

30 Agustus 2024

ALAT UKUR SINYAL LORA UNTUK MENGETAHUI JANGKAUAN ANTARA END DEVICE DENGAN GATEWAY

Delfin Daffa Pebrian
IOT ENGINEER

DAFTAR ISI

DAFTAR ISI.....	1
DAFTAR GAMBAR.....	2
MODUL I ALAT UKUR SINYAL LORA	3
1.1 DASAR TEORI	3
1.2 ALAT DAN BAHAN	5
1.3 INSTALASI DAN KONFIGURASI	8
1.4 ARSITEKTUR DAN CARA KERJA.....	24

DAFTAR GAMBAR

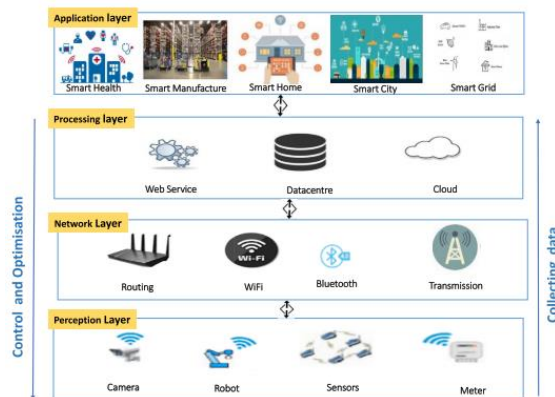
Gambar 3.1 Arsitektur Internet of Things.....	3
Gambar 3.2 LoRa and LoRaWAN Technology Stack	4
Gambar 3.3 Arsitektur LoRaWAN	5
Gambar 3.4 Blok Diagram Protokol MQTT	5
Gambar 3.5 Laptop MSI gf63 Thin.....	6
Gambar 3.6 Gateway Dragino PG1301 Raspberry Pi	6
Gambar 3.7 End Device SenseCAP K1100.....	6
Gambar 3.8 Chirpstack Architecture	7
Gambar 3.9 Platform Openremote	7
Gambar 3.10 Aplikasi Docker.....	8
Gambar 3.11 Arsitektur Alat Ukur Sinyal LoRa	24
Gambar 3.12 Flowchart Alat Ukur Sinyal LoRa	25

MODUL I

ALAT UKUR SINYAL LORA

1.1 DASAR TEORI

- A. Internet of Things merupakan perkembangan ilmu pengetahuan yang sangat menjanjikan untuk mengoptimalkan kehidupan dengan bantuan sensor pintar dan perangkat pintar yang bekerja sama melalui jaringan internet. Internet of Things umumnya dikenal sebagai IoT, adalah sistem tertanam yang bertujuan untuk memperluas penggunaan konektivitas internet yang selalu aktif.

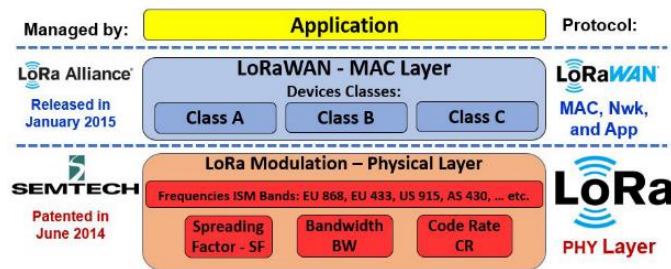


Gambar 3.1 Arsitektur Internet of Things

- B. Long Range (LoRa) merupakan sistem telekomunikasi jarak jauh yang memanfaatkan frekuensi radio (RF) dengan penggunaan daya rendah dan pengiriman data yang kecil. Pada saat ini, penggunaan teknologi LoRa mengirim dan menerima data secara luas dari *Gateway* ke *end-node* dengan jarak yang sangat jauh. LoRa termasuk kedalam teknologi *Low Power Wide Area Network* (LPWAN) yang memiliki kemampuan transmisi jarak jauh, efisiensi energi dan pengiriman data yang relatif kecil. LoRa beroperasi pada pita frekuensi *Industrial, Scientific, and Medical* (ISM) tidak berlisensi seperti pada frekuensi 2,4GHz, 868MHz, 915MHz, dan 433MHz tergantung pada regulasi masing-masing wilayah. Untuk di Indonesia, frekuensi yang digunakan adalah Indonesia 923 - 925 MHz AS923 LoRa memiliki daya jangkauan yang luas dengan konsumsi daya baterai rendah, sehingga LoRa sangat cocok untuk memantau

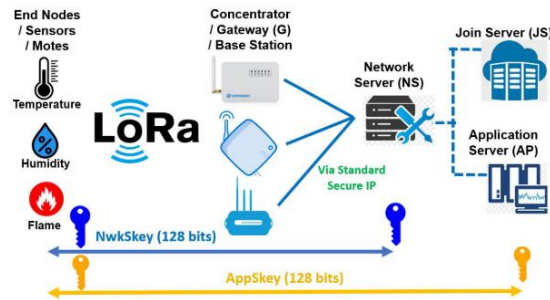
lingkungan lahan pertanian di Indonesia yang dikenal dengan negara agraris karena luasnya lahan pertanian yang dimiliki.

- C. LoRaWAN adalah teknologi *Low Power Wide Area Network* (LPWAN) yang menggunakan modulasi LoRa. LoRaWAN mencakup protokol komunikasi dan arsitektur sistem untuk jaringan sementara lapisan fisik LoRa, yang memungkinkan cakupan komunikasi jarak jauh. Protokol komunikasi LoRaWAN dapat digunakan untuk implementasi konsep IoT (*Internet of Things*)



Gambar 3.2 LoRa and LoRaWAN Technology Stack

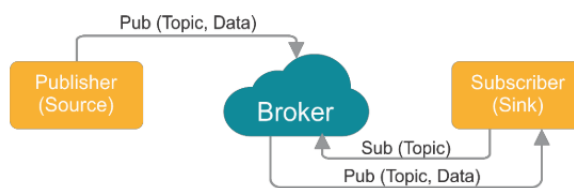
Modulasi LoRa memiliki *spreading factor* dari SF7 hingga SF12, di mana SF12 digunakan untuk komunikasi yang lebih jauh (dengan penguatan tinggi - amplitudo) karena memiliki waktu lebih lama di udara, yang dikenal sebagai *airtime*. Sebaliknya, SF7 memiliki penguatan rendah dengan laju data tinggi. Selain itu, LoRa menyajikan koneksi yang dapat diandalkan karena melakukan pengkodean kesalahan (deteksi dan koreksi) yang ditentukan oleh *coding rate*, yang dapat disesuaikan menjadi 4/5, 4/6, 4/7, dan 4/8. Terakhir, *bandwidth* dapat diatur menjadi 125 KHz, 250 KHz, dan 500 KHz. Lapisan-lapisan ini dapat disesuaikan untuk berbagai pengaturan seperti *spreading factor*, *coding rate*, *bandwidth*, frekuensi, panjang *preamble*, dan sebagainya.



Gambar 3.3 Arsitektur LoRaWAN

Arsitektur jaringan LoRaWAN dibangun menggunakan topologi *star*, di mana *Gateway* meneruskan pesan antara perangkat akhir dan *network server* yang memungkinkan perangkat bekerja dengan baterai dalam waktu yang lama. *Gateway* terhubung ke *server* jaringan melalui koneksi IP standar dan berfungsi sebagai jembatan, yang hanya mengkonversi paket RF menjadi paket IP dan sebaliknya.

- D. Protokol MQTT digunakan dalam *Internet of Things* (IoT) dengan model komunikasi *publish-subscribe*. Terdapat tiga komponen utama yaitu *publisher*, *subscriber*, dan *message broker*. *Publisher* mengirimkan data sensor, *subscriber* berlangganan data, dan *message broker* menyampaikan data antara keduanya. Dalam MQTT, pesan dikirim ke *broker* dengan topik yang relevan, lalu *broker* mengirimkannya ke *subscriber* yang berlangganan topik tersebut.



Gambar 3.4 Blok Diagram Protokol MQTT

1.2 ALAT DAN BAHAN

- A. *Hardware* atau perangkat keras yang digunakan untuk merancang perangkat fisik pada penelitian ini yaitu:
- Laptop adalah perangkat elektronik yang dapat menerima data dari berbagai sumber, memproses data tersebut, dan menghasilkan *output* berupa informasi. Pada penelitian ini laptop digunakan untuk konfigurasi *End Device*, *Gateway*, *Network Server* dan *Application*

Server. Penelitian ini menggunakan laptop seri MSI gf63 Thin dengan RAM 16 GB dan Prosesor Core i5 Gen 11.



