IBM NAN MUDHALVAN -INTERNET OF THINGS
(GROUP-4)

SMART WATER FOUNTAIN

PHASE-I Problem Definition and Design Thinking

ABSTRACT

- ✓ Water fountains have been a major tourist attraction these days which freeze the attention of tourists with their variety of lights, designs, and heights. And as we all know music holds a major part in our day-to-day lives.

- ✓ And hence our idea is to combine the beautiful water fountain with music which makes an extraordinary tourist attraction when constructively set with a range of frequencies that enables us to operate through various electronic devices.

- ✓ The project aims to enhance public water fountains by implementing IoT sensors to control water flow and detect malfunctions.

- ✓ The primary objective is to provide real-time information about water fountain status to residents through a public platform.

- ✓ This project includes defining objectives, designing the IoT sensor system, developing the water fountain status platform, and integrating them using IoT technology .

- ✓ Musical water fountain consists of Arduino UNO, sound sensor with external MIC, submersible motors, LCD, relay modules, sound generation using mobile, ARGB LED light strip &adapters

DESIGN THINKING:

- Sensor Integration
- Data Analytics and Processing
- User Interaction and Control

  Water Conservation and Sustainability
- Remote Maintenance and Diagnostics
- Security and Privacy

1. Sensor Integration

The Sensor Integration module is the foundation of our Smart Water Fountains system. It involves the installation of various sensors, such as water quality sensors, flow rate sensors, and proximity sensors, in and around the water fountain. These sensors continuously collect data, enabling real-time monitoring and analysis of water quality, usage patterns, and environmental conditions

2. Data Analytics and Processing

In this module, collected sensor data is processed and analyzed in real-time. Advanced data analytics algorithms are employed to detect anomalies in water quality, predict maintenance needs, and optimize water usage. Machine learning models can also be applied to identify usage trends, helping to improve fountain placement and resource allocation.

3. User Interaction and Control

The User Interaction and Control module offer an intuitive and user-friendly interface for both fountain users and administrators. Users can access the system via a dedicated mobile application or web portal to locate nearby fountains, check water quality, and receive real-time availability updates. Administrators gain control over fountain settings, allowing them to adjust water flow, monitor usage, and receive maintenance alerts.

4.Water Conservation and Sustainability

One of the primary goals of our Smart Water Fountains system is to promote water conservation and sustainability.
This module focuses on implementing water saving mechanisms, such as automatic shut off during low usage hours, intelligent scheduling, and feedback mechanisms to encourage responsible water consumption.

5. Remote Maintenance and Diagnostics

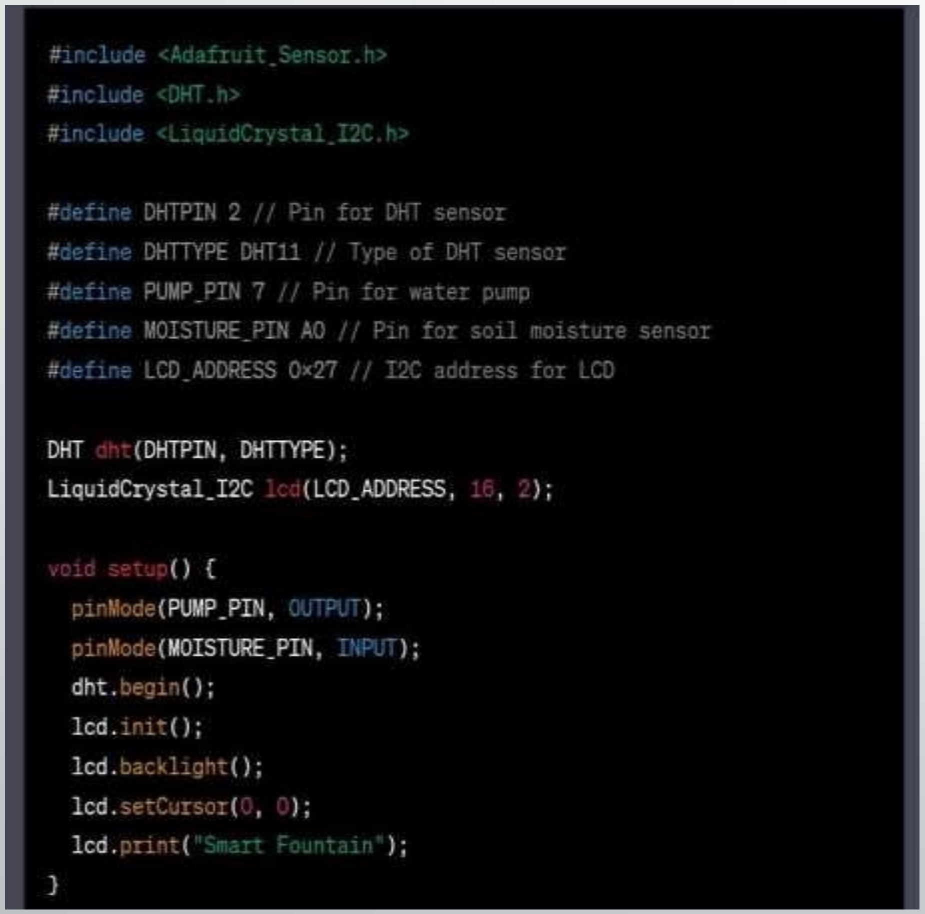
Ensure the continuous operation of the Smart Water Fountains, remote maintenance and diagnostics capabilities are integrated. Through IoT connectivity, administrators can remotely monitor fountain health, detect malfunctions, and schedule maintenance tasks. This proactive approach reduces downtime and enhances overall system reliability.

