IOT HACKTHIN

SMART IRRIGATION SYSTEM

USING ARDUINO AND ESP8266 WI-FI MODULE

TEAM :

PES2UG22CS419 : PUNEETH BU

PES2UG22CS431 : RAKESH C

Title

Smart IoT-Based System Using Arduino and ESP8266 WiFi Module

---

1. Hardware Components Used

1. **Arduino Uno**
    - Microcontroller board based on the ATmega328P.
2. **ESP8266 WiFi Module**
    - A low-cost WiFi chip with a built-in TCP/IP stack.
3. **Power Supply**
    - 5V DC for Arduino, 3.3V for ESP8266.
4. **Sensors**
    - Example: DHT11 (temperature and humidity sensor).
5. **Actuators** (e.g., relays, motors)

- Example: Relay module for switching devices.

6. **Jumper Wires**
    - For making connections between components.
7. **Resistors and Capacitors**
    - For signal conditioning and stability (e.g., pull-up resistors for I2C).
8. **Breadboard or PCB**
    - For prototyping and connections.

2. Development Boards Specifications

Arduino Uno

- **Microcontroller**: ATmega328P.
- **Operating Voltage**: 5V.
- **Digital I/O Pins**: 14 (6 PWM).
- **Analog Input Pins**: 6.
- **Clock Speed**: 16 MHz.
- **Communication Interfaces**: UART, I2C, SPI.

ESP8266 WiFi Module

- **Chip**: ESP8266EX.
- **Operating Voltage**: 3.3V.
- **WiFi Standards**: IEEE 802.11 b/g/n.
- **Communication Interfaces**: UART, SPI, I2C.
- **GPIO Pins**: 16.
- **Clock Speed**: 80 MHz (default), up to 160 MHz.

4. Predictive Analysis

Predictive analysis can be performed using the collected sensor data to forecast trends or events. Examples:

1. **Environmental Monitoring**:
    - Use temperature and humidity data to predict weather patterns.
2. **Smart Home Automation**:
    - Predict energy consumption based on historical appliance usage.
3. **IoT Data Analytics**:
    - Analyze data trends using cloud platforms (e.g., ThingSpeak or AWS IoT).

5. Cloud Platform Used

- **Platform**: Firebase
- **Purpose**:
    - Store sensor data in the cloud.
    - Visualize real-time data using graphs.
    - Analyze historical data for predictive analysis.

---

6. Security Measures Taken Care

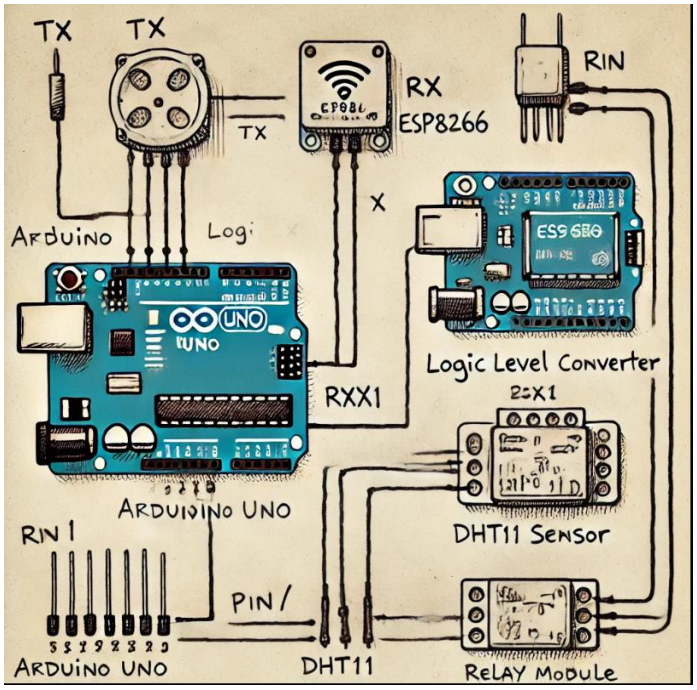
To ensure a secure system:

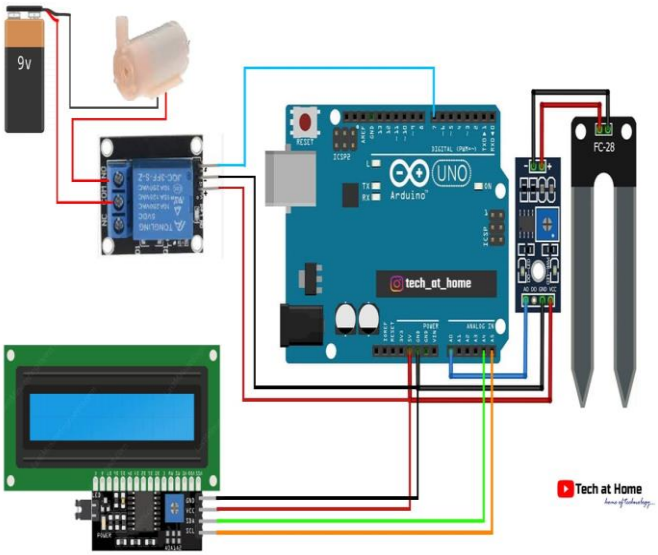
1. **Data Encryption**:
    - Use HTTPS for secure communication between the ESP8266 and the cloud.
2. **Authentication**:
    - Implement API keys for accessing the cloud platform.
3. **Network Security**:
    - Use WPA2 encryption for the WiFi network.
4. **Input Validation**:

- Validate data received from sensors to avoid incorrect or malicious inputs.