Gambar 3.5 Laptop MSI gf63 Thin

- b. *Gateway* Dragino PG1301 Raspberry Pi, digunakan sebagai perangkat untuk meneruskan data yang dikirim dari *end device* lalu mengirimkan data ke *network server*.



Gambar 3.6 *Gateway* Dragino PG1301 Raspberry Pi

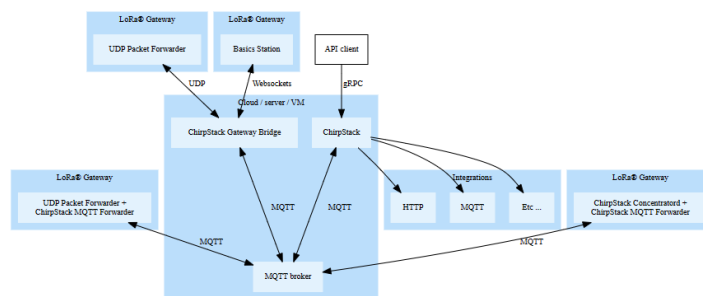
- c. *End Device* SenseCAP K1100, perangkat ini digunakan sebagai perangkat untuk mengukur kekuatan sinyal LoRa, Dimana terdapat sensor SHT40 dan sensor *Light* yang digunakan sebagai data yang akan dikirimkan ke *Gateway* melalui modul Grove LoRa E5.



Gambar 3.7 *End Device* SenseCAP K1100

- B. *Software* atau perangkat lunak yang digunakan dalam membantu perancangan pada penelitian ini yaitu:

- a. *Visual Studio Code* yang digunakan untuk pembuatan dan perancangan *codingan application server* dan *network server*.
- b. *Arduino* yang digunakan untuk membuat *codingan perangkat end device*.
- c. *ChirpStack* adalah *LoRaWAN Network Server* yang bersifat *opensource* dan dapat digunakan untuk mengatur jaringan *LoRaWAN* pribadi atau publik. *ChirpStack* menyediakan antarmuka *web* untuk pengelolaan *Gateway*, perangkat, dan penyewa serta untuk mengatur integrasi data dengan penyedia *cloud*, *database*, dan layanan yang umum digunakan untuk mengelola data perangkat. *ChirpStack* menyediakan API berbasis *gRPC* yang dapat digunakan untuk mengintegrasikan atau memperluas fungsionalitas *ChirpStack*.



Gambar 3.8 Chirpstack Architecture

- d. *Platform Openremote* merupakan *Application Server*, yang berfungsi untuk mengintegrasikan semua data, sensor, dan kontrol untuk mengaksesnya secara terpusat, mengubah data menjadi informasi dan mendukung desain aplikasi (*mobile*) yang disesuaikan memonitoring perangkat *end device* melalui tampilan *dashboard*.



Gambar 3.9 Platform Openremote

- e. *Docker* adalah layanan yang menyediakan kemampuan untuk mengemas dan menjalankan sebuah aplikasi dalam sebuah lingkungan terisolasi yang disebut dengan *container*. Dengan adanya isolasi dan

keamanan yang memadai memungkinkan untuk menjalankan banyak *container* di waktu yang bersamaan pada *host* tertentu [26].

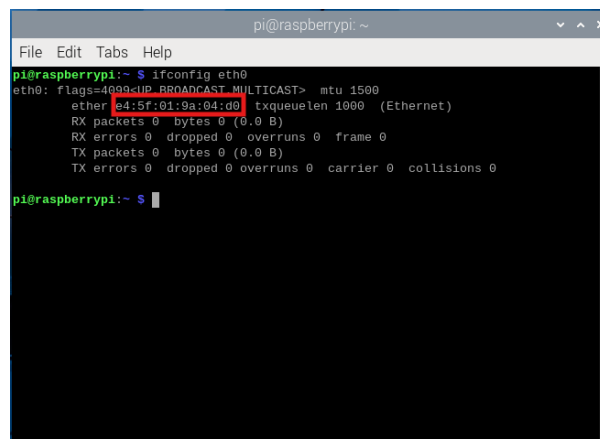


Gambar 3.10 Aplikasi Docker

1.3 INSTALASI DAN KONFIGURASI

A. Konfigurasi gateway Dragino PG1301 ke network server chirpstack

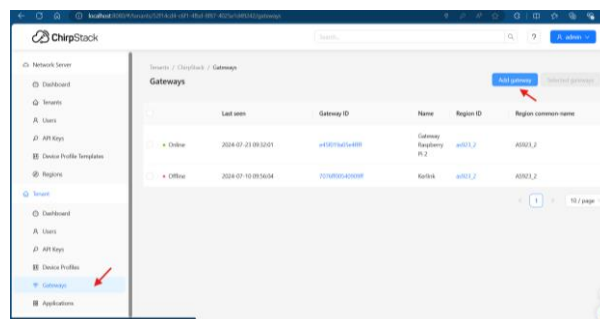
1. Pertama pastikan raspberry pi sudah siap digunakan dan sudah terinstall os rasbian dan modul LoRa sudah terpasang
2. Lalu buka terminal raspberry pi, ketik “ifconfig eth0” untuk melihat Gateway ID atau DevUI Gateway (E45F019A04D0)

A screenshot of a terminal window on a Raspberry Pi. The prompt is "pi@raspberrypi: ~". The command "ifconfig eth0" has been entered, and the output is displayed. The MAC address "e4:5f:01:9a:04:d0" is highlighted with a red box. The output shows: "eth0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500", "ether e4:5f:01:9a:04:d0 txqueuelen 1000 (Ethernet)", "RX packets 0 bytes 0 (0.0 B)", "RX errors 0 dropped 0 overruns 0 frame 0", "TX packets 0 bytes 0 (0.0 B)", and "TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0".

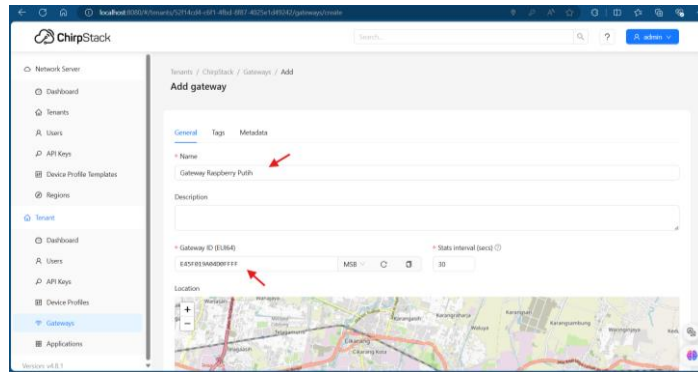
```
pi@raspberrypi:~$ ifconfig eth0
eth0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether e4:5f:01:9a:04:d0 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

pi@raspberrypi:~$
```

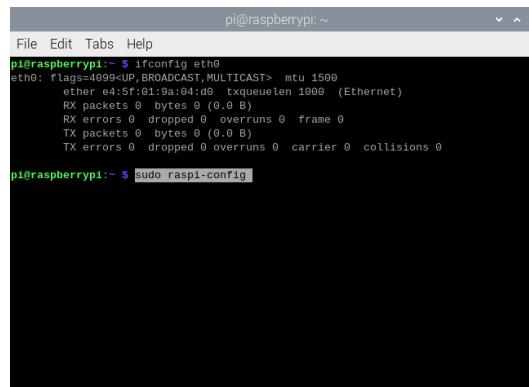
3. Buka halaman chirpstack, lalu pilih bagian gateway, dan klik add



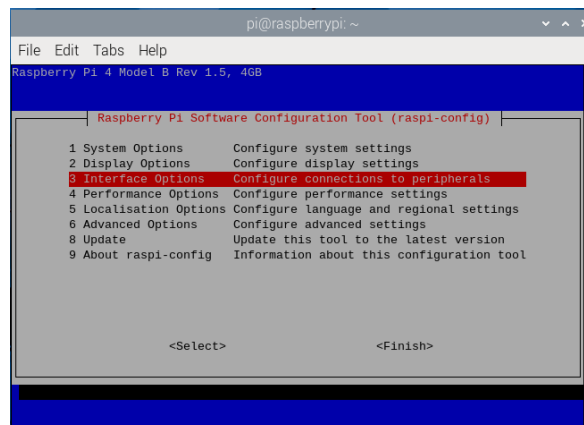
4. Isi nama, dan Gateway ID (E45F019A04D0FFFF) tambahkan FFFF pada akhir nomor gateway ID. Lalu klik submit



