ขั้นตอนการติดตั้ง

1. ขั้นตอนการติดตั้งเพื่อใช้งานระบบทั้งหมด

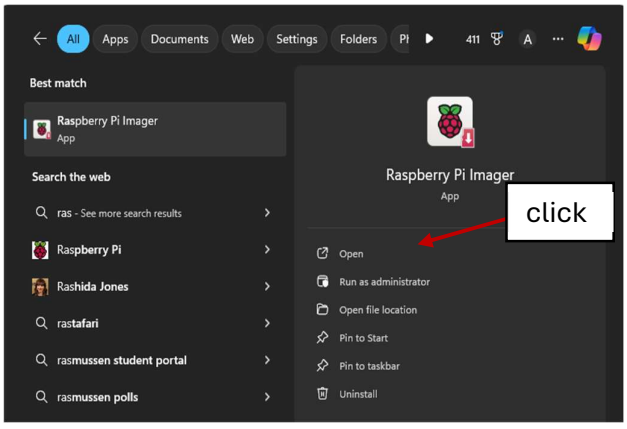
a) ติดตั้ง Raspberry Pi OS

i) อุปกรณ์ที่ต้องใช้

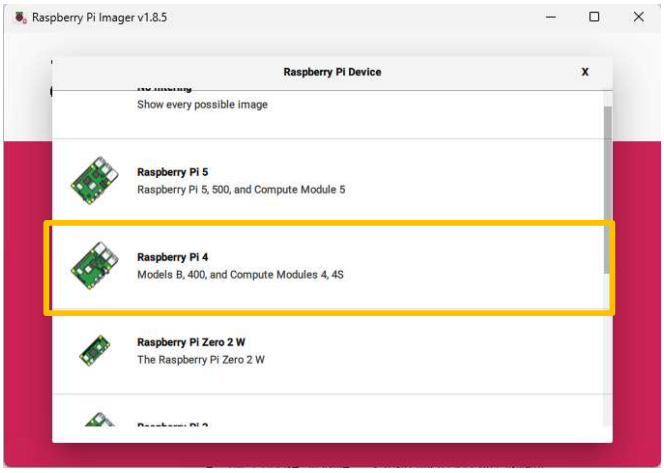
1) Raspberry Pi 4 Model B
2) MicroSD Card (แนะนำขนาด 16GB หรือ 32GB, Class 10 ขึ้นไป)
3) เครื่องคอมพิวเตอร์ (Windows, macOS หรือ Linux)
4) อะแดปเตอร์จ่ายไฟ (USB-C 5V/3A สำหรับ Raspberry Pi 4)
5) สาย HDMI และจอภาพ (ถ้าต้องการต่อจอแสดงผล)
6) คีย์บอร์ดและเมาส์ (ถ้าต้องการตั้งค่าผ่าน GUI)

ii) ขั้นตอนที่ 1: ดาวน์โหลดและติดตั้ง Raspberry Pi OS ลงบน MicroSD Card

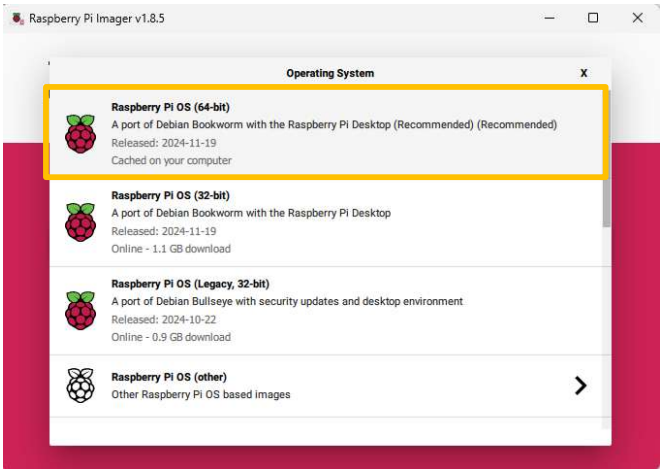
1) ดาวน์โหลด Raspberry Pi Imager
   - ไปที่เว็บไซต์ https://www.raspberrypi.com/software/
   - ดาวน์โหลดและติดตั้ง Raspberry Pi Imager สำหรับระบบปฏิบัติการของคุณ (Windows/macOS/Linux)
2) ใส่ MicroSD Card เข้าไปในเครื่องคอมพิวเตอร์
   - ใช้ Card Reader หากคอมพิวเตอร์ไม่มีช่องใส่ MicroSD
3) เลือก OS และติดตั้งลง MicroSD Card
   - เปิดโปรแกรม Raspberry Pi Imager ที่ช่องค้นหา(Search)



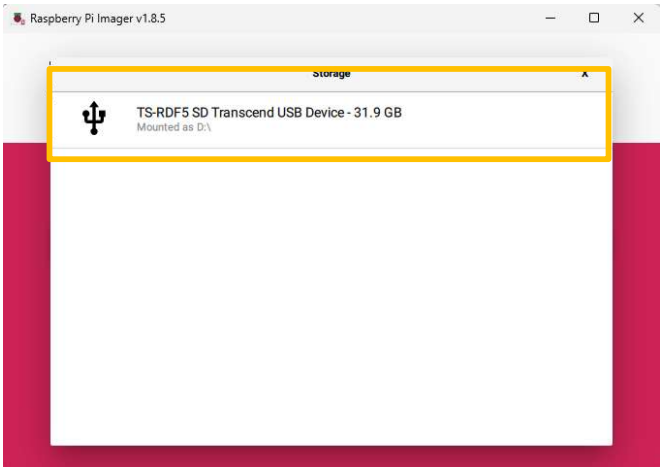
- กด "Choose Device" เลือก Raspberry Pi Module ที่ใช้



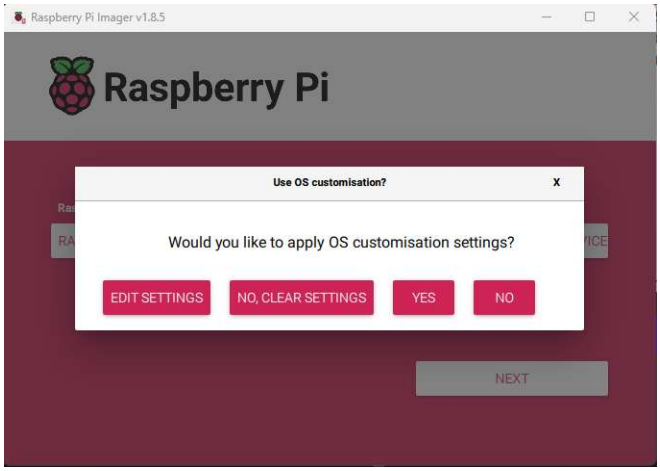
- กด "Choose OS" เลือก "Raspberry Pi OS (64-bit)" หรือ "Raspberry Pi OS (32-bit)" ตามที่ต้องการ