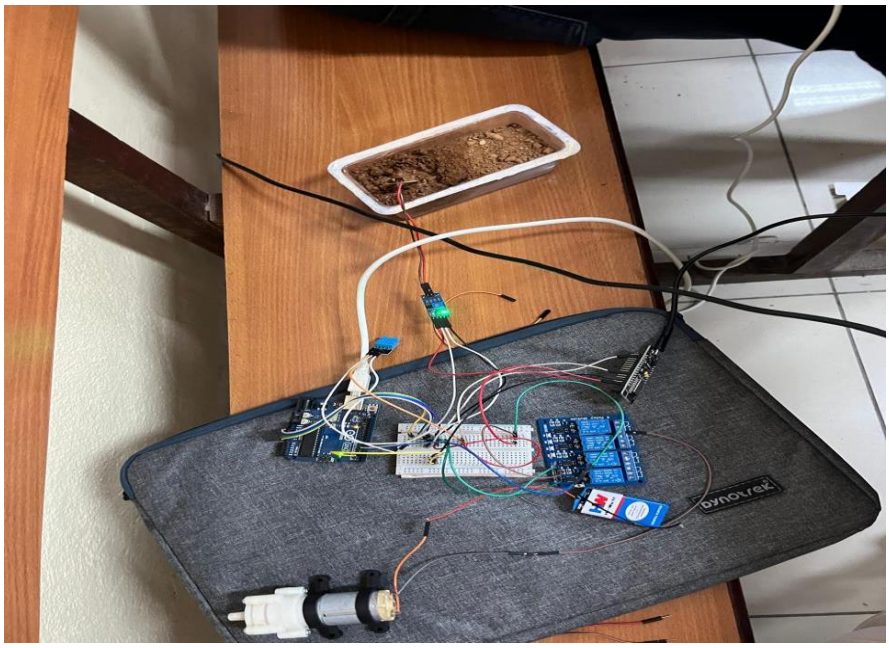
5. **Firmware Updates**:
    - Ensure the ESP8266 and Arduino have the latest firmware for improved security.

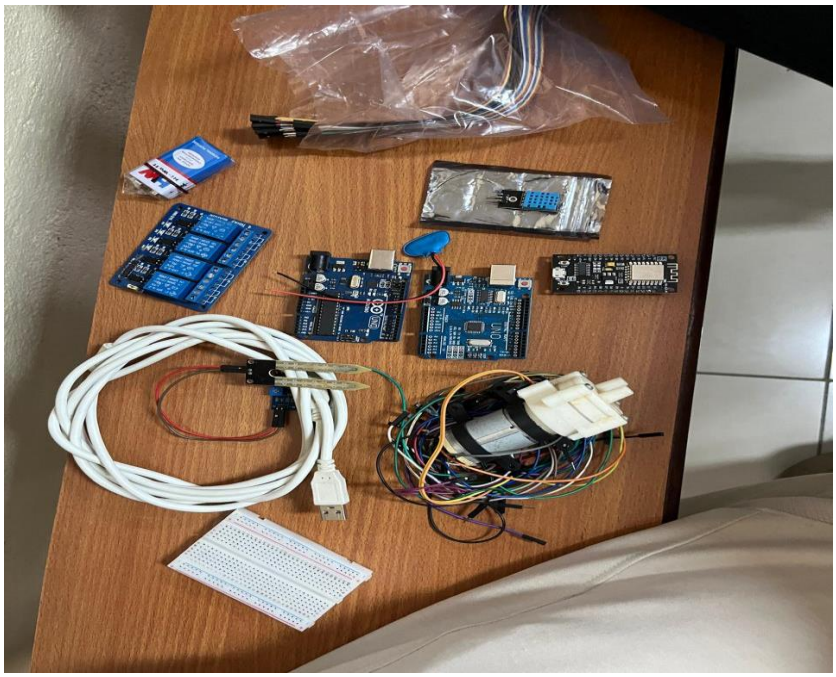
Circuit Diagram



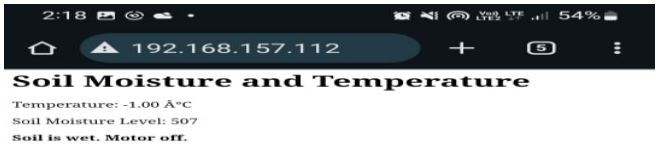


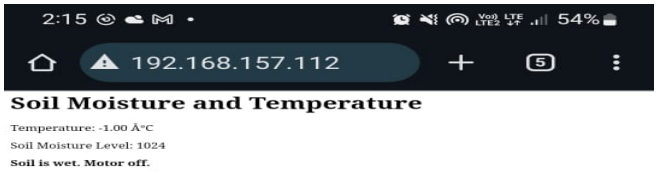
SCREENSHOT OF PROJECT





SCREEN SHOT IN PHONE





1. Purpose of Predictive Analysis

- To leverage sensor data (e.g., temperature, humidity, or other environmental parameters) collected by the Arduino-ESP8266 system.

