

« Internet Of Things » :

Solution LoRa pour la surveillance de la pollution intérieure d'un bâtiment

Module Objets Communicants

Master 2 SETI, APP5 EES

Isabelle Vin (isabelle.vin@universite-paris-saclay.fr)

Introduction

L'objectif du TP est de réaliser un système d'Internet des Objets pour la surveillance de la pollution intérieure d'un bâtiment (salle de TP de Polytech) en utilisant la technologie LoRa, particulièrement adaptée pour les communications sans fil dédiées à l'IoT, caractérisées par de faibles débits, une faible consommation et, souvent, un besoin de longue portée. Les données des capteurs de pollution seront mises à disposition sur le cloud via un serveur MQTT ouvert sur Internet, et seront visualisées en temps réel en parallèle sur une interface graphique pour réaliser le monitoring de la qualité de l'air intérieur, à distance.

L'objectif pédagogique du TP est d'utiliser la technologie radio LoRa et de découvrir quelques outils logiciels open source incontournables, pour mettre en œuvre une chaîne IoT :

- **le serveur d'applications (logiciel) Node-RED** : il s'agit d'un outil de gestion de flux permettant de connecter des devices entre eux et/ou à un poste de contrôle (ex : PC distant), ou encore de transmettre les données à une application ou s'y connecter (twitter, MQTT, envoi de sms par le service twilio, e-mails, stockage dans une base de données, requête HTTP, etc..) et vice-versa. Il sera détaillé plus amplement par la suite.

Note importante : Dans un réseau LoRaWAN, ce sont les stations de base LoRaWAN (ou passerelles LoRaWAN) qui transmettent les paquets de données envoyées par les devices en LoRa vers la plateforme cloud (connexion directe au réseau Internet) en passant par un réseau de backhaul (TCP/IP, voire 3G, 4G..). Si nous étions sous la couverture d'une station de base LoRaWAN opérationnelle dans notre secteur, nous aurions pu l'utiliser pour transmettre les données de nos capteurs sur le serveur The Things Network (TTN), qui aurait ensuite transmis les informations des devices à la bonne application (et vice-versa) tout comme le fait Node-RED. La solution d'utiliser un « relais » LoRa connectée à un PC où est installé un serveur Node-RED est donc une alternative à l'utilisation d'une passerelle LoRaWAN connecté au serveur TTN en ligne.

- **le protocole MQTT** : il permet de publier des données (données capteurs par exemple) et/ou de s'y abonner (au réseau de capteurs par exemple) pour récupérer des données. Il s'agit d'un protocole publish/subscribe.
- **le monitoring des données récoltées sur un tableau de bord (ou dashboard)**. Dans un premier temps, nous utiliserons le dashboard intégré dans Node-RED. Nous utiliserons ensuite le dashboard de Grafana, plus riche en fonctionnalités, qui permet par ailleurs d'être relié en temps réel à une base de données (InfluxDB, Graphite, MySQL, OpenTSDB, Prometheus, etc..).
- **le stockage des données dans une base de données InfluxDB**. Il s'agit d'une base de données organisée par séries chronologiques, qui contient un horodatage, une valeur mesurée et éventuellement des métadonnées.

NB : Contrairement aux bases de données SQL, relationnelles, les bases de données InfluxDB, « noSQL », sont plus flexibles, permettent d'avoir une architecture logicielle autre que sous forme d'un tableau de propriétés fixes, et permettent de distribuer facilement les données sur plusieurs serveurs. En contrepartie, les requêtes noSQL sont moins riches et plus complexes.

Ci-dessous le synoptique du système global de surveillance de pollution indoor.



Les devices utilisés sont composés d'une carte microcontrôleur Nucleo L073RZ (STM32), équipée d'une carte d'extension avec le module radio LoRa SX1272 et son antenne sur la bande des 868 MHz.