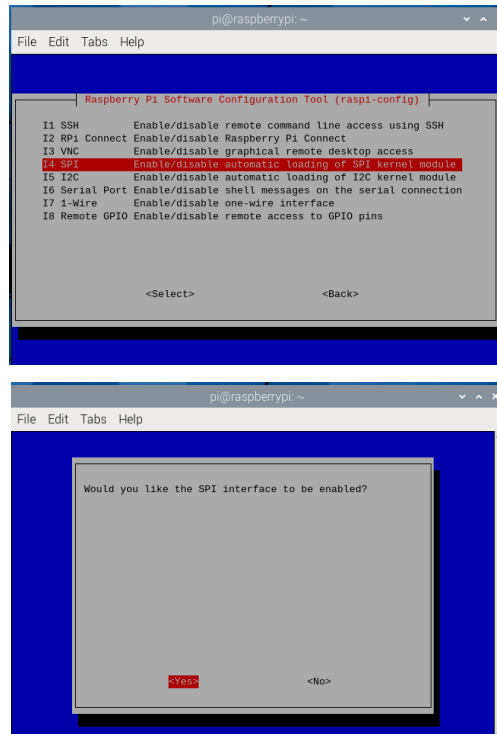
5. Balik ke raspberry pi, ketik “sudo raspi-config”



6. Pilih interface options, lalu enter



7. Lalu enable SPI, jika selesai maka klik finish



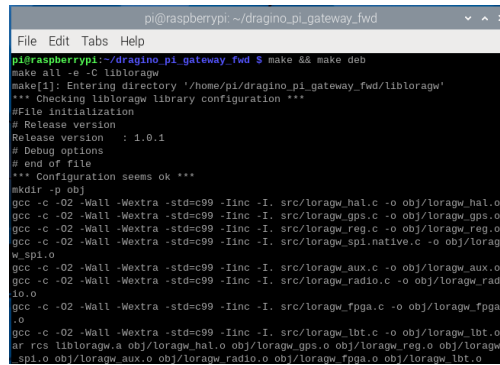
8. Download packet forwarder untuk LoRa, ketik “git clone https://github.com/fhessel/dragino_pi_gateway_fwd”

```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~$ git clone https://github.com/fhessel/dragino_pi_gateway_fwd.git  
Cloning into 'dragino_pi_gateway_fwd'...  
remote: Enumerating objects: 364, done.  
remote: Counting objects: 100% (48/48), done.  
remote: Compressing objects: 100% (9/9), done.  
remote: Total 364 (delta 36), reused 31 (delta 31), pack-reused 324  
Receiving objects: 100% (364/364), 344.50 KiB | 177.00 KiB/s, done.  
Resolving deltas: 100% (223/223), done.  
pi@raspberrypi:~$
```

9. Masuk ke directori folder yang diclone tadi

```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~$ git clone https://github.com/fhessel/dragino_pi_gateway_fwd.git  
Cloning into 'dragino_pi_gateway_fwd'...  
remote: Enumerating objects: 364, done.  
remote: Counting objects: 100% (48/48), done.  
remote: Compressing objects: 100% (9/9), done.  
remote: Total 364 (delta 36), reused 31 (delta 31), pack-reused 324  
Receiving objects: 100% (364/364), 344.50 KiB | 177.00 KiB/s, done.  
Resolving deltas: 100% (223/223), done.  
pi@raspberrypi:~$ cd dragino_pi_gateway_fwd/  
pi@raspberrypi:~/dragino_pi_gateway_fwd$
```

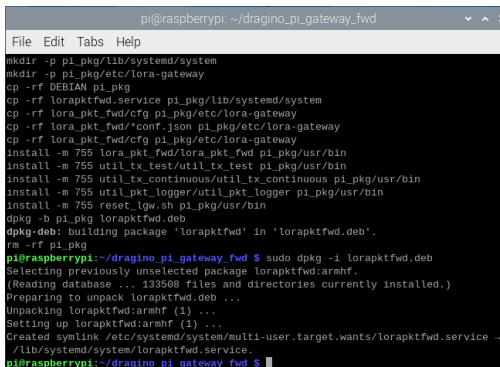
10. Lalu ketik “make && make deb”



```
pi@raspberrypi: ~/dragino_pi_gateway_fwd
File Edit Tabs Help

pi@raspberrypi:~/dragino_pi_gateway_fwd$ make && make deb
make all -e -C libloragw
make[1]: Entering directory '/home/pi/dragino_pi_gateway_fwd/libloragw'
*** Checking libloragw library configuration ***
#File initialization
# Release version
Release version : 1.0.1
# Debug options
# end of file
*** Configuration seems ok ***
mkdir -p obj
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_hal.c -o obj/loragw_hal.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_gps.c -o obj/loragw_gps.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_reg.c -o obj/loragw_reg.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_spi.native.c -o obj/loragw_spi.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_aux.c -o obj/loragw_aux.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_radio.c -o obj/loragw_radio.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_fpga.c -o obj/loragw_fpga.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_tbt.c -o obj/loragw_tbt.o
ar rcs libloragw.a obj/loragw_hal.o obj/loragw_gps.o obj/loragw_reg.o obj/loragw_spi.o obj/loragw_aux.o obj/loragw_radio.o obj/loragw_fpga.o obj/loragw_tbt.o
```

11. Lalu ketik “sudo dpkg -i lorapktfwd.deb”

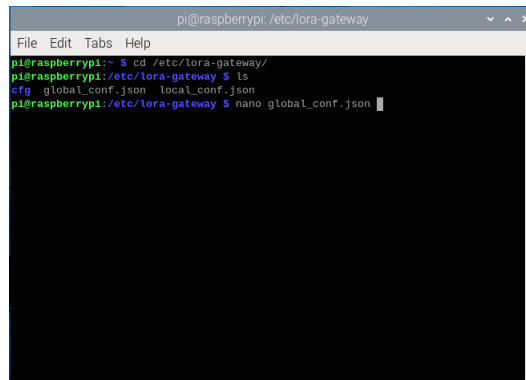


```
pi@raspberrypi: ~/dragino_pi_gateway_fwd
File Edit Tabs Help

mkdir -p pi_pkg/lib/systemd/system
mkdir -p pi_pkg/etc/lora-gateway
cp -rf DEBIAN pi_pkg
cp -rf lorapktfwd.service pi_pkg/lib/systemd/system
cp -rf lora_pkt_fwd/cfg pi_pkg/etc/lora-gateway
cp -rf lora_pkt_fwd/conf.json pi_pkg/etc/lora-gateway
cp -rf lora_pkt_fwd/cfg pi_pkg/etc/lora-gateway
install -m 755 lora_pkt_fwd/lora_pkt_fwd pi_pkg/usr/bin
install -m 755 util_tx_test/util_tx_test pi_pkg/usr/bin
install -m 755 util_tx_continuous/util_tx_continuous pi_pkg/usr/bin
install -m 755 util_pkt_logger/util_pkt_logger pi_pkg/usr/bin
install -m 755 reset_lgw.sh pi_pkg/usr/bin
dpkg -b pi_pkg lorapktfwd.deb
dpkg-deb: building package 'lorapktfwd' in 'lorapktfwd.deb'.
rm -rf pi_pkg

pi@raspberrypi:~/dragino_pi_gateway_fwd$ sudo dpkg -i lorapktfwd.deb
Selecting previously unselected package lorapktfwd:armhf.
(Reading database ... 133508 files and directories currently installed.)
Preparing to unpack lorapktfwd.deb ...
Unpacking lorapktfwd:armhf (1) ...
Setting up lorapktfwd:armhf (1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/lorapktfwd.service → /lib/systemd/system/lorapktfwd.service.
pi@raspberrypi:~/dragino_pi_gateway_fwd$
```

