# รายงานวิจัยฉบับสมบูรณ์

# การศึกษาความเป็นไปได้ด้านการออกแบบและพัฒนาเทคโนโลยีคู่แฝดดิจิทัลบนพื้นฐานของเทคโนโลยี PLCnext (ระยะที่ 1)

# Feasibility Study of Design and Development of Digital Twin Technology based on PLCnext Technology (Phase 1)

นายประจักษ์ จิตเงินมะดัน หัวหน้าโครงการวิจัย

โครงการวิจัยประเภทเงินรายได้
คณะวิทยาการสารสนเทศ
ประจำปีงบประมาณ พ.ศ. 2566
มหาวิทยาลัยบูรพา

สัญญาเลขที่ 005/2566

# รายงานวิจัยฉบับสมบูรณ์

การศึกษาความเป็นไปได้ด้านการออกแบบและพัฒนาเทคโนโลยีคู่แฝดดิจิทัลบนพื้นฐาน
ของเทคโนโลยี PLCnext (ระยะที่ 1)

Feasibility Study of Design and Development of Digital Twin Technology
based on PLCnext Technology (Phase 1)

นายประจักษ์ จิตเงินมะดัน หัวหน้าโครงการวิจัย

คณะวิทยาการสารสนเทศ
มิถุนายน 2566

# กิตติกรรมประกาศ

งานวิจัยนี้ได้รับทุนสนับสนุนการวิจัยจากเงินรายได้ คณะวิทยาการสารสนเทศ ประจำปีงบประมาณ พ.ศ. 2566 เลขที่สัญญา 005/2566

คณะผู้วิจัยขอขอบคุณคณะวิทยาการสารสนเทศ และห้องปฏิบัติการวิจัยสื่อดิจิทัลและปฏิสัมพันธ์ (Digital Media and Interaction Laboratory) ที่เอื้อเฟื้องบประมาณในการดำเนินการ และสถานที่ในการวางแผนปฏิบัติ และทดสอบชิ้นงานให้มีความสมบูรณ์ รวมถึงการให้บริการด้านสาธารณูปโภคต่าง ๆ ด้วย

และในท้ายที่สุด คณะผู้วิจัยขอขอบคุณผู้ร่วมทดสอบการใช้งานได้และประสิทธิภาพของ โครงการ Feasibility Study of Design and Development of Digital Twin Technology based on PLCnext Technology (Phase 1) และทีมงานสนับสนุนทุกคนที่ทุ่มเทแรงกายแรงใจในการดำเนินการจนงานวิจัยนี้สำเร็จลุล่วงไปได้ด้วยดี

มิถุนายน 2567

# บทคัดย่อ

เทคโนโลยีดิจิทัลทวิน (Digital Twin Technology: DTT) เป็นแนวคิดสำคัญในยุคอุตสาหกรรม 4.0 ที่ช่วยสร้างแบบจำลองเสมือนจริงของระบบทางกายภาพเพื่อการวิเคราะห์ การจำลอง และการควบคุมระบบได้ในเวลาเดียวกัน การสื่อสารที่มีประสิทธิภาพระหว่างระบบดิจิทัลและระบบทางกายภาพเป็นสิ่งสำคัญสำหรับการโต้ตอบแบบเรียลไทม์ งานวิจัยนี้มุ่งเน้นการศึกษาและประเมินประสิทธิภาพของโปรโตคอลการสื่อสารสามประเภท คือ MQTT, HTTP และ WebSocket ในการตั้งค่าระบบดิจิทัลทวินที่ประกอบด้วย PLCnext เทคโนโลยี ควบคู่กับแพลตฟอร์ม IoT ชื่อ ThingsBoard และเอนจิน 3D Unity โดยการวัดค่าความหน่วงเวลาหรือ latency ในการส่งและเรียกใช้งานข้อมูลระหว่างระบบ การวัดค่าความหน่วงเวลาได้มีการดำเนินการในกระบวนการส่งข้อมูลไปยังแพลตฟอร์ม IoT และการเรียกใช้งานข้อมูลจากแพลตฟอร์ม IoT ไปยัง Unity ผลการทดลองแสดงให้เห็นว่าโปรโตคอล MQTT (126.667 ms) มีความหน่วงเวลาต่ำกว่าโปรโตคอล HTTP (341.997 ms) ในการส่งข้อมูลไปยังแพลตฟอร์ม ThingsBoard เนื่องจากการออกแบบที่เรียบง่ายและเบากว่า นอกจากนี้โปรโตคอล WebSocket (259.322 ms) ยังแสดงความได้เปรียบเล็กน้อยเหนือ HTTP ในการเรียกใช้งานข้อมูลจากแพลตฟอร์ม ThingsBoard ไปยัง Unity ซึ่งสำคัญสำหรับการโต้ตอบแบบเรียลไทม์ในระบบคู่แฝดดิจิทัล ผลลัพธ์จากการศึกษานี้ เป็นข้อมูลสำคัญในการเลือกใช้โปรโตคอลการสื่อสารที่เหมาะสมในระบบคู่แฝดดิจิทัลสำหรับอุตสาหกรรม 4.0 เพื่อปรับปรุงการโต้ตอบแบบเรียลไทม์ในอุตสาหกรรมระบบอัตโนมัติ การวิจัยนี้เป็นขั้นตอนสำคัญในการพัฒนาและปรับปรุงการใช้งานเทคโนโลยีคู่แฝดดิจิทัลในอนาคตเพื่อสร้างระบบอัตโนมัติที่มีประสิทธิภาพและยั่งยืนมากยิ่งขึ้น

# Abstract

Digital Twin Technology (DTT) is a crucial concept in Industry 4.0, enabling the creation of virtual replicas of physical systems for simultaneous analysis, simulation, and control. Effective communication between digital and physical systems is essential for real-time interactions. This research focuses on studying and evaluating the performance of three communication protocols: MQTT, HTTP, and WebSocket, in a Digital Twin setup comprising PLCnext technology, an IoT platform called ThingsBoard, and the Unity 3D engine. Latency measurements were taken during data uploads to the IoT platform and data retrieval from the IoT platform to Unity. The results indicate that the MQTT protocol (126.667 ms) has lower latency compared to the HTTP protocol (341.977 ms) when sending data to the ThingsBoard platform, owing to its simpler and lighter design. Additionally, the WebSocket protocol (259.322 ms) shows a slight advantage over HTTP in retrieving data from the ThingsBoard platform to Unity, which is crucial for real-time interactions in a Digital Twin setup. The findings from this study provide critical information for selecting appropriate communication protocols in Digital Twin systems for Industry 4.0, aiming to enhance real-time interactions in the automation industry. This research marks an important step in the development and improvement of Digital Twin technology applications, fostering more efficient and sustainable automation systems in the future.

# สารบัญ

# สารบัญตาราง

# สารบัญภาพ

# 1 บทนำ

## 1.1 ที่มาและความสำคัญ

การผลิตในโรงงานอุตสาหกรรมในปัจจุบันมีความก้าวหน้าไปมาก มีการใช้เทคโนโลยีการผลิตที่เป็นการควบคุมเครื่องจักรการผลิตควบคู่ไปกับการใช้เทคโนโลยีสารสนเทศและดิจิทัลมากขึ้น ก่อให้เกิดการทำงานแบบอัตโนมัติในการผลิต ซึ่งการผลิตในปัจจุบันนั้นเป็นการเข้าสู่ยุคที่เรียกว่า อุตสาหกรรม 4.0 ที่มีการใช้เทคโนโลยีดิจิทัลในการควบคุมการผลิตแบบอัตโนมัติ โดยในการควบคุมการผลิตนั้นมีการใช้เครื่องควบคุมประเภท PLC (Programmable Logic Control) เข้ามาใช้งานอย่างยาวนานและถือว่าเป็นเทคโนโลยีที่ได้รับการยอมรับกันเป็นมาตรฐานสากล และในอนาคตนั้นเมื่อความก้าวหน้าด้านเทคโนโลยีดิจิทัลได้พัฒนาไปมาก จึงเกิดการใช้งานวัตถุเสมือนจริงเพื่อทำการจำลองการทำงานของเครื่องจักรหรือลักษณะทางกายภาพของเครื่องจักรและการทำงานของเครื่องจักร โดยเทคโนโลยีที่ถูกพัฒนาขึ้นนี้เรียกว่า Digital Twin หรือ คู่แฝดดิจิทัล (IBM, 2023)

อย่างไรก็ตาม เทคโนโลยีและการประยุกต์ใช้งาน Digital Twin นั้นยังอยู่ในยุคของการเริ่มต้นใช้งานเท่านั้น ดังนั้นจึงต้องมีการศึกษาความเป็นไปได้ การใช้ประโยชน์ และผลกระทบที่อาจจะเกิดขึ้นจากการใช้งานเทคโนโลยีนี้อย่างละเอียด การศึกษาในครั้งนี้จะเน้นไปที่การประยุกต์ใช้เทคโนโลยี Digital Twin ในการผลิตสินค้าแบบอัตโนมัติในอุตสาหกรรม 4.0 บนพื้นฐานของเทคโนโลยี PLCnext (PLCnext Community, 2023) ซึ่งเป็นเทคโนโลยีที่ได้รับการพัฒนาให้มีประสิทธิภาพมากกว่า PLC ทั่วไป และสามารถรองรับการสื่อสารด้วยโปรโตคอลที่เป็นที่นิยมในปัจจุบัน เช่น OPC UA, MQTT, REST API เป็นต้น ซึ่งสามารถเชื่อมต่อกับอุปกรณ์อินเทอร์เน็ตของสรรพสิ่ง (Internet of Things: IoT) และการเก็บข้อมูลหรือเรียกข้อมูลจากระบบคลาวด์ได้อีกด้วย

ในการใช้งาน Digital Twin นั้น การสื่อสารระหว่างระบบดิจิทัลและระบบทางกายภาพเป็นสิ่งสำคัญอย่างยิ่ง เนื่องจากการเชื่อมต่อและการส่งผ่านข้อมูลที่มีประสิทธิภาพจะทำให้การโต้ตอบแบบเรียลไทม์เป็นไปอย่างราบรื่น จากงานวิจัยของ Fuller et al. (Fuller, Fan, Day, & Barlow, 2020) พบว่ามีการใช้ Digital Twin ในรูปแบบของการเชื่อมต่อของกายภาพ (คือ เครื่องจักร) และส่วนของรูปแบบเสมือนแบบไร้รอยต่อ ซึ่งส่งผลดีต่อการส่งผ่านข้อมูล การปรับปรุงสถานะแบบเรียลไทม์ หรือการควบคุมโดยใช้เทคโนโลยีที่เกี่ยวข้อง เช่น IoT หรือ AI (Artificial Intelligence) เป็นต้น นอกจากนั้น Liu et al. (Liu, Fang, Dong, & Xu, 2021) ยังได้สำรวจลึกลงไปถึงเครื่องมือที่ใช้ในการสร้างเทคโนโลยี Digital Twin ทั้งในด้านฮาร์ดแวร์และซอฟต์แวร์ และผลกระทบที่อาจจะเกิดขึ้นได้ รวมถึงการสำรวจทิศทางของเทคโนโลยีนี้อีกด้วย

ในมุมมองทางเศรษฐกิจนั้น Digital Twin มีผลอย่างมากต่อการเปลี่ยนแปลงวิธีการผลิตหรือวิธีการจำลองข้อมูลก่อนการผลิต (Simulation) โดยข้อมูลจาก World Economic Forum ระบุว่า Digital Twin จะเข้ามามีบทบาทในการผลิตมากในช่วงทศวรรษนี้ โดยที่อุตสาหกรรมยานยนต์และการบินจะได้รับการเปลี่ยนแปลงมากที่สุด นอกจากนั้น Digital Twin ยังจะช่วยในการวางแผนเมืองอัจฉริยะในอนาคต การใช้ชีวิตอย่างยั่งยืนด้วยการวางแผนลดคาร์บอน เป็นต้น (World Economic Forum, 2023) ในทำนองเดียวกัน IBM ก็กล่าวถึงการพัฒนา Digital Twin ที่จะมีผลกระทบต่อการทำงานในอนาคต โดยเฉพาะการเข้ามาแทนที่การจำลองด้วยโมเดลคณิตศาสตร์แบบทั่วไป เพราะ Digital Twin จะเป็นการเชื่อมโยงข้อมูลของเครื่องจักรกายภาพกับเครื่องจักรเสมือนอย่างแท้จริง โดยผู้ใช้งานสามารถปรับเปลี่ยนที่เครื่องจักรเสมือนเพื่อจำลองสถานการณ์หรือดูผลกระทบได้อย่างรอบด้าน และทำได้ในราคาที่ถูกกว่าการจำลองจริงอีกด้วย ซึ่งจะช่วยร่นระยะเวลาการออกแบบและการผลิตผลิตภัณฑ์อีกด้วย (IBM, 2023)

## 1.2 วัตถุประสงค์

1) เพื่อศึกษาความเป็นไปได้ในการออกแบบและพัฒนาเทคโนโลยีคู่แฝดดิจิทัลบนพื้นฐานของเทคโนโลยี PLCnext
2) เพื่อทดสอบประสิทธิภาพของการใช้งานเทคโนโลยีคู่แฝดดิจิทัลกับการผลิตแบบอัตโนมัติในโรงงานอุตสาหกรรมอัจฉริยะ

### 1.2.1 อุปกรณ์ด้านฮาร์ดแวร์ (Hardware)

1) PLCnext AXC F 2152: เป็น PLC ที่มีความสามารถสูงในการรวบรวมและส่งข้อมูลจากโรงงานหรือโรงงานอัจฉริยะไปยังแพลตฟอร์ม IoT
2) คอมพิวเตอร์: ใช้ในการจัดเก็บและแสดงผลข้อมูลการทดสอบ

### 1.2.2 ซอฟต์แวร์ที่ใช้ในการพัฒนา (Software)

1) PLCnext Engineer: ใช้ในการตั้งค่าและโปรแกรม PLC เพื่อรวบรวมและส่งข้อมูล
2) ThingsBoard: ใช้เป็นแพลตฟอร์ม IoT ในการเก็บรวบรวมและวิเคราะห์ข้อมูลที่ได้รับจาก PLC และอุปกรณ์ IoT
3) Unity: ใช้ในการสร้างและแสดงผลแบบจำลองดิจิทัลเสมือนจริง โดยการดึงข้อมูลจาก ThingsBoard เพื่อแสดงผลแบบเรียลไทม์
4) MQTT Broker: ใช้ในการส่งข้อมูลระหว่าง PLC และ ThingsBoard
5) HTTP Server: ใช้ในการดึงข้อมูลจาก ThingsBoard ไปยัง Unity

6) WebSocket Server: ใช้ในการดึงข้อมูลแบบเรียลไทม์จาก ThingsBoard ไปยัง Unity

## 1.3 ขั้นตอนการดำเนินงาน

### 1.3.1 วางแผนการดำเนินงาน

1) กำหนดวัตถุประสงค์ของการศึกษา
2) วางแผนการรวบรวมข้อมูลจาก PLC
3) วางแผนการทดสอบโปรโตคอลการสื่อสาร

### 1.3.2 ศึกษาความเป็นไปได้

1) ทำการศึกษาโปรโตคอลการสื่อสาร MQTT, HTTP และ WebSocket
2) ทดสอบการส่งข้อมูลระหว่าง PLC และ ThingsBoard โดยใช้โปรโตคอล MQTT และ HTTP
3) ทดสอบการดึงข้อมูลจาก ThingsBoard ไปยัง Unity โดยใช้โปรโตคอล HTTP และ WebSocket

### 1.3.3 วิเคราะห์และออกแบบ

1) ออกแบบโครงสร้างของระบบดิจิทัลทวิน
2) กำหนดขั้นตอนการส่งข้อมูลจาก PLC ไปยัง ThingsBoard
3) กำหนดขั้นตอนการดึงข้อมูลจาก ThingsBoard ไปยัง Unity
4) ออกแบบการแสดงผลแบบเรียลไทม์ใน Unity

### 1.3.4 พัฒนาระบบ

1) ตั้งค่าและโปรแกรม PLC เพื่อรวบรวมและส่งข้อมูลผ่าน Node-RED
2) ตั้งค่า Node-RED ให้สามารถส่งข้อมูลขึ้น ThingsBoard
3) ตั้งค่า ThingsBoard เพื่อรับและจัดเก็บข้อมูลจาก PLC
4) สร้างโปรแกรม ใน Unity และเชื่อมต่อกับ ThingsBoard

### 1.3.5 ทดสอบแบบจำลองและปรับปรุงแก้ไข

1) ทดสอบการส่งข้อมูลระหว่าง PLC และ ThingsBoard โดยใช้โปรโตคอล MQTT และ HTTP

2) ทดสอบการดึงข้อมูลจาก ThingsBoard ไปยัง Unity โดยใช้โปรโตคอล HTTP และ WebSocket
3) วิเคราะห์ผลการทดสอบและปรับปรุงระบบตามความต้องการ

## 1.4 ประโยชน์ที่คาดว่าจะได้รับ

การศึกษานี้คาดหวังที่จะได้รับประโยชน์ดังนี้

1) สามารถเพิ่มประสิทธิภาพในการสื่อสารแบบเรียลไทม์ในระบบดิจิทัลทวิน
2) ลดความหน่วงเวลาในการส่งข้อมูลและเพิ่มความเสถียรในการสื่อสาร
3) เพิ่มความสามารถในการตรวจสอบและทำนายปัญหาที่อาจเกิดขึ้นในระบบ
4) ช่วยลดค่าใช้จ่ายและเพิ่มความปลอดภัยในการดำเนินงานในอุตสาหกรรมระบบอัตโนมัติ
5) ช่วยเพิ่มประสิทธิภาพในการจัดการทรัพยากรและลดการสูญเสียที่ไม่จำเป็น

# 2 ทฤษฎีและโครงการที่เกี่ยวข้อง

## 2.1 ทฤษฎีที่เกี่ยวข้อง

### 2.1.1 คู่แฝดดิจิทัล หรือ ดิจิทัลทวิน (Digital Twin)

ดิจิทัลทวิน คือ โมเดลเสมือนจริงที่สะท้อนการทำงานของระบบทางกายภาพ โดยใช้ข้อมูลจากเซ็นเซอร์และอุปกรณ์ IoT ทำให้สามารถวิเคราะห์และควบคุมระบบได้แบบเรียลไทม์ ดิจิทัลทวินช่วยให้สามารถติดตามการทำงานของระบบในเวลาจริงและสามารถทำนายปัญหาที่อาจเกิดขึ้นในอนาคตได้ การใช้งานดิจิทัลทวินครอบคลุมตั้งแต่การจำลองกระบวนการผลิต การทำนายการเสื่อมสภาพของเครื่องจักร การจำลองสถานการณ์ต่างๆ เพื่อหาวิธีแก้ไขที่เหมาะสม ไปจนถึงการพัฒนาผลิตภัณฑ์ใหม่ๆ ในอุตสาหกรรม ดิจิทัลทวินช่วยให้การตัดสินใจในการปรับปรุงกระบวนการผลิตมีความแม่นยำมากขึ้นโดยใช้ข้อมูลที่เก็บรวบรวมมาเพื่อวิเคราะห์และปรับปรุงการทำงานได้อย่างมีประสิทธิภาพ นอกจากนี้ ดิจิทัลทวินยังลดความเสี่ยงในการทดลองแนวคิดใหม่ๆ โดยสามารถทดลองบนแบบจำลองเสมือนจริงก่อนการนำไปใช้จริง ซึ่งช่วยลดเวลาและค่าใช้จ่ายได้อย่างมาก (Lu et al., 2020; Deng et al., 2021)

### 2.1.2 การบำรุงรักษาเชิงพยากรณ์ (Predictive Maintenance)

การบำรุงรักษาเชิงพยากรณ์ด้วยดิจิทัลทวินช่วยให้สามารถทำนายการเสียหายของอุปกรณ์ล่วงหน้าและดำเนินการบำรุงรักษาได้อย่างมีประสิทธิภาพ ลดการหยุดชะงักของการผลิตและเพิ่มความปลอดภัยในระบบ โดยใช้ข้อมูลจากดิจิทัลทวินเพื่อทำนายการเสื่อมสภาพของเครื่องจักรและอุปกรณ์ ข้อมูลจากเซ็นเซอร์จะถูกวิเคราะห์เพื่อหาสัญญาณที่บ่งบอกถึงการเสื่อมสภาพหรือปัญหาที่อาจเกิดขึ้น ทำให้สามารถดำเนินการบำรุงรักษาได้ก่อนที่ปัญหาจะเกิดขึ้นจริง ซึ่งช่วยลดความเสียหายและการหยุดชะงักของการผลิต การบำรุงรักษาเชิงพยากรณ์ยังช่วยให้การจัดการทรัพยากรมีประสิทธิภาพมากขึ้น โดยสามารถวางแผนการบำรุงรักษาได้อย่างเหมาะสม ลดค่าใช้จ่ายในการบำรุงรักษาและการซ่อมแซม นอกจากนี้ยังช่วยเพิ่มความปลอดภัยในการทำงานโดยสามารถตรวจจับปัญหาที่อาจเกิดขึ้นและแก้ไขได้ทันที (Madni et al., 2019)

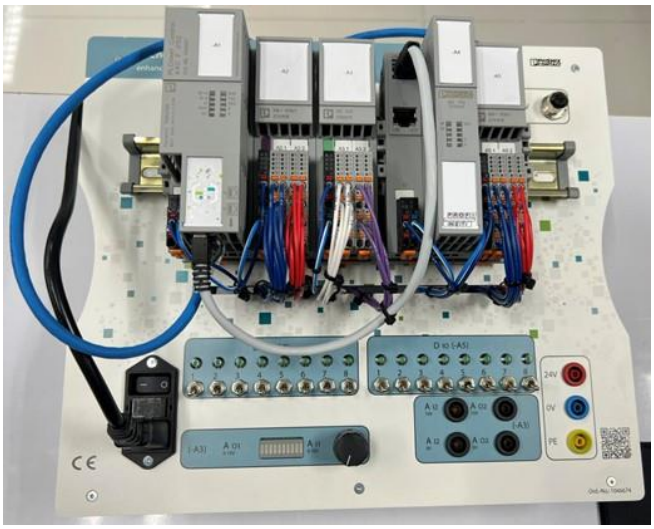
### 2.1.3 การจำลองและการวิเคราะห์ (Simulation and Analysis)

ดิจิทัลทวินช่วยให้สามารถจำลองการทำงานของระบบและวิเคราะห์ข้อมูลเพื่อปรับปรุงกระบวนการทำงานได้ การจำลองช่วยให้สามารถทดลองแนวคิดใหม่ๆ โดยไม่ต้องทำการทดสอบจริงซึ่งอาจมีค่าใช้จ่ายสูง การวิเคราะห์ข้อมูลช่วยให้สามารถปรับปรุงกระบวนการทำงานได้อย่างต่อเนื่อง การจำลองใช้ดิจิทัลทวินเพื่อสร้างแบบจำลองเสมือนจริงของระบบทางกายภาพ ทำให้สามารถทดลองสถานการณ์ต่างๆ และประเมินผลกระทบของการตัดสินใจได้อย่างละเอียด ลดความเสี่ยงในการทดลองแนวคิดใหม่ๆ และเพิ่มความรวดเร็วในการพัฒนา การวิเคราะห์ข้อมูลใช้ข้อมูลจากดิจิทัลทวินและ IoT เพื่อหาสาเหตุของปัญหาและหาวิธีการแก้ไขที่เหมาะสม ช่วยให้กระบวนการทำงานมีประสิทธิภาพมากขึ้น นอกจากนี้ยังช่วยในการพัฒนาผลิตภัณฑ์ใหม่โดยสามารถทดลอง

แนวคิดและปรับปรุงการออกแบบบนแบบจำลองเสมือนจริงก่อนการผลิตจริง ช่วยลดเวลาและค่าใช้จ่ายในการพัฒนาได้อย่างมาก (Fuller et al., 2020)

