|  |
| --- |
| **Laporan Proyek**  **Internet of Things Smart Home**  ***“Parkir Mobil”*** |
|  |
| **Kelompok 24**   |  |  | | --- | --- | | **13321007** | **Albert Panggabean** | | **13321045** | **Cindy Thresya Situmeang** | |
| **INSTITUT TEKNOLOGI DEL**  **D3 TEKNOLOGI KOMPUTER**  **FAKULTAS VOKASI 2023** |

# **Smart Home “Sistem Parkir Mobil”**

# **Sistem Parkir Mobil menggunakan Sensor Ultrasonik**

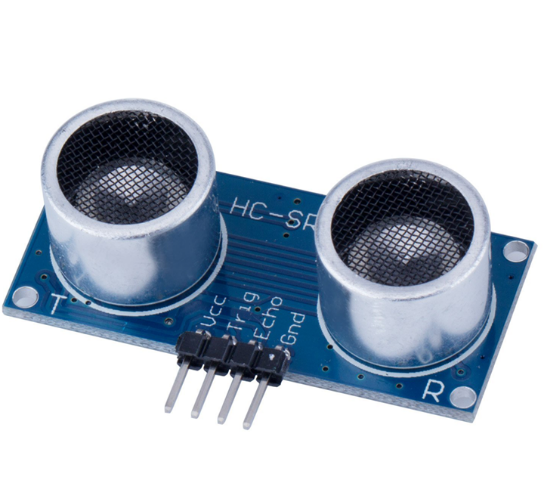
Sistem parkir adalah perangkat elektronik yang mendeteksi jarak antara kendaraan dan objek di sekitarnya saat parkir. Sistem parkir dengan sensor ultrasonik mengurangi risiko kerusakan akibat tabrakan, terutama saat parkir di tempat yang sempit. Selain itu, sistem ini meningkatkan kenyamanan dan kemudahan parkir bagi pengemudi, terutama bagi mereka yang mengalami kesulitan bermanuver di tempat parkir.

## **Komponen**

Berikut ini akan ditampilkan terkait dari komponen pada sistem Parkir Mobil.

* **Sensor Ultrasonik**

Sensor ultrasonik adalah perangkat elektronik yang menggunakan gelombang ultrasonik untuk mengukur jarak antara sensor dan objek di sekitarnya. Sensor ultrasonik banyak digunakan dalam sistem parkir untuk membantu pengemudi dalam melakukan manuver parkir dengan memberikan informasi tentang jarak antara mobil dan objek di sekitarnya.



**Gambar 1. Sensor Ultrasonik**

Berikut ini adalah definisi dan penggunaan sensor ultrasonik dalam sistem parkir:

1. **Deteksi jarak**

Sensor ultrasonik digunakan dalam parkir digunakan untuk mengukur jarak antara sistem mobil dan objek di sekitarnya.

Hal ini memungkinkan pengemudi untuk mengetahui seberapa dekat mereka dengan objek seperti dinding atau kendaraan lain saat parkir.

1. **Peringatan dan Alarm**

Jika jarak antara kendaraan dan suatu objek terlalu dekat, sensor ultrasonik akan mengeluarkan peringatan suara atau visual untuk memperingatkan pengemudi tentang risiko tabrakan. Hal ini dapat mencegah kerusakan pada mobil Anda dan benda-benda di sekitarnya.

1. **Bantuan Parkir Otomatis**

Beberapa sistem parkir memiliki fitur bantuan parkir otomatis yang menggunakan sensor ultrasonik untuk mendeteksi tempat parkir yang luas. Sistem ini memberikan instruksi kepada pengemudi tentang cara memarkir kendaraan dengan aman.

1. **Panduan Visual**

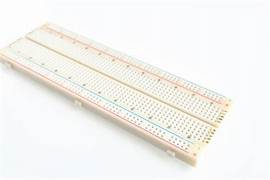
Sensor ultrasonik sering diintegrasikan ke dalam layar mobil dan memberikan representasi grafis dari objek di sekitarnya. Panduan visual ini membantu pengemudi melihat seberapa dekat objek.

1. **Penghindaran Tabrakan**

Sensor ultrasonik membantu mencegah tabrakan dan benturan dengan objek di sekitarnya selama manuver parkir, terutama di tempat yang sempit

## **Breadboard**

Breadboard digunakan sebagai tempat untuk menempatkan komponen-komponen elektronik seperti sensor ultrasonik.



**Gambar 2. Breadboard**

* **Raspberry pi 4 model 8**

Raspberry Pi dapat digunakan dalam berbagai proyek otomatisasi, termasuk sistem parkir mobil. Dalam konteks sistem parkir mobil, Raspberry Pi dapat berperan sebagai otak pusat yang mengendalikan dan memantau berbagai komponen, termasuk sensor, kamera, dan tindakan otomatisasi.



**Gambar 3. Raspberry Pi**

* **Kabel Jumper**

Jumper digunakan untuk menghubungkan komponen yang digunakan dan memastikan bahwa sistem parkir mobil dapat beroperasi dengan aman dan efektif.



**Gambar 4. Kabel Jumper**

* **Buzzer**

Buzzer pada sistem parkir mobil berfungsi sebagai alarm yang akan berbunyi ketika jarak mobil mendekati benda pembatas pada suatu tempat parkir. Buzzer ini akan memberikan peringatan kepada pengemudi mobil untuk segera menghentikan mobil dari pembatas dan membuat jarak antara mobil dengan pembatas. Prinsip kerja buzzer pada sistem parkir mobil adalah dengan menghubungkan buzzer pada mikrokontroler atau relay.



**Gambar 5. Buzzer**

* **Relay**

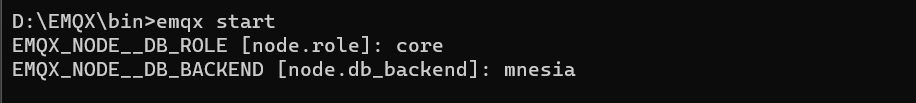
Relay pada sistem parkir mobil berfungsi sebagai penghubung antara panel kontrol dan buzzer. Relay ini akan mengaktifkan buzzer ketika adanya jarak yang terlalu dekat dengan mobil.

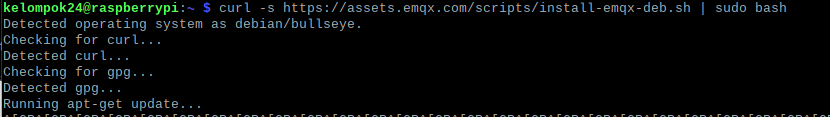
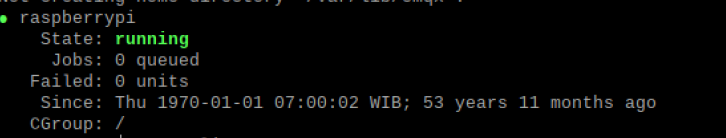


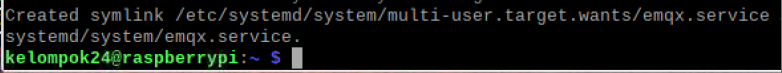
**Gambar 6. Relay**

## **Cara Instalasi**

1. **Dashboard EMQX**
2. Cara Running

* Masuk ke emqx/bin lalu CMD
* Ketikan Perintah 

1. Akses menggunkan ip windows dan port <http://192.168.89.111:18083>
2. **Setting Raspberry Pi**
3. Download the emqx repository
4. Install EMQX



