



# Les objets connectés

## (Partie 3 : MQTT)



LE CREN Anthony

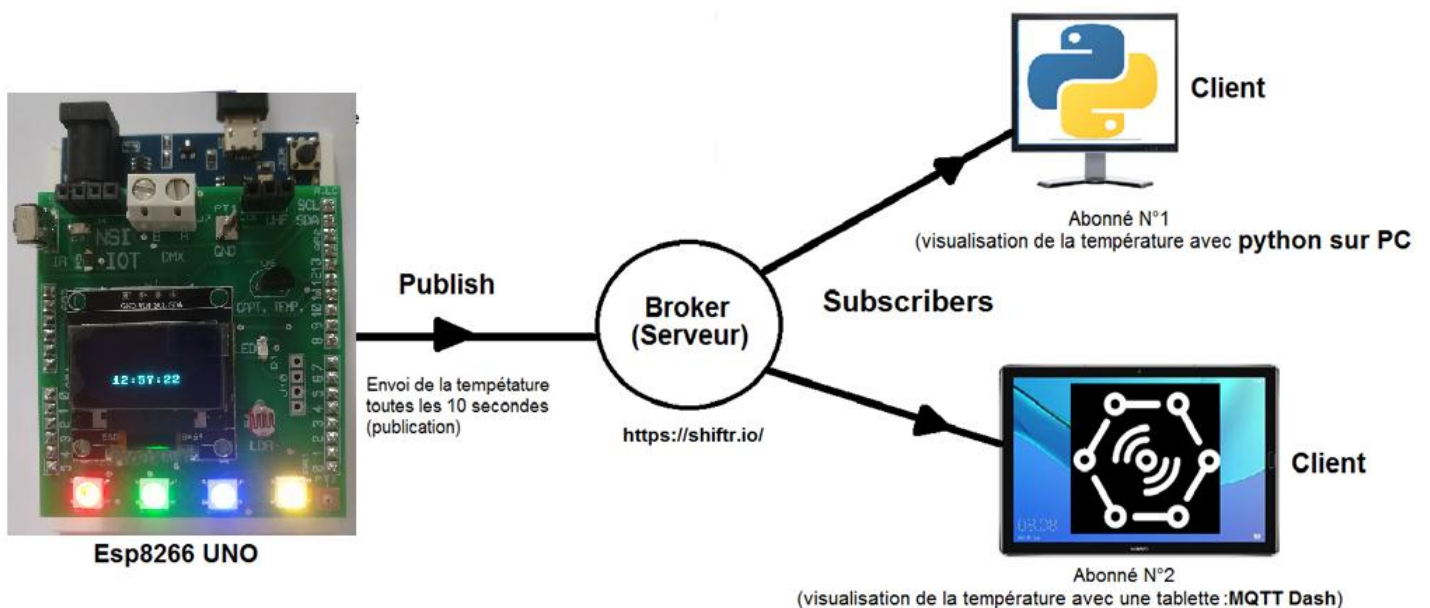
Les logins et les mots de passe décrits dans ce document sont donnés comme exemple.

### 1 Qu'est-ce que MQTT ?

MQTT (Message Queuing Telemetry Transport) est une messagerie publish-subscribe basé sur le protocole TCP/IP. MQTT est utilisé pour les IOT (Internet of Things / objets connectés). Il est conçu comme un transport de messagerie de publication / abonnement extrêmement léger en termes de ressources.

Mise en situation :

Vous avez réalisé un système à base d'ESP8266 permettant de mesurer la température d'une pièce. Vous voulez connaître cette température quand vous êtes à l'extérieur de votre maison. A première vue, une solution serait de concevoir une page WEB afin de pouvoir y accéder depuis un navigateur. MQTT va vous permettre de remplir cet objectif plus rapidement en utilisant une bande passante très réduite. Il est aussi possible de déclencher un mécanisme à distance comme une des volets roulants etc... La communication peut ainsi être bidirectionnelle.



Communication en 2 étapes :

- Un objet connecté va publier (**publish**) son message vers le Broker (Serveur)
- Pour recevoir des messages le client va souscrire (**subscribe**) leurs réceptions auprès du Broker.

Pour que les messages ne se mélangent pas, ils sont publiés sur une chaîne (topic). Par exemple /capteurs/temperature. Voir la [vidéo](#) de présentation.

# 1 Publication de température

**Q1** A l'aide de l'annexe 1, **configurer** votre broker sur le site <https://www.shiftr.io/cloud/>

Noter l'adresse, le login et le mot de passe utilisé lors de l'enregistrement dans le tableau ci-dessous :

Adresse du serveur (URL)	touchard.cloud.shiftr.io
Client id (myMqttClient)	Python ou esp8266 ou arduino
Login (USER_ID)	touchard
Mot de passe (MQTT_API_KEY)	MFmD747BIp8YIJYI
Topic ou token	capteurs

**Q2** Vérifier la connexion sur le point d'accès internet en utilisant le programme

« **wifi\_connexion.py** »

**Configurer et tester** le programme ci-dessous avec vos propres identifiants conformément à la question **Q1**. (Q2-mqtt-publish.py)

```
from umqtt.robust import MQTTClient
import network
import sys
import time
from time import sleep_ms
from machine import Pin
from onewire import OneWire
from ds18x20 import DS18X20

sta_if = network.WLAN(network.STA_IF)
print(sta_if.active())
print(sta_if.ifconfig())

myMqttClient = b"micropython"

URL = b"touchard.cloud.shiftr.io"
USER_ID = b'touchard'
MQTT_API_KEY = b'MFmD747BIp8YIJYI'
client = MQTTClient(client_id=myMqttClient,
                    server=URL,
                    user=USER_ID,
                    password=MQTT_API_KEY,
                    ssl=False)

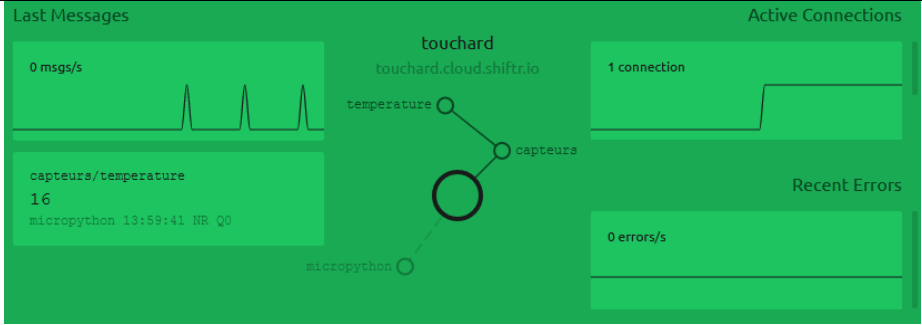
try:
    client.connect()
    print("connection ok");
except Exception as e:
    print('could not connect to MQTT server {}'.format(type(e).__name__, e))
    sys.exit()

bus = OneWire(Pin(12))
ds = DS18X20(bus)
capteur_temperature = ds.scan()

PUBLISH_PERIOD_IN_SEC = 10

while True:
    try:
        ds.convert_temp()
        sleep_ms( 750 )
        temp_celsius = ds.read_temp(capteur_temperature[0])
        print("Température : ",temp_celsius )
        client.publish("/capteurs/temperature", str(int(temp_celsius)))
        print("publish ok");
        time.sleep(PUBLISH_PERIOD_IN_SEC)
    except KeyboardInterrupt:
        print('Ctrl-C pressed...exiting')
        client.disconnect()
        sys.exit()
        print("exit")
```

