

2 ème Année
Cycle supérieur-SIQ

Systèmes Communicants et Intelligents
Rapport

Système IoT complet avec utilisation d'une plateforme IoT

Réalisé par :

- BOUROUINA Anfal
- SIQ-3-Binome-05

Encadré par :

- Mr. Abdennour SEHAD

2021-2022

1. Partie théorique

1. Introduction aux différentes plateformes IoT

Le concept d'Internet des objets implique la création d'un réseau distribué composé de nombreux objets physiques équipés de logiciels intégrés, de capteurs et d'options de connectivité qui collectent et partagent des données entre eux et avec la plateforme centrale via un Internet (ou un serveur local).

Les capteurs et les actionneurs collectent des données directement à partir d'objets physiques (dispositifs, équipements, machines, véhicules, appareils ménagers, personnes, animaux, etc.). Les passerelles et les systèmes d'acquisition de données convertissent les données recueillies du format analogique au format numérique , ils les envoient ensuite au centre de traitement via des plateformes IoT.

Une plateforme IoT est un ensemble de composants qui permet aux développeurs de déployer les applications, de collecter des données à distance, de sécuriser la connectivité et d'exécuter la gestion des capteurs. Une plateforme IoT gère la connectivité des appareils et permet aux développeurs de créer de nouvelles applications logicielles mobiles.

Types de plateformes de l'internet des objets

- Matériel : fournissent des cartes de développement physique pour créer des dispositifs IoT, notamment des microcontrôleurs, des microprocesseurs, des systèmes sur puce (SoC), des systèmes sur module (SoM).
- Logiciel : servent d'environnement de développement intégré (IDE) avec des outils et des fonctionnalités pour coder des applications.
- Technologies de communication : fournissent des technologies de communication pour connecter les objets physiques au centre de données (local ou dans le cloud) et transmettre des informations entre eux. Parmi les protocoles et les normes de connectivité les plus populaires pour l'Internet des objets, citons MQTT, DDS, AMQP, Bluetooth, ZigBee, WiFi, Cellulaire, LoRaWAN, etc.
- Référentiel central (cloud ou local)
- Applications pour l'utilisateur final : couvrent tous les aspects des produits IoT, du développement et de la connectivité à la gestion et à la visualisation des données (Blynk par exemple).

Parmi les plateformes IoT les plus populaires, on compte :

- Blynk, Google Cloud IoT, Azure IoT Hub, Cisco IoT connect, AWS IoT, Watson IBM IoT ...

2. Étude théorique de la plateforme à utiliser

Pour le présent TP , la plateforme IoT à utiliser est **Blynk**.

Blynk est une plateforme IoT pour les smartphones iOS ou Android qui est utilisée pour contrôler Arduino, Raspberry Pi ,NodeMCU et d'autres via Internet. Cette application permet

de créer une interface graphique ou une interface homme-machine (IHM) en compilant et en fournissant l'adresse appropriée sur les widgets disponibles.

Blynk a été conçu pour l'Internet des objets. Il peut contrôler le matériel à distance, il peut afficher les données des capteurs, il peut stocker des données, les visualiser et faire beaucoup d'autres choses cool.

La plateforme comprend trois composants principaux:

Blynk App : - Elle vous permet de créer des interfaces étonnantes pour vos projets en utilisant divers widgets qui sont fournis.

Le serveur Blynk : - Il est responsable de toutes les communications entre le smartphone et le matériel. Vous pouvez utiliser le Blynk Cloud ou exécuter votre propre serveur Blynk localement. Il est open-source, peut facilement gérer des milliers d'appareils et peut même être lancé sur un Raspberry Pi.

Bibliothèques Blynk : - Elles permettent la communication, pour toutes les plateformes matérielles courantes, avec le serveur et traitent toutes les commandes entrantes et sortantes.

Le processus qui se produit lorsque quelqu'un appuie sur le bouton dans l'application Blynk est que les données vont se déplacer vers Blynk Cloud, où les données trouvent magiquement leur chemin vers le matériel qui a été installé. Cela fonctionne dans le sens inverse et tout se passe en un clin d'œil comme le montre la figure suivante.