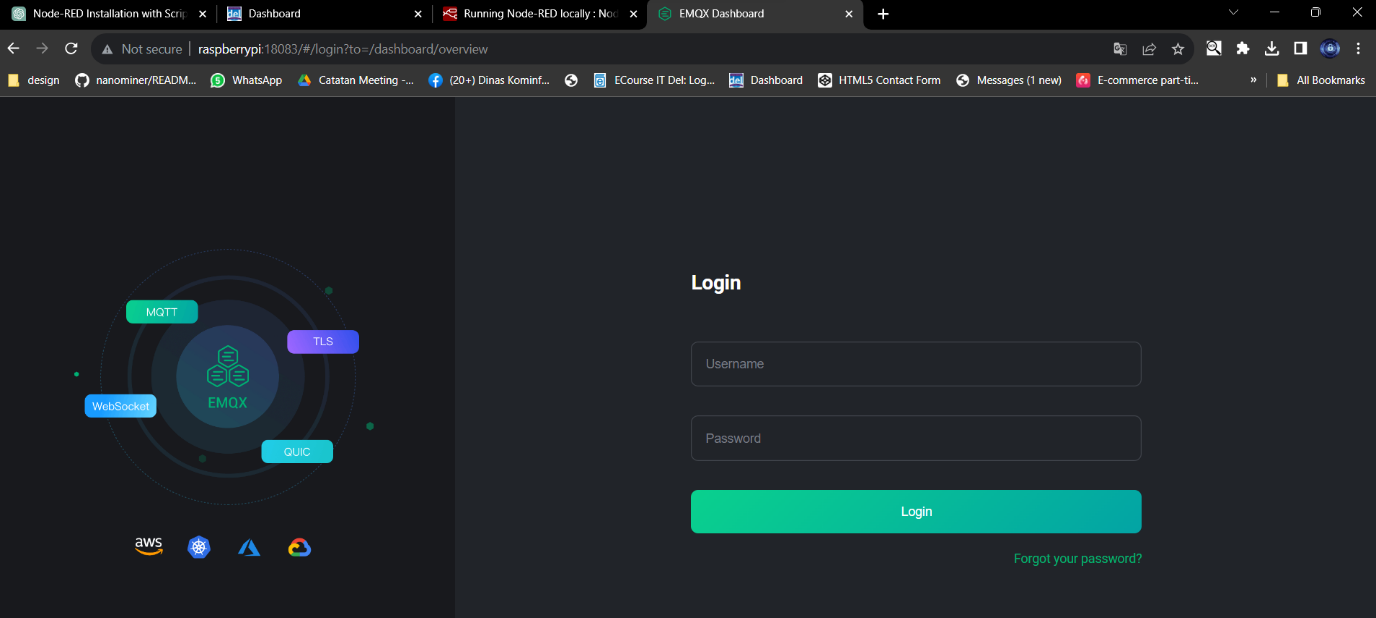
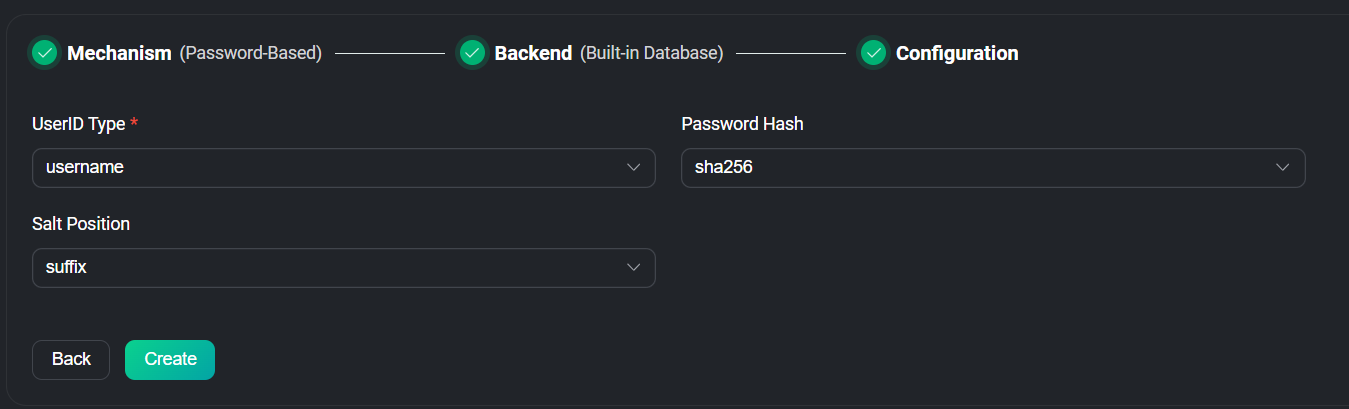
1. Run EMQX dari Rasoberry