**Q3 Vérifier** l'envoi de données au broker. (<https://touchard.cloud.shiftr.io/>)



The screenshot shows the Shiftr.io dashboard for the user 'touchard'. It includes a 'Last Messages' section with a message graph and details for a message on the 'capteurs/temperature' topic. The 'Active Connections' section shows 1 connection. The 'Recent Errors' section shows 0 errors. A central message graph shows a flow from 'micropython' to 'capteurs' to 'temperature'.

```
True

('192.168.1.104',
'255.255.255.0',
'192.168.1.1',
'192.168.1.1')

connection ok

Température : 16.25

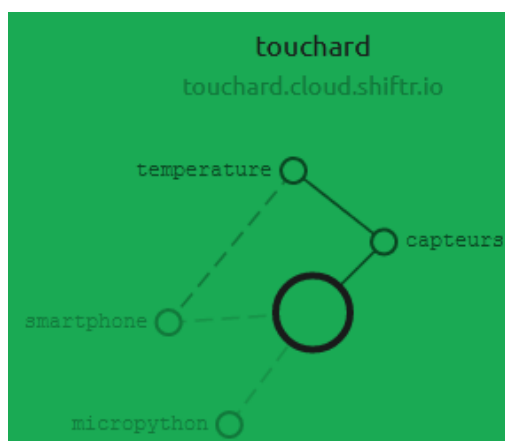
publish ok
```

**Q4** A l'aide de l'annexe 2,3 ou 4, **configurer** le client MQTT sur un smartphone. Préférence pour **MQTT panel** : annexe 3)

Noter à nouveau la clé et le mot de passe en lien avec le broker (**Q1**) dans le tableau ci-dessous :

Adresse du serveur	touchard.cloud.shiftr.io
Client id	smartphone
login	touchard
Mot de passe	MFmD747Blp8YIJYI
Topic ou token	capteurs

**Q5 Vérifier** l'envoi de données du broker vers le smartphone.



**Q6 Identifier** le subscriber et le publisher en complétant le tableau ci-dessous par oui ou non

	Subscriber	Publisher
L'esp8266 est le	Non	Oui
Le smartphone est le	Oui	Non

La qualité de service (QoS) ou quality of service (QoS) est la capacité à véhiculer dans de bonnes conditions un type de trafic donné.

<p>Other settings</p> <p><input checked="" type="radio"/> QoS(0)</p> <p><input type="radio"/> QoS(1)</p> <p><input type="radio"/> QoS(2)</p>	<ul style="list-style-type: none"> <li>- QoS0. Le message envoyé n'est pas stocké par le Broker. Il n'y a pas d'accusé de réception. Le message sera perdu en cas d'arrêt du serveur ou du client. C'est le mode par défaut</li> <li>- QoS1. Le message sera livré au moins une fois. Le client renvoie le message jusqu'à ce que le broker envoie en retour un accusé de réception.</li> <li>- QoS2. Le broker sauvegarde le message et le transmettra jusqu'à ce qu'il ait été réceptionné par tous les souscripteurs connectés</li> </ul>
--	--

## 2 Commande de la Led de l'IOT avec le smartphone

**Q7 Configurer** et tester le programme ci-dessous avec vos propres identifiants conformément à la question **Q1** (Q7-mqtt-subscribe-led.py)

```
from umqtt.robust import MQTTClient
import network
import sys
from machine import Pin

led = Pin(0, Pin.OUT)

def cb(topic, msg):
    print((topic, msg))
    valeur=int(msg.decode("ascii"))
    if valeur==1:
        led.on()
    elif valeur==0:
        led.off()

sta_if = network.WLAN(network.STA_IF)
print(sta_if.active())
print(sta_if.ifconfig())

myMqttClient = b"micropython"

URL = b"touchard.cloud.shiftr.io"
USER_ID = b'touchard'
MQTT_API_KEY = b'MFmD747Bip8YIJYI'
client = MQTTClient(client_id=myMqttClient,
                    server=URL,
                    user=USER_ID,
                    password=MQTT_API_KEY,
                    ssl=False)

client.set_callback(cb)

try:
    client.connect()
    print("connection ok");
    client.subscribe("/actionneurs/led")
except Exception as e:
    print('could not connect to MQTT server {}'.format(type(e).__name__, e))
    sys.exit()

while True:
    try:
        client.check_msg()
    except KeyboardInterrupt:
        print('Ctrl-C pressed...exiting')
        client.disconnect()
        sys.exit()
    print("exit")
```

**Q8** Ajouter un panel appelé « **Switch** » de commande comme le montre l'exemple ci-dessous, puis valider la commande de la led sur le site shiftr.io

Free 55% 19:32

[← Edit panel](#)

Panel name \*  
led

Topic \*  
/actionneurs/led

Subscribe Topic ?

Payload on \*  
1

Payload off \*  
0

☒ Use icon switch

☐ Choose on icon

☒ Choose off icon

Icon color  
#27933c

Icon color  
#c21818

touchard

touchard.cloud.shiftr.io

**Q9** Identifier le subscriber et le publisher en complétant le tableau ci-dessous par oui ou non

	Subscriber	Publisher
L'esp8266 est le	Oui	Non
Le smartphone est le	Non	Oui

### 3 Utilisation de MQTT avec un IDE Python (en remplacement du smartphone)

Prérequis : Installer `paho-mqtt` sur votre PC.