- กด "Choose Storage" ⟶ เลือก MicroSD Card ของคุณ แล้วกด NEXT

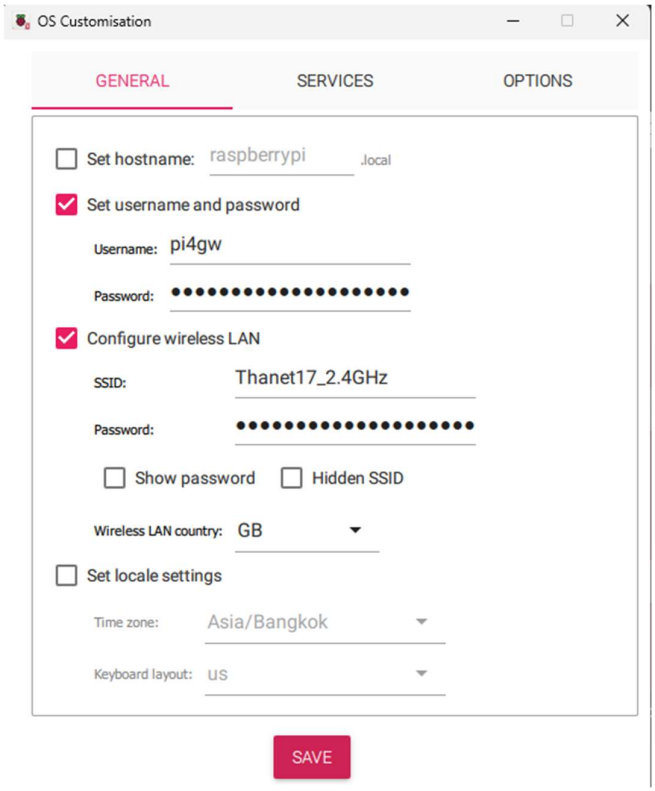


- กด EDIT SETTINGS เพื่อตั้งค่า User และ WiFi เพื่อ login เข้าหน้าDesktop ของ PI

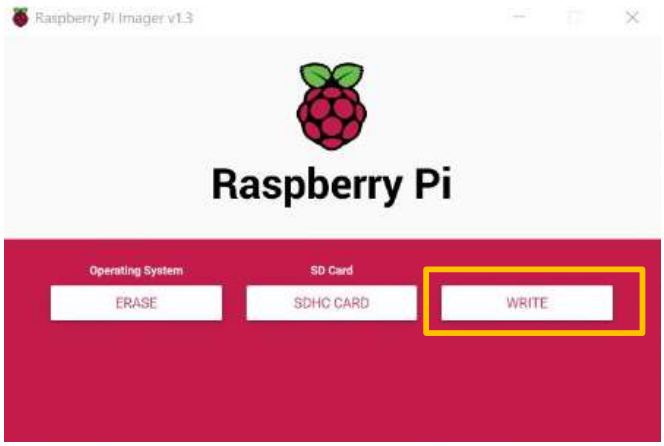


- จากนั้นทำการตั้ง ชื่อ USER , รหัสผ่าน, ชื่อ WiFi และรหัสผ่าน ให้เรียบร้อย เมื่อตั้งเสร็จกด SAVE

  (ห้ามลืมชื่อ กับ รหัสผ่านเพราะจะใช้ login เข้าหน้าDesktop ของ PI)



- กด "Write" ⟶ โปรแกรมจะเขียน OS ลงการ์ดและฟอร์แมตข้อมูลเก่าเป็นอันเสร็จสิ้น



- ทำการลง OS เสร็จเรียบร้อย พร้อมนำไปติดตั้งที่ตัวบอร์ด Raspberry pi

iii) ขั้นตอนที่ 2: ตั้งค่า Raspberry Pi ก่อนเปิดเครื่อง

1) เปิดใช้งาน SSH (ถ้าใช้แบบ Headless – ไม่มีจอ)

- หลังจากเขียน OS ลง MicroSD Card แล้ว ให้นำการ์ดออกและใส่กลับเข้าไปที่คอมพิวเตอร์
- ไปที่ไดร์ฟของ MicroSD (ชื่อ boot)
- สร้างไฟล์ชื่อ ssh (ห้ามมีนามสกุลไฟล์) เพื่อเปิดใช้งาน SSH

2) ตั้งค่า Wi-Fi (ถ้าใช้แบบ Headless)

- ในโฟลเดอร์ boot สร้างไฟล์ชื่อ wpa_supplicant.conf และใส่ข้อมูลดังนี้ (เปลี่ยน SSID และ PASSWORD ตามเครือข่ายของคุณ)

```
country=TH
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
        ssid="YOUR_WIFI_SSID"
        psk="YOUR_WIFI_PASSWORD"
        key_mgmt=WPA-PSK
}
```

- บันทึกและนำการ์ดไปใส่ Raspberry Pi

iv) ขั้นตอนที่ 3: เปิดเครื่องและตั้งค่าเริ่มต้น

1) ใส่ MicroSD Card ลงใน Raspberry Pi

2) เชื่อมต่ออะแดปเตอร์ไฟ USB-C เพื่อเปิดเครื่อง

3) ถ้าใช้จอแสดงผล

- Raspberry Pi จะบูตเข้า GUI (Desktop) และให้ตั้งค่าภาษา, โซนเวลา และเชื่อมต่อ Wi-Fi

4) ถ้าใช้แบบ Headless (ไม่มีจอ)

- ใช้ SSH เพื่อเชื่อมต่อจากคอมพิวเตอร์ (หาหมายเลข IP ของ Raspberry Pi ในเราเตอร์หรือใช้ ping raspberrypi.local)
- ใช้คำสั่ง

```
ssh pi@raspberrypi.local
```

รหัสผ่านเริ่มต้น: raspberry

v) ขั้นตอนที่ 4: อัปเดตระบบและตั้งค่าพื้นฐาน

1) อัปเดตแพ็กเกจ

```
sudo apt update && sudo apt upgrade -y
```

2)

```
sudo raspi-config
```

- ตั้งค่า Wi-Fi, เปิดใช้งาน I2C, SPI, Camera หรือสิ่งที่ต้องการ
- แล้วพิมพ์คำสั่งใน Command Line เพื่อที่จะดูip ของ raspi ใช้คำสั่ง if config