- Predict outcomes or behaviors, such as weather conditions, system failures, or resource consumption, to improve decision-making and efficiency.

Predictive Analysis for Your Arduino and ESP8266 Project

Predictive analysis involves using data collected by sensors and devices to forecast future trends, behaviors, or events. Here's how it applies to your Arduino and ESP8266-based project:

---

1. Purpose of Predictive Analysis

- To leverage sensor data (e.g., temperature, humidity, or other environmental parameters) collected by the Arduino-ESP8266 system.
- Predict outcomes or behaviors, such as weather conditions, system failures, or resource consumption, to improve decision-making and efficiency.

---

2. Data Sources

1. **DHT11 Sensor (Temperature and Humidity)**:
    - Collect temperature and humidity readings over time.
    - Analyze patterns like rising temperatures or humidity spikes.
2. **Relay Module (Actuator Control)**:
    - Track how often the relay is activated (e.g., motor usage).
    - Correlate usage patterns with external conditions (e.g., temperature or soil moisture).
3. **ESP8266 Data Logs**:
    - Store data in a cloud platform for historical analysis.
    - Use timestamps to identify trends or anomalies.

Predictive Analysis for Your Arduino and ESP8266 Project

Predictive analysis involves using data collected by sensors and devices to forecast future trends, behaviors, or events. Here's how it applies to your Arduino and ESP8266-based project:

---

1. Purpose of Predictive Analysis

- To leverage sensor data (e.g., temperature, humidity, or other environmental parameters) collected by the Arduino-ESP8266 system.
- Predict outcomes or behaviors, such as weather conditions, system failures, or resource consumption, to improve decision-making and efficiency.

---

2. Data Sources

1. **DHT11 Sensor (Temperature and Humidity)**:
    - Collect temperature and humidity readings over time.
    - Analyze patterns like rising temperatures or humidity spikes.
2. **Relay Module (Actuator Control)**:
    - Track how often the relay is activated (e.g., motor usage).
    - Correlate usage patterns with external conditions (e.g., temperature or soil moisture).
3. **ESP8266 Data Logs**:
    - Store data in a cloud platform for historical analysis.
    - Use timestamps to identify trends or anomalies.

---

3. Example Use Cases for Predictive Analysis

a. Environmental Monitoring

- Use historical temperature and humidity data to forecast weather patterns.
- Predict high-humidity periods to activate a dehumidifier.

b. Smart Irrigation System

- Predict when soil moisture will drop below a critical level based on past trends.
- Automate irrigation before plants are water-stressed.

c. Energy Optimization

- Monitor relay usage and predict high-demand periods.
- Optimize power supply to reduce energy costs or prevent overloads.

d. Fault Prediction

- Detect abnormal behavior in sensor readings (e.g., constant temperature).
- Alert users to potential sensor or actuator malfunctions.

4. Methods for Predictive Analysis

a. Linear Regression

- Use to predict future temperature or humidity values based on historical data trends.

b. Time Series Analysis

- Analyze patterns in sensor data over time (e.g., seasonal changes).

c. Machine Learning

- Train models using cloud platforms (e.g., AWS, Azure) to make more complex predictions.
- Use historical data to train models for anomaly detection and prediction.

d. Threshold-Based Alerts

- Set predefined thresholds for parameters (e.g., temperature > 35°C) to trigger early warnings.

5. Workflow

1. **Data Collection**:
    - Sensor readings are sent from Arduino to ESP8266.
    - ESP8266 uploads the data to a cloud platform like ThingSpeak or Firebase.
2. **Data Storage and Visualization**:
    - Store data in the cloud.
    - Use real-time graphs for visualization.
3. **Analysis**:
    - Use cloud-based tools or local systems (e.g., Python with libraries like Pandas and Matplotlib) to analyze trends.
4. **Predictions**:
    - Apply models to predict future values or trends.
5. **Actionable Insights**:
    - Use predictions to trigger actions (e.g., turning on/off relays).

6. Tools and Platforms

- **Cloud Platforms**:
    - ThingSpeak: For real-time data visualization and analysis.
    - AWS IoT Core: For advanced analytics and machine learning integration.
- **Programming Tools**:
    - Python (with libraries like NumPy, Pandas, SciKit-learn for analysis).
- **IoT Integration**:

- Use MQTT or HTTP protocols to enable seamless data transfer between ESP8266 and the cloud.