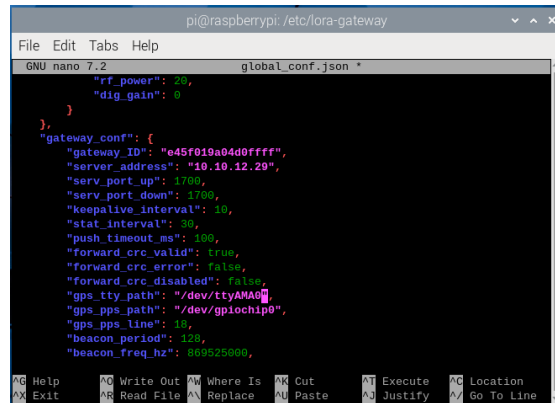
12. Masuk ke direktori /etc/lora-gateway/, lalu edit file global_conf.json dengan cara ketik “nano global_conf.json”



```
pi@raspberrypi: /etc/lora-gateway
File Edit Tabs Help

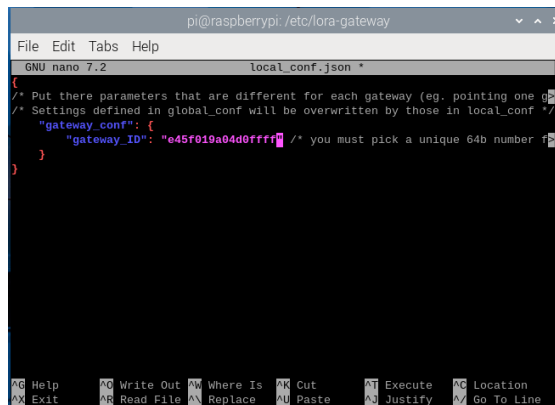
pi@raspberrypi:~$ cd /etc/lora-gateway/
pi@raspberrypi:/etc/lora-gateway$ ls
cfg global_conf.json local_conf.json
pi@raspberrypi:/etc/lora-gateway$ nano global_conf.json
```

13. Ubah gateway_ID dengan yang sudah didaftarkan pada chirpstack, lalu ganti ip address menjadi ip laptop (buka cmd, ketik ipconfig). Ubah juga gps tty path menjadi ttyAMA0. Save settingan dengan cara ketik CTRL+X, lalu pilih Y, dan enter



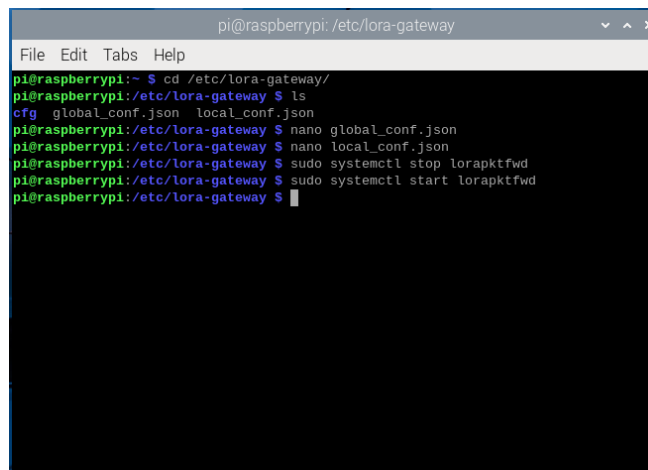
```
pi@raspberrypi: /etc/lora-gateway
File Edit Tabs Help
GNU nano 7.2 global_conf.json *
{
  "rff_power": 20,
  "dig_gain": 0
},
"gateway_conf": {
  "gateway_ID": "e45f019a04d0ffff",
  "server_address": "10.10.12.29",
  "serv_port_up": 1700,
  "serv_port_down": 1700,
  "keepalive_interval": 10,
  "stat_interval": 30,
  "push_timeout_ms": 100,
  "forward_crc_valid": true,
  "forward_crc_error": false,
  "forward_crc_disabled": false,
  "gps_tty_path": "/dev/ttyAMA0",
  "gps_pps_path": "/dev/gpiochip0",
  "gps_pps_line": 14,
  "beacon_period": 120,
  "beacon_freq_hz": 869525000,
}
```

14. Edit juga untuk file local_conf.json, caranya sama seperti global



```
pi@raspberrypi: /etc/lora-gateway
File Edit Tabs Help
GNU nano 7.2 local_conf.json *
/* Put there parameters that are different for each gateway (eg. pointing one gp
/* Settings defined in global_conf will be overwritten by those in local_conf */
"gateway_conf": {
  "gateway_ID": "e45f019a04d0ffff" /* you must pick a unique 64b number f
}
```

15. Jalankan perintah untuk stop dan restart lorapktfwd



```
pi@raspberrypi: /etc/lora-gateway
File Edit Tabs Help
pi@raspberrypi:~$ cd /etc/lora-gateway/
pi@raspberrypi:/etc/lora-gateway$ ls
cfg global_conf.json local_conf.json
pi@raspberrypi:/etc/lora-gateway$ nano global_conf.json
pi@raspberrypi:/etc/lora-gateway$ nano local_conf.json
pi@raspberrypi:/etc/lora-gateway$ sudo systemctl stop lorapktfwd
pi@raspberrypi:/etc/lora-gateway$ sudo systemctl start lorapktfwd
pi@raspberrypi:/etc/lora-gateway$
```

16. Jalankan perintah “sudo journalctl -u lorapktfwd -f” untuk melihat log apakah sudah berhasil terhubung ke chirpstack atau belum

```

pi@raspberrypi: /etc/lora-gateway
File Edit Tabs Help
pi@raspberrypi:~$ cd /etc/lora-gateway/
pi@raspberrypi:/etc/lora-gateway$ ls
cfg global_conf.json local_conf.json
pi@raspberrypi:/etc/lora-gateway$ nano global_conf.json
pi@raspberrypi:/etc/lora-gateway$ nano local_conf.json
pi@raspberrypi:/etc/lora-gateway$ sudo systemctl stop lorapktfwd
pi@raspberrypi:/etc/lora-gateway$ sudo systemctl start lorapktfwd
pi@raspberrypi:/etc/lora-gateway$ sudo journalctl -u lorapktfwd -f
Jul 23 10:25:13 raspberrypi reset_lgw.sh[3580]: /usr/bin/reset_lgw.sh: 26: echo:
echo: I/O error
Jul 23 10:25:14 raspberrypi reset_lgw.sh[3580]: /usr/bin/reset_lgw.sh: 29: canno
t create /sys/class/gpio/gpio23/direction: Directory nonexistent
Jul 23 10:25:15 raspberrypi reset_lgw.sh[3580]: /usr/bin/reset_lgw.sh: 32: canno
t create /sys/class/gpio/gpio23/value: Directory nonexistent
Jul 23 10:25:16 raspberrypi reset_lgw.sh[3580]: /usr/bin/reset_lgw.sh: 33: canno
t create /sys/class/gpio/gpio23/value: Directory nonexistent
Jul 23 10:25:17 raspberrypi reset_lgw.sh[3580]: /usr/bin/reset_lgw.sh: 36: canno
t create /sys/class/gpio/gpio23/direction: Directory nonexistent
Jul 23 10:25:19 raspberrypi systemd[1]: Started lorapktfwd.service - packet forw
arder.
Jul 23 10:25:19 raspberrypi lora_pkt_fwd[3586]: Starting Packet Forwarder at 202
40723032519
Jul 23 10:25:19 raspberrypi lora_pkt_fwd[3586]: src/lora_pkt_fwd.c:1290:main():
ERROR~ [main] failed to start the concentrator

```

17. Untuk melihat status gateway sudah terhubung ke chirpstack atau belum, maka bisa mengetikan “sudo systemctl status lorapktfwd”

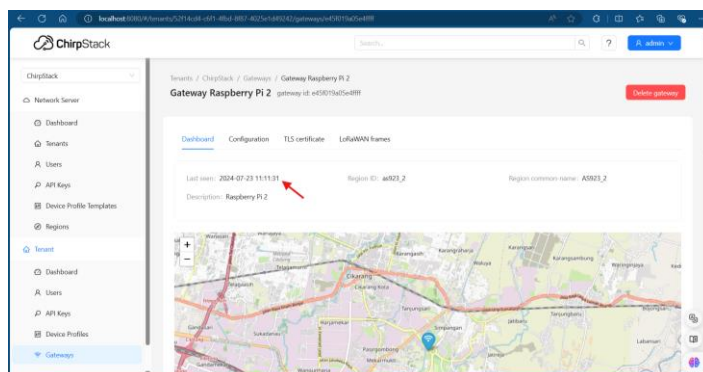
```

File Edit Tabs Help
pi@raspberrypi:~$ sudo systemctl status lorapktfwd
• lorapktfwd.service - packet forwarder
   Loaded: loaded (/lib/systemd/system/lorapktfwd.service; enabled; vendor pr
   Active: active (running) since Tue 2024-07-02 16:55:15 +07; 2 weeks 6 days>
   Process: 1772 ExecStartPre=/usr/bin/reset_lgw.sh start (code=exited, status
   Main PID: 1841 (lora_pkt_fwd)
   Tasks: 8 (limit: 3933)
   CPU: 3h 55min 38.383s
   CGroup: /system.slice/lorapktfwd.service
           └─1841 /usr/bin/lora_pkt_fwd

Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ ### [UPSTREAM] ###
Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ # RF packets received b
Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ # CRC OK: 0.00%, CRC FA
Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ # RF packets forwarded:
Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ # PUSH_DATA datagrams s
Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ # PUSH_DATA acknowledge
Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ ### [DOWNSTREAM] ###
Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ # PULL_DATA sent: 3 (16
Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ # PULL_RESP(onse) datag
Jul 23 11:09:32 raspberrypi lora_pkt_fwd[1841]: REPORT~ # RF packets sent to co
lines 1-20/20 (END)