Les capteurs de qualité de l'air, fournis par Seeed Studio (Grove Air quality sensor v1.3), sont basés sur des semi-conducteurs MOSFET (substrat alumine Al_2O_3 et composite céramique $\text{Al}_2\text{O}_3\text{-Cr}$) dont la conductivité - et donc la tension (V_{sensor}) sur l'électrode de mesure du capteur - augmente avec la concentration des particules cibles (CO, alcool, acétone, diluant, formaldéhyde, fumée...). Le capteur fonctionne suivant le même principe que la cellule électrochimique, avec une électrolyte solide (le semi-conducteur MOS) qui est chauffée à une certaine température dite de travail (V_{heater} : potentiel électrique associé). Les électrodes produisent des réactions chimiques qui transforment l'énergie chimique du combustible (particules-cibles : alcool, acétone, ..) en énergie électrique (f.e.m. qui se forme sur les électrodes). Pour les capteurs Grove Air Quality, en fonction du potentiel mesuré sur l'électrode de mesure (V_{sensor}), l'interprétation est la suivante :

- $V_{\text{sensor}} > 700 \text{ mV}$: pollution élevée
- $700 \text{ mV} > V_{\text{sensor}} > 300 \text{ mV}$: faible pollution
- $V_{\text{sensor}} < 300 \text{ mV}$: air sain

Ces capteurs seront connectés sur certaines cartes Nucleo pour constituer des « modules capteurs émetteurs » qui enverront régulièrement leurs données.

Réalisation du système de mesure de pollution indoor connecté


Avant de commencer, vous pouvez récupérer le dossier « Fichier étudiants APP5A.zip » sur le lien suivant :

https://drive.google.com/drive/folders/1t6RDks7-bmn4vr1ESu8kI1nsD9xiBWPl?usp=drive_link

Le dossier contient :

- l'API des capteurs de pollution utilisés en TP
- l'exécutable pour remettre à jour le firmware des microcontrôleurs Nucleo
- un dossier « InfluxDB - Grafana » contenant les fichiers nécessaires pour lancer les serveurs

- l'énoncé de ce TP

 Cet énoncé est le fil conducteur pour développer un système IOT s'appuyant sur du LoRa, vous êtes bien entendu encouragés à personnaliser par la suite votre application IOT.

1) Étape 1 : Communication LoRaWAN pour la remontée des données capteur

- ✓ Aller récupérer l'exemple « SX1272 Ping Pong » sur le site de mbed. Ce programme permet à 2 cartes Nucleo L073RZ équipées du module radio LoRa SX1272MBxAS de s'envoyer respectivement les messages « ping » et « pong ».

Vérifier que la communication radio fonctionne avec cet exemple (se regrouper par 2 binômes).

Note : Si un problème survient au moment de téléverser le programme sur la carte Nucleo, vérifier que les jumpers sont bien positionnés. Il peut s'agir également d'un problème de firmware à mettre à jour sur les cartes Nucleo : dans ce cas, lancer l'application « ST-LinkUpGrade.exe » fournie dans le dossier « firmware upgrade ».

- ✓ A partir de ce programme-exemple, préparer 2 programmes :
 - l'un pour la carte « émettrice » reliée au capteur de qualité de l'air, qui se chargera d'envoyer à intervalle régulier l'indice de qualité d'air Q et la valeur de tension capteur (V_{sensor}) par transmission sans fil LoRa.
Utiliser l'API du capteur dans le dossier « Grove_Air_Quality_Sensor_Library ».

Instancier un objet de type « Air_Quality ». Veiller à bien l'initialiser avec les pins du capteur dans le bon ordre : pin1 → A2 (V_{sensor}), pin2 → A1 (V_{heater}).

Le message à envoyer sera stockée dans *Buffer* de longueur *BUFFER_SIZE*, contenant des entiers. La première case de Buffer contiendra un identifiant compris entre 01 et 13 (selon le binôme d'étudiant), les 2 autres cases contiendront les données capteur Q et V_{sensor} .