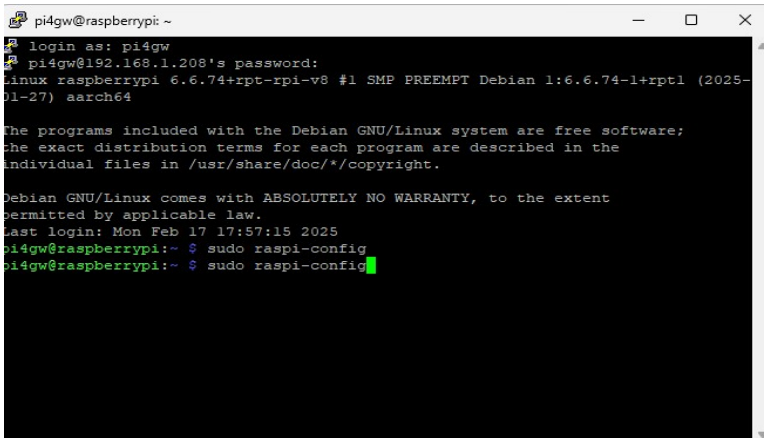
3) เปลี่ยนรหัสผ่านเริ่มต้น

- passwd

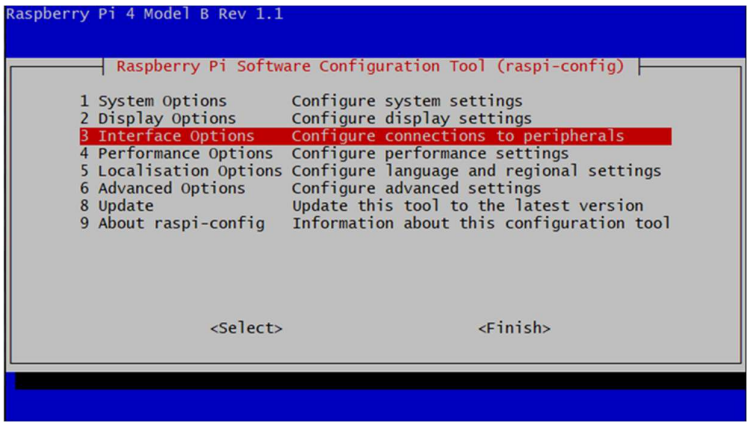
b) ติดตั้ง LoraWan Gateway บน Raspberry Pi 4 สำหรับ WM1302

vi) ต่อมาหลังจากเข้าสู่ระบบใน Raspberry Pi แล้วพิมพ์คำสั่งใน Command Line แล้วจะได้หน้าต่าง config raspi ดังนี้

```
sudo raspi-config
```



1) Select `Interface Options`



- เลือก **SPI** จากนั้นเลือก Yes เพื่อเปิดใช้งาน
- เลือก **I2C** จากนั้นเลือก Yes เพื่อเปิดใช้งาน
- เลือก Serial Port หลังจากนั้นเลือก No เมื่อเจอคำถามนี้ Would you like a login shell... และ เลือก Yes เมื่อเจอคำถามนี้ Would you like the serial port hardware...
- เมื่อ ตั้งค่าเสร็จแล้ว เลือก Finish ให้ทำการ Reboot pi

2) ต่อมา ติดตั้ง git และ ดาวน์โหลด ไลบรารีและโปรแกรมสำหรับ SX1302 LoRa Gateway จาก GitHub พิมพ์คำสั่งนี้ใน Command Line

```
sudo apt update
sudo apt install -y git
cd ~
git clone https://github.com/Lora-net/sx1302_hal
```

3) ย้ายไปยังsx1302_halโฟลเดอร์และคอมไพล์ทุกอย่าง

```
cd ~/sx1302_hal
make
```

4) แก้ไขreset pinสำหรับ SX1302 และ SX1261 ในreset_lgw.sh

```
nano tools/reset_lgw.sh
```

5) ในไฟล์ reset_lgw.sh ค่าเดิมจะเป็นอย่างงี้ ต้องเปลี่ยนเป็นขาของ wm1302 gateway SX1302_RESET_PIN=17 SX1261_RESET_PIN=5

```
# GPIO mapping has to be adapted with HW
SX1302_RESET_PIN=17     # SX1302 reset
SX1302_POWER_EN_PIN=18  # SX1302 power enable
SX1261_RESET_PIN=5      # SX1261 reset (LBT / Spectral Scan)
AD5338R_RESET_PIN=13    # AD5338R reset (full-duplex CN490
reference design
```

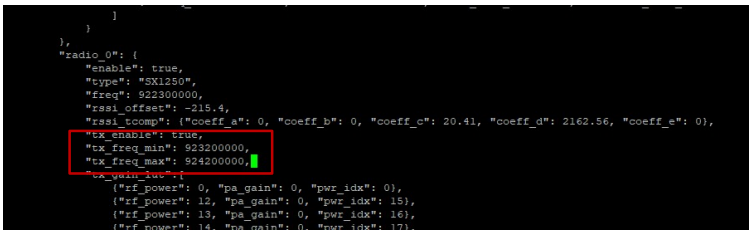
6) บันทึกการเปลี่ยนแปลงเหล่านี้โดยกด CTRL + x และ y สุดท้ายกด Enter เพื่อปิดตัวแก้ไขข้อความ

7) คัดลอกไฟล์ reset_lgw.sh ไปยัง packet_forwarder โฟลเดอร์

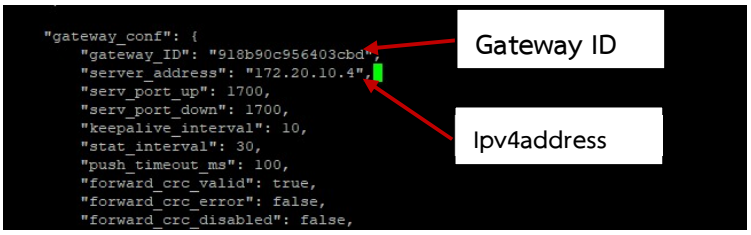
```
cp tools/reset_lgw.sh packet_forwarder/
cd packet_forwarder
```

8) แก้ไขไฟล์ nano global_conf.json.sx1250.AS923.USB

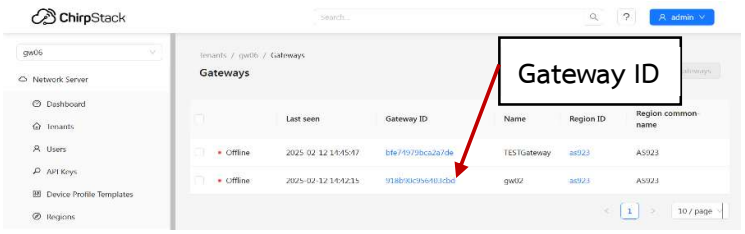
- ส่วนของ radio_0 ให้เปลี่ยน tx_freq_min=9232000000 และ tx_freq_max=9242000000



- ส่วนของ gateway_conf ให้เปลี่ยน ค่า gateway_ID= Gateway IDที่ได้ลงทะเบียนไว้กับchirpstack และ server_address= ip address Pi



หน้า chirpstack



link code gateway:

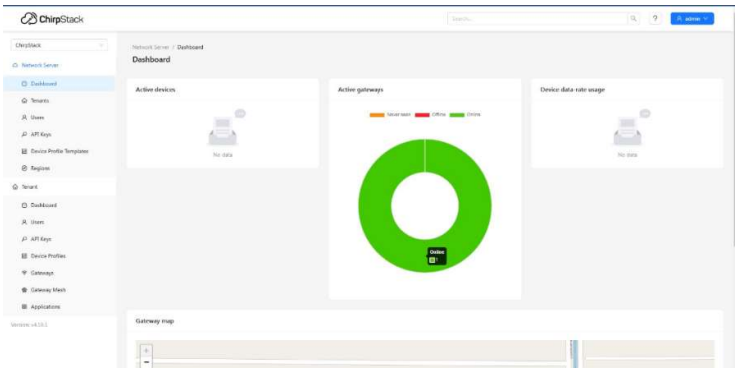
https://github.com/MrArmiami/Project-install/blob/main/Install%20Gateway