### 2.1.4 ระบบควบคุมอัตโนมัติ PLCnext

PLCnext เป็นระบบควบคุมอัตโนมัติที่สามารถปรับแต่งได้ตามความต้องการของผู้ใช้งาน โดย PLCnext รองรับการเขียนโปรแกรมด้วยภาษา IEC 61131-3 และมีความสามารถมากกว่า PLC ทั่ว ๆ ไป คือ การรองรับการเขียนโปรแกรมด้วยภาษาระดับสูง อาทิ ภาษาโปรแกรมที่ได้รับความนิยมอื่น ๆ เช่น C/C++, C#, และ MATLAB Simulink ทำให้สามารถสร้างโปรแกรมควบคุมที่มีความซับซ้อนได้อย่างง่ายดาย นอกจากนี้ PLCnext ยังรองรับการเชื่อมต่อกับโปรโตคอลการสื่อสารต่าง ๆ ทำให้สามารถใช้งานร่วมกับอุปกรณ์และระบบอื่น ๆ ได้อย่างมีประสิทธิภาพ



ภาพที่ 1 PLCnext AXC F 2152

จากภาพที่ 1 PLC เครื่องนี้คือ PLCnext AXC F 2152 ซึ่งใช้สำหรับการส่งข้อมูลอินพุตโดยการกดสวิตช์เปิด/ปิด เพื่อเป็นตัวแทนของอุปกรณ์ที่จะต่อกับเครื่อง PLC นี้ในสถานการณ์การใช้งานจริงในโรงงาน ค่าที่ส่งออกไปคือค่า true และ false ที่จะใช้แทนว่าอุปกรณ์ที่ต่อกับ PLC ทำงานอยู่หรือไม่

### 2.1.5 โปรแกรม Node-RED

โปรแกรม Node-RED เป็นโปรแกรมที่ใช้ในการเชื่อมต่ออุปกรณ์ IoT และการประมวลผลข้อมูลแบบเรียลไทม์ โดยใช้วิธีการเขียนโปรแกรมแบบภาพ (Flow-Based Programming) ซึ่งช่วยให้นักพัฒนาไม่จำเป็นต้องเขียนโค้ดที่ซับซ้อน Node-RED รองรับการทำงานร่วมกับโปรโตคอลต่าง ๆ เช่น MQTT, HTTP และ WebSocket

ทำให้สามารถเชื่อมต่อกับเซ็นเซอร์และอุปกรณ์ต่าง ๆ ได้อย่างง่ายดาย นอกจากนี้ Node-RED ยังมีการใช้งานในหลากหลายโครงการ ทั้งในด้านการเกษตร การแพทย์ และการอุตสาหกรรม


ภาพที่ 2 โปรแกรม Node-RED

Node-RED (ภาพที่ 2) สามารถที่จะรองรับการทำงานของ communication protocol ได้อย่างหลากหลาย จึงใช้ Node-RED เพื่อการส่งข้อมูลไปยัง ThingsBoard โดยใช้ communication protocol ที่หลากหลาย และยังสามารถใช้ Node-RED เพื่อการเก็บ Log ข้อมูลที่เกิดขึ้นเพื่อนำมาคำนวณและการหน่วงของการทำงานได้อีกด้วย

### 2.1.6 โปรแกรม Unity

โปรแกรม Unity เป็นเอนจินพัฒนาเกมที่ได้รับความนิยมมากที่สุดในปัจจุบัน ซึ่งรองรับการพัฒนาเกมในรูปแบบ 2D และ 3D รวมถึงการสร้างประสบการณ์แบบ VR และ AR โปรแกรม Unity มีเครื่องมือที่ครบครันสำหรับการสร้างเกมและแอปพลิเคชันที่มีความซับซ้อนสูง โดยสามารถใช้งานร่วมกับภาษาโปรแกรม C# และมีเอกสารและชุมชนผู้ใช้งานที่ใหญ่ ทำให้ผู้พัฒนาสามารถเรียนรู้และแก้ไขปัญหาได้อย่างรวดเร็ว


ภาพที่ 3 โปรแกรม Unity

Unity (ภาพที่ 3) เป็นเอนจินที่ได้รับความนิยมและเหมาะสมสำหรับการทำ simulation และการทำแบบจำลองที่สมจริง ทางผู้พัฒนาจึงเล็งเห็นสมควรว่าการศึกษาความเป็นไปได้ในการสร้างดิจิทัลทวินด้วย Unity นั้นมีความเป็นไปได้สูง โดยในการศึกษานี้ จะใช้ C# สคริปต์ในการดึงข้อมูลและประมวลผลข้อมูลที่ดึงมาใช้งาน

### 2.1.7 แพลตฟอร์ม ThingsBoard

แพลตฟอร์ม ThingsBoard (ภาพที่ 4) เป็นแพลตฟอร์ม IoT ที่ใช้ในการจัดการและแสดงผลข้อมูลจากอุปกรณ์ IoT ต่าง ๆ ThingsBoard รองรับการเชื่อมต่อกับโปรโตคอลการสื่อสารหลายประเภท เช่น MQTT, CoAP และ HTTP ทำให้สามารถเก็บรวบรวมและวิเคราะห์ข้อมูลจากอุปกรณ์ต่าง ๆ ได้ในเวลาเดียวกัน นอกจากนี้

ThingsBoard ยังมีฟีเจอร์ที่ช่วยในการสร้างแดชบอร์ดและการแจ้งเตือน ซึ่งช่วยให้ผู้ใช้งานสามารถติดตามและควบคุมอุปกรณ์ได้อย่างมีประสิทธิภาพ


ภาพที่ 4 แพลตฟอร์ม ThingsBoard

ด้วยข้อจำกัดของ Unity ที่ออกแบบมาเพื่อเป็นเกมเอนจินทำให้ตัว Unity เองไม่สามารถรับสัญญาณจาก PLC หรือ Node-RED โดยตรงได้ ทำให้ ThingsBoard เข้ามาเป็นตัวกลางสำหรับให้ Unity สามารถดึงข้อมูลเพื่อนำไปใช้งานต่อนั่นเอง

## 2.2 งานที่เกี่ยวข้อง (Related Works)

### 2.2.1 โปรโตคอลที่ใช้กันอย่างแพร่หลายในเทคโนโลยีคู่แฝดดิจิทัลกับ PLCnext

การสื่อสารระหว่างแบบจำลองดิจิทัลและคู่สัญญาในโลกจริงสามารถทำได้ผ่านโปรโตคอลต่างๆ ซึ่งแต่ละโปรโตคอลมีคุณสมบัติเฉพาะที่เหมาะสมกับความต้องการที่แตกต่างกัน ในเทคโนโลยีคู่แฝดดิจิทัล (DTT) ที่ใช้ร่วมกับคอนโทรลเลอร์ PLCnext มีโปรโตคอลหลายตัวที่ได้รับความนิยมเนื่องจากมีความน่าเชื่อถือในการส่งข้อมูลแบบเรียลไทม์ หนึ่งในนั้นคือ MQTT (Message Queuing Telemetry Transport) ซึ่งเป็นที่นิยมเนื่องจากมีน้ำหนักเบาและมีประสิทธิภาพในการส่ง/รับข้อความ ทำให้เหมาะสำหรับสถานการณ์ที่ต้องการแบนด์วิธต่ำและการอัปเดตอย่างรวดเร็ว อีกโปรโตคอลหนึ่งที่ใช้งานกันคือ HTTP (HyperText Transfer Protocol) ซึ่งได้รับความนิยมเนื่องจากมีความน่าเชื่อถือและใช้งานง่าย โดยเฉพาะในปฏิสัมพันธ์บนเว็บระหว่างคู่แฝดดิจิทัลและแพลตฟอร์มเช่น ThingsBoard โปรโตคอลการสื่อสารอีกตัวที่ได้รับความนิยมคือ OPC UA (Open Platform Communications Unified Architecture) ซึ่งมีชื่อเสียงในการทำงานได้ดีกับหลายแพลตฟอร์มและมีคุณสมบัติความปลอดภัยที่แข็งแกร่ง ทำให้เป็นตัวเลือกที่ดีสำหรับโครงการอัตโนมัติอุตสาหกรรมที่ใช้คอนโทรลเลอร์ PLCnext ในการดึงข้อมูลการผลิตจากโรงงานผลิต จำเป็นต้องมีอุปกรณ์ที่ทำหน้าที่นี้ ซึ่งก็คือ PLC (Programmable Logic Controller) ในกรณีนี้ PLCnext ถูกนำมาใช้ เทคโนโลยี PLCnext ที่พัฒนาโดยบริษัท

Phoenix Contact ให้สภาพแวดล้อมที่เปิดกว้างและสนับสนุนโปรโตคอลการสื่อสารหลายตัว ช่วยปรับปรุงการโต้ตอบและการแบ่งปันข้อมูลแบบเรียลไทม์ระหว่างระบบทางกายภาพและคู่แฝดดิจิทัล จึงเหมาะสำหรับเทคโนโลยีคู่แฝดดิจิทัล ความสามารถในการปรับตัวของเทคโนโลยี PLCnext ช่วยให้สามารถผสานรวมกับโปรโตคอลการสื่อสารต่างๆ ได้อย่างง่ายดาย ขยายประสิทธิภาพของเทคโนโลยีคู่แฝดดิจิทัลในการตรวจสอบ การควบคุม และการวิเคราะห์กระบวนการอุตสาหกรรม (Wukkadada et al., 2018).

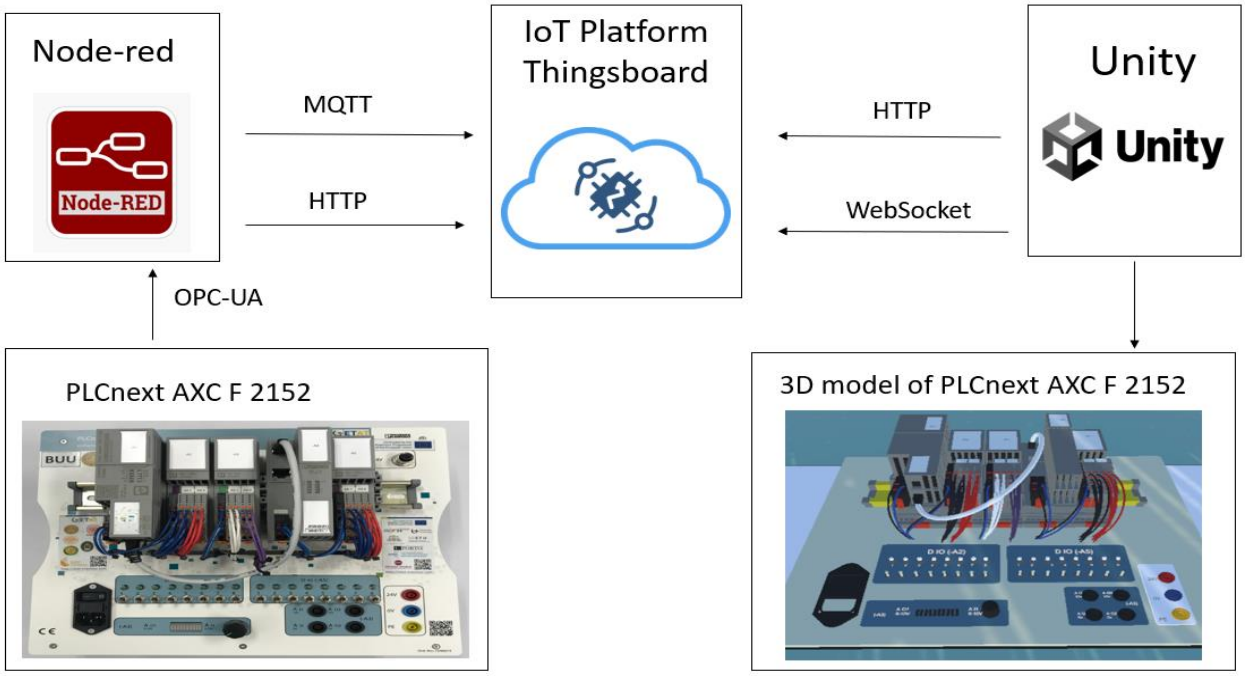
### 2.2.2 คำนิยามของเทคโนโลยีคู่แฝดดิจิทัล

ในภาคการผลิต แนวคิดของเทคโนโลยีคู่แฝดดิจิทัล (DTT) ถูกนำเสนอครั้งแรกโดย Michael Grieves จากมหาวิทยาลัยมิชิแกนในงานนำเสนอเกี่ยวกับการจัดการวงจรชีวิตผลิตภัณฑ์ (PLM) ในปี 2002 (Grieves & Vickers, 2017) ผู้เขียนได้นำเสนอโมเดลเชิงแนวคิดสำหรับตัวแทนดิจิทัลเสมือนของผลิตภัณฑ์ทางกายภาพ ซึ่งรวมถึงพื้นที่จริง พื้นที่เสมือน และการไหลของข้อมูลระหว่างพื้นที่ทั้งสอง เมื่อเวลาผ่านไป รายละเอียดเพิ่มเติมของ DTT ถูกนำเสนอ อธิบายว่า DTT เป็นการผสมผสานระหว่างโลกจริงและเสมือนที่สะท้อนกันเพื่อตรวจสอบว่าชิ้นส่วนในโลกจริงทำงานอย่างไรตลอดอายุการใช้งาน สิ่งนี้ให้มุมมองเกี่ยวกับ DTT และสิ่งที่เกี่ยวข้อง จากนั้นมีการกล่าวถึงว่า DT มีข้อมูลทางกายภาพและการทำงานที่สำคัญทั้งหมดของเครื่องจักรหรือระบบ (Madni et al., 2019) ซึ่งเน้นถึงวิธีการเคลื่อนย้ายข้อมูลและการควบคุมระหว่างโมเดลจริงและเสมือน เรื่องราวของ DTT พัฒนาต่อไปและถูกอธิบายว่าเป็นกลุ่มข้อมูลเสมือนที่แสดงผลิตภัณฑ์ทางกายภาพจริงหรือที่เป็นไปได้อย่างละเอียด ตั้งแต่ชิ้นส่วนที่เล็กที่สุดไปจนถึงรูปร่างโดยรวม ความสำคัญของการเปรียบเทียบคู่แฝดดิจิทัลกับการออกแบบจริงเพื่อทำให้ผลิตภัณฑ์สุดท้ายดีขึ้นโดยการแก้ไขช่องว่างระหว่างการวางแผนและการดำเนินการถูกเน้นย้ำ ต่อมา DTT ถูกเรียกว่าเป็นโมเดลที่มีชีวิตซึ่งแสดงสิ่งที่เป็นแบบเดียวกันกับสิ่งในโลกจริงหรือระบบจริงซึ่งมีการเปลี่ยนแปลงตามข้อมูลใหม่และสามารถทำนายสิ่งที่จะทำต่อไปได้อย่างแม่นยำ (Raza et al., 2020; Zheng et al., 2021; Deng et al., 2021) คำนิยามเหล่านี้ชี้ให้เห็นถึงธรรมชาติของ DTT และความสามารถในการปรับปรุงได้ตลอดเวลาพร้อมกับข้อมูลเพิ่มเติมจากสิ่งที่แท้จริง มุ่งเน้นไปที่การดูแลในเวลาจริงและการทำนายการบำรุงรักษาซึ่งเป็นประโยชน์ในงานบางอย่างเช่นงานก่อสร้าง งานการผลิต เป็นต้น การศึกษานี้นิยามเทคโนโลยีคู่แฝดดิจิทัลว่าเป็นตัวแทนเสมือนของสิ่งที่แท้จริง ซึ่งเป็นไปได้โดยเซ็นเซอร์ เครือข่ายการสื่อสาร ข้อมูล และโมเดล 3 มิติ เครื่องมือเหล่านี้อนุญาตให้มีการอัปเดตแบบเรียลไทม์และการสื่อสารสองทาง ทำให้แน่ใจว่าโมเดลเสมือนยังคงเป็นสำเนาของระบบหรือเครื่องจักรในโลกจริง และการพัฒนาโมเดลคู่แฝดดิจิทัลช่วยให้สามารถประเมินความเสี่ยงการชนได้แบบเรียลไทม์ การวิจัยที่ดำเนินการโดย (Leonardo et al., 2020) แสดงให้เห็นถึงการใช้คู่แฝดดิจิทัลในการประเมินความเสี่ยงการชนในสภาพแวดล้อมก่อสร้างโดยใช้ข้อมูลแบบเรียลไทม์เพื่อปรับปรุงความปลอดภัยและลดอุบัติเหตุ

# 3 วิธีการดำเนินงานโครงการ

## 3.1 ภาพรวมของระบบ

ในบทนี้จะอธิบายถึงวิธีการศึกษาและการดำเนินงานในการใช้ดิจิทัลทวินในอุตสาหกรรมระบบอัตโนมัติ โดยเน้นที่การประเมินประสิทธิภาพของโปรโตคอลการสื่อสารที่ใช้ในการส่งข้อมูลระหว่างระบบทางกายภาพและระบบดิจิทัล การศึกษานี้ใช้ PLCnext AXC F 2152 Node-RED แพลตฟอร์ม IoT ThingsBoard และเอนจิน 3D Unity โดยมีเป้าหมายเพื่อวัดความหน่วงเวลาในการส่งข้อมูลและการดึงข้อมูล




ภาพที่ 5 ภาพรวมระบบ

จากภาพที่ 5 ในภาพนี้จะแสดงถึงโครงสร้างระบบที่ใช้ในการทำการทดสอบครั้งนี้ โดยจะเริ่มต้นที่เครื่อง PLCnext AXC F 2152 จะใช้สำหรับการส่งอินพุตข้อมูล ด้วย OPC UA ไปที่ Node-RED หลังจากนั้น จะใช้ Node-RED แปลงสัญญาณข้อมูลที่ได้เป็นโปรโตคอลเพื่อการสื่อสารจำนวน 2 ประเภท ได้แก่ 1. MQTT และ 2. HTTP โดยข้อมูลจาก Node-RED จะถูกส่งขึ้นไปที่ IoT platform ThingsBoard และจะถูกเรียกใช้ใน Unity โดยการใช้โปรโตคอล HTTP และ WebSocket และข้อมูลที่เรียกมาจะถูกแสดงผลผ่านโปรแกรม Unity

## 3.2 ขั้นตอนการดำเนินงาน

งานวิจัยนี้คือการตรวจสอบการหน่วงเวลาในการส่งข้อมูลเมื่อใช้วิธีการและโปรโตคอลการสื่อสารต่าง ๆ ในการตั้งค่าเพื่อนำไปสู่ Digital Twin และเพื่อรองรับกับการทำงานร่วมกับโรงงานอัจฉริยะในอุตสาหกรรมระบบอัตโนมัติ 4.0 การตั้งค่านี้จะประกอบไปด้วย PLCnext AXC F 2152, Node-RED เป็นตัวกลางระหว่าง PLCnext และแพลตฟอร์มข้อมูล, ThingsBoard เป็นแพลตฟอร์มข้อมูล IoT, และ Unity เป็นเอนจินเกมสำหรับการแสดงภาพสามมิติ ระบบทำงานดังนี้ ขั้นแรก PLC ส่งข้อมูลไปยัง Node-RED จากนั้น Node-RED ส่งต่อข้อมูลไปยัง ThingsBoard โดยใช้โปรโตคอล MQTT หรือ HTTP หลังจากนั้น Unity จะได้รับข้อมูลนี้จาก ThingsBoard โดยใช้โปรโตคอล WebSocket หรือ HTTP ส่วนต่อไปนี้จะอธิบายการตั้งค่าในส่วนต่างๆ

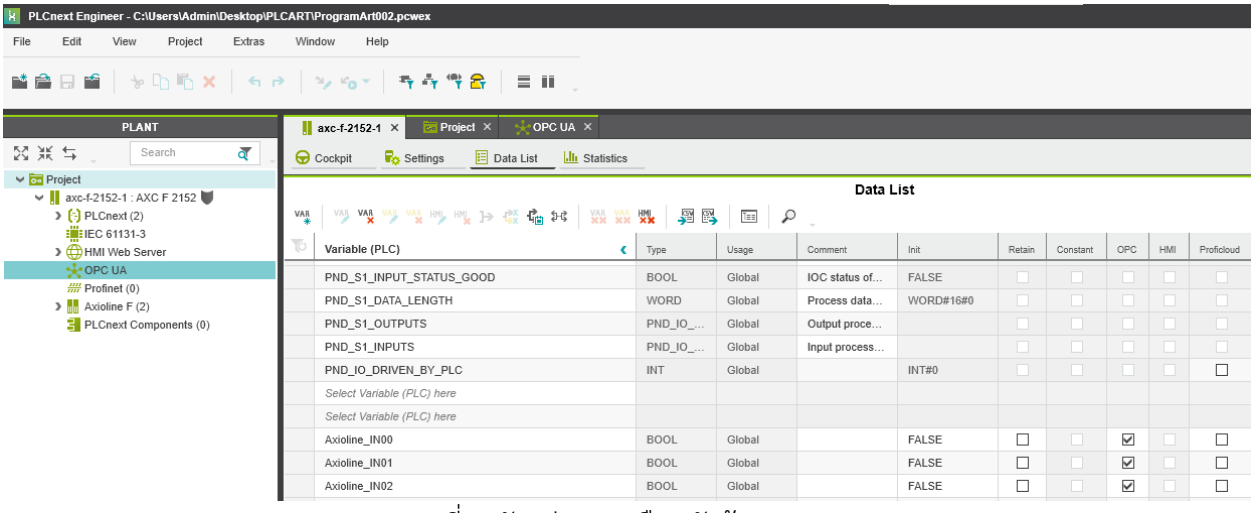
### 3.2.1 การวางแผนการศึกษา

การวางแผนการศึกษารวมถึงการกำหนดวัตถุประสงค์ การรวบรวมข้อมูล และการวางแผนการทดสอบโปรโตคอลการสื่อสาร การดำเนินงานจะถูกแบ่งออกเป็นหลายส่วนย่อยเพื่อให้สามารถวิเคราะห์และเปรียบเทียบผลลัพธ์ได้อย่างละเอียด

### 3.2.2 การตั้งค่าและโปรแกรม PLC และ UA expert

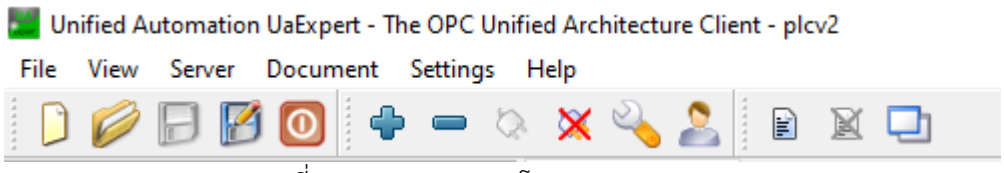
ในการตั้งค่าและโปรแกรม PLCnext AXC F 2152 จะใช้ซอฟต์แวร์ PLCnext Engineer เพื่อทำการโปรแกรมและตั้งค่าให้สามารถรวบรวมและส่งข้อมูลไปยังแพลตฟอร์ม ThingsBoard ได้

1. ไปที่หัวข้อ OPC UA และเลือกหัวข้อของข้อมูลที่ต้องการติดตามสถานะ ในการทดลองนี้ผู้พัฒนาเลือก Axioline_IN0-2 เป็นตัวอย่างดังภาพที่ 6



