

การใช้งาน ThingsBoard IoTs Platform เพื่อสร้างและจัดการระบบอัจฉริยะ ThingsBoard IoTs Platform for smart system

2/4 – ThingsBoard IoTs Platform

- IoTs Platform
- Best IoT Platforms for building IoT projects
- การโปรแกรมเพื่อแสดงค่าจาก ESP32 ไปที่ ThingsBoard
- การโปรแกรมเพื่อควบคุมและแสดงค่าระหว่าง ESP32 และ ThingsBoard
- โครงการ “ตรวจสอบอุณหภูมิความชื้นของพาร์มไก่ ให้มีการควบคุมอุปกรณ์และมีการแจ้งเตือน”
- คำถ้าหากทับที่เพื่อทดสอบความเข้าใจ

1/6 -- IoTs Platform

<https://www.scimath.org/article-technology/item/9084-2018-10-18-07-45-35>

ประเทศไทย 4.0 โมเดลขับเคลื่อนประเทศไทยสู่ความมั่งคั่ง มั่นคง และยั่งยืนนั้น เป็นยุคเทคโนโลยี Creative และ Innovation เน้นการสร้างให้คนไทยสามารถคิดเองได้ การที่จะให้บุคลากรในประเทศไทยได้ใช้เทคโนโลยีที่เหมาะสมในการบูรณาการกับความรู้ทางด้านวิทยาศาสตร์และคณิตศาสตร์ เพื่อแก้ปัญหาหรือพัฒนางานด้วยกระบวนการออกแบบ เชิงวิศวกรรม ที่นำไปสู่ การคิดค้นสิ่งประดิษฐ์ หรือสร้างนวัตกรรมต่าง ๆ ที่เอื้อประโยชน์ต่อการดำเนินชีวิตได้นั้น จำเป็นต้องมีระบบ生態系 (Ecosystem) ด้าน IoT (Internet of Things) ซึ่งเป็นเทคโนโลยีที่เป็นตัวพลักดันขับเคลื่อน ประเทศไทย 4.0

1.1 IoT Platform คืออะไร

IoT Platform เป็นองค์ประกอบที่สำคัญของระบบบันทึกของ IoT แต่หลายคนไม่ทราบว่า IoT Platform คืออะไร ความแตกต่างระหว่างแต่ละคืออะไร

ดังนั้น IoT Platform คืออะไร มาดูกันแต่ละข้อ

- Run data** ระบบ IoT ที่สมบูรณ์ต้องใช้ hardware เชน เชิงเซอร์หรืออุปกรณ์ เชิงเซอร์และอุปกรณ์เหล่านี้รวบรวมข้อมูลจากสิ่งแวดล้อม (เช่น เชิงเซอร์วัดความชื้น) หรือดำเนินการในสิ่งแวดล้อม (เช่น การรดน้ำพืช)
2. ระบบ IoT ที่สมบูรณ์แบบต้องการการเชื่อมต่อ ฐานะเว็บต้องมีวิธีในการส่งข้อมูลทั้งหมดไปยังระบบคลาวด์ (เช่น การส่งข้อมูลความชื้น) หรือต้องการวิธีรับคำสั่งจากระบบคลาวด์ (เช่น ให้น้ำตอนนี้)
- Save Cloud** สำหรับระบบ IoT บางระบบอาจมีขั้นตอนกลางระหว่าง hardware และการเชื่อมต่อ กับระบบคลาวด์ เช่น การรับข้อมูลความชื้นที่ผ่านเซ็นเซอร์แล้วทำการตัดสินใจ (เช่น รู้ว่า ก็อข้อมูลความชื้นที่ฝนตกและบอกระบบชลประทานไม่ให้เปิดวันนี้)
3. ระบบ IoT ที่สมบูรณ์แบบต้องการซอฟต์แวร์ ซอฟต์แวร์นี้เป็นโอลต์ในระบบคลาวด์ และมีหน้าที่ในการประมวลผลข้อมูลที่รับมาจากเซ็นเซอร์และทำการตัดสินใจ (เช่น รู้ว่า ก็อข้อมูลความชื้นที่ฝนตกและบอกระบบชลประทานไม่ให้เปิดวันนี้)
4. สุดท้ายระบบ IoT ที่สมบูรณ์แบบจำเป็นต้องมีส่วนติดต่อผู้ใช้ เพื่อให้สัมผัสถึงประโยชน์ต้องมีวิธีสำหรับผู้ใช้โดยทั่วไป ระบบ IoT (เช่น แอปบนเว็บหรือแอปพลิเคชัน IoT Platform มีอยู่ใน 3 ส่วนแรก และมักเป็นส่วนที่ 4 ของสิ่งที่อธิบายข้างต้น กับทุกชนิดที่แตกต่างกันของ hardware และตัวเลือกการเชื่อมต่อที่แตกต่างกันจะต้องมีวิธีการทำให้ทุกอย่างทำงานร่วมกัน แพลตฟอร์ม IoT ช่วยแก้ปัญหาดังกล่าว)

IoT Platform เป็นซอฟต์แวร์สนับสนุนที่เชื่อมต่อทุกอย่างในระบบ IoT แพลตฟอร์ม IoT ช่วยให้การสื่อสาร การให้ผลข้อมูลการจัดการอุปกรณ์และการทำงานของแอปพลิเคชัน IoT Platform มีอยู่ใน 3 ส่วนแรก และมักเป็นส่วนที่ 4 ของสิ่งที่อธิบายข้างต้น กับทุกชนิดที่แตกต่างกันของ hardware และตัวเลือกการเชื่อมต่อที่แตกต่างกันจะต้องมีวิธีการทำให้ทุกอย่างทำงานร่วมกัน แพลตฟอร์ม IoT ช่วยแก้ปัญหาดังกล่าว

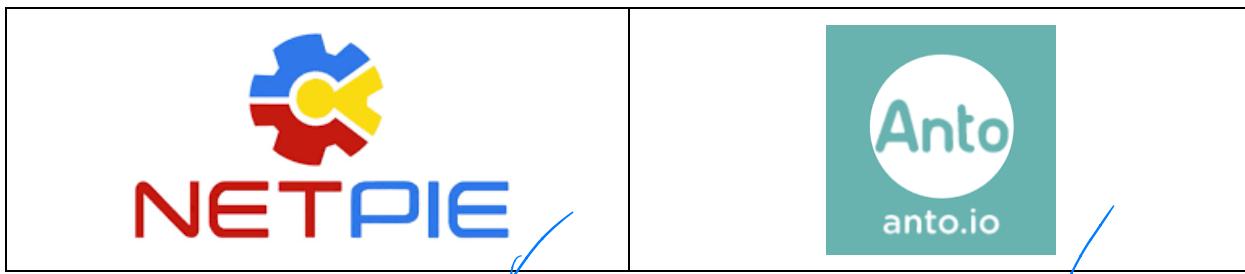
1.2 IoT Platform ช่วยให้

- เสื่อมต่อ hardware เซ็นเซอร์และอุปกรณ์
- จัดการโปรแกรมการสื่อสารของฟ็อกแวร์และ hardware ที่แตกต่างกัน
- ให้การรักษาความปลอดภัยและการตรวจสอบลิขสิทธิ์สำหรับอุปกรณ์และผู้ใช้
- เก็บภาพและวิเคราะห์ข้อมูลที่เขียนเซอร์และอุปกรณ์ต่าง ๆ รวมรวม
- ผสานรวมทั้งหมดข้างต้นกับบริการเว็บอื่น ๆ

1.3 IoT Platform ฝีมือคนไทย

ในปัจจุบันมี IoT Platform ที่พัฒนาโดยคนไทย ให้เลือกใช้งานมากมาย เช่น NETPIE, Anto.io (Anto IoT platform), IoTweet, AIS NB-IoT และ True NB-IoT

- **NETPIE:** เป็นผลงานวิจัยของ ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ (NECTEC) ที่สนับสนุน Arduino Platform และ Raspberry Pi โดยใช้ชุดไลบ์แลรี่ที่ชื่อว่า Microgear โดยมีหน้าจอ แดชบอร์ดสำหรับสร้างเครื่องมือควบคุมได้
- **Anto.io (Anto IoT platform):** เป็น IoT Platform ที่ทำให้เราสามารถเรียนรู้และเข้าใจเกี่ยวกับความรู้พื้นฐานเบื้องต้นเกี่ยวกับ IoT ได้อย่างรวดเร็ว ด้วยความที่ Anto นั้นได้เรียงลำดับในการอธิบายได้ดี ซึ่งเริ่มจาก Introduction จะพูดถึง Anto มีหน้าที่อะไร และรองรับอุปกรณ์ใดบ้าง พร้อมทั้งมี Prepare และ Quick Start ที่มีขั้นตอนการอธิบายที่เรียบง่าย ทำให้เราสามารถใช้เวลาทำงานได้อย่างรวดเร็ว แต่ Anto นี้สามารถรองรับได้เพียง ESP8266 และ Arduino เท่านั้น จุดเด่นหลักของ Anto ก็คือมี Pattern การสอนที่เรียบง่าย ทำให้ตามได้อย่างรวดเร็วจนรู้สึกว่า เอ้า!! เลร์นแล้วหรอ นอกจากนี้ Anto ยังมี Dashboard ในตัวซึ่งมีความสนุกตรงที่ Dashboard นั้นมีความสวยงามและยืดหยุ่นมาก พร้อมทั้งยังสามารถติดตามและปรับค่าของ Thing ได้ จากรูปลักษณ์แล้วทำให้ผู้ใช้งานเข้าถึงได้ง่ายมาก และหน้าตาออกแบบน่าใช้งาน พร้อมทั้งสามารถเปิดใช้ผ่านมือถือได้ นั่นคือเราสามารถเปิดเว็บ Anto และ Login เข้ามาในหน้านี้ก็สามารถควบคุมอุปกรณ์ได้
- **IoTweet:** เป็นผลงานของคุณ อิศราณุ จันทร์ทอง สามารถใช้งาน สนับสนุน Arduino Platform และ Raspberry Pi โดยมีหน้าจอแดชบอร์ดใช้งานง่ายไม่กี่นาที ก็สามารถควบคุมอุปกรณ์ในการรับ-ส่งข้อมูลได้แล้ว
- **AIS NB-IoT:** เป็นผลิตภัณฑ์ของบริษัท แอดวานซ์ อินโนเฟร์ เซอร์วิส จำกัด (มหาชน) หรือ AIS ที่มี platform ที่ใช้งานชื่อว่า Magellan
- **True NB-IoT:** เป็นผลิตภัณฑ์ของบริษัท ทรู ดิจิตอล แอนด์ มีเดีย แพลตฟอร์ม จำกัด ที่มี platform ที่ใช้งานชื่อว่า Thingsboard



1.4 นำ IoT Platform ไปใช้ในด้านใดบ้าง

NETPIE

- ชุด KidBright เชื่อมต่อ Internet of things (IoT) ผ่านแอปพลิเคชันบนสมาร์ทโฟนด้วย KidBright App (Android)
- บุคคลธรรมดายังสามารถที่นำไปใช้ในงานต้นทางการเรียนการสอนการเชื่อมต่ออุปกรณ์ในรูปแบบ IoT (Internet of Things)

Anto.io (Anto IoT platform)

- ระบบงานต่างๆ ที่ใช้งานด้าน IoT (Internet of Things)
- บุคคลธรรมดายังสามารถที่นำไปใช้ในงานต้นทาง
- การเรียนการสอนการเชื่อมต่ออุปกรณ์ในรูปแบบ IoT (Internet of Things)

IoTtweet

- ระบบงานต่างๆ ที่ใช้งานด้าน IoT (Internet of Things)
- บุคคลธรรมดายังสามารถที่นำไปใช้ในงานต้นทาง
- การเรียนการสอนการเชื่อมต่ออุปกรณ์ในรูปแบบ IoT (Internet of Things)

AIS NB-IoT

- บมจ.ปตท. : นำ IoT ไปตรวจสอบท่อส่งก๊าซธรรมชาติ ผ่านเซ็นเซอร์ที่จะส่งข้อมูลให้ผู้ควบคุมได้ทันที
- บมจ.พีโอเพอร์ตี้ เพอร์เฟค : นำ IoT ไปยกระดับใน 15 โครงการที่พักอาศัยให้เป็นรูปแบบ Smart City
- โคตรตอนกรุ๊ป : ผู้ผลิต และจำหน่ายเครื่องหมายด้วยกลุ่รูปแบบต่าง ๆ ประยุกต์ใช้ IoT กับเซ็นเซอร์ ตรวจจับเรียลไทม์ในเครื่อง เมื่อเรียลไทม์สามารถนำไปเก็บได้ทันเวลา
- มหาวิทยาลัยสงขลานครินทร์ นำเซ็นเซอร์ตรวจวัดสภาพอากาศ และบริหารจัดการน้ำ ไปใช้งานจริงที่ เทศบาลนครหาดใหญ่ จังหวัดสงขลาและจังหวัดภูเก็ต
- มหาวิทยาลัยขอนแก่น นำ Smart Trash Bin แก้ไขปัญหาขยะล้นถัง ด้วยเซ็นเซอร์ที่ส่งลัญญาณรายได้ IoT Platform
- MoBike ที่สามารถรายงานตำแหน่งของจักรยานจำนวนมากได้จากอุปกรณ์ติดตั้ง ทำให้คุณย์กลาง สามารถติดตามจักรยานและจัดการวางแผนจักรยานตามความต้องการของผู้ใช้ได้

True NB-IoT

- ระบบงานต่าง ๆ ที่ใช้งานด้าน IoT (Internet of Things)
- บุคคลธรรมดายังสามารถที่นำไปใช้ในงานต้นทาง

 IoTtweet	 ✓
 true IoT	 ✓

2/6 -- Best IoT Platforms for building IoT projects

<https://iotbyhvm.ooo/best-iot-platforms/>

In this post You can find out a list of the Best IoT platforms. Today, Internet of things (IoT) is one of the fastest growing industries. IoT enabled Devices are around us such as Smart watches and wearables, Smart homes and so on. An IoT platform offers several services that simplify the project development and a set of tools to remotely manage devices. Generally speaking, an IoT platform is a multi-layer technology that enables users to manage connected devices.

Below there is a list of Best IoT platforms in a random order. Some of these platforms have a free account and some have a premium account that enables other interesting features.

Basically, An IoT Platform provides these services:

- Data ingestion
- Dashboard creation
- Data transformation
- Device management
- Rule management
- Platform integration
- Security services

2.1 Best IoT Platforms

ThingSpeak

ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. You can send data to ThingSpeak™ from your devices, create instant visualizations of live data, and send alerts using web services like Twitter® and Twilio®. With MATLAB® analytics inside ThingSpeak, you can write and execute MATLAB code to perform preprocessing, visualizations, and analyses. ThingSpeak enables engineers and scientists to prototype and build IoT systems without setting up servers or developing web software.

Features:

- Realtime sensor data visualization
- Data aggregation from 3rd parties providers
- Schedule IoT analytics tasks to analyze data
- Event scheduling
- Run actions according to data acquired

Kaa

Kaa is an enterprise-grade IoT platform built on a modern cloud-native architecture and a fully customizable feature set. Based on flexible microservices, Kaa easily adapts to almost any need and application. It scales from a tiny start-up to a massive corporation and supports advanced deployment models for multicloud IoT solutions. But you can also use it to put together a smart thermostat for your living room. As long as it's IoT, it's Kaa.

Features:

- Device connectivity
- Device management
- Data collection
- Data processing and analysis
- Data visualization
- Command execution

Adafruit IO

Adafruit.io is a *cloud service* – that just means we run it for you and you don't have to manage it. You can connect to it over the Internet. It's meant primarily for storing and then retrieving data but it can do a lot more than just that!

Features:

- Display your data in real-time, online
- Make your project internet-connected: Control motors, read sensor data, and more!
- Connect projects to web services like Twitter, RSS feeds, weather services, etc.
- Connect your project to other internet-enabled devices
- The best part? All of the above is do-able for **free** with Adafruit IO

AWS IoT

AWS IoT (Amazon internet of things) is an Amazon Web Services platform that collects and analyzes data from internet-connected devices and sensors and connects that data to AWS cloud applications.

AWS offers a set of services:

- Amazon FreeRTOS: This is an OS for microcontrollers that provides some services as connectivity, security, easy programming
- AWS Greengrass: It is a software that enables to run local computation on edge devices
- AWS IoT Analytics: It enables us to run sophisticated analytics on gathered data
- AWS IoT device management: It simplifies the process of device management especially when there are thousands of connected devices
- AWS IoT Core: It is the core of the AWS IoT and enables connected devices to interact with cloud services easily and securely

Google Cloud IoT

Google Cloud IoT is one of the most interesting enterprise platforms. This IoT platform has a set of tools to manage connected devices and the data at the edge level or in the cloud. The connected device can use cloud Pub/Sub to publish the data. Moreover, we can apply BigQuery analysis or we can apply Machine learning on this data. Google Cloud IoT has a reference architecture that describes the role of each component that builds this platform. There are countless possibilities to explore using professional services. This platform has all the services an IoT platform must have starting from the security aspects.

Microsoft Azure IoT

Microsoft Azure IoT is another IoT platform. This is a professional platform with several services. It supports bi-directional communication between connected devices and the platform itself using IoT standard protocols. Moreover, it supports device authentication to address all the security aspects. Microsoft Azure IoT simplify the process of IoT project development addressing all the challenges we have to face during this process starting from the security aspects.