9) ทำการรันไฟล์ global_conf.json.sx1250.AS923.USB จากนั้นใช้คำสั่ง lora_pkt_fwd เพื่อ LoraWan Gateway บน Raspberry Pi 4 สำหรับ WM1302

```
./lora_pkt_fwd -c global_conf.json.sx1250.AS923
```

จะได้หน้าตาประมาณนี้ บน terminal pi

10) แล้วสถานะ gateway บน Chirpstack ก็จะขึ้นสถานะออนไลน์



b) ติดตั้ง Chirpstack Gateway OS บน Raspberry Pi 4

การติดตั้ง ChirpStack v4 Docker บน Raspberry Pi 4 ทำตามขั้นตอนนี้:

1) อัพเดทระบบแล้วติดตั้ง Docker บน terminal Pi

```
sudo apt update && sudo apt upgrade -y
curl -fsSL https://get.docker.com | sudo sh
sudo usermod -aG docker $USER2)
```

2) รีบูตเครื่อง เพื่อให้สิทธิ์ Docker ทำงาน บน terminal Pi -> sudo reboot

3) ติดตั้ง Docker Compose และ ตรวจสอบเวอร์ชั่นบน terminal Pi

```
sudo apt install -y python3-pip
pip3 install docker-compose
docker-compose version
```

4) ดาวน์โหลดไฟล์ Compose ของ ChirpStack แล้วเข้าไปที่ไฟล์ chirpstack-docker

```
git clone https://github.com/chirpstack/chirpstack-docker.git
cd chirpstack-docker
```

5) แก้ไขไฟล์ docker-compose.yml โดยเลือก port ที่ไม่ชนกับ Pi

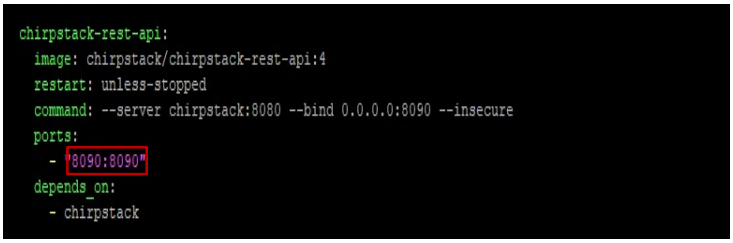
```
nano docker-compose.yml
```

- ส่วน chirpstack เปลี่ยนport เป็น.”8080:8080”

- ต่อมาส่วนของ chirpstack-gateway-bridge เปลี่ยนport เป็น.1700:1700udp แล้วก็อย่าลืมเปลี่ยนภูมิภาค เป็น as923 ด้วย ตอนแรกที่ไฟล์เดิมจะกำหนด ไว้อันเดิมคือ eu868

- ต่อมาส่วนของ chirpstack-gateway-bridge-basicstation เลือกport “3001:3001” และเปลี่ยนภูมิภาคเป็น as923

- ต่อมาส่วนของ chirpstack-rest-api เลือกport “8090:8090”



- ต่อมาส่วนของ postgres กับ redis ไม่ต้องตั้งค่าอะไร

```
postgres:
  image: postgres:14-alpine
  restart: unless-stopped
  volumes:
    - ./configuration/postgresql/initdb:/docker-entrypoint-initdb.d
    - postgresqldata:/var/lib/postgresql/data
  environment:
    - POSTGRES_USER=chirpstack
    - POSTGRES_PASSWORD=chirpstack
    - POSTGRES_DB=chirpstack

redis:
  image: redis:7-alpine
  restart: unless-stopped
  command: redis-server --save 300 1 --save 60 100 --appendonly no
  volumes:
    - redisdata:/data
```

- ต่อมาส่วนของ mosquitto เลือกport “1884:1884”

```
mosquitto:
  image: eclipse-mosquitto:2
  restart: unless-stopped
  ports:
    - "1884:1884"
  volumes:
    - ./configuration/mosquitto/config/:/mosquitto/config/

volumes:
  postgresqldata:
  redisdata:
```

- หลังจากตั้งค่าตามนี้หมดแล้วกดบันทึก (CTRL + X -> Y -> ENTER)

6) แก้ไขไฟล์ chirpstack.toml โดยใช้คำสั่ง nano บน teminal pi ดังนี้

- เข้าไปยังโฟลเดอร์ configuration โดยใช้คำสั่ง cd

  cd configuration

- เข้าไปยังโฟลเดอร์ chirpstack โดยใช้คำสั่ง cd

  cd chirpstack

- จากนั้นแก้ไขไฟล์ chirpstack.toml โดยเปลี่ยน region=as923 กับ port=1884

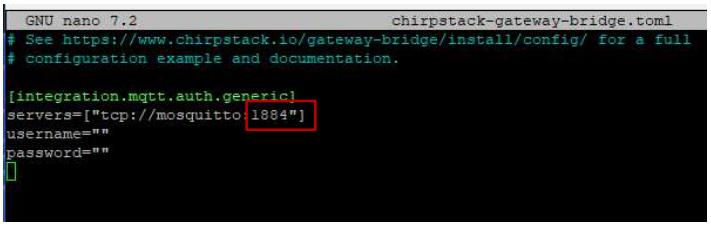
```
# Multiple regions can be enabled simultaneously. Each region must match
# the 'name' parameter of the region configuration in '[[regions]]'.
enabled_regions=[
  "as923",
]

# API interface configuration.
[api]

# interface:port to bind the API interface to.
bind="0.0.0.0:8080"

# Secret.
#
# This secret is used for generating login and API tokens, make sure this
# is never exposed. Changing this secret will invalidate all login and API
# tokens. The following command can be used to generate a random secret:
#   openssl rand -base64 32
secret="you-must-replace-this"



[integration]
enabled=["mqtt"]

[integration.mqtt]
  server="tcp://$MQTT_BROKER_HOST:1884/"
  json=true
```

หลังจากตั้งค่าตามนี้หมดแล้วกดบันทึก (CTRL + X -> Y -> ENTER)