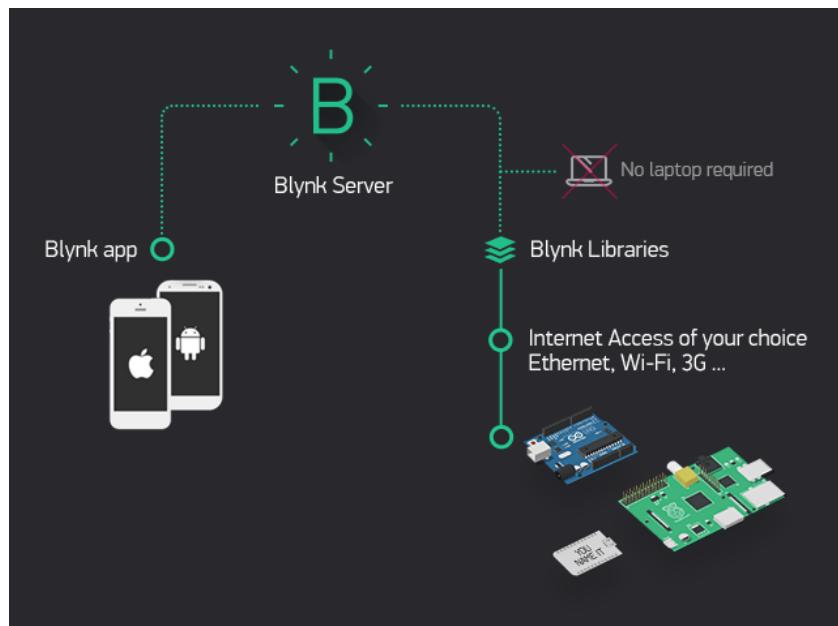


Figure 1. Architecture de communication dans Blynk.

2. Partie pratique

Dans la partie pratique, nous allons travailler sur la plateforme Blynk avec ces deux modes différents : Cloud et Local , en testant et communiquant avec une carte Arduino mkr dotée des capteurs et des actionneurs.

2.1. Installer Blynk avec serveur cloud

En mode Cloud, l'arduino communique au serveur distant de Blynk (le cloud) et s'identifier à travers un token, l'utilisateur va être capable par la suite de configurer et contrôler les composantes IoT selon le schéma suivant :

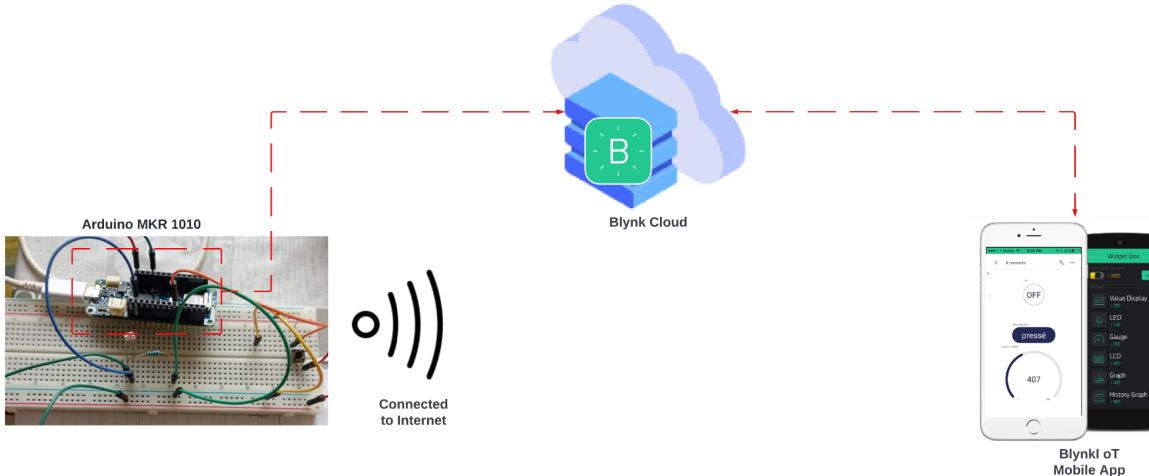
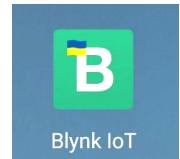


Figure 2. Flux de données dans un système IoT à base de Blynk.

2.1.1. Etapes de réalisation

Pour créer un projet sur Blynk, on doit suivre les étapes suivantes:

- Installer l'application mobile. “**Blynk IoT**”.
- S'inscrire sur l'application / ou sur Blynk Console¹.
- Installer la librairie **Blynk**² sur la carte utilisée.
- Créer un template (un schéma du projet) sur l'application qui comprend le type de carte connectée (Arduino, Raspberry...) et le type de connexion (WIFI, Bluetooth...), les senseurs utilisés. Une fois le template créé, il aura un `template_id` comme identifiant unique permettant à un projet d'être relié à ce template.
- Créer un appareil (suit un des templates déjà créés) sur l'application, on y spécifie plus de détails comme le type de l'Arduino utilisé, la version de blynk, si on utilise SSL (pour chiffrer la communication). Une fois l'appareil créé, il aura un auth-token, si une carte se connecte avec ce token, on saura que c'est cet appareil.
- On copie le template-id et auth-token sur le code qui sera uploadé sur la carte, pour pouvoir l'identifier.
- On doit aussi spécifier le SSID d'une connexion WIFI et son mot de passe si on veut connecter la carte par wifi (d'autres options comme Bluetooth sont aussi envisageables).
- On code la fonction à réaliser quand on reçoit un événement de l'application Blynk vers la carte ou dans le sens inverse.