`paho-mqtt` 1.4.0

**Q10** Reprendre le programme à la question **Q2 dans l'esp8266**, puis tester le programme suivant dans **un IDE python sur PC** (Q10-mqtt\_subscribe\_PC.py)

```
import paho.mqtt.client as mqtt

# The callback for when a PUBLISH message is received from the server.
def on_message(client, userdata, msg):
    print(msg.topic+" "+str(msg.payload))

client = mqtt.Client("python") #id must be unique
client.username_pw_set("touchard", "MFmD747B1p8YIJYI") #login, password
client.on_message = on_message

client.connect("touchard.cloud.shiftr.io", 1883, 60)

client.subscribe("/capteurs/temperature")

while True:
    client.loop()
```

```
/capteurs/temperature b'20'
/capteurs/temperature b'20'
/capteurs/temperature b'20'
```

L'information de température s'affiche sur le PC et non sur le smartphone

**Q11** Identifier le subscriber et le publisher en complétant le tableau ci-dessous par oui ou non

	Subscriber	Publisher
L'esp8266 est le	Non	Oui
Le PC est le	Oui	Non

**Q12** Reprendre le programme à la question **Q7 dans l'esp8266**, puis tester le programme suivant dans un **IDE python sur PC**. (Q12-mqtt\_publish\_PC.py)

```
import paho.mqtt.client as mqtt

client = mqtt.Client("python") #id must be unique
client.username_pw_set("touchard", "MFmD747Bip8YIJYI") #login, password

client.connect("touchard.cloud.shiftr.io", 1883, 60)

while True:
    #client.loop()
    valeur=input("saisir l'etat de la led 0/1 ?")
    client.publish("/actionneurs/led", valeur)
```

saisir l'etat de la led 0/1 ?1

saisir l'etat de la led 0/1 ?0

La Led câblée sur l'ESP8266 doit s'allumer ou s'éteindre en fonction de la commande 0 ou 1

**Q13** Identifier le subscriber et le publier en complétant le tableau ci-dessous par oui ou non

	Subscriber	Publisher
L'esp8266 est le	Oui	Non
Le PC est le	Non	Oui

## Mini projets

**Q14** Utiliser le smartphone pour commander le ruban de Led avec des couleurs préprogrammées.

**Q15** Le smartphone envoie un texte sur l'afficheur OLED.



**Q16** Le smartphone reçoit une alerte d'intrusion détectée par le capteur de lumière.

**Q17** Votre projet personnel (le programme de votre choix)

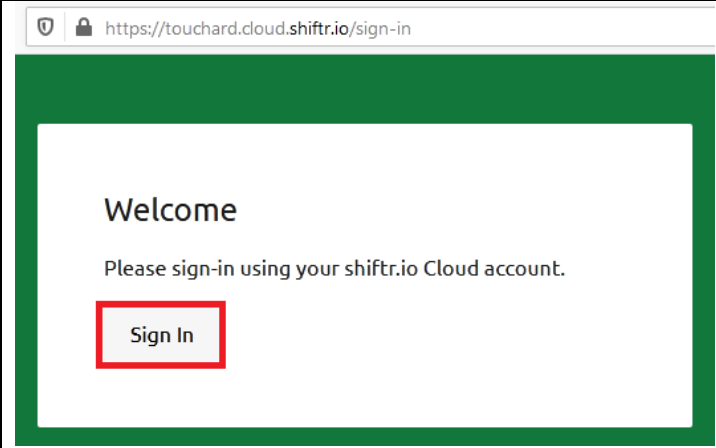
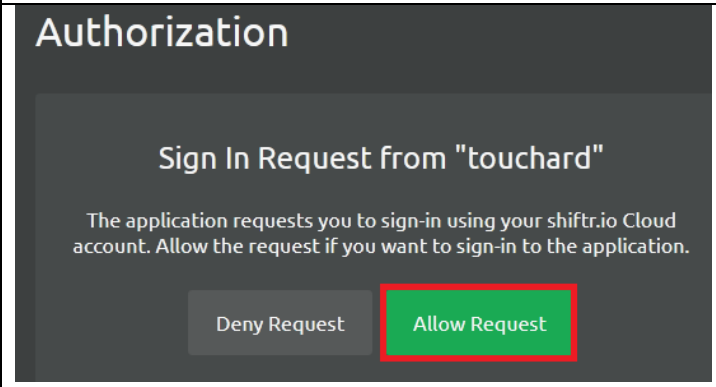
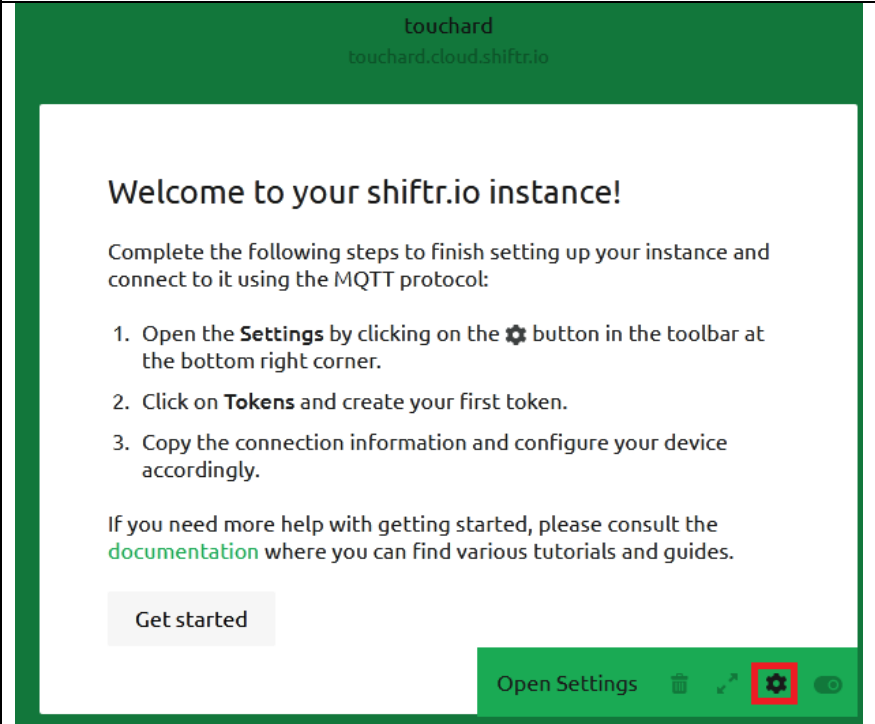


