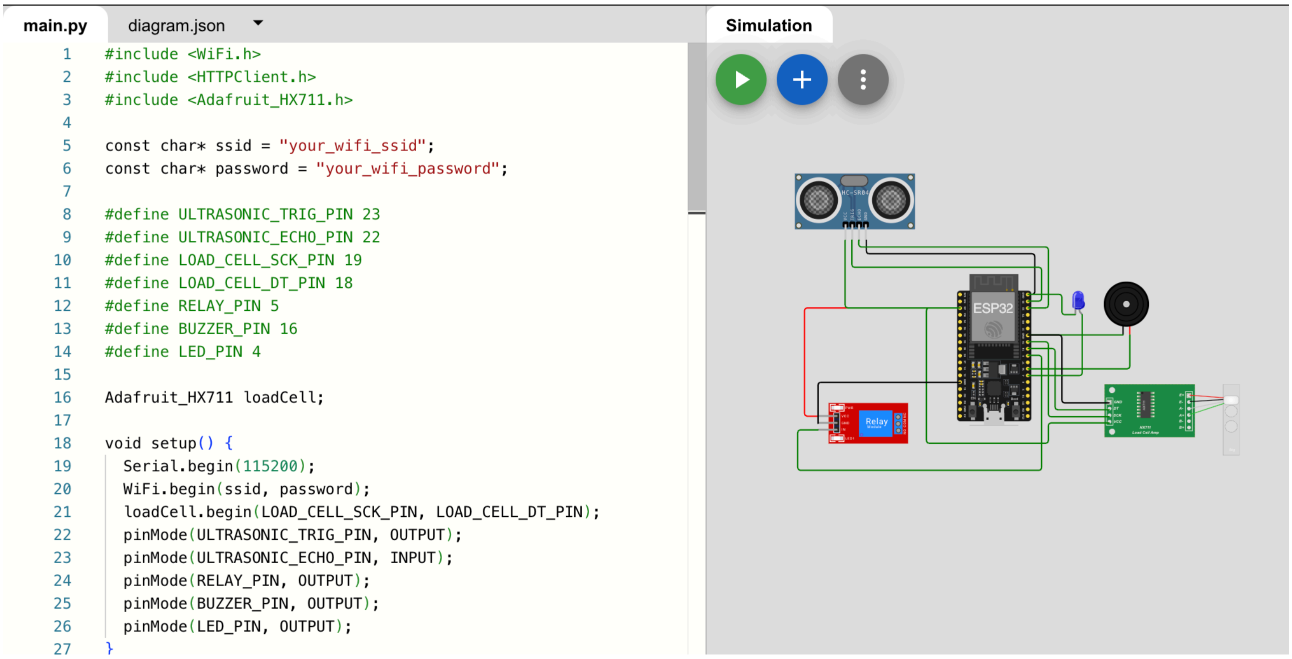
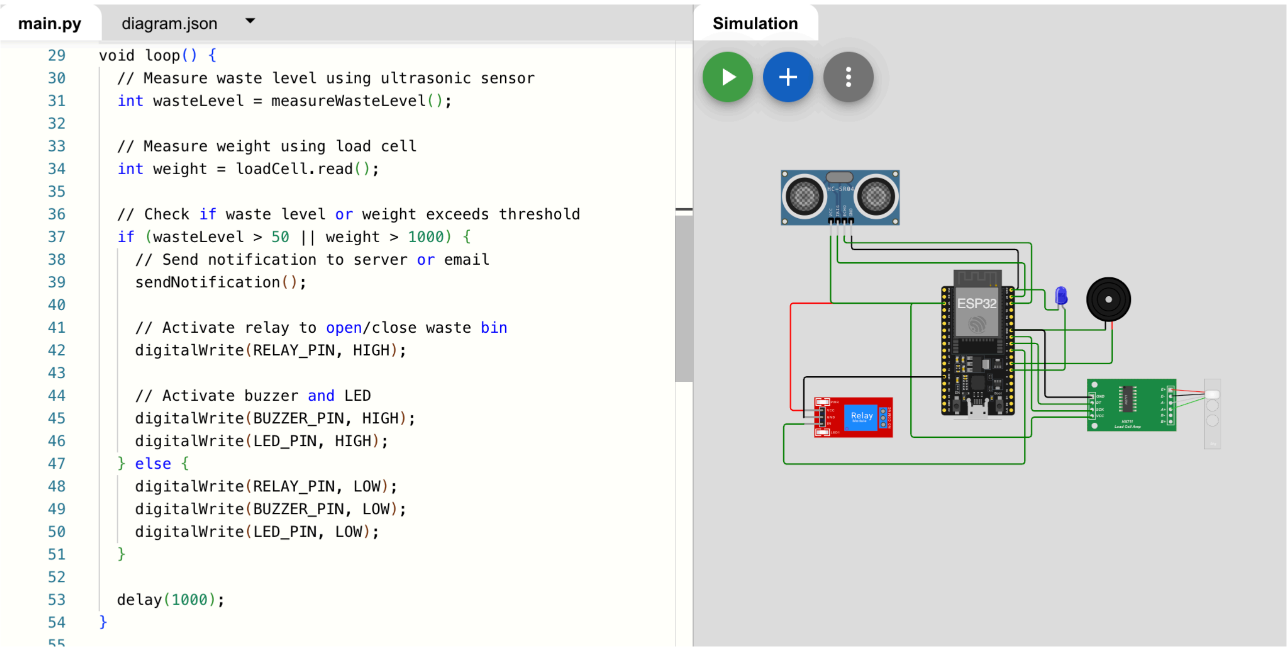
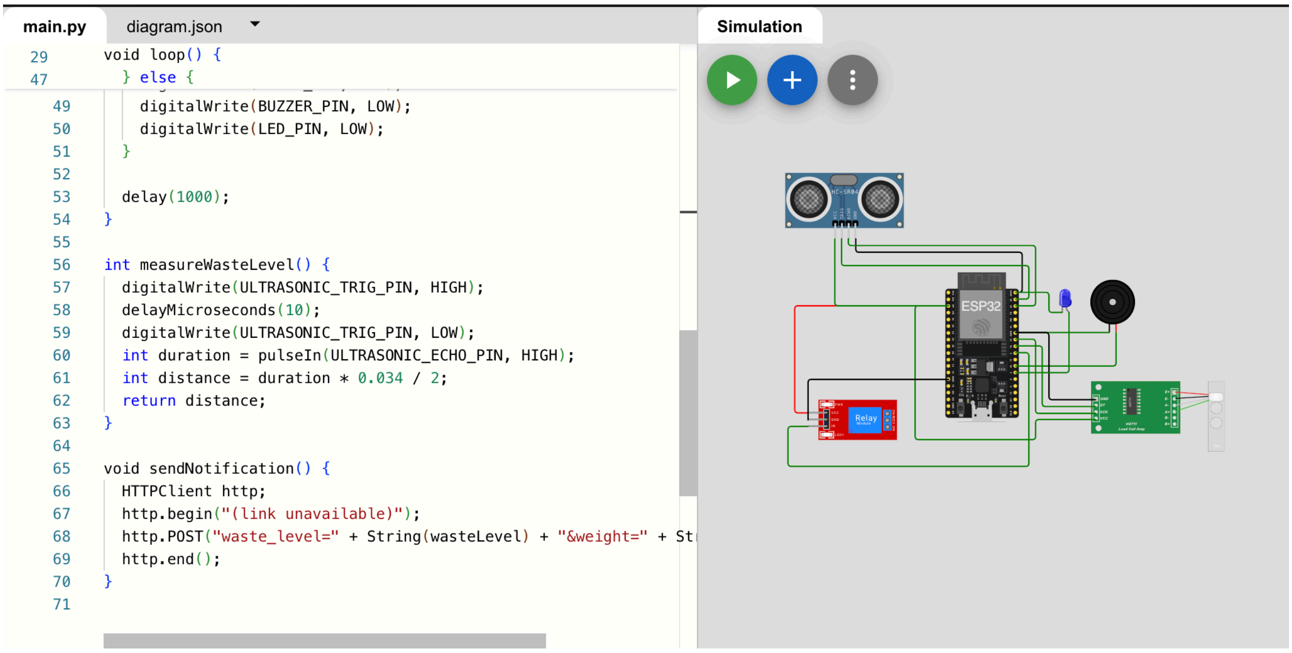
main.py:



```
#include <WiFi.h>
#include <HTTPClient.h>
#include <Adafruit_HX711.h>

const char* ssid = "your_wifi_ssid";
const char* password = "your_wifi_password";

#define ULTRASONIC_TRIG_PIN 23
#define ULTRASONIC_ECHO_PIN 22
#define LOAD_CELL_SCK_PIN 19
#define LOAD_CELL_DT_PIN 18
#define RELAY_PIN 5
#define BUZZER_PIN 16
#define LED_PIN 4

Adafruit_HX711 loadCell;

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  loadCell.begin(LOAD_CELL_SCK_PIN, LOAD_CELL_DT_PIN);
  pinMode(ULTRASONIC_TRIG_PIN, OUTPUT);
  pinMode(ULTRASONIC_ECHO_PIN, INPUT);
  pinMode(RELAY_PIN, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(LED_PIN, OUTPUT);
}
```

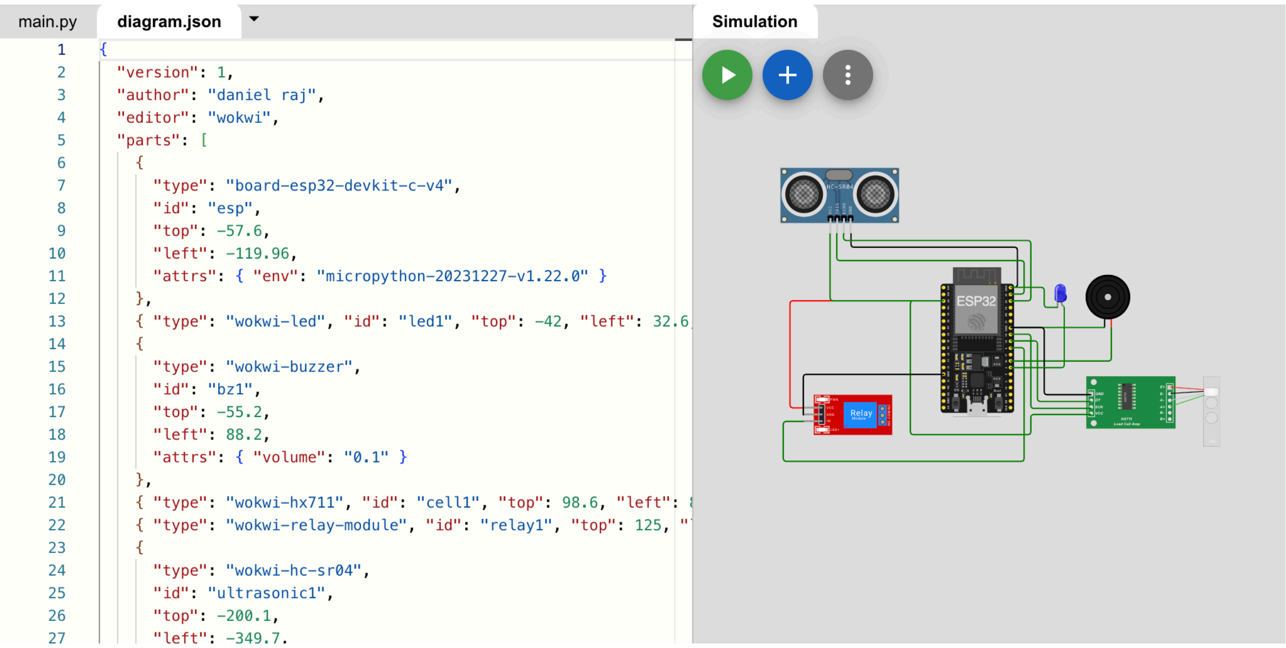