¹ <https://ny3.blynk.cloud/dashboard/global>

² <https://www.arduino.cc/reference/en/libraries/blynk/>



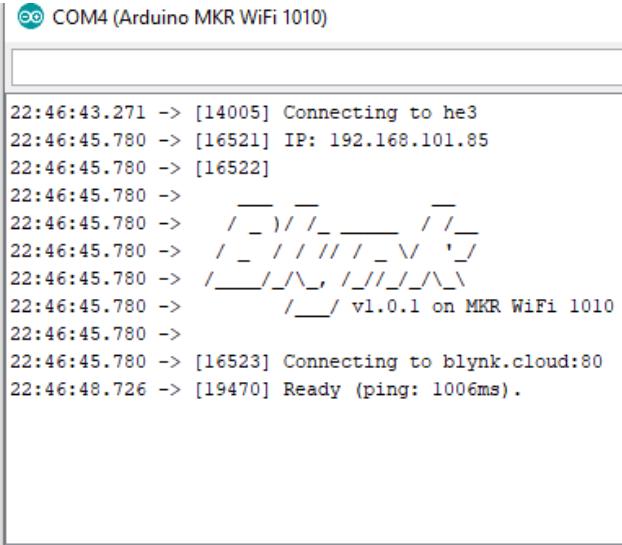
Figure 2. Informations générales d'un template.

Datastreams									
<input type="text"/> Search datastream									
Id	Name	Alias	Color	Pin	Data Type	Units	Is Raw	Min	Max
1	led	led		V0	Integer		False	0	1

Figure 2. Déclaration d'une LED dans Datastreams du template.

Dashboard	Timeline	Device Info	Metadata	Actions Log	
STATUS ● Offline		LAST UPDATED 10:07 AM Today	FIRMWARE CONFIGURATION		
LAST ONLINE 11:36 AM Today		LATEST METADATA UPDATE 10:12 AM Today by kenzamakhloifi2002@gmail.com	#define BLYNK_TEMPLATE_ID "TMPLKprz_Vn" #define BLYNK_DEVICE_NAME "secondtry" #define BLYNK_AUTH_TOKEN "P6D_GnKHavmFLF0gTof2m2_KBudAEsW-"		
DEVICE ACTIVATED 12:04 PM May 19, 2022 by kenzamakhloifi2002@gmail.com		ORGANIZATION My organization - 1210LY	Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code.		
AUHTTOKEN P6D_- **** - **** - ****		TEMPLATE NAME secondtry	Region: ny3 Privacy		

Figure 3. Informations générales d'un appareil lié au template précédent.



```

#define BLYNK_TEMPLATE_ID "TMPLKprz_Vn"

#include <SPI.h>
#include <WiFiNINA.h>
#include <BlynkSimpleWiFiNINA.h>
char auth[] = "P6D_GnKHavmflFOgTof2m2_KBudAEsW-";

char ssid[] = "he3";
char pass[] = "kenza's network";

void setup()
{
    // Debug console
    Serial.begin(9600);

    Blynk.begin(auth, ssid, pass);
}

```

COM4 (Arduino MKR WiFi 1010)

22:46:43.271 -> [14005] Connecting to he3
22:46:45.780 -> [16521] IP: 192.168.101.85
22:46:45.780 -> [16522]
22:46:45.780 ->
22:46:45.780 -> / _) / / _ _ _ _ / / _
22:46:45.780 -> / _ _ / / / / _ \ \ ' _ /
22:46:45.780 -> / _ _ / \ _ , / / / / \ _ \
22:46:45.780 -> / _ / v1.0.1 on MKR WiFi 1010
22:46:45.780 ->
22:46:45.780 -> [16523] Connecting to blynk.cloud:80
22:46:48.726 -> [19470] Ready (ping: 1006ms).

Figure 4. carte arduino connectée sur Blynk Cloud.

2.2. Configuration de deux capteurs

2.2.1. code Arduino

```

#include <WiFiNINA.h>
#include <BlynkSimpleWiFiNINA.h>

// Configuration des infos sur le template et appareil
#define BLYNK_TEMPLATE_ID "TMPLKprz_Vn"
#define BLYNK_DEVICE_NAME "secondtry"
char auth[] = "P6D_GnKHavmflFOgTof2m2_KBudAEsW-";

// Configuration de la connexion internet
char ssid[] = "he3";
char pass[] = "kenza's network";

//déclaration des valeurs lues par les capteurs
int valLuminosite;
int valBouton;
//un délai pour l'envoi des māj au serveur (évite la saturation du serveur)
BlynkTimer timer;

void myTimer() {
    //Les données à envoyer pour chaque māj
    Blynk.virtualWrite(V1, valLuminosite);
    Serial.println("valLuminosite");
    Serial.println(valLuminosite);
    Blynk.virtualWrite(V2, valBouton);
    Serial.println("valBouton");
    Serial.println(valBouton);
}

void setup(){
    pinMode(LED_BUILTIN, OUTPUT);
    pinMode(0, INPUT);
}