To

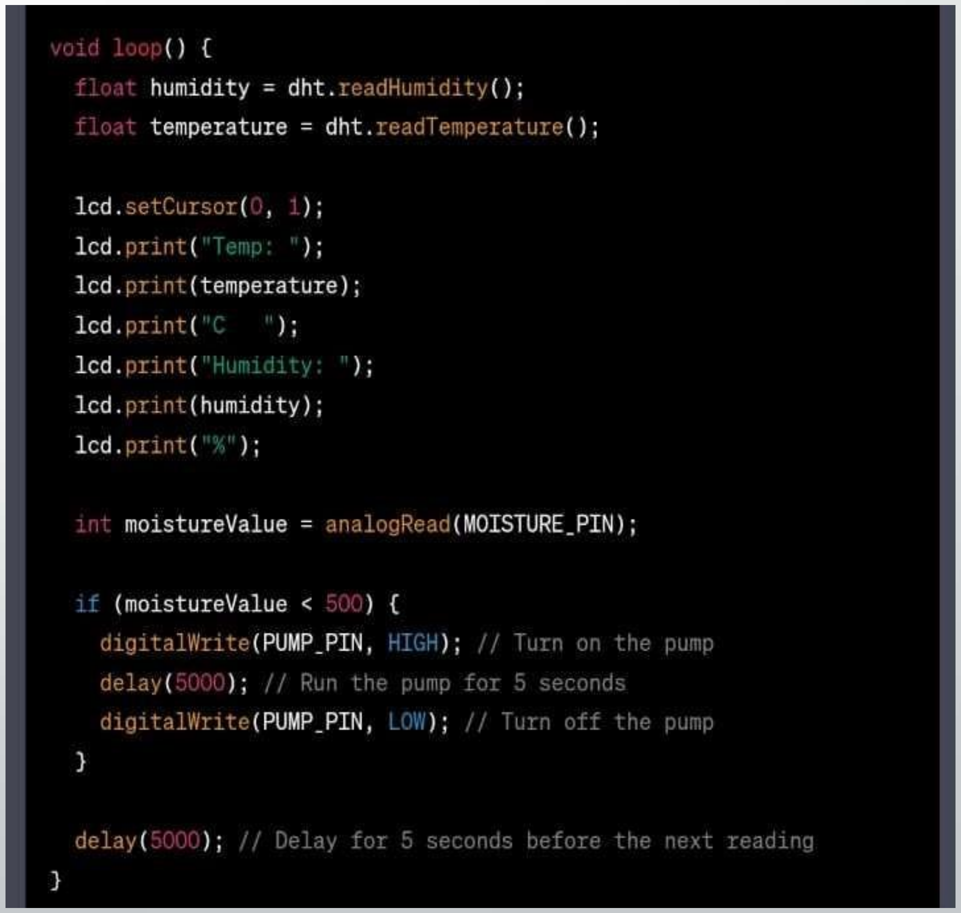
6. Security and Privacy

The Security and Privacy module is a critical aspect of our system. Robust security measures, including encryption, authentication, and access control, are implemented to protect user data and prevent unauthorized access. Privacy concerns are addressed through transparent data handling practices and compliance with relevant regulations.

