IoT-Based Voice Controlled Home Automation

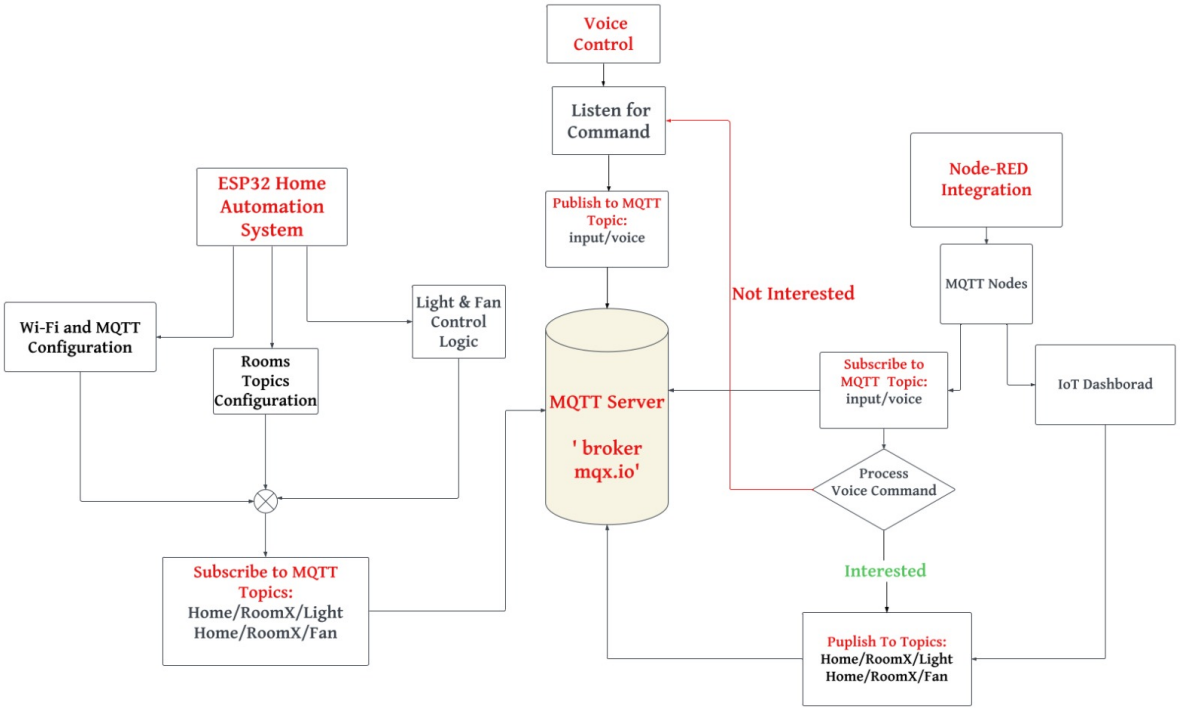
EasyLife Team

Members:
Abdullah Al-Shateri (L)
Mazen Mohammed
Mohanad Ashraf

Documentation prepared by: Abdullah Al-Shateri

Introduction:

The project combines an ESP32-based home automation system, a voice control module using Python, and integrates with Node-RED for additional processing and UI capabilities. The system also includes DHT22 sensor monitoring through a Raspberry Pi 4 since pi act as Node-RED server. The goal is to provide a flexible and user-friendly solution for home automation.



System Architecture Flowchart

Requirements and Steps to Recreate:

1. ESP32 Home Automation System:
    a. Requirements:
        i. ESP32
        ii. Arduino IDE

    b. Libraries
    Include essential libraries for Wi-Fi connectivity (WiFi.h), MQTT communication (PubSubClient.h), and motor control (L298N.h).

    c. Configure Wi-Fi and MQTT:
    Set up Wi-Fi credentials, MQTT broker information, and a unique client ID for the ESP32.

    d. Define Room Configuration:
    Define MQTT topics and pin configurations for both rooms, specifying light and fan control topics and corresponding pin assignments for the L298N motor driver.

    e. Initialize Objects:
    Create instances of the WiFiClient and PubSubClient classes to handle Wi-Fi and MQTT communication.

    f. Implement Callback Function:
    Implement a function invoked when an MQTT message is received. Parse the topic and payload, calling the appropriate functions for light or fan control.

    g. Light and Fan Control Functions:
    Develop functions that handle the logic for controlling lights and fan speed based on the received MQTT payload.

    h. Setup Function:
    Establish serial communication for debugging, connect to the Wi-Fi network, configure the MQTT client with the broker's details, and subscribe to relevant topics.

    i. Loop Function:
    Continuously check the MQTT client's connection status, attempt reconnection on disconnection, and maintain the MQTT connection while handling incoming messages.

