บทที่ 11
พัฒนางาน IoT กับ WIO Terminal โดยใช้ Blynk IoT App

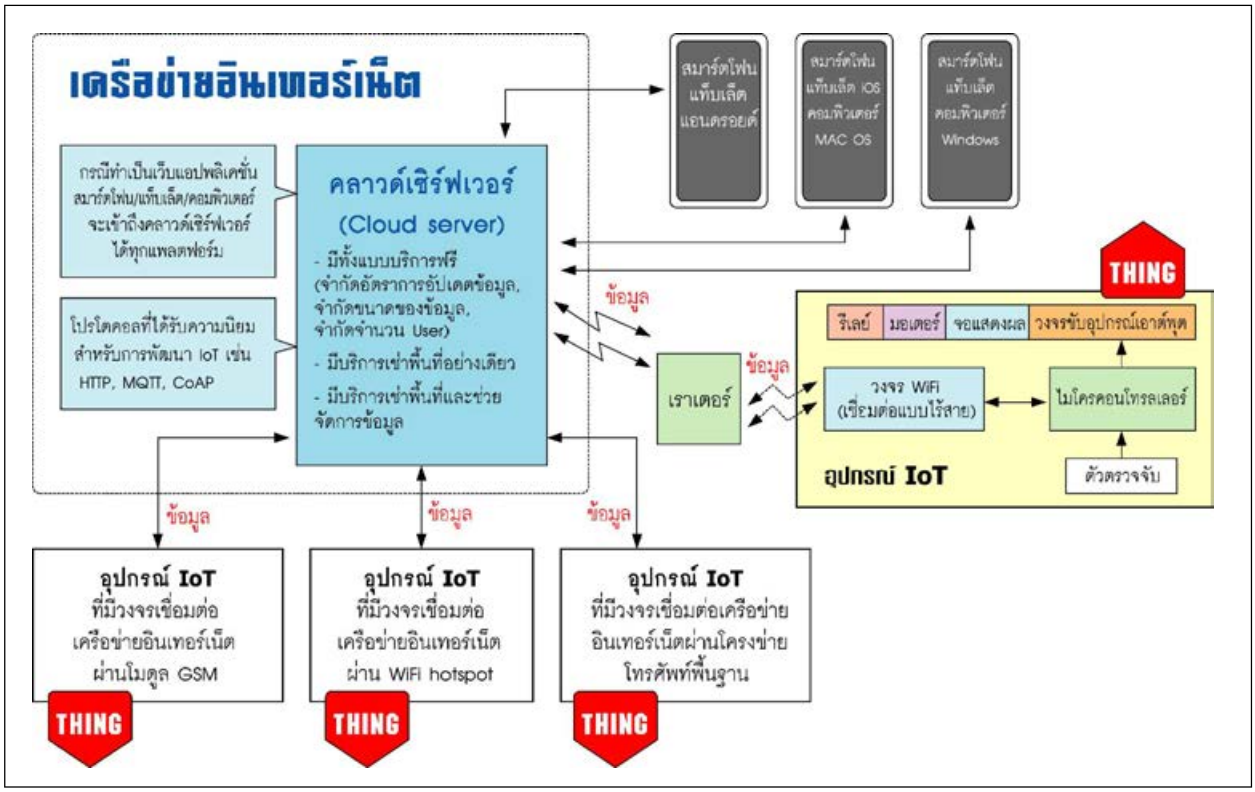
การพัฒนางาน IoT (Internet of Things) เพื่อใช้งานสมาร์ตโฟนหรือแท็บเล็ตควบคุมอุปกรณ์เครื่องใช้ฟ้า แสดงค่าของตัวตรวจจับทางไกลผ่านเครือข่ายอินเทอร์เน็ต เป็นหนึ่งในการเรียนรู้ที่ผู้เรียนระบบสมองกลฝังตัวสมัยใหม่ควรให้ความสำคัญ ในบทนี้นำเสนอถึงแนวทางและตัวอย่างการพัฒนาอุปกรณ์ IoT ที่ใช้ WIO Terminal เป็นส่วนประกอบหลักร่วมกับแอปพลิเคชั่น **Blynk** เพื่อช่วยให้เกิดการพัฒนาอุปกรณ์ IoT ได้อย่างรวดเร็ว และเหมาะสำหรับผู้เริ่มต้นที่ต้องการไปต่อยอดใช้งานจริง

11.1 รู้จักกับ IoT

Internet of Things คำนี้เกิดขึ้นมาตั้งแต่ปี ค.ศ. 1999 โดย **Kevin Ashton** แห่ง MIT's Media center เขาได้นำเสนอแนวคิดว่า *IoT คือ การนำสิ่งของต่างๆ ไม่ว่าจะเป็นคอมพิวเตอร์, เครื่องจักร และตัวตรวจจับมาเชื่อมต่อกับเครือข่ายอินเทอร์เน็ต เพื่อรายงานสถานะการทำงาน ข้อมูล และรับรู้คำสั่งควบคุม*

IoT หรือ Internet of Things หมายถึง เทคโนโลยีที่ก่อให้เกิดการเชื่อมโยงกันของสิ่งของ ผู้คน ข้อมูล และการบริการเข้ากับเครือข่ายอินเทอร์เน็ต ปัจจัยสำคัญในการทำให้เกิด IoT ได้คือ การบรรจุอุปกรณ์สมองกลฝังตัวหรือ embedded system device เข้าไปใน "สิ่งของ" หรือเครื่องมือ เครื่องใช้ต่างๆ มีตัวตรวจจับหรือเซนเซอร์เพื่อตรวจวัดค่าที่สนใจ แล้วส่งมายังส่วนสมองกล เพื่อส่งต่อมายังส่วนประมวลผลกลางและฐานข้อมูลผ่านเครือข่ายอินเทอร์เน็ต ในส่วนหลังนี้มีชื่อเรียกด้วยศัพท์สมัยใหม่ว่า **คลาวด์เซิรฟเวอร์ (cloud server)**

ด้วยการนำอุปกรณ์สมองกลฝังตัวบรรจุลงใน "สิ่งของ" ต่างๆ ทำให้ "สิ่งของ" เหล่านั้นทำงานในแบบอัจฉริยะได้ อุปกรณ์เครื่องใช้ต่างๆ ในบ้าน ในโรงงาน ในที่ทำงาน ในยานพาหนะ ล้วนแล้วแต่ใช้ระบบสมองกลฝังตัวมากขึ้น ทำให้มันทำงานได้ด้วยตัวเอง และ/หรือรวมเข้าเป็นส่วนหนึ่งของระบบใหญ่เกิดการเชื่อมโยงการทำงานเป็นระบบได้



รูปที่ 1 1–1 ส่วนประกอบและการทำงานของ IoT เบื้องต้น

ส่วนประกอบของ IoT มีดังนี้

1. สิ่งของ
2. อุปกรณ์ (ตัวควบคุม, ตัวตรวจจับ และอุปกรณ์ขับโหลดหรืออุปกรณ์เอาต์พุต)
3. ระบบเชื่อมต่ออินเทอร์เน็ต (จะเป็นแบบมีสายหรือไร้สายก็ได้)
4. ข้อมูล
5. ระบบจัดการฐานข้อมูลคลาวด์เซิรฟเวอร์ (Cloud server)

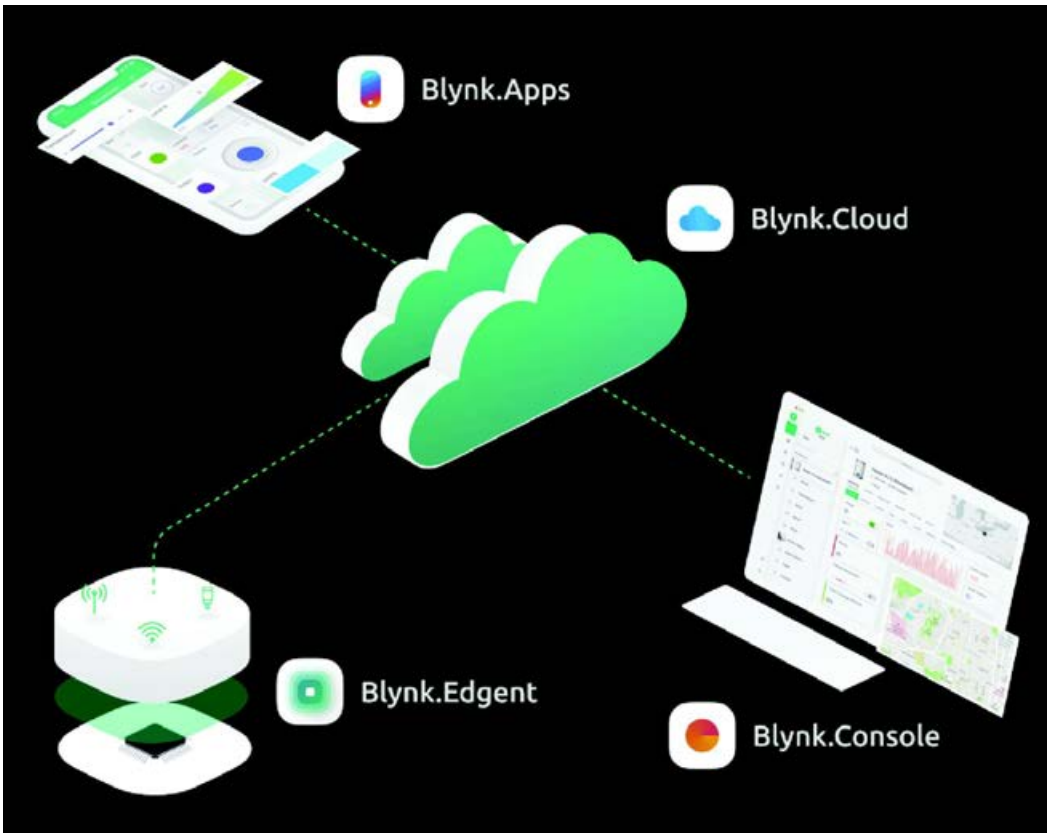
การทำให้“สิ่งของ”ทำงานร่วมกันผ่านเครือข่ายอินเทอร์เน็ต จึงทำให้เกิดนิยามของเทคโนโลยีนี้ขึ้น Internet of Things หรือ IoT เป็นการขยายขอบเขตการทำงานของอินเทอร์เน็ตให้กว้างและลึกลงไปถึงการเชื่อมต่อเพื่อสื่อสารข้อมูลกับ “สิ่งของ” ทำให้เกิดการรับส่งข้อมูลและตอบสนองในแบบทุกที่ ทุกเวลา และทุกสิ่งของได้ในที่สุด

11.2 มารู้จักกับ Blynk2 หรือ Blynk IoT

ปัจจุบันนี้การพัฒนางาน IoT หรือ Internet of Things ที่มีการใช้งานสมาร์ตโฟนหรือแท็บเล็ตควบคุมอุปกรณ์เครื่องใช้ฟ้าแสดงค่าของตัวตรวจจับทางไกลผ่านเครือข่ายอินเทอร์เน็ตทำได้ง่ายและรวดเร็ว เนื่องจากมีตัวช่วยที่ดี เด่น และฟรี ในชื่อ **Blynk App**

Blynk อ่านว่า“บลิ้ง” คือ ชุดของแอปพลิเคชั่นที่ทำให้การสร้างงาน IoT ทำได้ง่ายอย่างเบ็ดเสร็จ มีการเชื่อมต่อกับอุปกรณ์ที่อยู่ไกลผ่านเครือข่ายอินเทอร์เน็ต โดยใช้สมาร์ตโฟนเป็นอุปกรณ์หลักในการติดต่อกับผู้ใช้งานและอุปกรณ์ควบคุมปลายทาง ผู้พัฒนา IoT ไม่ต้องจัดเตรียมโปรแกรมอื่นๆ ไม่ว่าจะเป็นเว็บเซิร์ฟเวอร์, หน้าเว็บแสดงผลและควบคุม รวมถึงซอฟต์แวร์เพื่อการเชื่อมต่อใดๆ

Blynk เป็นผลงานของกลุ่มคนรุ่นใหม่ที่เสนอโครงการเข้าใน Kickstarter โดยผู้ก่อตั้งคือ **Mr. Pavel Baiborodin** โดยระดมทุนได้ 49,000 เหรียญสหรัฐ (US$) Blynk ได้เปิดตัวใช้งานตั้งแต่เดือนพฤษภาคม ค.ศ. 2015



รูปที่ 11-2 แสดงส่วนประกอบต่าง ๆ Blynk ในเวอร์ชันใหม่ เพื่อช่วยให้การพัฒนางาน IoT เป็นไปได้ สำหรับทุกคนที่สนใจ โดยไม่จำกัดอยู่ที่สมาร์ตโฟนเท่านั้นอีกต่อไป ผู้ใช้งานสามารถติดต่อผ่านทางเว็บไซต์โดยใช้อุปกรณ์ใดก็ได้ไม่ว่าจะเป็นสมาร์ตโฟน แท็บเล็ต หรือคอมพิวเตอร์

รูปที่ 11-3 ภาพแสดงความง่ายของการพัฒนางาน IoT กับ Blynk IoT App

Blynk ช่วยให้การพัฒนางาน IoT ง่าย โดยผู้พัฒนาไม่จำเป็นที่ต้องมีความรู้ด้านระบบฮาร์ดแวร์และซอฟต์แวร์ทางคอมพิวเตอร์มากมาย ตัวแอปพลิเคชั่นใช้งานกับฮาร์ดแวร์ที่เป็นที่นิยม ทั้งบอร์ด Arduino, ESP8266, ESP32, Raspberry Pi, SAMD21 เป็นต้น

ปัจจุบัน Blynk ได้พัฒนามาเป็นเวอร์ชัน 2 ในตอนแรกเรียกว่า **Blynk2** ต่อมาทางผู้พัฒนาได้ปรับ Blynk App ในเวอร์ชันแรกเป็น **Legacy Blynk** และหยุดการให้บริการในปีพ.ศ. 2565 ส่วน Blynk2 ได้ถูกเรียกชื่อใหม่ว่า **Blynk IoT** แต่ในเว็บไซต์ของผู้พัฒนาจะเรียกชื่อเพียงว่า **Blynk** ดังนั้นในเอกสารนี้จะยึดชื่อของแพลตฟอร์มตามการอัปเดตล่าสุดของผู้พัฒนา

Blynk ในเวอร์ชันแรกได้รับความนิยมสูง เนื่องจากใช้งานง่าย เมื่อปรับปรุงเป็นเวอร์ชัน 2 มีการเปลี่ยนแปลงหลายอย่าง เช่น การสร้างและใช้งานแดชบอร์ด (dashboard) ผ่านเว็บไซต์ได้ การจำกัดจำนวนวิดเจ็ต (widget) ที่ใช้งานสำหรับรุ่นฟรีเป็น 30 ตัวแทนการให้ค่าพลังงาน และรูปแบบการตั้งค่าที่แตกต่างไปจากเดิม กอปรกับการหยุดให้บริการ Blynk เซิร์ฟเวอร์รุ่นเดิมในปี พ.ศ. 2565 การเรียนรู้เพื่อใช้งานกับ Blynk App ในเวอร์ชันใหม่จึงเป็นสิ่งที่นักเล่น นักทดลอง นักพัฒนา รวมถึงนักเรียน นิสิต นักศึกษาที่ใช้งาน Blynk ในการพัฒนาอุปกรณ์ IoT ควรดำเนินกา

11.2.1 ที่สุดของความง่ายในการใช้งาน Blynk

Blynk มีขั้นตอนในการพัฒนาที่สุดแสนง่ายใน 3 ขั้นตอน คือ

(1) **หยิบแล้ววาง (Drag and Drop)** คือ การหยิบอุปกรณ์ที่ต้องการมาวางบนพื้นที่ทำงาน ซึ่งก็คือ หน้าจอแสดงผลของสมาร์ตโฟน

(2) **ดาวน์โหลดไลบรารี (Download library)** คือ การดึงไฟล์สนับสนุนของ Blynk มาใส่ในโปรแกรม สำหรับในโปรแกรม Arduino IDE จะมีไลบรารีสำหรับใช้งานกับ Blynk โดยเฉพาะซึ่งผู้พัฒนาโค้ดสามารถดาวน์โหลดไปติดตั้งและใช้งานได้ฟรี

(3) **อัปโหลดโค้ด (Upload Sketch)** คือ การส่งโค้ดไปยังบอร์ดไมโครคอนโทรลเลอร์

นั่นคือขั้นตอนหลักๆ ที่ง่ายๆ สำหรับกระบวนการพัฒนาอุปกรณ์ IoT โดยใช้อุปกรณ์ฮาร์ดแวร์เป็นบอร์ดไมโครคอนโทรลเลอร์อย่างบอร์ด WIO Terminal ซึ่งใช้ไมโครคอนโทรลเลอร์ ATSAMD21 และพัฒนาโปรแกรมภาษา C/C++ ด้วย Arduino IDE ด้วยความสามารถของแอปพลิเคชั่น Blynk IoT ช่วยให้ผู้สนใจทุกคนสามารถพัฒนางาน IoT เองได้ โดยไม่จำเป็นต้องมีทักษะทางการเขียนโปรแกรมมากมาย

11.2.2 การพัฒนางาน IoT โดยใช้ Blynk

ประกอบด้วยอุปกรณ์ทั้งหมด 3 ส่วนคือ

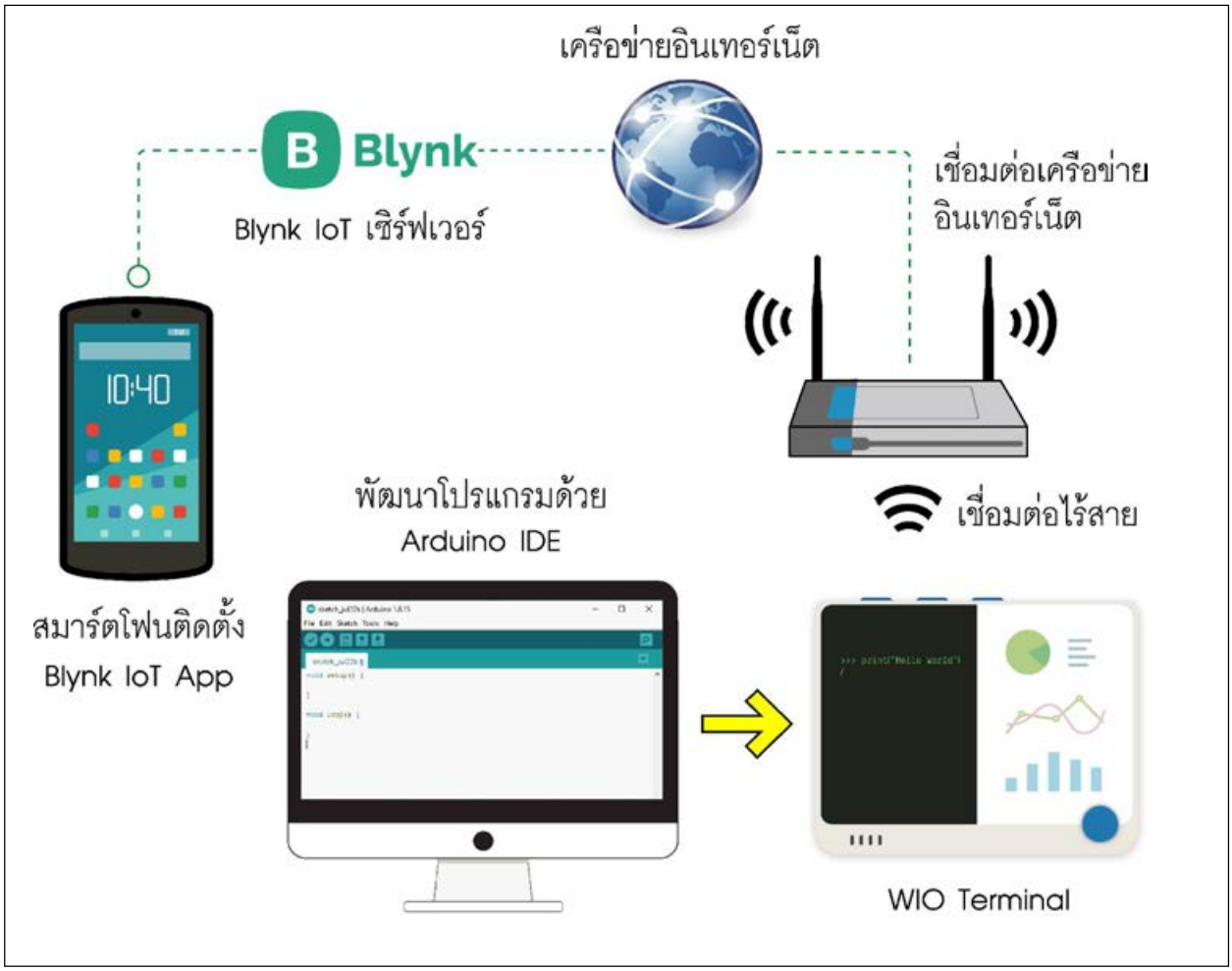
1. **Blynk.App** เป็นแอปพลิเคชั่นและเว็บเซิร์ฟเวอร์ที่ติดตั้งบนสมาร์ตโฟนหรือแท็บเล็ตทั้งบนระบบปฏิบัติการ iOS และ Android รวมถึงใช้งานผ่านเว็บไซต์ได้ เพื่อสร้างงานโดยใช้วิดเจ็ต (widgets) ที่จัดเตรียมให้

2. **Blynk IoT ไลบรารี** เป็นไลบรารีของคำสั่งเพิ่มเติมเพื่อพัฒนาโปรแกรมภาษา C/C++ สำหรับบอร์ดไมโครคอนโทรลเลอร์เพื่อติดต่อกับ Blynk.Cloud เซิร์ฟเวอร์

3. **ฮาร์ดแวร์** ในที่นี้เป็นกล่องสมองกล ***WIO Terminal*** ที่ได้รับการโปรแกรมให้ทำงานและติดต่อกับ Blynk เซิร์ฟเวอร์ รวมทั้งอุปกรณ์อินพุตเอาต์พุต

การทำงานในกรณีใช้งานกับสมาร์ตโฟนหรือแท็บเล็ต เมื่อมีการกดปุ่มบนหน้าจอของ Blynk app มันจะส่งข้อความคำสั่งผ่านอินเทอร์เน็ตไปยัง Blynk เซิร์ฟเวอร์ เพื่อส่งต่อไปยัง WIO Terminal เพื่อให้บอร์ดนำคำสั่งไปปฏิบัติตาม หรือทางกลับกันเมื่อมีการส่งค่าจาก WIO Terminal ไปยัง Blynk เซิร์ฟเวอร์เพื่อแสดงผลบนหน้าจอสมาร์ตโฟน

จากคำอธิบายเบื้องต้นพอจะอธิบายเป็นโครงสร้างได้ตามรูปที่ 11-4



รูปที่ 1 1–4 โครงสร้างการทำงานของระบบ IoT ที่ใช้ Blynk เซิร์ฟเวอร์ทำงานร่วมกับ WIO Terminal

11.3 ลงทะเบียนใช้งาน Blynk

11.3.1 ลงทะเบียนครั้งแรก

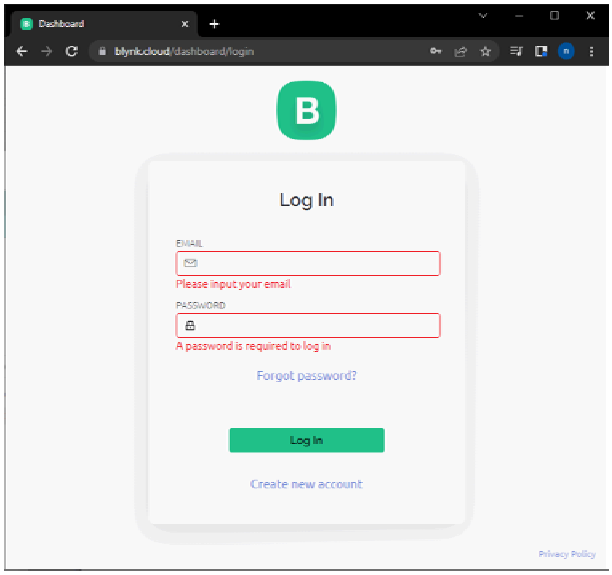
มีขั้นตอนดังนี้

(1) เชื่อมต่อคอมพิวเตอร์ที่ใช้ในการพัฒนาโปรแกรมเข้ากับเครือข่ายอินเทอร์เน็ต จากนั้นเปิดเว็บบราวเซอร์เพื่อเข้าไปยังเว็บเพจสำหรับเข้าสู่การใช้งานดังรูปที่ 11-5 ตามลิงก์ต่อไปนี้

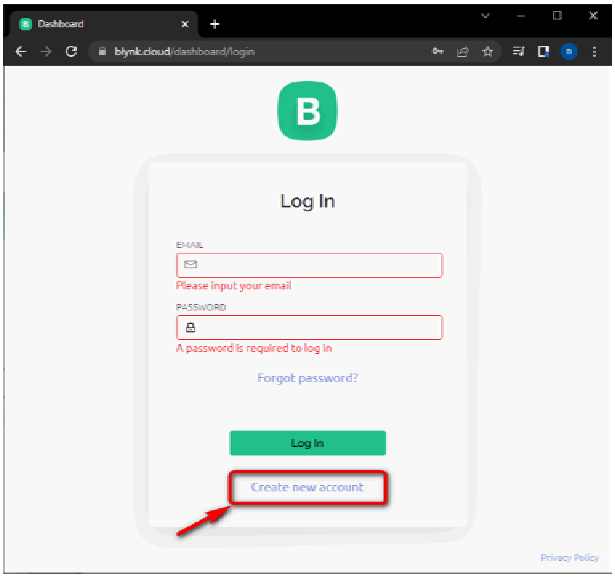
https://blynk.cloud/dashboard/login

(2) คลิกที่ปุ่ม **Create new account** ดังรูปที่ 11-6

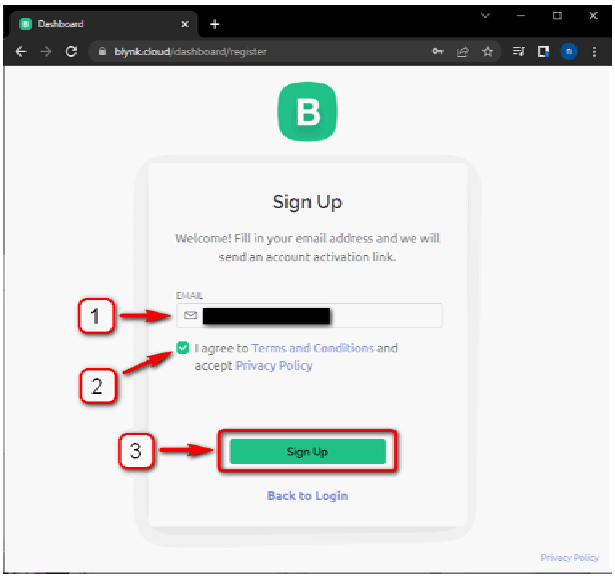
(3) จากนั้นจะปรากฏหน้าต่างสำหรับกรอกอีเมลที่ต้องการลงทะเบียนเพื่อสร้างบัญชีใช้งานกับ Blynk IoT ดังรูปที่ 11-7 ทำการระบุอีเมล (หมายเลข 1) แล้วคลิกเครื่องหมายถูกที่หัวข้อของการยอมรับเงื่อนไขในการใช้งาน (หมายเลข 2) ตามด้วยคลิกปุ่ม **Sign Up** (หมายเลข 3) ไปยังขั้นตอนถัดไป



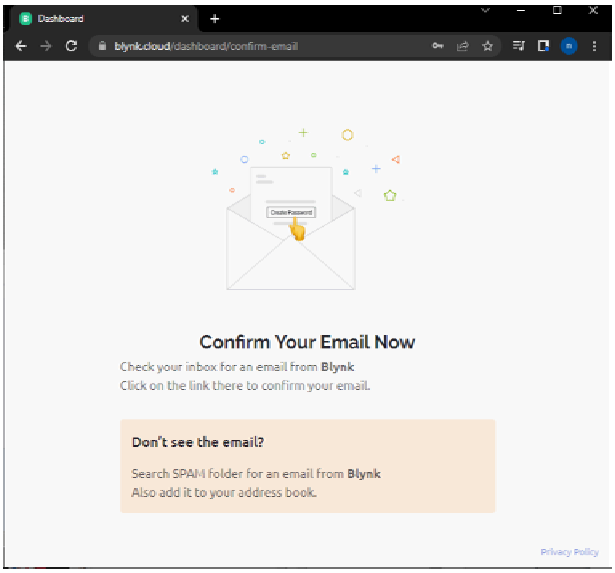
รูปที่ 11-5 หน้าต่าง login ของเว็บเพจ Blynk IoT



รูปที่ 11-6 แสดงปุ่ม Create new account



รูปที่ 11-7 แสดงขั้นตอนการกรอกอีเมลลงทะเบียนกับ Blynk IoT

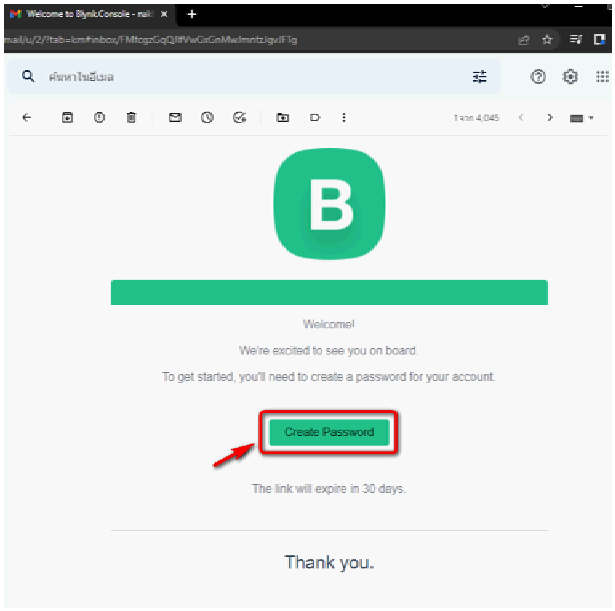


รูปที่ 11-8 หน้าต่างแจ้งให้ผู้พัฒนาทำการยืนยันการลงทะเบียน

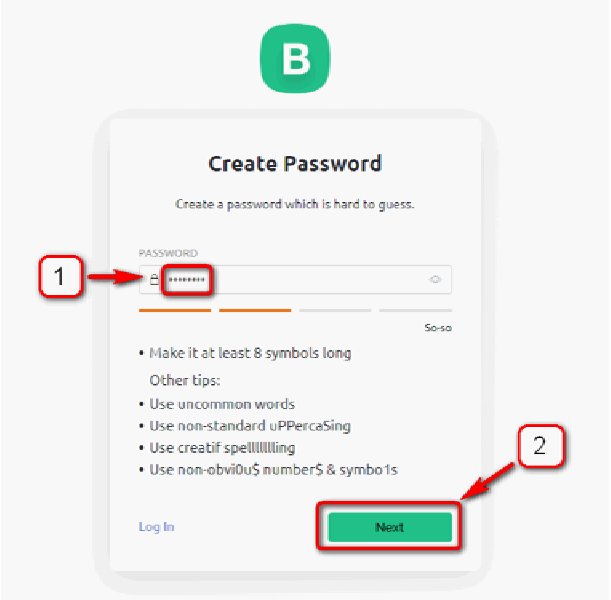
(4) หน้าต่างแจ้งเตือนให้ผู้พัฒนายืนยันการลงทะเบียนด้วยอีเมลที่ลงทะเบียนไว้ก่อนหน้านี้ ดังรูปที่ 11-8

(5) จากนั้นให้ผู้พัฒนาเปิดอีเมลที่ได้รับจาก Blynk IoT คลิกที่ปุ่ม **Create Password** เพื่อกำหนดรหัสผ่านตามรูปที่ 11-9

(6) กำหนดรหัสผ่านอย่างน้อย 8 ตัว เมื่อกำหนดเรียบร้อย คลิกปุ่ม **Next** ตามรูปที่ 11-10



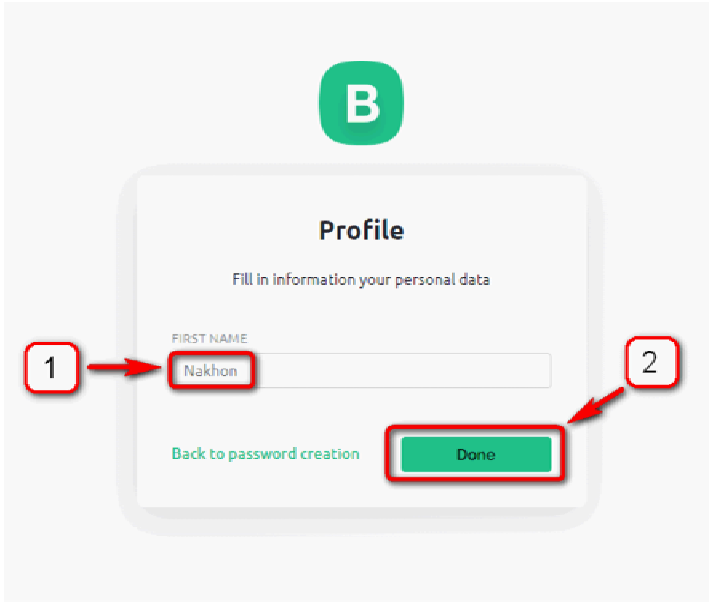
รูปที่ 11-9 เข้าสู่การกำหนดรหัสผ่านหลังจากได้รับอีเมลยืนยันการลงทะเบียนจาก Blynk IoT