BASIC ARDUINO CODE FOR SMART WATER FOUNTAIN



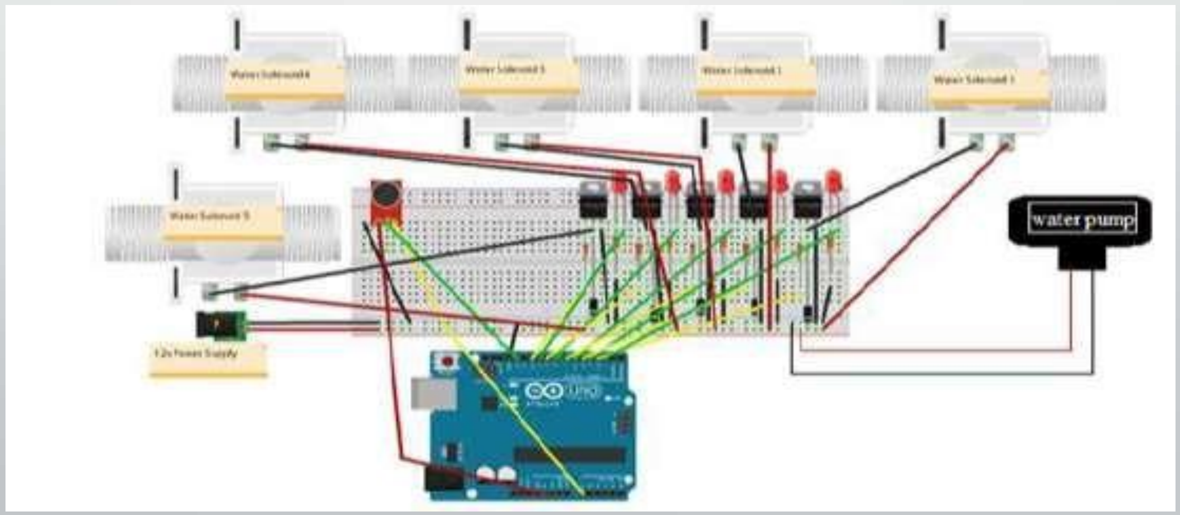
```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <LiquidCrystal_I2C.h>

#define DHTPIN 2 // Pin for DHT sensor
#define DHTTYPE DHT11 // Type of DHT sensor
#define PUMP_PIN 7 // Pin for water pump
#define MOISTURE_PIN A0 // Pin for soil moisture sensor
#define LCD_ADDRESS 0x27 // I2C address for LCD

DHT dht(DHTPIN, DHTTYPE);
LiquidCrystal_I2C lcd(LCD_ADDRESS, 16, 2);

void setup() {
  pinMode(PUMP_PIN, OUTPUT);
  pinMode(MOISTURE_PIN, INPUT);
  dht.begin();
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Smart Fountain");
}
```



```
void loop() {
  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature();

  lcd.setCursor(0, 1);
  lcd.print("Temp: ");
  lcd.print(temperature);
  lcd.print("C   ");
  lcd.print("Humidity: ");
  lcd.print(humidity);
  lcd.print("%");

  int moistureValue = analogRead(MOISTURE_PIN);

  if (moistureValue < 500) {
    digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
    delay(5000); // Run the pump for 5 seconds
    digitalWrite(PUMP_PIN, LOW); // Turn off the pump
  }

  delay(5000); // Delay for 5 seconds before the next reading
}
```

BASIC CIRCUIT

This code assumes you have a DHT temperature and humidity sensor, a soil moisture sensor, and an LCD screen connected to your Arduino. It measures temperature, humidity, and soil moisture. If the soil moisture falls below a certain threshold (500 in this example), it activates the water pump for 5 seconds to water the plants.Please note that this is a basic example, and you can enhance it with additional features like Wi-Fi connectivity for remote monitoring and control, scheduling, and more, depending on your project requirements. Also, ensure you have the necessary libraries installed in your Arduino IDE for the sensors and LCD display.



Conclusion:

The Smart Water Fountains Using IoT project represents a promising solution for improving water fountain efficiency and sustainability in public spaces. By integrating sensor technology, data analytics, user interaction, and water conservation mechanisms, this system offers numerous benefits, including reduced water waste, enhanced user experience, and simplified maintenance. With a focus on security and privacy, it aligns with the growing demand for responsible and data-driven water management in urban environments.