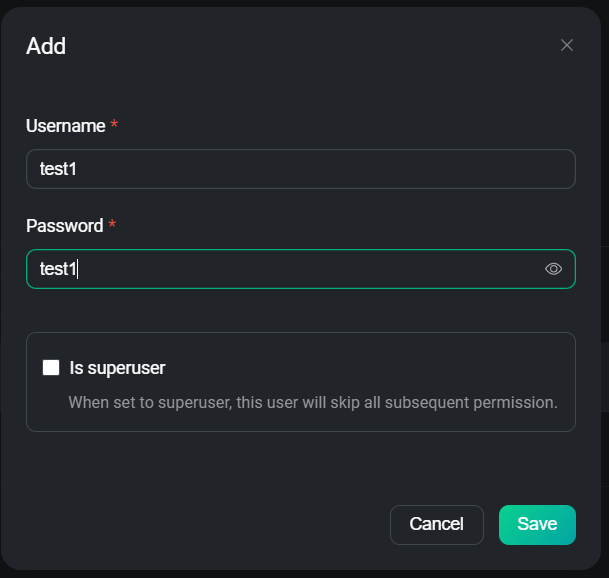


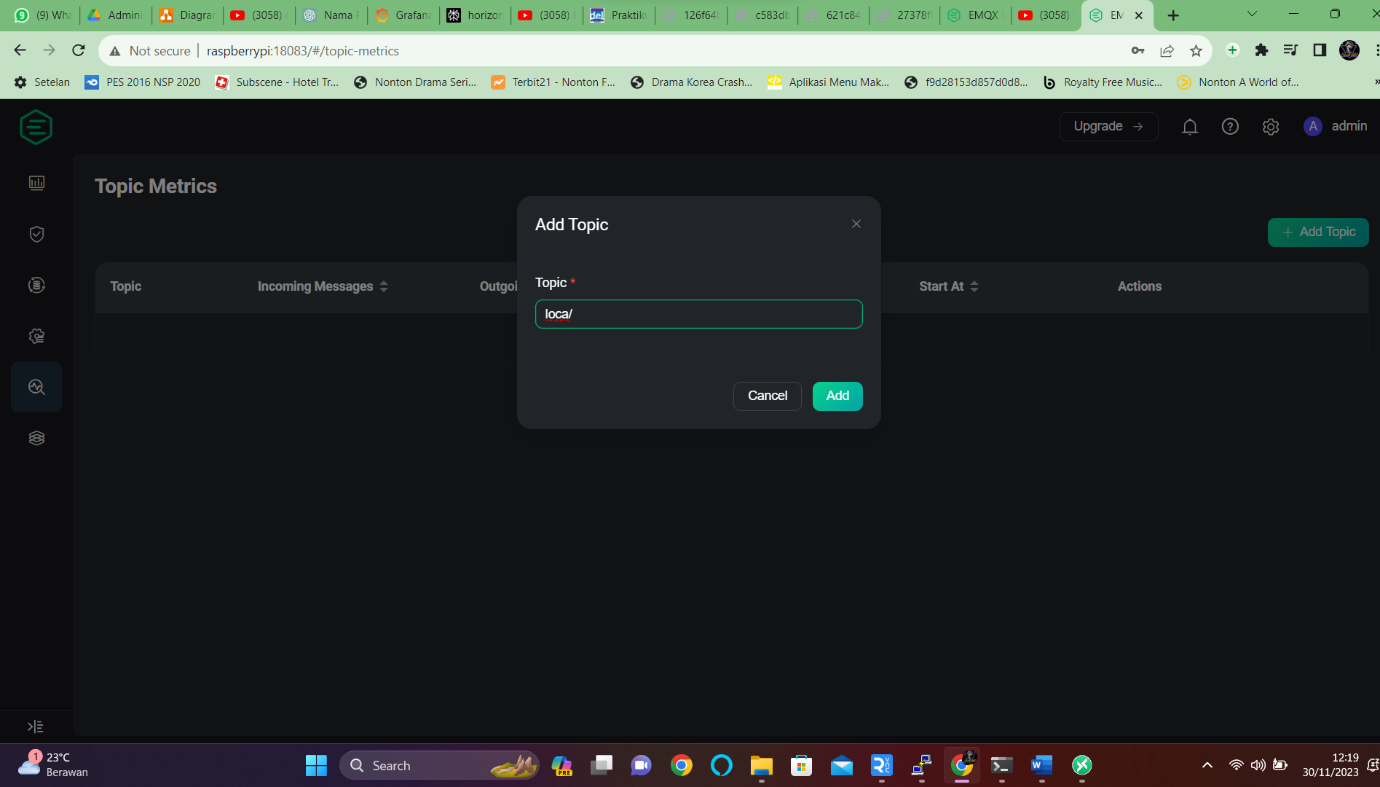
1. Buka dashboard **raspberrypi:18083**

Username: **”Admin”**

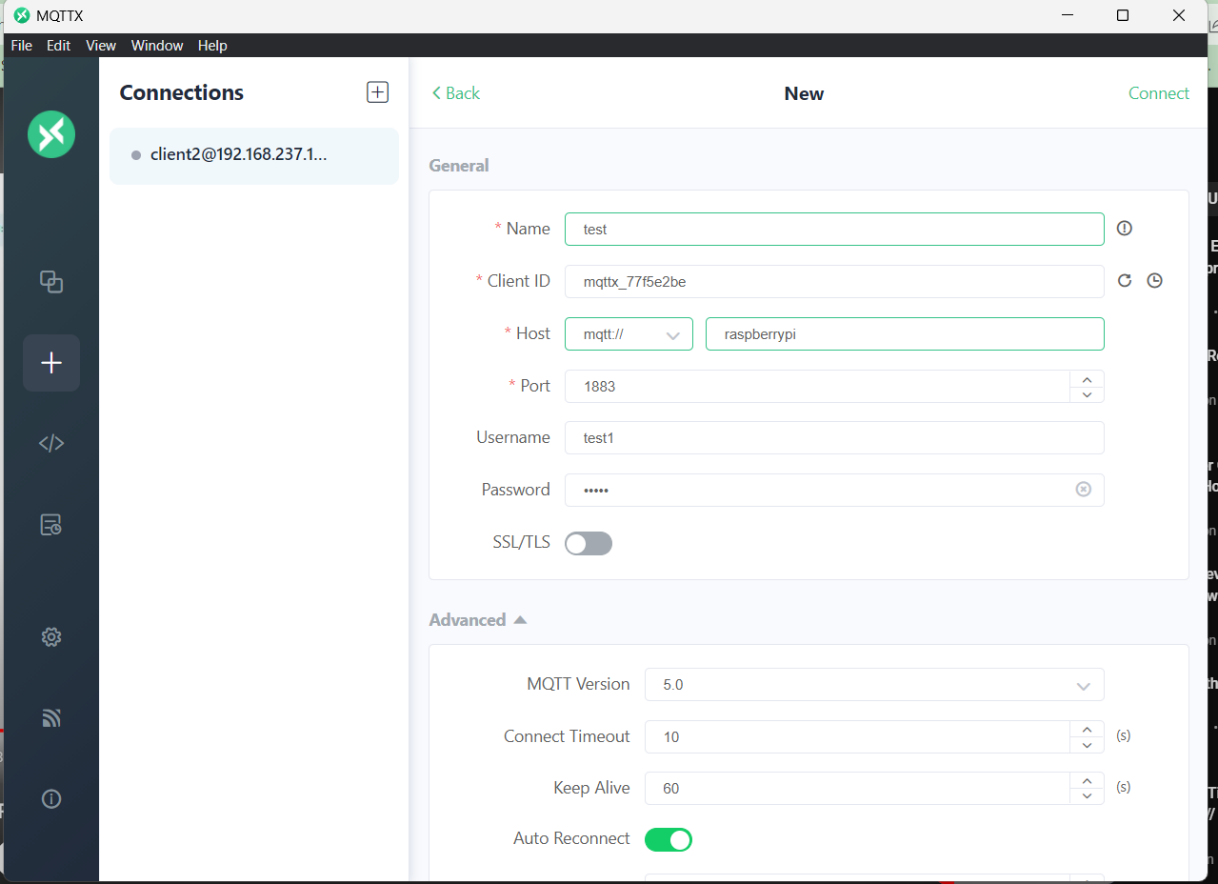
Password: **“Kelompok24”**

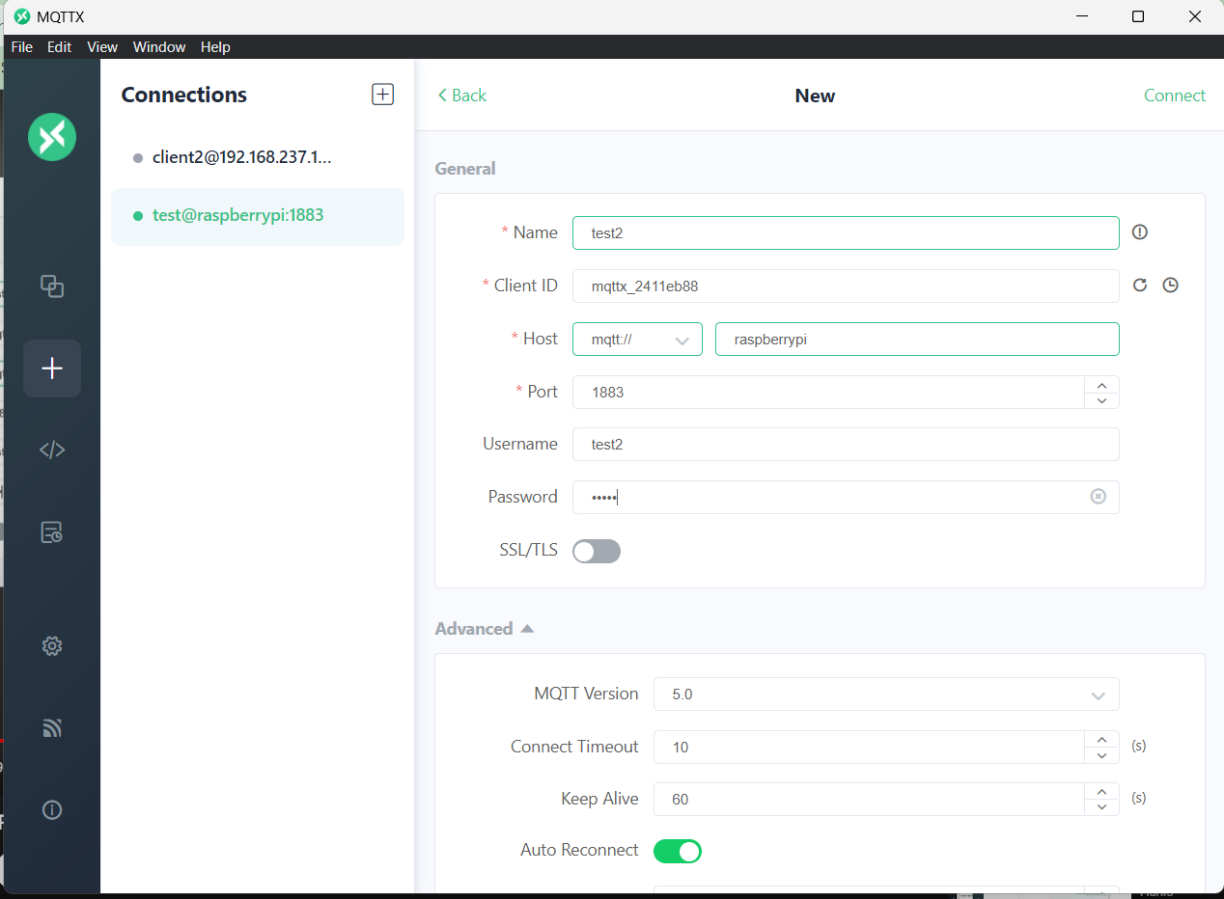
****

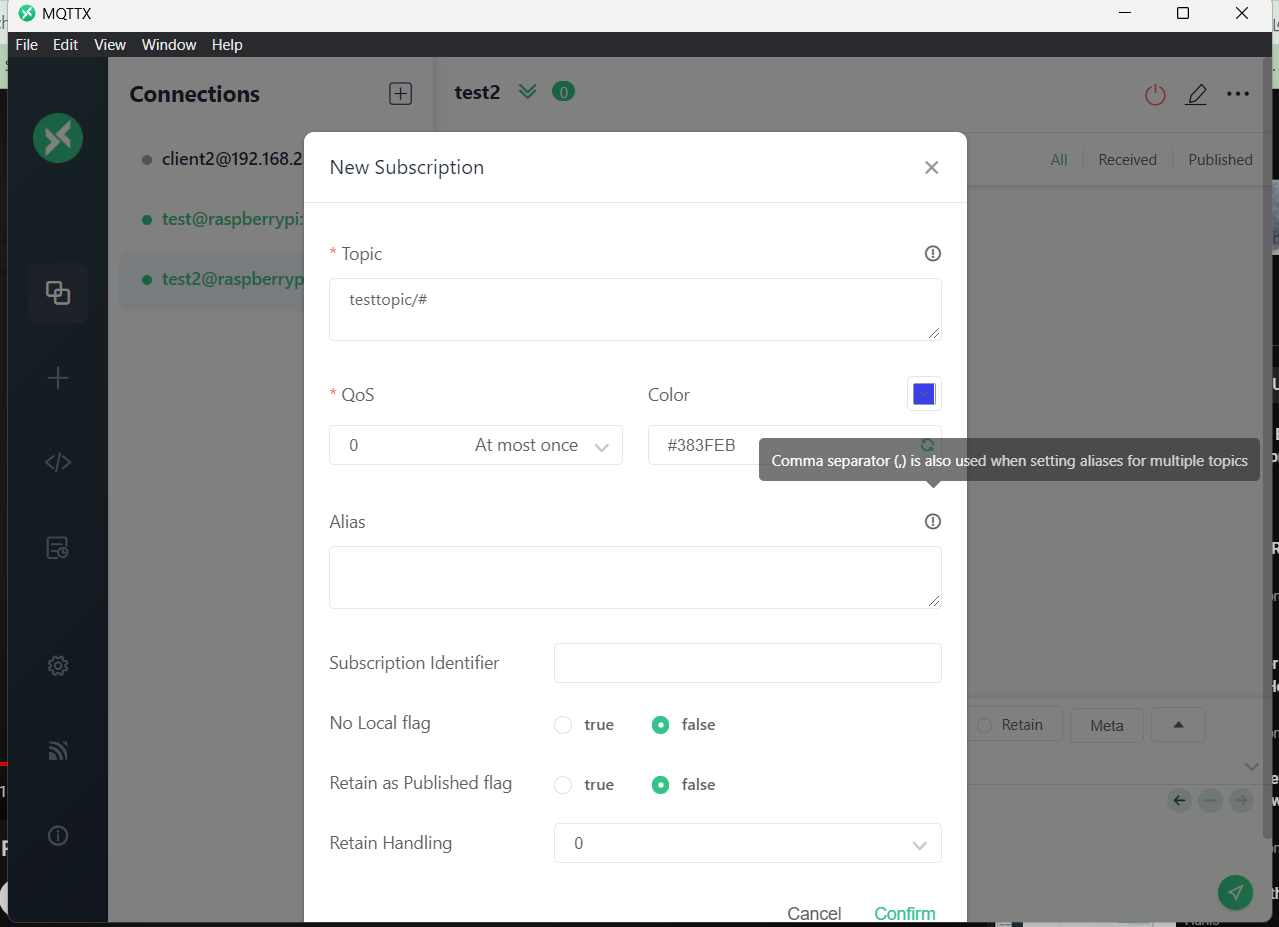