```
void loop() {
  // Measure waste level using ultrasonic sensor
  int wasteLevel = measureWasteLevel();

  // Measure weight using load cell
  int weight = loadCell.read();

  // Check if waste level or weight exceeds threshold
  if (wasteLevel > 50 || weight > 1000) {
    // Send notification to server or email
    sendNotification();

    // Activate relay to open/close waste bin
    digitalWrite(RELAY_PIN, HIGH);

    // Activate buzzer and LED
    digitalWrite(BUZZER_PIN, HIGH);
    digitalWrite(LED_PIN, HIGH);
  } else {
    digitalWrite(RELAY_PIN, LOW);
    digitalWrite(BUZZER_PIN, LOW);
    digitalWrite(LED_PIN, LOW);
  }

  delay(1000);
}
```

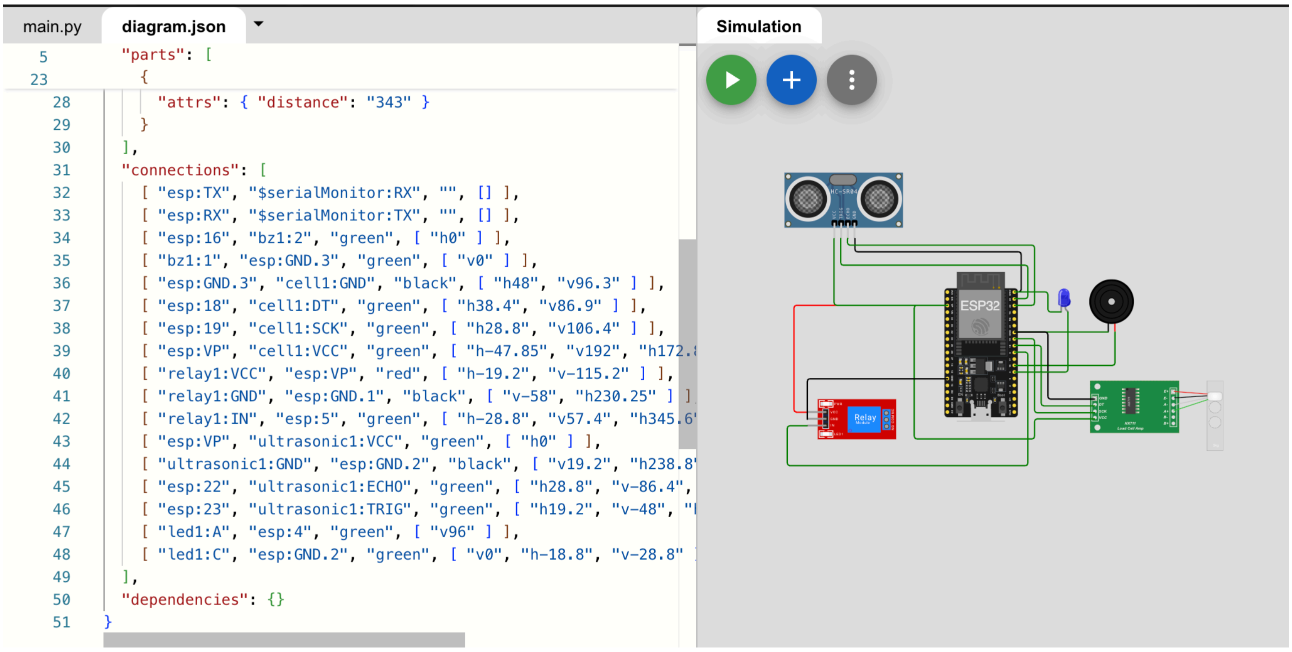


```
void loop() {
  } else {
    digitalWrite(BUZZER_PIN, LOW);
    digitalWrite(LED_PIN, LOW);
  }