IOT-enabled Environmental Monitoring in Parks system

PHASE-3

INTRODUCTION:-

- ✓Water fountains have been a major tourist attraction these days which freeze the attention of tourists with their variety of lights, designs, and heights. And as we all know music holds a major part in our day-to-day lives.
- ✓And hence our idea is to combine the beautiful water fountain with music which makes an extraordinary tourist attraction when constructively set with a range of frequencies that enables us to operate through various electronic devices.

- ✓The project aims to enhance public water fountains by implementing IoT sensors to control water flow and detect malfunctions.

THINGS USED IN THE PROJECT:-

COMPONENTS USED IN THE PROJECT:-

1. Microcontroller:

ESP8266 or ESP32: These are low-cost Wi-Fi modules with integrated microcontrollers. They are commonly used for IoT projects due to their Wi-Fi capabilities.

2. Sensors:

Water Level Sensor: To measure the water level in the fountain.

Motion Sensor (PIR Sensor): To detect the presence of people or animals near the fountain.

Temperature and Humidity Sensor: To monitor the environment around the fountain.

3. Actuators:

Water Pump: To control the flow of water in the fountain.

LEDs: For decorative lighting or indicating the fountain's status.

4. Communication:

Wi-Fi Module: Allows the fountain to connect to the internet.

MQTT Protocol: A lightweight messaging protocol for small sensors and mobile devices optimized for high-latency or unreliable networks.

5. Power Supply:

Power Source: Depending on the location of the fountain, you might use batteries, solar power, or a stable electrical source.

6. IoT Platform:

Cloud Service (e.g., AWS IoT, Google Cloud IoT, Azure IoT): A cloud platform to store data from the fountain and manage device communication.

IoT Development Board: Some development boards come with built-in support for IoT platforms, making it easier to connect your devices to the cloud.

7. User Interface:

Mobile App/Web App: Allows users to remotely control and monitor the fountain.

Push Notifications: Sends alerts or notifications to users based on fountain events (e.g., low water level).

8. Security:

Encryption and Authentication: Ensures secure communication between the fountain and the IoT platform.

9. Data Storage and Analysis:

Database: For storing historical data from the fountain (e.g., water usage patterns, user interactions).

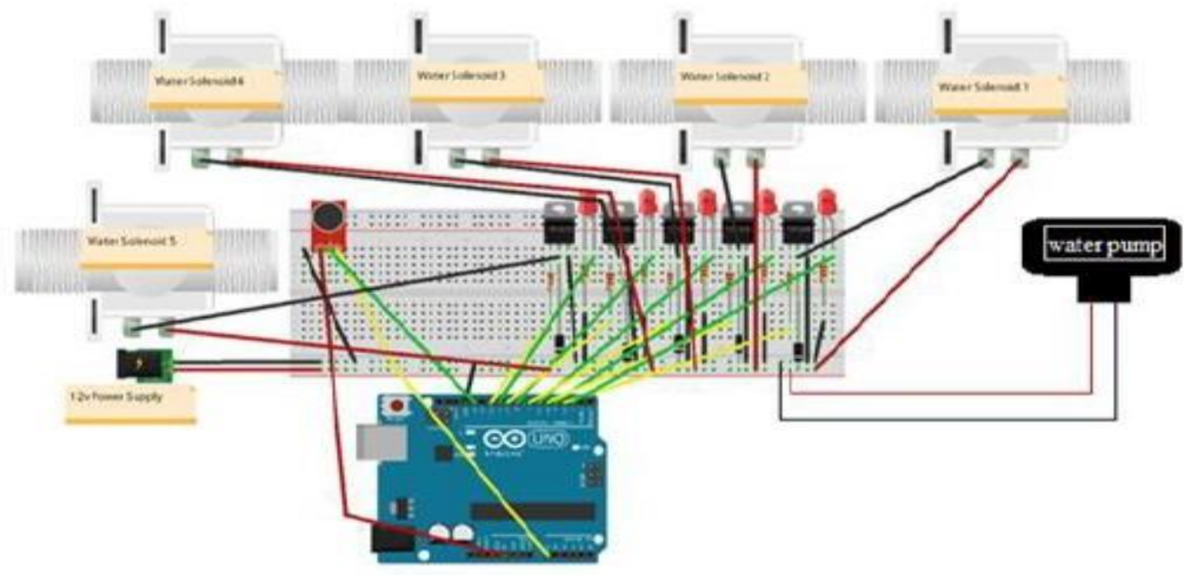
Analytics Tools: To gain insights from the collected data.