2. Voice Control Module:
    a. Requirements
        i. Python
        ii. SpeechRecognition
        iii. Paho-mqtt

    b. Install Required Python Packages:
    Run the command: ‘ pip install SpeechRecognition paho-mqtt ‘ on CMD

    c. Configure MQTT:
    Set up the configuration for the MQTT broker, including the server address, port, and the topic to which voice commands will be published.

    d. Initialize Speech Recognition Module:
    Create an instance of the recognizer class from the speech_recognition library to manage speech recognition.

    e. Listen For Command Function:
    Capture audio from the microphone, adjust for ambient noise, and use Google Speech Recognition to convert the audio to text. Return the recognized command or None if there's an issue.

    f. Publish Command Function:
    Publish the recognized command to the input/voice MQTT topic in Node-RED using the publish.single function.

    g. Main Execution:
    Enter an infinite loop to continuously listen for voice commands. If a command is recognized, publish it to the MQTT broker. Add a 2-second delay between iterations.

3. Node-RED Integration:

    a. Requirements:
        i. Raspberry Pi 4 ‘server’
        ii. Node-RED
        iii. DHT22

    b. Set Up Node-RED:
    Install Node-RED on a Raspberry Pi 4.

    c. Configure MQTT Nodes:
    Set up MQTT nodes to receive voice commands from input/voice.
    And send MQTT to esp32 home automation system

    d. Node-RED Flow:

        i. Create a user interface for monitoring and controlling the home automation system.
        ii. Add dashboard elements for lights and fans in each room.
        iii. Implement controls and feedback mechanisms based on MQTT data.
        iv. Display DHT22 sensor readings in the dashboard using appropriate dashboard nodes.

4. Usage:

   a. Deploy the ESP32 system .

   b. Run the Python script on a device with a microphone.

   c. Speak voice commands to control lights and fans.
      i. Turn on room 1 / 2 light / fan
      ii. Room 1 /2 light / fan 0% - 100%

   d. Utilize Node-RED on the Raspberry Pi 4 server for additional processing, UI features, and DHT22 sensor monitoring.

Progress Record:

12/11/23 - System Architecture Drafting:
Following initial research, the team dedicated the first week to sketching out the system architecture plan. This involved envisioning the integration of ESP32, voice control using Python, and Node-RED for centralized control.

19/11/23 - MQTT Protocol Troubleshooting and Prototype Planning:
In the second week, the team delved into troubleshooting MQTT protocol-related issues. Simultaneously, extensive planning took place to refine the prototype. Discussions centered around incorporating voice control seamlessly into the system, ensuring a smooth and intuitive user experience. This phase was critical for setting the direction of the project.

26/11/23 - Hardware Implementation and Node-RED Dashboard Development:
With the prototype plan in hand, the third week saw the transition to hardware implementation and testing. Additionally, a Node-RED dashboard was developed to facilitate centralized control. While most aspects progressed smoothly, a minor challenge emerged in implementing brightness and speed control via voice commands, prompting to brainstorm solutions.

30/11/23 - Project Completion:
The final week marked the successful completion of the project. The team addressed the remaining challenges, and the system was refined to meet the envisioned specifications. With hardware implementation, dashboard and voice control had finalized.

Conclusion:

The IoT-Based Voice Controlled Home Automation system provides an efficient and customizable solution for home automation. By combining the power of ESP32, Python, and Node-RED, the system offers a seamless user experience. The modular architecture allows for easy adaptation to specific home automation setups. The integration of voice control and sensor monitoring enhances the overall functionality and usability of the system.

Reference:

MQTT: The Standard for IoT Messaging

Node-RED and MQTT Prtocol for IoT Crash course ‘Arabic’

Node-RED Tutorials

Node-RED Json

How to Publish DHT11 Sensor Data from NodeMCU to Mosquitto MQTT Broker over LAN | NodeMCU | MQTT |

MQTT tutorial on Raspberry pi, Arduino and Python

Node-red-dashboard documentation

ESP32 MQTT – Publish and Subscribe with Arduino IDE