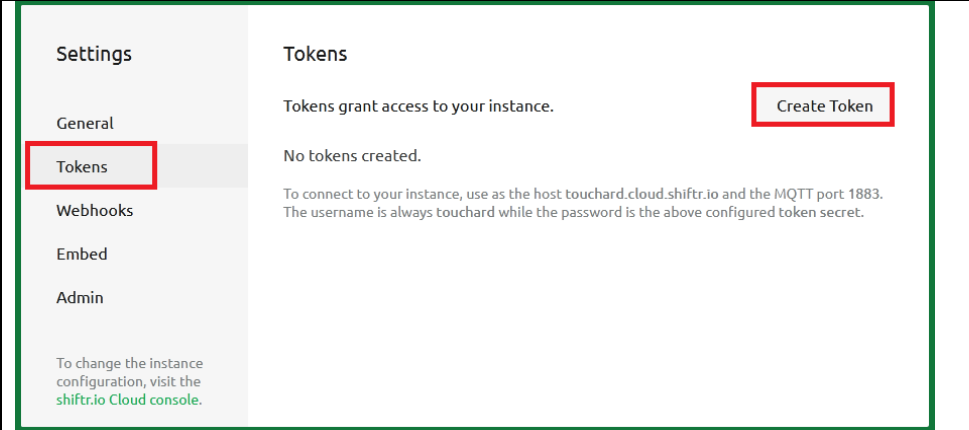
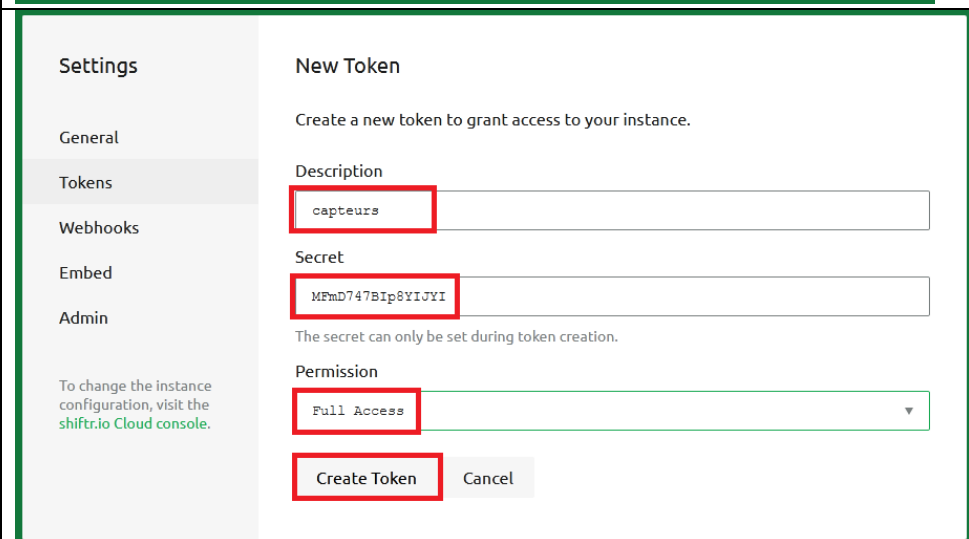
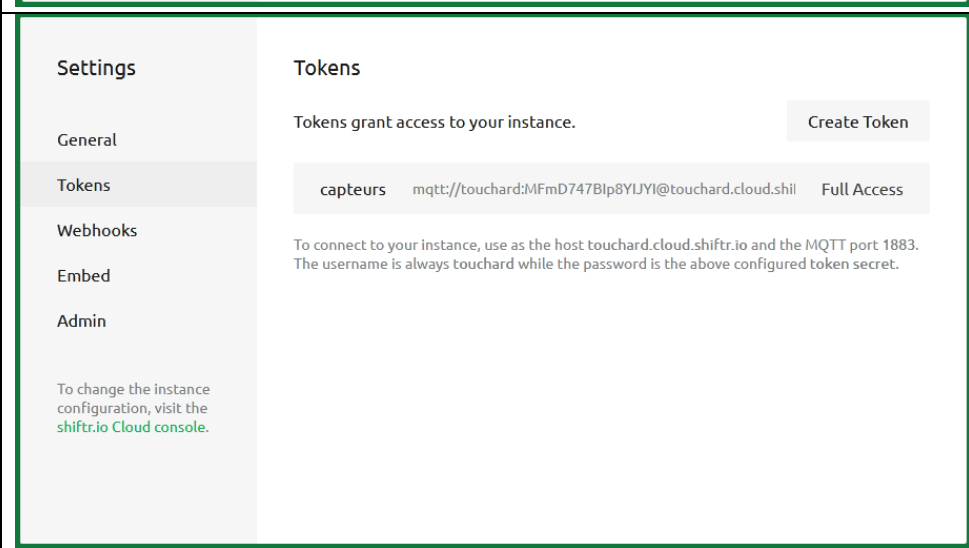
## Annexe 1 : Configuration d'un broker (serveur MQTT) en ligne

 <p>The screenshot shows the shiftr.io Cloud login page. It features a green arrow logo and the text 'shiftr.io Cloud'. Below the logo, it says 'Welcome to shiftr.io Cloud.' There are two green buttons: 'Sign In' and 'Sign Up'. The 'Sign Up' button is highlighted with a red border.</p>	<p>Créer un compte sur le site  <a href="https://www.shiftr.io/cloud/">https://www.shiftr.io/cloud/</a></p> <p>Cliquer sur Sign Up</p>
 <p>The screenshot shows the shiftr.io Cloud sign-up page. It features the green arrow logo and the text 'shiftr.io Cloud'. Below the logo, it says 'Please enter your name and email address to sign up for a shiftr.io Cloud account.' There are two input fields: one for the name 'nsi.touchard' and one for the email 'nsi.touchard@gmail.com'. Below the email field is a green checkmark and the text 'Je suis humain' next to an hCaptcha logo. At the bottom is a green 'Sign Up' button.</p>	<p>Indiquer un login et un compte mail</p> <p>Puis cliquer sur Sign up</p>
 <p>The screenshot shows the shiftr.io Cloud confirmation page. It features the green arrow logo and the text 'shiftr.io Cloud'. Below the logo, it says 'If no account with the email nsi.touchard@gmail.com exists yet, we have sent instructions to complete your sign-up.' and 'If there is an account registered with this email, we have not sent an email.' At the bottom, there is a blue envelope icon and the text 'nsi.touchard@gmail.com a reçu 1 nouveau message' and 'Your shiftr.io Cloud Sign Up'.</p>	<p>Confirmer le compte à partir de votre adresse mail</p>
 <p>The screenshot shows the shiftr.io Cloud success page. It features the green arrow logo and the text 'shiftr.io Cloud'. Below the logo, it says 'You're account has been successfully created. You can sign in now.' There is a green 'Sign In' button. At the bottom, there is a blue envelope icon and the text 'nsi.touchard@gmail.com a reçu 1 nouveau message' and 'Welcome to shiftr.io Cloud'.</p>	<p>Terminer la création du compte et afficher le  « Dashboard »</p> <p>Cliquer ensuite sur « Deploy Instance »</p>