รูปที่ 11-10 แสดงขั้นตอนกำหนดรหัสผ่านของบัญชีผู้ใช้งาน

(1) กำหนดรหัสผ่าน

(2) คลิกปุ่ม Next



รูปที่ 11-11 แสดงขั้นตอนตั้งชื่อโปรไฟล์ของบัญชีผู้ใช้งาน

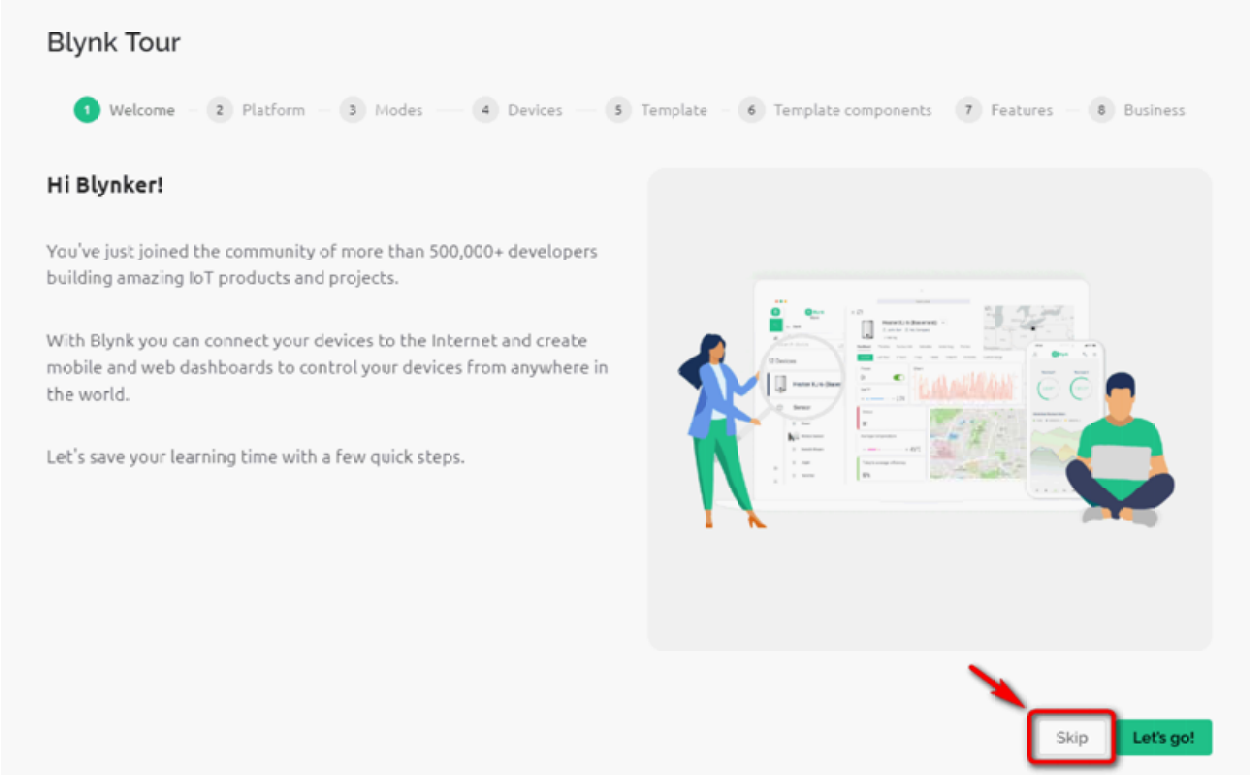
(1) ตั้งชื่อโปรไฟล์

(2) คลิกปุ่ม Done

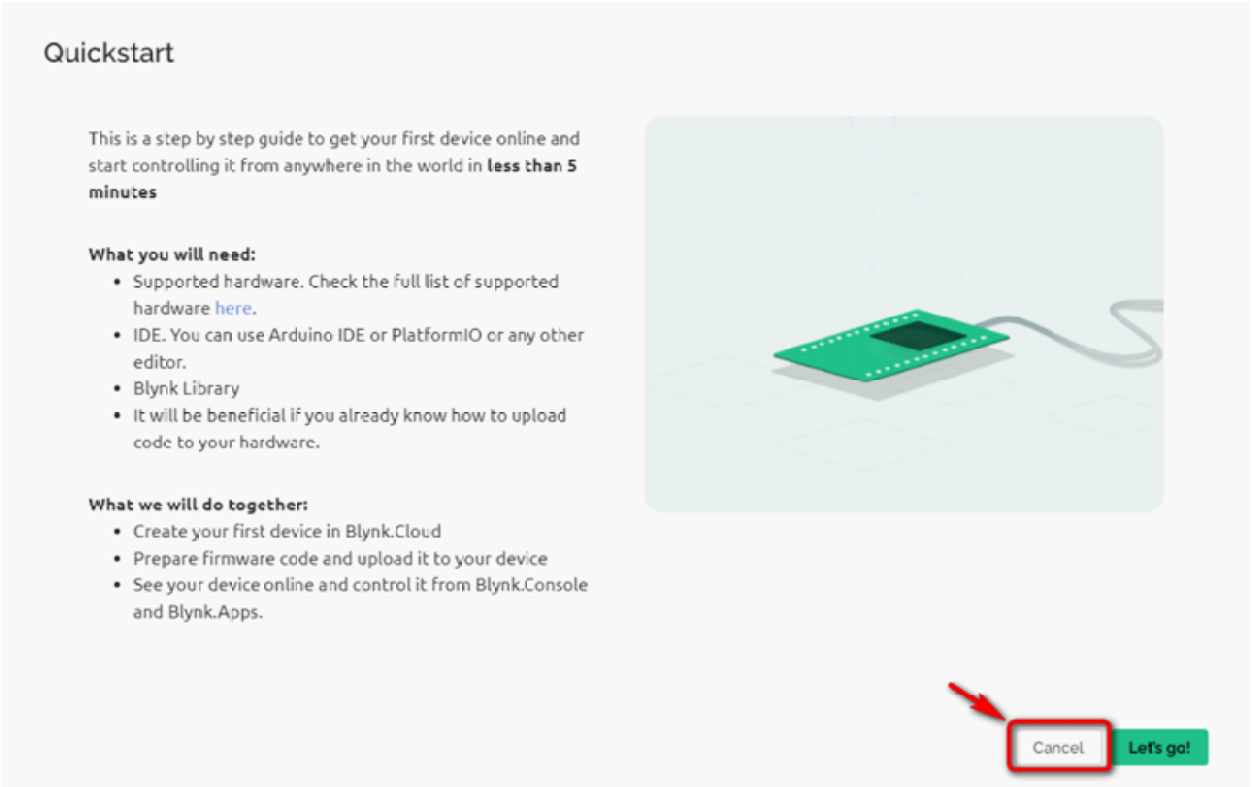
(7) หน้าต่างสำหรับตั้งชื่อโปรไฟล์ปรากฏขึ้นมา ผู้พัฒนาทำการกำหนดชื่อได้ตามต้องการในหัวข้อ **FIRST NAME** จากนั้น คลิกปุ่ม **Done** ตามรูปที่ 11-11

(8) หน้าต่าง **Blynk Tour** ปรากฏขึ้นมาเพื่อแนะนำรายละเอียดต่างเกี่ยวกับ Blynk IoT โดยในที่นี้จะข้ามขั้นตอนในส่วนนี้ไป โดยการคลิกปุ่ม **Skip** ตามรูปที่ 11-12

(9) หลังจากนั้น จะปรากฏหน้าต่าง **Quickstart** เพื่อแนะนำรายละเอียดต่างเกี่ยวกับ Blynk IoT ในที่นี้จะข้ามขั้นตอนในส่วนนี้ไปก่อน ด้วยการคลิกปุ่ม **Cancel** ตามรูปที่ 11-13

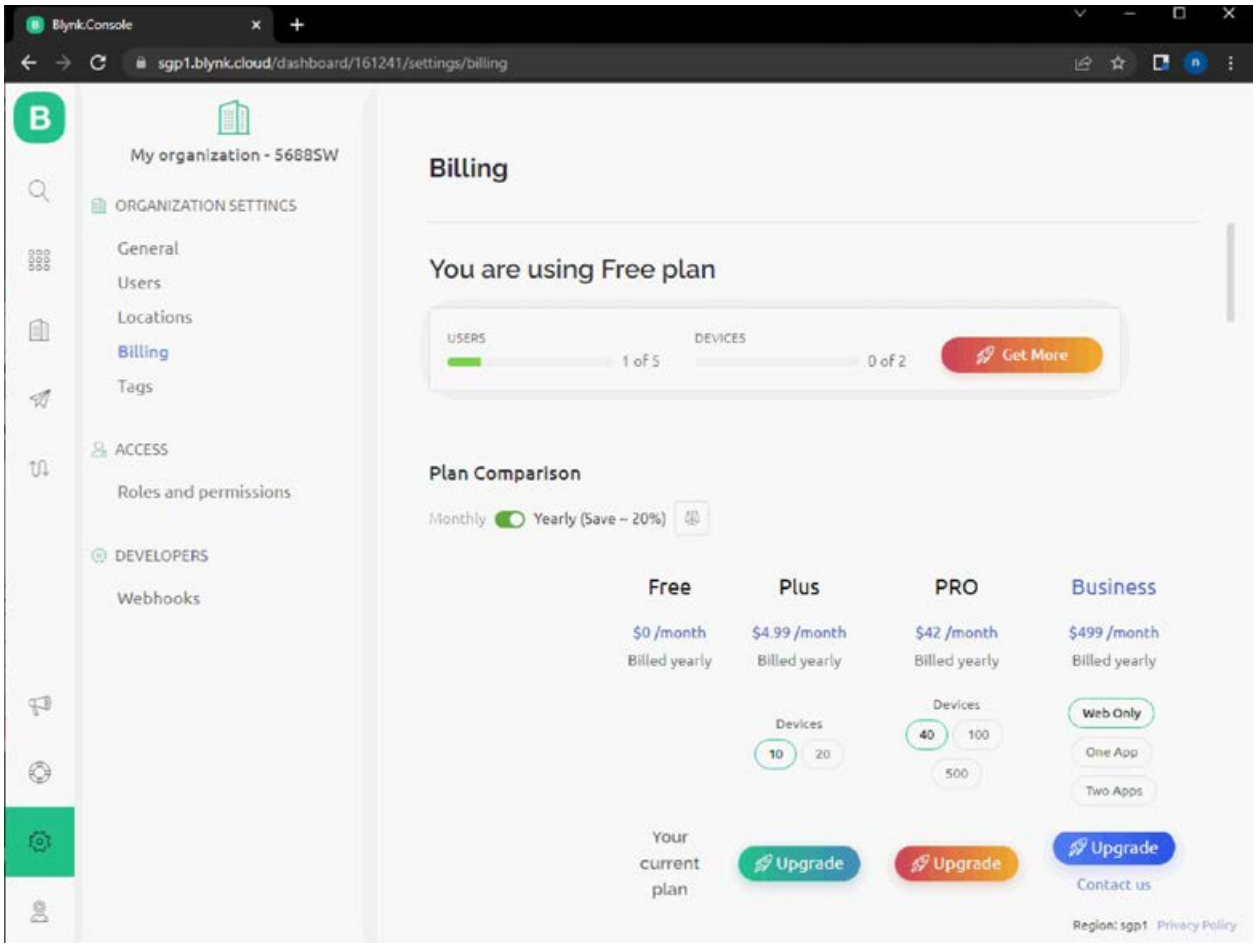


รูปที่ 11-12 หน้าต่างแนะนำรายละเอียดของ Blynk IoT



รูปที่ 11-13 แจ้งยกเลิกการรับชม Quick Start

(10) หน้าต่าง dashboard จะถูกแสดงขึ้นมา เมื่อการลงทะเบียนสำเร็จ โดยในหน้าต่างปัจจุบันนี้แสดงหัวข้อ Billing เป็นการใช้งานแบบไม่มีค่าใช้จ่ายใดๆ (Free plan) ตามรูปที่ 11-14



รูปที่ 11-14 แสดงหน้าต่าง dashboard ของ Blynk IoT เมื่อการลงทะเบียนสำเร็จ

11.3.2 ขั้นตอนติดตั้งแอปพลิเคชั่น Blynk IoT บนระบบปฏิบัติการ iOS

มีขั้นตอนดังนี้

(1) เข้าไปที่ AppStore ของระบบปฏิบัติการ iOS แล้วพิมพ์ค้นหาด้วยคำว่า **Blynk IoT** จากนั้นเลือกดาวน์โหลดแอปพลิเคชั่นตามรูปที่ 11-15 รอจนกระทั่งการติดตั้งแอปพลิเคชั่นเสร็จสมบูรณ์

(2) ทดสอบเปิดแอปพลิเคชั่นขึ้นมา จะพบหน้าตาตามรูปที่ 11-16 ซึ่งเป็นหน้า **Login** เพื่อเข้าใช้งาน โดยผู้พัฒนาสามารถใช้บัญชีหรือแอกเคาต์ที่เคยสมัครไว้ก่อนหน้านี้ล็อกอินใช้งานร่วมกับ Blynk IoT บนเว็บบราวเซอร์ได้ในเวลาเดียวกัน

รูปที่ 11-15 หน้าตาแอปพลิเคชั่น Blynk IoT บนระบบปฏิบัติการ iOS

รูปที่ 11-16 แสดงหน้า Login ของแอปพลิเคชั่น Blynk IoT

11.4 ขั้นตอนการติดตั้งไลบรารีต่างๆ ที่จำเป็นในการสร้างโค้ดเพื่อติดต่อกับ Blynk IoT บนโปรแกรม Arduino IDE

สำหรับไลบรารี C/C++ ที่ผู้พัฒนาจำเป็นต้องติดตั้งเพิ่มเติมสำหรับในการพัฒนาโปรแกรมเพื่อให้ WIO Terminal ทำงานกับ Blynk IoT ได้ มีดังนี้

1. ไลบรารี **Blynk**

2. ไลบรารี **ArduinoOTA**

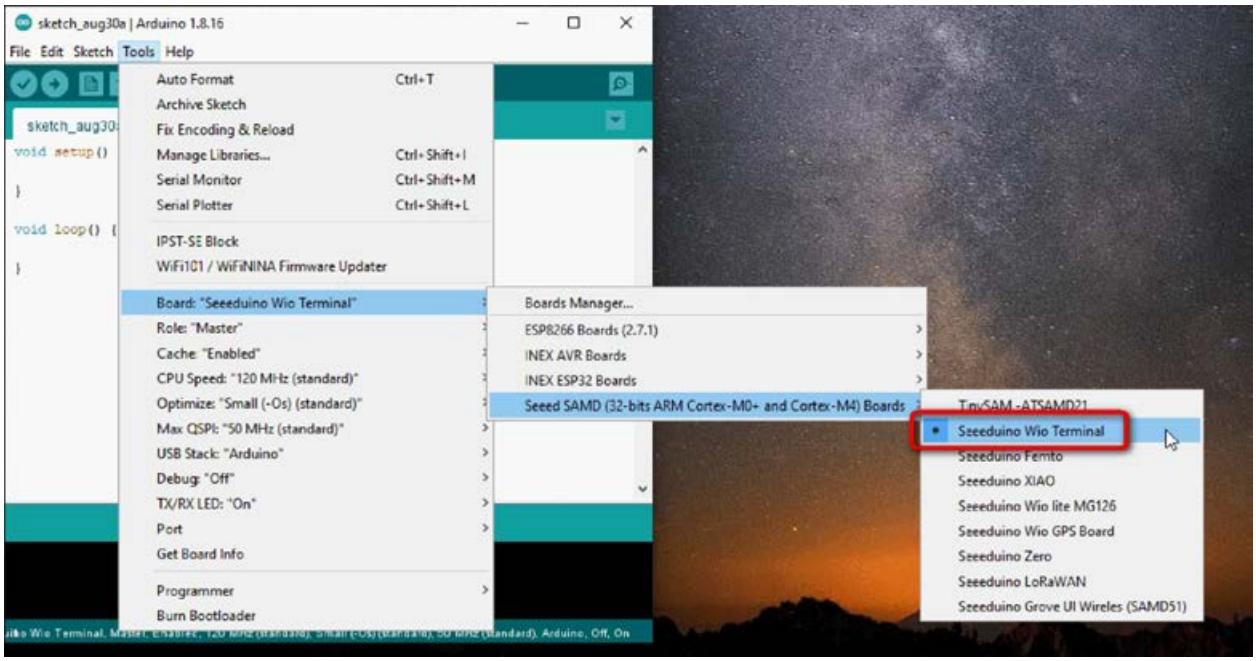
3. ไลบรารี **ArduinoHttpClient**

มีขั้นตอนดังนี้

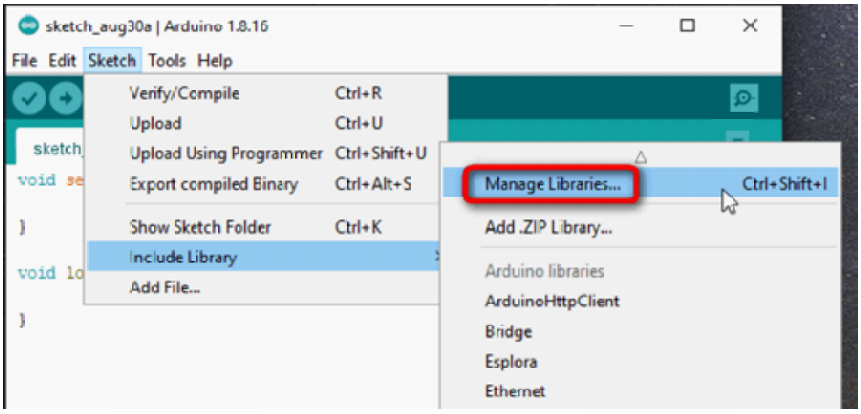
(1) เปิดโปรแกรม Arduino IDE เลือกบอร์ดพัฒนาเป็น **Seeeduino WIO Terminal** ตามรูปที่ 11-17

(2) เข้าสู่เมนู **Maneger Libraries...** เพื่อติดตั้งไลบรารีเพิ่มเติมทั้ง 3 ตัวที่กล่าวถึงไว้ข้างต้น ตามรูปที่ 11-18

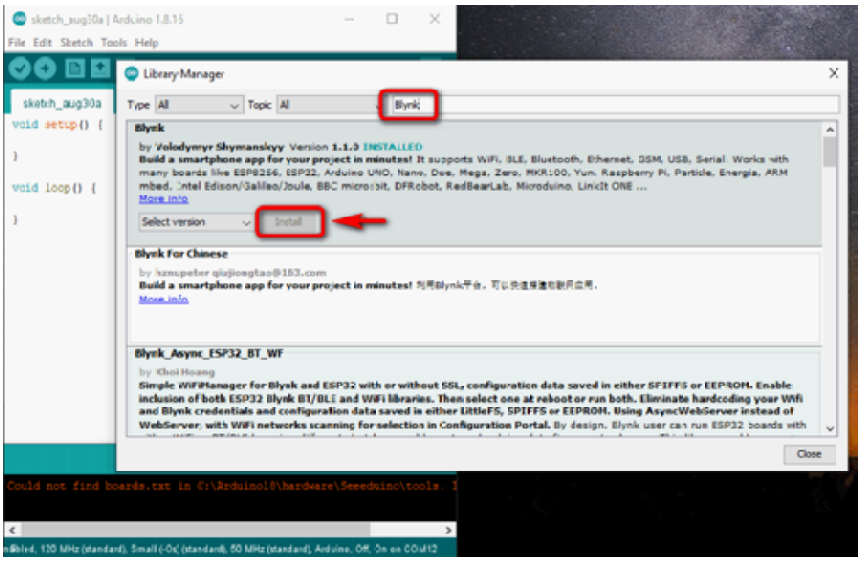
(3) ที่หน้าต่าง **Library Manager** พิมพ์ค้นหาชื่อไลบรารีตัวแรกชื่อ **Blynk** ตามรูปที่ 11-19 เมื่อพบแล้ว คลิกที่ปุ่ม **Install** เพื่อติดตั้งไลบรารี โดยเลือกเวอร์ชั่นล่าสุดได้ทันที



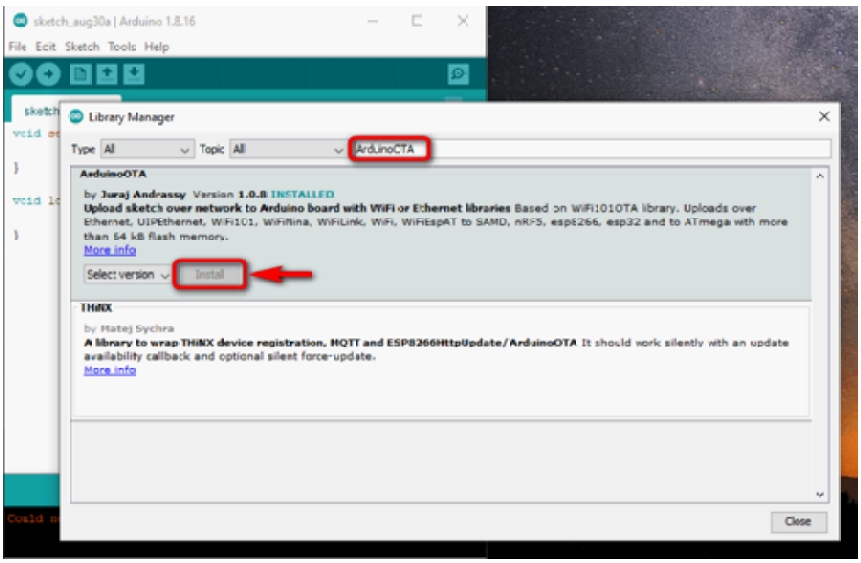
รูปที่ 11-17 แสดงการตั้งค่าบอร์ดพัฒนาเป็น WIO Terminal



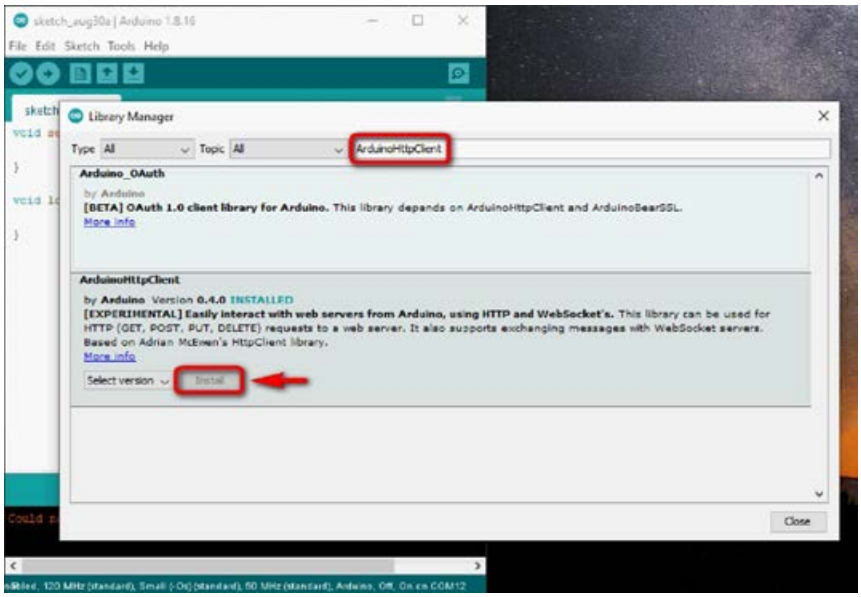
รูปที่ 11-18 แสดงการเข้าถึงเมนู Maneger Libraries...



รูปที่ 11-19 แสดงการเลือกติดตั้ง ไลบรารี Blynk



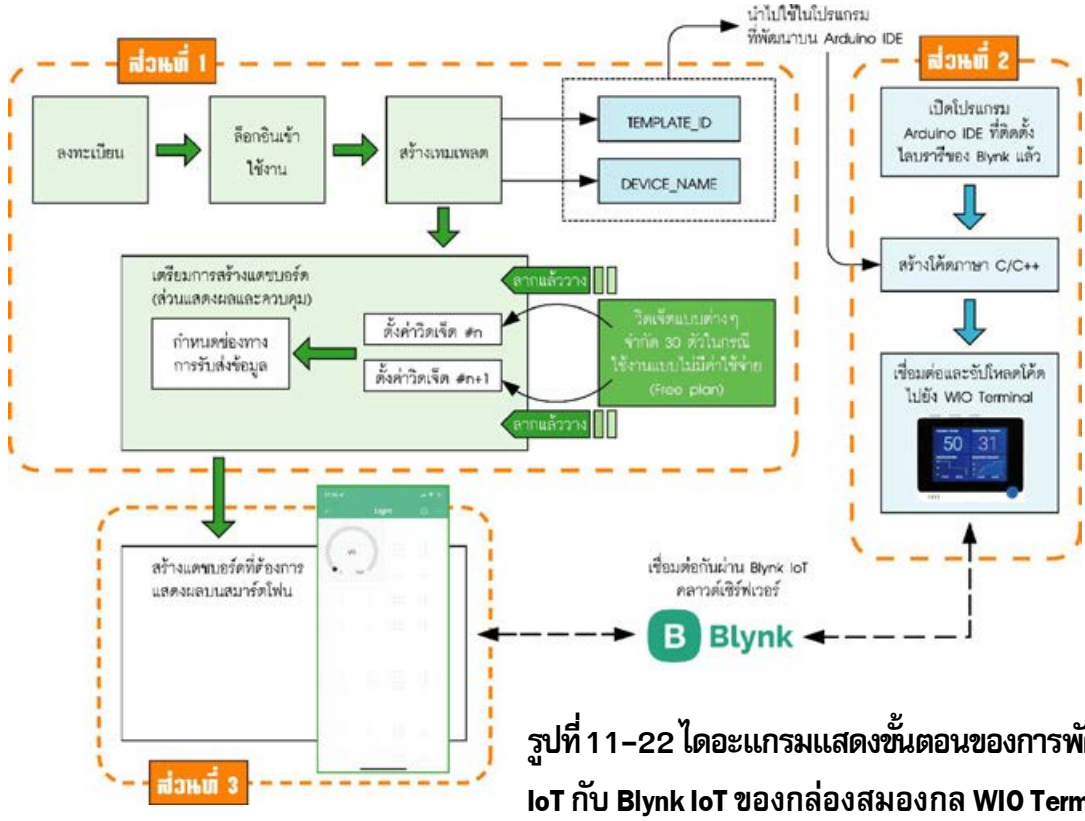
รูปที่ 11-20 แสดงการเลือกติดตั้ง ไลบรารี ArduinoOTA



รูปที่ 11-21 แสดงการติดตั้งไลบรารี ArduinoHttpClient

(4) ที่หน้าต่าง **Library Manager** พิมพ์ค้นหาชื่อไลบรารีตัวสองที่ชื่อ **ArduinoOTA** ตามรูปที่ 11-20 เมื่อพบแล้วคลิกที่ปุ่ม **Install** เพื่อเริ่มการติดตั้งเช่นเดียวกัน

(5) ที่หน้าต่าง **Library Manager** พิมพ์ค้นหาชื่อไลบรารีตัวสุดท้ายที่ชื่อ **ArduinoHttpClient** ตามรูปที่ 11-21 เมื่อพบแล้วคลิกที่ปุ่ม **Install** เพื่อทำการติดตั้ง



รูปที่ 11-22 ไดอะแกรมแสดงขั้นตอนของการพัฒนางาน IoT กับ Blynk IoT ของกล่องสมองกล WIO Terminal

11.5 ตัวอย่างการพัฒนาโปรแกรมเพื่อใช้งาน WIO Terminal กับ Blynk IoT

11.5.1 ตัวอย่างที่ 1 Light monitoring

เป็นตัวอย่างส่งค่าการทำงานของตัวตรวจจับแสงของ WIO Terminal ไปแสดงผลที่ Blynk IoT ผ่านเว็บบราวเซอร์และบนสมาร์ตโฟนอย่างง่าย สรุปเป็นไดอะแกรมตามรูปที่ 11-22 ส่วนขั้นตอนมีดังนี้

ส่วนที่ 1 การเตรียมการที่ Blynk IoT

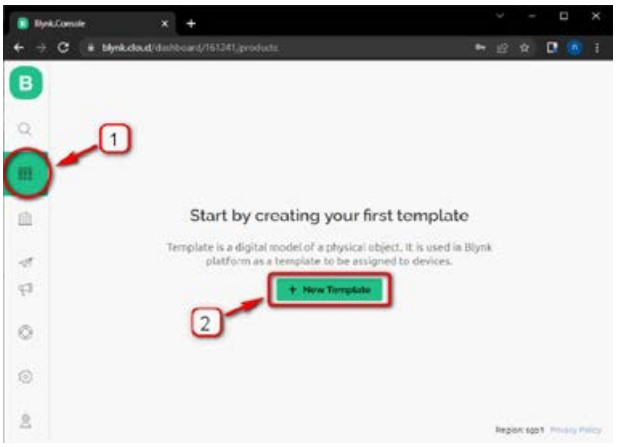
(1) เชื่อมต่อคอมพิวเตอร์เข้ากับเครือข่ายอินเทอร์เน็ต แล้วเปิดเว็บบราวเซอร์เพื่อไปยังเว็บไซต์ **https://blynk.cloud/dashboard/login** จากนั้นล็อกอินเพื่อเข้าใช้งาน

(2) คลิกเลือกเมนู **Template** ทางซ้ายมือ เพื่อเตรียมสร้างเทมเพลต (template) หรือแบบร่างขึ้นมาใช้งาน ตามรูปที่ 11-23

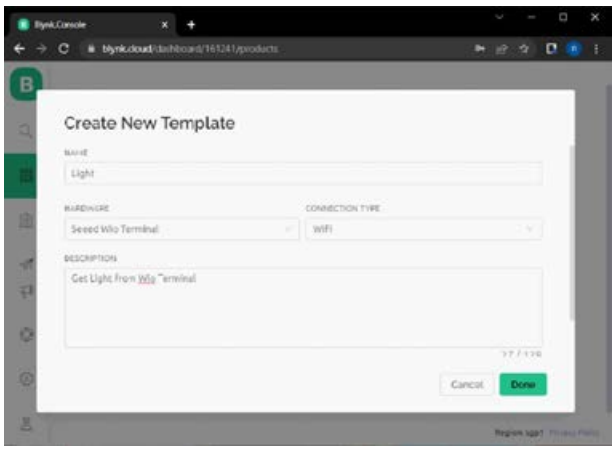
(3) หน้าต่าง **Create New Template** ปรากฏขึ้นตามรูปที่ 11-24 ทำการตั้งค่าดังนี้

- หัวข้อ **NAME** กำหนดชื่อเป็น ***Light***
- หัวข้อ **HARDWARE** เลือกอุปกรณ์เป็น ***Seeed WIO Terminal***
- หัวข้อ **CONNECTION TYPE** เลือกการเชื่อมต่อเป็น ***WiFi***
- หัวข้อ **DESCRIPTION** สำหรับกำหนดคำอธิบายการทำงานตามที่ผู้พัฒนาต้องการ

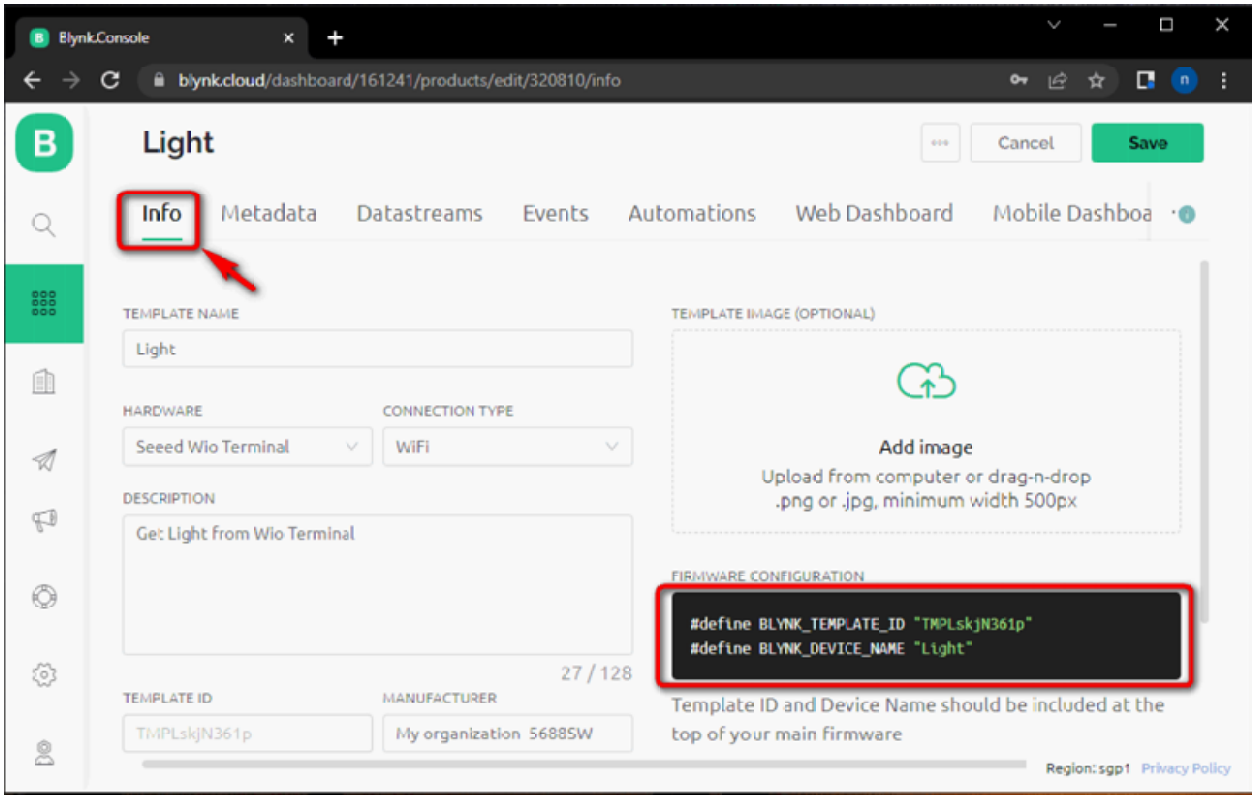
จากนั้นคลิกปุ่ม **Done**



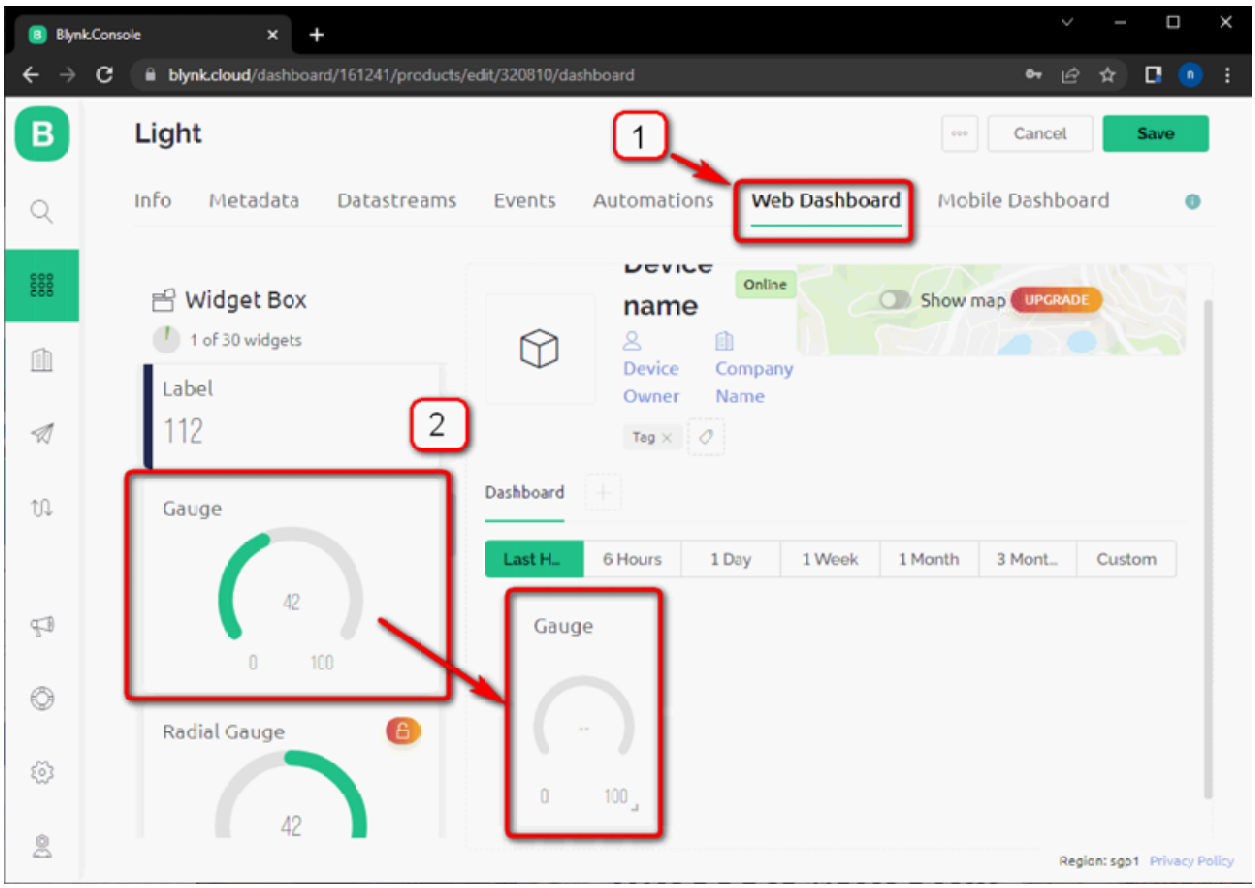
รูปที่ 11-23 แสดงการเข้าเมนู Template ของ Blynk IOT