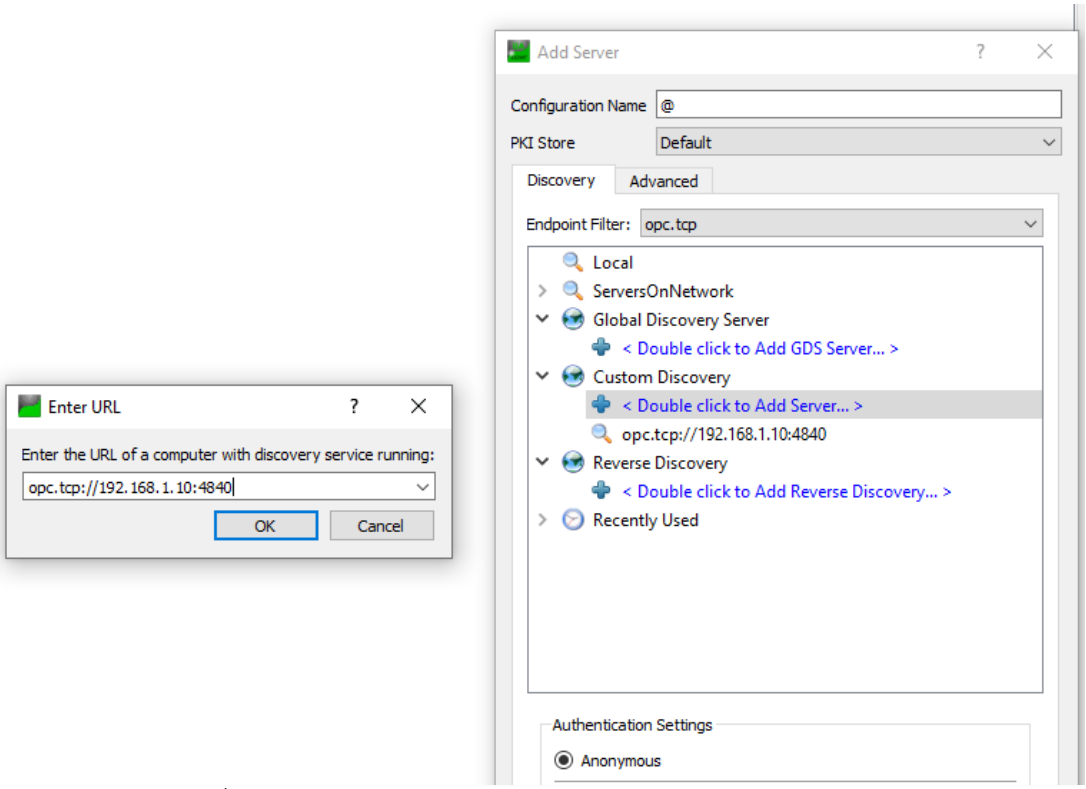

ภาพที่ 6 ตัวอย่างการเลือกหัวข้อ OPC UA

2. เพื่อที่จะสามารถทราบถึง Node ID ของสวิตช์ที่เลือกรวมถึงสามารถตรวจสอบสถานะข้อมูลได้ ให้ไปที่โปรแกรม UA expert ที่ เลือกไปที่ Menu bar และกดที่เครื่องหมาย + ตามภาพที่ 7 เพื่อเชื่อมต่อเข้ากับ PLC




ภาพที่ 7 Menu bar ของโปรแกรท UA expert

3. ดับเบิลคลิกที่ เครื่องหมาย + ใต้หัวข้อ Custom discovery จากนั้นใส่ IP OPC UA Address ของ PLC ที่เป็นพอร์ท 4840 โดยทั่วไปแล้วคือ IP address ของเครื่อง PLC แต่ใช้พอร์ท 4840 (ภาพที่ 8)

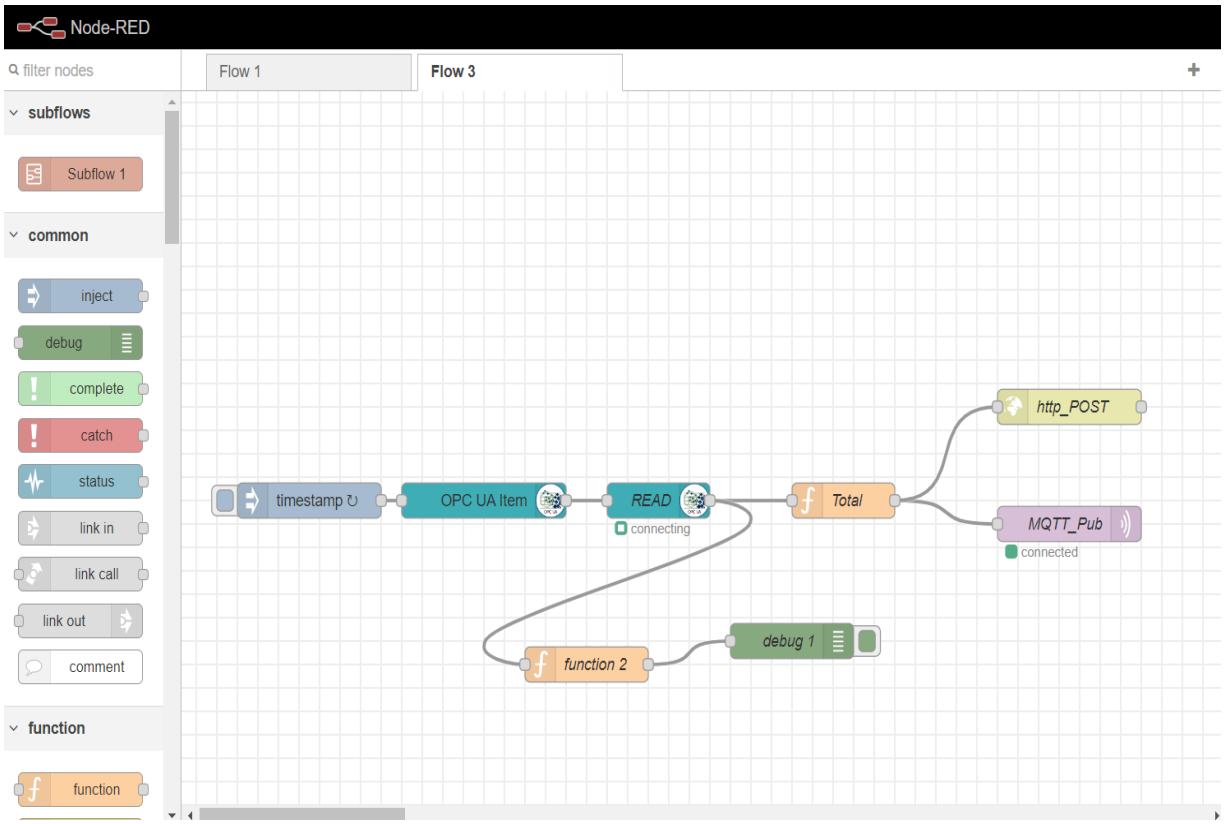



ภาพที่ 8 การใส่ IP OPC UA Server ในโปรแกรม UA expert

4. เลือกไปที่ PLC จะพบกับ สวิตช์ที่เลือกหัวข้อ OPC UA ไว้ ให้ลากวางไปที่ Data Access View
5. เลือกที่สวิตช์ จะเห็น ID ของสวิตช์นั้น ๆ ซึ่งจะใช้ ID ตัวนี้ใน OPC UA Item node ใน Node-RED

### 3.2.3 การตั้งค่าและโปรแกรม Node-RED

1. Workflow ของ Node-RED



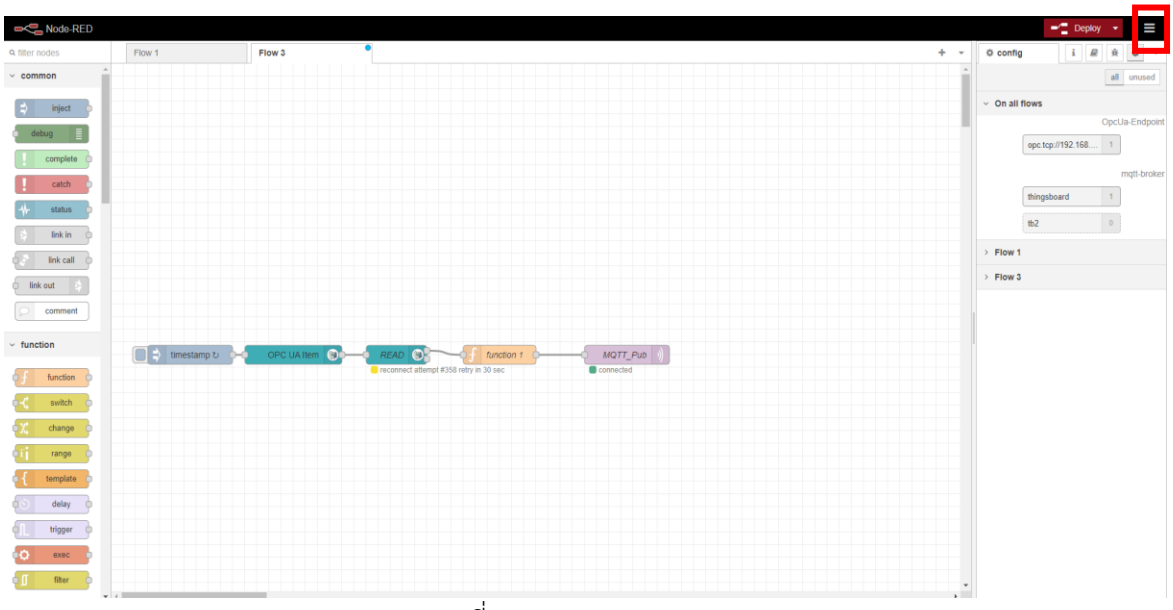

ภาพที่ 9 ภาพรวมของโปรแกรมใน Node-RED

จากภาพที่ 9 Work flow ของการทำงานนี้คือ ใช้ OPC UA Item เพื่อติดตามสถานะของ ID สวิตช์ที่ถูกเลือก จากนั้นโยงเข้ากับ READ node เพื่อทำการอ่านค่าจากสวิตช์ที่ถูกติดตามโดย OPC UA Item node หลังจากนั้น จะใช้ Function node เพื่อเขียนสคริปต์ แปลงสัญญาณที่อ่านได้ให้อยู่ในรูปของข้อความ จากนั้นจึงใช้ HTTP node และ MQTT node ในการ Post หรือ Publish ข้อความนี้เข้าสู่ ThingsBoard

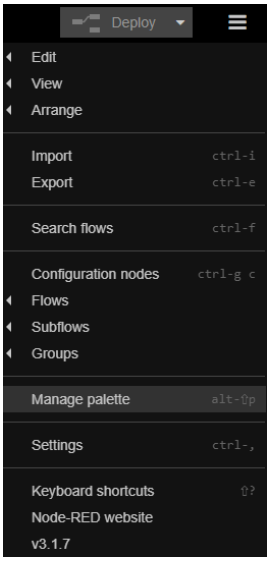
2. Library ที่จำเป็นในการใช้ Node-RED เพื่อเป็นตัวกลาง

ในการตั้งค่าและโปรแกรม Node-RED เพื่อทำการส่งข้อมูลขึ้น ThingsBoard จำเป็นต้องมีการติดตั้ง Library ใน Node-RED ดังนี้

2.1. เลือกเมนู  ตามภาพที่ 10 และเลือกหัวข้อ Manage palette ตามภาพที่ 11

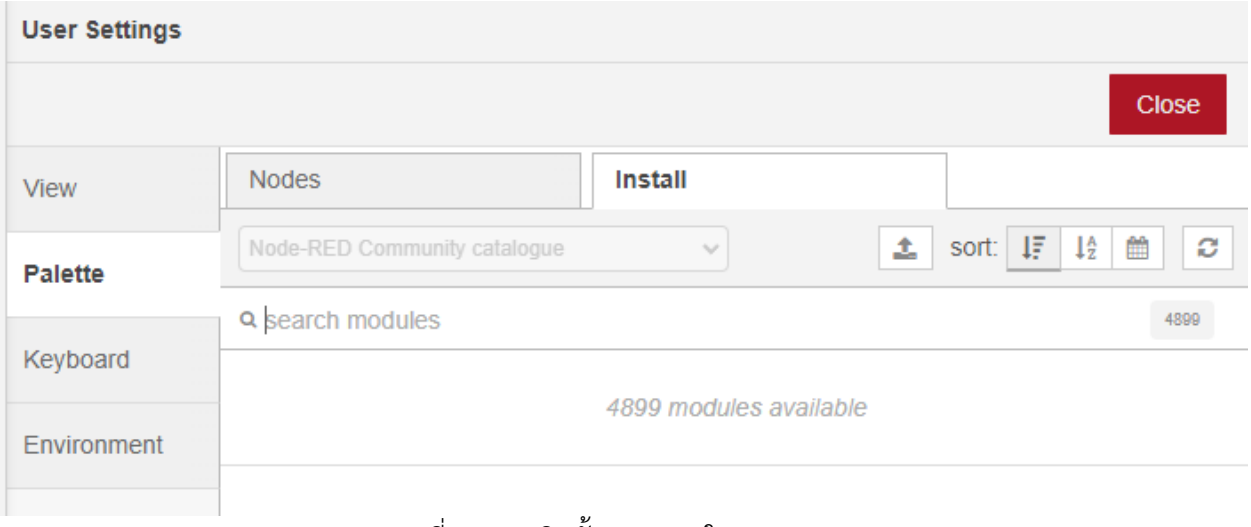



ภาพที่ 10 UI ของ Node-RED



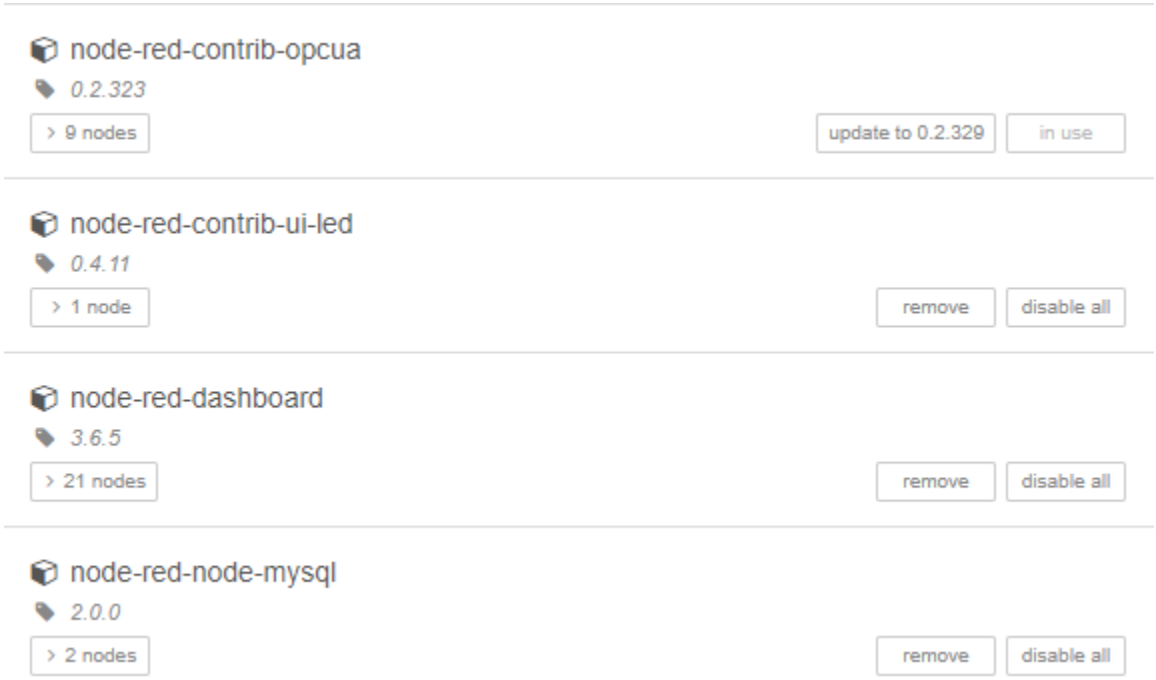

ภาพที่ 11 หน้าต่าง popup หลังจากกดที่ Menu

2.2. จากภาพที่ 12 เลือกหัวข้อ Install เพื่อทำการค้น Library ที่ต้องการใช้งาน



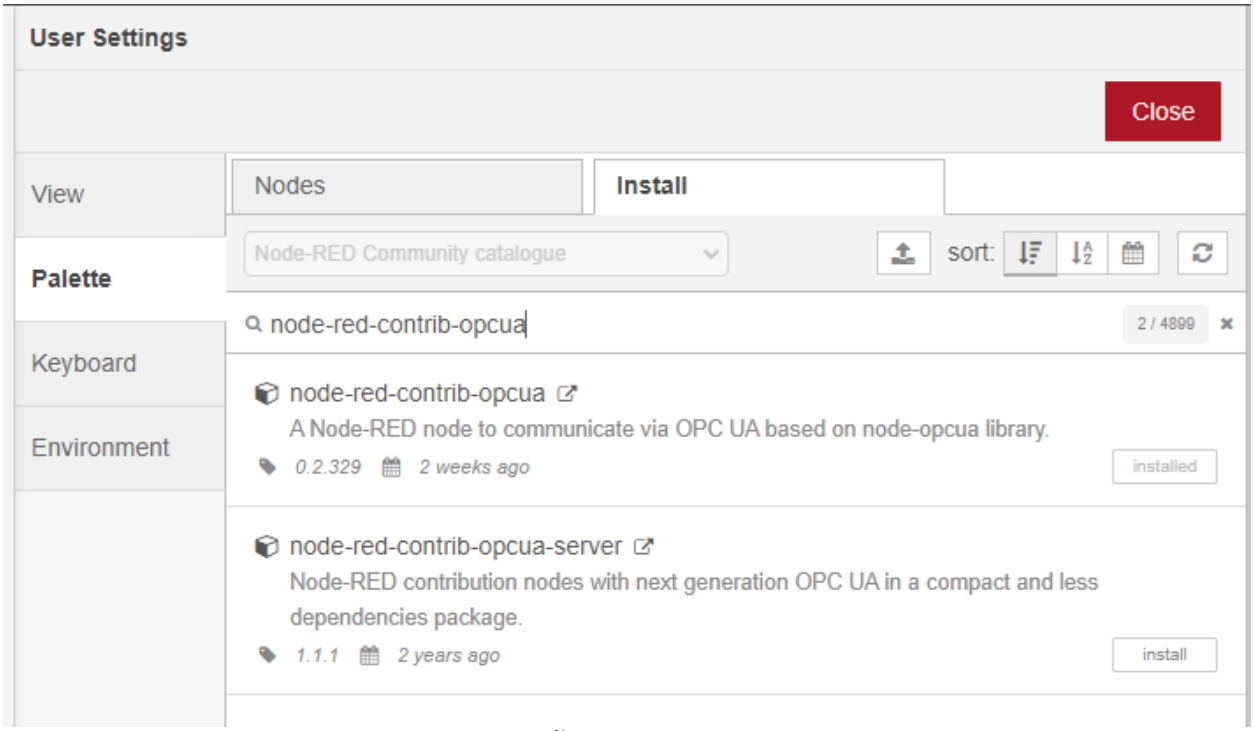

ภาพที่ 12 การติดตั้ง Library ใน Node-RED

2.3. Library ที่ให้ทำการติดตั้ง Library มีดังนี้




ภาพที่ 13 การติดตั้ง Library ใน Node-RED Library (ต่อ)

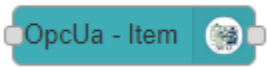
2.4. ให้พิมพ์ที่ช่องค้นหา จากนั้นให้กดปุ่ม install เมื่อการติดตั้งสำเร็จจะกลายเป็น installed (ภาพที่ 14)




ภาพที่ 14 ภาพการติดตั้งที่สำเร็จปุ่มจะกลายเป็น installed

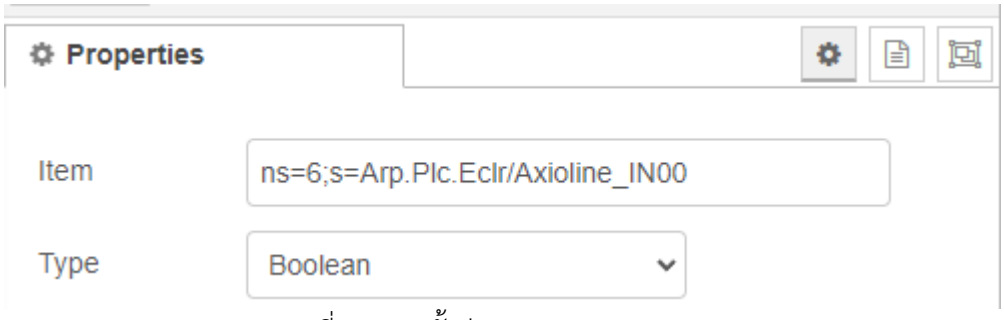
3. การวาง Node ที่ต้องการใช้งาน

3.1. ใช้ OPC UA Item node (ภาพที่ 15) เพื่อรับข้อมูลที่ส่งมาจาก PLC สามารถลากวางได้เลย




ภาพที่ 15 OPC UA Item node

3.2. ดับเบิลคลิกที่ OPC UA Item node เพื่อตั้งค่า OPC UA Item node และจะใส่ ID สวิตช์ที่ต้องการติดตามลงที่ช่อง Item สำหรับ Type ของข้อมูลให้พิจารณาตามข้อมูลที่จะใช้ ตามภาพที่ 16

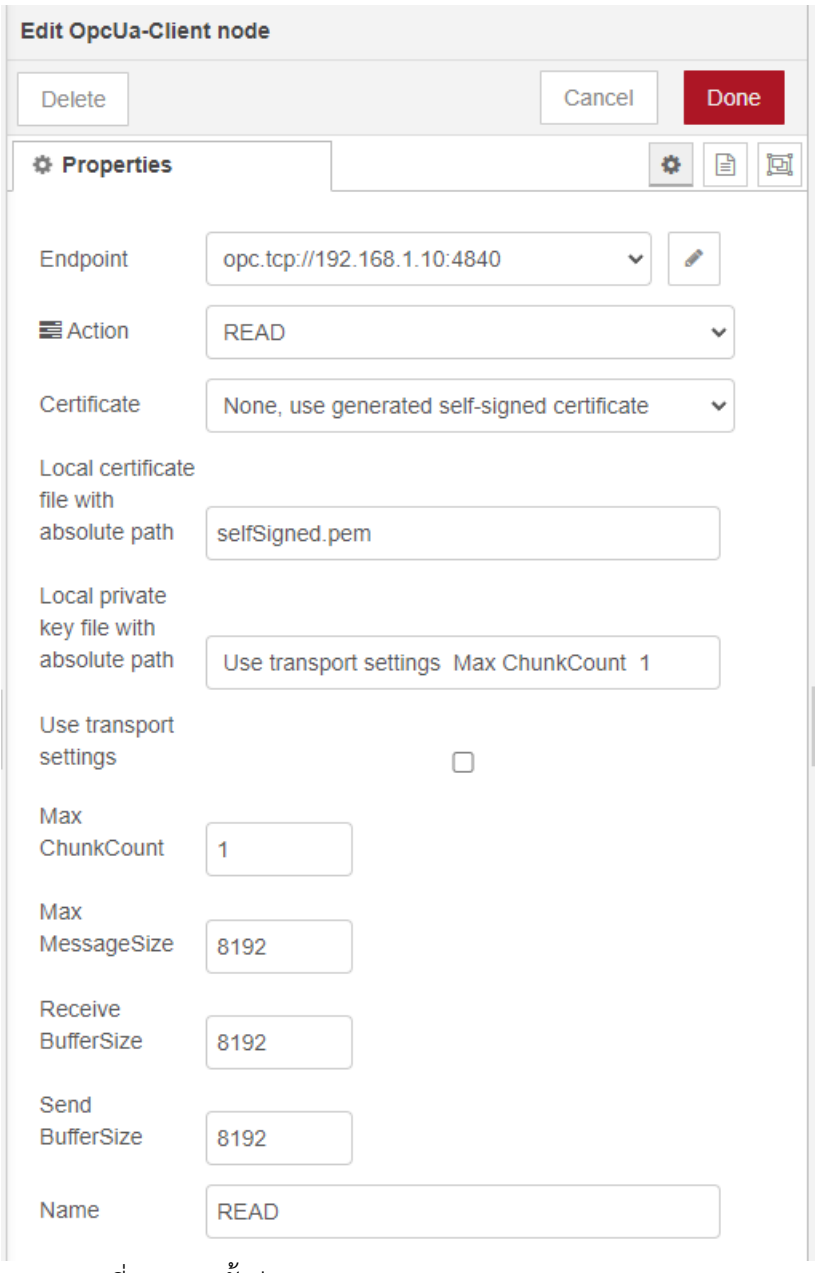



ภาพที่ 16 การตั้งค่า OPC UA Item node

3.3. ลากวาง READ (ภาพที่ 17)


ภาพที่ 17 READ node หรือ OPC UA Client node

3.4. เพื่ออ่านค่าที่ติดตามที่เลือกผ่าน OPC UA Item node หน้าที่ของ READ node จะติดตามเฉพาะค่านั้นๆ จาก OPC UA server ที่เลือก ให้ทำการดับเบิลคลิกและกรอก Endpoint เป็น OPC UA Server หรือก็คือ IP PLC ที่ใช้พอร์ท 4840 ตามภาพที่ 18




ภาพที่ 18 การตั้งค่า READ node และ OPC UA Client node

3.5. ลากวาง function node (ภาพที่ 19 )