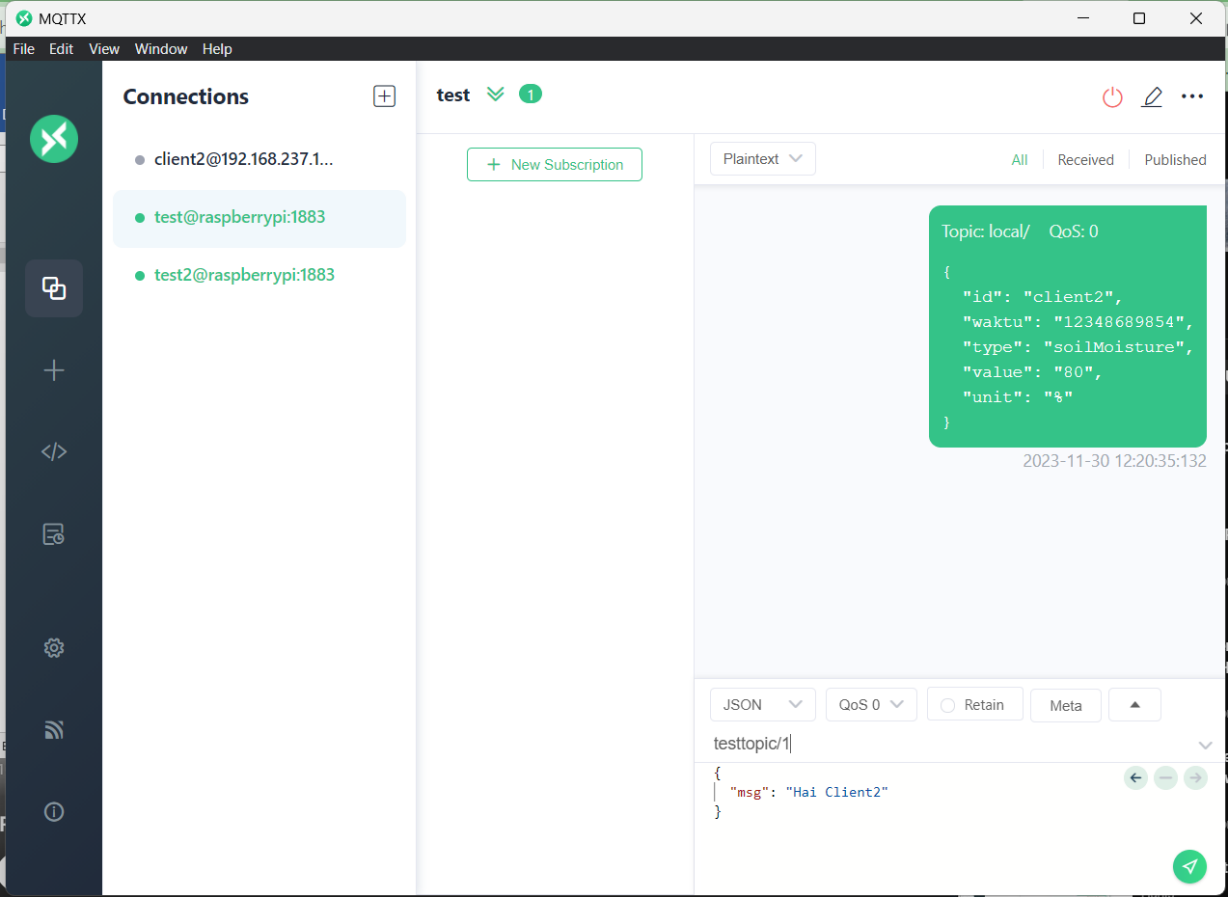


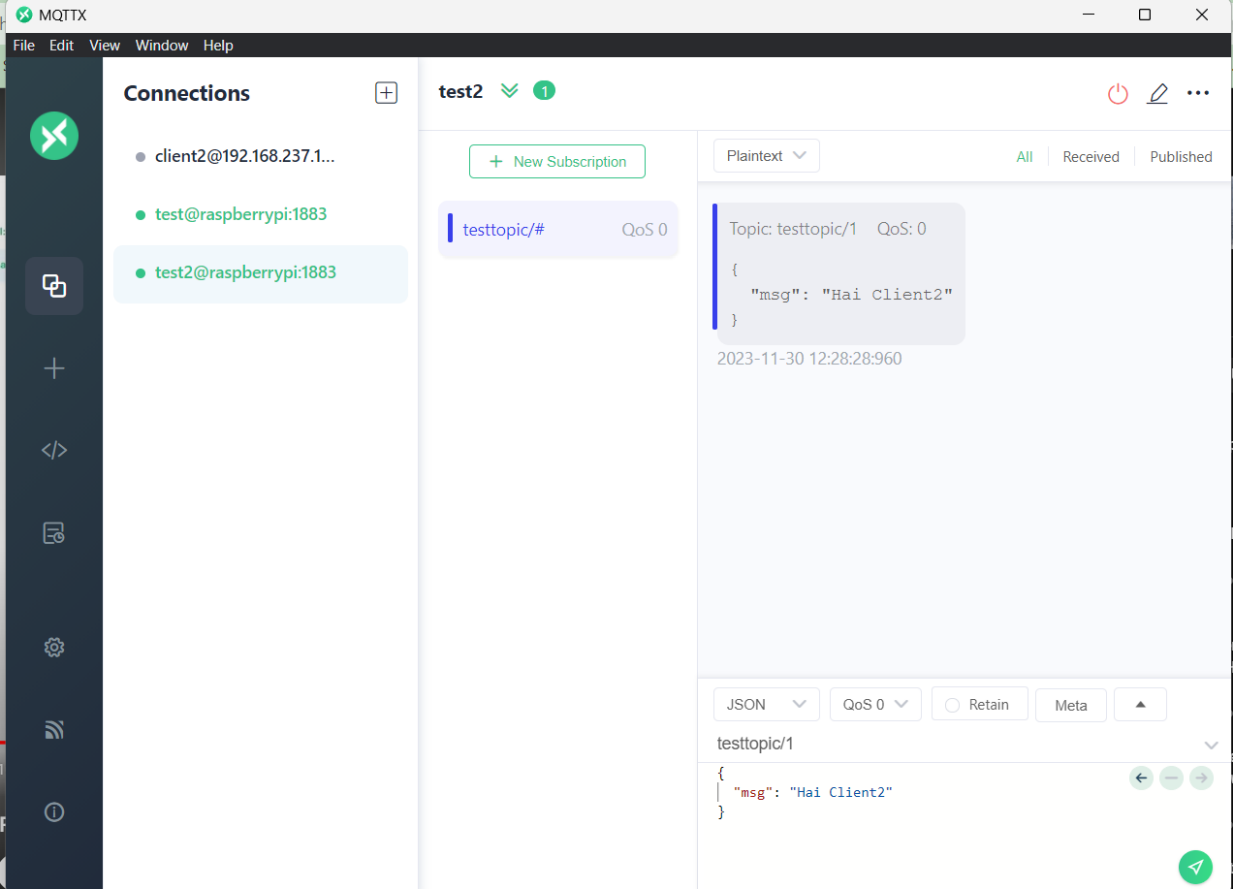
1. **Setting MQTTX**

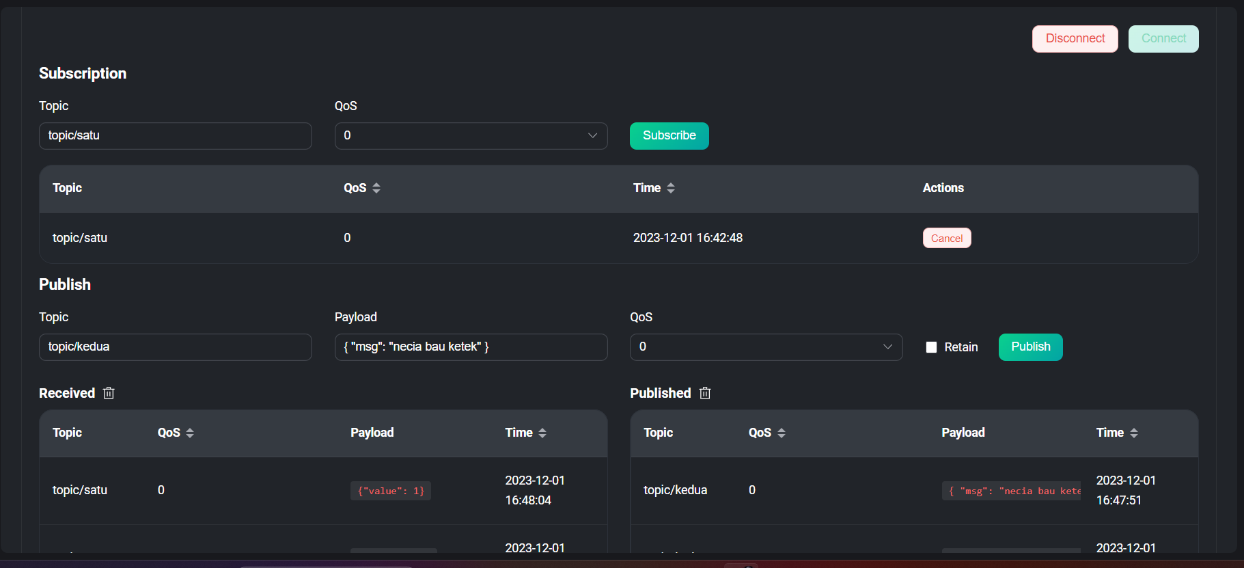




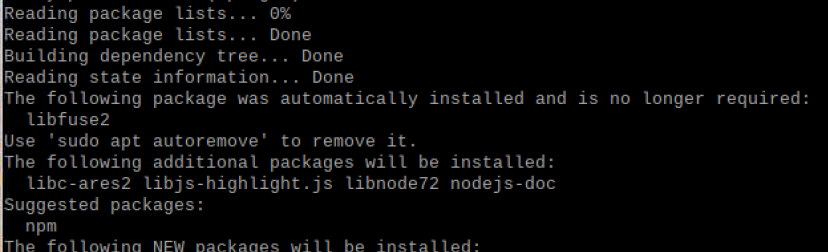


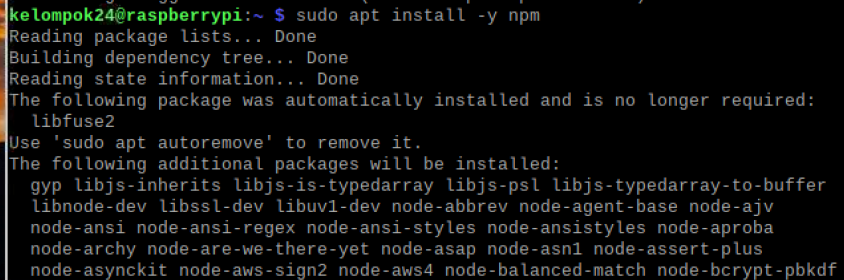
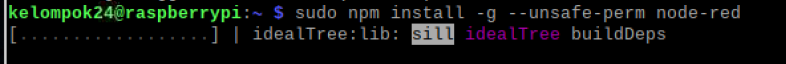


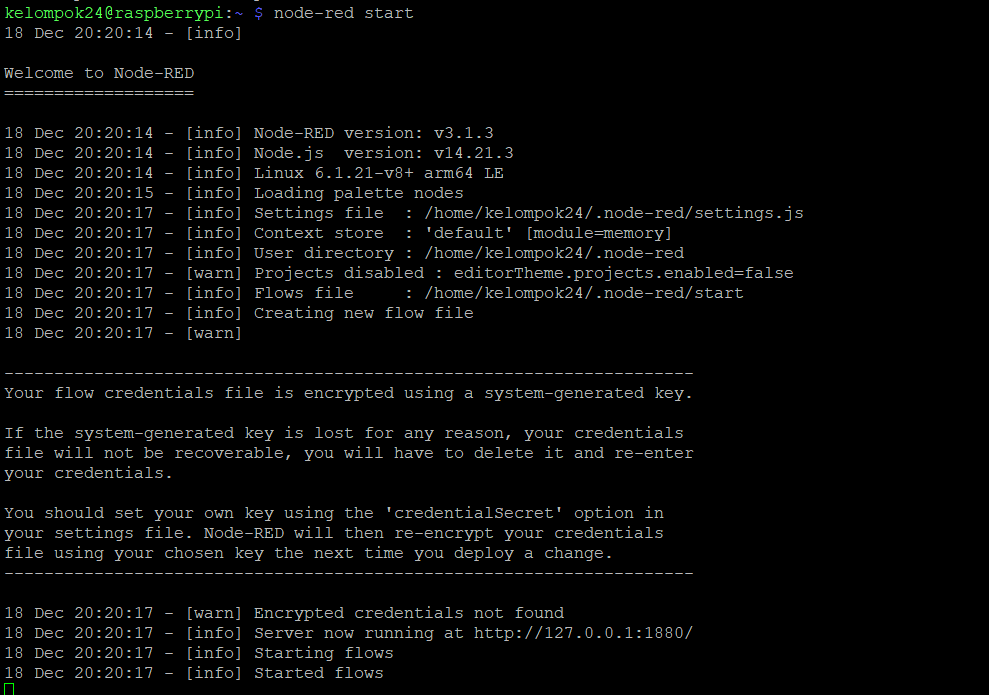