รูปที่ 11-24 แสดงหน้าต่าง Create New Template ใหม่



รูปที่ 11-25 แสดงหน้าต่างเทมเพลตที่ชื่อ Light ถูกสร้างขึ้นโดยหัวข้อ Info



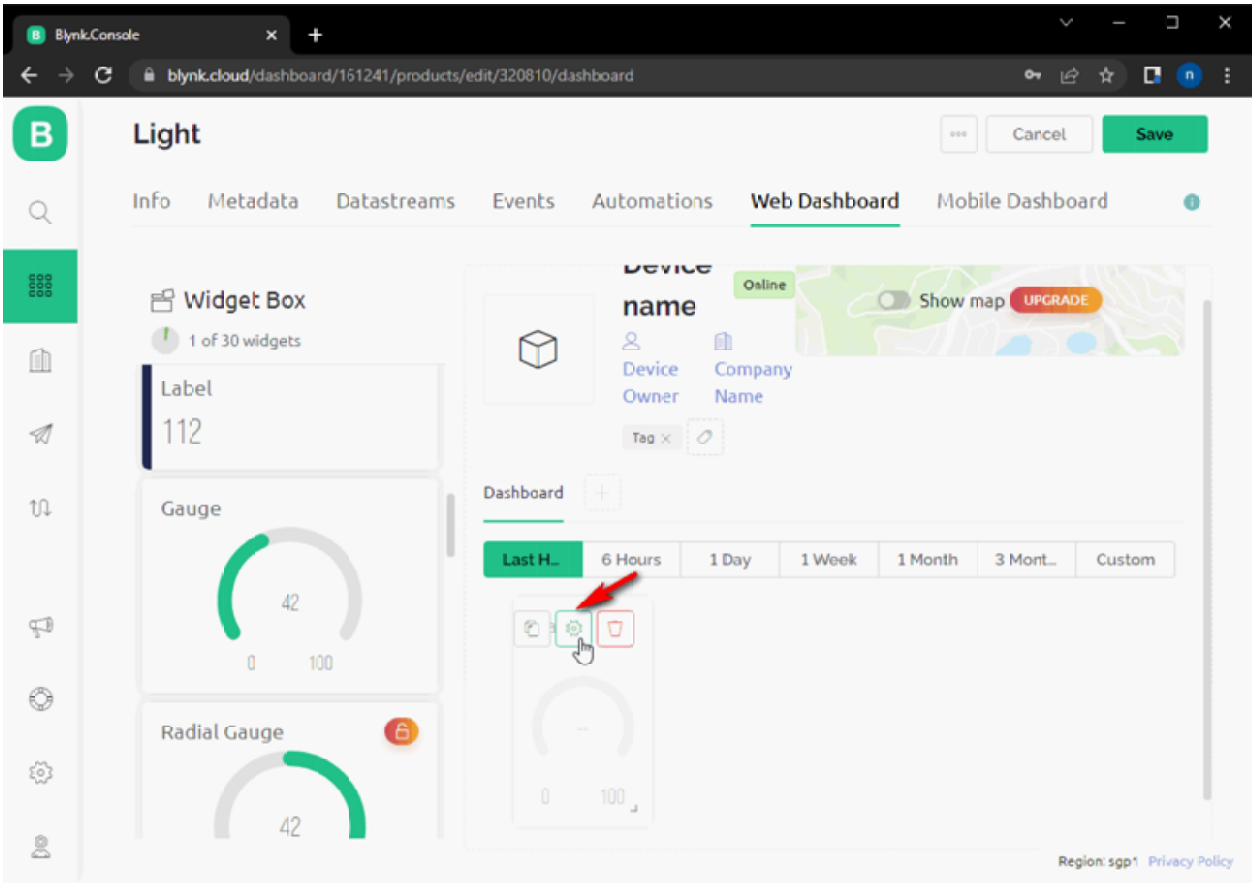
รูปที่ 11-26 แสดงการวางวิดเจ็ต Gauge เพิ่มเข้าไปในโปรเจ็กต์

(4) หน้าต่างเทมเพลตที่ชื่อ **Light** ถูกสร้างขึ้นในหัวข้อ **Info** ตามรูปที่ 11-25 โดยแสดงรหัสประจำตัวของเทมเพลตที่สำคัญ อันได้แก่

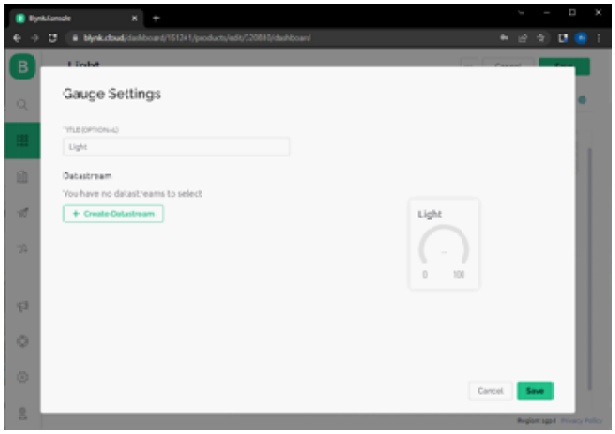
o **TEMPLATE_ID** และ **DEVICE_NAME** ผู้พัฒนาจำเป็นต้องนำค่ารหัสทั้งสองของเทมเพลตไปกำหนดไว้ที่ตอนต้นของโปรแกรมภาษา C/C++ เพื่อกำหนดการรับส่งข้อมูลระหว่างคลาวด์เซิร์ฟเวอร์ของ Blynk IoT และ WIO Terminal

(5) คลิกเลือกที่หัวข้อ **Web Dashboard** เลือกวิดเจ็ต **Gauge** สำหรับแสดงค่าระดับแสง โดยการลากไปวางที่หน้าต่างโปรเจ็กต์ ตามรูปที่ 11-26

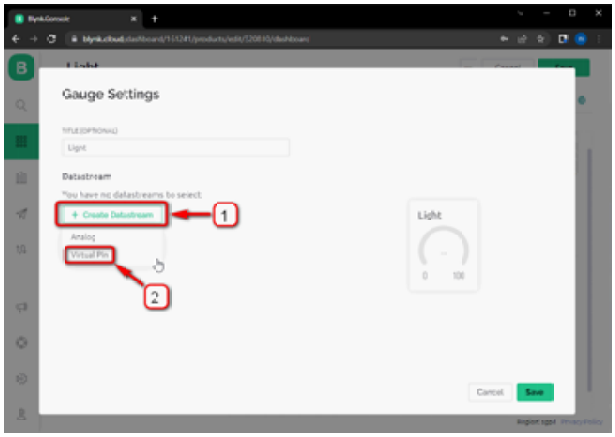
(6) ทำการกำหนดคุณสมบัติของวิดเจ็ต **Gauge** โดยการคลิกเมนูรูปฟันเฟืองที่วิดเจ็ต **Gauge** ตามรูปที่ 11-27



รูปที่ 11-27 แสดงเมนูรูปฟันเฟืองที่วิดเจ็ต Gauge สำหรับกำหนดคุณสมบัติ



รูปที่ 11-28 แสดงการกำหนดชื่อประจำตัววิดเจ็ต Gauge สำหรับแสดงค่าความสว่าง



รูปที่ 11-29 แสดงการสร้าง Datastream แบบ Virtual Pin

(7) จากนั้นหน้าต่าง **Gauge Setting** ปรากฏขึ้น ผู้พัฒนาสามารถกำหนดชื่อประจำตัววิดเจ็ต **Gauge** จากหัวข้อ **TITLE** ได้ตามต้องการ ในที่นี้กำหนดชื่อเป็น **LIGHT** ตามรูปที่ 11-28

(8) คลิกที่ปุ่ม **Create Datastream** เพื่อเพิ่มช่องทางการรับส่งข้อมูล แล้วเลือกรายการ **Virtual Pin** ตามรูปที่ 11-29

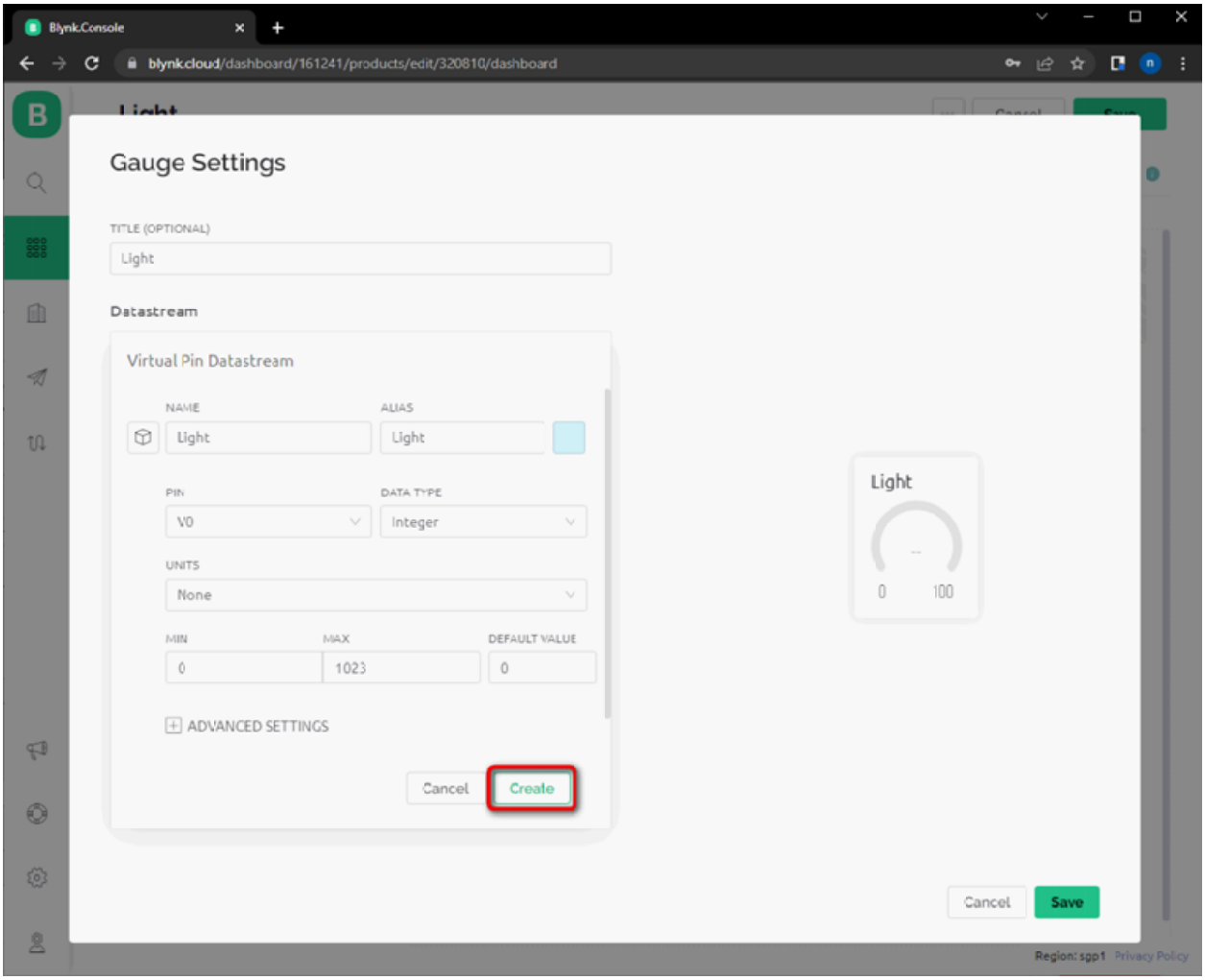
(9) จากนั้นหน้าต่างสำหรับกำหนดคุณสมบัติของ **Virtual Pin** จะปรากฏขึ้นตามรูปที่ 11-30 ทำการตั้งค่าดังนี้

- หัวข้อ **NAME** กำหนดชื่อเป็น ***Light***
- ที่หัวข้อ **ALIAS** มีค่าเป็น ***Light***
- หัวข้อ **MAX** (ค่าสูงสุด) เป็น ***1023***
- หัวข้อ **DEFAULT VALUE** เป็น ***0***

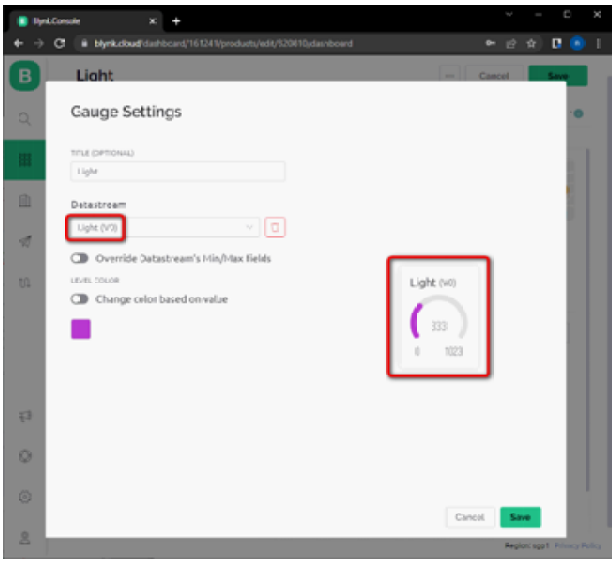
จากนั้น คลิกปุ่ม **Create**

(10) กลับมายังหน้าต่างหลักของ **Gauge Setting** ที่ถูกกำหนดคุณสมบัติไว้ก่อนหน้านี้ ตามรูปที่ 11-31 โดยที่หัวข้อ **Datastream** ได้กำหนดเป็นช่อง **V0** และขอบเขตของ **Gauge** จะถูกกำหนดการแสดงผลอยู่ในช่วงตั้งแต่ 0 ถึง 1023 จากนั้นคลิกปุ่ม **Save** เพื่อบันทึกการตั้งค่า

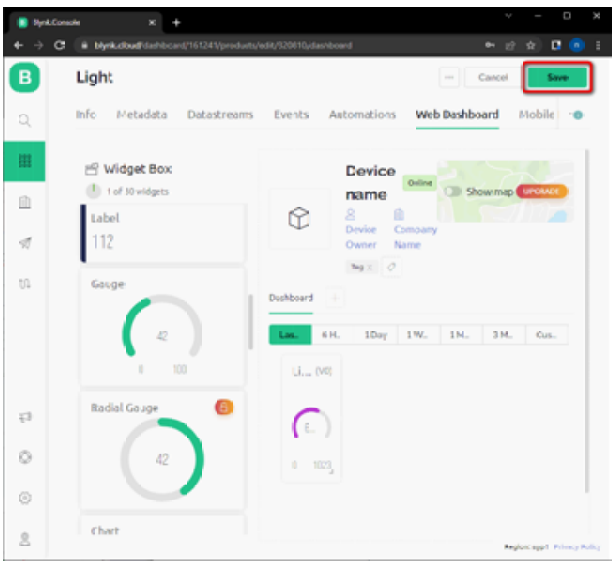
(11) จากนั้นกลับมายังหน้าต่างหลักของ **Web Dashboard** ตามรูปที่ 11-32 คลิกปุ่ม **Save** เพื่อบันทึกการตั้งค่าอีกครั้ง



รูปที่ 11-30 แสดงหน้าต่างสำหรับกำหนดคุณสมบัติของ Virtual Pin



รูปที่ 11-31 แสดงหน้าต่างหลักของ Gauge Setting ที่ได้รับการกำหนดคุณสมบัติไว้ก่อนหน้านี้

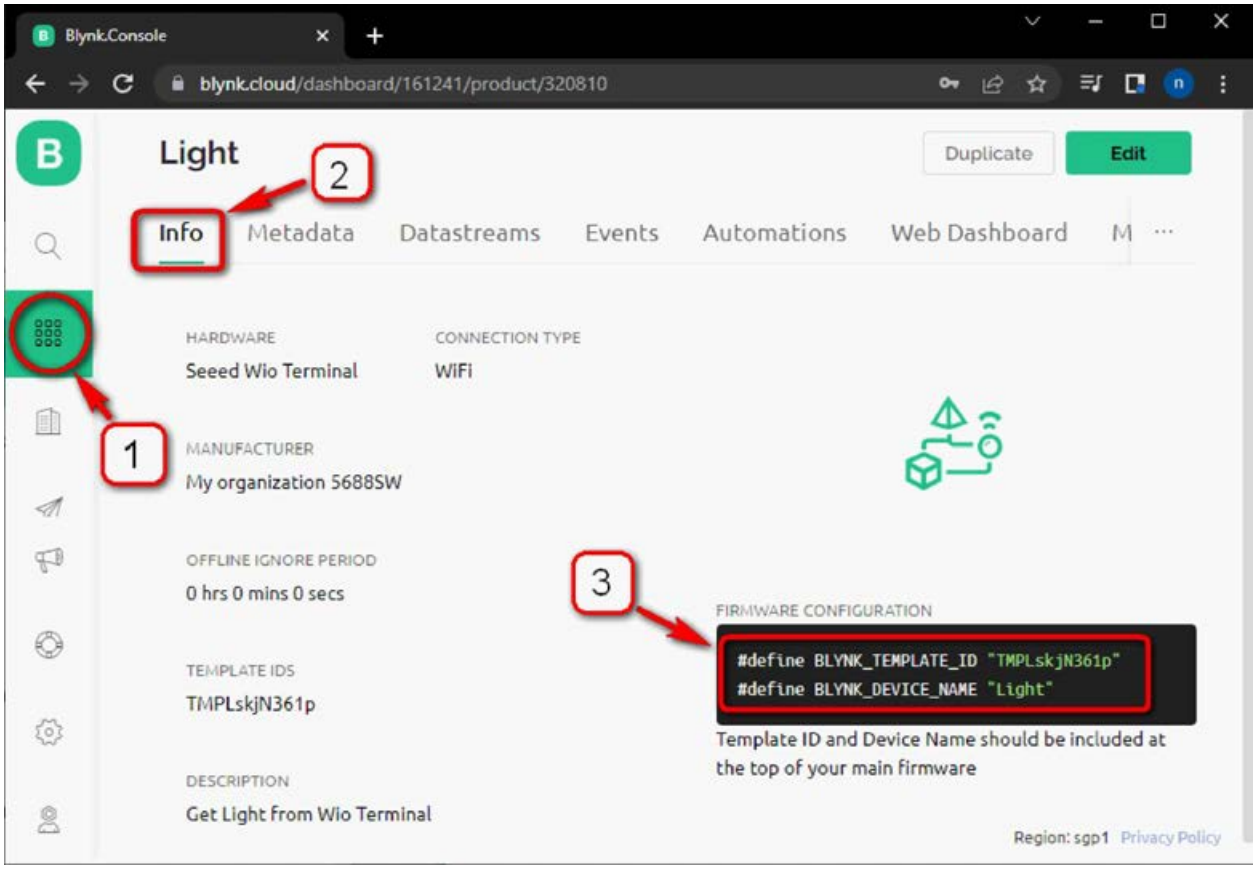


รูปที่ 11-32 แสดงการบันทึกหน้าต่างหลักของ Web Dashboard ของเทมเพลตที่ชื่อ Light

ส่วนที่ 2 การเตรียมการสร้างโค้ดโปรแกรมภาษา C/C++ ที่ Arduino IDE

(12) เปิดโปรแกรม Arduino IDE ขึ้นมาเพื่อทำการพัฒนาโปรแกรมให้กับบอร์ด WIO Terminal

(12A) สร้างโค้ดตามโปรแกรมที่ 11-1 โดย 2 บรรทัดแรกของโปรแกรมในส่วนของ **BLYNK_TEMPLATE_ID** และ **BLYNK_DEVICE_NAME** ผู้พัฒนาจำเป็นต้องกำหนดให้ตรงกับเทมเพลตที่สร้างไว้ก่อนหน้านี้ โดยคัดลอกค่ารหัสดังกล่าวจากหัวข้อของ **Template** ตามรูปที่ 11-33



รูปที่ 11-33 แสดงค่ารหัส BLYNK_TEMPLATE_ID และ BLYNK_DEVICE_NAME ของเทมเพลตที่ชื่อ Light

```
#define BLYNK_TEMPLATE_ID “xxxxxxx”        // สำหรับกำหนดค่ารหัส TEMPLATE ID
#define BLYNK_DEVICE_NAME “xxxxx”          // สำหรับกำหนดค่ารหัส DEVICE NAME
#define BLYNK_FIRMWARE_VERSION “0.1.0”     // สำหรับกำหนดค่ารหัส FIRMWARE VERSION
#define BLYNK_PRINT Serial
#define APP_DEBUG
#include “BlynkEdgent.h”                   // ผนวกไลบรารีสำหรับติดต่อ Blynk IoT
#include <TFT_eSPI.h>                      // ผนวกไลบรารีควบคุมการแสดงผล LCD
```

โปรแกรมที่ 11-1 ไฟล์ Blynk_Send_Light.ino โปรแกรมสำหรับทดสอบใช้งาน WIO Terminal อ่านค่าจากตัวตรวจจับแสงแล้วส่งขึ้นไปแสดงผลที่ Blynk IoT (มีต่อ)

```
TFT_eSPI tft;            // สร้างออปเจ็กต์ชื่อ tft จากคลาส TFT_eSPI สำหรับควบคุมการแสดงผล LCD
BlynkTimer timer;        // สร้างออปเจ็กต์ BlynkTimer ชื่อ timer
void sendLight()         // ฟังก์ชั่นสำหรับทำหน้าที่แสดงผลค่าระดับแสงและส่งไปแสดงผลที่ Blynk IoT App
{
   int light = analogRead(WIO_LIGHT);          // อ่านค่าความเข้มของแสงจากตัวตรวจจับแสง
   String str = "Light: "+ String(light);      // แปลงค่าระดับแสงเป็นข้อมูลสตริง
   tft.drawString(str+String("    "), 80, 120);// แสดงค่าระดับแสง
   Blynk.virtualWrite(V0, light);              // ส่งค่าระดับแสงไปแสดงผลที่ Blynk App ผ่านช่อง V0
}
void setup()
{
   Serial.begin(115200);                       //ใช้ Serial Monitor แจ้งสถานะการเชื่อมต่อเซิร์ฟเวอร์ Blynk
   delay(100);                                 // หน่วงเวลาเล็กน้อย
   pinMode(WIO_LIGHT, INPUT);                  // กำหนดขาพอร์ตที่เชื่อมต่อกับตัวตรวจจับแสงให้เป็นอินพุต
   tft.begin();                                // เริ่มต้นการทำงานของจอแสดงผล
   tft.setRotation(3);                         // กำหนดทิศทางการหมุนหน้าจอแสดงผล
   tft.fillScreen(TFT_BLACK);                  // ระบายสีดำเป็นพื้นหลังหน้าจอ LCD
   tft.setTextColor(TFT_ORANGE,TFT_BLACK);     // กำหนดสีตัวอักษรเป็นสีเขียว พื้นหลังสีดำ
   tft.setTextSize(3);                         // กำหนดขนาดฟอนต์เป็น 3
   tft.drawString("Wait...", 80, 120);         // แสดงข้อความ Wait...
   BlynkEdgent.begin();                        // เริ่มต้นเชื่อมต่อไปยังเซิร์ฟเวอร์ Blynk IoT
   timer.setInterval(1000L, sendLight);        // เปิดใช้งานไทเมอร์อินเตอร์รัปต์ทุกๆ 1 วินาที
}                                              // กำหนดให้ไปทำงานที่ฟังก์ชั่น sendLight
void loop()
{
   BlynkEdgent.run();                          // สั่งรัน Blynk IoT
   timer.run();                                // สั่งรันไทเมอร์
}
```

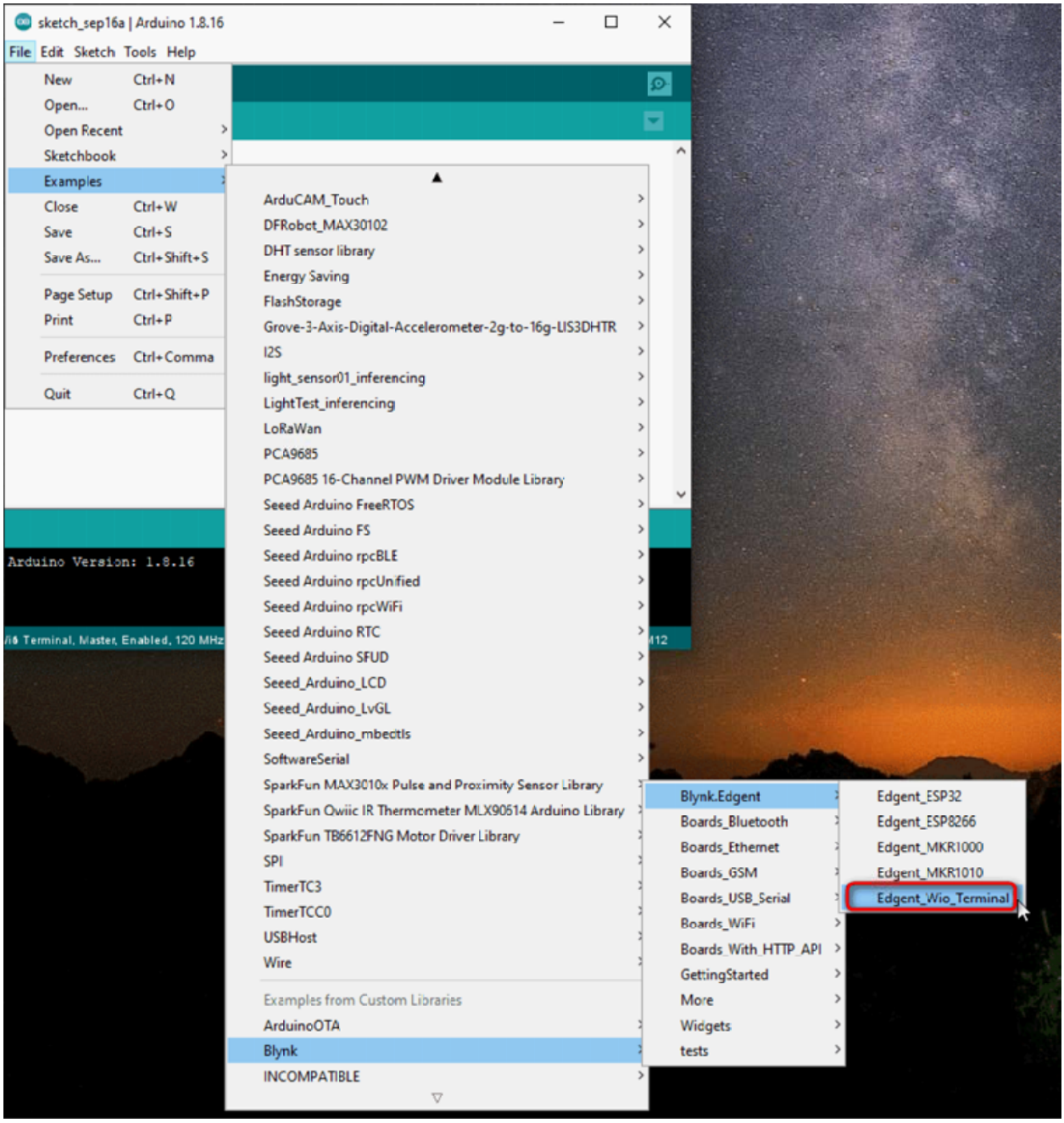
การทำงานของโปรแกรม

หลังจากเชื่อมต่อไปยังเซิร์ฟเวอร์ Blynk IoT ได้สำเร็จ ผ่านทางการเชื่อมต่อกับเครือข่ายอินเทอร์เน็ต โดย WIO Terminal ใช้โมดูล WiFi ติดต่อผ่านเราเตอร์เพื่อเข้าสู่เครือข่ายอินเทอร์เน็ตและ Blynk IoT App เราเตอร์หรือสมาร์ตโฟนที่เป็นตัวกระจายสัญญาณ WiFi ตัวบอร์ด WIO Terminal จะเกิดอินเตอร์รัปต์จากไทเมอร์ทุกๆ 1 วินาที เพื่อกระโดดไปทำงานที่ฟังก์ชั่น **sendLight** โดยโปรแกรมจะอ่านค่าระดับแสงที่ตรวจวัดได้ไปแสดงผลที่หน้าจอ LCD และส่งไปยังเซิร์ฟเวอร์ Blynk IOT ผ่านช่องทางพอร์ตเสมือน V0

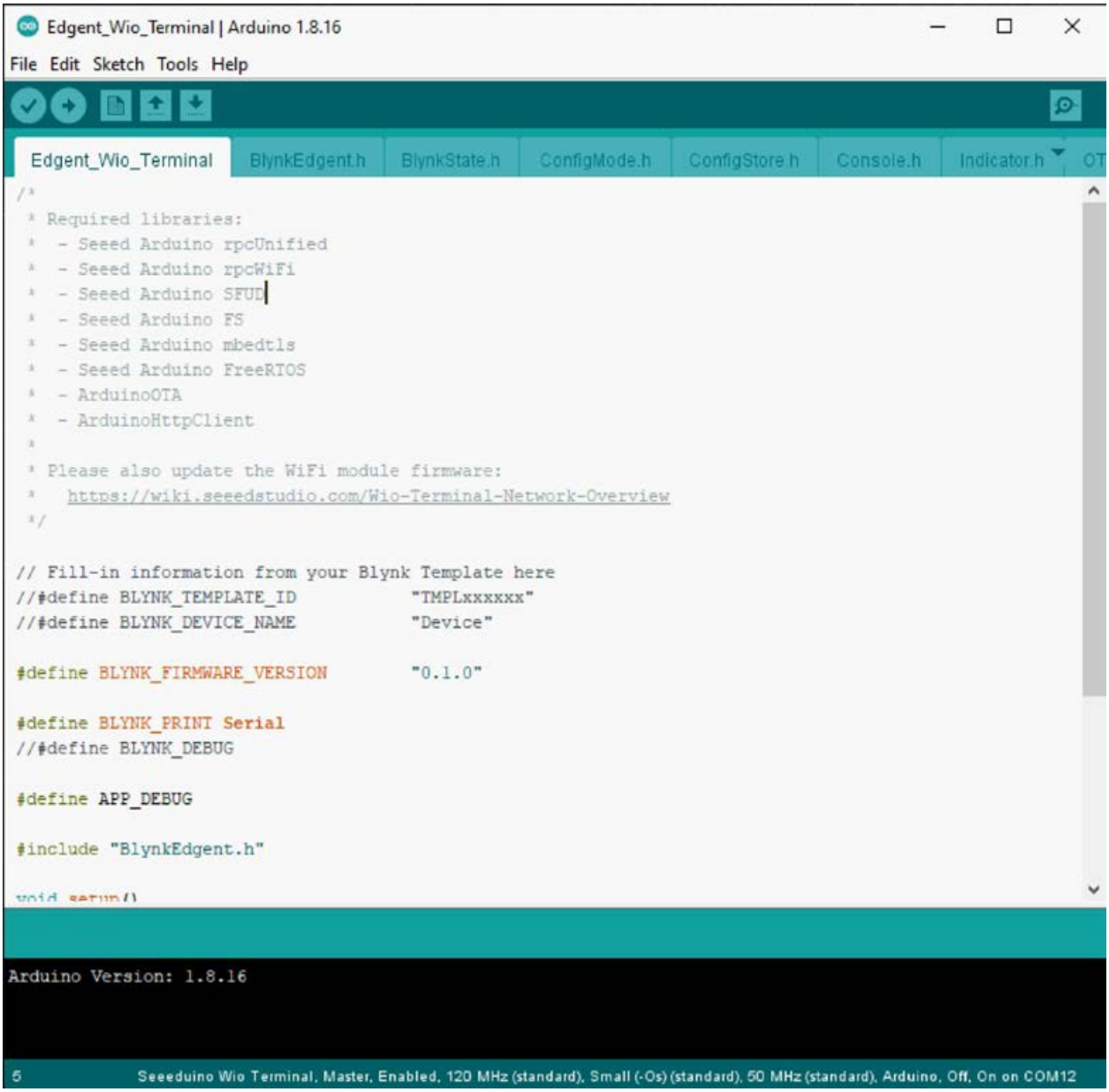
โปรแกรมที่ 11-1 ไฟล์ Blynk_Send_Light.ino โปรแกรมสำหรับทดสอบใช้งาน WIO Terminal อ่านค่าจากตัวตรวจจับแสงแล้วส่งขึ้นไปแสดงผลที่ Blynk IoT (จบ)

(12B) ผู้พัฒนาอาจเปิดไฟล์ของโปรแกรมตัวอย่าง (Example) จาก Arduino IDE ซึ่งมาพร้อมกับการติดตั้งไลบรารี Blynk เพื่อนำมาดัดแปลงเพิ่มเติม เป็นการช่วยอำนวยความสะดวกให้พัฒนาโปรแกรมได้เร็วขึ้น มีขั้นตอนดังต่อไปนี้