- l'autre programme pour la carte « réceptrice » reliée au PC fixe, qui récupère les données capteur et les transmet sur le port série du PC (LoRaGW). Pour transmettre plusieurs valeurs sur un envoi série, concaténer les valeurs (int) de : l'identifiant capteur, Q et V_{sensor} , sur respectivement 2 digits, 1 digit et 3 digits.

2) Étape 2 : Interfaçage de la « LoRaGW » avec Node-RED pour gérer les flux de données

Node-RED est une application qui permet de relier des sources de données (port série, IHM, http, objets connectés HW, raspberry, messages MQTT, OpenWeathermap...) à des blocs de traitements (locaux ou distants) et des sorties (emails, sms, renvoi port série, HW, bases de données, dashboards...) et donc de créer des chaînes de traitement dans un environnement web. Il repose sur *node.js* (javascript) qui est déjà installé sur les PC de la salle de TP et qui permet d'utiliser le langage Javascript sur le serveur même en dehors du navigateur.

- ✓ Tout d'abord, installer le serveur Node-RED. Il faut ouvrir le shell et se placer dans C:\Windows\System32, puis taper `> npm install -g -unsafe-perm node-red`
- ✓ Lancer ensuite le serveur node-red depuis l'emplacement où il a été installé (C:\Users\ votre_compte\AppData\Roaming\npm) via le shell : `> node-red`

Vous pouvez ensuite accéder à l'interface de l'application node-RED (**port 1880**)

- depuis votre PC où le serveur est installé (fenêtre navigateur : <http://localhost:1880>)
- depuis un autre PC s'il est connecté en local au PC où est installé node-RED (http://<adresse_ip_pc_serveur>:1880)

- ✓ Installer les modules (« nodes ») suivants, depuis l'interface GUI de Node-RED. Ces modules permettront d'augmenter les fonctionnalités du serveur node-RED installé :

node-red-node-serialport	permet de récupérer ou d'envoyer des données sur le port série du PC courant
node-red-dashboard	Tableau de bord « interne » de node-red : permet d'afficher des graphes, etc. et des IHM pour l'affichage
node-red-contrib-influxdb	Permet d'écrire ou de faire des requêtes sur une base de données de séries chronologiques InfluxDB
node-red-contrib-sms-twilio (optionnel)	Permet d'envoyer des sms (de notification par exemple) via la plateforme en ligne twilio. <i>Note : Pour pouvoir envoyer des sms, il faudra aller sur le site de twilio, créer un compte et récupérer le SID, le Token et le numéro de téléphone de twilio (format +336...). Ces informations sont nécessaires pour renseigner le node « twilio out »</i>

Vous pouvez également les installer via le shell par la ligne de commande suivante :

> `npm install nom_du_module_a_installer`

- ✓ Regarder et tester les fonctionnalités de Node-RED (inject, twilio, twitter, serialport, function, tweet, etc..).

NB : Node-RED utilise le langage Javascript. Les fonctions doivent donc être codées dans ce langage. Par défaut, le message récupérée en input s'appelle **msg**, avec au minimum un champ **msg.payload** et un champ **msg.timestamp**.

- Quelques exemples :

- récupérer la valeur reçue (payload) et la multiplier par 100 :

```
var Newmsg = {payload : 100*msg.payload} ;  
return Newmsg ;
```

- récupérer un octet binaire (exemple : la case 2) d'un buffer :

```
var Newmsg = {payload : msg.payload[2]} ;  
return Newmsg ;
```

- récupérer un *char buffer* et le convertir en nombre :

(ici dans l'exemple, on récupère les chars d'indice 4 à 7 (8 exclus))

```
var str = msg.payload ;  
var res= str.slice(4,8) ;  
msg.payload = parseInt(res) ;  
return msg ;
```

- émettre un nombre aléatoire entre 0 et 100 :

```
msg.payload = Math.round(Math.random()*100);  
return msg;
```

Pour plus d'informations sur la définition des fonctions dans Node-red, voir :

<https://nodered.org/docs/writing-functions#writing-a-function>

- ✓ Récupérer sur le port série les données entrantes (les données reçues en série sur 6 digits, donc 6 chars) et les afficher sur le debugger.