```

18. Lalu pada chirpstack akan terlihat waktu terakhir status gateway terhubung

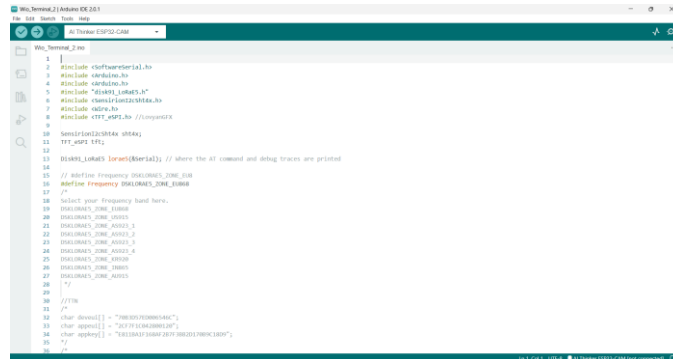


B. Instalasi Konfigurasi Device SenseCAP K1100 ke network server chirpstack.

1. Download file source code Wio Terminal terlebih dahulu

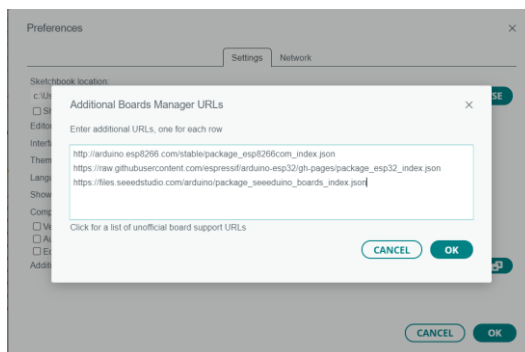
<https://drive.google.com/file/d/1sAx2gpNLOy6rqRAgnxV1FqxsKvHoE3aE/view?usp=sharing>

2. Buka file tersebut menggunakan Arduino IDE

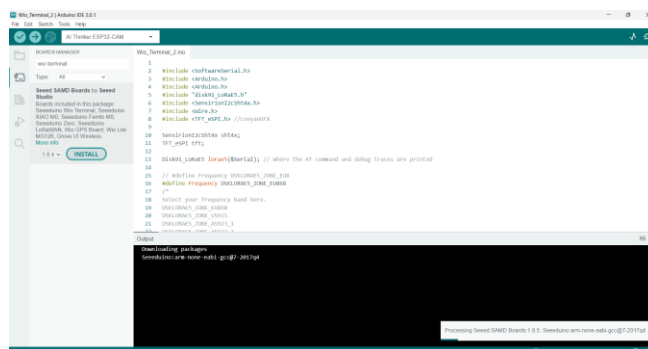


3. Tambahkan board wio terminal terlebih dahulu pada file > reference di Arduino Ide

“https://files.seeedstudio.com/arduino/package_seeedstudio_boards_index.json”



4. Lalu melakukan Instalasi board wio terminal, pada Tools > Board > Board Manager, ketik Wio Terminal, klik install



5. Download library disk91_LoRaE5 pada link berikut

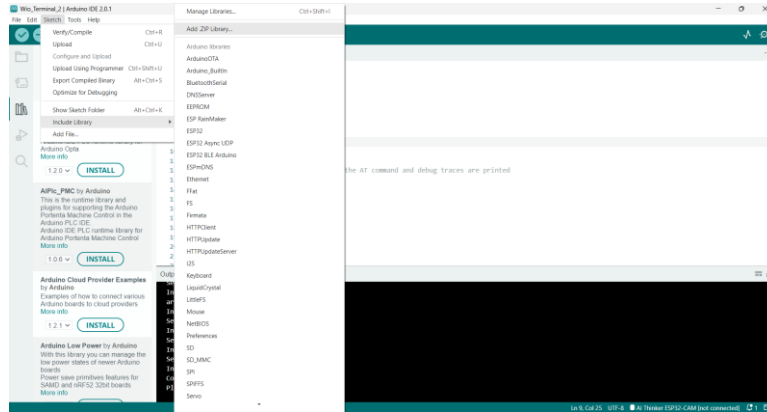
https://github.com/disk91/Disk91_LoRaE5

6. Download library SHT40 dan Arduino core pada link berikut

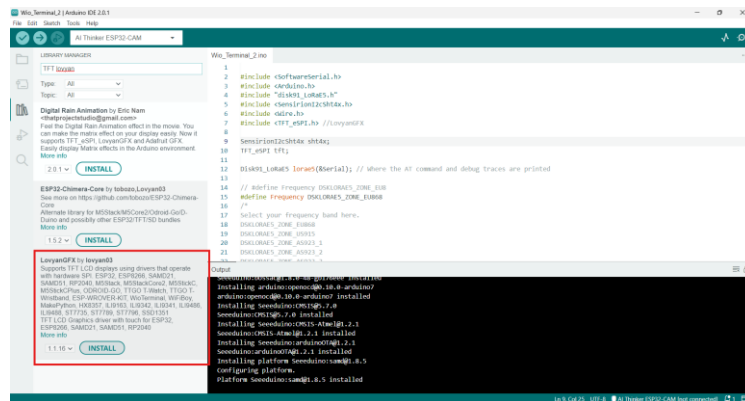
<https://github.com/Sensirion/arduino-i2c-sht4x>

<https://github.com/Sensirion/arduino-core>

7. Add zip library yang sudah didownload

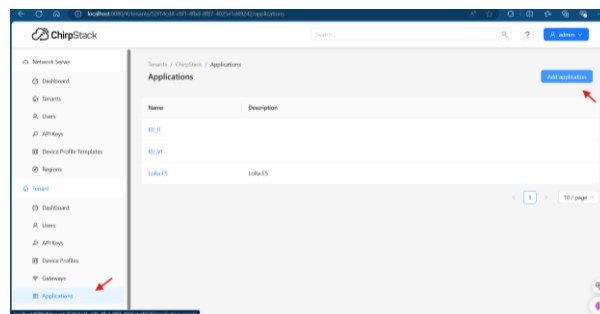


8. Install library TFT by LovyanGFX

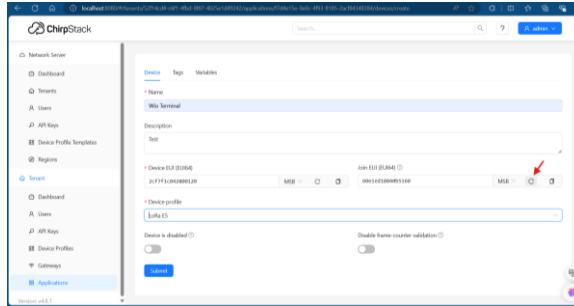


9. Daftarkan module LoRa E5 pada chirpstack, pertama lihat Device EUI yang terdapat pada module LoRa E5 (belakang perangkat)

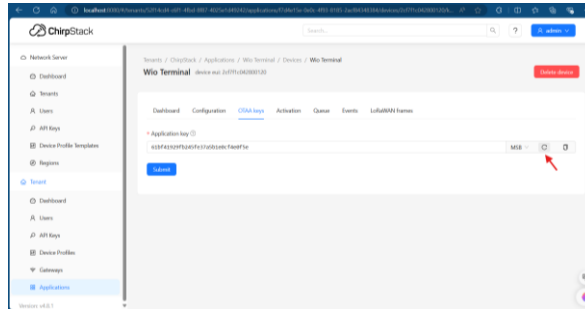
10. Buka chirpstack, masuk ke application, klik add application, isi nama dan deskripsi