(12B.1) ที่โปรแกรม Arduino IDE เลือกเมนู **File>Examples>Blynk> Blynk.Edgent >Edgent_Wio_Terminal** ตามรูปที่ 11-34



รูปที่ 11-34 แสดงเมนู File > Examples > Blynk > Blynk.Edgent > Edgent_Wio_Terminal



รูปที่ 11-35 แสดงไฟล์โปรแกรม Edgent_Wio_Terminal.ino บนโปรแกรม Arduino IDE

(12B.2) ไฟล์โปรแกรมตัวอย่างที่ชื่อ **Edgent_Wio_Terminal.ino** จะถูกเปิดขึ้นมาตามรูปที่ 11-35

(12B.3) แนะนำให้ทำการ **Save As** เพื่อเก็บไฟล์ต้นทางไว้ แล้วตั้งชื่อไฟล์ใหม่ตามต้องการ

(13) เมื่อทำการสร้างโค้ดหรือแก้ไขโค้ดให้เป็นไปตามโปรแกรมที่ 11-1 แล้ว บันทึกไฟล์

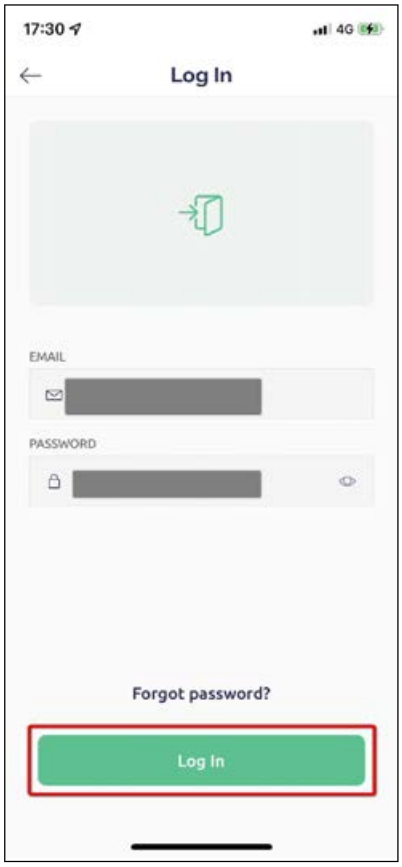
(14) เชื่อมต่อ WIO Terminal เข้ากับคอมพิวเตอร์ผ่านทางพอร์ต USB จากนั้นทำการอัปโหลดโปรแกรมไปยัง WIO Terminal ให้เรียบร้อย

ส่วนที่ 3 เชื่อมต่อ Blynk IoT App กับ WIO Terminal และทดสอบการ ทำงาน

เป้าหมายของส่วนนี้คือ ทำการออกแบบสร้างแดชบอร์ดบนเทมเพลตที่ตั้งค่าไว้ก่อนหน้านี้ และเชื่อมต่อกับ WIO Terminal รวมถึงการเข้าถึงเซิร์ฟเวอร์ Blynk IoT ในขณะที่โปรแกรมทำงาน สังเกตค่าระดับแสงที่ตรวจวัดได้บนจอแสดงผลของ WIO Terminal จากนั้นนำมือไปบังแสงที่ตกกระทบตัวตรวจจับที่อยู่บริเวณด้านหลังของ WIO Terminal สังเกตการเปลี่ยนแปลงค่าของระดับแสงที่ตรวจจับได้ และสังเกตที่หน้าแดชบอร์ดของ Blynk IoT App บนสมาร์ตโฟนและบนเว็บบราวเซอร์ ควรได้ผลลัพธ์ตรงกับที่แสดงบนจอแสดงผล WIO Terminal

สำหรับการออกแบบแดชบอร์ดสำหรับ Blynk IoT App และทดสอบการทำงาน มีขั้นตอนดังนี้

(15) เปิด Blynk App บนสมาร์ตโฟนที่แนะนำการติดตั้งไว้ก่อนหน้านี้ขึ้นมา โดยในที่นี้ผู้เขียนขอยกตัวอย่างจากการทำงานบนสมาร์ตโฟนที่ติดตั้งระบบปฏิบัติการ iOS ทำการล็อกอินเข้าใช้งาน Blynk IOT ด้วยบัญชีที่เคยสมัครไว้ตามรูปที่ 11-36

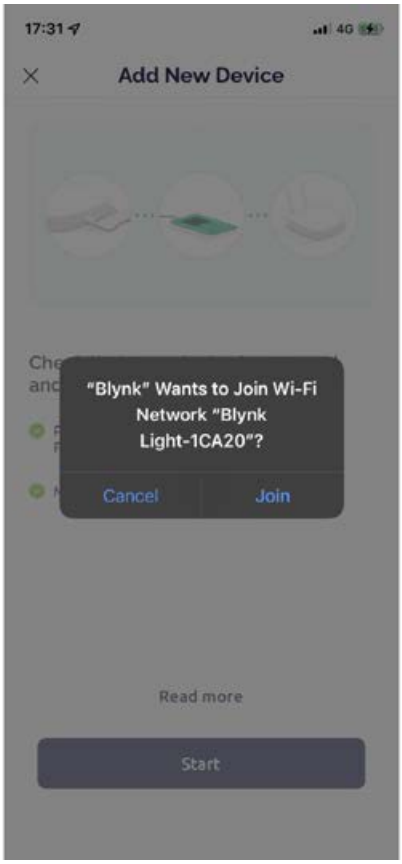


รูปที่ 11-36 หน้าต่างล็อกอินของแอป Blynk IOT

รูปที่ 11-37 แสดงปุ่ม Add New Device

รูปที่ 11-38 แสดงหน้าต่าง Add Device

รูปที่ 11-39 แสดงหน้าต่าง Add New Device เมื่อเริ่มต้นค้นหา WIO Terminal



รูปที่ 11-40 แสดงหน้าต่างแจ้งการตรวจพบ WIO Terminal ที่รอการเชื่อมต่อ

รูปที่ 11-41 หน้าต่างแสดงรายชื่อ Access Point ที่พบ เพื่อให้ WIO Terminal ทำการเชื่อมต่อเข้าสู่เครือข่ายอินเทอร์เน็ต

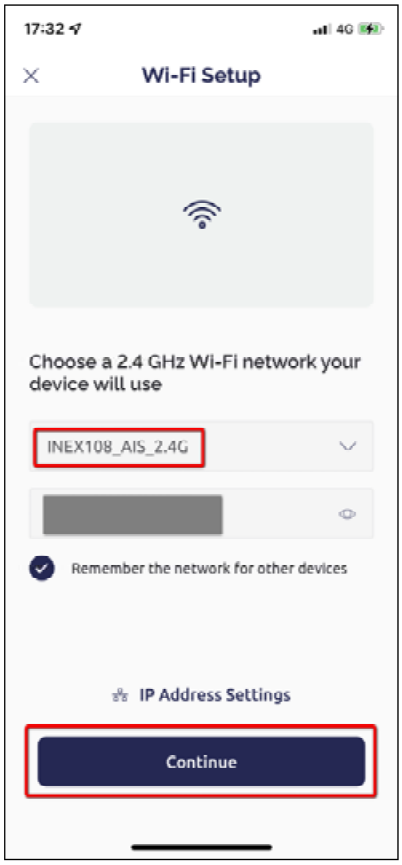
(16) แตะที่ปุ่ม **Add New Device** ตามรูปที่ 11-37

(17) หน้าต่าง **Add Device** แสดงขึ้นมา แตะเลือกรายการ **Connect to Wi-Fi** ตามรูปที่ 11-38

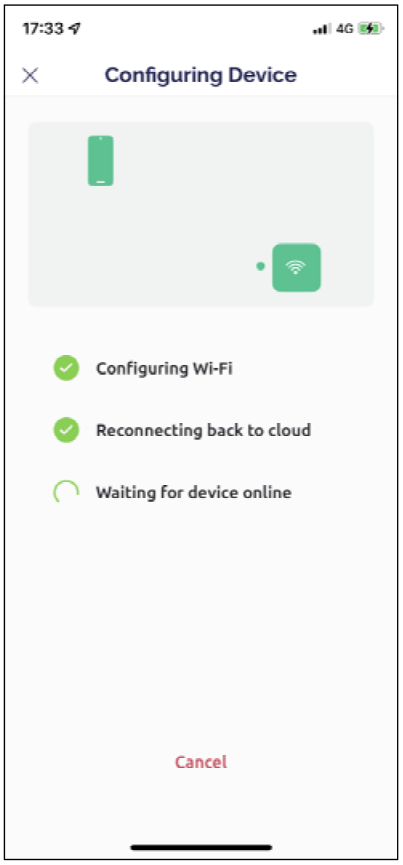
(18) ที่หน้าต่าง **Add New Device** แตะที่ปุ่ม **Start** เพื่อเริ่มต้นค้นหาและเชื่อมต่อไปยัง WIO Terminal ตามรูปที่ 11-39

(19) เมื่อ Blynk App พบ WIO Terminal ที่กำลังรอการเชื่อมต่อจะปรากฏหน้าต่างแจ้งเตือนตามรูปที่ 11-40 แตะปุ่ม **Join** เพื่อทำการเชื่อมต่อ

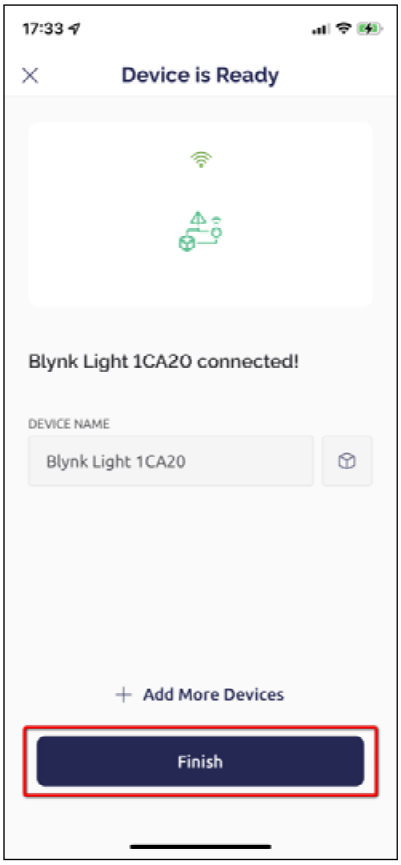
(20) รอจนกระทั่งปรากฏหน้าต่าง **Choose Wi-Fi network** ซึ่งแสดงรายชื่อเครือข่าย WiFi หรือ Access Point จากนั้นให้ผู้พัฒนาแตะเลือก Access Point ที่ต้องการให้ WIO Terminal เชื่อมต่อเข้าสู่เครือข่ายอินเทอร์เน็ตตามรูปที่ 11-41 แตะปุ่ม **Join** เพื่อทำการเชื่อมต่อ



รูปที่ 11-42 แสดงหน้าต่าง Wi-Fi Setup



รูปที่ 11-43 หน้าต่าง Configuring Device แสดงสถานะการเชื่อมต่อเครือข่ายอินเทอร์เน็ตของ WIO Terminal

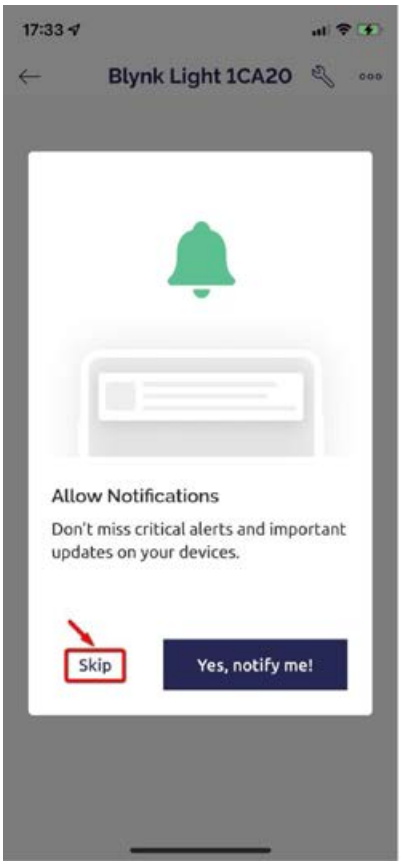


รูปที่ 11-44 หน้าต่าง Device is Ready แสดงความพร้อมในการเชื่อมต่อ

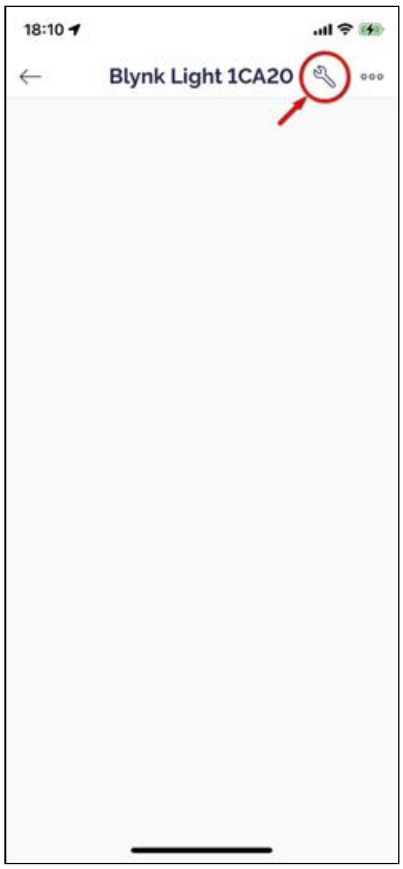
(21) จากนั้นจะปรากฏหน้าต่าง **Wi-Fi Setup** ของอุปกรณ์ Access Point ที่เลือก กรอกรหัสผ่านในช่องด้านล่างถัดมา ในกรณีที่ต้องการให้จำค่าการเชื่อมต่อของ Access Point นี้ให้แตะเลือกรายการ **Remember the network for other devices** จากนั้นแตะที่ปุ่ม **Continue** เพื่อไปยังขั้นตอนต่อไป ตามรูปที่ 11-42

(22) หน้าต่าง **Configuring Device** ปรากฏขึ้นมาแสดงสถานะการเชื่อมต่อเครือข่ายอินเทอร์เน็ตให้รอจนกระทั่ง WIO Terminal เชื่อมต่อกับอินเทอร์เน็ตเสร็จสมบูรณ์ดังรูปที่ 11-43

(23) หน้าต่าง **Device is Ready** ปรากฏขึ้นมาแสดงความพร้อมในการเชื่อมต่อตามรูปที่ 11-44 จากนั้นแตะที่ปุ่ม **Continue**



รูปที่ 11-45 แสดงหน้าต่าง Allow Notifications



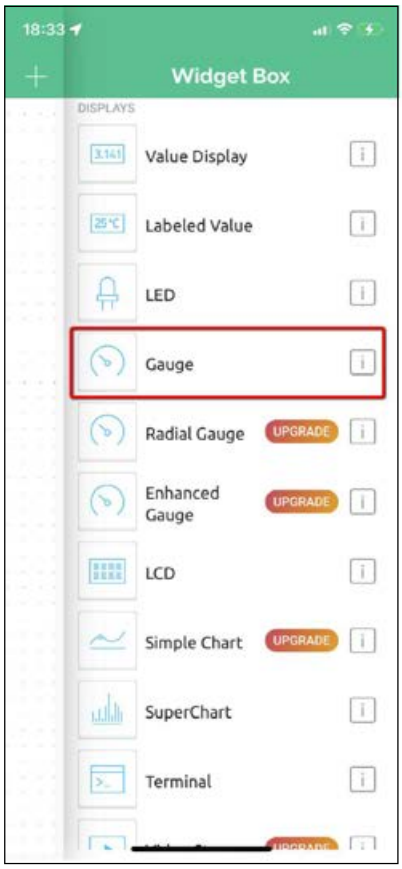
รูปที่ 11-46 การเข้าถึงโหมดการออกแบบผ่านปุ่มเครื่องมือ

รูปที่ 11-47 แสดงโหมดการออกแบบของเทมเพลต Light แตะเครื่องหมายบวกเพื่อเพิ่มวิดเจ็ตสำหรับออกแบบหน้าตาของแอปพลิเคชั่น

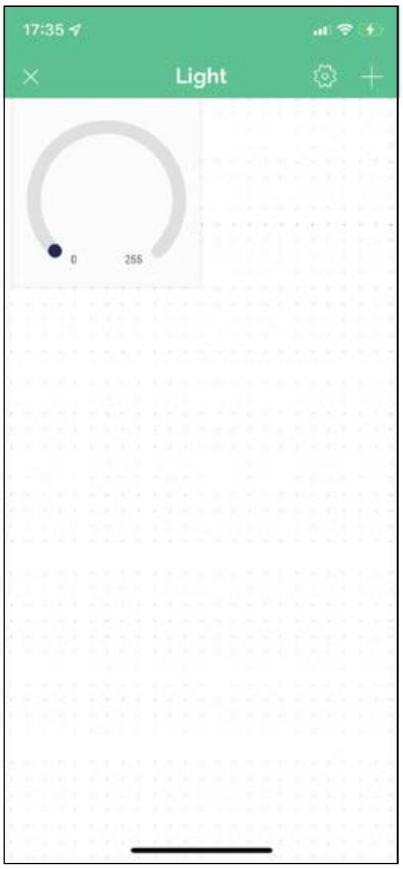
(24) จากนั้นปรากฏหน้าต่างป๊อบอัป **Allow Notifications** ให้แตะปุ่ม **Skip** เพื่อข้ามขั้นตอนการตั้งค่าแจ้งเตือนไปก่อน ตามรูปที่ 11-45

(24) แตะปุ่มเครื่องมือเพื่อเข้าถึงโหมดการออกแบบ ตามรูปที่ 11-46

(25) จากนั้นเข้าสู่โหมดการออกแบบของเทมเพลต **Light** ที่สร้างขึ้นก่อนหน้านี้ ทำการแตะที่เครื่องหมายบวก เพื่อเลือกวิดเจ็ตนำมาออกแบบหน้าตาของแอปพลิเคชั่นภายในเทมเพลตตามรูปที่ 11-47



รูปที่ 11-48 เลือกวิดเจ็ต Gauge



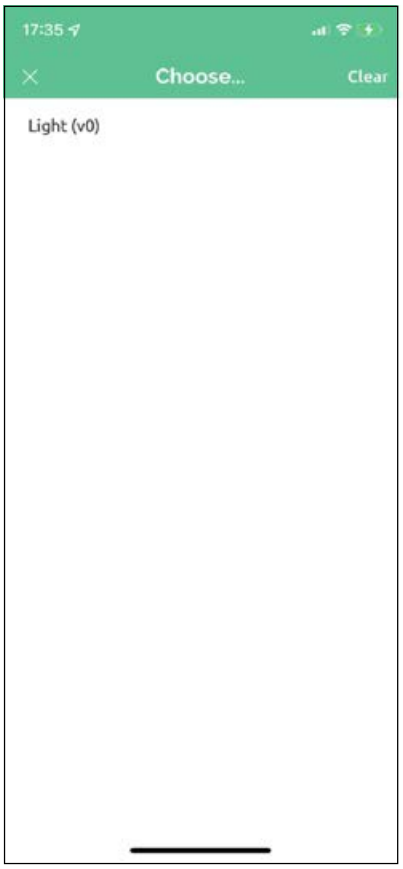
รูปที่ 11-49 แสดงการวางวิดเจ็ต Gauge เพิ่มเข้าไปในเทมเพลต

รูปที่ 11-50 แสดงการกำหนด Datastream

(26) รายการวิดเจ็ต (widget) ปรากกฏขึ้นมา เลื่อนรายการไปทางด้านล่าง จนพบวิดเจ็ต **Gauge** ทำการแตะเลือก **Gauge** สำหรับแสดงค่าระดับแสงเข้าไปวางที่หน้าเทมเพลตตามรูปที่ 11-48

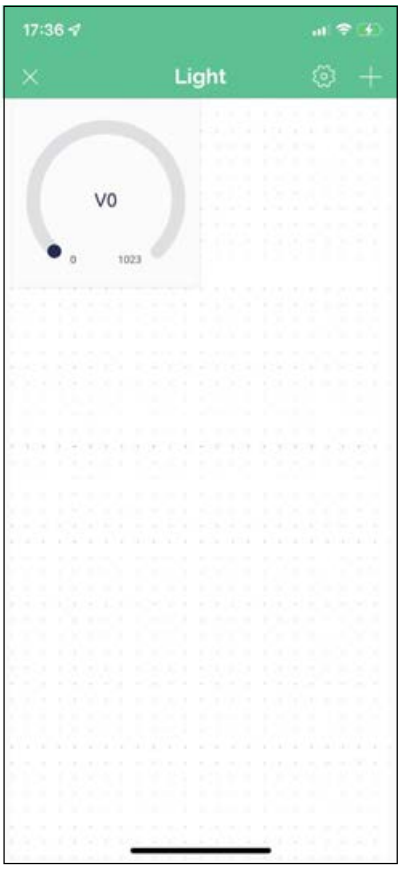
(27) จะได้วิดเจ็ต **Gauge** ที่เลือกมาใช้งานตามรูปที่ 11-49

(28) ทำการกำหนดคุณสมบัติของวิดเจ็ต **Gauge** โดยแตะที่ตัววิดเจ็ต จะปรากฏหน้าต่างกำหนดคุณสมบัติขึ้นมา ในหัวข้อ **DATASTREAM** แตะที่ช่อง **Select Datastream** ตามรูปที่ 11-50



รูปที่ 11-51 แสดงการเลือกช่อง Virtual Port สำหรับวิดเจ็ต Gauge เป็น v0

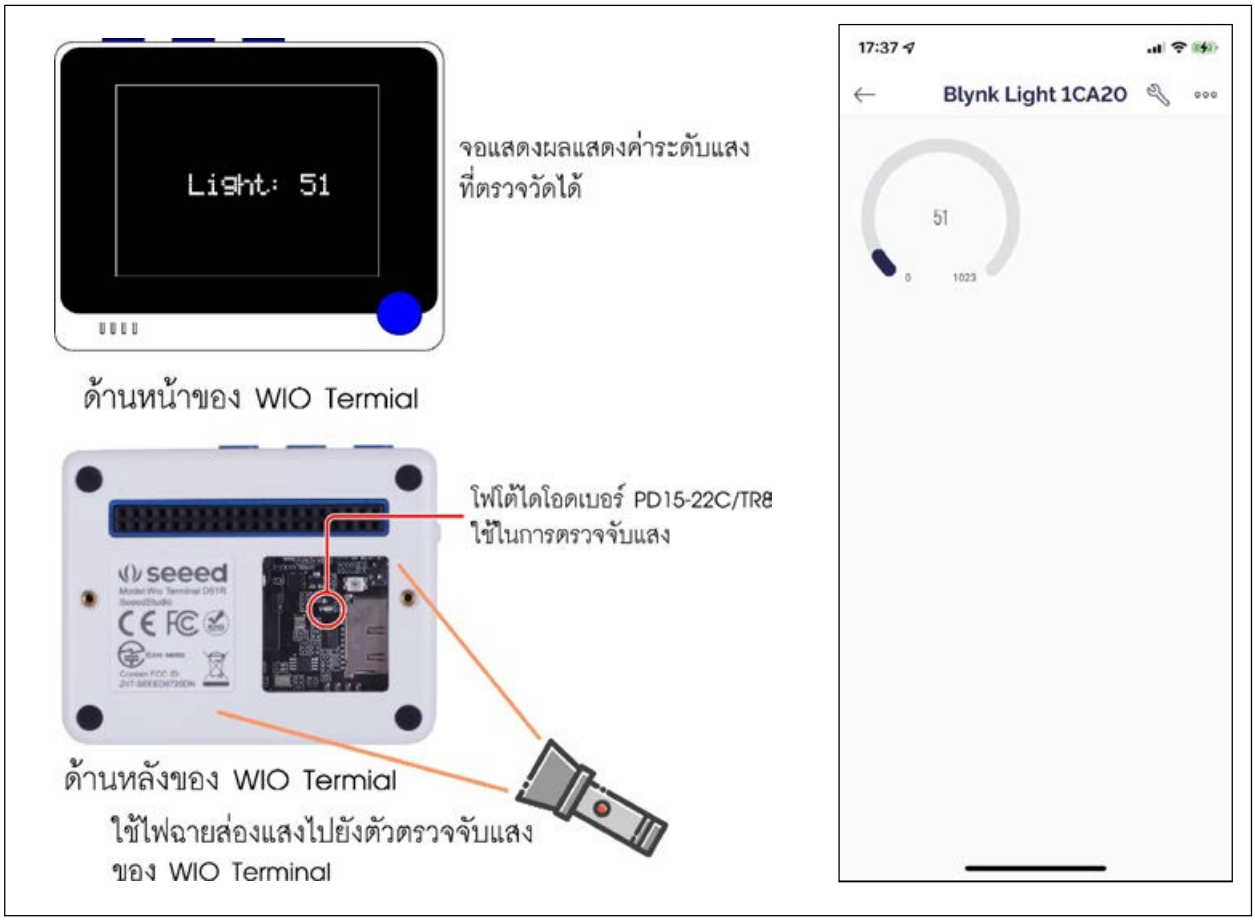
รูปที่ 11-52 แสดงวิดเจ็ต Gauge ถูกกำหนดใช้งาน Datastream ที่ช่อง Virtual Port



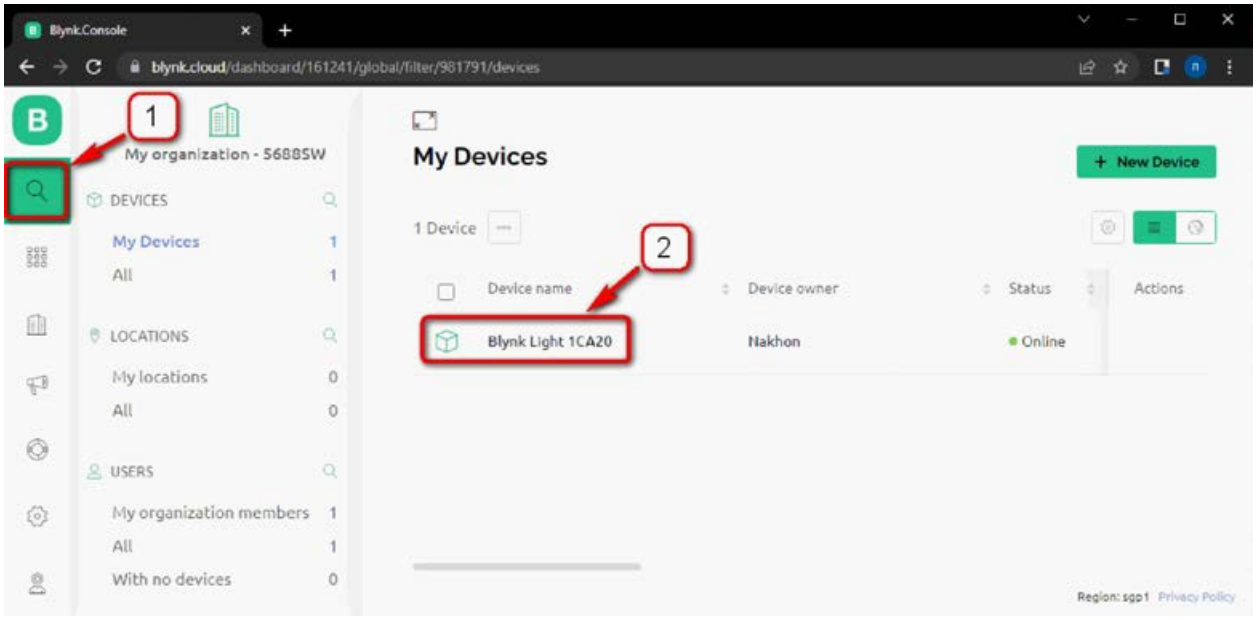
รูปที่ 11-53 ผลลัพธ์ที่สร้างขึ้นของวิดเจ็ต Gauge ในตัวอย่างนี้

(29) ปรากฏหน้าต่าง **Choose…** สำหรับเลือก Virtual Port ที่กำหนดข้อมูลจาก **Web Dashboard** ซึ่งกล่าวถึงก่อนหน้านี้ (จากส่วนที่ 1) เลือกรายการ **Light(v0)** ตามรูปที่ 11-51 ได้ผลลัพธ์ตามรูปที่ 11-52

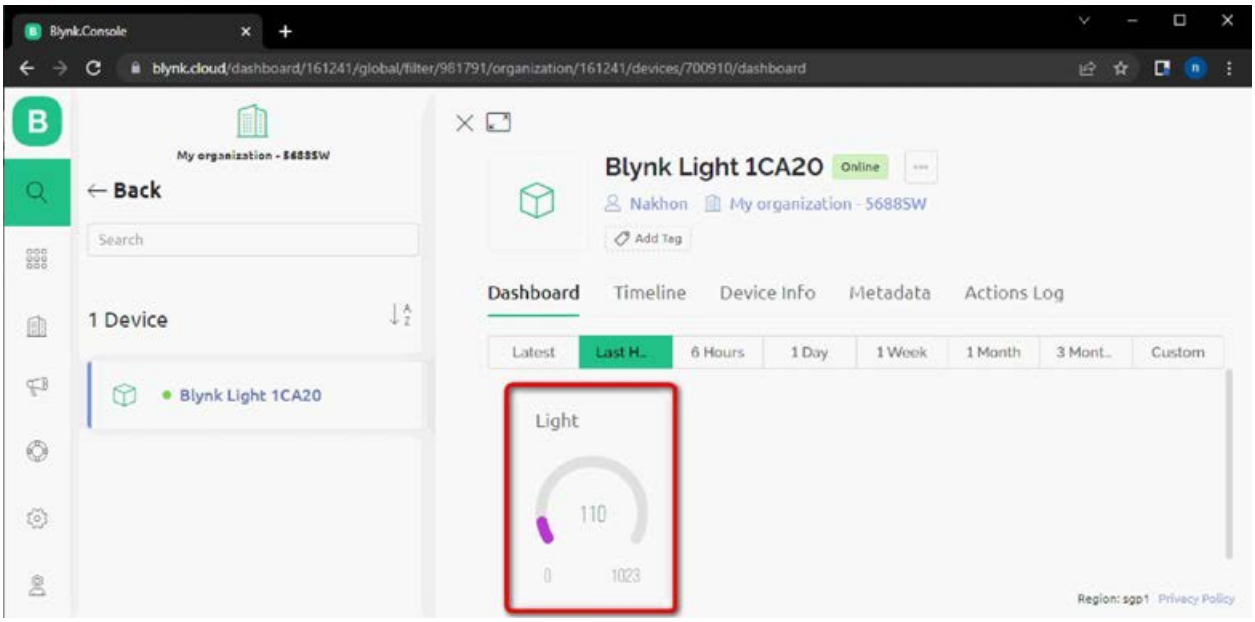
โดยช่วงข้อมูลของ Gauge มีค่า 0 ถึง 1023 ตามที่เคยกำหนดไว้ภายใน Web Dashboard ทำการแตะที่ปุ่มเครื่องหมายกากบาทมุมบนด้านซ้ายเพื่อปิดหน้าต่างนี้ และกลับไปยังหน้าหลักของการทำงานตามรูปที่ 11-53



รูปที่ 11-54 แสดงการทดสอบอ่านค่าระดับแสงของ WIO Terminal มีการแสดงผลการทำงาน ทั้งที่หน้าจอแสดงผลของ WIO Terminal เอง และส่งค่ามาแสดงผลที่วิดเจ็ต Gauge ที่ประสานการทำงานกับ Blynk IoT



รูปที่ 11-55 แสดงการเข้าถึง Device ที่ทำการกำหนดไว้ก่อนหน้านี้ของ Blynk IOT



รูปที่ 11-56 ค่าระดับแสงจาก Wio Terminal ถูกส่งมาแสดงผลที่ Gauge บน Dashboard บนเว็บบราวเซอร์ที่เคยออกแบบเอาไว้

(30) ที่หน้าต่างหลัก วิดเจ็ต **Gauge** จะแสดงค่าความสว่างทันที ดังรูปที่ 11-54

(31) จากนั้นทดสอบเข้าไปยัง Blynk IOT ในส่วนของเว็บบราวเซอร์ที่ล็อกอินอยู่ คลิกที่ปุ่มค้นหา **Device** จากนั้นที่หน้าต่างด้านขวามือจะปรากฏหน้าต่างหัวข้อ **My Devices** ขึ้นมา ให้เลือกเปิดตัวอุปกรณ์ที่ตั้งค่าไว้ก่อนหน้านี้ ซึ่งในที่นี้คือ อุปกรณ์ชื่อ **Blynk Light xxxxx** ตามรูปที่ 11-55

(32) หน้าต่าง **Dashboard** ของ Blynk IoT บนเว็บแสดงค่าระดับแสงที่ตรวจจับได้ผ่านทางวิดเจ็ต **Gauge** ที่ออกแบบไว้ก่อนหน้านี้ (ส่วนที่ 1) ตามรูปที่ 11-56

11.5.2 ตัวอย่างที่ 2 IoT switch

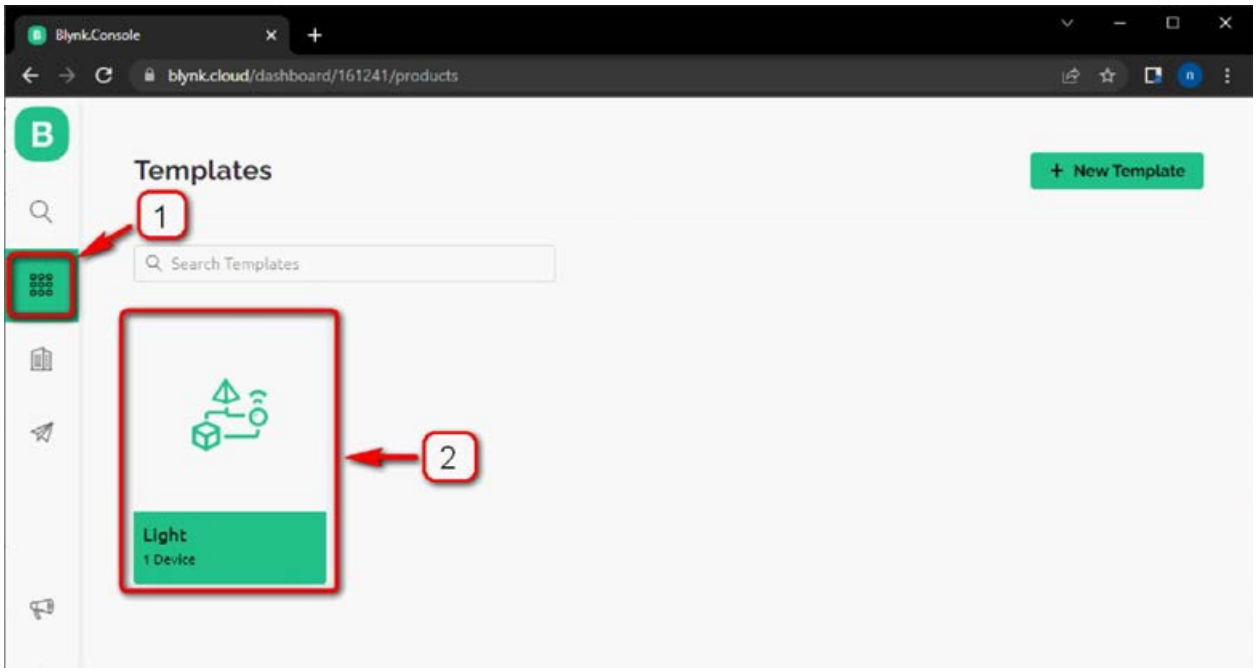
ตัวอย่างถัดมาเป็นการจำลองการสั่งงานเปิด/ปิดไฟบ้านผ่านแอป Blynk IOT อย่างง่าย เป็นการเชื่อมต่อกันระหว่าง WIO Terminal และ Blynk IOT เพื่อควบคุมอุปกรณ์เอาต์พุต โดยในตัวอย่างนี้จะทำการแก้ไขวิดเจ็ตที่มีอยู่เดิมเพื่อให้ได้หน้าตาของ UI (User Intrerface : ส่วนติดต่อกับผู้ใช้งาน) ในการทำงานใหม่ โดยลบวิดเจ็ตเดิมออก แต่ยังคงใช้เทมเพลตเดิม

