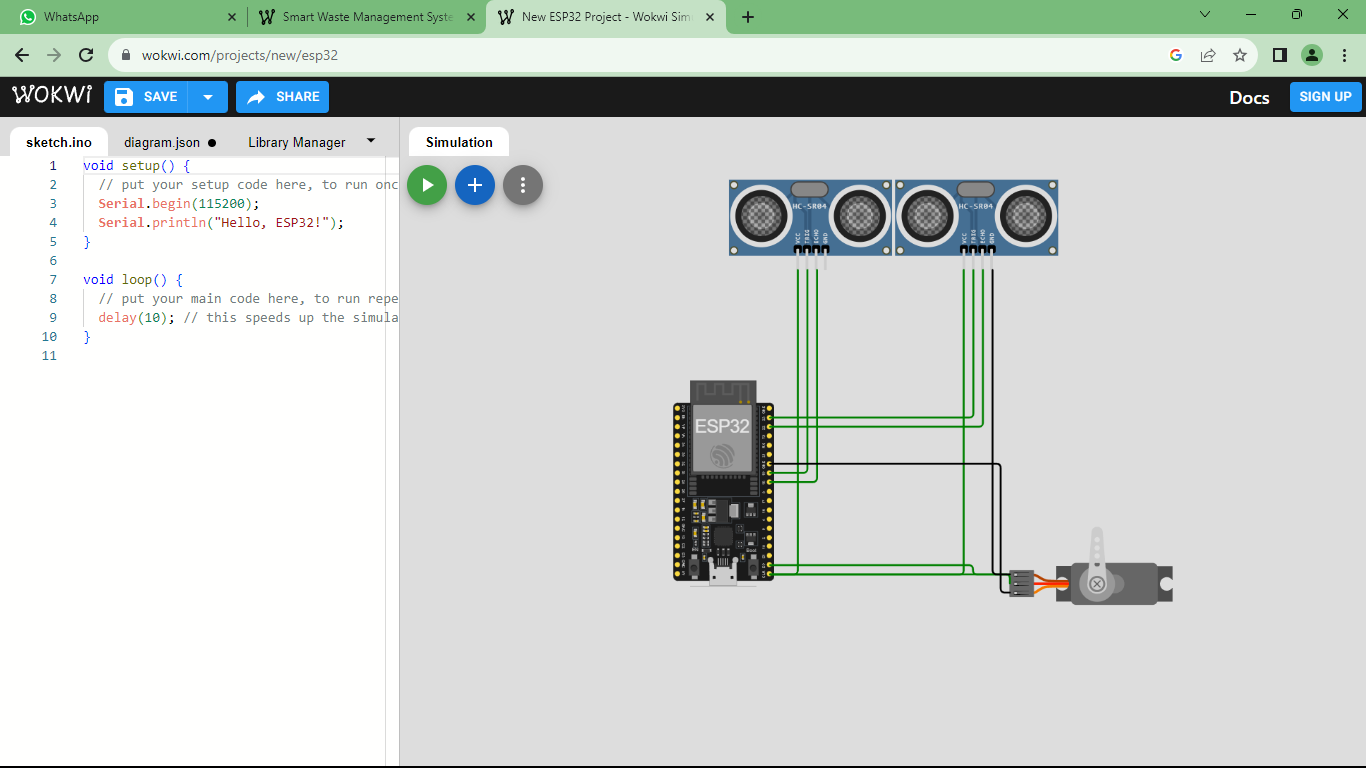
and drag it onto the virtual breadboard.