Artik Cloud

Artik Cloud is an IoT platform developed by Samsung. This platform enables devices to connect each other and connect to cloud services. It has a set of services to rapidly connect devices to the cloud and start gathering data. Moreover, this IoT platform has a set of connectors that can be used to connect to third-party services. Like other platforms, it is possible to store data coming from connected devices and aggregate this information.

IBM Watson IoT

IBM Watson IoT Platform (Connection Service and Analytics Service) is a ready-to-run, pre-integrated SaaS managed service IoT platform with capabilities in connectivity, data management and advanced analytics.

Features:

- Device management: Using this service, it is possible to act remotely on the device such as rebooting or firmware update
- Responsive, scalability, connectivity: The platform uses industry standard protocol MQTT to exchange data
- Secure communication: Secure data exchange using MQTT and TLS
- Data lifecycle management

 SAMSUNG ARTIK™ Cloud	 IBM Watson
theThings.iO	 Cayenne myDevices

TheThings.io

thethings.io is an another IoT platform that lets any kind of companies to deploy scalable and flexible IoT solutions for their customers and connected products.

Features:

- CONNECTIVITY AND NORMALIZATION
- DEVICES & LICENSES MANAGEMENT
- CLOUD CODE PROCESSING & ACTION MANAGEMENT
- DATA MONITORING & VISUALIZATION ANALYTICS, AI, PREDICTIVE & OTHER TOOLS
- INTEROPERABILITY AND INTEGRATIONS

myDevices Cayenne

Cayenne is the world's first drag and drop IoT project builder that empowers developers, designers and engineers to quickly prototype and share their connected device projects. Cayenne was designed to help users create Internet of Things prototypes and then bring them to production.

There are several major components in the platform:

- Cayenne Mobile Apps – Remotely monitor and control your IoT projects from the Android or iOS Apps.
- Cayenne Online Dashboard – Use customizable widgets to visualize data, set up rules, schedule events and more.

Ubidots

Ubidots technology and engineering stack was developed to deliver a secure, white-glove experience for our users. Device friendly APIs (accessed over HTTP/MQTT/TCP/UDP protocols) provide a simple and secure connection for sending and retrieving data to and from our cloud service in real-time. Ubidots' time-series backend services are performance optimized for IoT data storage, computation, and retrieval. Our application enablement platform supports interactive, real-time data visualization (widgets), and an IoT App Builder that allows developers to extend the platform with their own HTML/JS code for private customization when desired. Ubidots exists to empower your data from device to visualization.

Temboo

This platform uses **choreos** that are connectors toward external services, so that events in Arduino, like sensor signals, can be transformed into different kind of events. Moreover, it provides some logic like IF-THEN. Moreover, Temboo supports M2M applications using MQTT, CoAP, HTTP protocols. The main features provided by Temboo are:

- Code generation: This platform generates optimized code for several devices using many different languages such as Java, C/C++, Python and so on
- Interoperability: Temboo provides a set of services named choreos that simplify the integration process with other cloud services
- Data: Temboo stores and visualizes different kinds of data.

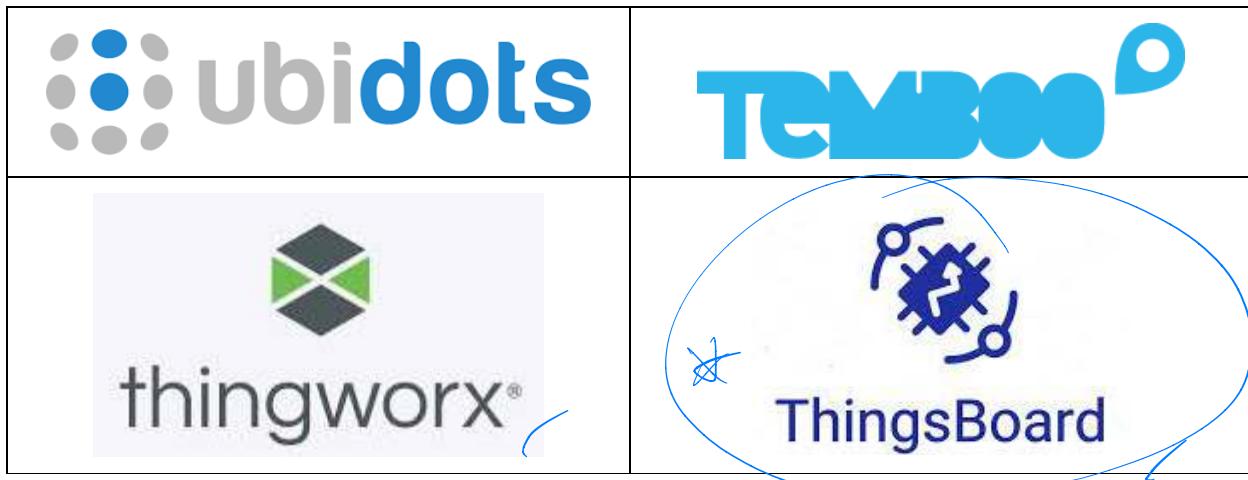
ThingWorx

The ThingWorx low-code IoT development environment provides you with the flexibility to rapidly connect, create, and deploy comprehensive industrial IoT applications. With pre-built extensions and widgets as well as a large ecosystem of partners, ThingWorx addresses the fundamental IoT development challenges, freeing you up to create solutions that accommodate constantly changing business needs.

ThingsBoard

ThingsBoard is an open-source IoT platform for data collection, processing, visualization, and device management

It enables device connectivity via industry standard IoT protocols - MQTT, CoAP and HTTP and supports both cloud and on-premises deployments. ThingsBoard combines scalability, fault-tolerance and performance so you will never lose your data.

**2.2 You may also like:**

- [Dynamic WLAN configuration for ESP32 Board | AutoConnect](#)
- [ESP32 BLE on Arduino IDE with UART Test](#)
- [ESP32 Bluetooth Low Energy \(BLE\) on Arduino IDE](#)
- [ArduinoOTA ESP32: Wi-Fi \(OTA\) Wireless Update from the Arduino IDE](#)
- [ESP32 with LoRa using Arduino IDE](#)
- [How To Use Grove-LCD RGB Backlight with NodeMCU](#)
- [NodeMcu to DHT Interface in Blynk app](#)
- [How To ON/OFF a bulb by Google voice assistant](#)
- [Arduino IDE | Arduino | Open Source Hardware/Softawre | Arduino Vs RPi](#)
- [WiFi LoRA 32 \(V2\) ESP32 | Overview | Introduction](#)
- [DHT11 sensor with ESP8266/NodeMCU using Arduino IDE](#)
- [Arduino Support for ESP8266 with simple test code](#)

2.3 The 7 thoughts on “Best IoT Platforms for building IoT projects”

- Pingback: [Best IoT Platforms for building IoT projects — IoTbyHVM – Bits & Bytes of IoT – hashstacks](#)
- Pingback: [W600-PICO | A W600-Based Board Running MicroPython for Only \\$2?](#)
- Pingback: [What is Azure Sphere ? - IoTbyHVM - Bits & Bytes of IoT](#)
- Pingback: [What is IoT ? | Internet of Things - IoTbyHVM - Bits & Bytes of IoT](#)
- Pingback: [Go Programming Language | Installation | Gobot Framework](#)
- Pingback: [How to setup a Mosquitto MQTT Server and receive data from OwnTracks](#)
- Pingback: [Best IoT Apps for Android - IoTbyHVM - Bits & Bytes of IoT](#)

2.4 Read More

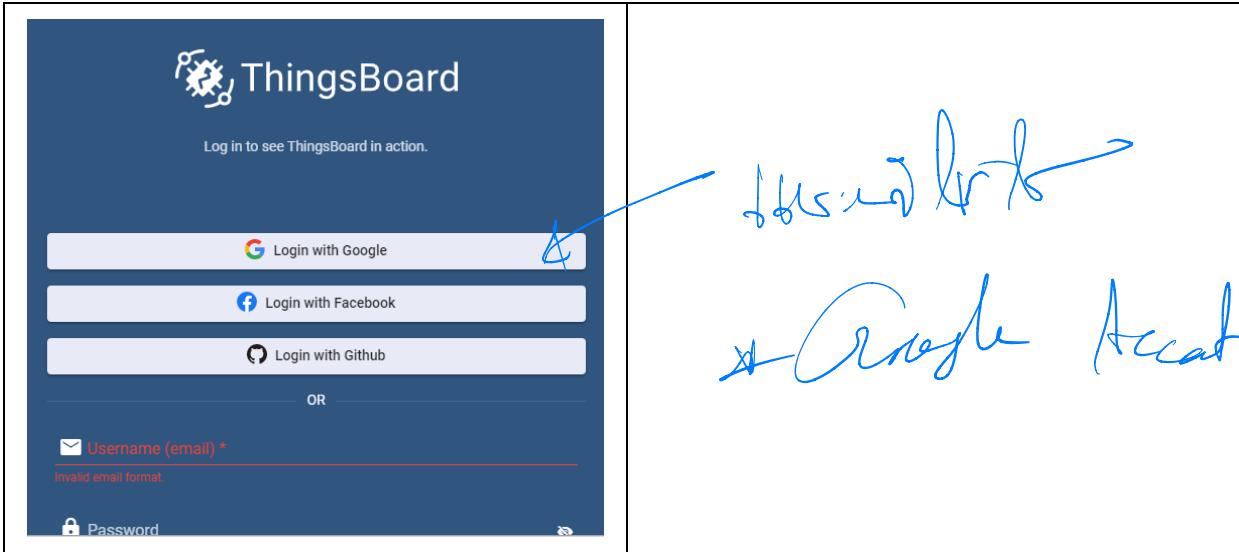
- <http://nilhcem.com/iot/cloud-iot-core-with-the-esp32-and-arduino>
- <https://ludovic-emo.medium.com/how-to-send-esp32-telemetry-to-google-cloud-iot-core-caf1a952020d>
- <https://cloud.google.com/community/tutorials/monitoring-iot-data-grafana>
- <https://www.survivingwithandroid.com/cloud-iot-core-esp32/>
- Cayenne <https://developers.mydevices.com/cayenne/docs/cayenne-mqtt-api/#cayenne-mqtt-api-using-arduino-mqtt>

3/6 -- การโปรแกรมเพื่อแสดงค่าจาก ESP32 ไปที่ ThingsBoard

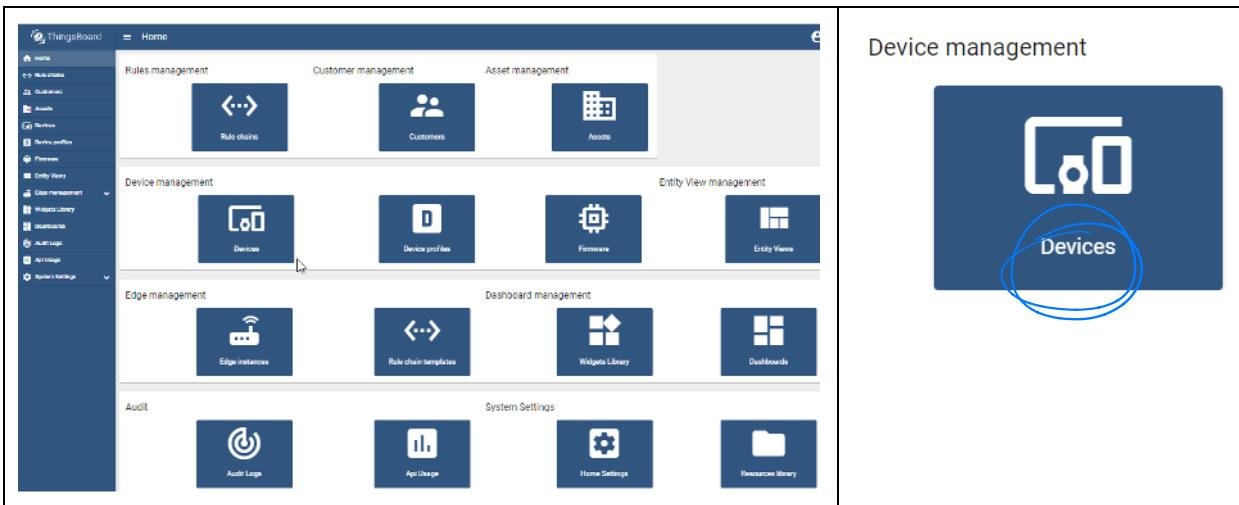
<https://thingsboard.io/docs/samples/esp8266/gpio/>
<https://thingsboard.io/docs/samples/esp32/gpio-control-pico-kit-dht22-sensor/>

Lab201 – ThingsBoard data monitor

- เข้าสู่ระบบ ที่ Web Site Live Demo ของ ThingsBoard <https://demo.thingsboard.io/>



- เข้ามาหน้า Main ของ ThingsBoard → เลือก Device ใน Device Management เพื่อสร้าง Device



Take key
 Device → Variable 2
 Variable 1

3. สร้าง Device ด้วยการกดเครื่องหมาย + และกด Add New Device → ให้กรอกข้อมูลของ Device ดังตัวอย่าง คือ
Name = PkTest1234, Type = Default, กด Add

The screenshot shows two panels. On the left is the main device list interface with a 'Tenant administrator' header, a 'Add Device' button, and search/filter icons (+, -, magnifying glass). On the right is the 'Add new device' dialog box:

- Device details:**
 - Name *: PkTest1234
 - Label: (empty)
 - Transport type *: Default (selected)
 - Description: Supports basic MQTT, HTTP and CoAP transport.
 - Device profile *: default (radio button selected)
 - Create new device profile: (radio button unselected)

4. จะปรากฏ Device PkTest1234 ที่หน้าเหล็ก → Click ที่ Device นี้ แล้วเลือก Copy Access Token Code

The screenshot shows two panels. The top panel is the device list with 'PkTest1234' selected. The bottom panel is a detailed view of 'PkTest1234' with the 'Copy access token' button highlighted.

5. หรือสามารถเรียกดู Access Token ได้จากไอคอน Manage Credential ได้เช่นกัน

The screenshot shows two panels. The left panel has a 'Manage credentials' button and a row of icons. The right panel is the 'Device Credentials' dialog box:

- Credentials type: Access token (selected)
- Access token *: fm6[REDACTED]

6. ติดตั้ง Arduino Library

- Add Library “ThingsBoard by ThingsBoard Team -- V 0.4.0”
- Add Library “ArduinoHttpClient by Arduino -- V 0.4.0”
- Add Library “ArduinoJson by Benoit Blanchon – V6.18.0”

The screenshot shows the Arduino Library Manager interface. The search bar at the top has "thingsboard" typed into it. Below the search bar, there are three library entries:

- ThingsBoard**: Version 0.4.0, INSTALLED. Description: "ThingsBoard library for Arduino. A library for connecting to the ThingsBoard IoT platform. Thin wrapper on PubSubClient." with a "More info" link.
- ArduinoHttpClient**: Version 0.4.0, INSTALLED. Description: "[EXPERIMENTAL] Easily interact with web servers from Arduino, using HTTP and WebSocket's. This library can be used for HTTP (GET, POST, PUT, DELETE) requests to a web server. It also supports exchanging messages with WebSocket servers. Based on Adrian McEwen's HttpClient library." with a "More info" link. There are "Select version" and "Install" buttons.
- ArduinoJson**: Version 6.18.0, INSTALLED. Description: "A simple and efficient JSON library for embedded C++. ArduinoJson supports ✓ serialization, ✓ deserialization, ✓ MessagePack, ✓ fixed allocation, ✓ zero-copy, ✓ streams, ✓ filtering, and more. It is the most popular Arduino library on GitHub ❤️❤️❤️. Check out [arduinojson.org](#) for a comprehensive documentation." with a "More info" link.

