# 1. Project Overview

## Short title: GreenSense ESP32 IoT System

## Objective

* Monitor environmental parameters (temperature, humidity, soil moisture).
* Control a water pump based on soil moisture levels.
* Transmit sensor data to a remote Raspberry Pi server.
* Enable Over-The-Air (OTA) firmware updates for easy maintenance.

# Hardware

* ESP32 Development Board.
* DHT11 Sensor (Temperature and Humidity).
* Capacitive Soil Moisture Sensor.
* Water pump module.

# Software

* PlatformIO with Arduino Framework.
* Wi-Fi for network communication.
* UDP protocol for data transmission.
* Rpi for central server logging.

# 2. Software Modules

## 2.1. Main Program (`main.cpp`)

**Responsibilities:**  
- Initialize Wi-Fi, OTA, data collection, and transmission modules.  
- Manage the main loop to handle tasks like pump control, data collection, and transmission.  
**Key Functions:**  
- `setup()`: Initializes all components.  
- `loop()`: Executes tasks at specific intervals using `millis()`.

## 2.2. Wi-Fi Connection (`wifi\_connect.h` and `wifi\_connect.cpp`)

**Responsibilities:**  
- Connect the ESP32 to a Wi-Fi network.  
**Key Function:**  
- `connectToWiFi(const char\* ssid, const char\* password)`.

## 2.3. OTA Updates (`ota\_handler.h` and ota\_handler.cpp`)

**Responsibilities:**  
- Enable OTA firmware updates.  
**Key Function:**  
- `setupOTA(const char\* hostname, const char\* password)`.

## 2.4. Sensor Data Collection (`data\_collector.h` and `data\_collector.cpp`)

**Responsibilities:**  
- Collect and store sensor data in a circular buffer.  
**Key Functions:**  
- `initDataCollector()`.  
- `collectSensorData()`.  
- `getSensorBuffer()`.

## 2.5. Data Transmission (`data\_transmitter.h` and data\_transmitter.cpp`)

**Responsibilities:**  
- Transmit sensor data to a remote server via UDP.  
**Key Functions:**  
- `initDataTransmitter(const char\* ip, int port)`.  
- `transmitData()`.

## 2.6. Pump Control (`pump\_actuator.h` and pump\_actuator.cpp`)

**Responsibilities:**  
- Actuate the pump based on soil moisture levels.  
**Key Functions:**  
- `setupPump()`.  
- `controlPump()`.

## 2.7. Sensor Drivers (`sensors.h` and `sensors.cpp`)

**Responsibilities:**  
- Provide functions to read sensor data.  
**Key Functions:**  
- `getTemperature()`.  
- `getHumidity()`.  
- `getSoilMoisture()`.

# 3. Workflow

# 3.1. Initialization (`setup()`)

* Connect to Wi-Fi.
* Initialize OTA updates.
* Initialize data collection and transmission modules.
* Initialize the pump actuator.

# 3.2. Main Loop (`loop()`)

* Handle OTA updates continuously.
* Control the pump every 1 second.
* Collect sensor data every 5 seconds.
* Transmit data to the server every 30 seconds.

# 4. Data Flow

* Sensor Data Collection: Temperature, humidity, and soil moisture are collected and stored in a circular buffer.
* Data Transmission: The buffer is transmitted to the server via UDP every 30 seconds.
* Pump Control:  
  - The pump is turned on if soil moisture drops below 30%.  
  - The pump is turned off if soil moisture exceeds 60%.

# 5. PlatformIO Configuration (`platformio.ini`)

[env:esp32dev]  
platform = espressif32  
board = esp32doit-devkit-v1  
framework = arduino  
lib\_deps = adafruit/DHT sensor library  
upload\_port = 192.168.29.86  
upload\_protocol = espota  
upload\_flags =  
 --port=3232  
 --auth=admin  
 --timeout=30

# 6. Troubleshooting

# Common Issues

* Wi-Fi Connection:  
  - Ensure the correct SSID and password are provided.  
  - Check the serial monitor for connection status.
* OTA Updates:  
  - Ensure the ESP32 and the computer are on the same network.

- Ensure correct IP address in platform.ini.  
- Verify the OTA port and password in platformio.ini.

* Data Transmission:  
  - Ensure the server IP and port are correct.  
  - Check for network connectivity issues.

# 7. Future Enhancements

* Add HTTPS for secure data transmission.
* Integrate the power system (Li-ion cell and solar panel) voltage monitoring.
* Enable additional sensors (e.g., pH sensor, npk, sunlight).