Lab 2

ссылка на github ; [https://github.com/Muhammeddada120/Labs\_internet-Things](https://github.com/EyAdOoOoOo/lab2/blob/main/lab2.py)

Аттиа Мохамед Магди Абдельхамид мохамед Отчёт

# 1. Цель

Целью этой лабораторной работы является симуляция IoTустройства со следующими возможностями:

* Симуляция данных датчиков и поведения исполнительного устройства.
* Предоставление графического интерфейса пользователя (GUI) для управления устройством.
* Связь с открытым MQTT-сервером для передачи данных в реальном времени и управления.

# 2. Описание разработанной программы

Программа симулирует IoT-устройство, предназначенное для измерения влажности почвы и управления водяным насосом. Она поддерживает два режима работы:

1. **Ручной режим:** Исполнительное устройство (насос) может быть включено или выключено вручную через GUI или команды MQTT.
2. **Автоматический режим:** Насос автоматически активируется или деактивируется на основе заданных порогов датчиков.

Программа включает следующие основные компоненты:

* **Графический интерфейс (GUI):** Создан с использованием tkinter, отображает данные датчиков в реальном времени, состояние насоса и текущий режим.
* **Симуляция датчиков:** Генерирует значения влажности почвы, которые уменьшаются со временем, если насос не активен.
* **MQTT-коммуникация:** Публикует данные датчиков на брокер и получает команды управления исполнительным устройством и изменения режимов.

# 3. Возможности Графический интерфейс пользователя

* Отображает текущий уровень влажности почвы.
* Отображает состояние насоса (ВКЛ/ВЫКЛ).
* Позволяет вручную управлять насосом (Включить/Выключить).
* Отображает текущий режим работы (Ручной/Автоматический).

# Интеграция с MQTT

* **Брокер:** Используется открытый MQTT-сервер test.mosquitto.org.
* **Топики:**
  + iot/sensor/soil\_moisture: Публикация данных о влажности почвы.
  + iot/actuator/pump\_command: Подписка на команды управления насосом (ВКЛ/ВЫКЛ).
  + iot/mode/set\_mode: Подписка на сообщения управления режимом (Ручной/Автоматический).
  + iot/status/pump\_state: Публикация текущего состояния насоса.
* **Период передачи данных:** Данные датчиков публикуются каждые 10 секунд.

# Режимы работы

* **Ручной режим:** Насос работает только при ручной активации.
* **Автоматический режим:** Насос включается, если влажность почвы падает ниже 30%, и выключается, если превышает 40%.

# 4. Реализация Используемые технологии •Язык программирования: Python

•**Библиотеки:**

otkinter для GUI opaho-mqtt для MQTT-коммуникации othreading для параллельного выполнения

# Структура кода

* **Настройка MQTT:** Обрабатывает подключение, подписку и обработку сообщений.
* **Симуляция датчиков:** Имитирует реалистичное изменение уровня влажности почвы в зависимости от активности насоса.
* **GUI:** Обеспечивает удобное взаимодействие пользователя с устройством.

# Логика выполнения

1. Программа инициализируется и подключается к MQTTброкеру.
2. Данные датчиков обновляются каждые 10 секунд и публикуются на брокер.
3. GUI обновляется каждую секунду для отображения изменений в реальном времени.
4. Полученные команды MQTT обрабатываются для управления насосом и изменения режима.

# 5. Результаты Успешные результаты

* Устройство успешно симулирует данные датчиков и поведение исполнительного устройства.
* Данные передаются на MQTT-брокер и видны в сторонних клиентах MQTT (например, MQTT Explorer).
* Состояние насоса и режим могут быть управляемы как через GUI, так и через команды MQTT.

# Наблюдения

* В ручном режиме насос работает только по явным командам.
* В автоматическом режиме насос поддерживает оптимальный уровень влажности почвы на основе заданных порогов.

# 6. Проблемы и решения Проблема: Проблемы совместимости с MQTT

**Решение:** Обновлена библиотека paho-mqtt до последней версии и адаптированы функции обратного вызова. **Проблема: Обновления GUI в реальном времени**

**Решение:** Использован метод after библиотеки tkinter для обеспечения плавного обновления интерфейса без блокировки других операций.

7. Как запустить программу

1. Установите зависимости:

pip install paho-mqtt

2.Сохраните программу как iot\_mqtt\_simulator.py.

3. Запустите программу: python iot\_mqtt\_simulator.py

4.Используйте клиент MQTT для тестирования команд:

Публикуйте ON или OFF в iot/actuator/pump\_command.

Публикуйте Manual или Automatic в iot/mode/set\_mode.

# 8. Заключение

Разработанный симулятор IoT эффективно демонстрирует интеграцию с MQTT-сервером, предоставляя основу для реальных приложений IoT. Он поддерживает интерактивное управление через GUI и удаленные команды, показывая потенциал IoT-систем для автоматизированного мониторинга и управления окружающей средой.