**Smart public restroom**

**Phase 4: Development Part 2**

**Introduction:**

We creating smart water management with Arduino UNO & ESP 32 in Wokwi simulator in this project, we’ll create a simple simulation of Availability and cleanliness.

**Components Required:**

* Led

# Hc-sr04

# Servo

# ESP32 Simulation

* Power Source

**Standard 5mm LED:**



## Pin names[​](https://docs.wokwi.com/parts/wokwi-led#pin-names)

| **Name** | **Description** |
| --- | --- |
| A | Anode (positive pin) |
| C | Cathode (negative pin) |

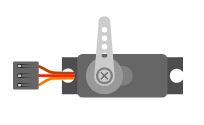
**HC-SR04 Ultrasonic Distance Sensor:**

## 

## Pin names[​](https://docs.wokwi.com/parts/wokwi-hc-sr04#pin-names)

| **Name** | **Description** |
| --- | --- |
| VCC | Voltage supply (5V) |
| TRIG | Pulse to start the measurement |
| ECHO | Measure the high pulse length to get the distance |
| GND | Ground |

**Standard Micro Servo Motor:**

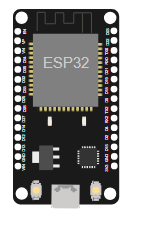


## Pin names[​](https://docs.wokwi.com/parts/wokwi-servo#pin-names)

| **Name** | **Description** |
| --- | --- |
| PWM | Servo control signal |
| V+ | Positive voltage (5V) |
| GND | Ground |

# ESP32 Simulation:

The ESP32 is a popular WiFi and Bluetooth-enabled microcontroller, widely used for IoT Projects.



## ESP32 boards[​](https://docs.wokwi.com/guides/esp32#esp32-boards)

| **Name** | **Chip** | **Description** |
| --- | --- | --- |
| ESP32 DevKit v1 | ESP32 | Popular ESP32 development board |
| ESP32-S2-DevKitM-1 | ESP32-S2 | Entry-level ESP32-S2 development board |
| [Franzininho WiFi](https://docs.wokwi.com/parts/board-franzininho-wifi) | ESP32-S2 | Board by the Franzininho Community |
| [Wemos S2 mini](https://wokwi.com/projects/355047217294313473) | ESP32-S2 | Small ESP32-S2 board by Wemos |
| ESP32-S3-DevKitC-1 | ESP32-S3 | Entry-level ESP32-S3 development board |
| ESP32-C3-DevKitM-1 | ESP32-C3 | Entry-level ESP32-C3 development board |
| Rust Board ESP32-C3 | ESP32-C3 | ESP32-C3 board designed for [Rust trainings](https://github.com/esp-rs/std-training) |
| ESP32-C6-DevKitC-1 | ESP32-C6 | Entry-level ESP32-C6 development board (beta) |
| ESP32-H2-DevKitM-1 | ESP32-H2 | Entry-level ESP32-H2 development board (alpha) |

**The Program Execution Code:**

#include<ESP32Servo.h>

#define TRIGGERPIN 32

#define ECHOPIN    35

#define RED\_LED    33

#define GREEN\_LED  25

Servo servo\_1;

long duration;

int pos, distance, i=0;

void setup()

{

  servo\_1.attach(18);

  Serial.begin(115200);

  pinMode(TRIGGERPIN, OUTPUT);

  pinMode(ECHOPIN, INPUT);

  pinMode(RED\_LED, OUTPUT);

  pinMode(GREEN\_LED, OUTPUT);

  Serial.println(" ");

  Serial.println("Sensing the Height");

  digitalWrite(RED\_LED, HIGH);

  digitalWrite(GREEN\_LED, LOW);

  pos = 0;

  servo\_1.write(pos);

}

void loop()

{

  digitalWrite(TRIGGERPIN, LOW);

  delayMicroseconds(3);

  digitalWrite(TRIGGERPIN, HIGH);

  delayMicroseconds(12); // it may be 10 us

  digitalWrite(TRIGGERPIN, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

  duration = pulseIn(ECHOPIN, HIGH);