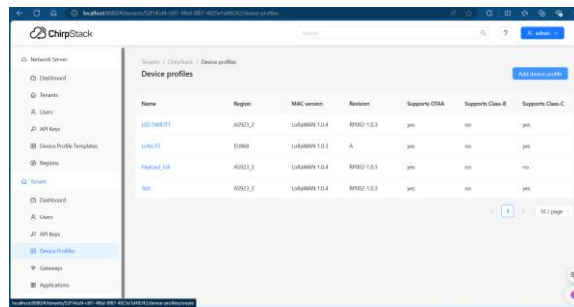
11. Klik add device, masukan nama, deskripsi, Device EUI, generate Join EUI dan pilih device profile, klik submit



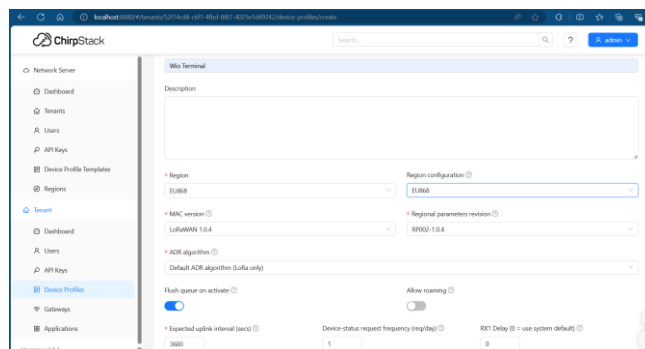
12. Generate OTTA keys / Application Keys, klik submit



13. Jika device profile belum buat, maka buat terlebih dahulu, masuk ke device profile, klik add

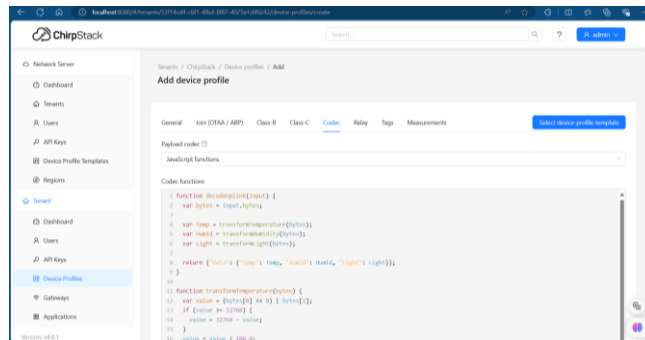


14. Isikan nama, pilih region, region configuration, pilih MAC version dan regional parameter version, klik submit



15. Tetap di device profile, masuk ke tab codec, pilih Javascript function, ganti codec function dengan codec pada link berikut, lalu klik submit

https://drive.google.com/file/d/10fpbOfO8xPMUGWKng0sPNB_xIq7BaaNg/view?usp=sharing



16. Selanjutnya pada Arduino Ide, Sesuaikan frekuensi regional yang akan dipakai pada LoRa E5

```
// #define Frequency DSKLORAE5_ZONE_EU8
#define Frequency DSKLORAE5_ZONE_EU868
/*
Select your frequency band here.
DSKLORAE5_ZONE_EU868
DSKLORAE5_ZONE_US915
DSKLORAE5_ZONE_AS923_1
DSKLORAE5_ZONE_AS923_2
DSKLORAE5_ZONE_AS923_3
DSKLORAE5_ZONE_AS923_4
DSKLORAE5_ZONE_KR920
DSKLORAE5_ZONE_IN865
DSKLORAE5_ZONE_AU915
*/
```

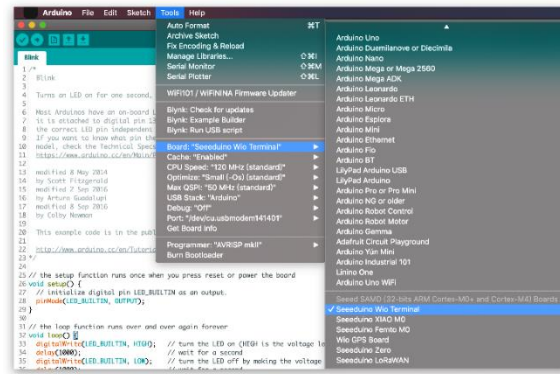
17. Sesuaikan DevEUI, AppEUI dan AppKey pada application di chirpstack yang sudah dibuat sebelumnya, untuk AppEUI adalah Join EUI pada Application.

```
//Chirpstack Laptop Lenovo
char deveui[] = "2cf7f1c042800120";
char appeui[] = "3c31be8492d01e72";
char appkey[] = "0c21c22f459cc111aa12248e4f33ee86";
```

18. Lalu untuk mengatur Spreading Factor, port dan power dapat diganti pada bagian code dibawah, panah warna merah menunjukan port (8), kuning menunjukan spreading factor (7) dan biru menunjukan power (14)

```
uint8_t rxBuff[16];
uint8_t rxSize = 16;
uint8_t rxPort;
if (lorae5.sendReceive_sync(8,data,sizeof(data),rxBuff,&rxSize,&rxPort,7,14,0) )
```

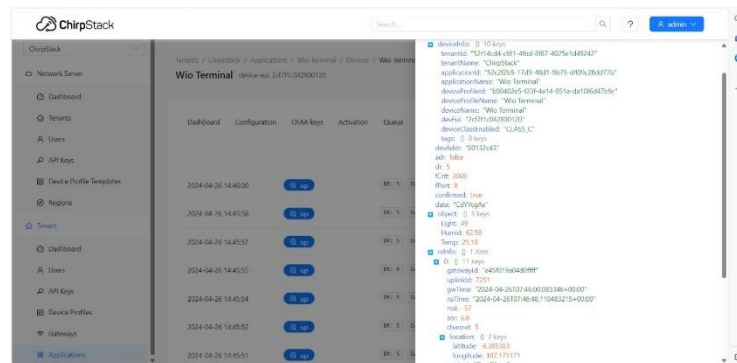
19. Jika sudah selesai semua, maka bisa upload program, pertama sesuaikan dulu pemilihan board dan portnya. lalu upload



20. Jika berhasil semua maka pada layar TFT alat akan menampilkan nilai dari sensor SHT40 dan nilai sinyal RSSI dan SNR dari komunikasi LoRa dan status.

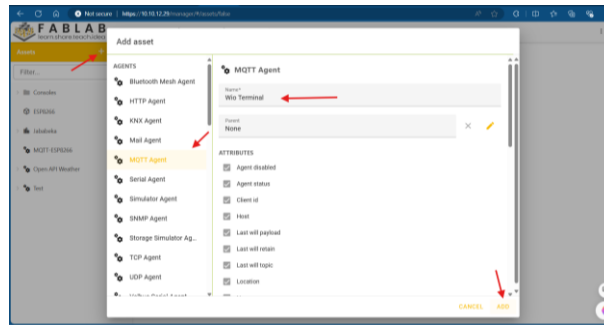


21. Sedangkan pada application chirpstack akan menampilkan data json pada event.

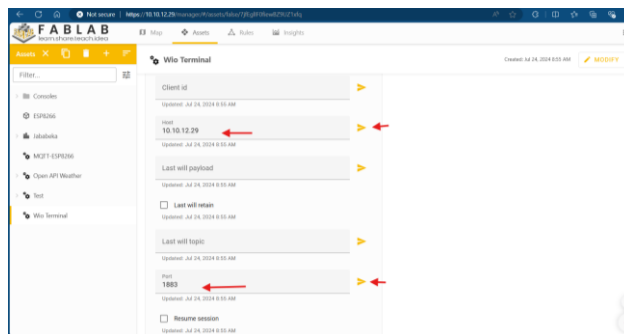


C. Konfigurasi kirim data dari Chirpstack ke Openremote

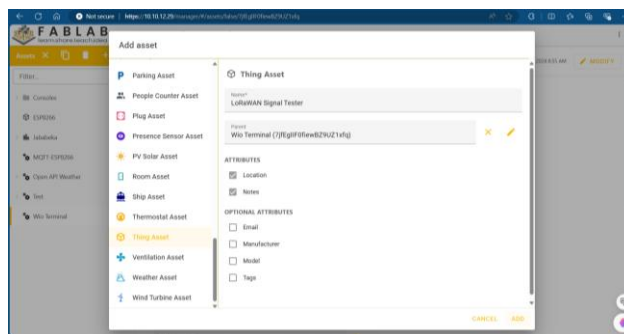
1. Pertama masuk ke openremote, lalu ke menu Assets, pilih add assets, pilih MQTT Agent, masukan nama dan klik add.