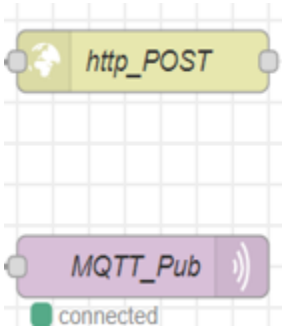

ภาพที่ 19 Function node

3.6. เขียนสคริปต์ลง Function node เพื่อแปลงข้อมูลที่รับมาให้อยู่ในรูปข้อความ ตามภาพที่ 20

```
1  var value = msg.payload;
2
3  // Structure it according to ThingsBoard telemetry format
4  msg.payload = {
5      "IN00": value
6  };
7
8  return msg;
```

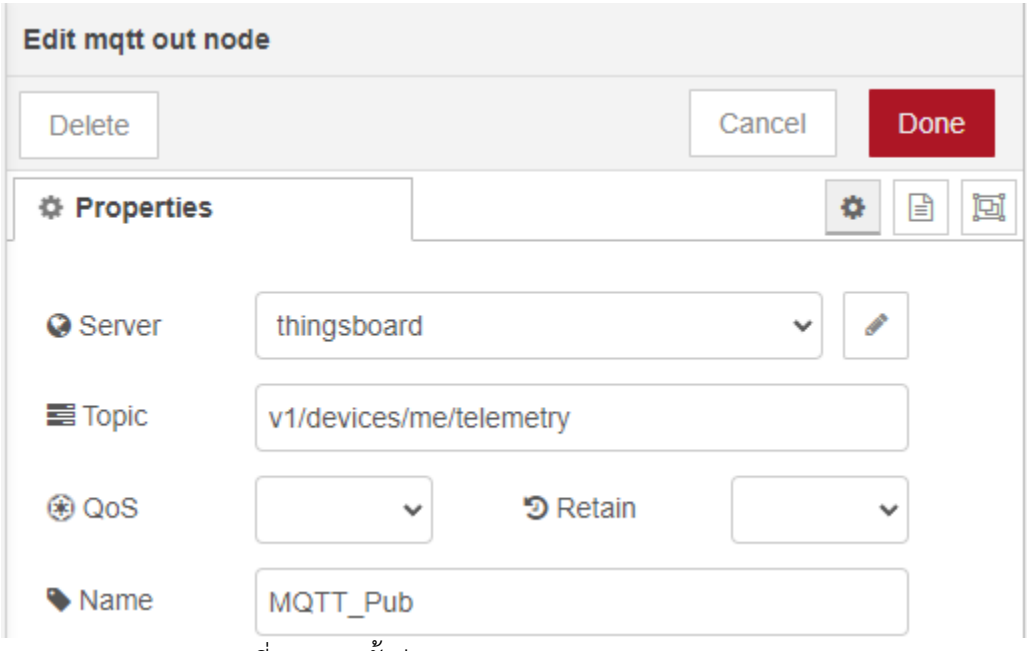
ภาพที่ 20 โค้ดสำหรับแปลงสัญญาณเป็นข้อความ

3.7. ลากวาง communication protocol node ที่ต้องการใช้งาน (HTTP และ MQTT) ตามภาพที่ 21



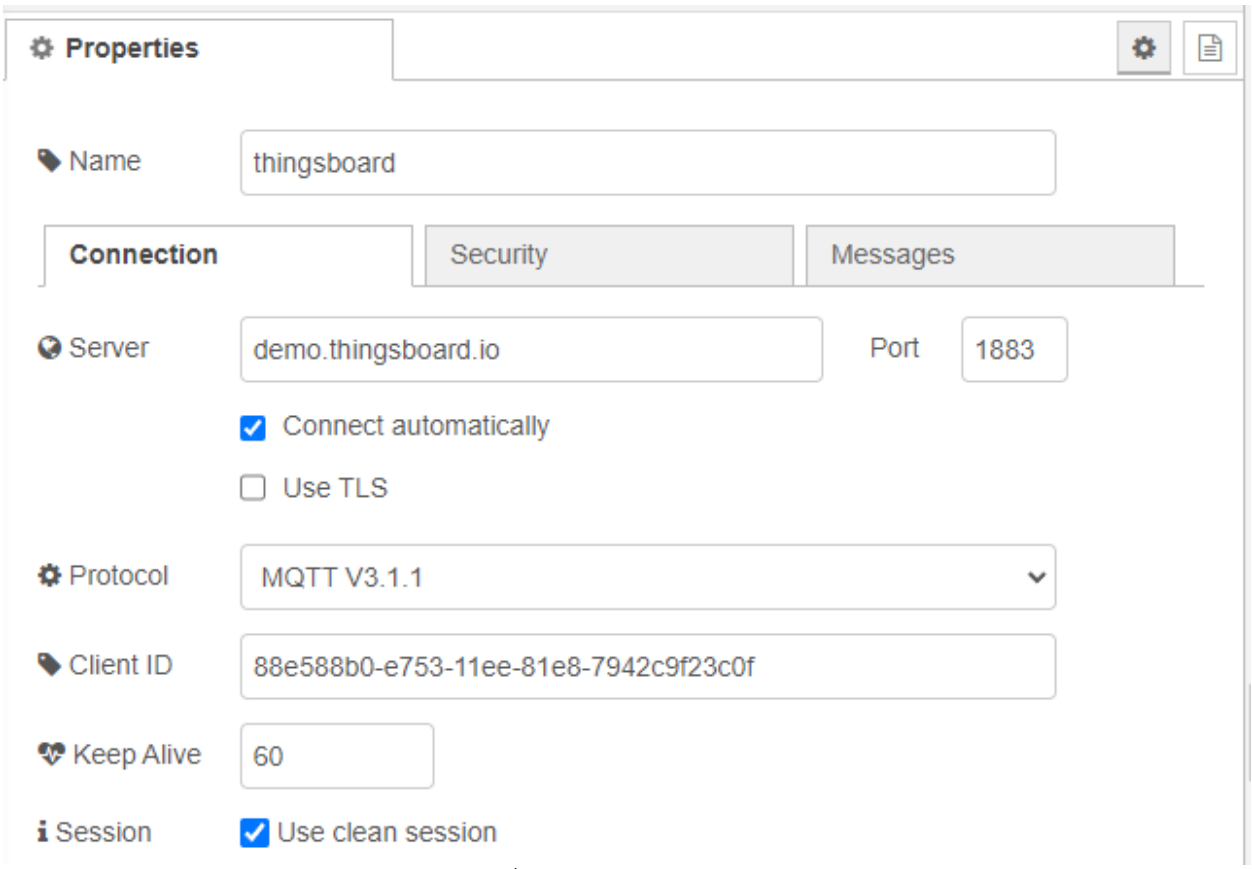

ภาพที่ 21 Communication protocol node (HTTP & MQTT)

3.8. การตั้งค่าโหนด MQTT และ HTTP ให้เลือก Topic เป็น v1/devices/me/telemetry ที่นี่คือที่อยู่ของ telemetry data ใน ThingsBoard จากนั้นให้คลิกที่ ไอคอน ดินสอ ข้างๆหัวข้อ Server เพื่อทำการตั้งค่าในขั้นตอนถัดไป สามารถตั้งค่าตามภาพที่ 22

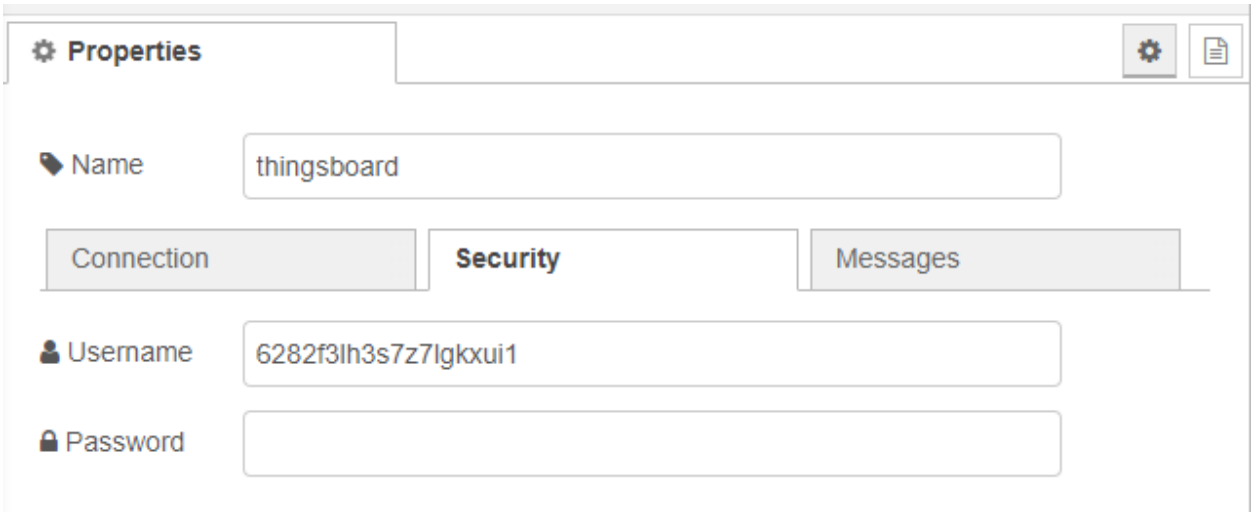



ภาพที่ 22 การตั้งค่า communication protocol node

3.9. ในขั้นตอนนี้ให้กรอก Server เป็นที่ที่อยู่ของ Server ThingsBoard ที่มี ในที่นี้ผู้พัฒนาเลือกใช้ demo.ThingsBoard.io ซึ่ง ThingsBoard เปิดให้ใช้ได้ฟรี จากนั้นให้กรอก Client ID เป็น Device ID (ภาพที่ 23 ) ซึ่งข้อมูลในส่วนนี้ จะได้ในขั้นตอนถัดไป จากนั้นเลือกหัวข้อ Security ให้กรอก access Token (ภาพที่ 24 ) ข้อมูลส่วนนี้ก็จะได้รับในขั้นตอนถัดไปเช่นกัน




ภาพที่ 23 การกรอก Device ID

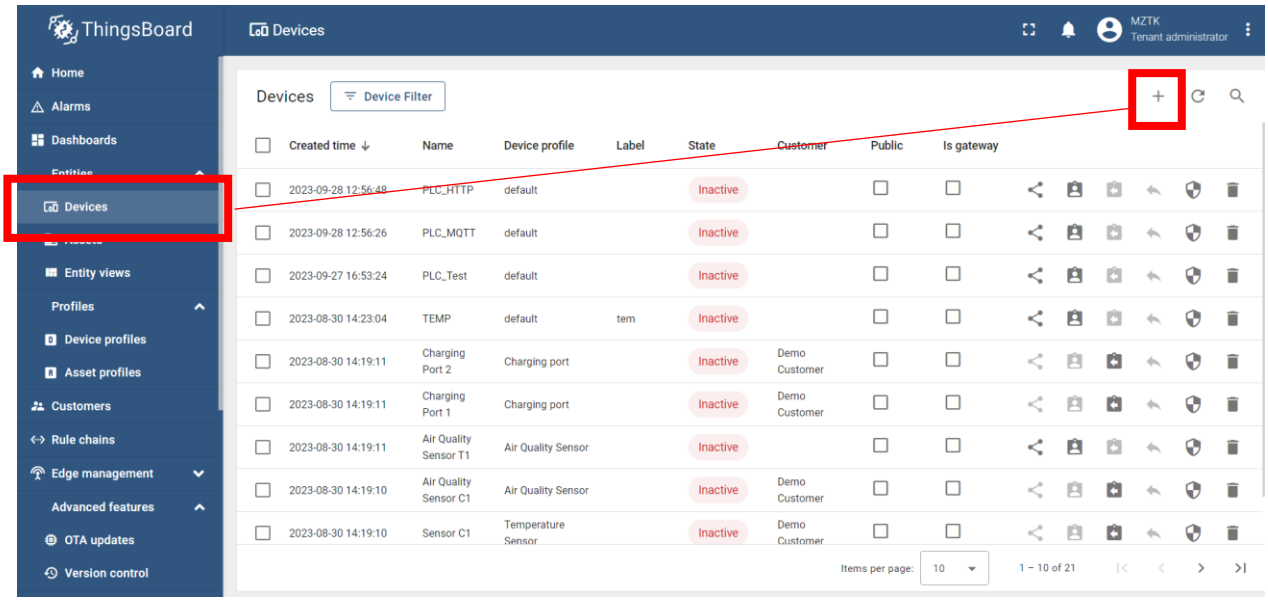



ภาพที่ 24 การกรอก Access token

### 3.2.4 การตั้งค่าแพลตฟอร์ม ThingsBoard

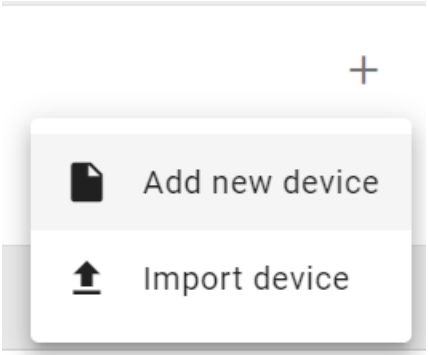
ใช้งาน แพลตฟอร์ม ThingsBoard สามารถใช้ demo.ThingsBoard.io ที่เปิดเป็นสาธารณะและฟรี

1. ให้สร้างหัวข้อ Device และกดไปที่ไอคอน เครื่องหมาย + ดังภาพที่ 25



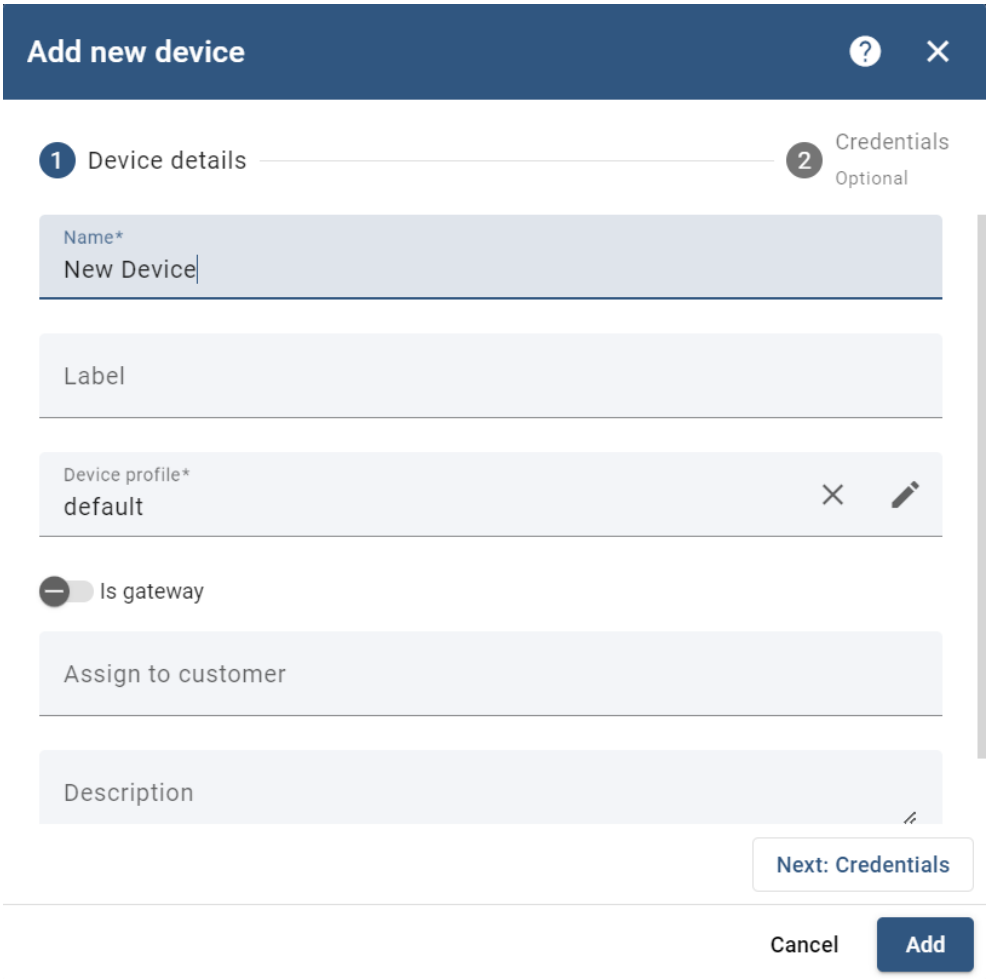

ภาพที่ 25 ปุ่มสร้าง Device

2. จากนั้นให้เลือก Add new device (ภาพที่ 26)



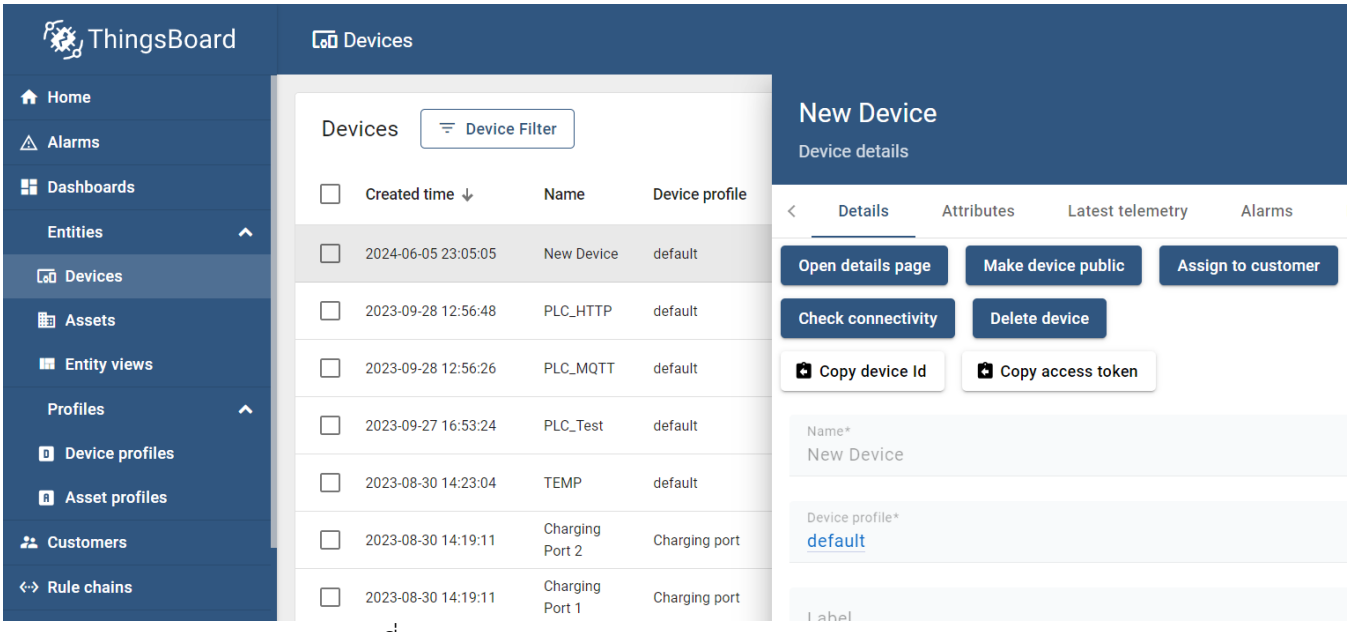

ภาพที่ 26 สร้าง Device

3. คลิก Add เพื่อยืนยันการสร้าง Device (ภาพที่ 27)




ภาพที่ 27 กดยืนยันการสร้าง Device

4. เลือก Device ที่เพิ่งสร้างมา และกด Copy device id และ access token (ภาพที่ 28) ไปใส่ที่ Node-RED ตามขั้นตอนก่อนหน้านี้




ภาพที่ 28 การ Copy device id และ access token

### 3.2.5 การพัฒนา Unity Script

1. เพื่อที่จะสามารถใช้งาน WebSocket protocol ภายใน Unity ได้นั้น จำเป็นต้องมี Library ก่อน โดยสามารถประกาศเรียกใช้ Library นั้นๆ ได้โดยการเรียกใช้ภายในสคริปต์ได้เลย โดยการใช้คำสั่งที่ ภาพที่ 29

```
using WebSocketSharp;
```

ภาพที่ 29 คำสั่งเรียกใช้ Library ใน Unity

2. จากนั้นจะทำให้สคริปต์เชื่อมต่อกับเว็บไซต์ที่ต้องการ โดยจะต้องมี access token ของเว็บนั้น ๆ ด้วย ซึ่งจะใช้ access token เดียวกับ token ที่ใส่ใน Node-RED

```
string connectionUrl = $"{serverUrl}?token={accessToken}";
```

ภาพที่ 30 เรียกใช้ตัวแปร access token

3. เลือกที่อยู่ข้อมูลที่ต้องการติดตาม (ภาพที่ 31) โดยก่อนที่จะเข้าถึงข้อมูลจะเปิดใช้การจับเวลาก่อน

```
stopwatch.Restart();
string subscriptionMessage = $"{{\"tsSubCmds\":[{{\"entityType\":\"DEVICE\",\"entityId\":\"{deviceId}\",\"keys\":\"temperature,humidity\"}}],\"historyCmds\":[],\"attrSubCmds\":[]}}";
ws.Send(subscriptionMessage);
```

ภาพที่ 31 ที่อยู่ของข้อมูลที่ต้องการติดตาม

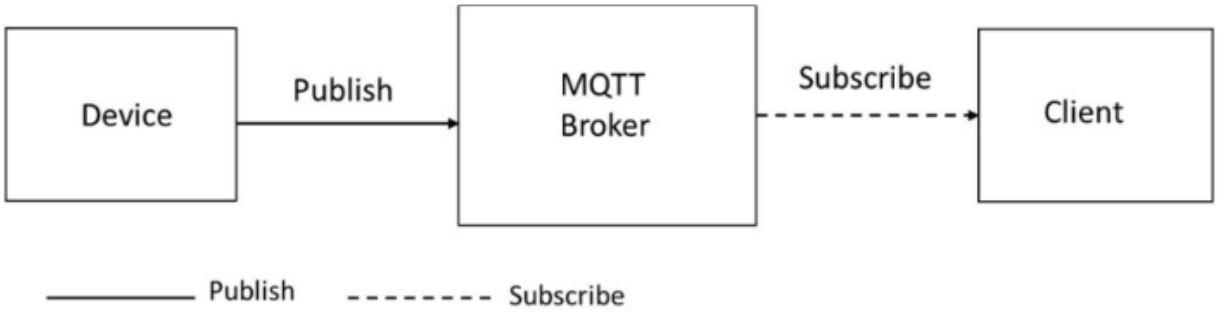
4. เมื่อได้รับข้อมูลแล้ว ให้หยุดฟังก์ชันจับเวลา (ภาพที่ 32) ทำการแสดงข้อมูลที่ console ของ Unity

```
stopwatch.Stop();  // Stop the stopwatch
UnityEngine.Debug.Log("Time taken: " + stopwatch.Elapsed.TotalMilliseconds + " ms");
```

ภาพที่ 32 การใช้ function stopwatch

### 3.2.6 การเลือกใช้โปรโตคอลในการส่งข้อมูล

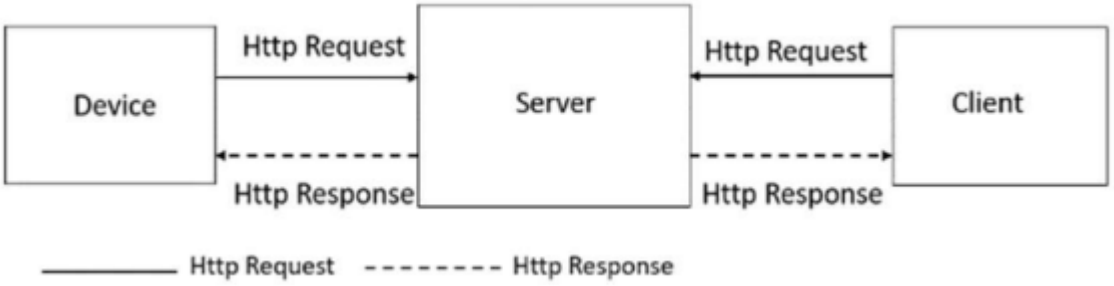
การเลือกโปรโตคอลการสื่อสารที่เหมาะสมเป็นการกระทำที่สำคัญสำหรับการส่งข้อมูลจาก PLC ไปยังแพลตฟอร์ม ThingsBoard เนื่องจากมันช่วยให้การไหลของข้อมูลเป็นไปอย่างราบรื่น ซึ่งมีความสำคัญอย่างยิ่งต่อการทำงานแบบเรียลไทม์ของการตั้งค่า Digital Twin ในการวิจัยนี้ มีการเลือกใช้โปรโตคอลสองตัวคือ MQTT และ HTTP เนื่องจากเป็นโปรโตคอลที่มีการใช้งานทั่วไปและมีคุณสมบัติที่เหมาะสมสำหรับโครงการ IoT และ Digital Twin