```

```

Serial.begin(9600);
//Connexion à internet
Blynk.begin(auth, ssid, pass);
// Intervalle d'envoi de māj chaque 1 seconde
timer.setInterval(1000L, myTimer);
}

void loop() {
// Lire la valeur de la luminosité et du bouton
valLuminosite = analogRead(A0);
valBouton = digitalRead(0);
// exécuter blynk
Blynk.run();
timer.run();
}

//écrire la valeur de la LED contrôlée par l'application
BLYNK_WRITE(V0) {
int pinValue = param.asInt();
Serial.println(pinValue);
digitalWrite(LED_BUILTIN, pinValue);
}

```

2.2.2. Branchement

On branche le capteur de lumière sur A0 et le bouton sur D0. C'est les pins qui sont configurés par la bibliothèque blynk pour être lus et leurs valeurs envoyées chaque seconde. Les figures qui suivent illustrent le branchement réel et sa schématisation.

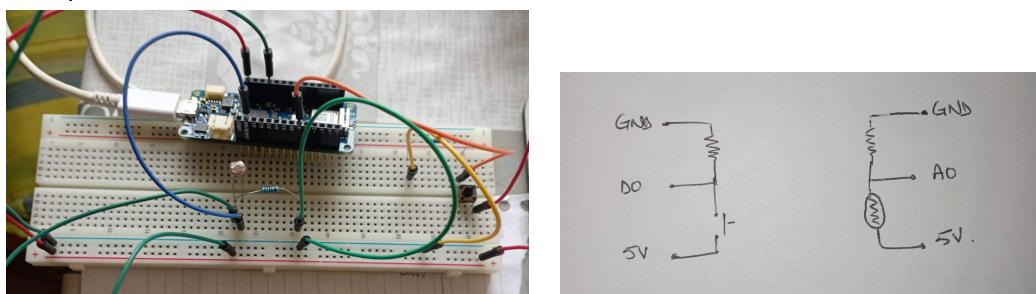


Figure 5. Branchement.

2.2.3. Résultats

On obtient des résultats synchronisés presque instantanément entre application mobile, cloud (sur PC) et arduino. Dans la figure qui suit on illustre le cas d'une LED allumée/éteinte et son état sur les 2 plateformes et sur matériel.