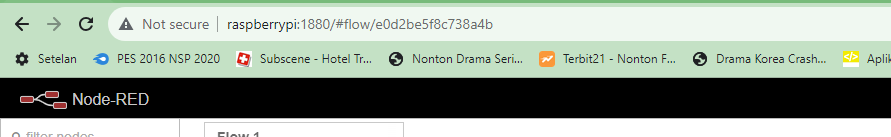
1. **Install nodejs**



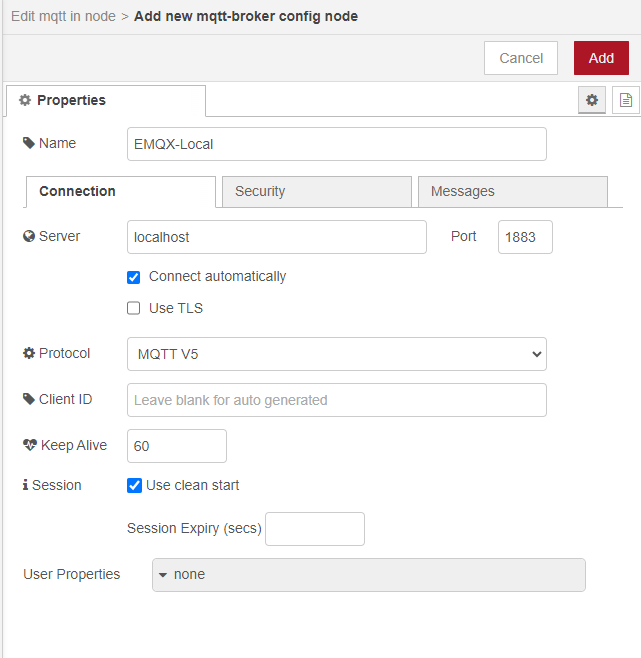
1. **Install npm** 
2. **Install Node-Red**
3. **Settingan Node Red**
4. bash <(curl -sL <https://raw.githubusercontent.com/node-red/linux> installers/master/deb/update-nodejs-and-nodered)
5. node-red-start > untuk running