ภาพที่ 33 Workflow ของ MQTT

โปรโตคอล MQTT เป็นที่รู้จักในเรื่องของการมีน้ำหนักเบาและต้องการแบนด์วิธเครือข่ายน้อย ทำให้มันเหมาะสมกับสภาพแวดล้อมที่มีการเชื่อมต่อจำกัด นอกจากนี้ยังรองรับความสามารถแบบเรียลไทม์ของสภาพแวดล้อมระบบ ทำให้มั่นใจได้ว่าการส่งข้อความเป็นไปอย่างทันท่วงที ซึ่งมีความสำคัญสำหรับการตรวจสอบและควบคุมแบบเรียลไทม์ เนื่องจากเป็นโปรโตคอลที่มีน้ำหนักเบา นอกเหนือจากการใช้งานใน IoT แล้ว ยังเหมาะสำหรับการสื่อสารระหว่างเครื่องจักร (M2M) โปรโตคอลนี้ทำงานบนหลักการของการเผยแพร่/สมัครรับข้อมูล (Publish/Subscribe)

ตามที่แสดงในภาพที่ 33 อุปกรณ์ IoT สามารถเผยแพร่ข้อมูลในหัวข้อ (topic) ไปยัง MQTT Broker ที่ทำหน้าที่เป็นเซิร์ฟเวอร์ข้อมูลได้ และไคลเอนต์สามารถสมัครรับข้อมูลในหัวข้อเดียวกันและรับข้อมูลจาก MQTT Broker ได้ หากข้อมูลในหัวข้อนี้มีการเปลี่ยนแปลงหรืออัปเดตเมื่อเวลาผ่านไป ไคลเอนต์จะได้รับข้อมูลใหม่จากหัวข้อนั้นโดยอัตโนมัติ ในกรณีนี้ แพลตฟอร์ม ThingsBoard ทำหน้าที่เป็น MQTT Broker




ภาพที่ 34 workflow ของ HTTP

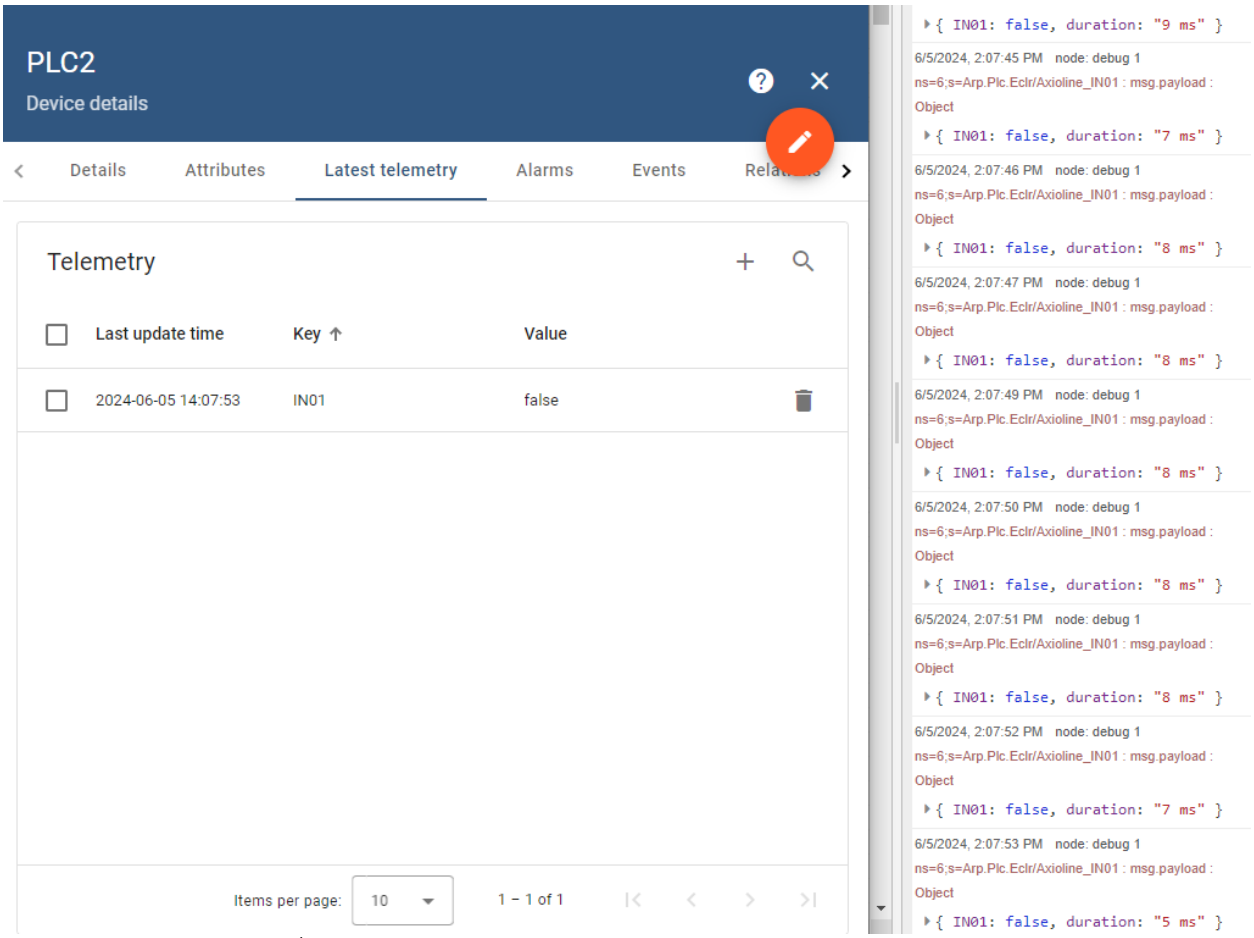
จากภาพที่ 34 แสดงการไหลของข้อมูลในโปรโตคอล HTTP ซึ่งอิงตามสถาปัตยกรรมการสื่อสารแบบไคลเอนต์-เซิร์ฟเวอร์ ไคลเอนต์จะส่งคำขอ HTTP ไปยังเซิร์ฟเวอร์ และเซิร์ฟเวอร์จะส่งคำตอบ HTTP กลับมา ข้อดีของโปรโตคอลนี้คือสามารถส่งข้อมูลปริมาณมากได้ อย่างไรก็ตาม การส่งข้อมูลจำนวนมากนี้อาจทำให้เกิดความหน่วง (latency) ในการสื่อสารมากขึ้น

## 3.3 การวัดค่า Latency

### 3.3.1 การส่งข้อมูลจาก PLC ไปยัง ThingsBoard

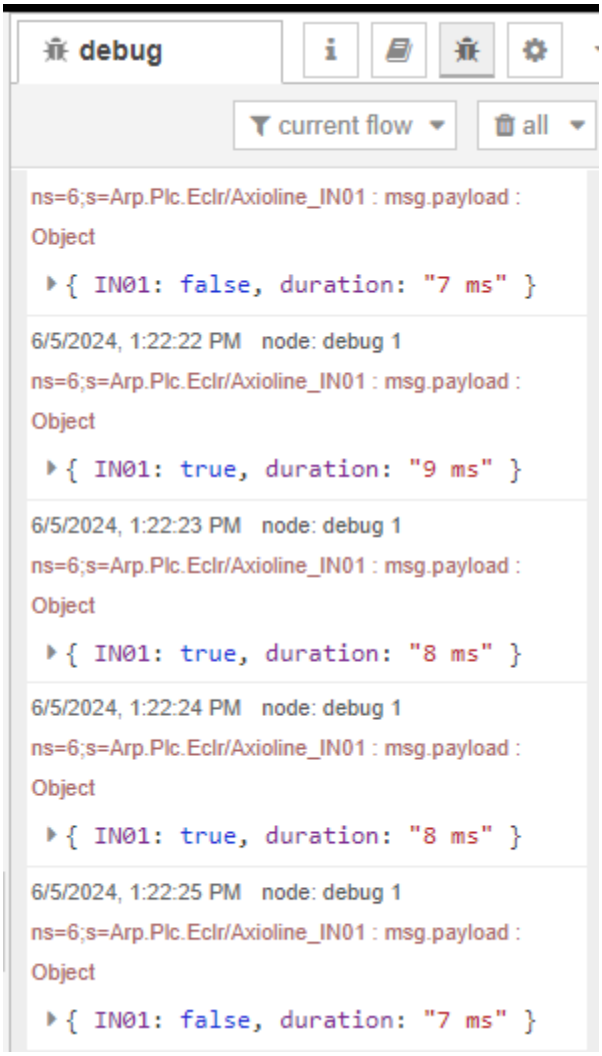
ในชุดทดสอบการส่งข้อมูลนี้ Node-RED ทำหน้าที่เป็นตัวกลางระหว่าง PLC และ ThingsBoard ช่วยในการส่งข้อมูล ในการตั้งค่านี้ Node-RED รับข้อมูลจาก PLC และส่งต่อไปยังแพลตฟอร์ม ThingsBoard โดยใช้โปรโตคอลการสื่อสารที่เลือกไว้ ไม่ว่าจะเป็นโปรโตคอล MQTT หรือโปรโตคอล HTTP

มีสองส่วนของความหน่วงในการส่งข้อมูลที่ต้องครอบคลุม ส่วนแรกคือความหน่วงในการส่งข้อมูลระหว่างอุปกรณ์ PLCnext กับแพลตฟอร์ม ThingsBoard ซึ่งวัดความหน่วงในการส่งข้อมูลด้วยโปรโตคอลต่าง ๆ จากอุปกรณ์ PLCnext ไปยังแพลตฟอร์ม ThingsBoard ความหน่วงนี้ได้จากการจับเวลา ณ จุดสองจุด เมื่อข้อมูลถูกรับที่โหนดอ่าน Node-RED และเมื่อข้อมูลถูกอัปเดตใน telemetry ของ ThingsBoard ตามที่แสดงในภาพที่ 35 ด้านล่าง ความหน่วงจะถูกคำนวณโดยการลบเวลาที่จับครั้งแรกออกจากเวลาที่จับครั้งที่สอง Flow ของ Node-RED ที่ใช้จะแสดงในภาพที่ 35 ด้านล่าง




ภาพที่ 35 การแสดง timestamp ของ ThingsBoard และ Node-RED

ภาพที่ 35 แสดงการจับเวลาของการส่งข้อมูลจากอุปกรณ์ PLCnext ไปยังแพลตฟอร์ม ThingsBoard โดยมีการจับเวลาครั้งแรก เมื่อข้อมูลถูกส่งไปถึง Node-RED และการจับเวลาครั้งที่สอง เมื่อข้อมูลถูกส่งไปถึงแพลตฟอร์ม ThingsBoard ความหน่วงถูกคำนวณโดยการลบเวลาครั้งแรกออกจากเวลาครั้งที่สอง ภาพที่ 36 แสดง log ของ Node-RED ที่ใช้ในการรับข้อมูลจากอุปกรณ์ PLCnext และส่งไปยังแพลตฟอร์ม ThingsBoard




ภาพที่ 36 log ตัวอย่างการส่งข้อมูลและแสดงระยะเวลาในส่งข้อมูลจาก Node-RED สู่ ThingsBoard

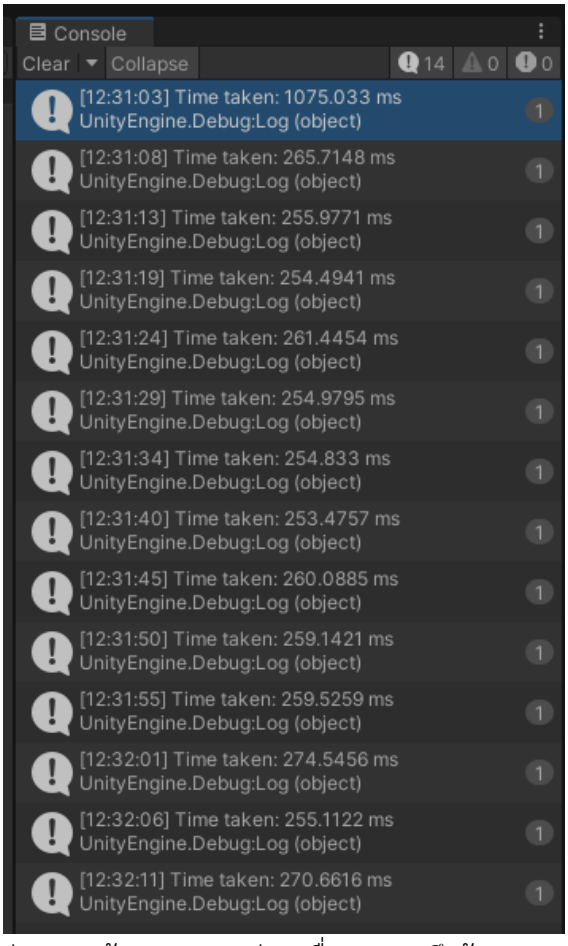
จากรูปที่ 36 เป็นตัวอย่างการส่งข้อมูลและแสดงระยะเวลาของการส่งผ่านข้อมูลไปยังแพลตฟอร์ม ThingsBoard โดยใช้โปรโตคอล MQTT โดย IN01 หรือ Axioline_IN01 ซึ่งจะแทนถึงสถานะการทำงานของอุปกรณ์หรือโรงงานอัจฉริยะ

### 3.3.2 การดึงข้อมูลจาก ThingsBoard ไปยัง Unity

ส่วนที่สองของการวัดความหน่วงในการส่งข้อมูลคือการส่งข้อมูลระหว่างแพลตฟอร์ม ThingsBoard และ Unity 3D engine สำหรับโปรโตคอล HTTP ใช้สคริปต์ Unity ในการส่งคำขอ HTTP ไปยังแพลตฟอร์ม ThingsBoard และวัดเวลาที่ใช้สำหรับแต่ละคำขอ โดยใช้คลาส Stopwatch ในสคริปต์ Unity เรียกใช้วิธี

stopwatch.Restart() ก่อนส่งคำขอ HTTP และเรียกใช้วิธี stopwatch.Stop() เมื่อได้รับข้อมูล เวลาที่ผ่านไปเป็นมิลลิวินาทีจะถูกบันทึกเพื่อตรวจสอบความหน่วง

สำหรับโปรโตคอล WebSocket ใช้สคริปต์ Unity ที่ใช้ไลบรารี WebSocketSharp เพื่อสร้างการเชื่อมต่อ WebSocket ไปยังแพลตฟอร์ม ThingsBoard โดยใช้คลาส Stopwatch วัดเวลาตั้งแต่การส่งข้อความการสมัครรับข้อมูลจนถึงเวลาที่ได้รับข้อมูล เมื่อการเชื่อมต่อถูกเปิดและส่งข้อความการสมัครรับข้อมูลไปยังแพลตฟอร์ม ThingsBoard จะเริ่มจับเวลา (stopwatch) เมื่อได้รับข้อความที่มีข้อมูลจากแพลตฟอร์ม ThingsBoard จะหยุดจับเวลา (stopwatch) เวลาที่ผ่านไป ซึ่งวัดเป็นมิลลิวินาทีจะถูกบันทึกเพื่อคำนวณความหน่วง (ภาพที่ 37) โดยใช้สคริปต์ Unity ในการส่งคำขอ HTTP และ WebSocket เพื่อวัดเวลาที่ใช้ในการดึงข้อมูล




ภาพที่ 37 ตัวอย่าง log ข้อมูลความหน่วง เมื่อ Unity ดึงข้อมูลจาก ThingsBoard

# 4 ผลการดำเนินการ

การทดลองถูกจัดเพื่อประเมินความหน่วงที่เกี่ยวข้องกับวิธีการสื่อสารและโปรโตคอลต่างๆ ในการส่งข้อมูลจากอุปกรณ์ PLCnext ไปยังแพลตฟอร์ม ThingsBoard ผลลัพธ์ของส่วนการสื่อสารนี้สามารถดูได้จากตารางที่ I หลังจากนั้น ผลลัพธ์ของส่วนการสื่อสารในการดึงข้อมูลจากแพลตฟอร์ม ThingsBoard ไปยัง Unity 3D engine สามารถดูได้จากตารางที่ II ข้อมูลถูกดึงมาจากแพลตฟอร์ม ThingsBoard จำนวน 100 ค่า โดยใช้ทั้งโปรโตคอล HTTP และโปรโตคอล WebSocket ผลลัพธ์แสดงให้เห็นถึงความแตกต่างที่ชัดเจนในความหน่วงของการส่งข้อมูลระหว่างโปรโตคอลทั้งสองนี้

## 4.1 ผลการส่งข้อมูลจากอุปกรณ์ PLCnext ไปยังแพลตฟอร์ม ThingsBoard

ตารางที่ 1 เผยให้เห็นความแตกต่างอย่างชัดเจนระหว่างโปรโตคอล MQTT และ HTTP ในด้านความเร็วในการส่งข้อมูลจากอุปกรณ์ PLCnext ไปยังแพลตฟอร์ม ThingsBoard โดย MQTT โดดเด่นด้วยเวลาเฉลี่ยในการส่งข้อมูลที่รวดเร็วเพียง 126.667 มิลลิวินาที (ms) ซึ่งเป็นการปรับปรุงที่ชัดเจนเมื่อเทียบกับค่าเฉลี่ยของ HTTP ที่ 341.977 ms ประสิทธิภาพนี้มีความสำคัญอย่างยิ่งในด้านระบบอัตโนมัติ 4.0 ที่การส่งข้อมูลทันทีมีความสำคัญมาก

ตารางที่ 1 ตารางสรุปผลการทดสอบส่งข้อมูลจาก PLCnext ไปยัง ThingsBoard

| Metric | MQTT (ms) | HTTP (ms) |
|---|---|---|
| Mean | 126.667 | 341.977 |
| St. Dev. | 47.625 | 59.111 |
| Max. | 320.789 | 634.207 |
| Min. | 90.123 | 300.358 |
| Median | 126.508 | 336.876 |

## 4.2 ผลการการดึงข้อมูลจากแพลตฟอร์ม ThingsBoard โดย Unity

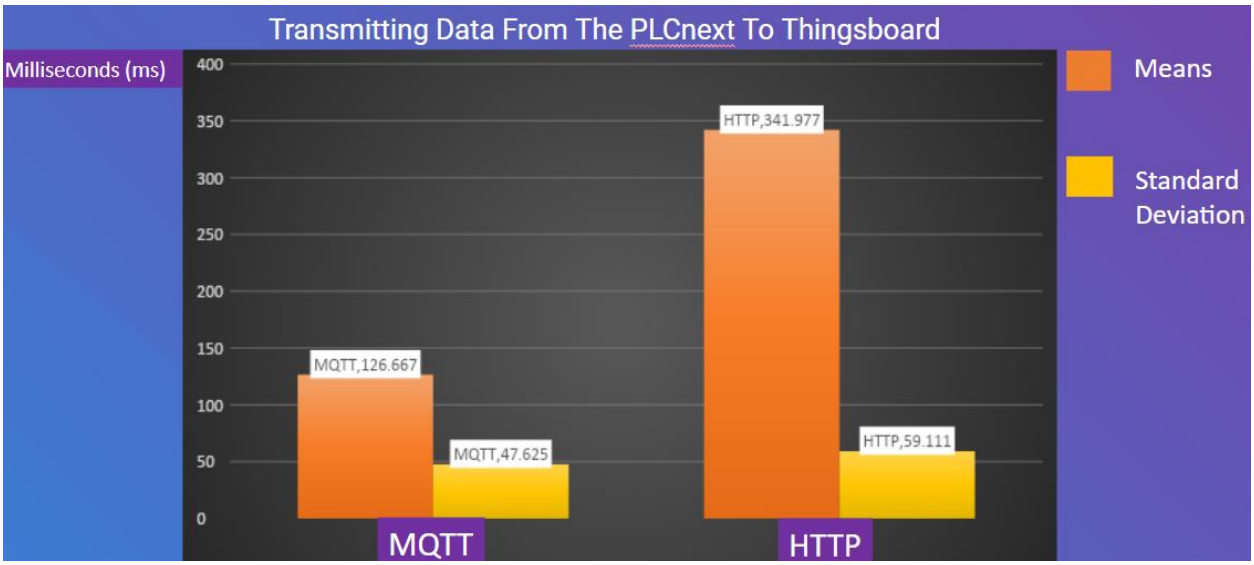
ในตารางที่ 2 มุ่งเน้นที่อัตราการดึงข้อมูลจากแพลตฟอร์ม ThingsBoard ไปยัง Unity 3D engine โดยโปรโตคอล WebSocket ทำได้ดีกว่า ด้วยเวลาเฉลี่ยในการดึงข้อมูลที่ 259.322 ms ซึ่งดีกว่า HTTP ที่มีค่าเฉลี่ย 287.882 ms นอกจากความเร็วแล้ว WebSocket ยังให้การเชื่อมต่อที่เสถียรกว่า ซึ่งเป็นปัจจัยสำคัญสำหรับประสิทธิภาพที่เชื่อถือได้ในการนำไปสู่ Digital Twin ภายในระบบอุตสาหกรรมอัตโนมัติ

ตารางที่ 2 ตารางสรุปผลการทดสอบดึงข้อมูลจาก ThingsBoard โดย Unity

| Metric | HTTP (ms) | WebSocket (ms) |
|---|---|---|
| Mean | 287.882 | 259.322 |
| St. Dev. | 78 | 70.2 |
| Max. | 1160.115 | 378.208 |
| Min. | 278.627 | 260.548 |
| Median | 288.952 | 265.006 |

## 4.3 ผลการใช้งานโปรแกรม

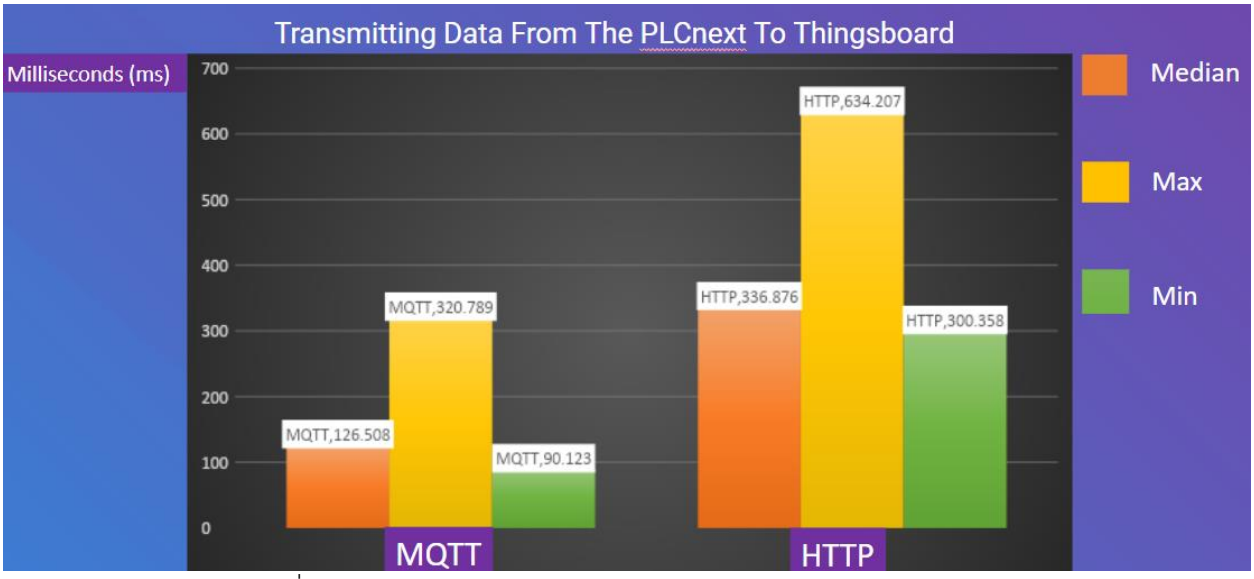
ปรากฏผลลัพธ์ที่ชัดเจนในการเปรียบเทียบประสิทธิภาพของ communication protocol จากทั้ง 2 กรณี ได้อย่างชัดเจน โดยผลลัพธ์จากการทดลองจะแสดงดังต่อไปนี้ ภาพที่ 38 ค่าเฉลี่ยและส่วนเบี่ยงเบนมาตรฐานของการส่งข้อมูลจาก PLCnext ไปยัง ThingsBoard, ภาพที่ 39 ค่ามัธยฐาน, ค่าสูงสุด และต่ำสุดของการส่งข้อมูลจาก PLCnext ไปยัง ThingsBoard