  delay(1000);
}

int measureWasteLevel() {
  digitalWrite(ULTRASONIC_TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(ULTRASONIC_TRIG_PIN, LOW);
  int duration = pulseIn(ULTRASONIC_ECHO_PIN, HIGH);
  int distance = duration * 0.034 / 2;
  return distance;
}

void sendNotification() {
  HTTPClient http;
  http.begin("(link unavailable)");
  http.POST("waste_level=" + String(wasteLevel) + "&weight=" + St
  http.end();
}
```

diagram.json:

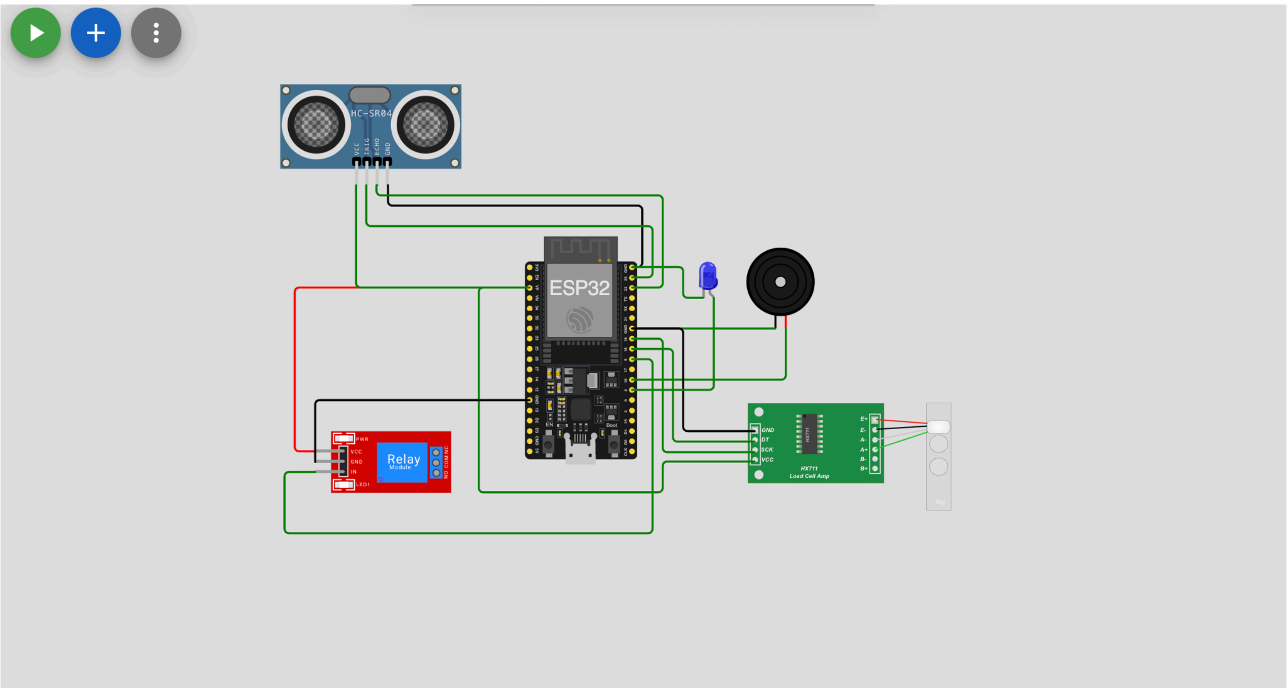


```
{
  "version": 1,
  "author": "daniel raj",
  "editor": "wokwi",
  "parts": [
    {
      "type": "board-esp32-devkit-c-v4",
      "id": "esp",
      "top": -57.6,
      "left": -119.96,
      "attrs": { "env": "micropython-20231227-v1.22.0" }
    },
    { "type": "wokwi-led", "id": "led1", "top": -42, "left": 32.6
    {