****

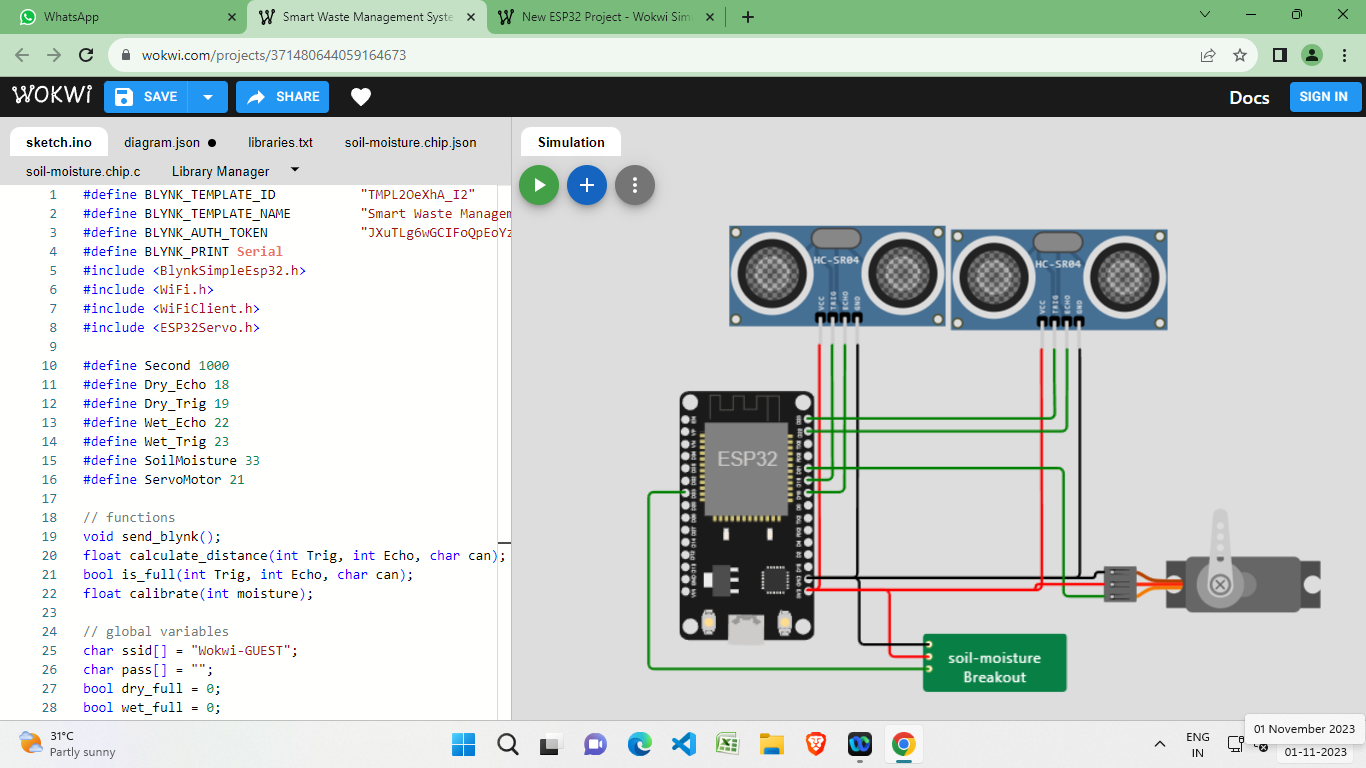
STEP 4: Add two Ultrasonic sensors and make the connection with gthe ESP32.

****

STEP5: Add the Servo motor to the virtual board and make the required connections.



STEP 6: Now add the soil-moisture break out to the virtual board.



**STEP 7: CODE**

#define BLYNK\_TEMPLATE\_ID “TMPL20eXhA\_I2”

#define BLYNK\_TEMPLATE\_NAME "Smart Waste Management System"

#define BLYNK\_AUTH\_TOKEN "JXuTLg6wGCIFoQpEoYzRzgWtFwD1LD9s"

#define BLYNK\_PRINT Serial

#include <BlynkSimpleEsp32.h>

#include <WiFi.h>

#include <WiFiClient.h>

#include <ESP32Servo.h>

#define Second 1000

#define Dry\_Echo 18

#define Dry\_Trig 19

#define Wet\_Echo 22

#define Wet\_Trig 23

#define SoilMoisture 33

#define ServoMotor 21

// functions

void send\_blynk();

float calculate\_distance(int Trig, int Echo, char can);

bool is\_full(int Trig, int Echo, char can);

float calibrate(int moisture);