11.5.2.1 พัฒนาเว็บแดชบอร์ดของ Blynk IoT

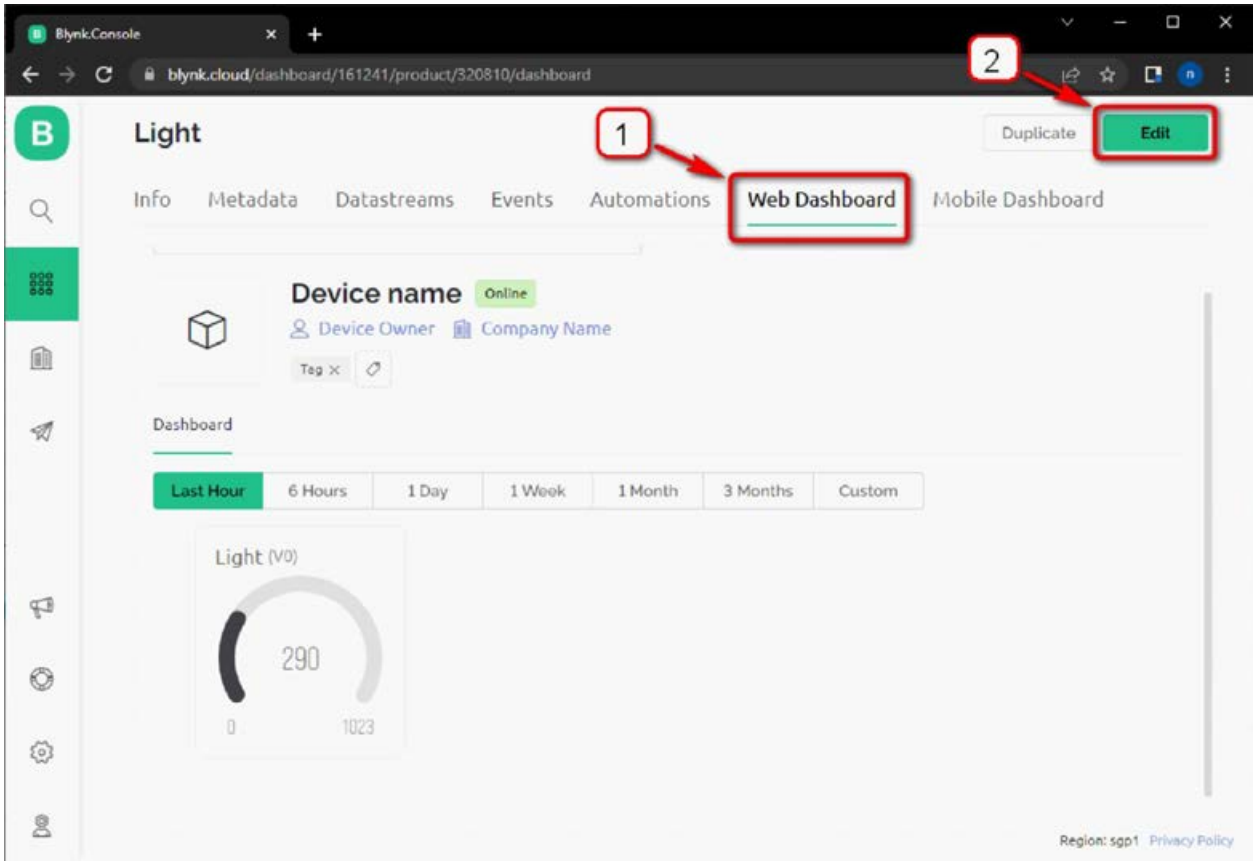
เดิมทีผู้พัฒนาใช้เทมเพลตที่มีวิดเจ็ต **Gauge** 1 ตัว ต้องลบออกเพื่อออกแบบ UI ใหม่ มีขั้นตอนดังนี้

(1) เข้าไปยังเมนู **Template** แล้วคลิกเลือกเทมเพลตที่ชื่อ **Light** ที่มีอยู่เดิม ตามรูปที่ 11-57

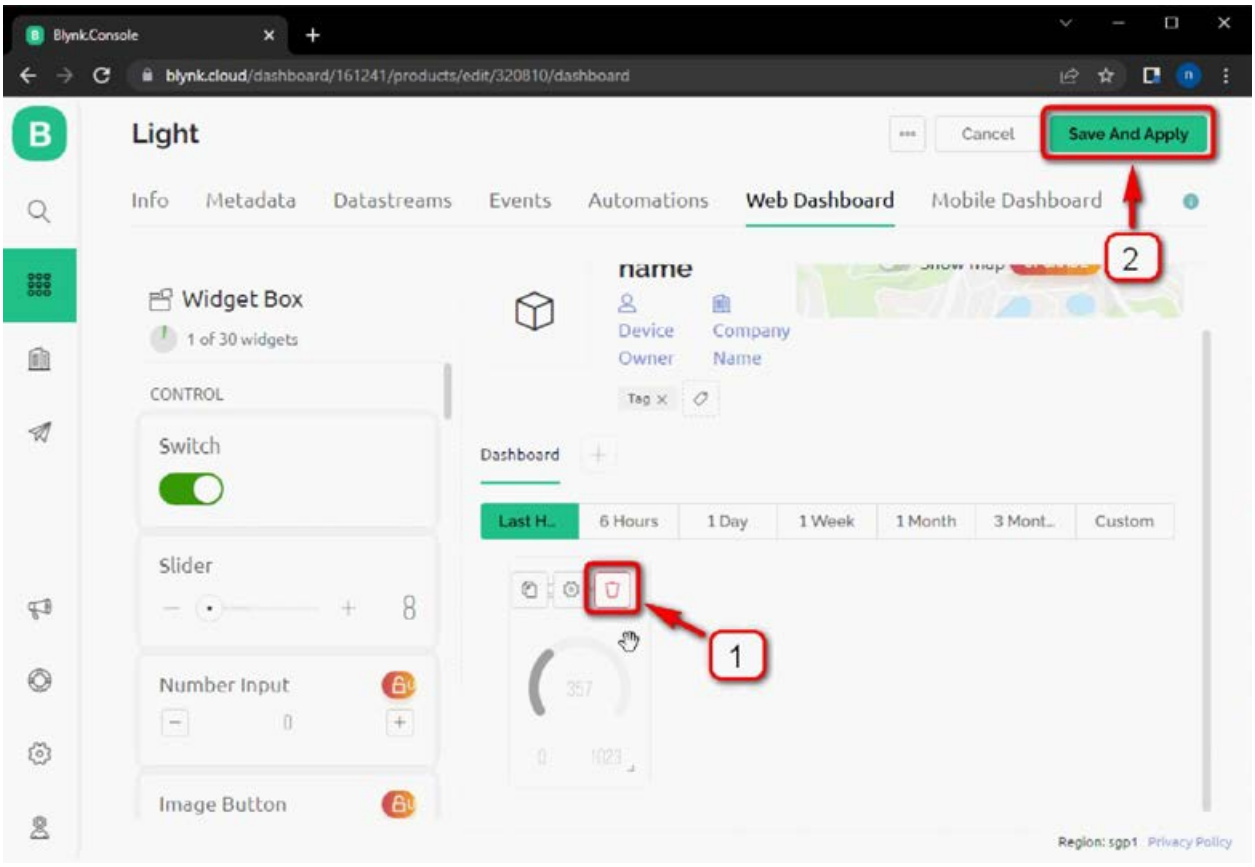
(2) เลือกหัวข้อ **Web Dashboard** แล้วคลิกปุ่ม **Edit** เพื่อเข้าไปแก้ไขตามรูปที่ 11-58



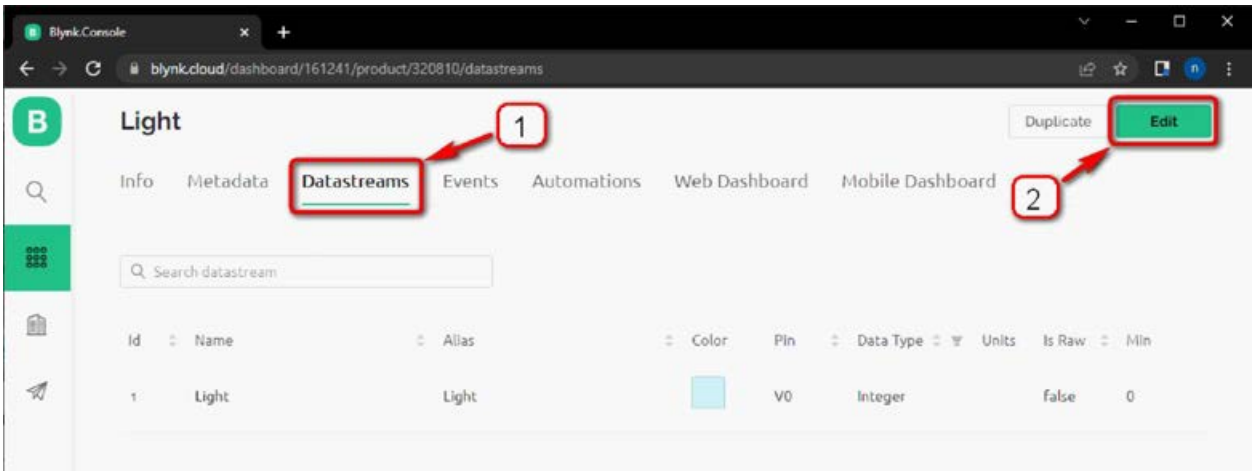
รูปที่ 11-57 แสดงเทมเพลตชื่อ Light ที่มีอยู่เดิม



รูปที่ 11-58 แสดงหัวข้อ Web Dashboard



รูปที่ 11-59 แสดงการลบ Gauge ออกจาก Template



รูปที่ 11-60 แสดงหัวข้อ Datastreams

(3) ใช้เมาส์ชี้ไปยังวิดเจ็ต **Gauge** คลิกเลือกไอคอนถังขยะเพื่อทำการลบวิดเจ็ต **Gauge** ออก จากนั้นคลิกปุ่ม **Save And Apply** ตามรูปที่ 11-59

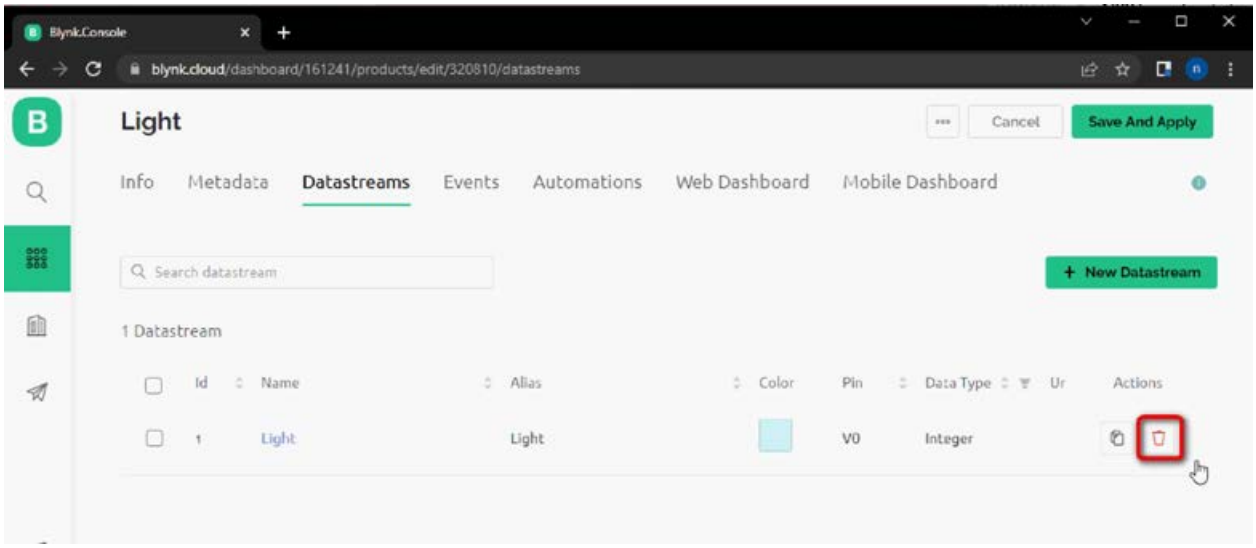
(4) เลือกหัวข้อ **Datastreams** คลิกปุ่ม **Edit** เพื่อเข้าไปลบ **Virtual Pin V0** ที่เคยตั้งค่าไว้ก่อนหน้านี้ ตามรูปที่ 11-60

(5) จากนั้นจะเข้าสู่โหมดการแก้ไขในส่วนของหัวข้อ **Datastreams** ใช้เมาส์ชี้ไปยังรายการ **Light** (การใช้งาน Virtual Pin V0) แล้วคลิกเลือกไอคอนถังขยะเพื่อลบการใช้งาน **Virtual Pin V0** ออกจากระบบ ตามรูปที่ 11-61

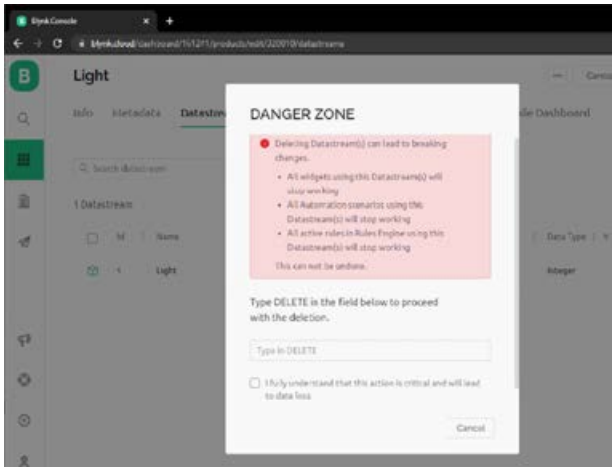
(6) จากนั้นจะปรากฏหน้าต่างแจ้งเตือนว่าต้องการแก้ไขข้อมูลการตั้งค่าในส่วนนี้หรือไม่ เพื่อป้องกันการเผลอลบโดยมิได้ตั้งใจ ตามรูปที่ 11-62

(7) ยืนยันการแก้ไขครั้งนี้ด้วยการกรอกข้อความ **DELETE** ในช่องด้านล่าง และคลิกเลือกรายการยืนยัน ตามด้วยการคลิกปุ่ม **Delete** ตามรูปที่ 11-63

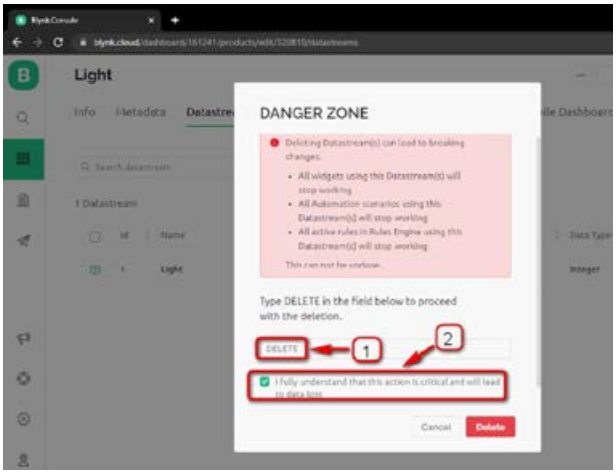
(8) การตั้งค่าใช้งาน **Virtual Pin V0** จะถูกลบออกจากหัวข้อ **Datastreams** ตามรูปที่ 11-64



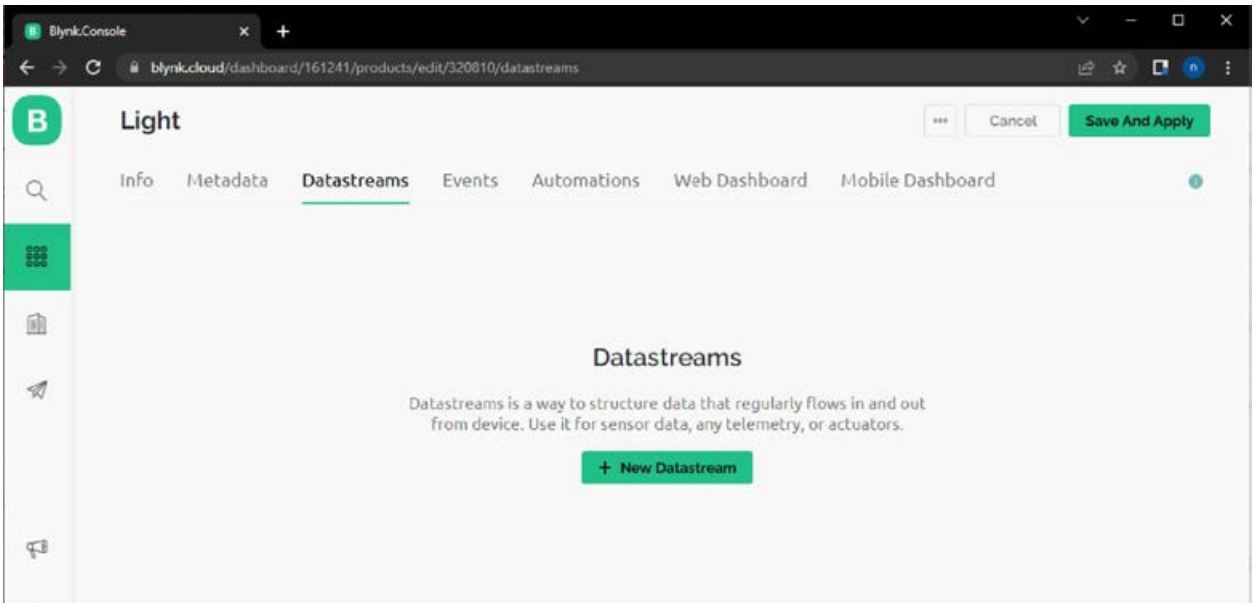
รูปที่ 11-61 แสดงไอคอนถังขยะเพื่อลบการใช้งาน Virtual Pin V0



รูปที่ 11-62 หน้าต่างแจ้งเตือนการแก้ไขข้อมูล



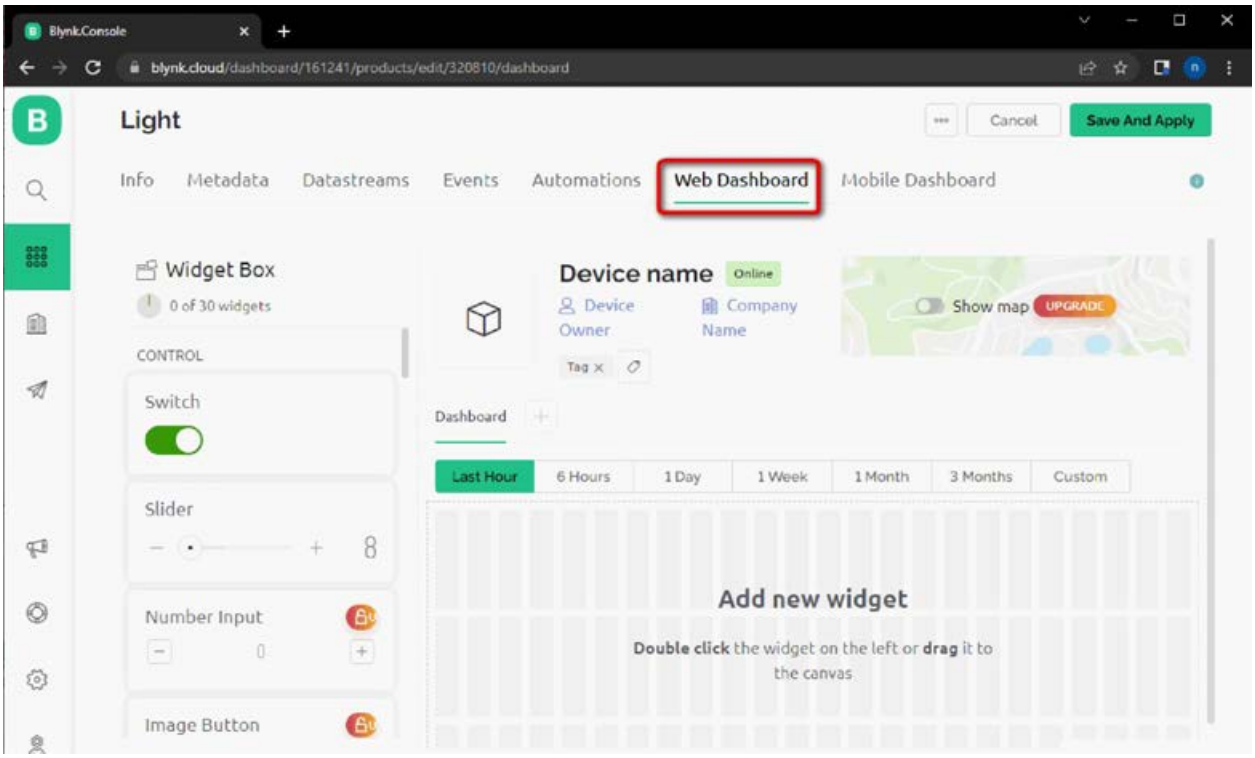
รูปที่ 11-63 แสดงขั้นตอนการยืนยันแก้ไขข้อมูล



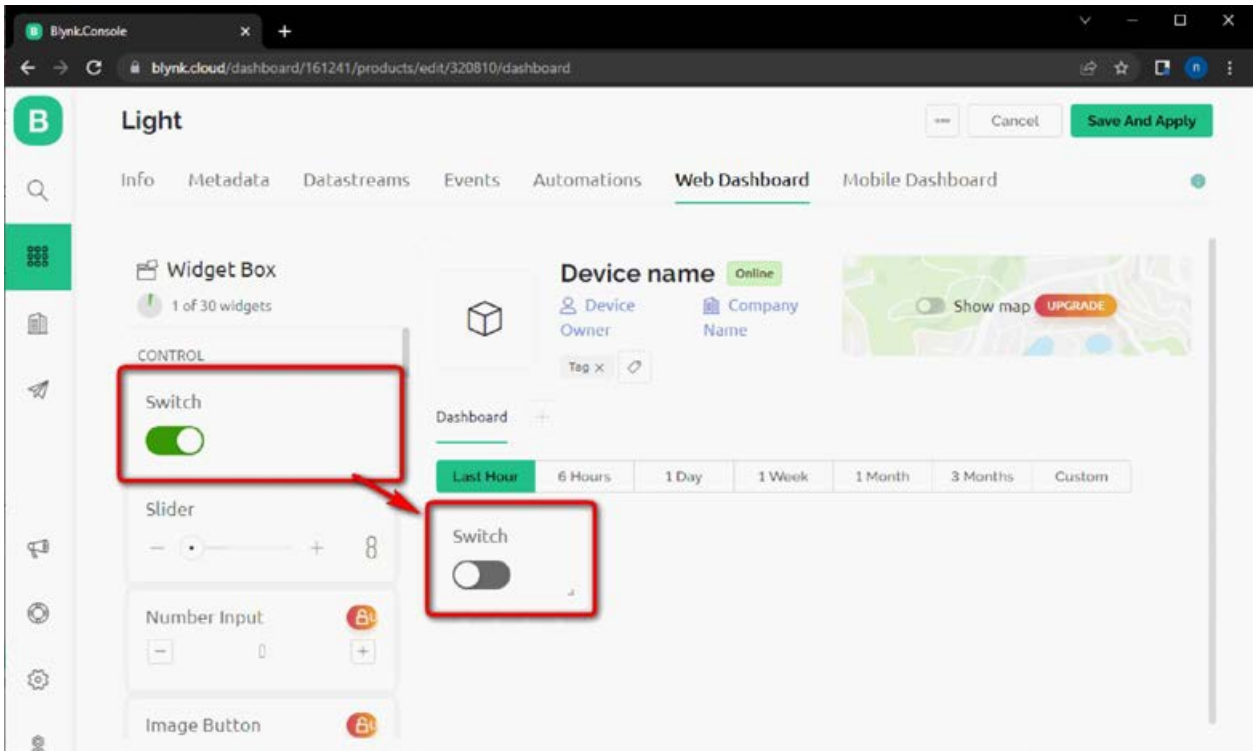
รูปที่ 11-64 แสดงให้เห็นว่า รายการ Virtual Pin V0 ถูกลบออกไปจากหัวข้อ Datastreams แล้ว

(9) เลือกหัวข้อ **Web Dashboard** ตามรูปที่ 11-65

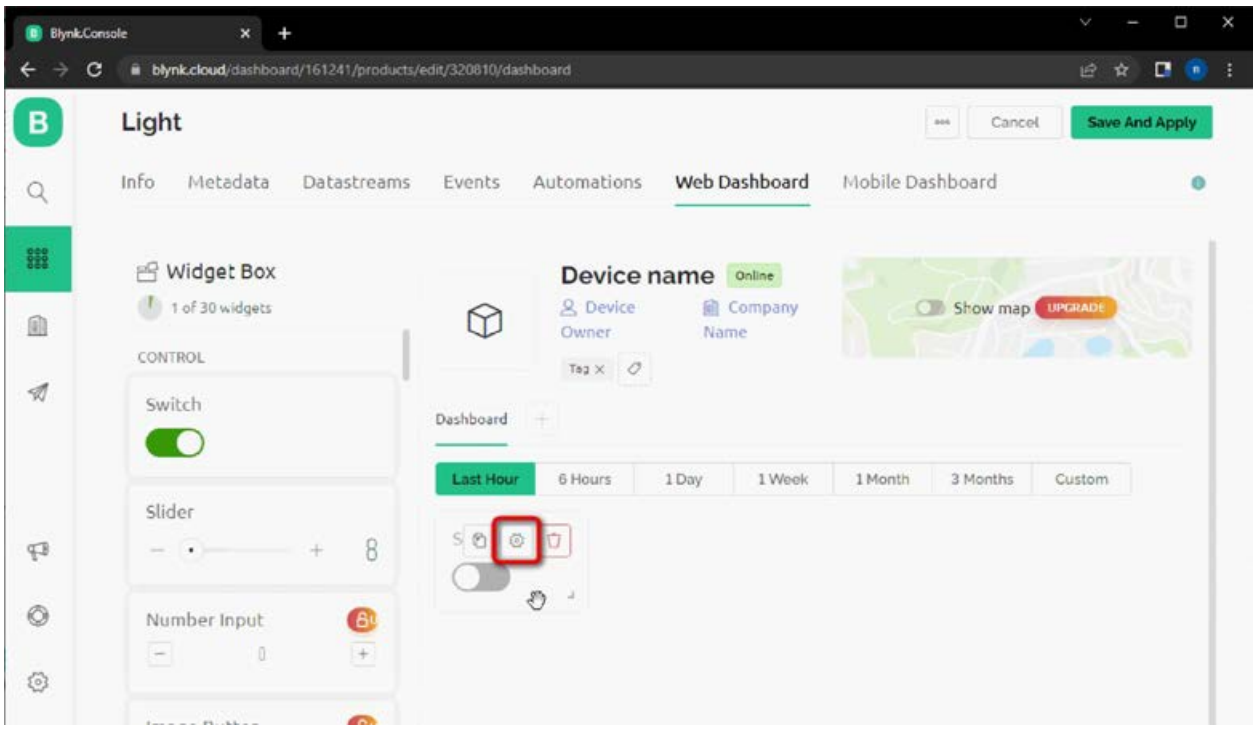
(10) ในหัวข้อ **Widget Box** เพิ่มวิดเจ็ต **Switch** เข้ามายัง **Dashboard** ตามรูปที่ 11-66



รูปที่ 11-65 แสดงหัวข้อ Web Dashboard



รูปที่ 11-66 แสดงการวางวิดเจ็ต Switch เพิ่มเข้ามาใน Dashboard



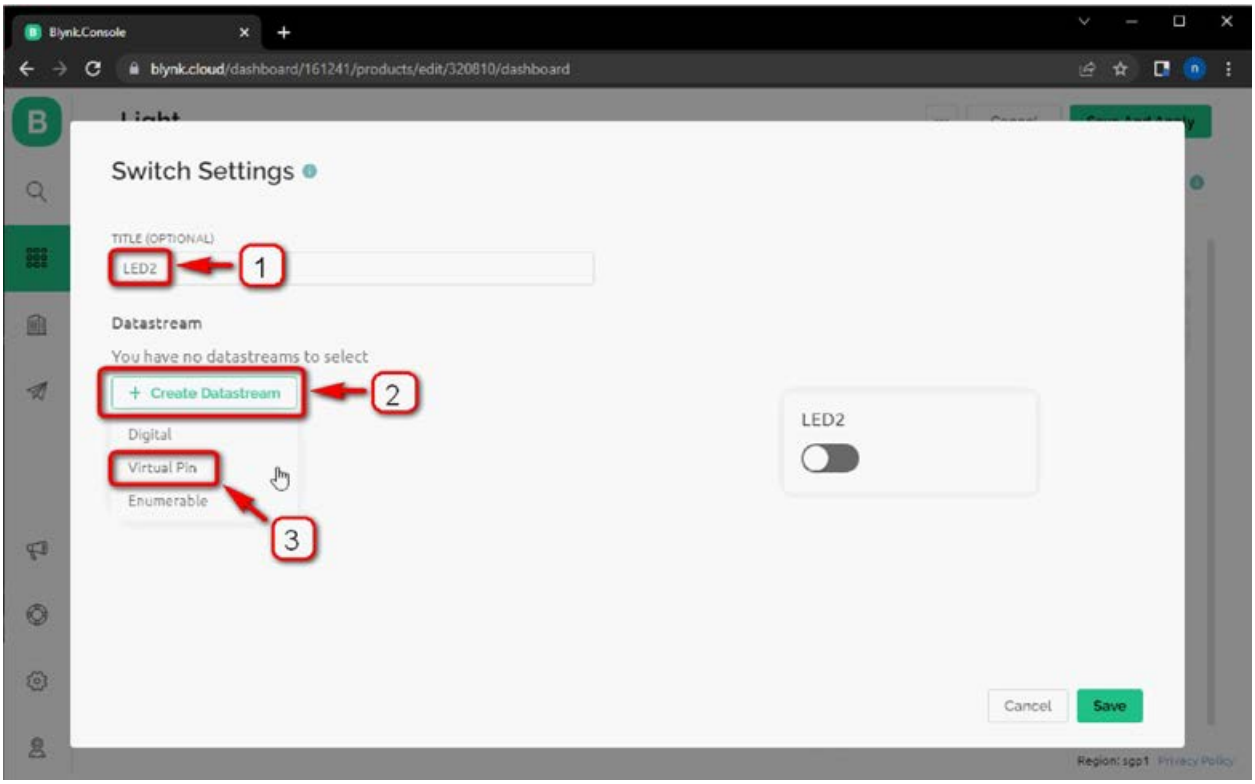
รูปที่ 11-67 แสดงไอคอนรูปฟันเฟืองสำหรับการตั้งค่าวิดเจ็ต Switch

(11) ใช้เมาส์ชี้ไปยังวิดเจ็ต **Switch** ที่หน้า **Dashboard** แล้วคลิกเลือกไอคอนรูปฟันเฟืองเพื่อทำการตั้งค่า ตามรูปที่ 11-67

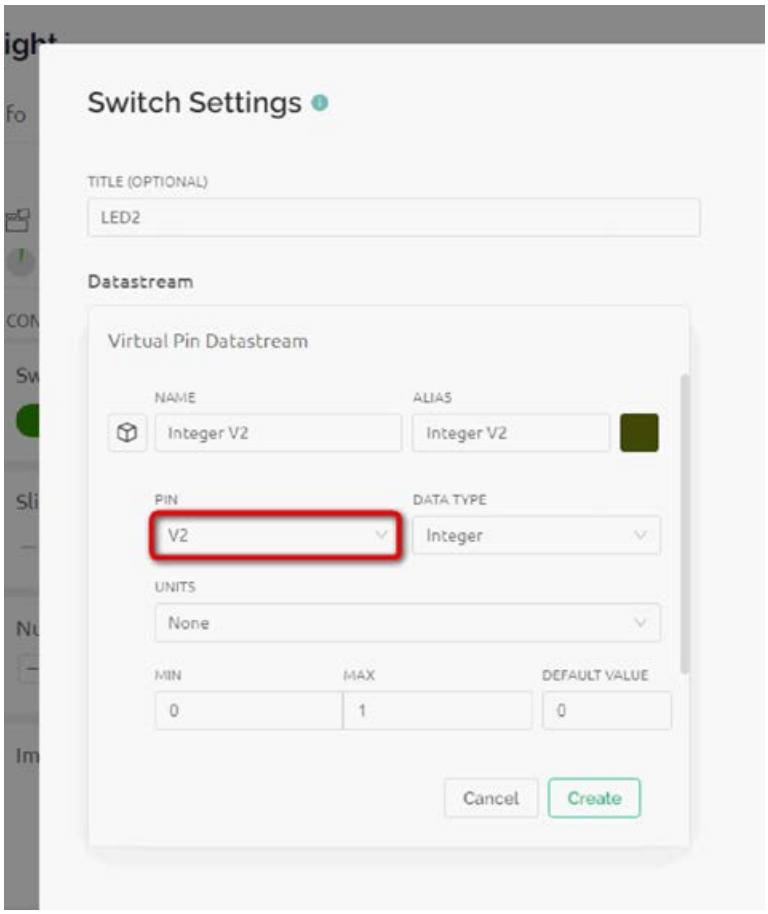
(12) หน้าต่าง **Switch Settings** ปรากฏขึ้นมาสำหรับกำหนดการใช้งานวิดเจ็ต **Switch** สำหรับ LED2 ตามรูปที่ 11-68