7. ทดสอบบอโปรเจกต์ต่อไปนี้ < อย่าลืมแก้ไข Wifi, Pass, Token Key > ดูการทำงานที่ Serial Monitor

```
#include "ThingsBoard.h"
#include <WiFi.h>
#define WIFI_AP "Test1234"
#define WIFI_PASSWORD "MailMeelingJing"
#define TOKEN "fm6EwZ1L9tZrSV1f06pM"
#define THINGSBOARD_SERVER "demo.thingsboard.io"

// Baud rate for debug serial
#define SERIAL_DEBUG_BAUD 115200

// Initialize ThingsBoard client
WiFiClient espClient;
// Initialize ThingsBoard instance
ThingsBoard tb(espClient);
// the Wifi radio's status
int status = WL_IDLE_STATUS;

void setup() {
    // initialize serial for debugging
    Serial.begin(SERIAL_DEBUG_BAUD);
    WiFi.begin(WIFI_AP, WIFI_PASSWORD);
    InitWiFi();
}

void loop() {
    if (WiFi.status() != WL_CONNECTED) {
        reconnect();
    }
    if (!tb.connected()) {
```

↑ key for
↓ Dev → Test
↓ Hard

```

// Connect to the ThingsBoard
Serial.print("Connecting to: ");
Serial.print(THINGSBOARD_SERVER);
Serial.print(" with token ");
Serial.println(TOKEN);
if (!tb.connect(THINGSBOARD_SERVER, TOKEN)) {
    Serial.println("Failed to connect");
    return;
}
}

Serial.print("Sending data...");

// Uploads new telemetry to ThingsBoard using MQTT.
// See https://thingsboard.io/docs/reference/mqtt-api/#telemetry-upload-api
// for more details
float xTemp = random(00, 50);
float xHmid = random(51, 99);
Serial.print(xTemp, 2);
Serial.print(",");
Serial.print(xHmid, 2);
Serial.println();

//tb.sendTelemetryInt("temperature", xTemp);
//tb.sendTelemetryInt("humidity", xHmid);
tb.sendTelemetryFloat("temperature", xTemp);
tb.sendTelemetryFloat("humidity", xHmid);

tb.loop();
delay(5000);
}

void InitWiFi()
{
    Serial.println("Connecting to AP ...");
    // attempt to connect to WiFi network

    WiFi.begin(WIFI_AP, WIFI_PASSWORD);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("Connected to AP");
}

void reconnect() {
    // Loop until we're reconnected
    status = WiFi.status();
    if (status != WL_CONNECTED) {
        WiFi.begin(WIFI_AP, WIFI_PASSWORD);
        while (WiFi.status() != WL_CONNECTED) {
            delay(500);
            Serial.print(".");
        }
        Serial.println("Connected to AP");
    }
}

```

Connecting to AP ...
Connected to AP
 Connecting to: demo.thingsboard.io with token fm6EwZ1L9tZrSV1f06pM
 Sending data...31.00,93.00
 Sending data...47.00,69.00

<input checked="" type="checkbox"/> Autoscroll	<input type="checkbox"/> Show timestamp	Both NL & CR	115200 baud	Clear output
--	---	--------------	-------------	--------------

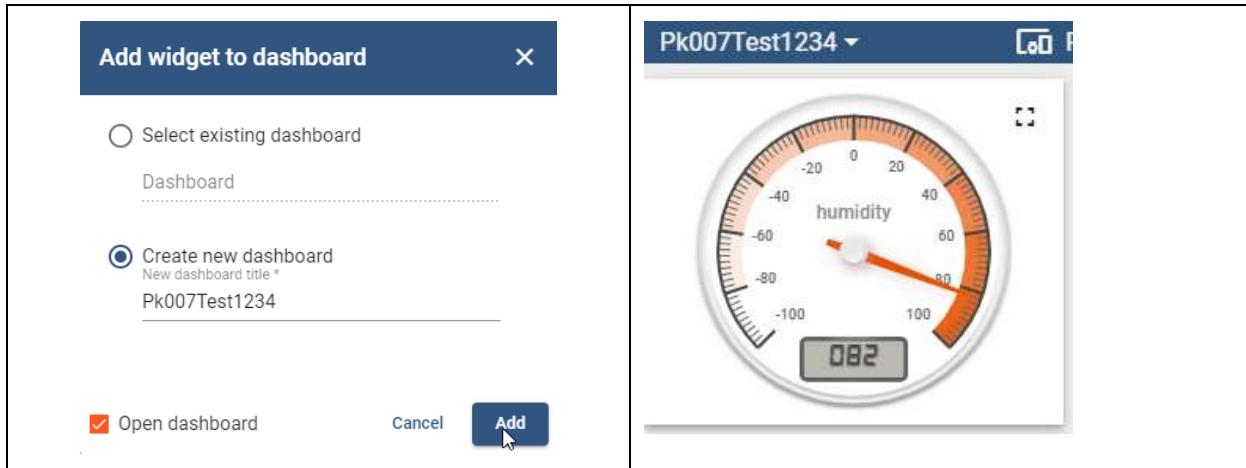
8. กลับมาที่ ThingsBoard หน้า Device ให้เลือก Latest Telemetry จะปรากฏข้อมูลที่ส่งเข้ามา

The screenshot shows the ThingsBoard interface. On the left, there is a sidebar titled "Devices" with a "Device profile" section and a "All" filter. Below it is a table listing five devices with columns for Created time, Name, and Device profile. The first device, "PkTest1234", is selected. On the right, the main panel is titled "PkTest1234" and "Device details". It has tabs for Details, Attributes, Latest telemetry (which is selected), and Alarms. The "Latest telemetry" section displays three data points: Last update time (2021-05-21 23:53:10), Key (humidity), Value (70); Last update time (2021-05-21 23:53:09), Key (temperature), Value (35).

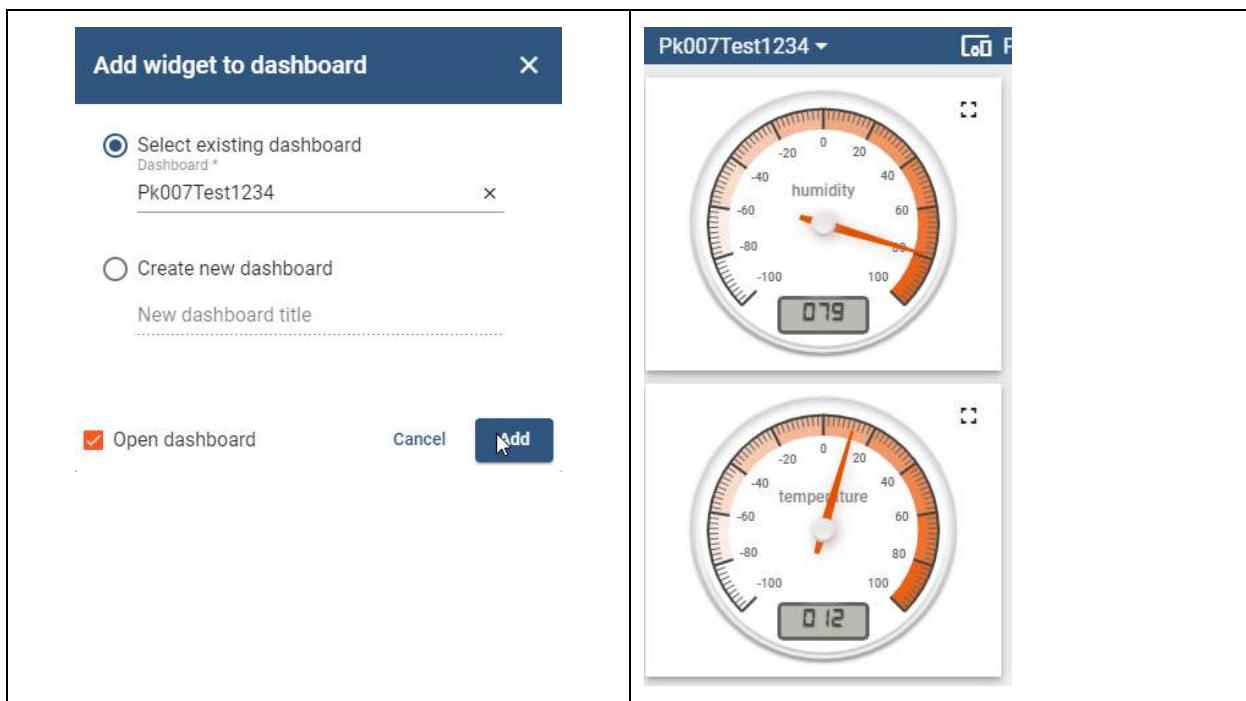
9. เลือกข้อมูล humidity ทั้งแล้วเลือก Show on Widget → ให้ใช้ Analogue Gauges → Add to dashboard

The screenshot shows the ThingsBoard interface. At the top, a message says "1 telemetry unit selected". Below it is a table with three rows: "Last update time" (checkbox icon), "Key ↑" (humidity), "Value" (53); "Last update time" (checkbox icon), "Key ↑" (temperature), "Value" (0). An orange button labeled "Show on widget" is visible. Below this is a section titled "Analogue gauges" with a sub-section "Current bundle". It shows a preview of an analogue gauge for "humidity" with a scale from -100 to 100 and a value of 75. An "Add to dashboard" button is located above the gauge preview.

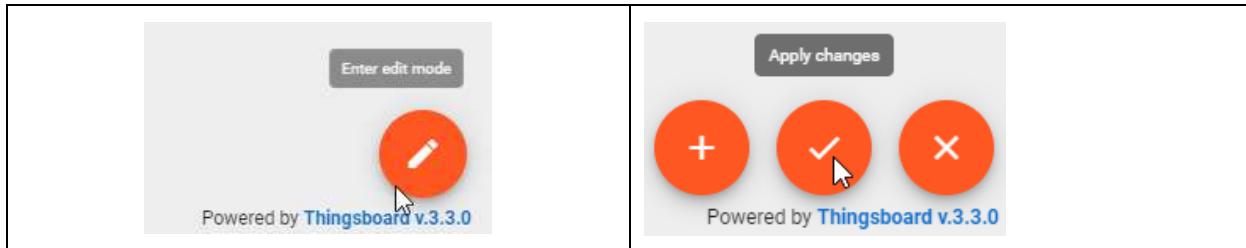
10. กรณี dashboard ใหม่ให้สร้างโดยการ Create new dashboard พร้อมทั้งตั้งชื่อและเลือก Open dashboard ตามด้วยการกด ADD



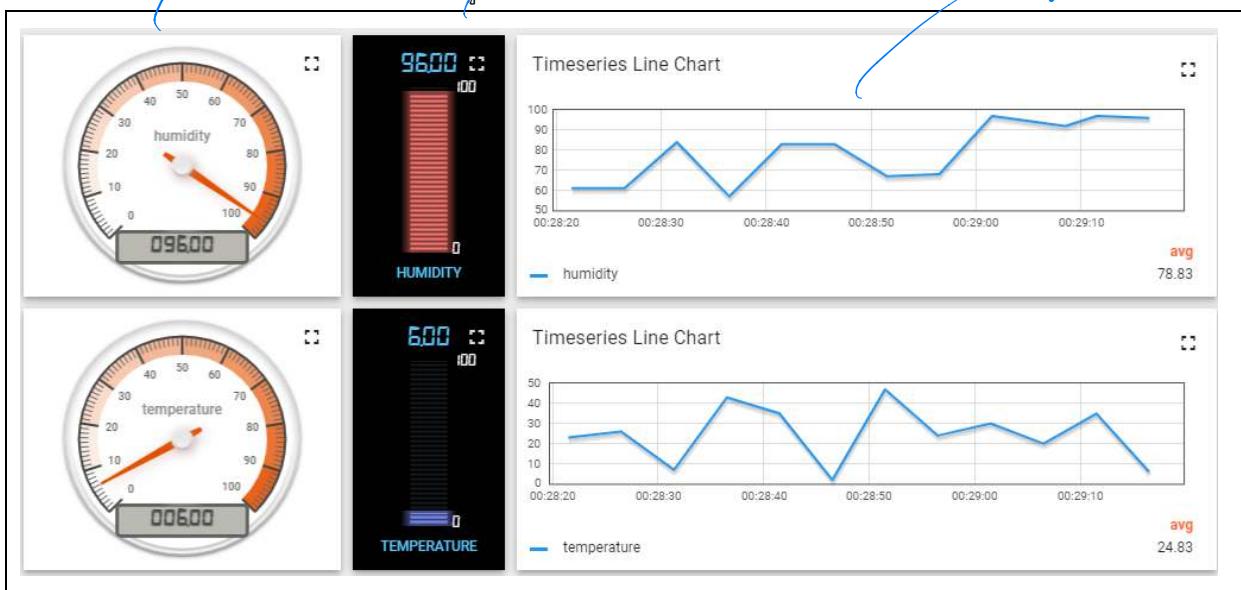
11. ทำข้ากับตัวแปร temperature



12. เข้าไปแก้ไข dashboard โดยการกดไอคอนรูป ติดสอด ด้านล่างขวาฝื้อ, กดเครื่องหมายถูกคือบันทึก



13. ปรับแก้ไข dashboard ให้เป็นตามรูป และมีข้อมูลล็อกเกต ดังนี้



- ค่า Analog Gauge หน้าปัดต้องมีค่าระหว่าง 0-100
- ค่า Digital Gauge หน้าปัดต้องมีค่าระหว่าง 0-100

14. แชร์ Dashboard ให้ User โดยการแชร์ Device และ แชร์ Dashboard

Devices

Created time	Name	Device profile	Label	Customer	Public	Is gateway
2021-05-21 23:23:26	PkTest1234	default		Public	<input checked="" type="checkbox"/>	

Dashboard is now public

Your dashboard **Pk007Test1234** is now public and accessible via next public [link](https://demo.thingsboard.io/dashboard/92acbd00-ba56-11eb-a311-cf80d7127bfd?publicId=40fa9860-ba10-11eb-a311-cf80d7127bfd):

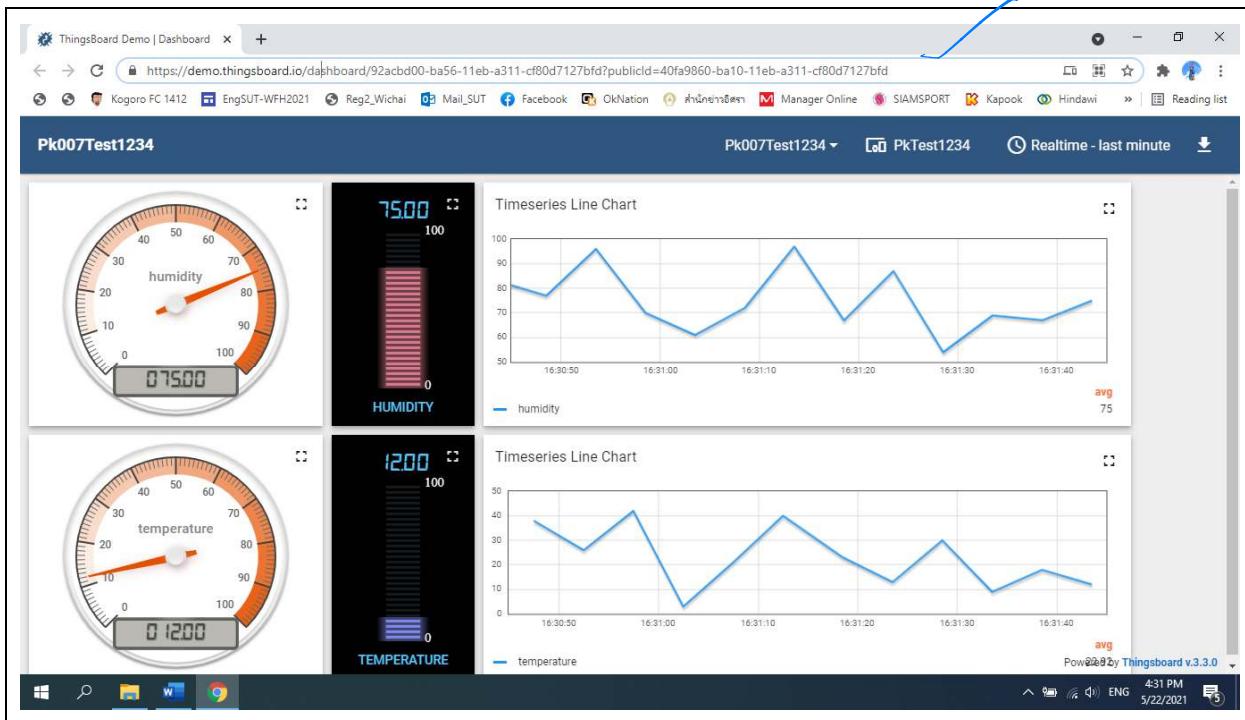
<https://demo.thingsboard.io/dashboard/92acbd00-ba56-11eb-a311-cf80d7127bfd?publicId=40fa9860-ba10-11eb-a311-cf80d7127bfd>

Note: Do not forget to make related devices public in order to access their data.