// global variables

char ssid[] = "Wokwi-GUEST";

char pass[] = "";

bool dry\_full = 0;

bool wet\_full = 0;

int servo\_pos = 90;

int max\_moisture = 4095;

int min\_moisture = 1095;

float threshold = calibrate(4000);

BlynkTimer timer;

Servo servo;

void setup() {

Serial.begin(115200);

Blynk.begin(BLYNK\_AUTH\_TOKEN, ssid, pass);

pinMode(Dry\_Trig, OUTPUT);

pinMode(Dry\_Echo, INPUT);

pinMode(Wet\_Trig, OUTPUT);

pinMode(Wet\_Echo, INPUT);

pinMode(SoilMoisture, INPUT);

servo.attach(ServoMotor, 500, 2400);

timer.setInterval(1000, send\_blynk);

}

void loop() {

Serial.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

// manage trash

int moisture\_value = analogRead(SoilMoisture);

float moisture = calibrate(moisture\_value);

Serial.print("Moisture : ");

Serial.print(moisture\_value);

Serial.print(" ----> calibrated : ");

Serial.print(moisture);

Serial.println("%");

bool wet\_trash = moisture > threshold;

if(wet\_trash){

Serial.print("The trash is wet! (threshold = ");

Serial.print(threshold);

Serial.println("%)");

// open gate

for (servo\_pos = 90; servo\_pos >= 0; servo\_pos -= 1) {

servo.write(servo\_pos);

delay(15);

}

delay(5 \* Second);

// close gate

for (servo\_pos = 0; servo\_pos <= 90; servo\_pos += 1) {

servo.write(servo\_pos);

delay(15);

}

}

// manage can

dry\_full = is\_full(Dry\_Trig, Dry\_Echo, 'D');

wet\_full = is\_full(Wet\_Trig, Wet\_Echo, 'W');

if (dry\_full){

Serial.println("The dry can is full.");

}

if (wet\_full){

Serial.println("The wet can is full.");

}

Blynk.run();

timer.run();

delay(10 \* Second);

}

void send\_blynk(){

if (wet\_full){

Blynk.logEvent("wet\_can\_is\_full") ;

Blynk.virtualWrite(V1, 1);

}else{

Blynk.virtualWrite(V1, 0);

}

if (dry\_full){

Blynk.logEvent("dry\_can\_is\_full") ;

Blynk.virtualWrite(V0, 1);

}else{

Blynk.virtualWrite(V0, 0);

}

}

float calculate\_distance(int Trig, int Echo, char can){

digitalWrite(Trig, LOW);

delay(2);

digitalWrite(Trig, HIGH);

delay(10);

digitalWrite(Trig, LOW);

int duration = pulseIn(Echo, HIGH);

float distance = duration \* 0.034 /2;

Serial.print(can);

Serial.print(" free space : ");

Serial.print(distance);

Serial.println(" CM");

return distance;

}

bool is\_full(int Trig, int Echo, char can){

float distance = calculate\_distance(Trig, Echo, can);

return distance < 4;