(12.1) ที่หัวข้อหัวข้อ **TITLE (OPTIONAL)** กำหนดชื่อเป็น ***LED2***

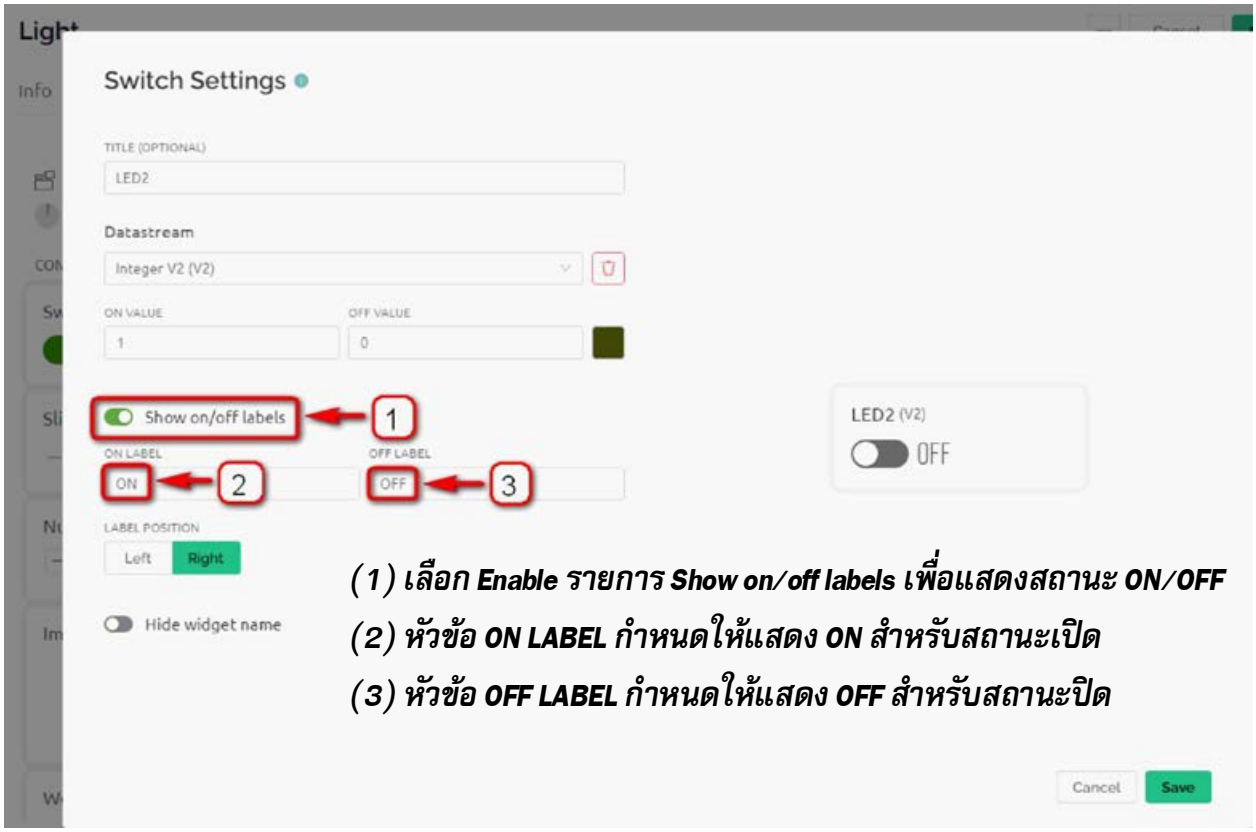
(12.2) กดปุ่ม **Create Datastream** แล้วเลือกรายการ ***Virtual Pin***



รูปที่ 11-68 แสดงหน้าต่าง Switch Settings



รูปที่ 11-69 แสดงการตั้งค่า Virtual Pin ของวิดเจ็ต Switch ตัวแรกเป็น V2



(1) เลือก Enable รายการ Show on/off labels เพื่อแสดงสถานะ ON/OFF

(2) หัวข้อ ON LABEL กำหนดให้แสดง ON สำหรับสถานะเปิด

(3) หัวข้อ OFF LABEL กำหนดให้แสดง OFF สำหรับสถานะปิด

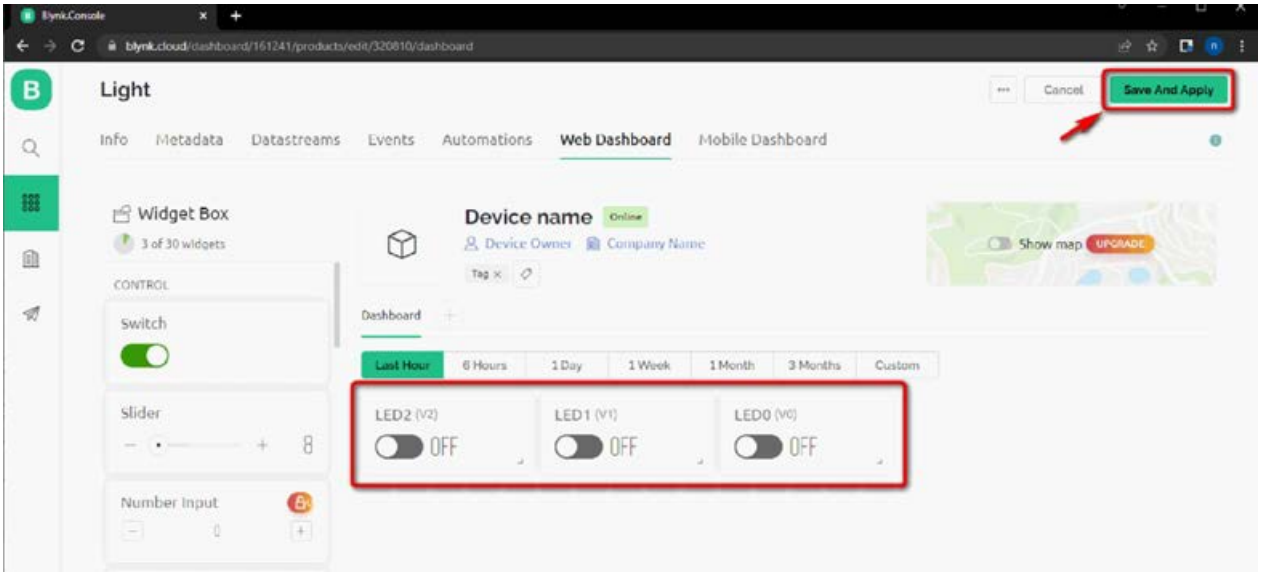
รูปที่ 11-70 การกำหนดสถานะต่าง ๆ ของวิดเจ็ต Switch

(13) จากนั้นจะปรากฏเมนูส่วนขยายตามรูปที่ 11-69 ในหัวข้อ PIN ตั้งค่าเป็น V2 จากนั้นกดปุ่ม **Create** เพื่อยืนยัน

(14) จากนั้นกลับมายังหน้าหลักของ **Switch Settings** อีกครั้ง ทำการเลือก Enable รายการ **Show on/off labels** เพื่อแสดงสถานะ **ON/OFF** ของวิดเจ็ต **Switch** ตามรูปที่ 11-70 โดยตั้งค่าดังนี้

- หัวข้อ **ON LABEL** กำหนดให้แสดงข้อความ ***ON*** สำหรับสถานะ ON หรือเปิด
- หัวข้อ **OFF LABEL** กำหนดให้แสดงข้อความ ***OFF*** สำหรับสถานะ OFF หรือปิด

(15) เพิ่มวิดเจ็ต **Switch** เข้ามายัง **Dashboard** อีก 2 ตัว ตั้งชื่อเป็น **LED1** ตั้งค่าใช้งานเป็น **Virtual Pin V1** และ **LED0** ตั้งค่าใช้งานเป็น **Virtual Pin V0** ได้ผลลัพธ์ตามรูปที่ 11-71 จากนั้นกดปุ่ม **Save And Apply** เพื่อยืนยันอีกครั้ง



รูปที่ 11-71 แสดงวิดเจ็ต Switch 2 ตัว, LED1 และ LED0 ที่ถูกเพิ่มเข้ามายัง Dashboard

11.5.2.2 พัฒนาโปรแกรมสำหรับ WIO Terminal บน Arduino IDE

(16) เปิดโปรแกรม Arduino IDE สร้างโค้ดตามโปรแกรมที่ 11-2 ตั้งชื่อไฟล์เป็น **Blynk_IOT_LED_Demo.ino** แล้วทำการอัปโหลดไปยัง WIO Terminal

สำหรับ 2 บรรทัดแรกของโปรแกรมในส่วนของ **BLYNK_TEMPLATE_ID** และ **BLYNK_DEVICE_NAME** ผู้พัฒนาจำเป็นต้องกำหนดให้ตรงกับเทมเพลตที่สร้างไว้ก่อนหน้านี้ โดยเข้าไปคัดลอกค่ารหัสดังกล่าวจากหัวข้อของเทมเพลตตามรูปที่ 11-26 (หัวข้อ 11.5.1 ก่อนหน้านี้)

```
#define BLYNK_TEMPLATE_ID “xxxxxxx”          // สำหรับกำหนดค่ารหัส TEMPLATE ID
#define BLYNK_DEVICE_NAME “xxxxx”            // สำหรับกำหนดค่ารหัส DEVICE NAME
#define BLYNK_FIRMWARE_VERSION  “0.1.0”      // สำหรับกำหนดค่ารหัส FIRMWARE VERSION
#define BLYNK_PRINT Serial
#define APP_DEBUG
#include “BlynkEdgent.h”                      // ผนวกไลบรารีสำหรับติดต่อ Blynk IOT
#include <TFT_eSPI.h>                        // ผนวกไลบรารีควบคุมการแสดงผล LCD
TFT_eSPI tft;                  // สร้างออปเจ็กต์ชื่อ tft จากคลาส TFT_eSPI สำหรับควบคุมการแสดงผล
BlynkTimer timer;              // สร้างออปเจ็กต์ BlynkTimer ชื่อ timer
bool statusLed[]={false,false,false}; // กำหนดตัวแปรเก็บค่าสถานะการติด/ดับของ LED (true=ติด,false=ดับ)
void updateStatus()            // ฟังก์ชั่นอัปเดทสถานะการติด/ดับของ LED แต่ละดวง
{
   tft.drawString(“LED Demo ”, 100, 20);          // แสดงข้อความพร้อมทำงาน
   if(statusLed[0])                               // LED0 ติดสว่าง ?
        tft.fillCircle(240,140,30,TFT_GREEN);     // แสดงกราฟิก LED0 ติดสว่าง
   else
        tft.fillCircle(240,140,30,TFT_DARKGREY);  // แสดงกราฟิก LED0 ดับ
   if(statusLed[1])                               // LED1 ติดสว่าง ?
        tft.fillCircle(160,140,30,TFT_GREEN);     // แสดงกราฟิก LED1 ติดสว่าง
   else
        tft.fillCircle(160,140,30,TFT_DARKGREY);  // แสดงกราฟิก LED1 ดับ
   if(statusLed[2])                               // LED2 ติดสว่าง ?
        tft.fillCircle(80,140,30,TFT_GREEN);      // แสดงกราฟิก LED2 ติดสว่าง
   else
        tft.fillCircle(80,140,30,TFT_DARKGREY);   // แสดงกราฟิก LED2 ดับ
```

โปรแกรมที่ 11-2 ไฟล์ Blynk_IoT_LED_Demo.ino โปรแกรมสำหรับทดสอบใช้งาน WIO Terminal ควบคุม LED ผ่านแอป Blynk IoT (มีต่อ)

```
    Blynk.virtualWrite(V0, statusLed[0]);          // ส่งข้อมูลสถานะการติด/ดับของ LED0 ไปยัง Blynk App
    Blynk.virtualWrite(V1, statusLed[1]);          // ส่งข้อมูลสถานะการติด/ดับของ LED1 ไปยัง Blynk App
    Blynk.virtualWrite(V2, statusLed[2]);          // ส่งข้อมูลสถานะการติด/ดับของ LED2 ไปยัง Blynk App
}
void scanButton()   // ฟังก์ชั่นตรวจจับการกด BUTTON A,B และ C
{
    if (digitalRead(WIO_KEY_A) == LOW)             // BUTTON A ถูกกด?
    {
        statusLed[0] = !statusLed[0];              // กลับสถานะ LED0 เมื่อ BUTTON A ถูกกด
        updateStatus();                            // ปรับปรุงสถานะการติด/ดับของ LED แต่ละดวง
    }
    else if (digitalRead(WIO_KEY_B) == LOW)        // BUTTON B ถูกกด?
    {
        statusLed[1] = !statusLed[1];              // กลับสถานะ LED1 เมื่อ BUTTON A ถูกกด
        updateStatus();                            // ปรับปรุงสถานะการติด/ดับของ LED แต่ละดวง
    }
    else if (digitalRead(WIO_KEY_C) == LOW)        // BUTTON C ถูกกด?
    {
        statusLed[2] = !statusLed[2];              // กลับสถานะ LED2 เมื่อ BUTTON A ถูกกด
        updateStatus();                            // ปรับปรุงสถานะการติด/ดับของ LED แต่ละดวง
    }
}
BLYNK_WRITE(V2) // Active for get data from Blynk app
{
    statusLed[2] = param[0].asInt();
    updateStatus();                                // ปรับปรุงสถานะการติด/ดับของ LED แต่ละดวง
}
BLYNK_WRITE(V1)
{
    statusLed[1] = param[0].asInt();
    updateStatus();                                // ปรับปรุงสถานะการติด/ดับของ LED แต่ละดวง
}
BLYNK_WRITE(V0)
{
    statusLed[0] = param[0].asInt();
    updateStatus();                                // ปรับปรุงสถานะการติด/ดับของ LED แต่ละดวง
}
```

โปรแกรมที่ 11-2 ไฟล์ Blynk_IoT_LED_Demo.ino โปรแกรมสำหรับทดสอบใช้งาน WIO Terminal ควบคุม LED ผ่านแอป Blynk IoT (มีต่อ)

```
void setup()
{
   pinMode(WIO_KEY_A, INPUT_PULLUP);      // กำหนดใช้งานพินที่เชื่อมต่อกับ BUTTON A เป็นอินพุต
   pinMode(WIO_KEY_B, INPUT_PULLUP);      // กำหนดใช้งานพินที่เชื่อมต่อกับ BUTTON B เป็นอินพุต
   pinMode(WIO_KEY_C, INPUT_PULLUP);      // กำหนดใช้งานพินที่เชื่อมต่อกับ BUTTON C เป็นอินพุต
   tft.begin();                           // เริ่มต้นการทำงานของ LCD
   tft.setRotation(3);                    // กำหนดทิศทางการหมุนหน้าจอของ LCD
   tft.fillScreen(TFT_BLACK);             // ระบายสีดำเป็นพื้นหลังหน้าจอ LCD
   tft.setTextColor(TFT_WHITE,TFT_BLACK); // กำหนดสีตัวอักษรสีขาวพื้นหลังสีดำ
   tft.setTextSize(2);                    // กำหนดขนาดฟอนต์เป็น 2
   tft.drawString("LED0", 220, 80);       // ข้อความลาเบลประจำตัว LED0
   tft.drawString("LED1", 140, 80);       // ข้อความลาเบลประจำตัว LED1
   tft.drawString("LED2", 60, 80);        // ข้อความลาเบลประจำตัว LED2
   tft.setTextColor(TFT_ORANGE,TFT_BLACK); // กำหนดสีตัวอักษรสีส้มพื้นหลังสีดำ
   tft.setTextSize(3);                       // กำหนดขนาดฟอนต์เป็น 3
   tft.drawString("Wait...", 100, 20);       // แสดงข้อความแจ้งเตือนรอการเชื่อมต่อให้เรียบร้อย
   Serial.begin(115200);   // กำหนดการใช้ Serial Monitor ในกรณีที่แจ้งเตือนการเชื่อมต่อเซอร์ฟเวอร์ Blynk
   BlynkEdgent.begin();    // เริ่มต้นเชื่อมต่อไปยังเซอร์ฟเวอร์ Blynk IOT
   timer.setInterval(500L, scanButton);   // เปิดใช้งานไทเมอร์อินเตอร์รัปต์ทุกๆ 0.5 วินาที
                                          // เรียกใช้งานไปยังฟังก์ชั่น scanButton
   updateStatus();                        // ปรับปรุงสถานะการติด/ดับของ LED ในตอนเริ่มต้น
 }
void loop()
{
   BlynkEdgent.run();                     // สั่งรัน Blynk IoT
   timer.run();                           // สั่งรันไทเมอร์
}
```

การทำงานของโปรแกรม

การทำงานของโปรแกรมหลักหลังจากเชื่อมต่กับเซิร์ฟเวอร์ Blynk ได้สำเร็จ จะเกิดอินเตอร์รัปต์จากไทเมอร์ทุกๆ 0.5 วินาที โปรแกรมจะกระโดดไปทำงานที่ฟังก์ชั่น **scanButton** ตามที่กำหนด เพื่อตรวจสอบการกด BUTTON A , B และ C ของ WIO Terminal พร้อมกับปรับปรุงการวาดสถานะของ LED แต่ละดวงที่จอแสดงผลเมื่อตรวจพบว่า BUTTON ถูกกด รวมถึงส่งข้อมูลสถานะไปปรับปรุงที่แอป Blynk IoT ด้วยการเรียกฟังก์ชั่น **updateStatus** ในกรณีที่ผู้ใช้งานกดปุ่ม BUTTON ตำแหน่ง LED0 ถึง LED2 ที่แอป Blynk IoT จะเป็นการส่งค่าสถานะการควบคุมกลับมายัง WIO Terminal ด้วยเช่นกัน

โปรแกรมที่ 11-2 ไฟล์ Blynk_IoT_LED_Demo.ino โปรแกรมสำหรับทดสอบใช้งาน WIO Terminal ควบคุม LED ผ่านแอป Blynk IoT (จบ)

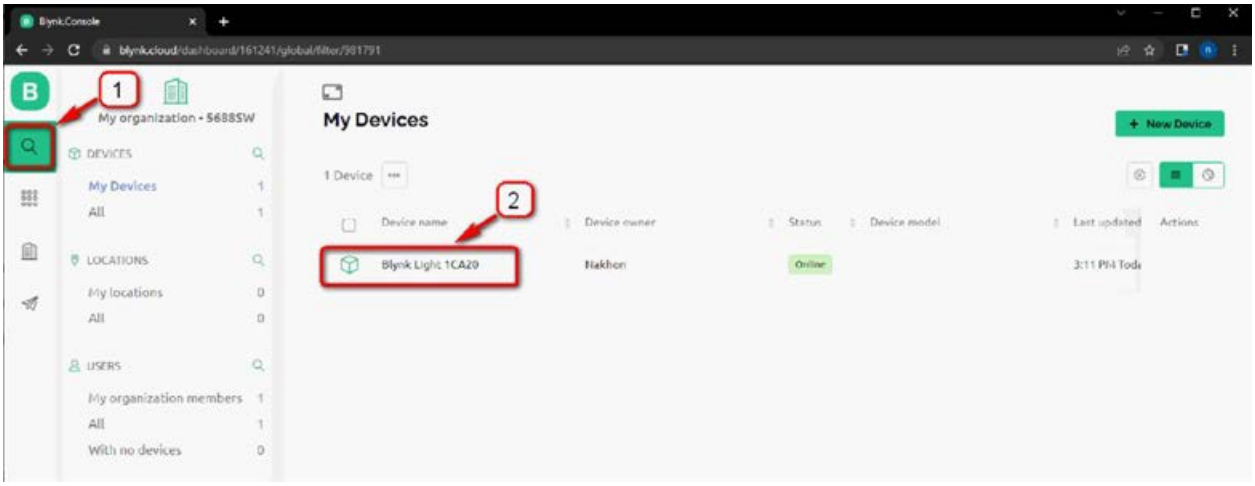
11.5.2.3 ทดสอบการทำงาน

(ก) ทดสอบการทำงานเทียบกันระหว่าง WIO Terminal และแดชบอร์ด Blynk IoT

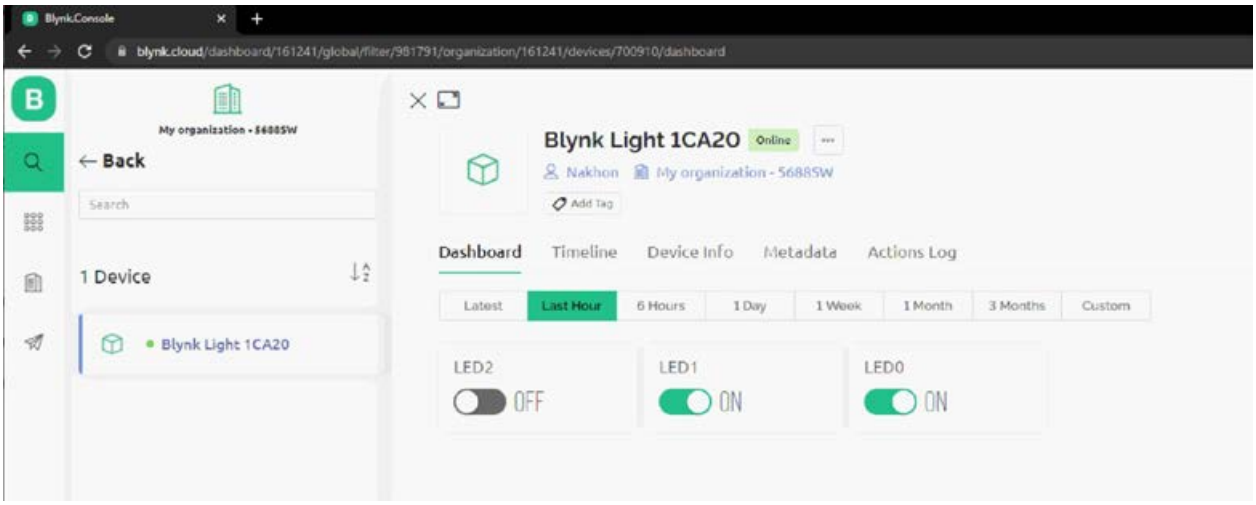
ในขณะที่โปรแกรมทำงานและ WIO Terminal เชื่อมต่อกับ Blynk IoT เรียบร้อยแล้ว ทำการทดสอบดังนี้

(1) ไปที่หน้า **My Devices** จากนั้นคลิกเลือกรายการอุปกรณ์ **Blynk Light xxx** (ในขณะนี้มีสถานะ **Online** โดยสังเกตที่ **Status**) ตามรูปที่ 11-72

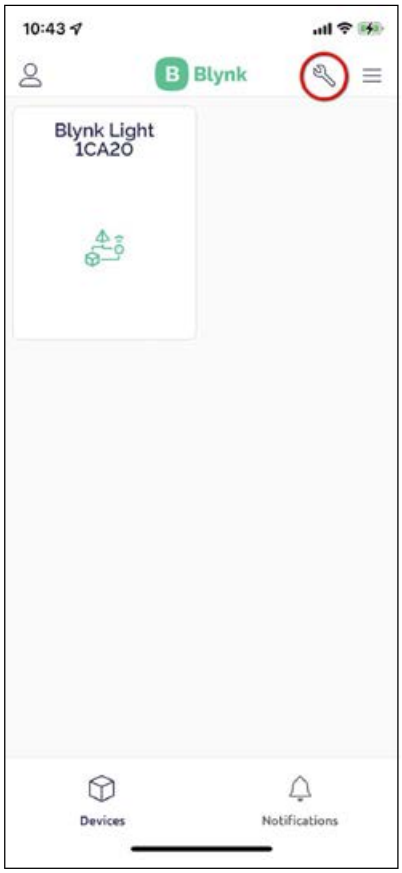
(2) จากนั้นที่หน้า **Dashboard** วิดเจ็ต **Switch** ทั้ง 3 ตัวจะแสดงสถานะตรงกับ LED2 ถึง LED0 ที่ WIO Terminal ตามรูปที่ 11-73



รูปที่ 11-72 แสดงรายการอุปกรณ์ Blynk Light xxx เพื่อเข้าไปยังหน้า Dashboard

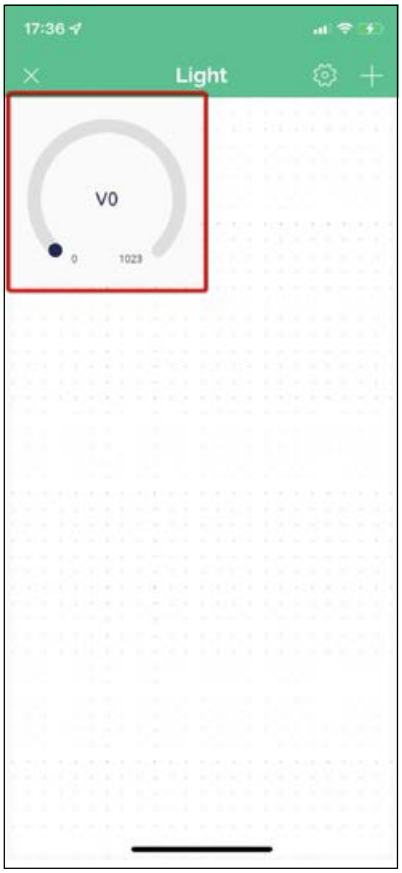


รูปที่ 11-73 วิดเจ็ต Switch ทั้ง 3 ตัวที่หน้า Dashboard แสดงสถานะตรงกับ LED2 ถึง LED0 บนจอแสดงผลของ WIO Terminal



รูปที่ 11-74 แสดงเทมเพลตที่เคยสร้างไว้ก่อนหน้านี้

รูปที่ 11-75 แสดง Template Light ที่ต้องการเข้าไปแก้ไข UI



รูปที่ 11-76 แสดง Gauge ที่มีอยู่ภายในเทมเพลต

(3) ทดสอบกดปุ่ม **BUTTON A** ของ WIO Terminal

รูป LED ที่ตำแหน่ง LED0 จะกลับสถานะการติด/ดับ (ติดเป็นสีเขียวและดับเป็นสีดำ) พร้อมกันนั้นสังเกตที่หน้าจอ Dashboard จะได้ผลลัพธ์ที่ LED0 ตรงกับจอแสดงผลของ WIO Terminal หรือไม่ (อาจต้องรอครู่หนึ่งเนื่องจากอาจมีช่วงหน่วงเวลาจากการทำงานของเซิร์ฟเวอร์ Blynk IoT อยู่บ้าง)

(4) กดวิดเจ็ต **Switch** ตำแหน่ง **LED0** ของ **Dashboard**

กราฟิกรูป LED ที่ตำแหน่ง LED0 จะกลับสถานะอีกครั้ง โดยแสดงด้วยข้อความ ON/OFF

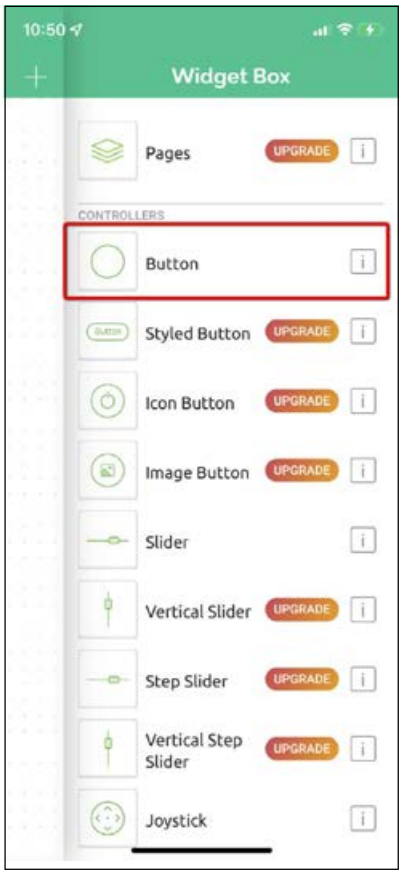
(5) ทำการทดสอบควบคุม LED1 และ LED2 เช่นเดียวกับข้อ 1 และ 2 ที่นำเสนอไปแล้ว

(ข) ทดสอบการทำงานเทียบกันระหว่าง WIO Terminal และแอป Blynk IoT บนสมาร์ตโฟน

(1) ที่สมาร์ตโฟนเปิด Blynk App ทำการล็อกอินเพื่อใช้งานจะปรากฏเทมเพลตที่เคยสร้างไว้ก่อนหน้านี้ จากนั้นแตะปุ่มเครื่องมือเข้าสู่ Developer Mode เพื่อแก้ไข UI ตามรูปที่ 11-74

รูปที่ 11-77 แสดงการลบวิดเจ็ต Gauge ที่มีอยู่ภายในเทมเพลต

รูปที่ 11-78 แสดงวิดเจ็ต Gauge ถูกลบออกจากเทมเพลต



รูปที่ 11-79 ที่หน้าต่าง Widget Box แสดงการเลือกวิดเจ็ต Button

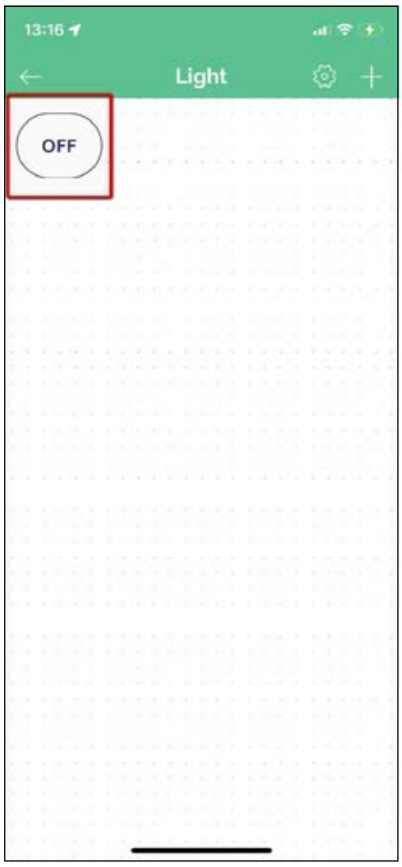
(2) แตะเลือก **Template Light** เพื่อเข้าไปแก้ไข UI ตามรูปที่ 11-75

(3) จากนั้นจะพบวิดเจ็ต **Gauge** ที่มีอยู่เดิม ให้แตะเลือกที่วิดเจ็ต **Gauge** ที่ต้องการลบตามรูปที่ 11-76

(4) จะพบหน้าต่างกำหนดคุณสมบัติของ **Gauge** ให้แตะปุ่ม **Delete** เพื่อลบออกตามรูปที่ 11-77 จากนั้นออกจากหน้าต่างนี้ด้วยการกดปุ่มกากบาทที่มุมบนด้านซ้าย

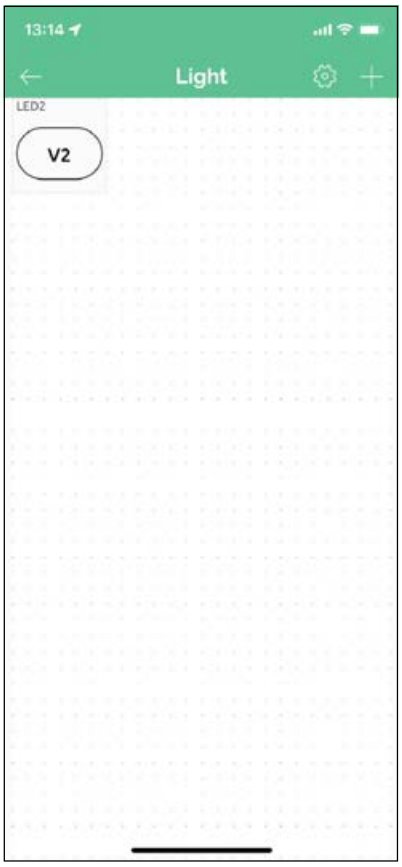
(5) ที่หน้าต่างออกแบบหลัก (Developer Mode) จะเห็นว่า วิดเจ็ต **Gauge** จะถูกลบออกแล้ว แตะที่ปุ่มเครื่องหมายบวกเพื่อเพิ่มวิดเจ็ตตัวใหม่เข้ามา ดังรูปที่ 11-78

(6) จากนั้นจะปรากฏรายการวิดเจ็ต ค้นหาวิดเจ็ต **Button** ดังรูปที่ 11-79 ทำการแตะเลือกมาวางที่หน้าของเทมเพลต



รูปที่ 11-80 แสดงการวางวิดเจ็ต Button เพิ่มเข้าไปใน เทมเพลต

รูปที่ 11-81 แสดงการกำหนดคุณสมบัติของวิดเจ็ต Button



รูปที่ 11-82 แสดงวิดเจ็ต Button ที่กำหนดใช้งานอ้างอิงกับ LED2

(7) จากนั้นจะได้วิดเจ็ต **Button** มาวางบนเทมเพลตตามรูปที่ 11-80

(8) แตะที่วิดเจ็ต **Button** เพื่อเข้าไปกำหนดคุณสมบัติ จะปรากฏหน้าต่างสำหรับตั้งค่าตามรูปที่ 11-81 ทำการกำหนดคุณสมบัติดังนี้

- ตั้งชื่อ **Button** ตัวแรกเป็น ***LED2***
- หัวข้อ **DATASTREAM** กำหนดเป็น ***V2***
- หัวข้อ **MODE** กำหนดเป็น ***SWITCH***

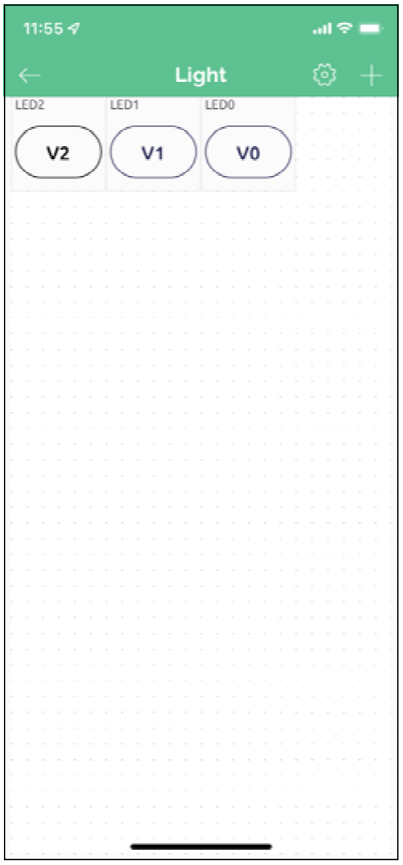
แตะปุ่มกากบาทที่มุมบนด้านซ้าย เพื่อออกจากหน้าต่างตั้งค่า ได้ผลลัพธ์ตามรูปที่ 11-82

