# Opzet Blockchain Server

Om zelf het blockchain server te starten moet je de github <https://github.com/JorisTDP/DIHM/> clonen. Je zal de volgende dependies moeten installeren via pip install:

* Flask (en werkzeug)
* Requests
* Base64
* Cryptography(.fernet)
* Json
* Datetime ( en timeit)

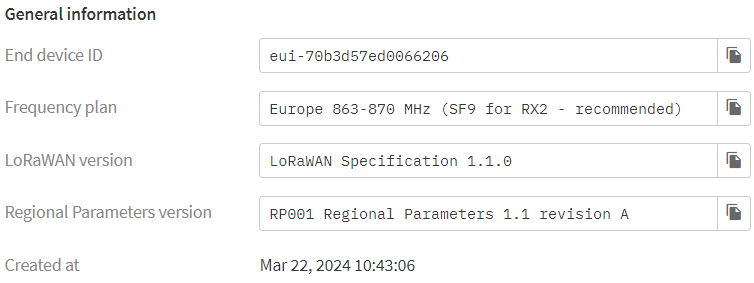
Als je de server wilt opstarten moet je zowel peer.py en run\_client py opstarten. 🡪 ``sudo python3 run\_client.py | Sudo python3 peer.py

Deze worden dan uitgevoerd op peer = localhost:80, client = localhost:8000

Je kan inloggen op de client via de webinterface met account Joris, wachtwoord: wachtwoord.  
In het admin account kun je accounts toevoegen en verwijderen, bijvoorbeeld voor de consensus. Als je een block wilt toevoegen moet je een consensus account gebruiken om het block goed te keuren.

Als je de blockchain wilt restarten moet je blockchain.txt verwijderen en het server opnieuw opstarten. Er wordt automatisch een nieuwe blockchain gegeneerd die leeg is.

# Opzet schip module

Maak een account aan bij The Things Network, Ga naar <https://console.cloud.thethings.network/> en registreer een nieuwe applicatie. Maak een nieuw end-device aan met de volgende specificaties:  
  
Ga naar general settings->join settings->reset join nonces->enabled.

Noteer de DEV\_EUI, APP\_KEY en NWK-key, define deze in de config.h bij de DIHM-module code.

Upload de laatste code (DIHM-module.ino). Sluit de antenne aan via de LoRa connector aan de achterkant. Sluit de module via usb aan en lees de serial monitor uit. Er zal herhaaldelijk geprobeerd worden LoRa verbinding te maken. De codes worden in de serial monitor geprint (0 = success, -1116 of -1102 = error). Als verbinding goed is zal het DevAddr ook overeenkomen met de DevAddr in de live data tab van The Things Network.

Als je met de module verbinding wil maken met de blockchain server moet je in TTN bij integrations een webhook toevoegen die verwijst naar <ip server>:80/webrequest, die wordt geactiveerd bij uplink messsages.

Ook moet je nog een API key generen en deze in peer.py in de header zetten:

headers = { "Authorization": "Bearer NNSXS.[HIER API KEY]

Ook moet je de url veranderen met de nieuwe namen van de applicatie, webhook en device van de TTN applicatie:  
[https://eu1.cloud.thethings.network/api/v3/as/applications/[name of application]/webhooks/[name of webhook]/devices/[name of device]/down/push](https://eu1.cloud.thethings.network/api/v3/as/applications/%5bname%20of%20application%5d/webhooks/%5bname%20of%20webhook%5d/devices/%5bname%20of%20device%5d/down/push)

Als dit allemaal goed is ingevuld kan je naar <ip server>:80/webrequest een post message sturen en dan zal er in de TTN console een antwoord te zien zijn. Ook zal de schip module na LoRa verbinding te hebben en dus een dev\_addr in de serial monitor uit te lezen. Om verbinding te maken met de server moet je de apk installeren in de apps folder. Door de app op je telefoon te gebruiken kan je via BLE het apparaat selecteren en verbinding maken. Dan klik je op connect via LoRa->Retrieve Blockchain, en dan zal je in de serial monitor ‘Req received’ zien, gevolgd door enkele verbinding codes. Als alles goed is ingevuld en er LoRa bereik is zal de module naar TTN verbinden waar de webhook naar de blockchain server verwijst waar de /webrequest pagina wordt aangeroepen en deze het antwoord weer naar ttn stuurt wat de module opvangt.

Vragen?

Email: [joriselfferich@gmail.com](mailto:joriselfferich@gmail.com)