Note importante : La communication série entre un device et Node-red ne pourra pas s'effectuer si le device est simultanément en communication série avec un terminal (communication point-à-point!)

3) Étape 3 : Utilisation de MQTT pour s'abonner et récupérer les données capteurs par différents publishers

MQTT est un protocole/service de type publish & subscribe, qui permet de :

- publier sur un « topic », sujet qui existe déjà où que l'on crée (ex : TP_IOT_2022/salleB007/air_quality)
- s'abonner à un topic (où plusieurs publishers peuvent contribuer bien entendu) et recevoir des notifications.

Le serveur (« broker ») MQTT peut être soit installé en local (sur le PC directement) soit *online* (mosquitto, hivemq, mosca, etc..). Pour limiter l'installation de serveurs, nous allons utiliser un serveur MQTT over websockets. Plusieurs possibilités :

	Adresse du serveur	Port	MQTT client tool (accès sur le web)
mosquitto	Test.mosquitto.org	1883/ 8883 (sécu.SSL)	eclipse.org/paho/clients/js/utility/
emqx	Broker.emqx.io	1883	
HiveMQ	broker.hivemq.com	1883	hivemq.com/demos/websocketclient/
	broker.mqstdashboard.com		

Dans notre cas, nous utiliserons le serveur MQTT de mosquitto. Chaque capteur (relié à une carte nucleo) sera un publisher sur un topic particulier (ex : TP_IOT_2022/salleB007/air_quality/sensor01). On pourra ensuite récupérer indépendamment et afficher les data sur différentes courbes.

Pour ne pas congestionner les communications LoRa et éviter les interférences entre eux, on ne déploiera que 5 modules émetteurs avec capteurs (capteur connecté) qui émettront à des intervalles de temps distincts. Nous conserverons les modules émetteurs d'identifiants : 01, 04, 07, 10, 12, et qui enverront leurs données respectivement tous les : 7s, 11s, 13s, 17s, et 19s.

- ✓ Utiliser un nœud MQTT « output » pour relayer les données (on ne s'intéressera qu'à V_{sensor}) d'un capteur en fonction de votre « couverture » (l'enseignant vous indiquera le plan pour chaque binôme de la classe), données reçues via le port série (récupérées par la LoRaGW). Selon l'ID du capteur, vous publierez sur le topic correspondant : « TP_IOT_2022/salleB007/air_quality/sensorX », X étant l'ID du capteur sur 2 digits.

*Note importante : Dans le node MQTT de Node-RED, ne pas mettre l'ID client (défini sur le MQTT client tool), car le pilotage simultané (depuis la plateforme mqtt client tool **ET** depuis node-RED) sera impossible !*

- ✓ Réaliser plusieurs « scénarios de flux » (au total 5) avec des nœuds MQTT « input » cette fois, où l'on récupérera les données capteurs publiées sur les topics MQTT correspondant aux 5 modules émetteurs, et les afficher.

Note importante : Ne pas oublier de convertir les données MQTT publiées et récupérées (string) en nombres entiers, afin qu'elles puissent en aval être affichées sur le tableau de bord dans l'étape suivante!

4) Étape 4 : Monitoring de la qualité de l'air : tableau de bord (« dashboard »)

La dernière étape de la chaîne IoT de surveillance de pollution est le monitoring des données sur une interface graphique à distance (service d'application de type « Dashboard »).