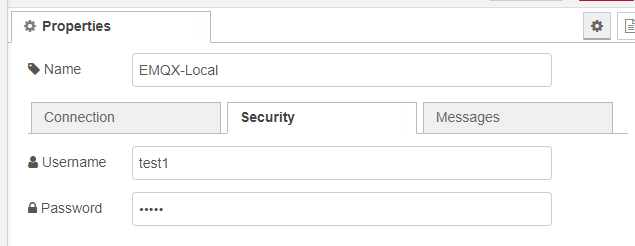


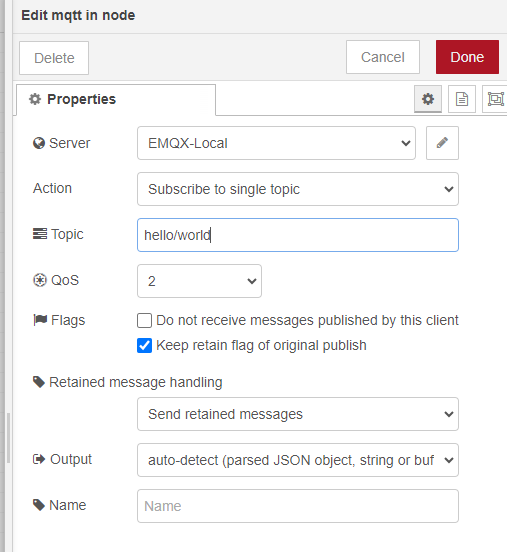
1. **Cara Membuka Tampilan Node-Red pada Raspberrypi**



1. **Untuk MQTT IN**



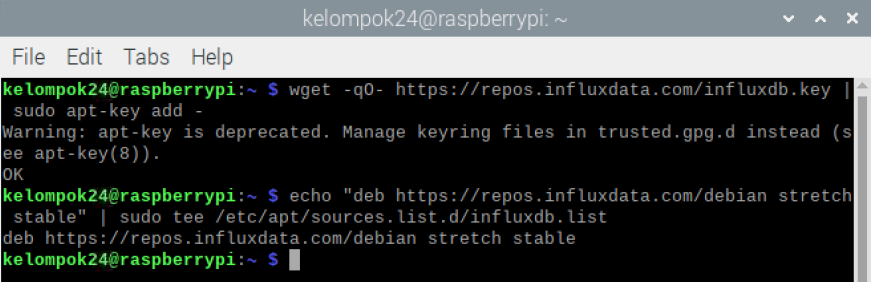




1. **Untuk MQTT OUT**



1. **Setting influxdb**
2. Cara install dan running

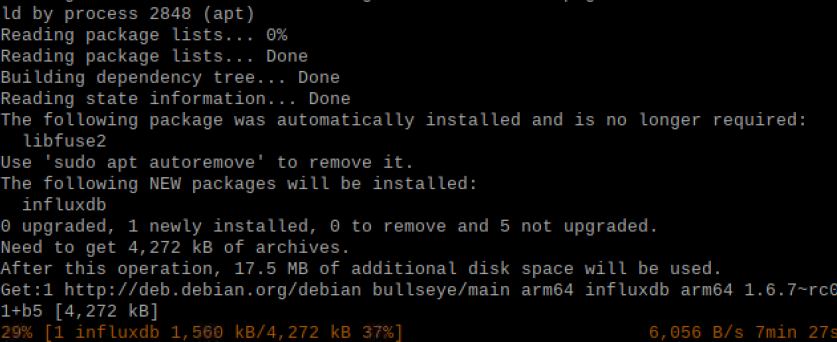
* wget -qO- https://repos.influxdata.com/influxdb.key | sudo apt-key add –
* echo "deb https://repos.influxdata.com/debian stretch stable" | sudo tee /etc/apt/sources.list.d/influxdb.list