      "type": "wokwi-buzzer",
      "id": "bz1",
      "top": -55.2,
      "left": 88.2,
      "attrs": { "volume": "0.1" }
    },
    { "type": "wokwi-hx711", "id": "cell1", "top": 98.6, "left": 
    { "type": "wokwi-relay-module", "id": "relay1", "top": 125, "
    {
      "type": "wokwi-hc-sr04",
      "id": "ultrasonic1",
      "top": -200.1,
      "left": -349.7,
```



```
  "parts": [
    {
      "attrs": { "distance": "343" }
    }
  ],
  "connections": [
    [ "esp:TX", "$serialMonitor:RX", "", [] ],
    [ "esp:RX", "$serialMonitor:TX", "", [] ],
    [ "esp:16", "bz1:2", "green", [ "h0" ] ],
    [ "bz1:1", "esp:GND.3", "green", [ "v0" ] ],
    [ "esp:GND.3", "cell1:GND", "black", [ "h48", "v96.3" ] ],
    [ "esp:18", "cell1:DT", "green", [ "h38.4", "v86.9" ] ],
    [ "esp:19", "cell1:SCK", "green", [ "h28.8", "v106.4" ] ],
    [ "esp:VP", "cell1:VCC", "green", [ "h-47.85", "v192", "h172.
    [ "relay1:VCC", "esp:VP", "red", [ "h-19.2", "v-115.2" ] ],
    [ "relay1:GND", "esp:GND.1", "black", [ "v-58", "h230.25" ] ]
    [ "relay1:IN", "esp:5", "green", [ "h-28.8", "v57.4", "h345.6
    [ "esp:VP", "ultrasonic1:VCC", "green", [ "h0" ] ],
    [ "ultrasonic1:GND", "esp:GND.2", "black", [ "v19.2", "h238.8
    [ "esp:22", "ultrasonic1:ECHO", "green", [ "h28.8", "v-86.4",
    [ "esp:23", "ultrasonic1:TRIG", "green", [ "h19.2", "v-48", "
    [ "led1:A", "esp:4", "green", [ "v96" ] ],
    [ "led1:C", "esp:GND.2", "green", [ "v0", "h-18.8", "v-28.8"
  ],
  "dependencies": {}
}
```

Output:


HC-SR04
VCC
TRIG
ECHO
GND
ESP32
Relay
Module
PWR
VCC
GND
IN
LED1
NO
COM
NC
GND
DT
SCK
VCC
HX711
Load Cell Amp
E+
E-
A-
A+
B-
B+