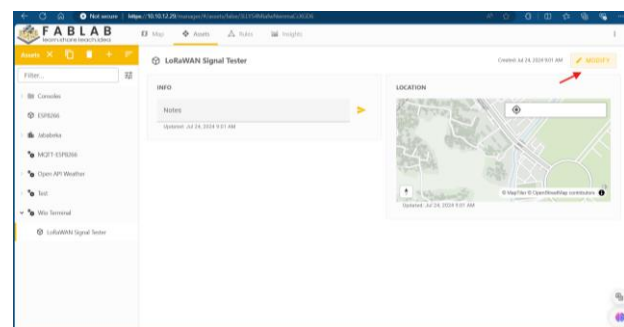
2. Masukkan host (ipconfig) dan port mqtt lalu klik button disamping



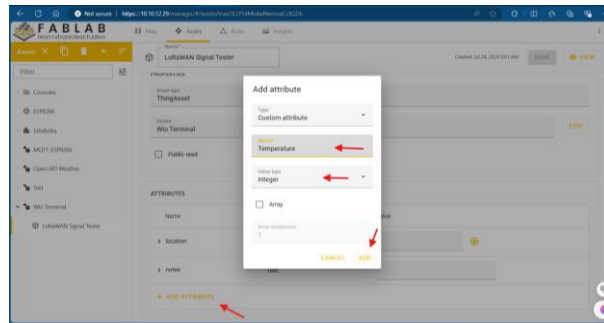
3. Lalu add assets lagi, pilih Things Assets, ubah nama klik add



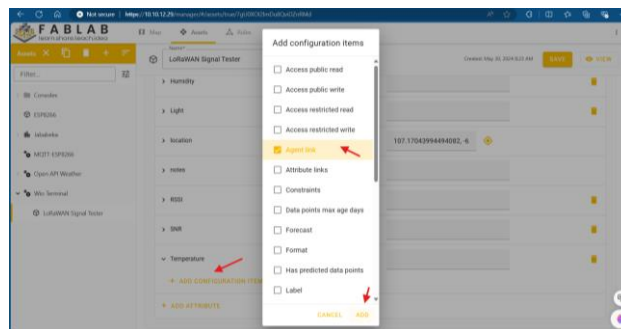
4. Klik things assets yang sudah dibuat tadi, lalu klik modify



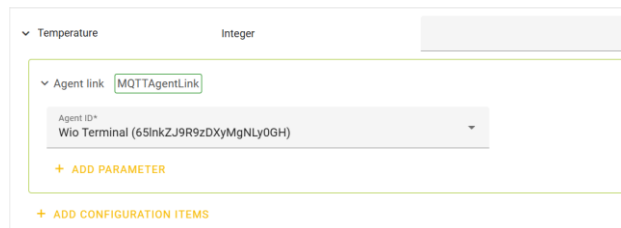
5. Tambahkan attribute, pilih custome attribute, berikan nama, pilih value type Integer, klik add



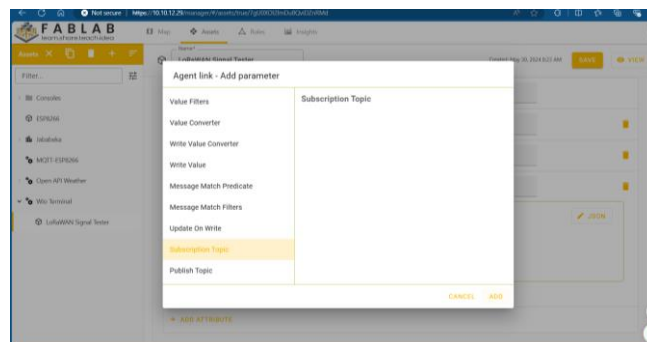
6. Add configuration items, pilih Agent Link, klik add



7. Pilih agent link, yang sudah dibuat sebelumnya (MQTT Agent)

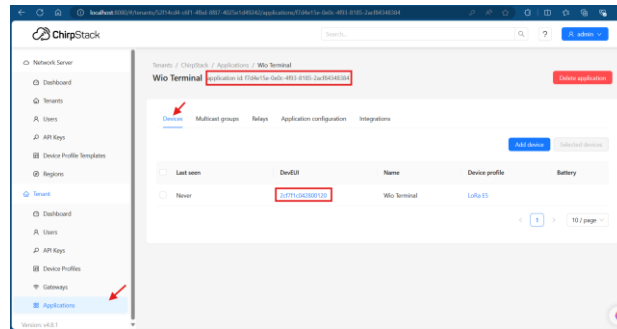


8. Add parameter, pilih subscription topic, klik add



9. Isi subscription topic sesuai format pada chirpstack

“application/APPLICATION_ID/device/DEV_EUI/event/up”,
sebagai contoh disini “application/f7d4e15e-0e0c-4f93-8185-
2acf84348384/device/2cf7f1c042800120/event/up”



Temperature Integer

Agent link MQTTAgentLink

Agent ID* Wio Terminal (65lnkZJ9R9zDxyMgNly0GH)

Subscription Topic application/f7d4e15e-0e0c-4f93-8185-2acf84348384/device/2cf7f1c042800120/event/up

+ ADD PARAMETER

+ ADD CONFIGURATION ITEMS

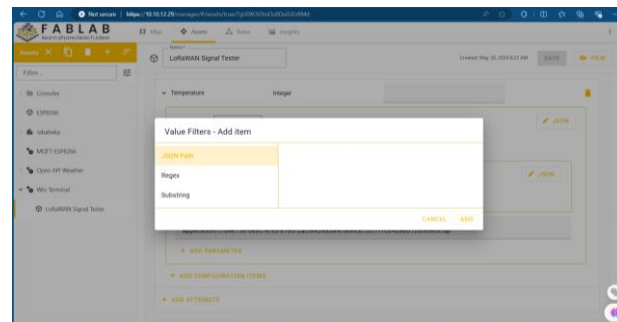
10. Add parameter lagi, lalu pilih Value Filters

Agent link - Add parameter

Value Filters	Value Filters
Value Converter	Defines ValueFilters to apply to an incoming value before it is written to a protocol linked attribute; this is particularly useful for generic protocols. The message should pass through the filters in array order
Write Value Converter	
Write Value	
Message Match Predicate	
Message Match Filters	
Update On Write	
Publish Topic	

CANCEL ADD

11. Pada value filters, add item, pilih JSON Path



12. Isi JSON Path dengan "\$.object.Temp" sesuai dengan format json chirpstack

Temperature Integer

Agent link MQTTAgentLink

Agent ID* Wio Terminal (65lnkZJ9R9zDxyMgNly0GH)

Value Filters

JSON Path JSON Path

Path \$.object.Temp

Return First ☒

Return Last ☐

+ ADD ITEM

Subscription Topic application/f7d4e15e-0e0c-4f93-8185-2acf84348384/device/2cf7f1c042800120/event/up

13. Tambahkan juga untuk attribute yang lain seperti, humidity, RSSI dan SNR

The image shows three screenshots of the ChirpStack configuration interface, specifically the 'Value Filters' section for different attributes. Each screenshot shows the configuration for a specific attribute: Humidity, RSSI, and SNR. The configuration includes the Agent link (MQTTAgentLink), the Agent ID (Wio Terminal), the JSON Path, and the Path (e.g., \$object.Humid, \$rxinfo[0].rssi, \$rxinfo[0].snr). The 'Return First' checkbox is checked for all three attributes.

Humidity Configuration:

- Attribute: Humidity (Integer)
- Agent link: MQTTAgentLink
- Agent ID: Wio Terminal (65lnkZJ9R9zDxyMghLy0Gh)
- JSON Path: \$object.Humid
- Path: \$object.Humid
- Return First: ☒
- Return Last: ☐

RSSI Configuration:

- Attribute: RSSI (Negative integer)
- Agent link: MQTTAgentLink
- Agent ID: Wio Terminal (65lnkZJ9R9zDxyMghLy0Gh)
- JSON Path: \$rxinfo[0].rssi
- Path: \$rxinfo[0].rssi
- Return First: ☒
- Return Last: ☐

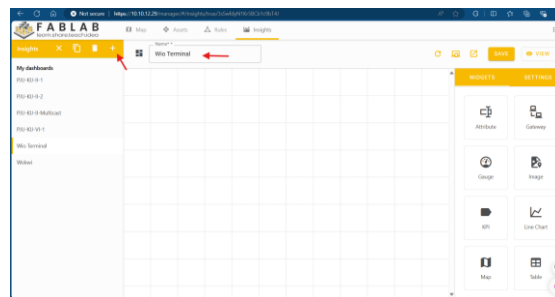
SNR Configuration:

- Attribute: SNR (Integer)
- Agent link: MQTTAgentLink
- Agent ID: Wio Terminal (65lnkZJ9R9zDxyMghLy0Gh)
- JSON Path: \$rxinfo[0].snr
- Path: \$rxinfo[0].snr
- Return First: ☐
- Return Last: ☐

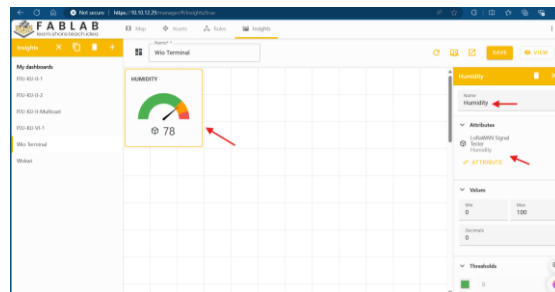
14. Jika sudah semua dan berhasil terhubung, maka nantinya pada openremote akan menampilkan nilai dari alat ukur sinyal LoRa yang sebelumnya dikirim ke chirpstack.