- เข้าไปยังโฟลเดอร์ chirpstack-gateway-bridge โดยใช้คำสั่ง cd

  cd chirpstack-gateway-bridge

- แก้ไขไฟล์ chirpstack-gateway-bridge.toml โดยเปลี่ยน port=1884

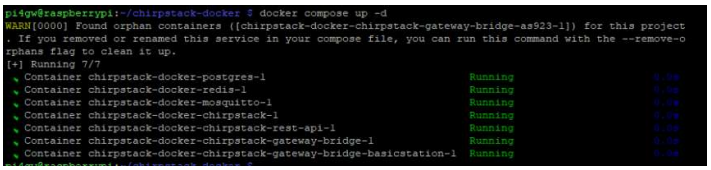


- หลังจากตั้งค่าตามนี้หมดแล้วกดบันทึก (CTRL + X -> Y -> ENTER)และ cd กลับไปที่หน้า chirpstack-docker

  cd chirpstack-docker

- ทำการรันserver chirpstack docker v4 โดยใช้คำสั่งนี้

  docker-compose up -d

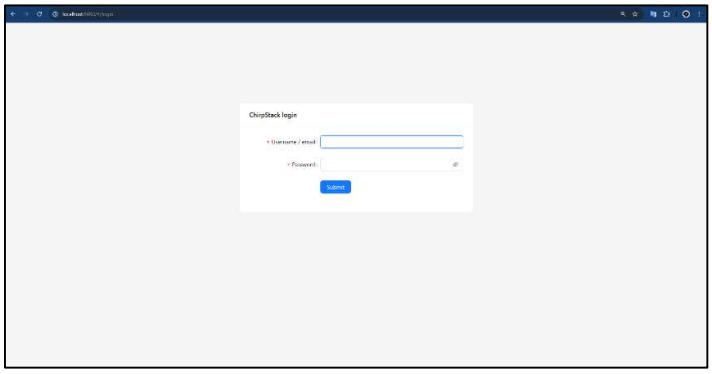


- เข้าสู่ระบบ ChirpStack Web Interface เข้าไปที่ http://localhost:8080/ หรือ http://IP-ADDRESS-OF-YOUR-PC:8080/ จะเจอหน้าจอเข้าสู่ระบบ

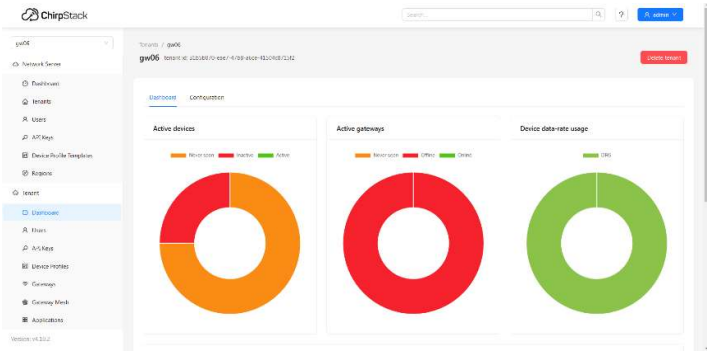
  ข้อมูลสำหรับเข้าสู่ระบบ (ค่าเริ่มต้น)

  Username: admin

  Password: admin



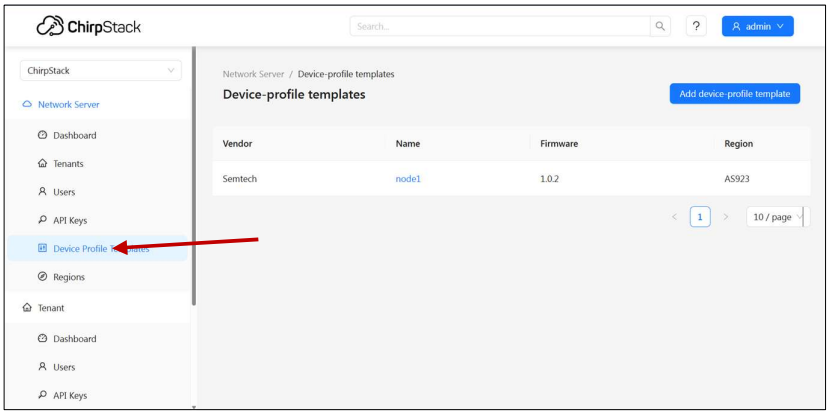
หลัง login เสร็จจะเป็นหน้า ChirpStack Web Interface



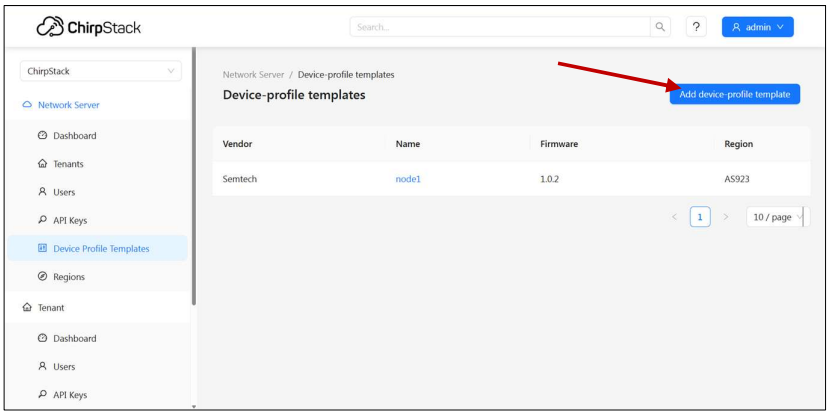
c) การ Add device ใน ChirpStack Web Interface

การเพิ่มอุปกรณ์ ใน ChirpStack Web Interface ทำตามขั้นตอนนี้:

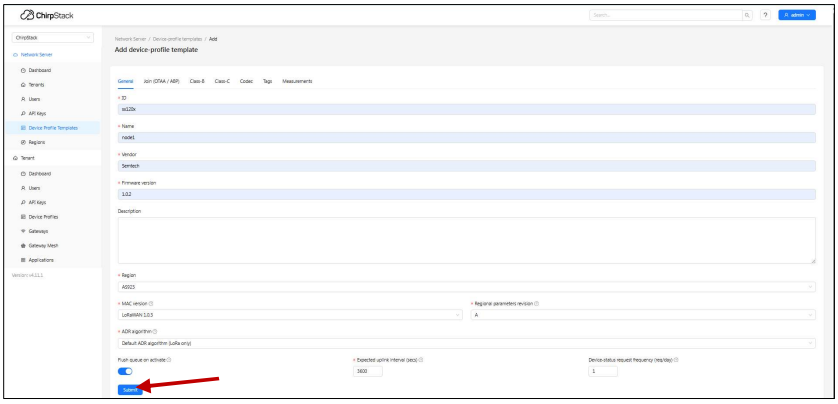
1) Select -> Device-profile templates



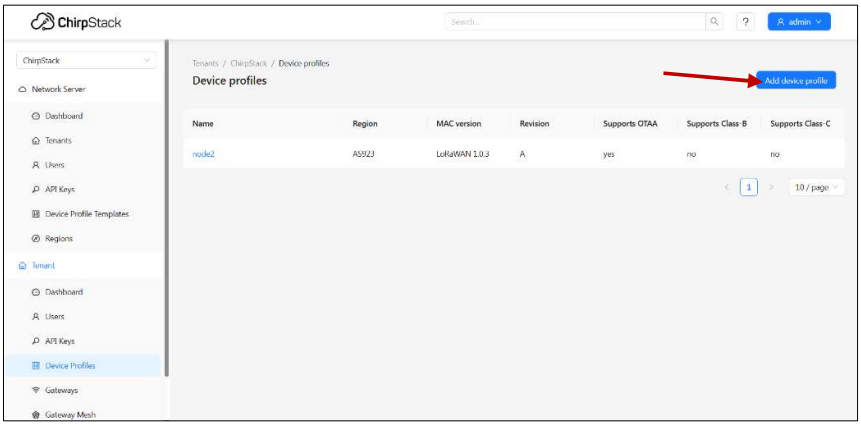
2) Select -> Add device-profile templates