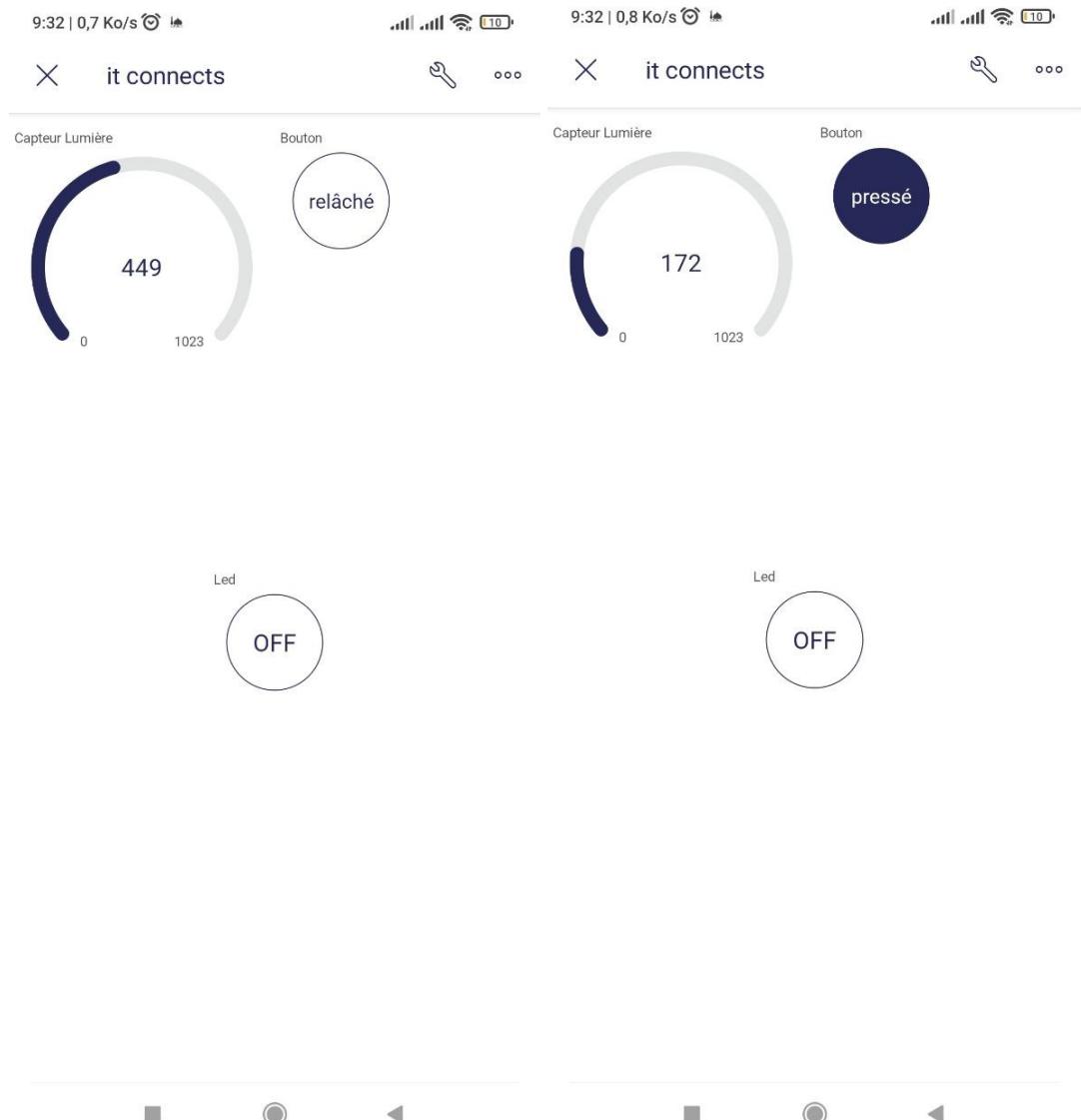


Figure 6. Contrôle et moniteur des capteurs/actionneurs d'arduino à travers Blynk IoT.

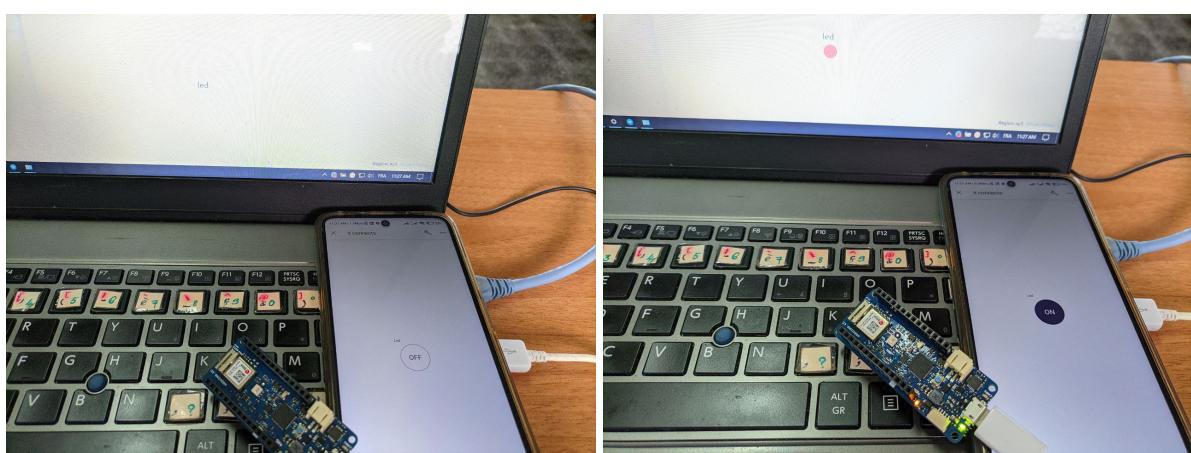
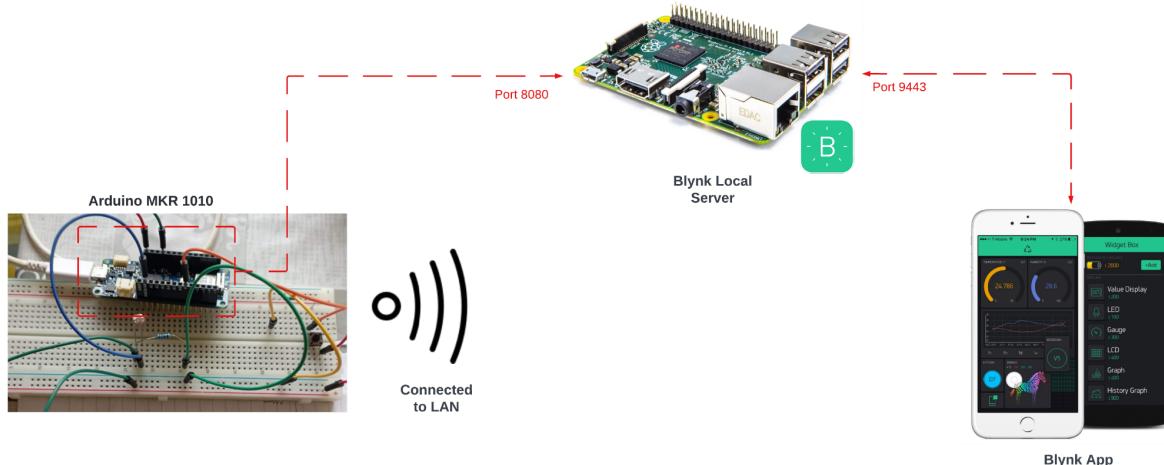


Figure 7. Set-up simple contrôlant led interne de l'arduino à travers Blynk IoT.