}

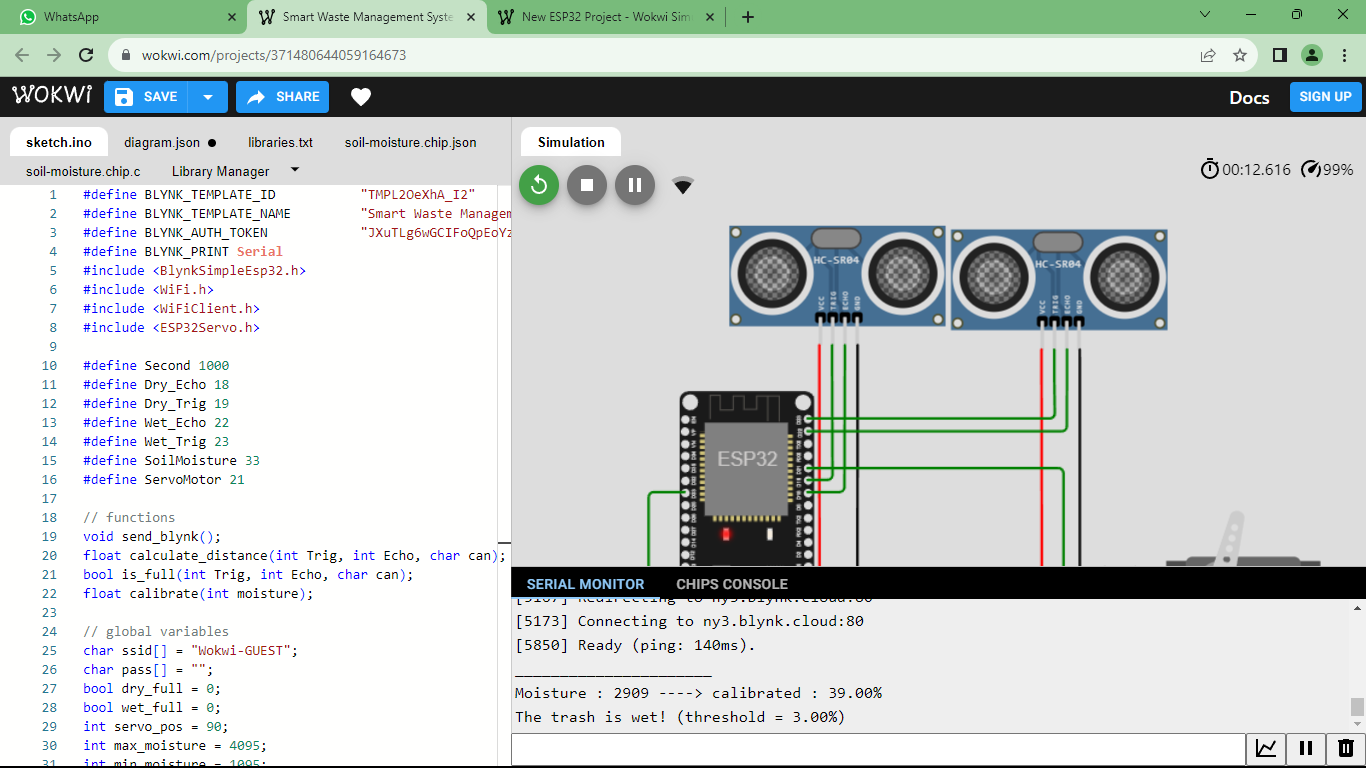
float calibrate(int moisture){

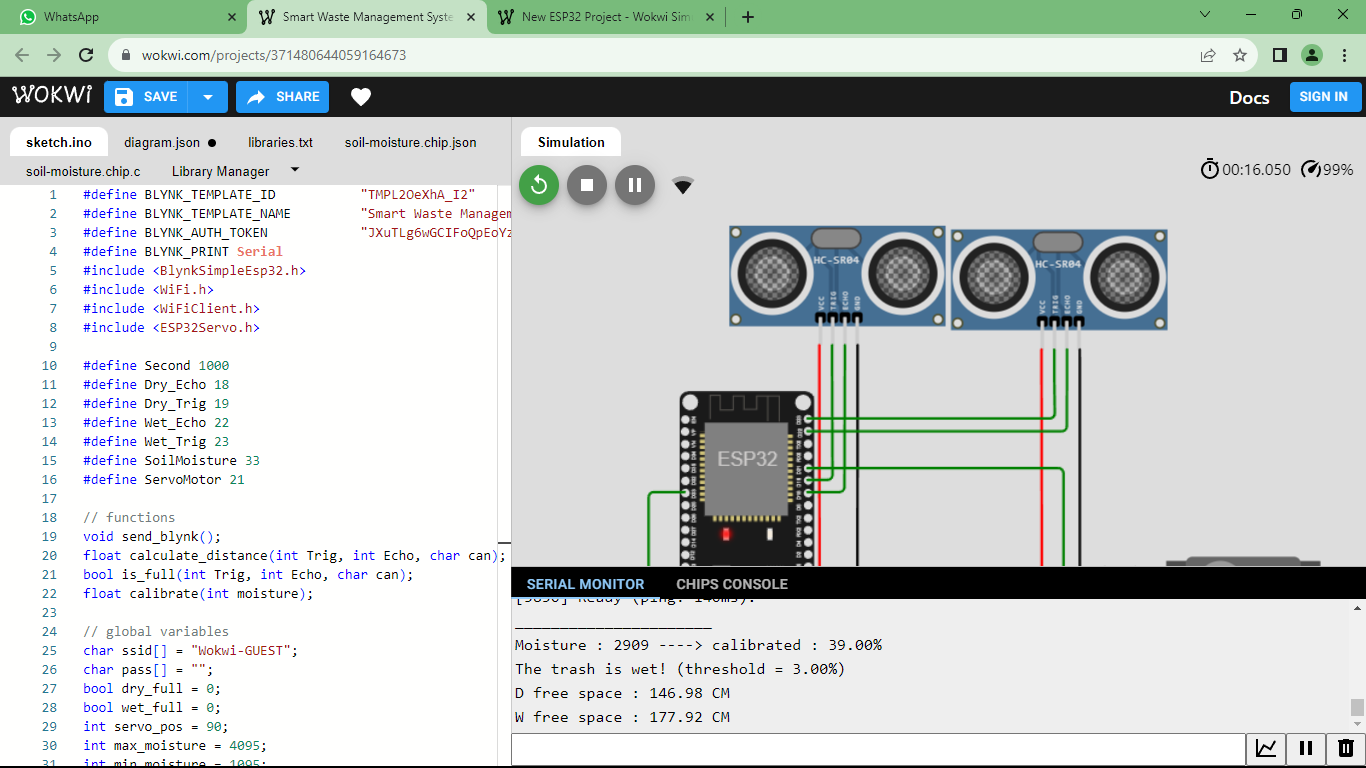
int x = (max\_moisture - min\_moisture)/100;

return (max\_moisture - moisture)/x;

}

STEP 8: Simulation





**WEB APPLICATION FOR SMART WASTE MANAGEMENT**

**CODE:**

<!DOCTYPE html>

<html>

<head>

<title>Smart Dustbin</title>

<style>

/\* Basic CSS for layout \*/

body {

font-family: Arial, sans-serif;

text-align: center;

}

.dustbin {

width: 300px;

margin: 250px auto;

}

.lid {

width: 100%;

height: 50px;

background-color: grey;

border-radius: 10px 10px 0 0;

}

.bin {

width: 100%;

height: 150px;

background-color: red;

border-radius: 0 0 10px 10px;

}

</style>

</head>

<body>

<div class="dustbin">

<div class="lid"></div>

<div class="bin"></div>

</div>

<script>

// JavaScript for simulating filling the dustbin

let binContent = 0; // Initially empty

function throwTrash() {

binContent += 10; // Increase the content by 10 units (you can adjust this as needed)

if (binContent >= 100) {

alert("Bin is full! Please empty it.");

binContent = 100; // Limit the content to a maximum of 100

}

document.querySelector('.bin').style.height = binContent + "px";

}

</script>

<button onclick="throwTrash()">Throw Trash</button>

<head>

<title>Smart Dustbin Moisture Content</title>

</head>

<body>

<h1>Moisture Content of Smart Dustbin</h1>

<p>Moisture Content: <span id="moistureValue">Fetching...</span></p>

<script>

function getMoistureData() {

// Simulated data for demonstration

const moisture = Math.floor(Math.random() \* 100); // Replace with actual moisture sensor data

return moisture;