ภาพที่ 38 สรุปผลการส่งข้อมูลจาก PLCnext ไปยัง ThingsBoard

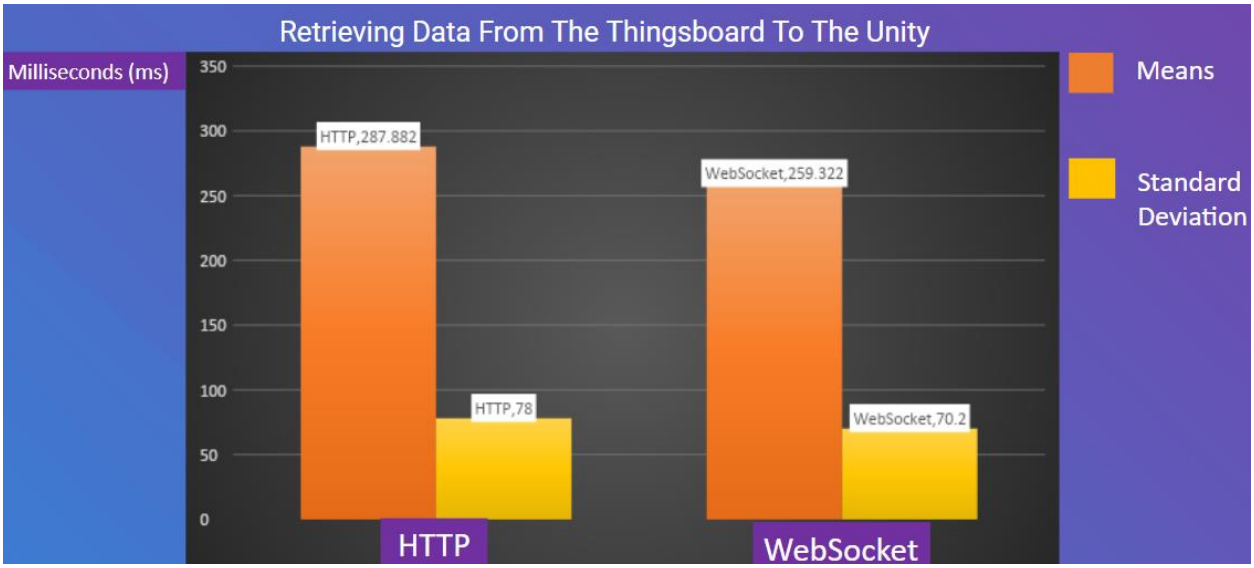
จากภาพที่ 38 ค่าความหน่วงเฉลี่ยของ MQTT เร็วกว่า HTTP ประมาณ 63% ซึ่งค่าความต่างอย่างชัดเจนนี้บ่งชี้ว่า MQTT สามารถส่งข้อมูลได้เร็วกว่า HTTP มาก ซึ่งเป็นสิ่งสำคัญสำหรับแอปพลิเคชันแบบเรียลไทม์ที่จำเป็นต้องลดความล่าช้าให้เหลือน้อยที่สุด

ค่าเบี่ยงเบนมาตรฐานของ MQTT ต่ำกว่า HTTP ประมาณ 19% ซึ่งบ่งชี้ถึงประสิทธิภาพที่สม่ำเสมอมากขึ้น ค่าเบี่ยงเบนมาตรฐานที่ต่ำกว่าหมายความว่าเวลาที่ใช้ในการส่งข้อมูลโดยใช้ MQTT นั้นคาดเดาได้มากกว่า ซึ่งเป็นประโยชน์สำหรับแอปพลิเคชันที่ต้องการการไหลผ่านของข้อมูลที่คงที่และเชื่อถือได้



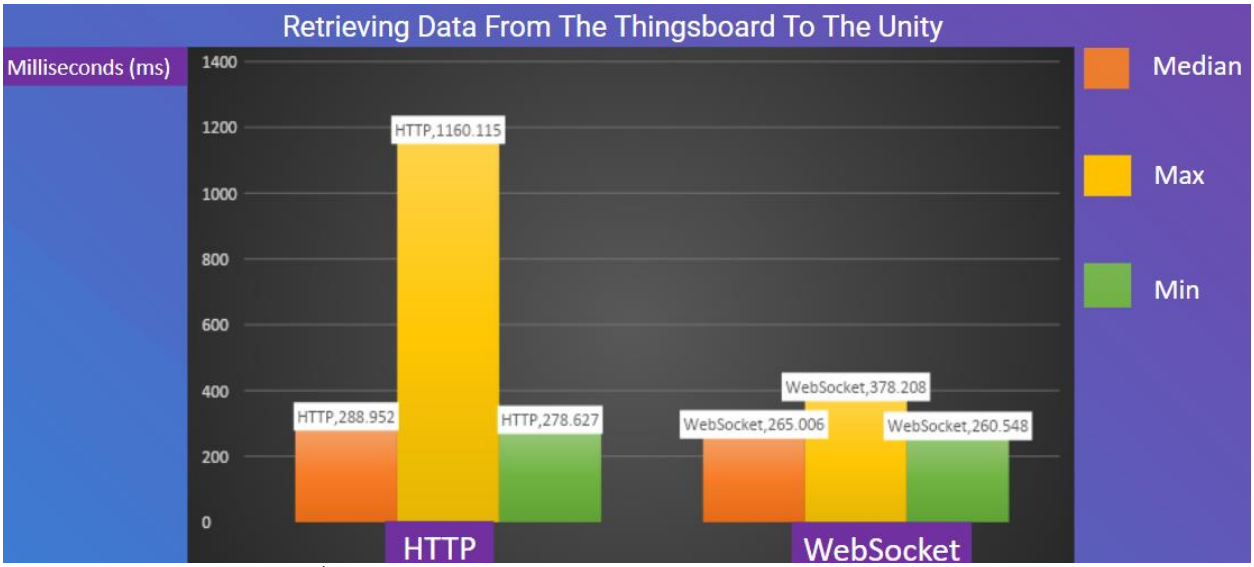

ภาพที่ 39 สรุปผลการส่งข้อมูลจาก PLCnext ไปยัง ThingsBoard (ต่อ)

จากภาพที่ 39 ค่าหน่วงเวลาเฉลี่ยของ MQTT ต่ำกว่า HTTP ประมาณ 62% ซึ่งแสดงถึงข้อเท็จจริงที่ว่า MQTT ให้เวลาในการส่งข้อมูลที่เร็วกว่าอย่างสม่ำเสมอ ค่าหน่วงเวลาสูงสุดของ MQTT ต่ำกว่า HTTP ประมาณ 49% ซึ่งถือว่ามีความสำคัญอย่างมาก เนื่องจากแสดงให้เห็นว่าแม้ในสถานการณ์ที่เลวร้ายที่สุด MQTT ก็ทำงานได้ดีกว่า HTTP ซึ่งช่วยลดความเสี่ยงของความล่าช้าที่สำคัญได้ ค่าหน่วงเวลาขั้นต่ำของ MQTT ต่ำกว่า HTTP ประมาณ 70% ซึ่งหมายความว่าเวลาในการส่งข้อมูลในกรณีที่ดีที่สุดจะเร็วกว่าเมื่อใช้ MQTT




ภาพที่ 40 สรุปผลการดึงข้อมูลจาก Unity จาก ThingsBoard

จากภาพที่ 40 ค่าความหน่วงเฉลี่ยของ WebSocket ต่ำกว่า HTTP ประมาณ 9.92% ซึ่งแสดงให้เห็นว่า WebSocket สามารถดึงข้อมูลได้เร็วกว่า HTTP โดยเฉลี่ย ซึ่งมีความสำคัญต่อการทำงานแบบเรียลไทม์ที่ต้องการลดความล่าช้าให้เหลือน้อยที่สุด ค่าเบี่ยงเบนมาตรฐานของ WebSocket ต่ำกว่า HTTP ประมาณ 10% ซึ่งบ่งชี้ถึงประสิทธิภาพที่สม่ำเสมอมากขึ้น ค่าเบี่ยงเบนมาตรฐานที่ต่ำกว่าหมายความว่าเวลาที่ใช้ในการดึงข้อมูลโดยใช้ WebSocket นั้นคาดเดาได้มากกว่า ซึ่งเป็นประโยชน์สำหรับแอปพลิเคชันที่ต้องการการไหลผ่านของข้อมูลที่คงที่และเชื่อถือได้




ภาพที่ 41 สรุปผลการดึงข้อมูลจาก Unity จาก ThingsBoard (ต่อ)

จากภาพที่ 41 ค่ามัธยฐานของ WebSocket ต่ำกว่า HTTP ประมาณ 8.3% ซึ่งบ่งชี้ว่า WebSocket สามารถดึงข้อมูลได้เร็วกว่า HTTP ในกรณีทั่วไป ค่านี้มีความสำคัญเนื่องจากเป็นตัวแทนของค่ากลางในชุดข้อมูล ค่าสูงสุดของ WebSocket ต่ำกว่า HTTP ประมาณ 67.4% ซึ่งหมายความว่าในสถานการณ์ที่แย่ที่สุด WebSocket ยังสามารถดึงข้อมูลได้เร็วกว่า HTTP อย่างมาก ซึ่งลดความเสี่ยงจากความหน่วงเวลาที่สูง ค่าต่ำสุดของ WebSocket ต่ำกว่า HTTP ประมาณ 6.5% ซึ่งแสดงให้เห็นว่าในสถานการณ์ที่ดีที่สุด WebSocket สามารถดึงข้อมูลได้เร็วกว่า HTTP เล็กน้อย

## 4.4 การอภิปรายผล

จากการทดลองวัดความเร็วนการรับส่งข้อมูลของโปรโตคอลชนิดต่าง ๆ จะเห็นได้ว่า โปรโตคอล MQTT สามารถรับส่งข้อมูลได้โดยใช้เวลาเฉลี่ยน้อยที่สุด นั่นหมายความว่า มีการหน่วงเวลาในการส่งข้อมูลน้อย สามารถส่งข้อมูลชนิดในเวลาจริงได้ดี และการสื่อสารระหว่างแพลตฟอร์มกับ Unity พบว่า โปรโตคอล WebSocket สามารถทำงานได้ดีที่สุด

แต่จากการรับส่งข้อมูลทั่วไป การใช้งานโปรโตคอล OPC UA จากเครื่อง PLCnext ไปยังแพลตฟอร์มข้อมูลโดยตรงจะมีความคล่องตัวและมีประสิทธิภาพสูงกว่า แต่การบริหารจัดการโปรโตคอลนี้ จำเป็นต้องใช้ค่าลิขสิทธิ์เฉพาะของแพลตฟอร์ม จึงยังไม่ได้มีการวัดประสิทธิภาพของการรับส่งข้อมูลด้วย OPC UA โดยตรง แต่จะมีการดำเนินการในอนาคต ซึ่งคาดว่า จะมีความสะดวกและเข้ากับเทคโนโลยีของโรงงานอุตสาหกรรมได้ดี

# 5 สรุปผล

## 5.1 สรุปผลการดำเนินงาน

สรุปผลการดำเนินงานของโครงการผลปรากฏว่า โครงงานสำเร็จลุล่วงเป็นไปตามวัตถุประสงค์ที่ตั้งไว้ ดังนี้

1. สามารถศึกษาความเป็นไปได้ในการออกแบบและพัฒนาเทคโนโลยีคู่แฝดดิจิทัลบนพื้นฐานของเทคโนโลยี PLCnext โดยสามารถสร้างแบบจำลองดิจิทัลที่มีความแม่นยำและสามารถใช้งานได้จริง

2. ทดสอบประสิทธิภาพของการใช้งานเทคโนโลยีคู่แฝดดิจิทัลกับระบบควบคุมการผลิตแบบอัตโนมัติในโรงงานอุตสาหกรรมอัจฉริยะ (PLCnext technology) โดยพบว่าโปรโตคอล MQTT มีความหน่วงเวลาในการส่งข้อมูลต่ำกว่า HTTP ถึง 63% และ WebSocket มีความหน่วงเวลาในการดึงข้อมูลต่ำกว่า HTTP ถึง 9.92% ซึ่งทำให้การสื่อสารข้อมูลเป็นไปอย่างรวดเร็วและมีประสิทธิภาพมากขึ้น

## 5.2 ปัญหา อุปสรรค และข้อจำกัด

1. การจำลองกระบวนการผลิตยังคงมีข้อจำกัดในเรื่องของความสมจริง เนื่องจากอ้างอิงจากแหล่งข้อมูลที่มีอยู่เท่านั้น

2. การพัฒนาและทดสอบเทคโนโลยีคู่แฝดดิจิทัลต้องใช้ทรัพยากรที่มีสมรรถนะสูง เช่น CPU, RAM, และ Graphic Card ซึ่งในบางครั้งอุปกรณ์ที่ใช้อาจไม่เพียงพอ

3. กระบวนการในการทดสอบ อ้างอิงจากอินเตอร์เน็ตของมหาวิทยาลัยในการทดสอบ

## 5.3 ข้อเสนอแนะ

1. ทดสอบ communication protocol หลาย ๆ ตัว

2. เพิ่มอุปกรณ์หรือทดสอบกับอุปกรณ์โรงงานที่ใช้จริง

3. ทดสอบกับเกมเอนจินตัวอื่นเพิ่มเติม เช่น Unreal Engine

## 5.4 แนวทางในการพัฒนาในอนาคต

1. นำเทคโนโลยีคู่แฝดดิจิทัลมาใช้ในการทดสอบโปรโตคอลการสื่อสารเพิ่มเติม เช่น AMQP, CoAP เพื่อเปรียบเทียบประสิทธิภาพในการส่งและดึงข้อมูลในบริบทที่แตกต่างกัน

2. พัฒนาระบบให้รองรับการส่งข้อมูลขนาดใหญ่และความถี่สูง เพื่อให้สามารถใช้ในงานที่ต้องการการประมวลผลข้อมูลแบบเรียลไทม์มากขึ้น เช่น การตรวจสอบและควบคุมกระบวนการผลิตแบบเรียลไทม์ในโรงงานอุตสาหกรรม

3. เพิ่มความสามารถในการวิเคราะห์ข้อมูลเชิงลึก เช่น การใช้ Machine Learning และ AI ในการทำนายและปรับปรุงกระบวนการผลิต โดยใช้ข้อมูลจากระบบคู่แฝดดิจิทัล

## 5.5 ผลงานวิจัยที่ได้รับการตีพิมพ์และเผยแพร่แล้ว

ผลงานนี้ได้รับการตีพิมพ์และเผยแพร่ที่ The 7th International Conference on Information Technology (InCIT2023) (https://ieeexplore.ieee.org/document/10413006/ )

The 7th International Conference on Information Technology (InCIT2023)

# Digital Twin in Automation Industry: Optimal Communication Protocol

Chalermpan Fongsamut
*Faculty of Engineering*
*Burapha University*
Chon Buri, Thailand
chalerm@eng.buu.ac.th

Tanat Kanangnanon
*Faculty of Informatics*
*Burapha University*
Chon Buri, Thailand
66910081@go.buu.ac.th

Benchaporn Jantarakongkul
*Faculty of Informatics*
*Burapha University*
Chon Buri, Thailand
jbenchaporn@gmail.com

Prajaks Jitngernmadan
*Faculty of Informatics*
*Burapha University*
Chon Buri, Thailand
prajaks@buu.ac.th

2023 7th International Conference on Information Technology (InCIT) | 979-8-3503-5869-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/InCIT60207.2023.10413006

***Abstract*— Digital Twin Technology (DTT) creates a digital copy of a physical object or system, allowing interaction and control between physical and digital counterparts. Effective communication between the digital and physical entities is crucial for real-time interactions. This study explored the performance of three common communication protocols, MQTT, HTTP, and WebSocket, in a Digital Twin setup involving a Programmable Logic Controller (PLC) included PLCnext technology, an IoT platform called Thingsboard, and a 3D Game Engine called Unity. The focus is on measuring the delay or latency in data transmission and retrieval using those mentioned protocols. The data transmission latency was measured during data uploading to and data retrieval from the IoT platform. The results show that the MQTT protocol is faster than the HTTP protocol in sending data to Thingsboard platform due to its simpler and lighter design. Additionally, the WebSocket protocol shows a slight advantage over HTTP in retrieving data from the Thingsboard platform to Unity, which is important for real-time interactions in the Digital Twin setup. The findings from this study serve as crucial information for choosing the right communication protocols in Digital Twin setups for the Industry 4.0 to improve real-time interactions in automation industry.**

***Keywords—Digital Twin, IoT, MQTT, HTTP, WebSocket, Unity***

### I. INTRODUCTION

Digital Twin Technology (DTT) is a crucial concept in modern industrial and automation fields, allowing for the creation of a digital copy of physical systems or objects. This concept brings the physical objects to the digital world as a whole, including the connection of the physical and virtual 3D models of the objects. In other words, the DTT serves as a dynamic model that reflects a physical asset or system, adjusting to changes based on real-time data, and helps in forecasting the future behavior of the physical counterpart [1]. The replica of a physical system in the digital world reveals various possibilities in Industry 4.0 and Automation 4.0 area. This technology is vital as it enables real-time monitoring, simulation, and control, thus improving decision-making and predictive analytics. Some of the worthful examples to be mentioned are e.g. real-time monitoring of construction projects and predictive maintenance are noted as important applications of DTT [2], simulation and preventive maintenance of a production plant in an manufacturing industry, real-time monitoring in a production process, etc.

Nowadays, the importance of DTT in boosting operational efficiency is well-recognized; however, the communication framework supporting this technology, including sensors, communication networks, and 3D models, is equally important. Effective communication protocols ensure a smooth flow of data between the physical and digital entities, making the choice of communication protocol critical for the successful deployment and operation of the Digital Twin framework. Thus, this research aimed to explore this aspect.

The main goal of this research is to determine the most efficient communication protocols for real-time data transmission and retrieval in a digital twin framework that uses a Game Engine called Unity. The focus was on assessing the delay or latency associated with different communication protocols when transmitting data from a Programmable Logic Controller (PLC) to an IoT data platform named Thingsboard, and then to the Unity engine. This evaluation went beyond a mere academic exercise and sought to understand how these communication protocols perform in real-world scenarios, aiming to suggest a more effective pathway for real-time data transmission and interaction within the digital twin framework in Industry 4.0 and Automation 4.0 setups.

The system architecture being studied consists of three main components: a PLC with the PLCnext AXC F 2152, the Thingsboard data platform, and the Unity 3D engine. The PLC in this case is the PLCnext Technology, created by Phoenix Contact company, representing a new, open system that offers a flexible and efficient platform for automation projects. The PLCnext AXC F 2152 controller, with its advanced features, plays a crucial role in gathering and transmitting data from a production plant or smart factory to the data platform called Thingsboard using selected communication protocols. This setup represents a modern approach in industrial automation technology, where real-time data from the PLCs is acquired and sent to the data platform Thingsboard, which acts as a central data hub of the whole system. Then, the Game Engine called Unity retrieves this data from the Thingsboard platform through selected communication protocols, enabling real-time interaction and visualization of the virtual 3D models of the digital replica within the Unity editor. A diagram showing the data flow from the PLC to Thingsboard and then to Unity is provided below in Figure 1 to give a visual understanding of the setup.


979-8-3503-5869-8/23/$31.00 ©2023 IEEE

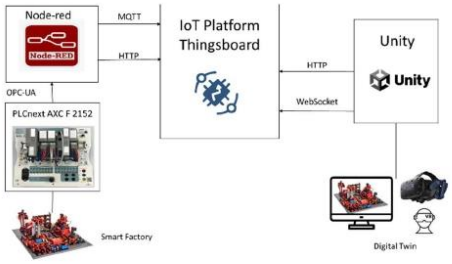



Fig. 1. Data flow from the PLC to Unity

From Fig. 1, various sensors on a smart factory or a manufacturing plant will be acquired by the PLC, in this case the PLCnext technology will be used. This data will be sent to the IoT data platform called Thingsboard that collects and stores data for the entire system. In the case of implementing the Digital Twin technology, the data will be then acquired from the Thingsboard platform by the Unity 3D engine via a communication protocol. This data will be then used to construct and control the 3D models of the system's digital replica and can be used either in the 3D First Person environment or Virtual Reality (VR) environment.

## II. STATE OF THE ART

### A. *Protocols Widely Used in Digital Twin Technology with PLCnext*

The communication between digital replicas and their real-world counterparts is made possible through different protocols, each having its own set of features suited for various needs. In the world of Digital Twin Technology (DTT) when paired with PLCnext controllers, some protocols have become popular due to their ability to transmit data in real-time reliably. Among these, MQTT (Message Queuing Telemetry Transport) is preferred for being lightweight and having an efficient way of sending/receiving messages, making it a good choice for situations needing low bandwidth and quick updates. Another in use protocol is the HTTP (HyperText Transfer Protocol) that is appreciated for its reliability and ease of use, especially in web-based interactions between digital twins and platforms like Thingsboard. The next popular communication protocol is the OPC UA (Open Platform Communications Unified Architecture). The OPC UA is known for working well across different platforms and having strong security features, making it one of the solid choices for industrial automation projects using PLCnext controllers. In case of acquiring production data from the production plants, one needs to have an equipment that enables this function. That equipment is a PLC (Programmable Logic Controller). In this case, the PLCnext is used. The PLCnext technology, created by Phoenix Contact company, offers an open environment that supports many communication protocols, improving the interaction and real-time data sharing between physical systems and their digital replicas, and therefore, suitable for digital twin technology. The adaptability of PLCnext technology allows for easy integration with various communication protocols, expanding the effectiveness of Digital Twin Technology in monitoring, controlling, and analyzing industrial processes.

### B. *Definition of Digital Twin Technology*

In the manufacturing sector, the concept of DTT was first introduced by Michael Grieves, the University of Michigan, in his presentation about Product Lifecycle Management (PLM) in 2002 [3]. The author presented a conceptual model for a virtual, digital representation equivalent of a physical product that includes the real space, virtual space, and the data and information flow between these two spaces.

As time went on, more details of the DTT were provided, explaining the DTT as a mixture of the real and virtual worlds that reflect each other to check how the real-world part is doing throughout its life [4]. This gave a peek into what the DTT is about and what is involved in it. Then, it was discussed that the DT holds all the important physical and working data of a product or system [5]. This highlighted how data moves and how control between the real and virtual models can be happened.

The story of the DTT kept evolving, and it was described as a bunch of virtual information that shows a real or possible physical product in detail, from its smallest parts to its overall shape. The importance of comparing a Digital Twin to its actual design to make the final product better by fixing any gaps between planning and doing was emphasized.

Later on, the DTT was called a living model that represents a real thing in the real world or the real system, which keeps changing based on new data and can predict what the real thing will do next [6, 7]. These definitions pointed out the lively nature of the DTT and its ability to get better over time with more information from the real things, focusing on watching it in real-time and predicting maintenance, which is quite useful in some works such as construction work, manufacturing work, etc.

Building on these ideas, this study defines Digital Twin Technology as a virtual replica of a real thing, made possible by sensors, communication networks, data, and 3D models. These tools allow for real-time updates and two-way communication and coordination, making sure that the virtual models stay a true copy of the real system in the real world. This definition wraps up the technology, features, and purpose of the DTT.

## III. METHODOLOGY

The goal of this research is to examine the time delay in data transmission involved when using different communication methods and protocols in a Digital Twin setup of a Smart Factory in industrial Automation 4.0. This setup includes a Programmable Logic Controller (PLCnext AXC F 2152), Node-RED as a go-between of a PLCnext and a data platform, Thingsboard as an IoT data platform, and the Unity game engine as a 3D visualizer. The system works as follows. First, the PLC sends data to the Node-RED, which then forwards the data to the Thingsboard using either the MQTT or the HTTP protocol. After that, Unity obtains this data from the Thingsboard using either the WebSocket or the HTTP protocol. The following sections explain the setup and the reasons for choosing these communication methods.

