Project Title :

IOT SMART WATER FOUNTAINS

Overview:

Create a smart water fountain system that can be controlled and monitored remotely through a mobile app or web interface. This project will ensure the efficient use of water, provide real-time data, and offer user-friendly control options.

Components :

Water Fountain: Choose or design a water fountain suitable for this project.

Water Flow Sensor:

Install a flow sensor to measure water consumption and detect leaks.

Water Level Sensor:

Implement a water level sensor to monitor the fountain’s water level.

Pump Control:

Connect a pump to the microcontroller for water circulation.

Wi-Fi Module:

Add a Wi-Fi module (e.g., ESP8266) for internet connectivity.

Mobile App/Web Interface:

Develop a user-friendly app or web interface for remote control and monitoring.

Features:

Remote Control: Users can turn the fountain on/off and adjust its settings remotely via the app or web interface.

Water Conservation:

Implement features to automate the fountain based on water level and flow, conserving water resources.

Real-time Monitoring:

Display real-time data such as water level, flow rate, and pump status on the app/web interface.

Alerts:

Send notifications/alerts to users in case of leaks or low water levels.

Customization:

Allow users to choose fountain patterns, timings, and water flow rates.

Energy Efficiency:

Incorporate energy-saving features like scheduling and power management.

How it Works :

The microcontroller communicates with the sensors to monitor water levels and flow rates.

User inputs through the app or web interface are sent to the microcontroller via Wi-Fi.

The microcontroller processes these inputs to control the pump and fountain operation accordingly.

Real-time data is sent to the app/web interface for monitoring.

Benefits :

\*Efficient water usage and conservation

\*Convenience and control for users.

\*Early detection of issues, reducing maintenance costs.

\*Fun and aesthetic appeal with customizable fountain patterns.

\*Potential for integration with smart home systems.

Challenges:

\*Ensuring a secure and reliable internet connection.

\*Designing a user-friendly interface.

\*Precision in water level and flow monitoring.

\*Power management for long-term use.

Remember to plan the project thoroughly, consider safety measures, and continuously test and refine your system. Good luck with your IoT smart water fountain project!

Code :

#include <Arduino.h>

#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>

// Wi-Fi settings

Const char\* ssid = “YourWiFiSSID”;

Const char\* password = “YourWiFiPassword”;

// Create an instance of the web server

ESP8266WebServer server(80);

// Pin for controlling the water pump

Const int pumpPin = D1; // Replace with your actual pin number

// Variables for water flow and level (replace with actual sensors)

Float flowRate = 0.0;

Float waterLevel = 0.0;

Void setup() {

pinMode(pumpPin, OUTPUT);

// Connect to Wi-Fi

WiFi.begin(ssid, password);

While (WiFi.status() != WL\_CONNECTED) {

Delay(1000);

Serial.println(“Connecting to WiFi…”);

}

Serial.println(“Connected to WiFi”);

// Define web server routes

Server.on(“/”, HTTP\_GET, handleRoot);

Server.on(“/on”, HTTP\_GET, handleOn);

Server.on(“/off”, HTTP\_GET, handleOff);

// Start web server

Server.begin();

}

Void loop() {

Server.handleClient();

// Read water flow and level from sensors (replace with actual code)

// flowRate = readFlowSensor();

// waterLevel = readLevelSensor();

// Check water level and control the pump (replace with your logic)

If (waterLevel < thresholdLevel) {

digitalWrite(pumpPin, HIGH);

} else {

digitalWrite(pumpPin, LOW);

}

}

Void handleRoot() {

String html = “<html><body>”;

Html += “<h1>Smart Water Fountain</h1>”;

Html += “<p>Flow Rate: “ + String(flowRate) + “ L/min</p>”;

Html += “<p>Water Level: “ + String(waterLevel) + “ cm</p>”;

Html += “<p><a href=’/on’>Turn On</a></p>”;

Html += “<p><a href=’/off’>Turn Off</a></p>”;

Html += “</body></html>”;

Server.send(200, “text/html”, html);

}

Void handleOn() {

digitalWrite(pumpPin, HIGH);

server.send(200, “text/plain”, “Fountain turned on”);

}

Void handleOff() {

digitalWrite(pumpPin, LOW);

server.send(200, “text/plain”, “Fountain turned off”);

}