(9) ทำการเพิ่มวิดเจ็ต **Button** อีก 2 ตัวตามขั้นตอนก่อนหน้านี้

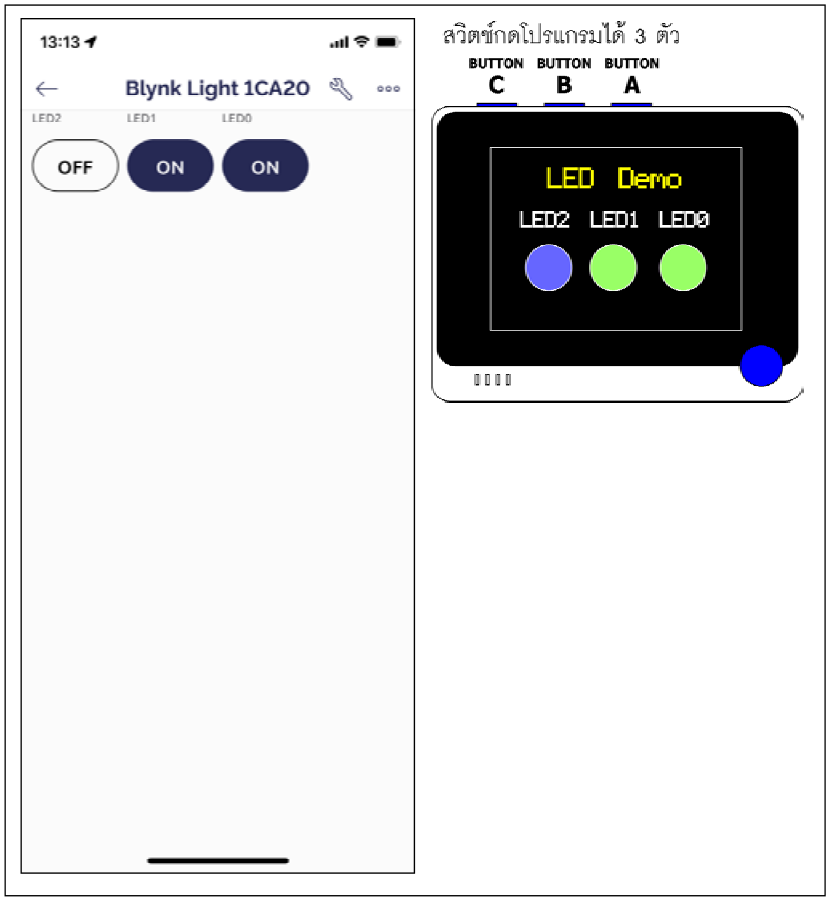
(9.1) ตั้งชื่อเป็น **LED1** ตั้งค่าใช้งานเป็น **V1**

(9.2) ตั้งชื่อเป็น **LED0** ตั้งค่าใช้งานเป็น **V0** ตามรูปที่ 11-83

จากนั้นกลับไปยังหน้าหลักด้วยการแตะปุ่มถอยหลังที่มุมบนด้านซ้าย



รูปที่ 11-83 แสดงวิดเจ็ต Button ทั้ง 3 ตัวทำหน้าที่แสดงสถานะ LED2 ถึง LED0 ภายในเทมเพลต



รูปที่ 11-84 แสดงหน้าหลักของ Blynk App ที่มีการทำงานสัมพันธ์กับ WIO Terminal

(10) ที่หน้าหลักของ Blynk App จะทำงานที่มีสถานะที่ตรงกับการแสดงผลของ WIO Terminal และ **Dashboard** ดังรูปที่ 11-84

(11) ผู้พัฒนาสามารถทดสอบปิด/เปิด LED แต่ละดวงที่ Blynk App เพื่อทดสอบการเปลี่ยนแปลงสถานะว่าตรงกันหรือไม่ เช่นเดียวกับการทดสอบกับ **Dashboard** บนเว็บบราวเซอร์ก่อนหน้านี้

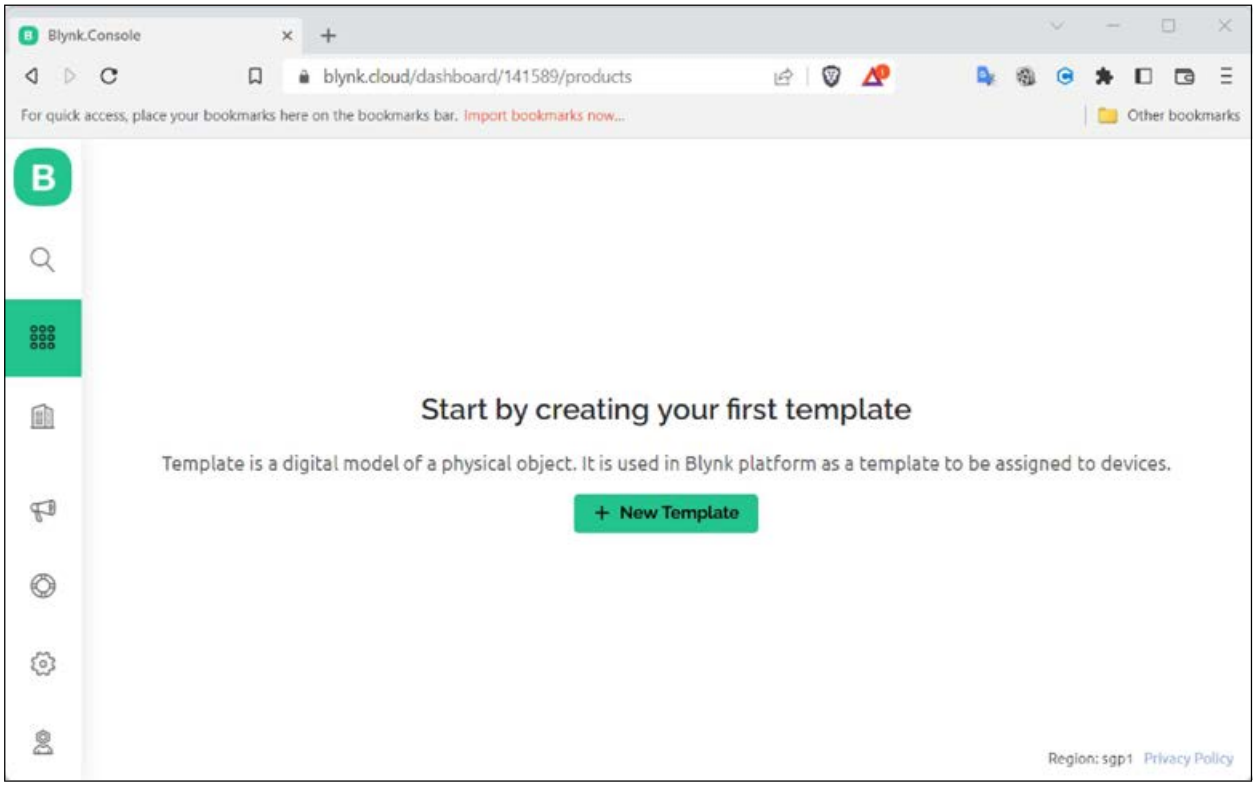
11.6 การใช้งาน Blynk IoT แบบระบุ ชื่อ Wi-Fi และ Token

ในการอธิบายตัวอย่างก่อนหน้านี้ทั้งหมด การเลือกเครือข่าย WiFi ให้ WIO Terminal เชื่อมต่อเพื่อเข้าสู่เครือข่ายอินเทอร์เน็ตจะกระทำผ่าน Blynk IoT App ทั้งบนสมาร์ตโฟนหรือผ่านทางเว็บไซต์ ในหัวข้อนี้เป็นการนำเสนออีกวิธีหนึ่งที่ทำการเชื่อมต่อเครือข่าย WiFi โดยการระบุลงในโค้ดที่อัปโหลดไปยัง WIO Terminal มีขั้นตอนดังนี้

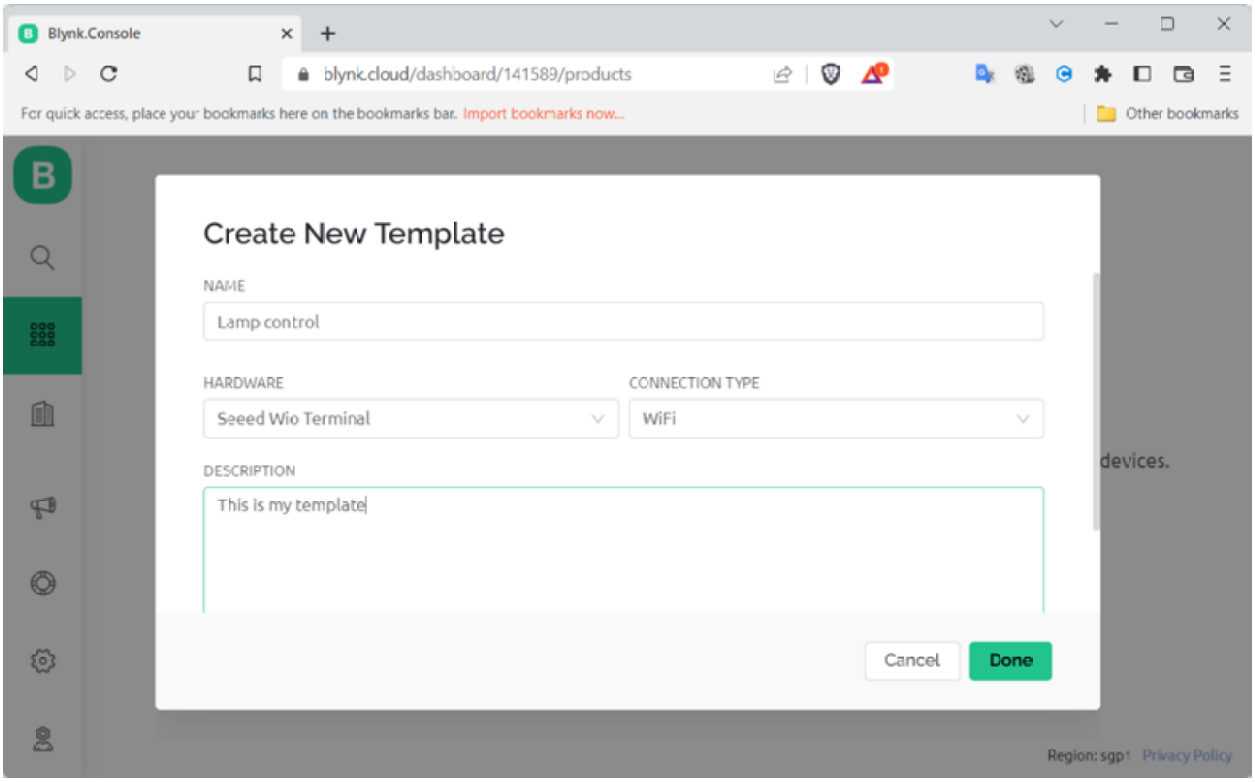
(1) เชื่อมต่อคอมพิวเตอร์ที่ใช้พัฒนาโปรแกรมเข้ากับเครือข่ายอินเทอร์เน็ต เปิดเว็บบราวเซอร์ด เพื่อไปยังเว็บไซต์ของ Blynk IoT ทำการล็อกอิน สร้างเทมเพลตใหม่ โดยการคลิกที่ **New Template** ดังรูปที่ 11-85

(2) ตั้งชื่อเทมเพลตเป็น **Lamp control** เลือกอุปกรณ์เป็น **Seeed Wio Terminal** เลือกเชื่อมต่อผ่าน **Wi-Fi** จากนั้นคลิกที่ปุ่ม **Done** ยืนยันการสร้างเทมเพลต ดังรูปที่ 11-86

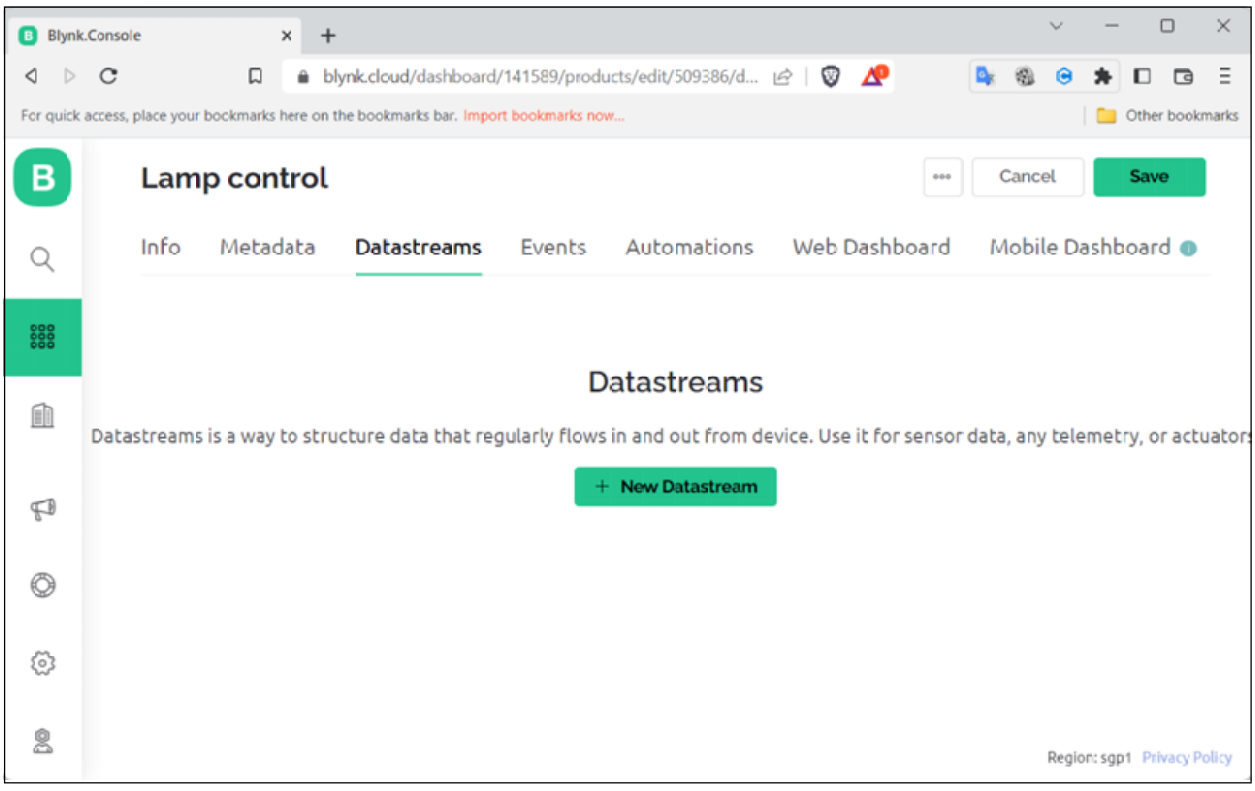
(3) สร้างช่องทางการรับสั่งข้อมูล โดยเลือกไปที่เมนู **Datastreams** เพิ่มช่องทางการรับสั่งข้อมูลด้วยการคลิกที่ปุ่ม **New Datastream** ดังรูปที่ 11-87



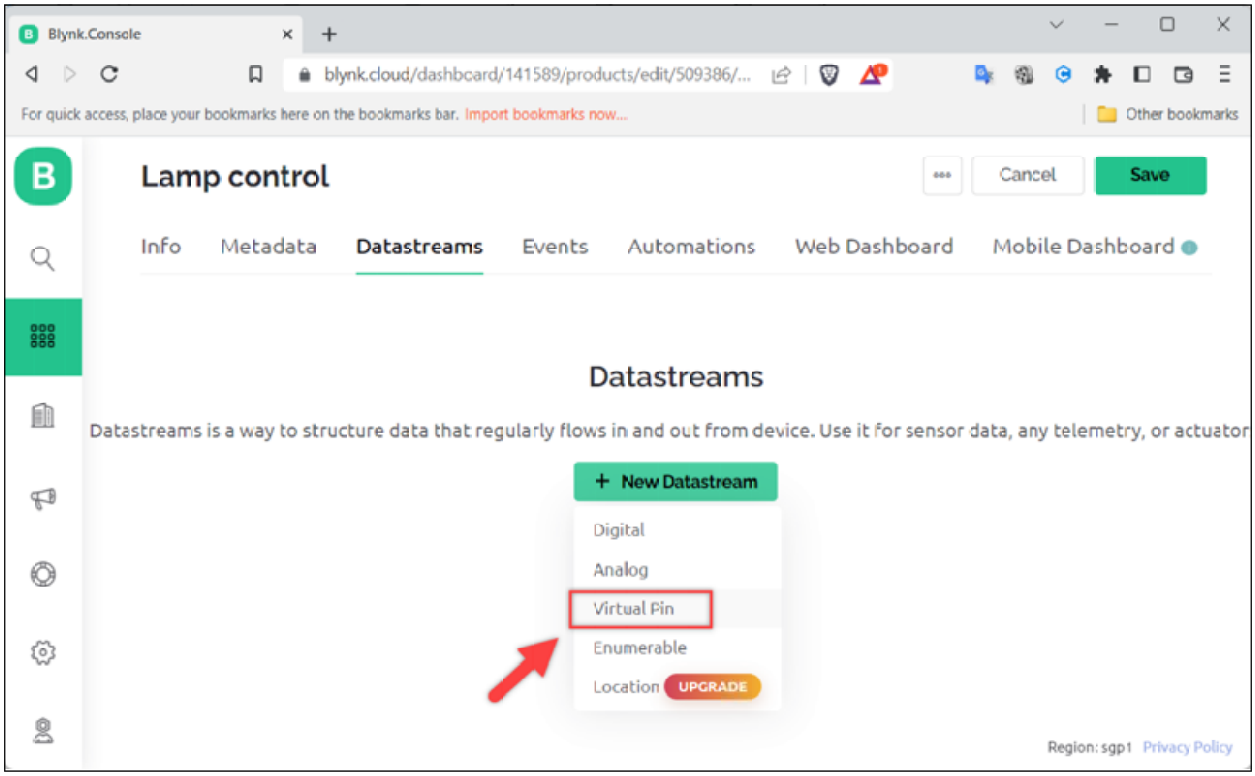
รูปที่ 11-85 เริ่มต้นสร้างเทมเพลตใหม่ของ Blynk IoT



รูปที่ 11-86 สร้างเทมเพลตใหม่ โดยเลือกฮาร์ดแวร์เป็น Seeed Wio Terminal



รูปที่ 11-87 สร้างช่องทางรับส่งข้อมูลใหม่สำหรับเทมเพลตชื่อ Lamp control



รูปที่ 11-88 เลือกช่องทางรับส่งข้อมูลผ่านขาพอร์ตเสมือนหรือ Virtual Pin

(4) เลือกกำหนดช่องทางรับส่งข้อมูลผ่าน **Virtual Pin** หรือขาพอร์ตอินพุตเอาต์พุตเสมือน โดยคลิกที่ + **New Datastream** แล้วเลือก **Virtual Pin** ดังรูปที่ 11-88

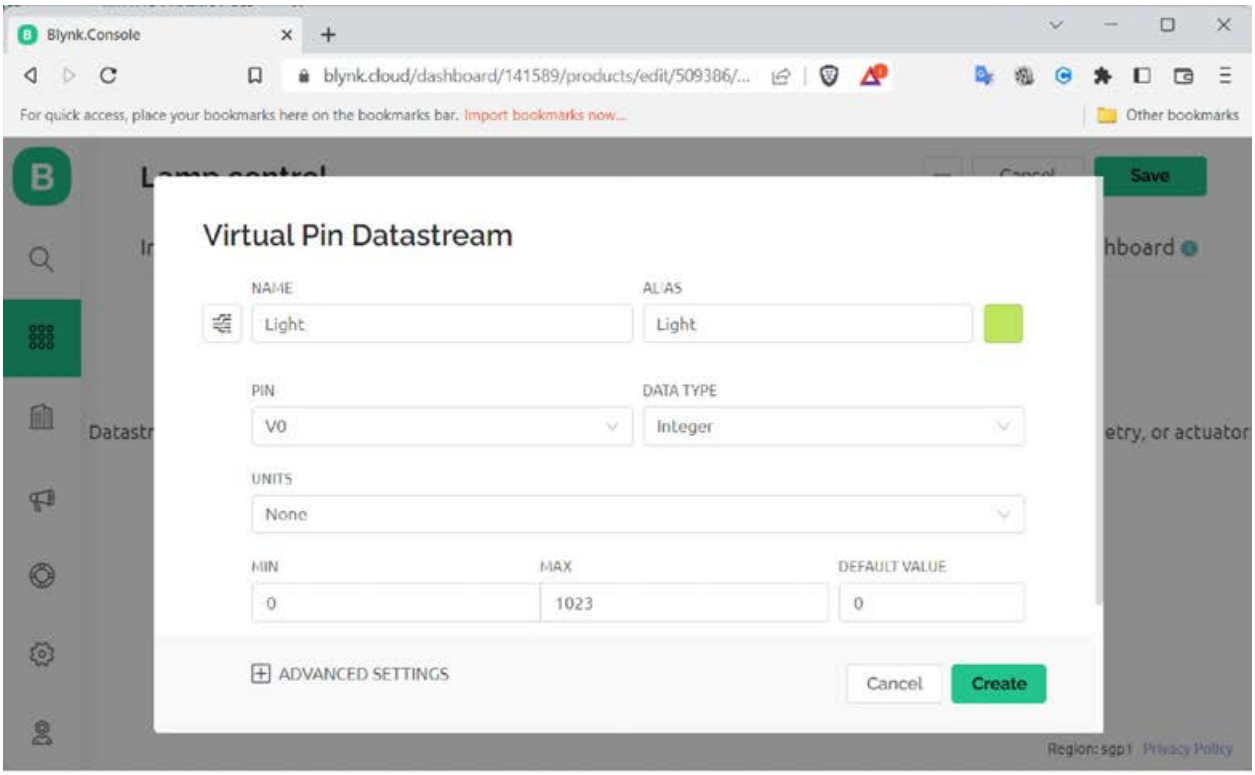
(5) ตั้งค่าการทำงานของช่องทางรับส่งข้อมูลเพื่อใช้ในการรับค่าจากตัวตรวจจับแสงของ WIO Terminal ดังรูปที่ 11-89 จากนั้นคลิกปุ่ม **Create** เพื่อยืนยัน

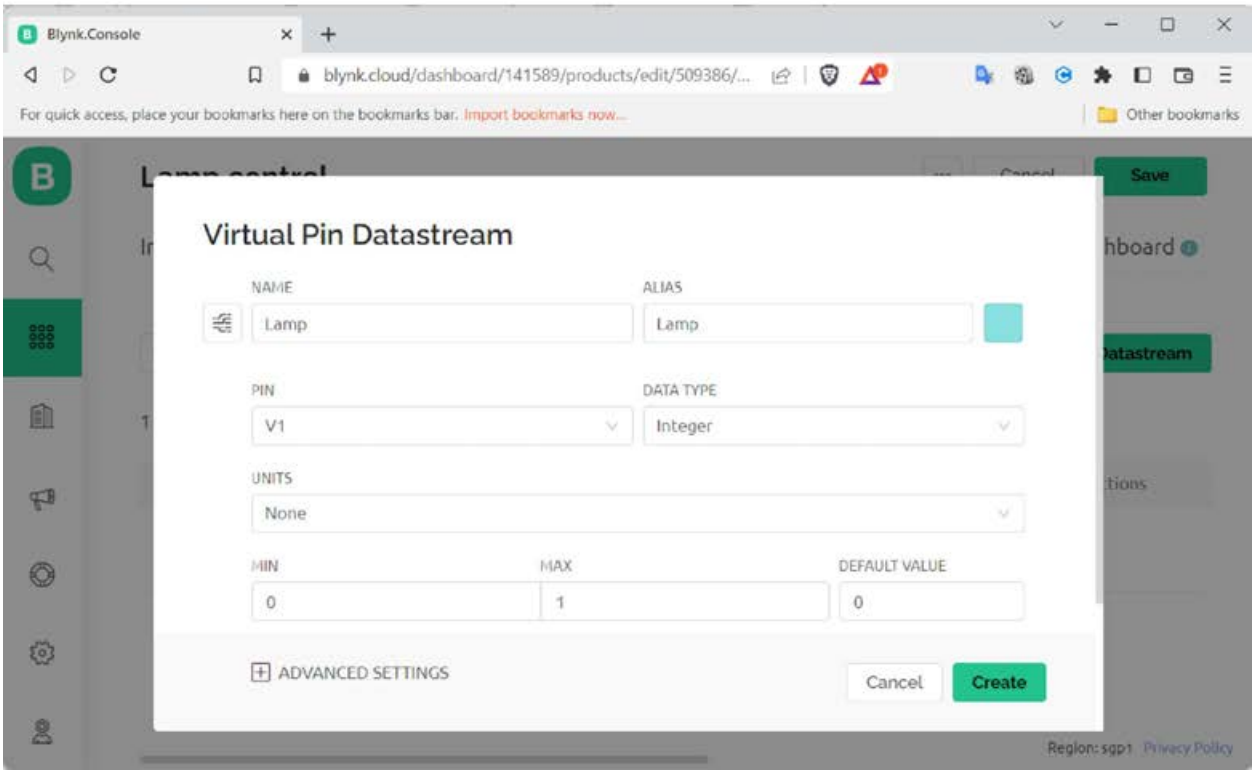
- **NAME** กำหนดเป็น ***Light***
- **PIN** กำหนดเป็น ***V0***
- **DATA TYPE** กำหนดเป็น ***Integer***
- **MAX** ตั้งค่าเป็น ***1023***

(6) สร้างช่องทางรับสั่งข้อมูลสำหรับรองรับค่าสถานะหลอดไฟหรือสวิตช์ ตั้งค่ารายละเอียดดังนี้

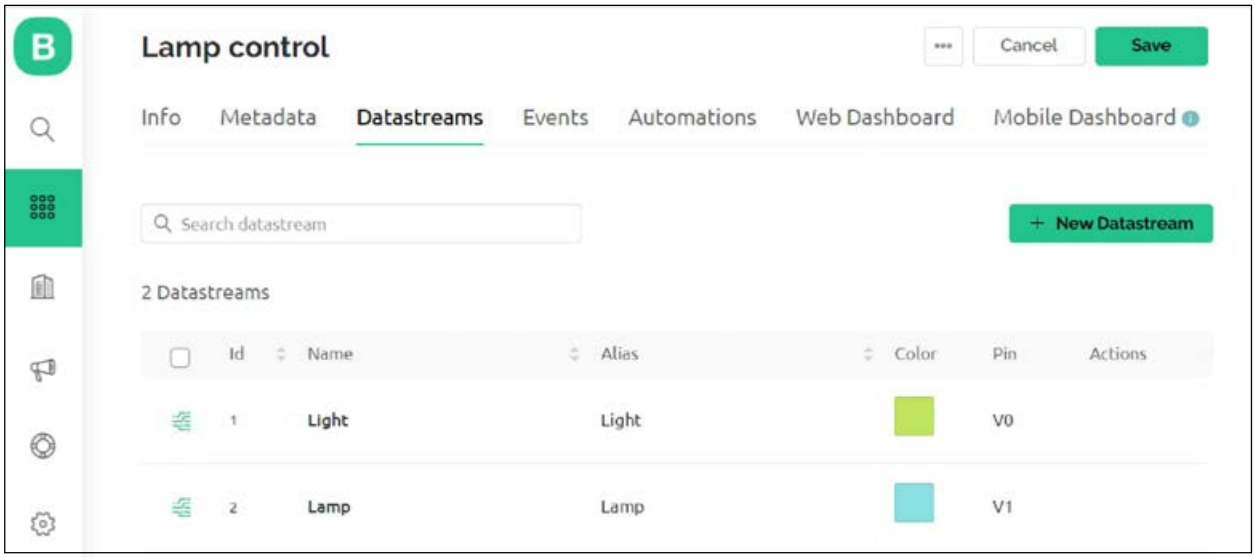
- **NAME** กำหนดเป็น ***Lamp***
- **PIN** กำหนดเป็น ***V1***
- **DATA TYPE** กำหนดเป็น ***Integer***
- **MAX** ตั้งค่าเป็น ***1***

เมื่อตั้งค่าเสร็จแล้ว คลิกปุ่ม **Create** เพื่อยืนยัน ดังรูปที่ 11-90





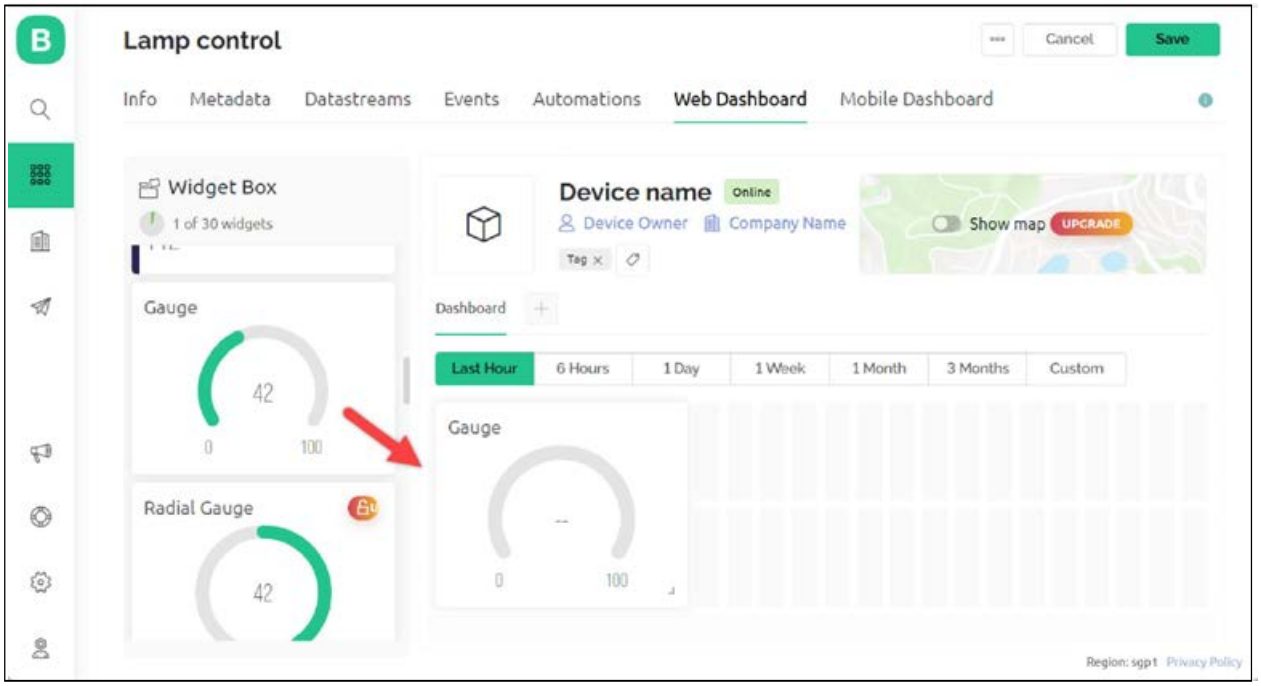
รูปที่ 11-90 หน้าต่างตั้งค่าช่องทางรับส่งข้อมูล ในที่นี้ใช้งานขาพอร์ตเสมือนหมายเลข 1 (V1) เพื่อขับอุปกรณ์แสดงผล (Lamp) ของ WIO Terminal โดยมีข้อมูลเป็นเลขจำนวนเต็ม (Integer) 2 ตัวคือ 0 และ 1



รูปที่ 11-91 หน้าเว็บแสดงเทมเพลต Lam control ที่มีช่องทางรับส่งข้อมูลหรือ Datastreams 2 รายการคือ Light และ Lamp

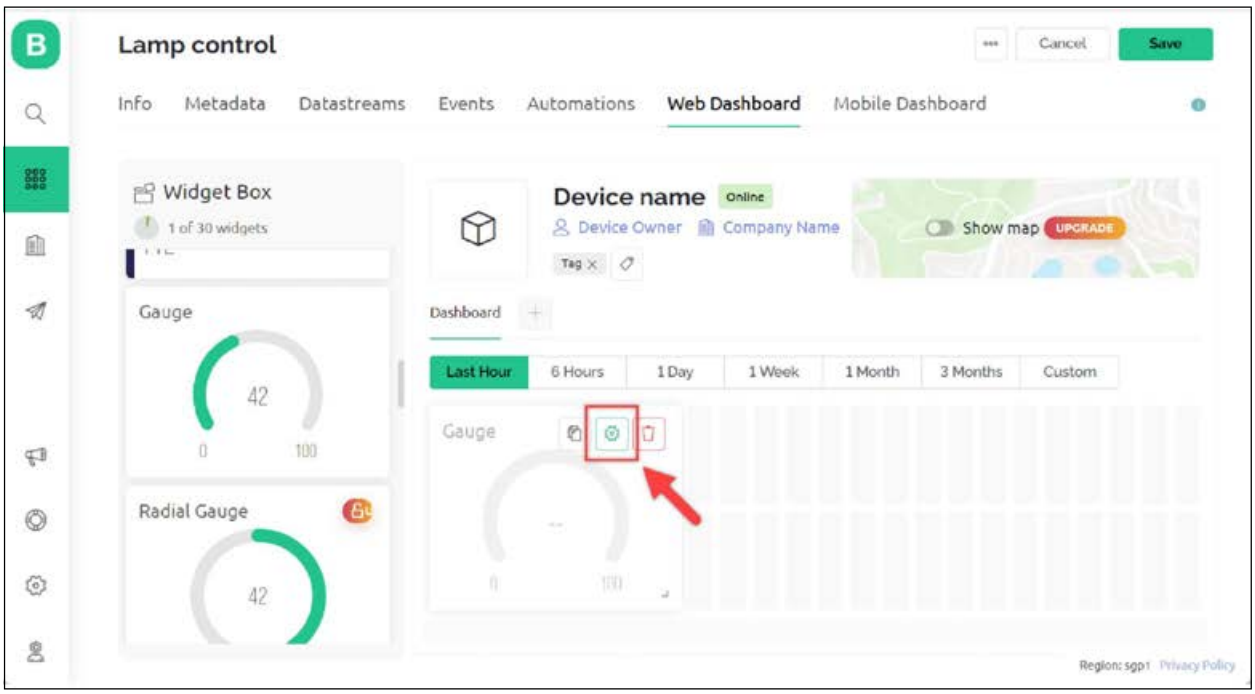
(7) เมื่อสร้างเรียบร้อย จะมีช่องทางรับส่งข้อมูล (Datastreams) 2 รายการคือ **Light** และ **Lamp** ดังรูปที่ 11-91

(8) สร้างแผงควบคุมและแสดงผลหรือแดชบอร์ด โดยไปที่ Web Dashboard จากนั้นเลือกวิดเจ็ตเป็น **Gauge** สำหรับแสดงค่าที่อ่านได้จากตัวตรวจจับแสง โดยการลากมาวางดังรูปที่ 11-92