### A. *Data Transmission from PLC to Thingsboard*

Selecting an appropriate communication protocol is a pivotal action for transmitting data from the PLC to the Thingsboard platform, as it ensures a smooth data flow crucial for the real-time operation of the Digital Twin setup. For this

research, two protocols, namely the MQTT and the HTTP protocol, are chosen due to their common usage and features conducive for IoT and Digital Twin projects.

The MQTT protocol is known for its lightweight nature. It also requires less network bandwidth that makes it suitable for environments with limited connectivity. It supports real-time capabilities of a system environment, ensuring timely message delivery, which is crucial for real-time monitoring and control. Since it is a lightweight protocol apart from the IoT, it is also suitable for M2M (machine to machine) communication. This protocol works basically on the principle of Publish/Subscribe couple as shown in Figure 2 [8].

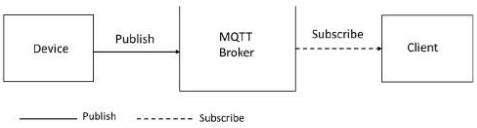



Fig. 2. Data flow of the MQTT protocol.

From Fig. 2, an IoT device can publish data in a topic to the MQTT Broker that acts as a data server. A client can subscribe to the same topic and then obtain the data from the MQTT broker. If, over time, the data of this topic has been changed or updated, the client will then get the new data from that topic automatically. In this case, the Thingsboard platform acts as an MQTT Broker.

On the other hand, the HTTP protocol is known for its robustness, being a well-established, reliable protocol for data communication. It also supports the SSL/TLS techniques for secure communications. It functions based on the well-known request/response architecture as shown in Figure 3. The HTTP protocol can transfer a large amount of data in tiny packets, which can possibly cause more bandwidth usage compared to the MQTT protocol. This behavior could potentially lead to higher latency of the data transmission.

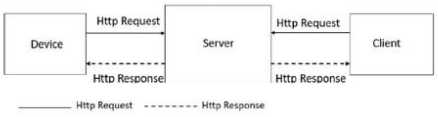



Fig. 3. Data flow of the HTTP protocol.

Fig. 3 depicts the HTTP protocol data flow that is based on the Client-Server communication architecture. The client will send the HTTP request to the server and the server will send the HTTP response as the answer. The advantage of this protocol is that one can send a large amount of data. However, this will cause more latency in communication.

In this data transmission setup, the Node-RED acts as a bridge between the PLC and the Thingsboard, aiding in data transmission. It is a programming tool for connecting hardware devices, APIs, and online services together using flow diagrams. In this setup, Node-RED receives data from the PLC and forwards it to the Thingsboard platform using the selected communication protocol, either the MQTT protocol or the HTTP protocol.

### *B. Retrieval Data via Unity 3D Engine*

When it comes to obtaining data into Unity 3D engine from the Thingsboard platform, selecting the most appropriate protocol is the key. The protocol chosen will impact how quickly and efficiently data can be accessed and used within the Unity 3D engine environment, crucial for real-time interaction in the Digital Twin setup. In this research, two protocols, namely the WebSocket protocol and the HTTP protocol, are being examined as they are both well-supported and can work well with Unity.

The WebSocket protocol is advantageous as it allows for Full-Duplex Communication, meaning it can handle two-way communication simultaneously, which helps in reducing latency. It keeps a Persistent Connection open, cutting down the time it takes to establish connections. However, it is a bit trickier to set up compared to the HTTP protocol. Furthermore, only some of the web environments support the WebSocket protocol, which can be a limitation to some applications.

The HTTP protocol stands out for its simplicity, being easy to set up and widely understood. It is also well-supported across many platforms and environments. However, it tends to have higher latency since each request needs a new connection, which can slow things down. Furthermore, it has more data overhead per request compared to the WebSocket protocol, which could be a concern to some applications.

The Unity 3D engine, a potent and flexible game development platform, is harnessed to craft a digital twin representation. In this setup, the Unity 3D engine will provide the 3D models and virtual objects, as well as the possibility to control and to interact with them according to the data received. The data retrieved from the Thingsboard platform is utilized within the Unity's editor to mirror the real-time status of the PLC, thereby actualizing the digital twin paradigm. A similar approach has been employed in a construction sector to assess collision hazards in real-time, where Unity 3D served as the platform for developing a digital twin of a construction site, updated by real-time data from sensors and enriched by artificial intelligence to forecast dangerous scenarios and avoid fatal accidents. The development of the digital twin in that scenario required modeling the environment, including physical space, sensors, and mechanical physics, all of which were handled efficiently within the Unity 3D environment [9].

### *C. Latency Measurement*

Two parts of the data transmission latency are to be covered. The first one is the data transmission latency between the PLCnext device and the Thingsboard platform. This is to measure the latency of the data transmission with different protocols from the PLCnext device to the Thingsboard platform. The latency is obtained by capturing timestamps at two points: when data is received at the *Node-RED read node*, and when the data is updated in *Thingsboard's telemetry* as shown in Figure 4 below. The latency is then calculated by subtracting the first timestamp from the second timestamp. The Node-RED flow that has been used is shown in Figure 5 below.

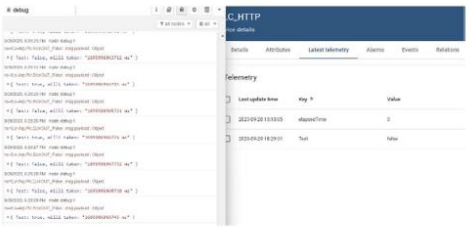


Fig. 4. Node-RED timestamp and Thingsboard's telemetry.

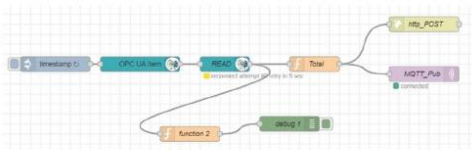


Fig. 5. Node-RED flow used for sending data and getting timestamps.

Fig. 4 shows the timestamps of the data being transmitted from the PLCnext device to the Thingsboard platform. This was seen as a first timestamp. The timestamps were then captured again at the Thingsboard platform where the transmitted data arrived. This was seen as a second timestamp. The latency is calculated by subtraction of the first and second timestamps. Fig. 5 shows the Node-RED flow used for acquiring the data from the PLCnext device and then send to the Thingsboard platform.

The second part of the data transmission latency measurement is the data transmission between the Thingsboard platform and the Unity 3D engine. To do this for the HTTP protocol, a Unity script is utilized to send HTTP requests to the Thingsboard platform and measure the time taken for each request. This was done using the Stopwatch class in the Unity script. The *stopwatch.Restart()* method is called before sending the HTTP request, and *stopwatch.Stop()* is called once the data is received. The elapsed time, in milliseconds, is logged to measure the latency.

For the WebSocket protocol, a Unity script, utilizing the *WebSocketSharp* library, establishes a WebSocket connection to the Thingsboard platform. The *Stopwatch* class is again used to measure the time it takes from the moment a subscription message is sent to the moment data is received. When the connection is opened and the subscription message is sent, the stopwatch is started. When a message containing data is received from the Thingsboard platform, the stopwatch is stopped. The time that has passed, measured in milliseconds, is then logged to calculate the latency.

These mentioned methods provide a structured approach to measure and to compare the latency associated with the HTTP protocol and the WebSocket protocol for data retrieval from the Thingsboard platform to Unity 3D engine within the Digital Twin setup.

## IV. Results

### A. Obtained values

The experiment was orchestrated to evaluate the latency associated with different communication methods and protocols in transmitting data from the PLCnext device to the Thingsboard platform. The results of this communication part can be obtained from Table I. Subsequently, the results of the communication part of retrieving data from the Thingsboard platform to the Unity 3D engine can be obtained from Table II. The data was acquired from the Thingsboard platform with 100 values using both the HTTP protocol and the WebSocket protocol. The results describe a distinguishable difference in the data transmission latency between these two protocols.

TABLE I. TRANSMITTING DATA FROM THE PLCNEXT DEVICE TO THE THINGSBOARD PLATFORM.

| Metric | MQTT (ms) | HTTP (ms) |
|---|---|---|
| Mean | 126.667 | 341.977 |
| St. Dev. | 47.625 | 59.111 |
| Max. | 320.789 | 634.207 |
| Min. | 90.123 | 300.358 |
| Median | 126.508 | 336.876 |

TABLE II. RETRIEVING DATA FROM THE THINGSBOARD PLATFORM TO THE UNITY 3D ENGINE.

| Metric | HTTP (ms) | WebSocket (ms) |
|---|---|---|
| Mean | 287.882 | 259.322 |
| St. Dev. | 78 | 70.2 |
| Max. | 1160.115 | 378.208 |
| Min. | 278.627 | 260.548 |
| Median | 288.952 | 265.006 |

### B. Findings

Table I reveals a significant contrast between the MQTT and HTTP protocols concerning the speed of data transfer from the PLCnext device to the Thingsboard platform. MQTT stands out due to its rapid average transfer time of 126.667 milliseconds (ms), a stark improvement over HTTP's average of 341.977 ms. This efficiency is paramount in the field of Automation 4.0, where immediate data transmission is crucial.

In Table II, we focused on the rate of data retrieval from the Thingsboard platform to the Unity 3D engine. WebSocket takes the lead with an average retrieval time of 259.322 ms, outperforming HTTP's average of 287.882 ms. More than just speed, WebSocket provides a more stable connection, a key factor for the dependable performance necessary in Digital Twin applications within advanced automation systems.

The reliability of WebSocket's data retrieval performance and MQTT's faster transfer times consistently underline their importance in Automation 4.0 ecosystems. PLCs are the backbone of these intricate networks, continually transferring data that is essential for real-time decision-making and system modifications. Because of this, the effectiveness of MQTT in quickly conveying data from PLCs to platforms, together with the dependability of WebSocket in data retrieval, is not only advantageous but also necessary. The operational integrity of Digital Twin models is strengthened by this synergy, which also improves the adaptability and intelligence of Automation 4.0 systems and optimizes system-wide reactions.

The learnings from this experiment are helpful in guiding the choice of communication protocols for Digital Twin setups, with the aim to reduce latency and improve real-time interactions. This finding adds to the growing understanding of how to optimize communication protocols for Digital Twin setups, supporting more efficient real-time monitoring, control, and decision-making. Especially in Industry 4.0 and Automation 4.0 where the real-time communication is one of the keys for successful digital twin implementation.

## V. CONCLUSION AND FUTURE WORK

The objective of this research was to explore the efficiency of different communication protocols for real-time data transmission in a Digital Twin setup utilizing the Unity game engine. Through a setup involving a PLCnext AXC F 2152 device, Node-RED flow programming, Thingsboard data platform, and Unity 3D engine, the delay associated with the MQTT, the HTTP, and the WebSocket protocol was examined. The data indicated a more consistent data retrieval with the WebSocket protocol compared to the HTTP protocol in the Unity 3D engine environment. Moreover, a lower data transmission latency with the MQTT protocol from the PLCnext device to the Thingsboard platform compared to the HTTP protocol is found out. These findings provide a preliminary understanding of how the choice of communication protocols could impact the latency in data transmission within a Digital Twin setup. The methodology, including the measurement of delay using a stopwatch mechanism within Unity, offered a way to evaluate the communication protocol performance. This research serves as a steppingstone towards a deeper understanding of communication frameworks in Digital Twin setups.

In the near future, the choices of these communication protocols will be taken for a Digital Twin setup for a Smart Factory setting. The data transmission latency in real setup can be then measured and evaluated. This can be used to adjust the settings and configurations of the Digital Twin setups in the future.

## ACKNOWLEDGMENT

This research was conducted at the Digital Media and Interaction Research Laboratory (DMI), Faculty of Informatics, Burapha University, and was co-funded by the Erasmus+ programme of the European Union (project ETAT Education & Training for Automation 4.0 in Thailand – n° 610154-EPP-1-2019-1-DE-EPPKA2-CBHE-JP). We express our sincere gratitude for the grant and the support provided.

## REFERENCES

[1] Y. Lu, C. Liu, K. I. K. Wang, H. Huang, and X. Xu, "Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues," *Robotics and Computer-Integrated Manufacturing*, vol. 61, 2020, doi: 10.1016/j.rcim.2019.101837. Available: https://www.sciencedirect.com/science/article/pii/S0736584519302480. [Accessed Aug. 15, 2023].

[2] A. Madni, C. Madni, and S. Lucero, "Leveraging Digital Twin Technology in Model-Based Systems Engineering," *Systems*, vol. 7, no. 1, 2019, doi: 10.3390/systems7010007. Available: https://www.researchgate.net/publication/330749986_Leveraging_Digital_Twin_Technology_in_Model-Based_Systems_Engineering. [Accessed Aug. 15, 2023].

[3] M. Grieves and J. Vickers, "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems," in *Transdisciplinary Perspectives on Complex Systems*, 2017, ch. Chapter 4, pp. 85-113, doi:10.1007/978-3-319-38756-7_4. Available: https://www.researchgate.net/publication/306223791_Digital_Twin_Mitigating_Unpredictable_Undesirable_Emergent_Behavior_in_Complex_Systems. [Accessed Aug. 16, 2023].

[4] M. Raza, P. M. Kumar, D. V. Hung, W. Davis, H. Nguyen, and R. Trestian, "A Digital Twin Framework for Industry 4.0 Enabling Next-Gen Manufacturing," presented at the *2020 9th International Conference on Industrial Technology and Management (ICITM)*, 2020, pp. 73-77, doi: 10.1109/ICITM48982.2020.9080395. Available: https://ieeexplore.ieee.org/document/9080395. [Accessed Aug. 16, 2023].

[5] M. Deng, C. C. Menassa, and V. R. Kamat, "From BIM to digital twins: a systematic review of the evolution of intelligent building representations in the AEC-FM industry," *Journal of Information Technology in Construction*, vol. 26, pp. 58-83, 2021, doi: 10.36680/j.itcon.2021.005. Available: https://www.researchgate.net/publication/349794746_From_BIM_to_digital_twins_A_systematic_review_of_the_evolution_of_intelligent_building_representations_in_the_AEC-FM_industry. [Accessed Aug. 16, 2023].

[6] J. Moyne et al., "A Requirements Driven Digital Twin Framework: Specification and Opportunities," *IEEE Access*, vol. 8, pp. 107781-107801, 2020, doi: 10.1109/access.2020.3000437. Available: https://ieeexplore.ieee.org/document/9109299. [Accessed Aug. 17, 2023].

[7] X. Zheng, J. Lu, and D. Kiritsis, "The emergence of cognitive digital twin: vision, challenges and opportunities," *International Journal of Production Research*, vol. 60, no. 24, pp. 7610-7632, 2021, doi: 10.1080/00207543.2021.2014591. Available: https://www.researchgate.net/publication/357314650_The_emergence_of_cognitive_digital_twin_vision_challenges_and_opportunities. [Accessed Aug. 18, 2023].

[8] D. Wukkadada, K. Wankhede, R. Nambiar, and A. Nair, "Comparison with HTTP and MQTT In Internet of Things (IoT)," *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2018, pp. 249-253, doi: 10.1109/ICIRCA.2018.8597401. Available: https://ieeexplore.ieee.org/document/8597401. [Accessed Aug. 18, 2023].

[9] M. Leonardo, N. Berardo, C. Alessandro, R. Luigi, and D. G. Giuseppe Martino, "Development of a Digital Twin Model for Real-Time Assessment of Collision Hazards," presented at the *Proceedings of the Creative Construction e-Conference 2020*, 2020, doi:10.3311/CCC2020-003. Available: https://www.researchgate.net/publication/343604460_Development_of_a_Digital_Twin_Model_for_Real-Time_Assessment_of_Collision_Hazards. [Accessed Aug. 20, 2023].

# 6 บรรณานุกรม

Deng, M., Menassa, C. C., & Kamat, V. R. (2021). From BIM to digital twins: a systematic review of the evolution of intelligent building representations in the AEC-FM industry. Journal of Information Technology in Construction, 26, 58–83. https://doi.org/10.36680/j.itcon.2021.005EEC. (2020, September 1). วิสัยทัศน์/พันธกิจ. https://www.eeco.or.th/th/vision-mission

Digital twins: What are they and why do they matter? (2022, May 24). World Economic Forum. Retrieved June 25, 2023, from https://www.weforum.org/agenda/2022/05/digital-twin-technology-virtual-model-tech-for-good/?DAG=3&gclid=Cj0KCQjwy9-kBhCHARIsAHpBjHjhx4hmpES46UD1wAu0uNJuznuXuUKyJPRdeKeDtxiiSJQrhkelmMwaApXeEALw_wcBMadni, A. M., Madni, C. C., & Lucero, S. D. (2019). Leveraging Digital twin technology in Model-Based Systems engineering. Systems, 7(1), 7. https://doi.org/10.3390/systems7010007

Fuller, A., Fan, Z., Day, C., & Barlow, C. (2020). Digital Twin: enabling technologies, challenges and open research. IEEE Access, 8, 108952–108971. https://doi.org/10.1109/access.2020.2998358

Grieves, M., & Vickers, J. (2016). Digital Twin: mitigating unpredictable, undesirable emergent behavior in complex systems. In Springer eBooks (pp. 85–113). https://doi.org/10.1007/978-3-319-38756-7_4

Leonardo, M., Berardo, N., Alessandro, C., Luigi, R., & Martino, D. G. G. (2020). Development of a Digital Twin Model for Real-Time Assessment of Collision Hazards. Creative Construction e-Conference 2020. https://doi.org/10.3311/ccc2020-003

Liu, M., Fang, S., Dong, H., & Xu, C. (2021). Review of digital twin about concepts, technologies, and industrial applications. Journal of Manufacturing Systems, 58, 346–361. https://doi.org/10.1016/j.jmsy.2020.06.017

Lu, Y., Liu, C., Wang, K. I., Huang, H., & Xu, X. (2020). Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues. Robotics and Computer-integrated Manufacturing, 61, 101837. https://doi.org/10.1016/j.rcim.2019.101837Wukkadada, B., Wankhede, K., Nambiar, R., & Nair, A. (2018). Comparison with HTTP and MQTT In Internet

of Things (IoT). 2018 International Conference on Inventive Research in Computing Applications (ICIRCA). https://doi.org/10.1109/icirca.2018.8597401

Madni, A. M., Madni, C. C., & Lucero, S. D. (2019). Leveraging Digital twin technology in Model-Based Systems engineering. Systems, 7(1), 7. https://doi.org/10.3390/systems7010007

Moyne, J., Qamsane, Y., Balta, E. C., Kovalenko, I., Faris, J., Barton, K., & Tilbury, D. M. (2020). A requirements driven Digital twin Framework: specification and opportunities. IEEE Access, 8, 107781–107801. https://doi.org/10.1109/access.2020.3000437

PLCNext community website. (2024, April 18). PLCnext Community. Retrieved May 9, 2024, from https://www.plcnext-community.net/

Raza, M., Kumar, P. M., Hung, D. V., Davis, W., Nguyen, H., & Trestian, R. (2020). A Digital Twin Framework for Industry 4.0 Enabling Next-Gen Manufacturing. 2020 9th International Conference on Industrial Technology and Management (ICITM). https://doi.org/10.1109/icitm48982.2020.9080395

What is a digital twin? | IBM. (2023, June 25). Retrieved August 13, 2023, from https://www.ibm.com/topics/what-is-a-digital-twin

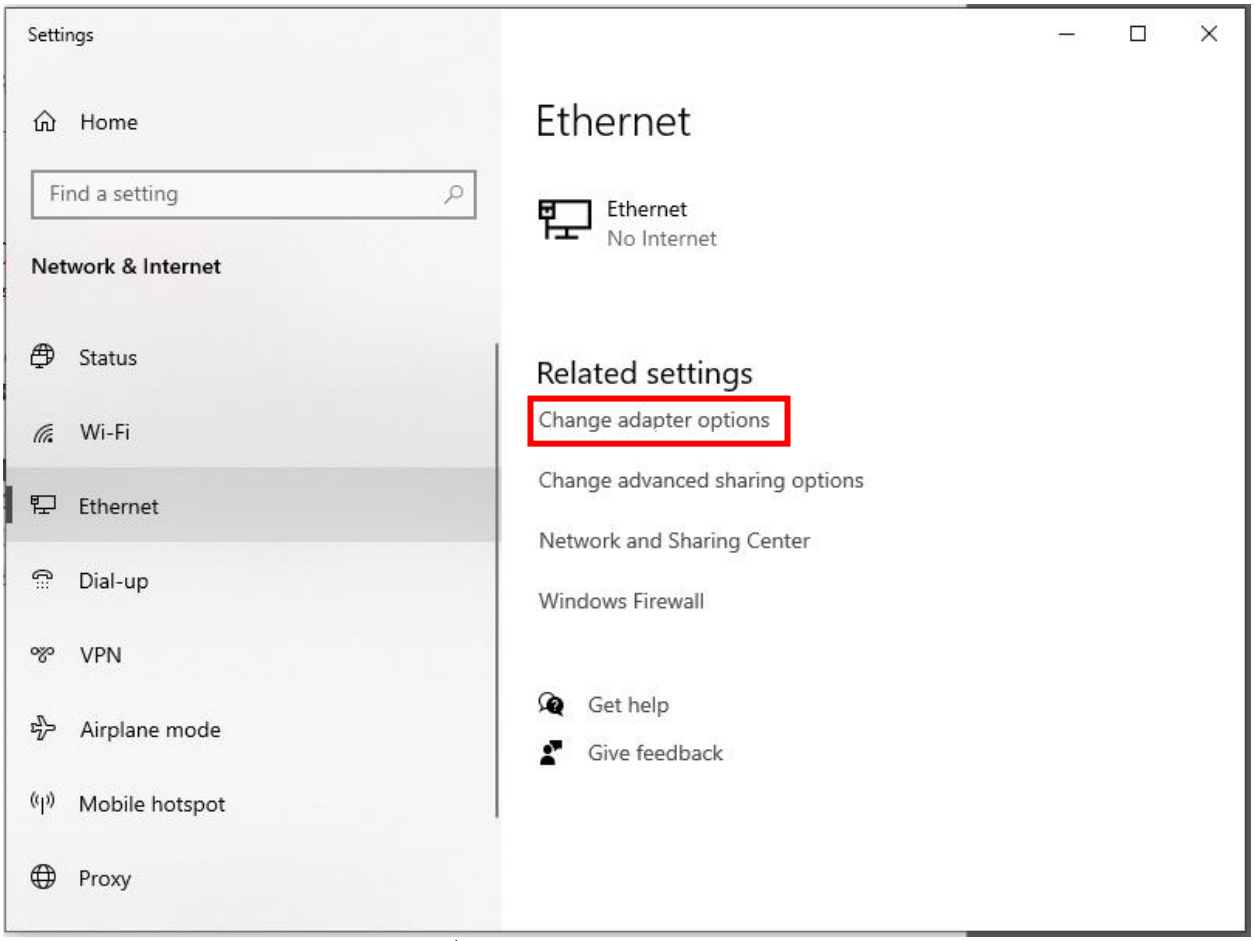
Wukkadada, B., Wankhede, K., Nambiar, R., & Nair, A. (2018). Comparison with HTTP and MQTT In Internet of Things (IoT). 2018 International Conference on Inventive Research in Computing Applications (ICIRCA). https://doi.org/10.1109/icirca.2018.8597401

Zheng, X., Lu, J., & Kiritsis, D. (2021). The emergence of cognitive digital twin: vision, challenges and opportunities. International Journal of Production Research, 60(24), 7610–7632. https://doi.org/10.1080/00207543.2021.2014591