10. Additional Components:

Real-Time Clock (RTC) Module: Maintains accurate time for scheduling events even when the microcontroller is powered off.

LCD Display: Provides real-time feedback or information about the fountain's status.

SCHEMATIC DIAGRAM:-

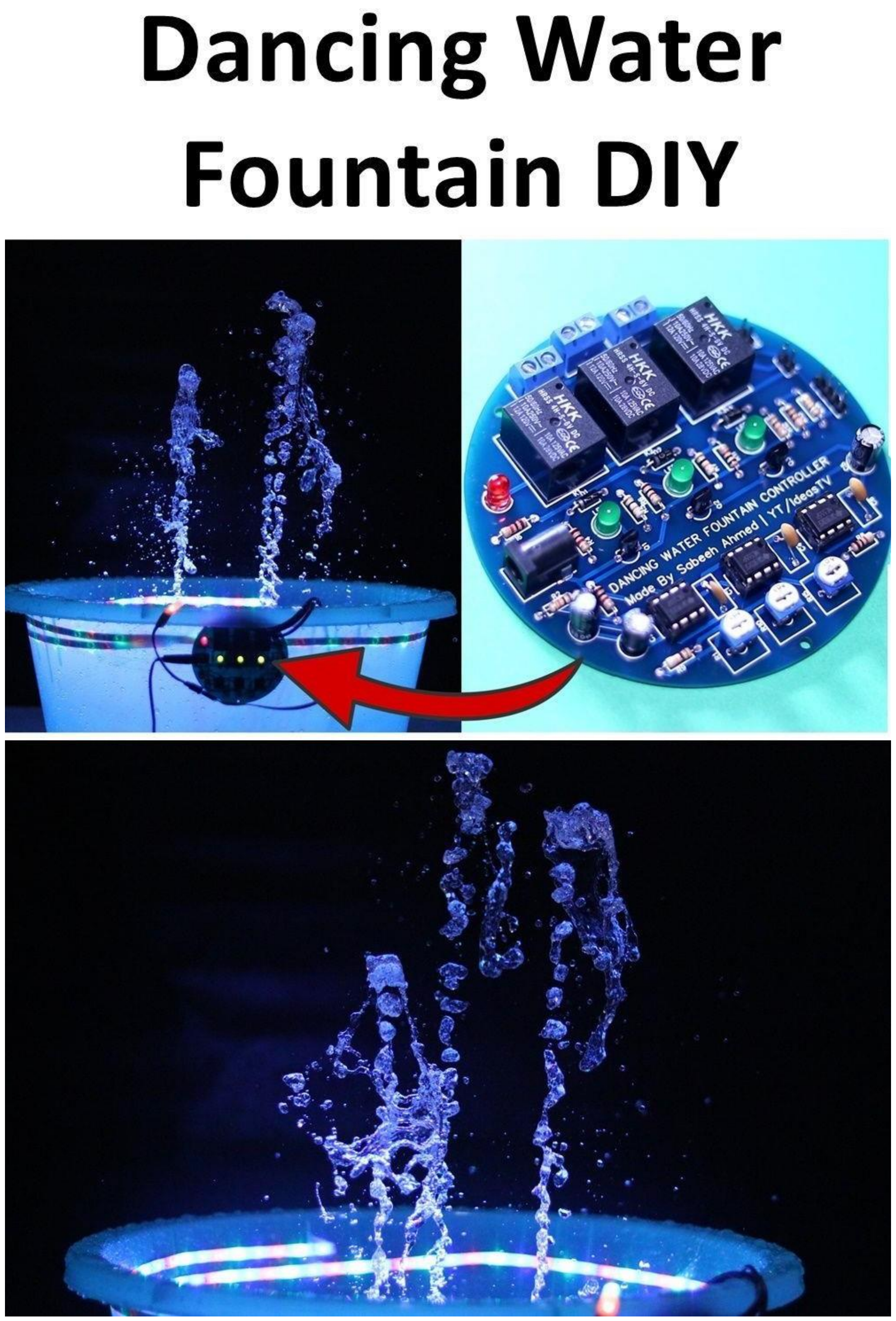


PYTHON CODE:-

```
import RPi.GPIO as GPIO
import time

# GPIO pins for water level sensor and
pump
WATER_LEVEL_PIN = 17
PUMP_PIN = 18

# Setup GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(WATER_LEVEL_PIN,
GPIO.IN)
GPIO.setup(PUMP_PIN, GPIO.OUT)

def is_water_available():
    """Check if water is available based
```

```
on the water level sensor."""
    return
GPIO.input(WATER_LEVEL_PIN) ==
GPIO.LOW

def turn_on_pump():
    """Turn on the water pump."""
    GPIO.output(PUMP_PIN,
GPIO.HIGH)
    print("Pump turned on")

def turn_off_pump():
    """Turn off the water pump."""
    GPIO.output(PUMP_PIN,
GPIO.LOW)
    print("Pump turned off")

try:
    while True:
        if is_water_available():
            turn_on_pump()