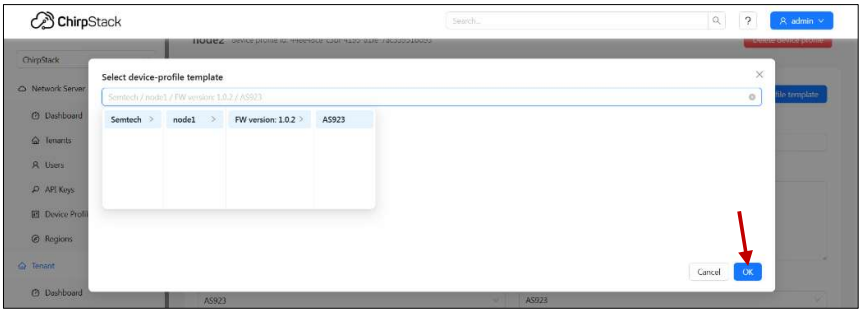
3) ตั้งค่า device-profile templates



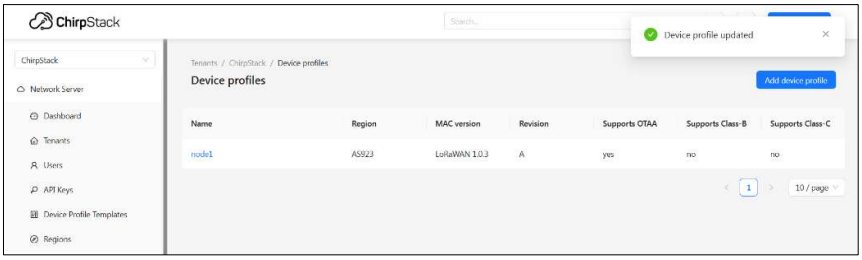
4) ตั้งค่า device-profile Select -> Add device-profile



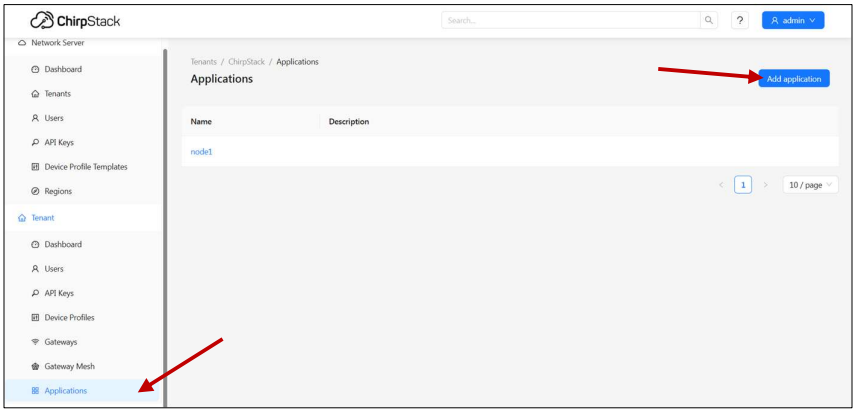
5) เลือก template ที่ได้สร้างไว้



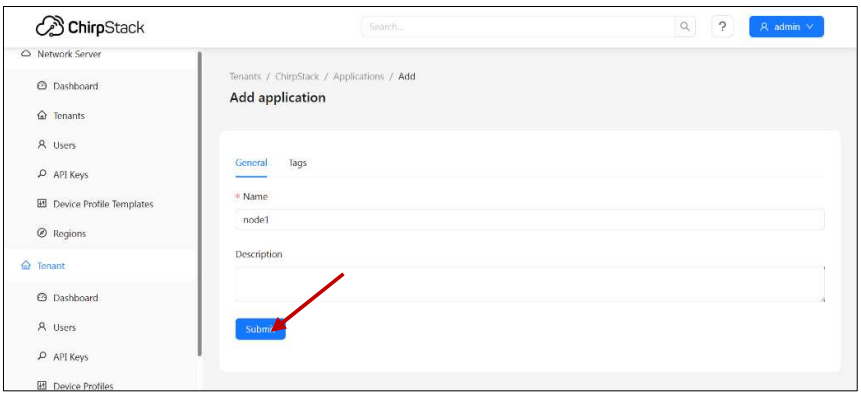
6) เมื่อกดไปแล้วก็จะเพิ่ม template device สำเร็จ



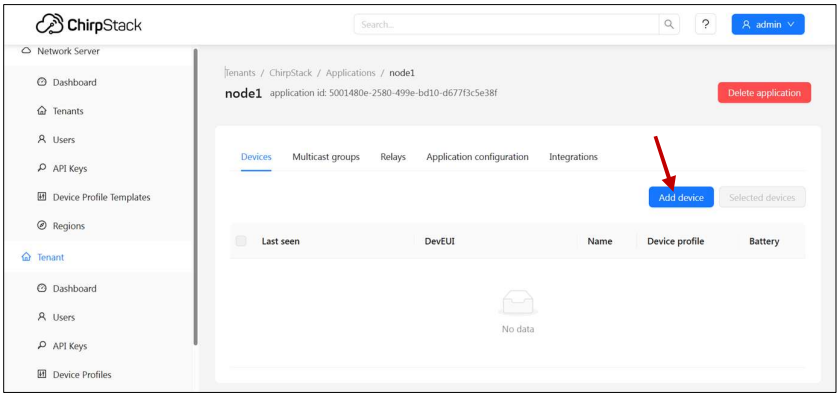
7) click -> Applications เพื่อที่จะ Add device-profile