4.a) Solution 1 : Utilisation du dashboard de node-red

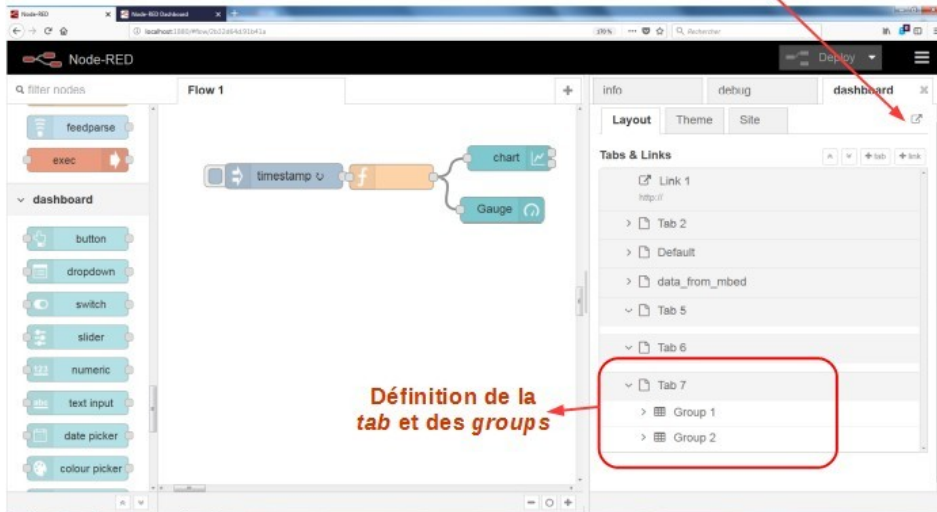
Nous allons utiliser pour cela le tableau de bord de Node-RED.

Dans l'onglet *dashboard* à droite, il faut créer une table (*tab*) par feuille graphique. Une *tab* sera ensuite composée de 1 à plusieurs groupes (*group*) qui correspondront à une partie (un emplacement/cadran) de la feuille graphique.

Vous pouvez ensuite utiliser les différents outils graphiques d'entrée/sortie/fonction mis à disposition (text, button, gauge, chart, switch, slider, text input...). Lorsque vous en insérez un, il faut bien entendu indiquer le *group* et la *tab* correspondants, afin que l'application du dashboard sache sur quelle feuille et où le placer.

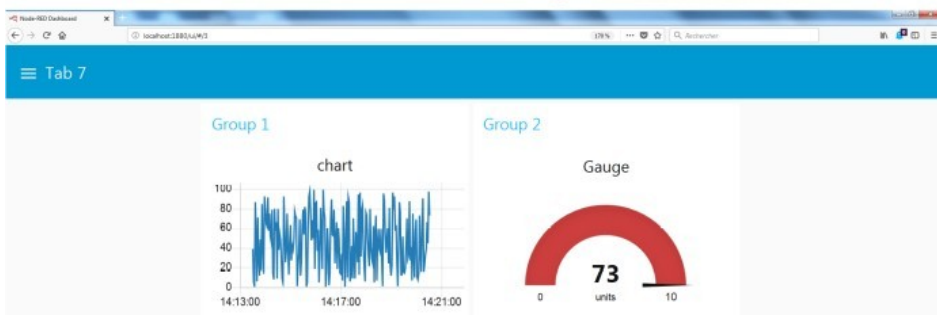
- ✓ Prendre connaissance des différents outils de Node-RED dashboard
- ✓ Réaliser les graphiques de Q (facteur de qualité) et de V_{sensor} pour chacun des 5 modules capteurs.

Lien pour visualiser le dashboard



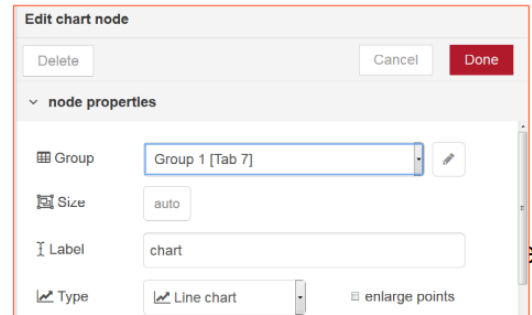
Définition de la tab et des groups

Visualisation du dashboard

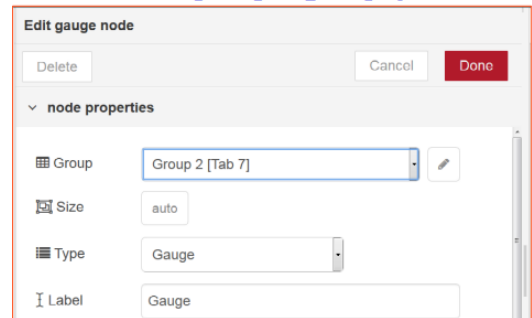


pas la bonne feuille graphique!