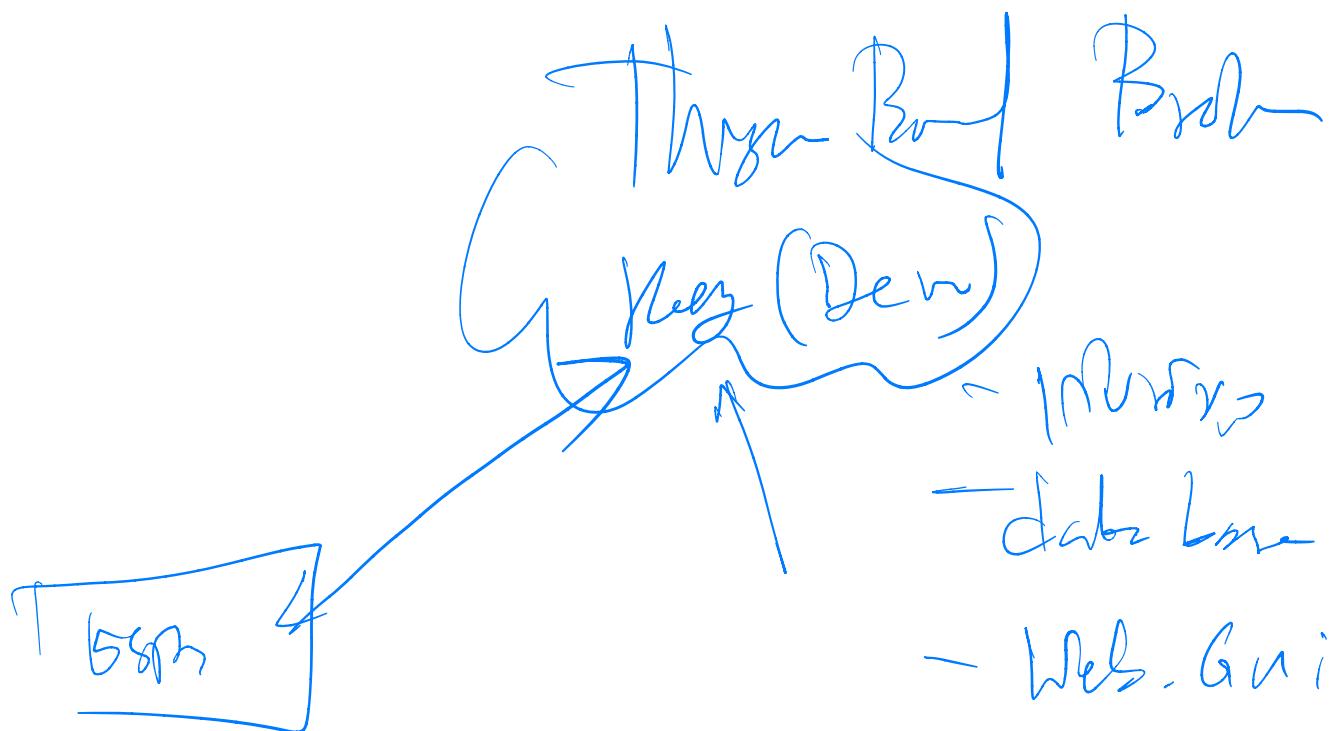
Dashboards

Created time	Title	Assigned to customers	Public
2021-05-22 00:04:15	Pk007Test1234		<input checked="" type="checkbox"/>

15. ตัวอย่างผู้ทดสอบ <https://demo.thingsboard.io/dashboard/92acbd00-ba56-11eb-a311-cf80d7127bfd?publicId=40fa9860-ba10-11eb-a311-cf80d7127bfd>



Mission 1/4: ให้ส่งข้อมูลค่า Humidity และ Temperatures จากเซ็นเซอร์ DHT-22 ไปยัง Dashboard นี้



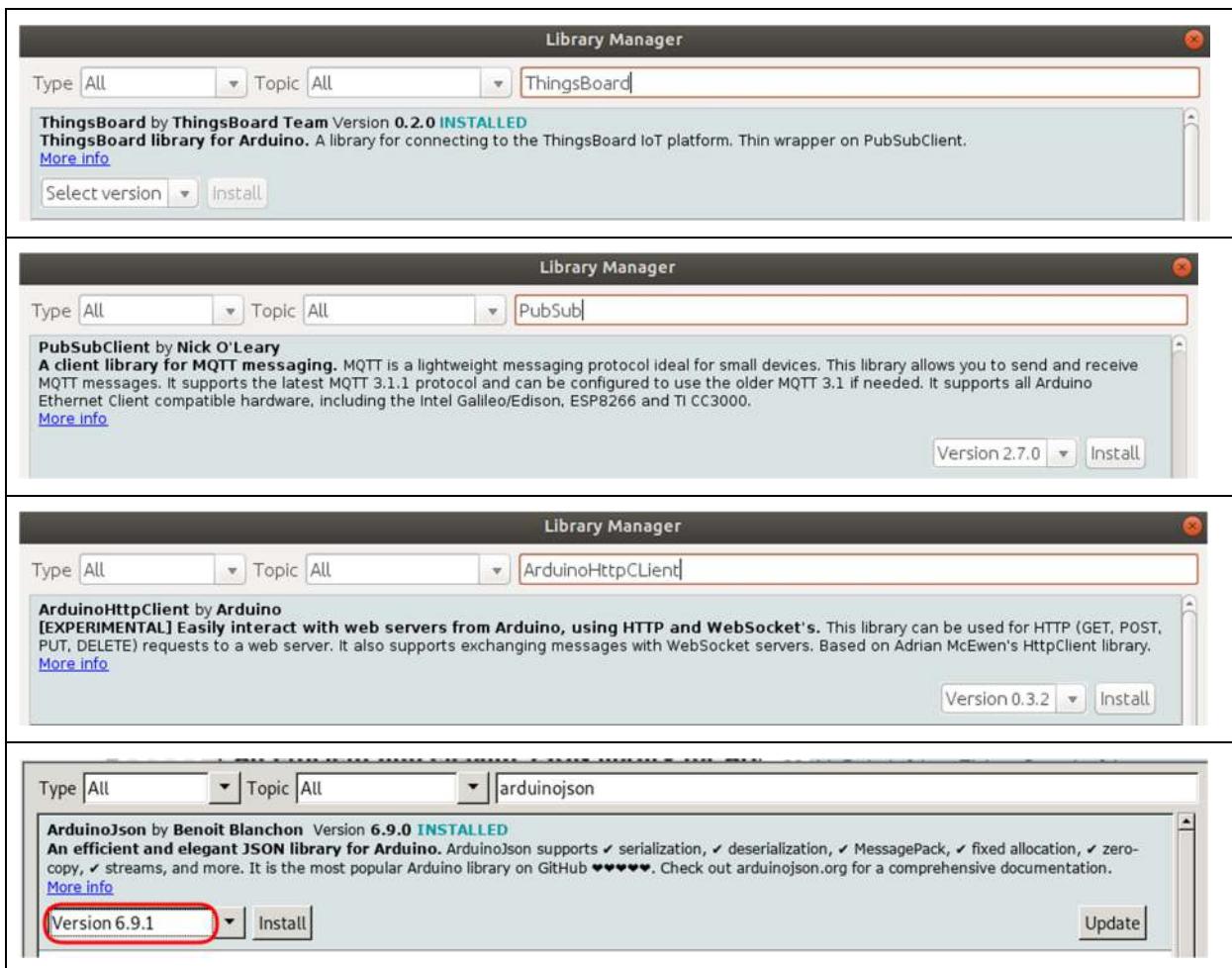
4/6 -- การโปรแกรมเพื่อควบคุมและแสดงค่าระหว่าง ESP32 และ ThingsBoard

<https://thingsboard.io/docs/samples/esp8266/gpio/><https://thingsboard.io/docs/samples/esp32/gpio-control-pico-kit-dht22-sensor/>

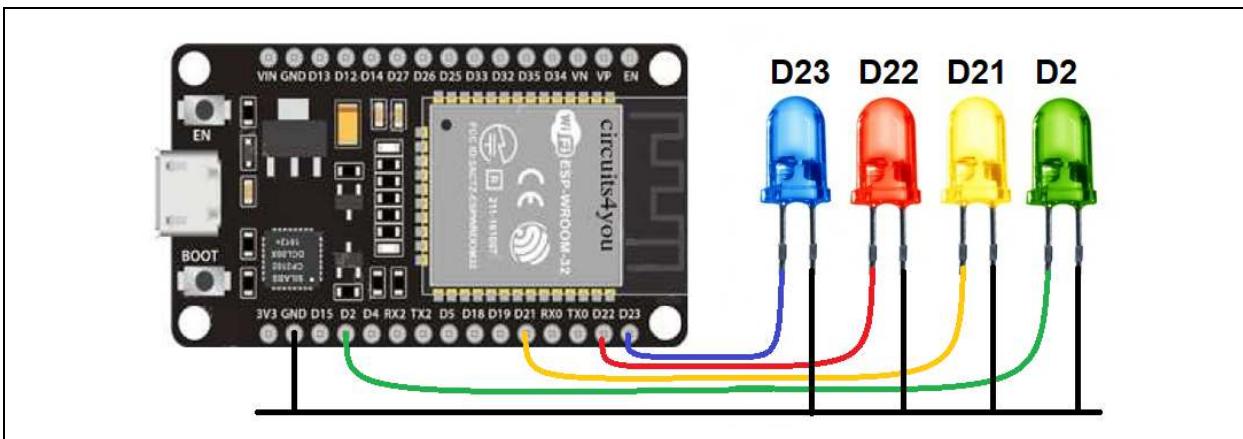
Lab202 -- ThingsBoard data monitor and control

16. ปรับปรุง Arduino Library

- ThingsBoard Arduino SDK Ver 0.2.0
- PubSubClient by Nick O'Leary Ver 2.7.0
- ArduinoHttpClient libraries. Ver 0.3.2
- ArduinoJSON library Ver 6.9.1



17. ต่อวงจร



18. โหลดโปรแกรมไปยัง ESP32 โดยโปรแกรมแบ่งออกเป็น 3 ไฟล์ และอย่าลืมแก้ Wifi-Name, Wifi-Pass, Token Key

_HandOnMQTT.h	5/24/2021 11:17 PM
_WifiConnect.h	5/24/2021 10:58 PM
ESP32_Mission3_GPIO	5/24/2021 11:17 PM

```

// Part 1 of 3
// Helper macro to calculate array size
#define COUNT_OF(x) ((sizeof(x)/sizeof(0[x])) / ((size_t)(!(sizeof(x) % sizeof(0[x])))))

#include <WiFi.h>
#include <ThingsBoard.h>
#define WIFI_AP_NAME "Test1234"
#define WIFI_PASSWORD "passssssssss"
#define TOKEN "kkkkkkkkkkkkkkkk"
#define THINGSBOARD_SERVER "demo.thingsboard.io"
#define pinLEDBlink 2

WiFiClient espClient;
ThingsBoard tb(espClient);
int status = WL_IDLE_STATUS;

uint8_t leds_PinControl[] = {21, 22, 23}; // 1
int leds_Status[] = {0, 0, 0}; // 2
char StringEcho[] = "stsLED_1";

int loopDelay = 20; // Main loop delay(ms)
int sendDataDelay = 2000; // Period of Sending Tempp/Humid.
int BlinkLEDDelay = 500; // Initial period of LED cycling.
int Count_BlinkLEDDelay = 0; // Time Counter Blink peroid
int Count_sendDataDelay = 0; // Time Counter Sending Tempp/Humid

bool Subscribed_Status = false; // Subscribed_Status for the RPC messages.
int ststus_BlinkLED = 0; // LED number that is currently ON.

#include "_ThingBoardRPC.h"
#include "_ConnectWifi.h"

//=====
void setup() {
  // Initialize serial for debugging
  Serial.begin(115200);
  WiFi.begin(WIFI_AP_NAME, WIFI_PASSWORD);
  WiFi_InitStructure();
}

// Pinconfig
pinMode(pinLEDBlink, OUTPUT);
for (size_t i = 0; i < COUNT_OF(leds_PinControl); ++i) {
  pinMode(leds_PinControl[i], OUTPUT);
}

//=====
void loop() {

```

1
2

```

// Step0/6 - Loop Delay
delay(loopDelay);
Count_BlinkLEDDelay += loopDelay;
Count_sendDataDelay += loopDelay; + Delay

// Step1/6 - Check if next LED Blink
if (Count_BlinkLEDDelay > BlinkLEDDelay) {
    digitalWrite(pinLEDBlink, status_BlinkLED);
    status_BlinkLED = 1 - status_BlinkLED;
    Count_BlinkLEDDelay = 0;
}

// Step 2/6 - Reconnect to WiFi, if needed
if (WiFi.status() != WL_CONNECTED) {
    reconnect();
    return;
}

// Step 3/6 - Reconnect to ThingsBoard, if needed
if (!tb.connected()) {
    Subscribed_Status = false;
    // Connect to the ThingsBoard
    Serial.print("Connecting to: "); Serial.print(THINGSBOARD_SERVER);
    Serial.print(" with token: "); Serial.println(TOKEN);
    if (!tb.connect(THINGSBOARD_SERVER, TOKEN)) {
        Serial.println("Failed to connect");
        return;
    }
}

// Step 4/6 - Subscribe for RPC, if needed
if (!Subscribed_Status) {
    Serial.println("Subscribing for RPC... ");
    // Perform a subscription. All consequent data processing will happen in
    // callbacks as denoted by callbacks[] array.
    if (!tb.RPC_Subscribe(callbacks, COUNT_OF(callbacks))) {
        Serial.println("Failed to subscribe for RPC");
        return;
    }
    Serial.println("Subscribe done");
    Subscribed_Status = true;
}

// Step 5/6 - Check if it is a time to send Tempp/Humid
if (Count_sendDataDelay > sendDataDelay) {
    Serial.print("Sending data... ");
    float temperature = random(20, 30);
    float humidity = random(40, 50);
    tb.sendTelemetryFloat("temperature", temperature);
    tb.sendTelemetryFloat("humidity", humidity);
    Serial.print("T=" + String(temperature, 2) + " ");
    Serial.print("H=" + String(humidity, 2) + " ");
    Serial.print("LED=");
    for (size_t i = 0; i < COUNT_OF(leds_PinControl); ++i) {
        StringEcho[7] = 0x30 + i; // Set 0 to "0"
        tb.sendTelemetryInt(StringEcho, leds_Status[i]);
        Serial.print(leds_Status[i]);
    }
    Serial.println();
    Count_sendDataDelay = 0;
}

// Step 6/6 - Process messages
tb.loop();
}

```

```

// Part 2 of 3
// _ThingBoardRPC.h

// Processes function for RPC call "setValue"
// RPC_Data is a JSON variant, that can be queried using operator[]
// See https://arduinojson.org/v5/api/jsonvariant/subscript/ for more details
//=====
RPC_Response processDelayChange(const RPC_Data &data) {
    Serial.println("Received the set delay RPC method");
    BlinkLEDDelay = data;
    Serial.print("Set new delay: ");
    Serial.println(BlinkLEDDelay);
    return RPC_Response(NULL, BlinkLEDDelay);
}

// Processes function for RPC call "getValue"
// RPC_Data is a JSON variant, that can be queried using operator[]
// See https://arduinojson.org/v5/api/jsonvariant/subscript/ for more details
//=====
RPC_Response processGetDelay(const RPC_Data &data) {
    Serial.println("Received the get value method");
    return RPC_Response(NULL, BlinkLEDDelay);
}

// Processes function for RPC call "setGpioStatus"
// RPC_Data is a JSON variant, that can be queried using operator[]
// See https://arduinojson.org/v5/api/jsonvariant/subscript/ for more details
//=====
RPC_Response processSetGpioState(const RPC_Data &data) {
    Serial.println("Received the set GPIO RPC method");
    int pin = data["pin"];
    bool enabled = data["enabled"];
    if (pin < COUNT_OF(leds_PinControl)) {
        Serial.print("Setting LED ");
        Serial.print(pin);
        Serial.print(" to state ");
        Serial.println(leds_Status[pin]);
        leds_Status[pin] = 1 - leds_Status[pin];
        digitalWrite(leds_PinControl[pin], leds_Status[pin]);
    }
    return RPC_Response(data["pin"], (bool)data["enabled"]);
}

// RPC handlers
//=====
RPC_Callback callbacks[] = {
    {"setValue", processDelayChange },
    {"getValue", processGetDelay },
    {"setGpioStatus", processSetGpioState },
};

```

JSON

Set

Pin { 0,1,2 }

JSON

```

// Part 3 of 3
// _ConnectWifi.h

//=====
void WiFi_Initial() {
    Serial.println("Connecting to AP ..."); // attempt to connect to WiFi network
    WiFi.begin(WIFI_AP_NAME, WIFI_PASSWORD);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("\nConnected to AP");
    Serial.print("Local IP = ");
    Serial.println(WiFi.localIP());
}

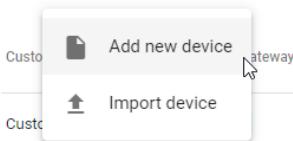
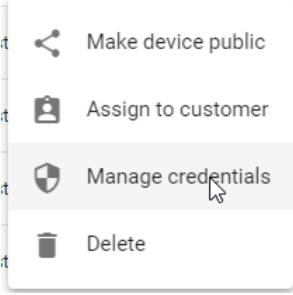
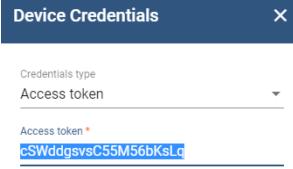
//=====
void reconnect() {
    status = WiFi.status(); // Loop until we're reconnected
    if (status != WL_CONNECTED) {
        WiFi.begin(WIFI_AP_NAME, WIFI_PASSWORD);
        while (WiFi.status() != WL_CONNECTED) {
            delay(500);
            Serial.print(".");
        }
        Serial.println("\nConnected to AP");
        Serial.print("Local IP = ");
        Serial.println(WiFi.localIP());
    }
}

```

no wifi

Minta D: re connect

LED-2 > ESP32_DHT22_GPIO			
Name	Date modified	Type	Size
_ConnectWifi.h	5/23/2021 11:41 PM	H File	1 KB
_ThingBoardRPC.h	5/24/2021 12:37 AM	H File	3 KB
ESP32_DHT22_GPIO	5/24/2021 12:36 AM	Arduino file	4 KB

18.1 Add New Device	
18.2 Get Token Key	
18.3 Copy Token Key	

19. ปรับปรุงและตั้งค่า Dashboard ดังรูป ประกอบด้วย Widget 4 ชนิด จำนวน 6 ตัว

Time Series

Monitor

Pk007_DHT22_GPIO
Device details

Latest telemetry

Last update time	Key ↑	Value
2021-05-24 12:39:23	humidity	41
2021-05-24 12:39:23	stsLED_0	0
2021-05-24 12:39:23	stsLED_1	0
2021-05-24 12:39:23	stsLED_2	0
2021-05-24 12:39:25	temperature	22

1/4 Add Widget 1

- เลือกข้อมูล humidity และ temper.
- ชนิด Chart → timeseries Line Chart
- Add to dashboard
- New → Pk007-Dashboad