            time.sleep(5)  # Run the pump
for 5 seconds (adjust as needed)
            turn_off_pump()
        else:
            print("No water available.
Waiting for refill...")
            time.sleep(2)  # Wait for 2
seconds before checking again

except KeyboardInterrupt:
    # Cleanup GPIO on keyboard
interrupt
    GPIO.cleanup()
```

OUTPUT:-


Dancing Water Fountain DIY
DANCING WATER FOUNTAIN CONTROLLER
Made By Sabeeh Ahmed | YT/IdeasTV
HKK

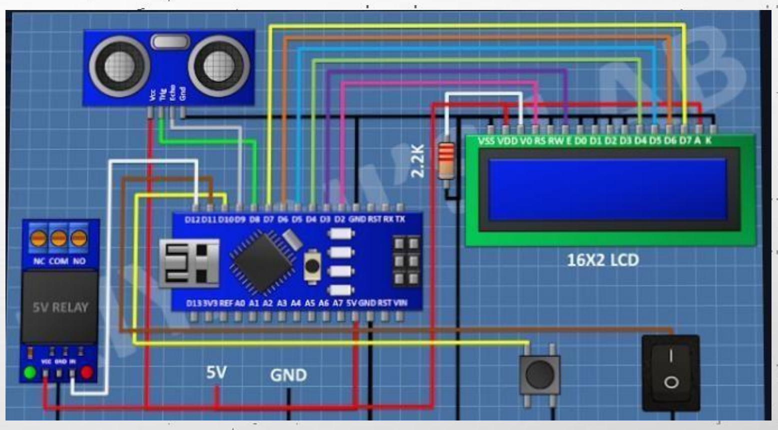
SMART WATER FOUNTAIN

OBJECTIVE :

- IT'S GREAT FOR BEGINNER PROJECTS AND SUPER EASY TO USE! THE PUMP IS BASICALLY A DC MOTOR THAT IS POWERED WITH 3V AND DRAWS 100MA. WHEN POWERED, THE PUMP SUCKS WATER IN FROM THE SIDE OF THE PLASTIC CASING AND PUSHES IT OUT THE TUBING PORT. THE PUMP MUST BE PRIMED BY KEEPING IT INSIDE WATER AT ALL TIMES.

COMPONENTS:

* ARDUINO UNO OR ARDUINO NAN
* 1K 0.25WATT RESISTORS - 8 NO (R1 - R8)
* 10K 0.25WATT RESISTORS - 4 NO (R9 - R12)
* BC547 NPN TRANSISTOR (Q1)
* LED 5MM - 7NO
* 2-PIN TERMINAL CONNECTORS (5 NO)



DESIGN FEATURES :

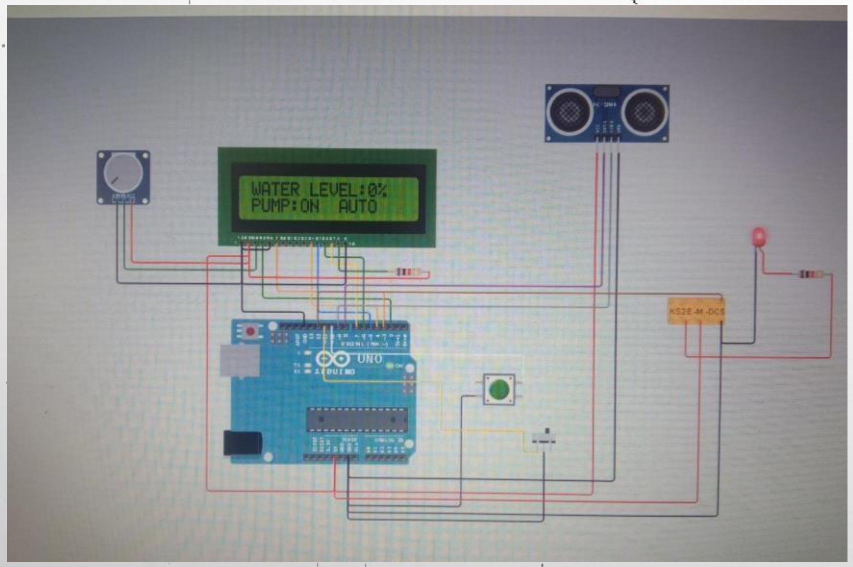
1.FOUNTAINS WIRELESSLY COMMUNICATE WITH BASE STATIONS

2. BASE STATIONS COLLECT AND TRANSMIT USAGE, FILTER, AND SYSTEM HEALTH INFORMATION TO THE CLOUD VIA ETHERNET

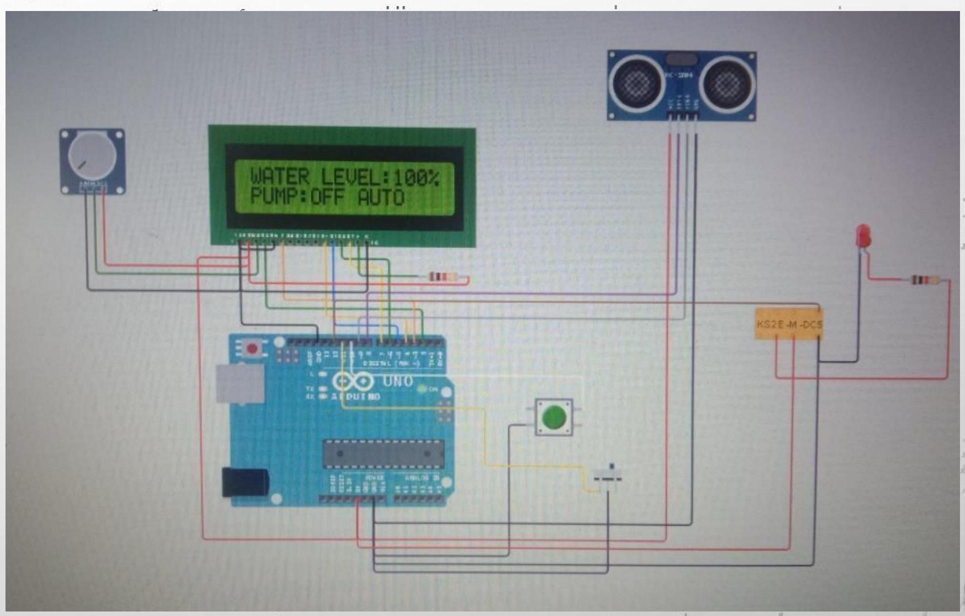
3.WIRELESS COMMUNICATIONS USE A LOW-POWER UNLICENSED BAND FOR IMPROVED SECURITY AND POWER SAVINGS

CIRCUIT :

- **AUTO:**

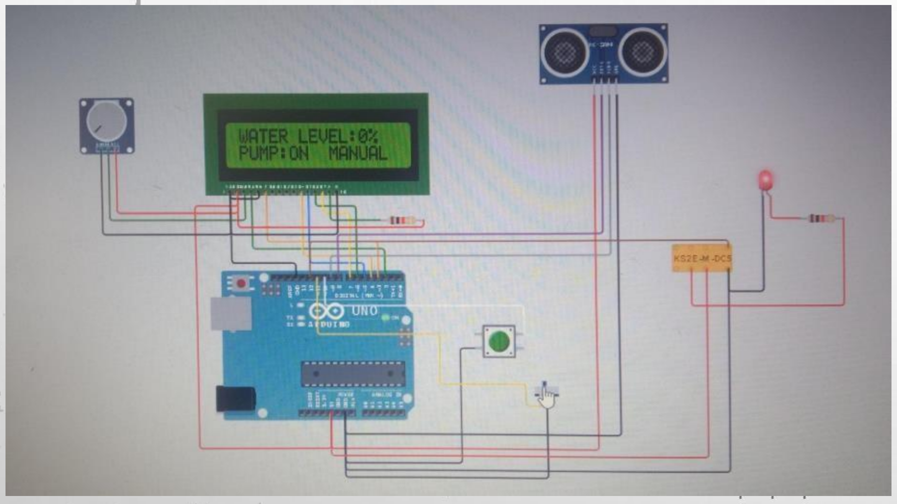


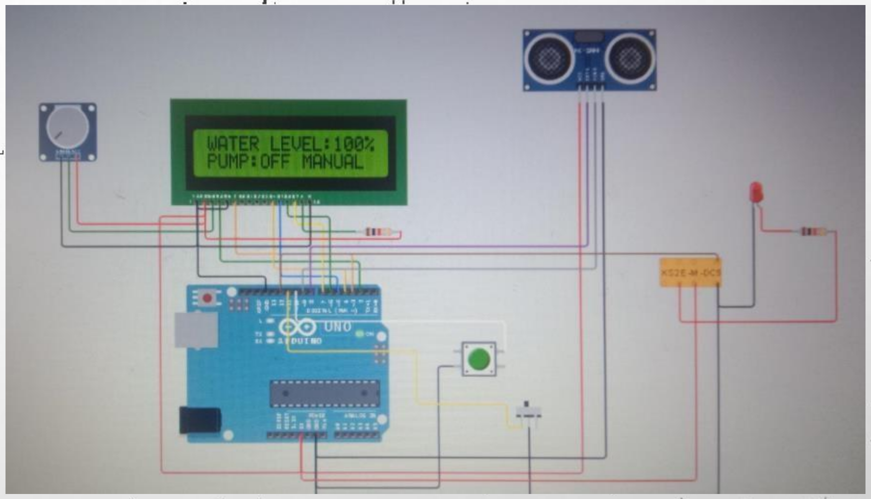
ON



OFF

MANUAL:





ON

OFF

WORKING:

- IN MANUAL;

  IF TANK IS EMPTY, WE NEED TO TURN ON MANUALLY

. IN AUTO;

  IF TANK IS EMPTY, MOTOR TURN ON AUTOMATICALLY

```
#include <EEPROM.h>
#include <LiquidCrystal.h>
LiquidCrystal lcd(2,3,4,5,6,7);
long duration, inches;
int set_val,percentage;
bool state,pump;
void setup(){
lcd.begin(16, 2);
 lcd.print("WATERLEVEL");
 lcd.setCursor(0,1);
 lcd.print("PUMP:OFFMANUAL");
 pinMode(8,OUTPUT);
pinMode(9,INPUT);
pinMode(10,INPUT_PULLUP);
pinMode(11,INPUT_PULLUP);
pinMode(12,OUTPUT);
```

```
set_val=EEPROM.read(0);
if(set_val>150)set_val=150;}
void loop() {
digitalWrite(3, LOW);
 delayMicroseconds(2);
digitalWrite(8, HIGH);
delayMicroseconds(10);
digitalWrite(8, LOW);
duration = pulseIn(9, HIGH);
 inches = microsecondsToInches(duration);
percentage=(set_val-inches)*100/set_val;
lcd.setCursor(12, 0);
if(percentage<0)percentage=0;
lcd.print(percentage);
 lcd.print("%    ");
if(percentage<30&digitalRead(11))pump=1;
if(percentage>99)pump=0;
digitalWrite(12,!pump);
```

```
lcd.setCursor(5,1);
if(pump==1)lcd.print("ON ");
else if(pump==0)  lcd.print("OFF");
 lcd.setCursor(9,1);
if(!digitalRead(11))lcd.print("MANUAL");
else lcd.print("AUTO  ");
if(!digitalRead(10)&!state&digitalRead(11)){
state=1;
  set_val=inches;
 EEPROM.write(0, set_val);
     }
 if(!digitalRead(10)&!state&!digitalRead(11)){
   state=1;
  pump=!pump;
}
 if(digitalRead(10))state=0;
  delay(500);
}
long microsecondsToInches(long microseconds){
return microseconds/  74 /  2;
}
```

OUTPUT LINK:

- HTTPS://WOKWI.COM/PROJECTS/379631344770984961

THANK YOU