Editer le chart (graphique)



Editer la gauge (jauge)



Note importante : Ne pas oublier de bien sélectionner (sur la page de l'application dashboard) la bonne tab en haut à gauche, sinon vous n'afficherez

Le problème de cette solution réside dans la non-persistance des données (les données ne sont pas sauvegardées une fois que l'on ferme le serveur node-red).

La solution alternative est d'utiliser une base de données pour le stockage, et d'utiliser un dashboard « indépendant » qui récupère les données de la base de données en temps réel. Pour ce faire, on se propose, dans la dernière partie, d'utiliser la base de données non-SQL **InfluxDB**, en collaboration avec le logiciel Grafana pour la partie interface graphique.

4.b) Solution 2 : Sauvegarde des données dans une base de données InfluxDB, et interface graphique de Grafana relié à InfluxDB

- ✓ Lancer le **serveur InfluxDB** : double-cliquer sur l'exécutable « influxd.exe ». L'application InfluxDB est alors accessible depuis le web (si votre PC est connecté au PC serveur) à l'adresse IP du PC serveur, **port 8086**.

Note importante : Il est possible qu'il y ait un conflit sur le port 8086 (le serveur voudra alors imposer le port 8088) car l'application « WAPT Service » est peut-être en cours sur ce même port. Dans ce cas, lancer le gestionnaire de tâches de Windows et arrêter ce service. Sur le fichier « influxdb.conf », décommenter la ligne 15 et mettre 8086.

- ✓ Lancer ensuite l'invite de commande client d'InfluxDB « `influx.exe` » permettant de communiquer avec le serveur InfluxDB. Le serveur est accessible à tout ordinateur connecté au PC où est installé InfluxDB.

Nous allons créer la database initiale via des lignes de commande sur la console client :

- > **CREATE DATABASE myDB** *création d'une base de données « myDB »*
- > **SHOW DATABASES** *montrer les bases de données existantes (permet de vérifier la création)*

On peut éventuellement créer un utilisateur avec un mot de passe et le rattacher à cette base de données

- > **CREATE USER Moi WITH PASSWORD 'password'**
- > **GRANT ALL ON myDB to Moi**

et créer une politique de conservation de la base de données, ou encore effectuer des copies (sauvegardes) de la base de données :

- > **CREATE RETENTION POLICY one_day_only DURATION 1d REPLICATION 3**
pour les unités de durées : m (minute), h (hour), d (day), w (week) et inf (infinity).

Pour effacer une base de données :

- > **DROP DATABASE myDB**

Pour pouvoir écrire ultérieurement sur la base de données via l'interface Node-red, il faudra donc au préalable la créer via ces lignes de commande. Vous pourrez ensuite visualiser sur la console influx la ou les mesures stockées dans la base de données :

- > **USE myDB** *on se place dans la bonne base de données*
- > **SELECT * FROM ma_mesure** *affichage des données de la mesure « ma_mesure »*

Si vous souhaitez visualiser sur le web les bases de données qui sont stockées sur votre serveur InfluxDB depuis le PC local, il faut entrer l'adresse suivante :