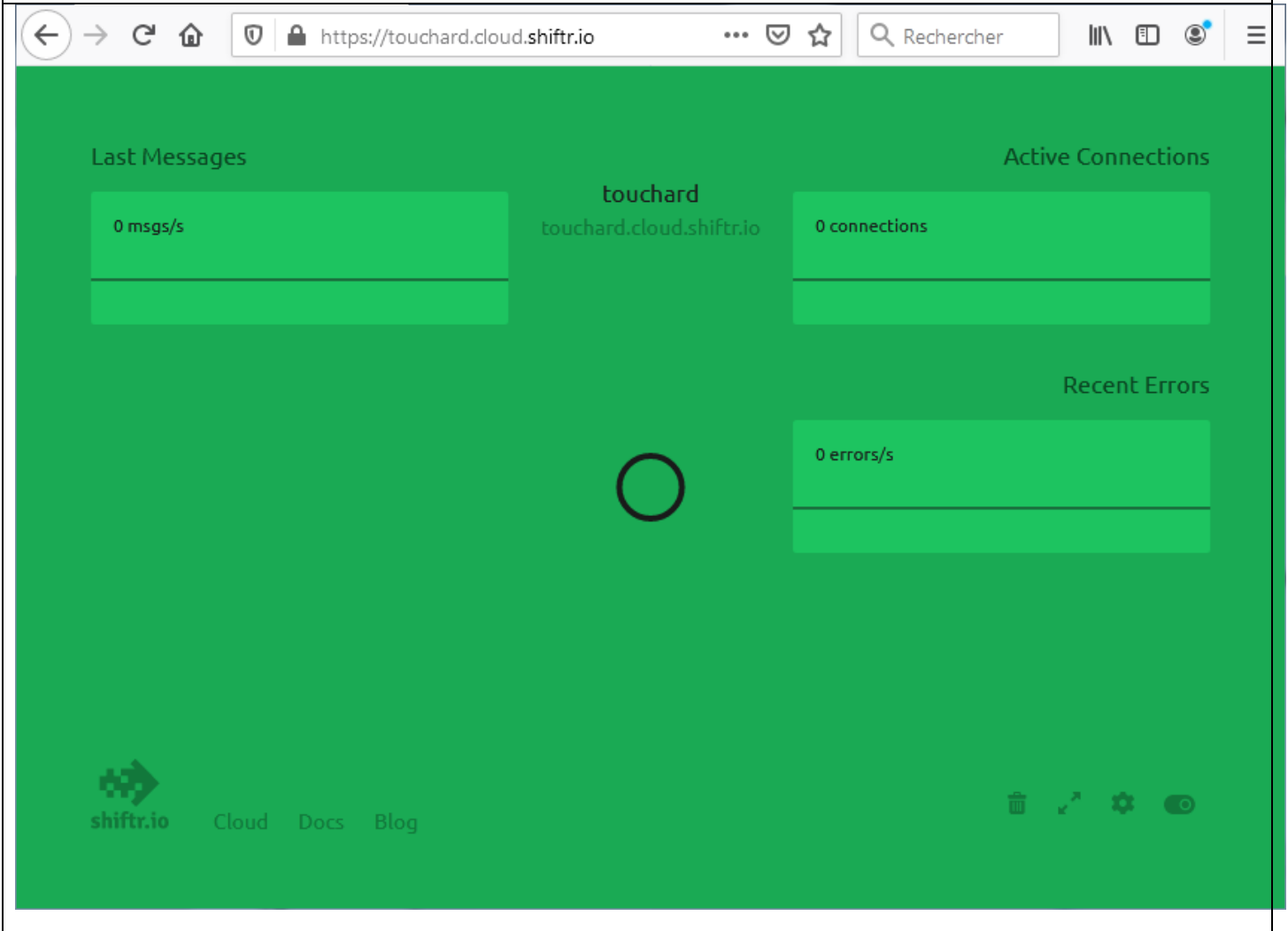
<div><div></div><div>shiftr.io Cloud</div></div> <div><p>Welcome nsi.touchard. Please choose a password and create your shiftr.io Cloud account.</p></div> <div><div><input type="text" value="nsi.touchard@gmail.com"/></div><div><input type="password" value="••••••••"/></div><div>Create Account</div></div>	<div><div></div><div>shiftr.io Cloud</div></div> <div><div>DeployInstancesSupport</div><div>N</div></div> <div><div>N nsi.touchard</div><div>Dashboard</div><div>Manage Groups</div><div>Account Settings</div><div>Sign Out</div></div> <div><div>Dashboard</div><div>Welcome to shiftr.io Cloud!</div><div>With shiftr.io Cloud you can launch and manage multiple shiftr.io instances. With the Basic plan you can get connected for free today.</div><div>Deploy Instance</div></div>	
<div><div>Setup</div><div><div>You need to create a group first.</div><div>Instances are managed in groups to consolidate billing and soon share them with other shiftr.io Cloud users.</div><div><input type="text" value="nsi"/></div><div>Create Group</div></div></div>	<div>Créer un groupe</div>	
<div><div>Setup</div><div><div>The group "nsi" has been created.</div><div>You can now continue with deploying an instance.</div><div>Deploy Instance</div><div>To change the group settings you can go to "Manage Groups" from your account drop down in the top right corner.</div></div></div>	<div>Cliquer sur « Deploy Instance »</div>	