2.3. Installer Blynk avec serveur local

Pour utiliser Blynk en standalone nous avons besoin alors d'un serveur Blynk local , le serveur Blynk est un serveur Java Open-Source qui est responsable de la transmission des messages entre l'application mobile Blynk et diverses cartes microcontrôleurs (Arduino et les composantes électroniques).

Pour cela nous avons utilisé Raspberry pi comme un serveur blynk local selon le schéma :



Pour mettre en place cet architecture on suit les étapes suivantes :

- Télécharger Java sur raspberry pi

```
sudo apt update  
sudo apt install default-jdk
```

- Télécharger module .jar pour exécuter le programme serveur depuis le répertoire officiel de blynk (<https://github.com/blynkkk/blynk-server/>)

```
mkdir blynk && cd blynk  
wget "https://github.com/blynkkk/blynk-server/releases/download/v0.41.17/server-0.41.17.jar"
```

- Activer la messagerie sur le serveur local (smtp)

```
nano mail.properties
```

```
pi@raspberrypi: ~  
GNU nano 5.4  
mail.smtp.auth=true  
mail.smtp.starttls.enable=true  
mail.smtp.host=smtp.gmail.com  
mail.smtp.port=587  
mail.smtp.username@example@gmail.com  
mail.smtp.password=email_password  
mail.smtp.connectiontimeout=30000  
mail.smtp.timeout=120000
```

remplacer l'email et le mot de passe par un vrai email et mot de passe, l'email doit être de type gmail et on doit autoriser les applications les moins sécurisées de se connecter au mail pour que le serveur puisse envoyer un token pour chaque projet créer sur blynk.

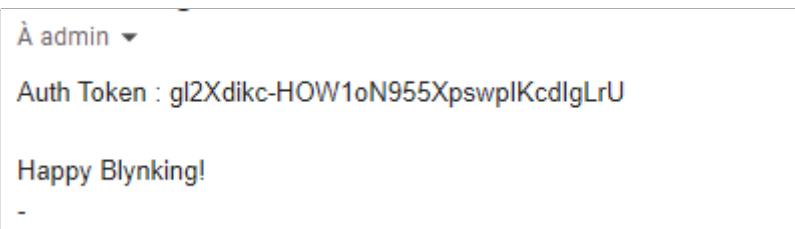
← Accès aux applications moins sécurisées

Certaines applications et certains appareils utilisent une technologie de connexion moins sécurisée, qui rend votre compte vulnérable. Vous pouvez désactiver l'accès pour ces applications (recommandé) ou l'activer si vous voulez les utiliser malgré les risques encourus. Google DÉSACTIVE automatiquement ce paramètre s'il n'est pas utilisé. [En savoir plus](#)

Paramètre "Autoriser les applications moins sécurisées"
activé



Le token d'authentification reçu à la création d'un nouveau projet sur Blynk :



redémarrer le serveur pour sauvegarder les changements.

reboot

- Lancer le serveur et Ouvrir les ports **9443** (pour l'application et le matériel avec ssl), **8080** (pour le matériel sans ssl).

```
java -jar server-0.41.17.jar -dataFolder ~/Documents/blynk  
pi@raspberrypi:~/Documents/blynk $ java -jar server-0.41.17.jar -dataFolder ~/Documents/blynk  
Blynk Server successfully started.  
All server output is stored in folder '/home/pi/Documents/blynk/logs' file.  
Admin password not specified. Random password generated.  
Your Admin url is https://127.0.1.1:9443/admin  
Your Admin login email is admin@blynk.cc  
Your Admin password is 0VvxE6PWQ0pJYOEYRctmMeJ4
```

- Se connecter sur l'application **Blynk** (l'appareil mobile doit être connecté sur le même réseau que le serveur blynk).

On doit utiliser l'ancienne application Blynk (et pas Blynk IoT) et on connecte en mode custom au serveur raspberry pi sur le port **9443**:



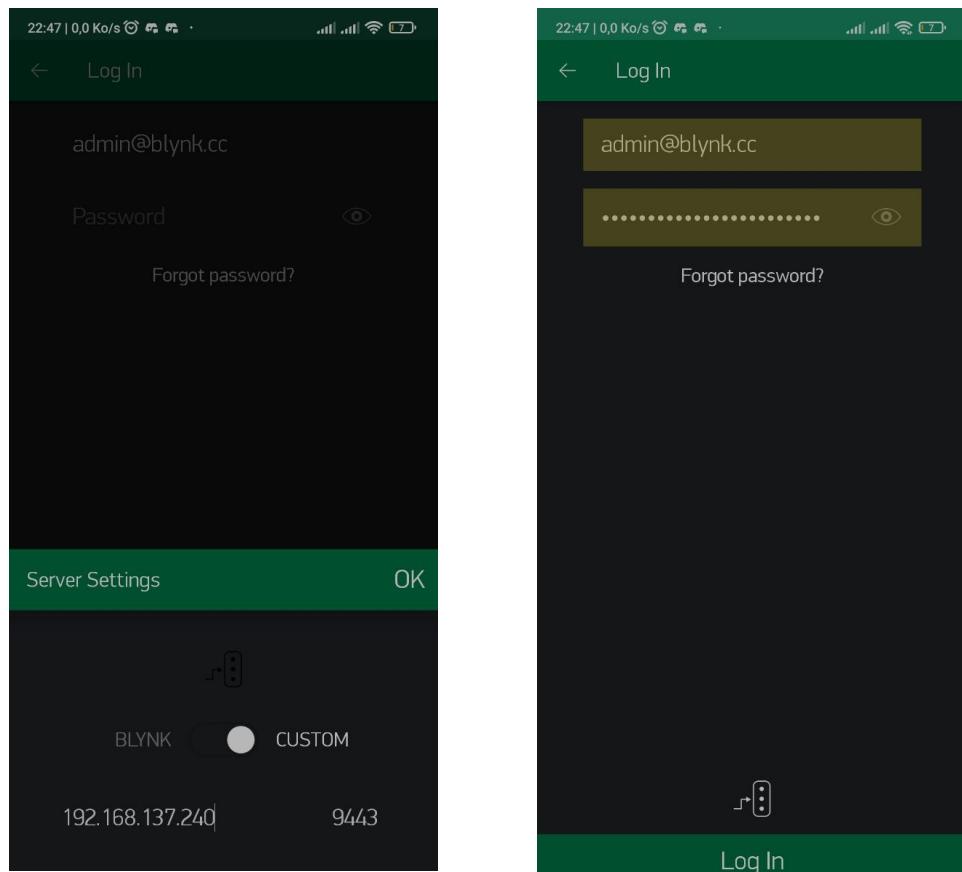


Figure 8. se connecter à l'application Blynk en mode local.

- Créer un nouveau projet et ajouter les composantes nécessaires en faisant drag-n-drop dans l'application puis en configurant les pins utilisées dans le branchement avec les composantes ajoutées. Ici, on a ajouté un bouton pour allumer/éteindre la led et un gauge pour récupérer la valeur générée par le capteur de lumière.

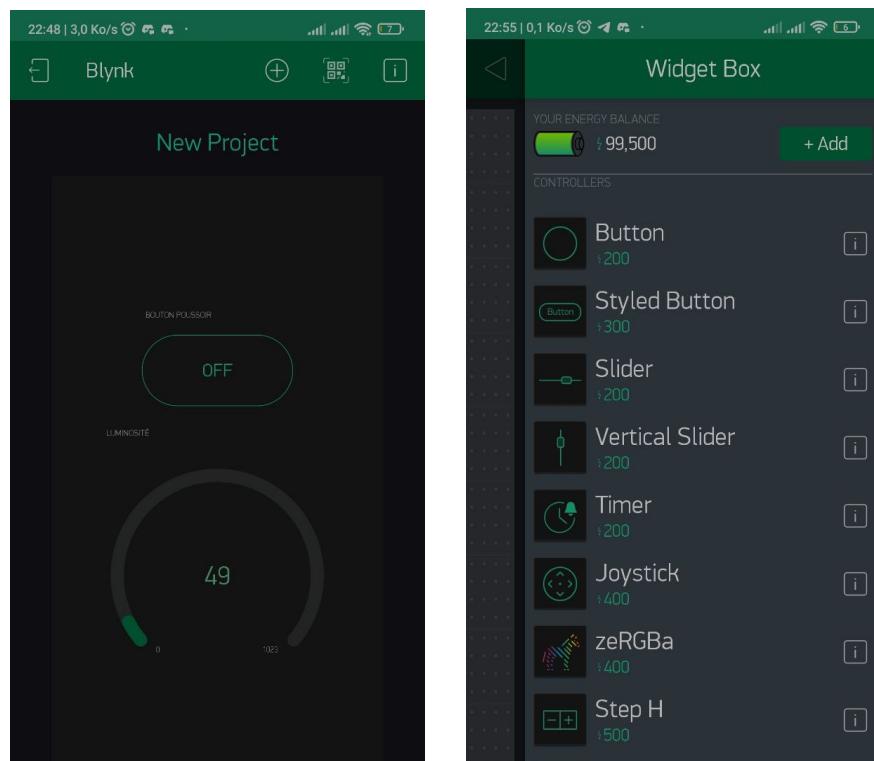


Figure 9. Ajout des composants sur l'application Blynk.