Add widget to dashboard

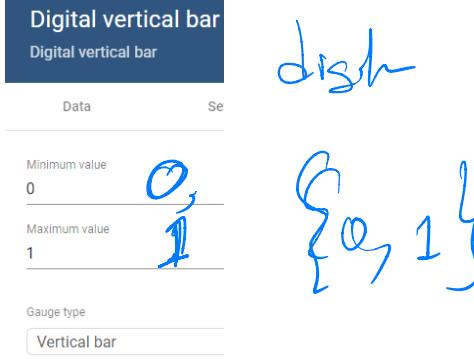
Select existing dashboard
Dashboard

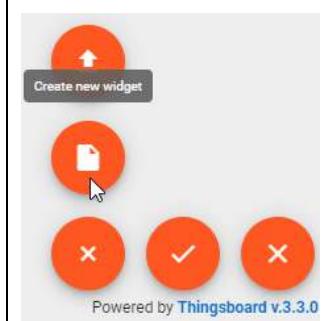
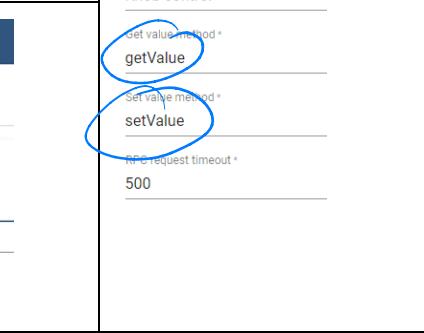
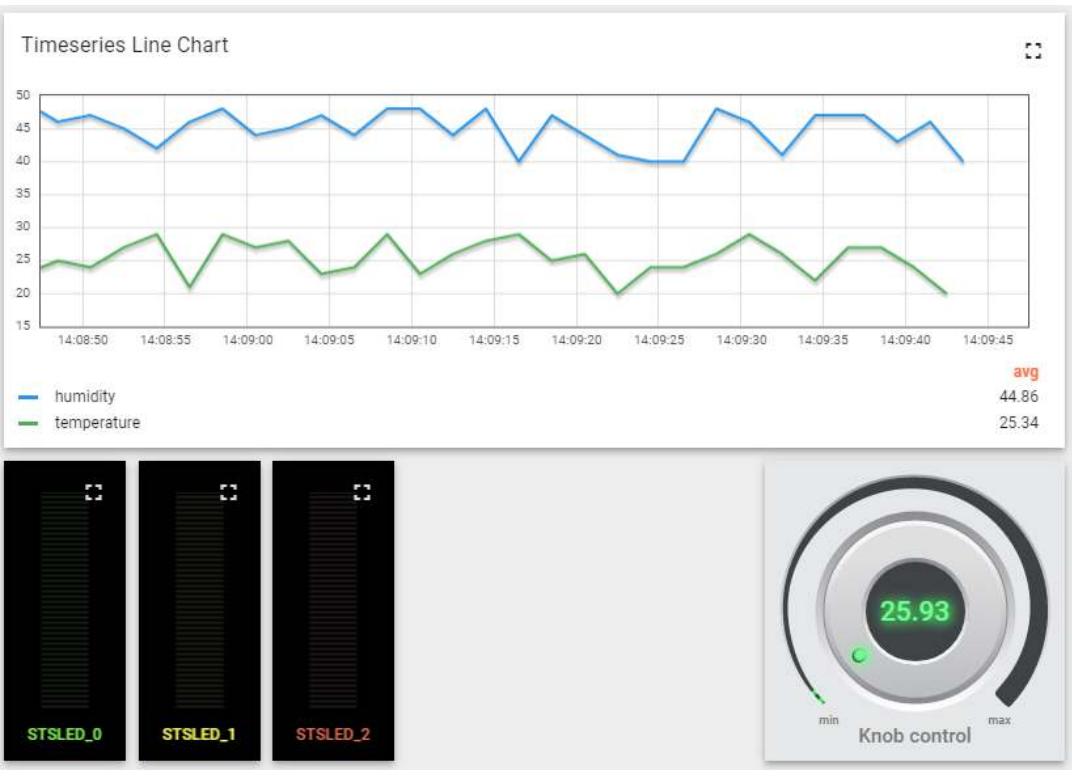
Create new dashboard
New dashboard title *
Pk007-Dashboard

Open dashboard Cancel **Add**

• กดปากกาเพื่อปรับตำแหน่ง และขนาด

Powered by [Thingsboard v.3.3.0](#)

<p>2/4 Add Widget 2,3,4</p> <ul style="list-style-type: none"> เลือกข้อมูล STSLED_0 ชนิด Digital Gauge → digital vertical bar Add to dashboard Select → Pk007-Dashboard 	
<ul style="list-style-type: none"> แก้ไขค่า Min,Max = 0, 1 แก้ไขสี Default Color เป็น โภนเขียว 	 <div style="display: flex; justify-content: space-between;"> Default color Background color of t rgb(32, 205, 72) </div> <p><input type="checkbox"/> Use precise val...</p>
<ul style="list-style-type: none"> Copy สำหรับ 3 LED ปรับ โภนสีเป็น Grn, Ylw, Red Ylw-Data เป็น stsLED_1 และ Red-Data เป็น stsLED_2 	 <p>Maximum 1 timeseries/attribute is allowed.</p>
	<p>Key * stsLED_1</p> <p>Label * stsLED_1</p> <p>Color * #f3f021</p> 
	

<p>3/4 Add Widget 5</p> <ul style="list-style-type: none"> • ติด Widget • เปิด Pk007-Dashboard • กดติดสอ • เลือก Create New Widget • เลือก Control Widget • เลือก Knob Control 	 <p>Powered by Thingsboard v.3.3.0</p>	<h3>New Knob Control</h3> <p>Knob Control</p> <table border="1"> <tr> <td>Data</td> <td>Setting</td> </tr> <tr> <td>Minimum value *</td> <td>0</td> </tr> <tr> <td>Maximum value *</td> <td>1000</td> </tr> <tr> <td>Initial value</td> <td>250</td> </tr> <tr> <td>Knob title</td> <td>Knob control</td> </tr> <tr> <td>Get value method *</td> <td>getValue</td> </tr> <tr> <td>Set value method *</td> <td>setValue</td> </tr> <tr> <td>RPC request timeout *</td> <td>500</td> </tr> </table>	Data	Setting	Minimum value *	0	Maximum value *	1000	Initial value	250	Knob title	Knob control	Get value method *	getValue	Set value method *	setValue	RPC request timeout *	500
Data	Setting																	
Minimum value *	0																	
Maximum value *	1000																	
Initial value	250																	
Knob title	Knob control																	
Get value method *	getValue																	
Set value method *	setValue																	
RPC request timeout *	500																	
<ul style="list-style-type: none"> • กำหนด Data เป็น Device ที่สร้างขึ้นมา • กำหนดค่าต่างๆ ดังรูปขวากลุ่ม • ปรับ Knob สั่งเกตการทำงานของ LED D2 	<p>Add Widget: Knob Control</p> <p>Data Settings Advanced</p> <p>Target device Pk007_DHT22_GPIO</p>																	
<p>Timeseries Line Chart</p>  <p>humidity</p> <p>temperature</p> <p>avg</p> <p>44.86</p> <p>25.34</p> <p>STSLED_0</p> <p>STSLED_1</p> <p>STSLED_2</p> <p>Knob control</p>																		

Cd p

4/4 Add Widget 6	ESP32 GPIO Control	Advance Setting Pin 0, GPIO 21,R 0, C 0 Pin 1, GPIO 22,R 1, C 0 Pin 2, GPIO 23,R 2, C 0
<ul style="list-style-type: none"> ตี Widget เปิด Pk007-Dashboard กดดินสอ เลือก Create New Widget เลือก GPIO Widget เลือก GPIO Control 		<i>Index Area</i> <i>Label</i> <i>Row</i> <i>Col</i>

Pk007-Dashboard

Timeseries Line Chart

humidity (avg 44.43)
temperature (avg 24.6)

ESP32 GPIO Control

GPIO 21 0
GPIO 22 1
GPIO 23 2

Knob control

20. สามารถ Export Dashboard เพื่อเก็บไว้ใช้งานใน Server อื่นๆ ได้



Mission 2/4: ให้ส่งข้อมูลค่า Humidity และ Temperatures จากเซ็นเซอร์ DHT-22 ไปยัง ThingsBoard พร้อมทั้งควบคุม On/Off - 4 LED และ Blink Speed สำหรับอีก 1 LED



Q

Q

Q

Lab203 – ThingsBoard Monitor and Control with MQTT Protocol

<https://thingsboard.io/docs/samples/esp8266/gpio/><https://blog.thingsboard.io/2017/01/esp8266-gpio-control-over-mqtt-using.html>

21. ปรับปรุง Arduino Library

- PubSubClient by Nick O'Leary. Ver 2.6 and Update PubSubClient.h
- ArduinoJSON library by Benoit Blanchon Ver 5.8.0

22. แก้ไข PubSubClient.h ให้สามารถใช้งานกับ ESP32 ด้วยการแก้ไขไฟล์

C:\Users\Pk007_20210404\Documents\Arduino\libraries\PubSubClient\src\PubSubClient.h

MQTT

// #ifdef ESP8266	Comment	#ifdef ESP8266
#if defined (ESP8266) defined(ESP32)	Insert	#if defined (ESP8266) defined(ESP32)

23. โหลดโปรแกรมไปยัง ESP32 (โปรแกรมแยกเป็น 3 ไฟล์) อย่าลืมไขข้อมูลเฉพาะบุคคล และต่อวงจรเพื่อใช้งาน

```

// File 1 of 3
// https://thingsboard.io/docs/samples/esp8266/gpio/
// https://blog.thingsboard.io/2017/01/esp8266-gpio-control-over-mqtt-using.html

#include <WiFi.h>
#include <ArduinoJson.h> // by Benoit Blanchon >> Ver 5.8.0
#include <PubSubClient.h> // by Nick O'Leary. >> Ver 2.6 and Update PubSubClient.h
// replace #ifdef ESP8266
// to #if defined (ESP8266) || defined(ESP32)

#define WIFI_AP_NAME    "Test1234"
#define WIFI_PASSWORD   "0816601929"
#define Device_Name     "Pk007_DHT22_GPIO"
#define Device_Token    "cSWddgsvsC55M56bKsLq"
#define thingsboardServer "demo.thingsboard.io"

#define GPIO1_ESP32Pin 22
#define GPIO2_ESP32Pin 23
boolean gpioState[] = {false, false};
int status = WL_IDLE_STATUS;

WiFiClient wifiClient;
PubSubClient client(wifiClient);

#include "_HandOnMQTT.h"
#include "_WifiConnect.h"

void setup() {
  Serial.begin(115200);
  // Set output mode for all GPIO pins
  pinMode(GPIO1_ESP32Pin, OUTPUT);
  pinMode(GPIO2_ESP32Pin, OUTPUT);
  delay(10);
  InitialWiFi();
  client.setServer( thingsboardServer, 1883 );
  client.setCallback(on_message);
}

void loop() {
  if ( !client.connected() ) {
    reconnect();
  }
  client.loop();
}

```

```

// File 2 of 3
// _HandOnMQTT.h
//=====
//=====
String get_gpio_status() {
    // Prepare gpios JSON payload string
    StaticJsonBuffer<200> jsonBuffer;
    JsonObject & data = jsonBuffer.createObject();
    data[String(GPIO1_ESP32Pin)] = gpioState[0];
    data[String(GPIO2_ESP32Pin)] = gpioState[1];
    char payload[256];
    data.printTo(payload, sizeof(payload));
    String strPayload = String(payload);
    Serial.print("Get GPIO Status: ");
    Serial.println(strPayload);
    return strPayload;
}

//=====
//=====
void set_gpio_status(int pin, boolean enabled) {
    if (pin == GPIO1_ESP32Pin) {
        gpioState[0] = 1 - gpioState[0];
        digitalWrite(GPIO1_ESP32Pin, gpioState[0]);
    }
    if (pin == GPIO2_ESP32Pin) {
        gpioState[1] = 1 - gpioState[1];
        digitalWrite(GPIO2_ESP32Pin, gpioState[1]);
    }
}

//=====
//=====
// The callback for when a PUBLISH message is received from the server.
void on_message(const char* topic, byte* payload, unsigned int length) {
    Serial.println("\nOn message");
    char json[length + 1];
    strncpy (json, (char*)payload, length);
    json[length] = '\0';
    Serial.print("Topic: "); Serial.println(topic);
    Serial.print("Message: "); Serial.println(json);

    // Decode JSON request
    StaticJsonBuffer<200> jsonBuffer;
    JsonObject& data = jsonBuffer.parseObject((char*)json);

    if (!data.success()) {
        Serial.println("parseObject() failed");
        return;
    }

    // Check request method
    String methodName = String((const char*)data["method"]);

    // If Reply with GPIO status
    if (methodName.equals("getGPIOStatus")) {
        String responseTopic = String(topic);
        responseTopic.replace("request", "response");
        client.publish(responseTopic.c_str(), get_gpio_status().c_str());
    }

    // If Update GPIO status and reply
    if (methodName.equals("setGPIOStatus")) {
        set_gpio_status(data["params"]["pin"], data["params"]["enabled"]);
        String responseTopic = String(topic);
        responseTopic.replace("request", "response");
        client.publish(responseTopic.c_str(), get_gpio_status().c_str());
        client.publish("v1/devices/me/attributes", get_gpio_status().c_str());
    }
}

```

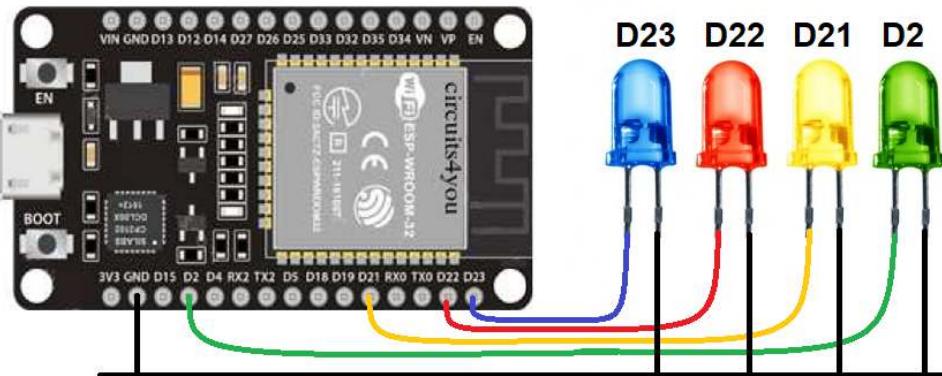
```

// File 3 of 3
// _WifiConnect.h
//=====
//=====
void InitialWiFi() {
    Serial.println("Connecting to AP ...");
    WiFi.begin(WIFI_AP_NAME, WIFI_PASSWORD);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("Connected to AP");
}

//=====
//=====
void reconnect() {
    // Loop until we're reconnected
    while (!client.connected()) {
        status = WiFi.status();
        if (status != WL_CONNECTED) {
            InitialWiFi();
        }
        Serial.print("Connecting to ThingsBoard node ...");
        // Attempt to connect (clientid, username, password)
        if (client.connect(Device_Name, Device_Token, NULL)) {
            Serial.println(" [DONE] ");
            // Subscribing to receive RPC requests
            client.subscribe("v1/devices/me/rpc/request/+");
            // Sending current GPIO status
            Serial.println("Sending current GPIO status ...");
            client.publish("v1/devices/me/attributes", get_gpio_status().c_str());
        } else {
            Serial.print(" [FAILED] [ rc = ");
            Serial.print( client.state() );
            Serial.println( " : retrying in 5 seconds]" );
            delay( 5000 ); // Wait 5 seconds before retrying
        }
    }
}

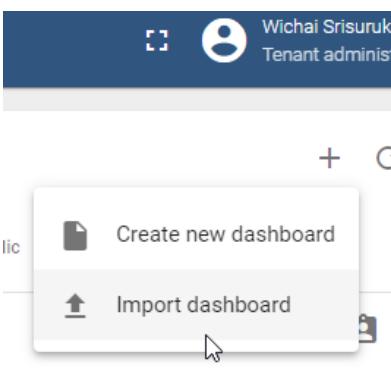
```

_HandOnMQTT.h	5/24/2021 11:17 PM
_WifiConnect.h	5/24/2021 10:58 PM
ESP32_Mission3_GPIO	5/24/2021 11:17 PM

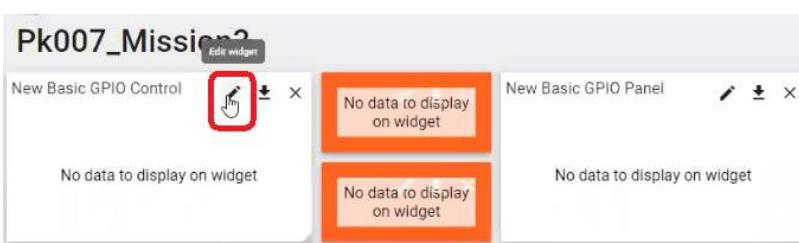
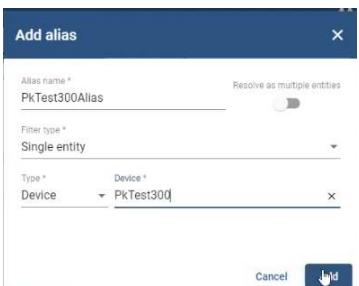


23. สร้าง Device ใหม่ให้มีชื่อเป็น “PkTest300” และ Copy Token Key เพื่อใส่ใน Arduino Code

