ENVIRONMENTAL MONITORING SYSTEM

PROPOSED SYSTEM :

•The program for NodeMCU can be written in any programming language. The Arduino software provides a better Integrated Development Environment (IDE) for programming the NodeMCU. It is a cross-platform application written in Java.

•This software consists of various features which include code editor, text cutting and pasting, replacing text and searching, brace matching, automatic indenting, and syntax highlighting.

•It usually consists of a central microcontroller to which other objects are connected. The smart garden consists of NodeMCU as a hub to which different types of sensors such as moisture sensor, humidity sensor, temperature sensor and ultrasonic sensor are connected.

• The ultrasonic sensor is connected to a water tank which indicated the level of water in the tank. Other sensors are connected to their respective positions and these sensors send the data to NodeMCU which consists of an inbuilt Wi-Fi technology.

NODEMCU ;

platform. It runs o•Smart Garden includes NodeMCU as a hub. NodeMCU is an open source IoT n ESP8266 Wi-Fi SoC from Espessif Systems, and hardware based on the ESP-12 module available at lowest cost.

•It is a Single – board microcontroller consists of 128kBytes of memory and 4Mbytes of storage. It was designed to for easyprogramming and allows easy prototyping for developers.

ARDUINO:

This software consists of various features which include code editor, text cutting and pasting, replacing text and searching, brace matching, automatic indenting, and syntax highlighting n.

•The program for NodeMCU can be written in any programming language. The Arduino software provides a better Integrated Development Environment (IDE) for programming

Coding:

import network

import time

from machine import Pin,ADC

import dht

import ujson

from umqtt.simple import MQTTClient

# MQTT Server Parameters

MQTT\_CLIENT\_ID = "micropython-weather-demo"

MQTT\_BROKER    = "broker.mqttdashboard.com"

MQTT\_USER      = ""

MQTT\_PASSWORD  = ""

MQTT\_TOPIC     = "wokwi-weather"

sensor = dht.DHT22(Pin(15))

MQ7=ADC(Pin(35))

MQ8=ADC(Pin(32))

button=Pin(34,Pin.IN)

led=Pin(33,Pin.OUT)

min\_rate=0

max\_rate=4095

print("Connecting to WiFi", end="")

sta\_if = network.WLAN(network.STA\_IF)

sta\_if.active(True)

whilenot sta\_if.isconnected():

  print(".", end sta\_if.connect('Wokwi-GU

EST', '')

="")

  time.sleep(0.1)

print(" Connected!")

print("Connecting to MQTT server... ", end="")

client = MQTTClient(MQTT\_CLIENT\_ID, MQTT\_BROKER, user=MQTT\_USER, password=MQTT\_PASSWORD)

client.connect()

print("Connected!")

prev\_weather = ""

whileTrue:

  CO\_sensor=(MQ7.read())\*100/(max\_rate)

  print("CO Sensor value: " + "%.2f" % CO\_sensor +"%")

  Hydrogen\_sensor=(MQ8.read())\*100/(max\_rate)

  print("Soil Sensor value: " + "%.2f" % Hydrogen\_sensor +"%")

  button\_value=button.value()

  if button\_value == True:

    led.value(000)

    print("It's Raining")

  else:

    led.value(0)

  print("Measuring weather conditions... ", end="")

  sensor.measure()

  message = ujson.dumps({

    "temp": sensor.temperature(),

    "humidity": sensor.humidity(),

  })

  if message != prev\_weather:

    print("Updated!")

    print("Reporting to MQTT topic {}: {}".format(MQTT\_TOPIC, message))

    client.publish(MQTT\_TOPIC, message)

    prev\_weather = message

  else:

    print("No change")

  time.sleep(1)