<p><b>Plan</b></p> <p>Choose a plan depending on your projected usage.</p> <div> <div> <p><b>Basic</b></p> <p>Start simple with the Basic plan which is ideal for testing the platform and driving small projects on demand.</p> <p><b>\$0* / month</b></p> <p>100 Active connections 5K Messages per second</p> </div> <div> <p><b>Plus</b></p> <p>Do you want to run your project 24/7 or require more power on demand? Choose the Plus plan and kickstart your project.</p> <p><b>\$7 / month</b></p> <p>200 Active connections 10K Messages per second</p> </div> <div> <p><b>Pro</b></p> <p>Do you have a lot of devices or want to send many messages? Go with the Pro plan to unlock more throughput and bandwidth.</p> <p><b>\$19 / month</b></p> <p>500 Active connections 30K Messages per second</p> </div> </div> <p><small>* Free instances are only allowed to run 6 hours a day. They will automatically sleep and recharge if not used.</small></p> <p><b>Settings</b></p> <p>Choose a name for your instance. Adjust the unique domain name of your instance.</p> <div> <input type="text" value="touchard"/> <input type="text" value="touchard"/> <input type="text" value=".cloud.shiftr.io"/> </div> <p><small>This name is used to refer to your instance throughout the system. The unique domain name cannot be changed later.</small></p> <p><b>Billing</b></p> <p>Choose the billable group for this instance.</p> <div> <input type="text" value="nsi"/> </div> <p><b>Summary</b></p> <p>Review the configuration and deploy your instance.</p> <table border="1"> <tr> <td>Plan</td> <td>Basic</td> <td>\$0.00</td> </tr> <tr> <td>Total per month</td> <td></td> <td>\$0.00</td> </tr> </table> <p><b>Deploy Instance</b></p>	Plan	Basic	\$0.00	Total per month		\$0.00	<p>Choisir la version basic « Gratuite »</p> <p>Puis donner un nom pour l'instance</p> <p>Ainsi que le nom de groupe crée précédemment.</p> <p>Puis cliquer sur « deploy Instance »</p>
Plan	Basic	\$0.00					
Total per month		\$0.00					
<p><b>Instances</b></p> <p>nsi</p> <div> <p>touchard</p> <p><b>touchard.cloud.shiftr.io</b></p> <p>Running n/a Basic</p> </div>	<p>Cliquer sur l'instance (lien en couleur verte)</p>						

	<p>Lier l'instance avec votre compte Crée au tout début</p>
	<p>Accepter l'autorisation</p>
	<p>Cliquer sur l'icône des paramètres</p>

	Créer un « token » ou topic.
	Renseigner la description du nouveau Token.  Par exemple « capteurs »
	La création est maintenant terminée

Aller sur la page accueil de l'instance <https://touchard.cloud.shiftr.io/>




La configuration du Broker (Serveur MQTT) est terminée.

Exemple de configuration

Adresse du serveur	<a href="https://touchard.cloud.shiftr.io">touchard.cloud.shiftr.io</a>
Client id	python ou micropython ou arduino
login	<a href="#">touchard</a>
Mot de passe	<a href="#">MFmD747Blp8YIJYI</a>
Topic ou token	<a href="#">capteurs</a>

Remarque :

Il est possible d'utiliser le [broker](#) shiftr.io hors ligne et pouvant être utilisé sur un réseau local sans avoir de connexion Internet. (Installation du programme  [shiftr-io-desktop.exe](#) )

Il existe d'autres brokers gratuits comme <https://www.maqiatto.com/>, mais il ne dispose pas comme shiftr.io d'une interface graphique indiquant les connexions en cours.

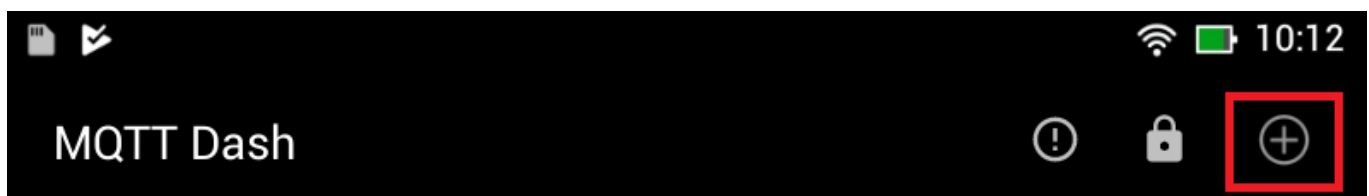
## Annexe 2 : Configuration du client sur Android (MQTT Dash)

Installer le programme MQTT dash

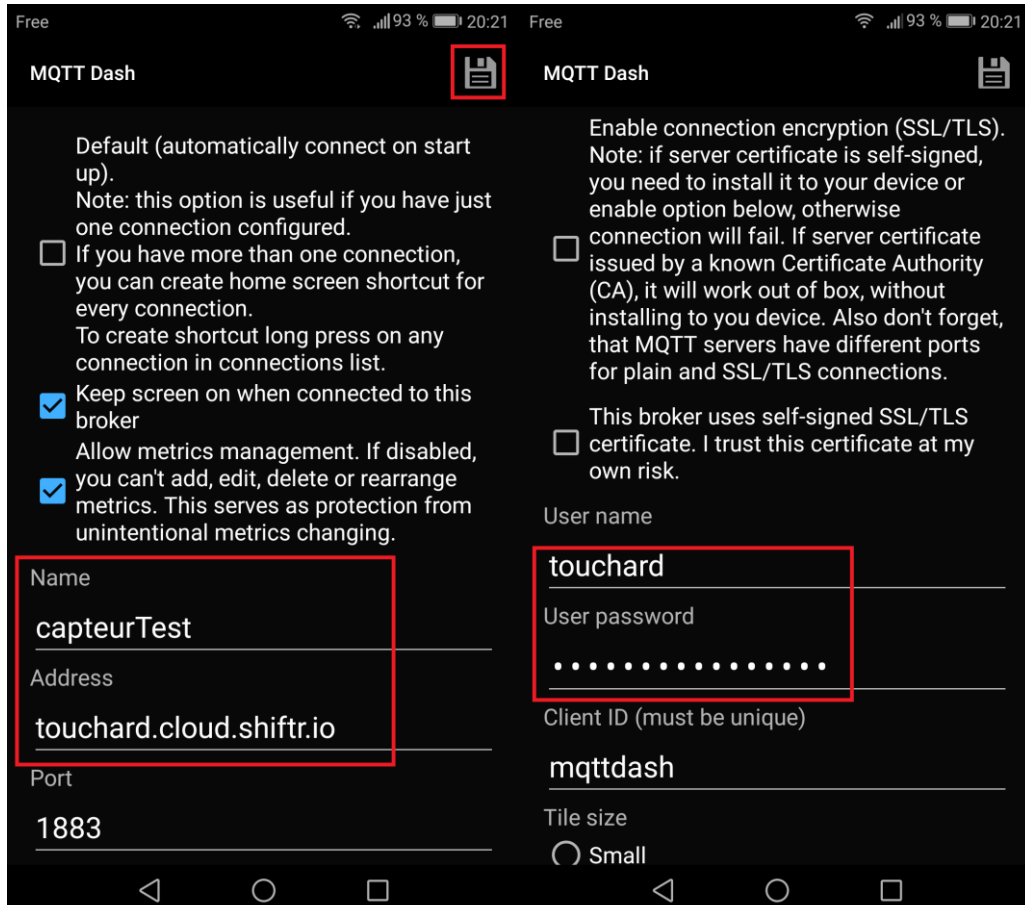


<https://play.google.com/store/apps/details?id=net.routix.mqttdash>

Configuration :