24. ตั้งค่า Dashboard ให้เป็นดังรูป ด้วยวิธีการ Import Dashboard

	<p>Dashboard</p> <p>Add → Import dashboard</p> <p>เลือกไฟล์ “pk007_mission3.json”</p>
---	---

25. Add Alias เพื่อตั้งค่าตัวแปรใน Device มาแสดง ด้วยการเข้า Edit Mode และ Setting

	<p>Edit Mode Setting</p>
	<p>PkTest300Alias Single entity Device PkTest300</p>
	<p>เปลี่ยน parameter ทั้ง 4 Device Entity → Pk300Alias</p>

26. ผลการทำงาน

Pk007_Mission3

Pk007_Mission3 ▾

Pk007_DHT22_GPIO

Realtime

New Basic GPIO Control

GPIO 22 22

GPIO 23 23

22 false

23 false

New Basic GPIO Panel

GPIO 22 22

GPIO 23 23

22 true

23 false

22 false

23 false

27. ทดสอบการทำงาน

```

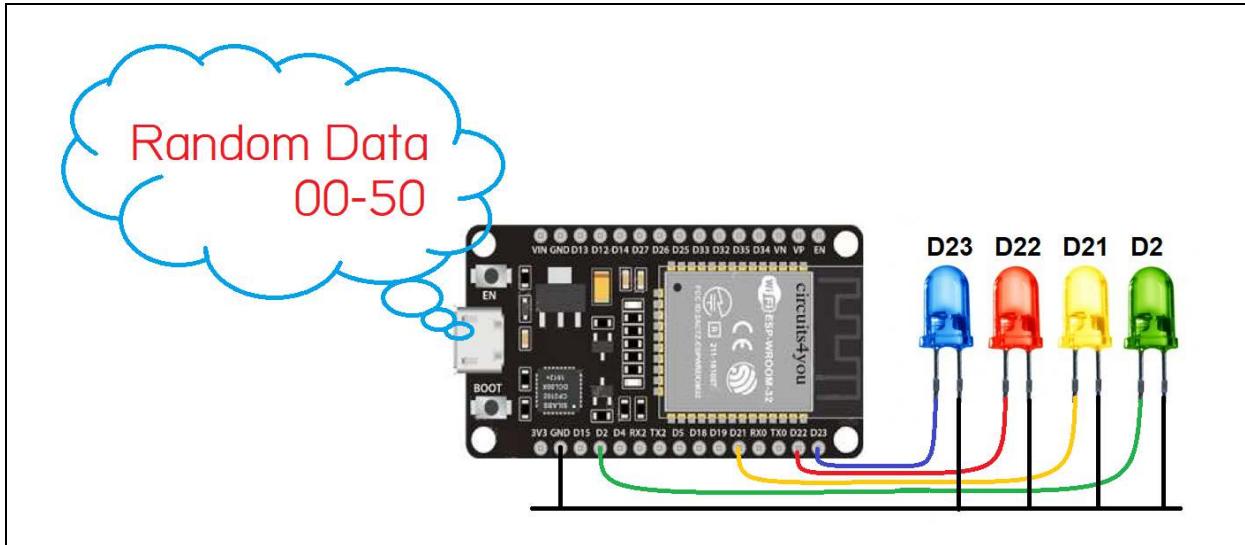
Connecting to AP ...
.....Connected to AP
Connecting to ThingsBoard node ... [DONE]
Sending current GPIO status ...
Get GPIO Status: {"22":false,"23":false}

On message
Topic: v1/devices/me/rpc/request/159
Message: {"method":"setGpioStatus","params":{"pin":22,"enabled":false}}
Get GPIO Status: {"22":true,"23":false}
Get GPIO Status: {"22":true,"23":false}

```

Mission 3/4: ให้ใช้ MQTT กับ ThingsBoard

1. ปรับปรุงเพื่อให้ทำงานควบคุมการ On/Off - 4 LED
2. เพิ่มเติม คือ ทดสอบส่งข้อมูล 1 ค่าแบบสุ่มระหว่าง 00 – 50 ไปแสดงที่ Dashboard ด้วย ได้หรือไม่ ทำอย่างไรบ้างให้อธิบาย {Read <https://thingsboard.io/docs/user-guide/device-profiles/>}



5/6 -- โครงการ “ตรวจวัดอุณหภูมิความชื้นของฟาร์มไก่ ให้มีการควบคุมอุปกรณ์และมีการแจ้งเตือน”

<https://www.smethailandclub.com/startups-6416-id.html>

<https://thingsboard.io/docs/getting-started-guides/helloworld/>

หัวข้อนี้จะเป็นการใช้งานเพื่อทดสอบการส่งข้อมูลจาก ESP-32 Device ไปยัง ThingsBoard.IO กำหนดให้เป็นโครงการเรื่อง “ตรวจวัดอุณหภูมิความชื้นของฟาร์มไก่ ให้มีการควบคุมอุปกรณ์และมีการแจ้งเตือน” โดยจะมีขั้นตอนการทำงาน ดังนี้

1. สร้าง Device
2. จัดการ Device ผ่าน Token
3. สร้าง Dashboard และส่งข้อมูลไปยัง Device
4. กำหนดเงื่อนไขการแจ้งเตือน
5. สร้างการแจ้งเตือน
6. กำหนดช่องทางการแจ้งเตือน
7. สร้างและแชร์ Dashboard เพื่อแสดงค่าที่ส่งจาก ESP32 Board

Lab204 – Smart Poultry House via ThingsBoard

Step 1. Provision Device

1. Login <https://demo.thingsboard.io/login>
2. สร้าง Device Name = My_Kai

Click Device	Add New Device

Step 2. Connect device

3. Create Token

The screenshot shows the ThingsBoard Device Management interface. On the left, a list of devices is displayed, with 'Ma_Kai' selected and highlighted by a red box. In the center, a detailed view of 'Ma_Kai' is shown with tabs for 'Details', 'Attributes', 'Latest telemetry', 'Alarms', 'Events', 'Relations', and 'Audit Log'. The 'Details' tab is active. A red box highlights the 'Copy access token' button under the 'Actions' section. A green notification at the bottom right states 'Device access token has been copied to clipboard'. The bottom navigation bar includes links for 'System Settings', '2021-05-25 12:28:06 PK007Exp01_MQTT', and '2021-09-15 22:40:08 Test Device A3'.

4. Test with curl Online Command

```
curl -v -X POST -d "{\"temperature\": 25}" $HOST_NAME/api/v1/$ACCESS_TOKEN/telemetry --header "Content-Type:application/json"
```

```
curl -v -X POST -d "{\"temperature\": 25}" https://demo.thingsboard.io/api/v1/$ACCESS_TOKEN/telemetry --header "Content-Type:application/json"
```

5. Goto >> <https://reqbin.com/curl>

```
curl -v -X POST -d "{\"temperature\": 25}" https://demo.thingsboard.io/api/v1/EMCttqEvfFLvgTRfQLB1/telemetry --header "Content-Type:application/json"
```

The screenshot shows the ReqBin website interface. At the top, there's a navigation bar with links like 'REQBIN', 'Request Examples', and 'Articles & Tutorials'. Below the navigation, a main heading says 'Run Curl Commands Online' with the sub-instruction 'Execute Curl commands directly from your browser. Learn Curl with live Curl examples. Test APIs with ReqBi'. There are dropdown menus for 'File', 'Generate Code', and 'Tools'. Below these are buttons for 'Curl', 'Raw', 'US', and 'Run'. A code editor window contains the curl command: 'curl -v -X POST -d "{\"temperature\": 25}" https://demo.thingsboard.io/api/v1/EMCttqEvfFLvgTRfQLB1/telemetry --header "Content-Type:application/json"'.

6. Result

The screenshot shows the ThingsBoard interface for a device named 'My_Kai'. The 'Latest telemetry' tab is active, displaying a single data point: 'temperature' with a value of '25'.

Last update time	Key ↑	Value
2021-07-31 15:06:42	temperature	25

7. ESP32 Coding – Random temperature generate

Check Arduino Library

- ThingsBoard Arduino SDK Ver 0.2.0
- PubSubClient by Nick O'Leary Ver 2.7.0
- ArduinoHttpClient libraries. Ver 0.3.2
- ArduinoJSON library Ver 6.9.1

```
#include "ThingsBoard.h"
#include <WiFi.h>

#define WIFI_AP           "Test1234"
#define WIFI_PASSWORD     "0816601929"
#define TOKEN             "EMCttqEvffLvgTRfQLB1"
#define THINGSBOARD_SERVER "demo.thingsboard.io"

WiFiClient espClient;
ThingsBoard tb(espClient);
int status = WL_IDLE_STATUS;

void setup() {
  Serial.begin(115200);
  WiFi.begin(WIFI_AP, WIFI_PASSWORD);
  InitWiFi();
}

void loop() {
  delay(1000);
  if (WiFi.status() != WL_CONNECTED) {
    reconnect();
  }

  if (!tb.connected()) {
    // Connect to the ThingsBoard
    Serial.print("Connecting to: ");
    Serial.print(THINGSBOARD_SERVER);
    Serial.print(" with token ");
    Serial.println(TOKEN);
    if (!tb.connect(THINGSBOARD_SERVER, TOKEN)) {
      Serial.println("Failed to connect");
      return;
    }
  }
}
```

```

}

Serial.println("\nSending data...");
// Uploads new telemetry to ThingsBoard using MQTT.
// See https://thingsboard.io/docs/reference/mqtt-api/#telemetry-upload-api
// for more details
float Tempp = random(2000, 6000) / 100.0;
Serial.println("Tempp = " + String(Tempp, 2) + "'C");
tb.sendTelemetryInt("temperature", Tempp);
//tb.sendTelemetryFloat("humidity", 42.5);
tb.loop();
}

void InitWiFi() {
Serial.println("Connecting to AP ...");
WiFi.begin(WIFI_AP, WIFI_PASSWORD);
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
Serial.println("Connected to AP");
}

void reconnect() {
status = WiFi.status();
if (status != WL_CONNECTED) {
WiFi.begin(WIFI_AP, WIFI_PASSWORD);
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
Serial.println("Connected to AP");
}
}

```

.....Connected to AP
Connecting to: demo.thingsboard.io with token EMCttqEvfFLvgTRfQLB1

Sending data...
Tempp = 54.07'C

Sending data...
Tempp = 51.67'C

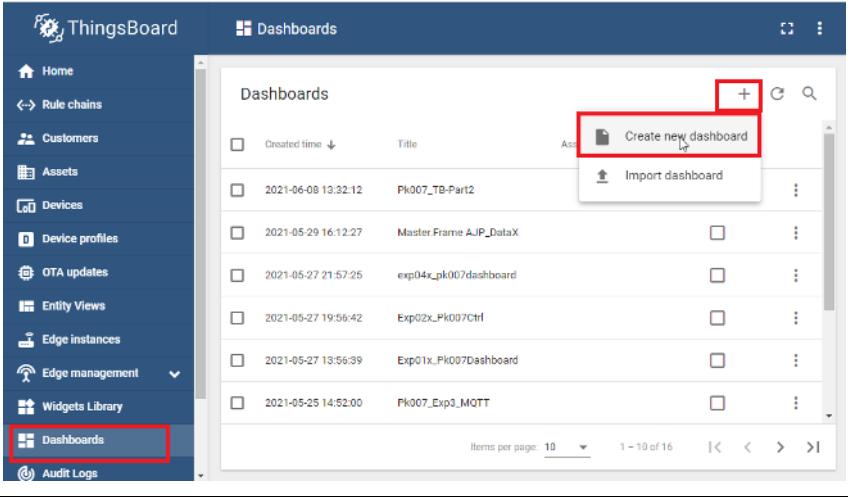
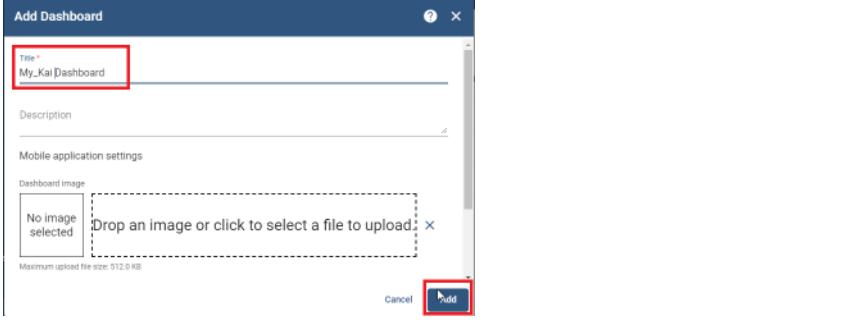
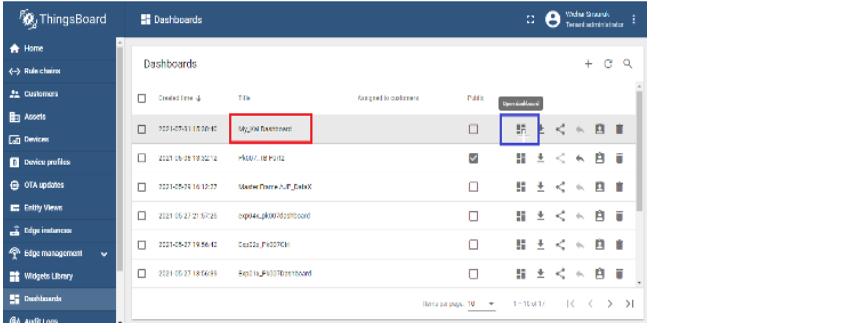
Sending data...
Tempp = 40.54'C

Sending data...
Tempp = 20.80'C

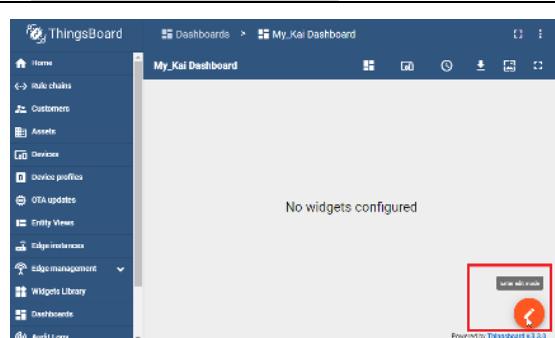
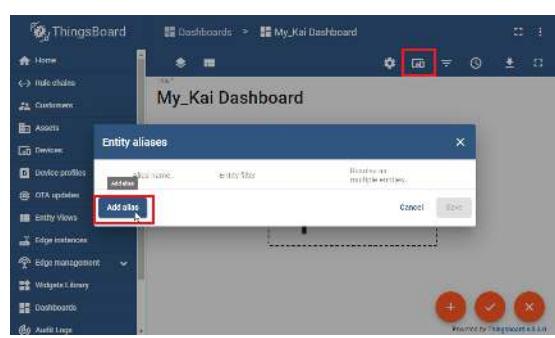
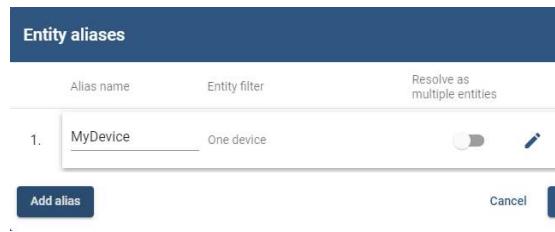
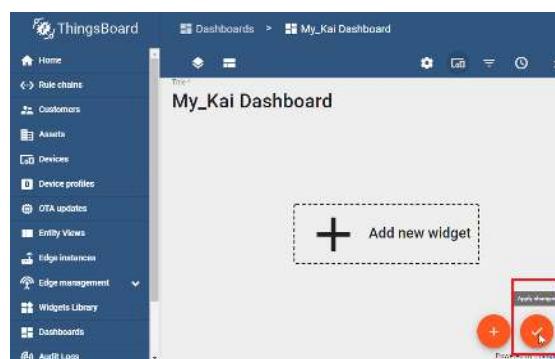
Autoscroll Show timestamp Carriage return ▾ 115200 baud ▾ Clear output