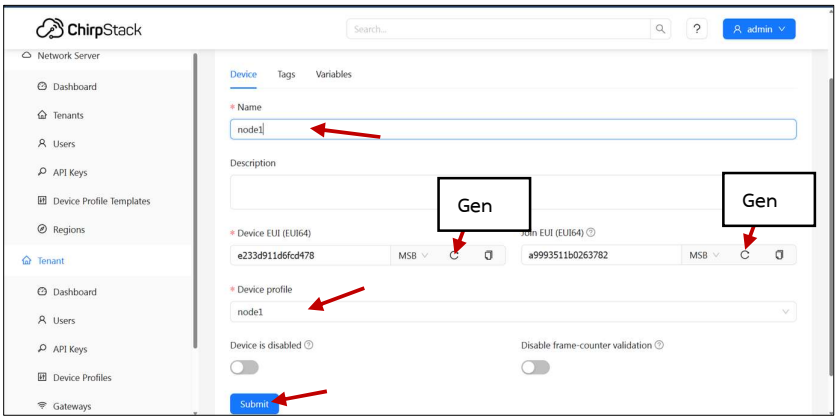
8) ตั้งชื่ออุปกรณ์



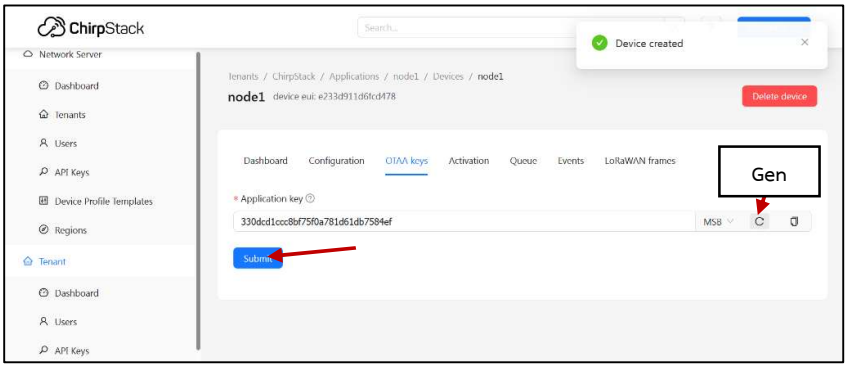
9) click -> Add device



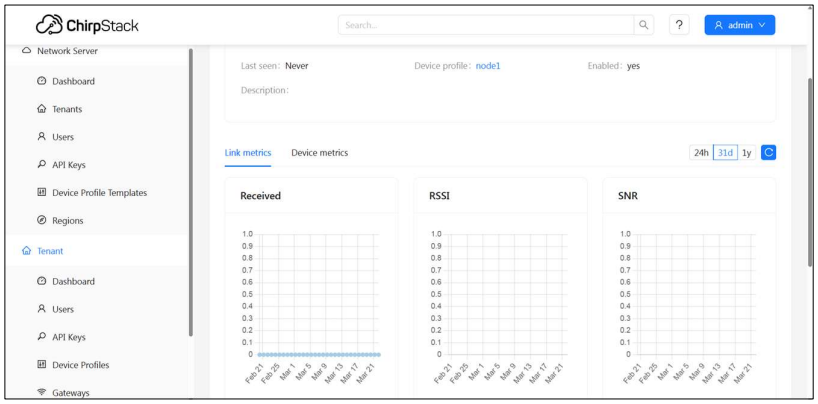
10) หน้า add device กด Gennarate Device EUI และ App EUI (แนะนำให้จดลง google sheet หรือ notepad) และเลือกDevice profile เป็น ที่ได้สร้างไว้ข้างต้น จากนั้นกด submit



ต่อไป กด Gennarate App key แล้วกด submit

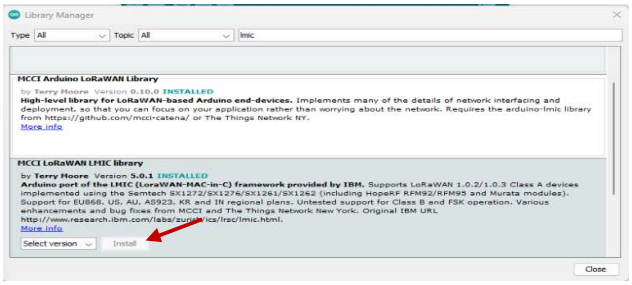


11) จะได้หน้าต่างอุปกรณ์ที่ลงทะเบียนกับchirpstack แล้ว



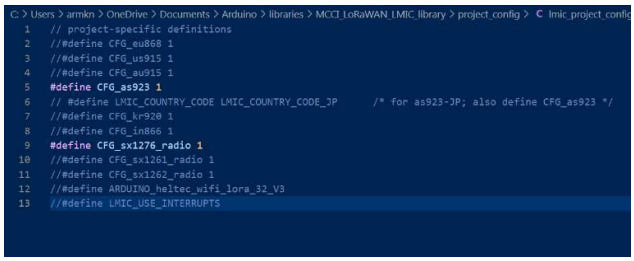
เขียนโค้ดตัวโหนดที่ต้องการจะเชื่อมต่อ แบบ OTAA กับ ChirpStack Web Interface code clickhere: https://github.com/MrArmiami/Project-install/blob/main/Add%20device%20Chirpstack

- ลงไลบรารี่ MCCI_LoRaWAN_LMIC_library ได้ที่ Tools->Manage Libraries->พิมพ์ lmic ->เลือกMCCI LoRaWAN LMIC library



กดInstall

- ต่อไปแก้ไขไฟล์ lmic_project_config.h ที่โฟลเดอร์ MCCI_LoRaWAN_LMIC_libraryเพื่อให้ตัวnode สามารถส่งสัญญานไปหา WM1302Gateway และ chirpstack ได้ path : MCCI_LoRaWAN_LMIC_library-> project_config-> lmic_project_config.h เปิดใช้งาน Region as923 TH



กด SAVE

- ต่อมาในส่วนของโค้ด APPEUI, DEVEUI,และAPPKEY ที่ได้จดไว้ก่อนหน้า ทำการแปลงค่าตามลำดับไบต์ (Endianness Conversion) โดยค่าที่อยู่ในรูปแบบ big-endian (จากซ้ายไปขวา) แปลงเป็น little-endian (จากขวาไปซ้าย) สำหรับการใช้งานใน Arduino หรือ LoRaWAN library ที่ต้องการให้ไบต์เรียงลำดับใหม่ แปลง แค่ APPEUI กับ DEVEUI จากที่ได้ลงทะเบียนไว้คือ

  APPEUI= a9ce7f0dbcd3ec22

  แปลงค่า APPEUI ได้= { 0x22, 0xEC, 0xD3, 0xBC, 0x0D, 0x7F, 0xCE, 0xA9 }

  DEVEUI= 07f9bab5f97a755e

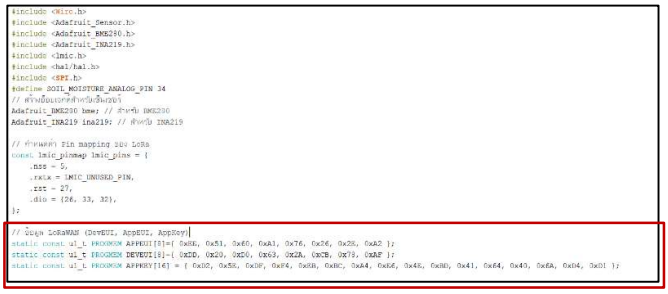
  แปลงค่า DEVEUI ได้= { 0x5E, 0x75, 0x7A, 0xF9, 0xB5, 0xBA, 0xF9, 0x07 }

  APPKEY= e72f8b4b99b9cc5941d7e47da6e371be

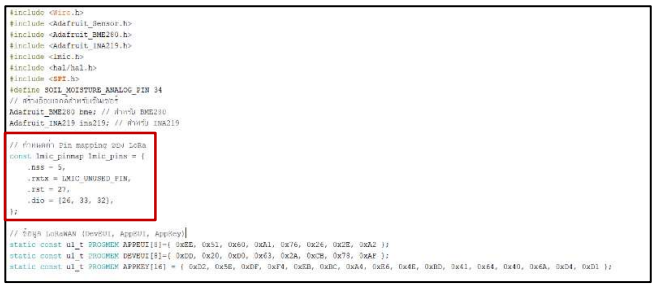
  ไม่ได้แปลงแต่ต้องนำมาไว้ในอาร์เรย์เหมือนกัน