ATTRIBUTES	
Humidity	67.17
Updated: Mar 27, 2024 12:44 PM	
RSSI	-41
Updated: Mar 27, 2024 12:44 PM	
SNR	8.5
Updated: Mar 27, 2024 12:44 PM	
Temperature	24.89
Updated: Mar 27, 2024 12:44 PM	

15. Selanjutnya membuat dashboard insight, pilih menu insights, klik add



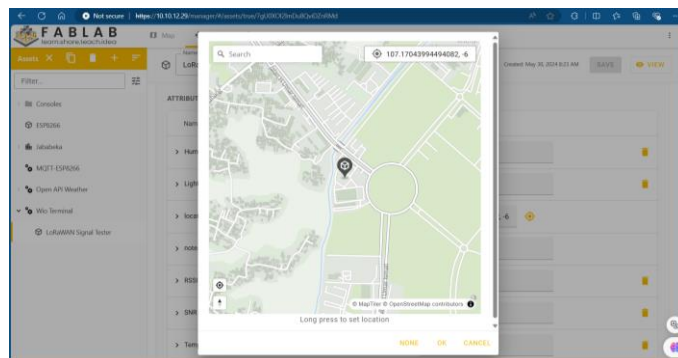
16. Pilih widget gauge, drag and drop di workspace, ubah nama, dan pilih salah satu attribute yang sebelumnya sudah dibuat pada MQTT Agent, dan Thing Assets.



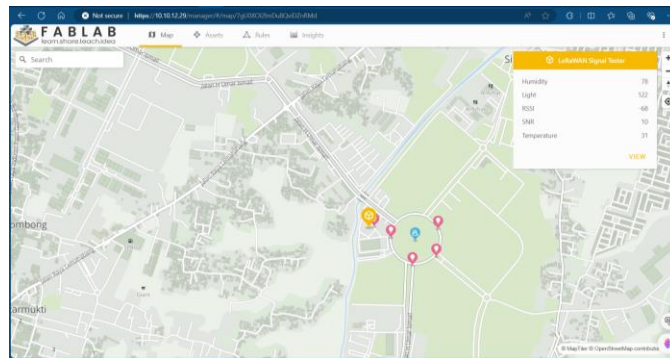
17. Buat juga untuk attribute yang lain, jika sudah maka hasilnya akan seperti ini



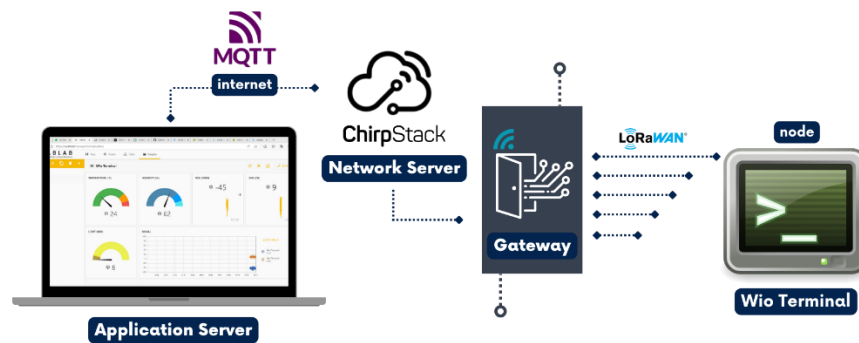
18. Lalu pada attribute location di menu assets, masukan koordinat atau pilih lokasi dari Thing Assets



19. Jika sudah, maka pada menu map akan muncul Thing Assets yang sudah dibuat tadi



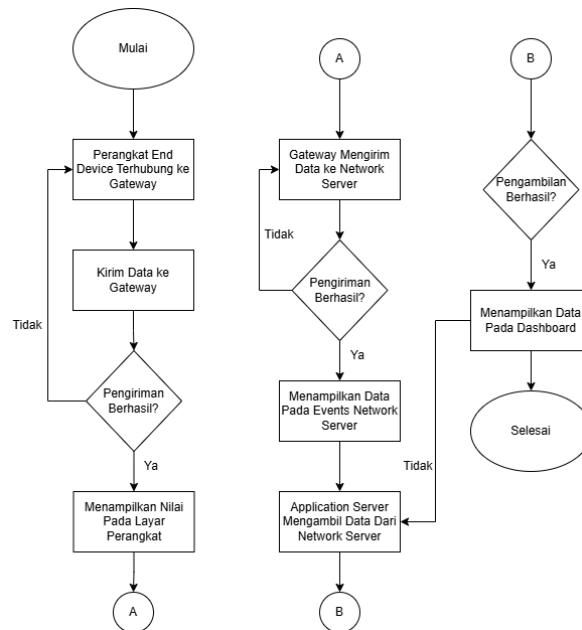
1.4 ARSITEKTUR DAN CARA KERJA



Gambar 3.11 Arsitektur Alat Ukur Sinyal LoRa

Pada Gambar 3.11, arsitektur alat ukur sinyal LoRa telah memenuhi kriteria arsitektur LoRaWAN. *Application Server* berfungsi untuk memantau perangkat sinyal LoRa melalui *dashboard Openremote*. *Network Server* terhubung dengan *Application Server Openremote* melalui jaringan internet dengan menggunakan protokol komunikasi MQTT. *Network Server* juga terhubung dengan *Gateway* melalui koneksi internet secara nirkabel atau melalui LAN *ethernet*. Fungsi *Network Server* adalah untuk mengolah dan meneruskan data yang dikirimkan dari *Gateway (Uplink)* maupun data dari *Application Server (Downlink)*. Perangkat *Gateway* berfungsi untuk meneruskan data dari *Network Server* ke perangkat alat ukur sinyal LoRa, dan sebaliknya. Namun, pengiriman data dari *Network Server* ke *Gateway* dilakukan melalui internet menggunakan komunikasi MQTT, sedangkan

pengiriman data dari perangkat alat ukur sinyal LoRa dilakukan melalui komunikasi LoRaWAN.



Gambar 3.12 *Flowchart* Alat Ukur Sinyal LoRa

Cara kerja dari alat ukur sinyal LoRa yaitu pertama, Ketika perangkat dinyalakan modul LoRa akan mencoba untuk Join server terlebih dahulu atau mencoba terhubung dengan Gateway dengan frekuensi yang sama, setelah terhubung perangkat akan mengirimkan data dari sensor-sensor yang terhubung seperti (Suhu, Kelembaban dan Cahaya), jika pengiriman berhasil maka pada layar perangkat akan menampilkan nilai dari data-data sensor yang dikirimkan sebelumnya. Selain itu perangkat juga menampilkan nilai RSSI (kekuatan sinyal), SNR (sinyal terhadap noise) dan koneksi yang didapat dari proses pengiriman data-data sensor, data sinyal tersebut dapat digunakan sebagai acuan untuk mengetahui luas jangkauan dari Gateway. Jadi dengan melihat data sinyal dan status koneksi maka pengguna dapat mengetahui apakah perangkat end device ini dapat terhubung pada Gateway jika dilakukan pengukuran disuatu wilayah yang ingin diukur.

Kedua Gateway akan mengirimkan data-data sensor dan data sinyal ke Network Server Chirpstack dan akan menampilkannya pada Log Events Network Server Chirpstack. Ketiga Application Server dapat mengambil data

dari Network server menggunakan subscription topic pada komunikasi MQTT dan ditampilkan pada dashboard.