Step 3. Create Dashboard

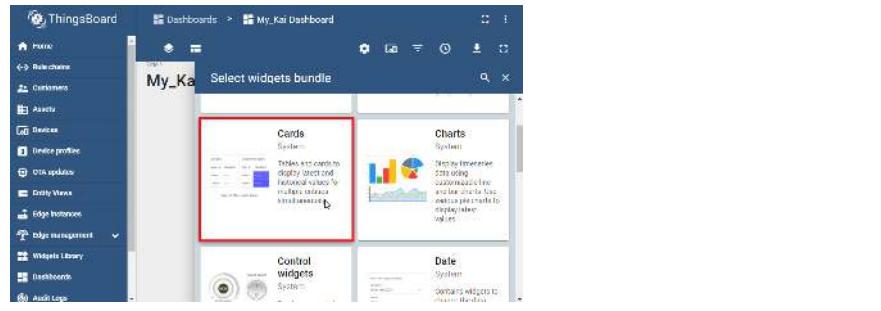
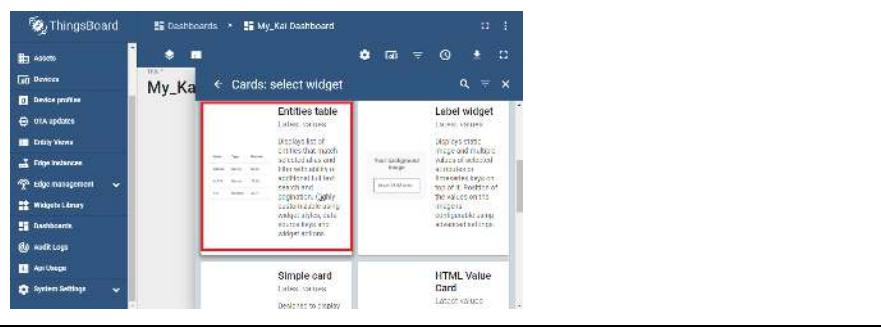
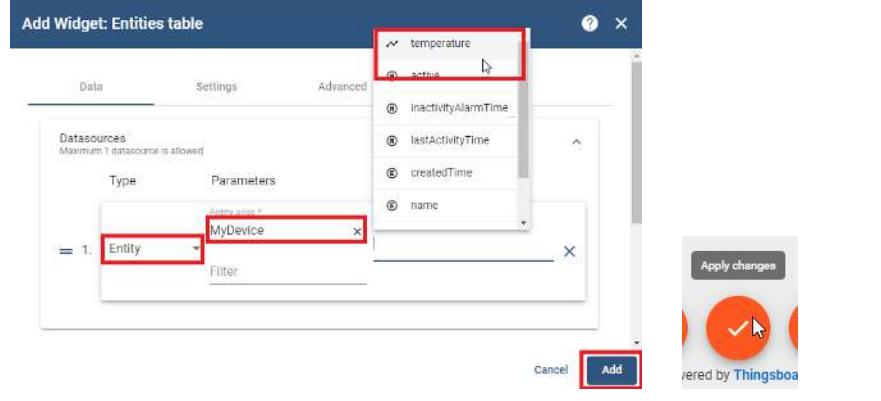
Step 3.1 Create Empty Dashboard

	<p>Create new dashboard</p>
	<p>Name = My_Kai Dashboard</p>
	<p>Goto Dashboard</p>

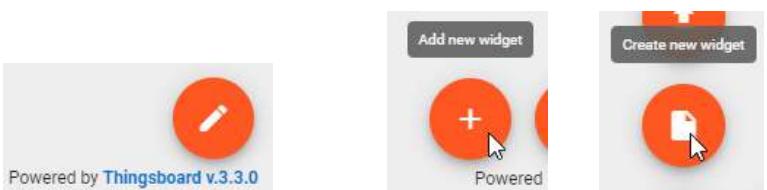
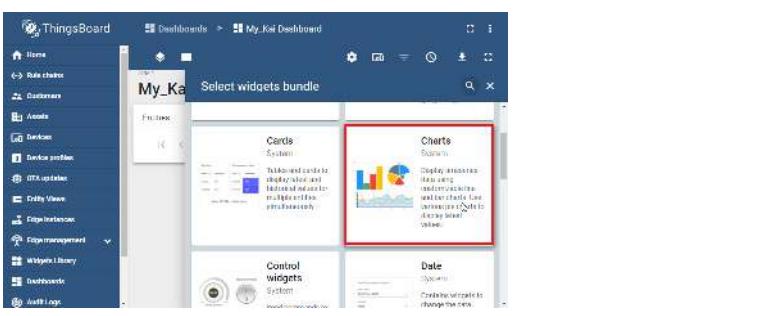
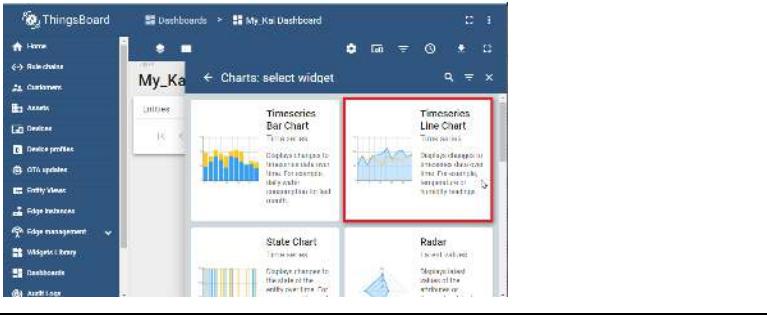
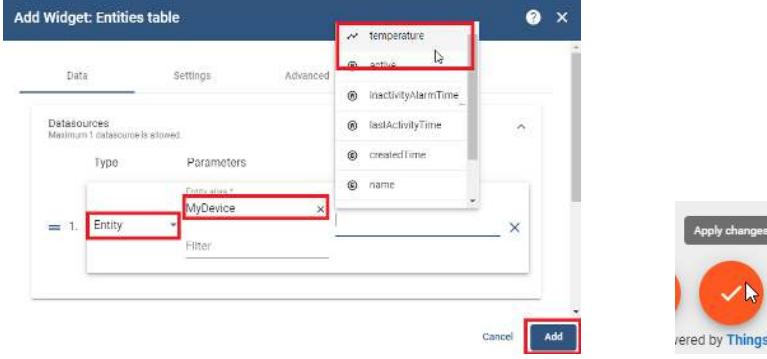
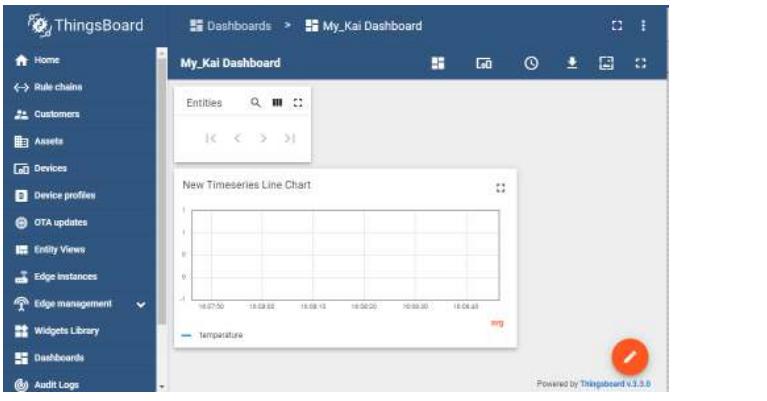
Step 3.2 Add Entity Alias

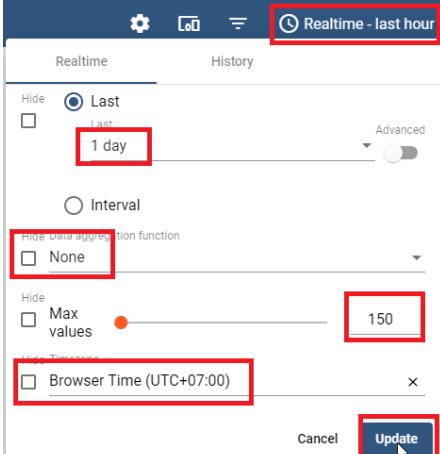
	Edit
	Add Alias
	<p>MyDevice</p> <p>Single entity</p> <p>Device</p> <p>My_Kai</p> <p>My_Kai</p>
	Save
	Apply

Step 3.3 Add Table Widget

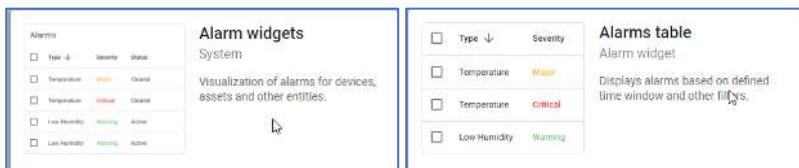
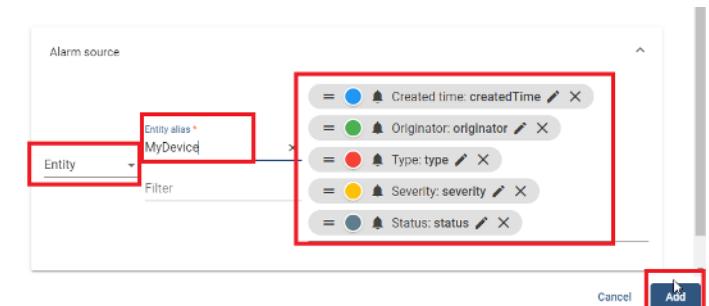
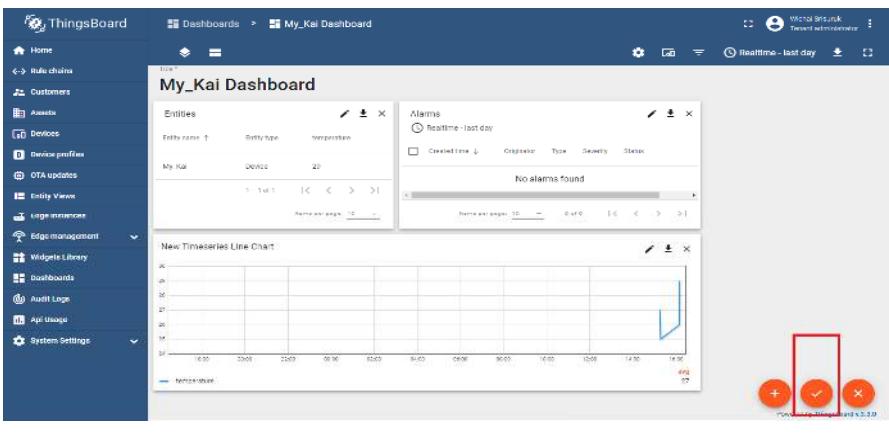
	Edit Add New Widget
	Cards
	Entities Table
	+Add Entity → MyDevice → temperature Add Apply

Step 3.4 Add Chart Widget

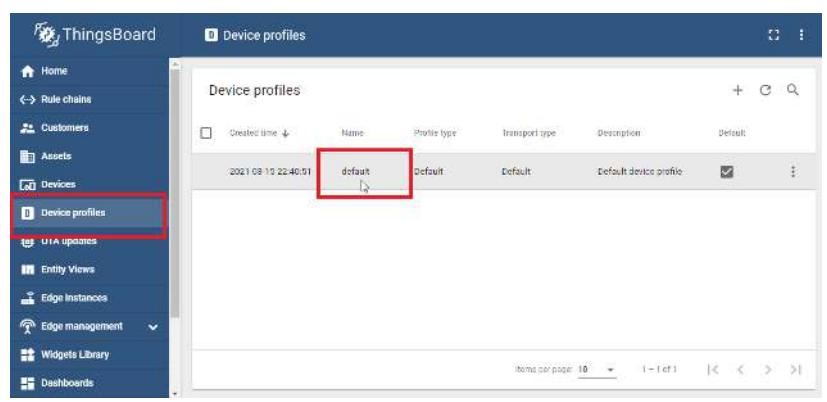
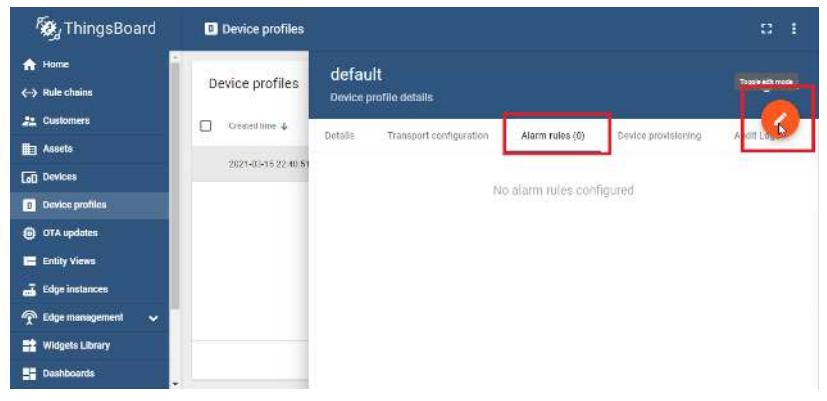
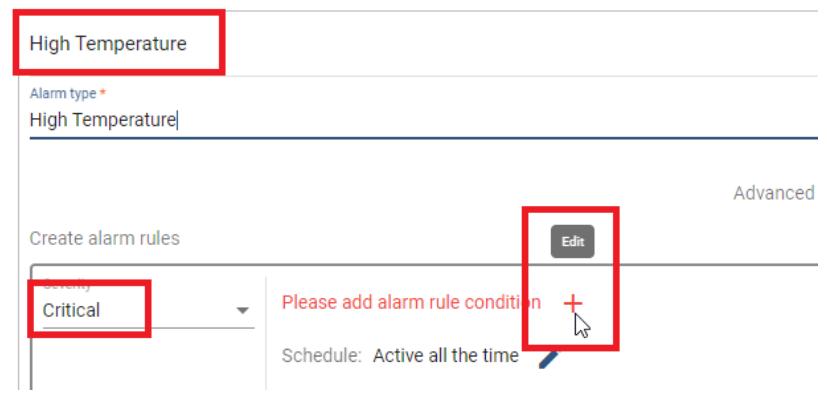
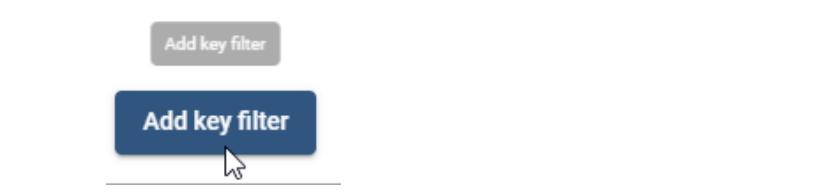
	Edit Add new widget Create new widget
	Charts
	Timeseries Line Chart
	+Add Entity → MyDevice → temperature Add Apply
	

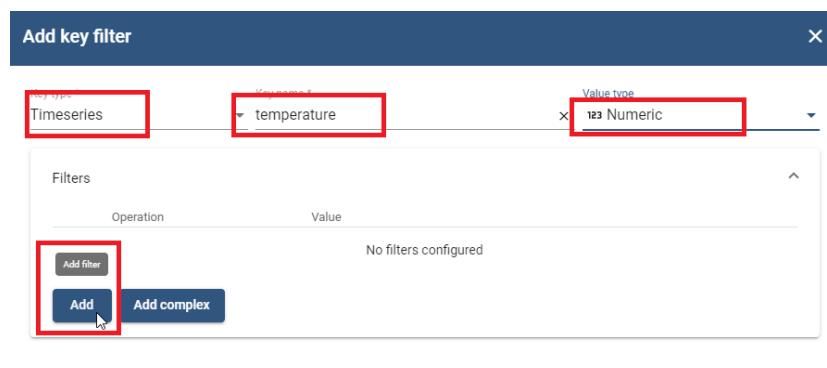
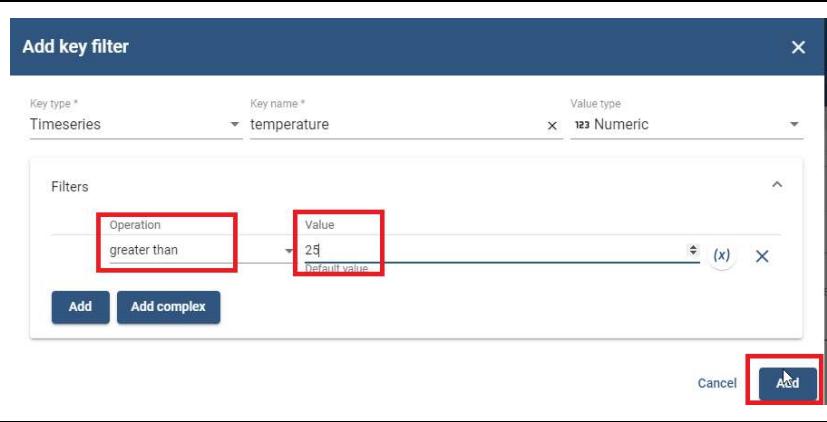
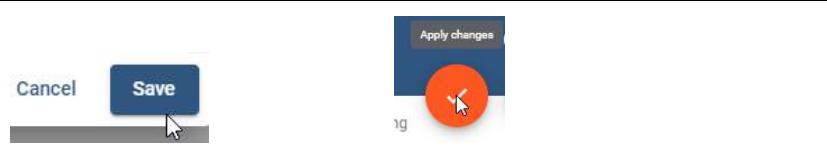
	<p>Realtime – last hour</p> <p>1-day</p> <p>None</p> <p>Max = 150</p> <p>Time(UTC+07:00)</p> <p>Update</p>
---	--

Step 3.5 Add Alarm Widget

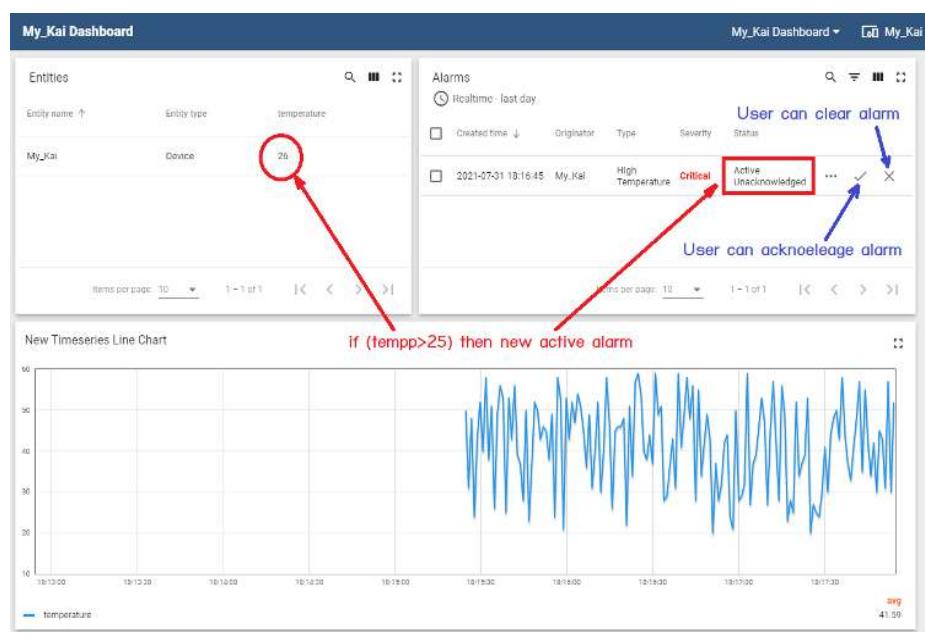
	<p>Add Alarm widget</p> <p>→ Alarm table</p>
	<p>Entity</p> <p>Device → MyDevice</p> <p>...</p> <p>Add</p>
	Apply