}

function updateMoistureValue() {

const moisture = getMoistureData();

document.getElementById('moistureValue').textContent = moisture + "%";

}

updateMoistureValue(); // Initial call

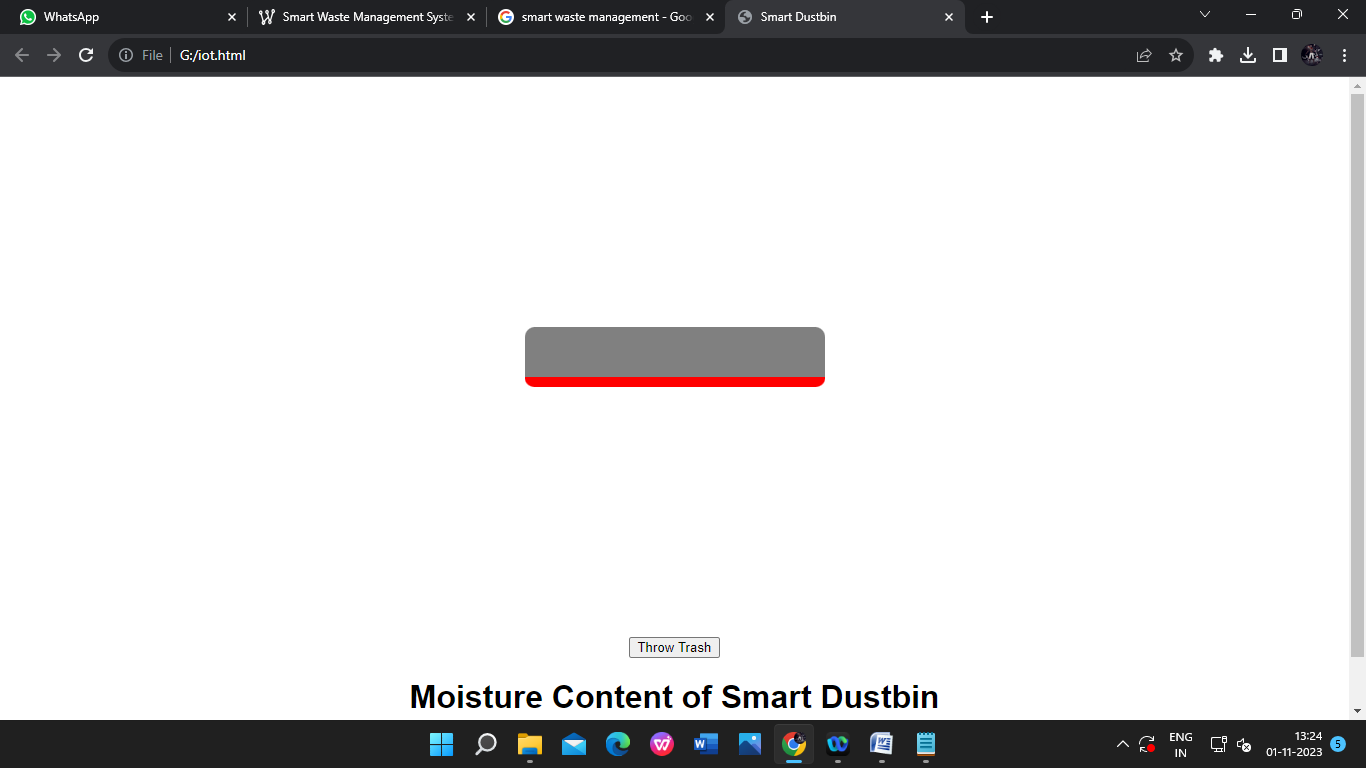
</script>

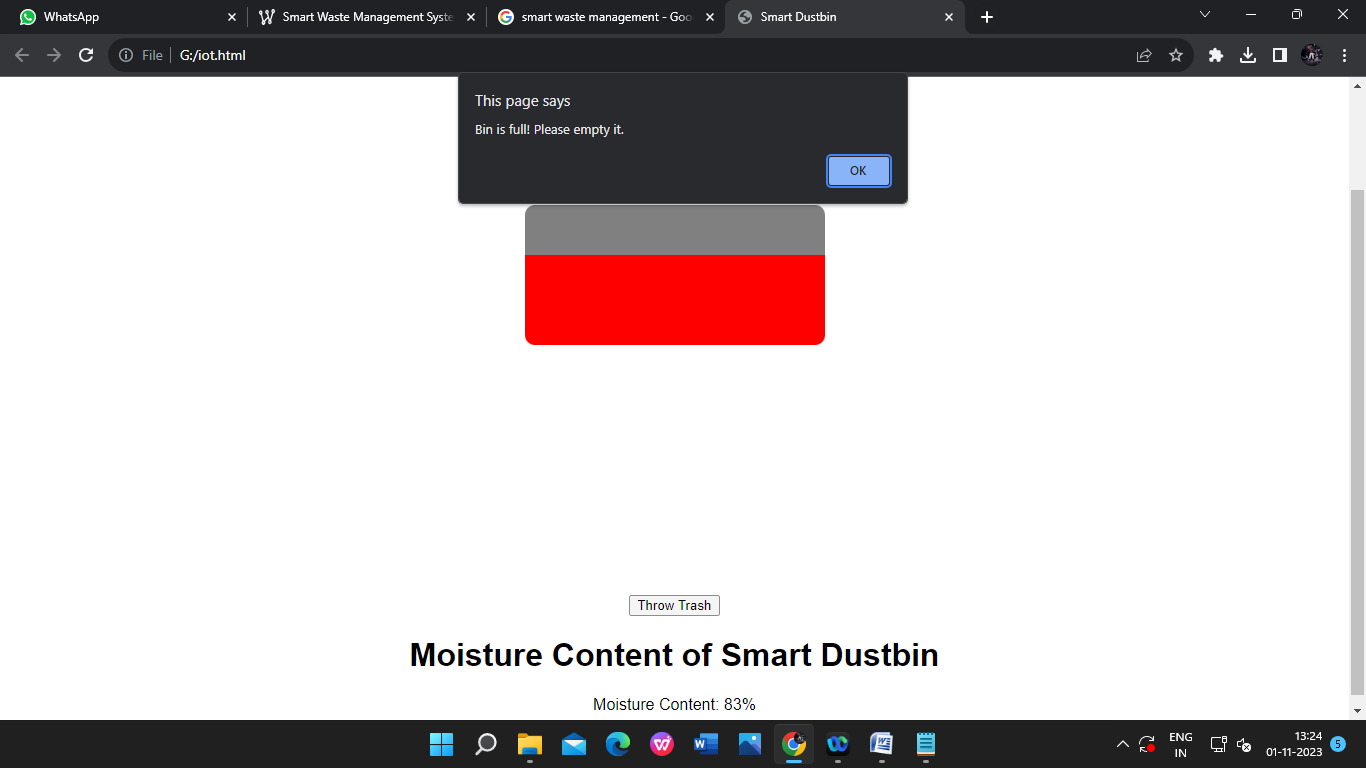
</body>

</body>

</html>

**OUTPUT:**

****

****

**CONCLUSION :**

IOT based Dustbins help the people to manage the waste easily and help them reduce the work of calling or waiting for the specific person to make the area clean and makes a diseases and the people will be fit and are not prone to diseases caused by these waste materials. The mission Swachh Bharat can also be implemented easily. This system assures the cleaning of dustbins soon when the garbage level reaches its maximum. It will take power supply with the help of Battery. If the dustbin is not cleaned in specific time, then the record is sent to the Sweeper or higher authority who can take appropriate action against the concerned contractor. It

ultimately helps in keeping the surrounding clean and the waste management can be much easier.