// Calculating the distance

  distance = (duration/2) / 29.1;

  // for Adult

  if (distance >= 100 && distance <= 150)

    {

      i = 1;

      if (pos != 180)

      {

        servo\_1.write(180);

        pos = 180;

        i = 1;

      }

    }

  // for Child

    else if (distance >= 200 && distance <= 250)

      {

        i = 1;

        if (pos != 0)

        {

         servo\_1.write(0);

         pos = 0;

         i = 1;

        }

      }

    else if (distance > 300 && i == 1)

      {

        digitalWrite(RED\_LED, LOW);

        digitalWrite(GREEN\_LED, HIGH);

        delay(5000);

        digitalWrite(RED\_LED, HIGH);

        digitalWrite(GREEN\_LED, LOW);

        i = 0;

      }

       delay (500);

  Serial.println(" ");

  Serial.print("Free Level : ");

  Serial.print(distance);

  Serial.print("   ");

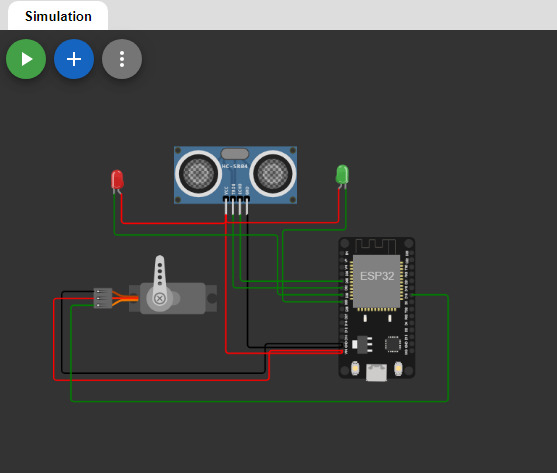
  Serial.print("Position : ");

  Serial.print(pos);

  delay (500);

}

**Simulation output :**



**Web code:**

**html**

<!DOCTYPE html>

<html>

<head>

<title>Restroom Information</title>

<link rel="stylesheet" type="text/css" href="styles.css">

</head>

<body>

<h1>Restroom Information</h1>

<div id="restroom-data">

<p>Availability: <span id="availability">Available</span></p>

<p>Cleanliness: <span id="cleanliness">Clean</span></p>

</div>

<script src="script.js"></script>

</body>

</html>

**css**

/\* Add CSS styling here \*/

body {

font-family: Arial, sans-serif;

text-align: center;

background-color: #f0f0f0;

}

h1 {

color: #333;

}

#restroom-data {

background-color: #fff;

padding: 20px;

border-radius: 5px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);

margin: 20px auto;

width: 300px;

}

span {

font-weight: bold;

}

**javascript**

// JavaScript for fetching and updating real-time data

function fetchRestroomData() {

// Simulate fetching data from a server (replace with actual data retrieval logic)

const data = {

availability: "Available",

cleanliness: "Clean"

};

// Update the HTML with real-time data

document.getElementById("availability").textContent = data.availability;

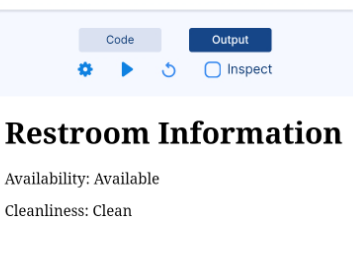
document.getElementById("cleanliness").textContent = data.cleanliness;

}

// Fetch data on page load

fetchRestroomData();

**Web code output:**



**Conclusion:**

smart public restrooms have the potential to revolutionize the way we use and manage public restroom facilities. They can provide a more convenient and comfortable experience for users, promote hygiene, and contribute to environmental sustainability. To successfully implement these facilities, it is essential to address design, privacy, security, and maintenance issues. Additionally, public and private stakeholders should work together to ensure that these facilities are accessible and cater to the diverse needs of the community.