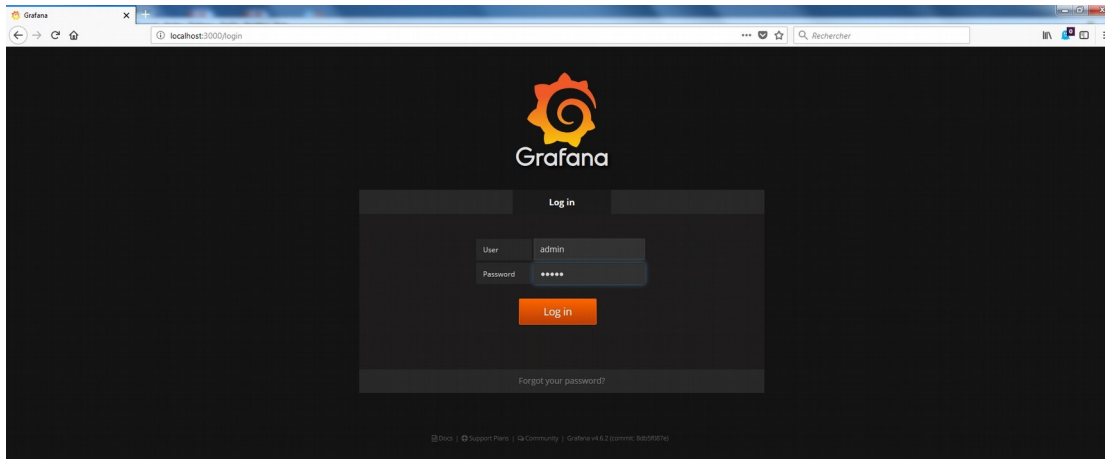
http://<adresse_ip_duPC_serveur>:8086/query?q=SHOW+DATABASES

- ✓ Créer la base de données (via la console influx) qui recevra les mesures des 5 différents modules capteurs.
- ✓ Dans Node-RED, vous pouvez à présent introduire 5 scénarios de flux associés aux données reçus (Q, ou V_{sensor}) issus des 5 topics des 5 modules capteurs nœud de sortie influxdb out dans un de vos scénarios de flux.
Il faut bien y préciser un nom pour le champ *measurement* (par exemple « Q_sensor07 » ou « Vs_sensor01 ») et configurer le serveur InfluxDB (adresse 127.0.0.1 port 8086)

Attention : Dans une base de données InfluxDB, les mesures que l'on injecte doivent être des nombres entiers (int). Il faudra donc penser à convertir en Int les caractères (Chars) récupérés sur le port série !

Une fois que la base de données InfluxDB est bien reliée aux données capteurs, vous pouvez à présent la relier à l'interface graphique (dashboard) de Grafana.

- ✓ Lancer le **serveur Grafana** : double-cliquer sur l'exécutable « `grafana-server.exe` » sur le PC. L'application sera alors accessible sur le web à l'adresse IP du PC où le serveur est installé, **port 3000** (ou sur 127.0.0.1 : 3000 depuis votre PC direct).



User : **admin**
Password : **admin**

✓ Créer pas à pas le dashboard :

➤ **Create your first data source:**

Bien spécifier le type (InfluxDB), les paramètres HTTP (<http://localhost:8086>, avec proxy), choisir une authentification basique, puis indiquer la base de données avec (éventuellement) l'utilisateur et le mot de passe.

Le test doit fonctionner pour valider la source de données influxDB.

➤ **Create your first dashboard :**

Choisir le type de représentation du tableau de bord (graphe, statistiques, tableau, cartographie avec graduation de couleurs, etc.), puis cliquer sur *Panel Title* → *Edit* pour configurer le dashboard.

Bien spécifier :

- la source de données (associée à une database)
- le nom de la mesure (une *database* peut contenir plusieurs *measurements*)
- pour la valeur, laisser à `mean()`. A noter que l'on peut appliquer une opération mathématique (exemple : diviser par 100), ce qui peut être utile notamment lorsque vous récupérez des valeurs « arrondies » à 2 chiffres après la virgule qui ont été multipliées par 100
- *time \$ _interval* : étant des séries temporelles, il faut spécifier l'intervalle de lecture des mesures (1s, 10s, etc.)
- *Add Query* : pour ajouter une autre courbe sur le graphe