# 7 ภาคผนวก

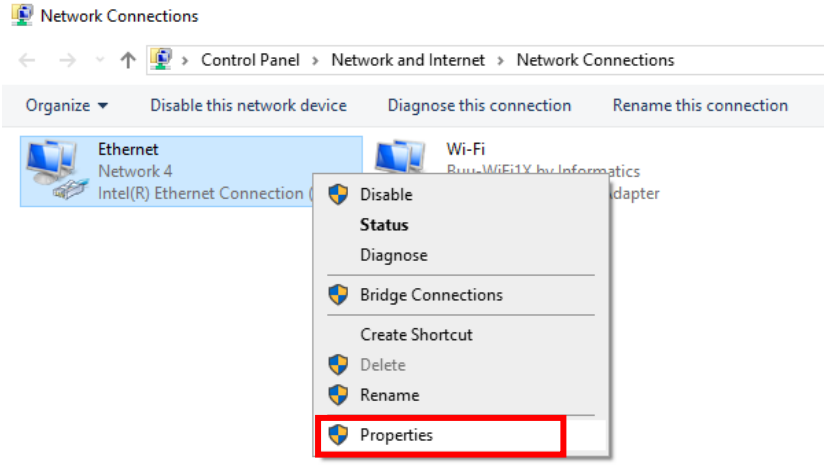
## 7.1 ขั้นตอนการเชื่อมต่อ PLCnext กับ PLCnext engineer

1. เชื่อมต่อเครื่อง PLC เข้ากับคอมพิวเตอร์ ด้วยสาย LAN
2. เปิด Network & Internet เลือกหัวข้อ Ethernet และคลิกที่ Change adapter options



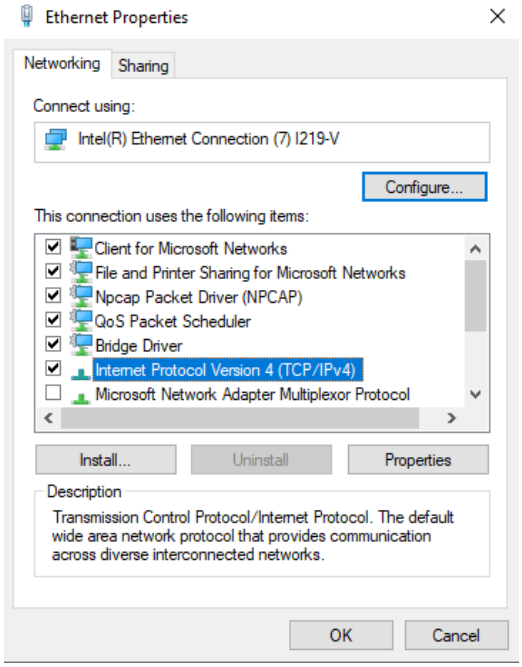

ภาพที่ 42 หน้าต่าง Network & Internet

3. คลิกขวาที่ Ethernet จากนั้นเลือก Properties



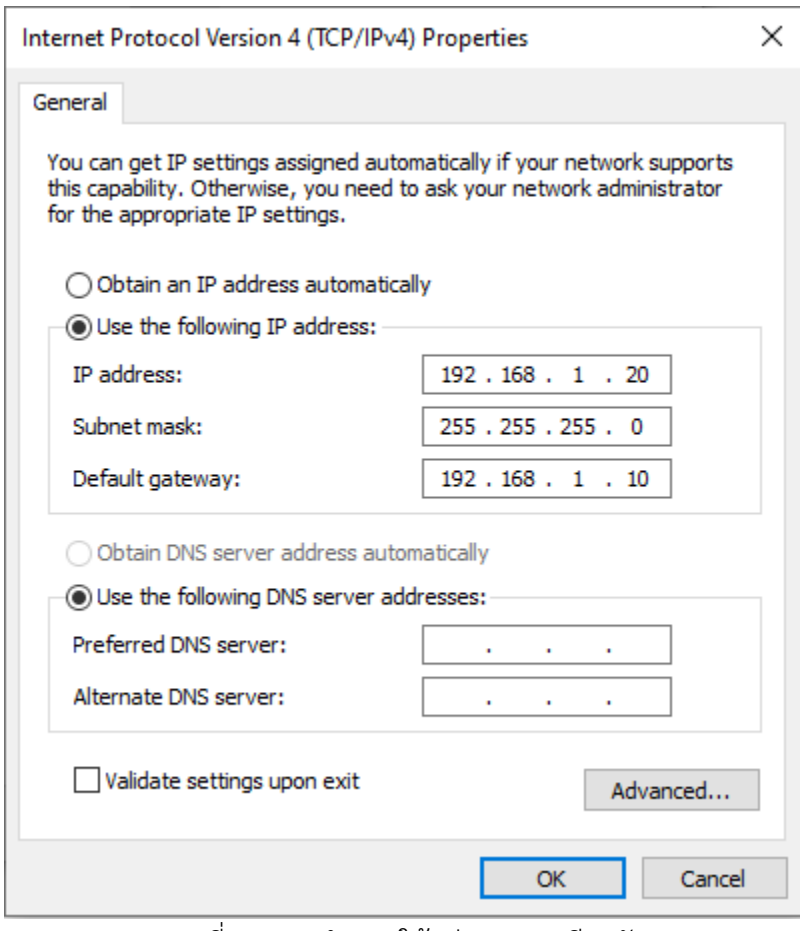

ภาพที่ 43 กด Properties ของ Ethernet

4. ดับเบิลคลิกที่ Internet Protocol Version 4 (TCP/IPv4)



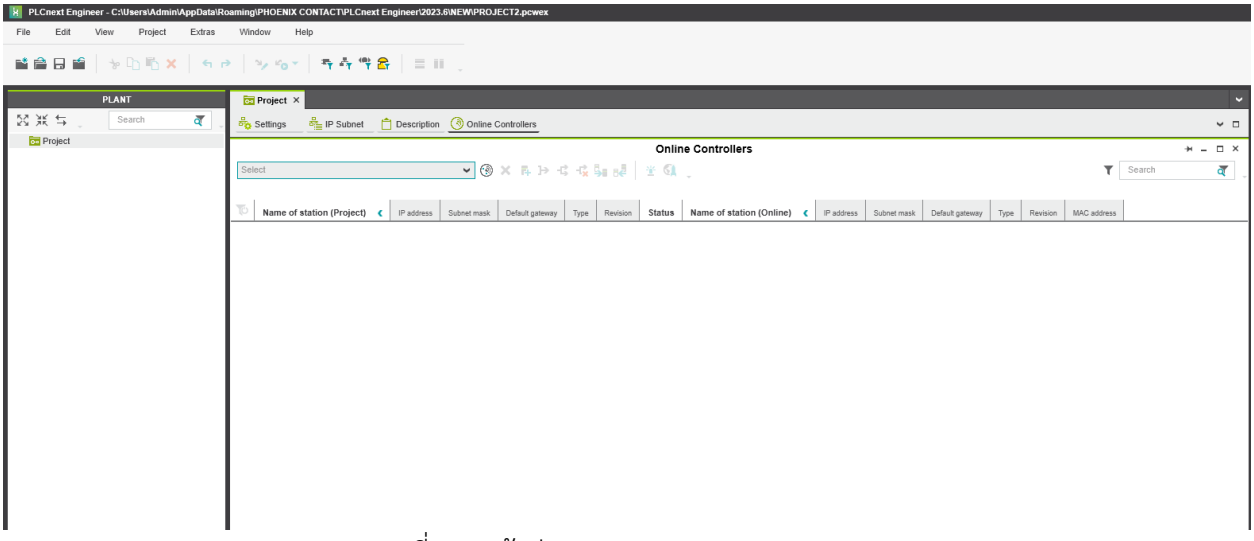

ภาพที่ 44 การเลือก IPv4

5. กำหนด IP address อยู่ในวงแลนเดียวกัน และห้ามทับกับ Default gateway ของ PLC



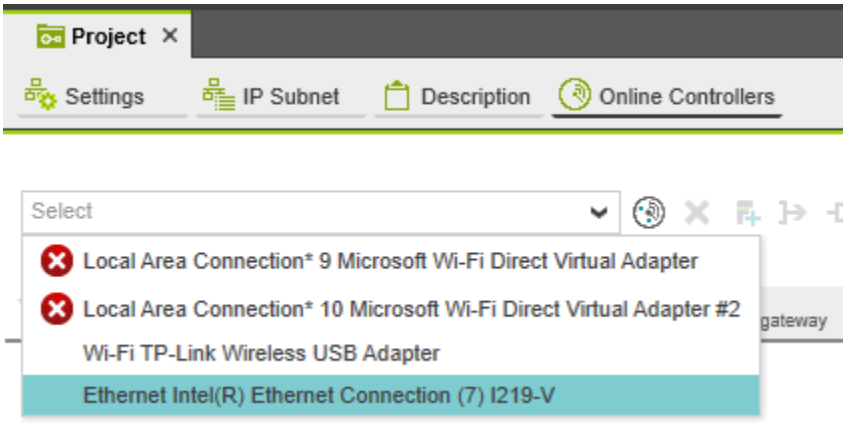

ภาพที่ 45 การกำหนดให้อยู่วง LAN เดียวกัน

6. เปิดโปรแกรม PLCnext engineer และเลือก Project ทางด้านซ้ายมือ และเลือกหัวข้อ Online Controllers



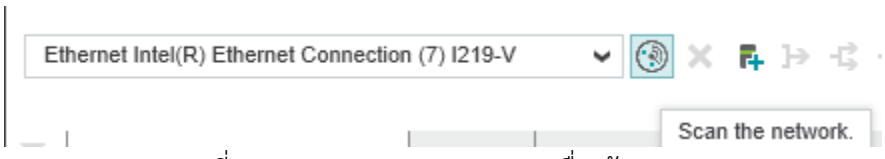

ภาพที่ 46 หน้าต่าง UI ของ PLCnext engineer

7. คลิกที่ drop down icon ให้เลือก Ethernet




ภาพที่ 47 เลือกพอร์ทที่เชื่อมต่อกับ PLC

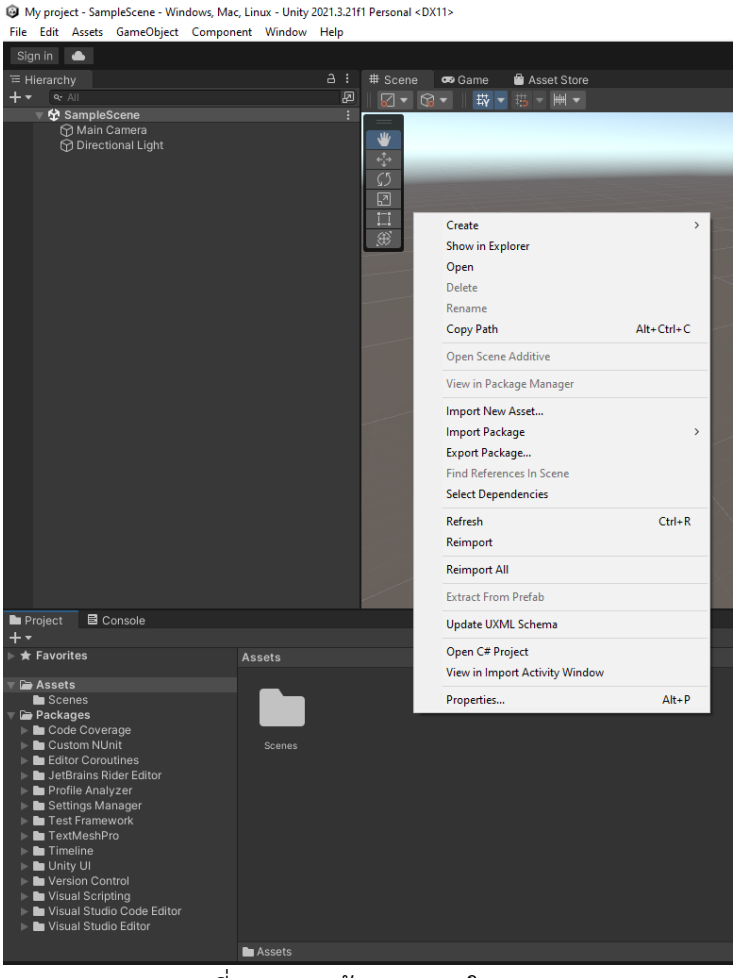
8. คลิกที่ปุ่ม Scan the network




ภาพที่ 48 การกด Scan network เพื่อ ค้นหา PLC

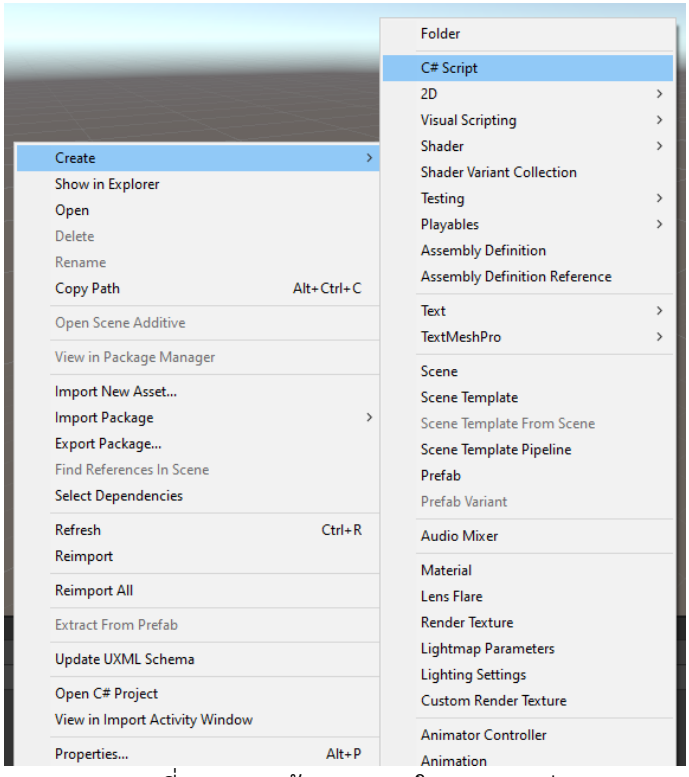
## 7.2 การสร้างและใช้งานสคริปต์ใน Unity

1. คลิกขวาที่พื้นที่ว่างในช่องของ Assets จากนั้นเลือก Create



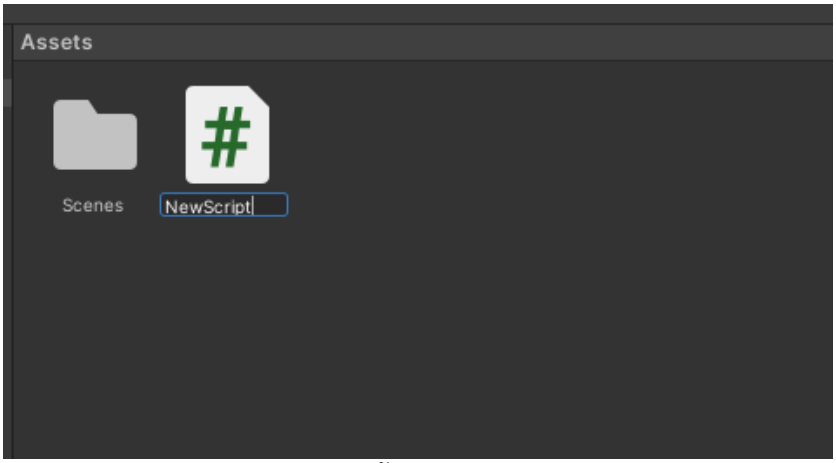

ภาพที่ 49 การสร้าง Script ใน Unity

2. หลังจากเลือก Create แล้ว ให้เลือก C# Script



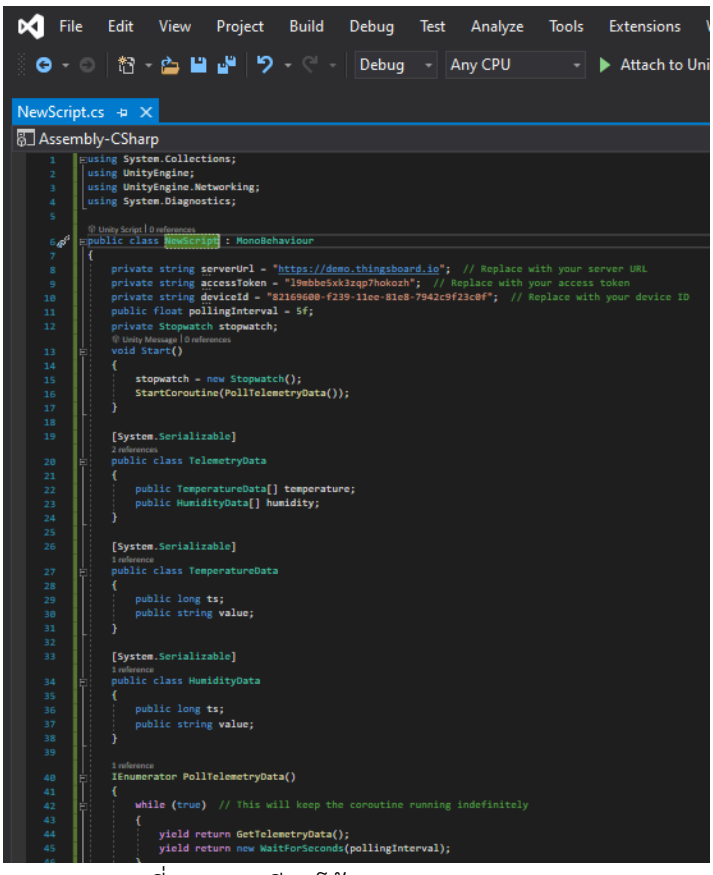

ภาพที่ 50 การสร้าง Script ใน Unity (ต่อ)

3. ตั้งชื่อสคริปต์ที่สร้างตามต้องการ



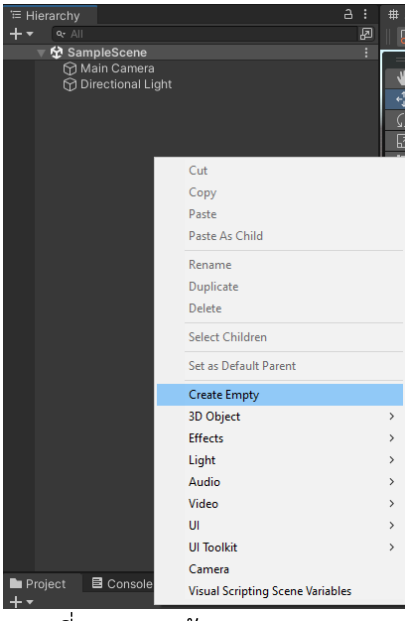

ภาพที่ 51 การตั้งชื่อ Script ใน Unity

4. ทำการ Coding ตามฟังก์ชันที่ใช้



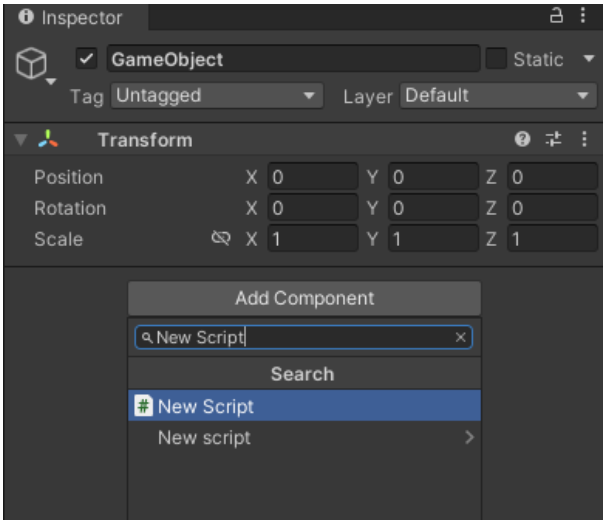

ภาพที่ 52 การเขียนโค้ดลง Script

5. คลิกขวาที่ช่อง Hierarchy เลือก Create Empty จะเป็นการสร้าง Game Object เพื่อใช้เป็นที่ใส่ Script ที่สร้างไว้ให้ใช้งานได้ใน Scene




ภาพที่ 53 การสร้าง Game Object

6. เลือก Game Object ที่สร้าง เลือก Add component ให้ใส่ Script ที่เขียนโค้ดเสร็จแล้ว




ภาพที่ 54 การใส่ Script ลงไปใน Game Object

## 7.3 ข้อมูลเพิ่มเติมสำหรับการอธิบายโค้ดที่ใช้เบื้องต้น

1. นำเข้า Library

```
using System.Collections;
using UnityEngine;
using UnityEngine.Networking;
using System.Diagnostics;
```

ภาพที่ 55 โค้ดการนำเข้า Library

2. ประกาศและแทนค่าตัวแปร

```
private string serverUrl = "https://demo.thingsboard.io";
private string accessToken = "19mbbe5xk3zqp7hokozh";
private string deviceId = "82169600-f239-11ee-81e8-7942c9f23c0f";
public float pollingInterval = 5f;
private Stopwatch stopwatch;
```

ภาพที่ 56 โค้ดการประกาศตัวแปร

3. ให้มีการดึงข้อมูลตลอดเวลาตาม interval ที่ตั้งค่าไว้

```
IEnumerator PollTelemetryData()
{
    while (true)
    {
        yield return GetTelemetryData();
        yield return new WaitForSeconds(pollingInterval);
    }
}
```

ภาพที่ 57 โค้ดกำหนดให้โค้ดรันเป็น Loop เป็นรอบวินาทีที่กำหนด

4. ทำการดึงข้อมูลจากเซิร์ฟเวอร์ โดยเริ่มจับเวลา ส่งคำขอดึงข้อมูล และหยุดจับเวลาเมื่อคำขอเสร็จสิ้น

```
IEnumerator GetTelemetryData()
{
    stopwatch.Restart();
    string url = $"{serverUrl}/api/plugins/telemetry/DEVICE/{deviceId}/values/timeseries";

    UnityWebRequest request = UnityWebRequest.Get(url);
    request.SetRequestHeader("Authorization", "Bearer " + accessToken);
    request.SetRequestHeader("Accept", "application/json");

    yield return request.SendWebRequest();
    stopwatch.Stop();

    if (request.result == UnityWebRequest.Result.Success)
    {
        string jsonResponse = request.downloadHandler.text;
        UnityEngine.Debug.Log("Telemetry Data: " + jsonResponse);
        TelemetryData telemetryData = JsonUtility.FromJson<TelemetryData>(jsonResponse);

    }
    else
    {

    }
    UnityEngine.Debug.Log("Time taken: " + stopwatch.Elapsed.TotalMilliseconds + " ms");
    yield return null;

}
```

ภาพที่ 58 ส่งคำขอดึงข้อมูลและการจับเวลา

# 8 ประวัตินักวิจัยและคณะ

## 8.1 หัวหน้าโครงการวิจัย

1. ชื่อ-นามสกุล ผศ.ดร.ประจักษ์ จิตเงินมะดัน

ชื่อ-นามสกุล Asst.Prof.Dr.techn. Prajaks Jitngernmadan

2. ตำแหน่งปัจจุบัน

อาจารย์ประจำคณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา

3. ที่อยู่ที่สามารถติดต่อได้

คณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา ต. แสนสุข อ.เมือง จ. ชลบุรี 20131

โทรศัพท์ 038-103061 โทรสาร -

E-mail: prajaks@buu.ac.th

4. ประวัติการศึกษา

| ปี | คุณวุฒิ | สถานศึกษา |
|---|---|---|
| 2560 | Doktor der technischen Wissenschaften (Doctor of Engineering Sciences, Doctor technicae) | Johannes Kepler University, Linz, Austria |
| 2550 | Master of Science in Electrical Engineering and Information Technologies | University of Applied Sciences Duesseldorf, Germany |
| 2548 | Bachelor of Science in Information Technology | University of Applied Sciences Duesseldorf, Germany |

5. สาขาที่มีความชำนาญพิเศษ

Accessibility, Usability, Human Computer Interaction, User-centered Design

6. ประสบการณ์ที่เกี่ยวข้องกับการบริหารงานวิจัย

6.1 ผู้อำนวยการแผนงานวิจัย :

6.2 หัวหน้าโครงการวิจัย :

- การออกแบบและพัฒนาระบบตรวจข้อสอบข้อเขียนแบบข้อความยาวโดยอัตโนมัติ
- การออกแบบและประยุกต์ใช้ระบบโทรศัพท์ผ่านเครือข่ายคอมพิวเตอร์โดยใช้ Asterisk เป็นพื้นฐานสำหรับคณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา

6.3 โครงการวิจัยที่ทำเสร็จแล้ว :

- การออกแบบและพัฒนาระบบตรวจข้อสอบข้อเขียนแบบข้อความยาวโดยอัตโนมัติ

- การออกแบบและประยุกต์ใช้ระบบโทรศัพท์ผ่านเครือข่ายคอมพิวเตอร์โดยใช้ Asterisk เป็นพื้นฐานสำหรับคณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา