## SMART WATER FOUNTAIN

### PHASE 3: DEVELOPMENT PART 1

Utilizing the Wokwi simulator, the first step in creating a smart water fountain is to mimic it. Below is a list of the components used for the simulation of a smart water fountain:

1. NODE MCU ESP32
2. water pump.
3. Relay module
4. An ultrasonic sensor (HC SR04)
5. Temperature sensor(DHT22)
6. Wokwi virtual components

The circuit is then constructed in Wokwi's circuit editor by adding parts such the NODE MCU ESP32, the water pump, the relay module, temperature sensor and the ultrasonic sensor. Relay module and ultrasonic sensors are linked to the NODE MCU ESP32 in the circuit.virtual components that offer web interface control for the smart water fountain, a button and range element is added.

NODE MCU ESP32 - this device functions as a microcontroller. Water is pumped from the tank to the fountain using a water pump. Relay module is used to manage the water pump.

An ultrasonic sensor (HC SR04) is used to gauge the fountain's water level. Wokwi virtual components for simulation and web interface. Temperature sensors are used to measure the temperature and humidity of the water.Then the python code was written for all the components.

.

**Coding:**

```
include <ESP32WiFi.h>

 #include <WiFiClient.h>

 #include < Ultrasonic.h>

#include<temperature.h>


 const char* ssid = "YourWiFiSSID";

 const char* password = "YourWiFiPassword";
```

```
const int trigPin = D2;
 const int echoPin = D3;
 const int relayPin = D1;

Ultrasonic ultrasonic(trigPin, echoPin);
WiFiServer server(80);

void setup() {
pinMode(relayPin, OUTPUT);
 digitalWrite(relayPin, LOW);
Serial.begin(115200);
WiFi.begin(ssid, password);
 while (WiFi.status() != WL_CONNECTED) {
 delay(1000);
 Serial.println("Connecting to WiFi...");
}
 server.begin();
}
 void loop()  {
WiFiClient client = server.available();
if (client) {
String request = client.readStringUntil('\r');
 if (request.indexOf("/on") != -1) {
 digitalWrite(relayPin, HIGH); // Turn the pump on
delay(2000); // Run the pump for 2 seconds
 digitalWrite(relayPin, LOW); // Turn the pump off
}
client.flush();
```

```
}
 float distance = ultrasonic.read();
 if (distance < 10) {
// Water is low, update the web interface
 // You can send an HTML response to the client here
}
}
```

Then the circuit was saved and simulated. With this setup, we can simulate a smart water fountain that can be remotely controlled, and water level is monitored.