  = { 0xE7, 0x2F, 0x8B, 0x4B, 0x99, 0xB9, 0xCC, 0x59, 0x41, 0xD7, 0xE4, 0x7D, 0xA6, 0xE3, 0x71, 0xBE }

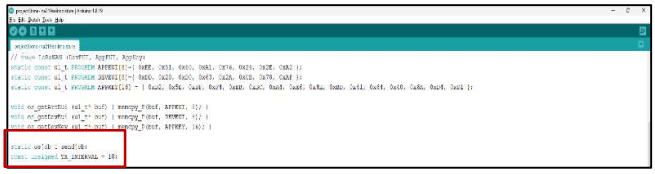
  นำค่าที่ได้มาใส่ในส่วนของ Join OTAA ของโค้ด



- จากนั้นกำหนดขา ตัว lorawan Module ตอนนี้ใช้ sx1276 923 Mhz



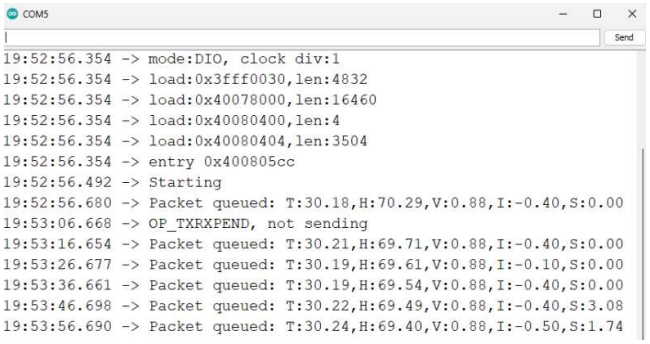
- ส่วนของการปรับค่าการส่งข้อมูลทุกๆ/วิ TX_INTERVAL = 10;



- ส่วนของข้อมูลที่จะส่งไปยัง chirpstack

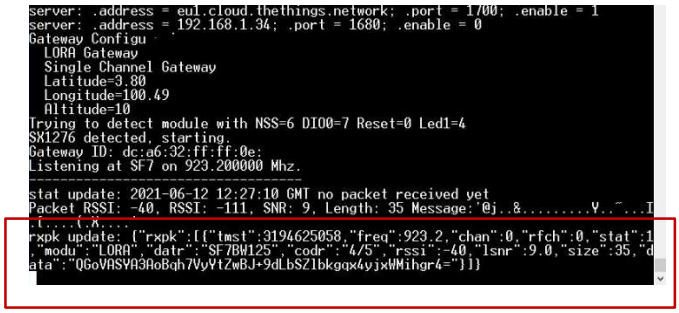
- เมื่อปรับโค้ดเสร็จแล้ว กด Upload

- ผลที่ได้บน Serial monitor



- ผลที่ได้บน Wm1302 Gateway

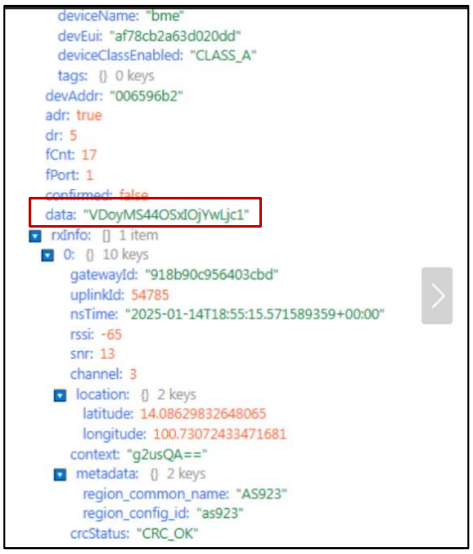
  Gateway รับข้อมูลจากโหนด แล้วส่งขึ้น chirpstack



- ผลที่ได้บน chirpstack

  device จะขึ้นสถานะ online

แล้วdata ที่ได้รับจาก gateway จะมี ดังนี้



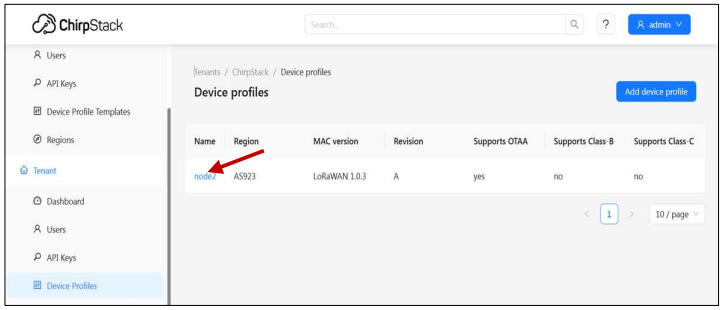
data จะถูกเข้ารหัสในรูปแบบ Base64 จึงต้องสร้าง function()decode เพื่อแปลงข้อมูลที่gatewayส่งมาให้อยู่ในรูปแบบตัวเลขปกติ

function() code : https://github.com/MrArmiami/Project-install/blob/main/function(%20)Decode

ขั้นตอนการเขียน function()decode

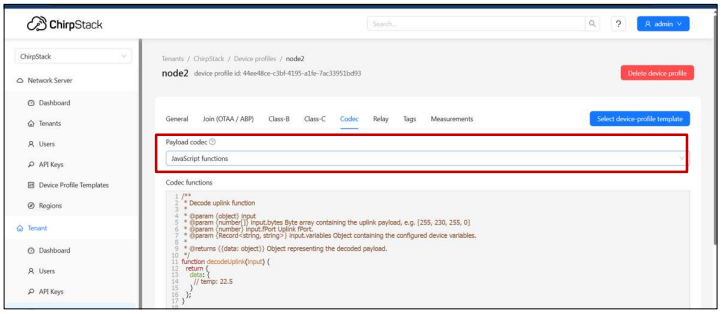
1) คลิ๊ก -> Device Profile

2) เลือก -> node2



3) คลิ๊ก Codec แล้วเลือก Javascript functions จากนั้นเขียนโค้ดที่ตามที่ให้

เป็นอันเสร็จสิ้นในการเขียน functions decode บน Chirpstack



หลังจากสร้าง functions decode เสร็จจะได้ข้อมูลที่เป็นตัวเลขมา

- เป็นการAdd อุปกรณ์ เสร็จสิ้น

- ADD device เพิ่มอีก 7 ชุดตามวิธีด้านข้างต้น

d) ติดตั้ง Local Server บน Raspberry Pi 4

สร้าง Local Server บน Raspberry Pi ที่รันบนพอร์ต 5000 ได้โดยใช้ **Flask** ใน Python ตามขั้นตอนนี้:

1) ติดตั้ง Flask โดยใช้ terminal pi พิมพ์คำสั่ง

   pip install flask

2) สร้างไฟล์ app.py

```
from flask import Flask
app = Flask(__localserverBlynk __)
@app.route('/')
def home():
return "Hello, Raspberry Pi!"
if __ localserverBlynk __ == '__main__':
app.run(host='0.0.0.0', port=5000)
```

3) รันเซิร์ฟเวอร์

   python app.py

4) เป็นการเสร็จสิ้นการติดตั้ง