Ajouter une connexion vers le broker shiftr.io



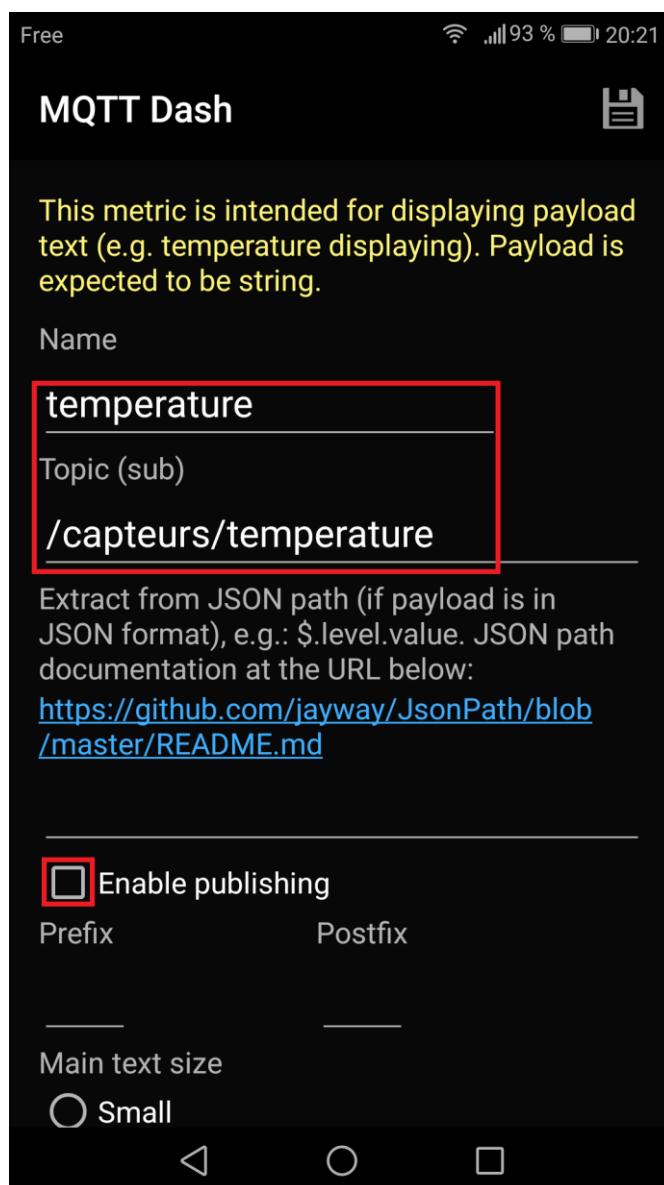
Cliquer sur sensorsTest



Puis ajouter le(s) topics nécessaires.



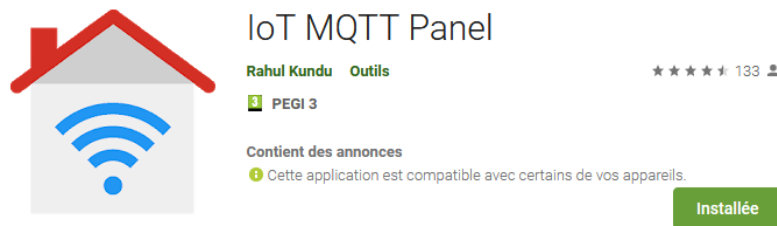
Exemple pour le Topic température (Ne pas oublier de **décocher** « Enable publishing »)






## Annexe 3 : Configuration du client sur Android (MQTT panel)

Installer le programme IoT MQTT Panel

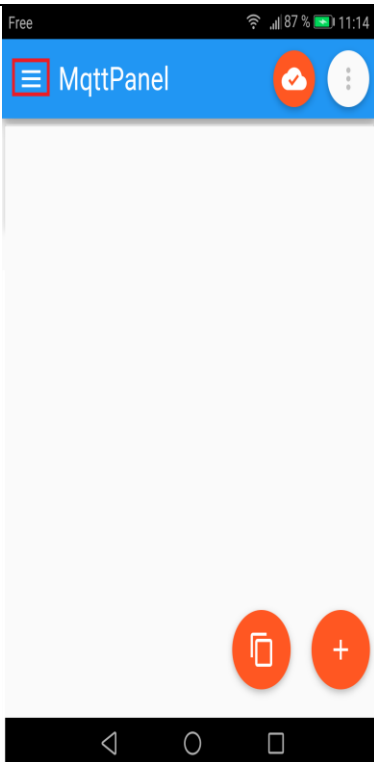


<https://play.google.com/store/apps/details?id=snr.lab.iotmqttpanel.prod>

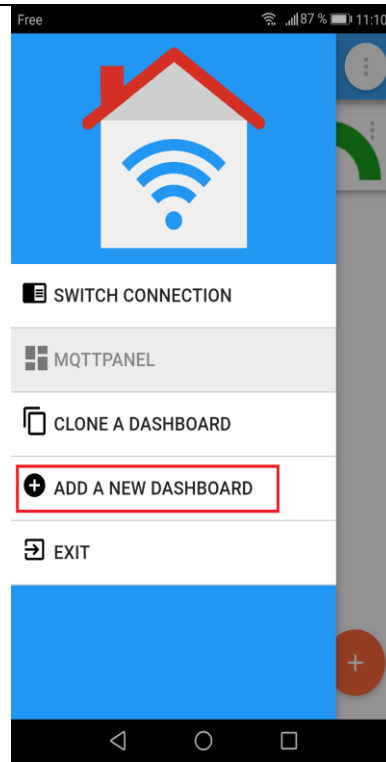
Ce programme un plus convivial que MQTT dash et possède des widgets. (Élément de base de l'interface graphique d'un logiciel : fenêtre, barre d'outils, par exemple). Une fois installé, cliquer sur ajouter une connexion icône , puis configurer le broker shifttr.io

Configuration de la connexion au Broker	Sélectionner MqttPanel

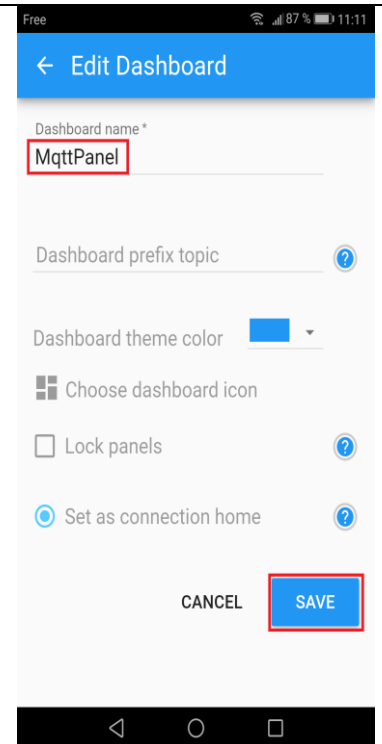
## Configuration de la connexion au Broker