1. Sudo apt update

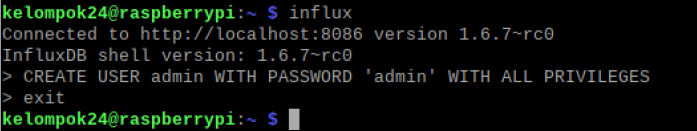


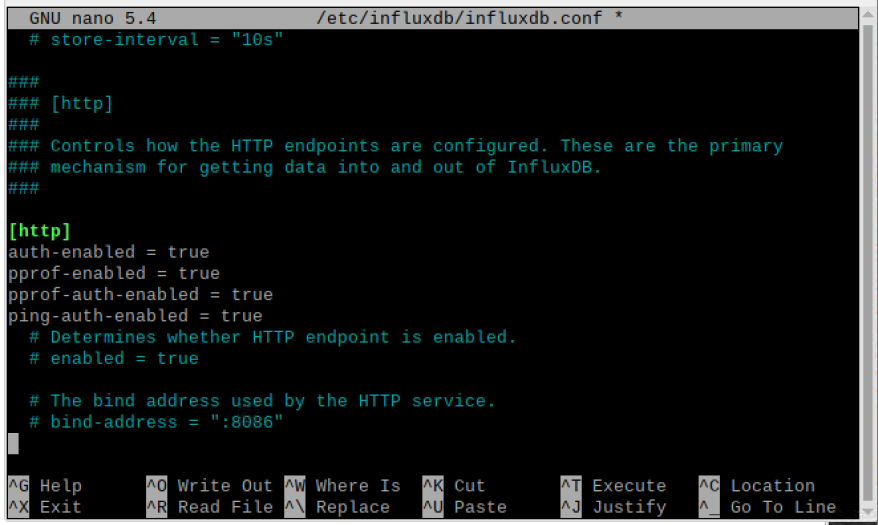
1. Sudo apt install influxdb

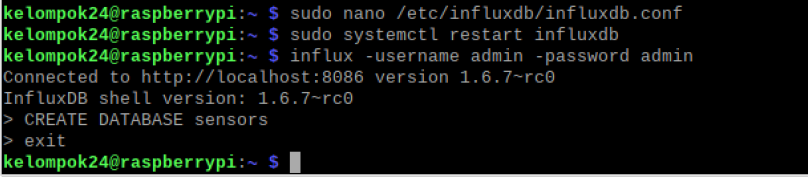


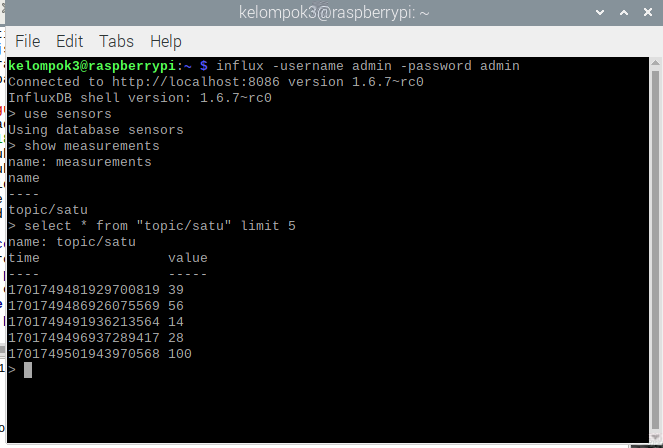


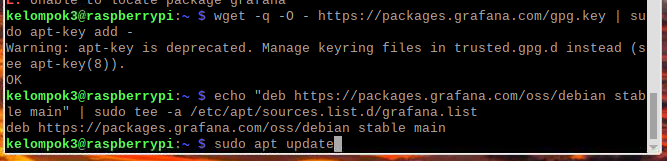
 

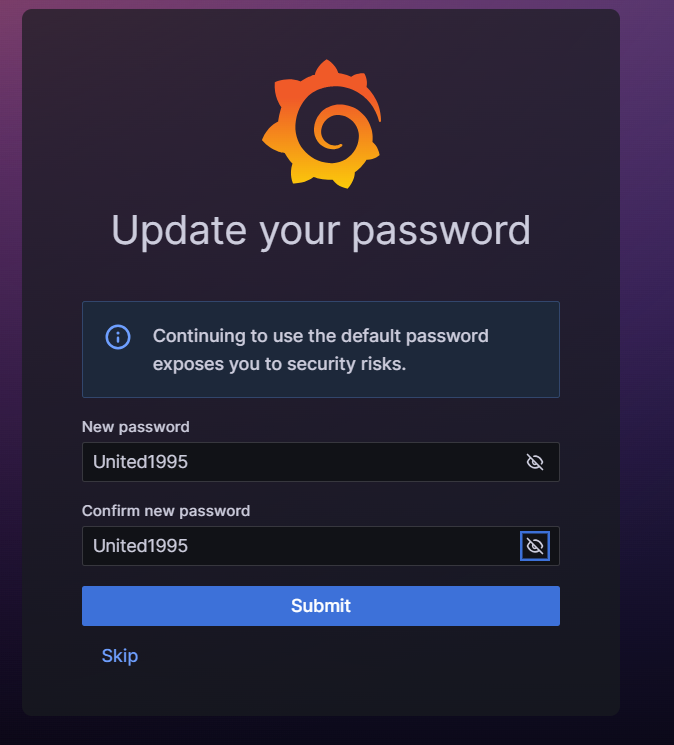


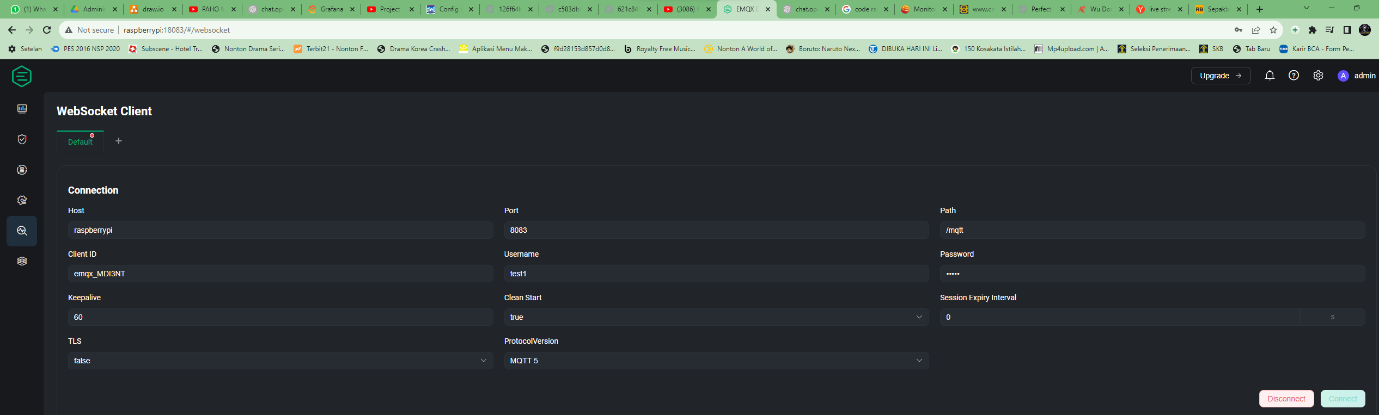


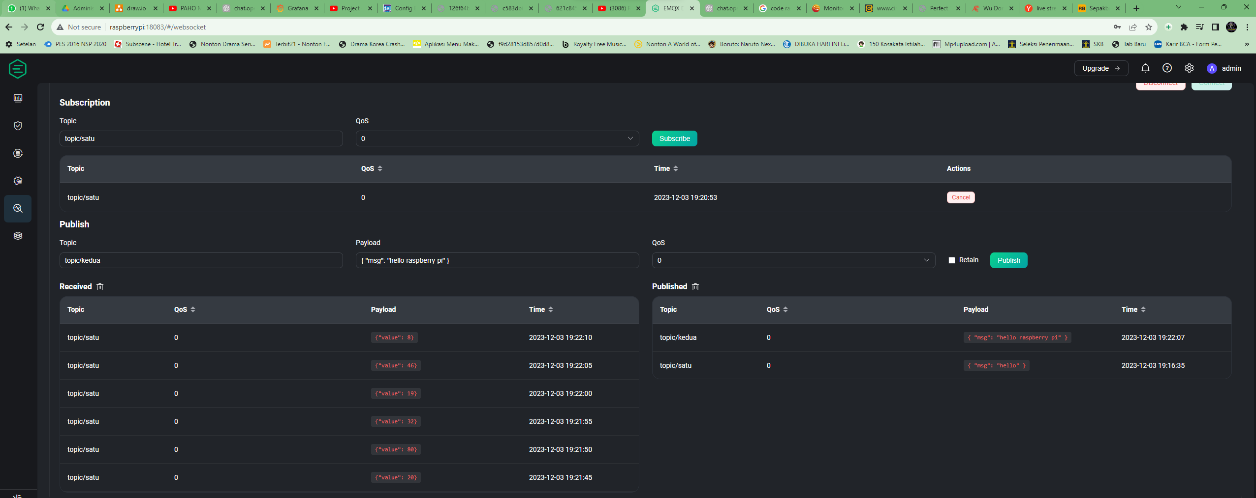


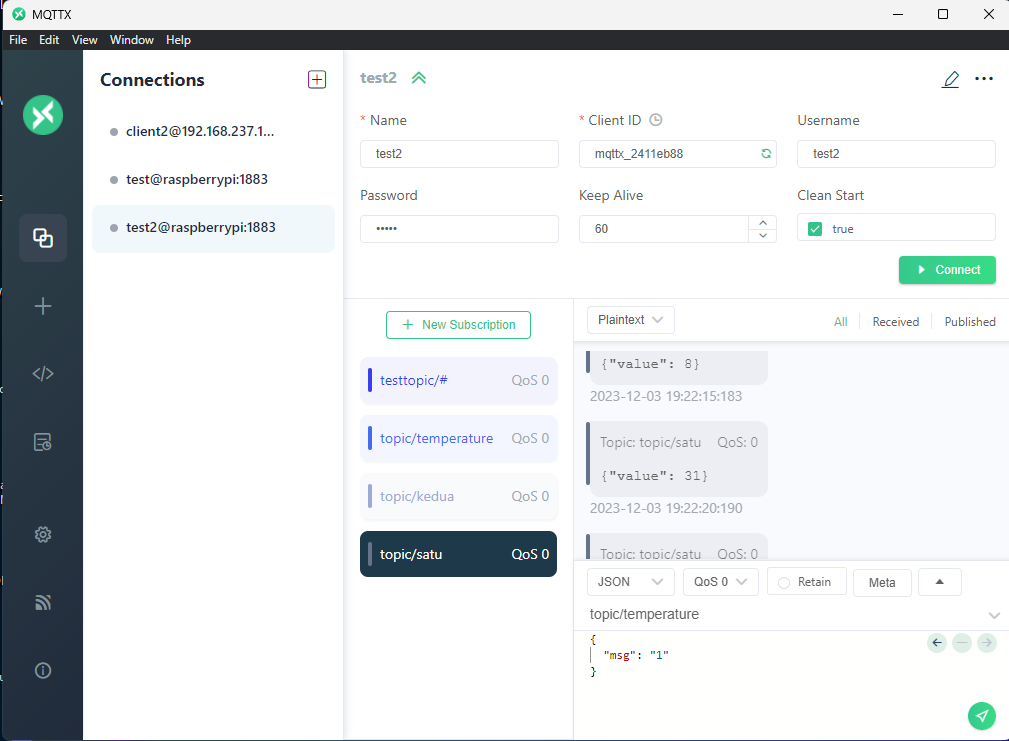


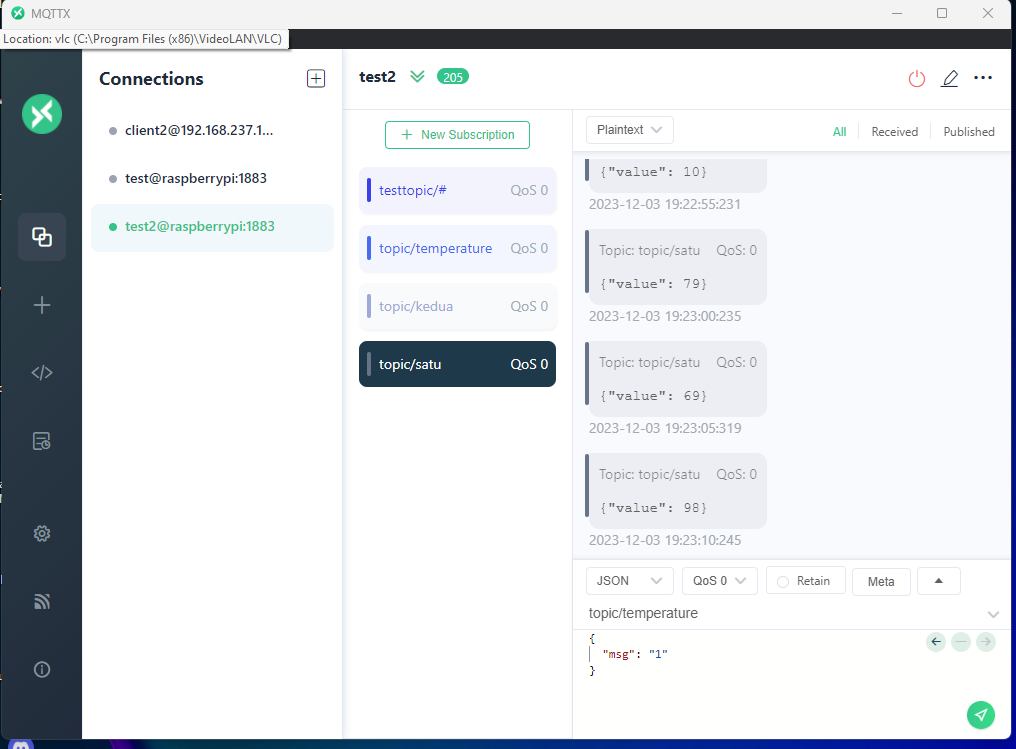
1. **Setting Grafana**

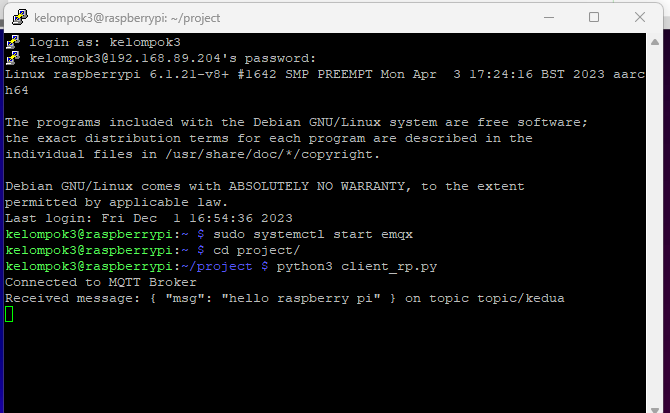


1. **Settingan Fix**

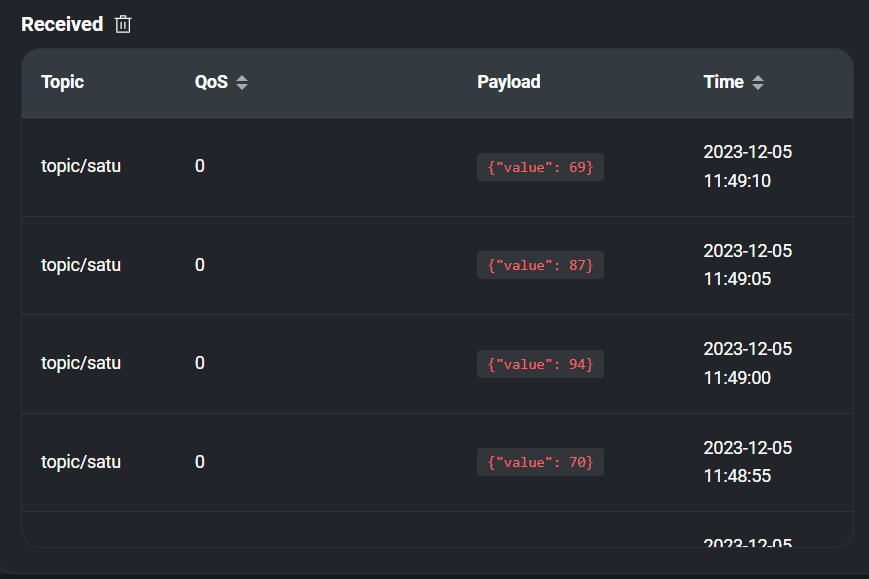


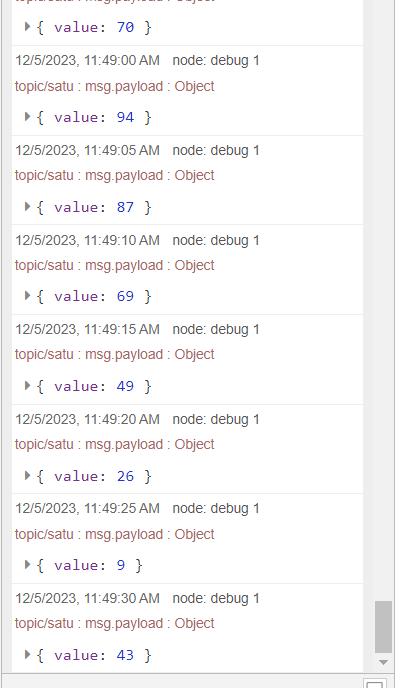


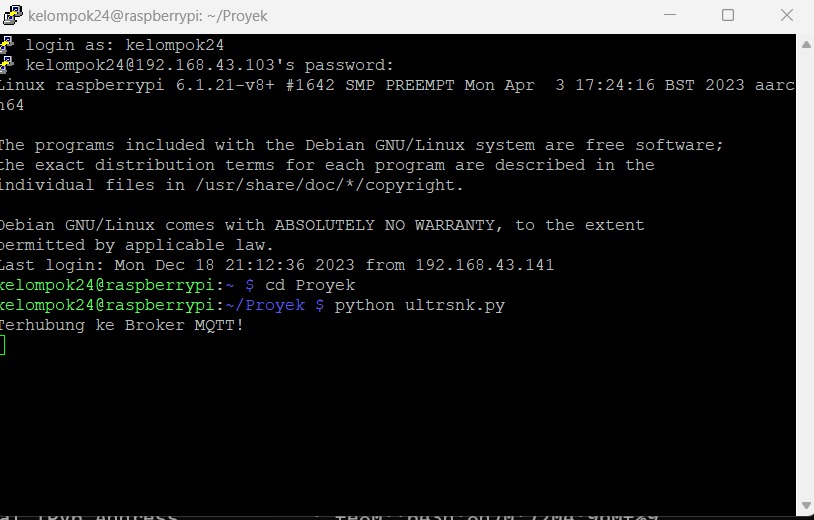












## **Kode Program**

|  |
| --- |
| #Kelompok 24  # 13321007 - Albert Panggabean  # 13321045 - Cindy Thresya Situmeang  import time  import json  import paho.mqtt.client as mqtt  import RPi.GPIO as GPIO  broker = '192.168.43.44'  # Ganti dengan alamat IP broker MQTT Anda  port = 1883  topic\_publish = "sensor/"  client\_id = 'python-mqtt'  username = 'test'  password = 'test'  GPIO.setmode(GPIO.BCM)  TRIG = 17  # Pin trigger sensor ultrasonik  ECHO = 27  # Pin echo sensor ultrasonik  BUZZER = 22  # Pin buzzer  # Konfigurasi pin buzzer sebagai output  GPIO.setup(BUZZER, GPIO.OUT)  GPIO.output(BUZZER, GPIO.LOW)  # Matikan buzzer pada awalnya  first\_connection = True  # Variabel untuk menandai koneksi pertama  def connect\_mqtt():      def on\_connect(client, userdata, flags, rc):          global first\_connection          if rc == 0 and first\_connection:  # Hanya kirim pesan jika koneksi pertama berhasil              print("Terhubung ke Broker MQTT!")              first\_connection = False          elif rc != 0:              print("Gagal terhubung, kode return %d\n" % rc)      client = mqtt.Client(client\_id)      client.username\_pw\_set(username, password)      client.on\_connect = on\_connect      client.connect(broker, port)      return