Step 4. Configure Alarm Rules

	Device profiles Default
	Alarm rules Edit
	Add alarm rule
	High Temperature Critical Edit
	Add key filter

 <p>The screenshot shows the 'Add key filter' dialog. At the top, 'Key type' is set to 'Timeseries', 'Key name' to 'temperature', and 'Value type' to '123 Numeric'. Below this, a 'Filters' section is shown with an 'Add filter' button highlighted with a red box. The 'Add' button at the bottom left is also highlighted.</p>	<p>Timeseries Temperature Numeric Add</p>
 <p>The screenshot shows the 'Add key filter' dialog with a more complex filter. The 'Operation' dropdown is set to 'greater than' and the 'Value' input field contains '25'. The 'Add' button at the bottom right is highlighted with a red box.</p>	<p>Greater than 25 Add</p>
 <p>The screenshot shows the 'Apply changes' dialog with two buttons: 'Save' and 'Apply'. Both buttons are highlighted with red boxes. A cursor is hovering over the 'Apply' button.</p>	<p>Save Apply</p>

Step 5. Create Alarm

 <p>The screenshot shows the 'My_Kai Dashboard'. On the left, there's a table of entities with one row for 'My_Kai' (Device) having a value of '26'. In the center, an 'Alarms' section shows a single active alarm for 'High Temperature' with severity 'Critical'. The status is 'Active' and 'Unacknowledged'. There are buttons for 'User can clear alarm' and 'User can acknowledge alarm'. On the right, a 'New Timeseries Line Chart' displays a fluctuating blue line representing temperature over time, with a value of '41.59' at the end.</p>	
---	--

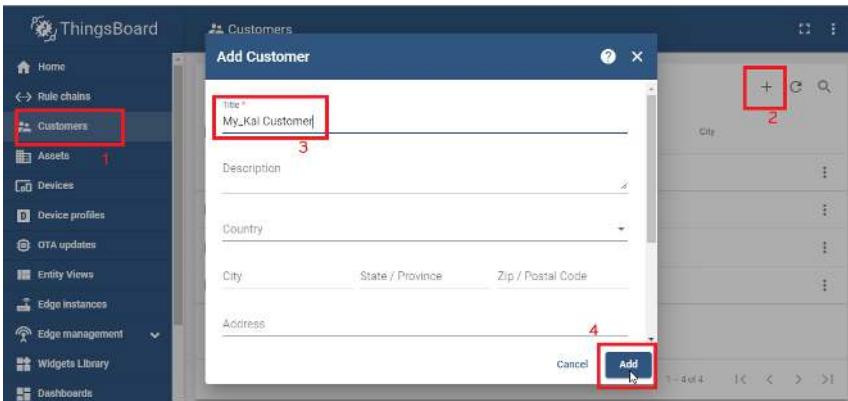
Step 6. Alarm notifications

It is quite easy to configure email or sms notifications for alarms. We recommend reviewing alarm rule [examples](#) and documentation about [alarm notifications](#).

Note: At the moment ThingsBoard supports AWS SNS and Twilio to send SMS. Both services are non-free and require you to create an account. However, you may integrate with other SMS/EMAIL gateways using [REST API call](#) node.

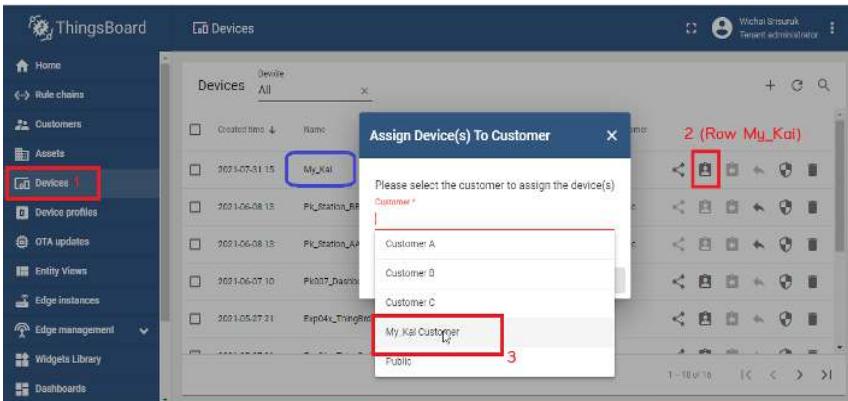
Step 7. Assign Device and Dashboard to Customer

Step 7.1 Create customer



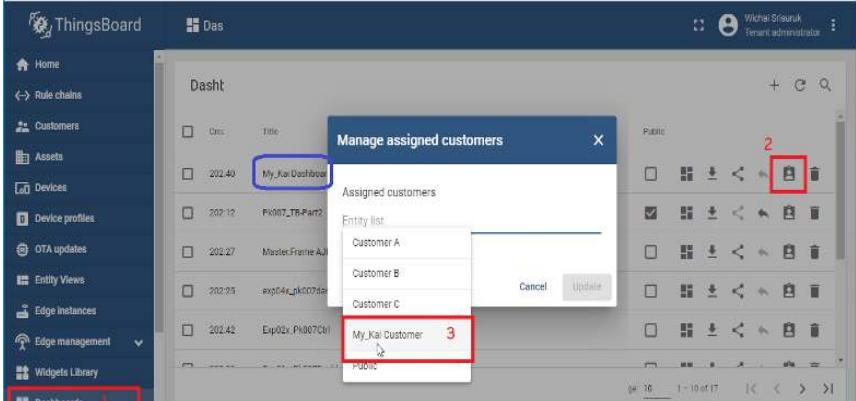
1. Customers
2. (+) Add Customers
3. Name = MyKai Customer
4. Add

Step 7.2 Assign device to Customer

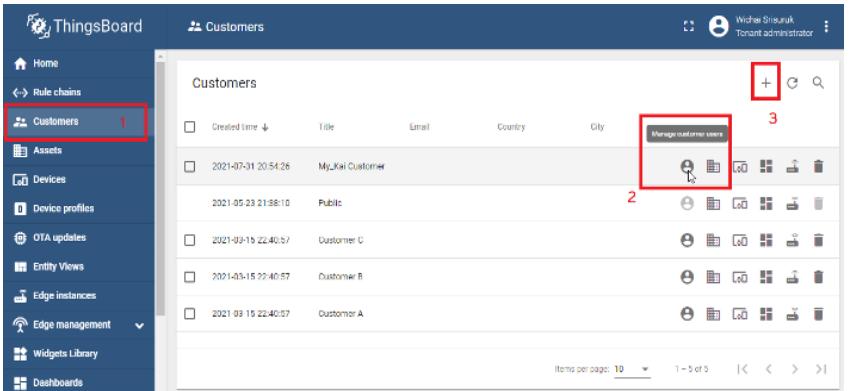
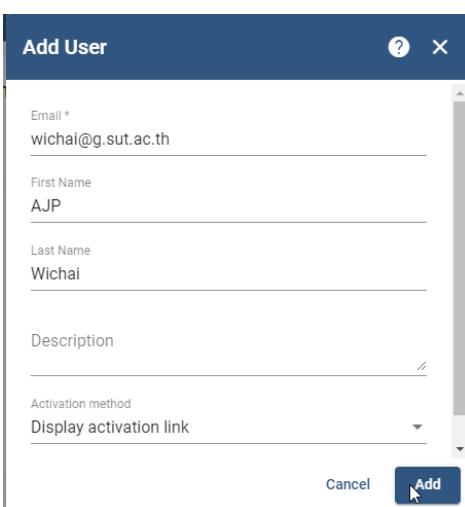


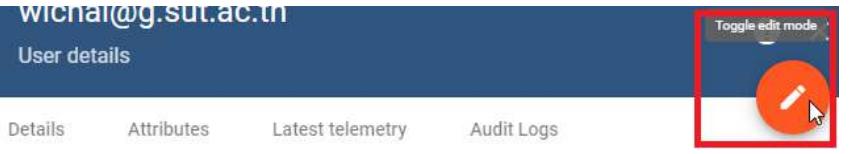
1. Device Tab
2. On row MyKai → Assign Device(s) to Customer
3. Select MyKai Customer
4. Assign

Step 7.3 Assign dashboard to Customer

	<ol style="list-style-type: none"> Dashboard Tab On row MyKai Dashboard → Assign Device(s) to Customer Select MyKai Customer
<p>Cancel</p> <p>Update</p>	4. Update

Step 7.4 Create customer user

	<ol style="list-style-type: none"> Customers Tab Manage customer user (+)Add
	<ol style="list-style-type: none"> Fill Email First Name Last Name Add

	Copy activate link
https://demo.thingsboard.io/api/noauth/activate?activateToken=H5Zrr8KykvzH8WsGnuoN4OJYGM2mq	
	Click to open user detail
	Edit Mode
	1. Allow access MyKai Dashboard 2. Hide toolbar 3. Apply Change

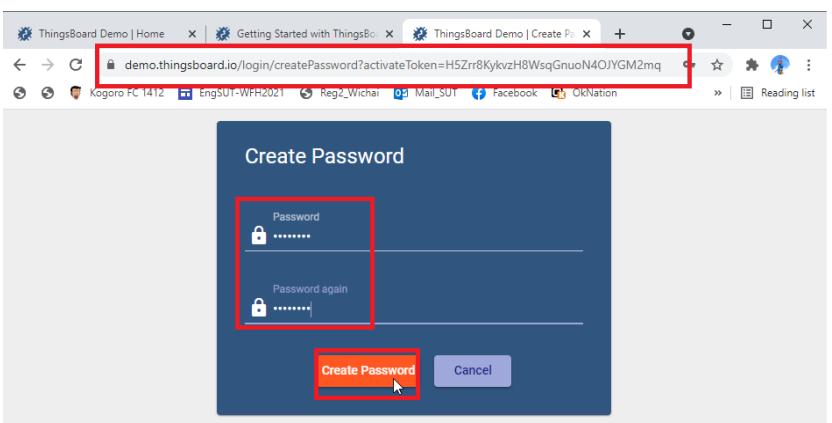
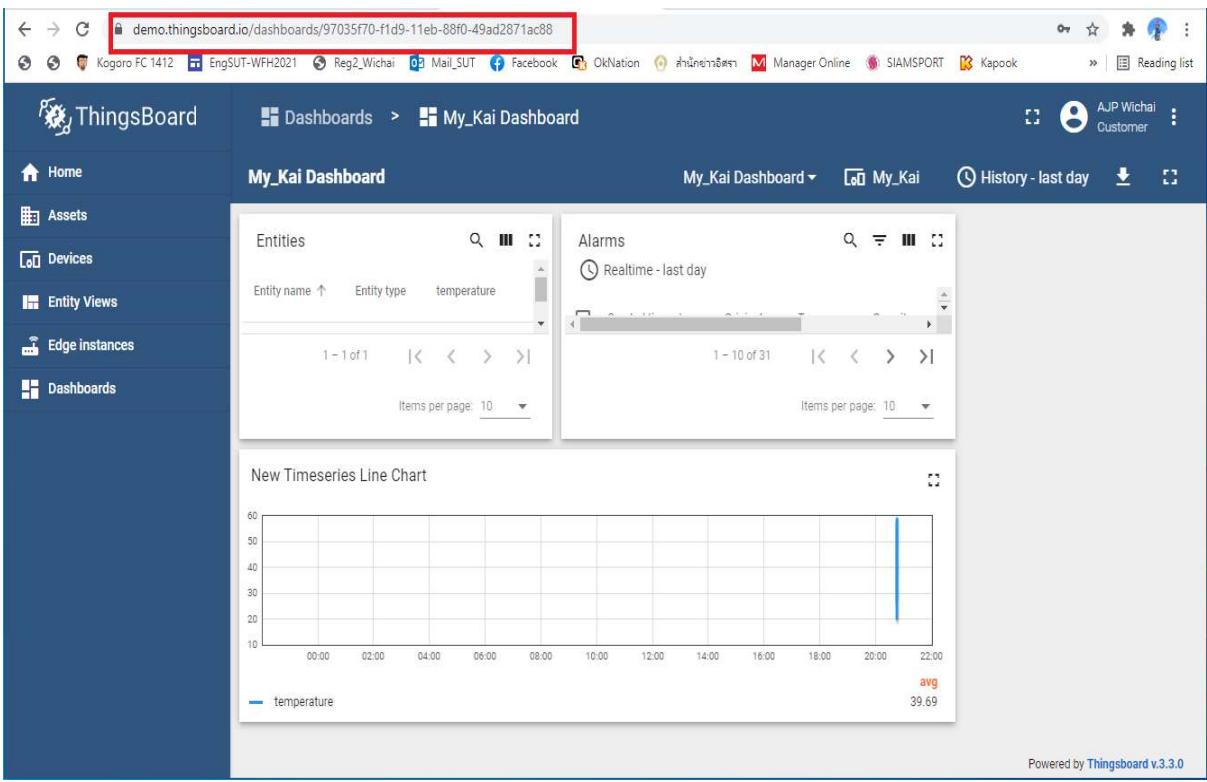
Step 7.5 Activate customer user

1. Open web browser

2. Insert activate link url

3. Fill password {12345678}

4. Create password

Dashboard URL = <https://demo.thingsboard.io/dashboards/97035f70-f1d9-11eb-88f0-49ad2871ac88>

การใช้งาน ThingsBoard IoTs Platform เพื่อสร้างและจัดการระบบอัตโนมัติ
ThingsBoard IoTs Platform for smart system

ข้อ-สกุล :

6/6 -- คำถ้ามท้ายบทเพื่อทดสอบความเข้าใจ

Quiz_101 – ThingsBoard Data Monitor

- Mission - 1/4: ให้ส่งข้อมูลค่า Humidity และ Temperatures จากเซ็นเซอร์ DHT-22 ไปยัง Dashboard

โปรแกรมที่ใช้ทดสอบ



Capture Dashboard



รูปการทดสอบ 1



รูปการทดสอบ 2

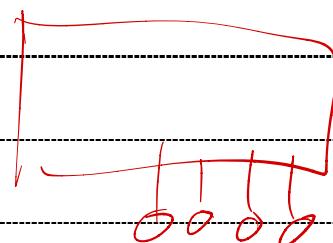
Quiz_102 – ThingsBoard Data Monitor and Control

- Mission 2/4: ให้ส่งข้อมูลค่า Humidity และ Temperatures จากเซ็นเซอร์ DHT-22 ไปยัง ThingsBoard พร้อมทั้งควบคุม On/Off - 4 LED และ Blink Speed สำหรับอีก 1 LED

โปรแกรมที่ใช้ทดสอบ

Capture Dashboard

รูปถ่ายหน้า Web Browser



รูปการทดสอบ 1

รูปการทดสอบ 2

Quiz_103 – ThingsBoard Data Monitor and control with MQTT Protocol

- Mission 3/4: ให้ใช้ MQTT กับ ThingsBoard
 - ปรับปรุงเพื่อให้ทำงานควบคุมการ On/Off - 4 LED
 - เพิ่มเติม คือ ทดสอบส่งข้อมูล 1 ค่าแบบสุ่มระหว่าง 00 – 50 ไปแสดงที่ Dashboard ด้วย ได้หรือไม่ ทำอย่างไรบ้างให้อธิบาย {Read <https://thingsboard.io/docs/user-guide/device-profiles/>}

โปรแกรมที่ใช้ทดสอบ	
Capture Dashboard	
รูปถ่ายหน้า Web Browser	
รูปการทดสอบ 1	
รูปการทดสอบ 2	

Quiz_104 – Web Control 4 LED and Monitor Humid/Temperature

- Mission 4/4: การตรวจสอบและควบคุม อุณหภูมิ-ความชื้น ของโรงเรือนเดี๋ยงไก่
 - ให้ใช้ ESP32 ส่งข้อมูลแบบสุ่มสองจำนวน คือ
 - Temp_A สุ่มระหว่าง 20-40
 - Hudmid_A สุ่มระหว่าง 60-80
 - ข้อมูลทั้งสองค่าจะนำมาแสดงที่ Dashboard
 - สร้าง Alarm โดย หาก Temp_A > 35 หรือ Hudmid_A > 70 ให้ Alarm
 - ศึกษาการตั้ง Alarm - <https://thingsboard.io/docs/user-guide/alarms/>
 - กำหนดรอบการตรวจสอบทุกๆ 20 วินาที
 - แชร์ Dashboard ไปให้ผู้ใช้งาน

โปรแกรมที่ใช้ทดสอบ	
Capture Dashboard	
รูปถ่ายหน้า Web Browser	
รูปการทดสอบ 1	
รูปการทดสอบ 2	