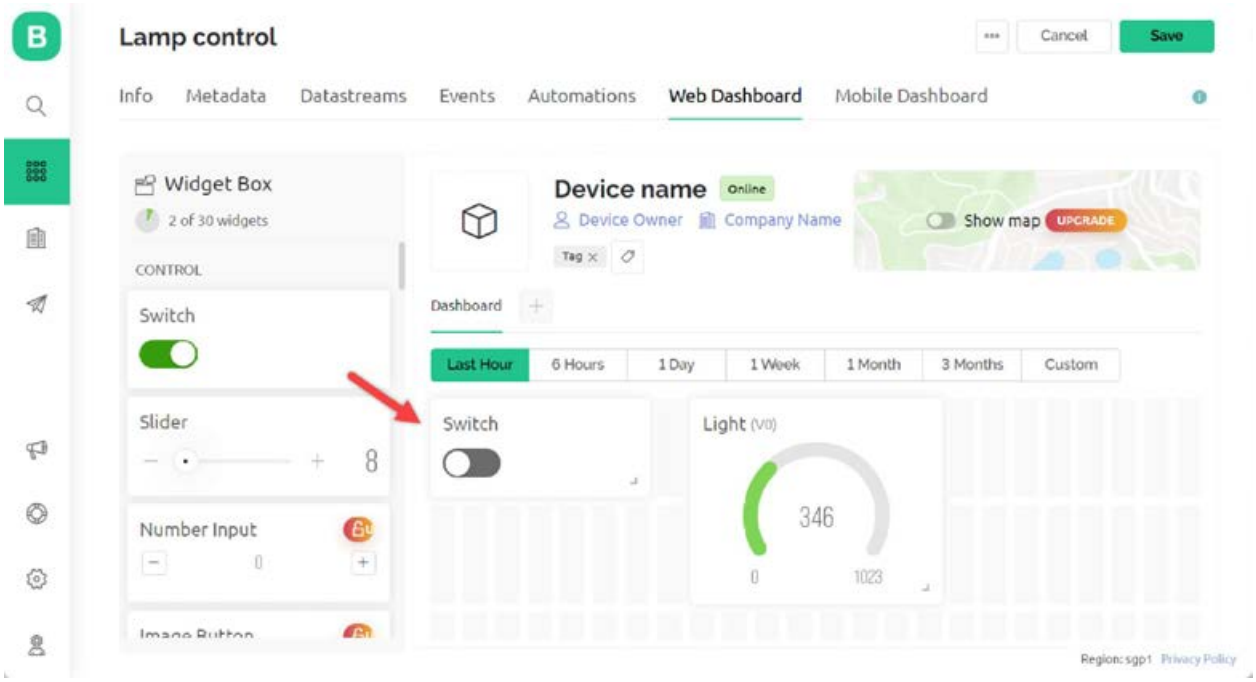
รูปที่ 11-92 เริ่มต้นสร้างแดชบอร์ด โดยลากวิดเจ็ต Gauge มาวางบนแดชบอร์ด

(9) ตั้งค่าวิดเจ็ตเพื่อใช้แสดงผลค่าแสงจากตัวตรวจจับ โดยคลิกที่รูปฟันเฟืองดังรูปที่ 11-93



(10) กำหนดชื่อของวิดเจ็ต **Gauge** เป็น ***Light*** และเลือก **Datastream** เป็น ***Light (V0)*** ดังรูปที่ 11-94 จากนั้นคลิกที่ปุ่ม **Save** เพื่อบันทึกการตั้งค่า

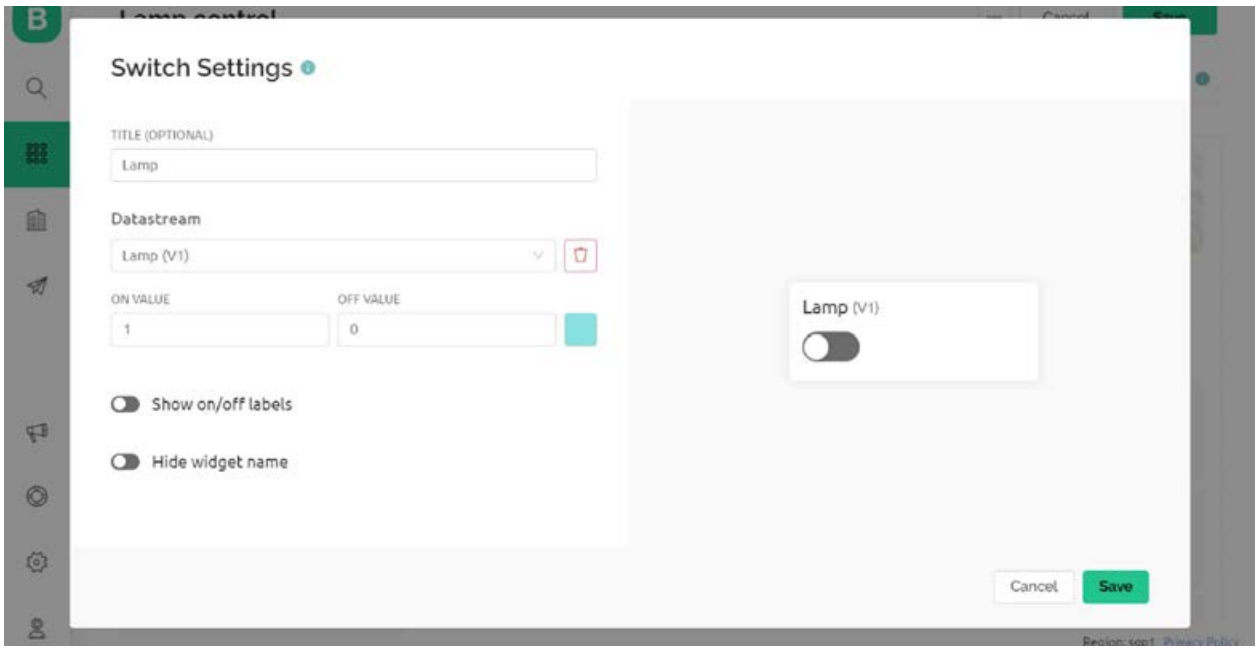
รูปที่ 11-94 กำหนดชื่อและช่องทางรับส่งข้อมูลของวิดเจ็ต Gauge



รูปที่ 11-95 เพิ่มวิดเจ็ต Switch บนพื้นที่สร้างแดชบอร์ด

(11) สร้างส่วนควบคุมการแสดงสถานะและเปิด-ปิดหลอดไฟ โดยเลือกวิดเจ็ต **Switch** เพื่อกำหนดการแสดงสถานะของหลอดไฟ โดยลากมาวางบนพื้นที่สร้างแดชบอร์ดดังรูปที่ 11-95

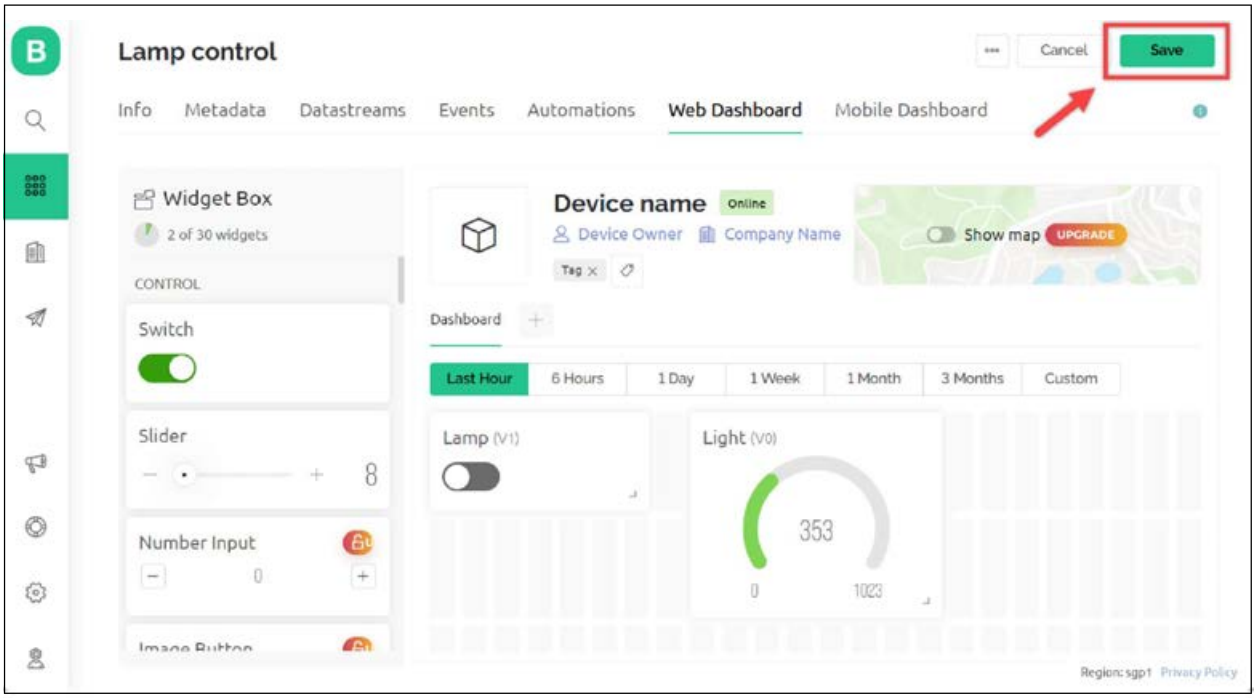
(12) ตั้งชื่อวิดเจ็ต **Switch** เป็น ***Lamp*** และกำหนด **Datastream** เป็น ***Lamp (V1)*** ดังรูปที่ 11-96 จากนั้นคลิกปุ่ม **Save** เพื่อบันทึกการตั้งค่า



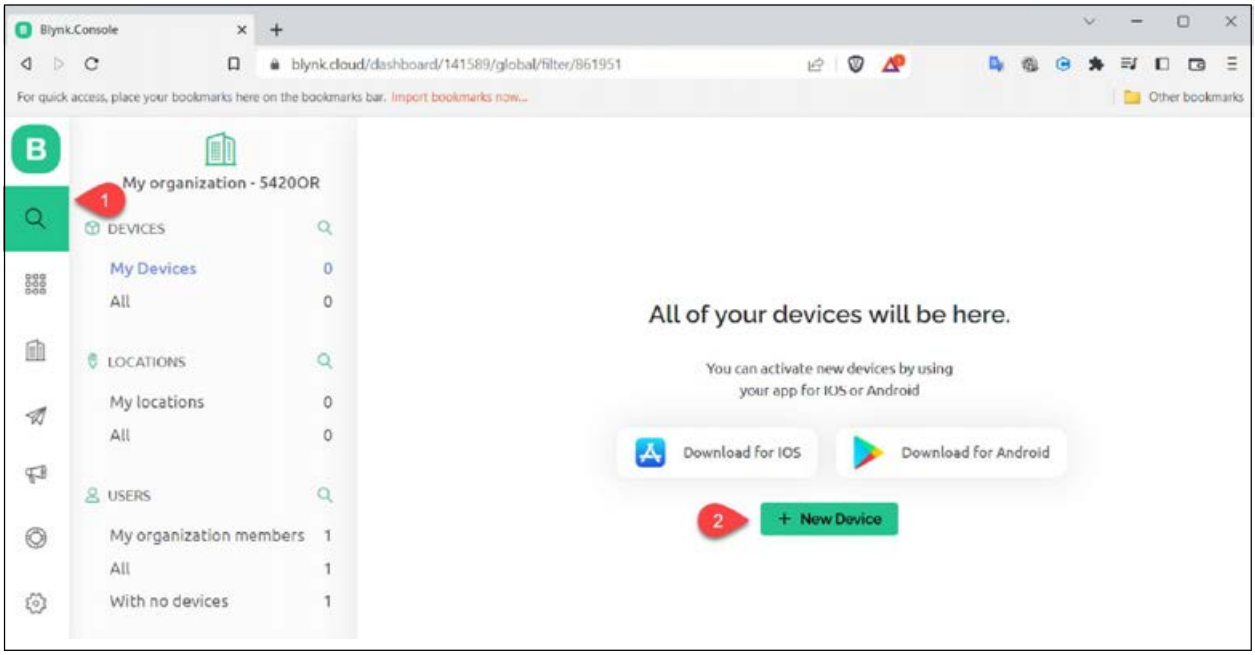
รูปที่ 11-96 กำหนดชื่อและช่องทางรับส่งข้อมูลของวิดเจ็ต Switch

(13) เมื่อเพิ่มวิดเจ็ตที่ต้องการครบแล้ว คลิกที่ปุ่ม **Save** ดังรูปที่ 11-97 เพื่อบันทึกทั้งหมด เป็นอันเสร็จขั้นตอนการสร้างเทมเพลต

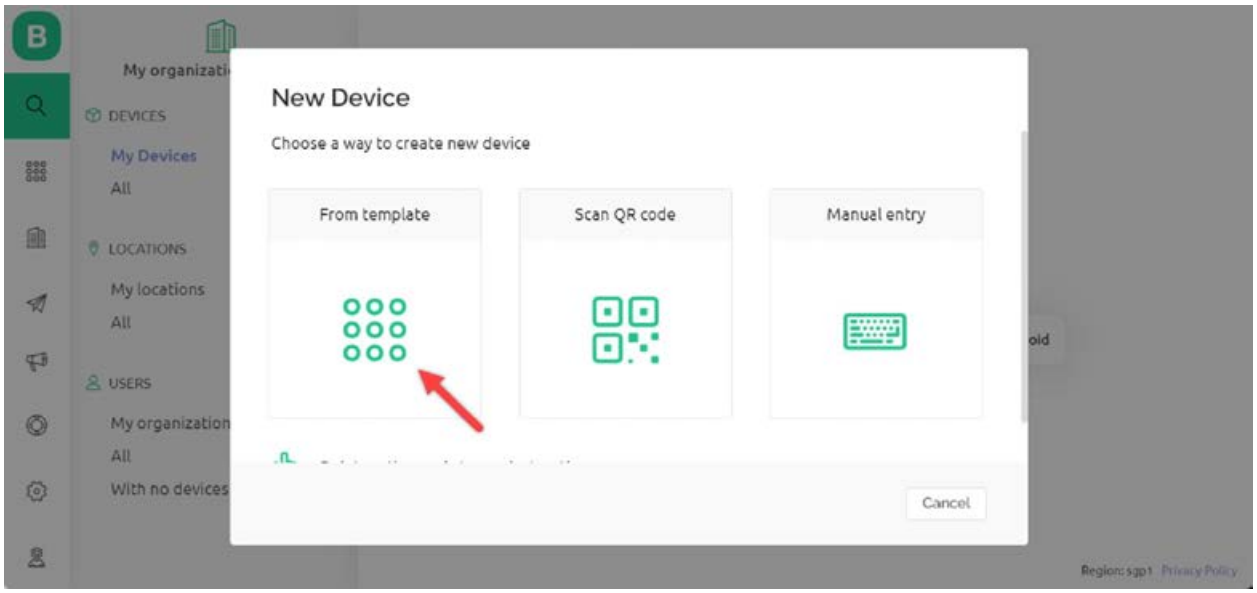
(14) ขั้นตอนต่อไปคือ ทำการเพิ่มอุปกรณ์ โดยไปที่รายการค้นหาดังรูปที่ 11-98 จากนั้นคลิกที่ปุ่ม **New Device**



รูปที่ 11-97 การบันทึกเทมเพลตของแดชบอร์ด Lamp control



รูปที่ 11-98 คลิกรายการค้นหาเพื่อเตรียมเพิ่มอุปกรณ์



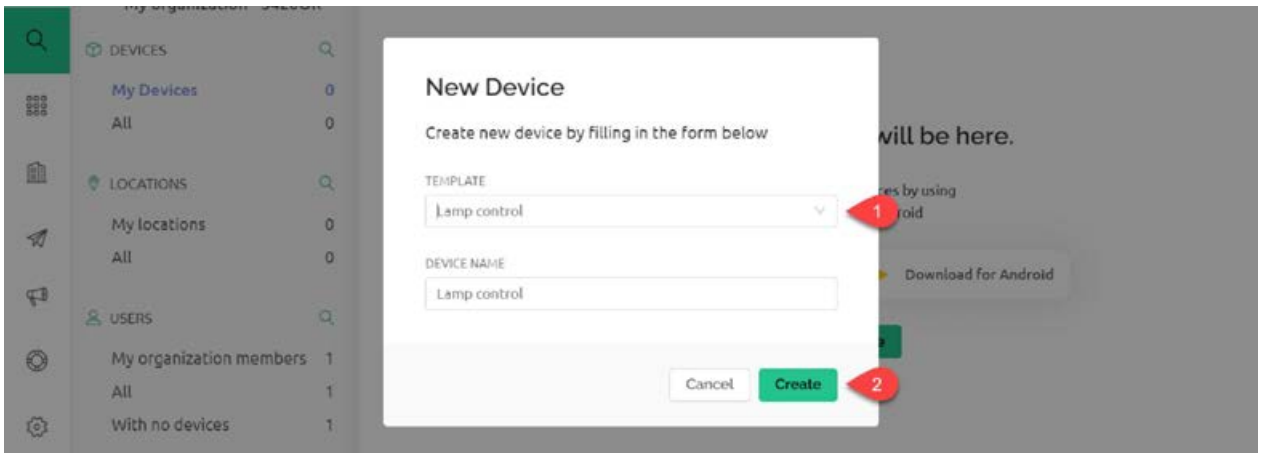
รูปที่ 11-99 เลือกเพิ่มอุปกรณ์จากเทมเพลต

(15) เลือกเพิ่มจากเทมเพลตที่ได้สร้างไว้ ดังรูปที่ 11-99

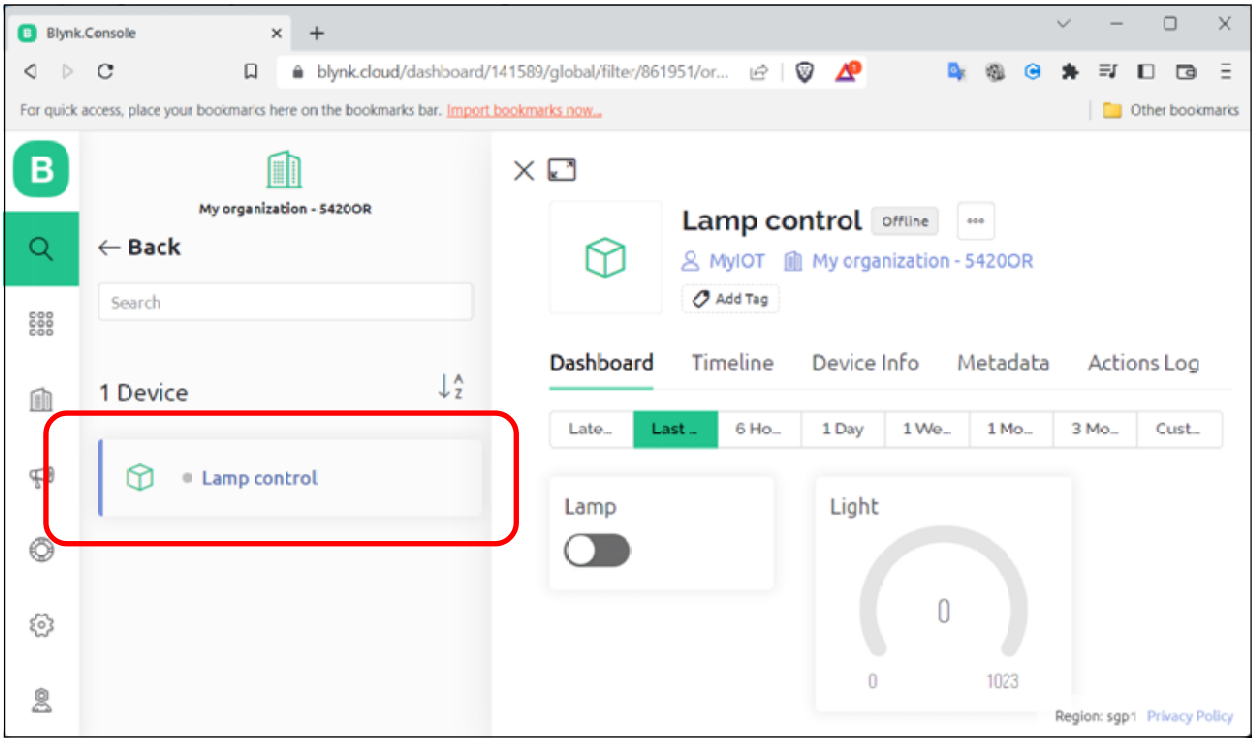
(16) เลือกเทมเพลตที่สร้างไว้ก่อนหน้านี้ นั่นคือ **Lamp control** จากนั้นคลิกปุ่ม **Create** เพื่อยืนยันการเพิ่มอุปกรณ์ ดังรูปที่ 11-100

(17) จากรูปที่ 11-101 จะเห็นว่า มีรายการอุปกรณ์เพิ่มขึ้นมา และอยู่ในสถานะ offline หรือปิด เพื่อรอการเชื่อมต่อ

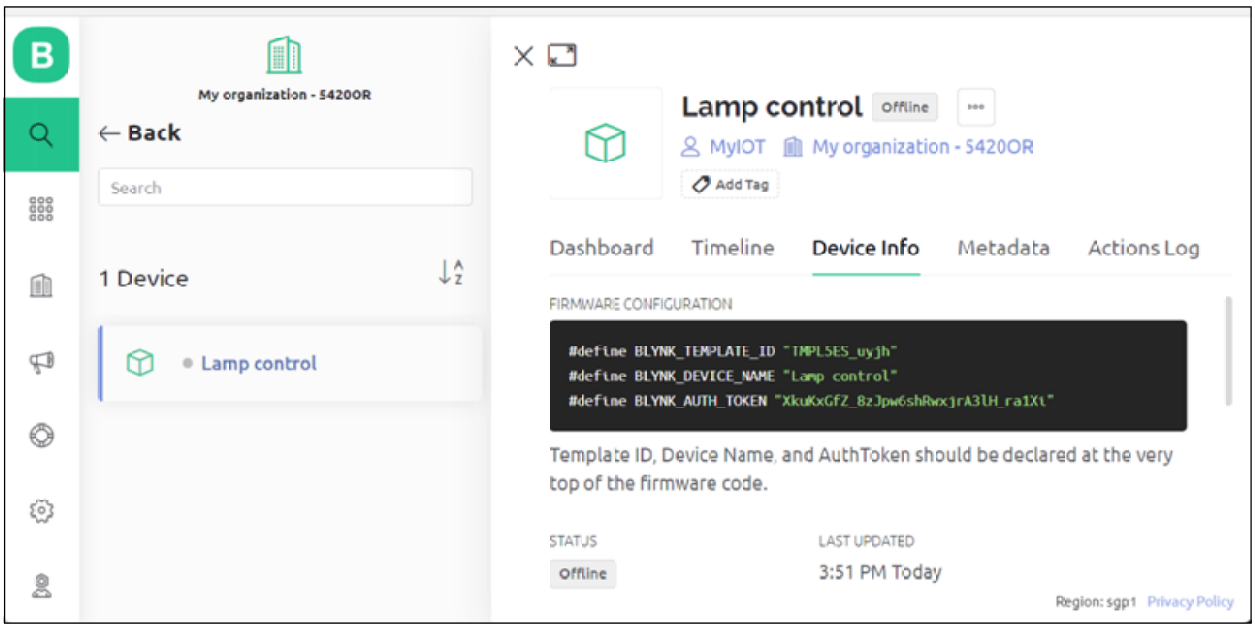
(18) เลือกไปที่แท็บ Device Info เพื่อคัดลอกรหัสโทเคน (token) สำหรับนำไปใส่ในโค้ดที่เขียนด้วย Arduino IDE ต่อไป ดังรูปที่ 11-102



รูปที่ 11-100 ขั้นตอนการเพิ่มอุปกรณ์



รูปที่ 11-101 แสดงการเพิ่มของอุปกรณ์ลงในเทมเพลต



รูปที่ 11-102 แสดงรหัส token ที่ทาง Blynk IoT App สร้างขึ้นมาให้

```
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID "TMPL5ES_uyjh"     // คัดลอกจาก Device Info
#define BLYNK_DEVICE_NAME "Lamp control"     // คัดลอกจาก Device Info
#define BLYNK_AUTH_TOKEN "XkuKxGfZ_8zJpW6shRWxjrA3lH_ra1Xi"     // คัดลอกจาก Device Info

char ssid[] = "xxxxxxxx";                    // ชื่อของตัวปล่อยสัญญาณ WiFi
char pass[] = "zzzzzzzz";                    // รหัสผ่านของตัวปล่อยสัญญาณ WiFi

#include <rpcWiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleWioTerminal.h>          // ไลบรารีติดต่อกับ Blynk สำหรับ WIO Terminal
#include <TFT_eSPI.h>
TFT_eSPI tft;
BlynkTimer     timer_Sersor;
BlynkTimer     timer_Button;
int statusLed = 0;

BLYNK_WRITE(V1)
{
   int st = param.asInt();
   Serial.print("Got a value: ");
   Serial.println(st);
   if (st == 0)
   {
        statusLed = 0;
   }
   else
   {
        statusLed = 1;
   }
}
void scanButton ()
{
   if (digitalRead(WIO_5S_PRESS) == LOW)          // ตรวจสอบการกดปุ่มของสวิตช์เข้ารหัส (สีน้ำเงิน)
   {
        statusLed = ! statusLed;                  // กลับสถานะตัวแปร
   }
   if (statusLed == 1)                            // ตรวจสอบสถานะของกราฟิก LED
   {
        tft.fillCircle(160, 140, 30, TFT_GREEN);  // แสดงกราฟิก LED ติดสว่าง
   }
```

โปรแกรมที่ 11-3 ไฟล์ Blynk_Lamp_Control.ino สำหรับกำหนดให้ WIO Terminal วัดความเข้มแสงในสภาพแวดล้อมนั้น แล้วส่งค่าไปแสดงยัง Blynk IoT และรับข้อมูลสั่งงานจาก Blynk IoT App มาควบคุมการติดดับของหลอดไฟจำลอง (มีต่อ)

```
    else
    {
        tft.fillCircle(160, 140, 30, TFT_DARKGREY);     // แสดงกราฟิก LED ดับ
    }
}
void sendLight()
{
    int light = analogRead(WIO_LIGHT);
    String str = "Light: " + String(light);
    tft.drawString(str + String("    "), 80, 60);
    Blynk.virtualWrite(V0, light);                      // ส่งค่าที่อ่านได้จากตัวตรวจจับแสง
    Blynk.virtualWrite(V1, statusLed);                  // ส่งค่าสถานะหลอดไฟจำลอง
}
void setup()
{
    Serial.begin(9600);
    delay(100);
    pinMode(WIO_LIGHT, INPUT);
    pinMode(WIO_5S_PRESS, INPUT_PULLUP);
    tft.begin();
    tft.setRotation(3);
    tft.fillScreen(TFT_BLACK);
    tft.setTextColor(TFT_ORANGE, TFT_BLACK);
    tft.setTextSize(3);
    tft.drawString("Wait...", 80, 120);

    Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);          // เรียกใช้ฟังก์ชัน sendLight ทุกๆ 1 วินาที
    timer_Sersor.setInterval(1000L, sendLight);         // เรียกใช้ฟังก์ชัน scanButton ทุกๆ 0.5 วินาที
    timer_Button.setInterval(500L, scanButton);
    tft.fillScreen(TFT_BLACK);
}
void loop()
{
    Blynk.run();
    timer_Sersor.run();
    timer_Button.run();
}
```

โปรแกรมที่ 11-3 ไฟล์ Blynk_Lamp_Control.ino สำหรับกำหนดให้ WIO Terminal วัดความเข้มแสงในสภาพแวดล้อมนั้น แล้วส่งค่าไปแสดงยัง Blynk IoT และรับข้อมูลสั่งงานจาก Blynk IoT App มาควบคุมการติดดับของหลอดไฟจำลอง (จบ)

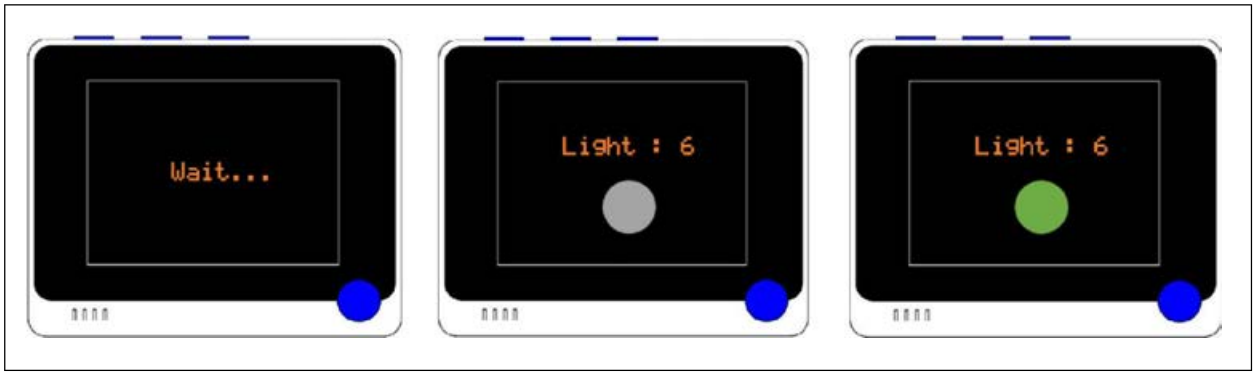
(19) ไปที่โปรแกรม Arduino IDE สร้างโค้ดสำหรับโปรแกรม ***Blynk_Lamp_Control*** ดังในโปรแกรมที่ 11-3

(20) เชื่อมต่อ WIO Terminal เข้ากับพอร์ต USB ของคอมพิวเตอร์ แล้วอัปโหลดโปรแกรม

(21) เมื่ออัปโหลดโปรแกรมเสร็จ บอร์ด WIO Terminal จะทำงานทันที

ผลลัพธ์การทำงาน

WIO Terminal จะอ่านค่าจากตัวตรวจจับแสงและสถานะหลอดไฟจำลองในขณะนั้นมาแสดงที่หน้าจอแสดงผลของ WIO TErminal ดังรูปที่ 11-103 และส่งไปยัง Blynk IoT คลาวด์เซิร์ฟเวอร์ผ่านเครือข่ายอินเทอร์เน็ต ผู้ใช้งานสามารถสั่งเปิด-ปิดหลอดไฟจำลองด้วยการกดปุ่มสวิตช์สีน้ำเงินหรือ Button บน WIO Terminal หรือบนแดชบอร์ดที่หน้าเว็บของ Blynk IoT คลาวด์เซิร์ฟเวอร์ก็ได้ จะได้ผลลัพธ์การทำงานที่ตรงกัน



รูปที่ 11-103 ผลการทำงานของโปรแกรมที่ 11-3 ที่บอร์ด WIO Terminal

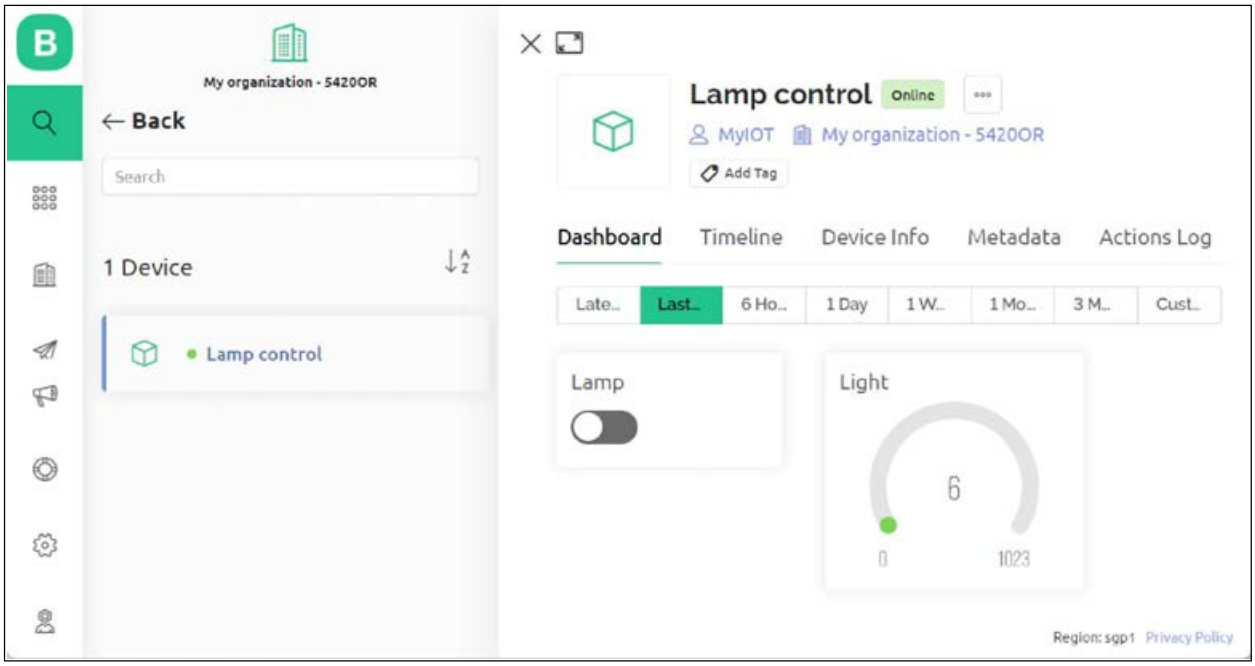
สำหรับการทำงานกับ Blynk IoT ของ WIO Terminal มีดังนี้

(1) รอการเชื่อมต่อ Blynk IoT คลาวด์เซิร์ฟเวอร์

(2) เมื่อเชื่อมต่อได้แล้ว จะแสดงค่าที่อ่านได้จากตัวตรวจจับแสง และสถานะหลอดไฟดับจำลอง เป็นวงกลมสีเทา

(3) เมื่อกดปุ่มสีน้ำเงินบน WIO Terminal จากวงกลมสีเทาที่หมายถึงดับหรือ OFF เปลี่ยนเป็นวงกลมสีเขียวที่หมายถึงทำงาน (ON)

*ที่หน้าแดชบอร์ดบนหน้าเว็บของ **Blynk IoT** จะแสดงสถานะ **Online** และการแสดงผลจะตรงกับหน้าจอแสดงผลของ WIO Terminal ดังรูปที่ 11-104*



รูปที่ 11-104 แดชบอร์ดของ Lamp control เมื่อทำงานและติดต่อกับ WIO Terminal เพื่ออ่านค่าจากตัวตรวจจับแสง และควบคุมการทำงานของหลอดไฟจำลอง