client  def read\_distance():      GPIO.setup(TRIG, GPIO.OUT)      GPIO.setup(ECHO, GPIO.IN)      GPIO.output(TRIG, False)      time.sleep(0.1)      GPIO.output(TRIG, True)      time.sleep(0.00001)      GPIO.output(TRIG, False)      pulse\_start = time.time()      pulse\_end = time.time()      while GPIO.input(ECHO) == 0:          pulse\_start = time.time()      while GPIO.input(ECHO) == 1:          pulse\_end = time.time()      pulse\_duration = pulse\_end - pulse\_start      distance = pulse\_duration \* 17150      distance = round(distance, 2)      return distance  def control\_buzzer(distance):      if distance < 20:  # Ubah nilai jarak sesuai kebutuhan          GPIO.output(BUZZER, GPIO.HIGH)  # Aktifkan buzzer jika jarak kurang dari 10 cm      else:          GPIO.output(BUZZER, GPIO.LOW)  # Matikan buzzer jika jarak lebih dari 10 cm  def publish\_sensor\_data(client):      while True:          start\_time = time.time()          distance = read\_distance()          ultrasonic\_message = {              "Distance": distance          }          ultrasonic\_msg = json.dumps(ultrasonic\_message)          client.publish(topic\_publish, ultrasonic\_msg)          print(f"Published Ultrasonic Data - Distance: {distance}")          control\_buzzer(distance)  # Kontrol buzzer berdasarkan jarak yang terbaca          end\_time = time.time()          elapsed\_time = end\_time - start\_time          if elapsed\_time < 5:              time.sleep(5 - elapsed\_time)  # Menunggu sisa waktu dalam 1 detik  def run():      client = connect\_mqtt()      client.loop\_start()      publish\_sensor\_data(client)  # Publish data sensor      try:          while True:              # Melakukan apapun yang diperlukan selama program berjalan              time.sleep(1)      except KeyboardInterrupt:          print("Keyboard interrupt detected. Cleaning up GPIO.")          GPIO.cleanup()  # Bagian utama kode  if \_name\_ == '\_main\_':      run() |

1. **Rangkaian**