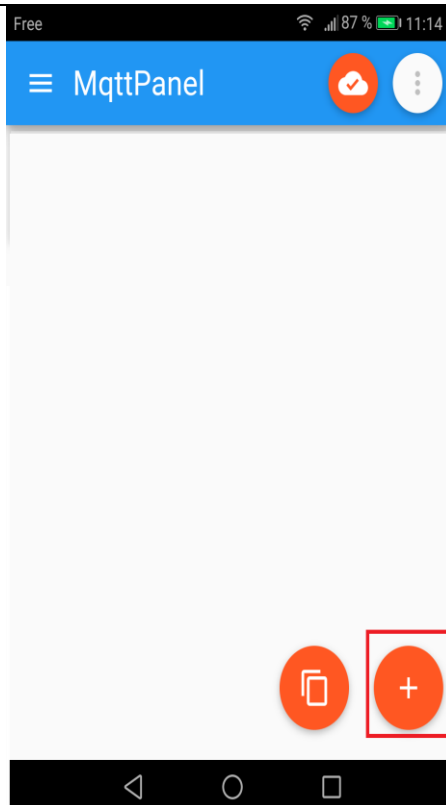
## Ajouter un nouveau Dashboard



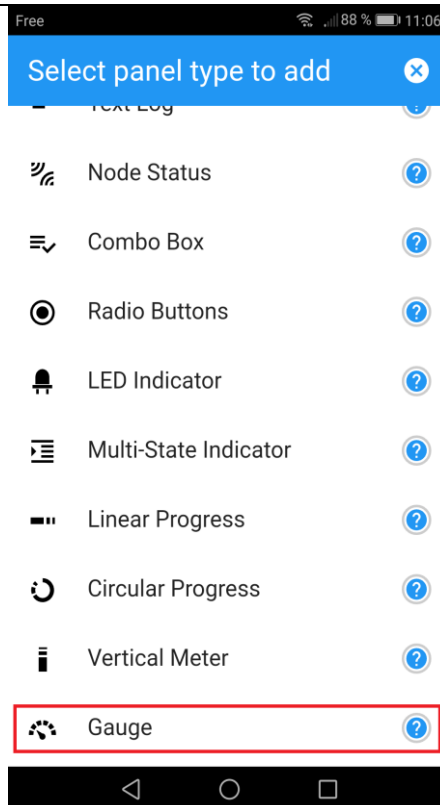
## Nommer le Dashboard



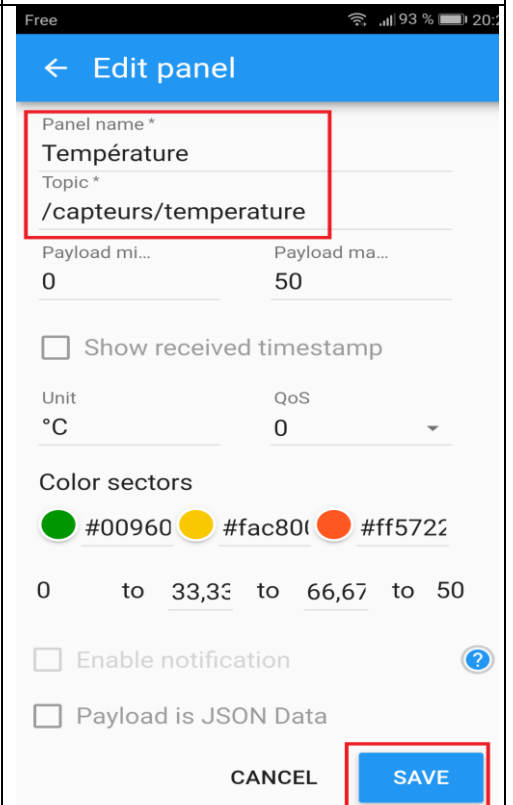
## Ajouter un panel en cliquant sur l'icône rouge +



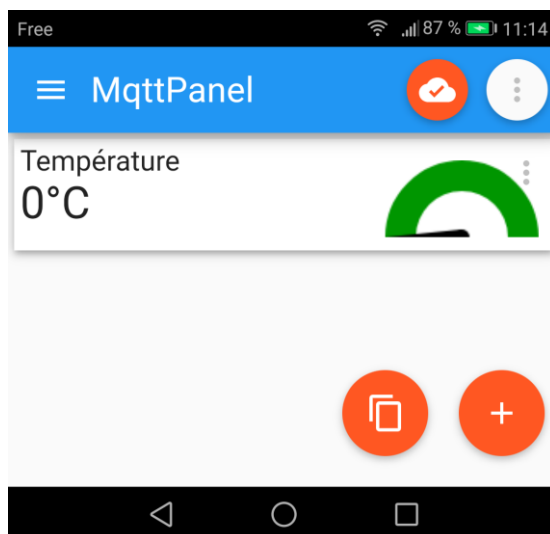
## Sélectionner Gauge



## Configurer le topic



La configuration du logiciel sur le smartphone est terminée. L'utilisateur est prêt à recevoir l'information de température en provenance de la carte IOT.



Exemple de configuration en lien avec le broker :

Name	MqttPanel
Adresse du serveur	<a href="https://touchard.cloud.shiftr.io">touchard.cloud.shiftr.io</a>
Client id	Smartphone
User name (key)	<a href="#">touchard</a>
User password (password)	<a href="#">MFmD747BIp8YIJYI</a>
Topic name	<a href="#">capteurs/temperature</a>

Port 1883 : communication non sécurisée (données en clair sur le réseau) par défaut dans ce document.

Port 8881 : communication non sécurisé SSL (Secure Socket Layer) / TLS (Transport Layer Security)

## Annexe 4 : Configuration du client sur iPhone (EasyMQTT)

Installer le programme EasyMQTT sur le téléphone



**EasyMQTT**  
Luca Kaufmann  
Conçu pour iPad  
★★★★★ 5,0 • 1 note  
Gratuit - Inclut des achats intégrés

<https://apps.apple.com/fr/app/easymqtt/id1523099606>

Ajouter une connexion vers le broker shiftr.io, puis tester la connexion