- Connecter arduino au serveur blynk local

On utilise le même programme utilisé avec Blynk sur cloud , on change que le ssid/pass pour connecter l'arduino au même réseau que le raspberry pi , on inclut aussi les informations qui permet à l'arduino de se connecter au serveur local Blynk (l'adresse ip du serveur et le port):

```
#include <WiFiNINA.h>
#include <BlynkSimpleWiFiNINA.h>

// To connect to Blynk local server
char auth[] = "gl2Xdikc-HOWloN955XpswpIKcdIgLrU";
char ssid[] = "anii76";
char pass[] = "00000000";
...
Et au niveau connexion:
...
//Connecting to Rasp server
Blynk.begin(auth, ssid, pass, "192.168.137.240", 8080);
```

- Tests

Quand la carte arduino connecte au serveur ,l'application la détecte par la suite et affiche les valeurs envoyées par le capteur de lumière , ainsi qu'on peut contrôler la led on utilisant le bouton on/off.

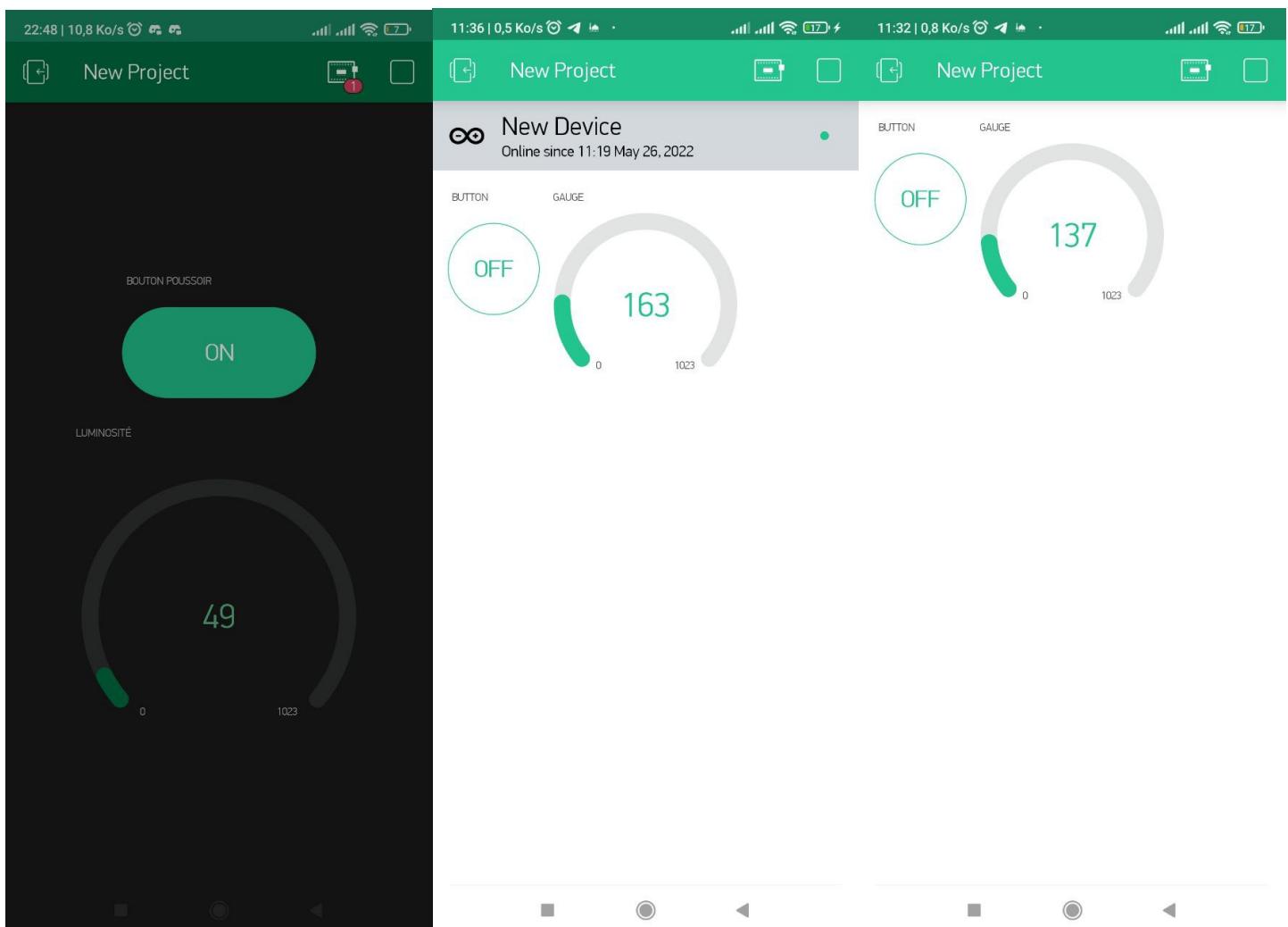


Figure 10. Application Blynk testé au local.